Investigar para las subconsultas (casos de uso, restricciones, ejemplos):

- SELECT:

  La instrucción SELECT es básica para realizar consultas y subconsultas. Esta permite obtener cierta información de una o un conjunto de consultas. Algunos casos de uso son:
  - Recuperar datos de una tabla (esta tabla puede ser de la base, o una tabla resultante de subconsultas). Se pueden recuperar todos los datos o datos específicos de la tabla sobre la que se está realizando la instrucción.
  - Es de gran utilidad para obtener información de tablas que no están explícitamente almacenadas en la base de datos. Esto puede llevar a la realización de varias subconsultas que permitan obtener una tabla temporal, la cual puede ser utilizada en otros momentos.

  Algunas restricciones que se deben de tomar en cuenta son:
  - Los permisos del usuario para obtener información de tablas.
  - Tener en cuenta qué información se necesita para no realizar consultas innecesarias.
  - No realizar demasiadas consultas, ya que puede generar un empobrecimiento del rendimiento de la base.
  - Tener en cuenta los valores que se retornan en la subconsulta es de suma importancia. Si se realiza una consulta sobre la subconsulta que retornó un valor inadecuado, puede generar errores.

  Ejemplos:

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

  Se puede observar que se solicita aquellos empleados cuyo salario está por encima del promedio. Esto fue posible con subconsultas.

  ```
  SELECT COUNT(*) FROM trabajadores WHERE id_trabajador IN (SELECT * FROM proyecto WHERE terminado = 'false');
  ```

  Se cuenta a todos los trabajadores que pertenecen a un proyecto activo. Así como se realizaron estas consultas, se puede concatenar más subconsultas; sin embargo, como se mencionó, esto no llega a ser óptimo.

- FROM:

  La instrucción FROM sirve para señalizar el uso cierta tabla para realizar una consulta. Igualmente, es de utilidad para utilizar información de tablas en subconsultas. Algunos casos de uso son:
  - Especifica ALIAS en consultas y subconsultas.

- Dentro del FROM se puede realizar una subconsulta de una tabla virtual. Esto va relacionado con lo que se mencionó en la sección de SELECT, sin embargo, al agregar la subconsulta dentro del FROM, es posible realizar una correlación de la información. Esto quiere decir que puede haber una relación entre la información de la consulta principal y la subconsulta (por ejemplo, si se define un alias de una tabla, es posible utilizar dicha tabla dentro de la subconsulta). Esto es de gran utilidad para poder relacionar la información de tablas y generar consultas más complejas.

Algunas restricciones que se deben de tomar en cuenta:

- Se debe de tener cuidado de mantener un número de columnas en la subconsulta que concuerde con la información de la consulta principal.
- Si se realizan correlaciones entre la consulta principal y la subconsulta, es de suma importancia que haya coherencia lógica para evitar errores.
- El exceso de complejidad de la subconsulta puede generar problemas de rendimiento y legibilidad.

Ejemplo:

```
SELECT departamento_id, AVG(salario) AS salario_promedio
FROM (
        SELECT departamento_id, salario
        FROM empleados
        WHEERE salario > 5000
) AS subconsulta GROUP BY departamento_id;
```

En este caso, utilizando la subconsulta dentro del FROM, es posible obtener una tabla temporal que contenga la información de todos los departamentos con los salarios de los empleados mayor a 50000. Esto se utiliza para calcular el promedio de salarios de cada departamento. Se observa que se utiliza la instrucción GROUP BY, el cual permite agrupar los salarios de cada uno de los departamentos.

- JOIN:

Los JOINS permiten obtener tablas que relacionen información a través de ciertas condiciones. Al igual que la cláusula FROM, los JOIN's pueden utilizar subconsultas para realizar combinación de tablas a través de ciertas consultas complejas. Este tipo de subconsultas permite realizar JOINS más complejos entra tablas que deban de cumplir ciertas condiciones. Al igual que la cláusula FROM:

- Se debe de tomar en cuenta la cantidad de columnas e información que resultará de la subconsulta dada en el JOIN para evitar errores lógicos con la consulta principal.
- Igualmente, si existe una correlación entre columnas, se debe de verificar que haya una relación válida entre tablas.
- Puede llegar a comprometer el rendimiento de la consulta, así como complicar la legibilidad del código.

Ejemplo:

```
SELECT e. nombre, e.salario FROM empleados AS e
JOIN (
    SELECT DISTINCT departamento_id
    FROM proyectos
    WHERE tipo = 'Desarrollo'
) AS p ON e.departamento_id = p.departamento_id;
```

Se observa que, mediante la subconsulta del JOIN es posible facilitar la obtención de la información de todos aquellos empleados que se encuentran trabajando de un proyecto de desarrollo.

- WHERE:

Con la instrucción WHERE es posible realizar álgebra relacional que permita obtener información bajo condiciones específicas. Utilizar subconsultas dentro de una instrucción WHERE puede ser de gran utilidad para obtener información dada ciertas condiciones particulares. Algunos casos de uso son:

- Obtener información asociada a un valor para realizar una comparación particular.

Por ejemplo, obtener un promedio de cierta tabla utilizándolo como criterio en la consulta principal:

```
SELECT nombre, promedio FROM alumnos
WHERE promedio > (SELECT AVG(promedio) FROM alumnos);
```

- Igualmente, si se utiliza la instrucción IN concatenada a la instrucción WHERE, es posible obtener cierta información para datos que se encuentren en la subconsulta realizada.
- Igualmente, es posible realizar una correlación en la subconsulta para tener una relación con la consulta principal. En este caso, también es importante tener en cuenta la lógica y la relación de la subconsulta con la consulta principal, especialmente cuando se realiza una correlación.

Ejemplo:

```
SELECT nombre, departamento_id FROM empleados AS e
WHERE EXISTS (
    SELECT 1 FROM proyectos AS p WHERE p.responsable_id = e. empleado_id
);
```

- HAVING:

La instrucción HAVING se utiliza con la instrucción GROUP BY, para señalizar condiciones que se deben de cumplir en un grupo de filas. Al igual que con WHERE, esto es de utilizar para

filtrar bajo ciertas condiciones. Es muy poderoso utilizar subconsultas para obtener resultados más específicos dentro de las consultas principales.

Ejemplo:

```
SELECT departamento_id, AVG(salario) AS salario_promedio FROM empleados
GROUP BY departamento_id HAVING AVG(salario) > (SELECT AVG (salario)
FROM empleados);
```

En este caso, la subconsulta obtiene el salario promedio de todos los empleados. Posteriormente, en la consulta principal, se está obteniendo la información de los salarios promedio por departamento, y, mediante la instrucción HAVING, se toma aquellos departamentos cuyo salario promedio sea mayor que el promedio de todos los empleados.

- CORRELACIONADAS:

Como se observó en los casos anteriores, las subconsultas correlacionadas permiten tener una relación lógica entre tablas de la consulta principal con tablas de la subconsulta para generar condiciones más complejas. Por lo tanto, son de gran utilidad cuando se necesita referenciar una columna de la consulta principal para cierta subconsulta.

Ejemplo:

```
SELECT cliente_id,
    (SELECT COUNT(*)
    FROM ordenes
    WHERE cliente_id = c.cliente_id AND fecha < '2023-01-01') AS
    num_ordenes_antes_2023
FROM clientes c;
```

Referencias

[1] “SQL joins”, W3schools.com. [En línea]. Disponible en: https://www.w3schools.com/sql/sql_join.asp. [Consultado: 09-nov-2023].

[2] “PostgreSQL 15.5 documentation”, PostgreSQL Documentation, 09-nov-2023. [En línea]. Disponible en: https://www.postgresql.org/docs/15/index.html. [Consultado: 10-nov-2023].