Base de Datos I TPE - Grupo 4

1Q2021

Fecha de Entrega: 10/06/21

Integrantes:

- Julián Francisco, **Arce Doncella** 60509
- Roberto José, **Catalán** 59174
- Gian Luca, **Pecile** 59235

Introducción

A lo largo del trabajo práctico especial se busca ampliar los conocimientos de los temas vistos en el primer parcial, poner en práctica y poder plasmar los conocimientos avanzados sobre SQL dados en la última parte de la materia. A modo de preparación para esto se realizó investigación previa en tanto la documentación de postgreSQL como las clases teóricas y prácticas que se encuentran accesibles gracias a la virtualidad, al igual que para las clases se hizo uso de la documentación aportada por la cátedra a fin de un mayor entendimiento de los temas a abarcar.

En cuanto a la división de tareas para lograr el objetivo planteado se asignaron los siguientes roles:

- Julián Arce: Encargado de funciones.
- Roberto Catalán: Encargado de triggers.
- Gian Luca Pecile: Encargado del informe.

Otros roles tales como el correcto funcionamiento global del proyecto, la investigación necesaria, entre otros fueron rotados a lo largo del desarrollo del proyecto donde se hizo uso de un repositorio compartido en github para mejor manejo de tareas en equipo.

Desarrollo

En principio se eligió para la tabla intermedia como llaves primarias *Quarter, Month, Week, Sales_channel, Product_Type y Customer_Type*. Luego, para la tabla definitiva se eligieron como llaves primarias *Sales_date, Product_type, Sales_channel, Customer_type* en base a lo que serán usadas en las próximas consultas.

Al momento de realizar triggers se hizo uso de lo investigado como fue mencionado en la introducción. El trigger desarrollado se llama *fillTableTrigger* que hace uso de la función *fillTable()* y se encarga de insertar las tuplas en el formato correspondiente en la tabla *definitiva* a medida que se insertan datos a la tabla *intermedia*. Antes de la inserción a la tabla *definitiva* se transforman los datos de *Quarter*, *Month* y *Week* a la columna *Sales_date* de tipo *date*.

Para la implementación de las funciones de cálculo de la mediana de margen móvil se debió investigar tanto sobre la posibilidad de sumar meses en postgreSQL, en bibliografía se puede ver en detalle sobre el método usado y el link consultado, además se investigó sobre la validación de errores en cuanto al resultado de la mediana. Para poder calcular la mediana móvil se hizo uso de la función agregación *percentile_cont* donde se hizo uso de la documentación de postgreSQL.

En cuanto al reporte de ventas histórico, el código en un principio comparte similitudes con el inciso anterior en cuanto a su estructura, con lo cual se tomó como punto de partida dicha estructura aunque fue alterada en su versión final. La principal dificultad encontrada fue presentar los datos de manera similar al enunciado de forma eficiente, siguiendo buenas prácticas y sin generar recorridos innecesarios a la tabla. Para lograr esto se hace uso del concepto de cursores, en primera instancia se hicieron implícitos aunque luego se optó por usar explícitos para mejorar claridad del código, luego de dicho cambio se consideró e implementó el uso de *refcursors,* los cuales se leyó la documentación al respecto y cumplieron el objetivo de simplificar junto con aumentar la claridad del código. Además, siguiendo buenas prácticas de programación en cuanto a la modularización de código se utiliza una función auxiliar llamada *printInfo* que se encarga de imprimir la información necesaria para el reporte de ventas por cada una de las tuplas. Por último, se utiliza un cursor implícito para calcular los valores totales de *Revenue*, *Cost* y *Margin*.

Finalmente, el pasaje de los datos del CSV, en primera instancia se hizo uso de una funcionalidad agregada como herramienta del IDE DataGrip de Jetbrains donde permite la importación de los datos de un archivo sin la necesidad de escribir el código para realizar los copy, similar a interfaces gráficas para el manejo de git. Luego de consultarlo con la cátedra se descartó el uso de dicha funcionalidad y se optó por el uso de COPY tal como dice en el enunciado. Por lo tanto, para poder ejecutar el comando se debe copiar el archivo a pampero, y luego conectarse a pampero mediante ssh:

```
scp ./SalesbyRegion.csv
<user>@pampero.itba.edu.ar:<path_directorio_csv>
```

```
ssh <user>@pampero.itba.edu.ar
```

Luego de haber copiado el CSV y encontrarse con una conexión de pampero se debe ejecutar la siguiente línea para el uso del comando:

```
psql -h bd1.it.itba.edu.ar -U <user> PROOF
```

Por último, ejecutar el siguiente comando, que se encargará de copiar los archivos a la tabla intermedia. Primero se debe haber creado las tablas y el trigger correspondiente:

```
\copy intermedia FROM <path_al_directorio/SalesbyRegion.csv> CSV HEADER DELIMITER ','
```

También se puede usar el path relativo al archivo csv.

Bibliografía

- Documentación brindada por la cátedra.
- PostgreSQL SPLIT_PART() function.
- PostgreSQL CREATE TRIGGER Statement By Practical Examples.
- Adding months to a date in PostgreSQL shows syntax error.
- PostgreSQL: Documentation: 9.4: Aggregate Functions.
- Import | DataGrip (jetbrains.com).
- PostgreSQL: Documentation: 10: COPY.
- PostgreSQL: Documentation: 9.1: Cursors