Universidad Nacional Autónoma de México

Facultad de Ciencias

Fundamentos de Bases de Datos - 7094

P R Á C T I C A 5

Equipo:

Del Monte Ortega Maryam Michelle - 320083527

Sosa Romo Juan Mario - 320051926

Castillo Hernández Antonio - 320017438

Erik Eduardo Gómez López - 320258211

Julio César Islas Espino - 320340594

Fecha de entrega:
04 de Octubre de 2024

Profesor:
M. en I. Gerardo Avilés Rosas

Ayudantes:
Luis Enrique García Gómez
Kevin Jair Torres Valencia
Ricardo Badillo Macías
Rocío Aylin Huerta González

Restauración del .backup

Introducción

En esta sección, se detalla el proceso de restauración de una base de datos en un entorno de contenedores Docker utilizando PostgreSQL. Se utilizó el archivo de respaldo `transporte.backup` proporcionado por el ayudante para restaurar una base de datos de ejemplo. Este informe documenta los pasos seguidos para lograr una restauración exitosa, así como las evidencias que validan el proceso, *(forma parte del punto IV de la entrega)*.

En escencia se siguieron los mismos pasos que se especifican en el documento de *RestaurarBackupDocker.pdf* .

Pasos Realizados:

1. Verificación de Contenedores

Inicialmente verificamos los contenedores disponibles en Docker utilizando el comando:

```
docker ps -a
```

```
..[abovewolf37845 MSI] - [~] - [sáb oct 05, 10:24]
..[$] <()> sudo docker ps -a
[sudo] password for abovewolf37845:
CONTAINER ID   IMAGE      COMMAND                  CREATED       STATUS                     PORTS
462aae37ea52   postgres   "docker-entrypoint.s…"   5 weeks ago   Exited (0) 5 weeks ago
9e16abf2a1b6   postgres   "docker-entrypoint.s…"   5 weeks ago   Created                    5432/tcp
b62921ed1f10   postgres   "docker-entrypoint.s…"   5 weeks ago   Created                    5432/tcp
a9e75cb18d62   postgres   "docker-entrypoint.s…"   5 weeks ago   Exited (255) 5 weeks ago   0.0.0.0:543
```

Despues de haber verificado los contenedores que había en el sistema, se procedió a obtener el ID de alguna de las imagenes de PostgreSQL que ya se encontraban creadas en el mismo. En este caso, se utilizó la imagen con ID `a9e75cb18d62` y procedimos a reiniciar el contenedor ya que había pasado mucho tiempo desde que se había creado.

```
..[$] <()> sudo docker start a9e75cb18d62
a9e75cb18d62
```

Checamos el estatus del contenedor con el comando `docker ps -a`.

```
..[$] <()> sudo docker ps
CONTAINER ID   IMAGE      COMMAND                  CREATED       STATUS         PORTS
a9e75cb18d62   postgres   "docker-entrypoint.s…"   5 weeks ago   Up 8 seconds   0.0.0.0:5432->5432/tcp,
```

2. Conexión al Contenedor

Una vez reiniciado el contenedor, procedimos a conectarnos a él utilizando el comando:

```
sudo docker exec -it <CONTAINER_ID> psql -U postgres
```

lo cual nos mete dentro de psql, una vez detro ejecutamos ℓ donde podemos observar lo siguiente:

salimos con `\q`.

```
..[$] <()> sudo docker exec -it a9e75cb18d62 psql -U postgres
psql (16.4 (Debian 16.4-1.pgdg120+1))
Type "help" for help.

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

3. Restauración del .backup

Como bien se menciona en el PDF, para hacer un backup necesitamos asegurarnos de que la base de datos esté vacía para evitar posibles conflictos con tablas repetidas. Entonces creamos una nueva base de datos a la cual llamaremos en esta ocasión *transporte* con el ID del contenedor que hemos estado utilizando de la siguiente manera:

```
sudo docker exec -it <CONTAINER_ID> createdb -U postgres ejemplo
```

```
..[$] <()> sudo docker exec -it a9e75cb18d62 createdb -U postgres transporte
[sudo] password for abovewolf37845:
..[abovewolf37845 MSI] - [~] - [dom oct 06, 12:51]
..[$] <()> sudo docker exec -it a9e75cb18d62 psql -U postgres transporte
psql (16.4 (Debian 16.4-1.pgdg120+1))
Type "help" for help.

transporte=# \dt
Did not find any relations.
transporte=#
```

Como se puede observar ejecutamos $\backslash dt$ para checar el contenido de la base de datos y vemos que efectivamente está vacía. De esta manera procedemos a la restauración del backup que en nuestro caso se encuentra en el archivo *transporte.backup*. utilizamos el siguiente comando.

```
sudo docker exec -i <CONTAINER_ID> pg_restore --verbose --clean --no-acl --no-owner -U
postgres -d transporte < /home/abovewolf37845/Downloads/backup/transporte.backup
```

Notemos que en lugar de utilizar *psql* utilizamos *pg_ restore* ya que parece que *psql* solamente para archivos sql y *psql* para dump files.

```
..[$] <()> sudo docker exec -i a9e75cb18d62 pg_restore --verbose --clean --no-acl --no-owner
[sudo] password for abovewolf37845:
pg_restore: connecting to database for restore
pg_restore: dropping FK CONSTRAINT trolebus trolebus_fkey
pg_restore: dropping FK CONSTRAINT trenligero trenligero_fkey
pg_restore: dropping FK CONSTRAINT tipotransporte tipotransporte_fkey
pg_restore: dropping FK CONSTRAINT telefono telefono_fkey
pg_restore: dropping FK CONSTRAINT taxi taxi_fkey
pg_restore: dropping FK CONSTRAINT tarjeta tarjeta_fkey
pg_restore: dropping FK CONSTRAINT rtp rtp_fkey
pg_restore: dropping FK CONSTRAINT reparartrolebus reparartrolebus_fkey2
pg_restore: dropping FK CONSTRAINT reparartrolebus reparartrolebus_fkey1
pg_restore: dropping FK CONSTRAINT reparartrenligero reparartrenligero_fkey2
pg_restore: dropping FK CONSTRAINT reparartrenligero reparartrenligero_fkey1
pg_restore: dropping FK CONSTRAINT reparartaxi reparartaxi_fkey2
pg_restore: dropping FK CONSTRAINT reparartaxi reparartaxi_fkey1
pg_restore: dropping FK CONSTRAINT repararrtp repararrtp_fkey2
pg_restore: dropping FK CONSTRAINT repararrtp repararrtp_fkey1
pg_restore: dropping FK CONSTRAINT repararmicrobus repararmicrobus_fkey2
pg_restore: dropping FK CONSTRAINT repararmicrobus repararmicrobus_fkey1
pg_restore: dropping FK CONSTRAINT repararmetrobus repararmetrobus_fkey2
pg_restore: dropping FK CONSTRAINT repararmetrobus repararmetrobus_fkey1
pg_restore: dropping FK CONSTRAINT repararmetro repararmetro_fkey2
pg_restore: dropping FK CONSTRAINT repararmetro repararmetro_fkey1
pg_restore: dropping FK CONSTRAINT pagartrolebus pagartrolebus_fkey2
```

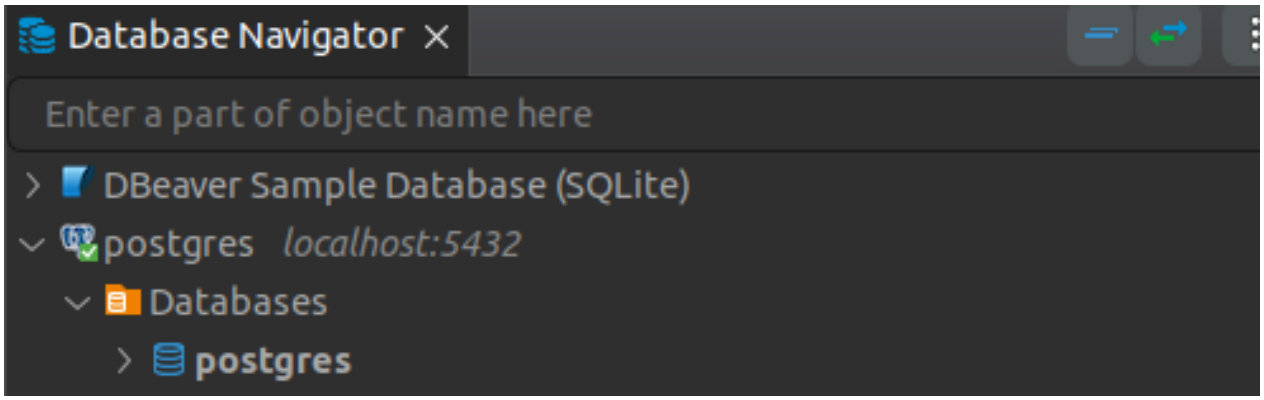
Después verificamos que la base de datos se haya restaurado correctamente con $\backslash dt$ dentro de psql donde veríamos algo como lo siguiente:

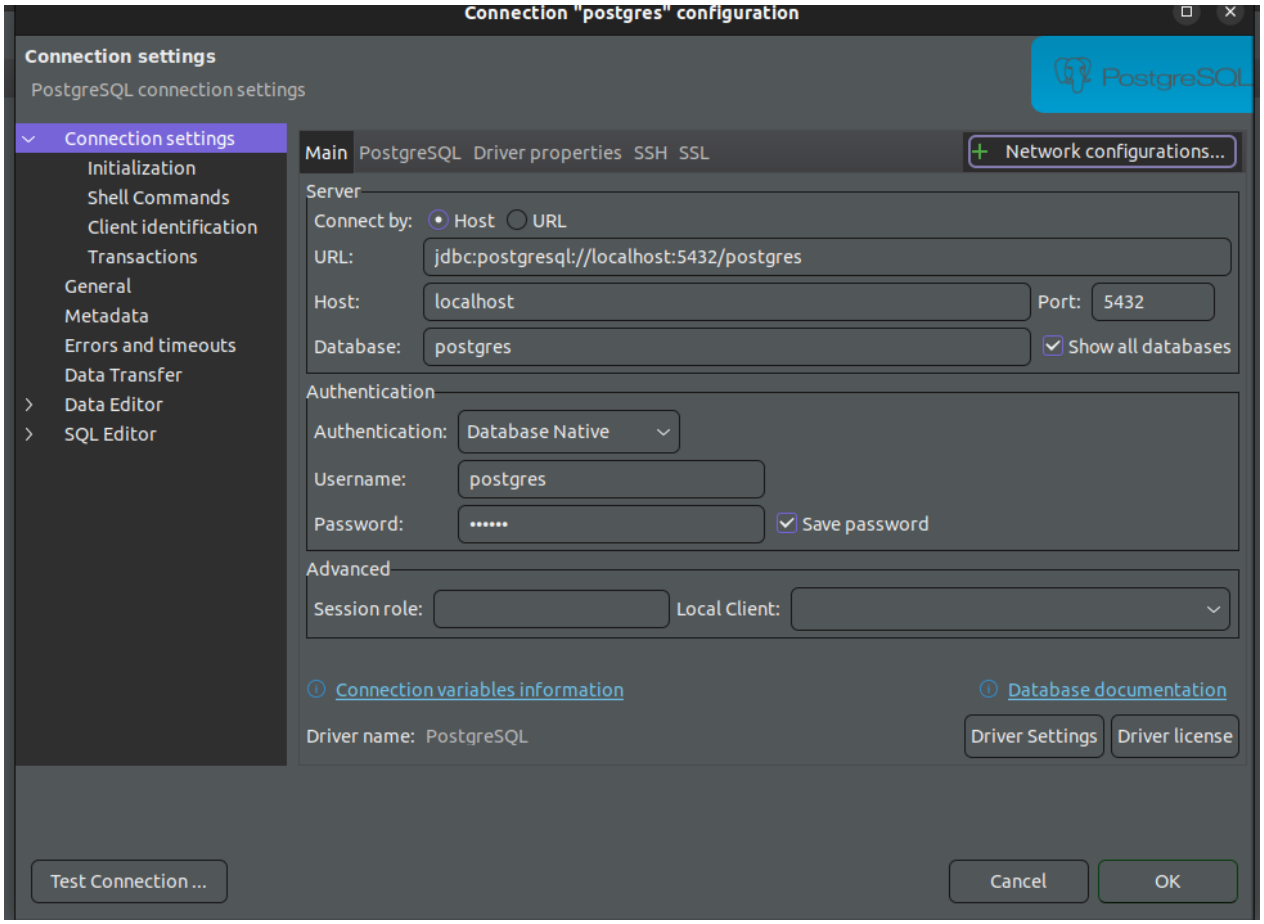
```
..[$] <()> sudo docker exec -it a9e75cb18d62 psql -U postgres transporte
psql (16.4 (Debian 16.4-1.pgdg120+1))
Type "help" for help.

transporte=# \dt
              List of relations
 Schema |       Name        | Type  |  Owner
--------+-------------------+-------+----------
 public | componerse        | table | postgres
 public | contar            | table | postgres
 public | deshabilitar      | table | postgres
 public | dia               | table | postgres
 public | especialidad      | table | postgres
 public | estacion          | table | postgres
 public | examenmedico      | table | postgres
 public | incidente         | table | postgres
 public | licencia          | table | postgres
 public | linea             | table | postgres
 public | manejarmetro      | table | postgres
 public | manejarmetrobus   | table | postgres
```

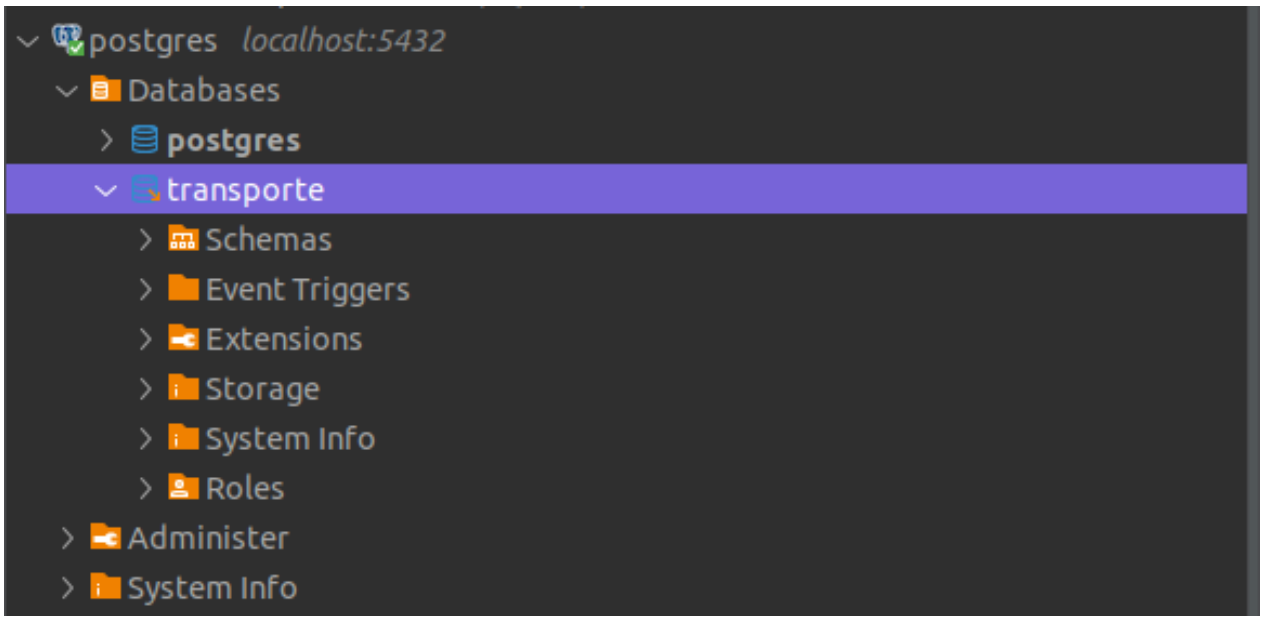
Lo cual nos indica que restauramos la base de datos correctamente.

Para ver las bases de datos que se crearon en Dbeaver

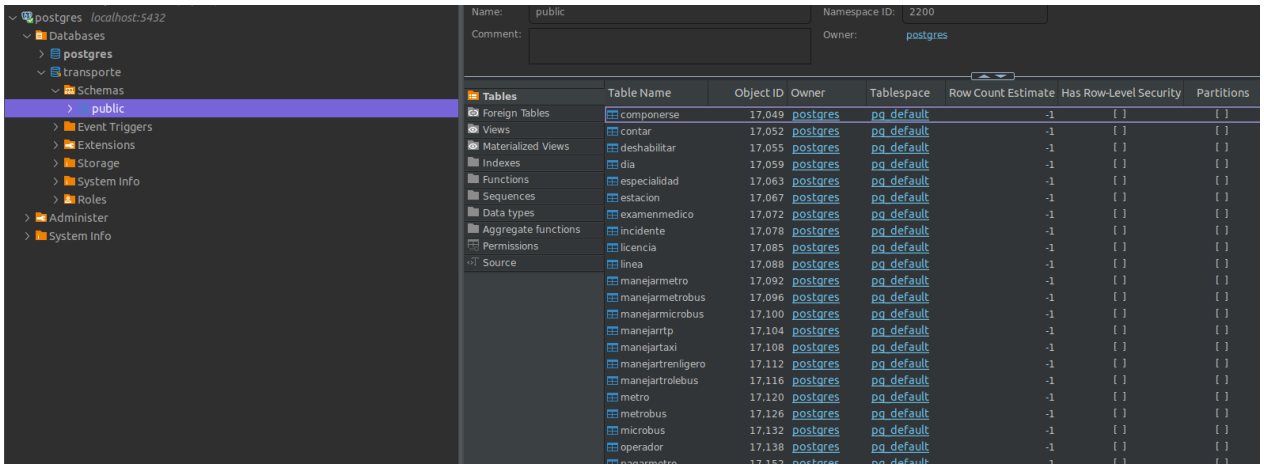
Seguimos las instrucciones para configurar correctamente el Database Tool DBeaver.







Ahora podemos ver las bases de datos que se crearon en DBeaver.



Tarea 1

Especificacion de reestriciones:

1.