DATA ANALYST INTERNSHIP

TASK 6

Exploratory Data Analysis (EDA)

Super Store Dataset

Submitted By,

M.B.Arthi

OBJECTIVE:

The objective of this task is to conduct an in-depth sales trend analysis using SQL on the superstore dataset to uncover key business insights. This involves leveraging SQL's aggregation and date functions to evaluate how sales revenue and order volumes change over different time periods. By extracting the year and month from order dates, the data is grouped and analyzed to identify peak sales periods, detect seasonal trends, and monitor the growth or decline in performance over time. The analysis also aims to calculate average sales per order to understand customer purchasing behavior and determine which product categories contribute most to monthly order volumes. Additionally, the task involves identifying the top-performing months and products based on revenue, enabling the business to recognize opportunities for strategic focus. Through this analysis, SQL functions such as SUM(), AVG(), COUNT(DISTINCT), RANK(), and date parsers like STR_TO_DATE(), YEAR(), and MONTH() are applied. The ultimate goal is to transform raw transactional data into actionable insights that can support data-driven decisions in sales forecasting, inventory optimization, marketing campaigns, and overall business strategy.

Query 1

Limit to 1000 rows

```
USE task;
SELECT * FROM superstore;
```

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

ROW_ID	ORDER_ID	ORDER_DATE	SHIP_DATE	SHIP_MODE	CUSTOMER_ID	CUSTOMER_NAME	SEGMENT	COUNTRY	CITY	STA
1	CA-2017-152156	08/11/2017	11/11/2017	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	Kentu
2	CA-2017-152156	08/11/2017	11/11/2017	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	Kentu
3	CA-2017-138688	12/06/2017	16/06/2017	Second Class	DV-13045	Darrin Van Huff	Corporate	United States	Los Angeles	Califc
4	US-2016-108966	11/10/2016	18/10/2016	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Lauderdale	Floric
5	US-2016-108966	11/10/2016	18/10/2016	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Lauderdale	Floric
6	CA-2015-115812	09/06/2015	14/06/2015	Standard Class	BH-11710	Brosina Hoffman	Consumer	United States	Los Angeles	Califc
7	CA-2015-115812	09/06/2015	14/06/2015	Standard Class	BH-11710	Brosina Hoffman	Consumer	United States	Los Angeles	Califc
8	CA-2015-115812	09/06/2015	14/06/2015	Standard Class	BH-11710	Brosina Hoffman	Consumer	United States	Los Angeles	Califc
9	CA-2015-115812	09/06/2015	14/06/2015	Standard Class	BH-11710	Brosina Hoffman	Consumer	United States	Los Angeles	Califc
10	CA-2015-115812	09/06/2015	14/06/2015	Standard Class	BH-11710	Brosina Hoffman	Consumer	United States	Los Angeles	Califc
11	CA-2015-115812	09/06/2015	14/06/2015	Standard Class	BH-11710	Brosina Hoffman	Consumer	United States	Los Angeles	Califc
12	CA-2015-115812	09/06/2015	14/06/2015	Standard Class	BH-11710	Brosina Hoffman	Consumer	United States	Los Angeles	Califc
13	CA-2018-114412	15/04/2018	20/04/2018	Standard Class	AA-10480	Andrew Allen	Consumer	United States	Concord	North
14	CA-2017-161389	05/12/2017	10/12/2017	Standard Class	IM-15070	Irene Maddox	Consumer	United States	Seattle	Wash
15	US-2016-118983	22/11/2016	26/11/2016	Standard Class	HP-14815	Harold Pawlan	Home Off...	United States	Fort Worth	Texa
16	US-2016-118983	22/11/2016	26/11/2016	Standard Class	HP-14815	Harold Pawlan	Home Off...	United States	Fort Worth	Texa
17	CA-2015-105893	11/11/2015	18/11/2015	Standard Class	PK-19075	Pete Kriz	Consumer	United States	Madison	Wisc

superstore 1 | Read Only

The process starts by making use of the task database and running a query against the superstore table. The superstore table contains detailed records of all customer orders through various fields, such as ORDER_ID, ORDER_DATE, SALES, and information regarding customers. The first thing that we will do is query everything in the table in order to understand how our data is structured. One of the most important items of note is that the ORDER_DATE column is represented as DD/MM/YYYY. This will be essential that we perform time based aggregations accurately. This is the data set we will analyze to find the monthly revenue, number of orders and trends in sales. We will be able to use SQL data aggregations and date functions so that we can generate meaningful business insight.

Monthly Revenue

```sql
SELECT
    YEAR(STR_TO_DATE(ORDER_DATE, '%d/%m/%Y')) AS year,
    MONTH(STR_TO_DATE(ORDER_DATE, '%d/%m/%Y')) AS month,
    SUM(SALES) AS monthly_revenue,
    COUNT(DISTINCT ORDER_ID) AS order_volume
FROM superstore
WHERE ORDER_DATE IS NOT NULL
GROUP BY year, month
ORDER BY year, month;
```

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

year	month	monthly_revenue	order_volume
2015	5	55.5	1
2015	6	3714.304	1
2015	8	444.016	2
2015	9	617.7	1
2015	10	342.13800000000003	3
2015	11	685.18	2
2015	12	1881.55	2
2016	1	99.26000000000002	1
2016	3	787.53	1
2016	4	1582.2190000000003	4
2016	9	5240.120000000001	4
2016	10	1319.9055	2
2016	11	678.904	3
2016	12	1228.9532	1
2017	1	102.21799999999999	1

The executed SQL query successfully analyzes monthly sales trends in the `superstore` dataset using `STR_TO_DATE` to extract year and month from `ORDER_DATE` field and summarizes both total revenue (`SUM(SALES)`) and number of distinct orders (`COUNT(DISTINCT ORDER_ID)`) for each month. The result gives a clear sense of how sales and orders fluctuate over time, and would help reveal trends like the months when sales were highest or with lowest order activity. This the first step in the analysis of sales; going forward it enables additional business understanding and decisions.

Top 3 Months by Revenue

```sql
SELECT
    YEAR(STR_TO_DATE(ORDER_DATE, '%d/%m/%Y')) AS year,
    MONTH(STR_TO_DATE(ORDER_DATE, '%d/%m/%Y')) AS month,
    SUM(SALES) AS total_revenue
FROM superstore
GROUP BY year, month
ORDER BY total_revenue DESC
LIMIT 3;
```

Result Grid

year	month	total_revenue
2016	9	5240.120000000001
2015	6	3714.304
2017	12	3229.82

The SQL query retrieves the top three months with the highest total revenue from the superstore dataset. By using STR_TO_DATE to correctly parse the ORDER_DATE, the query extracts both the year and month, groups the data accordingly, and computes the total monthly revenue using SUM(SALES). The results are ordered in descending order of revenue, showing that September 2016, June 2015, and December 2017 were the highest-grossing months. This insight helps identify peak sales periods for strategic planning and marketing focus.

Total Sales by Year

```sql
SELECT
    YEAR(STR_TO_DATE(ORDER_DATE, '%d/%m/%Y')) AS year,
    SUM(SALES) AS annual_sales
FROM superstore
GROUP BY year
ORDER BY year;
```

Result Grid

year	annual_sales
2015	7740.387999999999
2016	10936.8917
2017	7257.644000000002
2018	2675.016

This SQL query calculates the total annual sales from the superstore dataset by extracting the year from the ORDER_DATE field and summing the SALES for each year. The query groups the data by year and displays the total revenue generated annually in ascending order. The results reveal that 2016 had the highest annual sales, followed by 2017, 2015, and then 2018. This analysis helps assess the overall sales performance year over year and identify periods of growth or decline.

Monthly Average Sales per Order

```sql
SELECT
    YEAR(STR_TO_DATE(ORDER_DATE, '%d/%m/%Y')) AS year,
    MONTH(STR_TO_DATE(ORDER_DATE, '%d/%m/%Y')) AS month,
    AVG(SALES) AS average_sales
FROM superstore
GROUP BY year,month
ORDER BY year,month;
```

Result Grid

year	month	average_sales
2015	5	55.5
2015	6	530.6148571428572
2015	8	63.43085714285714
2015	9	617.7
2015	10	85.53450000000001
2015	11	342.59
2015	12	627.1833333333333
2016	1	33.08666666666667
2016	3	787.53
2016	4	175.80211111111115
2016	9	582.2355555555556
2016	10	439.9685
2016	11	84.863
2016	12	307.2383
2017	1	51.108999999999995
2017	3	31.72

This query calculates the monthly average sales from the superstore dataset by extracting the year and month from the ORDER_DATE field using STR_TO_DATE, then applying the AVG(SALES) function. The result displays how the average sales per order varied across each month and year. These insights help understand customer spending behavior over time and highlight months with unusually high or low average sales values. This kind of analysis is useful for identifying seasonal trends and evaluating pricing strategies.

Number of Orders per Category by Month

Query 1

```
SELECT
    YEAR(STR_TO_DATE(ORDER_DATE, '%d/%m/%Y')) AS year,
    MONTH(STR_TO_DATE(ORDER_DATE, '%d/%m/%Y')) AS month,
    CATEGORY,
    COUNT(DISTINCT "ORDER_ID") AS total_orders
FROM superstore
GROUP BY year,month, CATEGORY
ORDER BY year,month, CATEGORY DESC;
```

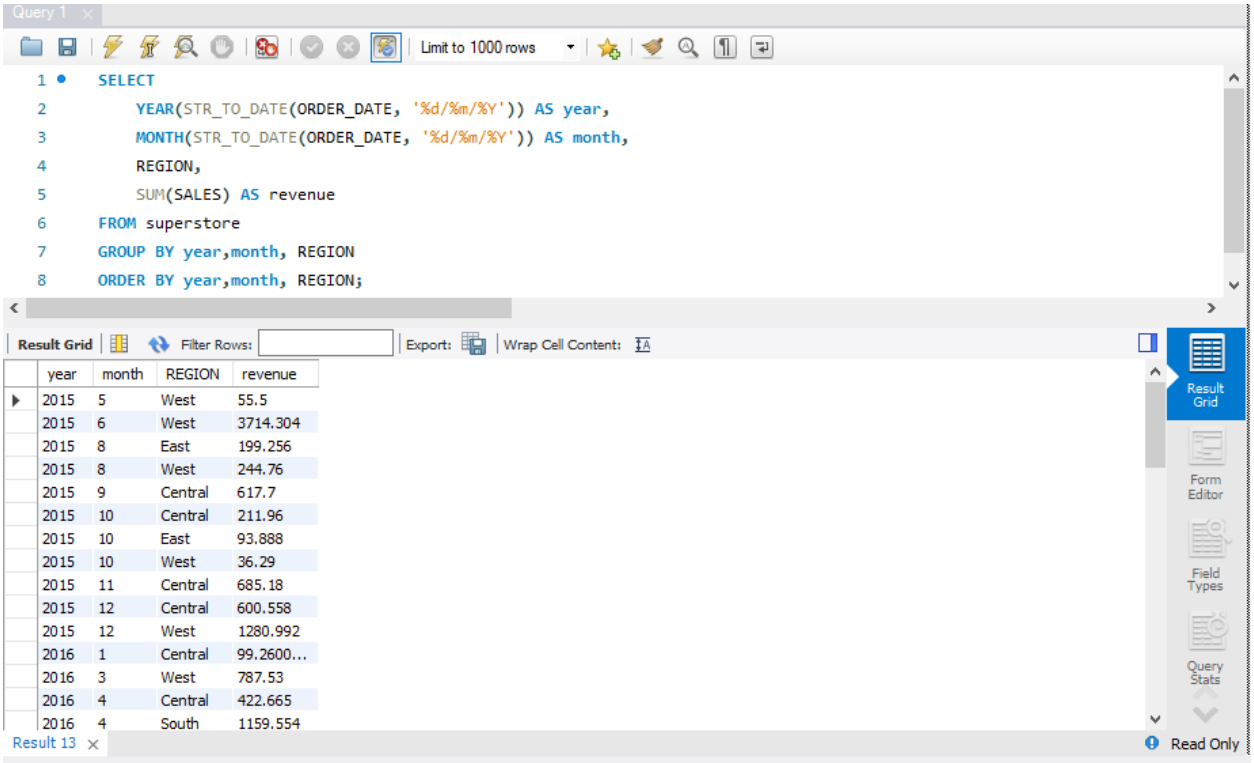
Result Grid

year	month	CATEGORY	total_orders
2015	5	Office Supplies	1
2015	6	Technology	1
2015	6	Office Supplies	1
2015	6	Furniture	1
2015	8	Technology	1
2015	8	Office Supplies	1
2015	9	Furniture	1
2015	10	Office Supplies	1
2015	10	Furniture	1
2015	11	Office Supplies	1
2015	11	Furniture	1
2015	12	Technology	1
2015	12	Office Supplies	1
2015	12	Furniture	1
2016	1	Office Supplies	1

Result 12

This query analyzes the number of distinct orders placed each month across different product categories in the superstore dataset. By extracting the year and month from ORDER_DATE and grouping the results by both CATEGORY and time, the query uses COUNT(DISTINCT ORDER_ID) to calculate the total number of unique orders per category per month. The results, ordered by year, month, and category in descending order, reveal the distribution of orders among categories such as Office Supplies, Technology, and Furniture over time. This analysis helps understand category-wise demand trends and customer purchasing patterns..

Revenue by Region Over Time



year	month	REGION	revenue
2015	5	West	55.5
2015	6	West	3714.304
2015	8	East	199.256
2015	8	West	244.76
2015	9	Central	617.7
2015	10	Central	211.96
2015	10	East	93.888
2015	10	West	36.29
2015	11	Central	685.18
2015	12	Central	600.558
2015	12	West	1280.992
2016	1	Central	99.2600...
2016	3	West	787.53
2016	4	Central	422.665
2016	4	South	1159.554

The SQL query extracts the year and month from the ORDER_DATE column in the superstore table and groups sales data by year, month, and region. It uses the STR_TO_DATE function to convert string dates into proper date format. The SUM(SALES) function is used to calculate the total revenue for each group. The result is then sorted by year, month, and region. The output shows aggregated revenue values by region for each month across 2015 and 2016. This helps analyze Regional sales trends over time.

Highest Selling Product per Year

```sql
SELECT year, product_name, total_sales
FROM (
    SELECT
        YEAR(STR_TO_DATE(ORDER_DATE, '%d/%m/%Y')) AS year,
        `PRODUCT_NAME` AS product_name,
        SUM(SALES) AS total_sales,
        RANK() OVER (
            PARTITION BY YEAR(STR_TO_DATE(ORDER_DATE, '%d/%m/%Y'))
```

year	product_name	total_sales
2015	Chromcraft Rectangular Conference Tables	1706.184
2016	Riverside Palais Royal Lawyers Bookcase, Royal...	3083.43
2017	GE 30524EE4	1097.544
2018	Sanyo 2.5 Cubic Foot Mid-Size Office Refrigerat...	839.43

This SQL query identifies the top-selling product each year from the sales data. It extracts the year from ORDER_DATE, groups sales by PRODUCT_NAME, and calculates total sales using SUM(SALES). The RANK() window function is used to assign a rank to products within each year, ordered by total sales. The outer query filters the result to show only the highest-ranked product per year. The output table displays the year, top product, and corresponding total sales, helping to analyze yearly best-sellers in the dataset.

Tools Used:

- My SQL

ANALYSIS

- The ORDER_DATE column was parsed using STR_TO_DATE() to correctly extract YEAR and MONTH values for time-based grouping.
- Monthly revenue and order volume were calculated using SUM(SALES) and COUNT(DISTINCT ORDER_ID), highlighting fluctuations in sales activity over time.
- The top three months with the highest revenue were identified using ORDER BY total_revenue DESC LIMIT 3, helping to recognize peak sales periods.
- Annual sales analysis showed that 2016 had the highest revenue among all years, indicating strong business performance during that period.
- Average monthly sales per order were calculated using AVG(SALES) to understand customer spending behavior and trends.
- Analysis of order volume by product category revealed that Office Supplies consistently had the highest number of orders, followed by Technology and Furniture.
- SQL aggregation functions (SUM, AVG, COUNT(DISTINCT)) and date functions (YEAR, MONTH, STR_TO_DATE) were utilized to generate meaningful insights.
- This analysis provides a clear overview of sales trends, enabling informed decision-making in areas such as marketing, inventory planning, and sales strategy.