SuperMarkets Performance Analysis

Presented by Salma Raafat

Project Story

This project analyzes the performance of retail stores using a dataset uploaded to SQL Server. Through 15 structured SQL questions, we explored patterns in store area, items available, customer traffic, and sales.

We started by finding averages, then identified top and bottom performing stores. We examined how store size and customer count affect sales, checked for outliers, and measured efficiency using metrics like sales per item and sales per customer.

Our findings show that:

- Bigger stores don’t always generate higher sales.
- Some stores outperform others despite having fewer resources.
- A few stores have unusually high or low sales — these may need deeper investigation.
- Sales per customer is a useful way to compare performance fairly.

This analysis helps highlight strong stores, underperformers, and opportunities for business improvement — all driven by simple SQL queries and clear business questions.

Project overview

This project aims to analyze store performance using a retail dataset uploaded to SQL Server. By writing and executing SQL queries, we explore relationships between store attributes like area, items available, customer traffic, and store sales. The project is driven by questions that help uncover patterns, inefficiencies, and high-performing stores.

Description

Column Name	Description
Store_ID	Unique identifier for each store
Store_Area	Size of the store in square feet
Items_Available	Total number of items available in the store
Daily_Customer_Count	Average number of customers per day
Store_Sales	Total daily sales for the store

1. What is the average sales value across all stores?

```sql
SELECT AVG(Store_Sales) AS Avg_Store_Sales
FROM Stores;
```

	Avg_Store_Sales
1	59351

2. Which store recorded the highest total sales?

```sql
SELECT TOP 1 Store_ID, Store_Sales
FROM Stores
ORDER BY Store_Sales DESC;
```

	Store_ID	Store_Sales
1	650	116320

3. Which store recorded the lowest total sales?

```sql
SELECT TOP 1 Store_ID, Store_Sales
FROM Stores
ORDER BY Store_Sales ASC;
```

	Store_ID	Store_Sales
1	32	14920

4. How do store sales vary based on store area?

```sql
SELECT TOP 5 Store_ID, Store_Area, Store_Sales
FROM Stores
ORDER BY Store_Area;
```

Store_ID	Store_Area	Store_Sales
159	775	42530
866	780	25600
745	854	77740
550	869	52540
73	891	67370

5. Which stores have the highest sales per square meter?

```sql
SELECT TOP 5
    Store_ID,
    Store_Sales,
    Store_Area,
    CAST(Store_Sales AS FLOAT) / Store_Area AS Sales_per_Sqm
FROM Stores
ORDER BY Sales_per_Sqm DESC;
```

	Store_ID	Store_Sales	Store_Area	Sales_per_Sqm
1	557	89760	932	96.3090128755365
2	469	97360	1060	91.8490566037736
3	745	77740	854	91.0304449648712
4	558	101780	1137	89.5162708883026
5	340	85470	966	88.4782608695652

6. Highest sales per customer

```sql
SELECT TOP 5
    Store_ID,
    Store_Sales,
    Daily_Customer_Count,
    CAST(Store_Sales AS FLOAT) / Daily_Customer_Count AS Sales_per_Customer
FROM Stores
ORDER BY Sales_per_Customer DESC;
```

	Store_ID	Store_Sales	Daily_Customer_Count	Sales_per_Customer
1	40	45480	10	4548
2	152	63730	70	910.428571428571
3	877	88270	100	882.7
4	888	92370	140	659.785714285714
5	231	72350	120	602.916666666667

7. Highest sales per item

```sql
SELECT TOP 5
    Store_ID,
    Store_Sales,
    Items_Available,
    CAST(Store_Sales AS FLOAT) / Items_Available AS Sales_per_Item
FROM Stores
ORDER BY Sales_per_Item DESC;
```

	Store_ID	Store_Sales	Items_Available	Sales_per_Item
1	557	89760	1138	78.8752196836555
2	745	77740	1018	76.3654223968566
3	469	97360	1279	76.1219702892885
4	558	101780	1374	74.0756914119359
5	690	94370	1276	73.9576802507837

8. High customer count but low sales

```sql
SELECT TOP 5
    Store_ID,
    Daily_Customer_Count,
    Store_Sales
FROM Stores
WHERE Daily_Customer_Count > (SELECT AVG(Daily_Customer_Count) FROM Stores)
  AND Store_Sales < (SELECT AVG(Store_Sales) FROM Stores)
ORDER BY Daily_Customer_Count DESC;
```

	Store_ID	Daily_Customer_Count	Store_Sales
1	350	1530	40350
2	536	1490	35420
3	313	1450	43640
4	335	1440	44660
5	84	1410	42670

9. Low customer count but high sales

```sql
SELECT TOP 5
    Store_ID,
    Daily_Customer_Count,
    Store_Sales
FROM Stores
WHERE Daily_Customer_Count < (SELECT AVG(Daily_Customer_Count) FROM Stores)
  AND Store_Sales > (SELECT AVG(Store_Sales) FROM Stores)
ORDER BY Store_Sales DESC;
```

	Store_ID	Daily_Customer_Count	Store_Sales
1	433	680	102920
2	558	700	101780
3	167	680	99570
4	693	480	99480
5	469	670	97360

10. Group by item range

```sql
SELECT TOP 5
  Items_Available,
  AVG(Store_Sales) AS Avg_Sales
FROM Stores
GROUP BY Items_Available
ORDER BY Items_Available;
```

	Items_Available	Avg_Sales
1	932	42530
2	951	25600
3	1018	77740
4	1050	52540
5	1059	75110

11. High sales with few items

```sql
SELECT TOP 5
    Store_ID,
    Items_Available,
    Store_Sales
FROM Stores
WHERE Items_Available < (SELECT AVG(Items_Available) FROM Stores)
  AND Store_Sales > (SELECT AVG(Store_Sales) FROM Stores)
ORDER BY Store_Sales DESC;
```

	Store_ID	Items_Available	Store_Sales
1	433	1638	102920
2	409	1587	102310
3	759	1758	101820
4	558	1374	101780
5	167	1763	99570

12. Top 5 by total sales

```sql
SELECT TOP 5
    Store_ID,
    Store_Sales
FROM Stores
ORDER BY Store_Sales DESC;
```

	Store_ID	Store_Sales
1	650	116320
2	869	105150
3	433	102920
4	409	102310
5	759	101820

13. Most efficient by area, customer, item

```sql
SELECT TOP 5
    Store_ID,
    CAST(Store_Sales AS FLOAT) / Store_Area AS Sales_per_Area,
    CAST(Store_Sales AS FLOAT) / Daily_Customer_Count AS Sales_per_Customer,
    CAST(Store_Sales AS FLOAT) / Items_Available AS Sales_per_Item
FROM Stores
ORDER BY Sales_per_Area DESC, Sales_per_Customer DESC, Sales_per_Item DESC;
```

	Store_ID	Sales_per_Area	Sales_per_Customer	Sales_per_Item
1	557	96.3090128755365	109.463414634146	78.8752196836555
2	469	91.8490566037736	145.313432835821	76.1219702892885
3	745	91.0304449648712	117.787878787879	76.3654223968566
4	558	89.5162708883026	145.4	74.0756914119359
5	340	88.4782608695652	94.9666666666667	72.9266211604096

14. Big store, small sales

```sql
SELECT TOP 5
    Store_ID,
    Store_Area,
    Store_Sales
FROM Stores
WHERE Store_Area > (SELECT AVG(Store_Area) FROM Stores)
  AND Store_Sales < (SELECT AVG(Store_Sales) FROM Stores)
ORDER BY Store_Area DESC;
```

	Store_ID	Store_Area	Store_Sales
1	399	2063	51480
2	568	2026	51240
3	785	2013	47060
4	62	2011	46100
5	345	2004	50130

15. Possible outliers - highest and lowest sales

```sql
SELECT TOP 5
    Store_ID,
    Store_Sales
FROM Stores
ORDER BY Store_Sales DESC;
-- To check lowest, run separately:
SELECT TOP 5
    Store_ID,
    Store_Sales
FROM Stores
ORDER BY Store_Sales ASC;
```

	Store_ID	Store_Sales
1	650	116320
2	869	105150
3	433	102920
4	409	102310
5	759	101820

	Store_ID	Store_Sales
1	32	14920
2	853	16370
3	776	17670
4	594	20270
5	353	21300

Conclusion

This analysis provided a deeper understanding of store performance beyond total sales figures. By exploring metrics like sales per area, per customer, and per item, we uncovered which stores operate most efficiently.

The findings show that high sales do not always equate to high efficiency, and some large stores may underperform despite their size.

Identifying outliers and comparing average performance helped pinpoint strengths, inefficiencies, and improvement opportunities.

- Ultimately, this project demonstrates how data-driven analysis can support smarter retail decisions, from resource allocation to performance benchmarking.

Thank You So Much

Salma Raafat