Blinkit Data Analysis

- **BUSINESS REQUIREMENT**

KPI's Requirements:-

i.   **Total Sales**: The overall revenue generated from all items sold.
ii. **Average Sales**: The average revenue per sale.
iii. **Number of Items**: The total count of different items sold.
iv. **Average Rating**: The average customer rating for items sold

Granular Requirements:-

i.   **Total Sales by Fat Content:**
   - Objective: Analyze the impact of fat content on total sales.
   - Additional KPI Metrics: Assess how other KPIs (Average Sales, Number of Items, Average Rating) vary with fat content.

ii. **Total Sales by Item Type**:
   - Objective: Identify the performance of different item types in terms of total sales.
   - Additional KPI Metrics: Assess how other KPIs (Average Sales, Number of Items, Average Rating) vary with fat content

iii. **Fat Content by Outlet for Total Sales**:
   - Objective: Compare total sales across different outlets segmented by fat content.
   - Additional KPI Metrics: Assess how other KPIs (Average Sales, Number of Items, Average Rating) vary with fat content.

iv. **Total Sales by Outlet Establishment**:
   - Objective: Evaluate how the age or type of outlet establishment influences total sales.

v.   **Percentage of Sales by Outlet Size:**
   - Objective: Analyze the correlation between outlet size and total sales.

vi. **Sales by Outlet Location:**
   - Objective: Assess the geographic distribution of sales across different locations

vii. **All Metrics by Outlet Type:**
   - Objective: Provide a comprehensive view of all key metrics (Total Sales, Average Sales, Number of Items, Average Rating) broken down by different outlet types.

- To See all the data imported:

```sql
SELECT * FROM blinkit_data
```

- **DATA CLEANING:**

  Cleaning the Item_Fat_Content field ensures data consistency and accuracy in analysis. The presence of multiple variations of the same category (e.g., LF, low fat vs. Low Fat) can cause issues in reporting, aggregations, and filtering. By standardizing these values, we improve data quality, making it easier to generate insights and maintain uniformity in our datasets.

```sql
UPDATE blinkit_data
SET Item_Fat_Content =
    CASE
        WHEN Item_Fat_Content IN ('LF', 'low fat') THEN 'Low Fat'
        WHEN Item_Fat_Content = 'reg' THEN 'Regular'
        ELSE Item_Fat_Content
        END;
```

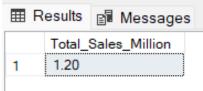 After executing this query to check the data has been cleaned or not, I used the below query

```sql
SELECT DISTINCT Item_Fat_Content FROM blinkit_data;
```

	Item_Fat_Content
1	Low Fat
2	Regular

KEY PERFORMANCE INDICATORS

1. TOTAL SALES:

```sql
SELECT CAST(SUM(Total_Sales) / 1000000.0 AS DECIMAL(10,2)) AS
Total_Sales_Million
FROM blinkit_data;
```

	Total_Sales_Million
1	1.20

2. AVERAGE SALES

```sql
SELECT CAST(AVG(Total_Sales) AS INT) AS Avg_Sales
FROM blinkit_data;
```

	Avg_Sales
1	140

3. NO OF ITEMS

```sql
SELECT COUNT(*) AS No_of_Orders
FROM blinkit_data;
```

	No_of_Orders
1	8523

4. AVG RATING

```sql
SELECT CAST(AVG(Rating) AS DECIMAL(10,1)) AS Avg_Rating
FROM blinkit_data;
```

	Avg_Rating
1	4.0

1. Total Sales by Fat Content:

```sql
SELECT Item_Fat_Content, CAST(SUM(Total_Sales) AS DECIMAL(10,2)) AS
Total_Sales
FROM blinkit_data
GROUP BY Item_Fat_Content
```

	Item_Fat_Content	Total_Sales
1	Low Fat	776319.68
2	Regular	425361.80

2. Total Sales by Item Type

```sql
SELECT Item_Type, CAST(SUM(Total_Sales) AS DECIMAL(10,2)) AS Total_Sales
FROM blinkit_data
GROUP BY Item_Type
ORDER BY Total_Sales DESC
```

	Item_Type	Total_Sales
1	Fruits and Vegetables	178124.08
2	Snack Foods	175433.92
3	Household	135976.53
4	Frozen Foods	118558.88
5	Dairy	101276.46
6	Canned	90706.73
7	Baking Goods	81894.74
8	Health and Hygiene	68025.84
9	Meat	59449.86
10	Soft Drinks	58514.16
11	Breads	35379.12
12	Hard Drinks	29334.68
13	Others	22451.89
14	Starchy Foods	21880.03
15	Breakfast	15596.70
16	Seafood	9077.87

3. Fat Content by Outlet for Total Sales

```sql
SELECT Outlet_Location_Type,
        ISNULL([Low Fat], 0) AS Low_Fat,
        ISNULL([Regular], 0) AS Regular
FROM
(
    SELECT Outlet_Location_Type, Item_Fat_Content,
            CAST(SUM(Total_Sales) AS DECIMAL(10,2)) AS Total_Sales
    FROM blinkit_data
    GROUP BY Outlet_Location_Type, Item_Fat_Content
) AS SourceTable
PIVOT
(
    SUM(Total_Sales)
    FOR Item_Fat_Content IN ([Low Fat], [Regular])
) AS PivotTable
ORDER BY Outlet_Location_Type;
```

	Outlet_Location_Type	Low_Fat	Regular
1	Tier 1	215047.91	121349.90
2	Tier 2	254464.77	138685.87
3	Tier 3	306806.99	165326.03

4. Total Sales by Outlet Establishment

```sql
SELECT Outlet_Establishment_Year, CAST(SUM(Total_Sales) AS DECIMAL(10,2))
AS Total_Sales
FROM blinkit_data
GROUP BY Outlet_Establishment_Year
ORDER BY Outlet_Establishment_Year
```

5. Percentage of Sales by Outlet Size

```sql
SELECT
    Outlet_Size,
    CAST(SUM(Total_Sales) AS DECIMAL(10,2)) AS Total_Sales,
    CAST((SUM(Total_Sales) * 100.0 / SUM(SUM(Total_Sales)) OVER()) AS
DECIMAL(10,2)) AS Sales_Percentage
FROM blinkit_data
GROUP BY Outlet_Size
ORDER BY Total_Sales DESC;
```


6. Sales by Outlet Location

```sql
SELECT Outlet_Location_Type, CAST(SUM(Total_Sales) AS DECIMAL(10,2)) AS
Total_Sales
FROM blinkit_data
GROUP BY Outlet_Location_Type
```

```sql
ORDER BY Total_Sales DESC
```

	Outlet_Location_Type	Total_Sales
1	Tier 3	472133.03
2	Tier 2	393150.64
3	Tier 1	336397.81

7. All Metrics by Outlet Type:

```sql
SELECT Outlet_Type,
        CAST(SUM(Total_Sales) AS DECIMAL(10,2)) AS Total_Sales,
        CAST(AVG(Total_Sales) AS DECIMAL(10,0)) AS Avg_Sales,
        COUNT(*) AS No_Of_Items,
        CAST(AVG(Rating) AS DECIMAL(10,2)) AS Avg_Rating,
        CAST(AVG(Item_Visibility) AS DECIMAL(10,2)) AS Item_Visibility
FROM blinkit_data
GROUP BY Outlet_Type
ORDER BY Total_Sales DESC
```

	Outlet_Type	Total_Sales	Avg_Sales	No_Of_Items	Avg_Rating	Item_Visibility
1	Supermarket Type1	787549.89	141	5577	3.96	0.06
2	Grocery Store	151939.15	140	1083	3.99	0.10
3	Supermarket Type2	131477.77	142	928	3.97	0.06
4	Supermarket Type3	130714.67	140	935	3.95	0.06