

[RAW Data Link](#)

Data Dictionary

Column Name	Description
Item_Identifier	Unique product code for each item.
Item_Weight	Weight of the item (in kg).
Item_Fat_Content	Indicates the fat level of the item (Low Fat, Regular).
Item_Visibility	Percentage visibility of an item in a store.
Item_Type	Category to which an item belongs (e.g., Dairy, Snacks).
Item_MRP	Maximum Retail Price of the item.
Outlet_Identifier	Unique code for each outlet.
Outlet_Establishment_Year	Year the outlet was established.
Outlet_Size	Size of the outlet (Small, Medium, High).
Outlet_Location_Type	Type of city (Tier 1, Tier 2, Tier 3).
Outlet_Type	Type of store (Supermarket Type1, Type2, etc.).
Item_Outlet_Sales	Total sales of the item in the outlet (target variable).

Data Cleaning Recommendations

Based on the data review, here are the cleaning steps required to ensure the dataset is ready for analysis and visualization in the dashboard:

COUNTBLANK() in EXCEL

A	B	C	D	E	F	G	H	I	J	K	L
Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Identifier	Outlet_Establishment_Year	Outlet_Size	Outlet_Location_Type	Outlet_Type	Item_Outlet_Sales
0	1463	0	0	0	0	0	0	2410	0	0	0

Select \*

From [BlinkIT Grocery Project]

Results												
	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Identifier	Outlet_Establishment_Year	Outlet_Size	Outlet_Location_Type	Outlet_Type	Item_Outlet_Sales
1	FDA15	9.3	Low Fat	0.016047301	Dairy	249.8092	OUT049	1999	Medium	Tier 1	Supermarket Type1	3735.138
2	DRC01	5.92	Regular	0.019278216	Soft Drinks	48.2692	OUT018	2009	Medium	Tier 3	Supermarket Type2	443.4228
3	FDN15	17.5	Low Fat	0.016760075	Meat	141.618	OUT049	1999	Medium	Tier 1	Supermarket Type1	2097.27
4	FDX07	19.2	Regular	0	Fruits and Vegetables	182.095	OUT010	1998	NULL	Tier 3	Grocery Store	732.38
5	NCD19	8.93	Low Fat	0	Household	53.8614	OUT013	1987	High	Tier 3	Supermarket Type1	994.7052
6	FDP36	10.395	Regular	0	Baking Goods	51.4008	OUT018	2009	Medium	Tier 3	Supermarket Type2	556.6088
7	FDO10	13.65	Regular	0.012741089	Snack Foods	57.6588	OUT013	1987	High	Tier 3	Supermarket Type1	343.5528
8	FDP10	NULL	Low Fat	0.127469857	Snack Foods	107.7622	OUT027	1985	Medium	Tier 3	Supermarket Type3	4022.7636
9	FDH17	16.2	Regular	0.016687114	Frozen Foods	96.9726	OUT045	2002	NULL	Tier 2	Supermarket Type1	1076.5986
10	FDU28	19.2	Regular	0.09444959	Frozen Foods	187.8214	OUT017	2007	NULL	Tier 2	Supermarket Type1	4710.535
11	FDY07	11.8	Low Fat	0	Fruits and Vegetables	45.5402	OUT049	1999	Medium	Tier 1	Supermarket Type1	1516.0266
12	FDA03	18.5	Regular	0.045463773	Dairy	144.1102	OUT046	1997	Small	Tier 1	Supermarket Type1	2187.153
13	FDX32	15.1	Regular	0.1000135	Fruits and Vegetables	145.4786	OUT049	1999	Medium	Tier 1	Supermarket Type1	1589.2646
14	FDS46	17.6	Regular	0.047257328	Snack Foods	119.6782	OUT046	1997	Small	Tier 1	Supermarket Type1	2145.2076
15	FDF32	16.35	Low Fat	0.0680243	Fruits and Vegetables	196.4426	OUT013	1987	High	Tier 3	Supermarket Type1	1977.426
16	FDP49	9	Regular	0.069088961	Breakfast	56.3614	OUT046	1997	Small	Tier 1	Supermarket Type1	1547.3192
17	NCR42	11.8	Low Fat	0.009596051	Health and Hygiene	115.2492	OUT018	2009	Medium	Tier 2	Supermarket Type2	1621.8888
✔ Query executed successfully.												

--Duplicate Table with Structure and Data  
SELECT \*  
INTO blinkit\_grocery\_cpy  
FROM [BlinkIT Grocery Project];

Task:

1. Handle Missing Values

Item\_Weight:

Action:

```
SELECT
    COUNT(*) AS Total_Rows,
    SUM(CASE WHEN Item_Weight IS NULL THEN 1 ELSE 0 END) AS Missing_Item_Weight,
    SUM(CASE WHEN Outlet_Size IS NULL THEN 1 ELSE 0 END) AS Missing_Outlet_Size
FROM
    blinkit_grocery_cpy;
```

<div><div><div></div></div>Results<div><div></div><div></div></div>Messages</div>			
	Total_Rows	Missing_Item_Weight	Missing_Outlet_Size
1	8523	1463	2410

---Create a Temporary Table with Average Weights:

```
SELECT
    Item_Identifier,
    AVG(Item_Weight) AS Avg_Item_Weight
INTO Temp_Item_Weights
FROM blinkit_grocery_cpy
WHERE Item_Weight IS NOT NULL
GROUP BY Item_Identifier;
```

---Update Missing Item\_Weight Values Using a JOIN

```
UPDATE blinkit_grocery_cpy
SET blinkit_grocery_cpy.Item_Weight = t.Avg_Item_Weight
FROM blinkit_grocery_cpy
JOIN Temp_Item_Weights t
ON blinkit_grocery_cpy.Item_Identifier = t.Item_Identifier
WHERE blinkit_grocery_cpy.Item_Weight IS NULL;
Check NULL
```

<div><div><div></div></div>Results<div><div></div><div></div></div>Messages</div>			
	Total_Rows	Missing_Item_Weight	Missing_Outlet_Size
1	8523	4	2410

```
-- Find the overall mode
SELECT TOP 1 Item_Weight, COUNT(*) AS Frequency
FROM blinkit_grocery_cpy
WHERE Item_Weight IS NOT NULL
GROUP BY Item_Weight
ORDER BY Frequency DESC;
Result 12.15 Frequent 103
```

-- Update missing weights

```
UPDATE blinkit_grocery_cpy
SET Item_Weight = 12.15
WHERE Item_Weight IS NULL;
```

<div><div><div></div></div>Results<div><div></div><div></div></div>Messages</div>			
	Total_Rows	Missing_Item_Weight	Missing_Outlet_Size
1	8523	0	2410

Outlet\_Size:

Action: Impute missing values based on Outlet\_Type and Outlet\_Location\_Type, as these attributes likely correlate with outlet size.

---Check how Outlet\_Size relates to Outlet\_Type and Outlet\_Location\_Type

```
SELECT
    Outlet_Type,
    Outlet_Location_Type,
    Outlet_Size,
    COUNT(*) AS Frequency
FROM blinkit_grocery_cpy
GROUP BY Outlet_Type, Outlet_Location_Type, Outlet_Size
ORDER BY Outlet_Type, Outlet_Location_Type, Frequency DESC;
```

<div><div><div></div></div>Results<div><div></div><div></div></div>Messages</div>				
	Outlet_Type	Outlet_Location_Type	Outlet_Size	Frequency
1	Grocery Store	Tier 1	Small	528
2	Grocery Store	Tier 3	NULL	555
3	Supermarket Type1	Tier 1	Medium	930
4	Supermarket Type1	Tier 1	Small	930
5	Supermarket Type1	Tier 2	NULL	1855
6	Supermarket Type1	Tier 2	Small	930
7	Supermarket Type1	Tier 3	High	932
8	Supermarket Type2	Tier 3	Medium	928
9	Supermarket Type3	Tier 3	Medium	935

---Find the most frequent Outlet\_Size for each combination of Outlet\_Location\_Type and Outlet\_Type

```
WITH ModeSize AS (  
  SELECT  
    Outlet_Location_Type,  
    Outlet_Type,  
    Outlet_Size,  
    COUNT(*) AS Frequency,  
    ROW_NUMBER() OVER (PARTITION BY Outlet_Location_Type, Outlet_Type ORDER BY COUNT(*) DESC) AS Rank  
  FROM blinkit_grocery_cpy  
  WHERE Outlet_Size IS NOT NULL  
  GROUP BY Outlet_Location_Type, Outlet_Type, Outlet_Size  
)  
SELECT  
  Outlet_Location_Type,  
  Outlet_Type,  
  Outlet_Size AS ImputedSize  
FROM ModeSize  
WHERE Rank = 1;
```

	Outlet_Location_Type	Outlet_Type	ImputedSize
1	Tier 1	Grocery Store	Small
2	Tier 1	Supermarket Type1	Small
3	Tier 2	Supermarket Type1	Small
4	Tier 3	Supermarket Type1	High
5	Tier 3	Supermarket Type2	Medium
6	Tier 3	Supermarket Type3	Medium

---Create a Temporary Table to Store the Modes

```
CREATE TABLE #ModeSize (  
  Outlet_Location_Type NVARCHAR(50),  
  Outlet_Type NVARCHAR(50),  
  ImputedSize NVARCHAR(50)  
);  
-- Insert mode values into the temporary table  
INSERT INTO #ModeSize (Outlet_Location_Type, Outlet_Type, ImputedSize)  
SELECT  
  Outlet_Location_Type,  
  Outlet_Type,  
  Outlet_Size AS ImputedSize  
FROM (  
  SELECT  
    Outlet_Location_Type,  
    Outlet_Type,  
    Outlet_Size,  
    COUNT(*) AS Frequency,  
    ROW_NUMBER() OVER (PARTITION BY Outlet_Location_Type, Outlet_Type ORDER BY COUNT(*) DESC) AS Rank  
  FROM blinkit_grocery_cpy  
  WHERE Outlet_Size IS NOT NULL  
  GROUP BY Outlet_Location_Type, Outlet_Type, Outlet_Size  
) AS RankedData  
WHERE Rank = 1;
```

select \*  
from #ModeSize

	Outlet_Location_Type	Outlet_Type	ImputedSize
1	Tier 1	Grocery Store	Small
2	Tier 1	Supermarket Type1	Small
3	Tier 2	Supermarket Type1	Small
4	Tier 3	Supermarket Type1	High
5	Tier 3	Supermarket Type2	Medium
6	Tier 3	Supermarket Type3	Medium

```
-- Update the Outlet_Size column based on the mode stored in the temporary table
UPDATE blinkit_grocery_cpy
SET Outlet_Size = (
  SELECT ImputedSize
  FROM #ModeSize
  WHERE #ModeSize.Outlet_Location_Type = blinkit_grocery_cpy.Outlet_Location_Type
  AND #ModeSize.Outlet_Type = blinkit_grocery_cpy.Outlet_Type
)
WHERE Outlet_Size IS NULL;
--Verify Null Values After Update
SELECT
  Outlet_Size,
  SUM(CASE WHEN Outlet_Size IS NULL THEN 1 ELSE 0 END) AS NullCount
FROM blinkit_grocery_cpy
GROUP BY Outlet_Size
```

	Outlet_Size	NullCount
1	NULL	555
2	High	0
3	Medium	0
4	Small	0

---Manually Update Remaining NULL Values: Use an UPDATE statement to fill in the remaining NULL values with the most appropriate value based on the business context or overall dataset trends.

```
UPDATE blinkit_grocery_cpy
SET Outlet_Size = 'Small' -- Replace 'Small' with the desired value
WHERE Outlet_Type = 'Grocery Store'
  AND Outlet_Location_Type = 'Tier 3'
  AND Outlet_Size IS NULL;
```

	Outlet_Size	NullCount
1	High	0
2	Medium	0
3	Small	0

## 2. Standardize Categorical Data

Item\_Fat\_Content:  
Inconsistent labeling (Low Fat, low fat, LF).  
Action: Standardize values to Low Fat and Regular.

---Steps to Standardize Item\_Fat\_Content(Low Fat,Regular)  
SELECT DISTINCT Item\_Fat\_Content, COUNT(\*) AS Frequency  
FROM blinkit\_grocery\_cpy  
GROUP BY Item\_Fat\_Content;

	Item_Fat_Content	Frequency
1	Low Fat	5201
2	reg	117
3	LF	316
4	Regular	2889

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

	Item_Fat_Content	Frequency
1	Low Fat	5517
2	Regular	3006

3. Remove Outliers

Item\_Visibility:  
Minimum visibility is 0, which is likely unrealistic.  
Action:

---Calculate Mean of Non-Zero  
SELECT COUNT(\*) AS ZeroCount  
FROM blinkit\_grocery\_cpy  
WHERE Item\_Visibility = 0;

	ZeroCount
1	526

SELECT AVG(Item\_Visibility) AS MeanVisibility  
FROM blinkit\_grocery\_cpy  
WHERE Item\_Visibility > 0;

	Messages
	Msg 8117, Level 16, State 1, Line 140 Operand data type nvarchar is invalid for avg operator.

SELECT COLUMN\_NAME, DATA\_TYPE  
FROM INFORMATION\_SCHEMA.COLUMNS  
WHERE TABLE\_NAME = 'blinkit\_grocery\_cpy' AND COLUMN\_NAME = 'Item\_Visibility';

	Results	Messages
	COLUMN_NAME	DATA_TYPE
1	Item_Visibility	nvarchar

ALTER TABLE blinkit\_grocery\_cpy  
ALTER COLUMN Item\_Visibility FLOAT;  
Repeat Step

	Results	Messages
	MeanVisibility	
1	0.0704818408506942	

UPDATE blinkit\_grocery\_cpy  
SET Item\_Visibility = 0.0704818408547522  
WHERE Item\_Visibility = 0;

	Results	Messages
	ZeroCount	
1	0	

4. Create Derived Fields

Add calculated fields for meaningful analysis:  
Outlet\_Age: Calculate the age of each outlet using the current year (e.g., 2024 - Outlet\_Establishment\_Year).  
Action:

---Derived Fields  
-- Add Outlet\_Age Column  
ALTER TABLE blinkit\_grocery\_cpy  
ADD Outlet\_Age INT;  
Select Outlet\_Age  
From blinkit\_grocery\_cpy

	Results	Messages
	Outlet_Age	
1	NULL	
2	NULL	

-- Calculate Outlet\_Age  
UPDATE blinkit\_grocery\_cpy  
SET Outlet\_Age = YEAR(GETDATE()) - Outlet\_Establishment\_Year;

	Results	Messages
	Outlet_Age	
1	26	
2	16	
3	26	
4	27	

**Sales\_Per\_Item: Divide Item\_Outlet\_Sales by Item\_MRP to calculate relative performance.**

```
-- Add Sales_Per_Item column
ALTER TABLE blinkit_grocery_cpy
ADD Sales_Per_Item FLOAT;
-- Calculate Sales_Per_Item
UPDATE blinkit_grocery_cpy
SET Sales_Per_Item = Item_Outlet_Sales / Item_MRP;
```

**---Verify the Update**

```
Select Sales_Per_Item
From blinkit_grocery_cpy
```

	Sales_Per_Item
1	14.9519633384199
2	9.18645430212226
3	14.809346269542
4	4.02196655591861
5	18.4678675266517
6	10.8287964389659
7	5.95837582467897

### 5. Normalize Numerical Data

Item\_MRP:  
Wide range of values; consider creating price tiers (e.g., Low, Medium, High).  
Action:

```
---Derived Fields
ALTER TABLE blinkit_grocery_cpy
ADD Price_Tier NVARCHAR(20);
---Calculate Quartiles
---First, create variables or temporary values for Q1 and Q3 using PERCENTILE_CONT.
-- Step 1: Declare Variables
DECLARE @Q1 FLOAT, @Q3 FLOAT;
-- Calculate and assign Quartiles to variables
SELECT
    @Q1 = PERCENTILE_CONT(0.25) WITHIN GROUP (ORDER BY Item_MRP) OVER (),
    @Q3 = PERCENTILE_CONT(0.75) WITHIN GROUP (ORDER BY Item_MRP) OVER ()
FROM blinkit_grocery_cpy;
```

**-- Verify the values of Q1 and Q3**  
SELECT @Q1 AS FirstQuartile, @Q3 AS ThirdQuartile;

	FirstQuartile	ThirdQuartile
1	93.8265	185.6437

```
-- Step 2: Update Price_Tier
UPDATE blinkit_grocery_cpy
SET Price_Tier = CASE
    WHEN Item_MRP <= @Q1 THEN 'Low'
    WHEN Item_MRP > @Q1 AND Item_MRP <= @Q3 THEN 'Medium'
    ELSE 'High'
END;
-- Step 3: Update verification
SELECT Item_MRP, Price_Tier FROM blinkit_grocery_cpy;
```

	Item_MRP	Price_Tier
1	249.8092	High
2	48.2692	Low
3	141.618	Medium
4	182.095	Medium
5	53.8614	Low

### 7. Add Descriptive Labels

```
---Add Descriptive Labels Using CASE
ALTER TABLE blinkit_grocery_cpy
ADD Outlet_Type_Description NVARCHAR(50);

UPDATE blinkit_grocery_cpy
SET Outlet_Type_Description = CASE
    WHEN Outlet_Type = 'Grocery Store' THEN 'Small Local Grocery Store'
    WHEN Outlet_Type = 'Supermarket Type1' THEN 'Supermarket Type 1'
    WHEN Outlet_Type = 'Supermarket Type2' THEN 'Supermarket Type 2'
    WHEN Outlet_Type = 'Supermarket Type3' THEN 'Supermarket Type 3'
    ELSE 'Unknown'
END;
```



---Verify the Update

```
SELECT Outlet_Type, Outlet_Type_Description, COUNT(*) AS Frequency
FROM blinkit_grocery_cpy
GROUP BY Outlet_Type, Outlet_Type_Description;
```

Results Messages			
	Outlet_Type	Outlet_Type_Description	Frequency
1	Grocery Store	Small Local Grocery Store	1083
2	Supermarket Type1	Supermarket Type 1	5577
3	Supermarket Type2	Supermarket Type 2	928
4	Supermarket Type3	Supermarket Type 3	935

---Repeat for Other Categorical Columns

```
ALTER TABLE blinkit_grocery_cpy
ADD Outlet_Location_Description NVARCHAR(50);
```

```
UPDATE blinkit_grocery_cpy
SET Outlet_Location_Description = CASE
    WHEN Outlet_Location_Type = 'Tier 1' THEN 'Urban Area'
    WHEN Outlet_Location_Type = 'Tier 2' THEN 'Suburban Area'
    WHEN Outlet_Location_Type = 'Tier 3' THEN 'Rural Area'
    ELSE 'Unknown'
END;
```

---Verify the Update

```
SELECT Outlet_Type,Outlet_Location_Type,Outlet_Location_Description, COUNT(*) AS Frequency
FROM blinkit_grocery_cpy
GROUP BY Outlet_Type, Outlet_Location_Type, Outlet_Location_Description;
```

Results Messages				
	Outlet_Type	Outlet_Location_Type	Outlet_Location_Description	Frequency
1	Supermarket Type1	Tier 2	Suburban Area	2785
2	Grocery Store	Tier 3	Rural Area	555
3	Supermarket Type1	Tier 1	Urban Area	1860
4	Supermarket Type1	Tier 3	Rural Area	932
5	Grocery Store	Tier 1	Urban Area	528
6	Supermarket Type2	Tier 3	Rural Area	928
7	Supermarket Type3	Tier 3	Rural Area	935

8. Verify Unique Identifiers

Item\_Identifier:

Ensure each Item\_Identifier is unique and not reused inappropriately.

**Action:** Investigate and clean if inconsistencies are found.

---Check for Duplicates

```
SELECT
    Item_Identifier,
    COUNT(*) AS Frequency
FROM blinkit_grocery_cpy
GROUP BY Item_Identifier
HAVING COUNT(*) > 1;
```

Results Messages		
	Item_Identifier	Frequency
614	FDV44	6
615	NCK18	7
616	FDG16	6
617	FDN31	4
618	FDA31	4
619	FDU26	7
620	NCJ30	9
621	FDR14	5
622	FDW34	7
623	DRJ13	6

---Investigate Duplicates

```
SELECT *
FROM blinkit_grocery_cpy
WHERE Item_Identifier IN (
    SELECT Item_Identifier
    FROM blinkit_grocery_cpy
    GROUP BY Item_Identifier
    HAVING COUNT(*) > 1
)
```

ORDER BY Item_Identifier;																	
Results Messages																	
	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Identifier	Outlet_Establishment_Year	Outlet_Size	Outlet_Location_Type	Outlet_Type	Item_Outlet_Sales	Outlet_Age	Sales_Per_Item	Price_Tier	Outlet_Type_Description	Outlet_Location_Description
1	DRA12	11.6000003814697	Low Fat	0.0411775037646294	Soft Drinks	140.315399169922	OUT017	2007	Small	Tier 2	Supermarket Type1	2552.67724609375	18	18.1924240760094	Medium	Supermarket Type 1	Suburban Area
2	DRA12	11.6000003814697	Low Fat	0.0704818408547522	Soft Drinks	141.61540222168	OUT045	2002	Small	Tier 2	Supermarket Type1	3829.01586914063	23	27.0381315102069	Medium	Supermarket Type 1	Suburban Area
3	DRA12	11.6000003814697	Low Fat	0.0409118235111237	Soft Drinks	142.315399169922	OUT013	1987	High	Tier 3	Supermarket Type1	2552.67724609375	38	17.9367606104656	Medium	Supermarket Type 1	Rural Area
4	DRA12	11.6000003814697	Low Fat	0.0704818408547522	Soft Drinks	141.915405273438	OUT035	2004	Small	Tier 2	Supermarket Type1	992.707824707031	21	6.99506739803418	Medium	Supermarket Type 1	Suburban Area
5	DRA12	11.6000003814697	Low Fat	0.041112694889307	Soft Drinks	142.015396118164	OUT018	2009	Medium	Tier 3	Supermarket Type2	850.892395019531	16	5.99155034086266	Medium	Supermarket Type 2	Rural Area
6	DRA12	11.6000003814697	Low Fat	0.068535037387337	Soft Drinks	143.015396118164	OUT010	1998	Small	Tier 3	Grocery Store	283.630798339844	27	1.98321863266734	Medium	Small Local Grocery Store	Rural Area
7	DRA24	19.3500003814697	Regular	0.0398950950249488	Soft Drinks	162.486801147461	OUT013	1987	High	Tier 3	Supermarket Type1	4422.24365234375	38	27.2160176772171	Medium	Supermarket Type 1	Rural Area
8	DRA24	19.3500003814697	Regular	0.0399903133511543	Soft Drinks	165.086807250977	OUT049	1999	Medium	Tier 1	Supermarket Type1	982.720825195313	26	5.95275201913204	Medium	Supermarket Type 1	Urban Area

SELECT

Item\_Identifier,

MAX(Item\_Weight) AS Item\_Weight,

MAX(Item\_MRP) AS Item\_MRP

FROM blinkit\_grocery\_cpy

GROUP BY Item\_Identifier;

	Item_Identifier	Item_Weight	Item_MRP
1	FDH19	19.35	175.4738
2	FDX20	7.365	228.372
3	NCO07	9.06	213.756
4	FDH20	16.1	98.441
5	NCP53	14.75	238.6906
6	FDX32	15.1	146.2786

9. Handle Data Type Consistency

Ensure all columns have appropriate data types:

---Verify Current Data Types

SELECT COLUMN\_NAME, DATA\_TYPE

FROM INFORMATION\_SCHEMA.COLUMNS

WHERE TABLE\_NAME = 'blinkit\_grocery\_cpy';

	COLUMN_NAME	DATA_TYPE
1	Item_Identifier	nvarchar
2	Item_Weight	float
3	Item_Fat_Content	nvarchar
4	Item_Visibility	float
5	Item_Type	nvarchar
6	Item_MRP	float
7	Outlet_Identifier	nvarchar
8	Outlet_Establishment_Year	int
9	Outlet_Size	nvarchar
10	Outlet_Location_Type	nvarchar
11	Outlet_Type	nvarchar
12	Item_Outlet_Sales	float
13	Outlet_Age	int
14	Price_Tier	nvarchar
15	Sales_Per_Item	float

---Convert Outlet\_Establishment\_Year to Categorical

ALTER TABLE blinkit\_grocery\_cpy

ALTER COLUMN Outlet\_Establishment\_Year NVARCHAR(10);

	COLUMN_NAME	DATA_TYPE
1	Item_Identifier	nvarchar
2	Item_Weight	float
3	Item_Fat_Content	nvarchar
4	Item_Visibility	float
5	Item_Type	nvarchar
6	Item_MRP	float
7	Outlet_Identifier	nvarchar
8	Outlet_Establishment_Year	nvarchar
9	Outlet_Size	nvarchar
10	Outlet_Location_Type	nvarchar
11	Outlet_Type	nvarchar
12	Item_Outlet_Sales	float
13	Outlet_Age	int
14	Price_Tier	nvarchar
15	Sales_Per_Item	float

---Add column to store the Outlet Age Category

SELECT

MIN(Outlet\_Age) AS Lowest\_Age,

MAX(Outlet\_Age) AS Highest\_Age

FROM blinkit\_grocery\_cpy;



Results			Messages		
	Lowest_Age	Highest_Age			
1	16	40			

ALTER TABLE blinkit\_grocery\_cpy  
ADD Outlet\_Age\_Category NVARCHAR(20);

DECLARE @CurrentYear INT = YEAR(GETDATE());

UPDATE blinkit\_grocery\_cpy  
SET Outlet\_Age\_Category = CASE  
    WHEN (@CurrentYear - Outlet\_Establishment\_Year) <= 25 THEN 'New'  
    WHEN (@CurrentYear - Outlet\_Establishment\_Year) BETWEEN 26 AND 35 THEN 'Moderate'  
    ELSE 'Old'  
END;

---Verify the Update

SELECT Outlet\_Establishment\_Year,  
    Outlet\_Age,  
    Outlet\_Age\_Category  
FROM blinkit\_grocery\_cpy  
ORDER BY Outlet\_Age;

Results				Messages	
	Outlet_Establishment_Year	Outlet_Age	Outlet_Age_Category		
1	2009	16	New		
2	2009	16	New		
3	2009	16	New		
4	2009	16	New		
5	2009	16	New		
6	2009	16	New		
7	2009	16	New		
8	2009	16	New		
9	2009	16	New		
10	2009	16	New		
11	2009	16	New		

Ensure Item\_Visibility, Item\_Weight, and Item\_MRP are numerical for calculations.

-----Validate Data Consistency

SELECT  
    SUM(CASE WHEN Item\_Visibility IS NULL THEN 1 ELSE 0 END) AS Null\_Item\_Visibility,  
    SUM(CASE WHEN Item\_Weight IS NULL THEN 1 ELSE 0 END) AS Null\_Item\_Weight,  
    SUM(CASE WHEN Item\_MRP IS NULL THEN 1 ELSE 0 END) AS Null\_Item\_MRP  
FROM blinkit\_grocery\_cpy;

Results				Messages	
	Null_Item_Visibility	Null_Item_Weight	Null_Item_MRP		
1	0	0	0		

Data Dictionary (After Cleaning )

Column Name	Description
Item_Identifier	Unique identifier for each item in the dataset.
Item_Weight	Weight of the product (in kilograms), rounded to two decimal places.
Item_Fat_Content	Indicates the fat content of the item (e.g., Low Fat, Regular).
Item_Visibility	The percentage of total display area allocated to this item in a store.
Item_Type	Categorical variable specifying the broad category of the item (e.g., Dairy, Snacks).
Item_MRP	Maximum Retail Price (MRP) of the item in the store.
Outlet_Identifier	Unique identifier for each outlet/store.
Outlet_Establishment_Year	The year in which the outlet was established.
Outlet_Size	Categorical variable specifying the size of the outlet (e.g., Small, Medium, Large).
Outlet_Location_Type	Specifies the type of location where the outlet is situated (e.g., Tier 1, Tier 2).
Outlet_Type	The category of the outlet (e.g., Grocery Store, Supermarket).
Item_Outlet_Sales	Sales of the item in the particular outlet.
Outlet_Age	Calculated age of the outlet, derived from 2023 – Outlet_Establishment_Year.
Price_Tier	Categorical variable dividing items into price ranges (e.g., Low, Medium, High).
Sales_Per_Item	Average sales per item, calculated as Item_Outlet_Sales / Item_Weight.
Outlet_Age_Category	Categorizes Outlet_Age into groups (e.g., “Low”, “Moderate” “High”).
Outlet_Type_Description	Detailed description of the outlet type
Outlet_Location_Description	Description of the outlet location type (e.g., “Tier 1 Urban Area”).

Download: [SQL Query](#)

Download: [Clean Data](#)

[If you have any further questions or would like to explore opportunities to work together, please don't hesitate to reach out to me.](#)

**THANK YOU**