Práctica 18

Niveles de Aislamiento

Objetivo

En esta práctica se realizan ejercicios para diferenciar los distintos niveles de aislamiento en que se pueden ejecutar las transacciones.

Introducción

El aislamiento es una manera efectiva de garantizar que cuando varias transacciones se ejecutan concurrentemente, sus resultados sean consistentes, por eso, una transacción en ejecución no puede revelar sus resultados a otras transacciones concurrentes antes de comprometerse, los resultados de las ejecuciones concurrentes deben ser los mismos que si ellas se ejecutan de manera secuencial.

Los niveles de aislamiento en SQL se basan en lo que la ANSI ha llamado *fenómenos*, que son las situaciones que se pueden presentar si no se tiene el grado de aislamiento necesario, y son de tres tipos:

Lectura Sucia. Es el caso cuando una transacción T_1 modifica un dato y luego otra transacción T_2 lo lee antes de que T_1 haga *commit* o *abort*. En este segundo caso, la lectura de T_2 es incierta.

Lectura Difusa. Es cuando una transacción T_1 lee un dato, luego otra transacción T_2 lo modifica o elimina y hace *commit* antes de que T_1 termine. De modo que si T_1 vuelve a releerlo va a encontrar un valor distinto.

Fantasmas. Es cuando T_1 hace una búsqueda determinada mientras que T_2 inserta nuevos registros a la base de datos que satisfacen los criterios de búsqueda de T_1. Si T_1 repite la búsqueda el resultado será diferente.

Basados en los fenómenos, es posible definir cuatro niveles de aislamiento:

- ***READ UNCOMMITTED*** (Lectura no comprometida). En este nivel se pueden presentar los tres tipos de fenómenos.
- ***READ COMMITTED*** (Lectura comprometida). En este nivel se pueden presentar lecturas difusas y fantasmas.
- ***REPEATABLE READ*** (Lectura repetitiva). En este nivel solamente se pueden

presentar fantasmas.

- ***SERIALIZABLE***. En este nivel no se presenta ninguno de los fenómenos anteriores.

Equipo necesario

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

Metodología

1. Localice y ejecute la herramienta ***SQL Shell (psql)***.
2. Ingrese como el usuario ***postgres*** y escriba la contraseña que definió en la primera práctica.
3. Ahora cambie a la base de datos ***pedidos***:

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

4. El nivel de aislamiento de las transacciones es por default ***READ COMMITTED***, pero es posible cambiar a otro nivel de aislamiento global, para el resto de la sesión actual:
5. En determinadas circunstancias puede ser necesario cambiar los niveles de aislamiento, y se puede hacer mediante la declaración SET TRANSACTION ISOLATION LEVEL, por ejemplo, para cambiar al nivel de aislamiento READ UNCOMMITTED, simplemente se escribe:

   ```
   pedidos=# SET DEFAULT_TRANSACTION_ISOLATION = "READ
   UNCOMMITTED";
   ```

6. Para verificar cuál es el nivel de aislamiento, ejecute la siguiente sentencia:

   ```
   pedidos=# SHOW DEFAULT_TRANSACTION_ISOLATION;
   ```

7. Puede fijarse un determinado nivel de aislamiento para una transacción en curso, incluyendo la declaración SET TRANSACTION ISOLATION LEVEL, por ejemplo:

   ```
   pedidos=# BEGIN;
   pedidos=*# SET TRANSACTION ISOLATION LEVEL READ COMMITTED;
   pedidos=*# SHOW TRANSACTION_ISOLATION;
   pedidos=*# UPDATE cliente SET Apellido='Montes' WHERE Clave=8;
   pedidos=*# DELETE FROM cliente WHERE Clave=18;
   pedidos=*# SELECT * FROM cliente;
   ```

8. Obtenga una imagen completa de la pantalla, mostrando el resultado de estas sentencias y consérvela como evidencia.
9. Deje en suspenso la transacción e inicie una segunda sesión ***psql***, para el usuario ***rosita***, con su contraseña (***Privada***).

10. Ahora cambie a la base de datos ***pedidos***:

```
postgres=> \c pedidos
```

11. E inicie una transacción, en la que se realicen un cambio:

```
pedidos=> BEGIN;
pedidos=*> DELETE FROM cliente WHERE Clave=17;
pedidos=*> INSERT INTO cliente VALUES (25, 'Tito', 'Lara',
'Zamora');
pedidos=*> UPDATE cliente SET Nombre='Jaime' WHERE Clave=8;
```

12. En este momento queda detenida la transacción, en espera de que termine la transacción que tiene el registro (Clave=8) bloqueado.
13. Obtenga una imagen completa de la pantalla, mostrando el resultado de estas sentencias y consérvela como evidencia.
14. Regrese a la conexión del usuario ***postgres***, y añada a su transacción la siguiente sentencia:

```
pedidos=*# UPDATE cliente SET Nombre='Alicia' WHERE Clave=17;
```

15. En este momento el sistema detecta que existe un interbloqueo (***Deadlock***) e inmediatamente aborta a esta transacción para romperlo, liberando a su vez a la transacción que estaba detenida.
16. Aunque intente realizar un COMMIT, la transacción ya ha sido abortada, por lo que el sistema responderá con ROLLBACK.

```
pedidos=!# COMMIT;
```

17. Obtenga una imagen completa de la pantalla, mostrando el mensaje de error, y explique por qué se presentó el interbloqueo.
18. Regrese a la sesión de rosita y finalice su transacción:

```
pedidos=*> COMMIT;
pedidos=> SELECT * FROM cliente;
```

19. Regrese a la conexión del usuario ***postgres***, e inicie una transacción con una consulta:

```
pedidos=# BEGIN;
pedidos=*# SELECT Nombre FROM cliente WHERE Clave =15;
```

20. Pase a la sesión de ***rosita*** y ejecute la siguiente modificación:

```
pedidos=> UPDATE cliente SET Nombre='Pedro' WHERE Clave=15;
```

21. De regreso al usuario ***postgres***, repita la consulta:

```
pedidos=*# SELECT Nombre FROM cliente WHERE Clave =15;
```

22. Si estando dentro de la misma transacción, se obtiene un valor diferente para la misma consulta en dos ocasiones distintas, se presenta un fenómeno llamado ***lectura difusa*** (NO REPETIBLE READ), si la transacción estuviera

suficientemente aislada (REPEATABLE READ) no podría ocurrir.

23. Obtenga una imagen completa de la pantalla, mostrando el resultado de ambas consultas y consérvela como evidencia.
24. Finalice la transacción.

```
pedidos=*# COMMIT;
```

25. Inicie una nueva transición con una consulta:

```
pedidos=# BEGIN;
pedidos=*# SELECT Nombre FROM cliente WHERE Ciudad= 'Zacapu';
```

26. Obtendrá tres nombres en la respuesta.
27. Pase a la sesión de ***rosita*** y ejecute la siguiente modificación:

```
pedidos=> UPDATE cliente SET Ciudad='Zacapu' WHERE Clave=18;
```

28. Regrese a la conexión de ***postgres*** y repita la consulta.

```
pedidos=*# SELECT Nombre FROM cliente WHERE Ciudad= 'Zacapu';
```

29. Ahora verá que se presenta un fenómeno llamado ***Fantasma***, en este caso aparece el nombre de ***Rita***.
30. Obtenga una imagen completa de la pantalla, mostrando el resultado de ambas consultas y consérvela como evidencia.
31. Estos ejemplos ilustran los posibles fenómenos que pueden presentarse debido al nivel de aislamiento que PostgreSQL tiene por default, pero que en sí mismos, no generan inconsistencias en los resultados de las transacciones.

```
pedidos=*# COMMIT;
```

32. Ahora cambie el nivel de aislamiento del usuario ***postgres*** a REPEATABLE READ.

```
pedidos=# SET DEFAULT_TRANSACTION_ISOLATION = "REPEATABLE
READ";
```

33. E inicie una nueva transacción con dos consultas similares a las anteriores:

```
pedidos=# BEGIN;
pedidos=*# SELECT Nombre FROM cliente WHERE Clave = 15;
pedidos=*# SELECT Nombre FROM cliente WHERE Ciudad= 'Morelia';
```

34. Pase a la sesión de ***rosita*** y ejecute las siguientes modificaciones:

```
pedidos=> UPDATE cliente SET Nombre='Pablo' WHERE Clave=15;
pedidos=> UPDATE cliente SET Ciudad='Morelia' WHERE Clave=18;
```

35. Regrese a la conexión de ***postgres*** y repita ambas consultas:

```
pedidos=*# SELECT Nombre FROM cliente WHERE Clave = 15;
pedidos=*# SELECT Nombre FROM cliente WHERE Ciudad= 'Morelia';
```

36. Ahora verá que los resultados de las consultas son idénticos, ya que el nivel de

asilamiento es mayor, y los cambios realizados por fuera no son visibles dentro de la transacción en curso.

37. Haga ***COMMIT*** y repita ambas consultas:

```
pedidos=*# COMMIT;
pedidos=# SELECT Nombre FROM cliente WHERE Clave = 15;
pedidos=# SELECT Nombre FROM cliente WHERE Ciudad= 'Morelia';
```

38. Ahora si se reflejan ambos cambios.
39. Obtenga una imagen completa de la pantalla, mostrando el resultado de estas consultas y consérvela como evidencia.
40. Cierre ambas sesiones.
41. 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 transacciones con la herramienta ***psql***, viendo el manejo de la concurrencia.

Bibliografía

- https://www.postgresql.org/docs/current/tutorial-transactions.html