

# How to Dump and Restore Postgres Plus<sup>(R)</sup> Databases Using pgAdmin

## A Postgres Evaluation Quick Tutorial From EnterpriseDB

December 7, 2009

EnterpriseDB Corporation, 235 Littleton Road, Westford, MA 01866, USA  
**T** +1 978 589 5700 **F** +1 978 589 5701 **E** [info@enterprisedb.com](mailto:info@enterprisedb.com) **www**.[enterprisedb.com](http://enterprisedb.com)

# Introduction

Learn how to use the pgAdmin GUI in Standard Server (Postgres Studio in Advanced Server) to safeguard Postgres Plus databases. You will then be able to build a database and an application for a Technical Evaluation, knowing you can easily create intermittent database backups of your work and restore them if needed.

This [EnterpriseDB Quick Tutorial](#) helps you get started with the [Postgres Plus Standard Server](#) or [Postgres Plus Advanced Server](#) database products in a Linux, Windows or Mac environment. It is assumed that you have already downloaded and installed Postgres Plus Standard Server or Postgres Plus Advanced Server on your desktop or laptop computer.

This Quick Tutorial is designed to help you expedite your Technical Evaluation of Postgres Plus Standard Server or Postgres Plus Advanced Server. For more informational assets on conducting your evaluation of Postgres Plus, visit the self-service web site, [Postgres Plus Open Source Adoption](#).

In this Quick Tutorial you will learn how to do the following using the pgAdmin GUI console:

- Distinguish between backup formats
- Choose among various backup and restore options
- Create a plain text backup and restore it
- Create a custom archive backup and restore it

## Feature Description

The graphical user interface for database administration in Postgres Plus Standard Server is named pgAdmin (Postgres Studio if you are using Advanced Server). The capabilities and appearance of pgAdmin and Postgres Studio are the same, and both give you a quick and easy way to back up and restore Postgres Plus database objects.

For the remainder of this Quick Tutorial, the discussion will refer to pgAdmin, though the capabilities described apply equally to Postgres Studio.

The actual backup and restore operations are carried out by the Postgres Plus command line utility programs `pg_dump` and `pg_restore`. When you use pgAdmin to back up or restore database objects, pgAdmin builds and executes a command that calls the `pg_dump` program or the `pg_restore` program with the appropriate parameters. You can view the `pg_dump` or `pg_restore` command built and executed by pgAdmin to help you better understand the backup or restore operation performed, and also to serve as a training aid for running `pg_dump` and `pg_restore` on the command line without using pgAdmin.

While using pgAdmin provides a simple and quick method of performing most common backup and restore operations, using `pg_dump` and `pg_restore` on the command line provides additional advanced options.

For complete information on how to create a backup file using `pg_dump`, see [pg\\_dump](#) in Chapter “PostgreSQL Client Applications” under VI. “Reference” of the PostgreSQL Core Documentation found on the [Postgres Plus documentation web page](#).

For complete information on how to restore a backup file using `pg_restore`, see [pg\\_restore](#) in Chapter “PostgreSQL Client Applications” under VI. “Reference” of the PostgreSQL Core Documentation found on the [Postgres Plus documentation web page](#).

This Quick Tutorial addresses one of several backup and restore strategies available in Postgres Plus. For a complete discussion of all the different backup and restore strategies available in Postgres Plus, see [Chapter 24, "Backup and Restore"](#) of the PostgreSQL Core Documentation found on the [Postgres Plus documentation web page](#).

## Tutorial Steps

### *Backup File Formats*

Three different backup file formats can be created by pgAdmin:

- **Plain-Text Format.** A plain-text script file containing SQL statements and commands that can be executed by the `psql` command line terminal program to recreate the database objects and load the table data. Use the `psql` program to restore from a plain-text backup file.
- **Custom Archive Format.** A binary file that allows for restoration of all or only selected database objects from the backup file. Use pgAdmin to restore from a custom archive backup file.
- **Tar Archive Format.** A tar archive file that allows for restoration of all or only selected database objects from the backup file. Use pgAdmin to restore from a tar archive backup file.

A plain-text backup file can be edited in a text editor if desired before restoring its database objects with the `psql` program. Plain-text format is normally recommended for smaller databases.

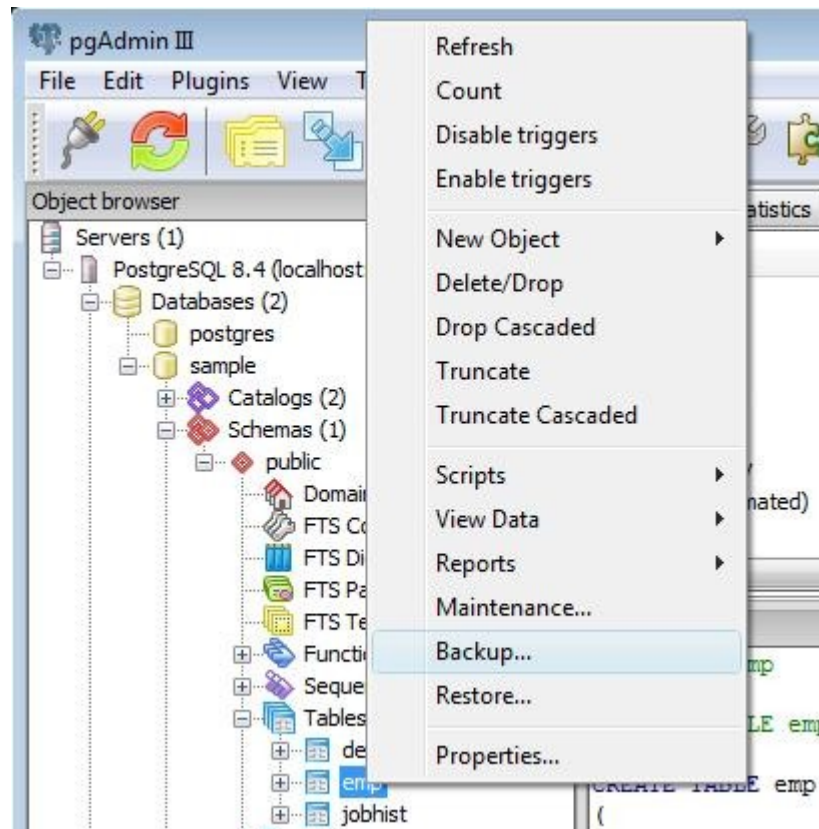
A custom archive backup file cannot be edited. However, you can use pgAdmin to select which database objects to restore from the backup file. Custom archive format is recommended for medium to large databases for which you may want to select the database objects to restore from the backup file.

A tar archive backup file can be manipulated by standard Linux tools such as `tar`. Like custom archive format, pgAdmin can be used to select which database objects to restore

from the backup file.

## ***Backup and Restore Options***

You select a database object for backup or restore by placing the mouse pointer over a database object in the pgAdmin Object Browser window, and then clicking the secondary mouse button. If pgAdmin has the capability to backup or restore the particular database object, you will see the menu options Backup or Restore in the object menu such as shown for the emp table in the following:



Depending upon the type of database object you chose as well as the backup file format, a number of options may be available:

- Dump or restore the schema only (table, view, and sequence definitions, constraints, triggers, and functions), not the table data. (If you are using Postgres Plus Advanced Server, SPL functions, procedures, triggers, and packages can also be backed up and restored.)
- Dump or restore the table data only, not the schema.
- Dump database objects belonging to a selected schema.

- Dump a selected table or restore data to a selected table.
- Allow the restore operation to create a new database with the same name as the database from which the backup was created, and restore the database objects into this newly created database.
- Restore database objects into any existing database.
- Retain ownership of restored database objects using the same role names that owned the objects when the backup was created.
- Assign the role name of the user running the restore operation as the owner of all restored database objects.

**Note:** The preceding options are not available for all archive formats using pgAdmin. These options plus additional capabilities are available by running `pg_dump` or `pg_restore` from the command line.

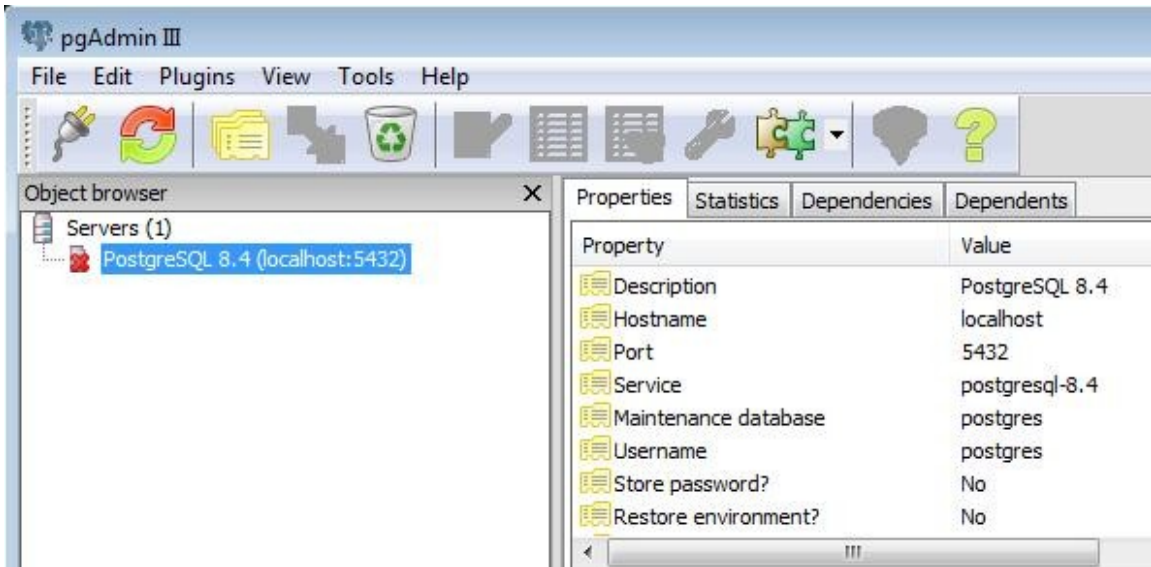
The instructions that follow illustrate a common scenario where you want to back up the entire contents of a database, and then at a later point in time, you want to recreate the entire database from the backup file.

The screen captures illustrate a Microsoft Windows<sup>®</sup> system, though the directions apply equally to other operating systems.

### ***Creating a Database Backup in a Plain-Text Backup File***

**Step 1:** Open pgAdmin (or Postgres Studio) from the Postgres Plus menu found on your operating system's application menu.

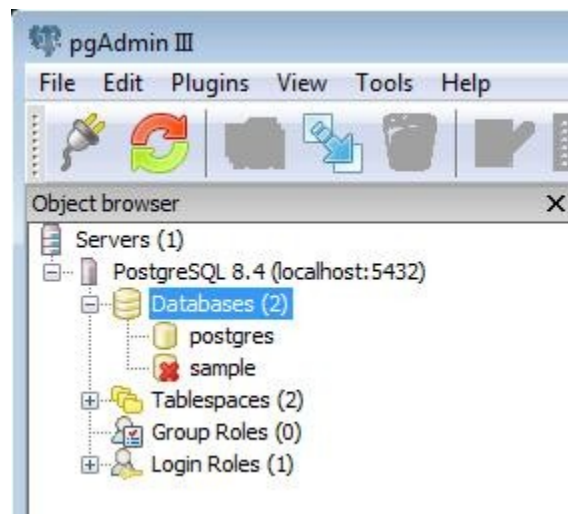
**Step 2:** Click on the Server node that contains the database that you want to back up. Be sure that the username that appears in the Username field of the Properties tab is a superuser.



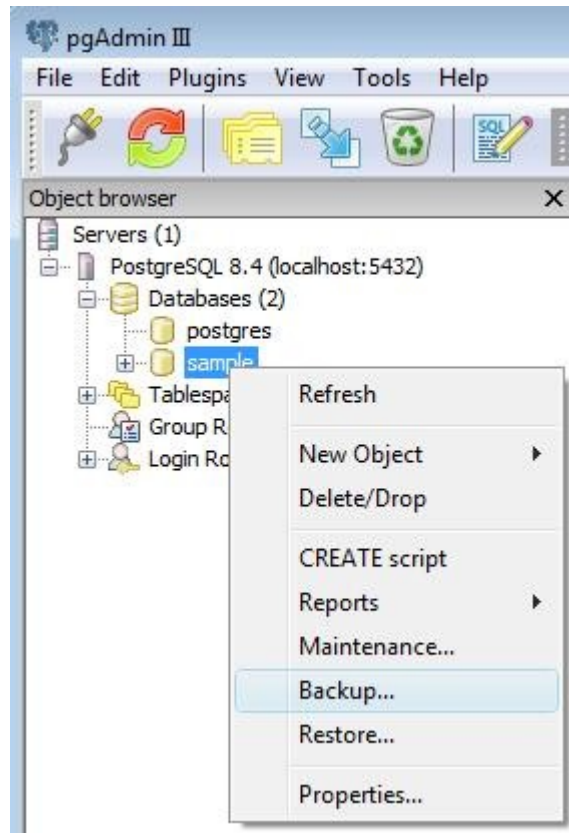
For Standard Server, the superuser, `postgres`, created during installation, should appear in the Username field. (For Advanced Server, the superuser is `enterprisedb`.)

**Note:** To change the username to a superuser, click the secondary mouse button on the Server node. In the menu that appears, click Properties. Change the Username field in the Server Properties dialog box. For an example of setting the server properties, see [Connect to Server](#) in Chapter “Using pgAdmin III” in [Postgres Studio \(pgAdmin\)](#) of the PostgreSQL Core Documentation found on the [Postgres Plus documentation web page](#).

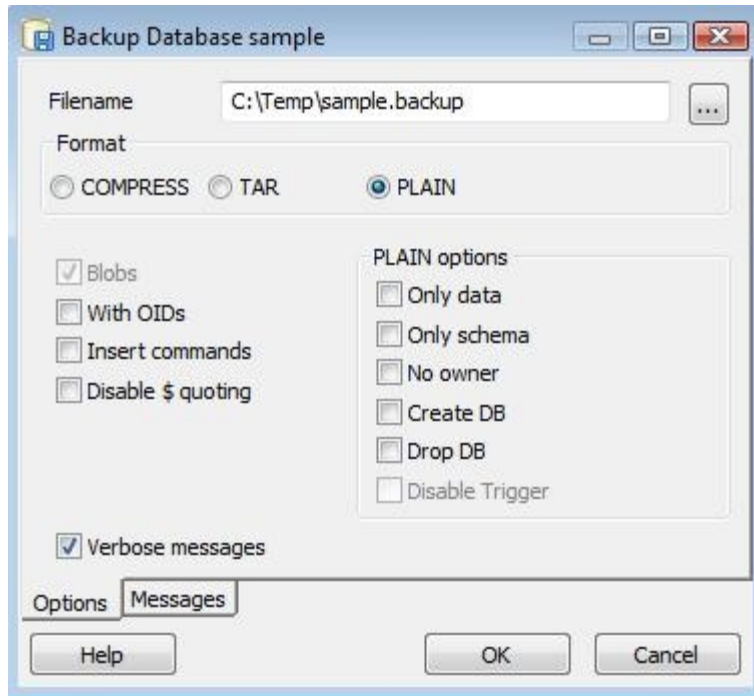
**Step 3:** Double-click the left mouse button on the Server node to connect to the server, and then double-click on the Databases node to expand the list of databases.



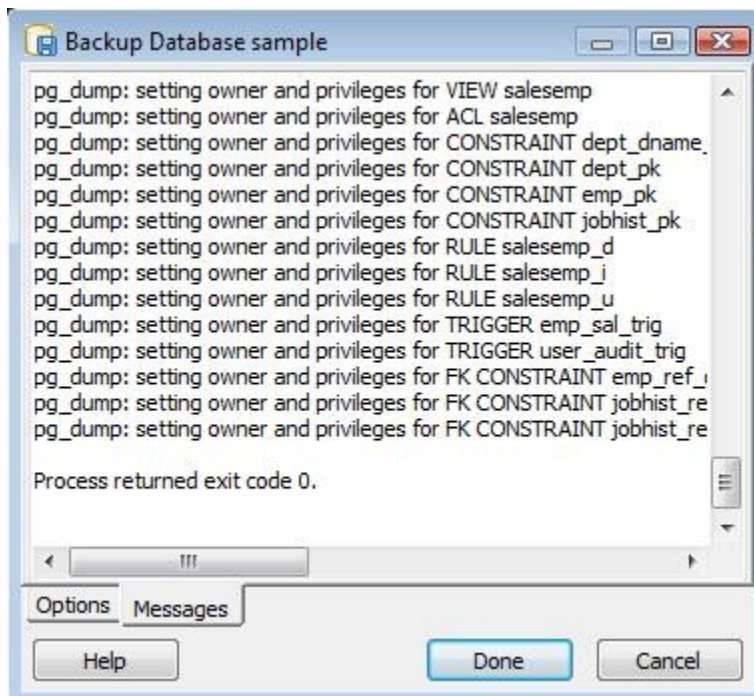
**Step 4:** Click the secondary mouse button on the database you want to back up. The Database menu appears.



**Step 5:** Click Backup in the Database menu. The Backup Database dialog box appears.

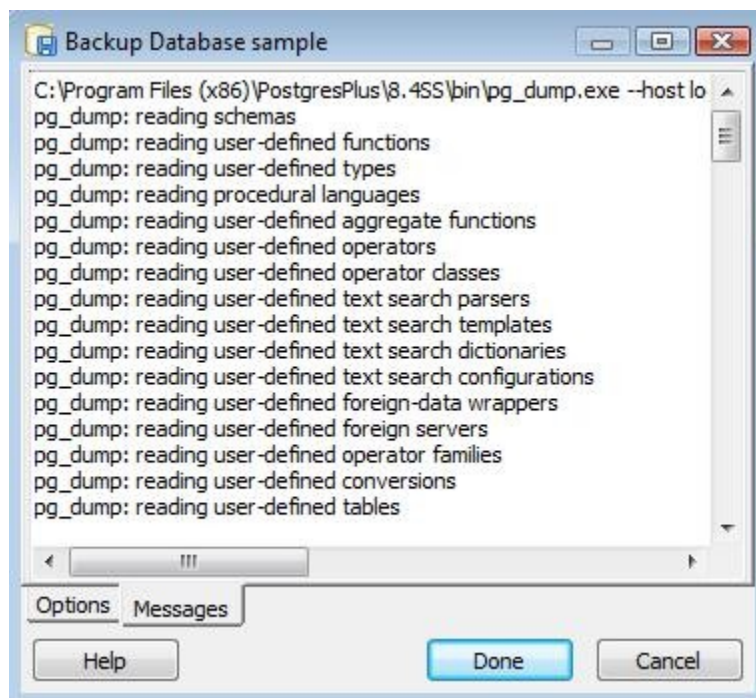


**Step 6:** In the Backup Database dialog box, enter the path and a file name in which you want the backup to be stored. Choose the PLAIN option. Leave the check boxes under PLAIN Options unselected. Click the OK button.



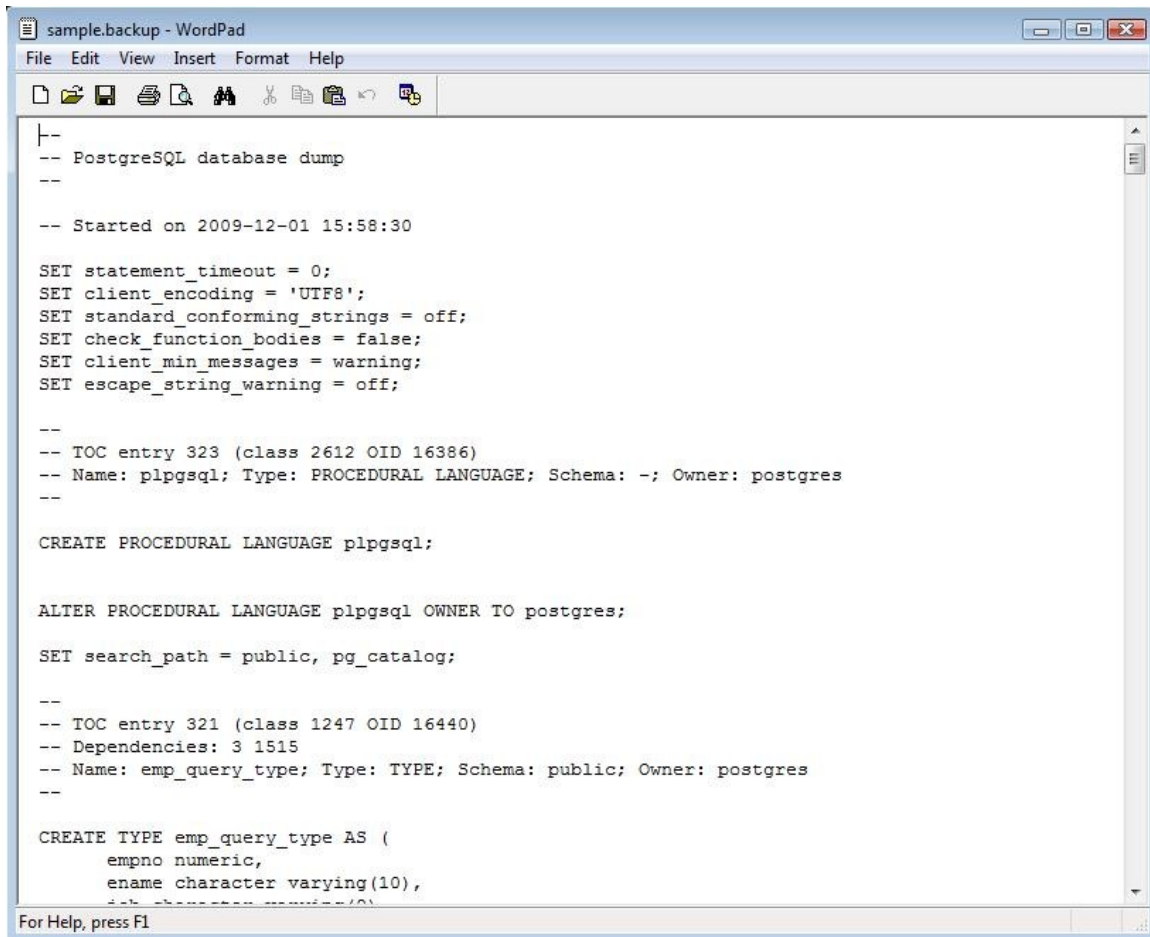
**Step 7:** If the backup operation ran successfully, `Process` returned exit code 0 appears at the bottom of the Messages window. If an exit code other than 0 appears, your backup file will not have been properly created. Scroll up the Messages window to find the problem. When you have identified the problem, click the Cancel button, correct the problem, and repeat the process from Step 4.

If you scroll to the top of the Messages window, you will see the `pg_dump` command that pgAdmin generated and executed.



**Step 8:** Click the Done button when you are finished viewing the Messages window.

You have just created a backup of the `sample` database to a plain-text backup file named `sample.backup`. You can view the `sample.backup` file with a text editor as shown by the following:



```
sample.backup - WordPad
File Edit View Insert Format Help

|--
-- PostgreSQL database dump
--

-- Started on 2009-12-01 15:58:30

SET statement_timeout = 0;
SET client_encoding = 'UTF8';
SET standard_conforming_strings = off;
SET check_function_bodies = false;
SET client_min_messages = warning;
SET escape_string_warning = off;

--
-- TOC entry 323 (class 2612 OID 16386)
-- Name: plpgsql; Type: PROCEDURAL LANGUAGE; Schema: -; Owner: postgres
--

CREATE PROCEDURAL LANGUAGE plpgsql;

ALTER PROCEDURAL LANGUAGE plpgsql OWNER TO postgres;

SET search_path = public, pg_catalog;

--
-- TOC entry 321 (class 1247 OID 16440)
-- Dependencies: 3 1515
-- Name: emp_query_type; Type: TYPE; Schema: public; Owner: postgres
--

CREATE TYPE emp_query_type AS (
    empno numeric,
    ename character varying(10),
    job character varying(10)
);

For Help, press F1
```

## Restoring a Database From a Plain-Text Backup File

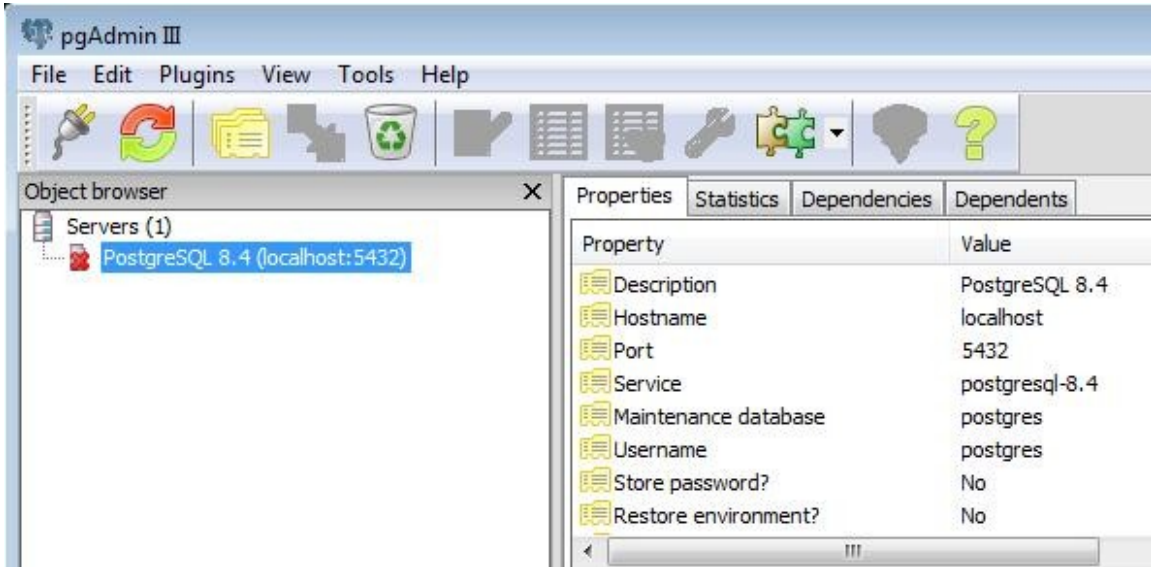
First, a new database will be created using pgAdmin.

Then, a plain-text backup file will be restored into this new database using the SQL command line terminal program `psql`. If you are using Postgres Plus Advanced Server, the equivalent command line terminal program `edb-psql` is used.

The plain-text backup file, `sample.backup`, created from the `sample` database in the preceding example will be used to restore all of its database objects into a new database named `new_sample`.

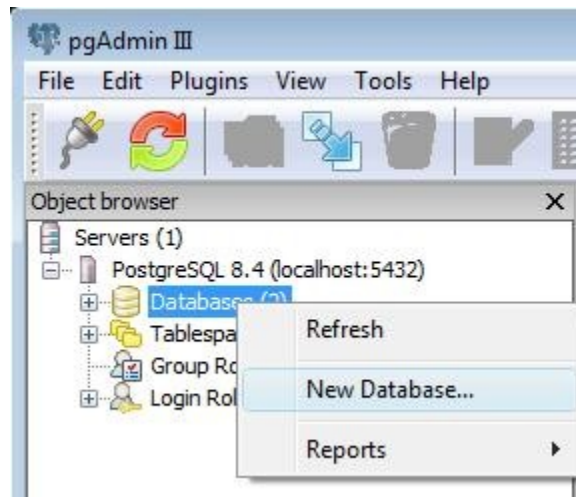
**Step 1:** Open pgAdmin (or Postgres Studio for Advanced Server) from the Postgres Plus menu found on your operating system's application menu.

**Step 2:** In pgAdmin, click on the Server node in which you want to create a new database to which the backup file will be restored. Be sure that the username that appears in the Username field of the Properties tab is a superuser.



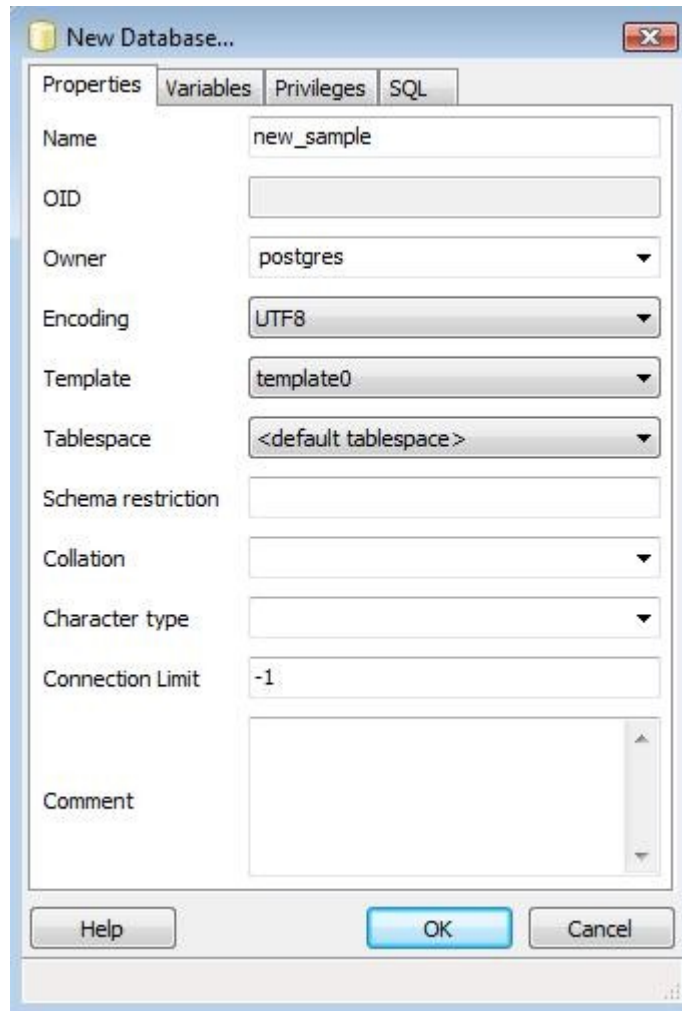
Follow the instructions for Step 2 in Creating a Database Backup in a Plain-Text Backup File to change Username to a superuser if necessary.

**Step 3:** Double-click the left mouse button on the Server node to connect to the server, and then click the secondary mouse button on the Databases node. The Databases menu appears.



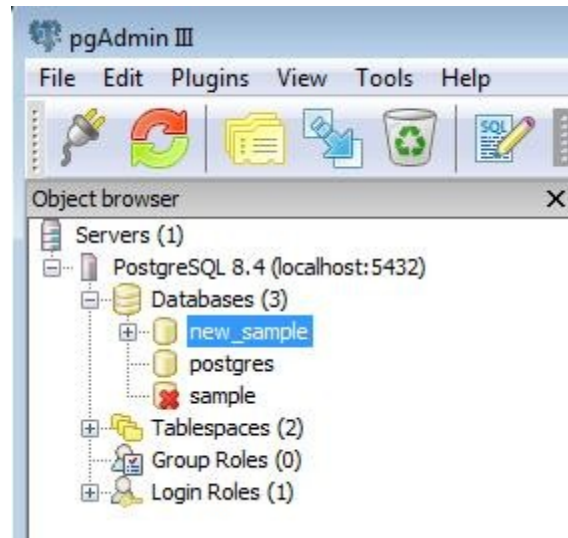
**Step 4:** Click New Database in the Databases menu. The New Database dialog box

appears.



**Step 5:** In the New Database dialog box, enter the name for your new database and select the database owner from the drop-down list. **For the Template field, be sure you select template0 from the drop-down list.** Click the OK button.

If you expand the database list, you should see a Database node for your new database.

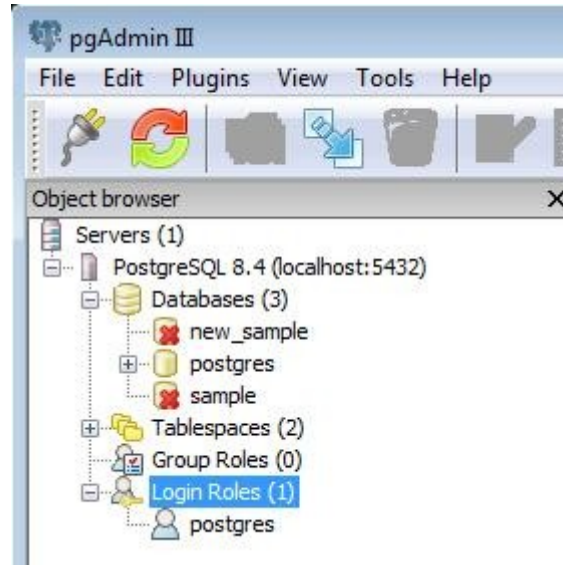


**Step 6:** If you are restoring into a different server than the one from which the backup file was created, or if you have deleted roles from your server, be sure that all role names that owned database objects when the backup file was created exist in the server into which you want to restore the backup file.

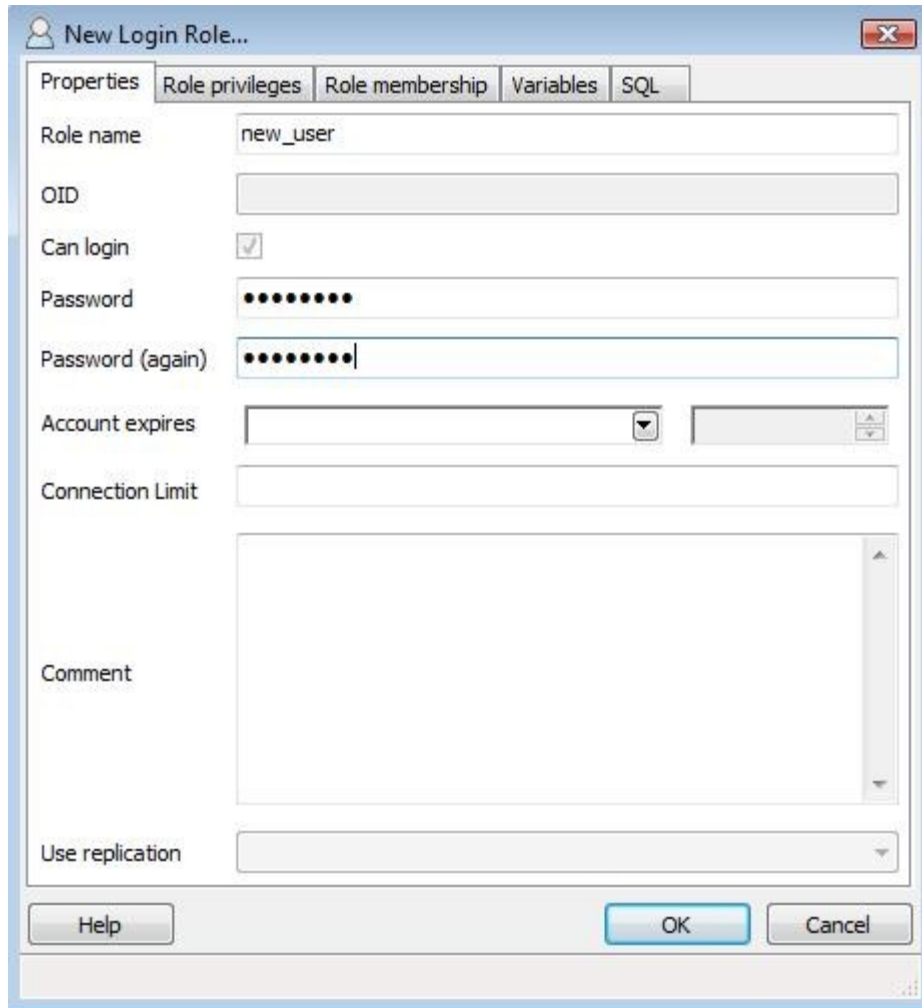
**Note:** If you do not know what roles owned database objects when the backup file was created, you can scan the backup file using a text editor for `ALTER object OWNER TO role` statements, some examples of which are shown by the following:

```
ALTER PROCEDURAL LANGUAGE plpgsql OWNER TO postgres;  
ALTER TYPE public.emp_query_type OWNER TO postgres;  
ALTER FUNCTION public.emp_comp(p_sal numeric, p_comm numeric) OWNER TO postgres;
```

You can list the roles that currently exist in a server by double-clicking on the Group Roles node and the Login Roles node:



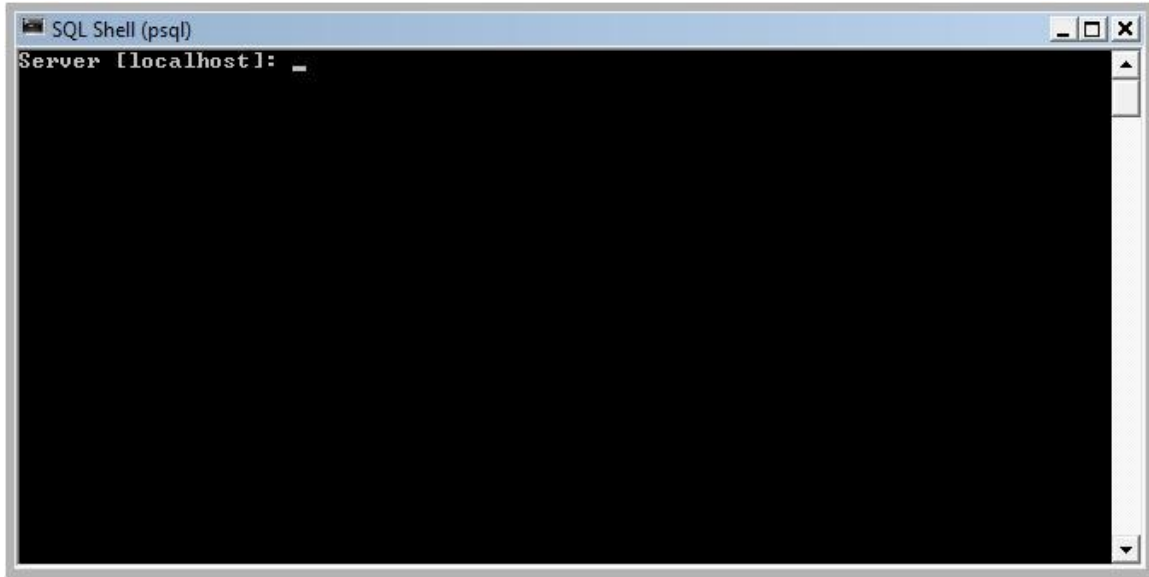
If you need to create new roles, click the secondary mouse button on the Login Roles node or the Group Roles node. From the menu that appears, click New Login Role or New Group Role and fill in the dialog box. Click the OK button when you are done.



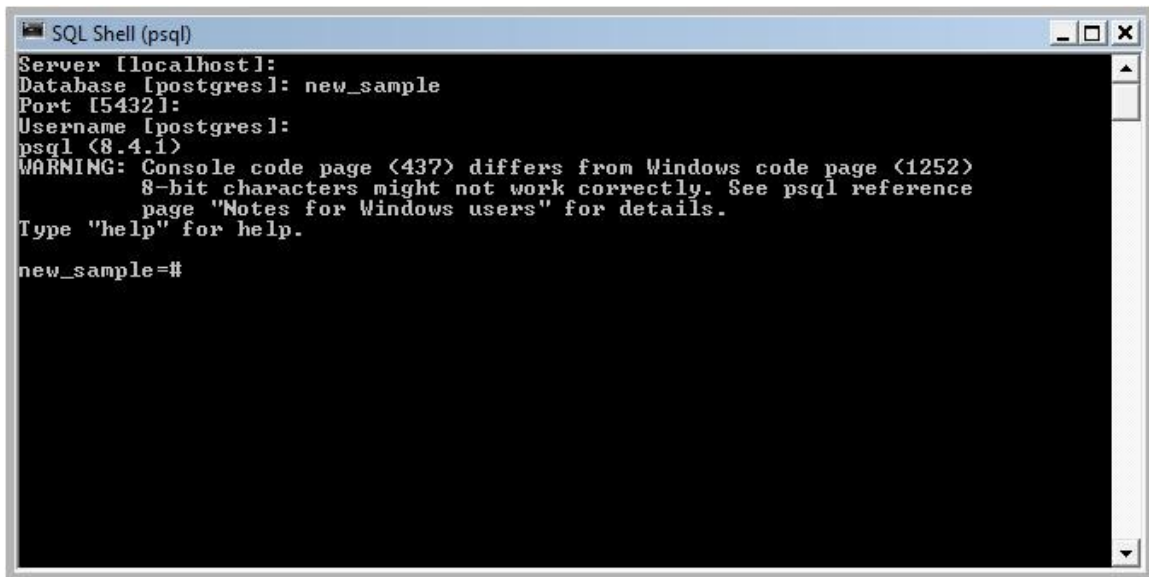
**Note:** If the original owner’s role name of a database object does not exist in the server into which you are restoring, an error message will be displayed when the ALTER statement cannot assign the ownership. The database object will end up being assigned to the role with which you are logged in when you perform the restore operation.

**Step 7:** Open the SQL command line terminal called SQL Shell (psql) on the Postgres Plus Standard Server submenu.

**Note:** If you are using Postgres Plus Advanced Server, the submenu option is called Run SQL Command Line. Click Run SQL Command Line to open another submenu with the choices EDB\*Plus and EDB-PSQL. Open EDB-PSQL.



**Step 8:** Enter the server connection information in response to the prompts. **Be sure to specify the name of the new database in which you want to restore the backup file in response to the Database prompt.**



**Step 9:** Run the `psql` command `\i` with the path to the plain-text backup file.

```

SQL Shell (psql)
Server [localhost]:
Database [postgres]: new_sample
Port [5432]:
Username [postgres]:
psql (8.4.1)
WARNING: Console code page (437) differs from Windows code page (1252)
         8-bit characters might not work correctly. See psql reference
         page "Notes for Windows users" for details.
Type "help" for help.

new_sample=# \i C:/Temp/sample.backup
    
```

**Note:** On Windows systems, you must use a forward slash (/) to separate the directory names in the path to the backup file when using the \i command.

You have just recreated the database objects in the new\_sample database from the plain-text backup file named sample.backup.

```

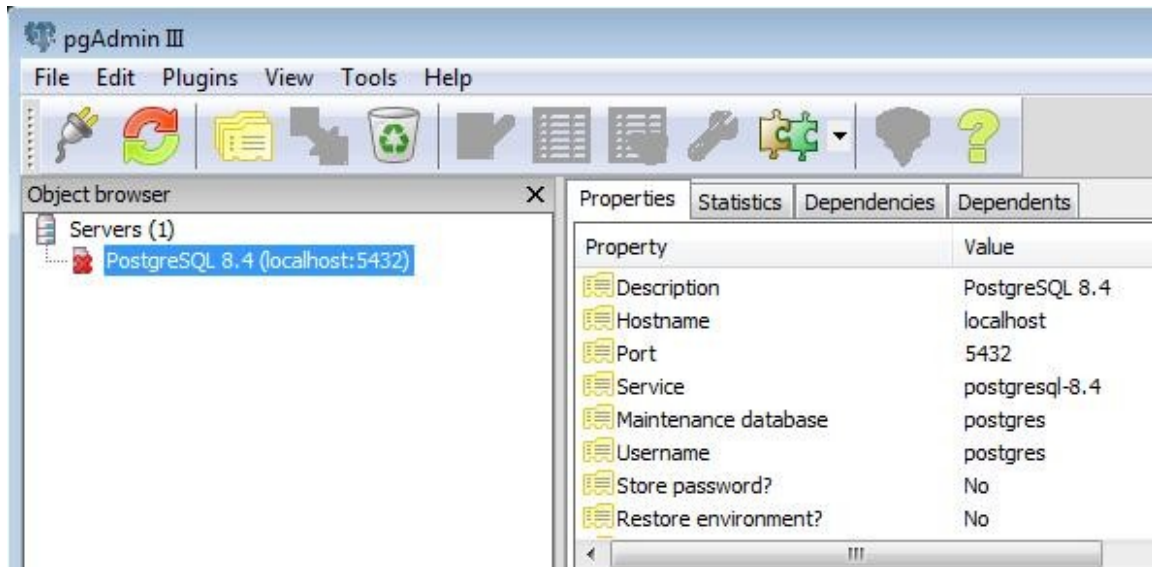
SQL Shell (psql)
GRANT
GRANT
REVOKE
REVOKE
GRANT
GRANT
REVOKE
REVOKE
GRANT
GRANT
REVOKE
REVOKE
GRANT
GRANT
new_sample=# \dt
      List of relations
Schema | Name   | Type  | Owner
-----+-----+-----+-----
public | dept   | table | postgres
public | emp    | table | postgres
public | jobhist | table | postgres
(3 rows)

new_sample=#
    
```

## Creating a Database Backup in a Custom Archive Backup File

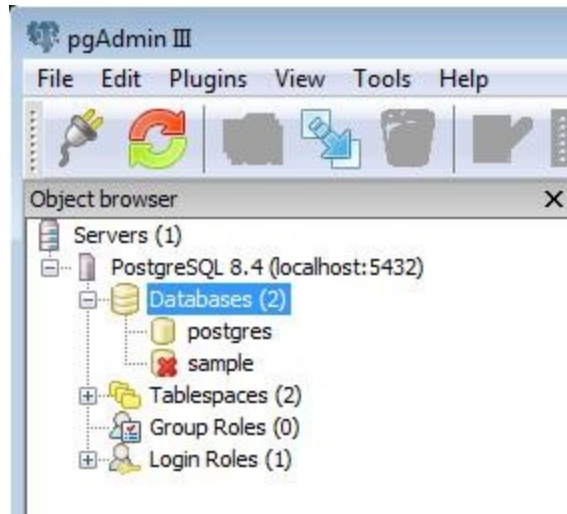
**Step 1:** Open pgAdmin (or Postgres Studio) from the Postgres Plus menu found on your operating system's application menu.

**Step 2:** Click on the Server node that contains the database that you want to back up. Be sure that the username that appears in the Username field of the Properties tab is a superuser.

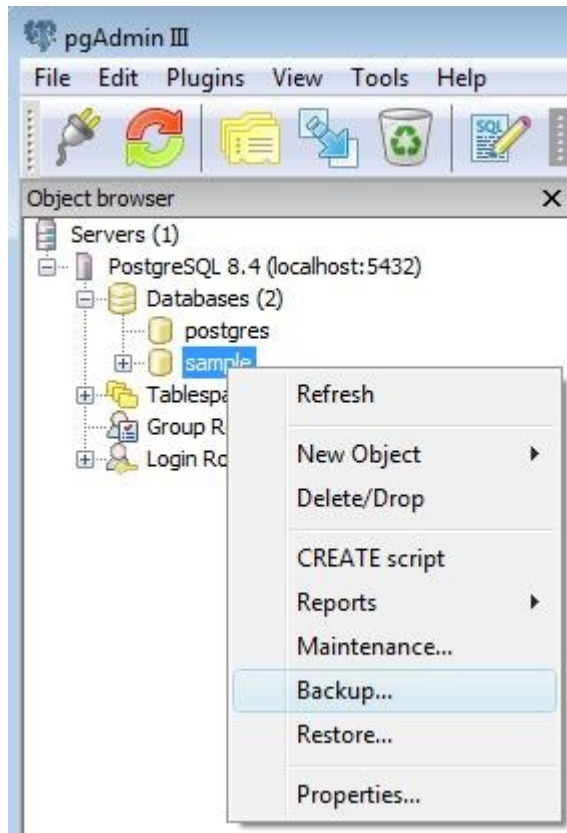


Follow the instructions for Step 2 in [Creating a Database Backup in a Plain-Text Backup File](#) to change Username to a superuser if necessary.

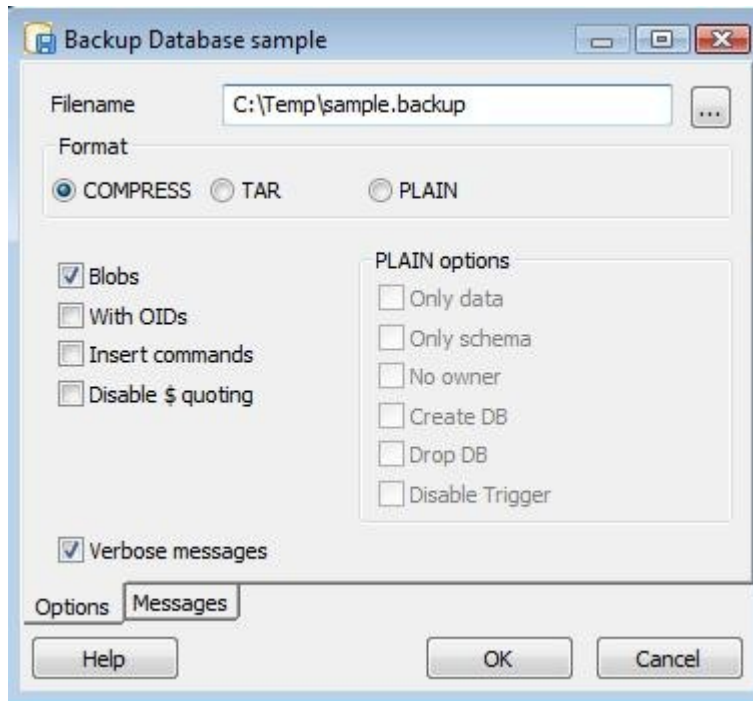
**Step 3:** Double-click the left mouse button on the Server node to connect to the server, and then double-click on the Databases node to expand the list of databases.



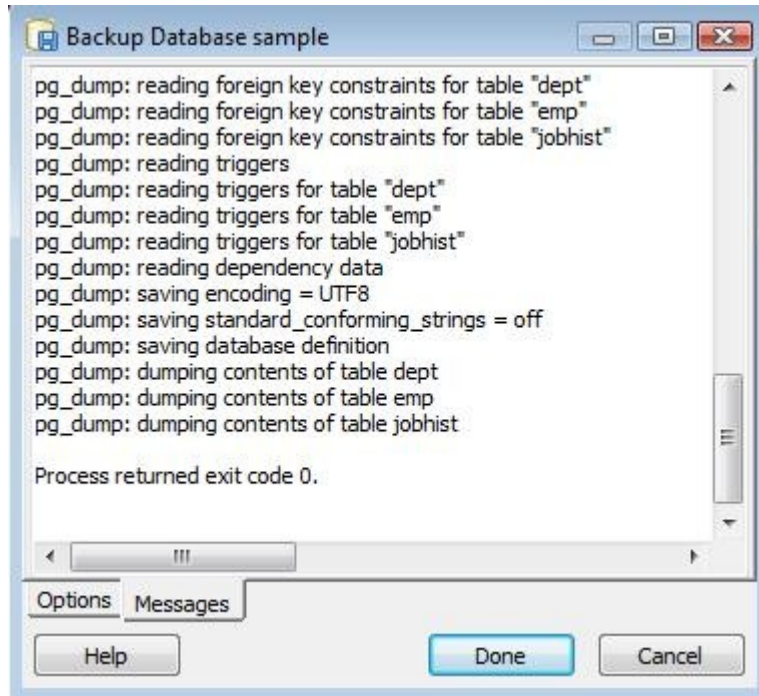
**Step 4:** Click the secondary mouse button on the database you want to back up. The Database menu appears.



**Step 5:** Click Backup in the Database menu. The Backup Database dialog box appears.

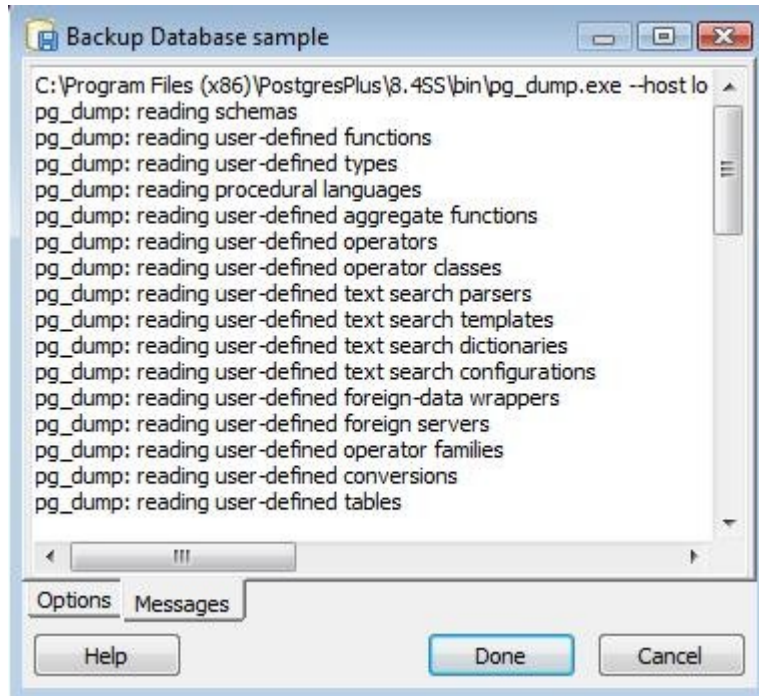


**Step 6:** In the Backup Database dialog box, enter the path and a file name in which you want the backup to be stored. Choose the COMPRESS option. Leave the check boxes under COMPRESS unselected except for Blobs if you want to back up large object data. Click the OK button.



**Step 7:** If the backup operation ran successfully, `Process returned exit code 0` appears at the bottom of the Messages window. If an exit code other than 0 appears, your backup file will not have been properly created. Scroll up the Messages window to find the problem. When you have identified the problem, click the Cancel button, correct the problem, and repeat the process from Step 4.

If you scroll to the top of the Messages window, you will see the `pg_dump` command that pgAdmin generated and executed.



**Step 8:** Click the Done button when you are finished viewing the Messages window.

You have just created a backup of the `sample` database to a custom archive backup file named `sample.backup`.

## ***Restoring a Database From a Custom Archive Backup File***

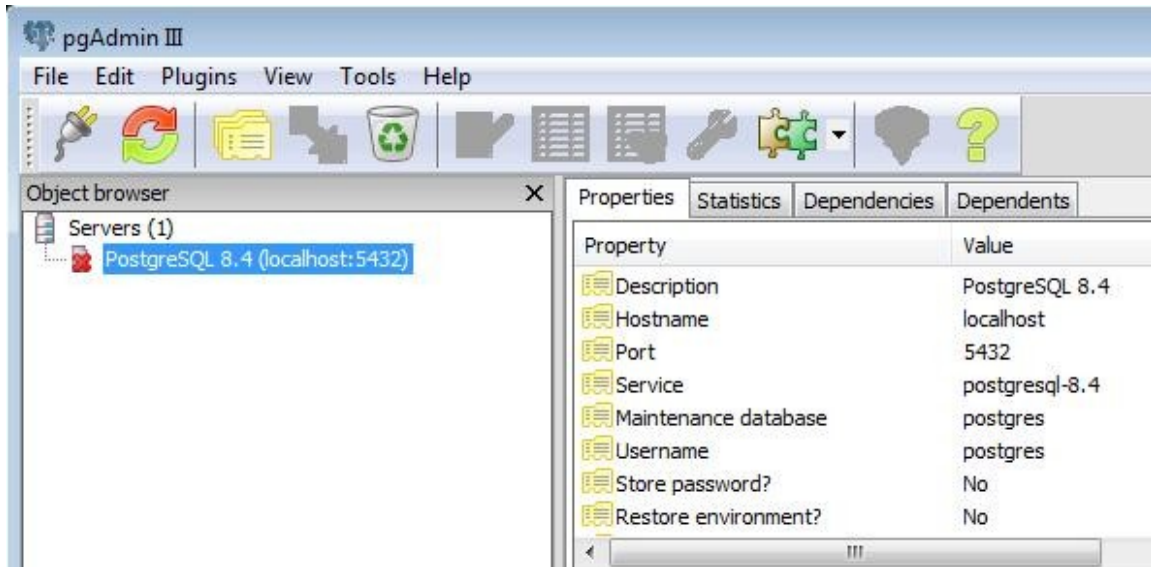
First, a new database will be created using pgAdmin.

Then, a custom archive backup file will be restored into this new database using pgAdmin.

The custom archive backup file, `sample.backup`, created from the `sample` database in the preceding example will be used to restore all of its database objects into a new database named `new_sample`.

**Step 1:** Open pgAdmin (or Postgres Studio for Advanced Server) from the Postgres Plus menu found on your operating system's application menu.

**Step 2:** In pgAdmin, click on the Server node in which you want to create a new database to which the backup file will be restored. Be sure that the username that appears in the Username field of the Properties tab is a superuser.



Follow the instructions for Step 2 in Creating a Database Backup in a Plain-Text Backup File to change Username to a superuser if necessary.

**Step 3:** Create a new database by following the instructions for steps 3 thru 5 in Restoring a Database From a Plain-Text Backup File.

**Step 4:** If you are restoring into a different server than the one from which the backup file was created, or if you have deleted roles from your server, be sure that all role names that owned database objects when the backup file was created exist in the server into which you want to restore the backup file.

**Note:** If you do not know what roles owned database objects when the backup file was created, you can run `pg_restore` from the command line to generate a SQL text version of the backup from the custom archive backup file. To accomplish this, run the `pg_restore` program giving the backup file as the only parameter. You can then scan the text for `ALTER object OWNER TO role` statements.

This method is shown in the following example:

```
>cd C:\Program Files\PostgresPlus\8.4SS\bin

>pg_restore C:\Temp\sample.backup | find "OWNER TO"
ALTER PROCEDURAL LANGUAGE plpgsql OWNER TO postgres;
ALTER TYPE public.emp_query_type OWNER TO postgres;
ALTER FUNCTION public.emp_comp(p_sal numeric, p_comm numeric) OWNER TO
postgres;
.
.
.
```

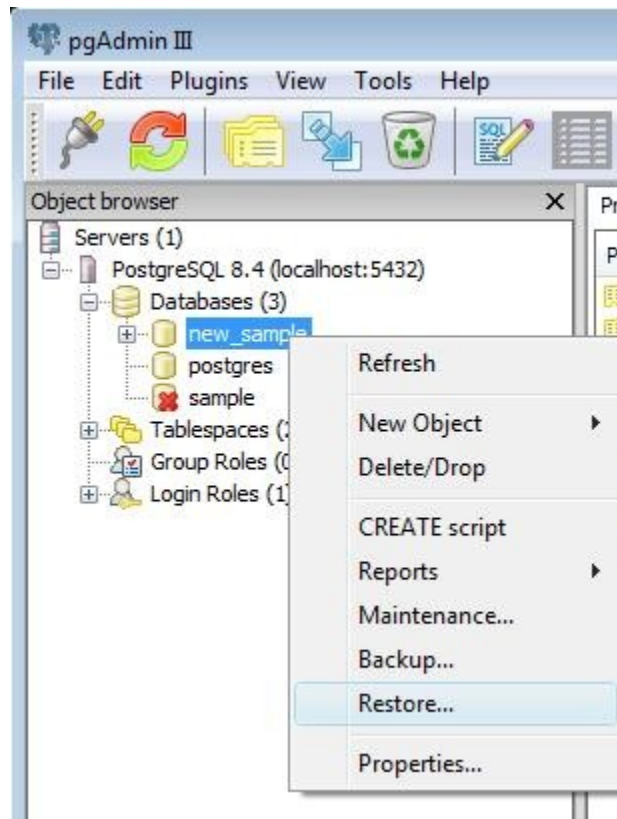
The identical operation performed on a Linux system appears as follows:

```
$ cd /opt/PostgresPlus/8.4SS/bin
$ ./pg_restore /home/user/sample.backup | grep 'OWNER TO'
ALTER PROCEDURAL LANGUAGE plpgsql OWNER TO postgres;
ALTER TYPE public.emp_query_type OWNER TO postgres;
ALTER FUNCTION public.emp_comp(p_sal numeric, p_comm numeric) OWNER TO
postgres;
.
.
.
```

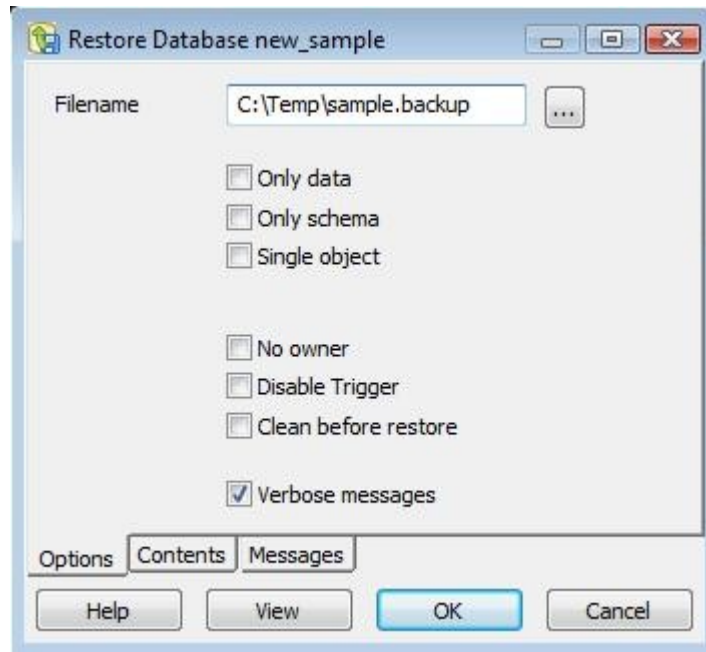
You can list the roles that currently exist in a server and create new roles if necessary by following the directions in Step 6 of Restoring a Database From a Plain-Text Backup File.

**Note:** If the original owner’s role name of a database object does not exist in the server into which you are restoring, an error message will be displayed when the ALTER statement cannot assign the ownership. The database object will end up being assigned to the role with which you are logged in when you perform the restore operation.

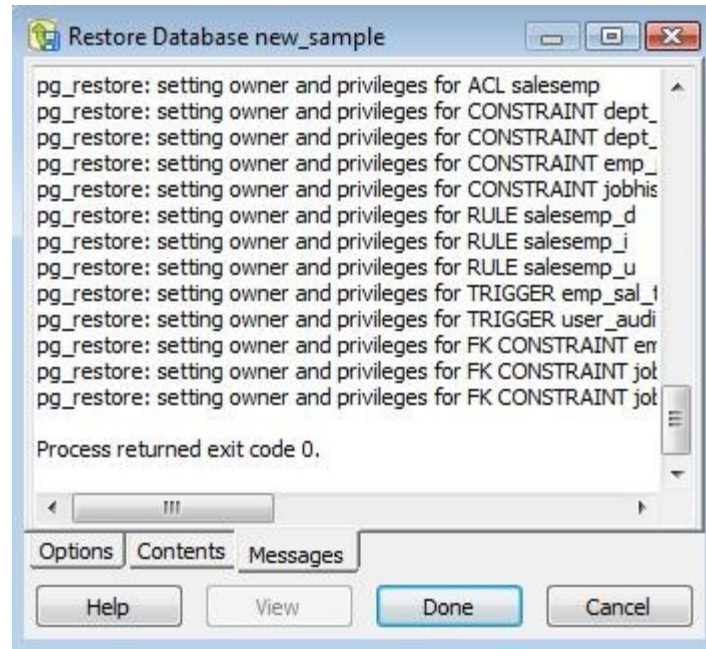
**Step 5:** Click the secondary mouse button on the new database to which you want to restore the backup file. The Database menu appears.



**Step 6:** Click Restore in the Database menu. The Restore Database dialog box appears.

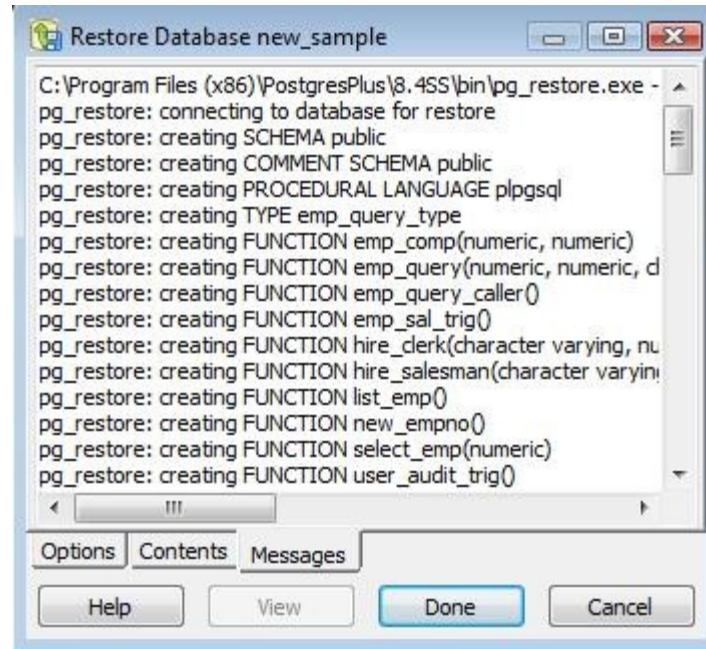


**Step 7:** In the Restore Database dialog box, enter the path to the backup file to be restored. Leave the other check boxes unselected except for Verbose Messages. Click the OK button.



**Step 8:** If the restore operation ran successfully, `Process returned exit code 0` appears at the bottom of the Messages window. If an exit code other than 0 appears, your restore operation may not have completed successfully. Scroll up the Messages window to find the problem. After you have identified the problem, click the Cancel button to close the Restore Database dialog box. Check which database objects have been restored using the pgAdmin Object Browser window. If necessary, correct the problem and repeat the process from Step 3.

If you scroll to the top of the Messages window, you will see the `pg_restore` command that pgAdmin generated and executed.



**Step 9:** You have just recreated the database objects in the `new_sample` database from the custom archive backup file named `sample.backup`. Click the Done button when you are finished viewing the Messages window.

## Conclusion

In this Quick Tutorial you learned how to perform the basic operations of backing up and restoring a Postgres Plus database using the pgAdmin or Postgres Studio database administration console.

You should now be able to proceed confidently with a Technical Evaluation of Postgres Plus. Using the backup and restore features will allow you to make backups of the different stages of your work and restore them as needed.

The following resources should help you move on with this step:

- [Postgres Plus Technical Evaluation Guide](#)
- [Postgres Plus Getting Started resources](#)
- [Postgres Plus Quick Tutorials](#)
- [Postgres Plus User Forums](#)
- [Postgres Plus Documentation](#)
- [Postgres Plus Webinars](#)