Teradata® DSA

User Guide

Release 17.20

November 2022

B035-3150-101K
DOCS.TERADATA.COM

Copyright and Trademarks

Copyright © 2013 - 2023 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
NOTICE	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.
CAUTION	Indicates a hazardous situation which, if not avoided, could result in minor or moderate personal injury.
WARNING	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 laws, all designs, specifications, statements, information, recommendations and content (collectively, "content") available from the Teradata Documentation website or contained in Teradata information products is presented "as is" and without any express or implied warranties, including, but not limited to, the implied warranties of merchantability, fitness for a particular purpose, or noninfringement, which are hereby disclaimed. In no event shall Teradata corporation, its suppliers or partners be liable for any direct, indirect, incidental, special, exemplary, or consequential damages (including, but not limited to, procurement of substitute goods or services; loss of use, data, or profits; or business interruption) however caused and on any theory of liability, whether in contract, strict liability, or tort (including negligence or otherwise) arising in any way out of the use of content, even if advised of the possibility of such damage.

The Content 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 Content 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 the Content at any time without notice.

The Content is subject to change without notice. Users are solely responsible for their application of the Content. The Content does not constitute the technical or other professional advice of Teradata, its suppliers or partners. Users should consult their own technical advisors before implementing any Content. Results may vary depending on factors not tested by Teradata.

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.

Confidential Information

Confidential Information means any and all confidential knowledge, data or information of Teradata, including, but not limited to, copyrights, patent rights, trade secret rights, trademark rights and all other intellectual property rights of any sort.

The Content available from the Teradata Documentation website or contained in Teradata information products may include Confidential Information and as such, the use of such Content is subject to the non-use and confidentiality obligations and protections of a non-disclosure agreement or other such agreements to protect Confidential Information that you have executed with Teradata.

Contents

Chapter 1: Data Stream Architecture 8
Welcome to Teradata DSA User Guide 8
Introduction to Data Stream Architecture 8
BAR Integration 10
BAR Job Workflow 12
BAR Portlets and Command-Line Interface 13

Chapter 2: DSA Features and Operations 15
New Features in this Release 15
DSA Permissions 16
DSA Restrictions 17
Azure: Use Multiple Storage Accounts for Single Backup 17
Multiple DSCs 18
DSA and MAPS Architecture 18
Component Deletion 18
Copy and Restore Definitions 18
Copy Restrictions 19
Copy Operations for Objects and Data Table Archives 19
HUT Locks in DSA 20
Active and Retired Jobs 22
Multiple DSA Domains 23
Public Cloud: Default Target Group for Elasticity Scale Out/In 23
Define Data Paths Using Network Fabric 25
Job Status Enhancements 26
Advanced Job Status Enhancements 27
Restart a Restore Job 28
Hot Standby Node Stream Distribution Enhancements 29
Enhanced Locking Management 29

Chapter 3: DSA CBB Backup and Restore 31
Enhanced Change Block Backup 32
Example Backup Strategy 32
Incomplete Backups 33

Chapter 4: Using Incremental Restore to Maintain Your DR System 35
Incremental Restore Limitations 35
Changes That Force Full Backup of IR-Enabled Backup Job 36
Incremental Restore with DBC 37
Changes That Force Full Restore of IR-Enabled Restore Job 37

Incremental Restore Examples . . . 39
Incremental Restore Workflow . . . 40
Verifying Incremental Restore is Enabled: BAR Setup Portlet . . . 42
Verifying Incremental Restore is Enabled: Command Line . . . 42
Verifying Incremental Restore for DBC is Enabled: Command Line . . . 42
Creating and Running an Incremental Restore Enabled Backup Job: BAR Operations Portlet . . . 42
Creating an Incremental Restore Job: BAR Operations Portlet . . . 44
Creating or Updating a Job: Command Line . . . 45
Creating an Incremental Restore with DBC Job . . . 46
New or Updated Command Line Commands for Incremental Restore . . . 46
BAR Setup Portlet: IR Support . . . 46

Chapter 5: Teradata BAR Portlets . . . 49
BAR Setup . . . 49
BAR Operations . . . 79

Chapter 6: DSA Command-Line Interface . . . 114
Accessing DSA Command Help . . . 114
DSA Command Types . . . 115
DSA Configuration: Command Line . . . 120
Planning a Job: Command Line . . . 131

Chapter 7: Troubleshooting . . . 144
Log Files . . . 144
TVI Logging . . . 145
BARNC (DSA Network Client) Error Codes . . . 147
Locked Object Restrictions . . . 148

Chapter 8: Database DBC Info . . . 150
Database DBC . . . 150
DBC Dictionary Tables . . . 152

Chapter 9: Using Teradata DSC in the Public Cloud . . . 154
Before You Begin - AWS . . . 154
Before You Begin - Azure . . . 154
Configuring Teradata DSC for the Public Cloud . . . 155
Configuring the DSC for the Public Cloud Using the CLI . . . 155
Proxy Server Support . . . 155

Appendix A: Administrative Tasks . . . 157

Appendix B: Required Access Rights . . . 197

Appendix C: Rules for Restoring and Copying Objects . . . 199

Appendix D: Values for XML Elements 202

Contents

Appendix E: Teradata DSA Commands 206

Appendix F: Additional Information 307

Data Stream Architecture

Welcome to Teradata DSA User Guide

Using the Teradata DSA User Guide

Teradata® Data Stream Architecture (DSA) helps you to back up and restore Teradata system data. DSA is optimized for Teradata MPP Architecture. It integrates with the Teradata® Viewpoint portlets: BAR Setup and BAR Operations. The portlets provide user interfaces to Teradata DSA that are like other Teradata ecosystem components. This integration uses Viewpoint account management features and enhances usability.

Why Would I Use this Content?

This content explains how to work with DSA to back up and restore your data from the Teradata system using its portlets.

How Do I Use this Content?

Use this content to learn how to configure, initiate, and monitor backup and restore jobs across various Teradata systems.

How Do I Get Started?

Start with BAR Portlets and Command-Line Interface to learn how to:

- Set up BAR environment.
- Configure systems and nodes, media servers, backup solutions, target groups, and alerts in BAR Setup portlet. These settings are available in BAR Operations portlet.
- You can use the BAR Operations portlet to create jobs, manage job settings, monitor job progress, and view job history details.
- Set up the DSC repository and system components.

References to Other Relevant Content

Refer to Related Documentation for more information on various DSA configurations and portlets.

Introduction to Data Stream Architecture

Teradata® Data Stream Architecture (DSA) enables you to back up and restore Teradata system data. DSA is optimized for Teradata MPP Architecture. It integrates with the Teradata® Viewpoint portlets: BAR Setup and BAR Operations. The portlets provide user interfaces to Teradata DSA that are similar to other Teradata ecosystem components. This integration uses Viewpoint account management features and enhances

usability. Teradata DSA also provides a command-line utility that you can use to configure, initiate, and monitor backup and restore jobs.

Data Stream Extensions and Data Stream Utility

Beginning with DSA 15.10, the product has been bundled into two components: Data Stream Extensions (DSE) and Data Stream Utility (DSU). Both components offer BAR portlet and command line functionality.

- DSE adds support for third-party backup applications. DSE offers advanced enterprise backup tools, such as scheduling, retention policies, archiving and allows customers to backup up directly to tape. DSE offers these backup targets:
  - Disk file systems
  - Veritas NetBackup
  - IBM Spectrum Protect
- DSU does not offer third-party backup application support. DSU is a solution offered for sites without a need for the extended footprint offered by Teradata DSE. DSU is also used for public cloud solutions. DSU offers these backup targets:
  - Disk file systems
  - Dell EMC Data Domain
  - Amazon S3
  - Azure Blob
  - Google Cloud

  In a typical DSU use case, the DSA Network Client (ClientHandler) is installed on the Teradata nodes, the DSC server is provided in a VM format, and a simple NFS environment is set up for use as a storage location for the backup files. A managed storage server can also act as a host server to the NFS environment if needed. When a Data Domain unit is used, EMC Data Domain Boost for DSU (DD Boost) allows a direct connection to the unit without using a third-party backup application.

  Note that DSA consumes a certain amount of memory on the Teradata nodes. The amount of consumption depends on factors such as the following:

  - Number of objects in the job plan
  - Configuration, including how many nodes and many streams
  - Throughput

  For an estimate you can use the following:

Fixed consumption	`2MiB + (Number of streams per node x 4MiB)` `Number of AMPs per node x 15MiB`
Variable consumption	With 80K objects, consumes around 1-2GiB per node (not constant consumption): `20MiB + (Number of objects x 400) + ((Total number of streams x 400) x 3)` For example, with 80K objects, 4000 total streams ~60MiB

Server Functionality

Server functionality includes the following servers:

- DSC server, which controls all BAR operations and is a part of all configurations. A DSC server must have the Data Stream Controller (DSC) installed.

  Teradata DSC can be installed on a physical server, AWS, Azure, Google Cloud, or a VM (Teradata DSC on VMware) and back up and restore data from and to the database on-premises, in AWS, Azure, Google Cloud, or VMware.
- Media server (physical or logical), which writes to the target storage device. A media server must have the DSA Network Client (ClientHandler) installed.

A machine in a DSA configuration can include different types of server functionality. For example, the managed storage server in a DSU configuration functions as disk storage, the DSC server, and a media server. In another configuration, the DSC server could be a standalone server.

Backup Solutions

DSA backup solutions can include any of the following:

- Dell EMC Data Domain
- Quantum Tape
- Disk file system
- Third-party backup application software, such as NetBackup or IBM Spectrum Protect
- Amazon S3
- Azure Blob
- Google Cloud Platform

BAR Integration

Teradata DSA features a Data Stream Controller (DSC) that controls BAR operations and enables communication between DSMain, the BAR portlets, and the DSA Network Client (ClientHandler). Teradata DSA records system setup information and DSA job metadata in the DSC Repository.

Data Stream Controller (DSC)

The DSC controls all BAR operations throughout an enterprise environment. The DSC is notified of all requested BAR operations and manages resources to ensure optimal system backup and restore job performance.

DSC Repository

The DSC repository is the storage database for job definitions, logs, archive metadata, and hardware configuration data. The DSC manages the repository using JDBC and is the only client component that can update the repository metadata. Direct access to the repository is not supported.

DSMain

DSMain runs on the Teradata nodes and receives job plans from the DSC. Job plans include stream lists, object lists, and job details. DSMain tracks the stream and object progress through the backup and restore process and communicates with the DSA Network Client.

DSA Network Client

The DSA Network Client (ClientHandler) controls the data path from DSMain to the storage device and verifies authentication from the database. The DSA Network Client then opens the connection to the appropriate device or API.

JMS Broker

Communication between the DSA components is performed using a JMS broker.

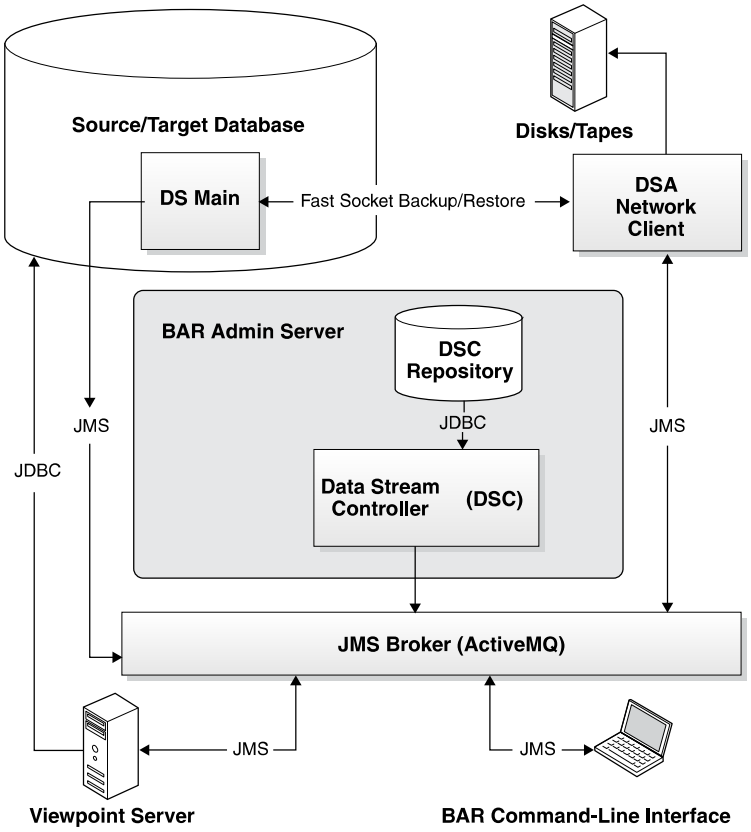
BAR Portlets

The BAR Setup and BAR Operations Viewpoint portlets allow you to manage the DSA configuration and job operations.

DSA Command-Line Interface (CLI)

The DSA command-line interface provides an alternative to the BAR portlets. The DSA command-line interface allows job launch, monitoring, and scheduling capabilities. It also provides commands to define DSA configuration.

DSA Components



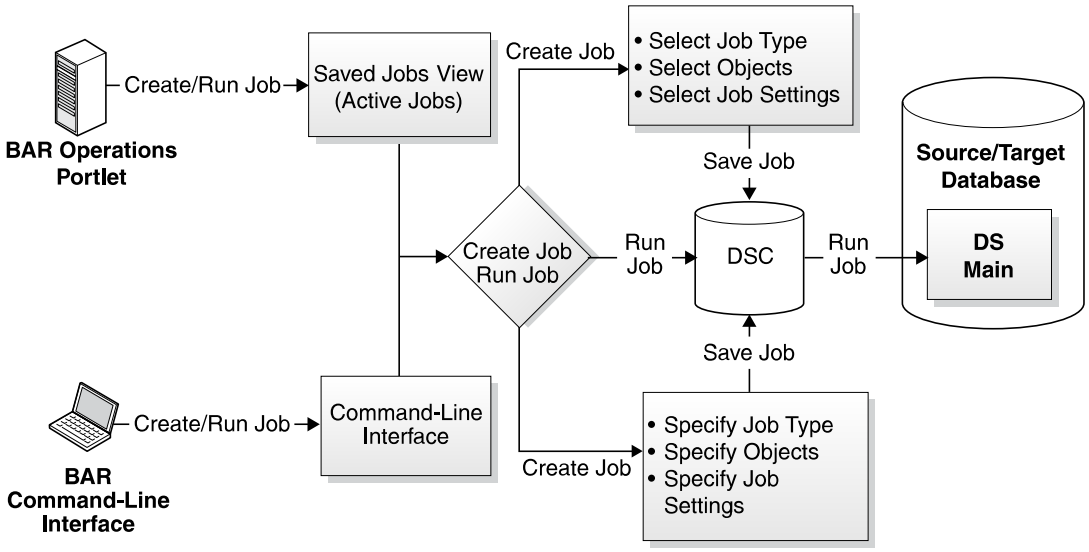
Note:

A CAM daemon resides on the Viewpoint server shown in the diagram. The CAM daemon is part of the alert messaging system. See Alerts for more information.

BAR Job Workflow

The BAR Operations portlet or DSA command-line interface communicates with the DSC when a backup, restore, or analyze job is created or run. The DSC controls the job flow by sending the job processing instructions to the appropriate DSA component. The DSC receives job status information from the DSA component and also notifies the database and other client applications of any action taken on a specific job. The job definition is stored in the DSC repository.

DSA Data Path



BAR Portlets and Command-Line Interface

Teradata DSA architecture has two user interfaces. The first interface, the **BAR Setup** and **BAR Operations** portlets, are Viewpoint portlets for DSA configuration and management. The second interface is a standard command-line interface (CLI) that includes the same functionality as the portlets, including DSA component configuration.

> **NOTICE**
>
> Enabling **General > Security Management** in the **BAR Setup** portlet increases security for the DSA environment. This setting requires users to provide Viewpoint credentials to execute some commands from the DSA CLI.

A DSA administrator might consider using the DSA CLI rather than the BAR portlets for specific situations, such as the following:

- Job scheduling, because this cannot be administered in the **BAR Operations** portlet.
- Exporting an XML file associated with a job created in the portlet.
- Streamlining the updates of multiple job definitions.
- Using scripts to automate DSA commands.

If you use job scripting automation through the DSA CLI, there should be a 30 second interval between DSC command requests, because BAR portlets are optimized to use caching in order to minimize the impact on DSC.

Initial Setup and Job Creation Using BAR Portlets

Setting up your BAR environment is a prerequisite for backing up your database. The systems configured and enabled in the **BAR Setup** portlet are available in the **BAR Operations** portlet. BAR setup configurations include systems and nodes, media servers, backup solutions, target groups, and alerts. These setup configurations are stored in the DSC repository, which manages BAR operations.

Note:

Beginning with DSA 16.10, nodes are configured through autodiscovery. You can view the node information but changing the node configuration is not allowed.

After configuring your BAR environment, you can use the **BAR Operations** portlet to create jobs, manage job settings, monitor job progress and view job history details.

Related Information:

Configuring BAR Setup
Managing Jobs

Configuration and Job Creation Using the Command Line

Setting up your BAR environment is a prerequisite for backing up your database. The DSA configuration commands allow you to set up the DSC repository and system components. Once configured other commands permit running jobs, viewing status, and other functions.

Note:

Beginning with DSA 16.10, nodes are configured through autodiscovery. You can view the node information but changing the node configuration is not allowed.

Related Information:

DSA Configuration: Command Line
Planning a Job: Command Line

DSA Features and Operations

New Features in this Release

DSA 17.20.04.00

- AJSE support at Job level
- Proxy Server Support for cloud platforms

DSA 17.20.03.00

- TDActiveMQ-5.17.2 support
- Teradata-jdk11 support
- Postgres upgrade to 10.22

DSA 17.20.02.00

- Support Azure AD authentication for Azure Blob
- Return DSA job responses within BaaS time limits
- Multiple DSCs sharing same ClientHandler Config File
- AJSE is disabled by default from DSA Client

DSA 17.20.01.00

- Support TDActiveMQ 5.16.4 to address Log4J vulnerabilities.

DSA 17.20.00.03

- Support for SLES 15 SP2 STND
- Upgrade log4j2 Version to 2.17.1 for addressing CVE-2021-44832han.
- Support for TDActiveMQ 5.16.1.0-6.
- AWS S3 Partner Attribution.
- Compatibility between TD repo and Postgres components.

DSA 17.20

- Support for Analytics Database 17.20 Release.
- TDActiveMQ Upgrade from 5.15.9 to 5.16.
- DSA Configuration and Validation Enhancements.
- GCP CMEK(Customer-Managed Encryption Keys) Support.
- Intuitive and Informative Error Message MVP2.
- Postgres to Postgres migration procedure.

- Postgres Upgrade to 10.17.

DSA 17.10.01.00

- Support 'clienthandler.properties' recovery during NFR.
- Intuitive and Informative Error Message MVP1.
- Update Postgres version used by the DSC repository to 10.15.

DSA 17.10

- Support for Advanced Job Status Enhancements (AJSE).
- Changed Block Backup Enhancement(CBBE).
- Incremental Restore with “include_DBC_Data” (Requires Analytics Database 17.20 or later).
- Support for Google Cloud through the BAR portlets.
- Support for multiple fabrics.

DSA 17.05

No new features for this release.

DSA 17.02.02

Added support of DSA on the following versions of SLES 12 SP3 for cloud and on-premises platforms.

- Cloud: SLES 12 SP3-TDC
- On-premises: SLES 12 SP3-STND

DSA 17.02.00

- Introduces support for the Google Cloud Platform. For this initial release, configuration takes place through the command line.
- Changes the DSC repository database from Analytics Database to Postgres.
  - Repository backups are made first to a landing zone.
  - From the landing zone, repository backups can be replicated to third party storage.

DSA Permissions

Users in a Viewpoint role that has been granted access to the **BAR Setup** portlet can use the portlet to add, remove, or edit the following resources in a BAR system configuration:

- Teradata systems
- Media Servers
- Fabrics
- Backup Solutions
- Target Groups

Viewpoint administrators can grant the BAR administrators privilege to any user mapped to a role. A BAR administrator has permissions to run all **BAR Operations** in the portlet.

Users who are not BAR administrators are only able to perform those actions on their own jobs, unless the job owner gives permission to that user.

DSA Restrictions

Teradata DSA currently has the following limitations:

- DSA does not support archiving or restoring journal definitions and data.
- DSA does not backup or restore volatile tables.
- Multibyte characters are not supported in the DSA command-line interface.
- [Teradata Database 15.0] Backup and restore jobs are subject to a database lock limit of up to 5,000 database objects.
- [Teradata Database 15.10.xx and earlier] The same user ID can run only one restore job at a time. If the user is already logged on and is running a BAR operation (including legacy BAR jobs), a DSA restore job is aborted.
- [Analytics Database 16.20 and later / Teradata Database 16.0 and later] The same user ID can run multiple restore jobs at once. However legacy BAR may reject the job, if the DSA job is started first for the same user.
- If the Teradata system restarts during an archive or restore operation, the host utility lock remains on the remaining unprocessed objects. The user must release the lock manually.
- To allow for parallelism during restore, the number of AMP Worker Tasks (AWT) dictates the number of DSA jobs that can run in parallel.

  [Analytics Database 16.20 and later / Teradata Database 16.10.01 and later] There is no longer a restriction, but Teradata recommends a maximum of three concurrent restore jobs be run on a system (based on 80 AWT). You can customize this value using Teradata Workload Management. See *Teradata Vantage™ - Workload Management User Guide*, B035-1197.
- Up to 20 backup jobs can be run concurrently (based on 80 AWT).
- DSA does not support an Analytics Database / Teradata Database version that is later than the DSA version.
- DSA does not support the Advanced Job Status Enhancements feature for Analyze and Repository jobs.

Azure: Use Multiple Storage Accounts for Single Backup

Azure users now have the option to send a single backup to multiple storage accounts. The use of multiple storage accounts provides additional flexibility and higher throughput.

To use multiple storage accounts, you can add storage accounts to the target group by using the **BAR Setup** portlet or the command line.

If you use the command line there is a new XML format. You can use `export_config` to get the new format and edit the XML. See also config_target_groups and config_target_group_map.

If you want to continue to use a single storage account, you aren't required to change anything.

Multiple DSCs

Prior to DSA 16.00, a separate Viewpoint server was required for each DSC. For example, a customer with production, test, and disaster recovery environments in separate locations was required to have three Viewpoint servers.

DSA portlets installation is no longer tied to a dedicated DSC instance. A single **Viewpoint** portlet can configure and monitor multiple DSCs. Also both Teradata systems of 16.0 and later and BARNC processes can subscribe to multiple DSCs.

The configuration of the DSC daemons is discovered based on the connection parameters (broker IP, broker port, and connection type).

Note:

Currently we only support one DSC server to one ActiveMQ server, you cannot configure one ActiveMQ server to multiple DSC servers.

After installation the configuration for each DSC is available using the **BAR Setup** portlet.

DSA and MAPS Architecture

Prior to Teradata Database 16.10, moving objects to a new hash map meant taking the system offline for a period of time. DSA takes advantage of the new Teradata MAPS Architecture introduced in Teradata Database 16.10 and later to allow you to back up objects from one map and restore them to another. See Objects Tab.

Component Deletion

A BAR component is an entity or defined relationship, such as a media server configuration, that is associated with a Teradata DSA job. A BAR component cannot be deleted from the Teradata DSA configuration if it is specified in a job, regardless of whether the job is in an active or retired state. Generally, in order for a component to be deleted, any job that references the component must be deleted first.

Exception: If the only reference to the BAR component is in a new job that has never been run, the component can be deleted.

Limitation: Components related to repository backups can never be deleted.

Copy and Restore Definitions

Copy

A copy operation moves data from an archived file to any existing Analytics Database / Teradata Database and creates a new object if the object does not already exist on the target database.

You can copy an object to create a new object with a different name, or keep the same name as the source object. In database-level copy operations, you can copy to a database with a different name or maintain the same name as the source.

Restore

A restore operation moves data to one of the following locations:

- From archived files back to the same database from which it was archived
- To a different database *if the DBC database from the source system is already restored to the destination system*

Copy Restrictions

To use the copy operation, the following conditions must be met:

- Restore access privileges on the target database or table are required.
- A target database must exist to copy a database or an individual object.
- When copying a table that does not exist on the destination system, you must have both CREATE TABLE and RESTORE database access privileges for the target database.

No Support for Copying DBC Database

Teradata DSA does not support copying the DBC database, which must be restored after a successful Teradata system initialization. Refer to Full System Restore Using the BAR Portlets or Full System Restore Using the Command Line.

No Support for Copying SYSUDTLIB Database

Teradata DSA does not support copying the SYSUDTLIB database, which is restored when the DBC database is restored.

Restriction on Copying TD_SERVER_DB Database

[Analytics Database 16.20 and later / Teradata Database 15.10 and later] TD_SERVER_DB database cannot be copied to a different name.

Copy Operations for Objects and Data Table Archives

Copy Operations: Large Objects (LOBs)

You can copy tables that contain large object columns. You can also copy large object columns to a system that uses a hash function that is different from the hash function used for the copy operation.

Copy Operations: Join Indexes and System Join Indexes (SJIs)

You can restore and copy join indexes, hash indexes, and system join indexes to the same database name. You can also copy system join indexes to a different database name.

Copy Operations: Data Table Archives

When you copy a data table to a new environment, Teradata DSA either creates a new table or replaces an existing table on the target database.

- DSA creates a new table if the target database does not have a table with the same name as the table being archived.
- DSA replaces the table if the target database has a table with the same name. The existing table data and table definition on the target database are replaced by the data from the archive.

When table data is copied, the following changes are allowed:

- Changing the data temperature
- Changing the data compression

HUT Locks in DSA

HUT Locks in Copy and Restore Operations

Copy and restore operations apply exclusive host utility (HUT) locks on all objects to be copied or restored.

SQL User-Defined Function

When you copy a SQL User-Defined Function, the corresponding DBC.DBASE row is locked with a write hash lock. This prevents the user from using any Data Definition Language on that database for the dictionary phase of the copy operation.

DSA Backup Locking Strategy for Offline Jobs

When processing a backup job begins, DSA gets a Host Utility (HUT) Read lock on every object in the job plan. If the job plan includes a database, DSA puts a Read HUT on the whole database. If it includes an object, DSA puts a Read HUT lock on the object. DSA requires all the HUT Read locks for every object in the job plan to be acquired before the objects are actually archived, so there is a consistent sync point and data integrity can be guaranteed when the data is restored.

Additionally, the backup job locks access on several DBC tables to get the object definitions. The locks are held for the duration of the DUMP command and are released as soon as possible. Once the DUMP command is complete, the locks are released so they are not held while the dictionary data is being written or during the rest of the dictionary phase.

Some of the tables that DSA locks are: TVM, DBASE, UDFINFO, TEXTTBL, IDCOL, DEPENDENCY, JAR_JAR_USAGE, ROUTINE_JAR_USAGE, ERRORTBLS, JARS, REFERENCEDTBLS, REFERENCINGTBLS, CONSTRAINTNAMES, TRIGGERSTBL, UIF_INFO, SERVERTBLOPINFO, DBCASSOCIATION, INDEXES, TVFIELDS, SERVERINFO, and TABLECONSTRAINTS. There is also a Read lock placed on the ARCHIVELOGGINGOBJSTBL. These are table level access locks.

The HUT Read lock is released as soon as the object is completely archived. For objects without a table header that are archived at the object level, the lock is released at the end of the dictionary phase. For tables that are archived at the object level, the lock is released as soon as the object is completely archived. For database level locks, the lock is released as soon as all the objects in the database have been completely archived.

The last database or object processed for the archive job requires the lock for the entire job.

DSA Backup Locking Strategy for Online Jobs

For online backups, the user can select the NoSync option. When NoSync is true, the locking process does the following:

1. Submits the Archive Logging On statement for all the objects in the job plan, which will try to lock all the objects to get a consistency point.
2. If DSMain cannot get the lock, it will do the following:
    a. Return the list of objects that had lock conflicts to DSC, and job_status_log can be used to display the objects with lock conflicts.
    b. Release the locks on those objects with lock conflicts, to ensure the locks are released in the event of a deadlock.
    c. Resubmit the lock statement for the objects with lock conflicts and wait indefinitely for the locks.

DSA Restore Locking Strategy

At the start of processing a restore job, DSA gets a Host Utility (HUT) Exclusive lock on every object in the job plan. Since the object definition and the data are being written, DSA needs an exclusive lock on these objects so there are no conflicts writing to the system. If the job plan includes a database, then DSA puts an Exclusive HUT lock on the whole database. If the job plan includes an object, then DSA puts an Exclusive HUT lock on the object.

In addition, the restore puts write locks on several DBC tables while the object definitions are being restored. These locks are held during the dictionary phase. Tables that are locked include: TEXTTBL, IDCOL, DEPENDENCY, JAR_JAR_USAGE, ROUTINE_JAR_USAGE, ERRORTBLS, JARS, STATSTBL, QUERYSTATSTBL, REFERENCEDTBLS, REFERENCINGTBLS, UNRESOLVEDREFERENCES, CONSTRAINTNAMES, TRIGGERSTBL, OBJECTUSAGE, UIF_INFO, SERVERTBLOPINFO,

DBCASSOCIATION, TVM, INDEXES, TVFIELDS, UDFINFO, and TABLECONSTRAINTS. Access locks are also placed on DBASE and DATASETSCHEMAINFO. These are table level locks.

A restarted job may place a table level HUT lock even if the original job definition is a database level restore.

For example, Job1 copies DB1 and DB2 at the database level. DB1 and DB2 have table1, table2, and table3 respectively, and the job aborts after DB1.table1 is complete. DB1 is partially complete and no object in DB2 is complete. In this case, the restart job places a table level lock on DB1.table2 and DB1.table3 and places a database level lock on DB2.

The HUT Exclusive lock is released as soon as the object is completely restored.

- Objects without a table header that are restored at the object level have the lock released at the end of the dictionary phase.
- Tables that are restored at the object level have the lock released as soon as the object is completely restored.
- Database level locks have the lock released as soon as all the objects in the database have been completely restored.
- [Analytics Database 16.20 and later / Teradata Database 16.0 and later] If the empty table option is selected during backup, the locks on the empty tables are released at the end of dictionary phase if they were restored at the table level.

Nowait Option for Offline Backups and Restore Jobs

As of DSA 15.10, jobs are always submitted with the Nowait option set to true for offline backups and restores. For offline backups and restores, the locking process will do the following:

1. Try to lock all the objects in the job plan.
2. If DSMain cannot get the lock, it will do the following:
    a. Return the list of objects that had lock conflicts to DSC. The job_status_log can be used to display the objects with lock conflicts.
    b. Release the locks on those objects with lock conflicts, to ensure the locks are released in the event of a deadlock.
    c. Resubmit the lock statement for the objects with lock conflicts and wait indefinitely for the lock. This lock statement also indicates the objects that had a lock conflict when the lock statement was submitted the first time.

Active and Retired Jobs

An active job refers to any job that is not scheduled to be deleted. Active jobs can be run from the **BAR Operations** portlet or the DSA command-line interface. When the job has a deletion date, it is considered retired and cannot be run. You can still access the job history, which is the log of each specified job run, until the deletion date of the job. The job and job history are deleted from the DSC repository at the deletion date of the job.

A DSC repository job is specifically designed to back up or restore the DSC repository. A DSC repository job does not have an active or retired designation, so DSC repository jobs are always considered active and cannot be retired.

Multiple DSA Domains

If your site has multiple DSA domains, the BAR admin user can export metadata from DSA domain A and import it later to DSA domain B using the DSA CLI. This migration is performed for each job and is necessary before any restore operation can be done on the target DSA domain B. As part of the migration, the administrator is responsible for transferring the related information for the NetBackup catalog for DSE operations. For DSU, the administrator is responsible for the management of related files on the disk file system.

Related Information:

DSA Job Migration to a Different Domain
Exporting Job Metadata
Importing Job Metadata
Using the export_import Tool to Automate DSA Job Migration to a Different Domain

Public Cloud: Default Target Group for Elasticity Scale Out/In

Beginning with DSA release 16.10, when DSC is deployed in the Public Cloud with ClientHandler installed on all the nodes, a default target group is created automatically. This default target group expands and shrinks when a database system scales out or in (you may see fold/unfold on some parameters).

The name of the default target group is as follows:

`defaultDFTG`*system_name*

where DFTG is disk file system target group and *system_name* is the name of the system.

The following events trigger the creation of the default target group:

- Configuring or reconfiguring a system
- Adding a backup solution
- Adding, deleting or restarting a media server

See the following examples of each scenario.

Default Target Group Example - Configuring or Reconfiguring a System

When a system is configured, a new system configuration is created to represent the nodes associated with the system. If all nodes have ClientHandler installed, a default target group is created. The number of devices allocated is based on the node level soft limit of the system load. See the following example:

System1 has four nodes. All nodes have ClientHandler installed.

node1 - Number of AMPs = 10

node2 - Number of AMPs = 20
node3 - Number of AMPs = 10
node4 - Number of AMPs = 20

The backup solution is defined for disk file system with two paths:

File system name: `/path1/` **Max number of open files:** `100`
File system name: `/path2/` **Max number of open files:** `400`

The default target group is created with the following configuration:

Target group name: defaultDFTGSystem1

Media Server	File System Path	Devices
ms1	`/path1`	2
	`/path2`	8
ms2	`/path1`	4
	`/path2`	16
ms3	`/path1`	2
	`/path2`	8
ms4	`/path1`	4
	`/path2`	16

ms1, ms2, ms3, ms4 are media servers on node1, node2, node3, and node4 respectively.

Default Target Group Example - Creating a Backup Solution

If a new backup solution is created based on the preceding system, a new default target group is created. See the following example:

System1 has four nodes. All nodes have ClientHandler installed.

node1 - Number of AMPs = 10
node2 - Number of AMPs = 20
node3 - Number of AMPs = 10
node4 - Number of AMPs = 20

The backup solution is defined for disk file system with three paths:

File system name: `/path1/` **Max number of open files:** `100`
File system name: `/path2/` **Max number of open files:** `400`
File system name: `/path3/` **Max number of open files:** `500`

The default target group is created with the following configuration:

Target group name: defaultDFTGSystem1

Media Server	File System Path	Devices
ms1	/path1	1
	/path2	4
	/path3	5
ms2	/path1	2
	/path2	8
	/path3	10
ms3	/path1	1
	/path2	4
	/path3	5
ms4	/path1	2
	/path2	8
	/path3	10

ms1, ms2, ms3, ms4 are media servers on node1, node2, node3, and node4 respectively.

Default Target Group Example - Adding, Deleting, or Restarting a Media Server

During unfolding, for example, from 4 to 8 nodes, 4 new nodes are added. Once all 8 nodes have ClientHandler installed, a default target group is created with all 8 media servers.

If a default target group already existed with 4 media servers, a new configuration is created with the new set of 8 media servers.

During folding, for example from 8 to 4 nodes, 4 nodes are removed from the system and the media servers on the nodes go offline. A new configuration (and default target group) is created with the available 4 media servers.

Restarting a media server behaves the same way as adding a new media server. A default target group is created after all the media servers are available.

Define Data Paths Using Network Fabric

Beginning with DSA 16.10.01, you can define network fabric. If a network fabric is defined, it applies a logical network mask when a job plan generates data paths between AMPs and BARNC processes acting as media servers. Prior to DSA 16.10, network masks were used for the routing of traffic between nodes and media servers. Starting with DSA 16.10 all nodes are autoconfigured and node IP addresses can no longer be edited or removed.

Customers who use their media servers for multiple Teradata systems may need to use network fabric. For example, if the media servers and TPA nodes are connected both through BYNET® and a LAN interface, defining fabric is required.

The network fabric belongs to a specific Teradata system and defines the path from the TPA node to the BARNC IP address. If fabric is defined, job submission skips the logical net mask and selects the existing fabric when defining job data paths.

The fabric component can include all the nodes and their paths to one or more media servers.

The target group must define one media server that belongs to the network fabric. For best performance, include all media servers that belong to the fabric.

For DSA release 16.20.12 and later, you can define the network fabric using either the **BAR Setup** portlet or the CLI config_fabrics command.

You can define multiple fabrics but each must be unique. For example, System A can have two fabrics for two target systems B and C, fabricAB and fabricAC. The information is saved into the repository and can be edited if new media servers or nodes are added.

For multiple fabrics, update `dsc.properties` for the number of fabrics allowed per Teradata system: `maxlimit.fabric=`*number-fabrics*. Default is 64.

The fabric is used only for backup, restore, and analyze_validate jobs.

Network Fabric Considerations

Important:

Network fabric is not supported for repository backup or restore.

- Define a fabric that includes all the nodes
- Use one or more IP address per media server
- Do not edit media server IPs after they are discovered
- Use multiple media servers per node, if possible
- For best performance:
  - Define a target group that includes all the media servers from the fabric
  - Make sure `dsc.properties` has `validate.fabric=on`

    For more flexibility, `validate.fabric=off` allows a job plan to be generated for a fabric that defines at least one node to a media server IP. Those undefined paths use bynet to move the data.

Job Status Enhancements

Beginning with DSA 16.20 and Analytics Database 16.20, enhanced job status information is provided to give you better information during the progress of the job. While the job is running, status is provided on a maximum of 500 INPROGRESS objects. When the job is complete, the maximum bucket size (number of objects returned per request) is 100 objects. The table gives examples of the type of status now provided.

Use Case	Additional Information
Dictionary ONLY job	Processed query count, processed table header, and processed dictionary row
Data job on single object	Byte counts increase over time
Data restore job	Object that is in progress
DBC restore job	Information about space accounting and DBC restore status

Advanced Job Status Enhancements

Advanced Job Status Enhancements provides a detailed status of running jobs and the progress of DSA jobs and commands. It provides detailed and consistent job progress and object status, for example, which step a job is currently running. In addition, it provides advanced reporting for job_status and job_status_log commands.

Beginning from jobs using the DBS 17.10, the Advanced Job Status Enhancements are enabled. However, AJSE feature is disabled by default from DSC 17.20.02, for all jobs, though supported by DBS. Beginning 17.20.04.00, it can be enabled at the job level even though it is disabled at DSC level for all jos. To enable the feature, set `adv_job_progress_stats` to TRUE in job_options during job creation.

Advanced Job Status Enhancements has a flag in `dsc.properties` that allows you to turn the feature on for all jobs if required. To enable the AJSE features for all the jobs, set the field `disable.ajse` to FALSE defined in the `$DSA_CONFIG_DIR/dsc.properties` file.

The following tables show new fields for Advanced Job Status Enhancements. They do not appear for analyze and repository jobs.

Job Status

New Field	Description	Applies to
HUT Lock Status	Overall lock status of the job's objects.	Backups, Restores, Data Migration, Data Movement, Copy jobs.
HUT Locks Acquired	Timestamp in the Dictionary phase that indicates when the job's objects have been locked.	Backups, Restores, Data Migration, Data Movement, Copy Jobs.
Task Sets	Given a job that is restored to a system with less AMPs than from the system it was backed up, amount of passes the streams need to do on the backup files in order to complete the restore. This field appears in the Data Phase.	Restores, Data Migration, Data Movement, Copy Jobs.
Primary Subtables Sorted Count, Online Subtables Sorted, Count Primary Lob, Subtables Sorted Count.	This information displays between Data and Build Phase.	Restores, Data Migration, Data Movement, Copy Jobs.

New Field	Description	Applies to
Fallback Subtables Built Count, Fallback Lob Subtables Built Count, Secondary Index Subtables Built Count, Fallback USI Subtables Built.	This information displays in the Build phase.	Restores, Data Migration, Data Movement, Copy Jobs.

Job Status Log

New Field	Description	Applies to
Online Status	Indicates whether the given object is online.	Backups, Restores, Data Migration, Data Movement.
Online Enabled Timestamp	Displays timestamp when the given object becomes online.	Backups, Restores, Data Migration, Data Movement.
Subtable type, State, Sort Start Time, and Sort End Time	Provides information about the given subtable object's sort phase.	Restores, Data Migration, Data Movement, Copy Jobs.

Limitations

Advanced Job Status Enhancements do not support ELM jobs as there is no phase identity in this type. ELM jobs allow for each object to perform dictionary and data independent of each other. Regular jobs make all objects go through dictionary phase and data or build phase together, hence it provides phase details for regular jobs.

Restart a Restore Job

Beginning with DSA 16.20.12 and Analytics Database 16.20 Feature Update 1, you have the ability to restart an restore job that failed, aborted, or completed with errors or warnings. The restarted job begins after the last successful step. This functionality is available from both the **BAR Operations** portlet and the command line.

The following items prevent a restore job from restarting:

- DBC only restore cannot be restarted
- Original restore job did not complete dictionary phase
- Change in job definition except: logging level, job description, retire date, or job state
- AMP down
- SYSINIT since original job

A restarted job may place a table level HUT lock even if the original job definition is a database level restore.

For example, Job1 copies DB1 and DB2 at the database level. DB1 and DB2 have table1, table2, and table3 respectively, and the job aborts after DB1.table1 is complete. DB1 is partially complete and no object in DB2

is complete. In this case, the restart job places a table level lock on DB1.table2 and DB1.table3 and places a database level lock on DB2.

Related Information:

Restarting an Unsuccessful Restore Job
Viewing Job Status: Portlets
run_job
job_status
job_status_log
list_job_history

Hot Standby Node Stream Distribution Enhancements

Prior to DSA 16.20.50, DSA used an internal ID to distribute the AMP stream. When the system configuration changed, for example, HSN becomes active, DSA did not guarantee the data from the same AMP would go to the same target device.

Beginning with DSA 16.20.50 or later, a new flag, **backup.sortByAmp**, is defined in the `$DSA_CONFIG_DIR/dsc.properties` file. It changes how the AMP stream distributes to third party media. With the default value of true, DSA uses the order of the AMP numbers to distribute the stream. This makes sure that DSA always uses the same stream distribution.

After you upgrade to DSA 16.20.50, backup job AMP distribution is different from previous job execution. After a new full backup is run, future AMP streams distributed to third party media are guaranteed as long as the AMP count and target group configuration remains the same, and media servers defined in the target group are all online.

For DDBoost after the upgrade to DSA 16.20.50:

- When `backup.sortByAmp=true` (the default), gen0 is generated for all existing customer backup jobs until the first full backup is run. Teradata recommends that you run a full backup as soon as possible. After the full backup is run, the HSN or down node becoming active will not trigger a gen0 backup.
- When `backup.sortByAmp=false`, gen0 is generated every time HSN becomes active and again after the down node becomes active. Avoid this by identifying when a gen0 backup may occur and setting the flag to true and running a full backup.

Enhanced Locking Management

Normally, DSA locks all objects at the beginning of the backup. The object is unlocked once it is backed up. This ensures data consistency between objects but keeps all objects locked until the backup is complete.

With enhanced locking management, DSA locks one object at a time and unlocks it once it is backed up. DSA does not manage data consistency. Use enhanced locking management on objects that are not empty.

NOTICE

Severe performance degradation and failed jobs can occur when using enhanced locking management on a large number of empty objects. For best job performance, use a separate backup without enhanced locking management to back up databases with many small objects.

DSA CBB Backup and Restore

Teradata implements backups using changed block backups (CBB). With CBB, a system can run a full backup or only back up data blocks that have changed since a prior backup. This greatly reduces backup time and storage, at the cost of increased restore time. Restore time is increased because DSA has to read multiple backup save sets and construct the complete save set to restore. CBB allows cumulative or delta backups but restores must be full restores (For incremental backup and restore, see Using Incremental Restore to Maintain Your DR System). CBB backup is applicable to both standard (offline) backups and online archive.

Use cumulative and delta backups for the following:

- Databases and tables that have a very low change rate compared to table size
- Primary Partition Index (PPI) tables where changes are limited to one or few partitions

Three types of backups are available: full, delta, and cumulative.

Backup Types

The first backup must always be a Full backup. The full backup is the baseline for all further backups.

Full

A *full* backup archives all data from the specified objects. This backup takes the longest time to complete, and uses the most backup storage space. However, a full backup has the shortest restore time, since all data required to restore the objects is contained within a single backup save set.

Delta

A *delta* backup archives only the data that has changed since the last backup. This backup completes in the shortest time and use the least storage space. However, a delta backup increases the time to restore the database, as it may add many backup save sets that must be processed before a set of objects can be fully restored.

Cumulative

A *cumulative* backup archives the data that has changed since the last full backup. This backup type consolidates changes from multiple delta backups or cumulative backups before a full backup is run. A cumulative backup has a shorter database restore time than a series of delta backups, and takes less time and space than a full backup.

Guidelines for Backups

- Regardless of the type of backup performed, the dictionary information for all objects is fully backed up. This makes sure that all non-data objects and object definitions are fully recovered in the event of a restore from any point in time.

- In the event of a restore or analyze_validate, you select the backup save set corresponding to the point in time when you want the objects restored. This can be a full, delta, or cumulative backup save set. For a given restore point, the following save sets are processed, relative to the selected backup save set:
  - Any delta backups after the most recent full or cumulative, and the selected restore point
  - The most recent cumulative backup, if any. Only if newer than the full backup.
  - The most recent full backup

In the event of an analyze_read, only the selected save set is analyzed.

NOTICE

Running a cumulative or delta change block backup of a DBC ALL backup job does not include the DBC system tables. The DBC database is used when you need to restore the whole system after a system initialization (sysinit). Therefore, run a separate FULL backup of the DBC database for every CBB backup job cycle run.

Enhanced Change Block Backup

You can set your system configuration to allow you to run backup jobs based on a prior full or cumulative job that completed with errors. See *Usage Notes* in config_systems for skip_force_full option.

Prior to DSA 17.10, a job that completed with errors requires the next job to be a full backup or cumulative backup. The exception is if skip_force_full is true. With DSA 17.10, Enhanced Change Block Backup allows erroneous objects from previous backups to be fully archived, ensuring data consistency during restore. This enhancement means you no longer need to run full or cumulative backups after a backup that completed with errors.

1. The CBB enhancement is on by default. Make sure to run the Configured Teradata System on 17.10 or later to avoid override on the default. To turn it off, set "dsc.cbb.enhancement=FALSE" in dsc.properties file.
2. Check if ECBB is on using the list_components command.
3. When Enhanced CBB is ON, skip_force_full is ignored.

Example Backup Strategy

Consider a site that performs a full backup every Sunday, a cumulative backup every Wednesday, and delta backups on the other days:

The table shows which save sets are restored when you select a given restore date.

Save Set	Day	Type	Restore These
1	Sunday	Full All data for all tables.	1
2	Monday	Delta	2, 1

Save Set	Day	Type	Restore These
		Data that changed today.	
3	Tuesday	Delta Data that changed today.	3, 2, 1
4	Wednesday	Cumulative All changes since the full backup.	4, 1
5	Thursday	Delta Data that changed today.	5, 4, 1
6	Friday	Delta Data that changed today.	6, 5, 4, 1
7	Saturday	Delta Data that changed today.	7, 6, 5, 4, 1

Incomplete Backups

If any part of the CBB backup is lost or corrupted, the integrity of the database data is compromised. If any delta, cumulative, or full backup save set required for a restore is missing or corrupt, a restore from any dependent backup save set fails.

Incomplete backups are not subject to this limitation. An incomplete backup occurs if any CBB backup completed with errors, or was aborted and not rerun. In the event of a failed backup, prior and subsequent CBB backups are not affected. Similarly, when a backup completes with non-fatal errors, prior and future backups do not use the backup image that received an error. Instead, subsequent backup jobs use the most recent successful backup as the base.

If necessary, you can use the system configuration flag skip_force_full to allow a backup if the previous backup completed with errors. It is important to fix the underlying cause of any error that occurs during backups, and to rerun the backup at the next available opportunity. See *Usage Notes* in config_systems for skip_force_full option.

The following backup jobs are always run as a full backup:

- DBC only backup job
- Backup job in dictionary phase

Changes That Force a Full Backup

The following situations require that a new full backup is generated before any further delta or cumulative backups are run:

- The system has gone through SYSINIT and/or a full DBC restore since the most recent full backup

- The system has had an access module processor (AMP) reconfiguration or rebuild since the most recent full backup
- Any of the following changes in the backup job from the portlet, xml, or command line:
  - Object list
  - Data phase
  - Target group (target_media)
  - No sync (nosync)
  - Track empty tables (track_empty_tables)
  - Skip join/hash index (skip_joinhash_index)
  - Limit the maximum streams per node for this job (streams_softlimit)
  - Skip statistics (skip_stats)
  - Query Band (query_band)
- The last full save set in the DSC repository has been removed or automatically retired
- The backup job is in the NEW state
- The backup job failed

Changes That Force a Cumulative Backup

You now have the option to rerun a failed cumulative backup job with another cumulative run instead of having to run a full backup.

If the source system is configured with skip_force_full, cumulative jobs that end in errors can be rerun as cumulative as well. See *Usage Notes* in config_systems for skip_force_full option.

Using Incremental Restore to Maintain Your DR System

Beginning with Analytics Database 16.20 Update 2, DSA allows incremental restores (IR) using delta and cumulative backup save sets. The primary use case for this feature is for keeping two systems in sync; for example, regular and disaster recovery systems. Incremental restore allows a delta or cumulative backup to be restored without including a full backup in the same restore job. This feature reduces restore time compared to the standard change block implementation of only allowing full restores. To prevent corruption and keep the systems in sync, incremental restore leaves the table in a read-only state.

Requirements for Incremental Restore

These criteria must be met:

- Analytics Database 16.20 Update 2
- Both the source and target database must have incremental restore enabled.

  Important:

  Teradata Support must enable incremental restore on the databases.

- Both the backup and the restore job must have incremental restore enabled.

Options for Incremental Restore

From an IR-enabled backup, you can select the following three types of restores:

- Next incremental restore: Restores the next save set after the last one that was restored.
- Specify a version (Full): Restores from the selected save set to the last full save set.
- Latest version (Full): Standard CBB restore, from the last available CBB save set to the last full backup.

Related Information:

Creating and Running an Incremental Restore Enabled Backup Job: BAR Operations Portlet
Creating an Incremental Restore Job: BAR Operations Portlet
Creating or Updating a Job: Command Line

Incremental Restore Limitations

For the first phase of incremental restore with Analytics Database 16.20 Update 2, the following limitations and restrictions apply:

- DBC only jobs are not supported.
- The incremental restore job must be created from an incremental restore enabled backup job.
- The next incremental restore option can only process one save set at at a time. You select the backup to start from and run the job repeatedly till you get to the save set you want restored.

- You cannot stop this command: build_restore_objects.
- You can stop these commands only through cnsterm 6:
  - To stop cleanup_ir_source_system, run this command from the source database:

    ```
    start bardsmain -a|abort c|cleanup
    ```
  - To stop reset_ir_target_system, run this command from the target database:

    ```
    start bardsmain -a|abort r|reset
    ```
- SQLE 17.00.09.01 and later versions support online incremental restore.
- You cannot restart an incremental restore job.
- Enhanced locking management is not supported with IR.
- IR does not work with a down AMP.
- LOGGING INCREMENTAL ARCHIVE is not retried indefinitely.

Changes That Force Full Backup of IR-Enabled Backup Job

Certain changes to the Job Definition of an IR-enabled backup require a full backup instead of incremental.

Changes to These Elements Force a Full Backup

XML File	BAR Operations Portlet
<skip_stats>	Advanced Settings button: Skip statistics
<track_empty_tables>	Advanced Settings button: Track empty tables
<skip_joinhash_index>	Advanced Settings button: Skip join/hash index

Changes to These Elements Do Not Affect IR-Enabled Backup

XML File	BAR Operations Portlet
<job_description>	Description
<auto_retire>	Job Settings Tab: Automatically Retire
<retire_value>	Job Settings Tab: enter value
<retire_units>	Job Settings Tab: DAYS or WEEKS
<dsmain_logging_level>	Job Settings Tab: Logging Level
<query_band>	Job Settings tab: Query Band
<skip_archive>	Abort On Access Rights Violation checkbox
<streams_softlimit>	Advanced Settings button: Limit the maximum streams per node for this job

XML File	BAR Operations Portlet
anything within <objectlist>	Objects tab: Objects
<target_media>	Target Group
<data_phase>	Job Settings tab: Backup Method Offline=data phase

Incremental Restore with DBC

1. Update DBC tables without a full system migration. The data for these tables will be available in the data archive.
2. Use job option to enable retrieve data for the DBC tables during archive and restore during Incremental Restore.

Important:

Entire data in these DBC tables are archived and all the rows from the source system are restored. Any changes in the DBC data on target system, such as creating a new database are lost after the restore.

Changes That Force Full Restore of IR-Enabled Restore Job

Certain changes to the Job Definition require a full backup for an IR enabled backup instead of incremental.These changes include the following:

- reset_ir_target_system was run on the IR-enabled target system
- Reset was run from the BAR Setup portlet on the target system:

  DSC Servers > Systems and Nodes > Systems > IR Support > Reset
- build_restore_objects was run with the -a|allow_write option on the IR-enabled restore job
- Build restore objects was run from the BAR Operations portlet:

  From an IR-enabled restore job, select **> Build Restore Objects**
- run_job was executed with the -a|allow_write option on the IR-enabled restore job
- Run job with allow write option was run from the BAR Operations portlet:

  From an IR-enabled restore job, select **> Run Job**, then select **Allow write for objects after restore**.

Changes to These Elements Force a Full Restore

XML File	BAR Operations Portlet
anything within <objectlist>	Objects Tab: Objects
<all_backup_objects>	Objects Tab: Objects

XML File	BAR Operations Portlet
<source_media>	Target Group
<enable_temperature_override>	Advanced Settings: Override Temperature
<temperature_override>	Advanced Settings: Override Temperature
<block_level_compression>	Advanced Settings: Block Level Compression
<disable_fallback> • [Analytics Database 16.20 Update 1 and later] Fallback cannot be disabled. Entry is ignored. • [Prior versions of Analytics Database / Teradata Database] Disable fallback is not available unless run_as_copy is true.	Job Settings tab: Disable fallback **Note:** • [Teradata Database 16.20 Update 1 and later] Fallback cannot be disabled. Checkbox is ignored. • [Prior versions of Teradata Database] Disable fallback is not available unless **Run as copy** is checked. The icon ⃠ appears when the mouse pointer is hovered over the checkbox.
<query_band>	Job Settings Tab: Query Band
<run_as_copy>	Job Settings Tab: Run as copy
<skip_stats>	Advanced Settings button: Skip statistics
<config_map_name>	Advanced Settings button: Destination System Hash Map
<map_to>	Advanced Settings button: Map to a different database on destination system
<skip_joinhash_index>	Advanced Settings button: Skip join/hash index
XML File	**BAR Setup Portlet**
<target_tdpid>	target systems (destination system) **DSC Servers > Systems and Nodes > Systems**

Changes to These Elements Do Not Affect IR-Enabled Restore

XML File	BAR Operations Portlet
<job_description>	Description
<auto_retire>	Job Settings Tab: Automatically Retire
<retire_value>	Job Settings Tab: enter value
<retire_units>	Job Settings Tab: DAYS or WEEKS
<dsmain_logging_level>	Job Settings Tab: Logging Level
<reblock>	Not available in portlet for edit
<skip_archive>	Abort On Access Rights Violation checkbox

XML File	BAR Operations Portlet
<concurrent_bld_per_tbl>	Advanced Settings: Concurrent Builds Per Table
<streams_softlimit>	Advanced Settings: Limit the maximum streams per node for this job
<skip_build_secondary_indexes>	Advance Settings: Skip build secondary indexes
Changing between next_incremental_ restore and backup_version	

Incremental Restore Examples

This example describes how incremental restores work when using full, cumulative, and delta IR-enabled backups in a five day backup strategy.

Save Set	Type	Description
1	full	Contains data from all tables
2	delta	Contains changed data blocks since last backup
3	full	Contains data from all tables
4	delta	Contains changed data blocks since last backup
5	cumulative	Contains all changes since previous full backup

If you create an IR-enabled restore job in the **BAR Operations** portlet and select **Latest version (Full)**, the selected backup save sets will be 5 and 3.

If you create an IR-enabled restore job using **Specify a version (Full)** in the **BAR Operations** portlet or `<backup_version>` in the job xml file, you get the following results:

- If you select Save Set 1, you get save set 1.
- If you select Save Set 2, you get save sets 1 and 2.
- If you select Save Set 5, you get save sets 5 and 3.

Next incremental restore normally applies only one save set at a time. To restore multiple save sets, you will need to run the job multiple times. If you create an IR-enabled restore job using **Next incremental restore** in the **BAR Operations** portlet or with <next_incremental_restore>=true, you get the following results:

- A newly created restore job points to the last created full backup (previous Sve Set 3).
- If a previous restore has occurred: if the last applied (restored) save set is 1, only save set 2 is used.
- If any of the conditions that force a full restore have occurred (see Changes That Force Full Restore of IR-Enabled Restore Job) and Save Set 1 was the last restored, then both save set 1 and 2 are selected.

Difference Between CBB Restore and IR-Enabled Restore

This table shows the difference between CBB restore and IR-enabled restore for a daily backup and restore to the DR system.

Save Set	Day	Type	CBB Restores These	IR-Enabled Restores These
1	Sunday	Full All data for all tables.	1	1
2	Monday	Delta Data that changed today.	2, 1	2
3	Tuesday	Delta Data that changed today.	3, 2, 1	3
4	Wednesday	Cumulative All changes since the full backup.	4, 1	4
5	Thursday	Delta Data that changed today.	5, 4, 1	5
6	Friday	Delta Data that changed today.	6, 5, 4, 1	6
7	Saturday	Delta Data that changed today.	7, 6, 5, 4, 1	7

Incremental Restore Workflow

General Use

This presents a high-level overview of the steps needed for incremental restore.

1. Have Teradata Support enable incremental restore (IR) on the source and target databases.
2. Verify that IR is enabled on both source and target databases.
   - Verifying Incremental Restore is Enabled: BAR Setup Portlet
   - Verifying Incremental Restore is Enabled: Command Line
3. Create IR-enabled backup job.
   - Creating and Running an Incremental Restore Enabled Backup Job: BAR Operations Portlet
   - Creating or Updating a Job: Command Line (specifically adding the `enable_backup_for_IR` option in create_job)
4. Run IR-enabled backup.
5. Create IR enabled restore job.

- Creating an Incremental Restore Job: BAR Operations Portlet
- Creating or Updating a Job: Command Line (specifically looking at the `enable_incremental_restore`, `skip_build_secondary_indexes`, `next_incremental_restore`, and `backup_version` options in create_job)

Disabling Incremental Restore

Disabling IR requires three separate processes: having Teradata disable it on the target system, disabling it at the object level on the target system, disabling incremental backup on the source system.

Disable IR on the Target Database at System Level

1. Have Teradata Support disable IR on the target database.

Disable IR on the Target Database at the Object Level

2. Use one of these techniques to disable IR:

Choice	Description
Run an IR restore job with the allow write option	Use one of these methods: • Run run_job with the -a|allow_write option on the IR-enabled restore job • At the BAR Operations portlet, from an IR-enabled restore job, select > **Run Job**, then select **Allow write for objects after restore**.
Run build restore objects with the allow write option	Use one of these methods: • Run build_restore_objects with the -a|allow_write option on the IR-enabled restore job • At the BAR Operations portlet, from an IR-enabled restore job, select > **Run Job**, then select **Allow write for objects after restore**.

Disable IR Enabled Backup on the Source System

3. Run cleanup on the IR disabled source system using one of the following:
   - Run cleanup_ir_source_system (Use list_cleanup_ir_source_system to display the output.).
   - On the BAR Setup portlet, select **DSC Servers > Systems and Nodes > Systems > IR Support > Cleanup**.

Re-Enabling Incremental Restore

Use these instructions only to re-enable IR after it has been disabled. You cannot initially enable IR using these steps.

1. Reset IR on the target system:
   [Command line] Run reset_ir_target_system.
   [BAR Setup portlet] Go to **DSC Servers > Systems and Nodes > Systems > IR Support** and click **Reset**.

2. Create then run an IR restore job either as latest version (full) or with a specified version. Do not use next incremental restore.
   [Command line] Run create_job with enable_incremental_restore=true and next_incremental_restore=false.
   [BAR Operations portlet] Create an IR enabled restore job, and **Edit** the Source Save Set. Select either **Latest version (Full)** or **Specify a version (Full)**

Verifying Incremental Restore is Enabled: BAR Setup Portlet

Repeat for both source and target database systems.

1. Open the **BAR Setup** portlet.
2. Under **DSC Servers**, select your DSC server.
3. From the **Categories** list, select **Systems and Nodes**.
4. From the **Systems** list, select the system to verify.
5. From the **Setup** list, select **System Details**.
   Look for the following:

```
Incremental restore for backup is: Enabled
Incremental restore for restore is: Enabled
```

Verifying Incremental Restore is Enabled: Command Line

Repeat for both source and target database systems.

1. Issue `list_components` command.
   `dsc list_components -n` *system-name* `-t system`
2. Check that these columns have `Y`.
   `Is Source IR Enabled?`
   `Is Target IR Enabled?`

Verifying Incremental Restore for DBC is Enabled: Command Line

1. 1. Issue list_components command. `dsc list_components -n system_name -t system`
2. 2. Check that the column ‘Is Include DBC Supported?’ has Y.

Creating and Running an Incremental Restore Enabled Backup Job: BAR Operations Portlet

Create the Job

1. Open the **BAR Operations** portlet.
2. From the **Saved Jobs** view, click **New Job**.
3. On the **New Job** screen:
   a. For **Job Type**, select **Backup for incremental restore**.
   b. Click **OK**.
4. Under **New Backup Job**:
   a. Enter a unique **Job Name**.
   b. Select a **Source System**.
   Only systems with incremental restore enabled are listed.
   c. In **Enter System Credentials**, enter a user name and password for the system.

   Account String information is not required.

   Note:

   The password is applied to all jobs associated with this system and user account.

   ---

   d. Select a **Target Group**.
   e. [Optional] Enter a job description.
5. Select **Objects**.
6. Select the objects from the source system to backup.
7. [Optional] Select **Selection Summary** to verify the selected objects..

   Note:

   Size information is not available for DBC only backup jobs. N/A displays as the size value for DBC only backup jobs.

   ---

8. [Optional] To adjust job settings for the job, click the **Job Settings** tab.
9. Click **Save**.
   The newly created backup job is listed in the **Saved Jobs** view.

Run the Job

10. Click next to a job.
11. Select **Run**.
12. Select **OK**.
13. Select the **Job Scope**: **Full**.

A newly created job only has Full as the available Job Scope. When you rerun an existing job, Delta and Cumulative are available.

14. Select **Run**.

Related Information:

Changing Job Permissions
Job Settings Tab

Creating an Incremental Restore Job: BAR Operations Portlet

Create the Job

1. Open the **BAR Operations** portlet.
2. From the **Saved Jobs** view of the **BAR Operations** portlet, do one of the following:

Option	Description
Create a new job	a. Click **New Job**. b. Select Job Type: **Restore as incremental** and click **OK**.
Create a job from a backup job save set	a. Click next to an IR-enabled backup job that has completed. You can select **Incremental Restore Supported** as a column so you can see which backup jobs are IR enabled. b. Select **Create Restore as Incremental** to create an incremental restore job from the selected save set.

3. Under **New Restore Job** enter a unique **Job Name**.
4. If the **Source Save Set** you want to use is not already displayed or you want to change it, click **Edit** and select the save set to use.

Option	Description
Next incremental restore	Automatically selects the save set after the last one that was applied (restored).
Latest version (Full)	Standard CBB restore: From the last full backup to the last available save set.
Specify a version (Full)	CBB restore: From the last full backup to the specified save set.

Note:

If the selected job is retired, the save set version information is not selectable.

5. Select the **Destination System** and enter the **Credentials** associated with it.

Note:

The password is applied to all jobs associated with this system and user account.

6. Select the **Target Group**.
7. [Optional] Add a job description.
8. To change the objects selected, clear the checkboxes and select others in the **Objects** tab.
9. [Optional] Select **Selection Summary** to verify the selected objects..

Note:

Size information is not available for DBC only backup jobs. N/A displays as the size value for DBC only backup jobs.

10. [Optional] To adjust job settings for the job, select the **Job Settings** tab.

    Settings can include specifying whether a job continues or aborts if an access rights violation is encountered on an object.

Note:

- [Teradata Database 16.20 Update 1 and later] Fallback cannot be disabled. Checkbox is ignored.
- [Prior versions of Teradata Database] Disable fallback is not available unless **Run as copy** is checked. The icon ⃠ appears when the mouse pointer is hovered over the checkbox.

11. Click **Save**.

Run the Job

12. Click next to a job.
13. Select **Run**.
14. [Optional] Select **Allow objects for write after restore**.
    Because the primary use case for incremental restore is to keep systems in sync, by default objects are restored in a read-only state. If you check the box and restore objects in a read-write state, you cannot run an incremental restore to the target system until after a full restore is done.

Related Information:

Changing Job Permissions
Job Settings Tab

Creating or Updating a Job: Command Line

The `create_job` command creates a job based on the values you specify for parameters in the command line or in the XML file. Parameter values you enter in the command line supersede any value you enter for those parameters in the parameters XML file.

You can also edit an existing job by using the `update_job` command.

Before you create a restore or analyze type job, a backup job must have been completed successfully or run with a warning. To create an incremental restore job, the backup job must have been created with incremental restore enabled.

1. Type `dsc create_job` or `dsc update_job` followed by the parameters, and press **Enter**. See create_job or update_job for syntax and xml tags, including `enable_incremental_restore`, `backup_version` and `next_incremental_restore`.
2. Type the user name and password, and press **Enter**.

Creating an Incremental Restore with DBC Job

1. Type dsc create_job followed by the parameters and press Enter. See create_job for syntax and xml tags, including include_DBC_data.
2. Type the username and password, and press Enter.

New or Updated Command Line Commands for Incremental Restore

These commands are either new, contain new parameters, or contain new status or information fields for incremental restore:

- create_job
- run_job
- job_status
- list_jobs
- list_save_sets
- list_components
- update_job
- build_restore_objects
- list_build_restore_objects
- reset_ir_target_system
- list_reset_ir_target_system
- cleanup_ir_source_system
- list_cleanup_ir_source_system

BAR Setup Portlet: IR Support

The **BAR Setup** portlet has a new section for IR support. Refer to these definitions if you need to use these options.

From the BAR Setup portlet, select your DSC server, **Systems and Nodes**, your system, then **IR Support**.

Option	Description
Cleanup Logging Level	Level of BAR log information to display for cleanup. Extensive logging information is typically only useful for support personnel when gathering information about a reported problem. **Error** Default. Enables minimal logging. Provides only error messages. **Warning** Adds warning messages to error message logging. **Info** Adds informational messages to warning and error message logging. **Debug** Full logging. All messages, including debug, are sent to the job log. **NOTICE** This setting can affect performance.
Cleanup Time Out	Time out value in minutes. Ends the current request as Failed, allowing a new request to be submitted. Default: 720
Cleanup the source system for incremental restore	Select **Cleanup** to delete the deleted rows logging subtables on the system after the system has been disabled for incremental restore.
Get Cleanup status	Provides status on the current cleanup process.
Reset Logging Level	Level of BAR log information to display for reset. Extensive logging information is typically only useful for support personnel when gathering information about a reported problem. **Error** Default. Enables minimal logging. Provides only error messages. **Warning** Adds warning messages to error message logging. **Info** Adds informational messages to warning and error message logging. **Debug** Full logging. All messages, including debug, are sent to the job log.

Option	Description
	NOTICE This setting can affect performance.
Reset Time Out	Time out value in minutes. Ends the current request as Failed, allowing a new request to be submitted. Default: 720
Reset the target system for incremental restore	Resets the target system to allow incremental restore again. **Important:** All existing incremental restore jobs are forced to run as full restore for their next run.
Get Reset status	Provides progress on the reset to incremental restore.

Teradata BAR Portlets

BAR Setup

Use the **BAR Setup** portlet to designate the hardware and software to use when backing up your database. Use this portlet to configure the following:

- DSCs
- Systems
- Media servers
- Fabrics
- Backup solutions; such as, Disk File System (including Spectrum Protect), NetBackup, DD Boost, Amazon S3, Azure Blob
- Hardware and software groups to use as targets for backup and restore operations
- Logical mappings between different target groups for restoring to different client configurations
- Automatic backup schedule for the DSC repository
- Custom alerts (the **Alert Setup** portlet is used in addition to the **BAR Setup** portlet to configure the alerts)

Beginning with DSA 16.10, nodes are configured through autodiscovery. You can view the node information but changing the node configuration is not allowed.

After the configuration is complete, the **BAR Setup** portlet employs the DSC Repository to save all of your configuration settings. These configured systems, media servers, backup solutions, and target groups are available for use in backup, restore, and analyze jobs. You can view the alerts that you configure in the **BAR Setup** and **Alert Setup** portlets through text, email, and the **Alert Viewer** portlet.

Configuring BAR Setup

Prerequisite:

Before you can work with the DSC in the BAR Setup portlet, you must add and enable the Teradata System and the Dictionary collector in the **Monitored Systems** portlet. Under **Setup** select **Data Collectors** and enable the **Dictionary** collector.

This task outlines configuration tasks involved in the **BAR Setup** portlet to make systems, media servers, backup applications, and target groups available in the **BAR Operations** portlet.

1. [Optional] Enabling or Adding a DSC Server
2. Adding or Editing a System and Node Configuration

3. Verifying the Media Server
   A media server must be defined so it can be made available for target groups.
4. [Optional] Configuring Network Fabric: Portlets
5. Add and configure a backup solution.
   - Adding or Editing a Disk File System, Including Spectrum Protect
   - Adding a NetBackup Server
   - Adding a DD Boost Server
   - Adding an Amazon S3 Account
   - Adding Azure Blob Storage
6. Adding or Copying a Target Group
   In order for data to be backed up to a device, a target group must be created to configure media servers to the backup application.
7. Scheduling Automatic Repository Backups
   Describes how to schedule a backup of the DSC repository.
8. DSC Servers
   Allows you to manage the DSC Servers, including editing settings and changing the DSC server name..

DSC Servers

The configuration of DSCs is based on the connection parameters (broker IP, broker port, and connection type) and DSC server name.

Enabling or Adding a DSC Server

Use these instructions to enable or add an additional DSC. If you have a DSC and are upgrading or adding another DSC, use this procedure before upgrading.

1. Open the **BAR Setup** portlet.
2. Click + next to **DSC Servers**.
3. Under **General System Details**, enter the broker information:

Option	Description
Broker IP/Host	Hostname or IP address of the machine running the ActiveMQ broker (where tdactivemq is installed), usually the DSC server.
Broker Port	Port number on the server where the ActiveMQ broker is listening (61616 for tcp, 61617 for ssl). Default: 61616
Broker Connectivity	Type of ActiveMQ connection (tcp or ssl). If tcp is chosen, the ActiveMQ connection is validated during install. If ssl is chosen, the ActiveMQ (jms ssl) password should match the DSA REST API https password. Default: tcp

4. Select **Enable DSC server**.
5. Select the DSC Server.

a. Click **Discover Servers**.
b. Select the **DSC Server Name** from the drop-down.

Troubleshooting:

If the DSC names are not populated in the menu, check the following:

- Verify that the Broker IP/Host is correct.
- Make sure ActiveMQ is running:

```
/etc/init.d/tdactivemq status
```

---

6. Enter the **Server Settings** and **BAR Logging** settings:

Option	Description
Security Management	Require Teradata Viewpoint authentication on the DSA command-line interface. If checked, a user submitting certain commands from the command-line interface must enter a valid Teradata Viewpoint user name and password.
BAR Logging	Level of BAR log information to display for the **Data Stream Controller** and the **BAR Network Client**. Extensive logging information is typically only useful for support personnel when gathering information about a reported problem. **Error** Default. Enables minimal logging. Provides only error messages. **Warning** Adds warning messages to error message logging. **Info** Adds informational messages to warning and error message logging. **Debug** Full logging. All messages, including debug, are sent to the job log. **NOTICE** This setting can affect performance.
Delete Retired Jobs	**Delete Retired Jobs** determines the time period before retired jobs are deleted. **After** Number of days from the date a job is retired to wait before deleting the job. **Never** Prevents deletion of retired jobs.

7. Click **Apply**.

Editing DSC Server Settings

1. Open the **BAR Setup** portlet.
2. Under **DSC Servers**, select your DSC server.
3. From the **Categories** list, click **General**.
4. Under **General System Details**, edit or view the following:

Setting	Description
DSC Server Name	View only. To change the DSC Server Name, see Changing the DSC Server Name.
Enable DSC server	Check or uncheck as needed.
Broker IP/Host	View only. **Important:** To edit this value, remove and then re-add the DSC Server. Do not edit the value here. See Removing a DSC Server and Enabling or Adding a DSC Server.
Broker Port	View only. **Important:** To edit this value, remove and then re-add the DSC Server. Do not edit the value here. See Removing a DSC Server and Enabling or Adding a DSC Server.
Broker Connectivity	View only. To switch between TCP and SSL, see Toggling SSL or TCP after Installation.
Security Management	Require Teradata Viewpoint authentication on the DSA command-line interface. If checked, a user submitting certain commands from the command-line interface must enter a valid Teradata Viewpoint user name and password.
BAR Logging	Level of BAR log information to display for the **Data Stream Controller** and the **BAR Network Client**. Extensive logging information is typically only useful for support personnel when gathering information about a reported problem. **Error** Default. Enables minimal logging. Provides only error messages. **Warning** Adds warning messages to error message logging. **Info** Adds informational messages to warning and error message logging. **Debug** Full logging. All messages, including debug, are sent to the job log.

Setting	Description
	NOTICE This setting can affect performance.
Delete Retired Jobs	**Delete Retired Jobs** determines the time period before retired jobs are deleted. **After** Number of days from the date a job is retired to wait before deleting the job. **Never** Prevents deletion of retired jobs.

Changing the DSC Server Name

1. Remove the DSC Server from the **BAR Setup** portlet (see Removing a DSC Server).
2. Run `$DSA_DSC_ROOT/modify_dsc_name.sh`.
   Enter the new DSC server name when prompted, maximum of 128 characters: alphanumeric, "-", and ".".
3. Edit `dsc.name` in the `$DSA_CONFIG_DIR/commandline.properties` file.
4. Add the new DSC Server in the **BAR Setup** portlet (see Enabling or Adding a DSC Server).

Removing a DSC Server

Use this procedure to remove a server from the BAR Setup portlet. For example, if you want to change between TCP and SSL, you need to remove the server and then re-add it (see Enabling or Adding a DSC Server).

1. Open the **BAR Setup** portlet.
2. Under **DSC Servers**, select your DSC server.
3. From the **Categories** list, click **General**.
4. Click **Remove DSC Server**.
5. Click **Remove**.

Systems and Nodes

You can add, configure, and set stream limits for systems in the **BAR Setup** portlet and by using DSA setup commands from the DSA command-line interface. After you enable configured systems, they are available for backup and restore jobs in the **BAR Operations** portlet and for DSA operation commands.

Beginning with DSA 16.10, nodes are configured through autodiscovery. You can view the node information but changing the node configuration is not allowed.

Adding or Editing a Teradata System

Prerequisite:

Before you can work with the DSC in the BAR Setup portlet, you must add and enable the Teradata System and the Dictionary collector in the **Monitored Systems** portlet. Under **Setup** select **Data Collectors** and enable the **Dictionary** collector.

If you are upgrading to Analytics Database 16.20 or later or Teradata Database 16.0 or later, run DIPBAR on the Teradata system before this procedure.

Follow these steps to configure and enable the target database system.

Note:

- Nodes are configured through autodiscovery. You can view but not edit them.
- [DSA 17.02 and later] There is no longer a preconfigured backup system, you just need to enable the Teradata system.

1. Open the **BAR Setup** portlet.
2. Under **DSC Servers**, select your DSC server.
3. From the **Categories** list, select **Systems and Nodes**.
4. Do one of the following:

Add a new target system	a. Click + next to **Systems**. b. Select **Add Teradata System**.
Edit an existing system	Select the name under **Systems**.

5. Under **Setup**, select **System Details** and enter the following:

Option	Description
System Name	[Adding a new system] Choose the system from the drop-down list. **Note:** Make a system available from the **Monitored Systems** portlet.
System Selector	[Optional] When editing a system, to change the system selector, click **Update**. The credentials to the system are verified before the update can occur. **Note:** You must stop and start DSMain in the database after changing the system selector.

Option	Description
SSL Communication	[Optional] Select the **Enable SSL over JMS Communication** checkbox to enable SSL communication. **Note:** You must add the TrustStore password created during SSL setup. You must stop and start DSMain in the database after enabling SSL communication.
Default Stream Limits For Nodes	Set the default limits for each node configured with the system. • **For each node:** maximum number of concurrent streams allowed per node. For example, 5 times the number of AMPs on the node. • **For each job on a node:** maximum number of concurrent streams allowed for each job on the node. Cannot exceed the number of AMPs on the node. If you enter a higher number, it is reduced to the number of AMPs on the node.

6. Click **Apply** and enter the database credentials.
7. Under **Setup** select **Nodes** to view the details for the nodes on this system.
8. Restart DSMain on the destination system:
   a. From the primary distribution node (usually Node 1), run `cnsterm 6`.
   b. Enter `start bardsmain –s -d` *dsc_name* (this stops DSMain on the destination system).

      Note:

      The `-d` *dsc_name* parameter applies to Analytics Database 16.20 or later / Teradata Database 16.10 or later. The -d parameter only starts and stops the ActiveMQ connection. It will not affect any feature that requires DSMain to be completed restarted; for example GDO flag 553 (EnableBackupsforIncrementalRestore).

   c. Enter `start bardsmain` (this starts DSMain).
   d. [Analytics Database 16.20 or later / Teradata Database 16.10 or later] Enter `start bardsmain -j` (this shows the status of the connections).

Note:

The system is automatically enabled.

Deleting a System

Use the following steps to delete a system from the **BAR Setup** portlet, which removes it as a source for restore or backup jobs from the **BAR Operations** portlet.

Note:

You cannot delete a system if it is in use by a job or the system is marked for repository backup.

1. From the **Categories** list, click **Systems and Nodes**.
2. From the **Systems** list, click next to the system to be deleted.
   A confirmation message appears.
3. Click **OK**.

Media Servers

Media servers manage data during system backups and restores. Media servers are made available to your BAR environment as soon as the DSA software is installed and running. Use the **BAR Setup** portlet to verify or delete media servers and assign media servers to target group configurations.

Verifying the Media Server

Media server data is autopopulated. You can view and edit if necessary using the **BAR Setup** portlet.

1. Open the **BAR Setup** portlet.
2. Under **DSC Servers**, select your DSC server.
3. From the **Categories** list, select **Media Servers**.
4. For each media server, verify the following:

Option	Description
BAR NC Port	Verify that the number of the BAR network server matches the server port setting in the DSA client handler property file. Default port: 15401 **Note:** If you change the port number in the `clienthandler.properties` file, you must restart the DSA Network Client using `/etc/init.d/clienthandler restart-hwupgrade`.
IP Address	This is the address of the media server. Do not use a link-local IPv6 address (begins with `fe80`). Additional addresses can be entered for network cards that are attached to the server. If there are multiple instances of DSA Network Client (ClientHandler), specify separate IP addresses. For example, configure the first media server configuration with the first IP address and the second media server configuration with the second IP address. **Note:** IP addresses are not validated. Teradata recommends verifying that you can ping from the media server to the database and from the database to the media server.
Network Mask	Refer to Network Masks for more information.

Deleting a Media Server

Deleting a media server from the **BAR Setup** portlet makes it unavailable for target groups in the **BAR Operations** portlet. If ClientHandler is restarted on the media server, the media server will be repopulated. To permanently delete the media server, remove ClientHandler from it.

Note:

You cannot delete a media server that is currently configured to a target group.

---

1. From the **Categories** list, click **Media Servers**.
2. Click 🗑 next to the media server you want to delete.
3. Click **OK**.

Network Masks

The subnet mask used by DSA is a logical mask, that is, it is treated as a mask to determine what connection paths are allowed between Teradata nodes and media servers. You can use a DSA network mask to create a data path between a Teradata node and a media server if the Teradata node and media server are physically connected. The DSA network mask setting does not override any physical subnet mask.

Use the default network mask, populated by DSA, that is based on the data path between Teradata nodes and media servers.

Note:

Remove network interfaces not used in the data path from the media server definition in the **BAR Setup** portlet.

Guidelines for DSA Network Masks

- Teradata nodes and media servers should be on the same logical subnet.
- If Teradata nodes and media servers are on different logical subnets, but can communicate with each other, open up the network mask as relevant.

Fabrics

Beginning with DSA 16.10.01, you can define network fabric. If a network fabric is defined, it replaces a logical network mask when a job plan generates data paths between AMPs and BARNC processes acting as media servers.

- The fabric can include all the nodes and their paths to one or more media servers.
- Fabric is used only for backup, restore, and analyze_validate jobs.

- You can define multiple fabrics but each must be unique. For example, System A can have two fabrics for two target systems B and C, fabricAB and fabricAC. The information is saved into the repository and can be edited if new media servers or nodes are added.
  - All media servers in a target group should be available in a single fabric.
  - The correct fabric is chosen based on the media servers in the target group.
  - Update `dsc.properties` for the number of fabrics allowed per Teradata system: `maxlimit.fabric=`*number-fabrics*. Default is 64.
- When fabrics are used, Teradata recommends not editing media server IP addresses.

Related Information:

Define Data Paths Using Network Fabric
Network Fabric Considerations

Configuring Network Fabric: Portlets

Beginning with DSA 17.06, you can configure multiple fabrics.

Update `dsc.properties` for the number of fabrics allowed per Teradata system: `maxlimit.fabric=`*number-fabrics*. Default is 64.

1. Restart ClientHandler to discover all changed IP addresses.
   `/etc/init.d/clienthandler restart-hwupgrade`
2. Open the **BAR Setup** portlet.
3. Under **DSC Servers**, select your DSC server.
4. From the **Categories** list, click **Fabrics**.
5. Click + next to **Fabrics**.
6. Under **Fabric Details**, enter the following:

Option	Description
Fabric Name	Alphanumeric characters plus "-" and "_".
System Name	Choose the system from the drop-down list.

7. Under **Nodes**, associate the media servers with the nodes:

Node	Choose the node from the drop-down list. Click + to add another.
Media server	Select a media server. Click + to add another.
IP Address	Select an IP address. Click + to add another.

Note:

For standby nodes, you can reduce maintenance by setting `dsc.properties` to `validate.fabric=off`. Restart the DSC after changing the properties file. The undefined paths use bynet to move the data. This solution degrades performance.

8. Select the **Enable fabric** checkbox.
9. Click **Apply**.

Deleting a Network Fabric

Deleting a network fabric re-enables netmask during job plan creation.

1. Under **DSC Servers**, select your DSC server.
2. From the **Categories** list, click **Fabrics**.
3. Under **Fabrics**, click next to the fabric name.
   A confirmation message appears.
4. Click **OK**.

Backup Solutions

Backup solutions are options for transferring data between a storage device and a database system. Available solutions include third-party server software and cloud-based solutions as well as local storage. Configuration options depend on the backup solution. Choose Disk File System for Spectrum Protect.

Adding or Editing a Disk File System, Including Spectrum Protect

When using a disk file system to back up and restore data, you must add and configure the disk file system using the **BAR Setup** portlet.

Note:

System names and open file limits are tied to media servers during the target group configuration.

1. Open the **BAR Setup** portlet.
2. Under **DSC Servers**, select your DSC server.
3. Under **Categories**, select **Backup Solutions**.
4. Under **Solutions**, select **Disk File System**.
5. Verify or add a disk file system:
   a. [Optional] To add a disk file system, from the **Disk File System Details** screen, click .
   b. Enter a **File system name** and path that meets the following criteria:
      - Entire file system path has necessary data write permissions by the DSCuser

- Unique, fully qualified path name that begin with a forward slash, for example, `/storage/mnt1/`
- Does not differ by case alone. For example, both `/storage/mnt1/` and `/storage/Mnt1/` cannot be configured.
- Contains no spaces.
- [Spectrum Protect] **File system name** must start with /Spectrum followed by the node name and management class for the node name; for example, `/Spectrum/`*NodeName*`/`*ManagementClass*

Note:

The disk file system filepath used by the replication target group cannot be used by the operational target group, and vice versa.

---

c. Enter the maximum number of open files allowed.

6. To edit an existing system, change the number of **Max open files** next to its name.

Deleting a Disk File System

Deleting a disk file system disassociates the server and its settings from the **BAR Setup** portlet. The server can no longer be used as a target for backups in the **BAR Operations** portlet.

Note:

If a disk file system is currently configured to a target group, it cannot be deleted.

---

1. Under **DSC Servers**, select your DSC server.
2. Under **Categories**, select **Backup Solutions**.
3. Under **Solutions**, select **Disk File System**.
4. Under **Disk File System Details**, click − next to the server you want to delete.
   A confirmation message appears.
5. Click **OK**.

Adding a NetBackup Server

When using a NetBackup server to back up and restore data, you must add and configure the NetBackup server using the **BAR Setup** portlet.

Note:

NetBackup policies are tied to media servers during the target group configuration. It is important that the policies entered for your NetBackup configuration coincide with the policies intended for the media server configuration mapped as a target. See config_nbu.

---

1. Under **DSC Servers**, select your DSC server.
2. Under **Categories**, select **Backup Solutions**.
3. From the **Solutions** list, click **NetBackup**.
4. Click + next to **Servers**.
5. Under **NetBackup Details**, enter the following:

Option	Description
Nickname	Alphanumeric characters and underscores, but no spaces.
Server Name (IP/DNS)	Server IP address or DNS

6. Under **Media Server and Policies**, select the **Media Server**.
   You can only select media servers with AXMNetBackup installed.
7. If the **Discover Policies** button is disabled, click **Apply** to enable it.
8. Click **Discover Policies**.
9. Select a **Policy Name** and enter the number of **Storage Devices**.
10. Click + to add additional policies.
11. Click **Apply**.

Deleting a NetBackup Server

Deleting a NetBackup server disassociates the server and its settings from the **BAR Setup** portlet. The server can no longer be used as a target for backups in the **BAR Operations** portlet.

Note:

If a NetBackup server is currently configured to a target group, it cannot be deleted.

---

1. Under **DSC Servers**, select your DSC server.
2. From the **Categories** list, click **Backup Solutions**.
3. Under **Solutions**, click **NetBackup**.
4. Under **Servers**, click 🗑 next to the server you want to delete.
   A confirmation message appears.
5. Click **OK**.

Adding a DD Boost Server

When using a DD Boost server to back up and restore data, you must add and configure the DD Boost server using the **BAR Setup** portlet.

Note:

DD Boost storage units are tied to media servers during the target group configuration. It is important that the storage units entered for your configuration coincide with the storage units intended for the media server and device configuration mapped as a target.

1. Under **DSC Servers**, select your DSC server.
2. Under **Categories**, select **Backup Solutions**.
3. From the **Solutions** list, click **DD Boost**.
4. Click + next to **Servers**.
5. Under **DD Boost Details**, enter the following:

Option	Description
Nickname	Alphanumeric characters and underscores, but no spaces
Server Name (IP/DNS)	Server IP address or DNS
User / Password	Data domain DD Boost credentials
Storage unit name	Name must match a data domain storage unit intended for the media server and device configuration mapped as a target. **Note:** DSC does not support the same storage unit name across different DD Boost servers.
Max number of open files	Maximum number of open files allowed

6. Click + after **Max number of open files** to add more storage units.
7. Click **Apply**.

Deleting a DD Boost Server

Deleting a DD Boost server disassociates the server and its settings from the **BAR Setup** portlet. The server can no longer be used as a target for backups in the **BAR Operations** portlet.

Note:

If a DD Boost server is currently configured to a target group, it cannot be deleted.

1. Under **DSC Servers**, select your DSC server.
2. Under **Categories**, select **Backup Solutions**.
3. From the **Solutions** list, click **DD Boost**.
4. Under **Servers**, click 🗑 next to the server you want to delete.
   A confirmation message appears.
5. Click **OK**.

Adding an Amazon S3 Account

When using Amazon S3 storage to back up and restore data, you must add and configure the S3 account using the **BAR Setup** portlet.

1. Under **DSC Servers**, select your DSC server.
2. Under **Categories**, select **Backup Solutions**.
3. From the **Solutions** list, click **Amazon S3**.
4. Click + next to **Accounts**.
5. Under **Amazon S3 Storage Details**, enter the **Account Name**.
   Account name is alphanumeric, maximum of 32 characters.
6. Select an access type and enter its values:

Access Type	Values
Key authentication	Enter these items as they are configured on AWS. **Account Id** AWS account ID **Account Key** IAM user access key **Region** Region associated with this bucket; for example: us-east-1 Regions are validated but you can override if a new region becomes available. **Bucket** S3 bucket name **Prefix** Alphanumeric, followed by / to be used as a folder **Storage Units** Maximum of 3 characters, numeric range between 1-999 You can enter multiple regions and/or buckets by selecting +.
IAM Role	Enter these items as they are configured on AWS. **Role Name** As established on AWS When roles are used, all components must be in the cloud and assigned to this role. **Region** Region associated with this bucket; for example: us-east-1 Regions are validated but you can override if a new region becomes available.

Access Type	Values
	Bucket S3 bucket name **Prefix** Alphanumeric, followed by / to be used as a folder **Storage Units** Maximum of 3 characters, numeric range between 1-999 You can enter multiple regions and/or buckets by selecting +.
Snowball	Enter these items as they are configured on AWS. **Account Id** AWS account ID **Account Key** IAM user access key **Network IP** Local IP address for the Snowball device **Region** Data target region, assigned with the Snowball **Note:** Snowball cannot be associated with multiple regions. **Bucket** S3 bucket name **Prefix** Alphanumeric, followed by / to be used as a folder **Storage Units** Maximum of 3 characters, numeric range between 1-999 You can enter multiple buckets by selecting +.

7. Click **Apply**.

Deleting an Amazon S3 Account

Deleting an Amazon S3 account disassociates the account and its settings from the **BAR Setup** portlet. The account can no longer be used as a target for backups in the **BAR Operations** portlet.

Note:

If an S3 account is currently configured to a target group, it cannot be deleted.

---

1. Under **DSC Servers**, select your DSC server.
2. Under **Categories**, select **Backup Solutions**.
3. From the **Solutions** list, click **Amazon S3**.
4. Under **Accounts**, click next to the account you want to delete.
   A confirmation message appears.
5. Click **OK**.

Adding Azure Blob Storage

When using Azure Blob storage to back up and restore data, you must add and configure the Azure Blob account using the **BAR Setup** portlet.

For optimal performance, configure 4 Blob containers per media server with 6 storage units each.

1. Under **DSC Servers**, select your DSC server.
2. Under **Categories**, select **Backup Solutions**.
3. From the **Solutions** list, click **Azure Blob Storage**.
4. Click next to **Accounts**.
5. From the **Azure Blob Storage Details** screen, configure the following:

Option	Description
Storage Account	Storage account name from Azure
Account Key	Account Key from Azure
Blob Type	Cool or Hot. Default: Cool
Specialized Endpoint	Leave blank. This field is only for National Clouds, which have sovereign endpoints. If you are in one of these national clouds, set the specialized endpoint entry as shown (the leading '.' is required). **Azure API Endpoint** / **Region Entry** Azure US Government: `.blob.core.usgovcloudapi.net` Azure Germany: `.blob.core.cloudapi.de` Azure China: `.blob.core.chinacloudapi.cn` For additional sovereign endpoints, check the Blob storage documentation for your national cloud.
Blob Container	Container name from Azure. The first 3 characters must be unique for each container.
Prefix	Alphanumeric, followed by / to be used as a folder
Storage Units	Maximum number of files you can write. Maximum of 3 numbers, range between 1-999

6. Click **Apply**.

Deleting Azure Blob Storage

Deleting an Azure Blob storage account disassociates the account and its settings from the **BAR Setup** portlet. The account can no longer be used as a target for backups in the **BAR Operations** portlet.

Note:

If an Azure Blob storage account is currently configured to a target group, it cannot be deleted.

---

1. Under **DSC Servers**, select your DSC server.
2. Under **Categories**, select **Backup Solutions**.
3. From the **Solutions** list, click **Azure Blob Storage**.
4. Under **Accounts**, click next to the account you want to delete.
   A confirmation message appears.
5. Click **OK**.

Adding a Google Cloud Account

When using Google Cloud to back up and restore data, you must add and configure the Google Cloud account using the **BAR Setup** portlet.

1. Under **DSC Servers**, select your DSC server.
2. Under **Categories**, select **Backup Solutions**.
3. From the **Solutions** list, select **Google Cloud**.
4. Click next to **Accounts**.
5. From the **Google Cloud Buckets Details** screen, configure the following:

Option	Description
Account Name	Storage account name from Google
Access Key	Access Key from Google
Bucket	Bucket name from Google.
Location	Select the appropriate region.
Folder	Alphanumeric, followed by / to be used as a folder
Devices	Number of storage devices: 1 to 999

6. Click **Apply**.

Deleting a Google Cloud Account

Deleting an Google Cloud account disassociates the account and its settings from the **BAR Setup** portlet. The account can no longer be used as a target for backups in the **BAR Operations** portlet.

Note:

If Google Cloud account is currently configured to a target group, it cannot be deleted.

---

1. Under **DSC Servers**, select your DSC server.
2. Under **Categories**, select **Backup Solutions**.
3. From the **Solutions** list, click **Google Cloud**.
4. Under **Accounts**, click 🗑 next to the account you want to delete.
   A confirmation message appears.
5. Click **OK**.

Target Groups

Target groups are locations used for storing backup data. DSA administrators create target groups, and assign media servers and devices. Target groups are then accessible to BAR backup jobs.

You need at least two target groups: a replication target group for the repository backup (DSA configuration and metadata) and one or more for data backups.

After a backup job has run to completion, you can create a BAR restore job to restore data using the same target group as the backup job. You can also create a target group map (restore group), which allows a BAR restore job to restore data from a different target group.

Virtual Target Groups

The **BAR Setup** portlet does not manage virtual target groups, which are target groups imported from a different DSC. If you need to manage a virtual target group, use the command line.

Adding or Copying a Target Group

The data from Teradata systems is sent through media servers for backup by backup solutions. These relationships are defined in target groups, which you can create and copy.

When DSC is deployed in the Public Cloud, a default disk file system target group is created (`defaultDFTG`*system_name*). This target group scales out/in with the database.

See Public Cloud: Default Target Group for Elasticity Scale Out/In.

1. Open the **BAR Setup** portlet.
2. Under **DSC Servers**, select your DSC server.
3. From the **Categories** list, select **Target Groups**.

4. From the **Target Groups** list, select **Remote Groups** or **Replication Groups**.
5. Do one of the following:

Option	Description
Add	Click + next to **Remote Groups**/**Replication Groups** to add a remote group.
Copy	Click ⧉ next to the name of the group you want to copy. If you copy the target group, some items cannot be changed.

Remote Target Group

6. Enter a **Target Group Name** for the new target group.
   You can use alphanumeric characters, dashes, dots, and underscores, but no spaces.
7. Select the **Enable target group** checkbox.
8. Select a **Solution Type**.
9. In the **Targets** and the **Remote Group Details** section, make selections for the Solution Type:
   - NetBackup: Select the **Target Entity**, the **BAR Media Server**, the **Policies** and the number of **Devices** for each server pair.

     You also can select **Reject backup job if any media servers are offline**.
   - DD Boost: Select the **Target Entity**, the **BAR Media Server**, the **Storage Unit** and set the **Open Files** limit.

     You also can select **Reject backup job if any media servers are offline**.
   - Disk File System: Select the **Bar Media Server**, the **Disk File System** and the **Open Files** limit.

     Use Disk File System for IBM Spectrum Protect.
   - Amazon S3: Select the **Account Name**, **Region**, **BAR Media Server**, **Bucket**, **Prefix**, and enter the number of **Storage Units**.
   - Azure Blob Storage: Select the **Storage Account**, **BAR Media Server**, **Prefix**, and set the number of **Storage Units**.

     To use multiple storage accounts, select + next to **Targets** and enter the information for another account.
   - Google Cloud: Select the **Service Account**, **BAR Media Server**, **Bucket**, **Folder**, and set the number of **Devices**.

Option	Description
Add	Click + to add; policies and devices, storage units and open files limit, or disk file systems and open files limit.
Remove	Click − to remove; policies and devices, storage units and open files limit, or disk file systems and open files limit.

10. Click **Apply**.

Replication Target Group

11. In the **Targets** and the **Replication Target Group Details** section, make selections for the Solution Type:
    - NetBackup: Select the **Target Entity**, the **BAR Media Server**, the **Policies** and the number of **Devices** for each server pair.
    - DD Boost: Select the **Target Entity**, the **BAR Media Server**, the **Storage Unit** and set the **Open Files** limit.
    - Disk File System: Select the **BAR Media Server**, the **Disk File System** and the **Open Files** limit.

      IBM Spectrum Protect cannot be used for replication.
    - Amazon S3: Select the **Account Name**, **Region**, **BAR Media Server**, **Bucket**, **Prefix**, and enter the number of **Storage Units**.
    - Azure Blob Storage: Select the **Storage Account**, **BAR Media Server**, **Prefix**, and set the number of **Storage Units**.
12. Click **Apply**.

Deleting a Target Group

Any target group except a repository target group can be deleted if it is not being used by a job in the **BAR Operations** portlet. The target in a target group cannot be deleted if a job has used the target group.

Important:

Deleting a replication target group also deletes all save set files associated with it.

---

1. Under **DSC Servers**, select your DSC server.
2. From the **Categories** list, click **Target Groups**.
3. From the **Target Groups** list, click **Remote Groups** or **Replication Groups**.
4. From the list, click 🗑 next to the name of the target group you want to delete.
   A confirmation message appears.
5. Click **OK**.

Adding or Editing a Restore Group

The device and media servers relationships defined in target groups can be selected to create target group maps called restore groups. In the CLI, this is referred to as target group mapping. If mapping is disabled, either the source or target group has been modified.

Note:

The disk file system backup solution has an autogenerated default target group. The target group mapping for that target group is automatically disabled when the system folds or unfolds.

---

1. Under **DSC Servers**, select your DSC server.
2. From the **Categories** list, click **Target Groups**.
3. From the **Target Groups** list, click **Restore Groups**.
4. Next to **Restore Groups**, do one of the following:

Option	Description
Add	Click to add a restore group.
Edit	Click in the row of the restore group you want to edit.

5. Select the **Solution type** from the list.
6. Select the **Backup Target Group** from the list.
   a. [Optional] Click next to the BAR media server associated with the backup target group to view policy and device details.
7. Select the **Restore Target Group** from the list.
   a. [Optional] Click next to the BAR media server associated with the restore target group to view policy and device details.
   b. If necessary, click the checkbox next to the policy to use.
8. Click **OK**.

Deleting a Restore Group

You can delete a restore group if it is not used by a job in the **BAR Operations** portlet.

1. Under **DSC Servers**, select your DSC server.
2. From the **Categories** list, click **Target Groups**.
3. From the **Target Groups** list, click **Restore Groups**.
4. From the **Restore Groups** list, select **> Delete** next to the name of the remote group you want to delete.
   A confirmation message appears.
5. Click **OK**.

Restoring: Same Target Group, Fewer Media Servers

Normally, backup and restore utilizes the same target group. This means that the media servers and storage devices used during the backup are identical to ones used for data restore. However, it is inevitable that media servers in the target group can be offline when a restore event occurs. In the end the media servers used for the restore need to have access to all the files generated at backup time.

Teradata DSE: Restoring to Same Target Group with Fewer Media Servers Using NetBackup

These steps use this sample scenario:

- Backup job performed using: mediaserver1, mediaserver2, mediaserver3, and mediaserver4
- Restore job using only: mediaserver4

> **NOTICE**
>
> Upon completion of the restore process and resuming normal configuration, the changes made to NetBackup and DSA must be reversed to reflect the previous configuration.

1. Make these changes for Veritas NetBackup:
   a. Open `bp.conf` on the master server for editing:
      `/usr/openv/netbackup # vi bp.conf`
   b. Enter `FORCE_RESTORE_MEDIA_SERVER =` *media_server_performing_restore* at the bottom of the file; in this example to restore using mediaserver4 only:
      ```
      FORCE_RESTORE_MEDIA_SERVER = mediaserver1 mediaserver4
      FORCE_RESTORE_MEDIA_SERVER = mediaserver2 mediaserver4
      FORCE_RESTORE_MEDIA_SERVER = mediaserver3 mediaserver4
      ```
   c. Create an empty file named: `No.Restrictions`, at this location on the master server:
      `/usr/openv/netbackup/db/altnames # touch No.Restrictions`

   Note:
   NetBackup changes are dynamic, so restarting NetBackup or its services is not required.

2. Make these changes for DSA in the **BAR Setup** portlet:
   a. Create a new target group to include mediaserver4.
   b. Create a restore group to map the old target group to the new target group.
   c. Create a restore job using the new restore target group.

Teradata DSU: Restoring to Same Target Group with Fewer Media Servers

Under DSU the following target types can be used:

- Disk file system
- DDBoost
- Amazon S3
- Azure Blob

Teradata DSU: Disk File System Targets: Restoring with Fewer Media Servers

These steps use this sample scenario:

- Backup job performed using: mediaserver1, mediaserver2, mediaserver3, and mediaserver4
- Restore job using only: mediaserver4

1. Using the **BAR Setup** portlet, create a new target group that includes mediaserver4.
2. Configure the file paths in the target group:
   - If an NFS (network file system) environment and all the media servers have access to all the paths used during the backup, then proceed to add all these paths to this new target group.
   - In the case where this media server does not have access to the same file paths, then the files will need to be moved to the path found on mediaserver4.
3. Create a restore group to map the original target group to the newly created one with mediaserver4.
4. Create a restore job with this new restore target group and desired system to finish the restore.

Teradata DSU: DDBoost, Amazon S3, or Azure Blob Targets: Restoring with Fewer Media Servers

These steps use this sample scenario:

- Backup job performed using: mediaserver1, mediaserver2, mediaserver3, and mediaserver4
- Restore job using only: mediaserver4

These steps assume that all four media servers are writing to a single storage unit.

1. Using the **BAR Setup** portlet, create a new target group that includes mediaserver4.
2. Create a restore group to map the original target group to the newly created one with mediaserver4.
3. Create a restore job with this new restore target group and desired system to finish the restore.

Restoring with a Different Target Group

In this scenario, the media servers and storage devices used during the backup are totally different from the ones used for data restore. In this situation, the user has to create a new restore group to map the backup media server and storage device to the restore media server and storage device.

Teradata DSE: Restoring with a Different Target Group

These steps use this sample scenario:

- Backup job performed using: mediaserver1, mediaserver2
- Restore job using: mediaserver3, mediaserver4

Note:

When several environments (i.e., DEV, PROD, QA, etc.) are sharing the same NetBackup environment, restore groups may not be the best solution. If the user creates network connections between the media servers and the system that it would be restored to, then this would be more effective.

NOTICE

Upon completion of the restore process and resuming normal configuration, the changes made to NetBackup and DSA must be reversed to reflect the previous configuration.

1. Make these changes for Veritas NetBackup:
   a. Open `bp.conf` on the master server for editing:
      `/usr/openv/netbackup # vi bp.conf`
   b. Enter `FORCE_RESTORE_MEDIA_SERVER =` *media_server_performing_restore* at the bottom of the file; in this example to restore using mediaserver3 and mediaserver4 only:
      ```
      FORCE_RESTORE_MEDIA_SERVER = mediaserver1 mediaserver3
      FORCE_RESTORE_MEDIA_SERVER = mediaserver2 mediaserver4
      ```
   c. Create an empty file named: `No.Restrictions`, at this location on the master server:
      `/usr/openv/netbackup/db/altnames # touch No.Restrictions`

   Note:

   All changes done in NetBackup are dynamic, so restart of NetBackup or its services is not required.

2. Make these changes for DSA in the **BAR Setup** portlet:
   a. Create a new target group to include mediaserver3 and mediaserver4.
   b. Create a Restore group to map the old target group to the new target group.
   c. Create a restore job using the new restore target group.

Teradata DSU: Restoring with a Different Target Group

These steps use this sample scenario:

- Backup job performed using: mediaserver1, mediaserver2
- Restore job using: mediaserver3, mediaserver4

1. Do one of the following:
   - NFS mount the directories where the backup files are stored.
   - Manually copy the files to directories that can be accessed by the media servers in the restore target group.
2. Using the **BAR Setup** portlet, create a new target group that includes mediaserver3 and mediaserver4.

3. Create a restore group to map the original target group to the newly created one.
4. Create a restore job with this new restore target group and desired system to finish the restore.

Restoring with Fewer File Targets

Though it is not the norm and will require custom tasks to be performed, it is possible to restore datasets using fewer file targets than were used in the backup.

These steps use this sample scenario:

- Backup job performed using one target group and four devices:

  TargetGroup1: mediaserver1 (two devices), mediaserver2 (two devices)

- Restore job using three devices:

  NewTargetGroup: mediaserver1 (two devices), mediaserver2 (one device)

Use the **BAR Setup** port to do the following:

1. Using the policies specified in the original target group, create a new target group with mediaserver1 having two devices and mediaserver2 having one device.
2. Create a restore group to map the original target group to the newly created one with mediaserver4.
3. Create a restore job with this new restore target group and desired system to finish the restore.

Note:

The restore process will take longer since it will require two passes. The first pass will be to restore the three datasets and the second pass, to restore just one dataset.

Managing the DSC Repository

DSA configuration settings and job metadata are stored in the Data Stream Controller (DSC) repository. You can automate a repository backup or initiate the backup manually.

⚠ NOTICE

Before you can recover the DSC Repository, a DSC repository backup job and an export of the repository backup configuration must have been completed successfully at least once.

The export of the repository backup configuration can only be performed using the DSA command line. See Saving the Repository Backup Configuration File for more information.

Failure to perform a successful repository backup and an export of the repository backup configuration results in an unrecoverable DSC repository in the case of a complete disaster.

A repository backup job backs up your DSC data to a replication target group. Any running DSC repository job (backup, restore, or analyze) prevents jobs from being submitted and DSA configuration settings from being changed.

Conversely, a database backup, restore, or analyze job prevents a repository backup job from running.

Configuration settings and DSC metadata can be restored to the DSC repository from a storage device.

NOTICE

Do not abort a DSC repository restore job while the job is in progress. If you abort a DSC repository restore job while the job is in progress or if the restore job fails, DSC triggers a command to restore all repository tables to their initial state, which is an empty table. The current data in the DSC repository is lost and must be recovered using disaster recovery.

Scheduling Automatic Repository Backups

You can schedule a periodic automatic backup of the DSC repository data through the **BAR Setup** portlet.

1. Under **DSC Servers**, select your DSC server.
2. From the **Categories** list, select **Repository Backup**.
3. In the **Frequency** box, enter how often the backup job will run.
4. Select the days of the week on which the backup will run.
5. Enter a **Start Time** for the backup.
6. Select a **Replication Target Group**.
7. Click **Apply**.

Commands That Run Immediately

8. On the **Backup & Replication** tab, select one of the following:
   - **Backup**: Backs up the repository to the landing zone.
   - **Backup & Replication**: Backs up the repository to the landing zone and then replicates the save set to the replication target group third party storage.
9. On the **Replication** tab, select one of the following:
   - **Landing Zone --> Replication Target (Replication To)**: Replicates an existing save set to the replication target third party storage.
   - **Replication Target --> Landing Zone (Replication From)**: Replicates a save set from the third party storage back to the landing zone.
10. Select the **Replication Target**.
11. Click **Run Job**.

Backing Up the Repository Manually (Portlets)

Prerequisite:

You must configure a target group before you can back up the repository.

You can back up the DSC repository immediately or by scheduling the backup.

Note:

Jobs cannot be submitted and DSA configuration settings cannot be changed during a repository backup.

1. Open the **BAR Setup** portlet.
2. Under **DSC Servers**, select your DSC server.
3. From the **Categories** list, click **Repository Backup**.
4. Select the **Replication Target Group**.
5. On the **Backup & Replication** tab, select one of the following:
   - **Backup**: Backs up the repository to the landing zone.
   - **Backup & Replication**: Backs up the repository to the landing zone and then replicates the save set to the replication target group third party storage.
6. Click **Run Job** to start the repository backup job.
   During the backup job, the **BAR Setup** and **BAR Operations** portlets are unavailable.

Restoring the Repository (Portlets)

Use these steps to restore a backup of DSC repository metadata.

NOTICE

If you abort a DSC repository restore job while the job is in progress or if the restore job fails, DSC repository metadata will be corrupted. DSC triggers a command to restore all repository tables to their initial state, which is an empty table.

1. Open the **BAR Setup** portlet.
2. Under **DSC Servers**, select your DSC server.
3. From the **Categories** list, click **Repository Backup**.
4. On the **Replication** tab, select one of the following:
   - **Landing Zone --> Replication Target (Replication To)**: Replicates an existing save set to the replication target third party storage.
   - **Replication Target --> Landing Zone (Replication From)**: Replicates a save set from the third party storage back to the landing zone.
5. Select the **Replication Target**.
6. Click **Run Job**.
7. If the restore job ends with a warning, follow these steps:
   a. Check the job status.
      In the following example, `2` is the bucket number and `replication-S3` is the replication target group.
      ```
      dsc save_sets_replication_status -b 2 -n replication-S3
      ```

b. If the status indicates the foreign keys were not restored, run the foreign key repair script:
`/opt/teradata/client/`*version*`/dsa/dsc/recreateFk_`*version*`.sh`
When the restore job is complete, the **BAR Setup** and **BAR Operations** portlets become available.

Alerts

Using the **BAR Setup** portlet, you can configure custom alerts that are triggered for specific events. Refer to the following topics to configure alerts:

- Configuring Repository Backup Job Alerts
- Configuring Job Status Alerts
- Configuring Media Server Alerts
- Configuring System Alerts

In the **Alert Setup** portlet, you can add alert actions so that the custom alerts send a notification, or take some other type of action, when a metric exceeds a threshold. After you add alert actions in the **Alert Setup** portlet, they appear in the **BAR Setup** portlet.

The types of alert actions you can choose include the following:

- Send alerts to the **Alert Viewer** portlet, which provides a consolidated view for all alerts configured at the enterprise level
- Send alerts through email or text notification
- Run a SQL query
- Notify SNMP system
- Enable configuring of Repository Backup Job Alerts

Configuring Repository Backup Job Alerts

1. Open the **Alert Setup** portlet.
2. From the **Setup Options** list, select **Alert Presets**.
3. From the **Preset Options** list, select **Action Sets**.
4. Click + next to **Action Sets**.
5. Configure an action set named `configRepository`.
   For more information on the **Alert Setup** portlet or **Alert Viewer** portlet, see *Teradata® Viewpoint User Guide* or refer to **Alert Setup** or **Alert Viewer** in Teradata Viewpoint help.

Configuring Job Status Alerts

You can configure multiple job status alerts using the **BAR Setup** portlet.

1. Under **DSC Servers**, select your DSC server.
2. From the **Categories** list, click **Alerts**.

3. From the **Alert Types** list, click **Job Status**.
4. In the **Alerts** list, do one of the following:
   - To add an alert, click next to **Alerts**.
   - To configure an existing alert, select the alert in the list.

     When disabled, appears next to the alert.
   - To delete an alert, click next to the alert.
5. Under **Alert Details** enter the following:
   a. If you are creating an alert, add a name to **Alert Name**.
   b. Select or clear the **Enable alert** checkbox.
   c. From the **Severity** list, select an alert severity.
6. Under **Alert Rules** create an alert equation:
   a. Next to **Job name**, select a condition (for example, is equal to) from the menu and enter the job name.
   b. From the **Job status is** list select a job status, such as aborting.
7. Under **Alert Action** configure the alert action:
   a. From the **Action** list, select an action.
   To appear in the **Action** list, you must activate the action using the **Alert Setup** portlet.
   b. Specify a time in the **Do not run twice in __ minutes** box.
   c. In the **Message** box, type the message to be sent when the alert criteria are met.
8. Click **Apply**.

Configuring Media Server Alerts

You can configure a media server alert using the **BAR Setup** portlet.

1. Under **DSC Servers**, select your DSC server.
2. From the **Categories** list, click **Alerts**.
3. From the **Alert Types** list, click **Media Server**.
4. From the **Media Servers** list, select a media server.
   When disabled, appears next to the alert.
5. [Optional] Click to copy alert settings from one or more media servers:
   a. In the **Copy Alerts Settings** dialog box, select **All media servers** or specific media servers from the **Copy Setting To** list.
   b. Click **OK**.
6. Under **Alert Details** configure the alert:
   a. Check or clear the **Enable alert** checkbox.
   b. Select a severity level from the **Severity** list.
7. Select **Available** or **Unavailable** for **Media Server Consumers**.
8. Under **Alert Action** configure the alert:
   a. From the **Action** list, select an action.

To appear in the **Action** list, you must activate the action using the **Alert Setup** portlet.

b. Specify a time in the **Do not run twice in __ minutes** box.

c. In the **Message** box, type the message to be sent when the alert criteria are met.

9. Click **Apply**.

Configuring System Alerts

You can configure a system alert using the **BAR Setup** portlet.

1. Under **DSC Servers**, select your DSC server.
2. From the **Categories** list, click **Alerts**.
3. From the **Alert Types** list, click **System**.
4. From the **Systems** list, select a system.
   When disabled, appears next to the system.
   The **Alert Details** list displays details for the system.
5. Under **Alert Details** configure the alert:
   a. Check or clear the **Enable alert** checkbox.
   b. Select a severity level from the **Severity** list.
6. Under **Alert Rules** create an alert equation:
   a. From the **Alert when matching** list, select a condition.
   b. Next to **System Status Is**, select a status.
   c. Next to **System Consumers Are**, select a status.
   d. To add another **Condition**, click .
7. Under **Alert Action** configure the alert action:
   a. From the **Action** list, select an action.
   To appear in the **Action** list, you must activate the action using the **Alert Setup** portlet.
   b. Specify a time in the **Do not run twice in __ minutes** box.
   c. In the **Message** box, type the message to be sent when the alert criteria are met.
8. Click **Apply**.

BAR Operations

The **BAR Operations** portlet allows you to manage the following functions:

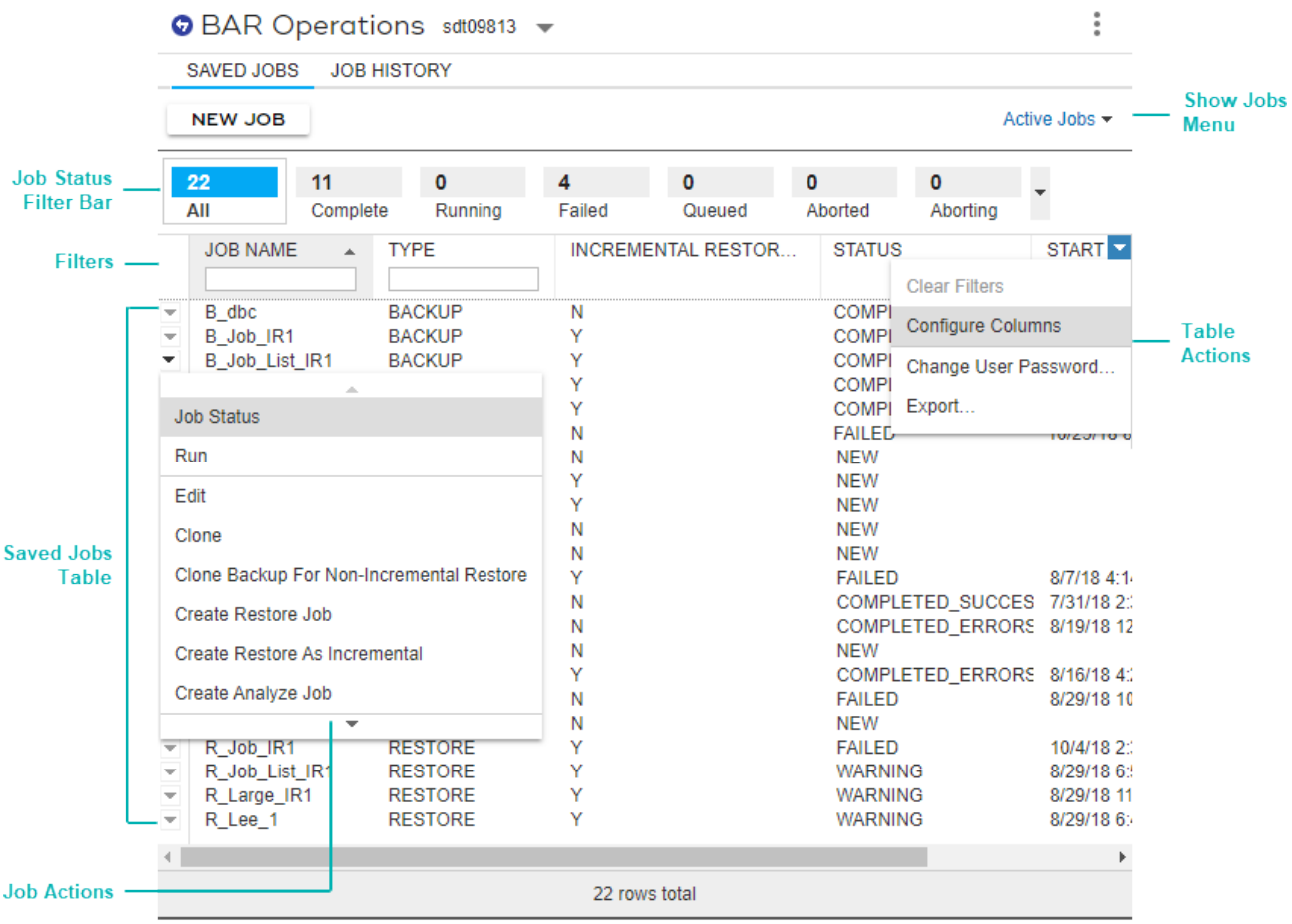
- Creating, managing, and submitting jobs
- Viewing job status and history
- Changing the system credentials for all jobs associated with a system.

Job types include backup, restore, analyze_read, and analyze_validate.

Saved Jobs View

The **Saved Jobs** view displays a table of Active, Retired, or Repository jobs, allows you to view the job status and job actions available for each job, and enables you to create a new job. Job types include backup, restore and analyze.

> **NOTICE**
>
> Repository jobs are only visible to users with BAR administrator privileges.



Show Jobs Menu

Filters the **Saved Jobs** view for Active, Retired, or Repository jobs. Active jobs are ready to be run. Retired jobs cannot be run. Repository jobs are specific to a DSC repository.

New Job Button

Creates a backup, restore, or analyze job. Only available when the **Show Jobs Menu** is showing **Active Jobs**.

Job Actions Menu

Actions you can perform on the selected job; for example, **Run** or **Job Status**. The available actions depend on the selected job. For example, **Restart** is available for restore jobs that were not successful, and **Create Restore As Incremental** is available for an IR-enabled backup job.

Action	Job Type
Restart	Unsuccessful restore job
Create Restore As Incremental	IR-enabled backup job
Clone As Non-Incremental Restore	IR-enabled restore job
Build Restore Objects	IR-enabled restore job
List Build Restore Objects	IR-enabled restore job
Clone Backup For Non-Incremental Restore	IR-enabled backup job
Clone Backup for Incremental Restore	non-IR-enabled backup job
Clone As Incremental Restore	non-IR-enabled restore job
Create Analyze Job	Backup

Job Status Filter Bar

Shows a count of the jobs by status and allows you to filter the **Job Table**. Only available when the **Show Jobs Menu** is showing **Active Jobs**.

Overflow Menu

Shows a list of job statuses. You can select another job status to replace a status on the **Job Status Filter Bar**.

Filters

Shows column filters and job information. Use **Configure Columns** from the Table Actions menu to select the visible columns; for example, **Incremental Restore Supported**, which tells you whether a job was IR enabled.

If you enter information in a filter, the table shows only the rows that match the filter.

Saved Jobs Table

Lists the job name, type, status, start time, end time, size and elapsed time of the job.

Table Actions

Configure Columns allows you to select, lock, and order the displayed columns.
Export creates a .csv file containing all available data.

Change User Password allows you to change the password used to run all BAR jobs associated with a system and user account.

Job Status Filter Bar

The job status filter bar allows you to filter on a specific job status in the **Saved Jobs** view.

The job status filter bar buttons provide a count of job runs for each status category. Click any button to filter for the selected job status or select a job status from the list. For example, click **Complete** to display all jobs that have run to completion.

You can select a job status from the **Overflow Menu** to replace a job status currently showing on the Job Status filter bar. Some examples of job statuses are:

Queued
Jobs that are waiting for resources to become available before starting

Aborted
Jobs that were been stopped by a user prior to completion

Aborting
Jobs that are in the process of being stopped by a user prior to completion

Warning
Jobs which ran to completion, but received warning messages regarding possible issues during the run

Not Responding
Job that DSC has not received any status for 15 minutes

New
Job that has never been run, or a job in which existing save sets were deleted because of deletion guidelines in the data retention policy

Update System Credentials

With DSA 16.00 release, you now have the ability to change the system credentials for all jobs associated with a system and user account. This eliminates the tedious task of updating system passwords for each job.

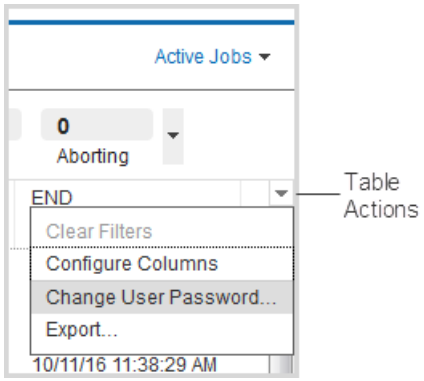
The password updates can occur in two ways:

- When you enter a password in the **Enter System Credentials** window when creating or editing a job, the password is applied to all jobs associated with the system and selected user account.
- You can use the new **Change User Password** option in the Table Action menu, see Changing the User Password.

Changing the User Password

Change User Password is available in the Table Action menu when at least one job with user credentials is present. The wizard changes the password for all jobs associated with the user on the selected system.

1. Select **Change User Password** from the **Table Actions** menu.



2. Select the **System** to update the user password for and click **Next**.
3. Select the **User** to update the password for and click **Next**.
4. Enter the new Password and click **Next**.
   A confirmation message indicating the system and user that will be updated displays.
5. Select **Update Password**.

Backup Jobs

Backup jobs archive objects from a Teradata source system to a target group. Target groups are defined by a BAR administrator in the **BAR Setup** portlet or command-line interface.

The **BAR Operations** portlet allows you to migrate the object list from an existing ARC script into a backup job. Objects in that list that exist in the specified source system are automatically selected in the object browser when a new job is created from the migrated ARC script.

When you run a backup job for the first time or when you change the target group for a backup job, all data from the specified objects is archived. After this initial full backup, you can choose the backup type:

- Full: Archives all data from the specified objects

- Delta: Archives only the data that has changed since the last backup operation
- Cumulative: Archives the data that has changed since the last full backup was run

Creating and Running a Teradata Backup Job

Use these steps to create a standard CBB backup job. For IR-enabled backups, see Using Incremental Restore to Maintain Your DR System.

Create the Job

1. Open the **BAR Operations** portlet.
2. Select **New Job**.
   If the **New Job** window does not appear, select the **Saved Jobs** tab.
3. On the **New Job** screen:
   a. Select **Backup** as the job type.
   b. [Optional] To migrate objects from an existing ARC or TARA script, click **Browse** and select the script.
   c. Click **OK**.
4. Under **New Backup Job**:
   a. Enter a unique **Job Name**.
   b. Select a **Source System**.
   c. In **Enter System Credentials**, enter a user name and password for the system.
   Account String information is not required.

   Note:
   The password is applied to all jobs associated with this system and user account.

   d. Select a **Target Group**.
   e. [Optional] Enter a job description.
5. Select the **Objects** tab.
6. Select the objects from the source system to backup.
7. [Optional] Select **Selection Summary** to verify the selected objects..

   Note:
   Size information is not available for DBC only backup jobs. N/A displays as the size value for DBC only backup jobs.

8. [Optional] To adjust job settings for the job, click the **Job Settings** tab.
9. Click **Save**.
   The newly created backup job is listed in the **Saved Jobs** view.

Run the Job

10. Click next to a job.
11. Select **Run**.
12. Select **OK**.
13. Select the **Job Scope**: **Full**.
    A newly created job only has Full as the available Job Scope. When you rerun an existing job, Delta and Cumulative are available.
14. Select **Run**.

Related Information:

Changing Job Permissions
Job Settings Tab
ARC Script Migration

ARC Script Migration

The **Migrate ARC script** allows users to import an existing ARC or TARA script into the **BAR Operations** portlet. Only the set of objects that define the backup job are migrated into the portlet. Information about target media, number of streams, and connection parameters will not migrate into the portlet from the ARC scripts.

ARC script syntax `EXCLUDE` is supported at object level. `EXCLUDE` is supported at database level, but a database range is not allowable for exclusion. If any objects in the script do not exist in the selected source system, they will not be included in the new job.

Running an Existing Backup Job

Use these steps to rerun an existing backup job. You can choose full, delta, or cumulative backup types.

1. Click next to a job.
2. Select **Run**.
3. Select **OK**.
4. Select the **Job Scope**: **Full**, **Delta**, or **Cumulative**.
5. Select **Run**.

Restore Jobs

Restore jobs are based on successful executions of Teradata backup jobs, and can only be created for a backup job that has successfully run to completion.

You can define a restore job to always restore the latest version of a backup save set or you can specify a save set version. A target Teradata system must be selected in order to define the restore job. By default, all objects from the save set are included in the restore job but the selections can be modified.

Creating and Running a Teradata Restore Job

Use these steps to create a standard CBB restore job. For IR-enabled restores, see Using Incremental Restore to Maintain Your DR System.

1. Open the **BAR Operations** portlet.
2. From the **Saved Jobs** view, do one of the following:

Option	Description
Create a new job	a. Click **New Job**. b. Select **Restore** as the job type and click **OK**.
Create a job from an existing job	a. Click next to a backup job that has completed. b. Select **Create Restore Job** to create a restore job from the selected save set.
Create a job from migrated job metadata **Note:** Migrated job metadata results when tapes and metadata information that pointed to a specific backup job were migrated from one DSA environment to a different one.	a. Click next to a migrated job. b. Select **Create Restore Job** to create a restore job from the selected migrated job.

3. Under **New Restore Job** enter a unique **Job Name**.
4. To change the save set, click **Edit**, select **Specify a version**, and select the save set to use.

   Note:

   If the selected job is retired, the version information is not selectable.

5. Select the **Destination System** and enter the **Credentials** for it.

   Note:

   The password is applied to all jobs associated with this system and user account.

6. Select the **Target Group**.
7. [Optional] Add a job description.
8. To change the objects selected, clear the checkboxes and select others in the **Objects** tab.
9. If you have created a backup job on the TD_SERVER_DB database, and the job contains a SQL-H object, you can map the restore job to a different database:
   a. Select the SQL-H object in the **Objects** tab.
   b. Click next to the SQL-H object.
   c. In the **Settings** box, map the restore job to a different database.

10. [Optional] Select **Selection Summary** to verify the selected objects..

    Note:

    Size information is not available for DBC only backup jobs. N/A displays as the size value for DBC only backup jobs.

11. [Optional] To adjust job settings for the job, select the **Job Settings** tab.

    Settings can include specifying whether a job continues or aborts if an access rights violation is encountered on an object.

    Note:

    - [Teradata Database 16.20 Update 1 and later] Fallback cannot be disabled. Checkbox is ignored.
    - [Prior versions of Teradata Database] Disable fallback is not available unless **Run as copy** is checked. The icon ⃠ appears when the mouse pointer is hovered over the checkbox.

12. Click **Save**.
13. To run the newly created restore job, in the **Saved Jobs** view:
    a. Click next to a job.
    b. Select **Run**.

Related Information:

Restarting an Unsuccessful Restore Job
Changing Job Permissions
Job Settings Tab

Restarting an Unsuccessful Restore Job

Beginning with DSA 16.20.12 and Analytics Database 16.20 Feature Update 1, you can restart an unsuccessful restore job. The job restarts after the last completed task.

1. From the **Saved Jobs** view, select the incomplete job.
2. Click next to the job.
3. Select **Restart**.

Analyze Jobs

An analyze job uses either a read-only or validate analysis method for each job. An analyze read-only job reads the data from the media device to verify that reads are successful. An analyze validate job sends the data to the AMPs, where it is interpreted and examined but not restored.

Creating a Teradata Analyze Job

1. From the **Saved Jobs** view of the **BAR Operations** portlet, do one of the following:

Option	Description
Create a new job	a. Click **New Job**. b. Select the Analyze job type and click **OK**.
Create a job from a backup job save set	a. Click next to a backup job that has completed. b. Select **Create analyze job** to use the selected save set for the analyze job.

2. From the **New Analyze Job** view:
   a. Enter a unique **Job Name**.
   b. [Optional] Select an analysis method, if not already specified.

   Note:
   If you change the analysis method to **Read and validate**, you must provide the **Destination system** and its credentials.

   c. Select the **Job to analyze**.
   d. Select the **Target Group**.
   e. [Optional] Provide a job description.
3. Specify a save set version from the **Save Set Version** tab.

   Note:
   If the selected job is retired, the Save Set Version information is not selectable.

4. To adjust job settings for the job, click the **Job Settings** tab.
5. Click **Save**.
6. To run the newly created analyze job, in the **Saved Jobs** view:
   a. Click next to a job.
   b. Select **Run**.

Related Information:

Job Settings

Job Settings Tab

The **Job Settings** tab is available when creating or cloning a job and allows you to select and define job settings for backup, restore, and analyze jobs.

Field	Description	Job Type
Automatically Retire	Determines whether a job is retired automatically. **Never** Default. The job is not retired automatically. The job retire must be set manually. **After** Specifies the time, in days or weeks, that a job is automatically retired.	Backup Restore Analyze
Backup Method	Determines the type of backup to perform. **Offline** Default. Backs up everything associated with each specified object while the database is offline. No updates can be made to the objects during the backup job run. **Online** Backs up everything associated with each specified object and initiates an online archive for all objects being archived. The online archive creates a log that contains all changes to the objects while the archive is prepared. **Note:** A DBC only backup job cannot be run as online. **Dictionary Only** Backs up only the dictionary and table header information for each object. **Note:** You cannot use cumulative or delta backup on a Dictionary Only backup job.	Backup
No sync checkbox	Determines where synchronization is done for the job. Only available for online backup jobs. **Unchecked** Default. Synchronization occurs across all tables simultaneously. If you try to run a job that includes objects that are already being logged, the job aborts. **Checked** There can be different synchronization points. If you try to run a job that includes objects that are already being logged, the job runs to completion and a warning is returned.	Backup
Logging Level	Determines the types of messages that the database job logs. **error** Default. Enables minimal logging. Provides only error messages.	Backup Restore Analyze_ Validate

Field	Description	Job Type
	warning Adds warning messages to error message logging. **info** Provides informational messages with warning and error messages to the job log. **debug** Enables full logging. All messages, including Debug, are sent to the job log.	
Job Permissions	• **Not shared** Job permissions have not been defined. • Shows the cumulative number of users and roles with shared permissions, if job permissions have been defined. • Job permissions include **Run**, **Edit**, or both. Click **Edit** to define or change permissions.	Backup Restore Analyze
Abort On Access Rights Violation	**Unchecked** Allows the Teradata backup or restore job to proceed even when a DUMP access rights violation is encountered on an object. **Checked** Default. Job aborts when the access rights violation is encountered. **Note:** This checkbox appears only when the following are true: • Source system is Analytics Database 16.20 or later / Teradata Database version 15.10.01.00 or later • Source system and target group credentials have been validated • When creating or editing a Teradata backup or restore job	Backup Restore
Query Band	Allows tagging of sessions or transactions with a set of user-defined name-value pairs to identify where a query originated. These identifiers are in addition to the current set of session identification fields, such as user ID, account string, client ID, and application name. **Note:** Valid query band values are defined on the database. DSA creates query bands for restore jobs when an override temperature or block level compression option has a value other than **DEFAULT**. You can enter different query bands in the bottom text box.	Backup Restore
Run as copy checkbox	When checked allows the restore to run as a copy job.	Restore

Field	Description	Job Type
	• A copy job assumes the destination database is not the original database system on the backup job and the database ID is different. • A copy job is used when restoring to a database with a different internal database ID than the one in the backup save set. **Note:** Do not use **Run as copy** when the database system is the original database and the database ID matches the one found in the backup save set.	
Disable Fallback checkbox	Fallback protection means that a copy of every table row is maintained on a different AMP in the configuration. Fallback-protected tables are always fully accessible and are automatically recovered by the system. **Note:** • [Teradata Database 16.20 Update 1 and later] Fallback cannot be disabled. Checkbox is ignored. • [Prior versions of Teradata Database] Disable fallback is not available unless **Run as copy** is checked. The icon ⃠ appears when the mouse pointer is hovered over the checkbox. **Unchecked** Default. Restored tables are recreated with fallback automatically enabled. **Checked** Fallback is not available.	Restore
DBC Credentials	DBC only backup. Click **Set Credentials** to open the **Enter Credentials** dialog box if DBC credentials need to be established.	Restore
Advanced Settings button		
Track empty tables	Tracks empty tables during a backup job. Supported in Teradata Database 16.0 and later.	Backup
Skip statistics	Skips collecting statistics during a backup or restore job. Supported in Analytics Database 16.20 and later / Teradata Database 16.0 and later.	Backup Restore
Enhanced Locking Management checkbox	Checked. DSA locks one object at a time. The object is unlocked once it is backed up. DSA does not manage data consistency. Unchecked. DSA locks all objects at the beginning of the backup. The object is unlocked once it is backed up. This ensures data consistency between objects.	Backup (offline only)
Override Temperature	Determines the temperature at which data is restored.	Restore

Field	Description	Job Type
	DEFAULT Default. This data is restored at the default temperature setting for the system. **HOT** This data is accessed frequently. **WARM** This data is accessed less frequently. **COLD** This data is accessed least frequently.	
Block Level Compression	Defines data compression used. **DEFAULT** Default. Applies same data compression as the backup job if allowed on the destination system. **ON** Compress data at the block level if allowed on the destination system. **OFF** Restore the data blocks uncompressed.	Restore
Concurrent Builds Per Table	Number of index and fallback subtables that can be built concurrently per table during restore. Supported in Analytics Database 16.20 and later / Teradata Database 16.0 and later.	Restore
Map to a different database on destination system	Database name to map to.	Restore
Limit the maximum streams per node for this job	When checked, overrides the system level stream soft limit. Range is 1 to 1 - *max* of the stream soft limit, where *max* is the number of AMPs per node in the system. **Note:** Job level entry overrides the system level setting.	Backup Restore Analyze_ Validate
Skip join/ hash index	When checked, skips archiving the join and hash index during a backup job or skips restoring the join index during a restore job.	Backup Restore
Skip build secondary indexes	When checked, skips building secondary indexes during incremental restore.	Incremental Restore

Field	Description	Job Type
[Analytics Database 16.20 Update 2]		

Objects Tab

The **Objects** tab is available when creating or cloning a job and displays the object browser. The object browser provides you with the controls to view a list of objects that are on a source database system and archive objects to a target group and restore these objects to a destination system. The object browser simplifies the process of viewing and selecting database objects for backup and restore jobs.

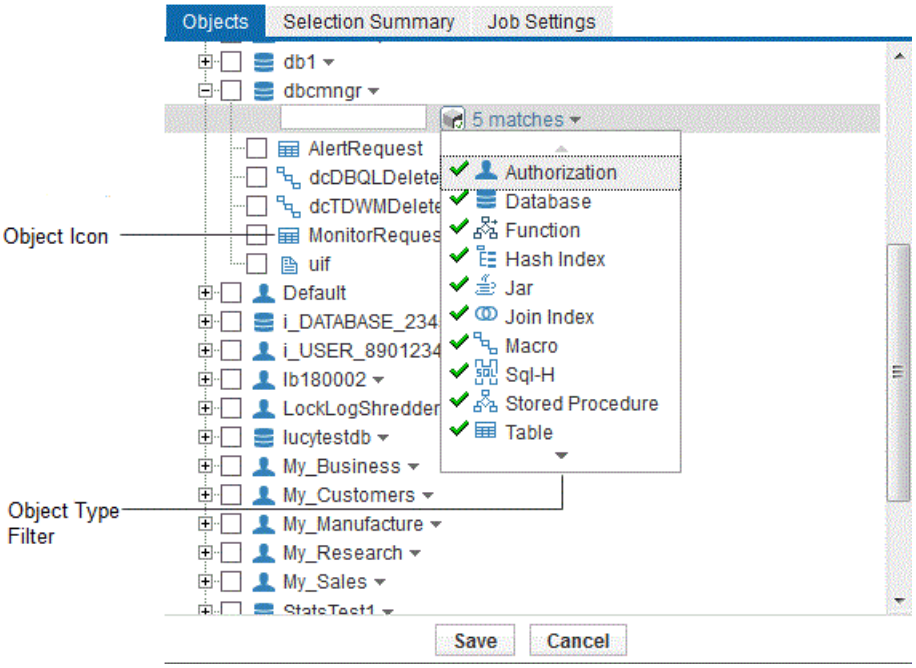
Database objects display as a hierarchically-organized tree. You can use filtering to limit the number of objects displayed in the tree. Expand a branch of objects in the tree by clicking next to the object type.

Controls in the Object Browser

Control	Action
Object Icon	Identifies the database object type. Hovering over the object icon shows the object type and full object name.
Object Type Filter	Enables you to select the type of object to display.
(Object Settings Menu)	Opens the Settings menu.

Object Settings

Field	Description	Job Type
Include all the children that are databases and users	Check to select all the child objects	Backup Restore (database object) Incremental Restore (database object)
Map to a different database on destination system	Select a different database on the destination system to restore to.	Restore (database object) Incremental Restore (database object)
Advanced Settings button		Restore Incremental Restore
Destination System Hash Map	Select the hash map to use on the destination system.	Incremental Restore



Save Set Version Tab

For analyze jobs, the **Save Set Version** tab is available after you select **Create Analyze Job** from the **Saved Jobs** view. It allows you to select a version of the save set against which to run your analyze job. You can select the latest version or you can specify a version if more than one save set exists.

When you specify a version, selecting the **Table Actions** menu, and then **Configure Columns** allows you to select, lock, and designate the order of columns.

The following columns are available:

Column Header	Description
BACKUP DATE	Start date and time job began
OBJECTS	A count of database objects copied during the job
SIZE	Aggregate size for the objects processed
BACKUP TYPE	Full, delta, or cumulative backup associated with the save set
TYPE	Backup
TARGET GROUP	Target group of the backup job
COMPLETION DATE	Date and time the backup job was completed
LOCATION	Location where the objects or third-party target group were backed up
JOB PHASE	Job phase associated with the save set

Selection Summary Tab

The **Selection Summary** tab is available when creating or cloning a job and provides a tabular view of the objects explicitly selected in the **Objects** tab. Only selected objects and object settings are displayed. You can select, lock, and designate the order of columns from the **Table Actions** menu.

The following columns are available:

Column Header	Description
PARENT	Parent object of the selected object
OBJECT	Name of the selected object
TYPE	Object type of the selected object
SIZE	Object size of the selected object **Note:** Size information is not available for DBC only backup jobs. N/A displays as the size value for DBC only backup jobs.
RENAME	Name to which the selected object will be renamed
REMAP	Database to which the selected object will be remapped

Managing Jobs

This task outlines the tasks you can use to manage BAR operations jobs.

1. Choose the type of job you need to create:

Job Type	Description
Backup	To create a new backup job, refer to: Creating and Running a Teradata Backup Job
Restore	To create a new restore job or a restore job from a backup job save set, refer to: Creating and Running a Teradata Restore Job
Analyze	To create a new analyze job or an analyze job from a backup job save set, refer to: Creating a Teradata Analyze Job

2. Select and define jobs settings.
3. Change job permissions if anything about the job changed.
4. Monitor a running job's status and view job phase log updates.
5. Abort a job from the **Saved Jobs** view or the **Job Status** view.
6. Retire a job from the **Saved Jobs** view.

7. Activate a job from the **Retired Jobs** view.
8. Delete an active, saved, or retired job.

Running a Job

1. From the **Saved Jobs** view, click next to the job you want to run.
2. Select **Run**.
   If you selected a backup job that can only be run as a full backup (for example, a new backup job, or a job with a new target group), the FULL backup runs automatically.
3. If you did not choose a backup job that can only be run as a full backup, select one of the following and click **Run**.

Backup Type	Description
Full	Archives all data from the specified objects
Delta	Archives only data that has changed since the last backup
Cumulative	Archives the data that has changed since the last full backup, consolidating multiple delta or cumulative backups

4. For a repository restore job, click **OK** to confirm that you want to restore the repository from the latest repository backup save set.
   The **BAR Setup** and **BAR Operations** portlets are unavailable while the repository restore job is running.

Related Information:

Restarting an Unsuccessful Restore Job

Editing Jobs

You can edit Teradata backup, restore, and analyze jobs. The fields that you can edit depend on the type of job you are editing.

Editing a Teradata Backup Job

You can edit any backup job, whether or not the job has been previously run.

1. From the **Saved Jobs** view:
   a. Click next to the backup job you want to edit.
   b. Select **Edit.**
2. From the **Edit Backup Job** view:
   a. [Optional for a backup job that has not been run] Change the **Source System** and **Credentials**.

Note:

After a backup job has been run successfully and has a save set, the source system for the job cannot be modified. For jobs that have not been run, changing the system or credentials can result in a mismatch between the selected objects and the available database hierarchy, which could cause the job to fail.

b. [Optional] Change the **Target Group**.

3. [Optional] In the **Objects** tab, change the objects from the source system.
4. [Optional] Select **Selection Summary** to verify the selected objects..

Note:

Size information is not available for DBC only backup jobs. N/A displays as the size value for DBC only backup jobs.

5. [Optional] To adjust job settings for the job, click the **Job Settings** tab.
   Settings can include specifying whether a job continues or aborts if an access rights violation is encountered on an object.
6. Click **Save**.

Related Information:

Job Settings
Changing Job Permissions

Editing a Teradata Restore Job

1. From the **Saved Jobs** view:
   a. Click next to the restore job you want to edit.
   b. Select **Edit**.
2. In the **Edit Restore Job** view:
   a. [Optional] Change the **Destination System** and **Credentials**.
   b. [Optional] Change the **Target Group**.
   c. [Optional] Add a job description.
3. [Optional] To change the objects selected, clear the checkboxes and select others in the **Objects** tab.
4. [Optional] Select **Selection Summary** to verify the selected objects..

Note:

Size information is not available for DBC only backup jobs. N/A displays as the size value for DBC only backup jobs.

5. [Optional] To adjust job settings for the job, click the **Job Settings** tab.

Settings can include specifying whether a job continues or aborts if an access rights violation is encountered on an object.

Note:

- [Teradata Database 16.20 Update 1 and later] Fallback cannot be disabled. Checkbox is ignored.
- [Prior versions of Teradata Database] Disable fallback is not available unless **Run as copy** is checked. The icon ⃠ appears when the mouse pointer is hovered over the checkbox.

---

6. Click **Save**.

Related Information:

Job Settings
Changing Job Permissions

Editing a Teradata Analyze Job

1. From the **Saved Jobs** view:
   a. Click next to the analyze job you want to edit.
   b. Select **Edit.**
2. From the **Edit Analyze Job** view:
   a. [Optional] Change the analysis method.
   b. [Optional] Select the job to analyze.
   c. [Optional] Change the **Destination System** and **Credentials**.
   d. [Optional] Change the **Target Group**.
   e. [Optional] Add a job description.
3. [Optional] Select a save set version from the **Save Set Version** tab.
4. [Optional] To adjust job settings for the job, click the **Job Settings** tab.
5. Click **Save**.

Related Information:

Job Settings
Changing Job Permissions

Cloning Jobs

A cloned job copies the parameters of an existing job, however, the job requires a different name. Any type of job can be cloned.

Note the following items:

- You cannot change source systems when cloning a backup job.

- You can change destination systems when cloning a restore job.

Cloning a Teradata Backup Job

1. From the **Saved Jobs** view:
   a. Click next to the backup job you want to clone.
   b. Select **Clone**.
2. In **Clone Backup Job** update as needed:

Option	Description
Job Name	[Required] Enter a unique name.
Source System (DO NOT CHANGE)	Do not change the source system. None of the objects will be cloned if you change the source system.
CREDENTIALS	[Optional] Changing the credentials can result in a mismatch between the selected objects and the available database hierarchy, which could cause the job to fail.
Target Group	[Optional]
Description	[Optional]

3. [Optional] To change the objects selected, clear and select checkboxes in the **Objects** tab.
4. [Optional] Select **Selection Summary** to verify the selected objects..

Note:

Size information is not available for DBC only backup jobs. N/A displays as the size value for DBC only backup jobs.

---

5. [Optional] Use the **Job Settings** tab to adjust job settings for the job.
6. Click **Save**.

Related Information:

Job Settings

Cloning a Teradata Restore Job

1. From the **Saved Jobs** view:
   a. Click next to a job.
   b. Select **Clone.**
2. From the **Clone Restore Job** view:
   a. Enter a unique **Job Name**.
   b. [Optional] To change the **Source Save Set**, click **Edit**, select **Specify a version**, and select the save set to use.

Note:

If the selected job is retired, the Save Set Version information is not selectable.

c. [Optional] Change the **Destination System** and the **Credentials** associated with it.
d. [Optional] Change the **Target Group**.
e. [Optional] Change the job description.

3. [Optional] To change the objects selected, clear the checkboxes and select others in the **Objects** tab.
4. [Optional] Select **Selection Summary** to verify the selected objects..

Note:

Size information is not available for DBC only backup jobs. N/A displays as the size value for DBC only backup jobs.

5. [Optional] To adjust job settings for the job, click the **Job Settings** tab.

Note:

- [Teradata Database 16.20 Update 1 and later] Fallback cannot be disabled. Checkbox is ignored.
- [Prior versions of Teradata Database] Disable fallback is not available unless **Run as copy** is checked. The icon ⃠ appears when the mouse pointer is hovered over the checkbox.

6. Click **Save**.
7. To run the cloned restore job, in the **Saved Jobs** view:
   a. Click next to a job.
   b. Select **Run**.

Related Information:

Job Settings

Cloning a Teradata Analyze Job

1. From the **Saved Jobs** view:
   a. Click next to a job.
   b. Select **Clone**.
2. Enter a unique **Job Name**.
3. [Optional] Change the **Analysis Method**.
4. [Optional] Change the **Target Group**.
5. [Optional] Change the job description.

6. [Optional] From the **Save Set Version** tab, change the save set version.
7. [Optional] To adjust job settings for the job, click the **Job Settings** tab.
8. Click **Save**.
9. To run the cloned analyze job, in the **Saved Jobs** view:
   a. Click next to a job.
   b. Select **Run**.

Related Information:

Job Settings

Retiring a Job

You can retire a job from the **Saved Jobs** view, if the job is not in **Running**, **New**, **Aborting**, **Not Responding**, or **Queued** status.

When you retire a job, the job moves from the **Active Jobs** view to the **Retired Jobs** view.

Note:

A retired job will be automatically deleted if this setting is configured through the **BAR Setup** portlet or DSA command-line interface. A warning message will appear before the job is retired reporting the deletion date.

---

1. Click next to a job.
2. Select **Retire**.
3. Click **OK** to confirm the job retirement.

Activating a Job

You can activate a job from the **Retired Jobs** view.

When you activate a job, the job is moved from the **Retired Jobs** view to the **Active Jobs** view.

1. Open the **Retired Jobs** view.

2. Click next to a job.
3. Select **Activate**.
4. Click **Yes** to confirm the job activation.

Deleting a Job

You can immediately delete a job from the **Retired Jobs** view. You can also delete a job from the **Saved Jobs** view, if the job has a status of **New**.

1. Click next to a job.
2. Select **Delete**.

Note:

If you are attempting to delete a backup job with dependent restore or analyze jobs, a message displays with the dependent job names that must be deleted before you can delete the backup job.

---

3. Click **Yes** to confirm the job deletion.
   The job and job history will be deleted immediately and cannot be restored.

Aborting a Job

You can abort a running job from either the **Saved Jobs** or the **Job Status** view.

1. Choose the view to select the job you need to abort:

View	Description
Saved Job	a. Click next to a job. b. Select **Abort**.
Job Status	a. Click **Abort**.

2. Click **OK** to confirm you want to abort the job run.

Changing Job Permissions

When you create a job, you can set permissions that allow some users or roles to run the job and some users or roles to edit the job. After a job is created, you can change permissions for users or roles.

To designate job permissions, you must be the owner of the job or the DSA administrator.

1. From the **Saved Jobs** view, create or edit a job.
2. Click the **Job Settings** tab and then click **Edit**.
3. Select users and roles to grant access.

Option	Description
Users	a. In the **Available Users** box, select one or more users and click to move it to the **Selected Users** box.

Option	Description
	b. Select a user and grant access to **Run** or **Edit**.
Roles	a. In the **Available Roles** box, select one or more roles and click > to move it to the **Selected Roles** box. b. Select a role and grant access to **Run** or **Edit**.

4. Click **OK**.

Viewing Job Status: Portlets

Depending on the type of the job and the run status, you can view the details from the **Saved Jobs** or **Job History** view.

1. Do one of the following, depending on the run status of the job:

View Option	Description
Status of running job or the most recent job	a. Click the **Saved Jobs** tab. b. Click next to a job. c. Select **Job status**. If the job is currently running, the **Streams** tab and a progress bar indicating the percentage complete display. If a job is restarted, the progress bar restarts at zero. While a job is running, the maximum number of objects reported is 500. The **Log** tab displays details about the objects included in the job.
Status logs of previously run jobs	a. Click the **Job History** tab. b. Click the row for the job. The **Log** tab displays details about the objects included in the job. The maximum number of completed objects is 100.

2. Select any of these links to see more details:
   Available links depend on job type.

Link	Description
View Phase Log	Backup and analyze_validate jobs: Provides dictionary and data phase details. Restore jobs: Provides dictionary, data, build, and postscript phase details.
View History	All jobs: Provides job history, including start and end times, duration, status, objects, and restart group ID (restore jobs). Restart group ID helps identify multiple runs that are part of a series of restarted restore jobs.
View Cumulative Status	Restore jobs that have restarted: Provides job history from initial incomplete run through the latest restarted execution, including start and end times, duration,

Link	Description
	objects, and restart group ID. This view is available when viewing the status of the last run in a group. Restart group ID helps identify multiple runs that are part of a series of restarted restore jobs.
View Save Sets	Backup jobs: Provides save set details, including backup date, objects, size, and backup type.
View Error Log	Failed analyze read jobs: Provides error log details, including error code, error message, warning code, and warning message.

Log Tab

The **Log** tab displays details about database objects for running and completed backup, restore, and analyze_validate jobs. The tab is available when viewing a job status.

You can filter the results by column. For example, to display only object names beginning with "order", type `order` in the **OBJECT NAME** box.

You can select, lock, and designate the order of columns from the **Table Actions** menu.

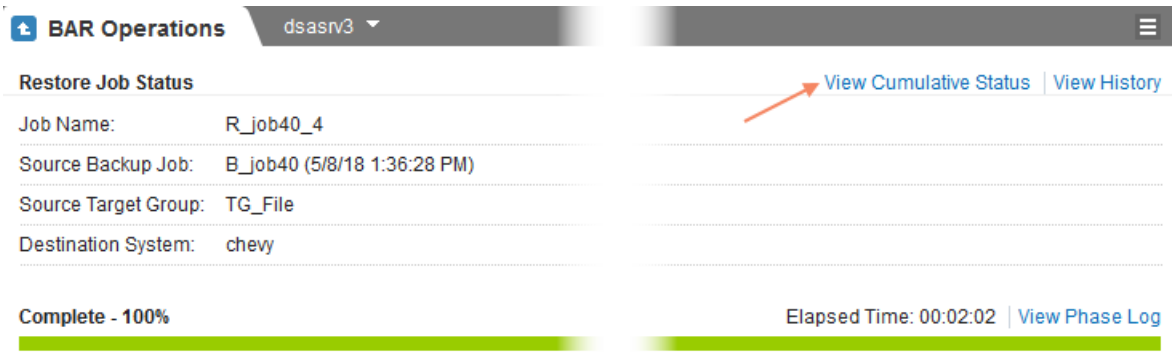
Field	Description	Job Type
START	Start date and time job began	Analyze_Read
END	Date and time job ended	Backup Restore Analyze_Read Analyze_Validate
FILE NAME	The backup files that contain the save set	Analyze_Read
OBJECT NAME	Name of the object being backed up, restored, or validated	Backup Restore Analyze_Validate
OBJECT TYPE	Type of object being backed up, restored, or validated	Backup Restore Analyze_Validate
PHASE	The job phase can be dictionary, data, build, or postscript.	Backup Restore Analyze_Validate
STATUS	The job status of the object	Backup Restore Analyze_Read Analyze_Validate

Field	Description	Job Type
PARENT NAME	Specifies the name of the parent of the object being backed up, restored, or validated	Backup Restore Analyze_Validate
BYTE COUNT	Total number of bytes copied	Backup Restore Analyze_Read Analyze_Validate
ROW COUNT	Total number of rows copied	Backup Restore Analyze_Validate
ERROR CODE	Specifies the error code encountered Hover over the code to see detailed message text.	Backup Restore Analyze_Read Analyze_Validate
WARNING CODE	Specifies the warning code encountered Hover over the code to see detailed message text.	Backup Restore Analyze_Read Analyze_Validate

Cumulative Status

Beginning with DSA 16.20.12 and Teradata Database 16.20 Feature Update 1, you have the ability to restart a restore job that failed, aborted, or completed with warnings or errors. **View Cumulative Status** is a new link that displays status information for restore jobs that have been restarted.

The **View Comulative Status** link is only available on restore jobs that have been restarted.



In addition to the standard Log tab, you now have a cumulative status section.

Field	Description
First Run	Start time of the first run
Last Run	End time of the last run
Total Run Time	Cumulative time of all runs until this one
Total Executions	Number of runs

Streams Tab

The **Streams** tab displays details about the job streams during a backup, restore, or analyze job. The tab is available when viewing the job status of a running job. You can select, lock, and designate the order of columns from the **Table Actions** menu.

Field	Description
Node	Specifies the node where the job stream is running
Stream	Numerically identifies a job stream
Object	Name of object being backed up, restored, or analyzed
Average Stream Rate (Data phase)	For a backup job, the number of bytes reported by DSMain since the stream started. For a restore and analyze_validate job, the number of bytes reported by the DSA Network Client (ClientHandler) since the stream started.

Phase Log

The **Phase Log** displays details about database objects in running and completed backup, restore, and analyze jobs. This information is read-only. The **Phase Log** is available when viewing the status of a job or a repository job.

Field	Description	Job Type
Job Phase	The job phase to which the information pertains. Backup jobs have two phases: Dictionary and Data. In addition to Dictionary and Data, restore jobs have Build and Postscript phases.	Backup Restore Analyze_ Validate

Field	Description	Job Type
Objects	Number of objects processed during the phase	Backup Restore Analyze_ Validate
SQL Count	Number of SQL queries issued by DSMain backend	Backup Restore
Dictionary Row Count	Number of dictionary rows received from Dump SQL Response	Backup Restore
Table Header Count	Total number of objects that have a table header	Backup Restore
Enabled Incremental Logging Count	Number of times the LOGGING INCREMENTAL ON statement is submitted for an IR-enabled backup job. If the count continues to increase, the backup job might be blocked by another user's locks.	IR-Enabled Backup
Start	Date and time the phase began Restarted restore: Start time of the first run	Backup Restore Analyze_ Validate
End	Date and time the phase ended Restarted restore: Start time of the last run	Backup Restore Analyze_ Validate
Average speed (Data phase)	• Backup jobs: average speed = sum of bytes reported by DSMain for all objects / Time interval from time first byte of data received from DSMain and last object backed up, plus the refresh rate (which is 30 seconds by default). The average backup rate includes tape mount, positioning, and close time. • Restore jobs: average speed = sum of bytes reported by DSMain for all objects / Time interval from first receipt of data for first object from the DSA Network Client through data transfer for last object, plus the refresh rate (which is 30 seconds by default). The average restore rate includes tape mount, positioning, and close time, and the time for the concurrent table index build process while the data is being restored. The time for any remaining table index builds after the restore data transfer of the last object is completed is not included.	Backup Restore Analyze_ Validate
Size (Data phase)	Size of the data processed during the phase duration	Backup Restore Analyze_ Validate
Total Run Time	Restarted restore: Total duration of each phase	Restore

Viewing Save Sets

You can view all of the save sets associated with a given backup job.

1. Do one of the following, depending on the tab you are currently viewing:

View Option	Description
Saved Jobs tab	• Click next to a job. • Select **Job status**.
Job History tab	• Click the row of the job

2. Click **View Save Sets**.
   The **Save Sets** view lists the save sets for the selected job. You can select, lock, and designate the order of columns from the **Table Actions** menu.

Column Header	Description
BACKUP DATE	Date and time the backup job started
OBJECTS	Number of objects processed
SIZE	Aggregate size of the objects processed
BACKUP TYPE	Full, delta, or cumulative backup
TYPE	Job type associated with the save set
TARGET GROUP	Target group associated with the save set
COMPLETION DATE	Date and time the backup job finished
LOCATION	Location where the objects or third party target group were backed up
JOB PHASE	Job phase associated with the save set

Viewing Backup IDs

You can view the backup IDs for a given job name and save set for a NetBackup job.

Note:

Currently in the portlet, you can only view backup IDs for a NetBackup job. To query and display backup IDs generated by a disk file system, DD Boost, or AWS use the CLI commands: `query_backupids` and `list_query_backupids`.

1. Do one of the following, depending on the run status of the job:

View Option	Description
Status of running job or the most recent job	• Click the **Saved Jobs** tab. • Click next to a job. • Select **Job status**.
Status logs of previously run jobs	• Click **Job History** tab. • Click the row for the job.

2. Click **View Save Sets**.
3. Click next to a save set and select **Backup IDs**.

The **BACKUP IDS** for the save set are listed. You can select, lock, and designate the order of columns from the **Table Actions** menu.

Column Header	Description
BACKUP ID	Backup ID for the given job name and save set.
FILE NAME	File name of the file associated with the backup ID.
FILE SIZE	File size of the file associated with the backup ID
DATE	Date and timestamp for the file created.

Job History View

The **Job History** view displays a table of BAR jobs that have been run, and allows you to view the details of the last job run.

Filters

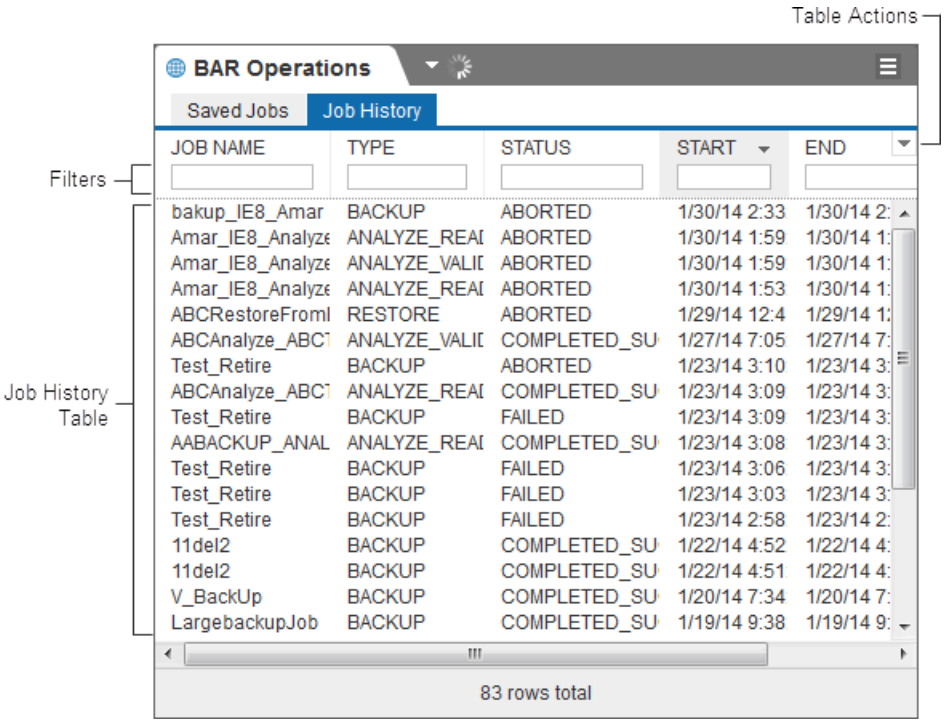
Shows only rows that match your filter criteria.

Job Table

Lists the job name, type, status, start time, end time, size and duration of the job.

Table Actions

Configure Columns allows you to select, lock, and order the displayed columns.
Export creates a .csv file containing all available data.



Viewing Job History

The **Job History** tab of the **BAR Operations** portlet displays a list of all job executions. You can view more detailed information about a single job execution from either the **Job History** or **Saved Jobs** view.

1. Click the job row to display job status.
2. Click **View History**.

The **Job History** for the job appears. You can select, lock, and designate the order of columns from the **Table Actions** menu.

Column Header	Description
START	Start date and time job began
END	Date and time job ended
DURATION	Total time the job ran
STATUS	The job status of the job run
OBJECTS	A count of database objects copied during the job
SOURCE	Source system (backup) of the job.
DESTINATION	Backup job: Target group to which the data is backed up. Restore job: Teradata system to which the data is restored.

Column Header	Description
RESTART GROUP ID	Restore job: Helps identify multiple runs that are part of a series of restarted restore jobs.

Viewing Save Sets

You can view all of the save sets associated with a given backup job.

1. Do one of the following, depending on the tab you are currently viewing:

View Option	Description
Saved Jobs tab	• Click next to a job. • Select **Job status**.
Job History tab	• Click the row of the job

2. Click **View Save Sets**.
   The **Save Sets** view lists the save sets for the selected job. You can select, lock, and designate the order of columns from the **Table Actions** menu.

Column Header	Description
BACKUP DATE	Date and time the backup job started
OBJECTS	Number of objects processed
SIZE	Aggregate size of the objects processed
BACKUP TYPE	Full, delta, or cumulative backup
TYPE	Job type associated with the save set
TARGET GROUP	Target group associated with the save set
COMPLETION DATE	Date and time the backup job finished
LOCATION	Location where the objects or third party target group were backed up
JOB PHASE	Job phase associated with the save set

Phase Log

The **Phase Log** displays details about database objects in running and completed backup, restore, and analyze jobs. This information is read-only. The **Phase Log** is available when viewing the status of a job or a repository job.

Field	Description	Job Type
Job Phase	The job phase to which the information pertains. Backup jobs have two phases: Dictionary and Data. In addition to Dictionary and Data, restore jobs have Build and Postscript phases.	Backup Restore Analyze_ Validate
Objects	Number of objects processed during the phase	Backup Restore Analyze_ Validate
SQL Count	Number of SQL queries issued by DSMain backend	Backup Restore
Dictionary Row Count	Number of dictionary rows received from Dump SQL Response	Backup Restore
Table Header Count	Total number of objects that have a table header	Backup Restore
Enabled Incremental Logging Count	Number of times the LOGGING INCREMENTAL ON statement is submitted for an IR-enabled backup job. If the count continues to increase, the backup job might be blocked by another user's locks.	IR-Enabled Backup
Start	Date and time the phase began Restarted restore: Start time of the first run	Backup Restore Analyze_ Validate
End	Date and time the phase ended Restarted restore: Start time of the last run	Backup Restore Analyze_ Validate
Average speed (Data phase)	• Backup jobs: average speed = sum of bytes reported by DSMain for all objects / Time interval from time first byte of data received from DSMain and last object backed up, plus the refresh rate (which is 30 seconds by default). The average backup rate includes tape mount, positioning, and close time. • Restore jobs: average speed = sum of bytes reported by DSMain for all objects / Time interval from first receipt of data for first object from the DSA Network Client through data transfer for last object, plus the refresh rate (which is 30 seconds by default). The average restore rate includes tape mount, positioning, and close time, and the time for the concurrent table index build process while the data is being restored. The time for any remaining table index builds after the restore data transfer of the last object is completed is not included.	Backup Restore Analyze_ Validate
Size (Data phase)	Size of the data processed during the phase duration	Backup

Field	Description	Job Type
		Restore Analyze_ Validate
Total Run Time	Restarted restore: Total duration of each phase	Restore

6

DSA Command-Line Interface

DSA provides a command-line interface (CLI) that allows you to carry out the same actions as in the **BAR Setup** and **BAR Operations** portlets, plus additional commands that are not available in the portlets. The CLI includes setup commands for configuring, updating, and deleting targets, sources, and the DSC repository. The CLI also includes operations commands for creating, running, and monitoring jobs.

After the DSA command-line interface package is installed, access the command line from the Linux console. You can run DSA commands from any file system location without navigating to the root directory.

Note:

If you are non-root, you must be in the **users** group to run the DSA command-line interface.

You can run commands on an ad hoc basis, as an alternative to the portlets, and you can switch between the CLI and portlets. For example, you can create jobs in the portlet and then view them using `list_jobs`. Or you can export the XML file associated with a job created in the portlet, and update the job definition using the command line. You can also develop scripts to automate DSA commands and to use with UNIX cron or other job-scheduling applications. For example, you might want to have a backup job run automatically every night at 1 a.m.

Each command has parameters that can be specified directly in the command line. In addition, many commands require that additional information is specified in an XML file. Sample XML files for system and component configuration and for job definition are located at `$BARCMDLINE_ROOT/samples`. The sample files include helpful comments and show the available settings for the corresponding command.

NOTICE

Enabling **General > Security Management** in the **BAR Setup** portlet increases security for the DSA environment. This setting requires users to provide Viewpoint credentials to execute some commands from the DSA CLI.

Accessing DSA Command Help

You can view a list and brief description of all of the commands available in the DSA command-line interface. The basic syntax and usage of commands is shown for each command.

You can also specify a command name to view more detailed information about the command, including a syntax example and a list and description of each of the parameters associated with the command.

- Type `dsc help` in the command line, and do one of the following:

Option	Description
View all commands	◦ Press **Enter** to view a list of all of the command names.

Option	Description
View a specific command	◦ Add a specific command name to view information about a particular command. ◦ Press **Enter**.

For example, for information about the `create_job` command, type `dsc help create_job` and press **Enter**. The following is displayed:

```
NAME:
create_job - Create Job Command

DESCRIPTION:
Creates a DSA job based on the file, with user modifications from the
following parameters.
User will be required to authenticate username and password to the source/target
system on the console.

EXAMPLE:
create_job -n|-name job1 -f|-file parameters.xml

PARAMETERS:
Parameter             Example          Description
n|name                job1             (Optional) Name for the job, must be unique
d|description         backup web apps  (Optional) Description of the job. For
multi-word surround by \\" must be escaped with backslash.
t|type                restore          (Optional) Type of the
job: [BACKUP,RESTORE,ANALYZE_READ,ANALYZE_VALIDATE]
b|backup_name         backupWeb1       (Optional) Backup job name(only for RESTORE
or ANALYZE jobs)
v|backup_version      60               (Optional) Backup version number(only for
RESTORE or ANALYZE jobs). Type LATEST or 0 for latest save set.
f|file                parameters.xml   XML File to upload as basis for new job.
u|user_authentication user             (Required when security management is
enabled) Supplies command with Viewpoint user
```

DSA Command Types

DSA commands can be categorized as the following two major types:

- DSA setup commands for administration and configuration allow you to create, update, and delete targets, sources, and the DSC repository. This functionality corresponds to that provided by the **BAR Setup** portlet.
- DSA operation commands for management and reporting allow you to create, execute, monitor, update, and delete jobs. This functionality corresponds to that provided by the **BAR Operations** portlet.

DSA Configuration Commands

The following table lists and briefly describes the administration and configuration commands:

Command Name	Description
`backup_application_gcp`	The `backup_application_gcp` command configures the Google Cloud Platform server in the DSA repository based on parameter files.

Command Name	Description
cleanup_ir_source_system	The cleanup_ir_source_system command clears deleted row logs in the source system. This cleans up the metadata needed for IR backup.
config_aws	The config_aws command configures the Amazon S3 Server in the DSA repository based on parameter files.
config_azure	The config_azure command configures the Microsoft Azure Server in the DSA repository based on parameter files.
config_dd_boost	The config_dd_boost command configures the DD Boost server for the DSA repository.
config_disk_file_system	The config_disk_file_system command configures the disk file system in the DSA repository.
config_fabrics	Beginning with DSA 16.10.01, the config_fabrics command defines the routing data path between TPA nodes and media servers using a parameter file. For best performance when using network fabric, make sure dsc.properties has validate.fabric=on.
config_general	The config_general command configures the general settings, based on the information contained in the parameters XML file.
config_media_servers	The config_media_servers command configures the BAR media servers.
config_nbu	The config_nbu command configures a DSA system to use Veritas NetBackup third-party software to back up and restore data.
config_repository_backup	The config_repository_backup command provides the configuration information to back up the DSC repository.
config_systems	The config_systems command configures the DSC settings for the Teradata system used for backup and restore jobs. It also sets the selector in the targeted system for ActiveMQ and establishes a JDBC connection to the target system. Nodes are configured automatically after restarting DSMain on the system.
config_target_groups	The config_target_groups command configures the target groups based on the target type and the information from the parameters file.
config_target_group_map	The config_target_group_map command configures the map between target groups when restoring to a different client configuration.
delete_component	The delete_component command deletes an existing component based on the information in the parameters.
delete_target_group_map	The delete_target_group_map command deletes a target group map for restoring to a different client configuration.

Command Name	Description
disable_component	The disable_component command disables an existing BAR component based on the component name and type.
enable_component	The enable_component command enables an existing BAR component based on the component name and type.
export_config	The export_config command exports the current XML definition for the requested BAR component.
export_job_metadata	The export_job_metadata command exports metadata of a job (job definition, save sets, and targets) based on the requested backup version. In the case of a disaster to the DSC repository, exporting and then importing job metadata enables job migration and restoration to a different DSA environment.
export_target_group_map	The export_target_group_map command exports a map between target groups for restoring to a different client configuration.
list_access_module	The list_access_module command lists available access module types for a named media server.
list_cleanup_ir_source_system	The list_cleanup_ir_source_system command lists the results of cleanup_ir_source_system.
list_components	The list_components command lists components defined and stored in the DSC repository. If a specific component is requested, that component definition is displayed. Otherwise, a list of the components matching any provided filters is displayed. Any partial component name returns all components matching the partial input. **Note:** The **Type** parameter is required.
list_consumers	The list_consumers command sends a request to ActiveMQ to provide information about all of the consumers of DSA Network Client and DSMain processes. It checks whether the selector values match the names of the DSA Network Client and DSMain systems and that the processes are running.
list_general_settings	The list_general_settings command lists all current general settings.
list_repository_backup_settings	The list_repository_backup_settings command lists all current repository backup settings.
list_reset_ir_target_system	The list_reset_ir_target_system command lists the results of reset_ir_target_system.
list_target_group_map	The list_target_group_map command lists the maps between target groups for restoring to a different client configuration.

Command Name	Description
reset_ir_target_system	The reset_ir_target_system command enables incremental restore on the target system.

DSA Operating Commands

The following table lists and briefly describes the management and reporting commands:

Command Name	Description
abort_job	The abort_job command aborts an actively running job, a job in the queue, or a job that is not responding.
activate_job	The activate_job command activates a retired job, so that it is available.
build_restore_objects	The build_restore_objects command enables read and write on all the objects in the restore job without running the restore and / or builds the secondary indexes of the object.
consolidate_job_logs	The consolidate_job_logs command uploads all logs for a completed job to a centralized location. If the job is running, the command is rejected.
create_job	The create_job command creates a job based on the values you specify for parameters in the command line or in the XML file. Parameter values you enter in the command line supersede any value you enter for those parameters in the parameters XML file.
delete_job	The delete_job command deletes a job and any data associated with it from the DSC repository. Any logs and job history are deleted and cannot be restored. Any backup save sets created for the job that exist on devices managed by third-party solutions must be deleted manually using the interface for that solution. **Note:** This command only deletes new or retired jobs.
delete_save_sets	The delete_save_sets command deletes the backup save sets for a backup job using a single or multiple job execution ID. It does not delete save sets for repository jobs.
export_job	The export_job command exports the current XML definition for the requested job.
export_job_metadata	The export_job_metadata command exports metadata of a job (job definition, save sets, and targets) based on the requested backup version. In the case of a disaster to the DSC repository, exporting and then importing job metadata enables job migration and restoration to a different DSA environment.
import_job_metadata	The import_job_metadata command imports metadata of a job (job definition, save sets, and targets) to the specified directory. In the case of a disaster to the

Command Name	Description
	DSC repository, exporting and then importing job metadata enables job migration and restoration to a different DSA environment.
job_status	The job_status command gets the latest status for a job with the given name and displays it on the screen. If the job is running, a detailed status message is displayed, unless the job is collecting cumulative status on a restarted restore job. In that case, the status is not available until the job is complete. If the job is not running, the status of the last run for that job is displayed.
job_status_log	The job_status_log command displays the latest status log for a job with the given name if the job is running. If the job is not running, the status log for the last run job is displayed.
list_build_ restore_objects	The list_build_restore_objects command lists the status of the build_ restore_objects command.
list_consumers	The list_consumers command sends a request to ActiveMQ to provide information about all of the consumers of DSA Network Client and DSMain processes. It checks whether the selector values match the names of the DSA Network Client and DSMain systems and that the processes are running.
list_jobs	The list_jobs command lists jobs defined and stored in the DSC repository. If a specific job is requested, that job definition is displayed. Otherwise, a list of job names matching any provided filters is displayed. If no parameters are provided, a list of jobs (excluding backup repository jobs) is displayed.
list_ query_backupids	The list_query_backupids command lists the results of the query returned from the query_backupids command.
list_recover_ backup_metadata	The list_recover_backup_metadata command lists the overall status and individual media server status of the recover_backup_metadata command.
list_save_sets	The list_save_sets command lists all valid save sets for a specified job name.
object_release	The object_release command releases all objects that are currently locked by a job. It does not release objects for new, running, or queued jobs.
object_ release_output	The object_release_output command lists the status of objects requested for release using object_release. It provides a more accessible information to release locked objects using DSC Client. It can be run for all jobs that perform locking on their objects.
purge_jobs	The purge_jobs command can be used for cleaning up DSC repository when resources are not released after aborting jobs. The command will abort any jobs and purge the resources used by any incomplete jobs. When no job name is provided, it purges resources used by all jobs in BAR and BARBACKUP.
query_backupids	The query_backupids command queries third-party software for information needed for duplication.

Command Name	Description
`recover_backup_metadata`	The `recover_backup_metadata` command queries the third party media to recover backup metadata and rebuild the backup job plan in the case of a disaster to the DSC repository. The command can only run on repository backup jobs with no save sets.
`retire_job`	The `retire_job` command retires an active job. It does not retire a running or queued status job, or one that is already in the retired state.
`run_job`	The `run_job` command runs a job as soon as all necessary resources are available. The DSC system limit is set at 20 concurrent running jobs, and up to 20 jobs can queue over that limit. The DSC also queues jobs if the defined target media is not available before the job starts.
`run_repository_job`	The `run_repository_job` command runs a job in the DSC repository.
`set_status_rate`	The `set_status_rate` command configures the status update rate between DSC and the media servers or Teradata systems.
`system_health`	The `system_health` command lists ActiveMQ system health information such as the memory limit and memory usage for the main DSA queues.
`sync_save_sets`	The `sync_save_sets` command sends a sync request to all NetBackup clients that have save sets older than the dataset.retention.days value configured in `dsc.properties`. If the save sets are expired on the NetBackup side and no restore or analyze job is configured to use them, DSC deletes them from the DSC repository. If it is the last save set and a restore or analyze job is configured to use it, it will not be deleted until there is a newer save set. No scheduled or ad hoc jobs can run until the deletion completes.
`update_job`	The `update_job` command updates an existing DSA job based on the information from the command line parameters or the XML file if provided. Parameter values specified in the command line supersede values entered for the same parameters in the XML file.

DSA Configuration: Command Line

DSA configuration commands enable you to set up the DSC repository and system components. These commands can be used as an alternative to using the **BAR Setup** portlet. DSA configuration commands enable you to perform a wide range of configuration activities, including the following:

- Setting general DSC repository settings
- Backing up the DSC repository, and exporting and importing repository backup information
- Configuring systems, network fabric, media servers, backup solutions (for example, third party applications), and target groups
- Enabling and disabling systems and target groups
- Viewing DSC repository and component information

The procedures for using the DSA configuration commands are described in detail in the following sections.

Systems and Nodes

You can add, configure, and set stream limits for systems in the **BAR Setup** portlet and by using DSA setup commands from the DSA command-line interface. After you enable configured systems, they are available for backup and restore jobs in the **BAR Operations** portlet and for DSA operation commands.

Note:

Beginning with DSA 16.10, nodes are configured through autodiscovery. You can view the node information but changing the node configuration is not allowed.

Configuring a System

Prerequisite:

Before using this command, you must prepare an XML file that contains the necessary system configuration information, including the system name and limits on the number of streams per node. For more information, see config_systems.

The `config_systems` command configures the DSC settings for the Teradata system used for backup and restore jobs. It also sets the selector in the targeted system for ActiveMQ and establishes a JDBC connection to the target system. Nodes are configured automatically after restarting DSMain on the system.

Even though the repository backup system is already pre-configured, you must rerun `config_systems` to set up the system selector and AMP discovery before the system can be enabled for use.

1. Type `dsc config_systems` followed by the parameters, and press **Enter**.
2. Type the source user name and password, and press **Enter**.

Media Servers

Media servers manage data during system backups and restores. Media servers are made available to your BAR environment as soon as the DSA software is installed and running. Use the **BAR Setup** portlet to add or delete media servers to the BAR configuration, and assign media servers to target group configurations.

Adding or Updating a Media Server

The `config_media_servers` command configures the BAR media servers.

To use the `config_media_servers` command, you must specify the XML file that contains the necessary configuration information, including the media server name, third-party client, port and IP address.

1. Type `dsc config_media_servers -f` followed by the full file path, and press **Enter**.

Fabrics

Beginning with DSA 16.10.01, you can define network fabric. If a network fabric is defined, it replaces a logical network mask when a job plan generates data paths between AMPs and BARNC processes acting as media servers.

- The fabric can include all the nodes and their paths to one or more media servers.
- Fabric is used only for backup, restore, and analyze_validate jobs.
- You can define multiple fabrics but each must be unique. For example, System A can have two fabrics for two target systems B and C, fabricAB and fabricAC. The information is saved into the repository and can be edited if new media servers or nodes are added.
  - All media servers in a target group should be available in a single fabric.
  - The correct fabric is chosen based on the media servers in the target group.
  - Update `dsc.properties` for the number of fabrics allowed per Teradata system: `maxlimit.fabric=`*number-fabrics*. Default is 64.
- When fabrics are used, Teradata recommends not editing media server IP addresses.

Related Information:

Define Data Paths Using Network Fabric
Network Fabric Considerations

Configuring Network Fabrics: Command Line

The config_fabrics command defines the routing data path between the TPA nodes and media servers.

To use the config_fabrics command you must specify the XML file that contains the necessary configuration information.

1. Restart ClientHandler to discover all changed IP addresses.
   `/etc/init.d/clienthandler restart-hwupgrade`
2. Type `dsc config_fabrics -f` *full_file_path* and press **Enter**.

Related Information:

config_fabrics

Backup Solutions

Backup solutions are options for transferring data between a storage device and a database system. Available solutions include third-party server software and cloud-based solutions as well as local storage. Configuration options depend on the backup solution. Choose Disk File System for Spectrum Protect.

Adding or Updating a Disk File System

The `config_disk_file_system` command configures the disk file system in the DSA repository.

To use the `config_disk_file_system` command, you must specify the XML file that contains the necessary configuration information, including the file system path and limit for the maximum number of open files.

When adding or updating a file system using the command line, first export the existing disk file configuration, which includes all the current file systems, to the configuration xml file. After exporting the configuration, add a new or update an existing file system and its parameters within the exported configuration xml file. After saving the configuration xml file, run the `config_disk_file_system` command to configure all the file systems using this configuration xml file.

- Type `dsc config_disk_file_system -f` followed by the full file path, and press **Enter**.

Adding or Updating a DD Boost Server

The `config_dd_boost` command configures the DD Boost server for the DSA repository.

To use the `config_dd_boost` command, you must specify the XML file that contains the necessary configuration information, including the server name, ip address, media type, and storage unit names and limit for the maximum number of open files.

- Type `dsc config_dd_boost -f` followed by the full file path, and press **Enter**.

Adding or Updating a NetBackup Server

When using a NetBackup server to back up and restore data, you must add and configure it using the **BAR Setup** portlet.

Note:

System names and open file limits are tied to media servers during the target group configuration.

---

1. Under **DSC Servers**, select your DSC server.
2. Under **Categories**, select **Backup Solutions**.
3. From the **Solutions** list, click **NetBackup**.
4. Click + next to **Servers**.
5. Under NetBackup Details, enter the following:

Nickname	Alphanumeric characters and underscores, but no spaces, max characters 32
Server Name (IP/DNS)	Server IP address or DNS

Policy Name	Policy class name, max characters 128, alphanumeric and underscores
Storage devices	1 - 999

6. Click + after **Storage devices** to add more policies.
7. Click **Apply**.

Target Groups

Target groups are locations used for storing backup data. DSA administrators create target groups, and assign media servers and devices. Target groups are then accessible to BAR backup jobs.

You need at least two target groups: a replication target group for the repository backup (DSA configuration and metadata) and one or more for data backups.

After a backup job has run to completion, you can create a BAR restore job to restore data using the same target group as the backup job. You can also create a target group map (restore group), which allows a BAR restore job to restore data from a different target group.

Virtual Target Groups

The **BAR Setup** portlet does not manage virtual target groups, which are target groups imported from a different DSC. If you need to manage a virtual target group, use the command line.

Adding or Updating a Target Group Using the Command Line

Prerequisite:

Configure a third-party application and media server before adding a target group.

The `config_target_groups` command configures the target groups based on the target type and the information from the parameters file.

To use the `config_target_groups` command, you must specify the XML file that contains the necessary configuration information, including the target group name, entity, BAR media server, and policy class information.

- Type `dsc config_target_groups` followed by the parameters, and press **Enter**.
  For more information, see config_target_groups.

Enabling a System or Target Group

Prerequisite:
A system or target group must be configured in your BAR environment before it can be enabled. Use the BAR Setup portlet or a configuration file to configure the component.

The `enable_component` command enables an existing BAR component based on the component name and type.

- Type `dsc enable_component` followed by the parameters, and press **Enter**.
  For more information, see enable_component.

Disabling a System or Target Group

Prerequisite:
The system or target group cannot be in use when you plan to disable it.

The `disable_component` command disables an existing BAR system or target group.

- Type `dsc disable_component` followed by the parameters, and press **Enter**.
  For more information, see disable_component.

Adding a Target Group Map

Prerequisite:
Configure a target group before adding a target group map.

Target group mapping is maintained in an XML file to designate the new configuration for the restore.

The `config_target_group_map` command configures the map between target groups when restoring to a different client configuration.

Sample XML files are located at `$BARCMDLINE_ROOT/samples/sample_target_mapping`. They include helpful comments and show the available settings for the corresponding commands.

1. Open the XML file containing the target mapping configuration information.
   A sample map, `sample_target_`*`targettype`*`_map.xml`, is supplied in the DSC sample library. Target type can be nbu, ddboost, fs, aws, azure, or gcp.

2. Specify target group map tags. The following example shows the tags for nbu:

Target Group Map Tag	Description
master_source_target_name	The backup target group name.
master_source_dsc_id	The ID of the source DSC server.
master_dest_target_name	The restore target group name.
target_group_maps	A sub target grouping consisting of a backup and restore configuration pair.
source_mediaserver_name	The backup media server.
source_policy_class_name	The third party backup application policy associated with the backup media server.
dest_mediaserver_name	The restore media server.
dest_policy_class_name	The third party backup application policy associated with the restore media server.

3. For example, to map a virtual target group to an existing physical target group in the destination domain:
   `dsc config_target_group_map -f cbar6.xml -V`
   For more information, see config_target_group_map.

Exporting a Target Group Map

The `export_target_group_map` command exports a map between target groups for restoring to a different client configuration.

- Type `dsc export_target_group_map` followed by the parameters, and press **Enter**.
  For more information, see export_target_group_map.

Deleting a Target Group Map

The `delete_target_group_map` command deletes a target group map for restoring to a different client configuration.

- Type `dsc delete_target_group_map` followed by the parameters, and press **Enter**.
  For more information, see delete_target_group_map.

Deleting a Component

The `delete_component` command deletes an existing component based on the information in the parameters.

To use the `delete_component` command, you must specify the component name and type. You do not need to specify the component name for disk_file_system types.

You can delete a system, media server, NetBackup server, disk file system, DD Boost server, Amazon S3 storage account, Azure Blob storage account, Google Cloud Account or a target group, except under the following conditions:

- System marked for repository backup
- System in use by a job
- Media server in use by a target group
- NetBackup server in use by a target group
- Target group in use by a job
- Target group in use by a target group map
- Target group in use for repository backups, unless it is a repository_replication target group
- Policy used by a target group

Note:

Beginning with the DSA 16.10 release, nodes are autodiscovered and cannot be manually deleted.

- Type `dsc delete_component` followed by the parameters, and press **Enter**.
  For more information, see delete_component.

Viewing Configuration Information

The list commands allow you to view component configurations, DSA general settings, active jobs, retired jobs, DSC jobs, job history, repository backup settings, valid save sets, and target group maps.

- Type the list command followed by the parameters, if applicable, and press **Enter**.

List Command	Description
`list_access_module`	The `list_access_module` command lists available access module types for a named media server. For more information, see list_access_module.
`list_components`	The `list_components` command lists components defined and stored in the DSC repository. If a specific component is requested, that component definition is displayed. Otherwise, a list of the components matching any provided filters is displayed. Any partial component name returns all components matching the partial input. **Note:** The **Type** parameter is required. For more information, see list_components.
`list_general_settings`	The `list_general_settings` command lists all current general settings. There are no parameters associated with this command.

List Command	Description
`list_jobs`	The `list_jobs` command lists jobs defined and stored in the DSC repository. If a specific job is requested, that job definition is displayed. Otherwise, a list of job names matching any provided filters is displayed. If no parameters are provided, a list of jobs (excluding backup repository jobs) is displayed. For more information, see list_jobs.
`list_job_history`	The `list_job_history` command lists the complete history of the job you specify, including the type of system used during the job execution. If you do not specify a job, the history of all jobs in the DSC repository are listed. For more information, see list_job_history.
`list_repository_backup_settings`	The `list_repository_backup_settings` command lists all current repository backup settings. There are no parameters associated with this command.
`list_save_sets`	The `list_save_sets` command lists all valid save sets for a specified job name. For more information, see list_save_sets.
`list_target_group_map`	The `list_target_group_map` command lists the maps between target groups for restoring to a different client configuration. For more information, see list_target_group_map.

Exporting DSA Component Configuration

The `export_config` command exports the current XML definition for the requested BAR component.

- Type `dsc export_config` followed by the parameters, and press **Enter**.
  For more information, see export_config.

Managing the DSC Repository

DSA configuration settings and job metadata are stored in the Data Stream Controller (DSC) repository. You can automate a repository backup or initiate the backup manually.

NOTICE

Before you can recover the DSC Repository, a DSC repository backup job and an export of the repository backup configuration must have been completed successfully at least once.

The export of the repository backup configuration can only be performed using the DSA command line. See Saving the Repository Backup Configuration File for more information.

Failure to perform a successful repository backup and an export of the repository backup configuration results in an unrecoverable DSC repository in the case of a complete disaster.

A repository backup job backs up your DSC data to a replication target group. Any running DSC repository job (backup, restore, or analyze) prevents jobs from being submitted and DSA configuration settings from being changed.

Conversely, a database backup, restore, or analyze job prevents a repository backup job from running.

Configuration settings and DSC metadata can be restored to the DSC repository from a storage device.

NOTICE

Do not abort a DSC repository restore job while the job is in progress. If you abort a DSC repository restore job while the job is in progress or if the restore job fails, DSC triggers a command to restore all repository tables to their initial state, which is an empty table. The current data in the DSC repository is lost and must be recovered using disaster recovery.

Scheduling and Configuring a Repository Backup

Prerequisite:

Before scheduling a DSC repository backup job, configure a remote target group for the job. Use the `sample_config_repository_backup.xml` file to identify the repository backup target group and have scheduling parameters ready.

The `config_repository_backup` command allows you to schedule a DSC repository backup job.

Note:

Before you can recover the DSC Repository, a DSC repository backup job and an export of the repository backup configuration must have completed successfully at least once.

1. Type `dsc config_repository_backup -f` *configRepositoryBackup*`.xml` and press **Enter**.
   For more information, including the xml file template, see config_repository_backup.

Backing Up the DSC Repository with the Command Line

Prerequisite:

Before backing up the DSC repository, run the `config_repository_backup` command.

The `run_repository_job` command allows you to back up the DSC repository.

Note:

Before you can recover the DSC Repository, a DSC repository backup job and an export of the repository backup configuration must have completed successfully at least once.

1. Type `dsc run_repository_job -t backup`, and press **Enter**.

   Note:

   The `run_repository_job` command can only be initiated if no operational jobs are running.

   For more information, see run_repository_job.

Saving the Repository Backup Configuration File

When you back up the repository to the landing zone one of the files that is created is `boostrap_`*`backupid`*`.out`.

1. After backing up the repository to the landing zone, retrieve the backupid using one of these methods:
   `dsc save_sets_replication_status`
   From inside the landing zone: `ls -lrt` retrieves the backup ID of the two most recent files.
2. Locate the `boostrap_`*`backupid`*`.out` file and store it in a safe place.

Restoring the DSC Repository with the Command Line

Prerequisite:

A valid DSC repository backup job save set must exist before a DSC repository restore job can be run.

The `run_repository_job` command allows you to restore the DSC repository.

1. Type `dsc run_repository_job -t restore`, and press **Enter**.
   For more information, including other parameters, see run_repository_job.

Rules for Aborting a DSC Repository Job

NOTICE

Follow these rules to avoid data loss.

Before aborting a DSC repository job, make sure of the following:

- Make sure you have an exported repository backup job configuration file, which is required to recover your repository.
- If you abort a DSC repository restore job while the job is in progress, DSC metadata will be corrupted.

  DSC triggers a command to restore all repository tables to their initial state, which is an empty table. This results in loss of data unless you follow the disaster recovery procedure to recover the DSC repository.

For more information, see Restoring the Repository (Portlets) and Restoring the Postgres DSC Repository When the Boostrap File Is Available: Command Line / Restoring the DSC Repository with the Command Line.

Planning a Job: Command Line

Creating a job includes choosing several options. There are different considerations before creating each job type. The following lists some of the considerations.

Backup Jobs

Backup jobs archive objects from a source system to a target group. Target groups are defined by a BAR administrator in the **BAR Setup** portlet or from the command-line interface.

The BAR Operations portlet allows you to migrate the object list from an existing ARC script into a backup job. Objects in that list that exist in the specified source system are automatically selected in the object browser when a new job is created from the migrated ARC script.

When you run a backup job for the first time or when you change the target group for a backup job, all data from the specified objects is archived. After this initial full backup, you may choose from the following backup types:

- Full: Archives all data from the specified objects.
- Delta: Archives only the data that has changed since the last backup operation.
- Cumulative: Archives the data that has changed since the last full backup was run.

Restore Jobs

Restore jobs are based on successful executions of backup jobs and can only be created for a backup job that has successfully run to completion.

You can define a restore job to always restore the latest version of a backup save set or you can specify a save set version. A target Teradata system must be selected in order to define the restore job. By default, all objects from the save set are included in the restore job but the selections can be modified.

Analyze Jobs

An analyze job can employ either a read-only or validate analysis method for each job. An analyze read-only job reads the data from the media device to verify that reads are successful. An analyze validate job sends the data to the AMPs, where it is interpreted and examined but not restored.

You also need to specify a save set version from a successful backup job run.

Creating or Updating a Job: Command Line

The `create_job` command creates a job based on the values you specify for parameters in the command line or in the XML file. Parameter values you enter in the command line supersede any value you enter for those parameters in the parameters XML file.

You can also edit an existing job by using the `update_job` command.

Before you create a restore or analyze type job, a backup job must have been completed successfully or run with a warning. To create an incremental restore job, the backup job must have been created with incremental restore enabled.

1. Type `dsc create_job` or `dsc update_job` followed by the parameters, and press **Enter**. See create_job or update_job for syntax and xml tags, including `enable_incremental_restore`, `backup_version` and `next_incremental_restore`.
2. Type the user name and password, and press **Enter**.

Automatically Retiring a Job

You can set jobs to automatically retire by enabling **auto_retire** in the XML file containing the job definition.

- Open the XML file containing the job definition and set the following:

Parameter	Description
auto_retire	`true` **Note:** False is the default value and means the job does not retire automatically.
retire_value	Integer to represent the number of days or weeks until the job is retired. Maximum of three digits.
retire_units	`days` or `weeks`

Including Objects

You can include database objects for any backup, restore, or analyze_validate job by specifying an object under **objectlist** in the XML file containing the job definition.

1. Open the XML file containing the job definition information.
2. Under **objectlist**, specify:

Option	Description
objectinfo	Contains the information on all objects for the job.
object_name	The name of the object. The name can have a maximum of 128 characters.
object_type	The type of object. The accepted values are: AGGREGATE_FUNCTION AUTHORIZATION COMBINED_AGGREGATE_FUNCTIONS CONTRACT_FUNCTION DATABASE EXTERNAL_PROCEDURE FUNCTION_ALIAS [Analytics Database 16.20 Feature Update 1] GLOP_SET HASH_INDEX INSTANCE_OR_CONSTRUCTOR_METHOD JAR JOIN_INDEX MACRO NO_PI_TABLE ORDERED_ANALYTICAL_FUNCTION QUEUE_TABLE SERVER_OBJECT STANDARD_FUNCTION STATISTICAL_FUNCTION STORED_PROCEDURE TABLE TABLE_FUNCTION TABLE_OPERATOR TRIGGER USER USER_DEFINED_METHOD USER_DEFINED_DATA_TYPE USER_INSTALLED_FILE VIEW
parent_name	The name of the object parent. The name can have a maximum of 128 characters. **Note:** All objects other than databases and users are required to have a parent name.
parent_type	[Optional] The type of parent. The accepted values are: DATABASE USER

The following XML sample file excerpt shows how to include the VIEW1 object in the DB1 database.

```
<objectlist>
    <objectinfo>
        <object_name>VIEW1</object_name>
        <object_type>VIEW</object_type>
        <parent_name>DB1</parent_name>
        <parent_type>DATABASE</parent_type>
    </objectinfo>
```

Excluding Objects

You can exclude objects from a backup or restore job by amending **excludeobjectinfo** in the XML file containing the job definition.

The following restrictions are true for backup or data migration jobs:

Version of DSA	Analytics Database / Teradata Database	Restricted Objects
Prior to DSA 16.20	Any	Cannot exclude individual: VIEW, MACRO, TRIGGER, JOIN_INDEX, HASH_INDEX
DSA 16.20	Teradata Database prior to 16.20	Cannot exclude individual: VIEW, MACRO, TRIGGER, JOIN_INDEX, HASH_INDEX
	Analytics Database 16.20 or later	Cannot exclude individual: JOIN_INDEX or HASH_INDEX Use the job level option `skip_joinhash_index` to skip join and hash index. See create_job.

1. Open the XML file containing the job definition information.
2. Under **excludeobjectinfo**, specify the object name and object type that you want to exclude.
   For example, to exclude tables T1 and T2 from a job that restores database ABC:

```
<objectinfo>
    <object_name>ABC</object_name>
    <object_type>DATABASE</object_type>
    <parent_name>SYSTEMFE</parent_name>
    <parent_type>DATABASE</parent_type>
    <!--    Optional    -->
    <exclude>
        <excludeobjectinfo>
            <object_name>T1</object_name>
            <object_type>TABLE</object_type>
        </excludeobjectinfo>
        <excludeobjectinfo>
            <object_name>T2</object_name>
```

```
            <object_type>TABLE</object_type>
        </excludeobjectinfo>
    </exclude>
</objectinfo>
```

Renaming Tables

You can rename tables in a restore job by adding **rename_to** to the table in the XML file containing the job definition.

1. Open the XML file containing the job definition information.
2. Under **objectinfo**, identify the table to rename and add the new table name inside **rename_to**. In the following example, table T1 in the ABC database is restored and renamed as T2:

```
<objectinfo>
    <object_name>T1</object_name>
    <object_type>TABLE</object_type>
    <parent_name>ABC</parent_name>
    <parent_type>DATABASE</parent_type>
    <object_attribute_list>
        <rename_to>T2</rename_to>
    </object_attribute_list>
</objectinfo>
```

Renaming a Database

You can rename a database in a restore job by adding **map_to** to the table in the XML file containing the job definition. This option restores the database to the destination system using a new name for the database.

1. Open the XML file containing the job definition information.
2. Under **objectinfo**, identify the database to rename and add the new database name inside **map_to**. In the following example, the objects under database ABC are copied to database NEWBAR:

```
<objectinfo>
   <object_name>ABC</object_name>
   <object_type>DATABASE</object_type>
   <parent_name>SYSTEMFE</parent_name>
   <parent_type>DATABASE</parent_type>
   <object_attribute_list>
       <map_to>NEWBAR</map_to>
   </object_attribute_list>
</objectinfo>
```

Mapping to a Different Database

You can map an object to a different database so that it is restored to the destination system under a different user or database than was originally designated for the backup job. You do this using the **map_to** tag in the XML file containing the job definition.

1. Open the XML file containing the job definition information.
2. Under **objectinfo**, identify the object, and add the new database name inside **map_to**.
   The following example restores table T1, originally backed up from database ABC, to database NEWBAR:

```
<objectinfo>
   <object_name>T1</object_name>
   <object_type>TABLE</object_type>
   <parent_name>ABC</parent_name>
   <parent_type>DATABASE</parent_type>
   <object_attribute_list>
      <map_to>NEWBAR</map_to>
   </object_attribute_list>
</objectinfo>
```

Note:
You can also use **map_to** in the **job_options** section.

Changing Job Options

You can change job options for any backup, restore, or analyze job by amending the job options in the XML file containing the job definition.

1. Open the XML file containing the job definition information.
2. Under `<job_options>`, list the necessary job options and place the appropriate value within the job tags.

   Note the following restrictions on job options:

   - If single_object_locking is true and online is true, the backup job fails.
   - If single_object_locking is true and data_phase is DICTIONARY, the backup job fails.
   - If single_object_locking is true, a DBC-only backup job is rejected.

   The following example shows the job options section in the XML file for a restore job:

```
<job_options>
    <!--    Optional, true/false  -->
    <data_phase>DATA</data_phase>
```

```
    <!--    Optional, true/false  -->
    <enable_temperature_override>true</enable_temperature_override>
    <!--    Optional, enum type, COLD/WARM/HOT  -->
    <temperature_override>HOT</temperature_override>
    <!--    Optional, enum type, DEFAULT/ON/OFF  -->
    <block_level_compression>ON</block_level_compression>
    <!--    Optional, true/false  -->
    <disable_fallback>false</disable_fallback>
    <!--    Optional, max 2048 characters  -->
    <query_band>queryband</query_band>
    <!--  Optional, starts with upper case ( Error/Info/Debug/Warning )  --
>
    <dsmain_logging_level>Debug</dsmain_logging_level>
    <!--  'reblock' - Optional, accepted values: true/false  -->
    <!--  Note: Only useful for RESTORE jobs.  -->
    <reblock>true</reblock>
    <!--    Optional, true/false  -->
    <run_as_copy>true</run_as_copy>
<job_options>
```

DSA Command-Line Interface Job Options

You can select and define job options for backup, restore, and analyze jobs by specifying the values for the options in the XML file containing the job definition.

Important:

The XML files in the sample library and the files produced by the **BAR Operations** portlet have the XML in a specific order. For the jobs to run correctly, do not change the order of the XML within the files.

Job Option	Description	Job Type
online	Determines the type of backup to perform. **false** Default. Backs up everything associated with each specified object while the database is offline. No updates can be made to the objects during the backup job run. **true** Backs up everything associated with each specified object and initiates an online archive for all objects being archived. The online archive creates a log that contains all changes to the objects while the archive is prepared.	Backup

Job Option	Description	Job Type
	Note: A DBC only backup job cannot be run as online.	
nosync	Determines where synchronization is done for the job. Only available for online backup jobs. **false** Default. Synchronization occurs across all tables simultaneously. If you try to run a job that includes objects that are already being logged, the job aborts. **true** Allows different synchronization points. If you try to run a job that includes objects that are already being logged, the job runs to completion and returns a warning.	Backup
data_phase	Determines the type of backup to perform. **DATA** Default. Performs a full backup. **DICTIONARY** Performs a dictionary-only backup.	Backup
enable_ backup_for_IR	Establishes that this backup is available for incremental restore. **false** Default. Backup is standard CBB backup. **true** Backup save set is available for incremental restore.	Backup
single_object_ locking	Determines when the locking of individual tables occurs during backup. **false** Default. All tables are locked initially when the database is backed up. **true** Tables are locked individually while backing up the database.	Backup
enable_ temperature_ override	Optional setting that pertains to restore jobs only. **true** The **temperature_override** value is applied when restoring the data. **false** The temperature from the backup is applied.	Restore
temperature_ override	Determines the temperature at which data is restored.	Restore

Job Option	Description	Job Type
	DEFAULT Default. This data is restored at the default temperature setting for the system. **HOT** This data is accessed frequently. **WARM** This data is accessed less frequently. **COLD** This data is accessed least frequently.	
block_level_ compression	Defines data compression used. **DEFAULT** Default. Applies same data compression as the backup job if allowed on the destination system. **ON** Compress data at the block level if allowed on the destination system. **OFF** Restore the data blocks uncompressed.	Restore
disable_fallback	Fallback protection means that a copy of every table row is maintained on a different AMP in the configuration. Fallback-protected tables are always fully accessible and are automatically recovered by the system. [Teradata Database 16.20 Update 1 and later] Always true. Changes are ignored. [Prior versions Teradata Database] Default is false. Must be entered in lowercase letters. **Note:** The disable_fallback option only takes effect when the run_as_copy value is true.	Restore
query_band	Allows tagging of sessions or transactions with a set of user-defined name-value pairs to identify where a query originated. These identifiers are in addition to the current set of session identification fields, such as user ID, account string, client ID, and application name. **Note:** Valid query band values are defined on the database.	Backup Restore
dsmain_ logging_level	Determines the types of messages that the database job logs. **error** Default. Enables minimal logging. Provides only error messages.	Backup Restore

Job Option	Description	Job Type
	warning Adds warning messages to error message logging. **info** Provides informational messages with warning and error messages to the job log. **debug** Enables full logging. All messages, including Debug, are sent to the job log.	
skip_joinhash_index	Determines whether join and hash indexes are skipped or excluded from a backup job. **false** Default. Indexes are backed up. **true** Join and hash indexes are skipped. Note the following for restore jobs: • If restore job is set to true and the backup contains join/hash indexes, the restore job fails. • If restore job is set to false and the backup job was set to true, the restore receives a warning.	Backup Restore Data Migration
reblock	Determines level of inserts, either row or block level. This option is valid only for the same configuration restore. Different configuration restore always use row level inserts. **false** Default. The restore uses block level inserts. **true** The restore uses row level inserts, forcing reblocking when tables must be restored matching the destination system default block size. • Required when the backup is restored to a different AMP configuration, hash function, hash bucket, row format or block alignment. • Optional when the backup is restored from a system with different block size than the destination system and the block size of the restored data objects must match the destination system block size.	Restore
run_as_copy	Allows restore to run as a copy. • Default value is false. • If true, the restore runs as a copy.	Restore
skip_archive	[Optional] true or false. Skip archive during backup or restore job in case of error in access rights.	Backup Restore

Job Option	Description	Job Type
	Supported in Analytics Database 16.20 and later / Teradata Database 15.10.01 and later.	
skip_stats	[Optional] true or false. Skips collecting statistics during a backup job. Skips restoring stats during a restore job. Supported in Analytics Database 16.20 and later / Teradata Database 16.0 and later.	Backup Restore
config_map_name	[Optional] Maximum of 128 characters. Hash map name to restore to. When applied to the DATABASE/USER level, only objects under DATABASE/USER and object type of TABLE, NO PI TABLE, or QUEUE TABLE are applied with the new map. If you have defined map_to, rename_to, or config_map_name for table level objects, this affects the results of a database level restore job definition. • If database is selected and includeAll attribute, map_to, or config_map_name is *not* set at the database level, the DSC expands objects under the database and treats the restore job as table level restore. • If database is selected and includeAll attribute, map_to, or config_map_name is set at the database level, the table level definitions are ignored, and the DSC treats the restore job as database level restore.	Restore
track_empty_tables	[Optional] true or false. Tracks empty tables during a backup job. Supported in Analytics Database 16.20 and later / Teradata Database 16.0 and later.	Backup
concurrent_bld_per_tbl	[Optional] Range is 1 to 5. Number of index and fallback subtables that can be built concurrently per table during restore. Supported in Analytics Database 16.20 and later / Teradata Database 16.0 and later.	Restore
map_to	[Optional] Map one database to a new database or map a table level object under multiple databases to a new database. Applies to all objects in the restore job. Supported in Analytics Database 16.20 and later / Teradata Database 16.0 and later If you have defined map_to, rename_to, or config_map_name for table level objects, this affects the results of a database level restore job definition. • If database is selected and includeAll attribute, map_to, or config_map_name is *not* set at the database level, the DSC expands objects under the database and treats the restore job as table level restore. • If database is selected and includeAll attribute, map_to, or config_map_name is set at the database level, the table level definitions are ignored, and the DSC treats the restore job as database level restore.	Restore
rename_to	[Optional] Maximum of 128 characters. Rename a table to this name. If you have defined map_to, rename_to, or config_map_name for table level objects, this affects the results of a database level restore job definition.	Restore

Job Option	Description	Job Type
	• If database is selected and includeAll attribute, map_to, or config_map_name is *not* set at the database level, the DSC expands objects under the database and treats the restore job as table level restore. • If database is selected and includeAll attribute, map_to, or config_map_name is set at the database level, the table level definitions are ignored, and the DSC treats the restore job as database level restore.	
stream_softlimit	[Optional] Range is 1-*max* soft limit of the stream soft limit, where *max* is the number of AMPs per node in the system. Overrides the system level stream soft limit.	Backup Restore Analyze_Validate
include_DBC_data	Establishes that this backup/restore is available for incremental restore with DBC. • False: Default. Backup does not include DBC tables. • True: Backup and Restore includes the DBC tables while execution.	Backup, Restore

Running a Job

- Type `dsc run_job` followed by the parameters, and press **Enter**. See run_job.

Viewing Job Status: Command Line

The `job_status` command gets the latest status for a job with the given name and displays it on the screen. If the job is running, a detailed status message is displayed, unless the job is collecting cumulative status on a restarted restore job. In that case, the status is not available until the job is complete. If the job is not running, the status of the last run for that job is displayed.

- Type `dsc job_status` followed by the parameters, and press **Enter**. See job_status.
  See also job_status_log.

Considerations for Aborting a Job

Reasons to abort a job after it has been submitted include:

- You forgot to include an object for backup in the job definition XML file.
- You see that data will fill the media if the job is completed.

NOTICE

If you abort a restore job of a DSC repository backup while the job is in progress, the DSC metadata is corrupted. DSC triggers a command to restore all repository tables to their initial state, which is an empty table. The current data in the DSC repository is lost and must be recovered using disaster recovery.

You can use the `abort_job` command to abort an actively running job or a job in the queue. An abort command creates two states for the job. While the job state is aborting, devices are being released for other jobs to use. After the job has reached an aborted state, the job releases DSA stream resources and the DSC job slot.

If you abort an actively running job, DSA does not create a save set. Any backed up files are rolled back to the state they were in following the last complete, successful run. Third-party backup management software does not keep records of jobs that are not run successfully.

Aborting a Job Using the Command Line

The `abort_job` command aborts an actively running job, a job in the queue, or a job that is not responding.

If the `abort_job` command is given for a job in queue to run, the job is removed from the queue. If the job is running when the request is sent to abort the job, it can take several minutes for the job to completely stop.

- Type `dsc abort_job` followed by the parameters, and press **Enter.** See abort_job.

Exporting a Job

The `export_job` command exports the current XML definition for the requested job.

- Type `dsc export_job` followed by the parameters, and press **Enter**. See export_job.

Retiring a Job

The `retire_job` command retires an active job. It does not retire a running or queued status job, or one that is already in the retired state.

- Type `dsc retire_job` followed by the parameters, and press **Enter**. See retire_job.

Activating a Job Using the Command Line

The `activate_job` command activates a retired job, so that it is available.

- Type `dsc activate_job` followed by the parameters, and press **Enter.** See activate_job.

Deleting a Job: Command Line

The `delete_job` command deletes a job and any data associated with it from the DSC repository. Any logs and job history are deleted and cannot be restored. Any backup save sets created for the job that exist on devices managed by third-party solutions must be deleted manually using the interface for that solution.

Note:

This command only deletes new or retired jobs.

- Type `dsc delete_job` followed by the parameters, and press **Enter**. See delete_job.

Troubleshooting

Log Files

DSA Logs

Log File	Location	Description
`dsc.`*`timestamp`*`.log`	Resides on the DSC server in one of the following locations: • `$DSA_LOG_DIR` • `/var/opt/teradata/dsa/logs`	DSA job metadata processing Defaults: • Log level: Info • Log file size: 50 MB • Log file count: 10
`clienthandler.log`	Resides on the DSC server or media servers in one of the following locations: • `$DSA_LOG_DIR` • `/var/opt/teradata/dsa/logs`	DSA Network Client (ClientHandler) processing
`dscCommandline.log`	Resides on the DSC server or media servers in one of the following locations: • `$DSA_LOG_DIR` • `/var/opt/teradata/dsa/logs`	Command-line process
`dsarest.log`	Resides on the DSC server in one of the following locations: • `$DSA_LOG_DIR` • `/var/opt/teradata/dsa/logs`	REST server status

Viewpoint Logs

Log File	Location	Description
Viewpoint logs	`/var/opt/teradata/viewpoint/portal/logs`	Viewpoint portal information

Third-Party Backup Application Logs

Log File	Location	Description
`dsa_tdbackex.log`	`/var/opt/teradata/dsa/logs/dsa_tdbackex.log`	NetBackup job log that contains error messages
`dsierror.log`	`/var/opt/teradata/dsa/logs/dsierror.log`	Spectrum Protect job log that contains error messages

Database Logs

Log File	Location	Description
`BARLog_rsgno_partitionno*.txt`	In one of the following locations: • `<PDE-temp-path>` • `/var/opt/teradata/tdtemp/bar`	DSMain information, which can be useful in determining why a job failed When DSMAIN logs are set to the DEBUG logging level, more than one log file is created in `/var/opt/Teradata/tdtemp/bar`. To troubleshoot do the following: 1. Gather all the log files in the directory. 2. Zip the directory on all nodes involved in the job. 3. Send the files through email to DSA Frontline.

Location	Description
`/var/opt/teradata/tdtemp/post_restore_<`*version*`>`	Logs post_data_restore and post_dbc_restore script information. This information is useful in determining what happened during the post-restore scripts.

TVI Logging

TVI allows critical failures to be reported to Teradata immediately. Logging by TVI is enabled by default, but can be disabled by setting the value of the `logger.useTviLogger` property in the `dsc.properties` and `clienthandler.properties` files to `false.`

Following TVI errors are reported to Teradata DSA by message ID number. All of the errors greater than 4000000 are critical. Event codes are supported in CMIC release 11.01.02 or later unless otherwise noted.

1751001

Synopsis: DSC Server was started.
Meaning: DSC Server was started.
Probable Cause: DSC Server was started.
Recommendations: n/a

1751002

Synopsis: DSC was stopped.
Meaning: DSC Server was stopped.
Probable Cause: A service was stopped manually or during a shut down or reboot, or the DSC repository is down.
Recommendations: Check the log file for DSC at $DSA_LOG_DIR/dsc.log.

1752001

Synopsis: Alert for ad-hoc collection of support bundle.
Meaning: DSA support information collected.
Probable Cause: DSA support script was run to collect support information.

Recommendations: The collected support information can be used for troubleshooting.

1753001

Synopsis: BARNC.
Meaning: BARNC (the DSA Network Client) was started.
Probable Cause: The DSA Network Client was started.
Recommendations: NA

1753002

Synopsis: BARNC.
Meaning: BARNC (the DSA Network Client) was stopped.
Probable Cause: A service was stopped manually or during a shut down or reboot. DSA Network Client should start automatically on reboot.
Recommendations: Check the log file for BARNC (DSA Network Client) at $DSA_LOG_DIR/clienthandler.log.

4751001

Synopsis: DSC and Commons version mismatch
Meaning: DSC version does not match Commons version, DSC failed to start.
Probable Cause: Installed package is using the wrong libraries and is not valid.
Recommendations: Request a new build.

4751002

Synopsis: DSC cannot connect to JMS broker.
Meaning: The JMS broker is unreachable from the DSC.
Probable Cause: ActiveMQ service may be down or DSC may have incorrect JMS broker port/url configuration.
Recommendations: Check ActiveMQ service is running. Verify JMS broker port/url on DSC.

4751003

Synopsis: JDBC connection to DSC repository failed.
Meaning: Fail to connect to DSC repository.
Probable Cause: DSC repository is unavailable or there is a network error.
Recommendations: Check database is running. Verify logon user/password.

4751005

Synopsis: DSC and DBS version mismatch.
Meaning: DSC version does not match DBS version, DSC version must be equal or higher than DBS version.
Probable Cause: Installed wrong package.
Recommendations: Upgrade DSC package or downgrade DBS version.

4751006

Synopsis: Memory usage of the DSA communication queue has reached the 75% threshold that triggers TVI logging.
Meaning: DSA hangs.
Probable Cause: DSC consumers are not available when status messages are sent from DSA Network Client and DSMAIN producers.
Recommendations: Set `deleteAllMessagesOnStartup="true"` in `/opt/teradata/tdactivemq/config/td-broker.xml`. Then restart the ActiveMQ broker.

4753003

Synopsis: BARNC crash.
Meaning: BARNC (DSA Network Client) crashed.
Probable Cause: DSA Network Client crashed due to segmentation violation, broken pipe, or unknown reasons.
Recommendations: Find related core files in the Teradata core directory.

4753004

Synopsis: NFS full
Meaning: The network file system is out of space.
Probable Cause: There is a problem writing to file due to lack of disk space.
Recommendations: Free up NFS mounted disk space.

4753005

Synopsis: NFS mount failure
Meaning: There is a network file system mount failure.
Probable Cause: There is a problem writing to file due to an NFS mount error.
Recommendations: Verify that NFS is properly mounted and running.

BARNC (DSA Network Client) Error Codes

BARNC (DSA Network Client (ClientHandler)) errors are reported in the `clienthandler.log` and `$DSA_LOG_DIR/dsc.log` files.

BSASendData failed
BSAGetData failed

Possible causes: An error occurred in the media management software that prevents the backup or restore operation from completing successfully. The message includes additional text indicating the reason for the failure.
Remedy: Check the media management software and backup devices to make sure they are operating correctly.

CBB Filter write open failed
BB Filter write seek failed
CBB Filter write failed

Possible Causes (for any of these three errors): The shared storage area used for Teradata change block backup restore is full or is not properly configured.
Remedy: Check that the allocated path is available, has the proper permissions set, and is not full.

Failed to bind any listen sockets.

Possible causes: An attempt to start the DSA Network Client (ClientHandler) service was made while it was already running, or another process is using the network port assigned to the DSA Network Client service.
Remedy: Use the `/etc/init.d/clienthandler status` command to determine if the DSA Network Client service is running. Also, check that the port number specified in the `clienthandler.properties` file is correct.

JMSSessionFactory could not be created

Possible causes: The ActiveMQ message broker service is not active or cannot be reached.
Remedy: Check that the ActiveMQ message broker service for DSA is running, and that network access between media servers is working properly. The message may include additional information that is helpful in determining the cause of the failure.

No matching file: "Filename"

Possible causes: An attempt was made to restore an expired or failed backup.
Remedy: Check that the backup of the job being restored was successfully completed and was not deleted or expired by the media management software.

Read passed end of file

Possible causes: An attempt was made to restore a corrupted or incomplete backup file.
Remedy: Check that the backup being restored was successfully completed. Also, try to run a read_analyze job on the backup to determine whether the file is readable.

Locked Object Restrictions

The Teradata system places locks on objects during the time that transactions take place. For example, if a transaction is taking place on a table, you cannot run a read operation until the previous transaction is complete.

If there are locks on objects included in a job, DSA waits indefinitely until the locks are released. When this happens, the object status in the **Log** tab of the **Job Status** view displays "BLOCKED" to indicate that the object is locked. Release the locks so that the job can be run when all of the objects can be accessed.

Releasing Locked Objects

The `object_release` command releases all objects that are currently locked by a job. It does not release objects for new, running, or queued jobs.

- Type `dsc object_release` followed by the parameters, and press **Enter**.

Note:

The locked objects are released by the `object_release` command only if the user has access privileges on the job objects.

Parameter	Description
n|name *Name*	The name of the job on which to perform the action. Must be unique for each job.
S|skip_prompt *SkipPrompt*	[Optional] Skips displaying a confirmation message before performing the command action.
u|user_authentication *User*	Required when security management is enabled. Supplies the command with the Viewpoint user, and triggers a password prompt for authentication.

Database DBC Info

Database DBC

The DBC database contains critical system tables that define the user databases in the Analytics Database / Teradata Database.

The following table lists the tables that are archived for database DBC.

Note:

If a DBC user table is not listed as shown, it will not be archived for database DBC. Instead, use SQL to copy the DBC user table into a user-level table in another database for Teradata DSA to back up. The same table cannot be copied back to the DBC user table during a restore process.

Table Name	Description
AccessRights	Specifies all granted rights.
AccLogRuleTbl	Defines access logging rules generated by executing BEGIN/END LOGGING statements.
Accounts	Lists all authorized account numbers.
AsgdSecConstraints	Stores information about the name and value pairs of security constraints that have been assigned to users or profiles.
BusinessCalendarException	Stores information about days that are exceptions to the active business calendar pattern.
BusinessCalendarPattern	Stores information about different patterns of business days and non-business days for a given calendar.
CollationTbl	Defines MULTINATIONAL collation.
ConnectRulesTbl	Contains information about CONNECT THROUGH privileges.
ConstraintFunctions	Stores information about the UDFs for security constraint objects.
ConstraintValues	Stores information about the name and value pairs for security constraint objects.
DBase	Defines each database and user.
DBQLRuleCountTbl	Is reserved for internal use only.
DBQLRuleTbl	Specifies the rule table for DBQL.
Hosts	Defines information about user-defined character sets used as defaults for client systems.

Table Name	Description
LogonRuleTbl	Defines information about logon rules generated by a GRANT LOGON statement.
Maps	Defines all sparse and contiguous maps.
MapGrants	Stores information about hash map grants to users and roles.
NetSecPolicyLogRuleTbl	Defines the rules when the network security policy components are to be logged.
Next	Generates table, stored procedure, and database identifiers (internal table).
OldPasswords	Lists passwords that are no longer in use, including user to whom the password was assigned, date the password was changed, and encrypted password string.
Owners	Defines all databases owned by another.
Parents	Defines all parent/child relationships between databases.
PasswordRestrictions	Restricts a password from being created that contains any word listed in the table.
Profiles	Defines the available roles and profiles on the Teradata machine.
RCConfiguration	Records the configuration for RCEvent rows.
RCEvent	Records all archive and recovery activities.
RCMedia	Records all removable devices used in archive activities.
ReconfigDeleteOrderTbl	Stores information about the order in which tables are processed during the Deletion phase of reconfiguration done using the Reconfig utility.
ReconfigInfoTbl	Stores summary and status information about reconfiguration done using the Reconfig utility.
ReconfigRedistOrderTbl	Stores information about the order in which tables are processed during the Redistribution phase of reconfiguration done using the Reconfig utility.
ReconfigTableStatsTbl	Stores reconfig statistics about each table processed during the redistribution phase and the deletion phase of a reconfig.
RoleGrants	Contains roles granted to users or other roles.
Roles	Defines the available roles and profiles on the Teradata machine.
SecConstraints	Stores information about security constraint objects.
SysSecDefaults	Defines the system security defaults of an Analytics Database / Teradata Database, for example, the minimum and maximum characters allowed in a password.
Translation	Defines hexadecimal codes that form translation tables for non-English character sets.

Table Name	Description
UDTCast	Contains information on the source and target data types that are involved in the casting operation.
UDTInfo	Captures the specifics contained within the CREATE TYPE statement.
UDTTransform	Contains the transform group name and the routine identifiers.
Zones	Defines all secure zones.
ZoneGrants	Stores information about secure zone access grants to users and roles.

Note:

- To archive the DBC, user must have DUMP access privilege on the DBC.
- To restore the DBC:
  - DBC only logons must be enabled.
  - User must be DBC with RESTORE access to DBC.

If included in an archive operation, database DBC is always archived first, regardless of alphabetical order. If included in a restore or copy operation, database DBC is always restored or copied first. Refer to Full System Restore Using the BAR Portlets or Full System Restore Using the Command Line for specific instructions on restoring database DBC.

DBC Dictionary Tables

An all-AMPs archive (or dictionary archive) of a database or table contains the Analytics Database / Teradata Database dictionary rows that are needed to define the entity.

This table alphabetically lists the dictionary rows or tables that are archived by an all-AMPs data archive (or a dictionary tables archive) of a user database or table.

Note:

The following tables and table rows should never be copied to a user-level table in another database and be archived from there.

Tables and Table Rows	Description
ConstraintNames	Constraints that have names
DBCAssociation	Retrieves information about tables that have been ported using the Dump /Restore facility
Dependency	Stores relationships among a UDT, its dependent routines, User- Defined Casts, User-Defined Transforms, User-Defined Orderings, and any dependency on any other database object

Tables and Table Rows	Description
ErrorTbls	Stores data and error table relations
IdCol	Identity column
Indexes	Columns that are indexes
JARS	Contains one row for each JAR object installed on the database system
JAR_JAR_USAGE	Stores one row for each JAR included in the SQL-Java path of another JAR. The SQL-Java path is created, altered, or dropped by calling SQLJ.Alter_Java_Path.
QueryStatsTbl	Saves information about collected statistics on the queries
ReferencedTbls	Referenced columns (that is, Parent Key) for Parent tables
ReferencingTbls	Referencing columns (that is, Foreign Key) for Child tables
ROUTINE_JAR_USAGE	Contains one row for each Java external stored procedure or Java user-defined function that names a JAR in its external Java reference string.
ServerInfo	DBC.ServerInfo provides the list of name/value pairs defined for a server
ServerTblOpInfo	DBC.ServerTblopInfo provides the details of the associated table operators to the foreign server
StatsTbl	Saves information about collected statistics on the base tables, views, or queries
TableConstraints	Returns information about constraints on a TABLE object by accessing the database.
TextTbl	System table containing overflow DDL text
TriggersTbl	Definition of all triggers in the database
TVFields	All columns in data tables and views
TVM	All objects in the database
UDFInfo	Information on user-defined functions
UIF_Info	Contains a record for each User Installed File on the database system
UnresolvedReferences	Unresolved referential constraints

Using Teradata DSC in the Public Cloud

The Teradata DSC can be used in the public cloud with AWS and Azure. If you are only working in the cloud, nothing else needs to be installed, just follow the rest of the instructions in this section.

If you want to back up to the cloud from an on-premises system, you need to install either the AXMS3 or AXMAzure packages. See *Teradata® DSA - DSU Installation, Configuration, and Upgrade Guide*, B035-3153.

If you are an experienced command line user and do not want to use the portlets, see Configuring the DSC for the Public Cloud Using the CLI.

Before You Begin - AWS

Before you can configure the Data Stream Controller (DSC) to prepare for backup and restore, you must have an Analytics Database / Teradata Database instance and the DSC set up in AWS. See the *Teradata® VantageCloud Enterprise on AWS (DIY) Installation and Administration Guide*, B035-2800, *Planning Worksheet for Vantage on AWS (DIY)* section and follow those instructions to launch Teradata instances and configure the DSC in AWS.

While setting up your Teradata system in AWS be aware of the following items:

- Select a database instance that includes the DSC AMI
- S3 storage is recommended with Teradata DSC

Once you have launched a database instance that includes the DSC, be sure to follow the instructions in these topics (in this order) under *Data Stream Controller Configuration*:

1. *Initializing a DSC Instance*
2. *Configuring and Initializing ClientHandler on Vantage Nodes*
3. *Configuring Additional EBS Storage for a DSC Instance*
4. *Configuring BAR Portlets on a Viewpoint Instance*

Before You Begin - Azure

Before you can configure the DSC, you must launch an Analytics Database / Teradata Database VM and DSC in Azure. See the *Teradata Vantage™ on Azure (DIY) Installation and Administration Guide*, B035-2810, *Quick Start* section and follow those instructions to get the DSC configured in Azure.

While setting up your Teradata system in Azure be aware of the following items:

- Select a database VM that includes the DSC VM
- Teradata DSC uses block blob storage

Once you have launched a database VM that includes the DSC, be sure to follow the instructions in these topics (in this order) under *Data Stream Controller Configuration*:

1. *Initializing a DSC VM*
2. *Configuring and Initializing ClientHandler on Vantage Nodes*
3. *Configuring BAR Portlets on a Viewpoint Instance*

Configuring Teradata DSC for the Public Cloud

These configuration tasks are required in the **BAR Setup** portlet before you can create backup and restore jobs in the **BAR Operations** portlet.

1. Adding or Editing a Teradata System
2. [AWS only] Adding an Amazon S3 Account
3. [Azure only] Adding Azure Blob Storage
4. Adding or Copying a Target Group
5. Adding or Editing a Restore Group
6. Scheduling Automatic Repository Backups

Configuring the DSC for the Public Cloud Using the CLI

CLI users can configure and use the DSC in the public cloud (AWS or Azure) by using the command line.

1. In the cloud product, run `dsu-init` from the DSC master node.

   [AWS] See *Initializing a DSC Instance*, in the *Teradata® VantageCloud Enterprise on AWS (DIY) Installation and Administration Guide*, B035-2800.
   [Azure] See *Initializing a DSC VM*, in the *Teradata Vantage™ on Azure (DIY) Installation and Administration Guide*, B035-2810.
2. In the cloud product, run `barnc-init` from each Teradata node.

   [AWS] See *Configuring and Initializing ClientHandler on Vantage Nodes*, in the *Teradata® VantageCloud Enterprise on AWS (DIY) Installation and Administration Guide*, B035-2800.
   [Azure] See *Configuring and Initializing ClientHandler on Vantage Nodes*, in the *Teradata Vantage™ on Azure (DIY) Installation and Administration Guide*, B035-2810.
3. If you want to back up to the cloud from an on-premises system, you must install the AXMS3 or AXMAzure packages, see *Teradata® DSA - DSU Installation, Configuration, and Upgrade Guide*, B035-3153 for the instructions.
4. Using the BAR CLI on the DSC master node, run the following commands:
   a. `config_systems`
   b. `enable_component`
   c. `config_aws` OR `config_azure`
5. Create target groups. See Target Groups.
6. Set up repository backup. See Backing Up the DSC Repository with the Command Line.
7. Create and run jobs. See Planning a Job: Command Line.

Proxy Server Support

Proxy Server is a dedicated server between Cloud Storage Server and Access Module. It redirects the Cloud Storage Server REST API request or response between access module and cloud storage. See https://ubuntu.com/server/docs/proxy-servers-squid

To enable Proxy Server Support in Access Module, you must pass the following parameters to the Access Module Library. To pass the parameters, set the following properties in `/etc/opt/teradata/dsa/clienthandler.properties` file.

AWS S3 Access Module:

```
axm.libAXMS3.so.s3proxyhost=proxyserverName
```

```
axm.libAXMS3.so.s3proxyport=port
```

Azure Access Module:

```
axm.libAXMAzure.so.s3proxyhost=proxyserverName
```

```
axm.libAXMAzure.so.s3proxyport=port
```

GCP Access Module:

```
axm.libAXMGCP.so.s3proxyhost=proxyserverName
```

```
axm.libAXMGCP.so.s3proxyport=port
```

Administrative Tasks

Saving the DSC Repository Configuration Information

Follow these steps to save DSC repository configuration information:

1. Locate the names of the media server, target group, and storage apps used in the replication target group:
   `dsc list_components -t target_group -r`
2. Collect the following application details:
   `dsc export_config -t MEDIA_SERVER -n <media_server_name> -f MEDIA_SERVER.xml`
   `dsc export_config -t TARGET_GROUP -n <replication_tg_name> -f replicating_tg.xml -r`
   `dsc export_config -t <TYPE_OF_CONFIGURATION> -n <APP_NAME> -f APP.xml`

   where TYPE_OF_CONFIGURATION can be NBU_SERVER, DD_BOOST, DISK_FILE_SYSTEM, AWS_APP, AZURE_APP or GCP_APP
3. Store these files (MEDIA_SERVER.xml, replicating_tg.xml, and APP.xml) in safe place outside the DSC. Anytime you change configuration or upgrade DSA, repeat these steps.

Example: Saving the DSC Configuration

Here is an example using AWS as a replication target group.

Following are the AWS group details:

Value	Description
Target Group Name	aws_replicating_tg
Is Enabled	Y
AWS Account Name	AWS
Region	ap-south-1
Media server name	localms
AWS APP name	AWS
Bucket Prefix	awsjanrelease
Storage Device	backup/1

1. Find the target group, account and media server name used in replication target group.
   `dsc list_components -aws_replicating_tg-r`

2. Collect the application details following the steps in Saving the DSC Repository Configuration Information
3. Export media server configuration detail to save in file.
   `dsc export_config -t MEDIA_SERVER -n localms -f MEDIA_SERVER.xml`
4. Export AWS APP configuration detail to save in file.
   `dsc export_config -t AWS_APP -n AWS -f AWS_APP.xml`
5. Export target group config detail to save in file.
   `dsc export_config -t TARGET_GROUP -n aws_replicating_tg -f aws_replicating_tg.xml -r`

Managing DBC and User Data in the BAR Portlets

The DBC database contains critical system tables that define the user databases in Analytics Database / Teradata Database. The system tables, views, and macros in the DBC database cannot be selectively backed up and must be backed up in their entirety. The **BAR Operations** portlet does not allow any of the DBC system objects to be selected for backup.

Note:

Full user access privileges are required for archiving or restoring DBC.

To back up and restore the DBC database and all user databases, you must create a separate backup job for the DBC database, then a separate backup job for the other databases. After the backup jobs have successfully completed, you must create a separate restore job for each of the backup jobs.

Note:

When restoring the DBC backup, the DBC password of the system that was backed up is required.

---

1. Back up the DBC and user data in BAR Operations.
2. Restore the DBC and user data in BAR Operations.

Backing Up DBC and User Data Using the BAR Portlets

You must create two backup jobs. One includes all the databases under DBC and excludes the DBC database automatically and one includes only the DBC database.

1. From the BAR Operations **Saved Jobs** view, create a backup job that saves the databases under DBC:
   a. Click **New Job**.
   b. Select the **Backup** job type, then click **OK**.
   c. In the **New Backup Job** view, enter a job name, such as `Backup-DBC-All`.
   d. Select a **Source System** from the list.
   e. Enter the user credentials.
   f. Select a **Target Group** from the list.
   g. Select the DBC database in the **Objects** tab.

h. Click next to the DBC database.
The **Settings** dialog box appears.
i. Check the **Include all children databases and users** box.
j. Click **OK**.
k. Click **Save**.
l. Click on `Backup-DBC-All` and select **Run**.

2. From the BAR Operations **Saved Jobs** view, create a backup job that saves only the DBC database:
a. Click **New Job**.
b. Select the **Backup** job type, then click **OK**.
c. In the **New Backup Job** view, enter a job name, such as `Backup-DBC-Only`.
d. Select a **Source System** from the list.
e. Enter the user credentials.
f. Select a **Target Group** from the list.
g. [Optional] Add a **Description**.
h. Select the DBC database in the **Objects** tab.
i. [Optional] To verify the parent and objects selected, click the **Selection Summary** tab.

Note:

Size information is not available for DBC backup jobs. N/A displays as the size value for DBC backup jobs.

---

j. [Optional] To adjust job settings for the job, click the **Job Settings** tab.
k. Click **Save**.
l. Click on `Backup-DBC-Only` and select **Run**.

3. Save the name of both backup job save sets.
The backup jobs must complete with a COMPLETED_SUCCESSFULLY or WARNING status before you can create a restore job.

Full System Restore Using the BAR Portlets

Prerequisite:

When restoring an archive made on a source system with hardware block-level compression, you must install the hardware compression driver package (`teradata-expressdx`) on all TPA nodes of the destination system. The installation requires a reboot of these nodes. Because it allows the destination system to read the compressed archive, this requirement applies even if the destination system is not set up for hardware compression.

Note:

The `teradata-expressdx` driver package is provided with systems that are equipped with compression hardware.

For additional information, see the *Teradata Vantage™ - Advanced SQL Engine Node Software Migration Guide* or *Teradata® Database Node Software Migration Guide* specific to your database version.

NOTICE

These steps wipe the system. Use these steps for a full system restore.

The backup jobs must have successfully completed to create restore jobs from the save sets.

You must create four restore jobs from the two backup job save sets. One includes only the DBC database and three are from the save set that includes all the databases under DBC and excludes DBC.

1. On the destination system, run SYSINIT with options appropriate to your system.
2. On the destination system, check the Time Zone setting status, and disable the setting if it is enabled.
3. On the destination system, start the DBS Database Initializing Program (DIP) and run the DIPMIG script, which runs DIPERRS, DIPDB, DIPVIEWS, DIPVIEWSV, and DIPBAR.
4. If you disabled the Time Zone setting on the destination system, enable it.
5. From the **BAR Setup** portlet, check the activation status of the destination system and do one of the following:
   - If the destination system is configured and enabled in the **BAR Setup** portlet, click **Update** for **Selector: Update system selector for JMS messages**.
   - If the destination system is not configured in the **BAR Setup** portlet, add the system and click **Apply**.
6. On the destination system, start DSMain from the Database Window (DBW) console supervisor screen by entering:
   `start bardsmain -d` *DSCName*, where *DSCName* is the unique name of your DSC server
7. Enable logons for the DBC user only.

On the destination system, enter the following from the DBW console supervisor screen:
`enable dbc logons`

8. Create a DBC-only restore job from the backup job save set that saved only the DBC database:

Important:

You need two sets of credentials for these steps: the DBC user and password for the *target* system and the DBC user and password for the *source* system.

---

a. In the Bar Operations **Saved Jobs** view, click for the DBC only backup job, and select **Create Restore Job**.
b. Enter a **Job Name**, such as `Restore-DBC-Only`.
c. Select a **Destination System** from the list.
d. When prompted, enter logon credentials for the current DBC user and password for the target system.
e. Select a **Target Group** from the list.
f. Click the **Job Settings** tab, then select **Set Credentials** to enter the credentials for the DBC user and password of the source system at the time the backup save set was generated.
g. Click **Save**.
h. Click for Restore-DBC-Only and select **Run**.

Note:

After the DBC restore is complete, the DBC password is set to the source system DBC password.

Troubleshooting:

If there are any errors, follow the instructions in the log file to correct the problem and run the post dbc script again.

- The post restore script output log files are saved in `/var/opt/teradata/tdtemp/post_restore_`*dbs version*.
- If the DBC restore job fails, check the DSC job status log to correct the issue.
- If the failure occurs before the postscript phase, you must run SYSINIT before rerunning the restore job.

If you ran post_dbc_restore manually, you must update the JMS system selector in the **BAR Setup** portlet once post_dbc_restore completes successfully.

- If the destination system is configured and enabled in the **BAR Setup** portlet, click **Update** for **Selector: Update system selector for JMS messages**.
- If the destination system is not configured in the **BAR Setup** portlet, add the system and click **Apply**.

---

9. Create a SYSLIB database restore job from the DBC ALL backup save set:

a. In the Bar Operations **Saved Jobs** view, click on the backup job created for the databases under DBC, and select **Create Restore Job**.
b. Enter a **Job Name**, such as `Restore-SYSLIB`.
c. Select a **Destination System** from the list.
d. When prompted, enter logon credentials for the current DBC user and password for the destination DBS.
e. Select a **Target Group** from the list.
f. On the **Objects** tab, clear the top checkbox, then expand the tree and select the checkbox for *only* **SYSLIB**.
g. Click **Save**.
h. Click on Restore-SYSLIB and select **Run**.

10. Create a restore job for all other data from the DBC ALL backup save set (excluding SYSLIB and TD_SERVER_DB):
    a. In the Bar Operations **Saved Jobs** view, click for the backup job for the databases under DBC, and select **Create Restore Job**.
    b. Enter a **Job Name**, such as `Restore-DBC-All`.
    c. Select a **Destination System** from the list.
    d. When prompted, enter logon credentials for the current DBC user and password for the destination DBS.
    e. Select a **Target Group** from the list.
    f. [Teradata Database 15.00 or later] Clear the checkbox for **TD_SERVER_DB** in the **Objects** tab. TD_SERVER_DB has dependencies that must be met before restoration.
    g. [Teradata Database 15.00 or later] Clear the checkbox for **SYSLIB** in the **Objects** tab.
    h. Click **Save**.
    i. Click for Restore-DBC-All and select **Run**.
    j. If there are any errors, follow the instructions in the log file to correct the problem and run the post data script again.
    The post restore script output log files are saved in `/var/opt/teradata/tdtemp/post_restore_`*`dbs version`*.
11. [Teradata Database 15.00 and later] From the BAR Operations **Saved Jobs** view, create a TD_SERVER_DB restore job from the backup job save set of the databases under DBC, excluding DBC:
    a. Click for the backup job for the databases under DBC, and select **Create Restore Job**.
    b. Enter a **Job Name**, such as `Restore-TD_SERVER_DB`.
    c. Select a **Destination System** from the list.
    d. When prompted, enter logon credentials for the current DBC user and password for the destination DBS.
    e. Select a **Target Group** from the list.
    f. On the **Objects** tab, clear the top checkbox, then expand the tree and select the checkbox for *only* TD_SERVER_DB.

g. Click **Save**.
h. Click on Restore-TD_SERVER_DB and select **Run**.

12. On the destination system, start the DIP and run the DIPALL script.
13. On the destination system, from the DBW console supervisor screen, enable logons for all users.
    `enable logons`

Troubleshooting a Full System Restore

If you have problems with a full system restore, try the following.

Verify the Media Server IP Addresses

1. Ping the IP address from the media server to the database.
2. Ping the IP address from the database to the media server.
3. Trim any invalid or unused IP addresses.

post_dbc_restore Script Fails to Launch

If the restore works, but fails to launch the post_dbc_restore script, you receive a message similar to the following:

```
1079 Post dbc restore script not started. Please run the post_dbc_restore
script manually with the following command:
/opt/teradata/PUTTools/td17<version>/IUMB_scripts/post_dbc_restore -s -N -j
3 TDT5WSDB/'dbc',<dbc_password>.
```

In addition, the table SYSBAR.DSAConnectionsTbl is not repopulated and you receive a message similar to this:

```
1001   SQL Exception 3807 3807: DBCSQL Exception: Object
'SYSBAR.DSAConnectionsTbl' does not exist.
DSAConnectionsTbl INSERT failed.  All DSCs should activate system <system_name>
```

Follow these steps:

1. Manually launch the post_dbc_restore script:

   `/opt/teradata/PUTTools/td17<version>/IUMB_scripts/post_dbc_restore -s -N -j 3 TDT5WSDB/'dbc',<dbc_password>`

2. Repopulate the DSAConnectionsTbl:

Interface	Steps
BAR Setup portlet	Check the activation status of the destination system in the **BAR Setup** portlet and do one of the following: • If the destination system is configured and enabled, click **Update** for **Selector: Update system selector for JMS messages**

Interface	Steps
	• If the destination system is not configured, add the system and click **Apply**.
Command line	Check the activation status of the system: `dsc list_components -t system` and do one of the following: • If the destination system is configured and enabled, run this command: `dsc config_systems -f configsystem.xml -s selector` • If the destination system is not configured, add the system: `dsc config_systems -f configsystem.xml` where `configsystem.xml` is similar to the example shown in Full System Restore Using the Command Line.

3. On the destination system, start DSMain from the Database Window (DBW) console supervisor screen by entering:

   `start bardsmain -d` *DSCName*, where *DSCName* is the unique name of your DSC server

Compile Errors Reported During a Restore

When restored by DSA, certain Teradata-provided stored procedures fail to recompile in the post_data_restore scripts. These procedures have dependencies that are not met until after DIPALL is run at the conclusion of the migration. You can can manually recompile them using ALTER statements after DIPALL completes.

"SYSLIB"."JSON_SHRED_BATCH_U"
"SYSLIB"."JSON_SHRED_BATCH"
"TD_SYSXML"."XSLT_SHRED_BATCH"
"TD_SYSXML"."AS_SHRED_BATCH"
"TD_SYSXML"."XSLT_SHRED"
"TD_SYSXML"."XMLPUBLISH"
"TD_SYSXML"."XMLPUBLISH_STREAM"
"LOCKLOGSHREDDER"."SP_LockLog_Shredder"

These errors are detected by the DSC when it runs the post_data_restore scripts and it concludes with a Warning job status rather than Complete. Check the log file noted in the error message provided by the DSC. The errors flagged by the DSC for these issues will not interfere with the rest of the restore.

The following commands will recompile the stored procedures after DIPALL has been run.

```
ALTER PROCEDURE "SYSLIB"."JSON_SHRED_BATCH_U" COMPILE;
ALTER PROCEDURE "SYSLIB"."JSON_SHRED_BATCH" COMPILE;
ALTER PROCEDURE "TD_SYSXML"."AS_SHRED_BATCH" COMPILE;
ALTER PROCEDURE "TD_SYSXML"."XSLT_SHRED" COMPILE;
ALTER PROCEDURE "TD_SYSXML"."XSLT_SHRED_BATCH" COMPILE;
ALTER PROCEDURE "TD_SYSXML"."XMLPUBLISH" COMPILE;
ALTER PROCEDURE "TD_SYSXML"."XMLPUBLISH_STREAM" COMPILE;
ALTER PROCEDURE "LOCKLOGSHREDDER"."SP_LockLog_Shredder" COMPILE;
```

Warning When the Database Exceeds Logical Space Limit

When the database exceeds its logical space limit while doing a restore operation, warning messages may appear. You can ignore the messages as the restore does not fail. However, when the restore is complete, you must modify the perm space to add objects to the database or to add new rows to the tables.

The warning message looks like this:

```
Warning code   Warning message
-------------------------------------------------------------------------------
1196           Database <database_name> has exceeded it's logical space limit.
```

Managing DBC and User Data in the Command Line

The DBC database contains critical system tables that define the user databases in the Analytics Database / Teradata Database. The system tables, views, and macros in the DBC database cannot be selectively backed up and must be backed up in their entirety. The BAR Operations portlet does not allow any of the DBC system objects to be selected for backup.

Note:

Full user access privileges are required for archiving or restoring DBC.

To back up and restore the DBC database and all user databases, you must create a separate backup job for the DBC database, then a separate backup job for the other databases. After the backup jobs have successfully completed, you must create a separate restore job for each of the backup jobs.

Note:

When restoring the DBC backup, the DBC password of the system that was backed up is required.

---

1. Back up the DBC and user data using the command line.
2. Restore DBC and user data using the command line.

Backing Up DBC and User Data Using the Command Line

Two backup jobs are required: one for all DBC user data and one for DBC only.

Each config or create step uses an XML file. The following examples include a representative XML file.

Note:

Before you create a restore job, a backup job must have completed successfully or with a warning.

---

1. Create a backup job for user data that includes all children of DBC but excludes the DBC database.
   Example: `dsc create_job -n DBC-All -f DBCAllJob.xml`

This example of `DBCAllJob.xml` includes all children of DBC by setting the `<includeAll>` attribute to `true`.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<dscCreateJob xmlns="http://schemas.teradata.com/v2012/DSC">
    <job_instance>
        <job_name>DBC-All</job_name>
        <job_description>Backup user data and all objects in
DBC</job_description>
        <job_type>BACKUP</job_type>
        <job_state>ACTIVE</job_state>
        <auto_retire>false</auto_retire>
        <objectlist>
            <objectinfo>
                <object_name>DBC</object_name>
                <object_type>DATABASE</object_type>
                <parent_name></parent_name>
                <parent_type>BACKUP_JOB</parent_type>
                <object_attribute_list>
                     <includeAll>true</includeAll>
                </object_attribute_list>
            </objectinfo>
        </objectlist>
    </job_instance>
    <source_tdpid>systemname</source_tdpid>
    <target_media>1_5_drives</target_media>
    <job_options>
        <online>false</online>
        <data_phase>DATA</data_phase>
        <query_band></query_band>
        <dsmain_logging_level>Error</dsmain_logging_level>
    </job_options>
</dscCreateJob>
```

2. If prompted, enter login credentials.
3. Run the job.
   Example: `dsc run_job -n DBC-All`
4. Create a backup job that includes only the DBC database.
   A DBC only backup job cannot be run as online.
   Example: `dsc create_job -n DBC-Only -f DBCOnlyJob.xml`

   This example of `DBCOnlyJob.xml` excludes all children of DBC by setting the `<includeAll>` attribute to `false`.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<dscCreateJob xmlns="http://schemas.teradata.com/v2012/DSC">
    <job_instance>
        <job_name>DBC-Only</job_name>
        <job_description>Backup DBC only and exclude all child
objects</job_description>
        <job_type>BACKUP</job_type>
        <job_state>ACTIVE</job_state>
        <auto_retire>false</auto_retire>
        <objectlist>
            <objectinfo>
                <object_name>DBC</object_name>
                <object_type>DATABASE</object_type>
                <parent_name></parent_name>
```

```
                <parent_type>BACKUP_JOB</parent_type>
                <object_attribute_list>
                     <includeAll>false</includeAll>
                </object_attribute_list>
            </objectinfo>
        </objectlist>
    </job_instance>
    <source_tdpid>systemname</source_tdpid>
    <target_media>1_5_drives</target_media>
    <job_options>
        <online>false</online>
        <data_phase>DATA</data_phase>
        <query_band></query_band>
        <dsmain_logging_level>Error</dsmain_logging_level>
    </job_options>
</dscCreateJob>
```

5. When prompted, enter login credentials.
6. Run the job.
   Example: `dsc run_job -n DBC-Only`
7. [Optional] Check the status of any running job:
   `dsc job_status -n JobName`
   `Job Status: COMPLETED_SUCCESSFULLY` indicates the job is finished. A job might also complete with warning or errors.

Full System Restore Using the Command Line

Prerequisite:

When restoring an archive made on a source system with hardware block-level compression, you must install the hardware compression driver package (`teradata-expressdx`) on all TPA nodes of the destination system. The installation requires a reboot of these nodes. Because it allows the destination system to read the compressed archive, this requirement applies even if the destination system is not set up for hardware compression.

Note:

The `teradata-expressdx` driver package is provided with systems that are equipped with compression hardware.

For additional information, see the *Teradata Vantage™ - Advanced SQL Engine Node Software Migration Guide* or *Teradata® Database Node Software Migration Guide* specific to your database version.

NOTICE

These steps wipe the system. Use these steps for a full system restore.

Before you create a restore job, a backup job must have completed either successfully or with a warning.

You must create four restore jobs, one for DBC only and three from the all DBC user data backup (SYSLIB, TD_Server_Database, and all other data).

1. On the destination system, run SYSINIT with options appropriate to your system.
2. On the destination system, check the Time Zone setting status, and disable the setting if it is enabled.
3. On the destination system, start the DBS Database Initializing Program (DIP) and run the DIPMIG script.
4. If you disabled the Time Zone setting on the destination system, enable it.
5. Check the activation status of the destination system using the dsc list_components command, `dsc list_components -t system`, and do one of the following:
   - If the destination system is configured and enabled, run a config_systems command with the skip parameter set to selector, `dsc config_systems -f configsystem.xml -s selector`.
   - If the destination system is not configured, add the system using the config_systems command, `dsc config_systems -f configsystem.xml`. See the following sample `configsystem.xml` file.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

          <!--****This command must be on one line.****-->
<dscConfigSystems xmlns="http://schemas.teradata.com/v2012/DSC"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:schemaLocation="DSC.xsd">

<system>
<!-- 'system_name' - Required, max 32 characters -->
<system_name>system7</system_name>

<!-- 'tdpid' - Required (unless skipped by option)-->
<tdpid>system7</tdpid>

<!-- 'database_query_method' -
Required, accepted values: BASE_VIEW/EXTENDED_VIEW -->
<database_query_method>BASE_VIEW</database_query_method>

<!-- 'streams_softlimit' - Required, number of streams per node per job -->
<streams_softlimit>20</streams_softlimit>

<!-- 'streams_hardlimit' - Required, max number of streams per node-->
<streams_hardlimit>20</streams_hardlimit>

<!-- 'reset_node_limit' - Optional, accepted values: true/false -->
<reset_node_limit>false</reset_node_limit>

<!-- 'skip_force_full' - Optional, accepted values: true/false -->
<skip_force_full>false</skip_force_full>
</system>
</dscConfigSystems>
```

6. On the destination system, start DSMain from the Database Window (DBW) console supervisor screen by entering:
   `start bardsmain -d` *DSCName*, where *DSCName* is the unique name of your DSC server
7. Enable logons for the DBC user only.

On the destination system, enter the following from the DBW console supervisor screen:
`enable dbc logons`

8. Create a DBC-Only restore job, excluding all children of DBC, for example:
   `dsc create_job -n Restore-DBC-Only -f RestoreDBCOnlyJob.xml`

   Sample `RestoreDBCOnlyJob.xml` file:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<dscCreateJob xmlns="http://schemas.teradata.com/v2012/DSC">
  <job_instance>
    <job_name>Restore-DBC-Only</job_name>
    <job_description>Restore DBC only and exclude all child
objects</job_description>
    <job_type>RESTORE</job_type>
    <job_state>ACTIVE</job_state>
    <auto_retire>false</auto_retire>
    <backup_name>DBC-Only</backup_name>
    <backup_version>0</backup_version>
    <all_backup_objects>true</all_backup_objects>
  </job_instance>
  <source_media>1_5_drives </source_media>
  <target_tdpid>systemname</target_tdpid>
  <job_options>
    <enable_temperature_override>false</enable_temperature_override>
    <temperature_override>DEFAULT</temperature_override>
    <block_level_compression>DEFAULT</block_level_compression>
    <disable_fallback>false</disable_fallback>
    <query_band></query_band>
    <dsmain_logging_level>Debug</dsmain_logging_level>
    <reblock>false</reblock>
    <run_as_copy>false</run_as_copy>
  </job_options>
</dscCreateJob>
```

9. When prompted, enter logon credentials:

Option	Description
Target username	Current DBC user for the target DBS.
Target password	Current DBC password for the target DBS.
Is this Restore job a DBC restore?	y
Password for the backup	DBC password of the source system at the time the backup save set was generated.

10. Run the job.
    `dsc run_job -n Restore-DBC-Only`
    After the DBC restore job is complete, the DBC password is set to the DBC password of the source system. The DBC database must restore successfully before you can restore user data.
11. Create a restore job for SYSLIB only:
    `dsc create_job -n Restore-SYSLIB -f R-Syslib.xml`

Important:

`all_backup_objects` must be false when processing individual object restore or copy. If `all_backup_objects` is false and an objectlist is not provided, the job cannot be created.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<dscCreateJob xmlns="http://schemas.teradata.com/v2012/DSC">
    <job_instance>
        <job_name>R-Syslib</job_name>
        <job_description></job_description>
        <job_type>RESTORE</job_type>
        <job_state>ACTIVE</job_state>
        <auto_retire>false</auto_retire>
        <objectlist>
            <objectinfo>
                <object_name>SYSLIB</object_name>
                <object_type>DATABASE</object_type>
                <parent_name>DBC</parent_name>
                <parent_type>USER</parent_type>
                <object_attribute_list>
                    <includeAll>false</includeAll>
                </object_attribute_list>
            </objectinfo>
        </objectlist>
        <backup_name>DBC-ALL</backup_name>
        <backup_version>0</backup_version>
        <all_backup_objects>false</all_backup_objects>
    </job_instance>
    <source_media>target1</source_media>
    <target_tdpid>sdt12687</target_tdpid>
    <job_options>
        <enable_temperature_override>true</enable_temperature_override>
        <disable_fallback>false</disable_fallback>
        <query_band></query_band>
        <dsmain_logging_level>Error</dsmain_logging_level>
        <reblock>false</reblock>
        <run_as_copy>false</run_as_copy>
        <skip_archive>false</skip_archive>
        <skip_stats>false</skip_stats>
        <concurrent_bld_per_tbl>5</concurrent_bld_per_tbl>
    </job_options>
</dscCreateJob>
```

12. Run the job.

```
dsc run_job -n R-Syslib
```

13. Create a DBC-All restore job for user data, including all children of DBC except SYSLIB and TD_SERVER_DB:

```
dsc create_job -n Restore-DBC-All -f RestoreDBCAllJob.xml
```

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<dscCreateJob xmlns="http://schemas.teradata.com/v2012/DSC">
    <job_instance>
        <job_name>Restore-DBC-All</job_name>
        <job_description></job_description>
        <job_type>RESTORE</job_type>
        <job_state>ACTIVE</job_state>
        <auto_retire>false</auto_retire>
        <objectlist>
            <objectinfo>
                <object_name>DBC</object_name>
                <object_type>DATABASE</object_type>
                <object_attribute_list>
                    <includeAll>false</includeAll>
                </object_attribute_list>
                <exclude>
                    <excludeobjectinfo>
                        <object_name>SYSLIB</object_name>
                        <object_type>DATABASE</object_type>
                    </excludeobjectinfo>
                    <excludeobjectinfo>
                        <object_name>TD_SERVER_DB</object_name>
                        <object_type>DATABASE</object_type>
                    </excludeobjectinfo>
                </exclude>
            </objectinfo>
        </objectlist>
        <backup_name>DBC-ALL</backup_name>
        <backup_version>0</backup_version>
        <next_incremental_restore>false</next_incremental_restore>
        <all_backup_objects>true</all_backup_objects>
    </job_instance>
    <source_media>target1</source_media>
    <target_tdpid>abutera</target_tdpid>
    <job_options>
        <enable_temperature_override>true</enable_temperature_override>
        <disable_fallback>false</disable_fallback>
        <query_band></query_band>
        <dsmain_logging_level>Debug</dsmain_logging_level>
```

```
        <nowait>true</nowait>
        <reblock>false</reblock>
        <run_as_copy>false</run_as_copy>
        <skip_archive>false</skip_archive>
        <skip_stats>false</skip_stats>
        <concurrent_bld_per_tbl>5</concurrent_bld_per_tbl>
    </job_options>
</dscCreateJob>
```

14. When prompted, enter logon credentials and run the job.
    `dsc run_job -n Restore-DBC-All`
15. Create a restore job for the TD_SERVER_DB database:
    `dsc create_job -n R-TDSERVER -f R-TDSERVER.xml`

Important:

`all_backup_objects` must be false when processing individual object restore or copy. If `all_backup_objects` is false and an objectlist is not provided, the job cannot be created.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<dscCreateJob xmlns="http://schemas.teradata.com/v2012/DSC">
    <job_instance>
        <job_name>R-TDSERVER</job_name>
        <job_description></job_description>
        <job_type>RESTORE</job_type>
        <job_state>ACTIVE</job_state>
        <auto_retire>false</auto_retire>
        <objectlist>
            <objectinfo>
                <object_name>TD_SERVER_DB</object_name>
                <object_type>DATABASE</object_type>
                <parent_name>DBC</parent_name>
                <parent_type>USER</parent_type>
                <object_attribute_list>
                    <includeAll>false</includeAll>
                </object_attribute_list>
            </objectinfo>
        </objectlist>
        <backup_name>DBC-ALL</backup_name>
        <backup_version>0</backup_version>
        <all_backup_objects>false</all_backup_objects>
    </job_instance>
    <source_media>target1</source_media>
    <target_tdpid>sdt12687</target_tdpid>
```

```
    <job_options>
        <enable_temperature_override>true</enable_temperature_override>
        <disable_fallback>false</disable_fallback>
        <query_band></query_band>
        <dsmain_logging_level>Error</dsmain_logging_level>
        <reblock>false</reblock>
        <run_as_copy>false</run_as_copy>
        <skip_archive>false</skip_archive>
        <skip_stats>false</skip_stats>
        <concurrent_bld_per_tbl>5</concurrent_bld_per_tbl>
    </job_options>
</dscCreateJob>
```

16. Run the job.
    `dsc run_job -n R-TDSERVER`
17. On the destination system, start the DIP and run the DIPALL script.
18. On the destination system, from the DBW console supervisor screen, enable logons for all users.
    `enable logons`

Troubleshooting a Full System Restore

If you have problems with a full system restore, try the following.

Verify the Media Server IP Addresses

1. Ping the IP address from the media server to the database.
2. Ping the IP address from the database to the media server.
3. Trim any invalid or unused IP addresses.

post_dbc_restore Script Fails to Launch

If the restore works, but fails to launch the post_dbc_restore script, you receive a message similar to the following:

```
1079 Post dbc restore script not started. Please run the post_dbc_restore
script manually with the following command:
/opt/teradata/PUTTools/td17<version>/IUMB_scripts/post_dbc_restore -s -N -j
3 TDT5WSDB/'dbc',<dbc_password>.
```

In addition, the table SYSBAR.DSAConnectionsTbl is not repopulated and you receive a message similar to this:

```
1001   SQL Exception 3807 3807: DBCSQL Exception: Object
'SYSBAR.DSAConnectionsTbl' does not exist.
DSAConnectionsTbl INSERT failed.  All DSCs should activate system <system_name>
```

Follow these steps:

1. Manually launch the post_dbc_restore script:

   `/opt/teradata/PUTTools/td17<version>/IUMB_scripts/post_dbc_restore -s -N -j 3 TDT5WSDB/'dbc',<dbc_password>`

2. Repopulate the DSAConnectionsTbl:

Interface	Steps
BAR Setup portlet	Check the activation status of the destination system in the **BAR Setup** portlet and do one of the following: • If the destination system is configured and enabled, click **Update** for **Selector: Update system selector for JMS messages** • If the destination system is not configured, add the system and click **Apply**.
Command line	Check the activation status of the system: `dsc list_components -t system` and do one of the following: • If the destination system is configured and enabled, run this command: `dsc config_systems -f configsystem.xml -s selector` • If the destination system is not configured, add the system: `dsc config_systems -f configsystem.xml` where `configsystem.xml` is similar to the example shown in Full System Restore Using the Command Line.

3. On the destination system, start DSMain from the Database Window (DBW) console supervisor screen by entering:

   `start bardsmain -d` *DSCName*, where *DSCName* is the unique name of your DSC server

Compile Errors Reported During a Restore

When restored by DSA, certain Teradata-provided stored procedures fail to recompile in the post_data_restore scripts. These procedures have dependencies that are not met until after DIPALL is run at the conclusion of the migration. You can can manually recompile them using ALTER statements after DIPALL completes.

"SYSLIB"."JSON_SHRED_BATCH_U"
"SYSLIB"."JSON_SHRED_BATCH"
"TD_SYSXML"."XSLT_SHRED_BATCH"
"TD_SYSXML"."AS_SHRED_BATCH"
"TD_SYSXML"."XSLT_SHRED"
"TD_SYSXML"."XMLPUBLISH"
"TD_SYSXML"."XMLPUBLISH_STREAM"
"LOCKLOGSHREDDER"."SP_LockLog_Shredder"

These errors are detected by the DSC when it runs the post_data_restore scripts and it concludes with a Warning job status rather than Complete. Check the log file noted in the error message provided by the DSC. The errors flagged by the DSC for these issues will not interfere with the rest of the restore.

The following commands will recompile the stored procedures after DIPALL has been run.

```
ALTER PROCEDURE "SYSLIB"."JSON_SHRED_BATCH_U" COMPILE;
ALTER PROCEDURE "SYSLIB"."JSON_SHRED_BATCH" COMPILE;
ALTER PROCEDURE "TD_SYSXML"."AS_SHRED_BATCH" COMPILE;
ALTER PROCEDURE "TD_SYSXML"."XSLT_SHRED" COMPILE;
ALTER PROCEDURE "TD_SYSXML"."XSLT_SHRED_BATCH" COMPILE;
ALTER PROCEDURE "TD_SYSXML"."XMLPUBLISH" COMPILE;
ALTER PROCEDURE "TD_SYSXML"."XMLPUBLISH_STREAM" COMPILE;
ALTER PROCEDURE "LOCKLOGSHREDDER"."SP_LockLog_Shredder" COMPILE;
```

Warning When the Database Exceeds Logical Space Limit

When the database exceeds its logical space limit while doing a restore operation, warning messages may appear. You can ignore the messages as the restore does not fail. However, when the restore is complete, you must modify the perm space to add objects to the database or to add new rows to the tables.

The warning message looks like this:

```
Warning code   Warning message
-------------------------------------------------------------------------------
1196           Database <database_name> has exceeded it's logical space limit.
```

Creating a Diagnostic Bundle for Support

Teradata DSA provides a command-line, interactive script to collect:

- Current time on the server
- Component versions, for Analytics Database/Teradata Database, DSC, DSA Network Client, NetBackup, Spectrum Protect, Viewpoint, BAR Command Line, ActiveMQ
- General system configurations, such as the number of media servers, DSC server status
- Property files and logs for the DSC component and the DSA Network Client (ClientHandler)

This information is useful for troubleshooting by Teradata Services.

The `dsa_support_info.sh` script is installed in the root directory associated with a DSA installation package:

- `$DSA_DSC_ROOT` on the DSC server
- `$CLIENTHANDLER_ROOT` on a media server
- `$BARCMDLINE_ROOT` on the command line server

1. [Optional] To set up an SSH trust relationship between the DSC server and BAR servers or remote servers, run `dsatrust.sh` with the following syntax:
   `dsatrust.sh [-a] [-l <hosts>] [-u user] [-v] [-h]`

Parameter	Description
-a	Sets trusted relations between the DSC server and other BAR servers.

Parameter	Description
-l *hosts*	Sets trusted relations to the listed hosts. Host names must be separated by commas.
-u *user*	Specify login user name. You must use this parameter with the -a or -l parameter.
-v	Displays the script version.
-h	Displays help.

2. Run `dsa_support_info.sh` on a server to collect local information, using the following syntax:
`dsa_support_info.sh [-b <nbu jid>][-c][-d <path>][g][-h][-i][-j <job>][-s <timestamp>][-t][-v][-x]`

Parameter	Description
-b *nbu jobid*	Collects NetBackup job information. Enter the NetBackup job ID as *nbu jobid*.
-c	Collects core dumps for the DSA Network Client (ClientHandler).
-d *path*	Specifies the dump directory path. Default: `/var/opt/teradata/dsa/support/`*hostname_current timestamp*
-g	Collects general support information.
-h	Displays help.
-i	Collects installation logs.
-j *job*	Collect job information. Enter the job name as *job*.
-s *timestamp*	Collects logs for the date and time specified in the timestamp. Specify the timestamp in the following format, for example, `2015-05-15 09:50`.
-t	Triggers a TVI event. If the -t parameter has been used to trigger a TVI event, the data is zipped. If the 50 MB limit per TVI event is exceeded, the support bundle is split into multiple zip files, and a separate TVI event is triggered for each zip file. The zip files are copied to `/var/opt/teradata/SupportBundle/`. You can access the unzipped files in the dump directory specified by the -d parameter.
-v	Displays the script version.
-x	Bypasses all user prompts.

After the script runs, files specific to the local server are output and can include:

- Component information in `.out` files
- Job output xml and `.out` files
- Log files

- Property files
- DSA Network Client (ClientHandler) core dumps

The output directory is `/var/opt/teradata/dsa/support/`*hostname_current timestamp* or a user-specified directory.

Protecting the DSC Repository

The DSC repository stores all DSA data, including configuration definitions and settings, job definitions, job status, and job history. Protecting the data in the DSC repository is critical. Without the DSC repository, database backup data sets cannot be restored.

Back up the DSC repository every day after all of the daily database backup jobs have completed. The DSC backup data set must also be included in the disaster recovery and data protection policies of the organization. This includes any vaulting or off-site storage of backup datasets.

1. Configure a target group for your DSC repository backup.
2. Preferred is to schedule a DSC repository backup using one of the following methods:
   - BAR Portlet: Scheduling Automatic Repository Backups
   - Command Line: Scheduling and Configuring a Repository Backup
3. Back up the repository to the landing zone and store the `boostrap_`*backupid*`.out` in a safe known location to be imported back into DSC in case of a disaster.
4. If you do not schedule the backups automatically, manually back up the repository using one of these methods:
   - BAR Portlet: Backing Up the Repository Manually (Portlets)
   - Command Line: Backing Up the DSC Repository with the Command Line

Backing Up the Repository to the Landing Zone: Command Line

1. Back up the repository to the landing zone.
   `dsc run_repository_job -t backup`
   The backup creates two files:
   - `bar_`*backupid*`.out` Contains the data to restore the non-barbackup tables to a certain point in time.
   - `bootstrap_`*backupid*`.out` Contains the data to recreate the barbackup tables for disaster recovery.
2. Verify the operation and retrieve the backup ID using one of these commands:
   `dsc save_sets_replication_status`
   From inside the landing zone: `ls -lrt` retrieves the backup ID of the two most recent files.

Backing Up and Replicating the Repository: Command Line

1. Back up the repository to the landing zone and replicate it to the third party storage..
   `dsc run_repository_job -t backup -r` *replication_target_group*

The backup creates two files that are created in the landing zone:

- `bar_`*backupid*`.out` Contains the data to restore the non-barbackup tables to a certain point in time. This file is replicated into the third party storage defined in the target group. Then it is removed from the landing zone.
- `bootstrap_`*backupid*`.out` Contains the data to recreate the barbackup tables for disaster recovery. This file remains in the landing zone.

2. Verify the operation and retrieve the backup ID using one of these commands:
   `dsc save_sets_replication_status`
   From inside the landing zone: `ls -lrt` retrieves the backup ID of the two most recent files.

Replicating an Existing Repository Backup to Third Party Storage: Command Line

1. Retrieve the backup ID using one of these commands:
   `dsc save_sets_replication_status`
   From inside the landing zone: `ls -lrt` retrieves the backup ID of the two most recent files.
2. Replicate the backup to the third party storage target group:
   `dsc run_repository_job -t replicate_to -v` *backupid* `-r` *replication_target_group*
   The replication creates a bootstrap file and replicates the existing bar file:
   - `bar_`*backupid*`.out` Contains the data to restore the non-barbackup tables to a certain point in time. This file is replicated into the third party storage defined in the target group. Then it is removed from the landing zone.
   - `bootstrap_`*backupid*`.out` Contains the data to recreate the barbackup tables for disaster recovery. This file remains in the landing zone.

Restoring the Postgres DSC Repository When the Boostrap File Is Available: Command Line

1. Copy the bootstrap_<backupID>.out into the landing zone
   (default is: /var/opt/teradata/dsa/postgres) and run the following commands.
2. Navigate to /opt/teradata/client/17.10/dsa/dsc:
   cd $DSA_DSC_ROOT
3. User:
   `su dscuser`
4. Restore the repository:
   `sh recover.sh -m recovery`
5. Check the replication status:
   `sh recover.sh -m save_sets_replication_status`
6. Get the bar_<backupID>.out from replication target group:
   `sh recover.sh -m replicate_from -v <backupID> -r <replicationTargetGroupName>`
   For example, sh recover.sh -m replicate_from -v 1602081495349 -r TGP

7. Check the replication status:
   `sh recover.sh -m save_sets_replication_status`
8. Restore the backup:
   `sh recover.sh -m restore -v <backupID>`
   For example, sh recover.sh -m restore -v 1602081495349
9. Exit dscuser:
   `exit`
10. Restart DSC:
    `/etc/init.d/dsc restart`

Restoring the Postgres DSC Reporitory When Any or Old Bootstrap File is Available: Command Line

1. Copy the bootstrap_<backupID>.out into the landing zone and run the following commands. (default is: /var/opt/teradata/dsa/postgres)
2. Navigate to /opt/teradata/client/17.06/dsa/dsc: cd $DSA_DSC_ROOT
3. User:
   `su dscuser`
4. Restore the repository:
   `sh recover.sh -m recovery`
5. Check the list of backup IDs available on the replication target group:
   `sh recover.sh -m list_backupids -r replicating_target_group`
6. Get the bar_<backupID>.out from replication target group:
   `sh recover.sh -m replicate_from -v <backupID> -r <replicationTargetGroupName>`
   For example, sh recover.sh -m replicate_from -v 1602081495349 -r TGP
7. Check the replication status:
   `sh recover.sh -m save_sets_replication_status`
8. Restore the backup:
   `sh recover.sh -m restore -v <backupID>`
   For example, sh recover.sh -m restore -v 1602081495349
9. Exit dscuser:
   `exit`
10. Restart DSC:
    `/etc/init.d/dsc restart`

Restoring the Postgres DSC Repository When the Bootstrap File is Missing: Command Line

For the first three steps, use the files you stored following the procedure in Saving the DSC Repository Configuration Information

1. Configure media server.

For example, dsc config_media_servers -f 31.media_server.xml

2. Configure backup application.
   For example, dsc backup_application_gcp -f config_gcp_user.xml
3. Configure target group for replication.
   For example, dsc config_target_groups -f tg_gcp.xml -t target_gcp -r
4. Navigate to /opt/teradata/client/17.06/dsa/dsc: cd $DSA_DSC_ROOT
5. User:
   `su dscuser`
6. Check the list of backup IDs available on the replication target group:
   `sh recover.sh -m list_backupids -r replicating_target_group`
7. Get the bar_<backupID>.out from replication target group:
   `sh recover.sh -m replicate_from -v <backupID> -r <replicationTargetGroupName>`
   For example, sh recover.sh -m replicate_from -v 1603437512425 -r gcp_tg
8. Restore the backup.
   `sh recover.sh -m restore -v <backupID>`
   For example, sh recover.sh -m restore -v 1603437512425
9. Exit dscuser:
   `exit`
10. Restart DSC:
    `/etc/init.d/dsc restart`

Recovering the DSC When the Database is Corrupt: Command Line

1. Run recovery using the `recover.sh` script.
   `sh recover.sh -m recovery`
2. Run replicate_from using the `recover.sh` script.
   `sh recover.sh -m replicate_from -r` *replication_target_group* `-v` *backupid*
3. Run restore using the `recover.sh` script.
   `sh recover.sh -m restore -v` *backupid*
4. Check the restore status:
   `dsc save_sets_replication_status`

Recovering the DSC When the Database is Corrupt and Bootstrap File is Missing: Command Line

1. Manually copy `bar_`*backupid*`.out` in the landing zone and verify it is owned by dscuser.users.
2. Generate `bootstrap_`*backupid*`.out` using the `recover.sh` script.
   `sh recover.sh -m generate_bootstrap`
3. Run restore using the `recover.sh` script.
   `sh recover.sh -m restore -v` *backupid*

4. Check the restore status:
   ```
   dsc save_sets_replication_status
   ```

Time Zone String Management

Time Zone Strings allow Analytics Database to automatically adjust for Daylight Savings Time changes, for locales that observe bi-annual time changes.

If the destination system has a Time Zone String setting, you must disable it before running the DIPMIG script. After running the DIPMIG script, you can re-enable the Time Zone String on the destination system.

Checking for a Time Zone String

Before running the DIPMIG script, use the DBS Control utility to check if a Time Zone String is set on the destination system.

1. From a command prompt on the destination system, enter `dbscontrol` to start the DBS Control utility. For more information about the DBS Control utility, see *Teradata Vantage™ - Database Utilities*, B035-1102.
2. At the DBS Control prompt, enter `di gen` to display the General DBS Control fields.
3. Inspect General field 18 to see if there is a value for TimeZoneString.
   - If the value showing for `System TimeZoneString` is `Not Set`, there is no Time Zone String set on the destination system, and you do not need to make any Time Zone String changes.
   - If there is any other value showing for `System TimeZoneString`, you need to remove the Time Zone String prior to restoring data. For example:

     ```
     18. System TimeZoneString = America Pacific
     ```

Restoring the Time Zone String

If you removed the Time Zone String on the destination system before running the DIPMIG script, restore the string after you run it.

1. On the destination system, access the directory where you saved the copy of tdlocaledef.txt. For example:

   ```
   # cd /opt/teradata/tdat/tdbms/XX.XX.XX.XX/etc/
   ```

2. Delete the tdlocaledef.txt file you edited to remove the Time Zone String, and rename the original tdlocaledef.txt.orig file to tdlocaledef.txt. For example:

   ```
   # rm tdlocaledef.txt

   # mv tdlocaledef.txt.orig tdlocaledef.txt
   ```

3. Run the tdlocaledef utility to compile the tdlocaledef.txt file.

```
# /usr/tdbms/bin/tdlocaledef -input  tdlocaledef.txt  -output new
```

4. Run `tpareset` to restart the database and put the new tdlocaledef settings into effect, restoring the original Time Zone String:

   ```
   # tpareset -f set the Time Zone String
   ```

5. Run the DBS Control utility and display the General fields to confirm that the Time Zone String has been restored.

   ```
   # dbscontrol

   Enter a command, HELP, or QUIT:
   di gen
   ```

   Confirm that the value for field 18 is no longer `Not Set`.

   ```
    18. System TimeZone String         = America Pacific
   ```

Removing the Time Zone String

If the destination system has a Time Zone String, remove the string before running the DIPMIG script. The Time Zone String is defined in a text file, tdlocaledef.txt, that defines locale-specific information for the database system, such as the currency and numeric units, names of days of the week and months, and default formats for different data types. The tdlocaledef.txt file is compiled by the Teradata Locale Definition utility in order to take effect on the database system. For more information about the Teradata Locale Definition utility, see *Teradata Vantage™ - Database Utilities*, B035-1102.

1. On the destination system, locate the directory for the `tdlocaledef.txt` file, and change to that directory. For example:

   ```
   # locate tdlocaledef.txt
   /opt/teradata/tdat/tdbms/XX.XX.XX.XX/etc/tdlocaledef.txt

   # cd /opt/teradata/tdat/tdbms/XX.XX.XX.XX/etc/
   ```

2. Save a copy of the current `tdlocaledef.txt` file. For example:

   ```
   # cp  tdlocaledef.txt  tdlocaledef.txt.orig
   ```

3. Use a text editor to edit tdlocaledef.txt, and remove the value for `TimeZoneString`, leaving only the quotation marks.
   For example, change a value that looks like this:

   ```
   TimeZoneString {"America Pacific"; "-8"; "0"; "2"; "4"; "4"; "1"; "0"; "0";
   "02:00:00"; "3"; "10"; "0"; "0"; "-1"; "02:00:00"; "1987"; "2006"; "-8";
   ```

```
"0"; "-7"; "0"; "4"; "3"; "8"; "0"; "0"; "02:00:00"; "4"; "11"; "1"; "0";
"0"; "02:00:00";"2007"; "9999"; "-8"; "0"; "-7"; "0"}
```

to this:

```
TimeZoneString {""}
```

Save the modified tdlocaledef.txt file.

4. Run the tdlocaledef utility to compile the tdlocaledef.txt file.

```
# /usr/tdbms/bin/tdlocaledef -input  tdlocaledef.txt  -output new
```

5. Run `tpareset` to restart the database and put the new tdlocaledef settings into effect, removing the old Time Zone String:

```
# tpareset -f Removed Time Zone String
```

6. Run the DBS Control utility and display the General fields to confirm that the Time Zone String has been removed.

```
# dbscontrol

Enter a command, HELP, or QUIT:
di gen
```

Confirm that the value for field 18 shows as `Not Set`.

Toggling SSL or TCP after Installation

If you have SSL set up and you want to switch to TCP, you can use these steps with the toggle script. You also must remove and re-add the DSC Server in the **BAR Setup** portlet.

If you want to switch from TCP to SSL, you need to follow the steps in *Enabling SSL - Setting Up Self-Signed Keys and Certificates* in your installation guide. See *Teradata® DSA - DSU Installation, Configuration, and Upgrade Guide*, *Teradata® DSA - DSE for Veritas NetBackup Installation, Configuration, and Upgrade Guide*, or *Teradata® DSA - DSE for IBM Spectrum Protect Installation, Configuration, and Upgrade Guide*.

1. Remove the DSC Server using the **BAR Setup** portlet (see Removing a DSC Server).
2. To toggle SSL for each client component (DSC, DSARest, BARCmdline, and ClientHandler), run the toggle script on each client.
   `./ssl_jms_toggle.sh` is located in each client's installed directory:
   - `$DSA_DSC_ROOT`
   - `$BARCMDLINE_ROOT`
   - `$CLIENTHANDLER_ROOT`

Syntax: ssl_jms_toggle.sh [**DSC** | **DSARest** | **BARCmdline** | **ClientHandler**] [**tcp** | **ssl**] [**61616** | **61617**]
For example: `ssl_jms_toggle.sh DSC ssl 61617`
For ClientHandler the following is true:

- The script prompts you for the SSL keystore password. This is the client's keystore password.

  Important:

  If DSARest web service is https, the SSL keystore password must match the DSARest web service keystore password.

  If you want to switch from TCP to SSL, you need to follow the steps in *Enabling SSL - Setting Up Self-Signed Keys and Certificates* in your installation guide. See *Teradata® DSA - DSU Installation, Configuration, and Upgrade Guide*, *Teradata® DSA - DSE for Veritas NetBackup Installation, Configuration, and Upgrade Guide*, or *Teradata® DSA - DSE for IBM Spectrum Protect Installation, Configuration, and Upgrade Guide*.

- If `clienthandler.properties` contains multiple brokers, the script asks if you want to use the same port number for all brokers.
  - y - All port numbers are changed to the number listed in the command
  - n - Port numbers are left as is. To change any broker port numbers, you must change them in `broker.list` in `clienthandler.properties`.

3. Add the DSC server using the **BAR Setup** portlet (see Enabling or Adding a DSC Server), select SSL or TCP as the Broker Connectivity and Broker Port.
4. From the **BAR Setup** portlet, select your DSC Server, and click **Systems and Nodes** under **Categories**.
5. Select the system name under **Systems**, then **System Details** under **Setup**.
6. To enable SSL, under **SSL Communication**:
   a. Check the **Enable SSL over JMS Communication** box.
   b. Enter the keystore password in the **Truststore Password** box.
   c. Click **Apply**, then follow on-screen instructions.
7. To configure TCP, under **SSL Communication**:
   a. Clear the **Enable SSL over JMS Communication** box.
   b. Remove the keystore password from the **Truststore Password** box.
   c. Click **Apply**, then follow on-screen instructions.
8. Restart the DSMain process on DSC server for the repository database cnsterm 6:
   a. Stop bardsmain:
      ```
      cnsterm 6
      start bardsmain -s
      ```
   b. Start bardsmain:
      ```
      cnsterm 6
      ```

```
start bardsmain
```

Running DSA Jobs Using Crontab

You can schedule and run DSA jobs from Crontab. However, you must first create a script with specific information and create a Crontab entry.

1. In your own directory, create a new script with the following information. Make sure the permission of this script is `755: # chmod` *script* `755:`

```
#!/bin/bash

# This will identify where BARCmdline is installed
# If non-TMSB it will also contain the path to JAVA
source /etc/profile.d/barcmdline-profile.sh
# These will identify where JAVA is on TMSB Servers
if [ -f /etc/profile.d/teradata-jdk11.sh ]; then
 source /etc/profile.d/teradata-jdk11.sh > /dev/null 2>&1
fi
# Here specify the specific job you want to run
dsc run_job -n
```

2. Edit Crontab:
   `# crontab -e`
3. Use Crontab syntax to create an entry at the beginning of the Crontab script.
4. In the Cron entry, run the script you just created.

Scheduling Jobs Using NetBackup

You must initially configure NetBackup policies and schedule policies in the NetBackup Administration Console before using the **BAR Setup** portlet or the command-line interface. For example, you must use the NetBackup Administration Console to add:

- A policy before using the **BAR Setup** portlet to assign a policy to a storage device
- A schedule policy before using the tdbackex script to schedule a job

NetBackup Policy Configuration

A NetBackup policy defines the backup criteria for a specific group of one or more clients and is required for any scheduled job.

To use Teradata DSA, define at least one Teradata policy. A configuration can have a single policy that includes all clients or there can be many policies, some of which include only one client. For DSA, the standard and recommended setup is to assign one NetBackup policy per device or file target.

Policy Requirements

When creating a policy, specify the following:

- Storage unit and media to use
- Policy Type as Teradata
- Backup schedule
- Script files to run on the clients
- Clients to back up

In addition to the attributes described here, there are other policy attributes to consider. Refer to the *NetBackup System Administrator's Guide* for detailed configuration instructions and information on all available attributes.

NetBackup Schedule Policy Configuration

A NetBackup schedule policy allows automated scheduling of DSA backup jobs. After you have created a schedule policy using the NetBackup Administration Console, the schedule policy uses the tdbackex script to initiate the scheduled backup jobs. A schedule policy is different than the policy that defines devices and dataset retention levels used in a backup job.

A NetBackup schedule policy can be configured to run one or more backup jobs at a designated time, or multiple schedules can be created for a single policy.

When scheduling full, delta, and cumulative backup jobs, at least three policy schedules must be created: one policy contains the schedule for the full backup, one policy contains the schedule for the delta backup, and one policy contains the schedule for the cumulative backup.

With delta or cumulative backup jobs, the latest full backup data set must be available to restore a subsequent delta or cumulative data set. This means that the full backup dataset and succeeding delta or cumulative backup data set must be retained until the latest delta or cumulative data set no longer need to be restored, and a new full backup data set is present.

Schedule Policy Requirements

When creating a schedule policy, specify the following:

- Host name server value as the media server on which the DSA BAR command-line software is installed and configured
- Policy Type as Teradata
- Schedule type as Automatic Backup
- Schedule properties
- Add Client (the client is the media server on which the DSA BAR command-line software is installed)
- Add Backup Selection and enter the job name and backup type (FULL, DELTA, or CUMULATIVE) that will be executed by the selected schedule policy

DSA Job Migration to a Different Domain

You can migrate backup save sets pointing to a specific DSA backup job in one DSA domain to a different DSA domain. The Administrator user can export backup job metadata (job definition, save sets, and targets) in XML format using the `export_job_metadata` command. The exported backup job metadata files, together with the storage media, can then be sent to the site where the other DSA domain is located. Users can use the `import_job_metadata` command on the DSA command line to create the same job and associated targets and save sets with the exported backup job metadata XML. The import of job metadata is handled by the other DSA environment in the following order:

1. Virtual systems, media servers, NetBackup servers, and targets are saved into the DSC Repository.
2. The job definition is saved into the DSC repository.
3. Job save sets are saved into the DSC repository.

After the job metadata has been successfully imported using the command `import_job_metadata`, use the `config_target_group_map` command to map the exported virtual targets to one of the physical targets in the new DSA domain. You can use the new backup job that was imported to create a restore job using the mapped target.

You can generate a job plan that uses multiple task-sets based on the streams soft limits set on the source and target. For best performance, use one-to-one mappings from the tapes or disk storage through the new target groups and streams passing the data to the AMPs. You can temporarily modify the stream soft limits and target groups to match on the source and destination system.

Restoring to a Different DSA Domain

Prerequisite:

For a backup save set to be restored to a different DSA domain:

- The backup job must have completed with a status of COMPLETED_SUCCESSFULLY, COMPLETED_ERRORS, or WARNING in the original DSA configuration.
- NetBackup save sets must not be expired.
- The DSA user ID and user account must be the same in the new domain as in the original domain because the new domain requires the same permissions to access the files created by the original domain. The default DSA user ID is 600. The default DSA user account is dscuser.
- Job names must be unique in a DSA domain. For example, `Prod_test1` cannot be imported into different domains if a job named `Prod_test1` already exists in the target domain without resulting in an error. However, multiple versions of the same job name can be imported into different domains.

These steps restore backup save sets to a different DSA domain.

1. In the original DSA domain, export the backup job definition and associated targets and save sets using the `export_job_metadata` command.

   When the `export_job_metadata` or `import_job_metadata` command is running, no other operations or commands can run at the same time. In addition, `export_job_metadata` and `import_job_metadata` cannot run if an operational job or repository job is running.
2. Query the backup IDs.
3. Migrate the exported metadata files.
   The exported job metadata files must exist on a DSA Teradata Managed Server for BAR in the destination DSA environment. If network connectivity exists between the two environments, copy the files using the file transfer mechanism allowed between the environments and ensure the time/date stamps are preserved. Alternately, the target environment can perform an NFS mount of the source disk file system.
4. Migrate the images to the new DSA location.
   For NetBackup, you can transfer the backup image data by physical tape transfer or electronic media transfer:
   - Physical tape transfer: Export the NetBackup images using the NetBackup feature for exporting tapes. The NetBackup save sets must not be expired.
   - Electronic media transfer: If NetBackup Auto-Image Replication (AIR) is active between the two domains, the images are automatically migrated.

   For DSU, if network connectivity exists between the two environments, copy the save set files using the file transfer mechanism allowed between environments and ensure the time/date stamps are preserved. Alternately, the target environment can perform an NFS mount of the source disk file system.
   When a Data Domain unit is used as an NFS server or with DD Boost, replicate the images from source to target using mtree replication. See Data Domain documentation for instructions on how to set up mtree replication.
5. In the new DSA domain, import the backup job definition and associated targets and save sets using the `import_job_metadata` command.
   The new DSA domain requires physical systems, media servers, NetBackup servers, and target groups to be configured so that you can import data to the new domain.
6. In the new DSA domain, map the virtual target group to a physical target group using the -V option with the `config_target_group_map` command.
7. In the new DSA domain, run `list_jobs -t migrated` to see the migrated jobs.
8. Run `list_components -e true` to see available system and target groups.
   a. If the system and target groups required for the restore job are not available, run `config_systems` and `config_target_groups` to enable them.
9. Create a restore job based on the migrated backup job.

Exporting Job Metadata

To migrate a job to another DSA environment, you must first export the job metadata (job definition, targets, and save sets).

- Type `dsc export_job_metadata` using only the name parameter to export all metadata files for the job, and press **Enter**.

  Parameters include:

Parameters	Description
n|name *Name*	The name of the job on which to perform the action. The job must be active.
d|directory *DirectoryPath*	[Optional] Directory where the files are exported to or imported from.
v|backup_ version *BackupVersion*	[Optional] Backup version number. Enter `latest` or `0` for the latest save set.
t|type *Type*	[Optional] The type of job metadata. You may enter `JOB` (for the job definition), `SAVESET`, or `TARGET`. If no value is specified, all three types of metadata are included.
u|user_ authentication *User*	Required when security management is enabled. Supplies the command with the Viewpoint user, and triggers a password prompt for authentication.

Querying NetBackup Backup IDs

NetBackup requires the backup IDs associated with the backup save set when save set images are being transferred. To get the IDs, you must query, then list the backup IDs.

Note:

You can use the `query_nbu_backupids` command in this procedure, or the `query_backupids` command to retrieve NetBackup backup IDs.

---

1. Type `dsc query_nbu_backupids -n` *name*.
   ```
   cbar6:/ # dsc query_nbu_backupids -n SDBA_Weekly
   ```
   In the example, NetBackup is queried for the backup IDs associated with the SDBA_Weekly save set. The IDs are stored in the DSC repository.
2. To list the backup IDs, type `list_query_nbu_backupids -n` *name*.
   ```
   cbar6:/ # dsc list_query_nbu_backupids -n SDBA_Weekly
   ```
   The following example resulted from the query to list backup IDs associated with the SDBA_Weekly save set.

   ```
   Data Stream Controller Command Line 15.00.00.03
   Command parameters:
   ```

```
-name : SDBA_Weekly
Connected to DSC version 15.00.00.03
Listing query NBU backup ids...

Job Name: SDBA_Weekly
Job Execution Id: 1


Media Server: cbar6

Backup ID                    Logical File Name                File Size
Date
------------------------------------------------------------------------------
-----------
cbar6_1413483708   SDBA_Weekly_1_1_file10_data_1413483639839   98.3 MB
10/16/14 11:21:48 AM
cbar6_1413483668   SDBA_Weekly_1_1_file11_data_1413483639839   104 MB
10/16/14 11:21:08 AM
cbar6_1413483720   SDBA_Weekly_1_1_file12_data_1413483639839   102 MB
10/16/14 11:22:00 AM
cbar6_1413483666   SDBA_Weekly_1_1_file13_data_1413483639839   92.4 MB
10/16/14 11:21:06 AM
cbar6_1413483675   SDBA_Weekly_1_1_file14_data_1413483639839   100 MB
10/16/14 11:21:15 AM
cbar6_1413483673   SDBA_Weekly_1_1_file15_data_1413483639839   95.1 MB
10/16/14 11:21:13 AM
…….
cbar6_1413483711   SDBA_Weekly_1_1_file95_data_1413483639839   103 MB
10/16/14 11:21:51 AM
cbar6_1413483715   SDBA_Weekly_1_1_file96_data_1413483639839   101 MB
10/16/14 11:21:55 AM
cbar6_1413483709   SDBA_Weekly_1_1_file9_data_1413483639839    98.8 MB
10/16/14 11:21:49 AM
cbar6_1413483647   SDBA_Weekly_1_1_file_dict_1413483639839     88.4 KB
10/16/14 11:20:47 AM
```

The SDBA_Weekly save set was generated from a 96 stream backup job. There are 97 backup IDs associated with the save set: 96 data stream files and 1 dictionary stream file.

Querying Backup IDs

The backup ID associated with a save set is required when a save set image is transferred. To get the IDs, you must query, then list the backup IDs. The `query_backupids` command does not return any data. You must run the `list_query_backupids` command to retrieve the IDs.

You can use the following procedure to retrieve DD Boost, disk file system, and NetBackup backup IDs.

1. Type `dsc query_backupids -n` *name*.
2. To list the backup ID, type `list_query_backupids -n` *name*.

Importing Job Metadata

Importing the metadata of a backup job (job definition, targets, and save sets) enables you to restore it in a different DSA environment.

The data must be imported in the following order:

1. Targets
2. Job definition
3. Save sets

- Type `dsc import_job_metadata` using only the name parameter to import all metadata files for the job, and press **Enter**.
  Parameters include:

Parameters	Description
n|name *Name*	The name of the job on which to perform the action. The job must be active.
d|directory *DirectoryPath*	[Optional] Directory where the files are exported to or imported from.
v|backup_ version *BackupVersion*	[Optional] Backup version number. Enter `latest` or `0` for the latest save set.
t|type *Type*	[Optional] The type of job metadata. You may enter `JOB` (for the job definition), `SAVESET`, or `TARGET`. If no value is specified, all three types of metadata are included.
u|user_ authentication *User*	Required when security management is enabled. Supplies the command with the Viewpoint user, and triggers a password prompt for authentication.

Updating a Target Group Map

To access imported save sets, the target group used in the source DSA domain must be mapped to a target group in the destination DSA domain.

1. To view the virtual components that are available for target group mapping, type
   `dsc list_components -t` *system* `-V`
   `dsc list_components -t nbu_server -V`
2. To map a virtual target group to an existing, physical target group in the destination domain, type
   `dsc config_target_group_map -f` *file* `-V`
   `cbar3:/SDBA_Weekly_imports # dsc config_target_group_map -f cbar6.xml -V`

Sample XML files are located at `$BARCMDLINE_ROOT/samples/sample_target_mapping`. They include helpful comments and show the available settings for the corresponding commands.

Target Group Map Tag	Description
master_source_target_name	The backup target group name.
master_source_dsc_id	The ID of the source DSC repository.
master_dest_target_name	The restore target group name.
target_group_maps	A sub target grouping consisting of a backup and restore configuration pair.
source_mediaserver_name	The backup media server.
source_policy_class_name	The third party backup application policy associated with the backup media server.
dest_mediaserver_name	The restore media server.
dest_policy_class_name	The third party backup application policy associated with the restore media server.

3. To list the target group map, type:
   `dsc list_target_group_map -V`
   ```
   cbar3:/SDBA_Weekly_imports # dsc list_target_group_map -V
   ```

Restoring a Migrated Job

When creating a restore job from the migrated save set, the restore job must specify the target group that was mapped in the source and destination domains.

If the save set is being restored to a database system that is different than the source, set the `run_as_copy` option to `true`. The database object must exist on the destination system with perm space allocated before the copy job runs.

1. Type `dsc create_job` followed by the parameters, and press **Enter**.

Parameter	Description
n|name *Name*	The name of the job on which to perform the action. Must be unique for each job.
d|description *Description*	A meaningful description of the job. To allow a multiword description, add \" before and after the description string. `\"A description of Job 1\"`
t|type *Type*	`restore`
run_as_copy	`true`
b|backup_ name *BackupName*	[Optional] An existing backup job name. For restore, analyze_read, and analyze_validate jobs only.

Parameter	Description
v|backup_ version *BackupVersion*	[Optional] Backup version number. For restore, analyze_read, and analyze_validate jobs only. Enter `latest` or `0` for the latest save set.
f|file *filename*	The full file path and file name of the file containing the necessary parameters to create the job. If the same parameters are provided both in the file and on the command line, Teradata DSA uses the values specified in the command line.
u|user_ authentication *User*	Required when security management is enabled. Supplies the command with the Viewpoint user, and triggers a password prompt for authentication.

2. Type the user name and password, and press **Enter**.
3. Type `dsc run_job` and press **Enter**.

Restoring a Migrated Job

The following example restores a migrated job called SDBA_Weekly.

Note the following restrictions on job options:

- If single_object_locking is true and online is true, the backup job fails.
- If single_object_locking is true and data_phase is DICTIONARY, the backup job fails.
- If single_object_locking is true, a DBC-only backup job is rejected.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<dscCreateJob xmlns="http://schemas.teradata.com/v2012/DSC">
    <job_instance>
        <job_name>rstr_air_job</job_name>
        <job_description></job_description>
        <job_type>RESTORE</job_type>
        <job_owner>dscuser</job_owner>
        <job_state>ACTIVE</job_state>
        <auto_retire>false</auto_retire>
        <backup_name>SDBA_Weekly</backup_name>
        <backup_version>0</backup_version>
        <all_backup_objects>true</all_backup_objects>
    </job_instance>
    <source_media>tg_cbar3</source_media>
    <target_tdpid>recall</target_tdpid>
    <job_options>
        <enable_temperature_override>false</enable_temperature_override>
        <temperature_override>DEFAULT</temperature_override>
        <block_level_compression>DEFAULT</block_level_compression>
        <disable_fallback>false</disable_fallback>
        <query_band></query_band>
```

```
        <dsmain_logging_level>Error</dsmain_logging_level>
        <reblock>false</reblock>
        <run_as_copy>true</run_as_copy>
    </job_options>
</dscCreateJob>
```

Using the export_import Tool to Automate DSA Job Migration to a Different Domain

This tool can export and/or import job metadata from Source DSC to Target DSC. To use this tool, you need to make sure configuration of Target DSC is similar to Source DSC (for example; Target group, Systems and Media server).

The system currently supports configuration of migrated jobs on Target DSC for DFS target group. Use this tool to create or run restore job for the MIGRATED job.

Pre-requisites

Python 3.4+ installed using "pip3 install -r requirements.txt".

Usage

```
python3 job_metadata.py -SourceDSCname SourceDSCNAME -SourceDSCusername
SourceDSCUSERNAME -TargetDSCname TargetDSCNAME -TargetDSCusername TargetDSCUSERNAME
-JobName JOBNAME -path PATH [-Type TYPE] [-config_target_group_map YES/NO]
```

Mandatory arguments

Option	Value	Description
-SourceDSCname	{DSC Name}	Name of the Source DSC
-SourceDSCusername	{Source DSC User Name}	User name of Source DSC
-TargetDSCname	{Target DSC Name}	Name of the Target DSC
-TargetDSCusername	{Target DSC User Name}	User Name of Target DSC
-JobName	{Name of the Job}	Jobnames File Path to export and import
-Path	{Path}	Job Metadata files location

Optional Arguments

Option	Value	Description
Type TARGET/JOB/SAVESET	{Job Metadata Type}	Type of Job Metadata
config_target_group_map YES/NO	{Yes or No}	Virtual Config Target group Map in TargetDSC

Note:

Ensure "-path" location/directory is present in both Source and Target DSC.

Sample Result

Job Metadata Summary Report

```
JOBNAME             JOB_METADATA    STATUS

------------------  --------------  --------

BKP_20210610021939  EXPORT          PASS

BKP_20210610021939  IMPORT          PASS

BKP_20210609003054  EXPORT          PASS

BKP_20210609003054  IMPORT          PASS
```

Export/Import Pass/Fail Count Summary Report

```
Export Jobs Passed :  2

Export Jobs Failed :  0

Import Jobs Passed :  2

Import Jobs Failed :  0
```

Overall Job Metadata Summary Report

```
No. Jobs Passed  :  2
No.Jobs Failed :  0
```

Virtual Target Group Map Configuration Summary Report

```
Target Group Name    STATUS

-------------------  --------

Target_DFS_1           PASS
Target_DFS_2           PASS

Total Virtual Configuration Target Group Map Passed Count :  2
Total Virtual Configuration Target Group Map Failed Count :  0
```

Configuring dsc.properties to Delete Data Mover Job Metadata

Modify `$DSA_CONFIG_DIR/dsc.properties` to periodically remove the entry in the DSC repository for Data Mover related job metadata.

- Run a cron job using these definitions to schedule the autodelete Data Mover job.

Position	Definition
1	Seconds
2	Minutes
3	Hours
4	Day of Month
5	Month
6	Day of Week
7	Year (optional)

Leave blank to disable the autodelete Data Mover job.

```
autodeleteDMjob.cronstring=0 0 3 * * ?
```

This string sets the job to run every day at 3am.

Required Access Rights

Required Access Rights for Database Objects

Object Type	Required Access Rights		
	Backup	Restore	Copy
Aggregate Function	Dump	Restore, Alter Function	Restore, Create Table, Alter Function, Create View
JAR	Dump	Restore, Drop Procedure	Restore, Create Table, Drop Procedure
External Stored Procedures	Dump	Restore, Alter External Procedure	Restore, Create Table, Alter External Procedure
Standard Function	Dump	Restore, Alter Function	Restore, Create Table, Alter Function, Create View
Function Alias as of Analytics Database 16.20 Feature Update 1	Dump	Restore, Alter Function	Restore, Create Table, Alter Function, Create View
Trigger	Dump	Restore	N/A Triggers can't be copied
Join Index	Dump	Restore, Drop Table, Create Table,	Restore, Create Table, Drop Table
Macro	Dump	Restore	Restore, Create Macro
Hash Index	Dump	Restore, Drop Table, Create Table	Restore, Create Table, Drop Table
Stored Procedure	Dump	Restore	Restore, Create Table
Queue Table	Dump	Restore	Restore, Create Table
Table Function	Dump	Restore, Alter Function	Restore, Create Table, Alter Function
Table Function (Operator)	Dump	Restore, Alter Function	Restore, Create Table, Alter Function, Create View
Table	Dump	Restore, Drop Table, Create Table	Restore, Create Table
PPI Table	Dump	Restore, Drop Table, Create Table	Restore, Create Table, Drop Table
NOPI Table	Dump	Restore, Drop Table, Create Table	Restore, Create Table
View	Dump	Restore	Restore, Create View
Authorization	Dump	Restore	Restore, Create View
Parser Contract Function	Dump	Restore	Restore, Create Table

Object Type	Required Access Rights		
	Backup	Restore	Copy
System Join Index	Dump	Restore, Drop Table, Create Table	Restore, Drop Table, Create Table
SQL-H	Dump	Restore	Restore, Create View
User Installed File (UIF)	Dump	Restore, Create External Procedure, Execute Procedure on the SYSUIF database is required	Restore, Create Table, Create External Procedure, Execute Procedure on the SYSUIF database is required
GLOP Set	Dump	Restore	Restore, Create View

Rules for Restoring and Copying Objects

Selective Backup and Restore Rules

The following table lists individual object types, whether they can be restored or copied, and under what conditions. When performing a selective backup and restore on a table or database, observe the following rules.

- Methods and UDTs are stored in SYSUDTLIB; SYSUDTLIB is only backed up or restored as part of a DBC backup.
- Although an operation might be allowed, there is no guarantee the object will function as desired after the operation is complete.

Object Type	Restore to Same DBS [1]		Copy to Same DBS		Copy and Rename to Same DBS		Copy to Different DBS		Copy and Rename to Different DBS	
	DB Level	Object Level	DB Level	Object Level	DB Level	Object Level	DB Level	Object Level	DB Level	Object Level
Aggregate Function	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	No
JAR	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	No
External Stored Procedures	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	No
Standard Function	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	No
Function Alias New with Analytics Database 16.20 Feature Update 1	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	No
Trigger	Yes	Yes	No	No	No	No	No	No	No	No
Instant or Constructor Method	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
Join Index	Yes	Yes	Yes	Yes	No	No	Yes	Yes	No	No
Macro	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	No
Hash Index	Yes	Yes	Yes	Yes	No	No	Yes	Yes	No	No
Stored Procedure	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	No
Queue Table	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Table Function	Yes	Yes	Yes	Yes	Yes	No	Yes		Yes	No

Object Type	Restore to Same DBS [1]		Copy to Same DBS		Copy and Rename to Same DBS		Copy to Different DBS		Copy and Rename to Different DBS	
	DB Level	Object Level	DB Level	Object Level	DB Level	Object Level	DB Level	Object Level	DB Level	Object Level
Table Function (Operator)	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	No
Table	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
PPI Table	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
NOPI Table	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
User-Defined Data Type (UDT)	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
View	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	No
Authorization	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	No
Parser Contract Function	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	No
System Join Index [2]	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
SQL-H [3,4]	Yes	Yes	Yes	Yes	No	No	Yes	Yes [4]	No	No
User Installed File (UIF)	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	No

[1] DBS indicates an Analytics Database / Teradata Database system.

[2] System Join Indexes (SJIs) can be copied to a different database and the database name can be changed but the object name cannot be changed.

[3] All dependent objects used by the SQL-H object must exist prior to the copy.

[4] To restore a backup job that contains a SQL-H object to a different database, the backup job source database must be the TD_SERVER_DB database.

Column Legend

Restore to Same DBS

Restore back to the same database name, same object name, and same Teradata system.

Copy to Same DBS

Copy back to the same database name, same object name, and same Teradata system.

Copy and Rename to Same DBS

Map to the same Teradata system and change either the database name, object name, or both.

Copy to Different DBS

Copy back to the same database name and same object name in a different Teradata system.

Copy and Rename to Different DBS

Map to a different Teradata system and change either the database name, object name, or both.

Values for XML Elements

Many DSA commands read an XML file that specifies setup information or job details by specifying values for XML elements.

The following table indicates accepted values, case sensitivity, and character number restrictions for XML element values. These are in accordance with typical XML standards for Analytics Database / Teradata Database.

XML Element	Requirements
all_backup_objects	Accepted values: true, false. `all_backup_objects` must be false when processing individual object restore or copy. If `all_backup_objects` is false and an objectlist is not provided, the job cannot be created.
auto_retire	Accepted values: true, false. Must be entered in lowercase letters.
bar_media_server	Maximum of 32 characters
barnc_logging_level	Accepted values: Error, Info, Debug, Warning. Must be entered as shown: first character uppercase, others lowercase.
blob_container	Maximum of 63 characters. Must be entered in lowercase letters.
block_level_compression	Accepted values: DEFAULT, ON, OFF. Must be entered in uppercase letters.
config_map_name	Optional, maximum of 128 characters.
concurrent_bld_per_tbl	Value between 1-5. Default: 5 Teradata Database 16.0 and later
database_query_method	Accepted values: BASE_VIEW, EXTENDED_VIEW. Must be entered in uppercase letters.
data_phase	Accepted values: DATA, DICTIONARY. Must be entered in uppercase letters.
disable_fallback	Accepted values: false, true. Must be entered in lowercase letters. [Teradata Database 16.20 Update 1 and later] Always true. Changes are ignored. [Prior versions Teradata Database] Default: false. Can be set to true only if run_as_copy is also true.
dsc_logging_level	Accepted values: Error, Info, Debug, Warning. Must be entered as shown: first character uppercase, others lowercase.
dsc_name	Maximum of 128 characters: alphanumeric, "-" and "." First character must be alphanumeric (a-z, A-Z, and 0-9) only.
dsmain_logging_level	Accepted values: Error, Info, Debug, Warning. Must be entered as shown: first character uppercase, others lowercase.

XML Element	Requirements
enable_ temperature_override	Accepted values: true, false. Must be entered in lowercase letters.
fabric_name	Maximum of 64 characters: alphanumeric and "-" and "_". The first character must be alphanumeric (a-z, A-Z, and 0-9) only.
frequency_value	Value must be between 1-999. Defaults to 7.
ip_address	Maximum of characters 50, alphanumeric plus "." and ":" Do not use link-local IPv6 addresses (beginning with fe80) for media servers.
is_delete_after	Accepted values: Y/N. Must be entered as an uppercase letter.
is_enabled	Accepted values: true, false. Must be entered in lowercase letters. Default: false
is_encrypted	Accepted values: Y/N. Must be entered as an uppercase letter.
job_description	Maximum of 256 characters
job_name	Maximum of 128 characters
job_type	Accepted values: BACKUP, RESTORE, ANALYZE_VALIDATE, ANALYZE_ READ. Must be entered in uppercase letters.
map_to	Maximum of 128 characters
media_server	Maximum of 32 characters: alphanumeric and "-", "_", and "." First character must be alphanumeric (a-z, A-Z, and 0-9) only.
media_server_name	Maximum of 32 characters: alphanumeric and "-", "_", and "." First character must be alphanumeric (a-z, A-Z, and 0-9) only.
netmask	Maximum of 64 characters, alphanumeric plus "." and ":"
node_name	Maximum of 32 characters, alphanumeric and "-", "_", and "." First character must be alphanumeric (a-z, A-Z, and 0-9) only.
nosync	Accepted values: true, false. Must be entered in lowercase letters.
object_name	Maximum of 128 characters
object_type	Must be entered in uppercase letters. Accepted values are: AGGREGATE_FUNCTION, AUTHORIZATION, COMBINED_AGGREGATE_ FUNCTIONS, CONTRACT_FUNCTION, DATABASE, EXTERNAL_ PROCEDURE, FUNCTION_ALIAS, GLOP_SET, HASH_INDEX, INSTANCE_ OR_CONSTRUCTOR_METHOD, JAR, JOIN_INDEX, MACRO, NO_PI_ TABLE, ORDERED_ANALYTICAL_FUNCTION, QUEUE_TABLE, SERVER_ OBJECT, STANDARD_FUNCTION, STATISTICAL_FUNCTION, STORED_ PROCEDURE, TABLE, TABLE_FUNCTION, TABLE_OPERATOR, TRIGGER, USER, USER_DEFINED_METHOD, USER_DEFINED_DATA_ TYPE, USER_INSTALLED_FILE, VIEW
online	Accepted values: true, false. Must be entered in lowercase letters.

XML Element	Requirements
parent_name	Maximum of 128 characters
parent_type	Accepted values: DATABASE, USER. Must be entered in uppercase letters.
policy_class_name	Maximum of 128 characters, alphanumeric and "-" and "_"
port	Maximum of 5 digits. Must be between 1 and 65535
prefix_name	Maximum of 256 characters
query_band	Maximum of 2,048 characters
rename_to	Maximum of 128 characters
reset_node_limit	Accepted values: true, false. Must be entered in lowercase letters.
retire_units	Accepted values: DAYS, WEEKS. Must be entered in uppercase letters.
role_name	AWS only. IAM Role. Maximum of 64 characters. Alphanumeric only.
run_as_copy	Accepted values: true, false. Must be entered in lowercase letters.
skip_archive	Accepted values: true, false. Must be entered in lowercase letters.
skip_force_full	Accepted values: true, false. Must be entered in lowercase letters.
skip_joinhash_index	Optional. Accepted values: true, false. Default is false.
skip_stats	Accepted values: true, false. Default is false.
source_media	Maximum of 32 characters
start_am_pm	Accepted values: AM, PM. Must be entered in uppercase letters.
start_time	Maximum of 5 characters. Values must be between 1:00 and 12:00 (AM and PM are entered in the start_am_pm element)
storage_account	Maximum of 24 characters. Must be entered in lowercase letters.
storage_devices	Maximum of 3 characters, numeric range between 1-999
storage_type	Accepted values: cool, hot. Must be entered in lowercase letters.
streams_softlimit	Optional, range is 1- *max* soft limit of the stream soft limit, where *max* is the number of AMPs per node in the system. Optimize for the following: • Best backup throughput or storage efficiency: use 1- # of AMPs per node ◦ Best deduplication ◦ Take NetBackup startup time and Data Domain limits into account • Backup performance with unreasonable backup job start up time due to the NBU 1 stream per second limit: target 100 streams per Data Domain unit. This is usually enough to hit the Data Domain ingest limit for multiplexed deduplication ratio • Restore to the same configuration: Use the best backup value

XML Element	Requirements
	• Restore to a different configuration: Never let the total stream count (limit * nodes) exceed the AMP count of the smallest system. Rough guidelines for deduplication impact. • If global reduction ratio is expected in 8x or higher range, reduce the expected ratio by 50% • If global reduction ratio is expected to be 3-8x, reduce the expected ratio by 25% • If global reduction ratio is expected to be lower than 3x, reduce the expected ratio by 10%
system_name	Maximum of 32 characters, alphanumeric and "-", "_", and "." The first character of the name can be alphanumeric (a-z, A-Z, and 0-9) only.
target_entity	Maximum of 32 characters
target_group_name	Maximum of 32 characters, alphanumeric and "-", "_", and "." The first character of the name can be alphanumeric (a-z, A-Z, and 0-9) only.
target_media	Maximum of 32 characters
table_name	Maximum of 63 characters
target_accountid	Maximum of 30 characters
target_name	Maximum of 32 characters
target_tdpid	Maximum of 32 characters
temperature_override	Accepted values: DEFAULT, HOT, WARM, COLD. Must be entered in uppercase letters.
third_party_media_type	Accepted values: net_backup, dd_boost. Must be entered in lowercase letters.
third_party_server_name	Maximum of 32 characters, alphanumeric plus "-" and "_"
threshold_units	Accepted values: MB, GB
track_empty_tables	Accepted values: true, false. Default is false.

Teradata DSA Commands

DSA Commands

There are two major types of DSA commands:

- Configuration commands for configuring systems, servers, target groups, and the DSC repository
- Management and reporting commands for creating, executing, updating, and monitoring jobs

Use a DSA command by typing it in the command line, and then typing in any necessary parameters. There are complex mappings between BAR components, so the configuration commands must read attributes from an XML file. Configuration commands must specify the XML file name and location. Some of the management and reporting commands also accept XML files in which you can specify additional attributes that cannot be entered as parameters in the command line.

The reference topic for each DSA command provides the command's purpose, syntax, an example of what you might enter, a list of necessary and optional parameters, and a sample XML file, if the command requires one. Note the following conventions for using the command line:

- In the syntax description, the word in italics following a parameter is a placeholder that indicates that you must enter a value for the parameter. For example, where the syntax says *Type*, you might enter `backup` to specify a backup job for the job type parameter in the command line.
- Unless a parameter is listed as [Optional], you must specify a value for it. For commands that support XML files, you may specify values for parameters in the XML rather than directly in the command line. If you specify different parameter values in both the command line and in the XML, the values in the command line are used.
- Every parameter can be typed as a word or as a single character. A vertical bar indicates that you can use either format. For example, n|name means that you can type `name` or `n` to enter the **name** parameter.
- Always type a hyphen before each parameter that you enter. For example, after the command name, type `-file` or `-f` to specify the file parameter.
- The value for the file parameter must specify the filename and location of the input or output XML file.
- The Example section shows literal examples of what you might enter at the command line prompt. For example: `dsc.sh config_systems -f configSystem.xml`
- The XML File Example sections show samples of XML input files that certain commands require. Some commands export information as output to an XML file. Output file examples are not shown in this section.

abort_job

Purpose

The `abort_job` command aborts an actively running job, a job in the queue, or a job that is not responding.

Syntax

abort_job -n|-name *Name* [parameters]

Examples

```
dsc abort_job -n job1 -S
```

```
dsc abort_job -B -S
```

Parameters

n|name *Name*

The name of the job on which to perform the action. Must be unique for each job.

O|override

[Optional] Can only be used for backup, restore, and analyze_validate jobs. A Teradata job must be in the *aborting* phase.

Note:

- Overriding the abort_job only changes the status in the repository and does not affect the database.
- Only use the override parameter when a job hangs, so that DSC resources can be freed to run other jobs.

S|skip_prompt *SkipPrompt*

[Optional] Skips displaying a confirmation message before performing the command action.

B|backup_repository

[Optional] Aborts any running DSC repository job. The name of the job is not required when this parameter is used.

u|user_authentication *User*

Required when security management is enabled. Supplies the command with the Viewpoint user, and triggers a password prompt for authentication.

Permissions required: Job Permissions: Run, Viewpoint Role: Administrator for override

Usage Notes

There are two phases to aborting a job: aborting and aborted. The *aborting* phase releases media devices to the pool. The *aborted* phase releases node stream limits.

If the `abort_job` command is given for a job in queue to run, the job is removed from the queue. If the job is running when the request is sent to abort the job, it can take several minutes for the job to completely stop.

XML File Example

This command does not support an XML file.

Related Information:

Job Settings Tab

activate_job

Purpose

The `activate_job` command activates a retired job, so that it is available.

Syntax

activate_job -n|-name *Name* [parameters]

Example

```
dsc activate_job -n job1 -S
```

Parameters

n|name *Name*

The name of the job on which to perform the action. Must be unique for each job.

Example:
`job1`

S|skip_prompt *SkipPrompt*

[Optional] Skips displaying a confirmation message before performing the command action.

u|user_authentication *User*

Required when security management is enabled. Supplies the command with the Viewpoint user, and triggers a password prompt for authentication.

Permissions required: Administrator

Usage Notes

The `activate_job` command sets the state of a retired job to *active*, which allows you to run the job.

You cannot activate a job that is already in the active state.

XML File Example

This command does not support an XML file.

backup_application_gcp

Purpose

The `backup_application_gcp` command configures GCP Storage based on a parameter file.

Syntax

backup_application_gcp -f|-file *filename*

Example

`backup_application_gcp -f` *file1.xml*

Parameters

f|file *filename*

The full path and name of the file containing the necessary configuration parameters.

u|user_authentication *User*

Required when security management is enabled. Supplies the command with the Viewpoint user, and triggers a password prompt for authentication.

Permissions required: Administrator role

Usage Notes

You need the following information for the XML file.

- account: Name of the GCP Service Account
- account_key: Private Key from the GCP Service Account See https://cloud.google.com/iam/docs/creating-managing-service-account-keys
- Configure Customer Managed Keys (CMEK) to the bucket at GCP. See https://cloud.google.com/storage/docs/encryption/using-customer-managed-keys

XML File Example

Here is a representative XML file. Note the following:

- See https://cloud.google.com/storage/docs/creating-buckets for GCP Storage bucket information.
- The prefix_name should be followed by "/" to be used as a folder.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<dscBackupApplicationGcp xmlns="http://schemas.teradata.com/v2012/DSC">
  <config_gcp_storage_list>
      <account>sa-name@project-id.iam.gserviceaccount.com</account>
      <!-- 'account_key' - Required, max length 4,096 -->
      <account_key>long_account_key</account_key>
      <encryption_key>customer_managed_key</encryption_key>
      <buckets>
          <!--
              'Bucket name' - Required,
              - only lowercase letters, numbers, dashes (-), underscores (_),
                and dots (.). Spaces are not allowed
              - start and end with a number or letter
              - contain 3-63 characters. Names containing dots can contain up
                to 222 characters, but each dot-separated component can be no
                longer than 63 characters
              - cannot be represented as an IP address in dotted-decimal
                notation (for example, 192.168.5.4)
              - cannot begin with the "google" prefix
              - cannot contain "google" or close misspellings, such as "g00gle"
          -->
          <bucket_name>bucket1</bucket_name>
          <!-- 'Location enumeration ' - Required, valid values: us, eu, asia -->
          <location>us</location>
          <!-- 'Storage class enumeration ' - Required, valid values: standard,
                nearline, coldline, archive -->
          <storage_class>standard</storage_class>
          <prefix_list>
              <!--
                  'Folder name' - Required
                  - may contain letters, digits, spaces, hyphens and underscores
                  - start and end with a letter or digit
                  - 30 characters or less
                  - distinct from all other folders that share its parent
              -->
              <prefix_name>folder1/</prefix_name>
              <storage_devices>2</storage_devices>
          </prefix_list>
      </buckets>

      <buckets>
          <bucket_name>bucket2</bucket_name>
          <location>us</location>
          <storage_class>standard</storage_class>
          <prefix_list>
              <prefix_name>folder2/</prefix_name>
              <storage_devices>10</storage_devices>
          </prefix_list>
      </buckets>
  </config_gcp_storage_list>
</dscBackupApplicationGcp>
```

build_restore_objects

Purpose

The `build_restore_objects` command enables read and write on all the objects in the restore job without running the restore and / or builds the secondary indexes of the object.

Syntax

build_restore_objects -n|name *Name* -b|build_si -a|allow_write -l||logging_level *Level* -t|time_out

Example

```
build_restore_objects -n job1 -b -a -l Error -t 2000
```

Parameters

n|name *Name*

The name of the job you want to either build secondary indexes for or make read-write.

b|build_si

Build secondary indexes of the objects. Required if -a is not specified.

a|allow_write

Enable read/write on the objects. Required if -b is not specified.

l|logging_level *level*

[Optional] Error, Info, Debug, Warning

Example:

```
Info
```

t|time_out

[Optional] Time out value in minutes. Ends the current request as Failed, allowing a new request to be submitted. Range is 1 to 10080 minutes. Default: 1440

Example:

```
2500
```

S|skip_prompt *SkipPrompt*

[Optional] Skips displaying a confirmation message before performing the command action.

u|user_authentication *User*

Required when security management is enabled. Supplies the command with the Viewpoint user, and triggers a password prompt for authentication.

Permissions required: Administrator role

XML File Example

This command does not support an XML file.

cleanup_ir_source_system

Purpose

You cannot run this command if the source system has Incremental Restore enabled. Previous jobs must have completed or failed; they cannot be in progress.

The `cleanup_ir_source_system` command clears deleted row logs in the source system. This cleans up the metadata needed for IR backup.

Syntax

cleanup_ir_source_system -s|-system *System* -d|dbaccount_id *account_id* -l|logging_level *Level* -t|time_out -S|skip_prompt

Example

```
cleanup_ir_source_system -s system1 -d jana234 -l Error -t -S
```

Parameters

s|system *System*

Name of the source system.

Example:
`system1`

d|dbaccount_id *database_account*

[Optional] Database account ID

Example:
`jana234`

l|logging_level *level*

[Optional] Error, Info, Debug, Warning

Example:
`Info`

t|time_out

[Optional] Time out value in minutes. Ends the current request as Failed, allowing a new request to be submitted. Default: 720

u|user_authentication *User*

Required when security management is enabled. Supplies the command with the Viewpoint user, and triggers a password prompt for authentication.

Permissions required: Administrator role

S|skip_prompt *SkipPrompt*
[Optional] Skips displaying a confirmation message before performing the command action.

XML File Example

This command does not support an XML file.

config_aws

Purpose

The `config_aws` command configures the Amazon S3 Server in the DSA repository based on parameter files.

Syntax

config_aws -f|-file *filename*

Example

`config_aws -f file1.xml`

Parameters

f|file *filename*
The full path and name of the file containing the necessary configuration parameters.

u|user_authentication *User*
Required when security management is enabled. Supplies the command with the Viewpoint user, and triggers a password prompt for authentication.

Permissions required: Administrator role

Usage Notes

role_name is required. This is the IAM role (see http://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles.html). When roles are used, all DSA components must be running in the AWS cloud and assigned to this role.

You will need to enter an AWS account ID (access id) and IAM user access key for the Amazon S3 application. See http://docs.aws.amazon.com/IAM/latest/UserGuide/id_credentials_access-keys.html#Using_CreateAccessKey.

```
dsc config_aws -f aws.xml
```

```
Data Stream Controller Command Line 17.00.00.00
Command parameters:
-file : aws.xml
Enter access id for account "acctName".
```

```
AKIAJYGWLNCJDDZ35AXQ
```

```
Enter access key for account "acctName".
```

```
*******
```

```
Connected to DSC version 17.00.00.00
Saving AWS Backup Application configuration...
```

XML File Example

Here is a representative XML file. Note the following:

- See http://docs.aws.amazon.com/AmazonS3/latest/UG/s3-ug.pdf for S3 usage information.
- The prefix_name should be followed by "/" to be used as a folder.

```
<?xml version="1.0" encoding="UTF-8"?>

             <!--****This command must be on one line.****-->
<dscConfigAmzS3 dscVersion="" xmlns="http://schemas.teradata.com/v2012/DSC"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://schemas.teradata.com/v2012/DSC DSC.xsd">

<config_aws_list>
 <!-- character must be alphanumeric, - Required, max characters 32 -->
 <account_name>acctName</account_name>

 <!-- character must be alphanueric, - Required, max characters 64 -->
 <role_name>roleName</role_name>

 <!-- must be region enum -->
 <buckets_by_region>
    <!-- valid regions: us-east-1, us-west-1, us-west-2, eu-west-1,eu-central-1,
      ap-southeast-1, ap-southeast-2,ap-northeast-1, ap-northeast-2, sa-east-1 -->
     <region>us-east-1</region>

       <buckets>
        <!-- character must be alphanumeric, - Required, max characters 512 -->
         <bucket_name>bucketName</bucket_name>

         <prefix_list>
            <!-- character must be alphanumeric, - Required, max characters 256 -->
             <prefix_name>prefix</prefix_name>

            <!-- Required, Integer, min value is 1, max value is 999 -->
          <storage_devices>1</storage_devices>
         </prefix_list>
       </buckets>
    </buckets_by_region>
```

```
    </config_aws_list>
</dscConfigAmzS3>
```

config_azure

Purpose

The `config_azure` command configures the Microsoft Azure Server in the DSA repository based on parameter files.

Syntax

config_azure -f|-file *FILE*

Example

`config_azure -f` *file1.xml*

Parameters

f|file *filename*

The full path and name of the file containing the necessary configuration parameters.

u|user_authentication *User*

Required when security management is enabled. Supplies the command with the Viewpoint user, and triggers a password prompt for authentication.

Permissions required: Administrator role

Usage Notes

You will need to enter an access_key for the Azure storage account. See https://docs.microsoft.com/azure/storage/storage-create-storage-account

XML File Example

Here is a representative XML file. Note that the prefix_name should be followed by "/" to be used as a folder.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<dscConfigAzureBlobStorage xmlns="http://schemas.teradata.com/v2012/DSC">
<config_azure_blob_storage>
<!-- 'Storage account' - Required, max length 24, lowercase -->
<storage_account>azurerbuda</storage_account>
 <!-- 'Storage account enumeration ' - Required, valid values: cool, hot -->
 <storage_type>cool</storage_type>
 <!-- Specialized endpoint for Azure Independent Clouds e.g. AzureUSGovernment,
  AzureChinaCloud and AzureGermanCloud - Optional -->
 <region></region>
 <blobs>
 <!--'Blob container name' - Required, max length 63, lowercase, at least one -->
 <blob_container>udaesblob01</blob_container>
```

```
   <prefix_list>
      <!-- 'Prefix name' - Required, max length 256, at least one -->
      <prefix_name>br186001-1</prefix_name>
      <storage_devices>2</storage_devices>
   </prefix_list>
 </blobs>
 <blobs>
  <blob_container>udaesblob02</blob_container>
     <prefix_list>
       <prefix_name>br186001-2</prefix_name>
       <storage_devices>2</storage_devices>
     </prefix_list>
  </blobs>
</config_azure_blob_storage>
</dscConfigAzureBlobStorage>
```

config_azure with Azure Active Directory

Purpose

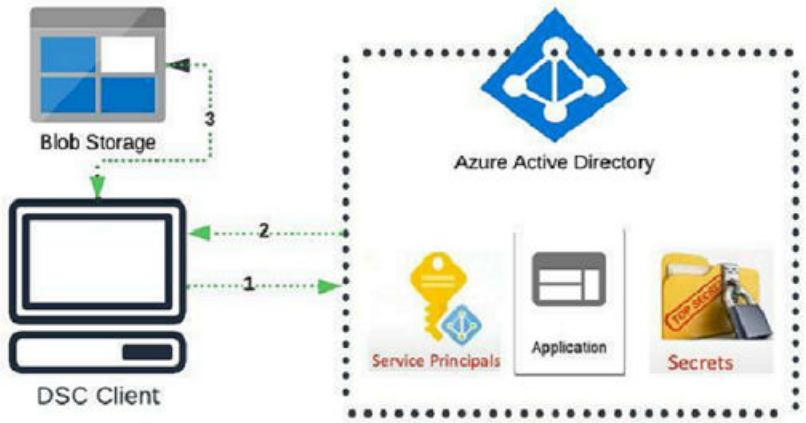
The DSA Azure AD helps prevent unauthorized access to the storage accounts. To prevent it, Teradata implemented Azure AD with DSA using the client's secret credentials approach based on the Authentication with service principles.

The `config_azure` command configures the Microsoft Azure Server in the DSA repository based on parameter files.

During configuration, enter the following parameters:

1. Tenant_id (Azure AD tenant)
2. Client_id (Registered app ID)
3. Client_secret (Client secret value)

Here is a representation of the high-level architecture:



Prerequisite

1. Register an app in the Azure portal to get the client ID, Azure AD tenant ID.
2. Create a client secret from the Azure portal to get the client's secret value.

Register app and assign role in the Azure portal

To register app and assign role, follow these steps:

1. In the Azure portal, click app registration from your AD tenant.
   a. Enter **app name**.
   b. Leave **redirect URL** blank.
   c. Note the **client ID** and **tenant ID**.
2. Create client secret for your registered app and note the client secret value. See https://docs.microsoft.com/en-us/azure/active-directory/develop/quickstart-register-app
3. To assign role for your storage account, see https://docs.microsoft.com/en-us/azure/role-based-access-control/role-assignments-portal?tabs=current

Syntax

config_azure -f|-file *FILE*

Example

```
config_azure -f file1.xml
```

Parameters

f|file *filename*

The full path and name of the file containing the necessary configuration parameters.

u|user_authentication *User*

Required when security management is enabled. Supplies the command with the Viewpoint user, and triggers a password prompt for authentication.

Permissions required: Administrator role

Usage Notes

Enter tenant ID, client ID, and client secret value for the Azure storage account.

XML File Example

Here's a representative XML file. Note that "/" must follow the prefix_name to use it as a folder.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<dscConfigAzureBlobStorage xmlns="http://schemas.teradata.com/v2012/DSC">
<config_azure_blob_storage>
<!-- 'Storage account' - Required, max length 24, lowercase -->
<storage_account>azurerbuda</storage_account>
 <!-- 'Storage account enumeration ' - Required, valid values: cool, hot -->
 <storage_type>cool</storage_type>
 <!-- Specialized endpoint for Azure Independent Clouds
e.g. AzureUSGovernment,
  AzureChinaCloud and AzureGermanCloud - Optional -->
 <region></region>
 <blobs>
 <!--'Blob container name' - Required, max length 63, lowercase, at least one -->
 <blob_container>udaesblob01</blob_container>
   <prefix_list>
      <!-- 'Prefix name' - Required, max length 256, at least one -->
      <prefix_name>br186001-1</prefix_name>
      <storage_devices>2</storage_devices>
   </prefix_list>
 </blobs>
 <blobs>
  <blob_container>udaesblob02</blob_container>
     <prefix_list>
       <prefix_name>br186001-2</prefix_name>
       <storage_devices>2</storage_devices>
     </prefix_list>
  </blobs>
<is_azure_ad>true</is_azure_ad>
</config_azure_blob_storage>
</dscConfigAzureBlobStorage>
```

config_dd_boost

Purpose

The `config_dd_boost` command configures the DD Boost server for the DSA repository.

Syntax

config_dd_boost -f|-file *FILE*

Example

```
dsc config_dd_boost -f configddboost.xml
```

Parameters

f|file *filename*

The full path and name of the file containing the necessary configuration parameters.

Example:
`configDDBoost.xml`

u|user_authentication *User*

Required when security management is enabled. Supplies the command with the Viewpoint user, and triggers a password prompt for authentication.

Permissions required: Administrator role

Usage Notes

When you use the `config_dd_boost` command, you will be prompted for the DD boost user name and password. For example:

```
dsc config_dd_boost -f ..\samples\sample_config_dd_boost.xml
Data Stream Controller Command Line 16.00.00.00
Command parameters:
-file : ..\samples\sample_config_dd_boost.xml
Enter DD Boost user name:
dd_user
Enter DD Boost user password:

Connected to DSC version 16.00.00.00
Saving dd_boost server...
Saving dd_boost server name: dddomain1
Saving dd_boost server with server id: 1
```

XML File Example

The following sample XML file contains the required parameters to configure a DD Boost server.

```
          <!--****This command must be on one line.****-->
<dscConfigDDBoost dscVersion="" xmlns="http://schemas.teradata.com/v2012/DSC"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://schemas.teradata.com/v2012/DSC DSC.xsd">

 <!-- 'third_party_server_name' Required, max characters 32, alphanumeric and
"-","_", "." first character must be alphanumeric -->
 <third_party_server_name>dddomain1</third_party_server_name>

 <!-- 'ip_address' - Required, max characters 50, alphanumeric and
"-","_",".",":" -->
 <ip_address>129.1.1.1</ip_address>
```

```
 <!-- 'third_party_media_type' - Required, accepted values: dd_boost, net_backup --
>
 <third_party_media_type>dd_boost</third_party_media_type>

 <!-- storage_units' - Required (at least one) -->
 <storage_units>

    <!-- 'storage_unit_name' - Required, max characters 50, alphanumeric and
"-","_"   -->
    <storage_unit_name>storage_unit1</storage_unit_name>

    <!-- 'max_files' - Required, Integer, min value is 1 -->
    <max_files>4</max_files>
 </storage_units>

 <storage_units>
    <storage_unit_name>storage_unit2</storage_unit_name>
    <max_files>5</max_files>
 </storage_units>
</dscConfigDDBoost>
```

config_disk_file_system

Purpose

The `config_disk_file_system` command configures the disk file system in the DSA repository.

Syntax

config_disk_file_system -f|-file *FILE*

Example

```
dsc config_disk_file_system -f configDiskFileSystem.xml
```

Parameters

f|file *filename*

The full path and name of the file containing the necessary configuration parameters.

Example:
`configDiskFileSystem.xml`

u|user_authentication *User*

Required when security management is enabled. Supplies the command with the Viewpoint user, and triggers a password prompt for authentication.

Permissions required: Administrator role

Usage Notes

File system names must be unique, fully qualified path names that begin with a forward slash, for example, `/dev/mnt1`.

File system names cannot differ by case alone. For example, both `/dev/mnt1` and `/dev/Mnt1` cannot be configured.

XML File Example

The following sample XML file contains the required parameters to configure the disk file system. The file_system parameter should use an absolute path, for example, `/tmp/dsu_mnt`.

```
<?xml version="1.0" encoding="UTF-8"?>

            <!--****This command must be on one line.****-->
<dscConfigDiskFileSystem xmlns="http://schemas.teradata.com/v2012/DSC"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://schemas.teradata.com/v2012/DSC DSC.xsd">

  <file_system_list>
    <!-- 'file_system' - Required, max characters 4096 -->
    <file_system>/tmp/barms1-1</file_system>
   <!-- 'max_files' - Required, Linux max file descriptor, use max number of streams
for data domain system -->
    <max_files>200</max_files>
  </file_system_list>
  <file_system_list>
    <!-- 'file_system' - Required, max characters 4096, use absolute path  -->
    <file_system>/tmp/barms1-2</file_system>
        <!-- 'max_files' - Required, Linux max file descriptor, use max number of
streams for data domain system -->
    <max_files>6</max_files>
  </file_system_list>
  <file_system_list>
    <!-- 'file_system' - Required, max characters 4096, use absolute path  -->
    <file_system>/tmp/barms1-3</file_system>
        <!-- 'max_files' - Required, Linux max file descriptor, use max number of
streams for data domain system -->
    <max_files>6</max_files>
  </file_system_list>
  <file_system_list>
    <!-- 'file_system' - Required, max characters 4096, use absolute path  -->
    <file_system>/tmp/barms2-1</file_system>
        <!-- 'max_files' - Required, Linux max file descriptor, use max number of
streams for data domain system -->
    <max_files>200</max_files>
  </file_system_list>
  <file_system_list>
    <!-- 'file_system' - Required, max characters 4096, use absolute path  -->
    <file_system>/tmp/barms2-2</file_system>
        <!-- 'max_files' - Required, Linux max file descriptor, use max number of
streams for data domain system -->
    <max_files>6</max_files>
  </file_system_list>
  <file_system_list>
    <!-- 'file_system' - Required, max characters 4096, use absolute path  -->
    <file_system>/tmp/barms3-1</file_system>
        <!-- 'max_files' - Required, Linux max file descriptor, use max number of
streams for data domain system -->
```

```
    <max_files>4</max_files>
  </file_system_list>
</dscConfigDiskFileSystem>
```

config_fabrics

Purpose

Beginning with DSA 16.10.01, the `config_fabrics` command defines the routing data path between TPA nodes and media servers using a parameter file. For best performance when using network fabric, make sure `dsc.properties` has `validate.fabric=on`.

Network fabric is not supported for repository backup and restore.

Syntax

config_fabrics -n|-name -f|-file *FILE*

Example

`config_fabrics -f` *file1.xml*

Parameters

n|name *fabric_name*

[Optional] Name of the fabric configuration.

f|file *filename*

The full path and name of the file containing the necessary configuration parameters.

u|user_authentication *User*

Required when security management is enabled. Supplies the command with the Viewpoint user, and triggers a password prompt for authentication.

Permissions required: Administrator role

Usage Notes

You can define multiple fabrics but each must be unique. For example, System A can have two fabrics for two target systems B and C, fabricAB and fabricAC. The information is saved into the repository and can be edited if new media servers or nodes are added.

- All media servers in a target group should be available in a single fabric.
- The correct fabric is chosen based on the media servers in the target group.
- Update `dsc.properties` for the number of fabrics allowed per Teradata system: `maxlimit.fabric=`*number-fabrics*. Default is 64.

The target group must define at least one media server that belongs to the fabric.

If a fabric is defined, job submission skips the logical net mask and selects the existing fabric when defining job data paths.

XML File Example

Here is a representative XML file. To use the prefix name as a folder, follow it with "/".

```
<?xml version="1.0" encoding="UTF-8"?>

          <!--****This command must be on one line.****-->
<dscConfigFabrics xmlns="http://schemas.teradata.com/v2012/DSC"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="http://schemas.teradata.com/v2012/DSC DSC.xsd ">

<!-- 'fabric_name' Required, max characters 32, alphanumeric and "-","_", first
character must be alphanumeric -->
<fabric_name>fabric1</fabric_name>

<!-- 'system_name' Required, max characters 32, alphanumeric and "-","_", first
character must be alphanumeric -->
<system_name>sdlc1955MMP</system_name>

<!-- 'is_enabled' Optional, accepted values: true/false. Default: false -->
<is_enabled>true</is_enabled>

<tpa_to_ms>
   <!-- Name of TPA node -->
   <node_name>sdlc1955001-01</node_name>
   <!-- 'ms_info' Required, one or more -->
   <ms_info>

     <!-- Required, max characters 32, alphanumeric plus "-","_", ".", first
character can only be alphanumeric -->
     <media_server>sdld0351</media_server>

    <!-- list of 'ip_address' - Required (min: one) , Only supporting IPV4 and IPV6
standards. No DNS support. Do not use link-local IPv6 addresses (beginning with
fe80) for media servers. -->
     <ip_address>10.25.34.237</ip_address>
   </ms_info>
</tpa_to_ms>
<tpa_to_ms>
   <node_name>sdlc1955001-02</node_name>
     <ms_info>
       <media_server>sdlc1955</media_server>
       <ip_address>153.64.24.236</ip_address>
     </ms_info>
</tpa_to_ms>
</dscConfigFabrics>
```

config_general

Purpose

The `config_general` command configures the general settings, based on the information contained in the parameters XML file.

Syntax

config_general -f|-file *File*

Example

```
dsc config_general -f configGeneral.xml
```

Parameters

f|file *filename*

The full path and name of the file containing the necessary configuration parameters.

Example:
`configGeneral.xml`

u|user_authentication *User*

Required when security management is enabled. Supplies the command with the Viewpoint user, and triggers a password prompt for authentication.

Permissions required: Administrator role

Usage Notes

Use the `config_general` command to change the default general configuration values.

XML File Example

The following sample XML file specifies general configuration settings.

```
<?xml version="1.0" encoding="utf-8" ?>
<dscConfigGeneral dscVersion="dscVersion1" xmlns="http://schemas.teradata.com/
v2012/DSC">
<!--  'warning_threshold' - Required, integer value. -->
<warning_threshold>10</warning_threshold>
<!--  'threshold_units' - Required, accepted values: MB/GB -->
<threshold_units>MB</threshold_units>
<!--  'is_encrypted' - Required, accepted values: Y/N -->
<is_encrypted>Y</is_encrypted>
<!--  'dsc_logging_level' -  Required, accepted values: Error,Info,Debug,Warning --
>
<dsc_logging_level>Error</dsc_logging_level>
<!--  'barnc_logging_level' - Required, accepted values:
Error,Info,Debug,Warning -->
<barnc_logging_level>Error</barnc_logging_level>
<!--  'delete_job_time' - Required, integer value. -->
<delete_job_time>40</delete_job_time>
<!--  'is_delete_after' - Required, accepted values: Y/N -->
<is_delete_after>Y</is_delete_after>
</dscConfigGeneral>
```

config_media_servers

Purpose

The `config_media_servers` command configures the BAR media servers.

Syntax

config_media_servers -f|-file *filename*

Example

```
dsc config_media_servers -f configMediaServers.xml
```

Parameters

f|file *filename*

The full path and name of the file containing the necessary configuration parameters.

Example:
`configMediaServers.xml`

u|user_authentication *User*

Required when security management is enabled. Supplies the command with the Viewpoint user, and triggers a password prompt for authentication.

Permissions required: Administrator role

Usage Notes

You must issue this command before creating a target group.

The media_server_name can only use the following characters: 'A-Z', 'a-z', '0-9', '-', '_', and '.'

XML File Example

The following sample XML file specifies parameters to configure BAR media servers.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>

          <!--****This command must be on one line.****-->
<dscConfigMediaServers dscVersion="dscVersion1" xmlns="http://schemas.teradata.com/
v2012/DSC"
xmlns:xsi="http://www.w3.org/2001/ XMLSchema-instance" xsi:schemaLocation="DSC.xsd">

<!-- 'media_server_name' - Required, max characters 32, alphanumeric plus "-","_",
".", first character must be alphanumeric -->
<media_server_name>media_server71</media_server_name>
<!-- Required, max characters 5, Value range 1-65535 -->

<ip_info>
<!-- 'ip_address' - Required (at least 1), Only supporting IPV4 and IPV6 standards
(do not use link-local IPv6 address [beginning with fe80]). No DNS support. -->
<ip_address>111.111.111.111</ip_address>

<!-- 'netmask' - Required (at least 1), Only supporting IPV4 and IPV6 standards (do
not use link-local IPv6 address [beginning with fe80]). No DNS support. -->
<netmask>255.255.255.0</netmask>
</ip_info>

<ip_info>
<!-- 'ip_address' - Required (at least 1), Only supporting IPV4 and IPV6 standards
(do not use link-local IPv6 address [beginning with fe80]). No DNS support. -->
<ip_address>222.222.222.222</ip_address>

<!-- 'netmask' - Required (at least 1), Only supporting IPV4 and IPV6 standards (do
not use link-local IPv6 address [beginning with fe80]). No DNS support. -->
<netmask>255.255.255.0</netmask>
</ip_info>
</dscConfigMediaServers>
```

config_nbu

Purpose

The `config_nbu` command configures a DSA system to use Veritas NetBackup third-party software to back up and restore data.

Syntax

config_nbu -f|-file *File*

Example

```
dsc config_nbu -f configNbu.xml
```

Parameters

f|file *filename*

The full path and name of the file containing the necessary configuration parameters.

Example:

`nbu_config.xml`

u|user_authentication *User*

Required when security management is enabled. Supplies the command with the Viewpoint user, and triggers a password prompt for authentication.

Permissions required: Administrator role

Usage Notes

The policies you define are case-sensitive and must match the policies defined in NetBackup.

The component name can only use the following characters: 'A-Z', 'a-z', '0-9' and '_'.

XML File Example

The following sample XML file specifies parameters to configure NetBackup servers.

```
<?xml version="1.0" encoding="UTF-8" ?>

          <!--****This command must be on one line.****-->
<dscConfigNbu xmlns="http://schemas.teradata.com/v2012/DSC"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://schemas.teradata.com/v2012/DSC DSC.xsd">

 <!-- 'third_party_server_name' Required, max characters 32, alphanumeric
and "-","_",
       first character must be alphanumeric -->
 <third_party_server_name>remoteNbu1</third_party_server_name>

 <!-- 'ip_address' - Required, max characters 50, alphanumeric and
"-","_",".",":" -->
 <ip_address>99.23.23.555</ip_address>

 <!-- 'third_party_media_type' - Required, accepted value: net_backup -->
 <third_party_media_type>net_backup</third_party_media_type>

 <!-- 'policy_class' - Optional -->
 <policy_class>

 <!--  'policy_class_name' - Required, max characters 128, alphanumeric and
"-","_" -->
 <policy_class_name>policy711</policy_class_name>

 <!--  'storage_devices' - Required, max characters 3, numeric range between
1-999 -->
 <storage_devices>200</storage_devices>
 </policy_class>
 <policy_class>
  <policy_class_name>policy712</policy_class_name>
  <storage_devices>100</storage_devices>
 </policy_class>
 <policy_class>
  <policy_class_name>policy713</policy_class_name>
  <storage_devices>100</storage_devices>
 </policy_class>
 <policy_class>
  <policy_class_name>policy721</policy_class_name>
  <storage_devices>100</storage_devices>
 </policy_class>
 <policy_class>
```

```
  <policy_class_name>policy722</policy_class_name>
  <storage_devices>100</storage_devices>
 </policy_class>

<!-- 'media_server_name' - Required, used to communicate with NBU server
to request
      information about the catalog -->
<media_server_name>bets_media_server</media_server_name>
</dscConfigNbu>
```

config_repository_backup

Purpose

The `config_repository_backup` command provides the configuration information to back up the DSC repository.

Syntax

config_repository_backup -f|-file *File*

Example

```
dsc config_repository_backup -f configRepositoryBackup.xml
```

Parameters

f|file *filename*

The full path and name of the file containing the necessary configuration parameters.

Example:
`configRepositoryBackup.xml`

u|user_authentication *User*

Required when security management is enabled. Supplies the command with the Viewpoint user, and triggers a password prompt for authentication.

Permissions required: Administrator role

Usage Notes

The DSC repository contains all BAR information, including configuration and job definitions. It should be backed up regularly to enable system recovery.

XML File Example

Here is a sample XML file containing configuration and scheduling information for a repository backup.

```
<dscRepositoryBackup dscVersion="dscVersion1" xmlns="http://schemas.teradata.com/
v2012/DSC">
```

```
    <!-- 'target_name' - Required, max 32 characters -->
    <target_name>SampleBackupRepoTargetGroup</target_name>

    <!-- 'frequency_value' - Required, Value between 1-4, Defaults to 1 -->
    <frequency_value>1</frequency_value>

    <!-- 'day_selection' - Required, accepted values: Su, Mo, Tu, We, Th, Fr, Sa -->
    <day_selection>Sa,Su</day_selection>

    <!-- 'start_time' - Required, Max characters 5, Values 1:00-12:00 -->
    <start_time>12:00</start_time>

    <!-- 'start_am_pm' - Required, accepted values: AM/PM -->
    <start_am_pm>AM</start_am_pm>

</dscRepositoryBackup>
```

config_systems

Purpose

The `config_systems` command configures the DSC settings for the Teradata system used for backup and restore jobs. It also sets the selector in the targeted system for ActiveMQ and establishes a JDBC connection to the target system. Nodes are configured automatically after restarting DSMain on the system.

Syntax

config_systems -f|-file *File* -s|skip_system_config *SkipSystemConfiguration*

Example

```
dsc config_systems -f configSystem.xml -s system
```

Parameters

f|file *filename*

The full path and name of the file containing the necessary configuration parameters.

Example:
`configSystem.xml`

s|skip_system_config *SkipSystemConfiguration*

[Optional] Flag indicating whether the systems and node configuration or selector setting is skipped when you run the command. If this option is not specified, then both parts of the configuration (system and selector) are run. Enter one of the following:

- `system`: Avoid configuring Teradata systems and nodes
- `selector`: Avoid setting the selector in the targeted system for ActiveMQ

Note:

If you choose the selector option, restart the DSMAIN on the database.

u|user_authentication *User*

Required when security management is enabled. Supplies the command with the Viewpoint user, and triggers a password prompt for authentication.

Permissions required: Administrator role

Usage Notes

System nodes are configured by auto discovery.

Item	Description
Component names	The component name can only use the following characters: 'A-Z', 'a-z', '0-9' and '_'.
	The first character of the component name cannot be '_'.
Allowing Backing Jobs Based on Full or Cumulative Backup Jobs Completed with Errors	By setting the `skip_force_full` option in the XML file, you can run a CBB backup job that is based on a prior full or cumulative backup job that completed with errors. **false** Default. If the full or cumulative backup job fails, aborts, or completes with errors, the next time the job is run, the job is forced to be a full backup. **true** If the full or cumulative backup job fails or aborts, the next time the job is run, the job is forced to be a full backup. Jobs that complete with errors are handled the same as jobs that complete successfully or with a warning.
	The `skip_force_full` option is set system-wide. You cannot use this option on a single-job basis.
System Configuration Attributes	The following XML file example shows system configuration attributes. System configuration is subject to the following restrictions and requirements.
	Setting `reset_node_limit` to `true` overwrites node limits with the system's hard and soft limits.
	Soft and hard limits for insertion of new nodes: • If node limits are specified, they are used. • If node limits are not specified, the system limits are used. • The soft limits cannot be greater than the hard limits.
	Soft and hard limits when updating existing nodes: • If the node limit is specified, the node limit is used.

Item	Description
	• If a node limit is not specified and the system limit = system limit in database, the node limit is used. • If a node limit is not specified and system limit is not the same as system limit in database, system limit is used.

XML File Example

Here is a representative XML file containing system configuration information.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

          <!--****This command must be on one line.****-->
<dscConfigSystems xmlns="http://schemas.teradata.com/v2012/DSC"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="DSC.xsd">

<system>
<!-- 'system_name' - Required, max 32 characters -->
<system_name>system7</system_name>

<!-- 'tdpid' - Required (unless skipped by option)-->
<tdpid>system7</tdpid>

<!-- 'database_query_method' - Required, accepted values: BASE_VIEW/
EXTENDED_VIEW -->
<database_query_method>BASE_VIEW</database_query_method>

<!-- 'streams_softlimit' - Required, number of streams per node per job -->
<streams_softlimit>20</streams_softlimit>

<!-- 'streams_hardlimit' - Required, max number of streams per node-->
<streams_hardlimit>20</streams_hardlimit>

<!-- 'reset_node_limit' - Optional, accepted values: true/false -->
<reset_node_limit>false</reset_node_limit>

<!-- 'skip_force_full' - Optional, accepted values: true/false -->
<skip_force_full>false</skip_force_full>
</system>
</dscConfigSystems>
```

config_target_groups

Purpose

The `config_target_groups` command configures the target groups based on the target type and the information from the parameters file.

- New `disable_gen_zero` flag for NetBackup and DDBoost. When true, backup jobs are rejected if any media server is offline.
- New `repository_replication` flag. When true, backup data is replicated from the system landing zone to third party storage.

Syntax

config_target_groups -t|-type *Target Type* -r|repository_replication -f|-file *File* -B|backup_repository -S|-skip_prompt

Example

```
dsc config_target_groups -t target_nbu -f TargetGroupParameters.xml -B -S
```

```
dsc config_target_groups -t remote_file_system -f test.xml
```

```
dsc config_target_groups -t TARGET_S3 -r -f replication_tg_S3.xml
```

Parameters

t|type *Type*

The type of the BAR component to add to the target group. Target group types are DUMMY, TARGET_NBU, REMOTE_FILE_SYSTEM, TARGET_DDBOOST, TARGET_S3, TARGET_AZURE, and TARGET_GCP.

Example:
`target_nbu`

f|file *filename*

The full path and name of the file containing the necessary configuration parameters.

Example:
`TargetGroupParameters.xml`

r|repository_replication

[Optional] Marks a target group for replication. Valid for target group types TARGET_NBU, REMOTE_FILE_SYSTEM, TARGET_DDBOOST, TARGET_S3, TARGET_AZURE, and TARGET_GCP.

S|skip_prompt *SkipPrompt*

[Optional] Skips displaying a confirmation message before performing the command action.

u|user_authentication *User*

Required when security management is enabled. Supplies the command with the Viewpoint user, and triggers a password prompt for authentication.

Permissions required: Administrator role

Usage Notes

When adding or removing a policy, storage unit, or file path from an existing target group, the following is true:

- You cannot remove a policy, storage unit, or file path from an existing target group if it has been used to generate a valid save set.
- When you remove a policy, storage unit, or file path from an existing target group, the DSC automatically disables the target group mapping if that policy, storage unit, or file path is used in either the source or target of target group mapping.
- When you add a new policy, storage unit, or file path to an existing target group, the DSC disables the target group mapping if the target group is defined as the source target group in the target group mapping.

When using a REMOTE_FILE_SYSTEM target group type, the following is true:

- If the path defined in <mounted_file_system> is the same among multiple media servers, the path implies that the media server is a NFS mount that all the media servers can access the data written under the same <mounted_file_system>.
- If one of the media servers is offline during a restore, analyze_read or query_backupids command, DSA will automatically replace the media server with a media server that can access the same mounted file system.

When using a TARGET_DDBOOST target group type, the following is true:

- If the values in <target_entity> and <storage_units> are the same among multiple media servers, the values imply that these media servers can access the data written on the same <target_entity> and <storage_units>.
- If one of the media servers is offline during a restore, analyze or query_backupids command, DSA will automatically replace the media server with a media server that can access the same target entity and storage units.

Before configuring a different type of target group, the media server must have the corresponding AXM module installed. Use list_access_module to verify which AXM modules are installed.

The component name can only use the following characters: 'A-Z', 'a-z', '0-9' and '_'.

XML File Example - Remote Media Target Group (NetBackup)

Here is a sample XML file to configure a remote media target group (TARGET_NBU).

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

                <!--****This command must be on one line.****-->
<dscConfigTargetGroupsRemoteMedia dscVersion="dscVersion1" xmlns="http://
schemas.teradata.com/v2012/DSC">

<!-- 'target_group_name' - Required,  max 32 characters, first character must be
alphanumeric only. -->
```

```
<target_group_name>SampleTargetGroup</target_group_name>

<!-- 'is_enabled' Optional, accepted values: true/false. Default : false -->
<is_enabled>true</is_enabled>

<!-- 'target_entity' - Required, max 30 characters -->
<target_entity>nbu52</target_entity>

<!-- 'targets' - Required (at least one) -->
<targets>

   <!-- 'bar_media_server' - Required, max 32 characters -->
   <bar_media_server>SampleMediaServer1</bar_media_server>

   <!-- 'policy_class' - Required (at least one) -->
   <policy_class>

     <!-- 'policy_class_name' - Required, max 128 characters. Policy must exist
on
     target_entity. -->
     <policy_class_name>policy75</policy_class_name>

     <!-- 'devices' - Required -->
     <devices>1</devices>

   </policy_class>
   <policy_class>
     <policy_class_name>policy76</policy_class_name>
     <devices>1</devices>
     </policy_class>
</targets>
<targets>
   <bar_media_server>SampleMediaServer2</bar_media_server>
   <policy_class>
     <policy_class_name>policy75</policy_class_name>
     <devices>1</devices>
   </policy_class>
   <policy_class>
     <policy_class_name>policy76</policy_class_name>
     <devices>1</devices>
   </policy_class>
</targets>

<!-- 'disable_gen_zero' Optional, accepted values: true/false. Default: false -->
```

```
<disable_gen_zero>true</disable_gen_zero>

</dscConfigTargetGroupsRemoteMedia>
```

XML File Example - Remote File System Target Group

Here is a sample XML file to configure a remote file system target group (REMOTE_FILE_SYSTEM).

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<dscConfigTargetGroupsRemoteFile xmlns="http://schemas.teradata.com/v2012/DSC">
  <target_group_name>t_remote_fs</target_group_name>
  <is_enabled>true</is_enabled>
  <mounted_file_system_remote>
  <bar_media_server>barms2</bar_media_server>
  <file_system>
  <mounted_file_system>/tmp/jing_test</mounted_file_system>
  <files>3</files>
  </file_system>
  </mounted_file_system_remote>
  </dscConfigTargetGroupsRemoteFile>
```

XML File Example - DDBoost Target Group

Here is a sample XML file to configure a DDBoost target group (TARGET_DDBOOST).

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<dscConfigTargetGroupsDDBoost dscVersion="dscVersion1" xmlns="http://
schemas.teradata.com/v2012/DSC">

<!-- 'target_group_name' - Required,  max 30 characters, first character must be
alphanumeric only. -->
<target_group_name>dd_boost_tg1</target_group_name>

<!-- 'is_enabled' Optional, accepted values: true/false. Default : false -->
<is_enabled>true</is_enabled>

<!-- 'targets' - Required (at least one) -->
<targets>
   <!-- 'target_entity' - Required, max 32 characters -->
   <target_entity>dddomain1</target_entity>

   <!-- 'bar_media_server' - Required, max 32 characters -->
   <media_storage_units>
     <!--media_storage_units' - Required (at least one) -->
     <bar_media_server>barvm12</bar_media_server>

     <!-- storage_units' - Required (at least one) -->
     <storage_units>
```

```
      <!-- 'storage_unit_name' - Required, max characters 50, alphanumeric and
      "-","_"   -->
      <storage_unit_name>storage_unit1</storage_unit_name>

      <!-- 'max_files' - Required, Integer, min value is 1-->
      <files>40</files>

    </storage_units>
  </media_storage_units>
</targets>

<!-- 'disable_gen_zero' Optional, accepted values: true/false. Default : false --
>
<disable_gen_zero>true</disable_gen_zero>

</dscConfigTargetGroupsDDBoost>
```

XML File Example - Amazon S3 Target Group

Here is a sample XML file to configure an S3 target group (TARGET_S3).

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<dscConfigTargetGroupsAmzS3 dscVersion="dscVersion1" xmlns="http://
schemas.teradata.com/v2012/DSC">

 <!-- 'target_group_name' - Required,  max 32 characters, first character must
 be alphanumeric only. -->
 <target_group_name>SampleTargetGroup</target_group_name>

 <!-- 'is_enabled' Optional, accepted values: true/false. Default : false -->
 <is_enabled>true</is_enabled>

 <!-- 'aws_account_name'- Required, max 32 characters -->
 <account_name>accountName</account_name>

 <!-- 'region' - Required, max 30 characters -->
 <region>us-east-1</region>

 <!-- 'targets' - Required (at least one) -->
 <targetMediaBuckets>
       <!-- 'bar_media_server' - Required, max 32 characters -->
       <bar_media_server>SampleMediaServer1</bar_media_server>

       <!-- 'buckets' - Required (at least one) -->
       <buckets
              <!-- 'bucket_name' - Required, max 512 characters. Bucket must exist
          on the region -->
              <bucket_name>bucket1</bucket_name>

              <!-- 'prefix_list' - Required (at least one) -->
              <prefix_list>
                     <!-- 'prefix_name' - Required, max 256 characters -->
                     <prefix_name>prefix1/prefix2/</prefix_name>
```

```
                        <!-- 'storage_devices' - Required -->
                        <storage_devices>1</storage_devices>
                </prefix_list>
        </buckets>
</targetMediaBuckets>
<targetMediaBuckets>
                <!-- 'bar_media_server' - Required, max 32 characters -->
                <bar_media_server>SampleMediaServer2</bar_media_server>

                <!-- 'buckets' - Required (at least one) -->
                <buckets>
                <!-- 'bucket_name' - Required, max 512 characters. Bucket must exist
           on the region -->
                <bucket_name>bucket2</bucket_name>

                <!-- 'prefix_list' - Required (at least one) -->
                <prefix_list>
                        <!-- 'prefix_name' - Required, max 256 characters -->
                        <prefix_name>prefix1/prefix2/</prefix_name>
                        <!-- 'storage_devices' - Required -->
                        <storage_devices>1</storage_devices>
                </prefix_list>
        </buckets>
</targetMediaBuckets>
</dscConfigTargetGroupsAmzS3>
```

XML File Example - Azure Blob Target Group

Here is a sample XML file to configure an Azure Blob target group (TARGET_AZURE).

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<dscConfigTargetGroupsAzureBlobStorage xmlns="http://schemas.teradata.com/
v2012/DSC">

  <target_group_name>azure_tg</target_group_name>
  <is_enabled>true</is_enabled>
  <targets>

  <!-- 'Storage account' - Required, max length 24, lower case -->
    <storage_account>azure_storage_account1</storage_account>

  <!-- 'Storage account enumeration ' - Required, valid values: cool, hot -->
    <storage_type>cool</storage_type>
  <targetMediaBlob>
    <!-- 'bar_media_server' - Required, max 32 characters -->
      <bar_media_server>mediaserver1</bar_media_server>

      <blobs>
        <!-- 'Blob container name' - Required, max length 63, lower case, at least
        one -->
          <blob_container>blob1</blob_container>
          <prefix_list>
          <!-- 'Prefix name' - Required, max length 256, at least one -->
            <prefix_name>prefix1/</prefix_name>

            <!-- 'storage_devices' - Required -->
              <storage_devices>100</storage_devices>
          </prefix_list>
       </blobs>
  </targetMediaBlob>

  <targetMediaBlob>
    <!-- 'bar_media_server' - Required, max 32 characters -->
```

```
      <bar_media_server>mediaserver2</bar_media_server>
      <blobs>
      <!-- 'Blob container name' - Required, max length 63, lower case, at least
      one -->
        <blob_container>blob1</blob_container>
        <prefix_list>
        <!-- 'Prefix name' - Required, max length 256, at least one -->
          <prefix_name>prefix1/</prefix_name>

          <!-- 'storage_devices' - Required -->
            <storage_devices>100</storage_devices>
        </prefix_list>
      </blobs>
    </targetMediaBlob>
  </targets>
  <targets>
    <!-- 'Storage account' - Required, max length 24, lower case -->
      <storage_account>azure_storage_account2</storage_account>
      <!-- 'Storage account enumeration' - Required, valid values: cool, hot -->
        <storage_type>cool</storage_type>
        <targetMediaBlob>
          <!-- 'bar_media_server' - Required, max 32 characters -->
            <bar_media_server>mediaserver2</bar_media_server>
     <blobs>
        <blob_container>blob2</blob_container>
          <prefix_list>
            <prefix_name>prefix1/</prefix_name>
              <storage_devices>100</storage_devices>
          </prefix_list>
      </blobs>
    </targetMediaBlob>
 </targets>
</dscConfigTargetGroupsAzureBlobStorage>
```

XML File Example - GCP Storage Target Group

Here is a sample XML file to configure an GCP Storage target group (TARGET_GCP).

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<dscConfigTargetGroupsGcpStorage xmlns="http://schemas.teradata.com/v2012/DSC">
   <!-- 'target_group_name' - Required, max 43 characters, can only consist of
   "a-z, A-Z, 0-9, _, -" characters and first character must be alphanumeric
   only. -->
   <target_group_name>gcp_tg</target_group_name>
   <is_enabled>true</is_enabled>
   <targets>
       <!-- 'account_name' - Required, max length 1024, lower case -->
       <account_name>accountName</account_name>
       <targetMediaBuckets>
           <!-- 'bar_media_server' - Required, max 32 characters -->
           <bar_media_server>mediaserver1</bar_media_server>
           <buckets>
              <!-- 'bucket_name' - Required
                 - only lowercase letters, numbers, dashes (-), underscores (_),
                 and dots (.). Spaces are not allowed
                 - start and end with a number or letter
                 - contain 3-63 characters. Names containing dots can contain up
                 to 222 characters, but each dot-separated component can be no
                 longer than 63 characters
                 - cannot be represented as an IP address in dotted-decimal
                 notation (for example, 192.168.5.4)
                 - cannot begin with the "goog" prefix
```

```
                - cannot contain "google" or close misspellings, such as "g00gle"
            -->
            <bucket_name>>bucketName1</bucket_name>
            <prefix_list>
                <!-- 'prefix_name' - Required
                   - may contain letters, digits, spaces, hyphens and
                underscores
                   - start and end with a letter or digit
                   - 30 characters or less
                   - distinct from all other folders that share its parent
                -->
                <prefix_name>folder1/</prefix_name>
                <!-- 'storage_devices' - Required -->
                <storage_devices>100</storage_devices>
            </prefix_list>
        </buckets>
    </targetMediaBuckets>
    <targetMediaBuckets>
        <!-- 'bar_media_server' - Required, max 32 characters -->
        <bar_media_server>mediaserver2</bar_media_server>
        <buckets>
            <!-- 'bucket_name' - Required
               - only lowercase letters, numbers, dashes (-), underscores (_),
               and dots (.). Spaces are not allowed
               - start and end with a number or letter
               - contain 3-63 characters. Names containing dots can contain up
               to 222 characters, but each dot-separated component can
               be no longer than 63 characters
               - cannot be represented as an IP address in dotted-decimal
               notation (for example, 192.168.5.4)
               - cannot begin with the "goog" prefix
               - cannot contain "google" or close misspellings, such as "g00gle"
            -->
            <bucket_name>>bucketName2</bucket_name>
            <prefix_list>
                <!-- 'prefix_name' - Required
                   - may contain letters, digits, spaces, hyphens and
                underscores
                   - start and end with a letter or digit
                   - 30 characters or less
                   - distinct from all other folders that share its parent
                -->
                <prefix_name>folder2/</prefix_name>
                <!-- 'storage_devices' - Required -->
                <storage_devices>100</storage_devices>
            </prefix_list>
        </buckets>
    </targetMediaBuckets>
  </targets>
</dscConfigTargetGroupsGcpStorage>
```

config_target_group_map

Purpose

The `config_target_group_map` command configures the map between target groups when restoring to a different client configuration.

Syntax

config_target_group_map -f|-file *Name*

Example

```
dsc config_target_group_map -f target_group_mapping.xml
```

Parameter

f|file *filename*

The full file path and file name of the file containing the target group mapping parameters

`target_group_mapping.xml`

u|user_authentication *User*

Required when security management is enabled. Supplies the command with the Viewpoint user, and triggers a password prompt for authentication.

Permissions required: Administrator role

V|virtual

[Optional] Indicates this is a mapping from a virtual target group to a physical target group.

Usage Notes

All backup (source) sub-targets must have a mapping in the XML file in order to succeed. Each backup (source) sub-target must be mapped to only one restore (destination).

Any changes to either target group using the `config_target_groups` command could disable the mappings attached to the specific target. If the new target group configuration adds or removes sub-targets, it will disable all mappings associated with the target group. To re-enable the mappings, reconfigure that mapping reflecting the target group changes using the `config_target_group_map` command.

XML File Examples

Here is a representative XML files containing target group mapping. Note the following:

- `master_source_target_name` is required.
- `master_dest_target_name` is required.
- In the list of `target_group_maps`, at least one is needed.
- `master_source_dsc_id` is optional if mapping between two physical target groups, but *required* if mapping between a virtual target and a physical target.

XML File Example - Remote Media Target Group Map (NetBackup)

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<dscConfigTargetGroupMap xmlns="http://schemas.teradata.com/v2012/DSC">
      <master_source_target_name>hawaii2_tg</master_source_target_name>
      <master_source_dsc_id>hawaii2</master_source_dsc_id>
      <master_dest_target_name>remote_nbu</master_dest_target_name>
      <target_group_maps>
            <source_mediaserver_name>hawaii2</source_mediaserver_name>
            <source_policy_class_name>policy_h2</source_policy_class_name>
            <dest_mediaserver_name>barms1.teradata.com</dest_mediaserver_name>
            <dest_policy_class_name>policy1</dest_policy_class_name>
      </target_group_maps>
      <target_group_maps>
            <source_mediaserver_name>hawaii2</source_mediaserver_name>
            <source_policy_class_name>policy_h2</source_policy_class_name>
            <dest_mediaserver_name>barms3</dest_mediaserver_name>
            <dest_policy_class_name>policy3</dest_policy_class_name>
        </target_group_maps>
</dscConfigTargetGroupMap>
```

XML File Example - Remote File System Target Group Map

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<dscConfigTargetGroupMap xmlns="http://schemas.teradata.com/v2012/DSC">
     <!--  This map to be used only for Remote File System target groups. -->
        <master_source_target_name>MasterSampSource</master_source_target_name>
        <master_source_dsc_id>MasterSrcDscIdSample</master_source_dsc_id>
        <master_dest_target_name>MasterSampDest</master_dest_target_name>
        <target_group_fs_maps>
                <source_mediaserver_name>SampSrcMS1</source_mediaserver_name>
                <source_path>/var/opt/teradata/datafiles/path1</source_path>
                <dest_mediaserver_name>SampDestMS1</dest_mediaserver_name>
                <dest_path>/var/opt/teradata/datafiles/path4</dest_path>
        </target_group_fs_maps>
        <target_group_fs_maps>
                <source_mediaserver_name>SampSrcMS1</source_mediaserver_name>
                <source_path>/var/opt/teradata/datafiles/path2</source_path>
                <dest_mediaserver_name>SampDestMS2</dest_mediaserver_name>
                <dest_path>/var/opt/teradata/datafiles/path3</dest_path>
        </target_group_fs_maps>
        <target_group_fs_maps>
                <source_mediaserver_name>SampSrcMS2</source_mediaserver_name>
```

```
                <source_path>/var/opt/teradata/datafiles/path3</source_path>
                <dest_mediaserver_name>SampDestMS2</dest_mediaserver_name>
                <dest_path>/var/opt/teradata/datafiles/path2</dest_path>
        </target_group_fs_maps>
        <target_group_fs_maps>
                <source_mediaserver_name>SampSrcMS3</source_mediaserver_name>
                <source_path>/var/opt/teradata/datafiles/path4</source_path>
                <dest_mediaserver_name>SampDestMS1</dest_mediaserver_name>
                <dest_path>/var/opt/teradata/datafiles/path1</dest_path>
        </target_group_fs_maps>
</dscConfigTargetGroupMap>
```

XML File Example - DDBoost Target Group Map

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<dscConfigTargetGroupMap xmlns="http://schemas.teradata.com/v2012/DSC">
     <!--  This map to be used only for DDBoost target groups. -->
        <master_source_target_name>MasterSampSource</master_source_target_name>
        <master_source_dsc_id>MasterSrcDscIdSample</master_source_dsc_id>
        <master_dest_target_name>MasterSampDest</master_dest_target_name>
        <target_group_ddboost_maps>
             <source_target_entity>SrcTE</source_target_entity>
                <source_mediaserver_name>SampSrcMS1</source_mediaserver_name>
                <source_storage_unit>src_storage_unit1</source_storage_unit>
                <dest_target_entity>DestTE</dest_target_entity>
                <dest_mediaserver_name>SampDestMS</dest_mediaserver_name>
                <dest_storage_unit>dest_storage_unit</dest_storage_unit>
        </target_group_ddboost_maps>
        <target_group_ddboost_maps>
          <source_target_entity>SrcTE</source_target_entity>
          <source_mediaserver_name>SampSrcMS1</source_mediaserver_name>
          <source_storage_unit>src_storage_unit2</source_storage_unit>
          <dest_target_entity>DestTE</dest_target_entity>
          <dest_mediaserver_name>SampDestMS</dest_mediaserver_name>
          <dest_storage_unit>dest_storage_unit</dest_storage_unit>
    </target_group_ddboost_maps>
        <target_group_ddboost_maps>
             <source_target_entity>SrcTE</source_target_entity>
                <source_mediaserver_name>SampSrcMS2</source_mediaserver_name>
                <source_storage_unit>src_storage_unit3</source_storage_unit>
                <dest_target_entity>DestTE</dest_target_entity>
                <dest_mediaserver_name>SampDestMS</dest_mediaserver_name>
                <dest_storage_unit>dest_storage_unit</dest_storage_unit>
```

```
        </target_group_ddboost_maps>
</dscConfigTargetGroupMap>
```

XML File Example - Amazon S3 Target Group Map

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<dscConfigTargetGroupMap xmlns="http://schemas.teradata.com/v2012/DSC">
    <!--  This map to be used only for AWS target groups. -->
        <master_source_target_name>tgaws1</master_source_target_name>
        <master_source_dsc_id>local-dsc-john</master_source_dsc_id>
        <master_dest_target_name>tgaws2</master_dest_target_name>
        <target_group_amzs3_maps>
                <source_mediaserver_name>sdld0280</source_mediaserver_name>
                <source_bucket_name>bucket01</source_bucket_name>
                <source_prefix_name>prefix01/</source_prefix_name>
                <dest_mediaserver_name>sdld0281</dest_mediaserver_name>
                <dest_bucket_name>bucket02</dest_bucket_name>
                <dest_prefix_name>prefix02/</dest_prefix_name>
        </target_group_amzs3_maps>
</dscConfigTargetGroupMap>
```

XML File Example - Azure Blob Target Group Map

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<dscConfigTargetGroupMap xmlns="http://schemas.teradata.com/v2012/DSC">
    <!--  This map to be used only for Azure target groups. -->
        <master_source_target_name>azure1_tg</master_source_target_name>
        <master_source_dsc_id>local-dsc-john</master_source_dsc_id>
        <master_dest_target_name>azure2_tg</master_dest_target_name>
        <target_group_azure_maps>
                <source_storage_account>storageacct1</source_storage_account>
                <source_blob_container>udaesblob01</source_blob_container>
                <source_prefix_name>prefix-1</source_prefix_name>
                <dest_storage_account>storageacct2</dest_storage_account>
                <dest_blob_container>udaesblob03</dest_blob_container>
                <dest_prefix_name>prefix-3/</dest_prefix_name>
        </target_group_azure_maps>
</dscConfigTargetGroupMap>
```

XML File Example - GCP Storage Target Group Map

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<dscConfigTargetGroupMap xmlns="http://schemas.teradata.com/v2012/DSC">
        <!--  This map to be used only for GCP target groups. -->
```

```
    <master_source_target_name>gcp_tg1</master_source_target_name>
    <master_source_dsc_id>local-dsc</master_source_dsc_id>
    <master_dest_target_name>gcp_tg2</master_dest_target_name>
    <target_group_gcp_maps>
        <source_account_name>storageaccount1</source_account_name>
        <source_bucket_name>bucket1</source_bucket_name>
        <source_prefix_name>prefix1/</source_prefix_name>
        <dest_account_name>storageaccount2</dest_account_name>
        <dest_bucket_name>bucket2</dest_bucket_name>
        <dest_prefix_name>prefix2/</dest_prefix_name>
    </target_group_gcp_maps>
</dscConfigTargetGroupMap>
```

consolidate_job_logs

Purpose

The `consolidate_job_logs` command uploads all logs for a completed job to a centralized location. If the job is running, the command is rejected.

Syntax

consolidate_job_logs -n|-name *Name* -I|-job_execution_id *Job Execution ID*

Example

```
dsc consolidate_job_logs -n job1 -I 5
```

Parameters

n|name *Name*

The name of the job for which you want to retrieve status.

I|job_execution_id *Job Execution ID*

[Optional] The job ID of the job execution for which you want to retrieve status. You cannot specify a running job. If you do not specify a job execution ID, the latest job execution ID is used.

T|Timestamp

[Optional] Prompts for the timestamp of the job for which you want to retrieve status.

Usage Notes

The job logs that `consolidate_job_logs` gathers are placed in a subdirectory, *jobName_jobExecutionId_currentTimestamp*, that is created under the upload directory defined in the

`dsarest.properties` file. It copies all DSC log files with a last modified timestamp after the job start time to the newly created directory.

Note:

To maintain available disk file space, delete old files in the upload directory periodically.

XML File Example

This command does not support an XML file.

create_job

Purpose

The `create_job` command creates a job based on the values you specify for parameters in the command line or in the XML file. Parameter values you enter in the command line supersede any value you enter for those parameters in the parameters XML file.

Syntax

create_job [-file *File* | parameters]

Example

```
dsc create_job -n job1 -f parameters.xml
```

Parameters

n|name *Name*

The name of the job on which to perform the action. Must be unique for each job.

Example:
`job1`

f|file *filename*

The full file path and file name of the file containing the necessary parameters to create the job. If the same parameters are provided both in the file and on the command line, Teradata DSA uses the values specified in the command line.

Example:
`parameters.xml`

d|description *Description*

A meaningful description of the job. To allow a multiword description, add \" before and after the description string.

Example:

```
\"backup web apps\"
```

t|type *Type*

The type of job. Enter one of the following:

- `BACKUP`
- `RESTORE`
- `ANALYZE_READ`
- `ANALYZE_VALIDATE`

Example:
`RESTORE`

b|backup_name *BackupName*

[Optional] An existing backup job name. For restore, analyze_read, and analyze_validate jobs only.

Example:
`backupWeb1`

u|user_authentication *User*

Required when security management is enabled. Supplies the command with the Viewpoint user, and triggers a password prompt for authentication.

Permissions required: User role

v|backup_version *BackupVersion*

[Optional] Backup version number to be replicated to or from third party storage. This number is visible in the backup filename (`bar_123456789.out`)

If you specify `next_incremental_restore`, do not specify a backup version number.

Example:
`2`

Usage Notes

A DBC only backup job cannot be run as online.

To get the backup version of a save set, use the `list_save_sets` command.

Set the skip_archive parameter to `true` to have the job continue to run if a table is non-existant; for example, access rights for objects are missing or a table specifically listed in a job file is missing (for example, dropped). By setting the parameter to `true`, the job skips the missing table and continues processing the

remaining tables. The default value for skip_archive is `false`. The default value for existing jobs should be `false` and can be set during the upgrade procedure.

In a restore job, the parent_name parameter uses the backup job name when exporting the XML.

In a restore job, the map_to parameter is used to map one database to a new database or to map a table level object under multiple databases to a new database. Job creation fails under the following circumstances:

- Multiple databases map to same database
- Same table level object name under different databases maps to the same database
- Database A is mapped to newA, and table level object x under database B is mapped to newA.x

If you have defined map_to, rename_to, or config_map_name for table level objects, this affects the results of a database level restore job definition.

- If database is selected and includeAll attribute, map_to, or config_map_name is *not* set at the database level, the DSC expands objects under the database and treats the restore job as table level restore.
- If database is selected and includeAll attribute, map_to, or config_map_name is set at the database level, the table level definitions are ignored, and the DSC treats the restore job as database level restore.

In a backup, restore, or analyze_validate job, use the streams_softlimit job option to override the system level streams soft limit. Range is 1- *max* soft limit of the stream soft limit, where *max* is the number of AMPs per node in the system. For more information about optimal settings, see Values for XML Elements.

In a backup or restore job, use the skip_joinhash_index job option to skip archiving the join and hash index during a backup job or to skip the join index restore during a restore job.

For an incremental restore backup job, use the enable_backup_for_IR job option to create the backup job for incremental restore. This can only be set during create_job. Once the job is created, it cannot be set for incremental restore.

Sample XML: Remote Backup Job

Job parameters for a remote backup job:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

              <!--****This command must be on one line.****-->
<dscCreateJob xmlns="http://schemas.teradata.com/v2012/DSC" xmlns:xsi="http://
www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="DSC.xsd">

<!--   Optional -->
<job_instance>
<!-- 'job_name' - Required, max 128 characters.-->
<!-- Note: Can be omitted in xml to be specified via command line. -->
  <job_name>SampleRemoteBackup1</job_name>
```

```
<!-- 'job_description' - Required, max 256 characters. -->
<!-- Note: Can be omitted in xml to be specified via command line. -->
  <job_description>This is a sample remote backup job.</job_description>

<!-- 'job_type' - Required, accepted values: BACKUP, RESTORE, ANALYZE_VALIDATE,
ANALYZE_READ -->
<!-- Note: Can be omitted in xml to be specified via command line. -->
  <job_type>BACKUP</job_type>

<!-- 'auto_retire' - Optional, accepted values: true/false    -->
  <auto_retire>true</auto_retire>

<!-- 'retire_value' - Required (if auto_retire is true) -->
  <retire_value>15</retire_value>

<!-- 'retire_units' - Required (if auto_retire is true), accepted values: DAYS,
WEEKS -->
  <retire_units>DAYS</retire_units>

<!-- 'objectlist' - Required for BACKUP, RESTORE, ANALYZE_VALIDATE -->
<!-- Note: 'objectlist' is NOT permitted for ANALYZE_READ jobs -->
  <objectlist>

  <objectinfo>
  <!-- 'object_name' - Required, max 128 characters -->
    <object_name>CreateQCFVer</object_name>

  <!-- 'object_type' - Required, accepted values: AGGREGATE_FUNCTION,
AUTHORIZATION, COMBINED_AGGREGATE_FUNCTIONS, DATABASE, EXTERNAL_PROCEDURE,
GLOP_SET, HASH_INDEX, INSTANCE_OR_CONSTRUCTOR_METHOD, JAR, JOIN_INDEX, MACRO,
NO_PI_TABLE, ORDERED_ANALYTICAL_FUNCTION, QUEUE_TABLE, STANDARD_FUNCTION,
STATISTICAL_FUNCTION, STORED_PROCEDURE, TABLE, TABLE_FUNCTION, TRIGGER, USER,
USER_DEFINED_METHOD, USER_DEFINED_DATA_TYPE, VIEW -->
    <object_type>TABLE</object_type>

      <!-- 'parent_name' - Required, max 128 characters -->
        <parent_name>SystemFe</parent_name>

      <!-- 'parent_type' - Required, accepted values: DATABASE/USER -->
        <parent_type>DATABASE</parent_type>
    </objectinfo>

    <objectinfo>
      <object_name>ABC</object_name>
```

```
      <object_type>DATABASE</object_type>
      <parent_name>DBC</parent_name>
      <parent_type>DATABASE</parent_type>
      <object_attribute_list>
        <includeAll>false</includeAll>
      </object_attribute_list>
      <!-- 'exclude' - Optional -->
        <exclude>
          <excludeobjectinfo>
            <object_name>T1</object_name>
            <object_type>TABLE</object_type>
          </excludeobjectinfo>
          <excludeobjectinfo>
            <object_name>T2</object_name>
            <object_type>TABLE</object_type>
          </excludeobjectinfo>
        </exclude>
      </objectinfo>
  </objectlist>
  </job_instance>

<!-- 'source_tdpid' - Required, max 32 characters -->
  <source_tdpid>system1</source_tdpid>
<!-- 'source_accountid' - Optional, max 30 characters -->
  <source_accountid>acctid</source_accountid>

<!-- 'target_media' - Required, max 32 characters -->
  <target_media>remote1_nbu</target_media>

<!-- 'job_options' - Required -->
  <job_options>
<!-- 'online'- Optional, accepted values: true/false -->
  <online>false</online>

<!-- 'nosync' - Optional, accepted values: true/false -->
  <nosync>false</nosync>

<!-- 'data_phase' - Required,  accepted values: DATA,DICTIONARY  -->
  <data_phase>DATA</data_phase>

<!-- 'query_band' - Optional, max 2048 characters -->
  <query_band>queryBandTest</query_band>

<!-- 'dsmain_logging_level' - Optional, accepted values: Error, Info, Debug,
```

```
Warning  -->
  <dsmain_logging_level>Error</dsmain_logging_level>

<!-- 'nowait' - Optional, accepted values: true/false -->
  <nowait>false</nowait>

<!-- skip_archive -Optional can be true/false. Skip archive during the backup
job in case error in access rights. Supported in DBS 15.10.01 and later -->
  <skip_archive>true</skip_archive>

<!-- skip_stats - Optional can be true/false, default is false. Skip collecting
the stats during the backup job. Supported in DBS 16.00 and later -->
  <skip_stats>true</skip_stats>

<!-- track_empty_tables -Optional can be true/false. When it's true, it will
track empty table during the backup. Supported in DBS 16.00 and above -->
  <track_empty_tables>true</track_empty_tables>

<!-- streams_softlimit -Optional for BACKUP/RESTORE/ANALYZE_VALIDATE -->
<!-- Job level stream softlimit. This value will override system level
stream softlimit-->
  <streams_softlimit>10</streams_softlimit>

<!-- single_object_locking - Optional, can be true/false, default is false.
Enables the Enhanced Lock Management feature, to lock objects in a backup one
at a time rather than all at the beginning of the backup. -->
  <single_object_locking>false</single_object_locking>

<!-- skip_joinhash_index - Optional can be true/false, default is false. Skip
archiving the join and hash index during the backup job. Supported in DBS 16.20
and later -->
  <skip_joinhash_index>true</skip_joinhash_index>

  </job_options>
</dscCreateJob>
```

Sample XML: Backup Job Enabled for Incremental Restore

Job parameters for a remote backup job enabled for incremental restore:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

              <!--****This command must be on one line.****-->
<dscCreateJob xmlns="http://schemas.teradata.com/v2012/DSC"
```

```
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:schemaLocation="DSC.xsd">
<!-- Optional -->
  <job_instance>
    <!-- 'job_name' - Required, max 128 characters.-->
    <!-- Note: Can be omitted in xml to be specified via command line. -->
      <job_name>SampleRemoteBackup1</job_name>

    <!-- 'job_description' - Required, max 256 characters. -->
    <!-- Note: Can be omitted in xml to be specified via command line. -->
      <job_description>This is a sample remote backup job.</job_description>

    <!-- 'job_type' - Required, accepted values: BACKUP, RESTORE,
ANALYZE_VALIDATE, ANALYZE_READ -->
    <!-- Note: Can be omitted in xml to be specified via command line. -->
      <job_type>BACKUP</job_type>

    <!-- 'auto_retire' - Optional, accepted values: true/false -->
      <auto_retire>true</auto_retire>

    <!-- 'retire_value' - Required (if auto_retire is true) -->
      <retire_value>15</retire_value>

    <!-- 'retire_units' - Required (if auto_retire is true), accepted values:
DAYS, WEEKS -->
      <retire_units>DAYS</retire_units>

    <!-- 'objectlist' - Required for BACKUP/RESTORE/ANALYZE_VALIDATE -->
    <!-- Note: 'objectlist' is NOT permitted for ANALYZE_READ jobs -->
      <objectlist>

        <objectinfo>
        <!-- 'object_name' - Required, max 128 characters -->
          <object_name>CreateQCFVer</object_name>

        <!-- 'object_type' - Required, accepted values: AGGREGATE_FUNCTION,
AUTHORIZATION, COMBINED_AGGREGATE_FUNCTIONS, DATABASE, EXTERNAL_PROCEDURE,
GLOP_SET, HASH_INDEX, INSTANCE_OR_CONSTRUCTOR_METHOD, JAR, JOIN_INDEX, MACRO,
NO_PI_TABLE, ORDERED_ANALYTICAL_FUNCTION, QUEUE_TABLE, STANDARD_FUNCTION,
STATISTICAL_FUNCTION, STORED_PROCEDURE, TABLE, TABLE_FUNCTION, TRIGGER, USER,
USER_DEFINED_METHOD, USER_DEFINED_DATA_TYPE, VIEW -->
           <object_type>TABLE</object_type>

             <!-- 'parent_name' - Required, max 128 characters -->
```

```
              <parent_name>SystemFe</parent_name>

            <!-- 'parent_type' - Required, accepted values: DATABASE, USER -->
              <parent_type>DATABASE</parent_type>
        </objectinfo>

        <objectinfo>
          <object_name>ABC</object_name>
          <object_type>DATABASE</object_type>
          <parent_name>DBC</parent_name>
          <parent_type>DATABASE</parent_type>
          <object_attribute_list>
            <includeAll>false</includeAll>
          </object_attribute_list>
          <!-- 'exclude' - Optional -->
            <exclude>
              <excludeobjectinfo>
                <object_name>T1</object_name>
                <object_type>TABLE</object_type>
              </excludeobjectinfo>
            <excludeobjectinfo>
              <object_name>T2</object_name>
              <object_type>TABLE</object_type>
            </excludeobjectinfo>
          </exclude>
        </objectinfo>
      </objectlist>
  </job_instance>

<!-- 'source_tdpid' - Required, max 32 characters -->
  <source_tdpid>system1</source_tdpid>
  <!-- 'source_accountid' - Optional, max 30 characters -->
    <source_accountid>acctid</source_accountid>

  <!-- 'target_media' - Required, max 32 characters -->
    <target_media>remote1_nbu</target_media>

  <!-- 'job_options' - Required -->
    <job_options>
    <!-- 'online'- Optional, accepted values: true/false -->
      <online>false</online>

    <!-- 'nosync' - Optional, accepted values: true/false -->
      <nosync>false</nosync>
```

```
    <!-- 'data_phase' - Required,  accepted values: DATA, DICTIONARY -->
      <data_phase>DATA</data_phase>

    <!-- 'query_band' - Optional, max 2048 characters -->
      <query_band>queryBandTest</query_band>

    <!-- 'dsmain_logging_level' - Optional, accepted values: Error, Info, Debug,
Warning -->
        <dsmain_logging_level>Error</dsmain_logging_level>

    <!--   'nowait' - Optional, accepted values: true/false -->
      <nowait>false</nowait>

    <!-- skip_archive -Optional can be true/false. Skip archive during the
backup job in case error in access rights. Supported in DBS 15.10.01 and later --
>
      <skip_archive>true</skip_archive>

    <!-- skip_stats - Optional can be true/false, default is false. Skip
collecting the stats during the backup job. Supported in DBS 16.00 and later -->
      <skip_stats>true</skip_stats>

    <!-- track_empty_tables -Optional can be true/false. When it's true, it will
track empty table during the backup. Supported in DBS 16.00 and later -->
      <track_empty_tables>true</track_empty_tables>

    <!-- streams_softlimit -Optional for BACKUP/RESOTRE/ANALYZE_VALIDATE -->
    <!-- Job level stream softlimit. This value will override system level
stream softlimit-->
      <streams_softlimit>10</streams_softlimit>

    <!-- skip_joinhash_index - Optional can be true/false, default is false.
Skip archiving the join and hash index during the backup job. Supported in DBS
16.20 and later -->
      <skip_joinhash_index>true</skip_joinhash_index>

    <!-- enable_backup_for_IR - Optional can be true/false, default is false.
Enable the IR feature during backup job. Supported in DBS 16.20 Update 2 and
later -->
      <enable_backup_for_IR>true</enable_backup_for_IR>
  </job_options>
</dscCreateJob>
```

Sample XML: Remote Restore Job

Job parameters for a remote restore job

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

          <!--****This command must be on one line.****-->
<dscCreateJob xmlns="http://schemas.teradata.com/v2012/DSC"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="DSC.xsd">

<!-- Optional -->
<job_instance>

<!-- 'job_name' - Required, max 128 characters.-->
<!-- Note: Can be omitted in xml to be specified via command line. -->
<job_name>SampleRemoteRestore1</job_name>

<!-- 'job_description' - Required, max 256 characters. -->
<!-- Note: Can be omitted in xml to be specified via command line. -->
<job_description>This is a sample remote restore job.</job_description>

<!-- 'job_type' - Required, accepted values: BACKUP,RESTORE, ANALYZE_VALIDATE,
ANALYZE_READ -->
<!-- Note: Can be omitted in xml to be specified via command line. -->
<job_type>RESTORE</job_type>

<!-- 'auto_retire' - Optional, accepted values: true/false -->
<auto_retire>true</auto_retire>

<!-- 'retire_value' - Required (if auto_retire is true) -->
<retire_value>15</retire_value>

<!-- 'retire_units' - Required (if auto_retire is true), accepted values: DAYS,
WEEKS -->
<retire_units>DAYS</retire_units>

<!-- 'objectlist' - Required for BACKUP, RESTORE, ANALYZE_VALIDATE -->
<!-- Note: 'objectlist' is NOT permitted for ANALYZE_READ jobs -->
<objectlist>

<objectinfo>
<!-- 'object_name' - Required, max 128 characters -->
<object_name>BAR1</object_name>

<!-- 'object_type' - Required, accepted values: AGGREGATE_FUNCTION, AUTHORIZATION,
COMBINED_AGGREGATE_FUNCTIONS, DATABASE, EXTERNAL_PROCEDURE, FUNCTION_ALIAS,
GLOP_SET, HASH_INDEX, INSTANCE_OR_CONSTRUCTOR_METHOD, JAR, JOIN_INDEX, JOURNAL,
MACRO, NO_PI_TABLE, ORDERED_ANALYTICAL_FUNCTION, QUEUE_TABLE, STANDARD_FUNCTION,
STATISTICAL_FUNCTION, STORED_PROCEDURE, TABLE, TABLE_FUNCTION, TRIGGER, USER,
USER_DEFINED_METHOD, USER_DEFINED_DATA_TYPE, VIEW -->
<object_type>VIEW</object_type>

<!-- 'parent_name' - Required, max 128 characters -->
<parent_name>DBC</parent_name>

<!-- 'parent_type' - Required, accepted values: DATABASE, USER -->
<parent_type>DATABASE</parent_type>
</objectinfo>

<objectinfo>
<object_name>BAR</object_name>
```

```
<object_type>DATABASE</object_type>
<parent_name>DBC</parent_name>
<parent_type>DATABASE</parent_type>

<!-- 'exclude' - Optional -->
<exclude>
<excludeobjectinfo>
<object_name>T1</object_name>
<object_type>TABLE</object_type>
</excludeobjectinfo>
<excludeobjectinfo>
<object_name>T2</object_name>
<object_type>TABLE</object_type>
</excludeobjectinfo>
</exclude>
</objectinfo>

<objectinfo>
<object_name>BAR2</object_name>
<object_type>VIEW</object_type>
<parent_name>DBC</parent_name>
<parent_type>DATABASE</parent_type>

<!-- 'object_attribute_list' Optional -->
<object_attribute_list>

<!-- 'map_to' - Optional, max 128 characters -->
<map_to>DBC2</map_to>

<!-- 'rename_to' - Optional, max 128 characters -->
<rename_to>BAR2</rename_to>

<!-- 'config_map_name' - Optional hash map name, max 128 characters -->
<config_map_name>objectLevelMap</config_map_name>
</object_attribute_list>
</objectinfo>
</objectlist>

<!-- 'backup_name'- Required for RESTORE, ANALYZE_READ, ANALYZE_VALIDATE jobs. -->
<backup_name>SampleRemoteBackup1</backup_name>

<!-- 'backup_version' - Required for RESTORE, ANALYZE_READ, ANALYZE_VALIDATE jobs.
Optional for incremental restore job. -->
<!-- Note: Value = '0', latest save set from backup. Value > 0 is an existing save
set.-->
<backup_version>0</backup_version>

<!-- next_incremental_restore - Optional for incremental restore job. Default is
false. Set to true to indicate restore to automatically apply the next available save
set. If set to true, you cannot specify backup_version. Supported in Analytics
Database 16.20 Update 2 and later-->
<next_incremental_restore>true</next incremental_restore>

<!-- 'all_backup_objects' - Required for RESTORE, ANALYZE_READ, ANALYZE_VALIDATE
jobs. -->
<!-- Accepted values: true/false. -->
<!-- Note: If False then objectlist is required. -->
<all_backup_objects>false</all_backup_objects>

</job_instance>

<!-- 'source_media' - Required, max 32 characters -->
<source_media>remote1_nbu</source_media>
```

```
<!-- 'target_tdpid' - Required, max 32 characters -->
<target_tdpid>system1</target_tdpid>

<!-- 'target_accountid' - Optional, max 30 characters -->
<target_accountid>acctid</target_accountid>

<!-- 'job_options' - Required -->
<job_options>

<!-- 'data_phase' - Optional, accepted values: DATA, DICTIONARY  -->
<data_phase>DATA</data_phase>

<!-- 'enable_temperature_override' - Optional, accepted values: true/false -->
<enable_temperature_override>false</enable_temperature_override>

<!-- 'temperature_override' - Optional, accepted values: DEFAULT, HOT, WARM, COLD --
>
<temperature_override>DEFAULT</temperature_override>

<!-- 'block_level_compression' - Optional, accepted values: DEFAULT, ON, OFF -->
<block_level_compression>DEFAULT</block_level_compression>

<!-- 'disable_fallback' - Optional, accepted values: true/false -->
<disable_fallback>false</disable_fallback>

<!-- 'query_band' - Optional, max 2048 characters -->
<query_band>queryBandTest</query_band>

<!-- 'dsmain_logging_level' - Optional, accepted values: Error, Info, Debug, Warning
-->
<dsmain_logging_level>Error</dsmain_logging_level>

<!-- 'reblock' - Optional, accepted values: true/false -->
<!-- Note: Only useful for RESTORE jobs.-->
<reblock>false</reblock>

<!-- 'run_as_copy' - Optional, accepted values: true/false -->
<!-- Note: Only useful for RESTORE jobs.-->
<run_as_copy>false</run_as_copy>

<!-- 'skip_archive' -Optional can be true/false. Skip archive during the restore job
in case error in access rights. Supported in DBS 15.10.01 and later -->
<skip_archive>true</skip_archive>

<!-- 'skip_stats' - Optional can be true/false, default is false. Skip restoring the
stats during the restore job. Supported in DBS 16.00 and later -->
<skip_stats>true</skip_stats>

<!-- 'config_map_name' - Optional, Hash map name, max 128 characters -->
<config_map_name>jobLevelMap</config_map_name>

<!-- 'concurrent_bld_per_tbl' - Optional, Number of index and fallback subtables
that can be built concurrently per table during restore. -->
<!-- Range 1-5 for database version 16.0 and later -->
<!-- Note: Only useful for RESTORE jobs. -->
<concurrent_bld_per_tbl>5</concurrent_bld_per_tbl>

<!--  'map_to' - Optional, apply map_to to all objects in the restore job. Supported
in Teradata Database 16.00 and later -->
<map_to>new_db</map_to>

<!-- 'streams_softlimit' - Optional for BACKUP, RESTORE, ANALYZE_VALIDATE -->
<!-- Job level stream softlimit. This value will override system level stream
softlimit. -->
```

```
<!-- Value can be 1 - max soft limit of the stream soft limit, where max is the
number of AMPS per node in the system. -->
<streams_softlimit>10</streams_softlimit>

<!-- skip_joinhash_index - Optional. Can be true/false. Default is false. Skip
restoring the join and hash index during the backup job. Supported in DBS 16.20 and
later -->
<skip_joinhash_index>true</skip_joinhash_index>

<!-- enable_incremental_restore - Optional. Can be true/false. Default is false.
Enable incremental restore feature when creating restore job. Supported in Analytics
Database 16.20 Update 2 and later -->
<enable_incremental_restore>false</enable_incremental_restore>

<!-- skip_build_secondary_indexes - Optional. Can be true/false. Default is false.
Skip building the secondary index during the restore job. Supported in Analytics
Database 16.20 Update 2 and later -->
<skip_build_secondary_indexes>fales</skip_build_secondary_indexes>

</job_options>
</dscCreateJob>
```

Command XML for Incremental Restore job

```
<enable_backup_for_IR>true</enable_backup_for_IR>


<!—include_DBC_data -- Optional. Can be true/false, default is false. Enable the
IR feature for DBC tables during backup job. Supported in DBS 17.10 and later -->


<include_DBC_data>true</include_DBC_data>
```

delete_component

Purpose

The `delete_component` command deletes an existing component based on the information in the parameters.

Syntax

dsc delete_component [parameters]

Example

```
dsc delete_component -n system1 -t target_group -S

dsc delete_component -t target_group -r -n replication-tg-S3

dsc delete_component -t target_group -r -n replication-tg-S3 -d
```

Parameters

n|name *Name*

The name of the BAR component. The name must be unique.

t|type *Type*

The type of BAR component. Enter one of the following:

- `system`
- `media_server`
- `nbu_server`
- `dd_boost`
- `disk_file_system`
- `target_group`
- `aws_app`
- `azure_app`
- `gcp_app`
- `network_fabric`

Example:
`target_group`

r|repository_replication

Indicates the target group for deletion is a replication target group.

d|delete_all_data

> **NOTICE**
>
> Deletes the replication target group and all save set files associated with it. Be very careful if you choose to use this parameter.

S|skip_prompt *SkipPrompt*

[Optional] Skips displaying a confirmation message before performing the command action.

u|user_authentication *User*

Required when security management is enabled. Supplies the command with the Viewpoint user, and triggers a password prompt for authentication.

Permissions required: Administrator role

Usage Notes

To use the `delete_component` command, you must specify the component name and type. You do not need to specify the component name for disk_file_system types.

You can delete a system, media server, NetBackup server, disk file system, DD Boost server, Amazon S3 storage account, Azure Blob storage account, Google Cloud Account or a target group, except under the following conditions:

- System marked for repository backup
- System in use by a job
- Media server in use by a target group
- NetBackup server in use by a target group
- Target group in use by a job
- Target group in use by a target group map
- Target group in use for repository backups, unless it is a repository_replication target group
- Policy used by a target group

Note:

Beginning with the DSA 16.10 release, nodes are autodiscovered and cannot be manually deleted.

XML File Example

This command does not support an XML file.

delete_job

Purpose

The `delete_job` command deletes a DSA job and any data associated with it from the DSC repository. Any logs and job history are deleted and cannot be restored. Any backup save sets created for the job that exist on devices managed by third-party solutions must be deleted manually using the interface for that solution.

Syntax

dsc delete_job -n|-name *Name*

Example

```
dsc delete_job -n job1 -S
```

Parameters

n|name *Name*

The name of the job on which to perform the action. Must be unique for each job.

Example:

```
job1
```

S|skip_prompt *SkipPrompt*

[Optional] Skips displaying a confirmation message before performing the command action.

u|user_authentication *User*

Required when security management is enabled. Supplies the command with the Viewpoint user, and triggers a password prompt for authentication.

Permissions required: DSA Job Permissions: Run and Edit.

Usage Notes

Generally, to delete a backup job, you must retire it first. There is one exception: you can also delete an active backup job if its status is New.

A backup job cannot be deleted if there are restore or analyze jobs associated with it.

XML File Example

This command does not support an XML file.

Related Information:

Job Settings Tab

delete_save_sets

Purpose

The `delete_save_sets` command deletes the backup save sets for a backup job using a single or multiple job execution ID. It does not delete save sets for repository jobs.

Syntax

dsc delete_save_sets -I|job_execution_id *id* -S|skip_prompt

Example

```
dsc delete_job -I 10011 -S
```

Parameters

I|job_execution_id *Job Execution ID*

The job execution ID of the previous execution of the job.

O|override

[Optional] Flag used to delete related save sets. For example, if a Full save set is related to a Delta save set, use the override flag to also delete the Delta save set.

f|file *file.xml*

The file to specify multiple execution ID to delete multiple save sets. Either one of -I or -f should be present.

S|skip_prompt *SkipPrompt*

[Optional] Skips displaying a confirmation message before performing the command action.

u|user_authentication *User*

Required when security management is enabled. Supplies the command with the Viewpoint user, and triggers a password prompt for authentication.

Permissions required: DSA Job Permissions: Run and Edit.

A|abort

[Optional] Abort the current running request so the DSC does not delete the metadata. Cannot abort the deletion of save set files.

C|current

[Optional] Returns the current status of the recently executed delete save sets request.

Usage Notes

- Delete_save_sets allows you to cleanup files and metadata when they are no longer needed. It deletes save sets and backup images from both the DSC repository and the backup solution.
- When backups go across multiple TMSS dsc servers, the image files are not deleted if the TMSS servers use the same REMOTE_FILE_SYSTEM in the target group definition. See config_target_groups for more information.

XML File Example

Here is a sample XML file.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<!--  Copyright (C) 2009 by Teradata Corporation. All Rights Reserved. TERADATA
        CORPORATION CONFIDENTIAL AND TRADE SECRET -->
<dscDeleteSaveSets
        xmlns="http://schemas.teradata.com/v2012/DSC"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="DSC.xsd">
```

```
        <!--  List of multiple execution Id for the same job to delete save
sets -->
        <job_execution_id>150</job_execution_id>
        <job_execution_id>151</job_execution_id>
        <job_execution_id>152</job_execution_id>
</dscDeleteSaveSets>
```

Related Information:

Job Settings Tab

delete_target_group_map

Purpose

The `delete_target_group_map` command deletes a target group map for restoring to a different client configuration.

Syntax

delete_target_group_map -s|-source *SampSrc* -d|-destination *SampDest*

Example

```
dsc delete_target_group_map -s SampSrc -d SampDest
```

Parameters

s|source *SampSrc*

The source target group represents the backup target group for mapping.

Example:
`SampSrc`

d|destination *SampDest*

The destination target group represents the restore target group for mapping.

Required if the virtual target group (parameter **V**) is specified. It indicates the physical target group to which the virtual target group is mapped.

Example:
`SampDest`

V|virtual

[Optional] Indicates this is a mapping from a virtual target group to a physical target group.

l | dsc_name *SampleDscName*

Required if -V is specified. The DSC name of the source DSC environment. Do not use without -V.

u|user_authentication *User*

Required when security management is enabled. Supplies the command with the Viewpoint user, and triggers a password prompt for authentication.

Permissions required: Administrator role

XML File Example

This command does not support an XML file.

disable_component

Purpose

The `disable_component` command disables an existing BAR component based on the component name and type.

Syntax

dsc disable_component -n|-name *Name* -t|-type *Type*

Example

```
dsc disable_component -n system1 -t system
```

Parameters

n|name *Name*

The name of the BAR component. The name must be unique.

Example:
`system1`

t|type *Type*

The type of BAR component. Enter `system` or `target_group`.

Example:
`SYSTEM`

u|user_authentication *User*

Required when security management is enabled. Supplies the command with the Viewpoint user, and triggers a password prompt for authentication.

Permissions required: Administrator role

Usage Notes

You cannot disable a component when it is in use.

XML File Example

This command does not support an XML file.

discover_internals

Purpose

The `discover_internals` command discovers internals based on AXM type support and type of backup application; for example, NBU policies.

Syntax

discover_internals s|server *thirdpartyserver* u|user_authentication *User*

Example

```
discover_internals -s nbu_server1 -u
```

Parameters

s|server *thirdpartyserver*

The name of the third party server

Example:
`nbu_server1`

u|user_authentication *User*

Required when security management is enabled. Supplies the command with the Viewpoint user, and triggers a password prompt for authentication.

Permissions required: Administrator role

Usage Notes

discover_internals collects the NBU policies and their associated number of devices and saves them into the DSC repository.

XML File Example

This command does not support an XML file.

enable_component

Purpose

The `enable_component` command enables an existing BAR component based on the component name and type.

Syntax

enable_component -n|-name *Name* -t|-type *Type*

Example

```
dsc enable_component -n system1 -t system
```

Parameters

n|name *Name*

The name of the BAR component. The name must be unique.

Example:
`system1`

t|type *Type*

The type of BAR component. Enter `system` or `target_group`.

Example:
`SYSTEM`

u|user_authentication *User*

Required when security management is enabled. Supplies the command with the Viewpoint user, and triggers a password prompt for authentication.

Permissions required: Administrator role

Usage Notes

You must enable a system or target group before running a job that uses it.

XML File Example

This command does not support an XML file.

export_config

Purpose

The `export_config` command exports the current XML definition for the requested BAR component.

Syntax

export_config -n|-name *Name* -t|-type *Type* -r|repository_replication -f|-file *File*

Examples

```
export_config -n component1 -t system -f System1Config.xml
export_config -t general -f GeneralConfig.xml
export_config -t target_group -r -n replication-tg-S3 -f ReplicationGroup.xml
```

Parameters

n|name *Name*

The name of the component you want to export. This parameter is optional for the **general, repository_backup, and disk_file_system** types.

Example:
`component1`

t|type *Type*

The type of component or repository backup you want to export. Enter one of the following:

- `system`
- `media_server`
- `nbu_server`
- `dd_boost`
- `repository_backup`
- `disk_file_system`
- `target_group`
- `general`
- `AWS_APP`
- `AZURE_APP`
- `GCP_APP`
- `NETWORK_FABRIC`

Example:
`system`

r|repository_replication

Designates the target group for export as a replication target.

f|file *filename*

The full file path and file name of the file to which to write the XML definition.

Usage Notes

This command can be used with the `configuration` commands to update a BAR configuration.

XML File Example

This command does not require an XML file as input. You must supply a file name and location to which the XML file results are exported as output.

export_job

Purpose

The `export_job` command exports the current XML definition for the requested job.

Syntax

export_job -n|-name *Name* -f|-file *File*

Example

```
export_job -n job1 -f job1Definition.xml
```

Parameters

n|name *Name*

The name of the job on which to perform the action. Must be unique for each job.

Example:
`job1`

f|file *filename*

The full file path and file name of the file to which to write the XML definition.

Example:
`job1Definition.xml`

Usage Notes

The XML file for a retired job cannot be exported using the export_job command.

The XML file that is exported by using this command can be used in the `update_job` command to create a job.

XML File Example

This command does not require an XML file as input. You must supply a file name and location to which the XML file results are exported as output.

export_job_metadata

Purpose

The `export_job_metadata` command exports metadata of a job (job definition, save sets, and targets) based on the requested backup version. In the case of a disaster to the DSC repository, exporting and then importing job metadata enables job migration and restoration to a different DSA environment.

The data must be exported in the following order:

1. Targets
2. Job definition
3. Save sets

Syntax

export_job_metadata -n|-name *Name* -d|-directory *DirectoryPath* -v|-backup_version *BackupVersion* -t|-type *Type* -u|-user_authentication *User*

Example

```
dsc export_job_metadata -n job1 -v latest
```

Parameters

n|name *Name*

The name of the job on which to perform the action. Must be unique for each job.

Example:
`job1`

d|directory *DirectoryPath*

[Optional] Directory where the files are exported to or imported from.

Example:
`var/opt/dsa`

v|backup_version *BackupVersion*

[Optional] Backup version number. Enter `latest` or `0` for the latest save set.

Example:
`latest`

t|type *Type*

[Optional] The type of job metadata. You may enter `JOB` (for the job definition), `SAVESET`, or `TARGET`. If no value is specified, all three types of metadata are included.

Example:
`JOB`

u|user_authentication *User*

Required when security management is enabled. Supplies the command with the Viewpoint user, and triggers a password prompt for authentication.

Permissions required: User role

Usage Notes

When the `export_job_metadata` or `import_job_metadata` command is running, no other operations or commands can run at the same time. In addition, `export_job_metadata` and `import_job_metadata` cannot run if an operational job or repository job is running.

XML File Example

This command does not support an XML file.

export_job_status

Purpose

The `export_job_status` command exports the current XML definition for the requested job status.

Syntax

export_job_status -n|-name *Name* -I|-job_execution_id *JobExecutionID* -f|-file *Filename*

Example

```
dsc export_job_status -n job1 -I job1ExecutionId -f job1_status.xml
```

```
dsc export_job_status -n restore1 -I restore1ExecutionId -C -f job1_status.xml
```

Parameters

n|name *Name*

The name of the job on which to perform the action. Must be unique for each job.

Example:

```
job1
```

l|job_execution_id *Job Execution ID*

[Optional] The execution ID for the job. Must be an integer.

The ID must match a version number listed in the list_save_sets command output for the same job.

Example:

```
2
```

f|file *filename*

The full file path and file name of the file to which to write the XML definition.

Example:

```
job1_status.xml
```

XML File Example

This command does not require an XML file as input. You must supply a file name and location to which the XML file results are exported as output.

export_job_status_log

Purpose

The `export_job_status_log` command exports the latest information from the job_status_log to XML.

Syntax

export_job_status_log -n|-name *Name* -I|-job_execution_id *JobExecutionID* -E|-full_export -m|-hashmap *MapName* -C|-cumulative -f|-file *Filename*

Example

```
dsc export_job_status_log -n job1 -I 123456 -m specialmap -f job1_jobstatuslog.xml
dsc export_job_status_log -n restore1 -I 465 -E -C -f job1_jobstatuslog.xml
```

Parameters

n|name *Name*

The name of the job on which to perform the action. Must be unique for each job.

Example:

```
job1
```

l|job_execution_id *Job Execution ID*

[Optional] The execution ID for the job. Must be an integer.

The ID must match a version number listed in the list_save_sets command output for the same job.

Example:
2

b|bucket *BucketNumber*

[Optional] Select a bucket number to display a grouping of data when too many results return to display at once. The command output notifies you if there are more buckets of data to display.

m|hashmap *map name*

[Optional] For backup jobs, lists the objects by the specified hash map used during the backup job. Default: All maps used in the backup job are listed.

f|file *filename*

The full file path and file name of the file to which to write the XML definition.

Example:
`job1_jobstatuslog.xml`

XML File Example

This command does not require an XML file as input. You must supply a file name and location to which the XML file results are exported as output.

export_target_group_map

Purpose

The `export_target_group_map` command exports a map between target groups for restoring to a different client configuration.

Syntax

export_target_group_map -s|-source *SampSrc* -d|-destination *SampDest* -f|-file *Name*

Example

```
dsc export_target_group_map -s SampSrc -d SampDest -f TargetGroupMap.xml
```

Parameters

s|source *SampSrc*

The source target group represents the backup target group for mapping.

Example:

```
SampSrc
```

d|destination *SampDest*

The destination target group represents the restore target group for mapping.

Required if the virtual target group (parameter **V**) is specified. It indicates the physical target group to which the virtual target group is mapped.

Example:

```
SampDest
```

f|file *filename*

The full file path and file name of the file to which to write the XML definition.

```
TargetGroupMap.xml
```

V|virtual

[Optional] Indicates this is a mapping from a virtual target group to a physical target group.

l | dsc_name *SampleDscName*

Required if -V is specified. The DSC name of the source DSC environment. Do not use without -V.

XML File Example

This command does not require an XML file as input. You must supply a file name and location to which the XML file results are exported as output.

import_job_metadata

Purpose

The `import_job_metadata` command imports metadata of a job (job definition, save sets, and targets) to the specified directory. In the case of a disaster to the DSC repository, exporting and then importing job metadata enables job migration and restoration to a different DSA environment.

The data must be imported in the following order:

1. Targets
2. Job definition

3. Save sets

Syntax

import_job_metadata -n|-name *Name* -d|-directory *DirectoryPath* -v|-backup_version *BackupVersion* -t|-type *Type* -u|-user_authentication *User*

Example

```
dsc import_job_metadata -n job1 -v latest
```

Parameters

n|name *Name*

The name of the job on which to perform the action. Must be unique for each job.

Example:
`job1`

d|directory *DirectoryPath*

[Optional] Directory where the files are exported to or imported from.

Example:
`var/opt/dsa`

v|backup_version *BackupVersion*

[Optional] Backup version number. Enter `latest` or `0` for the latest save set.

Example:
`latest`

t|type *Type*

[Optional] The type of job metadata. You may enter `JOB` (for the job definition), `SAVESET`, or `TARGET`. If no value is specified, all three types of metadata are included.

Example:
`JOB`

u|user_authentication *User*

Required when security management is enabled. Supplies the command with the Viewpoint user, and triggers a password prompt for authentication.

Permissions required: User role

Usage Notes

When the `export_job_metadata` or `import_job_metadata` command is running, no other operations or commands can run at the same time. In addition, `export_job_metadata` and `import_job_metadata` cannot run if an operational job or repository job is running.

XML File Example

This command does not support an XML file.

job_status

Purpose

The `job_status` command gets the latest status for a job with the given name and displays it on the screen. If the job is running, a detailed status message is displayed, unless the job is collecting cumulative status on a restarted restore job. In that case, the status is not available until the job is complete. If the job is not running, the status of the last run for that job is displayed.

This command displays the latest status for a job in the screen.

If Advanced Job Status Enhancements is enabled then,`job_status` command displays additional fields such as Lock Status and HUT Lock acquired details for the backup job and backup job component of DM or DTU jobs. Similarly, for the restore or copy job and the restore job component of DM or DTU jobs, it displays number of task sets completed, counts of Primary, Online, and Primary LOB sub-tables sorted during DSA's Data and Build Phase.

Syntax

dsc job_status -n|-name *Name* -I|-job_execution_id *JobExecutionID* -B|repository_backup -C|cumulative

Example

```
dsc job_status -n job1 -I job1ExecutionId
```

```
dsc job_status -n restore1 -I restore1ExecutionId -C
```

Parameters

n|name *Name*

The name of the job on which to perform the action. Must be unique for each job.

Example:
`job1`

I|job_execution_id *Job Execution ID*

[Optional] The execution ID for the job. Must be an integer.

The ID must match a version number listed in the list_save_sets command output for the same job.

Example:
2

B|backup_repository
[Optional] Flag to return status on repository backup jobs.

C|cumulative

[Optional] Flag to return cumulative job status for a restore job that has been restarted. Status is provided up to the last completed restore.

XML File Example

This command does not support an XML file.

job_status_log

Purpose

The `job_status_log` command displays the latest status log for a job with the given name if the job is running. If the job is not running, the status log for the last run job is displayed.

If Advanced Job Status Enhancements is enabled then, `job_status_log` command displays additional fields such as online status, online enabled time for the objects in the backup job, and the backup job component of DM/DTU jobs.

Similarly, for restore or copy job type and restore job component of DM or DTU jobs; additional fields such as SubTable object's sort phase type and state, sort start time and sort estimated end time will be displayed.

Syntax

job_status_log -n|-name *Name* -I|-job_execution_id *JobExecutionID* -E|-full_export -m|-hashmap *MapName* -C|-cumulative

Example

```
dsc job_status_log -n job1 -I 123456 -m specialmap
```

```
dsc job_status_log -n restore1 -I 465 -E -C
```

Parameters

n|name *Name*
The name of the job on which to perform the action. Must be unique for each job.

Example:

```
job1
```

l|job_execution_id *Job Execution ID*

[Optional] The execution ID for the job. Must be an integer.

The ID must match a version number listed in the list_save_sets command output for the same job.

Example:

```
2
```

B|backup_repository

[Optional] Flag to return status on repository backup jobs.

b|bucket *BucketNumber*

[Optional] Select a bucket number to display a grouping of data when too many results return to display at once. The command output notifies you if there are more buckets of data to display.

m|hashmap *map name*

[Optional] For backup jobs, lists the objects by the specified hash map used during the backup job. Default: All maps used in the backup job are listed.

E|full_export

[Optional] Retrieve and output the entire job status log to a landing zone on the DSC server to avoid the message size limit. The output file is a comma-separated value with the file name: <*job_name*>_<*job_execution_id*>.csv. The default landing zone directory is `/var/opt/teradata/dsa/export`. You can change the directory in `dsc.property` using the `fullExport.landingZone` property.

Note:

This parameter works only for BACKUP, RESTORE, and ANALYZE_VALIDATE jobs on the Analytics Database / Teradata Database. It does not work for DBC repository backup jobs.

C|cumulative

[Optional] Flag to return cumulative job status for a restore job that has been restarted. Status is provided up to the last completed restore.

XML File Example

This command does not support an XML file.

list_access_module

Purpose

The `list_access_module` command lists available access module types for a named media server.

Syntax

list_access_module -n|-name *Name*

Example

```
list_access_module -n|-name media_server1
```

Parameters

n|name *Name*

[Optional] The name of the media server.

XML File Example

This command does not support an XML file.

list_build_restore_objects

Purpose

The `list_build_restore_objects` command lists the status of the build_restore_objects command.

Syntax

list_build_restore_objects -n|name *Name* -b|bucket *BucketNumber*

Example

```
list_build_restore_objects -n job1 -b 150
```

Parameters

n|name *Name*

The name of the job to list the status of.

Example:
`job1`

b|bucket *BucketNumber*

The bucket to query the status of. Must display by bucket if more than 100 objects. Default: 1

Example:
`150`

u|user_authentication *User*

Required when security management is enabled. Supplies the command with the Viewpoint user, and triggers a password prompt for authentication.

Permissions required: Administrator role

XML File Example

This command does not support an XML file.

list_cleanup_ir_source_system

Purpose

The `list_cleanup_ir_source_system` command lists the results of cleanup_ir_source_system.

Syntax

list_cleanup_ir_source_system -s|-system *System*

Example

```
list_cleanup_ir_source_system -s system1
```

Parameters

s|system *System*

Name of the source system.

Example:
`system1`

XML File Example

This command does not support an XML file.

list_components

Purpose

The `list_components` command lists components defined and stored in the DSC repository. If a specific component is requested, that component definition is displayed. Otherwise, a list of the components matching any provided filters is displayed. Any partial component name returns all components matching the partial input.

Note:

The **Type** parameter is required.

Syntax

list_components -e|-enabled *true*|*false* -n|-name *Name* -s|-system *System* -t|-type *Type* -r|repository_replication

Example

```
list_components -e true -n system1 -t system
```

```
list_components -e true -t target_group -r
```

```
list_components -t target_group -r -n replication-S3
```

Parameters

e|enabled *Enabled*

[Optional] Filter components based on whether or not they are enabled. This applies only to system and target_group components.

Example:
`true`

n|name *Name*

[Optional] The name of the BAR component. The name must be unique.

Example:
`component1`

s|system *System*

[Optional] Filter components based on the associated Teradata system. This applies only to node components.

Example:
`system1`

t|type *Type*

The type of BAR component. Enter one of the following:

- `system`
- `node`
- `media_server`
- `nbu_server`
- `dd_boost`
- `disk_file_system`
- `target_group`
- `azure_app`
- `aws_app`
- `gcp_app`
- `network_fabric`

Example:
`system`

V|virtual

[Optional] Indicates this is a mapping from a virtual target group to a physical target group.

r|repository_replication

Used with type target_group, lists the replication targets.

XML File Example

This command does not support an XML file.

list_consumers

Purpose

The `list_consumers` command sends a request to ActiveMQ to provide information about all of the consumers of DSA Network Client and DSMain processes. It checks whether the selector values match the names of the DSA Network Client and DSMain systems and that the processes are running.

Syntax

list_consumers -n|-name *System1* -t|-type *SYSTEM*

Example

```
list_consumers -n System1 -t SYSTEM
```

Parameters

n|name *Name*

[Optional] The name of the BAR component. The name must be unique.

Example:
`System1`

t|type *Type*

[Optional] The type of BAR component. Enter `system` or `media_server`.

Example:
`SYSTEM`

Usage Notes

After configuring a system using the `config_systems` command, you may want to run the `list_consumers` command to validate that the selector values match the DSMain system names and that the processes are running.

After a media server configuration and restart or an auto-discovery of a media server, you may want to run the `list_consumers` command to validate that the selector value matches the DSA Network Client system name and that the consumer processes are running.

XML File Example

This command does not support an XML file.

list_discover_internals

Purpose

The `list_discover_internals` command lists internals based on AXM type support and type of backup application; for example, NBU policies.

Syntax

list_discover_internals s|server *thirdpartyserver*

Example

```
list_discover_internals -s nbu_server1
```

Parameters

s|server *thirdpartyserver*

The name of the third party server

Example:
`nbu_server1`

Usage Notes

Lists the NBU policies and their associated number of devices.

XML File Example

This command does not support an XML file.

list_general_settings

Purpose

The `list_general_settings` command lists all current general settings.

Syntax

list_general_settings

Example

```
dsc list_general_settings
```

Parameters

There are no parameters associated with this command.

XML File Example

This command does not support an XML file.

list_job_history

Purpose

The `list_job_history` command lists the complete history of the job you specify, including the type of system used during the job execution. If you do not specify a job, the history of all jobs in the DSC repository are listed.

Parameters

n|name *Name*

[Optional] The name of the DSA job to display history.

Example:
`job1`

b|bucket *BucketNumber*

[Optional] Select a bucket number to display a grouping of data when too many results return to display at once. The command output notifies you if there are more buckets of data to display.

XML File Example

This command does not support an XML file.

list_jobs

Purpose

The `list_jobs` command lists jobs defined and stored in the DSC repository. If a specific job is requested, that job definition is displayed. Otherwise, a list of job names matching any provided filters is displayed. If no parameters are provided, a list of jobs (excluding backup repository jobs) is displayed.

Syntax

list_jobs [optional parameters]

Example

```
dsc list_jobs -n job1
```

Parameters

n|name *Name*

[Optional] The name of the job definition to display.

Note:

If you specify a name, the command processes the name with wildcards at the beginning and end of the name. Specifying job1 might produce a list which includes job1, job 111 and backupjob1.

Example:
`job1`

s|state *State*

[Optional] Enter `active` or `retired`. The default is active.

Example:
`active`

S|status *Status*

[Optional] The latest status for a job. Enter one of the following:

- `running`
- `completed_successfully`
- `completed_errors`
- `failed`
- `queued`
- `aborting`
- `aborted`
- `new`
- `not_responding`
- `warning`

t|type *Type*

[Optional] The type of job. Enter one of the following:

- `BACKUP`
- `RESTORE`
- `ANALYZE_READ`
- `ANALYZE_VALIDATE`
- `DATA_MIGRATION`

Example:
`backup`

b|bucket *BucketNumber*

[Optional] Select a bucket number to display a grouping of data when too many results return to display at once. The command output notifies you if there are more buckets of data to display.

XML File Example

This command does not support an XML file.

list_query_backupids

Purpose

The `list_query_backupids` command lists the results of the query returned from the `query_backupids` command.

Syntax

list_query_backupids -n|name -v|backup_version -B|backup_repository

Example

```
dsc list_query_backupids -n MyJob -v LATEST
```

Parameters

n|name *Name*

The name of the job on which to perform the action. Must be unique for each job.

Example:
`MyJob`

v|backup_version *BackupVersion*

[Optional] Backup version number. Type LATEST or 0 for latest save set. Defaults to latest if no version is entered.

Example:
`LATEST`

B|backup_repository

[Optional] Flag to list backupid of repository backup job.

XML File Example

This command does not support an XML file.

list_recover_backup_metadata

Purpose

The `list_recover_backup_metadata` command lists the overall status and individual media server status of the `recover_backup_metadata` command.

Syntax

list_recover_backup_metadata -n|-name *targetGroupName*

Example

```
dsc list_recover_backup_metadata -n target_group1
```

Parameters

n|name *targetGroupName*

The name of the target group used for backup of the DSC repository.

Example:

```
target_group1
```

XML File Example

This command does not support an XML file.

list_repository_backup_settings

Purpose

The `list_repository_backup_settings` command lists all current repository backup settings.

Syntax

list_repository_backup_settings

Example

```
dsc list_repository_backup_settings
```

Parameters

There are no parameters associated with this command.

XML File Example

This command does not support an XML file.

list_reset_ir_target_system

Purpose

The `list_reset_ir_target_system` command lists the results of reset_ir_target_system.

Syntax

list_reset_ir_target_system -s|system_name *systemName*

Example

```
list_reset_ir_target_system -s system1
```

Parameters

-s|system_name *systemName*

Name of the target system.

Example:
`system1`

u|user_authentication *User*

Required when security management is enabled. Supplies the command with the Viewpoint user, and triggers a password prompt for authentication.

Permissions required: Administrator role

XML File Example

This command does not support an XML file.

list_save_sets

Purpose

The `list_save_sets` command lists all valid save sets for a specified job name.

Syntax

`list_save_sets -n|-name` *Name* `-C -F|-filter` *FILTER* `-v|backup_version` *BackupVersion* `-B|backup_repository`

Examples

```
dsc list_save_sets -n job1 -F LAST_WEEK
```

```
dsc list_save_sets -n targetgroup1 -F LAST_WEEK -B
```

```
dsc list_save_sets -n job2 -C
```

Parameters

n|name *Name*

The name of the job or target group for its save sets. If -B is not specified, name refers to a backup job. If -B is specified, name refers to a repository target group name.

Example:
`job1`

C|correlated_object

[Optional] Lists the correlated save sets that result when you run the skipped objects option with `run_job -I`. The save set missing the skipped objects is listed first and the save set containing the skipped objects is listed second.

The specified job must be part of a correlated pair.

Save sets are not listed if they have expired.

Example:

```
# dsc list_save_sets -n LV500001 -C
```

F|filter *FILTER*

[Optional] Filter the save sets by the stop_time. Enter one of the following (must be UPPERCASE):

- `LAST_WEEK`
- `LAST_MONTH`
- `LAST_YEAR`

Example:

```
LAST_MONTH
```

v|backup_version *BackupVersion*

[Optional] Backup version number. Enter `latest` or `0` for the latest save set.

Example:

```
latest
```

B|backup_repository

[Optional] View the backup repository for DSC repository backup save sets.

b|bucket *BucketNumber*

[Optional] Select a bucket number to display a grouping of data when too many results return to display at once. The command output notifies you if there are more buckets of data to display.

The **-b** parameter cannot be used with either the **-F** or **-v** parameters.

Usage Notes

The **-b** parameter cannot be used with either **-F** or **-v**.

If you specify the **-B** parameter and no save sets are available, the command builds a job plan from data or metadata files, if they exist.

If you run `list_save_sets -C` against a job that has no correlated pairs, the `No SaveSets found` error is generated.

XML File Example

This command does not support an XML file.

list_target_group_map

Purpose

The `list_target_group_map` command lists the maps between target groups for restoring to a different client configuration.

Syntax

list_target_group_map -s|-source *SampSrc* -d|-destination *SampDest*

Example

```
dsc list_target_group_map -s SampSrc -d SampDest
```

Parameters

The parameters are optional. However, if you specify one of the parameters, you must also specify the other parameter.

s|source *SampSrc*

The source target group represents the backup target group for mapping.

Example:
`SampSrc`

d|destination *SampDest*

The destination target group represents the restore target group for mapping.

Required if the virtual target group (parameter **V**) is specified. It indicates the physical target group to which the virtual target group is mapped.

Example:
`SampDest`

t|type *Type*

[Optional] The type of the target group map. This parameter is not used if source or destination is specified. Target group map types are DDBOOST, REMOTE_FILE, REMOTE_MEDIA, S_3, and AZURE.

V|virtual

[Optional] Indicates this is a mapping from a virtual target group to a physical target group.

XML File Example

This command does not support an XML file.

object_release

Purpose

The `object_release` command releases all objects that are currently locked by a job. It does not release objects for new, running, or queued jobs.

Syntax

object_release -n|-name *Name* -S|-skip_prompt

Example

```
dsc object_release -n job1 -S
```

Parameters

n|name *Name*

The name of the job on which to perform the action. Must be unique for each job.

Example:
`job1`

S|skip_prompt *SkipPrompt*

[Optional] Skips displaying a confirmation message before performing the command action.

u|user_authentication *User*

Required when security management is enabled. Supplies the command with the Viewpoint user, and triggers a password prompt for authentication.

Permissions required: Administrator role

Usage Notes

This command does not apply to analyze and analyze_read jobs.

XML File Example

This command does not support an XML file.

object_release_status

Purpose

Lists the status of objects requested for release by object_release.

The `object_release_status` command is applicable only if Advanced Job Status Enhancements is enabled. It displays JobName, SystemName, Start and End time for object release status, Status of object release and Command Status. If Advanced Job Status Enhancements is not supported or disabled, it throws an error message to enable it.

Syntax

object_release_status -n|-name *Name*

Example

```
dsc object_release_status -n job1
```

Parameters

n|name *Name*

The name of the job on which to perform the action. Must be unique for each job.

Example:
`job1`

Usage Notes

This command does not apply to analyze and analyze_read jobs.

XML File Example

This command does not support an XML file.

purge_jobs

Purpose

The `purge_jobs` command can be used for cleaning up DSC repository when resources are not released after aborting jobs. The command will abort any jobs and purge the resources used by any incomplete jobs. When no job name is provided, it purges resources used by all jobs in BAR and BARBACKUP.

Syntax

purge_jobs -n|-name *Name* -B|-repository_backup -S|-skip_prompt

Example

`dsc purge_jobs -B` (Cleans up the resources used by the repository backup.)

`dsc purge_jobs -n Job1` (Cleans up the resources used by a particular job.)

`dsc purge_jobs` (Cleans up the resources used by all jobs.)

Parameters

n|name *Name*

[Optional] The name of the job for which resources has to be cleaned up. Must be unique for each job.

Example:
`job1`

B|backup_repository

[Optional] Purges resources used by repository job in BARBACKUP database.

u|user_authentication *User*

Required when security management is enabled. Supplies the command with the Viewpoint user, and triggers a password prompt for authentication.

Permissions required: DSA Job Permissions: Run

S|skip_prompt *SkipPrompt*

[Optional] Skips displaying a confirmation message before performing the command action.

Usage Notes

purge_jobs command is not a replacement for the abort functionality. If a job hangs, users should abort the job first. Possible scenarios for using purge_jobs:

- Jobs are getting queued, but DSC shows no jobs are running.
- Job status in DSC shows that the job is not running, but when user tries to rerun the job, DSC shows that job is running. User is not able to abort the job either.

XML File Examples

This command does not support an XML file.

Related Information:

Job Settings Tab

query_backupids

Purpose

The `query_backupids` command queries third-party software for information needed for duplication.

Syntax

query_backupids -n|name -v|backup_version -B|backup_repository

Example

```
dsc query_backupids -n MyJob -v LATEST
```

Parameters

n|name *Name*

The name of the job on which to perform the action. Must be unique for each job.

Example:
`MyJob`

v|backup_version *BackupVersion*

[Optional] Backup version number. Type LATEST or 0 for latest save set. Defaults to latest if no version is entered.

Example:
`LATEST`

B|backup_repository

[Optional] Flag to query backupid of repository backup job.

u|user_authentication *User*

Required when security management is enabled. Supplies the command with the Viewpoint user, and triggers a password prompt for authentication.

Permissions required: User role

XML File Example

This command does not support an XML file.

recover_backup_metadata

Purpose

The `recover_backup_metadata` command queries the third party media to recover backup metadata and rebuild the backup job plan in the case of a disaster to the DSC repository. The command can only run on repository backup jobs with no save sets.

Syntax

recover_backup_metadata -n|-name *targetGroupName*

Example

```
dsc recover_backup_metadata -n target_group1
```

Parameters

n|name *targetGroupName*

The name of the target group used for backup of the DSC repository.

Example:
`target_group1`

u|user_authentication *User*

Required when security management is enabled. Supplies the command with the Viewpoint user, and triggers a password prompt for authentication.

Permissions required: Administrator

XML File Example

This command does not support an XML file.

reset_ir_target_system

Purpose

The `reset_ir_target_system` command enables incremental restore on the target system.

Syntax

reset_ir_target_system -s|system_name *systemName*

Example

```
reset_ir_target_system -s system1
```

Parameters

-s|system_name *systemName*

Name of the target system.

Example:

```
system1
```

d|dbaccount_id *database_account*

[Optional]Database account ID

Example:

```
jana234
```

l|logging_level *level*

[Optional] Error, Info, Debug, Warning

Example:

```
Info
```

t|time_out

[Optional] Time out value in minutes. Ends the current request as Failed, allowing a new request to be submitted. Default: 720

u|user_authentication *User*

Required when security management is enabled. Supplies the command with the Viewpoint user, and triggers a password prompt for authentication.

Permissions required: Administrator role

S|skip_prompt *SkipPrompt*

[Optional] Skips displaying a confirmation message before performing the command action.

XML File Example

This command does not support an XML file.

retire_job

Purpose

The `retire_job` command retires an active job. It does not retire a running or queued status job, or one that is already in the retired state.

Syntax

retire_job -n|-name *Name* -S|-skip_prompt

Example

```
dsc retire_job -n job1 -S
```

Parameters

n|name *Name*

The name of the job on which to perform the action. Must be unique for each job.

Example:
`job1`

S|skip_prompt *SkipPrompt*

[Optional] Skips displaying a confirmation message before performing the command action.

u|user_authentication *User*

Required when security management is enabled. Supplies the command with the Viewpoint user, and triggers a password prompt for authentication.

Permissions required: DSA Job Permissions: Run and Edit

Usage Notes

After a job is retired, no status information is returned for the job. The only commands you can use with a retired job are: delete_job, activate_job and list_jobs (with a retired state parameter).

XML File Example

This command does not support an XML file.

Related Information:

Job Settings Tab

run_job

Purpose

The `run_job` command runs a job as soon as all necessary resources are available. The DSC system limit is set at 20 concurrent running jobs, and up to 20 jobs can queue over that limit. The DSC also queues jobs if the defined target media is not available before the job starts.

Syntax

```
run_job -n|-name JobName -b|backup_type BackupType -p|-preview -r|runtime -f|-file
File -w|wait -q|query_status -u|user_authentication User -I|original_job_execution_ID
-j|job_phase JobPhase -R|restart
```

Examples

```
dsc run_job -n job1 -b cumulative -p -f file1.xml
```

```
dsc run_job -n job3 -b cumulative -I 13
```

```
dsc run_job -n job5 -R -w -q
```

Parameters

n|name *Name*

The name of the job on which to perform the action. Must be unique for each job.

Example:
`job1`

b|backup_type *BackupType*

Enter the type of backup: `full` , `delta` , or `cumulative`

Example:
`delta`

p|preview

[Optional] Generates an XML file that lists the job plan and settings. When the `-r` parameter is also used, a job plan is generated that includes only systems and media servers that are online. If the `-r` parameter is not used, the job plan includes all systems and media servers, even if they are not online.

r|runtime

[Optional] Checks to see if any of the media servers or systems are down, then generates a job plan to include only those online media servers or systems. To use the `r` parameter, you must also select the `-p` parameter.

For NetBackup and DDBoost, if `disable_gen_zero` is true in the target group configuration, backup jobs are rejected if any media server is offline. If `disable_gen_zero` is false, the job runs with a warning message.

f|file *filename*

[Optional] If you are previewing the job, this is the file path and file name of the output file to save the job plan.

Example:
`job1.xml`

w|wait

[Optional] Waits until the job has run, then displays a brief status, such as COMPLETED_ERRORS. You can add the `-q` parameter for a more detailed status. You cannot use the wait parameter with the preview parameter.

q|query_status

[Optional] Returns full status after the job has run. Status includes the percentage of completion and elapsed time of the job run. To use this parameter, you must also select the `-w` parameter.

R|restart

Restarts a restore job that failed, aborted, or completed with errors or warnings. Only restores the objects that were not restored.

u|user_authentication *User*

Required when security management is enabled. Supplies the command with the Viewpoint user, and triggers a password prompt for authentication.

Permissions required: DSA Job Permissions: Run

-I |job_execution_ID *jobID_of_original_full_backup_job*

> **NOTICE**
>
> The `run_job` command with the `-I` option is not intended to be mixed with delta or cumulative backup job executions. Running delta or cumulative operations after a backup that uses the `-I` option results in backups that cannot be restored or that result in an incomplete restore, including loss of data in the restored objects.

[Optional] Runs a backup or restore job with the objects that were skipped in the original full backup job. The new job contains only the skipped objects. The job execution ID must be from:

- A job that completed with errors or warnings
- A job with skipped objects
- The job specified with the -n option

The save set containing the skipped objects is not a base for delta or cumulative backup jobs, that is, the next backup for the job must be a full backup.

To obtain a job execution ID, you can use the `list_job_history` command.

Note:

If you run the backup job using the `-I` option, but the job completes with errors, you can use the original save set and run the job again with the `-I` option. The save set that results from running the job again includes the objects that were skipped in the original execution that completed with errors. The newly-generated save sets, together with the original save sets, are correlated and are required when restoring any object defined in the backup job definition.

Example:

```
# dsc run_job -n LV500001 -I 13
```

Usage Notes

If `disable_gen_zero` is true in the target group configuration, backup jobs are rejected if any media server is offline. If `disable_gen_zero` is false, the job runs with a warning message.

The `run_job` command cannot be used successfully for a retired job.

DSA allows users to run jobs when there are down AMPs in the system.

- Backup jobs are not rejected. Tables that have fallback protection are backed up as usual. Tables without fallback protection are skipped.
- Restore jobs are not rejected. Tables without fallback protection are skipped.
- CBB backup is rejected, since there is no way to keep track of WLSNs of down AMPs.

XML File Example

This command does not require an XML file as input. You must supply a file name and location to which the XML file results are exported as output.

Related Information:

Job Settings Tab

run_repository_job

Purpose

The `run_repository_job` command runs a job in the DSC repository.

DSA 17.02.00 adds replication so your repository backup goes to a landing zone and is then replicated to third party storage.

- The `run_repository_job` command can only be initiated if no operational jobs are running.
- The `-n` parameter is no longer supported.
- The `-v` parameter is only supported for backup jobs when used with the `-r` parameter. It is also supported for the new replicate job type. It is an optional parameter for restore and analyze jobs. If `-v` is not entered for a restore or analyze job, it defaults to the latest successful backup job version.

- You do not need to specify a target name for restore or analyze jobs; if you do not, the command takes the target name configured in `config_repository_backup`.

Syntax

run_repository_job -t|-type *Type*

run_repository_job -t|-type backup -r|repository_replication *rep_tg_name*

run_repository_job -v|backup_version

run_repository_job -v|backup_version -n|name

Examples

`dsc run_repository_job -t backup`

`dsc run_repository_job -t backup -r replication-tg-aws` (back up repository to landing zone and replicate to target group)

`dsc run_repository_job -t replicate-to -r replication-tg-aws -v 157838495` (replicate the particular backup saveset to the target group

`dsc run_repository_job -t replicate_from -r replication-tg-aws -v 157838495` (replicate the particular backup saveset from the target group to the landing zone)

Parameters

t|type *Type*

Enter `backup`, `replicate_to`, `replicate_from`, `restore`, `analyze_validate`, or `analyze_read`.

r|replication_target *rep_tg_name*

[Optional] Target group for replication. If not specified, backs up the repository only to landing zone.

v|backup_version *BackupVersion*

[Optional] Backup version number to be replicated to or from third party storage. This number is visible in the backup filename (`bar_123456789.out`)

If you specify `next_incremental_restore`, do not specify a backup version number.

Example:
`123456789`

u|user_authentication *User*

Required when security management is enabled. Supplies the command with the Viewpoint user, and triggers a password prompt for authentication.

Permissions required: Administrator

Usage Notes

This command automatically generates backup, restore, and analyze jobs based on the repository backup target group configuration the user created.

XML File Example

This command does not support an XML file.

Related Information:

Job Settings Tab

save_sets_replication_status

Purpose

Lists the status of repository save sets that have been replicated to third party storage.

Syntax

save_sets_replication_status -b *bucket number* -n|-name *Name*

Example

```
dsc save_sets_replication_status -b 1 -n replication-S3
```

Parameters

n|name *Name*

Name of the replication target group

replication-tg-S3

b|bucket

Number of the bucket

Usage Notes

This command does not apply to restore, analyze, and analyze_read jobs.

XML File Example

This command does not support an XML file.

set_status_rate

Purpose

The `set_status_rate` command configures the status update rate between DSC and the media servers or Teradata systems.

Syntax

set_status_rate -n|-name *Name* -t|-type *Type* -r|-rate *Rate*

Example

```
dsc set_status_rate -n mediaserver1 -t system -r 15
```

Parameters

n|name *Name*

The name of the database system or media server name.

Example:
`mediaserver1`

t|type *Type*

The type of BAR components to update. Enter `system` or `media_server`.

Example:
`SYSTEM`

r|rate *Rate*

The rate of the status refresh in seconds. Default is 30 seconds. Valid range is between 30 and 60 seconds.

Example:
`45`

u|user_authentication *User*

Required when security management is enabled. Supplies the command with the Viewpoint user, and triggers a password prompt for authentication.

Permissions required: Administrator role

XML File Example

This command does not support an XML file.

sync_save_sets

Purpose

The `sync_save_sets` command sends a sync request to all NetBackup clients that have save sets older than the dataset.retention.days value configured in `dsc.properties`. If the save sets are expired on the NetBackup side and no restore or analyze job is configured to use them, DSC deletes them from the DSC repository. If it is the last save set and a restore or analyze job is configured to use it, it will not be deleted until there is a newer save set. No scheduled or ad hoc jobs can run until the deletion completes.

The `sync_save_sets` user command is a manual alternative to using the checkretention.cronstring property in `dsc.properties` to schedule an automatic task.

Syntax

sync_save_sets -S|skip_prompt

Example

```
dsc sync_save_sets -S
```

Parameters

S|skip_prompt *SkipPrompt*

[Optional] Skips displaying a confirmation message before performing the command action.

u|user_authentication *User*

Required when security management is enabled. Supplies the command with the Viewpoint user, and triggers a password prompt for authentication.

Permissions required: Administrator role

c|current_status *CurrentStatus*

[Optional] Returns the current status of the sync save set request.

A|abort *Abort*

[Optional] Aborts the current running sync_save_sets request from further processing of the Sync Save Sets.

Usage Notes

Running this command removes information in DSC about save sets that previously existed on the backup application. This includes removing restore and analyze jobs that reference each save set. Restore and analyze jobs that point to an expired save set and are not set to the *latest* save set are deleted along with the expired save set. If the restore or analyze job points to the expired save set and is set to the *latest* save set, the restore or analyze job fails to run until the attached backup job is run again and creates a new save

set to which the restore or analyze job points. If you do not run a job for a long time and its save sets have been deleted by NetBackup, running this command could cause the job status to be changed to New.

If a save set with dependent save sets is expired, the dependent save sets are invalidated. Invalid save sets are not available for use in creating or editing jobs. Any jobs referencing invalid save sets are rejected at run time.

NOTICE

If the physical media of the backup save set becomes damaged, neither auto check retention job nor the `sync_save_sets` command should be run until the media is replaced or repaired. If the physical media cannot be repaired, use `sync_save_sets` to mark the save set so that any run depending on this save set is invalidated.

XML File Example

This command does not support an XML file.

system_health

Purpose

The `system_health` command lists ActiveMQ system health information such as the memory limit and memory usage for the main DSA queues.

Syntax

system_health

Example

```
dsc system_health
```

Parameters

There are no parameters associated with this command.

Usage Notes

The command can be used to troubleshoot messaging communication over ActiveMQ. If the memory used in a queue is close to 100% do the following:

- Edit `/opt/teradata/tdactivemq/config/td-broker.xml` to replace `deleteAllMessagesOnStartup="false"` with `deleteAllMessagesOnStartup="true"`.
- Restart the ActiveMQ broker.

XML File Example

This command does not support an XML file.

update_job

Purpose

The `update_job` command updates an existing DSA job based on the information from the command line parameters or the XML file if provided. Parameter values specified in the command line supersede values entered for the same parameters in the XML file.

You cannot use update_job to convert a standard backup or restore job to an incremental restore backup or restore job.

Syntax

update_job -n|-name *Name* [*parameters*|*File*]

Example

```
dsc update_job -n job1 -f parameters.xml
```

Parameters

n|name *Name*

The name of the job on which to perform the action. Must be unique for each job.

Example:
`job1`

d|description *Description*

A meaningful description of the job. To allow a multiword description, add \" before and after the description string.

Example:
`\"backup web apps\"`

t|type *Type*

[Optional] The type of job. Enter one of the following:

- `BACKUP`
- `RESTORE`
- `ANALYZE_READ`
- `ANALYZE_VALIDATE`

Example:
`backup`

b|backup_name *BackupName*

[Optional] An existing backup job name. For restore, analyze_read, and analyze_validate jobs only.

Example:
`backupWeb1`

v|backup_version *BackupVersion*

[Optional] Backup version number to be replicated to or from third party storage. This number is visible in the backup filename (`bar_123456789.out`)

If you specify `next_incremental_restore`, do not specify a backup version number.

Example:
`60`

f|file *filename*

The full file path and file name of the file containing the necessary parameters to create the job. If the same parameters are provided both in the file and on the command line, Teradata DSA uses the values specified in the command line.

Example:
`backupjob1.xml`

u|user_authentication *User*

Required when security management is enabled. Supplies the command with the Viewpoint user, and triggers a password prompt for authentication.

Permissions required: DSA Job Permissions: Edit

Usage Notes

You cannot update a retired job.

XML File Example

This command imports information from the XML file created from the `export_job` command.

Related Information:

Job Settings Tab

Additional Information

Changes and Additions

Date	Release	Description
March 2023	17.20.04.00	17.20.04.00 adds updates in the following: • AJSE support at Job level • Proxy Server Support for cloud platforms
November 2022	17.20.03.00	17.20.03.00 adds updates in the following: • TDActiveMQ-5.17.2 support • Teradata-jdk11 support • Teradata DSA commands • Postgres upgrade to 10.22
July 2022	17.20.02.00	17.20.02.00 adds the following: • Support Azure AD authentication for Azure Blob • Return DSA job responses within BaaS time limits • Multiple DSCs sharing same ClientHandler Config File • AJSE is disabled by default from DSA Client
March 2022	17.20.01.00	17.20.01.00 adds the following: • Support TDActiveMQ 5.16.4 to address Log4J vulnerabilities
February 2022	17.20.00.03	17.20.00.03 adds the following: • Supported releases - Veritas NetBackup • Support for SLES 15 SP2 STND • Upgrade log4j2 Version to 2.17.1 for addressing CVE-2021-44832. • Support for TDActiveMQ 5.16.1.0-6. • AWS S3 Partner Attribution. • Compatibility between TD repo and Postgres components.
October 2021	17.20	17.20 adds the following: • Support for SQLE 17.20 release. • TDActiveMQ upgrade. • DSA configuration and validation enhancements. • GCP CMEK(Customer-Managed Encryption Keys) support. • Error message MVP2. • Postgres to Postgres migration procedure. • Postgres upgrade to 10.17.
July 2021	17.10.01.00	17.10.01.00 adds the following: • Using the export_import to automate DSA job migration to a different domain.

Date	Release	Description
		• Saving the DSC repository configuration information.
March 2021	17.10	Initial release
January 2021	17.06	17.06 adds support for the following: • Configuring Google Cloud in the BAR portlets • Using multiple network fabrics
September 2020	17.05	Initial release for 17.05.

Supported Releases

This document supports the following versions of Teradata products.

- Teradata Vantage 2.2, which includes support for Analytics Database:
  - 17.05.xx
- Teradata Vantage 2.0 and Vantage 2.1, which include support for Analytics Database:
  - 17.00.xx
- Teradata Vantage 1.1, which includes support for Analytics Database:
  - 16.20 Feature Update 2
- Teradata Vantage 1.0, which includes support for Teradata Database:
  - 16.20 Feature Update 1
  - 16.20
- Teradata Database:
  - 16.10
  - 15.10
- Teradata DSA:
  - 17.10
- Teradata Viewpoint (minimum):
  - 16.50.01.00
  - 16.20.23.05
- IBM Spectrum Protect:
  - 8.1.6
- Veritas NetBackup:
  - NBU 9.1.0.1 on 64-bit SLES12 SP3 and SLES15 SP2
  - NBU 9.1.0.1 on 64-bit RHEL 7.3

- NBU 9.1 on 64-bit SLES12 SP3 and SLES15 SP2
- NBU 9.1 on 64-bit RHEL 7.3
- 8.1 on 64-bit SLES 11 SP3
- 8.1 on 64-bit RHEL 7.3
- 7.7.3 on 64-bit SLES 11 SP3
- 7.7.3 on 64-bit RHEL 7.3

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	Helpful resources in one place: • Support requests • Account management and software downloads • Knowledge base, community, and support policies • Product documentation • Learning resources, including Teradata University
https://www.teradata.com/University/Overview	Teradata education network
https://support.teradata.com/community	Link to Teradata community

Related Documentation

Title	Publication ID
Teradata® DSA Release Definition Summarizes new features and fixed issues associated with the release.	B035-3154
Teradata® DSA User Guide Describes how to use the Teradata Data Stream Architecture (DSA) portlets and command-line interface.	B035-3150
Teradata® DSA - DSE for Veritas NetBackup Installation, Configuration, and Upgrade Guide Describes how to configure Data Stream Extensions software and devices.	B035-3151
Teradata® DSA - DSU Installation, Configuration, and Upgrade Guide Describes how to configure Data Stream Utility software and devices.	B035-3153
Teradata® DSA - DSE for IBM Spectrum Protect Installation, Configuration, and Upgrade Guide Describes how to configure Data Stream Extensions software and devices.	B035-3155

Title	Publication ID
Teradata® DSA Quick Start Guide Guides you through a simple set up to verify connections.	B035-3156
Teradata® Viewpoint User Guide Describes the Teradata Viewpoint portal, portlets, and system administration features.	B035-2206
Teradata® Viewpoint Installation, Configuration, and Upgrade Guide for Customers Describes how to install Viewpoint software, configure settings, and upgrade a Teradata Viewpoint server.	B035-2207
Parallel Upgrade Tool (PUT) Reference Describes how to install application software using PUT.	B035-5716
Teradata Vantage™ - Database Administration Describes how to administer the Analytics Database.	B035-1093
Teradata® VantageCore VMware - Base, Advanced, Enterprise Tiers Getting Started Guide	B035-5958
Teradata® VantageCloud Enterprise on AWS (DIY) Installation and Administration Guide Describes how to deploy and configure Teradata software components to run in the AWS public cloud.	B035-2800
Teradata Vantage™ on Azure (DIY) Installation and Administration Guide Describes how to deploy and configure Teradata software components to run in the Azure public cloud.	B035-2810