Final Project

Objective: To build a data solutions using different Azure Technologies for different data realated challenges and comparing them including Cosmos DB, Dedicated SQL Pool, Spark Pool, Event Hubs & Stream Analytics.

Project outline:

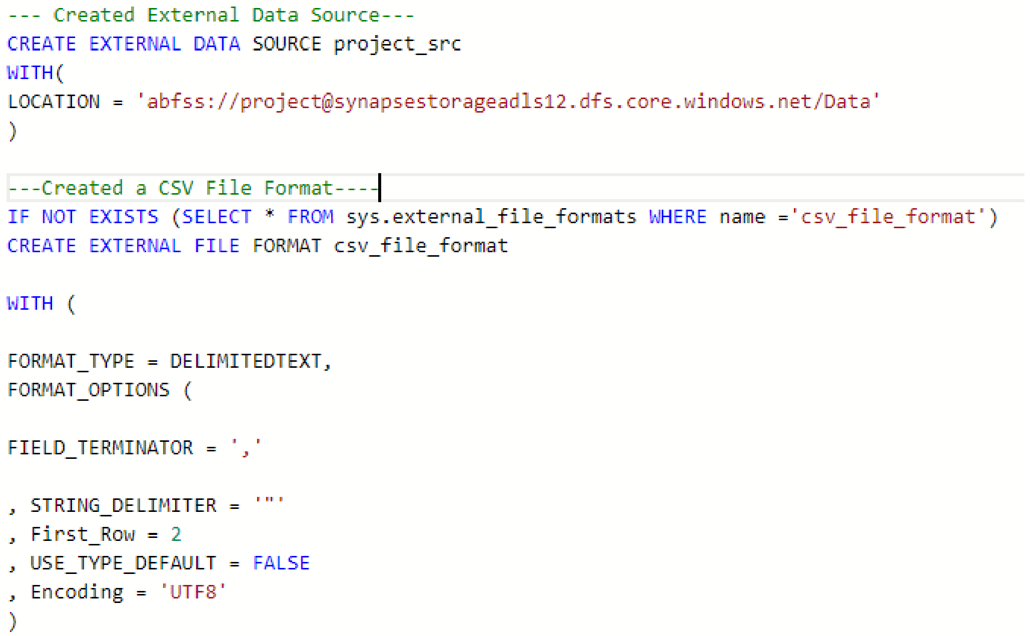
- Technologies used -
  - Azure Cosmos DB – NoSQL database for real-time data storage.
  - Azure Dedicated SQL Pool – Data warehousing for structured data.
  - Azure Synapse Spark Pool – Large-scale data processing with Spark.
  - Azure Event Hub – Real-time data ingestion.
  - Azure Stream Analytics – Stream processing for real-time analytics.
  - Azure Synapse Serverless Pool – SQL query engine in Synapse

- Prerequisites – Setting up technologies for each and using the created ADLS storage and container 'project' for the source data – business_employment.csv
- Data Ingestion & Preparation – Used Event Hubs to ingest the data in real time
- Data Transformation & Cleaning – Used Stream jobs, Spark pool, Dedicated SQL Pool, Serverless
- Conclusion

Dedicated SQL Pool: Perform large-scale batch processing and analytics on structured data.

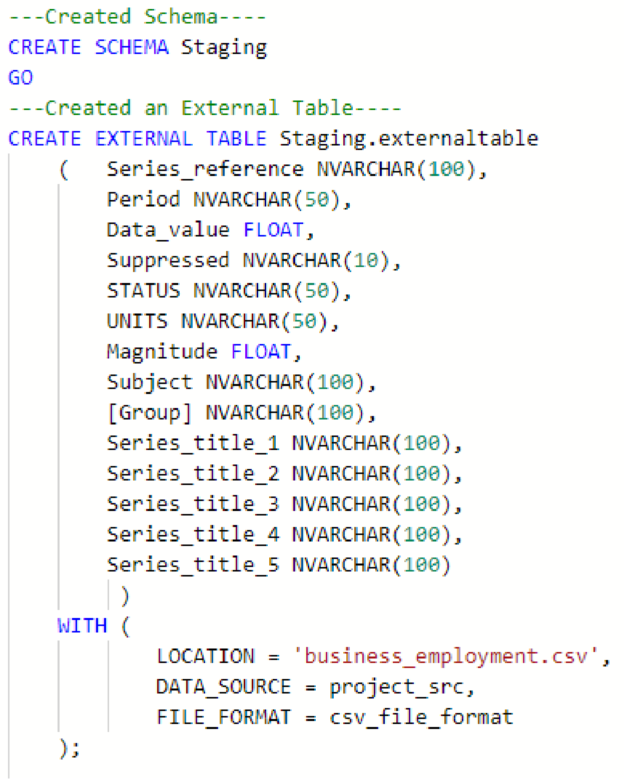
- Set up the Dedicated SQL Pool for scalable data storage.
- Develop SQL queries for data extraction and transformation.
- Implement keys (alternate and surrogate) and support Slowly Changing Dimensions (SCD).

Created an External Data Source - 'project_src' & created a file format for our 'CSV'



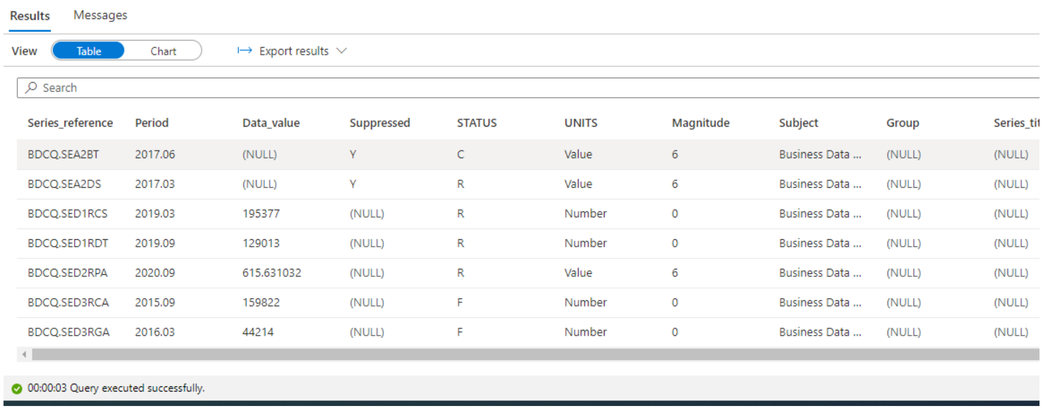
```
--- Created External Data Source---
CREATE EXTERNAL DATA SOURCE project_src
WITH(
LOCATION = 'abfss://project@synapsestorageadls12.dfs.core.windows.net/Data'
)

---Created a CSV File Format----
IF NOT EXISTS (SELECT * FROM sys.external_file_formats WHERE name ='csv_file_format')
CREATE EXTERNAL FILE FORMAT csv_file_format

WITH (

FORMAT_TYPE = DELIMITEDTEXT,
FORMAT_OPTIONS (

FIELD_TERMINATOR = ','

, STRING_DELIMITER = '"'
, First_Row = 2
, USE_TYPE_DEFAULT = FALSE
, Encoding = 'UTF8'
)
```

Created Schema Name ‘Staging’ & created an external table with the column names from the dataset ‘business_employment.csv’ present in ADLS storage



```
---Created Schema----
CREATE SCHEMA Staging
GO
---Created an External Table----
CREATE EXTERNAL TABLE Staging.externaltable
    (   Series_reference NVARCHAR(100),
        Period NVARCHAR(50),
        Data_value FLOAT,
        Suppressed NVARCHAR(10),
        STATUS NVARCHAR(50),
        UNITS NVARCHAR(50),
        Magnitude FLOAT,
        Subject NVARCHAR(100),
        [Group] NVARCHAR(100),
        Series_title_1 NVARCHAR(100),
        Series_title_2 NVARCHAR(100),
        Series_title_3 NVARCHAR(100),
        Series_title_4 NVARCHAR(100),
        Series_title_5 NVARCHAR(100)
        )
    WITH (
            LOCATION = 'business_employment.csv',
            DATA_SOURCE = project_src,
            FILE_FORMAT = csv_file_format
    );
```

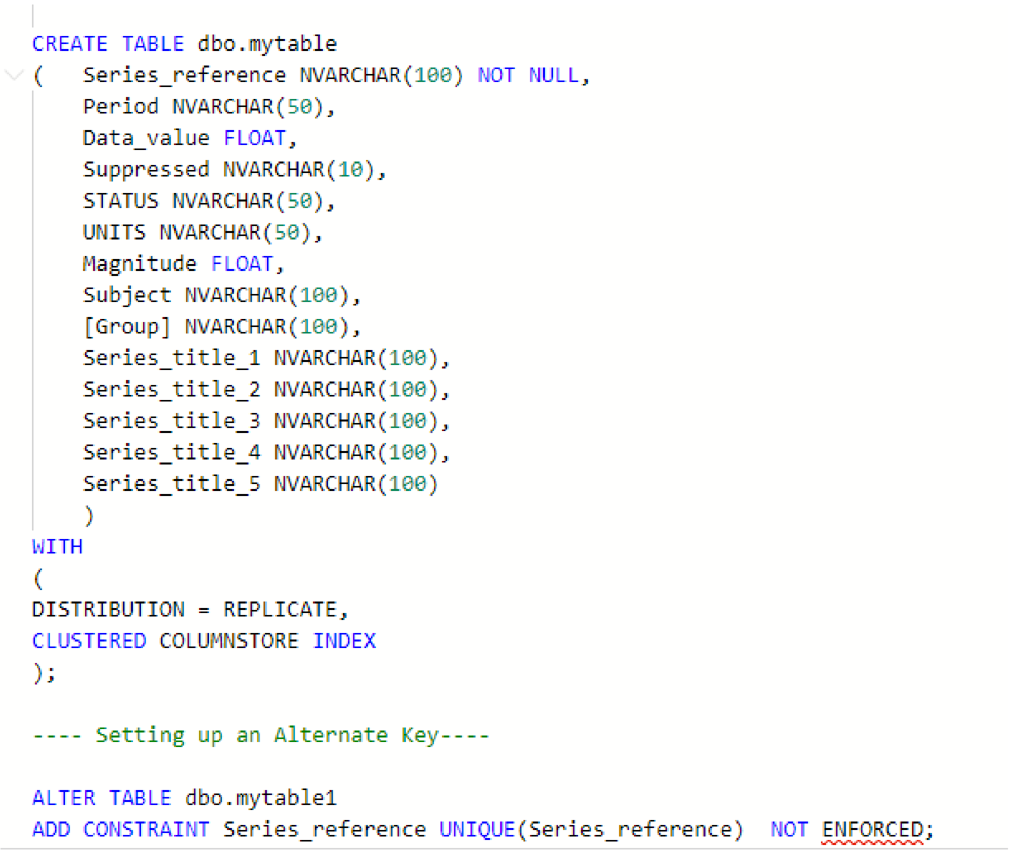
This our external table view after running Select* FROM External Table



Series_reference	Period	Data_value	Suppressed	STATUS	UNITS	Magnitude	Subject	Group	Series_ti
BDCQ.SEA2BT	2017.06	(NULL)	Y	C	Value	6	Business Data ...	(NULL)	(NULL)
BDCQ.SEA2DS	2017.03	(NULL)	Y	R	Value	6	Business Data ...	(NULL)	(NULL)
BDCQ.SED1RCS	2019.03	195377	(NULL)	R	Number	0	Business Data ...	(NULL)	(NULL)
BDCQ.SED1RDT	2019.09	129013	(NULL)	R	Number	0	Business Data ...	(NULL)	(NULL)
BDCQ.SED2RPA	2020.09	615.631032	(NULL)	R	Value	6	Business Data ...	(NULL)	(NULL)
BDCQ.SED3RCA	2015.09	159822	(NULL)	F	Number	0	Business Data ...	(NULL)	(NULL)
BDCQ.SED3RGA	2016.03	44214	(NULL)	F	Number	0	Business Data ...	(NULL)	(NULL)

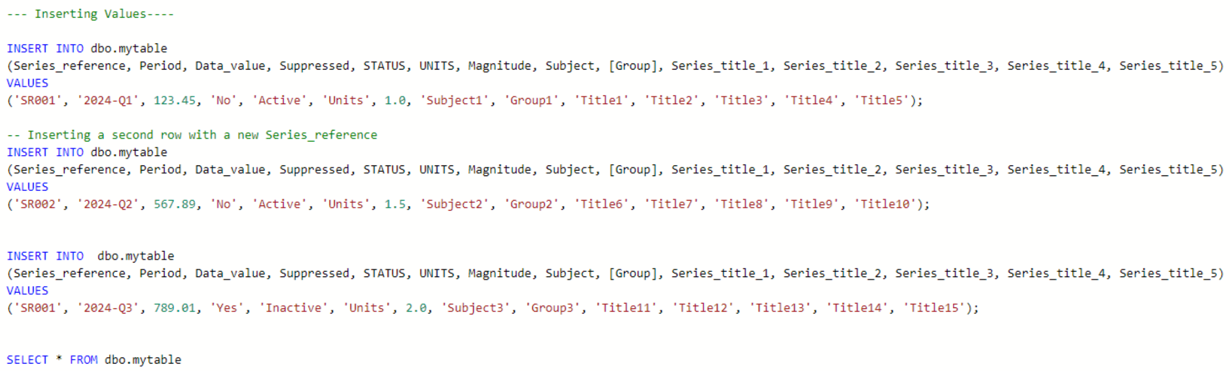
00:00:03 Query executed successfully.

Created a table 'mytable' and added alternate key



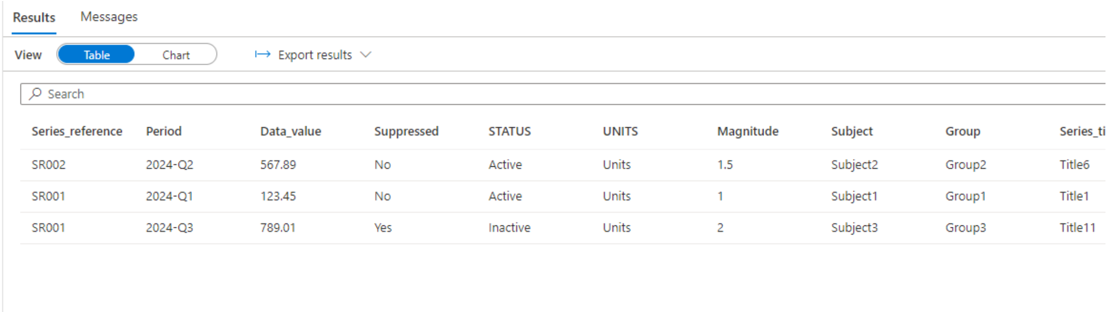
```
CREATE TABLE dbo.mytable
(    Series_reference NVARCHAR(100) NOT NULL,
     Period NVARCHAR(50),
     Data_value FLOAT,
     Suppressed NVARCHAR(10),
     STATUS NVARCHAR(50),
     UNITS NVARCHAR(50),
     Magnitude FLOAT,
     Subject NVARCHAR(100),
     [Group] NVARCHAR(100),
     Series_title_1 NVARCHAR(100),
     Series_title_2 NVARCHAR(100),
     Series_title_3 NVARCHAR(100),
     Series_title_4 NVARCHAR(100),
     Series_title_5 NVARCHAR(100)
     )
WITH
(
DISTRIBUTION = REPLICATE,
CLUSTERED COLUMNSTORE INDEX
);

---- Setting up an Alternate Key----

ALTER TABLE dbo.mytable1
ADD CONSTRAINT Series_reference UNIQUE(Series_reference)  NOT ENFORCED;
```

Given some input values to the above created table, see below:



```sql
--- Inserting Values----

INSERT INTO dbo.mytable
(Series_reference, Period, Data_value, Suppressed, STATUS, UNITS, Magnitude, Subject, [Group], Series_title_1, Series_title_2, Series_title_3, Series_title_4, Series_title_5)
VALUES
('SR001', '2024-Q1', 123.45, 'No', 'Active', 'Units', 1.0, 'Subject1', 'Group1', 'Title1', 'Title2', 'Title3', 'Title4', 'Title5');

-- Inserting a second row with a new Series_reference
INSERT INTO dbo.mytable
(Series_reference, Period, Data_value, Suppressed, STATUS, UNITS, Magnitude, Subject, [Group], Series_title_1, Series_title_2, Series_title_3, Series_title_4, Series_title_5)
VALUES
('SR002', '2024-Q2', 567.89, 'No', 'Active', 'Units', 1.5, 'Subject2', 'Group2', 'Title6', 'Title7', 'Title8', 'Title9', 'Title10');


INSERT INTO  dbo.mytable
(Series_reference, Period, Data_value, Suppressed, STATUS, UNITS, Magnitude, Subject, [Group], Series_title_1, Series_title_2, Series_title_3, Series_title_4, Series_title_5)
VALUES
('SR001', '2024-Q3', 789.01, 'Yes', 'Inactive', 'Units', 2.0, 'Subject3', 'Group3', 'Title11', 'Title12', 'Title13', 'Title14', 'Title15');


SELECT * FROM dbo.mytable
```

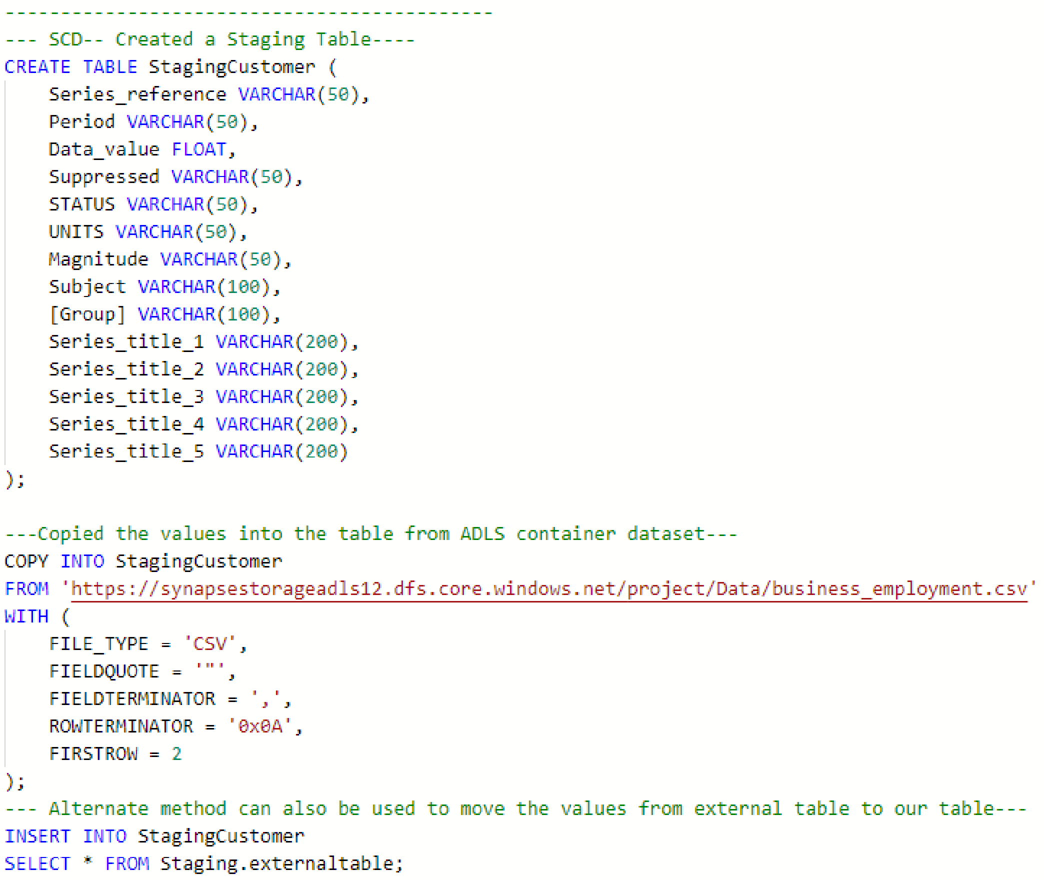
Below output, alternate key can be seen…



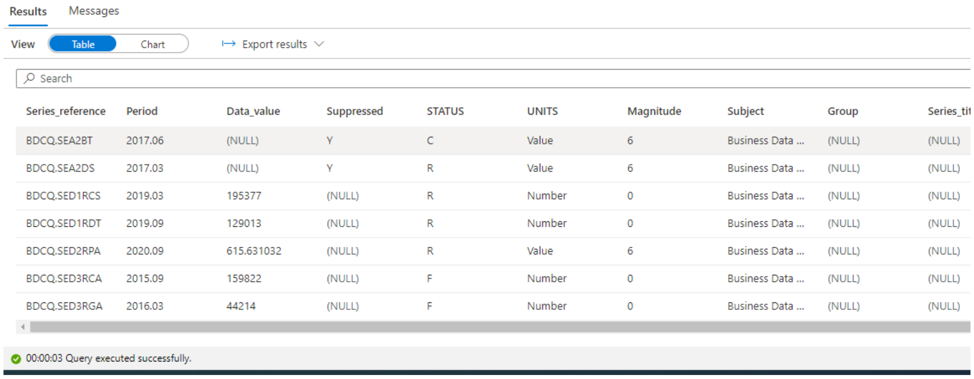
Series_reference	Period	Data_value	Suppressed	STATUS	UNITS	Magnitude	Subject	Group	Series_ti
SR002	2024-Q2	567.89	No	Active	Units	1.5	Subject2	Group2	Title6
SR001	2024-Q1	123.45	No	Active	Units	1	Subject1	Group1	Title1
SR001	2024-Q3	789.01	Yes	Inactive	Units	2	Subject3	Group3	Title11

Now created another table to demonstrate Slowly Changing Dimension Type 2:

Created a staging table as below and copied the data the from ADLS container dataset to the table.



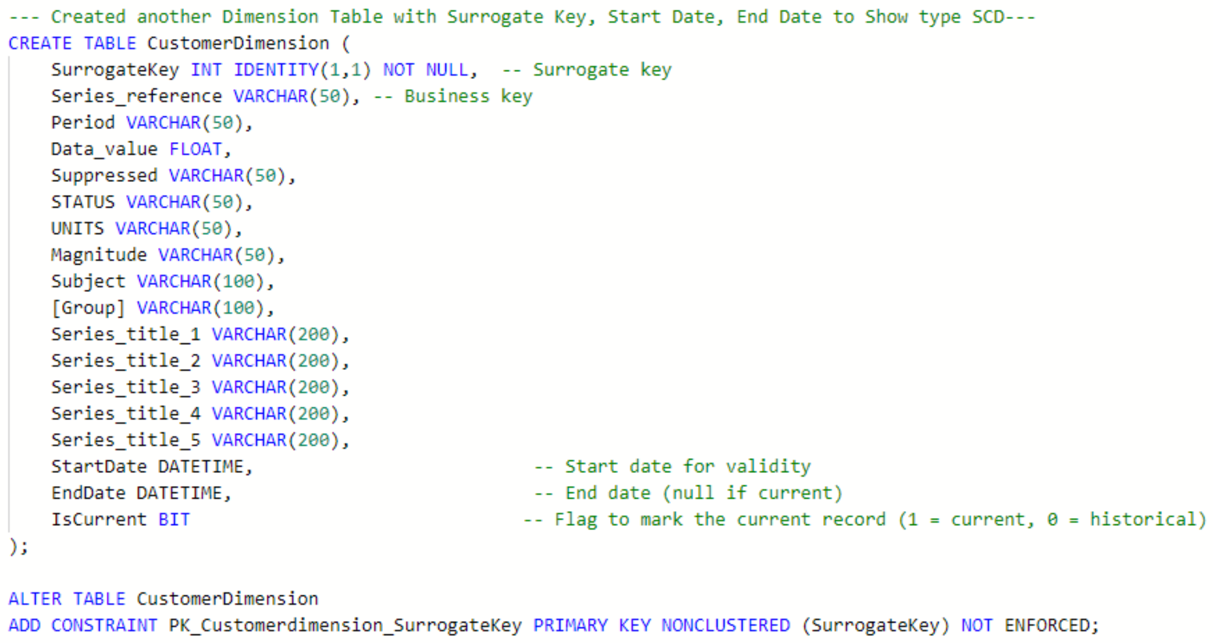
```sql
---------------------------------------------
--- SCD-- Created a Staging Table----
CREATE TABLE StagingCustomer (
    Series_reference VARCHAR(50),
    Period VARCHAR(50),
    Data_value FLOAT,
    Suppressed VARCHAR(50),
    STATUS VARCHAR(50),
    UNITS VARCHAR(50),
    Magnitude VARCHAR(50),
    Subject VARCHAR(100),
    [Group] VARCHAR(100),
    Series_title_1 VARCHAR(200),
    Series_title_2 VARCHAR(200),
    Series_title_3 VARCHAR(200),
    Series_title_4 VARCHAR(200),
    Series_title_5 VARCHAR(200)
);

---Copied the values into the table from ADLS container dataset---
COPY INTO StagingCustomer
FROM 'https://synapsestorageadls12.dfs.core.windows.net/project/Data/business_employment.csv'
WITH (
    FILE_TYPE = 'CSV',
    FIELDQUOTE = '"',
    FIELDTERMINATOR = ',',
    ROWTERMINATOR = '0x0A',
    FIRSTROW = 2
);
--- Alternate method can also be used to move the values from external table to our table---
INSERT INTO StagingCustomer
SELECT * FROM Staging.externaltable;
```



Series_reference	Period	Data_value	Suppressed	STATUS	UNITS	Magnitude	Subject	Group	Series_ti
BDCQ.SEA2BT	2017.06	(NULL)	Y	C	Value	6	Business Data ...	(NULL)	(NULL)
BDCQ.SEA2DS	2017.03	(NULL)	Y	R	Value	6	Business Data ...	(NULL)	(NULL)
BDCQ.SED1RCS	2019.03	195377	(NULL)	R	Number	0	Business Data ...	(NULL)	(NULL)
BDCQ.SED1RDT	2019.09	129013	(NULL)	R	Number	0	Business Data ...	(NULL)	(NULL)
BDCQ.SED2RPA	2020.09	615.631032	(NULL)	R	Value	6	Business Data ...	(NULL)	(NULL)
BDCQ.SED3RCA	2015.09	159822	(NULL)	F	Number	0	Business Data ...	(NULL)	(NULL)
BDCQ.SED3RGA	2016.03	44214	(NULL)	F	Number	0	Business Data ...	(NULL)	(NULL)

00:00:03 Query executed successfully.

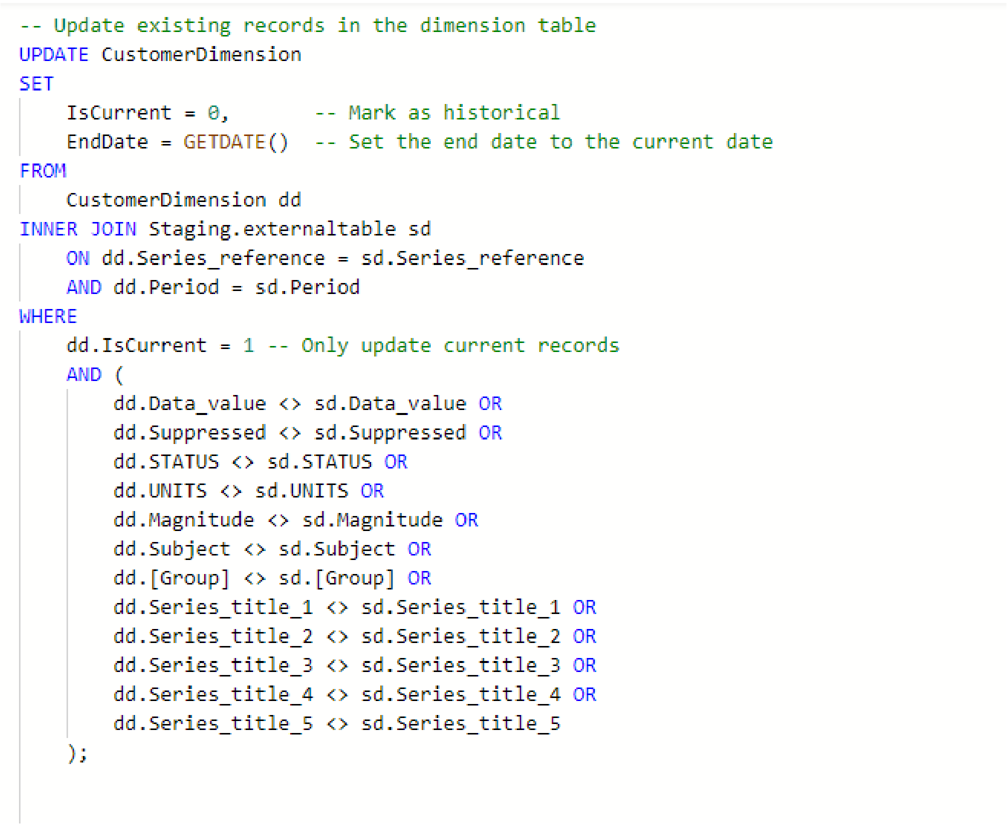
Now we are creating a Dimension table as below:



```
--- Created another Dimension Table with Surrogate Key, Start Date, End Date to Show type SCD---
CREATE TABLE CustomerDimension (
    SurrogateKey INT IDENTITY(1,1) NOT NULL,  -- Surrogate key
    Series_reference VARCHAR(50), -- Business key
    Period VARCHAR(50),
    Data_value FLOAT,
    Suppressed VARCHAR(50),
    STATUS VARCHAR(50),
    UNITS VARCHAR(50),
    Magnitude VARCHAR(50),
    Subject VARCHAR(100),
    [Group] VARCHAR(100),
    Series_title_1 VARCHAR(200),
    Series_title_2 VARCHAR(200),
    Series_title_3 VARCHAR(200),
    Series_title_4 VARCHAR(200),
    Series_title_5 VARCHAR(200),
    StartDate DATETIME,                          -- Start date for validity
    EndDate DATETIME,                            -- End date (null if current)
    IsCurrent BIT                               -- Flag to mark the current record (1 = current, 0 = historical)
);

ALTER TABLE CustomerDimension
ADD CONSTRAINT PK_Customerdimension_SurrogateKey PRIMARY KEY NONCLUSTERED (SurrogateKey) NOT ENFORCED;
```

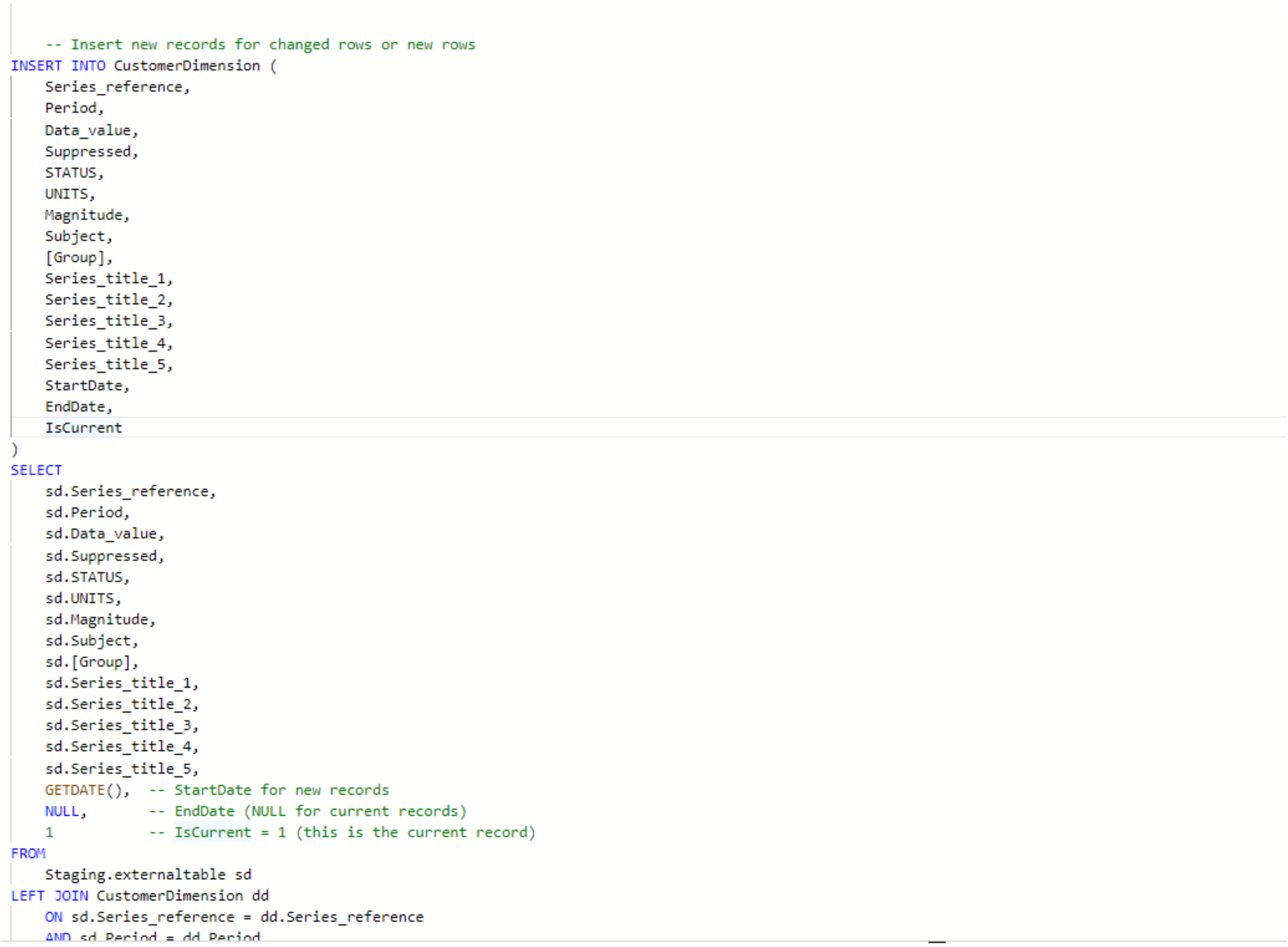
In order to perform Type 2 SCD,
We will compare the data in the staging table with the dimension table,

update existing records in the dimension table as historical by updating the IsCurrent flag and setting the EndDate.

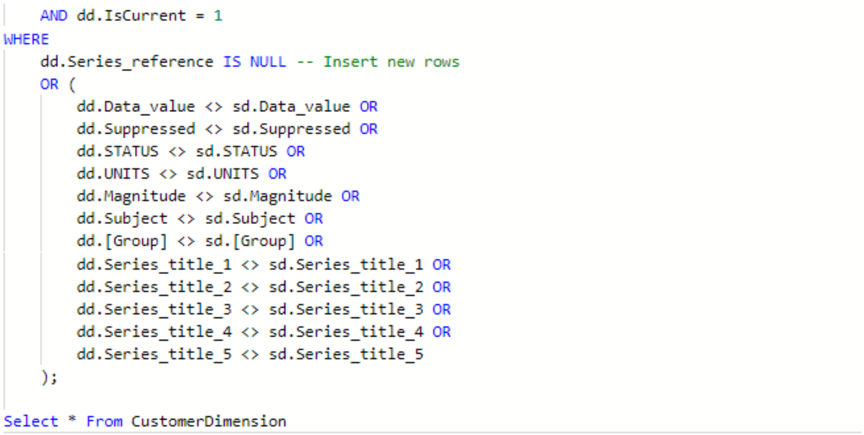


```
-- Update existing records in the dimension table
UPDATE CustomerDimension
SET
    IsCurrent = 0,       -- Mark as historical
    EndDate = GETDATE()  -- Set the end date to the current date
FROM
    CustomerDimension dd
INNER JOIN Staging.externaltable sd
    ON dd.Series_reference = sd.Series_reference
    AND dd.Period = sd.Period
WHERE
    dd.IsCurrent = 1 -- Only update current records
    AND (
        dd.Data_value <> sd.Data_value OR
        dd.Suppressed <> sd.Suppressed OR
        dd.STATUS <> sd.STATUS OR
        dd.UNITS <> sd.UNITS OR
        dd.Magnitude <> sd.Magnitude OR
        dd.Subject <> sd.Subject OR
        dd.[Group] <> sd.[Group] OR
        dd.Series_title_1 <> sd.Series_title_1 OR
        dd.Series_title_2 <> sd.Series_title_2 OR
        dd.Series_title_3 <> sd.Series_title_3 OR
        dd.Series_title_4 <> sd.Series_title_4 OR
        dd.Series_title_5 <> sd.Series_title_5
    );
```

Once the existing records are marked as historical, we will insert the updated records from the staging table into the dimension table as new current records (IsCurrent = 1).

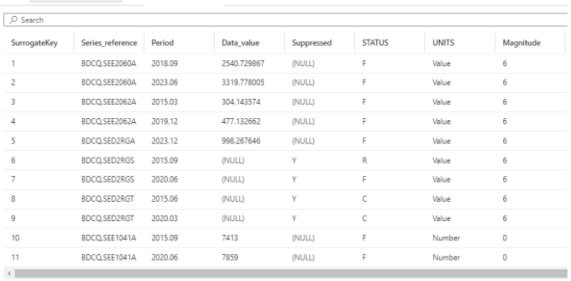


```
    -- Insert new records for changed rows or new rows
INSERT INTO CustomerDimension (
    Series_reference,
    Period,
    Data_value,
    Suppressed,
    STATUS,
    UNITS,
    Magnitude,
    Subject,
    [Group],
    Series_title_1,
    Series_title_2,
    Series_title_3,
    Series_title_4,
    Series_title_5,
    StartDate,
    EndDate,
    IsCurrent
)
SELECT
    sd.Series_reference,
    sd.Period,
    sd.Data_value,
    sd.Suppressed,
    sd.STATUS,
    sd.UNITS,
    sd.Magnitude,
    sd.Subject,
    sd.[Group],
    sd.Series_title_1,
    sd.Series_title_2,
    sd.Series_title_3,
    sd.Series_title_4,
    sd.Series_title_5,
    GETDATE(),  -- StartDate for new records
    NULL,       -- EndDate (NULL for current records)
    1           -- IsCurrent = 1 (this is the current record)
FROM
    Staging.externaltable sd
LEFT JOIN CustomerDimension dd
    ON sd.Series_reference = dd.Series_reference
    AND sd.Period = dd.Period
```



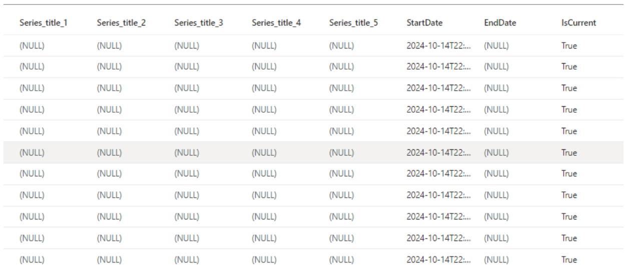
```
    AND dd.IsCurrent = 1
WHERE
    dd.Series_reference IS NULL -- Insert new rows
    OR (
        dd.Data_value <> sd.Data_value OR
        dd.Suppressed <> sd.Suppressed OR
        dd.STATUS <> sd.STATUS OR
        dd.UNITS <> sd.UNITS OR
        dd.Magnitude <> sd.Magnitude OR
        dd.Subject <> sd.Subject OR
        dd.[Group] <> sd.[Group] OR
        dd.Series_title_1 <> sd.Series_title_1 OR
        dd.Series_title_2 <> sd.Series_title_2 OR
        dd.Series_title_3 <> sd.Series_title_3 OR
        dd.Series_title_4 <> sd.Series_title_4 OR
        dd.Series_title_5 <> sd.Series_title_5
    );

Select * From CustomerDimension
```

We get the below output with Start Date, End Date and Surrogate Key:



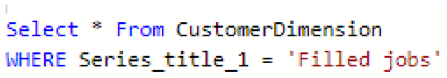
Search

SurrogateKey	Series_reference	Period	Data_value	Suppressed	STATUS	UNITS	Magnitude
1	BDCQ.SEE2060A	2018.09	2540.729867	(NULL)	F	Value	6
2	BDCQ.SEE2060A	2023.06	3319.778005	(NULL)	F	Value	6
3	BDCQ.SEE2062A	2015.03	304.143574	(NULL)	F	Value	6
4	BDCQ.SEE2062A	2019.12	477.132662	(NULL)	F	Value	6
5	BDCQ.SED2RGA	2023.12	998.267646	(NULL)	F	Value	6
6	BDCQ.SED2RGS	2015.09	(NULL)	Y	R	Value	6
7	BDCQ.SED2RGS	2020.06	(NULL)	Y	F	Value	6
8	BDCQ.SED2RGT	2015.06	(NULL)	Y	C	Value	6
9	BDCQ.SED2RGT	2020.03	(NULL)	Y	C	Value	6
10	BDCQ.SEE1041A	2015.09	7413	(NULL)	F	Number	0
11	BDCQ.SEE1041A	2020.06	7859	(NULL)	F	Number	0

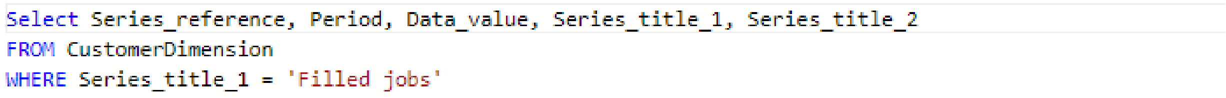


Series_title_1	Series_title_2	Series_title_3	Series_title_4	Series_title_5	StartDate	EndDate	IsCurrent
(NULL)	(NULL)	(NULL)	(NULL)	(NULL)	2024-10-14T22:...	(NULL)	True
(NULL)	(NULL)	(NULL)	(NULL)	(NULL)	2024-10-14T22:...	(NULL)	True
(NULL)	(NULL)	(NULL)	(NULL)	(NULL)	2024-10-14T22:...	(NULL)	True
(NULL)	(NULL)	(NULL)	(NULL)	(NULL)	2024-10-14T22:...	(NULL)	True
(NULL)	(NULL)	(NULL)	(NULL)	(NULL)	2024-10-14T22:...	(NULL)	True
(NULL)	(NULL)	(NULL)	(NULL)	(NULL)	2024-10-14T22:...	(NULL)	True
(NULL)	(NULL)	(NULL)	(NULL)	(NULL)	2024-10-14T22:...	(NULL)	True
(NULL)	(NULL)	(NULL)	(NULL)	(NULL)	2024-10-14T22:...	(NULL)	True
(NULL)	(NULL)	(NULL)	(NULL)	(NULL)	2024-10-14T22:...	(NULL)	True
(NULL)	(NULL)	(NULL)	(NULL)	(NULL)	2024-10-14T22:...	(NULL)	True
(NULL)	(NULL)	(NULL)	(NULL)	(NULL)	2024-10-14T22:...	(NULL)	True

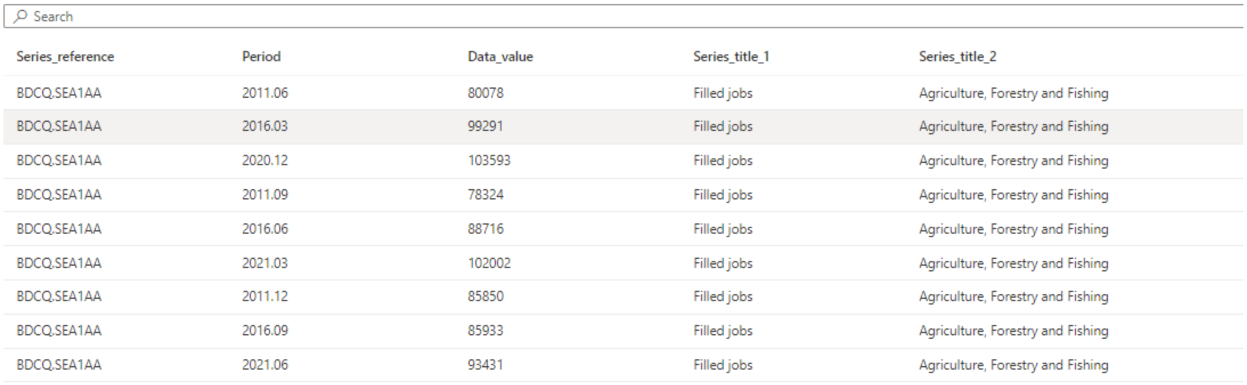
Did some transformation using where clause and selected few columns only



```
Select * From CustomerDimension
WHERE Series_title_1 = 'Filled jobs'
```



```
Select Series_reference, Period, Data_value, Series_title_1, Series_title_2
FROM CustomerDimension
WHERE Series_title_1 = 'Filled jobs'
```



Search

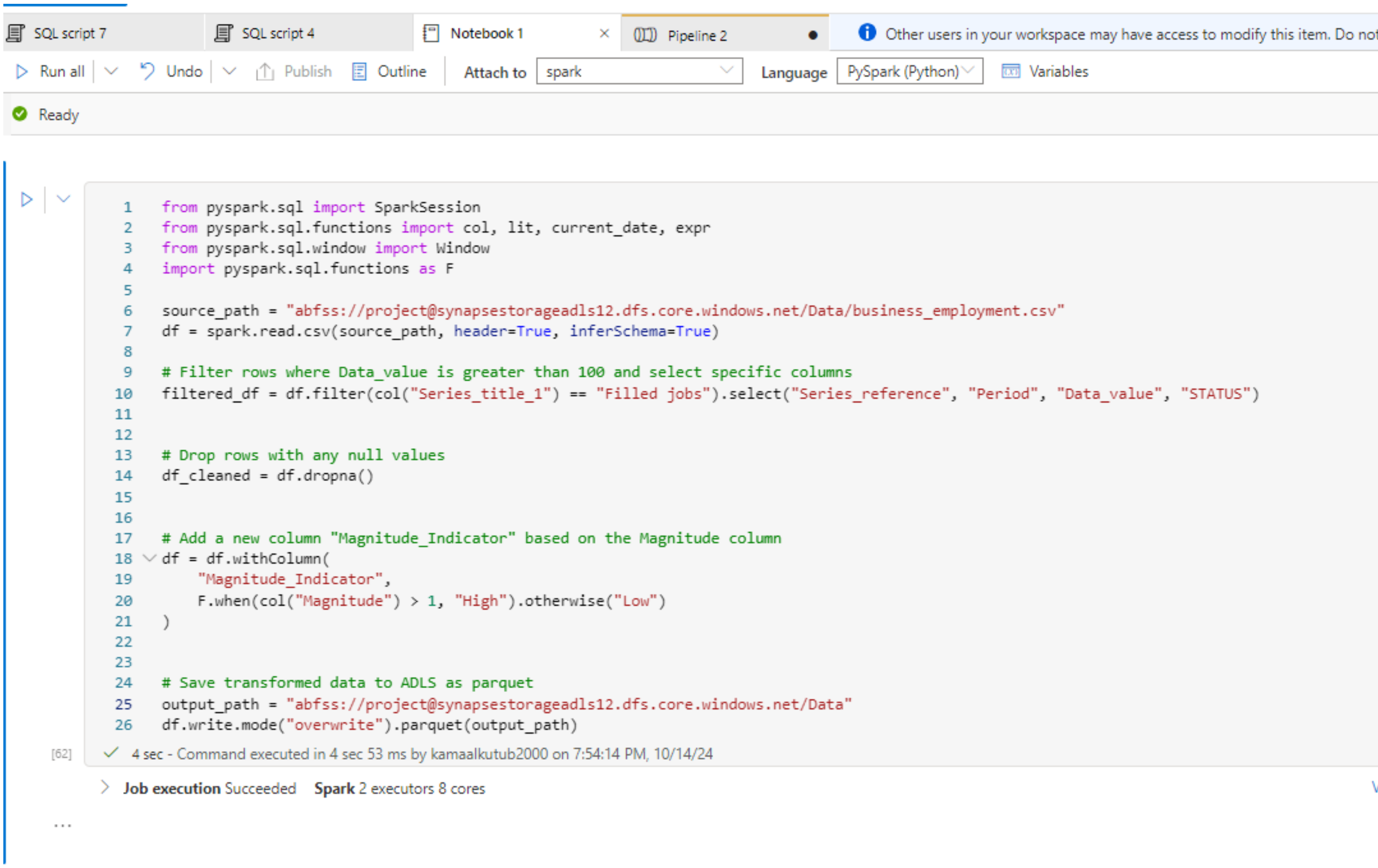
Series_reference	Period	Data_value	Series_title_1	Series_title_2
BDCQ.SEA1AA	2011.06	80078	Filled jobs	Agriculture, Forestry and Fishing
BDCQ.SEA1AA	2016.03	99291	Filled jobs	Agriculture, Forestry and Fishing
BDCQ.SEA1AA	2020.12	103593	Filled jobs	Agriculture, Forestry and Fishing
BDCQ.SEA1AA	2011.09	78324	Filled jobs	Agriculture, Forestry and Fishing
BDCQ.SEA1AA	2016.06	88716	Filled jobs	Agriculture, Forestry and Fishing
BDCQ.SEA1AA	2021.03	102002	Filled jobs	Agriculture, Forestry and Fishing
BDCQ.SEA1AA	2011.12	85850	Filled jobs	Agriculture, Forestry and Fishing
BDCQ.SEA1AA	2016.09	85933	Filled jobs	Agriculture, Forestry and Fishing
BDCQ.SEA1AA	2021.06	93431	Filled jobs	Agriculture, Forestry and Fishing

Apache Spark Pool: Enable large-scale data processing and transformations using Spark.

- Create and configure the Spark Pool.
- Write Python scripts for data transformation.
- Use Spark to integrate and process data in near real-time or batch mode.

Created spark pool and launched the Notebook,

Gave few queries and transformed the data and imported it to a pipeline



Other users in your workspace may have access to modify this item. Do not

Attach to spark

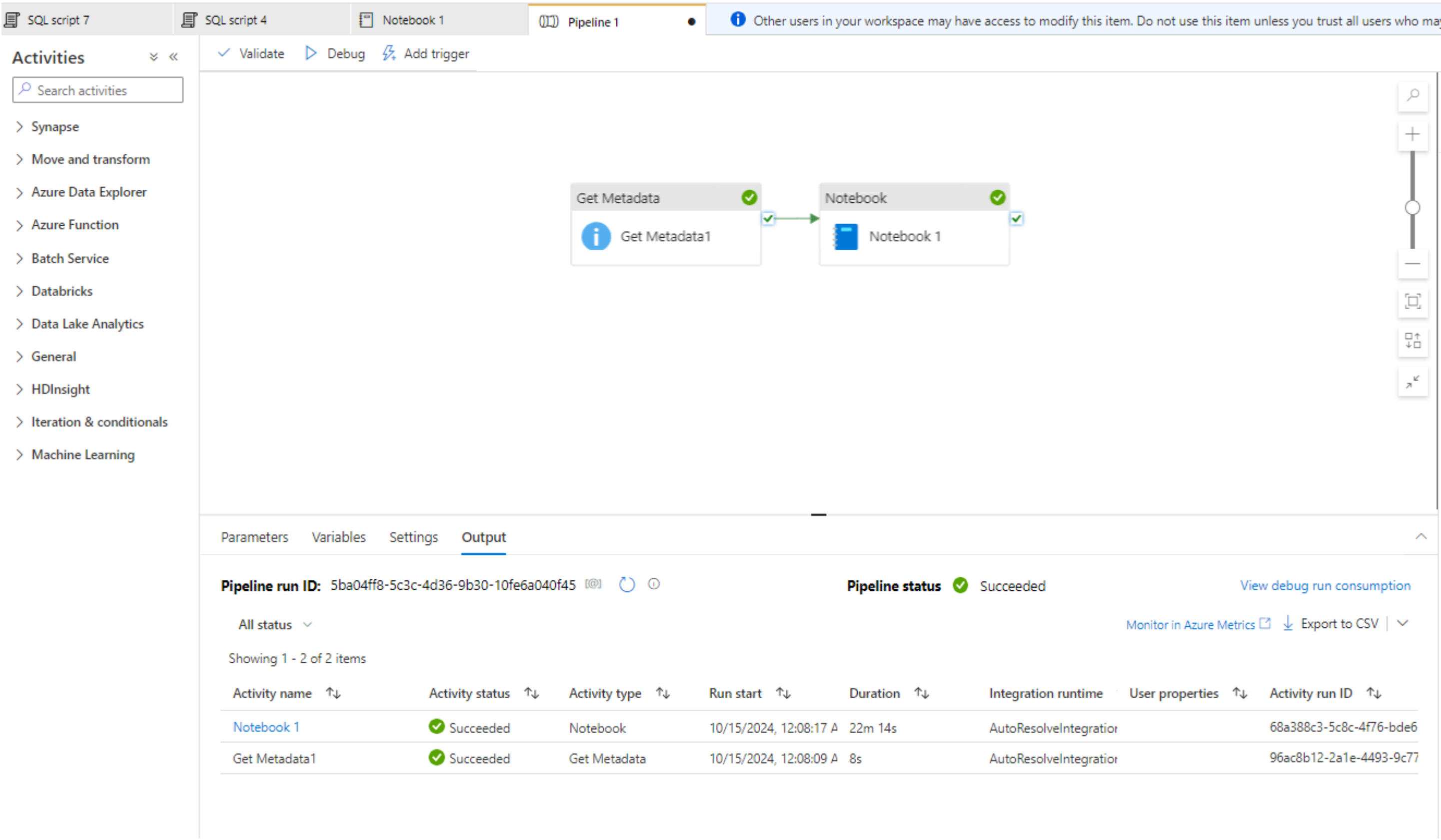
Language PySpark (Python)

```python
from pyspark.sql import SparkSession
from pyspark.sql.functions import col, lit, current_date, expr
from pyspark.sql.window import Window
import pyspark.sql.functions as F

source_path = "abfss://project@synapsestorageadls12.dfs.core.windows.net/Data/business_employment.csv"
df = spark.read.csv(source_path, header=True, inferSchema=True)

# Filter rows where Data_value is greater than 100 and select specific columns
filtered_df = df.filter(col("Series_title_1") == "Filled jobs").select("Series_reference", "Period", "Data_value", "STATUS")


# Drop rows with any null values
df_cleaned = df.dropna()


# Add a new column "Magnitude_Indicator" based on the Magnitude column
df = df.withColumn(
    "Magnitude_Indicator",
    F.when(col("Magnitude") > 1, "High").otherwise("Low")
)


# Save transformed data to ADLS as parquet
output_path = "abfss://project@synapsestorageadls12.dfs.core.windows.net/Data"
df.write.mode("overwrite").parquet(output_path)
```

[62] ✓ 4 sec - Command executed in 4 sec 53 ms by kamaalkutub2000 on 7:54:14 PM, 10/14/24

Job execution Succeeded Spark 2 executors 8 cores

Imported the Notebook and ran the pipeline along with data exists in Get Metadata.



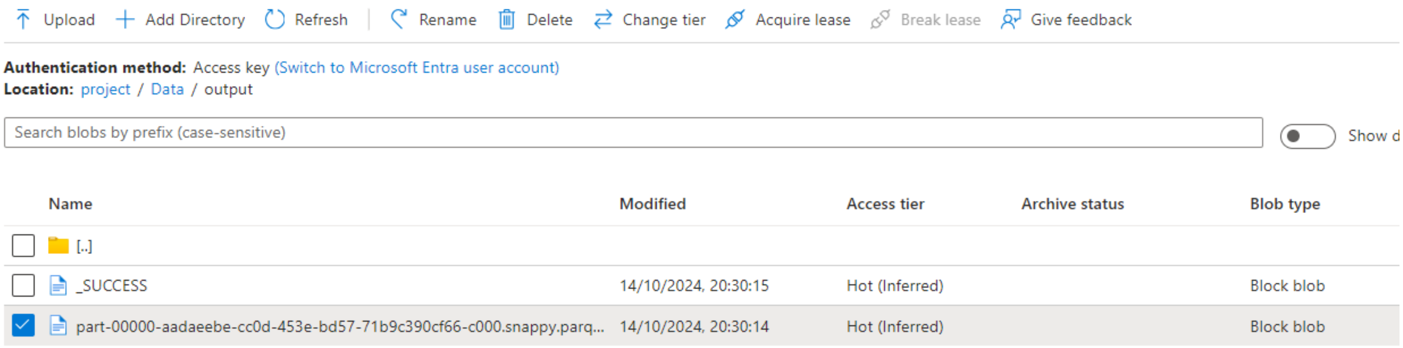
Pipeline run ID: 5ba04ff8-5c3c-4d36-9b30-10fe6a040f45

Pipeline status ✓ Succeeded

Showing 1 - 2 of 2 items

Activity name	Activity status	Activity type	Run start	Duration	Integration runtime	User properties	Activity run ID
Notebook 1	✓ Succeeded	Notebook	10/15/2024, 12:08:17 A	22m 14s	AutoResolveIntegratior		68a388c3-5c8c-4f76-bde6
Get Metadata1	✓ Succeeded	Get Metadata	10/15/2024, 12:08:09 A	8s	AutoResolveIntegratior		96ac8b12-2a1e-4493-9c77

The Output of the pipeline can be seen in the ADLS Container as below:



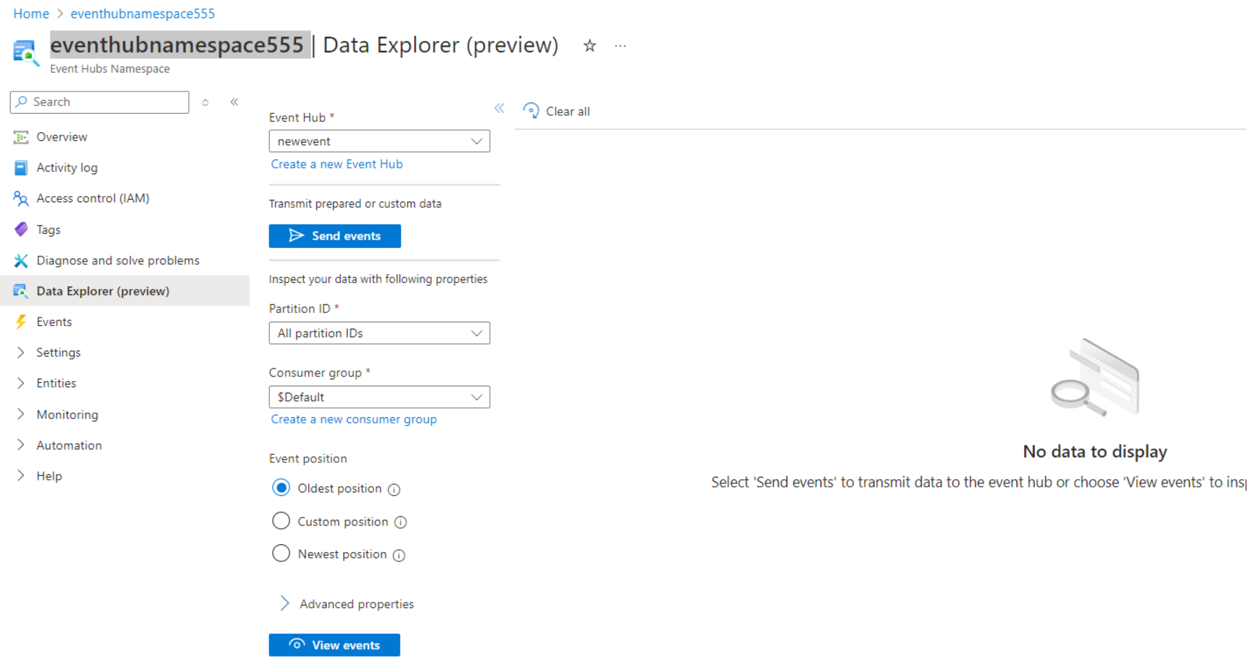
Authentication method: Access key (Switch to Microsoft Entra user account)
Location: project / Data / output

	Name	Modified	Access tier	Archive status	Blob type
□	[..]				
□	_SUCCESS	14/10/2024, 20:30:15	Hot (Inferred)		Block blob
☑	part-00000-aadaeebe-cc0d-453e-bd57-71b9c390cf66-c000.snappy.parq...	14/10/2024, 20:30:14	Hot (Inferred)		Block blob

<u>Azure Event Hubs: </u> To Ingest real-time data from various sources such as IoT devices and applications.

- Set up Event Hub for real-time data ingestion.
- Integrate Event Hub with Stream Analytics for real-time processing.

Created a new event hub workspace and created a new event:



eventhubnamespace555 | Data Explorer (preview)

Event Hubs Namespace

Event Hub *: newevent

Transmit prepared or custom data

Inspect your data with following properties

Partition ID *: All partition IDs

Consumer group *: $Default

Event position

● Oldest position

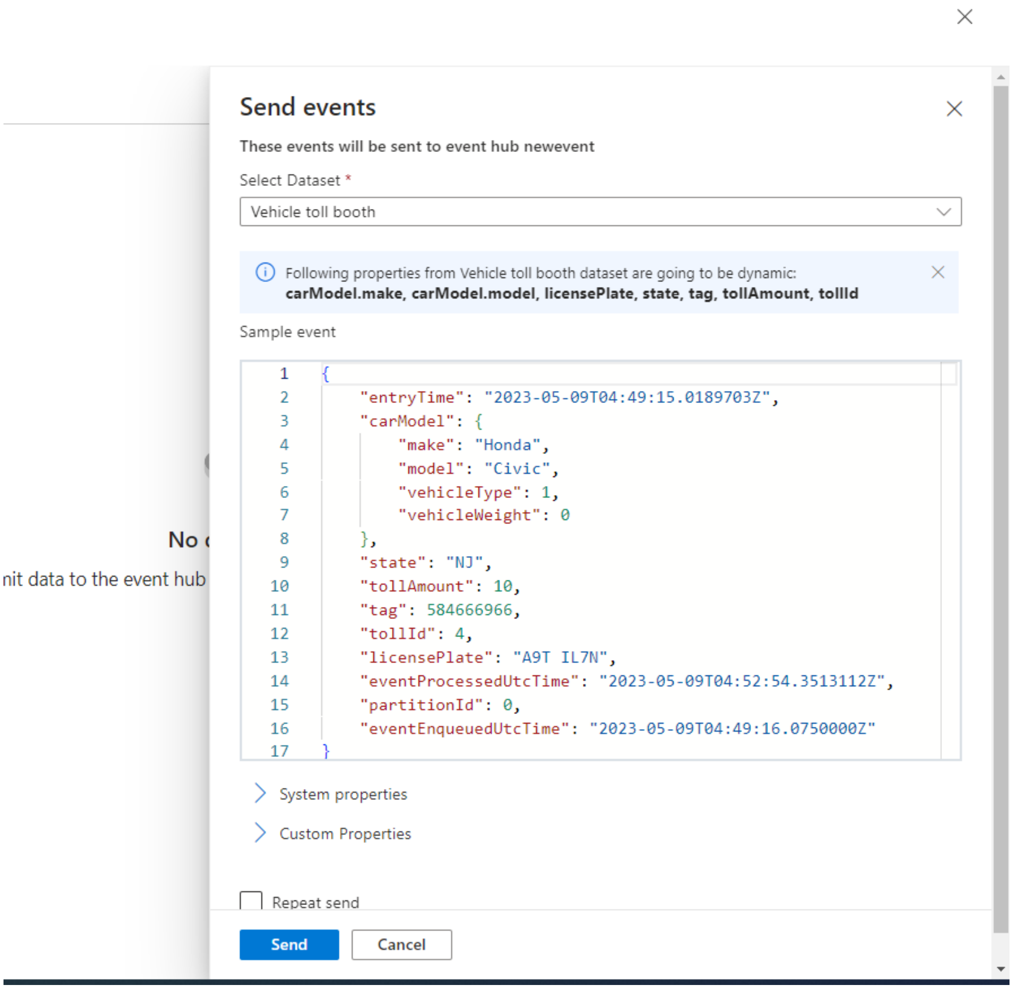
○ Custom position

○ Newest position

No data to display

Select 'Send events' to transmit data to the event hub or choose 'View events' to ins

Used the sample event to send the data from vehicle toll booth:



Send events

These events will be sent to event hub newevent

Select Dataset *

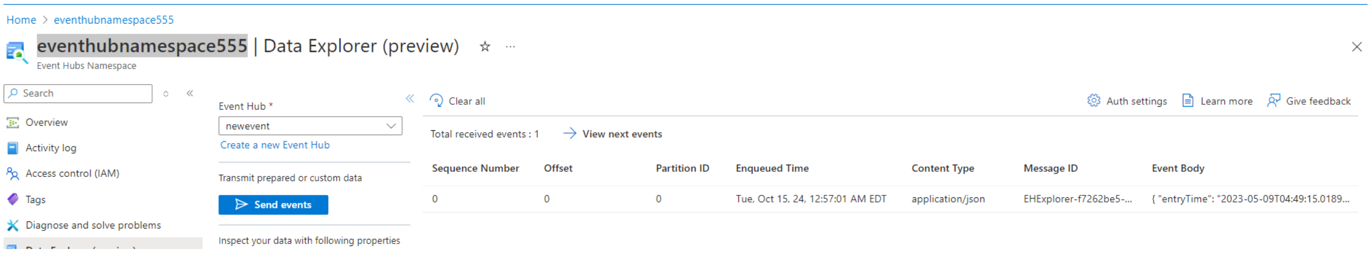
Vehicle toll booth

Following properties from Vehicle toll booth dataset are going to be dynamic: **carModel.make, carModel.model, licensePlate, state, tag, tollAmount, tollId**

Sample event

```
{
    "entryTime": "2023-05-09T04:49:15.0189703Z",
    "carModel": {
        "make": "Honda",
        "model": "Civic",
        "vehicleType": 1,
        "vehicleWeight": 0
    },
    "state": "NJ",
    "tollAmount": 10,
    "tag": 584666966,
    "tollId": 4,
    "licensePlate": "A9T IL7N",
    "eventProcessedUtcTime": "2023-05-09T04:52:54.3513112Z",
    "partitionId": 0,
    "eventEnqueuedUtcTime": "2023-05-09T04:49:16.0750000Z"
}
```

□ Repeat send



eventhubnamespace555 | Data Explorer (preview)

Event Hubs Namespace

Event Hub *

newevent

Transmit prepared or custom data

Inspect your data with following properties

Total received events : 1

Sequence Number	Offset	Partition ID	Enqueued Time	Content Type	Message ID	Event Body
0	0	0	Tue, Oct 15, 24, 12:57:01 AM EDT	application/json	EHExplorer-f7262be5-...	{ "entryTime": "2023-05-09T04:49:15.0189...

<u>Azure Stream Analytics:</u> To Perform real-time stream processing and transformation of data from Event Hub and route it to Dedicated SQL Pool.

- Set up Stream Analytics jobs for data transformation.
- Define queries to process data from Event Hub and route output to Cosmos DB or Dedicated SQL Pool.

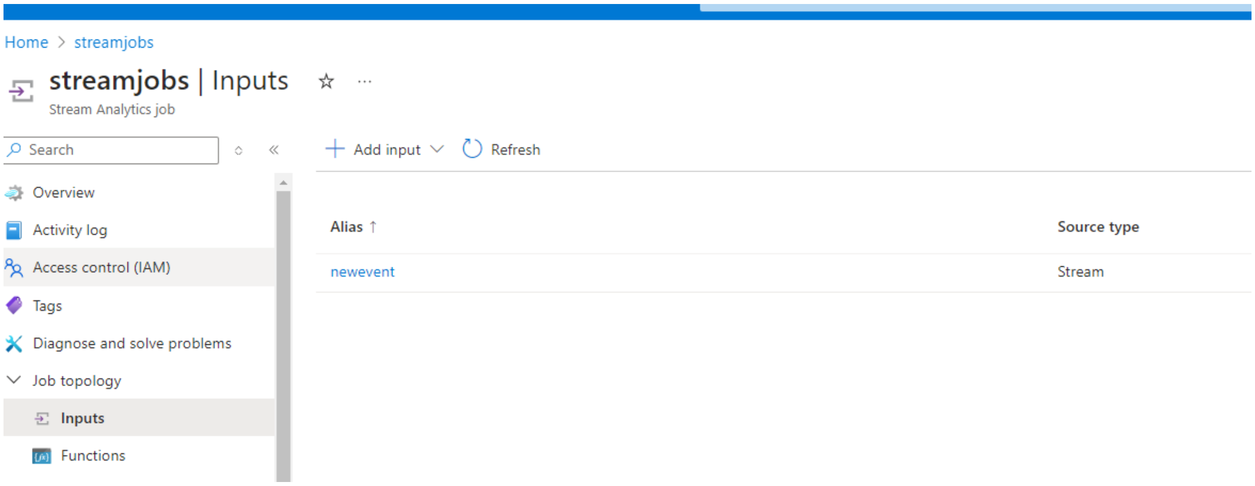
Created a stream job for the stream analytics real time data transformation

Created the below for stream jobs:

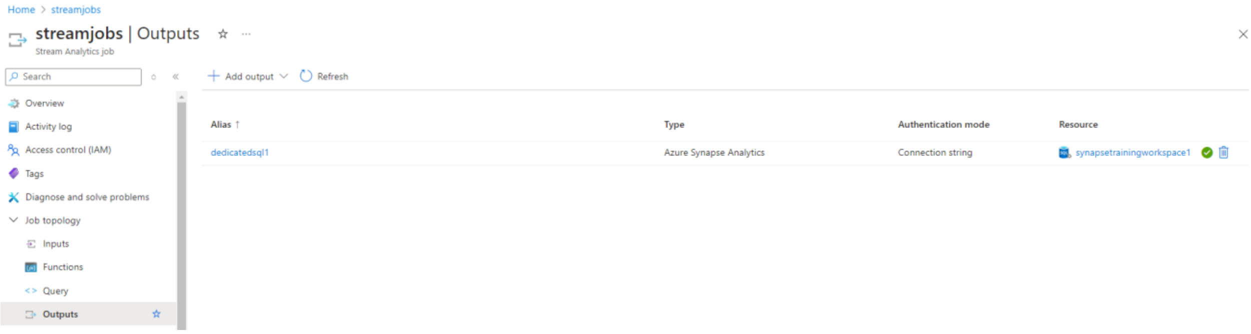
Input – to fetch from our Event Hubs created event

Output – To our dedicated sql Pool

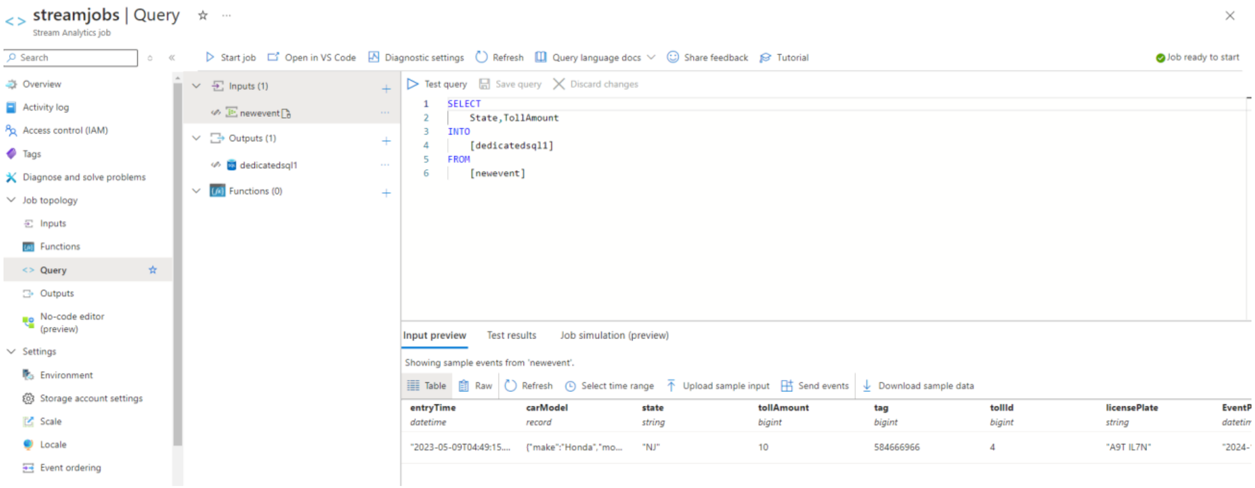
Query – Gave the query to run the stream jobs for gathering the data only for two columns.



Alias ↑	Source type
newevent	Stream



Alias ↑	Type	Authentication mode	Resource
dedicatedsql1	Azure Synapse Analytics	Connection string	synapsetrainingworkspace1

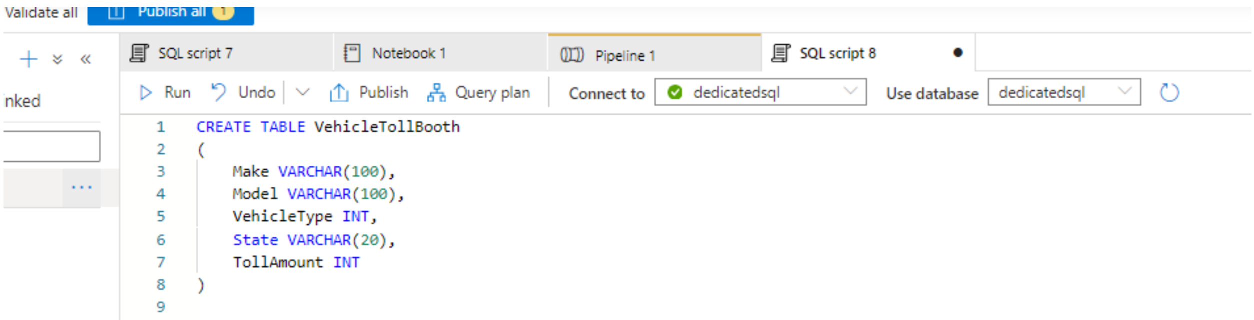


```
SELECT
    State,TollAmount
INTO
    [dedicatedsql1]
FROM
    [newevent]
```

Showing sample events from 'newevent'.

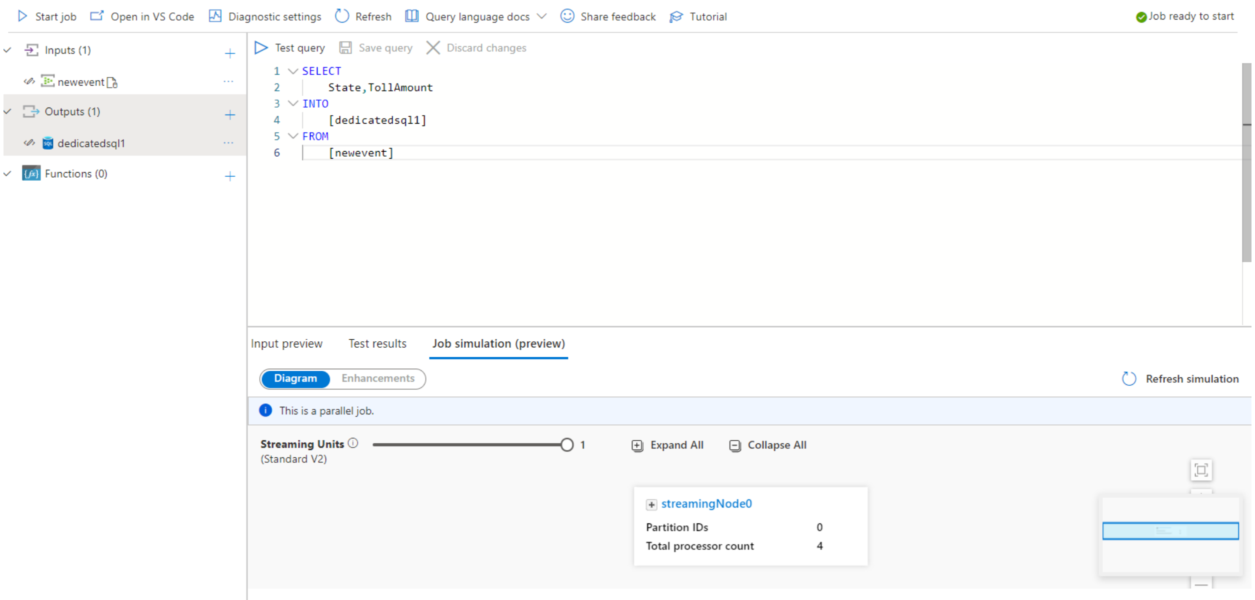
entryTime datetime	carModel record	state string	tollAmount bigint	tag bigint	tollId bigint	licensePlate string	EventP datetim
"2023-05-09T04:49:15....	{"make":"Honda","mo...	"NJ"	10	584666966	4	"A9T IL7N"	"2024-

Also created a table 'vehicletollbooth' in dedicated pool:



```
CREATE TABLE VehicleTollBooth
(
    Make VARCHAR(100),
    Model VARCHAR(100),
    VehicleType INT,
    State VARCHAR(20),
    TollAmount INT
)
```

And final ran the below query, saved the query, hit the start job.



```
SELECT
    State,TollAmount
INTO
    [dedicatedsql1]
FROM
    [newevent]
```

This is a parallel job.

Streaming Units (Standard V2): 1

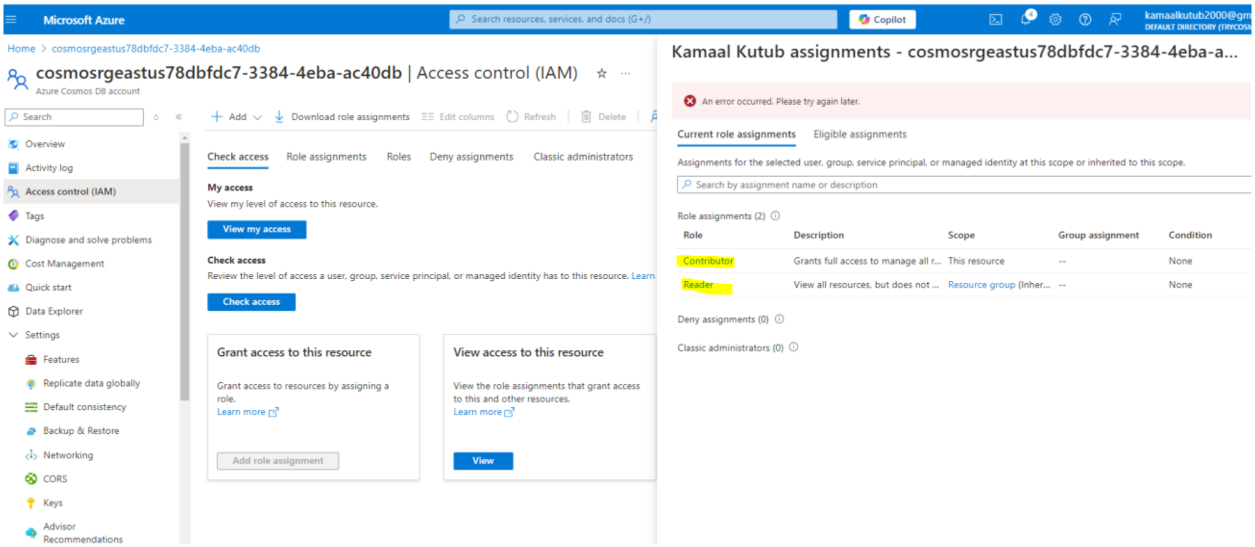
streamingNode0

Partition IDs	0
Total processor count	4

<u>Azure Cosmos DB:</u> Manage and store JSON documents in a NoSQL format for near real-time analytics.

- Set up and configure Cosmos DB.
- Design collections with proper indexing strategies (Cluster Indexing, Column Indexing).
- Integrate access control mechanisms.
- Store processed data from Stream Analytics or Spark Pool.

Created a Cosmos DB Workspace, checked for Role Based Access Control and did not change the access since I was having Contributor, Reader access.



cosmosrgeastus78dbfdc7-3384-4eba-ac40db | Access control (IAM)

Azure Cosmos DB account

My access

View my level of access to this resource.

Check access

Review the level of access a user, group, service principal, or managed identity has to this resource. Learn

Grant access to this resource

Grant access to resources by assigning a role.

View access to this resource

View the role assignments that grant access to this and other resources.

Kamaal Kutub assignments - cosmosrgeastus78dbfdc7-3384-4eba-a...

An error occurred. Please try again later.

Assignments for the selected user, group, service principal, or managed identity at this scope or inherited to this scope.

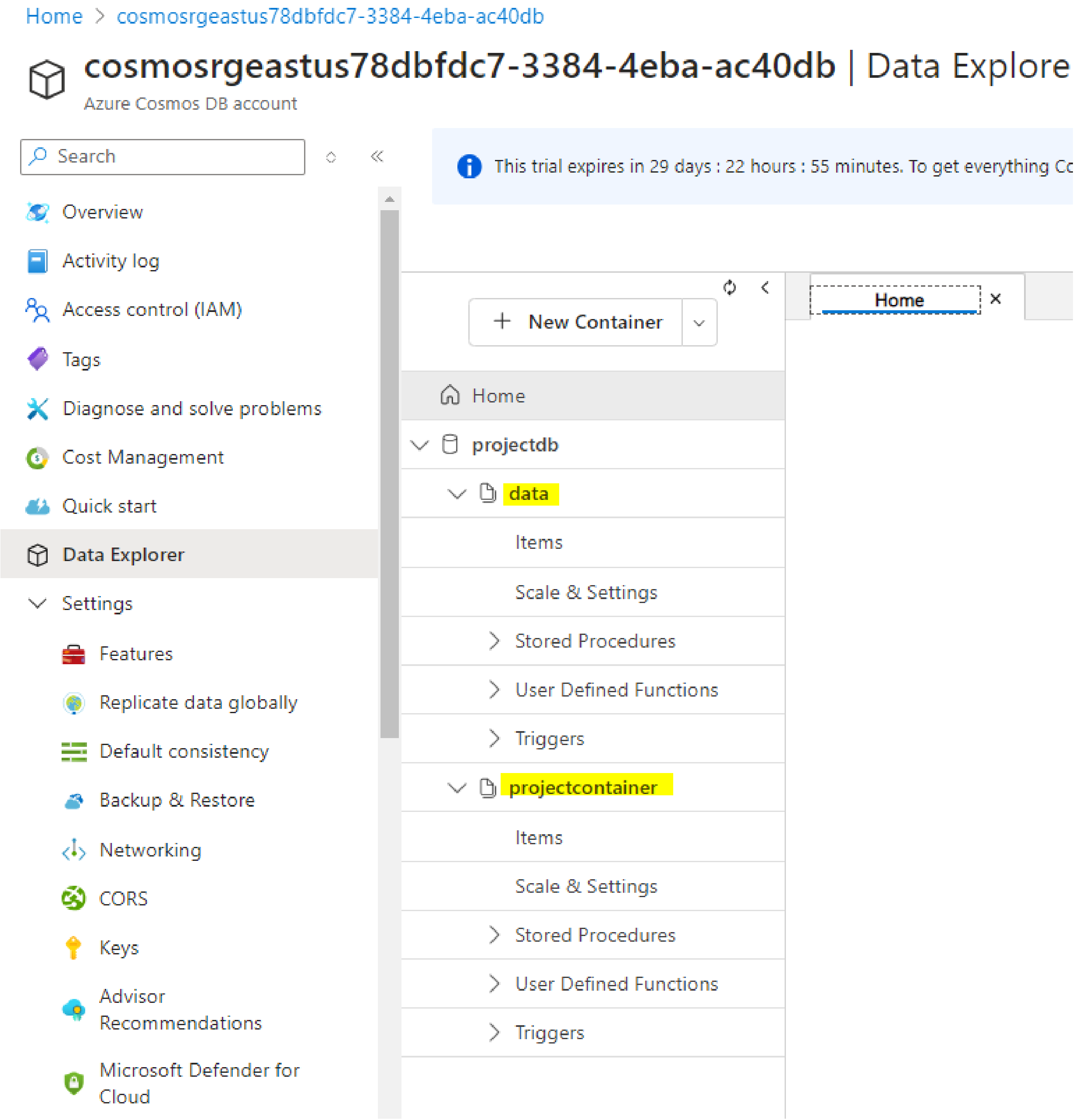
Role assignments (2)

Role	Description	Scope	Group assignment	Condition
Contributor	Grants full access to manage all r...	This resource	--	None
Reader	View all resources, but does not ...	Resource group (Inher...	--	None

Deny assignments (0)

Classic administrators (0)

Created a Database by the name 'projecdb' & containers 'projectcontainer' & 'data'



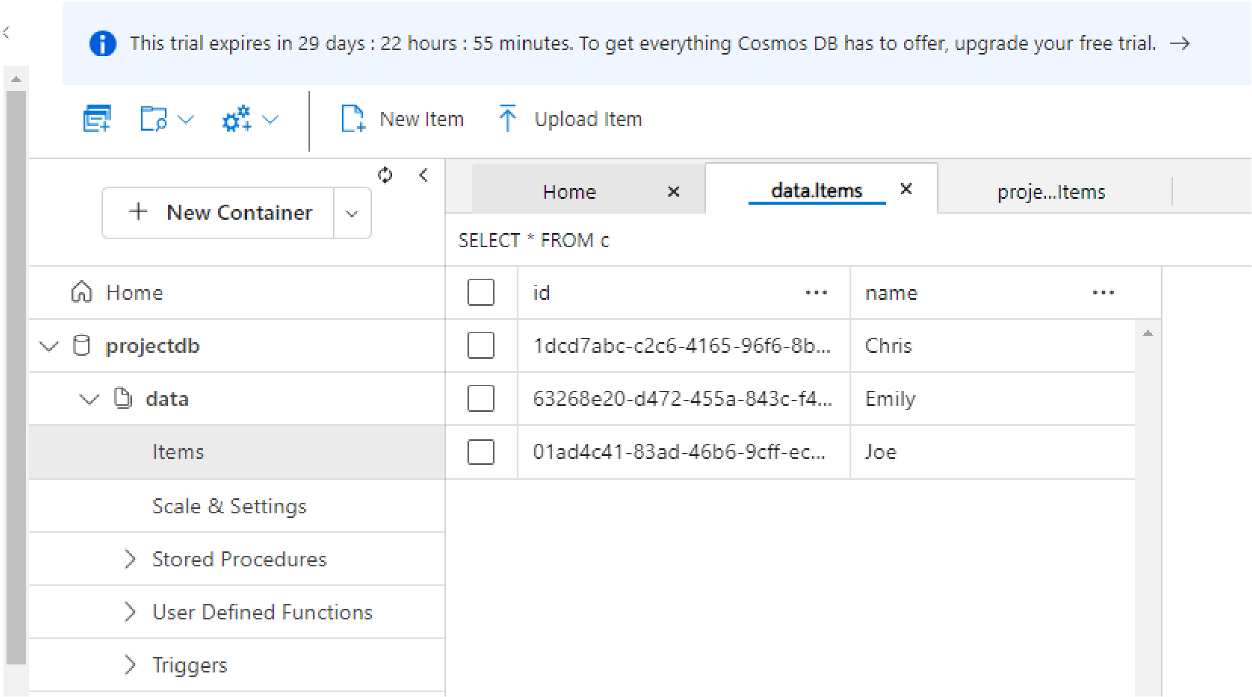
Home > cosmosrgeastus78dbfdc7-3384-4eba-ac40db

cosmosrgeastus78dbfdc7-3384-4eba-ac40db | Data Explore

Azure Cosmos DB account

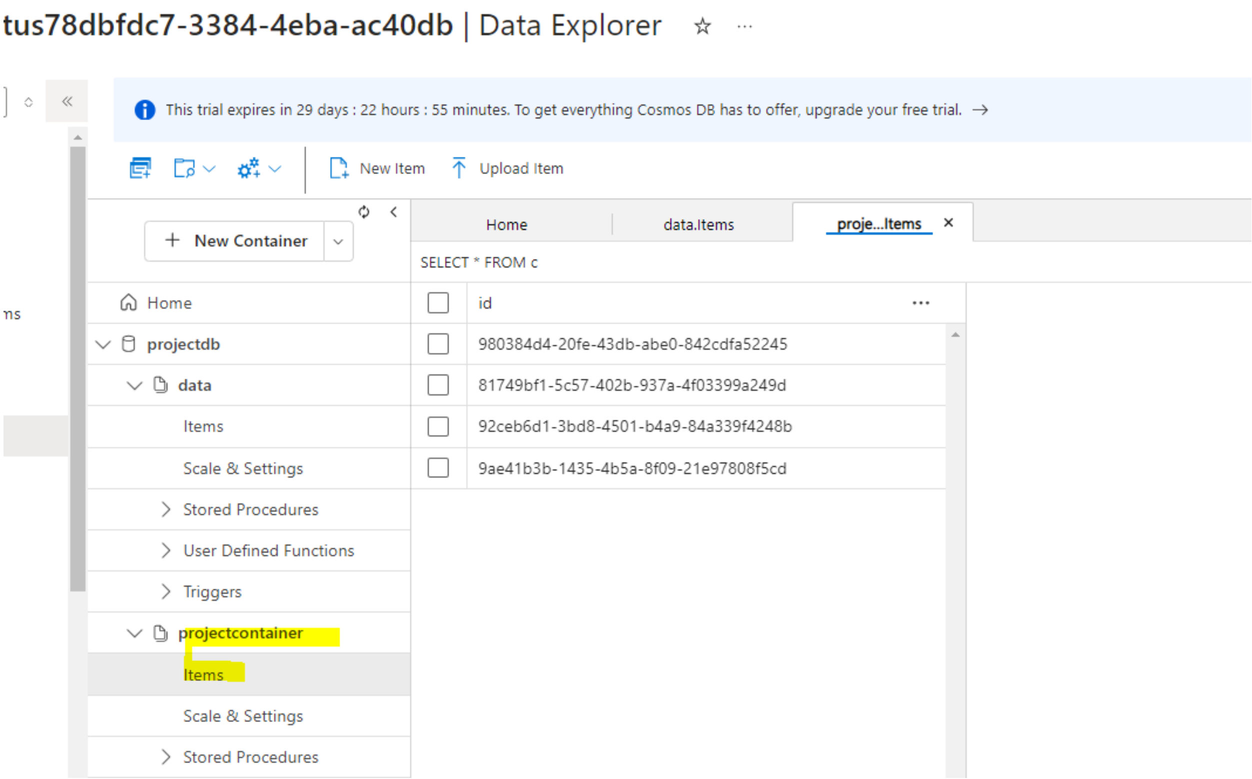
This trial expires in 29 days : 22 hours : 55 minutes. To get everything Co

Created items in both the containers as below with the partition id – name for Data Container:



SELECT * FROM c

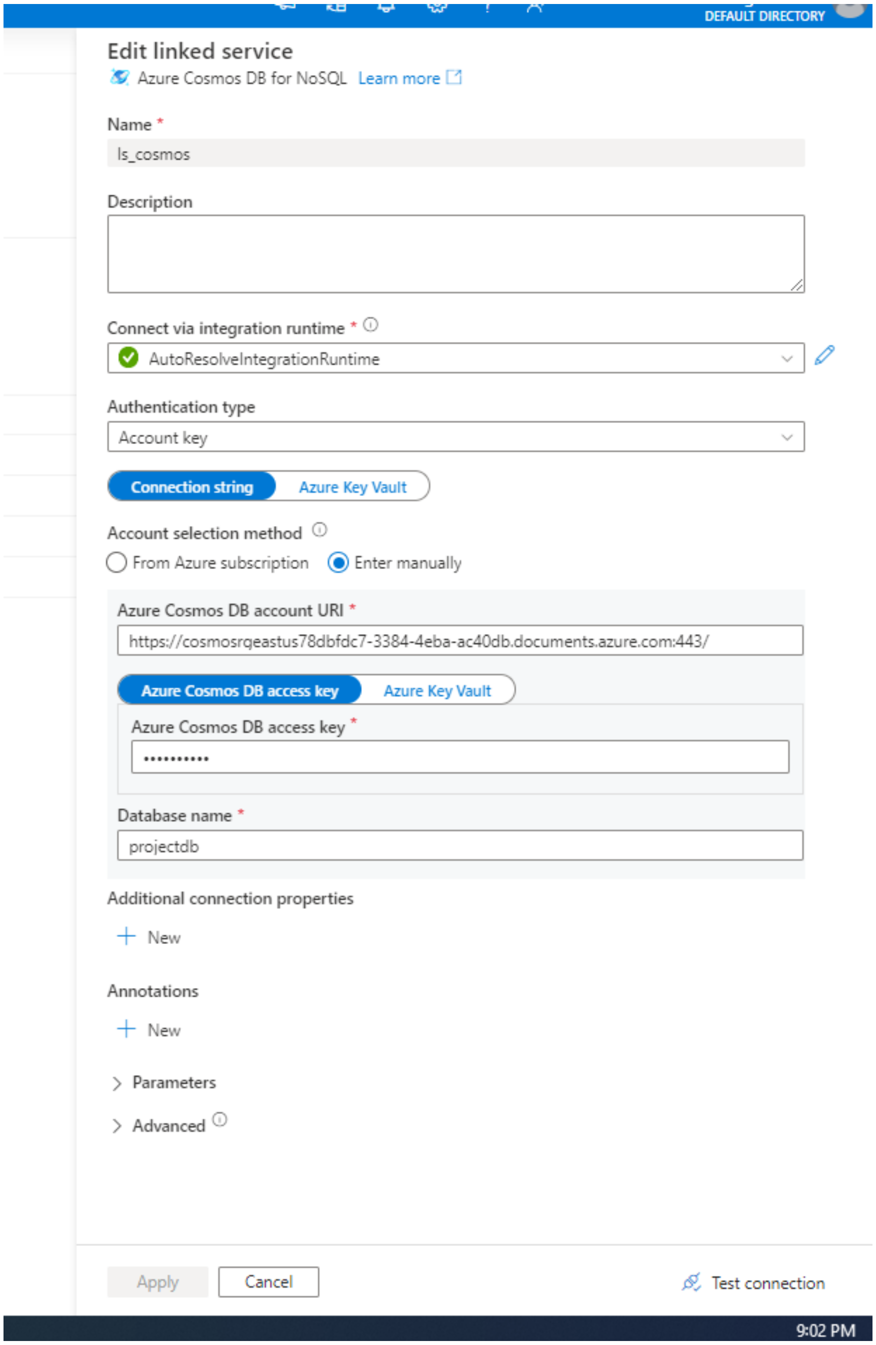
id	name
1dcd7abc-c2c6-4165-96f6-8b...	Chris
63268e20-d472-455a-843c-f4...	Emily
01ad4c41-83ad-46b6-9cff-ec...	Joe



SELECT * FROM c

id
980384d4-20fe-43db-abe0-842cdfa52245
81749bf1-5c57-402b-937a-4f03399a249d
92ceb6d1-3bd8-4501-b4a9-84a339f4248b
9ae41b3b-1435-4b5a-8f09-21e97808f5cd

Created Linked service for connecting the Cosmosdb to serverless SQL:



Edit linked service

Azure Cosmos DB for NoSQL Learn more

Name *

ls_cosmos

Description

Connect via integration runtime *

AutoResolveIntegrationRuntime

Authentication type

Account key

Connection string | Azure Key Vault

Account selection method

○ From Azure subscription ● Enter manually

Azure Cosmos DB account URI *

https://cosmosrqeastus78dbfdc7-3384-4eba-ac40db.documents.azure.com:443/

Azure Cosmos DB access key | Azure Key Vault

Azure Cosmos DB access key *

••••••••••

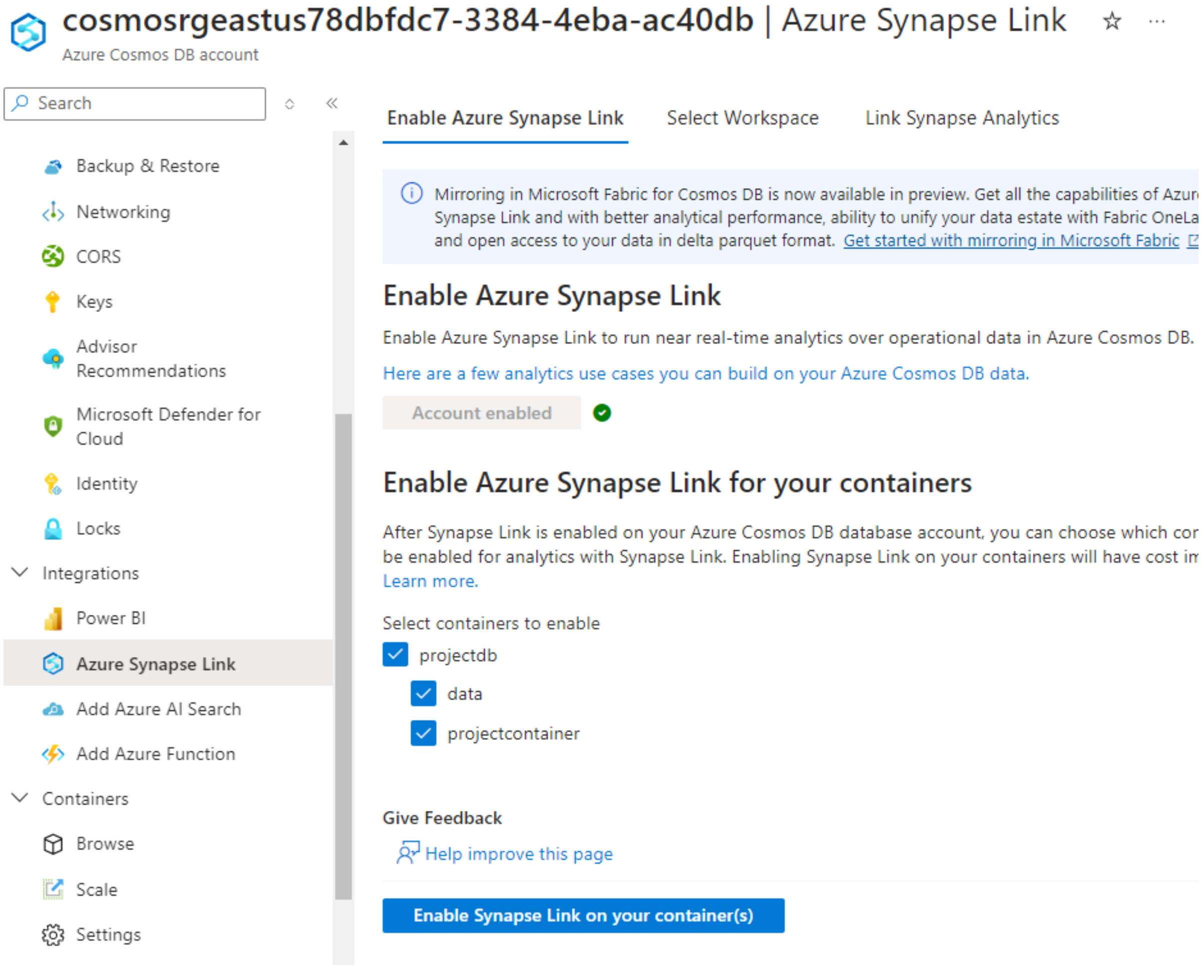
Database name *

projectdb

Additional connection properties

Annotations

Enabled Synapselink from cosmosdb from the left hand side of cosmosdb account:



cosmosrgeastus78dbfdc7-3384-4eba-ac40db | Azure Synapse Link

Azure Cosmos DB account

Mirroring in Microsoft Fabric for Cosmos DB is now available in preview. Get all the capabilities of Azur Synapse Link and with better analytical performance, ability to unify your data estate with Fabric OneLa and open access to your data in delta parquet format. Get started with mirroring in Microsoft Fabric

Enable Azure Synapse Link

Enable Azure Synapse Link to run near real-time analytics over operational data in Azure Cosmos DB.

Here are a few analytics use cases you can build on your Azure Cosmos DB data.

Account enabled ✓

Enable Azure Synapse Link for your containers

After Synapse Link is enabled on your Azure Cosmos DB database account, you can choose which cor be enabled for analytics with Synapse Link. Enabling Synapse Link on your containers will have cost in Learn more.

Select containers to enable

☑ projectdb

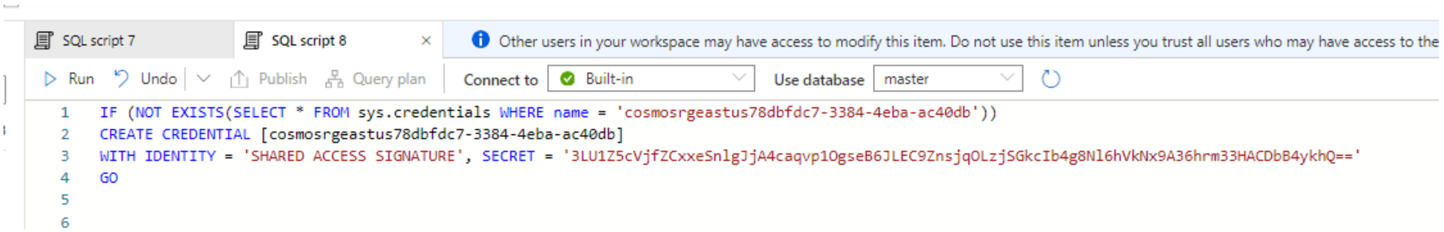
☑ data

☑ projectcontainer

Give Feedback

Help improve this page

Ran the below query to connect Synapse to Cosmos as below:



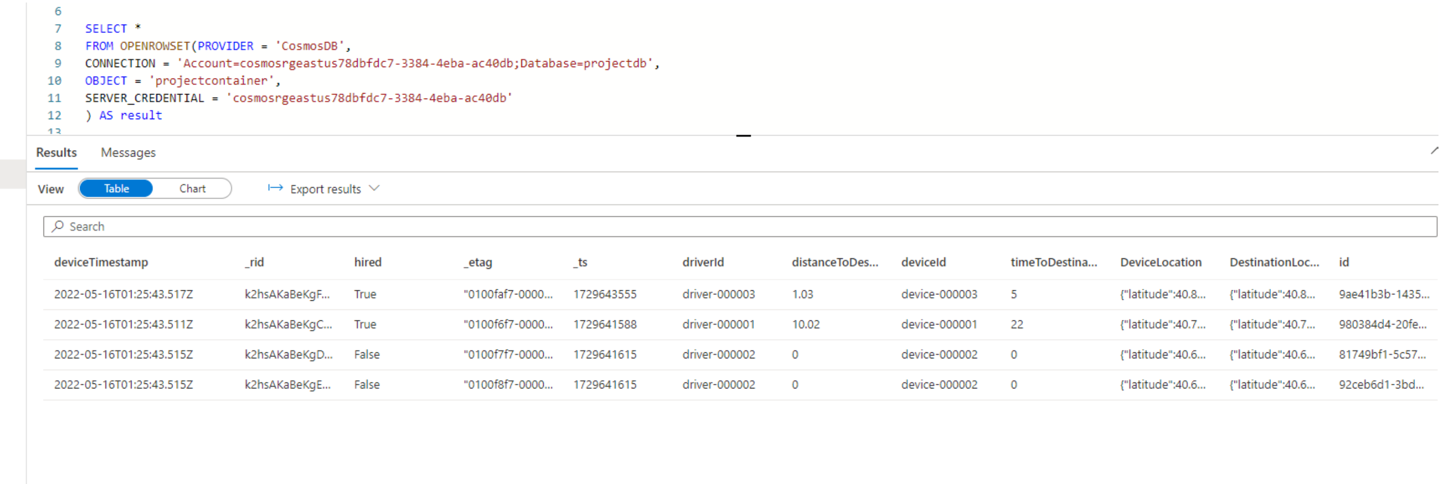
Other users in your workspace may have access to modify this item. Do not use this item unless you trust all users who may have access to the

Connect to: Built-in

Use database: master

```
IF (NOT EXISTS(SELECT * FROM sys.credentials WHERE name = 'cosmosrgeastus78dbfdc7-3384-4eba-ac40db'))
CREATE CREDENTIAL [cosmosrgeastus78dbfdc7-3384-4eba-ac40db]
WITH IDENTITY = 'SHARED ACCESS SIGNATURE', SECRET = '3LU1Z5cVjfZCxxeSnlgJjA4caqvp1OgseB6JLEC9ZnsjqOLzjSGkcIb4g8Nl6hVkNx9A36hrm33HACDbB4ykhQ=='
GO
```

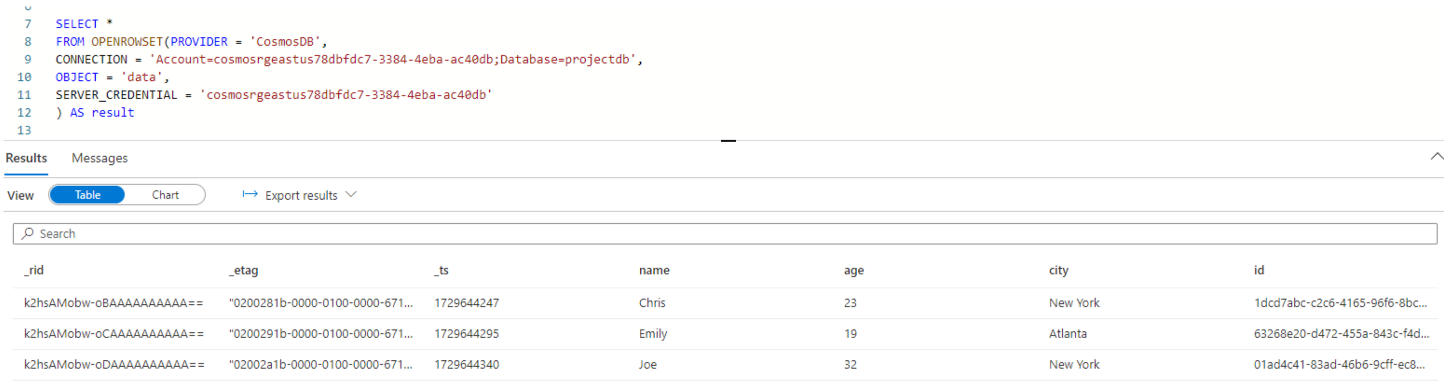
In Synapse, ran the below query to pull the items from cosmos db as below from container 1 - projectcontainer:



```
SELECT *
FROM OPENROWSET(PROVIDER = 'CosmosDB',
CONNECTION = 'Account=cosmosrgeastus78dbfdc7-3384-4eba-ac40db;Database=projectdb',
OBJECT = 'projectcontainer',
SERVER_CREDENTIAL = 'cosmosrgeastus78dbfdc7-3384-4eba-ac40db'
) AS result
```

deviceTimestamp	_rid	hired	_etag	_ts	driverId	distanceToDes...	deviceId	timeToDestina...	DeviceLocation	DestinationLoc...	id
2022-05-16T01:25:43.517Z	k2hsAKaBeKgF...	True	"0100faf7-0000...	1729643555	driver-000003	1.03	device-000003	5	{"latitude":40.8...	{"latitude":40.8...	9ae41b3b-1435...
2022-05-16T01:25:43.511Z	k2hsAKaBeKgC...	True	"0100f6f7-0000...	1729641588	driver-000001	10.02	device-000001	22	{"latitude":40.7...	{"latitude":40.7...	980384d4-20fe...
2022-05-16T01:25:43.515Z	k2hsAKaBeKgD...	False	"0100f7f7-0000...	1729641615	driver-000002	0	device-000002	0	{"latitude":40.6...	{"latitude":40.6...	81749bf1-5c57...
2022-05-16T01:25:43.515Z	k2hsAKaBeKgE...	False	"0100f8f7-0000...	1729641615	driver-000002	0	device-000002	0	{"latitude":40.6...	{"latitude":40.6...	92ceb6d1-3bd...

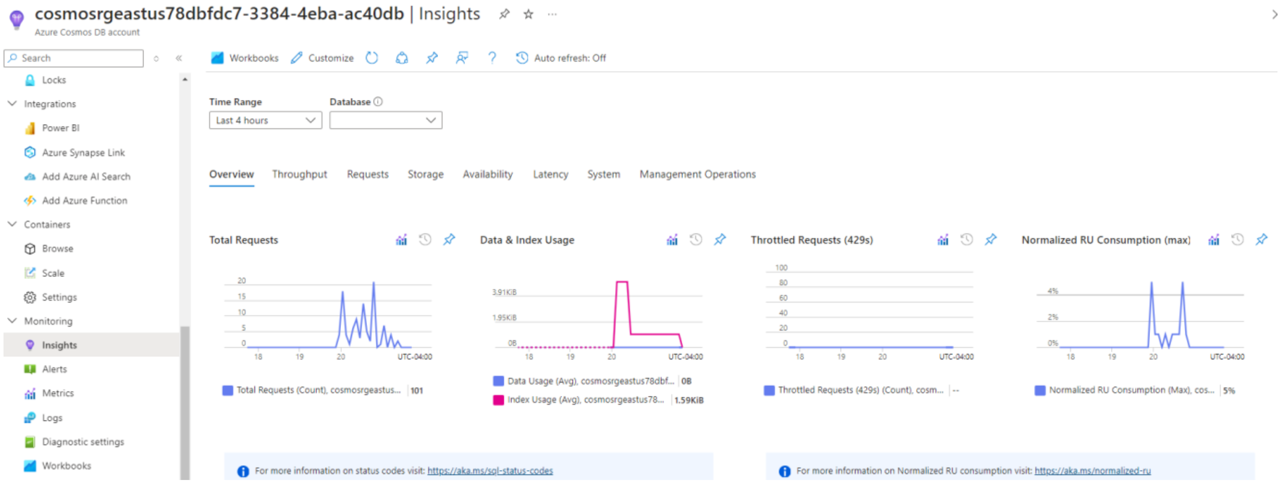
In Synapse, ran the below query to pull the items from cosmos db as below from container 2 -data:



```
SELECT *
FROM OPENROWSET(PROVIDER = 'CosmosDB',
CONNECTION = 'Account=cosmosrgeastus78dbfdc7-3384-4eba-ac40db;Database=projectdb',
OBJECT = 'data',
SERVER_CREDENTIAL = 'cosmosrgeastus78dbfdc7-3384-4eba-ac40db'
) AS result
```

_rid	_etag	_ts	name	age	city	id
k2hsAMobw-oBAAAAAAAAAA==	"0200281b-0000-0100-0000-671...	1729644247	Chris	23	New York	1dcd7abc-c2c6-4165-96f6-8bc...
k2hsAMobw-oCAAAAAAAAAA==	"0200291b-0000-0100-0000-671...	1729644295	Emily	19	Atlanta	63268e20-d472-455a-843c-f4d...
k2hsAMobw-oDAAAAAAAAAA==	"02002a1b-0000-0100-0000-671...	1729644340	Joe	32	New York	01ad4c41-83ad-46b6-9cff-ec8...

Visual of Cosmos DB Insights:



For more information on status codes visit: https://aka.ms/sql-status-codes

For more information on Normalized RU consumption visit: https://aka.ms/normalized-ru

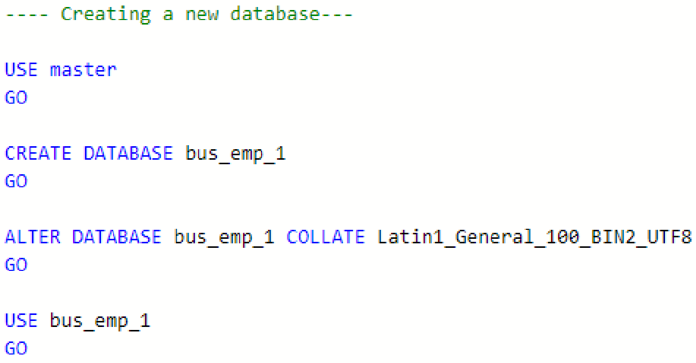
Azure Synapse Serverless Pool:

- Azure Synapse Workspace: Used an Azure Synapse workspace Severless SQL Pool.
- Azure Data Lake Gen 2 Storage: Used the ADLS Gen 2 account uploaded the data and used it as a source.
- Business Employment Data: Used the previously uploaded dataset called business employment CSV format for the project

Data ingestion & preparation:

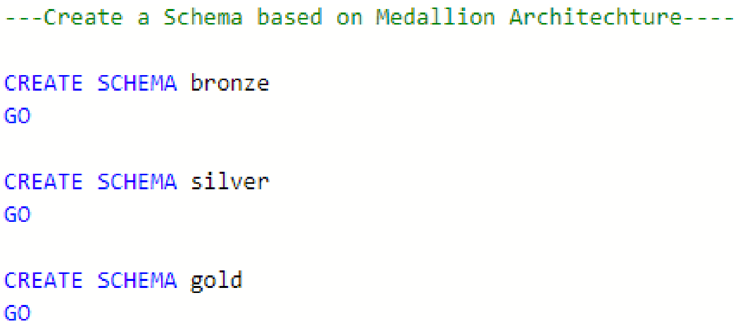
The previously uploaded in ADLS account dataset called business_employment in CSV format for the project.

Created a new database



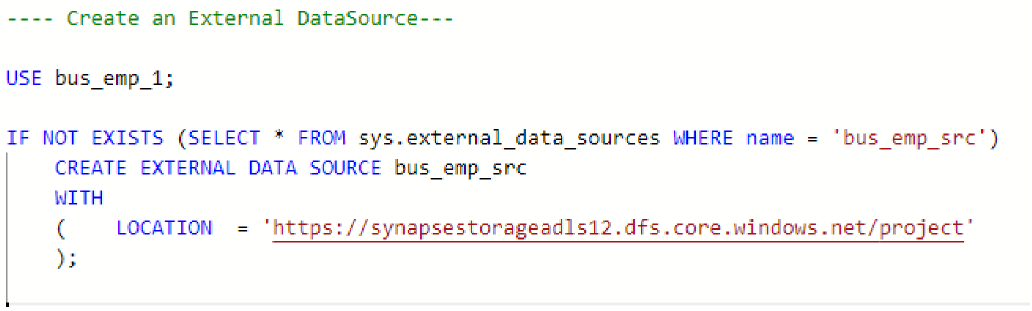
```
---- Creating a new database---

USE master
GO

CREATE DATABASE bus_emp_1
GO

ALTER DATABASE bus_emp_1 COLLATE Latin1_General_100_BIN2_UTF8
GO

USE bus_emp_1
GO
```

Created schema in the Database as below:



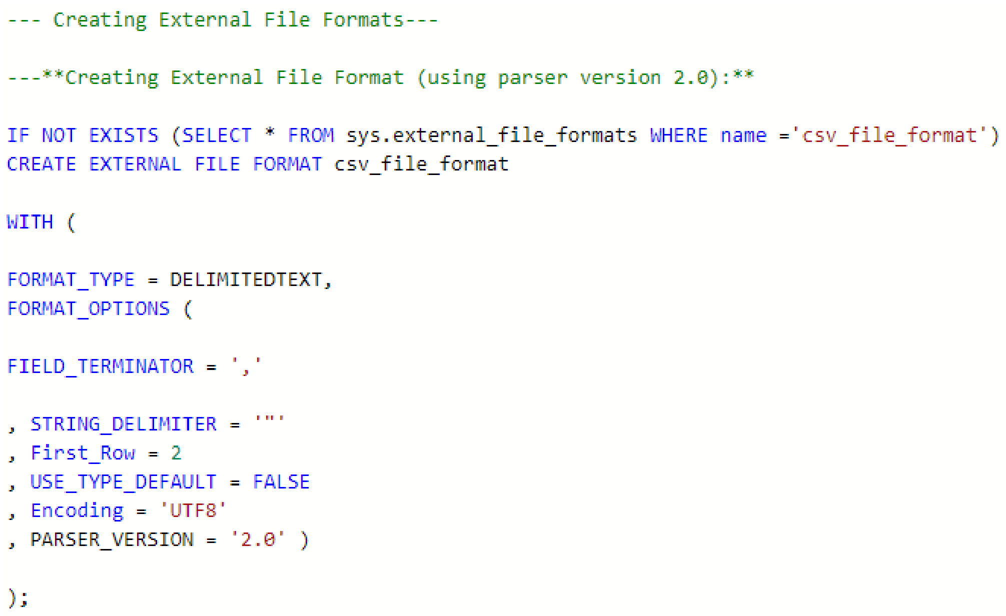
```
---Create a Schema based on Medallion Architechture----

CREATE SCHEMA bronze
GO

CREATE SCHEMA silver
GO

CREATE SCHEMA gold
GO
```

Created External Data Source Pointing towards our ADLS Source Storage:

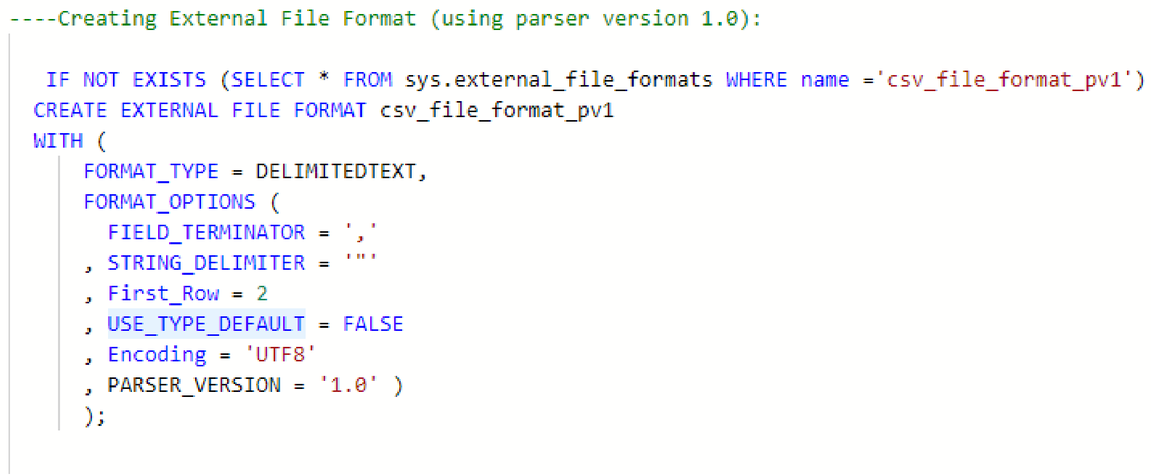


```
---- Create an External DataSource---

USE bus_emp_1;

IF NOT EXISTS (SELECT * FROM sys.external_data_sources WHERE name = 'bus_emp_src')
    CREATE EXTERNAL DATA SOURCE bus_emp_src
    WITH
    (    LOCATION   = 'https://synapsestorageadls12.dfs.core.windows.net/project'
    );
```

Created External File Formats for the CSV Formats:

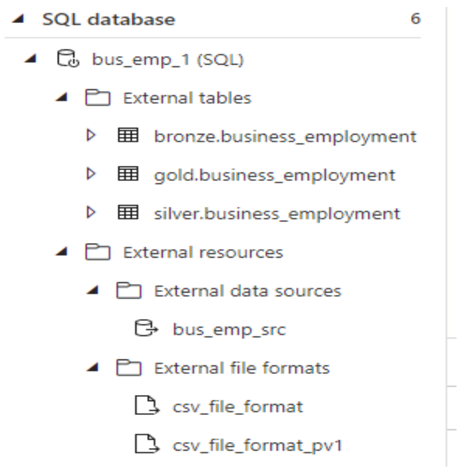


```
--- Creating External File Formats---

---**Creating External File Format (using parser version 2.0):**

IF NOT EXISTS (SELECT * FROM sys.external_file_formats WHERE name ='csv_file_format')
CREATE EXTERNAL FILE FORMAT csv_file_format

WITH (

FORMAT_TYPE = DELIMITEDTEXT,
FORMAT_OPTIONS (

FIELD_TERMINATOR = ','

, STRING_DELIMITER = '"'
, First_Row = 2
, USE_TYPE_DEFAULT = FALSE
, Encoding = 'UTF8'
, PARSER_VERSION = '2.0' )

);
```



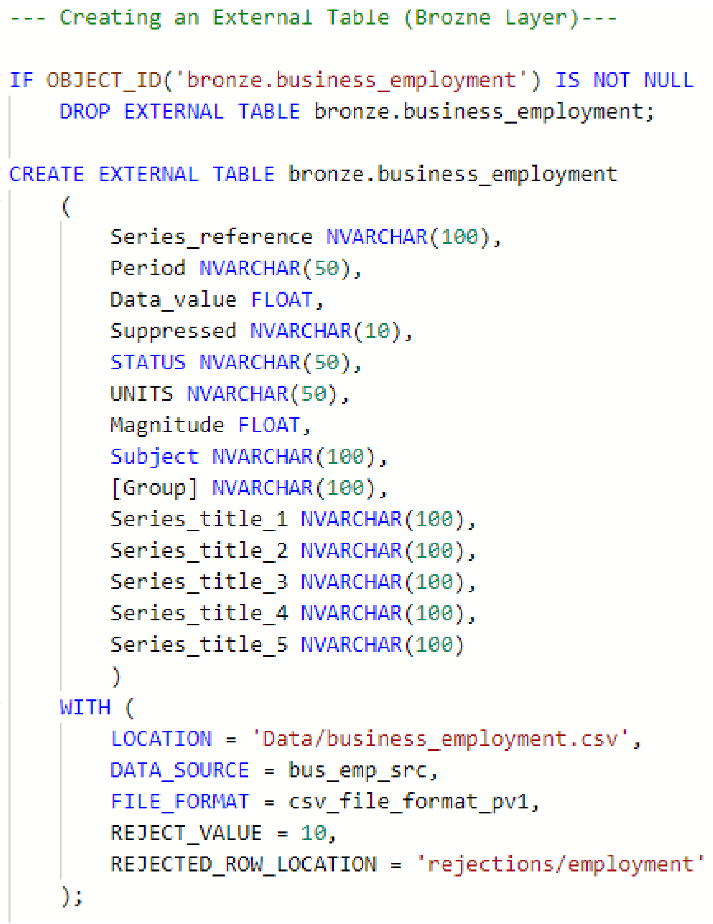
```
----Creating External File Format (using parser version 1.0):

 IF NOT EXISTS (SELECT * FROM sys.external_file_formats WHERE name ='csv_file_format_pv1')
CREATE EXTERNAL FILE FORMAT csv_file_format_pv1
WITH (
    FORMAT_TYPE = DELIMITEDTEXT,
    FORMAT_OPTIONS (
      FIELD_TERMINATOR = ','
    , STRING_DELIMITER = '"'
    , First_Row = 2
    , USE_TYPE_DEFAULT = FALSE
    , Encoding = 'UTF8'
    , PARSER_VERSION = '1.0' )
    );
```

See below for the setup created on Synapse for Database, External Tables, data source, file formats:



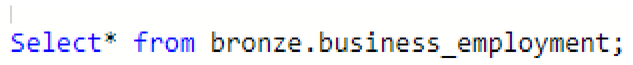
```
◢ SQL database                                6
  ◢ bus_emp_1 (SQL)
    ◢ External tables
      ▷ bronze.business_employment
      ▷ gold.business_employment
      ▷ silver.business_employment
    ◢ External resources
      ◢ External data sources
          bus_emp_src
      ◢ External file formats
          csv_file_format
          csv_file_format_pv1
```

Created External Table for the Bronze Layer:

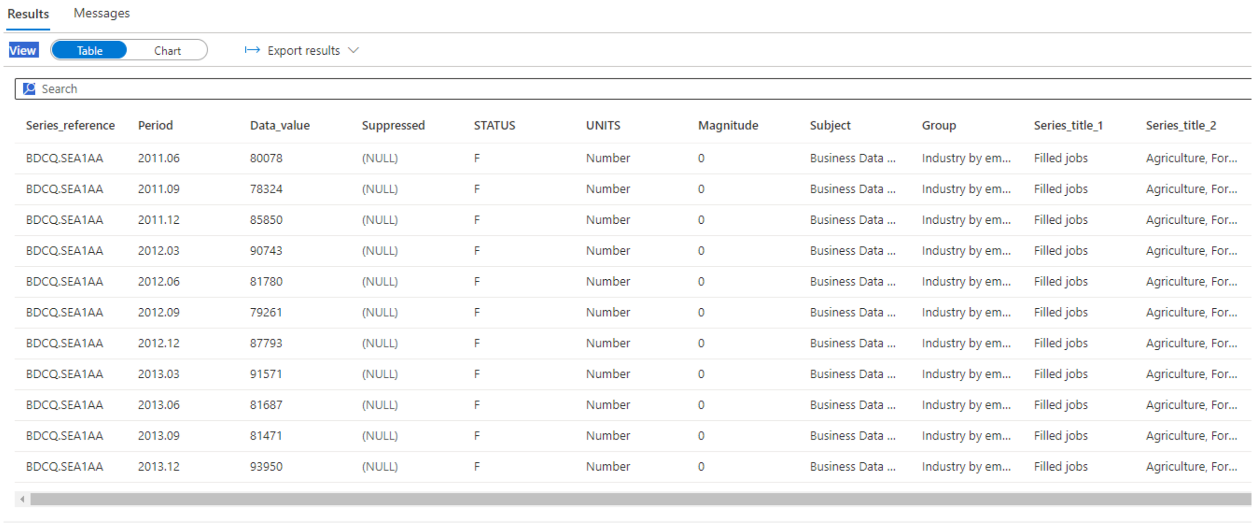


```
--- Creating an External Table (Brozne Layer)---

IF OBJECT_ID('bronze.business_employment') IS NOT NULL
    DROP EXTERNAL TABLE bronze.business_employment;

CREATE EXTERNAL TABLE bronze.business_employment
    (
        Series_reference NVARCHAR(100),
        Period NVARCHAR(50),
        Data_value FLOAT,
        Suppressed NVARCHAR(10),
        STATUS NVARCHAR(50),
        UNITS NVARCHAR(50),
        Magnitude FLOAT,
        Subject NVARCHAR(100),
        [Group] NVARCHAR(100),
        Series_title_1 NVARCHAR(100),
        Series_title_2 NVARCHAR(100),
        Series_title_3 NVARCHAR(100),
        Series_title_4 NVARCHAR(100),
        Series_title_5 NVARCHAR(100)
        )
    WITH (
        LOCATION = 'Data/business_employment.csv',
        DATA_SOURCE = bus_emp_src,
        FILE_FORMAT = csv_file_format_pv1,
        REJECT_VALUE = 10,
        REJECTED_ROW_LOCATION = 'rejections/employment'
    );
```

The below is our source data stored in Bronze Layer:

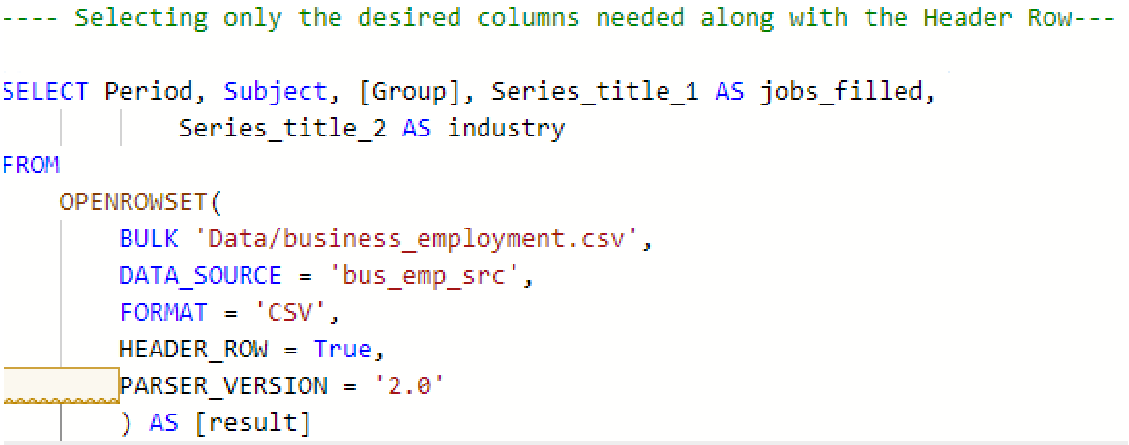


```
Select* from bronze.business_employment;
```

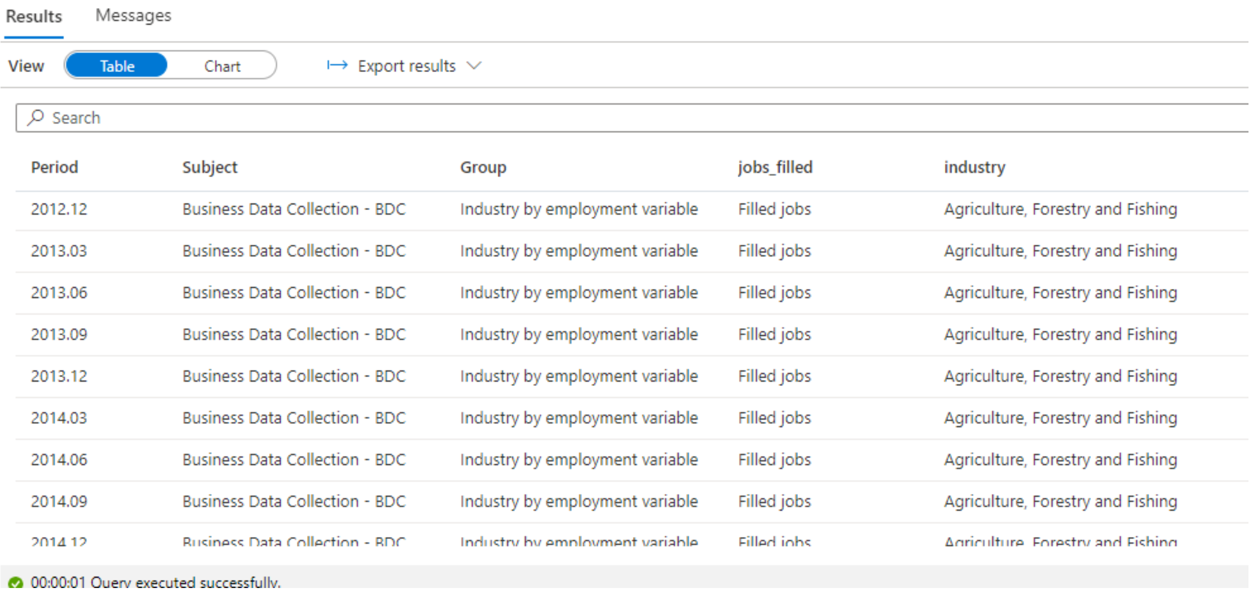


Series_reference	Period	Data_value	Suppressed	STATUS	UNITS	Magnitude	Subject	Group	Series_title_1	Series_title_2
BDCQ.SEA1AA	2011.06	80078	(NULL)	F	Number	0	Business Data ...	Industry by em...	Filled jobs	Agriculture, For...
BDCQ.SEA1AA	2011.09	78324	(NULL)	F	Number	0	Business Data ...	Industry by em...	Filled jobs	Agriculture, For...
BDCQ.SEA1AA	2011.12	85850	(NULL)	F	Number	0	Business Data ...	Industry by em...	Filled jobs	Agriculture, For...
BDCQ.SEA1AA	2012.03	90743	(NULL)	F	Number	0	Business Data ...	Industry by em...	Filled jobs	Agriculture, For...
BDCQ.SEA1AA	2012.06	81780	(NULL)	F	Number	0	Business Data ...	Industry by em...	Filled jobs	Agriculture, For...
BDCQ.SEA1AA	2012.09	79261	(NULL)	F	Number	0	Business Data ...	Industry by em...	Filled jobs	Agriculture, For...
BDCQ.SEA1AA	2012.12	87793	(NULL)	F	Number	0	Business Data ...	Industry by em...	Filled jobs	Agriculture, For...
BDCQ.SEA1AA	2013.03	91571	(NULL)	F	Number	0	Business Data ...	Industry by em...	Filled jobs	Agriculture, For...
BDCQ.SEA1AA	2013.06	81687	(NULL)	F	Number	0	Business Data ...	Industry by em...	Filled jobs	Agriculture, For...
BDCQ.SEA1AA	2013.09	81471	(NULL)	F	Number	0	Business Data ...	Industry by em...	Filled jobs	Agriculture, For...
BDCQ.SEA1AA	2013.12	93950	(NULL)	F	Number	0	Business Data ...	Industry by em...	Filled jobs	Agriculture, For...

Data Cleaning & Transformations:

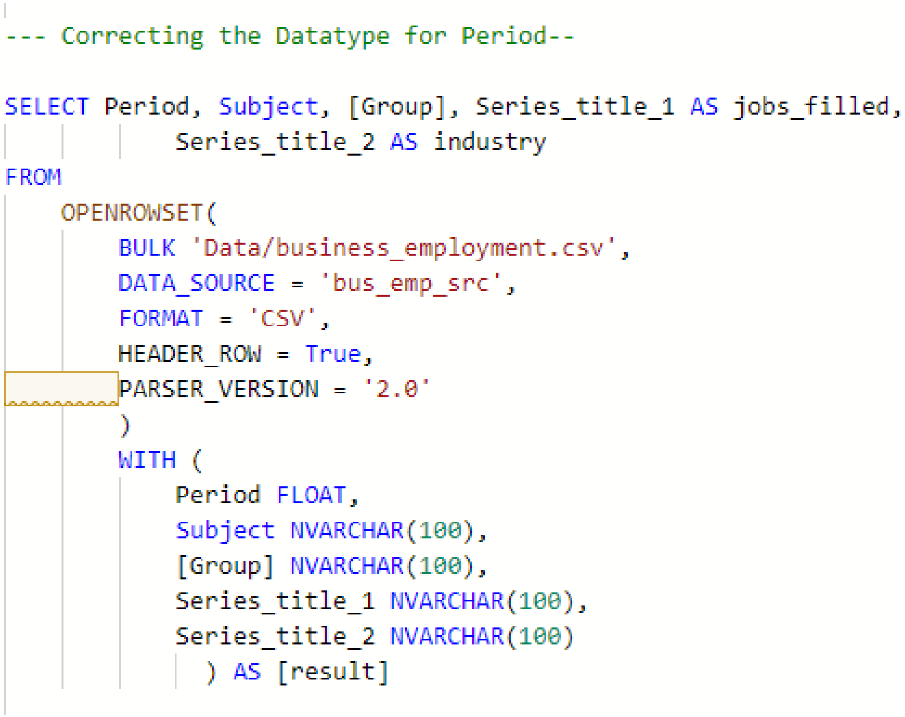


```
---- Selecting only the desired columns needed along with the Header Row---

SELECT Period, Subject, [Group], Series_title_1 AS jobs_filled,
            Series_title_2 AS industry
FROM
    OPENROWSET(
        BULK 'Data/business_employment.csv',
        DATA_SOURCE = 'bus_emp_src',
        FORMAT = 'CSV',
        HEADER_ROW = True,
        PARSER_VERSION = '2.0'
        ) AS [result]
```

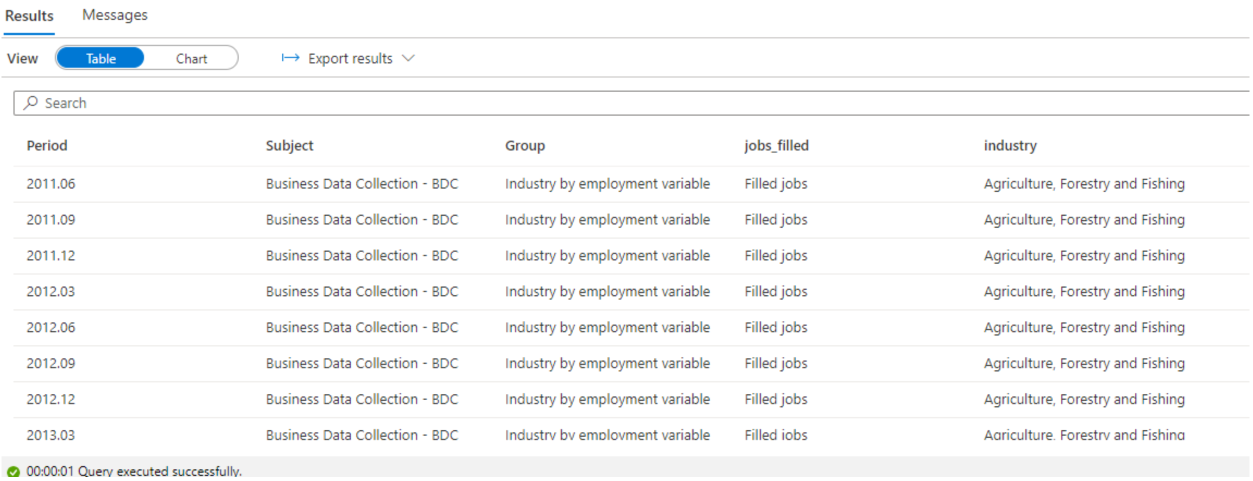


Period	Subject	Group	jobs_filled	industry
2012.12	Business Data Collection - BDC	Industry by employment variable	Filled jobs	Agriculture, Forestry and Fishing
2013.03	Business Data Collection - BDC	Industry by employment variable	Filled jobs	Agriculture, Forestry and Fishing
2013.06	Business Data Collection - BDC	Industry by employment variable	Filled jobs	Agriculture, Forestry and Fishing
2013.09	Business Data Collection - BDC	Industry by employment variable	Filled jobs	Agriculture, Forestry and Fishing
2013.12	Business Data Collection - BDC	Industry by employment variable	Filled jobs	Agriculture, Forestry and Fishing
2014.03	Business Data Collection - BDC	Industry by employment variable	Filled jobs	Agriculture, Forestry and Fishing
2014.06	Business Data Collection - BDC	Industry by employment variable	Filled jobs	Agriculture, Forestry and Fishing
2014.09	Business Data Collection - BDC	Industry by employment variable	Filled jobs	Agriculture, Forestry and Fishing
2014.12	Business Data Collection - BDC	Industry by employment variable	Filled jobs	Agriculture, Forestry and Fishing

00:00:01 Query executed successfully.

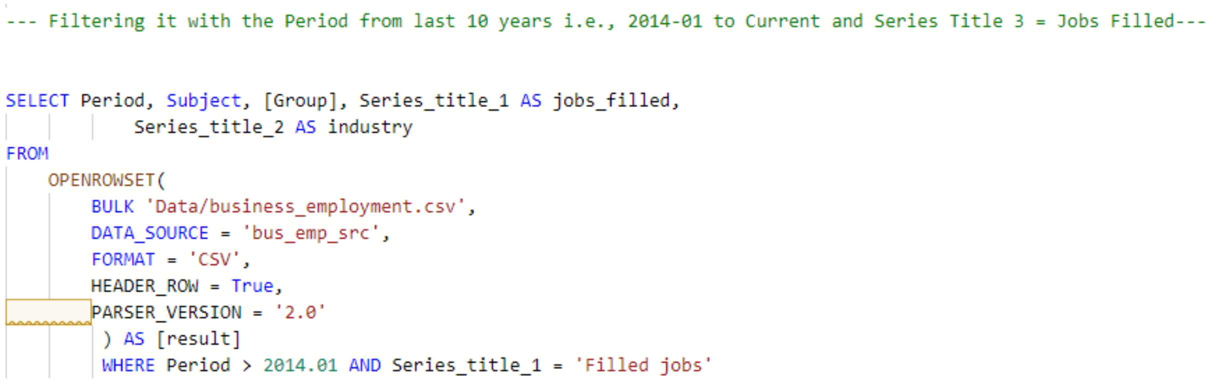


```
--- Correcting the Datatype for Period--

SELECT Period, Subject, [Group], Series_title_1 AS jobs_filled,
            Series_title_2 AS industry
FROM
    OPENROWSET(
        BULK 'Data/business_employment.csv',
        DATA_SOURCE = 'bus_emp_src',
        FORMAT = 'CSV',
        HEADER_ROW = True,
        PARSER_VERSION = '2.0'
        )
        WITH (
            Period FLOAT,
            Subject NVARCHAR(100),
            [Group] NVARCHAR(100),
            Series_title_1 NVARCHAR(100),
            Series_title_2 NVARCHAR(100)
              ) AS [result]
```

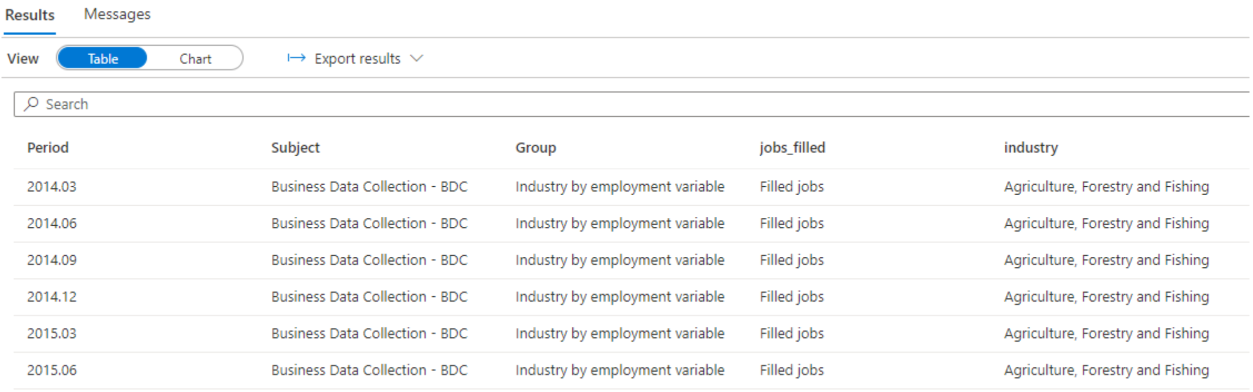


Period	Subject	Group	jobs_filled	industry
2011.06	Business Data Collection - BDC	Industry by employment variable	Filled jobs	Agriculture, Forestry and Fishing
2011.09	Business Data Collection - BDC	Industry by employment variable	Filled jobs	Agriculture, Forestry and Fishing
2011.12	Business Data Collection - BDC	Industry by employment variable	Filled jobs	Agriculture, Forestry and Fishing
2012.03	Business Data Collection - BDC	Industry by employment variable	Filled jobs	Agriculture, Forestry and Fishing
2012.06	Business Data Collection - BDC	Industry by employment variable	Filled jobs	Agriculture, Forestry and Fishing
2012.09	Business Data Collection - BDC	Industry by employment variable	Filled jobs	Agriculture, Forestry and Fishing
2012.12	Business Data Collection - BDC	Industry by employment variable	Filled jobs	Agriculture, Forestry and Fishing
2013.03	Business Data Collection - BDC	Industry by employment variable	Filled jobs	Agriculture, Forestry and Fishing

00:00:01 Query executed successfully.

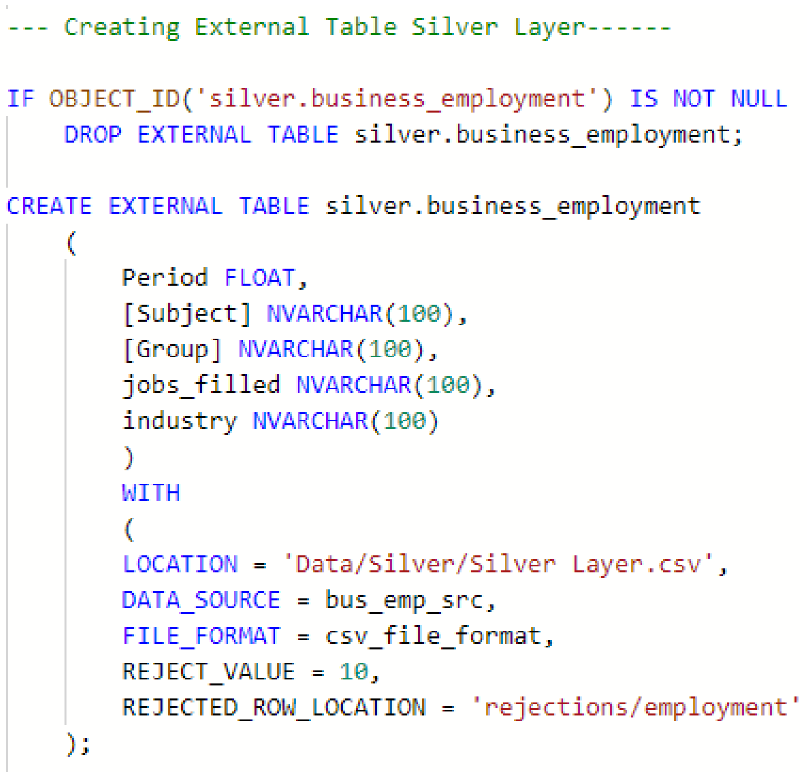


```
--- Filtering it with the Period from last 10 years i.e., 2014-01 to Current and Series Title 3 = Jobs Filled---


SELECT Period, Subject, [Group], Series_title_1 AS jobs_filled,
            Series_title_2 AS industry
FROM
    OPENROWSET(
        BULK 'Data/business_employment.csv',
        DATA_SOURCE = 'bus_emp_src',
        FORMAT = 'CSV',
        HEADER_ROW = True,
        PARSER_VERSION = '2.0'
         ) AS [result]
         WHERE Period > 2014.01 AND Series_title_1 = 'Filled jobs'
```

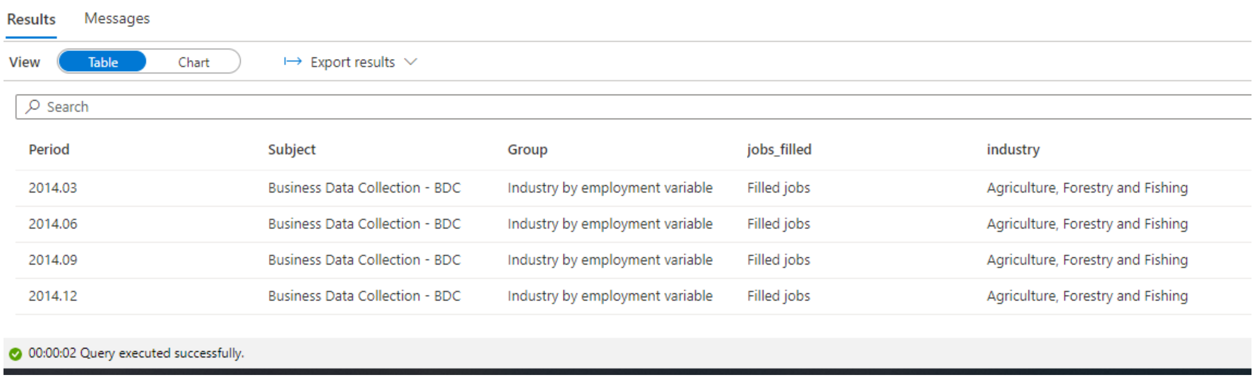


Period	Subject	Group	jobs_filled	industry
2014.03	Business Data Collection - BDC	Industry by employment variable	Filled jobs	Agriculture, Forestry and Fishing
2014.06	Business Data Collection - BDC	Industry by employment variable	Filled jobs	Agriculture, Forestry and Fishing
2014.09	Business Data Collection - BDC	Industry by employment variable	Filled jobs	Agriculture, Forestry and Fishing
2014.12	Business Data Collection - BDC	Industry by employment variable	Filled jobs	Agriculture, Forestry and Fishing
2015.03	Business Data Collection - BDC	Industry by employment variable	Filled jobs	Agriculture, Forestry and Fishing
2015.06	Business Data Collection - BDC	Industry by employment variable	Filled jobs	Agriculture, Forestry and Fishing

Created Silver Layer External Table after the Transformation:



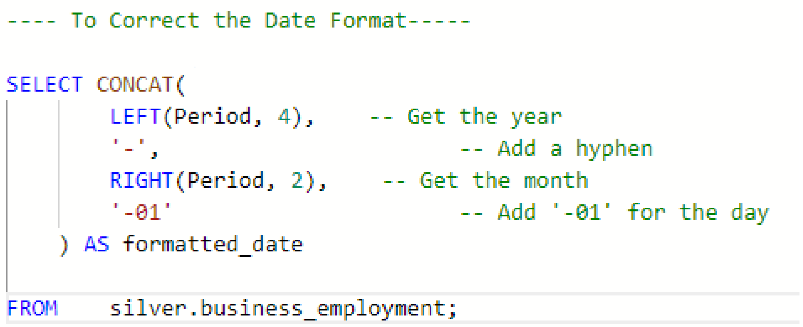
```
--- Creating External Table Silver Layer------

IF OBJECT_ID('silver.business_employment') IS NOT NULL
    DROP EXTERNAL TABLE silver.business_employment;

CREATE EXTERNAL TABLE silver.business_employment
    (
        Period FLOAT,
        [Subject] NVARCHAR(100),
        [Group] NVARCHAR(100),
        jobs_filled NVARCHAR(100),
        industry NVARCHAR(100)
        )
        WITH
        (
        LOCATION = 'Data/Silver/Silver Layer.csv',
        DATA_SOURCE = bus_emp_src,
        FILE_FORMAT = csv_file_format,
        REJECT_VALUE = 10,
        REJECTED_ROW_LOCATION = 'rejections/employment'
    );
```



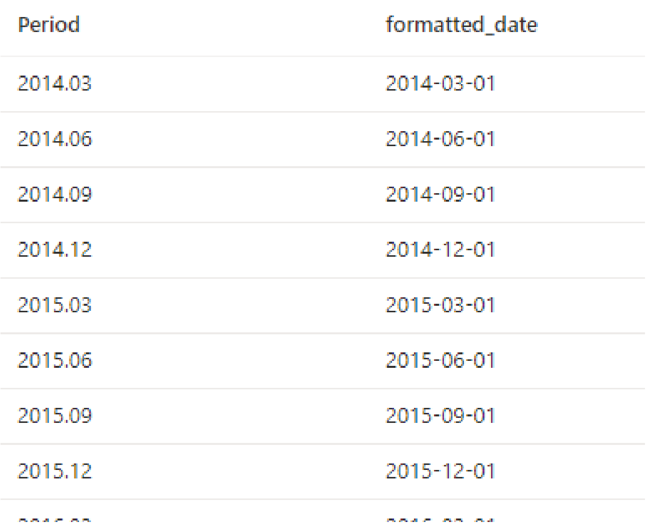
Period	Subject	Group	jobs_filled	industry
2014.03	Business Data Collection - BDC	Industry by employment variable	Filled jobs	Agriculture, Forestry and Fishing
2014.06	Business Data Collection - BDC	Industry by employment variable	Filled jobs	Agriculture, Forestry and Fishing
2014.09	Business Data Collection - BDC	Industry by employment variable	Filled jobs	Agriculture, Forestry and Fishing
2014.12	Business Data Collection - BDC	Industry by employment variable	Filled jobs	Agriculture, Forestry and Fishing

00:00:02 Query executed successfully.

Corrected the Date format from YYYY.MM to YYYY-MM-DD.

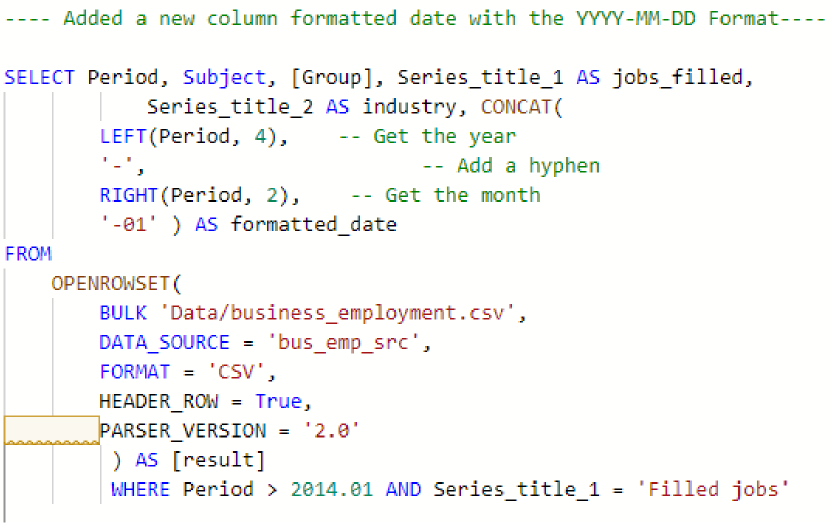


```
---- To Correct the Date Format-----

SELECT CONCAT(
        LEFT(Period, 4),    -- Get the year
        '-',                       -- Add a hyphen
        RIGHT(Period, 2),    -- Get the month
        '-01'                      -- Add '-01' for the day
    ) AS formatted_date

FROM    silver.business_employment;
```

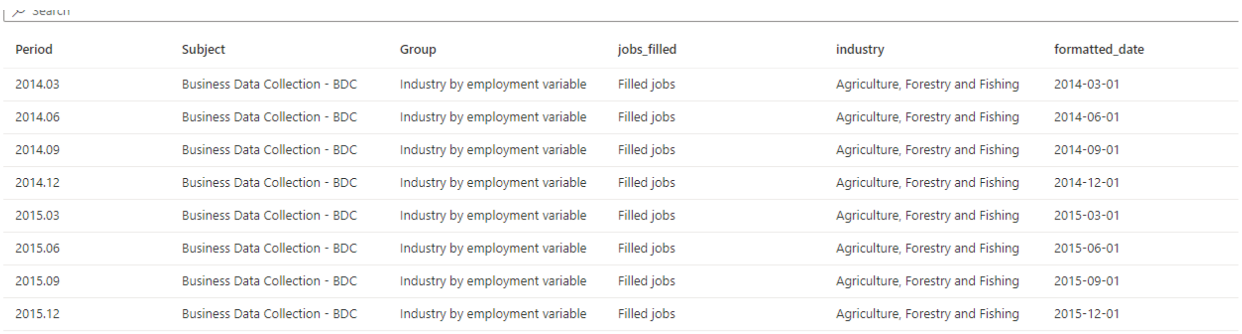


Period	formatted_date
2014.03	2014-03-01
2014.06	2014-06-01
2014.09	2014-09-01
2014.12	2014-12-01
2015.03	2015-03-01
2015.06	2015-06-01
2015.09	2015-09-01
2015.12	2015-12-01

Added a new column to insert the formatted date see below:

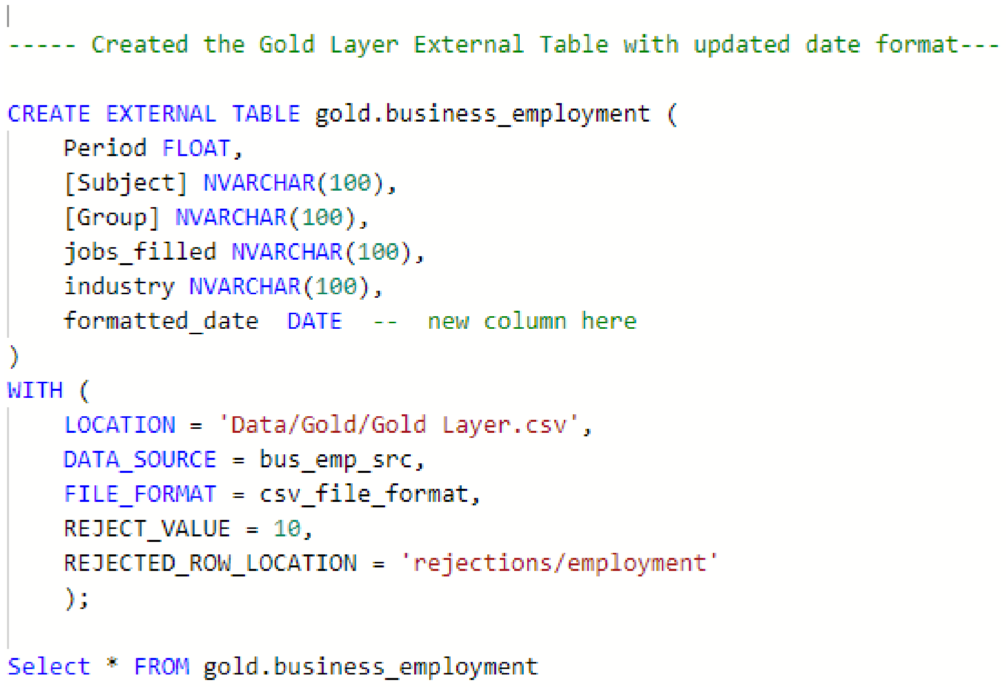


```
---- Added a new column formatted date with the YYYY-MM-DD Format----

SELECT Period, Subject, [Group], Series_title_1 AS jobs_filled,
            Series_title_2 AS industry, CONCAT(
        LEFT(Period, 4),     -- Get the year
        '-',                        -- Add a hyphen
        RIGHT(Period, 2),     -- Get the month
        '-01' ) AS formatted_date
FROM
    OPENROWSET(
        BULK 'Data/business_employment.csv',
        DATA_SOURCE = 'bus_emp_src',
        FORMAT = 'CSV',
        HEADER_ROW = True,
        PARSER_VERSION = '2.0'
         ) AS [result]
         WHERE Period > 2014.01 AND Series_title_1 = 'Filled jobs'
```

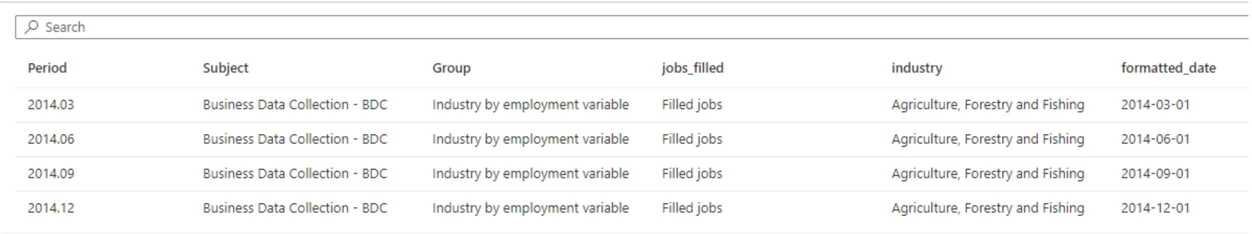


Period	Subject	Group	jobs_filled	industry	formatted_date
2014.03	Business Data Collection - BDC	Industry by employment variable	Filled jobs	Agriculture, Forestry and Fishing	2014-03-01
2014.06	Business Data Collection - BDC	Industry by employment variable	Filled jobs	Agriculture, Forestry and Fishing	2014-06-01
2014.09	Business Data Collection - BDC	Industry by employment variable	Filled jobs	Agriculture, Forestry and Fishing	2014-09-01
2014.12	Business Data Collection - BDC	Industry by employment variable	Filled jobs	Agriculture, Forestry and Fishing	2014-12-01
2015.03	Business Data Collection - BDC	Industry by employment variable	Filled jobs	Agriculture, Forestry and Fishing	2015-03-01
2015.06	Business Data Collection - BDC	Industry by employment variable	Filled jobs	Agriculture, Forestry and Fishing	2015-06-01
2015.09	Business Data Collection - BDC	Industry by employment variable	Filled jobs	Agriculture, Forestry and Fishing	2015-09-01
2015.12	Business Data Collection - BDC	Industry by employment variable	Filled jobs	Agriculture, Forestry and Fishing	2015-12-01

Created Gold Layer External Table after the Transformation:



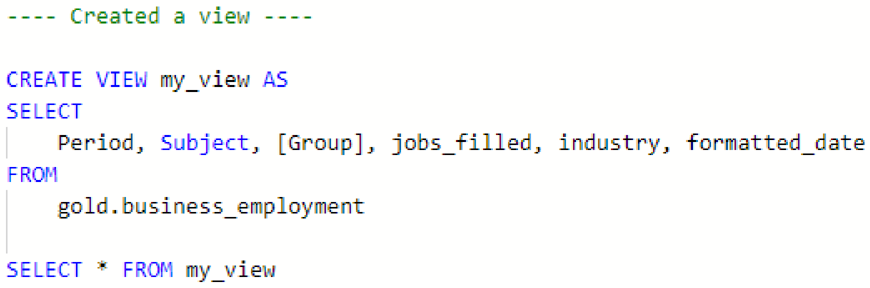
```
----- Created the Gold Layer External Table with updated date format---

CREATE EXTERNAL TABLE gold.business_employment (
    Period FLOAT,
    [Subject] NVARCHAR(100),
    [Group] NVARCHAR(100),
    jobs_filled NVARCHAR(100),
    industry NVARCHAR(100),
    formatted_date  DATE  --  new column here
)
WITH (
    LOCATION = 'Data/Gold/Gold Layer.csv',
    DATA_SOURCE = bus_emp_src,
    FILE_FORMAT = csv_file_format,
    REJECT_VALUE = 10,
    REJECTED_ROW_LOCATION = 'rejections/employment'
    );

Select * FROM gold.business_employment
```



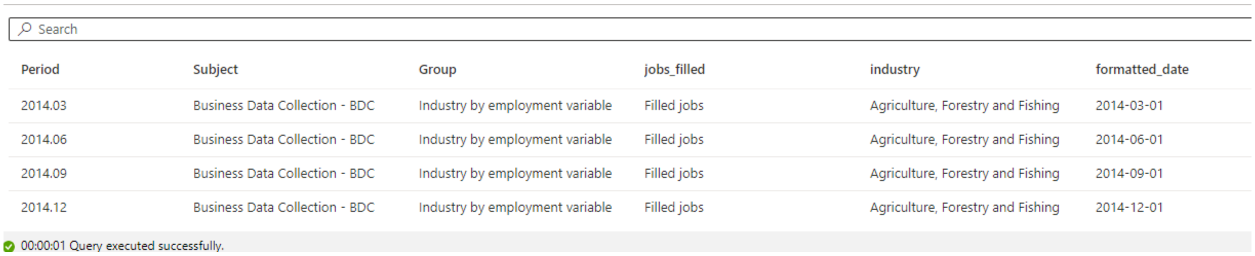
Period	Subject	Group	jobs_filled	industry	formatted_date
2014.03	Business Data Collection - BDC	Industry by employment variable	Filled jobs	Agriculture, Forestry and Fishing	2014-03-01
2014.06	Business Data Collection - BDC	Industry by employment variable	Filled jobs	Agriculture, Forestry and Fishing	2014-06-01
2014.09	Business Data Collection - BDC	Industry by employment variable	Filled jobs	Agriculture, Forestry and Fishing	2014-09-01
2014.12	Business Data Collection - BDC	Industry by employment variable	Filled jobs	Agriculture, Forestry and Fishing	2014-12-01

Created a view for ease of access with all the updates:

The below query of the dataset for key insights such as Total employment growth over the decade.



```
---- Created a view ----

CREATE VIEW my_view AS
SELECT
    Period, Subject, [Group], jobs_filled, industry, formatted_date
FROM
    gold.business_employment

SELECT * FROM my_view
```



Search

Period	Subject	Group	jobs_filled	industry	formatted_date
2014.03	Business Data Collection - BDC	Industry by employment variable	Filled jobs	Agriculture, Forestry and Fishing	2014-03-01
2014.06	Business Data Collection - BDC	Industry by employment variable	Filled jobs	Agriculture, Forestry and Fishing	2014-06-01
2014.09	Business Data Collection - BDC	Industry by employment variable	Filled jobs	Agriculture, Forestry and Fishing	2014-09-01
2014.12	Business Data Collection - BDC	Industry by employment variable	Filled jobs	Agriculture, Forestry and Fishing	2014-12-01

00:00:01 Query executed successfully.

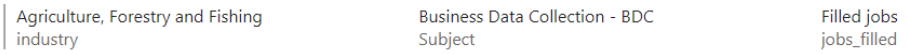
Visualizing Results:

Imported the Data to Power BI Desktop and created a Report indicating the jobs filled per year and per quarter in the last decade:

Pie-Chart: Showing the percentage of people employed over the last ten years.

Table-Chart: Break down of the jobs_filled as per quarter and yearly records and total records.

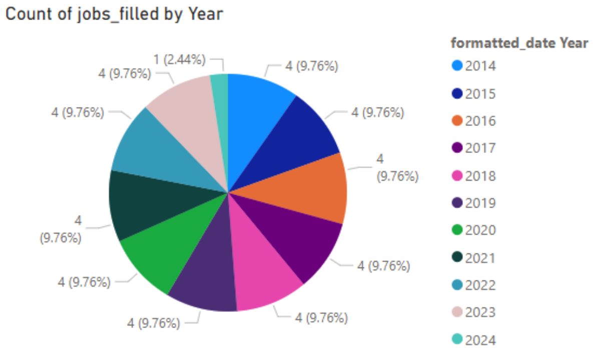
Multiple Card: On the top to display the Industry, Subject and jobs_filled.

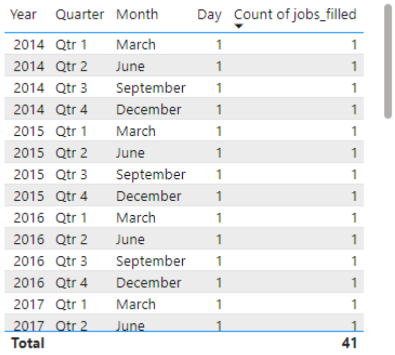


Agriculture, Forestry and Fishing
industry

Business Data Collection - BDC
Subject

Filled jobs
jobs_filled





Year	Quarter	Month	Day	Count of jobs_filled
2014	Qtr 1	March	1	1
2014	Qtr 2	June	1	1
2014	Qtr 3	September	1	1
2014	Qtr 4	December	1	1
2015	Qtr 1	March	1	1
2015	Qtr 2	June	1	1
2015	Qtr 3	September	1	1
2015	Qtr 4	December	1	1
2016	Qtr 1	March	1	1
2016	Qtr 2	June	1	1
2016	Qtr 3	September	1	1
2016	Qtr 4	December	1	1
2017	Qtr 1	March	1	1
2017	Qtr 2	June	1	1
Total				**41**

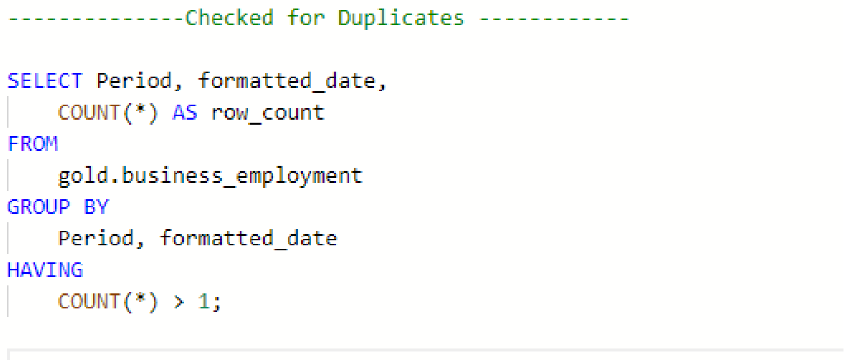
Insights:

After gathering the data and visualizing trends, document key insights such as:

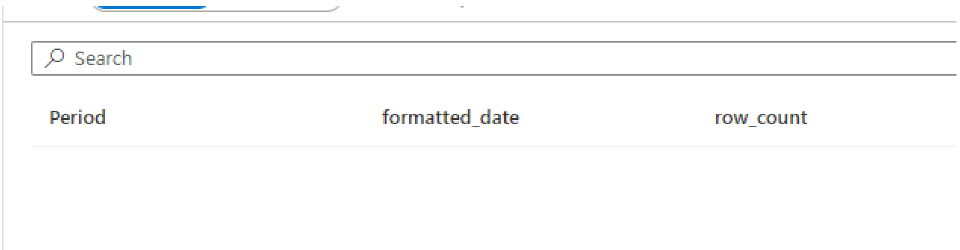
- Total employment over the decade.
- Year-on-year employment in the industry of Agriculture, Forest and Fishing.

For the given three tables Bronze, Silver and Gold we had 'Period' as a matching column.

Checked for any duplicates for Column: In order to select the key



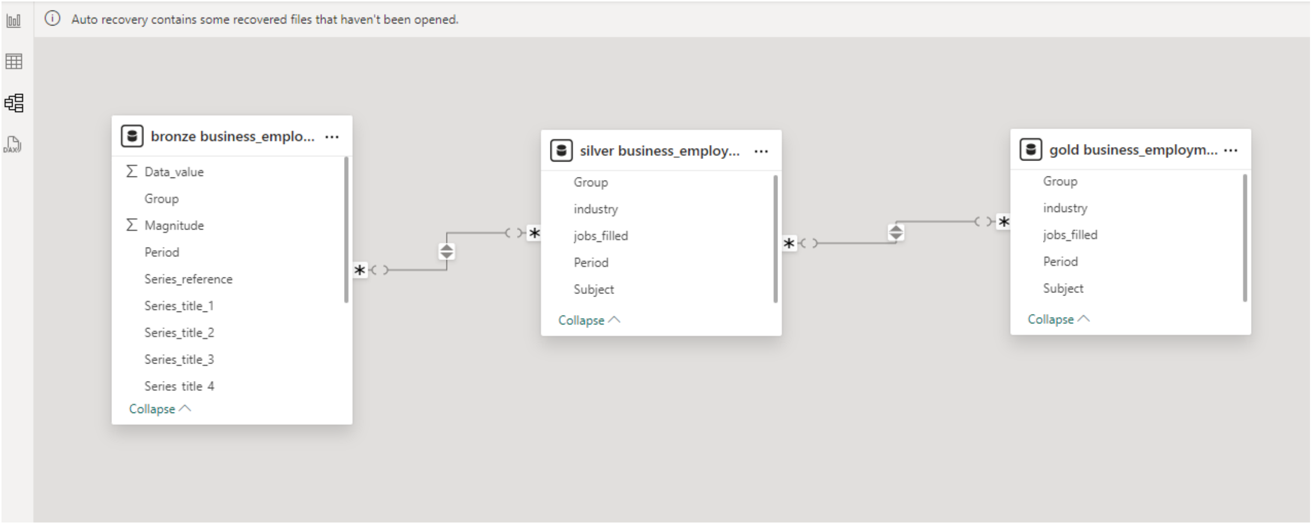
```
--------------Checked for Duplicates ------------

SELECT Period, formatted_date,
    COUNT(*) AS row_count
FROM
    gold.business_employment
GROUP BY
    Period, formatted_date
HAVING
    COUNT(*) > 1;
```



Search

Period	formatted_date	row_count

Created a Data Model as below:



Auto recovery contains some recovered files that haven't been opened.

Error Log:

Error Faced	Work around
Spark Pool Memory Error	Created with different number of nodes, raised a ticket to Microsoft and later after a few days started working
Dedicated SQL Pool ingestion Error	Autoresolved after a few attempts
SQL Alternate Key Error	Given Not Enforced to resolve
Stream Jobs error for output not found	Changed from Managed Identity and given SQL authentication, then connection was made succesfully
Mege Table Query Error	Used Alternative Update along with joins to compare the staging and dimension table
CosmosDB	Synapselinking error, was resolved after going into setting and enabling synapse linking and recreating linked service

Comparative Document on Built Solutions

Comparative solutions built across real-time, near real-time, and batch processing scenarios, focusing on key Azure technologies like Event Hub, Stream Analytics, Cosmos DB, Synapse Pools (Spark & Dedicated SQL) and Serverless SQL Pool. The goal is to compare how data ingestion, processing, and storage are handled.

Technologies used for Data Processing:

- **Real-Time Processing:** Event Hub, Stream Analytics, Cosmos DB/Dedicated SQL Pool.
- **Near Real-Time Processing:** Event Hub, Stream Analytics, Synapse Spark Pool, Cosmos DB**.**
- **Batch Processing:** Synapse Dedicated SQL Pool, Synapse Spark Pool, Data Lake, Serverless SQL Pool

1. Real-Time Processing Solution: Real-time data ingestion and analytics for Event Hubs, where data is streamed continuously and requires immediate insights.

Steps	Technologies Used
Data Ingestion	**Azure Event Hub** streams real-time data from a device.
Data Processing	**Azure Stream Analytics** processes the streaming data in real time using SQL-like queries.
Data Storage	The processed data is stored in **Dedicated SQL Pool** for as the output sink in Stream Analytics to store processed data for real-time querying.

2. Near Real-Time Processing Solution: Data provided from a place where slight delays in processing (seconds to minutes) are acceptable for real-time user insights.

Steps	Technologies Used
Data Ingestion	Sample streams real-time data from **Event Hubs.**
Data Processing	**Cosmos DB** connected to Azure Synapse Pool using Synapse Link and used **Spark Pool** or **Synapse Dedicated SQL/Serverless Pool** for more complex data transformations.
Data Storage	**Synapse Dedicated SQL/Serverless SQL** Pool for reporting and analysis.

3. Batch Processing Solution: Enterprise dataset from a case where large datasets are ingested, processed, and analyzed periodically (e.g., daily reports).

Steps	Technologies Used
Data Ingestion	Data is ingested in batches from **Azure Data Lake** or external databases
Data Processing	**Azure Synapse Dedicated SQL Pool or Serverless Pool and Spark Pool** for large-scale processing and transformations.

Data Storage	Processed data is stored in **Synapse SQL Pool** for reporting, or in Data Lake for further analysis.

Differences between Real-Time, Near Real-Time, and Batch Processing:

Aspect	**Real-Time Processing**	**Near Realtime Processing**	**Batch Processing**
Definition	Processing data immediately as it arrives (milliseconds/seconds).	Processing data with minimal delay (seconds/minutes).	Processing large datasets after accumulation over time (hours/days).
Azure Services	Event Hub, Stream Analytics, Cosmos DB	Event Hub, Stream Analytics, Synapse Spark Pool, Cosmos DB	Synapse Dedicated SQL Pool, Synapse Spark Pool, Serverless SQL Pool
Use Cases	IoT, financial transactions, live data streaming	Social media analytics, log processing, monitoring systems	Data warehousing, ETL pipelines, periodic reports
Data Ingestion	Continuous streaming from Event Hub	Streaming with slightly delayed processing	Batch data ingestion from data lakes, databases
Data Processing	Stream Analytics, real-time transformations	Stream Analytics, Spark Pool for slightly delayed processing	Spark Pool, SQL Pool for periodic data transformations
Storage	Cosmos DB, Synapse SQL Pools	Cosmos DB, Synapse SQL Pools	Synapse Dedicated SQL Pool, Data Lake

Conclusion: This project successfully demonstrates how **Azure's data services** can be leveraged to solve complex data challenges involving real-time data ingestion, processing, and analysis. By using a combination of **Cosmos DB**, **SQL Pool**, **Event Hub**, **Stream Analytics**, and **Spark Pool**, we built a robust data architecture capable of addressing the needs of modern data-driven applications, including real-time analytics and large-scale batch processing.

Moreover, the comparative approach enables us to understand the key differences between real-time, near real-time, and batch processing solutions using Azure services. By revisiting previous tasks and implementing Azure technologies, participants can develop robust data pipelines that meet various business needs. The final comparative document will serve as a reference for how Azure services can be used in different processing scenarios.