Práctica 12

Particiones declarativas con PostgreSQL

Objetivo

En esta práctica se conocen la manera en que se definen las particiones de tablas utilizando psql.

Introducción

PostgreSQL permite modificar una tabla existente para particionarla en diversos segmentos, utilizando una variedad de posibilidades, ya sea por listas de valores, rangos de valores o mediante un Hash. Incluso es posible definir subparticiones. Posteriormente se pude trabajar con la tabla íntegra o con cualquiera de sus partes, según se requiera.

Equipo necesario

Una computadora con sistema operativo ***Windows*** que cuente con ***psql***.

Metodología

1. En el menú de programas de Windows, localice y ejecute la aplicación ***psql***.
2. Entre con el usuario ***postgres***.
3. Escriba la contraseña y oprima ***enter***.
4. Cree la base de datos ***pedidos***:

   ```
   postgres=# CREATE DATABASE pedidos;
   ```

5. Seleccione a la base de datos ***pedidos***:

   ```
   postgres=# \c pedidos
   ```

6. Una vez dentro, se crea la tabla *Maestra* llamada pedidos, esta tabla no contendrá datos, así como ningún tipo de restricción, pero indicando la columna sobre la que se hará la partición.

   ```
   pedidos=# CREATE TABLE pedidos (num INT, fecha DATE,
   solicitante INT, producto VARCHAR(6), talla INT, color
   ```

```
VARCHAR(10), cant INT) PARTITION BY RANGE (fecha);
```

7. Ahora proceda a crear las cuatro tablas hijas que se usarán en este ejemplo, correspondientes a los años en que se tienen registrados pedidos, la cuales heredan todos los atributos de la tabla pedidos, y además incluyen una condición CHECK para los rangos de valores de fecha.

```
pedidos=# CREATE TABLE pedidos_2022 PARTITION OF pedidos FOR
VALUES FROM ('2022-01-01') TO ('2022-12-31');
pedidos=# CREATE TABLE pedidos_2023 PARTITION OF pedidos FOR
VALUES FROM ('2023-01-01') TO ('2023-12-31');
pedidos=# CREATE TABLE pedidos_2024 PARTITION OF pedidos FOR
VALUES FROM ('2024-01-01') TO ('2024-12-31');
pedidos=# CREATE TABLE pedidos_otros PARTITION OF pedidos
DEFAULT;
```

8. Obtenga una imagen completa de la pantalla, mostrando la creación de estas tablas y consérvela como evidencia.

9. El siguiente paso es crear las llaves primarias para cada una de las tablas hijas:

```
pedidos=# ALTER TABLE pedidos_2022 ADD PRIMARY KEY (num);
pedidos=# ALTER TABLE pedidos_2023 ADD PRIMARY KEY (num);
pedidos=# ALTER TABLE pedidos_2024 ADD PRIMARY KEY (num);
pedidos=# ALTER TABLE pedidos_otros ADD PRIMARY KEY (num);
```

10. Ahora proceda a insertar algunos registros en la tabla producción:

```
pedidos=# INSERT INTO pedidos VALUES (1,'2020-10-01',12,'MO-
21',32,'Azul',8), (2,'2021-12-17',20,'MO-21',40,'Blanco',10),
(3,'2022-06-10',15,'ME-14',30,'Verde',3), (4,'2023-02-14',12,
'ME-14',36,'Rojo',11), (5,'2023-07-21',15,'ME-14', 28,'Azul',
8), (6,'2024-09-11',23,'MO-21',42,'Negro',15);
```

11. Consulte las 5 tablas:

```
pedidos=# SELECT * FROM pedidos;
pedidos=# SELECT * FROM pedidos_2022;
pedidos=# SELECT * FROM pedidos_2023;
pedidos=# SELECT * FROM pedidos_2024;
pedidos=# SELECT * FROM pedidos_otros;
```

12. Obtenga una imagen completa de la pantalla, mostrando los resultados de estas consultas y consérvela como evidencia.

13. Ahora se crea una tabla maestra cliente que contiene una partición mediante una lista de valores:

```
pedidos=# CREATE TABLE cliente (clave INT, nombre VARCHAR(20),
apellido VARCHAR(20), ciudad VARCHAR(20)) PARTITION BY LIST
(ciudad);
```

14. Y ahora se procede a crear las tablas hijas:

```
pedidos=# CREATE TABLE cliente_uruapan PARTITION OF cliente FOR
VALUES IN ('Uruapan');
pedidos=# CREATE TABLE cliente_morelia PARTITION OF cliente FOR
VALUES IN ('Morelia');
pedidos=# CREATE TABLE cliente_zamora PARTITION OF cliente FOR
VALUES IN ('Zamora');
pedidos=# CREATE TABLE cliente_otro PARTITION OF cliente
DEFAULT;
```

15. Obtenga una imagen completa de la pantalla, mostrando la creación de estas tablas y consérvela como evidencia.

16. Ahora, defina las llaves primarias:

```
pedidos=# ALTER TABLE cliente_uruapan ADD PRIMARY KEY (clave);
pedidos=# ALTER TABLE cliente_morelia ADD PRIMARY KEY (clave);
pedidos=# ALTER TABLE cliente_zamora ADD PRIMARY KEY (clave);
pedidos=# ALTER TABLE cliente_otro ADD PRIMARY KEY (clave);
```

17. A continuación, se insertan algunos registros en ***cliente***:

```
pedidos=# INSERT INTO cliente VALUES (12, 'Juan', 'Cano',
'Zamora'), (15, 'Luis', 'Mota', 'Morelia'), (17, 'José',
'Bravo', 'Zacapu'), (20, 'Laura', 'Ruiz', 'Uruapan'), (23,
'Rosa', 'Torres', 'Uruapan');
```

18. Consulte las 4 tablas:

```
pedidos=# SELECT * FROM cliente;
pedidos=# SELECT * FROM cliente_uruapan;
pedidos=# SELECT * FROM cliente_morelia;
pedidos=# SELECT * FROM cliente_zamora;
pedidos=# SELECT * FROM cliente_otro;
```

19. Obtenga una imagen completa de la pantalla, mostrando los resultados de estas consultas y consérvela como evidencia.

20. El tercer ejemplo es para cuando se declara una partición mediante HASH, para ello debe crear la tabla maestra ***producto***:

```
pedidos=# CREATE TABLE producto (código VARCHAR(6), nombre
VARCHAR(25), talla INT, color VARCHAR(15), existencia INT)
PARTITION BY HASH(existencia);
```

21. Ahora defina cuatro tablas hijas, mediante una función hash basada en el cálculo del módulo(4):

```
pedidos=# CREATE TABLE producto_p1 PARTITION OF producto FOR
VALUES WITH (MODULUS 4, REMAINDER 0);
pedidos=# CREATE TABLE producto_p2 PARTITION OF producto FOR
VALUES WITH (MODULUS 4, REMAINDER 1);
pedidos=# CREATE TABLE producto_p3 PARTITION OF producto FOR
VALUES WITH (MODULUS 4, REMAINDER 2);
```

```
pedidos=# CREATE TABLE producto_p4 PARTITION OF producto FOR
VALUES WITH (MODULUS 4, REMAINDER 3);
```

22. Como en los ejemplos previos, se define una llave primaria en las tablas hijas:

```
pedidos=# ALTER TABLE producto_p1 ADD PRIMARY KEY (código,
talla, color);
pedidos=# ALTER TABLE producto_p2 ADD PRIMARY KEY (código,
talla, color);
pedidos=# ALTER TABLE producto_p3 ADD PRIMARY KEY (código,
talla, color);
pedidos=# ALTER TABLE producto_p4 ADD PRIMARY KEY (código,
talla, color);
```

23. Y se procede a insertar algunos productos:

```
pedidos=# INSERT INTO producto VALUES ('MO-21', 'Playera', 32,
'Azul', 23), ('MO-21', 'Playera', 34, 'Rojo', 10), ('MO-21',
'Playera', 40, 'Blanco', 18),('MO-21', 'Playera', 42, 'Negro',
30), ('ME-14', 'Short', 30, 'Verde', 14), ('ME-14', 'Short',
28, 'Azul', 21), ('ME-14', 'Short', 38, 'Amarillo', 30),('ME-
14', 'Short', 36, 'Rojo', 22);
```

24. Por último, consulte estas 5 tablas.

```
pedidos=# SELECT * FROM producto;
pedidos=# SELECT * FROM producto_p1;
pedidos=# SELECT * FROM producto_p2;
pedidos=# SELECT * FROM producto_p3;
pedidos=# SELECT * FROM producto_p4;
```

25. Obtenga una imagen completa de la pantalla, mostrando los resultados de estas consultas y consérvela como evidencia.

26. Cierre ambas sesiones de psql.

27. Fin de la Práctica

Evidencias

El alumno deberá enviar al instructor las evidencias requeridas durante la realización de la práctica.

Sugerencias didácticas

El instructor deberá atender a los alumnos que tengan dificultades en la interpretación y la realización de las instrucciones de la práctica.

Resultados

Se aprendió a crear tablas con diferentes tipos de particiones mediante la

herramienta ***psql***.

Bibliografía

- https://www.postgresql.org/docs/current/ddl-partitioning.html