How to Use EDB Postgres™ Enterprise Manager to Backup and Restore a Postgres Database

EDB Postgres™ Enterprise Manager
formerly Postgres Enterprise Manager

March 7, 2016

How to Use EDB Postgres Enterprise Manager
to Backup and Restore a Postgres Database
by EnterpriseDB® Corporation

EnterpriseDB Corporation, 34 Crosby Drive, Suite 100, Bedford, MA 01730, USA
T +1 781 357 3390 **F** +1 978 589 5701 **E** info@enterprisedb.com **www**.enterprisedb.com

Table of Contents

1 Introduction

Notice: The names for EDB's products have changed.

The product formerly referred to as Postgres Plus Advanced Server is now referred to as EDB Postgres Advanced Server (Advanced Server).

The product formerly referred to as Postgres Enterprise Manager (PEM) is now referred to as EDB Postgres Enterprise Manager (EDB Enterprise Manager).

Until a new version of this documentation is published, wherever you see an earlier version of a product name, you may substitute it with the current name. Name changes in software and software outputs will be phased in over time.

EDB Postgres Advanced Server (Advanced Server) provides all of the power and flexibility of open-source PostgreSQL, with additional functionality that provides simplified database administration, enhanced SQL capabilities, extended database and application security, performance monitoring and analysis, and application development and management utilities.

Postgres Enterprise Manager (PEM) is an enterprise management tool designed to assist database administrators, system architects, and performance analysts in administering, monitoring, and tuning PostgreSQL and EDB Postgres Advanced Server database servers. The PEM client is distributed with Advanced Server.

In this tutorial, you will learn how to use the PEM client to:

- select backup and restore options
- create a plain-text backup
- create a clone of a database
- create and restore from a custom archive backup

This tutorial will also demonstrate how to use a plain-text backup to create a clone of a database at the `psql` command line.

This EnterpriseDB tutorial assumes that you have already downloaded and installed Advanced Server on your desktop or laptop computer; the PEM client is installed by default with a typical Advanced Server installation.

For information about downloading and installing either Advanced Server or the PEM client, please visit the EnterpriseDB website at:

http://www.enterprisedb.com/

This tutorial uses the term Postgres to refer to EDB Postgres Advanced Server and PostgreSQL. While screenshots were taken in Linux, comparable options are supported on Windows.

2 Using the PEM Client to Backup or Restore

The PEM client provides an easy-to-use graphical interface that invokes the Postgres command line utility programs `pg_dump` and `pg_restore` with the command line options that correspond to your specifications.

2.1 Opening the PEM Client

To open the PEM client, navigate through the `Start` menu (on Windows) or `Applications` menu (on Linux), selecting `Postgres Enterprise Manager v6` from the `Postgres Plus Advanced Server 9.5` menu (see Figure 2.1).

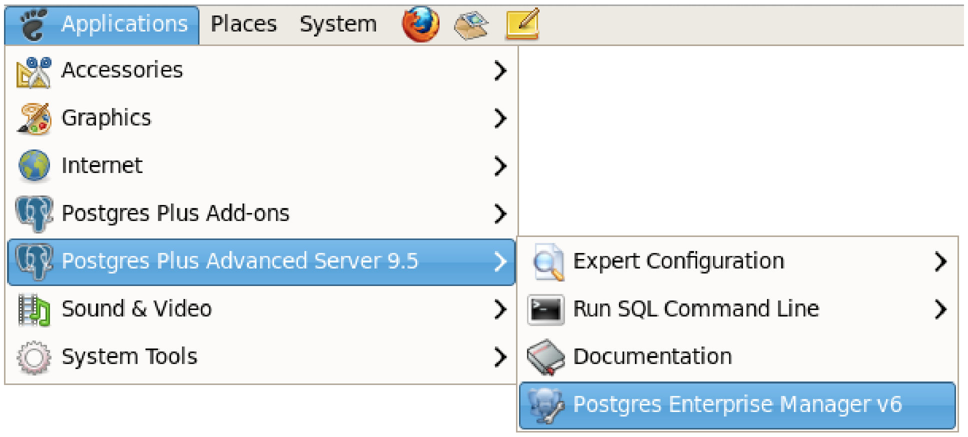
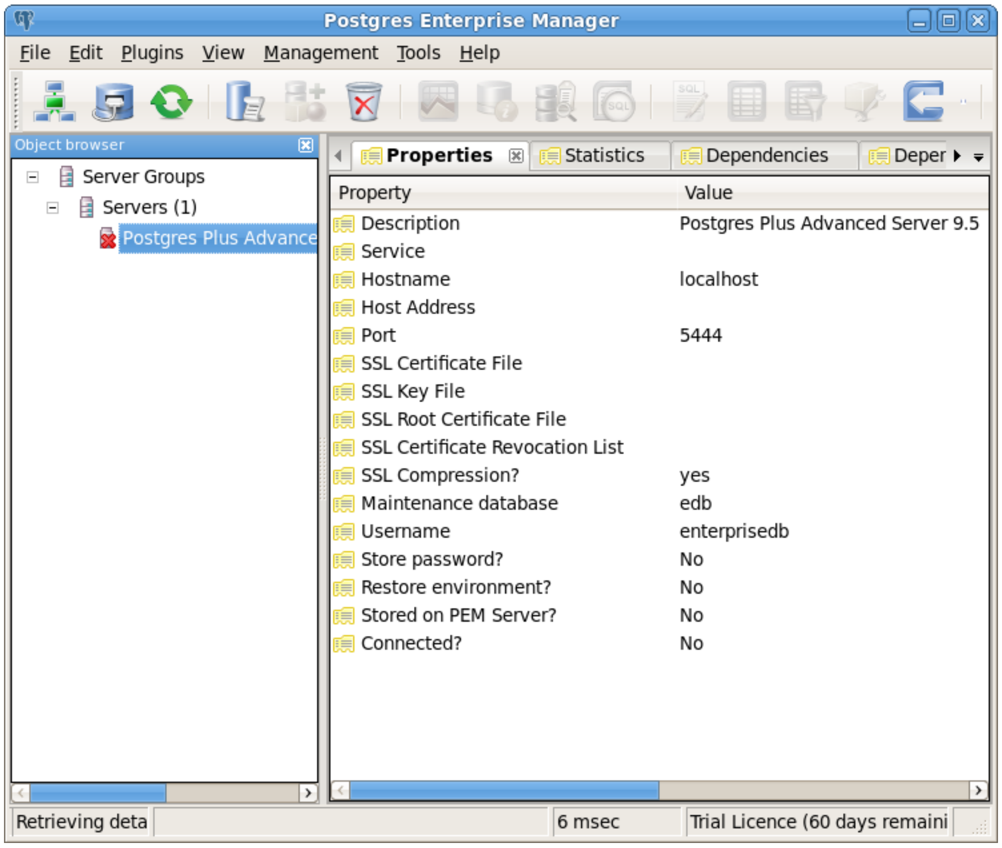


Figure 2.1 – Opening the PEM Client.

The PEM client opens as shown in Figure 2.2.



Property	Value
Description	Postgres Plus Advanced Server 9.5
Service	
Hostname	localhost
Host Address	
Port	5444
SSL Certificate File	
SSL Key File	
SSL Root Certificate File	
SSL Certificate Revocation List	
SSL Compression?	yes
Maintenance database	edb
Username	enterprisedb
Store password?	No
Restore environment?	No
Stored on PEM Server?	No
Connected?	No

Figure 2.2 – The PEM Client.

To connect to a server, right-click on the name of a server in the PEM client `Object browser` panel, and select `Connect` from the context menu; if prompted, provide the password and click `OK`. After connecting to a server, you can expand the tree control in the PEM client `Object browser` to view the currently defined database objects.

2.2 Creating a Backup File

The PEM client `Backup` dialog provides an easy-to-use interface for the `pg_dump` command line utility. Easy to use options on the PEM client allow you to:

- backup a schema definition only, omitting the table data.
- backup the table data only, omitting the schema definition.
- backup database object definitions within a selected schema.
- backup the data from a specific table.

By selecting fields on the `Backup` dialog, you instruct the PEM client which options should be included in a customized `pg_dump` command. The `pg_dump` command writes an archive that you can use with the PEM client's `Restore` dialog, the `psql` client, or `pg_restore` to recreate the objects backed up by the archive.

If you choose to create a plain-text backup, you can review the SQL commands that build the selected object to better help you understand how the object will be recreated. You can also optionally modify the content to create new database objects before restoring.

Supported File Formats

The drop-down listbox in the `Format` field on the `Backup` dialog allows you to select an archive format. Each format has advantages and disadvantages; select the format that is best suited for your application.

Plain. Select `Plain` to generate a plain-text script file containing SQL statements and commands that you can execute at the psql command line or with `pg_dump` to recreate the database objects and load the table data. A plain-text backup file can easily be edited in a text editor if desired before restoring the database objects with the `psql` program. Plain-text format is normally recommended for smaller databases.

Custom. Select `Custom` to generate a `pg_dump` formatted binary file that allows for restoration of all or only selected database objects from the backup file. You can use the PEM client to restore from a custom archive backup file. A custom archive backup file cannot be edited, but you can use the PEM client to select which database objects to restore from the backup file. Custom archive format is recommended for medium to large databases from which you may want to select the database objects to restore from the backup file.

Tar. Select `Tar` to generate a tar archive file that allows for restoration of all or only selected database objects from the backup file. You can use the PEM client to restore from a tar archive backup file.

Directory. Select `Directory` to generate a directory-format archive suitable for use with the PEM client's `Restore` dialog or `pg_restore`. This file format creates a directory with one file for each table and blob being dumped, plus a Table of Contents file describing the dumped objects in a machine-readable format that `pg_restore` can read. A directory format archive can be manipulated with standard Unix tools; for example, files in an uncompressed archive can be compressed with the `gzip` tool. This format is compressed by default and supports parallel dumps.

This tutorial explains the PEM client interface, and how you can use it to create and invoke a `pg_dump` command. For more information about using `pg_dump` options in combination to create complex backups, please consult the PostgreSQL core documentation, available at:

http://www.postgresql.org/docs/current/static/app-pgdump.html

2.2.1 Backing up a Database with PEM

You can use the PEM client to create a backup of an entire database that you can later review or restore. If you save the archive in `Custom`, `Tar`, or `Directory` format, you can also use the backup of an entire database to restore a single object that resides within that database, or table data. To create an archive that you can use to restore a database or database object:

Step 1 – Open the Backup Dialog

Right click on the name of the database in the PEM client tree control and select `Backup…` from the context menu (see Figure 2.3).

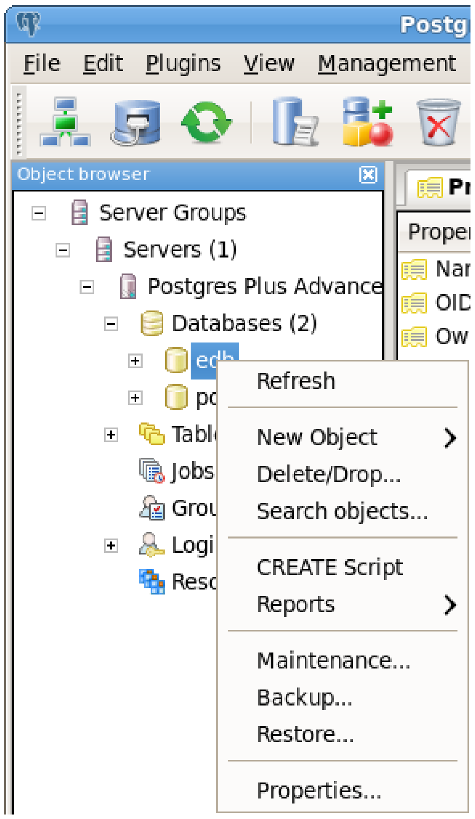


Figure 2.3 – The database context menu.

The `Backup` dialog opens as shown in Figure 2.4.

Step 2 – Specify your Preferences on the Backup database dialog

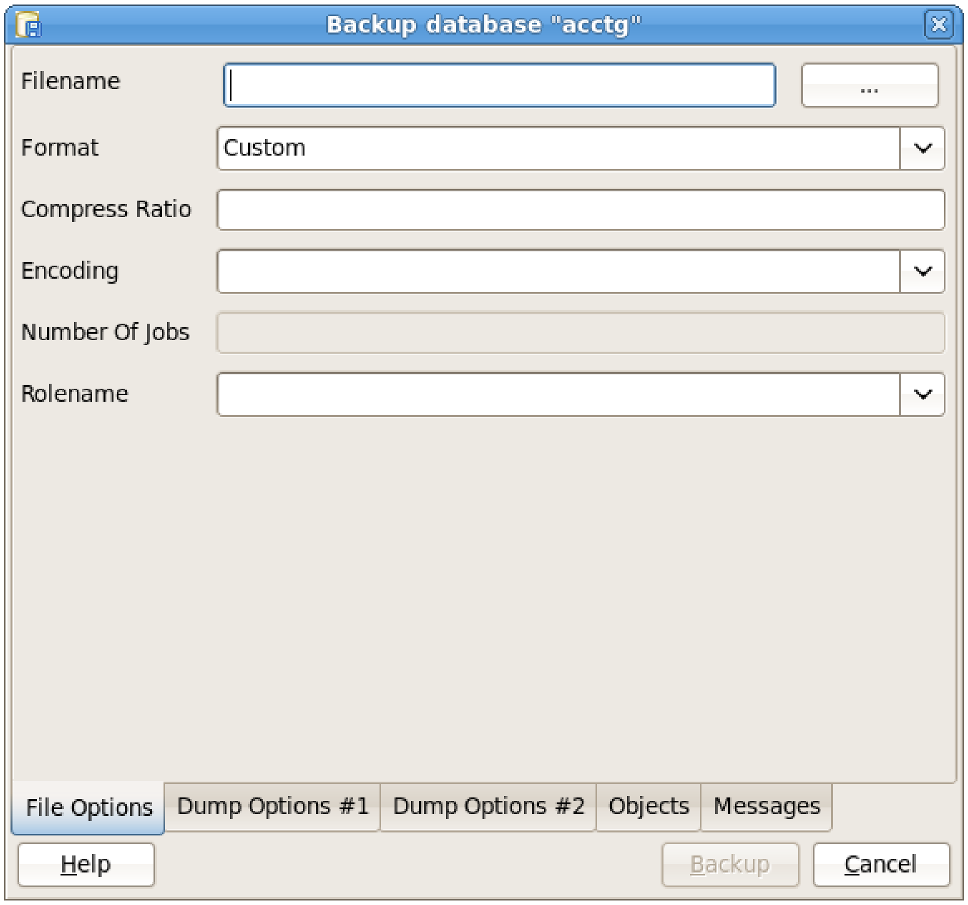


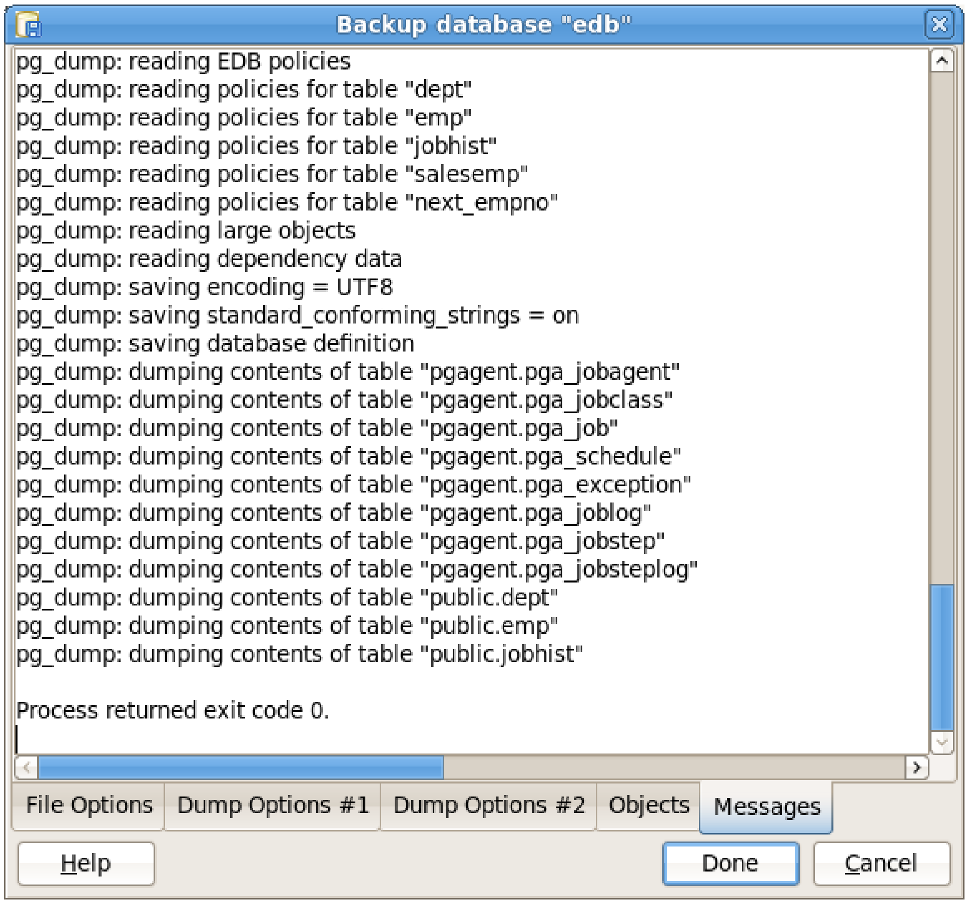
Figure 2.5 – The PEM Client Backup dialog.

- Use the `File Options` tab to specify general information about the backup archive, and select an archive format.

- Use the fields on the `Dump Options #1` and `Dump Options #2` tabs to specify general details about the backup.

- Use the fields on the `Objects` tab to omit any database objects from the backup that you wish to exclude from the archive.

- When you've specified your preferences, click the `Backup` button to build and execute a `pg_dump` command based on those preferences; the result will be displayed on the `Messages` tab (see Figure 2.6).



```
pg_dump: reading EDB policies
pg_dump: reading policies for table "dept"
pg_dump: reading policies for table "emp"
pg_dump: reading policies for table "jobhist"
pg_dump: reading policies for table "salesemp"
pg_dump: reading policies for table "next_empno"
pg_dump: reading large objects
pg_dump: reading dependency data
pg_dump: saving encoding = UTF8
pg_dump: saving standard_conforming_strings = on
pg_dump: saving database definition
pg_dump: dumping contents of table "pgagent.pga_jobagent"
pg_dump: dumping contents of table "pgagent.pga_jobclass"
pg_dump: dumping contents of table "pgagent.pga_job"
pg_dump: dumping contents of table "pgagent.pga_schedule"
pg_dump: dumping contents of table "pgagent.pga_exception"
pg_dump: dumping contents of table "pgagent.pga_joblog"
pg_dump: dumping contents of table "pgagent.pga_jobstep"
pg_dump: dumping contents of table "pgagent.pga_jobsteplog"
pg_dump: dumping contents of table "public.dept"
pg_dump: dumping contents of table "public.emp"
pg_dump: dumping contents of table "public.jobhist"

Process returned exit code 0.
```

Figure 2.6 – The Messages tab.

If the backup is successful, the `Messages` tab will display:

```
    Process returned exit code 0.
```

If you receive an exit code other than 0, scroll through the `Messages` window to locate the problem; after correcting the problem, you can repeat the backup process.

Scroll to the top of the `Messages` dialog to view the `pg_dump` command that created the backup archive. When you're finished, click `Done` to exit the `Backup` dialog.

2.2.2 The PEM Client Backup Dialog – Reference

To open the `Backup` dialog, right click on the name of a database or a named object in the tree control and select `Backup…` from the context menu. The `Backup` dialog opens as shown in Figure 2.7.

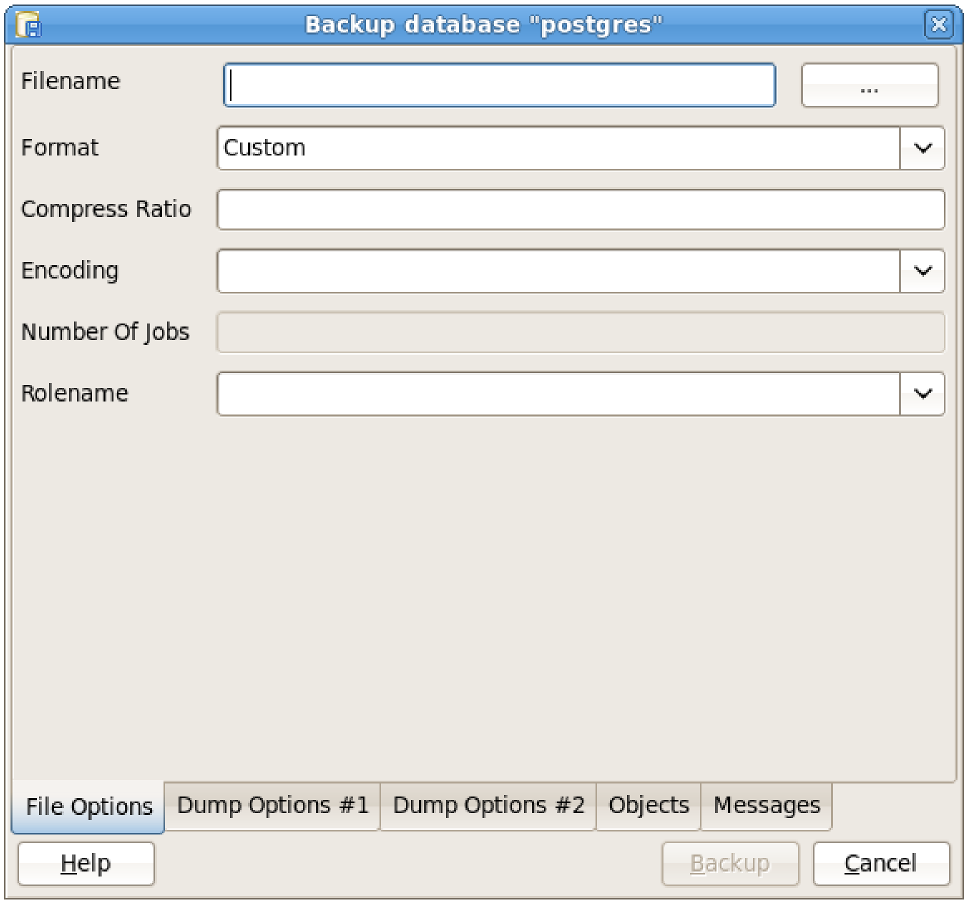


Figure 2.7 – The PEM Client Backup dialog.

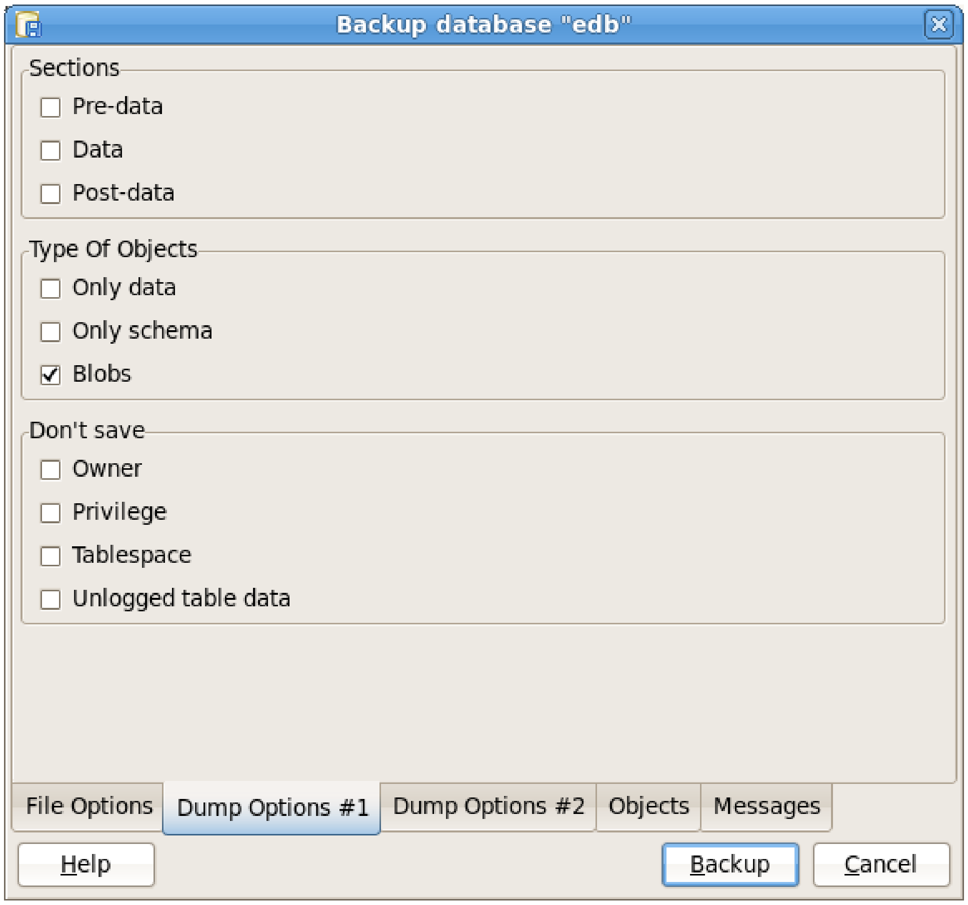
Use the fields on the `File Options` tab to specify general information about the backup.

- Enter the name of the backup file in the `Filename` field. Optionally, use the browser button to navigate into a directory and select a file that will contain the archive.
- Use the drop-down listbox in the `Format` field to select the file format for the backup.
- Use the `Compress Ratio` field to select a compression level for the backup. Specify a value of zero to mean use no compression; specify a maximum compression value of 9. Please note that tar archives do not support compression.

- Use the `Encoding` drop-down listbox to select the encoding that should be used for the archive.

- Use the `Number of Jobs` field (when applicable) to specify the number of tables that will be dumped simultaneously in a parallel backup.

When you've completed the `File Options` tab, navigate to the `Dump Options #1` tab (see Figure 2.8).


Backup database "edb"

Sections
- [ ] Pre-data
- [ ] Data
- [ ] Post-data

Type Of Objects
- [ ] Only data
- [ ] Only schema
- [✓] Blobs

Don't save
- [ ] Owner
- [ ] Privilege
- [ ] Tablespace
- [ ] Unlogged table data

Figure 2.8 – The Dump Options #1 tab.

Use the fields on the `Dump Options #1` tab to specify details about the type of objects that will be backed up.

- Use the checkboxes in the `Sections` box to select a portion of the object that will be backed up. By default, a backup will include all sections.

  Check the box next to `Pre-data` to include all data definition items not included in the `data` or `post-data` item lists.

Check the box next to `data` to backup actual table data, large-object contents, and sequence values.

Check the box next to `Post-data` to include definitions of indexes, triggers, rules, and constraints other than validated check constraints.

- Use the checkboxes in the `Type of Objects` box to select the objects that will be included in the backup. By default, all objects will be included in the backup.

  Check the box next to `Only data` to back up only the data.

  Check the box next to `Only schema` to back up only the schema (the data definitions).

  Check the box next to `Blobs` to include large objects in the backup.

- Use the checkboxes in the `Don't Save` box to select the objects that will not be included in the backup.

  Check the box next to `Owner` to omit commands that set object ownership.

  Check the box next to `Privilege` to omit commands that create access privileges.

  Check the box next to `Tablespace` to omit tablespaces.

  Check the box next to `Unlogged table data` to omit the contents of unlogged tables.

When you've completed the `Dump Options #1` tab, select the `Dump Options #2` tab (see Figure 2.9).

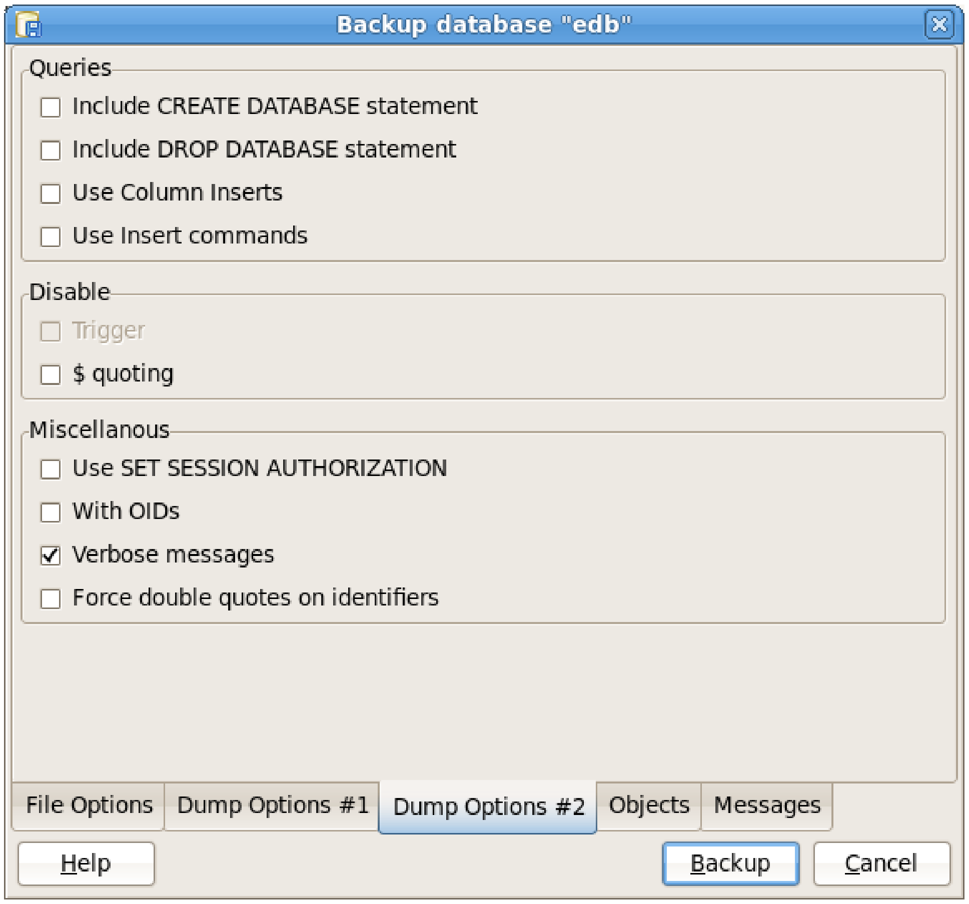


Figure 2.9 – The Dump Options #2 tab.

Use the fields on the `Dump Options #2` tab to specify details about the statements used within the backup file.

- Use the checkboxes in the `Queries` box to specify the type of statements that should be included in the backup.

  Check the box next to `Include CREATE DATABASE statement` to include a command in the backup that creates a new database when restoring from the backup.

  Check the box next to `Include DROP DATABASE statement` to include a command in the backup that drops any existing database before restoring from the backup.

  Check the box next to `Use Column Inserts` to dump the data in the form of `INSERT` statements, with explicit column names. Please note: this may make restoration from backup slow.

Check the box next to `Use Insert` commands to dump the data in the form of `INSERT` statements rather than using a `COPY` command. Please note: this may make restoration from backup slow.

- Use the checkboxes in the `Disable` box to specify the type of statements that should be excluded from the backup.

  Check the box next to `Trigger` (active when creating a data-only backup) to include commands that will disable triggers on the target table while the data is being loaded.

  Check the box next to`$ quoting` to disable dollar quoting within function bodies; if disabled, the function body will be quoted using SQL standard string syntax.

- Use the checkboxes in the `Miscellaneous` box to specify miscellaneous backup options.

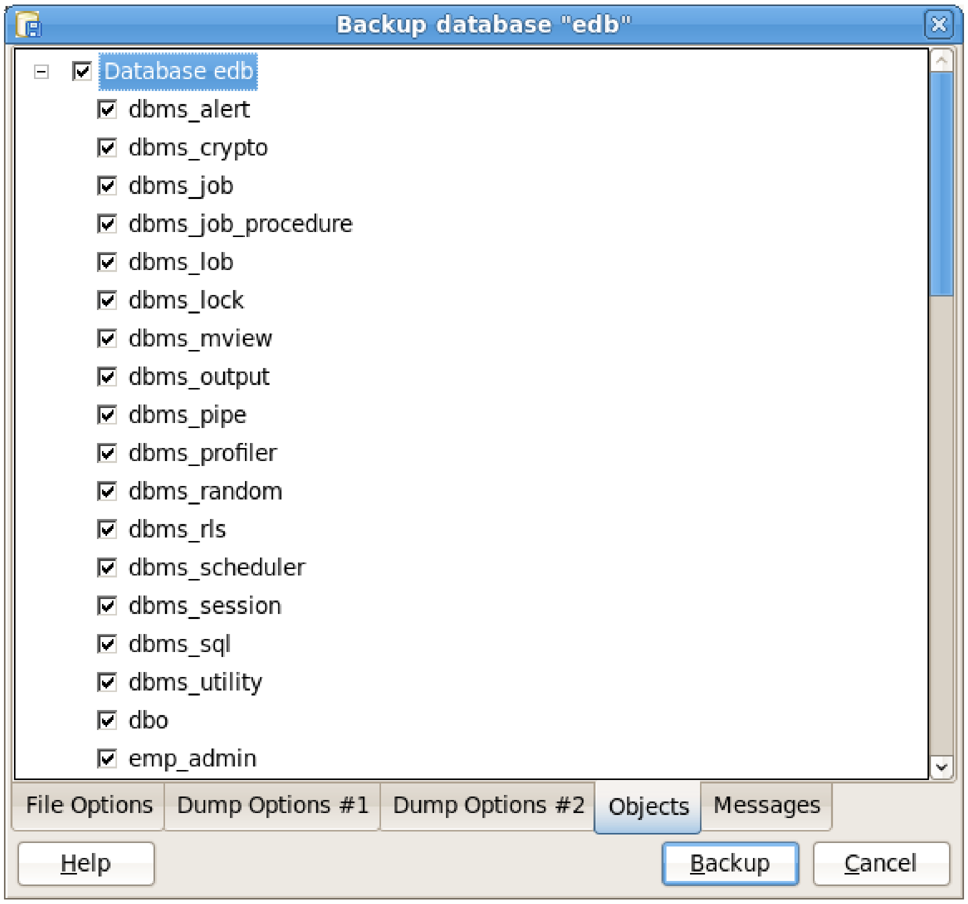
  Check the box next to `Use SET SESSION AUTHORIZATION` to include a statement that will use a `SET SESSION AUTHORIZATION` command to determine object ownership (instead of an `ALTER OWNER` command).

  Check the box next to `With OIDs` to include object identifiers as part of the table data for each table.

  Check the box next to `Verbose messages` to instruct `pg_dump` to use verbose messages.

  Check the box next to `Force double quotes on identifiers` to force the quoting of all identifiers.

When you've completed the `Dump Options #2` tab, select the `Objects` tab (see Figure 2.10).

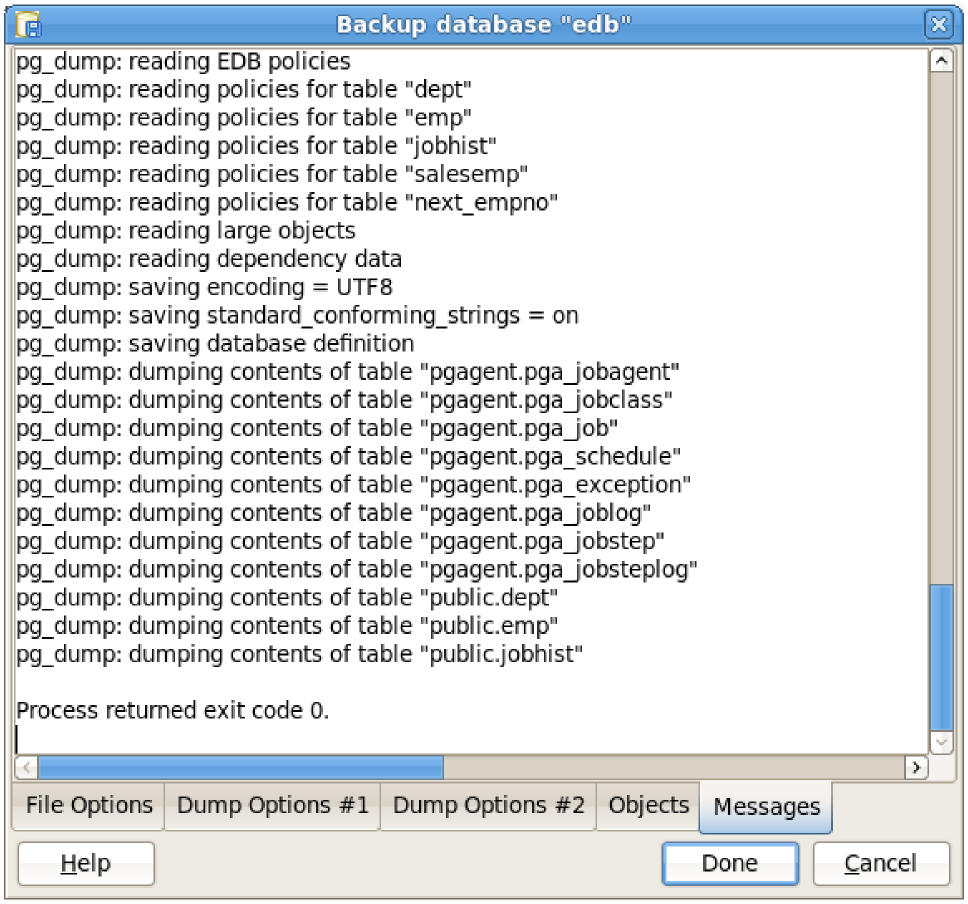


☑ Database edb
- ☑ dbms_alert
- ☑ dbms_crypto
- ☑ dbms_job
- ☑ dbms_job_procedure
- ☑ dbms_lob
- ☑ dbms_lock
- ☑ dbms_mview
- ☑ dbms_output
- ☑ dbms_pipe
- ☑ dbms_profiler
- ☑ dbms_random
- ☑ dbms_rls
- ☑ dbms_scheduler
- ☑ dbms_session
- ☑ dbms_sql
- ☑ dbms_utility
- ☑ dbo
- ☑ emp_admin

Figure 2.10 – The Objects tab.

Use the fields on the `Objects` tab to specify the objects that will be included in the backup; by default, when performing a database backup, all objects will be selected for inclusion in the archive. Deselect an object name to exclude that object from the archive.

When you've specified the details that will be incorporated into the `pg_dump` command, click the `Backup` button to build and execute a command based on those preferences; the result will be displayed on the `Messages` tab (see Figure 2.11).



```
pg_dump: reading EDB policies
pg_dump: reading policies for table "dept"
pg_dump: reading policies for table "emp"
pg_dump: reading policies for table "jobhist"
pg_dump: reading policies for table "salesemp"
pg_dump: reading policies for table "next_empno"
pg_dump: reading large objects
pg_dump: reading dependency data
pg_dump: saving encoding = UTF8
pg_dump: saving standard_conforming_strings = on
pg_dump: saving database definition
pg_dump: dumping contents of table "pgagent.pga_jobagent"
pg_dump: dumping contents of table "pgagent.pga_jobclass"
pg_dump: dumping contents of table "pgagent.pga_job"
pg_dump: dumping contents of table "pgagent.pga_schedule"
pg_dump: dumping contents of table "pgagent.pga_exception"
pg_dump: dumping contents of table "pgagent.pga_joblog"
pg_dump: dumping contents of table "pgagent.pga_jobstep"
pg_dump: dumping contents of table "pgagent.pga_jobsteplog"
pg_dump: dumping contents of table "public.dept"
pg_dump: dumping contents of table "public.emp"
pg_dump: dumping contents of table "public.jobhist"

Process returned exit code 0.
```

Figure 2.11 – The Messages tab.

If the backup is successful, the `Messages` tab will display:

```
Process returned exit code 0.
```

Scroll up to review the `pg_dump` command used to generate the archive, or to view any error messages that were returned during the backup.

2.3 Restoring a Database or a Database Object

If your archive is in a `Custom`, `Tar`, or `Directory` format, you can use the PEM client's `Restore` dialog to restore from the archive. If you have saved the backup in `Plain` format, use the `psql` client to restore.

Using the PEM client, you can restore:

- a schema definition only, omitting the table data.
- table data only, omitting the schema definition.
- a specific database object definition.
- data to a specific table.

By selecting fields on the `Restore` dialog, you instruct the PEM client which options should be included in a customized `pg_restore` command. The `pg_restore` command plays back an archive that recreates the database, database object, or data described by commands within the archive.

If you are restoring into an existing database, you must ensure that any objects that might create conflicts because of pre-existing constraints or dependencies are dropped or truncated; use the `DROP CASCADE` or `TRUNCATE CASCADE` options on the PEM client's context menu to clean up existing conflicts before performing a restore.

This tutorial explains the PEM client interface, and how you can use it to create and invoke a `pg_restore` command. For more information about using `pg_restore` options, please consult the PostgreSQL core documentation, available at:

http://www.postgresql.org/docs/current/static/app-pgrestore.html

2.3.1 Restoring a Database with PEM

You can use the PEM client to restore a database from a `Custom`, `Tar`, or `Directory` formatted backup into an empty database to create a clone of the original database. To create a clone of a database, first create the target database; to create an empty database:

Step 1 – Open the Context Menu

Right-click on the `Database` node of the PEM client tree control and select `New Database…` from the context menu (see Figure 2.12).

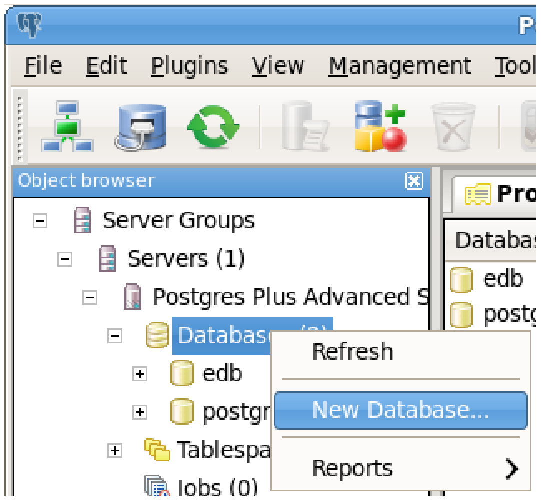


Figure 2.12 – Creating a target database.

Step 2 – Define the Target Database

Use the fields on the `New Database` dialog to define the target database; when you've finished defining the database, click the `OK` button.

Step 3 – Restore Into a Fresh Target Database

Right-click on the name of the new database and select `Restore` from the context menu (see Figure 2.13); when you restore into the fresh target database, the PEM client will re-create the objects in the backup in the new target database.

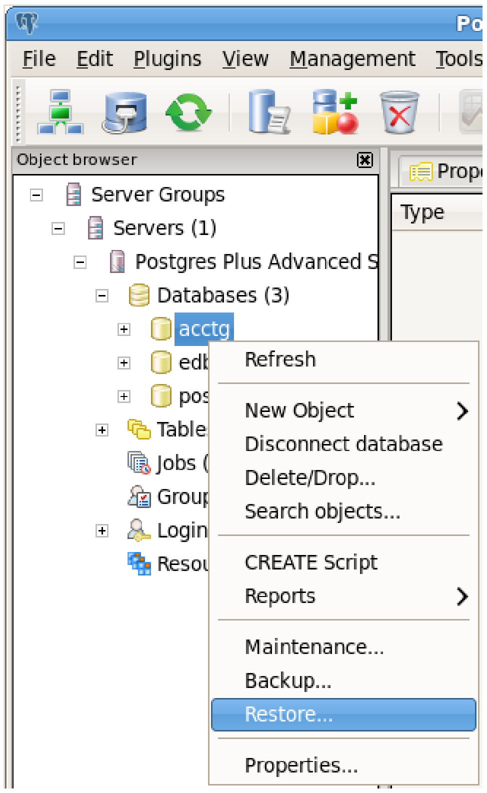


Figure 2.13 – Opening the Restore dialog.

Step 4 – Use the Restore Dialog

Use the `Restore` dialog (shown in Figure 2.14) to build and invoke a customized `pg_restore` command. For detailed information about the options accessible through the `Restore` dialog, see Section 2.3.3.

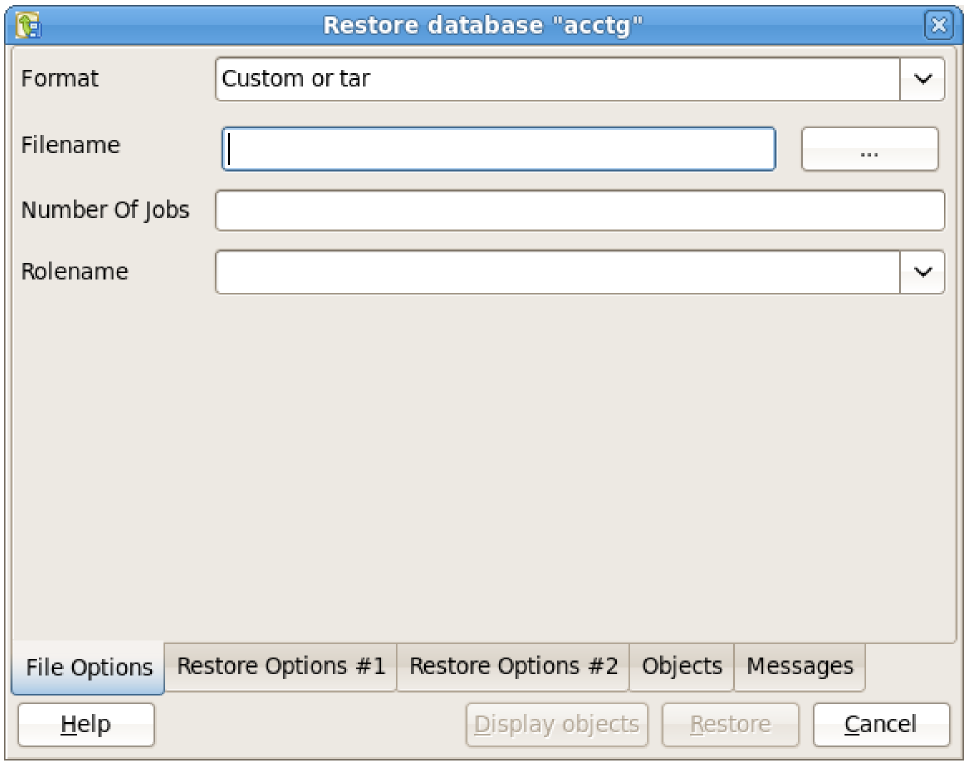
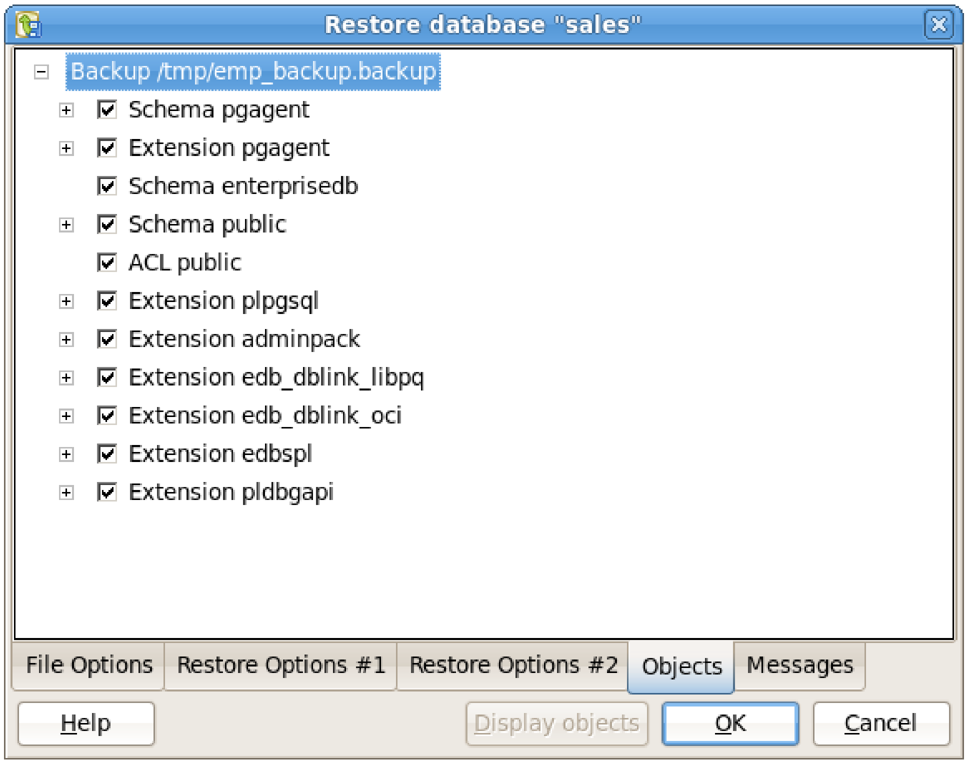


Figure 2.14 – The PEM Client Restore dialog.

Use the `File Options` tab to specify general information about the backup archive that will be invoked, and select an archive format.

Use the fields on the `Restore Options #1` and `Restore Options #2` tabs to specify general details for `pg_restore`.

Use the fields on the `Objects` tab to omit any objects from the restore that you wish to exclude (see Figure 2.15). Navigate to the tab, and select the `Display objects` tab to review a list of the objects that can be created by the selected archive.



Backup /tmp/emp_backup.backup
- ✓ Schema pgagent
- ✓ Extension pgagent
- ✓ Schema enterprisedb
- ✓ Schema public
- ✓ ACL public
- ✓ Extension plpgsql
- ✓ Extension adminpack
- ✓ Extension edb_dblink_libpq
- ✓ Extension edb_dblink_oci
- ✓ Extension edbspl
- ✓ Extension pldbgapi

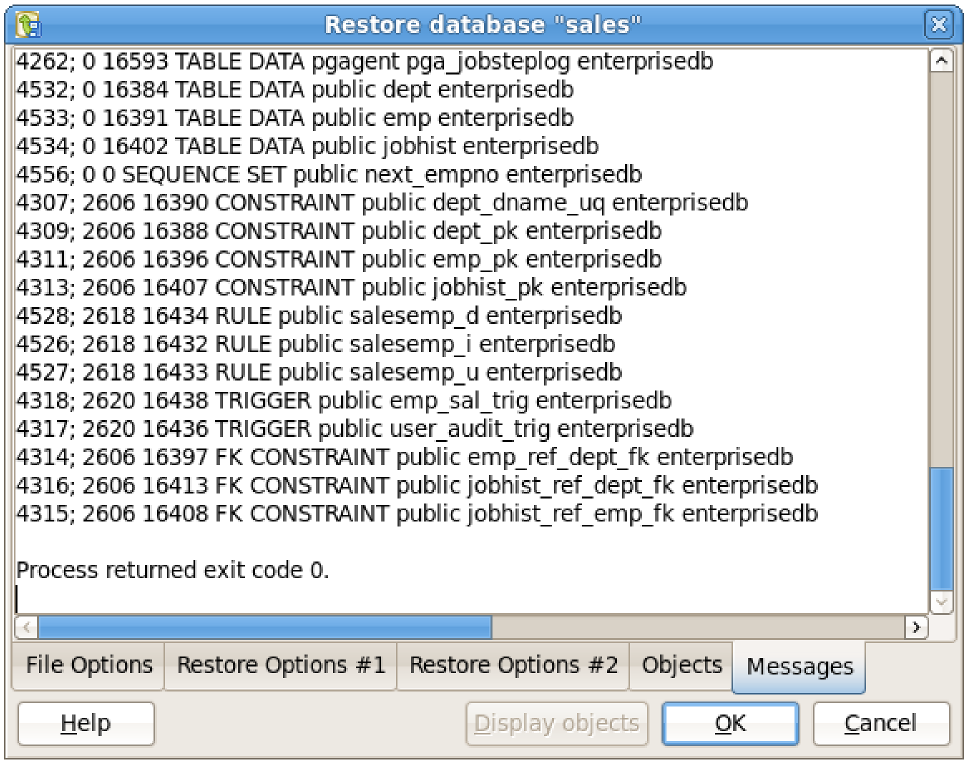
Figure 2.15 – The Objects tab.

Step 5 – Review the Content

Navigate to the tab, and select the `Display objects` tab to review a list of the objects that can be created by the selected archive.

Step 6 – Invoke pg_restore

When you've completed the tabs, navigate to the `Messages` tab, and press `OK` to invoke `pg_restore`.



```
4262; 0 16593 TABLE DATA pgagent pga_jobsteplog enterprisedb
4532; 0 16384 TABLE DATA public dept enterprisedb
4533; 0 16391 TABLE DATA public emp enterprisedb
4534; 0 16402 TABLE DATA public jobhist enterprisedb
4556; 0 0 SEQUENCE SET public next_empno enterprisedb
4307; 2606 16390 CONSTRAINT public dept_dname_uq enterprisedb
4309; 2606 16388 CONSTRAINT public dept_pk enterprisedb
4311; 2606 16396 CONSTRAINT public emp_pk enterprisedb
4313; 2606 16407 CONSTRAINT public jobhist_pk enterprisedb
4528; 2618 16434 RULE public salesemp_d enterprisedb
4526; 2618 16432 RULE public salesemp_i enterprisedb
4527; 2618 16433 RULE public salesemp_u enterprisedb
4318; 2620 16438 TRIGGER public emp_sal_trig enterprisedb
4317; 2620 16436 TRIGGER public user_audit_trig enterprisedb
4314; 2606 16397 FK CONSTRAINT public emp_ref_dept_fk enterprisedb
4316; 2606 16413 FK CONSTRAINT public jobhist_ref_dept_fk enterprisedb
4315; 2606 16408 FK CONSTRAINT public jobhist_ref_emp_fk enterprisedb

Process returned exit code 0.
```

Figure 2.16 – The Messages tab.

When the restoration completes, the `Messages` tab displays details about the restoration process (see Figure 2.16).

If the restore was successful, the `Messages` tab will display:

```
Process returned exit code 0.
```

If you receive an exit code other than `0`, scroll through the `Messages` window to locate the problem; after correcting the problem, you can repeat the process.

Scroll to the top of the `Messages` dialog to view the executed `pg_restore` command. When you're finished, click `Done` to exit the `Restore` dialog.

2.3.2 Restoring a Database from a Plain-Text Backup File

You can use the `psql` client to restore a database from a plain-text formatted backup into an empty database to create a clone of the original database. A plain-text file backup contains a series of SQL commands that, when executed, recreate the objects that were backed up.

To open the `psql` command line, navigate through the `Applications` menu (on Linux) or `Start` menu (on Windows) to the `Postgres Plus Advanced Server 9.5` menu, and select `EDB-PSQL` from the `Run SQL Command Line` menu.

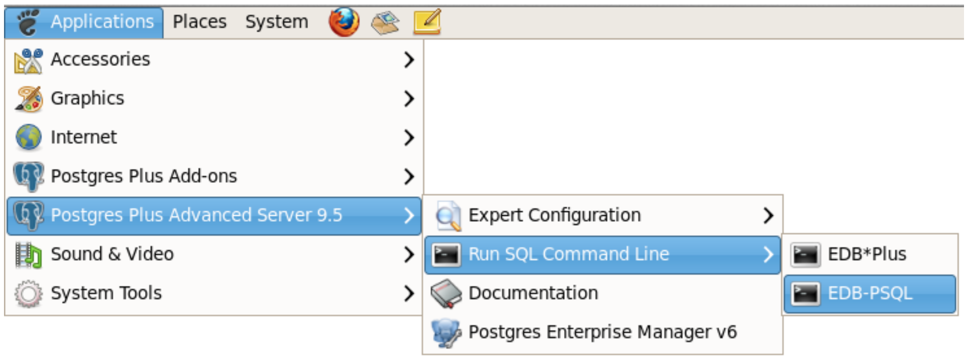


Figure 2.17 – Opening the psql client.

When the `psql` command line client opens, provide connection information for an existing database (see Figure 2.17). After connecting to the server, create the target database and invoke the plain-text script:

1. Execute the `CREATE DATABASE` command, specifying the following options:

   ```
   CREATE DATABASE db_name OWNER db_superuser
   TEMPLATE template0;
   ```

   Where:

   `db_name` is the name of the database.

   `db_superuser` is the name of a database superuser.

2. Connect to the new database with the following `psql` meta-command:

   ```
   \c db_name
   ```

   Where:

   `db_name` is the name of the database.

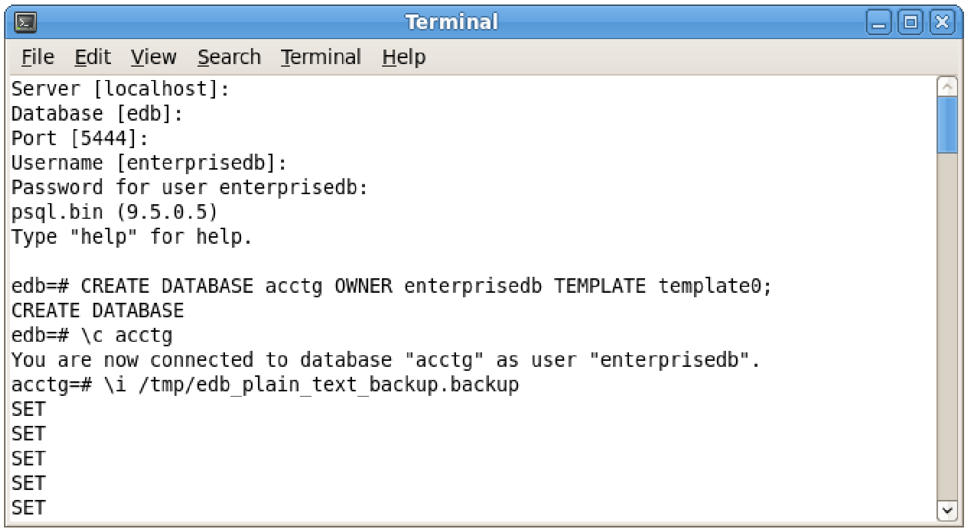
3. Then, to execute the commands in a plain-text file created by the PEM client, execute the command:

    ```
    \i path_name
    ```

    Where:

    `path_name` specifies the complete path to the backup archive.

The `\i` meta command invokes the commands within the archive one line at a time; as each command executes, the `psql` client will display the result of each SQL command (see Figure 2.18).

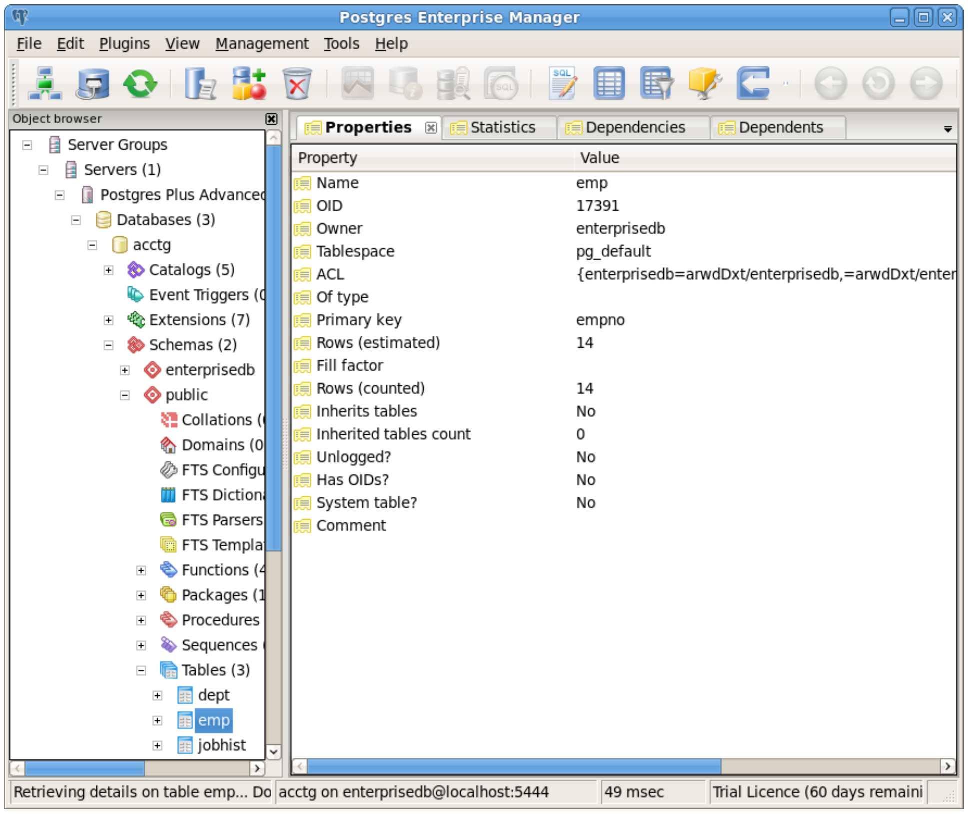


```
Server [localhost]:
Database [edb]:
Port [5444]:
Username [enterprisedb]:
Password for user enterprisedb:
psql.bin (9.5.0.5)
Type "help" for help.

edb=# CREATE DATABASE acctg OWNER enterprisedb TEMPLATE template0;
CREATE DATABASE
edb=# \c acctg
You are now connected to database "acctg" as user "enterprisedb".
acctg=# \i /tmp/edb_plain_text_backup.backup
SET
SET
SET
SET
SET
```

Figure 2.18 – Creating a database from a Plain-text archive.

When playback completes, you can view the populated database in the PEM client (see Figure 2.19), or use the `psql` command line to manage your new database.



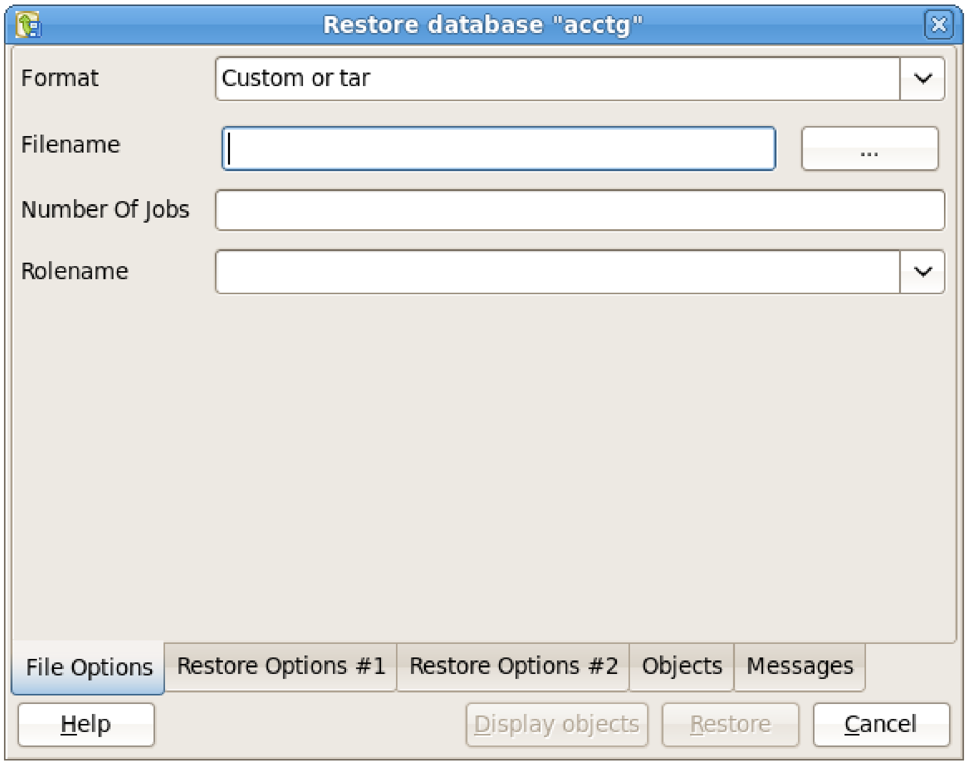
Property	Value
Name	emp
OID	17391
Owner	enterprisedb
Tablespace	pg_default
ACL	{enterprisedb=arwdDxt/enterprisedb,=arwdDxt/enter
Of type	
Primary key	empno
Rows (estimated)	14
Fill factor	
Rows (counted)	14
Inherits tables	No
Inherited tables count	0
Unlogged?	No
Has OIDs?	No
System table?	No
Comment	

```
Retrieving details on table emp... Do | acctg on enterprisedb@localhost:5444 | 49 msec | Trial Licence (60 days remaini
```

Figure 2.19 – The cloned database in the PEM client.

2.3.3 The PEM Client Restore Dialog - Reference

To open the `Restore` dialog, right click on the name of an object in the tree control and select `Restore` from the context menu. The `Restore` dialog opens as shown in Figure 2.20.


Restore database "acctg"

Format: Custom or tar

Filename:

Number Of Jobs:

Rolename:

Figure 2.20 – The PEM Client Restore dialog.

Use the fields on the `File Options` tab to general information about the backup archive.

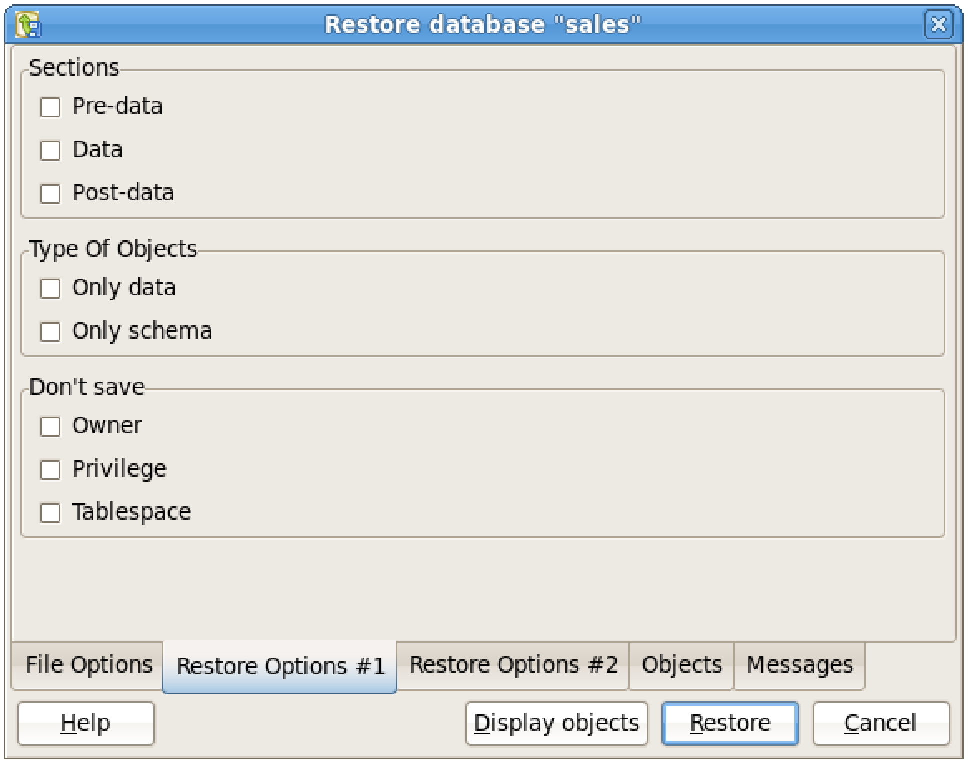
- Use the `Format` field to select the file format of the archive you are restoring. The PEM client can restore from a `Custom` file (`pg_dump` format), a `Tar` file, or a `Directory` format file.

- Use the `Filename` field to specify the name of the backup archive that will be used for the restore; optionally, use the file browser to navigate to and select the file.

- Use the `Number of Jobs` field to instruct the server to use multiple concurrent jobs for the restore. The optimal value for this option depends on the hardware setup of the server, of the client, and of the network. This option is supported only by the `Custom` and `Directory` archive formats.

- Use the `Rolename` field to specify the name of the role that will be used when invoking `pg_restore`.

When you've completed the `File Options` tab, navigate to the `Restore Options #1` tab (see Figure 2.21).


Restore database "sales"

Sections
☐ Pre-data
☐ Data
☐ Post-data

Type Of Objects
☐ Only data
☐ Only schema

Don't save
☐ Owner
☐ Privilege
☐ Tablespace

Figure 2.21 – The Restore Options #1 tab.

Use the fields on the `Restore Options #1` tab to specify details about the type of objects that will be restored.

- Use the checkboxes in the `Sections` box to select a portion of the object that will be restored. By default, a restore will include all sections.

  Check the box next to `Pre-data` to include all data definition items not included in the `data` or `post-data` item lists.

  Check the box next to `data` to include actual table data, large-object contents, and sequence values.

  Check the box next to `Post-data` to include definitions of indexes, triggers, rules, and constraints other than validated check constraints.

- Use the checkboxes in the `Type of Objects` box to select the objects that will be restored. By default, all objects will be included.

  Check the box next to `Only data` to restore only the data.

  Check the box next to `Only schema` to restore only the schema (the data definitions).

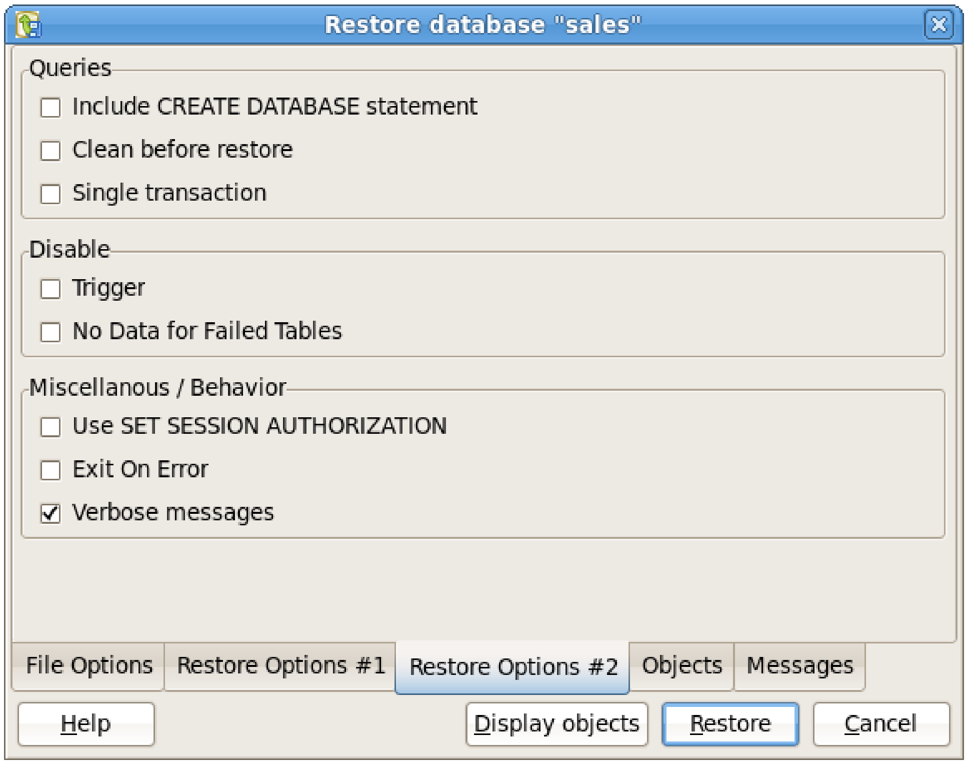
- Use the checkboxes in the `Don't Save` box to select the objects that will not be included.

  Check the box next to `Owner` to omit commands that set object ownership.

  Check the box next to `Privilege` to omit commands that create access privileges.

  Check the box next to `Tablespace` to omit tablespaces.

When you've completed the `Restore Options #1` tab, select the `Restore Options #2` tab (see Figure 2.22).



Queries
- [ ] Include CREATE DATABASE statement
- [ ] Clean before restore
- [ ] Single transaction

Disable
- [ ] Trigger
- [ ] No Data for Failed Tables

Miscellanous / Behavior
- [ ] Use SET SESSION AUTHORIZATION
- [ ] Exit On Error
- [✓] Verbose messages

Figure 2.22 – The Restore Options #2 tab.

- Use the checkboxes in the `Queries` box to specify how the restore process should handle statements.

    Check the box next to `Include CREATE DATABASE statement` to include a command in the backup that creates a new database when restoring from the backup.

    Check the box next to `Clean before restore` to instruct the server to drop an object before restoring it from the archive. Please note that this option does not remove all dependencies for all objects; manual cleanup may be required before restoring into an existing database.

    Check the box next to `Single transaction` to execute the restore as a single transaction; this ensures that all commands will complete successfully before the changes are applied.

- Use the checkboxes in the `Disable` box to specify if triggers should be disabled, or if the server should not attempt to load failed tables.

    Check the box next to `Trigger` (when creating a data-only backup) to include commands that will disable triggers on the target table while the data is being loaded.

    Check the box next to `No Data for Failed Tables` to instruct pg_restore to not load table data if the create command for a table fails. Specifying this option prevents duplicate or obsolete data from being loaded into an existing table.

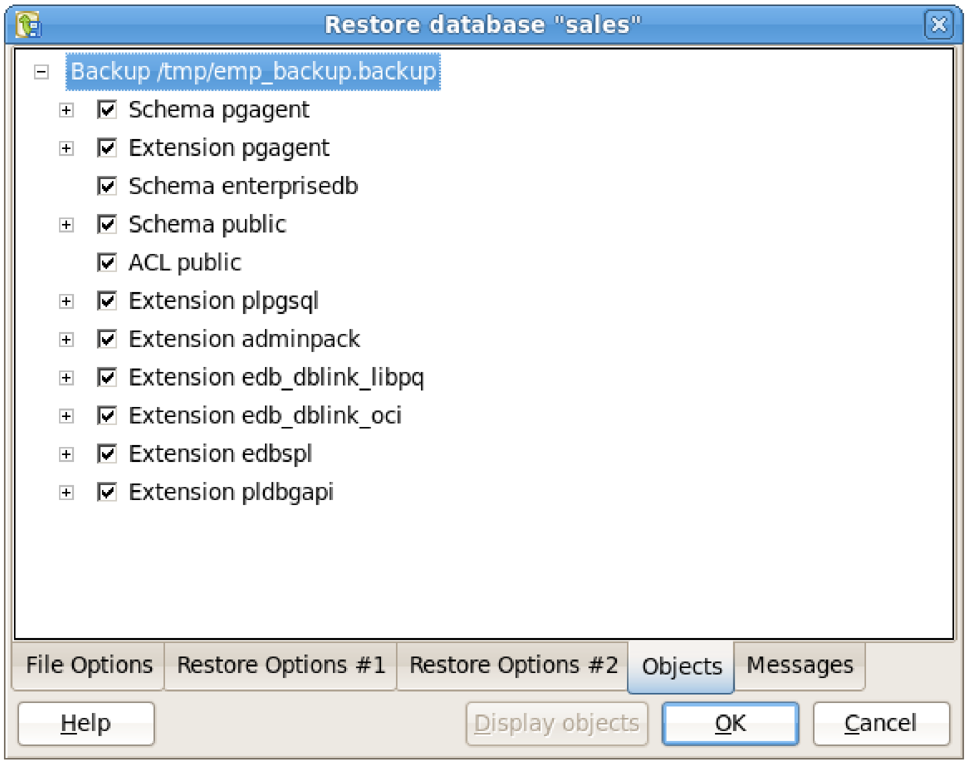
- Use the checkboxes in the `Miscellaneous Behavior` box to specify additional restore options.

    Check the box next to `Use SET SESSION AUTHORIZATION` to use SQL-standard `SET SESSION AUTHORIZATION` commands instead of `ALTER OWNER` commands to determine object ownership.

    Check the box next to `Exit on Error` to exit the restore if an error is encountered while sending SQL commands to the database.

    Check the box next to `Verbose messages` to instruct `pg_restore` to use verbose messages.

When you've completed the `Restore Options #2` tab, navigate to the `Objects` tab (see Figure 2.23).

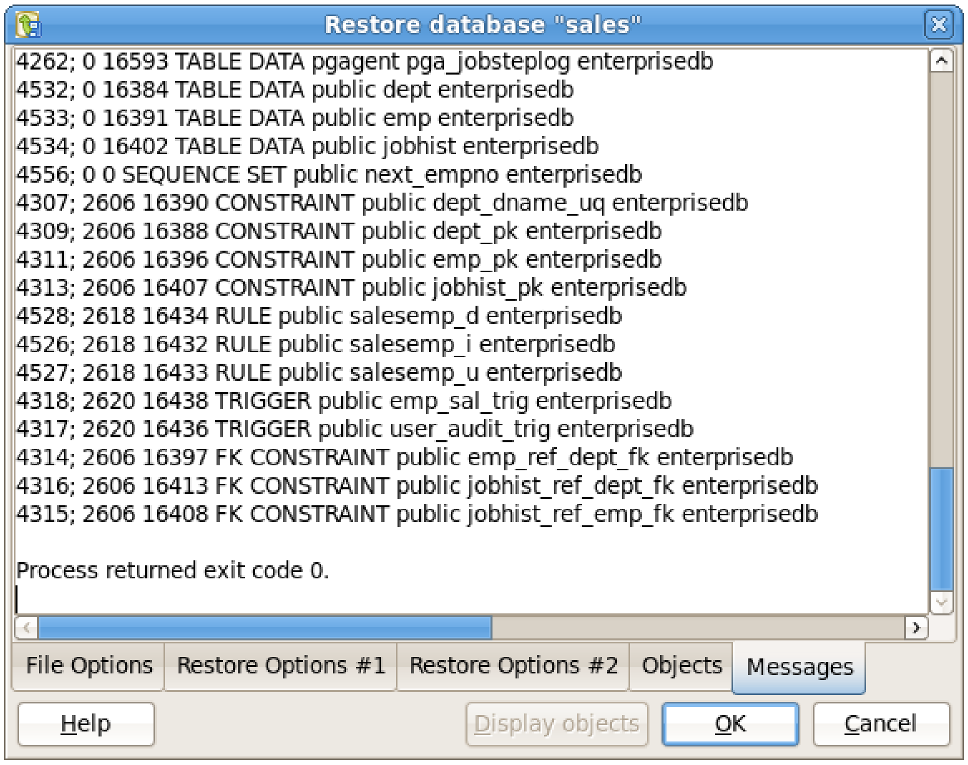


Backup /tmp/emp_backup.backup
- ☑ Schema pgagent
- ☑ Extension pgagent
- ☑ Schema enterprisedb
- ☑ Schema public
- ☑ ACL public
- ☑ Extension plpgsql
- ☑ Extension adminpack
- ☑ Extension edb_dblink_libpq
- ☑ Extension edb_dblink_oci
- ☑ Extension edbspl
- ☑ Extension pldbgapi

Figure 2.23 – The Objects tab.

Click the `Display objects` button to populate the tree control on the `Objects` tab; when the list of objects is displayed, check the box to the left of an object name to include that object from the restore.

When you've completed the tabs, navigate to the `Messages` tab, and press `OK` .



```
4262; 0 16593 TABLE DATA pgagent pga_jobsteplog enterprisedb
4532; 0 16384 TABLE DATA public dept enterprisedb
4533; 0 16391 TABLE DATA public emp enterprisedb
4534; 0 16402 TABLE DATA public jobhist enterprisedb
4556; 0 0 SEQUENCE SET public next_empno enterprisedb
4307; 2606 16390 CONSTRAINT public dept_dname_uq enterprisedb
4309; 2606 16388 CONSTRAINT public dept_pk enterprisedb
4311; 2606 16396 CONSTRAINT public emp_pk enterprisedb
4313; 2606 16407 CONSTRAINT public jobhist_pk enterprisedb
4528; 2618 16434 RULE public salesemp_d enterprisedb
4526; 2618 16432 RULE public salesemp_i enterprisedb
4527; 2618 16433 RULE public salesemp_u enterprisedb
4318; 2620 16438 TRIGGER public emp_sal_trig enterprisedb
4317; 2620 16436 TRIGGER public user_audit_trig enterprisedb
4314; 2606 16397 FK CONSTRAINT public emp_ref_dept_fk enterprisedb
4316; 2606 16413 FK CONSTRAINT public jobhist_ref_dept_fk enterprisedb
4315; 2606 16408 FK CONSTRAINT public jobhist_ref_emp_fk enterprisedb

Process returned exit code 0.
```

Figure 2.24 – The Messages tab.

When the restoration completes, the `Messages` tab displays details about the restoration process (see Figure 2.24).