**Blinkit Analysis**

* See all the data imported:

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.

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 check the data has been cleaned or not using below query

SELECT DISTINCT Item\_Fat\_Content FROM blinkit\_data;



**A. KPI’s**

**1. TOTAL SALES:**

SELECT CAST(SUM(Total\_Sales) / 1000000.0 AS DECIMAL(10,2)) AS Total\_Sales\_Million

FROM blinkit\_data;

****

**2. AVERAGE SALES**

SELECT CAST(AVG(Total\_Sales) AS INT) AS Avg\_Sales

FROM blinkit\_data;

****

**3. NO OF ITEMS**

SELECT COUNT(\*) AS No\_of\_Orders

FROM blinkit\_data;

****

**4. AVG RATING**

SELECT CAST(AVG(Rating) AS DECIMAL(10,1)) AS Avg\_Rating

FROM blinkit\_data;

****

**B. Total Sales by Fat Content:**

SELECT Item\_Fat\_Content, CAST(SUM(Total\_Sales) AS DECIMAL(10,2)) AS Total\_Sales

FROM blinkit\_data

GROUP BY Item\_Fat\_Content

****

**C. Total Sales by Item Type**

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

****

**D. Fat Content by Outlet for Total Sales**

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;

****

**Query Explanations**

This query aims to transform the blinkit\_data table to display total sales (Total\_Sales) for each combination of Outlet\_Location\_Type and Item\_Fat\_Content. The result will show Outlet\_Location\_Type as rows and Item\_Fat\_Content categories ("Low Fat" and "Regular") as columns. If there are no sales for a particular combination, the query will display 0 instead of NULL.

Detailed Explanation:

1. Subquery
   * Aggregation:

sql

CopyEdit

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

* + - Purpose: This subquery groups the data by Outlet\_Location\_Type and Item\_Fat\_Content, calculating the total sales for each combination.
    - CAST(SUM(Total\_Sales) AS DECIMAL(10,2)): Sums the Total\_Sales for each group and casts the result to a decimal with two decimal places for precision.

1. PIVOT Operation:
   * Pivoting:

sql

CopyEdit

PIVOT

(

SUM(Total\_Sales)

FOR Item\_Fat\_Content IN ([Low Fat], [Regular])

) AS PivotTable

* + - Purpose: Transforms the rows of Item\_Fat\_Content into columns ([Low Fat] and [Regular]).
    - SUM(Total\_Sales): Aggregates the Total\_Sales for each Item\_Fat\_Content category within each Outlet\_Location\_Type.

1. Main Query:
   * Selecting and Handling NULLs:

sql

CopyEdit

SELECT

Outlet\_Location\_Type,

ISNULL([Low Fat], 0) AS Low\_Fat,

ISNULL([Regular], 0) AS Regular

FROM

PivotTable

ORDER BY

Outlet\_Location\_Type;

* + - ISNULL([Low Fat], 0) AS Low\_Fat: Replaces any NULL values in the [Low Fat] column with 0 and renames the column to Low\_Fat.
    - ISNULL([Regular], 0) AS Regular: Similarly, replaces NULL values in the [Regular] column with 0.
    - ORDER BY Outlet\_Location\_Type: Sorts the final result set by Outlet\_Location\_Type.

Why Use ISNULL?

When performing a PIVOT operation, if a particular combination of Outlet\_Location\_Type and Item\_Fat\_Content doesn't exist in the data, the resulting cell will contain a NULL value. Using ISNULL(column)

**E. Total Sales by Outlet Establishment**

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

****

**F. Percentage of Sales by Outlet Size**

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;

**Query Explanation:**

**Outlet\_Size**: This column represents the size category of the outlet (e.g., Small, Medium, Large).

**CAST(SUM(Total\_Sales) AS DECIMAL(10,2)) AS Total\_Sales**:

* **SUM(Total\_Sales)**: Calculates the total sales for each Outlet\_Size.
* **CAST(... AS DECIMAL(10,2))**: Formats the resulting sum to a decimal number with two decimal places for precision.

**CAST((SUM(Total\_Sales) \* 100.0 / SUM(SUM(Total\_Sales)) OVER()) AS DECIMAL(10,2)) AS Sales\_Percentage**:

* **SUM(Total\_Sales) \* 100.0**: Multiplies the total sales of the current Outlet\_Size by 100 to prepare for percentage calculation.
* **SUM(SUM(Total\_Sales)) OVER()**:
  + **SUM(Total\_Sales)**: Within the GROUP BY context, this computes the total sales for each Outlet\_Size.
  + **SUM(... ) OVER()**: The outer SUM combined with the OVER() clause calculates the grand total of all Total\_Sales across all outlet sizes without collapsing the result set.
* **SUM(Total\_Sales) \* 100.0 / SUM(SUM(Total\_Sales)) OVER()**: Divides the total sales of the current Outlet\_Size by the grand total sales and multiplies by 100 to get the percentage contribution of each outlet size to the overall sales.
* **CAST(... AS DECIMAL(10,2))**: Formats the resulting percentage to two decimal places.

****

**G. Sales by Outlet Location**

SELECT Outlet\_Location\_Type, CAST(SUM(Total\_Sales) AS DECIMAL(10,2)) AS Total\_Sales

FROM blinkit\_data

GROUP BY Outlet\_Location\_Type

ORDER BY Total\_Sales DESC

****

**H. All Metrics by Outlet Type:**

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

****