SMART PHONE SALES ANALYTICS USING SQL

HANIF BINGINAPALLI

Project Overview

This project analyzes smartphone sales data to extract insights such as top-selling brands, revenue trends, and customer behavior using SQL queries.

SQL is used to clean, process, and analyze the data from multiple relational tables.

PROBLEM STATEMENT

Smartphone companies generate large volumes of sales data, but without proper analysis, it becomes difficult to understand performance trends.

This project solves the challenge of:

- Identifying top-performing brands
- Understanding sales patterns
- Monitoring pricing and revenue trends
- Helping businesses make informed decisions.. etc.

OBJECTIVES

- Identify best-selling brands and models
- Analyze revenue and monthly sales trends
- Evaluate OS-wise performance
- Discover products never sold
- Analyze customer and location-based order patterns

DATASET DESCRIPTION

The dataset contains multiple tables:

- Products: Product ID, Brand, Brand ID, Model, Color , Memory, Storage, Rating, Selling Price, Original Price
- Customers: Customer ID ,First Name, Last Name, Email , City, Country
- Orders: Order ID, Customer ID, Product ID, Order Date, Order Status , Selling Price ,Payment Method
- Order Details: Order details ID ,Order ID ,Product ID , Quantity
- Brand: Brand ID, Brand Name , Operating System

SQL QUERIES PERFORMED

1. Total number of orders
2. Total generated amount per order
3. Highest priced product
4. Top 5 most selling brands
5. Total products per brand
6. OS with highest sales
7. Count orders per order status
8. Count orders per payment method
9. Count orders per customer
10. Products never sold
11. Orders per country and city
12. Monthly orders summary

KEY SQL QUERIES

```sql
1   -- Retrieve The
2   -- Total Number of Orders Placed
3
4•  select count(Order_ID) from orders;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

	count(Order_ID)
▶	1000

```sql
-- Calculate The Total Revenue Generated Form The Orders

SELECT
    FORMAT(SUM(od.Quantity * p.Selling_Price),
        2) AS Total_Revenue_Generated
FROM
    order_details od
        JOIN
    products p ON od.Product_ID = p.Product_ID;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Total_Revenue_Generated
3,655,370,073.00

```sql
-- Identify The Highest Priced Product

SELECT * FROM products
WHERE Selling_Price =
(SELECT MAX(Selling_Price)
FROM products);
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

Product_ID	Brand	Brand_ID	Model	Color	Memory	Storage	Rating	Selling_Price	Original_Price
4045	Apple	1	iPhone 13 Pro Max	Silver	4GB	1 TB	4.7	179900	179900
4052	Apple	1	iPhone 13 Pro Max	Gold	4GB	1 TB	4.7	179900	179900
4058	Apple	1	iPhone 13 Pro Max	Graphite	4GB	1 TB	4.7	179900	179900
4059	Apple	1	iPhone 13 Pro Max	Sierra Blue	4GB	1 TB	4.7	179900	179900
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

```sql
-- Identify The Top 5 Most Selling Brands
SELECT
    p.Brand, COUNT(od.order_details_ID) AS Total_Count
FROM
    products p
        JOIN
    order_details od ON p.Product_ID = od.Product_ID
GROUP BY p.Brand
ORDER BY Total_Count DESC
LIMIT 5;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Brand	Total_Count
SAMSUNG	2627
Apple	1241
realme	1119
OPPO	966
Nokia	716

```sql
-- Calculates the total number of products for each unique Brand
SELECT
    Brand, COUNT(product_ID) AS Quantity
FROM
    products
GROUP BY Brand
ORDER BY Quantity DESC;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

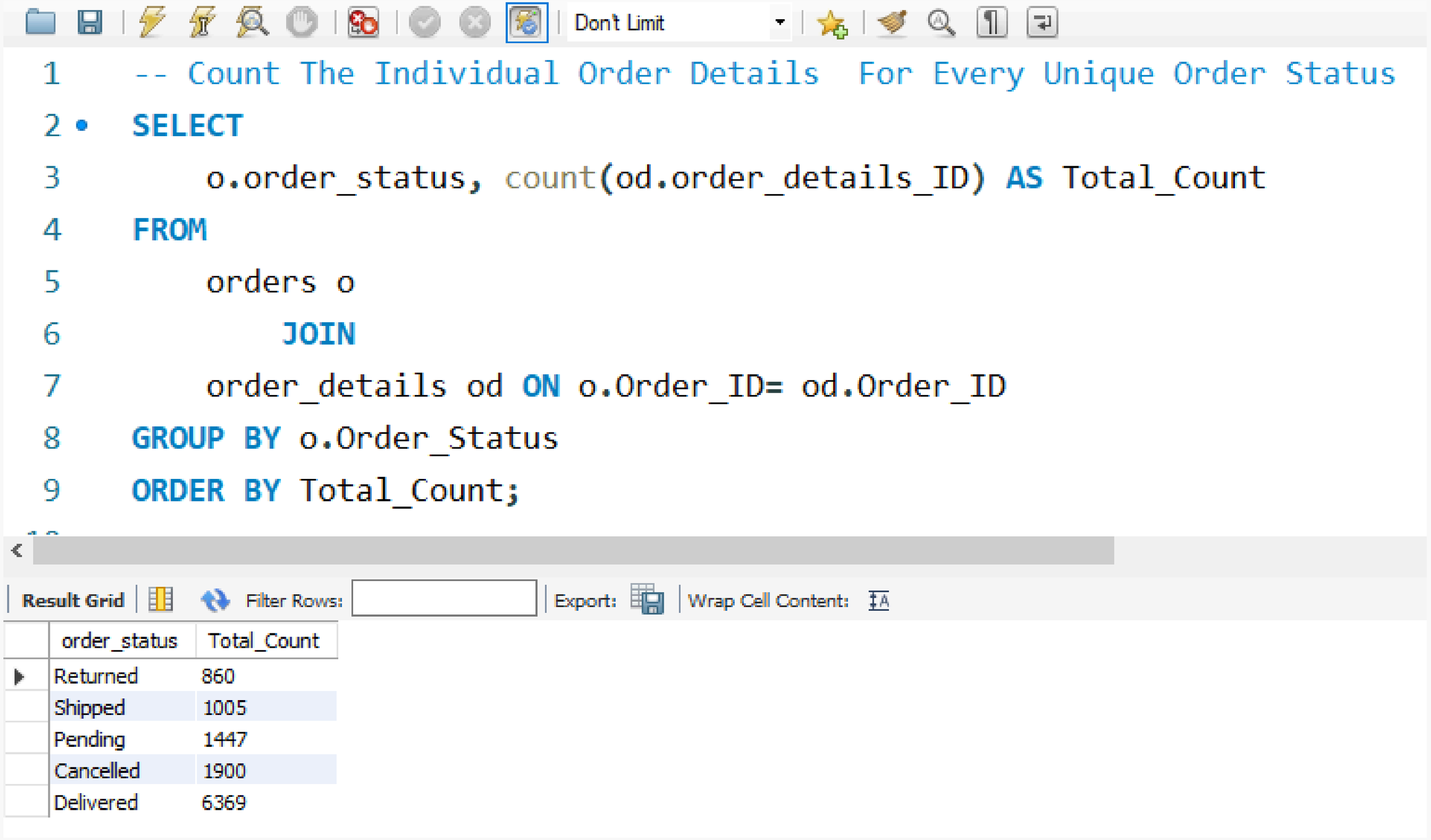
Brand	Quantity
SAMSUNG	708
Apple	334
realme	309
OPPO	251
Xiaomi	192
Nokia	188
Infinix	151
GIONEE	127
Lenovo	120
vivo	116
ASUS	111
Motorola	103
LG	98
POCO	73
HTC	55
Google Pi...	29

```sql
-- Which operating system has the highest number of units sold?
SELECT
    b.operating_system, COUNT(od.order_details_ID) AS Quantity
FROM
    brand b
        JOIN
    products p ON b.Brand_ID = p.Brand_ID
        JOIN
    order_details od ON od.Product_ID = p.Product_ID
GROUP BY b.operating_system
ORDER BY Quantity DESC;

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

operating_system	Quantity
Android	9826
IOS	1241


Don't Limit
-- Count The Individual Order Details For Every Unique Order Status
SELECT
o.order_status, count(od.order_details_ID) AS Total_Count
FROM
orders o
JOIN
order_details od ON o.Order_ID= od.Order_ID
GROUP BY o.Order_Status
ORDER BY Total_Count;
Result Grid
Filter Rows:
Export:
Wrap Cell Content:
order_status
Total_Count
Returned
860
Shipped
1005
Pending
1447
Cancelled
1900
Delivered
6369

```sql
-- count the total number of order details for each unique payment method.
SELECT
    payment_method, COUNT(order_details_ID) AS Total_Count
FROM
    orders
        JOIN
    order_details ON orders.Order_ID = order_details.Order_ID
GROUP BY Payment_Method
ORDER BY Total_Count desc;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

payment_method	Total_Count
Cash on Delivery	6522
UPI	2885
Credit Card	1658
Debit Card	516

```sql
-- Show The Count Of Orders Per Customers.
SELECT
    customer_ID, COUNT(order_ID) AS Total_Orders
FROM
    orders
GROUP BY Customer_ID
ORDER BY Total_Orders desc;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

customer_ID	Total_Orders
CID530	6
CID410	6
CID580	5
CID712	5
CID758	4
CID559	4
CID147	4
CID172	4
CID39	4
CID808	4
CID420	4
CID891	4
CID176	4

```sql
-- Find The Products that Were Never Sold.
SELECT p.* FROM products p LEFT JOIN
    order_details od ON p.Product_ID = od.Product_ID
WHERE
    od.Product_ID IS NULL;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Product_ID	Brand	Brand_ID	Model	Color	Memory	Storage	Rating	Selling_Price	Original_Price
1085	OPPO	12	Reno3 Pro	Midnight Black	8 GB	128 GB	4.4	24768	24990
1111	OPPO	12	F5	Black	4 GB	32 GB	4.4	15500	15500
1121	OPPO	12	F15	Lightening Black	8 GB	128 GB	4.4	18990	22990
1177	OPPO	12	K1	Astral Blue	6 GB	64 GB	4.3	13990	20990
1191	OPPO	12	Reno4 Pro Special Edition	Starry Night	8 GB	128 GB	4.4	34990	37990
1234	OPPO	12	F11 Pro	Thunder Black	6 GB	128 GB	4.5	17990	29990
1253	IQOO	7	3	Tornado Black	8 GB	256 GB	4.4	37990	40990
1337	LG	9	V20a	Titan	4 GB	64 GB	4.1	30990	30990
1389	ASUS	2	ZenFone 5Z	Midnight Blue	6 GB	128 GB	4.5	36299	36299
1456	ASUS	2	Zenfone Selfie	Pink	3 GB	32 GB	4	17999	17999
1538	realme	14	8 5G	Supersonic Black	8 GB	128 GB	4.3	18499	18999
1588	realme	14	Narzo 50A	Oxygen Green	4 GB	64 GB	4.4	11499	12999
1590	realme	14	2	Diamond Black	3 GB		4.5	9990	9990
1616	realme	14	X50 Pro	Moss Green	12 GB	256 GB	4.3	47999	47999
1667	realme	14	8	Cyber Silver	4 GB	128 GB	4.3	15999	16999
1704	realme	14	Narzo 20A	Victory Blue	4 GB	64 GB	4.4	9499	11999
1740	realme	14	6 Pro	Lightning Orange	8 GB	128 GB	4.4	19499	19499
1748	realme	14	X2	Pearl Green	6 GB	128 GB	4.5	19999	19999

```sql
-- Get The Total Orders Per Country & City
SELECT
    c.Country,c.city,
    COUNT(od.order_details_ID) AS Total_orders
FROM
    customers c
JOIN
    Orders o ON o.customer_ID = c.Customer_ID  -- Join Customers to Orders
JOIN
    order_details od ON od.Order_ID = o.Order_ID  -- Join Orders to Order_Details
GROUP BY
    c.city,c.Country
ORDER BY
    Total_Orders DESC;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Country	city	Total_orders
Canada	Vancouver	430
Ireland	Dublin	297
USA	Miami	276
UK	Bristol	211
Italy	Rome	205
Vietnam	Hanoi	205
Denmark	Copenhagen	203
China	Shanghai	188

```sql
-- Retrive The Monthly Orders
SELECT
    -- Use DATE_FORMAT to extract and format Year and Month
    DATE_FORMAT(o.Order_Date, '%Y-%m') AS Order_Month,
    COUNT( o.Order_ID) AS Monthly_Orders_Count
FROM
    orders o
JOIN
    order_details od ON o.Order_ID = od.Order_ID
GROUP BY
    Order_Month
ORDER BY
    Order_Month DESC;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Order_Month	Monthly_Orders_Count
2024-05	804
2024-04	857
2024-03	785
2024-02	472
2024-01	869
2023-11	84
2023-10	130
2023-09	104
2023-08	638
2023-07	832

CONCLUSION

- SQL helped extract meaningful patterns from raw sales data
- Identified top revenue drivers
- Showed seasonal and pricing trends
- Insights can help businesses optimize product strategy

FUTURE SCOPE

- Build a Power BI Dashboard
- Automate monthly reports
- Add forecasting using Python/ML
- Expand dataset with more features

THANK YOU

FOR WATCHING