REAL-TIME CAPSTONE PROJECT : RETAIL SALES & CUSTOMER INSIGHTS

Submitted by:
Thiageshwaran S
ADMM

Technology and Transformation

March 2026

RETAIL SALES & CUSTOMER INSIGHTS

1. Business Problem Statement

1.1 Background

The retail chain operates across multiple regions and manages a diverse customer base along with a wide range of product categories. Due to increasing data volume and operational complexity, the organization faced challenges in transforming raw transactional data into actionable business insights.

The absence of a centralized analytics system limited visibility into performance metrics, customer behavior patterns, and regional trends.

1.2 Key Business Challenges

1.2.1 Identifying Top-Performing Products

Management lacked clarity on:

- ✓ Which products generate the highest revenue
- ✓ Which categories drive the majority of sales
- ✓ Which products should be promoted, optimized, or discontinued

1.2.2 Understanding Complex Purchasing Patterns

Retail purchasing behavior is influenced by multiple factors:

- ✓ Seasonality (monthly and quarterly trends)
- ✓ Promotional campaigns
- ✓ Customer demographics (region and gender)

Without structured analysis, it was difficult to detect patterns, repeat purchase behavioror revenue concentration.

1.2.3 Visualizing Regional Sales Trends

Regional insights were required for:

- ✓ Inventory optimization
- ✓ Marketing strategy alignment

- ✓ Supply chain planning
- ✓ Performance benchmarking

However, inconsistent and unstructured data prevented meaningful regional comparison.

2. Project Objectives – KPIs and Goals

2.1 Primary Objectives

The primary objectives of the project were:

1. Design and implement a structured Data Warehouse
2. Develop meaningful business KPIs
3. Create an executive-level Power BI dashboard
4. Enable interactive drill-down analytics

2.2 Key Performance Indicators (KPIs)

The following KPIs were defined and implemented:

- ✓ Total Sales Revenue
- ✓ Sales Growth Rate
- ✓ Previous Period Sales
- ✓ Total Quantity Sold
- ✓ Customer Count
- ✓ Average Transaction Value (ATV)
- ✓ Customer Lifetime Value (CLV)
- ✓ Revenue Per Customer
- ✓ Total Product Sales

2.3 Strategic Goals

- Identify revenue-driving products
- Analyze seasonal and regional sales trends

- Evaluate customer lifetime value
- Support executive decision-making through visual analytics

3. ETL Process Using Python and Microsoft SQL Server

3.1 Tools and Technologies Used

- ✓ Python (Pandas, SQLAlchemy, PyODBC)
- ✓ Jupyter Notebook
- ✓ Microsoft SQL Server
- ✓ SQL Server Management Studio (SSMS)

3.2 Data Sources

S.No	Data Source	Format
1	Customer Data	JSON
2	Product Data	CSV
3	Sales Data	CSV

3.3 ETL Architecture

The ETL pipeline followed a structured approach:

1. Extract – Load raw data into Python
2. Transform – Clean, normalize, and validate data
3. Load – Insert cleaned data into SQL Server Data Warehouse

3.4 Extraction Phase

Customer data was extracted from JSON files.
Product and sales data were extracted from CSV files using Pandas.Data was loaded into DataFrames for transformation.

Cell 1 → Imports

```
import pandas as pd
import json
```

Cell 2 → Load Customers JSON

Generate + Code + Markdown

```
with open("customers.json") as f:
    customers_data = json.load(f)

df_customers = pd.DataFrame(customers_data)
df_customers.head()
```

	CustomerID	FirstName	LastName	Gender	Region	SSN
0	C0001	Gregory	Miller	M	Ohho	082573197
1	C0002	Marvin	None	male	california	309-6256-21
2	C0003	Gregory	Smith	Male	New Yorkk	407071725
3	C0004	Edward	Davis	male	Ohho	948.18.5740
4	C0005	Reginald	Dawson	male	Texas	975.72.3056

Cell 3 → Load Products CSV

```
df_products = pd.read_csv("products.csv")
df_products.head()
```

	ProductID	ProductName	Category
0	1000	Washing Machine	Home Appliances
1	1001	Refrigerator	Home Appliances
2	1002	Blender	Home Appliances
3	1003	Microwave	Home Appliances
4	1004	Vacuum Cleaner	Home Appliances

Cell 4 → Load Sales CSV

```
df_sales = pd.read_csv("sales 1.csv")
df_sales.head()
```

	SaleID	ProductID	CustomerID	SalesAmount	Quantity	Timestamp
0	c0078cd1-3ddc-439a-b41d-7498e8e75e62	1020	C0460	1639.10	18.0	2024-08-09T20:21:00
1	be8532af-16b7-432c-a13d-4c6e6836d12d	1029	C0198	186.11	42.0	2024-11-28T05:01:00
2	8666f277-cc58-4fde-9d58-697da9e040f2	1012	C0524	1712.72	49.0	2024-10-13T00:36:00
3	383e0a2b-7e00-454e-a269-b2f73ecd33f2	1016	C0285	NaN	50.0	2024-07-06T23:39:00
4	d9a7f974-9572-48dd-aa36-3f96b280dfc7	1021	C0369	66.36	21.0	2024-06-08T19:58:00

Check Columns

```
print(df_customers.columns)
print(df_products.columns)
print(df_sales.columns)
```

```
Index(['CustomerID', 'FirstName', 'LastName', 'Gender', 'Region', 'SSN'], dtype='object')
Index(['ProductID', 'ProductName', 'Category'], dtype='object')
Index(['SaleID', 'ProductID', 'CustomerID', 'SalesAmount', 'Quantity',
       'Timestamp'],
      dtype='object')
```

✅ Missing Data Check

```
print(df_customers.isnull().sum())
print(df_products.isnull().sum())
print(df_sales.isnull().sum())
```

```
CustomerID      0
FirstName       0
LastName      320
Gender          0
Region         69
SSN             0
dtype: int64
ProductID      0
ProductName    0
Category       0
```

Check Duplicates

```
print(df_customers.duplicated().sum())
print(df_products.duplicated().sum())
print(df_sales.duplicated().sum())
```

```
100
0
0
```

✅ Timestamp → DateTime

```
df_sales['Timestamp'] = pd.to_datetime(df_sales['Timestamp'])
```

✅ Numeric Columns

```
df_sales['SalesAmount'] = df_sales['SalesAmount'].astype(float)
df_sales['Quantity'] = df_sales['Quantity'].astype('Int64')
```

3.5 Transformation Phase

This phase involved extensive data cleaning and validation.

3.5.1 Issues Identified and Resolutions

Issue 1: CustomerID Format Mismatch

Customer IDs were in the format : C0001

SQL schema expected integer values.

Resolution:
Extracted numeric portion and converted to integer.

Issue 2: Region Inconsistencies

Examples identified:

- ✓ Nw York
- ✓ New Yorkk
- ✓ Ny
- ✓ Californiya
- ✓ Texaz
- ✓ Ohho

Resolution:

- ✓ Trimmed whitespace
- ✓ Standardized case
- ✓ Replaced incorrect spellings
- ✓ Applied consistent naming conventions

Issue 3: Duplicate Primary Keys

Error encountered during loading:

Violation of PRIMARY KEY constraint.

Root Cause: Duplicate CustomerID entries after transformation.

Resolution:
Removed duplicate records using Pandas before loading.

Issue 4: Data Type Conversion Errors

Conversion failed when inserting varchar into integer columns.

Resolution:
Aligned SQL schema with cleaned data types and validated data prior to loading.

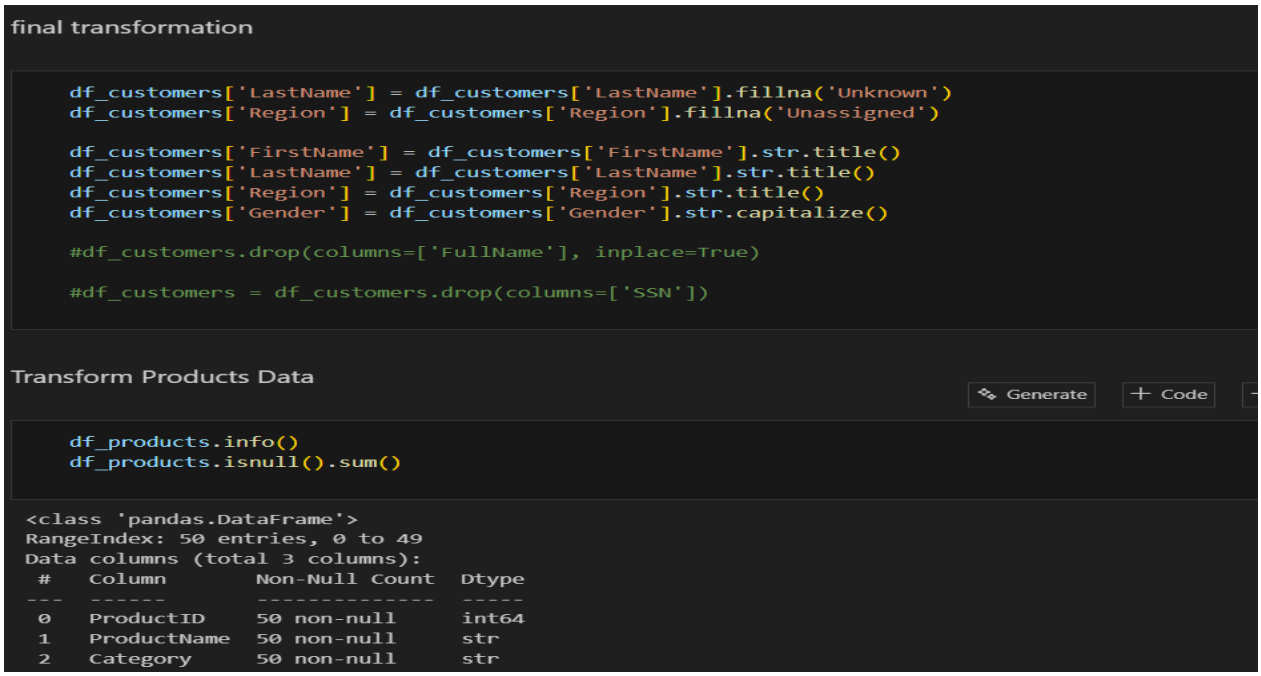
Issue 5: Foreign Key Constraints Blocking Table Replacement

Attempting to replace dimension tables caused foreign key constraint errors.

Resolution:
Followed proper loading order:

1. Drop Fact table
2. Drop Dimension tables
3. Recreate tables
4. Load Dimension tables first
5. Load Fact table last

This preserved referential integrity.



Fix Issues

```
df_products.dropna(subset=['ProductID'], inplace=True)

df_products['Category'] = df_products['Category'].str.lower().str.strip()
df_products['ProductName'] = df_products['ProductName'].str.strip()
```

Transform Sales Data

```
df_sales.info()
df_sales.isnull().sum()
```

```
<class 'pandas.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 6 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   SaleID       10000 non-null  str
 1   ProductID    10000 non-null  int64
 2   CustomerID   10000 non-null  str
 3   SalesAmount  9503 non-null   float64
 4   Quantity     9887 non-null   float64
 5   Timestamp    10000 non-null  str
dtypes: float64(2), int64(1), str(3)
memory usage: 468.9 KB
```

Fix Data Types

```
df_sales['Timestamp'] = pd.to_datetime(df_sales['Timestamp'])

df_sales['SalesAmount'] = df_sales['SalesAmount'].astype(float)
df_sales['Quantity'] = df_sales['Quantity'].astype('Int64')
```

Remove Critical Missing Rows

```
df_sales.dropna(subset=['SaleID', 'CustomerID', 'ProductID'], inplace=True)
```

3.6 Loading Phase

Data was loaded into SQL Server using SQLAlchemy.

Load Order:

1. DimCustomer
2. DimProduct
3. DimDate
4. FactSales

This ensured consistent foreign key relationships.

LOAD PHASE → Connect Python → SQL Server

```
from sqlalchemy import create_engine

server = 'localhost'
database = 'RetailDW'

connection_string = (
    "mssql+pyodbc://@localhost/RetailDW"
    "?driver=ODBC+Driver+17+for+SQL+Server"
    "&trusted_connection=yes"
)

engine = create_engine(connection_string)
```

```
pd.read_sql("SELECT TOP 5 * FROM dbo.DimCustomer", engine)
```

CustomerID	FirstName	LastName	Gender	Region

Load Dimension Tables FIRST

```
df_customers.head()
```

	CustomerID	FirstName	LastName	Gender	Region
0	1	Gregory	Miller	M	Ohho
1	2	Marvin	Unknown	Male	California
2	3	Gregory	Smith	Male	New Yorkk
3	4	Edward	Davis	Male	Ohho
4	5	Reginald	Dawson	Male	Texas

coverting customer id in the format C0043 to 43 as int

```
df_customers['CustomerID'] = df_customers['CustomerID'].str.replace('C', '').astype(int)
```

Generate + Code + Mari

```
df_sales['CustomerID'] = df_sales['CustomerID'].str.replace('C', '').astype(int)
```

```
df_customers['CustomerID'].duplicated().sum()
```

```
np.int64(0)
```

Load Fact Table LAST Load FactSales

```
df_sales.to_sql(
    name='FactSales',
    con=engine,
    if_exists='append',
    index=False
)
```

```
228
```

load dimCustomer

```
df_customers.to_sql(
    name='DimCustomer',
    con=engine,
    if_exists='append',
    index=False
)
```

```
162
```

Load DimProduct

```
df_products.to_sql(
    name='DimProduct',
    con=engine,
    if_exists='append',
    index=False
)
```

```
50
```

4. Exploratory Data Analysis and Data Warehouse Design

4.1 Exploratory Data Analysis (EDA)

EDA was performed to:

- ✓ Check null values
- ✓ Validate primary and foreign keys
- ✓ Analyze sales distribution
- ✓ Detect seasonal patterns
- ✓ Identify data inconsistencies

Key Findings:

- ✓ Monthly sales variation indicating seasonality
- ✓ Region inconsistencies

- ✓ Missing values in certain customer attributes
- ✓ Skewed revenue distribution among products

SQL CODE:

```
USE RetailDW;
CREATE TABLE DimCustomer (
    CustomerID INT PRIMARY KEY,
    FirstName VARCHAR(50) NOT NULL,
    LastName VARCHAR(50) NOT NULL,
    Gender VARCHAR(10),
    Region VARCHAR(50)
);
CREATE TABLE DimProduct (
    ProductID INT PRIMARY KEY,
    ProductName VARCHAR(100) NOT NULL,
    Category VARCHAR(50) NOT NULL
);

CREATE TABLE DimDate (
    DateKey INT PRIMARY KEY,
    FullDate DATE NOT NULL,
    Year INT,
    Quarter INT,
    Month INT,
    Day INT
);

CREATE TABLE FactSales (
    SaleKey INT IDENTITY PRIMARY KEY,
    SaleID VARCHAR(50),  -- UUID Business Key
    ProductID INT,
    CustomerID INT,
    SalesAmount DECIMAL(10,2) NOT NULL,
    Quantity INT NOT NULL,
    Timestamp DATETIME NOT NULL,

    FOREIGN KEY (ProductID) REFERENCES DimProduct(ProductID),
    FOREIGN KEY (CustomerID) REFERENCES DimCustomer(CustomerID)
);

SELECT * FROM DimCustomer;
SELECT * FROM DimProduct;
SELECT * FROM DimDate;
SELECT * FROM FactSales;

DROP TABLE FactSales;
DROP TABLE DimCustomer;

ALTER TABLE FactSales
ALTER COLUMN SalesAmount FLOAT NULL;

ALTER TABLE FactSales
ALTER COLUMN Quantity INT NULL;

SELECT COUNT(*) FROM dbo.DimCustomer;
SELECT COUNT(*) FROM DimProduct;
SELECT COUNT(*) FROM FactSales;
SELECT * FROM dbo.DimCustomer;
SELECT TOP 10 * FROM FactSales;
```

```sql
SELECT TOP 10 * FROM DimCustomer;
SELECT TOP 10 * FROM DimProduct;

-- KPI 1 → Total Sales Revenue
SELECT
    SUM(SalesAmount) AS TotalRevenue
FROM FactSales;

-- KPI 2 → Total Units Sold
SELECT
    SUM(Quantity) AS TotalUnitsSold
FROM FactSales;

-- KPI 3 → Top-Selling Products
-- (Which products generate most revenue?)
SELECT
    p.ProductName,
    SUM(f.SalesAmount) AS Revenue
FROM FactSales f
JOIN DimProduct p
    ON f.ProductID = p.ProductID
GROUP BY p.ProductName
ORDER BY Revenue DESC;

-- KPI 4 → Sales by Category
-- Which category performs best?
SELECT
    p.Category,
    SUM(f.SalesAmount) AS Revenue
FROM FactSales f
JOIN DimProduct p
    ON f.ProductID = p.ProductID
GROUP BY p.Category
ORDER BY Revenue DESC;

--KPI 5 → Sales by Region
-- Which regions drive revenue?
SELECT
    c.Region,
    SUM(f.SalesAmount) AS Revenue
FROM FactSales f
JOIN DimCustomer c
    ON f.CustomerID = c.CustomerID
GROUP BY c.Region
ORDER BY Revenue DESC;

-- KPI 6 → Average Transaction Value (ATV)
SELECT
    SUM(SalesAmount) / COUNT(SaleID) AS AvgTransactionValue
FROM FactSales;

-- KPI 7 → Monthly Sales Trend
SELECT
    YEAR(Timestamp) AS Year,
    MONTH(Timestamp) AS Month,
    SUM(SalesAmount) AS Revenue
FROM FactSales
GROUP BY
    YEAR(Timestamp),
    MONTH(Timestamp)
ORDER BY Year, Month;

-- Best-Selling Products by Quantity
```

```sql
SELECT
    p.ProductName,
    SUM(f.Quantity) AS UnitsSold
FROM FactSales f
JOIN DimProduct p
    ON f.ProductID = p.ProductID
GROUP BY p.ProductName
ORDER BY UnitsSold DESC;

-- Sales Growth Rate
-- How fast are sales increasing or decreasing?
WITH MonthlySales AS (
    SELECT
        YEAR(Timestamp) AS Year,
        MONTH(Timestamp) AS Month,
        SUM(SalesAmount) AS Revenue
    FROM FactSales
    GROUP BY
        YEAR(Timestamp),
        MONTH(Timestamp)
)
SELECT *,
    LAG(Revenue) OVER (ORDER BY Year, Month) AS PreviousRevenue,

    (Revenue - LAG(Revenue) OVER (ORDER BY Year, Month)) * 100.0
        / LAG(Revenue) OVER (ORDER BY Year, Month)
        AS GrowthRatePercent

FROM MonthlySales;

-- Customer Lifetime Value (CLV)

SELECT
    c.CustomerID,
    c.FirstName,
    c.LastName,
    SUM(f.SalesAmount) AS CustomerLifetimeValue
FROM FactSales f
JOIN DimCustomer c
    ON f.CustomerID = c.CustomerID
GROUP BY
    c.CustomerID,
    c.FirstName,
    c.LastName
ORDER BY CustomerLifetimeValue DESC;

-- Product Return Rate
SELECT
    p.ProductName,

    SUM(CASE WHEN f.Quantity < 0 THEN ABS(f.Quantity) ELSE 0 END) * 100.0
        / SUM(ABS(f.Quantity))
        AS ReturnRatePercent

FROM FactSales f
JOIN DimProduct p
    ON f.ProductID = p.ProductID

GROUP BY p.ProductName;
```

4.2 Data Warehouse Design

A Star Schema architecture was implemented.

4.2.1 Fact Table

Fact Sales:

- ✓ SaleID
- ✓ CustomerID
- ✓ ProductID
- ✓ SalesAmount
- ✓ Quantity
- ✓ Timestamp

4.2.2 Dimension Tables

- ✓ DimCustomer
- ✓ DimProduct
- ✓ DimDate

	AvgTransactionValue
1	1365.58097699999

	Year	Month	Revenue
1	2024	6	2044776.94
2	2024	7	2212950.47000001
3	2024	8	2688782.43000001
4	2024	9	2102306.85
5	2024	10	2319876.14
6	2024	11	2287116.94

	ProductName	UnitsSold
1	Doll	5770
2	Drone	5679
3	Board Game	5676
4	Chair	5597
5	Bicycle	5594

Results Messages

	Category	Revenue
1	toys	5906967.6
2	electronics	5047478.2
3	furniture	1482418.72
4	home appliances	1218945.25

	Region	Revenue
1	New York	1600897.42
2	Texas	1330879.07
3	Texaz	1324252.41
4	Ny	1310422.51
5	Californiya	1300042.74
6	New Yorkk	1291312.28
7	Ohho	1283658.39
8	Ohio	1144051.4

Results Messages

	TotalRevenue
1	13655809.7699999

	TotalUnitsSold
1	249465

	ProductName	Revenue
1	Action Figure	647238.46
2	Swing	403402.83
3	Stuffed Animal	400791.8
4	Board Game	393984.96
5	Doll	384252.42
6	Table	373000.63
7	Drone	348233.06
8	Keyboard	340968.95

	Year	Month	Revenue	PreviousRevenue	GrowthRatePercent
1	2024	6	2044776.94	NULL	NULL
2	2024	7	2212950.47000001	2044776.94	8.22454159718778
3	2024	8	2688782.43000001	2212950.47000...	21.5021513789235
4	2024	9	2102306.85	2688782.43000...	-21.81193887078...
5	2024	10	2319876.14	2102306.85	10.3490739232475
6	2024	11	2287116.94	2319876.14	-1.412109872383...

	CustomerID	FirstName	LastName	CustomerLifetimeValue
1	381	Roy	West	108682.71
2	51	David	Unknown	102484.04
3	435	Robert	Unknown	99826.71
4	555	Alec	Unknown	96522.37
5	122	Dustin	Cook	96242.94
6	983	Daniel	Unknown	94351.41
7	909	Steven	Hamilton	92069.23

4.3 Staging Area

Raw data was initially loaded into staging tables.

After validation and transformation, cleaned data was loaded into dimensional tables.

4.4 Rationale for Star Schema

- ✓ Optimized for BI queries
- ✓ Simplified reporting structure
- ✓ Improved performance

- ✓ Clear separation of measures and attributes

5. Power BI Implementation

5.1 Data Preparation

Connected Power BI to SQL Server database (RetailDW).

Import mode was used.

Loaded tables:

- ✓ FactSales
- ✓ DimCustomer
- ✓ DimProduct
- ✓ DimDate

Power Query was used for additional cleaning:

- ✓ Trim
- ✓ Replace values
- ✓ Standardize region names
- ✓ Correct data types

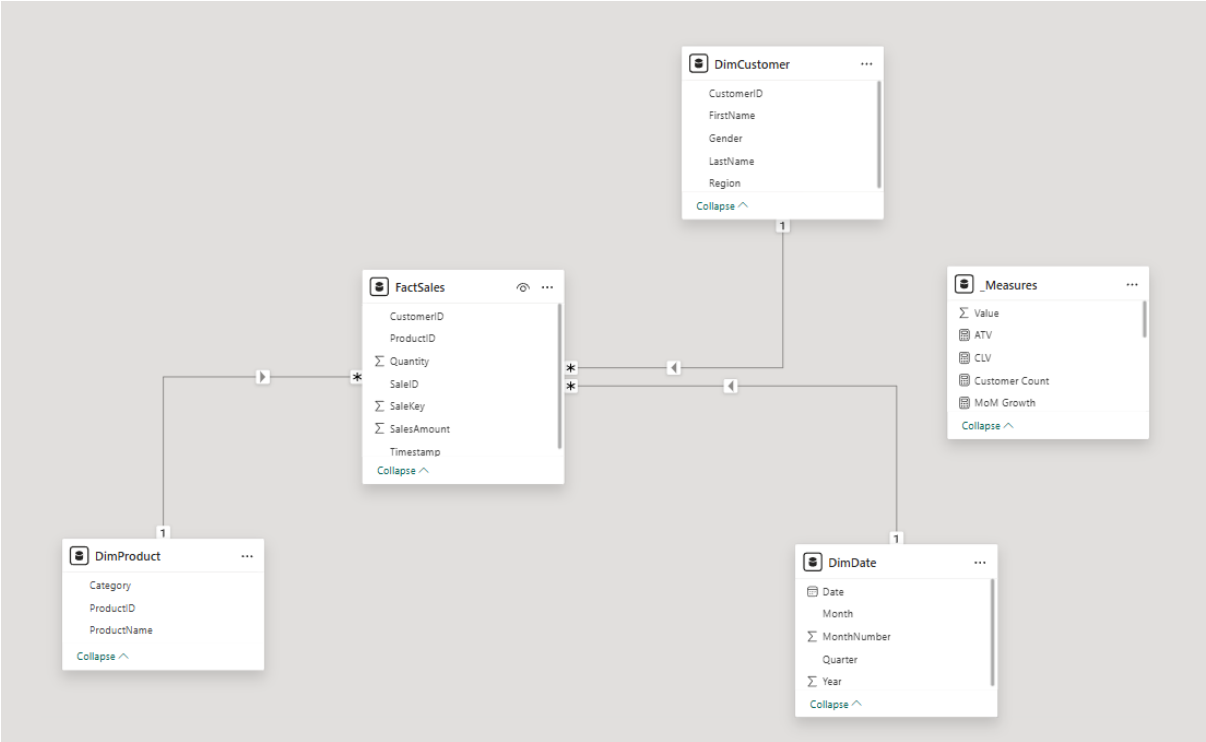
5.2 Data Modeling

Relationships defined:

- ✓ FactSales → DimCustomer (Customer ID)
- ✓ FactSales → DimProduct (Product ID)
- ✓ FactSales → DimDate (Date)

Relationship Type:

- ✓ One-to-Many
- ✓ Single-direction filter



5.3 KPI Measures (DAX Implementation)

Examples include:

Total Sales Revenue
Sales Growth Rate
Customer Lifetime Value
Revenue Per Customer
Average Transaction Value

Advanced measures included:

- ✓ Previous Period Sales
- ✓ Growth Percentage
- ✓ Repeat Purchase Rate

Total Sales Revenue

Total Sales Revenue =
SUM(FactSales[SalesAmount])

Total Quantity Sold

```
Total Quantity Sold =
SUM(FactSales[Quantity])
```

Customer Count

```
Customer Count =
DISTINCTCOUNT(DimCustomer[CustomerID])
```

Total Product Sales

```
Total Product Sales =
SUM(FactSales[SalesAmount])
```

Revenue Per Customer

```
Revenue Per Customer =
DIVIDE(
    [Total Sales Revenue],
    [Customer Count]
)
```

Average Transaction Value (ATV)

```
ATV =
DIVIDE(
    [Total Sales Revenue],
    DISTINCTCOUNT(FactSales[SaleID])
)
```

Previous Period Sales

```
Previous Period Sales =
CALCULATE(
    [Total Sales Revenue],
    SAMEPERIODLASTYEAR(DimDate[Date])
)
```

Sales Growth Rate

```
Sales Growth Rate =
DIVIDE(
    [Total Sales Revenue] − [Previous Period Sales],
    [Previous Period Sales]
)
```

Customer Lifetime Value (CLV)

```
CLV =
DIVIDE(
    [Total Sales Revenue],
    [Customer Count]
)
```

5.4 Dashboard Development

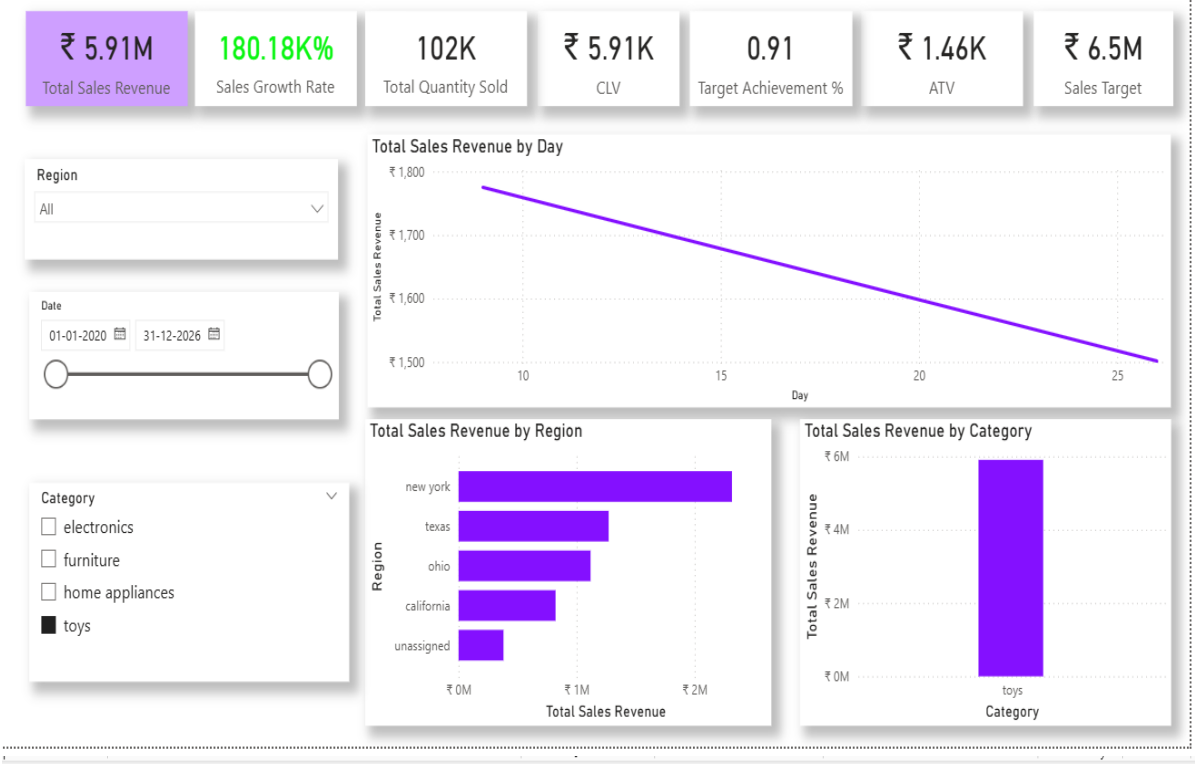
A four-page professional dashboard was created:

1. Executive Overview
2. Sales Performance Analysis
3. Customer Intelligence
4. Product and Category Insights

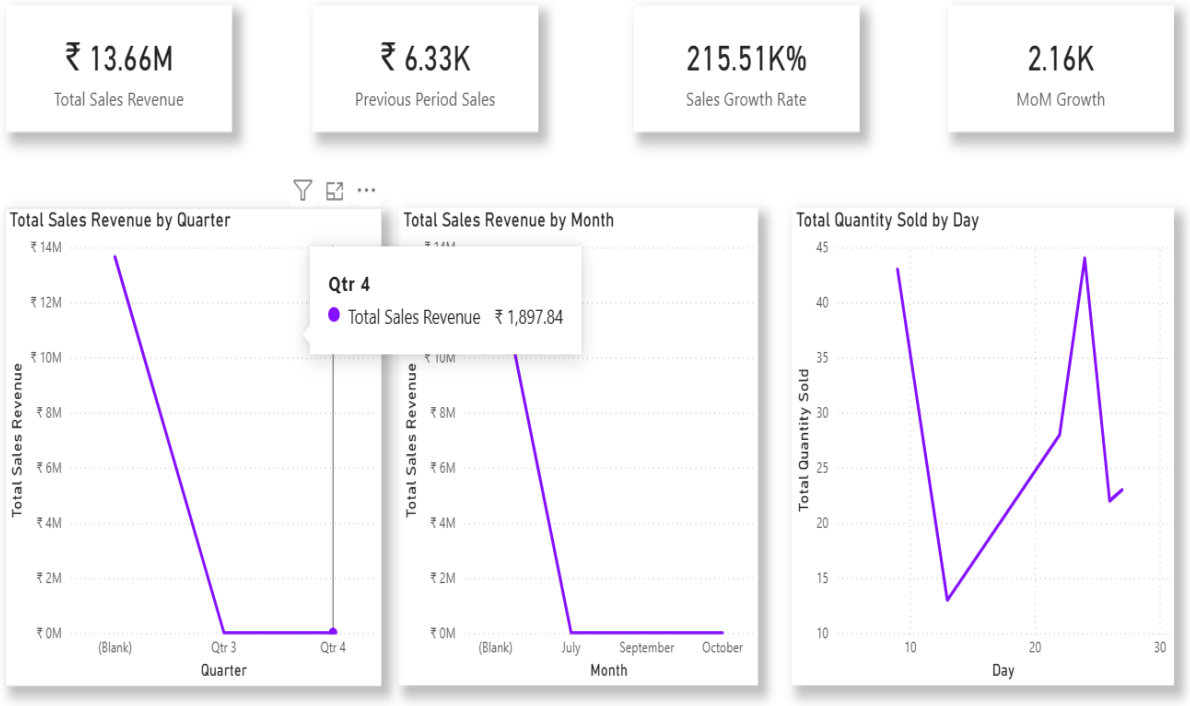
Features implemented:

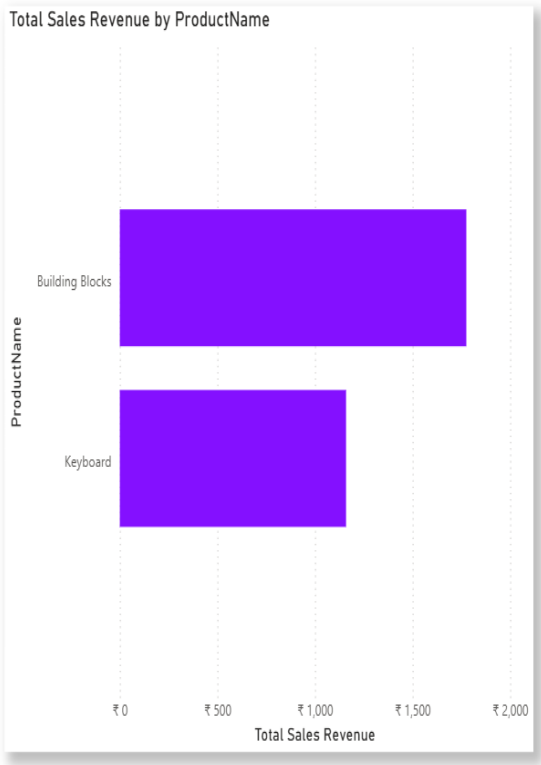
- ✓ KPI cards
- ✓ Trend analysis charts
- ✓ Regional comparison visuals
- ✓ Slicers and filters
- ✓ Drill-through pages
- ✓ Dynamic titles
- ✓ Custom tooltip pages
- ✓ Smart narrative insights

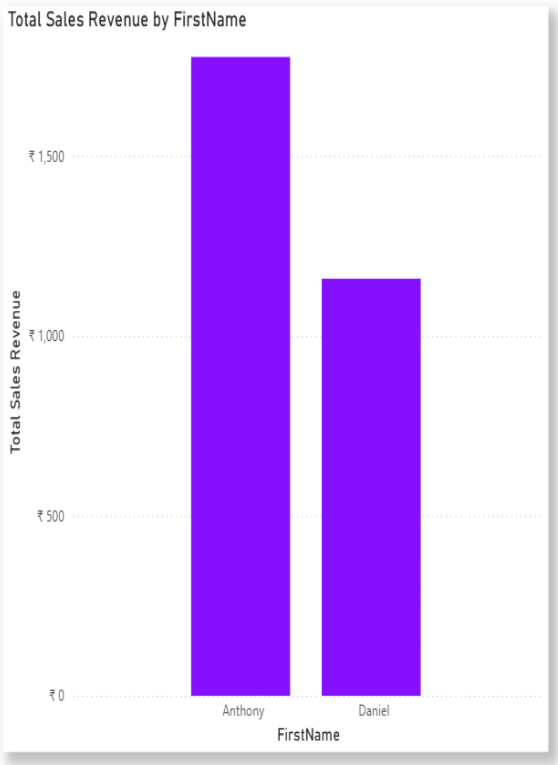
Executive Overview



Sales Performance



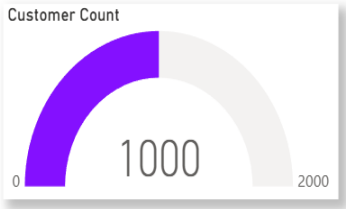


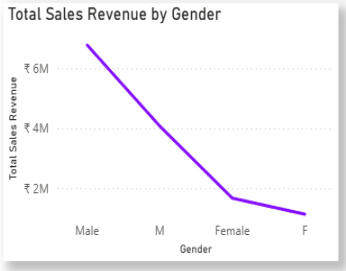


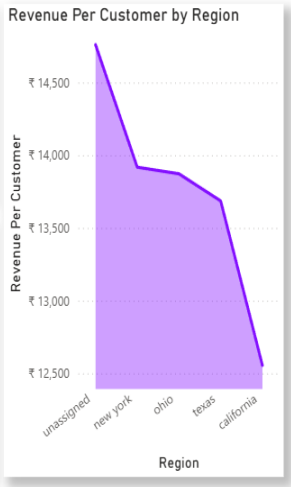
CUSTOMER INTELLIGENCE

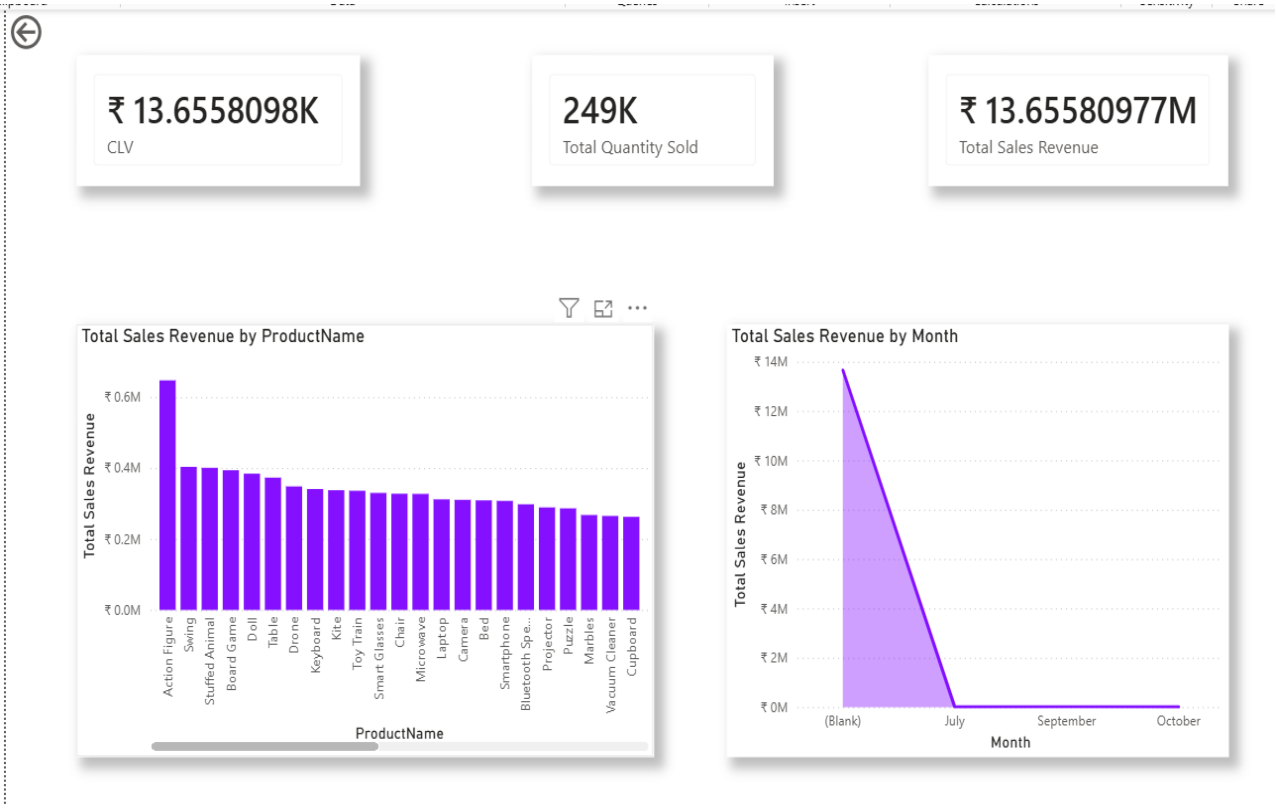
₹ 1.37K	₹ 13.66K	999	1.00
ATV	CLV	Repeat Purchase Customers	Repeat Purchase Rate

CustomerID	Total Sales Revenue	Total Quantity Sold	CLV
517	₹ 26,160.66	788	₹ 26,160.66
965	₹ 88,483.71	774	₹ 88,483.71
63	₹ 22,894.52	731	₹ 22,894.52
468	₹ 20,280.91	721	₹ 20,280.91
835	₹ 22,793.11	700	₹ 22,793.11
435	₹ 99,826.71	693	₹ 99,826.71
695	₹ 25,925.24	691	₹ 25,925.24
654	₹ 25,991.53	685	₹ 25,991.53
256	₹ 58,788.61	669	₹ 58,788.61
261	₹ 19,860.67	652	₹ 19,860.67
127	₹ 24,225.75	648	₹ 24,225.75
133	₹ 19,732.21	642	₹ 19,732.21
640	₹ 79,226.89	636	₹ 79,226.89
558	₹ 18,394.08	630	₹ 18,394.08
555	₹ 96,522.37	626	₹ 96,522.37
199	₹ 24,028.59	624	₹ 24,028.59
735	₹ 21,524.29	623	₹ 21,524.29
909	₹ 92,069.23	622	₹ 92,069.23
Total	**₹ 1,36,55,809.77**	**249465**	**₹ 13,655.8098**








₹ 13.6558098K
CLV
249K
Total Quantity Sold
₹ 13.65580977M
Total Sales Revenue
Total Sales Revenue by ProductName
Total Sales Revenue
₹ 0.6M
₹ 0.4M
₹ 0.2M
₹ 0.0M
Action Figure
Swing
Stuffed Animal
Board Game
Doll
Table
Drone
Keyboard
Kite
Toy Train
Smart Glasses
Chair
Microwave
Laptop
Camera
Bed
Smartphone
Bluetooth Spe...
Projector
Puzzle
Marbles
Vacuum Cleaner
Cupboard
ProductName
Total Sales Revenue by Month
Total Sales Revenue
₹ 14M
₹ 12M
₹ 10M
₹ 8M
₹ 6M
₹ 4M
₹ 2M
₹ 0M
(Blank)
July
September
October
Month


PRODUCT &
CATEGORY INSIGHTS
249K
Total Quantity Sold
₹ 13.66M
Total Product Sales
Product Share %
1.00
0.00
2.00
Total Sales Revenue by Category
Total Sales Revenue by Category
Total Sales Revenue
₹ 6M
₹ 4M
₹ 2M
₹ 0M
toys
electronics
furniture
home appliances
Category
Top 10 products
Action Figure
Swing
Stuffed Animal
Board Game
Doll
Table
Drone
Keyboard
ProductName
₹ 0.0M
₹ 0.5M
Total Sales Revenue
ATV
₹ 1.37K
₹ 0K
₹ 2.73K
Sum of Quantity and Sum of SalesAmount by ProductName
Sum of SalesAmount
0.6M
0.4M
0.2M
4500
5000
5500
Sum of Quantity

6. Challenges Faced and Resolutions

S.No	Challenge	Resolution
1	CustomerID format mismatch	Extracted numeric portion
2	Duplicate primary keys	Removed duplicates
3	Region inconsistencies	Standardized values
4	Foreign key drop errors	Correct table load order
5	Blank growth metrics	Corrected time-intelligence logic
6	Empty DimDate table	Generated proper date dimension

7. Project Outcome

The project delivered:

- ✓ A structured Data Warehouse
- ✓ Clean and validated retail dataset
- ✓ Business-aligned KPI framework
- ✓ Executive-ready Power BI dashboard

The solution enables:

- ✓ Performance monitoring
- ✓ Customer behavior analysis
- ✓ Product optimization decisions
- ✓ Regional sales tracking
- ✓ Data-driven strategic planning

8. Conclusion

This project demonstrates the complete lifecycle of a Business Intelligence solution:

- Data Engineering (ETL & SQL)
- Data Modeling (Star Schema Design)
- Analytical Computation (DAX KPIs)
- Visualization (Power BI Dashboard)

Raw retail data was successfully transformed into a centralized, structured, and actionable analytics system.