Investigación sobre Sentencias SQL en PostgreSQL

Hernandez Ramirez Miguel Angel

TAREA 14

1 SELECT

La sentencia 'SELECT' en PostgreSQL se utiliza principalmente para recuperar datos de una o más tablas en la base de datos. Su estructura básica es:

```
SELECT [columnas]
FROM [nombreTabla]
[WHERE condición]
[ORDER BY columna]
[LIMIT número];
```

1.1 Casos de uso

Algunos casos de uso comunes de la sentencia 'SELECT' incluyen:

- Recuperación de todos los datos de una tabla:

  ```
  SELECT * FROM empleados;
  ```

- Recuperación de datos de columnas específicas:

  ```
  SELECT nombre, salario FROM empleados;
  ```

- Uso de funciones de agregación para obtener estadísticas:

  ```
  SELECT COUNT(*), AVG(salario) FROM empleados;
  ```

1.2 Restricciones

Las restricciones de la sentencia 'SELECT' incluyen:

- Respeto por la integridad de los datos.
- Permisos de acceso: el usuario debe tener los permisos adecuados para acceder a los datos.

2 FROM

La cláusula 'FROM' en PostgreSQL se utiliza para especificar la tabla o tablas de las que se van a recuperar datos en una consulta 'SELECT'. Su sintaxis básica es:

```
FROM [nombreTabla]
```

2.1 Casos de uso

El principal caso de uso de la cláusula 'FROM' es especificar la tabla o tablas de las que se desean recuperar los datos.

2.2 Restricciones

Las restricciones de la cláusula 'FROM' incluyen:

- La tabla o tablas especificadas deben existir en la base de datos.
- La cláusula debe estar correctamente colocada después de 'SELECT'.

3 JOIN

La sentencia 'JOIN' en PostgreSQL se utiliza para combinar filas de dos o más tablas basadas en una condición de relación entre ellas. Su sintaxis básica es:

```
FROM tabla1
JOIN tabla2 ON tabla1.columna = tabla2.columna;
```

3.1 Casos de uso

Los diferentes tipos de 'JOIN' incluyen:

- 'INNER JOIN': Combina filas de ambas tablas donde la condición se cumple en ambas.

  ```
  SELECT empleados.nombre, departamentos.nombre
  FROM empleados
  INNER JOIN departamentos ON empleados.id_departamento = departamentos.id;
  ```

- 'LEFT JOIN': Devuelve todas las filas de la tabla izquierda y las coincidencias de la derecha.

  ```
  SELECT empleados.nombre, departamentos.nombre
  FROM empleados
  LEFT JOIN departamentos ON empleados.id_departamento = departamentos.id;
  ```

- ‘RIGHT JOIN‘: Devuelve todas las filas de la tabla derecha y las coincidencias de la izquierda.

```
SELECT empleados.nombre, departamentos.nombre
FROM empleados
RIGHT JOIN departamentos ON empleados.id_departamento = departamentos.id;
```

- ‘FULL JOIN‘: Devuelve todas las filas de ambas tablas, incluyendo las que no coinciden.

```
SELECT empleados.nombre, departamentos.nombre
FROM empleados
FULL JOIN departamentos ON empleados.id_departamento = departamentos.id;
```

3.2 Restricciones

Las restricciones de la sentencia ‘JOIN‘ incluyen:

- Los tipos de datos de las columnas utilizadas en la condición de unión deben coincidir.
- Los ‘JOIN‘ en tablas grandes pueden afectar el rendimiento de la consulta.

4 WHERE

La cláusula ‘WHERE‘ en PostgreSQL se utiliza para filtrar filas basadas en una condición específica. Su sintaxis es:

```
SELECT columnas
FROM nombreTabla
WHERE condición;
```

4.1 Casos de uso

Los casos de uso de la cláusula ‘WHERE‘ incluyen:

- Filtrar filas según valores específicos:

```
SELECT * FROM empleados
WHERE departamento = 'Ventas';
```

- Filtrar filas según rangos de valores:

```
SELECT * FROM productos
WHERE precio BETWEEN 100 AND 200;
```

- Filtrar filas según patrones de texto:

```
SELECT * FROM clientes
WHERE nombre LIKE 'M%';
```

4.2 Restricciones

Las restricciones de la cláusula 'WHERE' incluyen:

- Filtrar demasiadas filas puede afectar el rendimiento.

5 HAVING

La cláusula 'HAVING' en PostgreSQL se utiliza para filtrar resultados después de agruparlos con 'GROUP BY'. Su sintaxis es:

```
SELECT columna1, función(columna2)
FROM tabla
GROUP BY columna1
HAVING condición;
```

5.1 Casos de uso

Los casos de uso de la cláusula 'HAVING' incluyen:

- Filtrar grupos de filas según una función de agregación:

```
SELECT departamento, SUM(ventas) AS total_ventas
FROM registros_ventas
GROUP BY departamento
HAVING SUM(ventas) > 10000;
```

- Aplicar condiciones a múltiples funciones de agregación:

```
SELECT departamento, AVG(ventas) AS promedio_ventas
FROM registros_ventas
GROUP BY departamento
HAVING AVG(ventas) > 1500;
```

5.2 Restricciones

Las restricciones de la cláusula 'HAVING' incluyen:

- Debe usarse en conjunto con 'GROUP BY'.
- Se procesa después de 'GROUP BY'.

6 Subconsultas correlacionadas

Las subconsultas correlacionadas en PostgreSQL dependen de la consulta externa y se ejecutan una vez por cada fila devuelta por la consulta externa. Su sintaxis básica es:

```
SELECT columna1, columna2, ...
FROM tabla1
WHERE columna IN (
    SELECT columna FROM tabla2 WHERE condición
);
```

6.1 Casos de uso

Los casos de uso de las subconsultas correlacionadas incluyen:

- Filtrar resultados basados en una consulta externa:

```
SELECT nombre
FROM empleados
WHERE salario > (
    SELECT AVG(salario) FROM empleados
);
```

6.2 Restricciones

Las restricciones de las subconsultas correlacionadas incluyen:

- Pueden ser difíciles de entender debido a su estructura anidada.
- El rendimiento puede verse afectado si las tablas no están correctamente indexadas.

7 REFERENCIAS

Segovia, J., Segovia, J. (2019, 26 agosto). Sentencias fundamentales de SQL - TodoPostgreSQL. TodoPostgreSQL - Academia Online de PostgreSQL en Español. https://www.todopostgresql.com/sentencias-fundamentales-de-sql/

Ejemplo: Crear vistas de base de datos en PostgreSQL con SQL—ArcMap — Documentación. (s. f.). https://desktop.arcgis.com/es/arcmap/latest/manage-data/using-sql-with-gdbs/example-creating-a-view-in-postgresql-with-sql.htm