

Administración en PostgreSQL

Antonio Espín Herranz

Contenidos

- Administrar:
 - Bases de datos.
 - Esquemas
 - TableSpaces
 - Roles y privilegios
 - Usuarios y grupos

Administrar Bases de datos

- Comando **createdb** (dentro de la carpeta bin)
- Createdb [opcionesConex] [opciones][nombreDb] [descripción]
 - **Opciones de conexión:**
 - -h servidor
 - -p puerto
 - -U usuario
 - -w Ambas opciones -w y -W para que nos pida o no clave de acceso al servidor
 - -W
- Son opciones del servidor donde queremos crear la BD.

createdb

- **Opciones:**

- -D tablespace nombre del tablespace
- -e comandoShow muestra el comando
- -E codificación codificación de la BD, por defecto UTF-8
- -l ubicación ubicación
- -O propietario por defecto será el usuario postgres
- -T plantilla se puede usar una plantilla para crearla
- -V para la versión

createdb

- **nombredb** indicar el nombre de la BD, si no se indica la crea con el **nombre del usuario del sistema operativo**.
- **descripcion**: Se puede especificar una descripción de la BD.

Desde psql

- Desde **psql**

- Create database [nombreDB] [with **definiciones**];

- **Definiciones:**

- Owner = usuario
 - Template = plantilla
 - Encoding = 'codificación'
 - Tablespace = nombre tablespace
 - Connection limit = -1

por defecto: postgres

indicar una plantilla para crear la BD

por defecto utf-8

nombre del tablespace

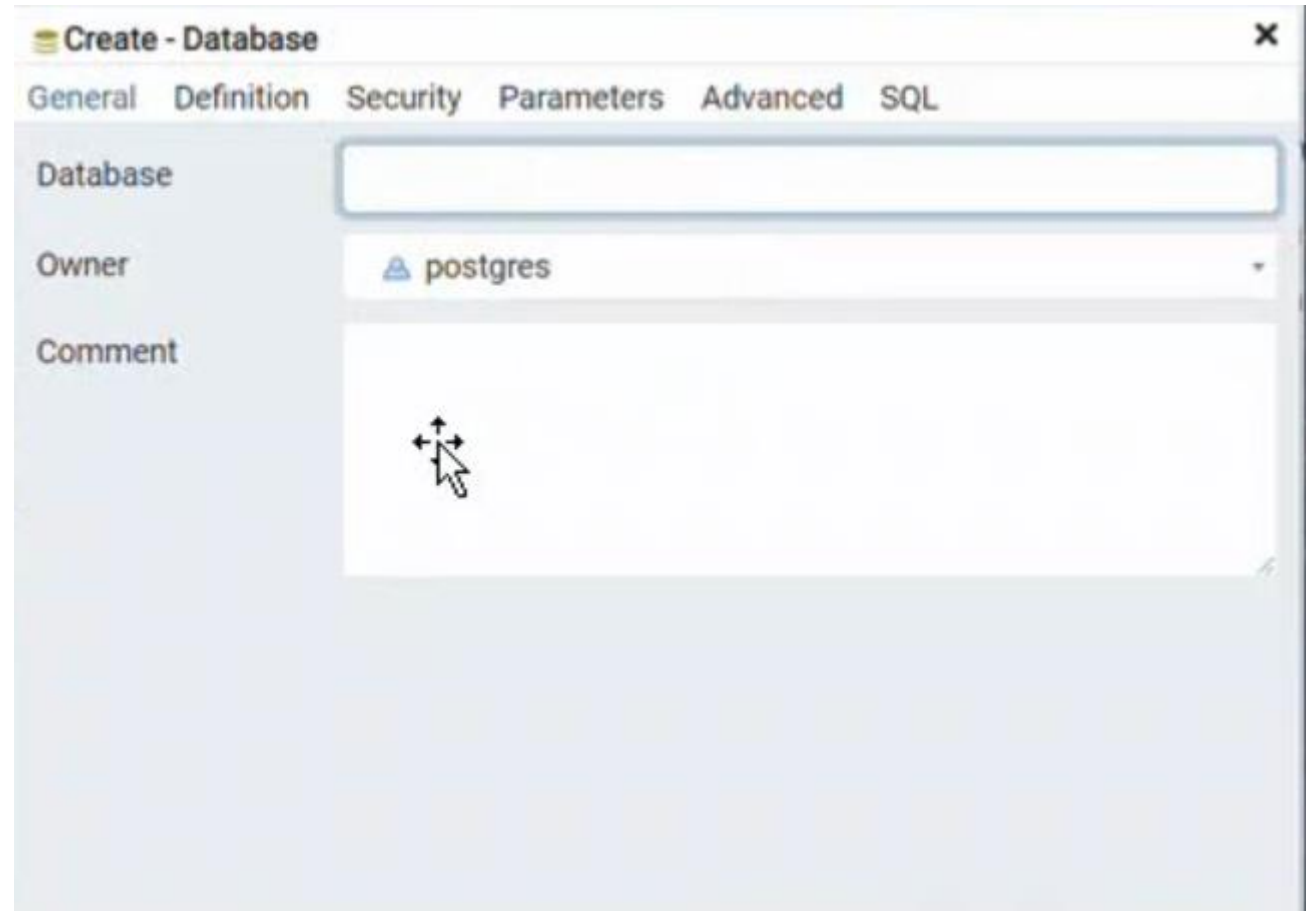
-1 no hay límite de conexiones

Desde psql

- Se puede añadir un comentario a la BD:
 - `Comment on database [nombreBD] is 'comentarios';`

Desde pgAdmin

- Utilizar el formulario:
 - General
 - Nombre de la BD
 - Propietario
 - Comentarios














The screenshot shows the 'Create - Database' dialog box in pgAdmin. The 'General' tab is selected, and the 'Definition' sub-tab is active. The 'Database' field is empty and highlighted with a blue border. The 'Owner' field is set to 'postgres' with a dropdown arrow. The 'Comment' field is empty and contains a mouse cursor icon.

Field	Value
Database	
Owner	postgres
Comment	

Desde pgAdmin

- La plantilla, por defecto toma la BD de postgres
- Tablespace: por defecto pg_default
- Collation y character type: 1252

General	Definition	Security	Parameters	Advanced	SQL
Encoding	UTF8   				
Template	 postgres 				
Tablespace	 pg_default 				
Collation	Spanish_Spain.1252  				
Character type	Spanish_Spain.1252  				
Connection limit	-1				

Administrar Esquemas

- Conceptualización
- Consideraciones generales
- ¿Porque utilizar esquemas?
- Ruta de búsqueda de objetos

Más operaciones con la BD

- ALTER DATABASE

- **Cambiar atributos de la BD:**

- ALTER DATABASE nombre_bd WITH option;

- Opción:

- IS_TEMPLATE

- CONNECTION LIMIT

- ALLOW_CONNECTIONS

- Solo para superusuarios o dueños de la BD

- **Cambiar el nombre de la BD:**

- ALTER DATABASE nombre RENAME TO nuevo_nombre;

Más operaciones con la BD

- ALTER DATABASE

- **Cambiar el propietario de la BD**

- ALTER DATABASE nombre_bd OWNER TO nuevo_owner | current_user | sesion_user

- Para obtener información del usuario:

- Select user;
 - Select current_user;
 - Select session_user;

- **Cambiar el tablespace de la BD**

- ALTER DATABASE nombre_bd SET TABLESPACE nuevo_tablespace;

Administrar Esquemas

- Un esquema es un namespace que contiene objetos de la BD.
- Tablas, Vistas, índices, operadores, funciones, procedures, tipos de datos, etc.
- Acceso a los objetos: nombreEsquema.nombreObjeto
- Si solo tenemos el esquema public el nombre del esquema no es necesario.
- Si el nombre del esquema lleva letras mayúsculas utilizar comillas dobles.

Administrar Esquemas

- Una BD puede tener 1 o más esquemas
- El esquema por defecto es public, si solo se indica el nombre de un objeto(por ejemplo: una tabla), por defecto lo busca en public.
- Se puede borrar si queremos.
- Se pueden duplicar objetos si están en esquemas diferentes.
- **public.productos** vs **esquema1.productos**.

Administrar esquemas

- Los esquemas permiten organizar los objetos de la base de datos.
- Permiten que múltiples usuarios utilicen una base de datos sin interferir con otros usuarios.
- `Create schema [if not exists] [nombreEsquema] [authorization user];`
- Si no se indica el nombre del esquema, postgres utilizará el nombre del usuario que lo está creando.
- Se puede indicar el dueño (authorization) y un comentario.
 - `comment on schema [nombreEsquema] is 'comentarios';`

Administrar esquemas

- Desde **pgAdmin** se crea con el formulario.
- Para eliminar el schema podemos con el botón derecho: **drop cascade** para borrar todo lo que está dentro del esquema.
- **Delete/drop** solo lo borra en caso de que esté vacío.

Ruta de búsqueda en esquemas

- Search path
 - Si no se define una ruta de búsqueda, esta se limita sólo al esquema público.
 - El comando `search_path` nos permite definir una ruta de búsqueda a un esquema:
 - `SET search_path TO esquema1 [, ...];`
 - Suponiendo que tenemos 3 esquemas: `public`, `schComercial`, `schFinanciero`
 - `SET search_path TO public, "schComercial", "schFinanciero";`
 - Establecemos una ruta de búsqueda (la posición indica el orden de búsqueda).

Ejemplo

- public
 - tables
 - países
- schComercial
 - tables
 - **ciudades**
- schFinanciero
 - tables
 - **clientes**
 - ciudades

Select * from clientes;
1- busca en public
2- busca en schComercial
3- **La encuentra en schFinanciero**

Select * from **ciudades;**
1- busca en public
2- busca en **schComercial** (lista la tabla **ciudades**)

Para listar ciudades del esquema schFinanciero
Select * from “schFinanciero”.ciudades;

Administrar TableSpaces

- Un tablespace es una ubicación en un almacenamiento
 - Donde se almacenan tablas e índices
- Cuando instalamos postgres ya viene con dos tablespace:
 - **pg_default**: almacena datos de usuarios
 - **pg_global**: almacena datos globales.
- Se pueden crear todos los tablespaces que queramos, pero **no pueden empezar por pg_**

Administrar TableSpaces

- El usuario que crea el tablespace es el dueño por defecto del tablespace.
 - Se puede indicar otro dueño utilizando **owner**.
- La variable **directory_path** es el path absoluto.
- Los tablespace son una parte integral del cluster de la BD.
- No definir tablespaces en espacios temporales o RAM, puede fallar y afectar al cluster de la BD.

Administrar TableSpaces

- Podemos controlar el diseño y el uso de espacios de almacenamiento.
 - Se puede administrar espacios, por ejemplo, porque un tablespace se quede sin espacio.
- Permiten optimizar el rendimiento
 - Se pueden colocar las tablas e índices de mayor acceso en discos SSD
 - Y datos históricos y datos con pocos accesos en otro medios de acceso más lentos.

Administrar TableSpaces

- Sintaxis

- **psql:**

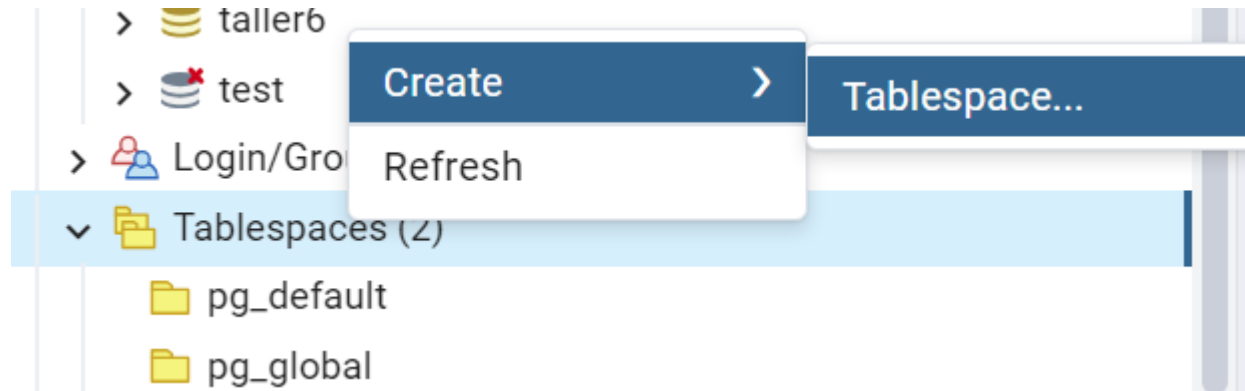
- ```
create tablespace [if not exists] nombreTableSpace
[owner nombreUser]
Location 'unidad:\directorio\subdirectorio';
```

- El usuario que se indica tiene que tener permisos de lectura y escritura sobre la ubicación indicada.

- Como en otros objetos se pueden añadir comentarios:

- `comment on tablespace nombre-tablespace is 'comentarios';`

# Administrar TableSpaces



- En las pestañas:
  - **General:** nombre, dueño y comentarios
  - **Definición:** Location → la ubicación

# Administrar TableSpaces

- Desde **psql**
  - Create tablespace temporal **location** 'C:\dbempresa\datos';
  - El owner por defecto será el usuario **postgres**.
- Se puede luego asignar otro dueño:
  - Alter tablespace nombre\_tablespace owner to nombreUser;
  - El nombre no puede empezar por **pg\_** → están reservados para **postgresql**



# Administrar TableSpaces

- Podemos crear una base de datos nueva en otro tablespace
- Create database dbtemporal tablespace temporal;
- Se pueden ubicar tablespace en otras unidades (que antes exista la ruta).
  - Create tablespace principal location 'D:\temporal';






# Administrar TableSpaces

- Para crear una tabla que apunte a un tablespace creado.
  - Create table clientes (código smallint, nombre varchar(20))
  - Tablespace principal;
  - En pgAdmin4 cuando seleccionamos un tablespace nos muestra los objetos que están almacenados en ese tablespace.
- |         |                 |            |
|---------|-----------------|------------|
| • Type  | name            | Database   |
| • Table | public.clientes | dbtemporal |


# Administrar TableSpaces

- Podemos tener un tablespace con una base de datos y objetos de esa base de datos se pueden incluir en otro tablespace distinto de la Bd.
- Por ejemplo,
  - Base de datos: bdTemporal → tablespace Temporal
  - Una tabla de bdTemporal → en tablespace principal.
- En el tablespace temporal contiene índices, vistas que se crean por defecto al crear una nueva base de datos.

# Tablespace: temporal / bdtemporal

| Type                                                                                       | Name                                | Database   |
|--------------------------------------------------------------------------------------------|-------------------------------------|------------|
|  Database |                                     | dbtemporal |
|  Index    | pg_aggregate_fnoid_index ON pg_c... | dbtemporal |
|  Index    | pg_am_name_index ON pg_catalog....  | dbtemporal |
|  Index    | pg_am_oid_index ON pg_catalog.pg... | dbtemporal |
|  Index    | pg_amop_fam_strat_index ON pg_c...  | dbtemporal |

# Tablespace: principal / tabla clientes

| Type                                                                                      | Name            | Database   |
|-------------------------------------------------------------------------------------------|-----------------|------------|
|  Table | public.clientes | dbtemporal |

# Administrar TableSpaces

- Cuando intentamos borrar un tablespace si no está vacío nos saltará un error.

# Comandos para tablespaces

Obtener la lista de los tablespaces existentes:

```
SELECT spcname FROM pg_tablespace;
```

Obtener el espacio utilizado por un tablespace específico:

```
SELECT pg_size_pretty (pg_tablespace_size ('nmTablespace'));
```

Renombrar un tablespace:

```
ALTER TABLESPACE nmTablespace RENAME TO nmNuevo
```

Cambiar el propietario de un tablespace:

```
ALTER TABLESPACE nmTablespace OWNER TO nmNuevoPropietario
```



# Comandos para tablespace

Definir tablespace por defecto (diferente a pg\_default):

**SET default\_tablespace = nmTablespace;**

Crear una base de datos en un tablespace específico:

**CREATE DATABASE nmDatabase TABLESPACE nmTablespace;**

Crear una tabla en un tablespace específico:

**CREATE TABLE nmTabla (nmCampo tipo) TABLESPACE nmTablespace;**

Crear un índice en un tablespace específico:

**CREATE INDEX nmIndice ON nmTabla (campo) TABLESPACE nmTablespace;**

# Comandos para tablespace

Eliminar un tablespace especifico:

```
DROP TABLESPACE [IF EXISTS] nmTablespace;
```

Modificar tablespace definido para una base de datos:

```
ALTER DATABASE nmDatabase SET TABLESPACE nmTablespace;
```

Modificar tablespace definido para una tabla especifica:

```
ALTER TABLE nmTabla SET TABLESPACE nmTablespace;
```

Modificar tablespace definido para TODAS las tablas:

```
ALTER TABLE ALL IN TABLESPACE nmTablespace
SET TABLESPACE nmNuevoTablespace;
```



# Comandos para tablespace

**Modificar propiedades del tablespace**

```
ALTER TABLESPACE nmTablespace SET (nmVariable = nuevoValor);
```

# Administrar Roles y Privilegios

- Conceptos
- Crear roles
- Otorgar privilegios
- Revocar privilegios
- Modificación de atributos
- Eliminación roles
- Miembros de un role
- Comandos útiles

# Conceptos

- Un usuario es cualquier entidad que se puede identificar en el sistema de postgresql.
  - Tiene asignado un role o varios
  - Cada role representa una colección de permisos.
- 
- Grupos: colecciones de usuarios
  - Los grupos pueden tener asignados uno o varios roles.
  - Un usuario puede ser miembro de uno o varios grupos.

# Crear roles (desde psql)

**CREATE ROLE** *nmRol* [ [ **WITH** ] *opciones* [ ... ] ]

|                                  |               |
|----------------------------------|---------------|
| SUPERUSER                        | NOSUPERUSER   |
| CREATEDB                         | NOCREATEDB    |
| CREATEROLE                       | NOCREATEROLE  |
| INHERIT                          | NOINHERIT     |
| LOGIN                            | NOLOGIN       |
| CONNECTION LIMIT connlimit       |               |
| PASSWORD ' <i>clave</i> '        | PASSWORD NULL |
| VALID UNTIL ' <i>timestamp</i> ' |               |
| IN ROLE <i>nmRole</i> [, ...]    |               |
| ROLE <i>nmRole</i> [, ...]       |               |
| ADMIN <i>nmRole</i> [, ...]      |               |

# Crear roles (desde psql)

- Estas opciones autorizan o desautorizan al role

|                   |                     |
|-------------------|---------------------|
| <b>SUPERUSER</b>  | <b>NOSUPERUSER</b>  |
| <b>CREATEDB</b>   | <b>NOCREATEDB</b>   |
| <b>CREATEROLE</b> | <b>NOCREATEROLE</b> |
| <b>INHERIT</b>    | <b>NOINHERIT</b>    |
| <b>LOGIN</b>      | <b>NOLOGIN</b>      |

# Crear roles (desde psql)

- Superuser puede sobrescribir todas las restricciones de acceso a una BD.
  - **Create role administrador with superuser;**
  - Son un riesgo de seguridad y solo se deberían de utilizar cuando sea estrictamente necesario.
  - Para poder otorgar este role, el que lo otorga también debe ser un superusuario.
  - El valor por defecto de esta opción es: **nosuperuser**

# Crear roles (desde psql)

- Create role administrador with **createdb**
  - Para poder crear bases de datos
  - Por defecto: nocreatedb
- Create role administrador with **createrole**
  - Puede crear roles y añadir o modificar permisos de otros roles.
  - Por defecto: nocreaterole
- **Inherit**
  - Determina si el nuevo role “hereda” los permisos del grupo al que pertenece.
  - Por defecto: noinherit → no hereda.

# Crear roles (desde psql)

- Create role administrador with **login**
  - Login | nologin
  - Determina si el nuevo role puede o no hacer login
  - Por defecto no login
  - Tendrá un nombre y password para conectar a la BD.
  - Los roles con este atributo son útiles para administrar privilegios de BD.
  - Pero no son usuarios.



# Ejemplo

- A un role se le puede añadir más de una característica.
- Create role administrador with superuser createdb createrole
- Luego a los usuarios se les asigna un role.

# Crear roles

```
CONNECTION LIMIT connlimit
PASSWORD 'clave' | PASSWORD NULL
VALID UNTIL 'timestamp'
IN ROLE nmRole [, ...]
ROLE nmRole [, ...]
ADMIN nmRole [, ...]
```

- **Connection limit connlimit**

- Indica si el nuevo role tiene alguna limitación en la cantidad de conexiones.
  - Es el número de conexiones concurrentes puede hacer el role.
  - -1 sin límite
- Solo se aplica para los roles que pueden hacer login

# Crear roles

```
CONNECTION LIMIT connlimit
PASSWORD 'clave' | PASSWORD NULL
VALID UNTIL 'timestamp'
IN ROLE nmRole [, ...]
ROLE nmRole [, ...]
ADMIN nmRole [, ...]
```

- **Password 'clave'**
  - Determina si el nuevo role (que debe tener acceso login) tiene una clave de acceso asignada.
  - Si no planea hacer login, puede dejar password null.

# Crear roles

```
CONNECTION LIMIT connlimit
PASSWORD 'clave' | PASSWORD NULL
VALID UNTIL 'timestamp'
IN ROLE nmRole [, ...]
ROLE nmRole [, ...]
ADMIN nmRole [, ...]
```

- **valid until 'timestamp'**

- Determina si hay definida una fecha de expiración de la clave de acceso definida para el role.
- La fecha / hora indica el instante en que la clave dejará de funcionar.
- Si se omite la cláusula, la clave nunca vencerá.

# Crear roles

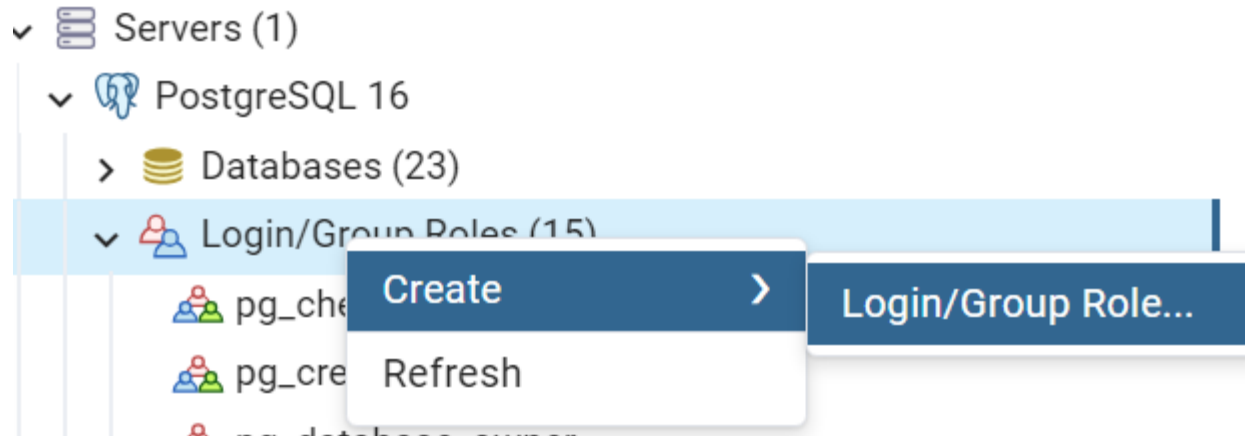
```
CONNECTION LIMIT connlimit
PASSWORD 'clave' | PASSWORD NULL
VALID UNTIL 'timestamp'
IN ROLE nmRole [, ...]
ROLE nmRole [, ...]
ADMIN nmRole [, ...]
```

- **in role nombreRole, ...**
  - Fija una lista de uno o varios roles existentes donde se añadirá este role.
- **Role nombreRole**
  - Fija una lista de uno o varios roles existentes donde se añadirá este role
- **Admin nombreRole**
  - Se comporta igual que la opción ROLE. Los roles indicados se añaden al nuevo role.

# Ejemplos

- Create role administrador with connection limit -1;
- Create role administrador with login superuser password '12345';
- Create role administrador with login superuser password '12345' valid until '2021-12-31 23:59:59';
  - Cuando la clave expire pedirá una nueva clave.
- Create role pruebas with login password '12345' in role administrador;

# Desde pgAdmin



- En el formulario que aparece:
  - Pestaña general: nombre y comentarios
  - Pestaña definición: password, fecha de expiración y número de conexiones
  - Pestaña privilegios: superuser, login, crear roles, etc.

# Otorgar privilegios

- A parte de crear los roles, tenemos que otorgar luego privilegios sobre los objetos de la bd.
- Por ejemplo, hemos creado un role: usuario1
- Si nos conectamos al servidor con este role e intentamos hacer un select a una tabla, tenemos que tener permiso para poder listar esa tabla.

```
postgres=> \c empresa3
Ahora está conectado a la base de datos «empresa3» con el usuario «antonio».
empresa3=> \dt
 Listado de relaciones
Esquema | Nombre | Tipo | Dueño
-----+-----+-----+-----
public | tbcategorias | tabla | postgres
public | tbclientes | tabla | postgres
public | tbcompaniaenvios | tabla | postgres
public | tbdetallespedidos | tabla | postgres
public | tbempleados | tabla | postgres
public | tbpedidos | tabla | postgres
public | tbproductos | tabla | postgres
public | tbproductos_copia | tabla | postgres
public | tbproveedores | tabla | postgres
(9 filas)
```

```
empresa3=> select * from tbcategorias;
ERROR: permiso denegado a la tabla tbcategorias
```



# Otorgar privilegios

- Es necesario otorgar permisos al role para poder interactuar con la BD.
- Se otorgan permisos para las tablas, las columnas, las bases de datos, funciones, procedimientos y demás objetos.
- Los privilegios se otorgan con el comando `GRANT`.

# Grant para tablas

```
GRANT [listaPrivilegios [, ...]
 | ALL]
ON [nmTabla [, ...]
 | ALL TABLES IN SCHEMA "nmEsquema"]
TO nmRole [, ...] [WITH GRANT OPTION];
```

# Grant para tablas

- La lista de privilegios:
  - **SELECT, INSERT, UPDATE, DELETE, TRUNCATE, REFERENCES, TRIGGER.**
- Si ponemos ALL hacemos referencia a todos los privilegios.
- Si puede hacer a nivel de tabla o varias tablas. También se puede otorgar a todas las tablas de un esquema.
- Luego a que ROLES se otorgan los privilegios.
- **With grant option:** indica que el role que recibe los privilegios también se los puede asignar a otros roles.

## Grant para columnas

```
GRANT [listaPrivilegios [, ...] (nmCol [, ...])
 | ALL (nmCol [, ...])]
ON nmTabla [, ...]
TO nmRole [, ...] [WITH GRANT OPTION];
```

# Grant para columnas

- Lista de privilegios para las columnas:
  - **SELECT, INSERT, UPDATE, REFERENCES**
  - Entre paréntesis se indican los nombres de las columnas, separadas por comas o indicar ALL para todas las columnas y especificar los nombres de las columnas
  - En una tabla determinada.
  - Luego se indica a que ROLE se otorga y
  - **WITH GRANT OPTION** para indicar si el role puede otorgar estos permisos a otros roles.

# Grant para Base de datos

```
GRANT [listaPrivilegios [, ...]
 | ALL]
ON DATABASE nmDatabase [, ...]
TO nmRole [, ...] [WITH GRANT OPTION];
```

# Grant para Base de datos

- Lista de privilegios:
  - **CREATE, CONNECT, TEMPORARY, TEMP**
  - O poner **ALL** para todos los privilegios.
- Indicar la base de datos
- To role: a que role se otorgan
- **WITH GRANT OPTION**: Si este role lo puede otorgar a otros roles.



# Grant Funciones / Procedimientos

**GRANT EXECUTE**

```
ON [{ FUNCTION | PROCEDURE } nmObjeto [, ...]
 | ALL { FUNCTIONS | PROCEDURES }
]
TO nmRole [, ...] [WITH GRANT OPTION];
```



# Revocar privilegios

- Igual que **Grant** se va utilizando a nivel de:
  - Tablas,
  - Columnas,
  - Bases de datos y
  - funciones / procedimientos
- Con **Revoke** pasa lo mismo.

# Revoke para Tablas

```
REVOKE [listaPrivilegios [, ...]
 | ALL]
ON [nmTabla [, ...]
 | ALL TABLES IN SCHEMA "nmEsquema"]
FROM nmRole [, ...] [CASCADE | RESTRICT];
```

# Revoke para Tablas

- La lista de privilegios es la misma que en las tablas:
  - SELECT, INSERT, UPDATE, DELETE, TRUNCATE, REFERENCES (*cuando definimos una FK a la tabla que referenciamos*), TRIGGER
- El ROLE al que se le retira los privilegios.
- El resto de parámetros: es lo mismo que en GRANT, pero revocando el privilegio.
- La última parte: CASCADE | RESTRICT
  - CASCADE: eliminar también los privilegios que dependan de este.
  - RESTRICT: opción por defecto, no se eliminan los privilegios.

# Revoke para Columnas

```
REVOKE [listaPrivilegios [, ...]
 | ALL] (nmcol1 [, ...])
 ON [nmTabla [, ...]
FROM nmRole [, ...] [CASCADE | RESTRICT];
```

# Revoke para Columnas

- Los privilegios:
  - SELECT, INSERT, UPDATE, REFERENCES (*cuando definimos una FK a la tabla que referenciamos*)
- Se pueden indicar varias columnas o indicar ALL
- On tabla (sobre que tabla)
- From role (a que role se revocan).

# Revoke para Bases de datos

```
REVOKE [listaPrivilegios [, ...]
 | ALL]
 ON DATABASE nmDatabase [, ...]
 FROM nmRole [, ...] [CASCADE | RESTRICT];
```

# Revoke para Bases de datos

- Lista de privilegios:
  - CREATE, CONNECT, TEMPORARY, TEMP
  - TEMPORARY / TEMP son sinónimos, es el permiso para crear tablas temporales dentro de la BD.



# Revoke para Funciones / Procedimientos

```
REVOKE EXECUTE
ON [{ FUNCTION | PROCEDURE } nmObjeto [, ...]
 | ALL { FUNCTIONS | PROCEDURES }
]
FROM nmRole [, ...] [CASCADE | RESTRICT];
```

Borrado en CASCADE, si tiene permisos dependientes también se eliminarán o no (RESTRICT)



# Modificación de atributos

- Para modificar un ROLE:
- ALTER ROLE nombre\_role [WITH] **opciones** [...]



|                         |  |               |
|-------------------------|--|---------------|
| SUPERUSER               |  | NOSUPERUSER   |
| CREATEDB                |  | NOCREATEDB    |
| CREATEROLE              |  | NOCREATEROLE  |
| INHERIT                 |  | NOINHERIT     |
| LOGIN                   |  | NOLOGIN       |
| CONNECTION LIMIT        |  | connlimit     |
| PASSWORD 'clave'        |  | PASSWORD NULL |
| VALID UNTIL 'timestamp' |  |               |

# Modificación de atributos

- SUPERUSUARIO: si es o no un superusuario. Puede modificar todo, pero se debe hacer a través de un usuario que sea un superuser.
- CREATEDB: Crear o no BD
- CREATEROLE: Crear o no ROLES
- INHERIT: Determina si el rol “hereda” los permisos del grupo al cual pertenece
- LOGIN: Puede hacer o no login.

|                   |                     |
|-------------------|---------------------|
| <b>SUPERUSER</b>  | <b>NOSUPERUSER</b>  |
| <b>CREATEDB</b>   | <b>NOCREATEDB</b>   |
| <b>CREATEROLE</b> | <b>NOCREATEROLE</b> |
| <b>INHERIT</b>    | <b>NOINHERIT</b>    |
| <b>LOGIN</b>      | <b>NOLOGIN</b>      |

# Ejemplo

- ALTER ROLE administrador WITH SUPERUSER CREATEDB CREATEROLE;

```
CONNECTION LIMIT connlimit
PASSWORD 'clave' | PASSWORD NULL
VALID UNTIL 'timestamp'
```

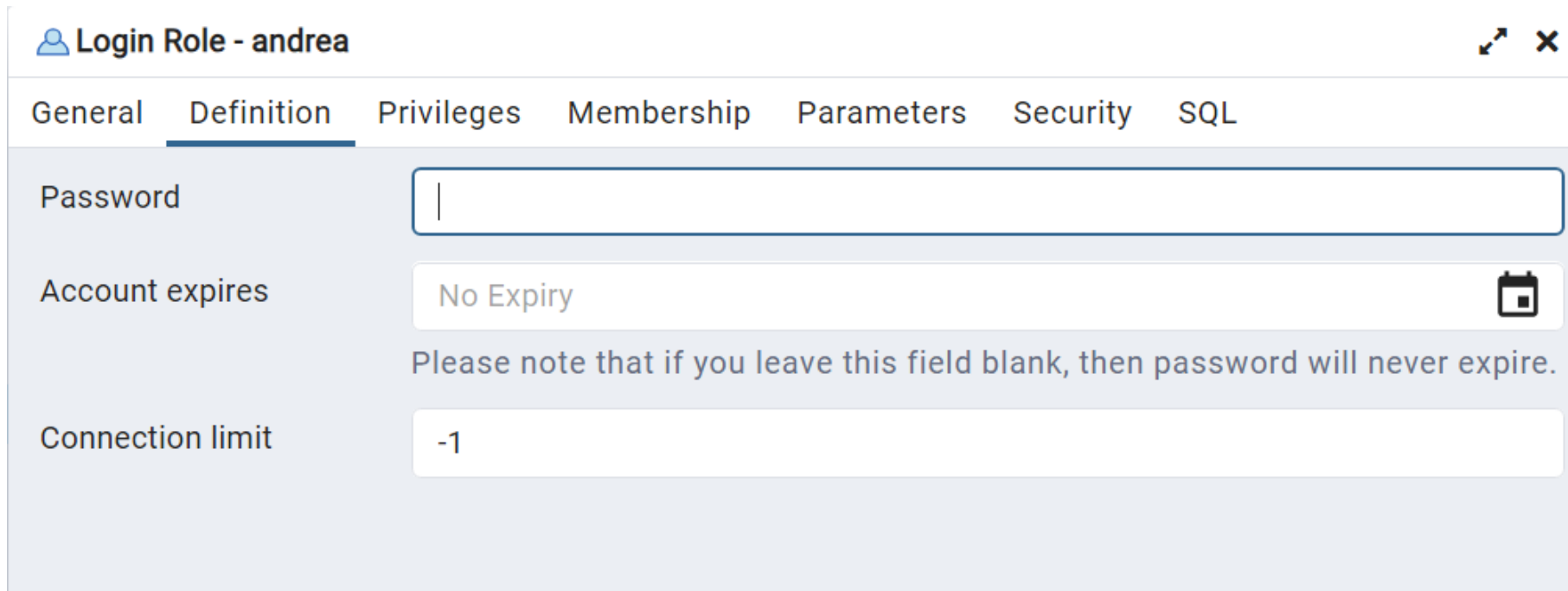
- El resto de opciones, indican:
  - Connection limit: número de conexiones concurrentes. -1 sin límite
  - Password 'clave', para indicar la password → solo para los que hacen login
    - Si no tiene login, se puede alterar con password null.
  - valid until 'timestamp'
    - Para indicar una fecha / hora de expiración de la clave.

# Ejemplos

- Create role administrador superuser login password 'miclave' valid until '2023-12-31 23:59:59';
- Create role andrea login password 'andrea123' in role administrador;
- Create role carlos nologin in role administrador;
- -- Cambiar un password:
  - Alter role andrea password 'ale2128';
- -- Añadir el login y createdb a un usuario:
  - Alter role carlos login password 'carlos123' createdb;

# En pgAdmin

- Mostrando las propiedades del role que queremos modificar, son las mismas pestañas cuando estamos creando un role.



The screenshot shows the 'Login Role - andrea' configuration window in pgAdmin. The 'Definition' tab is selected, showing fields for 'Password', 'Account expires', and 'Connection limit'. The 'Password' field is empty. The 'Account expires' field is set to 'No Expiry' with a calendar icon. The 'Connection limit' field is set to '-1'. A note below the 'Account expires' field states: 'Please note that if you leave this field blank, then password will never expire.'

| Field            | Value     |
|------------------|-----------|
| Password         |           |
| Account expires  | No Expiry |
| Connection limit | -1        |

# Eliminación de roles

- Drop role [if exists] nombre\_role;
- Tener en cuenta:
  - Para eliminar un role que tiene el atributo superuser el role debe tener el atributo superuser.
  - Para eliminar un role NO tenga superuser el role con el que conectamos debe de tener el atributo creatorole.
  - No se puede eliminar un role que tenga referencias en cualquier base de datos.
    - Indica si este role ha creado objetos en alguna BD.

# Eliminación de roles

- Eliminar un role con objetos dependientes en la base de datos:
- Utilizar **REASSIGN OWNED**
  - Reasignar el dueño de todos los objetos del role que queremos eliminar.
  - **Reassign owned by role1 to role2;** → asigna los objetos del role1 al role2.
  - Si queremos eliminar los objetos de un role:
  - **Drop owned by rafaël;**
- Revocar todos los permisos asignados al Role.
- DROP Role para eliminar el role.

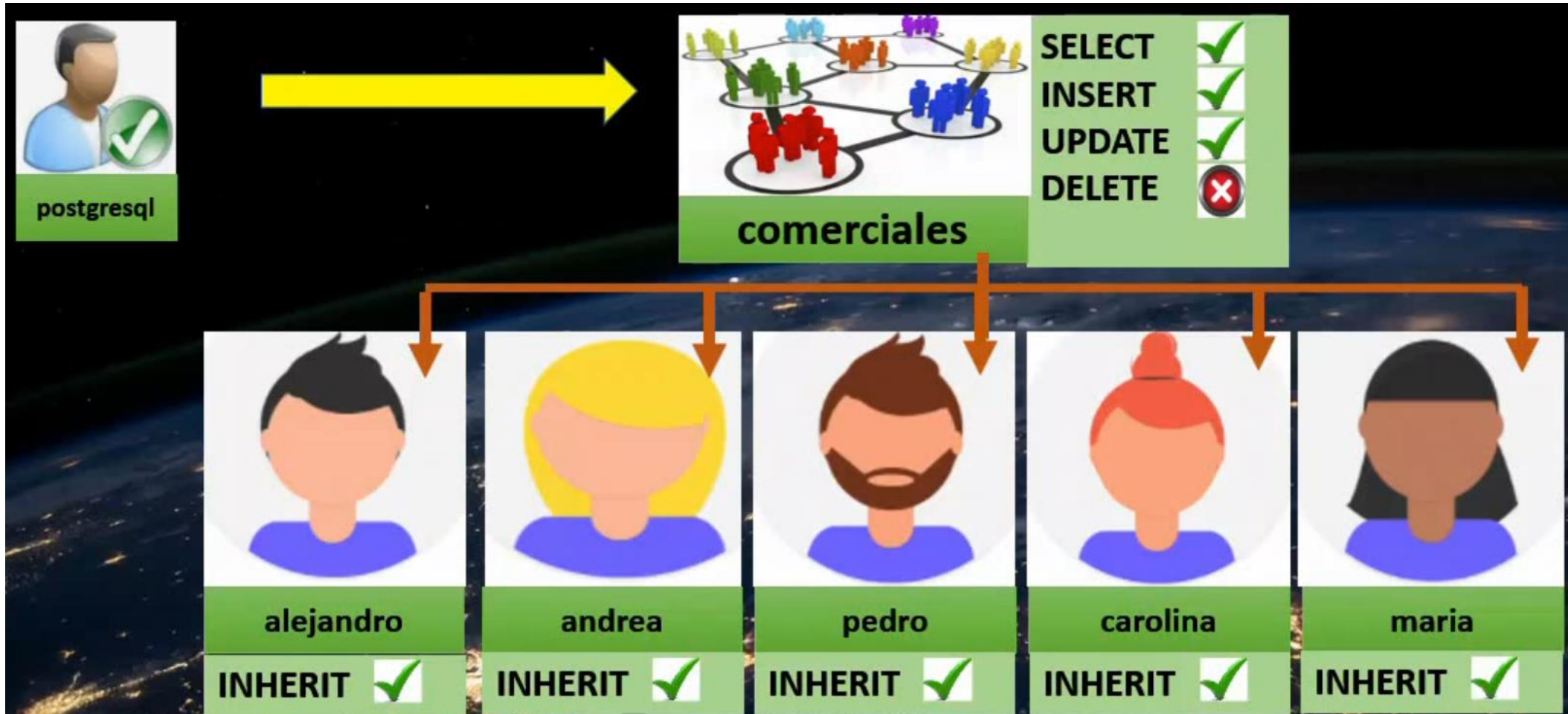
# Miembros de un role

- Grupos en los roles
- Si tenemos 5 roles con permisos de: select, insert, update y delete.
- Serían previamente creados por el superuser (por ejemplo: postgres) puede ser pesado ir uno a uno.
- La mejor solución sería: crear un role de grupo y le asignamos un nombre. Por ejemplo: **comerciales**.
- Y este role de grupo le asignamos los permisos de: **select, insert, update y delete**.
- Los 5 usuarios finales pueden heredar los permisos del role de grupo y no es necesario que se asignen de forma individual.



# Miembros de un role

- Para añadir un nuevo atributo basta con añadirlo al grupo comercial.



# Miembros de un role

- Los roles de grupo no tienen atributo de **LOGIN**.
- Podemos utilizar: `\z` para ver cómo están configurados los privilegios.

```
dbtaller15=# \z
```

| Esquema | Nombre       | Tipo  | Privilegios                                   | Privilegios de acceso a columnas | Políticas |
|---------|--------------|-------|-----------------------------------------------|----------------------------------|-----------|
| public  | clientes     | tabla | postgres=arwdDxt/postgres+<br>juan=r/postgres |                                  |           |
| public  | proyecciones | tabla |                                               |                                  |           |

(2 filas)

# Miembros de un role

- Para crear un role de grupo:
  - **Create role nombre\_role noinherit;**
- Asignar todos los permisos de una tabla al role de grupo
  - **Grant all on nombre\_tabla to role\_nombre;**
- Agregar un role como miembro de un role de grupo.
  - **Grant nombre\_role\_grupo to role;**

# Miembros de un role

- **Set role nombre\_role;**
  - Para asignar un role a un usuario
  - Se puede cambiar todas las veces que queramos
- **Reset role;**
  - Para resetear el role de un usuario, y volver al original.

# Miembros de un role

- Tener en cuenta.
  - Es recomendable administrar privilegios en roles de grupo y no en roles individuales.
  - Los roles con atributo INHERIT toman todos los privilegios del grupo del cual son miembros.

# Comandos útiles

- Comandos en **psql**:
- **\du**
  - Una lista con el estado de todos los roles que tenemos.

```
postgres=# \du
```

| Lista de roles |                                                            |
|----------------|------------------------------------------------------------|
| Nombre de rol  | Atributos                                                  |
| andrea         |                                                            |
| antonio        | Crear rol, Crear BD +                                      |
| consulta       | Constraseña válida hasta 2023-12-31 23:59:00+01            |
| daniel         |                                                            |
| juan           |                                                            |
| mercadeo       | Sin herencia, No puede conectarse                          |
| pedro          |                                                            |
| planeacion     | Sin herencia, No puede conectarse                          |
| postgres       | Superusuario, Crear rol, Crear BD, Replicación, Ignora RLS |

# Comandos útiles

- **\du+**
  - Muestra más información que el comando anterior, añade el comentario.
- Ver PDF:
  - **12\_utilidades-roles-postgresql.pdf**

# Usuarios y Grupos

- **Grupos de Usuarios**
- Esa propiedad nos permite agrupar a varios usuarios con el objetivo de asignar privilegios de manera general para optimizar tiempo. Luego podemos crear usuarios de manera independiente y enlazarlos con algún grupo.
- Sintaxis grupo:
- `CREATE GROUP [nombregrupo]`
- Sintaxis usuario:
- `CREATE USER [nombreusuario] WITH PASSWORD 'password' IN GROUP [nombregrupo]`



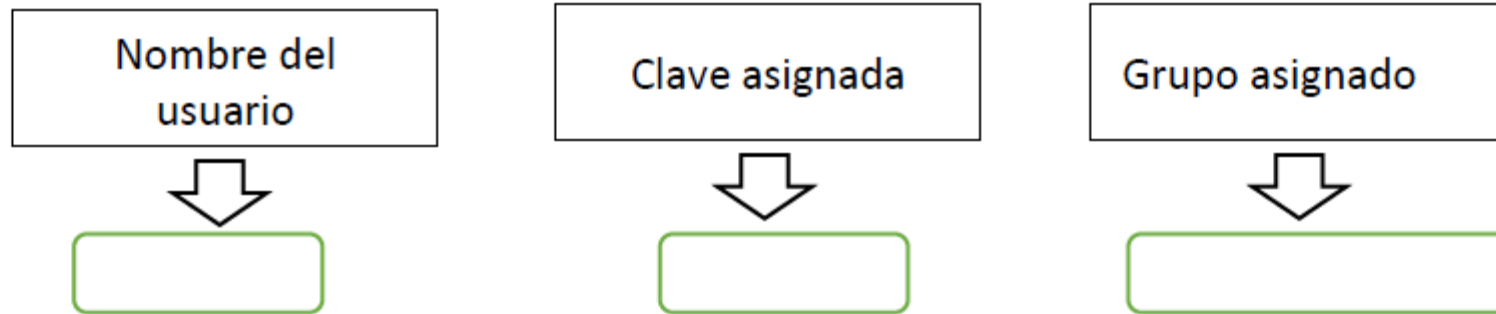
# Usuarios y Grupos

- **Privilegios**
- Con la asignación de privilegios a usuarios se da la autorización a que este o a un grupo de usuarios para que realice cualquier acción sobre una tabla específica. Dichas acciones pueden ser otorgadas con el comando “GRANT” o a su vez eliminadas con el comando “REVOKE”.
- Sintaxis:
- GRANT [SELECT, INSERT UPDATE, DELETE, ALL] ON [nombretabla] TO [nombreusuario o nombreGrupo]

# Ejemplo: los grupos y sus permisos

- Create group administradores;
- Create group directores;
- Create group supervisores;
- Create group empleados;
  
- Añadir permisos a los grupos:
- Grant all on “tbpersonas” to group administrativos;
- Grant all on “tbClientes” to group empleados;
- Etc.
  
- Podemos consultar permisos de los usuarios:
- `select * from pg_shadow;`

# Ejemplo: los usuarios



```
CREATE USER admin_jose WITH PASSWORD 'clave123' IN GROUP administrativos;
CREATE USER director_juan WITH PASSWORD 'abc123' IN GROUP directores;
CREATE USER superv_carlos WITH PASSWORD 'abc***' IN GROUP supervisores;
CREATE USER empl_luis WITH PASSWORD 'password' IN GROUP empleados;
```

# Usuarios y Grupos

- Alter user user1 with password 'nueva\_password';