PRESENTATION

By KEVIN SHIBU

BACKGROUND

The FMCG (Fast-Moving Consumer Goods) sector relies heavily on data-driven insights to optimize supply chains, improve sales performance, and enhance customer experience. This project focuses on analyzing grocery sales data from Blinkit, a leading instant commerce platform. By leveraging MySQL for data storage, cleaning, and querying, this project aims to extract meaningful business insights from the dataset, which includes sales, item categories, outlet information, and customer preferences.

PROJECT GOALS

- Data Cleaning & Standardization
- Ensure data consistency (e.g., standardizing item fat content categories).
- Remove inconsistencies for better analysis and reporting.
- Sales Analysis
- Calculate total and average sales.
- Determine sales distribution by item type and outlet type.
- Outlet Performance Evaluation
- Analyze sales by outlet location and establishment year.
- Compare sales performance based on outlet size and type.
- Customer Preferences & Demand Analysis
- Identify the best-selling grocery categories.
- Study rating trends and item visibility impact on sales.
- Data-Driven Decision Making
- Provide insights into stocking strategies for different outlet sizes.
- Suggest improvements based on sales percentage contribution per outlet.

DATA CLEANING

- Cleaning the Item_Fat_Cleaning 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.Content field ensures data consistency and accuracy in analysis.
  - UPDATE `blinkit grocery 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;

- CHANGING Some of THE COLUMN NAME

- **.ALTER TABLE `blinkit grocery data` CHANGE `item_fat_content` `Item_fat_Content` VARCHAR(50);**
- **ALTER TABLE `blinkit grocery data` CHANGE `Outlet Establishment Year` `Outlet_Establishment_Year` int;**
- **ALTER TABLE `blinkit grocery data` CHANGE `Outlet Size` `Outlet_Size` text;**

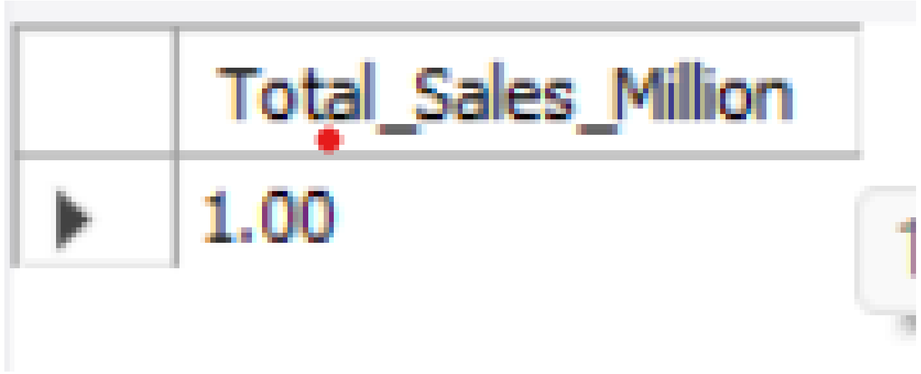
- verify all cleaning

SELECT * FROM `blinkit grocery data`;

1. TOTAL SALES:

```
SELECT CAST(SUM(`Total Sales`) / 1000000.0 AS DECIMAL(10,2)) AS Total_Sales_Million
FROM `blinkit grocery data`;
```

Output

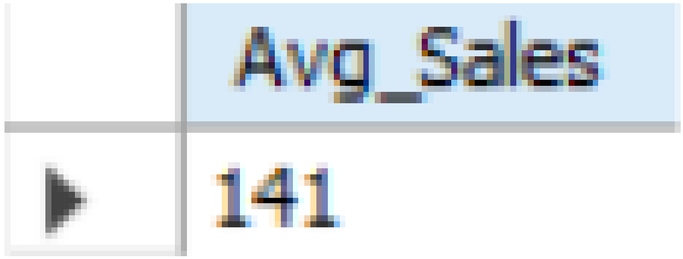


	Total_Sales_Million
▶	1.00

2. AVERAGE SALES:

```
SELECT CAST(AVG(`Total Sales`) AS SIGNED) AS Avg_Sales
FROM `blinkit grocery data`;
```

Output

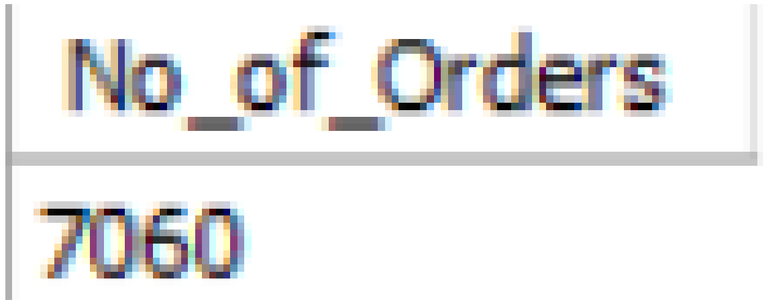


Avg_Sales
141

3. NUMBER OF ITEMS:

```
SELECT COUNT(*) AS No_of_Orders
FROM `blinkit grocery data`;
```

Output

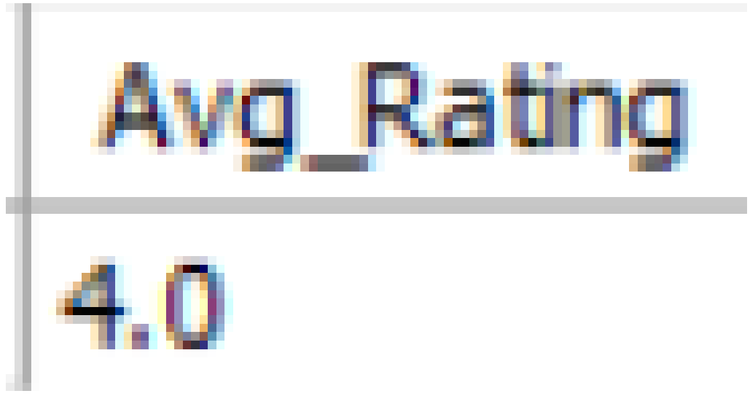


No_of_Orders
7060

4. AVERAGE RATING:

```
SELECT CAST(AVG(Rating) AS DECIMAL(10,1)) AS Avg_Rating
FROM `blinkit grocery data`;
```

Output



Avg_Rating
4.0

B. TOTAL SALES BY FAT CONTENT:

```
SELECT Item_Fat_Content, CAST(SUM(`Total Sales`) AS DECIMAL(10,2)) AS
 Total_Sales
 FROM `blinkit grocery data`
 GROUP BY Item_Fat_Content;
```

- Output

Item_Fat_Content	Total_Sales
Regular	352642.49
Low Fat	644516.73

C. TOTAL SALES BY ITEM TYPE:

```
SELECT `Item Type`,
      CAST(SUM(`Total Sales`) AS DECIMAL(10,2)) AS Total_Sales
FROM `blinkit grocery data`
GROUP BY `Item Type`
ORDER BY Total_Sales DESC;
```

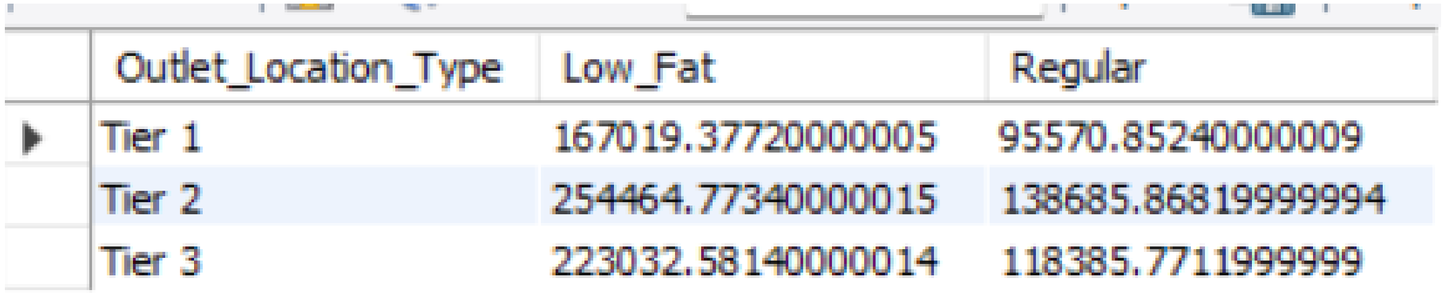
- Output

Item Type	Total_Sales
Fruits and Vegetables	147188.96
Snack Foods	144949.13
Household	113210.14
Frozen Foods	99961.88
Dairy	84526.49
Canned	75053.00
Baking Goods	67588.11
Health and Hygiene	56383.80
Soft Drinks	49294.67
Meat	47159.90
Breads	28663.18
Hard Drinks	25261.62
Starchy Foods	19199.98
Others	18624.60
Breakfast	12696.19
Seafood	7397.56

D. FAT CONTENT BY OUTLET FOR TOTAL SALES

```
SELECT
    Outlet_Location_Type,
    SUM(CASE WHEN Item_Fat_Content = 'Low Fat' THEN `Total Sales` ELSE 0 END) AS Low_Fat,
    SUM(CASE WHEN Item_Fat_Content = 'Regular' THEN `Total Sales` ELSE 0 END) AS Regular
FROM `blinkit grocery data`
GROUP BY Outlet_Location_Type
ORDER BY Outlet_Location_Type;
```

- Output



Outlet_Location_Type	Low_Fat	Regular
Tier 1	167019.37720000005	95570.85240000009
Tier 2	254464.77340000015	138685.86819999994
Tier 3	223032.58140000014	118385.771199999

E. TOTAL SALES BY OUTLET ESTABLISHMENT:

```
SELECT Outlet_Establishment_Year, CAST(SUM(`Total Sales`) AS DECIMAL(10,2)) AS Total_Sales
FROM `blinkit grocery data`
GROUP BY Outlet_Establishment_Year
ORDER BY Outlet_Establishment_Year;
```

- Output

Outlet_Establishment_Year	Total_Sales
2000	131809.02
2010	132113.37
2011	78131.56
2012	130476.86
2015	130942.78
2017	133103.91
2020	129103.96
2022	131477.77

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 / (SELECT SUM(`Total Sales`) FROM `blinkit grocery data`)) AS DECIMAL(10,2)) AS Sales_Percentage
FROM `blinkit grocery data`
GROUP BY Outlet_Size
ORDER BY Total_Sales DESC;
```

- Output

Outlet_Size	Total_Sales	Sales_Percentage
Medium	377181.05	37.83
Small	370986.59	37.20
High	248991.58	24.97

G. SALES BY OUTLET LOCATION:

```
SELECT Outlet_Location_Type, CAST(SUM(`Total Sales`) AS DECIMAL(10,2)) AS Total_Sales
FROM `blinkit grocery data`
GROUP BY Outlet_Location_Type
ORDER BY Total_Sales DESC;
```

- Output

Outlet_Location_Type	Total_Sales
Tier 2	393150.64
Tier 3	341418.35
Tier 1	262590.23

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 grocery data`
GROUP BY `Outlet Type`
ORDER BY Total_Sales DESC;
```

- Output

Outlet Type	Total_Sales	Avg_Sales	No_Of_Items	Avg_Rating	Item_Visibility
Supermarket Type1	787549.89	141	5577	3.95	0.06
Supermarket Type2	131477.77	142	928	3.95	0.06
Grocery Store	78131.56	141	555	3.97	0.10

I. TOP 10 BEST-SELLING ITEMS

```
SELECT `Item Identifier`, `Item Type`, CAST(SUM(`Total Sales`) AS DECIMAL(10,2)) AS Total_Sales
FROM `blinkit grocery data`
GROUP BY `Item Identifier`, `Item Type`
ORDER BY Total_Sales DESC
LIMIT 10;
```

- Output

Item Identifier	Item Type	Total_Sales
FDL58	Snack Foods	2111.65
FDP28	Frozen Foods	2087.85
FDB15	Dairy	1846.70
FDU12	Baking Goods	1844.40
FDF05	Frozen Foods	1841.84
FDR59	Breads	1832.92
FDA04	Frozen Foods	1812.27
FDF04	Frozen Foods	1806.31
FDT07	Fruits and Vegetables	1793.63
NCQ06	Household	1787.01

J. OUTLET PERFORMANCE CLASSIFICATION

```
SELECT
    `Outlet Identifier`,
    `Outlet Type`,
    CAST(SUM(`Total Sales`) AS DECIMAL(10,2)) AS Total_Sales,
    CASE
        WHEN SUM(`Total Sales`) > 500000 THEN 'High Performing'
        WHEN SUM(`Total Sales`) BETWEEN 250000 AND 500000 THEN 'Moderate Performing'
        ELSE 'Low Performing'
    END AS Performance_Category
FROM `blinkit grocery data`
GROUP BY `Outlet Identifier`, `Outlet Type`
ORDER BY Total_Sales DESC;
```

- Output

Outlet Identifier	Outlet Type	Total_Sales	Performance_Category
OUT035	Supermarket Type1	133103.91	Low Performing
OUT046	Supermarket Type1	132113.37	Low Performing
OUT013	Supermarket Type1	131809.02	Low Performing
OUT018	Supermarket Type2	131477.77	Low Performing
OUT045	Supermarket Type1	130942.78	Low Performing
OUT049	Supermarket Type1	130476.86	Low Performing
OUT017	Supermarket Type1	129103.96	Low Performing
OUT010	Grocery Store	78131.56	Low Performing

K. OUTLETS WITH HIGHEST CUSTOMER RATINGS

```
SELECT `Outlet Identifier`,`Outlet Type`, CAST(AVG(Rating) AS DECIMAL(10,2)) AS Avg_Rating
FROM `blinkit grocery data`
GROUP BY `Outlet Identifier`, `Outlet Type`
ORDER BY Avg_Rating DESC
LIMIT 5;
```

- Output

Outlet Identifier	Outlet Type	Avg_Rating
OUT049	Supermarket Type1	3.97
OUT017	Supermarket Type1	3.97
OUT010	Grocery Store	3.97
OUT045	Supermarket Type1	3.96
OUT018	Supermarket Type2	3.95

L. CLASSIFYING ITEMS BY WEIGHT AND ANALYZING THEIR TOTAL SALES

```
SELECT
    CASE
        WHEN `Item Weight` < 10 THEN 'Small Items'
        WHEN `Item Weight` BETWEEN 10 AND 20 THEN 'Medium Items'
        ELSE 'Large Items'
    END AS Item_Size,
    CAST(SUM(`Total Sales`) AS DECIMAL(10,2)) AS Total_Sales
FROM `blinkit grocery data`
GROUP BY
    CASE
        WHEN `Item Weight` < 10 THEN 'Small Items'
        WHEN `Item Weight` BETWEEN 10 AND 20 THEN 'Medium Items'
        ELSE 'Large Items'
    END
ORDER BY Total_Sales DESC;
```

OUTPUT

Item_Size	Total_Sales
Medium Items	598489.54
Small Items	329535.32
Large Items	69134.37

THANK YOU