Quest® NetVault® Backup Plug-in *for PostgreSQL* 12.3.1

User’s Guide

If you have any questions regarding your potential use of this material, contact:

Quest Software Inc.
Attn: LEGAL Dept.
4 Polaris Way
Aliso Viejo, CA 92656

Refer to our website (https://www.quest.com) for regional and international office information.

Patents

Quest Software is proud of our advanced technology. Patents and pending patents may apply to this product. For the most current information about applicable patents for this product, please visit our website at https://www.quest.com/legal.

Trademarks

Legend

■ **WARNING: A WARNING icon indicates a potential for property damage, personal injury, or death.**

! **CAUTION: A CAUTION icon indicates potential damage to hardware or loss of data if instructions are not followed.**

i **IMPORTANT NOTE**, **NOTE**, **TIP**, **MOBILE**, or **VIDEO:** An information icon indicates supporting information.

NetVault Backup Plug-in *for PostgreSQL* User's Guide
Updated - August 2021
Software Version - 12.3.1
PSG-101-12.3.1-EN-02

Contents

1

Introducing NetVault Backup Plug-in *for PostgreSQL*

NetVault Backup Plug-in *for PostgreSQL*: at a glance

Quest® NetVault® Backup Plug-in *for PostgreSQL* (Plug-in *for PostgreSQL*) increases confidence in recoverability of critical data and eliminates the need for writing complex backup-and-recovery scripts. Through a web-based user interface (WebUI) and automated workflow process, the plug-in offers a centralized way to set up, configure, and define backup and restore policies for your PostgreSQL databases.

Support for database-cluster backups and individual database and table backups in different backup formats lets you implement a backup policy that is flexible enough to account for many recovery scenarios without learning PostgreSQL internals. The plug-in offers a detailed level of control that minimizes downtime by allowing you to restore entire database clusters, individual databases, or individual tables reliably. Through integration with a range of backup devices, your data is protected and stored offsite to meet your disaster-recovery and business-continuity goals.

Key benefits

- **Increase confidence when deploying the plug-in**: Plug-in *for PostgreSQL* enables total protection of the PostgreSQL databases without requiring you to write complex backup scripts. The plug-in lets you back up the entire database cluster, individual databases, or tables in any of the supported backup formats. You can then use those backups to reconstruct entire or parts of database in the unfortunate event of data loss.
- **Speed up restores and reduce downtime**: With Plug-in *for PostgreSQL*, database administrators (DBAs) are no longer required to write scripts or manually run restore commands to recover lost data. Select what must be restored from a saved backup set, and the plug-in recovers the data without further interaction. The restores are faster due to reduced human interaction, and the chance for a syntax error in a manual execution is eliminated.
- **Ensures business continuity**: With offsite backups being an important part of the data-protection plan for business-critical applications, the plug-in takes advantage of NetVault Backup's integration with a range of backup devices. NetVault Backup lets you select which backup device to store the backup on. You can store the backup online in a virtual tape library (VTL). You can also duplicate the job to physical tape libraries, ensuring that your PostgreSQL environment is protected for disaster-recovery purposes.

Feature summary

- **Backup features**
  - Full database cluster or individual database or table; SQL logical backups
  - File-based baseline data Incremental Backups
  - Plain-text SQL Script, Tar Archive, and Custom Archive (Linux only) backup formats
  - Protection down to the table level
  - Global Objects-only backups
  - Template database backup
  - Data-only backups
  - Configuration files backup
- **Restore features**
  - Restore entire cluster, individual databases, or tables
  - If you are using PostgreSQL Server 9.6 or later, restore a whole PostgreSQL database cluster to a specific point in time (point-in-time recovery [PITR])
  - Rename databases during restore
  - Restore to alternate PostgreSQL database cluster
- **Other features**
  - Point-and-click WebUI
  - Integration with storage devices
  - Job scheduling

Target audience

While advanced DBA skills are not required to create and run routine backup operations, they are required for defining an efficient backup-and-recovery strategy.

Recommended additional reading

Quest recommends that you have the following documentation available for reference while setting up and using this plug-in.

- **PostgreSQL documentation**, which is available from http://www.postgresql.org/docs/.
- **NetVault Backup documentation:**
  - *Quest NetVault Backup Installation Guide*: This guide provides details on installing the NetVault Backup Server and Client software.
  - *Quest NetVault Backup Administrator's Guide*: This guide explains how to use NetVault Backup and describes the functionality common to all plug-ins.
  - *Quest NetVault Backup CLI Reference Guide*: This guide provides a description of the command-line utilities.

  You can download these guides from https://support.quest.com/technical-documents.

Installing and removing the plug-in

Installation prerequisites

Before installing Plug-in *for PostgreSQL*, verify that the following software is installed and properly configured on the machine designated as the PostgreSQL Server:

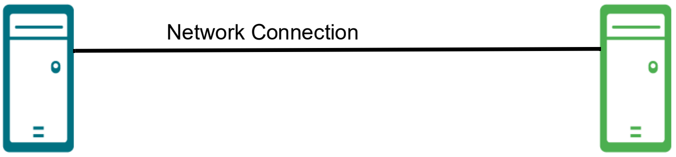
- **NetVault Backup Server or Client software:** At least the Client version of NetVault Backup must be installed on the machine configured as the PostgreSQL Server. For more information on supported versions of the **NetVault Backup** software, see the *Quest NetVault Backup Compatibility Guide*.
- **PostgreSQL software:** The machine must be running a supported version PostgreSQL. For more information, see the *Quest NetVault Backup Compatibility Guide*.

  The **Custom Archive** (Linux only) backup format requires the ***zlib compression library***. This library is included by default during PostgreSQL installation unless you specify the **--without-zlib** option. Do *not* use the **--without-zlib** option if you intend to use **Custom Archive** backup format.

Reviewing the recommended configuration

While you can set up a single machine as both the NetVault Backup Server and the PostgreSQL Server, that is, all software installation and configuration requirements are performed on a single machine, Quest recommends that these two entities exist on *separate* machines.

Table 1. Recommended configuration



PostgreSQL Server machine	**NetVault Backup Server machine**
Installed software and configuration	**Installed software and configuration**
• **PostgreSQL software**, version 9.4 or later) • **NetVault Backup Server or Client software** • **NetVault Backup Plug-in *for PostgreSQL***	• **NetVault Backup Server software** • **PostgreSQL Server added as a NetVault Backup Client**—for complete details on adding a Client to the NetVault Backup Server, see the *Quest NetVault Backup Administrator's Guide*.

> **IMPORTANT:** Regardless of the configuration that you use, that is, separate NetVault Backup Server and PostgreSQL Server, vs. a single machine configured as both, the plug-in must be installed on the host on which PostgreSQL resides.
>
> Sample procedures throughout this guide assume that you are using this two-machine environment and that configuration requirements have been met.

Installing or upgrading the plug-in

1 Access the **NetVault Configuration Wizard** or **Manage Clients** page.

> **NOTE:** If the selected clients are all the same type, you can use the configuration wizard to install the plug-in on multiple clients at the same time. When you select multiple clients, verify that the plug-in binary file is compatible with the OS and platforms of the target clients. From the **Manage Clients** page, you can only select one client for plug-in installation.

- To access the **NetVault Configuration Wizard** page:

  a In the Navigation pane, click **Guided Configuration**.

  b On the **NetVault Configuration Wizard** page, click **Install Plugins**.

  c On the next page, select the applicable clients.

- To access the **Manage Clients** page:

  a In the Navigation pane, click **Manage Clients**.

  b On the **Manage Clients** page, select the applicable machine, and click **Manage**.

  c On the **View Client** page, click the **Install Plugin** button (⊕).

2 Click **Choose Plug-in File**, navigate to the location of the **".npk"** installation file for the plug-in, for example, on the installation CD or the directory to which the file was downloaded from the website.

Based on the operating system (OS) in use, the path for this software may vary on the installation CD.

3. Select the file entitled **“pos-x-x-x-x-x.npk,”** where **xxxxx** represents the version number and platform, and click **Open**.
4. To begin installation, click **Install Plugin**.

   After the plug-in is successfully installed, a message is displayed.

Removing the plug-in

1. In the Navigation pane, click **Manage Clients**.
2. On the **Manage Clients** page, select the applicable client, and click **Manage**.
3. In the **Installed Software** table on the **View Client** page, select **Plug-in *for PostgreSQL***, and click the **Remove Plugin** button (⊖).
4. In the **Confirm** dialog box, click **Remove**.

3

Configuring the plug-in

Verifying installation of libpq library

The **libpq** library included in PostgreSQL contains a set of library functions that lets client programs submit queries to the PostgreSQL Server and receive the query results. Because the plug-in acts as a client and uses the **libpq** library, ensure that your PostgreSQL installation contains the **libpq** library, whether it installed on Linux, UNIX, or Windows. If the library is not installed, you might encounter a “Required Library Not Found” message while using the plug-in. For more information, see your PostgreSQL documentation.

Preparing the PostgreSQL Server to support continuous archiving and PITR

If you are using PostgreSQL Server 9.6 or later and you want to use the plug-in to perform Incremental Backups, also known as continuous archiving, and support point-in-time recovery (PITR), complete the following steps for your environment:

1. Create a directory that the plug-in can use to stage archived PostgreSQL write-ahead logging (WAL) files.

   Ensure that this directory is dedicated only to use by the plug-in as the data-protection process might involve deletion of this directory at some point.

   > **IMPORTANT:** If you have multiple database clusters that you are protecting using the plug-in, create a separate staging directory for each cluster. Quest recommends that you include the name of the cluster in the staging directory to ensure that you can easily identify each directory.

   Example: C:\NetVault\PosgreSQL\Database1\WALfiles

2. Make sure that the PostgreSQL user for your environment has full access to the complete path for *each* staging directory, and make sure that the same PostgreSQL user has permission to access the NetVault Backup **config**, **tmp** and **trace-container** directories.

   > **IMPORTANT:** If the PostgreSQL user does not have access, the WAL files are not archived to the staging directory.
   >
   > In Windows, the PostgreSQL user might run under the predefined **Network Service** account.

3. Enable archiving in the PostgreSQL Server.

a Open the **“postgresql.conf”** file, which is located in the database cluster **Data** directory, so that you can update it.

b Set **archive_mode** to **on**.

c Enter an **archive_command** that copies the WAL file from the **Database Cluster pg_wal** directory to the staging directory.

This command runs whenever a WAL file is ready for archiving. You can use the wildcards **%f**, which represents the filename, and **%p**, which represents the full path. For more information, see Additional information for completing the archive_command segment.

Example:

```
archive_mode = on # enables archiving; off, on, or always
# (change requires restart)
archive_command = '"C:\\Program Files\\Quest\\NetVault Backup\\
   bin\\nvpgwalutil.exe" "JobType=store" "DatabaseCluster=PostgreSQL 11"
   WALFilePath="%p" WALFileName="%f" '
```

4 After archiving is enabled, restart the PostgreSQL Server to force the changes to take effect.

Additional information for completing the archive_command segment

The following examples show additional ways that you might configure **archive_command**.

- **Windows example:**

```
archive_command = 'copy "%p" "C:\\NvbuPostgreSQL\\Test Wal\\PostgreSQL
   11\\%f"' # command to use to archive a logfile segment
```

- **Linux and UNIX example:**

```
archive_command = 'test ! -f /nvbupostgresql/walstage/postgresql11/%f
   && cp %p /nvbupostgresql/walstage/postgresql11/%f'  # Unix
```

The plug-in also includes utilities that you can use for the archive command. The applicable utility, **vpgwalutil** for Linux and UNIX and **nvpgwalutil.exe** for Windows, is located in the NetVault Backup installation **bin** directory. If you use a utility, use the following format:

```
archive_command = '"C:\\Program Files\\Quest\\NetVault Backup\\bin\\nvpgwalutil.exe"
"JobType=store" "DatabaseCluster=PostgreSQL 11" WALFilePath="%p" WALFileName="%f" '
```

If you use one of the utilities, enter the following parameters:

- **JobType=store**, instructs the utility to copy the WAL Log into the stage directory
- **DatabaseCluster=<*name*>** indicates the name of the database cluster that you intend to use when you add the cluster in the plug-in.
- **WALFilePath="%p"**
- **WALFileName="%f"**

Adding a database cluster

To begin using Plug-in *for PostgreSQL*, first add the target database cluster to the plug-in. If you are running more than one database cluster, follow this procedure for *each* database cluster to be protected using the plug-in.

1 In the Navigation pane of the NetVault Backup WebUI on the NetVault Backup Server, click **Create Backup Job**, and click **Create New** next to the **Selections** list.

2 In the selection tree, open the applicable client node.

3 Click **Plug-in *for PostgreSQL***, and select **Add Database Cluster** from the context menu.

4 In the **Add Database Cluster** dialog box, set the following parameters.

! **CAUTION: If the Add Database Cluster dialog box does not display a scroll bar on the right side, use the Tab key to keep moving through the options that might not be visible.**

- **Database Cluster Name**: Enter a name for the database cluster; by default, the local host name is used. This value is used in the NetVault Backup WebUI. Quest strongly recommends that you use a generic name for the PostgreSQL cluster instead of using a name associated with the machine on which the cluster resides. This generic name improves portability and policy management across all affected clients. When adding more than one cluster, ensure that a unique name is used for identification.

  For example, if the machine name is **test_postgresql_01_machine**, and the suggested name for the PostgreSQL cluster is the same, change the name for the instance to something such as **local_postgresql_database_cluster**. You can then use the same name, **local_postgresql_database_cluster**, when you configure the other PostgreSQL clients. Therefore, the cluster name on all clients is **local_postgresql_database_cluster**.

- **Port**: This field is preconfigured to port 5432, which is the default listener port for PostgreSQL. Enter the correct port number in this field, if PostgreSQL is configured to listen on any other port. When adding multiple database clusters, ensure that a different listener port is configured for each cluster.

- **Database SuperUser**: This field defaults to the superuser ***postgres***. If applicable, enter a different user with superuser privileges for the plug-in's use.

- **Password**: Enter the password associated with the user specified in the preceding field. Ensure that the password is correct. If the cluster is configured to use TRUST authentication for the HOST connection while using PASSWORD authentication for the LOCAL connection, the backups fail if the password is incorrect or blank. To avoid this issue, either use TRUST authentication for both the HOST and LOCAL connections, in which case no password is required, or use PASSWORD authentication for both items in the **"pg_hba.conf"** file, and provide the password here.

  > i **NOTE:** If the **"pg_hba.conf**" file contains entries for both IPv4 and IPv6, configure the same authentication for each. For example, use TRUST authentication for both IPv4 and IPv6 connections.
  >
  > You can use MD5 hash passwords.

- **PostgreSQL Bin Directory**: Enter the complete path to the PostgreSQL bin directory. For example, this path is **"C:\Program Files\PostgreSQL\10.0\bin"** for a default installation of PostgreSQL version 10.0 on the Windows platform.

- **PostgreSQL Library Directory or Path** (Linux/UNIX-based systems only): Enter the complete path to the directory that contains the **"libpq.so"** file or to the file itself, for example, **"/usr/lib/libpq.so.5.0"**.

- **PostgreSQL Data Directory**: Specify the complete path to the directory that contains the configuration files "postgresql.conf," "pg_hba.conf," and "pg_ident.conf." By default, PostgreSQL stores the configuration files in the **data** directory. For example, this path is **"C:\Program Files\PostgreSQL\10.0\data"** for a default installation of PostgreSQL version 10.0 on the Windows platform.

- **Database Used for Initial Connection**: For the plug-in's initial connection, the ***postgres*** database is used by default. To use a different database for connecting to the server, specify the database name in this field.

- **Template Database Used During CREATE Database**: The template specified in this field is used for creating an empty database during the restore process. Quest recommends that you use the default **template0**. Backups are performed relative to **template0**, which means that any languages and procedures added to **template1** are included in the backup. As a result, if you are using customized **template1**, you must create the database from an empty template such as **template0**. Otherwise, errors due to duplicate definitions might occur during the restore process.

- **Default Character Set**: From the list, select the Character Set Encoding that the plug-in uses to display the names of databases and tables in the set locale. The following table lists the supported encodings. If an incompatible encoding is configured, you cannot browse the tree, or perform backups or restores.

Character set encoding	Language
BIG5	Traditional Chinese
EUC_CN	Simplified Chinese
EUC_JP	Japanese
EUC_KR	Korean
EUC_TW	Traditional Chinese
GB18030	Simplified Chinese
LATIN1	• English • French • German
SQL_ASCII	Any
SJIS	Japanese
UTF8	Any

> **IMPORTANT:** The **UTF8** and **SQL_ASCII** encodings can be used with any language, however, the support is limited to ASCII character set. If the database or table names contain non-ASCII characters, set a locale-specific encoding, for example, EUC_JP for Japanese. For more information, see http://www.postgresql.org/docs/9.2/static/multibyte.html.

5 If you are using PostgreSQL Server 9.6 or later and you want to use the **File Level Data Copy** option for archive-based backups and restores, complete the following fields:

- **PostgreSQL Directory for Current Log Files:** The plug-in completes this field using the name of the directory used by your PosgreSQL Server. The default is **pg_wal**.

  > **NOTE:** PostgreSQL maintains a WAL file in the **pg_wal** subdirectory of the cluster's data directory For PostgreSQL 9.x and earlier, the WAL file is located in the **pg_xlog** subdirectory of the data directory.

- **Temporary Directory to Stage Log Files:** Enter the full path to the dedicated directory that you created previously.

6 To save the settings, click **Ok**.

The plug-in verifies the values specified and adds the database cluster if all parameters are properly configured.

An error message is displayed if an invalid entry is found. You can choose to ignore this message and click **Save** to add the database. The database cluster is added, however, note the following:

- If the superuser name or password *cannot* be validated, further access is denied.
- If the **PostgreSQL Data Directory** is invalid, while all other parameters are correctly configured, the backups of databases are completed but the backups of configuration files fail.
- If the **PostgreSQL Bin Directory** is invalid, while all other parameters are correctly configured, you are able to submit a job but the job fails.

If an error occurs while adding a database cluster and you must reconfigure the parameters, see Correcting invalid parameters.

Correcting invalid parameters

If a parameter is not correctly configured, an error message is displayed. You can ignore this message and save the information. However, further access is denied or backup jobs might not complete successfully until you correct the error.

1 In the error message that is displayed, click **Re-configure**.

2 In the **Add Database Cluster** dialog box, enter the correct values for the parameters reported as invalid.

> i **IMPORTANT:** The password field appears blank. If the previously configured password is correct, you do not have to re-enter the password; the plug-in uses the cached (old) password for authentication when this field is blank. Otherwise, enter the correct password in this field.

3 To reconfigure the parameters, click **Ok**.

Reconfiguring an added database cluster

1 In the Navigation Pane, click **Create Backup Job**, and click **Create New** next to the **Selections** list.

2 In the selection tree, open the applicable client node.

3 Open the **Plug-in *for PostgreSQL*** node.

4 Click the applicable database-cluster node, and select **Configure** from the context menu.

The **Configure** dialog box is displayed with all previous settings revealed, allowing you to make any necessary modifications.

5 On the **Configure Database Cluster** dialog box, update the parameters as required.

For more information, see Adding a database cluster.

Note the following:

- You cannot change the **Database Cluster Name**.
- The password field appears blank. If the previously configured password is still valid, you do not have to re-enter the password; the plug-in uses the cached (old) password for authentication when this field is blank. Otherwise, enter the correct password in this field.

6 To save the settings, click **Ok**.

> i **IMPORTANT:** If the password for the configured superuser is changed in the PostgreSQL database, update the password in the plug-in.

Removing an added database cluster

1 In the Navigation Pane, click **Create Backup Job**, and click **Create New** next to the **Selections** list.

2 In the selection tree, open the applicable client node.

3 Open the **Plug-in *for PostgreSQL*** node.

4 Click the applicable database-cluster node, and select **Remove** from the context menu.

5 In the confirmation dialog box that is displayed, click **OK**.

> i **IMPORTANT:** The Plug-in *for PostgreSQL* node will remain in an Open state even after the last database cluster is removed. To close this node and access the other plug-ins in the selection tree, double-click the plug-in node.

Configuring default settings

The plug-in lets you set default options for backup and restore jobs.

To configure default settings

1 In the Navigation Pane, click **Create Backup Job**.

2 Next to the Selections list, click **Create New**.

3 In the selection tree, open the applicable client node and click **Plug-in for PostgreSQL**.

4 From the context menu, select **Configure**.

5 Set the following default options:

- **Actions for error conditions**. See Setting default actions for error conditions (optional).
- **Allow File Level Data Copy Recovery to Modify postgresql.conf file**. Selecting this option gives NetVault Backup permission to automatically copy the contents of the **“recovery.signal”** file into the **“postgresql.conf”** file before it begins the recovery. This option applies to Plug-in *for PostgreSQL* File Level Data Copy restore and recovery jobs. By default, the option is not selected. To automate File Level Data Copy recovery, select this option. If not selected, then during a recovery, you must manually add the contents of the **“recovery.signal”** file to the PostgreSQL **“postgresql.conf”** file.

Setting default actions for error conditions (optional)

The plug-in lets you set default options for backup and restore jobs. You can override these options on a per job basis.

To set default actions for error conditions

1 In the Navigation pane, click **Change Settings**.

2 On the **Configuration** page, click **Server Settings** or **Client Settings**, as applicable.

3 If you selected **Client Settings**, select the applicable client, and click **Next**.

4 On the **NetVault Server Settings** or **Client Settings** page, click **Plugin Options**.

In the **Plug-in *for PostgreSQL*** section, the following items are listed:

- **Manually Selected Database Unavailable**: This issue occurs when an individual database that was manually selected, that is, explicitly clicked, for inclusion in the backup is unavailable for backup. The database might be unavailable for any reason, such as being dropped since the backup job was defined.
- **Manually Selected Table Unavailable**: This issue occurs when an individual table that was manually selected, that is, explicitly clicked, for inclusion in the backup is unavailable for backup. The table might be unavailable for any reason, such as being dropped since the backup job was defined.
- **Failed to Backup Configuration Files**: This issue occurs when a configuration file is not found or cannot be backed up for any other reason. The **“pg_hba.conf”** file is used for user authentication; if this file is missing during job execution, the backup fails because of login failure, regardless of the action set for the **Failed to Backup Configuration Files** error condition.
- **Allow File Level Data Copy Recovery to Modify postgresql.conf file.** Selecting this option gives NetVault Backup permission to automatically copy the contents of **“recovery.signal”** into the **“postgresql.conf”** file before it begins the recovery.

5 For each of these conditions, select one of the following settings:

- **Complete with Warnings — Saveset Retained:** The job returns a status of **“Backup Completed with warnings”** and a backup saveset is created that includes the items that were successfully backed up.
- **Complete without Warnings — Saveset Retained:** The job completes and returns a status of **“Backup Completed.”** The errors are logged in the NetVault Backup binary logs and ignored on the **Job Status** page. A backup saveset is created that includes the items that were backed up.
- **Fail — Saveset Retained:** The job returns a status of **“Backup Failed.”** However, a backup saveset is generated that includes the items that were successfully backed up.
- **Fail — No Saveset Retained:** The job returns a status of **“Backup Failed”** and no saveset of backed-up objects is kept. That is, even if some of the objects were successfully backed up, the saveset is discarded.

6 To save the settings, click **Apply**.

4

Backing up data

- Defining a backup strategy
- Performing a backup
- Backing up Global Objects only
- Backing up configuration files

ℹ **IMPORTANT:** On Windows, use appropriate encoding during database cluster initialization; that is, when you run **initdb**. Backups fail if you use **UTF-8** encoding and the database or table names contain non-ASCII characters.

Defining a backup strategy

Before commencing with database backups, ensure that you have a backup strategy that safeguards data against media failure, data corruption, user error, and loss of the database server. The following topics provide information that helps you devise a backup strategy for use with Plug-in *for PostgreSQL*.

- Reviewing the available backup methods
- Supported backup formats for the SQL Logical Data Dump backup method
- Available backup types for the SQL Logical Data Dump backup method
- Backup strategy considerations
- Examples of backup sequences

Reviewing the available backup methods

The plug-in offers the following backup methods:

- **SQL Logical Data Dump:** Selecting this method performs logical data dumps composed of SQL commands that are used to reconstruct the PostgreSQL data. The PostgreSQL Server utilities **pg_dump** and **pg_dumpall** are used to perform SQL Logical Data Dump Backups. This option supports database-cluster backups and individual database and table backups in different backup formats.
- **File Level Data Copy:** If you are using PostgreSQL Server 9.6 or later, this option supports archive-based backups and restores of the complete PostgreSQL database cluster. This option performs a physical backup of the PostgreSQL Server data files and works in combination with the **pg_start_backup** and **pg_stop_backup** utilities.

Supported backup formats for the SQL Logical Data Dump backup method

The plug-in supports backups in **Plain-text SQL Script File**, **TAR Archive**, and **Custom Archive** (Linux only) backup formats, which can be used to back up an entire database cluster or individual databases and tables.

- **Plain-text SQL Script File:** This format generates a plain-text file that contains the SQL commands required to reconstruct the database to the state it was in at the time of backup.
- **TAR Archive File**: This format generates a tape archive (TAR) file that is not compressed.
- **Custom Archive File (Linux only):** This format generates an archive file that is compressed by default. It lets you select the compression level and can be used for tables that are larger than the maximum file size supported by the OS.

  The custom archive format requires the **zlib compression library** and can be used only on PostgreSQL installations built with this library. This library is included by default during PostgreSQL installation unless you use the **--without-zlib** option to disable support for **Custom Archive**.

While similar backup options are available with all three formats, there are minor differences. All three formats let you restore only what is needed and restore data to a different cluster or server.

Available backup types for the SQL Logical Data Dump backup method

The plug-in provides the following backup types:

- **Full Database Cluster**: This option backs up all the databases within the cluster in the selected backup format. It also includes a backup of Global Objects, which are in the form of a Plain-text Script file, regardless of the backup format specified. The plug-in automatically includes the configuration files when the database cluster is selected; however, you can exclude it from the backup if applicable.
- **Individual Database/Table**: This option backs up selected databases or tables in the selected backup format. You can use it to back up databases that are updated more frequently than others or use it as a source when creating test environments. This option does *not* include Global Objects when the databases or tables are backed up individually.
- **Global Objects Only**: This option backs up only the Global Objects, which you can use to back up the updates made to the database users, groups, and access permissions.
- **Configuration Files Backup**: You can back up the configuration files with any of the preceding backup types, or separately. The following files are included when you select this option:
  - **postgresql.conf**: This file is the primary server configuration file that contains all the parameter settings and directives.
  - **pg_hba.conf**: This file is the host-based authentication (HBA) or client authentication file.
  - **pg_ident.conf**: This file stores ident maps, excluding **sameuser**, and is used for ident-based authentication.

  This backup type is similar to a file-system backup. You can use it to restore corrupt or lost configuration files, or to restore the cluster settings to a different server.

Backup strategy considerations

When defining the backup strategy, consider the following:

- Database size.
- Frequency of updates and changes.
- Off-peak period for full backups.
- Whether some databases are updated more frequently than others.
- Whether changes are confined to a small or large number of tables in a database.
- Available storage space and future expansion.

Examples of backup sequences

Following are a few sequences that might meet your PostgreSQL database protection requirements.

- **Full Database Cluster backups only**: If update characteristics are similar across multiple databases, **Full Database Cluster** backups scheduled every night ensures data protection up to the previous day.

  A **Full Database Cluster** backup consumes large amounts of storage space. However, restores are easier and quicker as only a single job is required to restore the entire cluster. Using the **Custom Archive** backup format can reduce the backup size.

- **Full Database Cluster backups and Individual Database/Table backups**: If some databases are updated more frequently than others, regular **Individual Database/Table** backup coupled with periodic **Full Database Cluster** backup is an ideal strategy. For example, **Full Database Cluster** backups can be performed every Sunday night at 11:00 p.m., while **Individual Database** backups for the databases that are updated frequently can be performed Monday through Saturday at 11:00 p.m. This ensures data protection up to the previous day. You can increase backup frequency to reduce data loss, and use the **Custom Archive** (Linux only) format with a higher compression level for large databases to reduce the backup size.

  Databases that are updated less frequently can be backed up two or three times a week or on demand after an update is made.

  During restores, the databases that have not been updated since the last backup can be restored from the **Full Database Cluster**, while the others can be restored from their last individual backup. Even though the backups are quicker, the restores can take longer due to the additional intervention that is required to run multiple restore jobs.

- **Global Objects Only backup**: Because the individual database backups exclude the global objects, you can use **Global Objects Only** after updates are made to the database users, groups, and access permissions. This option ensures that these recent backups can be used instead of **Full Database Cluster** backups.

- **Individual Database/Table backups and Global Objects Only backups**: You could use a backup strategy of **Individual Database/Table** backups plus **Global Objects Only** backup where you back up the individual databases according to their update frequency. You might back up databases that updated less frequently on a weekly basis and back up the remaining databases daily. Also, you might run **Global Objects Only** backups weekly or on demand after an update to the database groups, users, and access permissions is made.

Performing a backup

A backup using Plug-in *for PostgreSQL* includes the steps outlined in the following topics.

- Selecting data for a backup
- Setting backup options
- Finalizing and submitting the job

Selecting data for a backup

You must use sets—Backup Selection Set, Backup Options Set, Schedule Set, Target Set, and Advanced Options Set—to create a backup job. For more information, see the *Quest NetVault Backup Administrator's Guide*.

> **TIP:** To use an existing set, click **Create Backup Job**, and select the set from the **Selections** list.

1 In the Navigation pane, click **Create Backup Job**.

   You can also start the wizard from the Guided Configuration link. In the Navigation pane, click **Guided Configuration**. On the **NetVault Configuration Wizard** page, click **Create backup jobs**.

2 In **Job Name**, specify a name for the job.

   Assign a descriptive name that lets you easily identify the job when monitoring its progress or restoring data. The job name can contain alphanumeric and nonalphanumeric characters, but it cannot contain non-Latin characters. On Linux, the name can have a maximum of 200 characters. On Windows, there is no length restriction. However, a maximum of 40 characters is recommended on all platforms.

3 Next to the **Selections** list, click **Create New**.

4 In the list of plug-ins, open **Plug-in *for PostgreSQL***.

   The plug-in attempts to connect to the database server with the name and password specified in the **Configure** dialog box. After successful authentication, the added database clusters are displayed.

5 Open the target database-cluster node, and you see two nodes: **Configuration Files** and **All Databases**.

   - To make **Full Database Cluster** backups, do one of the following:
     - To back up all the databases, including Template Databases, *and* the configuration files contained in the cluster, select the database-cluster node.
     - To back up *only* the databases, including Template Databases, select the **All Databases** node, or clear the **Configuration Files** node after selecting the database-cluster node.
   - To select individual databases or tables, do the following:
     - Open the **All Databases** node, and select the required databases. To include Template Databases in the backup, select the node, for example, template1.
     - For individual table selection, open the applicable database node. Select the database node, which automatically includes all the tables, and then clear the check boxes for the tables that you want to exclude from the backup.

   The Template Database nodes cannot be opened further.

6 Click **Save**, enter a name in the **Create New Set** dialog box, and click **Save**.

   The name can contain alphanumeric and nonalphanumeric characters, but it cannot contain non-Latin characters. On Linux, the name can have a maximum of 200 characters. On Windows, there is no length restriction. However, a maximum of 40 characters is recommended on all platforms.

Setting backup options

The next step involves creating the Backup Options Set or selecting an existing one. The backup options that are available depend on the backup method that you select.

> ℹ **TIP:** To use an existing set, in the **Plugin Options** list, select the set that you want to use.

- Setting backup options for a SQL Logical Data Dump
- Setting backup options for a File Level Data Copy

Setting backup options for a SQL Logical Data Dump

1. Next to the **Plugin Options** list, click **Create New**.
2. In the **Backup Method** section on the **PostgreSQL Backup Options** tab, select **SQL Logical Data Dump**.
3. In the **Backup Type** section, select the applicable option:
   - **Full Database Cluster**
   - **Individual Database/Table**

   For more information, see Available backup types for the SQL Logical Data Dump backup method.
4. In the **Backup Format** section, select the applicable option:
   - **Plain-Text SQL Script File**
   - **TAR Archive File (Uncompressed)**
   - **Custom Archive File (Compressed)**

     > ℹ **NOTE:** Plug-in *for PostgreSQL* does not support Custom Archive File backup format on Windows operating systems.

   For more information, see Supported backup formats for the SQL Logical Data Dump backup method.

   > ℹ **IMPORTANT: NetVault Backup limitation on Windows**: NetVault Backup resets the selection in **Backup Format** to the default (**Plain Text SQL Script**) if you change the **Backup Type** *after* selecting the file format. To avoid this issue, select the applicable **Backup Format** explicitly after selecting the **Backup Type**.
   >
   > A similar problem also occurs when you save a **Backup Options** set. NetVault Backup resets the **Backup Format** to the default after you click **OK**. However, the saved **Backup Options** set contains your actual selection. Similarly, when you try to load a **Backup Options** set, the **Plain Text SQL Script** format is displayed as selected, which is incorrect. Internally, the plug-in uses the actual selected **Backup Format**.
5. If you selected a **Backup Format** of **Plain-Text SQL Script File**, select the applicable **Backup Options**:
   - **Include Data Only (No Schema Definitions)**: To back up only the data without the schema, select this option. Data-only restores can only be performed for backups performed with this option selected.

     > ℹ **IMPORTANT:** Template Databases are *not* included if you select this option.
   - **Disable Triggers**: This option is only relevant when creating a *data-only* dump. Select this option if there are referential integrity checks or other triggers on the tables. This option temporarily disables triggers on the target tables during restore. After a successful restore, the triggers are enabled.
   - **Include BLOBs**: To back up binary large objects (BLOBs), select this option. When this option is selected, all BLOBs within the selected database are backed up *regardless of whether the associated table is included*.

- **Include Object Identifiers**: This option dumps object identifiers (OIDs) as part of the data for every table. Use this option if the OID columns are referenced, for example, in a foreign key constraint. Otherwise, do not select this option.
- **Include Drop Object Commands**: To drop and re-create objects during the restore process, select this option.
- **Exclude Access Privileges**: To prevent dumping of access privileges, grant and revoke commands, select this option.
- **Disable $ Quoting for Function Bodies**: To disable the use of dollar quoting for function bodies and force them to be quoted using SQL-standard string syntax, select this option.
- **Dump Data as Copy Data:** To dump data using COPY commands instead of INSERT commands during a restore, select this option. Dumping data as INSERT commands generates a separate command for each row of data. This method is safer because any errors that occur result only in the loss of a row of data instead of losing the whole table. Dumping data as COPY commands might result in the loss of the whole table if an error occurs in one of the rows of data.

6 If you selected a **Backup Format** of **TAR Archive File**, select the applicable **Backup Options**:

- **Include BLOBs**: To back up BLOBs, select this option. When this option is selected, all BLOBs within the selected database are backed up *regardless of whether the associated table is included*.
- **Disable $ Quoting for Function Bodies**: To disable the use of dollar quoting for function bodies and force them to be quoted using SQL-standard string syntax, select this option.
- **Dump Data as Copy Data:** To dump data using COPY commands instead of INSERT commands during a restore, select this option. Dumping data as INSERT commands generates a separate command for each row of data. This method is safer because any errors that occur result only in the loss of a row of data instead of losing the whole table. Dumping data as COPY commands might result in the loss of the whole table if an error occurs in one of the rows of data.

7 If you selected a **Backup Format** of **Custom Archive File**, select the applicable **Backup Options**:

- **Include BLOBs**: To back up BLOBs, select this option. When this option is selected, all BLOBs within the selected database are backed up *regardless of whether the associated table is included*.
- **Disable $ Quoting for Function Bodies**: To disable the use of dollar quoting for function bodies and force them to be quoted using SQL-standard string syntax, select this option.
- **Compression Level**: Set the compression level between 0 and 9 to use for the custom archive format.

8 If you want to override the default action for an error condition, select the applicable options:

A default action for all backup jobs can be set as outlined in Setting default actions for error conditions (optional).

- For a **Full Database Cluster** backup, the action can be set for the following scenario: **Failed to Backup Configuration Files**. This issue occurs when a configuration file is not found or cannot be backed up for any other reason. The **“pg_hba.conf”** file is used for user authentication; if this file is missing during job execution, the backup fails because of login failure, regardless of the action set for the **Failed to Backup Configuration Files** error condition.
- For an **Individual Database/Table** backup, the action can be set for the following additional scenarios:
  - **Manually Selected Database Unavailable**: This issue occurs when an individual database that was manually selected for inclusion in the backup is unavailable for backup for any reason.
  - **Manually Selected Table Unavailable**: This issue occurs when an individual table that was manually selected for inclusion in the backup is unavailable for backup for any reason.

The plug-in can be configured to take one of following possible courses of action when any of the preceding error conditions occur during the job execution:

- **Complete with Warnings — Saveset Retained:** The job returns a status of **"Backup Completed with warnings"** and a backup saveset is created that includes the items that were successfully backed up.
- **Complete without Warnings — Saveset Retained:** The job completes and returns a status of **"Backup Completed."** The errors are logged in the NetVault Backup binary logs and ignored on the **Job Status** page. A backup saveset is created that includes the items that were backed up.
- **Fail — Saveset Retained:** The job returns a status of **"Backup Failed."** However, a backup saveset is generated that includes the items that were successfully backed up.
- **Fail — No Saveset Retained:** The job returns a status of **"Backup Failed"** and no saveset of backed-up objects is kept. That is, even if some of the objects were successfully backed up, the saveset is discarded.

9 Click **Save**.

10 In the **Create New Set** dialog box, specify a name for the set, and click **Save**.

The name can contain alphanumeric and nonalphanumeric characters, but it cannot contain non-Latin characters. On Linux, the name can have a maximum of 200 characters. On Windows, there is no length restriction. However, a maximum of 40 characters is recommended on all platforms.

Setting backup options for a File Level Data Copy

If you are using PostgreSQL Server 9.6 or later and you selected the File Level Data Copy method, complete the following steps:

1 Next to the **Plugin Options** list, click **Create New**.

2 In the **Backup Method** section on the **PostgreSQL Backup Options** tab, select **File Level Data Copy**.

3 Select the applicable **Backup Level**.

- **Base (Backs Up PostgreSQL Data Directory):** If you are creating a baseline backup of the PostgreSQL database cluster **Data** directory, **$PGDATA**, select this option. Selecting this option backs up the physical files located in the Data directory, $PGDATA. This option performs a non-exclusive backup. Read and write operations are allowed during Base Backups.
- **Incremental (Backs Up Log Files Changed):** If you are backing up the WAL logs that are stored in the staging directory prior to backing them up to NetVault Media, select this option. This option backs up only the WAL files located in the staging directory.

> **IMPORTANT:** PostgreSQL maintains a WAL file in the **pg_wal** subdirectory of the cluster's data directory (For PostgreSQL 9.x and earlier, the WAL file is located in the **pg_xlog** subdirectory of the data directory.) The log records every change to the data files of the cluster databases. Incremental Backups are based on backing up the WAL files. If needed, you can use a sequence of Base and Incremental Backups to reconstruct the database cluster data.
>
> **IMPORTANT:** Changes to the physical view of the database cluster—for example, adding a new database, adding a new table to a database, and so on)—result in changes to the physical file structure of the database cluster. These changes are not included in the WAL files, and thus are not part of Incremental Backups. Whenever you add new objects to the physical view of the Database Cluster, make sure that you run a Base Backup.

4 Select any additional options.

- **Include Tablespaces Outside Default Data Directory:** If you want to include PostgreSQL tablespaces that reside outside the default PostgreSQL Data directory, select this option. You can enter a comma-separated list of directories.
- **Perform Offline Backup:** If you want to omit the PostgreSQL **start** and **stop** commands to ensure that a copy of the existing files are retained in the database cluster, select this option. Selecting this

option before starting disaster recovery procedures retains a copy of the existing files in the database cluster as they existed before the disaster recovery procedure was performed.

> **IMPORTANT:** Before starting a disaster recovery procedure, Quest recommends that you perform an Offline Backup to keep a copy of the existing files in the data directory. During the disaster recovery, if one or more of the existing files are required or might be useful, the Offline Backup ensures that you have access to them.

- **Remove Backed Up Log Files from Logs Backup Directory:** If you do not want to retain log files in the staging directory between backups, select this option. If you want to retain log files in the staging directory between Base Backups, clear this option. Consider creating Base Backup jobs with this option cleared and Incremental Backups jobs with this option selected.

> **NOTE:** If you edit the **PostgreSQL Backup Options** for an existing **File Level Data Copy** job, the **Backup Format** and **Backup Options** sections erroneously appear on the page. These options do not apply and cannot be changed.

5 If you want to override the default action for an error condition, select the applicable options:

A default action for all backup jobs can be set as outlined in Setting default actions for error conditions (optional).

6 Click **Save**.

7 In the **Create New Set** dialog box, specify a name for the set, and click **Save**.

The name can contain alphanumeric and nonalphanumeric characters, but it cannot contain non-Latin characters. On Linux, the name can have a maximum of 200 characters. On Windows, there is no length restriction. However, a maximum of 40 characters is recommended on all platforms.

Finalizing and submitting the job

1 Use the **Schedule**, **Target Storage**, and **Advanced Options** lists to configure any additional required options.

2 Click **Save** or **Save & Submit**, whichever is applicable.

> **TIP:** To run a job that you have already created and saved, select **Manage Job Definitions** in the Navigation pane, select the applicable job, and click **Run Now**.

You can monitor progress on the **Job Status** page and view the logs on the **View Logs** page. For more information, see the *Quest NetVault Backup Administrator's Guide*.

> **IMPORTANT:** If you enabled the Dump Data as Copy Data option, Quest recommends that you also set up additional regular backup jobs, such as weekly or every other week, that do not use the Dump Data as Copy option.

Backing up Global Objects only

You can back up Global Objects in two ways:

- **By using the Full Database Cluster backup type**: A **Full Database Cluster** backup performed in any format includes the Global Objects. Because these objects are cluster-wide and not specific to any individual database, backups of type **Individual Database/Table** do not include Global Objects.
- **By using the Global Objects Only backup type**: You can also back up the Global Objects separately using the **Global Objects Only** backup type provided by the plug-in.

To create a Global Object Only backup, complete the following steps

1. Following the steps outlined in Selecting data for a backup, select the target database-cluster node.
2. If you want to exclude the configuration files, which are automatically selected when you select the database-cluster node, clear the **Configuration Files** node in the selection tree.
3. Following the steps outlined in Setting backup options, select **Global Objects Only** in the **Backup Type** section.
4. If you included the configuration files, select the required action to take if an error occurs during the backup:
    - **Complete with Warnings — Saveset Retained**
    - **Complete without Warnings — Saveset Retained**
    - **Fail — Saveset Retained**
    - **Fail — No Saveset Retained**

    Because the **"pg_hba.conf"** file is used for user authentication, backup fails if this file is missing during job execution. This login failure causes the backup to fail regardless of the action set for the **Failed to Backup Configuration Files** error condition.
5. Following the steps outlined in Finalizing and submitting the job, complete and submit the job.

Backing up configuration files

You can select configuration files for backup with any of the backup types. Regardless of the format selected, the plug-in performs a file-system backup of the following files:

- **postgresql.conf**
- **pg_hba.conf**
- **pg_ident.conf**

To back up these files separately, complete the following steps.

1. Following the steps outlined in Selecting data for a backup, select the **Configuration Files** node in the selection tree.
2. Because the **Backup Type** and **Backup Format** are not considered, select any of the options in **Backup Options** when you create the Backup Options Set by following the steps outlined in Setting backup options.
3. Following the steps outlined in Finalizing and submitting the job, complete and submit the job.

Restoring data

Selecting data for a restore

1 In the Navigation pane, click **Create Restore Job**.

2 On the **Create Restore Job—Choose Saveset** page, select **Plug-in *for PostgreSQL*** from the **Plugin Type** list.

3 To filter the items displayed in the saveset table further, use the **Client**, **Date**, and **Job ID** lists.

The table displays the saveset name (job title and saveset ID), creation date and time, and size. By default, the list is sorted by creation date.

4 In the saveset table, select the applicable saveset to display the database cluster that was the target of the backup.

When you select a saveset, the following details are displayed in the **Saveset Information** area: Job ID, job title, server name, client name, plug-in name, saveset date and time, retirement setting, Incremental Backup or not, Archive or not, saveset size, and snapshot-based backup or not.

The following table outlines the backup type identifiers:

Backup type identifier	Backup type and format
CA FULL FILESYSTEM	• Base File Level Data Copy
CA INCREMENTAL WAL	• Incremental File Level Data Copy
SQL FULL DB CLUSTER ARCHIVE	• Full Database Cluster • TAR/Custom Archive File
SQL FULL DB CLUSTER PLAIN	• Full Database Cluster • Plain-text SQL Script File
SQL INDIVIDUAL ARCHIVE	• Individual Database/Table • TAR/Custom Archive File
SQL INDIVIDUAL PLAIN	• Individual Database/Table • Plain-text SQL Script File
SQL GLOBAL OBJECT ONLY	• Global Objects Only

5 Click **Next**.

6 On the **Create Selection Set** page, select the data that you want to restore.

Based on the backup type, opening the database cluster displays the following:

- **Full Database Cluster**: The following nodes are displayed for this type:

    - **Configuration Files**, if included in the backup
    - **Global Objects**
    - **All Databases**
  - **Individual Database Cluster**: The following nodes are displayed for this type:
    - **Configuration Files**, if included in the backup
    - **All Databases**

7 To view the databases included in the backup, open **All Databases**.

8 For next-level selections, drill down further.

9 Select the database cluster, databases, or tables as required.

> i **IMPORTANT:** BLOBs included in the backup are displayed as a separate node, and not beneath the associated table. Selecting this node restores all BLOBs contained in the saveset and not just objects associated with the selected table. Also note that, to restore BLOBs, this node must be selected explicitly, that is, either this node or the parent database node must be selected. Selecting only the table does *not* restore the associated BLOBs.

Setting restore options

Select the applicable topic:

- If you are restoring a SQL Logical Data Dump, skip to Setting restore options for a SQL Logical Data Dump.
- If you are restoring a File-Level Data Copy, skip to Setting restore options for a File-Level Data Copy.

Setting restore options for a SQL Logical Data Dump

1 On the **Create Selection Set** page, click **Edit Plugin Options**, and configure the applicable parameters on the **Options** tabs.

2 If you are restoring a **Plain-Text SQL Script File** backup, select the applicable options in the **Pre-Restore Options** section to specify what actions should be taken before the plug-in begins the restoration of a database:

  - **Drop and Recreate Database**: Select this option if you want to drop an existing database before restoring it from the backup. The database is created using the template specified in the **Configure** dialog box, **Template Database Used during CREATE Database** option. All existing data is discarded if this option is selected.

    > i **IMPORTANT:** Regardless of whether the **Drop and Recreate Database** option is selected, the plug-in performs this action if you are restoring Template Databases.

  - **Create Database Only**: Select this option if you want to create a nonexistent database or to write data to the existing database. If the database is not found during restore, a new database is created. If the database is present, tables that are not part of the current restore job are not modified in any manner. If the selected tables are present in the database, the action is based on whether the **Include Drop Object Commands** option was selected during backup:
    - **Include Drop Object Commands was selected**: The table is dropped and re-created. The restored table only contains records that were present at the time of backup.
    - **Include Drop Object Commands was not selected**: For existing tables, you can use the **Truncate Table(s)** option to truncate existing records before restoring.

If **Truncate Table(s)** is not enabled, the plug-in attempts to insert all the backed-up records into the existing table. For tables on which the unique or primary keys have been defined, this action might result in constraint errors.

- **Truncate Table(s)**: Select this option if you want to delete all existing records from the tables and restore the records present in the backup saveset. After restoration, the tables only contain records that were present at the time of backup. Be aware of the following:
  - The database must exist.
  - For existing tables, the **"create table"** command during **psql** execution fails. If the **Stop on Error** option is selected, the restore stops immediately on encountering this error. Select the **Continue with Warnings** option if the selected tables are present to ignore these errors and continue with table truncation and data restoration.
  - No errors are encountered for tables that do not exist.

3 If you are restoring a **Plain-Text SQL Script File** backup, select the applicable options in the **Restore Options** section:

- **Stop on Error**: Select this option if you want the restore to stop immediately when an error occurs. Changes made before the error are not rolled back, which leaves the database in an inconsistent state.

  The entire restore job is not canceled; only the restoration of the current database stops. Restoration of other databases included in the job continues normally.

  If the **Disable Triggers** option was enabled for backup and the restoration of the database stops in between, the plug-in does *not* enable triggers defined for the last table that caused the failure.

- **Restore as Single Transaction**: Select this option if all commands in the database backup must complete successfully; otherwise, no changes are applied. The changes made are rolled back if an error occurs.

  The commit or roll-back action applies to a specific database and not the entire restore job. The databases for which restoration has completed successfully are not affected. Similarly, this process does not affect restoration of remaining databases in the job.

- **Continue Restore with Warnings**: Select this option if you want the plug-in to ignore the errors and continue with the restore. The errors encountered during job execution are recorded as warning messages in the NetVault Backup logs and the job status is reported as **"Completed with Warnings."**

  You can use this option to restore nonexistent or nonduplicate records into existing tables without truncating the existing records.

  > **IMPORTANT:** If you select this option, review the logs after the job is complete to confirm that they only contain warnings pertaining to the following:
  >
  > **Primary or Unique Key Constraints Database Object (like table, language, user, role, and others) Already Exists**
  >
  > You can ignore these errors. However, if any other type of error message is noted, take corrective action, such as restoring the database again by selecting **Drop and Recreate Database or Truncate Table(s)**.

4 If you are restoring a **TAR Archive File** or **Custom Archive File** backup, select the applicable options:

- **Stop on Error**: Select this option if you want the restore to stop immediately when an error occurs. Changes made before the error are not rolled back, which leaves the database in an inconsistent state.

  The entire restore job is not canceled; only the restoration of the current database stops. Restoration of other databases included in the job continues normally.

> ℹ **IMPORTANT:** If this option is not selected, the plug-in ignores the errors and continues the job. The errors encountered during job execution are recorded as warning messages in the NetVault Backup logs and the job status is reported as **“Completed with Warnings.”** Review the logs after the job is complete to confirm that they only contain warnings pertaining to the following:
>
> **Primary or Unique Key Constraints Database Object (like table, language, user, role, and others) Already Exists**
>
> You can ignore these errors. However, if any other type of error message is noted, take corrective action, such as restoring the database again by selecting Drop and **Recreate Database** or **Do Not Restore Data for Failed Create Table Commands**.

- **Drop and Recreate Database**: Select this option if you want to drop an existing database before restoring it from the backup. The database is created using the template specified in the **Configure** dialog box, **Template Database Used during CREATE Database** option.

  You can also use the **Exclude Access Privileges** option with **Create Database Only**. To prevent restoration of access privileges, grant and revoke commands, select this option. Without this option selected, the restoration fails if the required users or roles do not exist.

  > ℹ **IMPORTANT:** Regardless of whether the **Drop and Recreate Database** option is selected, the plug-in performs this action if you are restoring Template Databases.

- **Create Database Only**: Select this option if you want to create a nonexistent database or to write data to the existing database. If the database is not found during restore, a new database is created. If the database is present, tables that are not part of the current restore job are not modified in any manner. If the selected tables are present in the database, the action is based on the following options:

  - **Drop Database Objects**: To drop the existing table and restore it from the backup, select this option. If you select this option, you lose the existing records. The restored table only contains records that were present at the time of backup.
  - **Restore Data for Failed Create Table Commands**: If you want the plug-in to insert all the backed-up records into the existing table, select this option. For tables on which the unique or primary keys have been defined, this action might result in constraint errors.
  - If **Stop on Error** option is selected, the restoration of the database stops.
  - **Do Not Restore Data for Failed Create Table Commands**: With this option enabled, the existing tables are not restored.

  You can also use the **Exclude Access Privileges** option with **Create Database Only**. To prevent restoration of access privileges, grant and revoke commands, select this option. Without this option selected, the restoration fails if the required users or roles do not exist.

- **Restore Data Only (No Schema Definitions)**: Select this option if you want to restore only the data from the backup and exclude the schema definitions. The restore fails if the target database or table does not exist.

  > ℹ **IMPORTANT:** Template Databases are *not* restored if you select this option.

  You can also use the **Disable Triggers** option with **Restore Data Only**. Select this option if there are referential integrity checks or other triggers on the tables. This option temporarily disables triggers on the target tables during restore. After successful data restoration, the triggers are enabled. If **Stop on Error** is selected and the restoration of the database stops in between, the plug-in does *not* enable triggers defined for the last table that caused the failure.

5 Click **OK**.

Setting restore options for a File-Level Data Copy

1 On the **Create Selection Set** page, click **Edit Plugin Options**.

2 On the **Restore Options** tab, if you want to select a different location to restore the files to, select the **Relocate all restored files to the directory** check box, and enter the new path in the text box.

If you want to obtain access to a specific set of files without compromising your current environment, you can use this option.

> i **IMPORTANT:** The recovery process for a PostgreSQL Server requires that all the physical files are restored to their original locations; that is, PostgreSQL does not support recovery to alternate paths. However, you can restore files to alternate locations to retrieve specific files manually, before overwriting the original files in the database cluster Data directory. You can also restore files to an alternate location for testing and auditing purposes.
>
> For Incremental Backups, restore the WAL files to the staging directory. During recovery, PostgreSQL Server retrieves WAL files from the staging directory.

3 If you selected a new location and files already exist in the new location for any reason and you want to overwrite them, select the **If existing, overwrite files** option.

4 On the **Base Data Files** tab, select the applicable options:

- **Include Backup Label File with Restore:** If you want to generate a file named **“backup_label”** in the target directory during the restore of a Base Backup, select this option. This file contains information generated by PostgreSQL during the backup, which is useful during recovery. The predefined contents of the **“backup_label”** file are shown in the **Backup Label** field. If you select this option and the **Tablespace Map** field has content, an additional file, **“tablespace_map,”** is also generated during the restore of a Base Backup.
- **Do not Restore WAL Directory Contents (recommended):** If you want to exclude the contents of the WAL directory during a restore of a Base Backup, select this option. The WAL directory is **pg_wal** for PostgreSQL 10.x and later and **pg_xlog** for PostgreSQL Server 9.x and earlier.
- **Do not Restore pg_replslot Directory Contents:** If you want to exclude the contents of the replication slots directory, **pg_replslot**, during a restore of a Base Backup, select this option.
- **Do not Restore postmaster.pid and postmaster.opts Files:** If you want to exclude the **“postmaster.pid”** and **“postmaster.opts”** files, during a restore of a Base Backup, select this option.

The following fields are displayed for information and do not require selection or updating:

- **Backup Label:** This field displays information related to a Base Backup that is included in the **“backup_label”** file described previously. For Offline Backups, this field does not display anything. Unless directed to do so by Quest, do not modify the contents of this field.
- **Stop Log Sequence Number:** This field shows the log sequence number when the **stop backup** command was issued.
- **Tablespace Map:** This field displays information related to a Base Backup. If you select the Include **Backup Label File with Restore** option described previously, this information is included in the **“tablespace_map”** file. For Offline Backups, this field is null. Unless directed to do so by Quest, do not modify the contents of this field.

> **IMPORTANT:** **Tablespace Map** contains tuples of symbolic link names and directories. Before starting a recovery process, if this field displays a tuple, ensure that the database cluster data **pg_tblspc** directory contains a symbolic link with the listed name that points to the directory listed in the tuple.
>
> Examples of creating the link:
>
> Linux or UNIX: `ln -s /postgresdata/my_postgre_tbs_01 /opt/postgres/11/data/pg_tblspc/16405`
>
> Windows: `mklink /J "C:\Program Files\PostgreSQL\11\data\pg_tblspc\16405" "P:\postgresdata\my_postgre_tbs_01"`

5 On the **Point in Time Recovery** tab, select the applicable options:

- **Include Recover Command File with Restore:** If you want to generate a Recovery Command file, select this option. In the **Recovery Command File Location** box, enter the complete path to the location where the Recovery Command file should be placed. If you plan to perform a recovery procedure, With PostgreSQL versions 11 and earlier, PostgreSQL requires that you use the **recovery.conf** file name and that the file is located in the database cluster Data directory. With PostgreSQL 12 and later, for recovery to take place, PostgreSQL requires a file named **recovery.signal** to be present in the database cluster Data directory.

  > **IMPORTANT:** For PostgreSQL versions 12 and later, when you select **Recovery** in Plug-in *for PostgreSQL* Restore options, **recovery.signal** file will be placed in the PostgreSQL database cluster Data directory. The **recovery.signal** file indicates to PostgreSQL Server that recovery has been requested. NetVault Backup writes to **recovery.signal** file the pertinent recovery commands. PostgreSQL versions 12 and later require the recovery commands to be present in PostgreSQL **postgresql.conf** file. When you configure Plug-in *for PostgreSQL*, if you select the option **Allow File Level Data Copy Recovery to Modify postgresql.conf file**. then the plug-in appends the recovery commands in **recovery.signal** to the **postgresql.conf** file. If you do not select **Allow File Level Data Copy Recovery to Modify postgresql.conf file**, then during recovery, before restarting the PostgreSQL Server, ensure to manually copy the contents of **recovery.signal** into the **postgresql.conf** file.

- **Restore Command:** In this box, enter the WAL Log restore command to use when PostgreSQL needs to access a specific log during the recovery process. Use a restore command that copies WAL Log files from the staging directory, You can use the wildcards **%f**, which represents the filename, and **%p**, which represents the full path.

- **Include Recovery Target Time:** If you want to identify a specific point in time from which recovery be captured, select this option. In the text box, use the **recovery_target_time='*<DateAndTime>*'** command. For example, you might enter **recovery_target_time='2019-05-30 13:22:00 PDT'**.

- **Recovery Target Inclusive:** If the Recovery Command file contains the **recovery_target_inclusive** command, select this option. This command attempts to recover all data up to the specified time. If cleared, the process recovers data to the nearest instant prior to the targeted time, which might be a specified time or the current time.

> **IMPORTANT:** You must use a restore command in the Recovery Command file. The other commands are optional.

6 Click **OK**.

Finalizing and submitting the job

The final steps include setting additional options on the Schedule, Source Options, and Advanced Options pages, submitting the job, and monitoring the progress through the Job Status and View Logs pages. These pages and options are common to all NetVault Backup Plug-ins. For more information, see the *Quest NetVault Backup Administrator’s Guide*.

1 To save the settings, click **Ok**, and then click **Next**.

2 In **Job Name**, specify a name for the job if you do not want to use the default setting.

Assign a descriptive name that lets you easily identify the job when monitoring its progress. The job name can contain alphanumeric and nonalphanumeric characters, but it cannot contain non-Latin characters. On Linux, the name can have a maximum of 200 characters. On Windows, there is no length restriction. However, a maximum of 40 characters is recommended on all platforms.

3 In the **Target Client** list, select the machine on which you want to restore the data.

> i **TIP:** You can also click **Choose**, and then locate and select the applicable client in the **Choose the Target Client** dialog box.

4 Use the **Schedule**, **Source Options**, and **Advanced Options** lists to configure any additional required options.

5 Click **Save** or **Save & Submit**, whichever is applicable.

You can monitor progress on the **Job Status** page and view the logs on the **View Logs** page. For more information, see the *Quest NetVault Backup Administrator's Guide*.

6 If you restored a file-level data copy as a part of a recovery procedure intended to build or rebuild the database cluster **Data** directory, complete the following steps to ensure that recovery is completed.

a Restart the PostgreSQL service.

b After the PostgreSQL service has restarted, run the following command:

```
select pg_wal_replay_resume();
```

If the recovery has completed, a **recovery is not in progress** message is displayed. Otherwise, recovery is completed.

For more information, see the example in Example of restoring an archived-based backup sequence

Example of restoring an archived-based backup sequence

For PostgreSQL versions 11 and earlier, in the standard recovery scenario, you restore the Base Backup to the original location, and then restore the sequence of Incremental Backups to the staging directory for WAL files. The next step is to generate a Recovery Command file, **"recovery.conf,"** and place it in the database cluster Data directory. For the final step, you start the PostgreSQL Server, which then reads the Recovery Command file and applies the recovery.

PostgreSQL versions 12 and later do not use the **recovery.conf** file. In its place a file named recovery.signal may be placed in the PostgreSQL database cluster Data directory, to indicate that recovery is requested. In addition, for PostgreSQL 12 onward, the recovery commands are placed in the PostgreSQL **postgresql.conf** file.

For PostgreSQL versions 12 and later, in the standard recovery scenario, you restore the Base Backup to the original location, and then restore the sequence of Incremental Backups to the staging directory for WAL files. The next step is to generate the set of recovery commands and write them in the **postgresql.conf** file. Then, in the PostgreSQL database cluster Data directory, place a file named recovery.signal to indicate that recovery is requested. For the final step, you start the PostgreSQL Server, which then checks for the **recovery.signal** file, and if present then reads the recovery commands from the **postgresql.conf** file, and applies the recovery.

When you configured Plug-in *for PostgreSQL*, if you selected the option **Allow File Level Data Copy Recovery to Modify postgresql.conf file**, then the plug-in appended the recovery commands in **recovery.signal** to the **postgresql.conf** file automatically. If you did not select **Allow File Level Data Copy Recovery to Modify postgresql.conf file**, then during recovery, before restarting the PostgreSQL Server, ensure to manually copy the contents of **recovery.signal** into **postgresql.conf** file.

> i **IMPORTANT:** If selected, Plug-in *for PostgreSQL* places the recovery commands at the end of the **postgresql.conf** file. After recovery, the recovery commands placed in **postgresql.conf** file can be removed.

> ℹ **NOTE:** PostgreSQL Server might replay additional transactions beyond the selected stop time. If you prefer that the recovery only has access to the WAL file transactions that were included in the sequence of Base and Incremental Backups, before you start the recovery process, delete the contents of the PostgreSQL Database Cluster **pg_wal** or **pg_xlog** directory and the contents of the staging directory for WAL files. Transactions committed after the last Base or Incremental Backup of the restore sequence are not replayed.
>
> If you prefer to achieve a rollforward up to the latest available consecutive transaction, in addition to excluding a stop time, keep the existing contents in the **pg_wal** or **pg_xlog** directory and in the staging directory for WAL files.

The following outlines the steps involved in restoring the Base and Incremental Backups for a file-based Base Backup and associated Incremental Backups.

1. Make sure that all PostgreSQL Server services are stopped.

2. Generate an offline backup to savage the contents of the database cluster **Data** directory and the contents of the WAL files directory.

   When creating the offline backup job, select the **Perform Offline Backup** option on the **PostgreSQL Backup Options** tab. For more information, see Setting backup options for a File Level Data Copy.

3. Create a restore job, and select the applicable **Base File Level Data Copy** saveset to restore.

   If the files in the database cluster Data directory are damaged, or you intend to replace them, restore the Base Backup to the original location. To do so, on the **Restore Options** tab, clear the **Relocate all restored files to the directory** option. On the **Base Data Files** tab, select the **Include Backup Label File with Restore** option.

   For more information, see Selecting data for a restore and Setting restore options for a File-Level Data Copy.

   > ℹ If the pg_wal or pg_xlog directory is empty and you want to store the current WAL files in a separate filesystem, you can modify pg_wal or pg_xlog to be a symbolic link to the location of your choice. To do so, delete pg_wal or pg_xlog and replace it with a symbolic link.
   >
   > Examples of creating the link:
   >
   > Linux or UNIX: `ln -s /pg_wal_desired_location /opt/postgres/11/data/pg_wal`
   >
   > Windows: `mklink /J "C:\Program Files\PostgreSQL\11\data\pg_wal" "Q:\pg_wal_desired_location"`

4. Create a restore job, and select the first **Incremental File Level Data Copy** saveset to restore to the staging directory.

5. Repeat the previous step for each **Incremental File Level Data Copy** saveset to restore.

6. For the last **Incremental File Level Data Copy** saveset that you restore, select the **Include Recover Command File with Restore** option, and any other applicable options, on the **Point in Time Recovery** tab.

7. For PostgreSQL versions 12 and later, if you did not configure Plug-in *for PostgreSQL* with the option **Allow File Level Data Copy Recovery to Modify postgresql.conf file**, then append the contents of the **recovery.signal** file to the PostgreSQL **postgresql.conf** file.

8. Restart the PostgreSQL service.

9. After the PostgreSQL service has restarted, use the PostgreSQL **psql** tool to run the following command and complete the recovery:

   ```
   select pg_wal_replay_resume();
   ```

   Example:

   ```
   bash$ psql
   postgres=# select pg_wal_replay_resume();
   pg_wal_replay_resume
   ----------------------
   ```

```
 (1 row)
postgres=#
```

Performing advanced restore procedures

You can complete the following advanced procedures:

- Restoring Global Objects
- Restoring configuration files
- Renaming a database during restore
- Restoring data to a different database cluster
- Restoring to a database cluster on an alternate server

Restoring Global Objects

You can restore Global Objects from a Plain-Text SQL Script File version of a Full Database Cluster or Global Objects Only backup.

1. Select the **Global Objects** node from the backup saveset by following the instructions outlined in Selecting data for a restore.
2. Set the **Restore Options** by following the instructions outlined in Setting restore options and completing the following parameters:
    - **Stop on Error**: Select this option if you want the restore to stop immediately when an error occurs.
    - **Restore as Single Transaction**: Select this option if all commands in the database backup must complete successfully; otherwise, no changes are applied. The changes made are rolled back if an error occurs.
    - **Continue Restore with Warnings**: Select this option if you want the plug-in to ignore the errors and continue with the restore. The errors encountered during job execution are recorded as warning messages in the NetVault Backup logs and the job status is reported as **"Completed with Warnings."**

        IMPORTANT: PostgreSQL does not allow detailed restores of individual Global Objects, nor does it allow you to delete global objects before restoring. Therefore, when restoring Global Objects, all the Global Objects are restored. As a result, you encounter error messages about duplicate items during restore. Quest recommends that you restore Global Objects to an empty database cluster during disaster recovery or use the **Continue Restore with Warnings** option. If the Global Objects are being restored to recover dropped roles or users and the **Continue Restore With Warnings** option has been selected, warnings are logged in the NetVault Backup binary logs for roles or users that exist.
3. Continue with the restore procedure as explained in Finalizing and submitting the job.

To restore Global Objects from a **TAR Archive File** or **Custom Archive File** backup, no additional **Restore Options** are available. Perform Step 1 and Step 3 to restore Global Objects from these backup formats.

For restoring the data to a different database cluster, see Restoring data to a different database cluster.

Restoring configuration files

When restoring configuration files, you do not have to set any restore options—unless you are restoring to a different cluster or server; for more information, see Restoring data to a different database cluster. The plug-in does not let you select individual configuration files for restoration.

1. Select the **Configuration Files** node from the backup saveset by following the instructions outlined in Selecting data for a restore.
2. Set the **Restore Options** by following the instructions outlined in Setting restore options.
3. Continue with the restore procedure as explained in Finalizing and submitting the job.
4. After restoring the configuration files, restart the PostgreSQL Server to apply the settings.

    IMPORTANT: You can restore configuration files when the database server is not running. However, do not include database, table, and Global Objects in the same job. If you attempt to restore any of these items together with the configuration files when the database server is not running, the restore job fails as a result of login failure.

Renaming a database during restore

Renaming a database can be useful if you do not want to overwrite the existing version and want to create a copy of the database.

1 In the Navigation pane, click **Create Restore Job**, select **Plug-in *for PostgreSQL*** from the **Plugin Type** list, select the applicable saveset, and click **Next**.

   For more information, see Selecting data for a restore.

2 On the **Create Selection Set** page, click the database that you want to rename, and select **Rename** from the context menu.

3 In the **Rename/Relocate** dialog box, enter the new name in the **Rename to** box.

   You can enter the name of an existing database to restore the selected tables to this database.

   > i **IMPORTANT:** If the cluster is configured to use UTF-8 encoding at **initdb**, you cannot use a name that contains non-ASCII characters.

4 Click **Ok**.

   The database item is accompanied by renaming information in parentheses.

5 Continue with the restore procedure as explained in Setting restore options and Finalizing and submitting the job.

Restoring data to a different database cluster

You can restore a database to a different database cluster running on the database server. The following topics outline the procedure for performing this type of a restore.

Before initiating this type of restore, verify that the following prerequisites are met:

- **Same version of PostgreSQL installed**: The PostgreSQL version must be same as the version running on the existing database server.
- **Database cluster exists on the PostgreSQL Server**: A database cluster must exist on the PostgreSQL Server and this cluster *must* be added to the plug-in. To add the cluster, see Adding a database cluster.

1 Select the data to restore from the backup saveset by following the instructions outlined in Selecting data for a restore.

2 Set the **Restore Options** by following the instructions outlined in Setting restore options and completing the following parameter:

   Specify the name of the database cluster to which the selected database is to be restored using the **Database Cluster Name** box in the **Restore Target Details** section.

3 Complete the procedure by following the instructions outlined in Finalizing and submitting the job.

   > i **IMPORTANT:** You can also rename a database while performing this type of restore. For more information, see Renaming a database during restore.

Restoring to a database cluster on an alternate server

You can restore a database cluster to a different PostgreSQL Server. Before initiating this type of restore, verify that the following prerequisites are met on the new restore target:

- **Same OS**: The OS running on the target system must be same as the source system.
- **Same version of PostgreSQL installed**: The PostgreSQL version must be same as the version running on the existing database server.
- **NetVault Backup software and Plug-in *for PostgreSQL* installed**: The same version of NetVault Backup software and the plug-in must be installed and configured on the new restore target.
- **Client added to the NetVault Backup Server**: The target machine must be added to the NetVault Backup Server through the NetVault Backup WebUI.
- **Database cluster exists on the new restore target**: A database cluster must exist on the new restore target and the cluster *must* be added to the plug-in. To add the cluster, see Adding a database cluster.

i **IMPORTANT:** If you are not restoring from a Full Database Cluster backup, restore the Global Objects first.

1 Select the database to be restored from the backup saveset by following the instructions outlined in Selecting data for a restore.

2 Set the **Restore Options** by following the instructions outlined in Setting restore options**.**

In addition, specify the name of the database cluster to which the selected database is to be restored using the **Database Cluster Name** box in the **Restore Target Details** section.

3 To save the settings, click **Ok**, and then click **Next**.

4 In **Job Name**, specify a name for the job if you do not want to use the default setting.

Assign a descriptive name that lets you easily identify the job when monitoring its progress. The job name can contain alphanumeric and nonalphanumeric characters, but it cannot contain non-Latin characters. On Linux, the name can have a maximum of 200 characters. On Windows, there is no length restriction. However, a maximum of 40 characters is recommended on all platforms.

i **IMPORTANT:** Do not use special characters that are not supported in a filename on the target OS. For example, do not use the characters /,\,*, or @ on Windows. This requirement is because Plug-in *for PostgreSQL* tries to create a folder with the same name as the job title for restoring data temporarily.

5 In the **Target Client** list, select the new restore target.

i **TIP:** You can also click **Choose**, and then locate and select the applicable client in the **Choose the Target Client** dialog box.

6 Use the **Schedule** and **Advanced Options** lists to configure any additional required options.

7 Click **Save** or **Save & Submit**, whichever is applicable.

You can monitor progress on the **Job Status** page and view the logs on the **View Logs** page. For more information, see the *Quest NetVault Backup Administrator's Guide*.

A

Troubleshooting

This topic describes some common errors and their solutions. If an error occurs that is not described in this table, obtain the PostgreSQL error number from the NetVault Backup Logs, and then see the relevant PostgreSQL documentation for the resolution.

Table 2. Troubleshooting

Symptom	Error	Explanation
Login failed	Cannot connect to the database server.	The PostgreSQL Server is not running. Start the services.
	Password Authentication Failed for the User <*username*>.	The superuser password set in the **Configure** dialog box is invalid. Update the password in the plug-in by following the instructions in Reconfiguring an added database cluster.
	Authentication request failed.	The **“pg_hba.conf”** file is not available for user authentication. If the **“pg_hba.conf**” file contains entries for both IPv4 and IPv6, configure the same authentication for each. For example, use TRUST authentication for both IPv4 and IPv6 connections.

Table 2. Troubleshooting

Symptom	Error	Explanation
Backup failed	Login Failed.	- **Cannot connect to the database server**: The PostgreSQL Server is not running. Start the services. - **Fatal: Password Authentication Failed for the User <*username*>**: The superuser password was changed in the database server after job submission. Update the password in the plug-in by following the instructions in Reconfiguring an added database cluster. - **Authentication request failed**: The **"pg_hba.conf"** is not available for user authentication.
	Backup in **TAR Archive File** format fails with the following message: **Database found, but failed to backup Database <*dbname*>**.	Check the **pg_dump** command with the **TAR Archive** format from a command prompt or terminal session. If the following error is displayed, **"pg_dump: [tar archiver] actual file length <*xxxx*> does not match expected <*xxxx*>,"** you cannot use the **TAR Archive File** backup format on the current installation. Use the **Custom Archive File** format instead.
	On Windows, the backup fails with the following message: **Database was found, but failed to backup <*locale_name*> database.**	Check the **pg_dump** command from the command prompt. If the following error is displayed, **"pg_dump: Cannot access "??-UTF8-C" [database to be dumped]: FATAL: database "??-UTF8-C" does not exist pg_dump: *** Stopped by error,"** the problem is in the character-set encoding configured for the database cluster. The backing up of databases or tables named with non-ASCII characters is not supported if the encoding is set to UTF-8 at **initdb**. Appropriate encoding must be set for the database cluster, for example, EUC_JP for Japanese or EUC_KR for Korean, to back up the databases and tables.
	Backup fails with the following message: **Database found, but failed to backup Database XXXXX.**	Ensure that the password provided in the **Add Database Cluster/Configure Database Cluster** dialog box of the plug-in is correct. If the cluster is configured to use TRUST authentication for HOST connection while PASSWORD authentication for LOCAL connection, the backups fail if the password is incorrect or blank. If the **"pg_hba.conf**" file contains entries for both IPv4 and IPv6, configure the same authentication for each. For example, use TRUST authentication for both IPv4 and IPv6 connections. You can browse the tree and submit a job with a blank or incorrect password. However, the actual job fails. Quest recommends the following in this scenario: - In the **Configure Database Cluster** dialog box, enter the correct password, and run the job again. - To avoid this issue, either use TRUST authentication for both HOST and LOCAL connection—in which case no password is required—or use PASSWORD authentication for both in the **"pg_hba.conf"** file, and provide the correct password in the **Configure Database Cluster** dialog box.

Table 2. Troubleshooting

Symptom	Error	Explanation
Backup Completed with warnings	If the backup job shows this status, check the NetVault Backup Log entries for the job to see if they display one or both of the following messages: - **Failed to add backup record** - **Failed to write index of backup to the database** These messages indicate that the selected data was backed up, but the job's index information was not properly added by NetVault Backup to its database. Without this index information, the data cannot be properly restored.	**Method 1:** Access the **Manage Devices** page of the NetVault Backup WebUI, and perform a scan of the media targeted by the job. NetVault Backup stores index information for backup jobs in two locations: in the NetVault Database and on the media targeted by the backup. Performing this scan adds the index information to the NetVault Database. To verify that the information was added, open the **Create Restore Job—Choose Saveset** page and locate the specific job. If you can browse it and set up a restore job, the scan process has corrected the problem. **Method 2:** If the previous method failed, run the backup job again.
Restore failed	Login Failed.	- **Cannot connect to the database server**: The PostgreSQL Server is not running. Start the services. - **Fatal: Password Authentication Failed for the User <*username*>**: The superuser password was changed in the database server after job submission. Update the password in the plug-in by following the instructions in Reconfiguring an added database cluster. - **Authentication request failed**: The **"pg_hba.conf"** is not available for user authentication.
	Restore job failed and PostgreSQL log lists the following message: **ERROR: cannot drop constraint orders_pkey on table orders because other objects depend on it. HINT: Use DROP … CASCADE to drop the dependent objects too.**	Your PostgreSQL database uses foreign keys to maintain the integrity of the data. To drop the table manually from PostgreSQL, use the **DROP TABLE** command with the **CASCADE** parameter. **DROP TABLE** removes the table from the database, and **CASCADE** ensures that any objects that depend on the table are also dropped. For more information, see your PostgreSQL documentation.

About us

Quest provides software solutions for the rapidly-changing world of enterprise IT. We help simplify the challenges caused by data explosion, cloud expansion, hybrid datacenters, security threats, and regulatory requirements. We are a global provider to 130,000 companies across 100 countries, including 95% of the Fortune 500 and 90% of the Global 1000. Since 1987, we have built a portfolio of solutions that now includes database management, data protection, identity and access management, Microsoft platform management, and unified endpoint management. With Quest, organizations spend less time on IT administration and more time on business innovation. For more information, visit www.quest.com.

Technical support resources

Technical support is available to Quest customers with a valid maintenance contract and customers who have trial versions. You can access the Quest Support Portal at https://support.quest.com.

The Support Portal provides self-help tools you can use to solve problems quickly and independently, 24 hours a day, 365 days a year. The Support Portal enables you to:

- Submit and manage a Service Request.
- View Knowledge Base articles.
- Sign up for product notifications.
- Download software and technical documentation.
- View how-to-videos.
- Engage in community discussions.
- Chat with support engineers online.
- View services to assist you with your product.