Customer 360 Data Integration

Github Link: [ani97/Customer\_360\_Data\_Integration](https://github.com/ani97/Customer_360_Data_Integration)

**Objective:**

The goal of this project is to build a **Customer 360 view** by integrating data from multiple sources, including online transactions, in-store purchases, customer service interactions, and loyalty programs. This unified data approach empowers business users to gain valuable insights into customer behavior across different touchpoints, enabling better decision-making.

**Architecture Summary:**

A diagram of a computer

AI-generated content may be incorrect.

1. **Data Sources:**
   * Multiple external sources such as:
     + **APIs** (for real-time transactional or service data)
     + **CSV files** (containing historical or batch data for transactions, customers, products, etc.)
2. **Bronze Layer (Raw Data):**
   * **Storage:** Azure Data Lake Storage (ADLS)
   * **Ingestion:** Data from APIs and CSV files is ingested and stored in raw format.
   * **Purpose:** Preserve the original structure for traceability and auditing.
3. **Silver Layer :**
   * **Technology:** Azure Synapse Analytics
   * **Method:** Data from the Bronze layer is transformed and cleaned using **CETAS** (CREATE EXTERNAL TABLE AS SELECT).
   * **Purpose:** Standardize and cleanse data, applying schema enforcement and deduplication.
4. **Gold Layer (Business-Ready Data):**
   * **Storage:** Azure SQL Database
   * **Loading:** Processed, curated data from the Silver layer is aggregated and loaded into SQL tables for analytics-ready consumption.
   * **Purpose:** Store refined data models (fact and dimension tables) supporting business metrics and reporting.
5. **Analysis Layer:**
   * **SQL Views:** Created over Gold layer tables to generate insights such as:
     + **Average Order Value (AOV)**
     + **Customer Segmentation** (by spend, frequency, loyalty)
     + **Peak Transaction Times** (for online and in-store)
     + **Customer Service Metrics** (interactions, resolution rates)
6. **Visualization:**
   * **Power BI:** Connects directly to the Gold Layer for data visualization and dashboarding.
   * **Fabric Workspace:** Power BI reports are published to **Microsoft Fabric** for collaboration and distribution.

# Steps:

## Set Up ADLS and Synapse Workspace and ADLS

1. Search for Storage accounts->Create and checkmark Hierarchical namespace option.

A screenshot of a computer

AI-generated content may be incorrect.

1. Create container and add two directory **bronze** and **silver**.  **Bronze** will hold raw data and silver layer will hold clean and processed data from serverless pool.

A white background with text

AI-generated content may be incorrect.

1. Search for Synapse Workspace and create a workspace and link the container.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

## Load Raw Data in Bronze Folder

1. We will download dataset from [Kaggle](https://www.kaggle.com/datasets/varunkumari/customer-360-data) on local machine.
2. We will directly upload files in bronze layer.

A screenshot of a computer

AI-generated content may be incorrect.

## Transform and clean the data in Synapse Serverless Pool

1. Open synapse workspace.
2. Go to developer tab->Click on + button->SQL script

A screenshot of a computer

AI-generated content may be incorrect.

1. Create database that will hold our external tables.

A screenshot of a computer

AI-generated content may be incorrect.

1. Create a new script and create a external data source pointing to the raw container. Also, create external file format that tells synapse how to read the csv file.

A screenshot of a computer

AI-generated content may be incorrect.

1. Create external data source pointing to silver layer and create external file format so Synapse can write the cleaned data in parquet format.

A screenshot of a computer program

AI-generated content may be incorrect.

1. We will now create separate SQL script for each csv file.
2. Click on + button and name it Agents\_csv.
3. We are **creating an external table** that reads the **Agents.csv file from the Bronze (raw) layer** in ADLS, allowing us to **query the data directly without importing it into Synapse**. This creates a **metadata layer** that points to the file, enabling SQL queries on raw data.

A screenshot of a computer

AI-generated content may be incorrect.

1. We use CETAS (CREATE EXTERNAL TABLE AS SELECT) to transform the raw data by removing rows with null AgentID and filling other null fields with default values (like 'Unknown' or 'NA'). The cleaned data is saved in the Silver layer as Parquet files for optimized querying.

(**Writes query results as new files** (e.g., Parquet) to a **target location in ADLS** (like Silver layer).

**Creates metadata** in Synapse **for those new files**, allowing you to query them.)

A screenshot of a computer code

AI-generated content may be incorrect.

1. We will create external tables for all csv files and then write the cleaned data to silver layer using CETAS.

Customers:

A screenshot of a computer

AI-generated content may be incorrect.

CustomersServiceInteractions:

A screenshot of a computer

AI-generated content may be incorrect.

Instore Transactions:

A screenshot of a computer

AI-generated content may be incorrect.

Loyalty Accounts:

A screenshot of a computer

AI-generated content may be incorrect.

Loyalty Transactions:

A screenshot of a computer

AI-generated content may be incorrect.

Online Transactions:

A screenshot of a computer

AI-generated content may be incorrect.

Products:

A screenshot of a computer

AI-generated content may be incorrect.

Stores:

A screenshot of a computer

AI-generated content may be incorrect.

## Create Table schema in Azure SQL Database

1. On azure portal, search for Azure SQL database and create a new database. Set the login user and password.
2. Open the SQL database and create schema for our tables.

CREATE TABLE Customers (

CustomerID INT PRIMARY KEY,

Name VARCHAR(100),

Email VARCHAR(100),

Address VARCHAR(255)

);

CREATE TABLE Products (

ProductID INT PRIMARY KEY,

Name VARCHAR(100),

Category VARCHAR(50),

Price DECIMAL(10, 2)

);

CREATE TABLE OnlineTransactions (

OrderID INT PRIMARY KEY,

CustomerID INT,

ProductID INT,

DateTime DATETIME,

PaymentMethod VARCHAR(50),

Amount DECIMAL(10, 2),

Status VARCHAR(20),

FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID),

FOREIGN KEY (ProductID) REFERENCES Products(ProductID)

);

CREATE TABLE Stores (

StoreID INT PRIMARY KEY,

Location VARCHAR(100),

Manager VARCHAR(100),

OpenHours VARCHAR(50)

);

CREATE TABLE InStoreTransactions (

TransactionID INT PRIMARY KEY,

CustomerID INT,

StoreID INT,

DateTime DATETIME,

Amount DECIMAL(10, 2),

PaymentMethod VARCHAR(50),

FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID),

FOREIGN KEY (StoreID) REFERENCES Stores(StoreID)

);

CREATE TABLE Agents (

AgentID INT PRIMARY KEY,

Name VARCHAR(100),

Department VARCHAR(50),

Shift VARCHAR(50)

);

CREATE TABLE CustomerServiceInteractions (

InteractionID INT PRIMARY KEY,

CustomerID INT,

DateTime DATETIME,

AgentID INT,

IssueType VARCHAR(50),

ResolutionStatus VARCHAR(50),

FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID),

FOREIGN KEY (AgentID) REFERENCES Agents(AgentID)

);

CREATE TABLE LoyaltyAccounts (

LoyaltyID INT PRIMARY KEY,

CustomerID INT,

PointsEarned INT,

TierLevel VARCHAR(20),

JoinDate DATE,

FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)

);

CREATE TABLE LoyaltyTransactions (

LoyaltyID INT,

DateTime DATETIME,

PointsChange INT,

Reason VARCHAR(100),

PRIMARY KEY (LoyaltyID, DateTime),

FOREIGN KEY (LoyaltyID) REFERENCES LoyaltyAccounts(LoyaltyID)

);

## Create Pipeline to Copy data from ADLS into SQL Database

1. Click on integrate tab->Click on + to create a new pipeline.

A screenshot of a computer

AI-generated content may be incorrect.

1. Dra a foreach activity

A screenshot of a computer

AI-generated content may be incorrect.

1. Click on settings tab-> in items create a json schema. It will help to loop over different folders and then load the parquet file in corresponding table.

@json('[

    {"folder": "silver/Customers", "table": "Customers"},

    {"folder": "silver/Products", "table": "Products"},

    {"folder": "silver/Stores", "table": "Stores"},

    {"folder": "silver/Agents", "table": "Agents"},

    {"folder": "silver/LoyaltyAccounts", "table": "LoyaltyAccounts"},

    {"folder": "silver/LoyaltyTransactions", "table": "LoyaltyTransactions"},

    {"folder": "silver/OnlineTransactions", "table": "OnlineTransactions"},

    {"folder": "silver/InStoreTransactions", "table": "InStoreTransactions"},

    {"folder": "silver/CustomerServiceInteractions", "table": "CustomerServiceInteractions"}

]')

1. Click on sequential as we want to load table one by one due to foreign key constraints.

A screenshot of a computer

AI-generated content may be incorrect.

1. In activities->Click on pencil and inside foreach drag a copy activity.
2. In the source side->Create a dataset pointing to the silver folder.

A screenshot of a computer

AI-generated content may be incorrect.

1. Click on wildcard character->In directory pass value coming from foreach so it will point to that particular folder. @item().folder

And in filename pass \*.parquet so it picks every file ending with it.

A screenshot of a computer

AI-generated content may be incorrect.

1. In the sink side, create a dataset pointing to the azure sql database. Create a parameter tablename so we can pass table names dynamically.

A screenshot of a computer

AI-generated content may be incorrect.

1. Pass the value from foreach in the parameter @item().table

A screenshot of a computer

AI-generated content may be incorrect.

1. Click on debug.
2. Go to Database and verify.

A screenshot of a computer

AI-generated content may be incorrect.

## Create Views to Analyze Data

### View 1 - Average Order Value (AOV)

#### SUM(Amount) / COUNT(OrderID) per product, category, and location.

Create view view1

as

SELECT COUNT(c.OrderID) AS total\_orders, d.Category, b.Location, SUM(a.Amount + c.Amount) AS total\_amount

FROM dbo.InStoreTransactions AS a

INNER JOIN dbo.Stores AS b ON a.StoreID = b.StoreID

INNER JOIN dbo.OnlineTransactions AS c ON a.CustomerID = c.CustomerID

INNER JOIN dbo.Products AS d ON c.ProductID = d.ProductID

GROUP BY d.Category, b.Location

A screenshot of a computer

AI-generated content may be incorrect.

### View 2 - for Segment customers based on total spend, purchase frequency, and loyalty tier (**LoyaltyAccounts.TierLevel**).

#### Example: "High-Value Customers" (Top 10% spenders), "One-Time Buyers," "Loyalty Champions."

create view view\_2

as

WITH online\_instore AS (

SELECT a.CustomerID, COUNT(TransactionID) AS Purchase\_freq, SUM(b.Amount) AS Amount

FROM Customers a

LEFT JOIN InStoreTransactions b ON a.CustomerID = b.CustomerID

GROUP BY a.CustomerID

UNION ALL

SELECT a.CustomerID, COUNT(OrderID) AS Purchase\_freq, SUM(b.Amount) AS Amount

FROM Customers a

LEFT JOIN OnlineTransactions b ON a.CustomerID = b.CustomerID

GROUP BY a.CustomerID

),

total\_spend\_freq AS (

SELECT CustomerID, SUM(Purchase\_freq) AS TotalPurchaseFrequency, SUM(Amount) AS TotalSpend

FROM online\_instore

GROUP BY CustomerID

),

loyalty\_tier AS (

SELECT CustomerID,

CASE WHEN MIN(CASE WHEN TierLevel = 'Platinum' THEN 1 WHEN TierLevel = 'Gold' THEN 2 WHEN TierLevel = 'Silver' THEN 3 WHEN TierLevel = 'Bronze' THEN 4 ELSE 5 END) = 1 THEN 'Platinum'

WHEN MIN(CASE WHEN TierLevel = 'Platinum' THEN 1 WHEN TierLevel = 'Gold' THEN 2 WHEN TierLevel = 'Silver' THEN 3 WHEN TierLevel = 'Bronze' THEN 4 ELSE 5 END) = 2 THEN 'Gold'

WHEN MIN(CASE WHEN TierLevel = 'Platinum' THEN 1 WHEN TierLevel = 'Gold' THEN 2 WHEN TierLevel = 'Silver' THEN 3 WHEN TierLevel = 'Bronze' THEN 4 ELSE 5 END) = 3 THEN 'Silver'

WHEN MIN(CASE WHEN TierLevel = 'Platinum' THEN 1 WHEN TierLevel = 'Gold' THEN 2 WHEN TierLevel = 'Silver' THEN 3 WHEN TierLevel = 'Bronze' THEN 4 ELSE 5 END) = 4 THEN 'Bronze'

ELSE NULL END AS TierLevel

FROM LoyaltyAccounts

GROUP BY CustomerID

),

spend\_ranking AS (

SELECT tsf.CustomerID, tsf.TotalPurchaseFrequency, tsf.TotalSpend, lt.TierLevel, PERCENT\_RANK() OVER (ORDER BY tsf.TotalSpend DESC) AS SpendPercentile

FROM total\_spend\_freq tsf

LEFT JOIN loyalty\_tier lt ON tsf.CustomerID = lt.CustomerID

)

SELECT CustomerID, TotalSpend, TotalPurchaseFrequency, TierLevel,

CASE WHEN SpendPercentile >= 0.9 THEN 'High-Value Customer'

WHEN TotalPurchaseFrequency = 1 THEN 'One-Time Buyer'

WHEN TierLevel IS NOT NULL THEN 'Loyalty Champion'

ELSE 'Regular Customer' END AS Segment

FROM spend\_ranking;

A screenshot of a computer

AI-generated content may be incorrect.

### View 3 - for Analyze **DateTime** to find peak days and times in-store vs. online.

**INSTORE:**

CREATE VIEW instore\_view3

as

SELECT DATENAME(weekday, DateTime) AS DayOfWeek,

DATEPART(hour, DateTime) AS HourOfDay,

COUNT(\*) AS TotalTransactions

FROM dbo.InStoreTransactions

GROUP BY DATENAME(weekday, DateTime), DATEPART(hour, DateTime)

A screenshot of a computer

AI-generated content may be incorrect.

**Online:**

create view online\_view3

as

SELECT DATENAME(weekday, DateTime) AS DayOfWeek,

DATEPART(hour, DateTime) AS HourOfDay,

COUNT(\*) AS TotalTransactions

FROM dbo.OnlineTransactions

GROUP BY DATENAME(weekday, DateTime), DATEPART(hour, DateTime)

A screenshot of a computer

AI-generated content may be incorrect.

### View 4 - for Number of interactions and resolution success rates per agent (**ResolutionStatus**).

create view view4

as

SELECT AgentID, COUNT(InteractionID) AS total\_interactions,

SUM(CASE WHEN ResolutionStatus = 'Resolved' THEN 1 ELSE 0 END) AS resolved,

CAST(ROUND(CASE WHEN SUM(CASE WHEN ResolutionStatus = 'Resolved' THEN 1 ELSE 0 END) = 0 THEN 0 ELSE SUM(CASE WHEN ResolutionStatus = 'Resolved' THEN 1 ELSE 0 END) \* 100.0 / COUNT(InteractionID) END, 2)

AS DECIMAL(5, 2)) AS success\_ratio

FROM dbo.CustomerServiceInteractions

GROUP BY AgentID

A screenshot of a computer

AI-generated content may be incorrect.

### Connect Azure SQL with PowerBI

1. Open PowerBI
2. Click on Get Data and look for Azure SQL Server

A screenshot of a computer

AI-generated content may be incorrect.

1. Click on connect.
2. Paste the server name

A screenshot of a computer

AI-generated content may be incorrect.

1. Click on ok and enter the credentials

A screenshot of a computer

AI-generated content may be incorrect.

1. Select the Database.
2. Select the tables.

A screenshot of a computer

AI-generated content may be incorrect.

1. Go to model view and check connections.

A screenshot of a computer

AI-generated content may be incorrect.

1. Create Visualizations.

A screenshot of a graph

AI-generated content may be incorrect.

1. Click on Publish and select your Fabric Workspace.

A screenshot of a computer

AI-generated content may be incorrect.

1. Select the workspace

A screenshot of a computer

AI-generated content may be incorrect.