

Teradata Vantage™ - Resource Usage Macros and Tables

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.

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

Chapter 1: Introduction to Resource Usage Macros and Tables	6
Changes and Additions	6
Chapter 2: Resource Usage Data	7
Benefits of Resource Usage Data	7
Overview of Resource Usage Data	7
Data Reporting	8
Resource Usage Macros	8
Application Programming Interfaces	9
Chapter 3: Planning Your Resource Usage Data	10
Resource Usage Table Settings	10
Types of Resource Usage Tables	10
Logging Rate	12
Summary Mode	13
Active Row Filter Mode	14
Resource Usage Logging	14
Chapter 4: Resource Usage and Procedures	16
Enabling RSS Logging	16
General Macro Input Format	17
Macro Execution	19
Using ENABLE and DISABLE LOGON Commands	24
Purging Data	24
Chapter 5: Resource Usage Tables	26
Physical Table Naming Conventions	26
Relational Primary Index	26
Resource Usage Table Rows	27
Occasional Event Data	27
Types of Resource Usage Table Columns	27
About the Mode Column	30
Summary Mode in Resource Usage Tables	31
Chapter 6: ResUsageScpu Table	32
Housekeeping Columns	32
Statistics Columns	34
Reserved Columns	35
Spare Columns	35

Summary Mode	35
Chapter 7: ResUsageSpma Table	36
Housekeeping Columns	36
Statistics Columns	39
Reserved Columns	62
Spare Columns	62
Chapter 8: ResUsageSawt Table	64
Housekeeping Columns	64
Statistics Columns	66
Reserved Columns	70
Spare Columns	70
Summary Mode	71
Chapter 9: ResUsageShst Table	72
Housekeeping Columns	72
Statistics Columns	75
Reserved Columns	76
Spare Columns	77
Summary Mode	77
Chapter 10: ResUsageSldv Table	78
Housekeeping Columns	78
Statistics Columns	80
Reserved Columns	83
Spare Columns	83
Summary Mode	83
Chapter 11: ResUsageSmhm Table	85
Chapter 12: ResUsageSpdsk Table	86
Housekeeping Columns	86
Statistics Columns	88
Reserved Columns	93
Spare Columns	94
Summary Mode	94
Chapter 13: ResUsageSps Table	96
Housekeeping Columns	96
Statistics Columns	99
Reserved Columns	117
Spare Columns	117
Chapter 14: ResUsageSvdsk Table	119

Housekeeping Columns	119
Statistics Columns	121
Reserved Columns	124
Spare Columns	124
Summary Mode	125
Chapter 15: ResUsageSvpr Table	126
Housekeeping Columns	126
Statistics Columns	128
Reserved Columns	163
Spare Columns	163
Summary Mode	165
Chapter 16: Resource Usage Views	166
Chapter 17: Resource Usage Macros	167
Macro Output Format	167
ResAWT Macros	168
ResCPUByAMP Macros	171
ResCPUByPE Macros	173
ResCPUByNode Macros	175
ResHostByLink Macros	177
ResLdvByNode Macros	179
ResMemMgmtByNode Macros	180
ResNetByNode Macros	182
ResNode Macros	184
ResPdskByNode Macros	188
ResPs Macros	189
ResVdskByNode Macros	191
Appendix A: Notation Conventions	193
Appendix B: ResUsagelpma Table	196
Appendix C: ResUsagelvpr Table	206
Appendix D: Partition Assignments	218
Appendix E: Additional Information	221

Introduction to Resource Usage Macros and Tables

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.

These Teradata macros can be used to report resource usage data and, when enabled, the tables can be used to:

- Observe the balance of disk and CPU usage
- Obtain pdisk usage information
- Check for bottlenecks
- Provide an overall history of the system operation
- Monitor the utilization of AMP Worker Tasks (AWTs)

Related Documentation

For general information:

- Collecting RSS data, see *Teradata Vantage™ - Application Programming Reference*, B035-1090.
- Changes in the resource tables, views, or macros between database releases, see *Resource Porting Usage Guide*, B035-1183.

For information on workload management (WM) capacity on demand (COD) and setting limits on resource usage:

- *Teradata® Integrated Workload Management Teradata SLES 11 Orange Book*, TDN0001728
- *Teradata® Viewpoint User Guide*, B035-2206

Changes and Additions

Date	Description
July 2021	<ul style="list-style-type: none"> • Added spare columns for this release. • Updated the descriptions for IOCompleted and IOCompletedKB. • Added new Native Object Store columns to ResUsageSpma Table and ResUsageSvpr Table: <ul style="list-style-type: none"> ◦ NosFilesWritten ◦ NosPhysWriteIOs ◦ NosPhysWriteIOKB • Added LDSK device to LdvKind and PdiskType column descriptions.
June 2020	Added new permanent columns and conversions from spare columns to permanent columns for this release.

Resource Usage Data

This document explains the resource usage data and settings for a variety of installation configurations and environments in [Planning Your Resource Usage Data](#). To implement the settings you decide on, see [Resource Usage and Procedures](#).

The only maintenance required is to purge old data regularly. See [Purging Data](#).

For additional information on performance analysis and system tuning, see the following:

- *Teradata Vantage™ - Application Programming Reference*, B035-1090
- *Teradata Vantage™ - Database Administration*, B035-1093

For information on changes between releases, see *Resource Porting Usage Guide*, B035-1183.

Benefits of Resource Usage Data

Resource usage data is useful for the following purposes:

- Measuring system performance
- Measuring component performance
- Assisting with on-site job scheduling
- Identifying potential performance impacts
- Planning installation, upgrade, and migration
- Analyzing performance degradation and improvement
- Identifying problems such as bottlenecks, parallel inefficiencies, down components, and congestion

Overview of Resource Usage Data

Resource usage data is stored in system tables and views in the DBC database. Teradata macros generate reports that display the data.

To load the resource usage views and macros, you can run the Database Initialization Program (dip) script, DIPRUM. For more information on DIPRUM, see *Teradata Vantage™ - Database Utilities*, B035-1102.

Note:

You can run DIP scripts one at a time or all at once by running DIPALL. DIP scripts, such as DIPRUM and DIPSYFNC, can be executed in any order.

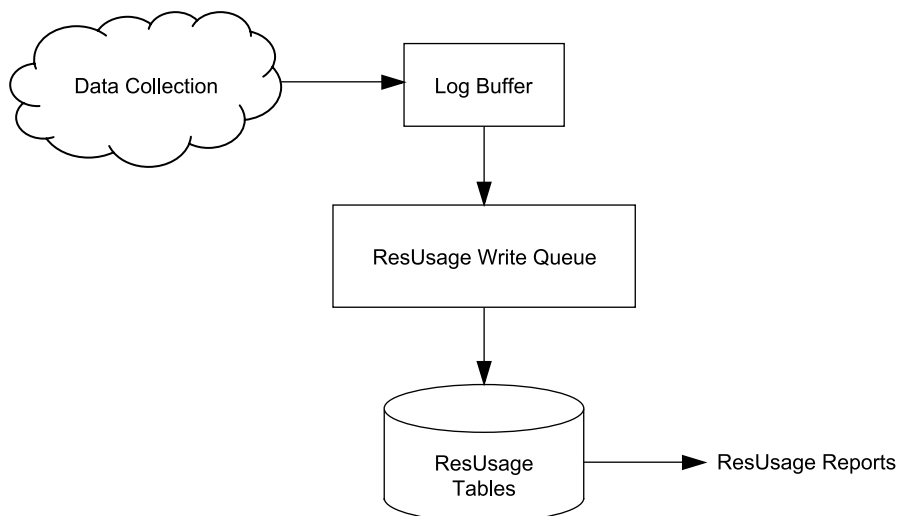
As with other database data, you can access resource usage data using SQL if you have the proper privileges. You can also write your own queries or macros on resource usage data.

The following table lists the topics covered by resource usage data.

Resource usage data covers ...	Which includes ...
BYNET traffic on a node	point-to-point messaging, broadcast messaging, and merge activities.
client-to-server traffic	data for each communication link.
CPU utilization	overhead, user service, and time of session execution.
storage device traffic	the number of reads/writes and amount of data transferred as seen from the storage driver.
pdisk device traffic	pdisk I/O, cylinder allocation, and migration statistics.
vdisk device traffic	all the cylinders allocated by an AMP (which can come from any pdisks in the clique).
Priority Scheduler information	<ul style="list-style-type: none"> Data by Priority Scheduler WD. Priority Scheduler is managed by TASM and is configured using the Teradata Viewpoint workload management portlets. For more information on those portlets, see <i>Teradata® Viewpoint User Guide</i> , B035-2206.
AMP Worker Task information	AMP Worker Task statistics.
memory management activity	memory allocation.

Data Reporting

Data is reported at the logging period. When a new logging period starts, the data is gathered in the Gather Buffer, then updated to the Log Buffer and logged to the database resource usage tables.



Resource Usage Macros

Resource usage macros produce reports from data logged to the resource usage tables. They can generate reports for a selected period of time and nodes.

You can use the reports to analyze key operational statistics and evaluate the performance of your system.

Like other macros, resource usage macros consist of one or more Teradata SQL statements stored in Vantage and executed by a single EXECUTE statement.

See [Resource Usage and Procedures](#) for more information on the resource usage macros, and *Teradata Vantage™ - SQL Stored Procedures and Embedded SQL*, B035-1148 for details about how to use the EXECUTE statement.

Application Programming Interfaces

You can use the Teradata application programming interfaces (APIs) to:

- Set the resource monitoring and logging rates
- Collect RSS data and return node, vproc, and WD oriented data

Examples of these APIs, include the System PMPC SET RESOURCE RATE, MONITOR PHYSICAL RESOURCE, MONITOR VIRTUAL RESOURCE, and MONITOR WD requests.

For more information on these APIs, see *Teradata Vantage™ - Application Programming Reference*, B035-1090.

Planning Your Resource Usage Data

Resource Usage Table Settings

The default resource usage settings provide a good starting point for system monitoring. The default results in the ResUsageSpma (SPMA) table being logged every 10 minutes (600 seconds).

The ResUsageSpma table provides a high level summary of how the system is operating and contains summarized or key elements from most of the other tables. If you want to record detailed statistics covered by any of the resource usage tables, you should enable them for logging, along with specifying the largest logging period that will meet your needs. You should not log data that you do not have a planned need for since this does incur additional database system overhead and uses up additional database space.

The more tables you enable for logging and the shorter the logging period used, the more overhead the system will use.

Tables Based on Needed Reports

If you plan on using the report macros provided in [Resource Usage Macros](#) you need to enable the associated table.

Related Information

For...	See...
instructions on setting resource usage tables	Enabling RSS Logging .
instructions on using macros	General Macro Input Format and Macro Execution .
descriptions and examples of the macros	Resource Usage Macros .

Types of Resource Usage Tables

The following table describes the tables and provides guidance about which ones to enable.

Note:

The shaded rows in the table below are the resource usage tables that are generally not used at customer sites.

Table Name	Covers	When You Should Enable
ResUsagelpma	System-wide node information, intended primarily for Teradata engineers.	---
ResUsagelvpr	System-wide virtual processor information, intended primarily for Teradata engineers.	---
ResUsageSawt	Data specific to the AMP Worker Tasks.	When you want to monitor the utilization of the AMP Worker Task and determine if work is backing up because the AMP Worker Tasks are all being used.
ResUsageScpu	Statistics on the CPUs within the nodes.	<p>When the performance analysis suggests that the overall performance is limited or to check if a program is spinning in an infinite loop on an individual processor.</p> <p>For example, saturation of a particular CPU on each node or on a particular node while others are idle could indicate a task always uses that CPU.</p> <p>Also, you should enable when the system is first brought online to verify the following:</p> <ul style="list-style-type: none"> • That all CPUs are functioning on all nodes • There is a good load balance among the CPUs
ResUsageShst	Statistics on the host channels and LANs that communicate with Vantage.	To determine details about the traffic over the IBM Host channels and if there is a bottleneck.
ResUsageSldv	System-wide, logical device statistics collected from the storage driver.	<p>To observe the balance of disk usage. The storage device statistics are often difficult to interpret with disk arrays attached due to multi-path access to disks.</p> <p>Note:</p> <p>Use the ResUsageSvdsk table first to observe general system disk utilization unless specifically debugging at a low level.</p>
ResUsageSmhm	Reserved for future use.	N/A
ResUsageSpdsk	Statistics collected from the pdisk device.	To obtain detailed usage information about pdisks.
ResUsageSpma	<p>System-wide node information provides a summary of overall system utilization incorporating the essential information from most of the other tables.</p> <p>Use the columns in ResUsageSpma to view BYNET utilization.</p>	To provide an overall history of the system operation.

Table Name	Covers	When You Should Enable
	Note: The BYNET can transmit and receive at the same time, resulting in 100% transmitting and 100% receiving values simultaneously. Another method of determining BYNET utilization and traffic is to use the blmstat tool.	
ResUsageSps	Statistics based on the WD the work is being performed for.	When you need to track utilization by the WD level.
ResUsageSvpr	Data specific to each virtual processor and its file system.	To view details about the resources being used by each vproc on the system. This table is useful for looking for hot AMPS or PEs that may be CPU bound or throttled on other resources.

Logging Rate

The logging rate controls the frequency (number of seconds) at which resource usage data is logged to the resource usage tables.

When you have decided what rate to set, see [Resource Usage and Procedures](#) for details on how to set the logging rate.

Logging Period

Resource usage logging means the writing of resource data as rows to one or more of the resource usage database tables. The tables are named DBC.ResUsagexxxx, where xxxx is the name of the resource usage table (for example, SPMA, IPMA, and so on) as listed in [Types of Resource Usage Tables](#).

The shorter the logging period, the more frequently data is logged, and the more disk space is used.

When the system is so busy that the resource usage table logging gets backed up, RSS will automatically double the logging period which effectively summarizes the data by providing values for a time period twice that provided by the previous logging period.

If you see the resource usage logging rates change without user intervention, this means that the database is busy. When no longer busy, the system resumes logging as before.

Note:

Events in the event logs related to this doubling of the logging period do not represent fatal errors but are informational to indicate that the automatic operations of the RSS are attempting to maintain data logging.

Rule

The following rule on the logging rate is imposed by the system.

Intervals must evenly divide into 3600 (the number of seconds in an hour). The following table shows the valid logging rate.

- The white area of the table shows rates recommended only for short-term use for debugging a specific issue.
- The highlighted area of the table shows rates recommended for production processing.

1	2	3	4	5	6
8	9	10	12	15	16
18	20	24	25	30	36
40	45	48	50	60	72
75	80	90	100	120	144
150	180	200	225	240	300
360	400	450	600	720	900
1200	1800	3600			

A practical minimum log interval during production processing is 60 seconds. Intermediate log intervals, such as 120 seconds or 300 seconds can also be used. The default rate is 600 seconds.

Rates and enabled tables may be changed at any time and the changes take effect immediately.

Summary Mode

You can use Summary Mode to reduce the system overhead from logging tables that produce multiple rows per logging period. Summary Mode helps reduce overhead by combining data from multiple rows into one or more summary rows based on specific criteria for each table. For example, if you want to log information provided in the ResUsageSvpr table but do not need data for each individual vproc, use Summary Mode to produce one row per vproc type instead of one row per vproc.

The ResUsageSpma table, in comparison, provides node level summary of key fields from most of the other resource usage tables. When more details are required than the ResUsageSpma table provides, the next level of information is provided by using Summary Mode logging for the table of interest. This helps minimize the cost of the data logging.

You can select Summary Mode for each table individually. For details on how Summary Mode affects that particular table, see the description for each table.

For example, for the ResUsageSvpr table in Summary Mode, all the individual vproc rows of the same vproc type are combined into a single row. Since the data values are added together, you need to divide the summary row data value by the number of rows that made up the Summary Mode row to get the average

per vproc. For example, divide the AMP summary row data value by the number of AMPs on that node to determine the average value per AMP. A similar computation needs to be done to derive the average value per PE from the summary row data value.

Note:

To determine the number of AMP, PEs, and all other vproc types on your system, you can use the ResUsageSpma table or use the Vproc Manager utility.

Resource usage columns that represent a maximum statistic are not summed together. Instead the maximum value from the rows is used. For example, the ResUsageSvpr table MsgWorkQLenMax column in the Summary Mode row for the AMPs will contain the maximum value from all the AMP rows that would have been logged in non-Summary Mode. The columns that represent a minimum statistic are summarized by storing the minimum value from all the constituent rows.

Summary Mode has either no effect on the values of the Housekeeping Columns or it is specifically detailed in the description of each affected field.

To enable Summary Mode, see [Enabling RSS Logging](#).

For more information on Summary Mode, see [Summary Mode in Resource Usage Tables](#).

Active Row Filter Mode

Active Row Filter Mode reduces the overhead of logging for some of the resource usage table by limiting the data rows that are logged to the database.

When you enable active row filtering, it may appear that rows are missing when looking at the query results. This is because the index values of the inactive rows varies over time so that a row with one index may be logged one period but not in another. To determine if rows are not being logged to the database, you should look in the event logs for messages indicating that rows have been lost.

Active Row Filtering should not be disabled for the ResUsageSps table.

Resource Usage Logging

The Cost of Logging

Logging resource usage data to database tables incurs costs:

- Writing to the database adds to the system I/O load. On a heavily loaded system, this could affect the production workload throughput.
- The rows written to the database take up space. If this space is never reclaimed, it will eventually grow to consume all available space in user DBC.
- In an extremely loaded system, it is possible that the RSS can fall behind in writing data to the database. Although it caches such data and eventually catches up if given a chance, the RSS is forced to start discarding rows if the system load persists and its cache capacity is exceeded.

Logging Cost Contributors

Logging costs are difficult to quantify. They depend on a number of interrelated factors:

- How busy is the system
- Which resource usage tables are enabled
- What resource usage logging rates are in effect
- The system configuration (vproc, CPU, host driver, logical devices or device controllers)

Operational Methods

To optimize performance and reduce the cost of resource usage logging on your system, Teradata recommends that you:

1. Use Summary Mode to reduce the number of rows inserted into the resource usage tables if Summary Mode data provides sufficient information for your needs.
2. Do not disable Active Row Filter Mode for the tables that it is by default enabled for (such as, the ResUsageSps table). Active Row Filter Mode limits the number of rows written to the database each logging period and minimizes the amount of system resources used.
3. Avoid unnecessarily using or exhausting available disk space by doing the following:
 - Never enable logging on tables that you do not intend to use.
For example, logging only to the ResUsageSpma table provides a lot of useful information with a minimal operational load on the system.
 - Use the largest rates that provide enough detail information for your purposes.
Generally, you should use a logging rate no smaller than 60. The default rate is 600.
These values can be adjusted any time, regardless of whether the database system is busy. New values take effect as soon as the adjustment command is issued. (For example, with ctl, when you issue the WRITE command.)
4. Purge old data from the resource usage tables periodically.

Related Information

For instructions on...	See...
enabling resource usage tables, setting the logging rates, and summarizing or filtering rows	Enabling RSS Logging.
purging old data from resource usage tables	Purging Data.

Resource Usage and Procedures

Enabling RSS Logging

You can enable the resource usage tables using the Control GDO Editor (ctl) utility or the database commands in Database Window (DBW).

Before you enable the resource usage tables, determine which tables apply to the resource usage macros you want to run. For more information, see:

- [Resource Usage Table Settings](#).
- [Logging Rate](#).

ctl

You can set various Vantage configuration settings using ctl. The > sc rss screen in ctl allows you to specify the rates of resource usage data logging. For detailed information on starting ctl and modifying the settings, see ctl in *Teradata Vantage™ - Database Utilities*, B035-1102.

DBW

Note:

For instructions on starting the Database Window, see *Teradata Vantage™ - Database Utilities*, B035-1102.

To enable RSS logging from DBW:

1. Open the **Supvr** window.
2. Set the Node Logging Rate using the database command below.

```
SET RESOURCE { LOGGING | LOG } number
```

where *number* is the number of seconds.

Note:

A rate of zero disables the logging function.

3. Specify the table you want to enable logging to using the database command below.

```
SET LOGTABLE { tablename | ALL } { ON | OFF }
```


where *tablename* is the suffix part of ResUsage Xxxx. For example, for the DBC.ResUsageSpma table, the *tablename* would be “Spma.”

After the table is enabled for logging, you can log rows in Summary Mode. For more information, see [Summary Mode](#).

Note:

To log rows in Summary Mode, you must enable the table specified in both the RSS Table Logging Enable group and in the RSS Summary Mode Enable group.

4. (Optional) Enable Summary Mode on the table specified using the command below.

```
SET SUMLOGTABLE tablename { ON | OFF }
```

Example: Enable Table Logging and Set the Logging Rate Using DBW

The following example shows you how to enable table logging and set the Logging rate using the database commands in DBW. For example, to enable the ResUsageShst table and set the logging rate for 10 minutes (600 seconds), enter the following:

```
set logtable shst on
set resource log 600
```

General Macro Input Format

As shown in the table below, there are four kinds of macros:

- Multiple-node
- One-node
- All-node
- ByGroup

For any given line in the following table, the macros on that line report the same statistics for either multiple nodes, one node, all nodes, or group nodes as indicated.

Description	Multinode Macro	One-Node Macro	All-Node Macro	ByGroup Macro
AWTs in use by node	ResAWTByAMP ResAWTByNode		ResAWT	
CPU usage by AMP Vprocs	ResCPUByAMP	ResCPUByAMPOneNode		ResAmpCpuByGroup
CPU usage by PE Vprocs	ResCPUByPE	ResCPUByPEOneNode		ResPeCpuByGroup
CPU usage by nodes	ResCPUByNode	ResCPUOneNode		ResCpuByGroup

Description	Multinode Macro	One-Node Macro	All-Node Macro	ByGroup Macro
Host statistics		ResHostOneNode	ResHostByLink	ResHostByGroup
Ldv disk statistics	ResLdvByNode	ResLdvOneNode		ResLdvByGroup
Memory management	ResMemMgmtByNode	ResMemMgmtOneNode		ResMemByGroup
General network statistics	ResNetByNode	ResNetOneNode		ResNetByGroup
General node-level statistics	ResNodeByNode	ResOneNode	ResNode	ResNodeByGroup
pdisk level I/O statistics	ResPdskByNode	ResPdskOneNode		ResPdskByGroup
Priority Scheduler and TASM Workload statistics	ResPsByNode			ResPsByGroup
AMP level I/O statistics	ResVdskByNode	ResVdskOneNode		ResVdskByGroup

Parameter Use for One-Node, Multiple-Node, All-Node, and Group Macros

The following table explains parameter use for one-node, multiple-node, all-node, and group macros.

Macro Type	Number of Parameters	Node Parameters Used
Multiple node	Six	<i>FromNode, ToNode</i>
One node	Five	Node
All node	Four	None; this macro reports system-wide statistics.
Group	Four	None; this macro reports statistics for all nodes in the group.

For instructions on using these macros, see [Macro Execution](#).

Input Format for One-Node Macros

One-node macro versions are primarily used on single-node systems. Alternatively, you can use the corresponding multiple-node macro to report on *just one node* by supplying equal *FromNode* and *ToNode* parameters. One-node versions are recommended, however, because they eliminate redundant report columns on a single-node system. Examples of redundant columns are the *NodeID* column and columns that focus on cross-node load balancing.

OneNode macros have the same general input format as the other macros. The only differences are that the single-node version of each macro has both of the following:

- *OneNode* qualifier in the macro name.
- A single *node* specification, instead of the *FromNode* and *ToNode* parameters to specify a range of nodes. The default is '001-01'.

Input Format for ByGroup Macros

ByGroup macro versions are used on systems with co-existing nodes. Co-existing nodes are nodes of different model types in the same configurations. Because of the differences, the nodes may become bottlenecks in the throughput of the system as a whole. Therefore, ByGroup macros were developed to provide the system user with a summary of the performance data based on node groupings.

Note:

The Database Administrator must identify the groupings of nodes when the system is first configured.

ByGroup macros are similar to the other macros. The only difference is that they use the GroupId column of the views to report system usage for a specific set of nodes grouped by a GroupId. The input format of the ByGroup macros is the same as the other macros except ByGroup appears as the qualifier in the macro name.

For an example of a ByGroup macro, see [ResCPUByAMP Macros Output](#).

Retaining Data From Prior Releases

If you expect an ongoing need to retain and analyze data from prior releases, ask your System Administrator to retain two sets of view and macro Data Definition Language (DDL) files in separate places. Rename the views and macros so that you can use either.

You could, for example, use ResNodeR12 as the name of the resource usage macro from an older release, and use it when you want to analyze the data from that release.

Macro Execution

EXECUTE MACRO

The EXECUTE MACRO command executes a macro. For details about each macro and its resulting report, see [Resource Usage Macros](#).

EXECUTE MACRO Syntax

```
{ EXECUTE | EXEC } {  
  
  MacroNameMultiNode (  
    [ fromDate ] , [ toDate ] ,  
    [ fromTime ] , [ toTime ] ,  
    [ fromNode ] , [ toNode ]  
  ) |
```

```

MacroNameAllNode (
  [ fromDate ] , [ toDate ] ,
  [ fromTime ] , [ toTime ]
) |

MacroNameOneNode (
  [ fromDate ] , [ toDate ] ,
  [ fromTime ] , [ toTime ] ,
  [ Node ]
) |

MacroNameByGroup (
  [ fromDate ] , [ toDate ] ,
  [ fromTime ] , [ toTime ]
)
}

```

EXECUTE MACRO Syntax Elements

MacroNameMultiNode

Name of a multinode resource usage macro:

- ResAwtByNode
- ResCPUByAMP
- ResCPUByPE
- ResCPUByNode
- ResLdvByNode
- ResMemMgmtByNode
- ResNetByNode
- ResNodeByNode
- ResPdiskByNode
- ResPsByNode
- ResVdiskByNode

FromDate

Start date to report resource usage data.

The date may be entered either as a character string (for example, character format for May 31, 2007 would appear as '2007-05-31') or as a numeric value (for the same date in

numeric format, 1070531). The character string is the recommended format. The default is the current system date.

For more information on using numeric dates with macros, see *Teradata Vantage™ - Data Types and Literals*, B035-1143.

Note:

The character string date format is changed from *yymmdd* to *'yyyy-mm-dd'* to accommodate dates in the 21st century.

ToDate

End date to report resource usage data.

See the *FromDate* syntax element column for a further explanation of date formats.

The character string is the recommended format.

FromTime

Start time to report resource usage data. The format is *hhmmss*. The default is 000000.

ToTime

End time to report resource usage data. The format is *hhmmss*. The default is 999999.

FromNode

Starting range of nodes to report resource usage data. The format is *'nnn-nn'*. A hyphen must be included in the fourth character position. The default is '000-00'.

Note:

To identify the node ID numbers for your system, type `get config` in the DBW Supervisor Window (Supvr).

ToNode

Ending range of nodes to report resource usage data. The format is *'nnn-nn'*. A hyphen must be included in the fourth character position. The default is '999-99'.

Note:

To identify the node ID numbers for your system, type `get config` in the DBW Supervisor Window (Supvr).

MacroNameAllNode

Name of an all-node resource usage macro:

- ResNode
- ResHostByLink

The ResHostByLink and ResNode macros do not use the *FromNode* and *ToNode* parameters.

MacroNameOneNode

Single-node ID to report resource usage data. The format is '*nnn-nn*', and hyphen must be included in the forth character position. For example, 1-01 should be typed out as '001-01'. The default is '001-01'.

MacroNameByGroup

Name of a ByGroup resource usage macro:

- ResAmpCpuByGroup
- ResCPUByGroup
- ResHostByGroup
- ResLdvByGroup
- ResMemByGroup
- ResNetByGroup
- ResNodeByGroup
- ResPeCpuByGroup
- ResPdskByGroup
- ResPsByGroup
- ResVdskByGroup

EXECUTE MACRO Examples**Example: Using the ResCPUByAMP Macro**

The following statement executes the ResCPUByAMP macro, producing a report for the period beginning 8:00 a.m. on December 25, 2006 and ending 12:00 midnight, on December 31, 2006. It includes data for nodes 123-02 through 125-04.

```
EXECUTE ResCPUByAmp( '2006-12-25', '2006-12-31',
080000, 240000, '123-02', '125-04');
```

where:

Statement Element	Description
ResCPUByAMP	Name of the resource usage macro
'2006-12-25'	Starting date of December 25, 2006
'2006-12-31'	Ending date of December 31, 2006
080000	Starting time of 8:00 a.m.
240000	Ending time of 12:00 midnight
'123-02'	Starting node of a range of nodes
'125-04'	Ending node of a range of nodes

For information on using numeric values for dates, see *Teradata Vantage™ - Data Types and Literals*, B035-1143.

Example: Using the ResCPUByAMPOneNode Macro

The following statement executes the OneNode version of the ResCPUByAMP macro shown in Example 1. It uses the same starting and ending dates and times (using character string format), except the report is for a single node, node 123-02.

```
EXECUTE ResCpuByAmpOneNode ('2006-12-25','2006-12-31',080000, 240000,'123-02');
```

where:

Statement Element	Description
ResCPUByAMPOneNode	Name of the resource usage macro
'2006-12-25'	Starting date of December 25, 2006
'2006-12-31'	Ending date of December 31, 2006
080000	Starting time of 8:00 a.m.
240000	Ending time of 12:00 midnight
'123-02'	Node

For information on using numeric values for dates, see *Teradata Vantage™ - Data Types and Literals*, B035-1143.

Example: Using the ResAMPCpuByGroup Macro

The following statement executes the ByGroup version of the ResCPUByAmp macro shown in Example 1. It uses the same starting and ending dates and times (using character string format), except the report is for a node grouping.

```
EXECUTE ResAMPCpuByGroup ('2006-12-25','2006-12-31',
080000, 240000);
```

where:

Statement Element	Description
ResAMPCpuByGroup	Name of the resource usage macro
'2006-12-25'	Starting date of December 25, 2006
'2006-12-31'	Ending date of December 31, 2006
080000	Starting time of 8:00 a.m.
240000	Ending time of 12:00 midnight

For information on using numeric values for dates, see *Teradata Vantage™ - Data Types and Literals*, B035-1143.

Using ENABLE and DISABLE LOGON Commands

The DISABLE LOGONS command prevents new sessions from logging on. When logons are disabled, resource usage data stops logging to the tables even if there are still active sessions logged on. (DISABLE ALL LOGONS prevents all users, including user DBC, from logging on and also stops logging to the tables.)

To enable logons from Database Window, run ENABLE LOGONS or ENABLE ALL LOGONS.

To enable logons from Teradata Command Prompt, use the Start With Logons field of the Screen Debug menu of ctl. For more information about ctl, see *Teradata Vantage™ - Database Utilities*, B035-1102.

For more information about enabling and disabling logons, see *Teradata Vantage™ - Database Administration*, B035-1093.

Purging Data

The RSS does not automatically delete data from the resource usage tables. You need to purge data you no longer need on a regular basis.

You can directly remove old resource usage data by submitting SQL statements. For example, use the following SQL statement to remove data more than seven days old from the ResUsageSpma table:


```
DELETE FROM ResUsageSpma WHERE TheDate < CURRENT_DATE - 7;
```

For more information about the DELETE syntax, see *Teradata Vantage™ - SQL Data Manipulation Language*, B035-1146.

Resource Usage Tables

Physical Table Naming Conventions

Each physical table name follows this general naming convention:

ResUsage Information_type Table_name

Information_type

- S = system-wide information
- I = internal SQL Engine information

Table_name

- awt = AMP worker task statistics
- cpu = CPU-specific information
- hst = mainframe and workstation host information
- ldv = logical device statistics
- mhm = reserved for future use
- pma = node information
- pdsk = pdisk device statistics
- ps = WD resolution statistics
- vdsk = vdisk device statistics
- vpr = vproc information

Relational Primary Index

All resource usage tables have the same nonunique primary index of TheDate, TheTime, and NodeID columns.

The primary index is nonunique because:

- Of rows that will appear with the same timestamp during daylight savings time.
- Some tables have multiple rows per logging period.
- Rows that have duplicate timestamps can be distinguished by the GmtTime column.

Because the primary index is nonunique, all resource usage tables are created as MULTiset tables. This prevents the system from checking for duplicate rows.

For more information on MULTiset tables, see the information about the CREATE TABLE (Table Kind Clause) statement in *Teradata Vantage™ - SQL Data Definition Language Syntax and*

Examples, B035-1144 or the information about duplicate rows in tables in *Teradata Vantage™ - SQL Fundamentals*, B035-1141.

Resource Usage Table Rows

For information on how rows will be inserted into these tables based on the current resource usage control settings, see [Planning Your Resource Usage Data](#). For information on the number of rows inserted in a resource usage table for each applicable log period, refer to [Summary Mode](#).

Occasional Event Data

Occasional event data is considered outside the scope of resource usage and is, therefore, logged in the ERRORLOG and the DBCINFO tables rather than in the resource usage tables.

Types of Resource Usage Table Columns

This document describes what each resource usage table column reports (that is, what each DBC.ResUsage *Xxxx.ColumnName* reports) in a table format.

Note:

The actual table definitions are obtainable by executing the SHOW TABLE statement. For more information about SHOW TABLE, see *Teradata Vantage™ - SQL Data Definition Language Syntax and Examples*, B035-1144.

For each resource usage table column, this document describes the:

- Column Name
- Mode

For descriptions of the different modes of data reporting, see [About the Mode Column](#).

- Description
- Data Type

For descriptions of the data types described in this document, see *Teradata Vantage™ - Data Types and Literals*, B035-1143.

The columns are grouped into either housekeeping columns or statistics columns. Statistic columns are further grouped by category and subcategory as shown in the following table.

Column Name	Mode	Description	Data Type
HOUSEKEEPING OR STATISTICS COLUMNS			
CATEGORY			
Subcategory			

Each table has:

- Housekeeping columns which contain statistics on timestamp and current logging characteristics.
- Statistics columns which can be further categorized into subcategories. Categories and subcategories may vary from table to table.

The following table shows the types of statistics subdivided into their respective subcategories.

Category	Subcategories	Description
File System	<ul style="list-style-type: none"> • AutoCylPack • Block-level compression (BLC) • Cylinder Defragmentation Overhead • Cylinder Management Overhead Events • Cylinder MiniCylPack Overhead • Cylinder Split and Migrate Overhead • File Segment (FSG) Cache Wait • FSG I/O • Data Block Creations • Data Block Merge • Data Block Prefetches • Data Block Update Operations • Data Segment Lock Requests • Depot • Master Index (MI) • Multi-Row Requests • Segments Acquired • Segments Released • Single-Row Requests • Synchronized Full Table Scans • Transient Journal Requests • Transient Journal Overhead • Miscellaneous Write Ahead Logging (WAL) 	Some of the columns can be viewed as a subset of memory columns by expanding on the operations performed on disk memory segments. Operations counted are logical memory and physical disk reads and writes (including aging) and locking control activities. Other columns identify the purpose of operations being performed on disk segments such as cylinder migration or data updates, the requests being made by database software on the file system, the number of cylinders that were selected or scanned by AutoCylPack, or the number of data blocks that have been compressed or uncompressed. The WAL columns identify the log-based file system recovery scheme in which modifications to permanent data are written to a log file, the WAL log.
General Concurrency Control	<ul style="list-style-type: none"> • Database Locks • Monitor Management 	These columns identify concurrency control activities. These activities provided are subdivided into control done for user level processing, system overhead processing, and database locks. It does not include control specific to disk, memory or net concurrency control, which are included in the disk, memory or net columns.
Host Controller (SHST)	<ul style="list-style-type: none"> • Channel Traffic • Channel Management • Controller Overhead 	These columns identify traffic on the host-to-node channels and TCP/IP networks. Some also give

Category	Subcategories	Description
		overhead and management information on the host channel and TCP/IP network.
Memory	<ul style="list-style-type: none"> • Memory Allocations • Memory Availability Management • Memory Pages • Memory Resident • Paging • Task Context Segment Usage 	<p>These columns collect memory allocation and deallocation, logical memory and physical disk reads and writes (including paging), access, deaccess and memory control.</p> <p>Memory management columns are also provided to identify events leading up to paging and aging activities. Finally, a detailed snapshot of the memory is provided by tracking the current states per memory types.</p>
Logical Device	<ul style="list-style-type: none"> • Input and Output Traffic • Outstanding Requests • Response Time • Seek Statistics 	<p>These columns identify individual logical device activities for external storage components connected through the buses.</p> <p>The storage device statistics are calculated only on what can be derived from statistics collected by the operating system, since the disk array controllers do not provide us with any useful data for resource usage.</p>
Net	<ul style="list-style-type: none"> • Broadcast Net Traffic • Group Coordination • Message Type • Message Delivery Times • Merge Services • Net Controller Status and Miscellaneous Management • Net Circuit Management • Net Miscellaneous Contention Management • Net Queues • Network Transport Data • Point-to-point Net Traffic • Work Mailbox Queue 	<p>These columns identify traffic over the BYNET through the number and direction of messages, subdivided into the type of transmission, as well as physical utilization of the BYNET. Logical messages and direction are identified through subdivisions of the message class. Controller overhead, channel utilization, and Teradata net contention are identified as well.</p>
Process Scheduling	<ul style="list-style-type: none"> • ChnSignal Status Tracking • CPU Utilization • Cylinder Read • Interrupted CPU Switching • PE and AMP User-Defined Function (UDF) CPU • Process Allocation • Process Pending Snapshot • Process Block Counts • Process Pending Wait Time • Scheduled CPU Switching 	<p>These columns provide a CPU-level snapshot of work started with current characteristics and states. Expanded detail is provided for work started but waiting on resources. This identifies the ability or inability of the system to effectively utilize resources. Time allotments are tracked by monitoring the time spent waiting for resources or processing code. Some of these columns also track the number of times processing was switched to another process for multitasking purposes or to perform interrupt services; and others can provide information about whether the UDFs were doing work for the AMP and PE vprocs.</p>

Category	Subcategories	Description
Reserved	None	These columns are not used.
Spare	None	These columns are reserved for future use or are for internal manipulation by Teradata developers.
TASM	<ul style="list-style-type: none"> AMP Worker Task In use and Max Array Data Priority Scheduler Worktype Descriptions Monitor WD 	These columns report statistics on the AMP Worker Tasks (AWTs) and Priority Scheduler. These columns also provide RSS data to the System PMPC APIs (for example, the MONITOR WD request and MonitorWD function). For information, see <i>Teradata Vantage™ - Application Programming Reference</i> , B035-1090.
Transient Journal Management	Purge Overhead	These columns identify background overhead associated with the occasional transient journal purge operation.
User Commands	<ul style="list-style-type: none"> User command User command Arrival and Departure 	These columns describe the types of commands given to Vantage by the user and the progress of those commands.
Teradata Virtual Storage (VS)	<ul style="list-style-type: none"> Allocation I/O Migration Node Agent 	These columns identify individual pdisk and vdisk device activities.

Invalid Columns

Some of the resource usage table columns described in this document are not currently valid. These columns are shaded in gray. For example:

Column Name	Mode	Description	Data Type
SampleColumn	track	An example of a column that is not currently valid.	FLOAT

About the Mode Column

The Mode column describes the kind of data reported by the resource usage column.

IF the Mode column is ...	THEN these resource usage columns ...
count	tally the number of times an event occurred, such as the number of disk reads or writes during the logging period.
max	report the maximum value recorded during the logging period. Most of the max columns have a max suffix in the column name (for example, loRespMax).

IF the Mode column is ...	THEN these resource usage columns ...
min	report the minimum value recorded during the logging period. The min columns have a min suffix in the column name (for example, AvailableMin).
track	show the value of a countable item achieved at the end of the current logging period. An example of a countable item is a queue length.

Summary Mode in Resource Usage Tables

Summary Mode combines data from the multiple data rows normally generated into one or more rows. When multiple rows are condensed into a single row, the data is combined using the rules in the following table.

For the following Mode ...	Summary fields are combined by...
Count	<p>summing the values from all the contributing rows.</p> <p>For example, the ResUsageSawt.FlowCtlCnt field provides the total number of times that the system entered flow control state from a non-flow control state.</p> <ul style="list-style-type: none"> • In normal mode, the values are reported per AMP. • In Summary Mode, the value needs to be divided by the number of AMPs if you wish to determine the average number per AMP rather than the total.
Max	<p>taking the maximum value from all the contributing rows.</p> <p>In Summary Mode, the reported value for a Max field such as ResUsageSpdsk.ConcurrentWriteMax is the maximum value from all the rows that are combined into a single summary row.</p>
Min	<p>taking the minimum value from all the contributing rows.</p> <p>In Summary Mode, the reported value for a Min field, such as AvailableMin is the minimum value from all the rows that are combined into a single summary row.</p>
Track	<p>summing the values from all the contributing rows.</p> <p>In Summary Mode, the Track values from each row to be combined are summed together. For example, ResUsageSawt.FlowControlled is a track field so that in Summary Mode, all the AMP vproc rows are combined into a single row and the FlowControlled field will report the summed value from each of the AMP vproc rows.</p>

Summary Mode is applicable to all tables except ResUsageSpma, ResUsagelpma, and ResUsageSps. If the information for a row of a table is in Summary Mode, the SummaryFlag value is set to 'S'. If the row is being logged normally, the SummaryFlag value is set to 'N'.

ResUsageScpu Table

This resource usage table contains resource usage information specific to the CPUs within the nodes. The ResUsageScpu table includes resource usage data for available system-wide, CPU information.

Teradata recommends that you use ResScpuView to access the data rather than accessing the ResUsageScpu table directly. For more information see [Resource Usage Views](#).

Note:

This table is created as a MULTiset table. For more information see [Relational Primary Index](#).

Housekeeping Columns

Relational Primary Index Columns

These columns, taken together form the nonunique primary index.

Column Name	Mode	Description	Data Type
TheDate	n/a	Date of the log entry.	DATE
TheTime	n/a	Nominal time of the log entry. Note: Under conditions of heavy system load, entries may be logged late (typically, by no more than one or two seconds), but this column will still contain the time value when the entry <i>should have been</i> logged. See the Secs and NominalSecs columns.	FLOAT
NodeID	n/a	Node ID on which the entry resides. The Node ID is formatted as ZZZ9-9999, where ZZZ9 denotes the four-digit cabinet number and 9999 denotes the four-digit chassis number of the node. For example, a node in chassis 9 of cabinet 3 has a node ID of '3-0009'. Note: SMP nodes have a chassis and cabinet number of 1. For example, the node ID of an SMP node is '1-0001'.	INTEGER

Miscellaneous Housekeeping Columns

These columns provide statistics on current logging characteristics.

Column Name	Mode	Description	Data Type
GmtTime	n/a	Greenwich Mean Time is not affected by the Daylight Saving Time adjustments that occur twice a year.	FLOAT
CabinetID	n/a	The physical cabinet number of the node.	INTEGER
ModuleID	n/a	The physical module number of the node.	INTEGER
NodeType	n/a	Type of node, representing the per node system family type.	CHAR(8)
CPUId	n/a	Identifies the CPU within this node. The values are 0 through NCPUsConfigured -1. In Summary Mode, the value is -1.	SMALLINT
Secs	n/a	Actual number of seconds in the log period represented by this row. Normally the same as NominalSecs, but can be different in three cases: <ul style="list-style-type: none"> • The first interval after a log rate change • A sample logged late because of load on the system • System clock adjustments affect reported Secs Useful for normalizing the <i>count</i> statistics contained in this row, for example, to a per-second measurement.	SMALLINT
CentiSecs	n/a	Number of centiseconds in the logging period. This column is useful when performing data calculations with small elapsed times where the difference between centisecond-based data and whole seconds results in a percentage error.	INTEGER
NominalSecs	n/a	Specified or nominal number of seconds in the logging period.	SMALLINT
SummaryFlag	n/a	Summarization status of this row. Possible values are 'N' if the row is a non-summary row, and 'S' if the row is a summary row.	CHAR (1)
Active	max	Controls whether or not the rows will be logged to the resource usage tables if Active Row Filter Mode is enabled. If Active is set to a non-zero value, the row contains data columns. If Active is set to a zero value, none of the data columns in the row have been updated during the logging period. For example, if you enable Active Row Filter Mode, the rows that have a zero Active column value will not be logged to the resource usage tables.	FLOAT
TheTimestamp	n/a	Number of seconds since midnight, January 1, 1970. This column is useful for aligning data with the DBQL log.	BIGINT
PM_COD_CPU	n/a	Platform Metering (PM) CPU Capacity On Demand (COD) value in one tenths of percent. For example, a value of 500 represents a PM CPU COD value of 50%. The value is set to 1000 if the PM CPU COD is disabled.	SMALLINT
WM_COD_CPU	n/a	Workload management (WM) CPU Capacity On Demand (COD) value in one tenths of a percent. For example, a value of 500 represents a WM CPU COD value of 50.0%.	SMALLINT

Column Name	Mode	Description	Data Type
		The value is set to 1000 if the WM CPU COD is disabled.	

Statistics Columns

Process Scheduling CPU Utilization Columns

These columns count all CPU activities, including activities performed for virtual processors.

The CPU utilization columns are aggregates representing all CPUs on the node. CPU utilization by user code is further subdivided by the vproc tables.

- CPU idle time = CPUIdle + CPUIoWait
- CPU busy time = CPUUServ + CPUUExec

Theoretically, the values of these four columns, for any given interval, account for total CPU time on the node. That is, these columns should total to 100 * Secs * number of CPUs on the node, since each CPU is always in exactly one of these four states. In practice, there is occasionally a very small plus or minus difference from this theoretical total.

Column Name	Mode	Description	Data Type
CPUIdle	count	Time in centiseconds the CPU is idle and not waiting for I/O.	FLOAT
CPUIoWait	count	Time in centiseconds CPU is waiting for I/O completion. Note: This represents another variety of Idle, since the CPU is only recorded as being in this state if there are no processes eligible for execution. This is because if there were any such process, the CPU would be immediately dispatched for that process.	FLOAT
CPUUServ	count	Time in centiseconds CPU is busy executing kernel system calls or servicing I/O and timer hardware interrupts.	FLOAT
CPUUExec	count	Time in centiseconds CPU is busy executing user execution code, that is, time spent in a user state on behalf of a process.	FLOAT

CPU Frequency Column

Column Name	Mode	Description	Data Type
CPUFrequencyMHz	track	Tracks CPU speed in MHz.	FLOAT

Elastic TCore Columns

Column Name	Mode	Description	Data Type
TDEnabled	track	Indicates whether the CPU is TD-enabled at the end of the reporting period. If the value is 1, it is TD-enabled. If the value is 0, it is not TD-enabled. Teradata tasks only have access to TD-enabled CPUs while Non-Teradata tasks have access to all online CPUs.	FLOAT

Reserved Columns

Column Name	Mode	Description	Data Type
ReservedS0	n/a	Reserved for future use.	CHAR (1)

Spare Columns

The ResUsageScpu table spare fields are named Spare00 through Spare04, and SpareInt.

The SpareInt field has a 32-bit internal resolution while all other spare fields have a 64-bit internal resolution. All spare fields default to count but can be converted to min, max, or track mode fields if needed when they are used.

Column Name	Mode	Description	Data Type
Spare00 - Spare04, SpareInt	count	Reserved for future use.	FLOAT

Related Information

For details on the different type of data fields, see [About the Mode Column](#).

Summary Mode

When Summary Mode is active for tables in this group, one row is written to the database for each node, summarizing all CPUs per node, for each log interval.

You can determine if a row is in Summary Mode by checking the SummaryFlag column for that row.

IF the SummaryFlag column value is...	THEN the data for that row is being logged...
'S'	in Summary Mode.
'N'	normally.

ResUsageSpma Table

The ResUsageSpma table includes resource usage data for available system-wide information.

You can use the ResUsageSpma table to identify node-level skew by comparing maximum CPU usage to the average CPU consumed across all nodes.

Teradata recommends that you use ResSpmaView to access the data rather than accessing the ResUsageSpma table directly. For more information, see [Resource Usage Views](#).

Note:

Summary Mode is not applicable to the ResUsageSpma table.

Additional system-wide information focused on Vantage internal data is available in [ResUsageSpma Table](#).

This table is created as a MULTiset table. For more information, see [Relational Primary Index](#).

Housekeeping Columns

Relational Primary Index Columns

These columns taken together form the nonunique primary index.

Column Name	Mode	Description	Data Type
TheDate	n/a	Date of the log entry.	DATE
TheTime	n/a	Nominal time of the log entry. Note: Under conditions of heavy system load, entries may be logged late (typically, by no more than one or two seconds), but this column will still contain the time when the entry <i>should have been</i> logged. For more information, see the Secs and NominalSecs columns.	FLOAT
NodeID	n/a	Node ID on which the entry resides. The Node ID is formatted as ZZZ9-9999, where ZZZ9 denotes the four-digit cabinet number and 9999 denotes the four-digit chassis number of the node. For example, a node in chassis 9 of cabinet 3 has a node ID of '3-0009'. Note: SMP nodes have a chassis and cabinet number of 1. For example, the node ID of an SMP node is '1-0001'.	INTEGER

Miscellaneous Housekeeping Columns

These columns provide a generalized picture of the vprocs running on this node, shown as Type n virtual processors where $n = 1$ to 7. Under the current implementation, only Type 1 (AMP), Type 2 (PE), Type 3 (GTW), Type 4 (RSG), and Type 5 (TVS) vprocs exist; vproc types 6 through 7 are not currently used.

Column Name	Mode	Description	Data Type
GmtTime	n/a	Greenwich Mean Time is not affected by the Daylight Saving Time adjustments that occur twice a year.	FLOAT
CabinetID	n/a	The physical cabinet number of the node.	INTEGER
ModuleID	n/a	The physical module number of the node.	INTEGER
NodeType	n/a	Type of node, representing the per node system family type.	CHAR(8)
TheTimestamp	n/a	Number of seconds since midnight, January 1, 1970. This column is useful for aligning data with the DBQL log.	BIGINT
CentiSecs	n/a	Actual number of centiseconds in the logging period. This column is useful when performing data calculations with small elapsed times where the difference between centisecond-based data and whole seconds results in a percentage error.	INTEGER
Secs	n/a	Actual number of seconds in the log period represented by this row. Normally the same as NominalSecs, but can be different in three cases: <ul style="list-style-type: none"> • The first interval after a log rate change • A sample logged late because of load on the system • System clock adjustments affect reported Secs Useful for normalizing the count statistics contained in this row, for example, to a per-second measurement.	SMALLINT
NominalSecs	n/a	Specified or nominal number of seconds in the logging period.	SMALLINT
PM_COD_CPU	n/a	Platform Metering (PM) CPU Capacity On Demand (COD) value in one tenths of percent. For example, a value of 500 represents a PM CPU COD value of 50%. The value is set to 1000 if the PM CPU COD is disabled.	SMALLINT
PM_COD_IO	n/a	Platform Metering (PM) I/O Capacity On Demand (COD) value in whole percent values for the entire system. For example, a value of 50 represents a PM I/O COD value of 50%. The value is set to 100 if the PM I/O COD is disabled.	SMALLINT
WM_COD_CPU	n/a	Workload Management (WM) CPU Capacity On Demand (COD) value in one tenths of a percent. For example, a value of 500 represents a WM CPU COD value of 50.0%. The value is set to 1000 if the WM CPU COD is disabled.	SMALLINT

Column Name	Mode	Description	Data Type
WM_COD_IO	n/a	Workload Management (WM) I/O Capacity On Demand (COD) value in whole percent. For example, a value of 50 represents a WM I/O COD value of 50.0%. The value is set to 100 if the WM I/O COD is disabled.	SMALLINT
TIER_FACTOR	n/a	I/O performance limit placed on a core-reduced node. For example, a value of 75 represents an I/O limit of 75.0% placed before other COD values. The field is sourced from tpanodep->tierfactor, which is derived from the /proc/tdmeter/tier_performance file. A value of 100 indicates that there is no I/O performance limit applied to the node.	SMALLINT
NCPUs	n/a	Number of online CPUs on this node. This column is useful for normalizing the CPU utilization column values for the number of CPUs on the node. This is especially important in: <ul style="list-style-type: none"> • Coexistence systems where the number of CPUs can vary across system nodes. • Elastic TCore systems where the number of online CPUs can change without database restart. 	SMALLINT
Vproc1	n/a	Current count of type 1 (AMP) virtual processors running on the node.	SMALLINT
Vproc2	n/a	Current count of type 2 (PE) virtual processors running under the node.	SMALLINT
Vproc3	n/a	Current count of type 3 (GTW) virtual processors running under the node.	SMALLINT
Vproc4	n/a	Current count of type 4 (RSG) virtual processors running under the node.	SMALLINT
Vproc5	n/a	Current count of type 5 (TVS) virtual processors running under the node.	SMALLINT
Vproc6	n/a	Current count of type 6 virtual processors running under the node. This column reports zeros.	SMALLINT
Vproc7	n/a	Current count of type 7 virtual processors running under the node. This column reports zeros.	SMALLINT
VprocType1	n/a	Type of virtual processor for Vproc1. When the vproc is present on the node, the value is AMP.	CHAR(4)
VprocType2	n/a	Type of virtual processor for Vproc2. When the vproc is present on the node, the value is PE.	CHAR(4)
VprocType3	n/a	Type of virtual processor for Vproc3. When the vproc is present on the node, the value is GTW.	CHAR(4)

Column Name	Mode	Description	Data Type
VprocType4	n/a	Type of virtual processor for Vproc4. When the vproc is present on the node, the value is RSG.	CHAR(4)
VprocType5	n/a	Type of virtual processor for Vproc5. When the vproc is present on the node, the value is TVS.	CHAR(4)
VprocType6	n/a	Type of virtual processor for Vproc6.	CHAR(4)
VprocType7	n/a	Type of virtual processor for Vproc7.	CHAR(4)
MemSize	n/a	Amount of memory on this node in mega bytes. Useful for performing memory usage calculations.	BIGINT
NodeNormFactor	n/a	A per CPU normalization factor that is used to normalize the reported CPU values of the ResUsageSpma table to a single 5100 CPU. This value is scaled by a factor of 100. For example, if the actual factor is 5.25, the value of the NodeNormFactor will be 525.	INTEGER
Active	max	Gets set to a non-zero value whenever one of the other data columns in the row is set.	FLOAT
NetSamples	count	Sample count for sampled statistics for a Bynet. Note: NetSamples is used to normalize all net time monitored statistics to a percent-of-time basis. For example, dividing (NetTxIdle/NetSamples) yields the transmitter-idle time ratio for the net statistics.	FLOAT

Statistics Columns

File System: Segments Acquired Columns

These columns identify the total disk memory segments acquired by the File System during the log period.

For more information, see “Segment Acquires Columns” in the [ResUsageSvpr Table](#).

Column Name	Mode	Description	Data Type
FileAcqKB	count	Total KB logically acquired by FileAcqs.	FLOAT
FileAcqOtherKB	count	Total number of scratch disk segments acquired in KB.	FLOAT
FileAcqReadKB	count	Total KB physically read by FileAcqReads.	FLOAT
FileAcqReads	count	Total number of disk segment acquired that caused a physical read.	FLOAT

Column Name	Mode	Description	Data Type
FileAcqs	count	Total number of disk segments acquires that were logically acquired. Note: Only the FileAcqs column is counted as a cache hit.	FLOAT
FileAcqsOther	count	Total number scratch disk segments that were logically acquired.	FLOAT

File System: Segments Released Columns

These columns identify the total disk memory segments released by the File System, as well as those segments that are dropped from memory during the log period.

For more information, see [ResUsageSvpr Table](#).

Column Name	Mode	Description	Data Type
FileRels	count	Total number of disk segments released by tasks.	FLOAT
FileRelsOther	count	Total number of scratch disk segments released as part of them being deleted.	FLOAT
FileRelKB	count	Total KB logically released by FileRels.	FLOAT
FileRelOtherKB	count	Total number of scratch disk segments released in KB.	FLOAT
FileWriteKB	count	Total KB physically written by FileWrites.	FLOAT
FileWrites	count	Total number of disk segment immediate or delayed physical writes.	FLOAT

File System: Data Block Prefetches Columns

These columns summarize the effects of prefetching data blocks on the file system. For more information, see [ResUsageSvpr Table](#).

A prefetch is either a cylinder read operation or individual block reads operation. Either of these operations are generically called a prefetch.

When all cylinder slots are in use, the cylinder reads revert back to the original algorithm of a block-at-a-time read ahead. So the column FilePreKB is the sum of the size of data blocks logically read by either cylinder reads or data block pre-reads. This also applies to the physical pre-reads. FilePreReadKB includes both physical cylinder reads and single block pre-reads.

The number of data blocks that are pre-read at a time is controlled by the DBS Control performance parameter ReadAhead Count. The default is one block at a time pre-read.

If you enable cylinder reads, there will be extra sectors read in on cylinder reads. An accurate calculation of the wasted KB read by cylinder read is not possible since there are legitimate logical pre-reads that do not incur physical pre-reads.

For more information about cylinder reads, see *Teradata Vantage™ - Database Administration*, B035-1093.

Column Name	Mode	Description	Data Type
FilePreKB	count	Sum of the sizes of data blocks logically loaded with data prefetches (for example, either cylinder reads or individual block reads). For cylinder reads, this column does not include the disk sectors in between the loaded data blocks.	FLOAT
FilePreReadKB	count	Size of the data prefetch (cylinder section or individual blocks being read) that is physically loaded from disk. For cylinder reads, this column includes the disk sectors in between the loaded data blocks.	FLOAT
FilePreReads	count	Number of times a data prefetch was physically performed either as a cylinder read or individual blocks read.	FLOAT
FilePres	count	Total number of times a logical data prefetch was performed (either as a cylinder read or individual block reads).	FLOAT

File System: Contiguous Writes Columns

These columns return the number of blocks marked as contiguous writable, the total size of data blocks marked for contiguous writes, and the number of I/Os on blocks marked for contiguous write.

Column Name	Mode	Description	Data Type
FileContigWlos	count	Number of I/Os completed on data blocks that were marked for contiguous write. You can get the average number of data blocks combined into a single I/O by using the calculation: $\text{FileContigWBlocks} / \text{FileContigWlos}$	FLOAT
FileContigWBlocks	count	Number of data blocks marked for contiguous write. Data blocks created using the contiguous write scheme are accumulated in the cache until the optimal I/O size, the end of the cylinder, or the end of the data created has been reached. At such a point, the blocks will be written with a single I/O and removed from the cache.	FLOAT
FileContigWKB	count	Total size, in KB, of the data blocks that were marked for contiguous write. You can get the average size of the writes that were candidates for being combined into one single I/O by using the calculation:	FLOAT

Column Name	Mode	Description	Data Type
		FileContigWKB /FileContigWBlocks	

File System: Data Segment Lock Requests Columns

These columns summarize the number of lock requests, blocks, and deadlocks on a disk segment.

Note:

The locks referred to here are short term file system locks and are distinct from transaction locks, which are counted in the fields documented in [General Concurrency Control Database Locks Columns](#).

For more information, see [ResUsageSvpr Table](#).

Column Name	Mode	Description	Data Type
FileLockBlocks	count	Number of lock requests that were blocked.	FLOAT
FileLockDeadlocks	count	Number of deadlocks detected on lock requests.	FLOAT
FileLockEnters	count	Number of times a lock was requested.	FLOAT

File System: Depot Columns

These columns summarize the physical writes to the Depot used to protect in-place modifications.

Column Name	Mode	Description	Data Type
FileLargeDepotBlocks	count	Total number of blocks (either WAL or database) that have been protected by large depot writes. Since a large depot write protects multiple blocks, the following calculation results in the average number of blocks protected by each large depot write: $\text{FileLargeDepotBlocks} / \text{FileLargeDepotWrites}$	FLOAT
FileSmallDepotBusy	count	Number of times a small depot I/O request waited for a free slot. A large number in this field indicates more small depot slots are needed. You can use the DBS Control field, SmallDepotCylsPerPdisk, to determine the number of depot cylinders the file system allocates per pdisk (storage device) to contain small slots. For more information about this DBS Control field, see <i>Teradata Vantage™ - Database Utilities</i> , B035-1102.	FLOAT

Column Name	Mode	Description	Data Type
FileLargeDepotBusy	count	Number of times a large depot I/O request waited for a free slot. A large number in this field indicates more large depot slots are needed. You can use the DBS Control field, LargeDepotCylsPerPdisk, to determine the number of depot cylinders the file system allocates per pdisk (storage device) to contain large slots. For more information about this DBS Control field, see <i>Teradata Vantage™ - Database Utilities</i> , B035-1102.	FLOAT
FileLargeDepotWrites	count	Number of large writes to the depot performed to protect in-place modifications. Each large depot write protects multiple in-place writes of either WAL data blocks or database data blocks. The large depot is typically used when blocks age out of memory in the background. Large depot writes are also counted against FileWrites. Therefore, FileWrites still indicates the total writes regardless of whether it was a depot write or a database write.	FLOAT
FileSmallDepotWrites	count	Number of small writes to the depot performed to protect in-place modifications. Each small depot write protects a single in-place write of either a write ahead logging (WAL) data block or a database data block. The small depot is typically used when the in-place writes are initiated by a foreground task. Small depot writes are also counted against FileWrites. Therefore, FileWrites still indicates the total writes regardless of whether it was a depot write or a database write.	FLOAT

General Concurrency Control Database Locks Columns

These columns identify database locking occurrences.

Column Name	Mode	Description	Data Type
DBLockBlocks	count	Number of times a database lock was blocked.	FLOAT
DBLockDeadlocks	count	Number of times a database lock was deadlocked.	FLOAT

Host Controller Channel and TCP/IP Network Traffic Columns

These columns identify the traffic between the host and the node channel and TCP/IP network connections in three levels of granularity:

- Blocks
- Messages
- KB

Blocks are made up of some amount of variable sized messages. ReadKB and WriteKB identify the KB involved in the traffic.

For host controller columns that provide overhead and management information, see [ResUsageShst Table](#) for details.

Column Name	Mode	Description	Data Type
HostBlockReads	count	Number of blocks read in from the host.	FLOAT
HostBlockWrites	count	Number of blocks written out to the host.	FLOAT
HostMessageReads	count	Number of messages read in from the host.	FLOAT
HostMessageWrites	count	Number of messages written out to the host.	FLOAT
HostReadKB	count	KB transferred in from the host.	FLOAT
HostWriteKB	count	KB transferred out to the host.	FLOAT

I/O Hard Limits Columns

The Input/Output Token Allocations (IOTA) columns provide the percentage of disk throughput that is being used.

Column Name	Mode	Description	Data Type
FullPotentiallota	count	Sum of the full potential input or output token allocations from all devices attached to the node. The input or output token allocations are not limited by the IO_COD setting. The value is computed by summing the cod_full_potential_iota field for each device in the /proc/tdmeter/disk_cod_stats file.	FLOAT
CodPotentiallota	count	Sum of the potential input or output token allocations from all devices attached to the node. The input or output token allocations are accumulated only when an I/O is pending on a device, and they are limited by the IO_COD setting. The value is computed by summing the cod_allocated_iota field for each device in the /proc/tdmeter/disk_cod_stats file.	FLOAT
Usedlota	count	Sum of the used input or output token allocations from all devices attached to the node. The value is computed by summing the cod_used_iota field for each device in the /proc/tdmeter/disk_cod_stats file.	FLOAT
IoThrottleCount	count	Number of times an I/O was throttled.	FLOAT

Column Name	Mode	Description	Data Type
IoThrottleTime	count	Total I/O Throttle Time in milliseconds.	FLOAT
IoThrottleTimeMax	max	Maximum I/O Throttle Time in milliseconds.	FLOAT
IoThrottleCntZerolotas	count	Number of times an I/O was throttled due to no IOTA tokens available in the bucket. This data is included in the IoThrottleCount.	FLOAT
IoThrottleCntInsufflotas	count	Number of times an I/O was throttled due to insufficient IOTA tokens available in the bucket. There were some IOTA tokens in the bucket, but not enough. This data is included in the IoThrottleCount.	FLOAT
IoThrottleCntInsufflotasHL	count	Number of times an I/O was throttled due to insufficient IOTA tokens available in the bucket for Hard Limits. There were some IOTA tokens in the bucket, but not enough. This data is included in the IoThrottleCount.	FLOAT
IoThrottleCntMaxQD	count	Number of times an I/O was throttled due to the queue depth to the storage being at maximum. This data is not part of IoThrottleCount.	FLOAT

Memory: Memory Allocation Column

Column Name	Mode	Description	Data Type
MemVprAllocKB	track	<p>Change in memory.</p> <p>MemVprAllocKB represents a delta from the previous reporting period and will report negative values as less memory is used.</p> <p>Note:</p> <p>The original meaning of this column was the total KB attributed to allocations and size-increasing alters for vproc memory types.</p>	FLOAT

Memory: Memory Pages Column

Column Name	Mode	Description	Data Type
MemFreeKB	track	<p>Approximate amount of memory available for use.</p> <p>The MemFreeKB value matches the value reported by the Linux “free” command.</p>	FLOAT

Memory: Memory Availability Management Columns

These columns identify overhead to managing memory when memory availability is a problem.

Column Name	Mode	Description	Data Type
MemCtxtPageReads	count	Number of pages swapped in.	FLOAT
MemCtxtPageWrites	count	Number of pages swapped out.	FLOAT
MemTextPageReads	count	Number of pages paged minus the pages swapped in.	FLOAT

Memory: TIM Column

These columns represent information about the VH (Very Hot) cache, which holds the hottest permanent table cylinders.

Tables that are assigned a temperature of very hot are kept in the FSG cache as long as they:

- Stay very hot.
- Fit into the memory assigned. If the tables cannot fit, the FSG cache considers a sorted list of the hottest segments and assigns them to the very hot cache in temperature sorted order. The temperature of the segment is hot enough to qualify.

The temperature takes into account both physical and logical disk accesses and cache hits (such as, physical and logical I/Os).

Column Name	Mode	Description	Data Type
VHCacheKB	track	Current size of the VH cache in KB. This field is populated by the FSG subsystem.	FLOAT

Memory: Node Level Buffering Columns

Column Name	Mode	Description	Data Type
NLBActiveSessionsMax	max	Maximum concurrent Node Level Buffering sessions. Node Level Buffering is typically used for redistributions (RRD) and duplications (DUP).	FLOAT
NLBSessionsInuse	track	Current number of active Node Level Buffering sessions. Node Level Buffering is typically used for redistributions (RRD) and duplications (DUP).	FLOAT
NLBSessionsCompleted	count	Number of Node Level Buffering sessions completed. Node Level Buffering is typically used for redistributions (RRD) and duplications (DUP).	FLOAT
NLBMsgFlowControlled	track	Number of Node Level Buffered messages being flow controlled on the sender side (or node). Node Level Buffering is typically used for redistributions (RRD) and duplications (DUP).	FLOAT

Column Name	Mode	Description	Data Type
		These messages use a different mailbox than the AMP work mailbox. They are allocated channel linked mailboxes.	
NLBMsgFlowControlledKB	track	Amount of segment memory in use by Flow Controlled Node Level Buffered Messages at the time of the RSS gather operation. Node Level Buffering is typically used for redistributions (RRD) and duplications (DUP). The pages associated with segments used by Node Level Buffered messages are pinned in memory for bynet DMA.	FLOAT

Memory: OS Memory Usage Column

Column Name	Mode	Description	Data Type
PageCacheKB	track	Current KB of Linux page cache. This field is the total of Buffers and Cached line data from /proc/meminfo.	FLOAT
SlabCacheKB	track	Current amount of memory cache managed by the Linux slab allocator.	FLOAT

Memory: PDE Kernel Memory Usage Columns

Column Name	Mode	Description	Data Type
KernMemInuseKB	track	Amount of Kernel memory in use by PDE in kilobytes. Note: The ResSpmaView has an alias field called TotalKernMemInuseKB, derived from: KernMemInuseKB + SegMDLInuseKB	FLOAT
SegMDLInuseKB	track	Total memory in use by segment MDLs. An MDL is an internal PDE data structure needed by the operation of the segment subsystem. The pages associated with MDL segments are pinned in memory for bynet DMA. Note: The ResSpmaView has an alias field called TotalKernMemInuseKB, derived from: KernMemInuseKB + SegMDLInuseKB	FLOAT
SegMaxAvailMB	track	Current MB of maximum PDE segment files available.	FLOAT

Column Name	Mode	Description	Data Type
SegInuseMB	track	Current MB of PDE segment files in use.	FLOAT
SegCacheMB	track	Current MB of segment cache.	FLOAT
SegMDLAlloc	count	Number of segments of all sizes allocated for the MDL pool during this period.	FLOAT
SegMDLFree	count	Number of dirty segments of all sizes freed from the MDL pool during this period.	FLOAT
SegMDLRelease	count	Number of clean segments of all sizes freed from the MDL pool during this period.	FLOAT
SegMDLRecycle	count	Number of segments of all sizes recycled to the MDL pool during this period.	FLOAT
SegMDLAllocKB	count	KB of segments of all sizes allocated for the MDL pool during this period.	FLOAT
SegMDLFreeKB	count	KB of dirty segments of all sizes freed from the MDL pool during this period.	FLOAT
SegMDLReleaseKB	count	KB of clean segments of all sizes freed from the MDL pool during this period.	FLOAT
SegMDLRecycleKB	count	KB of segments of all sizes recycled to the MDL pool during this period.	FLOAT
FsgCacheKB	track	Allocated memory for FSG cache. This includes the memory allocated for TIM (VHCacheKB).	FLOAT

Memory: Page Scan Columns

Column Name	Mode	Description	Data Type
PageScanDirects	count	<p>Number of pages scanned directly in kernel on user context to locate free memory.</p> <p>Note: The ResSpmaView has an alias field called PageScanPerSec, derived from: $(\text{PageScanDirects} + \text{PageScanKswapds}) * 100 / \text{CentiSecs}$</p>	FLOAT
PageScanKswapds	count	<p>Number of pages scanned by kswapd daemon to locate free memory.</p> <p>Note: The ResSpmaView has an alias field called PageScanPerSec, derived from: $(\text{PageScanDirects} + \text{PageScanKswapds}) * 100 / \text{CentiSecs}$</p>	FLOAT

Memory: Page Faults Columns

Column Name	Mode	Description	Data Type
PageMajorFaults	count	Number of major page faults. If the page is not loaded in memory at the time the fault is generated, then it is called a major or hard page fault.	FLOAT
PageMinorFaults	count	Number of minor page faults. If the page is loaded in memory at the time the fault is generated, but is not marked in the memory management unit as being loaded in memory, then it is called a minor or soft page fault.	FLOAT

Native Object Store Columns

Column Name	Mode	Description	Data Type
NosTables	count	Number of Native Object Store tables accessed. Note that access via READ_NOS does not increment this column, although it does increment the other NOS columns.	FLOAT
NosPhysReadIOs	count	Number of physical read I/Os issued against Native Object Store tables.	FLOAT
NosPhysReadIOKB	count	Total KB physically read from Native Object Store tables.	FLOAT
NosFiles	count	Number of Native Object Store files read in satisfying queries against Native Object Store tables.	FLOAT
NosFilesSkipped	count	Number of Native Object Store files skipped due to errors.	FLOAT
NosRecordsReturned	count	Number of records returned in satisfying queries against Native Object Store tables.	FLOAT
NosRecordsReturnedKB	count	Total KB of records returned from Native Object Store files (after any decompression and character conversion performed).	FLOAT
NosRecordsSkipped	count	Number of Native Object Store records skipped due to errors.	FLOAT
NosTotalIOWaitTime	count	Total time in seconds waiting for I/Os when reading Native Object Store files.	FLOAT
NosMaxIOWaitTime	max	Maximum single I/O wait time in seconds when reading Native Object Store files.	FLOAT

Column Name	Mode	Description	Data Type
NosCPUTime	count	Total CPU time in seconds spend decompressing, translating, or otherwise processing Native Object Store data as part of reading it.	FLOAT
NosFilesWritten	count	Number of file writes attempted by Native Object Store.	FLOAT
NosPhysWriteIOs	count	Total physical write IOs for Native Object Store files.	FLOAT
NosPhysWriteIOKB	count	Total KB of physical write IOs for Native Object Store files.	FLOAT

Net: Point-to-Point Net Traffic Columns

These columns identify the number (Reads, Writes) and amount (ReadKB, WriteKB) of input and output messages passing through the database nets through point-to-point (1:1) methods (PtP). It excludes TCP/IP traffic.

Column Name	Mode	Description	Data Type
MsgPtPReadKB	count	Total KB of net point-to-point messages input to processes on the node via the message subsystem.	FLOAT
MsgPtPReads	count	Number of net point-to-point messages input to processes on the node via the message subsystem.	FLOAT
MsgPtPWriteKB	count	Total KB of net point-to-point messages output from processes on the node via the message subsystem.	FLOAT
MsgPtPWrites	count	Number of net point-to-point messages output from processes on the node via the message subsystem.	FLOAT

Net: Broadcast Net Traffic Columns

These columns identify the number (Reads, Writes) and amount (ReadKB, WriteKB) of input and output messages passing through the Teradata nets through broadcast (1:many) methods (Brd).

Note:

If a single broadcast message is delivered to multiple processes in this node, the NetBrdReads and NetBrdReadKB are only incremented once.

Column Name	Mode	Description	Data Type
MsgBrdReadKB	count	Total KB of net broadcast messages input to processes on the node via the message subsystem.	FLOAT

Column Name	Mode	Description	Data Type
MsgBrdReads	count	Number of net broadcast messages input to processes on the node via the message subsystem.	FLOAT
MsgBrdWriteKB	count	Total KB of net broadcast messages output from processes on the node via the message subsystem.	FLOAT
MsgBrdWrites	count	Number of net broadcast messages output from processes on the node via the message subsystem.	FLOAT

Net: Network Transport Data Columns

These columns identify the number (Reads, Writes) and amount of input and output passing through the Teradata nets.

On a single-node (virtual network [vnet]) system, net-specific statistics are not meaningful and are always zero.

Column Name	Mode	Description	Data Type
NetMsgPtpWriteKB	count	Amount of point-to-point message data in KB transmitted by both Bynets.	FLOAT
NetMsgPtpWrites	count	Number of point-to-point messages transmitted by both Bynets.	FLOAT
NetMsgBrdWriteKB	count	Amount of broadcast message data in KB transmitted by both Bynets.	FLOAT
NetMsgBrdWrites	count	Number of broadcast messages transmitted by both Bynets	FLOAT
NetMsgPtpReadKB	count	Amount of point-to-point message data in KB received by both Bynets.	FLOAT
NetMsgPtpReads	count	Number of point-to-point messages received by both Bynets.	FLOAT
NetMsgBrdReadKB	count	Amount of broadcast message data in KB received by both Bynets.	FLOAT
NetMsgBrdReads	count	Number of broadcast messages received by both Bynets.	FLOAT
NetRxKBBrd	count	Total broadcast KB received over all Bynets.	FLOAT
NetRxKBpTP	count	Total point-to-point KB received over all Bynets.	FLOAT
NetTxKBBrd	count	Total broadcast KB transmitted over all Bynets.	FLOAT
NetTxKBpTP	count	Total point-to-point KB transmitted over all Bynets.	FLOAT

Net: Net Controller Status and Miscellaneous Management

These columns provide utilization and other status information about the Teradata net controllers.

On a single-node (VNET) system, net-specific statistics are not meaningful and always report zero.

Column Name	Mode	Description	Data Type
NetTxConnected	count	Number of samples showing the transmitter connected on a Bynet.	FLOAT
NetRxConnected	count	Number of samples showing the receiver connected on a Bynet.	FLOAT
NetTxRouting	count	Number of samples showing the transmitter routing on a Bynet.	FLOAT
NetTxIdle	count	Number of samples showing the transmitter idle on a Bynet.	FLOAT
NetRxIdle	count	Number of samples showing the receiver idle on a Bynet.	FLOAT

Net: Net Merge Columns

Column Name	Mode	Description	Data Type
NetActiveMrg	track	The number of concurrent active merges on all Bynets.	FLOAT
NetActiveMrgKB	track	Current memory usages by bynet merge jobs.	FLOAT
NetMrgBlock	count	Number of times a merge message was blocked until delivery of outstanding outgoing messages.	FLOAT
NetMrgCompleted	count	Number of bynet merges completed.	FLOAT
NetMsgChannelBlock	count	Number of times the net software was blocked because the channel was not in RxReady state on the receiver.	FLOAT
NetMsgFCBlock	count	Number of times the net software was blocked because the receiver was flow controlled.	FLOAT
NetMsgGroupBlock	count	Number of times the net software was blocked because the receiver could not implicitly enter the group.	FLOAT
NetMsgResourceBlock	count	Number of times the net software was blocked because the receiver could not get the necessary resources.	FLOAT
NetMsgRxBlock	count	Number of times the net software could not accept a message and caused a transmitter to block.	FLOAT

Net: Net Queues Columns

These columns identify lengths of the various internal queues used by the network controllers.

All of the aggregated sampled statistics columns in the following table are net-specific, that is, they relate to each specific Bynet. On a single-node system, net-specific statistics are not meaningful and are always zero.

Column Name	Mode	Description	Data Type
NetBlockQueue	track	Total number of services on the BlockableService queue at the current time. Services can be blocked for a variety of reasons including receiver flow control, receiver resource usage, and daemon services.	FLOAT
NetBlockQueueMax	max	Maximum number of services on the BlockableServices queue in this log interval.	FLOAT
NetBlockQueueTotal	count	Total number of services added to the BlockableServices queue in this log interval.	FLOAT

Net: Net Circuit Management Columns

The Net Circuit Management columns identify the management of Teradata net circuits (Circ) and raw data traffic on the network (hardware) on all networks.

Column Name	Mode	Description	Data Type
NetBackoffs	count	Software backoffs, defined as BNS service blocked occurrences, without regard for which net was involved.	FLOAT
NetHWBackoffs	count	Hardware backoffs reported by the BLM for all Bynets.	FLOAT
NetRxCircBrd	count	Total number (both normal and high priority) of broadcast circuits received on all Bynets.	FLOAT
NetRxCircPtp	count	Total number (both normal and high priority) of point-to-point circuits received on all Bynets.	FLOAT
NetTxCircBrd	count	Total number (both normal and high priority) of broadcast circuits transmitted on all Bynets.	FLOAT
NetTxCircHPBrd	count	Number of high priority broadcast circuits transmitted on all Bynets.	FLOAT
NetTxCircHPPtP	count	Number of high priority point-to-point circuits transmitted on all Bynets.	FLOAT
NetTxCircPtp	count	Total number (both normal and high priority) of point-to-point circuits transmitted on all Bynets.	FLOAT

Net: Group Coordination Messages Columns

These columns identify messages that are communicated through the Teradata net for coordination of a process among a group of vprocs. Coordination is handled either through semaphores, groups, or channels.

Column Name	Mode	Description	Data Type
NetChanInUse	track	Number of channels in use at the current time.	FLOAT
NetChanInUseMax	max	Maximum number of channels in use.	FLOAT
MsgChnLastDone	count	Number of last done events that occurred on this node. Note: The last AMP to finish an operation may send a last done broadcast message indicating the work is done for this step. This is used in tracking down the slowest node or AMP in the system. A node or AMP that has more last done messages than the others could be a bottleneck in the system performance.	FLOAT
NetGroupInUse	track	Number of groups in use at the current time. This number should be same across all nodes.	FLOAT
NetGroupInUseMax	max	Maximum number of groups in use during each log interval.	FLOAT
NetSemInUse	track	Number of semaphores in use at the current time.	FLOAT
NetSemInUseMax	max	Maximum number of semaphores in use during each log interval.	FLOAT

Net: Merge Services Columns

These columns identify activity occurring through merge (many:1) methods (Mrg) on Teradata net.

Column Name	Mode	Description	Data Type
NetMrgRxKB	count	Number of KB received, without regard to which net, by merge receive services for currently active merge operations.	FLOAT
NetMrgRxRows	count	Number of data rows received, without regard to which net, by merge receive services for currently active merge operations.	FLOAT
NetMrgTxKB	count	Number of KB transmitted, without regard to which net, by merge transmission services for currently active merge operations.	FLOAT
NetMrgTxRows	count	Number of data rows transmitted, without regard to which net, by merge transmission services for currently active merge operations.	FLOAT

Process Allocation Columns

These columns represent all currently allocated processes, subdivided into the possible process states of running, ready, blocked or suspended.

Column Name	Mode	Description	Data Type
ProcBlocked	track	Number of threads blocked waiting for I/O at the current time.	FLOAT
ProcReady	track	Number of runnable or ready tasks, also called threads, able to execute on CPUs when a CPU becomes available.	FLOAT
ProcReadyMax	max	Maximum number of ready tasks, also called threads, able to execute on CPUs when a CPU becomes available.	FLOAT

Process Scheduling: Elastic TCore Columns

Column Name	Mode	Description	Data Type
TDEnabledCPUs	track	Number of CPUs available for Teradata tasks at the end of the reporting period. These CPUs are TD-enabled CPUs. Teradata tasks only have access to TD-enabled CPUs while Non-Teradata tasks have access to all online CPUs. This count may change dynamically after database is up but is always less than or equal to the count reported in NCPUs.	FLOAT

Process Scheduling: Process Pending Snapshot Columns

These columns identify how many processes are blocked for each possible reason. These columns total (minus ProcPendDBLock) approximately ProcBlocked, since we can only be blocked on one blocking type at a time.

Note:

In analyzing resource usage, a distinction should be made between the following two kinds of process blocks:

- Block involves a process that is logically idle, waiting to receive work on its primary mailbox, or for a timer to elapse. This block does not affect throughput.
- Block involves a process that has work to do but is being prevented from proceeding by some circumstance like a segment lock or flow control. This kind of block does affect throughput.

The first kind of block is represented by column ProcPendNetRead; the second kind is represented by the remaining columns described here.

Column Name	Mode	Description	Data Type
ProcPendDBLock	track	Number of processes blocked pending database locks.	FLOAT
ProcPendFsgLock	track	Number of processes blocked pending an FSG lock.	FLOAT
ProcPendFsgRead	track	Number of processes blocked pending a File Segment (FSG) read from disk.	FLOAT
ProcPendFsgWrite	track	Number of processes blocked pending an FSG write to disk.	FLOAT
ProcPendMemAlloc	track	Number of processes blocked pending memory allocations.	FLOAT
ProcPendMisc	track	Number of processes blocked pending miscellaneous events.	FLOAT
ProcPendMonitor	track	Number of processes blocked pending a user monitor.	FLOAT
ProcPendMonResume	track	Number of processes blocked pending a user monitor resume from a yield.	FLOAT
ProcPendNetRead	track	<p>Number of processes blocked pending non-step work, that is, the number of processes blocked on any mailbox other than the work mailbox.</p> <p>Note: Non-step work is anticipated work the process spawned off and is now waiting for some type of response from the spawned process or processes. Non-step work is <i>not</i> unanticipated work such as a new work request sent when a user initiates a request from the host.</p>	FLOAT
ProcPendNetThrottle	track	Number of processes blocked pending delivery of outstanding outgoing messages.	FLOAT
ProcPendQnl	track	Number of processes blocked pending a TSKQNL lock.	FLOAT
ProcPendSegLock	track	Number of processes blocked pending a segment lock.	FLOAT

Process Scheduling: Process Block Count Columns

These columns identify how many times a process became blocked on which blocking type. Average time blocked can be approximated by dividing the corresponding process pending wait time column by the process block count column. For a list of these columns, see [Process Scheduling: Process Pending Wait Time Columns](#).

Column Name	Mode	Description	Data Type
ProcBlksFsgRead	count	Number of process blocks for an FSG read from disk.	FLOAT
ProcBlksFsgWrite	count	Number of process blocks for an FSG write to disk.	FLOAT

Column Name	Mode	Description	Data Type
ProcBlksDBLock	count	Number of process blocks for database locks. The AMP Worker Task can do other work while the lock is blocked.	FLOAT
ProcBlksFsgLock	count	Number of process blocks for an FSG lock.	FLOAT
ProcBlksTime	count	Number of process blocks waiting only for timer expiration.	FLOAT
ProcBlksMemAlloc	count	Number of process blocks for memory allocations.	FLOAT
ProcBlksMisc	count	Number of process blocks for miscellaneous events.	FLOAT
ProcBlksMonitor	count	Number of process blocks for a user monitor.	FLOAT
ProcBlksMonResume	count	Number of process blocks for a user monitor resume from a yield.	FLOAT
ProcBlksMsgRead	count	Number of process blocks for non-step work.	FLOAT
ProcBlksNetThrottle	count	Number of process blocks for delivery of outstanding outgoing messages.	FLOAT
ProcBlksQnl	count	Number of process blocks for a TSKQNL lock.	FLOAT
ProcBlksSegLock	count	Number of process blocks for a disk or task context (for example, scratch, stack, and so on) segment lock.	FLOAT

Process Scheduling: Process Pending Wait Time Columns

These columns identify the how long the processes were blocked for each possible reason listed below.

Note:

Since this time is only accounted for when a blocked process leaves the blocked state, it is possible for this statistic to be much larger than the amount of time available to all processes in a single log period.

Column Name	Mode	Description	Data Type
ProcWaitDBLock	count	Total time in centiseconds processes were blocked pending database locks.	FLOAT
ProcWaitFsgLock	count	Total time in centiseconds processes were blocked pending an FSG lock.	FLOAT
ProcWaitFsgRead	count	Total time in centiseconds processes were blocked pending an FSG read from disk.	FLOAT
ProcWaitFsgWrite	count	Total time in centiseconds processes were blocked pending an FSG write to disk.	FLOAT
ProcWaitMemAlloc	count	Total time in centiseconds processes were blocked pending memory allocations.	FLOAT

Column Name	Mode	Description	Data Type
ProcWaitMisc	count	Total time in centiseconds processes were blocked pending miscellaneous events.	FLOAT
ProcWaitMonitor	count	Total time in centiseconds processes were blocked pending a user monitor.	FLOAT
ProcWaitMonResume	count	Total time in centiseconds processes were blocked pending a user monitor resume from a yield.	FLOAT
ProcWaitMsgRead	count	Total time in centiseconds processes were blocked pending non-step work.	FLOAT
ProcWaitNetThrottle	count	Total time in centiseconds processes were blocked pending delivery of outstanding outgoing messages.	FLOAT
ProcWaitPageRead	count	Total time in centiseconds processes were blocked pending a <i>page read</i> from disk.	FLOAT
ProcWaitTime	count	Total time in centiseconds processes were blocked pending some amount of elapsed time only.	FLOAT
ProcWaitQnl	count	Total time in centiseconds processes were blocked pending a TSKQNL lock.	FLOAT
ProcWaitSegLock	count	Total time in centiseconds processes were blocked pending a disk or task context (for example, scratch, stack, and so on) segment lock.	FLOAT

Process Scheduling: Process Scheduling CPU Switching Column

Column Name	Mode	Description	Data Type
CPUProcSwitches	count	Number of times the scheduler switched a currently active process of a CPU to a new process.	FLOAT

Process Scheduling: CPU Utilization Columns

The CPU utilization columns count all CPU activities, including activities performed for virtual processors, and represent the sum of all CPUs on the node.

To obtain the average node CPU value for each column, CPU (Idle, IOWait, UServ, UExec), divide the column data by the number of CPUs per node (the value in the NCPUs column) and the number of centiseconds (CentiSecs column) in the logging interval.

- CPU idle time = CPUIdle + CPUIoWait
- CPU busy time = CPUUServ + CPUUExec

The NodeNormFactor is the per node normalization factor. This is related to the NodeType value reported in this resource usage table. The normalization factor modifies the reported CPU times to the equivalent time of a specified virtual processor. This does not add up to the reported CPU time.

To calculate the non-normalized total CPU time, use the following formula:

CentiSecs x NCPUs = CPUIdle + CPUIoWait + CPUUServ + CPUUExec

Note:

The CPU time returned in centiseconds is more accurate than those returned in seconds.

Column Name	Mode	Description	Data Type
CPUIdle	count	Time in centiseconds CPUs are idle and not waiting for I/O.	FLOAT
CPUIoWait	count	Time in centiseconds CPUs are idle and waiting for I/O completion. Note: This time represents another variety of Idle, since a CPU is only in this state if there are no processes eligible for execution. If there was a process available, the CPU would be immediately dispatched for that process.	FLOAT
CPUUExec	count	Time in centiseconds CPUs are busy executing user execution code, that is, time spent in a user state on behalf of a process. CPUUExec reports the CPU time not used in the system call, or in the kernel.	FLOAT
CPUUServ	count	Time in centiseconds CPUs are busy executing user service code, that is, privileged work performing system services on behalf of user execution processes which do not have root access. CPUUServ reports if a task executing a step used CPU while in the kernel.	FLOAT

Process Scheduling: CPU Hard Limits Columns

Column Name	Mode	Description	Data Type
CpuThrottleCount	count	Number of times that WM CPU COD throttling was triggered.	FLOAT
CpuThrottleTime	count	Time in centiseconds that WM CPU COD throttling was active.	FLOAT

Real Time Boosting

Two metrics track the Real Time Boosting statistics.

Column Name	Mode	Description	Data Type
RTCpuTime	count	Total CPU time in milliseconds consumed by boosted realtime tasks. Realtime tasks CPU time is not limited by COD or Hard Limit.	FLOAT
RTTasksMax	max	Maximum number of tasks running in boosted realtime priority.	FLOAT

TASM: AMP Worker Task Columns

These columns report statistics about the AMP Worker Tasks.

For more information about the ResUsageSawt table and columns, see [ResUsageSawt Table](#).

Column Name	Mode	Description	Data Type
AmpsFlowControlled	track	Number of AMPs currently in flow control on the work input mailbox.	FLOAT
FlowCtlCnt	count	Number of times this log period that the node entered the flow control state from a non-flow controlled state.	FLOAT
AwtInuse	track	Number of AMP Worker Tasks currently in use for this node.	FLOAT
AwtInuseMax	max	Peak number of AWTs for any one of the AMPs on the node. Note: AwtInuseMax is not the peak number in use on the node for all AMPs at any one point in time. AwtInuseMax represents the largest number of AWTs in use on any single AMP.	FLOAT

TASM: Priority Scheduler Columns

These columns provide data specific to the Priority Scheduler.

For more information about the ResUsageSps table and columns, see [ResUsageSps Table](#).

Column Name	Mode	Description	Data Type
PSNumRequests	count	Number of work requests received on this node.	FLOAT
PSQWaitTime	count	Time in centiseconds that work requests waited on an input queue before being serviced. To get an approximate average QWaitTime per request during this period, divide QWaitTime by NumRequests.	FLOAT
PSServiceTime	count	Time in centiseconds that work requests required for service. To get an approximate average ServiceTime per request during this period, divide ServiceTime by NumRequests.	FLOAT

Teradata VS: Disk Cache Columns

These columns track the node local NVMe SSD disk cache usage.

Column Name	Mode	Description	Data Type
DiskCacheInUseKB	track	Current KBs of PDISK cache in use. This column is valid on SLES11 SP3 or later systems only.	FLOAT
DiskCacheTotalKB	track	Total KBs of PDISK cache available. The value is 0 if PDISK cache is disabled or not used. This column is valid on SLES11 SP3 or later systems only.	FLOAT

Teradata VS Columns

These columns identify pdisk I/O statistics that are reported by the Node Agent.

Column Name	Mode	Description	Data Type
TvsReadCnt	count	Number of logical device reads. The TvsReadCnt column is a summary of the ReadCnt column in the SPDSK table.	FLOAT
TvsReadRespMax	max	Maximum read response time in centiseconds during the reporting period.	FLOAT
TvsReadRespTot	count	Total of individual read response time in centiseconds. The TvsReadRespTot column is a summary of the ReadRespTot column in the SPDSK table.	FLOAT
TvsWriteCnt	count	Number of logical device writes. The TvsWriteCnt column is a summary of the WriteCnt column in the SPDSK table.	FLOAT
TvsWriteRespMax	max	Maximum response time in centiseconds of logical device writes during the reporting period.	FLOAT
TvsWriteRespTot	count	Total of individual write response time in centiseconds. TvsWriteRespTot is a summary of the WriteRespTot column in the SPDSK table.	FLOAT

User Commands: Number of Commands Columns

These columns summarize the number of statements given to Vantage by the user.

For more information, see [ResUsageShst Table](#).

Column Name	Mode	Description	Data Type
CmdDDLStmts	count	Number of alter, modify, drop, create, replace, grant or revoke commands.	FLOAT
CmdDeleteStmts	count	Number of delete commands.	FLOAT
CmdInsertStmts	count	Number of insert commands.	FLOAT
CmdSelectStmts	count	Number of select commands.	FLOAT
CmdUpdateStmts	count	Number of update commands.	FLOAT
CmdUtilityStmts	count	Number of utility commands.	FLOAT
CmdOtherStmts	count	Number of other commands.	FLOAT

User Command: Arrival and Departure Columns

These columns summarize the arrival and departure of user statements.

For more information, see [ResUsageShst Table](#).

Column Name	Mode	Description	Data Type
CmdStmtErrors	count	Number of statements that departed in error.	FLOAT
CmdStmtFailures	count	Number of statements that departed in failure or were aborted.	FLOAT
CmdStmtSuccesses	count	Number of statements that departed normally.	FLOAT

Reserved Columns

Column Name	Mode	Description	Data Type
ReservedS0	n/a	Reserved for future use.	CHAR(4)
ReservedS1	n/a	Reserved for future use.	CHAR(8)
Reserved00	n/a	Reserved for future use.	SMALLINT

Spare Columns

The ResUsageSpma table spare fields are named Spare00 through Spare19, and SpareInt.

The SpareInt field has a 32-bit internal resolution while all other spare fields have a 64-bit internal resolution. All spare fields default to count but can be converted to min, max, or track mode fields if needed when they are used.

Column Name	Mode	Description	Data Type
Spare00 - Spare01	count	Reserved for future use.	FLOAT
Spare02	track	I/O capacity configured for a node in MB/s based on a 96k 90/10 R/W workload not adjusted for COD. The permanent column name NodeMbs is reserved for future use.	FLOAT
Spare03 - Spare06	count	Reserved for future use.	FLOAT
Spare07	count	Number of prioritized I/Os that were starved and had their priority updated to alleviate starvation. A prioritized I/O is one that Priority Scheduler issues to the Teradata I/O Scheduler (TDSCHED) Expedited or Normal priority RBTREE queues with a non-zero priority key that determines its position in the associated queue. One of these I/Os that is starved has waited in the TDSCHED queue longer than the Priority Scheduler I/O anti-starvation threshold, the value of which appears in "schmon -iodata -debug" output with the "iostarvedthreshold=" field. The permanent column name IoPrioStarvedCount is reserved for future use.	FLOAT
Spare08	count	Time in centiseconds processes were blocked pending non-step work (AWT only). This data is included in ProcWaitMsgRead. The permanent column name ProcWaitMsgReadAwt is reserved for future use.	FLOAT
Spare09 - Spare19, SpareInt	count	Reserved for future use.	FLOAT

Related Information

For more information about the different type of data fields, see [About the Mode Column](#).

ResUsageSawt Table

The ResUsageSawt table contains:

- The current and maximum number of AMP Worker Tasks in use by work type
- Flow control information

You can use the ResUsageSawt table to:

- Report the pattern in AMP Worker Task usage for standard or expedited work types.
- Monitor the length of each message queue of the AMP, the queue which holds work messages from the Dispatcher that are waiting to get an AMP Worker Task.
- Identify if one or more AMPs have entered the state of flow control, and how often, during the logging interval.

If you enable table logging, the data is written to the database once for each log period.

To consolidate and summarize the total number of rows written to the database, you can enable Summary Mode. For details, see [Summary Mode](#).

Teradata recommends that you use ResSawtView to access the data rather than accessing the ResUsageSawt table directly. For more information, see [Resource Usage Views](#).

Note:

This table is created as a MULTiset table. For more information see [Relational Primary Index](#).

Housekeeping Columns

Relational Primary Index Columns

These columns taken together form the nonunique primary index.

Column Name	Mode	Description	Data Type
TheDate	n/a	Date of the log entry.	DATE
TheTime	n/a	Nominal time of the log entry. Note: Under conditions of heavy system load, entries may be logged late (typically, by no more than one or two seconds), but this column will still contain the time value when the entry <i>should have been</i> logged. For more information, see the Secs and NominalSecs columns.	FLOAT

Column Name	Mode	Description	Data Type
NodeID	n/a	<p>Node ID on which the entry resides. The Node ID is formatted as ZZZ9-9999, where ZZZ9 denotes the four-digit cabinet number and 9999 denotes the four-digit chassis number of the node. For example, a node in chassis 9 of cabinet 3 has a node ID of '3-0009'.</p> <p>Note: SMP nodes have a chassis and cabinet number of 1. For example, the node ID of an SMP node is '1-0001'.</p>	INTEGER

Miscellaneous Housekeeping Columns

These columns provide statistics on current logging characteristics.

Column Name	Mode	Description	Data Type
GmtTime	n/a	Greenwich Mean Time is not affected by the Daylight Saving Time adjustments that occur twice a year.	FLOAT
CabinetID	n/a	The physical cabinet number of the node.	INTEGER
ModuleID	n/a	The physical module number of the node.	INTEGER
NodeType	n/a	Type of node, representing the per node system family type.	CHAR(8)
VprId	n/a	Identifies the vproc number. All Vprocs in this table are AMPS so there is no VprType column provided. In Summary Mode, this column is -1.	INTEGER
Secs	n/a	<p>Actual number of seconds in the log period represented by this row. Normally the same as NominalSecs, but can be different in three cases:</p> <ul style="list-style-type: none"> • The first interval after a log rate change • A sample logged late because of load on the system • System clock adjustments affect reported Secs <p>Useful for normalizing the count statistics contained in this row, for example, to a per-second measurement.</p>	SMALLINT
CentiSecs	n/a	Number of centiseconds in the logging period. This column is useful when performing data calculations with small elapsed times where the difference between centisecond-based data and whole seconds results in a percentage error.	INTEGER
NominalSecs	n/a	Specified or nominal number of seconds in the logging period.	SMALLINT
SummaryFlag	n/a	Summarization status of this row. If the value is 'N,' the row is a non-summary row. If the value is 'S,' the row is a summary row.	CHAR(1)
Active	max	Controls whether or not the rows will be logged to the resource usage tables if Active Row Filter Mode is enabled.	FLOAT

Column Name	Mode	Description	Data Type
		<p>If Active is set to a non-zero value, the row contains data columns.</p> <p>If Active is set to a zero value, none of the data columns in the row have been updated during the logging period.</p> <p>For example, if you enable Active Row Filter Mode, the rows that have a zero Active column value will not be logged to the resource usage tables.</p>	
TheTimestamp	n/a	<p>Number of seconds since midnight, January 1, 1970.</p> <p>This column is useful for aligning data with the DBQL log.</p>	BIGINT
PM_COD_CPU	n/a	<p>Platform Metering (PM) CPU Capacity On Demand (COD) value in one tenths of a percent. For example, a value of 500 represents a PM CPU COD value of 50.0%.</p> <p>The value is set to 1000 if the PM CPU COD is disabled.</p>	SMALLINT
WM_COD_CPU	n/a	<p>Workload Management (WM) CPU Capacity On Demand (COD) value in one tenths of a percent. For example, a value of 500 represents a WM CPU COD value of 50.0%.</p> <p>The value is set to 1000 if the WM CPU COD is disabled.</p>	SMALLINT

Statistics Columns

TASM: AMP Worker Task Columns

These columns report statistics about the AMP Worker Tasks.

Column Name	Mode	Description	Data Type
Available	track	<p>Number of unreserved AMP Worker Tasks from the pool that are not being used at the end of the interval.</p> <p>For example, if 12 of the normally 56 unreserved AMP Worker Tasks are removed from the pool by reserving them for expedited work, there could at most be 44 unreserved AMP Worker Tasks available.</p> <p>If in this log period, 10 unreserved AMP Worker Tasks are taken from the pool to service 10 queries that are still executing, there would be only 34 available at the end of the log period.</p>	FLOAT
AvailableMin	min	<p>Minimum number of unreserved AMP Worker Tasks available in the pool for each AMP for the logged period.</p> <p>For example, a zero value means there were no unreserved AMP Worker Tasks available in the pool at some point during the reporting period.</p>	FLOAT

Column Name	Mode	Description	Data Type
AvailableForWork00	track	Unused limit of AWTs for work type 00 at end period. For example, if the limit is 50 AWTs for work type 00, and 4 AWTs are in use for work type 00, this field contains $(50-4) = 46$.	FLOAT
AvailableForWork08	track	Unused limit of AWTs for work type 08 at end period. For example, if the limit is 50 AWTs for work type 08, and 4 AWTs are in use for work type 08, this field contains $(50-4) = 46$.	FLOAT
AvailableForWork00Min	min	The minimum value of AvailableForWork00 over the entire period.	FLOAT
AvailableForWork08Min	min	The minimum value of AvailableForWork08 over the entire period.	FLOAT
AWTsConfigured	const	Current setting for AMP Worker Task (for example, 80, 100, or so on) in the DBS Control MaxLoadAWT field. For more information, see information about MaxLoadAWT in <i>Teradata Vantage™ - Database Utilities</i> , B035-1102.	FLOAT
FlowControlled	track	Specifies if an AMP is in flow control. If the value is non-zero, the AMP is in flow control. Resource usage indicates flow control on work mailbox name 2-11 only. Work mailbox name 2-11 is the incoming mailbox for AMPs, where 2 is the mailbox number and 11 is the AMP Worker Task partition. However, resource usage indicates flow control on work mailbox name 2-11 for all work types. For example, if any work type is in flow control, the system is in flow control.	FLOAT
FlowCtlCnt	count	Number of times during the log period that the system entered the flow control state from a non-flow controlled state.	FLOAT
FlowCtlTime	count	Total time, in milliseconds, that an AMP is in flow control.	FLOAT
InuseMax	max	Maximum number of AMP Worker Tasks in use at any one time during the log period.	FLOAT
MailBoxDepth	track	Current depth of the AMP work mailbox at the end of the period.	FLOAT
UnReservedAwtPoolSize	const	Total number of AWTs in the unreserved AWT pool.	
WorkTypeInuse00 -WorkTypeInuse15	track	Current number of AMP Worker Tasks in use during the log period for each work type for the Vprld vproc.	FLOAT

Column Name	Mode	Description	Data Type
		Note: The WorkTypeInuse04 column value is always at least 1 due to the internal database design for the Control AMP rows in the ResUsageSawt table.	
WorkTypeMax00 - WorkTypeMax15	max	Maximum number of AMP Worker Tasks in use at one time during the log period for each work type for the Vprld vproc. In Summary Mode, the WorkTypeMax column values are the Max of the values for all the AMPs.	FLOAT
WorkTypeAWTEXhausted00 - WorkTypeAWTEXhausted15	count	Number of times a work message cannot be assigned to an AWT. This is caused when no AWT is available in the available pool, or when the number of AWTs in use for the work type reaches its limits.	FLOAT

TASM: Work Type Descriptions

The WorkTypeInuse and WorkTypeMax array data columns above each contain 16 Work Type entries that are described here. For example, WorktypeInuse00 contains the number of in use AMP Worker Tasks that are of Work Type MSGWORKNEW, and WorktypeInuse01 contains the values for MSGWORKONE.

These columns allow you to monitor the usage of the AMP Worker Tasks of each work type. This can be used to determine:

- If the usage is close to the maximum values defined.
- What type of work the AMP Worker Tasks are doing.
- Characteristics of the system during skew conditions or when there are AMP Worker Task shortages.

Use the tdntune utility to determine the settings for Flow Control. For information on Expedited Allocation Groups, see *Teradata Vantage™ - Database Utilities*, B035-1102.

Column Name	Mode	Description	Data Type
MSGWORKNEW	n/a	Used for new work requests. This work type has the lowest number, which means it is queued last. It also has the effect of honoring secondary requests needed to complete existing work items before any new ones are started. A zero value is used for new work items.	n/a
MSGWORKONE	n/a	First level secondary work items. Numbered work types are used for secondary work items. For example, work type one (MSGWORKONE) is used for secondary work requests spawned by new work items; work type two (MSGWORKTWO) requests are spawned from work type one requests and queued for delivery before work type	n/a

Column Name	Mode	Description	Data Type
		one requests; and so on. Each numbered work type is queued for delivery just before the one from which it is spawned.	
MSGWORKTWO	n/a	Second level secondary work items.	n/a
MSGWORKTHREE	n/a	Special types of database work.	n/a
MSGWORKFOUR	n/a	Start System Recover.	n/a
MSGWORKFIVE	n/a	<p>Reports new work for utilities, such as FastLoad, MultiLoad, and FastExport if utilities are configured to use a separate pool of work types.</p> <p>Note: This column is not normally used and the MSGWORKNEW, MSGWORKONE, and MSGWORKTWO columns report work requests for utilities.</p>	n/a
MSGWORKSIX	n/a	First level secondary work spawned work for utilities such as FastLoad, MultiLoad, and FastExport. If the utilities are not configured to use a separate pool of work types, they use MSGWORKNEW, MSGWORKONE, and MSGWORKTWO.	n/a
MSGWORKSEVEN	n/a	Second level secondary work for utilities such as FastLoad, MultiLoad, and FastExport. If the utilities are not configured to use a separate pool of work types, they use MSGWORKNEW, MSGWORKONE, and MSGWORKTWO.	n/a
MSGWORKEIGHT	n/a	New work for Expedited Allocation Groups.	n/a
MSGWORKNINE	n/a	First level spawned work for Expedited Allocation Groups.	n/a
MSGWORKTEN	n/a	Second level spawned work for Expedited Allocation Groups.	n/a
MSGWORKELEVEN	n/a	Not used.	n/a
MSGWORKABORT	n/a	Used for transaction abort requests. This work type has a higher value than the numbered work types so that abort requests are honored before beginning any additional work item for the transactions being aborted. The array number for MSGWORKABORT is 12.	n/a
MSGWORKSPAWN	n/a	Used for spawned abort requests and is delivered before normal aborts. The array number for MSGWORKSPAWN is 13.	n/a
MSGWORKNORMAL	n/a	Used for messages that do not fall within the standard work type hierarchy. This work type is delivered before any of the work items described above.	n/a

Column Name	Mode	Description	Data Type
		The array number for MSGWORKNORMAL is 14.	
MSGWORKCONTROL	n/a	Used for system control messages. These are delivered before any other kind of message. The array number for MSGWORKCONTROL is 15.	n/a

Space Accounting Columns for ResUsageSawt Table

Previously, the Teradata accounting system assumed an even distribution of rows to AMPs. The database and user space limits were set at the AMP level to the defined maximum limit values divided by the number of AMPs in the system. Space was managed and monitored at the AMP level.

Some features now allow tables on subsets of AMPs. The space accounting and space management infrastructure now manages the space limits at a global or system level and augments the AMP-level space management. These enhancements are referred to as *global space accounting*.

These columns report statistics for space accounting.

Column Name	Mode	Description	Data Type
SpaceReactiveCnt	track	Number of AWTs waiting for space allocations as last tracked at the end of logging period.	FLOAT
SpaceReactiveWaitTime	count	Total time in milliseconds that different worker tasks were in reactive mode waiting on space allocations during the logging period.	FLOAT

Reserved Columns

Column Name	Mode	Description	Data Type
ReservedS0	n/a	Reserved for future use.	CHAR(7)

Spare Columns

The ResUsageSawt table spare fields are named Spare00 through Spare19, and SpareInt.

The SpareInt field has a 32-bit internal resolution while all other spare fields have a 64-bit internal resolution. All spare fields default to count but can be converted to min, max, or track mode fields if needed when they are used.

Column Name	Mode	Description	Data Type
Spare00 - Spare19, SpareInt	count	Reserved for future use.	FLOAT

Related Information

For details on the different type of data fields, see [About the Mode Column](#).

Summary Mode

When Summary Mode is active for the ResUsageSawt table, one row is written to the database for each node in the system for each log interval. The AMP Worker Task data will be combined for all the AMP vprocs on the node.

You can determine if a row is in Summary Mode by checking the SummaryFlag column for that row.

IF the SummaryFlag column value is...	THEN the data for that row is being logged...
'S'	in Summary Mode.
'N'	normally.

ResUsageShst Table

The ResUsageShst table:

- Contains resource usage information specific to the host channels and TCP/IP networks communicating with Vantage.
- Includes resource usage data for system-wide, host information.

Teradata recommends that you use ResShstView to access the data rather than accessing the ResUsageShst table directly. For more information, see [Resource Usage Views](#).

Note:

This table is created as a MULTiset table. For more information see [Relational Primary Index](#).

Housekeeping Columns

Relational Primary Index Columns

These columns taken together form the nonunique primary index.

Column Name	Mode	Description	Data Type
TheDate	n/a	Date of the log entry.	DATE
TheTime	n/a	Nominal time of the log entry. Note: Under conditions of heavy system load, entries may be logged late (typically, by no more than one or two seconds), but this column will still contain the time value when the entry <i>should have been</i> logged. For more information, see the Secs and NominalSecs columns.	FLOAT
NodeID	n/a	Node ID on which the entry resides. The Node ID is formatted as ZZZ9-9999, where ZZZ9 denotes the four-digit cabinet number and 9999 denotes the four-digit chassis number of the node. For example, a node in chassis 9 of cabinet 3 has a node ID of '3-0009'. Note: SMP nodes have a chassis and cabinet number of 1. For example, the node ID of an SMP node is '1-0001'.	INTEGER

Miscellaneous Housekeeping Columns

These columns provide statistics on current logging characteristics.

Column Name	Mode	Description	Data Type
GmtTime	n/a	Greenwich Mean Time is not affected by the Daylight Saving Time adjustments that occur twice a year.	FLOAT
CabinetID	n/a	The physical cabinet number of the node.	INTEGER
ModuleID	n/a	The physical module number of the node.	INTEGER
NodeType	n/a	Type of node, representing the per node system family type.	CHAR(8)
VprId	n/a	Identifies the vproc number. For channel-connected hosts, the VprId value for: <ul style="list-style-type: none"> • IBMECS is zero. • IBMMUX is zero. For TCP/IP network-connected hosts, the VprId value for: <ul style="list-style-type: none"> • NETWORK is the Gateway vproc ID. • IBMNET is zero. In Summary Mode, VprId is -1.	INTEGER
HstId	n/a	Identifies the host ID value. <ul style="list-style-type: none"> • For IBMECS, the value is in decimal format RRRRRAAAUUUNNN, where: RRRRR = laddr = 5 digit Link address (max value = 65535) AAA = cua = 3 digit Channel unit address (max value = 255) UUU = lcu = 3 digit Logical control unit address (max value = 255) NNN = lchan = 3 digit Logical channel number (max value = 255) • For IBMNET, the value is the host group ID. • For NETWORK and IBMMUX, the value is BBMMPPHHH, where: BB = Bus MM = Module Number (or chassis number) PP = Port HHH = Three digit host group ID Note: Each of the fields above get two or three decimal digits of the resulting nine digit value. Note: The chassis number is always zero for networks connected hosts. In Summary Mode, the value is always -1.	BIGINT
HstType	n/a	Type of host connection: <ul style="list-style-type: none"> • IBMECS (Part TCP/IP and channel hardware connection) • IBMMUX (Channel hardware connection) • IBMNET (TCP/IP end-to-end, also called "pure network, " connection) • NETWORK (Gateway channel connection) 	CHAR(8)

Column Name	Mode	Description	Data Type
IPAddr0, IPAddr1	n/a	<p>These columns identify the IP address of the ECS node for the channel-connected IBMECS host. For other hosts, the IP address value is zero.</p> <p>IPv4 has 4 byte addresses and IPv6 has 16 byte addresses. The first 8 bytes (0 - 7) are stored in IPAddr0, a 64-bit integer field. The second 8 bytes (8 - 16) are stored in IPAddr1, also a 64-bit integer field.</p> <p>If bytes 4 to 15 are zeros, then the IP address is IPv4.</p> <p>The ResShstView provides the following aliases for IP addressing.</p> <ul style="list-style-type: none"> • IPv4 - displays IPv4 addresses • IPv6 - displays IPv6 addresses • IPAddr - displays IP address in either IPv4 or IPv6 format, depending on the type of the address 	BIGINT
Secs	n/a	Actual number of seconds in the log period represented by this row. This value is useful for normalizing the statistics contained in this row, for example, to a per-second measurement.	SMALLINT
CentiSecs	n/a	Number of centiseconds in the logging period. This column is useful when performing data calculations with small elapsed times where the difference between centisecond-based data and whole seconds results in a percentage error.	INTEGER
NominalSecs	n/a	Specified or nominal number of seconds in the logging period.	SMALLINT
SummaryFlag	n/a	Summarization status of this row. If the value is 'N,' the row is a non-summary row. If the value is 'S,' the row is a summary row.	CHAR(1)
Active	max	<p>Controls whether or not the rows will be logged to the resource usage tables if Active Row Filter Mode is enabled.</p> <p>If Active is set to a non-zero value, the row contains data columns.</p> <p>If Active is set to a zero value, none of the data columns in the row have been updated during the logging period.</p> <p>For example, if you enable Active Row Filter Mode, the rows that have a zero Active column value will not be logged to the resource usage tables.</p>	FLOAT
TheTimestamp	n/a	<p>Number of seconds since midnight, January 1, 1970.</p> <p>This column is useful for aligning data with the DBQL log.</p>	BIGINT
PM_COD_CPU	n/a	<p>Platform Metering (PM) CPU Capacity On Demand (COD) value in one tenths of a percent. For example, a value of 500 represents a PM CPU COD value of 50.0%.</p> <p>The value is set to 1000 if the PM CPU COD is disabled.</p>	SMALLINT
WM_COD_CPU	n/a	<p>Workload Management (WM) CPU Capacity On Demand (COD) value in one tenths of a percent. For example, a value of 500 represents a WM CPU COD value of 50.0%.</p> <p>The value is set to 1000 if the WM CPU COD is disabled.</p>	SMALLINT

Statistics Columns

Host Controller: Channel Management Columns

These columns identify overhead of channel management.

Column Name	Mode	Description	Data Type
HostReadFails	count	Number of failures transmitting from the host. Note: This column is populated for Teradata Channel software (TCHN) only.	FLOAT
HostWriteFails	count	Number of failures transmitting to the host. Note: This column is populated for TCHN only.	FLOAT

Host Controller: Channel Traffic Columns

These columns identify the traffic between the host and the node in three levels of granularity:

- Blocks
- Messages
- KB

Blocks are made up of some amount of variable sized messages. ReadKB and WriteKB identify the KB involved in the traffic.

For information on the ResUsageSpma table channel and TCP/IP network traffic columns, see [Host Controller Channel and TCP/IP Network Traffic Columns](#).

Column Name	Mode	Description	Data Type
HostBlockReads	count	Number of blocks read in from the host.	FLOAT
HostBlockWrites	count	Number of blocks written out to the host.	FLOAT
HostMessageReads	count	Number of messages read in from the host.	FLOAT
HostMessageWrites	count	Number of messages written out to the host.	FLOAT
HostReadKB	count	KB transferred in from the host.	FLOAT
HostWriteKB	count	KB transferred out to the host.	FLOAT

User Command: User Command Columns

These columns identify the type of commands given to Vantage by the user. Three levels of granularity are given:

- Transactions. These columns consist of one or more requests.
- Requests. These consist of one or more statements.
- Statements. These columns are subdivided into the various statement types.

Column Name	Mode	Description	Data Type
CmdAlterStmts	count	Number of alter, modify, or drop statement commands.	FLOAT
CmdArchUtilityStmts	count	Number of archival utility commands (for example, restore, archive and recovery).	FLOAT
CmdCreateStmts	count	Number of create or replace statement commands.	FLOAT
CmdDeleteStmts	count	Number of delete commands.	FLOAT
CmdGrantStmts	count	Number of grant or revoke commands.	FLOAT
CmdInsertStmts	count	Number of insert commands.	FLOAT
CmdLoadUtilityStmts	count	Number of FastLoad and MultiLoad utility commands. (Tpump commands cannot be distinguished, and are therefore counted by the INSERT, UPDATE and DELETE statements).	FLOAT
CmdOtherStmts	count	Number of other commands.	FLOAT
CmdRequests	count	Number of request commands.	FLOAT
CmdSelectStmts	count	Number of select commands.	FLOAT
CmdTransactions	count	Number of transaction commands.	FLOAT
CmdUpdateStmts	count	Number of update commands.	FLOAT

User Command: User Command Arrival and Departure Columns

These columns identify the arrival and departure times and status of user commands.

Column Name	Mode	Description	Data Type
CmdStmtErrors	count	Number of statements that departed in error.	FLOAT
CmdStmtFailures	count	Number of statements that departed in failure or abortion.	FLOAT
CmdStmtSuccesses	count	Number of statements that departed normally.	FLOAT

Reserved Columns

Column Name	Mode	Description	Data Type
ReservedS0	n/a	Reserved for future use.	CHAR(3)
ReservedS1	n/a	Reserved for future use.	CHAR(4)

Spare Columns

The ResUsageShst table spare fields are named Spare00 through Spare09, and SpareInt.

The SpareInt field has a 32-bit internal resolution while all other spare fields have a 64-bit internal resolution. All spare fields default to count but can be converted to min, max, or track mode fields if needed when they are used.

Column Name	Mode	Description	Data Type
Spare00 - Spare09, SpareInt	count	Reserved for future use.	FLOAT

Related Information

For details on the different type of data fields, see [About the Mode Column](#).

Summary Mode

When Summary Mode is active for the ResUsageShst table, one row is written to the database for each type of host (network or channel-connected) on each node in the system, summarizing the hosts of that type on that node.

You can determine if a row is in Summary Mode by checking the SummaryFlag column for that row.

IF the SummaryFlag column value is...	THEN the data for that row is being logged...
'S'	in Summary Mode.
'N'	normally.

ResUsageSldv Table

The ResUsageSldv table contains resource usage information for system-wide, logical device information. Statistics from this table are collected from the storage devices.

ResUsageSldv can be used to monitor AMP response time and write I/O levels.

Teradata recommends that you use ResSldvView to access the data rather than accessing the ResUsageSldv table directly. For more information, see [Resource Usage Views](#).

Note:

This table is created as a MULTiset table. For more information see [Relational Primary Index](#).

Housekeeping Columns

Relational Primary Index Columns

These columns taken together form the nonunique primary index.

Column Name	Mode	Description	Data Type
TheDate	n/a	Date of the log entry.	DATE
TheTime	n/a	Nominal time of the log entry. Note: Under conditions of heavy system load, entries may be logged late (typically, by no more than one or two seconds), but this column will still contain the time value when the entry <i>should have been</i> logged. For more information, see the Secs and NominalSecs columns.	FLOAT
NodeID	n/a	Node ID on which the entry resides. The Node ID is formatted as ZZZ9-9999, where ZZZ9 denotes the four-digit cabinet number and 9999 denotes the four-digit chassis number of the node. For example, a node in chassis 9 of cabinet 3 has a node ID of '3-0009'. Note: SMP nodes have a chassis and cabinet number of 1. For example, the node ID of an SMP node is '1-0001'.	INTEGER

Miscellaneous Housekeeping Columns

These columns provide statistics on current logging characteristics and storage device information.

Column Name	Mode	Description	Data Type
GmtTime	n/a	Greenwich Mean Time is not affected by the Daylight Saving Time adjustments that occur twice a year.	FLOAT
CabinetID	n/a	The physical cabinet number of the node.	INTEGER
ModuleID	n/a	The physical module number of the node.	INTEGER
NodeType	n/a	Type of node, representing the per node system family type.	CHAR(8)
CtlId	n/a	Controller number. The value is the decimal equivalent of the three digit controller ID in the Ldvid (the HHC digits). If the controller information is not available, its value is set to -1. In Summary Mode, the CtlId is set to -1.	INTEGER
LdvName	n/a	Storage device name.	CHAR(8)
Ldvid	n/a	Storage device in the Bus System where it resides. If the device address information is available, the Ldvid is derived from the Host, Channel, Id, and Lun information of the device (00HHCTL). If the device address information is not available, this column contains the null ID which is 0xFFFFFFFF. In Summary Mode, the value is set to 0xFFFFFFFF.	BYTE(4)
LdvType	n/a	Type of logical device. The value is either DISK for database disk or SDSK for system disk.	CHAR(4)
LdvKind		Type of storage device. The storage device can be one of the following: <ul style="list-style-type: none"> • HDD: A hard disk device. • SSD: A solid-state device. • RSSD: A read intensive solid-state device. • WSSD: A write intensive solid-state device. • NETW: A network device. • LDSK: A local spool device. In Summary mode, the value is set to '---'.	CHAR(4)
Major	n/a	Major number of the device.	INTEGER
Minor	n/a	Minor number of the device.	INTEGER
Secs	n/a	Actual number of seconds in the log period represented by this row. Normally the same as NominalSecs, but can be different in three cases: <ul style="list-style-type: none"> • The first interval after a log rate change • A sample logged late because of load on the system • System clock adjustments affect reported Secs Useful for normalizing the count statistics contained in this row, for example, to a per-second measurement.	SMALLINT

Column Name	Mode	Description	Data Type
CentiSecs	n/a	Number of centiseconds in the logging period. This column is useful when performing data calculations with small elapsed times where the difference between centisecond-based data and whole seconds results in a percentage error.	INTEGER
NominalSecs	n/a	Specified or nominal number of seconds in the logging period.	SMALLINT
SummaryFlag	n/a	Summarization status of this row. If the value is 'N,' the row is a non-summary row. If the value is 'S,' the row is a summary row.	CHAR (1)
Active	max	Controls whether or not the rows will be logged to the resource usage tables if Active Row Filter Mode is enabled. If Active is set to a non-zero value, the row contains data columns. If Active is set to a zero value, none of the data columns in the row have been updated during the logging period. For example, if you enable Active Row Filter Mode, the rows that have a zero Active column value will not be logged to the resource usage tables.	FLOAT
TheTimestamp	n/a	Number of seconds since midnight, January 1, 1970. This column is useful for aligning data with the DBQL log.	BIGINT
PM_COD_CPU	n/a	Platform Metering (PM) CPU Capacity On Demand (COD) value in one tenths of percent. For example, a value of 500 represents a PM CPU COD value of 50%. The value is set to 1000 if PM CPU COD is disabled.	SMALLINT
PM_COD_IO	n/a	Platform Metering (PM) I/O Capacity On Demand (COD) value in whole percent. For example, a value of 50 represents a PM I/O COD value of 50.0%. The value is set to 100 if the PM I/O COD is disabled.	SMALLINT
WM_COD_CPU	n/a	Workload Management (WM) CPU Capacity On Demand (COD) value in one tenths of a percent. For example, a value of 500 represents a WM CPU COD value of 50.0%. The value is set to 1000 if the WM CPU COD is disabled.	SMALLINT
WM_COD_IO	n/a	Workload Management (WM) I/O Capacity On Demand (COD) value in whole percent. For example, a value of 50 represents a WM I/O COD value of 50.0%. The value is set to 100 if WM I/O COD is disabled.	SMALLINT
TIER_FACTOR	n/a	I/O performance limit placed on a core-reduced node. For example, a value of 75 represents an I/O limit of 75.0% placed before other COD values. The field is sourced from tpanodep->tierfactor, which is derived from the /proc/tdmeter/tier_performance file. A value of 100 indicates that there is no I/O performance limit applied to the node.	SMALLINT

Statistics Columns

I/O Hard Limits Columns

Column Name	Mode	Description	Data Type
IoThrottleCount	count	Number of times an I/O was throttled.	FLOAT
IoThrottleTime	count	Total I/O Throttle Time in centiseconds.	FLOAT
IoThrottleCntZerolotas	count	Number of times an I/O was throttled due to no IOTA tokens available in the bucket. This data is included in the IoThrottleCount.	FLOAT
IoThrottleCntInsufflotas	count	Number of times an I/O was throttled due to insufficient IOTA tokens available in the bucket. There were some IOTA tokens in the bucket, but not enough. This data is included in the IoThrottleCount.	FLOAT
IoThrottleCntInsufflotasHL	count	Number of times an I/O was throttled due to insufficient IOTA tokens available in the bucket for Hard Limits. There were some IOTA tokens in the bucket, but not enough. This data is included in the IoThrottleCount.	FLOAT
IoThrottleCntMaxQD	count	Number of times an I/O was throttled due to the queue depth to the storage being at maximum. This data is not part of the IoThrottleCount.	FLOAT

Logical Device Columns Introduction

These columns identify individual logical device activities for storage components connected through the buses.

The storage device statistics are calculated only on what can be derived from statistics collected by the operating system since the disk array controllers do not provide us with any useful data for resource usage.

The logical device columns are grouped into several subcategories as shown below.

Logical Device: Input and Output Traffic Columns

These columns represent the number and amount, in KB, of data read from or written to the logical device.

Column Name	Mode	Description	Data Type
LdvReadKB	count	Number of KB read from the logical device.	FLOAT
LdvReads	count	Number of reads issued.	FLOAT

Column Name	Mode	Description	Data Type
LdvWriteKB	count	Number of KB written to the logical device.	FLOAT
LdvWrites	count	Number of writes issued.	FLOAT

Logical Device: Response Time Columns

These columns represent the response time to requests given to the logical device.

Column Name	Mode	Description	Data Type
LdvReadRespTot	count	Total of individual read response times in centiseconds.	FLOAT
LdvWriteRespTot	count	Total of individual write response times in centiseconds.	FLOAT

Logical Device: Outstanding Requests Columns

These columns represent the number of outstanding operation requests and the amount of time with outstanding requests for the logical device.

Column Name	Mode	Description	Data Type
LdvOutReqDiv	count	Average value calculated by the ResSldvView view as shown below. $\text{LdvOutReqAvg} = \text{LdvOutReqSum} / \text{LdvOutReqDiv}$	FLOAT
LdvOutReqSum	count	Time-weighted total of samples in centiseconds spent doing I/O. LdvOutReqDiv contains the divisor, which is the delta time in centiseconds for the sample.	FLOAT
LdvOutReqTime	count	Total time in centiseconds with (any) outstanding requests. The values in this column should be less than or equal to the reported logging period.	FLOAT

Logical Device: I/O Hard Limits Columns

The Input/Output Token Allocations (IOTA) columns provide the percentage of disk throughput that is being used.

Column Name	Mode	Description	Data Type
FullPotentialIota	count	The full potential input or output token allocations for a device. The input or output token allocations are not limited by the IO_COD setting.	FLOAT

Column Name	Mode	Description	Data Type
		The data source for this field is located in the <code>cod_full_potential_iota</code> field in the <code>/proc/tdmeter/disk_cod_stats</code> file.	
CodPotentiallota	count	The potential input or output token allocations for a device. The input or output token allocations are accumulated only when an I/O is pending on a device, and they are limited by the <code>IO_COD</code> setting. The data source for this field is located in the <code>cod_allocated_iota</code> field in the <code>/proc/tdmeter/disk_cod_stats</code> file.	FLOAT
Usedlota	count	Used input or output token allocations for a device. The data source for this field is located in the <code>cod_used_iota</code> field in the <code>/proc/tdmeter/disk_cod_stats</code> file.	FLOAT

Reserved Columns

Column Name	Mode	Description	Data Type
ReservedS0	n/a	Reserved for future use.	CHAR(1)
ReservedS1	n/a	Reserved for future use.	CHAR(4)
ReservedS2	n/a	Reserved for future use.	CHAR(8)

Spare Columns

The ResUsageSldv table spare fields are named Spare00 through Spare09, and SpareInt.

The SpareInt field has a 32-bit internal resolution while all other spare fields have a 64-bit internal resolution. All spare fields default to count but can be converted to min, max, or track mode fields if needed when they are used.

Column Name	Mode	Description	Data Type
Spare00 - Spare09, SpareInt	count	Reserved for future use.	FLOAT

Related Information

For details on the different type of data fields, see [About the Mode Column](#).

Summary Mode

When Summary Mode is active for the ResUsageSldv table, the following rows are written to the database for each node in the system for each log interval:

- One row summarizes the system logical devices
- One row summarizes Vantage logical devices

You can determine if a row is in Summary Mode by checking the SummaryFlag column for that row.

IF the SummaryFlag column value is...	THEN the data for that row is being logged...
'S'	in Summary Mode.
'N'	normally.

ResUsageSmhm Table

This table is reserved for future use.

ResUsageSpdsk Table

The ResUsageSpdsk table:

- Provides pdisk level statistics.
- Includes resource usage logs on cylinder I/O, allocation, and migration.

Teradata recommends that you use ResSpdskView to access the data rather than accessing the ResUsageSpdsk table directly. For more information, see [Resource Usage Views](#).

Note:

This table is created as a MULTiset table. For more information see [Relational Primary Index](#).

Housekeeping Columns

Relational Primary Index Columns

These columns taken together form the nonunique primary index.

Column Name	Mode	Description	Data Type
TheDate	n/a	Date of the log entry.	DATE
TheTime	n/a	Nominal time of the log entry. Note: Under conditions of heavy system load, entries may be logged late (typically, by no more than one or two seconds), but this field will still contain the time value when the entry <i>should have been</i> logged. For more information, see the Secs and NominalSecs columns.	FLOAT
NodeID	n/a	Node ID on which the entry resides. The Node ID is formatted as ZZZ9-9999, where ZZZ9 denotes the four-digit cabinet number and 9999 denotes the four-digit chassis number of the node. For example, a node in chassis 9 of cabinet 3 has a node ID of '3-0009'. Note: SMP nodes have a chassis and cabinet number of 1. For example, the node ID of an SMP node is '1-0001'.	INTEGER

Miscellaneous Housekeeping Columns

These columns provide statistics on current logging characteristics.

Column Name	Mode	Description	Data Type
GmtTime	n/a	Greenwich Mean Time is not affected by the Daylight Saving Time adjustments that occur twice a year.	FLOAT
CabinetID	n/a	The physical cabinet number of the node.	INTEGER
ModuleID	n/a	The physical module number of the node.	INTEGER
ArrayName	n/a	PDISK storage array name. In Summary mode, the value is set to '---'.	CHAR(32)
PdiskGlobalId	n/a	Identifies the pdisk in the system. Each pdisk in the system has a global ID which uniquely identifies the pdisk in the system. If a pdisk is connected to the nodes in a clique, all the nodes in that clique see the same pdisk global ID associated with that pdisk. In Summary Mode, the pdisk global ID is -1.	INTEGER
PdiskType	n/a	Type of pdisk. The pdisk can be one of the following: <ul style="list-style-type: none"> • DISK: A storage device. • FILE: A file. • SSD: A solid-state device. • RSSD: A read intensive solid-state device. • WSSD: A write intensive solid-state device. • NETW: A network device. • LDSK: A local spool device. 	CHAR(4)
PdiskDeviceId	n/a	Identifies the local pdisk device. <ul style="list-style-type: none"> • For both DISK and SSD pdisks, the pdisk device ID can be a major/minor number. The major number bit positions are 20-31 and the minor number is in bits 0-19. The format is similar to the one shown below. (MMMM MMMM MMMM mmmm mmmm mmmm mmmm mmmm) • For FILE pdisk, the pdisk device ID is 0xFFFFFFFF. In Summary Mode, the pdisk device ID is 0xFFFFFFFF.	BYTE(4)
NodeType	n/a	Type of node, representing the per node system family type.	CHAR(8)
Secs	n/a	Actual number of seconds in the log period represented by this row. Normally the same as NominalSecs, but can be different in three cases: <ul style="list-style-type: none"> • The first interval after a log rate change • A sample logged late because of load on the system • System clock adjustments affect reported Secs Useful for normalizing the count statistics contained in this row, for example, to a per-second measurement.	SMALLINT
CentiSecs	n/a	Number of centiseconds in the logging period. This field is useful when performing data calculations with small elapsed times where the difference between centisecond-based data and whole seconds results in a percentage error.	INTEGER
NominalSecs	n/a	Specified or nominal number of seconds in the logging period.	SMALLINT

Column Name	Mode	Description	Data Type
SummaryFlag	n/a	Summarization status of this row. If the value is 'N,' the row is a non-summary row. If the value is 'S,' the row is a summary row.	CHAR(1)
Active	max	Controls whether or not the rows will be logged to the resource usage tables if Active Row Filter Mode is enabled. If Active is set to a non-zero value, the row contains data columns. If Active is set to a zero value, none of the data columns in the row have been updated during the logging period. For example, if you enable Active Row Filter Mode, the rows that have a zero Active column value will not be logged to the resource usage tables.	FLOAT
TheTimestamp	n/a	Number of seconds since midnight, January 1, 1970. This column is useful for aligning data with the DBQL log.	BIGINT
PM_COD_CPU	n/a	Platform Metering (PM) CPU Capacity On Demand (COD) value in one tenths of a percent. For example, a value of 500 represents a PM CPU COD value of 50.0%. The value is set to 1000 if the PM CPU COD is disabled.	SMALLINT
PM_COD_IO	n/a	Platform Metering (PM) I/O Capacity On Demand (COD) value in whole percent. For example, a value of 50 represents a PM I/O COD value of 50.0%. The value is set to 100 if the PM I/O COD is disabled.	SMALLINT
WM_COD_CPU	n/a	Workload Management (WM) CPU Capacity On Demand (COD) value in one tenths of a percent. For example, a value of 500 represents a WM CPU COD value of 50.0%. The value is set to 1000 if the WM CPU COD is disabled.	SMALLINT
WM_COD_IO	n/a	Workload Management (WM) I/O Capacity On Demand (COD) value in whole percent. For example, a value of 50 represents a WM I/O COD value of 50.0%. The value is set to 100 if the WM I/O COD is disabled.	SMALLINT
TIER_FACTOR	n/a	I/O performance limit placed on a core-reduced node. For example, a value of 75 represents an I/O limit of 75.0% placed before other COD values. The field is sourced from tpanodep->tierfactor which is derived from the /proc/tdmeter/tier_performance file. A value of 100 indicates that there is no I/O performance limit applied to the node.	SMALLINT

Statistics Columns

I/O Hard Limits Columns

Column Name	Mode	Description	Data Type
IoThrottleCount	count	Number of times an I/O was throttled.	FLOAT
IoThrottleTime	count	Total I/O Throttle Time in centiseconds.	FLOAT
IoThrottleTimeMax	max	Maximum I/O Throttle Time in centiseconds.	FLOAT
IoThrottleCntZerolotas	count	Number of times an I/O was throttled due to no IOTA tokens available in the bucket. This data is included in the IoThrottleCount.	FLOAT
IoThrottleCntInsufflotas	count	Number of times an I/O was throttled due to insufficient IOTA tokens available in the bucket. There were some IOTA tokens in the bucket, but not enough. This data is included in the IoThrottleCount.	FLOAT
IoThrottleCntInsufflotasHL	count	Number of times an I/O was throttled due to insufficient IOTA tokens available in the bucket for Hard Limits. There were some IOTA tokens in the bucket, but not enough. This data is included in the IoThrottleCount.	FLOAT
IoThrottleCntMaxQD	count	Number of times an I/O was throttled due to the queue depth to the storage being at maximum. This data is not part of IoThrottleCount.	FLOAT
RawReadRespTot	count	Total raw I/O Read Response Times in centiseconds. The raw I/O Response Time is the time from when TDSCHED releases the I/O for execution until when the I/O is completed.	FLOAT
RawWriteRespTot	count	Total raw I/O Write Response Times in centiseconds. The raw I/O Response Time is the time from when TDSCHED releases the I/O for execution until when the I/O is completed.	FLOAT
RawReadRespMax	max	Maximum raw I/O Read Response Time in centiseconds. The raw I/O Response Time is the time from when TDSCHED releases the I/O for execution until when the I/O is completed.	FLOAT
RawWriteRespMax	max	Maximum raw I/O Write Response Time in centiseconds.	FLOAT

Column Name	Mode	Description	Data Type
		The raw I/O Response Time is the time from when TDSCHED releases the I/O for execution until when the I/O is completed	

Teradata VS: Allocation Columns

These columns identify the allocation statistics reported by the Allocator process.

Column Name	Mode	Description	Data Type
ExtAllocHot	count	Number of hot allocations made in the current log period. A hot allocation is an allocation whose estimated temperature falls within the pre-defined hot temperature range. Each allocation is for a cylinder size worth of data. The cylinder resides in some disk storage location and holds some data. Temperature is the frequency of access to the data by I/O independent of where the data resides.	FLOAT
ExtAllocNonPacing	count	Number of non-pacing allocations made in the current log period. A non-pacing allocation is an allocation whose data access affects neither system performance nor individual query performance.	FLOAT
ExtAllocStatic	count	Number of static allocations made in the current log period. A static allocation is an allocation whose requested temperature is used and the measured temperature is ignored during migration.	FLOAT
ExtAllocSystemPacing	count	Number of system pacing allocations made in the current log period. A system pacing allocation is an allocation whose data access affects system performance.	FLOAT
ExtAllocTotal	count	Total number of allocations made in the current log period. A number of computations can be derived. For example: Cold Allocation = ExtAllocTotal -ExtAllocHot -ExtAllocWarm QueryPacing Allocation = ExtAllocTotal -ExtAllocNonPacing -ExtAllocSystemPacing Dynamic Allocation = ExtAllocTotal - ExtAllocStatic	FLOAT
ExtAllocWarm	count	Number of warm allocations made in the current log period. A warm allocation is an allocation whose estimated temperature falls within the pre-defined warm temperature range. Each allocation is for a cylinder size worth of data. The cylinder resides in some disk storage location and holds some data. Temperature is the frequency of access to the data by I/O independent of where the data resides.	FLOAT

Teradata VS: I/O Statistics Columns

Note:

You can populate the ResUsageSpdsk table I/O statistics columns without having a Teradata VS license.

These columns identify the I/O statistics reported by the Extent Driver.

Column Name	Mode	Description	Data Type
ConcurrentMax	count	Maximum number of concurrent I/O requests.	FLOAT
ConcurrentReadMax	max	Maximum number of concurrent read I/O requests.	FLOAT
ConcurrentWriteMax	max	Maximum number of concurrent write I/O requests.	FLOAT
MigrationBlockedIos	count	Number of inputs and outputs that are blocked due to migration request.	FLOAT
ReadCnt	count	Number of logical device reads.	FLOAT
ReadKB	count	Number of KB read from the logical device.	FLOAT
ReadRespMax	max	Maximum of individual read response time in centiseconds.	FLOAT
ReadRespSq	count	Total of squares of the individual read response time in centiseconds.	FLOAT
ReadRespTot	count	Total of individual read response time in centiseconds.	FLOAT
WriteCnt	count	Number of logical device writes.	FLOAT
WriteKB	count	Number of KB written to the logical device.	FLOAT
WriteRespMax	max	Maximum of individual write response time in centiseconds.	FLOAT
WriteRespSq	count	Total of squares of the individual write response time in centiseconds.	FLOAT
WriteRespTot	count	Total of individual write response time in centiseconds.	FLOAT

Teradata VS: Disk Cache Columns

These columns track the node local NVMe SSD disk cache usage.

Column Name	Mode	Description	Data Type
DiskCacheReadCnt	count	Total number of 4KB blocks read. This column is valid on SLES11 SP3 or later systems only.	FLOAT

Column Name	Mode	Description	Data Type
DiskCacheWriteCnt	count	Total number of 4KB blocks written. This column is valid on SLES11 SP3 or later systems only.	FLOAT
DiskCacheReadHitCnt	count	Number of 4KB blocks read from the disk cache. This column is valid on SLES11 SP3 or later systems only.	FLOAT
DiskCacheWriteHitCnt	count	Number of 4KB blocks written to the disk cache. This column is valid on SLES11 SP3 or later systems only.	FLOAT
DiskCacheFillCnt	count	Number of 4KB blocks written to the disk cache due to PDISK read operations. Total 4KB blocks written to cache = DiskCacheWriteHitCnt+DiskCacheFillCnt. This column is valid on SLES11 SP3 or later systems only.	FLOAT
DiskCacheInUseKB	track	Current KBs of disk cache in use. This column is valid on SLES11 SP3 or later systems only.	FLOAT
DiskCacheTotalKB	track	Total KBs of disk cache available for the pdisk. The value is 0 if disk cache is disabled or not used. This column is valid on SLES11 SP3 or later systems only.	FLOAT

Teradata VS: Migration Columns

Note:

The ResUsageSpdsk table migration columns are populated only in Teradata VS. For more information, see *Teradata Vantage™ - Teradata® Virtual Storage*, B035-1179.

The migration columns identify the number of cylinders that migrated to a different location on a device as well as the time, in centiseconds, of all migration I/Os used, incurred, or saved during the log period.

Note:

Each allocation is for a cylinder size worth of data, also known internally in the allocator as an extent. Thus the column names begin with *Ext* for extent.

Column Name	Mode	Description	Data Type
ExtMigrateFaster	count	Number of cylinders migrated to a faster location on a device. This count is for cylinders that were allocated on this device and migrated to a different location within the same device or migrated to a completely different device.	FLOAT

Column Name	Mode	Description	Data Type
		The following formula calculates a ExtMigrateSlower value, which is the number of cylinders migrated to slower locations: ExtMigrateSlower = ExMigrateTotal - ExMigrateFaster.	
ExtMigrateIOTimeBenefit	count	Estimates the total I/O time savings achieved by migrations completing in the log period. The I/O time savings include the improvement in response time caused by the new data arrangement up to the time horizon. ExtMigrateIOTimeBenefit does not include the cost of the migration I/Os and is a gross benefit, not a net benefit.	FLOAT
ExtMigrateIOTimeCost	count	Estimates the total cost, in centiseconds, incurred by migration I/Os completing during the log period, where cost is the extra time waited by all non-migration I/Os as a result of the migration I/O.	FLOAT
ExtMigrateIOTimeImprove	count	Estimates the percent improvement in average I/O response time due to migrations completing in the log interval. For example, if, right before a particular log interval, the average I/O response time was 10 milliseconds (ms), then the migration logs an ExtMigrateIOTimeImprove value of 10% in this interval. The average I/O response time after the log interval should be $(100\% - 10\%) * 10\text{ms} = 9\text{ms}$. Migration then logs an ExtMigrateIOTimeImprove of 1% in the next interval. The average I/O response time in the new log interval is $(100\% - 1\%) * 9\text{ms} = 8.91\text{ms}$. ExtMigrateIOTimeImprove is only an estimate. Its permanent improvement remains in effect as long as the workload does not change and newer migrations do not significantly alter the data arrangement. When the workload changes or new migrations affect data arrangement, response time changes in an unquantifiable way. Despite this, ExtMigrateIOTimeImprove is useful because it predicts actual system performance at least for short periods of time and can be used to understand why the migration algorithm is doing what it is doing.	FLOAT
ExtMigrateReadRespTot	count	Migration read I/O response time.	FLOAT
ExtMigrateWriteRespTot	count	Migration write I/O response time.	FLOAT
ExtMigrateTotal	count	Total number of cylinders migrated to a different physical location. For more information, see the ExtMigrateFaster field.	FLOAT

Reserved Columns

Column Name	Mode	Description	Data Type
ReservedS0	n/a	Reserved for future use.	CHAR(1)
ReservedS1	n/a	Reserved for future use.	CHAR(8)

Spare Columns

The ResUsageSpdsk table spare fields are named Spare00 through Spare19, and SpareInt.

The SpareInt field has a 32-bit internal resolution while all other spare fields have a 64-bit internal resolution. All spare fields default to count but can be converted to min, max, or track mode fields if needed when they are used.

Column Name	Mode	Description	Data Type
Spare00 - Spare06	count	Reserved for future use.	FLOAT
Spare07	count	Number of prioritized I/Os that were starved and had their priority updated to alleviate starvation. A prioritized I/O is one that Priority Scheduler issues to the Teradata I/O Scheduler (TDSCHED) Expedited or Normal priority RBTREE queues with a non-zero priority key that determines its position in the associated queue. One of these I/Os that is starved has waited in the TDSCHED queue longer than the Priority Scheduler I/O anti-starvation threshold, the value of which appears in "schmon -iodata -debug" output with the "iostarvedthreshold=" field. The permanent column name IoPrioStarvedCount is reserved for future use.	FLOAT
Spare08 - Spare19, SpareInt	count	Reserved for future use.	FLOAT

Related Information

For details on the different type of data fields, see [About the Mode Column](#).

Summary Mode

When Summary Mode is active for the ResUsageSpdsk table, rows are summarized into a single row for each pdisk type (for example, DISK, FILE, or SSD) for each node in the system per log interval.

You can determine if a row is in Summary Mode by checking the SummaryFlag column for that row.

IF the SummaryFlag column value is...	THEN the data for that row is being logged...
'S'	in Summary Mode.
'N'	normally.

ResUsageSps Table

You can use the ResUsageSps table to:

- Get a historical view of workload behavior for utilities and SQL operations.
- Determine the number of workload requests that are using AMP Worker Task. See the NumRequests column in [TASM: Monitor WD Columns](#) for details.
- Examine queue wait and service time numbers to find backed up queries and allocation groups.
- Determine which workload is responsible for I/O skew.
- Monitor CPU usage managed by the Priority Scheduler.
- Identify the percent of CPU being used by different workloads.

Teradata recommends that you use ResSpsView to access the data rather than accessing the ResUsageSps table directly. For more information, see [Resource Usage Views](#).

For a complete description of the SLES 11 Priority Scheduler and its components, see *Priority Scheduler for Linux SLES 11 Orange Book*, 541-0008867.

Each WD is equivalent of one Priority Scheduler workload definition ID (pWDId).

If table logging is enabled on ResUsageSps, a row is written to the database once for every pWDId and VprType in the system for each log interval.

For more information on the pWDId column, see the RowIndex1 column in ResUsageSps [Miscellaneous Housekeeping Columns](#).

Note:

This table is created as a MULTISSET table.

Housekeeping Columns

Relational Primary Index Columns

These columns taken together form the nonunique primary index.

Column Name	Mode	Description	Data Type
TheDate	n/a	Date of the log entry.	DATE
TheTime	n/a	Nominal time of the log entry.	FLOAT

Column Name	Mode	Description	Data Type
		Note: Under conditions of heavy system load, entries may be logged late (typically, by no more than one or two seconds), but this column will still contain the time value when the entry <i>should have been</i> logged. For more information, see the Secs and NominalSecs columns.	
NodeID	n/a	Node ID on which the entry resides. The Node ID is formatted as ZZZ9-9999, where ZZZ9 denotes the four-digit cabinet number and 9999 denotes the four-digit chassis number of the node. For example, a node in chassis 9 of cabinet 3 has a node ID of '3-0009'. Note: SMP nodes have a chassis and cabinet number of 1. For example, the node ID of an SMP node is '1-0001'.	INTEGER

Miscellaneous Housekeeping Columns

These columns provide statistics on current logging characteristics.

Column Name	Mode	Description	Data Type
GmtTime	n/a	Greenwich Mean Time is not affected by the Daylight Saving Time adjustments that occur twice a year.	FLOAT
CabinetID	n/a	The physical cabinet number of the node.	INTEGER
ModuleID	n/a	The physical module number of the node.	INTEGER
NodeType	n/a	Type of node, representing the per node system family type.	CHAR(8)
PPId	n/a	This column is not used and returns a zero value.	BYTEINT
VprType	n/a	Type of vproc (for example, AMP, PE, and MISC). Rows reported as vproc type of MISC contain data for all vproc types other than the AMP and PE vproc types.	CHAR(4)
NumAMPs	n/a	Number of AMPs on the Node. NumAMPs is used to divide columns that are reporting data from all the AMPs. This allows the ResSpsView view to report the data columns on a per AMP basis.	SMALLINT
RowIndex1	n/a	This column contains the pWDId. The pWDId value ranges from 0 to 255.	SMALLINT
WDId	track	WD ID. The default PGs (System, L, M, H, R) have no associated WD and will have a WDID of zero in the table. Only PGs with a name that has "TDWMWD" (SLES 11 or higher) in it have a nonzero WDID.	FLOAT

Column Name	Mode	Description	Data Type
		<p>Resource usages of the workload definition (WD) to which a job instance was assigned can be obtained by joining the DBC.DBQLUtilityTbl table and DBC.ResSPSView on the WDID column for the duration of the job instance.</p> <p>The example below calculates the sums of IOTA (I/O token allocation) columns.</p> <pre> SELECT SUM(FullPotentialIota), SUM(CodPotentialIota), SUM(UsedIota) FROM DBC.ResSpsView S, DBC.DBQLUtilityTbl U WHERE S.WDID = U.WDID AND U.UtilityName = 'MULTLOAD' AND U.UserName = 'USER1' AND U.JobEndTime = TIMESTAMP '2013-04-19 16: 48:14.98' AND S.TheDate >= CAST(Phase1StartTime AS DATE) AND S.TheTime >= ((EXTRACT(HOUR FROM Phase1StartTime) * 10000) + (EXTRACT(MINUTE FROM Phase1StartTime) * 100) + (EXTRACT(SECOND FROM Phase1StartTime))) AND S.TheDate <= CAST(Phase2EndTime AS DATE) AND S.TheTime <= ((EXTRACT(HOUR FROM Phase2EndTime) * 10000) + (EXTRACT(MINUTE FROM Phase2EndTime) * 100) + (EXTRACT(SECOND FROM Phase2EndTime))); </pre>	
Secs	n/a	<p>Actual number of seconds in the log period represented by this row. Normally, this value is the same as NominalSecs, but can be different in three cases:</p> <ul style="list-style-type: none"> • The first interval after a log rate change • A sample logged late because of load on the system • System clock adjustments affect reported Secs <p>Useful for normalizing the count statistics contained in this row, for example, to a per-second measurement.</p>	SMALLINT
CentiSecs	n/a	<p>Number of centiseconds in the logging period. This column is useful when performing data calculations with small elapsed times where the difference between centisecond-based data and whole seconds results in a percentage error.</p>	INTEGER
NominalSecs	n/a	<p>Specified or nominal number of seconds in the logging period.</p>	SMALLINT
NCPUs	n/a	<p>Number of online CPUs on this node.</p> <p>This column is useful for normalizing the CPU utilization column values for the number of CPUs on the node. This is especially important in:</p> <ul style="list-style-type: none"> • Coexistence systems where the number of CPUs can vary across system nodes. 	SMALLINT

Column Name	Mode	Description	Data Type
		<ul style="list-style-type: none"> Elastic TCore systems where the number of online CPUs can change without database restart. 	
Active	max	<p>Controls whether or not the rows will be logged to the resource usage tables when Active Row Filter Mode is enabled.</p> <p>If Active is set to a non-zero value, the row contains data columns.</p> <p>If Active is set to a zero value, none of the data columns in the row have been updated during the logging period.</p>	FLOAT
TheTimestamp	n/a	<p>Number of seconds since midnight, January 1, 1970.</p> <p>This column is useful for aligning data with the DBQL log.</p>	BIGINT
PM_COD_CPU	n/a	<p>Platform Metering (PM) CPU COD value in one tenths of percent. For example, a value of 500 represents a PM CPU COD value of 50%.</p> <p>The value is set to 1000 if the PM CPU COD is disabled.</p>	SMALLINT
PM_COD_IO	n/a	<p>Platform Metering (PM) I/O Capacity On Demand (COD) value in whole percent values for the entire system. For example, a value of 50 represents a PM I/O COD value of 50%.</p> <p>The value is set to 100 if the PM I/O COD is disabled.</p>	SMALLINT
WM_COD_CPU	n/a	<p>Workload Management (WM) CPU Capacity On Demand (COD) value in one tenths of a percent. For example, a value of 500 represents a WM CPU COD value of 50.0%.</p> <p>The value is set to 1000 if the WM CPU COD is disabled.</p>	SMALLINT
WM_COD_IO	n/a	<p>Workload Management (WM) I/O Capacity On Demand (COD) value in whole percent. For example, a value of 50 represents a WM I/O COD value of 50.0%.</p> <p>The value is set to 100 if the WM I/O COD is disabled.</p>	SMALLINT
TIER_FACTOR	n/a	<p>I/O performance limit placed on a core-reduced node. For example, a value of 75 represents an I/O limit of 75.0% placed before other COD values</p> <p>The field is sourced from tpanodep->tierfactor which is derived from the /proc/tdmeter/tier_performance file. A value of 100 indicates that there is no I/O performance limit applied to the node.</p>	SMALLINT
VPIId	track	<p>Virtual partition ID.</p> <p>Only one VpId is associated with a pWDId and VprType row at any point in time. There can be multiple pWDId values associated with a VPIId.</p>	FLOAT

Statistics Columns

TASM: Monitor WD Columns

The Monitor WD columns provide RSS data to the following System PMPC APIs:

- MONITOR WD request
- MonitorWD function

For more information on these APIs, see *Teradata Vantage™ - Application Programming Reference*, B035-1090.

The Control AMP WorkTimeInuse and WorkTimeInuseMax columns described in the following table exclude uncompleted work time for the Control GDO AMP during recovery work. This will occur for rows with the RowIndex1 column equal to zero, but will typically not be noticed. Work time for recovery work that completes during the reporting period is reported for the Control AMP.

For more information on the RowIndex1 column, see [Miscellaneous Housekeeping Columns](#).

Column Name	Mode	Description	Data Type
ActiveSessions	track	Number of request-level workload management objects created. There is one object created per user request. For non-user work request, there is one object created per WD.	FLOAT
CPURunDelay	count	<p>Number of milliseconds tasks in the WD sat in the CPU run queue waiting to run over the reporting period. Tasks may wait for access to a CPU regardless of the presence of COD or VP/WD Hard Limits. In addition, Tactical work is not immune to delayed access to CPU because the number of tasks running on a system often outnumbers the number of available CPUs. Delay can also be attributed to the running of tasks in higher priority System and Internal WDs, context switching, or excessive task activity in a WD.</p> <p>This data can be used in determining demand for the virtual partition and WD Share workload management method. If the CPU and I/O percentages for a virtual partition or WD are below their relative share values and the CPURunDelay values are low, there was insufficient demand to meet the share percentage. If the CPURunDelay values are high, higher tier SQL requests were allocated more resources so that there were insufficient resources remaining to allocate to SQL requests in this WD to meet its relative share. It is more relevant and helpful to look at this value in relation to other WDs as opposed to looking at the value from one WD alone.</p> <p>For more information on WD share workload management method, see <i>Teradata® Viewpoint User Guide</i>, B035-2206.</p>	FLOAT

Column Name	Mode	Description	Data Type
DecayLevel1IO	count	<p>Number of times SQL requests in the WD hit decay level 1 due to I/O.</p> <p>The values are reported under VprType MISC because the summed usage from PE and AMP go toward the usage that trips the decay. The values reported from either PE or AMP usage alone would not be accurate.</p> <p>Note: DecayLevel1IO is used for Timeshare WDs only. For more information on WD share workload management method, see <i>Teradata® Viewpoint User Guide</i>, B035-2206.</p>	FLOAT
DecayLevel2IO	count	<p>Number of times SQL requests in the WD hit decay level 2 due to I/O.</p> <p>The values are reported under VprType MISC because the summed usage from PE and AMP go toward the usage that trips the decay. The values reported from either PE or AMP usage alone would not be accurate.</p> <p>Note: DecayLevel2IO is used for Timeshare WDs only. For more information on workload management methods, see <i>Teradata® Viewpoint User Guide</i>, B035-2206.</p>	FLOAT
DecayLevel1CPU	count	<p>Number of times SQL requests in the WD hit decay level 1 due to CPU.</p> <p>The values are reported under VprType MISC because the summed usage from PE and AMP go toward the usage that trips the decay. The values reported from either PE or AMP usage alone would not be accurate.</p> <p>Note: DecayLevel1CPU is used for Timeshare WDs only. For more information on workload management methods, see <i>Teradata® Viewpoint User Guide</i>, B035-2206.</p>	FLOAT
DecayLevel2CPU	count	<p>Number of times SQL requests in the WD hit decay level 2 due to CPU.</p> <p>The values are reported under VprType MISC because the summed usage from PE and AMP go toward the usage that trips the decay. The values reported from either PE or AMP usage alone would not be accurate.</p> <p>Note: DecayLevel2CPU is used for Timeshare WDs only. For more information on workload management methods, see <i>Teradata® Viewpoint User Guide</i>, B035-2206.</p>	FLOAT

Column Name	Mode	Description	Data Type
IOCompleted	count	Number of AgeOut Now data blocks not to keep in memory (fsgcache) and to be written to disk.	FLOAT
IOCompletedKB	count	KB of AgeOut Now data blocks not to keep in memory (fsgcache) and to be written to disk.	FLOAT
IOCriticalSubmitted	count	Number of I/Os submitted with a MI Lock Priority. These I/Os go into a Teradata Scheduler queue and are not prioritized based on the type of management method.	FLOAT
IOCriticalSubmittedKB	count	KB of I/O submitted with an MI Lock Priority. These I/Os go into a Teradata Scheduler queue and are not prioritized based on the type of management method.	FLOAT
IOSubmitted	count	Number of I/Os submitted on behalf of this WD.	FLOAT
IOSubmittedKB	count	KB of I/O submitted on behalf of this WD.	FLOAT
NumRequests	count	Number of AMP Worker Task messages or requests that got assigned AMP Worker Tasks to them.	FLOAT
NumTasks	track	<p>Average number of tasks of online nodes. The column is the result of:</p> $\text{NumTasks} = \text{SUM of } (\text{NumTasks-}i) / N$ <p>where:</p> <ul style="list-style-type: none"> • <i>NumTasks</i> is the number of tasks assigned to the WD at the end of the reporting period. • <i>i</i> varies from 1 to <i>N</i>. • <i>N</i> is the number of online nodes. 	FLOAT
QWaitTime	count	<p>Total time in milliseconds that work requests waited on an input queue before being serviced. If the work requests are not delivered, they are not counted.</p> <p>To calculate the average QWaitTime per request, divide by NumRequests, which is reported in DBC.ResSpsView as QWaitTimeRequestAvg.</p>	FLOAT
QWaitTimeMax	max	Maximum time in milliseconds that work requests waited on an input queue before being serviced.	FLOAT
ServiceTime	count	<p>Time in milliseconds that work requests required for service.</p> <p>To calculate an approximate average ServiceTime for each request during this period, divide the ServiceTime value by the NumRequests value, which is reported in DBC.ResSpsView as ServiceTimeRequestAvg.</p> <p>The service time is the elapsed time from the time the message was received to the time the AMP Worker Task was released. This is the amount of time the AMP Worker Task was held through sleeps, CPU, I/O, and so on until it is released.</p>	FLOAT

Column Name	Mode	Description	Data Type
ServiceTimeMax	max	Maximum time in milliseconds that work requests required for service.	FLOAT
TacticalExceptionIO	count	<p>Number of times SQL requests in the WD hit a tactical per-node exception due to I/O.</p> <p>The values are reported under VprType MISC because the summed usage from PE and AMP go toward the usage that trips the exception. The value reported from either PE or AMP usage alone would not be accurate.</p> <p>An exception, used only for Tactical WDs, is created for each Tactical WD. For more information on workload management methods, see <i>Teradata® Viewpoint User Guide</i>, B035-2206.</p>	FLOAT
TacticalExceptionCPU	count	<p>Number of times SQL requests in the WD hit a tactical per-node exception due to CPU.</p> <p>The values are reported under VprType MISC because the summed usage from PE and AMP go toward the usage that trips the exception. The value reported from either PE or AMP usage alone would not be accurate.</p> <p>Note: TacticalExceptionCPU is used for Tactical WDs only. For more information on workload management methods, see <i>Teradata® Viewpoint User Guide</i>, B035-2206.</p>	FLOAT
WaitIO	count	<p>Number of milliseconds tasks in WD waited for I/O over the reporting period.</p> <p>WaitIO is updated when the wait for I/O is completed.</p>	FLOAT
WorkMsgReceiveDelay	track	<p>Time for all messages not yet delivered at the end of each reporting period. This column is related to the QWaitTime column and represents a running total of delays attributed to the tasks that still have not been assigned an AMP Worker Task within this interval.</p> <p>When the task does receive an AMP Worker Task in a later interval, the time attributed here is counted again within QWaitTime of the interval where it was assigned an AMP Worker Task.</p>	FLOAT
WorkMsgReceiveDelayCnt	track	Number of messages that are still waiting for AMP Worker Tasks at the end of each reporting period.	FLOAT
WorkMsgReceiveDelayMax	track	Maximum delay time in milliseconds for messages that are still in the work box at the end of each reporting period.	FLOAT
WorkMsgSendDelay	count	Total time in milliseconds it takes to deliver work messages.	FLOAT

Column Name	Mode	Description	Data Type
WorkMsgSendDelayCnt	count	Number of messages that are delivered to the work box.	FLOAT
WorkMsgSendDelayMax	max	Maximum time in milliseconds that it takes to deliver any single work message.	FLOAT
WorkTimeInuse	count	Service time consumed by a WD during the current reporting period. Note: This is not the running sum of a WD that exists over multiple intervals.	FLOAT
WorkTimeInuseMax	max	Maximum service time of a single task in a WD that is running or has finished in the current reporting period. This includes time used during previous intervals for that task.	FLOAT

TASM: AMP Worker Task Columns

These columns report statistics about the AMP Worker Tasks. For more information about the ResUsageSawt table and columns, see [ResUsageSawt Table](#).

The data is reporting the contribution of the respective WD to the column and the values are not the same as the values reported in the ResUsageSawt table. The ResUsageSps table values should add up to the ResUsageSawt table for columns like WorkTypeInuse. The Max columns will not be able to be correlated to the ResUsageSawt table Max values in such a direct way since the ResUsageSps Max columns report the Max value of the ResUsageSps table InUse column for the WD and not the Max value of the ResUsageSawt table for all the WDs combined.

Column Name	Mode	Description	Data Type
AwtReleases	count	Number of AMP Worker Tasks released (that is, completed requests) while the NumRequests column reports the number of AMP Worker Task requests that arrived. For details, see NumRequests .	FLOAT
WorkTypeInuse00 -WorkTypeInuse15	track	Current number of AMP Worker Tasks in use for each work type.	FLOAT
WorkTypeMax00 - WorkTypeMax15	max	Maximum of the WorkTypeInuse values reported during the logging period. When multiple data samples occur during the reporting period the value is the maximum of the sampled values. The true maximum number of in-use AMP Worker Tasks of a WorkType may occur at a different time	FLOAT

Column Name	Mode	Description	Data Type
		during the reporting period and not be seen and, therefore, not be reported.	
WorkTypeAWTEXhausted00 - WorkTypeAWTEXhausted15	count	Number of times a work message cannot be assigned to an AWT. This could be caused by no AWTs available in the available pool, or by the number of AWTs in use for the work type reaching its limits.	FLOAT

Process Scheduling: CPU Utilization Columns

These columns represent CPU activities on the node associated with the AMP Worker Task, Dispatcher, Parser, or miscellaneous activities.

Note:

When an external routine (such as C, C++, Java UDF, or external stored procedure) forks a child process or thread, the CPU time is not reported to these fields. As a result, the resource usage table shows a lower CPU usage than shown in the ResUsageSpma table even if the external routine consumes a large amount of CPU time. If there are child processes or threads running on your system, this may account for the larger CPU times reported in the ResUsageSpma table compared to this table. To confirm that the difference in CPU usage reported by the ResUsageSpma table is caused by child processes or threads, contact your Teradata Support Center personnel.

Column Name	Mode	Description	Data Type
CPUUServ	count	CPU time in milliseconds spent by AMP Worker Task or Dispatcher/Parser executing user service code. If VprType is AMP, the field contains the system-level AMP worker task CPU usage. If VprType is PE, the field contains the system-level Dispatcher/Parser CPU usage.	FLOAT
CPUUServOther	count	Time in milliseconds CPUs are busy executing miscellaneous activities for user service code. This is the system-level time spent on a process.	FLOAT
CPUUExec	count	CPU time in milliseconds spent by AMP Worker Task or Dispatcher/Parser executing user execution code. If VprType is AMP, the field contains the user-level AMP worker task CPU usage. If VprType is PE, the field contains the user-level Dispatcher/Parser CPU usage.	FLOAT
CPUUExecOther	count	Time in milliseconds CPUs are busy executing miscellaneous activities for user execution code. This is the user-level time spent on a process.	FLOAT

Process Scheduling: CPU Hard Limits Columns

Column Name	Mode	Description	Data Type
CpuVpThrottleCount	count	Number of times that Virtual Partition (VP) hard limits throttling was triggered.	FLOAT
CpuVpThrottleTime	count	Time in milliseconds that VP hard limits throttling was active.	FLOAT
CpuThrottleCount	count	Number of times that WD hard limits throttling was triggered.	FLOAT
CpuThrottleTime	count	Time in milliseconds that WD hard limits throttling was active.	FLOAT

Process Scheduling: Process Block Counts Columns

These columns identify how many times a process became blocked.

For more information about process block counts columns, see [ResUsageSpsma Table](#)

Column Name	Mode	Description	Data Type
ProcBlksDBLock	count	Number of times processes were blocked for a database lock.	FLOAT
ProcBlksFsgLock	count	Number of times processes were blocked for an FSG lock.	FLOAT
ProcBlksSegLock	count	Number of times processes were blocked for a disk or task context (for example, scratch, stack, and so on).	FLOAT
ProcBlksTime	count	Number of times processes were blocked for a timer expiration.	FLOAT
ProcBlksFlowControl	count	Number of times processes were blocked by delays caused by the flow control conditions.	FLOAT
ProcBlksFsgNIOs	count	Number of times processes were blocked waiting for task I/Os to complete.	FLOAT
ProcBlksFsgRead	count	Number of times processes were blocked for an FSG read.	FLOAT
ProcBlksFsgWrite	count	Number of times processes were blocked for an FSG write from disk.	FLOAT
ProcBlksMisc	count	Number of times processes were blocked for miscellaneous events.	FLOAT
ProcBlksMonitor	count	Number of times processes were blocked for a user monitor.	FLOAT

Column Name	Mode	Description	Data Type
ProcBlksMonResume	count	Number of times processes were blocked for a user monitor resume from a yield.	FLOAT
ProcBlksNetThrottle	count	Number of times processes were blocked for delivery of outstanding outgoing messages.	FLOAT
ProcBlksCpuThrottle	count	Number of times processes were throttled due to WM CPU COD or CPU hard limits.	FLOAT
ProcBlksSegMDL	count	Number of times processes were blocked waiting for an MDL resource to become available. An MDL is an internal PDE data structure needed by the operation of the segment subsystem.	FLOAT
ProcBlksSegNoVirtual	count	Number of times processes were blocked waiting for virtual memory for a segment.	FLOAT
ProcBlksQnl	count	Number of times processes were blocked for a TSKQNL lock.	FLOAT

Process Scheduling: Process Pending Wait Time Columns

These columns identify the how long the processes were blocked for each possible reason listed below.

The following column definition descriptions can also be found in [Process Scheduling: Process Pending Wait Time Columns](#) of ResUsageSpma table.

Column Name	Mode	Description	Data Type
ProcWaitFlowControl	count	Total time in milliseconds processes were blocked pending the delays caused by flow control conditions.	FLOAT
ProcWaitFsgLock	count	Total time in milliseconds processes were blocked pending an FSG lock.	FLOAT
ProcWaitDBLock	count	Total time in milliseconds processes were blocked pending a database lock.	FLOAT
ProcWaitSegLock	count	Total time in milliseconds processes were blocked pending a disk or task context (for example, scratch, stack, and so on) lock.	FLOAT
ProcWaitFsgRead	count	Total time in milliseconds processes were blocked pending an FSG read from disk.	FLOAT
ProcWaitFsgWrite	count	Total time in milliseconds processes were blocked pending an FSG write from disk.	FLOAT
ProcWaitFsgNIOs	count	Total time in milliseconds processes were blocked waiting for task I/Os to complete.	FLOAT

Column Name	Mode	Description	Data Type
ProcWaitMisc	count	Total time in milliseconds processes were blocked pending miscellaneous events.	FLOAT
ProcWaitMonitor	count	Total time in milliseconds processes were blocked pending a user monitor.	FLOAT
ProcWaitMonResume	count	Total time in milliseconds processes were blocked pending a user monitor resume from a yield.	FLOAT
ProcWaitNetThrottle	count	Total time in milliseconds processes were blocked pending delivery of outstanding outgoing messages.	FLOAT
ProcWaitCpuThrottle	count	Time in milliseconds processes were throttled due to WM CPU COD or CPU hard limits.	FLOAT
ProcWaitSegMDL	count	Total time in milliseconds processes were blocked waiting for an MDL resource to become available. An MDL is an internal PDE data structure needed by the operation of the segment subsystem.	FLOAT
ProcWaitSegNoVirtual	count	Total time in milliseconds processes were blocked waiting for virtual memory for a segment.	FLOAT
ProcWaitTime	count	Total time in milliseconds processes were blocked pending some amount of elapsed time only.	FLOAT
ProcWaitQnl	count	Total time in milliseconds processes were blocked pending an TSKQNL lock.	FLOAT

Process Scheduling: UDF CPU Utilization Columns

Column Name	Mode	Description	Data Type
UdfServ	count	System level UDF CPU time in milli-seconds, excluding the CPU time of the STO child processes.	FLOAT
UdfExec	count	User level UDF CPU time in milli-seconds, excluding the CPU time of the STO child processes.	FLOAT
UdfServOther	count	System level UDF CPU time in milli-seconds of other UDF processes such as the STO child processes.	FLOAT
UdfExecOther	count	User level UDF CPU time in milli-seconds of other UDF processes such as the STO child processes.	FLOAT

Real Time Boosting

Several new metrics were added to the SPS table to track the Real Time Boosting statistics.

Column Name	Mode	Description	Data Type
RTBoostCount	count	Number of tasks boosted to real time priority.	FLOAT
RTCpuTime	count	Total CPU time in milliseconds consumed by boosted realtime tasks. Real time tasks CPU time is not limited by COD or Hard Limit.	FLOAT
RTTasksMax	max	Maximum number of tasks running in boosted realtime priority.	FLOAT

File System: Segments Acquired Columns

These columns identify the total disk memory segments acquired by the File System during the log period. Logical acquires (Acqs) and the logical amount acquired (AcqKB) are identified. Acquires causing physical reads (AcqReads) and the amount read (AcqReadKB) are identified as a subset of logical acquires.

Column Name	Mode	Description	Data Type
FileAPtAcqs	count	Total number of append table or permanent journal table data block or cylinder index disk segments acquired.	FLOAT
FilePCiAcqs	count	Total number of permanent cylinder index disk segments acquired.	FLOAT
FilePDbAcqs	count	Total number of permanent data block disk segments acquired.	FLOAT
FileSCiAcqs	count	Total number of regular or restartable spool index disk segments acquired.	FLOAT
FileSDbAcqs	count	Total number of regular or restartable spool data block disk segments acquired.	FLOAT
FileTJtAcqs	count	Total number of transient journal table disk segments acquired.	FLOAT
FileAPtAcqKB	count	Total KB acquired by FileAPtAcqs.	FLOAT
FilePCiAcqKB	count	Total KB acquired by FilePCiAcqs.	FLOAT
FilePDbAcqKB	count	Total KB acquired by FilePDbAcqs.	FLOAT
FileSCiAcqKB	count	Total KB acquired by FileSCiAcqs.	FLOAT
FileSDbAcqKB	count	Total KB acquired by FileSDbAcqs.	FLOAT
FileTJtAcqKB	count	Total KB acquired by FileTJtAcqs.	FLOAT
FileAPtAcqReads	count	Number of append table or permanent journal table data block or cylinder index disk segments that caused a physical read.	FLOAT
FilePCiAcqReads	count	Number of permanent cylinder index disk segment acquires that caused a physical read.	FLOAT

Column Name	Mode	Description	Data Type
FilePDbAcqReads	count	Number of permanent data block disk segment acquires that caused a physical read.	FLOAT
FileSCiAcqReads	count	Number of regular or restartable spool index disk segments that caused a physical read.	FLOAT
FileSDbAcqReads	count	Number of regular or restartable spool data block disk segments that caused a physical read.	FLOAT
FileTJtAcqReads	count	Number of transient journal table disk segments that caused a physical read.	FLOAT
FileAPtAcqReadKB	count	KB physically read by FileAPtAcqReads.	FLOAT
FilePCiAcqReadKB	count	KB physically read by FilePCiAcqReads.	FLOAT
FilePDbAcqReadKB	count	KB physically read by FilePDbAcqReads.	FLOAT
FileSCiAcqReadKB	count	KB physically read by FileSCiAcqReads.	FLOAT
FileSDbAcqReadKB	count	KB physically read by FileSDbAcqReads.	FLOAT
FileTJtAcqReadKB	count	KB physically read by FileTJtAcqReads.	FLOAT

File System: Data Block Prefetch Columns

These columns identify File Segment Prefetch activities.

Note:

A prefetch is either a cylinder read operation or an individual block read operation. Either of these operations are generically called a prefetch.

Column Name	Mode	Description	Data Type
FileAPtPres	count	Total number of append table or permanent journal table data block or cylinder index disk segments prefetched.	FLOAT
FilePCiPres	count	Total number of permanent cylinder index disk segments prefetched.	FLOAT
FilePDbPres	count	Total number of permanent data block disk segments prefetched.	FLOAT
FileSCiPres	count	Total number of regular or restartable spool index disk segments prefetched.	FLOAT
FileSDbPres	count	Total number of regular or restartable spool data block disk segments prefetched.	FLOAT
FileTJtPres	count	Total number of transient journal table disk segments prefetched.	FLOAT

Column Name	Mode	Description	Data Type
FileAPtPresKB	count	Total number of KB prefetched by FileAPtPres.	FLOAT
FilePCiPresKB	count	Total number of KB prefetched by FilePCiPres.	FLOAT
FilePDbPresKB	count	Total number of KB prefetched by FilePDbPres.	FLOAT
FileSCiPresKB	count	Total number of KB prefetched by FileSCiPres.	FLOAT
FileSDbPresKB	count	Total number of KB prefetched by FileSDbPres.	FLOAT
FileTJtPresKB	count	Total number of KB prefetched by FileTJtPres.	FLOAT
FileAPtPreReads	count	Total number of append table or permanent journal table data block or cylinder index disk segment prefetches that caused a physical read.	FLOAT
FilePCiPreReads	count	Total number of permanent cylinder index disk segment prefetches that caused a physical read.	FLOAT
FilePDbPreReads	count	Total number of permanent data block disk segment prefetches that caused a physical read.	FLOAT
FileSCiPreReads	count	Total number of regular or restartable spool index disk segment prefetches that caused a physical read.	FLOAT
FileSDbPreReads	count	Total number of regular or restartable spool data block disk segment prefetches that caused a physical read.	FLOAT
FileTJtPreReads	count	Total number of transient journal table disk segment prefetches that caused a physical read.	FLOAT
FileAPtPreReadKB	count	Total number of KB physical read by FileAPtPreReads.	FLOAT
FilePCiPreReadKB	count	Total number of KB physical read by FilePCiPreReads.	FLOAT
FilePDbPreReadKB	count	Total number of KB physically read by FilePDbPreReads.	FLOAT
FileSCiPreReadKB	count	Total number of KB physical read by FileSCiPreReads.	FLOAT
FileSDbPreReadKB	count	Total number of KB physical read by FileSDbPreReads.	FLOAT
FileTJtPreReadKB	count	Total number of KB physical read by FileTJtPreReads.	FLOAT

File System: Segments Released Columns

These columns identify the total disk memory segments released by the File System, as well as those segments that are dropped from memory during the log period. When a segment is released, the segment is either:

- Force out of memory (F)
- Remains resident in memory (R)
- Aged out of memory (A), from segments that remain resident

Both the number of segments (Rels, Writes, Drps) and the size of the segments (RelKB, WriteKB, DrpKB) are counted. When a segment leaves memory, it must be written to disk only if the segment is dirty, that is, modified (Dy). Otherwise, the clean or unmodified (Cn) segment is simply dropped.

Most spool blocks are simply dropped from a task and put on the age queue. This may happen multiple times. Each of these will be counted as a resident release. If the system is low on memory and the age queue must be processed, this may also result in an age write or age drop. Forced writes are always also counted as either clean resident releases or forced drops, depending on whether age normal or age out now was specified.

Column Name	Mode	Description	Data Type
FileAPtDyRRels	count	Number of dirty append table and permanent journal data block or cylinder index disk segment resident releases.	FLOAT
FilePCiDyRRels	count	Number of dirty permanent cylinder index disk segment resident releases.	FLOAT
FilePDbDyRRels	count	Number of dirty permanent data block disk segment resident releases.	FLOAT
FileSCiDyRRels	count	Number of dirty regular or restartable spool cylinder index disk segment resident releases.	FLOAT
FileSDbDyRRels	count	Number of dirty regular or restartable spool data block disk segment resident releases.	FLOAT
FileTJtDyRRels	count	Number of dirty transient journal table or WAL data block or WAL cylinder index disk segment resident releases.	FLOAT
FileAPtDyRRelKB	count	KB released by FileAPtDyRRels.	FLOAT
FilePCiDyRRelKB	count	KB released by FilePCiDyRRels.	FLOAT
FilePDbDyRRelKB	count	KB released by FilePDbDyRRels.	FLOAT
FileSCiDyRRelKB	count	KB released by FileSCiDyRRels.	FLOAT
FileSDbDyRRelKB	count	KB released by FileSDbDyRRels.	FLOAT
FileTJtDyRRelKB	count	KB released by FileTJtDyRRels.	FLOAT
FileAPtFWrites	count	Number of append table and permanent journal data block or cylinder index disk segment forced releases or specific I/O requests causing an immediate physical write.	FLOAT
FilePCiFWrites	count	Number of permanent cylinder index disk segment forced releases or specific I/O requests causing an immediate physical write.	FLOAT
FilePDbFWrites	count	Number of permanent data block disk segment forced releases or specific I/O requests causing an immediate physical write.	FLOAT
FileSCiFWrites	count	Number of regular or restartable spool cylinder index disk segment forced releases or specific I/O requests causing an immediate physical write.	FLOAT

Column Name	Mode	Description	Data Type
FileSdbFWrites	count	Number of regular or restartable spool data block disk segment forced releases or specific I/O requests causing an immediate physical write.	FLOAT
FileTJtFWrites	count	Number of transient journal table or WAL data block or WAL cylinder index disk segment forced releases or specific I/O requests causing an immediate physical write.	FLOAT
FileAPtFWriteKB	count	KB written by FileAPtFWrites.	FLOAT
FilePCiFWriteKB	count	KB written by FilePCiFWrites.	FLOAT
FilePDbFWriteKB	count	KB written by FilePDbFWrites.	FLOAT
FileSCiFWriteKB	count	KB written by FileSCiFWrites.	FLOAT
FileSdbFWriteKB	count	KB written by FileSdbFWrites.	FLOAT
FileTJtFWriteKB	count	KB written by FileTJtFWrites.	FLOAT

I/O Hard Limits Columns

The Input/Output Token Allocations (IOTA) columns provide the percentage of disk throughput that is used.

Column Name	Mode	Description	Data Type
FullPotentiallota	count	Sum of the full potential input or output token allocations from all devices attached to the node. This field is reported only to the AMP vproc rows. The input or output token allocations are not limited by the I/O_COD setting. The data source for this field is located in the all_wd_full_potential_iota field in the /proc/tdmeter/limit_stats file.	FLOAT
CodPotentiallota	count	Sum of the potential input or output token allocations from all devices attached to the node for this workload. This field is reported only to the AMP vproc rows. The input or output token allocations are accumulated only when an I/O is pending on a device, and they are limited by the I/O_COD setting. The data source for this field is located in the wd_potential_iota field in the /proc/tdmeter/limit_stats file.	FLOAT
Usedlota	count	Sum of the used input or output token allocations from all devices attached to the node for this workload. This field is reported only to AMP vproc rows. The data source for this field is located in the wd_used_iota field in the /proc/tdmeter/limit_stats file.	FLOAT
IoThrottleCount	count	Number of times an I/O was throttled.	FLOAT

Column Name	Mode	Description	Data Type
IoThrottleTime	count	Total I/O Throttle Time in milliseconds.	FLOAT
IoThrottleTimeMax	max	Maximum I/O Throttle Time in milliseconds.	FLOAT
IoThrottleCntZerolotas	count	Number of times an I/O was throttled due to no IOTA tokens available in the bucket. This data is included in the IoThrottleCount.	FLOAT
IoThrottleCntInsufflotas	count	Number of times an I/O was throttled due to insufficient IOTA tokens available in the bucket. There were some IOTA tokens in the bucket, but not enough. This data is included in the IoThrottleCount.	FLOAT
IoThrottleCntInsufflotasHL	count	Number of times an I/O was throttled due to insufficient IOTA tokens available in the bucket for Hard Limits. There were some IOTA tokens in the bucket, but not enough. This data is included in the IoThrottleCount.	FLOAT
IoThrottleCntMaxQD	count	Number of times an I/O was throttled due to the queue depth to the storage being at maximum. This data is not part of IoThrottleCount.	FLOAT

I/O Response Time Columns

Column Name	Mode	Description	Data Type
ReadRespMax	max	Maximum I/O Read Response Time in milliseconds.	FLOAT
WriteRespMax	max	Maximum I/O Write Response Time in milliseconds.	FLOAT
ReadRespTot	count	Total I/O Read Response Time in milliseconds.	FLOAT
WriteRespTot	count	Total I/O Write Response Time in milliseconds.	FLOAT

TIM Columns

These columns represent information about the VH (Very Hot) cache, which holds the hottest permanent table cylinders.

Tables that are assigned a temperature of very hot are kept in the FSG cache as long as they:

- Stay very hot.
- Fit into the memory assigned. If the tables cannot fit, the FSG cache considers a sorted list of the hottest segments and assigns them to the very hot cache in temperature sorted order. The temperature of the segment is hot enough to qualify.

The temperature takes into account both physical and logical disk accesses and cache hits (such as, physical and logical I/Os).

Column Name	Mode	Description	Data Type
VHLogicalDBRead	count	Number of logical reads from VH cache. This field is populated by the FSG subsystem.	FLOAT
VHLogicalDBReadKB	count	Volume of logical reads in KB from VH cache. This field is populated by the FSG system.	FLOAT
VHPhysicalDBRead	count	Number of very hot reads that were handled by physical disk I/O due to a VH cache miss (that is, data not found in the VH cache). This field is populated by the FSG system.	FLOAT
VHPhysicalDBReadKB	count	Volume of very hot reads in KB that were handled by physical disk I/O due to a VH cache miss. This field is populated by the FSG system.	FLOAT

Memory Allocation Columns

These columns represent the number and amount of memory allocations, subdivided into (the only applicable) generic node memory type.

Column Name	Mode	Description	Data Type
MemAllocs	count	Number of successful SEG memory allocations.	FLOAT
MemAllocKB	count	Total KB attributed to SEG memory allocations.	FLOAT

Net: Broadcast Net Traffic Columns

These columns identify the number (Reads, Writes) and amount (ReadKB, WriteKB) of input and output messages passing through the Teradata nets through broadcast (1:many) methods (Brd) per net.

Column Name	Mode	Description	Data Type
NetBrdReads	count	Number of broadcast messages input to the vproc.	FLOAT
NetBrdWrites	count	Number of broadcast messages output from the vproc.	FLOAT

Net: Point-to-Point Net Traffic Columns

These columns identify the number (Reads, Writes) and amount (ReadKB, WriteKB) of input and output messages passing through either Teradata net through point-to-point (1:1) methods (PtP).

Column Name	Mode	Description	Data Type
NetPtPReads	count	Number of point-to-point messages input to the vproc on behalf of the WD.	FLOAT
NetPtPWrites	count	Number of point-to-point messages output from the vproc on behalf of the WD.	FLOAT
NetPtPReadKB	count	Total KB of point-to-point messages input to the vproc on behalf of the WD.	FLOAT
NetPtPWriteKB	count	Total KB of point-to-point messages output to the vproc on behalf of the WD.	FLOAT

Row Redistribution Columns

Column Name	Mode	Description	Data Type
RRDHCount	count	Total count of hashed row redistributions (RRDs) processed by the node-level buffered redistribution services (NLB). The RRDHCasc[All Most Some] fields count subsets of those RRDs based on the percentage of AMPs in the RRD cascade tree where the rows do not overflow the cascade buffer but are fully contained within the cascade rather than being sent directly to their targeted destination AMPs via NLB's minicast S2S-Redist messages. This percentage is referred to as the "Cascade Factor" and varies between 0% where the RRD rows overflow the cascade buffer in every AMP, and 100% where the rows are cascaded all the way to the cascade root AMP and do not overflow the cascade buffer even there.	FLOAT
RRDHCascAll	count	Count of hashed RRDs with a Cascade Factor of 100%. In this case all rows are cascaded up to the cascade root AMP without overflowing the cascade buffer anywhere, including in the cascade root itself.	FLOAT
RRDHCascSome	count	Count of hashed RRDs with a Cascade Factor greater than 0% but less than 100%. In this case the rows are cascaded without a buffer overflow in some AMPs, even as few as one, but less than all AMPs. Note: This field is not equivalent to the Svpr table's similarly named RRDHCascSome field, as it also includes the cases counted in Svpr's RRDHCascMost field. There is no Sps field (e.g., "RRDHCascNone") that counts the hashed RRDs with a Cascade Factor of exactly 0%, where the rows overflow the cascade buffer in every AMP. However, this virtual count can be computed from the other fields as follows: RRDHCascNone = RRDHCount - (RRDHCascAll + RRDHCascSome)	FLOAT

Reserved Columns

Column Name	Mode	Description	Data Type
ReservedS0	n/a	Reserved for future use.	CHAR(2)
ReservedS1	n/a	Reserved for future use.	CHAR(7)
Reserved00	n/a	Reserved for future use.	SMALLINT

Spare Columns

The ResUsageSps table spare fields are named Spare00 through Spare19, and SpareInt.

The SpareInt field has a 32-bit internal resolution while all other spare fields have a 64-bit internal resolution. All spare fields default to count but can be converted to min, max, or track mode fields if needed when they are used.

Column Name	Mode	Description	Data Type
Spare00	count	Number of time that processes are blocked waiting for a lock on the CI. The permanent column name CISleeps is reserved for future use.	FLOAT
Spare01	count	Total amount of time in milliseconds that processes are blocked waiting for a lock on the CI. The permanent column name CISleepTime is reserved for future use.	FLOAT
Spare02	max	Maximum amount of time in milliseconds that processes are blocked waiting for a lock on the CI. The permanent column name CISleepTimeMax is reserved for future use.	FLOAT
Spare03 - Spare06	count	Reserved for future use.	FLOAT
Spare07	count	Number of prioritized I/Os that were starved and had their priority updated to alleviate starvation. A prioritized I/O is one that Priority Scheduler issues to the Teradata I/O Scheduler (TDSCHED) Expedited or Normal priority RBTREE queues with a non-zero priority key that determines its position in the associated queue. One of these I/Os that is starved has waited in the TDSCHED queue longer than the Priority Scheduler I/O anti-starvation threshold, the value of which appears in "schmon -iodata -debug" output with the "iostarvedthreshold=" field. The permanent column name IoPrioStarvedCount is reserved for future use.	FLOAT
Spare08	count	Number of times that processes are blocked waiting for a lock on the MI. The permanent column name MISleeps is reserved for future use.	FLOAT

Column Name	Mode	Description	Data Type
Spare09	count	Total amount of time in milliseconds that processes are blocked waiting for a lock on the MI. The permanent column name MISleepTime is reserved for future use.	FLOAT
Spare10	max	Maximum amount of time in milliseconds that processes are blocked waiting for a lock on the MI. The permanent column name MISleepTimeMax is reserved for future use.	FLOAT
Spare11 - Spare14	count	Reserved for future use.	FLOAT
Spare15	count	Number of I/Os submitted to local disks. This data is included in IOSubmitted. The permanent column name IOSubmittedLocal is reserved for future use.	FLOAT
Spare16	count	Amount of I/O submitted, in KB, to local disks. This data is included in IOSubmittedKB. The permanent column name IOSubmittedLocalKB is reserved for future use.	FLOAT
Spare17	count	Time in milliseconds processes were throttled because of PSF WM I/O COD or I/O hard limits. The permanent column name ProcWaitIoThrottle is reserved for future use.	FLOAT
Spare18 - Spare19	count	Reserved for future use.	FLOAT
SpareInt	track	This column reports the number of CPUs available for Teradata tasks at the end of the reporting period. These CPUs are TD-enabled CPUs. Teradata tasks only have access to TD-enabled CPUs while Non-Teradata tasks have access to all online CPUs. This count may change dynamically after database is up but is always less than or equal to the count reported in NCPUs. In future use, this data will be reported in a permanent SPS statistics column named TDEnabledCPUs.	FLOAT

Related Information

For details on the different type of data fields, see [About the Mode Column](#).

ResUsageSvdsK Table

The ResUsageSvdsK table:

- Provides AMP-level storage statistics.
- Includes resource usage logs on cylinder allocation, migration, and I/O statistics.

If you enable table logging on ResUsageSvdsK, a row is written to the database once for every AMP vproc in the system for each log interval. To consolidate and summarize the total number of rows written to the database, you can enable Summary Mode. For details, see [Summary Mode](#).

Note:

This table is created as a MULTiset table.

Teradata recommends that you use ResSvdsKView to access the data rather than accessing the ResUsageSvdsK table directly. For more information, see [Resource Usage Views](#).

Housekeeping Columns

Relational Primary Index Columns

These columns taken together form the nonunique primary index.

Column Name	Mode	Description	Data Type
TheDate	n/a	Date of the log entry.	DATE
TheTime	n/a	Nominal time of the log entry. Note: Under conditions of heavy system load, entries may be logged late (typically, by no more than one or two seconds), but this column will still contain the time value when the entry <i>should have been</i> logged. For more information, see the Secs and NominalSecs columns.	FLOAT
NodeID	n/a	Node ID on which the entry resides. The Node ID is formatted as ZZZ9-9999, where ZZZ9 denotes the four-digit cabinet number and 9999 denotes the four-digit chassis number of the node. For example, a node in chassis 9 of cabinet 3 has a node ID of '3-0009'. Note: SMP nodes have a chassis and cabinet number of 1. For example, the node ID of an SMP node is '1-0001'.	INTEGER

Miscellaneous Housekeeping Columns

These columns provide the general characteristics of the housekeeping columns.

Column Name	Mode	Description	Data Type
GmtTime	n/a	Greenwich Mean Time is not affected by the Daylight Savings Time adjustments that occur twice a year.	FLOAT
CabinetID	n/a	The physical cabinet number of the node.	INTEGER
ModuleID	n/a	The physical module number of the node.	INTEGER
VprId	n/a	Identifies the AMP vproc. In Summary Mode, the value of the AMP vproc ID is -1.	INTEGER
NodeType	n/a	Type of node, representing the per node system family type.	CHAR(8)
Secs	n/a	Actual number of seconds in the log period represented by this row. Normally the same as NominalSecs, but can be different in three cases: <ul style="list-style-type: none"> • The first interval after a log rate change • A sample logged late because of load on the system • System clock adjustments affect reported Secs Useful for normalizing the count statistics contained in this row, for example, to a per-second measurement.	SMALLINT
CentiSecs	n/a	Number of centiseconds in the logging period. This column is useful when performing data calculations with small elapsed times where the difference between centisecond-based data and whole seconds results in a percentage error.	INTEGER
NominalSecs	n/a	Specified or nominal number of seconds in the logging period.	SMALLINT
SummaryFlag	n/a	Summarization status of this row. If the value is 'N,' the row is a non-summary row. If the value is 'S,' the row is a summary row.	CHAR(1)
Active	max	Controls whether or not the rows will be logged to the resource usage tables if Active Row Filter Mode is enabled. If Active is set to a non-zero value, the row contains data columns. If Active is set to a zero value, none of the data columns in the row have been updated during the logging period. For example, if you enable Active Row Filter Mode, the rows that have a zero Active column value will not be logged to the resource usage tables.	FLOAT
TheTimestamp	n/a	Number of seconds since midnight, January 1, 1970. This column is useful for aligning data with the DBQL log.	BIGINT
PM_COD_CPU	n/a	Platform Metering (PM) CPU Capacity On Demand (COD) value in one tenths of a percent. For example, a value of 500 represents a PM CPU COD value of 50.0%. The value is set to 1000 if the PM CPU COD is disabled.	SMALLINT

Column Name	Mode	Description	Data Type
PM_COD_IO	n/a	Platform Metering (PM) I/O Capacity On Demand (COD) value in whole percent. For example, a value of 50 represents a PM I/O COD value of 50.0%. The value is set to 100 if the PM I/O COD is disabled.	SMALLINT
WM_COD_CPU	n/a	Workload Management (WM) CPU Capacity On Demand (COD) value in one tenths of a percent. For example, a value of 500 represents a WM CPU COD value of 50.0%. The value is set to 1000 if the WM CPU COD is disabled.	SMALLINT
WM_COD_IO	n/a	Workload Management (WM) I/O Capacity On Demand (COD) value in whole percent. For example, a value of 50 represents a WM I/O COD value of 50.0%. The value is set to 100 if the WM I/O COD is disabled.	SMALLINT
TIER_FACTOR	n/a	I/O performance limit placed on a core-reduced node. For example, a value of 75 represents an I/O limit of 75.0% placed before other COD values. The field is sourced from tpanodep->tierfactor, which is derived from the /proc/tdmeter/tier_performance file. A value of 100 indicates that there is no I/O performance limit applied to the node.	SMALLINT

Statistics Columns

Teradata Virtual Storage (VS): Allocation Columns

These columns identify the allocation statistics reported by the Allocator process.

Column Name	Mode	Description	Data Type
ExtAllocHot	count	Number of hot allocations made in the current log period. A hot allocation is an allocation whose estimated temperature falls within the pre-defined hot temperature range. The cylinder resides in some disk storage location and holds some data. Temperature is the frequency of access to the data by I/O independent of where the data resides.	FLOAT
ExtAllocNonPacing	count	Number of non-pacing allocations made in the current log period. A non-pacing allocation is an allocation whose data access affects neither system performance nor individual query performance.	FLOAT
ExtAllocStatic	count	Number of static allocations made in the current log period. A static allocation is an allocation whose requested temperature is used and the measured temperature is ignored during migration.	FLOAT

Column Name	Mode	Description	Data Type
ExtAllocSystemPacing	count	Number of system pacing allocations made in the current log period. A system pacing allocation is an allocation whose data access affects system performance.	FLOAT
ExtAllocTotal	count	Total number of allocations made in the current log period. A number of computations can be derived. For example: Cold Allocation = ExtAllocTotal – ExtAllocHot – ExtAllocWarm QueryPacing Allocation = ExtAllocTotal – ExtAllocNonPacing – ExtAllocSystemPacing Dynamic Allocation = ExtAllocTotal - ExtAllocStatic	FLOAT
ExtAllocWarm	count	Number of warm allocations made in the current log period. A warm allocation is an allocation whose estimated temperature falls within the pre-defined warm temperature range. The cylinder resides in some disk storage location and holds some data. Temperature is the frequency of access to the data by I/O independent of where the data resides.	FLOAT

Teradata Virtual Storage (VS): I/O Statistics Columns

These columns identify the I/O statistics reported by FSG for each AMP.

Column Name	Mode	Description	Data Type
ConcurrentMax	max	Maximum number of concurrent I/O requests.	FLOAT
ConcurrentReadMax	max	Maximum number of concurrent read I/O requests.	FLOAT
ConcurrentWriteMax	max	Maximum number of concurrent write I/O requests.	FLOAT
ReadCnt	count	Number of logical device reads.	FLOAT
WriteCnt	count	Number of logical device writes.	FLOAT
ReadKB	count	Number of KB read from the logical device.	FLOAT
WriteKB	count	Number of KB written to the logical device.	FLOAT
ReadRespTot	count	Total of individual read response time in centiseconds.	FLOAT
WriteRespTot	count	Total of individual write response time in centiseconds.	FLOAT
ReadRespMax	max	Maximum of individual read response time in centiseconds.	FLOAT
WriteRespMax	max	Maximum of individual write response time in centiseconds.	FLOAT
ReadRespSq	count	Total of squares of the individual read response time in centiseconds.	FLOAT

Column Name	Mode	Description	Data Type
WriteRespSq	count	Total of squares of the individual write response time in centiseconds.	FLOAT
OutReqTime	count	Time with outstanding requests (busy time), in centiseconds.	FLOAT

Teradata Virtual Storage (VS): Migration Columns

Note:

The ResUsageSvdsk table migration columns are populated only in Teradata VS. For more information, see *Teradata Vantage™ - Teradata® Virtual Storage*, B035-1179.

The migration columns identify the number of cylinders that migrated to a different location on a device as well as the time, in centiseconds, of all migration I/Os used, incurred, or saved during the log period.

Note:

Each allocation is for a cylinder size worth of data, also known internally in the allocator as an extent. Therefore, the column names begin with Ext for extent.

Column Name	Mode	Description	Data Type
ExtMigrateFaster	count	Number of cylinders migrated to faster locations (that is, migrations whose gross benefits are positive) for the associated AMP. The following formula calculates a Slower Migration value, which is the number of cylinders migrated to slower locations: $\text{SlowerMigration} = \text{ExtMigrateTotal} - \text{ExtMigrateFaster}$ Cylinders are migrated to slower locations to make room for hotter cylinders to replace them.	FLOAT
ExtMigrateIOTimeCost	count	Estimates the total cost, in centiseconds, incurred by migration I/Os completing during the log period, where <i>cost</i> is the extra time waited by all non-migration I/Os as a result of the migration I/O.	FLOAT
ExtMigrateIOTimeBenefit	count	Estimates the total I/O time savings achieved by migrations completing in the log period. The I/O time savings include the improvement in response time caused by the new data arrangement up to the time horizon. This value does not include the cost of the migration I/Os and is a gross benefit, not a net benefit.	FLOAT

Column Name	Mode	Description	Data Type
ExtMigrateIOTimeImprove	count	<p>Estimates the percent improvement in average I/O response time due to migrations completing in the log interval.</p> <p>For example, if, right before a particular log interval, the average I/O response time was 10 milliseconds (ms), then the migration logs an ExtMigrateIOTimeImprove value of 10% in this interval. The average I/O response time after the log interval should be $(100\% - 10\%) * 10\text{ms} = 9\text{ms}$. Migration then logs an ExtMigrateIOTimeImprove of 1% in the next interval. The average I/O response time in the new log interval is $(100\% - 1\%) * 9\text{ms} = 8.91\text{ms}$.</p> <p>ExtMigrateIOTimeImprove is only an estimate. Its permanent improvement remains in effect as long as the workload does not change and newer migrations do not significantly alter the data arrangement.</p> <p>When the workload changes or new migrations affect data arrangement, response time changes in a nonquantifiable way.</p> <p>You can use this field to predict the actual system performance for short periods of time and to understand why the migration algorithm is doing what it is doing.</p>	FLOAT
ExtMigrateReadRespTot	count	Migration read I/O response time.	FLOAT
ExtMigrateWriteRespTot	count	Migration write I/O response time.	FLOAT
ExtMigrateTotal	count	Total number of cylinders migrated to a different physical location. For more information, see the ExtMigrateFaster column.	FLOAT

Reserved Columns

Column Name	Mode	Description	Data Type
ReservedS0	n/a	Reserved for future use.	CHAR(1)

Spare Columns

The ResUsageSvdsK table spare fields are named Spare00 through Spare19, and SpareInt.

The SpareInt field has a 32-bit internal resolution while all other spare fields have a 64-bit internal resolution. All spare fields default to count but can be converted to min, max, or track mode fields if needed when they are used.

Column Name	Mode	Description	Data Type
Spare00 - Spare19, SpareInt	count	Reserved for future use.	FLOAT

Related Information

For details on the different type of data fields, see [About the Mode Column](#).

Summary Mode

When Summary Mode is active for the ResUsageSvdsK table, one row is written to the database for each node in the system. This row summarizes all AMP vdisk data in each node per log interval.

You can determine if a row is in Summary Mode by checking the SummaryFlag column for that row.

IF the SummaryFlag column value is...	THEN the data for that row is being logged...
'S'	in Summary Mode.
'N'	normally.

ResUsageSvpr Table

ResUsageSvpr logical table includes resource usage data for available system-wide, virtual processor information.

You can use ResUsageSvpr to find AMP-level skew and compare resource usage across different vprocs.

Note:

This table is created as a MULTiset table. For more information see [Relational Primary Index](#).

Teradata recommends that you use ResSvprView to access the data rather than accessing the ResUsageSvpr table directly. For more information, see [Resource Usage Views](#).

Housekeeping Columns

Relational Primary Index Columns

These columns taken together form the nonunique primary index.

Column Name	Mode	Description	Data Type
TheDate	n/a	Date of the log entry.	DATE
TheTime	n/a	Nominal time of the log entry. Note: Under conditions of heavy system load, entries might be logged late (typically, by no more than one or two seconds), but this column will contain the time value when the entry <i>should have been</i> logged. For more information, see the Secs and NominalSecs columns.	FLOAT
NodeID	n/a	Node ID on which the entry resides. The Node ID is formatted as ZZZ9-9999, where ZZZ9 denotes the four-digit cabinet number and 9999 denotes the four-digit chassis number of the node. For example, a node in chassis 9 of cabinet 3 has a node ID of '3-0009'. Note: SMP nodes have a chassis and cabinet number of 1. For example, the node ID of an SMP node is '1-0001'.	INTEGER

Miscellaneous Housekeeping Columns

These columns provide the general characteristics of the housekeeping columns.

Column Name	Mode	Description	Data Type
GmtTime	n/a	Greenwich Mean Time is not affected by the Daylight Saving Time adjustments that occur twice a year.	FLOAT
CabinetID	n/a	The physical cabinet number of the node.	INTEGER
ModuleID	n/a	The physical module number of the node.	INTEGER
NodeType	n/a	Type of node, representing the per node system family type.	CHAR(8)
VprId	n/a	Identifies the vproc number (non-Summary Mode) or the vproc type (Summary Mode; 0 = NODE, 1 = AMP, 2 = PE, 3=GTW, 4=RSG, 5=TVS).	INTEGER
VprType	n/a	Type of vproc. The values can be NODE, AMP, PE, GTW, RSG, or TVS (see <i>Teradata Vantage™ - Teradata® Virtual Storage</i> , B035-1179).	CHAR(4)
Secs	n/a	Actual number of seconds in the log period represented by this row. Normally the same as NominalSecs, but can be different in three cases: <ul style="list-style-type: none"> • The first interval after a log rate change • A sample logged late because of load on the system • System clock adjustments affect reported Secs Useful for normalizing the count statistics contained in this row, for example, to a per-second measurement.	SMALLINT
CentiSecs	n/a	Number of centiseconds in the logging period. This column is useful when performing data calculations with small elapsed times where the difference between centisecond-based data and whole seconds results in a percentage error.	INTEGER
NominalSecs	n/a	Specified or nominal number of seconds in the logging period.	SMALLINT
NCPUs	n/a	Number of online CPUs on this node. This column is useful for normalizing the CPU utilization column values for the number of CPUs on the node. This is especially important in: <ul style="list-style-type: none"> • Coexistence systems where the number of CPUs can vary across system nodes. • Elastic TCore systems where the number of online CPUs can change without database restart. 	SMALLINT
SummaryFlag	n/a	Summarization status of this row. Possible values are 'N' if the row is a non-summary row and 'S' if it is a summary row.	CHAR (1)
Active	max	Controls whether or not the rows will be logged to the resource usage tables if Active Row Filter Mode is enabled. If Active is set to a non-zero value, the row contains data columns. If Active is set to a zero value, none of the data columns in the row have been updated during the logging period.	FLOAT

Column Name	Mode	Description	Data Type
		For example, if you enable Active Row Filter Mode, the rows that have a zero Active column value will not be logged to the resource usage tables.	
TheTimestamp	n/a	Number of seconds since midnight, January 1, 1970. This column is useful for aligning data with the DBQL log.	BIGINT
PM_COD_CPU	n/a	Platform Metering (PM) CPU Capacity On Demand (COD) value in one tenths of a percent. For example, a value of 500 represents a PM CPU COD value of 50.0%. The value is set to 1000 if the PM CPU COD is disabled.	SMALLINT
WM_COD_CPU	n/a	Workload Management (WM) CPU Capacity On Demand (COD) value in one tenths of a percent. For example, a value of 500 represents a WM CPU COD value of 50.0%. The value is set to 1000 if the WM CPU COD is disabled.	SMALLINT

Statistics Columns

File System: Synchronized Full Table Scans Columns

These columns contain statistics relating to synchronized full table scans.

Note:

The following columns are moved from the ResUsageIvpr table to the ResUsageSvpr table to avoid costly joins.

Column Name	Mode	Description	Data Type
FileSyncGroups	track	Number of groups of scanners involved in full table scans. A group consists of scanners who are able to use the same read I/O to obtain data from disk.	FLOAT
FileSyncScanners	track	Number of tasks involved in full table scans who are willing to synchronize with other scanners.	FLOAT
FileSyncScans	count	Number of attempts to synchronize a full table scan.	FLOAT
FileSyncSubtables	track	Number of subtables scanned by one or more full table scanners who are willing to synchronize scans.	FLOAT

File System: Segments Acquired Columns

These columns identify the total disk memory segments acquired by the File System during the log period.

The File XXxAcqs columns are the only columns counted as cache hits, where XXx represents one of the following entries:

- APt
- PCi
- PDb
- SCi
- SDb
- TJt

Column Name	Mode	Description	Data Type
FileAPtAcqs	count	Total number of append table or permanent journal table data block or cylinder index disk segments.	FLOAT
FilePCiAcqs	count	Total number of permanent cylinder index disk segments acquires that were logically acquired.	FLOAT
FilePDbAcqs	count	Total number of permanent data block disk segments acquires that were logically acquired.	FLOAT
FileSCiAcqs	count	Total number of regular or restartable spool cylinder index disk segments acquires that were logically acquired.	FLOAT
FileSDbAcqs	count	Total number of regular or restartable spool data block disk segments acquires that were logically acquired.	FLOAT
FileTJtAcqs	count	Total number of transient journal or WAL data block or WAL cylinder index disk segments acquires that were logically acquired.	FLOAT
FileAPtAcqKB	count	Total KB logically acquired by FileAPtAcqs.	FLOAT
FilePCiAcqKB	count	Total KB logically acquired by FilePCiAcqs.	FLOAT
FilePDbAcqKB	count	Total KB logically acquired by FilePDbAcqs.	FLOAT
FileSCiAcqKB	count	Total KB logically acquired by FileSCiAcqs.	FLOAT
FileSDbAcqKB	count	Total KB logically acquired by FileSDbAcqs.	FLOAT
FileTJtAcqKB	count	Total KB logically acquired by FileTJtAcqs.	FLOAT
FileAPtAcqReads	count	Number of append table or permanent journal table data block or cylinder index disk segment acquires that caused a physical read.	FLOAT
FilePCiAcqReads	count	Number of permanent cylinder index disk segment acquires that caused a physical read.	FLOAT
FilePDbAcqReads	count	Number of permanent data block disk segment acquires that caused a physical read.	FLOAT
FileSCiAcqReads	count	Number of regular or restartable spool cylinder index disk segment acquires that caused a physical read.	FLOAT

Column Name	Mode	Description	Data Type
FileSDBAcqReads	count	Number of regular or restartable spool data block disk segment acquires that caused a physical read.	FLOAT
FileTJtAcqReads	count	Number of transient journal or WAL data block or WAL cylinder index disk segment acquires that caused a physical read.	FLOAT
FileAPtAcqReadKB	count	Total KB physically read by FileAPtAcqReads.	FLOAT
FilePCiAcqReadKB	count	Total KB physically read by FilePCiAcqReads.	FLOAT
FilePDbAcqReadKB	count	Total KB physically read by FilePDbAcqReads.	FLOAT
FileSCiAcqReadKB	count	Total KB physically read by FileSCiAcqReads.	FLOAT
FileSDBAcqReadKB	count	Total KB physically read by FileSDBAcqReads.	FLOAT
FileTJtAcqReadKB	count	Total KB physically read by FileTJtAcqReads.	FLOAT
FileAPtAcqsOther	count	Total number of append table or permanent journal table data block or cylinder index scratch disk segments that were logically acquired.	FLOAT
FilePCiAcqsOther	count	Total number of permanent cylinder index scratch disk segments that were logically acquired.	FLOAT
FilePDbAcqsOther	count	Total number of permanent data block scratch disk segments that were logically acquired.	FLOAT
FileSCiAcqsOther	count	Total number of regular or restartable spool cylinder index scratch disk segments that were logically acquired.	FLOAT
FileSDBAcqsOther	count	Total number of regular or restartable spool data block scratch disk segments that were logically acquired.	FLOAT
FileTJtAcqsOther	count	Total number of transient journal or WAL data block or WAL cylinder index scratch disk segments that were logically acquired.	FLOAT
FileAPtAcqOtherKB	count	Total number of append table or permanent journal table data block or cylinder index scratch disk segments acquired in KB.	FLOAT
FilePCiAcqOtherKB	count	Total number of permanent cylinder index scratch disk segments acquired in KB.	FLOAT
FilePDbAcqOtherKB	count	Total number of permanent data block scratch disk segments acquired in KB.	FLOAT
FileSCiAcqOtherKB	count	Total number of regular or restartable spool cylinder index scratch disk segments acquired in KB.	FLOAT
FileSDBAcqOtherKB	count	Total number of regular or restartable spool data block scratch disk segments acquired in KB.	FLOAT

Column Name	Mode	Description	Data Type
FileTJtAcqOtherKB	count	Total number of transient journal or WAL data block or WAL cylinder index scratch disk segments acquired in KB.	FLOAT

File System: Data Block Prefetch Columns

These columns identify File Segment Prefetch activities.

Note:

A prefetch is either a cylinder read operation or an individual block read operation. Either of these operations are generically called a prefetch.

Column Name	Mode	Description	Data Type
FileAPtPres	count	Total number of append table or permanent journal table data block or cylinder index disk segments prefetched.	FLOAT
FilePCiPres	count	Total number of permanent cylinder index disk segments prefetched.	FLOAT
FilePDbPres	count	Total number of permanent data block disk segments prefetched.	FLOAT
FileSCiPres	count	Total number of regular or restartable spool index disk segments prefetched.	FLOAT
FileSDBPres	count	Total number of regular or restartable spool data block disk segments prefetched.	FLOAT
FileTJtPres	count	Total number of transient journal table disk segments prefetched.	FLOAT
FileAPtPresKB	count	Total number of KB prefetched by FileAPtPres.	FLOAT
FilePCiPresKB	count	Total number of KB prefetched by FilePCiPres.	FLOAT
FilePDbPresKB	count	Total number of KB prefetched by FilePDbPres.	FLOAT
FileSCiPresKB	count	Total number of KB prefetched by FileSCiPres.	FLOAT
FileSDBPresKB	count	Total number of KB prefetched by FileSDBPres.	FLOAT
FileTJtPresKB	count	Total number of KB prefetched by FileTJtPres.	FLOAT
FileAPtPreReads	count	Total number of append table or permanent journal table data block or cylinder index disk segment prefetches that caused a physical read.	FLOAT
FilePCiPreReads	count	Total number of permanent cylinder index disk segment prefetches that caused a physical read.	FLOAT

Column Name	Mode	Description	Data Type
FilePDbPreReads	count	Total number of permanent data block disk segment prefetches that caused a physical read.	FLOAT
FileSCiPreReads	count	Total number of regular or restartable spool index disk segment prefetches that caused a physical read.	FLOAT
FileSDbPreReads	count	Total number of regular or restartable spool data block disk segment prefetches that caused a physical read.	FLOAT
FileTJtPreReads	count	Total number of transient journal table disk segment prefetches that caused a physical read.	FLOAT
FileAPtPreReadKB	count	Total number of KB physical read by FileAPtPreReads.	FLOAT
FilePCiPreReadKB	count	Total number of KB physical read by FilePCiPreReads.	FLOAT
FilePDbPreReadKB	count	Total number of KB physically read by FilePDbPreReads.	FLOAT
FileSCiPreReadKB	count	Total number of KB physical read by FileSCiPreReads.	FLOAT
FileSDbPreReadKB	count	Total number of KB physical read by FileSDbPreReads.	FLOAT
FileTJtPreReadKB	count	Total number of KB physical read by FileTJtPreReads.	FLOAT

File System: Segments Released Columns

These columns identify the total disk memory segments released by the File System, as well as those segments that are dropped from memory during the log period. When a segment is released, the segment is either:

- Forced (F)
- Remains resident in memory (R)
- Aged out of memory (A), from segments that are currently resident

Both the number of segments (Rels, Writes, Drps) and the size of the segments (RelKB, WriteKB, DrpKB) are counted. When a segment leaves memory, it must be written to disk only if the segment is dirty (Dy), that is, modified. Otherwise, the clean (Cn), that is, unmodified segment is simply dropped.

Most spool blocks for a small table remain resident when they are created and age there. Each of these will be counted as a dirty resident release (DyRRel columns). If a block survives in the cache, it would be reacquired (whenever the system creates spool data, a subsequent step will read it) and released again. The release will still be counted as a dirty resident release, since the block survived in a modified state. On the other hand, if there is contention for room in the FSG cache, the segment might be removed from memory. Because it is a modified segment, it must be written out first. This is counted as a dirty age write (DyAWrite columns). When it is reacquired it will no longer be modified, so the subsequent release will be counted as a clean resident release (CnRRel columns).

If the segments are modified, only DyAWrites, DyAWriteKB, CnADrps, and CnADrpKB columns are incremented.

If the segments are unmodified, only CnADrps and CnADrpKB columns are incremented.

Note:

To determine the clean segments that aged out of memory for the CnADrps column, subtract the DyAWrites value from the CnADrps value. To determine the clean segments that aged out of memory for the CnADrpKB column, subtract the DyAWriteKB value from the CnADrpKB value.

Full table modification operations make one pass on the table and modify each block only once. Since these operations do not access a block multiple times, there is no point keeping them in the cache. If a block that was examined did not contain any rows that qualify for the modification, when the block is released, it will be dropped from memory immediately (FDrp columns). However, if the block was modified, when it is released the system issues the write as part of the release so it is counted as a forced write (FWrite columns). Since the system also drops the block from memory as soon as the write is complete, this release is also counted as a forced drop (FDrp columns).

Column Name	Mode	Description	Data Type
FileAPtDyRRels	count	Number of dirty append table or permanent journal table data block or cylinder index disk segment resident releases.	FLOAT
FilePCiDyRRels	count	Number of dirty permanent cylinder index disk segment resident releases.	FLOAT
FilePDbDyRRels	count	Number of dirty permanent data block disk segment resident releases.	FLOAT
FileSCiDyRRels	count	Number of dirty regular or restartable spool cylinder index disk segment resident releases.	FLOAT
FileSDbDyRRels	count	Number of dirty regular or restartable spool data block disk segment resident releases.	FLOAT
FileTJtDyRRels	count	Number of dirty transient journal table or WAL data block or WAL cylinder index disk segment resident releases.	FLOAT
FileAPtDyRRelsKB	count	KB released by FileAPtDyRRels.	FLOAT
FilePCiDyRRelsKB	count	KB released by FilePCiDyRRels.	FLOAT
FilePDbDyRRelsKB	count	KB released by FilePDbDyRRels.	FLOAT
FileSCiDyRRelsKB	count	KB released by FileSCiDyRRels.	FLOAT
FileSDbDyRRelsKB	count	KB released by FileSDbDyRRels.	FLOAT
FileTJtDyRRelsKB	count	KB released by FileTJtDyRRels.	FLOAT
FileAPtFWrites	count	Number of append table or permanent journal table data block or cylinder index disk segment forced releases or specific I/O requests causing an immediate physical write.	FLOAT

Column Name	Mode	Description	Data Type
FilePCiFWrites	count	Number of permanent cylinder index disk segment forced releases or specific I/O requests causing an immediate physical write.	FLOAT
FilePDbFWrites	count	Number of permanent data block disk segment forced releases or specific I/O requests causing an immediate physical write.	FLOAT
FileSCiFWrites	count	Number of regular or restartable spool cylinder index disk segment forced releases or specific I/O requests causing an immediate physical write.	FLOAT
FileSdbFWrites	count	Number of regular or restartable spool data block disk segment forced releases or specific I/O requests causing an immediate physical write.	FLOAT
FileTJtFWrites	count	Number of transient journal table or WAL data block or WAL cylinder index disk segment forced releases or specific I/O requests causing an immediate physical write.	FLOAT
FileAPtFWriteKB	count	KB written by FileAPtFWrites.	FLOAT
FilePCiFWriteKB	count	KB written by FilePCiFWrites.	FLOAT
FilePDbFWriteKB	count	KB written by FilePDbFWrites.	FLOAT
FileSCiFWriteKB	count	KB written by FileSCiFWrites.	FLOAT
FileSdbFWriteKB	count	KB written by FileSdbFWrites.	FLOAT
FileTJtFWriteKB	count	KB written by FileTJtFWrites.	FLOAT
FileAPtDyAWrites	count	Number of dirty append table or permanent journal table data block or cylinder index disk segments aged out of memory causing a delayed physical write.	FLOAT
FilePCiDyAWrites	count	Number of dirty permanent cylinder index disk segments aged out of memory causing a delayed physical write.	FLOAT
FilePDbDyAWrites	count	Number of dirty permanent data block disk segments aged out of memory causing a delayed physical write.	FLOAT
FileSCiDyAWrites	count	Number of dirty regular or restartable spool cylinder index disk segments aged out of memory causing a delayed physical write.	FLOAT
FileSdbDyAWrites	count	Number of dirty regular or restartable spool data block disk segments aged out of memory causing a delayed physical write.	FLOAT
FileTJtDyAWrites	count	Number of dirty transient journal table or WAL data block or WAL cylinder index disk segments aged out of memory causing a delayed physical write.	FLOAT
FileAPtDyAWriteKB	count	KB written by FileAPtDyAWrites.	FLOAT

Column Name	Mode	Description	Data Type
FilePCiDyAWriteKB	count	KB written by FilePCiDyAWrites.	FLOAT
FilePDbDyAWriteKB	count	KB written by FilePDbDyAWrites.	FLOAT
FileSCiDyAWriteKB	count	KB written by FileSCiDyAWrites.	FLOAT
FileSDbDyAWriteKB	count	KB written by FileSDbDyAWrites.	FLOAT
FileTJtDyAWriteKB	count	KB written by FileTJtDyAWrites.	FLOAT
FileAPtCnRRels	count	Number of clean append table or permanent journal table data block or cylinder index disk segment resident releases.	FLOAT
FilePCiCnRRels	count	Number of clean permanent cylinder index disk segment resident releases.	FLOAT
FilePDbCnRRels	count	Number of clean permanent data block disk segment resident releases.	FLOAT
FileSCiCnRRels	count	Number of clean regular or restartable spool cylinder index disk segment resident releases.	FLOAT
FileSDbCnRRels	count	Number of clean regular or restartable spool data block disk segment resident releases.	FLOAT
FileTJtCnRRels	count	Number of clean transient journal table or WAL data block or WAL cylinder index disk segment resident releases.	FLOAT
FileAPtCnRRelKB	count	KB released by FileAPtCnRRels.	FLOAT
FilePCiCnRRelKB	count	KB released by FilePCiCnRRels.	FLOAT
FilePDbCnRRelKB	count	KB released by FilePDbCnRRels.	FLOAT
FileSCiCnRRelKB	count	KB released by FileSCiCnRRels.	FLOAT
FileSDbCnRRelKB	count	KB released by FileSDbCnRRels.	FLOAT
FileTJtCnRRelKB	count	KB released by FileTJtCnRRels.	FLOAT
FileAPtFDrps	count	Number of append table or permanent journal table data block or cylinder index disk segment forced releases causing an immediate memory drop.	FLOAT
FilePCiFDrps	count	Number of permanent cylinder index disk segment forced releases causing an immediate memory drop.	FLOAT
FilePDbFDrps	count	Number of permanent data block disk segment forced releases causing an immediate memory drop.	FLOAT
FileSCiFDrps	count	Number of regular or restartable spool cylinder index disk segment forced releases causing an immediate memory drop.	FLOAT

Column Name	Mode	Description	Data Type
FileSdbFDrps	count	Number of regular or restartable spool data block disk segment forced releases causing an immediate memory drop.	FLOAT
FileTjtFDrps	count	Number of transient journal table or WAL data block or WAL cylinder index disk segment forced releases causing an immediate memory drop.	FLOAT
FileAPtFDrpKB	count	KB dropped by FileAPtFDrps.	FLOAT
FilePCiFDrpKB	count	KB dropped by FilePCiFDrps.	FLOAT
FilePDbFDrpKB	count	KB dropped by FilePDbFDrps.	FLOAT
FileSCiFDrpKB	count	KB dropped by FileSCiFDrps.	FLOAT
FileSdbFDrpKB	count	KB dropped by FileSdbFDrps.	FLOAT
FileTjtFDrpKB	count	KB dropped by FileTjtFDrps.	FLOAT
FileAPtCnADrps	count	Number of clean append table or permanent journal table data block or cylinder index disk segments aged out of memory.	FLOAT
FilePCiCnADrps	count	Number of clean permanent cylinder index disk segments aged out of memory.	FLOAT
FilePDbCnADrps	count	Number of clean permanent data block disk segments aged out of memory.	FLOAT
FileSCiCnADrps	count	Number of clean regular or restartable spool cylinder index disk segments aged out of memory.	FLOAT
FileSdbCnADrps	count	Number of clean regular or restartable spool data block disk segments aged out of memory.	FLOAT
FileTjtCnADrps	count	Number of clean transient journal table or WAL data block or WAL cylinder index disk segments aged out of memory.	FLOAT
FileAPtCnADrpKB	count	KB dropped by FileAPtCnADrps.	FLOAT
FilePCiCnADrpKB	count	KB dropped by FilePCiCnADrps.	FLOAT
FilePDbCnADrpKB	count	KB dropped by FilePDbCnADrps.	FLOAT
FileSCiCnADrpKB	count	KB dropped by FileSCiCnADrps.	FLOAT
FileSdbCnADrpKB	count	KB dropped by FileSdbCnADrps.	FLOAT
FileTjtCnADrpKB	count	KB dropped by FileTjtCnADrps.	FLOAT
FileAPtRelsOther	count	Total number of scratch append table or permanent journal table data block or cylinder index disk segments released as part of them being deleted.	FLOAT

Column Name	Mode	Description	Data Type
FilePCiRelsOther	count	Total number of scratch permanent cylinder index disk segments released as part of them being deleted.	FLOAT
FilePDbRelsOther	count	Total number of scratch permanent data block disk segments released as part of them being deleted.	FLOAT
FileSCiRelsOther	count	Total number of scratch regular or restartable spool cylinder index disk segments released as part of them being deleted.	FLOAT
FileSDbRelsOther	count	Total number of scratch regular or restartable spool data block disk segments released as part of them being deleted.	FLOAT
FileTJtRelsOther	count	Total number of scratch transient journal table or WAL data block or WAL cylinder index disk segments released as part of them being deleted.	FLOAT
FileAPtRelOtherKB	count	Total number of scratch append table or permanent journal table data block or cylinder index disk segments released in KB.	FLOAT
FilePCiRelOtherKB	count	Total number of scratch permanent cylinder index disk segments released in KB.	FLOAT
FilePDbRelOtherKB	count	Total number of scratch permanent data block disk segments released in KB.	FLOAT
FileSCiRelOtherKB	count	Total number of scratch regular or restartable spool cylinder index disk segments released in KB.	FLOAT
FileSDbRelOtherKB	count	Total number of scratch regular or restartable spool data block disk segments released in KB.	FLOAT
FileTJtRelOtherKB	count	Total number of scratch transient journal table or WAL data block or WAL cylinder index disk segments released in KB.	FLOAT

File System: Data Segment Lock Requests Columns

These columns identify the number of lock requests, blocks, and deadlocks on a disk segment, including those implied for segment acquires.

Note:

The locks referred to here are short term file system locks and are distinct from transaction locks, which are counted in the fields documented in [General Concurrency Control Database Locks Columns](#).

Column Name	Mode	Description	Data Type
FileLockBlocks	count	Number of lock requests that were blocked. This column is calculated as follows:	FLOAT

Column Name	Mode	Description	Data Type
		Total locks - Locks blocked = Locks with immediate grants	
FileLockDeadlocks	count	Number of deadlocks detected on lock requests.	FLOAT
FileLockEnters	count	Number of lock requests on disk segments.	FLOAT

File System: Cylinder Management Overhead Events Columns

These columns identify the number of times the File System software performed a cylinder management event. The table ResUsagelvpr further breaks down the I/Os associated with these events. For more information, see [ResUsagelvpr Table](#).

Column Name	Mode	Description	Data Type
FileCylAllocs	count	Number of new cylinders allocated.	FLOAT
FileCylDefrags	count	Number of cylinder defragments performed.	FLOAT
FileCylFrees	count	Number of logical or physical cylinders freed.	FLOAT
FileCylMigrs	count	Number of cylinder migrations.	FLOAT
FileMCylPacks	count	Number of MiniCylPack operations performed.	FLOAT

File System: Miscellaneous Write Ahead Logging Columns

These columns identify the number of times the File System software performed a cylinder management event associated with the WAL log.

Column Name	Mode	Description	Data Type
FileWALBufBusy	count	Number of times a WAL append operation resulted in a buffer switch that waited for the next buffer to exit a busy state.	FLOAT
FileWALSize	track	Current size of the WAL log in cylinders.	FLOAT
FileWCylAllocs	count	Number of new WAL cylinders allocated.	FLOAT
FileWCylFrees	count	Number of times the File System logically frees a cylinder.	FLOAT

File System: MI Columns

Column Name	Mode	Description	Data Type
MIWriteLocks	count	Number of write locks acquired on a MI.	FLOAT
MIWriteLockTime	count	Total MI write lock hold time in milliseconds.	FLOAT

Column Name	Mode	Description	Data Type
MIWriteLockTimeMax	max	Maximum MI write lock hold time in milliseconds.	FLOAT
MISleeps	count	Number of times that processes are blocked waiting for a lock on the MI.	FLOAT
MISleepTime	count	Total amount of time in milliseconds that processes are blocked waiting for a lock on the MI.	FLOAT
MISleepTimeMax	max	Maximum amount of time in milliseconds that processes are blocked waiting for a lock on the MI.	FLOAT

File System: Data Block Merge Columns

These columns merge small data blocks into one large block so that subsequent merge attempts will have less I/Os.

Column Name	Mode	Description	Data Type
DBMergeDone	count	Number of successful data block merges done. The number of times the data block being modified successfully merged with some number of adjacent data blocks as part of the modification. Subtracting DBMergeDone from DBMergeTried will result in the number of data block merge operations that were tried and failed.	FLOAT
DBMergeElim	count	Number of data blocks eliminated due to data block merges. If n data blocks are merged into one large block, where n is the number of data blocks, this number is incremented by $n-1$.	FLOAT
DBMergeExtraIO	count	Number of additional physical I/Os performed in the data block merge process over and above those done if no data block merges were attempted. This includes any extra physical I/Os that were performed regardless of whether a particular merge attempt succeeded or not.	FLOAT
DBMergeTried	count	Number of times the data block being modified tried to be merged with adjacent data blocks as part of the modification.	FLOAT

File System: AutoCylPack Columns

These columns track the number of cylinders used during AutoCylPack, a background task that runs periodically to maintain the set levels of free space percent (FSP) on table cylinders.

For more information about AutoCylPack, see *Teradata Vantage™ - Database Administration*, B035-1093.

Column Name	Mode	Description	Data Type
FileACPCylsMigr	count	Number of successful migrations performed by AutoCylPack.	FLOAT
FileACPCylsPostponed	count	Number of cylinders AutoCylPack selected for performing a migration, but could not be performed at the current time. This can happen due to conflicts with foreground tasks modifying the cylinder at around the same time as AutoCylPack. AutoCylPack, therefore, postpones the work until the next time it scans the MI from the beginning. When AutoCylPack comes across the cylinder again, and it still qualifies, the cylinder is selected again for processing.	FLOAT
FileACPCylsSkipped	count	Number of cylinders AutoCylPack skipped when scanning the MI since nothing needed to be done.	FLOAT
FileACPCylsUnFSEOnly	count	Number of cylinders AutoCylPack selected for performing a migration, but could not be performed at the current time. This can happen because of a locking conflict or a cylinder being modified or marked down. Instead, AutoCylPack cleans-up the unfree sector entries (UnFSEs) on all cylinders, except for those that are down.	FLOAT

File System: BLC Columns

These columns collect various BLC statistics for use in performance analysis and debugging.

BLC enables data compression at the data block (DB) level of the Teradata file system. Compression reduces the amount of storage required for a given amount of data. The BlockLevelCompression field of DBS Control enables and disables BLC.

For more information on BLC and compression-related DBS Control settings, see *Teradata Vantage™ - Database Utilities*, B035-1102.

Note:

You must enable BLC to collect statistics from the columns below or zero will be returned for each column.

Column Name	Mode	Description	Data Type
FileCompDBs	count	Total number of data blocks compressed.	FLOAT
FileCompDBs_HW	count	Total number of data blocks compressed by the hardware compression card. This amount is included in the FileCompDBs column.	FLOAT

Column Name	Mode	Description	Data Type
FileCompCPU	count	Total compression time, including any overhead. The column measures, in microseconds, the time it takes from the beginning of the compression operation to the end.	FLOAT
FilePostCompMB	count	Total number of MBs of compressed data that result from uncompressed blocks being compressed. This column is used together with the FilePreCompMB column to calculate the compression ratio.	FLOAT
FilePreCompMB	count	Total number of MBs for the data block that will be compressed before any compression starts. This column is used together with the FilePostCompMB column to calculate the compression ratio.	FLOAT
FileCompCylMigrs	count	Number of cylinder migrates done as part of a Ferret COMPRESS command or by the AutoTempComp background task. The AutoTempComp background task is responsible for finding the cylinders that should be compressed or decompressed based on their temperature. For more information on this background task, see <i>Teradata Vantage™ - Database Design</i> , B035-1094. For more information on the Ferret utility, COMPRESS command, see <i>Teradata Vantage™ - Database Utilities</i> , B035-1102.	FLOAT
FileCompFerretDBs	count	Number of blocks compressed as a result of the compression operation called from the Ferret utility (see <i>Teradata Vantage™ - Database Utilities</i> , B035-1102).	FLOAT
FileCompTempDBs	count	Number of data blocks compressed by the AutoTempComp background task because they were colder than the DBS Control TempBLCThresh field setting. See <i>Teradata Vantage™ - Database Design</i> , B035-1094 for more information on the DBS Control TempBLCThresh field and AutoTempComp background task. Note: You must enable the temperature-based block-level compression (TBBLC) feature when using this column or it will return a zero. To determine if TBBLC is enabled, see EnableTempBLC in <i>Teradata Vantage™ - Database Utilities</i> , B035-1102.	FLOAT
FilePreUncompMB	count	Total number of MBs for the compressed data block that will be uncompressed before uncompression starts. This column is used together with the FilePostUncompMB column to calculate the uncompression ratio.	FLOAT
FilePostUncompMB	count	Total number of MBs of uncompressed data that result from compressed blocks being uncompressed.	FLOAT

Column Name	Mode	Description	Data Type
		This column is used together with the FilePreUncompMB column to calculate the uncompression ratio.	
FileTempCPU	count	<p>CPU time, in microseconds, spent by the AutoTempComp background task either compressing or uncompressing data, including any overhead.</p> <p>For more information on the AutoTempComp background task, see <i>Teradata Vantage™ - Database Utilities</i>, B035-1102.</p> <p>Note:</p> <p>You must enable the temperature-based block-level compression (TBBLC) feature when using this column or it will return zero. To determine if TBBLC is enabled, see EnableTempBLC in <i>Teradata Vantage™ - Database Utilities</i>, B035-1102.</p> <p>Also, any time counted against this column is not counted in either the FileCompCPU or FileUnCompCPU column.</p>	FLOAT
FileUnCompCPU	count	Total uncompression time, including any overhead. The column measures, in microseconds, the time it takes from the beginning of the uncompression operation to the end.	FLOAT
FileUnCompCylMigrs	count	<p>Number of cylinder migrates done as part of a Ferret UNCOMPRESS command or by the AutoTempComp background task.</p> <p>For more information on the Ferret utility, UNCOMPRESS command, or AutoTempComp background task, see <i>Teradata Vantage™ - Database Utilities</i>, B035-1102.</p>	FLOAT
FileUnCompDBs	count	Total number of data blocks uncompressed.	FLOAT
FileUnCompDBs_HW	count	Total number of data blocks uncompressed by the hardware compression card. This amount is included in the FileUnCompDBs column.	FLOAT
FileUnCompFerretDBs	count	Number of blocks uncompressed as a result of the uncompression operation called from the Ferret utility (see <i>Teradata Vantage™ - Database Utilities</i> , B035-1102).	FLOAT
FileUnCompTempDBs	count	Number of data blocks uncompressed by the AutoTempComp background task because they were warmer than the DBS Control TempBLCThresh field setting. See <i>Teradata Vantage™ - Database Design</i> , B035-1094 for more information on the DBS Control TempBLCThresh field and AutoTempComp background task.	FLOAT

Column Name	Mode	Description	Data Type
		Note: You must enable the TBBLC feature when using this column or it will return zero. To determine if TBBLC is enabled, see EnableTempBLC in <i>Teradata Vantage™ - Database Utilities</i> , B035-1102.	

File System: FSG I/O Column

This column identifies the I/O statistics reported from the FSG.

Column Name	Mode	Description	Data Type
IoRespMax	max	Maximum I/O response time in milliseconds on an AMP.	FLOAT

File System: FSG Cache Wait Columns

Column Name	Mode	Description	Data Type
FSGCacheWaits	count	Number of times the File System waits for memory to become available in the file segment cache when trying to read data from disk.	FLOAT
FSGCacheWaitTime	count	Total amount of time the File System waits for memory to become available in the file segment cache when trying to read data from disk.	FLOAT
FSGCacheWaitTimeMax	max	Maximum amount of time the File System waits for memory to become available in the File System cache when trying to read data from disk.	FLOAT

General Concurrency Control Database Locks Columns

These columns identify database locking activities.

Column Name	Mode	Description	Data Type
DBLockBlocks	count	Number of times a database lock was blocked.	FLOAT
DBLockEnters	count	Number of times a database lock was entered for holding.	FLOAT
DBLockDeadlocks	count	Number of times a database lock was deadlocked.	FLOAT

Memory: Memory Resident Columns

These columns represent the amount of memory resident specific to virtual processor activities, subdivided into memory types. The columns do not include any memory allocations specific to the node the vproc is running under.

Disk memory segments can be in one of four states:

- Clean (unmodified) and Unaccessed by any process (CU)
- Dirty (modified) and Unaccessed (DU)
- Clean and Accessed (CA)
- Dirty and Accessed (DA)

Permanent segments for an entire table can be user-locked-in to memory. These are called frozen segments (Frz), and no state subdivision is necessary because they cannot be aged or forced out of memory.

Column Name	Mode	Description	Data Type
MemAPtKBResCA	track	Current KB resident in memory for append table or permanent journal table data block or cylinder index disk segments that are currently clean and accessed. Note: This column is not currently valid. It should not be used.	FLOAT
MemAPtKBResCU	track	Current KB resident in memory for append table or permanent journal table data block or cylinder index disk segments that are currently clean and not accessed. Note: This column is not currently valid. It should not be used.	FLOAT
MemAPtKBResDA	track	Current KB resident in memory for append table or permanent journal table data block or cylinder index disk segments that are currently dirty and accessed. Note: This column is not currently valid. It should not be used.	FLOAT
MemAPtKBResDU	track	Current KB resident in memory for append table or permanent journal table data block or cylinder index disk segments that are currently dirty and unaccessed. Note: This column is not currently valid. It should not be used.	FLOAT
MemPCiKBResCA	track	Current KB resident in memory for permanent cylinder index disk segments that are currently clean and accessed.	FLOAT

Column Name	Mode	Description	Data Type
		Note: This column is not currently valid. It should not be used.	
MemPCiKBResCU	track	Current KB resident in memory for permanent cylinder index disk segments that are currently clean and not accessed. Note: This column is not currently valid. It should not be used.	FLOAT
MemPCiKBResDA	track	Current KB resident in memory for permanent cylinder index disk segments that are currently dirty and accessed. Note: This column is not currently valid. It should not be used.	FLOAT
MemPCiKBResDU	track	Current KB resident in memory for permanent cylinder index disk segments that are currently dirty and unaccessed. Note: This column is not currently valid. It should not be used.	FLOAT
MemPDbKBResCA	track	Current KB resident in memory for permanent data block segments that are currently clean and accessed. Note: This column is not currently valid. It should not be used.	FLOAT
MemPDbKBResCU	track	Current KB resident in memory for permanent data block disk segments that are currently clean and not accessed. Note: This column is not currently valid. It should not be used.	FLOAT
MemPDbKBResDA	track	Current KB resident in memory for permanent data block disk segments that are currently dirty and accessed. Note: This column is not currently valid. It should not be used.	FLOAT
MemPDbKBResDU	track	Current KB resident in memory for permanent data block disk segments that are currently dirty and unaccessed. Note: This column is not currently valid. It should not be used.	FLOAT
MemSCiKBResCA	track	Current KB resident in memory for regular or restartable spool cylinder index disk segments that are currently clean and accessed.	FLOAT

Column Name	Mode	Description	Data Type
		Note: This column is not currently valid. It should not be used.	
MemSCiKBResCU	track	Current KB resident in memory for regular or restartable spool cylinder index disk segments that are currently clean and not accessed. Note: This column is not currently valid. It should not be used.	FLOAT
MemSCiKBResDA	track	Current KB resident in memory for regular or restartable spool cylinder index disk segments that are currently dirty and accessed. Note: This column is not currently valid. It should not be used.	FLOAT
MemSCiKBResDU	track	Current KB resident in memory for regular or restartable spool cylinder index disk segments that are currently dirty and unaccessed. Note: This column is not currently valid. It should not be used.	FLOAT
MemSDbKBResCA	track	Current KB resident in memory for regular or restartable spool data block disk segments that are currently clean and accessed. Note: This column is not currently valid. It should not be used.	FLOAT
MemSDbKBResCU	track	Current KB resident in memory for regular or restartable spool data block disk segments that are currently clean and not accessed. Note: This column is not currently valid. It should not be used.	FLOAT
MemSDbKBResDA	track	Current KB resident in memory for regular or restartable spool data block disk segments that are currently dirty and accessed. Note: This column is not currently valid. It should not be used.	FLOAT
MemSDbKBResDU	track	Current KB resident in memory for regular or restartable spool data block disk segments that are currently dirty and unaccessed. Note: This column is not currently valid. It should not be used.	FLOAT

Column Name	Mode	Description	Data Type
MemTJtKBResCA	track	Current KB resident in memory for transient journal table or WAL data block or WAL cylinder index disk segments that are currently clean and accessed. Note: This column is not currently valid. It should not be used.	FLOAT
MemTJtKBResCU	track	Current KB resident in memory for transient journal table or WAL data block or WAL cylinder index disk segments that are currently clean and not accessed. Note: This column is not currently valid. It should not be used.	FLOAT
MemTJtKBResDA	track	Current KB resident in memory for transient journal table or WAL data block or WAL cylinder index disk segments that are currently dirty and accessed. Note: This column is not currently valid. It should not be used.	FLOAT
MemTJtKBResDU	track	Current KB resident in memory for transient journal table or WAL data block or WAL cylinder index disk segments that are currently dirty and unaccessed. Note: This column is not currently valid. It should not be used.	FLOAT

Memory: Memory Allocation Column

Column Name	Mode	Description	Data Type
MemCtxtAllocs	count	Number of successful memory allocations and size-increasing memory alters on task context pages. Note: Only scratch pages are allocated. All other task context pages <i>appear</i> resident at some point soon after component restart.	FLOAT

Memory: Large Memory Sort Columns

Column Name	Mode	Description	Data Type
FileSorts	count	Number of sort requests, including the large memory sort requests.	FLOAT

Column Name	Mode	Description	Data Type
FileLargeMemSorts	count	Number of large memory sort requests executed. Large memory sort uses large buffers (1MB buffers) and up to 1GB of memory.	FLOAT
FileLargeMemSortsDenied	count	Number of large memory sort requests denied.	FLOAT

Memory: TIM Columns

These columns represent information about the VH (Very Hot) cache, which holds the hottest permanent table cylinders.

Tables that are assigned a temperature of very hot are kept in the FSG cache as long as they:

- Stay very hot.
- Fit into the memory assigned. If the tables cannot fit, the FSG cache considers a sorted list of the hottest segments and assigns them to the very hot cache in temperature sorted order. The temperature of the segment is hot enough to qualify.

The temperature takes into account both physical and logical disk accesses and cache hits (such as, physical and logical I/Os).

Column Name	Mode	Description	Data Type
VHAgedOut	count	Number of data segments that were aged out of VH cache.	FLOAT
VHAgedOutKB	count	Volume of data segments in KB that were aged out of VH cache.	FLOAT
VHLogicalDBRead	count	Number of logical reads from VH cache.	FLOAT
VHLogicalDBReadKB	count	Volume of logical reads in KB from VH cache.	FLOAT
VHPhysicalDBRead	count	Number of very hot reads that were handled by physical disk I/O due to a VH cache miss (that is, data not found in the VH cache).	FLOAT
VHPhysicalDBReadKB	count	Volume of very hot reads in KB that were handled by physical disk I/O due to a VH cache miss.	FLOAT
VHCACHEInuseKB	track	Size of VH cache in KB currently in use.	FLOAT

Memory: FSG Cache Column

These columns represent information about the VH (Very Hot) cache, which holds the hottest permanent table cylinders.

Tables that are assigned a temperature of very hot are kept in the FSG cache as long as they:

- Stay very hot.
- Fit into the memory assigned. If the tables cannot fit, the FSG cache considers a sorted list of the hottest segments and assigns them to the very hot cache in temperature sorted order. The temperature of the segment is hot enough to qualify.

The temperature takes into account both physical and logical disk accesses and cache hits (such as, physical and logical I/Os).

Column Name	Mode	Description	Data Type
FsgCachelnuseKB	track	Amount of FSG cache in use. This includes the memory in use by TIM (VHCachelnuseKB). For non-AMP vproc, this field is 0.	FLOAT

Memory: Task Context Segment Usage Columns

These columns identify the usage of task context segments and how they leave memory.

Column Name	Mode	Description	Data Type
MemCtxtAccesses	count	Number of segments accessed.	FLOAT
MemCtxtAccessKB	count	KB of segments accessed.	FLOAT
MemCtxtDeaccesses	count	Number of segments deaccessed. Deaccessed segments remain in memory until paged out through aging.	FLOAT
MemCtxtDeaccessKB	count	KB of segments deaccessed.	FLOAT
MemCtxtDestroyKB	count	KB of segments destroyed.	FLOAT
MemCtxtDestroys	count	Number of segments destroyed. Destroyed segments are immediately dropped from memory.	FLOAT

Native Object Store Columns

Column Name	Mode	Description	Data Type
NosPhysReadIOs	count	Number of physical read IOs issued against Native Object Store tables.	FLOAT
NosPhysReadIOKB	count	Total KB physically read from Native Object Store tables.	FLOAT
NosFiles	count	Number of Native Object Store files read in satisfying queries against Native Object Store tables.	FLOAT
NosFilesSkipped	count	Number of Native Object Store files skipped due to errors.	FLOAT

Column Name	Mode	Description	Data Type
NosRecordsReturned	count	Number of records returned in satisfying queries against Native Object Store tables.	FLOAT
NosRecordsReturnedKB	count	Total KB of records returned from Native Object Store files (after any decompression and character conversion performed).	FLOAT
NosRecordsSkipped	count	Number of Native Object Store records skipped due to errors	FLOAT
NosTotalIOWaitTime	count	Total time in seconds waiting for IOs when reading Native Object Store files.	FLOAT
NosMaxIOWaitTime	max	Maximum single IO wait time in seconds when reading Native Object Store files.	FLOAT
NosCPUTime	count	Total CPU time in seconds spend decompressing, translating, or otherwise processing Native Object Store data as part of reading it.	FLOAT
NosFilesWritten	count	Number of file writes attempted by Native Object Store.	FLOAT
NosPhysWriteIOs	count	Total physical write IOs for Native Object Store files.	FLOAT
NosPhysWriteOKB	count	Total KB of physical write IOs for Native Object Store files.	FLOAT

Net: Point-to-Point Net Traffic Columns

These columns identify the number (Reads, Writes) and amount (ReadKB, WriteKB) of input and output messages passing through either database net through point-to-point (1:1) methods (PtP).

Column Name	Mode	Description	Data Type
NetPtPReadKB	count	Total KB of point-to-point messages input to the vproc.	FLOAT
NetPtPReads	count	Number of point-to-point messages input to the vproc.	FLOAT
NetPtPWriteKB	count	Total KB of point-to-point messages output to the vproc.	FLOAT
NetPtPWrites	count	Number of point-to-point messages output from the vproc.	FLOAT

Net: Broadcast Net Traffic Columns

These columns identify the number (Reads, Writes) and amount (ReadKB, WriteKB) of input and output messages passing through the database nets through broadcast (1:many) methods (Brd) per net.

Column Name	Mode	Description	Data Type
NetBrdReadKB	count	Total KB of broadcast messages input to the vproc.	FLOAT

Column Name	Mode	Description	Data Type
NetBrdReads	count	Number of broadcast messages input to the vproc.	FLOAT
NetBrdWriteKB	count	Total KB of broadcast messages output from the vproc.	FLOAT
NetBrdWrites	count	Number of broadcast messages output from the vproc.	FLOAT

Net: Work Mailbox Queue Columns

These columns identify the virtual processor work mailbox queue length where requested work awaits the allocation of a process to perform the work.

Column Name	Mode	Description	Data Type
MsgWorkQLen	track	Total number of work requests waiting at the current time.	FLOAT
MsgWorkQLenMax	max	Maximum number of work requests waiting during each log interval. The Max count, unlike the MsgWorkQLen field, tracks the maximum count over the log period. Therefore, the MsgWorkQLen field can be zero even though the Max count can be non-zero over the log period.	FLOAT

Process Scheduling: ChnSignal Status Tracking Column

Column Name	Mode	Description	Data Type
MsgChnLastDone	count	Number of last done events that occurred for this vproc. Note: The last AMP to finish an operation may send a last done broadcast message indicating the work is done for this step. This is used in tracking down the slowest AMP in the system. An AMP that has more last done messages than the others could be a bottleneck in the system performance.	FLOAT

Process Scheduling: CPU Utilization Columns

These columns represent CPU activities associated with this virtual processor, subdivided into 48 partitions. Partition 0 is reserved for use by PDE processes. For more information on the other partitions see [Partition Assignments](#).

For definitions of user service and user execution, see [ResUsageSvpr Table](#).

Column Name	Mode	Description	Data Type
CPUUServPart00 -CPUUServPart48	count	Time in centiseconds CPUs are busy in the partition doing user service. This is the system level time spent on a process.	FLOAT
CPUUExecPart00 -CPUUExecPart48	count	<p>Time in centiseconds CPUs are busy in the partition doing user execution. This is the user level time spent on a process.</p> <p>Note: Some CPU times may not be reported to these fields if child processes or threads are running on your system. For information about the difference in CPU usage reported and possible cause, see Process Scheduling: CPU Utilization Columns in ResUsageSps table.</p>	FLOAT

Process Scheduling: Cylinder Read Columns

These columns represent the File System resource usage statistics. The Cylinder Read feature uses these statistics for tracking performance and utilization.

Column Name	Mode	Description	Data Type
FileFcrRequests	count	<p>Total number of requests for the File System to use Cylinder Read.</p> <p>This column records the number of attempts to use Cylinder Read independent of whether the request will be issued to FSG or not. A request can be denied due to insufficient data blocks or because there is insufficient space in the FSG cache. Requests can also be denied at both the user and kernel level. Each of these items is counted in other FileFcr resource usage columns. This column can perform a number of calculations:</p> <ul style="list-style-type: none"> • Requests issued to FSG = FileFcrRequests - FileFcrDeniedUser • Successful Cylinder Reads = FileFcrRequests - FileFcrDeniedUser - FileFcrDeniedKern 	FLOAT
FileFcrRequestsAdaptive	count	<p>Number of adaptive requests from File System. This column records the number of requests for adaptive-style Cylinder Reads.</p> <p>Note: This column is not currently used.</p>	FLOAT

Column Name	Mode	Description	Data Type
FileFcrBlocksDeniedCache	count	<p>Number of data blocks contained in Cylinder Read requests rejected by the FSG subsystem due to insufficient cache.</p> <p>This column records the number of data blocks that were part of attempts to use Cylinder read that were denied by the FSG subsystem due to insufficient cache space; therefore, also incremented the FileFcrDeniedCache column.</p>	FLOAT
FileFcrBlocksDeniedKern	count	<p>Number of data blocks contained in the FSG subsystem rejected requests for Cylinder Read.</p> <p>This column records the number of data blocks that were part of attempts to use Cylinder Read that were denied by the FSG subsystem, and therefore, also incremented the FileFcrDeniedKern column.</p>	FLOAT
FileFcrBlocksDeniedUser	count	<p>Number of data blocks contained in the File System rejected requests for Cylinder Read.</p> <p>This column records the number of data blocks that were part of attempts to use Cylinder Read that were denied by the File System.</p>	FLOAT
FileFcrBlocksDeniedThreshKern	count	<p>Number of data blocks contained in Cylinder Read requests rejected for threshold by the FSG subsystem.</p> <p>This column records the number of data blocks that were part of attempts to use Cylinder read that were denied by the FSG subsystem due to the number of blocks being below the threshold, and therefore, also incremented the FileFcrDeniedThreshKern column.</p>	FLOAT
FileFcrBlocksDeniedThreshUser	count	<p>Number of data blocks contained in Cylinder Read requests rejected for threshold by the File System.</p> <p>This column records the number of data blocks that were part of attempts to use Cylinder Read that were denied by the File System due to the number of blocks being below the threshold, and therefore, also incremented the FileFcrDeniedThreshUser column.</p>	FLOAT
FileFcrBlocksRead	count	<p>Number of data blocks read in using Cylinder Read.</p> <p>This column records the total number of data blocks read in by successful Cylinder Read operations.</p> <p>The average number of data blocks in a successful Cylinder read can be calculated as follows: Average data blocks/ Cylinder Read = FileFcrBlocksRead / (FileFcrRequests - FileFcrDeniedUser - FileFcrDeniedKern)</p>	FLOAT

Column Name	Mode	Description	Data Type
FileFcrDeniedCache	count	Number of Cylinder Read requests denied by FSG due to insufficient cache. This column records the number of Cylinder Read requests denied due to insufficient FSG cache space for a cylinders worth of data.	FLOAT
FileFcrDeniedKern	count	Number of Cylinder Read requests denied by the FSG subsystem. This column records the number of Cylinder Read requests issued to the FSG subsystem which, for any reason, have been denied. A request can be denied due to insufficient data blocks (for example, the FileFcrDeniedThreshKern column) or because there is insufficient space in the FSG cache (for example, the FileFcrDeniedCache column). The FSG subsystem can reject a request containing insufficient data blocks that the File System thought had enough blocks because the FSG subsystem reduces the count by the number of data blocks that are already resident in the cache.	FLOAT
FileFcrDeniedUser	count	Number of Cylinder Read requests denied by the File System. This column records the number of Cylinder Read attempts denied by the File System. A request can be denied by the File System due to insufficient number of data blocks being requested (for example, the FileFcrDeniedThreshUser column). For information, see the FileFcrDeniedThreshUser column.	FLOAT
FileFcrDeniedThreshKern	count	Number of Cylinder Read requests denied by the FSG subsystem due to insufficient data blocks. This column records the number of Cylinder Read requests denied due to the data block threshold criteria. There is a minimum threshold of data blocks for an individual Cylinder Read request. If the number of data blocks is below this threshold, the overhead of the Cylinder Read operation is considered too large and issuing individual data block reads is considered more efficient. Therefore, the Cylinder Read request is denied. FSG must reevaluate the threshold for a request that the File System considered valid since FSG eliminates any data blocks from the request list that are already resident in the cache. This could reduce the count that the File System thought was above the threshold to one that is now below.	FLOAT
FileFcrDeniedThreshUser	count	Number of Cylinder Read requests denied by the File System due to insufficient data blocks.	FLOAT

Column Name	Mode	Description	Data Type
		This column records the number of Cylinder Read requests denied due to the data block threshold criteria. There is a minimum threshold of data blocks for an individual Cylinder Read request. If the number of data blocks is below this threshold, the overhead of the Cylinder Read operation is considered too large and issuing individual data block reads is considered more efficient. Therefore, the Cylinder Read request is denied.	

Process Scheduling: UDF CPU and Memory Columns

These columns report the UDF CPU and Memory data under the AMP, PE, or NODE.

All external routines are invoked by either an AMP or PE and dispatched to the UDF secure server for execution. Some UDF servers are specific to an AMP or PE, and some are for all AMPs or PEs on the node.

The number of secure servers specific to an AMP or PE can be set by the Cufconfig utility, and defaults to 2.

These server processes execute the C/C++ UDFs, XSPs, ExecR table operators, SCRIPT table operators (excluding their child processes), and Java UDFs with EXTERNAL SECURITY clause defined for the CREATE FUNCTION or REPLACE FUNCTION statement. The usage of these server processes is reported to the AMP or PE columns.

Java UDFs that are not defined with the EXTERNAL SECURITY clause are run using a hybrid server. Each node has 1 hybrid server which provides a multithreaded execution of Java UDFs for all AMPs and PEs on the node. The number of hybrid server threads can be set by Cufconfig, and defaults to 20. The usage of these server processes is reported to the NODE columns.

Child processes spawned by the SCRIPT table operators are run outside the secure servers. Therefore, their usage is not included in the column data.

PDE also reports UDF CPU usage to the node partition by design. Therefore, the CPUU[Exec|Serv]Part00 columns will report the UDF CPU usage, and the CPUU[Exec|Serv][Part11|Part13] will not.

The UDF CPU value over multiple periods averages 100% or less.

Note:

When an external routine forks a child process, the CPU value is not reported to these fields. As a result, the resource usage table shows a lower CPU usage than shown in the ResUsageSpma table, even if the external routine consumes a large amount of CPU time. If there are child processes running on your system, this may account for the larger CPU value reported in the ResUsageSpma table compared to this table.

Column Name	Mode	Description	Data Type
UdfServ	count	Reported system-level UDF CPU time value under the vproc. VprType determines if it is for an AMP or PE. If VprType is AMP, the field contains AMP CPU usage. If VprType is PE, the field contains PE usage. For other VprType, the field is 0.	FLOAT
UdfExec	count	Reported user-level UDF CPU time value under the vproc. VprType determines if it is for an AMP or PE. If VprType is AMP, the field contains AMP CPU usage. If VprType is PE, the field contains PE usage. For other VprType, the field is 0.	FLOAT
UdfMemInuseKB	track	Current total KB of resident memory in use by the UDF secure server processes. The usage is reported in AMP, PE, or NODE vprocs. For other VprType, the field is 0.	FLOAT
UdfVmSizeKB	track	Current total KB of virtual memory occupied by the UDF secure server processes. The usage is reported in AMP, PE, or NODE vprocs. For other VprType, the field is 0.	FLOAT

Process Scheduling: Process Block Count Columns

These columns identify how many times a process became blocked on which blocking type.

Column Name	Mode	Description	Data Type
ProcBlksDBLock	count	Number of process blocks for database locks.	FLOAT
ProcPendDBLock	track	Number of process blocks pending database locks.	FLOAT

Process Scheduling: Process Pending Wait Time Columns

Column Name	Mode	Description	Data Type
ProcWaitDBLock	count	Total time in centiseconds processes were blocked pending database locks.	FLOAT

Row Redistribution Columns

Column Name	Mode	Description	Data Type
RRDHCount	count	Total count of hashed row redistributions (RRDs) processed by the node-level buffered redistribution services (NLB). The RRDHCasc[All Most Some] fields count subsets of those RRDs based on the percentage of AMPs in the RRD cascade tree where the rows do not overflow the cascade buffer but are fully contained within the cascade rather than being sent directly to their targeted destination AMPs by NLB's minicast S2S-Redist	FLOAT

Column Name	Mode	Description	Data Type
		messages. This percentage is referred to as the "Cascade Factor" and varies between 0% where the RRD rows overflow the cascade buffer in every AMP, and 100% where the rows are cascaded all the way to the cascade root AMP and do not overflow the cascade buffer even there.	
RRDHCascAll	count	Count of hashed RRDs with a Cascade Factor of 100%. In this case all rows are cascaded up to the cascade root AMP without overflowing the cascade buffer anywhere, including in the cascade root itself.	FLOAT
RRDHCascMost	count	Count of hashed RRDs with a Cascade Factor of 90% or more, but less than 100%. In this case the rows are cascaded without a buffer overflow in most but not all AMPs, where "most" is based on a 90% super-majority, not a simple majority in the usual sense of anything over 50%.	FLOAT
RRDHCascSome	count	Count of hashed RRDs with a Cascade Factor greater than 0% but less than 90%. In this case the rows are cascaded without a buffer overflow in some AMPs, even as few as one, but less than "most" AMPs as defined by the 90% super-majority. Note: Note that there is no field (e.g., "RRDHCascNone") that counts the hashed RRDs with a Cascade Factor of exactly 0%, where the rows overflow the cascade buffer in every AMP. However, this virtual count can be computed from the other fields as follows: $RRDHCascNone = RRDHCount - (RRDHCascAll + RRDHCascMost + RRDHCascSome)$	FLOAT

Space Accounting Columns for ResUsageSvpr Table

Previously, the Teradata accounting system assumed an even distribution of rows to AMPs. The database and user space limits were set at the AMP level to the defined maximum limit values divided by the number of AMPs in the system. Space was managed and monitored at the AMP level.

Some features now allow tables on subsets of AMPs. The space accounting and space management infrastructure now manages the space limits at a global or system level and augments the AMP-level space management. These enhancements are referred to as *global space accounting*.

These columns report statistics for space accounting.

Column Name	Mode	Description	Data Type
SpaceProactiveAllocs	count	Total number of proactive allocations made in the AMP during the logging period. This field is populated by the spalocal task.	FLOAT

Column Name	Mode	Description	Data Type
SpaceGroupAllocs	count	Total number of group allocations initiated in the AMP during the logging period. This field is populated by the spaglobal task.	FLOAT
SpaceMsgsCnt	count	Total number of allocations or de-allocation messages processed by the space accounting task managing the system limit for a database or user. This field is populated by the spaglobal task.	FLOAT
SpaceAllocated	count	Total number of bytes of space allocated to the AMP. This field is populated by the spalocal task.	FLOAT
SpaceDeallocated	count	Total number of bytes of space de-allocated from the AMP. This field is populated by the spalocal task.	FLOAT

Teradata Virtual Storage (VS): Allocation Columns

These columns identify the allocation statistics reported by the Allocator.

Column Name	Mode	Description	Data Type
AllocatorExtentAllocReqs	count	Number of cylinder allocation requests received by the allocator.	FLOAT
AllocatorExtentFreeReqs	count	Number of cylinder free requests received by the allocator.	FLOAT
AllocatorMapIOsStarted	count	Number of map I/Os initiated by the allocator.	FLOAT
AllocatorMapIOsDone	count	Number of map I/Os completed by the allocator.	FLOAT

Teradata Virtual Storage (VS): Extent Driver I/O Columns

These columns identify the I/O statistics reported from the extent driver.

Column Name	Mode	Description	Data Type
ReadsCold	count	<p>Total number of reads issued to all cylinders that are considered COLD.</p> <p>A cylinder is considered to be COLD if the response time of its associated physical storage is less than TVS Percentile Cold Lower Bound configuration setting.</p> <p>Note: This column is not currently valid. It should not be used.</p>	FLOAT

Column Name	Mode	Description	Data Type
ReadsHot	count	<p>Total number of reads issued to all cylinders that are considered HOT.</p> <p>A cylinder is considered HOT if the response time of its associated physical storage is greater than the Teradata VS Percentile Hot Lower Bound configuration setting.</p> <p>Note: This column is not currently valid. It should not be used.</p>	FLOAT
ReadsWarm	count	<p>Total number of reads issued to all cylinders that are considered WARM.</p> <p>A cylinder is considered to be WARM if the response time of its associated physical storage is between TVS Percentile Hot Lower Bound and TVS Percentile Cold Upper Bound configuration settings.</p> <p>Note: This column is not currently valid. It should not be used.</p>	FLOAT
WritesCold	count	<p>Total number of writes issued to all cylinders that are considered COLD.</p> <p>Note: This column is not currently valid. It should not be used.</p>	FLOAT
WritesHot	count	<p>Total number of writes issued to all cylinders that are considered HOT.</p> <p>Note: This column is not currently valid. It should not be used.</p>	FLOAT
WritesWarm	count	<p>Total number of writes issued to all cylinders that are considered WARM.</p> <p>Note: This column is not currently valid. It should not be used.</p>	FLOAT
ReadResponseColdTotal	count	<p>Total read response time of all cylinders that are considered COLD</p> <p>Note: This column is not currently valid. It should not be used.</p>	FLOAT
ReadResponseHotTotal	count	<p>Total read response time of all cylinders that are considered HOT.</p> <p>Note: This column is not currently valid. It should not be used.</p>	FLOAT

Column Name	Mode	Description	Data Type
ReadResponseWarmTotal	count	Total read response time of all cylinders that are considered WARM. Note: This column is not currently valid. It should not be used.	FLOAT
WriteResponseColdTotal	count	Total write response time of all cylinders that are considered COLD. Note: This column is not currently valid. It should not be used.	FLOAT
WriteResponseHotTotal	count	Total write response time of all cylinders that are considered HOT. Note: This column is not currently valid. It should not be used.	FLOAT
WriteResponseWarmTotal	count	Total write response time of all cylinders that are considered WARM. Note: This column is not currently valid. It should not be used.	FLOAT
ReadResponseColdMax	max	Maximum read response time of all cylinders that are considered COLD. Note: This column is not currently valid. It should not be used.	FLOAT
ReadResponseHotMax	max	Maximum read response time of all cylinders that are considered HOT. Note: This column is not currently valid. It should not be used.	FLOAT
ReadResponseWarmMax	max	Maximum read response time of all cylinders that are considered WARM. Note: This column is not currently valid. It should not be used.	FLOAT
ReadResponseColdMin	min	Minimum read response time of all cylinders that are considered COLD. This column has a default value of the largest value of a 64-bit integer column. Note: This column is not currently valid. It should not be used.	FLOAT

Column Name	Mode	Description	Data Type
ReadResponseHotMin	min	<p>Minimum read response time of all cylinders that are considered HOT.</p> <p>This column has a default value of the largest value of a 64-bit integer column.</p> <p>Note:</p> <p>This column is not currently valid. It should not be used.</p>	FLOAT
ReadResponseWarmMin	min	<p>Minimum read response time of all cylinders that are considered WARM.</p> <p>This column has a default value of the largest value of a 64-bit integer column.</p> <p>Note:</p> <p>This column is not currently valid. It should not be used.</p>	FLOAT
WriteResponseColdMax	max	<p>Maximum write response time of all cylinders that are considered COLD.</p> <p>Note:</p> <p>This column is not currently valid. It should not be used.</p>	FLOAT
WriteResponseHotMax	max	<p>Maximum write response time of all cylinders that are considered HOT.</p> <p>Note:</p> <p>This column is not currently valid. It should not be used.</p>	FLOAT
WriteResponseWarmMax	max	<p>Maximum write response time of all cylinders that are considered WARM.</p> <p>Note:</p> <p>This column is not currently valid. It should not be used.</p>	FLOAT
WriteResponseColdMin	min	<p>Minimum write response time of all cylinders that are considered COLD.</p> <p>This column has a default value of the largest value of a 64-bit integer column.</p> <p>Note:</p> <p>This column is not currently valid. It should not be used.</p>	FLOAT
WriteResponseHotMin	min	<p>Minimum write response time of all cylinders that are considered HOT.</p> <p>This column has a default value of the largest value of a 64-bit integer column.</p> <p>Note:</p> <p>This column is not currently valid. It should not be used.</p>	FLOAT

Column Name	Mode	Description	Data Type
WriteResponseWarmMin	min	<p>Minimum write response time of all cylinders that are considered WARM.</p> <p>This column has a default value of the largest value of a 64-bit integer column.</p> <p>Note:</p> <p>This column is not currently valid. It should not be used.</p>	FLOAT

Teradata Virtual Storage (VS): Node Agent Columns

The Node Agent columns are populated only in Teradata VS. For more information on these columns, see *Teradata Vantage™ - Teradata® Virtual Storage*, B035-1179.

Column Name	Mode	Description	Data Type
NodeAgentMigrationsDone	count	Number of migration requests completed by the Node Agent.	FLOAT
NodeAgentMigrationsStarted	count	Number of migration requests started by the Node Agent.	FLOAT
NodeAgentStatProcessed	count	Number of statistics buffers processed by the Node Agent.	FLOAT

Transient Journal Purge Overhead Columns

Column Name	Mode	Description	Data Type
TJPurges	count	The number of purge passes in which a block-by-block scan is done.	FLOAT
TJDbPurgeReads	count	The number of blocks actually mapped in during the purge scan. This is a reasonable approximate measure of the I/O load. The system uses full-cylinder read mode, but the block count would still be roughly proportionate to the I/O load.	FLOAT
TJDbPurgeDeletes	count	<p>The number of blocks mapped in during the scan that were included in the ranges of blocks that were deleted.</p> <p>Before Write Ahead Logging (WAL), the ratio of deletes to reads would have been a useful measure of the effectiveness of the purge processing. However, with WAL, the ratio cannot be interpreted quite so simply because:</p> <ul style="list-style-type: none"> The range of deleted blocks could include blocks that were not actually mapped in (and therefore not counted). Blocks that contain only WAL records are not mapped in during the scan, as they are automatically filtered out. Under 	FLOAT

Column Name	Mode	Description	Data Type
		<p>typical conditions, there are probably relatively few such blocks. TJ and WAL records are typically generated in an interleaved sequence by regular SQL transactions. But during periods when the system workload is dominated by MultiLoad /FastLoad work, there will be relatively few TJ records written, so the proportion of WAL-only blocks would probably be significant.</p> <ul style="list-style-type: none"> • Post-WAL, neither TJDbPurgeReads nor TJDbPurgeDeletes gets incremented during a normal purge pass. Instead of scanning the active data blocks, a pointer to the oldest active transaction is maintained which is a quicker method. Therefore, PurgeTJ() can simply compute the bounds of the range of records that can be deleted in the part of the WAL /TJ that precedes the start of the oldest transaction. This does not require any scanning and the system cannot definitely determine how many blocks actually get deleted. • If the oldest transaction remains open for a long time, the quick purge method is not effective. Therefore, the system reverts back to the full scan method. <p>The TJDbPurgeReads and TJDbPurgeDeletes are only incremented during a full scan.</p>	

Reserved Columns

Column Name	Mode	Description	Data Type
ReservedS0	n/a	Reserved for future use.	CHAR(3)
ReservedS1	n/a	Reserved for future use.	CHAR(4)
Reserved00	n/a	Reserved for future use.	SMALLINT

Spare Columns

The ResUsageSvpr table spare fields are named Spare00 through Spare29, and SpareInt.

The SpareInt field has a 32-bit internal resolution while all other spare fields have a 64-bit internal resolution. All spare fields default to count but can be converted to min, max, or track mode fields if needed when they are used.

Column Name	Mode	Description	Data Type
Spare00 - Spare02	count	Reserved for future use.	FLOAT

Column Name	Mode	Description	Data Type
Spare03	count	Number of spool data segments that were forced to change their age to ageoutnow. In future use, this data will be reported in a permanent SVPR statistics column named FileSpoolForceAgeoutnow.	FLOAT
Spare04	count	Volume of spool data segments in KB that were forced to change their age to ageoutnow. In future use, this data will be reported in a permanent SVPR statistics column named FileSpoolForceAgeoutnowKB.	FLOAT
Spare05	count	Number of spool data segments that keep their age irrespective of FSGINFOSPOOLAGEOUTNOW flag. In future use, this data will be reported in a permanent SVPR statistics column named FileSpoolKeepAge.	FLOAT
Spare06	count	Volume of spool data segments in KB that keep their age irrespective of FSGINFOSPOOLAGEOUTNOW flag. In future use, this data will be reported in a permanent SVPR statistics column named FileSpoolKeepAgeKB.	FLOAT
Spare07	count	Reserved for future use.	FLOAT
Spare08	count	Number of times processes are blocked waiting for a lock on the CI. The permanent column name CISleeps is reserved for future use.	FLOAT
Spare09	count	Total amount of time in milliseconds that processes are blocked waiting for a lock on the CI. The permanent column name CISleepTime is reserved for future use.	FLOAT
Spare10	max	Maximum amount of time in milliseconds that processes are blocked waiting for a lock on the CI. The permanent column name CISleepTimeMax is reserved for future use.	FLOAT
Spare11 - Spare29	count	Reserved for future use.	FLOAT
SpareInt	track	This column reports the number of CPUs available for Teradata tasks at the end of the reporting period. These CPUs are TD-enabled CPUs. Teradata tasks only have access to TD-enabled CPUs while Non-Teradata tasks have access to all online CPUs. This count may change dynamically after database is up but is always less than or equal to the count reported in NCPUs. In future use, this data will be reported in a permanent SVPR statistics column named TDEnabledCPUs.	FLOAT

Related Information

For details on the different type of data fields, see [About the Mode Column](#).

Summary Mode

When Summary Mode is active for the ResUsageSvpr table, one row is written to the database for each type of vproc on each node in the system, summarizing the vprocs of that type on that node, for each log interval.

You can determine if a row is in Summary Mode by checking the SummaryFlag column for that row.

IF the SummaryFlag column value is...	THEN the data for that row is being logged...
'S'	in Summary Mode.
'N'	normally.

Resource Usage Views

The following provides information for accessing the resource usage views.

Note:

Views are the recommended method to access the resource usage table data.

To see the view definitions, execute `SHOW VIEW viewname`, where *viewname* is the name of the view whose most recent SQL create text is to be reported. For details on using the SHOW VIEW statement, see *Teradata Vantage™ - SQL Data Definition Language Detailed Topics*, B035-1184.

The following views report the table column, GroupId. A homogeneous system requires no changes to use this macro because all the nodes will be assigned to group A. For a coexisting system, however, the values need to be set up when the system is installed or reconfigured so that each type of node is assigned to a specific group ID.

Since each of the views include all the columns for the represented table, the views will be changed whenever the base table columns are changed.

Note:

Do not change or delete columns in these views. If the columns are modified, the resource usage macros that use these views may not work properly. You can, however, safely add columns.

View	Description
ResCPUUsageByAMPView	Describes CPU usage per AMP
ResCPUUsageByPEView	Describes CPU usage by each PE
ResSawtView	Based on the ResUsageSawt table
ResScpuView	Based on the ResUsageScpu table
ResShstView	Based on the ResUsageShst table
ResSldvView	Based on the ResUsageSldv table
ResSmhmView	Based on the ResUsageSmhm table. Reserved for future use.
ResSpdskView	Based on the ResUsageSpdsk table
ResSpmaView	Based on the ResUsageSpma table
ResSpsView	Based on the ResUsageSps table
ResSvdskView	Based on the ResUsageSvdsk table
ResSvprView	Based on the ResUsageSvpr table.

Resource Usage Macros

The following describes the output format of the resource usage macros and each macro.

Macro Output Format

Resource usage macros provide output in the following general format.

<Report Date>			<Title of Report>			Page <num>	

Date	Time	Type	1st Id	2nd Id	1st Stat	2nd Stat	3rd Stat ...

99/99/99	99:99:99	AAAA	999-99	999-99	999.99%	99999.99	99999.99
					999.99%	99999.99	99999.99
						
		AAAA	999-99	999-99	999.99%	99999.99	99999.99
			999-99	999-99	999.99%	99999.99	99999.99
99:99:99		AAAA	999-99	999-99	999.99%	99999.99	99999.99
						

where:

Column	Description
Date	The date at the end of a log interval.
Time	The time at the end of a log interval. Statistics on each line cover the time period ending at the indicated time.
Type	A virtual processor type, logical device type, host type, or a special type for certain reports.
1st ID, 2nd ID, etc.	The appropriate identifier, which varies, depending on the macro. It is one or more of the following: <ul style="list-style-type: none"> • NodeID • VprocID • HostID • GroupID
1st Stat, 2nd Stat, etc.	The appropriate statistics. Details are given with the descriptions of each macro. Numbers are generally displayed with the appropriate fixed format (for example, 'zzzz9.99') unless the number represents a percentage or sum of percentages. Percentages are displayed with the appropriate format (for example, 'zz9.9%', 'zz9' or 'zz9.99').

Unless otherwise specified, all statistical numbers are expressed as either:

- Percentage
- Millisecond (ms)
- Kilobyte (KB)

Columns whose values depend on the logging rate are never reported as raw data. Instead, they are converted to a normalized value, such as *per second*.

All reports are ordered by date, time, type, 1st ID, 2nd ID, and so forth. Repeated date, time, type, and ID column values are suppressed until a new row presents a different value.

Question Marks

Question marks used as values in the output examples are generated when a division by zero is made. It represents data that is not available. The numbers in the columns are calculated, for example, by dividing KB by number of blocks read. When there are no blocks read, KB is divided by zero. A question mark does not mean there is an error, but indicates that there is no information to report for this time period.

Usage Notes

To get current data, logging must be enabled on the resource usage table used by the view or macro.

ResAWT Macros

Function

Macro...	Reports the average AMP Worker Task...
ResAWT	in use for all AMPs in the system.
ResAWTByAMP	in use for each AMP.
ResAWTByNode	on all AMPs in each node.

Input Format Examples

The input forms of these three macros are described below.

```
EXEC ResAWT
( FromDate,ToDate,FromTime,ToTime );
```

```
EXEC ResAWTByAMP
( FromDate,ToDate,FromTime,ToTime );
```


EXEC ResAWTByNode

```
( FromDate,ToDate,FromTime,ToTime,FromNode,ToNode );
```

See [Macro Execution](#) for a description of the *FromDate*, *ToDate*, *FromTime*, *ToTime*, *FromNode*, *ToNode* and *Node* parameters.

Usage Notes

You must have logging enabled on the ResUsageSawt table.

ResAWT Macros Output

The reports generated by the ResAWT, ResAWTByAMP, and ResAWTByNode macros are described in this section.

In the ResAWT output report, the statistics columns, after the Date and Time columns, provide a summary of the AMP Worker Tasks resource usage.

The following table describes the statistics columns, after the Date and Time columns, in the ResAWTByAMP output.

Statistics columns	Description
1	Node ID.
2	AMP ID.
3 through 24	Summary of AMP Worker Tasks resource usage.

The following table describes the statistics columns, after the Date and Time columns, in the ResAWTByNode output report.

Statistics columns	Description
1	Node ID.
2 through 23	Summary of AMP Worker Tasks resource usage.

The following table describes the columns in all output reports (with the exception of ResAWTByNode, which has the NodeID column, and ResAWTByAMP, which has the Node ID and AMP ID columns).

Column...	Reports the...
Mail box Depth	current depth of the AMP work mailbox.
In Flow Ctl	AMP that is or is not in flow control.
Flow Ctls Per Sec	number of times during the log period that the system entered the flow control state from a non-flow controlled state.

Column...	Reports the...
Work New AWTs	current number of AMP Worker Tasks in use during the log period for each new first-level secondary work type for the Vprld vproc.
Work One AWTs	current number of AMP Worker Tasks in use during the log period for each first-level secondary work type for the Vprld vproc.
New + One AWTs	summary of the previous two columns, Work New AWTs and Work One AWTs.
Work Two AWTs	current number of AMP Worker Tasks in use during the log period for each second-level secondary work type for the Vprld vproc.
Work 3 AWTs	current number of AMP Worker Tasks in use during the log period for each third-level secondary work type for the Vprld vproc.
Work Abrt AWTs	current number of AMP Worker Tasks in use during the log period for each transaction abort request for the Vprld vproc.
Work Spwn AWTs	current number of AMP Worker Tasks in use during the log period for each spawned abort request for the Vprld vproc.
Work Norm AWTs	current number of AMP Worker Tasks in use during the log period for each message that does not fall within the standard work type hierarchy for the Vprld vproc.
Work Ctl AWTs	Note: This column is not currently used.
Work Exp AWTs	current number of AMP Worker Tasks in use during the log period for each expedited Allocation Groups for the Vprld vproc. (Expedited Allocation Groups exist when using the reserved AMP Worker Task pool.
Max Work New AWTs	the maximum number of AMP Worker Tasks in use at one time during the log period for each new work type for the Vprld vproc.
Max Work One AWTs	the maximum number of AMP Worker Tasks in use at one time during the log period for each first-level secondary work type for the Vprld vproc.
Max Work Two AWTs	the maximum number of AMP Worker Tasks in use at one time during the log period for each second-level secondary work type for the Vprld vproc.
Max Work 3 AWTs	the maximum number of AMP Worker Tasks in use at one time during the log period for each third-level secondary work type for the Vprld vproc.
Max Work Abrt AWTs	the maximum number of AMP Worker Tasks in use at one time during the log period for each transaction abort request for the Vprld vproc.
Max Work Spwn AWTs	the maximum number of AMP Worker Tasks in use at one time during the log period for each spawned abort request for the Vprld vproc.
Max Work Norm AWTs	the maximum number of AMP Worker Tasks in use at one time during the log period for each message that does not fall within the standard work type hierarchy for the Vprld vproc.
Max Work Ctl AWTs	Note: This column is not currently used.

Column...	Reports the...
Max Work Exp AWTs	the maximum number of AMP Worker Tasks in use at one time during the log period for each expedited Allocation Groups for the Vprld vproc. (Expedited Allocation Groups exist when using the reserved AMP Worker Task pool.

For a complete description of the columns above, see [ResUsageSawt Table](#).

ResAWT Examples

```
exec resawt(,,);

exec resawtByAMP(,,);

exec resawtByNode(,,,,);
```

ResCPUByAMP Macros

Function

Macro...	Reports the following...
ResCPUByAMP	how each AMP on each node utilizes the CPUs.
ResCPUByAMPOneNode	how each AMP on a specific node utilizes the CPUs.
ResAmpCpuByGroup	the summary of AMP CPU usage by node grouping.

Input Format Examples

The input forms of these three macros are described below.

```
EXECUTE ResCPUByAMP
  (FromDate, ToDate, FromTime, ToTime, FromNode, ToNode) ;

EXECUTE ResCPUByAMPOneNode
  (FromDate, ToDate, FromTime, ToTime, Node) ;

EXECUTE ResAMPCpuByGroup
  (FromDate, ToDate, FromTime, ToTime) ;
```

See [Macro Execution](#) for a description of the *FromDate*, *ToDate*, *FromTime*, *ToTime*, *FromNode*, *ToNode* and *Node* parameters.

Usage Notes

You must have logging enabled on the ResUsageSvpr table.

If parallel efficiency, as shown in the ResUsageSpma table, is not approaching 100% look for the reasons for the imbalance; use this macro to find AMP-level CPU skew.

ResCPUByAMP Macros Output

The reports generated by the ResCPUByAMP, the ResCPUByAMPOneNode, and the ResAmpCpuByGroup macros, output the following:

Column...	Reports the percent of time AMPs were busy doing user...
Awt User Serv%	service for the AMP Worker Task (Awt) partition.
Misc User Serv%	service for miscellaneous (all other except Partition 0) AMP partitions.
Awt User Exec%	execution within the AMP Worker Task (Awt) partition.
Misc User Exec%	execution within miscellaneous (all other except Partition 0) AMP partitions.
Total User Serv%	service work. This is the sum of the Awt User Serv%, the Misc User Serv%, and AMP Partition 0 user service%. Note: User service is the time that a CPU is busy executing kernel system calls or servicing I/O and timer hardware interrupts.
Total User Exec%	execution work. This is the sum of the Awt User Exec%, Misc User Exec%, and AMP Partition 0 user execution. Note: User execution is the time a CPU is busy executing user execution code, which is the time spent in a user state on behalf of a process.
Total Busy%	service and execution work. This is the sum of the Total User Serv% and the Total User Exec% columns.

Note:

The above CPU statistics represent the aggregate of all time spent in the indicated way by all CPUs on the node. Because there are multiple CPUs, the Total Busy % should be compared to a theoretical maximum of 100% times the number of CPUs on the node.

The Node CPU column reports the number of CPUs (NCPUs).

The NodeID column only appears in the ResCPUByAMP report.

The GroupID column only appears in the ResAmpCPUByGroup report.

See also the information about AWT Monitor (awtmon) in *Teradata Vantage™ - Database Utilities*, B035-1102.

ResCPUByAMP Examples

```
exec resCPUByAMP(,,,,);

exec resCPUByAMPOneNode(,,,,);

exec resAmpCpuByGroup(,,);
```

Normalized Viewing of CPU Usage by AMP

Some users may prefer to view CPU usage by AMP in a normalized fashion. Conceptually, this restates each of the above statistics in terms of percentage of total CPU capacity of the node.

The following SQL example shows how to perform this normalization for the Total Busy % statistic.

```
SEL TheDate, TheTime, Vproc, NodeID,
  (AmpTotalUserExec+AmpTotalUserServ)
 /Secs/NCPUs
 (FORMAT 'zz9%',TITLE 'Total// Busy// %')
FROM ResCpuUsageByAMPView
WHERE TheDate = CURRENT_DATE AND TheTime>080000
ORDER BY 1,2,3;
```

ResCPUByPE Macros

Function

Macro...	Reports...
ResCPUByPE	how each PE on each node is utilizing the CPUs.
ResCPUByPEOneNode	how each PE on a specific node is utilizing the CPUs.
ResPeCpuByGroup	the PE CPU utilization summarized by a node grouping.

Input Format Examples

The input forms of the these three macros are described below.

```
EXEC ResCPUByPE
  (FromDate,ToDate,FromTime,ToTime,FromNode,ToNode);
```

```
EXEC ResCPUByPEOneNode  
(FromDate, ToDate, FromTime, ToTime, Node);
```

```
EXEC ResPeCpuByGroup  
(FromDate, ToDate, FromTime, ToTime);
```

See [Macro Execution](#) for a description of the *FromDate*, *ToDate*, *FromTime*, *ToTime*, *FromNode*, *ToNode* and *Node* parameters.

Usage Notes

You must have logging enabled on the ResUsageSvpr table.

ResCPUByPE Macros Output

The reports generated by the ResCPUByPE, ResCPUByPEOneNode, and ResPeCPUByGroup macros, output the following:

Column...	Reports the percent of time PEs are busy doing user...
Disp User Serv%	service for the Dispatcher partition of the PE.
Ses User Serv%	service for the Session Control partition of the PE.
Misc User Serv%	service for miscellaneous (all other, except Partition 0) PE partitions.
Disp User Exec%	execution within the Dispatcher partition of the PE.
Ses User Exec%	execution within the Session Control partition of the PE.
Misc User Exec%	execution within miscellaneous (all other, except Partition 0) PE partitions.
Total User Serv%	service work. This is the sum of the four user service columns above plus PE Partition 0 user service.
Total User Exec%	execution work. This is the sum of the four user execution columns above plus PE Partition 0 user execution.
Total Busy%	service and execution work. This is the sum of the Total User Serv% and the Total User Exec% columns.

Note:

The above CPU statistics represent the aggregate of all time spent in the indicated way by all CPUs on the node. Because there are multiple CPUs, the Total Busy % should be compared to a theoretical maximum of 100% times the number of CPUs on the node.

The Node CPU column in the following sample outputs reports the number of CPUs (NCPUs).

The NodeID column only appears in the ResCPUByPE output report.

The GroupID column only appears in the ResPeCpuByGroup output report.

ResCPUByPE Examples

```
exec resCPUByPE(,,,,);

exec resCPUByPEOneNode(,,,,);

exec resPeCpuByGroup(,,);
```

Normalized Viewing of CPU Usage by PE

Some users may prefer to view CPU usage by PEs in a normalized fashion. Conceptually, this restates each of the above statistics in terms of percentage of total CPU capacity of the node.

The following SQL example shows how to perform this normalization for the Total Busy % statistic.

```
SEL TheDate, TheTime,Vproc,NodeID,
(PETotalUserExec+PETotalUserServ)
/Secs/NCPUs
(FORMAT 'zz9%',TITLE 'Total// Busy// %')
FROM ResCpuUsageByPEView
WHERE TheDate = CURRENT_DATE AND TheTime>080000
ORDER BY 1,2,3;
```

ResCPUByNode Macros

Function

Macro...	Reports how...
ResCPUByNode	each individual node is utilizing its CPUs.
ResCPUOneNode	a specific node is utilizing its CPUs.
ResCPUByGroup	a specified Node Group is utilizing the system CPUs.

Input Format Examples

The input forms of these three macros are described below.

```
EXEC ResCPUByNode
( FromDate, ToDate, FromTime, ToTime, FromNode, ToNode );
```

```
EXEC ResCPUOneNode
( FromDate, ToDate, FromTime, ToTime, Node );
```

```
EXEC ResCPUByGroup
( FromDate, ToDate, FromTime, ToTime );
```

See [Macro Execution](#) for a description of the *FromDate*, *ToDate*, *FromTime*, *ToTime*, *FromNode*, *ToNode* and *Node* parameters.

Usage Notes

You must have logging enabled on the ResUsageSpma table.

ResCPUByNode Macros Output

The reports generated by the ResCPUByNode, the ResCPUOneNode, and the ResCPUByGroup macros, output the following information.

The following columns are the averages for all CPUs on the node.

This column ...	Lists percentage of time spent ...
I/O Wait %	idle and waiting for I/O completion.
Total User Serv %	busy doing user service work.
Total User Exec %	busy doing user execution work.
Total Busy %	busy doing user service and execution work. This is the sum of Total User Serv % and the Total User Exec % columns.

where:

This variable...	Describes the time a CPU is busy executing...
User service	kernel system calls or servicing I/O and timer hardware interrupts.
User execution	user execution code, which is the time spent in a user state on behalf of a process.

The NodeID column only appears in the ResCPUByNode output report.

The GroupID column only appears in the ResCpuByGroup output report.

ResCPUByNode Examples

```
exec resCpuByNode(,,,,);

exec resCpuOneNode(,,,,);
```



```
exec resCpuByGroup(,,);
```

ResHostByLink Macros

Function

Macro...	Reports the host traffic for...
ResHostByLink	every communication link in the system.
ResHostOneNode	the communication links of a specific node.
ResHostByGroup	the communication links of a node grouping.

Input Format Examples

The input forms of these three macros are described below.

```
EXEC ResHostByLink  
( FromDate,ToDate,FromTime,ToTime );
```

Note:

The ResHostByLink macro syntax does not include the *FromNode* and *ToNode* parameters to specify a range of nodes.

```
EXEC ResHostOneNode  
( FromDate,ToDate,FromTime,ToTime,Node );
```

```
EXEC ResHostByGroup  
( FromDate,ToDate,FromTime,ToTime );
```

See [Macro Execution](#) for a description of the *FromDate*, *FromTime*, *ToDate*, *ToTime*, and *Node* parameters.

Usage Notes

The ResHostByLink macros help you answer the following questions:

- Is my set up correct?
- Am I making good use of the channels? If not, how high are they? If not high, then there may not be enough host resources.

Study the incoming traffic. Problems with incoming traffic may be simply caused by an incorrect configuration. Once configured correctly, if there is still a traffic problem, consider studying the TCP/IP network traffic, for example, when doing an export, the ResUsageSpma table may show 30 million rows/log period.

There must be ResUsageShst table rows logged for the period of time that the macro is going to produce a report for.

ResHostByLink Macros Output

The reports generated by the ResHostByLink, the ResHostOneNode, and the ResHostByGroup macros, output the following:

Column...	Reports the...
Host Type	type of host connection: <ul style="list-style-type: none"> • IBMECS • IBMMUX • IBMNET • NETWORK
Host Id	Logical ID of host (or client) with sessions logged on.
KBs Read/ Sec	number of KB read per second.
KBs Write/ Sec	number of KB written per second.
Blks Read/ Sec	number of successful blocks read per second.
Blks Write/ Sec	number of successful blocks written per second.
Blk Read Fail %	percentage of block read attempts that failed.
Blk Write Fail %	percentage of block write attempts that failed.
KBs/Blk Read	average number of KB per block read.
KBs/Blk Write	average number of KB per block written.
Msgs/Blk Read	average number of messages per block read.
Msgs/Blk Write	average number of messages per block written.
Avg ReqQ Len	average number of messages queued for output to the host.
Max ReqQ Len	maximum number of messages queued for output to the host.

The NodeID column only appears in the ResHostByLink output report.

The GroupID column only appears in the ResHostByGroup output report.

ResHostByLink Examples

```
exec resHostByLink(,,);

exec resHostOneNode(,,,);

exec resHostByGroup(,,);
```

ResLdvByNode Macros

Function

Macro...	Reports the logical device traffic channeled through...
ResLdvByNode	each node by totaling its controller links into one summarized node output line.
ResLdvOneNode	a specific node by totaling all its controller links into one summarized node output line.
ResLdvByGroup	a node grouping.

Input Format Examples

The input forms of these three macros are described below.

```
EXEC ResLdvByNode
( FromDate,ToDate,FromTime,ToTime,FromNode,ToNode );

EXEC ResLdvOneNode
( FromDate,ToDate,FromTime,ToTime,Node );

EXEC ResLdvByGroup
( FromDate,ToDate,FromTime,ToTime );
```

See [Macro Execution](#) for a description of the *FromDate*, *FromTime*, *ToDate*, *ToTime*, *FromNode*, *ToNode* and *Node* parameters.

Usage Notes

You must have logging enabled on the ResUsageSldv table.

ResLdvByNode Macros Output

The reports generated by the ResLdvByNode, the ResLdvOneNode, and the ResLdvByGroup macros, output the following:

Column...	Reports the...
Reads / Sec	average number of logical device reads per second.
Writes / Sec	average number of logical device writes per second.
Rd KB/ I/O	average number of KB per logical device read.
Wrt KB / I/O	average number of KB per logical device write.
Avg I/O Resp	average response time for a logical device read or write in seconds.
Max Concur Rqsts	maximum number of concurrent requests during the log period.
Avg Out Rqsts	average number of outstanding requests.
Out Rqst Time %	percent of time there are outstanding requests.

The NodeID column only appears in the ResLdvByNode output report.

The GroupID column only appears in the ResLdvByGroup output report.

ResLdvByNode Examples

```
exec resLdvByNode(,,,,);

exec resLdvOneNode(,,,,);

exec resLdvByGroup(,,);
```

ResMemMgmtByNode Macros

Function

Macro...	Reports memory management activity for...
ResMemMgmtByNode	each individual node.
ResMemMgmtOneNode	a specific node.
ResMemByGroup	a node grouping.

Input Format Examples

The input forms of these three macros are described below.

```
EXEC ResMemMgmtByNode
( FromDate,ToDate,FromTime,ToTime,FromNode,ToNode );
```

```
EXEC ResMemMgmtOneNode
( FromDate,ToDate,FromTime,ToTime,Node );
```

```
EXEC ResMemByGroup
( FromDate,ToDate,FromTime,ToTime );
```

See [Macro Execution](#) for a description of the *FromDate*, *FromTime*, *ToDate*, *ToTime*, *FromNode*, *ToNode* and *Node* parameters.

Usage Notes

You must have logging enabled on the ResUsageSpma table.

ResMemMgmtByNode Macros Output

The reports generated by the ResMemMgmtByNode, the ResMemMgmtOneNode, and the ResMemByGroup macros, output the following:

Column...	Reports the...
% Mem Free	current snapshot of the percent of memory that is unused.
Text Alocs/ Sec	average number of text page allocations per second. text pages are allocations of memory for code that is not associated with system-level overhead tasks.
VPR Alocs/ Sec	average number of vproc-specific page and segment allocations per second on a node.
KB/ VPR Aloc	average KB per vproc-specific page and segment allocation on a node.
Aloc Fail %	percent of memory allocation attempts that failed.
Ages/ Sec	average number of times memory pages were aged out per second.
# Proc Swp	current number of processes that are swapped out.
Page Drops/ Sec	average number of text pages dropped from memory per second. Page drops are text pages that are dropped from memory to increase the amount of available memory.
Page Reads KBs/ Sec	average number of memory paging KB, read from disk per second.

Column...	Reports the...
	Page reads include both memory text pages and task context pages, such as scratch, stack, etc.
Page Writes/ Sec	average number of memory pages written to disk per second. Page writes include only task context pages.
Swap Drops/ Sec	average number of disk segments dropped from memory per second. Swap drops include all disk segments dropped from memory because their ancestor processes were swapped out.
Swap Reads/ Sec	average number of disk segments reread back into memory, after being swapped, out per second. Swap reads include all reread disk segments that had been previously dropped from memory because their ancestor processes were swapped out.
KB/Swp Drp	average size, in KB, of disk segments dropped from memory because their ancestor processes were swapped out.
KB/Swp Rd	average size, in KB, of reread disk segments that had been previously dropped from memory because their ancestor processes were swapped out.
P+S Drops/ Sec	average number of paged, swapped page, or segment drops per second. This statistic includes both the memory text pages (Pg Drps/ Sec), and the disk segments (Swp Drps/ Sec), that were dropped.
P+S Read KBs/ Sec	average total number of paged, swapped page, or segment read KB, per second. Includes both the memory text pages and task context pages (Pg Rds/ Sec), and the disk segments (Swp Rds/ Sec), reread back into memory after being swapped out.
P+S Writes/ Sec	average total number of paged, swapped page, or segment writes per second.
P+S IO KB %	percent of logical device input and output KB done for paging or swapping inputs and outputs.

The NodeID column only appears in the ResMemMgmtByNode output report.

The GroupID column only appears in the ResMemByGroup output report.

ResMemMgmtByNode Examples

```
exec resMemMgmtByNode(,,,,);
```

```
exec resMemMgmtOneNode(,,,,);
```

```
exec resMemByGroup(,,);
```

ResNetByNode Macros

Function

Macro...	Reports net traffic for...
ResNetByNode	each node.
ResNetOneNode	a specific node.
ResNetByGroup	nodes summarized by node groups.

Input Format Examples

The input forms of these three macros are described below.

```
EXEC ResNetByNode
( FromDate, ToDate, FromTime, ToTime, FromNode, ToNode );
```

```
EXEC ResNetOneNode
( FromDate, ToDate, FromTime, ToTime, Node );
```

```
EXEC ResNetByGroup
( FromDate, ToDate, FromTime, ToTime );
```

See [Macro Execution](#) for a description of the *FromDate*, *FromTime*, *ToDate*, *ToTime*, *FromNode*, *ToNode* and *Node* parameters.

Usage Notes

You must have logging enabled on the ResUsageSpma table.

ResNetByNode Macros Output

The reports generated by the ResNetByNode, the ResNetOneNode, and the ResNetByGroup macros, output the following:

Column...	Reports the...
% Retries	percent of total net circuit attempts that caused software backoffs (BNS service-blocked occurrences).

Column...	Reports the...
	Note: This value reflects how many times the hardware backed off a connection because the switch nodes could not route to the end point. That implies that the end point was busy or, in switch node terms, the routing path was busy. A value over 100% does not imply a problem, but shows that there were multiple attempts to send new messages while the Bynet path was busy. On a busy system, this can be a normal level of activity.
Total Reads/ Sec	average number of net reads per second.
Total Writes/ Sec	average number of net writes per second.
Total IOs/ Sec	average number of net reads and writes per second.
KB/ IO	average KB per net read or write.
% PtP	percent of total net reads and writes that are point-to-point reads and writes.
% Brd	percent of total net reads and writes that are broadcast reads and writes.

Note:

In the following examples, the NodeID column appears only in the ResNetByNode output report. The GroupID column only appears in the ResNetByGroup output report. For all the examples, the values in the Total Reads/ Sec and Total Writes/ Sec are expected to be equal on SMP (single-node, vnet) systems.

ResNetByNode Examples

```
exec resNetByNode(,,,,);

exec resNetOneNode(,,,,);

exec resNetByGroup(,,,);
```

ResNode Macros**Function**

Macro...	Provides a summary of resource usage ...
ResNode	averaged across all nodes, excluding any PE-only nodes.
ResOneNode	by returning the node requested.
ResNodeByNode	by returning the nodes requested.

Macro...	Provides a summary of resource usage ...
ResNodeByGroup	by defined groups of nodes.

Input Format Examples

The input forms of these four macros are described below.

```
EXEC ResNode
( FromDate,ToDate,FromTime,ToTime );
```

Note:

The ResNode macro syntax does not include the *FromNode* and *ToNode* parameters to specify a range of nodes.

```
EXEC ResOneNode
( FromDate,ToDate,FromTime,ToTime,Node );
```

```
EXEC ResNodeByNode
( FromDate,ToDate,FromTime,ToTime,FromNode,ToNode );
```

```
EXEC ResNodeByGroup
( FromDate,ToDate,FromTime,ToTime );
```

See [Macro Execution](#) for a description of the *FromDate*, *ToDate*, *FromTime*, *ToTime*, *FromNode*, *ToNode*, and *Node* parameters.

Usage Notes

You must have logging enabled on the ResUsageSpma table.

ResNode Macros Output

The reports generated by the ResNode, the ResOneNode, the ResNodebyNode, and the ResNodeByGroup macros, output the following.

The following table describes the statistics columns, after the Date and Time columns, in the ResNode output report.

Statistics columns	Description
1 through 3	CPU usage.
4 through 8	Logical device interface.

Statistics columns	Description
9 through 14	Memory interface.
15 through 17	Net interface.
18 and 19	General node process scheduling.

The following table describes the statistics columns, after the Date and Time columns, in the ResOneNode output report.

Statistics columns	Description
1 and 2	CPU usage.
3 through 6	Logical device interface.
7 through 11	Memory interface.
12 through 14	Net interface.
15 and 16	General node process scheduling.

The following table describes the statistics columns, after the Date and Time columns, in the ResNodebyNode output report.

Statistics columns	Description
1 and 2	CPU usage.
3 through 6	Logical device interface.
7 through 12	Memory interface.
13 through 15	Net interface.
16 and 17	General node process scheduling.

The following table describes the statistics columns, after the Date and Time columns, in the ResNodeByGroup output report.

Statistics columns	Description
1	GroupId as defined in the associated view as a grouping of one or more nodes.
2 and 3	CPU usage.
4 through 7	Logical device interface.
8 through 12	Memory interface.
13 through 15	Net interface.

Statistics columns	Description
16 and 17	General node process scheduling.

The following table describes the statistics columns in all output reports (with the exception of ResNodeByNode, which has a NodeID column, and ResNodeByGroup, which has a GroupID column).

Column...	Reports the...
CPU Bsy %	percent of time the CPUs are busy, based on average CPU usage per node.
CPU Eff % (ResNode report)	parallel efficiency of node CPU usage. Parallel efficiency is the total percent of time nodes are busy. It is the average for all nodes of total busy divided by the total busy time of the busiest node.
WIO %	percent of time the CPUs are idle and waiting for completion of an I/O operation.
Ldv IO KBs /Sec	average number of logical device read and write KB per second.
Ldv Eff % (ResNode report)	parallel efficiency of the logical device (disk) I/Os. It is the average number of I/Os per node divided by the number of I/Os performed by the node with the most I/Os.
P+S % of IO KBs	percent of logical device read and write KB that are for paging purposes.
Read % of IO KBs	percent of logical device read and write KB that are reads.
Ldv KB / IO	average size of a non-swap/page logical device read or write.
Fre Mem %	percent of memory that is unused.
TPtP IOs /Sec	total point-to-point net reads and writes per second, per node.
TMIIOs /Sec	total multicast (broadcast or merge) net reads and writes per second, per node.
Net Rtry %	percent of transmission attempts that resulted in retries.
Prc Blks / Sec	number of times per second, per node, that processes other than message and timer waits are blocked.
ms /Blk	average time, in milliseconds, spent waiting for a blocked process other than message and timer waits.
Net Rx % Bsy	percent of time the network was busy either receiving.
Net Tx % Bsy %	percent of time the network was busy transmitting.

ResNode Examples

```
exec resNode(,,,);
```

```
exec resOneNode(,,,,);
```

```
exec resNodeByNode(,,,,);
```

```
exec resNodeByGroup(,,);
```

ResPdskByNode Macros

Function

Macro...	Reports the device traffic...
ResPdskByNode	by a physical node.
ResPdskOneNode	for a specified node.
ResPdskByGroup	node grouping.

Input Format Examples

The input forms of these three macros are described below.

```
EXEC ResPdskByNode  
( FromDate,ToDate,FromTime,ToTime,FromNode,ToNode );
```

```
EXEC ResPdskOneNode  
( FromDate,ToDate,FromTime,ToTime,Node );
```

```
EXEC ResPdskByGroup  
( FromDate,ToDate,FromTime,ToTime );
```

See [Macro Execution](#) for a description of the *FromDate*, *FromTime*, *ToDate*, *ToTime*, *FromNode*, *ToNode* and *Node* parameters.

Usage Notes

You must have logging enabled on the ResUsageSpdsk table.

ResPdskByNode Macros Output

The following table describes the statistics columns in all output reports (note, ResPdskByNode reports by NodeID columns and ResPdskByGroup reports by NodeType column).

Column...	Reports the...
Reads/Sec	average number of device reads per second.
Writes/Sec	average number of device writes per second.
Rd KB/ I/O	average number of KB per device read.
Wrt KB/ I/O	average number of KB per device write.
Avg I/O Resp	average response time for a device read or write in seconds.
Max Concur Rqsts	maximum number of concurrent requests during the log period.
Out Rqst Time %	percent of time there are outstanding requests.

ResPdskByNode Examples

```
exec resPdskByNode(,,,,);
```

```
exec resPdskOneNode(,,,,);
```

```
exec resPdskByGroup(,,);
```

ResPs Macros

Function

Macro...	Provides a summary of the Priority Scheduler resource usage...
ResPsByNode	The ResPsByNode macro produces one row of data for every set as defined in the GROUP BY clause.
ResPsByGroup	by coexistence group, produces one set of rows of data for each for each node type in the system for each log interval.

Input Format Examples

The input forms of the macros are described below.

EXEC ResPsByNode

```
( FromDate,ToDate,FromTime,ToTime,FromNode,ToNode );
```

EXEC ResPsByGroup

```
( FromDate,ToDate,FromTime,ToTime );
```

See [Macro Execution](#) for a description of the *FromDate*, *ToDate*, *FromTime*, *ToTime*, *FromNode*, *ToNode*, and *Node* parameters.

Note:

Coexistence support can be accomplished using the *NodeType* column to do a *group by* in SQL directly. Therefore, the *GroupId* column is not needed and the *ResUsageSps* table view is not provided.

Usage Notes

To use ResPs macros, existing rows must exist in the table.

These macros can be used to report historical data.

ResPs Macros Output

The reports in the following sections are sample output reports from the *ResPsByNode* and *ResPsByGroup* macros.

The following table describes the statistics columns, after the *Date* and *Time* columns, in the *ResPsByNode* output report.

Statistics columns	Description
1	Node ID.
2	pWDId.
3 through 12	Summary of the Priority Scheduler statistics.

The following table describes the statistics columns, after the *Date* and *Time* columns, in the *ResPsByGroup* output report.

Statistics columns	Description
1	Node type.
2 through 11	Summary of the Priority Scheduler statistics.

The following table describes the statistics columns in all output reports (with the exception of *ResSpsByNode*, which reports by *NodeID* columns, and *ResSpsByGroup*, which reports by *NodeType* column).

Column...	Reports the...
CPU Bsy %	percent of time the CPUs are busy, based on average CPU usage per node.
IO Blks Sec	number of logical data blocks read or written.

Column...	Reports the...
Num Procs	average number of tasks of online nodes.
Num Requests	number of AMP Worker Task messages or requests that got assigned AMP Worker Tasks to them.
Avg QWait Time	average time in milliseconds that work requests waited on an input queue before being serviced.
Max QWait Time	maximum time in milliseconds that work requests waited on an input queue before being serviced.
Q Length	average number of messages queued for output to the host.
Q Len Max	maximum number of messages queued for output to the host.
Avg Svc Time	average time in milliseconds that work requests required for service
Max Svc Time	maximum time in milliseconds that work requests required for service.

For a complete description of the above columns, see [ResUsageSps Table](#).

ResPs Examples

```
exec resPsByNode(,,,,);

exec resPsByGroup(,,);
```

ResVdskByNode Macros

Function

Macro...	Reports the logical device traffic by...
ResVdskByNode	a physical node.
ResVdskOneNode	for a specified node.
ResVdskByGroup	a node grouping.

Input Format Examples

The input forms of these three macros are described below.

```
EXEC ResVdskByNode
( FromDate,ToDate,FromTime,ToTime,FromNode,ToNode );
```

```
EXEC ResVdskOneNode
( FromDate,ToDate,FromTime,ToTime,Node );
```

```
EXEC ResVdskByGroup
( FromDate,ToDate,FromTime,ToTime );
```

See [Macro Execution](#) for a description of the *FromDate*, *FromTime*, *ToDate*, *ToTime*, *FromNode*, *ToNode* and *Node* parameters.

Usage Notes

To use the macros, existing rows must exist in the ResUsageSvdsk table.

ResVdskByNode Macros Output

The following table describes the statistics columns in all output reports (with the exception of ResVdskByNode, which has the NodeID column, and ResVdskByGroup, which has the NodeType column).

Column...	Reports the...
Read Cnt / Sec	average number of logical device reads per second.
Write Cnt / Sec	average number of logical device writes per second.
Rd KB / I/O	average number of KB per logical device read.
Wrt KB / I/O	average number of KB per logical device write.
Avg I/O Resp	average response time for a logical device read or write in seconds.
Out Rqst Time %	percent of time there are outstanding requests.
Max Concur Rqsts	maximum number of concurrent requests during the log period.

ResVdskByNode Examples

```
exec resVdskByNode(,,,,);

exec resVdskOneNode(,,,,);

exec resVdskByGroup(,,,);
```


Notation Conventions

How to Read Syntax

This document uses the following syntax conventions.

Syntax Convention	Meaning
KEYWORD	Keyword. Spell exactly as shown. Many environments are case-insensitive. Syntax shows keywords in uppercase unless operating system restrictions require them to be lowercase or mixed-case.
<i>variable</i>	Variable. Replace with actual value.
<i>number</i>	String of one or more digits. Do not use commas in numbers with more than three digits. Example: 10045
[x]	x is optional.
[x y]	You can specify x, y, or nothing.
{ x y }	You must specify either x or y.
x [...]	You can repeat x, separating occurrences with spaces. Example: x x x See note after table.
x [, ...]	You can repeat x, separating occurrences with commas. Example: x, x, x See note after table.
x [delimiter...]	You can repeat x, separating occurrences with specified delimiter. Examples: <ul style="list-style-type: none"> If <i>delimiter</i> is semicolon: x; x; x If <i>delimiter</i> is { , OR }, you can do either of the following: <ul style="list-style-type: none"> x, x, x x OR x OR x See note after table.

Note:

You can repeat only the immediately preceding item. For example, if the syntax is:

```
KEYWORD x [...]
```

You can repeat x. Do not repeat KEYWORD.

If there is no white space between x and the delimiter, the repeatable item is x and the delimiter. For example, if the syntax is:

```
[ x, [...] ] y
```

- You can omit x: y
- You can specify x once: x, y
- You can repeat x and the delimiter: x, x, x, y

Character Shorthand Notation Used in This Document

This document uses the Unicode naming convention for characters. For example, the lowercase character 'a' is more formally specified as either LATIN CAPITAL LETTER A or U+0041. The U+xxxx notation refers to a particular code point in the Unicode standard, where xxxx stands for the hexadecimal representation of the 16-bit value defined in the standard.

In parts of the document, it is convenient to use a symbol to represent a special character, or a particular class of characters. This is particularly true in discussion of the following Japanese character encodings:

- KanjiEBCDIC
- KanjiEUC
- KanjiShift-JIS

These encodings are further defined in *Teradata Vantage™ - Advanced SQL Engine International Character Set Support*, B035-1125.

Character Symbols

The symbols, along with character sets with which they are used, are defined in the following table.

Symbol	Encoding	Meaning
a-z A-Z 0-9	Any	Any single byte Latin letter or digit.
<u>a-z</u> <u>A-Z</u> <u>0-9</u>	Any	Any fullwidth Latin letter or digit.

Symbol	Encoding	Meaning
<	KanjiEBCDIC	Shift Out [SO] (0x0E). Indicates transition from single to multibyte character in KanjiEBCDIC.
>	KanjiEBCDIC	Shift In [SI] (0x0F). Indicates transition from multibyte to single byte KanjiEBCDIC.
T	Any	Any multibyte character. The encoding depends on the current character set. For KanjiEUC, code set 3 characters are always preceded by <code>ss3</code> .
!	Any	Any single byte Hankaku Katakana character. In KanjiEUC, it must be preceded by <code>ss2</code> , forming an individual multibyte character.
<u>Δ</u>	Any	Represents the graphic pad character.
Δ	Any	Represents a single or multibyte pad character, depending on context.
ss 2	KanjiEUC	Represents the EUC code set 2 introducer (0x8E).
ss 3	KanjiEUC	Represents the EUC code set 3 introducer (0x8F).

For example, string “TEST”, where each letter is intended to be a fullwidth character, is written as **TEST**. Occasionally, when encoding is important, hexadecimal representation is used.

For example, the following mixed single byte/multibyte character data in KanjiEBCDIC character set:

LMN<TEST>QRS

is represented as:

D3 D4 D5 0E 42E3 42C5 42E2 42E3 0F D8 D9 E2

Pad Characters

The following table lists the pad characters for the various character data types.

Server Character Set	Pad Character Name	Pad Character Value
LATIN	SPACE	0x20
UNICODE	SPACE	U+0020
GRAPHIC	IDEOGRAPHIC SPACE	U+3000
KANJISJIS	ASCII SPACE	0x20
KANJI1	ASCII SPACE	0x20

ResUsagelpma Table

The ResUsagelpma table includes resource usage data for system-wide, node information.

Note:

The ResUsagelpma table is generally not used at customer sites and is only used by Teradata engineers.

Teradata recommends that you use ReslpmaView to access the data, rather than accessing the ResUsagelpma table directly.

This table is created as a MULTiset table. For more information, see [Relational Primary Index](#).

Note:

Summary Mode is not applicable to this table.

Housekeeping Columns

Relational Primary Index Columns

These columns taken together form the nonunique primary index.

Column Name	Mode	Description	Data Type
TheDate	n/a	Date of the log entry.	DATE
TheTime	n/a	Nominal time of the log entry. Note: Under conditions of heavy system load, entries may be logged late (typically, by no more than one or two seconds), but this column will still contain the time value when the entry <i>should have been</i> logged. For more information, see the Secs and NominalSecs columns.	FLOAT
NodeID	n/a	Node ID on which the entry resides. The Node ID is formatted as ZZZ9-9999, where ZZZ9 denotes the four-digit cabinet number and 9999 denotes the four-digit chassis number of the node. For example, a node in chassis 9 of cabinet 3 has a node ID of '3-0009'. Note: SMP nodes have a chassis and cabinet number of 1. For example, the node ID of an SMP node is '1-0001'.	INTEGER

Miscellaneous Housekeeping Columns

These columns provide a generalized picture of the vprocs running on this node, shown as Type n virtual processors where $n = 1$ to 7. Under the current implementation, only Type 1 (AMP), Type 2 (PE), Type 3 (GTW), Type 4 (RSG), and Type 5 (TVS) vprocs exist; vproc types 6 through 7 are not currently used.

Column Name	Mode	Description	Data Type
GmtTime	n/a	Greenwich Mean Time is not affected by the Daylight Savings Time adjustments that occur twice a year.	FLOAT
CabinetID	n/a	The physical cabinet number of the node.	INTEGER
ModuleID	n/a	The physical module number of the node.	INTEGER
NodeType	n/a	Type of node, representing the per node system family type.	CHAR(8)
NCPUs	n/a	Number of online CPUs on this node. This column is useful for normalizing the CPU utilization column values for the number of CPUs on the node. This is especially important in: <ul style="list-style-type: none"> Coexistence systems where the number of CPUs can vary across system nodes. Elastic TCore systems where the number of online CPUs can change without database restart. 	SMALLINT
Vproc1	n/a	Current count of type 1 (AMP) virtual processors running under the node.	SMALLINT
Vproc2	n/a	Current count of type 2 (PE) virtual processors running under the node.	SMALLINT
Vproc3	n/a	Current count of type 3 (GTW) virtual processors running under the node.	SMALLINT
Vproc4	n/a	Current count of type 4 (RSG) virtual processors running under the node.	SMALLINT
Vproc5	n/a	Current count of type 5 (TVS) virtual processors running under the node.	SMALLINT
Vproc6	n/a	Current count of type 6 virtual processors running under the node. This column reports zeros.	SMALLINT
Vproc7	n/a	Current count of type 7 virtual processors running under the node. This column reports zeros.	SMALLINT
VprocType1	n/a	Type of virtual processor for Vproc1. Value is always AMP.	CHAR(4)
VprocType2	n/a	Type of virtual processor for Vproc2. Value is always PE.	CHAR(4)
VprocType3	n/a	Type of virtual processor for Vproc3. Value is always GTW.	CHAR(4)

Column Name	Mode	Description	Data Type
VprocType4	n/a	Type of virtual processor for Vproc4. Value is always RSG.	CHAR(4)
VprocType5	n/a	Type of virtual processor for Vproc5. The value is always TVS (see <i>Teradata Vantage™ - Teradata® Virtual Storage</i> , B035-1179).	CHAR(4)
VprocType6	n/a	The type of virtual processor for Vproc6.	CHAR(4)
VprocType7	n/a	The type of virtual processor for Vproc7.	CHAR(4)
MemSize	n/a	Amount of memory on this node in megabytes. Useful for performing memory usage calculations.	BIGINT
NodeNormFactor	n/a	<p>A per CPU normalization factor that is used to normalize the reported CPU values of the ResUsageSpma table to a single 5100 CPU.</p> <p>This value is scaled by a factor of 100. For example, if the actual factor is 5.25, the value of the NodeNormFactor will be 525.</p> <p>Note:</p> <p>The normalization factor is related to the NodeType value reported in the ResUsageSpma table. For information on this value, see ResUsageSpma Table.</p>	INTEGER
Secs	n/a	<p>Actual number of seconds in the log period represented by this row. Normally the same as NominalSecs, but can be different in three cases:</p> <ul style="list-style-type: none"> • The first interval after a log rate change • A sample logged late because of load on the system • System clock adjustments affect reported Secs <p>Useful for normalizing the count statistics contained in this row, for example, to a per-second measurement.</p>	SMALLINT
CentiSecs	n/a	Number of centiseconds in the logging period. This column is useful when performing data calculations with small elapsed times where the difference between centisecond-based data and whole seconds results in a percentage error.	INTEGER
NominalSecs	n/a	Specified or nominal number of seconds in the logging period.	SMALLINT
Active	max	<p>Controls whether or not the rows will be logged to the resource usage tables if Active Row Filter Mode is enabled.</p> <p>If Active is set to a non-zero value, the row contains data columns.</p> <p>If Active is set to a zero value, none of the data columns in the row have been updated during the logging period.</p> <p>For example, if you enable Active Row Filter Mode, the rows that have a zero Active column value will not be logged to the resource usage tables.</p>	FLOAT
TheTimestamp	n/a	Number of seconds since midnight, January 1, 1970.	BIGINT

Column Name	Mode	Description	Data Type
PM_COD_CPU	n/a	Platform Metering (PM) CPU Capacity On Demand (COD) value in one tenths of a percent. For example, a value of 500 represents a PM CPU COD value of 50.0%. The value is set to 1000 if the PM CPU COD is disabled.	SMALLINT
PM_COD_IO	n/a	Platform Metering (PM) I/O Capacity On Demand (COD) value in whole percent values for the entire system. For example, a value of 50 represents a PM I/O COD value of 50%. The value is set to 100 if the PM I/O COD is disabled.	SMALLINT
WM_COD_CPU	n/a	Workload Management (WM) CPU Capacity On Demand (COD) value in one tenths of a percent. For example, a value of 500 represents a WM CPU COD value of 50.0%. The value is set to 1000 if the WM CPU COD is disabled.	SMALLINT
WM_COD_IO	n/a	Workload Management (WM) I/O Capacity On Demand (COD) value in whole percent. For example, a value of 50 represents a WM I/O COD value of 50.0%. The value is set to 100 if the WM I/O COD is disabled.	SMALLINT
TIER_FACTOR	n/a	I/O performance limit placed on a core-reduced node. For example, a value of 75 represents an I/O limit of 75.0% placed before other COD values. The field is sourced from tpanodep->tierfactor, which is derived from the /proc/tdmeter/tier_performance file. A value of 100 indicates that there is no I/O performance limit applied to the node.	SMALLINT

Statistics Columns

Process Scheduling: Scheduled CPU Switching Column

Column Name	Mode	Description	Data Type
ProcGenClockInts	count	Number of timer interrupts.	FLOAT

MDL Segment Columns

Column Name	Mode	Description	Data Type
SegMDLAlloc[00-10]	count	Number of segments (between 64 KB for SegMDLAlloc00 and 64 MB for SegMDLAlloc10) allocated for the MDL pool during this period.	FLOAT

Column Name	Mode	Description	Data Type
SegMDLFree[00-10]	count	Number of dirty segments (between 64 KB for SegMDLFree00 and 64 MB for SegMDLFree10) freed from the MDL pool during this period.	FLOAT
SegMDLRelease[00-10]	count	Number of clean segments (between 64 KB for SegMDLRelease00 and 64 MB for SegMDLRelease10) freed from the MDL pool during this period.	FLOAT
SegMDLRecycle[00-10]	count	Number of segments (between 64 KB for SegMDLRecycle00 and 64 MB for SegMDLRecycle10) recycled to the MDL pool during this period.	FLOAT

Net: Message Type Columns

These columns subdivide all messages sent and received into the type of message, where:

- Hash messages (Hash) are data sent to a destination through its primary or fallback hash value
- Processor messages (Proc) are data sent to a destination through a vproc ID
- Group messages (Group) are broadcasted messages to be received by members of a group
- Local messages (Local) are messages communicated locally within the node
- Channel messages (Chan) are data sent between vprocs through channel IDs for purposes of a private conversation to perform functions such as row redistribution, and so on
- Mailbox messages (Mbox) are data sent between vprocs through mailbox IDs for similar purposes as channel messages.

A duplicated accounting is done with two different perspectives, since Hash + Proc + Group + Local messages = Chan + MBox messages.

Column Name	Mode	Description	Data Type
MsgChanReads	count	Number of channel messages read by this node.	FLOAT
MsgChanWrites	count	Number of channel messages written by this node.	FLOAT
MsgHashReads	count	Number of hash messages read by this node.	FLOAT
MsgHashWrites	count	Number of hash messages written by this node.	FLOAT
MsgGroupReads	count	Number of group messages read by this node.	FLOAT
MsgGroupWrites	count	Number of group messages written by this node.	FLOAT
MsgLocalReads	count	Number of local messages read by this node.	FLOAT
MsgLocalWrites	count	Number of local messages written by this node.	FLOAT
MsgMboxReads	count	Number of mailbox messages read by this node.	FLOAT
MsgMboxWrites	count	Number of mailbox messages written by this node.	FLOAT

Column Name	Mode	Description	Data Type
MsgProcReads	count	Number of processor messages read by this node.	FLOAT
MsgProcWrites	count	Number of processor messages written by the node.	FLOAT

Net: Message Delivery Times Columns

These columns identify the time it took for hash, processor, group and local messages to reach their destination. Two times are provided:

- Message transmission to mailbox delivery (MDelivery)
- Mailbox delivery to process delivery (PDelivery)

Column Name	Mode	Description	Data Type
MsgHashMDelivery	count	Total amount of time read hash messages took for mailbox delivery. Note: This column is not currently valid. It should not be used.	FLOAT
MsgHashPDelivery	count	Total amount of time read hash messages took for process delivery. Note: This column is not currently valid. It should not be used.	FLOAT
MsgGroupMDelivery	count	Total amount of time read group messages took for mailbox delivery. Note: This column is not currently valid. It should not be used.	FLOAT
MsgGroupPDelivery	count	Total amount of time read group messages took for process delivery. Note: This column is not currently valid. It should not be used.	FLOAT
MsgLocalMDelivery	count	Total amount of time read local messages took for mailbox delivery. Note: This column is not currently valid. It should not be used.	FLOAT
MsgLocalPDelivery	count	Total amount of time read local messages took for process delivery. Note: This column is not currently valid. It should not be used.	FLOAT

Column Name	Mode	Description	Data Type
MsgProcMDelivery	count	Total amount of time read processor messages took for mailbox delivery. Note: This column is not currently valid. It should not be used.	FLOAT
MsgProcPDelivery	count	Total amount of time read processor messages took for process delivery. Note: This column is not currently valid. It should not be used.	FLOAT

Net: Net Circuit Management Columns

For names and descriptions of these columns, see [ResUsageSpma Table](#).

Net: Per Bynet Network Transport Data Columns

For the names and descriptions of these columns, see [ResUsageSpma Table](#).

Net: Net Miscellaneous Contention Management Columns

These columns identify some additional contention management not addressed in the other contention management areas.

Column Name	Mode	Description	Data Type
NetBrdWindowOverrun	count	Broadcast window overruns on all Bynets. Note: This column is net-specific, that is, it relates to each specific Bynet. On a single-node system, net-specific statistics are not meaningful and are always zero	FLOAT
NetMrgBufWaits	count	Number of times an IOP task encountered an empty row-block buffer on all Bynets.	FLOAT
NetMrgHeapFails	count	Number of times a merge operation heap space request failed.	FLOAT
NetMrgHeapRequests	count	Number of times a merge operation requested heap space.	FLOAT
NetMsgFCSleep	count	Number of times a transmitter process was put to sleep because it was flow controlled.	FLOAT

Net: Net Queues Columns

These columns identify lengths of the various internal queues used by the network controllers.

- NetSamples can be used to normalize all aggregated sampled statistics to an average queue-length basis.
- Example: Dividing (NetPtPQueue/NetSamples) yields the average point-to-point queue length over all samples on all networks taken during this log interval.
- All of the aggregated sampled statistics columns in the following table are net-specific, that is, they relate to each specific Bynet. On a single-node system, net-specific statistics are not meaningful and are always zero.

Column Name	Mode	Description	Data Type
NetPtPQueue	count	Aggregated sample point-to-point normal priority queue length on all Bynets.	FLOAT
NetBrdQueue	count	Aggregated sample broadcast normal priority queue length on all Bynets.	FLOAT
NetPtPQueueMax	max	Maximum reported value of NetPtPQueue in this reporting interval.	FLOAT
NetBrdQueueMax	max	Maximum reported value of NetBrdQueue in this reporting interval.	FLOAT
NetHPBrdQueue	count	Aggregated sample broadcast high priority queue length on all networks.	FLOAT
NetHPBrdQueueMax	max	Maximum value of NetHPBrdQueue in this reporting interval.	FLOAT
NetPendMrgQueue	count	Current count of pending merges, regardless of which net. A merge may be queued for reasons such as: <ul style="list-style-type: none"> • The local IOP memory is saturated • System memory is trashing. 	FLOAT
NetHPPtPQueue	count	Aggregated sample point-to-point high priority queue length on all Bynets.	FLOAT
NetHPPtPQueueMax	max	Maximum value of NetHPPtPQueue in this reporting interval.	FLOAT

RSS Internal Diagnostics Columns

These columns are used internally for RSS diagnostics.

Column Name	Mode	Description	Data Type
ResdTodStartSec	track	RSS internal data.	FLOAT

Column Name	Mode	Description	Data Type
ResdTodStartUsec	track	RSS internal data.	FLOAT
ResdTodSleepSec	track	RSS internal data.	FLOAT
ResdTodSleepUsec	track	RSS internal data.	FLOAT
ResdTodWakeupSec	track	RSS internal data.	FLOAT
ResdTodWakeupUsec	track	RSS internal data.	FLOAT
ResdReqNum	track	RSS internal data.	FLOAT
ResdWaitTimeMsecs	track	RSS internal data.	FLOAT
GatherTodCurrSec	track	RSS internal data.	FLOAT
GatherTodCurrUsec	track	RSS internal data.	FLOAT
RSSInternal[00-06]	track	RSS internal data.	FLOAT
RSSInternal[07-08, 10-11, 13-14, 16-19]	count	RSS internal data.	FLOAT
RSSInternal[09, 12, 15]	max	RSS internal data.	FLOAT

Reserved Columns

Column Name	Mode	Description	Data Type
ReservedS0	n/a	Reserved for future use.	CHAR(4)
ReservedS1	n/a	Reserved for future use.	CHAR(8)
Reserved00	n/a	Reserved for future use.	SMALLINT

Spare Columns

The ResUsagelpma table spare fields are named Spare00 through Spare09, and SpareInt.

The SpareInt field has a 32-bit internal resolution while all other spare fields have a 64-bit internal resolution. All spare fields default to count but can be converted to min, max, or track mode fields if needed when they are used.

Column Name	Mode	Description	Data Type
Spare00 - Spare09, SpareInt	count	Reserved for future use.	FLOAT

Related Information

For details on the different type of data fields, see [About the Mode Column](#).

ResUsagelvpr Table

The ResUsagelvpr table includes resource usage data for system-wide information.

Teradata recommends that you use ReslvprView to access the data, rather than accessing the ResUsagelvpr table directly

Note:

The ResUsagelvpr table is generally not used at customer sites and is only used by Teradata engineers.

Housekeeping Columns

Relational Primary Index Columns

These columns taken together form the primary index.

Column Name	Mode	Description	Data Type
TheDate	n/a	Date of the log entry.	DATE
TheTime	n/a	Nominal time of the log entry. Note: Under conditions of heavy system load, entries may be logged late (typically, by no more than one or two seconds), but this column will still contain the time value when the entry <i>should have been</i> logged. For more information, see the Secs and NominalSecs columns.	FLOAT
NodeID	n/a	Node ID on which the entry resides. The Node ID is formatted as ZZZ9-9999, where ZZZ9 denotes the four-digit cabinet number and 9999 denotes the four-digit chassis number of the node. For example, a node in chassis 9 of cabinet 3 has a node ID of '3-0009'. Note: SMP nodes have a chassis and cabinet number of 1. For example, the node ID of an SMP node is '1-0001'.	INTEGER

Miscellaneous Housekeeping Columns

Column Name	Mode	Description	Data Type
GmtTime	n/a	Greenwich Mean Time is not affected by the Daylight Saving Time adjustments that occur twice a year.	FLOAT

Column Name	Mode	Description	Data Type
CabinetID	n/a	The physical cabinet number of the node.	INTEGER
ModuleID	n/a	The physical module number of the node.	INTEGER
NodeType	n/a	Type of node, representing the per node system family type.	CHAR(8)
VprId	n/a	Identifies the vproc number (non-Summary Mode) or the vproc type (Summary Mode; 0 = NODE, 1 = AMP, 2 = PE, 3=GTW, 4=RSG, 5=TVS).	INTEGER
VprType	n/a	The values can be NODE, AMP, PE, GTW, RSG, or TVS (see <i>Teradata Vantage™ - Teradata® Virtual Storage</i> , B035-1179).	CHAR(4)
Secs	n/a	Actual number of seconds in the log period represented by this row. Normally the same as NominalSecs, but can be different in three cases: <ul style="list-style-type: none"> • The first interval after a log rate change • A sample logged late because of load on the system • System clock adjustments affect reported Secs Useful for normalizing the count statistics contained in this row, for example, to a per-second measurement.	SMALLINT
CentiSecs	n/a	Number of centiseconds in the logging period. This column is useful when performing data calculations with small elapsed times where the difference between centisecond-based data and whole seconds results in a percentage error.	INTEGER
NominalSecs	n/a	Specified or nominal number of seconds in the logging period.	SMALLINT
SummaryFlag	n/a	Summarization status of this row. Possible values are 'N' if the row is a non-summary row, and 'S' if the row is a summary row.	CHAR (1)
NCPUs	n/a	Number of online CPUs on this node. This column is useful for normalizing the CPU utilization column values for the number of CPUs on the node. This is especially important in: <ul style="list-style-type: none"> • Coexistence systems where the number of CPUs can vary across system nodes. • Elastic TCore systems where the number of online CPUs can change without database restart. 	SMALLINT
Active	max	Controls whether or not the rows will be logged to the resource usage tables if Active Row Filter Mode is enabled. If Active is set to a non-zero value, the row contains data columns. If Active is set to a zero value, none of the data columns in the row have been updated during the logging period. For example, if you enable Active Row Filter Mode, the rows that have a zero Active column value will not be logged to the resource usage tables.	FLOAT
TheTimestamp	n/a	Number of seconds since midnight, January 1, 1970. This column is useful for aligning data with the DBQL log.	BIGINT

Column Name	Mode	Description	Data Type
PM_COD_CPU	n/a	Platform Metering (PM) CPU Capacity On Demand (COD) value in one tenths of a percent. For example, a value of 500 represents a PM CPU COD value of 50.0%. The value is set to 1000 if the PM CPU COD is disabled.	SMALLINT
WM_COD_CPU	n/a	Workload Management (WM) CPU Capacity On Demand (COD) value in one tenths of a percent. For example, a value of 500 represents a WM CPU COD value of 50.0%. The value is set to 1000 if the WM CPU COD is disabled.	SMALLINT

Statistics Columns

File System: Cylinder Defragmentation Overhead Columns

These columns identify background file system overhead associated with fragmented free space to achieve one large free space within that cylinder (CylDefrag).

- Event counts are found in ResUsageSvpr.
- Each cylinder defragment event implies one logical cylinder index read and one logical cylinder index write.
- Only logical I/Os and the amount moved (KB) are identified. Cylinder defragments are done on cylinders containing permanent tables (including append and transient journal tables) only.

Column Name	Mode	Description	Data Type
FileDbCylDefragKB	count	KB moved by the FileDbCylDefragIO.	FLOAT
FileDbCylDefragIO	count	Number of permanent data block logical I/Os due to cylinder defragmentation.	FLOAT

File System: Cylinder MiniCylPack Overhead Columns

These columns identify file system overhead associated with MiniCylPack operations that get performed to make available a free cylinder when one is needed but not available.

- Event counts are found in ResUsageSvpr.
- Only logical I/Os and the amount moved (KB) are identified, except the amount moved for cylinder indexes because they can be calculated by multiplying the current cylinder index fixed size and the I/Os.
- MiniCylPack operations are done on cylinders containing permanent tables (including append and transient journal tables) only.

Column Name	Mode	Description	Data Type
FilePCiMCylPackIO	count	Number of permanent cylinder index logical I/Os due to performing MiniCylPack operations.	FLOAT
FilePDbMCylPackKB	count	KB moved by FilePDbCylPackIOs.	FLOAT
FilePDbMCylPackIO	count	Number of permanent data block logical I/Os due to performing MiniCylPack operations.	FLOAT

File System: Cylinder Split and Migrate Overhead Columns

These columns further identify file system cylinder split/migrate (CylMigr) overhead performed when cylinders can not accommodate new data.

Note:

Event counts are found in [ResUsageSvpr Table](#).

Only logical I/Os and the amount moved (KB) for data blocks are identified. Each cylinder migration event implies one logical read and three logical writes of the cylinder index. Only permanent tables (including append and transient journal tables) are migrated.

Column Name	Mode	Description	Data Type
FileDbCylMigrKB	count	KB moved by FileDbCylMigrIOs.	FLOAT
FileDbCylMigrIO	count	Number of data block logical I/Os due to cylinder migration.	FLOAT

File System: Data Block Creation Columns

These columns identify the file system operations required when a data block is being created (BlkCreate).

Note:

These columns do not include data blocks created due to any of the new data blocks created when a data block was updated as described in the Data Block Update Operations Columns description.

Column Name	Mode	Description	Data Type
FilePDbCreateKB	count	KB created by FilePDbCreates.	FLOAT
FilePDbCreates	count	Number of permanent table (including append and transient journal tables) data blocks created.	FLOAT
FileSDBCreateKB	count	KB created by FileSDBCreates.	FLOAT
FileSDBCreates	count	Number of spool data blocks created.	FLOAT

Column Name	Mode	Description	Data Type
FileDbUpdNoWAL	count	Number of times a data block was updated without WAL (written to disk without appending DB-specific WAL records).	FLOAT

File System: Data Block Update Operations Columns

These columns identify the file system operations required when a data block is being updated (BlkUpd). When a block is updated, it can be 'in place' and requires no new data blocks, or it could spill over the current data block and require one, two, three or more new data blocks in addition to the current data block. Only logical I/Os and the amount moved (KB) are identified, except for the amount moved for cylinder indexes because they can be calculated by multiplying the current fixed cylinder index size by the I/Os. Data block updates should only be performed on permanent tables (including append and transient journal tables), so no attempt is made to separate permanent and spool data segments.

Column Name	Mode	Description	Data Type
FileCiUpd0IO	count	Number of cylinder index logical I/Os performed for a block update operation requiring no new data blocks.	FLOAT
FileCiUpd1IO	count	Number of cylinder index logical I/Os performed for a block update operation requiring 1 new data blocks.	FLOAT
FileCiUpd2IO	count	Number of cylinder index logical I/Os performed for a block update operation requiring two new data blocks.	FLOAT
FileCiUpd3IO	count	Number of cylinder index logical I/Os performed for a block update operation requiring three new data blocks.	FLOAT
FileCiUpdNIO	count	Number of cylinder index logical I/Os performed for a block update operation requiring over three new data blocks.	FLOAT
FileDbUpd0IO	count	Number of data block logical I/Os performed for a block update operation requiring no new data blocks.	FLOAT
FileDbUpd1IO	count	Number of data block logical I/Os performed for a block update operation requiring one new data blocks.	FLOAT
FileDbUpd2IO	count	Number of data block logical I/Os performed for a block update operation requiring two new data blocks.	FLOAT
FileDbUpd3IO	count	Number of data block logical I/Os performed for a block update operation requiring three new data blocks.	FLOAT
FileDbUpd0KB	count	KB moved by FileDbUpd0IO.	FLOAT
FileDbUpd1KB	count	KB moved by FileDbUpd1IO.	FLOAT
FileDbUpd2KB	count	KB moved by FileDbUpd2IO.	FLOAT
FileDbUpd3KB	count	KB moved by FileDbUpd3IO.	FLOAT
FileDbUpdNKB	count	KB moved by FileDbUpdNIO.	FLOAT

Column Name	Mode	Description	Data Type
FileDbUpdNIO	count	Number of data block logical I/Os performed for a block update operation requiring over three new data blocks.	FLOAT

File System: Multi-Row Requests Columns

These columns identify the significant multi-row requests made by application software on the file system. Rows are distinguished as permanent data (P), spool (S) or user append table / permanent journal table (APt).

Column Name	Mode	Description	Data Type
FileAPtBlkRead	count	Number of requests for an append data block read.	FLOAT
FileAPtBlkReplace	count	Number of requests for an append data block replace.	FLOAT
FileAPtRownins	count	Number of requests for an append data multi-row insert.	FLOAT
FileAPtRowNDel	count	Number of requests for an append data multi-row delete.	FLOAT
FileAPtRowNUpd	count	Number of requests for an append data multi-row update.	FLOAT
FileAPtSorttable	count	Number of requests for an append table sort.	FLOAT
FileAPtTabdelete	count	Number of requests for an append table delete.	FLOAT
FileAPtTabdelra	count	Number of requests for an append multi-row delete.	FLOAT
FileAPtTabmrows	count	Number of requests for an append table modification.	FLOAT
FileAPtTabrblocks	count	Number of requests for an append table multi-block read.	FLOAT
FilePBlkRead	count	Number of requests for a permanent data block read.	FLOAT
FilePBlkReplace	count	Number of requests for a permanent data block replace.	FLOAT
FilePRowNDel	count	Number of requests for a permanent data multi-row delete.	FLOAT
FilePRownins	count	Number of requests for a permanent data multi-row insert.	FLOAT
FilePRowNUpd	count	Number of requests for a permanent data multi-row update.	FLOAT
FilePSorttable	count	Number of requests for permanent table sort.	FLOAT
FilePTabdelete	count	Number of requests for a permanent table delete.	FLOAT
FilePTabdelra	count	Number of requests for a multi-row delete.	FLOAT
FilePTabmrows	count	Number of requests for a permanent table modification.	FLOAT
FilePTabrblocks	count	Number of requests for a permanent table multi-block read.	FLOAT
FileSBlkRead	count	Number of requests for a spool data block read.	FLOAT

Column Name	Mode	Description	Data Type
FileSBlkReplace	count	Number of requests for a spool data block replace.	FLOAT
FileSRowNDel	count	Number of requests for a spool data multi-row delete.	FLOAT
FileSRownins	count	Number of requests for a spool data multi-row insert.	FLOAT
FileSRowNUpd	count	Number of requests for a spool data multi-row update.	FLOAT
FileSSortable	count	Number of requests for spool table sort.	FLOAT
FileSTabdelete	count	Number of requests for a spool table delete.	FLOAT
FileSTabdelra	count	Number of requests for a spool multi-row delete.	FLOAT
FileSTabmrows	count	Number of requests for a spool table modification.	FLOAT
FileSTabrblocks	count	Number of requests for a spool table multi-block read.	FLOAT

File System: Single-Row Requests Columns

These columns identify the significant single-row requests made by application software on the file system. Rows are distinguished as permanent data (P), spool (S) or user append table / permanent journal table (APt).

Column Name	Mode	Description	Data Type
FileAPtRowAppend	count	Number of requests for an append row append.	FLOAT
FileAPtRowDelete	count	Number of requests for an append row delete.	FLOAT
FileAPtRowInsert	count	Number of requests for an append row insert.	FLOAT
FileAPtRowReadInit	count	Number of requests for an initial append row read.	FLOAT
FileAPtRowReadCont	count	Number of requests for an continued append row read.	FLOAT
FileAPtRowReplace	count	Number of requests for an append row replace/update.	FLOAT
FilePRowAppend	count	Number of requests for a row append.	FLOAT
FilePRowDelete	count	Number of requests for a permanent row delete.	FLOAT
FilePRowInsert	count	Number of requests for a permanent row insert.	FLOAT
FilePRowReadCont	count	Number of requests for a continued permanent row read.	FLOAT
FilePRowReadInit	count	Number of requests for an initial permanent row read.	FLOAT
FilePRowReplace	count	Number of requests for a permanent row replace.	FLOAT
FileSRowAppend	count	Number of requests for a row append.	FLOAT
FileSRowDelete	count	Number of requests for a spool row delete.	FLOAT

Column Name	Mode	Description	Data Type
FileSRowInsert	count	Number of requests for a spool row insert.	FLOAT
FileSRowReadCont	count	Number of requests for a continued spool row read.	FLOAT
FileSRowReadInit	count	Number of requests for an initial spool row read.	FLOAT
FileSRowReplace	count	Number of requests for a spool row replace/update.	FLOAT

File System: Transient Journal Overhead Columns

These columns identify file system overhead associated with maintaining a transient journal (TJ).

Column Name	Mode	Description	Data Type
FileTJBufUpdates	count	Number of transient journal buffer updates.	FLOAT

File System: Transient Journal Requests Column

These columns identify the significant transient journal requests made by application software on the file system.

Column Name	Mode	Description	Data Type
FileTJCalls	count	Number of transient journal calls.	FLOAT
FileTJDbUpdates	count	Number of WAL data blocks modified. The modification can either be an update or a delete of an existing WAL or TJ record.	FLOAT

File System: Miscellaneous Write Ahead Logging Columns

These columns identify the log-based file system recovery scheme in which modifications to permanent data are written to a log file, the WAL log.

Column Name	Mode	Description	Data Type
FileTJAppends	count	Number of times transient journal records were appended to the WAL log. A single append call can append multiple transient journal rows. A transient journal append by itself does not imply a write of a WAL block, nor a WAL Cylinder Index (WCI) modification.	FLOAT
FileTJFlush	count	Number of times a request to force transient journal records within the WAL log to be written to disk is issued. An increment of this counter may or may not result in an I/O depending on whether the request was to flush records that were already on disk.	FLOAT

Column Name	Mode	Description	Data Type
FileWAppends	count	Number of times a record was appended to the WAL log. A single append call can append multiple rows. Subtracting FileTJAppends from this counter results in the number of times non-transient journal rows were appended to the WAL log.	FLOAT
FileWDBCreates	count	Number of WAL data blocks created. The block can contain either TJ records, WAL records or both	FLOAT
FileWFlush	count	Number of times a request to force any record in the WAL log to be written to disk is issued. An increment of this counter may or may not result in an I/O depending on whether the request was to flush records that were already on disk. Subtracting FileTJFlush from this counter results in the number of times a non-transient journal WAL flush was issued.	FLOAT
FileWRowDelete	count	Number of times rows were deleted from the WAL log.	FLOAT
FileWTabDelRa	count	Number of requests for a WAL multi-row delete.	FLOAT

General Concurrency Control Monitor Management Columns

These columns identify monitor activities for concurrency control.

Column Name	Mode	Description	Data Type
MonAllocates	count	Number of monitors allocated.	FLOAT
MonBlocks	count	Number of times entry into a monitor was blocked. For example, the number of requests - the number of blocks = immediate grants.	FLOAT
MonEnters	count	Number of times entry into a monitor was requested.	FLOAT
MonYields	count	Number of times a monitor yield was requested.	FLOAT

Net: Message Type Columns

These columns subdivide all messages sent and received into the type of message, where:

- Hash messages (Hash) are data sent to a destination through its primary or fallback hash value.
- Processor messages (Proc) are data sent to a destination through a vproc ID.
- Group messages (Group) are broadcasted messages to be received by members of a group.
- Local messages (Local) are messages communicated locally within the node.
- Channel messages (Chan) are data sent between vprocs through channel IDs for purposes of a private conversation to perform functions such as row redistribution, and so on.
- Mailbox messages (Mbox) are data sent between vprocs through mailbox IDs for similar purposes as channel messages.

A duplicated accounting is done with two different perspectives, since Hash + Proc + Group + Local messages = Chan + MBox messages.

Column Name	Mode	Description	Data Type
MsgChanReads	count	Number of channel messages read by this vproc.	FLOAT
MsgChanWrites	count	Number of channel messages written by this vproc.	FLOAT
MsgHashReads	count	Number of hash messages read by this vproc.	FLOAT
MsgHashWrites	count	Number of hash messages written by this vproc.	FLOAT
MsgGroupReads	count	Number of group messages read by this vproc.	FLOAT
MsgGroupWrites	count	Number of group messages written by this vproc.	FLOAT
MsgLocalReads	count	Number of local messages read by this vproc.	FLOAT
MsgLocalWrites	count	Number of local messages written by this vproc.	FLOAT
MsgProcReads	count	Number of processor messages read by this vproc.	FLOAT
MsgProcWrites	count	Number of processor messages written by this vproc.	FLOAT
MsgMboxReads	count	Number of mailbox messages read by this vproc.	FLOAT
MsgMboxWrites	count	Number of mailbox messages written by this vproc.	FLOAT

Net: Message Delivery Times Columns

These columns identify the time it took for hash, processor, group and local messages to reach their destination. Two times are provided:

- Message transmission to mailbox delivery (MDelivery).
- Mailbox delivery to process delivery (PDelivery).

Column Name	Mode	Description	Data Type
MsgHashMDelivery	count	Total amount of time read hash messages took for mailbox delivery. Note: This column is not currently valid. It should not be used.	FLOAT
MsgHashPDelivery	count	Total amount of time read hash messages took for process delivery. Note: This column is not currently valid. It should not be used.	FLOAT
MsgGroupMDelivery	count	Total amount of time read group messages took for mailbox delivery.	FLOAT

Column Name	Mode	Description	Data Type
		Note: This column is not currently valid. It should not be used.	
MsgGroupPDelivery	count	Total amount of time read group messages took for process delivery. Note: This column is not currently valid. It should not be used.	FLOAT
MsgLocalMDelivery	count	Total amount of time read local messages took for mailbox delivery. Note: This column is not currently valid. It should not be used.	FLOAT
MsgLocalPDelivery	count	Total amount of time read local messages took for process delivery. Note: This column is not currently valid. It should not be used.	FLOAT
MsgProcMDelivery	count	Total amount of time read processor messages took for mailbox delivery. Note: This column is not currently valid. It should not be used.	FLOAT
MsgProcPDelivery	count	Total amount of time read processor messages took for process delivery. Note: This column is not currently valid. It should not be used.	FLOAT

Reserved Columns

Column Name	Mode	Description	Data Type
ReservedS0	n/a	Reserved for future use.	CHAR(3)
ReservedS1	n/a	Reserved for future use.	CHAR(4)
Reserved00	n/a	Reserved for future use.	SMALLINT

Spare Columns

The ResUsagelvpr table spare fields are named Spare00 through Spare09, and SpareInt.

The SpareInt field has a 32-bit internal resolution while all other spare fields have a 64-bit internal resolution. All spare fields default to count but can be converted to min, max, or track mode fields if needed when they are used.

Column Name	Mode	Description	Data Type
Spare00 - Spare09, SpareInt	count	Reserved for future use.	FLOAT

Related Information

For details on the different type of data fields, see [About the Mode Column](#).

Summary Mode

When Summary Mode is active for the ResUsagelvpr table, one row is written to the database for each type of vproc on each node in the system, summarizing the vprocs of that type on that node, for each log interval.

You can determine if a row is in Summary Mode by checking the SummaryFlag column for that row.

IF the SummaryFlag column value is...	THEN the data for that row is being logged...
'S'	in Summary Mode.
'N'	normally.

Partition Assignments

With regards to the database, there is more than one definition of partition. The partitions here refer to the following Parallel Database Extensions (PDE) and vproc definition:

- A partition is a collection of tasks and associated resources grouped within a virtual processor according to the function of the tasks. There are multiple partitions within a single virtual processor. Partitions are the primary mechanism used by the database to manage parallel programs.
- Partitions are the subdivision of vproc software processes into 48 semi-isolated domains.

For example, in an AMP vproc, Partition 11 is the AMP Worker Task partition. In all other vproc types, Partition 11 is unused.

Another partition description is only meaningful in a dialog between client programs and the database. It has nothing to do with PDE vproc partitions, but is a way of enforcing rules about what a client session is allowed to do and of keeping client sessions isolated from each other. This concept of partitions is centered in the CLIV2 interface, specifically the CONNECT parcel.

Partition reservation is as follows:

- Partitions 0 through 6 are reserved by PDE
- Partitions 7 through 47 are for use by the database

The table listed under [Partition Assignment Listing](#) describes the individual partitions. The database uses the following vprocs:

Vproc Type	Description
AMP	Access module processors perform database functions, such as executing database queries. Each AMP owns a portion of the overall database storage.
GTW	Gateway vprocs provide a socket interface to the database.
Node	The node vproc handles PDE and operating system functions not directly related to AMP and PE work. Node vprocs cannot be externally manipulated, and do not appear in the output of the Vproc Manager utility.
PE	Parsing engines perform session control, query parsing, security validation, query optimization, and query dispatch.
RSG	Relay Services Gateway provides a socket interface for communication with the Teradata Data Stream Utility.
TVS	Manages database storage. AMPs acquire their portions of database storage through the TVS vproc.

For more information on partition usage, see [Process Scheduling: CPU Utilization Columns](#).

Table Conventions

The following table describes the table symbols used in the partition assignments table below.

The symbol used in the Partition Assignment Listing...	Indicates...
—	partition is unused.
?	activity is observed but not identified.

Partition Assignment Listing

The following table lists the Node, AMP, PE, GTW, RSG, and TVS usage of PDE vproc partitions by PDE and the database.

Partition:		Usage in Vprocs of Type					
#	Name	Node	AMP	PE	GTW	RSG	TVS
0	Kernel	PDE daemons	pdevproc				
1	System Debugger	System Debugger tasks					
3	Console	PDE console control process (cnscim)	-----				
3-6	Interactive 1 through 4	Console interactive partition programs					
7	Service	Console utilities	scpstart	-----			
8	Host Utility Procedures	-----	utadvtsk	-----			
9	Filesys	-----	File System processes	-----			
10	Gateway	-----			Gateway processes	-----	
11	AWT	-----	AMP Worker Tasks	-----			
12	Session	-----		Session Control tasks	-----		
13	Dispatch	-----		Dispatcher/ Parser Partition	-----		

Partition:		Usage in Vprocs of Type					
#	Name	Node	AMP	PE	GTW	RSG	TVS
14	[Unused]	-----					
15	Startup	-----	Startup tasks	-----			
16	[Unused]	-----					
17	RSS Startup	-----	File System RSS startup	-----			
18	Distributed Database File (DDF) Server	DDF services	-----				
20-29	Interactive Partitions	-----	DBCCONS utilities				
30	[Unused]	-----					
31	Allocator	-----					tvsaallocator
32	Node Agent	-----					tvsa_agent
33	Clique Coordinator	-----					Clique Coordinator services
34-39	[Unused]	-----					
40	UDF Debugger	server	-----				
41	UDF Debugger	controller	-----				
42	GDO Monitor	gdom	-----				
43	Parallel Data Collector	pdcmaster	-----				
44	Dump save	csp	-----				
45	Dump save or clear	csp	-----				
46	Dump list	csp	-----				
47	Replication	-----				Replication Gateway rsgdbsmain	

Additional Information

Teradata Links

Link	Description
https://docs.teradata.com/	Search Teradata Documentation, customize content to your needs, and download PDFs. Customers: Log in to access Orange Books.
https://support.teradata.com	One-stop source for Teradata community support, software downloads, and product information. Log in for customer access to: <ul style="list-style-type: none">• Community support• Software updates• Knowledge articles
https://www.teradata.com/University/Overview	Teradata education network
https://support.teradata.com/community	Link to Teradata community