

BLINKIT SALES ANALYSIS

This case study aims to analyze BlinkIT's grocery sales data to uncover key performance trends across products, outlet types, and customer behavior. The goal is to generate actionable insights for improving sales efficiency, shelf space utilization, and outlet-level strategy.

- See all the imported data:

```
SELECT * FROM blinkit;
```

Item_Fat_Content	Item_Identifier	Item_Type	Outlet_Establishment_Year	Outlet_Identifier	Outlet_Location_Type	Outlet_Size	Outlet_Type	Item_Visibility	Item_Weight	Sales	Rating
Regular	FDX32	Fruits and Vegetables	2012	OUT049	Tier 1	Medium	Supermarket Type1	0.100014	15.1	145.479	5
Low Fat	NCB42	Health and Hygiene	2022	OUT018	Tier 3	Medium	Supermarket Type2	0.00859605	11.8	115.349	5
Regular	FDR28	Frozen Foods	2016	OUT046	Tier 1	Small	Supermarket Type1	0.0258965	13.85	165.021	5
Regular	FDL50	Canned	2014	OUT018	Tier 3	High	Supermarket Type1	0.0422779	12.15	126.501	5
Low Fat	DR125	Soft Drinks	2015	OUT045	Tier 2	Small	Supermarket Type1	0.0339702	19.6	55.1614	5
Low Fat	FDS52	Frozen Foods	2020	OUT017	Tier 2	Small	Supermarket Type1	0.00550548	8.89	102.402	5
Low Fat	NCU05	Health and Hygiene	2011	OUT010	Tier 3	Small	Grocery Store	0.0983124	11.8	81.4616	5
Low Fat	NCD30	Household	2015	OUT045	Tier 2	Small	Supermarket Type1	0.0269037	19.7	96.0726	5
Low Fat	FDW20	Fruits and Vegetables	2014	OUT013	Tier 3	High	Supermarket Type1	0.0241293	20.75	124.173	5
Low Fat	FDX25	Canned	2018	OUT027	Tier 3	Medium	Supermarket Type3	0.101562	NULL	181.929	5
Low Fat	FDX21	Snack Foods	2018	OUT027	Tier 3	Medium	Supermarket Type3	0.0845546	NULL	109.891	5
Low Fat	NCU41	Health and Hygiene	2017	OUT035	Tier 2	Small	Supermarket Type1	0.052045	18.85	192.185	5
Low Fat	FDL20	Fruits and Vegetables	2022	OUT018	Tier 3	Medium	Supermarket Type2	0.128938	17.1	112.389	5
Low Fat	NCR54	Household	2014	OUT013	Tier 3	High	Supermarket Type1	0.0904668	16.35	195.211	5
Low Fat	FDH19	Meat	2018	OUT027	Tier 3	Medium	Supermarket Type3	0.0329282	NULL	173.174	5
Regular	FDB57	Fruits and Vegetables	2017	OUT035	Tier 2	Small	Supermarket Type1	0.0188015	20.25	222.177	5

- Data Cleaning:

```
UPDATE blinkit
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, we can check whether the data has been cleaned or not by the following query:

```
SELECT DISTINCT(item_fat_content) FROM blinkit;
```

Result Grid		
	Filter Row	
item_fat_content		
Regular		
Low Fat		

A) KPIs:

1. Total Sales:

```
SELECT CONCAT(CAST(SUM(Sales)/1000000 AS DECIMAL(10,2)),' M') as
Total_Sales_Millions from blinkit;
```

Result Grid		
	Filter Row	
Total_Sales_Millio...		
1.20 M		

2. Average Sales:

```
SELECT CAST(AVG(Sales) AS DECIMAL(10,0)) as Avg_Sales from blinkit;
```

Result Grid		
	Filter Row	
Avg_Sales		
141		

3. Number of Items:

```
SELECT COUNT(*) AS No_of_Items FROM blinkit;
```

Result Grid	
	No_of_Items
	8523

4. Average Rating:

```
SELECT CAST(AVG(Rating) AS DECIMAL(10,2)) AS Avg_Rating FROM blinkit;
```

Result Grid	
	Avg_Rating
	3.97

5. Total Sales, Average Sales, Number of Items and Average Rating by Item_Fat_Content:

```
SELECT Item_Fat_Content,
       CAST(SUM(Sales) AS DECIMAL(10,2)) AS Total_Sales,
       CAST(AVG(Sales) AS DECIMAL(10,1)) as Avg_Sales,
       COUNT(*) AS No_of_Items,
       CAST(AVG(Rating) AS DECIMAL(10,2)) AS Avg_Rating
  FROM blinkit
 GROUP BY Item_Fat_Content
 ORDER BY Total_Sales DESC;
```

Result Grid Filter Rows: Search Export:

	Item_Fat_Content	Total_Sales	Avg_Sales	No_of_Items	Avg_Rating	
	Low Fat	776319.69	140.7	5517	3.97	
	Regular	425361.80	141.5	3006	3.97	

6. Total Sales, Average Sales, Number of Items and Average Rating by Item_Type:

```
SELECT Item_Type,
       CAST(SUM(Sales) AS DECIMAL(10,2)) AS Total_Sales,
       CAST(AVG(Sales) AS DECIMAL(10,1)) as Avg_Sales,
       COUNT(*) AS No_of_Items,
       CAST(AVG(Rating) AS DECIMAL(10,2)) AS Avg_Rating
  FROM blinkit
 GROUP BY Item_Type
 ORDER BY Total_Sales DESC;
```

Result Grid Filter Rows: Search Export:

	Item_Type	Total_Sales	Avg_Sales	No_of_Items	Avg_Rating	
	Fruits and Vegetables	178124.08	144.6	1232	3.96	
	Snack Foods	175433.92	146.2	1200	3.95	
	Household	135976.53	149.4	910	4.00	
	Frozen Foods	118558.88	138.5	856	3.97	
	Dairy	101276.46	148.5	682	3.97	
	Canned	90706.73	139.8	649	3.99	
	Baking Goods	81894.74	126.4	648	3.98	
	Health and Hygiene	68025.84	130.8	520	3.99	
	Meat	59449.86	139.9	425	4.02	
	Soft Drinks	58514.17	131.5	445	3.92	
	Breads	35379.12	141.0	251	3.88	
	Hard Drinks	29334.68	137.1	214	3.91	
	Others	22451.89	132.9	169	3.95	
	Starchy Foods	21880.03	147.8	148	3.92	
	Breakfast	15596.70	141.8	110	3.93	
	Seafood	9077.87	141.8	64	3.96	

7. Total Sales by **Outlet_Location_Type** and **Item_Fat_Content**;

```
SELECT
    Outlet_location_type,
    ROUND(SUM(CASE WHEN item_fat_content = 'Low Fat' THEN sales ELSE
    0 END), 2) AS `Low Fat`,
    ROUND(SUM(CASE WHEN item_fat_content = 'Regular' THEN sales ELSE
    0 END), 2) AS `Regular`
FROM blinkit
GROUP BY outlet_location_type
ORDER BY outlet_location_type;
```

outlet_location_t...	Low Fat	Regular
Tier 1	215047.91	121349.9
Tier 2	254464.78	138685.87
Tier 3	306807	165326.04

8. Total Sales, Avg_Sales, No_of_items and Avg_Rating by **Outlet_Establishment_Year**:

```
SELECT Outlet_Establishment_Year,
    ROUND(SUM(Sales),2) AS Total_Sales
    ROUND(AVG(Sales),2) as Avg_Sales,
    COUNT(*) AS No_of_Items,
    ROUND(AVG(Rating),2) AS Avg_Rating
FROM blinkit
GROUP BY Outlet_Establishment_Year
ORDER BY Outlet_Establishment_Year;
```

Result Grid Filter Rows: Search Export:

	Outlet_Establishment_Year	Total_Sales	Avg_Sales	No_of_Items	Avg_Rating	
	2011	78131.57	140.78	555	3.98	
	2012	130476.86	140.3	930	3.99	
	2014	131809.02	141.43	932	3.95	
	2015	130942.78	140.95	929	3.96	
	2016	132113.37	142.06	930	3.96	
	2017	133103.91	143.12	930	3.94	
	2018	204522.26	139.8	1463	3.97	
	2020	129103.96	139.42	926	3.98	
	2022	131477.78	141.68	928	3.97	

9. Total Sales, Sales %, Avg_Sales, No_of_items and Avg_Rating by **Outlet_Size**:

```

SELECT
    Outlet_Size,
    ROUND(SUM(Sales),2) AS Total_Sales,
    ROUND(SUM(Sales)/ (SELECT SUM(Sales) FROM blinkit)*100,2) AS 'Sales %',
    ROUND(AVG(Sales),2) as Avg_Sales,
    COUNT(*) AS No_of_Items,
    ROUND(AVG(Rating),2) AS Avg_Rating
FROM blinkit
GROUP BY Outlet_Size;
  
```

Result Grid Filter Rows: Search Export:

	Outlet_Size	Total_Sales	Sales %	Avg_Sales	No_of_Items	Avg_Rating	
	Medium	507895.74	42.27	139.88	3631	3.98	
	Small	444794.17	37.01	141.7	3139	3.96	
	High	248991.59	20.72	142.04	1753	3.95	

10. Total Sales, Sales %, Avg_Sales, No_of_items and Avg_Rating by **Outlet_Location_Type**:

```
SELECT
    Outlet_Location_Type,
    ROUND(SUM(Sales),2) AS Total_Sales,
    ROUND(SUM(Sales)/ (SELECT SUM(Sales) FROM blinkit)*100,2) AS 'Sales %',
    ROUND(AVG(Sales),2) as Avg_Sales,
    COUNT(*) AS No_of_Items,
    ROUND(AVG(Rating),2) AS Avg_Rating
FROM blinkit
GROUP BY Outlet_Location_Type
ORDER BY Outlet_Location_Type ASC;
```

Result Grid					
Outlet_Location_Type	Total_Sales	Sales %	Avg_Sales	No_of_Items	Avg_Rating
Tier 1	336397.81	27.99	140.87	2388	3.98
Tier 2	393150.65	32.72	141.17	2785	3.96
Tier 3	472133.03	39.29	140.94	3350	3.96

11. Total Sales, Sales %, Avg_Sales, No_of_items and Avg_Rating by **Outlet_Type**:

```
SELECT
    Outlet_Type,
    ROUND(SUM(Sales),2) AS Total_Sales,
    ROUND(SUM(Sales)/ (SELECT SUM(Sales) FROM blinkit)*100,2) AS 'Sales %',
    ROUND(AVG(Sales),2) as Avg_Sales,
    COUNT(*) AS No_of_Items,
    ROUND(AVG(Rating),2) AS Avg_Rating
FROM blinkit
GROUP BY Outlet_Type
ORDER BY Total_Sales ASC;
```

Result Grid Filter Rows: Search Export:

Outlet_Type	Total_Sales	Sales %	Avg_Sales	No_of_Items	Avg_Rating
Supermarket Type3	130714.67	10.88	139.8	935	3.95
Supermarket Type2	131477.78	10.94	141.68	928	3.97
Grocery Store	151939.15	12.64	140.29	1083	3.99
Supermarket Type1	787549.89	65.54	141.21	5577	3.96

12. Which Items_Type have the highest and lowest visibility and Sales_per_Visibility:

```

SELECT
    Item_Type,
    ROUND(AVG(Item_Visibility),3) AS Avg_Visibility,
    ROUND(SUM(Sales),2) AS Total_Sales,
    ROUND(SUM(Sales)/SUM(Item_Visibility),2) AS Sales_per_Visibility
FROM blinkit
GROUP BY Item_Type
ORDER BY Total_Sales DESC;

```

Result Grid Filter Rows: Search Export:

Item_Type	Avg_Visibility	Total_Sales	Sales_per_Visibili...
Fruits and Vegetables	0.069	178124.08	2110.27
Snack Foods	0.067	175433.92	2186.9
Household	0.061	135976.53	2436.71
Frozen Foods	0.066	118558.88	2109.88
Dairy	0.072	101276.46	2050.32
Canned	0.068	90706.73	2051.45
Baking Goods	0.069	81894.74	1827.12
Health and Hygiene	0.055	68025.84	2369.22
Meat	0.062	59449.86	2245.88
Soft Drinks	0.064	58514.17	2055.46
Breads	0.066	35379.12	2127.42
Hard Drinks	0.065	29334.68	2110.76
Others	0.06	22451.89	2205.33
Starchy Foods	0.068	21880.03	2188.13
Breakfast	0.086	15596.7	1654.03
Seafood	0.075	9077.87	1891.83

What do we infer from the above metrics?

- **High visibility + low sales** → overexposed items (inefficient)
- **Low visibility + high sales** → underexposed hidden gems
- **High sales per visibility** → best use of space (promote more!)
- **Low sales per visibility** → maybe reduce shelf space or promotions

13. Relation between **Item_Weight** and **Sales**:

For checking the relation between item_weight and sales, we will add a new column Weight_Category which will be used to check the sales by item weight category.

- **Light Weight** - items with weight less than 10
- **Normal Weight** - items with weight greater than 10 and less than 15
- **Heavy Weight** - items with weight greater than 15

Adding a new column:

```
ALTER TABLE blinkit
ADD COLUMN Weight_Category varchar(20);
```

Updating with categories:

```
UPDATE blinkit
SET Weight_Category = CASE
    WHEN Item_Weight<10 THEN 'Light Weight'
    WHEN Item_Weight>15 THEN 'Heavy Weight'
    ELSE 'Normal Weight'
END;
```

Final query:

```
SELECT
    Weight_Category,
    ROUND(SUM(sales), 2) AS Total_Sales,
    COUNT(*) AS item_count
FROM blinkit
GROUP BY Weight_Category
ORDER BY Weight_Category;
```

Weight_Catego...	Total_Sales	item_count
Heavy Weight	372203.34	2613
Light Weight	329535.33	2377
Normal Weight	499942.83	3534

We will look for which weight group items generate more sales revenue.

- Normal weight items tend to generate more revenue.

14. Underperforming items: items with high visibility but low sales.

```
SELECT
    Item_Identifier,
    Item_Type,
    ROUND( Item_Visibility,4) as Visibility,
    ROUND(Sales,2) as Total_Sales
FROM blinkit
WHERE
    Item_Visibility > (SELECT AVG(Item_Visibility) FROM blinkit)
    &
    Sales < ( SELECT AVG(Sales) FROM blinkit)
ORDER BY Item_Visibility DESC;
```

Item_Identifier	Item_Type	Visibility	Total_Sales
FDU13	Canned	0.3284	146.04
NCZ18	Household	0.3258	252.77
FDJ56	Fruits and Vegetables	0.3211	100.77
FDT24	Baking Goods	0.3111	79.23
NCE31	Household	0.3094	33.22
FDG32	Fruits and Vegetables	0.3081	222.08
DRF01	Soft Drinks	0.3065	147.31
FDI32	Fruits and Vegetables	0.3053	116.68
FDS12	Baking Goods	0.3049	125.44
FDR14	Dairy	0.3047	54.73
FDY21	Snack Foods	0.3037	196.01
FDC34	Snack Foods	0.3025	155.5
FDY15	Dairy	0.2991	157.86

'FDU13', 'Canned', '0.3284', '146.04'

'NCZ18', 'Household', '0.3258', '252.77'

The above items have a high visibility rate but don't get converted to sales.

15. Top 10 items contributing total sales:

```
SELECT
    Item_Identifier,
    Item_Type,
    ROUND(SUM(Sales),2) AS Total_Sales
FROM blinkit
GROUP BY Item_Identifier,Item_Type
ORDER BY Total_Sales DESC
LIMIT 10;
```

Item_Identifier	Item_Type	Total_Sales
FDU12	Baking Goods	2371.01
FDT07	Fruits and Vegetables	2306.9
NCQ06	Household	2294.71
FDL58	Snack Foods	2111.65
NCB31	Household	2104.73
FDX31	Fruits and Vegetables	2104.46
FDF05	Frozen Foods	2103.13
FDR59	Breads	2096.58
FDP28	Frozen Foods	2087.85
FDA04	Frozen Foods	2072.07

16. Which outlet_type performs the best for specific item_type:

```
WITH ranking_table AS(
    SELECT
        item_type,
        outlet_type,
        ROUND(SUM(sales), 2) AS total_sales,
        RANK() OVER (PARTITION BY item_type ORDER BY SUM(sales) DESC) AS
sales_rank
    FROM blinkit
    GROUP BY item_type, outlet_type)
```

```
)
SELECT * FROM ranking_table
WHERE sales_rank = 1;
```

Result Grid Filter Rows: Search Export:

item_type	outlet_type	total_sales	sales_rank
Baking Goods	Supermarket Type1	53665.46	1
Breads	Supermarket Type1	22490.81	1
Breakfast	Supermarket Type1	9632.65	1
Canned	Supermarket Type1	59649.31	1
Dairy	Supermarket Type1	67179.69	1
Frozen Foods	Supermarket Type1	79348.38	1
Fruits and Vegetables	Supermarket Type1	117431.99	1
Hard Drinks	Supermarket Type1	19610.76	1
Health and Hygiene	Supermarket Type1	44050.75	1
Household	Supermarket Type1	89143.41	1
Meat	Supermarket Type1	36169.76	1
Others	Supermarket Type1	14826.67	1
Seafood	Supermarket Type1	5681.52	1
Snack Foods	Supermarket Type1	114296.13	1
Soft Drinks	Supermarket Type1	39002.05	1
Starchy Foods	Supermarket Type1	15370.55	1

- Supermarket Type 1 is the best performing outlet_type for every item_type.

17. Which outlet_location_type has the highest average rating:

```

SELECT
    Outlet_Location_Type,
    ROUND(AVG(Rating),2) AS Avg_Rating,
    COUNT(*) AS Number_of_items
FROM blinkit
GROUP BY Outlet_Location_Type
ORDER BY Avg_Rating DESC;
```

Result Grid Filter Rows: Search Exit

Outlet_Location_Type	Avg_Rating	Number_of_it...
Tier 1	3.98	2388
Tier 3	3.96	3350
Tier 2	3.96	2785

- Tier 1 has the highest average rating.

18. Sales performance by outlet age group:

We will create a new column Age_group which will categorise Outlet_Establishment_Year into:

- Old : Outlet_Establishment_Year before 2013
- Medium : Outlet_Establishment_Year between 2013 and 2018
- New : Outlet_Establishment_Year after 2018

```
ALTER TABLE blinkit
ADD COLUMN Age_Category VARCHAR(20);
```

Updating with categories:

```
UPDATE blinkit
SET Age_Category = CASE
    WHEN Outlet_Establishment_Year > 2018 THEN 'New'
    WHEN Outlet_Establishment_Year < 2013 THEN 'Old'
    ELSE 'Medium'
END;
```

Final Query:

```
SELECT
    Age_category,
    ROUND(SUM(Sales),2) AS Total_sales
FROM blinkit
GROUP BY Age_category
ORDER BY Total_sales DESC ;
```

Result Grid Filter Rows:

Age_category	Total_sales
Medium	732491.33
New	260581.74
Old	208608.43

- Outlets established in medium age group produce the most sales.

19. Difference in sales between high rated and low rated items:

We will create a new column rating_category which categorize the rating of the items:

- 4-5 : ratings between 4 to 5.
- 3-4 : ratings between 3 to 4.
- 1-3 : ratings between 1-3.

```
ALTER TABLE blinkit
ADD COLUMN rating_category varchar(20);
```

Updating with categories:

```
UPDATE blinkit
SET rating_category = CASE
    WHEN rating>=4 THEN '4-5'
    WHEN rating>=3 AND rating<4 THEN '3-4'
    ELSE '1-3'
END;
```

Final query:

```
SELECT
    rating_category,
    ROUND(SUM(Sales),2) AS Total_sales
FROM blinkit
GROUP BY rating_category
ORDER BY total_sales DESC;
```

rating_category	Total_sales
4-5	919567.58
3-4	223110.61
1-3	59003.3

20. What is the Sales per unit visibility:

- Measures how much sales is generated for every unit of visibility.

SELECT

```
item_type,
ROUND(SUM(sales),2) AS Total_Sales,
ROUND(SUM(Item_Visibility),2) AS Total_visibility,
ROUND((SUM(sales)/SUM(Item_Visibility)),2) AS sales_per_visiblty
FROM blinkit
GROUP BY item_type
ORDER BY sales_per_visiblty DESC;
```

item_type	Total_Sales	Total_visibility	sales_per_visiblty
Fruits and Vegetables	178124.08	84.41	2110.27
Snack Foods	175433.92	80.22	2186.9
Frozen Foods	118558.88	56.19	2109.88
Household	135976.53	55.8	2436.71
Dairy	101276.46	49.4	2050.32
Baking Goods	81894.74	44.82	1827.12
Canned	90706.73	44.22	2051.45
Health and Hygiene	68025.84	28.71	2369.22
Soft Drinks	58514.17	28.47	2055.46
Meat	59449.86	26.47	2245.88
Breads	35379.12	16.63	2127.42
Hard Drinks	29334.68	13.9	2110.76
Others	22451.89	10.18	2205.33
Starchy Foods	21880.03	10	2188.13
Breakfast	15596.7	9.43	1654.03
Seafood	9077.87	4.8	1891.83

21. What is the Sales per unit weight:

- Measures how much sales is generated for every unit of weight.

SELECT

```
item_type,  
ROUND(SUM(sales),2) AS Total_Sales,  
ROUND(SUM(Item_Weight),2) AS Total_weight,  
ROUND((SUM(sales)/SUM(Item_Weight)),2) AS sales_per_weight  
FROM blinkit  
GROUP BY item_type  
ORDER BY sales_per_weight DESC;
```

item_type	Total_Sales	Total_weight	sales_per_weight
Breads	35379.12	2314.78	15.28
Seafood	9077.87	640.2	14.18
Hard Drinks	29334.68	2086.26	14.06
Meat	59449.86	4319.45	13.76
Breakfast	15596.7	1136.37	13.73
Canned	90706.73	6632.78	13.68
Snack Foods	175433.92	12832.03	13.67
Household	135976.53	10159.02	13.38
Dairy	101276.46	7599.16	13.33
Fruits and Vegetables	178124.08	13476.04	13.22
Soft Drinks	58514.17	4430.95	13.21
Frozen Foods	118558.88	9238.55	12.83
Baking Goods	81894.74	6580.53	12.45
Starchy Foods	21880.03	1779.8	12.29
Health and Hygiene	68025.84	5651.2	12.04
Others	22451.89	1897.9	11.83

22. Items with unusually high or low visibility that may skew performance:

- Spotting the outliers is crucial for misplaced promotions or data errors.
- We will define outliers as items outside the range:-
 $\text{avg} + 2 * \text{std_dev}$ and $\text{avg} - 2 * \text{std_dev}$

```

WITH stats AS (
    SELECT
        AVG(item_visibility) AS avg_vis,
        STDDEV(item_visibility) AS std_vis
    FROM blinkit
)
SELECT
    item_identifier,
    item_type,
    item_visibility,
    sales,
    Rating
FROM blinkit, stats
WHERE item_visibility > (avg_vis + 2 * std_vis)
    OR item_visibility < (avg_vis - 2 * std_vis)
ORDER BY item_visibility DESC;

```

	item_identif...	item_type	item_visibil...	sales	rating	
	DRI49	Soft Drinks	0.183473	82.0276	4.3	
	FDD52	Dairy	0.183295	111.657	4.1	
	FDD52	Dairy	0.18326	108.557	4	
	FDD52	Dairy	0.183142	110.357	4	
	NCJ31	Others	0.182938	243.02	4	
	NCJ31	Others	0.182654	243.02	4	
	FDJ56	Fruits and Vegetables	0.182515	98.77	5	
	NCJ31	Others	0.182502	239.22	5	
	NCQ29	Health and Hygiene	0.182494	258.828	4.3	
	FDD52	Dairy	0.182407	109.157	4	
	FDL02	Canned	0.182237	107.162	5	
	FDZ59	Baking Goods	0.182128	165.65	4	
	FDO57	Snack Foods	0.181959	161.058	4.5	
	NCJ31	Others	0.181769	240.62	4	
	FDS57	Snack Foods	0.181114	141.647	4.5	
	NCO06	Household	0.180821	35.9558	4.1	
	FDE34	Snack Foods	0.180588	182.263	4	
	DRI37	Soft Drinks	0.180097	57.3904	4	
	FDO27	Meat	0.179807	95.2752	3.6	