Universidad Nacional Autónoma de México

Facultad de Ingeniería

Asignatura: Bases de Datos (1644)

Profesor: Ing. Fernando Arreola Franco

Semestre 2026-2

"Tarea 2"

Grupo: 1

Alumna: Bautista Reyes Sofía

No. de cuenta: 321103891

Fecha de entrega: 16 de febrero de 2026

Tarea 2

Investigar:

¿Qué requiero para conectarme a una BD?

Antes de establecer una conexión con PostgreSQL es necesario verificar algunos elementos básicos, primero, que el sistema esté instalado en el equipo o que se tenga acceso a un servidor remoto donde ya esté funcionando. También se necesitan las credenciales de acceso: la dirección del servidor (host o IP), el puerto (que normalmente es el 5432), el nombre de usuario, la contraseña y el nombre de la base de datos a la que se desea entrar.

La conexión puede realizarse mediante la herramienta de consola de PostgreSQL, llamada psql, y para usarla, se abre la terminal y después, se ejecuta el comando correspondiente en psql indicando los datos de conexión, con esto se accede a la base de datos y se empiezan a realizar consultas o tareas de administración.

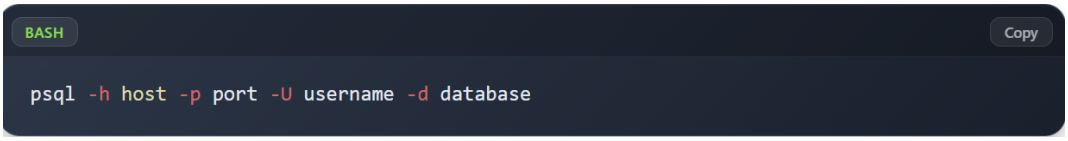

Figura 1: Sintaxis para realizar conexión a la base de datos.

Los significados de lo que se puede ver en la Figura 1 son los siguientes:

- -h host: Indica la dirección del servidor al que se realizará la conexión, ya sea local (localhost) o remoto mediante una IP.
- -p puerto: Señala el puerto donde se encuentra activo PostgreSQL, que normalmente es el 5432.
- -U nombre de usuario: Define el usuario con el que se intentará acceder a PostgreSQL.
- -d base de datos: Especifica la base de datos a la que se desea entrar.

Por ejemplo, tenemos una base de datos que se llama 'mydb' en una máquina local, y el nombre de usuario es postgres, entonces la sintaxis quedaría así:

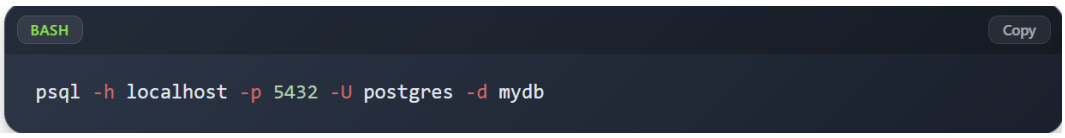

Figura 2: Ejemplo de conexión a base de datos.

Y una vez hecho lo que se ve en la Figura 2, Se pedirá que introduzca la contraseña del usuario postgres [1].

Permisos a nivel sistema y objeto.

A nivel sistema:
Los permisos a nivel sistema son aquellos que afectan al servidor de bases de datos en general, no solo a una tabla u objeto específico, y permiten controlar quién puede crear bases de datos, usuarios o conectarse al sistema. En PostgreSQL estos permisos se asignan principalmente a través de roles y sus atributos.

Un rol puede tener capacidades especiales que le dan control sobre el sistema completo, entre los permisos más importantes están:

- LOGIN: permite que el rol se conecte al servidor de base de datos.
- SUPERUSER: otorga control total, ya que ignora todas las verificaciones de permisos.
- CREATEDB: permite crear nuevas bases de datos.
- CREATEROLE: permite crear, modificar o eliminar otros roles.
- REPLICATION: autoriza conexiones para replicación de datos.
- BYPASSRLS: permite omitir las políticas de seguridad a nivel de fila.
- CONNECTION LIMIT: define cuántas conexiones simultáneas puede tener un rol.

También existen permisos generales como:

- CONNECT: permite conectarse a una base de datos.
- CREATE: permite crear esquemas dentro de la base de datos.
- TEMP: permite crear tablas temporales.

Y en conjunto estos permisos se clasifican como de nivel sistema porque controlan acciones globales del servidor y la administración general de la base de datos [2].

A nivel objetos:
Cuando se crea un objeto en PostgreSQL, se le asigna un propietario, que normalmente es el rol que ejecutó la instrucción de creación. Al inicio, solo ese propietario (o un superusuario) tiene control total sobre el objeto y si se quiere que otros roles puedan usarlo, es necesario otorgarles permisos. Existen distintos tipos de privilegios, como:

- SELECT: permite leer o consultar datos de una tabla o vista.
- INSERT: permite agregar nuevos registros.
- UPDATE: permite modificar registros existentes.
- DELETE: permite eliminar registros.

- CREATE: permite crear objetos dentro de una base de datos o esquema (tablas, vistas, etc.).
- CONNECT: permite conectarse a una base de datos.
- EXECUTE: permite ejecutar funciones o procedimientos almacenados.
- USAGE: permite usar ciertos objetos como esquemas, secuencias o tipos de datos, sin modificarlos.
- Entre otros privilegios según el tipo de objeto.

Cada uno permite realizar acciones específicas y su aplicación depende del tipo de objeto, por ejemplo, una tabla, una función o un esquema.

El propietario siempre conserva el derecho de modificar o eliminar el objeto; este derecho no se puede otorgar ni quitar directamente, aunque puede heredarse dentro de un rol.

También es posible cambiar el propietario de un objeto mediante el comando ALTER, lo que transfiere la propiedad y los privilegios al nuevo dueño.

```
ALTER TABLE table_name OWNER TO new_owner;
```

Figura 3: Ejemplo de iso de comando ALTER.

Los superusuarios pueden hacerlo sin restricciones, mientras que un rol común solo puede cambiar la propiedad si es el dueño actual y tiene permisos para asignar el nuevo rol propietario [3], [4].

¿Cómo dar/quitar permisos?

Los gestores de bases de datos relacionales, como PostgreSQL, incluyen comandos que permiten administrar el acceso a la información, y estos comandos forman parte del Lenguaje de Control de Datos (DCL) del SQL, estos se utilizan para asignar o retirar permisos sobre los distintos objetos de una base de datos.

Dos instrucciones fundamentales para este control son GRANT y REVOKE y con ellas, el administrador puede dar o quitar privilegios a uno o varios roles.

GRANT se utiliza para otorgar permisos específicos, ya que permite asignar privilegios sobre distintos objetos, como tablas, columnas, vistas, funciones, esquemas o incluso la base de datos completa. También puede emplearse para dar permisos a un rol en particular o a varios roles al mismo tiempo y es posible aplicar permisos a todos los objetos de un mismo tipo dentro de un esquema.

Por otro lado, REVOKE sirve para retirar permisos que ya fueron concedidos, un rol solo puede revocar los privilegios que él mismo otorgó, y si un rol concede permisos

a otro y este a su vez los comparte con un tercero, se pueden eliminar usando opciones como CASCADE, que retiran el privilegio en cadena.

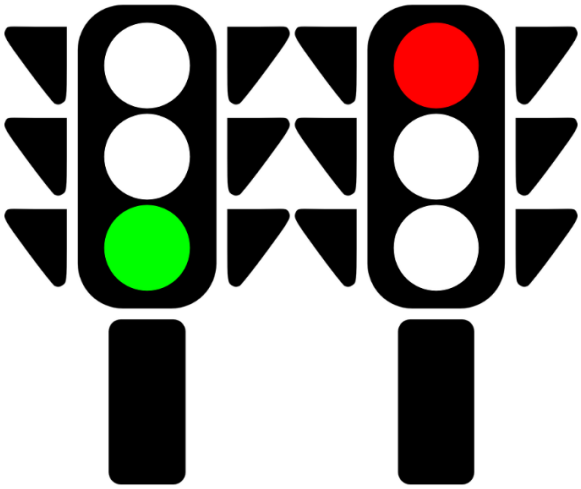

Figura 4: Los permisos se conceden con GRANT (verde) y se retiran con REVOKE (rojo).

Estos comandos permiten controlar de manera precisa quién puede acceder a los objetos de la base de datos y qué acciones puede realizar sobre ellos [5].

Diferencia entre rol y usuario.

Los conceptos de usuario y rol están muy relacionados, y esto se debe a que el sistema maneja todo como roles, solo que la diferencia principal depende de si ese rol puede iniciar sesión o no.

Un usuario es simplemente un rol que tiene permiso para conectarse a la base de datos, es decir, cuenta con credenciales de acceso (nombre y contraseña) y puede iniciar sesión para trabajar con la información. Mientras que un rol es un concepto más general que puede representar a un usuario individual, a un grupo de usuarios o a un conjunto de permisos.

Los roles pueden tener varios atributos:

- LOGIN: Permite que el rol pueda iniciar sesión y acceder a la base de datos.
- SUPERUSER: Otorga control total, sin restricciones de permisos.
- CREATEDB: Autoriza la creación de nuevas bases de datos.
- CREATEROLE: Permite crear y administrar otros roles.
- REPLICATION: Habilita conexiones destinadas a procesos de replicación.
- BYPASSRLS: Permite ignorar las políticas de seguridad a nivel de fila.

- CONNECTION LIMIT: Establece el número máximo de conexiones simultáneas para el rol [2].

Figura 5: Usuario vs Rol.

Los roles pueden asignarse entre sí, lo que facilita organizar y administrar los privilegios de manera más flexible dentro de la base de datos [6], [7].

Referencias

[1] AlexHost. Cómo conectarse a una base de datos postgresql. AlexHost. Consultado: feb. 2026. [Online]. Available: https://alexhost.com/es/faq/how-to-connect-to-a-postgresql-database/#1_Requisitos_previos_para_conectarse_a_PostgreSQL

[2] Cubepath Support Team. (2026, Jan.) Gestión de usuarios y permisos en postgresql: Guía completa de seguridad. Cubepath. Consultado: feb. 2026. [Online]. Available: https://cubepath.com/docs/gesti%C3%B3n-de-bases-de-datos/gestion-de-usuarios-y-permisos-en-postgresql

[3] The PostgreSQL Global Development Group. 5.8. privileges. PostgreSQL. Consultado: feb. 2026. [Online]. Available: https://www.postgresql.org/docs/current/ddl-priv.html

[4] J. Romanowski. (2025, Mar.) Los comandos más comunes de postgresql: una guía para principiantes. LearnSQL. Consultado: feb. 2026. [Online]. Available: https://learnsql.es/blog/los-comandos-mas-comunes-de-postgresql-una-guia-para-principiantes/

[5] J. Segovia. (2018, Nov.) Controlar permisos en PostgreSQL; GRANT & REVOKE. TodoPostgreSQL by Abatic. Consultado: feb. 2026. [Online]. Available: https://www.todopostgresql.com/controlar-permisos-en-postgresql-grant-revoke/

[6] J. Ayyalusamy. (2025, Jun.) Usuarios y roles de postgresql: una guía completa para el control de acceso. Medium. Consultado: feb. 2026. [Online]. Available: https://medium.com/@jramcloud1/postgresql-users-and-roles-explained-a-complete-guide-for-access-control-d80bdeb13d45

[7] J. Segovia. (2017, Sep.) Como crear usuarios en postgresql. TodoPostgreSQL by Abatic. Consultado: feb. 2026. [Online]. Available: https://www.todopostgresql.com/crear-usuarios-en-postgresql/