

# Teradata Vantage™ - Data Dictionary

---

Release 17.10




July 2021

# Copyright and Trademarks

Copyright © 2000 - 2021 by Teradata. All Rights Reserved.

All copyrights and trademarks used in Teradata documentation are the property of their respective owners. For more information, see [Trademark Information](#).

## Product Safety

| Safety type   | Description  |
|---|--|
|  | Indicates a situation which, if not avoided, could result in damage to property, such as to equipment or data, but not related to personal injury. |
|  | Indicates a hazardous situation which, if not avoided, could result in minor or moderate personal injury.  |
|  | Indicates a hazardous situation which, if not avoided, could result in death or serious personal injury.   |

## Third-Party Materials

Non-Teradata (i.e., third-party) sites, documents or communications ("Third-party Materials") may be accessed or accessible (e.g., linked or posted) in or in connection with a Teradata site, document or communication. Such Third-party Materials are provided for your convenience only and do not imply any endorsement of any third party by Teradata or any endorsement of Teradata by such third party. Teradata is not responsible for the accuracy of any content contained within such Third-party Materials, which are provided on an "AS IS" basis by Teradata. Such third party is solely and directly responsible for its sites, documents and communications and any harm they may cause you or others.

## Warranty Disclaimer

**Except as may be provided in a separate written agreement with Teradata or required by applicable law, the information available from the Teradata Documentation website or contained in Teradata information products is provided on an "as-is" basis, without warranty of any kind, either express or implied, including the implied warranties of merchantability, fitness for a particular purpose, or noninfringement.**

The information available from the Teradata Documentation website or contained in Teradata information products may contain references or cross-references to features, functions, products, or services that are not announced or available in your country. Such references do not imply that Teradata Corporation intends to announce such features, functions, products, or services in your country. Please consult your local Teradata Corporation representative for those features, functions, products, or services available in your country.

The information available from the Teradata Documentation website or contained in Teradata information products may be changed or updated by Teradata at any time without notice. Teradata may also make changes in the products or services described in this information at any time without notice.

## Machine-Assisted Translation

Certain materials on this website have been translated using machine-assisted translation software/tools. Machine-assisted translations of any materials into languages other than English are intended solely as a convenience to the non-English-reading users and are not legally binding. Anybody relying on such information does so at his or her own risk. No automated translation is perfect nor is it intended to replace human translators. Teradata does not make any promises, assurances, or guarantees as to the accuracy of the machine-assisted translations provided. Teradata accepts no responsibility and shall not be liable for any damage or issues that may result from using such translations. Users are reminded to use the English contents.

## Feedback

To maintain the quality of our products and services, e-mail your comments on the accuracy, clarity, organization, and value of this document to: [docs@teradata.com](mailto:docs@teradata.com).

Any comments or materials (collectively referred to as "Feedback") sent to Teradata Corporation will be deemed nonconfidential. Without any payment or other obligation of any kind and without any restriction of any kind, Teradata and its affiliates are hereby free to (1) reproduce, distribute, provide access to, publish, transmit, publicly display, publicly perform, and create derivative works of, the Feedback, (2) use any ideas, concepts, know-how, and techniques contained in such Feedback for any purpose whatsoever, including developing, manufacturing, and marketing products and services incorporating the Feedback, and (3) authorize others to do any or all of the above.

# Contents

|   |           |
|---|-----------|
| <b>Chapter 1: Introduction to Data Dictionary</b> | <b>8</b>  |
| Changes and Additions                             | 8         |
| <b>Chapter 2: Overview</b>                        | <b>9</b>  |
| Installing Data Dictionary                        | 9         |
| What the Data Dictionary Stores                   | 9         |
| <b>Chapter 3: Data Dictionary Views</b>           | <b>11</b> |
| About Data Dictionary Views                       | 11        |
| Types of Views                                    | 11        |
| Access to Data Dictionary Views                   | 17        |
| Trusted Sessions Support in X Views               | 17        |
| Querying Data Dictionary Views                    | 19        |
| Tracking Resource Usage                           | 20        |
| Tracking User Activity                            | 24        |
| Using System Views                                | 26        |
| Using Unicode Views to Update Object Names        | 34        |
| Teradata QueryGrid                                | 34        |
| TDMaps  | 35        |
| <b>Chapter 4: Views Reference</b>                 | <b>36</b> |
| Overview  | 36        |
| AccessLogV  | 38        |
| AccLogRulesV                                      | 42        |
| AccountInfoV[X]                                   | 51        |
| ActionHistoryV[X]                                 | 53        |
| ActionsV[X]                                       | 56        |
| All_RI_ChildrenV[X]                               | 61        |
| All_RI_ParentsV[X]                                | 63        |
| AllGlobalSpaceV[X]                                | 65        |
| AllRightsV[X]                                     | 67        |
| AllRoleRightsV                                    | 69        |
| AllSpaceV[X]                                      | 70        |
| AllTempTablesV[X]                                 | 74        |
| AMPUsageV[X]                                      | 74        |
| ArchiveLoggingObjsV[X]                            | 76        |
| AssociationV[X]                                   | 77        |
| AuthorizationsV[X]                                | 80        |
| BusinessCalendar                                  | 82        |

|                               |     |
|-------------------------------|-----|
| BusinessCalendarExceptions    | 86  |
| BusinessCalendarPatterns      | 87  |
| CDSTableSizeV                 | 88  |
| CharSetsV                     | 89  |
| CharTranslationsV             | 89  |
| ChildrenV[X]                  | 91  |
| CollationsV                   | 92  |
| ColumnStatsV                  | 93  |
| ColumnsV[X]                   | 97  |
| ColumnUseCountV[X]            | 111 |
| ConnectRulesV                 | 112 |
| ConstraintFunctionsV          | 115 |
| ConstraintValuesV             | 115 |
| CostProfiles_v                | 116 |
| CostProfileTypes_v            | 116 |
| CostProfileValues_v           | 116 |
| Database_Default_JournalsV[X] | 117 |
| Databases2V[X]                | 118 |
| DatabasesV[X]                 | 119 |
| DatabaseUseCountV[X]          | 122 |
| DatasetSchemaDependenciesV    | 123 |
| DatasetSchemaInfoV            | 123 |
| DBCInfoV                      | 124 |
| DBQLRulesV                    | 125 |
| DeleteAccessLogV              | 132 |
| DeleteOldInDoubtV             | 133 |
| DeleteUseCountV[X]            | 134 |
| DiskGlobalSpaceErrorV         | 135 |
| DiskSpaceErrorV               | 136 |
| DiskSpaceV[X]                 | 137 |
| ErrorTblsV[X]                 | 143 |
| Events_ConfigurationV[X]      | 145 |
| Events_MediaV[X]              | 147 |
| EventsV[X]                    | 148 |
| ExclusionListsV[X]            | 153 |
| ExportWidthV                  | 154 |
| ExpStatsV                     | 156 |
| ForeignTablesInfoV[X]         | 159 |
| ForeignTablesV[X]             | 160 |
| ExternalSPsV[X]               | 160 |
| FunctionAliasInfoV[X]         | 164 |
| FunctionAliasV[X]             | 165 |
| FunctionsV[X]                 | 165 |
| GlobalDBSpaceV[X]             | 170 |
| HostsInfoV                    | 175 |

|                               |     |
|-------------------------------|-----|
| IndexConstraintsV[X]          | 176 |
| IndexStatsV[X]                | 179 |
| IndexUseCountV[X]             | 183 |
| IndicesV[X]                   | 185 |
| InDoubtLogV                   | 192 |
| InsertUseCountV[X]            | 193 |
| JoinIndicesV                  | 194 |
| JournalsV[X]                  | 197 |
| LoadTablesInfoV[X]            | 198 |
| LogOnOffV[X]                  | 199 |
| LogonRulesV                   | 210 |
| MapGrantsV[X]                 | 211 |
| MapListsV[X]                  | 212 |
| MapsV[X]                      | 213 |
| MultiColumnStatsV             | 216 |
| MultiExpStatsV                | 219 |
| ObjectsInSparseMapsV          | 222 |
| ObjectListsV[X]               | 223 |
| ObjectUseCountV[X]            | 224 |
| PartitioningConstraintsV[X]   | 225 |
| PeriodsV[X]                   | 230 |
| ProfileAsgdSecConstraintsV[X] | 232 |
| ProfileInfoV[X]               | 233 |
| QryLockLogXMLV                | 236 |
| QryLogAmpDataV                | 238 |
| QryLogClientAttrV             | 240 |
| QryLogEventHisV               | 252 |
| QryLogEventsV                 | 256 |
| QryLogExceptionsV             | 259 |
| QryLogExplainDocV             | 262 |
| QryLogExplainV                | 264 |
| QryLogFeatureListV            | 265 |
| QRYLOGFEATUREUSECOUNTV        | 268 |
| QryLogFeatureUseJSON          | 270 |
| QryLogObjectsV                | 272 |
| QryLogParamJSON               | 275 |
| QryLogParamV                  | 278 |
| QryLogSQLDocV                 | 280 |
| QryLogSQLV                    | 282 |
| QryLogStepsV                  | 284 |
| QryLogSummaryV                | 300 |
| QryLogTdwmsumV                | 307 |
| QryLogTDWMV                   | 313 |
| QryLogUtilityV                | 317 |
| QryLogV                       | 329 |

|                            |     |
|----------------------------|-----|
| QRYLogXMLDocV              | 347 |
| QryLogXMLV                 | 350 |
| RCC_ConfigurationV[X]      | 354 |
| RCC_MediaV[X]              | 355 |
| ReconfigDeleteOrderV       | 356 |
| ReconfigInfoV              | 358 |
| ReconfigRedistOrderV       | 362 |
| ReconfigTableStatsV        | 363 |
| RepCaptureRulesV           | 367 |
| RepTablesV[X]              | 368 |
| ResolvedDTSV[X]            | 368 |
| RestrictedWordsV           | 370 |
| RI_Child_TablesV[X]        | 370 |
| RI_Distinct_ChildrenV[X]   | 373 |
| RI_Distinct_ParentsV[X]    | 374 |
| RI_Parent_TablesV[X]       | 376 |
| RoleInfoV[X]               | 379 |
| RoleMembersV[X]            | 380 |
| SecConstraintsV[X]         | 381 |
| SecurityDefaultsV          | 382 |
| SecurityLogV[X]            | 383 |
| ServerInfoV[X]             | 385 |
| ServerV[X]                 | 386 |
| SessionInfoV[X]            | 388 |
| SettingsV                  | 401 |
| SHOWCOLCHECKSV[X]          | 402 |
| SHOWTBLCHECKSV[X]          | 404 |
| Software_Event_LogV        | 406 |
| SparseMapAmpsV             | 408 |
| StatsV                     | 409 |
| StatUseCountV[X]           | 413 |
| Table_LevelConstraintsV[X] | 414 |
| Tables2V[X]                | 415 |
| Tables3V[X]                | 416 |
| TableSizeV[X]              | 417 |
| TableStatsV                | 419 |
| TablesV[X]                 | 423 |
| TableTextV[X]              | 435 |
| TableToSparseMapSizingV[X] | 436 |
| TblSrvInfoV[X]             | 438 |
| TblSrvV[X]                 | 439 |
| TempTableStatsV            | 440 |
| TriggersV[X]               | 443 |
| UDTInfoV                   | 447 |
| UDTTransformV              | 449 |

|   |            |
|---|------------|
| UpdateUseCountV[X] . . . . .  | 451        |
| User_Default_JournalsV[X] . . . . .                                 | 452        |
| UserGrantedRightsV . . . . .  | 453        |
| UserRightsV . . . . .   | 454        |
| UserRoleRightsV . . . . .   | 456        |
| UsersV . . . . .  | 457        |
| UsrAsgdSecConstraintsV[X] . . . . .                                 | 462        |
| ZoneGuestsV[X] . . . . .  | 463        |
| ZonesV[X] . . . . .   | 463        |
| <b>Chapter 5: Data Dictionary Tables . . . . .</b>                  | <b>465</b> |
| How Tables Are Created . . . . .                                    | 465        |
| Accessing Tables . . . . .  | 465        |
| Nonhashed Tables . . . . .  | 466        |
| DBCExtension Tables . . . . .                                       | 467        |
| Updating Tables . . . . .   | 467        |
| Character Data . . . . .  | 467        |
| Maintaining System Logs . . . . .                                   | 468        |
| <b>Appendix A: View Column Values . . . . .</b>                     | <b>471</b> |
| <b>Appendix B: LogonSource Column Fields and Examples . . . . .</b> | <b>479</b> |
| <b>Appendix C: Database Objects . . . . .</b>                       | <b>495</b> |
| <b>Appendix D: Additional Information . . . . .</b>                 | <b>506</b> |

# Introduction to Data Dictionary

Teradata Vantage™ is our flagship analytic platform offering, which evolved from our industry-leading Teradata® Database. Until references in content are updated to reflect this change, the term Teradata Database is synonymous with Teradata Vantage.

This document contains information about Data Dictionary system views and columns. You can use Teradata Studio or Teradata Studio Express to list the Data Dictionary system views and view details about each column. For information about these tools and to download them:

1. Go to <https://support.teradata.com>.
2. Log in.

Data Dictionary tables only change in major Advanced SQL Engine releases, but Data Dictionary views can change in minor releases.

## Changes and Additions

| Date      | Description  |
|-----------|--|
| July 2021 | 17.10 includes following the following change(s): <ul style="list-style-type: none"><li>• LogOnOffV, QryLogClientAttrV, and SessionInfoV have additional columns for TLS 1.2</li></ul> |
| June 2020 | <ul style="list-style-type: none"><li>• New MergeInto value for StatementGroup field of QryLogV. See <a href="#">Usage Notes</a>.</li></ul>  |

## Overview

The Teradata Vantage Data Dictionary is composed of tables and views primarily in database DBC. The dictionary also includes tables and views from in a number of other system databases (for example, Sys\_Calendar, TD\_SYSFNLB, SYSUDTLIB, SystemFE, and SYSUIF), and from other databases (for example, TDMaps and SQLJ).

Dictionary tables contain metadata about system objects, privileges, events, and usage. These tables are reserved for system use, and should not be accessed directly. Users can retrieve data from tables indirectly, by accessing predefined Data Dictionary views. The database administrator can limit user access to dictionary views on a user-by-user or role-by-role basis.

## Installing Data Dictionary

The system databases, tables and associated views and macros are created at system initialization (sysinit) time and by executing a set of Dictionary Initialization Program (DIP) scripts. The DIPALL option executes all of the DIP scripts that are installed on every system.

Optional DIP scripts include:

- DIPACC (supports database access logging)
- DIPPCR (supports infrastructure used to analyze system performance issues)

## Related Topics

| For information about ...  | See ...   |
|--|---|
| the DIP utility and its executable SQL scripts (such as DIPPCR, DIPACC, DIPSYSUIF, DIPVIEWS, and DIPALL)             | <i>Teradata Vantage™ - Database Utilities</i> , B035-1102.                          |
| the macros that are created by the DIPVIEWS script   | <i>Teradata Vantage™ - Database Administration</i> , B035-1093.                     |
| using the DIPACC script to create the DBC.ACCLogRule macro, which is required for setting up database access logging | <i>Teradata Vantage™ - Advanced SQL Engine Security Administration</i> , B035-1100. |

## What the Data Dictionary Stores

Information about each database object is stored in the system tables.

System tables cannot be directly modified and are used by the system to manage and maintain the integrity of the database.

Information in the system tables is used to create, access, modify and execute objects and user data stored in Vantage.

The primary system tables include:

- **Dbase:** This table contains information about every database installed on the system. Database information includes names associated with this database (for example, database name, owner name, and account name), timestamps, passwords, and so on.
- **DataBaseSpace:** This table contains space allocation for each database.
- **TVM and TVFields:** These two tables contain information about every table, view, macro and other objects that are stored in the databases.
- **Accessrights:** This table contains all of the information about user permissions for each type object stored in the database.

Vantage supports the following database objects:

- **Stored Procedures** written in SQL and external procedures written in C/C++ and Java.
- **User-defined types (UDTs), user-defined functions (UDFs), and user-defined methods (UDMs).** These functions and methods provide you with the toolset to perform whatever type of processing and manipulation of data is required.

For detailed descriptions of these objects, see [Database Objects](#).

## Other System Objects

For details about system objects not covered in this document, see [Database Objects](#).

| For information about ...             | See ...  |
|---------------------------------------|--|
| SystemFE database                     | <i>Teradata Vantage™ - SystemFE Macros</i> , B035-1103.                  |
| Resource Usage tables                 | <i>Teradata Vantage™ - Resource Usage Macros and Tables</i> , B035-1099. |
| Sys_Calendar database and DBQL tables | <i>Teradata Vantage™ - Database Administration</i> , B035-1093.          |

# Data Dictionary Views

Data Dictionary tables only change in major SQL Engine releases, but Data Dictionary views can change in minor releases.

## About Data Dictionary Views

During initial system setup, the Database Initialization Program (DIP) creates pre-defined Data Dictionary views, which allow convenient access to data stored in Data Dictionary base tables. For more information about DIP, see *Teradata Vantage™ - Database Utilities*, B035-1102.

The DIP script run as part of system setup creates several types of Data Dictionary views with varying purposes. For more information about the Data Dictionary views, see [Types of Views](#).

Many Data Dictionary views are accessible by all users by default, but some are restricted. For information about the privileges for Data Dictionary views, see [Access to Data Dictionary Views](#).

You can query views using SQL requests. For details, see [Querying Data Dictionary Views](#).

Data Dictionary views are used by administrators and by administrative tools, such as Teradata Viewpoint. For more about how Data Dictionary views are used, see [Tracking Resource Usage](#).

## Types of Views

During system installation, the database administrator can load the following views into the DBC user space.

### Unicode Views

Because data dictionary tables can change, views provide greater stability.

In Unicode views, object names are returned in the session character set. The substitution character for the session character set is returned for any characters in the object name that do not exist in the session character set.

Unicode view names use the following formats:

- `view_name V`
- `view_name VX`

For example, `All_RI_ChildrenV` or `All_RI_ChildrenVX`, where the “X” denotes a partial view that omits certain restricted data, such as security-related information.

## Related Topics

| For more information about ...   | See ...  |
|--|--|
| Unicode views that end with a suffix "VX"                              | <a href="#">X Views.</a>   |
| Unicode views that end with a suffix "V"                               | <a href="#">Non-X Views.</a>   |
| standard language support systems or Japanese language support systems | <i>Teradata Vantage™ - Advanced SQL Engine International Character Set Support, B035-1125.</i> |
| the types of privileges  | <a href="#">Access to Data Dictionary Views.</a>   |

## Compatibility Views

Compatibility views convert the native variable length Unicode object names into 30 bytes of either Latin or Kanji1. This can cause loss of information by truncation or inability to convert object names longer than 30 characters into Latin or Kanji1. Characters that cannot be converted are replaced by the substitution character, which is 0x1A for both Latin and Kanji1.

As with all character data, when object names are returned to the user they are converted to the session character set. This conversion can produce loss of information if the characters in the object name cannot be converted to the session character set or exceed the export width for the character data.

Compatibility view names use the format:

- `view_name`
- `view_nameX`

### Note:

Some non compatibility views also use this format and are not deprecated, such as BusinessCalendar.

For example, All\_RI\_Children or All\_RI\_ChildrenX, where the X denotes a partial view that omits restricted data, such as security-related information.

A Unicode view exists for every compatibility view in database DBC. The information available in compatibility views is available from the corresponding Unicode view, and compatibility views should no longer be used to obtain the required information.

### Note:

Teradata strongly encourages you to switch over to Unicode views. The following compatibility views and compatibility X views are deprecated.

In the following list, views ending in "[X]" indicate there is both a compatibility view and a compatibility X view; for example, DBC.AccountInfo and DBC.AccountInfoX are shown as DBC.AccountInfo[X].

|  |   |  |
|--|---|--|
| <ul style="list-style-type: none"> <li>• DBC.AccessLog</li> <li>• DBC.AccLogRules</li> <li>• DBC.AccountInfo[X]</li> <li>• DBC.AllRights[X]</li> <li>• DBC.AllRoleRights</li> <li>• DBC.AllSpace[X]</li> <li>• DBC.AllTempTables[X]</li> <li>• DBC.All_RI_Children[X]</li> <li>• DBC.All_RI_Parents[X]</li> <li>• DBC.AMPUsage[X]</li> <li>• DBC.Association[X]</li> <li>• DBC.Authorizations[X]</li> <li>• DBC.CharSets</li> <li>• DBC.CharTranslations</li> <li>• DBC.Children[X]</li> <li>• DBC.Collations</li> <li>• DBC.Columns[X]</li> <li>• DBC.ConnectRules</li> <li>• DBC.CSPSESSIONINFO</li> <li>• DBC.Databases[X]</li> <li>• DBC.Databases2[X]</li> <li>• DBC.Database_Default_Journals[X]</li> <li>• DBC.DBCInfo</li> <li>• DBC.DBQLRules</li> <li>• DBC.DeleteAccessLog</li> <li>• DBC.DeleteOldInDoubt</li> <li>• DBC.DiskSpace[X]</li> <li>• DBC.Events[X]</li> <li>• DBC.Events_Configuration[X]</li> </ul> | <ul style="list-style-type: none"> <li>• DBC.Events_Media[X]</li> <li>• DBC.ExternalSPs[X]</li> <li>• DBC.Functions[X]</li> <li>• DBC.HostsInfo</li> <li>• DBC.IndexConstraints[X]</li> <li>• DBC.Indices[X]</li> <li>• DBC.InDoubtLog</li> <li>• DBC.Journals[X]</li> <li>• DBC.LogOnOff[X]</li> <li>• DBC.LogonRules</li> <li>• DBC.ownerdb</li> <li>• DBC.ProfileInfo[X]</li> <li>• DBC.QryLog[X]</li> <li>• DBC.QryLogEventHis</li> <li>• DBC.QryLogEvents</li> <li>• DBC.QryLogExceptions</li> <li>• DBC.QryLogExplain</li> <li>• DBC.QryLogObjects</li> <li>• DBC.QryLogParam</li> <li>• DBC.QryLogSQL</li> <li>• DBC.QryLogSteps</li> <li>• DBC.QryLogSummary</li> <li>• DBC.QryLogTDWM</li> <li>• DBC.QryLogTDWMSum</li> <li>• DBC.RCC_Configuration[X]</li> <li>• DBC.RCC_Media[X]</li> <li>• DBC.RepTables[X]</li> <li>• DBC.ResolvedDTS[X]</li> <li>• DBC.RestrictedWords</li> </ul> | <ul style="list-style-type: none"> <li>• DBC.RI_Child_Tables[X]</li> <li>• DBC.RI_Distinct_Children[X]</li> <li>• DBC.RI_Distinct_Parents[X]</li> <li>• DBC.RI_Parent_Tables[X]</li> <li>• DBC.RoleInfo[X]</li> <li>• DBC.RoleMembers[X]</li> <li>• DBC.SecurityDefaults</li> <li>• DBC.SecurityLog[X]</li> <li>• DBC.SessionInfo[X]</li> <li>• DBC.SHOWCOLCHECKS[X]</li> <li>• DBC.SHOWTBLCHECKS[X]</li> <li>• DBC.Software_Event_Log</li> <li>• DBC.Table_LevelConstraints[X]</li> <li>• DBC.Tables[X]</li> <li>• DBC.Tables2[X]</li> <li>• DBC.TableSize[X]</li> <li>• DBC.TableText[X]</li> <li>• DBC.Triggers[X]</li> <li>• DBC.UserGrantedRights</li> <li>• DBC.UserRights</li> <li>• DBC.UserRoleRights</li> <li>• DBC.Users</li> <li>• DBC.User_Default_Journals[X]</li> </ul> |
|--|---|--|

## X Views

The Unicode X views and compatibility X views contain security constraints in their definition to limit the result set to only the rows associated with the requesting user, such as:

- Databases
- Users
- Objects owned or created by the user
- Objects on which the user has been granted privileges

The following Unicode X views and compatibility X views also return rows associated with the current role of the user and any nested roles of that current role.

|  |   |   |
|--|---|---|
| <ul style="list-style-type: none"> <li>• DBC.AccessLogV</li> <li>• DBC.AccLogRulesV</li> <li>• DBC.AccountInfoV[X]</li> <li>• DBC.AllRightsV[X]</li> <li>• DBC.AllRoleRightsV</li> <li>• DBC.AllSpaceV[X]</li> <li>• DBC.AllTempTablesV[X]</li> <li>• DBC.All_RI_ChildrenV[X]</li> <li>• DBC.All_RI_ParentsV[X]</li> <li>• DBC.AMPUsageV[X]</li> </ul> | <ul style="list-style-type: none"> <li>• DBC.ExportWidthV</li> <li>• DBC.ExpStatsV</li> <li>• DBC.ExternalSPsV[X]</li> <li>• DBC.ForeignTablesInfoV[X]</li> <li>• DBC.ForeignTablesV[X]</li> <li>• DBC.FunctionAliasInfoV[X]</li> <li>• DBC.FunctionAliasV[X]</li> <li>• DBC.FunctionsV[X]</li> <li>• DBC.HostsInfoV</li> <li>• DBC.IndexConstraintsV[X]</li> </ul> | <ul style="list-style-type: none"> <li>• DBC.QueryStatsV</li> <li>• DBC.RCC_ConfigurationV[X]</li> <li>• DBC.RCC_MediaV[X]</li> <li>• DBC.ReconfigDeleteOrderV</li> <li>• DBC.ReconfigInfoV</li> <li>• DBC.ReconfigRedistOrderV</li> <li>• DBC.ReconfigTableStatsV</li> <li>• DBC.RepCaptureRulesV</li> <li>• DBC.RepTablesV[X]</li> <li>• DBC.ResolvedDTSV[X]</li> </ul> |
|--|---|---|

|   |  |  |
|---|--|--|
| <ul style="list-style-type: none"> <li>• DBC.ArchiveLoggingObjsV[X]</li> <li>• DBC.ARC_SessionInfoVX</li> <li>• DBC.AssociationV[X]</li> <li>• DBC.AuthorizationsV[X]</li> <li>• DBC.BAR_ChildrenV</li> <li>• DBC.BAR_Databases2V</li> <li>• DBC.BAR_FunctionsV</li> <li>• DBC.BAR_JoinIndicesV</li> <li>• DBC.BAR_ownerdbV</li> <li>• DBC.BAR_RI_Distinct_ParentsV</li> <li>• DBC.BAR_StatsV</li> <li>• DBC.BAR_Tables2V</li> <li>• DBC.BAR_TablesV</li> <li>• DBC.CharsetsV</li> <li>• DBC.CharTranslationsV</li> <li>• DBC.ChildrenV[X]</li> <li>• DBC.CollationsV</li> <li>• DBC.ColumnStatsV</li> <li>• DBC.ColumnsV[X]</li> <li>• DBC.ColumnsqV[X]</li> <li>• DBC.ColumnsjqV[X]</li> <li>• DBC.ColumnsUseCountV[X]</li> <li>• DBC.ConnectRulesV</li> <li>• DBC.ConstraintFunctionsV</li> <li>• DBC.ConstraintValuesV</li> <li>• DBC.CSPSESSIONINFOV</li> <li>• DBC.Databases2V[X]</li> <li>• DBC.DatabasesV[X]</li> <li>• DBC.DatabaseUseCountV[X]</li> <li>• DBC.Database_Default_JournalsV[X]</li> <li>• DBC.DBCInfoV</li> <li>• DBC.DBQLRulesV</li> <li>• DBC.DeleteAccessLogV</li> <li>• DBC.DeleteOldInDoubtV</li> <li>• DBC.DeleteUseCountV [X]</li> <li>• DBC.DiskSpaceErrorV</li> <li>• DBC.DiskSpaceV[X]</li> <li>• DBC.ErrorTblsV[X]</li> <li>• DBC.EventsV[X]</li> <li>• DBC.Events_ConfigurationV[X]</li> <li>• DBC.Events_MediaV[X]</li> </ul> | <ul style="list-style-type: none"> <li>• DBC.IndexStatsV</li> <li>• DBC.IndexUseCountV[X]</li> <li>• DBC.IndicesV[X]</li> <li>• DBC.InDoubtLogV</li> <li>• DBC.InsertUseCountV[X]</li> <li>• DBC.JoinIndicesV</li> <li>• DBC.JournalsV[X]</li> <li>• DBC.LoadTablesInfoV[X]</li> <li>• DBC.LogOnOffV[X]</li> <li>• DBC.LogonRulesV</li> <li>• DBC.MapGrantsV[X]</li> <li>• DBC.MapsV[X]</li> <li>• DBC.MultiColumnStatsV</li> <li>• DBC.MultiExpStatsV</li> <li>• DBC.ObjectUseCountV[X]</li> <li>• DBC.ownerdbV</li> <li>• DBC.PartitioningConstraintsV[X]</li> <li>• DBC.PeriodsV[X]</li> <li>• DBC.ProfileAsgdSecConstraintsV[X]</li> <li>• DBC.ProfileInfoV[X]</li> <li>• DBC.QryLockLogXMLV</li> <li>• DBC.QryLogEventHisV</li> <li>• DBC.QryLogEventsV</li> <li>• DBC.QryLogExceptionsV</li> <li>• DBC.QryLogExplainDocV</li> <li>• DBC.QryLogExplainV</li> <li>• DBC.QryLogFeatureUseCountV</li> <li>• DBC.QryLogFeatureListV</li> <li>• DBC.QryLogObjectsV</li> <li>• DBC.QryLogParamV</li> <li>• DBC.QryLogSQLDocV</li> <li>• DBC.QryLogSQLV</li> <li>• DBC.QryLogStepsV</li> <li>• DBC.QryLogSummaryV</li> <li>• DBC.QryLogTDWMV</li> <li>• DBC.QryLogTdwSumV</li> <li>• DBC.QryLogUtilityV</li> <li>• DBC.QryLogV</li> <li>• DBC.QRYLogXMLDocV</li> <li>• DBC.QryLogXMLV</li> </ul> | <ul style="list-style-type: none"> <li>• DBC.RestrictedWordsV</li> <li>• DBC.RI_Child_TablesV[X]</li> <li>• DBC.RI_Distinct_ChildrenV[X]</li> <li>• DBC.RI_Distinct_ParentsV[X]</li> <li>• DBC.RI_Parent_TablesV[X]</li> <li>• DBC.RoleInfoV[X]</li> <li>• DBC.RoleMembersV[X]</li> <li>• DBC.SecConstraintsV[X]</li> <li>• DBC.SecurityDefaultsV</li> <li>• DBC.SecurityLogV[X]</li> <li>• DBC.ServerV[X]</li> <li>• DBC.SessionInfoV[X]</li> <li>• DBC.SHOWCOLCHECKSV[X]</li> <li>• DBC.SHOWTBLCHECKSV[X]</li> <li>• DBC.Software_Event_LogV</li> <li>• DBC.StatsV</li> <li>• DBC.StatUseCountV[X]</li> <li>• DBC.Tables2V[X]</li> <li>• DBC.Tables3VX</li> <li>• DBC.TableSizeV[X]</li> <li>• DBC.TableStatsV</li> <li>• DBC.TablesV[X]</li> <li>• DBC.TableTextV[X]</li> <li>• DBC.Table_LevelConstraintsV[X]</li> <li>• DBC.TblSrvInfoV[X]</li> <li>• DBC.TblSrvV[X]</li> <li>• DBC.TempTableStatsV</li> <li>• DBC.TriggersV[X]</li> <li>• DBC.UDTInfoV</li> <li>• DBC.UpdateUseCountV[X]</li> <li>• DBC.UserGrantedRightsV</li> <li>• DBC.UserRightsV</li> <li>• DBC.UserRoleRightsV</li> <li>• DBC.UsersV</li> <li>• DBC.User_Default_JournalsV[X]</li> <li>• DBC.UsrAsgdSecConstraintsV[X]</li> <li>• DBC.ViewStatsV</li> <li>• DBC.ZonesV[X]</li> <li>• DBC.ZoneGuestsV[X]</li> </ul> |
|---|--|--|

For example, the DBC.TablesVX and DBC.TablesX views provide the following information for each table to which the role of the user has access:

- If the role has privilege on all in TVMs
- If the role has privilege on this specific TVM

It also depends on the current session role of the user to determine what roles are used to verify privileges.

| If the current role is. . . Then . . . |  |
|--|--|
| not NULL or ALL                        | the current role and its nested role are used for security checks. |

| If the current role is . . . | Then . . .  |
|------------------------------|---|
| ALL                          | all the role(s) directly granted to the user and all their nested role(s) are used for security checks. |

X views run slower than their corresponding non-X versions because the X views have security checks. Compatibility X views are named the same as their corresponding non-X views followed by the character X. Unicode X views are named the same as their corresponding non-X views followed by the character V.

## Non-X Views

A non-X view does not end with the letter X (for example, DBC\_RCC\_MediaV). Such views return every row of every column defined on the underlying table.

## Other Views

Cost Profile views do not select any Vantage object names, so there is no Compatibility or Unicode distinction. Note, the Cost Profile views are only for the use of Teradata Services personnel.

- CostProfiles\_v
- CostProfileTypes\_v
- CostProfileValues\_v

Resource Usage views are discussed in *Teradata Vantage™ - Resource Usage Macros and Tables*, B035-1099.

|   |  |   |
|---|--|---|
| <ul style="list-style-type: none"> <li>• ResCPUUsageByAMPView</li> <li>• ResCPUUsageByPEView</li> <li>• ReslpmaView</li> <li>• ReslvprView</li> </ul> | <ul style="list-style-type: none"> <li>• ResSawtView</li> <li>• ResScpuView</li> <li>• ResShstView</li> <li>• ResSldvView</li> <li>• ResSpdskView</li> </ul> | <ul style="list-style-type: none"> <li>• ResSpmaView</li> <li>• ResSpsView</li> <li>• ResSvdsView</li> <li>• ResSvprView</li> </ul> |
|---|--|---|

## Teradata Secure Zone Views

For Vantage systems configured with secure zones, data dictionary access is constrained by zone.

The data dictionary contains zone specific system views ending in \_SZ for related Unicode and compatibility views. Zone specific views are not needed for VX views because they already contain security constraints in their definition that limit the result set to only the rows associated with the requesting user.

Even with Teradata Secure Zones configured, all users query the non \_SZ views. Privileges are checked against the non \_SZ view.

The following table shows data dictionary view access for the users.

| User  | Data Dictionary View Access  |
|---|--|
| <ul style="list-style-type: none"> <li>• DBC</li> <li>• Non-zone user with ZONE OVERRIDE</li> </ul> | These users get system-wide information from views without zone filtration. For example, if DBC or a non-zone user with ZONE OVERRIDE does a SELECT from the QryLogXMLV view, they will get information from the QryLogXMLV view without any filtration by zone. |
| <ul style="list-style-type: none"> <li>• Zone user</li> </ul>                                       | Access is constrained by zone. For example, if a zone user does a SELECT from the QryLogXMLV view, information is returned from the zone constrained view.   |
| <ul style="list-style-type: none"> <li>• Non-zone user without ZONE OVERRIDE</li> </ul>             | These users are restricted to non-zone dictionary data only.   |

The following views are constrained by zone.

|   |  |   |
|---|--|---|
| <ul style="list-style-type: none"> <li>• DBC.AccessLog[V]</li> <li>• DBC.AccLogRules[V]</li> <li>• DBC.AccountInfo[V]</li> <li>• DBC.AllRights[V]</li> <li>• DBC.AllRoleRights[V]</li> <li>• DBC.AllSpace[V]</li> <li>• DBC.All_RI_Children[V]</li> <li>• DBC.All_RI_Parents[V]</li> <li>• DBC.AMPUsage[V]</li> <li>• DBC.ArchiveLoggingObjsV</li> <li>• DBC.Authorizations[V]</li> <li>• DBC.BAR_ChildrenV</li> <li>• DBC.BAR_Databases2V</li> <li>• DBC.BAR_JoinIndicesV</li> <li>• DBC.BAR_ownerdbV</li> <li>• DBC.BAR_RI_Distinct_ParentsV</li> <li>• DBC.BAR_StatsV</li> <li>• DBC.BAR_Tables2V</li> <li>• DBC.BAR_TablesV</li> <li>• DBC.Children[V]</li> <li>• DBC.Columns[V]</li> <li>• DBC.ColumnsqV</li> <li>• DBC.ColumnsjqV</li> <li>• DBC.ColumnsUseCountV</li> <li>• DBC.ConnectRules[V]</li> <li>• DBC.ConstraintFunctionsV</li> <li>• DBC.Databases2[V]</li> <li>• DBC.Databases[V]</li> <li>• DBC.DatabaseUseCountV</li> <li>• DBC.Database_Default_Journals[V]</li> <li>• DBC.DBQLRules[V]</li> <li>• DBC.DeleteAccessLog[V]</li> </ul> | <ul style="list-style-type: none"> <li>• DBC.DeleteUseCountV</li> <li>• DBC.DiskSpace[V]</li> <li>• DBC.ErrorTblsV</li> <li>• DBC.ExpStatsV</li> <li>• DBC.IndexConstraints[V]</li> <li>• DBC.IndexStatsV</li> <li>• DBC.IndexUseCountV</li> <li>• DBC.Indices[V]</li> <li>• DBC.InsertUseCountV</li> <li>• DBC.JoinIndicesV</li> <li>• DBC.Journals[V]</li> <li>• DBC.LoadTablesInfoV</li> <li>• DBC.LogOnOff[V]</li> <li>• DBC.LogonRules[V]</li> <li>• DBC.MapGrantsV[X]</li> <li>• DBC.MapsV[X]</li> <li>• DBC.MultiColumnStatsV</li> <li>• DBC.MultiExpStatsV</li> <li>• DBC.ObjectUseCountV</li> <li>• DBC.ownerdb[V]</li> <li>• DBC.PartitioningConstraintsV</li> <li>• DBC.PeriodsV</li> <li>• DBC.ProfileAsgdSecConstraintsV</li> <li>• DBC.ProfileInfo[V]</li> <li>• DBC.QryLockLogXMLV</li> <li>• DBC.QryLog[V]</li> <li>• DBC.QryLogExplain[V]</li> <li>• DBC.QryLogExplainDocV</li> <li>• DBC.QryLogObjects[V]</li> <li>• DBC.QryLogParamJSON</li> <li>• DBC.QryLogParamV</li> <li>• DBC.QryLogSQL[V]</li> <li>• DBC.QryLogSQLDocV</li> <li>• DBC.QryLogSteps[V]</li> <li>• DBC.QryLogSummary[V]</li> </ul> | <ul style="list-style-type: none"> <li>• DBC.QryLogTDWM[V]</li> <li>• DBC.QryLogXMLDocV</li> <li>• DBC.QryLogUtilityV</li> <li>• DBC.QueryStatsV</li> <li>• DBC.ResolvedDTS[V]</li> <li>• DBC.RI_Child_Tables[V]</li> <li>• DBC.RI_Distinct_Children[V]</li> <li>• DBC.RI_Distinct_Parents[V]</li> <li>• DBC.RI_Parent_Tables[V]</li> <li>• DBC.RoleInfo[V]</li> <li>• DBC.RoleMembers[V]</li> <li>• DBC.SecConstraintsV</li> <li>• DBC.SecurityLog[V]</li> <li>• DBC.SessionInfo[V]</li> <li>• DBC.SHOWCOLCHECKS[V]</li> <li>• DBC.SHOWTBLCHECKS[V]</li> <li>• DBC.StatsV</li> <li>• DBC.StatUseCountV</li> <li>• DBC.Tables[V]</li> <li>• DBC.TableSize[V]</li> <li>• DBC.TableText[V]</li> <li>• DBC.Table_LevelConstraints[V]</li> <li>• DBC.TempTableStatsV</li> <li>• DBC.Triggers[V]</li> <li>• DBC.UpdateUseCountV</li> <li>• DBC.Userdb[V]</li> <li>• DBC.User_Default_Journals[V]</li> <li>• DBC.UsrAsgdSecConstraintsV</li> <li>• DBC.ViewStatsV</li> <li>• DBC.ZoneGuestsV</li> <li>• DBC.ZonesV</li> </ul> |
|---|--|---|

To see the secure zones views in the data dictionary, the Secure Zones feature must be enabled first. Contact your Teradata representative to enable this feature.

## Access to Data Dictionary Views

### Default PUBLIC Privileges for Views

The system grants the SELECT privilege to PUBLIC on most Data Dictionary views by default. All Vantage users have PUBLIC privileges.

For security and data integrity reasons, the system does not grant INSERT, UPDATE, and DELETE privileges to PUBLIC on Data Dictionary views.

For information on accessing views, see [Querying Data Dictionary Views](#).

### Views for Which PUBLIC Privileges Are not Granted by Default

The system grants default access privileges on some views only to user DBC. These views contain security information or other data that is not for general use, so they are not included when the system grants default PUBLIC privileges on other views.

You must GRANT privileges to the following restricted views for any users or roles that need access to them.

|  |   |  |
|--|---|--|
| <ul style="list-style-type: none"> <li>• DBC.AccessLog</li> <li>• DBC.AccLogRules</li> <li>• DBC.CSPSessionInfo</li> <li>• DBC.DBQLRules</li> <li>• DBC.DeleteAccessLog</li> <li>• DBC.DeleteOldInDoubt</li> <li>• DBC.InDoubtLog</li> </ul> | <ul style="list-style-type: none"> <li>• DBC.LogonRules</li> <li>• DBC.QryLog</li> <li>• DBC.QryLogExceptions</li> <li>• DBC.QryLogExplain</li> <li>• DBC.QryLogEvents</li> <li>• DBC.QryLogObjects</li> <li>• DBC.QryLogSQL</li> </ul> | <ul style="list-style-type: none"> <li>• DBC.QryLogSteps</li> <li>• DBC.QryLogSummary</li> <li>• DBC.QryLogTDWM</li> <li>• DBC.QryLogTDWMSum</li> <li>• DBC.SecurityLog</li> <li>• DBC.Software_Event_Log</li> </ul> |
|--|---|--|

### Compatibility View Privileges

Privileges for the compatibility views have been changed to read only and INSERT, UPDATE, DELETE, and SELECT privileges are granted under the following views:

- CollationsV
- CharTranslationsV
- HostsInfoV

### Trusted Sessions Support in X Views

The X Views return results based on the current authorized user, if the current authorized user is set to a Trusted Sessions proxy user. If the user is not a proxy user, the results returned are based on the current user and the current user's access rights.

**Note:**

Compatibility X Views do not support Trusted Sessions.

For example, when logged in as a middle tier user TRM, a query using the AMPUsageVX view returns the usage for the user TRM as follows:

```
sel username (FORMAT 'X(20)'), sum(cputime), sum(diskio)
from dbc.ampusagevx group by 1;
```

Results:

| UserName | Sum(CpuTime) | Sum(DiskIO) |
|----------|--------------|-------------|
| -----    | -----        | -----       |
| TRM      | 0.42         | 10,595      |

If the session is then set to a permanent proxy user (defined WITHOUT ROLE), the query returns the usage for the user PERMUSER1 as follows:

```
SET QUERY_BAND='PROXYUSER=PERMUSER1;' FOR SESSION;

sel username (FORMAT 'X(20)'), sum(cputime), sum(diskio)
from dbc.ampusagevx group by 1;
```

Result:

| UserName  | Sum(CpuTime) | Sum(DiskIO) |
|-----------|--------------|-------------|
| -----     | -----        | -----       |
| PERMUSER1 | 0.19         | 2,185       |

If the session is set to an application proxy user or permanent proxy user with specified roles, then authorization is based solely on the roles of the proxy user. For X views that return data based on the current user name only, the query returns no rows:

```
SET QUERY_BAND='PROXYUSER=APPLUSER;' FOR SESSION;

sel username (FORMAT 'X(20)'), cputime, diskio
from dbc.ampusagevx order by 1;
```

Result: No rows are returned.

```
*** Query completed. No rows found.
*** Total elapsed time was 1 second.
```

## Querying Data Dictionary Views

You can submit an SQL request to access the data in a system view and display or print the results.

For example:

```
SELECT * FROM DBC.AccLogRules
WHERE UserName = 'ALL';
```

### Note:

If an SQL request returns a security violation error, a reference to the alternate version of the view name may yield results. If neither version is available, use the HELP statement to inquire about individual objects. If this is not adequate, consult with the database administrator.

The following sections show various methods for retrieving directory information.

For a description of several common administrative uses of the information in Data Dictionary views, see the topics starting with [Tracking Resource Usage](#).

## How to See All Columns of a View

The output of some views extends beyond an 80-character width display. To see all the columns, you can either:

- Cast the name columns to shorter lengths
- Use the SET FOLDLINE and SET SIDETITLES options in BTEQ

## Querying X Versus Non-X Views

The amount of information that can be retrieved from system views at a particular site depends on:

- Whether an X version is available for a particular view
- The privilege granted, if any, on each available view
- Whether the statement references “view\_name” or “view\_nameX”

Assuming that both the X and non-X versions of the views are installed, and that the SELECT privilege is granted to PUBLIC on both versions, the information returned by an unconditional SELECT depends on the specified view name, as follows:

| A view specified as ... | Returns information about ...   |
|-------------------------|---|
| DBC.viewname            | <p>all objects for which entries exist in the underlying table.</p> <p><b>Note:</b><br/>Unconditional SELECTs on non-X views may cause the result to exhaust the available spool space of the user.</p> |

| A view specified as ... | Returns information about ...   |
|-------------------------|---|
| DBC.viewnameX           | only those objects that the requesting user: <ul style="list-style-type: none"> <li>• owns</li> <li>• created</li> <li>• has been granted privileges on</li> <li>• has access to via current role and the current nested roles of the role</li> </ul> |

## Using HELP and COMMENT

If the SELECT privilege on one or more system views is revoked or not granted to PUBLIC, you can use the HELP and COMMENT statements to obtain directory information about a particular object for which you have access.

HELP does not require long or complex queries and the results can be formatted for printing.

The COMMENT statement returns descriptive information about a database or database object. This includes information about Data Dictionary views.

For example, the following COMMENT statement:

```
COMMENT ON DBC.UserGrantedRightsV;
```

returns the following description:

```
The DBC.UserGrantedRightsV view provides information on access rights that the
current user has granted to other users. The column names are: DatabaseName,
TableName, FieldName, Grantee, AccessRight, WithGrant and AllnessFlag.
```

For more information on HELP and COMMENT, see *Teradata Vantage™ - SQL Data Definition Language Syntax and Examples*, B035-1144.

## Tracking Resource Usage

Vantage usage information is collected by user and by account number. Control is determined by the definition of each CREATE USER statement.

An account is associated with a group of users whose usage charges are maintained by Vantage. Account IDs may be established for a user with the ACCOUNT phrase in the CREATE USER statement. If ACCOUNT is not specified, the default is the account ID of the immediate owner of the user. If the owner has multiple account IDs, the first account ID is taken for the user as the default.

The following sections describe the accounting information available for any Vantage session.

You can monitor usage with the following Data Dictionary views:

- AccountInfo

- AllSpace
- AMPUsage
- DiskSpace
- LogOnOff
- TableSize

The AllSpace, DiskSpace, TableSize, and AMPUsage views are useful in tracking Vantage resources for accounting purposes, and in determining how effectively resources are being utilized by accounts and users.

## AllSpace and TableSize Views

You use the AllSpace, Diskspace, and TableSize views that access data from the DatabaseSpace table to show how efficiently a table is distributed across the AMPs on which it is stored.

For example, the statement:

```
SELECT CurrentPerm, PeakPerm, AMP FROM DBC.AllSpace
WHERE TableName = 'Personnel.Department';
```

returns a row for each AMP on which the Department table is stored. Data in the CurrentPerm column shows, in bytes, how Department data is distributed across the AMPs.

If the distribution is uneven, you can tell from the CurrentPerm data. In addition, the PeakPerm column data indicates any fluctuations in distribution because the table was created.

## AMPUsage View

The AMPUsage view supplies information about AMP CPU time consumed, and the number of AMP to disk read and write operations generated by a given user or account.

This view also tracks the activities of any console utilities. A row is returned for each AMP in the system unless aggregate figures are specified.

When you ask for resource usage logging, data about CPU overhead, user service, and user execution is collected by vproc type and by node.

You can use the AMPUsage, AllSpace, DiskSpace, and TableSize views to summarize resource usage for all AMPs, or for AMPs on which data is stored.

## Example: Using DiskSpaceV

To obtain a list (in order of the amount of space used) of those databases currently using more than 80% of their permanent space allocation, enter:

```
SELECT DatabaseName, SUM(CurrentPerm)
FROM DBC.DiskSpaceV
```

```
GROUP BY DatabaseName
HAVING (SUM(CurrentPerm)/NULLIFZERO(SUM(MaxPerm))) > .8
ORDER BY SUM(CurrentPerm) DESC;
```

You can also use the AMPUsage and DiskSpace views to compile and maintain usage statistics that can later be selected and analyzed as described in the following sections.

## Compiling AMPUsage Statistics

You can use the AMPUsage view to build and maintain a history table of CPU time and disk I/O statistics for each username/accountname.

To create the history table, enter:

```
CREATE TABLE AMPUseHist
( AccountName VARCHAR(30),
  UserName VARCHAR(30),
  CPUtime INTEGER,
  DiskIO INTEGER,
  Date DATE, Time FLOAT )
PRIMARY INDEX (UserName, AccountName);
```

Periodically, collect usage statistics using this procedure:

1. Select statistics from the AMPUsage view and insert them in the history table.
2. Reset AMPUsage counters to zero for the next collection period.

This procedure can be carried out using the BTEQ script:

```
.LOGON username, password

INSERT INTO AMPUseHist
SELECT AccountName, UserName, SUM(CPUtime), SUM(DiskIO), DATE, TIME
FROM DBC.AMPUsage
GROUP BY AccountName, UserName, DATE, TIME;
UPDATE DBC.AMPUsageV
SET CPUSum = 0, DiskIO = 0 ALL;
.QUIT
```

The units in which Disk I/O are measured represent data block accesses. CPU time is measured in seconds.

Refer to the DiskSpaceV View to determine how you can use the DiskSpaceV view to build and maintain a table of disk space usage.

After a collection period, you may select AMPUsageV and DiskSpaceV statistics from the history tables to query the data or to archive the data on a client system. You can use the selected data in sequential data sets on the host computer for subsequent analysis.

For example, you can use a BTEQ script to:

- Create a client-resident file
- Use the BTEQ .EXPORT command to save the data being selected into that file
- Select all rows from the DiskSpaceV history table

Create the disk space history table.

```
CREATE TABLE DiskSpaceHist (DataBaseName VARCHAR(128) CHARACTER SET UNICODE,
    AccountName VARCHAR(128) CHARACTER SET UNICODE,
    MaxPerm FLOAT,
    MaxSpool FLOAT,
    CurrentPerm FLOAT,
    PeakPerm FLOAT,
    PeakSpool FLOAT,
    CollectDate DATE,
    CollectTime FLOAT )
PRIMARY INDEX (DataBaseName, AccountName);
```

Periodically, you can collect usage statistics using the following procedure:

1. Select statistics from the DiskSpaceV[X] view and insert them in the history table.
2. Reset DiskSpace counters to zero for the next collection period.

---

**Note:**

You can reset the maximum and peak DiskSpace counters to zero using the ClearPeakDisk macro, which is provided on the release tape.

---

This procedure can be carried out using the following BTEQ script:

```
.LOGON username, password

INSERT INTO DiskSpaceHist
    SELECT DataBaseName, AccountName,
    SUM(MaxPerm),
    SUM(MaxSpool),
    SUM(CurrentPerm),
    SUM(PeakPerm),
    SUM(PeakSpool),
    DATE, TIME
    FROM DBC.DiskSpaceV
```

```

GROUP BY DataBaseName, AccountName, DATE, TIME;

EXECUTE DBC.ClearPeakDisk;

.QUIT

```

The following example shows how a BTEQ job is used to select data from the DiskSpace history table.

```

//JOBNAME JOB    jobcard
//EXTRACT EXEC   PGM=ITBMAIN
//STEPLIB DD     DSN=TERADATA.APPLOAD,DISP=SHR
//SYSPRINT DD    SYSOUT=*
//SYSABEND DD    SYSOUT=*
//SAVEDATA DD    DSN=ACC.SAVEDATA.DATA,DISP=(NEW,CATLG)
//              UNIT=SYSDA,SPACE=(TRK,(1,1),RLSE),
//              DCB=(LRECL=80,RECFM=FB,BLKSIZE=800)
//SYSIN DD      DATA,DLM=##
.LOGON somebody,password
.EXPORT DATA DDNAME=SAVEDATA
SELECT * FROM DiskSpaceHist
ORDER BY Date,Time;
.QUIT
##

```

After it is selected and stored, historical data can be used for analysis, as follows:

- Client-resident software packages such as SAS can be used to perform analysis and other statistical manipulation on the data.
- Graphic software packages can be used to display the data.

## Related Topics

| For more information on ... | See ...   |
|-----------------------------|---|
| PM/API requests             | <i>Teradata Vantage™ - Application Programming Reference</i> , B035-1090. |
| resource usage              | <i>Teradata Vantage™ - Resource Usage Macros and Tables</i> , B035-1099.  |

## Tracking User Activity

The Software\_Event\_Log view displays, by date and time, any events that have affected processing, such as a memory parity error, a disabling of logons, a database restart, or execution of a PM/API SET command, along with an associated message. The processor is identified when appropriate.

The LogOnOff view, because it chronologically records all logon and logoff activity as well as the reasons for unsuccessful logons and logoffs, allows you to detect actual and attempted security violations. This view also lets you know how long any user is connected to Vantage.

## Tracking Logon Rules

The LogonRules view is used to review the rules generated by the GRANT LOGON and REVOKE LOGON statements. These statements define which user can log on from what mainframe or LAN host connection, and whether the logon string of the user is acceptable without a password.

The initial default is that all users may logon from all hosts, and that every logon string must contain a password. To change the default, use the GRANT LOGON and REVOKE LOGON statements.

## Tracking Privileges

The UserRightsV view contains information about the privileges that have been granted to any user.

See the description of the GRANT statement in *Teradata Vantage™ - SQL Data Control Language*, B035-1149, for an explanation of the types of privileges, and how they are granted.

If a more detailed audit trail is necessary, this information may be supplemented by log entries that provide an audit trail of the results of checks against requests to access table data. See *Teradata Vantage™ - Advanced SQL Engine Security Administration*, B035-1100.

## Accessing PM/API-Related Data in the Data Dictionary

Several Data Dictionary tables log Performance Monitor/Application Programming Interface (PM/API)-related values for some columns. You can use the PM/API information to track the use of MONITOR partition functions.

The following DBC tables contain PM/API information:

- DBC.AccessRights
- DBC.SessionTbl
- DBC.SW\_Event\_Log

---

### Note:

Execution of the PM/API SET RESOURCE, SET SESSION, and ABORT SESSION requests are considered major system events and are logged to the DBC.SW\_Event\_Log table.

---

- DBC.Dbase
- DBC.TVM

The following Data Dictionary views, which are built on those tables, also carry PM/API-related data:

- AllRightsV[X]
- AllSpaceV[X]

- DatabasesV[X]
- DiskSpaceV[X]
- LogOnOffV
- Software\_Event\_LogV
- SessionInfoV[X]
- UserGrantedRightsV
- UserRightsV

The following example queries return information about PM/API-related activities.

### Example: Using SessionInfoV to Determine Who Is Using the Monitor

To determine who is using the monitor, enter the following:

```
SELECT UserName, IFPNo FROM DBC.SessionInfoV
WHERE Partition = 'MONITOR' ;
```

### Example: Using AllRightsV to Determine User Access Rights

To determine which users have the privilege to force other users off the system, enter the following:

```
SELECT DISTINCT UserName FROM DBC.AllRightsV
WHERE AccessRight = 'AS' ;
```

The 'AS' indicates the ABORT SESSION privilege.

### Example: Using LogOnOffV to Determine Which Users Were Forced Off the System

To find out which users have been forced off the system (using PM/API ABORT SESSION) in the past two days, enter the following:

```
SELECT DISTINCT UserName FROM DBC.LogOnOffV
WHERE Event = 'Forced Off'
AND LogDate > DATE - 3 ;
```

## Using System Views

### System Calendar View

The Sys\_Calendar.CALENDAR system view helps to extend the properties of a DATE data type column by means of a join. The columns of the view contain data only for the active calendar for the session.

The calendar dates range from 1900 to 2100 and are stored in a table in the Sys\_Calendar database.

The administrator must run DIPCAL SQL and DIPSYSFNC scripts from the DIP utility to create the Sys\_Calendar database and the versions of the Calendar view.

There are three versions of the Sys\_Calendar.Calendar view:

- Sys\_Calendar.CALENDAR
- Sys\_Calendar.CALENDAR\_TD\_ISO\_COMPATIBLE (view used for the current release)
- Sys\_Calendar.CALENDAR\_TD1310 (legacy view that uses arithmetic to compute column values)

Sys\_Calendar.CALENDAR and Sys\_Calendar.CALENDAR\_TD\_ISO\_COMPATIBLE have the same definition after running the DIP utility.

The current version (Sys\_Calendar.CALENDAR\_TD\_ISO\_COMPATIBLE) is an internal view and can only be accessed by user DBC.

### Sys\_Calendar.CALENDAR\_TD1310

---

#### Note:

Sys\_Calendar.CALENDAR\_TD1310 can only be used for the system-defined calendar. If you try to use it when the session calendar is ISO or COMPATIBLE, the returned values will not be valid because they will always be from the system-defined calendar.

---

The current versions (Sys\_Calendar.CALENDAR and Sys\_Calendar.CALENDAR\_TD\_ISO\_COMPATIBLE) use embedded services system functions to compute some column values. The Sys\_Calendar.CALENDAR\_TD1310 version computes all column values using arithmetic, which generally takes less time than computing values using UDFs.

To use the legacy TD1310 version of the view, replace the definition of Sys\_Calendar.CALENDAR with the definition of Sys\_Calendar.CALENDAR\_TD1310. Follow these steps:

1. Use this statement to view the definition of the Sys\_Calendar.Calendar\_TD1310 version:

```
SHOW VIEW Sys_Calendar.Calendar_TD1310;
```

2. Use REPLACE to replace the definition of Sys\_Calendar.CALENDAR with the definition of the Sys\_Calendar.CALENDAR\_TD1310 version.

After you redefine the current version, it no longer uses embedded services system functions to compute column values for the following columns:

- day\_of\_month
- weekday\_of\_month
- month\_of\_quarter
- month\_of\_year
- quarter\_of\_year
- year\_of\_calendar

## To Revert the View to the Current Version

If the view was redefined to the Sys\_Calendar.CALENDAR\_TD1310 view definition and you want to use the current version, you need to redefine Sys\_Calendar.CALENDAR so that it has the definition of the current version. The current version (named Sys\_Calendar.CALENDAR\_TD\_ISO\_COMPATIBLE), is an internal view and has restricted access.

---

### Note:

The current version can be used with the Teradata, ISO, and COMPATIBLE session calendars.

---

To redefine Sys\_Calendar.CALENDAR from the Sys\_Calendar.CALENDAR\_TD1310 definition, replace the Sys\_Calendar.CALENDAR definition with the definition of Sys\_Calendar.CALENDAR\_TD\_ISO\_COMPATIBLE:

1. Use this statement to view the definition of the current view:

```
SHOW VIEW Sys_Calendar.Calendar_TD_ISO_COMPATIBLE;
```

2. Use REPLACE to replace the definition of Sys\_Calendar.CALENDAR with the definition of Sys\_Calendar.Calendar\_TD\_ISO\_COMPATIBLE.

After you redefine Sys\_Calendar.CALENDAR, it uses Embedded Services functions to compute column values for the following columns:

- day\_of\_month
- weekday\_of\_month
- month\_of\_quarter
- month\_of\_year
- quarter\_of\_year
- year\_of\_calendar

## Privilege

By default, the system grants the SELECT privilege on Sys\_Calendar.Calendar to PUBLIC.

## Example: Using Sys\_Calendar.Calendar

You are encouraged to define views on the Calendar system view because of its convenience.

A useful view to define on Calendar is Today:

```
CREATE VIEW Today AS (
SELECT * FROM Sys_Calendar.Calendar
  WHERE Sys_Calendar.Calendar.calendar_date = DATE
);
```

The Calendar system view permits easy specification of arithmetic expressions and aggregation. This is particularly useful in online analytical processing environments where requests commonly aggregate values by weeks, months, year-to-date, years, and so on. The following is an example.

What are the dollar sales for this week, last week, and the same weeks last year for all items in the sportswear department for women?

```
SELECT a2.week_of_calendar, SUM(a1.price)
  FROM Sales a1, CALENDAR a2, Item a3, Department a4, Today a5
 WHERE a1.calendar_date=a2.calendar_date
 AND (a2.week_of_calendar=a5.week_of_calendar
 OR a2.week_of_calendar=a5.week_of_calendar - 1
 OR a2.week_of_calendar=a5.week_of_calendar - 52
 OR a2.week_of_calendar=a5.week_of_calendar - 53
 )
 AND a1.itemID=a3.itemID
 AND a3.classID=a4.classID
 AND a4.classDesc='Women's Sportswear'
 GROUP BY a2.week_of_calendar
 ORDER BY a2.week_of_calendar;
```

## SQLJ System Views

The SQLJ database and its views are used by the system to manage JAR files that implement Java external stored procedures.

The SQLJ database and its components are created via a DIP script called DIPSQLJ. The DIPSQLJ script is run as part of DIPALL and follows the pattern set by the DBC Data Dictionary initialization process. The DIPSQLJ script revokes all privileges, which could result in modification of SQLJ database views and external stored procedures.

The SQLJ database requires sufficient space for all required components to be defined within it, and the initial space allocation for this database is determined based upon that fact. The best practice is to avoid the

placement of additional items into this database, as the SQLJ database could be considered an extension to the Vantage Data Dictionary defined in the DBC database.

The SQLJ system database contains the views:

- JAR\_JAR\_USAGE
- JARS
- ROUTINE\_JAR\_USAGE

## JAR\_JAR\_USAGE

The JAR\_JAR\_USAGE view identifies each JAR owned by a given user or database on which other JARs defined on the system are dependent.

| View Column Name | Description   | Data Type  | Format |
|------------------|---|--|--------|
| Databaseld       | The identifier of the database or user in which the JAR identified by JarName is defined.   | BYTE(4)<br>NOT NULL  | X(8)   |
| JarName          | The name designator for a JAR that depends on code from another JAR.  | VARCHAR(128)<br>UNICODE<br>UPPERCASE<br>NOT CASESPECIFIC<br>NOT NULL | X(128) |
| PathJarName      | The name designator for a JAR, which is in the SQL-Java path of the JAR identified by JarName. The SQL-Java path is the search path defined for a particular JAR. With a SQL-Java path defined, a method defined in a particular JAR (JAR A) may invoke a method which is defined in another JAR (JAR B), if JAR B (and all its classes) is contained in the SQL-Java path of JAR A. A SQL-Java path for a JAR can only be created, altered, or dropped via a call to SQLJ.Alter_Java_Path. | VARCHAR(128)<br>UNICODE<br>UPPERCASE<br>NOT CASESPECIFIC<br>NOT NULL | X(128) |

## Corresponding System Tables

The corresponding system tables for SQLJ.JAR\_JAR\_USAGE are:

- DBC.JAR\_JAR\_USAGE
- DBC.DBase
- DBC.TVM

### Example: Using SQLJ.Jar\_Jar\_Usage

The following SELECT statement retrieves information about each JAR owned by a given user or database on which other JARs are dependent.

```
==> SELECT * FROM sqlj.jar_jar_usage;
```

Result:

| DatabaseId | JarName | PathJarName |
|------------|---------|-------------|
| 0000B905   | JAR1    | JAR4        |
| 0000B905   | JAR2    | JAR3        |

## JARS

The JARS view identifies the installed JARs defined on the system that are accessible to the current user or database.

| View Column Name | Description   | Data Type  | Format |
|------------------|---|--|--------|
| Databaseld       | The identifier of the database or user in which the JAR identified by JarName is defined. | BYTE(4)<br>NOT NULL  | X(8)   |
| JarName          | The name designator for an installed JAR.   | VARCHAR(128)<br>UNICODE<br>UPPERCASE<br>NOT CASESPECIFIC<br>NOT NULL | X(128) |
| JarId            | The unique identifier for the JAR identified by JarName.                                  | BYTE(6)<br>NOT NULL  | X(12)  |

### Corresponding System Tables

The corresponding system tables for SQLJ.JARS are:

- DBC.DBase
- DBC.JARS

### Example: Finding JAR Objects Accessible to a USER from SQLJ.Jars

The following SELECT statement retrieves information about all JAR objects that are accessible to the current user or database.

```
==> SELECT * FROM sqlj.jars;
```

Result:

| DatabaseId | JarName | JarId        |
|------------|---------|--------------|
| 0000B905   | JAR1    | 000061280000 |
| 0000B905   | JAR2    | 000062280000 |
| 0000B905   | JAR3    | 000063280000 |

## ROUTINE\_JAR\_USAGE

The ROUTINE\_JAR\_USAGE view identifies the JARs owned by a given user or database on which external Java routines defined on the system are dependent.

| View Column Name | Description   | Data Type  | Format |
|------------------|---|--|--------|
| Databaseld       | The identifier of the database or user in which the JAR identified by JarName is defined.                   | BYTE(4)<br>NOT NULL  | X(8)   |
| RoutineName      | The name designator for a Java external stored procedure that depends on the JAR identified by JarName.     | VARCHAR(128)<br>UNICODE<br>UPPERCASE<br>NOT CASESPECIFIC<br>NOT NULL | X(128) |
| JarName          | The name designator for a JAR that contains code that the RoutineName external stored procedure depends on. | VARCHAR(128)<br>UNICODE<br>UPPERCASE<br>NOT CASESPECIFIC<br>NOT NULL | X(128) |
| Signature        | The signature defined for Java external stored procedure identified by RoutineName.                         | VARCHAR(6400)<br>LATIN<br>CASESPECIFIC                               | X(255) |

## Corresponding System Tables

The corresponding system tables for SQLJ.ROUTINE\_JAR\_USAGE are:

- DBC.ROUTINE\_JAR\_USAGE
- DBC.DBase
- DBC.TVM

## Example: Determining Users JAR Objects Using SQLJ.Routine\_JAR\_Usage

The following SELECT statement retrieves information about all of the JARs owned by a given user or database on which dependent external Java routines are defined. Information about these routines is also returned.

```
==> SELECT * FROM sqlj.routine_jar_usage;
```

Result:

```
DatabaseId  0000B905
  JarName  JAR1
RoutineName MYINT
  Signature (I[I)V
DatabaseId  0000B905
  JarName  JAR2
RoutineName MYDEC
  Signature (Ljava/math/BigDecimal;[Ljava/math/BigDecimal;)V
DatabaseId  0000B905
  JarName  JAR3
RoutineName MYVARBYTE
  Signature ([B[Ljava/lang/String;)V
```

## User Installed Files View

The user installed files (UIF\_V) view in the SYSUIF database is based on the DBC.UIF\_INFO table. This view identifies the user installed files (UIFs) defined on the system that are accessible to the current user or database.

## View Definition

This table lists and describes the columns of the view.

| View Column | Data Type                | Format | Source Table.Column |
|-------------|--------------------------|--------|---------------------|
| Databaseld  | BYTE(4)<br>NOT NULL      | X(8)   | Dbase.Databaseld    |
| TVMNameI    | VARCHAR(128)<br>NOT NULL | X(128) | TVM.NameI           |
| UIFID       | BYTE(6)<br>NOT NULL      | X(12)  | UIF_INFO.UIFID      |

## Example: Using SYSUIF.UIF\_V

The following SELECT statement displays the UIFs that are accessible by database or user ID 00000104.

```
SELECT * from sysuif.uif_v;
```

Result:

```
DatabaseId 00000104
UIFFileName r2.r
  TVMNameI R2
    UIFId 0000060A0000
DatabaseId 00000104
UIFFileName p1.py
  TVMNameI P1
    UIFId 0000070A0000
```

## Using Unicode Views to Update Object Names

Based on the language support mode, a compatibility view casts the object name to 30 fixed characters in the Latin or Kanji server character set. Whenever data is not accessed via direct reference, the view cannot be used to update the underlying table. Therefore, you cannot use compatibility views to update object names in the underlying tables. Instead, you must use Unicode views.

Processes that ran on Release 12.0 or earlier may attempt to make updates through the DBC.Collations, DBC.CharTranslations or DBC.HostsInfo views. These views are deprecated and have not been supported since Release 12.0. Teradata strongly recommends that you update such processes to reference the corresponding Unicode views (for example, CollationsV, CharTranslationsV, and HostsInfoV) instead of the compatibility views.

## Teradata QueryGrid

The Teradata QueryGrid™ connectors provide an SQL interface for transferring data between Vantage and remote hosts. For example, Teradata QueryGrid™ allows data transfers between Vantage and Presto or Vantage and Hive, and so on. The following views are related to Teradata QueryGrid:

- ServerInfoV[X]
- ServerV[X]
- TblSrvInfoV[X]
- TblSrvV[X]

The following views also carry Teradata QueryGrid™-related data:

- QryLogStepsV - see the ServerByteCount column
- QryLogV - see the TotalServerByteCount column

## TDMaps

The TDMaps database, and its tables and views, stores metadata that is used by the system when moving tables to new maps. TDMaps also contains SQL procedures that perform tasks related to system expansion.

For information about MAPS architecture (MAPS), see *Teradata Vantage™ - Database Administration*, B035-1093 and *Teradata Vantage™ - SQL Operators and User-Defined Functions*, B035-1210.

# Views Reference

## Overview

The following topics describe all the Data Dictionary views in alphabetical order. For views with a VX view, the columns shown in the tables are from the VX view and the title of the section is <viewname>V[X]; otherwise, the title indicates the exact view from which the columns are listed, such as AccessLogV.

You can use Teradata Studio or Teradata Studio Express to view the Data Dictionary.

To see the source table for a column, query the definition for the view with the following SQL statement: `show view <viewname>;`. For example:

```
show view tablesvx;
```

DBC.UserDB and DBC.OwnerDB system views are not covered in this document because they are not usually referenced directly by users. These views are used only to join other system tables and views (especially the X version of views).

Data Dictionary views fall into the following categories:

| Type                 | Information Stored   |
|----------------------|--|
| Operations           | Internal database operations   |
| Database             | Database instance and owner  |
| Schema               | Database schema (tables, columns, and so on)   |
| Integrity            | Data integrity (constraints and so on)   |
| Security             | Data security (roles, grants, access, and so on)   |
| Query                | Database Query Log   |
| Accounting           | Database usage (accounts, space, and so on)  |
| Optimizer Statistics | Statistics collected on indexed, non-indexed columns (including row and column PARTITION statistics) and expressions on permanent tables, base and materialized global temporary tables (for the current session) and join and hash indexes. |
| TDMaps               | Stores metadata that is used by the system when moving tables to new maps.   |

### Note:

The results shown in the examples in the following sections are for illustration purposes only. You can use utilities and tools, such as BTEQ or other third-party products, to enter queries and format the results differently.

## TDMaps Views

The TDMaps database, and its tables and views, stores metadata that is used by the system when moving tables to new maps. TDMaps also contains SQL procedures that perform tasks related to system expansion.

The procedures and tables defined within TDMaps fall into two general categories: Advisor and Mover. Advisor procedures are responsible for analyzing user tables within logged query plans to make recommendations for moving a set of tables onto new maps. The recommendations can optionally be customized by callers and then input to the Mover. Mover procedures are responsible for moving the data for a specified list of tables onto new maps.

Users can issue SELECT statements to view the data in the views. For example, select all the rows from the action history view:

```
SELECT * FROM TDMaps.ActionHistoryVX;
```

---

### Note:

The user must have been granted SELECT on TDMaps to execute this statement; for example, GRANT SELECT ON TDMaps to <user>;

---

TDMaps provides the following views used by Advisor procedures:

- **ActionsV[X]** – list of planned map actions for a table or database. The advisor populates the ActionsTbl and the mover reads the actions from the table and schedules them for execution. Available actions are: move a table to a new map or do not perform any action on the table for the specified destination map. This is a secure zone constrained view. ActionsVX returns only objects to which the user has access.
- **ExclusionListsV[X]** – list of databases, tables, join indexes, and hash indexes that are not to be moved. The CreateExclusionList table is created by calling CreateExclusionListSP. This is a secure zone constrained view. ExclusionListsVX returns only the objects to which the user has access.
- **MapListsV[X]** – list of maps and the maps included within each list. This is a secure zone constrained view. MapListsVX returns only the maps to which the user has access.
- **ObjectListsV[X]** – list of databases, tables, join indexes, and hash indexes. This is a secure zone constrained view. ObjectListsVX returns only the objects to which the user has access.
- **SettingsV** – stores settings that are used by the stored procedures and views. Please contact Teradata Support for assistance before making changes to SettingsTbl.
- **TableToSparseMapSizingV[X]** – identifies which map the Advisor thinks a table should be in. This view can be joined with the ObjectUseCountV view to get object usage information: the DatabaseName and TableName columns in TableToSparseMapSizingV can be joined with the DatabaseName and ObjectName columns in ObjectUseCountV. This is a secure zone constrained view. TableToSparseMapSizingVX returns only objects to which the user has access.

TDMaps provides the following view used by Mover procedures:

- **ActionHistoryV[X]** – lists map related actions that are running or have completed. The ActionHistoryTbl is updated by the mover procedure. After the mover takes an action from the ActionsTbl, it inserts

a row into ActionHistoryTbl and sets its status to In Progress. After the action completes, the mover updates the status to Complete. This is a secure zone constrained view. ActionHistoryVX returns only the objects the user has access to.

## AccessLogV

**Category:** Security

**Database:** DBC

| View Column   | Data Type   | Format         | Comment  |
|---------------|---|----------------|--|
| LogDate       | DATE NOT NULL   | YY/MM/DD       | Returns the date that the access log entry was made.   |
| LogTime       | FLOAT NOT NULL  | 99:99:99       | Returns the time of day that the event occurred as HH:MM:SS.   |
| LogonDate     | DATE NOT NULL   | YY/MM/DD       | Returns the date that the session for which the log entry was made was logged on to the Teradata Database.   |
| LogonTime     | FLOAT NOT NULL  | 99:99:99       | Returns the time on which logon for the session occurred (useful on logoff events).  |
| LogicalHostId | SMALLINT NOT NULL                                       | ZZZ9           | Returns a unique identifier of the logon source for the logged query. A value of zero indicates an internal session.                                   |
| IFPNo         | SMALLINT NOT NULL                                       | -(5)9          | Returns the vproc number of the PE through which the session was connected or assigned.  |
| SessionNo     | INTEGER NOT NULL  | --,---,---,--9 | Returns the session identifier assigned to the session by the TDP or LAN interface.  |
| UserName      | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)         | The AccessLogV.UserName field shows the name of the user who was using the session.  |
| AccountName   | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)         | Expanded account in effect when a request was submitted. SYSTEMACCOUNTNO tracks console utility activity such as table rebuild, Diskcopy, or Scandisk. |
| OwnerName     | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC             | X(128)         | The AccessLogV.OwnerName field shows the user whose rights are being logged when a macro or view references other objects.                             |
| AccessType    | CHAR(2) LATIN<br>NOT CASESPECIFIC<br>NOT NULL           | X(2)           | (for "Acr"-type columns) Returns the type of privilege for which the check that generated this log entry was performed.                                |
| Frequency     | CHAR(1) LATIN<br>NOT CASESPECIFIC<br>NOT NULL           | X(1)           | Logging frequency (occurrences): B (FIRST and LAST), E (each), F (first), L (last).  |

| View Column   | Data Type   | Format          | Comment  |
|---------------|---|-----------------|--|
| EventCount    | INTEGER   | --,---,---,---9 | Returns the number of duplicate rows (that is, same occurrences) that preceded a 'LAST-occurrence- only' row.  |
| AccLogResult  | CHAR(1) LATIN<br>NOT CASESPECIFIC<br>NOT NULL           | X(1)            | Returns a code to indicate how the access request for which this log entry was made.   |
| DatabaseName  | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)          | Returns the database or user name of the object for which this log entry was made.   |
| TVMName       | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC             | X(128)          | Returns the table, view, stored procedure, macro, user-defined types, user-defined methods, or user-defined function name of the object for which this log entry was made. |
| ColumnName    | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC             | X(128)          | The AccessLogV.ColumnName field shows the name of the column which was entered on the access log.  |
| StatementType | VARCHAR(20) LATIN<br>NOT CASESPECIFIC<br>NOT NULL       | X(20)           | Statement type issued by the query. Final statement type in a multistatement request.  |
| StatementText | VARCHAR(8192)<br>UNICODE<br>NOT CASESPECIFIC            | X(8192)         | Returns (if so defined in the associated rule) the text of the statement that caused the privilege check for which this log entry was made.                                |
| QueryBand     | VARCHAR(12304)<br>UNICODE<br>NOT CASESPECIFIC           | X(12304)        | Returns the band under which the query is submitted.   |
| ProxyUser     | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC             | X(128)          | Returns the name of the proxy user.  |
| ConstraintId  | BYTE(4)   | X(8)            | The AccessLogV.ConstraintId field shows the id of the security constraint which was entered on the access log.   |

## Usage Notes

Each row displays the results of a privilege check. Whether a privilege check is logged depends on the presence and the criteria of an access logging rule (see [AccLogRulesV](#)).

### Possible Values for AccessType

| Value | Description              |
|-------|--------------------------|
| AE    | ALTER EXTERNAL PROCEDURE |

| Value | Description  |
|-------|--|
| AF    | ALTER FUNCTION   |
| AN    | ANY PRIVILEGE * h (*indicates a HELP or SHOW statement for which at least one privilege, but no specific privilege, is required) |
| AP    | ALTER PROCEDURE  |
| AS    | ABORT SESSION  |
| CA    | CREATE AUTHORIZATION   |
| CD    | CREATE DATABASE  |
| CE    | CREATE EXTERNAL PROCEDURE  |
| CF    | CREATE FUNCTION  |
| CG    | CREATE TRIGGER   |
| CM    | CREATE MACRO   |
| CO    | CREATE PROFILE   |
| CP    | CHECKPOINT   |
| CR    | CREATE ROLE  |
| CT    | CREATE TABLE   |
| CU    | CREATE USER  |
| CV    | CREATE VIEW  |
| D     | DELETE   |
| DA    | DROP AUTHORIZATION   |
| DD    | DROP DATABASE  |
| DF    | DROP FUNCTION  |
| DG    | DROP TRIGGER   |
| DM    | DROP MACRO   |
| DO    | DROP PROFILE   |
| DP    | DUMP   |
| DR    | DROP ROLE  |
| DT    | DROP TABLE   |
| DU    | DROP USER  |
| DV    | DROP VIEW  |

| Value | Description  |
|-------|--|
| E     | EXECUTE  |
| EF    | EXECUTE FUNCTION   |
| GC    | CREATE GLOP  |
| GD    | DROP GLOP  |
| GM    | GLOP MEMBER  |
| HR    | HUT RELEASE LOCK * (*indicates a client system utility lock is involved, which could require a check for one or more privileges associated with DUMP and RESTORE.) |
| I     | INSERT   |
| IX    | INDEX  |
| MR    | MONITOR RESOURCE   |
| MS    | MONITOR SESSION  |
| OP    | CREATE OWNER PROCEDURE   |
| PC    | CREATE PROCEDURE   |
| PD    | DROP PROCEDURE   |
| PE    | EXECUTE PROCEDURE  |
| RF    | REFERENCES   |
| RS    | RESTORE  |
| S     | RETRIEVE/SELECT  |
| SR    | SET RESOURCE RATE  |
| SS    | SET SESSION RATE   |
| ST    | STATISTICS   |
| U     | UPDATE   |
| UM    | UDT METHOD   |
| UT    | UDT TYPE   |
| UU    | UDT USAGE  |
| WL    | WRITE LOCK * (*indicates a locking object name FOR WRITE is involved, which may require checks for INSERT, UPDATE, and/or DELETE privileges.)                      |

## Possible Values for AccLogResult

| Value | Description  |
|-------|--|
| G     | GRANTED  |
| D     | DENIED   |
| U     | U indicates that authorization is granted, but the access rights check was bypassed. |

## Example: Using AccessLogV

The following SELECT retrieves the name of the submitting user from the AccessLogV view, the type of request, and the request text of each request that caused a privilege check to be logged on a specific date. The response shows that one request caused a privilege check to be logged on that date. (The statement text column has been truncated in the results.)

```
SELECT LogDate, UserName, AccessType, StatementText
FROM DBC.AccessLogV WHERE LogDate = 890510;
```

Result:

| LogDate  | UserName | AccessType | StatementText                 |
|----------|----------|------------|-------------------------------|
| 89/05/10 | Jones    | CT         | CREATE TABLE Jones.EmpDup (Em |

## AccLogRulesV

**Category:** Security

**Database:** DBC

| View Column  | Data Type   | Format | Comment  |
|--------------|---|--------|--|
| UserName     | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | The AccLogRulesV.UserName field shows the user to which the access logging rule applies. It may be 'All' to say all users.   |
| DatabaseName | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | Returns the database or user name of the object for which this log entry was made.   |
| TVMName      | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | Returns the table, view, stored procedure, macro, user-defined types, user-defined methods, or user-defined function name of the object for which this log entry was made. |

| View Column        | Data Type  | Format | Comment   |
|--------------------|--|--------|---|
| ConstraintName     | VARCHAR(128)<br>UNICODE<br>UPPERCASE<br>NOT NULL | X(128) | The AccLogRulesV.ConstraintName field shows security constraint to which the access logging rule applies. It may be 'All' to say all security constraints.          |
| AcrAlterFunction   | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | The AccLogRulesV.AcrAlterFunction field shows the logging in effect for the Alter Function privilege on the object(s) and/or users to which the rule applies.       |
| AcrCheckPoint      | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | The AccLogRulesV.AcrCheckPoint field shows the logging in effect for the CHECKPOINT privilege on the objects or users (or both) to which the rule applies.          |
| AcrCreateDataBase  | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | The AccLogRulesV.AcrCreateDataBase field shows the logging in effect for the CREATE DATABASE privilege on the objects or users (or both) to which the rule applies. |
| AcrCreateFunction  | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | The AccLogRulesV.AcrCreateFunction field shows the logging in effect for the CREATE FUNCTION privilege on the objects or users (or both) to which the rule applies. |
| AcrCreateMacro     | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | The AccLogRulesV.AcrCreateMacro field shows the logging in effect for the CREATE MACRO privilege on the objects or users (or both) to which the rule applies.       |
| AcrCreateTable     | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | The AccLogRulesV.AcrCreateTable field shows the logging in effect for the CREATE TABLE privilege on the objects or users (or both) to which the rule applies.       |
| AcrCreateUser      | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | The AccLogRulesV.AcrCreateUser field shows the logging in effect for the CREATE USER privilege on the objects or users (or both) to which the rule applies.         |
| AcrCreateView      | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | The AccLogRulesV.AcrCreateView field shows the logging in effect for the CREATE VIEW privilege on the objects or users (or both) to which the rule applies.         |
| AcrCreateProcedure | CHAR(3) LATIN<br>NOT                             | X(3)   | The AccLogRulesV.AcrCreateProcedure field shows the logging in effect for the CREATE  |

| View Column        | Data Type  | Format | Comment  |
|--------------------|--|--------|--|
|                    | CASESPECIFIC<br>NOT NULL                         |        | PROCEDURE privilege on the objects or users (or both) to which the rule applies.   |
| AcrCreExtProcedure | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | The AccLogRulesV.<br>AcrCreExtProcedure field shows the logging in effect for the CREATE EXTERNAL PROCEDURE privilege on the objects or users (or both) to which the rule applies. |
| AcrDelete          | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | The AccLogRulesV.AcrDelete field shows the logging in effect for the DELETE privilege on the objects or users (or both) to which the rule applies.                                 |
| AcrDropDatabase    | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | The AccLogRulesV.AcrDropDatabase field shows the logging in effect for the DROP DATABASE privilege on the objects or users (or both) to which the rule applies.                    |
| AcrDropFunction    | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | The AccLogRulesV.AcrDropDatabase field shows the logging in effect for the DROP FUNCTION privilege on the objects or users (or both) to which the rule applies.                    |
| AcrDropMacro       | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | The AccLogRulesV.AcrDropMacro field shows the logging in effect for the DROP MACRO privilege on the objects or users (or both) to which the rule applies.                          |
| AcrDropTable       | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | The AccLogRulesV.AcrDropTable field shows the logging in effect for the DROP TABLE privilege on the objects or users (or both) to which the rule applies.                          |
| AcrDropUser        | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | The AccLogRulesV.AcrDropUser field shows the logging in effect for the DROP USER privilege on the objects or users (or both) to which the rule applies.                            |
| AcrDropView        | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | The AccLogRulesV.AcrDropView field shows the logging in effect for the DROP VIEW privilege on the objects or users (or both) to which the rule applies.                            |
| AcrDropProcedure   | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | The AccLogRulesV.AcrDropProcedure field shows the logging in effect for the DROP PROCEDURE privilege on the objects or users (or both) to which the rule applies.                  |
| AcrDump            | CHAR(3) LATIN<br>NOT                             | X(3)   | The AccLogRulesV.AcrDump field shows the logging in effect for the DUMP  |

| View Column         | Data Type  | Format | Comment   |
|---------------------|--|--------|---|
|                     | CASESPECIFIC<br>NOT NULL                         |        | privilege on the objects or users (or both) to which the rule applies.  |
| AcrExecute          | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | The AccLogRulesV.AcrExecute field shows the logging in effect for the EXECUTE privilege on the objects or users (or both) to which the rule applies.                    |
| AcrExecuteFunction  | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | The AccLogRulesV.AcrExecuteFunction field shows the logging in effect for the EXECUTE FUNCTION privilege on the objects or users (or both) to which the rule applies.   |
| AcrExecuteProcedure | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | The AccLogRulesV.AcrExecuteProcedure field shows the logging in effect for the EXECUTE PROCEDURE privilege on the objects or users (or both) to which the rule applies. |
| AcrGrant            | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | The AccLogRulesV.AcrGrant field shows the logging in effect for the GRANT privilege on the objects or users (or both) to which the rule applies.                        |
| AcrIndex            | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | The AccLogRulesV.AcrIndex field shows the logging in effect for the CREATE/DROP INDEX privilege on the object(s) or users (or both) to which the rule applies.          |
| AcrInsert           | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | The AccLogRulesV.AcrInsert field shows the logging in effect for the INSERT privilege on the objects or users (or both) to which the rule applies.                      |
| AcrReference        | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | The AccLogRulesV.AcrReference field shows the logging in effect for the Reference privilege on the object(s) or users (or both) to which the rule applies.              |
| AcrRestore          | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | The AccLogRulesV.AcrRestore field shows the logging in effect for the RESTORE privilege on the objects or users (or both) to which the rule applies.                    |
| AcrSelect           | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | The AccLogRulesV.AcrSelect field shows the logging in effect for the SELECT privilege on the objects or users (or both) to which the rule applies.                      |
| AcrUpdate           | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | The AccLogRulesV.AcrUpdate field shows the logging in effect for the Update privilege on the object(s) and/or users to which the rule applies.                          |

| View Column          | Data Type  | Format | Comment   |
|----------------------|--|--------|---|
| AcrCreateTrigger     | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | The AccLogRulesV.AcrCreateTrigger field shows the logging in effect for the CREATE TRIGGER privilege on the objects or users (or both) to which the rule applies.               |
| AcrDropTrigger       | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | The AccLogRulesV.AcrDropTrigger field shows the logging in effect for the DROP TRIGGER privilege on the objects or users (or both) to which the rule applies.                   |
| AcrCreateRole        | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | Returns the logging in effect for the CREATE ROLE privilege.  |
| AcrDropRole          | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | Returns the logging in effect for the DROP ROLE privilege.  |
| AcrCreateProfile     | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | Returns the logging in effect for the CREATE PROFILE privilege.   |
| AcrDropProfile       | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | Returns the logging in effect for the DROP PROFILE privilege.   |
| AcrAlterProcedure    | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | The AccLogRulesV.AcrAlterProcedure field shows the logging in effect for the Alter Procedure privilege on the object(s) and/or users to which the rule applies.                 |
| AcrRepControl        | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | The AccLogRulesV.AcrRepControl field shows the logging in effect for the RepControl privilege on the object(s) and/or users to which the rule applies.                          |
| AcrAlterExtProcedure | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | The AccLogRulesV.AcrAlterExtProcedure field shows the logging in effect for the ALTER EXTERNAL PROCEDURE privilege on the objects or users (or both) to which the rule applies. |
| AcrUDTUsage          | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | Returns the logging privilege checks corresponding to privileges for UDTUsage.  |

| View Column          | Data Type   | Format              | Comment   |
|----------------------|---|---------------------|---|
| AcrUDTType           | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL        | X(3)                | Returns the logging privilege checks corresponding to privileges for UDTType.   |
| AcrUDTMethod         | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL        | X(3)                | Returns the logging privilege checks corresponding to privileges for UDTMethod.   |
| AcrCreAuthorization  | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL        | X(3)                | The AccLogRulesV.<br>AcrCreAuthorization field shows the logging in effect for the CREATE AUTHORIZATION privilege on the objects or users (or both) to which the rule applies.    |
| AcrDropAuthorization | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL        | X(3)                | The AccLogRulesV.<br>AcrDropAuthorization field shows the logging in effect for the DROP AUTHORIZATION privilege on the objects or users (or both) to which the rule applies.     |
| AcrStatistics        | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL        | X(3)                | The AccLogRulesV.AcrStatistics field shows the logging in effect for the Statistics privilege on the object(s) and /or users to which the rule applies.                           |
| AcrShow              | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL        | X(3)                | The AccLogRulesV.AcrShow field shows the logging in effect for the SHOW privilege on the objects or users (or both) to which the rule applies.                                    |
| AcrCreOwnerProcedure | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL        | X(3)                | The AccLogRulesV.<br>AcrCreOwnerProcedure field shows the logging in effect for the CREATE OWNER PROCEDURE privilege on the objects or users (or both) to which the rule applies. |
| AcrConnectThrough    | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL        | X(3)                | Returns the logging in effect for the CTControl privilege on the users to which the rule applies.   |
| CreatorName          | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)              | Table or database creator. For DBC. AllRights, the grantor of explicit rights, otherwise the user who executed the CREATE statement.  |
| CreateTimeStamp      | TIMESTAMP(0)  | YYYY-MM-DDBHH:MI:SS | Returns the date and time that the object in the row was created.   |

| View Column     | Data Type  | Format | Comment   |
|-----------------|--|--------|---|
| AcrCreateGLOP   | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | Returns the access logging rules for the CREATE GLOP privilege.   |
| AcrDropGLOP     | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | Returns the access logging rules for the DROP GLOP privilege.   |
| AcrGLOPMember   | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | Returns the access logging rules for the GLOP MEMBER privilege.   |
| AcrConstrDef    | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | The AccLogRulesV.AcrConstrDef field shows the logging in effect for the Constraint Definition privilege on the object(s) and/or users to which the rule applies.  |
| AcrConstrAsgn   | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | The AccLogRulesV.AcrConstrAsgn field shows the logging in effect for the Constraint Assignment privilege on the object(s) and/or users to which the rule applies. |
| AcrOverrideIns  | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | The AccLogRulesV.AcrOverrideIns field shows the logging in effect for the Override Insert privilege on the object(s) and/or users to which the rule applies.      |
| AcrOverrideSel  | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | The AccLogRulesV.AcrOverrideSel field shows the logging in effect for the Override Select privilege on the object(s) and/or users to which the rule applies.      |
| AcrOverrideUpd  | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | The AccLogRulesV.AcrOverrideUpd field shows the logging in effect for the Override Update privilege on the object(s) and/or users to which the rule applies.      |
| AcrOverrideDel  | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | The AccLogRulesV.AcrOverrideDel field shows the logging in effect for the Override Delete privilege on the object(s) and/or users to which the rule applies.      |
| AcrOverrideDump | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | The AccLogRulesV.AcrOverrideDump field shows the logging in effect for the Override Dump privilege on the object(s) and/or users to which the rule applies.       |

| View Column            | Data Type  | Format | Comment   |
|------------------------|--|--------|---|
| AcrOverrideRestore     | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | The AccLogRulesV.<br>AcrOverrideRestore field shows the logging in effect for the Override Restore privilege on the object(s) and/or users to which the rule applies. |
| AcrCreateZone          | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | The AccLogRulesV.AcrCreateZone field shows the logging in effect for the Create Zone privilege on the object(s) and/or users to which the rule applies.               |
| AcrDropZone            | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | The AccLogRulesV.AcrDropZone field shows the logging in effect for the Create Zone privilege on the object(s) and/or users to which the rule applies.                 |
| AcrZoneOverride        | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | The AccLogRulesV.AcrZoneOverride field shows the logging in effect for the Create Zone privilege on the object(s) and/or users to which the rule applies.             |
| AcrCreateDatasetSchema | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | Returns the logging in effect for the CREATE DATASET SCHEMA privilege.  |
| AcrDropDatasetSchema   | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | Returns the logging in effect for the DROP DATASET SCHEMA privilege.  |
| AcrWithDatasetSchema   | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | Returns the logging in effect for the WITH DATASET SCHEMA privilege.  |
| AcrCreateMap           | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | Returns the logging in effect for the CREATE MAP privilege.   |
| AcrDropMap             | CHAR(3) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(3)   | Returns the logging in effect for the DROP MAP privilege.   |

## Usage Notes

The underlying table of this view is populated only if the DBC.AccLogRule security macro is installed and the DBA or security administrator has executed one or more BEGIN LOGGING statements. For more information about this security macro, see *Teradata Vantage™ - Advanced SQL Engine Security Administration*, B035-1100.

To install the DBC.AccLogRule security macro, you must manually run the DIP script, DIPACC. For more information about the DIPACC script, see *Teradata Vantage™ - Database Utilities*, B035-1102.

Each row in the underlying table defines a rule controlling what privilege check is to be logged when a specific user attempts to access a specific object.

When a request is submitted that involves any of the rule criteria, the details of the involvement are recorded in the access log.

In AccLogRulesV, each Access Rule (Acr...) column is named for a particular privilege, which is also associated with an access action and a SQL statement. In each column, each character position represents the frequency with which checks performed on that privilege are to be logged, as follows:

1. Position 1 (every privilege check) indicates how often to log checks on this privilege when performed against any requests (submitted by a specified user) that attempt to access the specified object. Possible values that could appear in each position are as follows:

B = Both FIRST and LAST occurrences are to be logged.

E = Each occurrence is to be logged.

F = FIRST occurrence is to be logged.

L = LAST occurrence is to be logged.

Blank = No logging.

2. Position 2 indicates how often to log checks on this privilege when performed against requests (submitted by a specified user) that are not allowed to access the specified object (that is, check results are Denials).

B = Both FIRST and LAST occurrences are to be logged.

E = Each occurrence is to be logged.

F = FIRST occurrence is to be logged.

L = LAST occurrence is to be logged.

Blank = No logging.

3. Position 3 (save text of request) indicates whether to record the text of the requests that cause a check on this privilege.

- = Save text only for Denial entries.

+ = Save text for all entries.

= = Save text for all entries (specified in multiple BEGIN LOGGING statements).

Blank = No WITH TEXT option specified.

## Referenced Columns

Many of the Data Dictionary view columns have referenced table columns. That is, the value in the view column corresponds to a value in the selected column referenced in the table. It would be meaningful to join the view and the referenced table based on the selected column and the referenced column.

Referenced columns for this view are:

| View Column    | Referenced Column             |
|----------------|-------------------------------|
| UserName       | Dbase.DatabaseName            |
| DatabaseName   | Dbase.DatabaseName            |
| TVMName        | TVM.TVMName                   |
| ConstraintName | SecConstraints.ConstraintName |

## Example: Using AccLogRulesV

If the following statements are submitted, a SELECT statement retrieving the AccLogRules entries for User1 returns the rows as shown:

```
BEGIN LOGGING ON EACH CREATE TABLE BY Jones ON USER Jones ;
BEGIN LOGGING DENIALS WITH TEXT ON FIRST CREATE DATABASE
  BY Jones ON DATABASE Personnel ;
SELECT * FROM DBC.AccLogRulesV WHERE UserName = 'Jones' ;
```

Result:

| UserName | DatabaseName | TVMName | CPT | CDB | CMC | CTB | CUS |     |
|----------|--------------|---------|-----|-----|-----|-----|-----|-----|
| Jones    | Jones        | All     |     |     |     | E   |     | ... |
| Jones    | Personnel    | All     |     | F-  |     |     |     | ... |

- In the first row, the UserName “Jones”, the DatabaseName “Jones”, and the “E” in the first position of the CTB column indicate that a log entry is to be made each time a check for the CREATE TABLE privilege is performed in response to a request by Jones to create a table in his own space.
- In the second row, the UserName “Jones”, the DatabaseName “Personnel”, and the “F” in the second position of the CDB column indicate that a log entry is to be made the first time a check for a CREATE DATABASE privilege that results in a denial is performed in response to a request by Jones to create a database in the Personnel database. The “-” in the third position of the CDB column indicates that the text of the denied statement is to be saved in the log entry.

## AccountInfoV[X]

**Category:** Accounting

**Database:** DBC

| View Column   | Data Type   | Format | Comment  |
|---------------|---|--------|--|
| UserName      | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | Returns the name or profile of a user.   |
| AccountName   | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | Expanded account in effect when a request was submitted. SYSTEMACCOUNTNO tracks console utility activity such as table rebuild, Diskcopy, or Scandisk. |
| UserOrProfile | CHAR(7) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL         | X(7)   | Returns an indication whether the account name is for a user or a profile using the following values: User, Profile.                                   |

## Usage Notes

You can use this view to query the accounts the user can control.

## Example: Using AccountInfoV

The following SELECT statement returns the accounts the user controls:

```
SELECT * FROM DBC.AccountInfoV;
```

Result:

| Name         | AccountName  | UserOrProfile |
|--------------|--------------|---------------|
| -----        | -----        | -----         |
| DBC          | DBC          | User          |
| CONSOLE      | DBC          | User          |
| SystemFe     | SystemFe     | User          |
| Crashdumps   | Crashdumps   | User          |
| TDPUSER      | \$H          | User          |
| SysAdmin     | SysAdmin     | User          |
| Sys_Calendar | Sys_Calendar | User          |
| V2R5IN       | DBC          | User          |

## Related Topics

| For information about ...                                     | See ...   |
|---|---|
| controlling access, space, and ownership                      | <i>Teradata Vantage™ Database Design</i> , B035-1094. |
| the current account name (unexpanded) in effect for a session | <a href="#">SessionInfoV[X]</a> .                     |

| For information about ...   | See ...   |
|---|---|
| the default account name for a user or database                   | <a href="#">DatabasesV[X]</a> or <a href="#">UsersV</a> . |
| the default account name if set by profile for a user or database | <a href="#">ProfileInfoV[X]</a> .                         |

## ActionHistoryV[X]

**Category:** TDMaps

**Database:** TDMaps

| View Column    | Data Type                                   | Format                        | Comment  |
|----------------|---|-------------------------------|--|
| ZoneName       | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128)                        | Zone name  |
| JobNumber      | INTEGER                                     | -(10)9                        | Job number specified<br>by user calling<br>ManageMoveTablesSP,<br>which is optional                          |
| JobStartTime   | TIMESTAMP(6)<br>WITH TIME ZONE              | YYYY-MM-DDBHH:MI:<br>SS.S(6)Z | Starting time of<br>Mover Manager  |
| Action         | VARCHAR(24)<br>LATIN NOT<br>CASESPECIFIC    | X(24)                         | Type of action to perform<br>on the table  |
| Status         | VARCHAR(24)<br>LATIN NOT<br>CASESPECIFIC    | X(24)                         | Current status of<br>the action  |
| DatabaseName   | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128)                        | Database on which the<br>action applies  |
| TableName      | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128)                        | Table on which the<br>action applies   |
| GroupOrder     | DECIMAL(18,2)                               | --Z(15)9.99                   | Priority rank of an action<br>within its group. Applicable<br>only for rows whose Action<br>value is 'Alter' |
| SourceMap      | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128)                        | The map of the table prior<br>to performing the action   |
| DestinationMap | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128)                        | The target map used to<br>perform the action   |

| View Column            | Data Type  | Format                        | Comment   |
|------------------------|--|-------------------------------|---|
| ActionSQLText          | VARCHAR(10000)<br>UNICODE NOT<br>CASESPECIFIC    | X(10000)                      | The SQL text of the DDL statement that would perform this action  |
| ActionListName         | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC      | X(128)                        | Named list of actions of which this action is a member of   |
| TableSize              | BIGINT   | -(19)9                        | Approximate size of table in bytes  |
| FractionOfPermDBFree   | FLOAT  | -9.<br>9999999999999999E-999  | Fraction of PERM DB free  |
| Issues                 | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(1)                          | 'Y' or 'N'. If 'Y', it indicates there is something that administrator may want to check                |
| PrevGroupsBytesPerSec  | DECIMAL(38,6)                                    | --(32).9(6)                   | Previous Group bytes per second   |
| EstElapsedTimeInSecond | DECIMAL(38,6)                                    | --(32).9(6)                   | Estimated elapsed time for executing action   |
| ElapsedTime            | DECIMAL(38,6)                                    | --(32).9(6)                   | Actual elapsed time for executing action  |
| StartTime              | TIMESTAMP(6)<br>WITH TIME ZONE                   | YYYY-MM-DDBHH:MI:<br>SS.S(6)Z | Action start timestamp  |
| EndTime                | TIMESTAMP(6)<br>WITH TIME ZONE                   | YYYY-MM-DDBHH:MI:<br>SS.S(6)Z | Action end timestamp  |
| ErrorCode              | INTEGER  | --,---,---,-9                 | DBS Error code if failure occurred. Otherwise, zero or NULL   |
| RunUID                 | BYTE(4)  | X(8)                          | User who executed this action   |
| ActionsTblDescription  | VARCHAR(1024)<br>UNICODE NOT<br>CASESPECIFIC     | X(1024)                       | Advisor generated comment providing further explanation for the recommended action                      |
| Description            | VARCHAR(1024)<br>UNICODE NOT<br>CASESPECIFIC     | X(1024)                       | Additional info describing the completed or failed action   |
| WorkerId               | INTEGER  | -(10)9                        | The ID of the worker who performs action on the table. 0 for serial worker. 1,2... for parallel workers |

| View Column | Data Type | Format | Comment   |
|-------------|-----------|--------|---|
| HostId      | INTEGER   | -(10)9 | Host Id of the worker who performs action on the table    |
| SessionId   | INTEGER   | -(10)9 | Session Id of the worker who performs action on the table |

## Usage Notes

Rows in the ActionHistoryV[X] views represent map related actions that are currently running or have completed. The ActionHistoryTbl is updated by the mover procedure. After the mover takes an action from the ActionsTbl, it inserts a row into ActionHistoryTbl and sets its status to In Progress. After the action completes, the mover updates the status to Complete. These views are security zone constrained. ActionHistoryVX returns only the objects to which the user has access.

### ActionListName

ActionListName contains a named list of actions of which this action is a member. See related Advisor procedure parameter OutputActionList and Mover procedure parameter TableActionList.

### FractionOfPermDBFree

The fraction of free permanent space the table takes in the database. ALTER TABLE MAP takes twice the space during the operation, and this value indicates the fraction of free space that will be used.

### PrevGroupsBytesPerSec

PrevGroupsBytesPerSec indicates how many bytes per second were processed in the previous group of tables.

### TableSize

TableSize is the approximate size of table in bytes after the action completed.

### HostId, SessionId, and WorkerId

The HostId is the host ID of the worker that wrote the row. The SessionId is the session ID of the worker that wrote the row.

The WorkerId is the ID of the worker that wrote the row:

| Value       | Description   |
|-------------|---|
| 0           | 0 is the value of WorkerId for a serial move.                             |
| 1 and above | 1 and above indicates the number of parallel workers for a parallel move. |

## Possible Values for Action

Type of action to perform on the table.

| Value   | Description  |
|---------|--|
| Alter   | Move existing table data to new destination map.<br>For example, use the ALTER statement to change the Action: <pre>ALTER {TABLE   JOIN INDEX   HASH INDEX} &lt;TableName&gt; MAP = &lt;DestinationMap&gt; [COLOCATE USING [&lt;ColocateName&gt;];</pre> |
| Exclude | Do not perform any action on this table for the specified destination map.   |

## Possible Values for Issues

| Value | Description   |
|-------|---|
| Y     | Y indicates there are issues with this action. For details about what the issues are, see the ActionsTblDescription column. |
| N     | N indicates there are no issues.  |

## Possible Values for Status

Type of action to perform on the table.

| Value      | Description  |
|------------|--|
| Aborted    | The action was aborted by a restart or abort.  |
| Complete   | The action was successfully completed.   |
| InProgress | The action is currently running, was aborted, or a database restart occurred during the move.  |
| Failed     | The action failed. See related column ErrorCode for details.   |
| Stopped    | An already queued action was stopped prior to executing.   |
| Skipped    | The action was skipped and not started because its estimated elapsed time was greater than the time remaining in the caller's specified TimeLimit. If TableName is non-NULL, the individual table was skipped. If Tablename is NULL and GroupOrder is non-NULL, all of the tables within the specified group were skipped. |

## ActionsV[X]

**Category:** TDMaps

**Database:** TDMaps

| View Column    | Data Type   | Format      | Comment   |
|----------------|---|-------------|---|
| ZoneName       | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC             | X(128)      | Zone name   |
| Action         | VARCHAR(24) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL    | X(24)       | Type of action to perform<br>on the table   |
| Origin         | CHAR(1) LATIN NOT<br>CASESPECIFIC                       | X(1)        | Source of this action.<br>'A' indicates Advisor<br>procedures. 'U' or<br>NULL indicates a user<br>inserted action.                    |
| DatabaseName   | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)      | Database on which the<br>action applies   |
| TableName      | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)      | Table, Join Index, or<br>Hash Index on which the<br>action applies  |
| SourceMap      | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC             | X(128)      | The current map of the<br>table prior to performing<br>the recommended action   |
| DestinationMap | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC             | X(128)      | The target map for<br>performing the specified<br>action against  |
| CoLocateName   | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC             | X(128)      | Advisor recommended<br>qualified name for<br>COLOCATE USING<br>clause. Applicable only<br>when DestinationMap is<br>defined as SPARSE |
| ActionSQLText  | VARCHAR(10000)<br>UNICODE NOT<br>CASESPECIFIC           | X(10000)    | The SQL text of the<br>DDL statement that<br>would perform this<br>action. This column is<br>informational only                       |
| ActionListName | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)      | Named list of actions of<br>which this action is a<br>member of   |
| GroupOrder     | DECIMAL(18,2)   | --Z(15)9.99 | Common identifier and<br>priority rank of the group<br>in which this action is<br>a member  |

| View Column           | Data Type                               | Format                    | Comment  |
|-----------------------|---|---------------------------|--|
| ActionOrder           | DECIMAL(10,2)                           | zz9.99                    | Priority rank of an action within its group. Applicable only for rows whose Action value is 'Alter'  |
| TableSize             | BIGINT                                  | -(19)9                    | Advisor computed size of the table in bytes at the time of analysis  |
| FractionOfPermDBFree  | FLOAT                                   | -9.999999999999999E-999   | fraction of free database space available for the table  |
| UDICnt                | BIGINT                                  | -(19)9                    | Number of rows Updated, Deleted, or Inserted on this the table by logged SQL statements during the period analyzed                                 |
| QueryCnt              | BIGINT                                  | -(19)9                    | Number of logged DML SQL requests referencing the table during the period analyzed   |
| PKJoinCnt             | BIGINT                                  | -(19)9                    | Number of join steps on this table requiring no data redistribution as recorded by logged DML statements during the period analyzed by the Advisor |
| LastAlterTimeStamp    | TIMESTAMP(6) WITH TIME ZONE NOT NULL    | YYYY-MM-DDBHH:MI:SS.S(6)Z | Last modification time on the metadata describing this action  |
| LastAlterUID          | BYTE(4) NOT NULL                        | X(8)                      | User who last modified the metadata for this action  |
| Issues                | CHAR(1) LATIN NOT CASESPECIFIC NOT NULL | X(1)                      | 'Y' or 'N'. If 'Y', it indicates there is something that administrator may want to check   |
| ActionsTblDescription | VARCHAR(1024) UNICODE NOT CASESPECIFIC  | X(1024)                   | Advisor generated comment providing further explanation for the recommended action   |

## Usage Notes

The ActionsV[X] views are security zone constrained views that contain recommended map actions, such as move a table to a new map. The mover procedure reads the actions from this table and schedules them for execution. Users can manually override actions in ActionsTbl, using SQL statements, such as UPDATE, DELETE, and INSERT.

When executing DML statements directly against ActionsTbl, users must include a WHERE condition on column ActionListName to limit the changes to a designated action list.

ActionsVX returns only the objects to which the user has access.

## Inserting Rows in ActionsTbl

Each row inserted into ActionsTbl must specify a non-null value for the following columns which represent the required metadata for defining a map related action: Action, DatabaseName, TableName, DestinationMap, ActionListName, LastAlterTimeStamp, LastAlterUID, and Issues. These columns are automatically populated. The NOT NULL constraint defined for each of these columns ensures that manual INSERTs performed on this table populate these columns as well. Although it is recommended that manual UPDATES performed on ActionsTbl update the LastAlterTimeStamp and LastAlterUID columns, the system does not enforce this.

## ActionListName

Named list of actions of which this action is a member. See the related Advisor procedure parameter OutputActionList and see the Mover procedure.

## FractionOfPermDBFree

The fraction of free permanent space the table takes in the database. ALTER TABLE MAP takes twice the space during the operation, and this value indicates the fraction of free space that will be used.

## GroupOrder and ActionOrder

GroupOrder is a common identifier and priority rank of the group in which this action is a member. ActionOrder is a priority rank of an action within its group. These columns are applicable only for rows whose Action value is Alter.

When determining the priority execution order for Alter actions in ActionsTbl, values in column GroupOrder are compared to establish a ranking between groups and then ActionOrder column values are compared to establish a ranking within each group. Because they represent ranks, lower values have a higher priority. GroupOrder and ActionOrder values generated by the AnalyzeSP procedure are assigned consecutive positive integer values starting with 1. Users that wish to manually add a new group between existing ones can use the fractional value portion of the GroupOrder column. Similarly, users wishing to add new actions to an existing group can use the fractional portion of the ActionOrder column. When comparing two values for ordering, the fractional portion is only applicable when the integer portions are equal. In addition, only the digits in the fractional portion are considered when ranking; the decimal point has no bearing. A GroupOrder value of NULL designates the action is not part of any group and is assigned the lowest

possible group ranking. An ActionOrder value of NULL designates the lowest possible ranking within its group. An action with NULL specified for both GroupOrder and ActionOrder is assigned the lowest possible ranking among all actions.

## UDICnt

The number of rows updated, deleted, or inserted on this the table by logged SQL statements during the period analyzed. Indicator of the volatility for the table.

## Possible Values for Action

Type of action to perform on the table.

| Value   | Description  |
|---------|--|
| Alter   | Move existing table data to new destination map.<br>For example, use the ALTER statement to change the Action: <pre>ALTER {TABLE   JOIN INDEX   HASH INDEX} &lt;TableName&gt; MAP = &lt;DestinationMap&gt; [COLOCATE USING [&lt;ColocateName&gt;];</pre> |
| Exclude | Do not perform any action on this table for the specified destination map.   |

## Possible Values for Issues

| Value | Description   |
|-------|---|
| Y     | Y indicates there are issues with this action. For details about what the issues are, see the ActionsTblDescription column. |
| N     | N indicates there are no issues.  |

## Possible Values for Issues and ActionsTblDescription

| Issues Value | ActionsTblDescription Value  |
|--------------|--|
| N            | Table is considered to be empty, and is not allowed to be put in a sparse map.                           |
| N            | Table is in the exclusion list.  |
| N            | Move table for system expansion.<br>The target map has more AMPs than the map the table is currently in. |
| N            | Destination map has less or the same number of AMPs as the source map.                                   |
| N            | Table is very small. One-AMP sparse map is recommended.  |
| N            | Small tables are optimized by moving to sparse map.  |
| N            | Table is too big for a sparse map.   |
| N            | Small tables may not be optimized in a contiguous map.   |

| Issues Value | ActionsTblDescription Value  |
|--------------|--|
| N            | Moving to bigger sparse map.   |
| Y            | Moving to smaller sparse map.  |
| Y            | Source and destination maps are the same.<br>A table is marked Exclude if the source and target maps are the same.                       |
| Y            | Journal will be invalidated after ALTER TABLE.   |
| Y            | Join Index will be invalidated after ALTER TABLE.<br>One or more join indexes will be invalidated after the table is moved to a new map. |
| Y            | Hash Index will be invalidated after ALTER TABLE.<br>One or more hash indexes will be invalidated after the table is moved to a new map. |
| Y            | Not enough free perm space.<br>There is not enough free space to move the table to a new map.  |

### Possible Values for Origin

| Value | Description   |
|-------|---|
| A     | A indicates that the Advisor procedures are the source of the action to perform.      |
| U     | U or NULL indicates a user inserted this action (the source of the action is a user). |

## Example: Updating ActionsTbl Row

The example shows updating a row in ActionsTbl, in which the user changes the destination map to TD\_Map3:

```
UPDATE ActionsTbl
  SET ActionSQLText = 'ALTER TABLE FSK.customer, MAP=TD_Map3',
      DestinationMap = 'TD_Map3',
      Origin = 'U'
 WHERE ActionListName = 'fskobject'
    AND DatabaseName = 'fsk'
    AND TableName = 'customer';
```

## All\_RI\_ChildrenV[X]

**Category:** Database

**Database:** DBC

| View Column       | Data Type   | Format              | Comment  |
|-------------------|---|---------------------|--|
| IndexID           | SMALLINT NOT NULL                                       | ---,--9             | Returns the reference index number.<br>Note: This is not the same as IndexNumber.  |
| IndexName         | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC             | X(128)              | Returns the name of the reference index.   |
| ChildDB           | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)              | Returns the name of the referencing database.  |
| ChildTable        | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)              | Returns the name of the referencing table.   |
| ChildKeyColumn    | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)              | Returns the name of a column in the referencing key.   |
| ParentDB          | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)              | Returns the name of the referenced database.   |
| ParentTable       | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)              | Returns the name of the referenced table.  |
| ParentKeyColumn   | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)              | Returns the name of the column in a referenced key.  |
| InconsistencyFlag | CHAR(1) LATIN<br>UPPERCASE NOT NULL                     | X(1)                | Inconsistencies allowed between related parent-child object definitions after restore: Y (yes), N (no).                              |
| CreatorName       | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)              | Table or database creator. For DBC. AllRights, the grantor of explicit rights, otherwise the user who executed the CREATE statement. |
| CreateTimeStamp   | TIMESTAMP(0)  | YYYY-MM-DDBHH:MI:SS | Returns the date and time that the object in the row was created.  |

## Usage Notes

The All\_RI\_ChildrenV[X] view is designed for use in a SELECT statement with a WHERE clause to narrow the selection criteria.

The All\_RI\_ChildrenV[X] view is similar to the RI\_Child\_TablesV[X] view but contains the database, table, and column names instead of the IDs for access control purposes. The administrator can control who has access to internal ID numbers by limiting the access to the RI\_Child\_TablesV[X] view while allowing more (or all) users to access the names via the All\_RI\_ChildrenV[X] view.

## Corresponding Tables

The X view references these additional tables:

- DBC.AccessRights
- DBC.Owners
- DBC.Roles
- DBC.RoleGrants

## InconsistencyFlag

If the value in the InconsistencyFlag column is Y, it may be possible to validate the reference indexes that have been marked as inconsistent.

## All\_RI\_ParentsV[X]

**Category:** Database

**Database:** DBC

| View Column | Data Type   | Format  | Comment   |
|-------------|---|---------|---|
| IndexID     | SMALLINT NOT NULL                                       | ---,--9 | Returns the reference index number.<br>Note: This is not the same as IndexNumber. |
| IndexName   | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC             | X(128)  | Returns the name of the reference index.  |
| ParentDB    | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)  | Returns the name of the referenced database.                                      |
| ParentTable | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)  | Returns the name of the referenced table.   |

| View Column       | Data Type   | Format                  | Comment  |
|-------------------|---|-------------------------|--|
| ParentKeyColumn   | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                  | Returns the name of the column in a referenced key.  |
| ChildDB           | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                  | Returns the name of the referencing database.  |
| ChildTable        | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                  | Returns the name of the referencing table.   |
| ChildKeyColumn    | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                  | Returns the name of a column in the referencing key.   |
| InconsistencyFlag | CHAR(1) LATIN<br>UPPERCASE NOT NULL                     | X(1)                    | Inconsistencies allowed between related parent-child object definitions after restore: Y (yes), N (no).                              |
| CreatorName       | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                  | Table or database creator. For DBC. AllRights, the grantor of explicit rights, otherwise the user who executed the CREATE statement. |
| CreateTimeStamp   | TIMESTAMP(0)  | YYYY-MM-DD<br>BHH:MI:SS | Returns the date and time that the object in the row was created.  |

## Usage Notes

The All\_RI\_ParentsV[X] view is designed for use in a SELECT statement with a WHERE clause to narrow the selection criteria. This view is similar to the RI\_Parent\_TablesV[X] view, but contains the database, table, and column names instead of the IDs for access control purposes.

The administrator can control who has access to internal ID numbers by limiting the access to the RI\_Parent\_TablesV[X] view while allowing more (or all) users to access the names via the All\_RI\_ParentsV[X] view.

## Corresponding Tables

The X view references these additional tables:

- DBC.AccessRights
- DBC.Owners
- DBC.Roles

- DBC.RoleGrants

### InconsistencyFlag

If the value in the InconsistencyFlag column is Y, it may be possible to validate the reference indexes that have been marked as inconsistent.

## AllGlobalSpaceV[X]

**Category:** Accounting

**Database:** DBC

| View Column            | Data Type   | Format                         | Comment  |
|------------------------|---|--------------------------------|--|
| DatabaseName           | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                         | Returns the name of a database or one of the special system keywords: ALL / DEFAULT / PUBLIC.  |
| AccountName            | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                         | Expanded account in effect when a request was submitted. SYSTEMACCOUNTNO tracks console utility activity such as table rebuild, Diskcopy, or Scandisk. |
| TableName              | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                         | The TableName field identifies a table within a database.  |
| MaxPerm                | BIGINT<br>NOT NULL                                      | --,---,---,---,<br>---,---,--9 | Returns the maximum allocation in bytes for permanent and user defined temporary table space.  |
| MaxSpool               | BIGINT<br>NOT NULL                                      | --,---,---,---,<br>---,---,--9 | Returns the maximum allocation in bytes for spool and system defined temporary table space.  |
| MaxTemp                | BIGINT<br>NOT NULL                                      | --,---,---,---,<br>---,---,--9 | Returns the maximum allocation in bytes for temporary table space.   |
| CurrentPerm            | BIGINT  | --,---,---,---,<br>---,---,--9 | Returns the maximum used in bytes for permanent and user defined temporary table space.  |
| CurrentSpool           | BIGINT  | --,---,---,---,<br>---,---,--9 | Returns the maximum used in bytes for spool and user defined temporary table space.  |
| CurrentPersistentSpool | BIGINT  | --,---,---,---,<br>---,---,--9 | Returns the persistent spool space in bytes currently in use.  |
| CurrentTemp            | BIGINT  | --,---,---,---,<br>---,---,--9 | Returns the maximum used in bytes for temporary table space.   |

| View Column         | Data Type          | Format                         | Comment  |
|---------------------|--------------------|--------------------------------|--|
| PeakPerm            | BIGINT             | --,---,---,---,<br>---,---,--9 | The peak number of bytes of permanent space allocated for a given user/database across system since the time the field was last initialized.                   |
| PeakSpool           | BIGINT             | --,---,---,---,<br>---,---,--9 | The peak number of bytes of spool space allocated to the user across the system since the time the field was last initialized.                                 |
| PeakPersistentSpool | BIGINT             | --,---,---,---,<br>---,---,--9 | The peak number of bytes of temporary space allocated to a given user across the system since the time the field was last initialized.                         |
| PeakTemp            | BIGINT             | --,---,---,---,<br>---,---,--9 | Returns the peak number of bytes used at one time by a temporary table per vproc.  |
| MaxProfileSpool     | BIGINT             | --,---,---,---,<br>---,---,--9 | Returns the profile SPOOL space limit for the user if the user is assigned a profile which has a SPOOL space setting. Otherwise, this column has a null value. |
| MaxProfileTemp      | BIGINT             | --,---,---,---,<br>---,---,--9 | Returns the profile TEMP space limit for the user if the user is assigned a profile which has a TEMP space setting. Otherwise, this column has a null value.   |
| AllocatedPerm       | BIGINT<br>NOT NULL | --,---,---,---,<br>---,---,--9 | Returns the current total allocation in bytes for permanent and user defined temporary table space.  |
| AllocatedSpool      | BIGINT<br>NOT NULL | --,---,---,---,<br>---,---,--9 | Returns the current total allocation in bytes for spool and other system defined temporary table space.  |
| AllocatedTemp       | BIGINT<br>NOT NULL | --,---,---,---,<br>---,---,--9 | Returns the current total allocation in bytes for temporary tables space.  |
| PermSkew            | SMALLINT           | ---,--9                        | Returns the permissible skew limit percent for permanent space usage at the AMP level.   |
| SpoolSkew           | SMALLINT           | ---,--9                        | Returns the permissible skew limit percent for spool space usage at the AMP level.   |
| TempSkew            | SMALLINT           | ---,--9                        | Returns the permissible skew limit percent for temporary space usage at the AMP level.   |

## Usage Notes

AllGlobalSpaceV returns the space limits for a given database or user at the system level. It returns the databases and tables along with the actual current space usage and maximum permissible space usage. The difference between this view and the corresponding AllSpaceV[X] is that this view returns global level information on space usages while AllSpaceV[X] provides space usage information at the AMP level.

You can use the DBC.ClearPeakDisk macro to reset PeakPerm, PeakSpool, PeakPersistentSpool, and PeakTemp.

### AppProxyUser and TrustUserName

These columns are not returned in the X or VX views.

## AllRightsV[X]

**Category:** Accounting

**Database:** DBC

| View Column    | Data Type   | Format | Comment   |
|----------------|---|--------|---|
| UserName       | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | Returns the name of a user to whom the privilege was granted.   |
| DatabaseName   | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | Returns the name of the database on which a privilege was granted. The possible values are: ALL/PUBLIC.   |
| TableName      | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | Returns the name of a table, view, stored procedure, trigger, macro, user-defined types, user-defined methods, or user-defined function on which a privilege was granted. |
| ColumnName     | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | Returns the column name to which a privilege has been granted.  |
| AccessRight    | CHAR(2) LATIN<br>UPPERCASE<br>NOT NULL                  | X(2)   | Returns a code that identifies a privilege granted on the object.   |
| GrantAuthority | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL                  | X(1)   | Returns the WITH GRANT OPTION attribute of the access right held by the user.   |

| View Column     | Data Type                                   | Format              | Comment  |
|-----------------|---|---------------------|--|
| GrantorName     | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC | X(128)              | Returns the name of the grantor who granted the privilege.   |
| AllnessFlag     | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL      | X(1)                | Returns Y (yes) or N (no) to indicate whether or not the privilege was granted to all subordinate users, or to all users who are owned by the grantee. |
| CreatorName     | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC | X(128)              | Table or database creator. For DBC.AllRights, the grantor of explicit rights, otherwise the user who executed the CREATE statement.                    |
| CreateTimeStamp | TIMESTAMP(0)                                | YYYY-MM-DDBHH:MI:SS | Returns the date and time that the object in the row was created.  |

## Usage Notes

The AllRights view does not contain information about implicit privileges for a user, only explicit privileges granted on the object. For information about the types of explicit privileges, see "AccessRight Column."

## Corresponding Tables

The X view references these additional tables:

- DBC.Owners
- DBC.Roles
- DBC.RoleGrants

## Example: Using AllRightsV

The following SELECT statement displays the privileges user Jones has on tables:

```
SELECT Tablename, AccessRight, GrantorName
      FROM DBC.AllRightsV WHERE UserName = 'Jones';
```

Result:

| TableName | AccessRight | GrantorName |
|-----------|-------------|-------------|
| -----     | -----       | -----       |
| project   | RS          | SYSTEMAD    |
| project   | DP          | SYSTEMAD    |
| project   | DT          | SYSTEMAD    |
| project   | D           | SYSTEMAD    |

|          |   |          |
|----------|---|----------|
| project  | I | SYSTEMAD |
| project  | U | SYSTEMAD |
| project  | R | SYSTEMAD |
| employee | I | SYSTEMAD |
| employee | U | SYSTEMAD |

## AllRoleRightsV

**Category:** Security

**Database:** DBC

| View Column     | Data Type  | Format                  | Comment   |
|-----------------|--|-------------------------|---|
| RoleName        | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128)                  | Returns the name of a role.                                       |
| DatabaseName    | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128)                  | Returns the name of a database.                                   |
| TableName       | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128)                  | Returns the name of a table, join index, or hash index.           |
| ColumnName      | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128)                  | Returns the name of the column or parameter.                      |
| AccessRight     | CHAR(2) LATIN<br>UPPERCASE NOT NULL                  | X(2)                    | Returns a code that identifies a privilege granted on the object. |
| GrantorName     | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC             | X(128)                  | Returns the name of the grantor who granted the privilege.        |
| CreateTimeStamp | TIMESTAMP(0)   | YYYY-MM-DD<br>BHH:MI:SS | Returns the date and time that the object in the row was created. |

## Usage Notes

The AllRoleRightsV view is similar to the AllRightsV view except that it does not include the following columns:

- AllnessFlag
- CreatorName column

For information about the possible values for the AccessRight column, see "AccessRight Column."

## Corresponding Tables

The AllRoleRightsV view also references the DBC.Owners table.

## Example: Using AllRoleRightsV

The following SELECT statement returns all the privileges granted to each role:

```
SELECT CAST(RoleName as CHAR(16)) as RoleName,
CAST(DatabaseName as CHAR(15)) as Databases,
CAST(TableName as CHAR(15)) as TVMs,
CAST(AccessRight as CHAR(5)) as AccRights,
CAST(GrantorName as CHAR(15)) as Grantor
FROM DBC.AllRoleRightsV
WHERE RoleName like 'roles017%'
ORDER BY 1,2,3,5;
```

Result:

| RoleName         | Databases     | TVMs          | AccRights | Grantor    |
|------------------|---------------|---------------|-----------|------------|
| -----            | -----         | -----         | -----     | -----      |
| roles017_dbc_r1b | roles017_3_db | roles017_3_m3 | E         | roles017_3 |
| roles017_r1a     | roles017_3_db | roles017_3_t1 | R         | roles017_3 |
| roles017_r1c     | roles017_3_db | roles017_3_t4 | R         | monthly    |
| roles017_r1f     | roles017_3_db | roles017_3_v2 | R         | roles017_3 |
| roles017_r2d     | roles017_3_db | roles017_3_v2 | R         | roles017_3 |
| roles017_r2e     | roles017_3_db | roles017_3_t5 | R         | monthly    |

## AllSpaceV[X]

**Category:** Accounting

**Database:** DBC

| View Column  | Data Type   | Format         | Comment   |
|--------------|---|----------------|---|
| Vproc        | INTEGER<br>NOT NULL                                     | --,---,---,--9 | Identifies the virtual processor for which an event was logged.                                       |
| DatabaseName | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)         | Returns the name of a database or one of the following special system keywords: ALL, DEFAULT, PUBLIC. |
| AccountName  | VARCHAR(128)<br>UNICODE NOT                             | X(128)         | Expanded account in effect when a request was submitted. SYSTEMACCOUNTNO                              |

| View Column            | Data Type   | Format                         | Comment   |
|------------------------|---|--------------------------------|---|
|                        | CASESPECIFIC<br>NOT NULL                                |                                | tracks console utility activity such as table rebuild, Diskcopy, or Scandisk.   |
| TableName              | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                         | Returns the name of a table, join index, or hash index.   |
| MaxPerm                | BIGINT  | --,---,---,---,<br>---,---,--9 | Returns an integer that represents the maximum permanent space, in bytes, that is allocated to the database on a specified AMP (or on all AMPs if the SUM aggregate is specified).                    |
| MaxSpool               | BIGINT  | --,---,---,---,<br>---,---,--9 | Returns an integer that represents the maximum spool space, in bytes, that is allocated to the database on a specified AMP (or on all AMPs if the SUM aggregate is specified).                        |
| MaxTemp                | BIGINT  | --,---,---,---,<br>---,---,--9 | Returns an integer that represents the maximum temporary space, in bytes, that is allocated to the database on a specified AMP (or on all AMPs if the SUM aggregate is specified).                    |
| CurrentPerm            | BIGINT  | --,---,---,---,<br>---,---,--9 | This field gives the amount of permanent space per AMP currently being used by the database or table.   |
| CurrentSpool           | BIGINT  | --,---,---,---,<br>---,---,--9 | This field gives the amount of spool space per AMP currently being used by the user.  |
| CurrentPersistentSpool | BIGINT  | --,---,---,---,<br>---,---,--9 | This field gives the amount of persistent spool space per AMP currently being used by the user.   |
| CurrentTemp            | BIGINT  | --,---,---,---,<br>---,---,--9 | Returns the number of bytes currently used by a temporary table per vproc.  |
| PeakPerm               | BIGINT  | --,---,---,---,<br>---,---,--9 | Returns the maximum amount of permanent space per AMP that has been used by the database since the last time the DBC.ClearPeakDisk macro was run.   |
| PeakSpool              | BIGINT  | --,---,---,---,<br>---,---,--9 | Returns an integer that represents the maximum spool space, in bytes, that was used at any one time by the database on a specified AMP (or on all AMPs if the SUM aggregate is specified).            |
| PeakPersistentSpool    | BIGINT  | --,---,---,---,<br>---,---,--9 | Returns an integer that represents the maximum persistent spool space, in bytes, that was used at any one time by the database on a specified AMP (or on all AMPs if the SUM aggregate is specified). |

| View Column     | Data Type | Format                         | Comment  |
|-----------------|-----------|--------------------------------|--|
| PeakTemp        | BIGINT    | --,---,---,---,<br>---,---,--9 | Returns the peak number of bytes used at one time by a temporary table per vproc.  |
| MaxProfileSpool | BIGINT    | --,---,---,---,<br>---,---,--9 | Returns the profile SPOOL space limit per AMP for the user if the user is assigned a profile which has a SPOOL space setting. Otherwise, this column has a null value.         |
| MaxProfileTemp  | BIGINT    | --,---,---,---,<br>---,---,--9 | Returns the profile TEMPORARY space limit per AMP for the user if the user is assigned a profile which has a TEMPORARY space setting. Otherwise, this column has a null value. |
| AllocatedPerm   | BIGINT    | --,---,---,---,<br>---,---,--9 | Returns the current total allocation in bytes for permanent and user defined temporary table space.  |
| AllocatedSpool  | BIGINT    | --,---,---,---,<br>---,---,--9 | Returns the current total allocation in bytes for spool and other system defined temporary table space.  |
| AllocatedTemp   | BIGINT    | --,---,---,---,<br>---,---,--9 | Returns the current total allocation in bytes for temporary tables space.  |
| PermSkew        | SMALLINT  | ---,--9                        | Returns the permissible skew limit percent for permanent space usage at the AMP level.   |
| SpoolSkew       | SMALLINT  | ---,--9                        | Returns the permissible skew limit percent for spool space usage at the AMP level.   |
| TempSkew        | SMALLINT  | ---,--9                        | Returns the permissible skew limit percent for temporary space usage at the AMP level.   |

## Usage Notes

When a database, user, or table is created, allocated disk space is divided evenly among all AMPs. The AllSpace view returns one row of usage information for each AMP in the system configuration (or for all AMPs if the SUM aggregate is used).

When a database is created, a space row is added to each AMP, with the processor field in each row initialized to 0. The first time the space row is updated (such as when a table is created in the database, or when the system is restarted), the processor field in each row is updated to indicate the actual processor number.

When a query applies a SUM aggregate to either the MaxPerm or CurrentPerm column without a WHERE clause, or with a WHERE clause that references only one TableName or DatabaseName, the returned values are double the desired result.

For example, the following query, which returns the correct amount of space allocated to Peterson, also returns twice the amount of space currently being used by Peterson (see [DiskSpaceV\[X\]](#) and [TableSizeV\[X\]](#)).

```
SELECT SUM(MaxPerm), SUM(CurrentPerm)
FROM DBC.AllSpaceV
WHERE DatabaseName = 'Peterson';
```

You can use the DBC.ClearPeakDisk macro to reset PeakPerm, PeakSpool, PeakPersistentSpool, and PeakTemp.

## Corresponding Tables

The X view references these additional tables:

- DBC.AccessRights
- DBC.Owners
- DBC.Roles
- DBC.RoleGrants

## AppProxyUser and TrustUserName

These columns are not returned in the X or VX views.

## Example: Using AllSpaceV

The following SELECT statement displays how the space currently used by the data table named Department is distributed on each AMP:

```
SELECT DatabaseName,TableName,AMP,CurrentPerm FROM DBC.AllSpaceV
WHERE TableName='Department' ORDER BY 1,2,3 ;
```

Result:

| DatabaseName | TableName  | AMP  | CurrentPerm |
|--------------|------------|------|-------------|
| -----        | -----      | ---- | -----       |
| Test         | DEPARTMENT | 1-0  | 1,024       |
| Test         | DEPARTMENT | 1-1  | 512         |
| Test         | DEPARTMENT | 1-2  | 1,024       |
| Test         | DEPARTMENT | 1-3  | 512         |
| PERSONNEL    | department | 1-0  | 2,048       |
| PERSONNEL    | department | 1-1  | 1,536       |
| PERSONNEL    | department | 1-2  | 1,536       |
| PERSONNEL    | department | 1-3  | 1,536       |
| User1        | department | 1-0  | 2,048       |

|       |            |     |       |
|-------|------------|-----|-------|
| User1 | department | 1-1 | 1,536 |
| User1 | department | 1-2 | 1,536 |
| User1 | department | 1-3 | 1,536 |

## AllTempTablesV[X]

**Category:** Operations

**Database:** DBC

| View Column    | Data Type   | Format         | Comment  |
|----------------|---|----------------|--|
| HostNo         | SMALLINT NOT NULL                                       | ---,--9        | Returns the number of the client system through which the user logged on to the Teradata Database.         |
| SessionNo      | INTEGER NOT NULL  | --,---,---,--9 | Returns the session identifier assigned to the session by the TDP or LAN interface.                        |
| UserName       | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)         | Returns the name or profile of a user.   |
| B_DatabaseName | VARCHAR(128)<br>UNICODE UPPERCASE<br>NOT NULL           | X(128)         | Returns the name of the database in which the base global table resides.                                   |
| B_TableName    | VARCHAR(128)<br>UNICODE UPPERCASE<br>NOT NULL           | X(128)         | Returns the name of the base global temporary table.   |
| E_TableId      | BYTE(6) NOT NULL  | X(12)          | Returns the identifier for the materialized (effective) temporary table for a base global temporary table. |

## Example: Using AllTempTablesVX

After a global temporary table definition is created, you can use the INSERT statement to create a local instance of the global temporary table for use during the session.

The following statement shows all temporary tables materialized by the login user in the system:

```
SELECT * FROM DBC.AllTempTablesVX;
```

## AMPUsageV[X]

**Category:** Integrity

**Database:** DBC

| View Column | Data Type   | Format                 | Comment  |
|-------------|---|------------------------|--|
| AccountName | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                 | Expanded account in effect when a request was submitted. SYSTEMACCOUNTNO tracks console utility activity such as table rebuild, Diskcopy, or Scandisk. |
| UserName    | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                 | Returns the name of a user.  |
| CpuTime     | FLOAT NOT NULL  | ---,---,---,--9.<br>99 | Returns the number of seconds of AMP insert CPU time used by the user and account.   |
| DiskIO      | FLOAT NOT NULL  | ---,---,---,--9        | Returns the total number of reads and writes to disk by each AMP in the system (or the total for all AMPs if the sum aggregate is specified).          |
| CpuTimeNorm | FLOAT NOT NULL  | ---,---,---,--9.<br>99 | Returns the normalized AMP CPU time used by the user and account.  |
| Vproc       | SMALLINT NOT NULL                                       | -(5)9                  | Identifies the virtual processor for which an event was logged.  |
| VprocType   | CHAR(4) LATIN<br>NOT CASESPECIFIC<br>NOT NULL           | X(4)                   | Returns the type of Vproc for which an event was logged.   |
| Model       | CHAR(4) LATIN<br>NOT CASESPECIFIC<br>NOT NULL           | X(4)                   | Returns the PMA (Processor Module Assembly) model information for the Vproc for which an event was logged.   |

## Usage Notes

### CPUTimeNorm

The CpuNorm column contains the normalized CPU value and is calculated:

CPU x scaling factor

### UserName

For this column, the SYSTEMUSERID is a system user name that tracks console utility activity such as table rebuild or Scandisk.

## Example: Using AMPUsageV

The following SELECT statement displays, for a given account, total CPU time and total disk accesses for all AMPs:

```
SELECT AccountName,SUM(CPUTime),SUM(Diskio)
      FROM DBC.AMPusageV WHERE AccountName='7654'
      GROUP BY AccountName;
```

Result:

| AccountName | Sum(CPUTime) | Sum(DiskIO) |
|-------------|--------------|-------------|
| -----       | -----        | -----       |
| 7654        | 204,352.88   | 5,226,742   |

## Related Topics

For more information about how to control access, space, and ownership, see *Teradata Vantage™ - Database Design*, B035-1094.

## ArchiveLoggingObjsV[X]

**Category:** Schema

**Database:** DBC

| View Column     | Data Type  | Format              | Comment  |
|-----------------|--|---------------------|--|
| DatabaseName    | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128)              | Returns the name of the database in which the archive logs reside.                                   |
| TVMName         | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128)              | Returns the name of the table.   |
| LogLevel        | CHAR(1) LATIN NOT<br>CASESPECIFIC NOT NULL           | X(1)                | Indicates whether the online archive logging is activated at either a table level or database level. |
| CreatorName     | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128)              | Archive logging creator.   |
| CreateTimeStamp | TIMESTAMP(0) NOT NULL                                | YYYY-MM-DDBHH:MI:SS | Returns the date and time that the object in the row was created.                                    |

## Example: Using ArchiveLoggingObjsV

The following SELECT statement displays information about the tables that have online archive active logs and their online archive logging levels. In this example, the logging levels are either T or D (O is also a possible value).

- T = Online Archive Logging was activated on the table level.
- D = Online Archive Logging was activated on the database level.
- O = Nontable objects.

```
SELECT databasename, tvmmname, loglevel FROM dbc.archiveloggingobjsv;
```

Result:

| DatabaseName    | TVMName    | LogLevel |
|-----------------|------------|----------|
| -----           | -----      | -----    |
| oarc_otop005db1 | Tab_nfb    | T        |
| oarc_oth007db1  | Tab_nfb    | D        |
| oarc_oth007db1  | tab_fb     | D        |
| oarc_otop010db1 | Tab_nfb    | D        |
| oarc_oth007db1  | tab_ppi    | D        |
| oarc_otop010db1 | tab_nusi   | D        |
| oarc_otop016db1 | Tab_nfb    | T        |
| oarc_oth008db1  | Tab_queue  | T        |
| oarc_otop010db1 | tab_bigcol | D        |
| oarc_otop001db1 | Tab_nfb    | T        |
| oarc_otop018db1 | Tab_nfb    | T        |
| oarc_comb002db1 | tab_fb     | T        |

## AssociationV[X]

**Category:** Operations

**Database:** DBC

| View Column  | Data Type  | Format         | Comment  |
|--------------|--|----------------|--|
| DatabaseName | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128)         | Returns the name of the database or user space in which the imported object resides. |
| TableName    | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128)         | Returns the imported object name.  |
| EventNum     | INTEGER NOT NULL                                     | --,---,---,--9 | Returns the client system event number of the restore operation.                     |

| View Column              | Data Type  | Format  | Comment   |
|--------------------------|--|---------|---|
| Original_ DatabaseName   | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128)  | Returns the name of the original database in which the object resided.  |
| Original_ TableName      | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128)  | Returns the name of the original object.  |
| Original_ TableKind      | CHAR(1) LATIN<br>NOT CASESPECIFIC                    | X(1)    | Original type of imported object.   |
| Original_ Version        | SMALLINT NOT NULL                                    | ---,--9 | Returns the original version of the imported object.  |
| Original_ ProtectionType | CHAR(1) LATIN<br>UPPERCASE NOT NULL                  | X(1)    | Returns the original protection type of the imported object, using the following codes: F = Fallback, N = None. |
| Original_ JournalFlag    | CHAR(2) LATIN NOT<br>CASESPECIFIC NOT NULL           | X(2)    | Original journaling for the imported object. Image status: first character BEFORE, second character AFTER.      |
| Original_ CreatorName    | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128)  | Returns the original creator of the imported table.   |
| Original_ CommentString  | VARCHAR(255) UNICODE<br>NOT CASESPECIFIC             | X(255)  | Returns the original comment on the imported table.   |

## Usage Notes

The Association view contains information about entities that were restored using the Archive and Recovery COPY utility. If a copied object is subsequently dropped, the information is deleted and is no longer available.

## Corresponding Tables

The X view references these additional tables:

- DBC.AccessRights
- DBC.Owner
- DBC.RoleGrants
- DBC.Roles

## Possible Values for Original\_JournalFlag

| Value | Description |
|-------|-------------|
| N     | No Journal  |

| Value | Description                               |
|-------|---|
| S     | Single Journal                            |
| D     | Dual Journal                              |
| L     | Local AFTER journal (non used for BEFORE) |

### Possible Values for Original\_TableKind

| Value | Description                    |
|-------|--------------------------------|
| T     | Data Table                     |
| V     | View                           |
| M     | Macro                          |
| J     | Journal Table                  |
| I     | Join Index Table               |
| P     | Stored Procedure               |
| G     | Trigger                        |
| F     | Scalar UDF                     |
| A     | Aggregate UDF                  |
| N     | Hash Index Table               |
| U     | User-defined Type              |
| H     | Instance or Constructor Method |
| E     | External Stored Procedure      |
| R     | Table Function                 |
| X     | Authorization                  |

### Example: Using AssociationV

The following SELECT statement selects information about tables copied into the Personnel database:

```
SELECT Original_DatabaseName,Original_TableName,TableName
FROM DBC.AssociationV WHERE DatabaseName = 'Personnel';
```

Result:

| Original_DatabaseName | Original_TableName | TableName   |
|-----------------------|--------------------|-------------|
| -----                 | -----              | -----       |
| OldPersonnel          | Empl_Addr          | Emp_Address |
| Personnel2            | Empl_Dept          | Empl_Dept   |

## Related Topics

For more information on recovery control, see the following:

- *Teradata Vantage™ - Database Administration*, B035-1093
- *Teradata® Archive/Recovery Utility Reference*, B035-2412

## AuthorizationsV[X]

**Category:** Security

**Database:** DBC

| View Column          | Data Type  | Format  | Comment  |
|----------------------|--|---------|--|
| DataBaseName         | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128)  | Returns the name of the database or user space in which the imported object resides.                             |
| AuthorizationName    | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128)  | Returns the name of the authorization.   |
| AuthorizationId      | BYTE(6) NOT NULL                                     | X(12)   | Returns the unique ID of the authorization.  |
| TableKind            | CHAR(1) LATIN<br>NOT CASESPECIFIC                    | X(1)    | Type of table.   |
| Version              | SMALLINT NOT NULL                                    | ---,--9 | Returns the version count, which is incremented each time the table is altered with a data definition statement. |
| AuthorizationType    | CHAR(1)<br>LATIN UPPERCASE                           | X(1)    | Returns the type of authorization.   |
| AuthorizationSubType | CHAR(1)<br>LATIN UPPERCASE                           | X(1)    | Specifies whether the authorization is a default authorization.  |
| OSDomainName         | VARCHAR(256)<br>LATIN CASESPECIFIC                   | X(256)  | Returns the domain that the user belongs to.   |
| OSUserName           | VARCHAR(512)<br>LATIN CASESPECIFIC                   | X(512)  | Returns the OS user name.  |

## Usage Notes

For information about the possible values for the TableKind column, see "TableKind Column."

### Corresponding Tables

The X view references these additional tables:

- DBC.AccessRights
- DBC.Owner

### Possible Values for AuthorizationName

If AuthorizationType is I, two entries appear for the same database name: one where AuthorizationName is the same as the name specified in the CREATE AUTHORIZATION statement and one where AuthorizationName is a default name.

### Possible Values for AuthorizationSubType

| Value | Description |
|-------|-------------|
| D     | Default     |
| N     | Non-default |

### Possible Values for AuthorizationType

| Value | Description     |
|-------|-----------------|
| D     | Definer         |
| I     | Invoker         |
| T     | Invoker Trusted |
| S     | Definer Trusted |

## Example: Using AuthorizationsV

The following SELECT statement returns details about the database DBA.

```
SELECT * FROM AuthorizationsV WHERE databasename = 'dba';
```

Result:

```

DatabaseName dba
AuthorizationName INVOKER_DEFAULT
AuthorizationId 00001F0A0000

```

```

TableKind X
Version 1
AuthorizationType I
AuthorizationSubType D
OSDomainName
OSUserName gdUsrGrp
DatabaseName dba
AuthorizationName myinvoker
AuthorizationId 00001E0A0000
TableKind X
Version 1
AuthorizationType I
AuthorizationSubType D
OSDomainName
OSUserName gdUsrGrp

```

## BusinessCalendar

**Category:** Operations

**Database:** Sys\_Calendar

| View Column      | Data Type | Format   | Comment   |
|------------------|-----------|----------|---|
| calendar_date    | DATE      | YY/MM/DD | The BUSINESSCALENDAR.calendar_date returns the default date format.   |
| day_of_week      | INTEGER   | -(10)9   | The BUSINESSCALENDAR.day_of_week returns a integer value which ranges from 1 to 7.                            |
| day_of_month     | INTEGER   | -(10)9   | The BUSINESSCALENDAR.day_of_month returns a integer value which ranges from 1 to 31.                          |
| day_of_year      | INTEGER   | -(10)9   | The BUSINESSCALENDAR.day_of_year returns a integer value which ranges from 1 to 366.                          |
| day_of_calendar  | INTEGER   | -(10)9   | The Sys_Calendar.BUSINESSCALENDAR.day_of_calendar returns the number of days since the begin of the calendar. |
| weekday_of_month | INTEGER   | -(10)9   | The BUSINESSCALENDAR.weekday_of_month returns the nth occurrence of the weekday in the month(1-5).            |
| week_of_month    | INTEGER   | -(10)9   | The BUSINESSCALENDAR.week_of_month returns the week number of month which ranges from 0 to 6.                 |
| week_of_quarter  | INTEGER   | -(10)9   | None  |
| week_of_year     | INTEGER   | -(10)9   | The BUSINESSCALENDAR.week_of_year returns the week number of year which ranges from 0 to 54.                  |

| View Column         | Data Type | Format   | Comment   |
|---------------------|-----------|----------|---|
| week_of_calendar    | INTEGER   | -(10)9   | The BUSINESSCALENDAR.week_of_calendar returns the relative week number associated with the calendar to which the input date belongs.          |
| month_of_quarter    | INTEGER   | -(10)9   | The BUSINESSCALENDAR.month_of_quarter returns the relative month number associated with the quarter to which the input date belongs.          |
| month_of_year       | INTEGER   | -(10)9   | The BUSINESSCALENDAR.month_of_year returns the relative month number associated with the year to which the input date belongs.                |
| month_of_calendar   | INTEGER   | -(10)9   | The BUSINESSCALENDAR.month_of_calendar returns the relative month number associated with the calendar to which the input date belongs.        |
| quarter_of_year     | INTEGER   | -(10)9   | The BUSINESSCALENDAR.quarter_of_year returns the relative quarter number associated with the year to which the input date belongs.            |
| quarter_of_calendar | INTEGER   | -(10)9   | The BUSINESSCALENDAR.quarter_of_calendar returns the relative quarter number associated with the calendar to which the input date belongs.    |
| year_of_calendar    | INTEGER   | -(10)9   | The BUSINESSCALENDAR.year_of_calendar returns the relative year number associated with the business calendar to which the input date belongs. |
| WeekBegin           | DATE      | YY/MM/DD | The BUSINESSCALENDAR.weekbegin returns the Week Begin of the given date to which week it belongs to.  |
| WeekEnd             | DATE      | YY/MM/DD | The BUSINESSCALENDAR.weekend returns the Week End of the given date to which week it belongs to.  |
| MonthBegin          | DATE      | YY/MM/DD | The BUSINESSCALENDAR.monthbegin returns the Month Begin of the given date to which Month it belongs to.                                       |
| MonthEnd            | DATE      | YY/MM/DD | The BUSINESSCALENDAR.monthend returns the Month End of the given date to which Month it belongs to.   |
| QuarterBegin        | DATE      | YY/MM/DD | The BUSINESSCALENDAR.quarterbegin returns the Quarter Begin of the given date to which Quarter it belongs to.                                 |
| QuarterEnd          | DATE      | YY/MM/DD | The BUSINESSCALENDAR.quarterend returns the Quarter End of the given date to which Quarter it belongs to.                                     |
| YearBegin           | DATE      | YY/MM/DD | The BUSINESSCALENDAR.yearbegin returns the year Begin of the given date to which Year it belongs to.  |

| View Column          | Data Type | Format   | Comment  |
|----------------------|-----------|----------|--|
| YearEnd              | DATE      | YY/MM/DD | The BUSINESSCALENDAR.yearend returns the year end of the given date to which Year it belongs to.                               |
| IsBusinessDay        | BYTEINT   | -(3)9    | The BUSINESSCALENDAR.isbusinessday returns whether the given date is a Business Day or not.                                    |
| BusinessWeekBegin    | DATE      | YY/MM/DD | The BUSINESSCALENDAR.businessweekbegin returns the Business Week Begin of the given date to which week it belongs to.          |
| BusinessWeekEnd      | DATE      | YY/MM/DD | The BUSINESSCALENDAR.businessweekend returns the Business Week End of the given date to which week it belongs to.              |
| BusinessMonthBegin   | DATE      | YY/MM/DD | The BUSINESSCALENDAR.businessmonthbegin returns the Business Month Begin of the given date to which Month it belongs to.       |
| BusinessMonthEnd     | DATE      | YY/MM/DD | The BUSINESSCALENDAR.businessmonthend returns the Business Month End of the given date to which Month it belongs to.           |
| BusinessQuarterBegin | DATE      | YY/MM/DD | The BUSINESSCALENDAR.businessquarterbegin returns the Business Quarter Begin of the given date to which Quarter it belongs to. |
| BusinessQuarterEnd   | DATE      | YY/MM/DD | The BUSINESSCALENDAR.businessquarterend returns the Business Quarter End of the given date to which Quarter it belongs to.     |
| BusinessYearBegin    | DATE      | YY/MM/DD | The BUSINESSCALENDAR.businessyearbegin returns the Business year Begin of the given date to which Year it belongs to.          |
| BusinessYearEnd      | DATE      | YY/MM/DD | The BUSINESSCALENDAR.businessyearend returns the Business year end of the given date to which Year it belongs to.              |

## Usage Notes

### Note:

As mentioned previously many views that do not end in V or VX are deprecated. However, although this view does not end in V or VX it is not deprecated.

### Week\_of\_quarter

The week\_of\_quarter column can contain a range of possible values from 0 through 14.

## Corresponding Tables

The corresponding tables for this view are:

- DBC.BusinessCalendarPattern
- DBC.BusinessCalendarException

## Examples of Queries of Calendar Data

These examples show simple queries that you can use to determine the day of the week (day\_of\_week) and the date at the beginning of the week (weekBegin). The session calendars are the ISO and COMPATIBLE system-defined calendars.

- The first two examples provide queries that return the day of the week (for example, day 1, 2, or 7) for a particular date.
- The last two examples provide queries that return the date of the first day of a week (day 1) for the week to which a particular date belongs.

### Example: Day of the Week (ISO Calendar)

```

Sel day_of_week from Sys_Calendar.Calendar where calendar_date =
date '2011-01-01';
day_of_week
-----
6

```

### Example: Day of the Week (COMPATIBLE Calendar)

```

Sel day_of_week from Sys_Calendar.Calendar where calendar_date =
date '2011-01-01';
day_of_week
-----
1

```

### Example: Beginning of the Week (ISO Calendar)

```

Sel weekBegin from Sys_Calendar.BusinessCalendar where calendar_date =
date '2011-01-01';
WeekBegin
-----
10/12/27

```

**Example: Beginning of the Week (COMPATIBLE Calendar)**

```

Sel weekBegin from Sys_Calendar.BusinessCalendar where calendar_date =
date '2011-01-01';
WeekBegin
-----
11/01/01

```

**Related Topics**

For more information about the following topics, see *Teradata Vantage™ SQL Date and Time Functions and Expressions*, B035-1211:

- Teradata, ISO, and COMPATIBLE system-defined business calendars
- How the basic units (weeks, months, quarter, and year) of the ISO system-defined calendar are defined
- Business calendar functions

**BusinessCalendarExceptions**

**Category:** Operations

**Database:** Sys\_Calendar

| View Column        | Data Type   | Format   | Comment  |
|--------------------|---|----------|--|
| CalendarName       | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)   | The BusinessCalendarExceptions.<br>CalendarName returns the name of the<br>business calendar.  |
| ExceptionIndicator | VARCHAR(3)<br>UNICODE<br>NOT CASESPECIFIC               | X(3)     | The BusinessCalendarExceptions.<br>ExceptionIndicator returns ON(working day)<br>or OFF(Non working day) for each<br>exception date. |
| ExceptionDate      | DATE NOT NULL   | YY/MM/DD | The BusinessCalendarExceptions.<br>ExceptionDate returns the exception date<br>for which exception is created.                       |
| ExceptionReason    | VARCHAR(1024)<br>UNICODE<br>NOT CASESPECIFIC            | X(1024)  | The BusinessCalendarExceptions.<br>ExceptionReason returns the reason for the<br>exception date.                                     |
| CreatorName        | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)   | The BusinessCalendarExceptions.<br>CreatorName returns the user name who<br>created the exception.                                   |

| View Column  | Data Type                | Format              | Comment   |
|--------------|--------------------------|---------------------|---|
| CreationTime | TIMESTAMP(0)<br>NOT NULL | YYYY-MM-DDBHH:MI:SS | The BusinessCalendarExceptions. CreationTime returns the timestamp when the exception is created. |

## Usage Notes

### ExceptionData

The date specified in the ExceptionData column must be within the calendar period (that is, the value specified in the BusinessCalendar view YearEnd column).

### ExceptionReason

The value of the ExceptionReason column can be NULL.

## BusinessCalendarPatterns

**Category:** Operations

**Database:** Sys\_Calendar

| View Column    | Data Type  | Format              | Comment   |
|----------------|--|---------------------|---|
| CalendarName   | VARCHAR(128)<br>UNICODE NOT CASESPECIFIC<br>NOT NULL | X(128)              | The BusinessCalendarPatterns. CalendarName returns the name of the business calendar.               |
| DayName        | VARCHAR(9)<br>UNICODE<br>NOT CASESPECIFIC            | X(9)                | The BusinessCalendarPatterns.DayName returns day name like SUNDAY, MONDAY, etc.                     |
| Pattern        | VARCHAR(3)<br>UNICODE<br>NOT CASESPECIFIC            | X(3)                | The BusinessCalendarPatterns.Pattern returns ON(working day) or OFF(Non working day) for each day.  |
| PatternComment | VARCHAR(1024)<br>UNICODE<br>NOT CASESPECIFIC         | X(1024)             | The BusinessCalendarPatterns. PatternComment returns the comment for each day in the pattern.       |
| CreatorName    | VARCHAR(128)<br>UNICODE NOT CASESPECIFIC<br>NOT NULL | X(128)              | The BusinessCalendarPatterns. CreatorName returns the user name who created/modified the pattern.   |
| LastModified   | TIMESTAMP(0)<br>NOT NULL                             | YYYY-MM-DDBHH:MI:SS | The BusinessCalendarPatterns. LastModified returns the timestamp when the pattern is last modified. |

## Usage Notes

### CreatorName

A CreatorName column value of DBC indicates a system-defined business calendar.

## CDSTableSizeV

**Category:** Accounting

**Database:** DBC

| View Column            | Data Type   | Format                       | Comment  |
|------------------------|---|------------------------------|--|
| DatabaseName           | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                       | The DatabaseName field stores the DatabaseName of the table.   |
| TableName              | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                       | The TableName field stores the TableName of the table.   |
| CurPerm                | BIGINT  | --,---,---,---,---,---,--9   | The CurPerm field stores the Current Physical Size of the table in bytes.  |
| CurPermWithoutFallback | BIGINT  | --,---,---,---,---,---,--9   | The CurPermWithoutFallback field stores the Current Physical Size of the table without fallback in bytes.          |
| CurPermCDS             | DECIMAL(28,3)   | --(25).9(3)                  | The CurPermCDS field stores the Logical (CDS) Size of the table in bytes. It excludes Fallback subtable.           |
| CompressionFactor      | FLOAT   | -9.<br>9999999999999999E-999 | The CompressionFactor field stores the X value which is 1:X times when compared to the physical size of the table. |
| LastCollectTimeStamp   | TIMESTAMP(0)  | YYYY-MM-DDBHH:<br>MI:SS      | The LastCollectTimeStamp field stores the Last statistics collection time stamp.                                   |
| CurrentTimeStamp       | TIMESTAMP(0)  | YYYY-MM-DDBHH:<br>MI:SS      | The CurrentTimeStamp field stores the Current TimeStamp value.   |
| IsValid                | VARCHAR(1)<br>UNICODE NOT<br>CASESPECIFIC               | X(1)                         | The IsValid field indicates if for the table collect stats has been run or not.                                    |

## CharSetsV

**Category:** Operations

**Database:** DBC

| View Column | Data Type   | Format | Comment  |
|-------------|---|--------|--|
| CharSetName | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | Returns the name of a user-defined character set that is installed and available for the current session, or the name assigned to the character set defined by this row. |

## Usage Notes

Teradata Vantage can support many user-defined character sets (see view CharTranslationsV). You can install a maximum of 12 character sets at any given time. The CharSets view contains the names of character sets that are currently installed and can be specified at the session level. If the view does not exist or no rows are found, then no user-defined character sets are available.

### CharSetName

Each name shown in CharSets can be used as the identifier in the BTEQ `[.]SET SESSION CHARSET <'name'>` command or the CLIV2 call `CHARSET <name>`. However, the specified character set should be compatible with the internal code of the logon client system.

If a CharSetName is ambiguous as to its compatibility with the logon client system of the viewer session, consult the database administrator.

## Example: Using CharSetsV

The following example shows that two user-defined character sets are available for the requesting user:

```
SELECT * FROM DBC.CharSetsV;
```

Result:

```
CharSetName
French_EBCDIC
Swedish_EBCDIC
```

## CharTranslationsV

**Category:** Operations

**Database:** DBC

| View Column | Data Type   | Format | Comment  |
|-------------|---|--------|--|
| CharSetName | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | Returns the name of a user-defined character set that is installed and available for the current session, or the name assigned to the character set defined by this row. |
| CharSetId   | BYTEINT NOT NULL  | -(3)9  | Returns a number uniquely identifying the character set of the CHARSET_COLL collation.   |
| InstallFlag | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL                  | X(1)   | Character set to be installed is already on the Teradata Database: Y (yes), N (no).  |
| E2I         | BYTE(256) NOT NULL                                      | X(512) | Returns the client system (external) to Teradata Database (internal) Hex translation codes.  |
| E2IUp       | BYTE(256) NOT NULL                                      | X(512) | Returns the client system (external) to Teradata Database (internal) and uppercase Hex translation codes.  |
| I2E         | BYTE(256) NOT NULL                                      | X(512) | The CharTranslationsV.I2E column is the translation table from DBC internal character set to the external character set  |
| I2EUp       | BYTE(256) NOT NULL                                      | X(512) | The CharTranslationsV.I2EUp column is the translation table from DBC internal character set to external, upper cased character set.                                      |

## Usage Notes

If client system connections are to use the defined character sets, the system administrator specifies which character set is assigned to which client system (see [HostsInfoV](#)). Otherwise, the standard default is used. Also, the user may specify a defined character set after a session is started (see [CharSetsV](#)).

When specifying a character set for a session, the choice should be compatible with the internal code of the logon client system; that is, an EBCDIC-compatible character set for sessions initiated from an IBM mainframe, ASCII-compatible sets for all others. It is suggested, therefore, that a convention be used for naming character sets which differentiates between EBCDIC and ASCII compatibility (see the example below).

### CharSetId

If the character set is user-defined, the ID should also exist in DBC.CharTranslations.CharSetID.

For constraints not involving comparison of character data or not using CHARSET\_COLL for evaluation, the value is NULL.

### InstallFlag

The database must be reset to install the rows containing a Y in the InstallFlag field. If the value of InstallFlag is Y in 12 rows or fewer, each Y row is loaded. If InstallFlag is Y in more than 12 rows, then the

CharSetName values are sorted in ascending ASCII sequence, and rows are loaded in alphabetical order until 12 sets are installed or the names are exhausted.

## Example: Using CharTranslationsV

The example below shows that the hexadecimal translation tables for 6 character sets have been defined, and that two of these are flagged for loading.

```
SELECT * FROM DBC.CharTranslationsV;
```

Result:

| CharSetName      | Set Id | Flag  | E2I                       |
|------------------|--------|-------|---------------------------|
| -----            | -----  | ----- | -----                     |
| German_EBCDIC    | 100    | N     | 00010203A809A97FD1D2D3... |
| Italian_EBCDIC   | 105    | N     | 00010203A809A97F2395EE... |
| Spanish_EBCDIC   | 103    | N     | 00010203A809A97FD1D2D3... |
| French_EBCDIC    | 104    | Y     | 00010203A809A97FD1D2D3... |
| Norwegian_EBCDIC | 101    | N     | 00010203A809A97FD1D2D3... |
| Swedish_EBCDIC   | 102    | Y     | 00010203A809A97FD1D2D3... |

## ChildrenV[X]

**Category:** Database

**Database:** DBC

| View Column | Data Type                                      | Format | Comment   |
|-------------|--|--------|---|
| Child       | VARCHAR(128) UNICODE NOT CASESPECIFIC NOT NULL | X(128) | Returns the name of a referencing database or user. |
| Parent      | VARCHAR(128) UNICODE NOT CASESPECIFIC NOT NULL | X(128) | Returns the name of a parent database or user.      |

## Usage Notes

You can use this view to show all the databases under a user.

### Corresponding Tables

The V view also references the DBC.Owners table.

The X view references these additional tables:

- DBC.AccessRights

- DBC.Owners
- DBC.Roles
- DBC.RoleGrants

## Example: Using ChildrenV

The following SELECT statement displays databases and users that are owned by the Finance database:

```
SELECT Parent, Child FROM DBC.ChildrenV
      WHERE Parent = 'Finance';
```

Result:

| Parent  | Child      |
|---------|------------|
| -----   | -----      |
| Finance | Personnel  |
| Finance | Jones      |
| Finance | Accounting |

## CollationsV

**Category:** Integrity

**Database:** DBC

| View Column  | Data Type   | Format | Comment   |
|--------------|---|--------|---|
| CollName     | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | This is the name of the collation. Only the collation with name MULTINATIONAL and the CollInstall flag set to "Y" is installed on the DBC.                  |
| CollInstall  | CHAR(1) LATIN<br>NOT CASESPECIFIC<br>NOT NULL           | X(1)   | This is the install flag for the collation. "Y" indicates yes, otherwise no. This flag applies only to Multinational collation. Others are never installed. |
| CollEqvClass | BYTE(256) NOT NULL                                      | X(512) | This is the translation table used for first-level comparison.  |
| CollOrderCS  | BYTE(256) NOT NULL                                      | X(512) | This is the translation table used for second-level, case-specific comparison.  |
| CollOrderUC  | BYTE(256) NOT NULL                                      | X(512) | This is the translation table used for second-level, uppercase comparison.  |

## Usage Notes

The DBC.Collations view initially contains five rows:

- KANJI5026\_STANDARD
- KANJIKATA\_STANDARD
- KANJI5035\_STANDARD
- NORWEGIAN\_STANDARD
- SWEDISH\_STANDARD

Database administrators can run the CollInstallMulti macro to specify which row in the DBC.Collations view to use as the collation sequence when the user or session COLLATION option is set to MULTINATIONAL. You must reset (initialize) the SQL Engine before the new collation sequence can take effect.

When you define a new collation with a name other than 'MULTINATIONAL,' set the CollInstall flag to N to avoid extra processing during startup.

## Example: Using CollationsV

The following statement returns the collation information for all collation sequences defined in the Collations view:

```
SELECT CollName
FROM DBC.CollationsV;
```

The result is the following list:

```
CollName
-----
KANJI5026_STANDARD
KANJIKATA_STANDARD
KANJI5035_STANDARD
NORWEGIAN_STANDARD
SWEDISH_STANDARD
```

## Related Topics

For more information about the MULTINATIONAL collation sequence, see *Teradata Vantage™ - Advanced SQL Engine International Character Set Support*, B035-1125.

## ColumnStatsV

**Category:** Optimizer Statistics

**Database:** DBC

| View Column        | Data Type   | Format                   | Comment   |
|--------------------|---|--------------------------|---|
| DatabaseName       | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                   | Returns the database name of the table, column, group of columns, index, view, or query on which statistics were collected. |
| TableName          | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                   | Returns the name of the table that contains the column, group of columns, or index on which the statistics were collected.  |
| ColumnName         | VARCHAR(10000)<br>UNICODE NOT<br>CASESPECIFIC           | X(10000)                 | The ColumnName column identifies a column or columns.   |
| StatsName          | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC             | X(128)                   | The StatsName column contains the name associated with the statistic  |
| StatsSource        | CHAR(1)<br>LATIN UPPERCASE                              | X(1)                     | The StatsSource column records the method this statistic is acquired.   |
| ValidStats         | CHAR(1)<br>LATIN UPPERCASE                              | X(1)                     | The ValidStats column indicates whether the statistics are valid or not.  |
| DBSVersion         | VARCHAR(32)<br>LATIN UPPERCASE                          | X(32)                    | Returns the version of the database that contains the objects on which the statistics were collected.                       |
| IndexNumber        | SMALLINT  | ---,--9                  | The IndexNumber column is the Index number of the index on which statistics are collected.                                  |
| SampleSignature    | VARCHAR(256)<br>LATIN NOT<br>CASESPECIFIC               | X(256)                   | The SampleSignature column is the Sample options encoded as a 10 character signature.                                       |
| SampleSizePct      | DECIMAL(5,2)  | ----.99                  | The SampleSizePct column is the Sample size percent used when collecting statistics.  |
| ThresholdSignature | VARCHAR(512)<br>LATIN NOT<br>CASESPECIFIC               | X(512)                   | The ThresholdSignature column is the Threshold options encoded as a 17 character signature.                                 |
| MaxIntervals       | SMALLINT  | ---,--9                  | The MaxIntervals column is the User-specified maximum number of intervals.  |
| MaxValueLength     | INTEGER   | --,---,---,--9           | The MaxValueLength column is the User-specified maximum value length.   |
| RowCount           | FLOAT   | ----,---,---,---,<br>--9 | The RowCount column is the row count of the table, view or query on which statistics are collected.                         |

| View Column           | Data Type    | Format                  | Comment  |
|-----------------------|--------------|-------------------------|--|
| UniqueValueCount      | FLOAT        | ---,---,---,---,<br>--9 | The UniqueValueCount column is the Number of unique values of the ExpressionList.                                    |
| PNullUniqueValueCount | FLOAT        | ---,---,---,---,<br>--9 | The PNullUniqueValueCount column is the Number of unique values from rows with partial nulls.                        |
| NullCount             | FLOAT        | ---,---,---,---,<br>--9 | The NullCount column is the Number of nulls of the ExpressionList.   |
| AllNullCount          | FLOAT        | ---,---,---,---,<br>--9 | The AllNullCount column is the Number of all nulls of the ExpressionList.  |
| HighModeFreq          | FLOAT        | ---,---,---,---,<br>--9 | The HighModeFreq column is the highest frequency of values of the ExpressionList.                                    |
| PNullHighModeFreq     | FLOAT        | ---,---,---,---,<br>--9 | The PNullHighModeFreq column is the highest frequency of values having partial nulls of the ExpressionList.          |
| StatsSkipCount        | INTEGER      | --,---,---,--9          | The StatsSkipCount column indicates how many times the statistics collection on the ExpressionList has been skipped. |
| CreateTimeStamp       | TIMESTAMP(0) | YYYY-MM-DDBHH:MI:SS     | The CreateTimeStamp column is the statistics creation time stamp.  |
| LastCollectTimeStamp  | TIMESTAMP(0) | YYYY-MM-DDBHH:MI:SS     | The LastCollectTimeStamp column is the Last statistics collection time stamp.  |
| LastAlterTimeStamp    | TIMESTAMP(0) | YYYY-MM-DDBHH:MI:SS     | The LastAlterTimeStamp column is the last user updated time stamp.   |

## Usage Notes

### ColumnName

- If more than one column or expression is specified, each column or expression is separated by a comma.
- The maximum number of columns is 64.
- If expressions are in the list, the maximum number of columns can be reduced past the limit of 64, depending on the combined total size of the text in the expressions.
- If the combined total size of the expression text causes the maximum column limit to be less than the actual number of columns in the list, an error occurs.

## Referenced Columns

Many of the Data Dictionary view columns have referenced table columns. That is, the value in the view column corresponds to a value in the selected column referenced in the table. It would be meaningful to join the view and the referenced table based on the selected column and the referenced column.

Referenced columns for this view are:

| View Column | Referenced Column       |
|-------------|-------------------------|
| IndexNumber | DBC.Indexes.IndexNumber |

## MaxInterval and MaxValueLength

If these statistics are collected with system determined maximum intervals and maximum value length, the MaxInterval and MaxValueLength columns are NULL.

## SampleSizePct

If sampling is not used, the SampleSizePct column is set to 0 or 100.

## StatsSource

The StatsSource column records the method by which this statistic is acquired. For information about the possible values for the StatsSource column, see "StatsSource Column."

## To Get Information Not Contained in This View

This view does not contain:

- Statistics on tables protected by row-level security. To get these statistics, create views on DBC.StatsTbl. You can view details about this table in Teradata SQL Assistant or Teradata Studio Express.
- Information about column attributes. To get this information, join the DBC.StatsTbl table to the DBC.TVFields table. You can view details about this table in Teradata SQL Assistant or Teradata Studio Express.

## Example: Using ColumnStatsV

This example assumes the following statistics have been collected:

```
STATISTICS
  COLUMN o_orderkey
    ON Orders;
STATISTICS
  COLUMN o_orderdatetime
    ON Orders;
```

This query can be used to retrieve the statistics:

```
SELECT * FROM dbc.ColumnStatsV
      WHERE Databasename = 'sales'
      AND TableName      = 'Orders'
```

## Related Topics

| For information about statistics collected on ...       | See ...   |
|---|---|
| groups of non-indexed columns                           | <a href="#">MultiColumnStatsV.</a>                      |
| indexes for which two or more columns have been defined | <a href="#">IndexStatsV[X].</a>                         |
| tables  | <a href="#">StatsV</a> and <a href="#">TableStatsV.</a> |
| materialized temporary tables                           | <a href="#">TempTableStatsV.</a>                        |
| single expressions                                      | <a href="#">ExpStatsV.</a>                              |
| multiple expressions                                    | <a href="#">MultiExpStatsV.</a>                         |

## ColumnsV[X]

**Category:** Schema

**Database:** DBC

| View Column  | Data Type   | Format | Comment  |
|--------------|---|--------|--|
| DatabaseName | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | Returns the name of a database.                                |
| TableName    | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | Returns the name of a table, view, stored procedure, or macro. |
| ColumnName   | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | Returns the name of the column or parameter.                   |
| ColumnFormat | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC             | X(128) | Returns the format of column or parameter.                     |
| ColumnTitle  | VARCHAR(256)<br>UNICODE NOT<br>CASESPECIFIC             | X(256) | Returns the heading for the column.                            |

| View Column             | Data Type  | Format            | Comment  |
|-------------------------|--|-------------------|--|
| SPParameterType         | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(1)              | Returns the data type of a stored procedure parameter and is also used as the data type of a UDF/UDM.  |
| ColumnType              | CHAR(2) LATIN<br>UPPERCASE                       | X(2)              | Returns the type of data in a table or view column, or stored procedure or macro parameter.  |
| ColumnUDTName           | VARCHAR(128)<br>UNICODE<br>UPPERCASE             | X(128)            | Returns the name of a UDT if that column data type is a UDT.   |
| ColumnLength            | INTEGER  | Z,ZZZ,<br>ZZZ,ZZ9 | Returns the length of a column as the maximum number of bytes used to physically store a column value in a table row.  |
| DefaultValue            | VARCHAR(1024)<br>UNICODE NOT<br>CASESPECIFIC     | X(1024)           | Returns any default value assigned to the column or parameter.   |
| Nullable                | CHAR(1) LATIN<br>UPPERCASE                       | X(1)              | Returns a code to indicate whether or not a column may have a null value.  |
| CommentString           | VARCHAR(255)<br>UNICODE NOT<br>CASESPECIFIC      | X(255)            | Returns user-supplied text or commentary on the column, database, table, view, macro, user-defined function, user-defined types, user-defined methods, stored procedure, role, profile, or user. |
| DecimalTotalDigits      | SMALLINT   | -ZZ9              | Returns an integer indicating the total number of decimal digits (if the column is defined as a decimal).  |
| DecimalFractionalDigits | SMALLINT   | -ZZ9              | Returns an integer indicating the total number of fractional digits (if the column is defined as a decimal).   |
| ColumnId                | SMALLINT<br>NOT NULL                             | ---,--9           | Returns the internal identifier assigned to the column by the DBC.   |
| UpperCaseFlag           | CHAR(1) LATIN<br>UPPERCASE                       | X(1)              | Returns the case indicator flag for the column, and whether comparisons on the column are case specific.   |
| Compressible            | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC             | X(1)              | Compressible indicates whether the column can be compressed and the type of compression defined for the column.  |
| CompressValue           | NULL   | NULL              | Returns the column is removed from tvfields so NULL is being returned.   |

| View Column         | Data Type   | Format                             | Comment  |
|---------------------|---|------------------------------------|--|
| ColumnConstraint    | VARCHAR(8192)<br>UNICODE NOT<br>CASESPECIFIC            | X(8192)                            | Returns the condition text for column level Check.   |
| ConstraintCount     | SMALLINT<br>NOT NULL                                    | ---,--9                            | Returns the count of table level constraints referencing this column.  |
| CreatorName         | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                             | Table or database creator. For DBC. AllRights, the grantor of explicit rights, otherwise the user who executed the CREATE statement. |
| CreateTimeStamp     | TIMESTAMP(0)  | YYYY-MM-DDBHH:MI:SS                | Returns the date and time that the object in the row was created.  |
| LastAlterName       | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                             | Returns the name of the user who last updated the dictionary row.  |
| LastAlterTimeStamp  | TIMESTAMP(0)  | YYYY-MM-DDBHH:MI:SS                | Returns the time the dictionary row was last updated.  |
| CharType            | SMALLINT  | ---,--9                            | Returns the type of the different character data types.  |
| IdColType           | CHAR(2) LATIN<br>UPPERCASE                              | X(2)                               | Indicates if a column is a regular column or an identity column.   |
| AccessCount         | BIGINT  | --,---,---,<br>---,---,---,<br>--9 | Returns the access count for the corresponding database object.  |
| LastAccessTimeStamp | TIMESTAMP(0)  | YYYY-MM-DDBHH:MI:SS                | Returns the time that the corresponding object was last accessed.  |
| CompressValueList   | VARCHAR(8192)<br>UNICODE NOT<br>CASESPECIFIC            | X(8192)                            | Returns the algorithm names in addition to the compress values.  |
| TimeDimension       | CHAR(1) LATIN<br>UPPERCASE                              | X(1)                               | Returns the ValidTime and TransactionTime properties for a period column.  |
| VTCheckType         | CHAR(1) LATIN<br>UPPERCASE                              | X(1)                               | Returns the ValidTime dimension information for the CHECK condition.   |
| TTCheckType         | CHAR(1) LATIN<br>UPPERCASE                              | X(1)                               | Returns the TransactionTime dimension for the CHECK constraint on a table with TransactionTime.                                      |

| View Column                | Data Type                              | Format  | Comment  |
|----------------------------|--|---------|--|
| ConstraintId               | BYTE(4)                                | X(8)    | The DBC.COLUMNSVX.ConstraintId field contains the id of the security constraint if it is a security constraint column.   |
| ArrayColNumberOfDimensions | BYTEINT                                | -(3)9   | The DBC.ColumnsVX.ArrayColNumberOfDimensions field contains a numeric value that indicates the number of dimensions for an ARRAY type column.  |
| ArrayColScope              | VARCHAR(3200)<br>LATIN<br>UPPERCASE    | X(3200) | The DBC.ColumnsVX.ArrayColScope field contains a string of the format '[n:m]...' which describes the lower and upper bounds values for each dimension of the ARRAY type column.                              |
| ArrayColElementType        | CHAR(2) LATIN<br>UPPERCASE             | X(2)    | The DBC.ColumnsVX.ArrayColElementType field contains a 2-character data type code for the element of an ARRAY type column.   |
| ArrayColElementUdtName     | VARCHAR(128)<br>UNICODE<br>UPPERCASE   | X(128)  | The DBC.ColumnsVX.ArrayColElementUdtName field contains a UDT Name when the element's data type of an ARRAY type column is UDT.  |
| PartitioningColumn         | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL | X(1)    | DBC.ColumnsVX.PartitioningColumn is N if not a partitioning column and Y if a partitioning column in a partitioning expression.  |
| ColumnPartitionNumber      | BIGINT<br>NOT NULL                     | -(19)9  | DBC.ColumnsVX.ColumnPartitionNumber is 0 if not column partitioned   |
| ColumnPartitionFormat      | CHAR(2) LATIN<br>UPPERCASE<br>NOT NULL | X(2)    | DBC.ColumnsVX.ColumnPartitionFormat is NA if not applicable (i.e., not column partitioned). First character: C = COLUMN format, R = ROW format. Second character: S = system-determined, U = user-specified. |
| ColumnPartitionAC          | CHAR(2) LATIN<br>UPPERCASE<br>NOT NULL | X(2)    | DBC.ColumnsVX.ColumnPartitionAC is NA if not applicable (i.e., not column partitioned), NC if no auto compress, and AC if auto compress.   |
| PseudoUDTFieldId           | SMALLINT                               | ---,--9 | The PseudoUDTFieldId field specifies the field id of the derived period column in the table in which the begin   |

| View Column        | Data Type                                 | Format             | Comment   |
|--------------------|---|--------------------|---|
|                    |   |                    | or end column of the derived period column participate; otherwise NULL.   |
| PseudoUDTFieldType | CHAR(2) LATIN<br>UPPERCASE                | X(2)               | The PseudoUDTFieldType specifies whether the column is a derived period column or begin column of a derived period column or end column of a derived period column; otherwise NULL. |
| StorageFormat      | VARCHAR(6)<br>UNICODE NOT<br>CASESPECIFIC | X(6)               | The StorageFormat specifies the storage format of a any column which has multiple storage formats, or a NULL value if the column does not have this characteristic.                 |
| DatasetSchemaName  | VARCHAR(128)<br>UNICODE<br>UPPERCASE      | X(128)             | The name of a schema associated with a DATASET data type column, or a NULL value if the column is not a DATASET data type or does not have a schema associated with it.             |
| InlineLength       | INTEGER                                   | --,---,---,<br>--9 | Returns the inline length of a UDT.   |
| TSColumnType       | CHAR(2) LATIN<br>UPPERCASE                | X(2)               | The TSColumnType identifies a Time Series column type of TimeBucket, TimeCode or SeqNo; NULL if a column does not fall into these types.  |
| AutoColumn         | CHAR(1) LATIN<br>UPPERCASE                | X(1)               | Returns if a column is an AUTO COLUMN.  |
| RowVersionNo       | SMALLINT                                  | ---,--9            | Returns the row version number.   |

## Usage Notes

### ColumnsqV[X] and ColumnsjqV[X]

When querying DBC.Columns for a view, information on column attributes (for example, length and type) is NULL. Because column attributes correspond to the table for which they were defined, they are not stored in the Data Dictionary and are not accessible through this view.

Obtain view column information using DBC.Columnsq and DBC.Columnsjq. Access the information in the following views from a SELECT statement, and the result set is joinable to information in other views and tables:

- ColumnsqV and ColumnsqVX (collectively referred to as ColumnsqV[X])
- ColumnsjqV and ColumnsjqVX (collectively referred to as ColumnsjqV[X])

ColumnsqV[X] views provide the same column information as ColumnsV[X], including all the information for view columns. ColumnsjqV[X] provides the same information as ColumnsqV[X], but only for tables, NoPI tables, and views.

### **ANSI Temporal Table Support**

Some of the column values show information related to a Teradata temporal table or an ANSI temporal table. For more information about these tables, see *Teradata Vantage™ ANSI Temporal Table Support*, B035-1186 and *Teradata Vantage™ Temporal Table Support*, B035-1182.

Use the SHOW TABLE statement, HELP COLUMN statement, and the ColumnsV[X] and DBC.PartitioningConstraintsV[X] views to obtain partitioning information for a table or join index.

The HELP INDEX statement Ordered or Partitioned attribute shows if an index is partitioned. For a NoPI table, the HELP INDEX statement cannot be used to determine whether or not the table or join index is partitioned.

### **ArrayColElementUdtName**

If the object is an ARRAY data type the name is only recorded in the ArrayColElementUdtName column if the element type for it is a UDT.

The ArrayColElementUdt column is only included in the associated views when:

- TVFields.FieldType = A1 or AN
- UDTInfo.TypeId = TVFields.TableId

### **ColumnPartitionNumber**

This column is set to the column partition number of the column partition in which the column belongs. 0 indicates the column is not partitioned.

---

#### **Note:**

Columns of a table or join index with the same column partition number belong to the same column partition.

---

### **CompressValueList**

The column contains the name of the compression routine where any complex data type column is defined to use the complex data type internal compression (for example, for a JSON data type, the CompressValueList column value would be "JSON\_COMPRESS").

### **DecimalTotalDigits**

A value of -128 for the DecimalTotalDigits column indicates that the default is used.

### **DecimalFractionalDigits**

The DecimalFractionalDigits column is always -128.

## InlineLength

Returns the inline length of a UDT; specifically, this column stores the inline storage size for ST\_GEOMETRY, DATASET, XML, and JSON. This column is NULL for all other UDTs and types.

## TimeDimension

For information about the possible values of the TimeDimension column, see "TimeDimension Column."

## Possible Values for ArrayColElementType

| Value | Description                      |
|-------|----------------------------------|
| A1    | One dimensional ARRAY data type  |
| AN    | Multidimensional ARRAY data type |

The ArrayColElementType column is only included in the associated views when:

- TVFields.FieldType = A1 or AN
- UDTInfo.TypeId = TVFields.TableId

## Possible Values for ArrayColNumberOfDimensions

The range of possible values is 1 through 5.

The ArrayColNumberOfDimensions column is only included in the view when all of the following are true:

- TVFields.FieldType = A1 or AN
- UDTInfo.TypeId = TVFields.TableId

For more information about ARRAY data type dimensions, see *Teradata Vantage™ SQL Data Definition Language Syntax and Examples*, B035-1144 and *Teradata Vantage™ Data Types and Literals*, B035-1143.

## Possible Values for ArrayColScope

The ArrayColScope column is only populated when the data type is created. The bound values for each dimension are a string in the [n:m] format.

The ArrayColScope column is only included in the view when all of the following are true:

- TVFields.FieldType = A1 or AN
- UDTInfo.TypeId = TVFields.TableId

## Possible Values for Compressible

| Value | Description                          |
|-------|--------------------------------------|
| A     | Algorithmic compression (ALC) column |
| C     | Multi-value compression column       |

| Value | Description  |
|-------|--|
| N     | Non-compress columns   |
| U     | Any complex data type column defined to use the Vantage internal compression scheme (for example, the JSON data type). Complex data types are provided by Vantage. They are similar in functionality to user-defined types because they follow an object-oriented model. |

### Possible Values for CharType

| Value | Description          |
|-------|----------------------|
| 1     | Latin                |
| 2     | Unicode              |
| 3     | KanjiSJIS            |
| 4     | Graphic              |
| 5     | Kanji1               |
| 0     | All other data types |

To get information about the storage format of a JSON type column, see Possible Values for StorageFormat.

### Possible Values for ColumnPartitionAC

| Value | Description            |
|-------|------------------------|
| NA    | Not column partitioned |
| NC    | No auto compress       |
| AC    | Auto compress          |

### Possible Values for ColumnPartitionFormat

| Value | Description                     |
|-------|---------------------------------|
| NA    | Not column partitioned          |
| CS    | System-determined COLUMN format |
| CU    | User-specified COLUMN format    |
| RS    | System-determined ROW format    |
| RU    | User-determined ROW format      |

## Possible Values for ColumnType

### Note:

UF, UV, LF, and LV are internally generated column types for Data Dictionary (database DBC) data only. For character user data, the field type is set to CF or CV.

| Value | Description                       |
|-------|-----------------------------------|
| ++    | TD_ANYTYPE                        |
| A1    | One dimensional ARRAY data type   |
| AN    | Multi-dimensional ARRAY data type |
| AT    | ANSI Time                         |
| BF    | BYTE Fixed                        |
| BO    | Byte Large Object                 |
| BV    | Byte Varying                      |
| CF    | Character Fixed                   |
| CO    | Character Large Object            |
| CV    | Character Varying Latin           |
| D     | Decimal                           |
| DA    | Date                              |
| DH    | Interval Day To Hour              |
| DM    | Interval Day To Minute            |
| DS    | Interval Day To Second            |
| DT    | DATASET type                      |
| DY    | Interval Day                      |
| F     | Float                             |
| HM    | Interval Hour To Minute           |
| HR    | Interval Hour                     |
| HS    | Interval Hour To Second           |
| I1    | 1 Byte Integer                    |
| I2    | 2 Byte Integer                    |
| I8    | 8 Byte Integer                    |

| Value | Description   |
|-------|---|
| I     | 4 Byte integer  |
| JN    | JSON document   |
| LF    | Pre-TD12.0 Character Fixed Locale (Kanji1 or Latin)<br><b>Note:</b><br>This column type is internally generated for Data Dictionary (database DBC) data only. |
| LV    | Pre-TD12.0 Character Varying Locale (Kanji1 or Latin)   |
| MI    | Interval Minute   |
| MO    | Interval Month  |
| MS    | Interval Minute To Second   |
| N     | Number  |
| PD    | PERIOD(DATE)  |
| PM    | PERIOD(TIMESTAMP(n) WITH TIMEZONE)  |
| PS    | PERIOD(TIMESTAMP (n))   |
| PT    | PERIOD(TIME(n))   |
| PZ    | PERIOD (TIME(n) WITH TIME ZONE)   |
| SC    | Interval Second   |
| SZ    | Timestamp With Time Zone  |
| TS    | Timestamp   |
| TZ    | ANSI Time With Time Zone  |
| UF    | Character Fixed Unicode<br><b>Note:</b><br>This column type is internally generated for Data Dictionary (database DBC) data only.                             |
| UT    | UDT Type  |
| UV    | Character Varying Unicode<br><b>Note:</b><br>This column type is internally generated for Data Dictionary (database DBC) data only.                           |
| VA    | TD_VALIST Type  |
| XM    | XML document  |
| YM    | Interval Year To Month  |
| YR    | Interval Year   |

**Possible Values for IdColType**

| Value | Description  |
|-------|--|
| NULL  | Non-identity column  |
| GA    | Generated always   |
| GE    | Generated always as row end. The required CREATE/ALTER TABLE (ANSI system-time table form) column attribute that defines the ending bound of a system-time period.<br>For more information about the CREATE/ALTER TABLE (ANSI system-time table form) statement, see <i>Teradata Vantage™ ANSI Temporal Table Support</i> , B035-1186 .      |
| GD    | Generated by default   |
| GS    | Generated always as row start. The required CREATE/ALTER TABLE (ANSI system-time table form) column attribute that defines the beginning bound of a system-time period.<br>For more information about the CREATE/ALTER TABLE (ANSI system-time table form) statement, see <i>Teradata Vantage™ ANSI Temporal Table Support</i> , B035-1186 . |

**Possible Values for PartitioningColumn**

| Value | Description                                      |
|-------|--|
| N     | Not a partitioning column                        |
| Y     | Partitioning column of a partitioning expression |

**Possible Values for PseudoUDTFieldType**

| Value | Description   |
|-------|---|
| PB    | Start column of the derived period column   |
| PE    | End column of the derived period column   |
| PP    | Derived period column   |
| NULL  | The column is not a start or end column of a derived period column or is a derived period column. |

**Possible Values for SPPParameterType**

| Value | Description                                    |
|-------|--|
| B     | INOUT parameter (for stored procedure only)    |
| C     | Column for TABLE function                      |
| E     | Result type of the external function or method |
| I     | Input parameter of a function or method        |

| Value | Description                           |
|-------|---------------------------------------|
| O     | OUT parameter of a function or method |
| S     | SELF parameter                        |

### Possible Values for StorageFormat

StorageFormat specifies the storage format of any column that has multiple storage formats (such as a JSON column), or NULL if the column does not have this characteristic.

| Value  | Description   |
|--------|---|
| NULL   | A column that does not have multiple storage formats.                                   |
| TEXT   | Column contains JSON content in JSON CHARACTER SET LATIN or JSON CHARACTER SET UNICODE. |
| BSON   | Column contains JSON content in Binary JSON format.                                     |
| UBJSON | Column contains JSON content in Universal Binary JSON format.                           |
| AVRO   | Column contains DATASET content stored in AVRO format.                                  |
| CSV    | Column contains DATASET content stored in CSV format.                                   |

There are several commands that provide storage format information:

- HELP TABLE
- HELP COLUMN
- HELP CAST
- HELP TYPE
- SHOW TABLE

HELP COLUMN returns information about the storage format of the JSON type; for example, NULL (for a column that does not have multiple storage formats), JSON stored as LATIN or UNICODE text, Binary JSON, or Universal Binary JSON format. Note, when running these commands the data type format is always JN for a JSON column, regardless of the storage format.

HELP TABLE provides information about the character set of the JSON type.

HELP COLUMN returns information about the storage format of the DATASET data type; for example, AVRO.

HELP TYPE is allowed on the DATASET type, but only when referenced using the name 'DATASET' followed by one of the storage formats of the type.

For more information see *Teradata Vantage™ SQL Data Definition Language Syntax and Examples*, B035-1144.

**Possible Values for TSColumnType**

| Value | Description   |
|-------|---|
| TB    | TD_TIMEBUCKET column  |
| TC    | TD_TIMECODE column  |
| TN    | TD_SEQNO column   |
| NULL  | Any columns that are not the TD_TIMEBUCKET, TD_TIMECODE, or TD_SEQNO column from a time series table. |

**Possible Values for TTCheckType**

| Value | Description   |
|-------|---|
| A     | ANSIQUALIFIER<br><b>Note:</b><br>The TTCheckType column returns the value A when the qualifier is ANSIQUALIFIER for column-level CHECK constraints. |
| NULL  | No transaction-time dimension   |
| C     | CURRENT TRANSACTIONTIME   |

**Possible Values for UpperCaseFlag****Note:**

Case flags U, C, and B are valid only for CHAR, VARCHAR, and LONG VARCHAR columns.

| Value | Description                 |
|-------|-----------------------------|
| U     | Uppercase, not specific     |
| C     | Not uppercase, specific     |
| N     | Not uppercase, not specific |
| B     | Both                        |

**Possible Values for VTCheckType**

| Value | Description    |
|-------|----------------|
| A     | ANSIQUALIFIER. |

| Value | Description  |
|-------|--|
|       | <b>Note:</b><br>The VTCheckType column returns the value A when the qualifier is ANSIQUALIFIER for column-level CHECK constraints. |
| NULL  | No valid-time dimension  |
| C     | CURRENT VALIDTIME  |
| S     | SEQUENCED VALIDTIME  |
| N     | NONSEQUENCED VALIDTIME   |

## Example: Using ColumnsV

This example shows a statement that selects from DBC.ColumnsV the name, format, null status, and data type of all columns in the Personnel.Employee table:

```
SELECT ColumnName,ColumnFormat,Nullable,ColumnType
      FROM DBC.ColumnsV WHERE DatabaseName='Personnel'
      AND TableName = 'Employee';
```

Partial Results:

| ColumnName | ColumnFormat | Nullable | ColumnType |
|------------|--------------|----------|------------|
| -----      | -----        | -----    | -----      |
| EmpNo      | 9(5)         | N        | I          |
| Name       | X(12)        | N        | CV         |
| DeptNo     | 999          | Y        | I          |
| JobTitle   | X(12)        | Y        | CV         |
| Salary     | zzz,zz9.99   | Y        | D          |
| YrsExp     | z9           | Y        | I          |

## Example: Select the CommentString Column from ColumnsV

This example shows a statement that selects any available commentary about columns in the Employee table:

```
SELECT ColumnName,CommentString FROM DBC.ColumnsV
      WHERE DatabaseName='Personnel' AND
      TableName='Employee'
      ORDER BY Columnid;
```

Result:

```

ColumnName      CommentString
-----
EmpNo           Employee serial number.
Name            Employee name, last then first initial.
DeptNo
JobTitle
Salary
YrsExp

```

## Related Topics

For more information about partitioning columns, see *Teradata Vantage™ SQL Data Definition Language Syntax and Examples*, B035-1144.

## ColumnUseCountV[X]

**Category:** Accounting

**Database:** DBC

| View Column  | Data Type   | Format                         | Comment  |
|--------------|---|--------------------------------|--|
| DatabaseName | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                         | Returns the Database Name of the object for which access count and/or UDI counts are recorded.   |
| TableName    | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                         | Returns the name of the table whose column was accessed by a query.  |
| FieldName    | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                         | Returns Field Name of a table if usage type is a DML type; StatsId if the usage is for statistics (the optimizer usage of statistics). |
| UsageType    | CHAR(3) LATIN<br>UPPERCASE NOT NULL                     | X(3)                           | Returns the usage type of the object. It can be either DML or STA.   |
| AccessCount  | BIGINT  | --,---,---,---,<br>---,---,--9 | Returns the number of accesses since the last reset by the user.   |

## Usage Notes

### Possible Values for UsageType

| Name | Description                   |
|------|-------------------------------|
| DML  | Data Manipulation Language    |
| STA  | Optimizer usage of statistics |

## Example: Using ColumnUseCountV

The following SELECT statement shows the number of accesses occurring on the columns of a particular table:

```
SELECT FieldName, AccessCount FROM DBC.ColumnUseCountV WHERE DatabaseName =
'Personnel' AND TableName = 'Employee';
```

Result:

```
FieldName  AccessCount
-----
id          15
name        13
```

## ConnectRulesV

**Category:** Security

**Database:** DBC

| View Column   | Data Type   | Format | Comment  |
|---------------|---|--------|--|
| TrustUser     | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | Returns the permanent user that has been granted the CONNECT THROUGH privilege.                        |
| ProxyUser     | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | Returns the name of the proxy user.  |
| ProxyUserType | VARCHAR(1)<br>UNICODE<br>NOT CASESPECIFIC               | X(1)   | Returns one of the following proxy user types: P - A Teradata permanent user, A - An application user. |

| View Column | Data Type                                     | Format | Comment  |
|-------------|---|--------|--|
| GrantStatus | CHAR(1) LATIN<br>NOT CASESPECIFIC<br>NOT NULL | X(1)   | Indicates whether the rule grants the CONNECT THROUGH privilege (G), or whether it has been revoked (R). |
| WithoutRole | CHAR(1) LATIN<br>NOT CASESPECIFIC<br>NOT NULL | X(1)   | Rule created with: Y (Without Role clause), N (Roles specified)  |
| ProxyRole1  | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC   | X(128) | Returns the role that can be set for the proxy user in a proxy connection.                               |
| ProxyRole2  | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC   | X(128) | The ConnectRulesV.ProxyRole2 field shows the name of a role granted to the proxy user.                   |
| ProxyRole3  | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC   | X(128) | The ConnectRulesV.ProxyRole3 field shows the name of a role granted to the proxy user.                   |
| ProxyRole4  | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC   | X(128) | The ConnectRulesV.ProxyRole4 field shows the name of a role granted to the proxy user.                   |
| ProxyRole5  | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC   | X(128) | The ConnectRulesV.ProxyRole5 field shows the name of a role granted to the proxy user.                   |
| ProxyRole6  | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC   | X(128) | The ConnectRulesV.ProxyRole6 field shows the name of a role granted to the proxy user.                   |
| ProxyRole7  | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC   | X(128) | The ConnectRulesV.ProxyRole7 field shows the name of a role granted to the proxy user.                   |
| ProxyRole8  | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC   | X(128) | The ConnectRulesV.ProxyRole8 field shows the name of a role granted to the proxy user.                   |
| ProxyRole9  | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC   | X(128) | The ConnectRulesV.ProxyRole9 field shows the name of a role granted to the proxy user.                   |
| ProxyRole10 | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC   | X(128) | The ConnectRulesV.ProxyRole10 field shows the name of a role granted to the proxy user.                  |
| ProxyRole11 | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC   | X(128) | The ConnectRulesV.ProxyRole11 field shows the name of a role granted to the proxy user.                  |
| ProxyRole12 | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC   | X(128) | The ConnectRulesV.ProxyRole12 field shows the name of a role granted to the proxy user.                  |

| View Column     | Data Type   | Format                 | Comment   |
|-----------------|---|------------------------|---|
| ProxyRole13     | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC             | X(128)                 | The ConnectRulesV.ProxyRole13 field shows the name of a role granted to the proxy user. |
| ProxyRole14     | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC             | X(128)                 | The ConnectRulesV.ProxyRole14 field shows the name of a role granted to the proxy user. |
| ProxyRole15     | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC             | X(128)                 | The ConnectRulesV.ProxyRole15 field shows the name of a role granted to the proxy user. |
| CreatorName     | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                 | Rule creator.   |
| CreateTimeStamp | TIMESTAMP(0)  | YYYY-MM-DD<br>HH:MI:SS | Returns the date and time that the object in the row was created.                       |
| TrustOnly       | CHAR(1) LATIN<br>NOT CASESPECIFIC<br>NOT NULL           | X(1)                   | Indicates whether the ProxyUser must be set in a Trusted request.                       |
| ProfileName     | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC             | X(128)                 | Returns the ProfileName associated with the application ProxyUser.                      |

## Example: Select the Connect Through Privileges for All Proxy Users from ConnectRulesV

The following select statement returns the connect through privileges for all proxy users:

```
SELECT TrustUser (FORMAT 'X(20)'), ProxyUser (FORMAT 'X(20)'),
       ProxyUserType FROM DBC.ConnectRulesV order by 2;
```

Result:

| TrustUser  | ProxyUser   | ProxyUserType |
|------------|-------------|---------------|
| -----      | -----       | -----         |
| trustuser1 | APPPXYUSER1 | A             |
| trustuser1 | APPPXYUSER2 | A             |
| trustuser1 | APPPXYUSER3 | A             |
| trustuser1 | APPPXYUSER4 | A             |
| trustuser1 | PXYUSER1    | P             |
| trustuser1 | PXYUSER2    | P             |

|            |          |   |
|------------|----------|---|
| trustuser1 | PXYUSER3 | P |
| trustuser1 | PXYUSER4 | P |

## Example: Select the Connect Through Privileges for a Proxy User from ConnectRulesV

The following select statement returns the connect through privilege for a specified proxy user including the granted roles:

```
SELECT * FROM DBC.ConnectRulesV WHERE ProxyUser='PxyUser3';
```

## ConstraintFunctionsV

**Category:** Integrity

**Database:** DBC

| View Column    | Data Type   | Format | Comment                                       |
|----------------|---|--------|---|
| ConstraintName | VARCHAR(128) UNICODE<br>UPPERCASE NOT NULL        | X(128) | User assigned name, in<br>input format        |
| Action         | CHAR(2) LATIN UPPERCASE<br>NOT NULL               | X(2)   | IN=insert, UP=update,<br>DE=delete, SE=select |
| DatabaseName   | VARCHAR(128) UNICODE NOT<br>CASESPECIFIC NOT NULL | X(128) | Name of database<br>containing function       |
| FunctionName   | VARCHAR(128) UNICODE NOT<br>CASESPECIFIC NOT NULL | X(128) | User assigned name, in<br>input format        |

## ConstraintValuesV

**Category:** Integrity

**Database:** DBC

| View Column    | Data Type   | Format | Comment                                       |
|----------------|---|--------|---|
| ConstraintName | VARCHAR(128) UNICODE<br>UPPERCASE NOT NULL        | X(128) | User assigned name, in input format           |
| ValueName      | VARCHAR(128) UNICODE NOT<br>CASESPECIFIC NOT NULL | X(128) | User assigned value name in<br>input format   |
| ValueConstant  | SMALLINT NOT NULL                                 | -(5)9  | User assigned code value<br>for constraint    |
| ValueIsBitPos  | CHAR(1) LATIN UPPERCASE                           | X(1)   | 'Y'- value is bit position, 'N' - is smallint |

## CostProfiles\_v

**Category:** Operations

**Database:** DBC

| View Column     | Data Type                                      | Format  | Comment  |
|-----------------|--|---------|--|
| ProfileTypeName | CHAR(30) LATIN<br>NOT CASESPECIFIC<br>NOT NULL | X(15)   | ProfileTypeName provides the type of cost profile for the cost profile instance.   |
| ProfileName     | CHAR(30) LATIN<br>NOT CASESPECIFIC<br>NOT NULL | X(30)   | ProfileName is the unique name assigned to this cost profile instance in the system.   |
| ProfileId       | INTEGER NOT NULL                               | -----9  | ProfileId is the unique number assigned to the cost profile instance in the system.  |
| ProfileCat      | CHAR(1) LATIN<br>NOT CASESPECIFIC<br>NOT NULL  | X(1)    | ProfileCat is the cost profile category: F-Profile instances which are fixed and cannot be changed; V-Profile instances which can be modified. |
| ProfileDesc     | VARCHAR(2048) LATIN<br>NOT CASESPECIFIC        | X(2048) | ProfileDesc is the description of the cost profile instance.   |

## Usage Notes

The Cost Profile views are only for the use of Teradata Support Center personnel.

## CostProfileTypes\_v

**Category:** Operations

**Database:** DBC

| View Column     | Data Type                                   | Format  | Comment  |
|-----------------|---|---------|--|
| ProfileTypeName | CHAR(30) LATIN NOT<br>CASESPECIFIC NOT NULL | X(30)   | ProfileTypeName provides the name of a type of cost profile.               |
| ProfileTypeDesc | VARCHAR(2048) LATIN<br>NOT CASESPECIFIC     | X(2048) | ProfileTypeDesc provides a description of the associated costprofile type. |

## Usage Notes

The Cost Profile views are only for the use of Teradata Support Center personnel.

## CostProfileValues\_v

**Category:** Operations

**Database:** DBC

| View Column | Data Type                                      | Format                | Comment  |
|-------------|--|-----------------------|--|
| ProfileName | CHAR(30) LATIN<br>NOT CASESPECIFIC<br>NOT NULL | X(15)                 | ProfileName is the unique name assigned to this cost profile instance in the system.   |
| ProfileId   | INTEGER NOT NULL                               | -----9                | ProfileId is the unique number assigned to the cost profile instance in the system.  |
| ConstName   | CHAR(64) LATIN<br>NOT CASESPECIFIC<br>NOT NULL | X(30)                 | ConstName is the name associated with this constant.   |
| ConstId     | INTEGER NOT NULL                               | -----9                | ConstId identifies this constant within the cost profile.  |
| ConstCat    | CHAR(1) LATIN<br>NOT CASESPECIFIC<br>NOT NULL  | X(1)                  | ConstCat is the category for this constant. For Legacy and Subops profile types, the constant categories are: C-Cost Method Parameter, I-Initialization Parameter. |
| ConstVal    | FLOAT  | ----,---,--9.<br>9999 | ConstVal is the specified value of the constant in this cost profile instance. A NULL value implies there is no specified value.                                   |
| ConstDesc   | VARCHAR(2048)<br>UNICODE<br>NOT CASESPECIFIC   | X(2048)               | ConstDesc is the description associated with this constant.  |

## Usage Notes

The Cost Profile views are only for the use of Teradata Support Center personnel.

## Database\_Default\_JournalsV[X]

**Category:** Database

**Database:** DBC

| View Column  | Data Type  | Format | Comment   |
|--------------|--|--------|---|
| DatabaseName | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128) | Returns the name of the database for which a default journal table has been defined.                        |
| Journal_DB   | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128) | Returns the name of the database or user space in which the default journal table for DatabaseName resides. |

| View Column | Data Type  | Format | Comment  |
|-------------|--|--------|--|
| JournalName | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128) | Returns the name of the journal table defined as the default for UserName. |

## Usage Notes

### Corresponding Tables

The X view references these additional tables:

- DBC.AccessRights
- DBC.Owners
- DBC.RoleGrants
- DBC.Roles

## Example: Using Database\_Default\_JournalsV

The following SELECT statement selects the information on each database accessible by the requesting user for which a default journal table is defined:

```
SELECT * FROM DBC.Database_Default_JournalsV;
```

Result:

| DatabaseName | Journal_DB | JournalName |
|--------------|------------|-------------|
| DtBs1        | DtBs1      | DtBs1Jrn1   |
| DtBs2        | DtBs2      | DtBs2Jrn1   |
| DtBs3        | DtBs1      | DtBs1Jrn1   |

## Databases2V[X]

**Category:** Database

**Database:** DBC

| View Column  | Data Type                                     | Format | Comment   |
|--------------|---|--------|---|
| DatabaseName | VARCHAR(128)<br>UNICODE UPPERCASE<br>NOT NULL | X(128) | Returns the name of the database with the indicated count of unresolved references. |
| Databaseld   | BYTE(4) NOT NULL                              | X(8)   | Returns the ID of the database with the indicated count of unresolved references.   |

| View Column      | Data Type         | Format  | Comment  |
|------------------|-------------------|---------|--|
| UnResolvedRCount | SMALLINT NOT NULL | ---,--9 | Returns the total number of unresolved Referential Integrity (RI) constraints in the database. |

## Usage Notes

The administrator can control who has access to internal ID numbers by limiting the access to the Databases2 view while allowing more (or all) users to access the names via the Databases view.

## Corresponding Tables

The X view references these additional tables:

- DBC.AccessRights
- DBC.Owners
- DBC.RoleGrants
- DBC.Roles

## DatabasesV[X]

**Category:** Database

**Database:** DBC

| View Column  | Data Type   | Format | Comment  |
|--------------|---|--------|--|
| DatabaseName | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | Returns the name of a database or one of the special system keywords: ALL / DEFAULT / PUBLIC.  |
| CreatorName  | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | Returns the Teradata user who issued the CREATE DATABASE statement.  |
| OwnerName    | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | Database owner name.   |
| AccountName  | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | Expanded account in effect when a request was submitted. SYSTEMACCOUNTNO tracks console utility activity such as table rebuild, Diskcopy, or Scandisk. |

| View Column         | Data Type   | Format                         | Comment  |
|---------------------|---|--------------------------------|--|
| ProtectionType      | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL                  | X(1)                           | Returns F (Fallback) or N (None) to indicate whether the tables in the database are protected by the Fallback option.  |
| JournalFlag         | CHAR(2) LATIN<br>UPPERCASE<br>NOT NULL                  | X(2)                           | Journaling in effect for the table, or the journal default for tables. Settings: first character BEFORE, second character AFTER.   |
| PermSpace           | BIGINT NOT NULL   | --,---,---,---,<br>---,---,--9 | Returns an integer indicating the total space allocated to the database on all AMPs.   |
| SpoolSpace          | BIGINT NOT NULL   | --,---,---,---,<br>---,---,--9 | Returns an integer indicating the maximum spool space allowed for the database. SpoolSpace is 0 if DatabaseName is PUBLIC.   |
| TempSpace           | BIGINT NOT NULL   | --,---,---,---,<br>---,---,--9 | Returns the maximum temporary space allocated for a database, profile, or user in bytes.   |
| CommentString       | VARCHAR(255)<br>UNICODE NOT<br>CASESPECIFIC             | X(255)                         | Returns user-supplied text or commentary on the column, database, table, view, macro, user-defined function, user-defined types, user-defined methods, stored procedure, role, profile, or user. |
| CreateTimeStamp     | TIMESTAMP(0)  | YYYY-MM-<br>DDBHH:<br>MI:SS    | Returns the date and time that the object in the row was created.  |
| LastAlterName       | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                         | Returns the name of the user who last updated the dictionary row.  |
| LastAlterTimeStamp  | TIMESTAMP(0)  | YYYY-MM-<br>DDBHH:<br>MI:SS    | Returns the time the dictionary row was last updated.  |
| DBKind              | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL                  | X(1)                           | Returns information that indicates whether the row information represents a database (D) or a user (U).  |
| AccessCount         | BIGINT  | --,---,---,---,<br>---,---,--9 | Returns the access count for the corresponding database object.  |
| LastAccessTimeStamp | TIMESTAMP(0)  | YYYY-MM-<br>DDBHH:<br>MI:SS    | Returns the time that the corresponding object was last accessed.  |

| View Column               | Data Type                                   | Format         | Comment   |
|---------------------------|---|----------------|---|
| DefaultMapName            | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128)         | Returns the default mapname for the database or user. NULL indicates the database or user does not have a default map.                  |
| MapOverride               | VARCHAR(1)<br>LATIN NOT<br>CASESPECIFIC     | X(1)           | MapOverride indicates no override or override on error for the default map. "N" indicates no override. "E" indicates OVERRIDE ON ERROR. |
| IncrementalRestoreEnabled | INTEGER                                     | --,---,---,--9 | Indicates if Incremental Restore has been enabled for the database.   |

## Usage Notes

You can use this view to query the characteristics of the databases you own and which you have privileges on.

For information about the possible values for the JournalFlag column, see "JournalFlag Column."

## Corresponding Tables

The X view references these additional tables:

- DBC.AccessRights
- DBC.Owners
- DBC.RoleGrants
- DBC.Roles

## JournalFlag

The indicators in the JournalFlag column depend on:

- The FALLBACK and JOURNAL settings for the database, which serve as the default for all tables created in that database
- Any FALLBACK and JOURNAL settings defined in the CREATE TABLE and ALTER TABLE statements

The settings defined for an individual table override the database defaults on platforms where NO FALLBACK is allowed. For all the account names associated with a database or user, use the AccountInfo[X] system view.

## Example: Using DatabasesV

The statement shown in the following selects information about the Personnel database:

```
SELECT AccountName,ProtectionType,PermSpace,SpoolSpace
      FROM DBC.DatabasesV WHERE DatabaseName = 'Personnel';
```

Result:

| AccountName     | ProtectionType | PermSpace | SpoolSpace    |
|-----------------|----------------|-----------|---------------|
| -----           | -----          | -----     | -----         |
| Teradata_Sample | F              | 100,000   | 1,339,884,032 |

## DatabaseUseCountV[X]

**Category:** Accounting

**Database:** DBC

| View Column  | Data Type   | Format                         | Comment  |
|--------------|---|--------------------------------|--|
| DatabaseName | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                         | Returns the Database Name of the object for which access count and/or UDI counts are recorded. |
| UsageType    | CHAR(3) LATIN<br>UPPERCASE<br>NOT NULL                  | X(3)                           | Returns the usage type of the object. It can be either DML or STA.                             |
| AccessCount  | BIGINT  | --,---,---,---,---,<br>---,--9 | Returns the number of accesses since the last reset by the user.                               |

## Usage Notes

### Possible Values for UsageType

| Name | Description                   |
|------|-------------------------------|
| DML  | Data Manipulation Language    |
| STA  | Optimizer usage of statistics |

## Example: Using DatabaseUseCountV

The following SELECT statement shows the number of accesses occurring on a particular database:

```
SELECT AccessCount FROM DBC.DatabaseUseCountV WHERE DatabaseName = 'Personnel';
```

Result:

AccessCount

-----  
16

## DatasetSchemaDependenciesV

**Category:** Dataset Schema

**Database:** DBC

| View Column       | Data Type                            | Format | Comment   |
|-------------------|--------------------------------------|--------|---|
| DatabaseName      | VARCHAR(128)<br>UNICODE<br>UPPERCASE | X(128) | Returns the database name of the object which is dependent on a schema  |
| TableName         | VARCHAR(128)<br>UNICODE<br>UPPERCASE | X(128) | Returns the name of the table which is dependent on a schema  |
| DatasetSchemaName | VARCHAR(128)<br>UNICODE<br>UPPERCASE | X(128) | Returns the name of the schema upon which a table depends. The name corresponds to the column TVMName in the DBC. TVM table |

## Usage Notes

DatasetSchemaDependenciesV is used to identify dependencies on schemas registered by CREATE SCHEMA and associated with a particular column.

## DatasetSchemaInfoV

**Category:** Dataset Schema

**Database:** DBC

| View Column       | Data Type                                  | Format | Comment  |
|-------------------|--|--------|--|
| DatasetSchemaId   | BYTE(6) NOT NULL                           | X(12)  | Returns the identifier for the Schema  |
| DatabaseName      | VARCHAR(128) UNICODE<br>UPPERCASE NOT NULL | X(128) | Returns the name of the database in which the schema is defined                          |
| DatasetSchemaName | VARCHAR(128) UNICODE<br>UPPERCASE NOT NULL | X(128) | Returns the schema name. The name corresponds to the column TVMName in the DBC.TVM table |
| StorageFormat     | VARCHAR(5) UNICODE<br>NOT CASESPECIFIC     | X(5)   | Returns the storage format for which the schema is specified                             |

| View Column       | Data Type                  | Format   | Comment                         |
|-------------------|----------------------------|----------|---------------------------------|
| DatasetSchemaUTF8 | VARBYTE(10000)<br>NOT NULL | X(20000) | Returns the UTF8 encoded schema |

## Usage Notes

The DBC.DatasetSchemaInfo table is used to register schema definitions created by the CREATE SCHEMA statement. The DatasetSchemaInfoV view allows users to get information about registered schemas from the table.

### Possible Values for StorageFormat

StorageFormat specifies the storage format of any column with multiple storage formats.

| Value | Description  |
|-------|--|
| AVRO  | Column contains DATASET content stored in AVRO format. |
| CSV   | Column contains DATASET content stored in CSV format.  |
| NULL  | The column does not have multiple storage formats.     |

## DBCInfoV

**Category:** Operations

**Database:** DBC

| View Column | Data Type                                   | Format   | Comment  |
|-------------|---|----------|--|
| InfoKey     | VARCHAR(30) LATIN NOT CASESPECIFIC NOT NULL | X(30)    | Returns the key identifying the attribute value in the InfoData field. |
| InfoData    | VARCHAR(16384) UNICODE NOT CASESPECIFIC     | X(16384) | Returns the attribute identified by the value of the InfoKey field.    |

## Example: Using DBCInfoV

This SELECT statement retrieves the version and release of the current Advanced SQL Engine software.

```
SELECT infokey (format 'x(30)'),
       infodata (format 'x(20)')
FROM DBC.DBCInfoV
ORDER BY infokey;
```

Result:

| InfoKey               | InfoData     |
|-----------------------|--------------|
| -----                 | -----        |
| LANGUAGE SUPPORT MODE | Japanese     |
| RELEASE               | 14.10j.00.00 |
| VERSION               | 14.10j.00.00 |

For the language support mode, InfoData is either Standard or Japanese.

The values for Version and Release are broken down into 4 parts:

- Major release
- Minor release
- Maintenance release
- E-fix release

For example, 14.10.02.03 represents the following:

| Value | Description                   |
|-------|-------------------------------|
| 14    | Major release number          |
| 10    | Minor release number          |
| 02    | Maintenance release number    |
| 03    | E-fix or patch release number |

## DBQLRulesV

**Category:** Query

**Database:** DBC

| View Column   | Data Type   | Format | Comment  |
|---------------|---|--------|--|
| UserName      | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | Returns the name of the user associated with the rule.   |
| AccountString | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | Returns the specific account for the user with a DBQL rule, or blank if all account strings apply. |
| ApplName      | CHAR(30) LATIN<br>NOT CASESPECIFIC<br>NOT NULL          | X(30)  | Returns the name of the application.   |

| View Column    | Data Type                                  | Format | Comment  |
|----------------|--|--------|--|
| TypeOfRule     | VARCHAR(22)<br>UNICODE<br>NOT CASESPECIFIC | X(22)  | Indicates whether logging is enabled or disabled by this rule.   |
| ExplainFlag    | VARCHAR(1)<br>UNICODE<br>NOT CASESPECIFIC  | X(1)   | The following codes are used: T = Explain text is stored in DBC.DBQLExplainTbl F = No Explain text is provided.  |
| ObjFlag        | VARCHAR(1)<br>UNICODE<br>NOT CASESPECIFIC  | X(1)   | are stored in DBC.DBQLObjTbl, F = No object data.  |
| SqlFlag        | VARCHAR(1)<br>UNICODE<br>NOT CASESPECIFIC  | X(1)   | SQL text is stored in DBC.DBQLSqlTbl: T (yes), F (no).   |
| StepFlag       | VARCHAR(1)<br>UNICODE<br>NOT CASESPECIFIC  | X(1)   | The following codes are used: T = Step-level data is stored in DBC.DBQLStepTbl, F = Step-level data is not provided in DBC.DBQLStepTbl.  |
| XMLPlanFlag    | VARCHAR(1)<br>UNICODE<br>NOT CASESPECIFIC  | X(1)   | Indicates whether XML query plan logging is on or off. These are the values: T = On, F = Off.  |
| StatsUsageFlag | VARCHAR(1)<br>UNICODE<br>NOT CASESPECIFIC  | X(1)   | StatsUsage: T = Statistics usage will be stored, F = No statistics usage.  |
| Verbose        | VARCHAR(1)<br>UNICODE<br>NOT CASESPECIFIC  | X(1)   | Verbose: T = Verbose details provided, F = No verbose details provided   |
| DetailedStats  | VARCHAR(1)<br>UNICODE<br>NOT CASESPECIFIC  | X(1)   | DetailedStats: T = Statistics details provided, F = No Statistics details provided   |
| NoColumns      | VARCHAR(1)<br>UNICODE<br>NOT CASESPECIFIC  | X(1)   | NoColumns: T = Column Details are not logged, F = Column Details are logged  |
| SummaryFlag    | VARCHAR(1)<br>UNICODE<br>NOT CASESPECIFIC  | X(1)   | The following codes are used: T = Summary information is stored in DBC.DBQLSummaryTbl, F = Data is not summarized.   |
| ThresholdFlag  | VARCHAR(1)<br>UNICODE<br>NOT CASESPECIFIC  | X(1)   | T (queries shorter or equal to SummaryVal1 in DBC.DBQLSummaryTbl, detailed data on long queries in DBC.DBQLLogTbl), F (detailed data for all queries in DBC.DBQLLogTbl unless SummaryFlag is True) |
| ObjectUsage    | VARCHAR(1)<br>UNICODE<br>NOT CASESPECIFIC  | X(1)   | ObjectUsage: T = Object use counts will be collected, F = No Object use count collection.  |

| View Column     | Data Type                                  | Format         | Comment   |
|-----------------|--|----------------|---|
| ParamFlag       | VARCHAR(1)<br>UNICODE<br>NOT CASESPECIFIC  | X(1)           | Param: T = Parameters and values will be stored, F = No parameter logging.  |
| FeatureUsage    | VARCHAR(1)<br>UNICODE<br>NOT CASESPECIFIC  | X(1)           | FeatureUsage: T = Feature usage will be collected, F = No Feature usage logging.  |
| UtilityInfoFlag | VARCHAR(1)<br>UNICODE<br>NOT CASESPECIFIC  | X(1)           | Indicates whether utility information will be collected.  |
| TextSizeLimit   | INTEGER                                    | --,---,---,--9 | Returns the number of characters of SQL text to be stored in the DBQLogTbl.   |
| SummaryVal1     | SMALLINT                                   | ---,--9        | If Summary or Threshold are T, values <= SummaryVal1 in Elapsed, CPU or IO,   |
| SummaryVal2     | SMALLINT                                   | ---,--9        | Group 2: SummaryVal1 < values <= SummaryVal2 in Elapsed, CPU or IO,   |
| SummaryVal3     | SMALLINT                                   | ---,--9        | Group 3: SummaryVal2 < values <= SummaryVal3 in Elapsed, CPU or IO, & 4: larger than SummaryVal3 in Elapsed, CPU or IO. |
| LockDelay       | INTEGER                                    | --,---,---,--9 | Returns the information if the LockDelay is captured as part of the DBQL query logging rule(renamed to LockDelay).      |
| AlgMode         | BYTEINT                                    | -(3)9          | The DBQL CPU/IO Collection Algorithm to use for this rule.  |
| TypeOfCriterion | VARCHAR(22)<br>UNICODE<br>NOT CASESPECIFIC | X(22)          | Returns a criterion used for Summary or Threshold.  |
| DetailDiag      | INTEGER                                    | --,---,---,--9 | Detail Diagnostic flags   |

## Usage Notes

Only users with DBC or SystemFE privileges are able to use this view.

### LockDelay

The LockDelay column returns the minimum elapsed time of a lock contention in centiseconds. Any lock contention greater than this time will be recorded in the DBC.DBQLXMLLockTbl table as specified in the BEGIN QUERY LOGGING WITH LOCK statement. The minimum LockDelay value is 5 centiseconds.

**Possible Values for DetailedStats**

| Value | Description                           |
|-------|---------------------------------------|
| T     | Detailed statistics are provided.     |
| F     | Detailed statistics are not provided. |

**Possible Values for ExplainFlag**

| Value | Description                   |
|-------|-------------------------------|
| T     | EXPLAIN text is logged.       |
| F     | EXPLAIN text is not provided. |

**Possible Values for FeatureUsage**

| Value | Description                                 |
|-------|---|
| T     | Feature usage information is collected.     |
| F     | Feature usage information is not collected. |

**Possible Values for NoColumns**

The NoColumns column indicates if columns are logged as part of object logging. When query logging is enabled with the OBJECTS option, a separate row for each table and column referenced is added to the DBQL object table. With NoColumns set to T, the columns are not logged.

| Value | Description                                      |
|-------|--|
| T     | Query logging is enabled without column logging. |
| F     | Query logging is enabled with column logging.    |

**Possible Values for ParamFlag**

ParamFlag shows whether parameters and values are stored for parameterized requests.

| Value | Description                                 |
|-------|---|
| T     | Query parameters and values are logged.     |
| F     | Query parameters and values are not logged. |

**Possible Values for ObjFlag**

ObjFlag shows whether data (columns and indexes) are stored.

| Value | Description                |
|-------|----------------------------|
| T     | Object data is logged.     |
| F     | Object data is not logged. |

### Possible Values for ObjectUsage

| Value | Description                       |
|-------|-----------------------------------|
| T     | Object use counts are logged.     |
| F     | Object use counts are not logged. |

### Possible Values for SQLFlag

| Value | Description               |
|-------|---------------------------|
| T     | SQL text is logged.       |
| F     | SQL text is not provided. |

### Possible Values for StatsUsageFlag

| Value | Description                     |
|-------|---------------------------------|
| T     | Statistics usage is logged.     |
| F     | Statistics usage is not logged. |

### Possible Values for StepFlag

| Value | Description                      |
|-------|----------------------------------|
| T     | Step-level data is logged.       |
| F     | Step-level data is not provided. |

### Possible Values for SummaryFlag

| Value | Description                          |
|-------|--------------------------------------|
| T     | Summary information is logged.       |
| F     | Summary information is not provided. |

**Possible Values for ThresholdFlag**

| Value | Description   |
|-------|---|
| T     | Counts the queries that are shorter or equal to SummaryVal1 in DBC.DBQLSummaryTbl. Provides detailed data on longer queries in DBC.DBQLogTbl. |
| F     | Provides detailed data for all queries in DBC.DBQLogTbl unless the SummaryFlag is True.   |

**Possible Values for TypeOfCriterion**

| Value | Description   |
|-------|---|
| 0     | ElapsedSec (Elapsed seconds)  |
| 1     | CPUTime (CPU time in centiseconds)  |
| 2     | IOCount (I/O count)   |
| 3     | CPUTimeNorm (Normalized CPU time in centiseconds for coexistence systems) |
| 4     | ElapsedTime (Elapsed time in centiseconds)                                |
| 5     | No logging criterion.   |

**Possible Values for TypeOfRule**

- Logging enabled
- WITH NONE (No logging)

**Possible Values for UtilityInfoFlag**

| Value | Description                             |
|-------|---|
| T     | TASM utility statistics are logged.     |
| F     | TASM utility statistics are not logged. |

**Possible Values for Verbose**

| Value | Description   |
|-------|---|
| T     | Verbose EXPLAIN and related information in XML format are provided.     |
| F     | Verbose EXPLAIN and related information in XML format are not provided. |

**Possible Values for XMLPlanFlag**

| Value | Description                 |
|-------|-----------------------------|
| T     | XML query plans are logged. |

| Value | Description                     |
|-------|---------------------------------|
| F     | XML query plans are not logged. |

### SummaryVal1, SummaryVal2, and SummaryVal3

| Column      | Description  |
|-------------|--|
| SummaryVal1 | Returns the high value for interval 1 if the SummaryFlag or ThresholdFlag columns are T. This value can be in seconds, CPU centiseconds, or I/Os.  |
| SummaryVal2 | Returns the high value for interval 2 if the SummaryFlag or ThresholdFlag columns are T. This value can be in seconds, CPU centiseconds, or I/Os (see TypeOfCriterion).  |
| SummaryVal3 | Returns the high value for interval 3 if the SummaryFlag or ThresholdFlag columns are T. The same value is used for interval 4. This value can be in seconds, CPU centiseconds, or I/Os (see TypeOfCriterion). |

### Example: Using DBQLRulesV

The following SELECT statement retrieves the rules in effect for users:

```
SELECT * from DBC.DBQLRulesV;
```

Result:

```

      UserName  All
      Account
ApplicationName
      TypeOfRule Logging enabled
      Explain   F
      Object    F
      SQL       F
      Step      T
      XMLPlan   F
      StatsUsage F
      Verbose    F
DetailedStats  F
      NoColumns  F
      Summary    F
      Threshold  F
      ObjectUsage F
      Param      F
      FeatureUsage T

```

|                 |              |     |
|-----------------|--------------|-----|
| UtilityInfo     | F            |     |
| TextSize        |              | 200 |
| Summary//Low    | ?            |     |
| Med             | ?            |     |
| High            | ?            |     |
| LockDelay       |              | 0   |
| AlgMode         | ?            |     |
| TypeOfCriterion | NO CRITERION |     |
| DetailDiag      |              | ?   |

**Note:**

This rule was initiated as:

```
replace query logging with stepinfo, featureinfo on all;
```

The statement enables STEP and Feature Usage logging.

## DeleteAccessLogV

**Category:** Security

**Database:** DBC

| View Column | Data Type      | Format   | Comment  |
|-------------|----------------|----------|--|
| LogDate     | DATE NOT NULL  | YY/MM/DD | Returns the date that the access log entry was made.         |
| LogTime     | FLOAT NOT NULL | 99:99:99 | Returns the time of day that the event occurred as HH:MM:SS. |

## Usage Notes

The access log contains entries according to the application of the access logging rules (see [AccessLogV](#) and [AccLogRulesV](#)).

The view also may be used to display information about records that are eligible for deletion before the delete operation is performed.

## Example: Using DeleteAccessLogV

The following statement deletes entries logged more than 30 days before the current calendar date:

```
DELETE FROM DBC.DeleteAccessLogV ALL;
```

Result:

DELETE COMPLETED. 79 RECORDS DELETED.

## DeleteOldInDoubtV

**Category:** Operations

**Database:** DBC

| View Column            | Data Type   | Format         | Comment   |
|------------------------|---|----------------|---|
| LogicalHostId          | SMALLINT<br>NOT NULL                                    | ---,--9        | Returns a unique identifier of the logon source for the logged query. A value of zero indicates an internal session.                                  |
| SessionNumber          | INTEGER NOT NULL  | --,---,---,--9 | Note: SessionID, SessionNo, and SessionNumber have the same meaning. Returns the session identifier of the session that had the in-doubt transaction. |
| CoordTaskId            | VARBYTE(30)<br>NOT NULL                                 | X(60)          | column identifies the coordinator Id of the in-doubt session.   |
| RunUnitId              | VARBYTE(30)<br>NOT NULL                                 | X(60)          | Returns the identity of the run unit that had the in-doubt transaction.   |
| LogonUserName          | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)         | Returns the ID of the user who ran the in-doubt transaction.  |
| ResolvingUserLogonName | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)         | Returns the identity of the user who resolved the in-doubt transaction.   |
| CommitOrRollback       | CHAR(1) LATIN<br>NOT CASESPECIFIC<br>NOT NULL           | X(1)           | This column identifies whether the in-doubt session was committed (C) or rolled back (R).   |
| UserLogonDate          | DATE NOT NULL   | YY/MM/DD       | Returns the date that the specified user logged on.   |
| UserLogonTime          | FLOAT NOT NULL  | 99:99:99       | Returns the time that the specified user logged on.   |
| CompletionDate         | DATE NOT NULL   | YY/MM/DD       | This column identifies the time the in-doubt session was resolved.  |
| CompletionTime         | FLOAT NOT NULL  | 99:99:99       | This column identifies the time the in-doubt session was resolved.  |
| Options                | CHAR(1) LATIN<br>NOT CASESPECIFIC                       | X(1)           | This field is assigned to NULL initially. It is not used by the system  |

| View Column | Data Type | Format | Comment   |
|-------------|-----------|--------|---|
|             |           |        | and users may update this one-character column to suit their needs. |

## Usage Notes

The DeleteOldInDoubt view purges entries from the in-doubt transaction log that are more than 30 days old. Before a delete operation is performed the view may also be used to display information about records eligible for deletion.

### Possible Values for CommitOrRollback

| Value | Description |
|-------|-------------|
| C     | Committed   |
| R     | Rolled back |

## Example: Using DeleteOldInDoubtV

The following statement deletes entries logged against in-doubt transactions that were entered more than 30 days before the current calendar date:

```
DELETE FROM DBC.DeleteOldInDoubtV ALL;
```

Result:

```
DELETE COMPLETED. 5 ROWS REMOVED.
```

## DeleteUseCountV[X]

**Category:** Accounting

**Database:** DBC

| View Column  | Data Type   | Format | Comment  |
|--------------|---|--------|--|
| DatabaseName | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | Returns the Database Name of the object for which access count and/or UDI counts are recorded. |
| ObjectName   | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | Returns the Object Name of the object for which access count and/or UDI counts are recorded.   |

| View Column | Data Type                           | Format                         | Comment  |
|-------------|-------------------------------------|--------------------------------|--|
| UsageType   | CHAR(3) LATIN<br>UPPERCASE NOT NULL | X(3)                           | Returns the usage type of the object. It can be either DML or STA. |
| DeleteCount | BIGINT                              | --,---,---,---,---,<br>---,--9 | Returns the number of deletes since the last reset by the user.    |

## Usage Notes

### Possible Values for UsageType

| Name | Description                   |
|------|-------------------------------|
| DML  | Data Manipulation Language    |
| STA  | Optimizer usage of statistics |

## Example: Using DeleteUseCountV

The following SELECT statement shows the number of deletes occurring on a particular object:

```
SELECT DeleteCount FROM DBC.DeleteUseCountV WHERE DatabaseName = 'Personnel'
AND ObjectName = 'Employee';
```

Result:

| DeleteCount |
|-------------|
| -----       |
| 4           |

## DiskGlobalSpaceErrorV

**Category:** Accounting

**Database:** DBC

| View Column  | Data Type   | Format                         | Comment   |
|--------------|---|--------------------------------|---|
| DatabaseName | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                         | Returns the name of a database or one of the special system keywords: ALL / DEFAULT / PUBLIC. |
| MaxPerm      | BIGINT NOT NULL   | --,---,---,---,---,<br>---,--9 | Returns the maximum allocation in bytes for permanent and user defined temporary table space. |

| View Column   | Data Type       | Format                         | Comment   |
|---------------|-----------------|--------------------------------|---|
| CurrentPerm   | BIGINT          | --,---,---,---,---,<br>---,--9 | Returns the maximum CurrentPermSpace in bytes for permanent and user defined temporary table space. |
| AllocatedPerm | BIGINT NOT NULL | --,---,---,---,---,<br>---,--9 | Returns the current total allocation in bytes for permanent and user defined temporary table space. |

## Usage Notes

DiskGlobalSpaceErrorV returns the databases that exceeded the permissible space usage at the system level. This view is similar to DiskSpaceErrorV, except that the AMP level information is not provided. The skew is not provided in this view because skew applies at the AMP level, not the system level.

## DiskSpaceErrorV

**Category:** Accounting

**Database:** DBC

| View Column   | Data Type   | Format                         | Comment   |
|---------------|---|--------------------------------|---|
| Vproc         | INTEGER NOT NULL  | --,---,---,--9                 | The DiskSpaceErrorV.Vproc field identifies the AMP Vproc reporting the space.   |
| DatabaseName  | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                         | Returns the name of a database.   |
| MaxPerm       | BIGINT  | --,---,---,---,---,---,<br>--9 | The DiskSpaceErrorV.MaxPerm field specifies the permanent space allocated to the database per AMP.                      |
| CurrentPerm   | BIGINT  | --,---,---,---,---,---,<br>--9 | The DiskSpaceErrorV.CurrentPerm field gives the amount of permanent space currently being used by the database per AMP. |
| AllocatedPerm | BIGINT  | --,---,---,---,---,---,<br>--9 | Returns the current total allocation in bytes for permanent and user defined temporary table space.                     |

## Usage Notes

Use DBC.DiskSpaceErrorV to determine if any database has exceeded its maximum perm space or permissible skew limits. If so, you must manually adjust the maxperm space for those databases to prevent future database updates from exceeding the maximum and experiencing out of space errors.

Some applications (such as FastLoad, MultiLoad, and restore operations) may require more temporary space during processing.

Fields in DBS Control can be set to allow applications to exceed the maxperm space as long as there is physically enough disk space available. After the operation completes, reset the fields.

## Related Topics

| For more information on...  | See...  |
|---|---|
| Setting the DBS Control field GlobalSpaceSoftLimitPercent to increase the space limit for permanent, spool, and temporary space | <i>Teradata Vantage™ - Database Utilities</i> , B035-1102 .     |
| Update Space utility  | <i>Teradata Vantage™ - Database Utilities</i> , B035-1102 .     |
| Database space accounting   | <i>Teradata Vantage™ - Database Administration</i> , B035-1093. |

## DiskSpaceV[X]

**Category:** Accounting

**Database:** DBC

| View Column  | Data Type  | Format                         | Comment  |
|--------------|--|--------------------------------|--|
| Vproc        | INTEGER NOT NULL                                     | --,---,---,--9                 | Identifies the virtual processor for which an event was logged.  |
| DatabaseName | VARCHAR(128)<br>UNICODE NOT CASESPECIFIC<br>NOT NULL | X(128)                         | Returns the name of a database or one of the special system keywords: ALL / DEFAULT / PUBLIC.  |
| AccountName  | VARCHAR(128)<br>UNICODE NOT CASESPECIFIC<br>NOT NULL | X(128)                         | Expanded account in effect when a request was submitted. SYSTEMACCOUNTNO tracks console utility activity such as table rebuild, Diskcopy, or Scandisk.                             |
| MaxPerm      | BIGINT   | --,---,---,---,<br>---,---,--9 | Returns an integer that represents the maximum permanent space, in bytes, that is allocated to the database on a specified AMP (or on all AMPs if the SUM aggregate is specified). |
| MaxSpool     | BIGINT   | --,---,---,---,<br>---,---,--9 | Returns an integer that represents the maximum spool space, in bytes, that is allocated to the database on a specified AMP (or on all AMPs if the SUM aggregate is specified).     |

| View Column            | Data Type | Format                         | Comment   |
|------------------------|-----------|--------------------------------|---|
| MaxTemp                | BIGINT    | --,---,---,---,<br>---,---,--9 | Returns an integer that represents the maximum temporary space, in bytes, that is allocated to the database on a specified AMP (or on all AMPs if the SUM aggregate is specified).                    |
| CurrentPerm            | BIGINT    | --,---,---,---,<br>---,---,--9 | Returns the permanent space (in bytes) currently used by the database or table.   |
| CurrentSpool           | BIGINT    | --,---,---,---,<br>---,---,--9 | Returns the spool space (in bytes) currently used by the database on a specified AMP (or on all AMPs if the SUM aggregate is specified).  |
| CurrentPersistentSpool | BIGINT    | --,---,---,---,<br>---,---,--9 | Returns the persistent spool space (in bytes) currently used by the database on a specified AMP (or on all AMPs if the SUM aggregate is specified).   |
| CurrentTemp            | BIGINT    | --,---,---,---,<br>---,---,--9 | Returns the number of bytes currently used by a temporary table per vproc.  |
| PeakPerm               | BIGINT    | --,---,---,---,<br>---,---,--9 | Returns the maximum amount of permanent space per AMP that has been used by the database since the last time the DBC.ClearPeakDisk macro was run.   |
| PeakSpool              | BIGINT    | --,---,---,---,<br>---,---,--9 | Returns an integer that represents the maximum spool space, in bytes, that was used at any one time by the database on a specified AMP (or on all AMPs if the SUM aggregate is specified).            |
| PeakPersistentSpool    | BIGINT    | --,---,---,---,<br>---,---,--9 | Returns an integer that represents the maximum persistent spool space, in bytes, that was used at any one time by the database on a specified AMP (or on all AMPs if the SUM aggregate is specified). |
| PeakTemp               | BIGINT    | --,---,---,---,<br>---,---,--9 | Returns the peak number of bytes used at one time by a temporary table per vproc.   |
| MaxProfileSpool        | BIGINT    | --,---,---,---,<br>---,---,--9 | Returns the profile SPOOL space limit per AMP for the user if the user is assigned a profile which has a SPOOL space setting. Otherwise, this column has a null value.                                |
| MaxProfileTemp         | BIGINT    | --,---,---,---,<br>---,---,--9 | Returns the profile TEMPORARY space limit per AMP for the user if the user is assigned a profile which has a.   |
| AllocatedPerm          | BIGINT    | --,---,---,---,<br>---,---,--9 | Returns the current allocation in bytes for permanent and user defined temporary table space on the AMP.  |

| View Column    | Data Type | Format                         | Comment  |
|----------------|-----------|--------------------------------|--|
| AllocatedSpool | BIGINT    | --,---,---,---,<br>---,---,--9 | Returns the current allocation in bytes for spool and other system-defined temporary table space on the AMP. |
| AllocatedTemp  | BIGINT    | --,---,---,---,<br>---,---,--9 | Returns the current allocation in bytes for temporary tables space on the AMP.                               |
| PermSkew       | SMALLINT  | ---,--9                        | Returns the permissible skew limit percent for permanent space usage at the Global level.                    |
| SpoolSkew      | SMALLINT  | ---,--9                        | Returns the permissible skew limit percent for spool space usage at the Global level.                        |
| TempSkew       | SMALLINT  | ---,--9                        | Returns the permissible skew limit percent for temporary space usage at the Global level.                    |

## Usage Notes

When a database or user is created, allocated disk space is divided evenly among all AMPs. The DiskSpace view returns one row of usage information for each AMP in the system (or for all AMPs if the SUM aggregate is used).

When a database is created, a space row is added on each AMP, with the processor field in each row initialized to 0. The first time the space row is updated, such as when a table is created in the database or the system is restarted, the processor field in each row is updated to reflect the actual processor number.

You can use the DiskSpaceV[X] view to build and maintain a table of disk space usage statistics for each username or accountname.

To create the history table, enter the following statement:

```
CREATE TABLE DiskSpaceHist (DataBaseName VARCHAR(128) CHARACTER SET UNICODE,
    AccountName VARCHAR(128) CHARACTER SET UNICODE,
    MaxPerm FLOAT,
    MaxSpool FLOAT,
    CurrentPerm FLOAT,
    PeakPerm FLOAT,
    PeakSpool FLOAT,
    CollectDate DATE,
    CollectTime FLOAT )
PRIMARY INDEX (DataBaseName, AccountName);
```

You can periodically collect usage statistics using the following procedure:

1. Select statistics from the DiskSpaceV[X] view and insert them in the history table.

2. Reset DiskSpace counters to zero for the next collection period.

**Note:**

You can reset the maximum and peak DiskSpace counters to zero using the ClearPeakDisk macro, which is provided on the release tape.

This procedure can be done using the following BTEQ script:

```
.LOGON username, password

INSERT INTO DiskSpaceHist
  SELECT DataBaseName, AccountName,
     SUM(MaxPerm),
     SUM(MaxSpool),
     SUM(CurrentPerm),
     SUM(PeakPerm),
     SUM(PeakSpool),
     DATE, TIME
  FROM DBC.DiskSpaceV
  GROUP BY DataBaseName, AccountName, DATE, TIME;

EXECUTE DBC.ClearPeakDisk;

.QUIT
```

**Corresponding Tables**

The X view references these additional tables:

- DBC.AccessRights
- DBC.Owners
- DBC.RoleGrants
- DBC.Roles

**AppProxyUser and TrustUserName**

These columns are not returned in the X or VX views.

**PeakPerm, PeakSpool, PeakPersistentSpool, and PeakTemp**

You can use the DBC.ClearPeakDisk macro to reset the PeakPerm, PeakSpool, PeakPersistentSpool, and PeakTemp.

**Examples: Using DiskSpaceV**

**Example: Using DiskSpaceV to Show Disk Space Currently in Use**

To display disk space currently in use:

```
SELECT SUM(CurrentPerm), SUM(MaxPerm),
((CAST(SUM(currentperm) AS FLOAT)/ NULLIFZERO(SUM(maxperm)) * 100))
(TITLE '%MaxPerm', FORMAT 'zz9.99')
FROM DBC.DiskSpaceV;
```

**Example: Using DiskSpaceV to Show the Percentage of Disk Space Available for Spool**

To display the percentage of disk space that is available for spool:

```
SELECT ((CAST((SUM(MaxSpool) - SUM(CurrentSpool)) AS FLOAT) /
NULLIFZERO(SUM(MaxSpool))) * 100)
(TITLE '% Avail for Spool', format 'zz9.99')
FROM DBC.DiskSpaceV;
```

**Example: Using DiskSpaceV to Show the Percentage of Space Used by Each Database**

To display percentage of space used by each database in the system:

```
SELECT Databasename (format 'X(12)')
,SUM(maxperm)
,SUM(currentperm)
,(CAST(SUM(currentperm) AS FLOAT)/
NULLIFZERO (SUM(maxperm)) * 100)
(FORMAT 'zz9.99%', TITLE 'Percent // Used')
FROM DBC.DiskSpaceV
GROUP BY 1
ORDER BY 4 DESC
WITH SUM (currentperm), SUM(maxperm);
```

**Example: Using DiskSpaceV to Show Users Who are Running Out of Permanent Space**

To display the users who are running out of permanent space:

```
SELECT Databasename (format 'X(12)')
,SUM(maxperm)
,SUM(currentperm)
,(CAST(SUM(currentperm) AS FLOAT)/
NULLIFZERO (SUM(maxperm)) * 100)
(format 'zz9.99%', TITLE 'Percent // Used')
FROM DBC.DiskSpaceV
```

```
GROUP BY 1 HAVING (CAST(SUM(currentPerm) AS FLOAT)/
NULLIFZERO(SUM(maxperm))) > 0.5
ORDER BY 4 DESC;
```

**Note:**

You can change the value 0.9 to whatever threshold ratio is appropriate for your site.

**Example: Using DiskSpaceV to Show AMPs That are Running Out of Physical Space**

To display AMPs that are close to running out physical space:

```
SELECT SUM(maxperm) m, SUM(currentperm+currentspool+currentttemp) c ,
((m-c)*100.00)/m (decimal(5,2)) p, vproc
FROM DBC.diskpacev GROUP BY vproc HAVING p < 5.00 ;
```

**Note:**

The above query returns all AMPs whose actual usage is leaving less than 5% of their total space. Change 5.00 to another suitable value to as required.

**Example: Using DiskSpaceV to Show Users Who are Using a Lot of Spool**

To display the users who are using a lot of spool:

```
SELECT databasename
,SUM(peaks pool)
FROM DBC.DiskSpaceV
GROUP BY 1 HAVING SUM(peaks pool) > 5000000000
ORDER BY 2 DESC;
```

**Note:**

You can change the value 5000000000 to a value that is appropriate for your site. Some sites with more space may have a higher tolerance for higher spool usage and spool limits.

**Example: Using DiskSpaceV to Show Databases with Permanent Space Skewed**

To show users or databases that have permanent space skewed, but are defined with a 0 skewlimit:

```
SELECT
COALESCE((MAX(currentPerm)/NULLIFZERO(AVG(currentPerm)) - 1), 0) * 100 (FORMAT
'999.99') AS spaceskew, databasename
FROM DBC.diskpacev GROUP BY 2
```

```
HAVING spaceskew > 125.0
WHERE permskew = 0;
```

**Note:**

The example query checks databases that have the space usage skewed beyond the normal 25% usage. Change 25.0 to any other value that best suits your query needs. For resulting databases, consider defining a permskewlimit with reduction in the maximum limit to better manage space to avoid over-defining the maximum space limit. SkewLimit > 0 implies need-based space allocations happen to AMPs instead of allocating the sufficient per-AMP quota to each of the AMPs without getting out-of-space errors.

**Example: Using DiskSpaceV to Show Current Permanent Usage and AMP Space Allocations**

To show current permanent usage and AMP level space allocations for a given database:

```
SELECT vproc, currentperm, maxperm, AllocatedPerm
FROM DBC.DiskSpaceV
WHERE databasename = 'xxxx';
```

**Note:**

Replace 'xxxx' with a database name, such as 'SYSBAR'.

**Example: Using DiskSpaceV to Show the Permanent Space Usage for Databases in a Map**

To show the permanent space usage for databases and users associated with a given default map:

```
SELECT dbv.databasename, sum(currentperm), sum(maxperm)
FROM DBC.DiskSpaceV dbspace, DBC.DatabasesV dbv WHERE dbspace.databasename
= dbv.databasename
AND mapname = 'MyMap1' GROUP BY 1;
```

## ErrorTblsV[X]

**Category:** Schema

**Database:** DBC

| View Column  | Data Type  | Format | Comment   |
|--------------|--|--------|---|
| ErrTblDbName | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128) | Returns the name of the database where the error table resides. |

| View Column     | Data Type  | Format              | Comment   |
|-----------------|--|---------------------|---|
| ErrTblName      | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128)              | Returns the name of the error table.                        |
| BaseTblDbName   | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC             | X(128)              | Name of database where base table resides.                  |
| BaseTblName     | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC             | X(128)              | Returns the name of the table that contains an error table. |
| CreatorName     | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC             | X(128)              | Error table creator.  |
| CreateTimeStamp | TIMESTAMP(0) NOT NULL                                | YYYY-MM-DDBHH:MI:SS | Time stamp of error table creation.                         |

## Usage Notes

### Corresponding Tables

The X view references these additional tables:

- DBC.AccessRights
- DBC.Owner
- DBC.RoleGrants
- DBC.Roles

## Example: Using ErrorTblsV

The following example shows base data table u2.t1 has an error table u2.ET\_t1, which is created by u2:

```
SELECT ErrTblDbName (char(8)),
       ErrTblName (char(8)),
       BaseTblDbName (char(8)),
       BaseTblName (char(8)),
       CreatorName (char(4)),
       CreateTimeStamp
FROM   DBC.ErrorTblsV;
```

Result:

```
ErrTblDbName  ErrTblName  BaseTblDbName  BaseTblName
CreatorName    CreateTimeStamp
-----
-----
```

```
u2          ET_t1          u2          t1          u2
2006-08-28 11:47:56
```

## Events\_ConfigurationV[X]

**Category:** Operations

**Database:** DBC

| View Column    | Data Type   | Format         | Comment  |
|----------------|---|----------------|--|
| CreateDate     | DATE NOT NULL   | YY/MM/DD       | Returns the date when the event took place.  |
| CreateTime     | FLOAT NOT NULL  | 99:99:99.99    | Returns the time when the event took place.  |
| EventNum       | INTEGER NOT NULL  | --,---,---,--9 | Returns the client system event number of the restore operation.   |
| EventType      | CHAR(30) LATIN<br>NOT CASESPECIFIC<br>NOT NULL          | X(30)          | Returns the type of event that occurred.   |
| UserName       | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)         | Returns the username associated with the event.  |
| LogProcessor   | SMALLINT  | -(5)9          | Returns the logical processor ID for an AMP not affected by the event.   |
| PhyProcessor   | SMALLINT  | -(5)9          | Returns the physical processor ID for an AMP not affected by the event.  |
| Vproc          | SMALLINT  | -(5)9          | Identifies the virtual processor for which an event was logged.  |
| ProcessorState | CHAR(1) LATIN<br>NOT CASESPECIFIC<br>NOT NULL           | X(1)           | Returns D (the event was for all AMPs and the processor was down) or U (the event was for specific AMPs).  |
| RestartSeqNum  | SMALLINT  | ---,--9        | Returns an integer (0 through n) to indicate the number of times that the Teradata Database had to be restarted during the event. 0 indicates that no restarts took place. |

## Usage Notes

The Events\_ConfigurationV[X] view contains rows for each archive activity that does not affect all AMPs in the system configuration.

If the activity is for all AMPs and there are AMPs off-line, a row is inserted for each off-line AMP. If the activity is for specific AMPs, a row is inserted for each AMP that is specified and online.

## Corresponding Tables

The X view references these additional tables:

- DBC.AccessRights
- DBC.DBase
- DBC.Owners
- DBC.RoleGrants
- DBC.Roles

## CreateDate and CreateTime

The CreateDate and CreateTime columns are updated by the PE on which the session is running. That is, all events for a given session are timestamp-ordered.

However, if multiple or concurrent sessions are running on different PEs, any discrepancy in AMP clocks may be reflected in the timestamp sequence. This may also occur if the database is connected to more than one client system and the client system clocks are not synchronized.

## Possible Values for EventType

- CHECKPOINT
- RESTORE
- DELETE
- ROLLBACK
- DUMP
- ROLLFORWARD

## Example: Using Events\_ConfigurationV

The statement on the following screen selects information concerning the requesting user from the DBC.Events\_ConfigurationX view:

```
SELECT CreateDate, CreateTime, EventNum, EventType
      FROM DBC.Events_ConfigurationV;
```

Result:

| CreateDate | CreateTime | EventNum | EventType   |
|------------|------------|----------|-------------|
| -----      | -----      | -----    | -----       |
| 87/03/18   | 08:53:48   | 30       | Rollforward |
| 87/03/18   | 08:57:49   | 44       | Rollforward |
| 87/03/18   | 08:54:42   | 33       | Rollforward |
| 87/03/20   | 11:26:26   | 98       | Dump        |
| 87/03/18   | 09:00:05   | 52       | Rollforward |

|          |          |    |             |
|----------|----------|----|-------------|
| 87/03/18 | 09:30:59 | 55 | Restore     |
| 87/03/18 | 08:57:02 | 41 | Rollforward |

## Events\_MediaV[X]

**Category:** Operations

**Database:** DBC

| View Column    | Data Type   | Format         | Comment  |
|----------------|---|----------------|--|
| CreateDate     | DATE NOT NULL   | YY/MM/DD       | Returns the date when the event took place.  |
| CreateTime     | FLOAT NOT NULL  | 99:99:99.99    | Returns the time when the event took place.  |
| EventNum       | INTEGER NOT NULL  | --,---,---,--9 | Returns the client system event number of the restore operation.   |
| EventType      | CHAR(30) LATIN<br>NOT CASESPECIFIC<br>NOT NULL          | X(30)          | Returns the type of event that occurred.   |
| UserName       | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)         | Returns the username associated with the client system utility dump or restore event.                      |
| DataSetName    | VARCHAR(44) UNICODE<br>NOT CASESPECIFIC                 | X(44)          | Returns the client system data set name for a dump or restore event.                                       |
| VolSerialId    | CHAR(6) LATIN<br>NOT CASESPECIFIC<br>NOT NULL           | X(6)           | Returns the unique six character volume serial assigned to a device.                                       |
| VolSequenceNum | SMALLINT  | ---,--9        | Returns the sequence number of the volume, which defines the position of the volume in a multi-volume set. |
| DupeDumpSet    | CHAR(1) LATIN<br>NOT CASESPECIFIC<br>NOT NULL           | X(1)           | Returns a code to indicate whether the dump event created a duplicate archive dataset.                     |

## Usage Notes

### Corresponding Tables

The X view references these additional tables:

- DBC.AccessRights
- DBC.DBase
- DBC.Owners

- DBC.RoleGrants
- DBC.Roles

### CreateDate and CreateTime

The CreateDate and CreateTime columns are updated by the PE on which the session is running. That is, all events for a given session are timestamp-ordered.

However, if multiple or concurrent sessions are running on different PEs, any discrepancy in AMP clocks may be reflected in the timestamp sequence. This may also occur if the database is connected to more than one client system and the client system clocks are not synchronized.

### Possible Values for EventType

- CHECKPOINT
- RESTORE
- DELETE
- ROLLBACK
- DUMP
- ROLLFORWARD

### Example: Using Events\_MediaV

In this example, the requesting user is researching the Events\_Media view for events associated with the user named 'PAL'.

```
SELECT DataSetName,VolSerialId,DupeDumpSet
      FROM DBC.Events_MediaV WHERE UserName = 'PAL' ;
```

Result:

| DataSetName   | VolSerialId | DupeDumpSet |
|---------------|-------------|-------------|
| -----         | -----       | -----       |
| BRM.DBC.TEXT1 | 000469      | N           |
| BRM.DBC.TEXT1 | 000469      | N           |
| BRM.DBC.TEXT2 | 000469      | N           |
| BRM.DBC.TEXT2 | 000469      | N           |
| BRM.DBC.TEXT1 | BRM001      | Y           |
| BRM.DBC.TEXT1 | BRM002      | Y           |
| BRM.DBC.TEXT2 | BRM001      | N           |
| BRM.DBC.TEXT2 | BRM002      | N           |

## EventsV[X]

**Category:** Operations

**Database:** DBC

| View Column         | Data Type   | Format          | Comment   |
|---------------------|---|-----------------|---|
| CreateDate          | DATE NOT NULL   | YY/MM/DD        | Returns the date when the event took place.   |
| CreateTime          | FLOAT NOT NULL  | 99:99:99.<br>99 | Returns the time when the event took place.   |
| EventNum            | INTEGER NOT NULL  | --,---,---,--9  | Returns the client system event number of the restore operation.  |
| EventType           | CHAR(30) LATIN<br>NOT CASESPECIFIC<br>NOT NULL          | X(30)           | Returns the type of event that occurred.  |
| UserName            | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)          | Returns the username associated with the event.   |
| DatabaseName        | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)          | Returns the name of the database in which the event occurred.   |
| ObjectType          | CHAR(1) LATIN<br>NOT CASESPECIFIC                       | X(1)            | This field gives the level of object that was specified for the event. This is a D is for database level, T for table level, Q for Backup or restore of selected partitions or J for journal. |
| AllAmpsFlag         | CHAR(1) LATIN<br>NOT CASESPECIFIC<br>NOT NULL           | X(1)            | Returns a code that indicates whether the event was executed on all AMPs, clusters, or processors.  |
| RestartSeqNum       | SMALLINT  | ---,--9         | Returns an integer (0 through n) to indicate the number of times that the Teradata Database had to be restarted during the event. 0 indicates that no restarts took place.                    |
| OperationInProgress | CHAR(1) LATIN<br>NOT CASESPECIFIC<br>NOT NULL           | X(1)            | Returns a code to indicate whether the event is still processing. The following codes are used: Y = Yes, N = No.  |
| TableName           | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC             | X(128)          | Returns the name of the table affected by the event. If the object is a database, then the value is set to NULL.  |
| CheckPointName      | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC             | X(128)          | Returns the checkpoint name as specified by the user in the CHECKPOINT statement.   |

| View Column     | Data Type                                  | Format         | Comment   |
|-----------------|--|----------------|---|
| LinkingEventNum | INTEGER                                    | --,---,---,--9 | Returns the terminating event number specified by the user for a rollforward or rollback event.                                   |
| DataSetName     | VARCHAR(44)<br>UNICODE<br>NOT CASESPECIFIC | X(44)          | Returns the client system data set name for a dump or restore event.  |
| LockMode        | CHAR(1) LATIN<br>NOT CASESPECIFIC          | X(1)           | Returns a code to indicate the type of lock used by the event. The codes are as follows: A = ACCESS or Group READ, R = full READ. |
| JournalUsed     | CHAR(1) LATIN<br>NOT CASESPECIFIC          | X(1)           | Returns one of the following codes to indicate which part of the journal table was used: C = Current, R = Restored, S = Saved.    |
| JournalSaved    | CHAR(1) LATIN<br>NOT CASESPECIFIC          | X(1)           | Returns one of the following codes to indicate whether or not the SAVE option was used in the event: Y = Yes, N = No.             |
| IndexPresent    | CHAR(1) LATIN<br>NOT CASESPECIFIC          | X(1)           | Returns Y (yes) or N (no) to indicate whether or not the INDEX option was used in a dump event.                                   |
| DupeDumpSet     | CHAR(1) LATIN<br>NOT CASESPECIFIC          | X(1)           | Returns a code to indicate whether the dump event created a duplicate archive dataset.  |

## Usage Notes

### Corresponding Table

The X view references these additional tables:

- DBC.AccessRights
- DBC.DBBase
- DBC.Owners
- DBC.RoleGrants
- DBC.Roles

### Types of Events and Fields

The EventsV[X] view returns a row for each archive or recovery operation. The types of event rows are:

| Type of event row ... | A row is created for each ... |
|-----------------------|-------------------------------|
| Checkpoint            | journal checkpointed.         |
| Delete                | journal deleted.              |

| Type of event row ... | A row is created for each ...     |
|-----------------------|-----------------------------------|
| Dump                  | database or table dumped.         |
| Restore               | database or table restored.       |
| Rollback              | database or table rolled back.    |
| Rollforward           | database or table rolled forward. |

The EventsV[X] view contains the following standard and optional fields. Optional fields can return NULL. Standard fields are not nullable and always return a value.

| Standard Fields     | Optional Fields |
|---------------------|-----------------|
| EventNum            | DataSetName     |
| CreateDate          | TableName       |
| CreateTime          | CheckpointName  |
| UserName            | LinkingEventNum |
| EventType           | LockMode        |
| DatabaseName        | JournalUsed     |
| ObjectType          | JournalSaved    |
| AllAMPsFlag         | IndexPresent    |
| RestartSeqNum       | DupeDumpSet     |
| OperationInProgress |                 |

### CreateDate and CreateTime

The CreateDate and CreateTime columns are updated by the PE on which the session is running. That is, all events for a given session are timestamp-ordered.

However, if multiple or concurrent sessions are running on different PEs, any discrepancy in AMP clocks may be reflected in the timestamp sequence. This may also occur if the database is connected to more than one client system and the client system clocks are not synchronized.

### Possible Values for AllAMPsFlag

| Value | Description    |
|-------|----------------|
| A     | ALL AMPS USED  |
| C     | ALL CLUSTERS   |
| P     | ALL PROCESSORS |

**Possible Values for EventType**

- CHECKPOINT
- RESTORE
- DELETE
- ROLLBACK
- DUMP
- ROLLFORWARD

**Possible Values for ObjectType**

| Value | Description                              |
|-------|--|
| D     | Database                                 |
| J     | Journal Table                            |
| Q     | Backup or restore of selected partitions |
| T     | Table                                    |

**Possible Values for OperationInProgress**

| Value | Description                             |
|-------|---|
| Y     | Indicates an event is still processing. |
| N     | Event is not processing.                |

**TableName**

If the object is a database, the TableName column is set to NULL.

**Example: Using EventsV**

The following SELECT statement selects information associated with the requesting user from the DBC.EventsV view:

```
SELECT CreateDate, CreateTime, EventType, JournalUsed FROM DBC.EventsV;
```

Result:

| CreateDate | CreateTime | EventType   | JournalUsed |
|------------|------------|-------------|-------------|
| -----      | -----      | -----       | -----       |
| 87/03/18   | 11:10:45   | Rollforward | R           |
| 87/03/18   | 11:18:20   | Restore     | R           |

|          |          |             |   |
|----------|----------|-------------|---|
| 87/03/19 | 12:06:34 | Rollforward | R |
| 87/02/12 | 14:13:38 | Dump        | S |

## ExclusionListsV[X]

**Category:** TDMaps

**Database:** TDMaps

| View Column        | Data Type  | Format                            | Comment   |
|--------------------|--|-----------------------------------|---|
| ZoneName           | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC             | X(128)                            | Zone name   |
| ListName           | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128)                            | Exclusion list name                               |
| DictionaryDatabase | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128)                            | Dictionary Database                               |
| CreatedTimeStamp   | TIMESTAMP(6) WITH<br>TIME ZONE                       | YYYY-MM-<br>DDBHH:MI:SS.<br>S(6)Z | Timestamp from when<br>the list was created       |
| LastModified       | TIMESTAMP(6) WITH<br>TIME ZONE                       | YYYY-MM-<br>DDBHH:MI:SS.<br>S(6)Z | Timestamp from when<br>the list was last modified |
| CreateListUser     | BYTE(4)  | X(8)                              | ID of the user that created<br>the list           |
| DatabaseName       | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128)                            | Database name                                     |
| ObjectName         | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC             | X(128)                            | Object name                                       |

## Usage Notes

The ExclusionListsV[X] views are security zone constrained views that contain object lists and the objects (database, table, join index, or hash index) included in each list. The exclusion object lists specify objects that are not to be moved. ExclusionListsVX returns only the objects to which the user has access.

### DictionaryDatabase

DictionaryDatabase is an optional column to define where the data dictionary tables are stored. This should be set when running AnalyzeSP on a system that is different from the system where the Exclusions are defined. The contents of Maps, Dbase, TVM, TVFields, DBQLStepTbl, ObjectUsage, and DatabaseSpace

for the objects to analyze need to be copied from the source system to DictionaryDatabase on the system where AnalyzeSP and CreateExclusionListSP will be run. Note that the following views need to be created in DictionaryDatabase: Tables2V, Databases2V, MapsV, QryLogStepsV, ColumnsV, InsertUseCountV, UpdateUseCountV, DeleteUseCountV, and DiskSpaceV. Finally, the rows written to ActionsTbl need to be copied to the source system.

## ExportWidthV

**Category:** Schema

**Database:** DBC

| View Column          | Data Type                               | Format | Comment                            |
|----------------------|---|--------|------------------------------------|
| ExportDefinitionName | VARCHAR(30) LATIN<br>UPPERCASE NOT NULL | X(30)  | The name of the ExportWidthRuleSet |
| ExportWidthRuleSet   | BYTE(20) NOT NULL                       | X(40)  | The export width Rule Set          |

## How to Interpret ExportWidthRuleSet Column Values

The values stored in the ExportWidthRuleSet column represent the export width rule set for the current export width definitions (pre-defined and user-defined export width definitions). Each export width definition has its own rule set.

An export width rule set is the set of export width conversion multiplier values for all of the fields of the server character sets for a single export width definition. A total of 40 values make up the rule set for one export width definition.

### Example Export Width Rule Sets

This is an example of the rule sets for the pre-defined export width definitions.

| ExportDefinitionName | ExportWidthRuleSet                       |
|----------------------|--|
| -----                | -----                                    |
| EXPECTED             | 1112211111222232222211121111112222322222 |
| MAXIMUM              | 1322323221332232322321123111122322323222 |
| COMPATIBILITY        | 111221111111123111111121111112222322222  |

### What Each Digit Indicates

A single digit in an export width rule set indicates two things:

- The export width for the server character set.
- The export width for the session character set.

You can easily identify the export width rules for a server character set. The same 10 digits are always used to indicate the export width rules for the same server character set.

- First set of 10 digits: LATIN
- Second set of 10 digits: UNICODE
- Third set of 10 digits: KANJISJIS
- Fourth set of 10 digits: GRAPHIC

Each of the 10 digits for a server character set are used to indicate the export width rule for the session character set. The same digit (based on the position of the digit) is always used to indicate the export width rule for the same session character set.

This table lists the export width rule indicated by each digit in the set of 10 digits used for each server character set.

**Note:**

With the exception of the UTF16 session character set export width, the possible values for every digit in an export width rule set are 1, 2, 3, or 4. The possible values for the UTF16 session character set digit are 2 or 4.

| Digit (by position) | Export Width For...  |
|---------------------|--|
| 1                   | <ul style="list-style-type: none"> <li>• Any session character set that ends in the string _OI</li> <li>• The KATAKANAEBDIC session character set</li> </ul> |
| 2                   | Any session character set that ends in the string _OU  |
| 3                   | Any session character set that ends in the string _OS  |
| 4                   | The session character set UTF16<br><b>Note:</b><br>The possible values for this digit are 2 or 4   |
| 5                   | The session character set UTF8   |
| 6                   | Any site-defined session character set with STATEMACHINE EUC1211   |
| 7                   | Any site-defined session character set with STATEMACHINE EUC1223   |
| 8                   | Any site-defined session character set with STATEMACHINE S80   |
| 9                   | Any site-defined session character set with STATEMACHINE S80A1E0   |
| 10                  | Any site-defined session character set with STATEMACHINE SOSI0E0F  |

If you know how the ExportWidthRuleSet column values are grouped and the meaning of each of the 10 digits described in this table, you can determine the export width for the current server character set and session character set.

This list gives the export width for an export width definition that has this rule set: 1112211111 2222322222 1112111111 2222322222.

- When exporting from the LATIN server character set and the session character set is UTF8, the export width is 2. The digit that represents the export width for this combination of character sets is in the 5th position in the first set of 10 digits in the rule set.
  - 5th position digit indicates the UTF8 session character set.
  - First set of 10 digits indicates the LATIN server character set.
- When exporting from the KANJISJIS server character set and the session character set is UTF16, the export width is 2. The digit that represents the export width for this combination of character sets is in the 4th position in the third set of 10 digits in the rule set.
  - 4th position digit indicates the UTF16 session character set.
  - Third set of 10 digits indicates the KANJISJIS server character set.

## Related Topics

| For more information about ...   | See ...   |
|--|---|
| export width and using the DBSControl utility to make export width definition changes              | <i>Teradata Vantage™ - Advanced SQL Engine International Character Set Support.</i> |
| using the ReplaceExportDefinition stored procedure to create user-defined export width definitions | <i>Teradata Vantage™ SQL Operators and User-Defined Functions, B035-1210.</i>       |

## ExpStatsV

**Category:** Optimizer Statistics

**Database:** DBC

| View Column  | Data Type   | Format   | Comment   |
|--------------|---|----------|---|
| DatabaseName | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)   | The DatabaseName column is the name of the database in which the table resides. |
| TableName    | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)   | The TableName column is the name of the containing table.                       |
| ColumnName   | VARCHAR(10000)<br>UNICODE NOT<br>CASESPECIFIC           | X(10000) | The ColumnName column identifies a column or columns.                           |
| StatsName    | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC             | X(128)   | The StatsName column contains the alias name associated with the statistic      |

| View Column           | Data Type                              | Format                   | Comment   |
|-----------------------|--|--------------------------|---|
| StatsSource           | CHAR(1)<br>LATIN UPPERCASE             | X(1)                     | The StatsSource column records the method this statistic is acquired.                                 |
| ValidStats            | CHAR(1)<br>LATIN UPPERCASE             | X(1)                     | The ValidStats column indicates whether the statistics are valid or not.                              |
| DBSVersion            | VARCHAR(32)<br>LATIN UPPERCASE         | X(32)                    | Returns the version of the database that contains the objects on which the statistics were collected. |
| IndexNumber           | SMALLINT                               | ---,--9                  | The IndexNumber column is the Index number of the index on which statistics are collected.            |
| SampleSignature       | VARCHAR(256)<br>LATIN NOT CASESPECIFIC | X(256)                   | The SampleSignature column is the Sample options encoded as a 10 character signature.                 |
| SampleSizePct         | DECIMAL(5,2)                           | ----.99                  | The SampleSizePct column is the Sample size percent used when collecting statistics.                  |
| ThresholdSignature    | VARCHAR(512)<br>LATIN NOT CASESPECIFIC | X(512)                   | The ThresholdSignature column is the Threshold options encoded as a 17 character signature.           |
| MaxIntervals          | SMALLINT                               | ---,--9                  | The MaxIntervals column is the User-specified maximum number of intervals.                            |
| MaxValueLength        | INTEGER                                | --,---,---,--9           | The MaxValueLength column is the User-specified maximum value length.                                 |
| RowCount              | FLOAT                                  | ----,---,---,---,<br>--9 | The RowCount column is the row count of the table, view or query on which statistics are collected.   |
| UniqueValueCount      | FLOAT                                  | ----,---,---,---,<br>--9 | The UniqueValueCount column is the Number of unique values of the ExpressionList.                     |
| PNullUniqueValueCount | FLOAT                                  | ----,---,---,---,<br>--9 | The PNullUniqueValueCount column is the Number of unique values from rows with partial nulls.         |
| NullCount             | FLOAT                                  | ----,---,---,---,<br>--9 | The NullCount column is the Number of nulls of the ExpressionList.                                    |
| AllNullCount          | FLOAT                                  | ----,---,---,---,<br>--9 | The AllNullCount column is the Number of all nulls of the ExpressionList.                             |
| HighModeFreq          | FLOAT                                  | ----,---,---,---,<br>--9 | The HighModeFreq column is the highest frequency of values of the ExpressionList.                     |

| View Column          | Data Type    | Format                   | Comment   |
|----------------------|--------------|--------------------------|---|
| PNullHighModeFreq    | FLOAT        | ----,---,---,---,<br>--9 | The PNullHighModeFreq column is the highest frequency of values having partial nulls of the ExpressionList. |
| CreateTimeStamp      | TIMESTAMP(0) | YYYY-MM-DDBHH:MI:SS      | The CreateTimeStamp column is the statistics creation time stamp.   |
| LastCollectTimeStamp | TIMESTAMP(0) | YYYY-MM-DDBHH:MI:SS      | The LastCollectTimeStamp column is the Last statistics collection time stamp.                               |
| LastAlterTimeStamp   | TIMESTAMP(0) | YYYY-MM-DDBHH:MI:SS      | The LastAlterTimeStamp column is the last user updated time stamp.  |

## Usage Notes

### Referenced Columns

Many of the Data Dictionary view columns have referenced table columns. That is, the value in the view column corresponds to a value in the selected column referenced in the table. It would be meaningful to join the view and the referenced table based on the selected column and the referenced column.

Referenced columns for this view are:

| View Column | Referenced Column       |
|-------------|-------------------------|
| IndexNumber | DBC.Indexes.IndexNumber |

### IndexNumber

The IndexNumber field contains the index number when statistics are collected on an index. Otherwise, it is NULL.

### MaxInterval and MaxValueLength

If these statistics are collected with system determined maximum intervals and maximum value length, the MaxInterval and MaxValueLength columns are NULL.

### SampleSizePct

If sampling is not used, the SampleSizePct column is set to 0 or 100.

### StatsSource

The StatsSource column records the method by which this statistic is acquired. For information about the possible values for the StatsSource column, see "StatsSource Column."

## To Get Information Not Contained in This View

This view does not contain:

- Statistics on tables protected by row-level security. To get these statistics, create views on DBC.StatsTbl. You can view details about this table in Teradata SQL Assistant or Teradata Studio Express.
- Information about column attributes. To get this information, join the DBC.StatsTbl table to the DBC.TVFields table. You can view details about this table in Teradata SQL Assistant or Teradata Studio Express.

## Example: Using ExpStatsV

This example assumes the following statistics are collected:

```
STATISTICS
COLUMN CAST(o_orderdatetime AS DATE) AS Stats_OrderDate
ON Orders;
```

This query can be used to retrieve the statistics collected on single expressions:

```
SELECT * FROM dbc.ExpStatsV
WHERE databasename = 'sales'
AND tablename = 'orders';
```

## Related Topics

| For more information about statistics collected on ...  | See ...   |
|---|---|
| non-indexed columns and single-column indexes           | <a href="#">ColumnsV[X]</a> .                           |
| indexes for which two or more columns have been defined | <a href="#">IndexStatsV[X]</a> .                        |
| tables  | <a href="#">StatsV</a> or <a href="#">TableStatsV</a> . |
| materialized temporary tables                           | <a href="#">TempTableStatsV</a> .                       |
| multiple columns  | <a href="#">MultiColumnStatsV</a> .                     |
| multiple expressions                                    | <a href="#">MultiExpStatsV</a> .                        |

## ForeignTablesInfoV[X]

**Category:** Operations

**Database:** DBC

| View Column  | Data Type                                      | Format   | Comment   |
|--------------|--|----------|---|
| DatabaseName | VARCHAR(128) UNICODE NOT CASESPECIFIC NOT NULL | X(128)   | Returns the name of the database.                         |
| TableName    | VARCHAR(128) UNICODE UPPERCASE NOT NULL        | X(128)   | Returns the name of the FOREIGN table.                    |
| OptionInfo   | VARCHAR(128) UNICODE NOT CASESPECIFIC NOT NULL | X(128)   | Returns the name portion specified in the custom-clause.  |
| ValueInfo    | VARCHAR(31000) UNICODE NOT CASESPECIFIC        | X(31000) | Returns the value portion specified in the custom-clause. |

## ForeignTablesV[X]

**Category:** Operations

**Database:** DBC

| View Column  | Data Type                                      | Format | Comment                                       |
|--------------|--|--------|---|
| DataBaseName | VARCHAR(128) UNICODE NOT CASESPECIFIC NOT NULL | X(128) | Lists the database name of the foreign table. |
| TableName    | VARCHAR(128) UNICODE NOT CASESPECIFIC NOT NULL | X(128) | Returns the name of the FOREIGN table.        |

## ExternalSPsV[X]

**Category:** Operations

**Database:** DBC

| View Column           | Data Type                                      | Format | Comment  |
|-----------------------|--|--------|--|
| DatabaseName          | VARCHAR(128) UNICODE NOT CASESPECIFIC NOT NULL | X(128) | Returns the name of the database in which the External Stored Procedure resides. |
| ExternalProcedureName | VARCHAR(128) UNICODE UPPERCASE NOT NULL        | X(128) | Identifies the name of an External Stored Procedure.                             |
| ExternalProcedureId   | BYTE(6) NOT NULL                               | X(12)  | Specifies the unique ID of an External Stored Procedure.                         |
| NumParameters         | SMALLINT NOT NULL                              | ---,-9 | Returns the number of parameters of a function.                                  |

| View Column        | Data Type                                | Format  | Comment  |
|--------------------|--|---------|--|
| ExternalName       | CHAR(30) LATIN<br>NOT NULL               | X(30)   | Returns the external name of a function. For SQL UDFs, the value is always SQLUDF.   |
| SrcFileLanguage    | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL   | X(1)    | Returns the source file language used to implement a user-defined function (UDF).  |
| NoSQLDataAccess    | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL   | X(1)    | Returns the access indicated in the 'Create Procedure' statement. Y = No SQL in the external stored procedures; C = Contains SQL; R = Reads SQL data; M = Modifies SQL data; |
| ParameterStyle     | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL   | X(1)    | Returns the parameter passing convention for the function or an External Stored Procedure. The following codes are used: S = SQL, G = TD_GENERAL, I = for internal.          |
| ExecProtectionMode | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL   | X(1)    | Returns whether the function can be executed directly with a call statement (not protected) or indirectly as a separate process (protected).                                 |
| ExtFileReference   | VARCHAR(1000)<br>UNICODE<br>CASESPECIFIC | X(1000) | Saves the external file reference provided in the CREATE/REPLACE FUNCTION or external stored procedure statement.  |
| CharacterType      | SMALLINT NOT NULL                        | ---,-9  | Returns the character type that was the default when the function or External Stored Procedure was created.  |
| Platform           | CHAR(8) LATIN<br>UPPERCASE<br>NOT NULL   | X(8)    | Returns the platform of the system on which the object is compiled.  |
| RoutineKind        | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL   | X(1)    | Returns a single character indicating the family of the routine.   |
| ParameterUDTIds    | VARBYTE(512)                             | X(1024) | Returns a series of 4-byte UDT-type identifiers.   |
| AuthIdUsed         | BYTE(6)                                  | X(12)   | Returns the identifier of the authorization being used for the External Stored Procedure or User Defined Function.   |
| AppCategory        | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL   | X(1)    | Returns the application category of an external stored procedure.  |

| View Column         | Data Type                            | Format | Comment   |
|---------------------|--------------------------------------|--------|---|
| GLOPSetDatabaseName | VARCHAR(128)<br>UNICODE<br>UPPERCASE | X(128) | Returns the name of the database to which the GLOP set belongs for the member clause.   |
| GLOPSetMemberName   | VARCHAR(128)<br>UNICODE<br>UPPERCASE | X(128) | Returns the name of the GLOP set to which the XSP/UDF/UDM belongs. It is zero length when the external routine or UDF/UDM is not a member of the set. |

## Usage Notes

For information about the possible values for the NoSQLDataAccess and RoutineKind columns, see "NoSQLDataAccess Column" and "RoutineKind Column."

## Corresponding Tables

The X view references these additional tables:

- DBC.AccessRights
- DBC.Owners
- DBC.RoleGrants
- DBC.Roles

## Platform

The Platform column value is LINUX64.

## Possible Values for AppCategory

| Value | Description                                |
|-------|--|
| C     | CLI  |
| O     | ODBC                                       |
| N     | .NET                                       |
| J     | JAVA                                       |
| S     | C and CPP external stored procedure or UDF |

## Possible Values for ExecProtectionMode

| Value | Description      |
|-------|------------------|
| P     | Protected mode   |
| U     | Unprotected mode |

**Possible Values for ParameterStyle**

| Value | Description       |
|-------|-------------------|
| S     | SQLStyle          |
| G     | GeneralStyle      |
| I     | InternalStyle     |
| J     | JavaStyle         |
| O     | SQLTable          |
| N     | InternalSQL_Style |

**Possible Values for SrcFileLanguage**

| Value | Description |
|-------|-------------|
| C     | C           |
| P     | C++         |
| J     | JAVA        |
| S     | SQL         |
| A     | SAS         |

**Example: Using ExternalSPsV**

The following SELECT statement returns information about the external stored procedures in database 'dba':

```
SELECT * FROM DBC.ExternalSPsV
WHERE databasename = 'dba';
```

Result:

```
DataDatabaseName      dba
ExternalProcedureName  XSP_100
ExternalProcedureId    00006E0F0000
NumParameters          2
ExternalName           xsp_100
SrcFileLanguage        C
NoSQLDataAccess        Y
ParameterStyle         S
```

```

ExecProtectionMode      P
ExtFileReference        CS!xsp_100!/home/i18n/ck1/udftest/xsp_100.c
CharacterType           1
Platform                LINUX64
RoutineKind             R
ParameterUDTIds
AuthIdUsed              00001E0A0000
AppCategory             S
GLOPSetDatabaseName     ?
GLOPSetMemberName       ?

```

## FunctionAliasInfoV[X]

**Category:** Operations

**Database:** DBC

| View Column       | Data Type  | Format   | Comment  |
|-------------------|--|----------|--|
| FunctionAliasName | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128)   | Returns the name of the Function Alias.  |
| NameInfo          | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128)   | Returns the name portion specified in the custom-clause.   |
| ValueInfo         | VARCHAR(31000) UNICODE<br>NOT CASESPECIFIC           | X(31000) | Returns the value portion specified in the custom-clause.  |
| ValueType         | CHAR(1) LATIN<br>NOT CASESPECIFIC                    | X(1)     | Returns the identifier to identify whether the value specified in the custom-clause is either SELECT/Value/System. |
| NVPTYPE           | VARCHAR(7) UNICODE<br>NOT CASESPECIFIC               | X(7)     | Returns whether the name value pair is either IN/OUT/GENERIC.  |

## Usage Notes

### Possible Values for NVPTYPE

NVPTYPE returns the name value pair type or UNKNOWN.

- OUT
- IN
- GENERIC
- UNKNOWN

## FunctionAliasV[X]

**Category:** Operations

**Database:** DBC

| View Column        | Data Type   | Format              | Comment  |
|--------------------|---|---------------------|--|
| DataBaseName       | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)              | Returns the name of the database or user in which function alias object resides.   |
| FunctionAliasName  | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)              | Returns the name of the Function Alias.  |
| FunctionAliasType  | VARCHAR(6)<br>UNICODE<br>NOT CASESPECIFIC               | X(6)                | Returns if the Function Alias is for native or remote.   |
| CreatorName        | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)              | Returns the Teradata User who issued the CREATE FUNCTION MAPPING statement.  |
| AuthorizationName  | VARCHAR(128)<br>UNICODE<br>UPPERCASE                    | X(128)              | Returns name of the authorization when authorization is defined for function mapping object else NULL  |
| AuthorizationType  | CHAR(1)<br>LATIN UPPERCASE                              | X(1)                | Returns T when INVOKER TRUSTED authorization is defined for function mapping, S when DEFINER TRUSTED authorization is defined for server name of the authorization when authorization is defined for function mapping object else NULL |
| CreateTimeStamp    | TIMESTAMP(0)  | YYYY-MM-DDBHH:MI:SS | Returns the time when CREATE FUNCTION MAPPING Statement was issued.  |
| LastAlterName      | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)              | Returns the name of the user who last replaced the dictionary row.   |
| LastAlterTimeStamp | TIMESTAMP(0)  | YYYY-MM-DDBHH:MI:SS | Returns the time the dictionary row was last updated.  |

## FunctionsV[X]

**Category:** Operations

**Database:** DBC

| View Column        | Data Type   | Format  | Comment  |
|--------------------|---|---------|--|
| DatabaseName       | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)  | Returns the name of the database in which the function resides.  |
| FunctionName       | VARCHAR(128)<br>UNICODE<br>UPPERCASE<br>NOT NULL        | X(128)  | Returns the user-assigned name in internal format.   |
| SpecificName       | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)  | Returns the specific (unique) name of a function.  |
| FunctionId         | BYTE(6) NOT NULL  | X(12)   | Returns the unique ID of a function.   |
| NumParameters      | SMALLINT<br>NOT NULL                                    | ---,--9 | Returns the number of parameters of a function.  |
| ParameterDataTypes | VARCHAR(256)<br>LATIN UPPERCASE                         | X(256)  | Returns the parameter list of a function with a two-character string representing the data type for each parameter. The maximum parameter is 128.                            |
| FunctionType       | CHAR(1) LATIN<br>NOT CASESPECIFIC                       | X(1)    | Function type: A (aggregate), B (aggregate and statistical), F (scalar), R (table), S (statistical).   |
| ExternalName       | CHAR(30) LATINC<br>NOT NULL                             | X(30)   | Returns the external name of a function. For SQL UDFs, the value is always SQLUDF.   |
| SrcFileLanguage    | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL                  | X(1)    | Returns the source file language used to implement a user-defined function (UDF).  |
| NoSQLDataAccess    | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL                  | X(1)    | Returns the access indicated in the 'Create Procedure' statement. Y = No SQL in the external stored procedures; C = Contains SQL; R = Reads SQL data; M = Modifies SQL data; |
| ParameterStyle     | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL                  | X(1)    | Returns the parameter passing convention for the function or an External Stored Procedure. The following codes are used: S = SQL, G = TD_GENERAL, I = for internal.          |

| View Column         | Data Type                                | Format             | Comment   |
|---------------------|--|--------------------|---|
| DeterministicOpt    | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL   | X(1)               | Returns whether the function returns the same results for identical inputs or not. These are the values: Y = deterministic, N = not deterministic. The default is N.  |
| NullCall            | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL   | X(1)               | Returns whether the function is called on null input.   |
| PrepareCount        | CHAR(1)<br>LATIN UPPERCASE               | X(1)               | Returns the single character field representing the Prepare Count statistical option.   |
| ExecProtectionMode  | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL   | X(1)               | Returns whether the function can be executed directly with a call statement (not protected) or indirectly as a separate process (protected) a call statement (not protected) or indirectly as a separate process (protected). |
| ExtFileReference    | VARCHAR(1000)<br>UNICODE<br>CASESPECIFIC | X(1000)            | Saves the external file reference provided in the CREATE/REPLACE FUNCTION or external stored procedure statement.   |
| CharacterType       | SMALLINT<br>NOT NULL                     | ---,--9            | Returns the character type that was the default when the function or External Stored Procedure was created.   |
| Platform            | CHAR(8) LATIN<br>UPPERCASE<br>NOT NULL   | X(8)               | Returns the platform of the system on which the object is compiled.   |
| InterimFldSize      | INTEGER NOT NULL                         | --,---,---,<br>--9 | Returns the interim field size defined for an aggregate user-defined function.  |
| RoutineKind         | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL   | X(1)               | Returns a single character indicating the family of the routine.  |
| ParameterUDTIds     | VARBYTE(512)                             | X(1024)            | Returns a series of 4-byte UDT-type identifiers.  |
| AuthIdUsed          | BYTE(6)                                  | X(12)              | Returns the identifier of the authorization being used for the External Stored Procedure or User Defined Function.  |
| MaxOutParameters    | SMALLINT<br>NOT NULL                     | ---,--9            | Returns the maximum output parameters for a table function.   |
| GLOPSetDatabaseName | VARCHAR(128)<br>UNICODE<br>UPPERCASE     | X(128)             | Returns the name of the database to which the GLOP set belongs for the member clause.   |

| View Column       | Data Type                                   | Format | Comment   |
|-------------------|---|--------|---|
| GLOPSetMemberName | VARCHAR(128)<br>UNICODE<br>UPPERCASE        | X(128) | Returns the name of the GLOP set to which the XSP/UDF/UDM belongs. It is zero length when the external routine or UDF/UDM is not a member of the set. |
| RefQueryband      | CHAR(1)<br>LATIN UPPERCASE                  | X(1)   | Returns Y if the function references the current query bands for the session. The default is N.   |
| ExecMapName       | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC | X(128) | Returns the execution map name of a table operator. NULL indicates the table operator does not have a execute map.                                    |
| ExecMapColocName  | VARCHAR(128)<br>UNICODE<br>UPPERCASE        | X(128) | Returns the ColocationName of the execution map of a table operator.  |

## Usage Notes

You can use this view to query for information on a particular function.

For information about the possible values for the NoSQLDataAccess and RoutineKind columns, see "NoSQLDataAccess Column" and "RoutineKind Column."

## Corresponding Tables

The X view references these additional tables:

- DBC.AccessRights
- DBC.Owners

## Possible Values for ExecProtectionMode

| Value | Description      |
|-------|------------------|
| P     | Protected mode   |
| U     | Unprotected mode |

## Possible Values for FunctionType

| Value | Description               |
|-------|---------------------------|
| A     | Aggregate                 |
| B     | Aggregate and statistical |
| C     | Contract function         |

| Value | Description               |
|-------|---------------------------|
| E     | External stored procedure |
| F     | Scalar                    |
| H     | User-defined method       |
| I     | Internal type method      |
| L     | Table operator            |
| R     | Table                     |
| S     | Statistical               |

### Possible Values for MaxOutParameters

| Value     | Description                                  |
|-----------|--|
| 0         | Table function with fixed output parameters. |
| 1 to 2048 | Varying output column table function.        |

### Possible Values for ParameterStyle

| Value | Description       |
|-------|-------------------|
| S     | SQLStyle          |
| G     | GeneralStyle      |
| I     | InternalStyle     |
| J     | JavaStyle         |
| O     | SQLTable          |
| N     | InternalSQL_Style |

### Possible Values for PrepareCount

| Value | Description   |
|-------|---|
| Y     | PrepareCount option is selected for the statistical function. |
| N     | PrepareCount option is not selected.                          |

**Possible Values for RefQueryband**

| Value | Description  |
|-------|--|
| Y     | The function references the session query bands.         |
| N     | The function does not reference the session query bands. |

**Possible Values for SrcFileLanguage**

| Value | Description |
|-------|-------------|
| C     | C           |
| P     | C++         |
| J     | JAVA        |
| S     | SQL         |
| A     | SAS         |

**Example: Using FunctionsV**

The following SELECT statement displays information about the overloaded functions named 'concat' in 'dba' database:

```
SELECT SpecificName, NumParameters, ParameterDataTypes
FROM FunctionsV
WHERE DatabaseName = 'dba' AND FunctionName = 'concat'
ORDER BY 1,2,3;
```

Result:

| SpecificName | NumParameters | ParameterDataType |
|--------------|---------------|-------------------|
| -----        | -----         | -----             |
| concat       | 2             | CFCF              |
| concat_3     | 3             | CFCFCF            |
| concat_4     | 4             | CFCFCFCF          |

**GlobalDBSpaceV[X]**

**Category:** Accounting

**Database:** DBC

| View Column        | Data Type   | Format                         | Comment  |
|--------------------|---|--------------------------------|--|
| DatabaseName       | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                         | Returns the name of a database or one of the special system keywords: ALL / DEFAULT / PUBLIC.  |
| AccountName        | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                         | Expanded account in effect when a request was submitted. SYSTEMACCOUNTNO tracks console utility activity such as table rebuild, Diskcopy, or Scandisk.         |
| MaxPerm            | BIGINT NOT NULL   | --,---,---,---,<br>---,---,--9 | Returns the maximum allocation in bytes for permanent and user defined temporary table space.  |
| MaxSpool           | BIGINT NOT NULL   | --,---,---,---,<br>---,---,--9 | Returns the maximum allocation in bytes for spool and system defined temporary table space.  |
| MaxTemp            | BIGINT NOT NULL   | --,---,---,---,<br>---,---,--9 | Returns the maximum allocation in bytes for temporary table space.   |
| AllocatedPerm      | BIGINT NOT NULL   | --,---,---,---,<br>---,---,--9 | Returns the current total allocation in bytes for permanent and user defined temporary table space.  |
| AllocatedSpool     | BIGINT NOT NULL   | --,---,---,---,<br>---,---,--9 | Returns the current total allocation in bytes for spool and other system defined temporary table space.  |
| AllocatedTemp      | BIGINT NOT NULL   | --,---,---,---,<br>---,---,--9 | Returns the current total allocation in bytes for temporary tables space.  |
| PeakAllocatedPerm  | BIGINT NOT NULL   | --,---,---,---,<br>---,---,--9 | Returns the peak permanent space allocated in the database/user across the system since the last initialize.   |
| PeakAllocatedSpool | BIGINT NOT NULL   | --,---,---,---,<br>---,---,--9 | Returns the peak spool space allocated to the user across the system since the last initialize.  |
| PeakAllocatedTemp  | BIGINT NOT NULL   | --,---,---,---,<br>---,---,--9 | Returns the peak temporary space allocated to the user across the system since the last initialize.  |
| MaxProfileSpool    | BIGINT  | --,---,---,---,<br>---,---,--9 | Returns the profile SPOOL space limit for the user if the user is assigned a profile which has a SPOOL space setting. Otherwise, this column has a null value. |
| MaxProfileTemp     | BIGINT  | --,---,---,---,<br>---,---,--9 | Returns the profile TEMP space limit for the user if the user is assigned a profile which has a TEMP space setting. Otherwise, this column has a null value.   |

| View Column  | Data Type   | Format  | Comment  |
|--------------|---|---------|--|
| PermSkew     | SMALLINT  | ---,--9 | Returns the permissible skew limit percent for permanent space usage at the AMP level. |
| SpoolSkew    | SMALLINT  | ---,--9 | Returns the permissible skew limit percent for spool space usage at the AMP level.     |
| TempSkew     | SMALLINT  | ---,--9 | Returns the permissible skew limit percent for temporary space usage at the AMP level. |
| SpaceMapName | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)  | Returns the contiguous space map name of the database/user.                            |

## Usage Notes

GlobalDBSpaceV[X] returns the space limits for a given database or user at the system level. This view returns the total allocated space sizes to the AMPs in the system and the total peak allocated space sizes across the system. This view also returns the skew limits associated with the different space values.

### AppProxyUser and TrustUserName

These columns are not returned in the X or VX views.

## Example Requests for Space Information

### About These Requests

The following table provides examples of querying Data Dictionary tables for space information:

| Information Required           | Example Request   |
|--------------------------------|---|
| Total disk space in the system | <pre>SELECT SUM(MaxPerm) FROM DBC.DiskSpaceV;</pre> <p>You could also use the following request. The information is in a single row for a given database and the request is more efficient. DBC.DiskSpaceV has one row per database per AMP, so it scans more rows.</p> <pre>SELECT SUM(MaxPerm) FROM DBC.GlobalDBSpaceV;</pre> |
| Disk space currently in use    | <pre>SELECT SUM(CurrentPerm), SUM(MaxPerm),<br/>(cast(SUM(currentperm) as float) / cast((NULLIFZERO(SUM(maxperm))<br/>) as float) * 100)<br/>(TITLE '%DiskSpace_in_use', FORMAT 'zz9.99')<br/>FROM DBC.DiskSpaceV;</pre>  |

| Information Required                                    | Example Request   |
|---|---|
| Disk space for a given database                         | <pre>SELECT sum(MaxPerm) FROM DBC.DiskSpaceV WHERE databasename='xxxx';</pre> <p>The preceding request uses a partition scan to obtain the result. This processing is expected to be very efficient. Alternatively, use the following request; notice aggregation is not required, and this request is even more efficient because it is a row-hashed look up.</p> <pre>SELECT MaxPerm FROM DBC.GlobalDBSpaceV WHERE databasename='xxxx';</pre>               |
| Percentage of disk space that is available for spool    | <pre>SELECT ((cast((SUM(MaxPerm) - SUM(CurrentPerm)) as float)/ cast(NULLIFZERO(SUM(MaxPerm))) as float))*100 (TITLE'% Avail for Spool', format'zz9.99') FROM DBC.DiskSpaceV;</pre>   |
| Percentage of space used by each database in the system | <pre>SELECT Databasename (format 'X(12)') ,SUM(maxperm) ,SUM(currentperm) ,(((cast(SUM(currentperm) as float))/ cast(NULLIFZERO(SUM(maxperm)) as float)) * 100) (FORMAT 'zz9.99%', TITLE 'Percent // Used') FROM DBC.DiskSpaceV GROUP BY 1 ORDER BY 4 DESC WITH SUM (currentperm), SUM(maxperm);</pre>  |
| Users who are running out of permanent space            | <pre>SELECT Databasename (format 'X(12)') ,SUM(maxperm) ,SUM(currentperm) ,(((cast(SUM(currentperm) as float))/ cast(NULLIFZERO(SUM(maxperm)) as float)) * 100) (format 'zz9.99%', TITLE 'Percent // Used') FROM DBC.DiskSpaceV GROUP BY 1 HAVING ((cast(SUM(currentperm) as float)) /cast(NULLIFZERO(SUM(m axperm)) as float)) &gt; 0.9 ORDER BY 4 DESC;</pre> <p>You can change the value 0.9 to whatever threshold ratio is appropriate for your site.</p> |
| Users who are using a lot of spool                      | <pre>SELECT databasename ,SUM(peaks pool) FROM DBC.DiskSpaceV GROUP BY 1 HAVING SUM(peaks pool) &gt; 5000000000 ORDER BY 2 DESC;</pre>  |

| Information Required   | Example Request  |
|--|--|
|  | <p>You can change the value 500000000 to whatever value is appropriate for your site. Some sites with more space may have greater tolerance for higher spool usage and spool limits.</p>   |
| Users and databases with a non-zero skew limit on permanent space (global PERM space accounting) | <pre>SELECT databasename ,permskew FROM DBC.GlobalDBSpaceV WHERE permskew &gt; 0 ORDER BY 2 DESC;</pre> <p>The request provides the users and databases for which the system allocates space dynamically to AMPs as needed. The skew limit allows some AMPs to exceed the per-AMP quota.</p>   |
| Users and databases having permanent space skewed but defined with 0 skewlimit                   | <pre>SELECT COALESCE((cast(MAX(currentPerm) as float)/ cast(NULLIFZERO(AVG(currentPerm)) as float) - 1), 0) * 100 (FORMAT '9999.99') AS spaceskew, databasename FROM DBC.diskspacev GROUP BY 2 HAVING spaceskew &gt; 125.0 WHERE permskew = 0;</pre> <p>The request checks databases that have space usage skewed beyond a normal 25%. Change 125.0 to any value that suits your needs. For resulting databases, you may want to consider defining a smaller permskewlimit to better manage space. SkewLimit &gt; 0 implies need-based space allocations for AMPs.</p> |
| Current permanent usage and AMP level space allocations for a given database                     | <pre>SELECT vproc, currentperm, maxperm, AllocatedPerm FROM DBC.DiskSpaceV WHERE databasename='xxxx' order by vproc;</pre>   |
| Allocation for databases that have skewed permanent usage defined                                | <pre>SELECT databasename, allocatedperm, maxperm FROM DBC.GlobalDBSpaceV WHERE permskew &gt; 0 or permskew IS NULL;</pre>  |
| Unallocated space remaining in the database  | <pre>SELECT databasename, allocatedperm, maxperm,maxperm- allocatedperm (TITLE 'UnallocatedSpace'), permskew FROM DBC.GlobalDBSpaceV;</pre>  |
| AMPs that are almost running out physical space  | <pre>SELECT SUM(maxperm) m, SUM(currentperm+currentspool+currenttemp) c , cast(((m-c)*100.00) as float)/cast(m as float) (format 'zz9. 99%') p, vproc FROM DBC.diskspacev group by vproc having p &lt; 5.0;</pre>  |

| Information Required  | Example Request  |
|---|--|
|   | The preceding request returns all AMPs whose actual usage leaves less than 5% of their total space remaining. Change 5.00 to another value as required.  |
| The permanent space use for databases and users associated with a given default map | <pre>SELECT dbv.databasename, sum(currentperm), sum(maxperm) FROM DBC.DiskSpaceV dbspace, DBC.DatabasesV dbv WHERE dbspace.databasename = dbv.databasename AND defaultmapname='td_map5' GROUP BY 1;</pre>  |
| The size of the tables defined in a given map in a given database                   | <pre>SELECT tabsz.databasename, tabsz.tablename, vproc, sum(currentperm) FROM DBC.TableSizeV tabsz, DBC.TablesV tabv WHERE tabsz.databasename = tabv.databasename AND tabsz.tablename = tabv.tablename AND tabv.mapname='td_map1' and tabsz.databasename='SysAdmin' GROUP BY 1,2,3 ORDER BY 1,2,3;</pre> |
| Permanent space use by map for each database in each AMP                            | <pre>SELECT tabv.mapname, tabv.databasename, vproc, sum(currentperm) FROM DBC.TableSizeV tabsz, DBC.TablesV tabv WHERE tabsz.databasename = tabv.databasename AND tabsz.tablename = tabv.tablename GROUP BY 1,2,3 ORDER BY 1,2,3;</pre>  |
| Permanent space use by map for each database in the system                          | <pre>SELECT tabv.mapname, tabv.databasename, sum(currentperm) FROM DBC.TableSizeV tabsz, DBC.TablesV tabv WHERE tabsz.databasename = tabv.databasename AND tabsz.tablename = tabv.tablename GROUP BY 1,2 ORDER BY 1,2,3;</pre>   |
| The maximum allocated peak spool for a given user                                   | <pre>SELECT databasename, PeakAllocatedSpool from DBC.GlobalDBSpaceV;</pre>  |

## HostsInfoV

**Category:** Operations

**Database:** DBC

| View Column    | Data Type  | Format | Comment  |
|----------------|--|--------|--|
| LogicalHostId  | SMALLINT NOT NULL                                    | -(5)9  | Returns a unique identifier of the logon source for the logged query. A value of zero indicates an internal session. |
| HostName       | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128) | Returns the unique name defined for this client system.  |
| DefaultCharSet | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128) | Returns the name of the character set to be used as the default for this client system.                              |

## Usage Notes

If this view does not exist or no rows are found, user-defined international character sets are not available (see [CharsetsV](#)), or have not been assigned as host defaults. In this case, the standard default is used (EBCDIC for IBM mainframe hosts and ASCII for all others).

### Possible Values for DefaultCharSet

- EBCDIC
- ASCII
- The name of a user-defined character set as displayed in the CharSets view.

## Example: Using HostsInfoV

The following SELECT statement selects any character sets assigned by the user as the defaults for the client systems in the system configuration:

```
SELECT * FROM DBC.HostsInfoV;
```

Result:

| LogicalHostId | HostName | DefaultCharSet   |
|---------------|----------|------------------|
| -----         | -----    | -----            |
| 136           | VM       | Norwegian_EBCDIC |
| 137           | LAN      | ASCII            |

## Related Topics

For more information about the user-defined character sets and the values in DBC.HostsInfo, see *Teradata Vantage™ - Advanced SQL Engine International Character Set Support*, B035-1125.

## IndexConstraintsV[X]

**Category:** Integrity

**Database:** DBC

| View Column         | Data Type   | Format   | Comment   |
|---------------------|---|----------|---|
| DatabaseName        | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)   | Returns the name of the database in which the table resides.  |
| TableName           | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)   | Returns the name of the table on which the index is built.  |
| IndexName           | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC             | X(128)   | Returns the name of the reference index.  |
| IndexNumber         | SMALLINT  | ---,--9  | IndexConstraintsVX.IndexNumber is the internal number assigned to the associated index, if any. This field is 1 for a primary index or primary AMP index. This field is NULL if the partitioning is not associated with an index. |
| ConstraintType      | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL                  | X(1)     | IndexConstraintsVX.ConstraintType is Q (partitioning constraint).   |
| ConstraintText      | VARCHAR(16000)<br>UNICODE<br>CASESPECIFIC               | X(16000) | IndexConstraintsVX.ConstraintText is the unresolved condition text of the partitioning constraint (implied constraint determined from the partitioning levels).   |
| ConstraintCollation | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL                  | X(1)     | IndexConstraintsVX.ConstraintCollation is the collation to be used in evaluating the constraint: E (EBCDIC), A (ASCII), M (MULTINATIONAL), C (CHARSET_COLL), J (JIS_COLL), or U (use the session collation).                      |
| CollationName       | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC             | X(128)   | Returns the name of the CHARSET_COLL collation. If ConstraintCollation is not U, it is the constraint collation name; otherwise, the value is NULL.   |
| CreatorName         | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)   | Index constraint creator.   |

| View Column               | Data Type                   | Format                    | Comment   |
|---------------------------|-----------------------------|---------------------------|---|
| CreateTimeStamp           | TIMESTAMP(0)                | YYYY-MM-DDBHH:MI:SS       | IndexConstraintsVX.CreateTimeStamp is the date and time the constraint was created.                                     |
| CharSetID                 | BYTEINT                     | -(3)9                     | Returns a number uniquely identifying the character set of the CHARSET_COLL collation.                                  |
| SessionMode               | CHAR(1)<br>LATIN UPPERCASE  | X(1)                      | Returns the name of the session mode used to determine the default case sensitivity in evaluating the index constraint. |
| ResolvedCurrent_Date      | DATE                        | YY/MM/DD                  | Returns the last resolved value of CURRENT_DATE.  |
| ResolvedCurrent_TimeStamp | TIMESTAMP(6) WITH TIME ZONE | YYYY-MM-DDBHH:MI:SS.S(6)Z | Returns the last resolved value of CURRENT_TIMESTAMP.   |

## Usage Notes

This view is deprecated. Use the [PartitioningConstraintsV\[X\]](#) system view instead.

For information about the possible values for the ConstraintType column, see "ConstraintType Column."

## Corresponding Tables

The X view references these additional tables:

- DBC.AccessRights
- DBC.Owners
- DBC.RoleGrants
- DBC.Roles

## CharSetID

If the character set is user-defined, the ID in the CharSetID column should also exist in DBC.CharTranslations.CharSetID.

## IndexName

For a partitioning constraint of a partitioned table or join index with neither a primary index nor a primary AMP index, the IndexName is NULL. For a partitioning constraint of a partitioned table with a primary index or a primary AMP index, the IndexName is NULL or not NULL depending on whether the index was given a name.

### Possible Values for ResolvedCurrent\_TimeStamp and ResolvedCurrent\_Date

- This is the last reconciled timestamp or date if the object is a join index or a table that is defined using:
  - CURRENT\_TIMESTAMP
  - CURRENT\_DATE or DATE
 Either the partition, JI definition, or temporal table has a system-defined join index.
- NULL in all other cases.

### Possible Values for SessionMode

| Value | Description   |
|-------|---|
| A     | ANSI  |
| T     | Teradata  |
| NULL  | For constraints not involving comparison of character data. |

### Example: Select a List of Tables and Join Indexes with Partitioning from IndexConstraints

The following query retrieves a list of tables and join indexes that have partitioning, including their index constraint text:

```
SELECT DatabaseName, TableName (TITLE 'Table/Join Index Name'),
       ConstraintText
FROM DBC.IndexConstraints
WHERE ConstraintType = 'Q'
ORDER BY DatabaseName, TableName;
```

### Example: Select a List of Tables and Join Indexes with Single-level Partitioning from IndexConstraints

The following query retrieves a list of tables and join indexes that have single-level partitioning:

```
SELECT DatabaseName, TableName (TITLE 'Table/Join Index Name')
FROM DBC.IndexConstraints
WHERE ConstraintType = 'Q'
AND ( SUBSTRING(ConstraintText FROM 1 FOR 13) < 'CHECK (/02*/'
      OR SUBSTRING(ConstraintText FROM 1 FOR 13) > 'CHECK (/15*/')
ORDER BY DatabaseName, TableName;
```

# IndexStatsV[X]

**Category:** Optimizer Statistics

**Database:** DBC

| View Column        | Data Type   | Format          | Comment   |
|--------------------|---|-----------------|---|
| DatabaseName       | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)          | Returns the database name of the table, column, group of columns, index, view, or query on which statistics were collected. |
| TableName          | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)          | Returns the name of the table that contains the column, group of columns, or index on which the statistics were collected.  |
| ColumnName         | VARCHAR(10000)<br>UNICODE NOT<br>CASESPECIFIC           | X(10000)        | The ColumnName column identifies a column or columns.   |
| StatsName          | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC             | X(128)          | The StatsName column contains the alias name associated with the statistic.   |
| StatsSource        | CHAR(1)<br>LATIN UPPERCASE                              | X(1)            | The StatsSource column records the method this statistic is acquired.   |
| ValidStats         | CHAR(1)<br>LATIN UPPERCASE                              | X(1)            | The ValidStats column records Database version statistics collected on.   |
| DBSVersion         | VARCHAR(32)<br>LATIN UPPERCASE                          | X(32)           | The ValidStats column indicates whether the statistics are valid or not.  |
| IndexNumber        | SMALLINT  | ---,--9         | Returns an internal number assigned to the index. A primary index has an index number of 1.                                 |
| SampleSignature    | VARCHAR(256)<br>LATIN NOT<br>CASESPECIFIC               | X(256)          | The SampleSignature column is the Sample options encoded as a 10 character signature.                                       |
| SampleSizePct      | DECIMAL(5,2)  | ----.99         | The SampleSizePct column is the Sample size percent used when collecting statistics.  |
| ThresholdSignature | VARCHAR(512)<br>LATIN NOT<br>CASESPECIFIC               | X(512)          | The ThresholdSignature column is the Threshold options encoded as a 17 character signature.                                 |
| MaxIntervals       | SMALLINT  | ---,--9         | The MaxIntervals column is the User-specified maximum number of intervals.  |
| MaxValueLength     | INTEGER   | --,---,----,--9 | The MaxValueLength column is the User-specified maximum value length.   |

| View Column           | Data Type    | Format                  | Comment   |
|-----------------------|--------------|-------------------------|---|
| RowCount              | FLOAT        | ---,---,---,---,<br>--9 | The RowCount column is the row count of the table, view or query on which statistics are collected.         |
| UniqueValueCount      | FLOAT        | ---,---,---,---,<br>--9 | The UniqueValueCount column is the Number of unique values of the ExpressionList.                           |
| PNullUniqueValueCount | FLOAT        | ---,---,---,---,<br>--9 | The PNullUniqueValueCount column is the Number of unique values from rows with partial nulls.               |
| NullCount             | FLOAT        | ---,---,---,---,<br>--9 | The NullCount column is the Number of nulls of the ExpressionList.  |
| AllNullCount          | FLOAT        | ---,---,---,---,<br>--9 | The AllNullCount column is the Number of all nulls of the ExpressionList.                                   |
| HighModeFreq          | FLOAT        | ---,---,---,---,<br>--9 | The HighModeFreq column is the highest frequency of values of the ExpressionList.                           |
| PNullHighModeFreq     | FLOAT        | ---,---,---,---,<br>--9 | The PNullHighModeFreq column is the highest frequency of values having partial nulls of the ExpressionList. |
| CreateTimeStamp       | TIMESTAMP(0) | YYYY-MM-DDBHH:MI:SS     | The CreateTimeStamp column is the statistics creation time stamp.   |
| LastCollectTimeStamp  | TIMESTAMP(0) | YYYY-MM-DDBHH:MI:SS     | The LastCollectTimeStamp column is the Last statistics collection time stamp.                               |
| LastAlterTimeStamp    | TIMESTAMP(0) | YYYY-MM-DDBHH:MI:SS     | The LastAlterTimeStamp column is the last user updated time stamp.  |

## Usage Notes

### Referenced Columns

Many of the Data Dictionary view columns have referenced table columns. That is, the value in the view column corresponds to a value in the selected column referenced in the table. It would be meaningful to join the view and the referenced table based on the selected column and the referenced column.

Referenced columns for this view are:

| View Column | Referenced Column       |
|-------------|-------------------------|
| IndexNumber | DBC.Indexes.IndexNumber |

## ColumnName

- If more than one column or expression is specified, each column or expression is separated by a comma.
- The maximum number of columns is 64.
- If expressions are in the list, the maximum number of columns can be reduced past the limit of 64, depending on the combined total size of the text in the expressions.
- If the combined total size of the expression text causes the maximum column limit to be less than the actual number of columns in the list, an error occurs.

## Possible Values for ValidStats

| Value | Description               |
|-------|---------------------------|
| T     | Statistics are valid.     |
| F     | Statistics are not valid. |

### Note:

Statistics can be not valid if the query cannot be parsed. For example, if a table is dropped, all statistics referencing it are not valid.

## SampleSizePct

If sampling is not used, the SampleSizePct column is set to 0 or 100.

## StatsSource

The StatsSource column records the method by which this statistic is acquired. For information about the possible values for the StatsSource column, see "StatsSource Column."

## To Get Information Not Contained in This View

This view does not contain:

- Statistics on tables protected by row-level security. To get these statistics, create views on DBC.StatsTbl. You can view details about this table in Teradata SQL Assistant or Teradata Studio Express.
- Information about column attributes. To get this information, join the DBC.StatsTbl table to the DBC.TVFields table. You can view details about this table in Teradata SQL Assistant or Teradata Studio Express.

## Example: Using IndexStatsV

This example assumes the following statistics have been collected:

```

STATISTICS
  INDEX (o_orderkey)
  ,INDEX (o_custkey, o_orderstatus)
  ON Orders;

```

This query can be used to retrieve the statistics:

```

SELECT * FROM DBC.IndexStatsV
  WHERE databasename = 'sales'
  AND tablename = 'orders';

```

## Related Topics

| For more information about statistics collected on ... | See ...   |
|--|---|
| non-indexed columns and single-column indexes          | <a href="#">ColumnStatsV</a> .                          |
| groups of non-indexed columns                          | <a href="#">MultiColumnStatsV</a> .                     |
| tables   | <a href="#">StatsV</a> and <a href="#">TablesV[X]</a> . |
| materialized temporary tables                          | <a href="#">TempTableStatsV</a> .                       |
| single expressions                                     | <a href="#">ExpStatsV</a> .                             |
| multiple expressions                                   | <a href="#">MultiExpStatsV</a> .                        |

## IndexUseCountV[X]

**Category:** Accounting

**Database:** DBC

| View Column  | Data Type   | Format | Comment  |
|--------------|---|--------|--|
| DatabaseName | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | Returns the Database Name of the object for which access count and/or UDI counts are recorded. |
| TableName    | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | Returns the name of the table whose index was used for a request.                              |
| IndexName    | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC             | X(128) | Returns the name of an index which got accessed for a query.                                   |

| View Column | Data Type   | Format                         | Comment  |
|-------------|---|--------------------------------|--|
| IndexNumber | SMALLINT NOT NULL                                       | ---,--9                        | Returns the primary or secondary index of a table used for table access.   |
| FieldName   | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                         | Returns Field Name of a table if usage type is a DML type; StatsId if the usage is for statistics (the optimizer usage of statistics). |
| UsageType   | CHAR(3) LATIN<br>UPPERCASE NOT NULL                     | X(3)                           | Returns the usage type of the object. It can be either DML or STA.   |
| AccessCount | BIGINT  | --,---,---,---,<br>---,---,--9 | Returns the number of accesses since the last reset by the user.   |

## Usage Notes

### Possible Values for IndexNumber

The IndexNumber is an internal number assigned to the index.

| Value   | Description                        |
|---|------------------------------------|
| 1   | Primary index or primary AMP index |
| Multiple of 4 (that is, a number between 4 and 128) | Secondary index                    |

### Possible Values for UsageType

| Name | Description                   |
|------|-------------------------------|
| DML  | Data Manipulation Language    |
| STA  | Optimizer usage of statistics |

## Example: Using IndexUseCountV

The following SELECT statement shows the number of index accesses occurring on a particular object:

```
SELECT IndexNumber, AccessCount FROM DBC.IndexUseCountV WHERE DatabaseName =
'Personnel' AND TableName = 'Employee';
```

Result:

|             |             |
|-------------|-------------|
| IndexNumber | AccessCount |
| -----       | -----       |
| 1           | 1           |

## IndicesV[X]

**Category:** Schema

**Database:** DBC

| View Column    | Data Type   | Format | Comment   |
|----------------|---|--------|---|
| DatabaseName   | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | Returns the name of a database.   |
| TableName      | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | Returns the name of a table, join index, or hash index.   |
| IndexNumber    | SMALLINT<br>NOT NULL                                    | ZZ9    | IndicesVX.IndexNumber is the internal number assigned to the index. A primary index or primary AMP index has an index number of 1. A secondary index has an index number that is a multiple of 4 between 4 and 128.   |
| IndexType      | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL                  | X(1)   | IndicesVx.IndexType is P(NPPI), Q(PPI), A(PA), S(SI), U(unique), K(primary key), J(join index), N(hash index), V(value-ordered SI), H(hash-ordered ALL SI), O(value-ordered ALL SI), I(composite SI ordering column), 1(field1), 2(field2), G(geospatial NUSI). |
| UniqueFlag     | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL                  | X(1)   | Returns a code to indicate whether the index is unique. The following codes are used: Y = Yes; index is unique, N = No; index is not unique.  |
| IndexName      | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC             | X(128) | Returns the name of the reference index.  |
| ColumnName     | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | Returns column names participating in an index that is accessible to the current user.  |
| ColumnPosition | SMALLINT<br>NOT NULL                                    | Z9     | Returns the position of the column in the index.  |

| View Column         | Data Type   | Format                             | Comment   |
|---------------------|---|------------------------------------|---|
| CreatorName         | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                             | Table or database creator. For DBC. AllRights, the grantor of explicit rights, otherwise the user who executed the CREATE statement.  |
| CreateTimeStamp     | TIMESTAMP(0)  | YYYY-MM-DDBHH:MI:SS                | The CreateTimeStamp column returns the date and time the index was created.   |
| LastAlterName       | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                             | Returns the name of the user who last updated the dictionary row.   |
| LastAlterTimeStamp  | TIMESTAMP(0)  | YYYY-MM-DDBHH:MI:SS                | Returns the time the dictionary row was last updated.   |
| IndexMode           | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC                    | X(1)                               | IndicesVX.IndexMode is H (secondary index rows are hash distributed to the AMPs), L (secondary index rows are on the same AMP as the referenced data row), or NULL (PI or PA). If the index type is J or N, index mode is L but has no meaning. |
| AccessCount         | BIGINT  | --,---,---,<br>---,---,---,<br>--9 | Returns the access count for the corresponding database object.   |
| LastAccessTimeStamp | TIMESTAMP(0)  | YYYY-MM-DDBHH:MI:SS                | Returns the time that the corresponding object was last accessed.   |
| UniqueOrPK          | CHAR(1) LATIN<br>UPPERCASE                              | X(1)                               | DBC.INDICESVX.UniqueOrPK is U (unique), K (primarykey), or NULL (When the unique or primarykey constraint is not associated with time dimension or the row is not for a unique or primarykey constraint description.)                           |
| VTConstraintType    | CHAR(1) LATIN<br>UPPERCASE                              | X(1)                               | DBC.INDICESVX.VTConstraintType is C (current), S (sequenced), N (nonsequenced), or NULL (is used for all rows on a table that do not support ValidTime, otherwise an index row.)  |
| TTConstraintType    | CHAR(1) LATIN<br>UPPERCASE                              | X(1)                               | DBC.INDICESVX.TTConstraintType is C (current), S (sequenced), N (nonsequenced), or NULL (is used for all rows on a table that do not support TransactionTime, otherwise an index row.)  |

| View Column       | Data Type   | Format       | Comment   |
|-------------------|---|--------------|---|
| SystemDefinedJI   | CHAR(1) LATIN<br>UPPERCASE                              | X(1)         | DBC.INDICESVX.SystemDefinedJI records whether a join index is system-defined or user-defined. Y if the entry is corresponding to a system-defined join index or NULL otherwise  |
| IndexDatabaseName | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)       | The IndicesVX.IndexDatabaseName field identifies the data base, if any, for the index.  |
| LDIType           | CHAR(1) LATIN<br>UPPERCASE                              | X(1)         | LDIType is set to NULL for any secondary index on non-LDI tables. For LDI tables/ join indexes it is set to "E" for a USI and EIRID NUSI on a LDI table/join index, it is set to "N" if the NUSI does not contain the RowLoadID.                            |
| RowSizeFormat     | VARCHAR(1)<br>LATIN NOT<br>CASESPECIFIC                 | X(1)         | RowSizeFormat indicates the row size; "0" for a format of up to 64KB, and "1" for a format of up to 1MB.  |
| TimeZero          | DATE  | YY/MM<br>/DD | Time Zero is the Timezeor_date parameter specified in Primary Time Index clause of CREATE TABLE. It is a starting time point associated with a Time Series table.   |
| TimeBucketUnit    | BYTEINT   | -(3)9        | Time Bucket Unit can have CAL_YEARS(1), CAL_MONTHS(2), CAL_DAYS(3), WEEKS(4), DAYS(5), HOURS(6), MINUTES(7), SECONDS(8), MILLISECONDS(9), MICROSECONDS(10). It is the timebucket_duration parameter specified in Primary Time Index clause of CREATE TABLE. |
| TimeBucketValue   | SMALLINT  | ---,--9      | Time Bucket Value in conjunction with TimeBuctUnit determines the time bucket size for a Time Series table. It is the timebucket_duration parameter specified in Primary Time Index clause of CREATE TABLE.   |
| TSFlags           | BYTE(1)   | X(2)         | bit 1 - 1 a Time Series (TS) table, 0 not a TS table. bit 2 - 1 a TS table with TD_SEQNO column, 0 no TD_SEQNO. bit 3 - 1 a TS table with TD_TIMEBUCKET column, 0 no TD_TIMEBUCKET. bit 4 - 1 a TS table is defined with COLUMNS clause, 0 no COLUMNS.      |

## Usage Notes

One row is returned from the Indices view for each column in each index. Therefore, a query on an index made up of multiple columns returns multiple rows.

Some of the column values show information related to a Teradata temporal table or an ANSI temporal table. For more information about these tables, see *Teradata Vantage™ ANSI Temporal Table Support*, B035-1186 and *Teradata Vantage™ Temporal Table Support*, B035-1182.

## Corresponding Tables

The X view references these additional tables:

- DBC.AccessRights
- DBC.Owners
- DBC.Roles
- DBC.RoleGrants

## Possible Values for IndexMode

| Value | Description  |
|-------|--|
| H     | Secondary index rows are hash distributed to the AMPs.   |
| L     | Secondary index rows are on the same AMP as the base data row.<br><b>Note:</b><br>If the index type is a join (J) or hash index (N) index, L is returned. In this instance, L represents an empty value. |
| NULL  | Primary index or primary AMP index   |

## Possible Values for IndexNumber

The IndexNumber is an internal number assigned to the index.

| Value   | Description                        |
|---|------------------------------------|
| 1   | Primary index or primary AMP index |
| Multiple of 4 (that is, a number between 4 and 128) | Secondary index                    |

## Possible Values for RowSizeFormat

The RowSizeFormat indicates the row size:

| Value | Description            |
|-------|------------------------|
| 0     | Row size of up to 64KB |

| Value | Description           |
|-------|-----------------------|
| 1     | Row size of up to 1MB |

## IndexType

| Value | Description   |
|-------|---|
| P     | Nonpartitioned primary index, except if the primary index is also a PRIMARY KEY or UNIQUE, then K or U, respectively, is used for this column                                     |
| Q     | Partitioned primary index with row, column, or both partitioning, except if the primary index is also a PRIMARY KEY or UNIQUE, then K or U, respectively, is used for this column |
| A     | Indicates a primary AMP index.  |
| S     | Secondary index, except if the primary index is also a PRIMARY KEY or UNIQUE, then K or U, respectively, is used for this column  |
| J     | Join index  |
| N     | Hash index  |
| K     | Primary key   |
| U     | Unique constraint   |
| V     | Value-ordered secondary index   |
| H     | Hash-ordered ALL covering secondary index   |
| O     | Valued-ordered ALL covering secondary index   |
| I     | Ordering column of a composite secondary index  |
| 1     | Indicates all the columns that comprise the first field of the join index.<br><b>Note:</b><br>This value is used for join and hash indexes.                                       |
| 2     | Indicates all the columns that comprise the second field of the join index<br><b>Note:</b><br>This value is used for join and hash indexes.                                       |
| G     | Geospatial nonunique secondary index.   |

### Note:

To determine if a table has partitioning, including when the primary index is for a primary key or unique constraint, see the `DBC.TablesV[X].PartitioningLevels` column or the `DBC.PartitioningConstraintsV[X].PartitioningLevels` column.

To determine if a primary-indexed table has partitioning (when the primary index is not for a primary key or unique constraint) view the value of the IndexType column. IndexType is set to P for a nonpartitioned primary index or Q for a partitioned primary index.

### Example: Using IndicesV

The following statement results in DBC.IndicesV[X].IndexType of 1 and 2:

```
CREATE JOIN INDEX BB_OTB_DATA.ORDER_JOIN_LINE ,NO FALLBACK ,CHECKSUM =
DEFAULT AS
SELECT (BB_OTB_DATA.lineitem.l_orderkey ,BB_OTB_DATA.orders.o_orderdate,
        BB_OTB_DATA.orders.o_custkey ,BB_OTB_DATA.orders.o_totalprice),
        (BB_OTB_DATA.lineitem.l_partkey ,BB_OTB_DATA.lineitem.l_quantity,
        BB_OTB_DATA.lineitem.l_extendedprice,          BB_OTB_DATA.lineitem.l_shipdate )
FROM
(BB_OTB_DATA.lineitem LEFT OUTER JOIN BB_OTB_DATA.orders ON
BB_OTB_DATA.lineitem.l_orderkey = BB_OTB_DATA.orders.o_orderkey )
ORDER BY BB_OTB_DATA.orders.o_orderdate ASC
PRIMARY INDEX ( l_orderkey );
```

This query results in four rows in DBC.IndicesV[X] with value “1” (the four columns in the first field of the join index) and four rows with value “2” (the four columns in the second field of the join index).

### Possible Values for LDIType

USI or NUSI defined on load isolated (LDI) tables or join indexes. Possible values are shown in the table.

| Value | Description   |
|-------|---|
| NULL  | Any secondary index on tables that are not load isolated and for a join index, primary index, or a primary AMP index on a table. LDIType is set to NULL for a join index, primary index, or a primary AMP index on a load isolated table as well as on a non-LDI table. |
| E     | The base table and the secondary index are load isolated.   |
| N     | The base table is load isolated, but the secondary index is not.  |

### Possible Values for SystemDefinedJI

| Value | Description  |
|-------|--|
| Y     | The TVM row describes a join index defined by the system. Such an index is defined when there are temporal constraints on the underlying base table. |
| NULL  | Any other objects in the system or a user-defined join index.  |

**Possible Values for TTConstraintType**

| Value | Description   |
|-------|---|
| C     | CURRENT   |
| N     | NONSEQUENCED  |
| NULL  | Used when all rows on a table do not support TRANSACTIONTIME.<br>If the table supports TRANSACTIONTIME, a value of NULL indicates that the row is for an index and not a temporal unique or primary constraint. |
| S     | SEQUENCED   |

**Possible Values for UniqueOrPK**

| Value | Description   |
|-------|---|
| U     | Unique  |
| K     | Primary key   |
| NULL  | The unique or primary key constraint is not associated with a time dimension, or the row is not for a unique or primary key constraint description. |

**Possible Values for VTConstraintType**

| Value | Description   |
|-------|---|
| A     | ANSIQUALIFIER<br><b>Note:</b><br>ANSI temporal tables require that the session temporal qualifier for systems using Teradata temporal tables be explicitly set to ANSIQUALIFIER.                                |
| C     | CURRENT   |
| N     | NONSEQUENCED  |
| NULL  | Used when all rows on a table do not support VALIDTIME.<br>If the table supports VALIDTIME, a value of NULL indicates that the row is for an index and not a temporal unique or primary constraint.             |
| S     | SEQUENCED   |
| W     | Unique or primary key constraint defined with WITHOUT OVERLAPS. ANSI valid-time table definitions can include primary key and unique constraints that prevent rows from having valid-time periods that overlap. |

## Example: Using Indices

The following SELECT statement displays index information for all the tables in the Personnel database:

```
SELECT TableName,ColumnName,ColumnPosition,IndexType,UniqueFlag FROM
  DBC.Indices
  WHERE DatabaseName = 'Personnel'
  ORDER BY TableName,ColumnPosition ;
```

The results of this query are as follows:

| TableName  | ColumnName | ColumnPosition | IndexType | UniqueFlag |
|------------|------------|----------------|-----------|------------|
| Charges    | Proj_id    | 1              | S         | N          |
| Charges    | EmpNo      | 1              | P         | N          |
| Charges    | Proj_id    | 2              | P         | N          |
| Department | DeptNo     | 1              | P         | Y          |
| Employee   | EmpNo      | 1              | P         | Y          |
| Employee   | Name       | 1              | S         | N          |
| Project    | Proj_id    | 1              | P         | Y          |

## InDoubtLogV

**Category:** Operations

**Database:** DBC

| View Column   | Data Type                   | Format         | Comment   |
|---------------|-----------------------------|----------------|---|
| LogicalHostId | SMALLINT<br>NOT NULL        | ---,--9        | Returns a unique identifier of the logon source for the logged query. A value of zero indicates an internal session.                                  |
| SessionNumber | INTEGER<br>NOT NULL         | --,---,---,--9 | Note: SessionID, SessionNo, and SessionNumber have the same meaning. Returns the session identifier of the session that had the in-doubt transaction. |
| CoordTaskId   | VARBYTE(30)<br>NOT NULL     | X(60)          | Returns the coordinator that had the in-doubt transaction.  |
| RunUnitId     | VARBYTE(30)<br>NOT NULL     | X(60)          | Returns the identity of the run unit that had the in-doubt transaction.   |
| LogonUserName | VARCHAR(128)<br>UNICODE NOT | X(128)         | Returns the ID of the user who ran the in-doubt transaction.  |

| View Column            | Data Type   | Format       | Comment  |
|------------------------|---|--------------|--|
|                        | CASESPECIFIC<br>NOT NULL                                |              |  |
| ResolvingUserLogonName | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)       | Returns the identity of the user who resolved the in-doubt transaction.  |
| CommitOrRollback       | CHAR(1) LATIN NOT<br>CASESPECIFIC<br>NOT NULL           | X(1)         | The InDoubtLogV.CommitOrRollback column identifies whether the in-doubt session was committed (C) or rolled back (R).                      |
| UserLogonDate          | DATE NOT NULL   | YY/MM<br>/DD | Returns the date that the specified user logged on.  |
| UserLogonTime          | FLOAT NOT NULL  | 99:99:99     | Returns the time that the specified user logged on.  |
| CompletionDate         | DATE NOT NULL   | YY/MM<br>/DD | The InDoubtLogV.CompletionDate column identifies the date the in-doubt session was resolved.   |
| CompletionTime         | FLOAT NOT NULL  | 99:99:99     | The InDoubtLogV.CompletionTime column identifies the time the in-doubt session was resolved.   |
| Options                | CHAR(1) LATIN NOT<br>CASESPECIFIC                       | X(1)         | This field is assigned to NULL initially. It is not used by the system and users may update this one-character column to suit their needs. |

## Usage Notes

### Possible Values for CommitOrRollback

| Value | Description |
|-------|-------------|
| C     | Committed   |
| R     | Rolled back |

## InsertUseCountV[X]

**Category:** Accounting

**Database:** DBC

| View Column  | Data Type   | Format                         | Comment  |
|--------------|---|--------------------------------|--|
| DatabaseName | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                         | Returns the Database Name of the object for which access count and/or UDI counts are recorded. |
| ObjectName   | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                         | Returns the Object Name of the object for which access count and/or UDI counts are recorded.   |
| UsageType    | CHAR(3) LATIN<br>UPPERCASE NOT NULL                     | X(3)                           | Returns the usage type of the object. It can be either DML or STA.                             |
| InsertCount  | BIGINT  | --,---,---,---,---,<br>---,--9 | Returns the number of inserts since the last reset by the user.                                |

## Usage Notes

### Possible Values for UsageType

| Name | Description                   |
|------|-------------------------------|
| DML  | Data Manipulation Language    |
| STA  | Optimizer usage of statistics |

## Example: Using InsertUseCountV

The following SELECT statement shows the number of inserts occurring on a particular object:

```
SELECT InsertCount FROM DBC.InsertUseCountV WHERE DatabaseName = 'Personnel'
AND ObjectName = 'Employee';
```

Result:

```
InsertCount
-----
9
```

## JoinIndicesV

**Category:** Schema

**Database:** DBC

| View Column         | Data Type                                      | Format | Comment   |
|---------------------|--|--------|---|
| DataBaseName        | VARCHAR(128) UNICODE NOT CASESPECIFIC NOT NULL | X(128) | Returns the name of a database.                         |
| TableName           | VARCHAR(128) UNICODE NOT CASESPECIFIC NOT NULL | X(128) | Returns the name of a table, join index, or hash index. |
| JoinIdxDataBaseName | VARCHAR(128) UNICODE NOT CASESPECIFIC NOT NULL | X(128) | Returns the join index database name.                   |
| JoinIdxName         | VARCHAR(128) UNICODE NOT CASESPECIFIC NOT NULL | X(128) | Returns the join index name.                            |
| IndexType           | CHAR(1) LATIN UPPERCASE NOT NULL               | X(1)   | Returns the type of an index as Index type.             |

## Usage Notes

### IndexType

| Value | Description   |
|-------|---|
| P     | Nonpartitioned primary index, except if the primary index is also a PRIMARY KEY or UNIQUE, then K or U, respectively, is used for this column                                     |
| Q     | Partitioned primary index with row, column, or both partitioning, except if the primary index is also a PRIMARY KEY or UNIQUE, then K or U, respectively, is used for this column |
| A     | Indicates a primary AMP index.  |
| S     | Secondary index, except if the primary index is also a PRIMARY KEY or UNIQUE, then K or U, respectively, is used for this column  |
| J     | Join index  |
| N     | Hash index  |
| K     | Primary key   |
| U     | Unique constraint   |
| V     | Value-ordered secondary index   |
| H     | Hash-ordered ALL covering secondary index   |
| O     | Valued-ordered ALL covering secondary index   |
| I     | Ordering column of a composite secondary index  |
| 1     | Indicates all the columns that comprise the first field of the join index.<br><b>Note:</b><br>This value is used for join and hash indexes.                                       |

| Value | Description   |
|-------|---|
| 2     | Indicates all the columns that comprise the second field of the join index<br><b>Note:</b><br>This value is used for join and hash indexes. |
| G     | Geospatial nonunique secondary index.   |

**Note:**

To determine if a table has partitioning, including when the primary index is for a primary key or unique constraint, see the DBC.TablesV[X].PartitioningLevels column or the DBC.PartitioningConstraintsV[X].PartitioningLevels column.

To determine if a primary-indexed table has partitioning (when the primary index is not for a primary key or unique constraint) view the value of the IndexType column. IndexType is set to P for a nonpartitioned primary index or Q for a partitioned primary index.

## Example: Create Join Index

The following statement results in DBC.IndicesV[X].IndexType of 1 and 2:

```
CREATE JOIN INDEX BB_OTB_DATA.ORDER_JOIN_LINE ,NO FALLBACK ,CHECKSUM =
DEFAULT AS
SELECT (BB_OTB_DATA.lineitem.l_orderkey ,BB_OTB_DATA.orders.o_orderdate,
        BB_OTB_DATA.orders.o_custkey ,BB_OTB_DATA.orders.o_totalprice),
        (BB_OTB_DATA.lineitem.l_partkey ,BB_OTB_DATA.lineitem.l_quantity,
        BB_OTB_DATA.lineitem.l_extendedprice,          BB_OTB_DATA.lineitem.l_shipdate )
FROM
(BB_OTB_DATA.lineitem LEFT OUTER JOIN BB_OTB_DATA.orders ON
BB_OTB_DATA.lineitem.l_orderkey = BB_OTB_DATA.orders.o_orderkey )
ORDER BY BB_OTB_DATA.orders.o_orderdate ASC
PRIMARY INDEX ( l_orderkey );
```

This query results in four rows in DBC.IndicesV[X] with value “1” (the four columns in the first field of the join index) and four rows with value “2” (the four columns in the second field of the join index).

## Example: Using JoinIndicesV

The following SELECT statement displays the join indexes defined on the table user1.oneoneone:

```
SELECT joinidxdatabasename, joinidxname
FROM DBC.JoinIndicesV
WHERE databasename = 'user1' and tablename = 'oneoneone';
```

Result:

|                     |             |
|---------------------|-------------|
| JoinIdxDataBaseName | JoinIdxName |
| -----               | -----       |
| user1               | ji          |

## JournalsV[X]

**Category:** Schema

**Database:** DBC

| View Column | Data Type  | Format | Comment   |
|-------------|--|--------|---|
| Tables_DB   | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128) | Returns the name of the database in which a data table resides that has change images written to a journal table. |
| TableName   | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128) | Returns name of a data table that has change images recorded in a journal table.                                  |
| Journals_DB | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128) | Returns the name of the database in which a journal table resides.  |
| JournalName | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128) | Returns the name of the journal table defined as the default for UserName.  |

## Example: Using JournalsV

The statement on the following screen selects information from the Journals view for the table named PriceA:

```
SELECT TableName,Tables_DB,Journals_DB,JournalName
      FROM DBC.JournalsV WHERE Tablename = 'PriceA' ;
```

Result:

|           |           |             |             |
|-----------|-----------|-------------|-------------|
| TableName | Tables_DB | Journals_DB | JournalName |
| -----     | -----     | -----       | -----       |
| PriceA    | Acctng    | Acctng      | JNLA        |
| .         | .         | .           | .           |
| .         | .         | .           | .           |
| .         | .         | .           | .           |

## LoadTablesInfoV[X]

**Category:** Operations

**Database:** DBC

| View Column    | Data Type   | Format                        | Comment  |
|----------------|---|-------------------------------|--|
| TableName      | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                        | TableName is the name of the LDI table in a concurrent load operation.   |
| DatabaseName   | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                        | DatabaseName is the database name to which the LDI table in concurrent load operation belongs to.                            |
| LoadQueryBand  | VARCHAR(256)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(256)                        | LoadQueryBand is the Query_Band value of the concurrent load operation.  |
| SingleSession  | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL                  | X(1)                          | SingleSession specifies if the concurrent load operation is performed by a single session or performed in multiple sessions. |
| ControlSessNum | INTEGER NOT NULL  | --,---,---,--9                | ControlSessNum is the session number of the session that issued BEGIN ISOLATED LOADING.                                      |
| ControlHostNum | SMALLINT NOT NULL                                       | ---,--9                       | ControlHostNum is the logical host ID of the session that issued BEGIN ISOLATED LOADING.                                     |
| StartTimeStamp | TIMESTAMP(6) WITH<br>TIME ZONE NOT NULL                 | YYYY-MM-DDBHH:MI:<br>SS.S(6)Z | StartTimeStamp is the timestamp value at the time BEGIN ISOLATED LOADING is processed.                                       |

## Usage Notes

The LoadTablesInfoV[X] view provides information on load isolated tables for all load operations from the time a BEGIN ISOLATED LOADING statement is issued.

### Possible Values for SingleSession

| Value | Description   |
|-------|---|
| Y     | Default. Specifies that the concurrent load operation is performed by a single session. |
| N     | Specifies the concurrent load operation is performed in multiple sessions.              |

## LogOnOffV[X]

**Category:** Operations

**Database:** DBC

| View Column   | Data Type  | Format             | Comment  |
|---------------|--|--------------------|--|
| LogDate       | DATE NOT NULL  | YY/MM/DD           | Returns the date that the access log entry was made.   |
| LogTime       | FLOAT NOT NULL                                       | 99:99:99.99        | Returns the time of day that the event occurred as HH:MM:SS.   |
| UserName      | VARCHAR(128)<br>UNICODE NOT CASESPECIFIC<br>NOT NULL | X(128)             | Returns the username associated with the event.  |
| AccountName   | VARCHAR(128)<br>UNICODE NOT CASESPECIFIC<br>NOT NULL | X(128)             | Expanded account in effect when a request was submitted. SYSTEMACCOUNTNO tracks console utility activity such as table rebuild, Diskcopy, or Scandisk.                     |
| Event         | CHAR(12) LATIN<br>NOT CASESPECIFIC<br>NOT NULL       | X(12)              | Returns a description of the type of action, using the following descriptions: Logon, Logoff, Logon failed.  |
| LogicalHostId | SMALLINT NOT NULL                                    | -(5)9              | Returns a unique identifier of the logon source for the logged query. A value of zero indicates an internal session.   |
| IFPNo         | SMALLINT NOT NULL                                    | -(5)9              | Returns the vproc number of the PE through which the session was connected or assigned.  |
| SessionNo     | INTEGER NOT NULL                                     | --,---,---,<br>--9 | Returns the session identifier assigned to the session by the TDP or LAN interface.  |
| LogonDate     | DATE NOT NULL  | YY/MM/DD           | Returns the date that the session for which the log entry was made was logged on to the Teradata Database.   |
| LogonTime     | FLOAT NOT NULL                                       | 99:99:99.99        | Returns the time on which logon for the session occurred (useful on logoff events).  |
| LogonSource   | VARCHAR(128)<br>UNICODE NOT CASESPECIFIC             | X(128)             | Channel-Attached Systems Using the CLIV2 API Returns the origin of the CLIV2 mainframe session being reported, such as the user ID or session number of the client system. |

| View Column          | Data Type                                    | Format             | Comment   |
|----------------------|--|--------------------|---|
| ClientIpAddress      | VARCHAR(45)<br>UNICODE NOT<br>CASESPECIFIC   | X(45)              | The ClientIpAddress column is the standard text representation of the IP address where the user accessed the system |
| ClientProgramName    | VARCHAR(1024)<br>UNICODE NOT<br>CASESPECIFIC | X(1024)            | The ClientProgramName is the client system program name   |
| ClientSystemUserId   | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC  | X(128)             | The ClientSystemUserId is the client user id  |
| ClientConnectionType | BYTEINT                                      | -(3)9              | The ClientConnectionType is 1 for TCP/IP or 2 for Channel connect   |
| ClientCoordName      | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC  | X(128)             | The ClientCoordName is the Coordinator name for clients using CICS or IMS   |
| ClientEnvName        | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC  | X(128)             | The ClientEnvName is the TDP Environment Name   |
| ClientJobId          | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC  | X(128)             | The ClientJobId is the TDP Job ID   |
| ClientJobName        | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC  | X(128)             | The ClientJobName is the TDP Job Name   |
| ClientOsName         | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC  | X(128)             | The ClientOsName is the TDP Operating System Name   |
| ClientProcThreadId   | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC  | X(128)             | The ClientProcThreadId is the client process/thread identifier  |
| ClientSecProdGrp     | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC  | X(128)             | The ClientSecProdGrp is the TDP Security Product Group  |
| ClientSecProdUserId  | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC  | X(128)             | The ClientSecProdUserId is TDP Security Product Userid  |
| ClientTcpPortNumber  | INTEGER                                      | --,---,---,<br>--9 | The ClientTcpPortNumber is the TCP port number on the client system   |
| ClientTdHostName     | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC  | X(128)             | The ClientTdHostName is the Teradata Database hostname that the client used to connect to the Teradata Database     |

| View Column                    | Data Type                                   | Format | Comment   |
|--------------------------------|---|--------|---|
| ClientTerminalId               | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128) | The ClientTerminalId is TDP<br>Terminal Identifier                                    |
| ClientTransactionId            | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128) | The ClientTransactionId is TDP<br>Transaction Identifier                              |
| ClientUserOperId               | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128) | The ClientUserOperId is the TDP<br>User/Operator Identifier                           |
| ClientVmName                   | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128) | The ClientVmName is the TDP<br>Virtual Machine Name                                   |
| ClientVmUserId                 | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128) | The ClientVmUserId is the TDP<br>Virtual Machine User Id                              |
| MechanismName                  | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128) | The MechanismName is<br>authentication method used to<br>authenticate the user        |
| ClientTDPReleaseId             | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128) | The ClientTDPReleaseId is the TDP<br>Release Identifier                               |
| ClientCLlv2ReleaseId           | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128) | The ClientCLlv2ReleaseId is the<br>CLlv2 Release Identifier                           |
| ClientSessionDesc              | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128) | The ClientSessionDesc is the TDP<br>session description                               |
| ClientWorkload                 | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128) | The ClientWorkload is the<br>TDP Workload   |
| ClientJobData                  | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128) | The ClientJobData is the client<br>Job Data from the LSINFO<br>environmental variable |
| ClientODBCDriverVersion        | VARCHAR(16)<br>UNICODE NOT<br>CASESPECIFIC  | X(16)  | The ClientODBCDriverVersion is<br>the client ODBC Driver Version                      |
| ClientNetDataProviderVersion   | VARCHAR(16)<br>UNICODE NOT<br>CASESPECIFIC  | X(16)  | The ClientNetDataProviderVersion<br>is the client .NET Data<br>Provider Version       |
| ClientODBCDriverManagerVersion | CHAR(30)<br>UNICODE NOT<br>CASESPECIFIC     | X(30)  | The<br>ClientODBCDriverManagerVersion<br>is the client ODBC Driver<br>Manager Version |

| View Column                | Data Type                                   | Format             | Comment   |
|----------------------------|---|--------------------|---|
| ClientNetFrameworkVersion  | CHAR(30)<br>UNICODE NOT<br>CASESPECIFIC     | X(30)              | The ClientNetFrameworkVersion is the client .NET Framework Version  |
| ClientAttributesEx         | VARCHAR(512)<br>UNICODE NOT<br>CASESPECIFIC | X(512)             | The ClientAttributesEx field contains extra description of the client that do not match any of the other Client Attributes fields |
| ClientJDBCDriverVersion    | VARCHAR(16)<br>UNICODE NOT<br>CASESPECIFIC  | X(16)              | The ClientJDBCDriverVersion is the client JDBC Driver Version   |
| ClientJavaVersion          | VARCHAR(30)<br>UNICODE NOT<br>CASESPECIFIC  | X(30)              | The ClientJavaVersion is the client Java Framework Version  |
| RecoverableNetworkProtocol | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL      | X(1)               | The RecoverableNetworkProtocol column indicates whether the client supports the RecoverableNetwork client-database interface.     |
| LogonRedrive               | VARCHAR(33)<br>UNICODE NOT<br>CASESPECIFIC  | X(33)              | This field is reserved for future use.  |
| ClientIPAddrByClient       | VARCHAR(45)<br>UNICODE NOT<br>CASESPECIFIC  | X(45)              | Client IP Address by Client   |
| ClientPortByClient         | INTEGER                                     | --,---,---,<br>--9 | Client Port by Client   |
| ServerIPAddrByClient       | VARCHAR(45)<br>UNICODE NOT<br>CASESPECIFIC  | X(45)              | Serve IP Address by Client  |
| ServerPortByClient         | INTEGER                                     | --,---,---,<br>--9 | Server Port by Client   |
| ClientIPAddrByUnity        | VARCHAR(45)<br>UNICODE NOT<br>CASESPECIFIC  | X(45)              | Client IP Address by Unity  |
| ClientPortByUnity          | INTEGER                                     | --,---,---,<br>--9 | Client Port by Unity  |
| UnityClientSideIPAddr      | VARCHAR(45)<br>UNICODE NOT<br>CASESPECIFIC  | X(45)              | Unity Client Side IP Address  |
| UnityClientSidePort        | INTEGER                                     | --,---,---,<br>--9 | Unity Client Side Port  |

| View Column               | Data Type                                    | Format             | Comment                         |
|---------------------------|--|--------------------|---------------------------------|
| UnityServerSideIPAddr     | VARCHAR(45)<br>UNICODE NOT<br>CASESPECIFIC   | X(45)              | Unity Server Side IP Address    |
| UnityServerSidePort       | INTEGER                                      | --,---,---,<br>--9 | Unity Server Side Port          |
| ServerIPAddrByUnity       | VARCHAR(45)<br>UNICODE NOT<br>CASESPECIFIC   | X(45)              | Server IP Address by Unity      |
| ServerPortByUnity         | INTEGER                                      | --,---,---,<br>--9 | Server Port by Unity            |
| ServerIPAddrByServer      | VARCHAR(45)<br>UNICODE NOT<br>CASESPECIFIC   | X(45)              | Server IP Address by Server     |
| ServerPortByServer        | INTEGER                                      | --,---,---,<br>--9 | Server Port by Server           |
| ClientCOPSuffixedHostName | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC  | X(128)             | Client COP Suffixed Hostname    |
| UnitySessNo               | INTEGER                                      | --,---,---,<br>--9 | Unity Session Number            |
| UnityVersion              | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC  | X(128)             | Unity Version                   |
| UnityAuthMechName         | VARCHAR(1571)<br>UNICODE NOT<br>CASESPECIFIC | X(1571)            | Unity Authorized Mechanism Name |
| UnityMechanismName        | VARCHAR(1571)<br>UNICODE NOT<br>CASESPECIFIC | X(1571)            | Unity Mechanism Name            |
| UserAuthenticatedBy       | VARCHAR(1)<br>UNICODE NOT<br>CASESPECIFIC    | X(1)               | User Authenticated by           |
| ClientTDSessionPoolName   | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC  | X(128)             | Client TD Session Pool Name     |
| UnityTcpPortNumber        | INTEGER                                      | --,---,---,<br>--9 | Unity TCP Port Number           |
| UnityIpAddress            | VARCHAR(45)<br>UNICODE NOT<br>CASESPECIFIC   | X(45)              | Unity IP Address                |

| View Column                    | Data Type                                   | Format | Comment  |
|--------------------------------|---|--------|--|
| SecurityPolicy                 | VARCHAR(256)<br>UNICODE NOT<br>CASESPECIFIC | X(256) | Security Policy  |
| UnitySecurityPolicy            | VARCHAR(256)<br>UNICODE NOT<br>CASESPECIFIC | X(256) | Unity Security Policy  |
| Unity_AuthUser                 | VARCHAR(256)<br>UNICODE NOT<br>CASESPECIFIC | X(256) | Unity AuthUser   |
| DirUserNetConfidentiality      | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC        | X(1)   | This column contains the user's network confidentiality policy that applies on the connection between the client and the gateway or between the client and Unity, obtained from the LDAP directory used for security policy.                             |
| DirUserNetPolicyLevel          | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC        | X(1)   | When DirUserNetConfidentiality is "I" or "C", this column contains the level of protection required, obtained by lookup in the LDAP directory used for security policy.  |
| UnityNetConfidentiality        | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC        | X(1)   | This column contains the user's network confidentiality policy that applies on the connection between Unity and the gateway, obtained from the LDAP directory used for security policy.  |
| UnityNetPolicyLevel            | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC        | X(1)   | When UnityNetConfidentiality is "I" or "C", this column contains the level of protection required, obtained by lookup in the LDAP directory used for security policy.  |
| EffectiveSessionNetConf        | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC        | X(1)   | This column contains the effective network confidentiality policy for the session, that applies on the connection between the client and the gateway or between the client and Unity, integrated from all sources of security policy that do not change. |
| EffectiveSessionNetPolicyLevel | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC        | X(1)   | When EffectiveSessionNetConfidentiality is "I" or "C", this column contains the level of protection required, integrated from all sources of security policy that do not change within a session.  |

| View Column           | Data Type                             | Format | Comment   |
|-----------------------|---------------------------------------|--------|---|
| ProxyLogon            | CHAR(1) LATIN NOT CASESPECIFIC        | X(1)   | The ProxyLogon column indicates T if logon request uses Passwordless Proxy for TD2. |
| ErrorCode             | SMALLINT                              | -(5)9  | Identifies the Error Code returned by Logon Failure parcels.                        |
| ErrorMsg              | VARCHAR(256) UNICODE NOT CASESPECIFIC | X(256) | Identifies the Error message returned by Logon Failure parcels.                     |
| ClientConfType        | CHAR(1) LATINC                        | X(1)   | The ClientConfType column indicates Client Confidentiality Type.                    |
| ServerConfType        | CHAR(1) LATINC                        | X(1)   | The ServerConfType column indicates Server Confidentiality Type.                    |
| UnityConfType         | CHAR(1) LATINC                        | X(1)   | The UnityConfType column indicates Unity Confidentiality Type.                      |
| ServerUnityConfType   | CHAR(1) LATINC                        | X(1)   | The ServerUnityConfType column indicates Server Unity Confidentiality Type.         |
| ClientConfVersion     | VARCHAR(16) LATIN NOT CASESPECIFIC    | X(16)  | The ClientConfVersion column indicates Client TLS version number.                   |
| ClientConfCipherSuite | VARCHAR(64) LATIN NOT CASESPECIFIC    | X(64)  | The ClientConfCipherSuite column indicates Client TLS cipher suite.                 |
| UnityConfVersion      | VARCHAR(16) LATIN NOT CASESPECIFIC    | X(16)  | The UnityConfVersion column indicates Unity TLS version number.                     |
| UnityConfCipherSuite  | VARCHAR(64) LATIN NOT CASESPECIFIC    | X(64)  | The UnityConfCipherSuite column indicates Unity TLS cipher suite.                   |

## Usage Notes

### LogonSource

Teradata recommends using alternative columns instead of the LogonSource column, if available. For information about the recommended columns for LogonSource, see "LogonSource Column Fields and Examples."

### ClientConfCipherSuite and UnityConfCipherSuite

ASCII String representing the confidentiality cipher suite. Currently, this is the TLS cipher suite, for example: "TLS\_AES\_256\_GCM\_SHA384".

### ClientConfVersion and UnityConfVersion

ASCII String representing the confidentiality version number. Currently, this is the TLS version number, for example: "TLS 1.2".

### Unity\_AuthUser

Provides audit trail information for the security policy.

### UserName

The UserName column returns "Non-existent User" when a user tries to log on with a bad username.

### Possible Values for ClientConfType

| Value | Description  |
|-------|--|
| C     | TLS used for encryption. Client validated the Certificate-Authority chain but ignored the Subject-Alternative-Name and the Common-Name.  |
| E     | TDGSS used for encryption. The application does not have the option to change this during the session.   |
| F     | TLS was attempted but failed, so this is a fallback to using TDGSS for encryption because ENCRYPTDATA is specified.  |
| H     | TLS was attempted but failed, so this is a fallback to unencrypted because ENCRYPTDATA is not specified.   |
| O     | May be encrypted using TDGSS or unencrypted. The application has the option of changing this at any time. This is possible only with CLlV2 apps, because drivers (e.g. ODBC, JDBC) do not toggle encryption on and off within a session. |
| R     | TLS used for encryption. Server certificate was ignored; client did not validate the identity of the server.   |
| U     | Unencrypted. The application does not have the option to change this during the session.   |
| V     | TLS used for encryption. Client validated the Certificate-Authority chain and the Subject-Alternative-Name or the Common-Name.   |

### Possible Values for ClientConnectionType

| Value | Description   |
|-------|---|
| 1     | Client is connected using TCP/IP via the gateway.                   |
| 2     | Client is connected from a mainframe via a mainframe-attached host. |

## Possible Values for DirUserNetConfidentiality, UnityNetConfidentiality, EffectiveSessionNetConf

When these columns are set to I or C it indicates the level of protection required.

| Value | Description   |
|-------|---|
| I     | Indicates the level of protection required for Integrity, which is obtained by lookup in the LDAP directory used for security policy. The levels are: Default (D), Low (L), Medium (M), and High (H).       |
| C     | Indicates the level of protection required for Confidentiality, which is obtained by lookup in the LDAP directory used for security policy. The levels are: Default (D), Low (L), Medium (M), and High (H). |

## Possible Values for Event

### Note:

The names of the following values are truncated if they are more than 12 characters in length. If they are less than 12 characters in length, blank spaces are added.

- Logon
- Logoff
- Logon failed

This value means a logon failed for reasons other than “Bad User,” “Bad Account,” “Bad Password,” “Bad profile,” “IP restrict,” “Auth failed,” “Bad auth,” or “Secur policy.” Currently, those other reasons are a failure to conform to Logon Rules or an attempt to log on without a password, where the TDP logon exit does not approve the logon.

- Bad user
- Bad account

This value means the user provided an account string during logon time, but that string does not match any of the account names specified for the user in the SQL CREATE USER or MODIFY USER statement.

- Bad password
- Bad profile

The external profile associated with the logon request does not exist. External profiles are stored in the directory server.

- Forced off

This value indicates that the user session was terminated from the system console or the PM/API.

- IP restrict

This value indicates the user is not permitted to log on from the IP address used.

- Auth failed

This value indicates an authentication error.

- Bad auth

This value may indicate other authentication errors: bad authentication field, deprecated logons, decryption failure, and so forth.

- Secur policy

This value indicates that a logon attempt failed because of a security policy violation.

### Possible Values for LogonRedrive

| Value                         | Description  |
|-------------------------------|--|
| ' '                           | Not participating  |
| MEMORY NON-FALLBACK RESPONSES | Memory-based Redrive participation   |
| NULL or blanks                | Session is not participating in Redrive and database restarts will not be transparent to applications and users. |

### Possible Values for ProxyLogon

| Value | Description  |
|-------|--|
| T     | The ProxyLogon column indicates if Unity has logged a user onto a TD2 session using existing credentials, when that user was successfully logged on by another Unity-managed Vantage system using TD2. When this occurs, ProxyLogon is set to T. |
| F     | False indicates the TD2 session logged on or attempted to log on with a valid password.  |

### Possible Values for RecoverableNetworkProtocol

| Value | Description |
|-------|-------------|
| T     | True        |
| F     | False       |

### Possible Values for SecurityPolicy and UnitySecurityPolicy

- No Policy
- Plaintext
- Integrity, Default
- Integrity, Low
- Integrity, Medium
- Integrity, High
- Confidentiality, Default

- Confidentiality, Low
- Confidentiality, Medium
- Confidentiality, High

SecurityPolicy and UnitySecurityPolicy are used to provide audit trail information for the security policy.

### Possible Values for ServerConfType

| Value | Description   |
|-------|---|
| E     | TDGSS used for encryption, either enforced by policy or asserted by a Client.                             |
| O     | May be encrypted using TDGSS or unencrypted, as asserted by a Client, or because it cannot be determined. |
| T     | TLS used for encryption.  |
| U     | Unencrypted, as asserted by a Client Interface.   |

### Possible Values for ServerUnityConfType

| Value | Description   |
|-------|---|
| E     | TDGSS used for encryption, either enforced by policy or asserted by Unity.        |
| O     | May be encrypted using TDGSS or unencrypted. Cannot be determined by the gateway. |
| T     | TLS used for encryption.  |
| U     | Unencrypted, as asserted by Unity.  |

### Possible Values for UnityConfType

| Value | Description   |
|-------|---|
| E     | TDGSS used for encryption.  |
| F     | TLS was attempted but the handshake failed, so this is an attempt to fallback to using TDGSS for encryption. This is otherwise equivalent to "E". |
| T     | TLS used for encryption.  |
| U     | Unencrypted.  |

## Example: Using LogOnOffV

The following SELECT statement displays information about the type of security policy enforced by the Gateway:

```
SELECT Logdate,Logtime,Event,SessionNo,SecurityPolicy,
       UnitySecurityPolicy, Unity_AuthUser
FROM LogonOffV WHERE UnitySecurityPolicy is not NULL
ORDER BY Logdate,LogTime;
```

Result:

```
LogDate 12/11/08
LogTime 13:55:47.86
Event Bad User
SessionNo      1,001
SecurityPolicy No Policy
UnitySecurityPolicy No Policy
Unity_AuthUser CN=proxyconfhigh,DC=jrrlinux
LogDate 12/11/08
LogTime 14:12:26.11
Event Bad User
SessionNo      1,002
SecurityPolicy No Policy
UnitySecurityPolicy No Policy
Unity_AuthUser CN=proxyconfhigh,DC=jrrlinux
```

## Related Topics

| For more information about ...   | See ...   |
|--|---|
| how to control access, space, and ownership                                  | <i>Teradata Vantage™ - Database Design</i> , B035-1094.                             |
| the security policy (either enforced by Gateway or Unity) and the Unity user | <i>Teradata Vantage™ - Advanced SQL Engine Security Administration</i> , B035-1100. |

## LogonRulesV

**Category:** Security

**Database:** DBC

| View Column | Data Type   | Format | Comment   |
|-------------|---|--------|---|
| UserName    | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | The LogonRulesV.UserName field shows the name of the user to whom the logon rule applies. May be 'Default' to indicate all users. |

| View Column     | Data Type   | Format              | Comment  |
|-----------------|---|---------------------|--|
| LogicalHostID   | SMALLINT NOT NULL                                       | ZZZ9                | Returns a unique identifier of the logon source for the logged query. A value of zero indicates an internal session.                 |
| LogonStatus     | CHAR(1) LATIN<br>NOT CASESPECIFIC<br>NOT NULL           | X(1)                | Returns if the rule grants (G) or refuses (R) permission for the named user to log on from the identified client system.             |
| NullPassword    | CHAR(1) LATIN<br>NOT CASESPECIFIC<br>NOT NULL           | X(1)                | Returns a code that specifies whether the rule allows this user to log on without a password from the specified client system.       |
| CreatorName     | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)              | Table or database creator. For DBC. AllRights, the grantor of explicit rights, otherwise the user who executed the CREATE statement. |
| CreateTimeStamp | TIMESTAMP(0)  | YYYY-MM-DDBHH:MI:SS | Returns the date and time that the object in the row was created.  |

## Usage Notes

The LogonRules view retrieves information about logon rules generated as a result of successfully processed GRANT LOGON/REVOKE LOGON statements.

The LOGON rules can be used to redefine the SQL Data Control Language defaults. For more information, see *Teradata Vantage™ SQL Data Control Language*, B035-1149.

The initial default is that all users can log on from all connected client systems.

## MapGrantsV[X]

**Category:** Security

**Database:** DBC

| View Column    | Data Type  | Format | Comment   |
|----------------|--|--------|---|
| UserOrRoleName | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128) | Returns UserName or RoleName of the grantee.  |
| MapName        | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128) | Returns the MapName of the map granted.   |
| GrantOption    | CHAR(1) LATIN NOT<br>CASESPECIFIC NOT NULL           | X(1)   | Returns Y if the recipient can grant the map to another user or role, N otherwise. This is always N for a role. |

| View Column     | Data Type                                      | Format              | Comment   |
|-----------------|--|---------------------|---|
| GranteeKind     | CHAR(1) LATIN NOT CASESPECIFIC NOT NULL        | X(1)                | Returns U for user and R for role.                        |
| CreateTimeStamp | TIMESTAMP(0)                                   | YYYY-MM-DDBHH:MI:SS | Returns the the date and time the map was granted.        |
| GrantorName     | VARCHAR(128) UNICODE NOT CASESPECIFIC NOT NULL | X(128)              | Returns the name of the user or role who granted the map. |

## Usage Notes

The MapGrantsV[X] view stores information about each map granted to the user and the user's role. MapGrantsV shows all map grants and MapGrantsVX shows only map grants to which the current user or role has access.

### Possible Values for GranteeKind

| Value | Description |
|-------|-------------|
| U     | User.       |
| R     | Role.       |

### Possible Values for GrantOption

| Value | Description  |
|-------|--|
| Y     | The recipient can grant the map to another user or role.                                 |
| R     | The recipient cannot grant the map to another user or role. This is always N for a role. |

## MapListsV[X]

**Category:** TDMaps

**Database:** TDMaps

| View Column | Data Type                                      | Format | Comment       |
|-------------|--|--------|---------------|
| ZoneName    | VARCHAR(128) UNICODE NOT CASESPECIFIC          | X(128) | Zone name     |
| MapListName | VARCHAR(128) UNICODE NOT CASESPECIFIC NOT NULL | X(128) | Map list name |

| View Column        | Data Type  | Format                            | Comment   |
|--------------------|--|-----------------------------------|---|
| DictionaryDatabase | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128)                            | Dictionary Database                               |
| CreatedTimeStamp   | TIMESTAMP(6) WITH<br>TIME ZONE                       | YYYY-MM-<br>DDBHH:MI:SS.<br>S(6)Z | Timestamp from when the<br>list was created       |
| LastModified       | TIMESTAMP(6) WITH<br>TIME ZONE                       | YYYY-MM-<br>DDBHH:MI:SS.<br>S(6)Z | Timestamp from when the<br>list was last modified |
| CreateListUser     | BYTE(4)  | X(8)                              | ID of the user that created<br>the list           |
| MapName            | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128)                            | Map name  |

## Usage Notes

The MapListsV[X] views are security zone constrained views that contains map lists and the maps included in each list.

### DictionaryDatabase

DictionaryDatabase is an optional column to define where the data dictionary tables, such as TVM, are stored.

### ZoneName

ZoneName contains the secure zone name associated with the object (the secure zone associated with the DatabaseName or TableName).

## MapsV[X]

**Category:** Schema

**Database:** DBC

| View Column | Data Type   | Format  | Comment                                    |
|-------------|---|---------|--|
| MapName     | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)  | Returns name of the map.                   |
| MapSlot     | SMALLINT NOT NULL                                       | ---,--9 | Returns the physical number of<br>the map. |

| View Column        | Data Type   | Format              | Comment   |
|--------------------|---|---------------------|---|
| MapKind            | CHAR(1) LATIN<br>NOT CASESPECIFIC<br>NOT NULL           | X(1)                | Returns MapKind C for a contiguous map S for a sparse map.  |
| SystemDefined      | CHAR(1) LATIN<br>NOT CASESPECIFIC<br>NOT NULL           | X(1)                | Returns Y if the map is created during a system initialization (sysinit) or reconfiguration. It is N for a map created by CREATE MAP. |
| SystemDefault      | CHAR(1) LATIN<br>NOT CASESPECIFIC<br>NOT NULL           | X(1)                | Returns Y if the map is the system-default map, N otherwise.  |
| ParentMapName      | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)              | Returns name of the parent contiguous map that a sparse map is based on. It is null for a contiguous map.                             |
| NumberOfAMPs       | SMALLINT NOT NULL                                       | ---,--9             | Returns the number of AMPs in a contiguous or sparse map.   |
| ContiguousStartAmp | SMALLINT NOT NULL                                       | ---,--9             | Returns the lowest AMP number in a contiguous map or, for a sparse map, in the parent contiguous map.                                 |
| ContiguousEndAmp   | SMALLINT NOT NULL                                       | ---,--9             | Returns the highest AMP number in a contiguous map or for a sparse map, in the parent contiguous map.                                 |
| CreatorName        | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)              | Returns username of the user that created the map if the map is a sparse map. This is null for a system-defined map.                  |
| ZoneName           | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC             | X(128)              | Returns the secure zone name.   |
| CreateTimeStamp    | TIMESTAMP(0)  | YYYY-MM-DDBHH:MI:SS | Returns the date and time the map was created.  |
| CommentString      | VARCHAR(255)<br>UNICODE<br>NOT CASESPECIFIC             | X(255)              | Returns optional comment on the created map.  |
| FallbackKind       | CHAR(1) LATIN<br>NOT CASESPECIFIC<br>NOT NULL           | X(1)                | Returns D for dedicated fallback or L for legacy fallback.  |
| TotalNumberOfAMPs  | SMALLINT NOT NULL                                       | ---,--9             | Returns the total number of AMPs, both primary and fallback.  |

## Usage Notes

The MapsV[X] view returns map information. The MapsV view returns information for all maps in the system. The MapsVX view only returns maps to which the user has access.

### ContiguousStartAmp and ContiguousEndAmp

ContiguousStartAMP and ContiguousEndAMP are the lowest and highest AMP numbers for a contiguous map. For a sparse map, they are the same as its parent contiguous map's starting and ending AMPs. For the contiguous maps created by a system initialization or upgrade, ContiguousStartAMP is set to zero and ContiguousEndAMP to NumberOfAMPs for the contiguous map minus one. Contiguous maps may have a different range of values.

### MapName, MapSlot

MapName is the name of the map. MapNames start with TD\_, such as TD\_Map1, for contiguous maps. The MapName for sparse maps created by CREATE MAP is not allowed to start with TD\_.

Mapslot is the physical number of the map. Valid values for MapSlot range from 0 to 63.

### Possible Values for FallbackKind

| Value | Description  |
|-------|--|
| D     | Dedicated fallback. With dedicated fallback, one AMP in the cluster contains a backup copy for another AMP's data in the cluster.                          |
| L     | Legacy fallback. With legacy fallback each AMP, in the cluster contains a backup copy of a portion of the primary data for every other AMP in the cluster. |

### Possible Values for MapKind

| Value | Description     |
|-------|-----------------|
| C     | Contiguous map. |
| S     | Sparse map.     |

### Possible Values for SystemDefault

SystemDefault returns Y if the map is the system-default map; N otherwise.

### Possible Values for SystemDefined

| Value | Description   |
|-------|---|
| Y     | The map is created during a system initialization (sysinit), reconfiguration, DIP, or an upgrade to Teradata Database 16.0 or later, or Vantage Advanced SQL Engine 16.20 or later. |
| N     | The map is created by CREATE MAP.   |

## MultiColumnStatsV

**Category:** Optimizer Statistics

**Database:** DBC

| View Column        | Data Type   | Format   | Comment   |
|--------------------|---|----------|---|
| DatabaseName       | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)   | Returns the database name of the table, column, group of columns, index, view, or query on which statistics were collected. |
| TableName          | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)   | Returns the name of the table that contains the column, group of columns, or index on which the statistics were collected.  |
| ColumnName         | VARCHAR(10000)<br>UNICODE NOT<br>CASESPECIFIC           | X(10000) | The ColumnName column identifies a column or columns.   |
| StatsName          | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC             | X(128)   | The StatsName column contains the alias name associated with the statistic  |
| StatsSource        | CHAR(1)<br>LATIN UPPERCASE                              | X(1)     | The StatsSource column records the method this statistic is acquired.   |
| ValidStats         | CHAR(1)<br>LATIN UPPERCASE                              | X(1)     | The ValidStats column records Database version statistics collected on.   |
| DBSVersion         | VARCHAR(32)<br>LATIN UPPERCASE                          | X(32)    | Returns the version of the database that contains the objects on which the statistics were collected.                       |
| IndexNumber        | SMALLINT  | ---,--9  | The IndexNumber column is the Index number of the index on which statistics are collected.                                  |
| SampleSignature    | VARCHAR(256)<br>LATIN NOT<br>CASESPECIFIC               | X(256)   | The SampleSignature column is the Sample options encoded as a 10 character signature.                                       |
| SampleSizePct      | DECIMAL(5,2)  | ----.99  | The SampleSizePct column is the Sample size percent used when collecting statistics.  |
| ThresholdSignature | VARCHAR(512)<br>LATIN NOT<br>CASESPECIFIC               | X(512)   | The ThresholdSignature column is the Threshold options encoded as a 17 character signature.                                 |
| MaxIntervals       | SMALLINT  | ---,--9  | The MaxIntervals column is the User-specified maximum number of intervals.  |

| View Column           | Data Type    | Format                  | Comment   |
|-----------------------|--------------|-------------------------|---|
| MaxValueLength        | INTEGER      | --,---,---,--9          | The MaxValueLength column is the User-specified maximum value length.                                       |
| RowCount              | FLOAT        | ---,---,---,---,<br>--9 | The RowCount column is the row count of the table, view or query on which statistics are collected.         |
| UniqueValueCount      | FLOAT        | ---,---,---,---,<br>--9 | The UniqueValueCount column is the Number of unique values of the ExpressionList.                           |
| PNullUniqueValueCount | FLOAT        | ---,---,---,---,<br>--9 | The PNullUniqueValueCount column is the Number of unique values from rows with partial nulls.               |
| NullCount             | FLOAT        | ---,---,---,---,<br>--9 | The NullCount column is the Number of nulls of the ExpressionList.  |
| AllNullCount          | FLOAT        | ---,---,---,---,<br>--9 | The AllNullCount column is the Number of all nulls of the ExpressionList.                                   |
| HighModeFreq          | FLOAT        | ---,---,---,---,<br>--9 | The HighModeFreq column is the highest frequency of values of the ExpressionList.                           |
| PNullHighModeFreq     | FLOAT        | ---,---,---,---,<br>--9 | The PNullHighModeFreq column is the highest frequency of values having partial nulls of the ExpressionList. |
| CreateTimeStamp       | TIMESTAMP(0) | YYYY-MM-DDBHH:MI:SS     | The CreateTimeStamp column is the statistics creation time stamp.   |
| LastCollectTimeStamp  | TIMESTAMP(0) | YYYY-MM-DDBHH:MI:SS     | The LastCollectTimeStamp column is the Last statistics collection time stamp.                               |
| LastAlterTimeStamp    | TIMESTAMP(0) | YYYY-MM-DDBHH:MI:SS     | The LastAlterTimeStamp column is the last user updated time stamp.  |

## Usage Notes

### Referenced Columns

Many of the Data Dictionary view columns have referenced table columns. That is, the value in the view column corresponds to a value in the selected column referenced in the table. It would be meaningful to join the view and the referenced table based on the selected column and the referenced column.

Referenced columns for this view are:

| View Column | Referenced Column       |
|-------------|-------------------------|
| IndexNumber | DBC.Indexes.IndexNumber |

### SampleSizePct

If sampling is not used, the SampleSizePct column is set to 0 or 100.

### StatsSource

The StatsSource column records the method by which this statistic is acquired. For information about the possible values for the StatsSource column, see "StatsSource Column."

## To Get Information Not Contained in This View

This view does not contain:

- Statistics on tables protected by row-level security. To get these statistics, create views on DBC.StatsTbl. You can view details about this table in Teradata SQL Assistant or Teradata Studio Express.
- Information about column attributes. To get this information, join the DBC.StatsTbl table to the DBC.TVFields table. You can view details about this table in Teradata SQL Assistant or Teradata Studio Express.

## Example: Using MultiColumnStatsV

This example assumes the following statistics have been collected:

```
STATISTICS
  COLUMN (o_orderkey, o_orderdatetime)
  ON Orders;
```

This query can be used to retrieve the statistics:

```
SELECT * FROM dbc.MultiColumnStatsV
  WHERE databasename = 'sales'
     AND tablename = 'orders';
```

## Related Topics

| For more information about statistics collected on ...  | See ...                          |
|---|----------------------------------|
| non-indexed columns and single-column indexes           | <a href="#">ColumnsV[X]</a> .    |
| indexes for which two or more columns have been defined | <a href="#">IndexStatsV[X]</a> . |

| For more information about statistics collected on ... | See ...   |
|--|---|
| tables   | <a href="#">StatsV</a> or <a href="#">TableStatsV</a> . |
| materialized temporary tables                          | <a href="#">TempTableStatsV</a> .                       |
| single expressions                                     | <a href="#">ExpStatsV</a> .                             |
| multiple expressions                                   | <a href="#">MultiExpStatsV</a> .                        |

## MultiExpStatsV

**Category:** Optimizer Statistics

**Database:** DBC

| View Column     | Data Type   | Format   | Comment   |
|-----------------|---|----------|---|
| DatabaseName    | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)   | The DatabaseName column is the name of the database in which the table resides.                       |
| TableName       | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)   | The TableName column is the name of the containing table.   |
| ColumnName      | VARCHAR(10000)<br>UNICODE NOT<br>CASESPECIFIC           | X(10000) | The ColumnName column identifies a column or columns.   |
| StatsName       | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC             | X(128)   | The StatsName column contains the alias name associated with the statistic                            |
| StatsSource     | CHAR(1)<br>LATIN UPPERCASE                              | X(1)     | The StatsSource column records the method this statistic is acquired.                                 |
| ValidStats      | CHAR(1)<br>LATIN UPPERCASE                              | X(1)     | The ValidStats column indicates whether the statistics are valid or not.                              |
| DBSVersion      | VARCHAR(32)<br>LATIN UPPERCASE                          | X(32)    | Returns the version of the database that contains the objects on which the statistics were collected. |
| IndexNumber     | SMALLINT  | ---,--9  | The IndexNumber column is the Index number of the index on which statistics are collected.            |
| SampleSignature | VARCHAR(256)<br>LATIN NOT<br>CASESPECIFIC               | X(256)   | The SampleSignature column is the Sample options encoded as a 10 character signature.                 |

| View Column           | Data Type                                 | Format                   | Comment   |
|-----------------------|---|--------------------------|---|
| SampleSizePct         | DECIMAL(5,2)                              | ----.99                  | The SampleSizePct column is the Sample size percent used when collecting statistics.                        |
| ThresholdSignature    | VARCHAR(512)<br>LATIN NOT<br>CASESPECIFIC | X(512)                   | The ThresholdSignature column is the Threshold options encoded as a 17 character signature.                 |
| MaxIntervals          | SMALLINT                                  | ---,--9                  | The MaxIntervals column is the User-specified maximum number of intervals.                                  |
| MaxValueLength        | INTEGER                                   | --,---,---,--9           | The MaxValueLength column is the User-specified maximum value length.                                       |
| RowCount              | FLOAT                                     | ----,---,---,---,<br>--9 | The RowCount column is the row count of the table, view or query on which statistics are collected.         |
| UniqueValueCount      | FLOAT                                     | ----,---,---,---,<br>--9 | The UniqueValueCount column is the Number of unique values of the ExpressionList.                           |
| PNullUniqueValueCount | FLOAT                                     | ----,---,---,---,<br>--9 | The PNullUniqueValueCount column is the Number of unique values from rows with partial nulls.               |
| NullCount             | FLOAT                                     | ----,---,---,---,<br>--9 | The NullCount column is the Number of nulls of the ExpressionList.  |
| AllNullCount          | FLOAT                                     | ----,---,---,---,<br>--9 | The AllNullCount column is the Number of all nulls of the ExpressionList.                                   |
| HighModeFreq          | FLOAT                                     | ----,---,---,---,<br>--9 | The HighModeFreq column is the highest frequency of values of the ExpressionList.                           |
| PNullHighModeFreq     | FLOAT                                     | ----,---,---,---,<br>--9 | The PNullHighModeFreq column is the highest frequency of values having partial nulls of the ExpressionList. |
| CreateTimeStamp       | TIMESTAMP(0)                              | YYYY-MM-DDBHH:MI:SS      | The CreateTimeStamp column is the statistics creation time stamp.   |
| LastCollectTimeStamp  | TIMESTAMP(0)                              | YYYY-MM-DDBHH:MI:SS      | The LastCollectTimeStamp column is the Last statistics collection time stamp.                               |
| LastAlterTimeStamp    | TIMESTAMP(0)                              | YYYY-MM-DDBHH:MI:SS      | The LastAlterTimeStamp column is the last user updated time stamp.  |

## Usage Notes

### Referenced Columns

Many of the Data Dictionary view columns have referenced table columns. That is, the value in the view column corresponds to a value in the selected column referenced in the table. It would be meaningful to join the view and the referenced table based on the selected column and the referenced column.

Referenced columns for this view are:

| View Column | Referenced Column       |
|-------------|-------------------------|
| IndexNumber | DBC.Indexes.IndexNumber |

### SampleSizePct

If sampling is not used, the SampleSizePct column is set to 0 or 100.

### StatsSource

The StatsSource column records the method by which this statistic is acquired. For information about the possible values for the StatsSource column, see "StatsSource Column."

## To Get Information Not Contained in This View

This view does not contain:

- Statistics on tables protected by row-level security. To get these statistics, create views on DBC.StatsTbl. You can view details about this table in Teradata SQL Assistant or Teradata Studio Express.
- Information about column attributes. To get this information, join the DBC.StatsTbl table to the DBC.TVFields table. You can view details about this table in Teradata SQL Assistant or Teradata Studio Express.

## Example: Using MultiExpStatsV

This example assumes the following statistics have been collected:

```
STATISTICS
  COLUMN (o_orderno, CAST(o_orderdatetime AS DATE)) AS Stats_OrderDate
  ON Orders;
```

This query can be used to retrieve statistics collected on multiple columns involving expressions:

```
SELECT * FROM dbc.MultiExpStatsV
  WHERE databasename = 'sales'
  AND tablename = 'orders';
```

## Related Topics

| For more information about statistics collected on ...  | See ...  |
|---|--|
| non-indexed columns and single-column indexes           | <a href="#">ColumnStatsV.</a>                          |
| indexes for which two or more columns have been defined | <a href="#">IndexStatsV[X].</a>                        |
| tables  | <a href="#">StatsV</a> or <a href="#">TableStatsV.</a> |
| materialized temporary tables                           | <a href="#">TempTableStatsV.</a>                       |
| multiple columns  | <a href="#">MultiColumnStatsV.</a>                     |
| single expressions                                      | <a href="#">ExpStatsV.</a>                             |

## ObjectsInSparseMapsV

**Category:** Schema

**Database:** DBC

| View Column         | Data Type  | Format  | Comment  |
|---------------------|--|---------|--|
| DatabaseName        | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128)  | The name of the database where the object is from. |
| ObjectName          | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128)  | The name of the object.                            |
| ObjectKind          | VARCHAR(1) LATIN<br>NOT CASESPECIFIC                 | X(1)    | The objectkind of the object.                      |
| MapName             | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128)  | The map name of the object.                        |
| MapSlot             | SMALLINT NOT NULL                                    | ---,--9 | The map slot of the object.                        |
| NumberOfPrimaryAMPs | SMALLINT NOT NULL                                    | ---,--9 | The number of primary amps used for the object.    |
| ColocationName      | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC             | X(128)  | The colocation of the object.                      |

## Usage Notes

### ColocationName

ColocationName is the colocation name that is used for the table, join index, or hash index having a sparse map. Objects with the same sparse map reside on the same AMPs if they have the same colocation name.

### ObjectKind

The ObjectKind is set to table, join index, or hash index.

## ObjectListsV[X]

**Category:** TDMaps

**Database:** TDMaps

| View Column        | Data Type  | Format                            | Comment   |
|--------------------|--|-----------------------------------|---|
| ZoneName           | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC             | X(128)                            | Zone name   |
| ListName           | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128)                            | Object list name                                  |
| DictionaryDatabase | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128)                            | Dictionary Database                               |
| CreatedTimeStamp   | TIMESTAMP(6) WITH<br>TIME ZONE                       | YYYY-MM-<br>DDBHH:MI:SS.<br>S(6)Z | Timestamp from when<br>the list was created       |
| LastModified       | TIMESTAMP(6) WITH<br>TIME ZONE                       | YYYY-MM-<br>DDBHH:MI:SS.<br>S(6)Z | Timestamp from when<br>the list was last modified |
| CreateListUser     | BYTE(4)  | X(8)                              | ID of the user that created<br>the list           |
| DatabaseName       | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128)                            | Database name                                     |
| ObjectName         | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC             | X(128)                            | Object name                                       |

## Usage Notes

ObjectListsV[X] are security zone constrained views of the object lists and the objects included in them. ObjectListsVX returns only the objects to which the user has access.

**ListName**

ListName contains the user specified name for the list that must be unique in TDMaps.

**DictionaryDatabase**

DictionaryDatabase is an optional column to define where the data dictionary tables, such as TVM, are stored.

**ZoneName**

ZoneName contains the secure zone name associated with the object (the secure zone associated with the DatabaseName or TableName).

**ObjectUseCountV[X]**

**Category:** Accounting

**Database:** DBC

| View Column  | Data Type   | Format                         | Comment  |
|--------------|---|--------------------------------|--|
| DatabaseName | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                         | Returns the Database Name of the object for which access count and/or UDI counts are recorded. |
| ObjectName   | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                         | Returns the Object Name of the object for which access count and/or UDI counts are recorded.   |
| UsageType    | CHAR(3) LATIN<br>UPPERCASE NOT NULL                     | X(3)                           | Returns the usage type of the object. It can be either DML or STA.                             |
| AccessCount  | BIGINT  | --,---,---,---,---,<br>---,--9 | Returns the number of accesses since the last reset by the user.                               |

**Usage Notes****Possible Values for UsageType**

| Name | Description                   |
|------|-------------------------------|
| DML  | Data Manipulation Language    |
| STA  | Optimizer usage of statistics |

## Example: Using ObjectUseCountV

The following SELECT statement shows the number of accesses occurring on a particular object:

```
SELECT AccessCount FROM DBC.ObjectUseCountV WHERE DatabaseName = 'Personnel'
AND ObjectName = 'Employee';
```

The query returns the following result:

```
AccessCount
-----
16
```

## PartitioningConstraintsV[X]

**Category:** Integrity

**Database:** DBC

| View Column    | Data Type   | Format   | Comment  |
|----------------|---|----------|--|
| DatabaseName   | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)   | PartitioningConstraintsVX.<br>DatabaseName is the name<br>of the database in which the table or<br>join index with the partitioning resides.   |
| TableName      | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)   | PartitioningConstraintsVX.TableName<br>is the name of the table or join index<br>with the partitioning.  |
| IndexName      | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC             | X(128)   | PartitioningConstraintsVX.IndexName<br>is the name of the associated<br>index, if any. This field<br>is NULL if there is no associated index<br>or the index does not have a name.   |
| IndexNumber    | SMALLINT  | ---,--9  | PartitioningConstraintsVX.<br>IndexNumber is the internal<br>number assigned to the associated<br>index, if any. This field is 1 for a primary<br>index. This field is 0 if the partitioning<br>is not associated with an index. |
| ConstraintType | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL                  | X(1)     | PartitioningConstraintsVX.<br>ConstraintType is Q<br>(partitioning constraint).  |
| ConstraintText | VARCHAR(16000)<br>UNICODE<br>CASESPECIFIC               | X(16000) | PartitioningConstraintsVX.<br>ConstraintText is the<br>unresolved condition text   |

| View Column          | Data Type   | Format              | Comment  |
|----------------------|---|---------------------|--|
|                      |   |                     | of the partitioning constraint (implied constraint determined from the partitioning levels).   |
| ConstraintCollation  | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL                  | X(1)                | PartitioningConstraintsVX.<br>ConstraintCollation is the collation to be used in evaluating the constraint: E (EBCDIC), A (ASCII), M (MULTINATIONAL), C (CHARSET_COLL), J (JIS_COLL), or U (use the session collation).                                      |
| CollationName        | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC             | X(128)              | PartitioningConstraintsVX.<br>CollationName is the CHARSET_COLL collation name if ConstraintCollation is C; it is NULL if ConstraintCollation is U; otherwise, it is the collation name (EBCDIC, ASCII, MULTINATIONAL, or JIS_COLL).                         |
| CreatorName          | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)              | PartitioningConstraintsVX.<br>CreatorName is the name of the user that created the constraint.   |
| CreateTimeStamp      | TIMESTAMP(0)  | YYYY-MM-DDBHH:MI:SS | PartitioningConstraintsVX.<br>CreateTimeStamp is the date and time the constraint was created.   |
| CharSetID            | BYTEINT   | -(3)9               | PartitioningConstraintsVX.CharSetID is a number identifying the character set of the CHARSET_COLL collation, if this collation is used; otherwise, NULL. If the character set is user-defined, this ID should also exist in DBC.CharTranslations.CharSetID.  |
| SessionMode          | CHAR(1) LATIN<br>UPPERCASE                              | X(1)                | PartitioningConstraintsVX.<br>SessionMode is the name of the session mode used to determine the default case sensitivity in evaluating the index constraint: A (ANSI), T (Teradata). For constraints not involving comparison of character data, it is NULL. |
| ResolvedCurrent_Date | DATE  | YY/MM/DD            | PartitioningConstraintsVX.<br>ResolvedCurrent_Date is the last reconciled date if the object is a table or join index that is defined using DATE or CURRENT_DATE; otherwise, NULL.   |

| View Column                   | Data Type                      | Format                                    | Comment   |
|-------------------------------|--------------------------------|---|---|
| ResolvedCurrent_<br>TimeStamp | TIMESTAMP(6)<br>WITH TIME ZONE | YYYY-<br>MM-<br>DDBHH:<br>MI:SS.<br>S(6)Z | PartitioningConstraintsVX.<br>ResolvedCurrent_TimeStamp is<br>the last reconciled timestamp<br>if the object is a table or join index<br>that is defined using CURRENT_<br>TIMESTAMP; otherwise, NULL.  |
| DefinedCombinedPartitions     | BIGINT NOT NULL                | -(19)9                                    | PartitioningConstraintsVX.<br>DefinedCombinedPartitions indicates<br>the number of currently<br>defined combined partitions. This<br>value can change as long as it does<br>not exceed MaxCombinedPartitions.                                       |
| MaxCombinedPartitions         | BIGINT NOT NULL                | -(19)9                                    | PartitioningConstraintsVX.<br>MaxCombinedPartitions indicates<br>the maximum number<br>of combined partitions<br>allowed. This value is greater than or<br>equal to DefinedCombinedPartitions.<br>This value cannot change<br>for a nonempty table. |
| PartitioningLevels            | SMALLINT<br>NOT NULL           | -(5)9                                     | PartitioningConstraintsVX.<br>PartitioningLevels indicates<br>the number of partitioning levels<br>(a value between 1 and 62, inclusive).   |
| ColumnPartitioningLevel       | SMALLINT<br>NOT NULL           | -(5)9                                     | PartitioningConstraintsVX.<br>ColumnPartitioningLevel indicates<br>the level number for<br>the column partitioning level<br>(a value between 1 and 62, inclusive)<br>or no column partitioning (a value of 0).                                      |

## Usage Notes

The PartitioningConstraintsV[X] system views provide information about partitioning constraints, which are constraints derived from a PARTITION BY clause for a table or join index that has partitioning.

For information about the possible values for the ConstraintType column, see "ConstraintType Column."

## Corresponding Tables

The X view references these additional tables:

- DBC.AccessRights
- DBC.Owners
- DBC.RoleGrants
- DBC.Roles

Use the SHOW TABLE statement, HELP COLUMN statement, and the ColumnsV[X] and DBC.PartitioningConstraintsV[X] views to obtain partitioning information for a table or join index.

The HELP INDEX statement Ordered or Partitioned attribute shows if an index is partitioned. For a NoPI table, the HELP INDEX statement cannot be used to determine whether or not the table or join index is partitioned.

### Possible Values for ColumnPartitioningLevel

| Value                        | Description  |
|------------------------------|--|
| Between 1 and 62 (inclusive) | The level number for the column partitioning level, if ConstraintType is 'Q' |
| 0                            | There is no column partitioning or ConstraintType is not 'Q'                 |

### Possible Values for ConstraintCollation

| Value | Description               |
|-------|---------------------------|
| A     | ASCII                     |
| C     | CHARSET_COLL              |
| E     | EBCDIC                    |
| J     | JIS_COLL                  |
| M     | MULTINATIONAL             |
| U     | Use the session collation |

### Possible Values for DefinedCombinedPartitions

- This is the product of the number of defined partitions for each level used.
- Zero if ConstraintType is not 'Q'
- Cannot exceed MaxCombinedPartitions

If the partitioning is altered when a table is nonempty, this value can change to be smaller or larger provided that it does not exceed MaxCombinedPartitions.

### Possible Values for MaxCombinedPartitions

- Zero if ConstraintType is not 'Q'
- Greater than or equal to the DefinedCombinedPartitions column
- Zero if DefinedCombinedPartitions is zero

---

#### Note:

The MaxCombinedPartitions column value cannot change for a nonempty table.

---

## Possible Values for PartitioningLevels

| Value                        | Description  |
|------------------------------|--|
| Between 1 and 62 (inclusive) | The number of partitioning levels for the table or join index, if ConstraintType is 'Q'. |
| 0                            | The table or join index is not partitioned.  |

## Examples

This section contains examples of the use of this view to retrieve information about objects that have different types of partitioning and different levels of partitioning.

### Example: Retrieving Any Partitioned Objects from PartitioningConstraintsV

The PartitioningConstraintsV view is used to retrieve a list of partitioned objects, regardless of the type of partitioning.

This query could be used to retrieve this information. It does not contain any clauses to specify the type of partitioning.

```
SELECT DatabaseName, TableName (TITLE 'Table/Join Index Name')
FROM DBC.PartitioningConstraintsV
ORDER BY DatabaseName, TableName;
```

### Example: Retrieving Objects with Column Partitioning from PartitioningConstraintsV

The PartitioningConstraintsV view is used to retrieve a list of objects that have column partitioning.

This query is very similar to the query used in Example: Retrieving Any Partitioned Objects from PartitioningConstraintsV, but it contains a WHERE clause that specifies the type of partitioning.

```
SELECT DatabaseName, TableName (TITLE 'Table/Join Index Name')
FROM DBC.PartitioningConstraintsV
WHERE ColumnPartitioningLevel >= 1
ORDER BY DatabaseName, TableName;
```

### Example: Retrieving Objects with 8-byte Multilevel Partitioning and Column Partitioning from PartitioningConstraintsV

The PartitioningConstraintsV view is used to retrieve a list of object that:

- Have 8-byte, multilevel partitioning, and
- One of the levels is column partitioning.

This query could be used to retrieve this information.

```
SELECT DatabaseName, TableName (TITLE 'Table/Join Index Name')
FROM DBC.PartitioningConstraintsV
WHERE MaxCombinedPartitions >= 65536
      AND PartitioningLevels >= 2 AND ColumnPartitioningLevel >= 1
ORDER BY DatabaseName, TableName;
```

### Example: Retrieving Objects with 2-byte Single Level Column Partitioning from PartitioningConstraintsV

The PartitioningConstraintsV view is used to retrieve a list of objects that:

- Have 2-byte, single-level partitioning, and
- The type of partitioning is column partitioning.

This query could be used to retrieve this information.

```
SELECT DatabaseName, TableName (TITLE 'Table/Join Index Name')
FROM DBC.PartitioningConstraintsV
WHERE MaxCombinedPartitions <= 65535
      AND PartitioningLevels = 1 AND ColumnPartitioningLevel = 1
ORDER BY DatabaseName, TableName;
```

## Related Topics

For more information about partitioning levels, see *Teradata Vantage™ - Database Design*, B035-1094 and *Teradata Vantage™ SQL Data Definition Language Syntax and Examples*, B035-1144.

## PeriodsV[X]

**Category:** Schema

**Database:** DBC

| View Column  | Data Type   | Format | Comment   |
|--------------|---|--------|---|
| DatabaseName | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | PeriodsVX.DatabaseName is the name of the database in which the table having period column. |
| Table_Name   | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | PeriodsVX.Table_Name is the name of the table in which period column exists.                |
| Period_Name  | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | PeriodsVX.Period_Name is the name of a period column in a table.                            |

| View Column       | Data Type                                   | Format  | Comment  |
|-------------------|---|---------|--|
| Fid               | SMALLINT NOT NULL                           | ---,--9 | PeriodsVX.Fid is the field id of a period column in a table.   |
| Tid               | BYTE(6) NOT NULL                            | X(12)   | PeriodsVX.Tid is the table id of the table in which period column exists.                                |
| DBid              | BYTE(4) NOT NULL                            | X(8)    | PeriodsVX.DBid is the database id of the database in which the table is having period column.            |
| START_COLUMN_NAME | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC | X(128)  | PeriodsVX.START_COLUMN_NAME is the name of the begin column of a derived period, otherwise returns null. |
| END_COLUMN_NAME   | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC | X(128)  | PeriodsVX.END_COLUMN_NAME is the name of the end column of a derived period, otherwise returns null.     |

## Usage Notes

For information about non temporal period columns that are derived from the physical DateTime columns that store the beginning and ending bound values of the derived periods, see *Teradata Vantage™ ANSI Temporal Table Support*, B035-1186 .

### START\_COLUMN\_NAME

The START\_COLUMN\_NAME column is the name of the column that stores the start column name of the valid-time derived period column, system-time derived period column, or nontemporal derived period column.

### END\_COLUMN\_NAME

The END\_COLUMN\_NAME is the name of the column that stores the end column name of the valid-time derived period column, system-time derived period column, or nontemporal derived period column.

## Example: Using PeriodsVX

Suppose you have the following table definition:

```
CREATE SET TABLE TEST.employee ,NO FALLBACK ,
  NO BEFORE JOURNAL,
  NO AFTER JOURNAL,
  CHECKSUM = DEFAULT,
  DEFAULT MERGEBLOCKRATIO
(
  eid INTEGER,
  jobdurstart DATE FORMAT 'YY/MM/DD' NOT NULL,
```

```

jobbdurend DATE FORMAT 'YY/MM/DD' NOT NULL,
PERIOD FOR jobdur (jobdurstart, jobbdurend),
ename CHAR(100) CHARACTER SET LATIN NOT CASESPECIFIC)
PRIMARY INDEX ( eid );

```

The following statement entered in BTEQ retrieves rows from the PeriodsV[X] view for the employee table created:

```

BTEQ -- Enter your SQL request or BTEQ command:
.SET FOLDLINE ON
.SET SIDETITLES ON
SELECT * FROM DBC.PeriodsVX WHERE table_Name = 'employee';
*** Query completed. One row found. 8 columns returned.
*** Total elapsed time was 2 seconds.
    DatabaseName test
      Table_Name employee
    Period_Name jobdur
        Fid  1,028
        Tid 00008D090000
        DBid 0000FA03
START_COLUMN_NAME jobdurstart
END_COLUMN_NAME  jobbdurend

```

## ProfileAsgdSecConstraintsV[X]

**Category:** Integrity

**Database:** DBC

| View Column    | Data Type  | Format | Comment   |
|----------------|--|--------|---|
| ProfileName    | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128) | Returns name of the profile assigned to the current user.   |
| ConstraintName | VARCHAR(128) UNICODE<br>UPPERCASE NOT NULL           | X(128) | Returns the name of the table level check. This is null if unnamed.   |
| ValueName      | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128) | Returns the name of the security constraint value that was assigned to a profile under current user.                              |
| IsDefault      | CHAR(1) LATIN<br>UPPERCASE NOT NULL                  | X(1)   | Returns whether the security constraint value assigned to the profile is the default value for the constraint under current user. |
| Assignor       | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128) | The name of the user that assigned the security constraint value to profile.  |

## Usage Notes

### Possible Values for IsDefault

| Value | Description     |
|-------|-----------------|
| Y     | Default         |
| N     | Not the default |

## ProfileInfoV[X]

**Category:** Security

**Database:** DBC

| View Column     | Data Type   | Format                         | Comment  |
|-----------------|---|--------------------------------|--|
| ProfileName     | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                         | Returns the name of the profile.   |
| DefaultAccount  | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC             | X(128)                         | Returns the name of the default account, if any, for the user.   |
| DefaultDB       | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC             | X(128)                         | Returns the name of the default database.  |
| SpoolSpace      | BIGINT  | --,---,---,---,<br>---,---,--9 | Returns an integer indicating the maximum spool space allowed for the database. SpoolSpace is 0 if DatabaseName is PUBLIC.             |
| TempSpace       | BIGINT  | --,---,---,---,<br>---,---,--9 | Returns the maximum temporary space allocated for a database, profile, or user in bytes.   |
| ExpirePassword  | SMALLINT  | ---,--9                        | Returns the number of days to elapse before the password expires. 0 indicates the password does not expire.                            |
| PasswordMinChar | BYTEINT   | -(3)9                          | Returns the minimum number of characters in a valid password string.   |
| PasswordMaxChar | BYTEINT   | -(3)9                          | Returns the maximum number of characters in a valid password string. PasswordMaxChar must be equal to or greater than PasswordMinChar. |

| View Column           | Data Type                                    | Format              | Comment  |
|-----------------------|--|---------------------|--|
| PasswordDigits        | CHAR(1)<br>LATIN UPPERCASE                   | X(1)                | Digits are allowed in the password string: Y/y (yes), N/n (no), R/r (at least one digit required).   |
| PasswordSpecChar      | CHAR(1)<br>LATIN UPPERCASE                   | X(1)                | Returns a code to indicate which special characters are to be allowed in the password. For the definition of the codes, see DBC.SysSecDefaults.  |
| PasswordRestrictWords | CHAR(1)<br>LATIN UPPERCASE                   | X(1)                | Indicates whether to restrict certain words from being contained within a password string. These are the valid values: Y (Yes), N (No). The default is N.  |
| MaxLogonAttempts      | BYTEINT                                      | -(3)9               | Number of erroneous logons before the user is locked. 0 (user never locked).   |
| LockedUserExpire      | SMALLINT                                     | ---,--9             | Returns the number of minutes to elapse before a locked user is unlocked. 0 indicates immediate unlock. -1 indicates user is locked indefinitely.  |
| PasswordReuse         | SMALLINT                                     | ---,--9             | Returns the number of days to elapse before a password can be reused. 0 indicates immediate reuse.   |
| CommentString         | VARCHAR(255)<br>UNICODE NOT<br>CASESPECIFIC  | X(255)              | Returns user-supplied text or commentary on the column, database, table, view, macro, user-defined function, user-defined types, user-defined methods, stored procedure, role, profile, or user. |
| CreatorName           | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC  | X(128)              | Profile creator.   |
| CreateTimeStamp       | TIMESTAMP(0)<br>NOT NULL                     | YYYY-MM-DDBHH:MI:SS | Returns the date and time that the object in the row was created.  |
| LastAlterName         | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC  | X(128)              | Returns the name of the user who last updated the dictionary row.  |
| LastAlterTimeStamp    | TIMESTAMP(0)<br>NOT NULL                     | YYYY-MM-DDBHH:MI:SS | Returns the time the dictionary row was last updated.  |
| Queryband             | VARCHAR(4096)<br>UNICODE NOT<br>CASESPECIFIC | X(4096)             | The query band set at logon. NULL indicates that there is no profile query band.   |
| QuerybandDefault      | CHAR(1)<br>LATIN UPPERCASE                   | X(1)                | A D indicates the names in the profile query band are default values. Default values can be set with different values  |

| View Column           | Data Type                                    | Format  | Comment   |
|-----------------------|--|---------|---|
|                       |  |         | in the session and transaction query bands. An R indicates the names are restricted and cannot be set to different values.              |
| IgnoreQuerybandValues | VARCHAR(4096)<br>UNICODE NOT<br>CASESPECIFIC | X(4096) | The ignore query band values. These name-value pairs will be discarded if found in a SET QUERY_BAND statement.                          |
| DefaultMapName        | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC  | X(128)  | Returns the default mapname for the profile. NULL indicates the profile does not have a default map.                                    |
| MapOverride           | VARCHAR(1) LATIN<br>NOT<br>CASESPECIFIC      | X(1)    | MapOverride indicates no override or override on error for the default map. "N" indicates no override. "E" indicates OVERRIDE ON ERROR. |

## Usage Notes

In ProfileInfoV[X], if no profile is set up for the user or for ProfileInfo, and if no profile is defined, the view contains the following message:

```
No rows found
```

If the profile creator has been dropped, the ProfileInfoV[X] view contains the information of that profile, which is shown with the text "Dropped User" for CreatorName or LastAlterName.

## Example: Using ProfileInfoV

The following query lists the parameter settings of all profiles in the system together with their parameter settings:

```
SELECT CAST(ProfileName AS CHAR(15)),
       CAST (DefaultDB AS CHAR(15)),
       CAST (DefaultAccount AS CHAR(15)),
       SpoolSpace,
       TempSpace
FROM DBC.ProfileInfoV ORDER BY 1;
*** Query completed. 3 rows found. 5 columns returned.
```

Result:

| ProfileName | DefaultDB | DefaultAccount | SpoolSpace | TempSpace |
|-------------|-----------|----------------|------------|-----------|
| kanji       | japan     | i18n           | 50000      | 50000     |
| HResources  | personnel | ?              | ?          | ?         |
| NTGroup     | NT        | ncrsandiego    | 50000      | 50000     |

If multiple accounts are specified for the ACCOUNT parameters, only the first account (the default account) in the list is reflected in the display. The remaining accounts in the list can be retrieved by performing a SELECT on the existing view DBC.AccountInfoV. Parameter settings for the profile assigned to the user may similarly be displayed by performing a SELECT on the system view DBC.ProfilesInfoVX.

## QryLockLogXMLV

**Category:** Query

**Database:** DBC

| View Column      | Data Type  | Format                   | Comment   |
|------------------|--|--------------------------|---|
| ProcID           | DECIMAL(5,0)<br>NOT NULL                                 | -(5)9                    | Unique processor ID of the AMP vproc that processed the request.  |
| CollectTimeStamp | TIMESTAMP(6)<br>NOT NULL                                 | YYYY-MM-DDBHH:MI:SS.S(6) | Date and time when the lock contention is recorded. This is different from the time when the contention first occurs. A field inside XMLTextInfo represents the start time of the contention. |
| QueryID          | DECIMAL(18,0)<br>NOT NULL                                | --Z(17)9                 | (Internally generated identifier of the query and the foreign key to other DBQL tables.   |
| XMLTextInfo      | CLOB(1048576)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(32000)                 | Returns the XML text describing the lock contention of each AMP involved in lock.   |

## Usage Notes

The QryLockLogXMLV view of DBQLXMLLOCKTbl is populated if the WITH LOCK option is requested in addition to the default information row. Each row stores a subset of the XML document recording lock delays.

Access to this view is granted based on the security policy of your site. The data from the QryLockLogXMLV view comes directly from the DBQLXMLLockTbl table.

The DBQLXMLLockTbl table logs lock contentions in XML format. The lock logger displays lock objects as follows.

| LockObjectRequested | Description                |
|---------------------|----------------------------|
| D                   | Database                   |
| T                   | Table                      |
| R                   | Row hash                   |
| TP                  | Table Partition range      |
| RP                  | RowHash in Partition range |
| RK                  | RowHash in one partition   |
| RN                  | RowKey range               |

## Shredding the XMLTextInfo Column

The XMLTextInfo column stores the XML data that needs to be shredded.

The XML shredding feature is enabled by default in Teradata Vantage. Using Teradata client software, such as BTEQ or Teradata Viewpoint, you can call the SP\_LockLog\_Shredder stored procedure to shred the XML data in the DBQLXMLLockTbl table.

### Note:

You cannot shred the contents of the XMLTextInfo column in QryLockLogXMLV. To shred this column, you must use the underlying table, DBQLXMLLockTbl.

For information about shredding the XMLTextInfo column, see *Teradata Vantage™ - Database Administration*, B035-1093.

## Example: Using QryLockLogXMLV

The following statement retrieves all database lock contentions in XML format from the DBC.QryXMLLockTbl table:

```
sel xmltextinfo from DBC.QryLockLogXMLV;
```

The LockObjectRequested entries show various lock objects.

## Related Topics

| For more information about ...             | You can use ...  |
|--|--|
| the DBQLXMLLockTbl table                   | Teradata Studio or Teradata Studio Express to view the DBQLXMLLockTbl table. |
| monitoring and displaying lock information | the Lock Viewer portlet in Teradata Viewpoint.                               |

## QryLogAmpDataV

**Category:** Query

**Database:** DBC

| View Column | Data Type                       | Format                   | Comment   |
|-------------|---------------------------------|--------------------------|---|
| LogTime     | TIMESTAMP(6)                    | YYYY-MM-DDBHH:MI:SS.S(6) | Time when this row data was captured by this row specific AMP.                        |
| QueryID     | DECIMAL(18,0)                   | --Z(17)9                 | (Foreign Key) System wide unique value to join DBQL tables.                           |
| AmpNum      | INTEGER                         | -(10)9                   | Number of the AMP vproc that collected this data row.                                 |
| StepClass   | CHAR(10) LATIN NOT CASESPECIFIC | X(10)                    | Step Class in text form.  |
| StepKind    | CHAR(15) LATIN NOT CASESPECIFIC | X(15)                    | Step Kind in text form.   |
| StepLvl1    | INTEGER                         | -(10)9                   | Step number Level 1.  |
| StepLvl2    | INTEGER                         | -(10)9                   | Step number Level 2, if this column has non-zero value, then this is a parallel step. |
| CumCPU      | FLOAT                           | ----,---,---,---,--9.999 | Cumulative CPU time for the specific session in seconds.                              |
| CumIO       | FLOAT                           | ----,---,---,---,--9     | Cumulative IO count for the specific session.   |
| CPUTime     | FLOAT                           | ----,---,---,---,--9.999 | CPU time in seconds for this step execution in this AMP vproc.                        |
| IOCount     | FLOAT                           | ----,---,---,---,--9     | IO count for this step execution in this AMP vproc.                                   |
| IOKB        | FLOAT                           | ----,---,---,---,--9.999 | IO in KB for this step execution in this AMP vproc.                                   |
| PhysIOCount | FLOAT                           | ----,---,---,---,--9     | Physical IO count for this step execution in this AMP vproc.                          |
| PhysIOKB    | FLOAT                           | ----,---,---,---,--9.999 | Physical IO in KB for this step execution in this AMP vproc.                          |
| VHIOCount   | FLOAT                           | ----,---,---,---,--9     | Very Hot IO count for this step execution in this AMP vproc.                          |

| View Column        | Data Type | Format                         | Comment   |
|--------------------|-----------|--------------------------------|---|
| VHIOKB             | FLOAT     | ---,---,---,---,<br>--9.999    | Very Hot IO in KB for this step execution in this AMP vproc.          |
| VHPhysIOCount      | FLOAT     | ---,---,---,---,--9            | Very Hot Physical IO count for this step execution in this AMP vproc. |
| VHPhysIOKB         | FLOAT     | ---,---,---,---,<br>--9.999    | Very Hot Physical IO in KB for this step execution in this AMP vproc. |
| PSFwddid           | INTEGER   | -(10)9                         | PSF wddid for this step in this AMP vproc.                            |
| TDWMwddid          | INTEGER   | -(10)9                         | TDWM wddid for this step in this AMP vproc.                           |
| CPUDecayLevel      | BYTEINT   | -(3)9                          | CPU Decay Level for this step execution in this AMP vproc.            |
| IODecayLevel       | BYTEINT   | -(3)9                          | IO Decay Level for this step execution in this AMP vproc.             |
| CPUException       | BYTEINT   | -(3)9                          | CPU Exception for this step execution in this AMP vproc.              |
| IOException        | BYTEINT   | -(3)9                          | IO Exception for this step execution in this AMP vproc.               |
| UsedIOTA           | FLOAT     | ---,---,---,---,<br>--9.999    | Used IO Token Allocation for this step execution in this AMP vproc.   |
| CollectAlg         | SMALLINT  | ---,--9                        | DBQL data collection algorithm used for this collection.              |
| UDFCpuTime         | FLOAT     | ---,---,---,---,<br>--9.999    | UDF CPU Usage for this step execution in this AMP vproc.              |
| UDFMemUsage        | FLOAT     | ---,---,---,---,<br>--9.999    | UDF Memory Usage for this step execution in this AMP vproc.           |
| UDFVMPeak          | FLOAT     | ---,---,---,---,<br>--9.999    | UDF Virtual Memory Peak usage for this step in this AMP vproc.        |
| UDFVMData          | FLOAT     | ---,---,---,---,<br>--9.999    | UDF Virtual Memory Data usage for this step in this AMP vproc.        |
| NosRecordsReturned | BIGINT    | --,---,---,---,---,<br>---,--9 | NOS Records Returned for this step in this AMP vproc.                 |
| NosRecordsSkipped  | BIGINT    | --,---,---,---,---,<br>---,--9 | NOS Records skipped for this step in this AMP vproc.                  |
| NosPhysReadIO      | BIGINT    | --,---,---,---,---,<br>---,--9 | NOS Physical read IO count for this step in this AMP vproc.           |
| NosPhysReadIOKB    | FLOAT     | ---,---,---,---,<br>--9.999    | NOS Physical read IO in KB for this step in this AMP vproc.           |

| View Column          | Data Type | Format                         | Comment  |
|----------------------|-----------|--------------------------------|--|
| NosRecordsReturnedKB | FLOAT     | ---,---,---,---,<br>--9.999    | NOS Records Returned in KB for this step in this AMP vproc.  |
| NosTotalIOWaitTime   | FLOAT     | ---,---,---,---,<br>--9.999    | NOS Total IO Wait time for this step in this AMP vproc.      |
| NosMaxIOWaitTime     | FLOAT     | ---,---,---,---,<br>--9.999    | NOS Maximum IO Wait time for this step in this AMP vproc.    |
| NosCPUTime           | FLOAT     | ---,---,---,---,<br>--9.999    | NOS CPU time for this step in this AMP vproc.                |
| NosFiles             | INTEGER   | --,---,---,--9                 | NOS Files count for this step in this AMP vproc.             |
| NosFilesSkipped      | INTEGER   | --,---,---,--9                 | NOS files skipped count for this step in this AMP vproc.     |
| NosPhysWriteIO       | BIGINT    | --,---,---,---,---,<br>---,--9 | NOS Physical write IO count for this step in this AMP vproc. |
| NosPhysWriteIOKB     | FLOAT     | ---,---,---,---,<br>--9.999    | NOS Physical write IO in KB for this step in this AMP vproc. |
| NosFilesWritten      | INTEGER   | --,---,---,--9                 | NOS files written count for this step in this AMP vproc.     |

## QryLogClientAttrV

**Category:** Query

**Database:** DBC

| View Column | Data Type   | Format          | Comment  |
|-------------|---|-----------------|--|
| QueryID     | DECIMAL(18,0)<br>NOT NULL                               | --Z(17)9        | (Foreign Key) Systemwide unique value to join DBQL tables. |
| DateFld     | DATE NOT NULL   | YY/MM/DD        | Date when the event took place.                            |
| TimeFld     | FLOAT<br>NOT NULL                                       | 99:99:<br>99.99 | Time when the event took place.                            |
| UserName    | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)          | Username associated with the event.                        |
| AccountName | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)          | Expanded account in effect when a request was submitted.   |

| View Column          | Data Type   | Format             | Comment   |
|----------------------|---|--------------------|---|
| Event                | CHAR(12) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(12)              | A description of the type of action.  |
| LogicalHostId        | SMALLINT<br>NOT NULL                              | -(5)9              | Unique identifier of the logon source for the logged query. A value of zero indicates an internal session.  |
| IFPNo                | SMALLINT<br>NOT NULL                              | -(5)9              | Vproc number of the PE through which the session was connected or assigned.   |
| SessionNo            | INTEGER<br>NOT NULL                               | --,---,---,<br>--9 | Returns the session identifier assigned to the session by the TDP or LAN interface.   |
| LogonDate            | DATE NOT NULL                                     | YY/MM<br>/DD       | The date that the session for which the log entry was made was logged on to the Teradata Database   |
| LogonTime            | FLOAT<br>NOT NULL                                 | 99:99:<br>99.99    | The time that the session for which the log entry was made was logged on to the Teradata Database.  |
| LogonSource          | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC       | X(128)             | Channel-Attached Systems Using the CLlv2 API Returns the origin of the CLlv2 mainframe session being reported,such as the user ID or session number of the client system. |
| ClientConnectionType | BYTEINT   | -(3)9              | Communication type, with the unsigned integer 2 indicating a channel connection.  |
| ClientCoordName      | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC       | X(128)             | Optional coordinator name as an EBCDIC value.   |
| ClientEnvName        | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC       | X(128)             | Environment, as an EBCDIC value of NULLBATCHNULL, NULLTSONULL, NULLCICSNULL, or NULLIMSNNULL.   |
| ClientJobId          | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC       | X(128)             | Job id as an EBCDIC value.  |
| ClientJobName        | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC       | X(128)             | Jobname as an EBCDIC value.   |
| ClientOsName         | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC       | X(128)             | Canonical operating system designation, with the EBCDIC value NULLMVSNULL.  |

| View Column          | Data Type                                   | Format             | Comment   |
|----------------------|---|--------------------|---|
| ClientProcThreadId   | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128)             | processor thread id of client.  |
| ClientSecProdGrp     | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128)             | Client's security product group.  |
| ClientSecProdUserId  | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128)             | Client's security product user id.  |
| ClientTcpPortNumber  | INTEGER                                     | --,---,---,<br>--9 | Client's TCP/IP port number.  |
| ClientTdHostName     | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128)             | TDPid as an EBCDIC value.   |
| ClientTerminalId     | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128)             | Terminal identifier as an EBCDIC value, if the environment is NULLCICSNULL or NULLIMSNUL.             |
| ClientTransactionId  | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128)             | Transaction identifier as an EBCDIC value, if the environment is NULLCICSNULL or NULLIMSNUL.          |
| ClientUserOperId     | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128)             | the user or operator identifier as an EBCDIC value, if the environment is NULLCICSNULL or NULLIMSNUL. |
| ClientVmName         | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128)             | Client's virtual machine's name.  |
| ClientVmUserId       | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128)             | Client's virtual machine's user id.   |
| MechanismName        | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128)             | Identifies the authentication mechanism used for connections to the data source.                      |
| ClientTDPReleaseId   | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128)             | TDP release as an EBCDIC value.   |
| ClientCLlv2ReleaseId | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128)             | CLlv2 release as an EBCDIC value.   |
| ClientSessionDesc    | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128)             | Optional CLlv2 application Session-desc specification as an EBCDIC value.                             |

| View Column        | Data Type                                    | Format  | Comment   |
|--------------------|--|---------|---|
| ClientWorkload     | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC  | X(128)  | Optional CLIV2 application Workload specification as an EBCDIC value.   |
| ClientJobData      | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC  | X(128)  | Client Job data.  |
| ClientIpAddress    | VARCHAR(45)<br>UNICODE NOT<br>CASESPECIFIC   | X(45)   | ClientIpAddress column is the standard text representation of the IP address where the user accessed the system.  |
| ClientProgramName  | VARCHAR(1024)<br>UNICODE NOT<br>CASESPECIFIC | X(1024) | Application program name as an EBCDIC value.  |
| ClientSystemUserId | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC  | X(128)  | Client's system user id.  |
| AuthMethod         | BYTEINT                                      | -(3)9   | Authorization method used by client to authorize logon.   |
| AuthUser           | VARCHAR(256)<br>UNICODE<br>UPPERCASE         | X(256)  | The userNULLs LDAP URL, when one of the following is true: LDAP authentication is used or The user is a KRB5 or SPNEGO user that has been authorized using a directory service. In all other cases, the column stores the userNULLs login name. |
| MechanismOid       | VARBYTE(32)                                  | X(64)   | Mechanism Object ID.  |
| RFU1               | VARCHAR(256)<br>UNICODE NOT<br>CASESPECIFIC  | X(256)  | Security Policy.  |
| RFU2               | VARCHAR(256)<br>UNICODE NOT<br>CASESPECIFIC  | X(256)  | Unity Security Policy.  |
| RFU3               | VARCHAR(256)<br>UNICODE NOT<br>CASESPECIFIC  | X(256)  | Unity Auth User.  |
| RFU4               | VARCHAR(256)<br>UNICODE NOT<br>CASESPECIFIC  | X(256)  | Unity Proxy Logon.  |
| RFU5               | VARCHAR(256)<br>UNICODE NOT<br>CASESPECIFIC  | X(256)  | This field is reserved for future use.  |

| View Column                | Data Type                                   | Format             | Comment   |
|----------------------------|---|--------------------|---|
| ErrorMsg                   | VARCHAR(256)<br>UNICODE NOT<br>CASESPECIFIC | X(256)             | Identifies the Error message returned by Logon Failure parcel.  |
| ClientInterfaceKind        | VARCHAR(1)<br>UNICODE NOT<br>CASESPECIFIC   | X(1)               | ClientInterfaceKind column stores the type of interface.  |
| ClientInterfaceVersion     | VARCHAR(16)<br>UNICODE NOT<br>CASESPECIFIC  | X(16)              | Version of the interface used by the client application to establish the session to Teradata Database.                  |
| ClientAttributesEx         | VARCHAR(512)<br>UNICODE NOT<br>CASESPECIFIC | X(512)             | Version of the client ODBC Driver Manager used by the client application to establish the session to Teradata Database. |
| RecoverableNetworkProtocol | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL      | X(1)               | Indicates whether the client supports the RecoverableNetwork client-database interface.                                 |
| LogonRedrive               | CHAR(2) LATIN<br>UPPERCASE<br>NOT NULL      | X(2)               | Indicates the sessions participation in Redrive.  |
| ClientIPAddrByClient       | VARCHAR(45)<br>UNICODE NOT<br>CASESPECIFIC  | X(45)              | Client IP Address by Client.  |
| ClientPortByClient         | INTEGER                                     | --,---,---,<br>--9 | Client Port by Client.  |
| ServerIPAddrByClient       | VARCHAR(45)<br>UNICODE NOT<br>CASESPECIFIC  | X(45)              | Serve IP Address by Client.   |
| ServerPortByClient         | INTEGER                                     | --,---,---,<br>--9 | Server Port by Client.  |
| ClientIPAddrByUnity        | VARCHAR(45)<br>UNICODE NOT<br>CASESPECIFIC  | X(45)              | Client IP Address by Unity.   |
| ClientPortByUnity          | INTEGER                                     | --,---,---,<br>--9 | Client Port by Unity.   |
| UnityClientSideIPAddr      | VARCHAR(45)<br>UNICODE NOT<br>CASESPECIFIC  | X(45)              | Unity Client Side IP Address.   |
| UnityClientSidePort        | INTEGER                                     | --,---,---,<br>--9 | Unity Client Side Port.   |
| UnityServerSideIPAddr      | VARCHAR(45)<br>UNICODE NOT<br>CASESPECIFIC  | X(45)              | Unity Server Side IP Address.   |

| View Column               | Data Type                                    | Format             | Comment                                |
|---------------------------|--|--------------------|--|
| UnityServerSidePort       | INTEGER                                      | --,---,---,<br>--9 | Unity Server Side Port.                |
| ServerIPAddrByUnity       | VARCHAR(45)<br>UNICODE NOT<br>CASESPECIFIC   | X(45)              | Serve IP Address by Unity.             |
| ServerPortByUnity         | INTEGER                                      | --,---,---,<br>--9 | Server Port by Unity.                  |
| ServerIPAddrByServer      | VARCHAR(45)<br>UNICODE NOT<br>CASESPECIFIC   | X(45)              | Serve IP Address by Server.            |
| ServerPortByServer        | INTEGER                                      | --,---,---,<br>--9 | Server Port by Server.                 |
| ClientCOPSuffixedHostName | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC  | X(128)             | This field is reserved for future use. |
| UnitySessNo               | INTEGER                                      | --,---,---,<br>--9 | Unity Session Number.                  |
| UnityVersion              | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC  | X(128)             | Unity Version.                         |
| UnityAuthMechName         | VARCHAR(1571)<br>UNICODE NOT<br>CASESPECIFIC | X(1571)            | Unity Authorized Mechanism Name.       |
| UnityMechanismName        | VARCHAR(1571)<br>UNICODE NOT<br>CASESPECIFIC | X(1571)            | Unity Mechanism Name.                  |
| UserAuthenticatedBy       | VARCHAR(1)<br>UNICODE NOT<br>CASESPECIFIC    | X(1)               | This field is reserved for future use. |
| ClientTDSessionPoolName   | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC  | X(128)             | Client TD Session Pool Name.           |
| UnityAuthUser             | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC  | X(128)             | Reserved for future use.               |
| UnityTcpPortNumber        | INTEGER                                      | --,---,---,<br>--9 | Reserved for future use.               |
| UnityIpAddress            | VARCHAR(45)<br>UNICODE NOT<br>CASESPECIFIC   | X(45)              | Reserved for future use.               |
| NegotiatingMechanismOid   | VARBYTE(32)                                  | X(64)              | Reserved for future use.               |

| View Column                  | Data Type                                   | Format | Comment  |
|------------------------------|---|--------|--|
| NegotiatingMechanismName     | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128) | Reserved for future use.   |
| DirUserNetCompression        | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC        | X(1)   | Reserved for future use.   |
| DirUserNetConfidentiality    | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC        | X(1)   | Contains the user's network confidentiality policy that applies on the connection between the client and the gateway or between the client and Unity, obtained from the LDAP directory used for security policy. |
| DirUserNetPolicyLevel        | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC        | X(1)   | When DirUserNetConfidentiality is "I" or "C", this column contains the level of protection required, obtained by lookup in the LDAP directory used for security policy.  |
| UnityNetCompression          | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC        | X(1)   | Reserved for future use.   |
| UnityNetConfidentiality      | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC        | X(1)   | Contains the user's network confidentiality policy that applies on the connection between Unity and the gateway, obtained from the LDAP directory used for security policy.                                      |
| UnityNetPolicyLevel          | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC        | X(1)   | When UnityNetConfidentiality is "I" or "C", this column contains the level of protection required, obtained by lookup in the LDAP directory used for security policy.  |
| DirProfileNetCompression     | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC        | X(1)   | Reserved for future use.   |
| DirProfileNetConfidentiality | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC        | X(1)   | Reserved for future use.   |
| DirProfileNetPolicyLevel     | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC        | X(1)   | Reserved for future use.   |
| DirNetGroupNetCompression    | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC        | X(1)   | Reserved for future use.   |

| View Column                    | Data Type                            | Format | Comment  |
|--------------------------------|--------------------------------------|--------|--|
| DirNetGroupNetConfidentiality  | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC | X(1)   | Reserved for future use.   |
| DirNetGroupNetPolicyLevel      | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC | X(1)   | Reserved for future use.   |
| HostIdNetCompression           | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC | X(1)   | Reserved for future use.   |
| HostIdNetConfidentiality       | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC | X(1)   | Reserved for future use.   |
| HostIdNetPolicyLevel           | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC | X(1)   | Reserved for future use.   |
| UserProfileNetCompression      | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC | X(1)   | Reserved for future use.   |
| UserProfileNetConfidentiality  | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC | X(1)   | Reserved for future use.   |
| UserProfileNetworkConfLevel    | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC | X(1)   | Reserved for future use.   |
| UserNetCompression             | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC | X(1)   | Reserved for future use.   |
| UserNetConfidentiality         | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC | X(1)   | Reserved for future use.   |
| UserNetPolicyLevel             | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC | X(1)   | Reserved for future use.   |
| EffectiveSessionNetCompression | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC | X(1)   | Reserved for future use.   |
| EffectiveSessionNetConf        | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC | X(1)   | Contains the effective network confidentiality policy for the session, that applies on the connection between the client and the gateway or between the client and Unity, integrated from all sources of security policy that do not change. |

| View Column                    | Data Type                                | Format             | Comment   |
|--------------------------------|--|--------------------|---|
| EffectiveSessionNetPolicyLevel | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC     | X(1)               | When EffectiveSessionNetConfidentiality is "I" or "C", this column contains the level of protection required, integrated from all sources of security policy that do not change within a session. |
| Zoneld                         | BYTE(4)<br>NOT NULL                      | X(8)               | ID of the zone, the user belongs to.  |
| ErrorCode                      | SMALLINT                                 | -(5)9              | Identifies the Error Code returned by Logon Failure parcel.   |
| ProxyLogon                     | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC     | X(1)               | The ProxyLogon column indicates T if logon request uses Passwordless Proxy for TD2.   |
| ClientReconnectTimeout         | INTEGER                                  | --,---,---,<br>--9 | Reserved for future use.  |
| GtwVproc                       | SMALLINT                                 | ---,--9            | Reserved for future use.  |
| ClientConfType                 | CHAR(1) LATINC                           | X(1)               | The ClientConfType column indicates Client Confidentiality Type.  |
| ServerConfType                 | CHAR(1) LATINC                           | X(1)               | The ServerConfType column indicates Server Confidentiality Type.  |
| UnityConfType                  | CHAR(1) LATINC                           | X(1)               | The UnityConfType column indicates Unity Confidentiality Type.  |
| ServerUnityConfType            | CHAR(1) LATINC                           | X(1)               | The ServerUnityConfType column indicates Server Unity Confidentiality Type.   |
| ClientConfVersion              | VARCHAR(16)<br>LATIN NOT<br>CASESPECIFIC | X(16)              | The ClientConfVersion column indicates Client TLS version number.   |
| ClientConfCipherSuite          | VARCHAR(64)<br>LATIN NOT<br>CASESPECIFIC | X(64)              | The ClientConfCipherSuite column indicates Client TLS cipher suite.   |
| UnityConfVersion               | VARCHAR(16)<br>LATIN NOT<br>CASESPECIFIC | X(16)              | The UnityConfVersion column indicates Unity TLS version number.   |
| UnityConfCipherSuite           | VARCHAR(64)<br>LATIN NOT<br>CASESPECIFIC | X(64)              | The UnityConfCipherSuite column indicates Unity TLS cipher suite.   |

## Usage Notes

DBC.QryLogClientAttrV is used to obtain client attributes for each request. Previously DBC.DBQLogTbl had limited client attributes. DBC.QryLogClientAttrV represents a one-to-many relation between DBC.EventLog and DBC.DBQLogTbl for the queries logged under DBQL default logging. It is recommended to use this view to get an accurate and rich set of client attributes for each request.

### ClientConfCipherSuite and UnityConfCipherSuite

ASCII String representing the confidentiality cipher suite. Currently, this is the TLS cipher suite, for example: "TLS\_AES\_256\_GCM\_SHA384".

### ClientConfVersion and UnityConfVersion

ASCII String representing the confidentiality version number. Currently, this is the TLS version number, for example: "TLS 1.2".

### Possible Values for ClientConfType

| Value | Description   |
|-------|---|
| C     | TLS used for encryption. Client validated the Certificate-Authority chain but ignored the Subject-Alternative-Name and the Common-Name.   |
| E     | TDGSS used for encryption. The application does not have the option to change this during the session.  |
| F     | TLS was attempted but failed, so this is a fallback to using TDGSS for encryption because ENCRYPTDATA is specified.   |
| H     | TLS was attempted but failed, so this is a fallback to unencrypted because ENCRYPTDATA is not specified.  |
| O     | May be encrypted using TDGSS or unencrypted. The application has the option of changing this at any time. This is possible only with CLv2 apps, because drivers (e.g. ODBC, JDBC) do not toggle encryption on and off within a session. |
| R     | TLS used for encryption. Server certificate was ignored; client did not validate the identity of the server.  |
| U     | Unencrypted. The application does not have the option to change this during the session.  |
| V     | TLS used for encryption. Client validated the Certificate-Authority chain and the Subject-Alternative-Name or the Common-Name.  |

### Possible Values for ProxyLogon

| Value | Description  |
|-------|--|
| T     | The ProxyLogon column indicates if Unity has logged a user onto a TD2 session using existing credentials, when that user was successfully logged on by another Unity-managed Vantage system using TD2. When this occurs, ProxyLogon is set to T. |

| Value | Description   |
|-------|---|
| F     | False indicates the TD2 session logged on or attempted to log on with a valid password. |

### Possible Values for ServerConfType

| Value | Description   |
|-------|---|
| E     | TDGSS used for encryption, either enforced by policy or asserted by a Client.                             |
| O     | May be encrypted using TDGSS or unencrypted, as asserted by a Client, or because it cannot be determined. |
| T     | TLS used for encryption.  |
| U     | Unencrypted, as asserted by a Client Interface.   |

### Possible Values for ServerUnityConfType

| Value | Description   |
|-------|---|
| E     | TDGSS used for encryption, either enforced by policy or asserted by Unity.        |
| O     | May be encrypted using TDGSS or unencrypted. Cannot be determined by the gateway. |
| T     | TLS used for encryption.  |
| U     | Unencrypted, as asserted by Unity.  |

### Possible Values for UnityConfType

| Value | Description   |
|-------|---|
| E     | TDGSS used for encryption.  |
| F     | TLS was attempted but the handshake failed, so this is an attempt to fallback to using TDGSS for encryption. This is otherwise equivalent to "E". |
| T     | TLS used for encryption.  |
| U     | Unencrypted.  |

## Examples: Using QryLogClientAttrV

### Example: Count the Number of Rows Stored in QryLogClientAttrV

The following SELECT statement counts all the rows stored in QryLogClientAttrV.

```
SELECT count(*) FROM dbc.qrylogclientattrv;
```

Result:

|          |
|----------|
| Count(*) |
| -----    |
| 3        |

**Example: SELECT All Rows**

The following SELECT statement retrieves all the rows stored in QryLogClientAttrV.

```
SELECT * FROM dbc.qrylogclientattrv;
```

Result:

|       |               |                    |
|-------|---------------|--------------------|
|       | QueryID       | 307181787909344876 |
|       | DateFld       | 16/03/02           |
|       | TimeFld       | 07:26:04.68        |
|       | UserName      | USER1              |
|       | AccountName   | DBC                |
|       | Event         | Logon              |
|       | LogicalHostId | 1                  |
|       | IFPNo         | 30718              |
|       | SessionNo     | 1,465              |
|       | LogonDate     | 16/03/02           |
|       | LogonTime     | 07:26:04.64        |
| [...] |               |                    |
|       | QueryID       | 307181787909344874 |
|       | DateFld       | 16/03/02           |
|       | TimeFld       | 07:25:56.62        |
|       | UserName      | DBC                |
|       | AccountName   | DBC                |
|       | Event         | Logon              |
|       | LogicalHostId | 1                  |
|       | IFPNo         | 30718              |
|       | SessionNo     | 1,464              |
|       | LogonDate     | 16/03/02           |
| [...] |               |                    |
|       | QueryID       | 307181787909344878 |
|       | DateFld       | 16/03/02           |
|       | TimeFld       | 07:26:08.04        |
|       | UserName      | DBC                |
|       | AccountName   | DBC                |
|       | Event         | Logon              |
|       | LogicalHostId | 1                  |

```

      IFPNo  30718
SessionNo           1,466
LogonDate 16/03/02

```

[...]

## QryLogEventHisV

**Category:** Query

**Database:** DBC

| View Column      | Data Type                                   | Format                    | Comment   |
|------------------|---|---------------------------|---|
| ProcID           | DECIMAL(5,0)<br>NOT NULL                    | -(5)9                     | Returns the process ID of the dispatcher.   |
| CollectTimeStamp | TIMESTAMP(6)<br>NOT NULL                    | YYYY-MM-DDBHH:MI:SS.S(6)  | Returns a timestamp unique to each buffer cache.  |
| EntryTS          | TIMESTAMP(6)<br>NOT NULL                    | YYYY-MM-DDBHH:MI:SS.S(F)Z | The timestamp when the event was detected.  |
| EntryKind        | CHAR(10) LATIN<br>NOT CASESPECIFIC          | X(10)                     | Indicates what kind of event-related construct caused something to happen.  |
| EntryID          | INTEGER                                     | --,---,---,--9            | Returns the internal ID of the causing construct.   |
| EntryType        | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC | X(128)                    | For EntryKind of EVENT this represents the Entry Type from Events table   |
| EntryName        | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC | X(128)                    | Returns the name of the causing construct.  |
| EventValue       | FLOAT                                       | ---,---,---,---,--9       | Returns the current (latest) value of the primary event criterion, such as number of AMP Worker Tasks (AWTs).                     |
| LastValue        | FLOAT                                       | ---,---,---,---,--9       | Returns the value collected over the last interval for this event.  |
| Activity         | CHAR(10) LATIN<br>NOT CASESPECIFIC          | X(10)                     | Returns the activity being logged.  |
| ActivityId       | INTEGER                                     | --,---,---,--9            | Returns the internal ID of the construct made active when Activity is EXPRESSION, SYSCON, or OPENV; otherwise, it is set to zero. |

| View Column       | Data Type                                   | Format               | Comment  |
|-------------------|---|----------------------|--|
| ActivityName      | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC | X(128)               | Returns the name of the construct made active when Activity is EXPRESSION, SYSCON, or OPENV; otherwise, the field is null.                   |
| ConfigId          | INTEGER                                     | --,---,---,--9       | The configuration this event was created in.   |
| SeqNo             | SMALLINT                                    | -(5)9                | Returns a sequence number used to help the user understand the sequence of events, expression, syscon, and openv changes that have occurred. |
| Spare1            | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC | X(128)               | Spare char field.  |
| Spare2            | INTEGER                                     | --,---,---,--9       | Spare int field.   |
| Spare3            | FLOAT                                       | ----,---,---,---,--9 | Spare float field.   |
| FlexThrottleID    | DECIMAL(18,0)                               | -(18)9               | A time ordered representation of an invocation of the Flex Expression  |
| FlexThrottleSeqNo | SMALLINT                                    | -(5)9                | An ordering of distinct invocations of a single Flex Expression  |

## Usage Notes

### EntryType, FlexThrottleID, and FlexThrottleSeqNo

The EntryType column represents the event type of the event when EntryKind is EVENT. EntryType is NULL if EntryKind is not EVENT.

The FlexThrottleID field represents an invocation of the Flex Throttle expression. All action rows that are common to this invocation share a common ID. When there are multiple actions performed due to consecutive applications of the Flex Throttle expression, FlexThrottleSeqNo is used to associate the action to the specific expression associated with it. These columns are NULL if the row is not related to Flex Throttles.

### Possible Values for Activity

| Value      | Description   |
|------------|---|
| ACTIVE     | The Event...State is becoming active  |
| INACTIVE   | The Event...State is becoming inactive  |
| EXPRESSION | The Event entry was used to determine that an expression is to be made active |
| SYSCON     | The Expression entry was used to determine that a syscon is to be made active |

| Value | Description   |
|-------|---|
| OPENV | The Expression entry was used to determine that an openv is to be made active |

### Possible Values for EntryKind

- EVENT
- EXPRESSION
- SYSCON
- OPENV
- STATE

### Example: Using DBC.QryLogEventHisV

The following is an example SQL statement that demonstrates how data can be extracted from the QryLogEventHisV view to give an explanation of the expression and events which caused a RED SysCon.

```
WITH RECURSIVE
  CausalAnalysis(EntryTS,
    EntryKind, EntryID, EntryName, Activity, ActivityId) AS
(
  SELECT EntryTS,
    EntryKind, EntryID, EntryName, Activity, ActivityId
  FROM DBC.QryLogEventHisV
  WHERE EntryKind = 'SYSCON'
    AND EntryName = 'RED'
    AND Activity = 'ACTIVE'
  UNION ALL
  SELECT Cause.EntryTS,
    Cause.EntryKind,
    Cause.EntryID,
    Cause.EntryName,
    Cause.Activity,
    Cause.ActivityId
  FROM CausalAnalysis Condition INNER JOIN
    DBC.QryLogEventHisV Cause
  ON Condition.EntryKind = Cause.ActivityId AND
    Condition.EntryID = Cause.ActivityId
)
SELECT *
FROM CausalAnalysis
ORDER BY 1 DESC;
```

## Example: Selecting All Columns from QryLogEventHisV

Select all columns from QryLogEventHisV.

```
select * from dbc.QryLogEventHisV;
```

The query returns the following result:

| ProcID       | CollectTimeStamp           | EntryTS                    | EntryKind         | EntryID      |
|--------------|----------------------------|----------------------------|-------------------|--------------|
| EntryType    | EntryName                  | EventValue                 | LastValue         | Activity     |
| ConfigId     | SeqNo                      | Spare1                     | Spare2            | Spare3       |
|              |                            | FlexThrottleID             | FlexThrottleSeqNo | ActivityName |
| 30718        | 2016-04-25 14:43:13.670000 | 2016-04-25 14:43:10.670000 | EVENT             | 2            |
| FLEX-CPU-PCT | FLEX-CPUT                  | 1                          | 1                 | INACTIVE     |
| 5,000        | 1                          | ?                          | ?                 | ?            |
| 30718        | 2016-04-25 14:38:32.660000 | 2016-04-25 14:38:28.560000 | EXPRESSION        |              |
| 1            | ?                          | FlexUDEExpress             | ?                 | ?            |
| 5,000        | 1                          | ?                          | ?                 | ?            |
| 30718        | 2016-04-25 14:43:38.630000 | 2016-04-25 14:43:35.610000 | EXPRESSION        |              |
| 2            | ?                          | FLEX-EXPR                  | ?                 | ?            |
| 5,000        | 1                          | ?                          | ?                 | ?            |
| 30718        | 2016-04-25 14:41:38.250000 | 2016-04-25 14:41:35.230000 | EXPRESSION        |              |
| 2            | ?                          | FLEX-EXPR                  | ?                 | ?            |
| 5,000        | 1                          | ?                          | ?                 | ?            |
| 30718        | 2016-04-25 14:38:32.660000 | 2016-04-25 14:38:28.560000 | EVENT             |              |
| 1            | ?                          | FlexUDEEvent               | ?                 | ?            |
| 5,000        | ?                          | ?                          | ?                 | ?            |
| 2            | ?                          | ?                          | ?                 | ?            |

## Example: Using QryLogEventHisV to Show Flex Throttle Events

The following is an example SQL statement that demonstrates how data can be extracted from the QryLogEventHisV view to show Flex Throttle events.

```
sel entryts,
    substr(entrykind,1,10) "kind",
    substr(entryname,1,20) "name",
    cast (eventvalue as float format '999.9999') "evt value",
    cast (lastvalue as float format '999.9999') "last value",
    spare2 "spare Int",
    substr(activity,1,10) "activity id",
    substr(activityname,1,20) "act name", seqno, flexthrottleid,
    flexthrottleseqno from dbc.tdwmeventhistory order by entryts, seqno;
```

The query returns the following result:

| name | SeqNo | EntryTS        | kind | name              | evt value | last value | spare Int | activity id | act |
|------|-------|----------------|------|-------------------|-----------|------------|-----------|-------------|-----|
|      |       | FlexThrottleID |      | FlexThrottleSeqNo |           |            |           |             |     |
|      |       |                |      |                   |           |            |           |             |     |

```

2016-04-25 14:37:02.710000 OPENV Always ? ? ?
ACTIVE ? 1 ? ? ?
2016-04-25 14:37:27.850000 EVENT FLEX-CPU 000.1308 000.0828 ?
ACTIVE ? 1 ? ? ?
2016-04-25 14:38:03.020000 EVENT FLEX-AWT 004.0000 004.0000 ?
ACTIVE ? 1 ? ? ?
2016-04-25 14:38:03.030000 EXPRESSION FLEX-EXPR ? ? ?
ACTIVE ? 1 307186992487951003 1 ? ?
2016-04-25 14:38:03.030000 EVENT FLEX-CPU 000.0951 ? ? EXPRESSION FLEX-
EXPR 2 307186992487951003 1 ? ? EXPRESSION FLEX-
2016-04-25 14:38:03.030000 EVENT FLEX-AWT 004.0000 ? ? EXPRESSION FLEX-
EXPR 3 307186992487951003 1 ? ? EXPRESSION FLEX-
2016-04-25 14:38:23.540000 EVENT FlexUDEEvent ? ? ?
ACTIVE ? 1 ? ? ?

```

## QryLogEventsV

**Category:** Query

**Database:** DBC

| View Column      | Data Type                                   | Format                       | Comment  |
|------------------|---|------------------------------|--|
| ProcID           | DECIMAL(5,0)<br>NOT NULL                    | -(5)9                        | Returns the process ID of the dispatcher.  |
| CollectTimeStamp | TIMESTAMP(6)<br>NOT NULL                    | YYYY-MM-DDBHH:MI:SS.<br>S(6) | (Prime Key) Time and date when the TDWM event cache was written.   |
| SessionID        | INTEGER                                     | --,---,---,--9               | Returns the session identifier.  |
| LogicalHostID    | SMALLINT                                    | -(5)9                        | Returns a unique identifier of the logon source for the logged query. A value of zero indicates an internal session. |
| WDID             | INTEGER                                     | --,---,---,--9               | Returns the workload definition under which the query was classified.  |
| OpEnvID          | INTEGER                                     | --,---,---,--9               | Returns the ID of the operational environment currently activated by TDWM.   |
| SysConID         | INTEGER                                     | --,---,---,--9               | Returns the ID of the system condition currently activated by TDWM.  |
| EventTime        | TIMESTAMP(6)<br>NOT NULL                    | YYYY-MM-DDBHH:MI:SS.<br>S(6) | Identifies the time when the event occurred.   |
| EventCode        | INTEGER NOT NULL                            | --,---,---,--9               | Returns a Teradata error code.   |
| EventSubCode     | INTEGER                                     | --,---,---,--9               | This field is reserved for future use. The value will be 0.  |
| EventInfo        | VARCHAR(200)<br>UNICODE NOT<br>CASESPECIFIC | X(200)                       | Returns information about the event that occurred.   |
| FlexThrottleID   | DECIMAL(18,0)                               | -(18)9                       | A time ordered representation of an invocation of the Flex Expression  |

| View Column       | Data Type | Format | Comment   |
|-------------------|-----------|--------|---|
| FlexThrottleSeqNo | SMALLINT  | -(5)9  | An ordering of distinct invocations of a single Flex Expression |

## Usage Notes

These events may include a Teradata dynamic workload management software rule update or errors encountered during a Teradata dynamic workload management software rule activate or deactivate.

### FlexThrottleID and FlexThrottleSeqNo

The FlexThrottleID field represents an invocation of the Flex Throttle expression. All action rows that are common to this invocation share a common ID. When there are multiple actions performed due to consecutive applications of the Flex Throttle expression the FlexThrottleSeqNo field is used to associate the action to the specific expression associated with it. These columns are NULL if the row is not related to Flex Throttles.

## Example: Using QryLogEventsV

The following statement retrieves rows from the QryLogEventsV view:

```
SELECT * from DBC.QryLogEventsV;
```

The query returns the following result:

```

ProcID          30718
CollectTimeStamp 2016-04-25 14:38:37.570000
SessionID       ?
LogicalHostID   ?
WDID            ?
OpEnvID         ?
SysConID        ?
EventTime       2016-04-25 14:38:33.570000
EventCode       9,707
EventSubCode     4
EventInfo Released 1 OF ALLOWABLE 1 WITH 4 FLEX CANDIDATES REMAINING IN
DELAY QUEUE
FlexThrottleID   307186992487951003
FlexThrottleSeqNo 2

```

## Example: Using QryLogEventsV to Query Flex Throttle Columns

The following statement retrieves Flex Throttle related columns from the QryLogEventsV view:

```
SELECT FlexThrottleID, FlexThrottleSeqNo, EventInfo from DBC.QryLogEventsV
order by 1,2;
```

The query returns the following result:

```
*** Query completed. 45 rows found. 3 columns returned.
*** Total elapsed time was 1 second.

FlexThrottleID FlexThrottleSeqNo EventInfo
-----
? ? For Rule Id 42, State 4 info duplicates default and is ignored
? ? For Rule Id 41, OpEnv 4 SLG info duplicates default and
is ignored
? ? For Rule Id 44, OpEnv 4 SLG info duplicates default and
is ignored
? ? For Rule Id 39, OpEnv 4 SLG info duplicates default and
is ignored
? ? For Rule Id 46, OpEnv 4 SLG info duplicates default and
is ignored
? ? For Rule Id 37, State 4 info duplicates default and is ignored
? ? Priority Scheduler settings updated successfully.
? ? Available AWT Event Triggering Algorithm 0 enabled
? ? Distributed rules changes for config 5000,
state=3, openv=3, ...
? ? Avg Int: Event=FLEX-CPU, Threshold/EI =
0.800000, buckets=6, ...
? ? Event Monitoring enabled with interval of 5 seconds.
? ? Flex Throttle feature enabled
? ? Asynchronous Exception Monitoring enabled w/exception interval
of 5s
? ? Ignoring unsolicited ChangeState msg for seq# 1, to same
state 3, ...
? ? For Rule Id 43, OpEnv 4 SLG info duplicates default and
is ignored
? ? Rules change completed - Read=0.32,
WriteGDO=0.04, LastResp=0.01
? ? For Rule Id 45, OpEnv 4 SLG info duplicates default and
is ignored
? ? For Rule Id 42, OpEnv 4 SLG info duplicates default and
is ignored
? ? Asynchronous Exception Monitoring disabled
? ? All static TDWM rules have been deactivated
? ? Error occurred. Rules change rejected due to error 3312 ...
? ? Received valid control request 1 for config 5000, cat1=1 ...
? ? For Rule Id 35, State 4 info duplicates default and is ignored
? ? For Rule Id 36, State 4 info duplicates default and is ignored
? ? For Rule Id 38, State 4 info duplicates default and is ignored
? ? For Rule Id 38, OpEnv 4 SLG info duplicates default and
is ignored
? ? For Rule Id 40, OpEnv 4 SLG info duplicates default and
is ignored
? ? State change processed for config 5000, state 4, openv 4,
seq# 1
? ? State change completed - Read=0.00,
WriteGDO=0.02, LastResp=0.00
? ? ConfigId=5000, StateId= 4
? ? ConfigId=5000, StateId= 4
? ? ConfigId=5000, StateId= 4
? ? ConfigId=5000, StateId= 4
```

```

?      ? ConfigId=5000, StateId= 4
?      ? ConfigId=5000, StateId= 4
?      ? ConfigId=5000, StateId= 4
?      ? Rules change initiated for config id 5000
?      ? Priority Scheduler settings updated successfully.
?      ? Received valid control request 2 for config 5000,
cat1=0 cat2=0 ...
307186992487951003 1 Released 0 OF ALLOWABLE 1 WITH 0 FLEX CANDIDATES REMAIN IN
DELAY QUEUE
307186992487951003 2 Released 1 OF ALLOWABLE 1 WITH 4 FLEX CANDIDATES REMAIN IN
DELAY QUEUE
307186992487951003 3 Released 1 OF ALLOWABLE 1 WITH 3 FLEX CANDIDATES REMAIN IN
DELAY QUEUE
307186992487951003 4 Released 1 OF ALLOWABLE 1 WITH 2 FLEX CANDIDATES REMAIN IN
DELAY QUEUE
307186992487951003 5 Released 1 OF ALLOWABLE 1 WITH 1 FLEX CANDIDATES REMAIN IN
DELAY QUEUE

```

## QryLogExceptionsV

**Category:** Query

**Database:** DBC

| View Column      | Data Type                                   | Format                       | Comment  |
|------------------|---|------------------------------|--|
| ProcID           | DECIMAL(5,0)<br>NOT NULL                    | -(5)9                        | Returns the process ID of the dispatcher.  |
| CollectTimeStamp | TIMESTAMP(6)<br>NOT NULL                    | YYYY-MM-DDBHH:MI:SS.<br>S(6) | (Prime Key) Time and date when the TDWM exception cache was written.   |
| QueryID          | DECIMAL(18,0)<br>NOT NULL                   | --Z(17)9                     | Returns a system-wide unique ID to identify the query.   |
| UserName         | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC | X(128)                       | Name of the user with this exception.  |
| SessionID        | INTEGER                                     | --,---,---,--9               | Returns the session identifier.  |
| RequestNum       | INTEGER                                     | --,---,---,--9               | Client request number for all queries. For statements within stored procedure CALL statements, the request number is the same as the CALL. |
| LogicalHostID    | SMALLINT                                    | -(5)9                        | Returns a unique identifier of the logon source for the logged query. A value of zero indicates an internal session.                       |
| AcctString       | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC | X(128)                       | Returns the user unexpanded logon account string.  |
| WDID             | INTEGER                                     | --,---,---,--9               | Returns the workload definition under which the query was classified.  |

| View Column        | Data Type                                    | Format                   | Comment  |
|--------------------|--|--------------------------|--|
| OpEnvID            | INTEGER                                      | --,---,---,--9           | Returns the ID of the operational environment currently activated by TDWM.   |
| SysConID           | INTEGER                                      | --,---,---,--9           | Returns the ID of the system condition currently activated by TDWM.  |
| ClassificationTime | TIMESTAMP(6)                                 | YYYY-MM-DDBHH:MI:SS.S(6) | Returns the time when the query was classified.  |
| ExceptionTime      | TIMESTAMP(6)<br>NOT NULL                     | YYYY-MM-DDBHH:MI:SS.S(6) | Returns the time when the query encountered an exception.  |
| ExceptionValue     | INTEGER                                      | --,---,---,--9           | Exception type.  |
| ExceptionAction    | CHAR(10) LATIN<br>NOT CASESPECIFIC           | X(10)                    | The action taken by TDWM on this exception.  |
| NewWDID            | INTEGER                                      | --,---,---,--9           | Specifies the workload definition into which the query was moved because of exception handling.  |
| ExceptionCode      | INTEGER NOT NULL                             | --,---,---,--9           | Returns the error code encountered.  |
| ExceptionSubCode   | INTEGER                                      | --,---,---,--9           | .  |
| ErrorText          | VARCHAR(1024)<br>UNICODE<br>NOT CASESPECIFIC | X(1024)                  | Returns the text from the error if ErrorCode is not 0.   |
| ExtraInfo          | VARCHAR(200)<br>UNICODE<br>NOT CASESPECIFIC  | X(200)                   | Used to track queries.   |
| RuleID             | INTEGER                                      | --,---,---,--9           | Returns the rule ID that caused TDWM to reject a query or logon. Note: This field is not populated in all cases.   |
| WarningOnly        | CHAR(1) LATIN<br>NOT CASESPECIFIC            | X(1)                     | Returns warning code T for true if the error was reported while running TDWM in the warning mode, or it is not logged at all. A null is found in the field if WarningOnly is not true. |
| RejectionCat       | SMALLINT                                     | -(5)9                    | Contains the category of TDWM rules that caused this rejection.  |

## Usage Notes

This view contains entries for:

1. Logons rejected due to the Teradata dynamic workload management software throttle limits

2. Queries rejected due to the Teradata dynamic workload management software object access violations
3. Queries rejected due to the Teradata dynamic workload management software throttle limits if abort option is chosen
4. Queries subject to WD exception handling

For more information about the possible values for the ExceptionValue column, see "ExceptionValue Column."

### Possible Values for ExceptionAction

| Value | Description   |
|-------|---|
| A     | Abort. Cannot be combined with other actions.   |
| C     | Change workload definition (WD). NewWlCld contains the new WD.                                    |
| L     | Log.  |
| E     | Execute Program. ExProgram contains the program name.   |
| T     | Alert. ExAlert contains the alert name.   |
| N     | No action. This option cannot be combined with other actions and disables exception detection.    |
| S     | Abort if the statement is a SELECT and no update has been done in the current (user) transaction. |
| Q     | Post to queue table.  |

### Possible Values for RejectionCat

| Value | Description                             |
|-------|---|
| 1     | A rejection due to a TASM Filter rule   |
| 2     | A rejection due to a TASM Throttle rule |
| 3     | A rejection due to a TASM Workload rule |

## Example: Using QryLogExceptionsV

The following SELECT statement retrieves the view for QryLogExceptionsV:

```
SELECT * from DBC.QryLogExceptionsV;
```

The query returns the following result:

```
ProcID 16383
CollectTimeStamp 2004-06-21 16:38:34.21
QueryID 201205134619838011
```

```

      UserName TEST1
      SessionID ?
      RequestNum ?
      LogicalHostID 1
      AcctString DBC
      WDID ?
      OpEnvID ?
      SysConID ?
      ClassificationTime ?
      ExceptionTime 2004-06-21 17:38:34.21
      ExceptionValue ?
      ExceptionAction ?
      NewWDID ?
      ExceptionCode 3,152
      ExceptionSubCode ?
      ErrorText No access allowed, For all requests, For user TEST1,
                Regulation applies from 00:00 to 24:00
      ExtraInfo ?
      RuleID ?
      WarningOnly ?
      RejectionCat ?

```

## QryLogExplainDocV

**Category:** Query

**Database:** DBC

| View Column      | Data Type                                       | Format                       | Comment  |
|------------------|---|------------------------------|--|
| ProcID           | DECIMAL(5,0) NOT NULL                           | -(5)9                        | Returns the process ID of the dispatcher.                          |
| CollectTimeStamp | TIMESTAMP(6) NOT NULL                           | YYYY-MM-DDBHH:MI:SS.<br>S(6) | (Prime Key) Time and date when the DBQL Explain cache was written. |
| QueryID          | DECIMAL(18,0) NOT NULL                          | --Z(17)9                     | Returns a system-wide unique ID to identify the query.             |
| ExplainTextDoc   | CLOB(1048544000)<br>UNICODE<br>NOT CASESPECIFIC | X(32000)                     | Returns a full Explain Text document.                              |

## Usage Notes

QryLogExplainDocV provides a view into DBQLExplainTbl that combines multiple rows of ExplainText with the same QueryID into a single ExplainTextDoc document.

The QryLogExplainDocV view is populated if the WITH EXPLAIN option is requested in addition to the information row.

## Referenced Columns

Many of the Data Dictionary view columns have referenced table columns. That is, the value in the view column corresponds to a value in the selected column referenced in the table. It would be meaningful to join the view and the referenced table based on the selected column and the referenced column.

Referenced columns for this view are:

| View Column | Description   |
|-------------|---|
| QueryID     | QueryID is a system-wide unique field; You can use QueryID to join DBQL tables or the DBC.TdwmExceptionLog table with DBQL tables without needing ProcID as an additional join field. |

## Example: Using QryLogExplainDocV

The following SELECT statement retrieves the ExplainText for a particular QueryID.

```
.export file=explaindoc.txt;
SELECT explaintextdoc FROM QryLogExplainDocV where QueryID=307199040105795996;
.export reset;
```

### Note:

The output from the SELECT statement (the ExplainTextDoc) does not contain any format; it is just one long string.

## Improving Explain Text Readability

To combine multiple ExplainText doc rows with the same query ID into one easily readable document, perform a SELECT request on the QryLogExplainDocV view.

### Example

```
sel queryid,explaintextdoc from qrylogexplaindocv
where queryid=307188741814475132;
```

The result looks like the following partial output:

```
*** Query completed. One row found. 2 columns returned.
*** Total elapsed time was 1 second.
```

QueryID ExplainTextDoc

```
307188741814475132 1) First, we lock Test_DB.b12 for read on a reserved
rowHash to prevent global deadlock. 2) Next, we lock Test_DB.b3 for
read on a reserved rowHash to prevent global deadlock. 3) We lock
Test_DB.b12 for read, and we lock Test_DB.b3 for read. 4) We do an
all-AMPs RETRIEVE step from Test_DB.b3 by way of an all-rows scan with a
condition of ("NOT (Test_DB.b3.c2 IS NULL)") into Spool 2 (all_amps), which is
redistributed by the hash code of (Test_DB.b3.c2) to all AMPs. Then we do a
SORT to order Spool 2 by row hash. The size of Spool 2 is estimated with
low confidence to be 4 rows (68 bytes). The estimated time for this step is
0.03 seconds. 5) We execute the following steps in parallel....
```

#### Note:

Export the result set to a file. On the monitor, some of the lines may truncate.

The memory allocated for the input is 32 MB. To see the memory setting, use the following:

```
cufconfig -o | grep MallocLimit
```

To add more memory, adjust the MallocLimit setting using cufconfig. For more information about cufconfig, see *Teradata Vantage™ - Database Utilities*, B035-1102.

## QryLogExplainV

**Category:** Query

**Database:** DBC

| View Column      | Data Type              | Format                   | Comment  |
|------------------|------------------------|--------------------------|--|
| ProcID           | DECIMAL(5,0) NOT NULL  | -(5)9                    | Returns the process ID of the dispatcher.  |
| CollectTimeStamp | TIMESTAMP(6) NOT NULL  | YYYY-MM-DDBHH:MI:SS.S(6) | (Prime Key) Time and date when the DBQL Explain cache was written.                               |
| QueryID          | DECIMAL(18,0) NOT NULL | --Z(17)9                 | Returns a system-wide unique ID to identify the query.   |
| ExpRowNo         | INTEGER NOT NULL       | --,---,---,---9          | Needed in case multiple rows are used for Explain text. If the additional Explain text uses more |

| View Column | Data Type   | Format   | Comment  |
|-------------|---|----------|--|
|             |   |          | than 31000 characters, the system generates multiple rows. |
| ExplainText | VARCHAR(31000)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(31000) | Returns a full Explain Text segment.                       |

## Usage Notes

The QryLogExplainV view is populated if the WITH EXPLAIN option is requested in addition to the information row.

Because the explain can be larger than 31,000 characters, it may use multiple rows to hold the data.

## Referenced Columns

Many of the Data Dictionary view columns have referenced table columns. That is, the value in the view column corresponds to a value in the selected column referenced in the table. It would be meaningful to join the view and the referenced table based on the selected column and the referenced column.

Referenced columns for this view are:

| View Column | Description   |
|-------------|---|
| QueryID     | QueryID is a system-wide unique field; You can use QueryID to join DBQL tables or the DBC.TdwmExceptionLog table with DBQL tables without needing ProcID as an additional join field. |

## Example: Using QryLogExplainV

The following SELECT statement retrieves the QueryID and ExplainText for a particular QueryID.

```
SELECT Queryid, Explaintext
FROM QryLogExplainV WHERE Queryid = 201205134619838011;
```

### Note:

The output from the SELECT statement (the ExplainText) does not contain any format; it is just one long string.

## QryLogFeatureListV

**Category:** Query

**Database:** DBC

| View Column   | Data Type                              | Format | Comment  |
|---------------|--|--------|--|
| FeatureBitPos | INTEGER                                | -(10)9 | Bit position of a feature in FeatureUsage column in DBC.DBQLogTbl. |
| FeatureName   | VARCHAR(512) LATIN<br>NOT CASESPECIFIC | X(512) | Descriptive name of the feature being represented by the bit.      |

## Usage Notes

The QryLogFeatureListV view lists all the feature names that can be tracked when feature usage is enabled.

This view is populated whether or not the WITH FEATUREINFO option for BEGIN/REPLACE QUERY LOGGING is enabled. QryLogFeatureListV provides a view into the table function SYSLIB.FeatureNames\_TBF. For more information about the table function, see *Teradata Vantage™ SQL Operators and User-Defined Functions*, B035-1210.

Use this view to obtain the following information:

- How many times a feature was used in a given period of time; for example, how many times a feature was used in a day or in the last week, and so on
- List the most used feature in a release
- List what percentage of the requests used a certain feature
- List the last time a particular feature was used

---

### Note:

The feature list is subject to change each Teradata release. To see the current list of features that are tracked, see [Examples: Using QryLogFeatureListV](#).

---

To enable feature use logging, see BEGIN QUERY LOGGING in *Teradata Vantage™ SQL Data Definition Language Syntax and Examples*, B035-1444.

## Examples: Using QryLogFeatureListV

### Example: Using QryLogFeatureListV to List Features

The example displays the current list of features being logged:

```
SELECT featurename FROM DBC.QryLogFeatureListV;
```

---

### Note:

The feature list is subject to change each Teradata release.

---

### Example: Using QryLogFeatureListV to Generate a Feature Usage Report

The example generates a report on feature usage based on all the DBQLogTbl rows with FeatureUsage bitmap:

```
select cast(b.FeatureName as char(50)),
sum(GetBit(a.FeatureUsage,(2047 - b.FeatureBitpos))) as FeatureuseCount
from DBC.DbqLogTbl a, DBC.QryLogFeatureListV b
group by b.FeatureName;
```

Result:

| FeatureName                                       | FeatureuseCount |
|---|-----------------|
| -----   | -----           |
| Character Partition Primary Index                 | 4               |
| Multi Level Partition Primary Index               | 52              |
| Increased Partition Level Partition Primary Index | 1               |
| Partition Primary Index                           | 65              |

### Example: Using QryLogFeatureListV to Find the Percentage of Requests that Use Block Level Compression

The example finds the percentage of requests in DBQLogTbl that used Block Level Compression:

```
select cast(b.FeatureName as char(50)),
cast( cast(NULLIFZERO(sum(GetBit(a.FeatureUsage,(2047 - b.FeatureBitpos))))
as FLOAT)/count(a.QueryID)*100 as FLOAT FORMAT '----,---,---,---,--9.999' )
as FeatureUsePercent
from DBC.dbqlogtbl a, DBC.QryLogFeatureListV b group by b.FeatureName
where b.FeatureName = 'Block Level Compression';
```

Result:

| FeatureName             | FeatureUsePercent |
|-------------------------|-------------------|
| -----                   | -----             |
| Block Level Compression | 65.164            |

### Example: Using QryLogFeatureListV to Find the Usage Percentage of All Requests

The example finds the usage percentage of all requests:

```
select cast(b.FeatureName as char(50)),
cast( cast(NULLIFZERO(sum(GetBit(a.FeatureUsage,(2047 - b.FeatureBitpos))))
```

```
as FLOAT)/count(a.QueryID)*100 as FLOAT FORMAT '----,---,---,---,--9.999' )
as FeatureUsePercent
from DBC.dbqlogtbl a, DBC.QryLogFeatureListV b group by b.FeatureName order by
2 desc;
```

Result:

| FeatureName               | FeatureUsePercent |
|---------------------------|-------------------|
| -----                     | -----             |
| Block Level Compression   | 65.164            |
| Primary Index             | 65.164            |
| Hashed Table              | 65.164            |
| [...]                     |                   |
| Unique Hashed Index       | 23.478            |
| Parameterized Query       | 14.876            |
| Partition Level Locking   | 12.915            |
| Secondary Index           | 9.575             |
| Teradata Stored Procedure | 9.139             |
| [...]                     |                   |

### Example: Using QryLogFeatureListV to Query Native Object Store

The Native Object Store feature is logged using DBQL feature logging and can be queried using the following SQL:

```
SELECT CAST(b.FeatureName AS CHAR(75)) ,
SUM(GetBit(a.FeatureUsage,(2047 - b.FeatureBitpos))) AS FeatureuseCount
FROM DBC.DbqLogTbl a, DBC.QryLogFeatureListV b
WHERE b.FeatureName='Native Object Store'
GROUP BY b.FeatureName;
```

## QRYLOGFEATUREUSECOUNTV

**Category:** Query

**Database:** DBC

| View Column     | Data Type                           | Format | Comment   |
|-----------------|-------------------------------------|--------|---|
| FEATURENAME     | CHAR(100) LATIN<br>NOT CASESPECIFIC | X(100) | Name of the feature that is being counted.                              |
| FEATUREUSECOUNT | INTEGER                             | -(10)9 | Number of times a Feature is used by all the requests in DBC.DBQLogTbl. |

## Usage Notes

QryLogFeatureUseCountV provides use counts based on the current data in DBC.DBQLogTbl.

The QryLogFeatureUseCountV view is populated if the WITH FEATUREINFO option for BEGIN/REPLACE QUERY LOGGING and this view (QryLogFeatureUseCountV) is included in a request. QryLogFeatureUseCountV provides a view into the FeatureUsage column of DBQLogTbl and the QryLogFeatureListV view.

## Specific and Generic Feature Names

The feature list includes specific feature names and, in certain cases, generic feature names. The generic feature name is logged when any of the corresponding specific feature names is logged. Not all specific features have a corresponding generic feature group. The following table is a partial list of specific feature names with corresponding generic feature names.

### Note:

The list below is provided as an example of specific and generic feature grouping, but is not a complete list of features. The features list and the grouping of features is subject to change.

| Specific Feature Name                       | Generic Feature Name                         |
|---|--|
| 24 Teradata Columnar                        | 109 Column Partitioning                      |
| 27 Multi Level Partitioning                 | 26 Row Partitioning                          |
| 28 8 Byte Partitioning                      |  |
| 29 2 Byte Partitioning                      |  |
| 30 Column Partitioning and Row Partitioning | 26 Row Partitioning, 109 Column Partitioning |
| 31 Column-Partitioned and a NoPI            | 109 Column Partitioning                      |
| 32 Column-Partitioned and a PI              |  |
| 33 Column-Partitioned and a PA              |  |
| 42 XML Data Type                            | 41 User Defined Type                         |
| 43 JSON Data Type                           |  |
| 44 Distinct Data Type                       |  |
| 45 Structure Data Type                      |  |
| 46 Geospatial                               |  |
| 47 3D Geospatial                            |  |
| 48 Period Data Type                         |  |

| Specific Feature Name              | Generic Feature Name   |
|------------------------------------|------------------------|
| 49 Array Data Type                 | 110 Union All Pushdown |
| 50 Number Data Type                |                        |
| 104 Aggregation Push for Union All |                        |
| 105 Binary Join Push for Union All |                        |

### Example: Using QryLogFeatureUseCountV

The example generates a report on feature usage.

```
SELECT cast(featurename as CHAR(50)) as FEATURENAME, FEATUREUSECOUNT
FROM DBC.QryLogFeatureUseCountV order by 2 desc;
```

Result:

| FEATURENAME               | FEATUREUSECOUNT |
|---------------------------|-----------------|
| -----                     | -----           |
| Block Level Compression   | 8371            |
| Primary Index             | 8371            |
| Hashed Table              | 8371            |
| SET Table                 | 7366            |
| Non Unique Hashed Index   | 6018            |
| Fallback                  | 3298            |
| Unique Hashed Index       | 3016            |
| Parameterized Query       | 1911            |
| Partition Level Locking   | 1659            |
| Secondary Index           | 1230            |
| Teradata Stored Procedure | 1174            |
| [...]                     |                 |
| Autoreparse               | 15              |
| User Defined Function     | 11              |
| Join Index                | 8               |
| Bit Manipulation Function | 5               |
| Fast Path Function        | 5               |
| MloadX                    | 0               |
| [...]                     |                 |

## QryLogFeatureUseJSON

Category: Query

**Database:** DBC

| View Column      | Data Type     | Format   | Comment   |
|------------------|---------------|----------|---|
| QueryID          | DECIMAL(18,0) | --Z(17)9 | (Foreign Key) System wide unique value to join DBQL tables. |
| FeatureUsageJSON | JSON          | X(64000) | Feature Usage information in JSON format for each request.  |

## Usage Notes

The QryLogFeatureUseJSON view returns a list of features used in JSON format.

QryLogFeatureUseJSON provides a view into the FeatureUsage column of DBQLLogTbl. This view uses the system function TD\_SYSFNLIB.TD\_DBQLFUL to convert the FeatureUsage bitmap column into a JSON document with a list of features used by particular requests. The names of the used features are shown in JSON format. For details on the system function TD\_SYSFNLIB.TD\_DBQLFUL, see *Teradata Vantage™ SQL Operators and User-Defined Functions*, B035-1210.

## Referenced Columns

Many of the Data Dictionary view columns have referenced table columns. That is, the value in the view column corresponds to a value in the selected column referenced in the table. It would be meaningful to join the view and the referenced table based on the selected column and the referenced column.

Referenced columns for this view are:

| View Column | Description   |
|-------------|---|
| QueryID     | QueryID is a system-wide unique field; You can use QueryID to join DBQL tables or the DBC.TdwmExceptionLog table with DBQL tables without needing ProcID as an additional join field. |

## Example: Using QryLogFeatureUseJSON

The following SELECT statement converts the FeatureUsage column data in DBC.DBQLLogTbl from bitmap into JSON documents containing a list of features used by the request:

```
select * from QryLogFeatureUseJSON sample 1;
```

Result:

```
QueryID      307180509041677178
FeatureUsageJSON {"QueryID":"307180509041677178","FeatureInfo":["DBQL STEP
Logging","Partition Level Locking"]}
```

## QryLogObjectsV

**Category:** Query

**Database:** DBC

| View Column        | Data Type                                     | Format                   | Comment  |
|--------------------|---|--------------------------|--|
| ProcID             | DECIMAL(5,0)<br>NOT NULL                      | -(5)9                    | Returns the process ID of the dispatcher.  |
| CollectTimeStamp   | TIMESTAMP(6)<br>NOT NULL                      | YYYY-MM-DDBHH:MI:SS.S(6) | (Prime Key) Time and date when the DBQL objects cache was written.   |
| QueryID            | DECIMAL(18,0)<br>NOT NULL                     | --Z(17)9                 | Returns a system-wide unique ID to identify the query.   |
| ObjectDatabaseName | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC   | X(128)                   | Name of the database that owns the target object.  |
| ObjectTableName    | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC   | X(128)                   | Returns the name of table or view.   |
| ObjectColumnName   | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC   | X(128)                   | Returns the name of the column. If the object type is I or Index, the system returns the name of the column associated with the index. |
| ObjectID           | BYTE(4) NOT NULL                              | X(8)                     | Returns the internal ID of the object.   |
| ObjectNum          | INTEGER                                       | --,---,---,--9           | Used for indexes or columns.   |
| ObjectType         | CHAR(3) LATIN<br>NOT CASESPECIFIC<br>NOT NULL | X(3)                     | DB-Database, Tab-Table, Col-Column, Idx-Index, Mac-Macro, Viw-View, etc.   |
| FreqofUse          | INTEGER                                       | --,---,---,--9           | Returns the count of the number of optimizer references to the object from the parse tree.   |
| TypeOfUse          | VARCHAR(46)<br>UNICODE<br>NOT CASESPECIFIC    | X(46)                    | Returns the use of the object.   |

## Usage Notes

With the WITH OBJECTS option, DBQL logs one row in DBQLObjTbl and the associated view, QryLogObjectsV, for each data object referenced by the query. An object can be a database, data table, column, secondary index, join index, or journal table. (If the object is a secondary index, its number is logged rather than a name.) DBQL gets the use counts from the Optimizer and not the SQL statement itself.

The system logs a count of the number of times the Optimizer accessed the object. If objects are requested, object information is stored in a DBQL table that contains a row for each object used in the query.

**Note:**

Any DBC database tables and columns used by the system while processing a query are not reflected in the DBQL object rows for that query. This means, for example, that statements like CREATE TABLE or SELECT FROM DBC.xxx will not have objects logged through DBQL because they deal with DBC tables and columns.

If objects are requested, object information is stored in a DBQL table that contains a row for each object used in the query.

| IF the object is ... | THEN  |
|----------------------|---|
| an index             | the field ID is logged instead of a name.   |
| a column             | <ul style="list-style-type: none"> <li>the field ID is logged instead of a name</li> <li>the object ID can be used to qualify the field ID, which is not unique across the system.</li> </ul> |

Object frequency can also be logged. This is the number of times an object is used in a query.

**Possible Values for ObjectType**

| Value | Description  |
|-------|--|
| Agg   | User-defined aggregate function  |
| AgS   | User-defined aggregate STAT function   |
| Aut   | Security authorization   |
| Col   | Column   |
| DB    | Database   |
| GLP   | GLOP set   |
| HIx   | Hash index   |
| Idx   | Index. For each index, there is a database name, table name, and column name. The ObjectID column is the identifier of the table and the ObjectNum column is the number of the index in that table. For multi-column indexes, there is one row for each column of the index that a query used. For example, if an index consists of three columns and the query uses all three, there will be three rows, each with a different column name. The column name will be null for an index for statements such as COLLECT STATISTICS, ALTER PROCEDURE, SHOW PROCEDURE, or SELECT COUNT(*). |
| JIx   | Join index. For each join index, there is a database name and join index name in the ObjectTableName field. For these rows, the ColumnName indicates a column referred to by the join index. <ul style="list-style-type: none"> <li>ObjectType is 'JIx.'</li> </ul>  |

| Value | Description   |
|-------|---|
|       | <ul style="list-style-type: none"> <li>• ObjectId matches the ID of the join index.</li> <li>• ObjectNum is 0.</li> </ul> |
| Jrl   | Journal   |
| Mac   | Macro   |
| NoT   | No type (unknown)   |
| SP    | Stored procedure  |
| Sta   | User-defined STAT function  |
| Tab   | Table   |
| TbF   | Table function  |
| Tmp   | Temporary   |
| TbO   | Table operator  |
| TbC   | Contract function   |
| Trg   | Trigger   |
| UDF   | User-defined function   |
| UDM   | User-defined method   |
| UDT   | User-defined type   |
| Viw   | View  |
| Vol   | Volatile  |
| XSP   | External stored procedure   |

### Possible Values for TypeOfUse

| Value | Description                      |
|-------|----------------------------------|
| 1     | Found in the resolver            |
| 2     | Accessed during query processing |
| 3     | Reference, access                |
| 4     | Found in a conditional text      |
| 6     | Access, conditional              |
| 7     | Reference, access, conditional   |
| 8     | Found in inner join condition    |

| Value | Description                               |
|-------|---|
| 10    | Access, inner join                        |
| 14    | Access, conditional, inner join           |
| 16    | Found in outer join condition             |
| 18    | Access, outer join                        |
| 22    | Access, conditional, outer join           |
| 30    | Access, conditional, inner and outer join |
| 32    | Found in sum node                         |
| 34    | Access, sum                               |
| 38    | Access, conditional, sum                  |
| 46    | Access, conditional, sum, inner join      |
| 54    | Access, conditional, sum, outer join      |
| 64    | Found in a full outer join condition      |
| 70    | Access, conditional, full outer join      |
| 102   | Access, conditional, sum, full outer join |

## Example: Using QryLogObjectsV

The following SELECT statement retrieves the object information of a query:

```
SELECT QueryID, ObjectDatabaseName (Named ObjDBName), ObjectTableName (Named
ObjTblName), ObjectColumnName (Named ObjColName), ObjectID (Named ObjID),
ObjectNum (Named ObjNum), ObjectType (Named ObjType), FreqofUse
FROM DBC.QryLogObjectsV where queryid = 201205134619838031;
```

Result:

| QueryID<br>eqofUse | ObjDBName   | ObjTblName | ObjColName | ObjID    | ObjNum | ObjType | Fr    |
|--------------------|-------------|------------|------------|----------|--------|---------|-------|
| -----              | -----       | -----      | -----      | -----    | -----  | -----   | ----- |
| 201205134619838031 | D_PERSONNEL | ?          | ?          | 00001604 | 0      | DB      | 1     |
| 201205134619838031 | D_PERSONNEL | DEPARTMENT | ?          | 00009005 | 0      | Tab     | 1     |
| 201205134619838031 | D_PERSONNEL | DEPARTMENT | DeptNo     | 00009005 | 1,025  | Col     | 2     |
| 201205134619838031 | D_PERSONNEL | DEPARTMENT | DeptName   | 00009005 | 1,026  | Col     | 1     |
| 201205134619838031 | D_PERSONNEL | DEPARTMENT | EmpCount   | 00009005 | 1,027  | Col     | 1     |
| 201205134619838031 | D_PERSONNEL | DEPARTMENT | Loc        | 00009005 | 1,028  | Col     | 1     |

# QryLogParamJSON

**Category:** Query

**Database:** DBC

| View Column | Data Type     | Format   | Comment   |
|-------------|---------------|----------|---|
| QueryID     | DECIMAL(18,0) | --Z(17)9 | (Foreign Key) System wide unique value to join DBQL tables.                     |
| RowNum      | INTEGER       | -(10)9   | Row number indicating the sequence of ParamInfo and Data Record JSON documents. |
| ParamJSON   | JSON          | X(64000) | Parameter and data rows in JSON format.   |

## Usage Notes

This view provides access to parameter and metadata information in Teradata JSON UDT type.

This view contains confidential user information and should be accessed only by trusted personnel with access to the restrictive user DBC password. For more information about the user DBC password, see *Teradata Vantage™ - Advanced SQL Engine Security Administration*, B035-1100.

The QryLogParamJSON view is populated if parameterized request logging is enabled with the PARAMINFO option for BEGIN/REPLACE QUERY LOGGING and this view (QryLogParamJSON) is included in a request.

For example:

```
SELECT QueryID, RowNum, ParamJSON from QryLogParamJSON;
```

converts all the rows in DBQLParamTbl to JSON documents. With this view, users can perform JSON operations on the ParamJSON column data.

For more information about this option, see *Teradata Vantage™ SQL Data Definition Language Syntax and Examples*, B035-1144.

## RowNum

This column exists only in the QryLogParamJSON view, which is generated by the TD\_SYSFNLIB.TD\_DBQLParam table operator function at runtime. For more information about this function, see *Teradata Vantage™ SQL Operators and User-Defined Functions*, B035-1210.

## ParamJSON

This column is generated by the TD\_SYSFNLIB.TD\_DBQLParam table operator function at runtime. This column converts the parameter values and metadata logged to the DBC.DBQLParamTbl.ParamInfo table column in Teradata JSON UDT type. For more information about this TD\_SYSFNLIB.TD\_DBQLParam table operator function, see *Teradata Vantage™ SQL Operators and User-Defined Functions*, B035-1210.

### Example: Select JSON Data from QryLogParamJSON

This example assumes the PARAMINFO option is enabled in the BEGIN/REPLACE QUERY LOGGING statement. The SELECT statement selects data from QryLogParamJSON and converts all rows in the DBC.DBQLParamTbl table to a JSON document.

Run this query from BTEQ with width set to 500.

```
.width 500;
select QueryID, RowNum, ParamJSON from QryLogParamJSON;
```

Result:

| QueryID            | RowNum | ParamJSON  |
|--------------------|--------|--|
| -----              |        | -----  |
| 307190733539634351 | 1      | {"QueryID":"307190733539634351","HostCharSet":"127",<br>"ParamInfo":[{"Name":"x","Type":"INTEGER","Size":4,<br>"Position":1}]} |
| 307190733539634351 | 2      | {"QueryID":"307190733539634351","Data Record":{"x":"10"}}  |

### Example: Select PARAMINFO from QryLogParamJSON

This example returns the PARAMINFO information for each parameter in the JSON document in a row. This returns the name, type, position, and value for each parameter in the parameterized request.

```
select QueryID, RowNum, ParamJSON.ParamInfo from QryLogParamJSON;
```

Result:

| d<br>QueryID       | RowNum | ParamJSON.ParamInfo                                   |
|--------------------|--------|---|
| -----              | -----  | -----   |
| 307190733539634351 | 1      | [{"Name":"x","Type":"INTEGER","Size":4,"Position":1}] |
| 307190733539634351 | 2      | ?   |

## Example: Extract JSON Name/Value Pairs from the JSON Document

This example selects particular name/value pairs from the JSON document using JSON string syntax. If a row does not have a specific name requested, null is returned.

```
select QueryID, RowNum, ParamJSON."Data Record" from QryLogParamJSON;
```

Result:

| QueryID            | RowNum | ParamJSON.Data Record |
|--------------------|--------|-----------------------|
| 307190733539634351 | 1      | ?                     |
| 307190733539634351 | 2      | {"x": "10"}           |

## Example: Extract Specific JSON Values from the JSON Document

This example selects specific values from the JSON name/value pair using JSON string syntax. If a row does not have a specific name requested, null is returned.

```
select QueryID, RowNum, ParamJSON."Data Record".x from QryLogParamJSON;
```

Result:

| QueryID            | RowNum | ParamJSON.Data Record.x |
|--------------------|--------|-------------------------|
| 307190733539634351 | 1      | ?                       |
| 307190733539634351 | 2      | 10                      |

## QryLogParamV

**Category:** Query

**Database:** DBC

| View Column | Data Type     | Format   | Comment   |
|-------------|---------------|----------|---|
| QueryID     | DECIMAL(18,0) | --Z(17)9 | (Foreign Key) System wide unique value to join DBQL tables.                     |
| RowNum      | INTEGER       | -(10)9   | Row number indicating the sequence of ParamInfo and Data Record JSON documents. |

| View Column | Data Type                                  | Format   | Comment                                 |
|-------------|--|----------|---|
| ParamJSON   | CLOB(2097088000) LATIN<br>NOT CASESPECIFIC | X(64000) | Parameter and data rows in JSON format. |

## Usage Notes

This view contains confidential user information and should be accessed only by trusted data personnel with access to the restrictive user DBC password. For more information about the user DBC password, see *Teradata Vantage™ - Advanced SQL Engine Security Administration*, B035-1100.

The QryLogParamV view is populated if parameterized request logging is enabled with the PARAMINFO option for BEGIN/REPLACE QUERY LOGGING and this view (QryLogParamV) is included in a request.

For example:

```
SELECT QueryID, RowNum, ParamJSON from QryLogParamV;
```

converts all the rows in DBQLParamTbl to JSON documents. The output is in CLOB format. To perform JSON operations on the data, use the QryLogParamJSON view instead.

For more information, see *Teradata Vantage™ SQL Data Definition Language Syntax and Examples*, B035-1144.

### RowNum

This column is generated by the TD\_SYSFNLIB.TD\_DBQLParam table operator function at runtime. For more information about this function, see *Teradata Vantage™ SQL Operators and User-Defined Functions*, B035-1210.

### ParamJSON

This column is generated by the TD\_SYSFNLIB.TD\_DBQLParam table operator function at runtime. This column converts the parameter values and metadata logged to the DBC.DBQLParamTbl.ParamInfo table column into JSON format. For more information about this TD\_SYSFNLIB.TD\_DBQLParam table operator function, see *Teradata Vantage™ SQL Operators and User-Defined Functions*, B035-1210.

## Example: Using QryLogParamV

This example assumes the PARAMINFO option is enabled in the BEGIN/REPLACE QUERY LOGGING statement. The following SELECT statement converts all rows in the DBC.DBQLParamTbl table to a JSON document:

```
SELECT QueryID, RowNum, ParamJSON from QryLogParamV;
```

Result:

```

QueryID          RowNum ParamJSON
-----
307192920408671138 1
{"QueryID":"307192920408671138","HostCharSet":"127","ParamInfo":
    [{"Name":"xABc","Type":"INTEGER","Size":4,"Position":1},
     {"Name":"yBflt","Type":"REAL","Size":8,"Position":2},
     {"Name":"zCdbl","Type":"REAL","Size":8,"Position":3},
     {"Name":"fxStr","Type":"CHAR","Size":20,"Position":4},
     {"Name":"varStr","Type":"VARCHAR","Size":25,"Position":5},
     {"Name":"fxByte","Type":"BYTE","Size":4,"Position":6},
     {"Name":"vrByte","Type":"VARBYTE","Size":25,"Position":7},
     {"Name":"nmbr","Type":"NUMBER","Size":18,"Position":8},
     {"Name":"dcml","Type":"DECIMAL","Size":8,"Position":9},
     {"Name":"dt","Type":"DATE","Size":4,"Position":10},
     {"Name":"ts","Type":"CHAR","Size":26,"Position":11},
     {"Name":"blb","Type":"BLOB","Size":60,"Position":12},
     {"Name":"clb","Type":"CLOB","Size":60,"Position":13},
     {"Name":"intrvl","Type":"CHAR","Size":5,"Position":14},
     {"Name":"tme","Type":"CHAR","Size":15,"Position":15}]}
307192920408671138 2
{"QueryID":"307192920408671138","Data
Record":{"xABc":"1","yBflt":"+5.78000000000000E000",
"zCdbl":"+9.86700000000000E-001","fxStr":null,
"varStr":"Test Var String 01","fxByte":"00005AB1",
"vrByte":"5ABCFE6789EFBCAB5EF0","nmbr":"1234.679","dcml
":"54328567.45","dt":"2013/09/10","ts":"2013-09-10
10:41:32.000000","blb":"BAABBCCDDEEFF123456789AABBCCDDEEFF",
"clb":"This is a CLOB column string 01","intrvl":"
7859",
"tme":"10:56:35.000000"}}

```

## QryLogSQLDocV

**Category:** Query

**Database:** DBC

| View Column      | Data Type             | Format                   | Comment  |
|------------------|-----------------------|--------------------------|--|
| ProcID           | DECIMAL(5,0) NOT NULL | -(5)9                    | Returns the process ID of the dispatcher.                      |
| CollectTimeStamp | TIMESTAMP(6) NOT NULL | YYYY-MM-DDBHH:MI:SS.S(6) | (Prime Key) Time and date when the DBQL SQL cache was written. |

| View Column | Data Type                                       | Format   | Comment  |
|-------------|---|----------|--|
| QueryID     | DECIMAL(18,0) NOT NULL                          | --Z(17)9 | Returns a system-wide unique ID to identify the query.                                 |
| SqlTextDoc  | CLOB(1048544000)<br>UNICODE<br>NOT CASESPECIFIC | X(32000) | Returns a full SQL text segment unless an error is encountered, or the SQL is aborted. |

## Usage Notes

The QryLogSQLDocV view is populated if you request that SQL is shown for a query along with the default information. The WITH SQL option causes DBQL to attempt to log the full statement text, no matter how large, into the DBQLSQLTbl table.

### SqlTextDoc

SQLTextDoc is a result of combining multiple DBQLSQLTbl.SQLTextInfo rows with the same QueryID into a single XML document as CLOB type.

## Example: Using QryLogSQLDocV

The following SELECT statement retrieves the SQL text information of a query.

```
.export file=sqldoc.txt;
sel sqltextdoc from QryLogSQLDocV where queryid=307199040105796016;
.export reset;
```

Result: The following shows a partial result of the query.

```
SqlTextDoc
-----
-----
SELECT * /* This is a test of query with large query text On the Insert tab,
the galleries include items that are designed to coordinate with the overall look
of your document. You can use these galleries to insert tables, headers,
footers, lists, cover pages, and other document building blocks. When you create
pictures, charts, or diagrams, they also coordinate with your current document
look. You can easily change the formatting of selected text in the document text
by choosing a look for the selected text from the Quick Styles gallery on the
Home tab. You can also format text directly by using the other controls on the
Home tab. Most controls offer a choice of using the look from the current theme
or using a format that you specify directly. To change the overall look of your
document, choose new Theme elements on the Page Layout tab. To change the looks
available in the Quick Style gallery, use the Change Current Quick Style Set
```

command. Both the Themes gallery and the Quick Styles gallery provide reset commands so that you can always restore the look of your document to the original contained in your current template. On the Insert tab, the galleries include items that are designed to coordinate with the overall look of your document. You can use these galleries to insert tables, headers, footers, lists, cover pages, and other document building blocks. When you create pictures, charts, or diagrams, they also coordinate with your current document look. You can easily change the formatting of selected text in the document text by choosing a look for the selected text from the Quick Styles gallery on the Home tab. You can also format text directly by using the other controls on the Home tab. Most controls offer a choice of using the look from the current theme or using a format that you specify directly. To change the overall look of your document, choose new Theme elements on the Page Layout tab. To change the looks available in the Quick Style gallery, use the Change Current Quick Style Set command.

.

.

.

## Improving SQL Text Readability

To combine multiple SQLTextDoc rows with the same query ID into one easily readable document, perform a SELECT request on the QryLogSQLDocV view.

### Example

```
sel queryid,sqltextdoc from qrylogsqldocv where queryid=977188741514475132;
```

### Note:

Export the result set to a file. On the monitor, some of the lines may truncate.

The memory allocated for the input is 32 MB. To see the memory setting, use the following:

```
cufconfig -o | grep MallocLimit
```

To add more memory, adjust the MallocLimit setting using cufconfig. For more information, see *Teradata Vantage™ - Database Utilities*, B035-1102.

## QryLogSQLV

**Category:** Query

**Database:** DBC

| View Column      | Data Type  | Format                    | Comment  |
|------------------|--|---------------------------|--|
| ProcID           | DECIMAL(5,0) NOT NULL                            | -(5)9                     | Returns the process ID of the dispatcher.                                  |
| CollectTimeStamp | TIMESTAMP(6) NOT NULL                            | YYYY-MM-DDDBHH:MI:SS.S(6) | (Prime Key) Time and date when the DBQL SQL cache was written.             |
| QueryID          | DECIMAL(18,0) NOT NULL                           | --Z(17)9                  | Returns a system-wide unique ID to identify the query.                     |
| SqlRowNo         | INTEGER NOT NULL                                 | --,---,---,--9            | Returns row number in the case of multiple rows are used for the SQL.      |
| SqlTextInfo      | VARCHAR(31000) UNICODE NOT CASESPECIFIC NOT NULL | X(31000)                  | A full SQL text segment, a string of up to approximately 31,000 characters |

## Usage Notes

The QryLogSQLV view is populated if you request that SQL is shown for a query along with the default information. The WITH SQL option causes DBQL to attempt to log the full statement text, no matter how large, into the DBQLSQLTbl table. If the SQL text is greater than 31,000 characters, multiple rows are generated.

### Possible Values for SQLTextInfo

SQLTextInfo contains a full SQL text segment, a string of up to approximately 32,000 characters. If the SQL text is greater than 31,000 characters, multiple rows are generated. Note, BTEQ has a limited column length; use Teradata Studio to display longer lengths.

SQLTextInfo can also contain the following:

- Unavailable
- Null

---

#### Note:

SqlTextInfo may be Null or it may be Unavailable when the SQL is not available to DBQL at the end of the query or when a query fails. Unavailable is seen when it fails to translate correctly, for example, when the SQL is submitted from a mainframe in EBCDIC format or when UNICODE translation is required.

---

## Example: Using QryLogSQL

The following SELECT statement retrieves the SQL text information of a query.

```
SELECT QueryID, SqlRowNo, SqlTextInfo from DBC.QryLogSQL
WHERE QueryId=201205134619838024;
```

Result:

| QueryID            | SqlRowNo | SqlTextInfo   |
|--------------------|----------|---|
| 201205134619838024 | 1        | CREATE TABLE table1A,<br>FALLBACK, NO BEFORE JOURNAL,<br>NO AFTER JOURNAL (i INTEGER,<br>i2 INTEGER)<br>PRIMARY INDEX( i ); |

**Note:**

The SQL text is not edited to remove any blanks; it is stored the same way you entered the text.

## QryLogStepsV

**Category:** Query

**Database:** DBC

| View Column      | Data Type                                     | Format                   | Comment  |
|------------------|---|--------------------------|--|
| ProcID           | DECIMAL(5,0)<br>NOT NULL                      | -(5)9                    | Returns (Prime Key) ID of the Processing Engine from which the query was executed.     |
| CollectTimeStamp | TIMESTAMP(6)<br>NOT NULL                      | YYYY-MM-DDBHH:MI:SS.S(6) | Returns (Prime Key) Time and date when the DBQL steps cache was written.               |
| QueryID          | DECIMAL(18,0)<br>NOT NULL                     | --Z(17)9                 | Returns (Foreign Key) Systemwide unique value to join DBQL tables.                     |
| StepLev1Num      | SMALLINT<br>NOT NULL                          | ---,--9                  | Returns Level 1 step number.   |
| StepLev2Num      | SMALLINT                                      | ---,--9                  | Returns Level 2 step number for parallel steps.  |
| StepName         | CHAR(6) LATIN<br>NOT CASESPECIFIC<br>NOT NULL | X(6)                     | Returns an abbreviation of the step name. For example: DEL for delete step.            |
| StepStartTime    | TIMESTAMP(6)                                  | YYYY-MM-DDBHH:MI:SS.S(6) | Returns the date and time to the nearest microsecond when the step is sent to the AMP. |

| View Column          | Data Type    | Format                   | Comment   |
|----------------------|--------------|--------------------------|---|
| StepStopTime         | TIMESTAMP(6) | YYYY-MM-DDBHH:MI:SS.S(6) | Returns the date and time to the nearest microsecond when the step returns from the AMP.  |
| ElapsedTime          | HS           | -h(4):mm:ss.s(6)         | Returns the difference between last response time and start time (to the nearest microsecond)   |
| EstProcTime          | FLOAT        | ---,---,---,---,--9.999  | Returns the estimated processing time in seconds from the Optimizer.  |
| EstCPUCost           | FLOAT        | ZZ,ZZZ,ZZ9.999           | Returns an estimate (from the optimizer) of the milliseconds of CPU time for the step.  |
| CPUTime              | FLOAT        | ZZ,ZZZ,ZZ9.999           | Amount of CPU time in seconds used by the step.   |
| IOcount              | FLOAT        | ---,---,---,---,--9      | Number of logical IOs issued by the step.   |
| EstRowCount          | FLOAT        | ---,---,---,---,--9      | Returns the estimated row count from the Optimizer; For PRPD step, the total estimated row count.   |
| EstRowCountSkew      | FLOAT        | ---,---,---,---,--9      | For PRPD step, returns the estimated row count of skew spool.   |
| EstRowCountSkewMatch | FLOAT        | ---,---,---,---,--9      | For PRPD step, returns the estimated row count of skewmatch spool.  |
| RowCount             | FLOAT        | ---,---,---,---,--9      | Returns the number of rows inserted for the MultiLoad EXE step and the Merge Row Multiple (MRM) step; For PRPD step, the total row count of all split spools; For other steps, RowCount is the number of rows returned by the step. |
| RowCount2            | FLOAT        | ---,---,---,---,--9      | Returns the number of rows updated by the MultiLoad EXE step or the MRM step. For FastLoad LFI step, RowCount2 is the number of rows loaded by the step; For PRPD step, the row count of skew spool.                                |
| RowCount3            | FLOAT        | ---,---,---,---,--9      | For MultiLoad EXE step, number of rows deleted. For PRPD step, the row count of skewmatch spool.  |
| NumOfActiveAMPs      | INTEGER      | --,---,---,--9           | Returns the number of AMPs that were active for this query.   |

| View Column       | Data Type | Format                         | Comment   |
|-------------------|-----------|--------------------------------|---|
| MaxAmpCPUTime     | FLOAT     | ZZ,ZZZ,ZZ9.<br>999             | Returns the CPU time of the highest CPU utilized AMP in the query in seconds.                                   |
| MaxCPUAmpNumber   | INTEGER   | --,---,---,--9                 | Returns the number of the AMP with the highest CPU activity.  |
| MinAmpCPUTime     | FLOAT     | ZZ,ZZZ,ZZ9.<br>999             | Returns the CPU time of the lowest CPU utilized AMP in the step in seconds.                                     |
| MaxAmpIO          | FLOAT     | ---,---,---,---,<br>--9        | (renamed from HotAmpIO)<br>Returns the I/O count of the highest I/O utilized AMP in the step.                   |
| MaxIOAmpNumber    | INTEGER   | --,---,---,--9                 | Returns the number of the AMP with the highest IO usage for this step.  |
| MinAmpIO          | FLOAT     | ---,---,---,---,<br>--9        | (renamed from LowAmpIO)<br>Returns the I/O count of the lowest I/O utilized AMP in the query.                   |
| SpoolUsage        | BIGINT    | --,---,---,---,<br>---,---,--9 | Peak Spool usage (bytes) of this step (DataCollectAlg=3). Otherwise, number of bytes of spool used by the step. |
| MaxAMPSpool       | BIGINT    | --,---,---,---,<br>---,---,--9 | Returns the highest spool usage on an AMP. This is the number of bytes of spool at the step level.              |
| MaxSpoolAmpNumber | INTEGER   | --,---,---,--9                 | Returns the number of the AMP with high spool usage.  |
| MinAMPSpool       | BIGINT    | --,---,---,---,<br>---,---,--9 | Returns the lowest Spool usage on an AMP. This is the number of bytes of spool at the step level.               |
| StepWD            | INTEGER   | --,---,---,--9                 | Returns the identifier of the workload for the step.  |
| LSN               | INTEGER   | --,---,---,--9                 | Logon Sequence Number used by the load utility.   |
| UtilityTableID    | BYTE(4)   | X(8)                           | Returns the table ID for the load utilities.  |
| RowsWComprColumns | FLOAT     | ---,---,---,---,<br>--9        | For each step, the system returns the number of rows which contains at least one compressed column.             |
| EstIOCost         | FLOAT     | ZZ,ZZZ,ZZ9.<br>999             | Returns an estimate (from the optimizer) of service time in milliseconds for IO for the step.                   |

| View Column           | Data Type                             | Format                         | Comment   |
|-----------------------|---------------------------------------|--------------------------------|---|
| EstNetCost            | FLOAT                                 | ZZ,ZZZ,ZZ9.<br>999             | Returns an estimate (from the optimizer) of the BYNET service time in milliseconds for the step.  |
| EstHRCost             | FLOAT                                 | ZZ,ZZZ,ZZ9.<br>999             | Returns an estimate (from the optimizer) of other costs for the step.   |
| CPUtimeNorm           | FLOAT                                 | ZZ,ZZZ,ZZ9.<br>999             | Normalized amount of CPU time in seconds used by the step.  |
| MaxAmpCPUTimeNorm     | FLOAT                                 | ZZ,ZZZ,ZZ9.<br>999             | Returns the normalized maximum CPU Time in seconds for AMP.   |
| MaxCPUAmpNumberNorm   | INTEGER                               | --,---,---,--9                 | Returns the number of the AMP with highest normalized CPU time for the step.  |
| MinAmpCPUTimeNorm     | FLOAT                                 | ZZ,ZZZ,ZZ9.<br>999             | Returns the normalized minimum CPU Time in seconds for AMP.   |
| NumCombinedPartitions | BIGINT                                | --,---,---,---,<br>---,---,--9 | Returns the number of combined partitions accessed (not eliminated).  |
| NumContexts           | INTEGER                               | --,---,---,--9                 | Returns the number of contexts allocated to simultaneously access partitions of a source or target table with column partitioning.  |
| NumCPReferences       | INTEGER                               | --,---,---,--9                 | Returns the number of column partitions referenced in a source or target table that has column partitioning.  |
| StepInstance          | INTEGER                               | --,---,---,--9                 | Step instance number for iterative steps in order of dispatch. If a step is dispatched 3 times, there will be 3 steps logged with the same step number and step instance 1, 2, 3 to distinguish them. |
| StepStatus            | CHAR(10) LATIN<br>NOT<br>CASESPECIFIC | X(10)                          | Step status, indicating dispatcher and AMP step disposition.  |
| DispatchSeq           | INTEGER                               | --,---,---,--9                 | Dispatch sequence, the order in which the step was "handled by the dispatcher itself" or dispatched to the AMPS beginning with 1.   |
| StatementNum          | INTEGER                               | --,---,---,--9                 | Statement number with which the step is associated, starting with 1. The highest related statement number if step is related to multiple statements.  |

| View Column      | Data Type                                 | Format                         | Comment   |
|------------------|---|--------------------------------|---|
| TriggerNestLevel | SMALLINT                                  | -(5)9                          | Trigger nesting level.  |
| TriggerKind      | VARCHAR(128)<br>LATIN NOT<br>CASESPECIFIC | X(128)                         | Trigger kind, space separated string of abbreviated values for the bit flags.                                       |
| FragmentNum      | INTEGER                                   | --,---,---,--9                 | Fragement number to which the step belongs in a IPE plan execution of request.                                      |
| IOKB             | FLOAT                                     | ---,---,---,---,<br>--9.999    | Logical AMP I/Os in KB generated by the step, or set of parallel steps if the second step level is 1.               |
| VHLogicalIO      | FLOAT                                     | ---,---,---,---,<br>--9.999    | Number of Very Hot Logical AMP I/Os generated by the step, or set of parallel steps if the second step level is 1.  |
| VHPhysIO         | FLOAT                                     | ---,---,---,---,<br>--9.999    | Number of Very Hot Physical AMP I/Os generated by the step, or set of parallel steps if the second step level is 1. |
| VHLogicalIOKB    | FLOAT                                     | ---,---,---,---,<br>--9.999    | Very Hot Logical AMP I/Os in KB generated by the step, or set of parallel steps if the second step level is 1.      |
| VHPhysIOKB       | FLOAT                                     | ---,---,---,---,<br>--9.999    | Very Hot Physical AMP I/Os in KB generated by the step, or set of parallel steps if the second step level is 1.     |
| PhysIO           | FLOAT                                     | ---,---,---,---,<br>--9.999    | Number of Physical AMP I/Os generated by the step, or set of parallel steps if the second step level is 1.          |
| PhysIOKB         | FLOAT                                     | ---,---,---,---,<br>--9.999    | Physical AMP I/Os in KB generated by the step, or set of parallel steps if the second step level is 1.              |
| LockDelay        | FLOAT                                     | ---,---,---,---,<br>--9.999    | Maximum wait time to get a lock on object in centi-seconds.   |
| SSRReceiverCount | INTEGER                                   | --,---,---,--9                 | The total number of SSR receiver AMP that the step uses.  |
| DMLLoadID        | INTEGER                                   | --,---,---,--9                 | Returns the load value used by the CLDI modification step performed on LDI table/join index.                        |
| ServerByteCount  | BIGINT                                    | --,---,---,---,<br>---,---,--9 | Total bytes sent and received from the foreign server by the step.  |

| View Column            | Data Type     | Format                         | Comment  |
|------------------------|---------------|--------------------------------|--|
| PersistentSpool        | BIGINT        | --,---,---,---,<br>---,---,--9 | Persistent part of SpoolUsage.   |
| OneMBRowCount          | FLOAT         | ---,---,---,---,<br>--9        | The number of 1MB rows (1MB > size > 64KB) in the set of rows returned by the step.  |
| MaxOneMBRowSize        | INTEGER       | --,---,---,--9                 | The largest row size among 1MB rows returned by step RET, JIN, SAMP, STATFN.   |
| MaxNumMapAMPs          | INTEGER       | --,---,---,--9                 | The number of AMPs in the largest map used by the step.  |
| MinNumMapAMPs          | INTEGER       | --,---,---,--9                 | Number of AMPs in the smallest contiguous map used by this step.   |
| SysDefNumMapAMPs       | INTEGER       | --,---,---,--9                 | Number of AMPs in the system-default map.  |
| SourceMapNo            | INTEGER       | --,---,---,--9                 | The map number for the source table in the step.   |
| DestMapNo              | INTEGER       | --,---,---,--9                 | The map number for the output table in the step.   |
| StepObjectInfo         | VARBYTE(1024) | X(2048)                        | Contains binary encoded information about tables, spools, and join indexes accessed by the step and is intended for use by Advisor procedures. |
| UsedIota               | FLOAT         | ---,---,---,---,<br>--9.999    | Return IO Tokens used by the step.   |
| PermUsage              | BIGINT        | --,---,---,---,<br>---,---,--9 | Returns the permanent space in bytes added or deleted by the step.   |
| MaxAMPPerm             | BIGINT        | --,---,---,---,<br>---,---,--9 | Returns the maximum permanent space in bytes added or deleted by an AMP in the step.   |
| MaxPermAmpNumber       | INTEGER       | --,---,---,--9                 | Returns the AMP number that added or deleted the maximum permanent space in the step.  |
| SpaceDelay             | FLOAT         | ---,---,---,---,<br>--9.999    | Returns the maximum wait time in centi-seconds to get space from the global space accounting system.   |
| MaxSpaceDelayAmpNumber | INTEGER       | --,---,---,--9                 | Returns the AMP number that waited the maximum to obtain space.  |

| View Column          | Data Type                             | Format                         | Comment  |
|----------------------|---------------------------------------|--------------------------------|--|
| UAFName              | CHAR(10) LATIN<br>NOT<br>CASESPECIFIC | X(10)                          | Name of an Analytic Function being executed by EXECUTE FUNCTION in this step. The default is NULL. |
| InMemBulkQualStatus  | BYTEINT                               | -(3)9                          | Returns the bulk qualification status of a step.   |
| NosRecordsReturned   | BIGINT                                | --,---,---,---,<br>---,---,--9 | Number of records returned by a Native Object Store step.  |
| NosRecordsSkipped    | BIGINT                                | --,---,---,---,<br>---,---,--9 | Number of records skipped by a Native Object Store step.   |
| NosPhysReadIO        | BIGINT                                | --,---,---,---,<br>---,---,--9 | Total physical IOs for Native Object Store files for this step.                                    |
| NosPhysReadIOKB      | FLOAT                                 | ---,---,---,---,<br>--9.999    | Total KB of physical IOs for Native Object Store files in this step.                               |
| NosRecordsReturnedKB | FLOAT                                 | ---,---,---,---,<br>--9.999    | Total KB of records returned for Native Object Store files in this step.                           |
| NosTotalIOWaitTime   | FLOAT                                 | ---,---,---,---,<br>--9.999    | Total of the Native Object Store IO wait time in seconds in this step.                             |
| NosCPUTime           | FLOAT                                 | ---,---,---,---,<br>--9.999    | CPU time in seconds for reading Native Object Store files in this step.                            |
| NosMaxIOWaitTime     | FLOAT                                 | ---,---,---,---,<br>--9.999    | Max Native Object Store IO wait time for any individual IO in seconds.                             |
| NosFiles             | INTEGER                               | --,---,---,--9                 | Number of file reads attempted by Native Object Store in this step.                                |
| NosFilesSkipped      | INTEGER                               | --,---,---,--9                 | Number of Native Object Store files that were skipped in this step.                                |
| AWTTime              | FLOAT                                 | ---,---,---,---,<br>--9.999    | Returns AWT elapsed time in seconds for the step.  |
| MaxAWTTime           | FLOAT                                 | ---,---,---,---,<br>--9.999    | Returns Max AWT elapsed time in seconds for the step.  |
| MaxAWTTimeAmpNum     | INTEGER                               | --,---,---,--9                 | Returns AMP number that had max AWTTime for the step.  |
| MinAWTTime           | FLOAT                                 | ---,---,---,---,<br>--9.999    | Returns Min AWT elapsed time in seconds for the step.  |
| UDFMemUsage          | FLOAT                                 | ---,---,---,---,<br>--9.999    | Memory used by UDF for this step.  |

| View Column          | Data Type | Format                         | Comment   |
|----------------------|-----------|--------------------------------|---|
| MaxUDFMemUsage       | FLOAT     | ---,---,---,---,<br>--9.999    | Max memory in bytes used by UDF for this step.                              |
| MaxUDFMemUsageAmpNum | INTEGER   | --,---,---,--9                 | AMP number that had max memory used by UDF for this step.                   |
| UDFVMPeak            | FLOAT     | ---,---,---,---,<br>--9.999    | UDF Virtual memory peak in bytes for the step.                              |
| UDFVMData            | FLOAT     | ---,---,---,---,<br>--9.999    | UDF Virtual memory data size in bytes for the step.                         |
| NosPhysWriteIO       | BIGINT    | --,---,---,---,<br>---,---,--9 | Total physical write IOs for Native Object Store files for this step.       |
| NosPhysWriteIOKB     | FLOAT     | ---,---,---,---,<br>--9.999    | Total KB of physical write IOs for Native Object Store files for this step. |
| NosFilesWritten      | INTEGER   | --,---,---,--9                 | Number of file writes attempted by Native Object Store in this step.        |

## Usage Notes

The QryLogStepsV view of DBQLStepTbl is populated if you specify the WITH STEPINFO option. When the query completes, the system logs one row for each query step, including parallel steps.

### CPUTimeNorm

The CPUTimeNorm calculation is made for each PE in the system. It is used for systems with processors where each node may have CPUs with different scaling factors.

### FragmentNum

This column is NULL for static plans only.

### RowCount

For:

- The MRM (Merge Row Multiple) or EXE (MultiLoad) steps, RowCount is the number of rows inserted.
- All other steps, RowCount is the actual number of rows the step returns (indicating activity count). For a PRPD plan split step, RowCount includes the rows from all split spools.

#### Note:

If “split into” appears in the EXPLAIN of a RETRIEVE or JOIN step, the database is using PRPD.

If the row count for a step is 18,446,744,073,709,551,615, the number is logged in the RowCount column as 1.8446744073709552e+19.

**Note:**

The value formatted as a decimal value is 18,446,744,073,709,551,616 due to float arithmetic.

**RowCount2**

For:

- The MRM (Merge Row Multiple) or EXE (MultiLoad) steps, RowCount2 is the number of updated rows.
- An LFI (FastLoad) step, RowCount2 is the number of rows loaded.
- A PRPD plan split step, RowCount2 is the actual number of rows in the skew split pool.

**RowCount3**

For:

- The EXE (MultiLoad) step, RowCount3 is the number of rows deleted.

**Note:**

For the number of rows inserted by the EXE step, see RowCount. For the number of rows updated by the EXE step, see RowCount2.

- A PRPD plan split step, RowCount3 is the actual number of rows in the skew match split pool.

**Possible Values for InMemBulkQualStatus**

| Field Value | Value in DBQL XMLPLAN | Description   |
|-------------|-----------------------|---|
| 0           | NoBulk                | Bulk qualification is disabled.   |
| 1           | Bulk                  | Bulk qualification is enabled without a condition. For a JIN step, bulk qualification is enabled on both the right and left join tables.                                    |
| 2           | BulkWC                | Bulk qualification is enabled with a condition. For a JIN step, bulk qualification is enabled on both the right and left join tables.                                       |
| 3           | LeftBulk              | <ul style="list-style-type: none"> <li>• Left join table: Bulk qualification is enabled without a condition.</li> <li>• Right join table: No bulk qualification.</li> </ul> |
| 4           | LeftBulkWC            | <ul style="list-style-type: none"> <li>• Left join table: Bulk qualification is enabled with a condition.</li> <li>• Right join table: No bulk qualification.</li> </ul>    |
| 5           | RightBulk             | <ul style="list-style-type: none"> <li>• Left join table: No bulk qualification.</li> <li>• Right join table: Bulk qualification is enabled without a condition.</li> </ul> |
| 6           | RightBulkWC           | <ul style="list-style-type: none"> <li>• Left join table: No bulk qualification.</li> <li>• Right join table: Bulk qualification is enabled with a condition.</li> </ul>    |
| 7           | LeftBulkRightBulkWC   | <ul style="list-style-type: none"> <li>• Left join table: Bulk qualification is enabled without a condition.</li> </ul>   |

| Field Value | Value in DBQL XMLPLAN | Description  |
|-------------|-----------------------|--|
|             |                       | <ul style="list-style-type: none"> <li>Right join table: Bulk qualification is enabled with a condition.</li> </ul>  |
| 8           | LeftBulkWCRightBulk   | <ul style="list-style-type: none"> <li>Left join table: Bulk qualification is enabled with a condition.</li> <li>Right join table: Bulk qualification is enabled without a condition.</li> </ul> |

Field values 1 and 2 are applicable for Retrieve, Sum, and Join steps. In the case of a Join step, these values represent the bulk status of both the left and right source.

Field values 3 through 8 are applicable for only the Join step.

### Possible Values for NumCombinedPartitions

The value of the NumCombinedPartitions column is non-zero only if:

- There is static partition elimination for the step (for a query submitted to Teradata Database 14.0 or later, or Teradata Vantage Advanced SQL Engine 16.20 or later).
- A source table has column partitions.

Otherwise, the NumCombinedPartitions column is NULL.

### Possible Values for StepStatus

| Value      | Description                                     |
|------------|---|
| Dispatched | Step was dispatched to the AMP.                 |
| AMPExec    | Step was executed in the AMP.                   |
| AMPOkay    | Step completed okay in the AMP.                 |
| AMPerror   | Step failed in the AMP and returned an error.   |
| AMPReqAB   | Step aborted in the AMP for request abort.      |
| AMPTxnAB   | Step aborted in the AMP for transaction abort.  |
| Completed  | Step completed by the dispatcher.               |
| Skipped    | Conditional step was skipped by the dispatcher. |

### Possible Values for TriggerKind

- BegLoop
- FetchQualRows
- BldUsingRow
- GetActvCnt
- IdColWithRowTrig
- SkipQualRows

- EndLoop

## Step Descriptions

The steps reflected in the DBQLStepTbl are generated by the database to execute a query. They may not always coincide with steps seen in the Explain because some steps are for internal use only.

The following table lists the possible logged step names and their meanings.

| Step Internal Name | Description  |
|--------------------|--|
| 44-3               | Performance Monitor spoil step   |
| AAB                | Asynchronous abort of a DBC/SQL request  |
| ALT                | Insert target and originating table IDs into the HUT Control Segment for a Logon Sequence Number |
| BIL                | Begin isolated loading   |
| BLC                | Block-level compress/uncompress  |
| BMSMS              | Bit map set manipulations such as intersect and union  |
| CIX                | Create secondary index   |
| CkNPK              | Check N parent keys  |
| CKP                | Checkpoint database  |
| CRI                | Create reference index   |
| CSSUM              | Collect Statistics aggregate operations (this step is now obsolete)                              |
| CSUPD              | Collect Statistics update  |
| Ctb                | Create table header  |
| CTRts              | Create table privileges  |
| CWD                | Check workload step  |
| DBQLDC             | DBQL data collection   |
| DEL                | Delete row   |
| DELQT              | Delete row from queue table  |
| DIRI               | Drop inconsistent reference index  |
| DIX                | Delete secondary index   |
| DJT                | Delete journal table   |
| DRI                | Delete reference index   |
| DRIDR              | Delete RID Row   |

| Step Internal Name | Description  |
|--------------------|--|
| DSPRET             | Dispatcher retrieve step   |
| DTB                | Drop table   |
| DTmp               | Delete temporary tables  |
| EBD                | BEGIN EDIT DELETE  |
| EBE                | BEGIN MLOAD  |
| EDM                | Prepare for deblocker/application task (DML)   |
| EDR                | End request  |
| Edt                | End transaction  |
| EED                | Prepare MLOAD work tables  |
| EEN                | END edit step  |
| EFE                | End FastExport   |
| EIL                | End isolated loading   |
| EIXSUM             | Collect EIX statistics aggregate operations  |
| EIXUPD             | Collect EIX statistics update  |
| EPF                | End plan fragment; the presence of this step indicates the end of a plan fragment for a dynamic plan |
| ERE                | Release MLOAD work tables  |
| ESD                | Data acquisition   |
| ESR                | Sort MLOAD work tables   |
| EVT                | Create event table row   |
| EXE                | Execute edit   |
| ExecSP             | Execute Stored Procedure   |
| ExpHR              | Export Horizontal Redistribution step  |
| ExpRL              | Export Release Locks Step  |
| ExpVR              | Export Vertical Redistribution step  |
| Fail               | Operation did not work   |
| FCF                | Forward configuration  |
| FDS                | Flush DBSpace accounting table entry   |
| FLGRI              | Flag reference index   |

| Step Internal Name | Description  |
|--------------------|--|
| Hcs                | Add table record info and privileges into the Hut Control Segment for a Logon Sequence Number    |
| HLA                | High-Level Large Object Access   |
| HUL                | Host utility set lock  |
| ILR                | Contain access log entries   |
| INS                | Insert a row   |
| INSLDC             | Insert Deferred Lob Constant   |
| INSQT              | Insert a row to a queue table  |
| InvJHI             | Invalidate Join/Hash index   |
| JIN                | Join   |
| JTmp               | Journal temporary tables   |
| LBG                | Data-Load BEGIN LOADING  |
| LCM                | Load config map  |
| LCP                | Data-Load CHECKPOINT LOADING   |
| LFI                | Data-Load END LOADING  |
| LIN                | Data-Load USING ... INSERT   |
| LobFet             | Large object fetch   |
| LogAOf             | Logging for online archive off   |
| LogAOn             | Logging for online archive on  |
| LOT                | Large object transport   |
| LOT2VM             | Forward LOT to Virtual Mailbox   |
| MDT                | Merge delete tables  |
| Milns              | Merge-Into insert  |
| MiRet              | Merge-Into retrieve<br>There is a Merge-Into table reference in a RET (that is, retrieve) query. |
| MiUpd              | Merge-Into update  |
| MLK                | Multi-Lock: several LCK Messages in one  |
| MRD                | Merge delete two tables based on row hash code   |
| MRG                | Merge two tables based on row hash code  |
| MRL                | Merge utility locks  |

| Step Internal Name | Description   |
|--------------------|---|
| MRU                | Merge update  |
| MRM                | Merge Row Multiple Operations Step  |
| MTB                | Modify table. To modify the table internally, the table header must also be modified  |
| MTH                | Modify table header   |
| MTmp               | Materialize temporary tables  |
| MVN                | Modify version number in table header   |
| NEXTOK             | It is okay to send the next step  |
| OAR                | One AMP reply   |
| Okay               | Operation worked  |
| OKELCT             | Send StpOKElicitData message to notify Dispatcher that it is fine to send next step within the same request. It is required to prevent deadlocks and serve as a form of flow control. |
| PFN                | Process function  |
| PKA                | Prime key abort   |
| PRELOB             | InsPreLob step  |
| QryLgn             | Query logon on step   |
| RAE                | Remove abort entry from TOABORT list  |
| RET                | Retrieve row  |
| RSF                | Release spool files   |
| SAMP               | Perform sampling from table/spool   |
| SAT                | Synchronous abort test step from DBC/SQL statement  |
| SMS                | Set manipulations such as intersect, union, minus   |
| Spl004             | Dictionary data spoil step for role   |
| Spl011             | Dictionary data spoil step for security constraint  |
| Spl012             | Stats cache spoil step  |
| SplIDB             | Spoil database information in Data Dictionary   |
| SplIFL             | DBQL cache flush spoil step   |
| SplIOUC            | Object use count spoil step   |
| SplIPSC            | Spoil Parser session information  |
| SplIQL             | Spoil the DBQL Rule cache   |

| Step Internal Name | Description                                |
|--------------------|--|
| SplRTS             | Spoil Stored Procedure cache information   |
| SplTAB             | Spoil Table information in Data Dictionary |
| SQB                | Set Query_Band                             |
| SQL                | SQL step for Request Text storage          |
| SRD                | Sort a table                               |
| STATFN             | Perform statistical functions              |
| StpRLK             | Release lock                               |
| SUM                | Perform local aggregate operations         |
| TRET               | Trigger retrieve step                      |
| UPD                | Update row                                 |
| UpsIns             | Upsert insert                              |
| UpsUpd             | Upsert update                              |
| VJT                | Validate journal table                     |
| VRQ                | 2PC vote step                              |
| Warn               | Warning response                           |
| YCM                | Ynet conclude merge                        |
| YPM                | Ynet prepare merge                         |

## Example: Using QryLogStepsV

The following SELECT statement retrieves one sample row from the QryLogStepsV view:

```
select * from QryLogStepsV sample 1;
```

The query returns the following result:

```

ProcID    30718
CollectTimeStamp 2016-11-04 12:11:52.250000
QueryID    307180509041677206
StepLev1Num      5
StepLev2Num      7
StepName  CTRts
StepStartTime 2016-11-04 12:11:52.130000
StepStopTime  2016-11-04 12:11:52.130000

```

|                       |                |       |
|-----------------------|----------------|-------|
| ElapsedTime           | 0:00:00.000000 |       |
| EstProcTime           |                | 0.000 |
| EstCPUTime            | 0.000          |       |
| CPUTime               | 0.000          |       |
| IOcount               |                | 0     |
| EstRowCount           |                | 0     |
| EstRowCountSkew       |                | ?     |
| EstRowCountSkewMatch  |                | ?     |
| RowCount              |                | 1     |
| RowCount2             |                | ?     |
| RowCount3             |                | ?     |
| NumOfActiveAMPs       | 1              |       |
| MaxAmpCPUTime         | 0.000          |       |
| MaxCPUAmpNumber       | ?              |       |
| MinAmpCPUTime         | 0.000          |       |
| MaxAmpIO              |                | 14    |
| MaxIOAmpNumber        | 1              |       |
| MinAmpIO              |                | 0     |
| SpoolUsage            |                | 0     |
| MaxAMPSpool           |                | 0     |
| MaxSpoolAmpNumber     | ?              |       |
| MinAMPSpool           |                | 0     |
| StepWD                | 13             |       |
| LSN                   | ?              |       |
| UtilityTableID        | ?              |       |
| RowsWComprColumns     |                | 0     |
| EstIOWCost            | 0.000          |       |
| EstNetCost            | 0.000          |       |
| EstHRCost             | 0.000          |       |
| CPUTimeNorm           | 0.000          |       |
| MaxAmpCPUTimeNorm     | 0.000          |       |
| MaxCPUAmpNumberNorm   | ?              |       |
| MinAmpCPUTimeNorm     | 0.000          |       |
| NumCombinedPartitions |                | ?     |
| NumContexts           | ?              |       |

## Example: Using QryLogStepsV with ServerByteCount

The following SELECT statement gives the user name and the elapsed time of the steps whose queries have transferred more than 10 MB of data.

```
SELECT lv.username, sv.elapsedtime FROM DBC.QryLogStepsV AS sv,
DBC.QryLogV AS lv WHERE ServerByteCount / (1024*1024) GT 10 AND
sv.queryid = lv.queryid;
```

The query returns the following result:

```
username TOM
ElapsedTime 0:10:22.220000
username JOHN
ElapsedTime 0:21:32.510000
```

## QryLogSummaryV

**Category:** Query

**Database:** DBC

| View Column      | Data Type                                | Format                   | Comment  |
|------------------|--|--------------------------|--|
| ProcID           | DECIMAL(5,0)<br>NOT NULL                 | -(5)9                    | Returns the process ID of the dispatcher.  |
| CollectTimeStamp | TIMESTAMP(6)<br>NOT NULL                 | YYYY-MM-DDBHH:MI:SS.S(6) | (Prime Key) Time and date when the DBQL summary cache was written.   |
| UserID           | BYTE(4)                                  | X(8)                     | Returns the ID of the user.  |
| AcctString       | VARCHAR(128)<br>UNICODE NOT CASESPECIFIC | X(128)                   | Returns the user unexpanded logon account string.  |
| LogicalHostID    | SMALLINT                                 | -(5)9                    | Returns a unique identifier of the logon source for the logged query. A value of zero indicates an internal session. |
| AppID            | CHAR(30)<br>UNICODE NOT CASESPECIFIC     | X(30)                    | Returns the application ID.  |
| ClientID         | CHAR(30)<br>UNICODE NOT CASESPECIFIC     | X(30)                    | Returns the client ID.   |
| ClientAddr       | CHAR(45) LATIN<br>NOT CASESPECIFIC       | X(45)                    | Returns the IP address of the client who submitted the query.  |
| ProfileID        | BYTE(4)                                  | X(8)                     | Returns the unique number assigned to the cost profile instance in the system.                                       |

| View Column          | Data Type                            | Format                      | Comment   |
|----------------------|--------------------------------------|-----------------------------|---|
| SessionID            | INTEGER<br>NOT NULL                  | --,---,---,--9              | Returns the session identifier.   |
| QueryCount           | INTEGER<br>NOT NULL                  | --,---,---,--9              | Returns the number of queries run in a 10-minute interval. Used with the "SUMMARY" or "THRESHOLD" SQL option only.                                |
| ValueType            | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC | X(1)                        | Returns the type that was used to determine the threshold from the values listed below. C = CPU seconds I = IO count S = Elapsed time in seconds. |
| QuerySeconds         | FLOAT NOT NULL                       | ---,---,---,---,<br>--9.999 | Returns the total number of seconds used by QueryCount. QuerySeconds can be used to obtain an average.  |
| AverageTime          | FLOAT                                | ZZZ,ZZZ.999                 | Returns the average elapsed time (to the nearest hundredth of a second).  |
| TotalIOCount         | FLOAT                                | ---,---,---,---,<br>--9     | Returns the number of IOs from AMPs that were generated by the query.   |
| AverageIO            | FLOAT                                | Z(10)                       | Returns the average Logical IO Count.   |
| AMPCPUTime           | FLOAT                                | ZZZ,ZZZ.999                 | Returns the total AMP CPU time in seconds used for query.   |
| AverageAmpCPU        | FLOAT                                | ZZZ,ZZZ.999                 | Returns the average number of AMP CPU seconds used by QueryCount.   |
| ParserCPUTime        | FLOAT                                | ZZZ,ZZZ.999                 | Returns the total parser and dispatcher CPU time in seconds used for the query.   |
| AverageParserCPU     | FLOAT                                | ZZZ,ZZZ.999                 | Returns the average number of CPU seconds used in the parser by QueryCount.   |
| AMPCPUTimeNorm       | FLOAT                                | ZZZ,ZZZ.999                 | Returns the normalized AMP CPU time in seconds for co-existence systems.  |
| AverageAmpCPUNorm    | FLOAT                                | ZZZ,ZZZ.999                 | Average number of normalized AMP CPU seconds used by QueryCount.  |
| ParserCPUTimeNorm    | FLOAT                                | ZZZ,ZZZ.999                 | Returns the normalized parser CPU time in seconds for co-existence systems.   |
| AverageParserCPUNorm | FLOAT                                | ZZZ,ZZZ.999                 | Average number of normalized CPU seconds used in the parser by QueryCount.  |

| View Column | Data Type                                   | Format                       | Comment  |
|-------------|---|------------------------------|--|
| LowHist     | FLOAT NOT NULL                              | ---,---,---,---,<br>--9.999  | For SUMMARY option, lowest value specified as query execution differentiation. For THRESHOLD option, threshold specified by the user.                        |
| HighHist    | FLOAT NOT NULL                              | ---,---,---,---,<br>--9.999  | Returns the highest value specified as a query execution time differentiation. Used with the "SUMMARY" SQL option only. If THRESHOLD is used, HighHist is 0. |
| UsedIota    | FLOAT                                       | ---,---,---,---,<br>--9.999  | Return IO Tokens used by the requests that are logged in summary mode.   |
| ReqPhysIO   | FLOAT                                       | ---,---,---,---,<br>--9.999  | Returns the total number of Physical I/Os for the whole request in summary mode.   |
| ReqPhysIOKB | FLOAT                                       | ---,---,---,---,<br>--9.999  | Returns the total Physical I/Os in kilobytes for the whole request in summary mode.  |
| StartTime   | TIMESTAMP(6)                                | YYYY-MM-DD<br>BHH:MI:SS.S(6) | Records the timestamp of the first query included for this summary row.  |
| StopTime    | TIMESTAMP(6)                                | YYYY-MM-DD<br>BHH:MI:SS.S(6) | Record the timestamp of the last query included for this summary row.  |
| UserName    | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128)                       | Returns the name of the userid, if any, under which the user submitted the query.  |
| ProfileName | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128)                       | Returns the name of the profile, if any, under which the user submitted the query.   |

## Usage Notes

The QryLogSummaryV view of DBQLSummaryTbl holds the counts logged for queries of users with THRESHOLD or SUMMARY rules. It is only populated if summary information is requested or a threshold value is used.

### Possible Values for Value Type

The THRESHOLD criterion is as follows:

| Value | Description                     |
|-------|---------------------------------|
| C     | AMP CPU seconds.                |
| H     | Elapsed hundredths of a second. |
| I     | I/O count.                      |
| N     | Normalized AMP CPU seconds.     |
| S     | Elapsed time in seconds.        |

### LowHist and HighHist

| Value    | Description   |
|----------|---|
| HighHist | <ul style="list-style-type: none"> <li>Highest value for the SUMMARY option.</li> <li>Zero for the THRESHOLD option.</li> <li>If the modifiers CPUTIME or IOCOUNT are used, then this value is the highest CPU seconds or I/O counts set by the user.</li> </ul> <p>Use CPUTIMENORM for hundredths of a second of normalized CPU time. To measure subsecond queries, use ELAPSEDTIME. The default, elapsed time in seconds, can be explicitly specified with ELAPSEDSEC.</p>  |
| LowHist  | <ul style="list-style-type: none"> <li>Lowest value for the SUMMARY option.</li> <li>For the THRESHOLD option, this is the threshold time set by the user. The default time is 5.</li> <li>If the modifiers CPUTIME or IOCOUNT are used, then this value is the lowest CPU seconds or I/O counts set by the user. Use CPUTIMENORM for hundredths of a second of normalized CPU time. To measure subsecond queries, use ELAPSEDTIME. The default, elapsed time in seconds, can be explicitly specified with ELAPSEDSEC.</li> </ul> |

For the SUMMARY case, a row exists for each collection if the count of the queries run in that time frame (LowHist to HighHist) is greater than 0.

For the THRESHOLD case, one row (with LowHist=TypeofCriterion and HighHist=0) exists for the collection if there were queries shorter than LowHist seconds.

### Rows Generated by the THRESHOLD Option

If a threshold value is used, the DBQLSummaryTbl will log queries based on the criterion of elapsed seconds, elapsed hundredths, normalized time, logical I/O count, or AMP CPU hundredths of a second. The ValueType column in DBQLSummaryTbl tells which criterion are used.

Queries with values less than or equal to the given limit will be counted in the DBQLSummaryTbl. Queries with more than the given limit will be logged in DBQLLogTbl.

For example, if the criterion is elapsed seconds, for each session being logged, DBQL with the THRESHOLD option does the following:

- Each time a query completes within the threshold time, increments the counters for the session

- Every 10 minutes, writes the cumulative count for each session as a separate row in DBQLSummaryTbl
- DBQL generates a default row in DBQLLogTbl for queries that exceed the threshold time

For example, if you specified “LIMIT THRESHOLD=500 CPUTIME ON user1” you could find out how many queries user1 submitted that required more than 5 CPU seconds.

The following is an example of how to query the DBQLSummaryTbl table, using the QryLogSummary[V] view, to find the total I/O count and total CPU time:

```
SELECT collecttimestamp, sessionid, querycount, ampcputime, totaliocomount
FROM QryLogSummaryV ORDER BY collecttimestamp;
```

| CollectTimeStamp    | SessionID | QueryCount | AMPCPUTime | TotalIOCount |
|---------------------|-----------|------------|------------|--------------|
| -----               | -----     | -----      | -----      | -----        |
| 2006-07-29 10:30:05 | 1,001     | 5          | .031       | 4            |
| 2006-07-29 11:07:10 | 1,001     | 1          | .015       | 0            |
| 2006-07-29 11:07:10 | 1,013     | 1          | .000       | 0            |
| 2006-07-29 11:07:10 | 1,000     | 2          | .047       | 0            |
| 2006-07-29 11:07:10 | 1,014     | 71         | .907       | 2,427        |
| 2006-07-29 11:17:10 | 1,017     | 96         | 1.234      | 2,983        |
| 2006-07-29 11:17:10 | 1,014     | 26         | .329       | 552          |
| 2006-07-29 11:17:10 | 1,031     | 1          | .016       | 0            |
| 2006-07-29 11:17:10 | 1,023     | 94         | 1.093      | 2,483        |
| 2006-07-29 11:17:10 | 1,026     | 42         | .578       | 1,196        |

## Rows Generated by the SUMMARY Option

DBQL behavior resulting from the SUMMARY option is unique in that:

- No default rows are generated to DBQLLogTbl for summarized queries.
- You define the summary criteria and then DBQL summarizes queries into those buckets. Each bucket counts the number of queries for the session that fall into that bucket and also sums up their elapsed time, I/O count and various CPU times.
- DBQL maintains summary counters in cache. The contents are committed to the DBQLSummaryTbl table every 10 minutes, when the cache is flushed.

The following table describes the summary criteria that make up the buckets.

| Using the following modifier on the SUMMARY Option... | Means the system logs the values specified as...   |
|---|--|
| CPUTIME   | hundredths of a second of AMP CPU time. So for example:<br>BEGIN QUERY LOGGING LIMIT SUMMARY=5, 10, 15 CPUTIME ON user1; |

| Using the following modifier on the SUMMARY Option... | Means the system logs the values specified as...   |
|---|--|
|   | means counting queries based on CPU time where 5 is 0.05 seconds of CPU time, 10 is 0.10 seconds and 15 is 0.15 seconds of CPU time.   |
| CPUTIMENORM   | hundredths of a second of normalized AMP CPU time. This is similar to CPUTIME above, but uses the times from a co-existence system that are normalized.  |
| ELAPSEDSEC  | Without an modifier to the SUMMARY option:<br>BEGIN QUERY LOGGING LIMIT SUMMARY=5, 10, 15 ON user1;<br>elapsed seconds is used. You can explicitly set elapsed seconds as a criteria by using the following statement:<br>BEGIN QUERY LOGGING LIMIT SUMMARY=5, 10, 15 ELAPSEDSEC ON user1; |
| ELAPSEDTIME   | elapsed time where n is specified in hundredths of a second.   |
| IOCOUNT   | the number of logical I/Os used. For example:<br>BEGIN QUERY LOGGING LIMIT SUMMARY=1000,5000,10000 IOCOUNT ON ALL;   |

DBQL gathers information for the collection interval (the default is every 10 minutes). For every collection interval and every active session at the end of the interval, there is as many as four rows per session containing the query count, the number of seconds of elapsed time for those queries, the amount of CPU time and the number of logical I/Os. Each row represents one of four possible buckets of information.

In the default case, where elapsed time is used to qualify a query for one of the four buckets, the following buckets are possible:

- 0 to 5 seconds
- Longer than 5 seconds to 10 seconds
- Longer than 10 seconds to 15 seconds
- Longer than 15 seconds

For example, if during that next logging period for the user, two queries ran under 5 seconds, three queries ran for 7 seconds, and one query ran for over 15 seconds, the following rows would be written to DBQLSummaryTbl for the session:

| COUNT | SECONDS | LOWHIST | HIGHHIST   |
|-------|---------|---------|------------|
| ----  | -----   | -----   | -----      |
| 2     | 1       | 0       | 5          |
| 3     | 21      | 5       | 10         |
| 1     | 200     | 15      | 4294967295 |

For this example, there were:

- No queries between 10 and 15 seconds

- To determine the average time for each query counted, divide SECONDS by COUNT (for example, the two queries in the first row averaged 0.5 seconds each; the three queries in the second row averaged 7 seconds each).

If you specified LIMIT THRESHOLD=500 CPUTIME ON user1, you could find out how many queries user1 submitted that required more than 5 CPU seconds.

### Example: Using QryLogSummaryV

The following SELECT statement retrieves the summary information of a session (if logging is initiated with “Replace Query Logging limit Summary = 10,15,20 IOCOUNT on all;”):

```
replace query logging limit summary = 10,15,20 IOCOUNT on all;
```

To see the rules:

```
select * from dbqlrulesv;
```

Result:

|                 |                 |     |
|-----------------|-----------------|-----|
| UserName        | All             |     |
| Account         |                 |     |
| ApplicationName |                 |     |
| TypeOfRule      | Logging enabled |     |
| Explain         | F               |     |
| Object          | F               |     |
| SQL             | F               |     |
| Step            | F               |     |
| XMLPlan         | F               |     |
| StatsUsage      | F               |     |
| Verbose         | F               |     |
| DetailedStats   | F               |     |
| NoColumns       | F               |     |
| Summary         | T               |     |
| Threshold       | F               |     |
| ObjectUsage     | F               |     |
| Param           | F               |     |
| FeatureUsage    | F               |     |
| UtilityInfo     | F               |     |
| TextSize        |                 | 200 |
| Summary//Low    | 10              |     |
| Med             | 15              |     |
| High            | 20              |     |
| LockDelay       |                 | 0   |

```

AlgMode      ?
TypeOfCriterion IOCount
DetailDiag           ?

```

The following SELECT statement retrieves the summary information of a session:

```
select * from QrylogSummaryV sample 1;
```

Result:

```

ProcID      30718
CollectTimeStamp 2016-11-04 12:45:35.460000
UserID      00000100
AcctString   DBC
LogicalHostID      1
AppID      DBCCONS
ClientID     PTEUSER
ClientAddr   10.25.176.76
ProfileID    ?
SessionID           1,386
QueryCount           1
ValueType      I
QuerySeconds           0.000
AverageTime           .000
TotalIOCount           0
AverageIO
AMPCPUTime           .000
AverageAmpCPU         .000
ParserCPUTime         .004
AverageParserCPU      .004
AMPCPUTimeNorm        .000
AverageAmpCPUNorm     .000
ParserCPUTimeNorm     .253
AverageParserCPUNorm  .253
LowHist           0.000
HighHist          10.000
UsedIota           0.000
ReqPhysIO          0.000
ReqPhysIOKB        0.000
StartTime 2016-11-04 12:38:20.390000
StopTime  2016-11-04 12:38:20.390000
UserName   DBC

```

# QryLogTdwmSumV

**Category:** Query

**Database:** DBC

| View Column      | Data Type                | Format                       | Comment  |
|------------------|--------------------------|------------------------------|--|
| ProcID           | DECIMAL(5,0)<br>NOT NULL | -(5)9                        | Returns the process ID of the dispatcher.  |
| CollectTimeStamp | TIMESTAMP(6)<br>NOT NULL | YYYY-MM-DDBHH:MI:SS.<br>S(6) | (Prime Key) Time and date when the TDWM summary cache was written.   |
| WDID             | INTEGER<br>NOT NULL      | --,---,---,--9               | Returns the workload definition under which the query was classified.  |
| OpEnvID          | INTEGER                  | --,---,---,--9               | Returns the ID of the operational environment currently activated by TDWM.   |
| SysConID         | INTEGER                  | --,---,---,--9               | Returns the ID of the system condition currently activated by TDWM.  |
| StartColTime     | TIMESTAMP(6)<br>NOT NULL | YYYY-MM-DDBHH:MI:SS.<br>S(6) | Indicates the time at the start of the summary interval.   |
| Arrivals         | INTEGER<br>NOT NULL      | --,---,---,--9               | Indicates how many queries were classified in the summary interval for this workload definition.   |
| ActiveCount      | INTEGER<br>NOT NULL      | --,---,---,--9               | Returns the number of queries that are active in the WD in the collection period.  |
| Completions      | INTEGER<br>NOT NULL      | --,---,---,--9               | The number of queries that completed in this WD in this collection period.   |
| MinRespTime      | FLOAT NOT NULL           | ZZZZZZ.99                    | Returns the fastest response time in centiseconds of all queries completed in the summary interval for the workload definition.            |
| MaxRespTime      | FLOAT NOT NULL           | ZZZZZZ.99                    | Returns the slowest response time in centiseconds of all queries completed in the summary interval for the workload definition.            |
| AvgRespTime      | FLOAT NOT NULL           | ZZZZZZ.99                    | Returns the average response time in centiseconds for all queries completed in the summary interval for the workload definition.           |
| MinCPUTime       | FLOAT NOT NULL           | Z,ZZZ,ZZZ,<br>ZZ9.99         | Returns the least CPU time in seconds (with .001 resolution) of all queries completed in the summary interval for the workload definition. |

| View Column      | Data Type        | Format               | Comment   |
|------------------|------------------|----------------------|---|
| MaxCPUTime       | FLOAT NOT NULL   | Z,ZZZ,ZZZ,<br>ZZ9.99 | Returns the most CPU time in seconds (with .001 resolution) of all queries completed in the summary interval for the workload definition.             |
| AvgCPUTime       | FLOAT NOT NULL   | Z,ZZZ,ZZZ,<br>ZZ9.99 | Returns the average CPU time in seconds (with .001 resolution) for all queries completed in the summary interval for the workload definition.         |
| DelayedCount     | INTEGER NOT NULL | --,---,---,--9       | Indicates how many of all the queries that were completed in the summary interval for the workload definition were.                                   |
| AvgDelayTime     | FLOAT NOT NULL   | ---,---,---,---,--9  | Returns the average delay time in centiseconds for all queries completed in the summary interval for the workload definition.                         |
| ExceptionAbCount | INTEGER NOT NULL | --,---,---,--9       | Returns the number of queries that were aborted due to an exception action for this workload definition during this interval.                         |
| ExceptionMvCount | INTEGER NOT NULL | --,---,---,--9       | Returns the number of queries that moved to another workload definition due to an exception action for this workload definition during this interval. |
| ExceptionCoCount | INTEGER NOT NULL | --,---,---,--9       | Returns the number of queries that encountered an exception but continued in this workload definition during this interval.                           |
| ExceptionCount   | INTEGER          | -(10)9               | Returns the number of queries with exceptions in this workload definition in this collection period.  |
| MetSLGCount      | INTEGER NOT NULL | --,---,---,--9       | Indicates how many of all the queries completed in the summary interval for the workload definition met their SLG objectives.                         |
| AbortCount       | INTEGER NOT NULL | --,---,---,--9       | Returns the count of queries that were classified into the WD but were subsequently aborted (no DBS error code).                                      |
| ErrorCount       | INTEGER NOT NULL | --,---,---,--9       | Returns the count of queries that completed with an error (error code) in the collection period for this WD.  |
| RejectedCount    | INTEGER NOT NULL | --,---,---,--9       | Returns the number of queries that were rejected by TDWM for this workload definition during this interval.   |
| MovedInCount     | INTEGER          | --,---,---,--9       | Returns the number of queries that were moved into this workload definition due to an exception during this interval.                                 |

| View Column      | Data Type | Format              | Comment  |
|------------------|-----------|---------------------|--|
| IntervalDelayCnt | INTEGER   | --,---,---,--9      | Returns the number of queries that are currently delayed during this interval.   |
| DelayedQueries   | INTEGER   | --,---,---,--9      | Returns the number of queries that are delayed for this workload definition.   |
| OtherCount       | INTEGER   | --,---,---,--9      | The number of queries that finished in this WD that had no error or abort, AMPs or Parser CPU.                         |
| VirtualPartNum   | INTEGER   | --,---,---,--9      | The Virtual Partition number of the WD.  |
| AvgIOWaitTime    | FLOAT     | ---,---,---,---,--9 | The average IO wait time in milliseconds for those queries completed in this WD in this collection period.             |
| MaxIOWaitTime    | FLOAT     | ---,---,---,---,--9 | The maximum IO wait time in milliseconds for those queries completed in this WD in this collection period.             |
| AvgOtherWaitTime | FLOAT     | ---,---,---,---,--9 | The average other wait time in seconds for those queries completed in this WD in this collection period.               |
| MaxOtherWaitTime | FLOAT     | ---,---,---,---,--9 | The maximum other wait time in seconds for those queries completed in this WD in this collection period.               |
| AvgCPURunDelay   | FLOAT     | ---,---,---,---,--9 | The average CPU run delay in milliseconds for those queries completed in this WD in this collection period.            |
| MaxCPURunDelay   | FLOAT     | ---,---,---,---,--9 | The maximum CPU run delay in milliseconds for those queries completed in this WD in this collection period.            |
| AvgSeqRespTime   | FLOAT     | ---,---,---,---,--9 | The average sequential response time in milliseconds for those queries completed in this WD in this collection period. |
| MaxSeqRespTime   | FLOAT     | ---,---,---,---,--9 | The maximum sequential response time in milliseconds for those queries completed in this WD in this collection period. |
| AvgLogicalIO     | FLOAT     | ---,---,---,---,--9 | The average logical IO for those queries completed in this WD in this collection period.                               |
| MaxLogicalIO     | FLOAT     | ---,---,---,---,--9 | The maximum logical IO for those queries completed in this WD in this collection period.                               |
| AvgLogicalKBs    | FLOAT     | ---,---,---,---,--9 | The average logical KBs for those queries completed in this WD in this collection period.                              |

| View Column      | Data Type           | Format              | Comment  |
|------------------|---------------------|---------------------|--|
| MaxLogicalKBs    | FLOAT               | ---,---,---,---,--9 | The maximum logical KBs for those queries completed in this WD in this collection period.                      |
| AvgPhysicalIO    | FLOAT               | ---,---,---,---,--9 | The average physical IO for those queries completed in this WD in this collection period.                      |
| MaxPhysicalIO    | FLOAT               | ---,---,---,---,--9 | The maximum physical IO for those queries completed in this WD in this collection period.                      |
| AvgPhysicalKBs   | FLOAT               | ---,---,---,---,--9 | The average physical KBs for those queries completed in this WD in this collection period.                     |
| MaxPhysicalKBs   | FLOAT               | ---,---,---,---,--9 | The maximum physical KBs for those queries completed in this WD in this collection period.                     |
| ThrottleBypassed | INTEGER             | --,---,---,--9      | The number of queries that are only running due to the ThrottleBypass feature.                                 |
| FlexActive       | INTEGER<br>NOT NULL | --,---,---,--9      | The number of queries that are only running due to the Flex Throttle feature.                                  |
| FlexComplete     | INTEGER<br>NOT NULL | --,---,---,--9      | The number of queries that completed due to the Flex Throttle feature.   |
| FlexArrivals     | INTEGER<br>NOT NULL | --,---,---,--9      | The number of queries that arrived due to Flex during this period  |
| DeferredCount    | INTEGER<br>NOT NULL | --,---,---,--9      | The number of queries in this WD which have been deferred due to an ARM rule in this collection period.        |
| DeferredQueries  | INTEGER<br>NOT NULL | --,---,---,--9      | The number of queries in this WD that were currently deferred due to an ARM rule when this data was collected. |
| AvgDeferTime     | FLOAT NOT NULL      | ---,---,---,---,--9 | The average defer time due to ARM rules in centiseconds for queries in this WD in this collection period.      |

## Usage Notes

This view contains a historical record of WD activity.

### Abort, Completions, and ErrorCount

For these columns, dashboard interval is used for the API data and logging interval is used for logging data to the disk.

## MetSLGCount

For this column, queries completing in a WD that do not have a declared Response Time Service Level Goal (SLG) are also counted as meeting the SLG.

## Example: Select from DBC.QryLogTDWMSum

The following SELECT statement retrieves the QryLogTDWMSum view:

```
SELECT * from DBC.QryLogTDWMSum;
```

The query returns the following result:

|                  |                            |
|------------------|----------------------------|
| ProcID           | 30718                      |
| CollectTimeStamp | 2013-12-16 16:52:33        |
| WDID             | 12                         |
| OpEnvID          | 1                          |
| SysConID         | 1                          |
| StartColTime     | 2013-12-16 16:52:33.130000 |
| Arrivals         | 4                          |
| ActiveCount      | 0                          |
| Completions      | 4                          |
| MinRespTime      | .01                        |
| MaxRespTime      | .13                        |
| AvgRespTime      | .05                        |
| MinCPUTime       | .27                        |
| MaxCPUTime       | 4.04                       |
| AvgCPUTime       | 1.61                       |
| DelayedCount     | 0                          |
| AvgDelayTime     | 0                          |
| ExceptionAbCount | 0                          |
| ExceptionMvCount | 0                          |
| ExceptionCoCount | 0                          |
| ExceptionCount   | 0                          |
| MetSLGCount      | 4                          |
| AbortCount       | 0                          |
| ErrorCount       | 0                          |
| RejectedCount    | 0                          |
| MovedInCount     | 0                          |
| IntervalDelayCnt | 0                          |
| DelayedQueries   | 0                          |
| OtherCount       | 0                          |
| VirtualPartNum   | 1                          |
| AvgIOWaitTime    | 0                          |

|                  |        |
|------------------|--------|
| MaxIOWaitTime    | 0      |
| AvgOtherWaitTime | 1      |
| MaxOtherWaitTime | 3      |
| AvgCPURunDelay   | 0      |
| MaxCPURunDelay   | 0      |
| AvgSeqRespTime   | 0      |
| MaxSeqRespTime   | 0      |
| AvgLogicalIO     | 125    |
| MaxLogicalIO     | 259    |
| AvgLogicalKBs    | 5,705  |
| MaxLogicalKBs    | 12,640 |
| AvgPhysicalIO    | 2      |
| MaxPhysicalIO    | 6      |
| AvgPhysicalKBs   | 62     |
| MaxPhysicalKBs   | 248    |
| ThrottleBypassed | 0      |

## QryLogTDWMV

**Category:** Query

**Database:** DBC

| View Column      | Data Type                                | Format                   | Comment  |
|------------------|--|--------------------------|--|
| ProcID           | DECIMAL(5,0)<br>NOT NULL                 | -(5)9                    | (Prime Key) ID of the Processing Engine from which the query was executed. |
| CollectTimeStamp | TIMESTAMP(6)<br>NOT NULL                 | YYYY-MM-DDBHH:MI:SS.S(6) | (Prime Key) Time and date when the DBQLog cache was allocated.             |
| QueryID          | DECIMAL(18,0)<br>NOT NULL                | --Z(17)9                 | (Foreign Key) Systemwide unique value to join DBQL tables.                 |
| UserID           | BYTE(4) NOT NULL                         | X(8)                     | Identifier for the user that issued the query.                             |
| UserName         | VARCHAR(128)<br>UNICODE NOT CASESPECIFIC | X(128)                   | Returns the name of the user who issued the query.                         |
| DefaultDatabase  | VARCHAR(128)<br>UNICODE NOT CASESPECIFIC | X(128)                   | Default Database name for the user that issued the query.                  |
| AcctString       | VARCHAR(128)<br>UNICODE NOT CASESPECIFIC | X(128)                   | Unexpanded Account String under which the user issued the query.           |

| View Column        | Data Type                | Format                   | Comment  |
|--------------------|--------------------------|--------------------------|--|
| SessionID          | INTEGER<br>NOT NULL      | --,---,---,--9           | SessionID of the query.  |
| LogicalHostID      | SMALLINT<br>NOT NULL     | -(5)9                    | Logical Host ID from which query was executed.   |
| RequestNum         | INTEGER<br>NOT NULL      | --,---,---,--9           | Host Request number for this query within the session.   |
| InternalRequestNum | INTEGER<br>NOT NULL      | --,---,---,--9           | Internal Request number when Stored Procedures are used within the session.                              |
| StartTime          | TIMESTAMP(6)<br>NOT NULL | YYYY-MM-DDBHH:MI:SS.S(6) | Time and date when the query was submitted to the system.  |
| TDWMAdmissionTime  | TIMESTAMP(6)             | YYYY-MM-DDBHH:MI:SS.S(6) | The time when a request is admitted into the system by workload management (i.e., after ARM processing). |
| FirstStepTime      | TIMESTAMP(6)<br>NOT NULL | YYYY-MM-DDBHH:MI:SS.S(6) | Returns the time the first step for this query is dispatched.  |
| FirstRespTime      | TIMESTAMP(6)             | YYYY-MM-DDBHH:MI:SS.S(6) | Time first response was sent to the host.  |
| LastStateChange    | TIMESTAMP(6)             | YYYY-MM-DDBHH:MI:SS.S(6) | Time of the last state change that occurred.   |
| DeferTime          | FLOAT                    | ZZ,ZZZ,ZZ9.999           | The number of seconds a query was deferred by workload management due to ARM rules.                      |
| DeferRuleID        | INTEGER                  | --,---,---,--9           | TDWM Rule ID of one of the ARM rules which caused this request to be deferred.                           |
| DelayTime          | FLOAT                    | ZZ,ZZZ,ZZ9.999           | Number of seconds the query was delayed by TDWM.   |
| TDWMRuleID         | INTEGER                  | --,---,---,--9           | RuleID of the TDWM rule causing an object delay.   |
| SessionWDID        | INTEGER                  | --,---,---,--9           | Workload Definition under which the query was parsed.  |
| WDID               | INTEGER                  | --,---,---,--9           | Workload Definition assigned to the query by TDWM.   |
| FinalWDID          | INTEGER                  | --,---,---,--9           | Workload Definition under which the query completed execution.   |

| View Column        | Data Type                                   | Format                       | Comment  |
|--------------------|---|------------------------------|--|
| OpEnvID            | INTEGER                                     | --,---,---,--9               | Operational Environment Id in which the query was executed.                                |
| SysConID           | INTEGER                                     | --,---,---,--9               | System Condition Id in which the query was executed.                                       |
| LSN                | INTEGER                                     | --,---,---,--9               | Logon Sequence Number used by the load utility.  |
| NoClassification   | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC        | X(1)                         | T if the query was not classified by TDWM.   |
| WDOVERRIDE         | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC        | X(1)                         | T if the Workload Definition was overridden by the user.                                   |
| ResponseTimeMet    | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC        | X(1)                         | T if the query met the service level goal of the workload definition.                      |
| ExceptionValue     | INTEGER                                     | --,---,---,--9               | Type of TDWM exception: e.g. 01-time, 02-CPU, 03-AMP CPU skew, etc.                        |
| TDWMEstMaxRows     | FLOAT                                       | ----,---,---,---,<br>--9     | Maximum estimated row count used to classify a query into a WD                             |
| TDWMEstLastRows    | FLOAT                                       | ----,---,---,---,<br>--9     | Last estimated row count made by the optimizer for WD classification.                      |
| TDWMEstTotalTime   | FLOAT                                       | ZZZZ9.999999                 | Total time in seconds estimated by the optimizer used to classify a query into a WD        |
| TDWMEstMemUsage    | FLOAT                                       | ----,---,---,---,<br>--9.999 | Estimated memory in MBs used to classify a query into a WD                                 |
| TDWMAllAmpFlag     | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC        | X(1)                         | T if all amp query. Used to classify the query to a WD                                     |
| TDWMConfLevelUsed  | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC        | X(1)                         | Confidence level used to determine what statistics to use to classify the query into a WD. |
| StatementGroup     | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128)                       | Description of statement(s) issued by the multi-statement or single request.               |
| MinRespHoldTime    | FLOAT                                       | Z,ZZ9.9                      | Seconds response was held to meet the minimum response time.                               |
| TotalFirstRespTime | FLOAT                                       | ZZ,ZZZ,ZZ9.<br>999           | DelayTime + Execution Time + MinRespHoldTime (in seconds)                                  |

| View Column      | Data Type                                   | Format         | Comment   |
|------------------|---|----------------|---|
| MaxNumMapAMPs    | INTEGER                                     | --,---,---,--9 | Returns the number of AMPs in the largest contiguous map used by the request. |
| MinNumMapAMPs    | INTEGER                                     | --,---,---,--9 | The number of AMPs in the smallest contiguous map used by this request.       |
| SysDefNumMapAMPs | INTEGER                                     | --,---,---,--9 | Number of AMPs in the system-default map.                                     |
| ProfileName      | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128)         | Returns Profile name of the user who submitted the query.                     |
| WdName           | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128)         | Returns Workload name assigned to the query.                                  |
| FinalWdName      | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128)         | Returns the workload definition in which the query completed execution.       |
| SessionWdName    | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128)         | The TDWM workload name used for parsing activity.                             |
| OpEnvName        | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128)         | The TDWM operating environment name associated with the OpEnvId.              |
| SysConName       | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128)         | The TDWM System Health Condition name associated with the SysConId.           |

## Usage Notes

QryLogTDWMV is the Teradata Dynamic Workload Management view of the main log table DBQLLogTbl.

The nonunique primary index for the underlying table is a combination of the ProcID and the CollectTimeStamp fields. This combination causes each dispatcher to write its DBQL cache entries to a single AMP. This is important for database efficiency.

For information about the possible values for the ExceptionValue column, see "ExceptionValue Column."

## Example: Select from QryLogTDWMV

The following SELECT statement retrieves the QryLogTDWMV view:

```
SELECT * from QryLogTDWMV;
```

The query returns the following result:

```

ProcID  30718
CollectTimeStamp 2013-12-16 16:35:27
      QueryID    307188518253277390
      UserID 00000104
      UserName TEST1
DefaultDatabase TEST1
AcctString SALES
SessionID          1,007
LogicalHostID      1
RequestNum          5
InternalRequestNum 5
LastStateChange 2013-12-16 08:50:32.960000
DelayTime          ?
WDID               12
OpEnvID            1
SysConID           1
LSN                ?
NoClassification
WDOVERRIDE
ResponseTimeMet
ExceptionValue      ?
FinalWDID           12
TDWMEstMaxRows      1,170,869,760
TDWMEstLastRows     1
TDWMEstTotalTime    2331.764361
TDWMEstMemUsage      4.375
TDWMAAllAmpFlag T
TDWMConfLevelUsed N
TDWMRuleID          ?
StatementGroup Select
SessionWDID         12

```

## QryLogUtilityV

**Category:** Query

**Database:** DBC

| View Column | Data Type                | Format | Comment                                   |
|-------------|--------------------------|--------|---|
| ProcID      | DECIMAL(5,0)<br>NOT NULL | -(5)9  | Returns the process ID of the dispatcher. |

| View Column       | Data Type                                 | Format                   | Comment   |
|-------------------|---|--------------------------|---|
| CollectTimeStamp  | TIMESTAMP(6)<br>NOT NULL                  | YYYY-MM-DDBHH:MI:SS.S(6) | Returns the time that the log entry was generated.  |
| UtilityName       | VARCHAR(30)<br>UNICODE NOT CASESPECIFIC   | X(30)                    | Returns the name of the utility.  |
| FastExportNoSpool | CHAR(1) LATIN<br>NOT CASESPECIFIC         | X(1)                     | For utilities implementing FastExport protocol, 'Y' indicates No Spool mode 'N' indicates Spool mode.   |
| ExtendedMLoad     | CHAR(1) LATIN<br>NOT CASESPECIFIC         | X(1)                     | 'Y' indicates MLOADX being used 'N' indicates MLOADX not being used.  |
| DSASOperation     | CHAR(1) LATIN<br>NOT CASESPECIFIC         | X(1)                     | 'D' indicates DUMP operation 'R' indicates RESTORE operation.   |
| UtilityRequest    | VARCHAR(2048)<br>UNICODE NOT CASESPECIFIC | X(2048)                  | The main utility SQL request for the job. It contains the target database and/or table name(s). The possible statements are: BEGIN LOADING (FastLoad), BEGIN MLOAD (MLOAD /MLOADX), SELECT (FastExport), DUMP/ RESTORE (DSA). |
| JobName           | VARCHAR(128)<br>UNICODE NOT CASESPECIFIC  | X(128)                   | DSA job name. Not used for other utilities at this time.  |
| JobInstanceID     | INTEGER                                   | --,---,---,--9           | DSA job execution ID. Not used for other utilities at this time.  |
| LSN               | INTEGER                                   | --,---,---,--9           | Logon Sequence Number used by the utility job.  |
| UserName          | VARCHAR(128)<br>UNICODE NOT CASESPECIFIC  | X(128)                   | Name of the user who ran the utility job.   |
| SessionID         | INTEGER<br>NOT NULL                       | --,---,---,--9           | Session ID of the control SQL session.  |
| LogicalHostID     | SMALLINT<br>NOT NULL                      | -(5)9                    | Logical Host ID of the control SQL session.   |

| View Column      | Data Type                                | Format                   | Comment   |
|------------------|--|--------------------------|---|
| LogonDateTime    | TIMESTAMP(6)<br>NOT NULL                 | YYYY-MM-DDBHH:MI:SS.S(6) | Date (YYYY-MM-DD) and Time (HH:MM:SS) of the control SQL session logon. |
| WDID             | INTEGER                                  | --,---,---,--9           | Workload definition assigned to the main utility work.                  |
| FinalWDID        | INTEGER                                  | --,---,---,--9           | Workload definition under which the main utility work completed.        |
| SessionWDID      | INTEGER                                  | --,---,---,--9           | Workload definition used for query parsing.                             |
| TDWMRuleID       | INTEGER                                  | --,---,---,--9           | RuleID of the TDWM rule that caused a delay.                            |
| CPUDecayLevel    | SMALLINT                                 | ---,--9                  | Contains most severe decay level reached for CPU usage.                 |
| IODecayLevel     | SMALLINT                                 | ---,--9                  | Contains most severe decay level reached for I/O usage.                 |
| UserID           | BYTE(4) NOT NULL                         | X(8)                     | Identifier of the user who submitted the utility job.                   |
| ZoneID           | BYTE(4) NOT NULL                         | X(8)                     | Zone ID of the user who submitted the utility job.                      |
| AcctString       | VARCHAR(128)<br>UNICODE NOT CASESPECIFIC | X(128)                   | Unexpanded Account String of the user who submitted the utility job.    |
| ExpandAcctString | VARCHAR(128)<br>UNICODE NOT CASESPECIFIC | X(128)                   | If account expansion was invoked, account string in expanded format.    |
| AcctStringTime   | FLOAT                                    | 99:99:99                 | Time (HH:MM:SS) if account string contains \$T.                         |
| AcctStringHour   | SMALLINT                                 | -(5)9                    | Hour (HH) if account string contains \$H.                               |
| AcctStringDate   | DATE                                     | YY/MM/DD                 | Date (YY/MM/DD) if account string contains \$D.                         |
| LogonSource      | CHAR(128) LATIN<br>NOT CASESPECIFIC      | X(128)                   | Identification of the place from where the user accessed the system.    |

| View Column      | Data Type                                     | Format                      | Comment   |
|------------------|---|-----------------------------|---|
| AppID            | CHAR(30)<br>UNICODE NOT<br>CASESPECIFIC       | X(30)                       | Application ID of the utility job.                                |
| ClientID         | CHAR(30)<br>UNICODE NOT<br>CASESPECIFIC       | X(30)                       | Client ID of the utility job.                                     |
| ClientAddr       | CHAR(45) LATIN<br>NOT<br>CASESPECIFIC         | X(45)                       | Client address of the utility job.                                |
| QueryBand        | VARCHAR(12304)<br>UNICODE NOT<br>CASESPECIFIC | X(12304)                    | Query band used in the utility job.                               |
| ProfileID        | BYTE(4)                                       | X(8)                        | Profile ID used for the utility job.                              |
| ProxyUser        | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC   | X(128)                      | Proxy user used for the utility job.                              |
| ProxyRole        | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC   | X(128)                      | Proxy role used for the utility job.                              |
| OpEnvID          | INTEGER                                       | --,---,---,--9              | Operational Environment ID in which the utility job was executed. |
| SysConID         | INTEGER                                       | --,---,---,--9              | System Condition ID in which the utility job was executed.        |
| NoClassification | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC          | X(1)                        | Reserved for future use.  |
| WDOVERRIDE       | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC          | X(1)                        | Reserved for future use.  |
| ResponseTimeMet  | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC          | X(1)                        | Reserved for future use.  |
| DelayTime        | FLOAT   | ---,---,---,---,<br>--9.999 | Number of seconds the utility job was delayed by TDWM.            |
| JobStartTime     | TIMESTAMP(6)                                  | YYYY-MM-DD<br>HH:MI:SS.S(6) | Job start time.   |

| View Column             | Data Type    | Format                      | Comment  |
|-------------------------|--------------|-----------------------------|--|
| JobEndTime              | TIMESTAMP(6) | YYYY-MM-DDBHH:MI:SS.S(6)    | Job end time.  |
| RowsInserted            | FLOAT        | ---,---,---,---,<br>--9     | Number of rows inserted by the utility job.  |
| RowsUpdated             | FLOAT        | ---,---,---,---,<br>--9     | Number of rows updated by the utility job.   |
| RowsDeleted             | FLOAT        | ---,---,---,---,<br>--9     | Number of rows deleted by the utility job.   |
| RowsExported            | FLOAT        | ---,---,---,---,<br>--9     | Number of rows exported by the utility job.  |
| NumSesOrBuildProc       | SMALLINT     | -(5)9                       | Number of sessions used. For DSA, number of build processes used.                        |
| MaxDataWaitTime         | FLOAT        | ---,---,---,---,<br>--9     | The highest wait time in seconds from external source for input data or output requests. |
| MaxDataWaitTimeID       | INTEGER      | --,---,---,--9              | ID associated with the highest wait time.  |
| Phase0StartTime         | TIMESTAMP(6) | YYYY-MM-DDBHH:MI:SS.S(6)    | Returns phase 0 start date and time of the utility job.                                  |
| Phase0EndTime           | TIMESTAMP(6) | YYYY-MM-DDBHH:MI:SS.S(6)    | Returns phase 0 end date and time of the utility job.                                    |
| Phase0TotalCPUTime      | FLOAT        | ZZ,ZZZ,ZZ9.999              | Phase 0 total CPU Time in seconds.   |
| Phase0TotalCPUTimeNorm  | FLOAT        | ZZ,ZZZ,ZZ9.999              | Phase 0 total normalized CPU Time in seconds.  |
| Phase0ParserCPUTime     | FLOAT        | ZZ,ZZZ,ZZ9.999              | Phase 0 Parser and Dispatcher CPU time in seconds.                                       |
| Phase0ParserCPUTimeNorm | FLOAT        | ZZ,ZZZ,ZZ9.999              | Phase 0 normalized Parser and Dispatcher CPU time in seconds.                            |
| Phase0TotalIO           | FLOAT        | ---,---,---,---,<br>--9     | Phase 0 logical I/O count.   |
| Phase0IOKB              | FLOAT        | ---,---,---,---,<br>--9.999 | Phase 0 logical I/O in KB.   |

| View Column               | Data Type    | Format                       | Comment  |
|---------------------------|--------------|------------------------------|--|
| Phase0PhysIO              | FLOAT        | ----,---,---,---,<br>--9.999 | Returns the number of physical I/Os in phase 0 of the utility job.           |
| Phase0PhysIOKB            | FLOAT        | ----,---,---,---,<br>--9.999 | Returns total physical I/O usage in kilobytes in phase 0 of the utility job. |
| Phase1StartTime           | TIMESTAMP(6) | YYYY-MM-DDBHH:MI:SS.S(6)     | Returns phase 1 start date and time of the utility job.                      |
| Phase1EndTime             | TIMESTAMP(6) | YYYY-MM-DDBHH:MI:SS.S(6)     | Returns phase 1 end date and time of the utility job.                        |
| Phase1TotalCPUTime        | FLOAT        | ZZ,ZZZ,ZZ9.999               | Phase 1 total CPU Time in seconds.   |
| Phase1TotalCPUTimeNorm    | FLOAT        | ZZ,ZZZ,ZZ9.999               | Phase 1 total normalized CPU Time in seconds.                                |
| Phase1MaxCPUTime          | FLOAT        | ZZ,ZZZ,ZZ9.999               | Phase 1 highest AMP CPU time in seconds.                                     |
| Phase1MaxCPUAmpNumber     | INTEGER      | --,---,---,--9               | Phase 1 - ID of AMP with highest CPU time.                                   |
| Phase1MaxCPUTimeNorm      | FLOAT        | ZZ,ZZZ,ZZ9.999               | Phase 1 highest normalized AMP CPU time in seconds.                          |
| Phase1MaxCPUAmpNumberNorm | INTEGER      | --,---,---,--9               | Phase 1 - ID of AMP with highest normalized CPU time.                        |
| Phase1ParserCPUTime       | FLOAT        | ZZ,ZZZ,ZZ9.999               | Phase 1 Parser and Dispatcher CPU time in seconds.                           |
| Phase1ParserCPUTimeNorm   | FLOAT        | ZZ,ZZZ,ZZ9.999               | Phase 1 normalized Parser and Dispatcher CPU time in seconds.                |
| Phase1RSGCPUTime          | FLOAT        | ZZ,ZZZ,ZZ9.999               | Phase 1 RSG CPU time in seconds.   |
| Phase1RSGCPUTimeNorm      | FLOAT        | ZZ,ZZZ,ZZ9.999               | Phase 1 normalized RSG CPU time in seconds.                                  |
| Phase1TotalIO             | FLOAT        | ----,---,---,---,<br>--9     | Phase 1 logical I/O count.   |
| Phase1MaxIO               | FLOAT        | ----,---,---,---,<br>--9     | Phase 1 highest logical I/O count.   |

| View Column             | Data Type    | Format                      | Comment  |
|-------------------------|--------------|-----------------------------|--|
| Phase1MaxIOAmpNumber    | INTEGER      | --,---,---,--9              | Phase 1 - ID of AMP with highest logical I/O count.                          |
| Phase1IOKB              | FLOAT        | ---,---,---,---,<br>--9.999 | Phase 1 logical I/O in KB.   |
| Phase1PhysIO            | FLOAT        | ---,---,---,---,<br>--9.999 | Returns the number of physical I/Os in phase 1 of the utility job.           |
| Phase1PhysIOKB          | FLOAT        | ---,---,---,---,<br>--9.999 | Returns total physical I/O usage in kilobytes in phase 1 of the utility job. |
| Phase1MaxAWTUsage       | BYTEINT      | -(3)9                       | Phase 1 highest AWT usage.   |
| Phase1MaxAMPMemoryUsage | FLOAT        | ---,---,---,---,<br>--9     | Phase 1 highest AMP memory usage in MBs.                                     |
| Phase1MaxRSGMemoryUsage | FLOAT        | ---,---,---,---,<br>--9     | Phase 1 highest RSG memory usage in MBs.                                     |
| Phase1RowCount          | FLOAT        | ---,---,---,---,<br>--9     | Phase 1 row count.   |
| Phase1BlockCount        | FLOAT        | ---,---,---,---,<br>--9     | Phase 1 block count.   |
| Phase1ByteCount         | FLOAT        | ---,---,---,---,<br>--9     | Phase 1 byte count.  |
| Phase2StartTime         | TIMESTAMP(6) | YYYY-MM-DDBHH:MI:SS.S(6)    | Returns phase 2 start date and time of the utility job.                      |
| Phase2EndTime           | TIMESTAMP(6) | YYYY-MM-DDBHH:MI:SS.S(6)    | Returns phase 2 end date and time of the utility job.                        |
| Phase2TotalCPUTime      | FLOAT        | ZZ,ZZZ,ZZ9.999              | Phase 2 total CPU Time in seconds .  |
| Phase2TotalCPUTimeNorm  | FLOAT        | ZZ,ZZZ,ZZ9.999              | Phase 2 total normalized CPU Time in seconds.                                |
| Phase2MaxCPUTime        | FLOAT        | ZZ,ZZZ,ZZ9.999              | Phase 2 highest AMP CPU time in seconds.                                     |
| Phase2MaxCPUAmpNumber   | INTEGER      | ZZ,ZZZ,ZZ9.999              | Phase 2 - ID of AMP with highest CPU time.                                   |
| Phase2MaxCPUTimeNorm    | FLOAT        | ZZ,ZZZ,ZZ9.999              | Phase 2 highest normalized AMP CPU time in seconds.                          |

| View Column               | Data Type | Format                  | Comment  |
|---------------------------|-----------|-------------------------|--|
| Phase2MaxCPUAmpNumberNorm | INTEGER   | --,---,---,--9          | Phase 2 - ID of AMP with highest normalized CPU time.                        |
| Phase2ParserCPUTime       | FLOAT     | ZZ,ZZZ,ZZ9.999          | Phase 2 Parser and Dispatcher CPU time in seconds.                           |
| Phase2ParserCPUTimeNorm   | FLOAT     | ZZ,ZZZ,ZZ9.999          | Phase 2 normalized Parser and Dispatcher CPU time in seconds.                |
| Phase2RSGCPUTime          | FLOAT     | ZZ,ZZZ,ZZ9.999          | Phase 2 RSG CPU time in seconds.   |
| Phase2RSGCPUTimeNorm      | FLOAT     | ZZ,ZZZ,ZZ9.999          | Phase 2 normalized RSG CPU time in seconds.                                  |
| Phase2TotalIO             | FLOAT     | ---,---,---,---,--9     | Phase 2 logical I/O count.   |
| Phase2MaxIO               | FLOAT     | ---,---,---,---,--9     | Phase 2 highest logical I/O count.   |
| Phase2MaxIOAmpNumber      | INTEGER   | --,---,---,--9          | Phase 2 - ID of AMP with highest logical I/O count.                          |
| Phase2IOKB                | FLOAT     | ---,---,---,---,--9.999 | Phase 2 logical I/O in KB.   |
| Phase2PhysIO              | FLOAT     | ---,---,---,---,--9.999 | Returns the number of physical I/Os in phase 2 of the utility job.           |
| Phase2PhysIOKB            | FLOAT     | ---,---,---,---,--9.999 | Returns total physical I/O usage in kilobytes in phase 2 of the utility job. |
| Phase2MaxAWTUsage         | BYTEINT   | -(3)9                   | Phase 2 highest AWT usage.   |
| Phase2MaxAMPMemoryUsage   | FLOAT     | ---,---,---,---,--9     | Phase 2 highest AMP memory usage in MBs.                                     |
| Phase2MaxRSGMemoryUsage   | FLOAT     | ---,---,---,---,--9     | Phase 2 highest RSG memory usage in MBs.                                     |
| Phase2RowCount            | FLOAT     | ---,---,---,---,--9     | Phase 2 row count.   |
| Phase2BlockCount          | FLOAT     | ---,---,---,---,--9     | Phase 2 block count.   |
| Phase2ByteCount           | FLOAT     | ---,---,---,---,--9     | Phase 2 byte count.  |

| View Column             | Data Type    | Format                   | Comment  |
|-------------------------|--------------|--------------------------|--|
| Phase3StartTime         | TIMESTAMP(6) | YYYY-MM-DDBHH:MI:SS.S(6) | Returns phase 3 start date and time of the utility job.                      |
| Phase3EndTime           | TIMESTAMP(6) | YYYY-MM-DDBHH:MI:SS.S(6) | Returns phase 3 end date and time of the utility job.                        |
| Phase3TotalCPUTime      | FLOAT        | ZZ,ZZZ,ZZ9.999           | Phase 3 total CPU Time in seconds.   |
| Phase3TotalCPUTimeNorm  | FLOAT        | ZZ,ZZZ,ZZ9.999           | Phase 3 total normalized CPU Time in seconds.                                |
| Phase3ParserCPUTime     | FLOAT        | ZZ,ZZZ,ZZ9.999           | Phase 3 Parser and Dispatcher CPU time in seconds.                           |
| Phase3ParserCPUTimeNorm | FLOAT        | ZZ,ZZZ,ZZ9.999           | Phase 3 normalized Parser and Dispatcher CPU time in seconds.                |
| Phase3RSGCPUTime        | FLOAT        | ZZ,ZZZ,ZZ9.999           | Phase 3 RSG CPU time in seconds.   |
| Phase3RSGCPUTimeNorm    | FLOAT        | ZZ,ZZZ,ZZ9.999           | Phase 3 normalized RSG CPU time in seconds.                                  |
| Phase3TotalIO           | FLOAT        | ---,---,---,---,--9      | Phase 3 logical I/O count.   |
| Phase3IOKB              | FLOAT        | ---,---,---,---,--9.999  | Phase 3 logical I/O in KB.   |
| Phase3PhysIO            | FLOAT        | ---,---,---,---,--9.999  | Returns the number of physical I/Os in phase 3 of the utility job.           |
| Phase3PhysIOKB          | FLOAT        | ---,---,---,---,--9.999  | Returns total physical I/O usage in kilobytes in phase 3 of the utility job. |
| Phase3MaxAWTUsage       | BYTEINT      | -(3)9                    | Phase 3 highest AWT usage.   |
| Phase3MaxAMPMemoryUsage | FLOAT        | ---,---,---,---,--9      | Phase 3 highest AMP memory usage in MBs.                                     |
| Phase3MaxRSGMemoryUsage | FLOAT        | ---,---,---,---,--9      | Phase 3 highest RSG memory usage in MBs.                                     |
| Phase4StartTime         | TIMESTAMP(6) | YYYY-MM-DDBHH:MI:SS.S(6) | Returns phase 4 start date and time of the utility job.                      |

| View Column             | Data Type                                | Format                   | Comment  |
|-------------------------|--|--------------------------|--|
| Phase4EndTime           | TIMESTAMP(6)                             | YYYY-MM-DDBHH:MI:SS.S(6) | Returns phase 4 end date and time of the utility job.                                    |
| Phase4TotalCPUTime      | FLOAT                                    | ZZ,ZZZ,ZZ9.999           | Phase 4 total CPU Time in seconds.   |
| Phase4TotalCPUTimeNorm  | FLOAT                                    | ZZ,ZZZ,ZZ9.999           | Phase 4 total normalized CPU Time in seconds.  |
| Phase4ParserCPUTime     | FLOAT                                    | ZZ,ZZZ,ZZ9.999           | Phase 4 Parser and Dispatcher CPU time in seconds.                                       |
| Phase4ParserCPUTimeNorm | FLOAT                                    | ZZ,ZZZ,ZZ9.999           | Phase 4 normalized Parser and Dispatcher CPU time in seconds.                            |
| Phase4TotalIO           | FLOAT                                    | ---,---,---,---,--9      | Phase 4 logical I/O count.   |
| Phase4IOKB              | FLOAT                                    | ---,---,---,---,--9.999  | Phase 4 logical I/O in KB.   |
| Phase4PhysIO            | FLOAT                                    | ---,---,---,---,--9.999  | Returns the number of physical I/Os in phase 4 of the utility job.                       |
| Phase4PhysIOKB          | FLOAT                                    | ---,---,---,---,--9.999  | Returns total physical I/O usage in kilobytes in phase 4 of the utility job.             |
| Phase0UsedIota          | FLOAT                                    | ---,---,---,---,--9.999  | Returns used IOTAs in phase 0 of the utility job.  |
| Phase1UsedIota          | FLOAT                                    | ---,---,---,---,--9.999  | Returns used IOTAs in phase 1 of the utility job.  |
| Phase2UsedIota          | FLOAT                                    | ---,---,---,---,--9.999  | Returns used IOTAs in phase 2 of the utility job.  |
| Phase3UsedIota          | FLOAT                                    | ---,---,---,---,--9.999  | Returns used IOTAs in phase 3 of the utility job.  |
| Phase4UsedIota          | FLOAT                                    | ---,---,---,---,--9.999  | Returns used IOTAs in phase 4 of the utility job.  |
| ProfileName             | VARCHAR(128)<br>UNICODE NOT CASESPECIFIC | X(128)                   | Returns the name of the profile, if any, under which the user submitted the utility job. |
| WDName                  | VARCHAR(128)<br>UNICODE NOT CASESPECIFIC | X(128)                   | Returns the workload definition (WD) name assigned to the utility job.                   |

| View Column        | Data Type                                   | Format                       | Comment   |
|--------------------|---|------------------------------|---|
| FinalWDName        | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128)                       | Returns the workload definition name in which the utility job completed execution.                                      |
| SessionWDName      | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128)                       | Returns workload definition name associated with the session.   |
| OpEnvName          | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128)                       | Returns the name of the operating environment currently activated by the Teradata dynamic workload management software. |
| SysConName         | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128)                       | Returns the name of the system condition currently activated by the Teradata dynamic workload management software.      |
| DeferTime          | FLOAT                                       | ZZ,ZZZ,ZZ9.<br>999           | The number of seconds a query was deferred by workload management due to ARM rules.                                     |
| DeferRuleID        | INTEGER                                     | --,---,---,--9               | TDWM Rule ID of one of the ARM rules which caused this request to be deferred.  |
| TDWMAadmissionTime | TIMESTAMP(6)                                | YYYY-MM-DD<br>BHH:MI:SS.S(6) | The time when a request is admitted into the system by workload management (i.e., after ARM processing).                |

## Usage Notes

The DBQLUtilityTbl is populated if you use the WITH UTILITYINFO option for BEGIN/REPLACE QUERY LOGGING. Each row of the QryLogUtilityV view of the DBQLUtilityTbl stores information for one completed load/export utility or Data Stream Architecture job.

## Phases

The following table describes the different column phases shown in QryLogUtilityV:

| Utility Protocols           | Phase Zero Name | Phase One Name | Phase Two Name | Phase Three Name | Phase Four Name |
|-----------------------------|-----------------|----------------|----------------|------------------|-----------------|
| FASTLOAD<br>MLOAD<br>MLOADX | Setup           | Acquisition    | Application    | Cleanup          | N/A             |
| FastExport                  | Setup           | Select         | Export         | Cleanup          | N/A             |
| DSA                         | N/A             | Dictionary     | Data           | Build            | Postscript      |

Phase1 and Phase2 columns are used for the required phases. Phase0, Phase3, and Phase4 columns are used for the optional phases.

### Possible Values for CPUDecayLevel

CPUDecayLevel is the decay level reached for CPU usage per AMP, for requests using the default Timeshare workload management method. Possible values are:

| Value | Description  |
|-------|--|
| 0     | Indicates that the request is running at the default priority.   |
| 1     | Indicates that the priority level is reduced and the resource allotment is halved.   |
| 2     | Indicates that the priority level is reduced again, the resource allotment is halved again, and it will not be reduced more. |

### Possible Values for IODecayLevel

IODecayLevel is the decay level reached for I/O usage per AMP, for requests using the default Timeshare workload management method.

| Value | Description  |
|-------|--|
| 0     | Indicates that the request is running at the default priority.   |
| 1     | Indicates that the priority level is reduced and the resource allotment is halved.   |
| 2     | Indicates that the priority level is reduced again, the resource allotment is halved again, and it will not be reduced more. |

### Possible Values for MaxDataWaitTimeID

- For FastLoad, MultiLoad, and FastExport: The ID of the AMP owning the utility session with the highest data wait time.
- For MLOADX: The session number of the session with the highest data wait time.
- For DSA: The ID of the stream with the highest data wait time.

## Possible Values for UtilityRequest

UtilityRequest is the main utility SQL request for the job. It contains the target database and/or table names. The possible statements are:

- BEGIN LOADING (FastLoad)
- BEGIN MLOAD (MLOAD/MLOADX)
- SELECT (FastExport)
- DUMP/RESTORE (DSA)

## Example: Using QryLogUtilityV

The following SELECT statement retrieves the start time, end time, and row counts for FastLoad or TPT Load jobs submitted by USER1 since December 1, 2013:

```
SELECT UtilityName, LSN, JobStartTime, JobEndTime, RowsInserted
FROM DBC.QryLogUtilityV
WHERE UserName = 'USER1'
   AND (UtilityName = 'FASTLOAD' OR
        UtilityName = 'TPTLOAD')
   AND JobStartTime > DATE '2013-12-01'
ORDER BY JobStartTime;
```

Result:

```
UtilityName LSN JobStartTime
JobEndTime          RowsInserted
-----
FASTLOAD      47  2013-12-02 09:19:22.150000 2013-12-02 09:30:59.670000 100,000
FASTLOAD      48  2013-12-02 09:19:24.250000 2013-12-02 09:30:44.820000 100,000
TPTLOAD       54  2013-12-02 10:07:41.700000 2013-12-02 10:07:48.940000  81,920
```

## QryLogV

**Category:** Query

**Database:** DBC

| View Column | Data Type                | Format | Comment                                   |
|-------------|--------------------------|--------|---|
| ProcID      | DECIMAL(5,0)<br>NOT NULL | -(5)9  | Returns the process ID of the dispatcher. |

| View Column        | Data Type                                | Format                   | Comment  |
|--------------------|--|--------------------------|--|
| CollectTimeStamp   | TIMESTAMP(6)<br>NOT NULL                 | YYYY-MM-DDBHH:MI:SS.S(6) | (Prime Key) Time and date when the DBQLog cache was allocated.   |
| QueryID            | DECIMAL(18,0)<br>NOT NULL                | --Z(17)9                 | Returns a system-wide unique ID to identify the query.   |
| UserID             | BYTE(4)<br>NOT NULL                      | X(8)                     | Returns the ID of the user.  |
| UserName           | VARCHAR(128)<br>UNICODE NOT CASESPECIFIC | X(128)                   | Returns the name of the user who issued the query.   |
| DefaultDatabase    | VARCHAR(128)<br>UNICODE NOT CASESPECIFIC | X(128)                   | Returns the name of the current default database used in the query.  |
| AcctString         | VARCHAR(128)<br>UNICODE NOT CASESPECIFIC | X(128)                   | Unexpanded Account String under which the user issued the query.   |
| ExpandAcctString   | VARCHAR(128)<br>UNICODE NOT CASESPECIFIC | X(128)                   | Returns the expanded logon string of the user.   |
| SessionID          | INTEGER<br>NOT NULL                      | --,---,---,--9           | Returns the unique identification number of the session.   |
| LogicalHostID      | SMALLINT<br>NOT NULL                     | -(5)9                    | Returns a unique identifier of the logon source for the logged query. A value of zero indicates an internal session.                       |
| RequestNum         | INTEGER<br>NOT NULL                      | --,---,---,--9           | Client request number for all queries. For statements within stored procedure CALL statements, the request number is the same as the CALL. |
| InternalRequestNum | INTEGER<br>NOT NULL                      | --,---,---,--9           | Returns the internal request number used by the Teradata Database.   |
| TxnUniq            | BYTE(4)                                  | X(8)                     | Transaction Uniq portion used with Proclid.  |
| LockLevel          | VARCHAR(10)<br>LATIN NOT CASESPECIFIC    | X(10)                    | Highest level lock associated with this request.   |
| LogonDateTime      | TIMESTAMP(6)<br>NOT NULL                 | YYYY-MM-DDBHH:MI:SS.S(6) | Returns the date and time that the session was logged on.  |

| View Column        | Data Type                                     | Format                   | Comment   |
|--------------------|---|--------------------------|---|
| AcctStringTime     | FLOAT   | 99:99:99                 | Returns the result from the &T code when the user has specified Account String Expansion (ASE).         |
| AcctStringHour     | SMALLINT                                      | -(5)9                    | Returns the result from the &H code when the user has specified Account String Expansion (ASE).         |
| AcctStringDate     | DATE  | YY/MM/DD                 | Returns the result from the &D code when the user has specified Account String Expansion (ASE).         |
| LogonSource        | CHAR(128) LATIN<br>NOT<br>CASESPECIFIC        | X(128)                   | Returns the logon source string text.   |
| ApplID             | CHAR(30)<br>UNICODE NOT<br>CASESPECIFIC       | X(30)                    | Returns the application ID.   |
| ClientID           | CHAR(30)<br>UNICODE NOT<br>CASESPECIFIC       | X(30)                    | Returns the client ID.  |
| ClientAddr         | CHAR(45) LATIN<br>NOT<br>CASESPECIFIC         | X(45)                    | The client address of the submitted query.  |
| QueryBand          | VARCHAR(12304)<br>UNICODE NOT<br>CASESPECIFIC | X(12304)                 | Returns the band under which the query is submitted.  |
| ProfileID          | BYTE(4)                                       | X(8)                     | Returns the unique number assigned to the cost profile instance in the system.                          |
| StartTime          | TIMESTAMP(6)<br>NOT NULL                      | YYYY-MM-DDBHH:MI:SS.S(6) | Returns the time the query is submitted.  |
| TDWMAadmissionTime | TIMESTAMP(6)                                  | YYYY-MM-DDBHH:MI:SS.S(6) | The time when a request is admitted into the system by workload management(i.e., after ARM processing). |
| FirstStepTime      | TIMESTAMP(6)<br>NOT NULL                      | YYYY-MM-DDBHH:MI:SS.S(6) | Returns the time the first step for this query is dispatched.   |
| FirstRespTime      | TIMESTAMP(6)                                  | YYYY-MM-                 | Time first response was sent to the host.   |

| View Column        | Data Type            | Format                         | Comment  |
|--------------------|----------------------|--------------------------------|--|
|                    |                      | DDBHH:<br>MI:SS.S(6)           |  |
| ElapsedTime        | HS                   | -h(4):mm:<br>ss.s(6)           | Returns the difference between first response time and start time.                         |
| NumSteps           | SMALLINT<br>NOT NULL | ---,--9                        | Returns the total number of (level 1) steps for this query.                                |
| NumJoinSteps       | SMALLINT             | ---,--9                        | Number of Join Steps in the request.   |
| NumSumSteps        | SMALLINT             | ---,--9                        | Number of Sum steps in the request.  |
| NumStepswPar       | SMALLINT             | ---,--9                        | Returns the number of (level 1) steps with parallel steps.                                 |
| MaxStepsInPar      | SMALLINT             | ---,--9                        | Returns the maximum number of (level 2) steps done in parallel for this query.             |
| NumResultRows      | FLOAT                | ---,---,---,<br>---,--9        | Returns the total number of rows returned for the query.                                   |
| NumResultOneMBRows | FLOAT                | ---,---,---,<br>---,--9        | The number of 1MB rows (1MB > size > 64KB) in the set of rows returned for the query.      |
| MaxOneMBRowSize    | INTEGER              | --,---,---,--9                 | The actual size in bytes of the largest 1MB row in the set of rows returned for the query. |
| TotalIOCount       | FLOAT                | ---,---,---,<br>---,--9        | Returns the number of IOs from AMPs that were generated by the query.                      |
| AMPCPUTime         | FLOAT                | ZZ,ZZZ,<br>ZZ9.999             | The total AMP CPU time in seconds used by this query.                                      |
| ParserCPUTime      | FLOAT                | ZZ,ZZZ,<br>ZZ9.999             | Returns the total parser and dispatcher CPU time in seconds used for the query.            |
| UtilityByteCount   | BIGINT               | --,---,---,---,<br>---,---,--9 | The number of bytes transferred by client as part of FastLoad or MultiLoad job.            |
| UtilityRowCount    | FLOAT                | ---,---,---,<br>---,--9        | Returns the number of rows loaded by FastLoad or MultiLoad.                                |
| ErrorCode          | INTEGER              | --,---,---,--9                 | Returns error code if the query caused a Parser syntax error.                              |

| View Column     | Data Type                                     | Format                  | Comment   |
|-----------------|---|-------------------------|---|
| ErrorText       | VARCHAR(1024)<br>UNICODE NOT<br>CASESPECIFIC  | X(1024)                 | Returns the text from the error if<br>ErrorCode is not 0.   |
| WarningOnly     | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC          | X(1)                    | Returns warning code T for true<br>if the error was reported while<br>running TDWM in the warning<br>mode, or it is not logged at all.<br>A null is found in the field if<br>WarningOnly is not true.           |
| AbortFlag       | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC          | X(1)                    | Returns T(yes) if this query<br>was aborted.  |
| CacheFlag       | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC          | X(1)                    | This field is blank if the query<br>is not found in step cache. The<br>possible values are T, G, S,<br>and A.   |
| StatementType   | CHAR(20) LATIN<br>NOT<br>CASESPECIFIC         | X(20)                   | Statement type issued by the<br>query. Final statement type in a<br>multistatement request.   |
| StatementGroup  | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC   | X(128)                  | Returns the grouping for the<br>query, whether it is a DDL,<br>DML, or SELECT statement.<br>For a multistatement request,<br>this value indicates the number<br>of different statement types in<br>the request. |
| QueryText       | VARCHAR(10000)<br>UNICODE NOT<br>CASESPECIFIC | X(10000)                | Returns the query text. The<br>default size is 200 characters.  |
| NumOfActiveAMPs | INTEGER                                       | --,---,---,--9          | Returns the number of AMPs<br>that were active for this query.  |
| MaxAMPCPUTime   | FLOAT   | ZZ,ZZZ,<br>ZZ9.999      | Returns the CPU time in<br>seconds of the highest CPU<br>utilized AMP in the query.   |
| MaxCPUAmpNumber | INTEGER                                       | --,---,---,--9          | Returns the number of the AMP<br>with the highest CPU activity.   |
| MinAmpCPUTime   | FLOAT   | ZZ,ZZZ,<br>ZZ9.999      | Returns the CPU time in<br>seconds of the lowest CPU<br>utilized AMP in the step.   |
| MaxAmpIO        | FLOAT   | ---,---,---,<br>---,--9 | (renamed from HotAmpIO)<br>Returns the I/O count of the<br>highest I/O utilized AMP in<br>the step.   |

| View Column         | Data Type | Format                 | Comment   |
|---------------------|-----------|------------------------|---|
| MaxIOAmpNumber      | INTEGER   | --,---,---,--9         | Returns the number of the AMP with the highest IO usage for this step.  |
| MinAmpIO            | FLOAT     | ---,---,---,---,--9    | (renamed from LowAmpIO)<br>Returns the I/O count of the lowest I/O utilized AMP in the query.   |
| SpoolUsage          | BIGINT    | --,---,---,---,---,--9 | Peak Spool usage (bytes) of any step in the query (DataCollectAlg=3). Otherwise, number of bytes used for spool in the query.             |
| LSN                 | INTEGER   | --,---,---,--9         | Returns the Logon Sequence Number used for a load utility.  |
| EstResultRows       | FLOAT     | ---,---,---,---,--9    | Estimated number of rows returned by this query.  |
| EstProcTime         | FLOAT     | ZZ,ZZZ,ZZ9.999         | EstProcTime returns the estimated processing time in seconds (with .001 resolution) from the Optimizer.                                   |
| EstMaxStepTime      | FLOAT     | ZZ,ZZZ,ZZ9.999         | The estimated maximum step time in seconds from the optimizer.  |
| EstMaxRowCount      | FLOAT     | ---,---,---,---,--9    | Maximum of estimated row count for the steps in this query.   |
| TDWMEstMemUsage     | FLOAT     | ZZ,ZZZ,ZZ9.999         | Maximum estimated memory in MBs for largest step.   |
| AMPCPUTimeNorm      | FLOAT     | ZZ,ZZZ,ZZ9.999         | Returns the normalized AMP CPU time in seconds for co-existence systems.  |
| ParserCPUTimeNorm   | FLOAT     | ZZ,ZZZ,ZZ9.999         | Returns the normalized parser CPU time in seconds for co-existence systems.   |
| MaxAMPCPUTimeNorm   | FLOAT     | ZZ,ZZZ,ZZ9.999         | MaxAMPCPUTimeNorm returns the normalized CPU time in seconds (with .001 resolution) of the AMP with maximum CPU utilization in the query. |
| MaxCPUAmpNumberNorm | INTEGER   | --,---,---,--9         | Returns the number of the AMP with the highest CPU activity.  |
| MinAmpCPUTimeNorm   | FLOAT     | ZZ,ZZZ,ZZ9.999         | MinAmpCPUTimeNorm returns the normalized CPU Time in  |

| View Column              | Data Type  | Format         | Comment   |
|--------------------------|--|----------------|---|
|                          |  |                | seconds (with .001 resolution) of the AMP with minimum CPU utilization in the query.                                      |
| ParserExpReq             | FLOAT  | ZZ,ZZZ,ZZ9.999 | Returns the number of seconds the parser waited on an express request.  |
| ProxyUser                | VARCHAR(128)<br>UNICODE NOT CASESPECIFIC             | X(128)         | Returns the name of the proxy user.   |
| ProxyRole                | VARCHAR(128)<br>UNICODE NOT CASESPECIFIC             | X(128)         | The proxy role utilized for the query.  |
| SessionTemporalQualifier | VARCHAR(1024)<br>LATIN NOT CASESPECIFIC              | X(1024)        | DBC.QryLogV.<br>SessionTemporalQualifier records the session temporal qualifier for the session apart from the SQL Stored |
| CalendarName             | VARCHAR(128)<br>UNICODE NOT CASESPECIFIC<br>NOT NULL | X(128)         | DBC.QryLogV.CalendarName records the Calendar Name set for the session  |
| CPUDecayLevel            | SMALLINT   | ---,--9        | Contains most severe decay level reached for CPU usage on any one node.   |
| IODecayLevel             | SMALLINT   | ---,--9        | Contains most severe decay level reached for IO usage on any one node.  |
| TacticalCPUException     | INTEGER  | --,---,---,--9 | The total number of nodes that had CPU exception.   |
| TacticalIOException      | INTEGER  | --,---,---,--9 | The total number of nodes that had IO exception.  |
| SeqRespTime              | FLOAT  | ZZ,ZZZ,ZZ9.999 | Contains sum of the response time of all steps as if they had been executed sequentially in units of seconds.             |
| ReqIOKB                  | FLOAT  | ZZ,ZZZ,ZZ9.999 | Total logical IO usage in kilobytes for the request.  |
| ReqPhysIO                | FLOAT  | ZZ,ZZZ,ZZ9.999 | Number of physical IOs for the request.   |
| ReqPhysIOKB              | FLOAT  | ZZ,ZZZ,ZZ9.999 | Total physical IO usage in kilobytes for the request.   |

| View Column      | Data Type                             | Format                   | Comment  |
|------------------|---------------------------------------|--------------------------|--|
| DataCollectAlg   | BYTEINT                               | -(3)9                    | Returns CPU/IO collection algorithm used by DBQL. Possible values include: 1-Classic Algo  |
| CallNestingLevel | BYTEINT                               | -(3)9                    | The level of nesting when the request is running in stored procedure.  |
| NumRequestCtx    | BYTEINT                               | -(3)9                    | Number of request contexts associated with the session.  |
| KeepFlag         | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC  | X(1)                     | This is a flag with value (Y) or (N) indicating that the response parcel should be kept until the client responds that it no longer needs it.  |
| QueryRedriven    | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC  | X(1)                     | This is a flag with value (Y) or (N) to indicate if the query was re-driven.   |
| ReDriveKind      | CHAR(10) LATIN<br>NOT<br>CASESPECIFIC | X(10)                    | Contains the type of reset protection with which the query was re-driven. (MN)-Memory Non-fallback responses, (DF) -Dictionary table Fallback responses, (DN)-Dictionary Table Non-fallback responses. |
| LastRespTime     | TIMESTAMP(6)                          | YYYY-MM-DDBHH:MI:SS.S(6) | End of the request, end of the host/client response phase where applicable (DataCollectAlg=3).   |
| DisCPUTime       | FLOAT                                 | ---,---,---,---,--9.999  | The CPU time used in the dispatcher by this query (seconds).   |
| Statements       | INTEGER                               | --,---,---,--9           | Number of statements in the request.   |
| DisCPUTimeNorm   | FLOAT                                 | ---,---,---,---,--9.999  | The Normalized CPU time used in the dispatcher by this query (seconds).  |
| TxnMode          | CHAR(10) LATIN<br>NOT<br>CASESPECIFIC | X(10)                    | Transaction mode of this query (ANSI, BTET).   |
| RequestMode      | CHAR(5) LATIN<br>NOT<br>CASESPECIFIC  | X(5)                     | Request mode of this query: Prep - only prepare query; PrepS - only prepare query, parameterized SQL; Exec - execute the query; Both - Prepare and Execute query.                                      |

| View Column          | Data Type  | Format                  | Comment  |
|----------------------|--|-------------------------|--|
| DBQLStatus           | INTEGER  | --,---,---,--9          | Internal DBQL logging status. Zero indicates no status conditions (DataCollectAlg=3).  |
| NumFragments         | INTEGER  | --,---,---,--9          | The number of fragments in a IPE plan execution of request.  |
| VHLogicalIO          | FLOAT  | ---,---,---,---,--9.999 | Total number of Very Hot logical I/Os for the whole request.   |
| VHPhysIO             | FLOAT  | ---,---,---,---,--9.999 | Total number of Very Hot Physical I/Os for the whole request.  |
| VHLogicalIOKB        | FLOAT  | ---,---,---,---,--9.999 | Total Very Hot logical I/Os in kilobytes for the whole request.  |
| VHPhysIOKB           | FLOAT  | ---,---,---,---,--9.999 | Total Very Hot Physical I/Os in kilobytes for the whole request.   |
| LockDelay            | FLOAT  | ---,---,---,---,--9.999 | Maximum wait time to get a lock on object in centi-seconds.  |
| CheckpointNum        | FLOAT  | ---,---,---,---,--9     | Checkpoint interval number for MLOADX.   |
| UnityTime            | FLOAT  | ---,---,---,---,--9.999 | Time in seconds the stored procedure request was being processed by Unity.   |
| UtilityInfoAvailable | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC                   | X(1)                    | Indication if a request also has utility information available in the DBQLUtilityTbl table.                                    |
| UnitySQL             | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC                   | X(1)                    | Y - indicates Unity requested the SQL be modified. This will be set when Unity asks for Unity transaction table to be updated. |
| ThrottleBypassed     | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC                   | X(1)                    | Returns whether the request was only allowed to run due to the throttlebypass feature of TDWM.                                 |
| FlexThrottle         | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC                   | X(1)                    | Flag 'T' indicates this query was released from the TDWM delay queue due to the Flex Throttle feature.                         |
| IterationCount       | INTEGER  | --,---,---,--9          | Iteration count for data parcel associated with a request.   |
| TTGranularity        | VARCHAR(30)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(30)                   | Records the Transaction Time Granularity Name set for the session  |

| View Column          | Data Type                                   | Format                         | Comment  |
|----------------------|---|--------------------------------|--|
| MaxStepMemory        | FLOAT                                       | ---,---,---,<br>---,--9.999    | The maximum memory in MBs used by any one step in the request.                     |
| TotalServerByteCount | BIGINT                                      | --,---,---,---,<br>---,---,--9 | Total bytes sent and received from the foreign server.                             |
| PersistentSpool      | BIGINT                                      | --,---,---,---,<br>---,---,--9 | Persistent part of SpoolUsage.   |
| RemoteQuery          | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC        | X(1)                           | Flag 'T' indicates a remote query has been submitted.                              |
| ProxyUserID          | BYTE(4)                                     | X(8)                           | Returns the internal ID of the proxy user if the proxy user is a permanent user.   |
| DelayTime            | FLOAT                                       | ZZ,ZZZ,<br>ZZ9.999             | Returns the number of seconds the query was delayed by Workload Management.        |
| TDWMRuleID           | INTEGER                                     | --,---,---,--9                 | Returns rule identifier of the query.  |
| MinRespHoldTime      | FLOAT                                       | Z,ZZ9.9                        | Seconds response was held to meet the minimum response time.                       |
| TotalFirstRespTime   | FLOAT                                       | ZZ,ZZZ,<br>ZZ9.999             | DelayTime + Execution Time + MinRespHoldTime. (in seconds)                         |
| ParamQuery           | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC        | X(1)                           | Flag 'T' indicates the query is parameterized.                                     |
| ProfileName          | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128)                         | Returns the name of the profile, if any, under which the user submitted the query. |
| WDName               | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128)                         | Returns the workload definition (WD) name assigned to the query.                   |
| MaxNumMapAMPs        | INTEGER                                     | --,---,---,--9                 | Returns the number of AMPs in the largest contiguous map used by the request.      |
| MinNumMapAMPs        | INTEGER                                     | --,---,---,--9                 | The number of AMPs in the smallest contiguous map used by this request.            |
| SysDefNumMapAMPs     | INTEGER                                     | --,---,---,--9                 | Number of AMPs in the system-default map.  |

| View Column             | Data Type                             | Format                      | Comment  |
|-------------------------|---------------------------------------|-----------------------------|--|
| UsedIota                | FLOAT                                 | ZZ,ZZZ,<br>ZZ9.999          | Return IO Tokens used by the requests.   |
| ImpactSpool             | BIGINT                                | -(19)9                      | System level spool impact by the request   |
| AutoDBAData             | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC  | X(1)                        | Values indicate if the request contains AutoDBA Data.  |
| UAFName                 | CHAR(10) LATIN<br>NOT<br>CASESPECIFIC | X(10)                       | Name of an Analytic Function being executed by EXECUTE FUNCTION. The default is NULL.                            |
| UnityQueryType          | BYTEINT                               | -(3)9                       | Indicates if it is a replayed SQL or CDA.  |
| DefaultDBCACHEUsed      | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC  | X(1)                        | Value of 'T' indicating that default DB part of the step cache used, 'F' otherwise. NULL if request not cached.  |
| ReqAWTTime              | FLOAT                                 | ---,---,---,<br>---,--9.999 | Returns AWT elapsed time in seconds for the request.   |
| MaxReqAwtTime           | FLOAT                                 | ---,---,---,<br>---,--9.999 | Returns Max AWT elapsed time in seconds for the request.   |
| MaxReqAWTTimeAmpNum     | INTEGER                               | --,---,---,--9              | Returns AMP number that had max AWTTime for the request.   |
| MinReqAWTTime           | FLOAT                                 | ---,---,---,<br>---,--9.999 | Returns Min AWT elapsed time in seconds for the request.   |
| UDFVMData               | FLOAT                                 | ---,---,---,<br>---,--9.999 | UDF Virtual memory data size for the request.  |
| UDFVMPeak               | FLOAT                                 | ---,---,---,<br>---,--9.999 | UDF Virtual memory peak for the request.   |
| TotalUDFMemUsage        | FLOAT                                 | ---,---,---,<br>---,--9.999 | Total memory in bytes used by UDF for the request.   |
| MaxReqUDFMemUsage       | FLOAT                                 | ---,---,---,<br>---,--9.999 | Max memory in bytes used by UDF for the request.   |
| MaxReqUDFMemUsageAmpNum | INTEGER                               | --,---,---,--9              | AMP number that had max memory used by UDF for this request.   |
| PGRCTimeToGetPlan       | FLOAT                                 | ---,---,---,<br>---,--9.999 | Time taken for communication between Cache Management Threads and session Thread to get plan from target PE when |

| View Column          | Data Type                            | Format                         | Comment   |
|----------------------|--------------------------------------|--------------------------------|---|
|                      |                                      |                                | Partitioned Global Request Cache feature is ON.   |
| PGRCTgtPENum         | SMALLINT                             | ---,--9                        | Returns the target PE number which holds the PSTEPS for this request when PGRC feature is ON. |
| NosRecordsReturned   | BIGINT                               | --,---,---,---,<br>---,---,--9 | Number of records returned by a Native Object Store request.                                  |
| NosRecordsSkipped    | BIGINT                               | --,---,---,---,<br>---,---,--9 | Number of records skipped by a Native Object Store request.                                   |
| NosPhysReadIO        | BIGINT                               | --,---,---,---,<br>---,---,--9 | Total physical read IOs for Native Object Store files.  |
| NosPhysReadIOKB      | FLOAT                                | ---,---,---,<br>---,--9.999    | Total KB of physical read IOs for Native Object Store files.                                  |
| NosRecordsReturnedKB | FLOAT                                | ---,---,---,<br>---,--9.999    | Total KB of records returned for Native Object Store files.                                   |
| NosTotalIOWaitTime   | FLOAT                                | ---,---,---,<br>---,--9.999    | Total of the Native Object Store IO wait time in seconds for the request.                     |
| NosMaxIOWaitTime     | FLOAT                                | ---,---,---,<br>---,--9.999    | Max Native Object Store IO wait time for the request. Units in seconds.                       |
| NosCPUTime           | FLOAT                                | ---,---,---,<br>---,--9.999    | CPU time in seconds for reading Native Object Store files for the request.                    |
| NosTables            | INTEGER                              | --,---,---,--9                 | Total number of Native Object Store tables accessed in the request.                           |
| NosFiles             | INTEGER                              | --,---,---,--9                 | Numbers of Native Object Store file reads were attempted on by this request.                  |
| NosFilesSkipped      | INTEGER                              | --,---,---,--9                 | Number of Native Object Store files that were skipped.  |
| StepCacheHash        | INTEGER                              | --Z(17)9                       | Step cache hash value of this request if cached, NULL otherwise.                              |
| ResponseTimeMet      | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC | X(1)                           | 'T' if the query met the service level goal of the workload definition.                       |
| TDWMMSRCount         | INTEGER                              | --,---,---,--9                 | The number of statements excluding BT, ET, Commit,  |

| View Column           | Data Type                                   | Format                         | Comment   |
|-----------------------|---|--------------------------------|---|
|                       |   |                                | Macros, Materialized Tmp Tables and Null statements in the Request.                 |
| DeferTime             | FLOAT                                       | ---,---,---,<br>---,--9.999    | The number of seconds a query was deferred by workload management due to ARM rules. |
| DeferRuleID           | INTEGER                                     | --,---,---,--9                 | TDWM Rule ID of one of the ARM rules which caused this request to be deferred.      |
| StmtDMLRowCount       | JSON  | X(128)                         | Returns the row count for DML(Insert or Update or Delete) statements.               |
| UnityQueryForeignInfo | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128)                         | UnityForeignQueryID and UnityForeignSystemID  |
| NosPhysWriteIO        | BIGINT                                      | --,---,---,---,<br>---,---,--9 | Total physical write IOs for Native Object Store files for this request.            |
| NosPhysWriteIOKB      | FLOAT                                       | ---,---,---,<br>---,--9        | Total KB of physical write IOs for Native Object Store files for this request.      |
| NosFilesWritten       | INTEGER                                     | --,---,---,--9                 | Number of file writes attempted by Native Object Store in this request.             |

## Usage Notes

Database Query Logging (DBQL) tables track query behavior. Each row in QryLogV contains default DBQL information for a query.

### The Default Row

The fields of the default row provide general query information that is usually adequate for investigating a query that is interfering with performance. When no options are specified, a default row includes:

- User ID and user name under which the session being logged was initiated
- Unique ID for the process, session, and host (client) connection
- Account string, expanded as appropriate, that was current when the query completed
- First 200 characters of the query SQL statement
- CPU and I/O statistics
- Default database name that was current when the query completed

**FlexThrottle**

Flex Throttles automatically release a throttle to allow full utilization of the system and release work from the TDWM delay queue to utilize these resources. This column is set to 'T' if the request was released due to the Flex Throttle feature. Otherwise, the column is NULL.

**InternalRequestNum**

The internal request number.

For commands other than those within a stored procedure the internal request number and the number in the RequestNum field will be the same. For stored procedures invoked within a session, the internal request number increments by 1 for every request made by the stored procedure. This means that RequestNum will be the request number of the CALL and the value in InternalRequestNum will continue to increment for all other queries issued by the session.

**LogonSource**

Teradata recommends using alternative columns instead of the LogonSource column, if available. For information about the recommended columns for LogonSource, see "LogonSource Column Fields and Examples."

**MaxAMPSPool**

If you are using DBQL CPU/IO Collection algorithm 3, MaxAMPSPool is the largest peak pool usage of any AMP for this step. For other algorithms, MaxAMPSPool is the highest pool usage on any AMP at the end of the step.

**MinAMPSPool**

If you are using DBQL CPU/IO Collection algorithm 3, MinAMPSPool is the smallest peak pool usage of any AMP for this step. For other algorithms, MinAMPSPool is the lowest pool usage of any AMP at the end of the step.

**SessionTemporalQualifier**

When a DML or SELECT request refers to a temporal table but omits a temporal qualifier, the system uses the value of the session temporal qualifier. This is the session temporal qualifier in effect when the query is logged.

For example, this column uses the ANSIQUALIFIER value if the session temporal qualifier is set to ANSIQUALIFIER for ANSI temporal tables.

For more information about the session temporal qualifiers, see *Teradata Vantage™ Temporal Table Support* , B035-1182 and *Teradata Vantage™ ANSI Temporal Table Support* , B035-1186 .

**Possible Values for CacheFlag**

This column is blank if the query is not found in the request cache. Possible values include:

| Value | Description  |
|-------|--|
| T     | Query execution plan is generic and found in the request cache.  |
| S     | Query plan is specific (either a static or dynamic plan).  |
| G     | Query plan is generic and not in the request cache but is a candidate for caching in subsequent runs of the same query.  |
| A     | If a Specific Always decision is taken. That is, for each query USING values are peeked during request parsing. The optimizer may choose a dynamic plan with results feedback. |
| D     | Generic plan from the request cache is used for the request execution.   |
| P     | Request is parsed to generate the plan by peeking the literal values.  |
| N     | Request is parsed to generate the plan without peeking the literal values (generic plan).  |
| I     | PRCE infrastructure has decided specific plan is always better and this request is ineligible for Dynamic Parametrization of Literals feature.                                 |

For more information on specific plans and generic plans, see “Peeking at Parameterized Values in the Data Parcel” in *Teradata Vantage™ SQL Request and Transaction Processing*, B035-1142.

### Possible Values for CPUDecayLevel

| Value | Description  |
|-------|--|
| 0     | Indicates that the request is running at the default priority.   |
| 1     | Indicates the priority level is reduced and the resource allotment is halved.  |
| 2     | Indicates that the priority level is reduced again, the resource allotment is halved again, and it will not be reduced more. |

### Possible Values for DataCollectAlg

| Value | Description   |
|-------|---|
| 1     | Use the classic algorithm with step adjustments.                          |
| 2     | Use AMP algorithm 2 (diagnostic only).                                    |
| 3     | Use AMP algorithm 3 (includes information on aborted and parallel steps). |

### Possible Values for IODecayLevel

| Value | Description   |
|-------|---|
| 0     | Indicates that the request is running at the default priority.                |
| 1     | Indicates the priority level is reduced and the resource allotment is halved. |

| Value | Description  |
|-------|--|
| 2     | Indicates that the priority level is reduced again, the resource allotment is halved again, and it will not be reduced more. |

### Possible Values for LockLevel

Locklevel is the highest lock level for locks associated with this request. For explicit transactions, LockLevel is the lock level for the transaction. LockLevel does not include row hash locks. Possible LockLevel values:

- NOLOCK
- ACCESS
- READ
- WRITE
- EXCLUSIVE
- UNKNOWN

### Possible Values for QueryBand

| Value | Description                   |
|-------|-------------------------------|
| T     | Transaction query band pairs. |
| S     | Session query band pairs.     |
| P     | Profile query band pairs.     |

### Possible Values for QueryRedriven

| Value | Description  |
|-------|--|
| E     | The request was reparsed due to an internal Parser error.        |
| M     | The request was reparsed due to an internal Parser memory limit. |
| R     | The request was redriven due to a database restart.              |
| N     | The request was not redriven or reparsed.                        |

### Possible Values for RedriveKind

| Value | Description                                     |
|-------|---|
| ''    | Not participating in redrive.                   |
| MN    | Memory-based protection, no fallback spool.     |
| MF    | Memory-based protection, with fallback spool.   |
| DN    | Dictionary-based protection, no fallback spool. |

| Value | Description                                       |
|-------|---|
| DF    | Dictionary-based protection, with fallback spool. |

### Possible Values for RequestMode

| Value | Description                                       |
|-------|---|
| P     | Prep: Prepare mode (P).                           |
| E     | Exec: Execute mode (E).                           |
| B     | Both: Prepare and execute mode (B).               |
| S     | PrepS: Prepare, supporting parameterized SQL (S). |

### Possible Values for StatementGroup

If there is a DDL statement in a request, StatementGroup reports which type:

- DDL ALTER
- DDL CREATE
- DDL GRANT

If the statement has only one DML statement or multiple DML statements that are all of the same type, StatementGroup will indicate the type. For example if there are three DELETE statements in a request, StatementGroup will report:

DML DELETE

Similarly, for requests with individual or multiple INSERT, INSERT/ SELECT, UPDATE, MERGE INTO, or SELECT statements, StatementGroup will report:

- DML INSERT
- DML INSERT/SELECT
- DML UPDATE
- DML MERGEINTO
- SELECT

In a multistatement request with different types of DML statements, you will see a list showing the number of statements of each type in the request. For example, a request with one insert and two update statements will appear as:

DML Del=0 Ins=1 InsSel=0 Upd=2 Sel=0 MergeInto=0

### Possible Values for ThrottleBypassed

ThrottleBypassed indicates whether an active request is active solely due to the ThrottleBypass ruleset attribute. This attribute overrides the throttle limits if the session owning the request has an object lock higher than the Access level. Possible values include:

| Value | Description                                  |
|-------|--|
| 0     | ThrottleBypass ruleset attribute is not set. |
| 1     | ThrottleBypass ruleset attribute is set.     |

### Possible Values for TTGranularity

| Value       | Description   |
|-------------|---|
| LOGICALROW  | Row is timestamped with the time the row is processed by the AMP.   |
| REQUEST     | Row is timestamped with the time the request is submitted.  |
| TRANSACTION | Row is timestamped with the time when the first non-locking reference is made to a temporal table, or when the built-in function TEMPORAL_TIMESTAMP is first accessed during the transaction. |

### Possible Values for UtilityInfoAvailable

UtilityInfoAvailable indicates whether an SQL request has utility information available in DBQLUtilityTbl. Possible values include:

| Value | Description  |
|-------|--|
| Y     | SQL request has utility information (for example, from DSA or the control SQL session of a load or export utility) logged in the DBQLUtilityTbl table. |
| N     | SQL request does not have utility information logged in the DBQLUtilityTbl table.  |

### Example: Select a Specific QueryID from QryLogV

The following SELECT statement retrieves all rows that match the specified query ID from the QryLogV view.

```
select * from QryLogV where queryid = 307190925762023013;
```

### Example: Track Data Volume for Specific Load Jobs from QryLogV

This example shows how to track data volume for specific load jobs.

```
SELECT UtilityByteCount,UtilityRowCount from DBC.QryLogV where UtilityByteCount  
is not NULL or UtilityRowCount is not NULL;
```

Result:

| UtilityByteCount | UtilityRowCount |
|------------------|-----------------|
| -----            | -----           |
| 431              | 25              |
| 52               | 1               |

## Example: Using StmtDMLRowCount

This example shows how to use the StmtDMLRowCount column to see the number of rows inserted, updated, or deleted for DML statements.

Run a multistatement request, such as:

```
INSERT INTO T3(1, 'abc', 'def')
; INSERT INTO T3(1, 'ghi', 'jkl')
; MERGE INTO T3 AS t USING
(SELECT 1 AS a, 'stf' AS b, 'xyz' AS c) AS s on t.a = s.a
WHEN MATCHED THEN UPDATE SET c = s.c
WHEN NOT MATCHED THEN INSERT VALUES (s.a,s.b,s.c)
; MERGE INTO T3 AS t USING
(SELECT 1 AS a, 'stf' AS b, 'xyz' AS c) AS s on t.a = s.a
WHEN MATCHED THEN UPDATE SET c = s.c
WHEN NOT MATCHED THEN INSERT VALUES (s.a,s.b,s.c)
; DELETE FROM T3;
```

The multistatement request inserted 2 rows to table T3, updated 4 rows in table T3, and deleted 2 rows from table T3:

```
SELECT StmtDMLRowCount FROM DBC.QryLogV;
```

Result:

```
StmtDMLRowCount
-----
{"Insert":2,"Update":4,"Delete":2}
```

## QRYLogXMLDocV

**Category:** Query

**Database:** DBC

| View Column          | Data Type                              | Format                   | Comment   |
|----------------------|--|--------------------------|---|
| ProcID               | DECIMAL(5,0)<br>NOT NULL               | -(5)9                    | Returns the process ID of the dispatcher.   |
| CollectTimeStamp     | TIMESTAMP(6)<br>NOT NULL               | YYYY-MM-DDBHH:MI:SS.S(6) | (Prime Key) Date and time when cache was written.   |
| QueryID              | DECIMAL(18,0)                          | --Z(17)9                 | Returns a system-wide unique ID to identify the query.  |
| XMLTextDoc           | XM                                     | NULL                     | Returns the query plan represented as an XML document.  |
| XMLDocType           | INTEGER                                | --,---,---,--9           | Identifies the query logging options whose associated data is included in column XMLTextInfo. |
| XMLPlanEnabled       | VARCHAR(1)<br>UNICODE NOT CASESPECIFIC | X(1)                     | XMLPlanEnabled = Y if XMLPLAN query logging option is enabled else N.                         |
| StatsUsageEnabled    | VARCHAR(1)<br>UNICODE NOT CASESPECIFIC | X(1)                     | StatsUsageEnabled = Y if STATSUSAGE query logging option is enabled else N.                   |
| VerboseEnabled       | VARCHAR(1)<br>UNICODE NOT CASESPECIFIC | X(1)                     | VerboseEnabled = Y if VERBOSE query logging option is enabled else N.                         |
| DetailedStatsEnabled | VARCHAR(1)<br>UNICODE NOT CASESPECIFIC | X(1)                     | DetailedStatsEnabled = Y if DETAILED query logging option is enabled else N.                  |
| HasXMLPlanData       | VARCHAR(1)<br>UNICODE NOT CASESPECIFIC | X(1)                     | HasXMLPlanData = Y if XMLPLAN option data is contained in XMLTextInfo else N.                 |
| HasStatsUsageData    | VARCHAR(1)<br>UNICODE NOT CASESPECIFIC | X(1)                     | HasStatsUsageData = Y if STATSUSAGE option data is contained in XMLTextInfo else N.           |
| HasVerboseData       | VARCHAR(1)<br>UNICODE NOT CASESPECIFIC | X(1)                     | HasVerboseData = Y if VERBOSE option data is contained in XMLTextInfo else N.                 |
| HasDetailedStatsData | VARCHAR(1)<br>UNICODE NOT CASESPECIFIC | X(1)                     | HasDetailedStatsData = Y if DETAILED option data is contained in XMLTextInfo else N.          |

## Usage Notes

QRYLogXMLDocV provides a view into DBQLXMLTbl that combines multiple rows of XMLTextInfo with the same QueryID into a single XMLTextDoc document.

**Note:**

For XMLTextInfo Teradata recommends using the QryLogXMLDocV view instead of QryLogXMLV.

QRYLogXMLDocV view shows query plans and statistics usage information in an XML document. This view combines multiple rows of XMLTextInfo with the same QueryID into a single XMLTextDoc document. These BEGIN QUERY LOGGING or REPLACE QUERY LOGGING options populate the QryLogXMLDocV view:

- WITH XMLPLAN
- WITH STATSUSAGE

### **Possible Values for HasXMLPlanData, HasStatsUsageData, HasVerboseData, and HasDetailedStatsData**

For these columns, a value of Y or N can be specified. If you specify Y, the data is contained in the XMLTextInfo column on behalf of the specific logging [*Option*], where *Option* represents the STATSUSAGE or XMLPlan.

### **Possible Values for XMLDocType**

**Note:**

Teradata recommends that you use the Has[*Option*]Data and [*Option*]Enabled columns rather than the XMLDocType column directly, where *Option* represents the STATSUSAGE or XMLPlan. Queries against these columns allow you to specify a Y or N value rather than bit mapped values. For more information about these columns, see "Possible Values for HasXMLPlanData, HasStatsUsageData, HasVerboseData, and HasDetailedStatsData" (above) or see "Possible Values for XMLPlanEnabled, StatsUsageEnabled, VerboseEnabled, and DetailedStatsEnabled" (below).

### **Possible Values for XMLPlanEnabled, StatsUsageEnabled, VerboseEnabled, and DetailedStatsEnabled**

For these columns, a value of Y or N can be specified. If you specify Y, the [*Option*], where *Option* represents the STATSUSAGE or XMLPlan, was enabled when the data in XMLTextInfo was logged.

## **Example: Export XML Documents from QryLogXMLDocV**

```
SELECT XMLTextDoc FROM QRYLogXMLDocV;
```

The result is exported to one or more XML documents.

## **Example: Verifying XMLTextDoc is a Valid XML Document**

This example shows how to verify that XMLTextDoc is a valid XML document.

```
SELECT QueryID,
case XMLTextDoc.isdocument()
  WHEN 1 THEN 'Yes'
  ELSE 'No'
  END as "Well-formed"
FROM DBC.QRYLogXMLDocV
ORDER BY QueryID;
```

## Example: Serialize XMLTextDoc Document Into a String

This example serializes the XMLTextDoc document into a string and returns the number of bytes contained in the string.

```
SELECT QueryID,
CHARACTER_LENGTH(xmlserialize(document XMLTextDoc)) as PlanSize
FROM DBC.QRYLogXMLDocV
ORDER BY QueryID;
```

## Improving XML Text Readability

To combine multiple XML text rows with the same query ID into one easily readable document, perform a SELECT request on the QryLogXMLDocV view.

### Example

```
sel queryid,xmltextdoc from qrylogxmldocv where queryid=588188772714475132;
```

### Note:

Export the result set to a file. On the monitor, some of the lines may truncate.

The memory allocated for the input is 32 MB. To see the memory setting, use the following:

```
cufconfig -o | grep MallocLimit
```

To add more memory, adjust the MallocLimit setting using cufconfig. For more information, see *Teradata Vantage™ - Database Utilities*, B035-1102.

## Related Topics

For more information about the DBQL logging options, see *Teradata Vantage™ - Database Administration*, B035-1093.

## QryLogXMLV

**Category:** Query

**Database:** DBC

| View Column          | Data Type  | Format                   | Comment  |
|----------------------|--|--------------------------|--|
| ProcID               | DECIMAL(5,0)<br>NOT NULL                               | -(5)9                    | Returns the process ID of the dispatcher.  |
| CollectTimeStamp     | TIMESTAMP(6)<br>NOT NULL                               | YYYY-MM-DDBHH:MI:SS.S(6) | (Prime Key) Date and time when cache was written.  |
| QueryID              | DECIMAL(18,0)<br>NOT NULL                              | --Z(17)9                 | Returns a system-wide unique ID to identify the query.   |
| XMLRowNo             | INTEGER NOT NULL                                       | --,---,---,--9           | Indicates the row number for multiple rows used to capture the XML query plan. If the XML text is greater than 31000 characters, the system generates multiple rows. |
| XMLTextInfo          | VARCHAR(31000)<br>UNICODE NOT CASESPECIFIC<br>NOT NULL | X(31000)                 | Returns the query plan represented as an XML document.   |
| XMLDocType           | INTEGER  | --,---,---,--9           | Identifies the query logging options whose associated data is included in column XMLTextInfo.  |
| XMLPlanEnabled       | VARCHAR(1)<br>UNICODE<br>NOT CASESPECIFIC              | X(1)                     | XMLPlanEnabled = Y if XMLPLAN query logging option is enabled else N.  |
| StatsUsageEnabled    | VARCHAR(1)<br>UNICODE<br>NOT CASESPECIFIC              | X(1)                     | StatsUsageEnabled = Y if STATSUSAGE query logging option is enabled else N.  |
| VerboseEnabled       | VARCHAR(1)<br>UNICODE<br>NOT CASESPECIFIC              | X(1)                     | VerboseEnabled = Y if VERBOSE query logging option is enabled else N.  |
| DetailedStatsEnabled | VARCHAR(1)<br>UNICODE<br>NOT CASESPECIFIC              | X(1)                     | DetailedStatsEnabled = Y if DETAILED query logging option is enabled else N.   |
| HasXMLPlanData       | VARCHAR(1)<br>UNICODE<br>NOT CASESPECIFIC              | X(1)                     | HasXMLPlanData = Y if XMLPLAN option data is contained in XMLTextInfo else N.  |
| HasStatsUsageData    | VARCHAR(1)<br>UNICODE<br>NOT CASESPECIFIC              | X(1)                     | HasStatsUsageData = Y if STATSUSAGE option data is contained in XMLTextInfo else N.  |

| View Column          | Data Type                                 | Format | Comment  |
|----------------------|---|--------|--|
| HasVerboseData       | VARCHAR(1)<br>UNICODE<br>NOT CASESPECIFIC | X(1)   | HasVerboseData = Y if VERBOSE option data is contained in XMLTextInfo else N.        |
| HasDetailedStatsData | VARCHAR(1)<br>UNICODE<br>NOT CASESPECIFIC | X(1)   | HasDetailedStatsData = Y if DETAILED option data is contained in XMLTextInfo else N. |

## Usage Notes

This view of DBQLXMLTbl shows query plans and statistics usage information in an XML document. These BEGIN QUERY LOGGING or REPLACE QUERY LOGGING options populate the QryLogXMLV view:

- WITH XMLPLAN
- WITH STATSUSAGE

### Possible Values for HasXMLPlanData, HasStatsUsageData, HasVerboseData, and HasDetailedStatsData

For these columns, a value of Y or N can be specified. If you specify Y, the data is contained in the XMLTextInfo column on behalf of the specific logging [*Option*], where *Option* represents the STATSUSAGE or XMLPlan. If N is specified, the specific data (detailed statistics, statistics usage, VERBOSE EXPLAIN, and so on) is not stored in the XMLTextInfo column.

### Possible Values for XMLDocType

#### Note:

Teradata recommends that you use the Has[*Option*]Data and [*Option*]Enabled columns rather than the XMLDocType column directly, where *Option* represents the STATSUSAGE or XMLPlan. Queries against these columns allow you to specify a Y or N value rather than bit mapped values. For more information about these columns, see "Possible Values for HasXMLPlanData, HasStatsUsageData, HasVerboseData, and HasDetailedStatsData" (above) or see "Possible Values for XMLPlanEnabled, StatsUsageEnabled, VerboseEnabled, and DetailedStatsEnabled" (below).

### Possible Values for XMLPlanEnabled, StatsUsageEnabled, VerboseEnabled, and DetailedStatsEnabled

For these columns, a value of Y or N can be specified. If you specify Y, the [*Option*], where *Option* represents the STATSUSAGE or XMLPlan, was enabled when the data in XMLTextInfo was logged.

### Possible Values for XMLDocType

XMLDocType describes the type of content stored in column XMLTextInfo. This column contains an encoded 32-bit integer value whose bit positions describe the type of data in the associated XML document stored in the XMLTextInfo column.

**Note:**

It is easier to use the [Option] Enabled and Has [Option] columns to see this information. Queries against these columns allow you to specify a Y or N value instead of bit-mapped values.

The XML DocType bit mask values are:

| Value        | Description                             |
|--------------|---|
| 0x0000000001 | XMLPLAN logging option was enabled.     |
| 0x0000000002 | STATSUSAGE logging option was enabled.  |
| 0x0000010000 | Data for XMLPLAN was logged.            |
| 0x0000020000 | Data for STATSUSAGE was logged.         |
| 0x0000000004 | VERBOSE logging suboption was enabled.  |
| 0x0000000008 | DETAILED logging suboption was enabled. |
| 0x0000040000 | Data for VERBOSE was logged.            |
| 0x0000080000 | Data for DETAILED was logged.           |

## Examples Using QryLogXMLV

### Example: Extract Query Plan Information from QryLogXMLV

This example shows how to extract query plan information from the DBC.QryLogXMLV view.

```
SELECT
cast(t.result_value as char(30)) as "ColumnName"
  FROM (SELECT QueryID, XMLTextInfo
        FROM DBC.QryLogXMLV
        WHERE QueryID = 163833139835340224) as x(QueryID, XMLTextInfo),
table(
sysxml.xmlextractvalues(cast(x.QueryID as varchar(18)), x.XMLTextinfo,
null, '//Field[@JoinAccessFrequency>0]/@Name')) as t
  ORDER BY 1;
```

Result:

```
ColumnName
-----
a1
b1
```

### Example: Select Documents with Statistics Recommendations from QryLogXMLV

The following SELECT statement retrieves all documents that have statistics recommendations:

```
SELECT xmltextinfo FROM dbc.QryLogXMLV WHERE HasStatsUsageData = 'Y';
```

### Example: Select StatsUsage Information from QryLogXMLV

The following SELECT statement retrieves the number of queries where the STATUSAGE option was enabled but had no statistics recommendations:

```
SELECT Count(*) FROM dbc.QryLogXMLV WHERE StatsUsageEnabled = 'Y' AND
HasStatsUsageData = 'N';
```

### Example: Select Documents with DetailedStatsEnabled from QryLogXMLV

The following SELECT statement retrieves all documents with the DETAILED sub-option enabled:

```
SELECT xmltextinfo FROM dbc.QryLogXMLV WHERE DetailedStatsEnabled = 'Y';
```

### Example: Select Documents with DetailedStatsEnabled with No Detailed Statistics Information from QryLogXMLV

The following SELECT statement retrieves the number of queries where the DETAILED sub-option was enabled but had no detailed statistics information:

```
SELECT Count(*) FROM dbc.QryLogXMLV WHERE DetailedStatsEnabled = 'Y' AND
HasDetailedStatsData = 'N';
```

## Related Topics

For more information about the DBQL logging options, see *Teradata Vantage™ - Database Administration*, B035-1093.

## RCC\_ConfigurationV[X]

**Category:** Operations

**Database:** DBC

| View Column | Data Type        | Format         | Comment  |
|-------------|------------------|----------------|--|
| EventNum    | INTEGER NOT NULL | --,---,---,--9 | Returns the client system event number of the restore operation. |

| View Column    | Data Type                                     | Format  | Comment  |
|----------------|---|---------|--|
| LogProcessor   | SMALLINT                                      | -(5)9   | Returns the logical processor ID for an AMP not affected by the event.   |
| PhyProcessor   | SMALLINT                                      | -(5)9   | Returns the physical processor ID for an AMP not affected by the event.  |
| Vproc          | SMALLINT                                      | -(5)9   | Identifies the virtual processor for which an event was logged.  |
| ProcessorState | CHAR(1) LATIN<br>NOT CASESPECIFIC<br>NOT NULL | X(1)    | Returns D (the event was for all AMPs and the processor was down) or U (the event was for specific AMPs).  |
| RestartSeqNum  | SMALLINT                                      | ---,--9 | Returns an integer (0 through n) to indicate the number of times that the Teradata Database had to be restarted during the event. 0 indicates that no restarts took place. |

## Example: Using RCC\_ConfigurationV

The following SELECT statement selects event and processor information from the RCC\_Configuration view:

```
SELECT EventNum, LogProcessor, PhyProcessor
FROM RCC_ConfigurationV;
```

Result:

| EventNum | LogProcessor | PhyProcessor |
|----------|--------------|--------------|
| -----    | -----        | -----        |
| 21       | 1            | 1-0          |
| 75       | 1            | 1-0          |
| 88       | 1            | 1-0          |
| .        | .            | .            |
| .        | .            | .            |
| 21       | 2            | 1-2          |
| 75       | 2            | 1-2          |
| .        | .            | .            |
| .        | .            | .            |

## RCC\_MediaV[X]

**Category:** Operations

**Database:** DBC

| View Column    | Data Type                                     | Format         | Comment  |
|----------------|---|----------------|--|
| EventNum       | INTEGER NOT NULL                              | --,---,---,--9 | Returns the client system event number of the restore operation.   |
| VolSerialId    | CHAR(6) LATIN<br>NOT CASESPECIFIC<br>NOT NULL | X(6)           | Returns the unique six character volume serial assigned to a device.                                       |
| VolSequenceNum | SMALLINT                                      | ---,--9        | Returns the sequence number of the volume, which defines the position of the volume in a multi-volume set. |
| DupeDumpSet    | CHAR(1) LATIN<br>NOT CASESPECIFIC<br>NOT NULL | X(1)           | Returns a code to indicate whether the dump event created a duplicate archive dataset.                     |

## Example: Using RCC\_MediaV

The following SELECT statement selects all rows and all columns from the RCC\_Media view:

```
SELECT * FROM DBC.RCC_MediaV;
```

Result:

| EventNum | VolSerialID | VolSequenceNum | DupeDumpSet |
|----------|-------------|----------------|-------------|
| -----    | -----       | -----          | -----       |
| 21       | KAZ002      | 1              | N           |
| 76       | RDB003      | 1              | N           |
| 66       | RDB007      | 1              | N           |
| 19       | KAZ002      | 1              | N           |
| 66       | RDB008      | 2              | N           |
| 37       | MET001      | 1              | N           |
| 77       | RDB003      | 1              | N           |
| .        | .           | .              | .           |
| .        | .           | .              | .           |

## ReconfigDeleteOrderV

**Category:** Operations

**Database:** DBC

| View Column | Data Type        | Format | Comment   |
|-------------|------------------|--------|---|
| OrderNumber | INTEGER NOT NULL | -(10)9 | A user-defined value that determines when the table is processed relative to other tables |

| View Column      | Data Type   | Format | Comment   |
|------------------|---|--------|---|
|                  |   |        | during the Deletion and Redistribution phases of reconfigurations.  |
| DatabaseName     | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | Return the user database name.  |
| TableName        | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | Return the name of a table in the database specified by DatabaseName.   |
| CheckTableOption | BYTEINT   | -(3)9  | Indicates whether the CheckTable utility is automatically run after the deletion/rebuild NUSI for a user table during a reconfiguration.                      |
| ProcessOffline   | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL                  | X(1)   | ProcessOffline indicates whether the table should be processed (deleted or redistributed) offline only, when logons are disabled and the system is quiescent. |

## Usage Notes

This view provides the users a way to access the ReconfigDeleteOrderTbl. Users with the appropriate privileges can SELECT, INSERT, UPDATE, or DELETE on the base table using the view. Note that only SELECT, INSERT, UPDATE, and/or DELETE should be granted to users by user DBC.

### Possible Values for ProcessOffline

#### Note:

For the ProcessOffline column, tables undergoing large-scale changes should not be processed during online redistribution. Flag these tables for offline redistribution by setting this field to Y.

| Value | Description  |
|-------|--|
| Y     | Table should be processed during the offline portion of the deletion or redistribution phase of a reconfiguration.                       |
| N     | Table should be processed during the online portion of the deletion or redistribution phase of a reconfiguration. (This is the default.) |

### OrderNumber

The OrderNumber column allows gaps in the set of user-defined values (for example, you can have a set of 10 values that include 1, 2, 3, 5, 7, 8, 9, 12, 13, and 18).

# ReconfigInfoV

**Category:** Operations

**Database:** DBC

| View Column          | Data Type   | Format                      | Comment  |
|----------------------|---|-----------------------------|--|
| ReconfigId           | INTEGER<br>NOT NULL                               | -(10)9                      | Returns the unique identification number for a reconfiguration. Every reconfiguration operation is automatically assigned this number. |
| Description          | VARCHAR(512)<br>UNICODE NOT<br>CASESPECIFIC       | X(512)                      | Returns the description of the reconfiguration.  |
| ReconfigType         | BYTEINT<br>NOT NULL                               | -(3)9                       | Returns the type of reconfiguration.   |
| BeginTimeStamp       | TIMESTAMP(0)<br>NOT NULL                          | YYYY-MM-<br>DDBHH:MI:<br>SS | Returns the timestamp at the start of the reconfiguration.   |
| EndTimeStamp         | TIMESTAMP(0)                                      | YYYY-MM-<br>DDBHH:MI:<br>SS | Returns the timestamp at the end of the reconfiguration.   |
| BeginRedistTimeStamp | TIMESTAMP(0)                                      | YYYY-MM-<br>DDBHH:MI:<br>SS | Returns the timestamp at which the reconfig Redistribution phase begins.   |
| EndRedistTimeStamp   | TIMESTAMP(0)                                      | YYYY-MM-<br>DDBHH:MI:<br>SS | Returns the timestamp at which the reconfig Redistribution phase ends.   |
| BeginDelTimeStamp    | TIMESTAMP(0)                                      | YYYY-MM-<br>DDBHH:MI:<br>SS | Returns the timestamp at which the reconfig Deletion phase begins.   |
| EndDelTimeStamp      | TIMESTAMP(0)                                      | YYYY-MM-<br>DDBHH:MI:<br>SS | Returns the timestamp at which the reconfig Deletion phase ends.   |
| Status               | CHAR(10) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL | X(10)                       | Returns the current status of the Reconfig.  |
| CurrByteCount        | FLOAT NOT NULL                                    | ---,---,---,---,<br>--9     | Returns the number of bytes reconfig processed so far.   |
| CurrTabRedistCount   | FLOAT NOT NULL                                    | ---,---,---,---,<br>--9     | Returns the number of tables reconfig redistributed so far.  |

| View Column         | Data Type         | Format                   | Comment  |
|---------------------|-------------------|--------------------------|--|
| CurrTabDeleteCount  | FLOAT NOT NULL    | ----,---,---,---,<br>--9 | Returns the number of tables reconfig deleted so far.  |
| EstRemainRedistSecs | FLOAT NOT NULL    | ----,---,---,---,<br>--9 | Returns the estimated remaining time (in seconds) for the Redistribution phase.                                |
| EstRemainDeleteSecs | FLOAT NOT NULL    | ----,---,---,---,<br>--9 | Returns the estimated remaining time (in seconds) for the Deletion/Rebuild NUSI phase.                         |
| AddAmpCount         | SMALLINT          | -(5)9                    | Returns the number of AMPs added.  |
| DelAmpCount         | SMALLINT          | -(5)9                    | Returns the number of AMPs reconfig deleted.   |
| MovAmpCount         | SMALLINT          | -(5)9                    | Returns the number of AMPs moved.  |
| ModAmpCount         | SMALLINT          | -(5)9                    | Returns the number of modified AMPs.   |
| NodeCount           | SMALLINT NOT NULL | -(5)9                    | Returns the number of nodes in the new Config map.   |
| TotTaskCount        | SMALLINT NOT NULL | -(5)9                    | Returns the total number of tasks involved in the Reconfig.  |
| TotTableCount       | INTEGER NOT NULL  | -(10)9                   | Returns the total number of tables processed.  |
| TotByteCount        | FLOAT NOT NULL    | ----,---,---,---,<br>--9 | Returns the total number of bytes processed during the time period defined by BeginTimeStamp and EndTimeStamp. |
| TotCatchUpByteCount | FLOAT NOT NULL    | ----,---,---,---,<br>--9 | Returns the total number of bytes processed for catch-up.  |
| TotJournalByteCount | FLOAT NOT NULL    | ----,---,---,---,<br>--9 | Returns the ReconfigJournalTbl size.   |
| ActualRedistSecs    | FLOAT NOT NULL    | ----,---,---,---,<br>--9 | Returns the actual elapsed time in seconds for the Redistribution phase.                                       |
| ActualDeleteSecs    | FLOAT NOT NULL    | ----,---,---,---,<br>--9 | Returns the actual elapsed time in seconds for the Deletion/NUSI Rebuild phase.                                |
| EstRedistSecs       | FLOAT NOT NULL    | ----,---,---,---,<br>--9 | Returns the estimated Redistribution phase time in seconds.  |

| View Column                | Data Type      | Format                      | Comment  |
|----------------------------|----------------|-----------------------------|--|
| EstDeleteSecs              | FLOAT NOT NULL | ----,---,---,---,<br>--9    | Returns the estimated Deletion /NUSI Rebuild elapsed time.                                   |
| BeginCalcHBTimeStamp       | TIMESTAMP(0)   | YYYY-MM-<br>DDBHH:MI:<br>SS | Returns the timestamp at which reconfig Phase 1 (Calculate Hash Buckets) begins.             |
| EndCalcHBTimeStamp         | TIMESTAMP(0)   | YYYY-MM-<br>DDBHH:MI:<br>SS | Returns the timestamp at which reconfig Phase 1 (Calculate Hash Buckets) ends.               |
| BeginWrSpaceTimeStamp      | TIMESTAMP(0)   | YYYY-MM-<br>DDBHH:MI:<br>SS | Returns the timestamp at which reconfig Phase 4.1 (Store Database Space Info) begins.        |
| EndWrSpaceTimeStamp        | TIMESTAMP(0)   | YYYY-MM-<br>DDBHH:MI:<br>SS | Returns the timestamp at which reconfig Phase 4.1 (Store Database Space Info) ends.          |
| BeginWrCfgTimeStamp        | TIMESTAMP(0)   | YYYY-MM-<br>DDBHH:MI:<br>SS | Returns the timestamp at which reconfig Phase 4.7 (Update Current Configuration Map) begins. |
| EndWrCfgTimeStamp          | TIMESTAMP(0)   | YYYY-MM-<br>DDBHH:MI:<br>SS | Returns the timestamp at which reconfig Phase 4.7 (Update Current Configuration Map) ends.   |
| BeginWrCfgNewTimeStamp     | TIMESTAMP(0)   | YYYY-MM-<br>DDBHH:MI:<br>SS | Returns the timestamp at which reconfig Phase 4.8 (Update New Configuration Map) begins.     |
| EndWrCfgNewTimeStamp       | TIMESTAMP(0)   | YYYY-MM-<br>DDBHH:MI:<br>SS | Returns the timestamp at which reconfig Phase 4.8 (Update New Configuration Map) ends.       |
| BeginWrBkupIdTimeStamp     | TIMESTAMP(0)   | YYYY-MM-<br>DDBHH:MI:<br>SS | Returns the timestamp at which reconfig Phase 4.9 (Update Backup-IDs-Array) begins.          |
| EndWrBkupIdTimeStamp       | TIMESTAMP(0)   | YYYY-MM-<br>DDBHH:MI:<br>SS | Returns the timestamp at which reconfig Phase 4.9 (Update Backup-IDs-Array) ends.            |
| BeginInsUpdMapRowTimeStamp | TIMESTAMP(0)   | YYYY-MM-<br>DDBHH:MI:<br>SS | Returns the timestamp at which reconfig Phase 4 Insert/Update DBC.Maps ends.                 |
| EndInsUpdMapRowTimeStamp   | TIMESTAMP(0)   | YYYY-MM-<br>DDBHH:MI:<br>SS | Returns the timestamp at which reconfig Phase 4 Insert/Update DBC.Maps begins.               |

| View Column            | Data Type    | Format              | Comment   |
|------------------------|--------------|---------------------|---|
| BeginWrBMTimestamp     | TIMESTAMP(0) | YYYY-MM-DDBHH:MI:SS | Returns the timestamp at which reconfig Phase 4.9.1 (Update BMHashTbl GDO) begins.    |
| EndWrBMTimestamp       | TIMESTAMP(0) | YYYY-MM-DDBHH:MI:SS | Returns the timestamp at which reconfig Phase 4.9.1 (Update BMHashTbl GDO) ends.      |
| BeginVProcCfgTimestamp | TIMESTAMP(0) | YYYY-MM-DDBHH:MI:SS | Returns the timestamp at which the reconfig final phase (update VprocCfg GDO) begins. |
| EndVProcCfgTimestamp   | TIMESTAMP(0) | YYYY-MM-DDBHH:MI:SS | Returns the timestamp at which the final reconfig phase (Update VprocCfg GDO) ends.   |

## Usage Notes

This view provides the end user a way to access the ReconfigInfoTbl. Users with the appropriate privileges can SELECT or DELETE from the base table using the view.

### Note:

Only SELECT and/or DELETE privileges should be granted to users by user DBC.

## Possible Values for ReconfigType

| Value | Description      |
|-------|------------------|
| 1     | Online Reconfig  |
| 2     | Offline Reconfig |

## Possible Values for Status

- Start
- Redist
- Delete
- Checktable
- Abort
- Idle
- Pause
- Offline
- Complete
- RedistFail

- DeleteFail
- RcoFail

## ReconfigRedistOrderV

**Category:** Operations

**Database:** DBC

| View Column    | Data Type   | Format | Comment   |
|----------------|---|--------|---|
| OrderNumber    | INTEGER NOT NULL  | -(10)9 | A user-defined value that determines when the table is processed relative to other tables during the Deletion and Redistribution phases of reconfigurations.  |
| DatabaseName   | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | Return the user database name.  |
| TableName      | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | Return the name of a table in the database specified by DatabaseName.   |
| ProcessOffline | CHAR(1) LATIN<br>UPPERCASE NOT NULL                     | X(1)   | ProcessOffline indicates whether the table should be processed (deleted or redistributed) offline only, when logons are disabled and the system is quiescent. |

## Usage Notes

The ReconfigRedistOrderV view provides the end user a way to access the ReconfigRedistOrderTbl. Users with the appropriate privileges can SELECT, INSERT, UPDATE, or DELETE on the base table via the view.

### Note:

Only SELECT, INSERT, UPDATE, and/or DELETE should be granted to users by user DBC.

## Possible Values for ProcessOffline

### Note:

For the ProcessOffline column, tables undergoing large-scale changes should not be processed during online redistribution. Flag these tables for offline redistribution by setting this field to Y.

| Value | Description  |
|-------|--|
| Y     | Table should be processed (deleted or redistributed) offline during the offline portion of the Reconfig deletion or redistribution phase.              |
| N     | Table should be processed (deleted or redistributed) online during the online of the Reconfig deletion or redistribution phase. (This is the default.) |

## ReconfigTableStatsV

**Category:** Optimizer Statistics

**Database:** DBC

| View Column    | Data Type   | Format                   | Comment  |
|----------------|---|--------------------------|--|
| ReconfigId     | INTEGER<br>NOT NULL                                     | -(10)9                   | Returns the unique identification number for a reconfiguration. Every reconfiguration operation is automatically assigned this number. |
| Phase          | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL        | X(1)                     | Specify the reconfiguration phase. 'R' for "redistribution phase". 'D' for "deletion/rebuild NUSI phase".                              |
| TableId        | BYTE(6) NOT NULL  | X(12)                    | Return the ID of table in the database specified by DatabaseName.  |
| DatabaseName   | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                   | Return the user database name.   |
| TableName      | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                   | Return the name of a table in the database specified by DatabaseName.  |
| Status         | BYTEINT   | -(3)9                    | Specify the status of the table 0 for "Not Completed" 1 for "Completed" 2 for "Retry".   |
| BeginTimeStamp | TIMESTAMP(0)  | YYYY-MM-DD<br>BHH:MI:SS  | Returns the timestamp for the start of the reconfig phase for this table.  |
| EndTimeStamp   | TIMESTAMP(0)  | YYYY-MM-DD<br>BHH:MI:SS  | Returns the timestamp for the end of the reconfig phase for this table.  |
| TotRowCount    | FLOAT NOT NULL  | ----,---,---,---,<br>--9 | Returns the total number of rows processed during the time period defined by BeginTimeStamp and EndTimeStamp.                          |

| View Column         | Data Type      | Format                   | Comment   |
|---------------------|----------------|--------------------------|---|
| TotByteCount        | FLOAT NOT NULL | ----,---,---,---,<br>--9 | Returns the total number of bytes processed during the time period defined by BeginTimeStamp and EndTimeStamp.                                    |
| TotCPUSecs          | FLOAT NOT NULL | ----,---,---,---,<br>--9 | TotCPUSecs returns the total number of CPU seconds (with .001 resolution) used during the time period defined by BeginTimeStamp and EndTimeStamp. |
| TotIOCount          | FLOAT NOT NULL | ----,---,---,---,<br>--9 | Returns the total number of IOs done during the time period defined by BeginTimeStamp and EndTimeStamp.   |
| LowRowCount         | FLOAT NOT NULL | ----,---,---,---,<br>--9 | Returns the lowest count of rows processed by an AMP.   |
| LowRowCountAmp      | SMALLINT       | -(5)9                    | Returns the number of the AMP with the lowest row count.  |
| HighRowCount        | FLOAT NOT NULL | ----,---,---,---,<br>--9 | Returns the highest count of rows processed by an AMP.  |
| HighRowCountAmp     | SMALLINT       | -(5)9                    | Returns the number of the AMP with the highest row count.   |
| LowByteCount        | FLOAT NOT NULL | ----,---,---,---,<br>--9 | Returns the lowest count of bytes processed by an AMP.  |
| LowByteCountAmp     | SMALLINT       | -(5)9                    | Returns the number of the AMP with the lowest byte count.   |
| HighByteCount       | FLOAT NOT NULL | ----,---,---,---,<br>--9 | Returns the highest count of bytes processed by an AMP.   |
| HighByteCountAmp    | SMALLINT       | -(5)9                    | Returns the number of the AMP with the highest byte count.  |
| LowCPUSecsCount     | FLOAT NOT NULL | ----,---,---,---,<br>--9 | LowCPUSecsCount returns the lowest CPU seconds count (with .001 resolution) used by an AMP.   |
| LowCPUSecsCountAmp  | SMALLINT       | -(5)9                    | Returns the number of the AMP with the lowest CPU seconds count.  |
| HighCPUSecsCount    | FLOAT NOT NULL | ----,---,---,---,<br>--9 | HighCPUSecsCount returns the highest count of CPU seconds (with .001 resolution) used by an AMP.  |
| HighCPUSecsCountAmp | SMALLINT       | -(5)9                    | Returns the number of the AMP with the highest CPU seconds count.   |
| LowIOCount          | FLOAT NOT NULL | ----,---,---,---,<br>--9 | Returns the lowest count of IOs on an AMP.  |

| View Column      | Data Type                              | Format                   | Comment   |
|------------------|--|--------------------------|---|
| LowIOCountAmp    | SMALLINT                               | -(5)9                    | Returns the number of the AMP with the lowest IO count.                           |
| HighIOCount      | FLOAT NOT NULL                         | ----,---,---,---,<br>--9 | Returns the highest count of IOs on an AMP.                                       |
| HighIOCountAmp   | SMALLINT                               | -(5)9                    | Returns the number of the AMP with the highest IO count.                          |
| NUSICount        | SMALLINT<br>NOT NULL                   | -(5)9                    | Returns the number of NUSIs for the table.  |
| FallBackFlag     | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL | X(1)                     | FallBackFlag indicates whether the table is a FallBack or Non-FallBack table.     |
| DBlockSize       | INTEGER<br>NOT NULL                    | -(10)9                   | Returns the data block size (Default 0).  |
| SortFlag         | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL | X(1)                     | Tell if the reconfig uses RowNlns or SortTable during redistribution of the rows. |
| ActualRedistSecs | FLOAT NOT NULL                         | ----,---,---,---,<br>--9 | Returns the actual elapsed time in seconds for the Redistribution phase.          |
| ActualDeleteSecs | FLOAT NOT NULL                         | ----,---,---,---,<br>--9 | Returns the actual elapsed time in seconds for the Deletion/NUSI Rebuild phase.   |
| EstRedistSecs    | FLOAT NOT NULL                         | ----,---,---,---,<br>--9 | Returns the estimated Redistribution phase time in seconds.                       |
| EstDeleteSecs    | FLOAT NOT NULL                         | ----,---,---,---,<br>--9 | Returns the estimated Deletion/NUSI Rebuild elapsed time.                         |
| FSGIOCount       | FLOAT NOT NULL                         | ----,---,---,---,<br>--9 | Returns the count of IOs done at the reconfig FSG (File Segment Group) cache.     |
| FSysReadCount    | FLOAT NOT NULL                         | ----,---,---,---,<br>--9 | Returns the count of reconfig file system Read operations.                        |
| FSysWriteCount   | FLOAT NOT NULL                         | ----,---,---,---,<br>--9 | Returns the count of reconfig file system Write operations.                       |
| FSysMiscCount    | FLOAT NOT NULL                         | ----,---,---,---,<br>--9 | Returns the count of miscellaneous reconfig file system operations.               |
| MsgRcvCount      | FLOAT NOT NULL                         | ----,---,---,---,<br>--9 | Returns the count of messages received.   |
| MsgSendCount     | FLOAT NOT NULL                         | ----,---,---,---,<br>--9 | Returns the count of messages sent.   |

| View Column     | Data Type        | Format                   | Comment   |
|-----------------|------------------|--------------------------|---|
| MsgMiscCount    | FLOAT NOT NULL   | ----,---,---,---,<br>--9 | Returns the count of miscellaneous messaging operations.                      |
| MsgWaitRcvTime  | FLOAT NOT NULL   | ----,---,---,---,<br>--9 | Returns the total time elapsed waiting to receive messages.                   |
| MsgWaitSendTime | FLOAT NOT NULL   | ----,---,---,---,<br>--9 | Returns the total time elapsed sending messages.                              |
| MsgWaitMiscTime | FLOAT NOT NULL   | ----,---,---,---,<br>--9 | Returns the total elapsed time waiting on miscellaneous messaging operations. |
| NoMemFlushCount | INTEGER NOT NULL | -(10)9                   | Returns the count of memory flushes when the memory is full.                  |
| CkptFlushCount  | INTEGER NOT NULL | -(10)9                   | Returns the number of checkpoints flushed.                                    |
| CheckTableError | FLOAT NOT NULL   | ----,---,---,---,<br>--9 | Returns the number of checktable errors.                                      |

## Usage Notes

This view provides the end user a way to access the ReconfigTableStatsTbl. Users with the appropriate privileges can SELECT or DELETE from the base table via the view. Note that only SELECT and/or DELETE privileges should be granted to users by user DBC.

### DBlockSize

The default value for DBlockSize is zero.

### Possible Values for FallBackFlag

| Value | Description              |
|-------|--------------------------|
| Y     | Fallback table (default) |
| N     | Non-fallback table       |

### Possible Values for Phase

| Value | Description                 |
|-------|-----------------------------|
| R     | Redistribution phase        |
| D     | Deletion/rebuild NUSI phase |

**Possible Values for Status**

| Value | Description   |
|-------|---------------|
| 0     | Not completed |
| 1     | Completed     |
| 2     | Retry         |

**RepCaptureRulesV****Category:** Security**Database:** DBC

| View Column    | Data Type   | Format | Comment  |
|----------------|---|--------|--|
| RuleSetName    | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | Returns the mixed-case user-defined rule set name.   |
| GroupName      | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | Returns the name of a replication group.   |
| ObjectKind     | CHAR(1) LATIN<br>UPPERCASE NOT NULL                     | X(1)   | Returns the kind of dictionary object to which this rule is matched. Object kinds are encoded as TableKind in the DBC.TVM table, as follows: T = Table, V = View, M = Macro, G = Trigger, I = Join index, Hash index, C = Temporary Table. |
| DefaultOpt     | CHAR(1) LATIN<br>UPPERCASE NOT NULL                     | X(1)   | Indicates whether this is a default rule.  |
| LikePattern    | VARCHAR(300)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(300) | Returns a pattern string that is matched using the LIKE operator to the names of objects of the specified kind.  |
| LikeEscape     | CHAR(1) UNICODE<br>NOT CASESPECIFIC                     | X(1)   | Returns an optional escape character to be used with LikePattern.  |
| NotLikePattern | VARCHAR(300)<br>UNICODE<br>NOT CASESPECIFIC             | X(300) | Returns an optional pattern string. If it is not null, it is used to exclude objects that would otherwise match this rule.   |
| NotLikeEscape  | CHAR(1) UNICODE<br>NOT CASESPECIFIC                     | X(1)   | Returns an optional escape character to be used with NotLikePattern.   |

## Example: Using RepCaptureRulesV

The following SELECT statement selects all non-default rules defined for a table object:

```
SELECT GroupName, RulesetName, LikePattern, LikeEscape,
NotLikePattern, NotLikeEscape
FROM DBC.RepCaptureRulesV
WHERE DefaultOpt = 'N' AND ObjectKind = 'T'
ORDER BY GroupName, RulesetName, LikePattern, LikeEscape,
NotLikePattern, NotLikeEscape;
```

Result:

```
GroupName repgroup
RuleSetName prmtablers
LikePattern repuser.prmstab%
LikeEscape ?
NotLikePattern ?
NotLikeEscape ?
GroupName repgroup
RuleSetName subtablers
LikePattern repuser.subtab%
LikeEscape ?
NotLikePattern repuser.subtabx%
NotLikeEscape ?
```

## RepTablesV[X]

**Category:** Schema

**Database:** DBC

| View Column | Data Type                                      | Format | Comment  |
|-------------|--|--------|--|
| GroupName   | VARCHAR(128) UNICODE NOT CASESPECIFIC NOT NULL | X(128) | Returns the name of a replication group.   |
| TableName   | VARCHAR(128) UNICODE NOT CASESPECIFIC NOT NULL | X(128) | Returns the name of the table, join index, or hash index that belongs to this replication group. |

## ResolvedDTSV[X]

**Category:** Schema

**Database: DBC**

| View Column                   | Data Type   | Format                            | Comment   |
|-------------------------------|---|-----------------------------------|---|
| ResolvedCurrent_<br>Date      | DATE  | YY/MM/DD                          | Returns the last resolved value of CURRENT_DATE.  |
| ResolvedCurrent_<br>TimeStamp | TIMESTAMP(6) WITH<br>TIME ZONE                          | YYYY-MM-<br>DDBHH:MI:SS.<br>S(6)Z | Returns the last resolved value of CURRENT_TIMESTAMP.   |
| TableName                     | VARCHAR(128)<br>UNICODE<br>UPPERCASE<br>NOT NULL        | X(128)                            | Returns the name of a table, view, stored procedure, trigger, macro, user-defined types, user-defined methods, or user-defined function on which a privilege was granted. |
| DatabaseName                  | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                            | Returns the name of a database.   |

## Usage Notes

### Possible Values for ResolvedCurrent\_TimeStamp and ResolvedCurrent\_Date

- This is the last reconciled timestamp or date if the object is a join index or a table that is defined using:
  - CURRENT\_TIMESTAMP
  - CURRENT\_DATE or DATE

Either the partition, JI definition, or temporal table has a system-defined join index.
- NULL in all other cases.

## Example: Using ResolvedDTSV

The following SELECT statement selects the last resolved date from the ResolvedDTSV view:

```
SELECT ResolvedCurrent_Date
FROM ResolvedDTSV
WHERE TableName='Customer'
AND DatabaseName='Sales'
AND ResolvedCurrent_Date IS NOT NULL;
```

The query returns the following result:

ResolvedCurrent\_Date

-----

09/01/01

## RestrictedWordsV

**Category:** Security

**Database:** DBC

| View Column    | Data Type                                 | Format | Comment   |
|----------------|---|--------|---|
| RestrictedWord | VARCHAR(30) UNICODE<br>UPPERCASE NOT NULL | X(30)  | Returns a default list of restricted word. For a complete list of restricted words, see SQL Fundamentals. |

## RI\_Child\_TablesV[X]

**Category:** Integrity

**Database:** DBC

| View Column  | Data Type                                   | Format  | Comment   |
|--------------|---|---------|---|
| IndexID      | SMALLINT NOT NULL                           | ---,--9 | Returns the reference index number.<br>Note: This is not the same as IndexNumber. |
| IndexName    | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC | X(128)  | Returns the name of the reference index.  |
| ChildDbID    | BYTE(4) NOT NULL                            | X(8)    | Returns the database ID of the referencing table.                                 |
| ChildTID     | BYTE(6) NOT NULL                            | X(12)   | Returns the table ID of the referencing table.                                    |
| ChildKeyFID  | SMALLINT NOT NULL                           | ---,--9 | Returns the field ID of a column in the referencing key.                          |
| ParentDbID   | BYTE(4) NOT NULL                            | X(8)    | Returns the database ID of the referenced table.                                  |
| ParentTID    | BYTE(6) NOT NULL                            | X(12)   | Returns the table ID of the referenced table.                                     |
| ParentKeyFID | SMALLINT NOT NULL                           | ---,--9 | Returns the field ID of a column in the referenced key.                           |

| View Column       | Data Type   | Format              | Comment  |
|-------------------|---|---------------------|--|
| InconsistencyFlag | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL                  | X(1)                | Inconsistencies allowed between related parent-child object definitions after restore: Y (yes), N (no).                              |
| CreatorName       | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)              | Table or database creator. For DBC. AllRights, the grantor of explicit rights, otherwise the user who executed the CREATE statement. |
| CreateTimeStamp   | TIMESTAMP(0)  | YYYY-MM-DDBHH:MI:SS | Returns the date and time that the object in the row was created.  |
| VTFKType          | CHAR(1)<br>LATIN UPPERCASE                              | X(1)                | DBC.RI_Child_TablesVX.VTFKType defines the ValidTime dimension for the foreign key   |
| TTFKType          | CHAR(1)<br>LATIN UPPERCASE                              | X(1)                | DBC.RI_Child_TablesVX.TTFKType defines the TransactionTime dimension for the foreign key   |
| VTPKType          | CHAR(1)<br>LATIN UPPERCASE                              | X(1)                | DBC.RI_Child_TablesVX.VTPKType defines the ValidTime dimension for the parent key  |
| TTPKType          | CHAR(1)<br>LATIN UPPERCASE                              | X(1)                | DBC.RI_Child_TablesVX.TTPKType defines the TransactionTime dimension for the parent key  |

## Usage Notes

This view is similar to the All\_RI\_ChildrenV[X] view, but contains the IDs of databases, tables, and columns instead of the names for access control purposes. The administrator can control who has access to internal ID numbers by limiting the access to the RI\_Child\_Tables view while allowing more (or all) users to access the names via the All\_RI\_Children view.

Some of the column values show information related to a Teradata temporal table or an ANSI temporal table. For more information about these tables, see *Teradata Vantage™ ANSI Temporal Table Support*, B035-1186 and *Teradata Vantage™ Temporal Table Support*, B035-1182.

## Corresponding Tables

The corresponding tables for DBC.RI\_Child\_TablesV is DBC.ReferencingTbl.

The corresponding tables for DBC.RI\_Child\_TablesV[X] are:

- DBC.AccessRights
- DBC.Dbase
- DBC.Owners
- DBC.ReferencingTbls

- DBC.RoleGrants
- DBC.Roles

## InconsistencyFlag

If the value in the InconsistencyFlag column is Y, it may be possible to validate the reference indexes that have been marked as inconsistent.

## Possible Values for VTFKType

| Value | Description  |
|-------|--|
| A     | ANSIQUALIFIER<br><b>Note:</b><br>The Teradata temporal valid-time tables are compatible with ANSI valid-time tables, however to use ANSI temporal tables requires that the session temporal qualifier for systems using Teradata temporal tables be explicitly set to ANSIQUALIFIER. |
| C     | Current foreign key  |
| N     | Nonsequenced foreign key   |
| P     | Referential integrity constraint for ANSI Temporal tables. Temporal constraints can be defined for valid-time period column names with the special PERIOD keyword.   |
| R     | Temporal relationship constraint (TRC) referential integrity   |
| S     | Sequenced foreign key  |
| NULL  | The child table is non-temporal or does not support ValidTime.   |

## Possible Values for TTFKType

| Value | Description   |
|-------|---|
| C     | Current foreign key   |
| N     | Nonsequenced foreign key  |
| NULL  | A child table is a nontemporal table or a table that does not support TransactionTime |
| S     | Sequenced foreign key   |

## Possible Values for TTPKType

| Value | Description             |
|-------|-------------------------|
| C     | Current parent key      |
| S     | Sequenced parent key    |
| N     | Nonsequenced parent key |

| Value | Description  |
|-------|--|
| NULL  | If the parent table is a nontemporal table or the table does not support either ValidTime or TransactionTime |

### Possible Values for VTPKType

| Value | Description  |
|-------|--|
| A     | ANSIQUALIFIER  |
| C     | Current parent key   |
| S     | Sequenced parent key   |
| N     | Nonsequenced parent key  |
| NULL  | If the parent table is a nontemporal table or the table does not support either ValidTime or TransactionTime   |
| P     | Referential integrity constraint for ANSI Temporal tables. Temporal referential constraints can be defined for valid-time period column in temporal tables by specifying the valid-time derived PERIOD column names with the special PERIOD keyword. |

## RI\_Distinct\_ChildrenV[X]

**Category:** Integrity

**Database:** DBC

| View Column | Data Type   | Format  | Comment   |
|-------------|---|---------|---|
| IndexID     | SMALLINT NOT NULL                                       | ---,--9 | Returns the reference index number.<br>Note: This is not the same as IndexNumber. |
| IndexName   | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC             | X(128)  | Returns the name of the reference index.  |
| ChildDB     | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)  | Returns the name of the referencing database.                                     |
| ChildTable  | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)  | Returns the name of the referencing table.  |
| ParentDB    | VARCHAR(128)<br>UNICODE NOT                             | X(128)  | Returns the name of the referenced database.                                      |

| View Column       | Data Type   | Format                      | Comment   |
|-------------------|---|-----------------------------|---|
|                   | CASESPECIFIC<br>NOT NULL                                |                             |   |
| ParentTable       | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                      | Returns the name of the<br>referenced table.  |
| InconsistencyFlag | CHAR(1) LATIN<br>UPPERCASE NOT NULL                     | X(1)                        | Inconsistencies allowed between<br>related parent-child object definitions<br>after restore: Y (yes), N (no).                                 |
| CreatorName       | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                      | Table or database creator. For DBC.<br>AllRights, the grantor of explicit rights,<br>otherwise the user who executed the<br>CREATE statement. |
| CreateTimeStamp   | TIMESTAMP(0)  | YYYY-MM-<br>DDBHH:MI:<br>SS | Returns the date and time that the<br>object in the row was created.  |

## Usage Notes

### Corresponding Tables

The corresponding tables for DBC.RI\_Distinct\_ChildrenV are:

- DBC.DBase
- DBC.ReferencingTbls
- DBC.TVM

The corresponding tables for DBC.RI\_Distinct\_ChildrenV[X] are:

- DBC.AccessRights
- DBC.DBase
- DBC.Owners
- DBC.ReferencingTbls
- DBC.RoleGrants
- DBC.Roles
- DBC.TVM

## RI\_Distinct\_ParentsV[X]

**Category:** Integrity

**Database:** DBC

| View Column       | Data Type   | Format              | Comment  |
|-------------------|---|---------------------|--|
| IndexID           | SMALLINT NOT NULL                                       | ---,--9             | Returns the reference index number.<br>Note: This is not the same as IndexNumber.  |
| IndexName         | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC             | X(128)              | Returns the name of the reference index.   |
| ParentDB          | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)              | Returns the name of the referenced database.   |
| ParentTable       | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)              | Returns the name of the referenced table.  |
| ChildDB           | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)              | Returns the name of the referencing database.  |
| ChildTable        | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)              | Returns the name of the referencing table.   |
| InconsistencyFlag | CHAR(1) LATIN<br>UPPERCASE NOT NULL                     | X(1)                | Inconsistencies allowed between related parent-child object definitions after restore: Y (yes), N (no).                              |
| CreatorName       | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)              | Table or database creator. For DBC. AllRights, the grantor of explicit rights, otherwise the user who executed the CREATE statement. |
| CreateTimeStamp   | TIMESTAMP(0)  | YYYY-MM-DDBHH:MI:SS | Returns the date and time that the object in the row was created.  |

## Usage Notes

### Corresponding Tables

The corresponding tables for DBC.RI\_Distinct\_ParentsV are:

- DBC.Dbase
- DBC.ReferencedTbls
- DBC.TVM

The corresponding tables for DBC.RI\_Distinct\_ParentsV[X] are:

- DBC.AccessRights
- DBC.Dbase
- DBC.Owners
- DBC.ReferencedTbls
- DBC.RoleGrants
- DBC.Roles
- DBC.TVM

## RI\_Parent\_TablesV[X]

**Category:** Integrity

**Database:** DBC

| View Column       | Data Type   | Format              | Comment  |
|-------------------|---|---------------------|--|
| IndexID           | SMALLINT NOT NULL                                       | ---,--9             | Returns the reference index number.<br>Note: This is not the same as IndexNumber.  |
| IndexName         | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC             | X(128)              | Returns the name of the reference index.   |
| ParentDbID        | BYTE(4) NOT NULL  | X(8)                | Returns the database ID of the referenced table.   |
| ParentTID         | BYTE(6) NOT NULL  | X(12)               | Returns the table ID of the referenced table.  |
| ParentKeyFID      | SMALLINT NOT NULL                                       | ---,--9             | Returns the field ID of a column in the referenced key.  |
| ChildDbID         | BYTE(4) NOT NULL  | X(8)                | Returns the database ID of the referencing table.  |
| ChildTID          | BYTE(6) NOT NULL  | X(12)               | Returns the table ID of the referencing table.   |
| ChildKeyFID       | SMALLINT NOT NULL                                       | ---,--9             | Returns the field ID of a column in the referencing key.   |
| InconsistencyFlag | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL                  | X(1)                | Inconsistencies allowed between related parent-child object definitions after restore: Y (yes), N (no).                              |
| CreatorName       | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)              | Table or database creator. For DBC. AllRights, the grantor of explicit rights, otherwise the user who executed the CREATE statement. |
| CreateTimeStamp   | TIMESTAMP(0)  | YYYY-MM-DDBHH:MI:SS | Returns the date and time that the object in the row was created.  |

| View Column | Data Type                  | Format | Comment   |
|-------------|----------------------------|--------|---|
| VTPKType    | CHAR(1)<br>LATIN UPPERCASE | X(1)   | DBC.RI_Parent_TablesVX.VTPKType defines the ValidTime dimension for the parent key        |
| TTPKType    | CHAR(1)<br>LATIN UPPERCASE | X(1)   | DBC.RI_Parent_TablesVX.TTPKType defines the TransactionTime dimension for the parent key  |
| VTFKType    | CHAR(1)<br>LATIN UPPERCASE | X(1)   | DBC.RI_Parent_TablesVX.VTFKType defines the ValidTime dimension for the foreign key       |
| TTFKType    | CHAR(1)<br>LATIN UPPERCASE | X(1)   | DBC.RI_Parent_TablesVX.TTFKType defines the TransactionTime dimension for the foreign key |

## Usage Notes

The RI\_Parent\_Tables view is similar to the All\_RI\_Parents view, but contains the IDs of databases, tables, and columns instead of the names for access control purposes.

The administrator can control who has access to internal ID numbers by limiting the access to the RI\_Parent\_Tables view while allowing more (or all) users to access the names via the All\_RI\_Parents view.

Some of the column values show information related to a Teradata temporal table or an ANSI temporal table. For more information about these tables, see *Teradata Vantage™ ANSI Temporal Table Support*, B035-1186 and *Teradata Vantage™ Temporal Table Support*, B035-1182.

## Corresponding Tables

The corresponding table for DBC.RI\_Parent\_TablesV is DBC.ReferencedTbls.

The corresponding tables for DBC.RI\_Parent\_TablesV[X] are:

- DBC.AccessRights
- DBC.DBase
- DBC.Owners
- DBC.ReferencedTbls
- DBC.RoleGrants
- DBC.Roles

## InconsistencyFlag

If the value in the InconsistencyFlag column is Y, it may be possible to validate the reference indexes that have been marked as inconsistent.

**Possible Values for TTFKType**

| Value | Description   |
|-------|---|
| C     | Current foreign key   |
| N     | Nonsequenced foreign key  |
| NULL  | A child table is a nontemporal table or a table that does not support TransactionTime |
| S     | Sequenced foreign key   |

**Possible Values for TTPKType**

| Value | Description  |
|-------|--|
| C     | Current parent key   |
| S     | Sequenced parent key   |
| N     | Nonsequenced parent key  |
| NULL  | If the parent table is a nontemporal table or the table does not support either ValidTime or TransactionTime |

**Possible Values for VTFKType**

| Value | Description  |
|-------|--|
| A     | ANSIQUALIFIER<br><b>Note:</b><br>The Teradata temporal valid-time tables are compatible with ANSI valid-time tables, however to use ANSI temporal tables requires that the session temporal qualifier for systems using Teradata temporal tables be explicitly set to ANSIQUALIFIER. |
| C     | Current foreign key  |
| N     | Nonsequenced foreign key   |
| P     | Referential integrity constraint for ANSI Temporal tables. Temporal constraints can be defined for valid-time period column names with the special PERIOD keyword.   |
| R     | Temporal relationship constraint (TRC) referential integrity   |
| S     | Sequenced foreign key  |
| NULL  | The child table is non-temporal or does not support ValidTime.   |

**Possible Values for VTPKType**

| Value | Description  |
|-------|--|
| A     | ANSIQUALIFIER  |
| C     | Current parent key   |
| S     | Sequenced parent key   |
| N     | Nonsequenced parent key  |
| NULL  | If the parent table is a nontemporal table or the table does not support either ValidTime or TransactionTime   |
| P     | Referential integrity constraint for ANSI Temporal tables. Temporal referential constraints can be defined for valid-time period column in temporal tables by specifying the valid-time derived PERIOD column names with the special PERIOD keyword. |

**RoleInfoV[X]****Category:** Security**Database:** DBC

| View Column     | Data Type   | Format              | Comment  |
|-----------------|---|---------------------|--|
| RoleName        | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)              | Returns the name of a role.  |
| CreatorName     | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC             | X(128)              | Returns the Teradata user who issued the CREATE ROLE statement.  |
| CommentString   | VARCHAR(255)<br>UNICODE<br>NOT CASESPECIFIC             | X(255)              | Returns user-supplied text or commentary on the column, database, table, view, macro, user-defined function, user-defined types, user-defined methods, stored procedure, role, profile, or user. |
| CreateTimeStamp | TIMESTAMP(0)  | YYYY-MM-DDBHH:MI:SS | Returns the date and time that the object in the row was created.  |
| ExtRole         | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL                  | X(1)                | Stores comma-delimited names of directory-assigned roles.  |

## Example: Using RoleInfoV

The following SELECT statement returns the names of the role creators:

```
SELECT rolename (char(8)), creatorname (char(8)), commentstring (char(20)),
createtimestamp, ExtRole from DBC.RoleInfoV;
```

Result:

| RoleName | CreatorName | CommentString  | CreateTimeStamp     | ExtRole |
|----------|-------------|----------------|---------------------|---------|
| -----    | -----       | -----          | -----               | -----   |
| r1       | u1          | Comments on r1 | 2002-08-13 10:26:19 | N       |
| r2       | u2          | Comments on r2 | 2002-08-13 10:26:25 | Y       |

## RoleMembersV[X]

**Category:** Security

**Database:** DBC

| View Column | Data Type   | Format                      | Comment   |
|-------------|---|-----------------------------|---|
| RoleName    | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                      | Returns the name of a role.   |
| Grantor     | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC             | X(128)                      | Returns the name of the user who granted the role.  |
| WhenGranted | TIMESTAMP(0)  | YYYY-MM-DD<br>BHH:MI:<br>SS | Returns the date a role was granted.  |
| DefaultRole | VARCHAR(1) UNICODE<br>NOT CASESPECIFIC                  | X(1)                        | Returns whether a role is a default role for the grantee. If the grantee is a USER and has the role as his default role, DefaultRole is 'Y'; otherwise, the value is 'N'. |
| WithAdmin   | CHAR(1) LATIN<br>UPPERCASE NOT NULL                     | X(1)                        | Returns 'Y' if Admin Option comes with the role grant; otherwise, the value is 'N'.   |

## Usage Notes

Roles without members are excluded from the result set of RoleMembers.

## Grantee and GranteeKind

These columns are not returned in the X or VX views, so they are not listed in the table because it shows the columns for RoleMembersVX.

These additional columns are returned in the V view:

- Grantee returns the name of a user who was granted a privilege; ALL can be specified. Grantee returns the name of a user or role who was granted a role.
- GranteeKind returns the kind of a grantee, either User or Role.

## Possible Values for WithAdmin

| Value | Description |
|-------|-------------|
| Y     | Yes         |
| N     | No          |

## Example: Using RoleMembersV

The following query lists all roles and their members:

```
SELECT CAST(rolename AS CHAR(15)),
       CAST(grantee AS CHAR(20)),
       CAST(grantor AS CHAR(20)),
       whengranted
FROM DBC.RoleMembersV ORDER BY 1,2;
```

Result:

| RoleName | Grantee | Grantor | WhenGranted       |
|----------|---------|---------|-------------------|
| -----    | -----   | -----   | -----             |
| role_a   | user_1  | DBA     | 97-10-15 14:32:59 |
| role_a   | user_2  | DBA     | 97-10-15 11:00:01 |
| role_b   | user_1  | SysFe   | 98-04-24 09:10:15 |
| . . .    | . . .   | . . .   | . . .             |

## SecConstraintsV[X]

**Category:** Integrity

**Database:** DBC

| View Column     | Data Type                                     | Format              | Comment   |
|-----------------|---|---------------------|---|
| ConstraintName  | VARCHAR(128)<br>UNICODE UPPERCASE<br>NOT NULL | X(128)              | Returns the name of the table-level check. This field is NULL if it is unnamed. |
| DataType        | CHAR(2) LATIN<br>UPPERCASE<br>NOT NULL        | X(2)                | Returns the data type for the security constraint object.                       |
| Nullable        | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL        | X(1)                | Returns a code to indicate whether or not a column may have a null value.       |
| SizeInBytes     | SMALLINT NOT NULL                             | -(5)9               | Returns the maximum number of bytes allowed in the column.                      |
| AssigneeCount   | SMALLINT NOT NULL                             | -(5)9               | Returns the number instances of the assignment of the security constraint.      |
| Creator         | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC   | X(128)              | Returns the name of the user who created the security constraint.               |
| CreateTimeStamp | TIMESTAMP(0)<br>NOT NULL                      | YYYY-MM-DDBHH:MI:SS | Returns the date and time that the object in the row was created.               |

## Usage Notes

### Possible Values for DataType

| Value | Description |
|-------|-------------|
| I1    | Byte array  |
| I2    | Smallint    |

### Possible Values for Nullable

| Value | Description |
|-------|-------------|
| Y     | Yes         |
| N     | No          |

## SecurityDefaultsV

**Category:** Security

**Database:** DBC

| View Column           | Data Type                              | Format  | Comment   |
|-----------------------|--|---------|---|
| ExpirePassword        | SMALLINT<br>NOT NULL                   | ---,--9 | Returns the number of days to elapse before the password expires. 0 indicates the password does not expire.   |
| PasswordMinChar       | BYTEINT<br>NOT NULL                    | -(3)9   | Returns the minimum number of characters in a valid password string.  |
| PasswordMaxChar       | BYTEINT<br>NOT NULL                    | ---9    | Returns the maximum number of characters in a valid password string. PasswordMaxChar must be equal to or greater than PasswordMinChar.                    |
| PasswordDigits        | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL | X(1)    | To define whether digits are allowed in a password string. A value of Y indicates yes while N indicates no. The default is Y.                             |
| PasswordSpecChar      | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL | X(1)    | Returns a code to indicate which special characters are to be allowed in the password. For the definition of the codes, see DBC.SysSecDefaults.           |
| PasswordRestrictWords | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL | X(1)    | Indicates whether to restrict certain words from being contained within a password string. These are the valid values: Y (Yes), N (No). The default is N. |
| MaxLogonAttempts      | BYTEINT<br>NOT NULL                    | -(3)9   | Number of erroneous logons before the user is locked. 0 (user never locked).  |
| LockedUserExpire      | SMALLINT<br>NOT NULL                   | ---,--9 | Returns the number of minutes to elapse before a locked user is unlocked. 0 indicates immediate unlock. -1 indicates user is locked indefinitely.         |
| PasswordReuse         | SMALLINT<br>NOT NULL                   | ---,--9 | Returns the number of days to elapse before a password can be reused. 0 indicates immediate reuse.  |

## Related Topics

For more information about controlling access, space, and ownership, see the following documents:

- *Teradata Vantage™ - Database Administration*, B035-1093
- *Teradata Vantage™ - Database Design*, B035-1094
- *Teradata Vantage™ - Advanced SQL Engine Security Administration*, B035-1100

## SecurityLogV[X]

**Category:** Security

**Database:** DBC

| View Column  | Data Type   | Format   | Comment  |
|--------------|---|----------|--|
| LogDate      | DATE NOT NULL   | YY/MM/DD | Returns the date that the access log entry was made.   |
| LogTime      | FLOAT NOT NULL  | 99:99:99 | Returns the time of day that the event occurred as HH:MM:SS.   |
| LogType      | SMALLINT  | ---,--9  | Returns the kind of statement for which the access log entry was made.   |
| UserName     | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)   | Returns the name of the user for whom the log entry was made.  |
| AccountName  | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)   | Expanded account in effect when a request was submitted. SYSTEMACCOUNTNO tracks console utility activity such as table rebuild, Diskcopy, or Scandisk. |
| DatabaseName | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)   | Returns the name of the database or user of the object for which this log entry was made.  |
| TableName    | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC             | X(128)   | Returns the name of the table, view, stored procedure, or macro for this log entry.  |
| Text         | VARCHAR(8192)<br>UNICODE<br>NOT CASESPECIFIC            | X(8192)  | Text for the numbered event in the error log.  |

## Usage Notes

### Corresponding Tables

The corresponding table for DBC.SecurityLogV is DBC.AccLogTbl.

The corresponding tables for DBC.SecurityLogV[X] are:

- DBC.AccLogTbl
- DBC.DBase
- DBC.Owners

This view also references the userdb view.

---

#### Note:

The DatabaseName column was previously named ObjectName.

---

## Possible Values for Partition

| Value    | Description  |
|----------|--|
| DBC/SQL  | SQL Session  |
| EXPORT   | FASTEXPORT   |
| FASTLOAD | FASTLOAD session   |
| HUTPARSE | ARC data session   |
| MLOAD    | MULTILOAD session  |
| MONITOR  | Sessions running in a performance monitoring application |
| NONE     | Session is recognized but not yet assigned               |

## ServerInfoV[X]

**Category:** Operations

**Database:** DBC

| View Column | Data Type                                      | Format   | Comment   |
|-------------|--|----------|---|
| ServerName  | VARCHAR(128) UNICODE NOT CASESPECIFIC NOT NULL | X(128)   | Returns the name of the FOREIGN SERVER.   |
| NameInfo    | VARCHAR(128) UNICODE NOT CASESPECIFIC NOT NULL | X(128)   | Returns the name portion specified in the custom-clause.  |
| ValueInfo   | VARCHAR(31000) UNICODE NOT CASESPECIFIC        | X(31000) | Returns the value portion specified in the custom-clause.   |
| ValueType   | CHAR(1) LATIN NOT CASESPECIFIC                 | X(1)     | Return the identifier to identify whether the value specified in custom-clause is either SELECT/Value/System. |
| NVPTYPE     | VARCHAR(7) UNICODE NOT CASESPECIFIC            | X(7)     | Returns the name value pair type.   |

## Usage Notes

This Teradata QueryGrid connector Data Dictionary view provides details about the name value pairs used by foreign servers defined in the Vantage system.

### NameInfo and ValueInfo

NameInfo returns the name portion specified in the USING custom clause. ValueInfo returns the value portion specified in the USING custom clause.

### Possible Values for NVPType

NVPType returns the name value pair type or UNKNOWN:

- IMPORT
- EXPORT
- GLOBAL
- UNKNOWN

### Possible Values for ValueType

ValueType identifies what type of value is specified in the USING custom clause.

| Value | Description   |
|-------|---|
| NULL  | ValueInfo column contains data value.                             |
| F     | Indicates a foreign table.  |
| R     | ValueInfo column contains a resolved query text mentioned in NVP. |
| S     | ValueInfo column contains query text mentioned in NVP query.      |
| V     | ValueInfo column contains a system variable.                      |

## ServerV[X]

**Category:** Operations

**Database:** DBC

| View Column  | Data Type   | Format | Comment   |
|--------------|---|--------|---|
| ServerID     | BYTE(6) NOT NULL  | X(12)  | Returns the identifier of the FOREIGN SERVER (TVMIId column in DBC. TVM).             |
| DataBaseName | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | Returns the name of the database or user in which the subject foreign server resides. |
| ServerName   | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | Returns the name of the FOREIGN SERVER.   |
| CreatorName  | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | Returns the Teradata User who issued the CREATE FOREIGN SERVER statement              |

| View Column        | Data Type  | Format              | Comment  |
|--------------------|--|---------------------|--|
| CreateTimeStamp    | TIMESTAMP(0)   | YYYY-MM-DDBHH:MI:SS | Returns the time when CREATE FOREIGN SERVER statement was issued.  |
| LastAlterName      | VARCHAR(128)<br>UNICODE NOT CASESPECIFIC<br>NOT NULL | X(128)              | Returns the name of the user who last updated the dictionary row.  |
| LastAlterTimeStamp | TIMESTAMP(0)   | YYYY-MM-DDBHH:MI:SS | Returns the time the dictionary row was last updated.  |
| AuthorizationName  | VARCHAR(128)<br>UNICODE<br>UPPERCASE                 | X(128)              | Returns the authorization name specified in the external security clause of the foreign server.                    |
| AuthorizationType  | VARCHAR(15)<br>UNICODE<br>NOT CASESPECIFIC           | X(15)               | Returns the type of authorization INVOKER TRUSTED or DEFINER TRUSTED.  |
| ExecMapName        | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC          | X(128)              | Returns the execution map name of a foreign server. NULL indicates the foreign server does not have a execute map. |
| ExecMapColocName   | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC          | X(128)              | Returns the ColocationName of the execution map of a foreign server.   |

## Usage Notes

This Teradata QueryGrid connector Data Dictionary view provides details about the foreign servers defined in the system.

### Possible Values for AuthorizationType

| Value | Description     |
|-------|-----------------|
| T     | Invoker Trusted |
| S     | Definer Trusted |
| ''    | Unknown         |

### Example of ServerV[X]

The following SELECT statement returns information about the foreign server objects created by user 'dba'.

```
select * from DBC.ServerV where CreatorName = 'dba';
```

Result:

```

      ServerID 000011960000
      DataBaseName TD_SERVER_DB
      ServerName SERVER_1
      CreatorName dba
      CreateTimeStamp 2014-12-02 19:51:46
      LastAlterName dba
      LastAlterTimeStamp 2014-12-02 19:51:46
      ServerID 000012960000
      DataBaseName TD_SERVER_DB
      ServerName SERVER_2
      CreatorName dba
      CreateTimeStamp 2014-12-02 19:51:50
      LastAlterName dba
      LastAlterTimeStamp 2014-12-02 19:51:50
      AuthorizationName user1
      AuthorizationType INVOKER TRUSTED

```

## SessionInfoV[X]

**Category:** Accounting

**Database:** DBC

| View Column     | Data Type   | Format             | Comment  |
|-----------------|---|--------------------|--|
| UserName        | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)             | Returns the Teradata Database<br>userid of the user who is currently<br>logged on.   |
| AccountName     | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)             | Expanded account in effect<br>when a request was submitted.<br>SYSTEMACCOUNTNO tracks<br>console utility activity such as table<br>rebuild, Diskcopy, or Scandisk. |
| SessionNo       | INTEGER NOT<br>NULL                                     | --,---,---,<br>--9 | Returns the session identifier<br>assigned to the session by the TDP<br>or LAN interface.  |
| DefaultDataBase | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)             | Returns the name of the current<br>default database for the user.  |

| View Column      | Data Type                             | Format      | Comment  |
|------------------|---------------------------------------|-------------|--|
| IFPNo            | SMALLINT NOT NULL                     | -(5)9       | Returns the vproc number of the PE through which the session was connected or assigned.  |
| Partition        | CHAR(16) LATIN NOT CASESPECIFIC       | X(16)       | Returns the name of the Teradata Database partition to which the user is currently attached.   |
| LogicalHostId    | SMALLINT NOT NULL                     | -(5)9       | Returns a unique identifier of the logon source for the logged query. A value of zero indicates an internal session.   |
| HostNo           | SMALLINT NOT NULL                     | ---,--9     | Returns the number of the client system through which the user logged on to the Teradata Database.   |
| CurrentCollation | CHAR(1) LATIN UPPERCASE NOT NULL      | X(1)        | Returns the current collation of the session.  |
| LogonDate        | DATE NOT NULL                         | YY/MM/DD    | Returns the date that the session for which the log entry was made was logged on to the Teradata Database.   |
| LogonTime        | FLOAT NOT NULL                        | 99:99:99.99 | Returns the time on which logon for the session occurred (useful on logoff events).  |
| LogonSequenceNo  | BYTE(4)                               | X(8)        | Returns the logon sequence number of the session.  |
| LogonSource      | VARCHAR(128) UNICODE NOT CASESPECIFIC | X(128)      | Channel-Attached Systems Using the CLIV2 API Returns the origin of the CLIV2 mainframe session being reported, such as the user ID or session number of the client system. |
| ExpiredPassword  | CHAR(1) LATIN UPPERCASE               | X(1)        | Returns a code that indicates whether or not the defined session is in the process of a logon for a user with an expired password.   |
| TwoPCMode        | VARCHAR(1) LATIN NOT CASESPECIFIC     | X(1)        | Returns one of the following codes: 2 = 2PC mode, N = non-2PC mode.  |
| Transaction_Mode | VARCHAR(1) LATIN NOT CASESPECIFIC     | X(1)        | Returns one of the following codes to indicate the mode of the session: T = TDBS, A = ANSI.  |
| CurrentRole      | VARCHAR(128) UNICODE NOT CASESPECIFIC | X(128)      | Returns the current role for a session.  |

| View Column         | Data Type   | Format  | Comment   |
|---------------------|---|---------|---|
| ProfileName         | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC             | X(128)  | Returns the name of the profile.  |
| LogonAcct           | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)  | Returns the account string used when a session is established.  |
| LDAP                | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL                  | X(1)    | The LDAP column denotes how a directory user for a given session is mapped. The column displays one of these options: P means "permanent user", X means "external user" and N means "not externally authenticated". |
| AuditTrailId        | VARCHAR(128)<br>UNICODE<br>UPPERCASE<br>NOT NULL        | X(128)  | Returns the identifier that is used for access logging.   |
| CurlIsolationLevel  | VARCHAR(16)<br>UNICODE NOT<br>CASESPECIFIC              | X(16)   | Returns the isolation level; SR - Serializable; RU - Read uncommitted; RC - Read Committed; RR - Repeatable Read.   |
| QueryBand           | VARCHAR(4096)<br>UNICODE NOT<br>CASESPECIFIC            | X(4096) | Returns the band under which the query is submitted.  |
| ProxyUser           | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC             | X(128)  | Returns the name of the proxy user.   |
| ProxyCurRole        | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC             | X(128)  | Returns the current role of the proxy user.   |
| TemporalQualifier   | VARCHAR(1024)<br>LATIN NOT<br>CASESPECIFIC              | X(1024) | DBC.SESSIONINFOVX. TemporalQualifier is used to reset the qualifier in the session across a database restart  |
| CalendarName        | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)  | The SESSIONINFOVX. CalendarName column describes the calendar set in the session  |
| ExtendedLogonSource | VARCHAR(2048)<br>UNICODE NOT<br>CASESPECIFIC            | X(2048) | The ExtendedLogonSource column is an identification of the place from where the user accessed the system  |
| ClientIpAddress     | VARCHAR(45)<br>UNICODE NOT<br>CASESPECIFIC              | X(45)   | The ClientIpAddress column is the standard text representation of the   |

| View Column          | Data Type                                 | Format             | Comment   |
|----------------------|---|--------------------|---|
|                      |   |                    | IP address where the user accessed the system   |
| ClientProgramName    | VARCHAR(1024)<br>UNICODE NOT CASESPECIFIC | X(1024)            | The ClientProgramName is the client system program name   |
| ClientSystemUserId   | VARCHAR(128)<br>UNICODE NOT CASESPECIFIC  | X(128)             | The ClientSystemUserId is the client user id  |
| ClientConnectionType | BYTEINT                                   | -(3)9              | The ClientConnectionType is 1 for TCP/IP or 2 for Channel connect   |
| ClientCoordName      | VARCHAR(128)<br>UNICODE NOT CASESPECIFIC  | X(128)             | The ClientCoordName is the Coordinator name for clients using CICS or IMS                                       |
| ClientEnvName        | VARCHAR(128)<br>UNICODE NOT CASESPECIFIC  | X(128)             | The ClientEnvName is the TDP Environment Name   |
| ClientJobId          | VARCHAR(128)<br>UNICODE NOT CASESPECIFIC  | X(128)             | The ClientJobId is the TDP Job ID   |
| ClientJobName        | VARCHAR(128)<br>UNICODE NOT CASESPECIFIC  | X(128)             | The ClientJobName is the TDP Job Name   |
| ClientOsName         | VARCHAR(128)<br>UNICODE NOT CASESPECIFIC  | X(128)             | The ClientOsName is the TDP Operating System Name   |
| ClientProcThreadId   | VARCHAR(128)<br>UNICODE NOT CASESPECIFIC  | X(128)             | The ClientProcThreadId is the client process/thread identifier  |
| ClientSecProdGrp     | VARCHAR(128)<br>UNICODE NOT CASESPECIFIC  | X(128)             | The ClientSecProdGrp is the TDP Security Product Group  |
| ClientSecProdUserId  | VARCHAR(128)<br>UNICODE NOT CASESPECIFIC  | X(128)             | The ClientSecProdUserId is TDP Security Product Userid  |
| ClientTcpPortNumber  | INTEGER                                   | --,---,---,<br>--9 | The ClientTcpPortNumber is the TCP port number on the client system   |
| ClientTdHostName     | VARCHAR(128)<br>UNICODE NOT CASESPECIFIC  | X(128)             | The ClientTdHostName is the Teradata Database hostname that the client used to connect to the Teradata Database |

| View Column                    | Data Type                                   | Format | Comment   |
|--------------------------------|---|--------|---|
| ClientTerminalId               | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128) | The ClientTerminalId is TDP<br>Terminal Identifier                                    |
| ClientTransactionId            | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128) | The ClientTransactionId is TDP<br>Transaction Identifier                              |
| ClientUserOperId               | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128) | The ClientUserOperId is the TDP<br>User/Operator Identifier                           |
| ClientVmName                   | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128) | The ClientVmName is the TDP<br>Virtual Machine Name                                   |
| ClientVmUserId                 | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128) | The ClientVmUserId is the TDP<br>Virtual Machine User Id                              |
| MechanismName                  | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128) | The MechanismName is<br>authentication method used to<br>authenticate the user        |
| ClientTDPReleaseId             | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128) | The ClientTDPReleaseId is the TDP<br>Release Identifier                               |
| ClientCLlv2ReleaseId           | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128) | The ClientCLlv2ReleaseId is the<br>CLlv2 Release Identifier                           |
| ClientSessionDesc              | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128) | The ClientSessionDesc is the TDP<br>session description                               |
| ClientWorkload                 | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128) | The ClientWorkload is the<br>TDP Workload   |
| ClientJobData                  | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128) | The ClientJobData is the client<br>Job Data from the LSINFO<br>environmental variable |
| ClientODBCDriverVersion        | VARCHAR(16)<br>UNICODE NOT<br>CASESPECIFIC  | X(16)  | The ClientODBCDriverVersion is<br>the client ODBC Driver Version                      |
| ClientNetDataProviderVersion   | VARCHAR(16)<br>UNICODE NOT<br>CASESPECIFIC  | X(16)  | The ClientNetDataProviderVersion<br>is the client .NET Data<br>Provider Version       |
| ClientODBCDriverManagerVersion | CHAR(30)<br>UNICODE NOT<br>CASESPECIFIC     | X(30)  | The<br>ClientODBCDriverManagerVersion<br>is the client ODBC Driver<br>Manager Version |

| View Column                | Data Type                                   | Format             | Comment   |
|----------------------------|---|--------------------|---|
| ClientNetFrameworkVersion  | CHAR(30)<br>UNICODE NOT<br>CASESPECIFIC     | X(30)              | The ClientNetFrameworkVersion is the client .NET Framework Version  |
| ClientAttributesEx         | VARCHAR(512)<br>UNICODE NOT<br>CASESPECIFIC | X(512)             | The ClientAttributesEx field contains extra description of the client that do not match any of the other Client Attributes fields   |
| ClientJDBCDriverVersion    | VARCHAR(16)<br>UNICODE NOT<br>CASESPECIFIC  | X(16)              | The ClientJDBCDriverVersion is the client JDBC Driver Version   |
| ClientJavaVersion          | VARCHAR(30)<br>UNICODE NOT<br>CASESPECIFIC  | X(30)              | The ClientJavaVersion is the client Java Framework Version  |
| ExportDefinitionName       | VARCHAR(30)<br>LATIN NOT<br>CASESPECIFIC    | X(30)              | The ExportDefinitionName is the current ExportDefinitionName of the session that is inherited from the user when a session is logged on.  |
| ExportWidthRuleSet         | BYTE(20)                                    | X(40)              | The ExportWidthRuleSet is the current ExportWidthRuleSet of the session that is inherited from the user when a session is logged on. It is maintained for the duration of the session (even if the user is modified while the session is active). |
| RecoverableNetworkProtocol | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL      | X(1)               | The RecoverableNetworkProtocol column indicates whether the client supports the RecoverableNetwork client-database interface  |
| LogonRedrive               | VARCHAR(33)<br>UNICODE NOT<br>CASESPECIFIC  | X(33)              | The LogonRedrive column indicates the sessions participation in Redrive   |
| ClientIPAddrByClient       | VARCHAR(45)<br>UNICODE NOT<br>CASESPECIFIC  | X(45)              | Client IP Address by Client   |
| ClientPortByClient         | INTEGER                                     | --,---,---,<br>--9 | Client Port by Client   |
| ServerIPAddrByClient       | VARCHAR(45)<br>UNICODE NOT<br>CASESPECIFIC  | X(45)              | Serve IP Address by Client  |
| ServerPortByClient         | INTEGER                                     | --,---,---,<br>--9 | Server Port by Client   |
| ClientIPAddrByUnity        | VARCHAR(45)<br>UNICODE NOT<br>CASESPECIFIC  | X(45)              | Client IP Address by Unity  |

| View Column               | Data Type                                    | Format             | Comment                         |
|---------------------------|--|--------------------|---------------------------------|
| ClientPortByUnity         | INTEGER                                      | --,---,---,<br>--9 | Client Port by Unity            |
| UnityClientSideIPAddr     | VARCHAR(45)<br>UNICODE NOT<br>CASESPECIFIC   | X(45)              | Unity Client Side IP Address    |
| UnityClientSidePort       | INTEGER                                      | --,---,---,<br>--9 | Unity Client Side Port          |
| UnityServerSideIPAddr     | VARCHAR(45)<br>UNICODE NOT<br>CASESPECIFIC   | X(45)              | Unity Server Side IP Address    |
| UnityServerSidePort       | INTEGER                                      | --,---,---,<br>--9 | Unity Server Side Port          |
| ServerIPAddrByUnity       | VARCHAR(45)<br>UNICODE NOT<br>CASESPECIFIC   | X(45)              | Server IP Address by Unity      |
| ServerPortByUnity         | INTEGER                                      | --,---,---,<br>--9 | Server Port by Unity            |
| ServerIPAddrByServer      | VARCHAR(45)<br>UNICODE NOT<br>CASESPECIFIC   | X(45)              | Server IP Address by Server     |
| ServerPortByServer        | INTEGER                                      | --,---,---,<br>--9 | Server Port by Server           |
| ClientCOPSuffixedHostName | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC  | X(128)             | Client COP Suffixed Hostname    |
| UnitySessNo               | INTEGER                                      | --,---,---,<br>--9 | Unity Session Number            |
| UnityVersion              | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC  | X(128)             | Unity Version                   |
| UnityAuthMechName         | VARCHAR(1571)<br>UNICODE NOT<br>CASESPECIFIC | X(1571)            | Unity Authorized Mechanism Name |
| UnityMechanismName        | VARCHAR(1571)<br>UNICODE NOT<br>CASESPECIFIC | X(1571)            | Unity Mechanism Name            |
| UserAuthenticatedBy       | VARCHAR(1)<br>UNICODE NOT<br>CASESPECIFIC    | X(1)               | User Authenticated by           |

| View Column               | Data Type                                   | Format             | Comment  |
|---------------------------|---|--------------------|--|
| ClientTDSessionPoolName   | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128)             | Client TD Session Pool Name  |
| UnityTcpPortNumber        | INTEGER                                     | --,---,---,<br>--9 | Unity TCP Port Number  |
| UnityIpAddress            | VARCHAR(45)<br>UNICODE NOT<br>CASESPECIFIC  | X(45)              | Unity IP Address   |
| TTGranularity             | VARCHAR(30)<br>UNICODE NOT<br>CASESPECIFIC  | X(30)              | The SESSIONINFOVX.<br>TTGranularity column describes the<br>Transaction Time Granularity set in<br>the session   |
| LoadingOp                 | CHAR(1) LATIN<br>UPPERCASE                  | X(1)               | The LoadingOp is set to "T" when<br>session isolated loading is enabled<br>and "F" when session isolated<br>loading is disabled.   |
| UnicodePassThrough        | CHAR(1) LATIN<br>UPPERCASE                  | X(1)               | The SessionInfoV.<br>UnicodePassThrough returns the<br>Unicode Pass Through attribute of<br>the session.   |
| DirUserNetConfidentiality | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC        | X(1)               | This column contains the user's<br>network confidentiality policy that<br>applies on the connection between<br>the client and the gateway or<br>between the client and Unity,<br>obtained from the LDAP directory<br>used for security policy. |
| DirUserNetPolicyLevel     | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC        | X(1)               | When DirUserNetConfidentiality is<br>"I" or "C", this column contains the<br>level of protection required, obtained<br>by lookup in the LDAP directory<br>used for security policy.  |
| UnityNetConfidentiality   | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC        | X(1)               | This column contains the user's<br>network confidentiality policy that<br>applies on the connection between<br>Unity and the gateway, obtained<br>from the LDAP directory used for<br>security policy.   |
| UnityNetPolicyLevel       | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC        | X(1)               | When UnityNetConfidentiality is "I"<br>or "C", this column contains the level<br>of protection required, obtained by<br>lookup in the LDAP directory used<br>for security policy.  |
| EffectiveSessionNetConf   | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC        | X(1)               | This column contains the effective<br>network confidentiality policy for<br>the session, that applies on the   |

| View Column                    | Data Type                             | Format | Comment   |
|--------------------------------|---------------------------------------|--------|---|
|                                |                                       |        | connection between the client and the gateway or between the client and Unity, integrated from all sources of security policy that do not change.   |
| EffectiveSessionNetPolicyLevel | CHAR(1) LATIN NOT CASESPECIFIC        | X(1)   | When EffectiveSessionNetConfidentiality is "I" or "C", this column contains the level of protection required, integrated from all sources of security policy that do not change within a session. |
| ProxyLogon                     | CHAR(1) LATIN NOT CASESPECIFIC        | X(1)   | The ProxyLogon column indicates T if logon request uses Passwordless Proxy for TD2.   |
| ClientConfType                 | CHAR(1) LATINC                        | X(1)   | The ClientConfType column indicates Client Confidentiality Type.  |
| ServerConfType                 | CHAR(1) LATINC                        | X(1)   | The ServerConfType column indicates Server Confidentiality Type.  |
| UnityConfType                  | CHAR(1) LATINC                        | X(1)   | The UnityConfType column indicates Unity Confidentiality Type.  |
| ServerUnityConfType            | CHAR(1) LATINC                        | X(1)   | The ServerUnityConfType column indicates Server Unity Confidentiality Type.   |
| ClientConfVersion              | VARCHAR(16) LATIN NOT CASESPECIFIC    | X(16)  | The ClientConfVersion column indicates Client TLS version number.   |
| ClientConfCipherSuite          | VARCHAR(64) LATIN NOT CASESPECIFIC    | X(64)  | The ClientConfCipherSuite column indicates Client TLS cipher suite.   |
| UnityConfVersion               | VARCHAR(16) LATIN NOT CASESPECIFIC    | X(16)  | The UnityConfVersion column indicates Unity TLS version number.   |
| UnityConfCipherSuite           | VARCHAR(64) LATIN NOT CASESPECIFIC    | X(64)  | The UnityConfCipherSuite column indicates Unity TLS cipher suite.   |
| TimeZoneString                 | VARCHAR(128) UNICODE NOT CASESPECIFIC | X(128) | Returns the time zone string set for the user session.  |

## Usage Notes

Information about current session pools, which are a collection of sessions that are logged on to the database under the same logonid, may be accessed by entering the DISPLAY POOL command. For more information on DISPLAY POOL, see *Teradata® Director Program Reference*, B035-2416.

### LogonSource

Teradata recommends using alternative columns instead of the LogonSource column, if available. For information about the recommended columns for LogonSource, see "LogonSource Column Fields and Examples."

### ClientConfCipherSuite and UnityConfCipherSuite

ASCII String representing the confidentiality cipher suite. Currently, this is the TLS cipher suite, for example: "TLS\_AES\_256\_GCM\_SHA384".

### ClientConfVersion and UnityConfVersion

ASCII String representing the confidentiality version number. Currently, this is the TLS version number, for example: "TLS 1.2".

### Possible Values for ClientConfType

| Value | Description   |
|-------|---|
| C     | TLS used for encryption. Client validated the Certificate-Authority chain but ignored the Subject-Alternative-Name and the Common-Name.   |
| E     | TDGSS used for encryption. The application does not have the option to change this during the session.  |
| F     | TLS was attempted but failed, so this is a fallback to using TDGSS for encryption because ENCRYPTDATA is specified.   |
| H     | TLS was attempted but failed, so this is a fallback to unencrypted because ENCRYPTDATA is not specified.  |
| O     | May be encrypted using TDGSS or unencrypted. The application has the option of changing this at any time. This is possible only with CLv2 apps, because drivers (e.g. ODBC, JDBC) do not toggle encryption on and off within a session. |
| R     | TLS used for encryption. Server certificate was ignored; client did not validate the identity of the server.  |
| U     | Unencrypted. The application does not have the option to change this during the session.  |
| V     | TLS used for encryption. Client validated the Certificate-Authority chain and the Subject-Alternative-Name or the Common-Name.  |

**Possible Values for ClientConnectionType**

| Value | Description   |
|-------|---|
| 1     | Client is connected using TCP/IP via the gateway.                   |
| 2     | Client is connected from a mainframe via a mainframe-attached host. |

**Possible Values for CurrentCollation**

| Value | Description   |
|-------|---------------|
| A     | ASCII         |
| E     | EBCDIC        |
| H     | Host          |
| M     | Multinational |
| C     | CharSet_Coll  |
| J     | JIS_Coll      |

**Possible Values for DirUserNetConfidentiality, UnityNetConfidentiality, EffectiveSessionNetConf**

When these columns are set to I or C it indicates the level of protection required.

| Value | Description   |
|-------|---|
| I     | Indicates the level of protection required for Integrity, which is obtained by lookup in the LDAP directory used for security policy. The levels are: Default (D), Low (L), Medium (M), and High (H).       |
| C     | Indicates the level of protection required for Confidentiality, which is obtained by lookup in the LDAP directory used for security policy. The levels are: Default (D), Low (L), Medium (M), and High (H). |

**Possible Values for ExpiredPassword**

| Value | Description   |
|-------|---|
| Y     | Yes, by Session Control procedures                            |
| N     | No, by the Parser when a new password is assigned to the user |

**Possible Values for LogonRedrive**

| Value | Description       |
|-------|-------------------|
| ' '   | Not participating |

| Value                         | Description  |
|-------------------------------|--|
| MEMORY NON-FALLBACK RESPONSES | Memory-based Redrive participation   |
| NULL or blanks                | Session is not participating in Redrive and database restarts will not be transparent to applications and users. |

### Possible Values for LoadingOp

| Value | Description   |
|-------|---|
| T     | Session isolated loading is enabled. This is the default setting for a session. |
| F     | Session isolated loading is disabled.   |

### Possible Values for Partition

| Value | Description   |
|-------|---|
| 7     | Console Utility Partition number  |
| 8     | Partition in which DBC console procedures will be started by the Host utility |
| 9     | File System Partition number  |
| 10    | Gateway partition number  |
| 11    | Worker Task Partition number  |
| 12    | Session Control Partition number  |
| 13    | Dispatcher Partition number   |
| 14    | Parser Partition number   |
| 15    | Startup Partition number  |
| 17    | Resource Sampling Subsystem (RSS) StartUp Partition number                    |
| 18    | Distributed Database File (DDF) Server Partition number                       |
| 19    | Relay Services Gateway (RSG) Partition number                                 |
| 47    | Replication Gateway rsgdbsmain partition number                               |

### Possible Values for ProxyLogon

| Value | Description  |
|-------|--|
| T     | The ProxyLogon column indicates if Unity has logged a user onto a TD2 session using existing credentials, when that user was successfully logged on by another Unity-managed Vantage system using TD2. When this occurs, ProxyLogon is set to T. |

| Value | Description   |
|-------|---|
| F     | False indicates the TD2 session logged on or attempted to log on with a valid password. |

### Possible Values for ServerConfType

| Value | Description   |
|-------|---|
| E     | TDGSS used for encryption, either enforced by policy or asserted by a Client.                             |
| O     | May be encrypted using TDGSS or unencrypted, as asserted by a Client, or because it cannot be determined. |
| T     | TLS used for encryption.  |
| U     | Unencrypted, as asserted by a Client Interface.   |

### Possible Values for ServerUnityConfType

| Value | Description   |
|-------|---|
| E     | TDGSS used for encryption, either enforced by policy or asserted by Unity.        |
| O     | May be encrypted using TDGSS or unencrypted. Cannot be determined by the gateway. |
| T     | TLS used for encryption.  |
| U     | Unencrypted, as asserted by Unity.  |

### Possible Values for TTGranularity

| Value       | Description   |
|-------------|---|
| LOGICALROW  | Row is timestamped with the time the row is processed by the AMP.   |
| REQUEST     | Row is timestamped with the time the request is submitted.  |
| TRANSACTION | Row is timestamped with the time when the first non-locking reference is made to a temporal table, or when the built-in function TEMPORAL_TIMESTAMP is first accessed during the transaction. |

### Possible Values for UnicodePassThrough

| Value | Description  |
|-------|--|
| S     | UNICODE PASS THROUGH session attribute is enabled.     |
| F     | UNICODE PASS THROUGH session attribute is not enabled. |

## Possible Values for UnityConfType

| Value | Description   |
|-------|---|
| E     | TDGSS used for encryption.  |
| F     | TLS was attempted but the handshake failed, so this is an attempt to fallback to using TDGSS for encryption. This is otherwise equivalent to "E". |
| T     | TLS used for encryption.  |
| U     | Unencrypted.  |

## Example: Using SessionInfoV

The following SELECT statement displays information on all current session.

```
SELECT UserName, SessionNo, DefaultDatabase, LogonSource
FROM DBC.SessionInfoV;
```

Result:

```
UserName  SessionNo  DefaultDatabase  LogonSource
-----
DBC        1,005      DBC              <TCP/IP> EB9F 141.206.1.84
```

## Related Topics

| For information about ...  | See ...   |
|--|---|
| interpreting the ExportWidthRuleSet column                           | <a href="#">ExportWidthV</a> .  |
| using the DBSControl utility to make export width definition changes | <i>Teradata Vantage™ - Advanced SQL Engine International Character Set Support</i> , B035-1125. |

## SettingsV

**Category:** TDMaps

**Database:** TDMaps

| View Column | Data Type                                      | Format | Comment      |
|-------------|--|--------|--------------|
| SettingName | VARCHAR(64) LATIN NOT CASESPECIFIC<br>NOT NULL | X(64)  | Setting name |

| View Column  | Data Type                           | Format | Comment               |
|--------------|-------------------------------------|--------|-----------------------|
| DefaultValue | VARCHAR(64) LATIN NOT CASESPECIFIC  | X(64)  | Default setting value |
| Description  | VARCHAR(128) LATIN NOT CASESPECIFIC | X(128) | Setting description   |

## Usage Notes

SettingsV stores settings that are used by the stored procedures and views.

### Note:

Changes to SettingsTbl should only be made to the DefaultValue column. Please contact Teradata Support for assistance before making changes to SettingsTbl.

## Possible Values for DefaultValue

| Setting                     | Default Value | Description   |
|-----------------------------|---------------|---|
| ANALYZER_LOG_LEVEL          | 1             | Level of logging into AnalyzeLogTbl. 0 means logging is turned off; 1 means logging is on.              |
| GROUPING_WEIGHT_FACTOR_COST | 1.0           | Weight factor in Grouping Tables - Step Cost  |
| GROUPING_WEIGHT_FACTOR_FREQ | 1.0           | Weight factor in Grouping Tables - Frequency  |
| MOVER_LOG_LEVEL             | 1             | Level of logging into LogTbl. 0 means logging is turned off; 1 means logging is on.                     |
| MULTI_AMP_MAP_FACTOR        | 1.0           | The number of AMPs in a multi-AMP sparse map is the number of nodes divided by this factor.             |
| SINGLE_AMP_MAP_SIZE_FACTOR  | 1.0           | The additional size limit multiplier for a table in a one-AMP sparse map.                               |
| SMALL_TABLE_SIZE_LIMIT      | 131072        | The maximum per-AMP size for a table in a sparse map.   |
| TABLE_HEADER_SIZE           | 1024          | The size of the table header for a simple table with a few columns.                                     |
| USE_PEAK_PERM               | Y             | If Y, then the peak perm value is used for sparse table candidates, otherwise the current perm is used. |

## SHOWCOLCHECKSV[X]

**Category:** Integrity

**Database:** DBC

| View Column     | Data Type   | Format              | Comment  |
|-----------------|---|---------------------|--|
| DatabaseName    | VARCHAR(128)<br>UNICODE UPPERCASE<br>NOT NULL           | X(128)              | Returns the name of the database containing a table with a column-level check.   |
| TableName       | VARCHAR(128)<br>UNICODE UPPERCASE<br>NOT NULL           | X(128)              | Returns the name of a table having column-level check constraints.   |
| ColumnName      | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)              | Returns the column name having column-level checks.  |
| ColCheck        | VARCHAR(8192)<br>UNICODE<br>NOT CASESPECIFIC            | X(8192)             | Returns the unresolved text for the column-level check condition.  |
| CreatorName     | VARCHAR(128)<br>UNICODE UPPERCASE<br>NOT NULL           | X(128)              | Table or database creator. For DBC. AllRights, the grantor of explicit rights, otherwise the user who executed the CREATE statement. |
| CreateTimeStamp | TIMESTAMP(0)  | YYYY-MM-DDBHH:MI:SS | Returns the date and time that the object in the row was created.  |
| TimeDimension   | CHAR(1)<br>LATIN UPPERCASE                              | X(1)                | Returns the ValidTime and TransactionTime properties for a period column.  |
| VTCheckType     | CHAR(1)<br>LATIN UPPERCASE                              | X(1)                | Returns the ValidTime dimension information for the CHECK condition.   |
| TTCheckType     | CHAR(1)<br>LATIN UPPERCASE                              | X(1)                | Returns the TransactionTime dimension for the CHECK constraint on a table with TransactionTime.                                      |

## Usage Notes

Some of the column values show information related to a Teradata temporal table or an ANSI temporal table. For more information about these tables, see *Teradata Vantage™ ANSI Temporal Table Support*, B035-1186 and *Teradata Vantage™ Temporal Table Support*, B035-1182.

For more information about the possible values for the TimeDimension column, see "TimeDimension Column."

## Corresponding Tables

The corresponding tables for DBC.ShowColChecksV are:

- DBC.DBase
- DBC.TVFields

- DBC.TVM

The corresponding tables for DBC.ShowColChecksV[X] are:

- DBC.DBase
- DBC.Owners
- DBC.RoleGrants
- DBC.Roles
- DBC.TVFields
- DBC.TVM

### Possible Values for TTCheckType

| Value | Description   |
|-------|---|
| A     | ANSIQUALIFIER<br><b>Note:</b><br>The TTCheckType column returns the value A when the qualifier is ANSIQUALIFIER for column-level CHECK constraints. |
| NULL  | No transaction-time dimension   |
| C     | CURRENT TRANSACTIONTIME   |

### Possible Values for VTCheckType

| Value | Description  |
|-------|--|
| A     | ANSIQUALIFIER.<br><b>Note:</b><br>The VTCheckType column returns the value A when the qualifier is ANSIQUALIFIER for column-level CHECK constraints. |
| NULL  | No valid-time dimension  |
| C     | CURRENT VALIDTIME  |
| S     | SEQUENCED VALIDTIME  |
| N     | NONSEQUENCED VALIDTIME   |

## SHOWTBLCHECKSV[X]

**Category:** Integrity

**Database:** DBC

| View Column     | Data Type                                     | Format              | Comment  |
|-----------------|---|---------------------|--|
| DatabaseName    | VARCHAR(128)<br>UNICODE UPPERCASE<br>NOT NULL | X(128)              | Returns the name of the database containing a table with a table-level check.  |
| TableName       | VARCHAR(128)<br>UNICODE UPPERCASE<br>NOT NULL | X(128)              | Returns the name of a table having table-level check constraints.  |
| CheckName       | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC   | X(128)              | Returns the name of the table-level check. This field is NULL if this is an unnamed table check.                                     |
| TblCheck        | VARCHAR(16000)<br>UNICODE<br>CASESPECIFIC     | X(16000)            | Returns the unresolved text for the table-level check condition.   |
| CreatorName     | VARCHAR(128)<br>UNICODE UPPERCASE<br>NOT NULL | X(128)              | Table or database creator. For DBC. AllRights, the grantor of explicit rights, otherwise the user who executed the CREATE statement. |
| CreateTimeStamp | TIMESTAMP(0)                                  | YYYY-MM-DDBHH:MI:SS | Returns the date and time that the object in the row was created.  |
| VTCheckType     | CHAR(1)<br>LATIN UPPERCASE                    | X(1)                | Returns the ValidTime dimension information for the CHECK condition.   |
| TTCheckType     | CHAR(1)<br>LATIN UPPERCASE                    | X(1)                | Returns the TransactionTime dimension for the CHECK constraint on a table with TransactionTime.                                      |

## Usage Notes

You can use this view to query for table constraints that are defined for a database.

For information about the possible values for the VTCheckType column, see "VTCheckType Column."

## Corresponding Tables

The X view references these additional tables:

- DBC.Owners
- DBC.RoleGrants
- DBC.Roles

## Possible Values for TTCheckType

| Value | Description                   |
|-------|-------------------------------|
| NULL  | No transaction-time dimension |

| Value | Description             |
|-------|-------------------------|
| C     | CURRENT TRANSACTIONTIME |

## Software\_Event\_LogV

**Category:** Operations

**Database:** DBC

| View Column | Data Type                          | Format       | Comment  |
|-------------|------------------------------------|--------------|--|
| TheDate     | DATE NOT NULL                      | YY/MM/DD     | Returns the calendar date on which the event was logged.   |
| TheTime     | FLOAT NOT NULL                     | 99:99:99.99  | The Software_Event_LogV.TheTime field gives the time of the event in the form hh:mm:ss.  |
| Event_Tag   | INTEGER NOT NULL                   | Z99-99999-99 | The Software_Event_LogV.Event_Tag field has the form of 10 digit decimal rbbeeee hh: r 0..3 reserved bb 0..99 Subsystem ID eeeee 0..99999 Event ID- Unique error code hh 0..99 Event Subcategory |
| Category    | BYTEINT                            | -(3)9        | Shows the category code assigned to a software event.  |
| Severity    | SMALLINT                           | -(5)9        | The Severity option returns a code identifying the severity of a software event.   |
| PMA         | INTEGER                            | ZZZ9-9999    | Identifies the Processor Module Assembly (PMA) on which the event occurred.  |
| Vproc       | INTEGER                            | -(10)9       | Identifies the virtual processor for which an event was logged.  |
| Partition   | BYTEINT                            | -(3)9        | Returns the name of the Teradata Database partition to which the user is currently attached.   |
| Task        | SMALLINT                           | -(5)9        | Returns the unique task number assigned to each task as it is created for execution.   |
| TheFunction | VARCHAR(32) LATIN NOT CASESPECIFIC | X(32)        | Returns the identification string for the entity reporting a software event.   |
| SW_Version  | VARCHAR(64) LATIN NOT CASESPECIFIC | X(64)        | The Software_Event_LogV.SW_Version field is the version number of OS or TPA.   |
| Line        | BYTEINT NOT NULL                   | -(3)9        | Text line number for a multi-line error message.   |
| Text        | VARCHAR(30000) UNICODE NOT         | X(30000)     | (Title: Message Text) Text for the logged event.   |

| View Column | Data Type                | Format | Comment |
|-------------|--------------------------|--------|---------|
|             | CASESPECIFIC<br>NOT NULL |        |         |

## Usage Notes

### Possible Values for Category

| Value | Description      |
|-------|------------------|
| 0     | None             |
| 1     | CPU hardware     |
| 2     | Memory hardware  |
| 3     | TDN hardware     |
| 4     | Disk hardware    |
| 5     | Channel hardware |
| 6     | Host             |
| 7     | Driver           |
| 8     | Resource         |
| 9     | System           |
| 10    | User             |
| 11    | Occurrence       |
| 12    | Abnormal         |
| 52    | Base TP          |
| 64    | Max TPSys        |

### Possible Values for Severity

| Value | Description   |
|-------|---|
| 0     | Hardware and software are informational events (Occurrence) |
| 10    | Software abnormality -- informational (SW abnormality)      |
| 20    | Hardware abnormality -- informational (HW abnormality)      |
| 30    | Recoverable user error (UserError retry)                    |
| 40    | Unrecoverable user error (UserError)                        |

| Value | Description   |
|-------|---|
| 50    | Unrecoverable user error, no user restart (UserFatal)             |
| 60    | Unrecoverable Vproc error, Vproc restart required (VprocError)    |
| 70    | Unrecoverable Vproc error, no Vproc restart (VprocFatal)          |
| 80    | Recoverable PDE error (OSErrorRetry)                              |
| 90    | Unrecoverable PDE error, restart required (OSError)               |
| 100   | Unrecoverable PDE error, restart not possible (OSFatal)           |
| 110   | Unrecoverable CPU hardware error, restart required (CPUError)     |
| 120   | Unrecoverable CPU hardware error, CPU stay down (CPUFatal)        |
| 130   | Unrecoverable PMA hardware error, PMA restart required (PMAError) |
| 140   | Unrecoverable PMA hardware error, PMA stays down (PMAFatal)       |
| 200   | Unrecoverable system error, system restart required (SystemError) |
| 210   | Unrecoverable System error, restart not possible (SystemFatal)    |

## Example: Using Software\_Event\_LogV

The following statement requests the software event log information for any event with a severity level of 50 (unrecoverable user error, no user restart):

```
SELECT TheDate, TheTime, Category, Severity
FROM Software_Event_LogV
Where Severity = '50' ;
```

Result:

| TheDate  | TheTime  | Category | Severity |
|----------|----------|----------|----------|
| -----    | -----    | -----    | -----    |
| 92/08/20 | 10:10:30 | 4        | 50       |

## SparseMapAmpsV

**Category:** Schema

**Database:** DBC

| View Column  | Data Type                             | Format | Comment   |
|--------------|---------------------------------------|--------|---|
| DatabaseName | VARCHAR(129) UNICODE NOT CASESPECIFIC | X(129) | The name of the database where the object is from.                            |
| ObjectName   | VARCHAR(129) UNICODE NOT CASESPECIFIC | X(129) | The name of the object.   |
| ObjectKind   | VARCHAR(1) LATIN NOT CASESPECIFIC     | X(1)   | The objectkind of the object.   |
| AmpNo        | SMALLINT                              | -(5)9  | The amp number for the object.  |
| PF           | CHAR(2) LATIN NOT CASESPECIFIC        | X(2)   | Output AMP type of the AMP number - Primary (P) or Fallback (F) or both (PF). |

## Usage Notes

The SparseMapAMPsV view returns the AMPs on which a table, join index, or hash index with a sparse map is defined.

## StatsV

**Category:** Optimizer Statistics

**Database:** DBC

| View Column  | Data Type                                      | Format   | Comment  |
|--------------|--|----------|--|
| DatabaseName | VARCHAR(128) UNICODE NOT CASESPECIFIC NOT NULL | X(128)   | The DatabaseName column is the name of the database in which the table resides.                              |
| TableName    | VARCHAR(128) UNICODE NOT CASESPECIFIC NOT NULL | X(128)   | The TableName column is the name of the containing table.  |
| ColumnName   | VARCHAR(10000) UNICODE NOT CASESPECIFIC        | X(10000) | The ColumnName column identifies a column or columns.  |
| FieldIdList  | VARCHAR(1000) LATIN UPPERCASE                  | X(1000)  | The FieldIdList column identifies List of fieldids on which statistics are collected separated by semicolon. |
| StatsName    | VARCHAR(128) UNICODE NOT CASESPECIFIC          | X(128)   | The StatsName column contains the alias name associated with the statistic                                   |

| View Column           | Data Type                                 | Format                   | Comment  |
|-----------------------|---|--------------------------|--|
| ExpressionCount       | SMALLINT                                  | ---,--9                  | The ExpressionCount column is Number of expressions statistics is collected on   |
| StatsId               | INTEGER<br>NOT NULL                       | --,---,---,--9           | The StatsId column is the Statistics identifier within each source.  |
| StatsType             | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL    | X(1)                     | The StatsType column is the statistics type. Possible values are T (Table), I (Join Index), N (Hash Index), V (View), Q (Query), L (Link Row). |
| StatsSource           | CHAR(1)<br>LATIN UPPERCASE                | X(1)                     | The StatsSource column records the method this statistic is acquired.  |
| ValidStats            | CHAR(1)<br>LATIN UPPERCASE                | X(1)                     | The ValidStats column indicates whether the statistics are valid or not.   |
| DBSVersion            | VARCHAR(32)<br>LATIN UPPERCASE            | X(32)                    | Returns the version of the database that contains the objects on which the statistics were collected.  |
| IndexNumber           | SMALLINT                                  | ---,--9                  | The IndexNumber column is the Index number of the index on which statistics are collected.   |
| SampleSignature       | VARCHAR(256)<br>LATIN NOT<br>CASESPECIFIC | X(256)                   | The SampleSignature column is the Sample options encoded as a 10 character signature.  |
| SampleSizePct         | DECIMAL(5,2)                              | ----.99                  | The SampleSizePct column is the Sample size percent used when collecting statistics.   |
| ThresholdSignature    | VARCHAR(512)<br>LATIN NOT<br>CASESPECIFIC | X(512)                   | The ThresholdSignature column is the Threshold options encoded as a 17 character signature.  |
| MaxIntervals          | SMALLINT                                  | ---,--9                  | The MaxIntervals column is the User-specified maximum number of intervals.   |
| MaxValueLength        | INTEGER                                   | --,---,---,--9           | The MaxValueLength column is the User-specified maximum value length.  |
| RowCount              | FLOAT                                     | ----,---,---,---,<br>--9 | The RowCount column is the row count of the table, view or query on which statistics are collected.  |
| UniqueValueCount      | FLOAT                                     | ----,---,---,---,<br>--9 | The UniqueValueCount column is the Number of unique values of the ExpressionList.  |
| PNullUniqueValueCount | FLOAT                                     | ----,---,---,---,<br>--9 | The PNullUniqueValueCount column is the Number of unique values from rows with partial nulls.  |

| View Column          | Data Type    | Format                  | Comment  |
|----------------------|--------------|-------------------------|--|
| NullCount            | FLOAT        | ---,---,---,---,<br>--9 | The NullCount column is the Number of nulls of the ExpressionList.   |
| AllNullCount         | FLOAT        | ---,---,---,---,<br>--9 | The AllNullCount column is the Number of all nulls of the ExpressionList.  |
| HighModeFreq         | FLOAT        | ---,---,---,---,<br>--9 | The HighModeFreq column is the highest frequency of values of the ExpressionList.  |
| PNullHighModeFreq    | FLOAT        | ---,---,---,---,<br>--9 | The PNullHighModeFreq column is the highest frequency of values having partial nulls of the ExpressionList.              |
| StatsSkipCount       | INTEGER      | --,---,---,--9          | The StatsSkipCount column indicates how many times the statistics collection on the ExpressionList has been skipped.     |
| CreateTimeStamp      | TIMESTAMP(0) | YYYY-MM-DDBHH:MI:SS     | The CreateTimeStamp column is the statistics creation time stamp.  |
| LastCollectTimeStamp | TIMESTAMP(0) | YYYY-MM-DDBHH:MI:SS     | The LastCollectTimeStamp column is the Last statistics collection time stamp.  |
| LastAlterTimeStamp   | TIMESTAMP(0) | YYYY-MM-DDBHH:MI:SS     | The LastAlterTimeStamp column is the last user updated time stamp.   |
| BLCCompRatio         | INTEGER      | --,---,---,--9          | The estimated space savings percentage for primary subtable of manually compressed BLC tables.                           |
| AvgRowSize           | FLOAT        | ---,---,---,---,<br>--9 | The AvgRowSize column is the average row size of an object on which statistics are collected.                            |
| BLCCompFactor        | FLOAT        | ---,---,---,---,<br>--9 | The estimated block compression factor for the entire table. This is used for calculating the Customer Data Space (CDS). |

## Usage Notes

### ColumnName

- If more than one column or expression is specified, each column or expression is separated by a comma.
- The maximum number of columns is 64.

- If expressions are in the list, the maximum number of columns can be reduced past the limit of 64, depending on the combined total size of the text in the expressions.
- If the combined total size of the expression text causes the maximum column limit to be less than the actual number of columns in the list, an error occurs.

### FieldIdList

The FieldIdList column is NULL for fields that involve expressions.

### MaxInterval and MaxValueLength

If these statistics are collected with system determined maximum intervals and maximum value length, the MaxInterval and MaxValueLength columns are NULL.

### Possible Values for StatsType

| Value | Description |
|-------|-------------|
| T     | Table       |
| I     | Join Index  |
| N     | Hash Index  |
| V     | View        |
| Q     | Query       |
| L     | Link Row    |

### SampleSizePct

If sampling is not used, the SampleSizePct column is set to 0 or 100.

### StatsSource

The StatsSource column records the method by which this statistic is acquired. For information about the possible values for the StatsSource column, see "StatsSource Column."

## To Get Information Not Contained in This View

This view does not contain:

- Statistics on tables protected by row-level security. To get these statistics, create views on DBC.StatsTbl. You can view details about this table in Teradata SQL Assistant or Teradata Studio Express.
- Information about column attributes. To get this information, join the DBC.StatsTbl table to the DBC.TVFields table. You can view details about this table in Teradata SQL Assistant or Teradata Studio Express.

## Example: Using StatsV

This example assumes the following statistics have been collected:

```
STATISTICS
  INDEX (o_orderkey)
  ,INDEX (o_custkey, o_orderstatus)
  ON Orders;
```

This query can be used to retrieve the statistics:

```
SELECT * FROM dbc.StatsV
  WHERE databasename = 'sales'
     AND tablename = 'orders';
```

## Related Topics

| For information about statistics collected on ...       | See ...                             |
|---|-------------------------------------|
| non-indexed columns and single-column indexes           | <a href="#">ColumnsV[X]</a> .       |
| indexes for which two or more columns have been defined | <a href="#">IndexStatsV[X]</a> .    |
| multiple columns  | <a href="#">MultiColumnStatsV</a> . |
| tables  | <a href="#">TableStatsV</a> .       |
| materialized temporary tables                           | <a href="#">TempTableStatsV</a> .   |
| single expressions                                      | <a href="#">ExpStatsV</a> .         |
| multiple expressions                                    | <a href="#">MultiExpStatsV</a> .    |

## StatUseCountV[X]

**Category:** Accounting

**Database:** DBC

| View Column  | Data Type   | Format | Comment  |
|--------------|---|--------|--|
| DatabaseName | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | Returns the Database Name of the object for which access count and/or UDI counts are recorded. |
| TableName    | VARCHAR(128)<br>UNICODE NOT                             | X(128) | Returns the name of the table that got used statistics.  |

| View Column         | Data Type                                     | Format                         | Comment  |
|---------------------|---|--------------------------------|--|
|                     | CASESPECIFIC<br>NOT NULL                      |                                |  |
| StatName            | VARCHAR(10000)<br>UNICODE<br>NOT CASESPECIFIC | X(10000)                       | Returns the name of statistics that are used for some query.     |
| LastAccessTimeStamp | TIMESTAMP(0)                                  | YYYY-MM-DD<br>HH:MI:SS         | Return the latest access time of the object.                     |
| AccessCount         | BIGINT  | --,---,---,---,---,<br>---,--9 | Returns the number of accesses since the last reset by the user. |

## Example: Using StatUseCountV[X]

The following SELECT statement shows the number of statistics accesses ocarina on a particular table:

```
SELECT StatName, AccessCount FROM DBC.StatUseCountV WHERE DatabaseName =
'Personnel' AND TableName = 'Employee';
```

The query returns the following result:

```
StatName  AccessCount
-----
ST1_id    10
```

## Table\_LevelConstraintsV[X]

**Category:** Integrity

**Database:** DBC

| View Column    | Data Type   | Format | Comment   |
|----------------|---|--------|---|
| DataBaseName   | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | Returns the name of the database containing a table with a table-level check.   |
| TableName      | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | Returns the name of a table having table-level check constraints.               |
| ConstraintName | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC             | X(128) | Returns the name of the table-level check. This field is NULL if it is unnamed. |

| View Column     | Data Type   | Format              | Comment  |
|-----------------|---|---------------------|--|
| ConstraintText  | VARCHAR(16000)<br>UNICODE<br>CASESPECIFIC               | X(16000)            | Returns the unresolved text for the table-level check condition.   |
| CreatorName     | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)              | Table or database creator. For DBC. AllRights, the grantor of explicit rights, otherwise the user who executed the CREATE statement. |
| CreateTimeStamp | TIMESTAMP(0)  | YYYY-MM-DDBHH:MI:SS | Returns the date and time that the object in the row was created.  |
| VTCheckType     | CHAR(1)<br>LATIN UPPERCASE                              | X(1)                | Returns the ValidTime dimension information for the CHECK condition.   |
| TTCheckType     | CHAR(1)<br>LATIN UPPERCASE                              | X(1)                | Returns the TransactionTime dimension for the CHECK constraint on a table with TransactionTime.                                      |

## Usage Notes

For information about the possible values for the VTCheckType column, see "VTCheckType Column."

## Corresponding Tables

The X view references these additional tables:

- DBC.Owners
- DBC.RoleGrants
- DBC.Roles

## Possible Values for TTCheckType

| Value | Description                   |
|-------|-------------------------------|
| NULL  | No transaction-time dimension |
| C     | CURRENT TRANSACTIONTIME       |

## Tables2V[X]

**Category:** Schema

**Database:** DBC

| View Column | Data Type                                     | Format  | Comment   |
|-------------|---|---------|---|
| TVMName     | VARCHAR(128)<br>UNICODE UPPERCASE<br>NOT NULL | X(128)  | Returns the name of the table.  |
| TVMId       | BYTE(6) NOT NULL                              | X(12)   | Returns the internal ID of the table.   |
| Databaseld  | BYTE(4) NOT NULL                              | X(8)    | Returns the ID of the database with the indicated count of unresolved references.   |
| ParentCount | SMALLINT NOT NULL                             | ---,--9 | Returns the number of parent tables for the table specified in the TVM row.   |
| ChildCount  | SMALLINT NOT NULL                             | ---,--9 | Returns the count of the referencing tables for the table. For stored procedures and external stored procedures, this field stores the number of result sets. |

## Usage Notes

### Corresponding Tables

The X view references these additional tables:

- DBC.AccessRights
- DBC.Owners
- DBC.RoleGrants
- DBC.Roles

## Tables3V[X]

**Category:** Schema

**Database:** DBC

| View Column  | Data Type  | Format | Comment   |
|--------------|--|--------|---|
| DatabaseName | VARCHAR(128) UNICODE<br>UPPERCASE NOT NULL           | X(128) | Returns the database or user name of the object for which this log entry was made.  |
| TableName    | VARCHAR(128) UNICODE<br>UPPERCASE NOT NULL           | X(128) | Returns the name of a table, view, stored procedure, trigger, macro, user-defined types, user-defined methods, or user-defined function on which a privilege was granted. |
| FieldName    | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128) | Returns the name of the field.  |
| TableId      | BYTE(6) NOT NULL                                     | X(12)  | Returns ID of the table   |

| View Column | Data Type         | Format  | Comment                      |
|-------------|-------------------|---------|------------------------------|
| FieldId     | SMALLINT NOT NULL | ---,--9 | Returns the Id of the field. |

## Usage Notes

### Corresponding Tables

The X view references these additional tables:

- DBC.AccessRights
- DBC.Owners
- DBC.RoleGrants
- DBC.Roles

## TableSizeV[X]

**Category:** Accounting

**Database:** DBC

| View Column  | Data Type   | Format                         | Comment  |
|--------------|---|--------------------------------|--|
| Vproc        | INTEGER NOT NULL  | --,---,---,--9                 | Identifies the virtual processor for which an event was logged.  |
| DataBaseName | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                         | Returns the name of a database.  |
| AccountName  | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                         | Expanded account in effect when a request was submitted. SYSTEMACCOUNTNO tracks console utility activity such as table rebuild, Diskcopy, or Scandisk. |
| TableName    | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                         | Returns the name of a table, join index, or hash index.  |
| CurrentPerm  | BIGINT  | --,---,---,---,<br>---,---,--9 | Returns the permanent space (in bytes) currently used by the database or table.  |
| PeakPerm     | BIGINT  | --,---,---,---,<br>---,---,--9 | Returns the maximum amount of permanent space per AMP that has been used by the database since the last time the DBC. ClearPeakDisk macro was run.     |

## Usage Notes

When a database or table is created, the allocated disk space is divided evenly among all AMPs. The TableSize view returns one row of usage information for each AMP in the system (or for all AMPs if the SUM aggregate is used).

### Corresponding Tables

The X view references these additional tables:

- DBC.AccessRights
- DBC.Owners
- DBC.RoleGrants
- DBC.Roles

### CurrentPerm

CurrentPerm returns the permanent space (in bytes) currently used by the database or table. The CurrentPerm column value includes all AMPs unless a specific AMP is requested.

### PeakPerm

Use the DBC.ClearPeakDisk macro to reset the PeakPerm column.

## Examples: Using TableSizeV

### Example: Using TableSizeV to Contrast Total Current Disk Space with Peak Usage

The following SELECT statement is used to contrast the total disk space currently being used by the Employee table with its peak usage figure:

```
SELECT SUM(PeakPerm), SUM(CurrentPerm)
       FROM DBC.TableSizeV WHERE TableName='Employee';
```

Result:

| Sum(PeakPerm) | Sum(CurrentPerm) |
|---------------|------------------|
| -----         | -----            |
| 260,608       | 260,608          |

### Example: Using TableSizeV to Display the Size of the Tables in a Map

To show the size of the tables defined in a given map:

```
SELECT tabv.databasename, tabv.tablename, vproc, sum(currentperm)
FROM DBC.TableSizeV tabsz, DBC.TablesV tabv
WHERE tabsz.databasename = tabv.databasename
AND tabsz.tablename = tabv.tablename AND mapname = 'MyMap1'
GROUP BY 1,2,3;
```

### Example: Using TableSizeV to Display Database Permanent Space Usage in Each AMP

To show the permanent space usage by map for each database in each AMP:

```
SELECT tabv.mapname, tabv.databasename, vproc, sum(currentperm)
FROM DBC.TableSizeV tabsz, DBC.TablesV tabv
WHERE tabsz.databasename = tabv.databasename
AND tabsz.tablename = tabv.tablename
GROUP BY 1,2,3;
```

### Example: Using TableSizeV to Display Database Permanent Space Usage by Map in the System

To show the permanent space usage by map for each database in the system:

```
SELECT tabv.mapname, tabv.databasename, sum(currentperm)
FROM DBC.TableSizeV tabsz, DBC.TablesV tabv
WHERE tabsz.databasename = tabv.databasename
AND tabsz.tablename = tabv.tablename
GROUP BY 1,2;
```

## TableStatsV

**Category:** Optimizer Statistics

**Database:** DBC

| View Column  | Data Type   | Format | Comment   |
|--------------|---|--------|---|
| DatabaseName | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | The DatabaseName column is the name of the Database in which the table resides. |
| TableName    | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | The TableName column is the name of the containing table.                       |

| View Column           | Data Type                                     | Format                   | Comment   |
|-----------------------|---|--------------------------|---|
| ColumnName            | VARCHAR(10000)<br>UNICODE NOT<br>CASESPECIFIC | X(10000)                 | The ColumnName column identifies a column or columns.   |
| StatsName             | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC   | X(128)                   | The StatsName column contains the name associated with the statistic                                |
| StatsSource           | CHAR(1)<br>LATIN UPPERCASE                    | X(1)                     | The StatsSource column records the method this statistic is acquired.                               |
| ValidStats            | CHAR(1)<br>LATIN UPPERCASE                    | X(1)                     | The ValidStats column indicates whether the statistics are valid or not.                            |
| DBSVersion            | VARCHAR(32)<br>LATIN UPPERCASE                | X(32)                    | The DBSVersion column records Database version statistics collected on.                             |
| IndexNumber           | SMALLINT                                      | ---,--9                  | The IndexNumber column is the Index number of the index on which statistics are collected.          |
| SampleSignature       | VARCHAR(256)<br>LATIN NOT<br>CASESPECIFIC     | X(256)                   | The SampleSignature column is the Sample options encoded as a 10 character signature.               |
| SampleSizePct         | DECIMAL(5,2)                                  | ----.99                  | The SampleSizePct column is the Sample size percent used when collecting statistics.                |
| ThresholdSignature    | VARCHAR(512)<br>LATIN NOT<br>CASESPECIFIC     | X(512)                   | The ThresholdSignature column is the Threshold options encoded as a 17 character signature.         |
| MaxIntervals          | SMALLINT                                      | ---,--9                  | The MaxIntervals column is the User-specified maximum number of intervals.                          |
| MaxValueLength        | INTEGER                                       | --,---,---,--9           | The MaxValueLength column is the User-specified maximum value length.                               |
| RowCount              | FLOAT   | ----,---,---,---,<br>--9 | The RowCount column is the row count of the table, view or query on which statistics are collected. |
| UniqueValueCount      | FLOAT   | ----,---,---,---,<br>--9 | The UniqueValueCount column is the Number of unique values of the ExpressionList.                   |
| PNullUniqueValueCount | FLOAT   | ----,---,---,---,<br>--9 | The PNullUniqueValueCount column is the Number of unique values from rows with partial nulls.       |
| NullCount             | FLOAT   | ----,---,---,---,<br>--9 | The NullCount column is the Number of nulls of the ExpressionList.                                  |

| View Column          | Data Type    | Format                   | Comment  |
|----------------------|--------------|--------------------------|--|
| AllNullCount         | FLOAT        | ----,---,---,---,<br>--9 | The AllNullCount column is the Number of all nulls of the ExpressionList.  |
| HighModeFreq         | FLOAT        | ----,---,---,---,<br>--9 | The HighModeFreq column is the highest frequency of values of the ExpressionList.  |
| PNullHighModeFreq    | FLOAT        | ----,---,---,---,<br>--9 | The PNullHighModeFreq column is the highest frequency of values having partial nulls of the ExpressionList.              |
| StatsSkipCount       | INTEGER      | --,---,---,--9           | The StatsSkipCount column indicates how many times the statistics collection on the ExpressionList has been skipped.     |
| CreateTimeStamp      | TIMESTAMP(0) | YYYY-MM-DDBHH:MI:SS      | The CreateTimeStamp column is the statistics creation time stamp.  |
| LastCollectTimeStamp | TIMESTAMP(0) | YYYY-MM-DDBHH:MI:SS      | The LastCollectTimeStamp column is the Last statistics collection time stamp.  |
| LastAlterTimeStamp   | TIMESTAMP(0) | YYYY-MM-DDBHH:MI:SS      | The LastAlterTimeStamp column is the last user updated time stamp.   |
| BLCCompRatio         | INTEGER      | --,---,---,--9           | The estimated space savings percentage for primary subtable of manually compressed BLC tables.                           |
| AvgRowSize           | FLOAT        | ----,---,---,---,<br>--9 | The AvgRowSize column is the average row size of an object on which statistics are collected.                            |
| BLCCompFactor        | FLOAT        | ----,---,---,---,<br>--9 | The estimated block compression factor for the entire table. This is used for calculating the Customer Data Space (CDS). |

## Usage Notes

### ColumnName

- If more than one column or expression is specified, each column or expression is separated by a comma.
- The maximum number of columns is 64.
- If expressions are in the list, the maximum number of columns can be reduced past the limit of 64, depending on the combined total size of the text in the expressions.

- If the combined total size of the expression text causes the maximum column limit to be less than the actual number of columns in the list, an error occurs.

**SampleSizePct**

If sampling is not used, the SampleSizePct column is set to 0 or 100.

**StatsSource**

The StatsSource column records the method by which this statistic is acquired. For information about the possible values for the StatsSource column, see "StatsSource Column."

**To Get Information Not Contained in This View**

This view does not contain:

- Statistics on tables protected by row-level security. To get these statistics, create views on DBC.StatsTbl. You can view details about this table in Teradata SQL Assistant or Teradata Studio Express.
- Information about column attributes. To get this information, join the DBC.StatsTbl table to the DBC.TVFields table. You can view details about this table in Teradata SQL Assistant or Teradata Studio Express.

**Example: Using TableStatsV**

This example assumes the following statistics have been collected:

```
STATISTICS
  INDEX (o_orderkey)
  ,INDEX (o_custkey, o_orderstatus)
  ON Orders;
```

This query can be used to retrieve the statistics:

```
SELECT * FROM dbc.TableStatsV
  WHERE databasename = 'sales'
  AND tablename = 'orders';
```

**Related Topics**

| For more information about statistics collected on ...  | See ...                            |
|---|------------------------------------|
| Non-indexed columns and single-column indexes           | <a href="#">ColumnStatsV.</a>      |
| Indexes for which two or more columns have been defined | <a href="#">IndexStatsV[X].</a>    |
| Multiple columns  | <a href="#">MultiColumnStatsV.</a> |

| For more information about statistics collected on ... | See ...                          |
|--|----------------------------------|
| Tables   | <a href="#">StatsV.</a>          |
| materialized temporary tables                          | <a href="#">TempTableStatsV.</a> |
| single expressions                                     | <a href="#">ExpStatsV.</a>       |
| multiple expressions                                   | <a href="#">MultiExpStatsV.</a>  |

## TablesV[X]

**Category:** Schema

**Database:** DBC

| View Column    | Data Type   | Format   | Comment   |
|----------------|---|----------|---|
| DataBaseName   | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)   | Returns the name of the database in which the table resides.  |
| TableName      | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)   | Returns the name of a table, join index, or hash index.   |
| Version        | SMALLINT<br>NOT NULL                                    | zzzz(9)  | Returns the version count, which is incremented each time the table is altered with a data definition statement.                                |
| TableKind      | CHAR(1) LATIN<br>NOT<br>CASESPECIFIC                    | X(1)     | Type of table.  |
| ProtectionType | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL                  | X(1)     | Returns F (Fallback) or N (None) to indicate whether the tables in the database are protected by the Fallback option.                           |
| JournalFlag    | CHAR(2) LATIN<br>NOT<br>CASESPECIFIC<br>NOT NULL        | X(2)     | Journaling in effect for the table, or the journal default for tables. Settings: first character BEFORE, second character AFTER.                |
| CreatorName    | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)   | Returns the Teradata user who issued the CREATE TABLE statement.  |
| RequestText    | VARCHAR(12500)<br>UNICODE NOT<br>CASESPECIFIC           | X(12500) | Returns the text of the most recent data definition statement that was used to change the table, view, join index, trigger, macro, user-defined |

| View Column          | Data Type   | Format              | Comment  |
|----------------------|---|---------------------|--|
|                      |   |                     | types, user-defined methods, or user defined function.   |
| CommentString        | VARCHAR(255)<br>UNICODE NOT<br>CASESPECIFIC             | X(255)              | Returns user-supplied text or commentary on the column, database, table, view, macro, user-defined function, user-defined types, user-defined methods, stored procedure, role, profile, or user. |
| ParentCount          | SMALLINT<br>NOT NULL                                    | ---,--9             | Returns the number of parent tables for the table specified in the TVM row.  |
| ChildCount           | SMALLINT<br>NOT NULL                                    | ---,--9             | Returns the count of the referencing tables for the table. For stored procedures and external stored procedures, this field stores the number of result sets.                                    |
| NamedTblCheckCount   | SMALLINT<br>NOT NULL                                    | ---,--9             | Returns the count of named table-level check constraint for the table.   |
| UnnamedTblCheckExist | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL                  | X(1)                | Returns an indicator for whether the table has an unnamed table-level check constraint, as follows: Y = Yes, N = No.   |
| PrimaryKeyIndexId    | SMALLINT  | ---,--9             | TablesVX.PrimaryKeyIndexId identifies the Primary Key index Id for the table. It is NULL if table does not have a Primary Key.   |
| RepStatus            | CHAR(1) LATIN<br>UPPERCASE                              | X(1)                | These are the possible values: C = connected, D = defined, F = failed, S = suspended, T = terminated, I = Initiated, NULL.   |
| CreateTimeStamp      | TIMESTAMP(0)  | YYYY-MM-DDBHH:MI:SS | Returns the date and time that the object in the row was created.  |
| LastAlterName        | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)              | Returns the name of the user who last updated the dictionary row.  |
| LastAlterTimeStamp   | TIMESTAMP(0)  | YYYY-MM-DDBHH:MI:SS | Returns the time the dictionary row was last updated.  |
| RequestTxtOverflow   | CHAR(1) LATIN<br>UPPERCASE                              | X(1)                | identifies that the Request Text for the object has overflowed.  |

| View Column           | Data Type                              | Format                             | Comment  |
|-----------------------|--|------------------------------------|--|
| AccessCount           | BIGINT                                 | --,---,---,<br>---,---,---,<br>--9 | Returns the access count for the corresponding database object.  |
| LastAccessTimeStamp   | TIMESTAMP(0)                           | YYYY-MM-DDBHH:MI:SS                | Returns the time that the corresponding object was last accessed.  |
| UtilVersion           | SMALLINT                               | -----9                             | Returns the utility version count.   |
| QueueFlag             | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL | X(1)                               | Stores the queuing option for the table from the values: Y = Yes (queuing option is set), N = No (queuing option is not set).  |
| CommitOpt             | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL | X(1)                               | The TablesVX.CommitOpt specifies the ON COMMIT option for a temporary table. Value P stands for ON COMMIT PRESERVE ROWS. Value D stands for ON COMMIT DELETE ROWS. Value N indicates the object is not a temporary table.                                |
| TransLog              | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL | X(1)                               | Specifies whether transaction journals are generated.  |
| CheckOpt              | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL | X(1)                               | Indicates whether or not the table allows duplicate rows.  |
| TemporalProperty      | CHAR(1) LATIN<br>UPPERCASE             | X(1)                               | DBC.TablesVX.TemporalProperty indicates whether the table/view/join index is a nontemporal, ValidTime, TransactionTime, or bi-temporal table   |
| ResolvedCurrent_Date  | DATE                                   | YY/MM/DD                           | Returns the last resolved value of CURRENT_DATE.   |
| ResolvedCurrent_Stamp | TIMESTAMP(6)<br>WITH TIME ZONE         | YYYY-MM-DDBHH:MI:SS.S(6)Z          | Returns the last resolved value of CURRENT_TIMESTAMP.  |
| SystemDefinedJI       | CHAR(1) LATIN<br>UPPERCASE             | X(1)                               | DBC.TablesVX.SystemDefinedJI records whether a table level object kind is a system-defined join index or any other object such as table,view, macro,user defined join index, stored procedure etc. Y if the entry is corresponding SysJI or N otherwise. |
| VTQualifier           | CHAR(1) LATIN<br>UPPERCASE             | X(1)                               | DBC.TablesVX.VTQualifier records the temporal qualifier for validtime  |

| View Column        | Data Type               | Format         | Comment   |
|--------------------|-------------------------|----------------|---|
|                    |                         |                | dimension. NULL indicates that there is no ValidTime Dimension, C indicates Current ValidTime, S indicates Sequenced ValidTime, N indicates Non-Sequenced ValidTime   |
| TTQualifier        | CHAR(1) LATIN UPPERCASE | X(1)           | DBC.TablesVX.TTQualifier records the temporal qualifier for TransactionTime dimension. NULL is no TransactionTime Dimension, C is Current TransactionTime, S is Sequenced TransactionTime, N is Non-Sequenced TransactionTime             |
| PIColumnCount      | SMALLINT NOT NULL       | -(5)9          | TablesVX.PIColumnCount is the number of columns in the primary index or primary AMP index (otherwise, 0).   |
| PartitioningLevels | SMALLINT NOT NULL       | -(5)9          | DBC.TablesVX.PartitioningLevels indicates the number of partitioning levels (a value between 1 and 62, inclusive) or no partitioning (a value of 0).  |
| LoadProperty       | CHAR(1) LATIN UPPERCASE | X(1)           | LoadProperty is NULL for non-LDI objects. For LDI tables, "I" indicates concurrent Load Isolation for only INSERTs, "A" for ALL DMLs, "D" for temporarily disabled and "S" for system determined for a join index defined on a LDI table. |
| CurrentLoadId      | INTEGER                 | --,---,---,--9 | CurrentLoadId indicates the CurrLoadID of the LDI table/join index. It is NULL for non-LDI objects.   |
| LoadIdLayout       | CHAR(1) LATIN UPPERCASE | X(1)           | LoadIdLayout indicates the RowLoadID layout type of the table. It is set to NULL for non-LDI objects, "R" when RowLoadID is recorded in the row along with the other columns and "C" when RowLoadID is recorded in a column partition.    |
| DelayedJI          | CHAR(1) LATIN UPPERCASE | X(1)           | DelayedJI is set to NULL for non-join index entries and set to a not-null value when a JI is declared for delayed JI maintenance.   |
| BlockSize          | INTEGER                 | Z,ZZZ,ZZZ,ZZ9  | DataBlockSize is the integer value in bytes and is rounded upward to the nearest multiple of the sector size in bytes. Null is specified if not specified or if it does not apply to the object.  |

| View Column               | Data Type                                      | Format | Comment  |
|---------------------------|--|--------|--|
| FreeSpacePercent          | BYTEINT  | Z9%    | FreeSpacePercent is the byte integer value in the range of 0 to 75, inclusive. NULL if not specified or if it does not apply to the object.  |
| MergeBlockRatio           | BYTEINT  | ZZ9%   | MergeBlockRatio is the byte integer value in the range of 0 to 100, inclusive. NULL if not specified or if it does not apply to the object.  |
| Checksum                  | CHAR(1) LATIN UPPERCASE                        | X(1)   | Checksum is "N" for OFF or "Y" for ON to indicate checksum.  |
| BlockCompression          | VARCHAR(8) UNICODE NOT CASESPECIFIC            | X(8)   | BlockCompression corresponds to the string token specified in the BLOCKCOMPRESSION clause. Valid values are "DEFAULT", "MANUAL", "AUTOTEMP", "NEVER", "ALWAYS" or NULL if not specified or if it does not apply to the object. |
| BlockCompressionAlgorithm | VARCHAR(7) UNICODE NOT CASESPECIFIC            | X(7)   | BlockCompressionAlgorithm corresponds to the string token specified in the BLOCKCOMPRESSIONALGORITHM clause. Valid values are "DEFAULT", "ZLIB", "ELZS_H" or NULL if not specified or if it does not apply to the object.      |
| BlockCompressionLevel     | VARCHAR(7) UNICODE NOT CASESPECIFIC            | X(7)   | BlockCompressionLevel corresponds to the string token specified in the BLOCKCOMPRESSIONLEVEL clause. Valid values are "DEFAULT", "1-9" or NULL if not specified or if it does not apply to the object.                         |
| TableHeaderFormat         | CHAR(1) LATIN UPPERCASE                        | X(1)   | TableHeaderFormat is a single character to describe the format of table header; "0" for "thin" format of up to 64KB, and "1" for "fat" format up to 1MB in-memory and up to 16 64KB rows on disk.                              |
| RowSizeFormat             | VARCHAR(1) LATIN NOT CASESPECIFIC              | X(1)   | RowSizeFormat indicates the row size; "0" for a format of up to 64KB, and "1" for a format of up to 1MB.   |
| MapName                   | VARCHAR(128) UNICODE NOT CASESPECIFIC NOT NULL | X(128) | Returns name of map where the table resides.   |

| View Column               | Data Type                                   | Format             | Comment  |
|---------------------------|---|--------------------|--|
| ColocationName            | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128)             | ColocationName is set to NULL for contiguous Map and set to a non-null value for sparse maps.  |
| TVMFlavor                 | CHAR(1) LATIN<br>UPPERCASE                  | X(1)               | TVMFlavor S is used in conjunction with TableKind T to identify an unique table kind; TVMFlavor N is used in conjunction with TableKind 2 to identify function mapping type; NULL if it is not in use. |
| FastAlterTable            | CHAR(1) LATIN<br>UPPERCASE                  | X(1)               | Returns T for FCA table and F and NULL for Non-FCA table   |
| IncrementalRestoreEnabled | INTEGER                                     | --,---,---,<br>--9 | Indicates if Incremental Restore has been enabled for the table.   |
| AuthName                  | VARCHAR(128)<br>UNICODE<br>UPPERCASE        | X(128)             | Returns the authorization name specified in the external security clause of the user defined function or external stored procedure.  |

## Usage Notes

The source of many of the columns in TablesV[X] is the DBC.TVM table. DBC.TVM contains one row for each table, view, stored procedure, join index, macro, UDT, UDM, or UDF.

For information about the possible values for JournalFlag or TableKind columns, see "JournalFlag Column" and "TableKind Column."

Some of the column values show information related to a Teradata temporal table or an ANSI temporal table. For more information about these tables, see *Teradata Vantage™ ANSI Temporal Table Support*, B035-1186 and *Teradata Vantage™ Temporal Table Support*, B035-1182.

## Corresponding Tables

The X view references these additional tables:

- DBC.AccessRights
- DBC.Owners
- DBC.RoleGrants
- DBC.Roles

## BlockSize

BlockSize is specified in the DATABLOCKSIZE clause of the CREATE TABLE or ALTER TABLE statements. The value is in bytes and is rounded upward to the nearest multiple of the sector size in bytes. That rounded value is not rounded up to a 4KB page boundary. NULL is inserted when a DATABLOCKSIZE clause is not specified for the table, or does not apply to the object (for example, a view).

**ColocationName**

ColocationName is the colocation name for an object with a sparse map. It is null for an object with a contiguous map and for a non-mapped object.

Objects with the same sparse map reside on the same AMPs if those objects have the same colocation names.

**CurrentLoadId**

A positive, even integer that indicates the last committed load value for a load isolated table or join index. Its initial value is 0 and it is incremented by 1 when a load is committed. This column is NULL for objects that are not load isolated.

**FreeSpacePercent**

FreeSpacePercent is specified in the FREESPACE clause of the CREATE TABLE or ALTER TABLE statements. The value is in the range 0 to 75, inclusive.

NULL is inserted when a FREESPACE clause is not specified for the table, or does not apply to the object (for example, a view).

**MapName**

MapName is the name of the map in which the corresponding table resides. MapNames start with TD\_, such as TD\_Map1, for contiguous maps. The MapName for sparse maps created by CREATE MAP is not allowed to start with TD\_.

**MergeBlockRatio**

MergeBlockRatio is specified in the MERGEBLOCKRATIO clause of the CREATE TABLE or ALTER TABLE statements. The value is in the range 0 to 100, inclusive. NULL is inserted when a MERGEBLOCKRATIO clause is not specified for the table, or does not apply to the object (for example, a view).

**RequestText**

The RequestText data reflects the definitions specified by the user. This may not always match the data returned by the SHOW TABLE statement, which reflects the reconstructed definitions as they exist in the Data Dictionary.

For example, when obsolete syntax that is still supported is converted internally to current syntax, RequestText returns the submitted (obsolete) syntax, while SHOW TABLE returns the converted (current) syntax.

If the table is renamed, the RequestText still contains the old table name. The new table name is not put in the RequestText.

### Possible Values for BlockCompression

BlockCompression is specified in the BLOCKCOMPRESSION clause of the CREATE TABLE, CREATE JOIN INDEX, CREATE HASH INDEX, or ALTER TABLE statements. Valid values are:

| Value | Description  |
|-------|--|
| 0     | DEFAULT  |
| 1     | MANUAL   |
| 2     | AUTOTEMP   |
| 3     | NEVER  |
| 4     | ALWAYS   |
| NULL  | BlockCompression is NULL if not specified or if it does not apply to the object. |

### Possible Values for BlockCompressionAlgorithn

BlockCompressionAlgorithm is specified in the BLOCKCOMPRESSIONALGORITHM clause of the CREATE TABLE, CREATE JOIN INDEX, CREATE HASH INDEX, or ALTER TABLE statements. Valid values are:

| Value | Description   |
|-------|---|
| 0     | DEFAULT   |
| 1     | ZLIB  |
| 2     | ELZS_H  |
| NULL  | BlockCompressionAlgorithm is NULL if not specified or if it does not apply to the object. |

### Possible Values for BlockCompressionLevel

BlockCompressionLevel is specified in the BLOCKCOMPRESSIONLEVEL clause of the CREATE TABLE, CREATE JOIN INDEX, CREATE HASH INDEX, or ALTER TABLE statements. The integer values stored are: 0 (DEFAULT), or 1 through 9.

A value of 0 (DEFAULT) indicates that the attribute of the table is the current system default setting in DBS Control. If the DBS Control setting changes this column is not updated.

An integer value of 4 is stored corresponding to the string token 'ALWAYS' specified in the BLOCKCOMPRESSION clause of the CREATE TABLE, CREATE JOIN INDEX, CREATE HASH INDEX, or ALTER TABLE statements.

BlockCompressionLevel is NULL when a BLOCKCOMPRESSIONLEVEL clause is not specified for the table, join index, hash index, or does not apply to the object (for example, a view).

### Possible Values for CheckSum

CheckSum is specified in the CHECKSUM clause of the CREATE TABLE, CREATE JOIN INDEX, CREATE HASH INDEX, or ALTER TABLE statements. The values stored are:

| Value | Description   |
|-------|---|
| N     | OFF   |
| Y     | ON  |
| NULL  | NULL is inserted when a CHECKSUM clause is not specified for the table, join index, or hash index, or does not apply to the object (for example, a view). |

Note, the old keywords indicating Checksums enabled (VERSION, LOW, MEDIUM, HIGH, and ALL) are mapped internally to the keyword ON, and the old keyword indicating Checksums disabled (NONE) is mapped internally to the keyword OFF. The old external keywords are no longer visible.

### Possible Values for LoadIdLayout

| Value | Description  |
|-------|--|
| NULL  | Tables that are not load isolated.                                 |
| R     | Load isolated table with row partitioning or with no partitioning. |
| C     | Load isolated table with column partitioning.                      |

### Possible Values for LoadProperty

| Value | Description   |
|-------|---|
| NULL  | Tables or join indexes that are not load isolated.  |
| I     | Concurrent load isolation for INSERTs only on a load isolated table.                              |
| A     | Concurrent load isolation for all Data Manipulation Language statements on a load isolated table. |
| D     | Temporarily disabled the concurrent load isolation on a load isolated table.                      |
| S     | A join index defined on a load isolated table.  |

### Possible Values for PrimaryKeyId

| Value   | Description   |
|---|---|
| 1   | Primary key is implemented using a unique primary index for the table.  |
| Multiple of 4 (that is, a number between 4 and 128) | Primary key is implemented using a unique secondary index for the table. The number is the index ID for the unique secondary index. |
| NULL  | Primary key is not defined for the table.   |

### Possible Values for PIColumnCount

PIColumnCount is set to the number of columns in the primary index or primary AMP index. 0 indicates there is neither a primary index nor a primary AMP index.

#### Note:

For a NoPI table, TableKind is 'O' if the table is not partitioned and is 'T' if the table is partitioned. For a join index, TableKind is 'I'.

To determine if a table or join index (with a nonzero PIColumnCount) has a primary index or a primary AMP index, the DBC.IndicesV[X] system views can be queried. If PIColumnCount is zero, the table or join index has neither a primary index nor a primary AMP index (that is, it is NoPI) and there are no rows in the DBC.IndicesV[X] views with an IndexNumber of 1. Note that a hash index always has a primary index.

### Possible Values for PartitioningLevels

| Value                        | Description  |
|------------------------------|--|
| Between 1 and 62 (inclusive) | The number of partitioning levels for the table or join index. |
| 0                            | The table or join index is not partitioned.                    |

### Possible Values for RowSizeFormat

RowSizeFormat describes the size format used to the store base table, join index, or hash index rows for this object at the time the last DDL statement was completed for this object. Valid values are:

| Value | Description   |
|-------|---|
| 0     | A row size format of up to 64KB   |
| 1     | A row size format of up to 1MB  |
| NULL  | NULL is inserted when object the does not have any base table rows (for example, a view). |

### Possible Values for ResolvedCurrent\_TimeStamp and ResolvedCurrent\_Date

- This is the last reconciled timestamp or date if the object is a join index or a table that is defined using:
  - CURRENT\_TIMESTAMP
  - CURRENT\_DATE or DATE

Either the partition, JI definition, or temporal table has a system-defined join index.
- NULL in all other cases.

### Possible Values for SystemDefinedJI

| Value | Description  |
|-------|--|
| Y     | The entry is corresponding to a system-defined join index. |
| NULL  | Any other objects in the system.                           |

### Possible Values for TableHeaderFormat

TableHeaderFormat describes the format used to process and store the table header for this object at the time the last DDL statement was completed for this object. Note that this field is not updated when the table header format changes during non-DDL operations (for example, Down AMP regions). Valid values are:

| Value | Description  |
|-------|--|
| 0     | Indicates a thin format of up to 64KB in-memory and as a single row on disk.             |
| 1     | Indicates a fat format of up to 1MB in-memory and up to 16 64KB rows on disk.            |
| NULL  | NULL is inserted when the object the does not have a table header (for example, a view). |

### Possible Values for TemporalProperty

| Value | Description   |
|-------|---|
| S     | A system-versioned system-time table.   |
| U     | A bitemporal (system-versioned system-time and valid-time) table.   |
| W     | A nontemporal table that contains a system-time derived period column but that is not system versioned.         |
| X     | A valid-time temporal table that contains a system-time derived period column but that is not system versioned. |

### Possible Values for TVMFlavor

TVMFlavor is used in conjunction with TableKind of O or T to identify the unique table kind. Possible values are:

| Value | Description                  |
|-------|------------------------------|
| D     | Direct Access foreign table. |
| M     | Materialized foreign table.  |
| S     | Time Series table.           |

## Possible Values for Version and UtilVersion

The UtilVersion column may be zero or null for tables created prior to V2R6.0. For tables created in V2R6.0 or later, both Version and UtilVersion are initially one.

The following table shows when the UtilVersion column is modified, and how it is modified:

| UtilVersion Current Value | Data Definition Change   | New Value or UtilVersion |
|---------------------------|--|--------------------------|
| Any value                 | Significant change (by an ALTER TABLE statement) to keep an archive from being restored or copied. | = Version                |
| NULL or zero              | Not significant  | Version -1               |
| Any other value           | Not significant  | No change                |

The Version column is increased by one for any data definition change other than enabling or disabling journaling for the table. Version in the above table is the version after it has been incremented.

The following table shows how the UtilVersion column is checked when selected partitions are restored or copied:

| UtilVersion Value | Requirements   |
|-------------------|--|
| NULL or zero      | The Version of the archive must match the Version of the table   |
| Any other value   | If the archive UtilVersion is zero, the Version of the Archive must match the UtilVersion of the table; otherwise, the UtilVersion column of the archive must equal the UtilVersion of the table |

### Note:

To restore or copy selected partitions, the table must already exist.

## Example: Using TablesV

The following SELECT statement displays information about tables, views, and macros in the Personnel database:

```
SELECT TableName,CreatorName,TableKind,ProtectionType FROM DBC.TablesV WHERE
DatabaseName = 'Personnel' ;
```

The query returns the following result:

| TableName | CreatorName | TableKind | ProtectionType |
|-----------|-------------|-----------|----------------|
| -----     | -----       | -----     | -----          |
| NewEmp    | GREENE      | M         | F              |

|              |        |   |   |
|--------------|--------|---|---|
| EmployeeInfo | GREENE | V | F |
| Employee     | DBC    | T | F |
| Department   | DBC    | T | F |
| Project      | JONES  | T | F |
| Charges      | JONES  | T | F |

## Related Topics

For more information about primary indexes and partitioning levels, see *Teradata Vantage™ SQL Data Definition Language Syntax and Examples*, B035-1144.

## TableTextV[X]

**Category:** Schema

**Database:** DBC

| View Column  | Data Type  | Format   | Comment  |
|--------------|--|----------|--|
| DataBaseName | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL   | X(128)   | Returns the name of the database in which the table resides.   |
| TableName    | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL   | X(128)   | Returns the name of a table, join index, or hash index.  |
| TableKind    | CHAR(1) LATIN<br>NOT CASESPECIFIC                      | X(1)     | Type of table.   |
| RequestText  | VARCHAR(32000) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(32000) | Returns the text of the most recent data definition statement used to change the table, view, or macro.                |
| LineNo       | SMALLINT NOT NULL                                      | ---,--9  | Returns the number that corresponds to the running sequence number of the multiple rows of text stored in DBC.TextTbl. |

## Usage Notes

For information about the possible values for the TableKind column, see "TableKind Column."

## Corresponding Tables

The X view references these additional tables:

- DBC.AccessRights
- DBC.DBBase

- DBC.Owners
- DBC.RoleGrants
- DBC.Roles
- DBC.TextTbl
- DBC.TVM

## RequestText

The request text data reflects the definitions specified by the user for the object.

| IF the request text ...   | THEN the request text is saved in ...   |
|---------------------------|---|
| is up to 12500 characters | TVM   |
| exceeds 12500 characters  | DBC.TextTbl<br>This view contains the complete text of the object definition.<br><br><b>Note:</b><br>Users do not need to use this view if there is no overflow in TVM. |

The request text for an object could be stored in one or more rows depending on the size of the text. The LineNo column provides the sequence of multiple rows stored in the TextTbl for the object.

To put the text in a proper sequence, you must select the text using ORDER BY clause on LineNo from this view.

## Example: Using TableTextV

The following SELECT statement displays information about tables, views, and macros in the Personnel database, assuming that text for the database object is more than 12500 characters.

```
SELECT TableName, TableKind, LineNo, RequestText( char(50)) FROM
DBC.TableTextV WHERE DatabaseName = 'Personnel' ORDER BY TableName, LineNo;
```

## TableToSparseMapSizingV[X]

**Category:** TDMaps

**Database:** TDMaps

| View Column  | Data Type   | Format | Comment                  |
|--------------|---|--------|--------------------------|
| DataBaseName | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | The name of the database |

| View Column         | Data Type   | Format  | Comment  |
|---------------------|---|---------|--|
| TableName           | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)  | The name of the table, join index, or hash index   |
| PeakDiskSpace       | BIGINT  | -(19)9  | An estimate of the amount of peak disk space all the primary rows take determined by looking at peak space and subtracting out an estimate for the table header.           |
| CurrentDiskSpace    | BIGINT  | -(19)9  | An estimate of the amount of current disk space all the primary rows take determined by looking at the current space and subtracting out an estimate for the table header. |
| DataSize            | BIGINT  | -(19)9  | An estimate of the amount of space all the primary rows take considering the statistics and average row size.  |
| RecommendedMap      | VARCHAR(26) LATIN<br>NOT CASESPECIFIC                   | X(26)   | The map the object should be in  |
| CurrentMap          | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)  | The map the object is currently in   |
| MapOk               | VARCHAR(1)<br>UNICODE<br>NOT CASESPECIFIC               | X(1)    | Y if data estimated for the table is appropriate for the map the table is currently in, N otherwise  |
| CurrentMapKind      | CHAR(1) LATIN<br>NOT CASESPECIFIC<br>NOT NULL           | X(1)    | C if the current map is contiguous, S if it is sparse  |
| CurrentMapNumOfAMPs | SMALLINT NOT NULL                                       | ---,--9 | The number of AMPs in the map the object is in   |
| ProtectionType      | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL                  | X(1)    | F if the table is fallback protected, N if not   |

## Usage Notes

The TableToSparseMapSizingV[X] views identify which map a table should be in. For example, TableToSparseMapSizingV[X] might show that tables in a contiguous map are recommended to move to a particular sparse map. Note that the size of a table's table header row depends on the description of the table. That information is not available to the view so an estimate is used. Because this is only an estimate, the recommended tables to move should be reviewed.

## Referenced Columns

Many of the Data Dictionary view columns have referenced table columns. That is, the value in the view column corresponds to a value in the selected column referenced in the table. It would be meaningful to join the view and the referenced table based on the selected column and the referenced column.

Referenced columns for this view are:

| View Column  | Referenced Column  |
|--------------|--------------------|
| DatabaseName | Dbase.DatabaseName |
| TableName    | TVM.TVMName        |

## CurrentMap and RecommendedMap

CurrentMap is the map the object is currently in. RecommendedMap is the map to which the object should be moved.

## TblSrvInfoV[X]

**Category:** Operations

**Database:** DBC

| View Column       | Data Type  | Format   | Comment  |
|-------------------|--|----------|--|
| ServerName        | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128)   | Returns the name of the FOREIGN SERVER.  |
| SrvDataBaseName   | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128)   | Returns the name of the database or user in which the subject foreign server resides.          |
| TblOpName         | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128)   | Returns the name of the table operator.  |
| TbpOpDataBaseName | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128)   | Returns the name of the database database or user in which the subject table operator resides. |
| NameInfo          | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128)   | Returns the value portion specified in the custom-clause.                                      |
| ValueInfo         | VARCHAR(31000)<br>UNICODE<br>NOT CASESPECIFIC        | X(31000) | Returns the value portion specified in the custom-clause.                                      |
| ValueType         | CHAR(1) LATIN<br>NOT CASESPECIFIC                    | X(1)     | Return the identifier to identify whether the value specified in                               |

| View Column       | Data Type                              | Format | Comment  |
|-------------------|--|--------|--|
|                   |  |        | custom-clause is either SELECT/Value/System.                         |
| TableOperatorType | VARCHAR(7) UNICODE<br>NOT CASESPECIFIC | X(7)   | Returns the single character field representing table operator type. |

## Usage Notes

This Teradata QueryGrid connector view returns the name value pairs defined for a foreign server for the associated table operator, such as IMPORT or EXPORT table operators.

### NameInfo and ValueInfo

NameInfo returns the name portion specified in the USING custom clause. ValueInfo returns the value portion specified in the USING custom clause.

### Possible Values for TableOperatorType

- IMPORT
- EXPORT
- UNKNOWN

### Possible Values for ValueType

ValueType identifies what type of value is specified in the USING custom clause.

| Value | Description   |
|-------|---|
| NULL  | ValueInfo column contains a data value.                           |
| S     | ValueInfo column contains query text mentioned in NVP query.      |
| R     | ValueInfo column contains a resolved query text mentioned in NVP. |
| V     | ValueInfo column contains a system variable.                      |

## TblSrvV[X]

**Category:** Operations

**Database:** DBC

| View Column | Data Type  | Format | Comment                                 |
|-------------|--|--------|---|
| ServerName  | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128) | Returns the name of the FOREIGN SERVER. |

| View Column       | Data Type  | Format | Comment  |
|-------------------|--|--------|--|
| SrvDataBaseName   | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128) | Returns the name of the database or user in which the subject foreign server resides.          |
| TblOpName         | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128) | Returns the name of the table operator.  |
| TblOpDBName       | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128) | Returns the name of the database database or user in which the subject table operator resides. |
| TableOperatorType | VARCHAR(7) UNICODE<br>NOT CASESPECIFIC               | X(7)   | Returns the single character field representing table operator type.                           |

## Usage Notes

This Teradata QueryGrid connector view returns information about the foreign servers and their associated table operators.

### Possible Values for TableOperatorType

- IMPORT
- EXPORT
- UNKNOWN

## TempTableStatsV

**Category:** Optimizer Statistics

**Database:** DBC

| View Column  | Data Type   | Format   | Comment   |
|--------------|---|----------|---|
| DatabaseName | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)   | The DatabaseName column is the name of the Database in which the table resides. |
| TableName    | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)   | The TableName column is the name of the containing table.                       |
| ColumnName   | VARCHAR(10000)<br>UNICODE NOT<br>CASESPECIFIC           | X(10000) | The ColumnName column identifies a column or columns.                           |

| View Column           | Data Type                                   | Format                   | Comment   |
|-----------------------|---|--------------------------|---|
| StatsName             | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC | X(128)                   | The StatsName column contains the name associated with the statistic                                  |
| StatsSource           | CHAR(1)<br>LATIN UPPERCASE                  | X(1)                     | The StatsSource column records the method this statistic is acquired.                                 |
| ValidStats            | CHAR(1)<br>LATIN UPPERCASE                  | X(1)                     | The ValidStats column indicates whether the statistics are valid or not.                              |
| DBSVersion            | VARCHAR(32)<br>LATIN UPPERCASE              | X(32)                    | Returns the version of the database that contains the objects on which the statistics were collected. |
| IndexNumber           | SMALLINT                                    | ---,--9                  | The IndexNumber column is the Index number of the index on which statistics are collected.            |
| SampleSignature       | VARCHAR(256)<br>LATIN NOT<br>CASESPECIFIC   | X(256)                   | The SampleSignature column is the Sample options encoded as a 10 character signature.                 |
| SampleSizePct         | DECIMAL(5,2)                                | ----.99                  | The SampleSizePct column is the Sample size percent used when collecting statistics.                  |
| ThresholdSignature    | VARCHAR(512)<br>LATIN NOT<br>CASESPECIFIC   | X(512)                   | The ThresholdSignature column is the Threshold options encoded as a 17 character signature.           |
| MaxIntervals          | SMALLINT                                    | ---,--9                  | The MaxIntervals column is the User-specified maximum number of intervals.                            |
| MaxValueLength        | INTEGER                                     | --,---,---,--9           | The MaxValueLength column is the User-specified maximum value length.                                 |
| RowCount              | FLOAT                                       | ----,---,---,---,<br>--9 | The RowCount column is the row count of the table, view or query on which statistics are collected.   |
| UniqueValueCount      | FLOAT                                       | ----,---,---,---,<br>--9 | The UniqueValueCount column is the Number of unique values of the ExpressionList.                     |
| PNullUniqueValueCount | FLOAT                                       | ----,---,---,---,<br>--9 | The PNullUniqueValueCount column is the Number of unique values from rows with partial nulls.         |
| NullCount             | FLOAT                                       | ----,---,---,---,<br>--9 | The NullCount column is the Number of nulls of the ExpressionList.                                    |
| AllNullCount          | FLOAT                                       | ----,---,---,---,<br>--9 | The AllNullCount column is the Number of all nulls of the ExpressionList.                             |

| View Column          | Data Type    | Format                  | Comment  |
|----------------------|--------------|-------------------------|--|
| HighModeFreq         | FLOAT        | ---,---,---,---,<br>--9 | The HighModeFreq column is the highest frequency of values of the ExpressionList.  |
| PNullHighModeFreq    | FLOAT        | ---,---,---,---,<br>--9 | The PNullHighModeFreq column is the highest frequency of values having partial nulls of the ExpressionList.              |
| CreateTimeStamp      | TIMESTAMP(0) | YYYY-MM-DDBHH:MI:SS     | The CreateTimeStamp column is the statistics creation time stamp.  |
| LastCollectTimeStamp | TIMESTAMP(0) | YYYY-MM-DDBHH:MI:SS     | The LastCollectTimeStamp column is the Last statistics collection time stamp.  |
| LastAlterTimeStamp   | TIMESTAMP(0) | YYYY-MM-DDBHH:MI:SS     | The LastAlterTimeStamp column is the last user updated time stamp.   |
| BLCCompRatio         | INTEGER      | --,---,---,--9          | The estimated space savings percentage for primary subtable of manually compressed BLC tables.                           |
| AvgRowSize           | FLOAT        | ---,---,---,---,<br>--9 | The AvgRowSize column is the average row size of an object on which statistics are collected.                            |
| BLCCompFactor        | FLOAT        | ---,---,---,---,<br>--9 | The estimated block compression factor for the entire table. This is used for calculating the Customer Data Space (CDS). |

## Usage Notes

### Referenced Columns

Many of the Data Dictionary view columns have referenced table columns. That is, the value in the view column corresponds to a value in the selected column referenced in the table. It would be meaningful to join the view and the referenced table based on the selected column and the referenced column.

Referenced columns for this view are:

| View Column | Referenced Column       |
|-------------|-------------------------|
| IndexNumber | DBC.Indexes.IndexNumber |

## ColumnName

- If more than one column or expression is specified, each column or expression is separated by a comma.
- The maximum number of columns is 64.
- If expressions are in the list, the maximum number of columns can be reduced past the limit of 64, depending on the combined total size of the text in the expressions.
- If the combined total size of the expression text causes the maximum column limit to be less than the actual number of columns in the list, an error occurs.

## SampleSizePct

If sampling is not used, the SampleSizePct column is set to 0 or 100.

## StatsSource

The StatsSource column records the method by which this statistic is acquired. For information about the possible values for the StatsSource column, see "StatsSource Column."

## Related Topics

| For information about statistics collected on ...       | See ...  |
|---|--|
| non-indexed columns and single-column indexes           | <a href="#">ColumnStatsV.</a>                          |
| indexes for which two or more columns have been defined | <a href="#">IndexStatsV[X].</a>                        |
| multiple columns  | <a href="#">MultiColumnStatsV.</a>                     |
| tables  | <a href="#">StatsV</a> or <a href="#">TableStatsV.</a> |
| single expressions                                      | <a href="#">ExpStatsV.</a>                             |
| multiple expressions                                    | <a href="#">MultiExpStatsV.</a>                        |

## TriggersV[X]

**Category:** Schema

**Database:** DBC

| View Column              | Data Type   | Format | Comment   |
|--------------------------|---|--------|---|
| DataBaseName             | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | Returns the name of the database where the trigger resides. |
| SubjectTableDataBaseName | VARCHAR(128)<br>UNICODE NOT                             | X(128) | Returns the name of the database or user in which           |

| View Column     | Data Type   | Format                      | Comment  |
|-----------------|---|-----------------------------|--|
|                 | CASESPECIFIC<br>NOT NULL                                |                             | the subject table of the trigger resides. You can use this to check for potential orphan triggers.   |
| TableName       | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                      | Returns the name of the table the trigger is defined against.  |
| TriggerName     | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                      | Returns the name of the trigger.   |
| EnabledFlag     | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL                  | X(1)                        | Returns a code that indicates whether the trigger is enabled or disabled.  |
| ActionTime      | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL                  | X(1)                        | Returns a code to signify when, in relation to the SQL statement, the trigger is fired. These are the values: B = Before, A = After.         |
| Event           | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL                  | X(1)                        | Returns a description of the type of action, using the following descriptions: Logon, Logoff, Logon failed.                                  |
| Kind            | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL                  | X(1)                        | Returns a code that indicates whether the trigger is evaluated for: S = the statement (once), R = each row changed by the triggering action. |
| OrderNumber     | SMALLINT  | ---,--9                     | Optionally specifies when triggers bearing the same action time and event executes.  |
| TriggerComment  | VARCHAR(255)<br>UNICODE<br>NOT CASESPECIFIC             | X(255)                      | Returns the optional comment for the trigger.  |
| RequestText     | VARCHAR(12500)<br>UNICODE<br>NOT CASESPECIFIC           | X(12500)                    | Returns the actual request text that was used to create the trigger.   |
| CreatorName     | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                      | Returns the Teradata User who issued the CREATE/REPLACE TRIGGER statement.   |
| CreateTimeStamp | TIMESTAMP(0)  | YYYY-MM-<br>DDBHH:MI:<br>SS | Returns the date and time that the object in the row was created.  |

| View Column         | Data Type   | Format              | Comment   |
|---------------------|---|---------------------|---|
| LastAlterName       | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)              | Returns the name of the user who last updated the dictionary row.   |
| LastAlterTimeStamp  | TIMESTAMP(0)  | YYYY-MM-DDBHH:MI:SS | Returns the time the dictionary row was last updated.   |
| AccessCount         | INTEGER   | --,---,---,--9      | Returns the access count for the corresponding database object.   |
| LastAccessTimeStamp | TIMESTAMP(0)  | YYYY-MM-DDBHH:MI:SS | Returns the time that the corresponding object was last accessed.   |
| CreateTxtOverflow   | CHAR(1)<br>LATIN UPPERCASE                              | X(1)                | Identifies that the CreateText for the object overflowed DBC.TVM when set to 'C', and the complete text is stored in DBC.TextTbl. |
| VTEventType         | CHAR(1) LATIN<br>NOT CASESPECIFIC                       | X(1)                | COLUMN DBC.TriggersVX. VTEventType records the VT qualifier on the subject table  |
| TTEventType         | CHAR(1) LATIN<br>NOT CASESPECIFIC                       | X(1)                | COLUMN DBC.TriggersVX. TTEventType records the TT qualifier on the subject table  |

## Usage Notes

Some of the column values show information related to a Teradata temporal table or an ANSI temporal table. For more information about these tables, see *Teradata Vantage™ ANSI Temporal Table Support* , B035-1186 and *Teradata Vantage™ Temporal Table Support* , B035-1182 .

## Corresponding Tables

The X view references these additional tables:

- DBC.AccessRights
- DBC.DBase
- DBC.Owners
- DBC.RoleGrants
- DBC.Roles
- DBC.TriggersTbl
- DBC.TVM

**Possible Values for Event**

| Value | Description |
|-------|-------------|
| U     | Update      |
| I     | Insert      |
| D     | Delete      |

**Possible Values for VTEventType**

| Value | Description  |
|-------|--|
| A     | ANSIQUALIFIER<br><b>Note:</b><br>ANSI temporal tables require that the session temporal qualifier for systems using Teradata temporal tables be explicitly set to ANSIQUALIFIER. |
| C     | CURRENT VALIDTIME  |
| N     | NONSEQUENCED VALIDTIME   |
| NULL  | No valid-time dimension  |
| S     | SEQUENCED VALIDTIME  |

**Possible Values for TTEventType**

| Value | Description   |
|-------|---|
| A     | ANSIQUALIFIER.<br><b>Note:</b><br>ANSI temporal tables require that the session temporal qualifier for systems using Teradata temporal tables be explicitly set to ANSIQUALIFIER. |
| C     | CURRENT TRANSACTIONTIME   |
| N     | NONSEQUENCED TRANSACTIONTIME  |
| NULL  | No transaction-time dimension   |
| S     | SEQUENCED TRANSACTIONTIME   |
| T     | Nontemporal to indicate that transaction-time is ignored even though the table supports a transaction-time dimension  |

**Example: Using TriggersV**

The following SELECT returns this information:

- Name of the database in which the triggering table is defined for those cases in which a trigger is defined in a different database than the triggering table
- Names of the triggering tables
- Names of the database in which the trigger is defined
- Trigger names

This query identifies those triggers for which the trigger must be dropped if the database containing the triggering table is deleted.

```
SELECT SubjectTableDatabaseName, TableName, DatabaseName, TriggerName
FROM TriggersV
WHERE DatabaseName <> SubjectTableDatabaseName
ORDER BY 1, 2, 3, 4;
```

## Related Topics

For more information about triggers, see *Teradata Vantage™ SQL Data Definition Language Syntax and Examples*, B035-1144.

## UDTInfoV

**Category:** Schema

**Database:** DBC

| View Column  | Data Type                                     | Format | Comment  |
|--------------|---|--------|--|
| TypeId       | BYTE(6) NOT NULL                              | X(12)  | Returns the identifier for the UDT   |
| DatabaseId   | BYTE(4) NOT NULL                              | X(8)   | Returns the identifier of the database in which the UDT is defined                           |
| TypeName     | VARCHAR(128)<br>UNICODE UPPERCASE<br>NOT NULL | X(128) | Returns the type designator. The name corresponds to the column TVMName in the DBC.TVM table |
| TypeKind     | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL        | X(1)   | Returns the classification of the UDT  |
| INSTANTIABLE | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL        | X(1)   | An indication of whether the UDT is instantiable   |
| FINAL        | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL        | X(1)   | An indication of whether the UDT is final  |
| Encryption   | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL        | X(1)   | An indication of whether encryption is supported for this UDT                                |

| View Column           | Data Type                                | Format  | Comment  |
|-----------------------|--|---------|--|
| Compression           | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL   | X(1)    | An indication of whether compression is supported for this UDT   |
| OperatorAll           | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL   | X(1)    | An indication of whether operators specific to predefined data types are supported for a distinct types base data type   |
| DefaultTransformGroup | VARCHAR(128)<br>UNICODE UPPERCASE        | X(128)  | Returns the name of the Transform group that is selected automatically for importing or exporting and is used in the absence of a specific choice made by the user |
| OrderingForm          | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL   | X(1)    | An indication of the level of comparison and sorting for the UDT.  |
| OrderingCategory      | CHAR(1)<br>LATIN UPPERCASE               | X(1)    | An indication of the manner of comparison.   |
| OrderingRoutineId     | BYTE(6)                                  | X(12)   | Specifies the identifier of the user-defined ordering routine  |
| CastCount             | BYTEINT NOT NULL                         | -(3)9   | Returns the number of user-defined casts involving this UDT  |
| ExtFileReference      | VARCHAR(1000)<br>UNICODE<br>CASESPECIFIC | X(1000) | An encoded string that contains the external name or path of each file component that together provide core DBS integration methods for the UDT                    |
| DefaultNull           | CHAR(1)<br>LATIN UPPERCASE               | X(1)    | Indicates whether the DEFAULT NULL clause was specified when the ARRAY data type was created   |
| ArrayNumDimensions    | BYTEINT                                  | -(3)9   | Contains a numeric value that indicates the number of dimensions for an ARRAY type   |
| ArrayScope            | VARCHAR(3200)<br>LATIN UPPERCASE         | X(3200) | Contains a string of the format '[n:m]...' which describes the lower and upper bounds values for each dimension of the ARRAY type                                  |

## Usage Notes

### OrderingRoutineId

You can specify an Embedded Services ordering routine by setting the OrderingRoutineId column to '000000000000'XB. Embedded Services refers to a procedure in which one of the database integration methods is called directly.

**Possible Values for OrderingCategory**

| Value | Description |
|-------|-------------|
| R     | Relative    |
| M     | Map         |

**Possible Values for OrderingForm**

| Value | Description |
|-------|-------------|
| N     | None        |
| F     | Full        |
| E     | Equals only |

**Possible Values for TypeKind**

| Value | Description |
|-------|-------------|
| S     | Structure   |
| D     | Distinct    |
| I     | Internal    |

**Example: Using UDTInfoV**

The following SELECT statement retrieves information about the predefined Array type, ArrayVec, which is provided by the DIP scripts:

```
SELECT TypeId, TypeKind, OrderingForm, OrderingCategory, CastCount
FROM DBC.UDTInfoV
WHERE TypeName = 'ArrayVec';
```

The query returns the following result:

| TypeId       | TypeKind | OrderingForm | OrderingCategory | CastCount |
|--------------|----------|--------------|------------------|-----------|
| -----        | -----    | -----        | -----            | -----     |
| 000090060000 | I        | F            | M                | 0         |

**UDTTransformV**

**Category:** Schema

**Database: DBC**

| View Column            | Data Type   | Format | Comment  |
|------------------------|---|--------|--|
| TypeName               | VARCHAR(128)<br>UNICODE<br>UPPERCASE<br>NOT NULL        | X(128) | Returns the type designator. The name corresponds to the column TVMName in the DBC.TVM table.    |
| GroupName              | VARCHAR(128)<br>UNICODE<br>UPPERCASE<br>NOT NULL        | X(128) | Returns the group name associated with this collection of UDT Transform routines.                |
| ToSQLRoutineId         | BYTE(6)   | X(12)  | Returns the identifier of the routine that performs the "To SQL" Transform for importing data.   |
| ToSQLDatabaseName      | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | Returns the name of the database for the "To SQL" Transform for importing data.                  |
| ToSQLRoutineName       | VARCHAR(128)<br>UNICODE<br>UPPERCASE<br>NOT NULL        | X(128) | Returns the name of the routine for the "To SQL" Transform for importing data.                   |
| ToSQLSpecificName      | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | Returns the name of the function for the "To SQL" Transform for importing data.                  |
| ToSQLParameterDataType | VARCHAR(256)<br>LATIN UPPERCASE                         | X(256) | Returns the list of the input parameters for the "To SQL" Transform for importing data.          |
| FromSQLRoutineId       | BYTE(6)   | X(12)  | Returns the identifier of the routine that performs the "From SQL" Transform for exporting data. |
| FromSQLDatabaseName    | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | Returns the name of the database for the "From SQL" Transform for exporting data.                |
| FromSQLRoutineName     | VARCHAR(128)<br>UNICODE<br>UPPERCASE<br>NOT NULL        | X(128) | Returns the name of the routine for the "From SQL" Transform for exporting data.                 |
| FromSQLSpecificName    | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | Returns the name of the function for the "From SQL" Transform for exporting data.                |

| View Column           | Data Type                  | Format | Comment   |
|-----------------------|----------------------------|--------|---|
| FromSQLReturnDataType | CHAR(2)<br>LATIN UPPERCASE | X(2)   | Returns the list of the input parameters for the "From SQL" Transform for exporting data. |

## Usage Notes

This view provides public access to information about data transforms.

## UpdateUseCountV[X]

**Category:** Accounting

**Database:** DBC

| View Column  | Data Type   | Format                         | Comment  |
|--------------|---|--------------------------------|--|
| DatabaseName | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                         | Returns the Database Name of the object for which access count and/or UDI counts are recorded.   |
| ObjectName   | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                         | Returns the Object Name of the object for which access count and/or UDI counts are recorded.   |
| FieldName    | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                         | Returns Field Name of a table if usage type is a DML type; StatsId if the usage is for statistics (the optimizer usage of statistics). |
| UsageType    | CHAR(3) LATIN<br>UPPERCASE NOT NULL                     | X(3)                           | Returns the usage type of the object. It can be either DML or STA.   |
| UpdateCount  | BIGINT  | --,---,---,---,<br>---,---,--9 | Returns the number of updates since the last reset by the user.  |

## Usage Notes

### Possible Values for UsageType

| Name | Description                   |
|------|-------------------------------|
| DML  | Data Manipulation Language    |
| STA  | Optimizer usage of statistics |

## Example: Using UpdateUseCountV

The following SELECT statement displays the number of updates occurring on a particular object:

```
SELECT FieldName, UpdateCount FROM DBC.UpdateUseCountV WHERE DatabaseName =
'Personnel' AND ObjectName = 'Employee';
```

The query returns the following result:

```
FieldName  UpdateCount
-----
id         1
name      0
```

## User\_Default\_JournalsV[X]

**Category:** Schema

**Database:** DBC

| View Column | Data Type  | Format | Comment   |
|-------------|--|--------|---|
| UserName    | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128) | Returns the name of a user space for which a default journal table has been defined.                        |
| Journal_DB  | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128) | Returns the name of the database or user space in which the default journal table for DatabaseName resides. |
| JournalName | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128) | Returns the name of the journal table defined as the default for UserName.                                  |

## Example: Using User\_Default\_JournalsVX

The following SELECT statement selects information on each user database to which the requesting user has access, and for which a default journal table is defined.

```
SELECT * FROM DBC.User_Default_JournalsVX;
```

Result:

```
UserName    Journal_DB    JournalName
-----
-----
```

|      |      |          |
|------|------|----------|
| Usr1 | Usr1 | Usr1Jrn1 |
| Usr2 | Usr2 | Usr2Jrn1 |
| Usr3 | Usr3 | Usr3Jrn1 |

## UserGrantedRightsV

**Category:** Security

**Database:** DBC

| View Column     | Data Type   | Format              | Comment  |
|-----------------|---|---------------------|--|
| DatabaseName    | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)              | Returns the name of a database or one of the special system keywords: ALL / DEFAULT / PUBLIC.  |
| TableName       | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)              | Returns the name of a table, view, stored procedure, trigger, macro, user-defined types, user-defined methods, or user.                                |
| ColumnName      | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)              | The UserGrantedRightsV.ColumnName field identifies the table column or the view column which a right has been granted.                                 |
| Grantee         | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)              | Returns the name of a user who was granted a privilege; ALL can be specified. Returns the name of a user or role who was granted a role.               |
| AccessRight     | CHAR(2) LATIN<br>UPPERCASE<br>NOT NULL                  | X(2)                | Returns a code that identifies a privilege granted on the object.  |
| GrantAuthority  | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL                  | X(1)                | Returns the WITH GRANT OPTION attribute of the access right held by the user.  |
| AllnessFlag     | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL                  | X(1)                | Returns Y (yes) or N (no) to indicate whether or not the privilege was granted to all subordinate users, or to all users who are owned by the grantee. |
| CreatorName     | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC             | X(128)              | Table or database creator. For DBC. AllRights, the grantor of explicit rights, otherwise the user who executed the CREATE statement.                   |
| CreateTimeStamp | TIMESTAMP(0)  | YYYY-MM-DDBHH:MI:SS | Returns the date and time that the object in the row was created.  |

## Usage Notes

For more information about the possible values for the AccessRight column, see "AccessRight Column."

### Possible Values for AllnessFlag

| Value | Description |
|-------|-------------|
| Y     | Yes         |
| N     | No          |

## Example: Using UserGrantedRightsV

The following SELECT statement displays all privileges that the current user has granted to other users.

```
SELECT DatabaseName,TableName,Grantee,AccessRight
       FROM DBC.UserGrantedRightsV;
```

Result:

| DatabaseName | TableName | Grantee | AccessRight |
|--------------|-----------|---------|-------------|
| -----        | -----     | -----   | -----       |
| Personnel    | Employee  | Greene  | R           |
| Personnel    | Employee  | Greene  | U           |
| Personnel    | Employee  | Greene  | I           |

## UserRightsV

**Category:** Security

**Database:** DBC

| View Column  | Data Type   | Format | Comment  |
|--------------|---|--------|--|
| DatabaseName | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | Returns the name of a database or one of the special system keywords: ALL / DEFAULT / PUBLIC.                              |
| TableName    | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | The UserRightsV.TableName field identifies a table, view, stored procedure or macro on which the user has an access right. |
| ColumnName   | VARCHAR(128)<br>UNICODE NOT                             | X(128) | Returns the name of the column or parameter.   |

| View Column     | Data Type                                   | Format                 | Comment  |
|-----------------|---|------------------------|--|
|                 | CASESPECIFIC<br>NOT NULL                    |                        |  |
| AccessRight     | CHAR(2) LATIN<br>UPPERCASE<br>NOT NULL      | X(2)                   | Returns a code that identifies a privilege granted on the object.  |
| GrantAuthority  | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL      | X(1)                   | Returns the WITH GRANT OPTION attribute of the access right held by the user.  |
| GrantorName     | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC | X(128)                 | Returns the name of the grantor who granted the privilege.   |
| CreatorName     | VARCHAR(128)<br>UNICODE<br>NOT CASESPECIFIC | X(128)                 | Table or database creator. For DBC. AllRights, the grantor of explicit rights, otherwise the user who executed the CREATE statement. |
| CreateTimeStamp | TIMESTAMP(0)                                | YYYY-MM-DD<br>HH:MI:SS | Returns the date and time that the object in the row was created.  |

## Usage Notes

To display the privileges that the user has been granted on database D, the SELECT statement must specify:

```
WHERE DatabaseName = 'D' AND TableName = 'All' ;
```

If privileges have been granted on the database, a row is returned for each privilege.

The UserRightsV view does not return information about implicit privileges of a user due to ownership of objects, nor does it return information about privileges inherited by a user from PUBLIC or roles.

To obtain information about privileges inherited from roles, use the UserRoleRightsV view. For implicit privileges, use the ChildrenV[X] and TablesV[X] views to determine all the objects owned by a user.

For information about the possible values for the AccessRight column, see "AccessRight Column."

## Example: Using UserRightsV

The following SELECT statement displays information about all tables in the Personnel database on which privileges were granted to the requesting user.

```
SELECT * FROM DBC.UserRightsV
      WHERE DatabaseName='Personnel'
      AND TableName = 'All' ;
```

Result:

| DatabaseName | TableName | AccessRight | GrantorName |
|--------------|-----------|-------------|-------------|
| -----        | -----     | -----       | -----       |
| Personnel    | Employee  | R           | DBC         |
| Personnel    | Employee  | U           | DBC         |
| Personnel    | Employee  | I           | DBC         |

## UserRoleRightsV

**Category:** Security

**Database:** DBC

| View Column     | Data Type  | Format              | Comment   |
|-----------------|--|---------------------|---|
| RoleName        | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128)              | Returns the name of a role.                                       |
| DatabaseName    | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128)              | Returns the name of a database.                                   |
| TableName       | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128)              | Returns the name of a table, join index, or hash index.           |
| ColumnName      | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128)              | Returns the name of the column or parameter.                      |
| AccessRight     | CHAR(2) LATIN<br>UPPERCASE NOT NULL                  | X(2)                | Returns a code that identifies a privilege granted on the object. |
| GrantorName     | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC             | X(128)              | Returns the name of the grantor who granted the privilege.        |
| CreateTimeStamp | TIMESTAMP(0)   | YYYY-MM-DDBHH:MI:SS | Returns the date and time that the object in the row was created. |

## Usage Notes

The UserRoleRightsV view is similar to the AllRoleRightsV view. However, UserRoleRightsV has additional WHERE conditions that restrict the result set to rows belonging to the current role of the user and all roles nested within that.

For information about the possible values for the AccessRight column, see "AccessRight Column."

## Example: Using UserRoleRightsV

The following SELECT statement returns all privileges granted to the current role of the user and the roles nested within the current role.

```
SELECT CAST(RoleName as CHAR(16)) as RoleName,
       CAST(DatabaseName as CHAR(15)) as Databases,
       CAST(TableName as CHAR(15)) as TVMs,
       CAST(ColumnName as CHAR(10)) as Columns,
       CAST(AccessRight as CHAR(5)) as AccRights,
       CAST(GrantorName as CHAR(15)) as Grantor
FROM DBC.UserRoleRightsV
ORDER BY 1,2,3,5;
```

Result:

| RoleName     | Databases     | TVMs          | AccRights | Grantor    |
|--------------|---------------|---------------|-----------|------------|
| -----        | -----         | -----         | -----     | -----      |
| roles017_r1f | roles017_3_db | roles017_3_v2 | R         | roles017_3 |
| roles017_r2d | roles017_3_db | roles017_3_v2 | R         | roles017_3 |

## UsersV

**Category:** Database

**Database:** DBC

| View Column | Data Type   | Format | Comment   |
|-------------|---|--------|---|
| UserName    | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | The UsersV.Username field identifies a user.                    |
| CreatorName | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128) | Returns the Teradata user who issued the CREATE USER statement. |

| View Column         | Data Type  | Format                         | Comment  |
|---------------------|--|--------------------------------|--|
| PasswordLastModDate | DATE   | YY/MM/DD                       | Returns the date that the user password was last modified.   |
| PasswordLastModTime | TIME(0) WITH TIME ZONE                               | HH:MI:SSZ                      | Returns the time that the user password was last modified.   |
| OwnerName           | VARCHAR(128)<br>UNICODE NOT CASESPECIFIC<br>NOT NULL | X(128)                         | User owner name.   |
| PermSpace           | BIGINT NOT NULL                                      | --,---,---,---,<br>---,---,--9 | The UsersV.PermSpace field specifies the total permanent space in bytes allocated to the user.   |
| SpoolSpace          | BIGINT   | -(19)9                         | Returns an integer indicating the maximum spool space allowed for the database. SpoolSpace is 0 if DatabaseName is PUBLIC.   |
| TempSpace           | BIGINT   | -(19)9                         | Returns the maximum temporary space allocated for a database, profile, or user in bytes.   |
| ProtectionType      | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL               | X(1)                           | Returns F (Fallback) or N (None) to indicate whether the tables in the database are protected by the Fallback option.  |
| JournalFlag         | CHAR(2) LATIN<br>UPPERCASE<br>NOT NULL               | X(2)                           | Journaling in effect for the table, or the journal default for tables. Settings: first character BEFORE, second character AFTER.   |
| StartupString       | VARCHAR(255)<br>UNICODE NOT CASESPECIFIC             | X(255)                         | Returns the startup string (macro or SQL statement) specified for the user.  |
| DefaultAccount      | VARCHAR(128)<br>UNICODE NOT CASESPECIFIC             | X(128)                         | Returns the name of the default account, if any, for the user.   |
| DefaultDataBase     | VARCHAR(128)<br>UNICODE NOT CASESPECIFIC             | X(128)                         | Returns the name of the current default database for the user.   |
| CommentString       | VARCHAR(255)<br>UNICODE NOT CASESPECIFIC             | X(255)                         | The UsersV.CommentString field contains any user-supplied text for the user.   |
| DefaultCollation    | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL               | X(1)                           | Returns the default collation for the user as defined in a CREATE/MODIFY USER statement; if not defined, the DefaultCollation is the collation of the logon client system. |

| View Column        | Data Type   | Format                         | Comment   |
|--------------------|---|--------------------------------|---|
| PasswordChgDate    | DATE  | YY/MM/DD                       | Returns the Julian date on which the current password was assigned to the user. This value is 0 for a new user.   |
| LockedDate         | DATE  | yy/mm/dd                       | Returns the Julian date on which the Dbase row was locked to logons due to excessive erroneous passwords. A null or 0 value indicates the row was never locked. |
| LockedTime         | INTEGER   | 99:99                          | Minutes after midnight when the row was locked to logons due to excessive erroneous passwords. NULL or zero (row never locked)                                  |
| LockedCount        | BYTEINT   | -(3)9                          | Number of successive unsuccessful attempts to logon to the user with an erroneous password. NULL or zero (no attempts).   |
| TimeZoneHour       | BYTEINT   | -(3)9                          | Returns the TimeZone Hour offset from UTC. Valid range is -12 to +14.   |
| TimeZoneMinute     | BYTEINT   | -(3)9                          | Returns the TimeZone Minute offset from UTC. Valid range is -59 to +59.   |
| DefaultDateForm    | CHAR(1) LATIN<br>UPPERCASE                              | X(1)                           | Returns either an INTEGERDATE or ANSIDATE that is set for a USER with the CREATE or MODIFY statement.   |
| CreateTimeStamp    | TIMESTAMP(0)  | YYYY-MM-<br>DDBHH:MI:<br>SS    | Returns the date and time that the object in the row was created.   |
| LastAlterName      | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC<br>NOT NULL | X(128)                         | Returns the name of the user who last updated the dictionary row.   |
| LastAlterTimeStamp | TIMESTAMP(0)  | YYYY-MM-<br>DDBHH:MI:<br>SS    | Returns the time the dictionary row was last updated.   |
| DefaultCharType    | SMALLINT  | ---,--9                        | Returns the type of the user default character type.  |
| RoleName           | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC             | X(128)                         | Returns the name of a role.   |
| ProfileName        | VARCHAR(128)<br>UNICODE NOT<br>CASESPECIFIC             | X(128)                         | Returns the name of the profile.  |
| AccessCount        | BIGINT  | --,---,---,---,<br>---,---,--9 | Returns the access count for the corresponding database object.   |

| View Column          | Data Type                                | Format              | Comment   |
|----------------------|--|---------------------|---|
| LastAccessTimeStamp  | TIMESTAMP(0)                             | YYYY-MM-DDBHH:MI:SS | Returns the time that the corresponding object was last accessed.   |
| ExportDefinitionName | VARCHAR(30)<br>LATIN NOT CASESPECIFIC    | X(30)               | The ExportDefinitionName is the name of the ExportWidthRuleSet for the user. A NULL value means to use the system defaults.   |
| ExportWidthRuleSet   | BYTE(20)                                 | X(40)               | The ExportWidthRuleSet is the export width rule set associated with the ExportDefinitionName and is updated if this association changes. A NULL value means to use the system defaults. |
| DBA                  | CHAR(1) LATIN<br>UPPERCASE<br>NOT NULL   | X(1)                | The UsersV.DBA column specifies if the user is DBA user or not.   |
| TimeZoneString       | VARCHAR(128)<br>UNICODE NOT CASESPECIFIC | X(128)              | Returns the time zone string set for a particular User.   |
| DefaultMapName       | VARCHAR(128)<br>UNICODE NOT CASESPECIFIC | X(128)              | Returns the default map name for the database or user. NULL indicates the database or user does not have a default map.   |
| MapOverride          | VARCHAR(1) LATIN<br>NOT CASESPECIFIC     | X(1)                | MapOverride indicates no override or override on error for the default map. "N" indicates no override. "E" indicates OVERRIDE ON ERROR.   |

## Usage Notes

During a session, the account name may be changed. For the current account name of the session, see [SessionInfoV\[X\]](#).

For information about the possible values for the JournalFlag column, see "JournalFlag Column."

## UserName

For this column, the SYSTEMUSERID is a system user name that tracks console utility activity such as table rebuild or Scandisk.

## Possible Values for DefaultCharType

| Value | Description |
|-------|-------------|
| 1     | Latin       |
| 2     | Unicode     |

| Value | Description |
|-------|-------------|
| 3     | KanjiSJIS   |
| 4     | Graphic     |
| 5     | Kanji1      |

### Possible Values for DefaultCollation

| Value | Description   |
|-------|---------------|
| A     | ASCII         |
| E     | EBCDIC        |
| H     | Host          |
| M     | Multinational |
| C     | CharSet_Coll  |
| J     | JIS_Coll      |

### Possible Value for DefaultDataBase

If the default database is not defined in a CREATE/MODIFY USER statement, the DefaultDataBase column is the user space and NULL is returned.

#### Note:

At logon, the DefaultDataBase column may be superseded by a default account in a profile, if the user has a profile that specifies a default account.

## Example: Using UsersV

The following SELECT statement displays information about all users owned or created by the current user, Jones.

```
SELECT UserName,CreatorName,PermSpace,SpoolSpace
FROM DBC.UsersV;
```

Result:

| UserName | CreatorName | PermSpace | SpoolSpace |
|----------|-------------|-----------|------------|
| -----    | -----       | -----     | -----      |
| Jones    | sysadmin    | 1,000,000 | 1,000,000  |
| Peterson | Jones       | 100,000   | 1,000,000  |

|        |       |         |           |
|--------|-------|---------|-----------|
| Moffit | Jones | 100,000 | 1,000,000 |
| Chin   | Jones | 100,000 | 1,000,000 |
| Greene | Jones | 100,000 | 1,000,000 |

## Related Topics

| For more information about ...                                       | See ...   |
|--|---|
| controlling access, space, and ownership                             | <i>Teradata Vantage™ - Database Design</i> , B035-1094.   |
| interpreting the ExportWidthRuleSet column                           | <a href="#">ExportWidthV</a> .  |
| using the DBSControl utility to make export width definition changes | <i>Teradata Vantage™ - Advanced SQL Engine International Character Set Support</i> , B035-1094. |

## UsrAsgdSecConstraintsV[X]

**Category:** Integrity

**Database:** DBC

| View Column    | Data Type  | Format | Comment  |
|----------------|--|--------|--|
| UserName       | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128) | Returns the name of the user that has been assigned the security constraint.   |
| ConstraintName | VARCHAR(128) UNICODE<br>UPPERCASE NOT NULL           | X(128) | Returns the name of the security constraint object. This field is NULL if it is unnamed.   |
| ValueName      | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128) | Returns the name of the security constraint value that was assigned to the user or profile. This value is typically user-defined.  |
| IsDefault      | CHAR(1) LATIN<br>UPPERCASE NOT NULL                  | X(1)   | IsDefault indicates whether the security constraint value assigned to the user or profile is the default value for the constraint. |
| Assignor       | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128) | Returns the name of the user that assigned the security constraint value to another user or profile.                               |

## Usage Notes

### Possible Values for IsDefault

| Value | Description     |
|-------|-----------------|
| Y     | Default         |
| N     | Not the default |

## ZoneGuestsV[X]

**Category:** Security

**Database:** DBC

| View Column      | Data Type  | Format              | Comment  |
|------------------|--|---------------------|--|
| ZoneName         | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128)              | Returns the name of a zone.                                |
| GuestName        | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128)              | Returns the name of a zone guest                           |
| GuestKind        | VARCHAR(4) UNICODE NOT<br>CASESPECIFIC NOT NULL      | X(4)                | Returns 'User' if the guest is user, otherwise 'Role'.     |
| GrantorName      | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128)              | Returns the name of a zone grantor.                        |
| GrantedTimeStamp | TIMESTAMP(0) NOT NULL                                | YYYY-MM-DDBHH:MI:SS | Returns the timestamp when the zone was granted to a user. |

## Usage Notes

### Possible Values for GuestKind

The GuestKind column returns 'User' if the guest is a user or it returns 'Role' if the guest is a a role.

## ZonesV[X]

**Category:** Security

**Database:** DBC

| View Column     | Data Type  | Format                 | Comment  |
|-----------------|--|------------------------|--|
| ZoneName        | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128)                 | Returns the name of a zone.  |
| RootName        | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128)                 | Returns the zone root name.  |
| RootType        | CHAR(1)<br>LATIN UPPERCASE                           | X(1)                   | Returns 'U' if the root of a zone is user and 'D' if the root of a zone is database. |
| ZoneDBAName     | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128)                 | Returns the zone's primary DBA name.   |
| CreatorName     | VARCHAR(128) UNICODE<br>NOT CASESPECIFIC<br>NOT NULL | X(128)                 | Returns the zone creator name.   |
| CreateTimeStamp | TIMESTAMP(0) NOT NULL                                | YYYY-MM-DD<br>HH:MI:SS | Returns the creation timestamp of a zone.  |

## Usage Notes

### Possible Values for RootType

| Value | Description                          |
|-------|--------------------------------------|
| U     | If the root of a zone is a user.     |
| D     | If the root of a zone is a database. |

# Data Dictionary Tables

You can use Teradata Studio or Teradata Studio Express to view the Data Dictionary. For information about these tools and to download them:

1. Go to <https://support.teradata.com>.
2. Log in.

## How Tables Are Created

The Data Dictionary tables are created during system initialization (sysinit), by the Database Initialization Program, or both. The tables store information about the system (system metadata) that is essential for it to function. The information is also essential to users that are responsible for managing the system.

The system automatically updates the information in the tables to reflect the current state of the system.

## Accessing Tables

Data Dictionary tables can be accessed only by users who have the required privileges to the tables. Access to the tables is strictly controlled to ensure that users (including system administrators) cannot modify them.

---

**Note:**

To ensure that the system functions properly, do not modify or delete any Data Dictionary tables. Use Data Dictionary views to access data in the tables to ensure that the tables are not accidentally modified or deleted.

---

For sites that use Teradata Secure Zones, SELECT privilege on DBQL tables is revoked from PUBLIC. Users can access DBQL table information only through DBC.QryLog\* views. This prevents zone users from accessing information not related to their zones. For more details, see [Teradata Secure Zone Views](#).

Zone users can see the information for objects in their zone in DBC views, but they cannot see the information for any objects outside their zone. If a non-zone user needs to see information for all of the zones, grant ZONE OVERRIDE privilege to the user. Users with ZONE OVERRIDE privilege can see information from all zones. You can also grant SELECT privilege on a specific view, but users with SELECT privileges on a view obtain different result sets, depending on their zone membership or non-zone status. See GRANT ZONE OVERRIDE and REVOKE ZONE OVERRIDE in *Teradata Vantage™ - SQL Data Control Language*, B035-1149.

Teradata recommends that you do not grant SELECT privileges on the following tables:

- DBC.DBQLExplainTbl
- DBC.DBQLObjTbl
- DBC.DBQLLogTbl

- DBC.DBQLSqlTbl
- DBC.DBQLStepTbl
- DBC.DBQLSummaryTbl
- DBC.DBQLXMLTbl
- DBC.DBQLUtilityTbl
- DBC.DBQLXMLLockTbl
- DBC.TDWMExceptionLog

## Nonhashed Tables

When a table is nonhashed, the rows of the table are not distributed using hash maps and are instead stored AMP-locally. For example, each row of the DBC.DatabaseSpace table keeps track of the space used locally by one AMP. Similarly, recovery tables only contain the information necessary to recover any activity or transaction that occurred on their own AMP.

Acctg and DatabaseSpace are allowed for SHOW TABLE, HELP, and so on, and these two tables can be queried if you have the privileges for executing the statement on the table. Note that SHOW TABLE and HELP INDEX indicate these two tables have unique primary indexes even though they are actually nonhashed tables.

The other tables have special formats for their rows and cannot be queried and are not allowed for SHOW TABLE and so on (an error is returned that the table cannot be accessed).

The following table lists the nonhashed and NO FALLBACK Data Dictionary tables.

| Nonhashed, NO FALLBACK Tables | Description                          |
|-------------------------------|--------------------------------------|
| Acctg                         | Resource usage by Acct/User          |
| ChangedRowJournal             | Down-AMP recovery journal            |
| DatabaseSpace                 | Database and table space accounting  |
| LocalSessionStatusTable       | Last request status by AMP           |
| LocalTransactionStatusTable   | Last transaction consensus status    |
| OrdSysChngTable               | AMP recovery journal                 |
| RecoveryLockTable             | Recovery session locks               |
| RecoveryPJTable               | Permanent journal recovery           |
| SavedTransactionStatus        | AMP recovery table                   |
| SysRcvStatJournal             | Recovery, Reconfig, and startup info |
| TransientJournal              | Backout uncommitted transactions     |
| UtilityLockJournalTable       | Host utility lock records            |

## DBCExtension Tables

The DBCExtension database is used to obtain the Global and Persistent (GLOP) sets used to define a particular mapping of data, how the data in the set is mapped, and the actual GLOP data that is mapped.

The following tables are contained in the DBCExtension system database.

| Table     | Description   |
|-----------|---|
| GLOP_Map  | The GLOP_Map table contains the possible mappings for the particular set.                       |
| GLOP_Set  | The GLOP_Set table describes the type of mapping for a set                                      |
| Glop_Data | The GLOP_Data table describes the actual data to be mapped for a particular GLOP set reference. |

## Related Topics

For information on the DBCExtension.GLOP\_Add, DBCExtension.GLOP\_Remove, DBCExtension.GLOP\_Change, and DBCExtension.GLOP\_Report stored procedures that provide user access to the tables, see *Teradata Vantage™ - SQL External Routine Programming*, B035-1147 or refer to the table column definitions in Teradata Studio or Teradata Studio Express.

## Updating Tables

When a data definition statement is processed, the system tables are updated automatically.

When a table is changed by an ALTER, CREATE, DROP, or RENAME statement, Vantage automatically increments the version count for that table. For more information about creating, changing, and dropping tables, see *Teradata Vantage™ - SQL Data Definition Language Detailed Topics*, B035-1184.

Most tables cannot be altered using INSERT or UPDATE, or DELETE SQL statements; however, Data Dictionary tables in the form of logs can be modified. See [Data Dictionary Logs that Require Manual Purging](#) for more information. Also, the object use count feature allows the AccessCount and LastAccessTimeStamp columns to be updated through SQL in the following dictionary tables: Dbase, TVM, TVFields, and Indexes.

## Character Data

Character data in the Data Dictionary affects the following kinds of view columns:

- Object Name
- Text Strings
- Other Character Data

## Object Names

To make the Data Dictionary field definitions consistent across all Teradata platforms, the following attributes apply to each object name:

- All object name columns are VARCHAR(128) CHARACTER SET UNICODE.
- The default format for the object names is defined as X(128).

## Text Strings

In the Data Dictionary, there are object names, SQL Text, and various other fields (for example, Title) that are Unicode.

## Other Character Data

The server character set of non-object name character fields in the Data Dictionary is Latin. These fields (for example, CHAR(1) or CHAR(2)) have a fixed set of values (for example, Y, N, A, CV, and so on) and contain characters from U+0020 to U+007F (for example, A is U+0041).

For more information about server character sets, see *Teradata Vantage™ - Advanced SQL Engine International Character Set Support*, B035-1125.

## Maintaining System Logs

Some Data Dictionary tables are in the form of logs, which accumulate increasing amounts of data over time. To avoid excess use of perm space, you should periodically delete older information from the log tables using either SQL DELETE requests or the Log Table Cleanup function in Teradata Viewpoint.

### Note:

Older log data can also be transferred to other storage media if site policy requires long-term log record retention. You can create backup log tables to store log history and periodically copy data from the active logs and archive it to backup tables or long-term table storage. using the BTEQ .EXPORT command, FastExport, or a Backup and Restore (BAR) Software.

## Data Dictionary Logs that Require Manual Purging

| Logs   | Description  |
|--|--|
| Logs that can be either purged manually or using Teradata Viewpoint. |  |
| Access Log<br>(DBC.AccLogTbl)  | Data collected as defined in a BEGIN LOGGING statement, for each privilege check of an attempted user access of a database object. |
| DBQ Logs:<br>• DBC.DBQLSummaryTbl                                    | Data collected during database query logging (DBQL), if DBQL is enabled.   |

| Logs   | Description   |
|--|---|
| <ul style="list-style-type: none"> <li>• DBC.DBQLStepTbl</li> <li>• DBC.DBQLogTbl</li> <li>• DBC.DBQLObjTbl</li> <li>• DBC.DBQLExplainTbl</li> <li>• DBC.DBQLSqlTbl</li> </ul>   |   |
| Event Log<br>(DBC.EventLog)  | Data collected automatically by the system for each user logon event.   |
| ResUsage Logs: <ul style="list-style-type: none"> <li>• DBC.ResUsageSpma</li> <li>• DBC.ResUsageSvpr</li> <li>• DBC.ResUsageShst</li> <li>• DBC.ResUsageIpma</li> <li>• DBC.ResUsageIvpr</li> <li>• DBC.ResUsageScpu</li> <li>• DBC.ResUsageSldv</li> <li>• DBC.ResUsageSawt</li> <li>• DBC.ResUsageSps</li> <li>• DBC.ResUsageSpdsk</li> <li>• DBC.ResUsageSvdsk</li> </ul> | ResUsage tables collect data on system resource usage. You can enable ResUsage data collection and set the collection rate using either: <ul style="list-style-type: none"> <li>• The ctl utility SCREEN RSS command</li> <li>• Teradata Viewpoint data collectors</li> </ul> The collection rate determines how quickly data accumulates and when the logs should be purged. |
| SWEvent Log<br>(DBC.SW_Event_Log)  | The system automatically inserts rows in response to software errors and system events, for use by Teradata Customer Service.   |
| TDWM Logs: <ul style="list-style-type: none"> <li>• DBC.TDWMSummaryLog</li> <li>• DBC.TDWMEventLog</li> <li>• DBC.TDWMEExceptionLog</li> </ul>   | Logs for Teradata Viewpoint workload management functions.  |
| Logs that can only be purged manually.   |   |
| In Doubt transaction Log<br>(DBC.InDoubtResLog)  | Contains a row for each transaction where completion was in doubt.  |

## Manually Deleting Old Log Data

You can use the Teradata SQL DELETE statement to purge outdated log data, for example DBC.LogOnOffV log data.

### Note:

The user executing the SQL statement must have the DELETE privilege on the corresponding view.

For example:

You can purge DBC.LogOnOffV log data based on age. Teradata recommends retaining 90 days of log data and purging directly from the DBC.EventLog table.

```
DELETE FROM DBC.EventLog
WHERE (DATE-DateFld) > 90;
```

You can purge information for an account after a cost accounting period has closed.

```
DELETE FROM DBC.AMPUsageV
WHERE Accountname = '$M619';
```

## Related Topics

| For more information on ...   | See ...   |
|---|---|
| how to manually purge old logs or copy the data to backup tables              | <i>Teradata Vantage™ - Database Administration</i> , B035-1093. |
| using the Log Table Clean Up function in Teradata Viewpoint to purge log data | <i>Teradata® Viewpoint User Guide</i> , B035-2206.              |

# View Column Values

Many columns can be found in multiple system views or tables. The values of the most common columns in the system are listed below.

## AccessRight Column

| Value | Description               |
|-------|---------------------------|
| AE    | ALTER EXTERNAL PROCEDURE  |
| AF    | ALTER FUNCTION            |
| AP    | ALTER PROCEDURE           |
| AS    | ABORT SESSION             |
| CA    | CREATE AUTHORIZATION      |
| CD    | CREATE DATABASE           |
| CE    | CREATE EXTERNAL PROCEDURE |
| CF    | CREATE FUNCTION           |
| CG    | CREATE TRIGGER            |
| CM    | CREATE MACRO              |
| CO    | CREATE PROFILE            |
| CP    | CHECKPOINT                |
| CR    | CREATE ROLE               |
| CS    | CREATE SERVER             |
| CT    | CREATE TABLE              |
| CU    | CREATE USER               |
| CV    | CREATE VIEW               |
| CZ    | CREATE ZONE               |
| C1    | CREATE DATASET SCHEMA     |
| D     | DELETE                    |
| DA    | DROP AUTHORIZATION        |
| DD    | DROP DATABASE             |

| Value | Description            |
|-------|------------------------|
| DF    | DROP FUNCTION          |
| DG    | DROP TRIGGER           |
| DM    | DROP MACRO             |
| DO    | DROP PROFILE           |
| DP    | DUMP                   |
| DR    | DROP ROLE              |
| DS    | DROP SERVER            |
| DT    | DROP TABLE             |
| DU    | DROP USER              |
| DV    | DROP VIEW              |
| DZ    | DROP ZONE              |
| D1    | DROP DATASET SCHEMA    |
| E     | EXECUTE (MACRO)        |
| EF    | EXECUTE FUNCTION       |
| GC    | CREATE GLOP            |
| GD    | DROP GLOP              |
| GM    | GLOP MEMBER            |
| I     | INSERT                 |
| IX    | INDEX                  |
| MC    | CREATE MAP             |
| MD    | DROP MAP               |
| MR    | MONITOR RESOURCE       |
| MS    | MONITOR SESSION        |
| NT    | NONTEMPORAL            |
| OA    | OVERRIDE DUMP          |
| OD    | OVERRIDE DELETE POLICY |
| OI    | OVERRIDE INSERT POLICY |
| OP    | CREATE OWNER PROCEDURE |
| OR    | OVERRIDE RESTORE       |

| Value | Description                                  |
|-------|--|
| OS    | OVERRIDE SELECT POLICY                       |
| OU    | OVERRIDE UPDATE POLICY                       |
| PC    | CREATE PROCEDURE                             |
| PD    | DROP PROCEDURE                               |
| PE    | EXECUTE PROCEDURE                            |
| R     | RETRIEVE/SELECT                              |
| RF    | REFERENCE                                    |
| RO    | REPLCONTROL                                  |
| RS    | RESTORE                                      |
| SA    | SECURITY CONSTRAINT ASSIGNMENT (system wide) |
| SD    | SECURITY CONSTRAINT DEFINITION (system wide) |
| SH    | SHOW   |
| SR    | SET RESOURCE RATE                            |
| SS    | SET SESSION RATE                             |
| ST    | STATISTICS                                   |
| TH    | CTCONTROL                                    |
| U     | UPDATE                                       |
| UM    | UDT METHOD                                   |
| UT    | UDT TYPE                                     |
| UU    | UDT USAGE                                    |
| W1    | WITH DATASET SCHEMA                          |
| ZO    | ZONE OVERRIDE                                |

## ConstraintType Column

| Value | Description                           |
|-------|---------------------------------------|
| C     | Explicit table-level constraint check |
| P     | Nonpartitioned Primary Index          |
| Q     | Partitioning constraint               |

| Value | Description                               |
|-------|---|
| S     | Hash-ordered Secondary Index without ALL  |
| K     | Primary key                               |
| U     | Unique constraint                         |
| R     | References constraint                     |
| V     | Value-ordered Secondary Index without ALL |
| H     | Hash-ordered Secondary Index with ALL     |
| O     | Value-ordered secondary with ALL          |

## ExceptionValue Column

These values show up as an integer. A conversion to hex extracts the bit values. For example, a value of 1024 converted to hex is 400.

| Value      | Description                           |
|------------|---------------------------------------|
| 0x00000001 | Exception time limit exceeded         |
| 0x00000002 | CPU time (AMP and PE) limit exceeded  |
| 0x00000004 | Blocked time limit exceeded           |
| 0x00000008 | Disk to CPU ratio exceeded            |
| 0x00000010 | AMP CPU skew limit exceeded           |
| 0x00000020 | AMP I/O count limit exceeded          |
| 0x00000040 | AMP I/O skew limit exceeded           |
| 0x00000080 | Max row count (for a step) exceeded   |
| 0x00000100 | Max row count (for a query) exceeded  |
| 0x00000200 | Spool space limit exceeded            |
| 0x00000400 | Number of AMPS used in query exceeded |
| 0x00000800 | Disk CPU ratio value exceeded         |
| 0x00001000 | I/O space value exceeded              |

## JournalFlag Column

| Value | Description                               |
|-------|---|
| N     | No journal (default)                      |
| S     | Single journal                            |
| D     | Dual journal                              |
| L     | Local AFTER journal (not used for BEFORE) |

## NoSQLDataAccess Column

| Value | Description                              |
|-------|--|
| Y     | No SQL in the external stored procedures |
| C     | Contains SQL                             |
| R     | Reads SQL data                           |
| M     | Modifies SQL data                        |

## ProcessOffline Column

### Note:

Tables undergoing large-scale changes should not be processed during online redistribution. Flag these tables for offline redistribution by setting this column to Y.

| Value | Description  |
|-------|--|
| Y     | Table should be process (deleted or redistributed) offline during the offline portion of the Reconfig deletion or redistribution phase.                |
| N     | Table should be processed (deleted or redistributed) online during the online of the Reconfig deletion or redistribution phase. (This is the default.) |

## RoutineKind Column

| Value | Description    |
|-------|----------------|
| C     | Constructor    |
| D     | Decompress UDF |
| E     | Compress UDF   |

| Value | Description      |
|-------|------------------|
| M     | Mutator          |
| O     | Observer         |
| R     | Regular function |

## StatsSource Column

| Value | Description                                      |
|-------|--|
| I     | Internally generated                             |
| S     | User collected with COLLECT STATS (system built) |
| U     | User collected with COLLECT STATS VALUES clause  |
| C     | Copied from other sources                        |
| T     | Transferred with CREATE TABLE...AS statement     |

## TableKind Column

| Value | Description   |
|-------|---|
| A     | Aggregate function  |
| B     | Combined aggregate and ordered analytical function  |
| C     | Table operator parser contract function   |
| D     | JAR   |
| E     | External stored procedure   |
| F     | Standard function   |
| G     | Trigger   |
| H     | Instance or constructor method  |
| I     | Join index  |
| J     | Journal   |
| K     | Foreign server object.<br><b>Note:</b><br>K is supported on the Teradata-to-Hadoop and Teradata-to-Teradata connectors. |
| L     | User-defined table operator   |

| Value | Description  |
|-------|--|
| M     | Macro  |
| N     | Hash index   |
| O     | Table with no primary index and no partitioning  |
| P     | Stored procedure   |
| Q     | Queue table  |
| R     | Table function   |
| S     | Ordered analytical function  |
| T     | Table with a primary index or primary AMP index, partitioning, or both. Or a partitioned table with NoPI |
| U     | User-defined type  |
| V     | View   |
| X     | Authorization  |
| Y     | GLOP set   |
| Z     | UIF  |
| 1     | A DATASET schema object created by CREATE SCHEMA.  |
| 2     | Function alias object.   |

## TimeDimension Column

| Value | Description                                   |
|-------|---|
| N     | Nontemporal column (default)                  |
| R     | Temporal relationship constraint (TRC) column |
| T     | Transaction-time column                       |
| V     | Valid-time column                             |
| S     | SYSTEM_TIME derived period column             |

## VTCheckType Column

| Value | Description             |
|-------|-------------------------|
| NULL  | No valid-time dimension |

| Value | Description            |
|-------|------------------------|
| C     | CURRENT VALIDTIME      |
| S     | SEQUENCED VALIDTIME    |
| N     | NONSEQUENCED VALIDTIME |

# LogonSource Column Fields and Examples

## LogonSource Column

### Note:

Teradata recommends using alternative columns rather than the LogonSource column, if available. The sections below provide information about the recommended columns for the sessions in which they are available.

The LogonSource column includes information about the source of sessions logged on from a client, including information on the TDP and job name. It is referenced in the following:

- Views:
  - LogOnOffV[X]
  - SessionInfoV[X]
  - QryLogV
- Tables:
  - EventLog
  - SessionTbl

## Mainframe-Attached Systems Using the CLlv2 API

The origin of the CLlv2 mainframe session being reported, such as the user ID or session number of the client system. LogonSource can contain the following names and identifiers.

Unless otherwise noted, each of the LogonSource string fields contains eight characters.

| Field Name                         | Description   |
|------------------------------------|---|
| Operating System Name<br>(Field 1) | Name of the client operating system for this logon.<br>The valid name is MVS. |
| TDP ID<br>(Field 2)                | Unique identifier for the mainframe TDP controlling this logon.               |
| Job Name<br>(Field 3)              | This is the job name.   |
| Environment Name<br>(Field 4)      | The valid environments are BATCH, CICS, IMS, and TSO.                         |

| Field Name                                 | Description   |
|--|---|
| User ID from Security Product<br>(Field 5) | This is the user ID provided by the security product in use.<br><b>Note:</b><br>If no security product is in use, this is blank.  |
| Group from Security Product<br>(Field 6)   | This is the group ID provided by the security product in use.<br>If no security product is in use, this is blank.   |
| Program Name<br>(Field 7)                  | Use of this field is deprecated because it is not possible to determine whether it specifies an executable application name or a unique identifier provided by an application using the CLlv2 Workload specification (see <i>Teradata® Call-Level Interface Version 2 Reference for Mainframe-Attached Systems</i> , B035-2417). Instead, you should use the Actual Program Name field, which provides a reliable name for the executable application |
| Coordinator Name<br>(Field 8)              | Name of the Coordinator if the logon is part of a two-phase commit logon. This is used only by the CICS and IMS mainframe transaction managers; otherwise, it is blank.   |
| Transaction ID<br>(Field 9)                | Unique identifier for the transaction as determined by the transaction manager. This is used only by the CICS and IMS mainframe transaction managers; otherwise, it is blank.   |
| Terminal ID<br>(Field 10)                  | Unique identifier for the terminal as determined by CICS. This is used only for CICS; otherwise, it is blank.   |
| User/Operator ID<br>(Field 11)             | Unique identifier for the user or operator as determined by CICS. This is used only for CICS; otherwise, it is blank.   |
| Actual Program Name<br>(Field 12)          | This field contains the name of the executable application.   |
| Job ID<br>(Field 13)                       | This field contains the job ID.   |
| Format ID<br>(Field 14)                    | Format identifier that can be used to parse the earlier positional information in the LogonSource attribute. If the field ends with the following four characters, the format ID is present:<br><br>LSS   |

| Field Name  | Description  |
|---|--|
|   | <p><b>Note:</b></p> <p>In the example above, the first character of the four character format ID is a blank space.</p> <p>The first set of two characters indicate the number of values found before the Actual Program Name, which is either 07 or 11.</p> <p>The second set of two characters indicates the version of the format ID.</p> <p>If the value is 01, the field contains the Job ID.</p> <p>If the value is 02, the field contains both the Actual Program Name and the Job ID value.</p> <p>If no Format ID is present, no Job ID value exists and only the first 7 or 11 fields of the LogonSource string are present.</p>  |
| Self-Defining EBCDIC Items<br>(Field 15 or later) | <p>The following self-defining EBCDIC items have variable lengths and contain the specified information within the parentheses that follow the field name in the LogonSource string:</p> <ul style="list-style-type: none"> <li>• <b>TDP():</b> TDP release ID. See <i>Teradata® Director Program Reference</i>, B035-2416.</li> <li>• <b>CL2():</b> CLlv2 release ID. See <i>Teradata® Call-Level Interface Version 2 Reference for Mainframe-Attached Systems</i>, B035-2417.</li> <li>• <b>SESSDESC():</b> Descriptive information specified by the application using the CLlv2 Session-desc specification. See <i>Teradata® Call-Level Interface Version 2 Reference for Mainframe-Attached Systems</i>, B035-2417.</li> <li>• <b>WORKLOAD():</b> Information specified by the application using the CLlv2 Workload specification. The information is intended to be used by Vantage. See <i>Teradata® Call-Level Interface Version 2 Reference for Mainframe-Attached Systems</i>, B035-2417.</li> </ul> <p><b>Note:</b></p> <p>If the length of the self-defining items causes the LogonSource string to exceed 128 characters, Vantage truncates them as necessary and indicates the truncation with the string '...' if enough space remains to do so.</p> |

## Workstation-Attached Systems Using CLlv2 API

The origin of the CLlv2 workstation-attached session being reported, such as the user ID or session number of the client system.

**Note:**

The corresponding LogOnOffV[X] or SessionInfoV[X] column shown in the table below appears in the LogOnOffV[X] and SessionInfoV[X] views and EventLog and SessionTbl tables only.

Teradata strongly recommends that applications which use the LogonSource fields instead use the corresponding LogOnOffV[X] or SessionInfoV[X] columns, also referred to as ClientAttribute columns.

You can use Teradata Studio or Teradata Studio Express to learn more about the ClientAttribute columns.

| Field Name                             | Description   | Corresponding LogOnOffV[X] or SessionInfoV[X] Column | Description   |
|--|---|--|---|
| Mode<br>(Field 1)                      | <p>The literal string indicating the connection type, for example:</p> <p>TCP/IP</p> <p>This field is always 8 characters long.</p> <p>This information is provided by the gateway.</p>   | ClientConnectionType                                 | <p>Value is 1 indicating the client is connected using TCP/IP via the gateway.</p> <p>To learn more about the possible values for this column see <a href="#">Possible Values for ClientConnectionType</a> or you can use Teradata Studio or Teradata Studio Express.</p> |
| TCP Port or Socket Number<br>(Field 2) | <p>Hexadecimal TCP port or socket number on the workstation-attached client system.</p> <p>This field contains 4 hexadecimal characters.</p> <p>The maximum value is 64K -1</p> <p>This information is provided by the gateway.</p> | ClientTcpPortNumber                                  | Integer value of the TCP port or socket number.   |
| IP Address<br>(Field 3)                | <p>IP address of the workstation-attached client system.</p> <p>This field contains a maximum of 45 characters.</p> <p>This information is provided by the gateway.</p>   | ClientIpAddress                                      | Standard string representation of the IPv4 or IPv6 IP address.  |
| TDP ID<br>(Field 4)                    | <p>Unique ID for the network TDP controlling this logon.</p> <p>The number of characters depends on the value for the TDP ID that was specified when you logged on.</p>   | ClientTdHostName                                     | Vantage host name that the client used to connect to Vantage.   |

| Field Name                  | Description  | Corresponding LogOnOffV[X] or SessionInfoV[X] Column | Description  |
|-----------------------------|--|--|--|
| Client Process ID (Field 5) | ID for the process on the workstation-attached client system.<br>This field contains a maximum of 10 decimal digits.   | ClientProcThreadId                                   | Client process or thread ID.   |
| Client User ID (Field 6)    | User ID for the logged on user as defined for the workstation-attached client system.<br>This field contains a maximum of 92 characters.   | ClientSystemUserId                                   | Client user ID.  |
| Client Program (Field 7)    | ID for the CLlv2-based client program.<br>This field contains a maximum of 256 characters.   | ClientProgramName                                    | Client system program name.  |
| Format ID (Field 8)         | A format ID that can be used to parse the earlier positional information in the LogonSource attribute.<br>If the field ends with the following four characters, the format ID is present:<br>LSS<br>(The first character in this example is blank.)  | ClientAttributesEx                                   | Description of the client that does not match any of the other client attributes fields. |
| LSINFO (Field 9)            | Optional text derived from the LSINFO environment variable.<br>The size of LSINFO depends on how many characters remain available in LogonSource after fields 1 through 8 have been assembled.<br>If the size of LSINFO exceeds the remaining space in LogonSource, CLlv2 truncates the data to fit the available space. | ClientJobData  | Client job data from the LSINFO environmental variable.                                  |

## Example: An Archive Job

(TCP/IP) 07DC 10.243.71.25 DW\_OLD 2482 ROOT ARCMAN 01 LSS

Field contents:

| Field              | Contents     | Description  |
|--------------------|--------------|--|
| 1                  | TCP/IP       | Connection mode  |
| 2                  | 07DC         | TCP port or societ identifier                                |
| 3                  | 10.243.71.25 | IP address of the client system                              |
| 4                  | DW_OLD       | TDP ID for the TDP making the connection with Teradata Datab |
| ase for this logon |              |  |
| 5                  | 2482         | Client process identifier                                    |
| 6                  | ROOT         | Client system user ID  |
| 7                  | ARCMAN       | Client program   |
| 8                  | 01 LSS       | Format ID  |

## Example: A BTEQ Job

```
(TCP/IP) 0675 141.206.34.18 CS4400S1 2304 AG110058 BTEQ 01 LSS "THIS IS
A TEST!"
```

## JDBC Driver API

When the application writing to LogonSource connects to Vantage using the JDBC driver, the definitions of the LogonSource string fields are as follows.

### Note:

The corresponding LogOnOffV[X] or SessionInfoV[X] column shown in the table below appears in the LogOnOffV[X] and SessionInfoV[X] views and EventLog and SessionTbl tables only.

Teradata strongly recommends that applications which use the LogonSource fields instead use the corresponding LogOnOffV[X] or SessionInfoV[X] columns, also referred to as ClientAttribute columns.

You can use Teradata Studio or Teradata Studio Express to learn more about the ClientAttribute columns.

| Field Name     | Description  | Corresponding LogOnOffV[X] or SessionInfoV[X] Column | Description  |
|----------------|--|--|--|
| Mode (Field 1) | The literal string indicating the connection type, for example:<br><br>TCP/IP<br><br>This field is always 8 characters long. | ClientConnectionType                                 | Value is 1 indicating the client is connected using TCP/IP via the gateway.<br>To learn more about the possible values for this column see |

| Field Name                          | Description  | Corresponding LogOnOffV[X] or SessionInfoV[X] Column | Description   |
|-------------------------------------|--|--|---|
|                                     | This information is provided by Vantage.   |  | <a href="#">Possible Values for ClientConnectionType</a> or you can use Teradata Studio or Teradata Studio Express. |
| TCP Port or Socket Number (Field 2) | Hexadecimal TCP port or socket number on the client system.<br>This field contains 4 hexadecimal characters.<br>The maximum value is 64K -1<br>This information is provided by Vantage.  | ClientTcpPortNumber                                  | Integer value of the TCP port or socket number.   |
| IP Address (Field 3)                | IP address of the client system.<br>This field contains a maximum of 45 characters.<br>This information is provided by Vantage.  | ClientIpAddress                                      | Standard string representation of the IPv4 or IPv6 IP address.  |
| TDP ID (Field 4)                    | TDP ID for the TDP making the connection with Vantage for this logon.<br>This field fills whatever space remains after fields 5 through 8 have been filled.<br>This field contains a maximum of 97 characters, and is truncated to the space remaining after all the other fields have been written.<br>This field comprises 5 components: <ul style="list-style-type: none"> <li>• Vantage host name specified by the application.</li> <li>• SEMICOLON character.</li> <li>• Host name or IP address of Vantage node that was connected.</li> <li>• COLON character.</li> <li>• Port number of Vantage node that was connected.</li> </ul> This information is provided by the Teradata JDBC driver. | ClientTdHostName                                     | Vantage host name that the client used to connect to Vantage.   |
| Client Process/ Thread ID (Field 5) | An ID for the Java process or thread on the client system.<br>The field contains a maximum of 12 characters.<br>Because Java threads are not tied to particular database connections, any thread can execute requests on a   | ClientProcThreadId                                   | Client process or thread ID.  |

| Field Name               | Description  | Corresponding LogOnOffV[X] or SessionInfoV[X] Column | Description  |
|--------------------------|--|--|--|
|                          | <p>connection that was originally created by a different thread.</p> <p>To avoid confusion, the Teradata JDBC driver provides a unique connection ID, which is derived from the hash code of the connection object.</p> <p>This field comprises 2 components:</p> <ul style="list-style-type: none"> <li>The four-character literal string, for example:<br/>CID=</li> <li>The unique connection ID.</li> </ul> <p>This information is provided by the Teradata JDBC driver.</p>   |  |  |
| Client User ID (Field 6) | <p>User ID for the logged on user as defined for the client system.</p> <p>This field contains a maximum of 20 characters.</p> <p>This information is provided by the Teradata JDBC driver.</p>  | ClientSystemUserId                                   | Client user ID.  |
| Client Program (Field 7) | <p>An ID for the Java client program.</p> <p>This field contains a maximum of 26 characters.</p> <p>This field comprises 4 components:</p> <ul style="list-style-type: none"> <li>The literal string JDBC.</li> <li>The JDBC driver version.</li> <li>SEMICOLON character.</li> <li>The system version number:<br/>getProperty</li> </ul> <p>This field is retrieved using the following method:</p> <pre>System.<br/>getProperty("java.version")</pre> <p>This information is provided by the Teradata JDBC driver.</p> | ClientProgramName                                    | Client system program name.  |
| Format ID (Field 8)      | <p>A format ID that can be used to parse the earlier positional information in the LogonSource attribute.</p> <p>If the field ends with the following four characters, the format ID is present:</p>   | ClientAttributesEx                                   | Description of the client that does not match any of the other client attributes fields. |

| Field Name | Description   | Corresponding LogOnOffV[X] or SessionInfoV[X] Column | Description |
|------------|---|--|-------------|
|            | <p>LSS</p> <p><b>Note:</b><br/>In the example above, the first character in the four character format ID is a blank space.<br/>This field is always 6 characters long.<br/>For JDBC connections, this value is always:</p> <p>01 LSS</p> <p>This information is provided by the Teradata JDBC driver.</p> |  |             |

## Example: LogonSource

```
(TCP/IP) 137D 153.64.135.140 CS4400S1;CS4400S1COP1/153.64.208.223:1025
CID=337D0F TN180005 JDBC14.10.00.00;1.4.2_o4 01 LSS
```

Field contents:

| d    | Field       | Contents                                 | Description   |
|------|-------------|--|---|
| ---- | -----       | -----                                    | -----   |
| 1    |             | TCP/IP                                   | Connection mode   |
| 2    |             | 137D                                     | TCP port/societ number on the network system  |
| 3    |             | 153.64.135.140                           | IP address of the network client system   |
| 4    |             | CS4400S1;CS4400S1COP1/153.64.208.223:102 |   |
| 5    | TDP         |  | ID for the TDP making the connection with Teradata Database for this logon5CID=337DoF |
|      | Client Java |  |   |
|      |             |  | process/thread ID   |
| 6    |             | TN180005                                 | Client user ID  |
| 7    |             | JDBC14.10.00.00;1.4.2_o4                 | Name of the Java client program   |
| 8    |             | 01 LSS                                   | Format ID   |

## ODBC Driver API

The origin of the ODBC session being reported, such as the user ID or process ID of the client system.

When the application writing to LogonSource connects to Vantage using the ODBC driver, the definitions of the LogonSource string fields are as follows.

**Note:**

The corresponding LogOnOffV[X] or SessionInfoV[X] column shown in the table below appears in the LogOnOffV[X] and SessionInfoV[X] views and EventLog and SessionTbl tables only.

Teradata strongly recommends that applications which use the LogonSource fields instead use the corresponding LogOnOffV[X] or SessionInfoV[X] columns, also referred to as ClientAttribute columns.

You can use Teradata Studio or Teradata Studio Express to learn more about the ClientAttribute columns.

| Field Name                             | Description   | Corresponding LogOnOffV[X] or SessionInfoV[X] Column | Description  |
|--|---|--|--|
| Mode<br>(Field 1)                      | The literal string indicating the connection type, for example:<br>TCP/IP<br><br>This field is always 8 characters long.<br>This information is provided by Vantage.                            | ClientConnectionType                                 | Value is 1 indicating the client is connected using TCP/IP via the gateway.<br>To learn more about the possible values for this column see <a href="#">Possible Values for ClientConnectionType</a> or you can use Teradata Studio or Teradata Studio Express. |
| TCP Port or Socket Number<br>(Field 2) | Hexadecimal TCP port or socket number on the network client system.<br>This field contains 4 hexadecimal characters.<br>The maximum value is 64K -1<br>This information is provided by Vantage. | ClientTcpPortNumber                                  | Integer value of the TCP port or socket number.  |
| IP Address<br>(Field 3)                | IP address of the network client system.<br>This field contains a maximum of 45 characters.<br>This information is provided by Vantage.   | ClientIpAddress                                      | Standard string representation of the IPv4 or IPv6 IP address.   |
| TDP ID<br>(Field 4)                    | TDP ID for the network TDP making the connection with Vantage for this logon.<br>This field fills whatever space remains after fields 5 and 6 have been filled.                                 | ClientTdHostName                                     | Vantage host name that the client used to connect to Vantage.  |

| Field Name                  | Description  | Corresponding LogOnOffV[X] or SessionInfoV[X] Column | Description                  |
|-----------------------------|--|--|------------------------------|
|                             | <p>This field comprises 5 components:</p> <ul style="list-style-type: none"> <li>• Vantage host name specified by the application.</li> <li>• SEMICOLON character.</li> <li>• Host name or IP address of Vantage node that was connected.</li> <li>• COLON character.</li> <li>• Port number of Vantage node that was connected.</li> </ul> <p>This information is provided by the Teradata ODBC driver.</p> |  |                              |
| Client Process ID (Field 5) | <p>ID for the ODBC process on the network client system.</p> <p>This field contains a maximum of 12 characters.</p> <p>This information is not provided by the Teradata ODBC driver.</p>   | ClientProcThreadId                                   | Client process or thread ID. |
| Client User ID (Field 6)    | <p>User ID for the logged on user as defined for the network client system.</p> <p>This field contains a maximum of 20 characters.</p> <p>This information is provided by the Teradata ODBC driver.</p>  | ClientSystemUserId                                   | Client user ID.              |

## Example: Teradata ODBC Driver

(TCP/IP) 0F6D 141.206.34.228 CS4400S1.TD.TERADATA.COM 3780 GR120994

Field contents:

| Field | Contents                 | Description  |
|-------|--------------------------|--|
| 1     | TCP/IP                   | Connection mode  |
| 2     | 0F6D                     | TCP port/societ number on the network system                                   |
| 3     | 141.206.34.228           | IP address of the network client system  |
| 4     | CS4400S1.FP.TERADATA.COM | TDP ID for the TDP making the connection with Teradata Database for this logon |

|   |          |                |
|---|----------|----------------|
| 5 | 3780     | Process ID     |
| 6 | GR120994 | Client user ID |

## .NET Data Provider for Teradata API

The origin of the .NET Data Provider for Teradata session being reported, such as the user ID or process ID of the client system. LogonSource can contain the following names and identifiers.

When the application writing to LogonSource connects to Vantage using the .NET Data Provider driver, the definitions of the LogonSource string fields are as follows.

### Note:

The corresponding LogOnOffV[X] or SessionInfoV[X] column shown in the table below appears in the LogOnOffV[X] and SessionInfoV[X] views and EventLog and SessionTbl tables only.

Teradata strongly recommends that applications which use the LogonSource fields instead use the corresponding LogOnOffV[X] or SessionInfoV[X] columns, also referred to as ClientAttribute columns.

You can use Teradata Studio or Teradata Studio Express to learn more about the ClientAttribute columns.

| Field Name                          | Description  | Corresponding LogOnOffV[X] or SessionInfoV[X] Column | Description  |
|-------------------------------------|--|--|--|
| Mode (Field 1)                      | The literal string indicating the connection type, for example:<br>TCP/IP<br><br>This field is always 8 characters long.<br>This information is provided by Vantage.                             | ClientConnectionType                                 | Value is 1 indicating the client is connected using TCP/IP via the gateway.<br>To learn more about the possible values for this column see <a href="#">Possible Values for ClientConnectionType</a> or you can use Teradata Studio or Teradata Studio Express. |
| TCP Port or Socket Number (Field 2) | Hexadecimal TCP port or socket number on the network client system.<br>This field contains 4 hexadecimal characters.<br>The maximum value is 64K -1.<br>This information is provided by Vantage. | ClientTcpPortNumber                                  | Integer value of the TCP port or socket number.  |
| IP Address (Field 3)                | IP address of the network client system.<br>This field contains a maximum of 45 characters.<br>This information is provided by Vantage.  | ClientIpAddress                                      | Standard string representation of the IPv4 or IPv6 IP address.   |
| TDP ID (Field 4)                    | TDP ID for the network TDP making the connection with Vantage for this logon.<br>This field fills whatever space remains after fields 5 through 8 have been filled.                              | ClientTdHostName                                     | Vantage host name that the client used to connect to Vantage.  |

| Field Name                  | Description  | Corresponding LogOnOffV[X] or SessionInfoV[X] Column | Description                  |
|-----------------------------|--|--|------------------------------|
|                             | <p>Fields 4 through 8 contains a maximum of 97 characters, and Vantage truncates the TDP ID field to the space remaining after all the other fields have been written. If not enough space remains, then the data for TDP ID is removed from the LogonSource string.</p> <p>This field comprises 5 components:</p> <ul style="list-style-type: none"> <li>• DataSource name specified by the application.</li> <li>• COLON character.</li> <li>• IP address of Vantage node that was connected.</li> <li>• COLON character.</li> <li>• Port number of Vantage node that was connected.</li> </ul> <p>This information is provided by .NET Data Provider for Teradata.</p>  |  |                              |
| Client Process ID (Field 5) | <p>ID for the client process.</p> <p>.NET Data Provider removes this field if both the TDP ID and Client User ID fields have been removed, but the resulting string is still longer than the allotted 97 characters for .NET Data Provider for Teradata-supplied fields in LogonSource.</p> <p>The field is retrieved using the following function call:</p> <pre>Process.GetCurrentProcessID().ID</pre> <p>This information is provided by .NET Data Provider for Teradata.</p>   | ClientProcThreadId                                   | Client process or thread ID. |
| Client User ID (Field 6)    | <p>User ID for the logged on user as defined for the client system.</p> <p>The field comprises 3 components:</p> <ul style="list-style-type: none"> <li>• The domain name.</li> </ul> <p>This value is retrieved using the following function call:</p> <pre>System.Environment.UserName</pre> <p>If a security exception is encountered, the field contains a QUESTION MARK character.</p> <ul style="list-style-type: none"> <li>• The REVERSE SOLIDUS (\) character (U+005C).</li> <li>• The user name.</li> </ul> <p>The value is retrieved using the following function call:</p> <pre>System.Environment.UserName</pre> <p>If a security exception is encountered, the field contains a QUESTION MARK character.</p> <p>This information is provided by .NET Data Provider for Teradata.</p> | ClientSystemUserId                                   | Client user ID.              |
| Client Program (Field 7)    | <p>ID for the program running on the client system.</p> <p>This field comprises 7 components:</p> <ul style="list-style-type: none"> <li>• The program name.</li> </ul> <p>This value is retrieved using the following function call:</p> <pre>Process.GetCurrentProcess().ProcessName</pre> <p>If a security exception is encountered, the field contains a QUESTION MARK character.</p> <ul style="list-style-type: none"> <li>• COLON character.</li> <li>• The literal character string NET.</li> </ul> <p>This identifies .NET Data Provider for Teradata.</p>  | ClientProgramName                                    | Client system program name.  |

| Field Name          | Description  | Corresponding LogOnOffV[X] or SessionInfoV[X] Column | Description  |
|---------------------|--|--|--|
|                     | <ul style="list-style-type: none"> <li>COLON character.</li> <li>The literal character string: SS<br/>This identifies the session as a standard SQL session.</li> <li>COLON character.</li> <li>The 12-character release number for the .NET Data Provider for Teradata in the format <i>nn.nn.nn.nnn</i>, where <i>n</i> represents an integer value.</li> </ul> <p>This information is provided by .NET Data Provider for Teradata.</p>  |  |  |
| Format ID (Field 8) | <p>Format ID that can be used to parse the earlier positional information in the LogonSource attribute.</p> <p>If the field ends with the following four characters, the format ID is present:</p> <pre>LSS</pre> <p><b>Note:</b><br/>In the example above, the first character in the four character format ID is a blank space.<br/>This field is always 6 characters long.<br/>For .NET Data Provider for Teradata connections, this value is always:</p> <pre>01 LSS</pre> <p>This information is provided by .NET Data Provider for Teradata.</p> | ClientAttributesEx                                   | Description of the client that does not match any of the other client attributes fields. |

**Note:**

The data in fields 4 through 7 can be truncated or removed entirely from the LogonSource string as required to ensure that the total length of the fields provided by .NET Data Provider for Teradata does not exceed 97 characters.

When field values must be eliminated, they are eliminated in the following order:

- TDP ID (Field 4)
- Client User ID (Field 6)
- Client Process/Thread ID (Field 5)

**Example: .NET Data Provider for Teradata API**

```
(TCP/IP) 057E 153.64.135.76 SALES:153.64.116.95:1025 3808 CORP\TDUSER
TESTAPP:NET:SS:12.00.00.000 01 LSS
```

Field contents:

| <u>Field</u> | <u>Contents</u>             | <u>Description</u>   |
|--------------|-----------------------------|--|
| 1            | TCP/IP                      | Connection mode  |
| 2            | 057E                        | TCP port/socket number on the network system                                   |
| 3            | 153.64.135.76               | IP address of the network client system  |
| 4            | SALES:153.64.116.95:1025    | TDP ID for the TDP making the connection with Teradata Database for this logon |
| 5            | 3608                        | Client process ID  |
| 6            | CORP\TDUSER                 | Client user ID   |
| 7            | TESTAPP:NET:SS:12.00.00.000 | Client program   |
| 8            | 01 LSS                      | The literal 01 LSS, indicating LogonSource string version 01                   |

## PDE Internal Session

The origin of the PDE internal session being reported, such as the node name or process ID of the client system. LogonSource can contain the following names and identifiers.

| Field Name                    | Description   |
|-------------------------------|---|
| Library Name<br>(Field 1)     | Name of the library that logged the session on.                             |
| Process ID<br>(Field 2)       | Unique identifier for the application process running on the client system. |
| Application Name<br>(Field 3) | Name of the application running on the client system.                       |
| Node Name<br>(Field 4)        | Number of the node on which the application logged on.                      |

## Example: PDE Internal Session

```
108E 153.64.137.84 LNXMPP 35
```

Field contents:

| Field     | Contents   | Description   |
|-----------|--|---|
| -----     | -----  | -----   |
| 1         | 108E   |   |
| Number of | the library (LIBDBSUTIL) that logged this session on |   |
| 2         | 153.64.137.84  | Unique ID for the application process running on the client system. |
| 3         | LNXMPP   | Name of the application running on the client system.               |
| 4         | 35   | Number of the node on which the application logged on.              |

## Data Types for Unicode Views

The data types for the compatibility and Unicode views are the same for all versions of the LogonSource column.

# Database Objects

The following provides details about each database object stored in the system table.

## Databases

The database objects in Vantage contain:

- Database name, creator name, owner name, and account name
- Space allocation (if any) including:
  - Permanent
  - Spool
  - Temporary
- Number of fallback tables
- Collation type
- Creation timestamp
- Date and time the database was last altered and the name that altered it
- Role and profile names
- Revision numbers for the UDF library and any XSP libraries by Application Category

## Corresponding Privileges

The privileges relating to databases are:

- CREATE DATABASE
- DROP DATABASE

## External Procedures

External procedures are defined and stored as database objects. The source code and the object code are stored in the database of the user space.

External procedure objects contain:

- C/C++ source and object code for the external procedure if its language is not Java.
- External procedure name
- External name
- Data types of the parameters
- Source file language
- Data accessing characteristic
- Parameter passing convention

- Execution protection mode
- Character type
- Platform type

## Corresponding Tables and Privileges

External procedures information is stored in:

- DBC.TVM (stores attributes)
- DBC.TVFields
- DBC.AccLogRuleTbl
- DBC.DBase
- DBC.AccessRights

The privileges relating to external procedures are:

- CREATE OWNER PROCEDURE
- CREATE PROCEDURE
- ALTER PROCEDURE
- DROP PROCEDURE
- EXECUTE PROCEDURE

## Related Topics

For more information on external procedures, see *Teradata Vantage™ - SQL External Routine Programming*, B035-1147.

## JARs

JAR files are created externally to the database but are defined and stored as a database object when installed on the system. The object code of the externally created JAR file is stored in the database of the user space.

The JAR objects contain:

- Java object code for the JAR
- JAR name
- External name
- Platform type
- Revision number

## Corresponding Tables and Privileges

JAR information is stored in:

- DBC.TVM (stores attributes)

- DBC.JARS (stores attributes)
- DBC.DBase
- DBC.AccessRights

The privileges relating to JARs are:

- CREATE EXTERNAL PROCEDURE
- DROP PROCEDURE
- CREATE FUNCTION
- DROP FUNCTION

## Related Topics

For more information on JARs, see *Teradata Vantage™ - SQL External Routine Programming*, B035-1147.

## Java External Procedures

Java external procedures are defined and stored as database objects.

Java external procedure objects contain:

- Java external procedure name
- External file reference
- Data types of the parameters
- Source file language
- Data accessing characteristic
- Parameter passing convention
- Execution protection mode
- Character type
- Platform type

## Corresponding Tables and Privileges

Java external procedure information is stored in:

- DBC.TVM (stores attributes)
- DBC.TVFields
- DBC.AccLogRuleTbl
- DBC.DBase
- DBC.AccessRights
- DBC.Routine\_Jar\_Usage (stores attributes)
- DBC.UDFInfo

The privileges relating to Java external procedures are:

- CREATE OWNER PROCEDURE
- CREATE EXTERNAL PROCEDURE
- ALTER PROCEDURE
- DROP PROCEDURE
- EXECUTE PROCEDURE

## Related Topics

For more information on Java external procedures, see *Teradata Vantage™ - SQL External Routine Programming*, B035-1147.

## Java UDFs

Java user-defined functions (UDFs) are defined and stored as database objects.

Java UDF objects contain:

- Function call name
- Specific name
- External name
- Data types of the parameters
- Function class
- Source file language
- Data accessing characteristic
- Parameter passing convention
- Deterministic characteristic
- Null-call characteristic
- Execution protection mode
- Character type
- Platform type

## Corresponding Tables and Privileges

Java UDF information is stored in:

- DBC.TVM (stores attributes)
- DBC.TVFields
- DBC.AccLogRuleTbl
- DBC.DBase
- DBC.AccessRights
- DBC.Routine\_Jar\_Usage (stores attributes)
- DBC.UDFInfo (stores attributes)

The privileges relating to Java UDFs are the following:

- ALTER FUNCTION
- CREATE FUNCTION
- DROP FUNCTION
- EXECUTE FUNCTION

## Related Topics

| For more information about ... | See ...  |
|--------------------------------|--|
| Java UDFs                      | <i>Teradata Vantage™ - SQL External Routine Programming</i> , B035-1147. |
| tables affected by UDFs        | <a href="#">Views Reference</a> .  |

## SQL Procedures

SQL procedures are database objects executed on Vantage. Typically, SQL procedures consist of:

- a procedure name
- input and output parameters
- procedure body

For more information see *Teradata Vantage™ - SQL Stored Procedures and Embedded SQL*, B035-1148.

For each SQL procedure, the database includes an SQL procedure table that contains the SQL procedure body you write and the corresponding compiled object code. Data dictionary tables contain procedure parameters and attributes.

The Transient Journal (TJ) record holds the host request number for a procedure, in addition to the DBS request number, to return a correct response to the query status for a request after DBS restarts.

## Corresponding Tables and Logging Rules

SQL procedure object information is stored in:

- DBC.TVM
- DBC.TVFields
- DBC.AccessRights
- DBC.AccLogRuleTbl

The SPObjCodeRows column in the DBC.TVM table references information on the status of the SQL procedure. The value of this column indicates the following SQL procedure creation-time attributes:

- Session mode
- Platform type
- Print option

- SQL procedure text storage option
- Version number
- Warning option
- Hardware architecture

The SPPParameterType column in the DBC.TVFields table contains information about the SQL procedure parameters. Parameter types for this column include IN, INOUT, or OUT.

The following table lists the privileges stored in the AccessRights table and their corresponding logging rules stored in the AccLogRuleTbl table.

| Privilege              | Logging Rule         |
|------------------------|----------------------|
| CREATE OWNER PROCEDURE | AcrCreOwnerProcedure |
| CREATE PROCEDURE       | AcrCreateProcedure   |
| ALTER PROCEDURE        | AcrAlterProcedure    |
| EXECUTE PROCEDURE      | AcrExecuteProcedure  |
| DROP PROCEDURE         | AcrDropProcedure     |

## Related Topics

| For more information about ...    | See ...  |
|-----------------------------------|--|
| tables affected by SQL procedures | <a href="#">Views Reference</a> .  |
| usage of SQL procedures           | <i>Teradata Vantage™ - SQL External Routine Programming</i> , B035-1147. |

## Tables

The table objects in the database contain:

- Location, identification, version
- Database name, table name, creator name, database name, and user names of all owners in the hierarchy
- Each column in the table, including column name, data type, length, and phrases
- User/creator privileges
- Indexes
- Constraints
- Table backup and protection (including fallback and permanent journaling status)
- Date and time the object was created

## Corresponding Privileges

The privileges relating to tables are:

- CREATE TABLE
- ALTER TABLE
- DROP TABLE
- REFERENCE
- INDEX
- DUMP
- RESTORE
- CHECKPOINT

## Triggers

The Trigger objects in the database contain:

- IDs of the:
  - Table
  - Trigger
  - Database and subject table database
  - User who created the trigger
  - User who last updated the trigger
- Timestamp for the last update
- Indexes
- Trigger name and:
  - Whether the trigger is enabled
  - The event that fires the trigger
  - The order in which triggers fire
- Default character set
- Creation text and time stamp
- Overflow text, that is, trigger text that exceeds a specified limit
- Fallback tables

## Corresponding Privileges

The privileges relating to triggers are:

- CREATE TRIGGER
- DROP TRIGGER

## Users

The user objects in the database contain:

- User name, creator name, and owner name
- Password string and password change date
- Space allocation, including:
  - Permanent
  - Spool
  - Temporary
- Default account, database, collation, character type, and date form
- Creation timestamp
- Name and time stamp of the last alteration made to the user
- Role and profile name

## Corresponding Privileges

The privileges relating to users are:

- CREATE USER
- DROP USER

## UDFs

UDFs are defined and stored as database objects. The UDF source code and object code is stored in the database of the user space.

The UDF objects contain:

- C source code and object code if its language is not Java
- Function call name
- Specific name
- External name
- Data types of the parameters
- Function class
- Source file language
- Data accessing characteristic
- Parameter passing convention
- Deterministic characteristic
- Null-call characteristic
- Execution protection mode
- Character type

- Platform type

## Corresponding Tables and Privileges

UDF information is stored in:

- DBC.TVM
- DBC.UDFInfo (stores attributes)
- DBC.TVFields
- DBC.AccLogRuleTbl
- DBC.DBase
- DBC.AccessRights

The privileges relating to UDFs are:

- ALTER FUNCTION
- CREATE FUNCTION
- DROP FUNCTION
- EXECUTE FUNCTION

## Related Topics

| For more information about ... | See ...   |
|--------------------------------|---|
| tables affected by UDFs        | <a href="#">Views Reference</a> .                                       |
| UDFs                           | <i>Teradata Vantage™ - SQL External Routine Programming, B035-1147.</i> |

## UDMs

User-defined methods (UDMs) are defined and stored as a database object. Object attributes are stored in DBC.UDFInfo, and the source code and object code are stored in the database of the user space.

The UDM objects contain:

- C source code and object code for the user defined method
- Function call name
- Name (specific and external)
- Data types of the parameters
- Function class
- Source file language
- Data accessing characteristic
- Parameter passing convention
- Deterministic characteristic
- Null-call characteristic

- Execution protection mode
- Character type
- Platform type

## Corresponding Privileges

The privilege relating to UDMs is:

- UDTMETHOD

## UDTs

The user-defined type (UDT) objects in the database contain:

- DBC.UDTInfo - one entry per UDT
  - Type name
  - Type kind (Distinct or Structured)
  - Whether the type is instantiable
  - Default transform group (name)
  - Ordering form (full ordering or equals only - distinct and structured types are always full)
  - Ordering category (map or relative - distinct and structured types are always map)
  - Ordering routine ID
  - Cast count
- DBC.UDTCast - one entry per cast for a UDT
  - Whether cast is implicit assignment
  - Cast routine ID
- DBC.UDFInfo - one entry for the auto-generated default constructor of the UDT; entries are the same as for a regular (C/C++) UDF
- DBC.UDTTransform - one entry for the UDT transform group and routine identifiers
  - Default transform group name
  - ToSQL routine ID
  - FromSQL routine ID

## Corresponding Privileges

The privileges relating to UDTs are:

- UDTUSAGE
- UDTTYPE

## Views or Macros

The view or macro objects in the database contain:

- View or macro text
- Creation time attributes
- User and creator privileges

## Corresponding Privileges

The privileges relating to views are:

- CREATE VIEW
- DROP VIEW

The privileges relating to macros are:

- CREATE MACRO
- DROP MACRO
- EXECUTE

## Additional Information

### Teradata Links

| Link  | Description  |
|---|--|
| <a href="https://docs.teradata.com/">https://docs.teradata.com/</a>                                     | Search Teradata Documentation, customize content to your needs, and download PDFs.<br>Customers: Log in to access Orange Books.  |
| <a href="https://support.teradata.com">https://support.teradata.com</a>                                 | One-stop source for Teradata community support, software downloads, and product information.<br>Log in for customer access to: <ul style="list-style-type: none"><li>• Community support</li><li>• Software updates</li><li>• Knowledge articles</li></ul> |
| <a href="https://www.teradata.com/University/Overview">https://www.teradata.com/University/Overview</a> | Teradata education network   |
| <a href="https://support.teradata.com/community">https://support.teradata.com/community</a>             | Link to Teradata community   |