Profesor
M. en I.A. Erick Orlando Matla Cruz
Ayudantes
L. en C.C. Efraín Hipólito Chamú
L. en C.C. Anahí Quiroz Jiménez
L. en C.C. Karen Monserrat Zavala Correa
Grupo: 9119

Práctica 10

Vistas, Índices y CTE's

1. **Objetivo General**

   Conocer el manejo de creación y uso de vistas, manipulación de índices para la optimización de consultas, así como el correcto uso de CTE's.

2. **Objetivos Secundarios**

   - Entender el uso de vistas y sus ventajas.
   - Entender la importancia del uso de índices.
   - Entender las ventajas de usar CTE y su uso.

3. **Introducción**

Al explotar el contenido de una base de datos podemos tener sentencias muy elaboradas, así como mucha información, que si no tenemos un buen diseño, estructura, optimizaciones, etc. podemos llegar a sobrecargar nuestras bases y por lo tanto no tener un buen uso de la misma.

La práctica presenta herramientas útiles que el propio SMBD aporta para su uso y que nos ayuda a mantener un mejor control de nuestras bases de datos.

3.1 Vistas.

Una vista se puede definir como una "tabla virtual" ya que se comporta como una tabla al realizar consultas sobre ella pero, a diferencia de una tabla, no siempre es posible realizar modificaciones sobre los datos de ella.

Dicho de otra manera, la información que conforma una vista es extraída de las tablas existentes en la base de datos, se puede decir que es información virtual, ya que es una vista de los datos reales de la base de datos.

Las vistas también pueden ser utilizadas para simplificar consultas complejas, al crear "tablas virtuales" sobre las cuales se realizarán consultas más simples.

I. Creación de Vistas.

La creación de Vistas presenta la siguiente sintaxis.

Sintaxis:

Profesor
M. en I.A. Erick Orlando Matla Cruz
Ayudantes
L. en C.C. Efraín Hipólito Chamú
L. en C.C. Anahí Quiroz Jiménez
L. en C.C. Karen Monserrat Zavala Correa
Grupo: 9119

```
CREATE OR REPLACE VIEW nombredevista
(column_list_virtual) AS consulta;
```

Argumentos:

- CREATE OR REPLACE
  Palabra reservada en PostgreSQL para la creación de Vistas. CREATE OR REPLACE indica que se desea crear o reemplazar, en caso de que exista previamente, la definición de una vista anterior por la actual. OR REPLACE se puede omitir si no se desea sobrescribir una vista que ya existe.
- VIEW
  Palabra reservada en PostgreSQL que indica que el objeto que se creará o reemplazará es una vista.
- nombredevista
  Nombre que se le da a la vista virtual.
- (column_list_virtual)
  Lista de columnas virtuales en la vista, se tiene la posibilidad de poner una o más columnas como argumentos separados por coma, si no se indica, las columnas virtuales serán las que se indiquen en la consulta.
- AS
  Palabra reservada en PostgreSQL, que permite dar un nombre a la vista creada, mediante este nombre la vista puede ser utilizada posteriormente.
- consulta
  Esta consulta es una sentencia, o también llamado query, que permite la extracción de la información en la base de datos, generalmente usada con un SELECT.

Uso:

El uso de la creación de una vista se presenta a continuación, ver Figura 10.1.

Continuando con el ejemplo anterior de la práctica 9 de Productos, se presenta la vista del total de los productos como ejemplo, en donde se utilizó la tabla Productos con nombre_product, category y total_productos como columnas. Generando así la vista Total_Productos con nombre_producto, categoria y productos_totales como columnas virtuales y restringiendo aquellos con productos mayores a 50.

```
CREATE VIEW Total_Productos (nombre_producto, categoria, productos_totales)

    AS SELECT name_product, category, total_products FROM productos where total_products> 50;
```

Figura 10.1 Creación de Vista Total_Productos.

Para consultar la Vista se hace una selección de todas las columnas en Total_Productos, ver Figura 10.2.

```
SELECT * FROM Total_Productos;
```

Figura 10.2 Consulta de Vista Total_Productos.

nombre_producto	categoria	productos_totales
Desktop	ELE	60
Laptop	ELE	70
Desktop	FG	60
Laptop	IT	70

Figura 10.3 Resultado de consulta de Total_Productos.

II. Manejo de Vistas.

El modificar las vistas permite reemplazar una consulta ya creada por una nueva. Si se desea modificar la Vista Total_Productos creada anteriormente, se deberá colocar la instrucción OR REPLACE como sigue, ver Figura 10.4.

```
CREATE OR REPLACE VIEW Total_Productos (nombre_producto, categoria, productos_totales, tipo)

    AS SELECT name_product, category, total_products, tipo FROM productos where total_products> 50;
```

Figura 10.4 Reemplazo de Vista Total_Productos.

En este caso se agregó la columna tipo quedando el resultado como sigue, ver Figura 10.5.

nombre_producto	categoria	productos_totales	tipo
Desktop	ELE	60	B
Laptop	ELE	70	B
Desktop	FG	60	B
Laptop	IT	70	B

Figura 10.5 Resultado de Vista Total_Productos

Para eliminar las vistas existentes, se deberá utilizar la instrucción DROP de la siguiente manera:

DROP VIEW Total_Productos;

A partir de la versión 9.3 de PostgreSQL las vistas simples pueden ser actualizadas, permitiendo insertar, actualizar y borrar, esto solo si cumple las siguientes condiciones:

- La vista debe tener exactamente una entrada en su lista de FROM, que debe ser una tabla u otra vista actualizable.
- La definición de vista no debe contener COUNT, DISTINCT, GROUP BY, HAVING o LIMIT.
- La definición de la vista no debe contener operaciones de conjuntos (UNION, INTERSECT o EXCEPT) en el nivel superior.
- Todas las columnas de lista de selección de la vista deben ser simples referencias a columnas de la relación subyacente. No pueden ser expresiones, literales o funciones. Las columnas no pueden ser referenciadas.
- La columna de la relación subyacente no puede aparecer más de una vez en la lista de la selección de la vista.
- La vista no debe tener la prioridad *security_barrier*.

Para mayor información se puede consultar la documentación oficial de PostgreSQL en la siguiente liga:

https://www.postgresql.org/docs/11/sql-createview.html

3.2 Índices

El índice de un texto sirve para encontrar de manera rápida un cierto elemento dentro de la totalidad de un contenido. Los índices en las bases de datos tienen una función similar, es decir, mediante el uso de índices se pueden acelerar los tiempos de respuesta de las consultas. Los índices por lo general se construyen para la o las columnas de una cierta tabla que se consulta con cierta frecuencia.

La sintaxis para la creación de índices, Figura 10.6.

```
CREATE INDEX Nombre_indice ON Nombre_Tabla (Nombre_columna);
```

Figura 10.6. Sintaxis utilizada en la creación de índices.

En esta sintaxis encontramos lo siguiente:

1. CREATE INDEX
   Palabras reservadas para comenzar con la instrucción de creado de índices.
2. Nombre_indice
   Es el nombre con el que se define el índice que se está creando.
3. ON
   Palabra reservada para declarar la tabla sobre la cual será creado el índice.
4. Nombre_Tabla
   Es el nombre de la tabla objetivo del índice.
5. (Nombre_columna)
   Es el nombre de la columna sobre la cual se hará el indexado.

A manera de ejemplo la figura 10.7 muestra la sintaxis para crear el índice que acelere las consultas sobre la tabla Productos para la columna *name_product*.

```
CREATE INDEX idx_name_product ON productos(name_product);
```

Figura 10.7. Sintaxis de ejemplo de índice para la tabla Productos.

3.3 CTE's

PostgreSQL proporciona una forma de escribir sentencias auxiliares y temporales para su uso en una consulta más grande. Estas declaraciones definidas como Common table expressions (CTE) o expresiones de tabla comunes, se pueden considerar como la definición de tablas temporales que existen solo para una consulta.

La sintaxis para la creación de un CTE, se describe a continuación Figura 10.8.

```
WITH cte_name (column_list) AS (
   CTE_query_definition
)
query_principal;
```

Figura 10.8. Sintaxis utilizada para la creación de un CTE.

En esta sintaxis encontramos lo siguiente:

1. WITH
   Palabra reservada para comenzar con la instrucción del CTE.
2. cte_name
   Es el nombre con el que se define el CTE que se está creando.

3. (column_list)
   Lista de columnas a mostrar del CTE (opcional), si no se indica, las define el cuerpo del CTE (CTE_query_definition).
4. CTE_query_definition
   Instrucción auxiliar del CTE indicado por sentencias.
5. query_principal
   Declaración principal indicada por sentencias.

Cada instrucción auxiliar (CTE_query_definition) y declaración principal puede ser de tipo SELECT, INSERT, UPDATE o DELETE.

A continuación se presentan ejemplos de tipo SELECT y DELETE.

El primer CTE usa una consulta temporal de productos clasificados dependiendo la categoría, mostrando aquellos que contienen productos mayores a 30, Figura 10.9.

```
WITH cte_producto_clase AS (
    SELECT
        tipo,
        name_product as nombre_producto,
        (CASE
            WHEN category = 'IT' THEN 'Alta'
            WHEN category = 'ELE' THEN 'Media'
            ELSE 'Otra'
        END) clase,
        category,
        total_products
    FROM
        productos
)
SELECT
    tipo,
    nombre_producto,
    clase,
    category,
    total_products
FROM
    cte_producto_clase
WHERE
    total_products > 30
ORDER BY
    total_products;
```

Figura 10.9. CTE cte_producto_clase de tipo SELECT.

tipo	nombre_producto	clase	category	total_products
B	Mobile	Media	ELE	40
A	Desktop	Alta	IT	50
B	Desktop	Media	ELE	60
B	Desktop	Otra	FG	60
B	Laptop	Media	ELE	70
B	Laptop	Alta	IT	70

Figura 10.10.Resultados del CTE cte_producto_clase.

El siguiente CTE de tipo DELETE borra todos los registros que contengan productos con un total de 40, devolviendo los resultados con la instrucción RETURNING de todos los campos (*) e inserta los resultados en una tabla previamente creada, Figura 10.11.

```
WITH moved_rows AS (
   DELETE FROM productos
   WHERE
      total_products=40
   RETURNING *
)
INSERT INTO products_log
SELECT * FROM moved_rows;
```

Figura 10.11. CTE moved_rows.

id_product	tipo	name_product	category	total_products
2	B	Mobile	ELE	40

Figura 10.12. Resultados del CTE moved_rows.

1. Ejercicios

(NOTA: Resuelve los siguientes ejercicios en relación al proyecto que realizarás durante el curso, en dado caso que no tengas un proyecto, utiliza la información en el apéndice Seguros parte 08 al final de esta práctica para realizarlos)

1. Realiza 3 vistas que incluyan al menos 3 tablas, usando operadores de comparación, funciones de agregación, agrupación y JOIN.
2. Crea índices a los catálogos del proyecto y a las tablas que consideren necesarias.
3. Realiza 2 CTE de tipo SELECT.
4. Realiza 2 CTE de tipo DELETE.

Entregables requeridos para prácticas subsecuentes:

- Creación de Vistas, Índices y CTE’s.