PRÁCTICA 06.

1. ¿Qué es una política de mantenimiento de llaves foráneas?

En PostgreSQL, una política de mantenimiento de llaves foráneas se refiere a las reglas que dictan cómo manejar la relación entre las tablas cuando hay una llave foránea. Una llave foránea es una restricción que asegura la integridad referencial entre dos tablas, garantizando que los valores en una columna (o conjunto de columnas) de una tabla deben coincidir con los valores en la llave primaria de otra tabla. Las políticas de mantenimiento especifican qué sucede con las filas en la tabla referenciada (padre) cuando se actualiza o elimina un registro que está referenciado por otra tabla (hija).

2. Para cada política que investigaron, ¿como se indica en SQL?

Existen tres políticas principales de mantenimiento de llaves foráneas en PostgreSQL:

- **ON DELETE/ON UPDATE CASCADE:** Elimina o actualiza automáticamente los registros relacionados cuando se elimina o actualiza la fila referenciada.

```
FOREIGN KEY (columna) REFERENCES tabla_referenciada(columna) ON DELETE CASCADE ON UPDATE CASCADE
```

- **ON DELETE/ON UPDATE SET NULL:** Establece el valor de la llave foránea como NULL cuando la fila referenciada se elimina o actualiza.

```
FOREIGN KEY (columna) REFERENCES tabla_referenciada(columna) ON DELETE SET NULL ON UPDATE SET NULL
```

- **ON DELETE/ON UPDATE RESTRICT/NO ACTION: I**mpide que se elimine o actualice la fila referenciada si existen filas dependientes en la tabla hija.

```
FOREIGN KEY (columna) REFERENCES tabla_referenciada(columna) ON DELETE RESTRICT ON UPDATE NO ACTION
```

- **SET DEFAULT**: Al eliminar o actualizar un registro referenciado, la columna de la clave foránea se establece en un valor por defecto.

3. Para cada política que investigaron, ¿cuál es su objeto y su funcionamiento?

- **CASCADE:**

Objeto: Mantener la consistencia automática entre tablas relacionadas.

Funcionamiento: Cuando se elimina o actualiza una fila en la tabla padre, todas las filas correspondientes en la tabla hija se eliminan o actualizan automáticamente. Esto es útil cuando las tablas tienen una relación de dependencia fuerte, y eliminar o actualizar un registro principal debe reflejarse en todas las tablas que lo referencian.

- **SET NULL**

Objeto: Evitar la eliminación de filas dependientes, permitiendo que la tabla hija conserve un valor nulo en lugar de la llave foránea.

Funcionamiento: Si se elimina o actualiza una fila en la tabla padre, las filas dependientes en la tabla hija quedan con un valor `NULL` en la columna de la clave foránea, indicando que ya no tienen una referencia válida al registro eliminado o actualizado.

- **RESTRICT/NO ACTION**

Objeto: Preservar la integridad referencial.

Funcionamiento: Estas políticas evitan que se elimine o actualice una fila en la tabla padre si hay filas dependientes en la tabla hija que la referencian. **NO ACTION** es el comportamiento predeterminado en PostgreSQL, y permite realizar la acción solo si no hay dependencias activas. **RESTRICT** es similar, pero detiene de inmediato la acción si se detectan dependencias.

- **SET DEFAULT**

Objeto: Establecer un valor por defecto en lugar de eliminar o actualizar las filas dependientes.

Funcionamiento: Si se elimina o actualiza un registro en la tabla padre, la clave foránea en la tabla dependiente se actualiza con un valor predeterminado especificado. Esto es útil cuando se desea evitar referencias nulas, pero se necesita un valor de sustitución en la clave foránea.

4. Para cada política que investigaron, ¿cuáles son sus ventajas y desventajas

- **CASCADE:**

Ventajas: Ahorra tiempo y esfuerzo al mantener automáticamente la consistencia entre tablas relacionadas sin necesidad de eliminar o actualizar manualmente las filas dependientes.

Desventajas: Puede ser peligrosa si no se usa con precaución, ya que puede eliminar o actualizar en cascada muchas filas sin un control adecuado.

- **SET NULL:**

Ventajas: Previene la eliminación de filas, pero permite que las filas dependientes sobrevivan con un valor NULL, lo que puede ser útil en ciertos escenarios.

Desventajas: Puede generar inconsistencias si los valores nulos en la llave foránea no se gestionan adecuadamente. También puede requerir lógica adicional para manejar estos valores NULL.

- **RESTRICT/NO ACTION:**

Ventajas: Garantiza que no se realicen eliminaciones o actualizaciones accidentales en la tabla padre si hay dependencias en la tabla hija, lo que protege la integridad referencial.

Desventajas: Puede ser restrictiva y hacer que las operaciones sean más complejas, especialmente en sistemas donde se necesitan actualizaciones o eliminaciones frecuentes de datos.

- **SET DEFAULT**

Ventajas: Similar a SET NULL, pero en lugar de establecer el valor en NULL, se utiliza un valor predeterminado, lo que puede ayudar a mantener la consistencia de los datos.

Desventajas: Puede introducir datos incorrectos o irrelevantes si el valor por defecto no es adecuado.

5. **Con base a lo anterior, ¿cuál política utilizarían para su esquema, y por qué motivo?**

La elección de la política depende del tipo de relación entre las tablas y el nivel de control que se quiera tener sobre las eliminaciones y actualizaciones. En general:

- Si la eliminación automática de filas dependientes no es problemática y se prefiere simplificar la lógica, ON DELETE/UPDATE CASCADE sería la opción, ya que automatiza las modificaciones de las tablas relacionadas.

- Si es crucial preservar las filas en la tabla hija, pero permitir que queden sin referencia, entonces ON DELETE/UPDATE SET NULL puede ser una buena opción.
- Si se necesita evitar que se eliminen o actualicen filas si tienen dependencias, ON DELETE/UPDATE RESTRICT o NO ACTION sería la política recomendada para garantizar una alta integridad referencial.

En nuestro caso, elegiríamos ON DELETE RESTRICT para proteger la integridad de los datos y evitar la eliminación accidental de registros padres que podrían afectar datos dependientes, manteniendo así un mayor control sobre las relaciones entre las tablas.

REFERENCIAS BIBLIOGRÁFICAS

- PostgreSQL Global Development Group. (2023). PostgreSQL Documentation: Foreign Keys. PostgreSQL. https://www.postgresql.org/docs/current/ddl-constraints.html
- Elmasri, R., & Navathe, S. (2016). Fundamentals of Database Systems (7th ed.). Pearson.
- Oppel, A. J. (2009). Databases Demystified (2nd ed.). McGraw-Hill Education.
- Celko, J. (2014). Joe Celko's SQL for Smarties: Advanced SQL Programming (5th ed.). Morgan Kaufmann.
- Ullman, J. D., & Widom, J. (2008). A First Course in Database Systems (3rd ed.). Pearson.