### **1. Customer Demographic Insights**

### By analyzing the demographic distribution of customers, Global Electronics can better understand their customer base and tailor marketing strategies accordingly.

### **Insight:**

### **Gender Distribution:** Identify if there is a gender skew in the customer base to tailor marketing efforts.

### **Age Distribution:** Identify the predominant age groups and design age-specific marketing campaigns.

### **Location-Based Insights:** Determine the cities, states, and countries with the highest number of customers to focus expansion and marketing efforts.

### **Query for Demographic Distribution:**

create table Customer\_Demographic\_Distribution as

SELECT

Gender,

City,

State\_x,

Country\_x,

Continent,

COUNT(Customer\_ID) AS total\_customers,

AVG(YEAR(CURDATE()) - YEAR(Birthday)) AS average\_age,

MIN(YEAR(CURDATE()) - YEAR(Birthday)) AS youngest\_age,

MAX(YEAR(CURDATE()) - YEAR(Birthday)) AS oldest\_age

FROM

Merged\_Data

GROUP BY

Gender, City, State\_x, Country\_x, Continent

ORDER BY

total\_customers DESC;

#Using Postgres sql

create table Customer\_Demographic\_Distribution as

SELECT

Gender,

City,

State\_x,

Country\_x,

Continent,

COUNT(Customer\_ID) AS total\_customers,

AVG(EXTRACT(YEAR FROM CURRENT\_DATE) - EXTRACT(YEAR FROM Birthday)) AS average\_age,

MIN(EXTRACT(YEAR FROM CURRENT\_DATE) - EXTRACT(YEAR FROM Birthday)) AS youngest\_age,

MAX(EXTRACT(YEAR FROM CURRENT\_DATE) - EXTRACT(YEAR FROM Birthday)) AS oldest\_age

FROM

global\_electronics\_data

GROUP BY

Gender, City, State\_x, Country\_x, Continent

ORDER BY

total\_customers DESC;

### **2. Customer Purchase Patterns**

Understanding customer purchase patterns helps in identifying high-value customers and their preferred products.

**Insight:**

* **Average Order Value**: Identify customers with high average order value for potential loyalty programs.
* **Purchase Frequency**: Determine the frequency of purchases to segment customers into categories like frequent buyers, occasional buyers, etc.
* **Preferred Products**: Identify the most frequently purchased products to optimize inventory and marketing campaigns.

### **Query for Purchase Patterns:**

create table purchase\_pattern as

SELECT

Customer\_ID,

COUNT(DISTINCT Order\_Number) AS total\_orders,

SUM(Unit\_Price\_USD \* Quantity) AS total\_spent,

AVG(Unit\_Price\_USD \* Quantity) AS average\_order\_value,

MIN(Order\_Date) AS first\_order\_date,

MAX(Order\_Date) AS last\_order\_date,

DATEDIFF(MAX(Order\_Date), MIN(Order\_Date)) / COUNT(DISTINCT Order\_Number) AS avg\_days\_between\_orders

FROM

global\_electronics\_data

GROUP BY

Customer\_ID

ORDER BY

total\_spent DESC;

#Using Postgres SQL

create table purchase\_pattern as

SELECT

Customer\_ID,

COUNT(DISTINCT Order\_Number) AS total\_orders,

SUM(Unit\_Price\_USD \* Quantity) AS total\_spent,

AVG(Unit\_Price\_USD \* Quantity) AS average\_order\_value,

MIN(Order\_Date) AS first\_order\_date,

MAX(Order\_Date) AS last\_order\_date,

EXTRACT(DAY FROM AGE(MAX(Order\_Date), MIN(Order\_Date))) / NULLIF(COUNT(DISTINCT Order\_Number), 0) AS avg\_days\_between\_orders

FROM

global\_electronics\_data

GROUP BY

Customer\_ID

ORDER BY

total\_spent DESC;

### **3. Customer Segmentation**

Segmenting customers based on demographics and purchasing behavior helps in targeted marketing and personalized customer experiences.

**Insight:**

* **Age-Based Segments**: Identify key age segments and tailor marketing campaigns accordingly.
* **Spending Segments**: Identify high-spending customers and create loyalty programs to retain them.

**Query for Customer Segmentation:**

#Using mysql:-

CREATE TABLE Customer\_Segmentation AS

SELECT

Gender,

CASE

WHEN (YEAR(CURDATE()) - YEAR(Birthday)) < 25 THEN 'Under 25'

WHEN (YEAR(CURDATE()) - YEAR(Birthday)) BETWEEN 25 AND 40 THEN '25-40'

WHEN (YEAR(CURDATE()) - YEAR(Birthday)) BETWEEN 41 AND 60 THEN '41-60'

ELSE 'Above 60'

END AS age\_group,

Country\_x,

State\_x,

City,

CASE

WHEN SUM(Unit\_Price\_USD \* Quantity) > 1000 THEN 'High Spender'

WHEN SUM(Unit\_Price\_USD \* Quantity) BETWEEN 500 AND 1000 THEN 'Medium Spender'

ELSE 'Low Spender'

END AS spending\_segment,

CASE

WHEN COUNT(DISTINCT Order\_Number) >= 10 THEN 'Frequent Buyer'

WHEN COUNT(DISTINCT Order\_Number) BETWEEN 5 AND 9 THEN 'Occasional Buyer'

ELSE 'Infrequent Buyer'

END AS frequency\_segment,

SUM(Unit\_Price\_USD \* Quantity) AS total\_spent,

COUNT(DISTINCT Order\_Number) AS total\_orders

FROM

global\_electronics\_data

GROUP BY

Gender, age\_group, Country\_x, State\_x, City

ORDER BY

total\_spent DESC;

#Using Postgressql

CREATE TABLE Customer\_Segmentation AS

SELECT

Gender,

CASE

WHEN (EXTRACT(YEAR FROM CURRENT\_DATE) - EXTRACT(YEAR FROM Birthday)) < 25 THEN 'Under 25'

WHEN (EXTRACT(YEAR FROM CURRENT\_DATE) - EXTRACT(YEAR FROM Birthday)) BETWEEN 25 AND 40 THEN '25-40'

WHEN (EXTRACT(YEAR FROM CURRENT\_DATE) - EXTRACT(YEAR FROM Birthday)) BETWEEN 41 AND 60 THEN '41-60'

ELSE 'Above 60'

END AS age\_group,

Country\_x,

State\_x,

City,

CASE

WHEN SUM(Unit\_Price\_USD \* Quantity) > 1000 THEN 'High Spender'

WHEN SUM(Unit\_Price\_USD \* Quantity) BETWEEN 500 AND 1000 THEN 'Medium Spender'

ELSE 'Low Spender'

END AS spending\_segment,

CASE

WHEN COUNT(DISTINCT Order\_Number) >= 10 THEN 'Frequent Buyer'

WHEN COUNT(DISTINCT Order\_Number) BETWEEN 5 AND 9 THEN 'Occasional Buyer'

ELSE 'Infrequent Buyer'

END AS frequency\_segment,

SUM(Unit\_Price\_USD \* Quantity) AS total\_spent,

COUNT(DISTINCT Order\_Number) AS total\_orders

FROM

global\_electronics\_data

GROUP BY

Gender,

Country\_x,

State\_x,

City,

age\_group

ORDER BY

total\_spent DESC;

### **4. Overall Sales Performance**

Analyzing sales trends over time helps in identifying seasonality and planning inventory and promotions accordingly.

**Insight:**

* **Sales Trends**: Identify peak sales periods and plan inventory and promotions around these times.
* **Seasonality**: Understand seasonal variations in sales to optimize marketing and stock levels.

**Query for Overall Sales Performance:**

create table Overall\_Sales\_Performance as

SELECT

YEAR(Order\_Date) AS Year,

QUARTER(Order\_Date) AS Quarter,

MONTH(Order\_Date) AS Month,

WEEK(Order\_Date) AS Week,

SUM(Unit\_Price\_USD \* Quantity) AS Total\_Sales

FROM

global\_electronics\_data

GROUP BY

Year, Quarter, Month, Week

ORDER BY

Year, Quarter, Month, Week;

#Using Postgres SQL

SELECT

EXTRACT (YEAR FROM Order\_Date) AS Year,

EXTRACT ( QUARTER FROM Order\_Date) AS Quarter,

EXTRACT(MONTH FROM Order\_Date) AS Month,

EXTRACT( WEEK FROM Order\_Date) AS Week,

SUM(Unit\_Price\_USD \* Quantity) AS Total\_Sales

FROM

global\_electronics\_data

GROUP BY

Year, Quarter, Month, Week

ORDER BY

Year, Quarter, Month, Week;

### **5. Store Performance**

Analyzing the performance of different stores helps in identifying high-performing locations and areas that need improvement.

**Insight:**

* **High-Performing Stores**: Identify stores with the highest sales and analyze what makes them successful.
* **Underperforming Stores**: Identify stores with low sales and explore reasons for their performance (e.g., location, size, etc.).

**Query for Sales by Store:**

create table sales\_by\_stores as SELECT

StoreKey,

SUM(Unit\_Price\_USD \* Quantity) AS Total\_Sales,

AVG(Square\_Meters) AS Average\_Store\_Size

FROM

global\_electronics\_data

GROUP BY

StoreKey

ORDER BY

Total\_Sales DESC;

#Using Postgres Sql

SELECT

StoreKey,

SUM(Unit\_Price\_USD \* Quantity) AS Total\_Sales,

AVG(Square\_Meters) AS Average\_Store\_Size

FROM

global\_electronics\_data

GROUP BY

StoreKey

ORDER BY

Total\_Sales DESC;

### **6. Top-Performing Products**

Identifying top-performing products helps in focusing on high-revenue items and optimizing the product portfolio.

**Insight:**

* **Product Popularity**: Determine the best-selling products and focus marketing efforts on these items.
* **Revenue Generation**: Identify products that generate the most revenue to ensure they are well-stocked and promoted.

### **Query for Sales by Product:**

create table Sales\_by\_Products as SELECT

Product\_Name,

SUM(Quantity) AS Total\_Quantity\_Sold,

SUM(Unit\_Price\_USD \* Quantity) AS Total\_Revenue

FROM

global\_electronics\_data

GROUP BY

Product\_Name

ORDER BY

Total\_Revenue DESC;

#Using Postgres SQL

SELECT

Product\_Name,

SUM(Quantity) AS Total\_Quantity\_Sold,

SUM(Unit\_Price\_USD \* Quantity) AS Total\_Revenue

FROM

global\_electronics\_data

GROUP BY

Product\_Name

ORDER BY

Total\_Revenue DESC;

### **7. Impact of Currency on Sales**

Understanding how different currencies impact sales helps in setting international pricing strategies.

**Insight:**

* **Currency Impact**: Identify currencies that contribute the most to sales and optimize pricing strategies accordingly.
* **Average Prices and Costs**: Analyze the average selling prices and costs in different currencies to ensure competitive pricing.

### **Query for Sales by Currency:**

create table sales\_by\_currency as SELECT

Currency\_Code,

SUM(Unit\_Price\_USD \* Quantity) AS Total\_Sales,

AVG(Unit\_Cost\_USD) AS Average\_Cost,

AVG(Unit\_Price\_USD) AS Average\_Price

FROM

Merged\_Data

GROUP BY

Currency\_Code

ORDER BY

Total\_Sales DESC;

#Using Postgres SQL

SELECT

Currency\_Code,

SUM(Unit\_Price\_USD \* Quantity) AS Total\_Sales,

AVG(Unit\_Cost\_USD) AS Average\_Cost,

AVG(Unit\_Price\_USD) AS Average\_Price

FROM

global\_electronics\_data

GROUP BY

Currency\_Code

ORDER BY

Total\_Sales DESC;

### **8. Product Profitability**

Calculating profit margins for products helps in focusing on high-margin items and optimizing pricing strategies.

**Insight:**

* **High-Profit Products**: Identify products with the highest profit margins and focus on promoting these items.
* **Low-Profit Products**: Identify products with low profit margins and explore ways to improve their profitability.

### **Query for Product Profitability Analysis:**

create table product\_profitability as

SELECT

Product\_Name,

AVG(Unit\_Price\_USD - Unit\_Cost\_USD) AS Average\_Profit\_Margin,

AVG(Unit\_Price\_USD) AS Average\_Selling\_Price,

AVG(Unit\_Cost\_USD) AS Average\_Cost

FROM

global\_electronics\_data

GROUP BY

Product\_Name

ORDER BY

Average\_Profit\_Margin DESC;

#Using Postgres SQL

SELECT

Product\_Name,

AVG(Unit\_Price\_USD - Unit\_Cost\_USD) AS Average\_Profit\_Margin,

AVG(Unit\_Price\_USD) AS Average\_Selling\_Price,

AVG(Unit\_Cost\_USD) AS Average\_Cost

FROM

global\_electronics\_data

GROUP BY

Product\_Name

ORDER BY

Average\_Profit\_Margin DESC;

**9. Product Popularity**

Analyzing product popularity helps identify which items are most in demand and informs inventory and marketing strategies.

**Insight:**

* **Best-Selling Products:** Identify products with the highest sales volume, enabling focus on maintaining stock levels and promoting these popular items.
* **Underperforming Products:** Highlight products with low sales, allowing the business to reassess marketing, pricing, or even phase out less popular items.

### **Query for Product Popularity Analysis:**

create table product\_popularity as

(

SELECT

ProductKey,

Product\_Name,

SUM(Quantity) AS total\_quantity\_sold

FROM

global\_electronics\_data

GROUP BY

ProductKey, Product\_Name

ORDER BY

total\_quantity\_sold DESC

LIMIT 10 -- Change this number as needed for the most popular products

)

UNION ALL

(

SELECT

ProductKey,

Product\_Name,

SUM(Quantity) AS total\_quantity\_sold

FROM

global\_electronics\_data

GROUP BY

ProductKey, Product\_Name

ORDER BY

total\_quantity\_sold ASC

LIMIT 10 -- Change this number as needed for the least popular products

);

#Using Postgres SQL

(

SELECT

ProductKey,

Product\_Name,

SUM(Quantity) AS total\_quantity\_sold

FROM

global\_electronics\_data

GROUP BY

ProductKey, Product\_Name

ORDER BY

total\_quantity\_sold DESC

LIMIT 10 -- Change this number as needed for the most popular products

)

UNION ALL

(

SELECT

ProductKey,

Product\_Name,

SUM(Quantity) AS total\_quantity\_sold

FROM

global\_electronics\_data

GROUP BY

ProductKey, Product\_Name

ORDER BY

total\_quantity\_sold ASC

LIMIT 10 -- Change this number as needed for the least popular products

);

**10. Category Analysis**

Evaluating product categories helps identify which segments contribute the most to sales and profitability, guiding strategic decisions on inventory and promotions.

**Insight:**

* **Top-Performing Categories:** Identify the product categories that generate the highest revenue, enabling focus on expanding and promoting these segments.
* **Low-Performing Categories:** Recognize categories with low sales, allowing the business to explore ways to improve their performance or consider discontinuation

### **Query for Category Analysis:**

create table category\_analysis as

SELECT

Category,

Subcategory,

SUM(Unit\_Price\_USD \* Quantity) AS total\_sales,

SUM(Quantity) AS total\_quantity\_sold

FROM

Merged\_Data

GROUP BY

Category, Subcategory

ORDER BY

total\_sales DESC;

#Using Postgres SQL

SELECT

Category,

Subcategory,

SUM(Unit\_Price\_USD \* Quantity) AS total\_sales,

SUM(Quantity) AS total\_quantity\_sold

FROM

global\_electronics\_data

GROUP BY

Category, Subcategory

ORDER BY

total\_sales DESC;