	Carátula para entrega de prácticas
Facultad de Ingeniería	Universidad Nacional Autónoma de México

Laboratorio de Bases De Datos

Profesor(a): Ing. Angel Brito Segura

Asignatura: Bases de Datos

Grupo Teoria: 06

Grupo de Laboratorio: 09

No de Práctica(s): 03

Nombre de la Practica Seguridad y Lenguaje de Control de Acceso a Datos

Alumno: Vidrio Lopez Mario Alexis

No. de cuenta 321162892

Semestre: 2026-2

Fecha de entrega: 25 de febrero del 2026

Fecha de elaboracion en laboratorio: 18 de febrero del 2026

Introducción

En esta práctica se tiene como objetivo principal el emplear comandos de control de datos para la creación de usuarios, al igual que tanto otorgar como revocar permisos a estos mismos, también se requiere de la creación de una base de datos la cual se utilizará en el ambiente personal del alumn@, a lo largo de la misma se dirá que comandos se ejecutaron al igual que se añadirá una captura de pantalla del comando ejecutado conjunto al resultado, para la primer parte de la práctica nos centraremos en practicar los comando de creación de tablas, roles, otorgar y revocar permisos y la comprobación de que estos hayan sido ejecutados de la manera correcta empleando la ayuda de un compañero ya sea para que el intente acceder a nuestra base de datos o nosotros a la suya, ya en la segunda parte nos centramos en la creacion de nuestra base de datos, lo cual fue simplemente el seguir las instrucciones ya establecidas en la práctica proporcionada por el profesor de laboratorio.

Objetivos.

-Utilizar comandos de control de datos para crear usuarios, otorgar y revocar permisos.

-Crear la base de datos que se utilizará en el ambiente personal del alumn@.

Contenido.

- **C1.-** Captura de pantalla del resultado de la consulta sobre pg_roles y respuesta a *¿cuántos roles vienen creados por defecto en PostgreSQL y cuál es su utilidad?*

```
pipardo@pipardo-Predator-PHN16-71:~$ psql -h 132.248.59.8 -p 5432 -U lbd09_2026_2_321162892 -d lbd09_
2026_2_321162892
Password for user lbd09_2026_2_321162892:
psql (16.11 (Ubuntu 16.11-0ubuntu0.24.04.1), server 15.4)
Type "help" for help.

lbd09_2026_2_321162892=> SELECT rolname, rolsuper, rolinherit, rolcreaterole,rolcreatedb, rolcanlogin
 FROM pg_roles WHERE rolname LIKE 'pg_%';
          rolname          | rolsuper | rolinherit | rolcreaterole | rolcreatedb | rolcanlogin
---------------------------+----------+------------+---------------+-------------+-------------
 pg_database_owner         | f        | t          | f             | f           | f
 pg_read_all_data          | f        | t          | f             | f           | f
 pg_write_all_data         | f        | t          | f             | f           | f
 pg_monitor                | f        | t          | f             | f           | f
 pg_read_all_settings      | f        | t          | f             | f           | f
 pg_read_all_stats         | f        | t          | f             | f           | f
 pg_stat_scan_tables       | f        | t          | f             | f           | f
 pg_read_server_files      | f        | t          | f             | f           | f
 pg_write_server_files     | f        | t          | f             | f           | f
 pg_execute_server_program | f        | t          | f             | f           | f
 pg_signal_backend         | f        | t          | f             | f           | f
 pg_checkpoint             | f        | t          | f             | f           | f
(12 rows)
```

Son 12 roles creados por defecto y estos tienen como utilidad ser roles de administración.

- **C2.-** Captura de pantalla del resultado de la ejecución del comando \dn y respuesta a *¿qué diferencia encuentras con la vista information_schema.views?*
  La principal diferencia que noto es en la informacion proporcionada por los comandos, dado que el comando "\dn" se limita a mostrar los esquemas contenedores de la base de datos, mientras que a vista "information_schema.views" nos muestra un catálogo exhaustivo de todas las vistas (objetos) definidas dentro de esos esquemas o por el propio sistema.

```
lbd09_2026_2_321162892=> CREATE SCHEMA "P03" AUTHORIZATION "lbd09_2026_2_321162892";
CREATE SCHEMA
lbd09_2026_2_321162892=> \dn
         List of schemas
  Name  |         Owner
--------+------------------------
 P03    | lbd09_2026_2_321162892
 P2     | lbd09_2026_2_321162892
 public | pg_database_owner
(3 rows)
```

- **C3.-** Sentencias SQL para la creación de tablas de pruebas y una captura de pantalla del resultado.

```
lbd09_2026_2_321162892=> CREATE TABLE "P03".tabla_lectura_22 (id INTEGER NOT NULL);
CREATE TABLE
lbd09_2026_2_321162892=> CREATE TABLE "P03".tabla_escritura_22 (nombre VARCHAR(100) NOT NULL);
CREATE TABLE
lbd09_2026_2_321162892=>
```

- **C4.-** Captura de pantalla del resultado de la creación del rol y respuesta a *¿qué pasaría si el usuario postgres no hubiera ejecutado este comando ALTER USER lbd09_2026_2_<num_cuenta> WITH CREATEROLE;?*

```
lbd09_2026_2_321162892=> CREATE ROLE read_write_321162892;
CREATE ROLE
lbd09_2026_2_321162892=>
```

  Al intentar ejecutar la sentencia "CREATE ROLE" el sistema responderia con un error del tipo "insufficient privileges", siendo que no contamos con los privilegios sufientes.

- **C5.-** Captura de pantalla del resultado del permiso de conexión a la base de datos y respuesta a *¿es necesario otorgar algún privilegio con el superusuario para realizar esta acción?*

```
lbd09_2026_2_321162892=> GRANT CONNECT ON DATABASE lbd09_2026_2_321162892 TO read_write_321162892;
GRANT
lbd09_2026_2_321162892=>
```

  No es necesario, debido a que es propietario de la base de datos el usuario al que estamos otorgando permisos

- **C6.-** Captura de pantalla de la comprobación del rol de lectura y escritura creado. Respuesta a *¿qué pasaría si tu compañer@ realizara una inserción y/o consulta en la tabla "P01".estudiante?*
  Ocurriría un error ya que no se han definido permisos específicos para esta acción.

```
pipardo@pipardo-Predator-PHN16-71:~$ psql -h 132.248.59.8 -p 5432 -U lbd09_2026_2_321162892 -d lbd09_2026_2_321162892 -d lbd09_2026_2_
315197815
Password for user lbd09_2026_2_321162892:
psql (16.11 (Ubuntu 16.11-0ubuntu0.24.04.1), server 15.4)
Type "help" for help.

lbd09_2026_2_315197815=> INSERT INTO "P03".tabla_escritura_0 VALUES('Angel')
lbd09_2026_2_315197815-> ;
INSERT 0 1
lbd09_2026_2_315197815=> SELECT * FROM "P03".tabla_escritura_0;
   nombre
------------
 Angel
 Angel
 ANGEL BRITO
(3 rows)

lbd09_2026_2_315197815=>
```

- **C7.-** Sentencias SQL para crear el rol de sólo lectura. Capturas de pantalla de una consulta a tabla_lectura_<num_lista> en el entorno de tu compañer@, así como una prueba de que sólo tiene permisos de lectura.

```
lbd09_2026_2_321162892=> CREATE ROLE readonly_321162892;
ERROR:  el rol «readonly_321162892» ya existe
lbd09_2026_2_321162892=> GRANT CONNECT ON DATABASE lbd09_2026_2_321162892 TO readonly_321162892;
GRANT
lbd09_2026_2_321162892=> GRANT USAGE ON SCHEMA "P03" TO readonly_321162892;
GRANT
lbd09_2026_2_321162892=> GRANT SELECT ON TABLE "P03".tabla_lectura_22" TO readonly_321162892;
lbd09_2026_2_321162892"> GRANT readonly_321162892 TO lbd09_2026_2_315197815;
lbd09_2026_2_321162892">
```

```
pipardo@pipardo-Predator-PHN16-71:~$ psql -h 132.248.59.8 -p 5432 -U lbd09_2026_2_321162892 -d lbd09_2026_2_315197815
Password for user lbd09_2026_2_321162892:
psql (16.11 (Ubuntu 16.11-0ubuntu0.24.04.1), server 15.4)
Type "help" for help.

lbd09_2026_2_315197815=> SELECT*FROM "P03".tabla_lectura_0;
 id
----
  0
(1 row)
```

- **C8.-** Sentencias SQL para eliminar los privilegios otorgados a tu compañer@ y capturas de pantalla de la ejecución de los comandos, así como ejecución de consultas no exitosas en el entorno de tu compañer@ por falta de permisos.

```
lbd09_2026_2_321162892=> REVOKE read_write_321162892 FROM lbd09_2026_2_315197815;
REVOKE ROLE
lbd09_2026_2_321162892=> REVOKE readonly_321162892 FROM lbd09_2026_2_315197815;
REVOKE ROLE
lbd09_2026_2_321162892=>
```

Consulta no exitosa:

```
lbd09_2026_2_315197815=> SELECT*FROM "P03".tabla_lectura_0;
ERROR:  permiso denegado a la tabla tabla_lectura_0
```

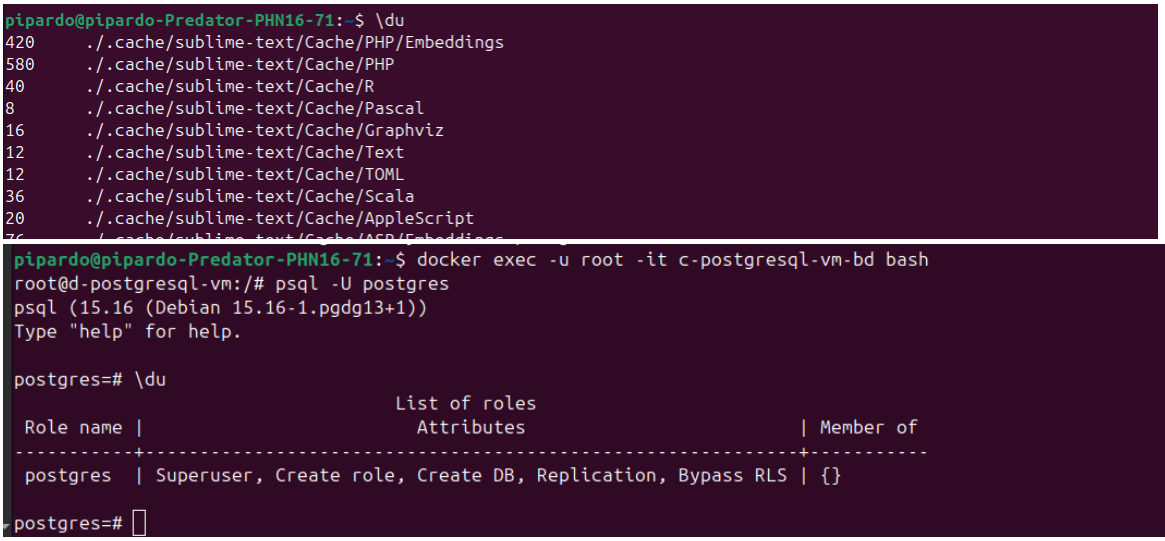
- **C9.-** Captura de pantalla del estatus del servidor de PostgreSQL.

```
pipardo@pipardo-Predator-PHN16-71:~$ docker build -t="postgresql15-vm" .
[+] Building 1.0s (5/5) FINISHED                                 docker:default
 => [internal] load build definition from Dockerfile                       0.0s
 => => transferring dockerfile: 95B                                        0.0s
 => [internal] load metadata for docker.io/library/postgres:15             0.6s
 => [internal] load .dockerignore                                          0.0s
 => => transferring context: 2B                                            0.0s
 => CACHED [1/1] FROM docker.io/library/postgres:15@sha256:2bc89eed549096  0.0s
 => => resolve docker.io/library/postgres:15@sha256:2bc89eed5490967e6b1fa  0.0s
 => exporting to image                                                     0.2s
 => => exporting layers                                                    0.0s
 => => exporting manifest sha256:04b56e2ec46344adbd63696f0e1f44bd500f34c5  0.0s
 => => exporting config sha256:d10ebbf4e6b7d456b8d90c1a19328fb66317fb8b12  0.0s
 => => exporting attestation manifest sha256:a27cdee2d145ca80f803ffc33085  0.0s
 => => exporting manifest list sha256:9dbaa371fa41366df39675476aa04cb85af  0.0s
 => => naming to docker.io/library/postgresql15-vm:latest                  0.0s
 => => unpacking to docker.io/library/postgresql15-vm:latest               0.0s

 1 warning found (use docker --debug to expand):
 - SecretsUsedInArgOrEnv: Do not use ARG or ENV instructions for sensitive data
(ENV "POSTGRES_PASSWORD") (line 3)
```

- **C10.-** Captura de pantalla de inicio de sesión exitoso con el usuario postgres a nivel SO.

```
pipardo@pipardo-Predator-PHN16-71:~$ sudo ls -la /u01/pgdata/15/data
[sudo] password for pipardo:
total 136
drwx------ 19 dnsmasq tape             4096 Feb 25 00:00 .
drwxr-xr-x  3 root    root             4096 Feb 17 08:14 ..
drwx------  6 dnsmasq systemd-journal  4096 Feb 25 00:13 base
drwx------  2 dnsmasq systemd-journal  4096 Feb 25 00:19 global
drwx------  2 dnsmasq systemd-journal  4096 Feb 24 23:58 pg_commit_ts
drwx------  2 dnsmasq systemd-journal  4096 Feb 24 23:58 pg_dynshmem
-rw-------  1 dnsmasq systemd-journal  4821 Feb 24 23:58 pg_hba.conf
-rw-------  1 dnsmasq systemd-journal  1636 Feb 24 23:58 pg_ident.conf
drwx------  4 dnsmasq systemd-journal  4096 Feb 25 00:15 pg_logical
drwx------  4 dnsmasq systemd-journal  4096 Feb 24 23:58 pg_multixact
drwx------  2 dnsmasq systemd-journal  4096 Feb 24 23:58 pg_notify
drwx------  2 dnsmasq systemd-journal  4096 Feb 24 23:58 pg_replslot
drwx------  2 dnsmasq systemd-journal  4096 Feb 24 23:58 pg_serial
drwx------  2 dnsmasq systemd-journal  4096 Feb 24 23:58 pg_snapshots
drwx------  2 dnsmasq systemd-journal  4096 Feb 24 23:58 pg_stat
drwx------  2 dnsmasq systemd-journal  4096 Feb 24 23:58 pg_stat_tmp
drwx------  2 dnsmasq systemd-journal  4096 Feb 24 23:58 pg_subtrans
drwx------  2 dnsmasq systemd-journal  4096 Feb 24 23:58 pg_tblspc
drwx------  2 dnsmasq systemd-journal  4096 Feb 24 23:58 pg_twophase
-rw-------  1 dnsmasq systemd-journal     3 Feb 24 23:58 PG_VERSION
drwx------  3 dnsmasq systemd-journal  4096 Feb 24 23:58 pg_wal
drwx------  2 dnsmasq systemd-journal  4096 Feb 24 23:58 pg_xact
-rw-------  1 dnsmasq systemd-journal    88 Feb 24 23:58 postgresql.auto.conf
-rw-------  1 dnsmasq systemd-journal 29517 Feb 24 23:58 postgresql.conf
-rw-------  1 dnsmasq systemd-journal    36 Feb 25 00:00 postmaster.opts
-rw-------  1 dnsmasq systemd-journal    94 Feb 25 00:00 postmaster.pid
pipardo@pipardo-Predator-PHN16-71:~$
```

- **C11.-** Captura de pantalla del resultado de la ejecución del comando \du.



Diferencias entre los métodos de autenticación trust, peer y md5 de acuerdo con la documentación oficial (auth-method).

Trust.

Este es el método más permisivo, como es indicado por su nombre, el servidro confia ciegamente en que cualquier usuario que intente conectarse ya está autorizado para acceder con el nombre de usuario de base de datos que proporcione, suele ser empleado en entornos de desarrollo muy controlados o mediante conexiones de socket locales de Unix si el sistema operativo ya restringe el acceso, comúnmente se usa para pruebas iniciales o conexiones locales dónde la seguridad del sistema ya es absoluta.

Peer.

Este método está basado en la identidad del sistema operativo, el servidor obtiene el nombre del usuario que inicio la conexión a nivel de kernel (por medio del socket de Unix) y verifica si coincide con el nombre del usuario a base de datos solicitado, este solo funciona en conexiones locales por medio de Unix Domanin Sockets, no es compatible con conexiones TCP/IP (red), en resumen si estamos logeados en Linux con un usuario “alumno” e intentamos entrar a la base de datos con el usuario “alumno”, el sistema nos dejara pasar en automático son pedir alguna contraseña dado que el kernel ya valido quienes somos.

md5.

Este es un método que está basado en contraseñas compartidas que utiliza un hash para así proteger la credencial durante él envió, el cliente deberá proporcionar una contraseña, en la cual se empleara un algoritmo de hashing (MD5) antes de ser enviado por la red, evitando así que se transmita en texto plano, las conexiones son por medio de la red (TCP/IP).

- **C12.-** Captura de pantalla del resultado de la ejecución del comando \l.

```
pipardo@pipardo-Predator-PHN16-71:~$ docker exec -u root -it c-postgresql-vm-bd bash
root@d-postgresql-vm:/#
```

```
postgres=# \list
                                                 List of databases
     Name     |   Owner    | Encoding |  Collate   |   Ctype    | ICU Locale | Locale Provider |   Access privileges
--------------+------------+----------+------------+------------+------------+-----------------+-----------------------
 lab_bd_09_22 | developers | UTF8     | en_US.utf8 | en_US.utf8 |            | libc            |
 postgres     | postgres   | UTF8     | en_US.utf8 | en_US.utf8 |            | libc            |
 template0    | postgres   | UTF8     | en_US.utf8 | en_US.utf8 |            | libc            | =c/postgres          +
              |            |          |            |            |            |                 | postgres=CTc/postgres
 template1    | postgres   | UTF8     | en_US.utf8 | en_US.utf8 |            | libc            | =c/postgres          +
              |            |          |            |            |            |                 | postgres=CTc/postgres
(4 rows)

postgres=#
```

- **C13.-** Captura de pantalla de la conexión a la base de datos lab_bd_09_<num_lista> con el usuario administrador.

```
lab_bd_09_22=> \du
                                   List of roles
 Role name  |                         Attributes                         | Member of
------------+------------------------------------------------------------+-----------
 developers |                                                            | {}
 postgres   | Superuser, Create role, Create DB, Replication, Bypass RLS | {}
 root       |                                                            | {}

lab_bd_09_22=>
```

```
postgres=#  \du
                                   List of roles
 Role name  |                         Attributes                         | Member of
------------+------------------------------------------------------------+-----------
 developers |                                                            | {}
 postgres   | Superuser, Create role, Create DB, Replication, Bypass RLS | {}
 root       |                                                            | {}

postgres=#
```

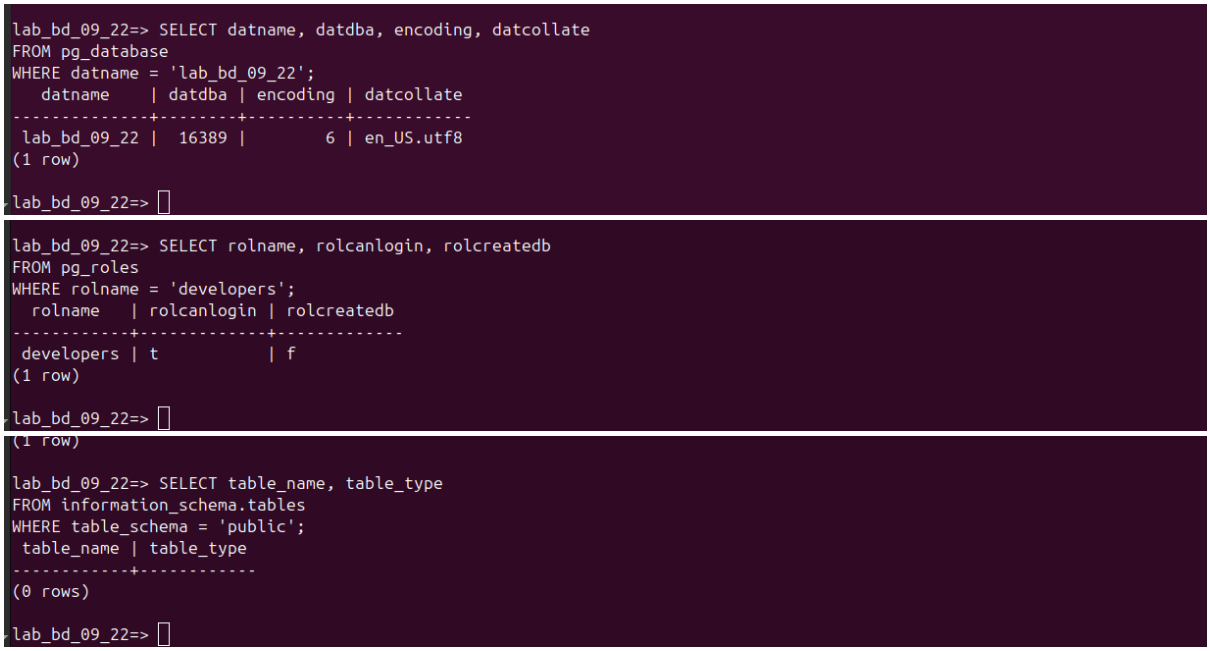
- **C14.-** Diferencia entre los métodos de autenticación (auth-method) peer, scram-sha-256 y trust.

```
postgres=# \list
                                                 List of databases
     Name     |  Owner   | Encoding |  Collate   |   Ctype    | ICU Locale | Locale Provider |   Access privileges
--------------+----------+----------+------------+------------+------------+-----------------+-----------------------
 lab_bd_09_22 | root     | UTF8     | en_US.utf8 | en_US.utf8 |            | libc            |
 postgres     | postgres | UTF8     | en_US.utf8 | en_US.utf8 |            | libc            |
 template0    | postgres | UTF8     | en_US.utf8 | en_US.utf8 |            | libc            | =c/postgres          +
              |          |          |            |            |            |                 | postgres=CTc/postgres
 template1    | postgres | UTF8     | en_US.utf8 | en_US.utf8 |            | libc            | =c/postgres          +
              |          |          |            |            |            |                 | postgres=CTc/postgres
(4 rows)

postgres=#
```

- **C15.-** Captura de pantalla de la conexión a la base de datos lab_bd_09_<num_lista> con el usuario developers.

```
pipardo@pipardo-Predator-PHN16-71:~$ docker exec -u root -it c-postgresql-vm-bd bash
root@d-postgresql-vm:/# psql -d lab_bd_09_22
psql (15.16 (Debian 15.16-1.pgdg13+1))
Type "help" for help.

lab_bd_09_22=>
```

- **C16.-** Captura de pantalla de la ejecución de las consultas a las *System Views*.



Conclusiones.

Tras la realización de esta práctica puedo concluir que se lograron por completo los objetivos prestablecidos al inicio de está siendo que se consiguió emplear comandos de control de datos para crear usuarios, otorgar y revocar permisos, en la cual agradezco la ayuda proporcionada por el profesor siendo que fue el que me ayudo para la comprobación de los comandos para otorgar y revocar permisos, para la segunda parte de la practica la cual consistía en la creación de una base de datos en nuestro entorno, me fue un poco complicado y tuve un par de complicaciones dado que no estaba bien instalado mi Docker en el directorio correcto, fuera de esto me fue bastante educativo y didáctico el realizar la instalación en mi propio entorno, realmente me es de utilidad lo visto en esta práctica ya que pienso que estos conocimientos los podre emplear en mi vida laboral al igual que en prácticas profesionales o mi servicio social.

Bibliografía.

1. *PostgreSQL: documentation*. (s. f.). The PostgreSQL Global Development Group. https://www.postgresql.org/docs/
2. Inc, D. (2026c, febrero 5). *Home*. Docker Documentation. https://docs.docker.com/
3. Segovia, J., & Segovia, J. (2021, 8 marzo). Cómo crear una base de datos en PostgreSQL - TodoPostgreSQL. *TodoPostgreSQL - Academia Online de PostgreSQL en Español*. https://www.todopostgresql.com/como-crear-base-de-datos-en-postgresql/