PostgreSQL Database Cluster Inventory

Introduction

A tool to inventory a database is always useful for administrations tasks, backup planning and monitoring, migrations, capacity management, forecasts, releasing new applications' versions into Production, code management of database objects, etc.

I have recently needed a tool to retrieve information about objects, database setup and space of a PostgreSQL database cluster. I couldn't find a script to dump all the information, so I developed mine. I share it with you on this post.

My inventory tool includes information about:

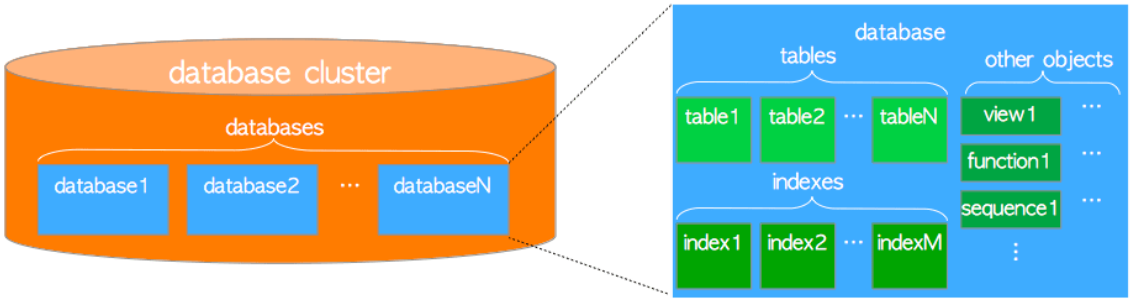
- Cluster's information
- List of databases
- List of schemas
- List of foreign servers
- Space
- Objects' DDLs
- Permissions

PostgreSQL Architecture

To make an inventory script and read the output, we must understand at a high-level the PostgreSQL Architecture. I make a brief description in this section as a support to the scripts I developed to inventory a PostgreSQL database.

A PostgreSQL server runs on a single host and manages a single database cluster. A **database cluster** is a collection of **databases** managed by a PostgreSQL server.

The figure below shows the logical structure of a database cluster. A **database** is a collection of **database objects**. A database object is a data structure used either to store or to reference data, as indexes, sequences, views, functions, etc.



http://www.interdb.jp/pg/img/fig-1-01.png

In PostgreSQL, databases themselves are also database objects and are logically separated from each other. All other database objects (e.g., tables, indexes, etc.) belong to their respective databases.

Users and groups of users are shared across the entire cluster, but no other data. That's to say, any given client connection to the server can access only the data in a single database, the one specified in the connection request.

A database contains one or more **schemas**, which in turn contain tables, views, indexes, sequences, data types, operators, and functions. The same object name can be used in different schemas without conflict. Unlike databases, schemas are not rigidly separated: a user can access objects in any of the schemas in the database he is connected to if he has privileges to do so.

The reasons to use schemas are:

- To allow many users to use one database without interfering with each other,
- To organise database objects into logical groups to make them more manageable and/ or
- To connect third-party applications to separate schemas, so they do not collide with the names of other objects.

PostgreSQL creates a schema named *public* for every new database. Whatever object you create without specifying the schema name, PostgreSQL will automatically put it into this *public* schema.

Users can only access objects in the schemas that they own. It means they cannot access any object in the schemas, which does not belong to them. To enable users to access the objects in the schema that they do not own, you must grant the USAGE privilege to the users on the schema. To allow users to create objects in the schema that they do not own, you need to give them the CREATE privilege on the schema.

Environment

I have used a Bitnami's PostgreSQL virtual machine to develop the code in this post.
bitnami-postgresql-12.0.0-1-r01-linux-debian-9-x86_64-nami.ova

I use the information in the links below to power and set up the virtual machine to work:

- Connect To PostgreSQL From A Different Machine
- Enable Or Disable The SSH Server

Scripts

These scripts run on psql, a command-line native client for PostgreSQL. You must run the file PostgreSQLInventory.psql to get most of the information.

psql -d *your_database_here* -U *your_username_here* -W -f PostgreSQLInventory.psql

To gather space information, run the script PostgreSQLInventory_Space.psql.

psql -d *your_database_here* -U *your_username_here* -W -f PostgreSQLInventory_Space.psql

If you need to get the detailed information of the structure of the relations, run the script PostgreSQLInventory_DDLs.psql.

psql -d *your_database_here* -U *your_username_here* -W -f PostgreSQLInventory_DDLs.psql

Connect with a user with reading access to the schema pg_catalog in all databases in the cluster.