ATLIQ
MARTS

OPTIMIZING SUPPLY CHAIN PERFORMANCE FOR ATLIQ MART

END-TO-END DATA ENGINEERING PORJECT FOR SOLVING SERVICE LEVEL ISSUES AT ATLIQ MART

1-خالد شهام

2-بسنت ياسر

3-لبنى عيد

4-أحمد علاء

5-محمد صبري
,

Prepared for

ATLIQ MART STAKEHOLDERS

Problem Statement

AtliQ Mart, a growing FMCG manufacturer, is experiencing service level issues across three cities: Surat, Ahmedabad, and Vadodara.

Key Concerns:

- Delivery Delays: Some key customers did not renew their annual contracts due to **consistent delays in product deliveries**.
- Incomplete Orders: Orders were either not delivered on time or not delivered in full, leading to customer dissatisfaction.

Business Goal:

- AtliQ Mart's management aims to track and improve On-Time and In-Full Delivery Service Levels on a daily basis.

✦ Metrics to track:

- On-time Delivery (OT) %
- In-full Delivery (IF) %
- On-Time In-Full (OTIF) %

Project Overview

✦ Objective:

- Address service level issues at AtliQ Mart by predicting late and incomplete deliveries.
- Track key metrics: On-time Delivery (OT%), In-full Delivery (IF%), and OTIF% (On Time In Full).

✦ Key Deliverables:

- A well-designed SQL database schema and populated database.
- Integrated Azure Data services setup.
- Deployed machine learning model.
- Final report and presentation.

✦ Technologies Used:

- SQL Server, Python (Pandas,), azure.

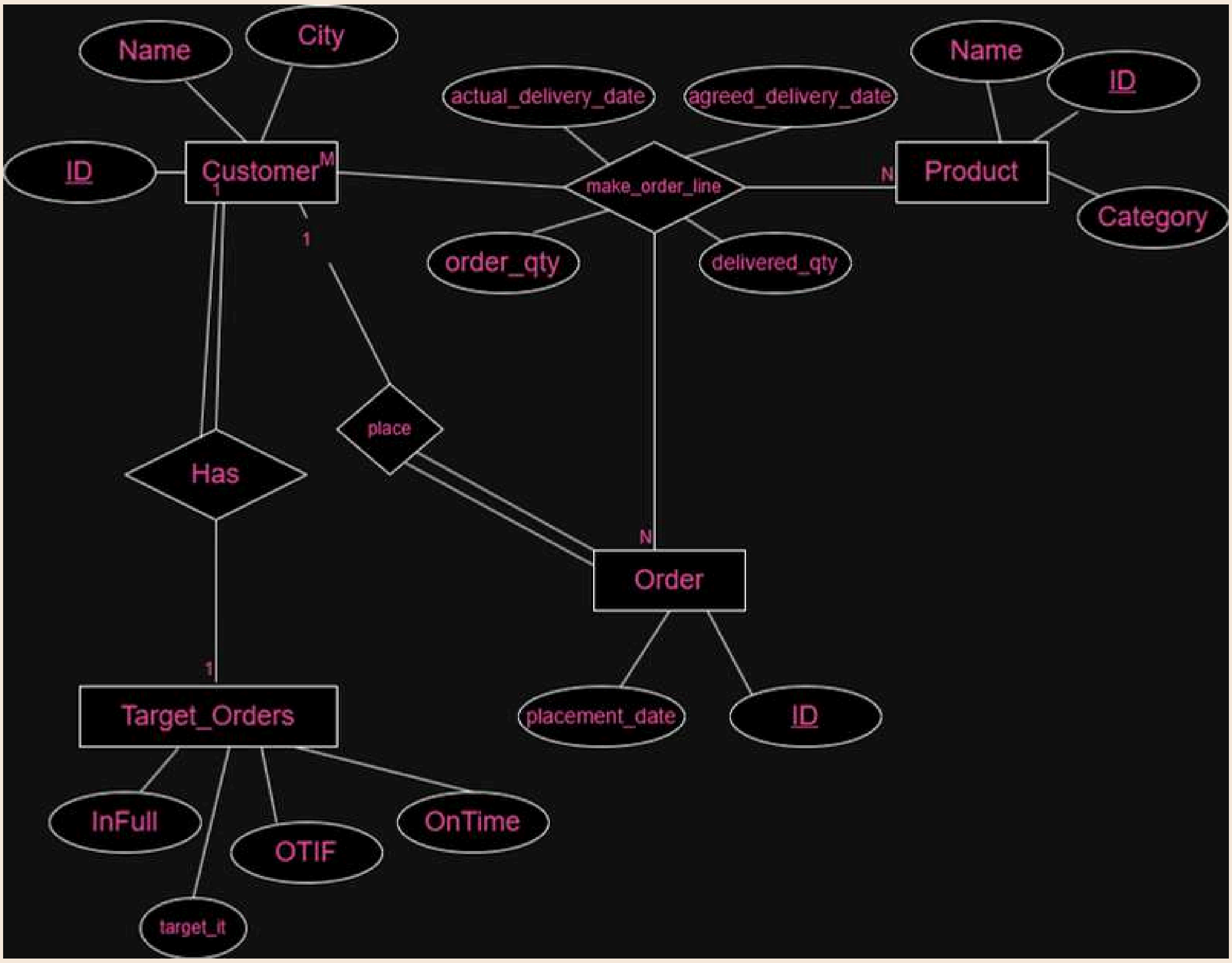
01 - Entity Relationship Diagram (ERD) and Data Mapping

Objective:

- **Design a relational database schema to store and manage all relevant data for analyzing service level issues at AtliQ Mart.**

ERD Creation Tool

- **Drawio**



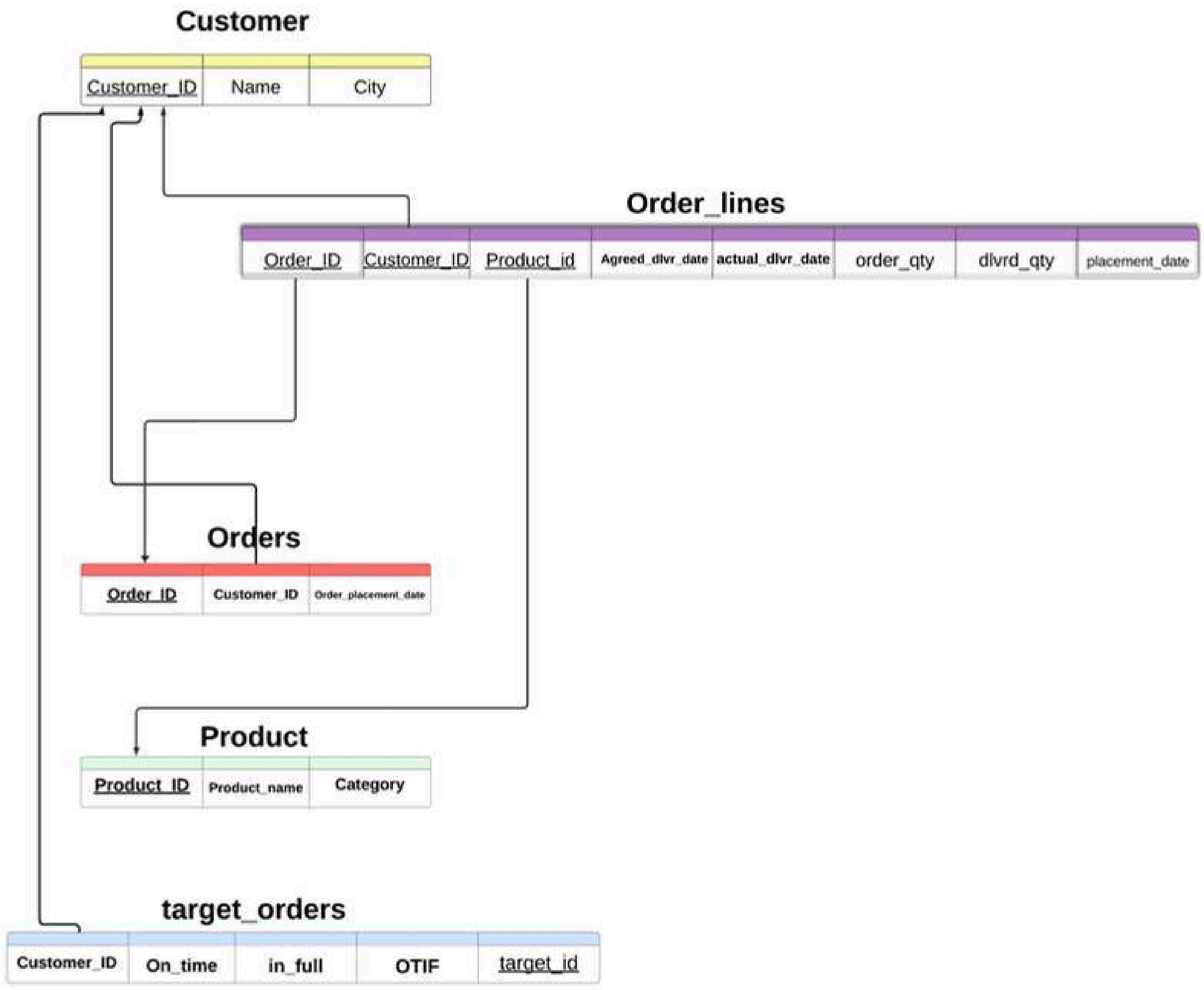
01 - Entity Relationship Diagram (ERD) and Data Mapping

Objective:

- **Design a relational database schema to store and manage all relevant data for analyzing service level issues at AtliQ Mart.**

Mapping creation tool:

- **lucidchart**



02 - Data Management and SQL Database Setup

✦ **Database Setup:**

- SQL Server schema design for tracking customer orders and deliveries.

✦ **Key Tables:**

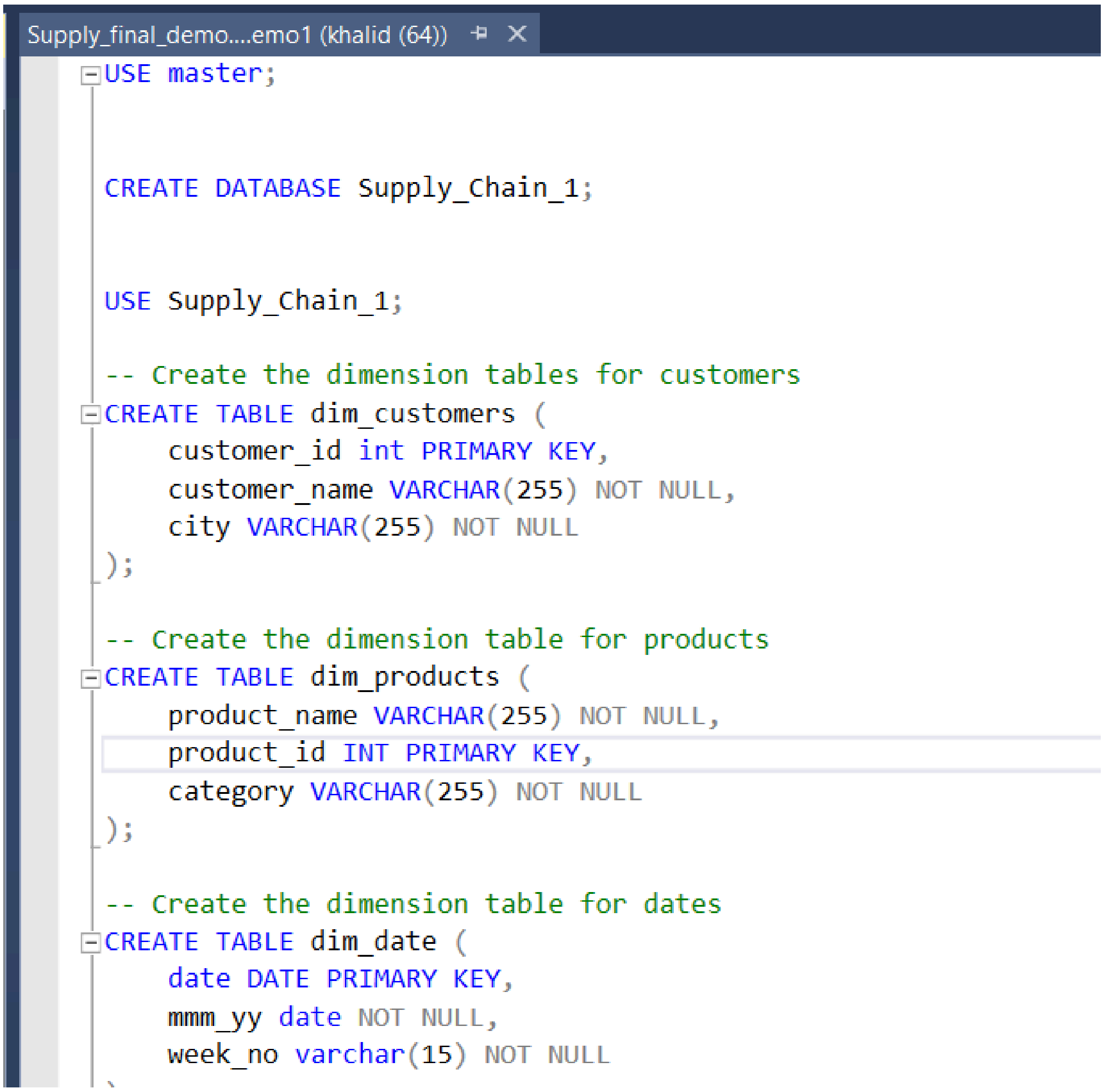
- dim_customers || dim_products || dim_date
- || dim_targets_orders || fact_order_lines || fact_orders_aggregate.

Tasks:

- Creating The Database.
- Data extraction from CSV files.
- Calculating new fields: on_time, in_full, otif.

02 - Data Management and SQL Database Setup

✦ Creating The Database:

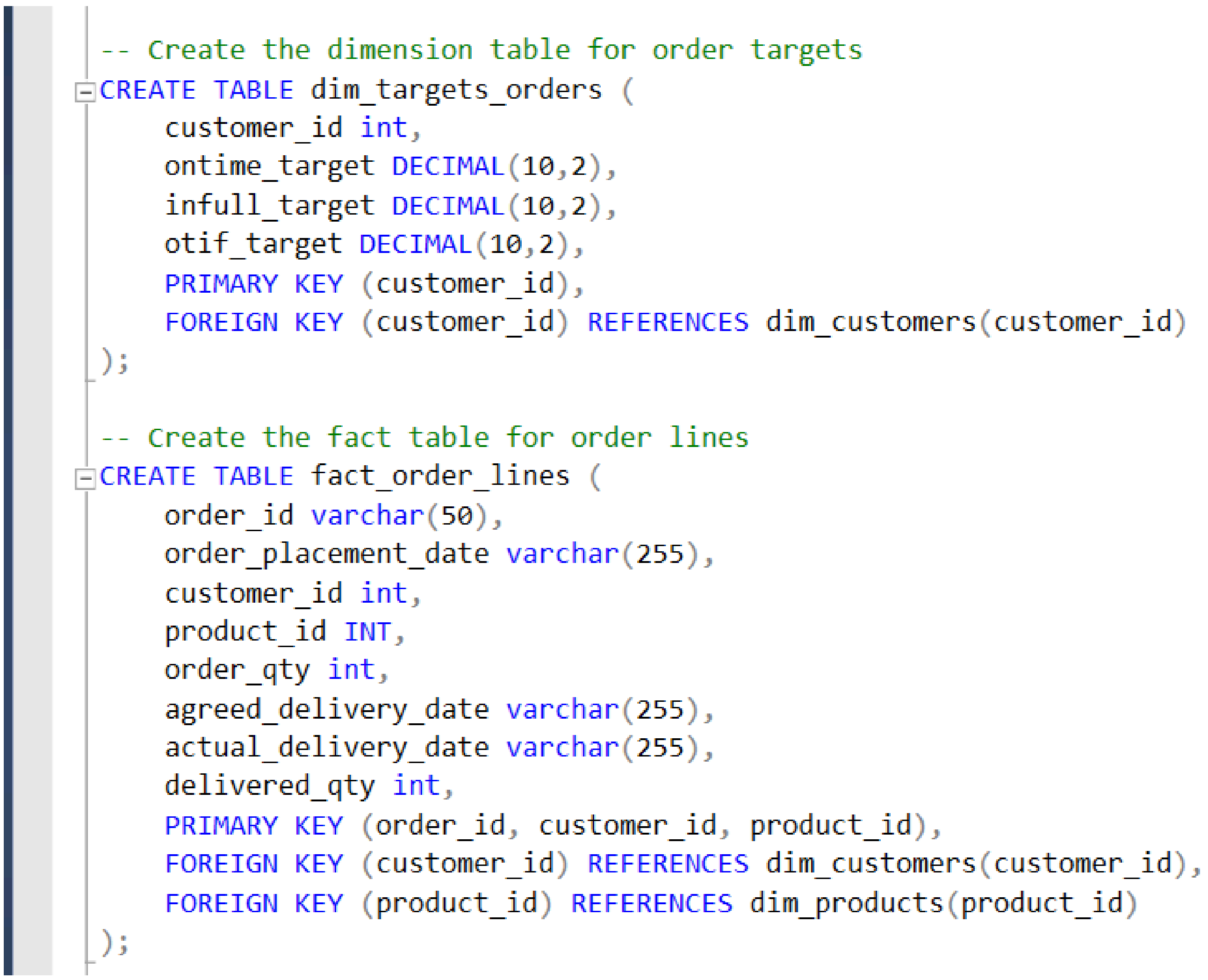


Supply_final_demo....emo1 (khalid (64))

```
USE master;


CREATE DATABASE Supply_Chain_1;


USE Supply_Chain_1;

-- Create the dimension tables for customers
CREATE TABLE dim_customers (
    customer_id int PRIMARY KEY,
    customer_name VARCHAR(255) NOT NULL,
    city VARCHAR(255) NOT NULL
);

-- Create the dimension table for products
CREATE TABLE dim_products (
    product_name VARCHAR(255) NOT NULL,
    product_id INT PRIMARY KEY,
    category VARCHAR(255) NOT NULL
);

-- Create the dimension table for dates
CREATE TABLE dim_date (
    date DATE PRIMARY KEY,
    mmm_yy date NOT NULL,
    week_no varchar(15) NOT NULL
```

02 - Data Management and SQL Database Setup

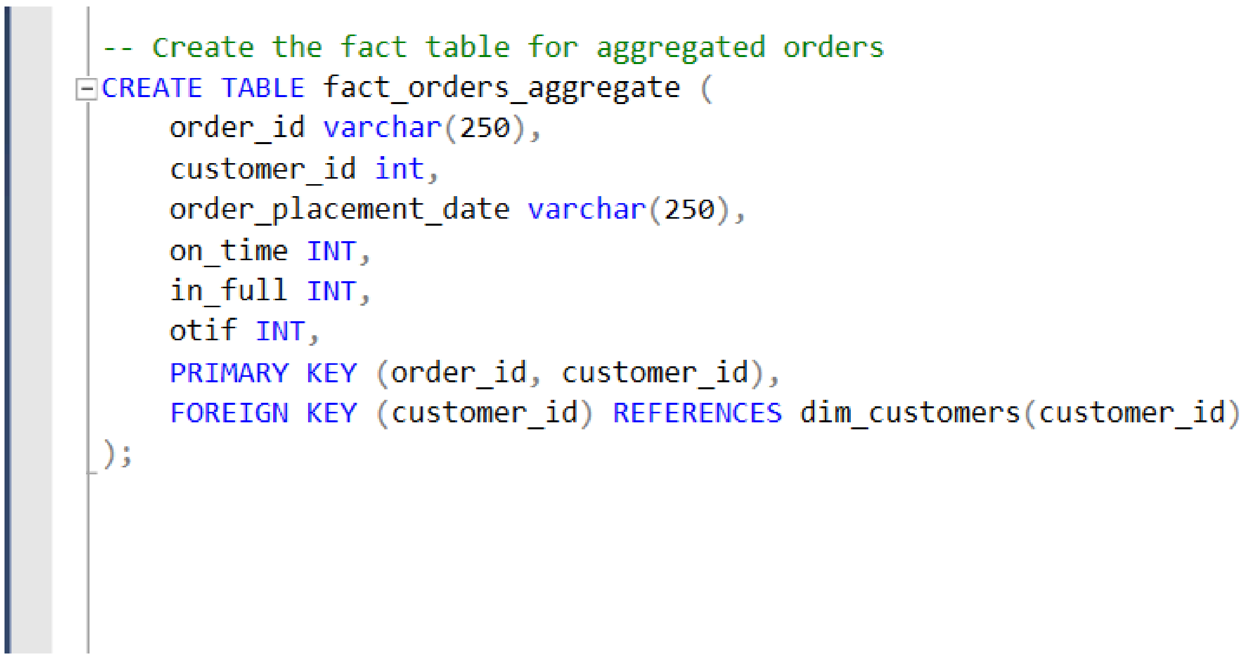
✦ Creating The Database:



```sql
-- Create the dimension table for order targets
CREATE TABLE dim_targets_orders (
    customer_id int,
    ontime_target DECIMAL(10,2),
    infull_target DECIMAL(10,2),
    otif_target DECIMAL(10,2),
    PRIMARY KEY (customer_id),
    FOREIGN KEY (customer_id) REFERENCES dim_customers(customer_id)
);

-- Create the fact table for order lines
CREATE TABLE fact_order_lines (
    order_id varchar(50),
    order_placement_date varchar(255),
    customer_id int,
    product_id INT,
    order_qty int,
    agreed_delivery_date varchar(255),
    actual_delivery_date varchar(255),
    delivered_qty int,
    PRIMARY KEY (order_id, customer_id, product_id),
    FOREIGN KEY (customer_id) REFERENCES dim_customers(customer_id),
    FOREIGN KEY (product_id) REFERENCES dim_products(product_id)
);
```

02 - Data Management and SQL Database Setup

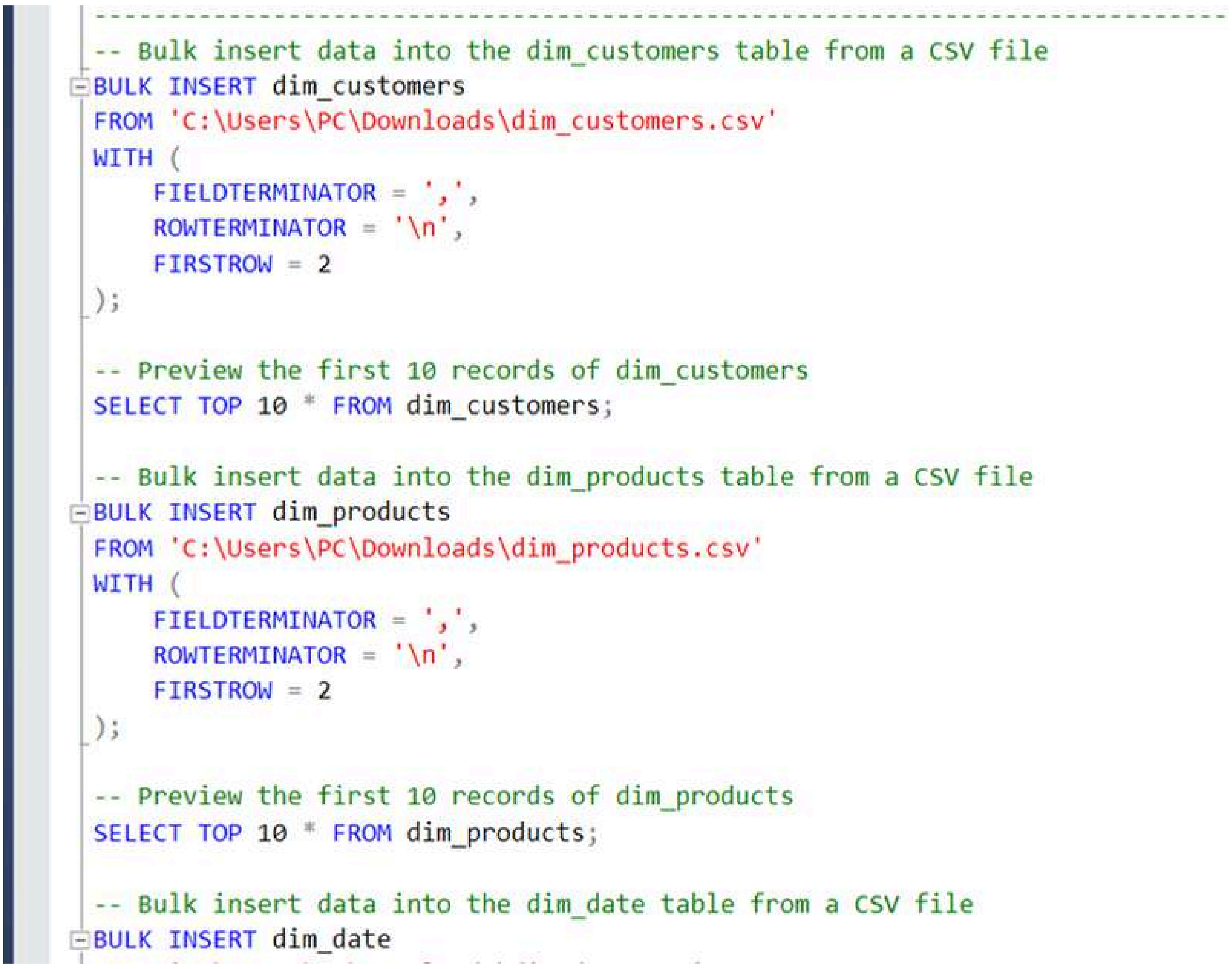
✦ Creating The Database:



```sql
-- Create the fact table for aggregated orders
CREATE TABLE fact_orders_aggregate (
    order_id varchar(250),
    customer_id int,
    order_placement_date varchar(250),
    on_time INT,
    in_full INT,
    otif INT,
    PRIMARY KEY (order_id, customer_id),
    FOREIGN KEY (customer_id) REFERENCES dim_customers(customer_id)
);
```

02 - Data Management and SQL Database Setup

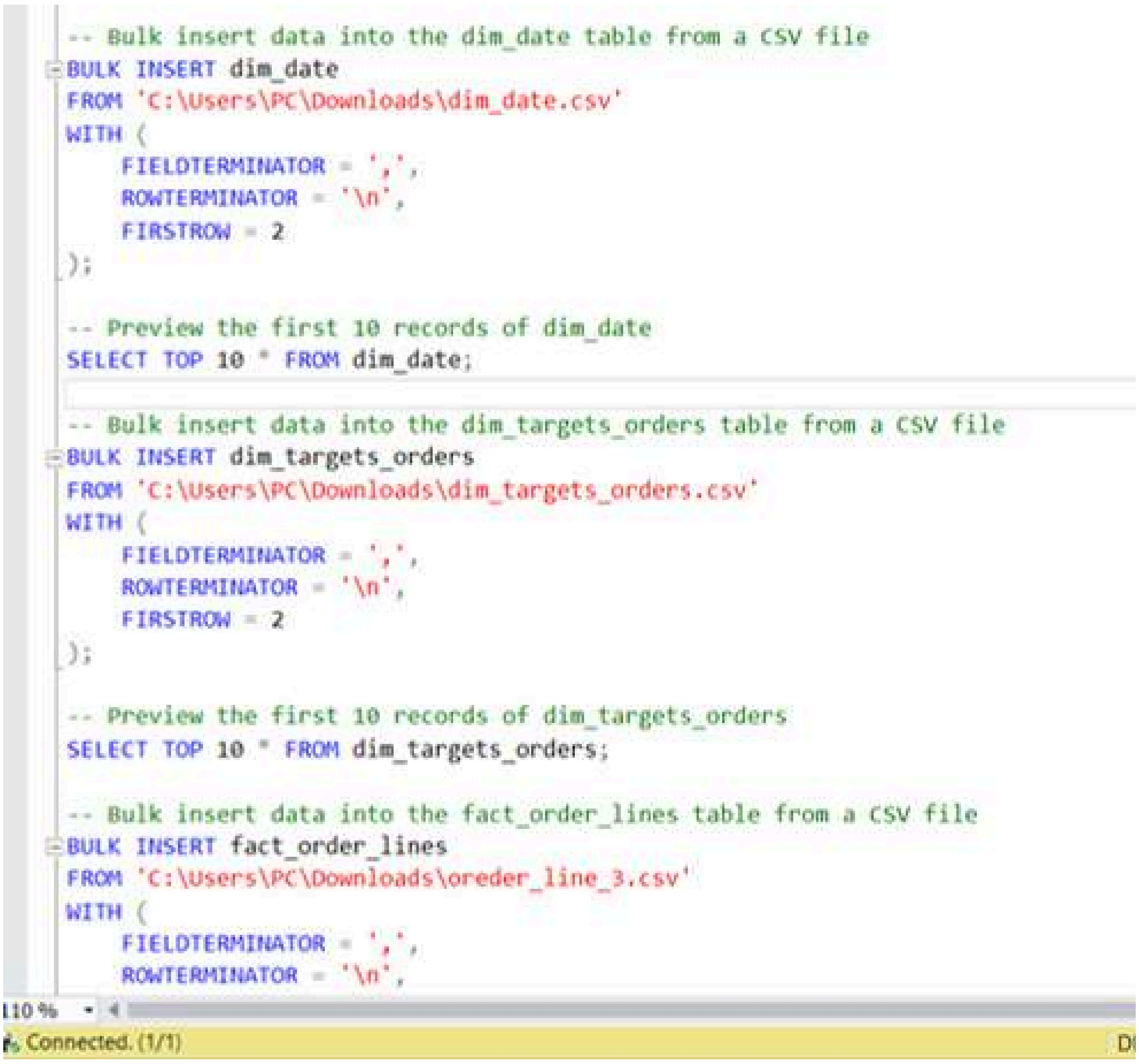
Data extraction from CSV files.



```
---------------------------------------------------------------------------
-- Bulk insert data into the dim_customers table from a CSV file
BULK INSERT dim_customers
FROM 'C:\Users\PC\Downloads\dim_customers.csv'
WITH (
    FIELDTERMINATOR = ',',
    ROWTERMINATOR = '\n',
    FIRSTROW = 2
);

-- Preview the first 10 records of dim_customers
SELECT TOP 10 * FROM dim_customers;

-- Bulk insert data into the dim_products table from a CSV file
BULK INSERT dim_products
FROM 'C:\Users\PC\Downloads\dim_products.csv'
WITH (
    FIELDTERMINATOR = ',',
    ROWTERMINATOR = '\n',
    FIRSTROW = 2
);

-- Preview the first 10 records of dim_products
SELECT TOP 10 * FROM dim_products;

-- Bulk insert data into the dim_date table from a CSV file
BULK INSERT dim_date
```

02 - Data Management and SQL Database Setup

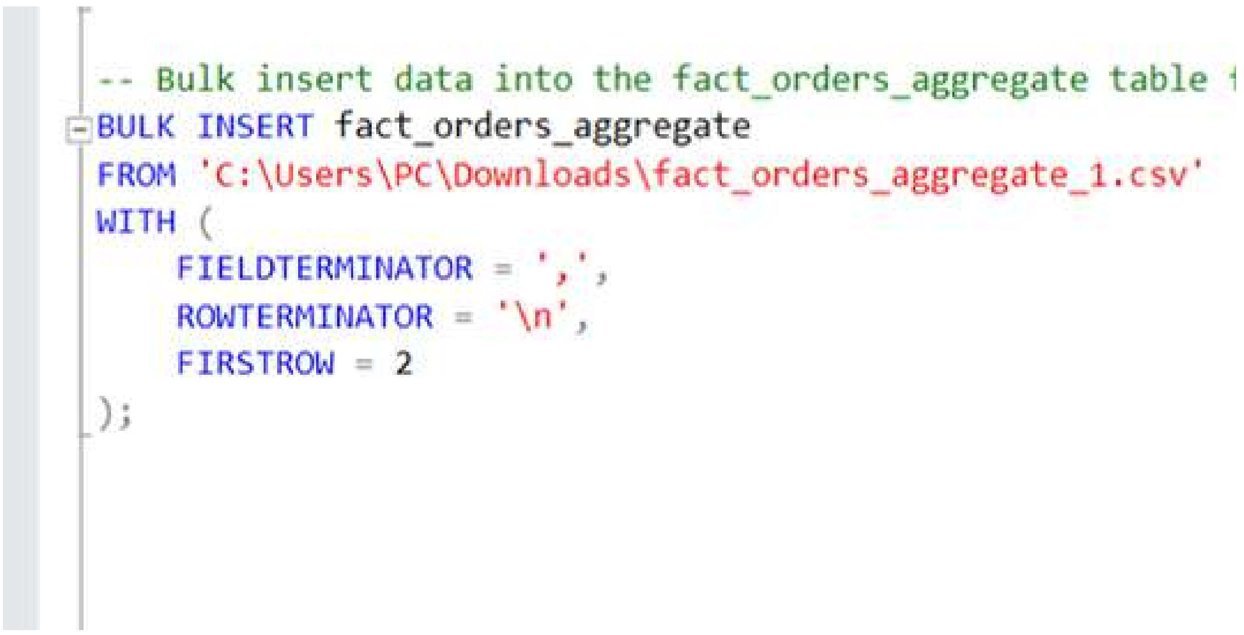
Data extraction from CSV files.



```
-- Bulk insert data into the dim_date table from a CSV file
BULK INSERT dim_date
FROM 'C:\Users\PC\Downloads\dim_date.csv'
WITH (
    FIELDTERMINATOR = ',',
    ROWTERMINATOR = '\n',
    FIRSTROW = 2
);

-- Preview the first 10 records of dim_date
SELECT TOP 10 * FROM dim_date;

-- Bulk insert data into the dim_targets_orders table from a CSV file
BULK INSERT dim_targets_orders
FROM 'C:\Users\PC\Downloads\dim_targets_orders.csv'
WITH (
    FIELDTERMINATOR = ',',
    ROWTERMINATOR = '\n',
    FIRSTROW = 2
);

-- Preview the first 10 records of dim_targets_orders
SELECT TOP 10 * FROM dim_targets_orders;

-- Bulk insert data into the fact_order_lines table from a CSV file
BULK INSERT fact_order_lines
FROM 'C:\Users\PC\Downloads\oreder_line_3.csv'
WITH (
    FIELDTERMINATOR = ',',
    ROWTERMINATOR = '\n',
```

02 - Data Management and SQL Database Setup

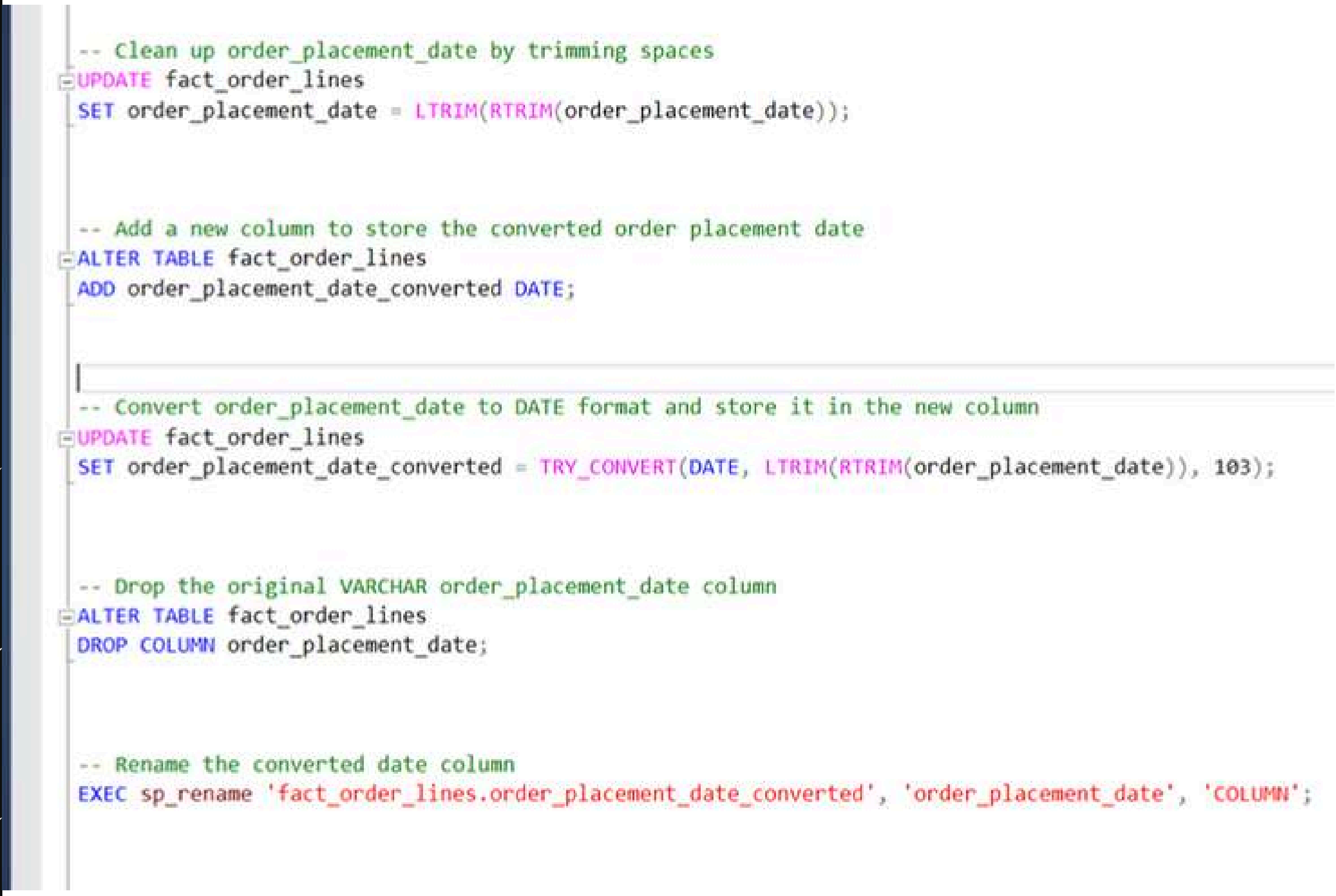
Data extraction from CSV files.



```
-- Bulk insert data into the fact_orders_aggregate table f
BULK INSERT fact_orders_aggregate
FROM 'C:\Users\PC\Downloads\fact_orders_aggregate_1.csv'
WITH (
    FIELDTERMINATOR = ',',
    ROWTERMINATOR = '\n',
    FIRSTROW = 2
);
```

02 - Data Management and SQL Database Setup

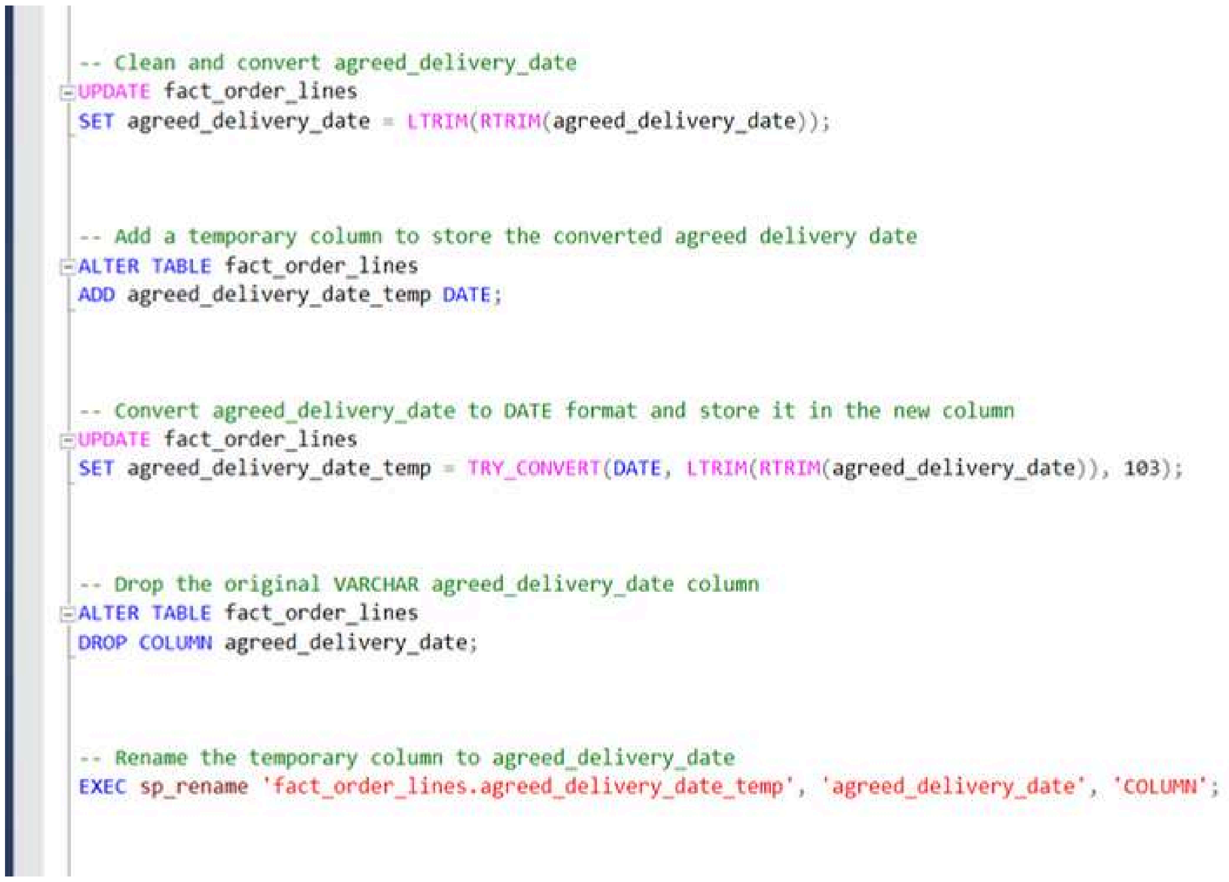
changing data type



```
-- Clean up order_placement_date by trimming spaces
UPDATE fact_order_lines
SET order_placement_date = LTRIM(RTRIM(order_placement_date));


-- Add a new column to store the converted order placement date
ALTER TABLE fact_order_lines
ADD order_placement_date_converted DATE;


-- Convert order_placement_date to DATE format and store it in the new column
UPDATE fact_order_lines
SET order_placement_date_converted = TRY_CONVERT(DATE, LTRIM(RTRIM(order_placement_date)), 103);


-- Drop the original VARCHAR order_placement_date column
ALTER TABLE fact_order_lines
DROP COLUMN order_placement_date;


-- Rename the converted date column
EXEC sp_rename 'fact_order_lines.order_placement_date_converted', 'order_placement_date', 'COLUMN';
```

02 - Data Management and SQL Database Setup

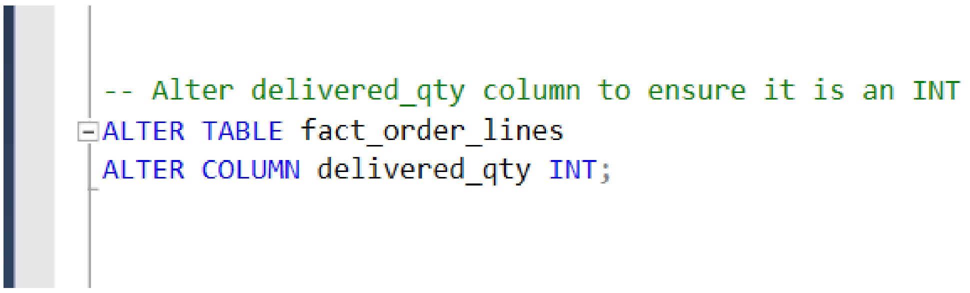
changing data type



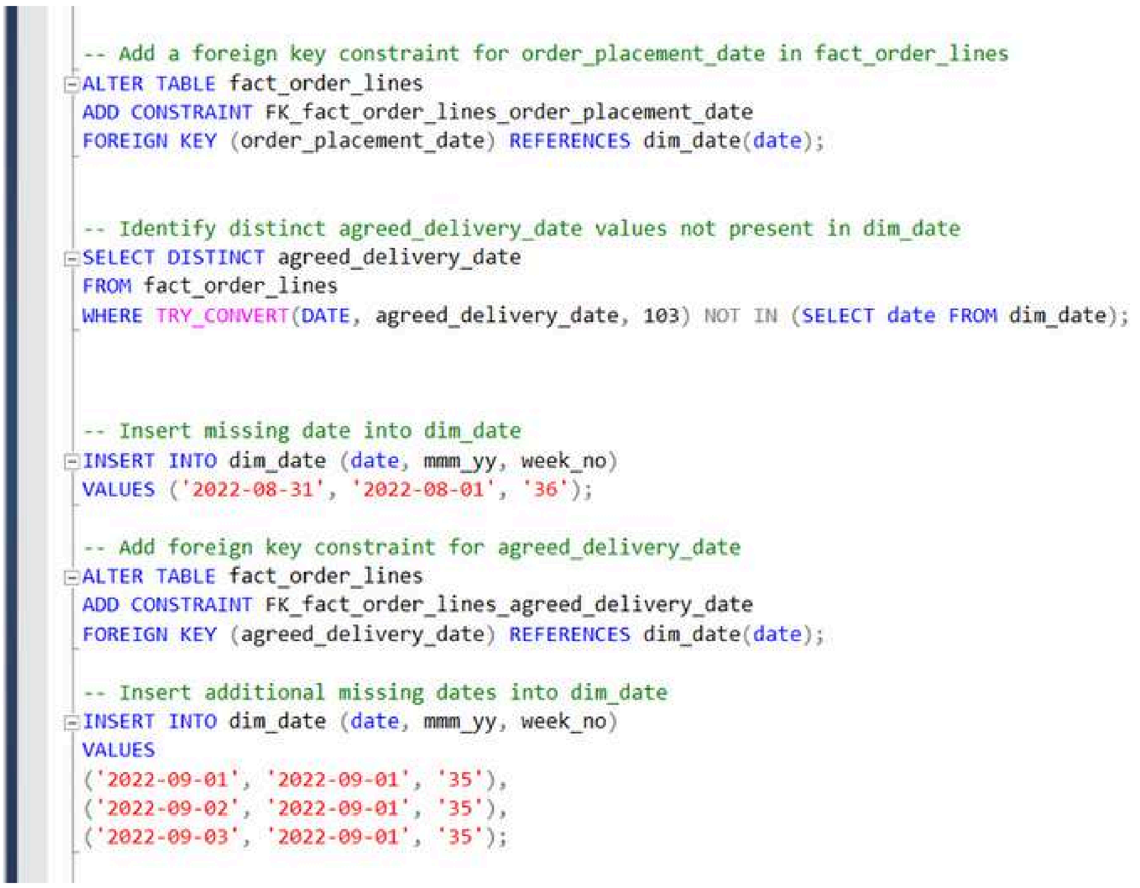
```
-- Clean and convert agreed_delivery_date
UPDATE fact_order_lines
SET agreed_delivery_date = LTRIM(RTRIM(agreed_delivery_date));


-- Add a temporary column to store the converted agreed delivery date
ALTER TABLE fact_order_lines
ADD agreed_delivery_date_temp DATE;


-- Convert agreed_delivery_date to DATE format and store it in the new column
UPDATE fact_order_lines
SET agreed_delivery_date_temp = TRY_CONVERT(DATE, LTRIM(RTRIM(agreed_delivery_date)), 103);


-- Drop the original VARCHAR agreed_delivery_date column
ALTER TABLE fact_order_lines
DROP COLUMN agreed_delivery_date;


-- Rename the temporary column to agreed_delivery_date
EXEC sp_rename 'fact_order_lines.agreed_delivery_date_temp', 'agreed_delivery_date', 'COLUMN';
```

02 - Data Management and SQL Database Setup

changing data type

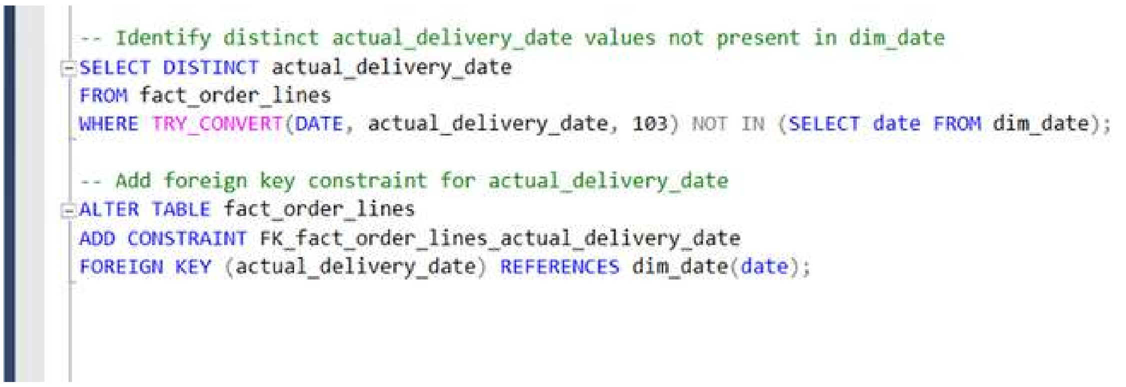


```sql
-- Alter delivered_qty column to ensure it is an INT
ALTER TABLE fact_order_lines
ALTER COLUMN delivered_qty INT;
```

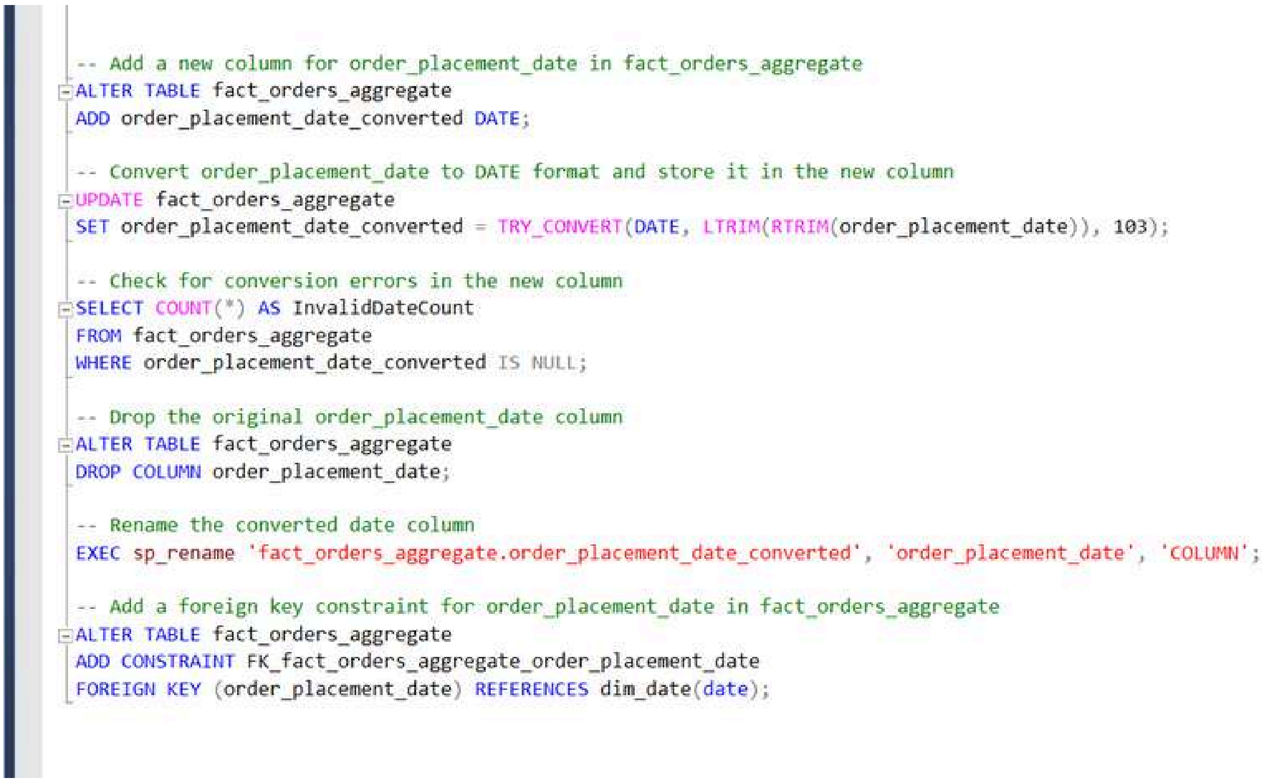


```sql
-- Add a foreign key constraint for order_placement_date in fact_order_lines
ALTER TABLE fact_order_lines
ADD CONSTRAINT FK_fact_order_lines_order_placement_date
FOREIGN KEY (order_placement_date) REFERENCES dim_date(date);


-- Identify distinct agreed_delivery_date values not present in dim_date
SELECT DISTINCT agreed_delivery_date
FROM fact_order_lines
WHERE TRY_CONVERT(DATE, agreed_delivery_date, 103) NOT IN (SELECT date FROM dim_date);


-- Insert missing date into dim_date
INSERT INTO dim_date (date, mmm_yy, week_no)
VALUES ('2022-08-31', '2022-08-01', '36');

-- Add foreign key constraint for agreed_delivery_date
ALTER TABLE fact_order_lines
ADD CONSTRAINT FK_fact_order_lines_agreed_delivery_date
FOREIGN KEY (agreed_delivery_date) REFERENCES dim_date(date);

-- Insert additional missing dates into dim_date
INSERT INTO dim_date (date, mmm_yy, week_no)
VALUES
('2022-09-01', '2022-09-01', '35'),
('2022-09-02', '2022-09-01', '35'),
('2022-09-03', '2022-09-01', '35');
```

02 - Data Management and SQL Database Setup

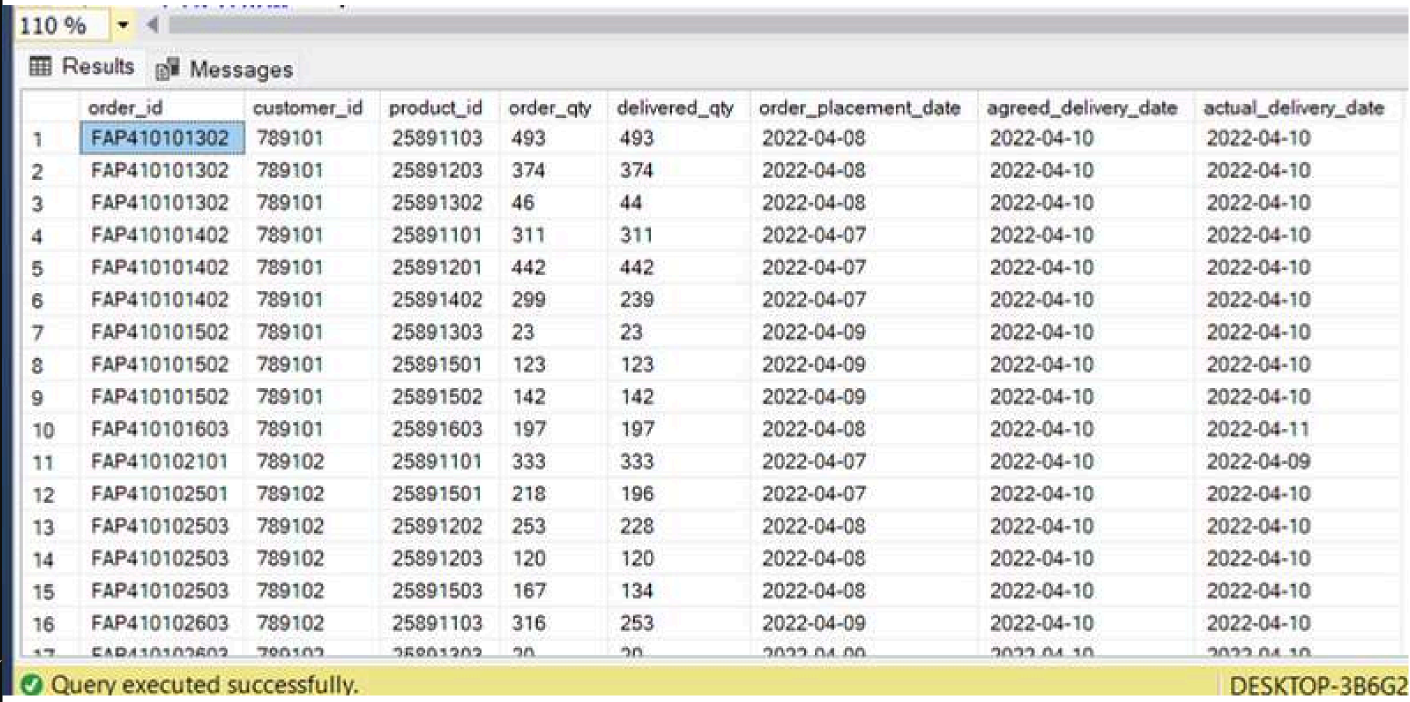


```sql
-- Identify distinct actual_delivery_date values not present in dim_date
SELECT DISTINCT actual_delivery_date
FROM fact_order_lines
WHERE TRY_CONVERT(DATE, actual_delivery_date, 103) NOT IN (SELECT date FROM dim_date);

-- Add foreign key constraint for actual_delivery_date
ALTER TABLE fact_order_lines
ADD CONSTRAINT FK_fact_order_lines_actual_delivery_date
FOREIGN KEY (actual_delivery_date) REFERENCES dim_date(date);
```



```sql
-- Add a new column for order_placement_date in fact_orders_aggregate
ALTER TABLE fact_orders_aggregate
ADD order_placement_date_converted DATE;

-- Convert order_placement_date to DATE format and store it in the new column
UPDATE fact_orders_aggregate
SET order_placement_date_converted = TRY_CONVERT(DATE, LTRIM(RTRIM(order_placement_date)), 103);

-- Check for conversion errors in the new column
SELECT COUNT(*) AS InvalidDateCount
FROM fact_orders_aggregate
WHERE order_placement_date_converted IS NULL;

-- Drop the original order_placement_date column
ALTER TABLE fact_orders_aggregate
DROP COLUMN order_placement_date;

-- Rename the converted date column
EXEC sp_rename 'fact_orders_aggregate.order_placement_date_converted', 'order_placement_date', 'COLUMN';

-- Add a foreign key constraint for order_placement_date in fact_orders_aggregate
ALTER TABLE fact_orders_aggregate
ADD CONSTRAINT FK_fact_orders_aggregate_order_placement_date
FOREIGN KEY (order_placement_date) REFERENCES dim_date(date);
```

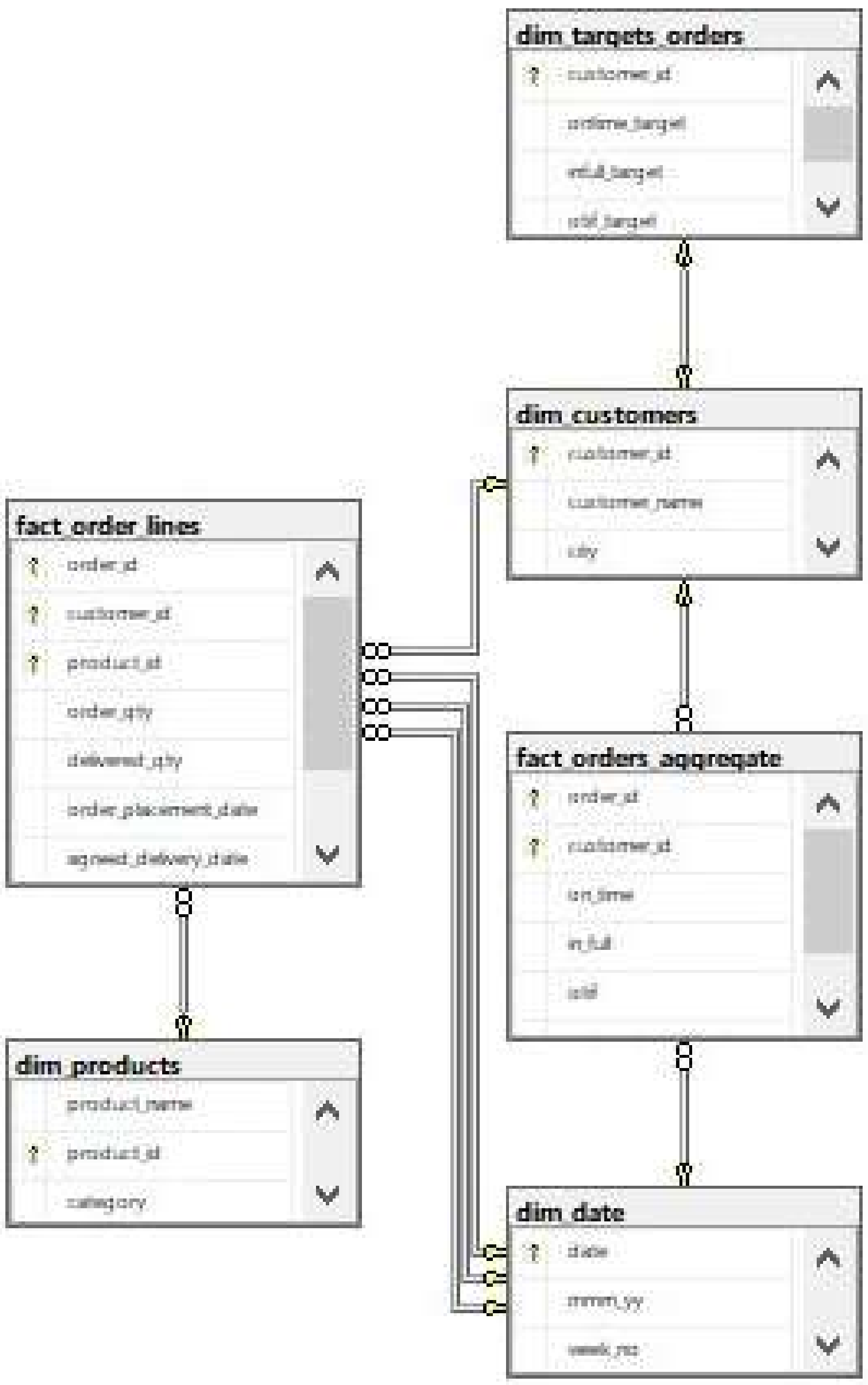
02 - Data Management and SQL Database Setup



	order_id	customer_id	product_id	order_qty	delivered_qty	order_placement_date	agreed_delivery_date	actual_delivery_date
1	FAP410101302	789101	25891103	493	493	2022-04-08	2022-04-10	2022-04-10
2	FAP410101302	789101	25891203	374	374	2022-04-08	2022-04-10	2022-04-10
3	FAP410101302	789101	25891302	46	44	2022-04-08	2022-04-10	2022-04-10
4	FAP410101402	789101	25891101	311	311	2022-04-07	2022-04-10	2022-04-10
5	FAP410101402	789101	25891201	442	442	2022-04-07	2022-04-10	2022-04-10
6	FAP410101402	789101	25891402	299	239	2022-04-07	2022-04-10	2022-04-10
7	FAP410101502	789101	25891303	23	23	2022-04-09	2022-04-10	2022-04-10
8	FAP410101502	789101	25891501	123	123	2022-04-09	2022-04-10	2022-04-10
9	FAP410101502	789101	25891502	142	142	2022-04-09	2022-04-10	2022-04-10
10	FAP410101603	789101	25891603	197	197	2022-04-08	2022-04-10	2022-04-11
11	FAP410102101	789102	25891101	333	333	2022-04-07	2022-04-10	2022-04-09
12	FAP410102501	789102	25891501	218	196	2022-04-07	2022-04-10	2022-04-10
13	FAP410102503	789102	25891202	253	228	2022-04-08	2022-04-10	2022-04-10
14	FAP410102503	789102	25891203	120	120	2022-04-08	2022-04-10	2022-04-10
15	FAP410102503	789102	25891503	167	134	2022-04-08	2022-04-10	2022-04-10
16	FAP410102603	789102	25891103	316	253	2022-04-09	2022-04-10	2022-04-10
17	FAP410102603	789102	25891303	20	20	2022-04-09	2022-04-10	2022-04-10

Query executed successfully. DESKTOP-3B6G2

02 - Data Management and SQL Database Setup

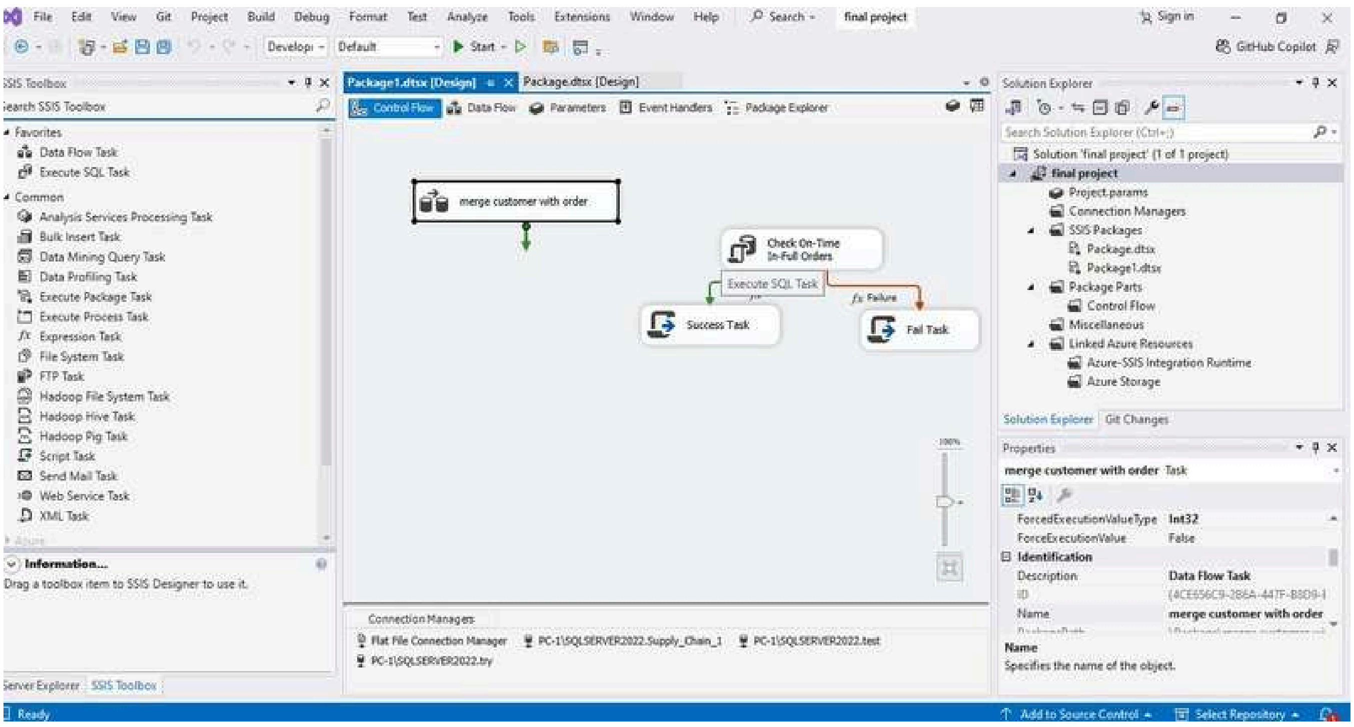


03

Data Warehousing and Python Programming

Data Warehouse Implementation

Tools used: SSIS , control flow, data flow, execute SQL



Properties

merge customer with order Task

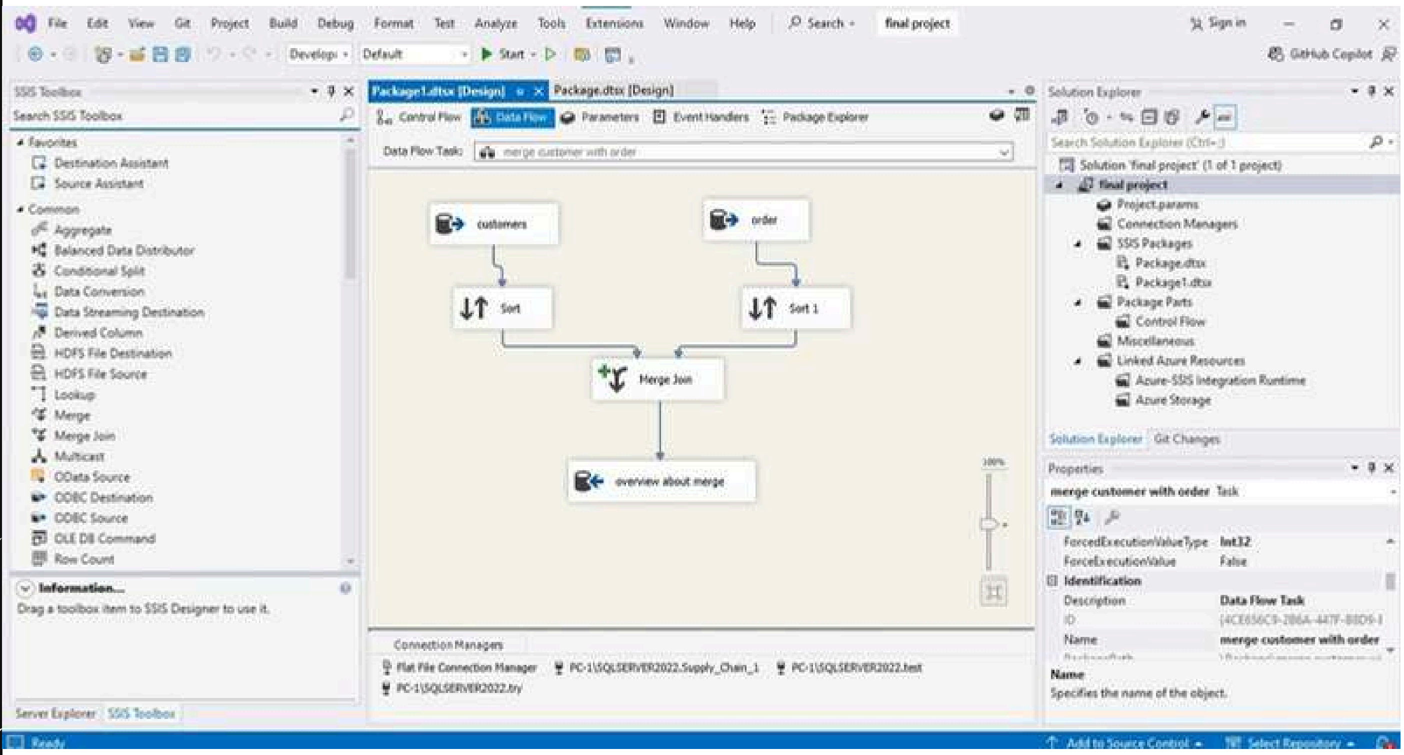
ForcedExecutionValueType	Int32
ForceExecutionValue	False
Identification	
Description	Data Flow Task
ID	{4CE656C9-2B6A-447F-B8D9-I
Name	merge customer with order

Name
Specifies the name of the object.

03

Data Warehousing and Python Programming

Data Warehouse Implementation

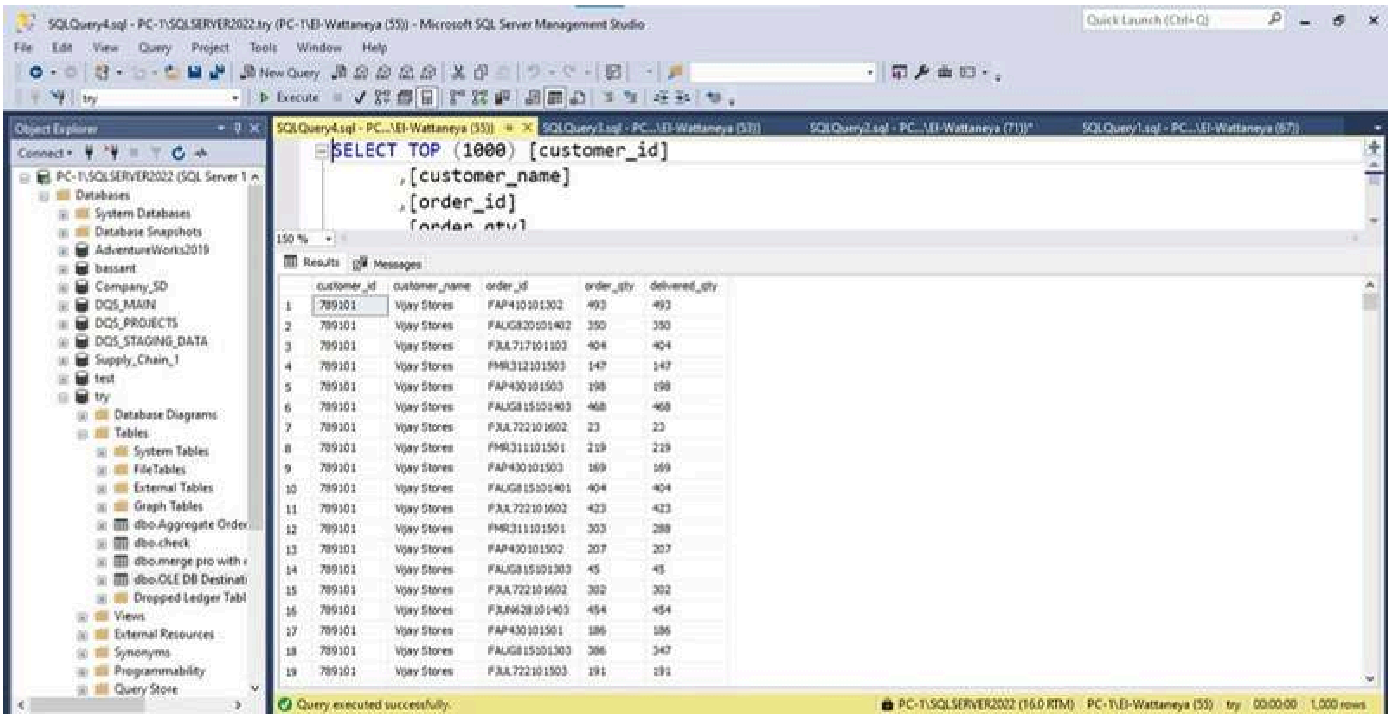


ForcedExecutionValueType	Int32
ForceExecutionValue	False
Identification	
Description	Data Flow Task
ID	{4CE656C9-2B6A-447F-B8D9-I
Name	merge customer with order

03

Data Warehousing and Python Programming

Data Warehouse Implementation



```sql
SELECT TOP (1000) [customer_id]
      ,[customer_name]
      ,[order_id]
      [order_qty]
```

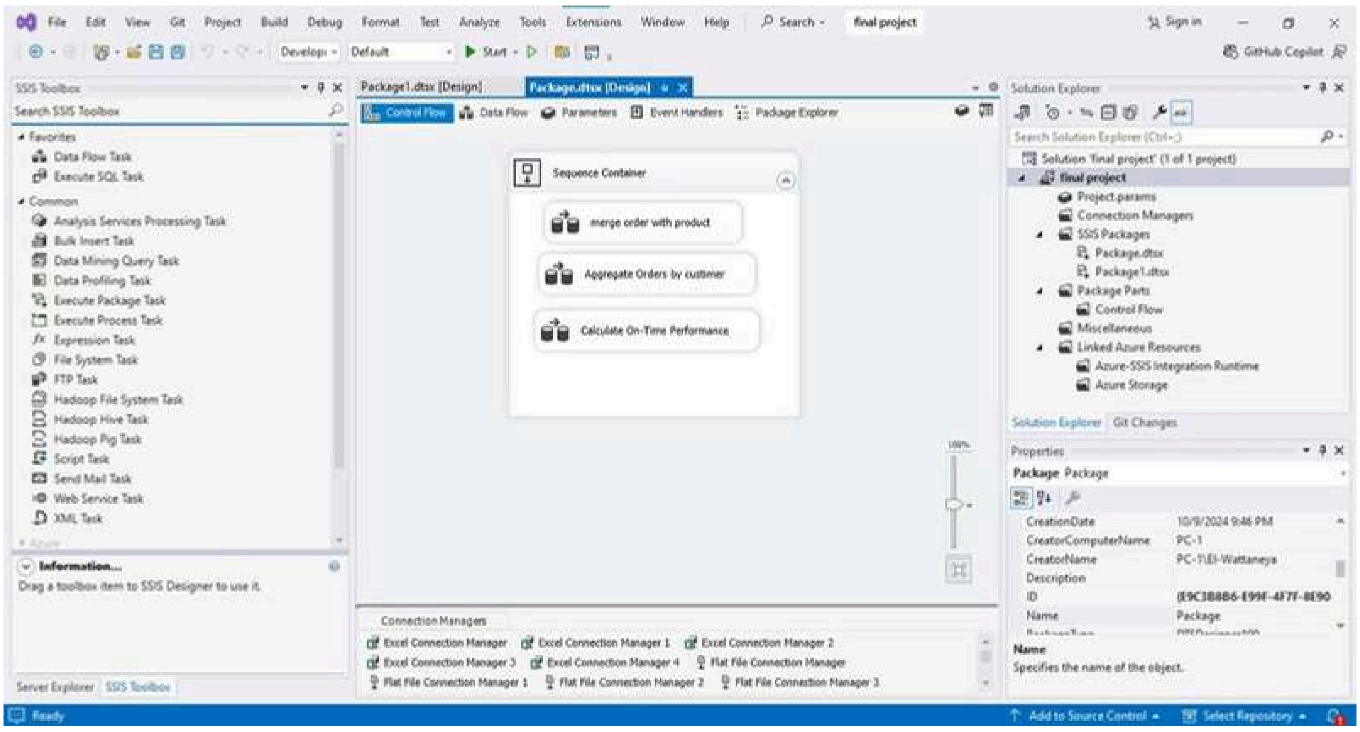
	customer_id	customer_name	order_id	order_qty	delivered_qty
1	789101	Vijay Stores	FAP410101302	493	493
2	789101	Vijay Stores	FAUG820101402	350	350
3	789101	Vijay Stores	FJUL717101103	404	404
4	789101	Vijay Stores	FMR312101503	147	147
5	789101	Vijay Stores	FAP430101503	198	198
6	789101	Vijay Stores	FAUG815101403	468	468
7	789101	Vijay Stores	FJUL722101602	23	23
8	789101	Vijay Stores	FMR311101501	219	219
9	789101	Vijay Stores	FAP430101503	169	169
10	789101	Vijay Stores	FAUG815101401	404	404
11	789101	Vijay Stores	FJUL722101602	423	423
12	789101	Vijay Stores	FMR311101501	303	288
13	789101	Vijay Stores	FAP430101502	207	207
14	789101	Vijay Stores	FAUG815101303	45	45
15	789101	Vijay Stores	FJUL722101602	302	302
16	789101	Vijay Stores	FJUN628101403	454	454
17	789101	Vijay Stores	FAP430101501	186	186
18	789101	Vijay Stores	FAUG815101303	386	347
19	789101	Vijay Stores	FJUL722101503	191	191

Query executed successfully.

03

Data Warehousing and Python Programming

Data Warehouse Implementation

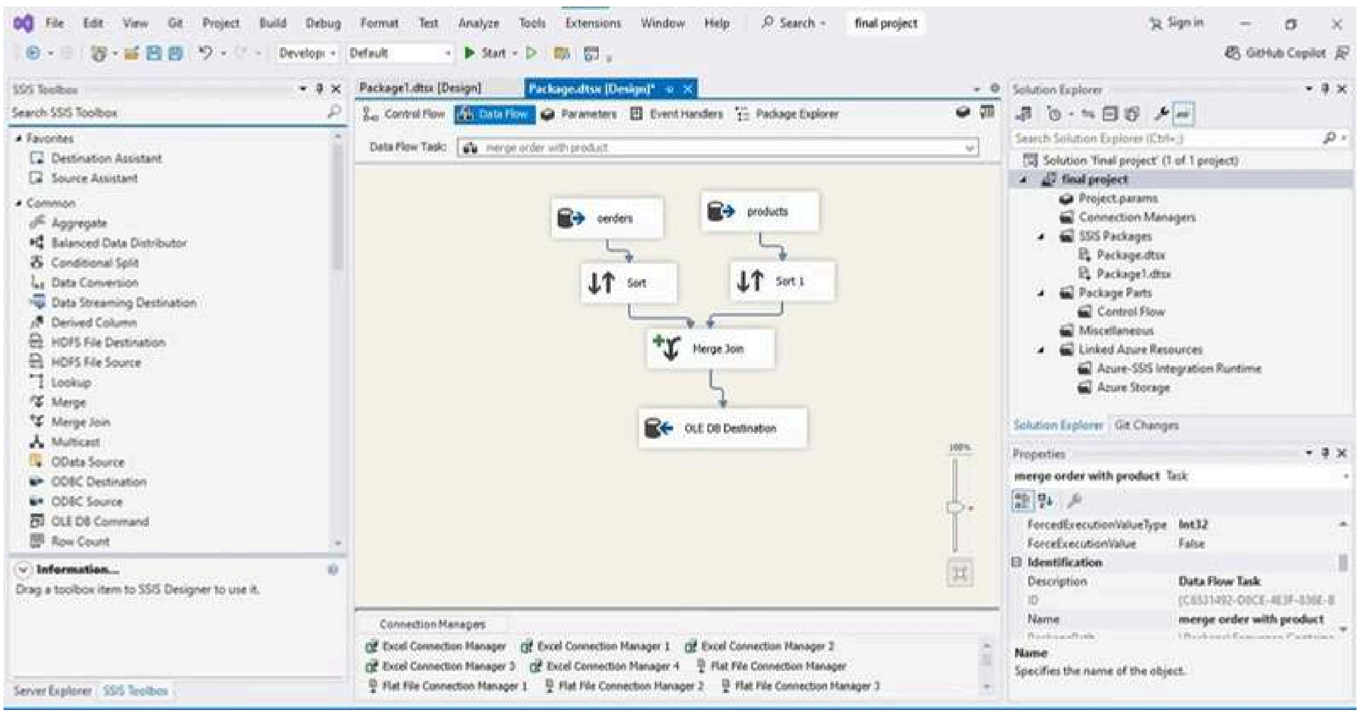


CreationDate	10/9/2024 9:46 PM
CreatorComputerName	PC-1
CreatorName	PC-1\El-Wattaneya
Description	
ID	{E9C3B8B6-E99F-4F7F-8E90
Name	Package

Name
Specifies the name of the object.

Data Warehousing and Python Programming

Data Warehouse Implementation



merge order with product Task

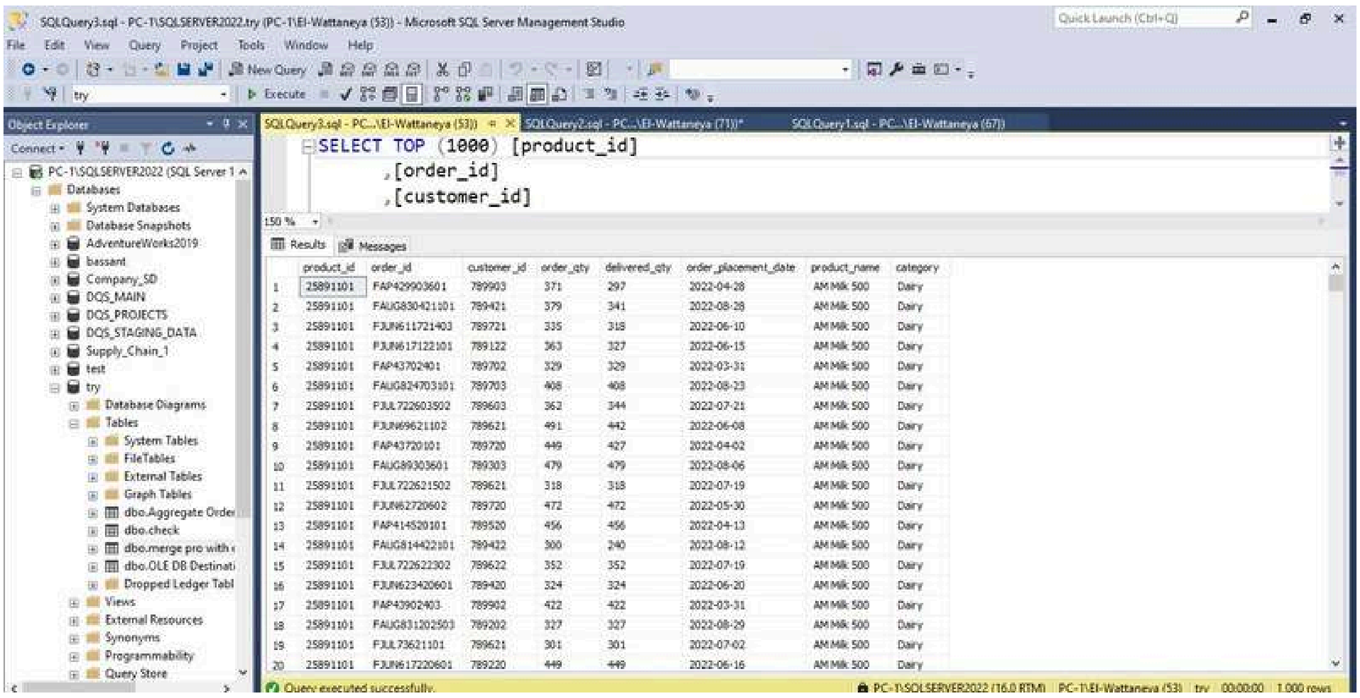
ForcedExecutionValueType	Int32
ForceExecutionValue	False
Identification	
Description	Data Flow Task
ID	{C6531492-D9CE-4E3F-836E-8
Name	merge order with product

Name
Specifies the name of the object.

03

Data Warehousing and Python Programming

Data Warehouse Implementation



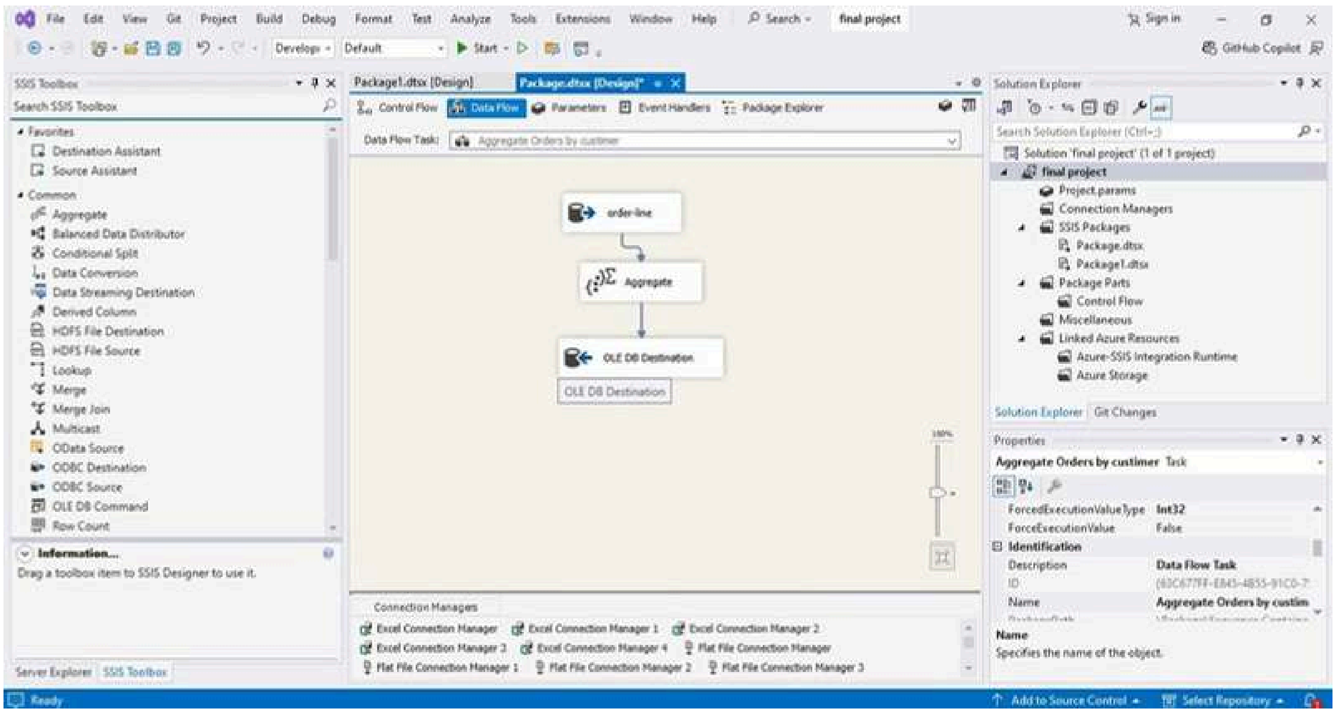
```
SELECT TOP (1000) [product_id]
      ,[order_id]
      ,[customer_id]
```

	product_id	order_id	customer_id	order_qty	delivered_qty	order_placement_date	product_name	category
1	25891101	FAP429903601	789903	371	297	2022-04-28	AM Milk 500	Dairy
2	25891101	FAUG830421101	789421	379	341	2022-08-28	AM Milk 500	Dairy
3	25891101	FJUN611721403	789721	335	318	2022-06-10	AM Milk 500	Dairy
4	25891101	FJUN617122101	789122	363	327	2022-06-15	AM Milk 500	Dairy
5	25891101	FAP43702401	789702	329	329	2022-03-31	AM Milk 500	Dairy
6	25891101	FAUG824703101	789703	408	408	2022-08-23	AM Milk 500	Dairy
7	25891101	FJUL722603502	789603	362	344	2022-07-21	AM Milk 500	Dairy
8	25891101	FJUN69621102	789621	491	442	2022-06-08	AM Milk 500	Dairy
9	25891101	FAP43720101	789720	449	427	2022-04-02	AM Milk 500	Dairy
10	25891101	FAUG89303601	789303	479	479	2022-08-06	AM Milk 500	Dairy
11	25891101	FJUL722621502	789621	318	318	2022-07-19	AM Milk 500	Dairy
12	25891101	FJUN62720602	789720	472	472	2022-05-30	AM Milk 500	Dairy
13	25891101	FAP414520101	789520	456	456	2022-04-13	AM Milk 500	Dairy
14	25891101	FAUG814422101	789422	300	240	2022-08-12	AM Milk 500	Dairy
15	25891101	FJUL722622302	789622	352	352	2022-07-19	AM Milk 500	Dairy
16	25891101	FJUN623420601	789420	324	324	2022-06-20	AM Milk 500	Dairy
17	25891101	FAP43902403	789902	422	422	2022-03-31	AM Milk 500	Dairy
18	25891101	FAUG831202503	789202	327	327	2022-08-29	AM Milk 500	Dairy
19	25891101	FJUL73621101	789621	301	301	2022-07-02	AM Milk 500	Dairy
20	25891101	FJUN617220601	789220	449	449	2022-06-16	AM Milk 500	Dairy

```
Query executed successfully.    PC-1\SQLSERVER2022 (16.0 RTM)   PC-1\El-Wattaneya (53)   try   00:00:00   1,000 rows
```

Data Warehousing and Python Programming

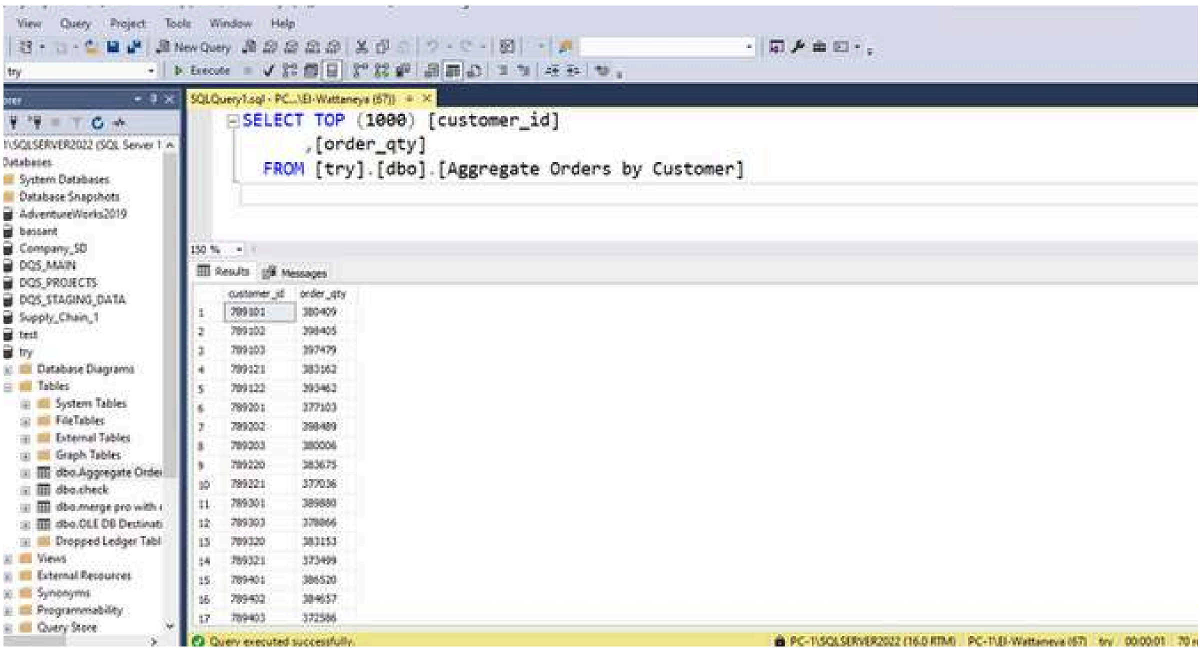
Data Warehouse Implementation



03

Data Warehousing and Python Programming

Data Warehouse Implementation



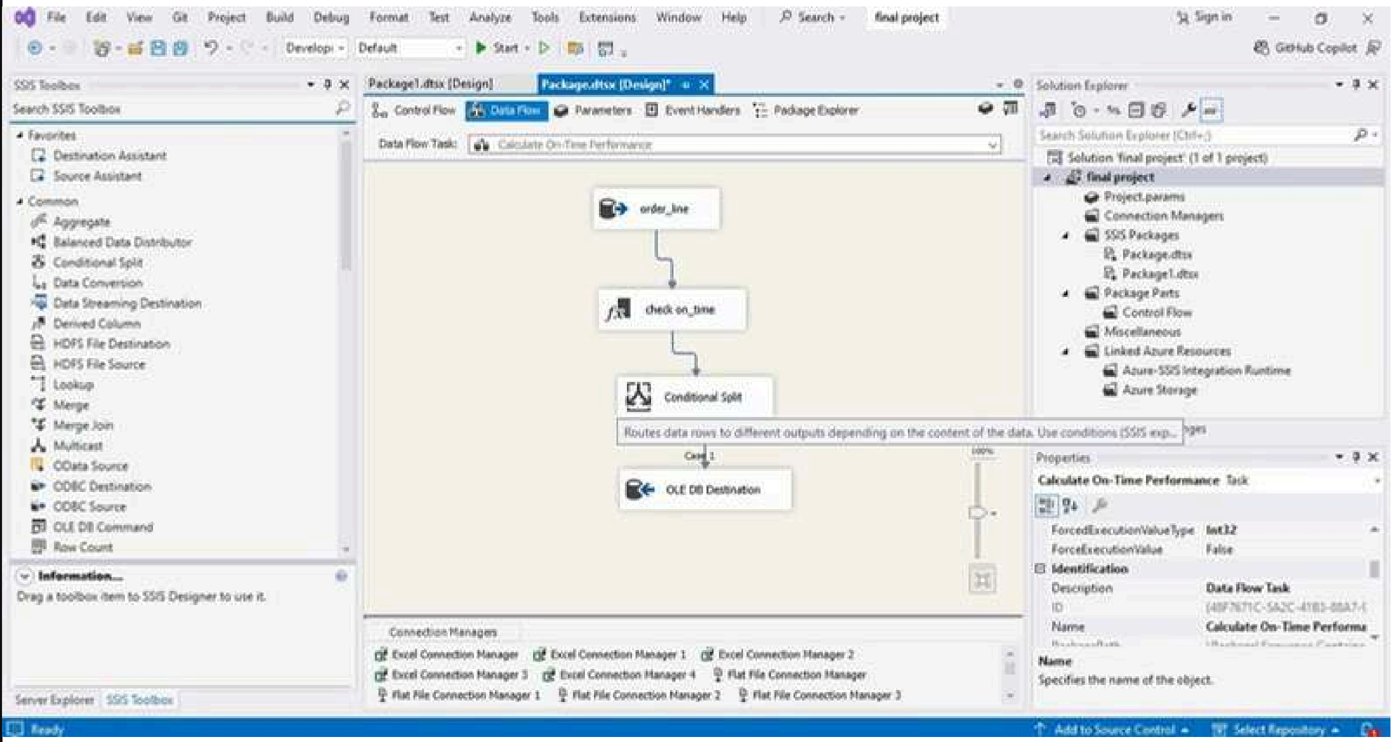
```sql
SELECT TOP (1000) [customer_id]
      ,[order_qty]
  FROM [try].[dbo].[Aggregate Orders by Customer]
```

	customer_id	order_qty
1	789101	380409
2	789102	398405
3	789103	397479
4	789121	383162
5	789122	393462
6	789201	377103
7	789202	398489
8	789203	380006
9	789220	383675
10	789221	377036
11	789301	389880
12	789303	378866
13	789320	383153
14	789321	373499
15	789401	386520
16	789402	384657
17	789403	372586

```
Query executed successfully.    PC-1\SQLSERVER2022 (16.0 RTM)   PC-1\El-Wattaneya (67)   try   00:00:01   70 r
```

Data Warehousing and Python Programming

Data Warehouse Implementation

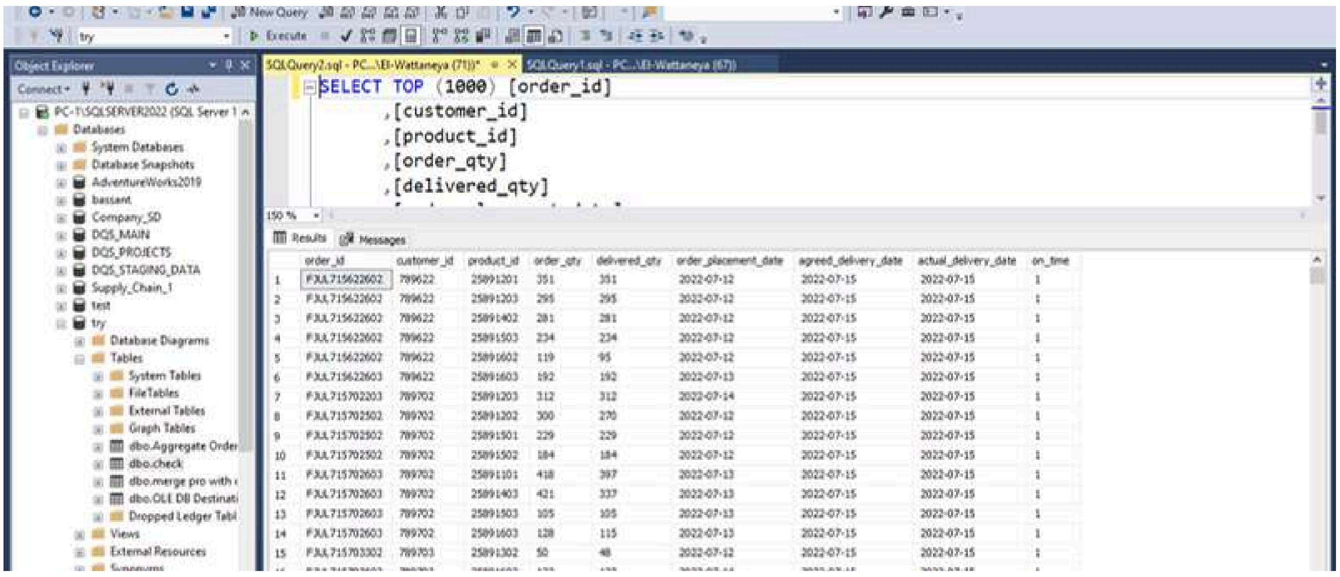


Routes data rows to different outputs depending on the content of the data. Use conditions (SSIS exp...

Information...
Drag a toolbox item to SSIS Designer to use it.

ForcedExecutionValueType	Int32
ForceExecutionValue	False
Description	Data Flow Task
ID	{48F7671C-5A2C-41B3-88A7-(
Name	Calculate On-Time Performa

Name
Specifies the name of the object.

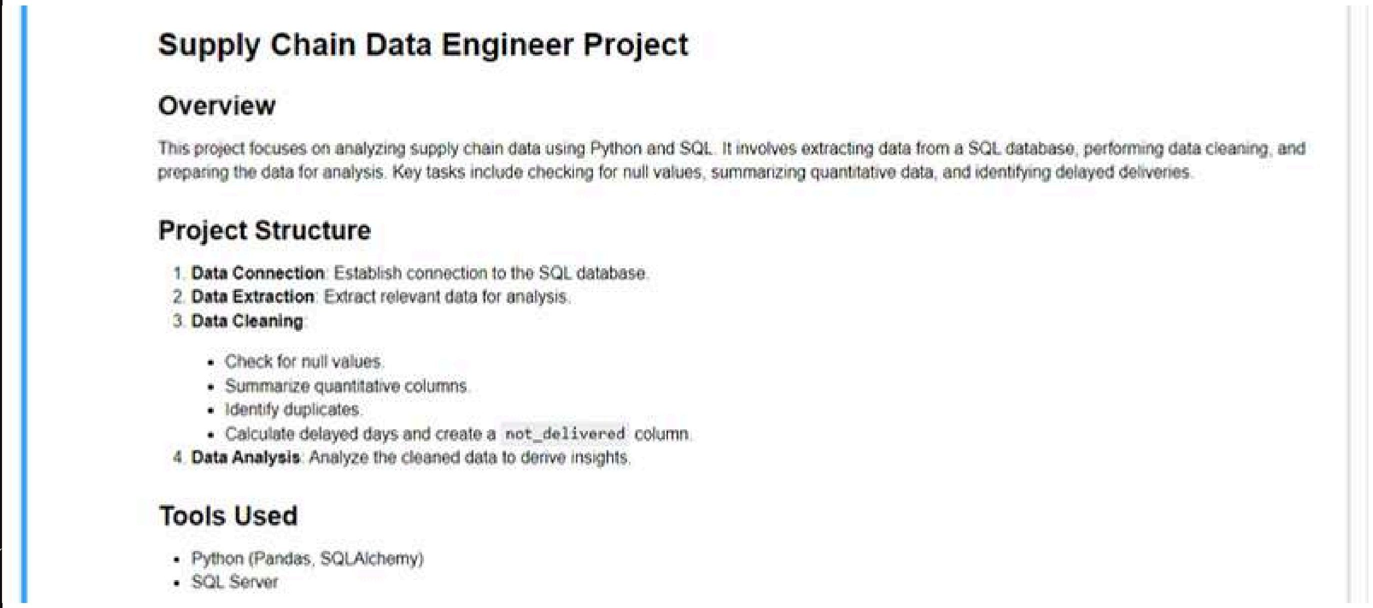


```
SELECT TOP (1000) [order_id]
      ,[customer_id]
      ,[product_id]
      ,[order_qty]
      ,[delivered_qty]
```

	order_id	customer_id	product_id	order_qty	delivered_qty	order_placement_date	agreed_delivery_date	actual_delivery_date	on_time
1	FJUL715622602	789622	25891201	351	351	2022-07-12	2022-07-15	2022-07-15	1
2	FJUL715622602	789622	25891203	295	295	2022-07-12	2022-07-15	2022-07-15	1
3	FJUL715622602	789622	25891402	281	281	2022-07-12	2022-07-15	2022-07-15	1
4	FJUL715622602	789622	25891503	234	234	2022-07-12	2022-07-15	2022-07-15	1
5	FJUL715622602	789622	25891602	119	95	2022-07-12	2022-07-15	2022-07-15	1
6	FJUL715622603	789622	25891603	192	192	2022-07-13	2022-07-15	2022-07-15	1
7	FJUL715702203	789702	25891203	312	312	2022-07-14	2022-07-15	2022-07-15	1
8	FJUL715702502	789702	25891202	300	270	2022-07-12	2022-07-15	2022-07-15	1
9	FJUL715702502	789702	25891501	229	229	2022-07-12	2022-07-15	2022-07-15	1
10	FJUL715702502	789702	25891502	184	184	2022-07-12	2022-07-15	2022-07-15	1
11	FJUL715702603	789702	25891101	418	397	2022-07-13	2022-07-15	2022-07-15	1
12	FJUL715702603	789702	25891403	421	337	2022-07-13	2022-07-15	2022-07-15	1
13	FJUL715702603	789702	25891503	105	105	2022-07-13	2022-07-15	2022-07-15	1
14	FJUL715702603	789702	25891603	128	115	2022-07-13	2022-07-15	2022-07-15	1
15	FJUL715703302	789703	25891302	50	48	2022-07-12	2022-07-15	2022-07-15	1

Data Warehousing and Python Programming

DATA CLEANING



Supply Chain Data Engineer Project

Overview

This project focuses on analyzing supply chain data using Python and SQL. It involves extracting data from a SQL database, performing data cleaning, and preparing the data for analysis. Key tasks include checking for null values, summarizing quantitative data, and identifying delayed deliveries.

Project Structure

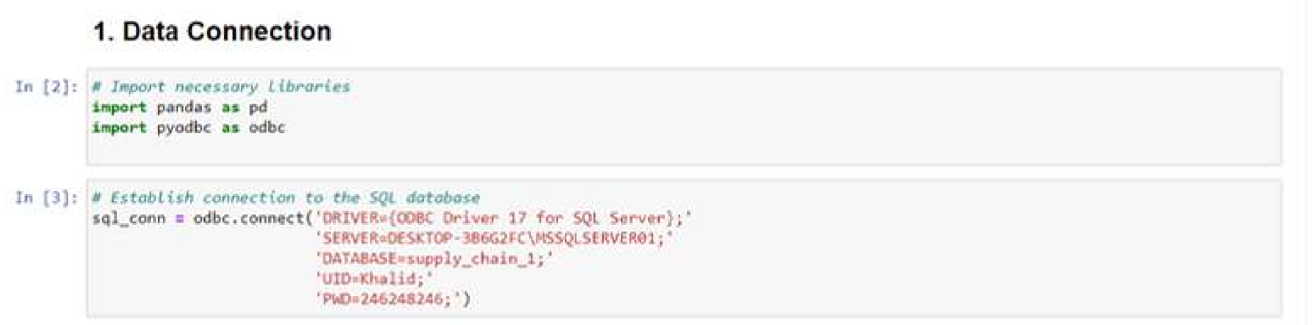
1. **Data Connection**: Establish connection to the SQL database.
2. **Data Extraction**: Extract relevant data for analysis.
3. **Data Cleaning**:
   - Check for null values.
   - Summarize quantitative columns.
   - Identify duplicates.
   - Calculate delayed days and create a `not_delivered` column.
4. **Data Analysis**: Analyze the cleaned data to derive insights.

Tools Used

- Python (Pandas, SQLAlchemy)
- SQL Server

Data Warehousing and Python Programming

DATA CLEANING



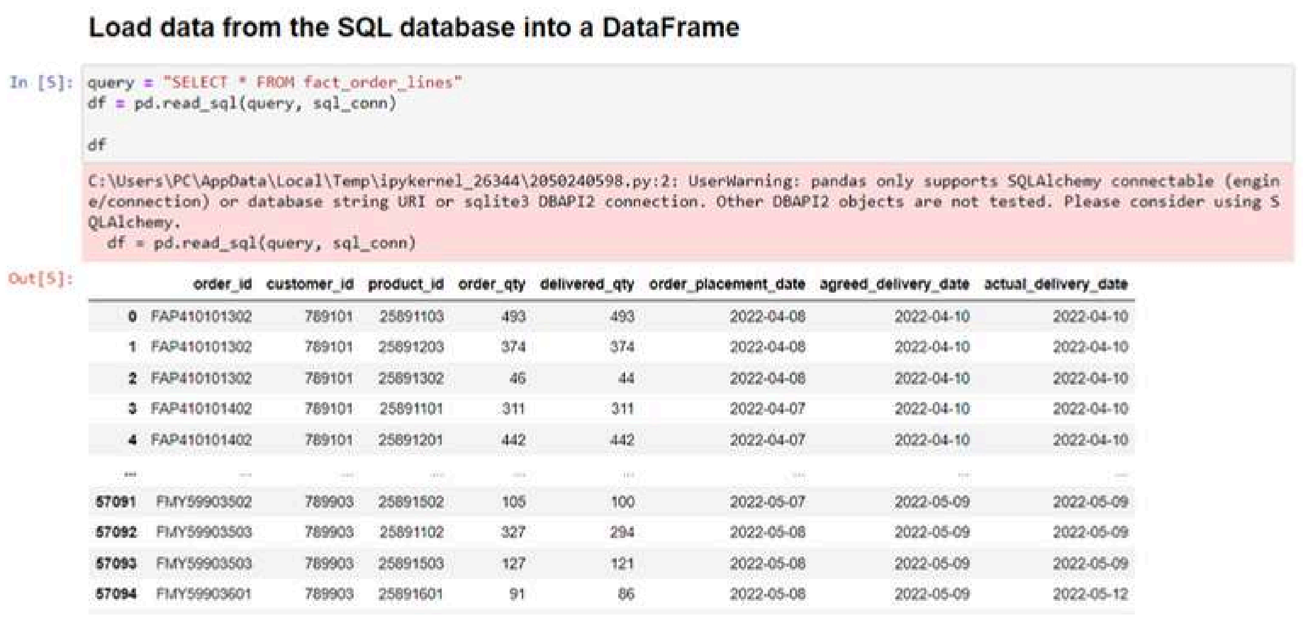
1. Data Connection

```
In [2]: # Import necessary libraries
        import pandas as pd
        import pyodbc as odbc
```

```
In [3]: # Establish connection to the SQL database
        sql_conn = odbc.connect('DRIVER={ODBC Driver 17 for SQL Server};'
                                'SERVER=DESKTOP-3B6G2FC\MSSQLSERVER01;'
                                'DATABASE=supply_chain_1;'
                                'UID=Khalid;'
                                'PWD=246248246;')
```

Data Warehousing and Python Programming

DATA CLEANING



Load data from the SQL database into a DataFrame

```
In [5]: query = "SELECT * FROM fact_order_lines"
        df = pd.read_sql(query, sql_conn)

        df
```

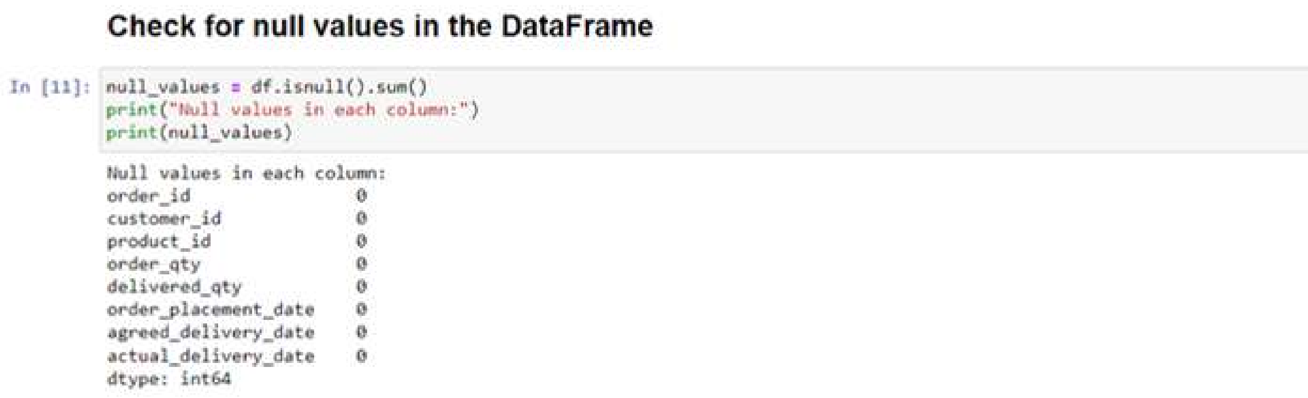
```
C:\Users\PC\AppData\Local\Temp\ipykernel_26344\2050240598.py:2: UserWarning: pandas only supports SQLAlchemy connectable (engin
e/connection) or database string URI or sqlite3 DBAPI2 connection. Other DBAPI2 objects are not tested. Please consider using S
QLAlchemy.
  df = pd.read_sql(query, sql_conn)
```

Out[5]:

	order_id	customer_id	product_id	order_qty	delivered_qty	order_placement_date	agreed_delivery_date	actual_delivery_date
0	FAP410101302	789101	25891103	493	493	2022-04-08	2022-04-10	2022-04-10
1	FAP410101302	789101	25891203	374	374	2022-04-08	2022-04-10	2022-04-10
2	FAP410101302	789101	25891302	46	44	2022-04-08	2022-04-10	2022-04-10
3	FAP410101402	789101	25891101	311	311	2022-04-07	2022-04-10	2022-04-10
4	FAP410101402	789101	25891201	442	442	2022-04-07	2022-04-10	2022-04-10
...	...	...	...	...	...	...	...	...
57091	FMY59903502	789903	25891502	105	100	2022-05-07	2022-05-09	2022-05-09
57092	FMY59903503	789903	25891102	327	294	2022-05-08	2022-05-09	2022-05-09
57093	FMY59903503	789903	25891503	127	121	2022-05-08	2022-05-09	2022-05-09
57094	FMY59903601	789903	25891601	91	86	2022-05-08	2022-05-09	2022-05-12

Data Warehousing and Python Programming

DATA CLEANING

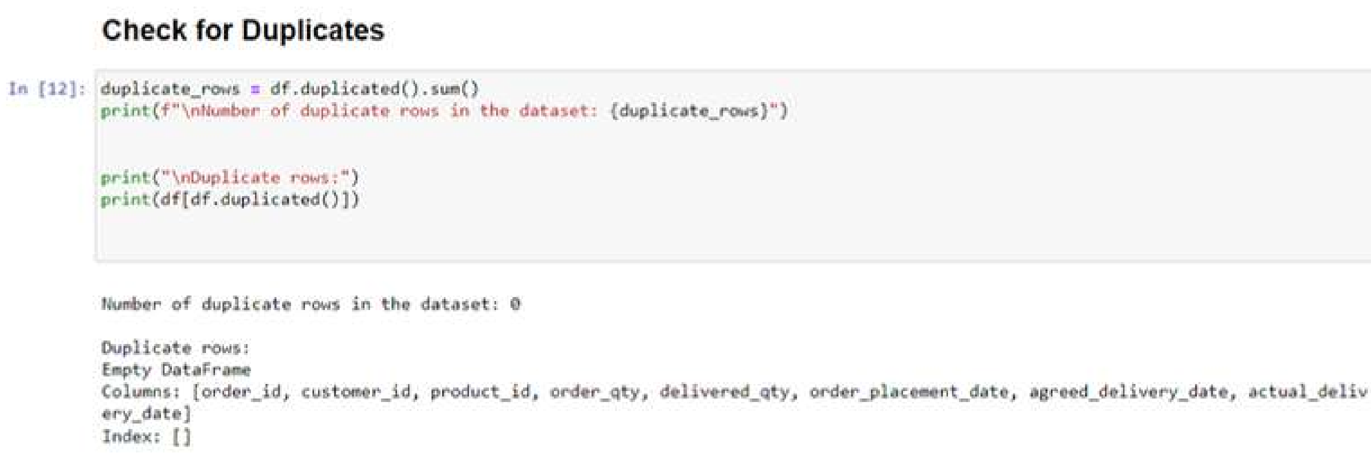


Check for null values in the DataFrame

```
In [11]: null_values = df.isnull().sum()
         print("Null values in each column:")
         print(null_values)

Null values in each column:
order_id                0
customer_id             0
product_id              0
order_qty               0
delivered_qty           0
order_placement_date    0
agreed_delivery_date    0
actual_delivery_date    0
dtype: int64
```

Data Warehousing and Python Programming

DATA CLEANING



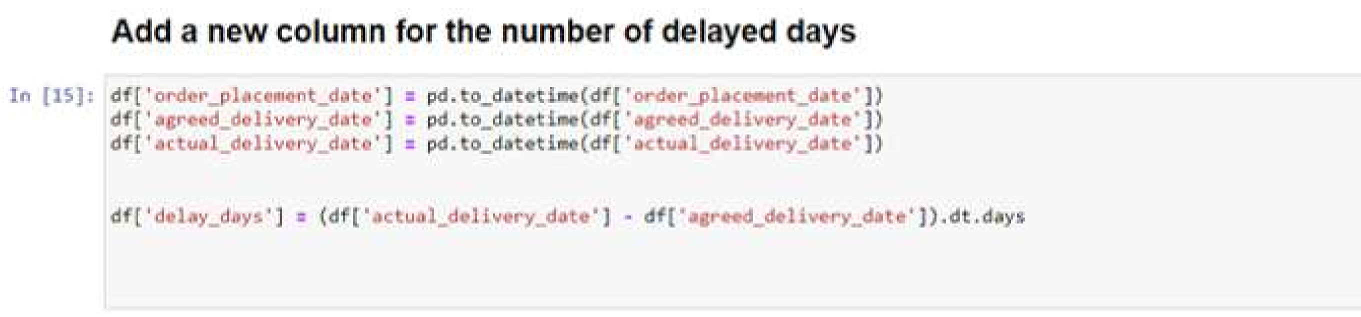
Check for Duplicates

```
In [12]: duplicate_rows = df.duplicated().sum()
         print(f"\nNumber of duplicate rows in the dataset: {duplicate_rows}")


         print("\nDuplicate rows:")
         print(df[df.duplicated()])
```

```
Number of duplicate rows in the dataset: 0

Duplicate rows:
Empty DataFrame
Columns: [order_id, customer_id, product_id, order_qty, delivered_qty, order_placement_date, agreed_delivery_date, actual_deliv
ery_date]
Index: []
```

Data Warehousing and Python Programming

DATA CLEANING

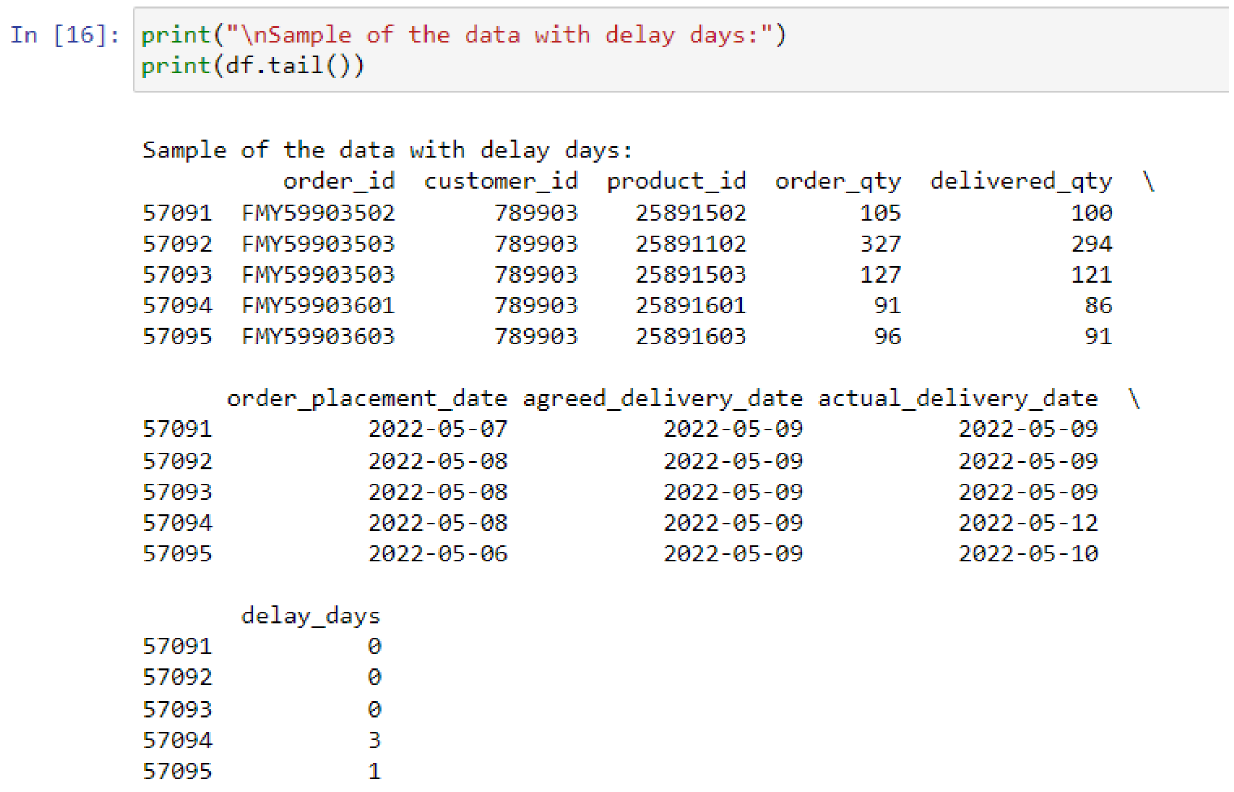


Add a new column for the number of delayed days

```
In [15]: df['order_placement_date'] = pd.to_datetime(df['order_placement_date'])
         df['agreed_delivery_date'] = pd.to_datetime(df['agreed_delivery_date'])
         df['actual_delivery_date'] = pd.to_datetime(df['actual_delivery_date'])


         df['delay_days'] = (df['actual_delivery_date'] - df['agreed_delivery_date']).dt.days
```

Data Warehousing and Python Programming

DATA CLEANING

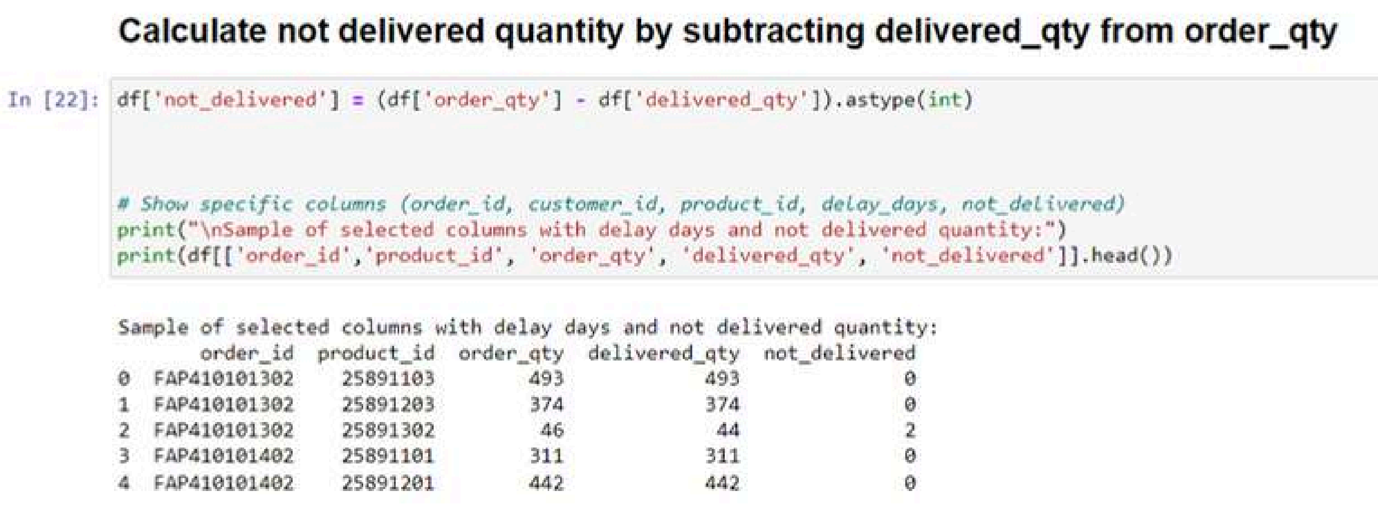


```
In [16]: print("\nSample of the data with delay days:")
         print(df.tail())
```

```
Sample of the data with delay days:
          order_id  customer_id  product_id  order_qty  delivered_qty  \
57091  FMY59903502       789903    25891502        105            100
57092  FMY59903503       789903    25891102        327            294
57093  FMY59903503       789903    25891503        127            121
57094  FMY59903601       789903    25891601         91             86
57095  FMY59903603       789903    25891603         96             91

      order_placement_date agreed_delivery_date actual_delivery_date  \
57091           2022-05-07           2022-05-09           2022-05-09
57092           2022-05-08           2022-05-09           2022-05-09
57093           2022-05-08           2022-05-09           2022-05-09
57094           2022-05-08           2022-05-09           2022-05-12
57095           2022-05-06           2022-05-09           2022-05-10

       delay_days
57091           0
57092           0
57093           0
57094           3
57095           1
```

Data Warehousing and Python Programming

DATA CLEANING



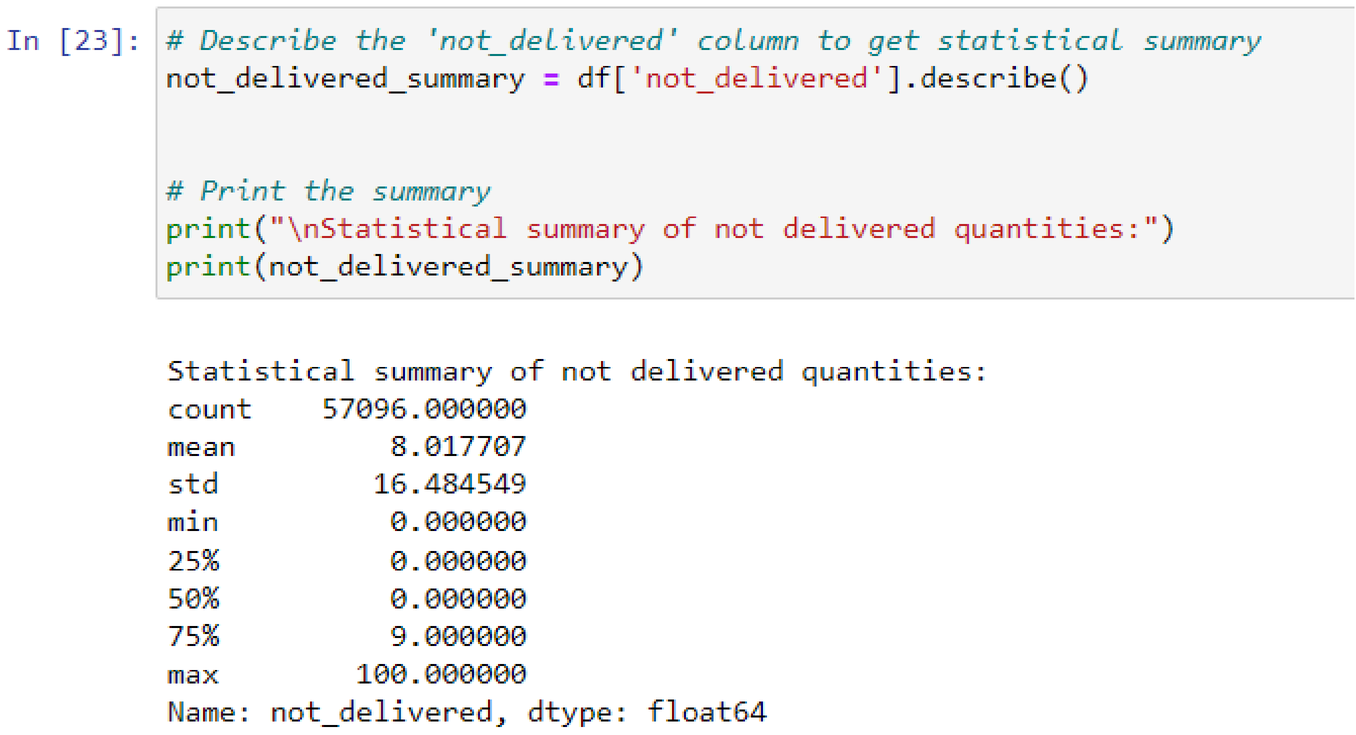
Calculate not delivered quantity by subtracting delivered_qty from order_qty

```
In [22]: df['not_delivered'] = (df['order_qty'] - df['delivered_qty']).astype(int)


         # Show specific columns (order_id, customer_id, product_id, delay_days, not_delivered)
         print("\nSample of selected columns with delay days and not delivered quantity:")
         print(df[['order_id','product_id', 'order_qty', 'delivered_qty', 'not_delivered']].head())
```

```
Sample of selected columns with delay days and not delivered quantity:
        order_id  product_id  order_qty  delivered_qty  not_delivered
0  FAP410101302    25891103        493            493              0
1  FAP410101302    25891203        374            374              0
2  FAP410101302    25891302         46             44              2
3  FAP410101402    25891101        311            311              0
4  FAP410101402    25891201        442            442              0
```

Data Warehousing and Python Programming

DATA CLEANING

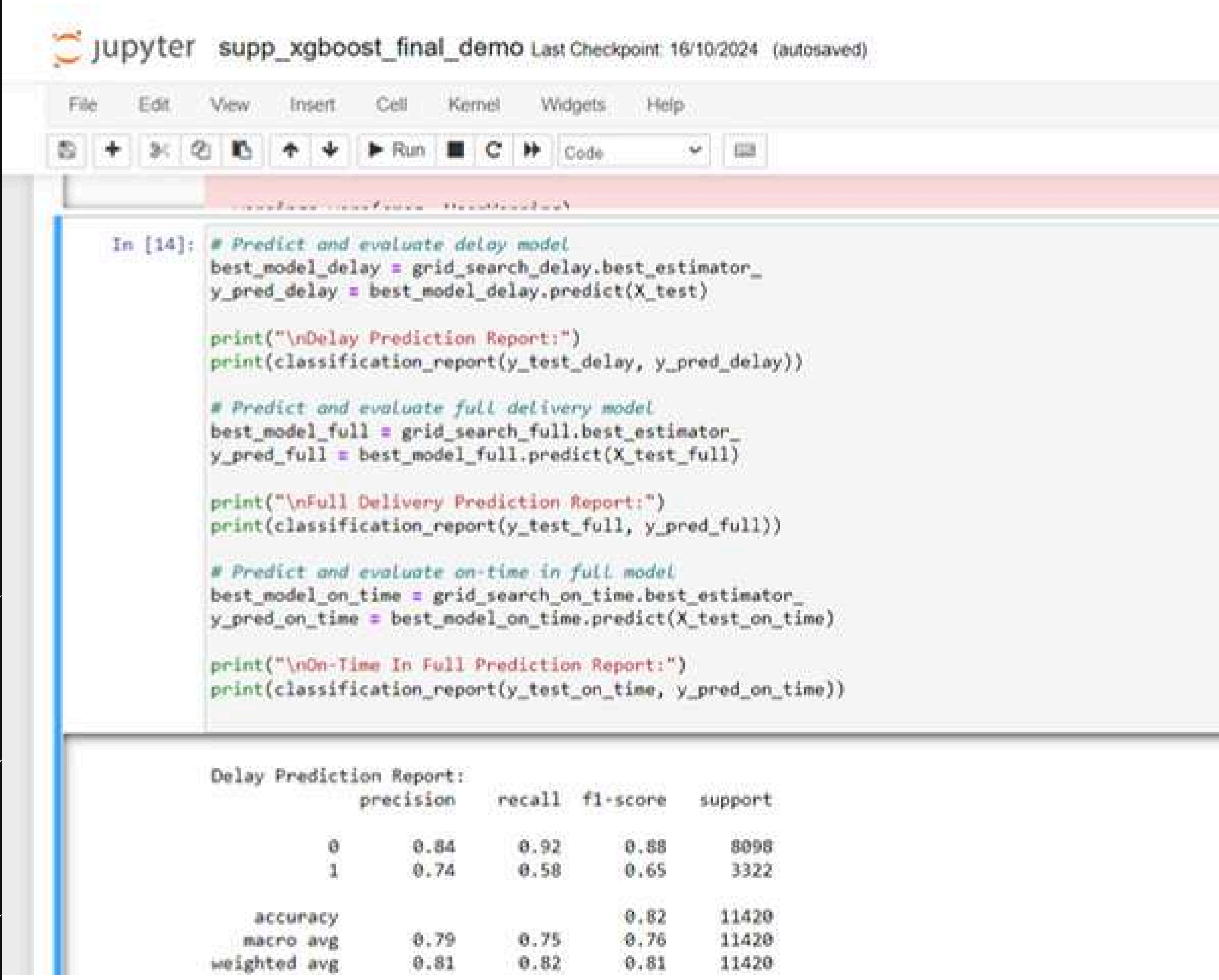


```
In [23]: # Describe the 'not_delivered' column to get statistical summary
         not_delivered_summary = df['not_delivered'].describe()


         # Print the summary
         print("\nStatistical summary of not delivered quantities:")
         print(not_delivered_summary)
```

```
Statistical summary of not delivered quantities:
count    57096.000000
mean         8.017707
std         16.484549
min          0.000000
25%          0.000000
50%          0.000000
75%          9.000000
max        100.000000
Name: not_delivered, dtype: float64
```

Data Warehousing and Python Programming

MODEL


jupyter supp_xgboost_final_demo Last Checkpoint: 16/10/2024 (autosaved)
File Edit View Insert Cell Kernel Widgets Help
Run
Code

```
In [14]: # Predict and evaluate delay model
         best_model_delay = grid_search_delay.best_estimator_
         y_pred_delay = best_model_delay.predict(X_test)

         print("\nDelay Prediction Report:")
         print(classification_report(y_test_delay, y_pred_delay))

         # Predict and evaluate full delivery model
         best_model_full = grid_search_full.best_estimator_
         y_pred_full = best_model_full.predict(X_test_full)

         print("\nFull Delivery Prediction Report:")
         print(classification_report(y_test_full, y_pred_full))

         # Predict and evaluate on-time in full model
         best_model_on_time = grid_search_on_time.best_estimator_
         y_pred_on_time = best_model_on_time.predict(X_test_on_time)

         print("\nOn-Time In Full Prediction Report:")
         print(classification_report(y_test_on_time, y_pred_on_time))
```

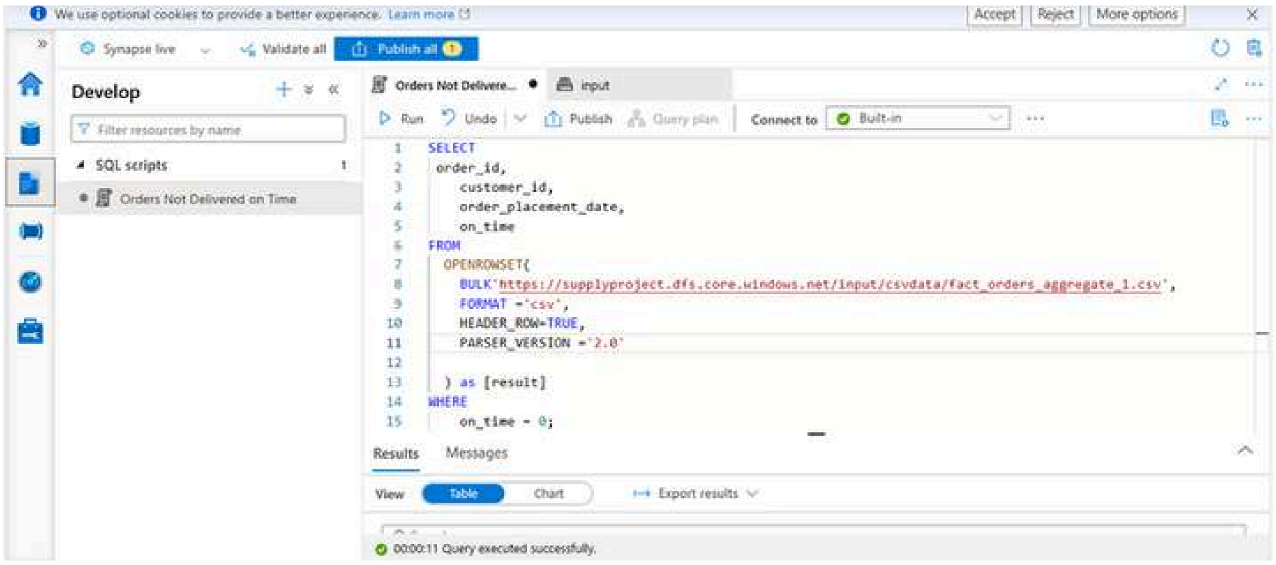
```
Delay Prediction Report:
              precision    recall  f1-score   support

           0       0.84      0.92      0.88      8098
           1       0.74      0.58      0.65      3322

    accuracy                           0.82     11420
   macro avg       0.79      0.75      0.76     11420
weighted avg       0.81      0.82      0.81     11420
```

Azure Integration

We use SQL Scripts in Synapse Analytics for Data Analysis:

retrieve information from data by writing queries to fetch, filter, and analyze data in csv files.

1-Orders not delivered on time

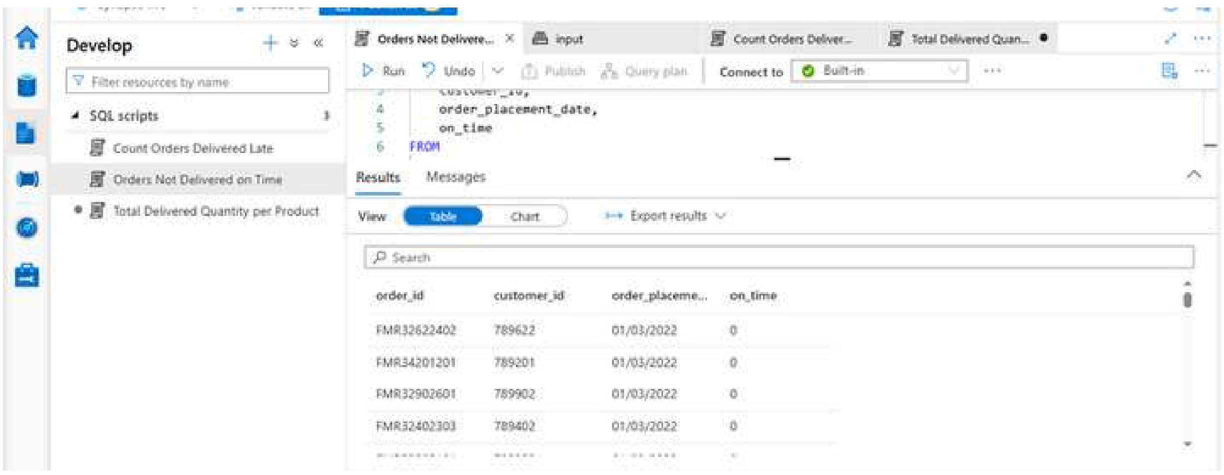


```sql
SELECT
 order_id,
   customer_id,
   order_placement_date,
   on_time
FROM
  OPENROWSET(
   BULK'https://supplyproject.dfs.core.windows.net/input/csvdata/fact_orders_aggregate_1.csv',
   FORMAT ='csv',
   HEADER_ROW=TRUE,
   PARSER_VERSION ='2.0'

  ) as [result]
WHERE
   on_time = 0;
```

00:00:11 Query executed successfully.

Azure Integration

Result in table and chart



```
    customer_id,
    order_placement_date,
    on_time
FROM
```

order_id	customer_id	order_placeme...	on_time
FMR32622402	789622	01/03/2022	0
FMR34201201	789201	01/03/2022	0
FMR32902601	789902	01/03/2022	0
FMR32402303	789402	01/03/2022	0

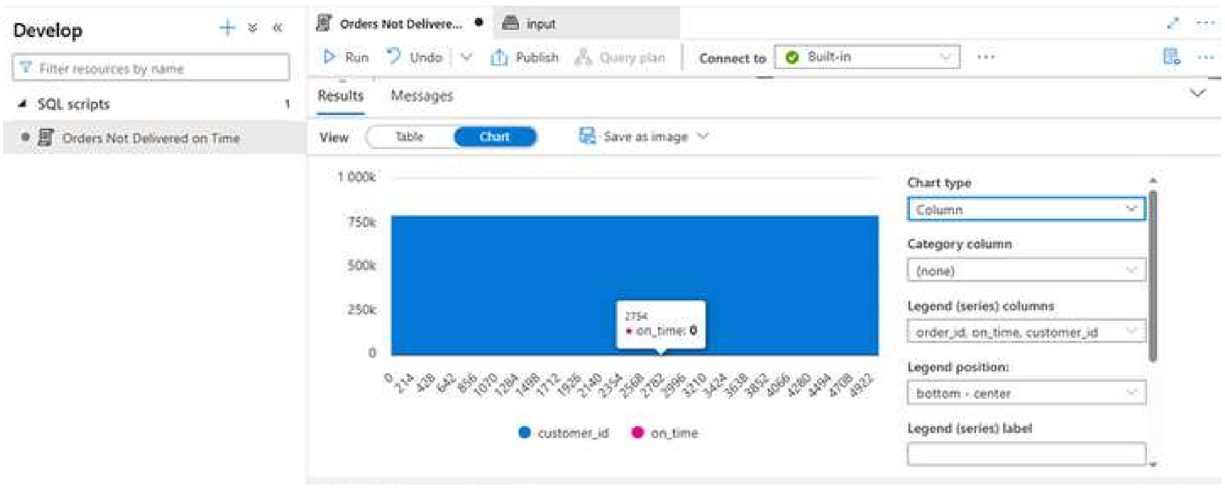
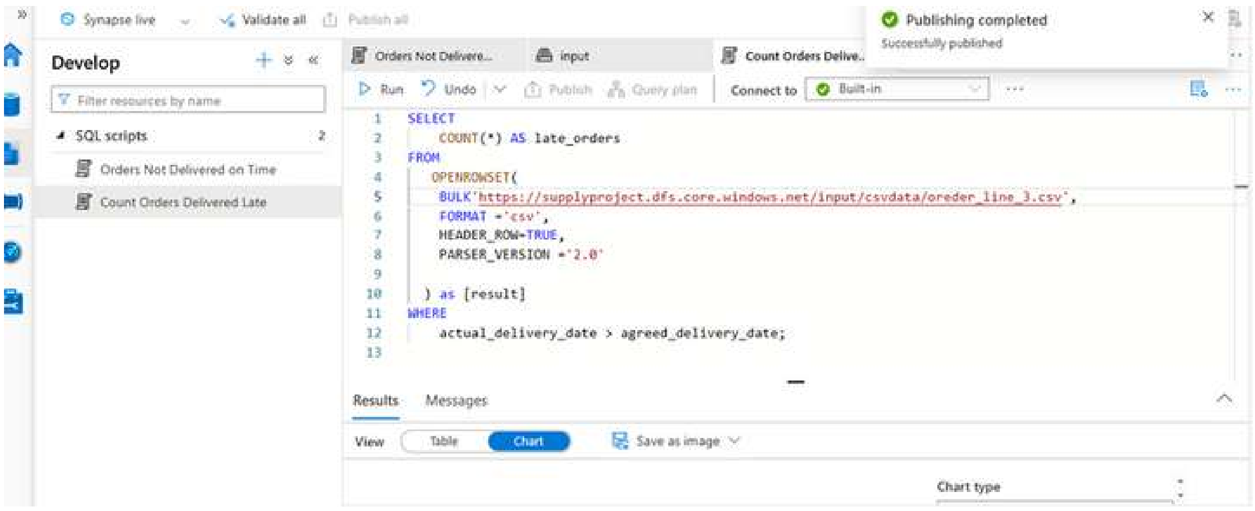


Chart type: Column
Category column: (none)
Legend (series) columns: order_id, on_time, customer_id
Legend position: bottom - center
Legend (series) label:

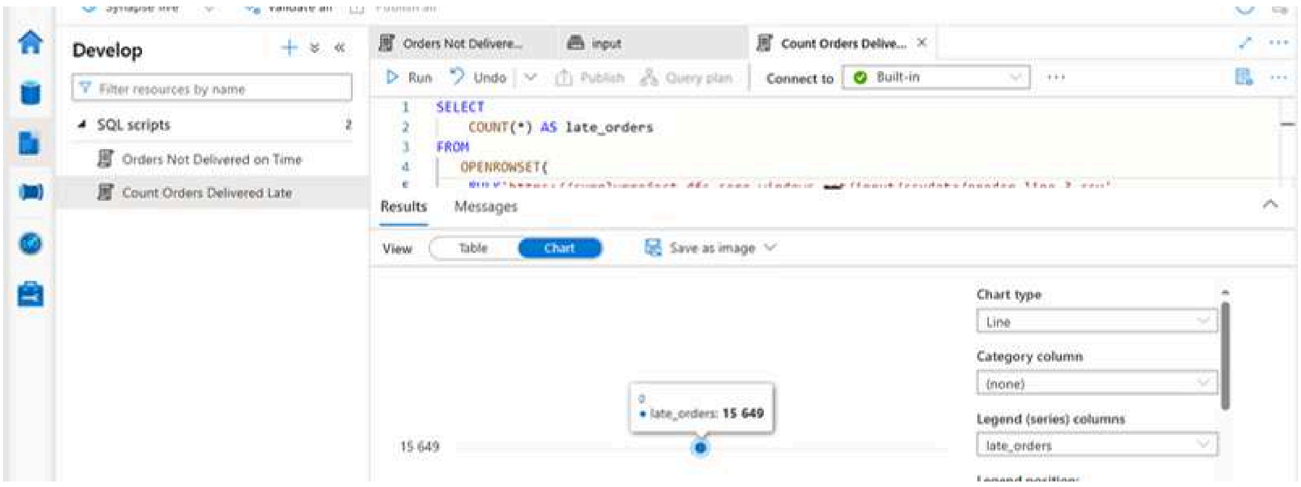
Azure Integration

2-Count orders delivered late



Publishing completed
Successfully published

```
SELECT
    COUNT(*) AS late_orders
FROM
   OPENROWSET(
    BULK'https://supplyproject.dfs.core.windows.net/input/csvdata/oreder_line_3.csv',
    FORMAT ='csv',
    HEADER_ROW=TRUE,
    PARSER_VERSION ='2.0'

  ) as [result]
WHERE
    actual_delivery_date > agreed_delivery_date;
```



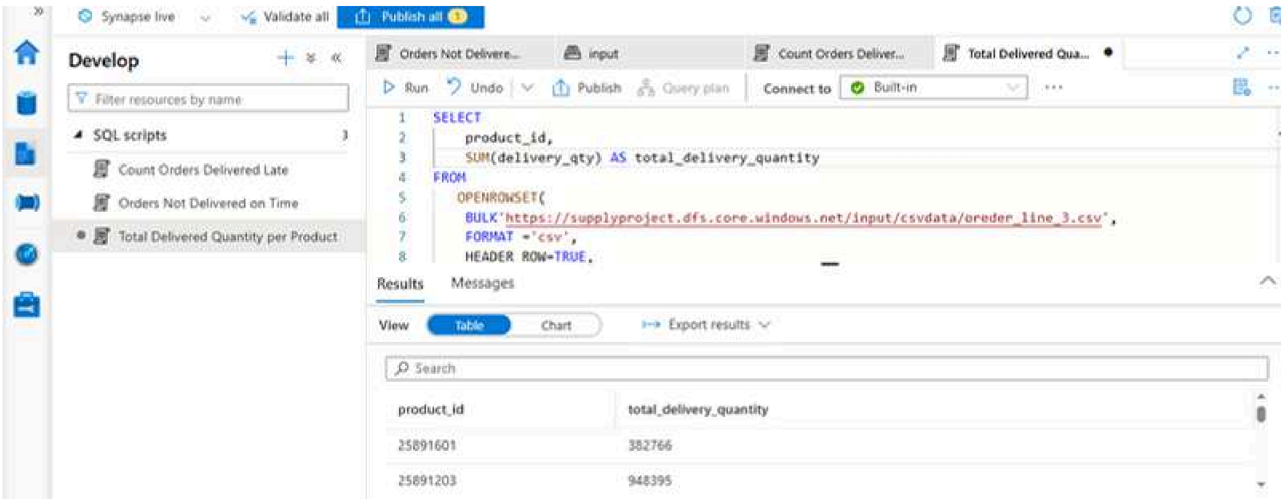
```
SELECT
    COUNT(*) AS late_orders
FROM
   OPENROWSET(
```

0
• late_orders: 15 649

Chart type
Line
Category column
(none)
Legend (series) columns
late_orders

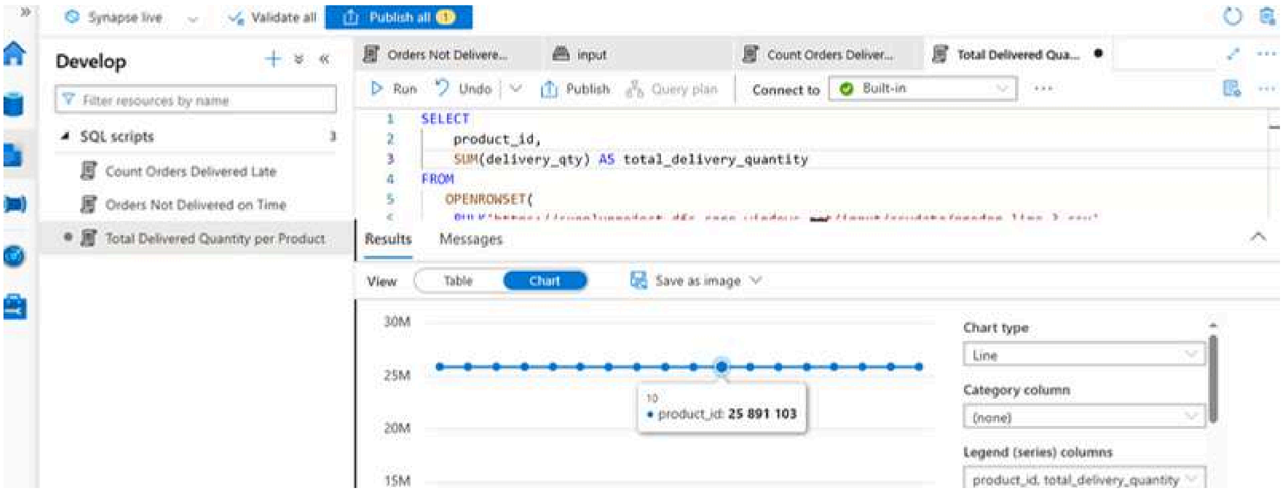
Azure Integration

3-Total delivered quantity per product



```
SELECT
    product_id,
    SUM(delivery_qty) AS total_delivery_quantity
FROM
  OPENROWSET(
    BULK'https://supplyproject.dfs.core.windows.net/input/csvdata/oreder_line_3.csv',
    FORMAT ='csv',
    HEADER_ROW=TRUE,
```

product_id	total_delivery_quantity
25891601	382766
25891203	948395

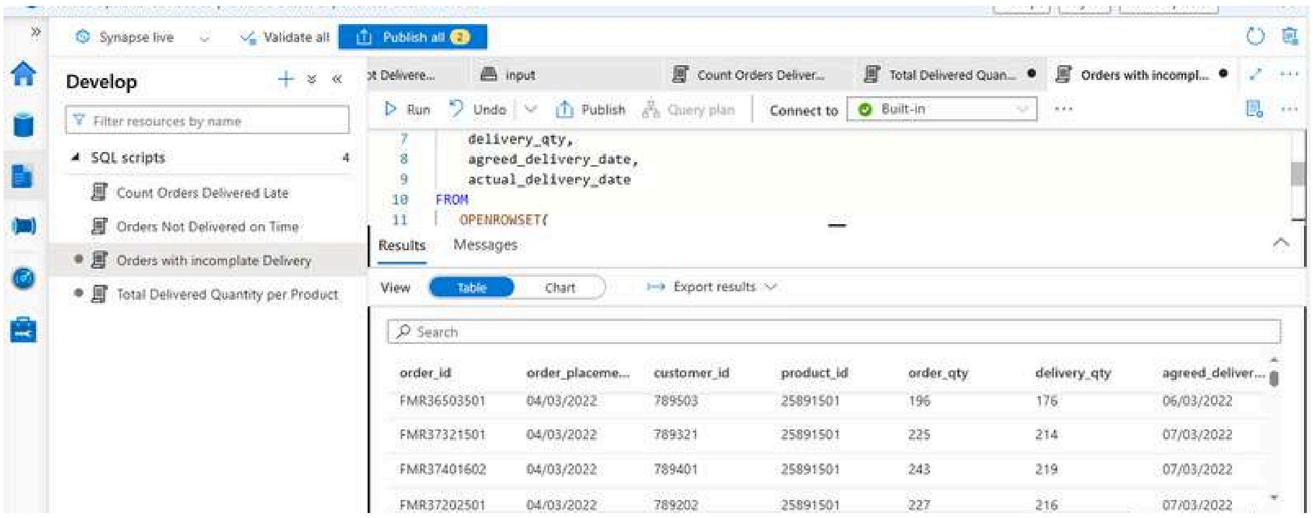


```
SELECT
    product_id,
    SUM(delivery_qty) AS total_delivery_quantity
FROM
  OPENROWSET(
```

Chart type: Line
Category column: (none)
Legend (series) columns: product_id, total_delivery_quantity

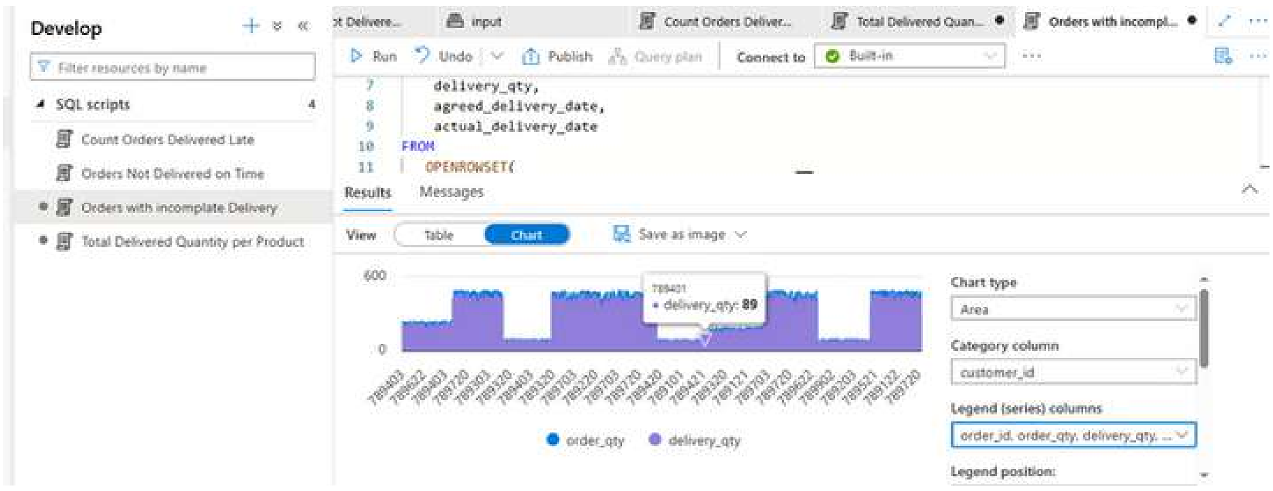
Azure Integration

4-Orders with incomplate delivery



```
    delivery_qty,
    agreed_delivery_date,
    actual_delivery_date
FROM
    OPENROWSET(
```

order_id	order_placeme...	customer_id	product_id	order_qty	delivery_qty	agreed_deliver...
FMR36503501	04/03/2022	789503	25891501	196	176	06/03/2022
FMR37321501	04/03/2022	789321	25891501	225	214	07/03/2022
FMR37401602	04/03/2022	789401	25891501	243	219	07/03/2022
FMR37202501	04/03/2022	789202	25891501	227	216	07/03/2022



```
    delivery_qty,
    agreed_delivery_date,
    actual_delivery_date
FROM
    OPENROWSET(
```

Chart type: Area

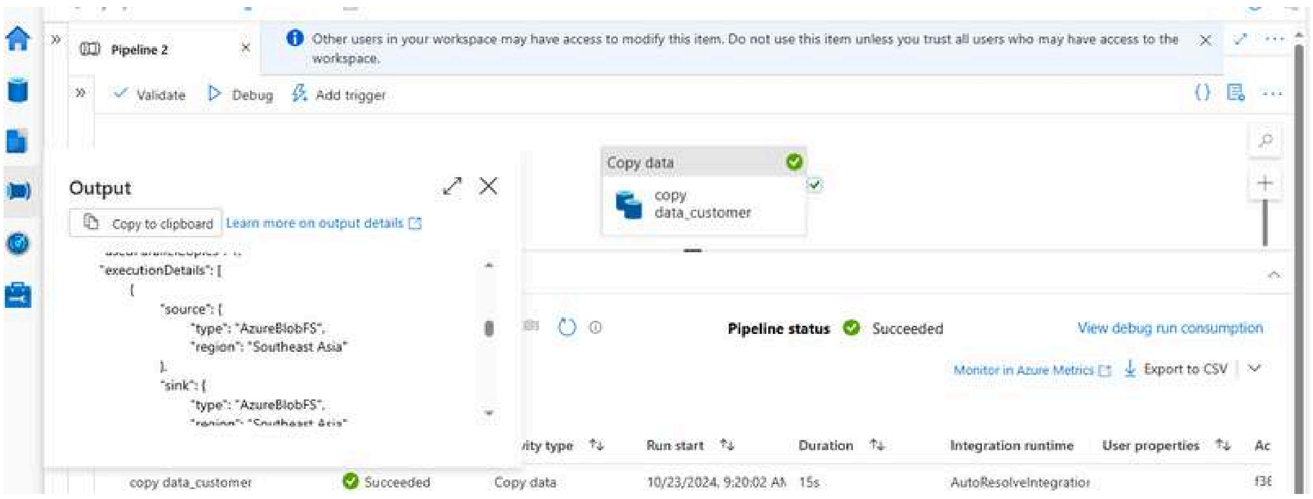
Category column: customer_id

Legend (series) columns: order_id, order_qty, delivery_qty, ...

Legend position:

Azure Integration

Using pipeline activities:
1-Copy Data

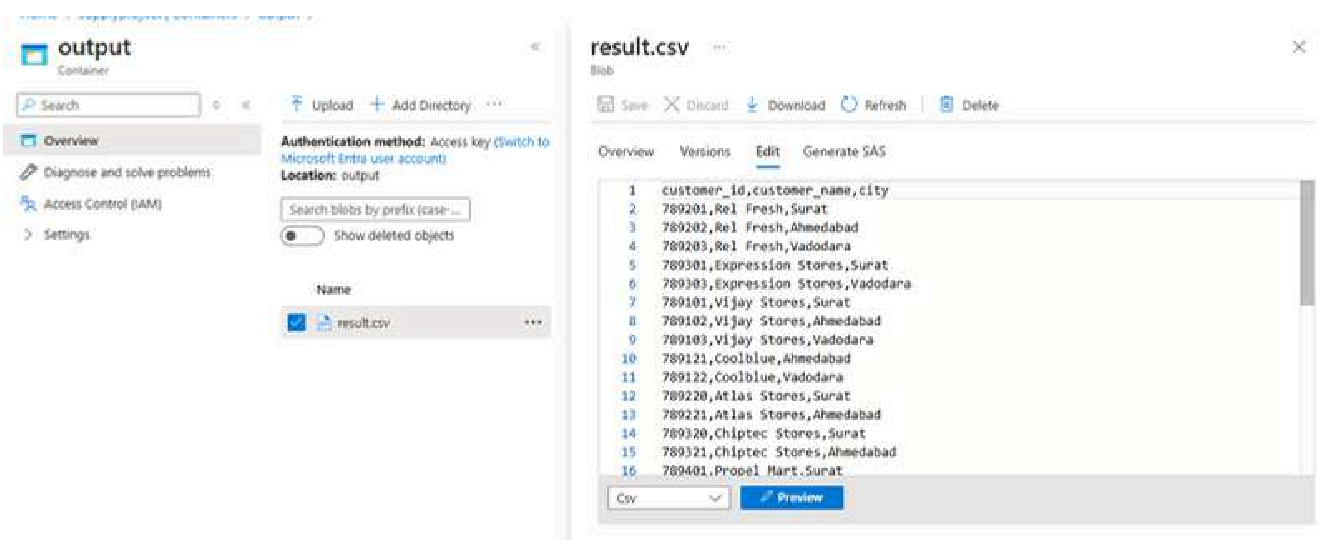


Other users in your workspace may have access to modify this item. Do not use this item unless you trust all users who may have access to the workspace.

```
"executionDetails": [
    {
        "source": {
            "type": "AzureBlobFS",
            "region": "Southeast Asia"
        },
        "sink": {
            "type": "AzureBlobFS",
```

ity type	Run start	Duration	Integration runtime	User properties	Ac
Copy data	10/23/2024, 9:20:02 AM	15s	AutoResolveIntegratior		f3ɛ

copy data_customer — Succeeded

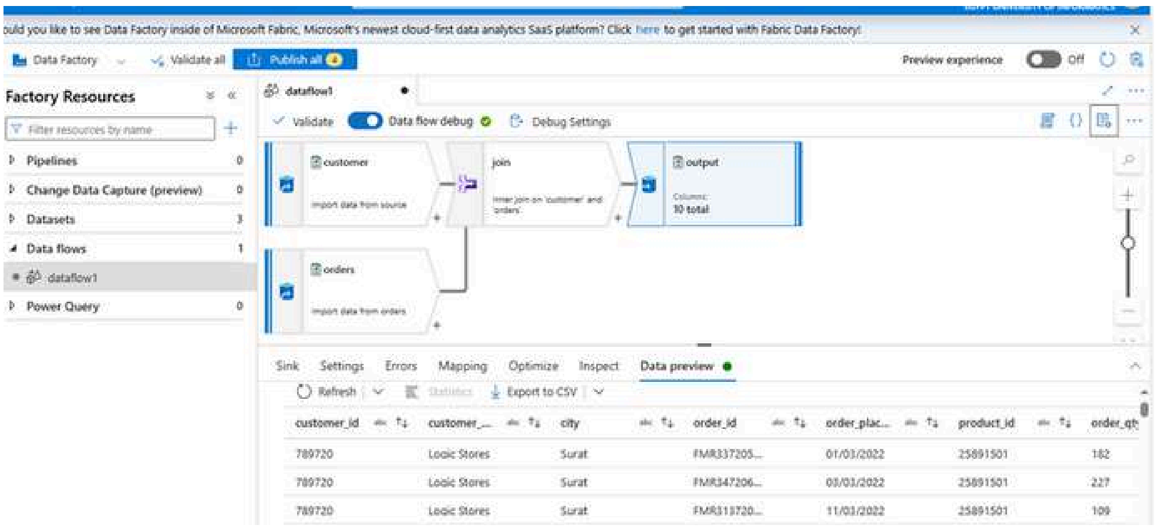


Authentication method: Access key (Switch to Microsoft Entra user account)
Location: output

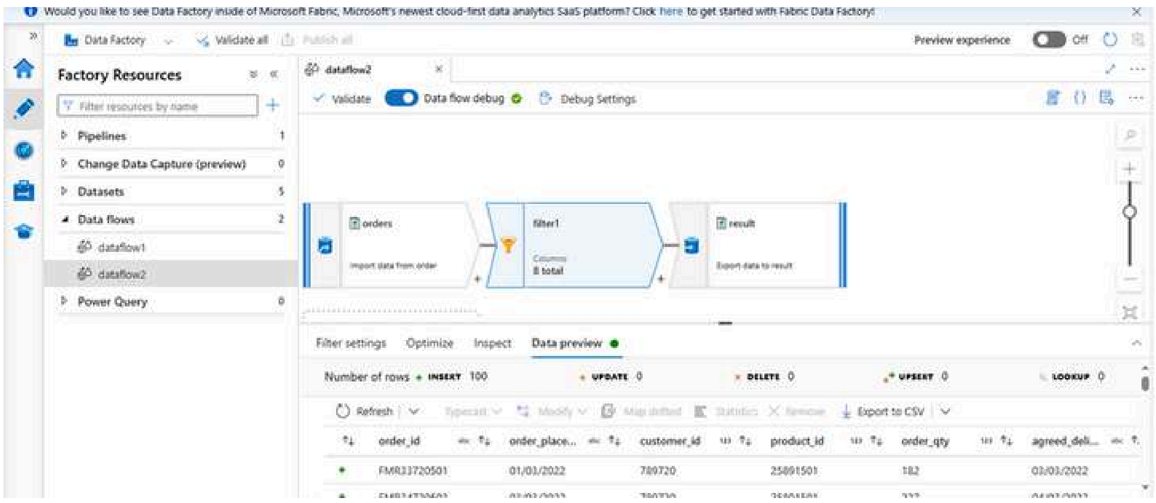
```
customer_id,customer_name,city
789201,Rel Fresh,Surat
789202,Rel Fresh,Ahmedabad
789203,Rel Fresh,Vadodara
789301,Expression Stores,Surat
789303,Expression Stores,Vadodara
789101,Vijay Stores,Surat
789102,Vijay Stores,Ahmedabad
789103,Vijay Stores,Vadodara
789121,Coolblue,Ahmedabad
789122,Coolblue,Vadodara
789220,Atlas Stores,Surat
789221,Atlas Stores,Ahmedabad
789320,Chiptec Stores,Surat
789321,Chiptec Stores,Ahmedabad
789401,Propel Mart,Surat
```

Azure Integration

Using Data Factory:
1-Data Flows

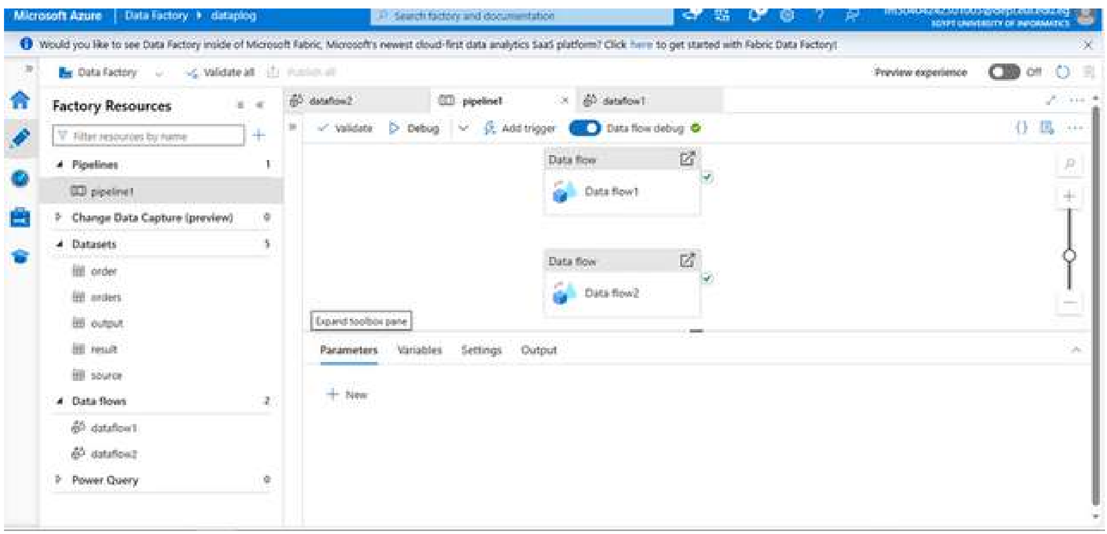


customer_id	customer_...	city	order_id	order_plac...	product_id	order_qty
789720	Logic Stores	Surat	FMR337205...	01/03/2022	25891501	182
789720	Logic Stores	Surat	FMR347206...	03/03/2022	25891501	227
789720	Logic Stores	Surat	FMR313720...	11/03/2022	25891501	109



Number of rows · INSERT 100 · UPDATE 0 · DELETE 0 · UPSERT 0 · LOOKUP 0

	order_id	order_place...	customer_id	product_id	order_qty	agreed_deli...
*	FMR33720501	01/03/2022	789720	25891501	182	03/03/2022



THANK YOU!!

instructor

Hanaa gharib

ATLIQ
GRANDS

ATLIQ

GRANDS