María Guadalupe Martínez Pavón Grupo 1
Bases de Datos

# Privilegios de PostgreSQL

Los privilegios determinan lo que alguien está autorizado a hacer con los datos y la base de datos. Debe asignar los privilegios según el tipo de trabajo que realiza la persona dentro de la organización.
Con PostgreSQL, se pueden crear usuarios y roles con permisos de acceso granulares. Al nuevo usuario o rol se les debe conceder selectivamente los permisos necesarios para cada objeto de base de datos. Esto da mucho poder al usuario final, pero al mismo tiempo, dificulta potencialmente el proceso de creación de usuarios y roles con los permisos correctos.

PostgreSQL permite conceder permisos directamente a los usuarios de la base de datos. Se recomienda crear varios roles con conjuntos específicos de permisos basados en los requisitos de aplicación y acceso. Posteriormente, podremos asignar el rol apropiado a cada usuario. Los roles deben utilizarse para aplicar un modelo de privilegios mínimos para acceder a objetos de base de datos. El usuario *postgres* solo debe utilizarse para tareas de administración de bases de datos, como la creación de otros usuarios, roles y bases de datos, y nunca debe utilizarse por la aplicación.

Se detalla a continuación el método recomendado para configurar un control de acceso detallado en PostgreSQL:

- Utilizar el usuario postgres para crear roles por aplicación o caso de uso, como *readonly* (solo lectura) y *readwrite* (escritura).
- Agregar permisos para permitir que estos roles tengan acceso a varios objetos de base de datos. Por ejemplo, el rol de *readonly* solo puede ejecutar consultas *SELECT*.
- Conceder a los roles los menos permisos posibles necesarios para la funcionalidad.
- Crear nuevos usuarios para cada aplicación o funcionalidad, como *app_user* (usuario de aplicación) y *reporting_user* (usuario de reportes).
- Asignar los roles aplicables a estos usuarios para otorgarles los mismos permisos que el rol. Por ejemplo, conceder el rol *readwrite* a *app_user* y conceder el rol *readonly* a *reporting_user*
- En cualquier momento, podremos quitar el rol del usuario para revocar los permisos.

## Usuarios, grupos y roles

Los usuarios, grupos y roles son lo mismo en PostgreSQL, y la única diferencia es que los usuarios tienen permiso para iniciar sesión de forma predeterminada. Las instrucciones *CREATE USER* y *CREATE GROUP* son en realidad alias de la instrucción *CREATE ROLE*.
Para crear un usuario de PostgreSQL, utilizaremos la siguiente instrucción SQL:

```
CREATE USER myuser WITH PASSWORD 'passwd';
```

También se podría crear un usuario con la siguiente instrucción SQL:

```
CREATE ROLE myuser WITH LOGIN PASSWORD 'passwd';
```

Ambas sentencias crean exactamente el mismo usuario. Este nuevo usuario no tiene ningún permiso, a excepción de los permisos predeterminados disponibles para el rol *public*. Todos los nuevos usuarios y roles heredan los permisos del rol *public*. En la siguiente sección se proporcionan más detalles sobre dicho rol.

# Esquema public y rol public

Cuando creamos una nueva base de datos, PostgreSQL crea de forma predeterminada un esquema denominado *public* y concede acceso en este esquema a un rol de backend denominado *public*. A todos los usuarios y roles nuevos se les concede de forma predeterminada el rol *public* y, por lo tanto, pueden crear objetos en el esquema *public*.

Cuando un usuario intenta crear una nueva tabla sin especificar el nombre del esquema, la tabla se crea en el esquema *public*. De forma predeterminada, todos los usuarios tienen acceso para crear objetos en el esquema *public* y esto se convierte en un problema si intentamos crear un usuario de solo lectura. Incluso si restringimos todos los privilegios, los permisos heredados a través del rol *public* permiten al usuario crear objetos en el esquema *public*.

Para solucionarlo, se debe revocar el permiso de creación predeterminado en el esquema *public* desde el rol *public* mediante la siguiente instrucción SQL:

```
REVOKE CREATE ON SCHEMA public FROM PUBLIC;
```

Nos aseguraremos de ser el propietario del esquema *public* o de formar parte de un rol que le permita ejecutar esta instrucción SQL. La siguiente declaración revoca la capacidad del rol público de conectarse a la base de datos:

```
REVOKE ALL ON DATABASE mydatabase FROM PUBLIC;
```

Esto garantiza que los usuarios no puedan conectarse a la base de datos de forma predeterminada a menos que se conceda explícitamente este permiso. La revocación de los permisos del rol *public* afecta a todos los usuarios y roles existentes. Los usuarios y roles que deben de poder conectarse a la base de datos o crear objetos en el esquema público, deben recibir los permisos explícitamente antes de revocar los permisos del rol *public* en el entorno de producción.

# Crear roles de base de datos

A continuación, se detalla el proceso de creación de nuevos roles y concesión de permisos para acceder a varios objetos de base de datos. Los permisos debemos concederlos a nivel de base de datos, esquema y objeto de esquema. Por ejemplo, si necesitamos conceder acceso a una tabla, también debemos asegurarnos de que el rol tenga acceso a la base de datos y al

esquema en que existe la tabla. Si falta alguno de los permisos, el usuario/rol no podrá acceder a la tabla.

## Rol de solo lectura

El primer paso consiste en crear un nuevo rol denominado readonly mediante la siguiente instrucción SQL:

```
CREATE ROLE readonly;
```

Este es un rol simple sin permisos ni contraseña. No se puede utilizar para iniciar sesión en la base de datos.

Concedemos permiso a este rol para conectarse a la base de datos de destino denominada “mydatabase”:

```
GRANT CONNECT ON DATABASE mydatabase TO readonly;
```

El siguiente paso es otorgar acceso al uso de este rol a su esquema. Supongamos que el esquema se llama myschema:

```
GRANT USAGE ON SCHEMA myschema TO readonly;
```

En este paso se concede permiso de rol de *readonly* para realizar alguna actividad dentro del esquema. Sin este paso, el rol *readonly* no puede realizar ninguna acción en los objetos de este esquema, incluso si se han concedido permisos para esos objetos.

El siguiente paso consiste en otorgar acceso al rol *readonly* para ejecutar las consultas en las tablas requeridas.

```
GRANT SELECT ON TABLE mytable1, mytable2 TO readonly;
```

Si el requisito es conceder acceso a todas las tablas y vistas del esquema, podemos utilizar el siguiente SQL:

```
GRANT SELECT ON ALL TABLES IN SCHEMA myschema TO readonly;
```

La instrucción SQL anterior otorga acceso *SELECT* al rol de solo lectura en todas las tablas y vistas existentes en el esquema llamado *myschema*. Debemos tener en cuenta que el usuario *readonly* no podrá acceder a las tablas nuevas que se añadan en el futuro. Para garantizar que también pueda acceder a nuevas tablas y vistas, deberemos ejecutar la siguiente instrucción para conceder permisos automáticamente:

```
ALTER DEFAULT PRIVILEGES IN SCHEMA myschema GRANT SELECT ON TABLES TO
readonly;
```

## Rol de lectura y escritura

El proceso de agregar un rol de lectura/escritura es muy similar al proceso de rol de solo lectura que se trató anteriormente. El primer paso es crear un rol:

```
CREATE ROLE readwrite;
```

Concedemos permiso a este rol para conectarse a la base de datos de destino:

```
GRANT CONNECT ON DATABASE mydatabase TO readwrite;
```

Concedemos privilegio de uso de esquemas:

```
GRANT USAGE ON SCHEMA myschema TO readwrite;
```

Si deseamos permitir que este rol cree nuevos objetos como tablas de este esquema, utilizaremos el siguiente SQL en lugar del anterior:

```
GRANT USAGE, CREATE ON SCHEMA myschema TO readwrite;
```

El siguiente paso sería conceder acceso a las tablas. Como hemos comentado en la sección anterior, la concesión puede realizarse en tablas individuales o en todas las tablas del esquema. Para tablas individuales, utilizaremos el siguiente SQL:

```
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE mytable1, mytable2 TO
readwrite;
```

Para todas las tablas y vistas del esquema, utilizaremos el siguiente SQL:

```
GRANT SELECT, INSERT, UPDATE, DELETE ON ALL TABLES IN SCHEMA myschema
TO readwrite;
```

Para conceder automáticamente permisos sobre tablas y vistas añadidas en el futuro:

```
ALTER DEFAULT PRIVILEGES IN SCHEMA myschema GRANT SELECT, INSERT,
UPDATE, DELETE ON TABLES TO readwrite;
```

Para los roles de lectura y escritura, normalmente existe el requisito de utilizar secuencias también. Podemos dar acceso selectivo de la siguiente manera:

```
GRANT USAGE ON SEQUENCE myseq1, myseq2 TO readwrite;
```

También podemos conceder permiso a todas las secuencias mediante la siguiente instrucción SQL:

```
GRANT USAGE ON ALL SEQUENCES IN SCHEMA myschema TO readwrite;
```

Para conceder permisos automáticamente a las secuencias añadidas en el futuro:

```
ALTER DEFAULT PRIVILEGES IN SCHEMA myschema GRANT USAGE ON SEQUENCES
TO readwrite;
```

Podremos conceder más o menos permisos en función de los requisitos. La documentación del comando GRANT de PostgreSQL proporciona más detalles sobre los objetos en los que se pueden conceder permisos y las sentencias SQL necesarias.

## Creación de usuarios de base de datos

Con los roles implementados, se simplifica el proceso de creación de usuarios. Simplemente crearemos el usuario y le concederemos uno de los roles existentes. Estas son las instrucciones SQL para este proceso:

```
CREATE USER user1 WITH PASSWORD 'passwd';

GRANT readonly TO user1;
```

Esta instrucción SQL otorga a *myuser1* los mismos permisos que el rol de solo lectura.

Del mismo modo, podemos conceder acceso de lectura y escritura a un usuario otorgando el rol *readwrite*. La documentación CREATE USER de PostgreSQL contiene más detalles sobre los parámetros que podemos establecer al crear un usuario. Por ejemplo, podemos especificar un plazo de caducidad para el usuario o permitir que el usuario cree bases de datos.

## Revocar o cambiar los permisos de usuario

Utilizando el método detallado anteriormente, resulta muy fácil revocar los privilegios de un usuario. Por ejemplo, podemos quitar el permiso de lectura y escritura (*readwrite*) de *user1* utilizando la siguiente instrucción SQL:

```
REVOKE readwrite FROM user1;
```

Del mismo modo, podemos otorgar un nuevo rol de la siguiente manera:

```
GRANT readonly TO user1;
```

María Guadalupe Martínez Pavón Grupo 1
Bases de Datos

Bibliografía

Lopez, J. (2021, 22 de diciembre). *Administración de usuarios y de roles de PostgreSQL*. Cloud, DevOps, Seguridad, Soporte InformÃ¡tico | Qualoom. https://www.qualoom.es/blog/administracion-usuarios-roles-postgresql/