TRENT LIMITED
A TATA Enterprise

Retail Data Processing and Visualization Framework

- Anchal Gupta
- Harsh Dixit
- Nikita Agre
- Pratik Patil
- Rajiv Jarhad
- Abhin S Jacob
- Don Maria Das

An End-to-End Process of Data Flow

We are creating pipelines to move data to the cloud and transform it using data flows and stored procedures. This process prepares the data for the Power BI team to use for their reports.

Data Description

1) cashorder: This table contains information about orders. It has the following columns:

- **storeid**: Identifier for the store
- **customernum**: Customer number
- **invoicenumber**: Invoice number
- **invoicetype**: Type of invoice
- **itemnum**: Item number
- **localamt**: Local amount
- **cashierid**: Identifier for the cashier

2) customer_data: This table provides information about customers. It includes:

- **customernumber**: Customer number
- **name**: Customer's name
- **phone1**: Primary phone number
- **phone2**: Secondary phone number
- **email**: Email address
- **address**: Physical address

3) products_data: This table contains information about products. It has:

- **productid**: Product identifier
- **name**: Product name
- **price**: Product price

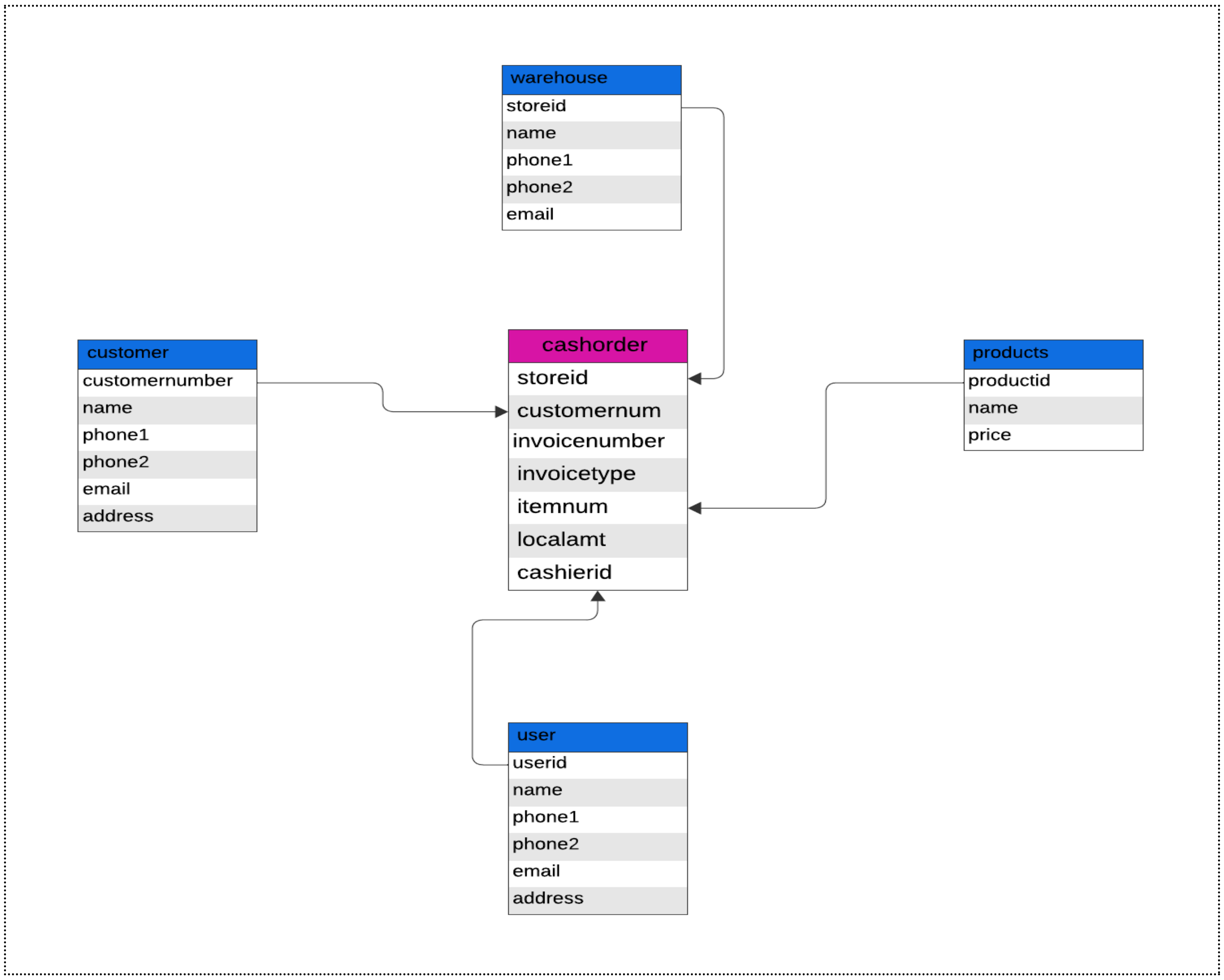
4) store_data: This table contains information about stores. It includes:

- **storeid**: Store identifier
- **name**: Store name
- **phone1**: Primary phone number
- **phone2**: Secondary phone number
- **email**: Email address

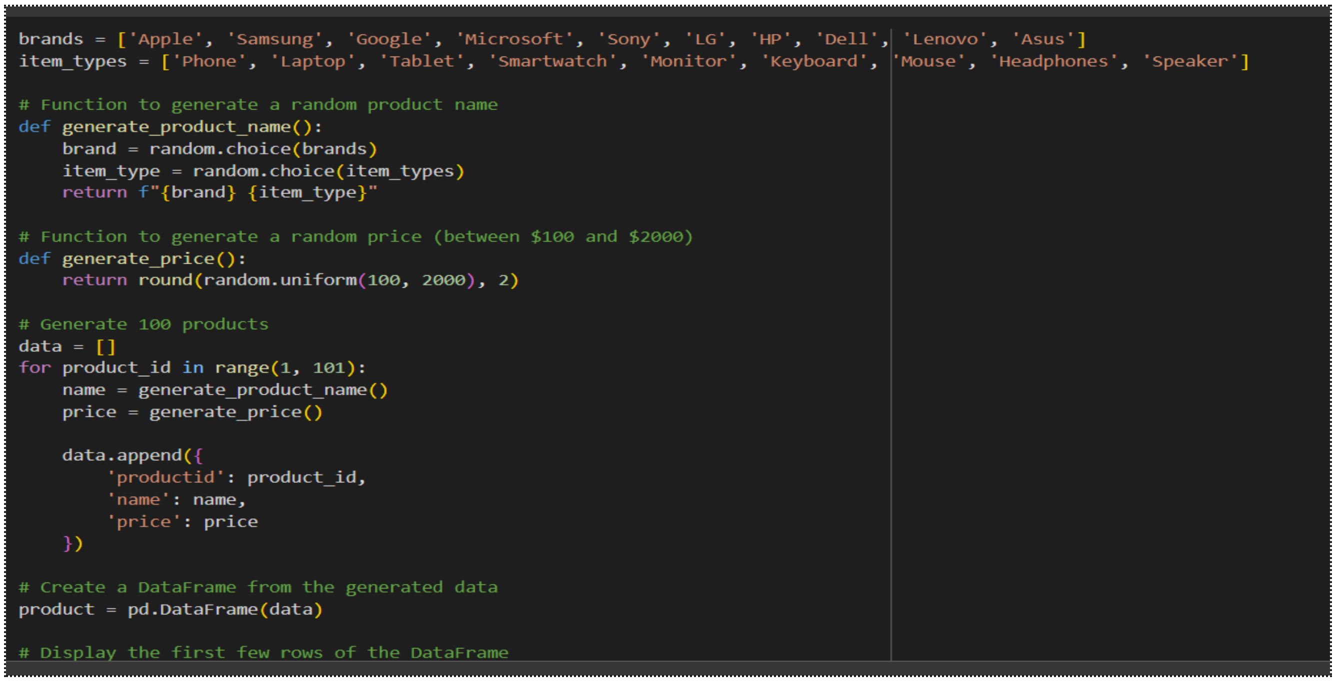
5) cashier_data: This table provides information about cashiers working at the billing desks. It has:

- **userid**: User identifier
- **name**: Cashier's name
- **phone1**: Primary phone number
- **phone2**: Secondary phone number
- **email**: Email address
- **address**: address

ER Diagram

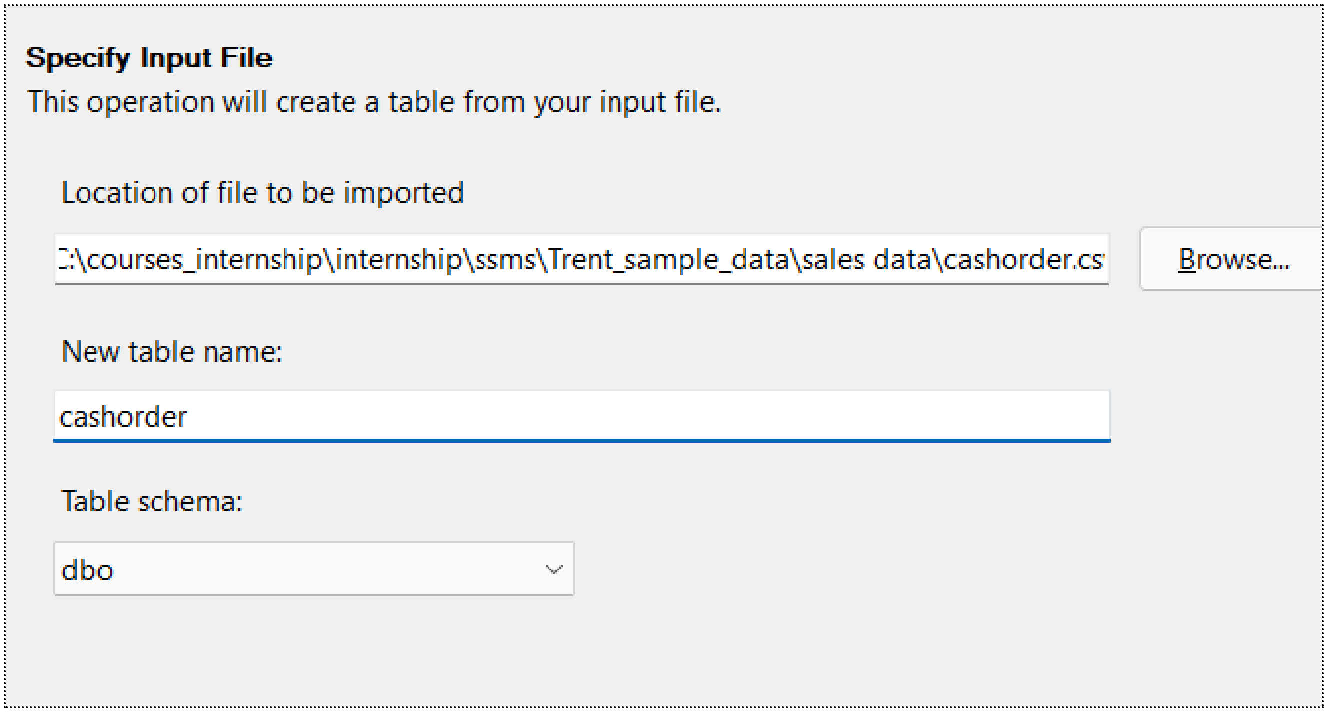


Python Code for Sample Data Creation



```
brands = ['Apple', 'Samsung', 'Google', 'Microsoft', 'Sony', 'LG', 'HP', 'Dell', 'Lenovo', 'Asus']
item_types = ['Phone', 'Laptop', 'Tablet', 'Smartwatch', 'Monitor', 'Keyboard', 'Mouse', 'Headphones', 'Speaker']

# Function to generate a random product name
def generate_product_name():
    brand = random.choice(brands)
    item_type = random.choice(item_types)
    return f"{brand} {item_type}"

# Function to generate a random price (between $100 and $2000)
def generate_price():
    return round(random.uniform(100, 2000), 2)

# Generate 100 products
data = []
for product_id in range(1, 101):
    name = generate_product_name()
    price = generate_price()

    data.append({
        'productid': product_id,
        'name': name,
        'price': price
    })

# Create a DataFrame from the generated data
product = pd.DataFrame(data)

# Display the first few rows of the DataFrame
```

Created SSMS tables using the sample data



Specify Input File

This operation will create a table from your input file.

Location of file to be imported

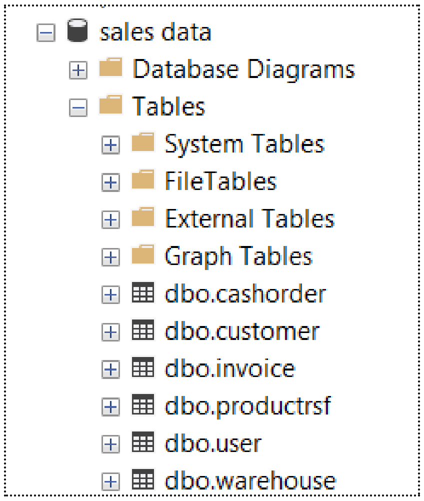
:\courses_internship\internship\ssms\Trent_sample_data\sales data\cashorder.cs

New table name:

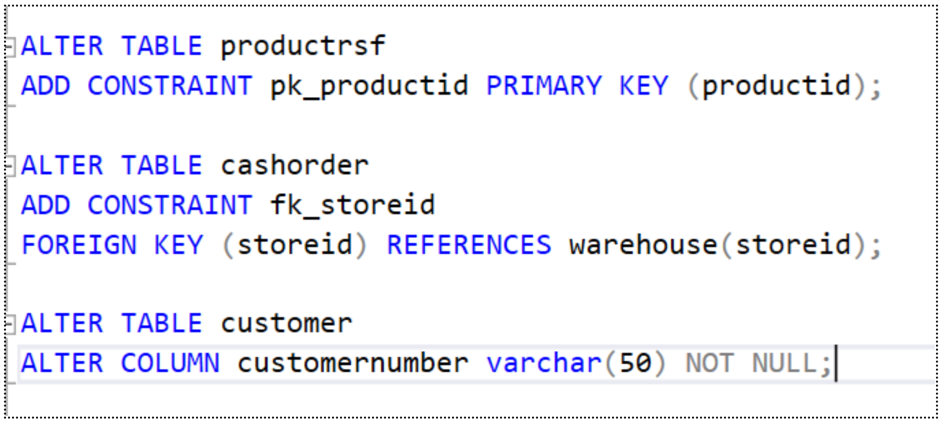
cashorder

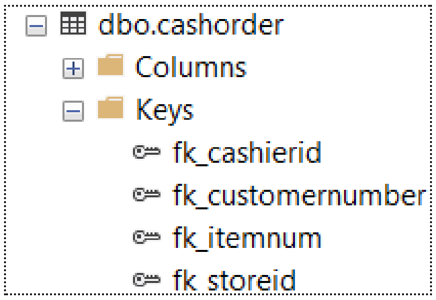
Table schema:

dbo

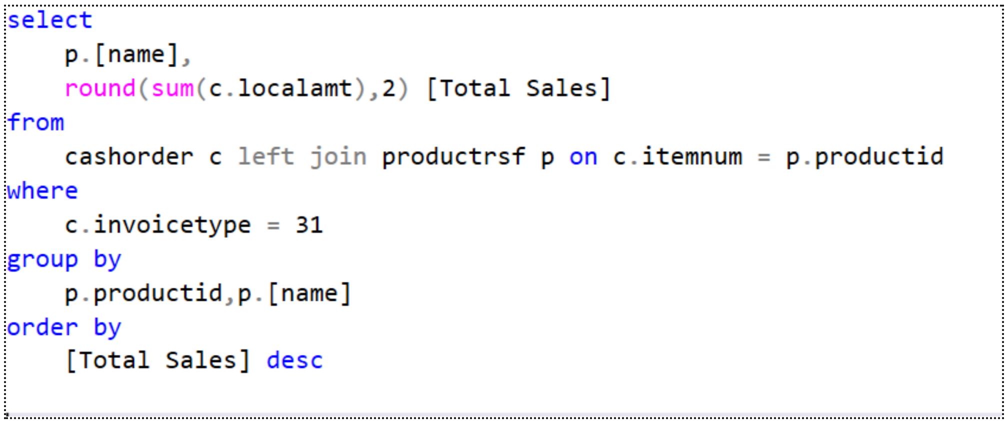

sales data
Database Diagrams
Tables
System Tables
FileTables
External Tables
Graph Tables
dbo.cashorder
dbo.customer
dbo.invoice
dbo.productrsf
dbo.user
dbo.warehouse

Created Keys and Constraints on the Table

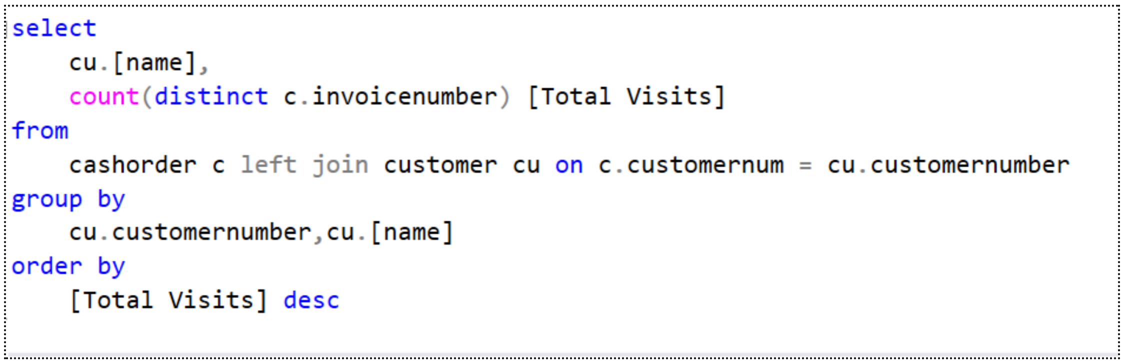

```
ALTER TABLE productrsf
ADD CONSTRAINT pk_productid PRIMARY KEY (productid);

ALTER TABLE cashorder
ADD CONSTRAINT fk_storeid
FOREIGN KEY (storeid) REFERENCES warehouse(storeid);

ALTER TABLE customer
ALTER COLUMN customernumber varchar(50) NOT NULL;
```


dbo.cashorder
Columns
Keys
fk_cashierid
fk_customernumber
fk_itemnum
fk_storeid

Retrieved Data using various report queries



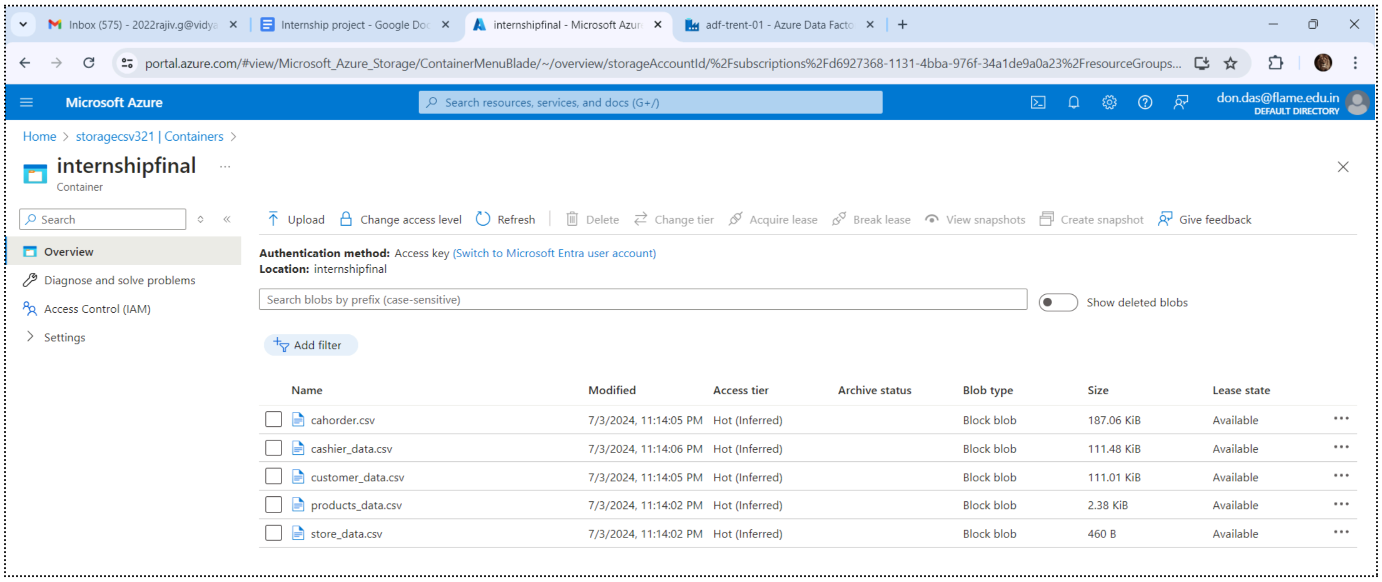
```sql
select
    p.[name],
    round(sum(c.localamt),2) [Total Sales]
from
    cashorder c left join productrsf p on c.itemnum = p.productid
where
    c.invoicetype = 31
group by
    p.productid,p.[name]
order by
    [Total Sales] desc
```



```sql
select
    cu.[name],
    count(distinct c.invoicenumber) [Total Visits]
from
    cashorder c left join customer cu on c.customernum = cu.customernumber
group by
    cu.customernumber,cu.[name]
order by
    [Total Visits] desc
```

Azure application

1. Load all these 5 files (on-premises data) in the container of the blob storage:
2. Upload the five data files from your local system to a designated folder (container) in Azure Blob Storage.


portal.azure.com/#view/Microsoft_Azure_Storage/ContainerMenuBlade/~/overview/storageAccountId/%2Fsubscriptions%2Fd6927368-1131-4bba-976f-34a1de9a0a23%2FresourceGroups...

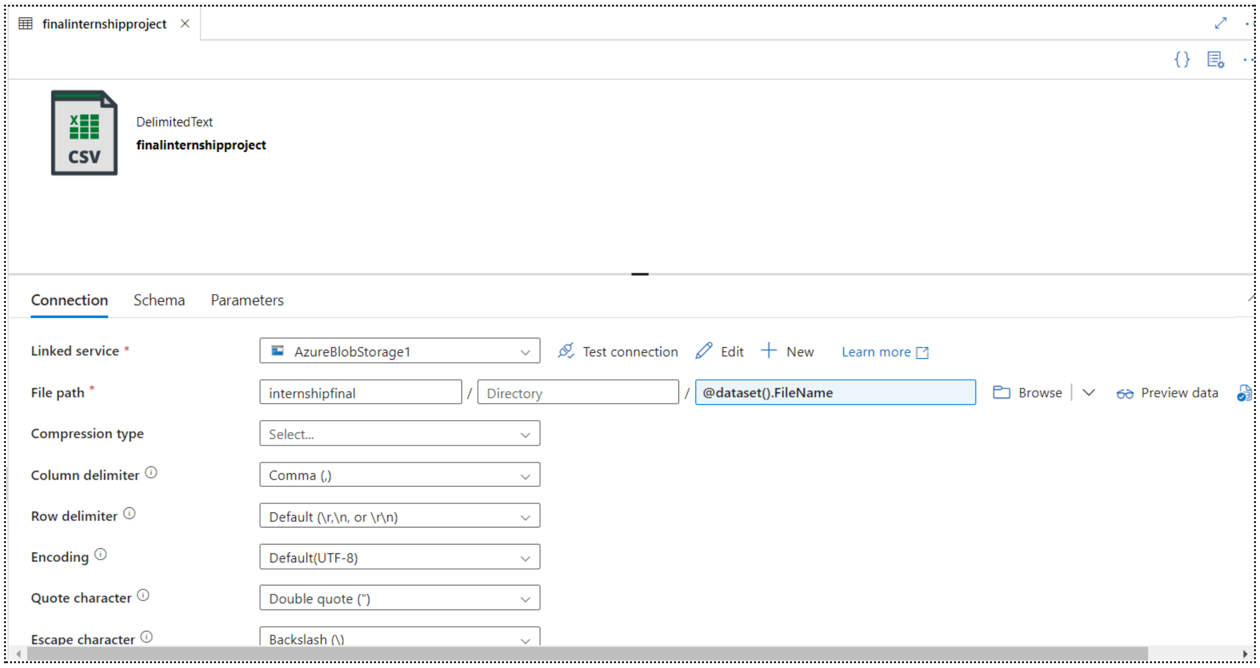
Home > storagecsv321 | Containers >

internshipfinal
Container

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

Name	Modified	Access tier	Archive status	Blob type	Size	Lease state
cahorder.csv	7/3/2024, 11:14:05 PM	Hot (Inferred)		Block blob	187.06 KiB	Available
cashier_data.csv	7/3/2024, 11:14:06 PM	Hot (Inferred)		Block blob	111.48 KiB	Available
customer_data.csv	7/3/2024, 11:14:05 PM	Hot (Inferred)		Block blob	111.01 KiB	Available
products_data.csv	7/3/2024, 11:14:02 PM	Hot (Inferred)		Block blob	2.38 KiB	Available
store_data.csv	7/3/2024, 11:14:02 PM	Hot (Inferred)		Block blob	460 B	Available

3. Create dataset- finalinternshipproject

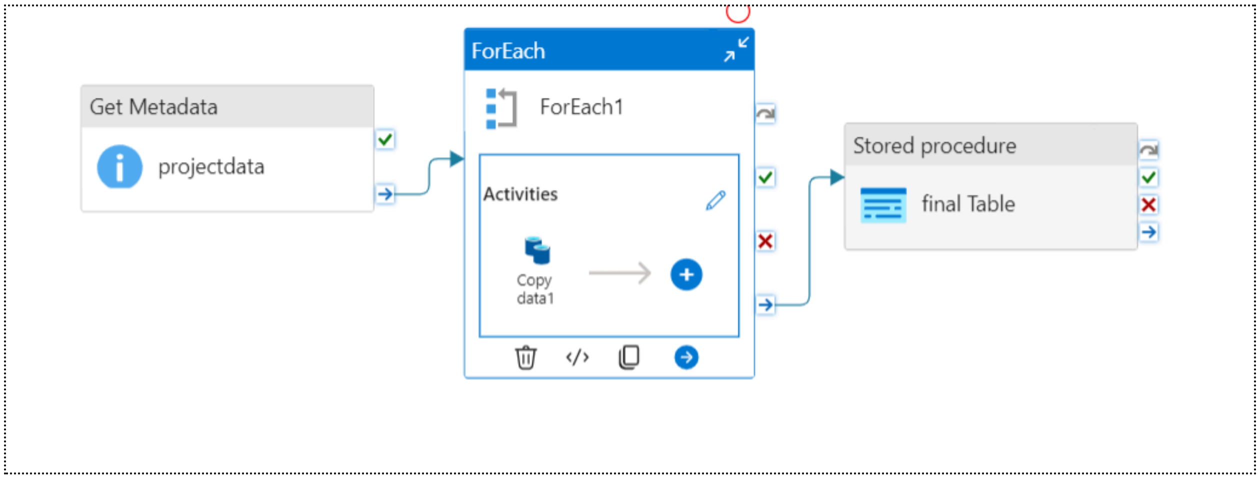

finalinternshipproject

DelimitedText
finalinternshipproject

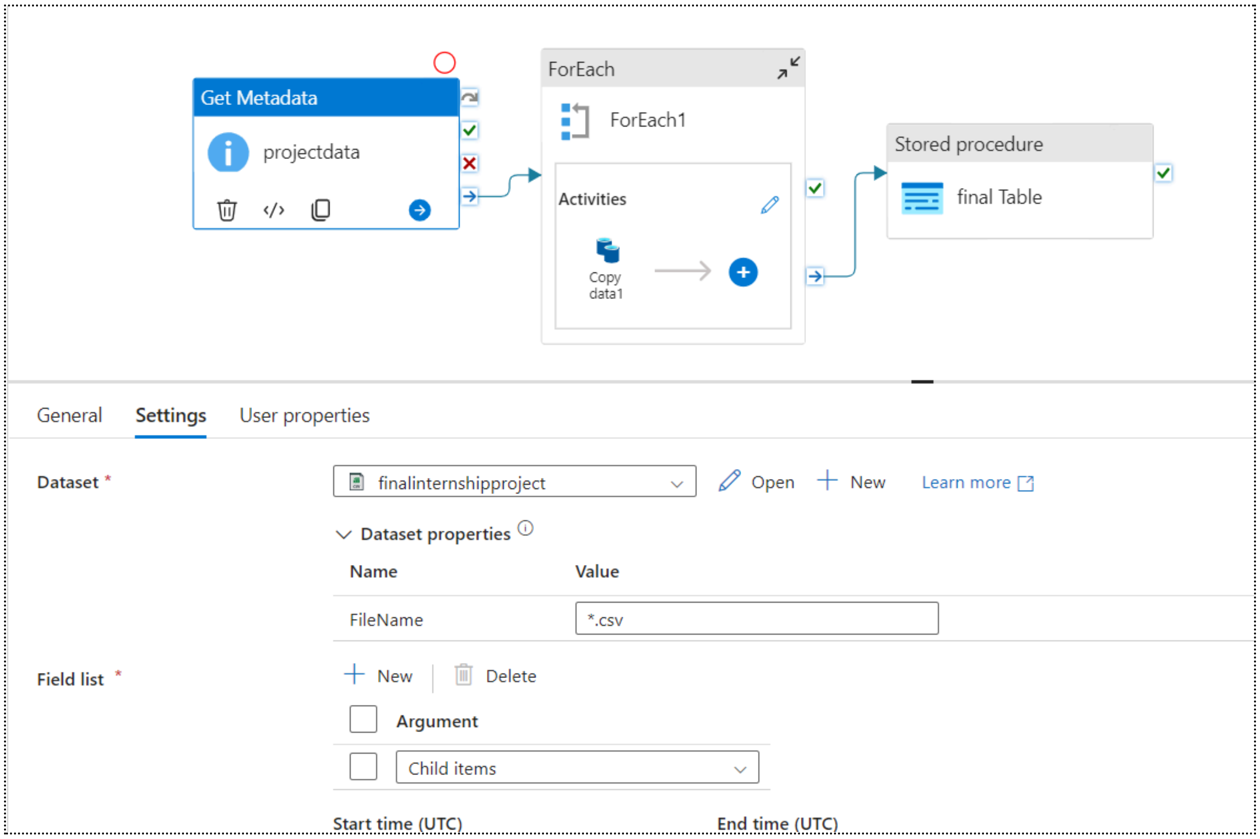
Connection | Schema | Parameters

Linked service *	AzureBlobStorage1
File path *	internshipfinal / Directory / @dataset().FileName
Compression type	Select...
Column delimiter	Comma (,)
Row delimiter	Default (\r,\n, or \r\n)
Encoding	Default(UTF-8)
Quote character	Double quote (")
Escape character	Backslash (\)

4. Develop a pipeline which moves all data to an azure SQL database and then join tables with stored procedures.



- Open a new pipeline. Drag meta data activity which fetches all 5 tables. Add child items in a field list.



Dataset * finalinternshipproject

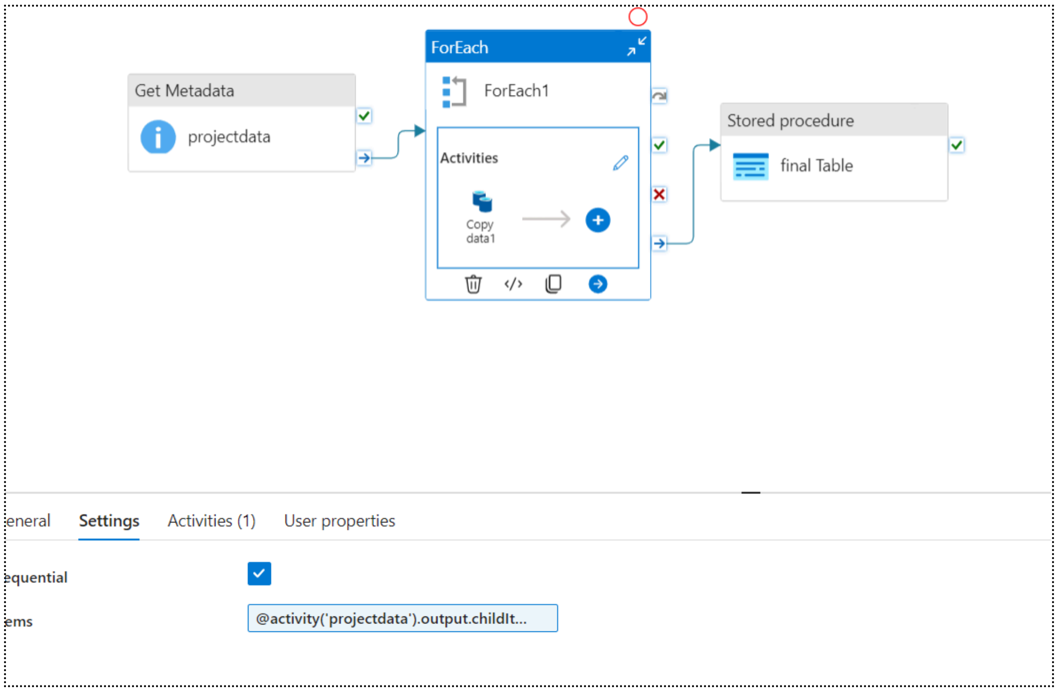
Dataset properties

Name	Value
FileName	*.csv

Field list *

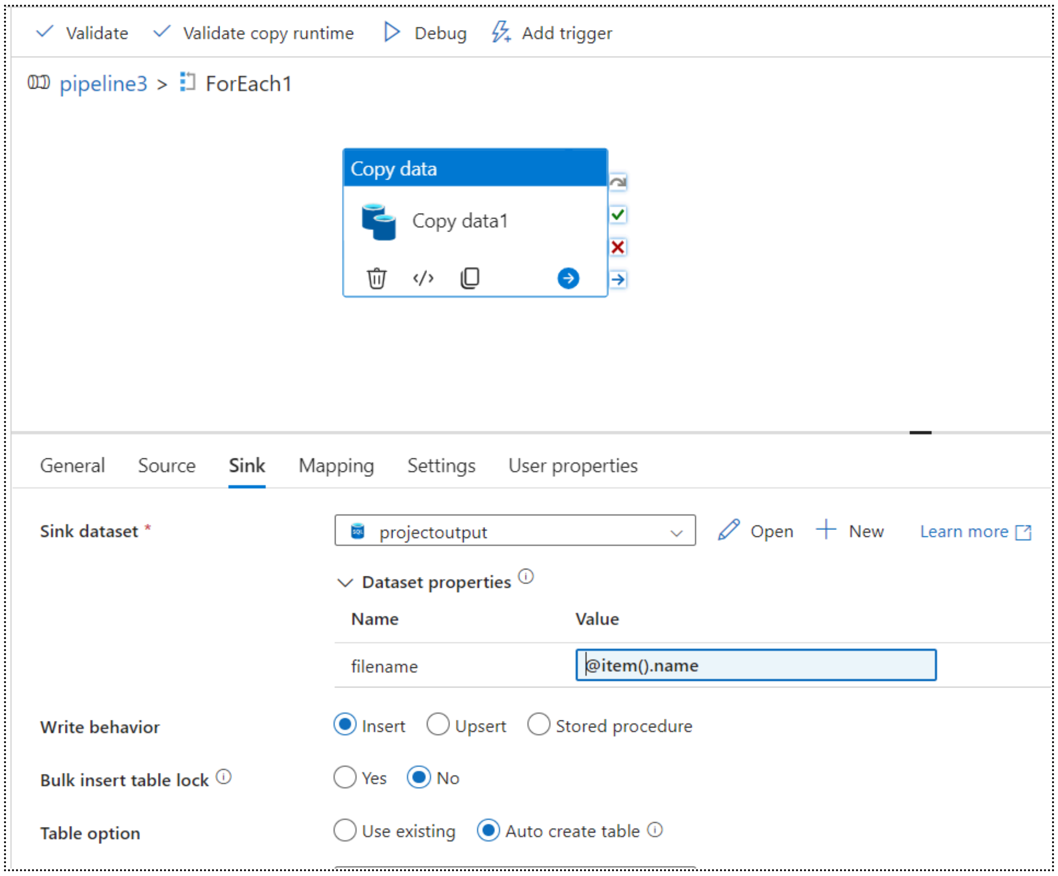
☐	Argument
☐	Child items

- Get another activity ForEach and connect to Get Metadata activity. Give output of Get Metadata activity to the ForEach activity run the copy activity inside it which copies data to Azure SQL.



equential ✓

ems @activity('projectdata').output.childIt...



Sink dataset * projectoutput

Open New Learn more

Dataset properties

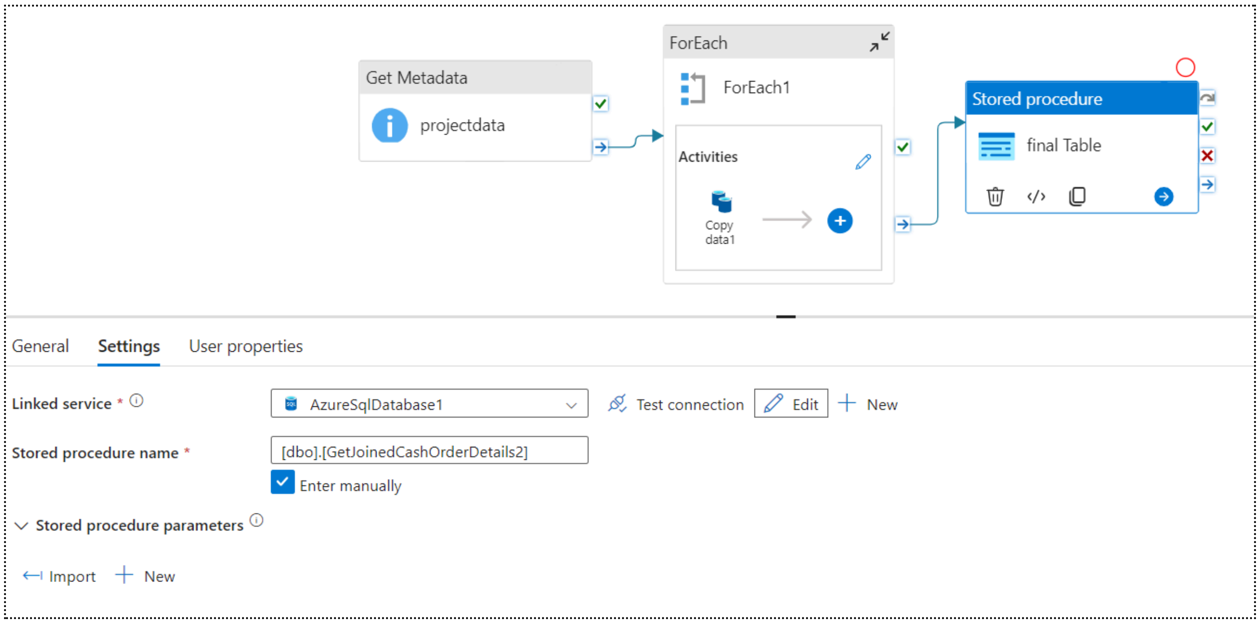
Name	Value
filename	@item().name

Write behavior [✓ Insert] Upsert Stored procedure

Bulk insert table lock Yes [✓ No]

Table option Use existing [✓ Auto create table]

5. Run the stored procedure which joins all these 5 tables.



General | Settings | User properties

Linked service *: AzureSqlDatabase1 — Test connection, Edit, New

Stored procedure name *: [dbo].[GetJoinedCashOrderDetails2]

☑ Enter manually

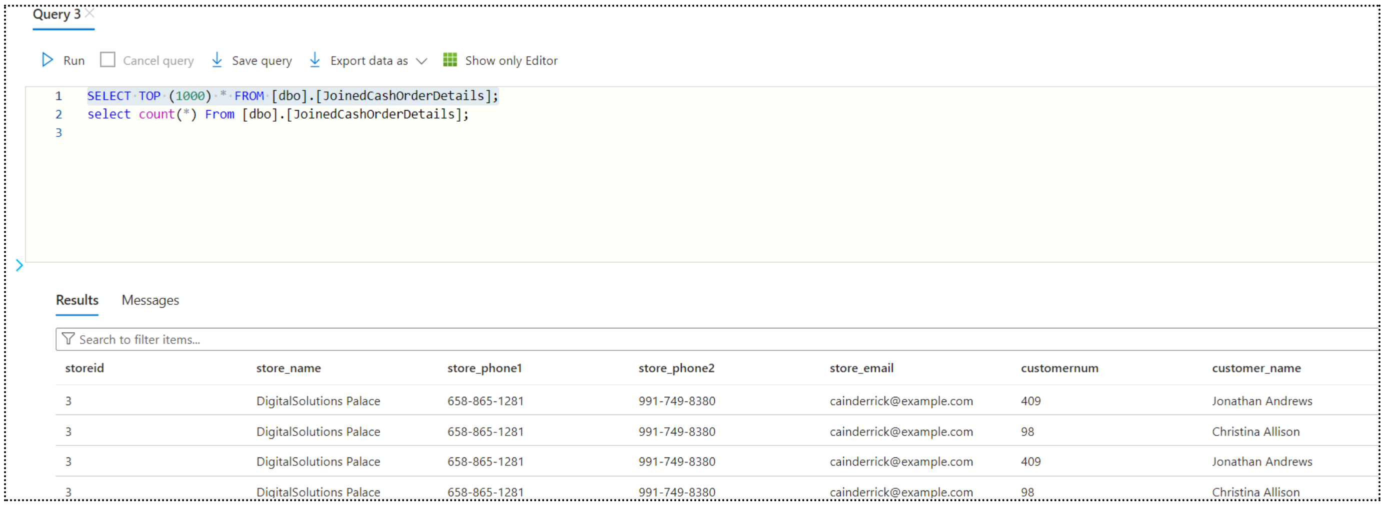
Stored procedure parameters

Import | New

```
CREATE PROCEDURE GetJoinedCashOrderDetails2
AS
BEGIN
    CREATE TABLE JoinedCashOrderDetails (
    storeid NVARCHAR(50),
    store_name NVARCHAR(255),
    store_phone1 NVARCHAR(50),
    store_phone2 NVARCHAR(50),
    store_email NVARCHAR(255),
    customernum NVARCHAR(255),
    customer_name NVARCHAR(255),
    customer_phone1 NVARCHAR(50),
    customer_phone2 NVARCHAR(50),
    customer_email NVARCHAR(255),
    customer_address NVARCHAR(255),
    invoicenumber NVARCHAR(50),
    invoicetype NVARCHAR(50),
    itemnum NVARCHAR(50),
    product_name NVARCHAR(255),
    product_price NVARCHAR(255),
    localamt NVARCHAR(255),
    cashierid NVARCHAR(255),
    cashier_name NVARCHAR(255),
    cashier_phone1 NVARCHAR(50),
    cashier_phone2 NVARCHAR(50),
        cashier_email NVARCHAR(255),
        cashier_address NVARCHAR(255)
    );

     insert into JoinedCashOrderDetails
        SELECT
            co.storeid,
            s.name AS store_name,
            s.phone1 AS store_phone1,
            s.phone2 AS store_phone2,
            s.email AS store_email,
            co.customernum,
            c.name AS customer_name,
            c.phone1 AS customer_phone1,
            c.phone2 AS customer_phone2,
            c.email AS customer_email,
            c.address AS customer_address,
            co.invoicenumber,
            co.invoicetype,
            co.itemnum,
            p.name AS product_name,
            p.price AS product_price,
            co.localamt,
            co.cashierid,
            ca.name AS cashier_name,
            ca.phone1 AS cashier_phone1,
            ca.phone2 AS cashier_phone2,
            ca.email AS cashier_email,
            ca.address AS cashier_address
        FROM
            CASHORDER co
        LEFT JOIN
            store s ON co.storeid = s.storeid
        LEFT JOIN
            customer c ON co.customernum = c.customernumber
        LEFT JOIN
            products p ON co.itemnum = p.productid
        LEFT JOIN
            cashier ca ON co.cashierid = ca.userid;
    END;
```

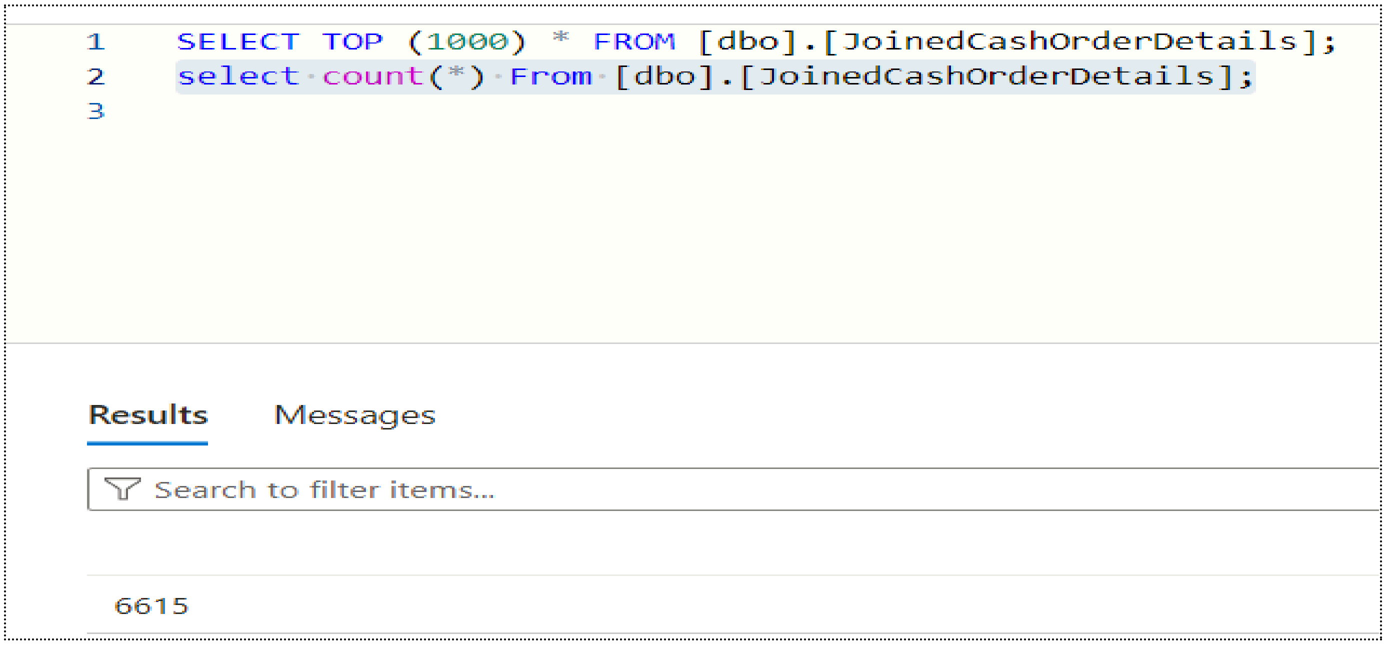
Join all these 5 tables based on following conditions:
cashorder.customernum== customer_data.customernumber
cashorder.productid== products_data.productid
cashorder.storeid== store_data.storeid
cashorder.cashiersid== cashier_data.userid

Final merged file has 6615 rows with 23 columns which get stored in the SQL database.



```sql
SELECT TOP (1000) * FROM [dbo].[JoinedCashOrderDetails];
select count(*) From [dbo].[JoinedCashOrderDetails];
```

storeid	store_name	store_phone1	store_phone2	store_email	customernum	customer_name
3	DigitalSolutions Palace	658-865-1281	991-749-8380	cainderrick@example.com	409	Jonathan Andrews
3	DigitalSolutions Palace	658-865-1281	991-749-8380	cainderrick@example.com	98	Christina Allison
3	DigitalSolutions Palace	658-865-1281	991-749-8380	cainderrick@example.com	409	Jonathan Andrews
3	DigitalSolutions Palace	658-865-1281	991-749-8380	cainderrick@example.com	98	Christina Allison



```sql
SELECT TOP (1000) * FROM [dbo].[JoinedCashOrderDetails];
select count(*) From [dbo].[JoinedCashOrderDetails];
```

```
6615
```

As this data doesn't need any transformation, connect Power BI to the SQL server and start making reports.

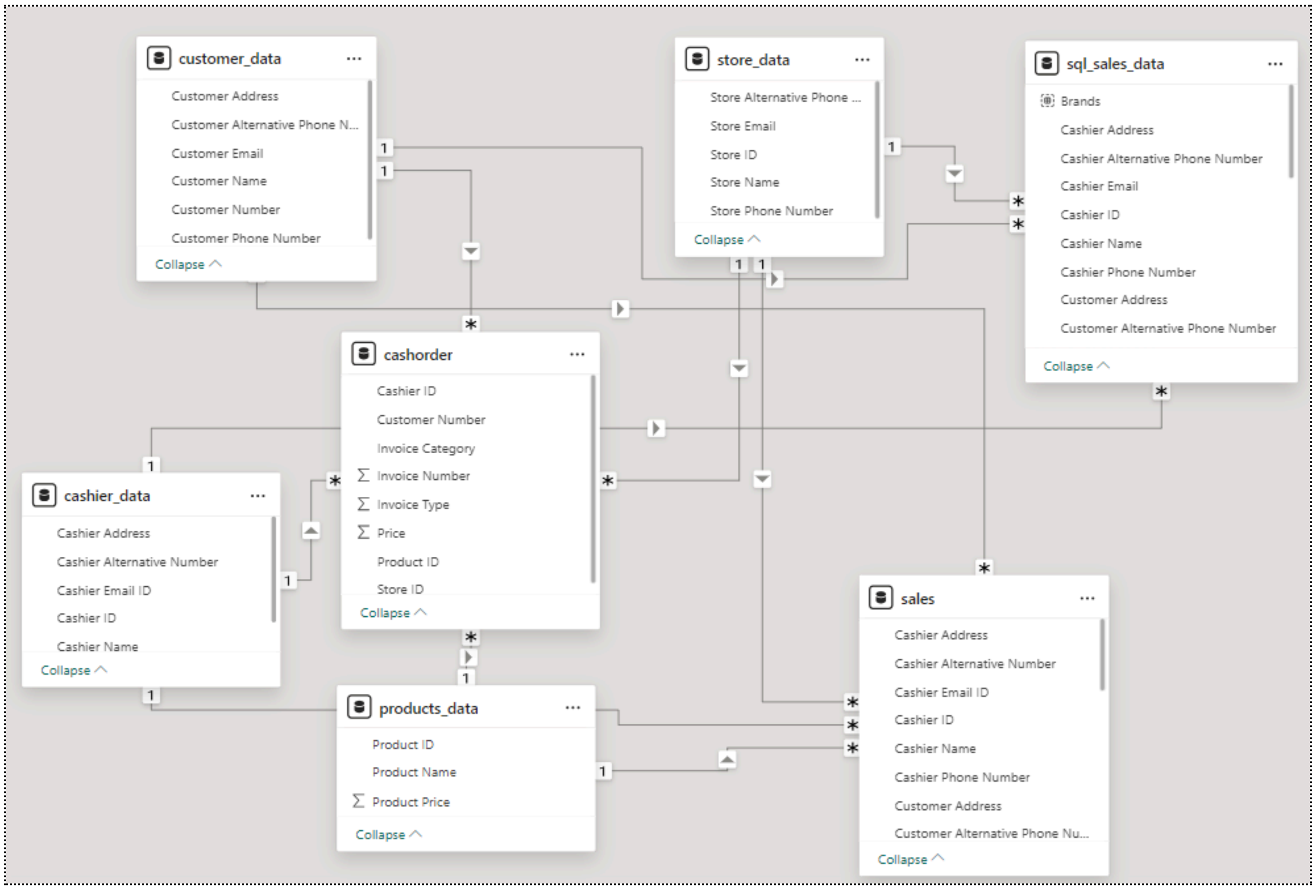
Challenges:

- Connection Issues and Data Source Configuration
- Data Quality Issues
- User Access Management
- Understanding Power BI Features
- SQL Queries Complexity
- Integrating with Other Tools such as Power BI

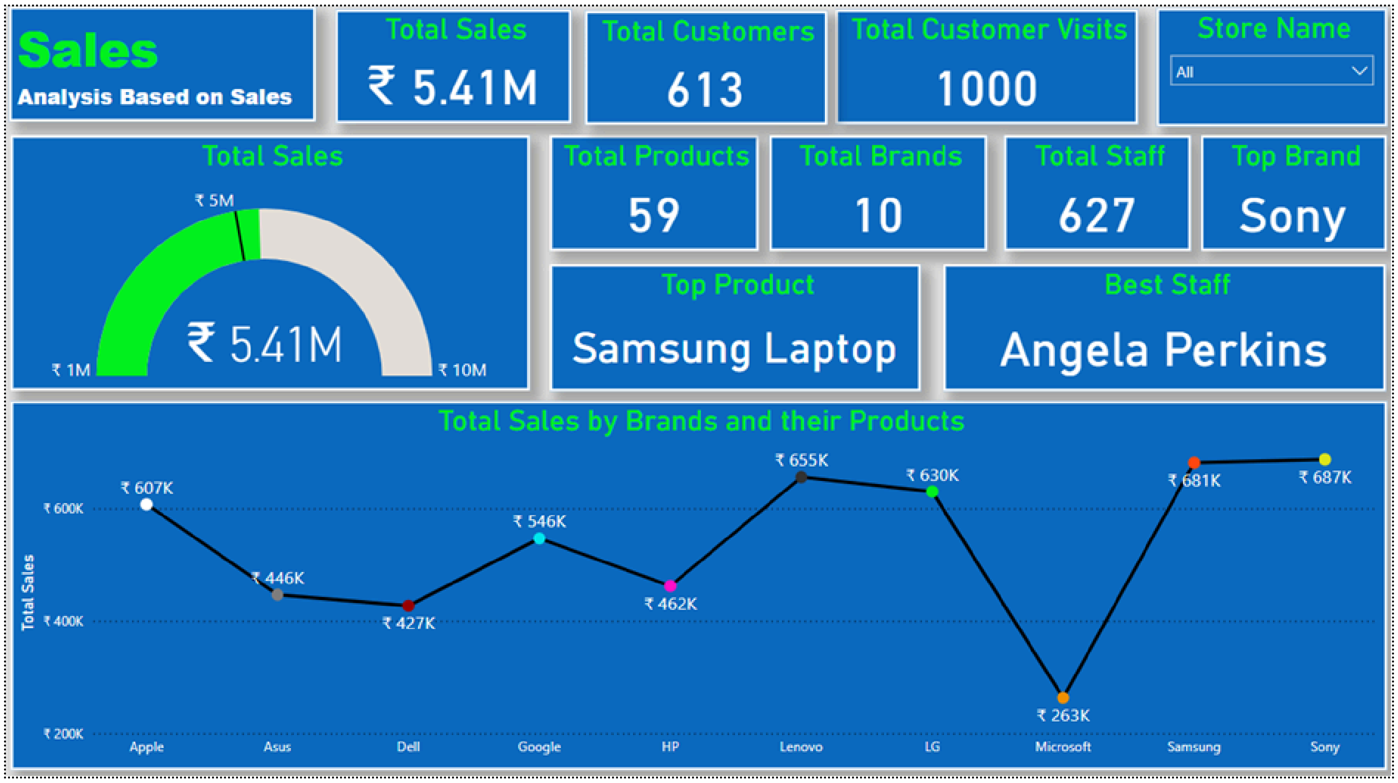
Power BI Application

In Power BI, we did the basic transformation in Power Query Editor like renaming the columns for better understanding, creating another column using the existing column where we have invoice type as a whole number (31 means Individual, 30 means final) and sorting of columns in ascending or descending to get proper insights. We used close and apply to move the final table from Power Query Editor to Power BI Desktop.

We grouped the data from one column to create another column for hierarchy visualization.

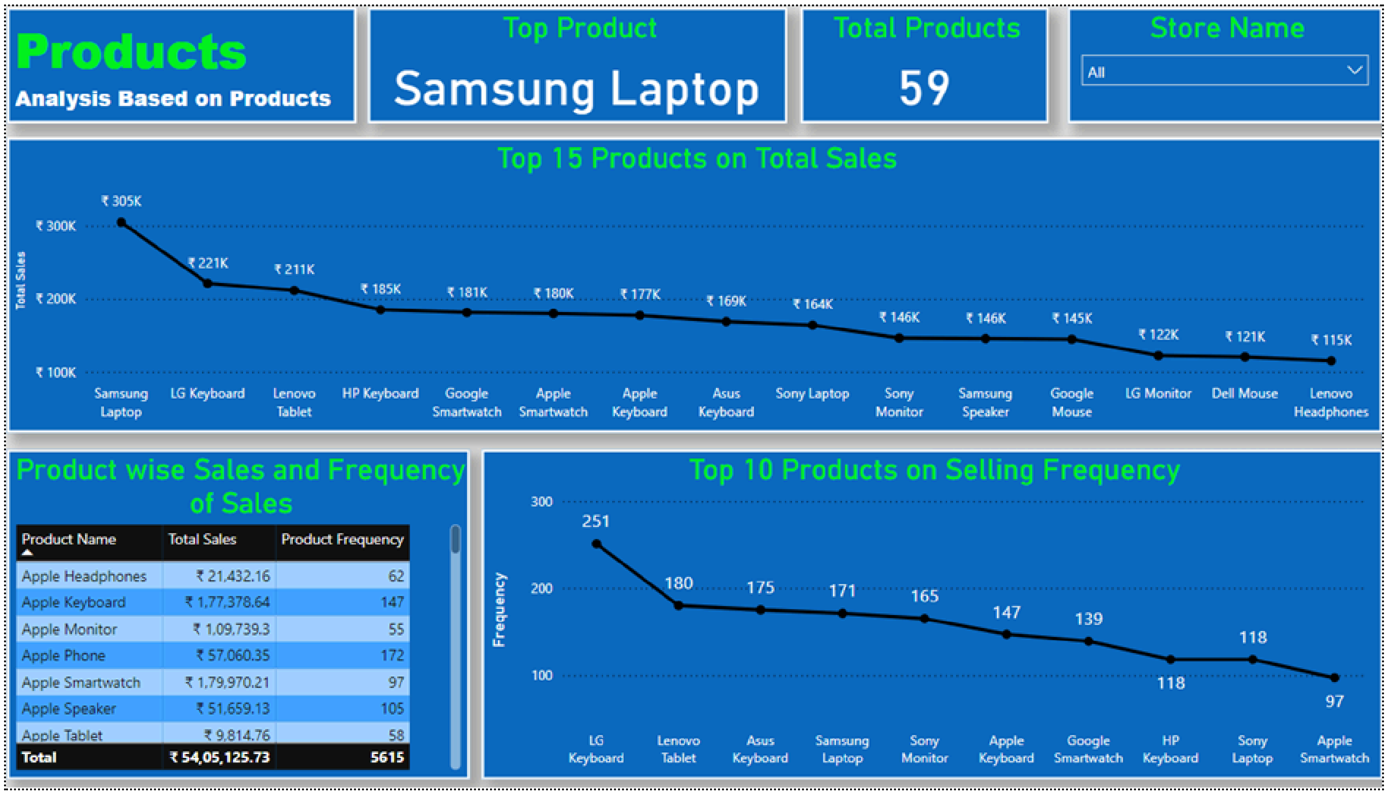


- **Sales Overview**



This is showing the total sales: ₹5.41 million. There are 613 customers and 1000 customer visits which shows there are some customers who visited the store multiple times, with 59 total products and 10 brands available. The top product is a Samsung Laptop, and the best staff member is Angela Perkins. The total sales by brands are plotted on a graph, with Sony leading at ₹687,000.

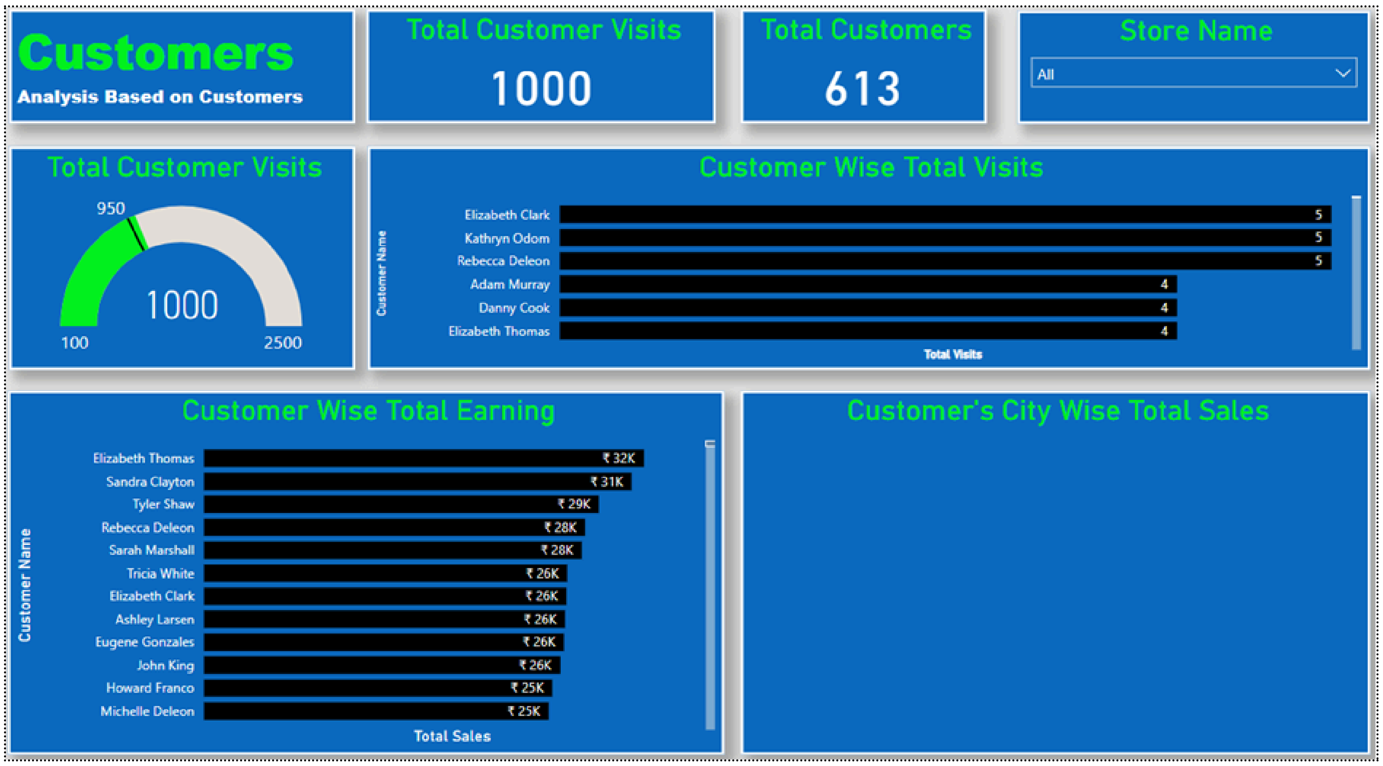
- **Products**



Product Name	Total Sales	Product Frequency
Apple Headphones	₹ 21,432.16	62
Apple Keyboard	₹ 1,77,378.64	147
Apple Monitor	₹ 1,09,739.3	55
Apple Phone	₹ 57,060.35	172
Apple Smartwatch	₹ 1,79,970.21	97
Apple Speaker	₹ 51,659.13	105
Apple Tablet	₹ 9,814.76	58
Total	**₹ 54,05,125.73**	**5615**

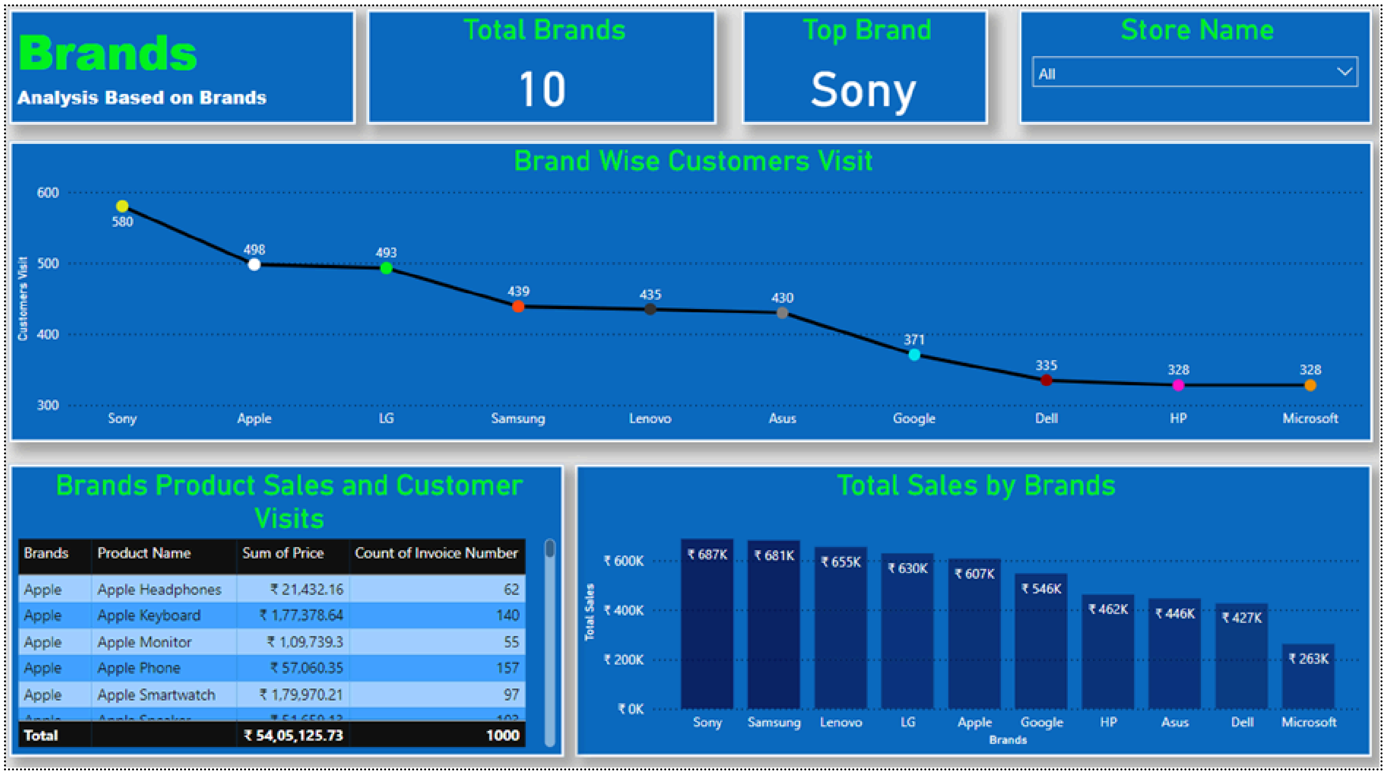
It displays product analysis for a store, highlighting the sales and frequency of various products. The top-selling product is a Samsung Laptop, and there are 59 products in total. The top 15 products by total sales are plotted on a graph, with samsung laptops leading at ₹3,05,000. Additionally, the product-wise sales and frequency of sales are listed, with LG Keyboards having the highest selling frequency at 251 units.

- **Customers**



This report is focusing on customer analysis for a store, displaying a total of 1000 customer visits and 613 total customers. The top customers by total visits are listed, with Elizabeth Clark, Kathryn Odom, and Rebecca Deleon having the highest number of visits at 5 each. Customer-wise total earnings show Elizabeth Thomas leading with ₹32K, followed closely by Sandra Clayton with ₹31K. It also includes a map for customer city-wise total sales to show the regions.

- **Brands**

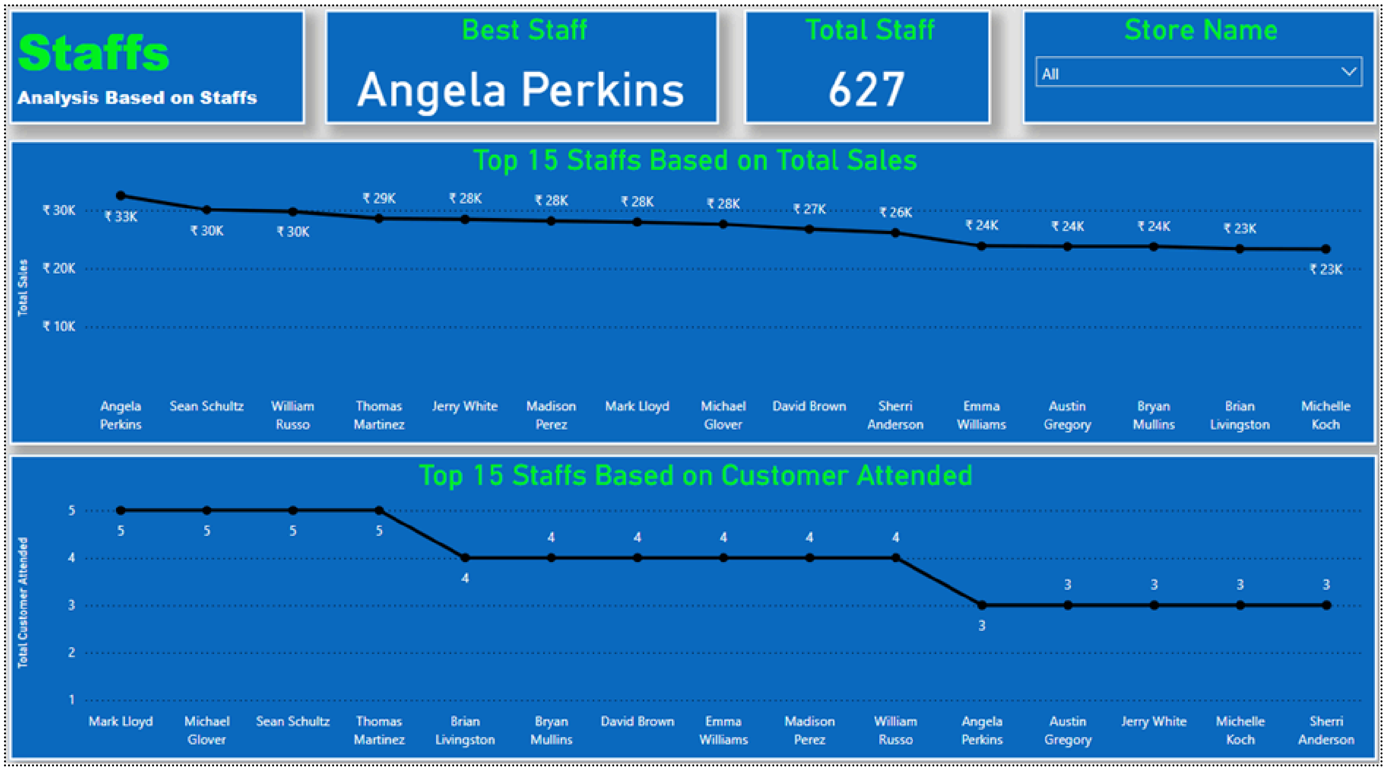


Brands Product Sales and Customer Visits

Brands	Product Name	Sum of Price	Count of Invoice Number
Apple	Apple Headphones	₹ 21,432.16	62
Apple	Apple Keyboard	₹ 1,77,378.64	140
Apple	Apple Monitor	₹ 1,09,739.3	55
Apple	Apple Phone	₹ 57,060.35	157
Apple	Apple Smartwatch	₹ 1,79,970.21	97
Apple	Apple Speakers	[illegible]	[illegible]
Total		**₹ 54,05,125.73**	**1000**

This report is showing the analyzing of brand performance in a retail context. It shows there are 10 total brands, with Sony being the top brand. The line chart illustrates brand-wise customer visits, where Sony leads with around 580 visits, followed by a declining trend for other brands. The table provides details of product sales and customer visits. And a bar graph showing total sales by brand, with Sony again leading in sales volume.

- **Staffs**



This represents an analysis of staff performance. It identifies Angela Perkins as the best staff member out of a total of 627 employees. The line graph showing the top 15 staff members based on total sales, with Angela Perkins leading at around ₹33K. And the below line chart displaying the top 15 staff based on customers attended, where Mark Lloyd, Michael Glover, Sean Schultz, Thomas Martinez, and Brian Livingston are tied for the highest number of customers. Each one attended a total of 5 customers.

Conclusion

This project effectively illustrates a data pipeline process from beginning to end for moving on-premises data to the cloud, transforming it with Azure Data Factory, and enabling Power BI reporting on that data. This project successfully demonstrates how data ingestion, transformation, and reporting procedures can be streamlined with cloud-based data solutions. . By leveraging Azure Data Factory and Power BI, we created a scalable and efficient pipeline that provides valuable business insights. The comprehensive reports generated in Power BI offer a clear view of sales, product performance, customer behavior, brand success, and staff efficiency, empowering decision-makers with actionable data.