# SQL QUERY RETAIL ANALYSIS

**Submitted By: Antara Biswas** 



# **RETAIL PRODUCTS DATASET**

Brief description of each column in the dataset:

- ➤ PRODUCT\_ID: Unique identifier for the products.
- ➤ PRODUCT\_NAME: Name of the product.
- > CATEGORY: Category to which the product belongs.
- > STOCK\_QUANTITY: Number of units of the product currently in stock.
- > SUPPLIER: Name of the supplier providing the product.
- > DISCOUNT: Percentage discount applied to the product.
- ➤ RATING: Rating of the product based on customer reviews.
- ➤ REVIEWS: Total number of customer reviews for the product.
- > SKU: Stock keeping unit code
- ➤ WAREHOUSE: Name of the warehouse where the product is stored.
- > RETURN\_POLICY: Days under which the product can be returned.
- > BRAND: Brand name of the product.
- > SUPPLIER CONTACT: Contact information for the product's supplier.
- > PRICE: Selling price of the product.

# Task 1: Identify products with prices higher than the average price within their category.

# **Explanation:**

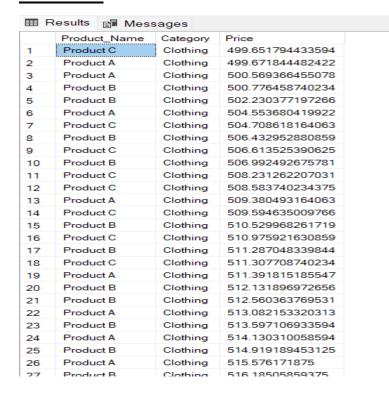
- In this SQL query, we select products from the Retail table where the price of each product is higher than the average price of products in the same category.
- The subquery (SELECT AVG(Price) FROM Retail AS R WHERE R.Category = Retail.Category) calculates the average price for each category.
- The WHERE clause compares each product's price to this average.
- The result is ordered by Category and then by Price.

# **INPUT:**

```
SELECT * FROM Retail;

-- Task 1:Products with prices higher than the average price within their category

SELECT Product_Name, Category, Price
FROM Retail
WHERE Price > (SELECT AVG(Price) FROM Retail AS R WHERE R.Category = Retail.Category)
ORDER BY Category, Price;
```



# **Task 2:** Find Categories with Highest Average Rating Across Products.

# **Explanation:**

- ➤ In this SQL query, we calculate the average rating for each product within its category and round the average rating to three decimal places.
- ➤ The GROUP BY clause groups the results by Category and Product Name.
- ➤ The ORDER BY clause sorts the results first by Category and then by average rating in descending order.

# **INPUT:**

⊞ F	Results					
	Category	Product_Name	Avg_Rating			
1	Clothing	Product B	2.996			
2	Clothing	Product C	2.965			
3	Clothing	Product A	2.963			
4	Electronics	Product C	3.015			
5	Electronics	Product A	2.996			
6	Electronics	Product B	2.92			
7	Home	Product B	3.035			
8	Home	Product C	2.992			
9	Home	Product A	2.958			

# **Task 3:** Find the most reviewed product in each warehouse.

# **Explanation:**

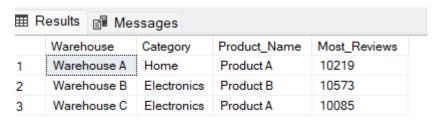
- ➤ In this SQL query,we first create a Common Table Expression (CTE) named Product to calculate the total reviews for each product and then we assign a row number based on the total number of reviews within each warehouse.
- ➤ The ROW\_NUMBER() function assigns a unique rank starting at 1 for the product with the highest reviews in each warehouse.
- ➤ Then in the main query, we select the products with the highest reviews (i.e., RowNum = 1) from each warehouse.

# **INPUT:**

```
-- Task 3: Find the most reviewed product in each warehouse

|WITH Product AS (
| SELECT Warehouse, Category, Product_Name, SUM(Reviews) as Most_Reviews,
| ROW_NUMBER() OVER (PARTITION BY Warehouse ORDER BY SUM(Reviews) DESC) AS RowNum
| FROM Retail
| GROUP BY Warehouse, Category, Product_Name
|)

| SELECT Warehouse, Category, Product_Name, Most_Reviews
| FROM Product |
| WHERE RowNum = 1; |
```



# <u>Task 4:</u> Find products that have higher-than-average prices within their category, along with their discount and supplier.

# **Explanation:**

- As we did in Task 1, this query selects products with prices higher than the average price within their category.
- > This query also retrieves the discount and supplier for each product.

# **INPUT:**

-- Task 4: Find products that have higher-than-average prices within their category, along with their discount and supplier | SELECT Product\_Name, Category, Price, Discount, Supplier | FROM Retail

WHERE Price > (SELECT AVG(Price) FROM Retail AS R WHERE R.Category = Retail.Category);

	Results	Category	Price	Discount	Supplier
1	Product C	Clothing	732.121276855469	10.5807666778564	Supplier Y
-			933.313293457031	15.32288646698	
2	Product B	Clothing			Supplier X
3	Product A	Clothing	649.901489257813	15.0801782608032	Supplier Y
4	Product A	Clothing	874.577758789063	4.01121473312378	Supplier X
5	Product B	Clothing	832.329406738281	27.4269504547119	Supplier Z
6	Product C	Clothing	875.730346679688	4.4839129447937	Supplier Y
7	Product B	Clothing	521.836242675781	34.2522583007813	Supplier Z
8	Product C	Clothing	534.276977539063	33.5766754150391	Supplier Z
9	Product C	Clothing	800.959106445313	47.6292991638184	Supplier Z
10	Product A	Clothing	805.006774902344	13.2055196762085	Supplier X
11	Product C	Clothing	722.383422851563	2.33885741233826	Supplier X
12	Product C	Clothing	800.499389648438	2.96242260932922	Supplier Y
13	Product B	Clothing	579.319702148438	39.8300247192383	Supplier X
14	Product B	Clothing	788.717651367188	2.78330731391907	Supplier Z
15	Product B	Clothing	708.302795410156	36.36767578125	Supplier Y
16	Product A	Clothing	766.609069824219	25.0464706420898	Supplier Y
17	Product A	Clothing	642.902526855469	21.6021366119385	Supplier Z
18	Product A	Clothing	999.754028320313	27.2006149291992	Supplier X
19	Product A	Clothing	727.681884765625	30.6492938995361	Supplier Z
20	Product B	Clothing	941.802551269531	29.4486293792725	Supplier X
21	Product C	Clothing	863.261535644531	33.118579864502	Supplier Y
22	Product C	Clothing	958.332946777344	17.6586723327637	Supplier X
23	Product A	Clothing	826.025329589844	41.3488426208496	Supplier Z
24	Product B	Clothing	513 597106933594	2 53226637840271	Supplier 7

# Task 5: Find the top 2 products with the highest average rating in each category.

# **Explanation:**

- ➤ In this SQL query, we create a CTE named Prod to calculate the average rating for each product within its category and then we assign a row number based on the average rating in descending order.
- ➤ The ROW\_NUMBER() function assigns a unique rank starting at 1 for the highest-rated products in each category.
- The main query then selects the top 2 products using WHERE clause (i.e., RatingRank <= 2) with the highest average rating from each category.

# **INPUT:**

⊞ F	Results 📳 N	Messages	
	Category	Product_Name	Avg_Rating
1	Clothing	Product B	2.99648716123842
2	Clothing	Product C	2.96519911683658
3	Electronics	Product C	3.01500870064042
4	Electronics	Product A	2.99595845636524
5	Home	Product B	3.03519329690848
6	Home	Product C	2.99245656077529

# **Task 6:** Analysis Across All Return Policy Categories.

# **Explanation:**

In this SQL query, we analyse the fields using GROUP BY clause which groups the results by return policy. The aggregate functions used are:

- ➤ Count(Product\_ID): Determines the number of products offered under each return policy.
- ➤ Avg(STOCK\_QUANTITY): Calculates the average stock available for products under each return policy.
- ➤ SUM(Stock\_Quantity): Sums the total stock for products under each return policy.
- ➤ SUM(RATING\*REVIEWS)/SUM(REVIEWS): Computes the weighted average rating considering the number of reviews for each return policy.
- > SUM(Reviews): Sums the total number of reviews for products under each return policy.
- ➤ Avg(Discount): Determines the average discount percentage for products under each return policy.
- Avg(Price): Calculates the average selling price for products under each return policy.
- ➤ Max(STOCK\_QUANTITY):Determines the maximum stock available under each return policy.

# **INPUT:**

```
-- Task 6: Analysis Across All Return Policy Categories

SELECT RETURN_POLICY, COUNT(PRODUCT_ID) AS ProductCount,AVG(STOCK_QUANTITY) AS AvgStockQuantity,

SUM(STOCK_QUANTITY) AS TotalStockQuantity,SUM(RATING * REVIEWS) / SUM(REVIEWS) AS WeightedAvgRating,

SUM(REVIEWS) AS TotalReviews,AVG(DISCOUNT) AS AvgDiscount_Percent,AVG(PRICE) AS AvgSellingPrice,

MAX(STOCK_QUANTITY) AS MaxStockQty

FROM Retail

GROUP BY RETURN_POLICY;
```

■R	⊞ Results @ Messages								
	RETURN_POLICY	ProductCount	AvgStockQuantity	TotalStockQuantity	WeightedAvgRating	TotalReviews	AvgDiscount_Percent	AvgSellingPrice	MaxStockQty
1	15 Days	1639	49	80744	2.99778157543317	82697	25.2520625554798	497.241804656494	99
2	30 Days	1664	49	81785	2.98148533670471	83076	25.6931101097311	508.491014549365	99
3	7 Days	1697	50	85709	2.97084874701466	85140	25.8903037726784	509.598194182446	99

# THE END