**SQL Coding Challenge – Question Number 1**

**Name: S.Yazhini (DE130)**

**Date: 08-11-2024**

**Question:**

1.Querying Data by Using Joins and Subqueries & subtotal

**Objective:**

* The main objective of this question is to understand the concepts of joins, subqueries and subtotal.
* This helps us to write a real time SQL query base on the problem and the datasets.

**Concepts Used:**

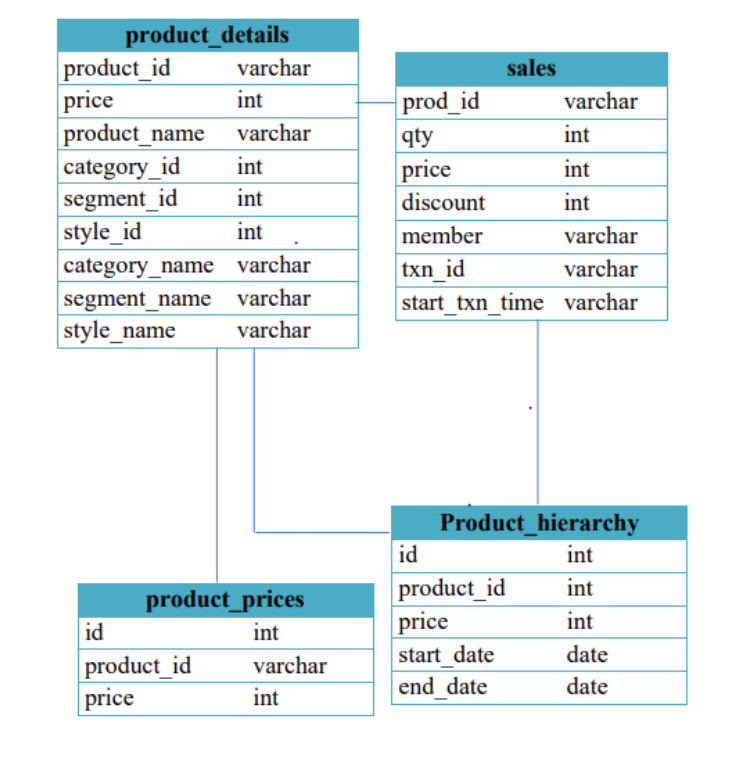
* **JOINS:** 
  + It helps to connect multiple tables.
  + It is used to retrieve data across multiple tables based on related columns.
  + **Types of joins:**

1. INNER JOIN: This Returns only the rows with matching values in both joined tables.
2. LEFT JOIN (or LEFT OUTER JOIN): This Returns all rows from the left table and matching rows from the right table; unmatched rows in the right table show as NULL.
3. RIGHT JOIN (or RIGHT OUTER JOIN): This Returns all rows from the right table and matching rows from the left table; unmatched rows in the left table show as NULL.
4. FULL JOIN (or FULL OUTER JOIN): This Returns all rows where there is a match in either table; unmatched rows from both tables are included with NULLs for missing values.
5. CROSS JOIN: This Returns the Cartesian product of both tables, creating every possible combination of rows.
6. SELF JOIN: This Joins a table to itself, often used for hierarchical or recursive data structures.

* **SUBQUERY:**
  + The query inside query is called sub-query.
  + It is used to write complex query for multiple purpose.
  + It can retrieve data to be used as input to the main query, often used for filtering or calculating aggregate values.
* **SUBTOTAL:**
  + Subtotal shows the sum of similar sets of data but it does not indicate the final total
  + Subtotals are mainly used in sales, finance.
* **DATASET USED FROM DRIVE:**
  + Texture Tales Dataset Link:

<https://drive.google.com/drive/folders/1EaO3PgoiMC-bG_-9PmwANAMRTuX-0quZ?usp=drive_link>

* **TABLES CREATED:**



* **SQL QUERY:**

Here are five sample queries that incorporate JOINs, SUBQUERIES, and subtotal calculations:

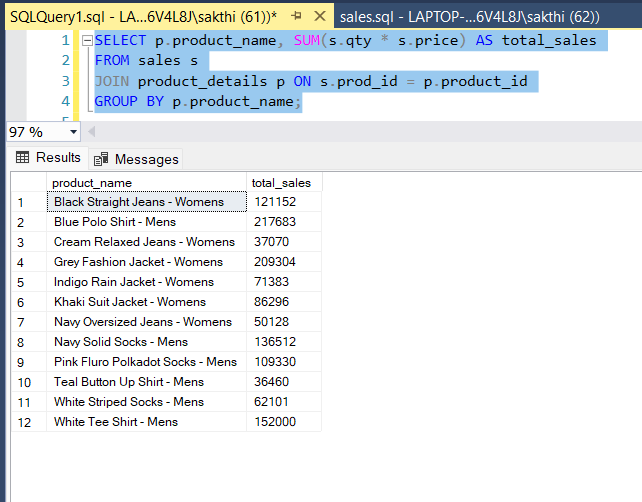
1. **Find Total Sales by Product (Using JOIN):**

SELECT p.product\_name, SUM(s.qty \* s.price) AS total\_sales

FROM sales s

JOIN product\_details p ON s.prod\_id = p.product\_id

GROUP BY p.product\_name;

Group by is used to arrange identical data into groups.

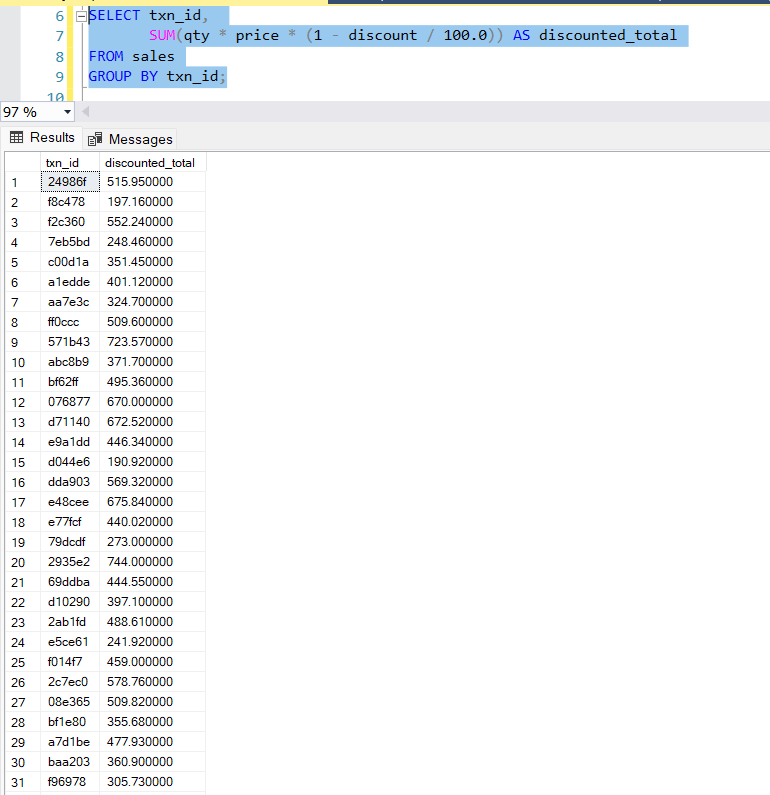
1. **Calculate Discounted Total for Each Transaction (Using Subquery)**

SELECT txn\_id,

SUM(qty \* price \* (1 - discount / 100.0)) AS discounted\_total

FROM sales

GROUP BY txn\_id;



1. **Subtotal Sales by Product Segment (Using JOIN and GROUP BY):**

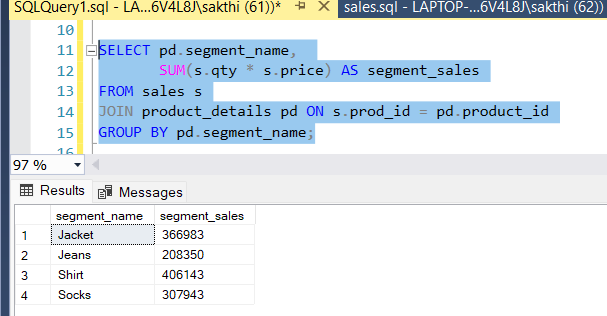
SELECT pd.segment\_name,

SUM(s.qty \* s.price) AS segment\_sales

FROM sales s

JOIN product\_details pd ON s.prod\_id = pd.product\_id

GROUP BY pd.segment\_name;



1. **Hierarchical Aggregation of Product Prices (Using JOIN and Hierarchy):**

SELECT ph.level\_name,

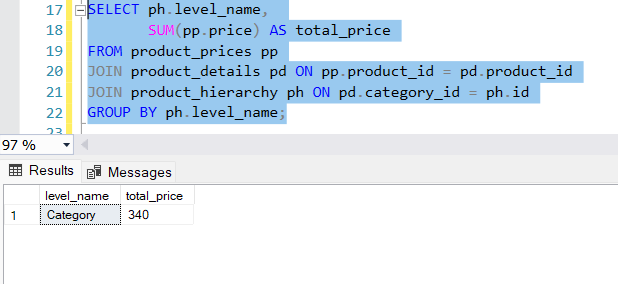
SUM(pp.price) AS total\_price

FROM product\_prices pp

JOIN product\_details pd ON pp.product\_id = pd.product\_id

JOIN product\_hierarchy ph ON pd.category\_id = ph.id

GROUP BY ph.level\_name;



1. **Comprehensive Query: Total Sales, Discounts, and Subtotals by Segment and Category (JOINs, Subqueries, and Aggregation):**

SELECT pd.segment\_name,

pd.category\_name,

SUM(s.qty \* s.price) AS total\_sales,

SUM(s.qty \* s.price \* (1 - s.discount / 100.0)) AS total\_discounted\_sales

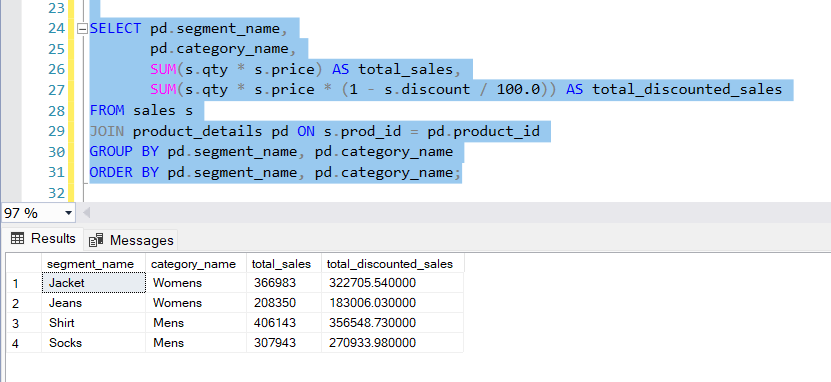
FROM sales s

JOIN product\_details pd ON s.prod\_id = pd.product\_id

GROUP BY pd.segment\_name, pd.category\_name

ORDER BY pd.segment\_name, pd.category\_name;

Order by is used to arrange the data as per given condition in ascending or descending.



NOTE: For question 2 refer another doc file in the same drive folder.

-----------------------------------------Thank you-----------------------------------------