Project – Designing Different Data Solutions

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

Project outline:

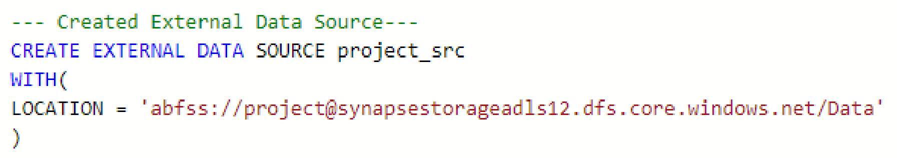
- Technologies used -
    - Azure Cosmos DB – NoSQL database for real-time data storage.
    - Azure Dedicated SQL Pool – Data warehousing for structured data.
    - Azure Synapse Spark Pool – Large-scale data processing with Spark.
    - Azure Event Hub – Real-time data ingestion.
    - Azure Stream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

- 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
- Conclusion

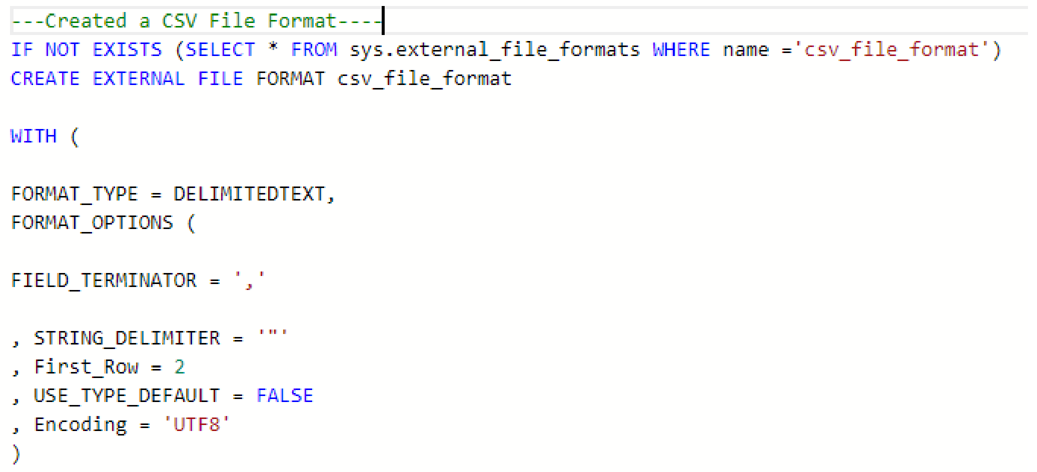
Dedicated SQL Pool: Perform large-scale batch processing and analytics on structured data.

- 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'

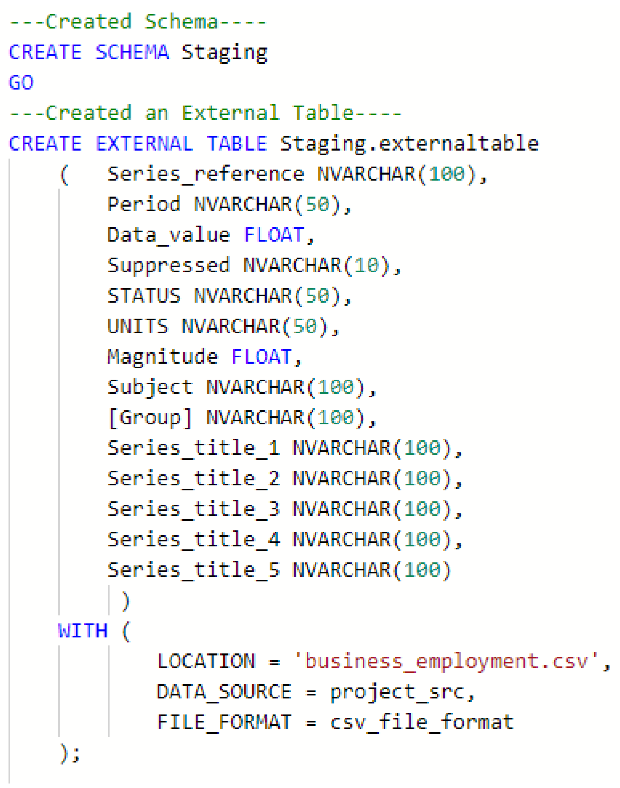


```
--- Created External Data Source---
CREATE EXTERNAL DATA SOURCE project_src
WITH(
LOCATION = 'abfss://project@synapsestorageadls12.dfs.core.windows.net/Data'
)
```



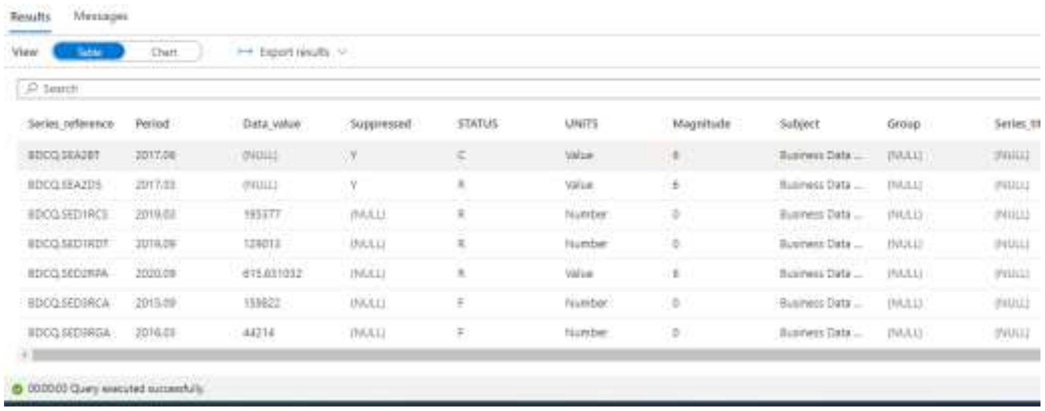
```
---Created a CSV File Format----
IF NOT EXISTS (SELECT * FROM sys.external_file_formats WHERE name ='csv_file_format')
CREATE EXTERNAL FILE FORMAT csv_file_format

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
    );
```

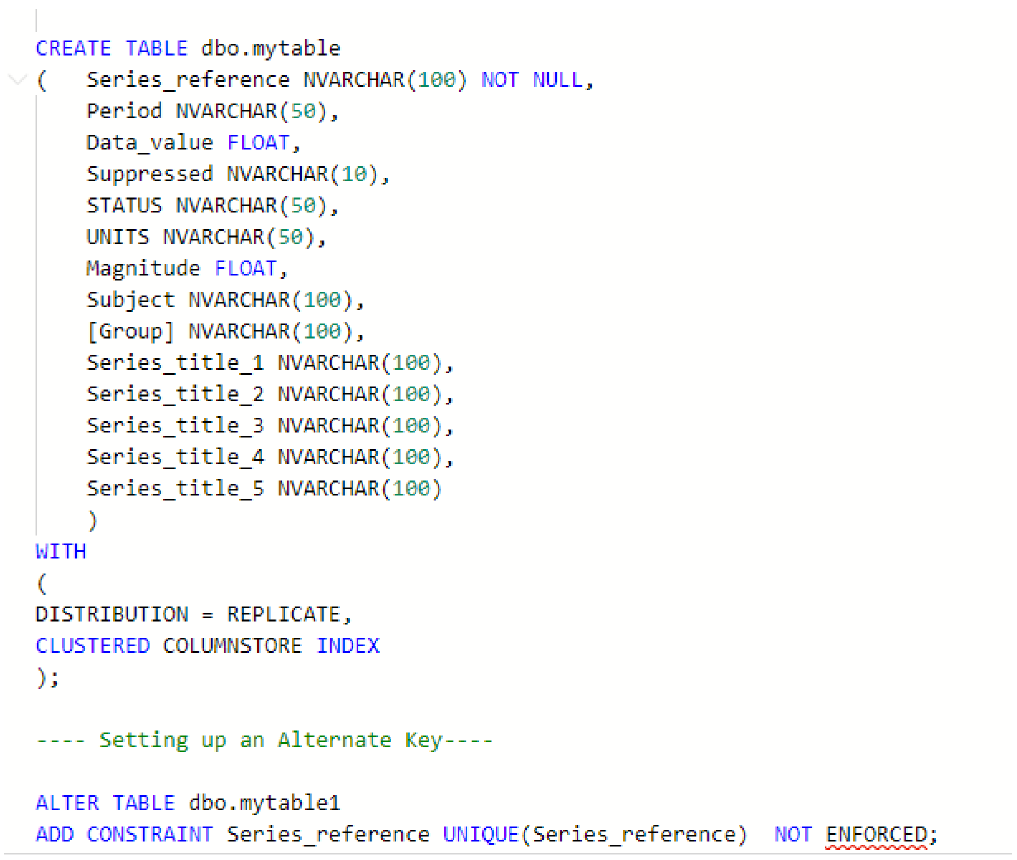
This our external table view after running Select* FROM External Table



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	185377	(NULL)	R	Number	0	Business Data ...	(NULL)	(NULL)
BDCQ.SED1RDT	2019.06	128013	(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)

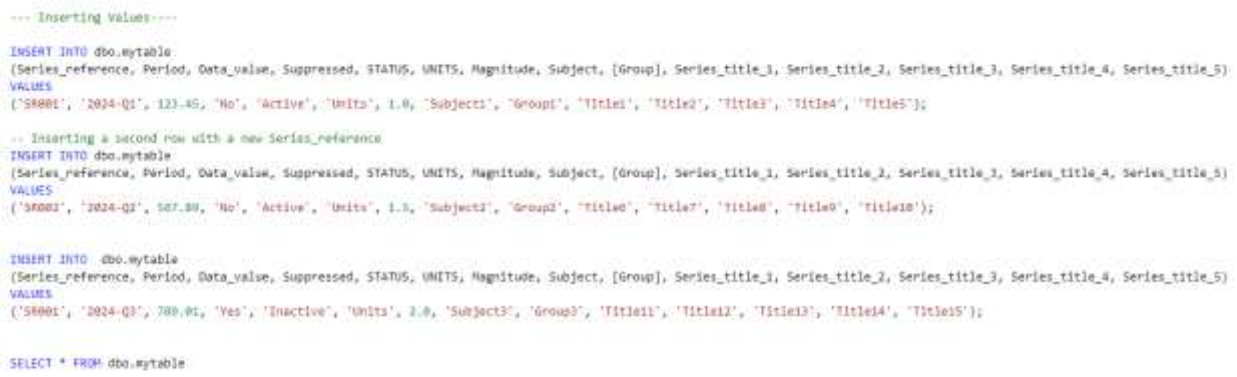
00:00:03 Query executed successfully

Created a table ‘mytable’ and added alternate key



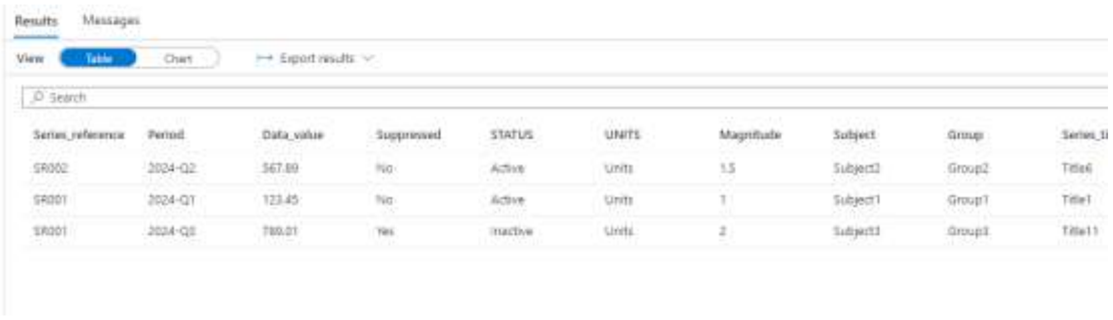
```
CREATE TABLE dbo.mytable
(   Series_reference NVARCHAR(100) NOT NULL,
    Period NVARCHAR(50),
    Data_value FLOAT,
    Suppressed NVARCHAR(10),
    STATUS NVARCHAR(50),
    UNITS NVARCHAR(50),
    Magnitude FLOAT,
    Subject NVARCHAR(100),
    [Group] NVARCHAR(100),
    Series_title_1 NVARCHAR(100),
    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:



```
---- 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…



Results Messages

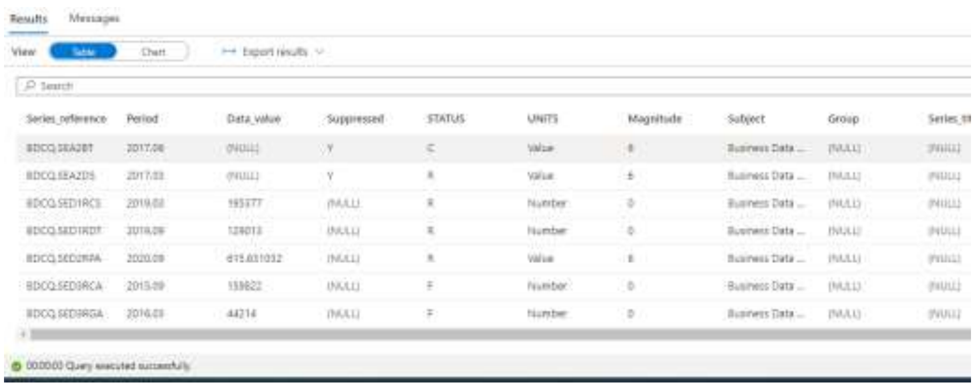
View Table Chart Export results

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.

```
--------------------------------------------
--- 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;
```


Results Messages

View Table Chart Export results

Search

Series_reference	Period	Data_value	Suppressed	STATUS	UNITS	Magnitude	Subject	Group	Series_tit
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	128013	(NULL)	R	Number	0	Business Data ...	(NULL)	(NULL)
BDCQ.SED2RPA	2020.09	615.831032	(NULL)	R	Value	6	Business Data ...	(NULL)	(NULL)
BDCQ.SED3RCA	2015.09	153822	(NULL)	F	Number	0	Business Data ...	(NULL)	(NULL)
BDCQ.SED3RGA	2016.03	44214	(NULL)	F	Number	0	Business Data ...	(NULL)	(NULL)

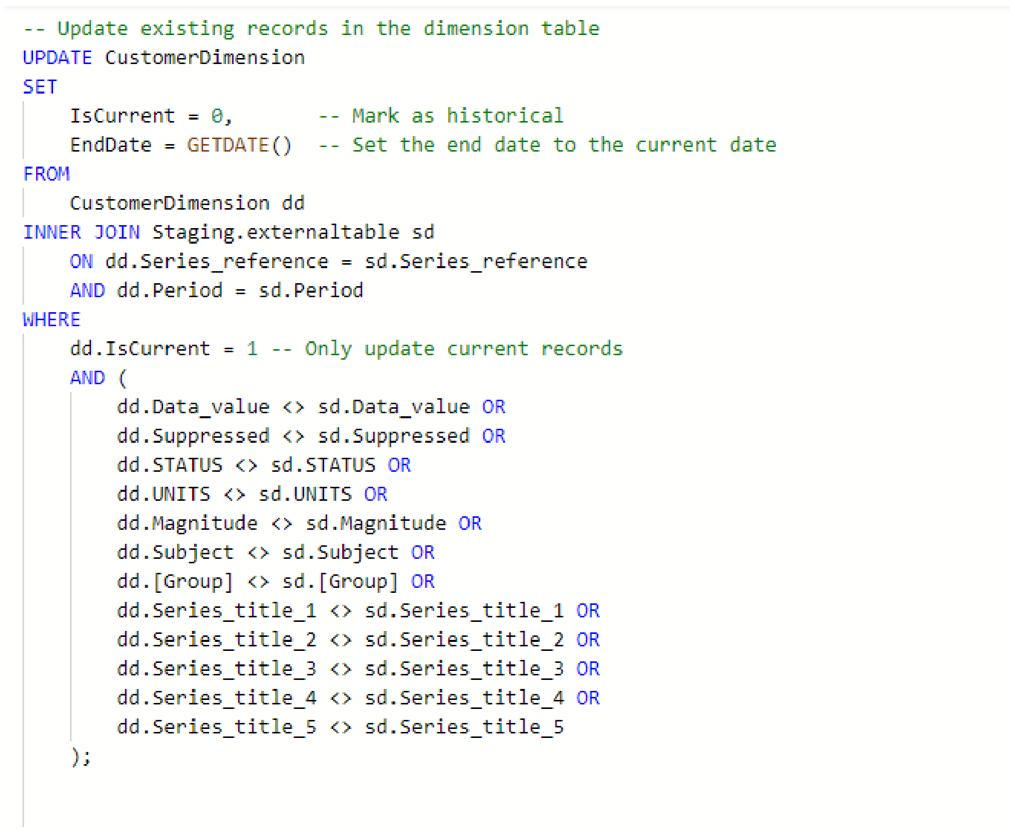
00:00:03 Query executed successfully

Now we are creating a Dimension table as below:

```sql
--- 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;
```

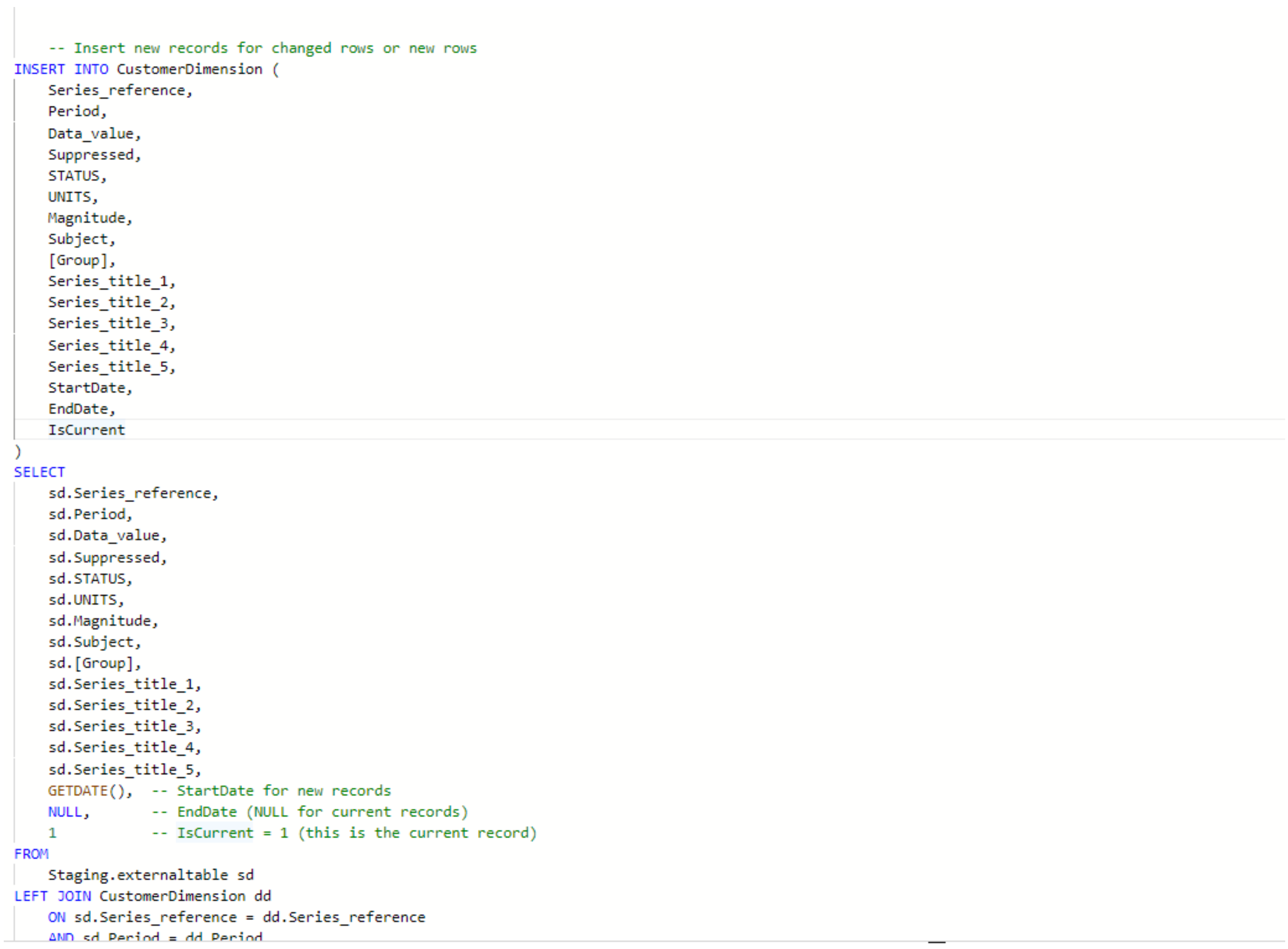
In order to perform Type 2 SCD,
We will compare the data in the staging table with the dimension table,

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

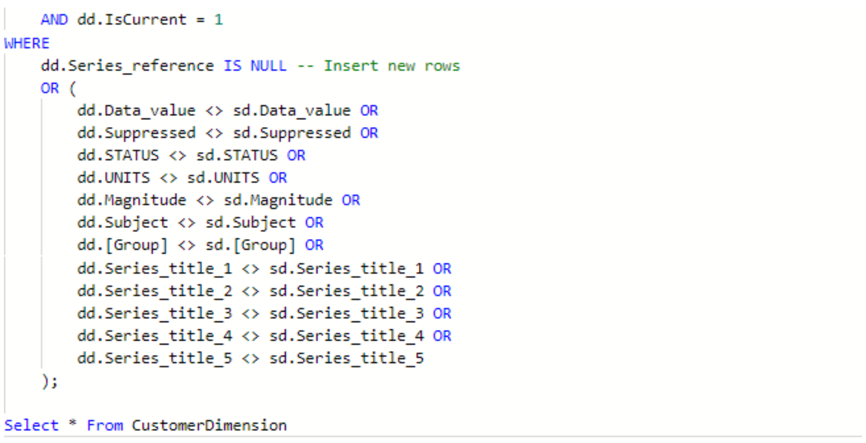


```sql
-- Update existing records in the dimension table
UPDATE CustomerDimension
SET
    IsCurrent = 0,       -- Mark as historical
    EndDate = GETDATE()  -- Set the end date to the current date
FROM
    CustomerDimension dd
INNER JOIN Staging.externaltable sd
    ON dd.Series_reference = sd.Series_reference
    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).

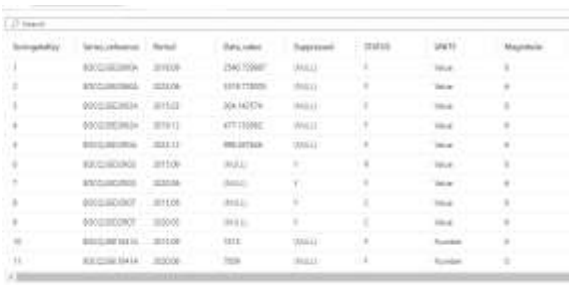


```sql
    -- Insert new records for changed rows or new rows
INSERT INTO CustomerDimension (
    Series_reference,
    Period,
    Data_value,
    Suppressed,
    STATUS,
    UNITS,
    Magnitude,
    Subject,
    [Group],
    Series_title_1,
    Series_title_2,
    Series_title_3,
    Series_title_4,
    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
```

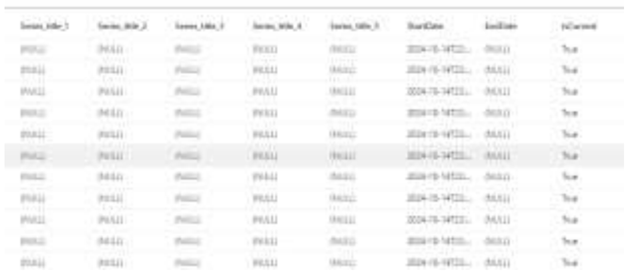


```sql
    AND dd.IsCurrent = 1
WHERE
    dd.Series_reference IS NULL -- Insert new rows
    OR (
        dd.Data_value <> sd.Data_value OR
        dd.Suppressed <> sd.Suppressed OR
        dd.STATUS <> sd.STATUS OR
        dd.UNITS <> sd.UNITS OR
        dd.Magnitude <> sd.Magnitude OR
        dd.Subject <> 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:



SurrogateKey	Series_reference	Period	Data_value	Suppressed	STATUS	UNITS	Magnitude
1	[illegible]	[illegible]	[illegible]	(NULL)	F	[illegible]	[illegible]
2	[illegible]	[illegible]	[illegible]	(NULL)	F	[illegible]	[illegible]
3	[illegible]	[illegible]	[illegible]	(NULL)	F	[illegible]	[illegible]
4	[illegible]	[illegible]	[illegible]	(NULL)	F	[illegible]	[illegible]
5	[illegible]	[illegible]	[illegible]	(NULL)	F	[illegible]	[illegible]
6	[illegible]	[illegible]	(NULL)	Y	[illegible]	[illegible]	[illegible]
7	[illegible]	[illegible]	(NULL)	Y	[illegible]	[illegible]	[illegible]
8	[illegible]	[illegible]	(NULL)	Y	[illegible]	[illegible]	[illegible]
9	[illegible]	[illegible]	(NULL)	Y	[illegible]	[illegible]	[illegible]
10	[illegible]	[illegible]	[illegible]	(NULL)	F	Number	[illegible]
11	[illegible]	[illegible]	[illegible]	(NULL)	F	Number	[illegible]

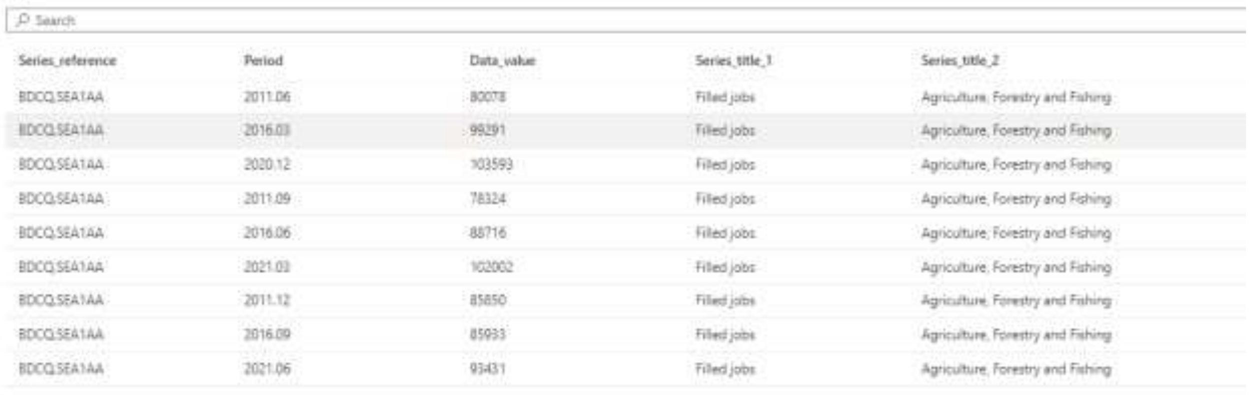


Series_title_1	Series_title_2	Series_title_3	Series_title_4	Series_title_5	StartDate	EndDate	IsCurrent
(NULL)	(NULL)	(NULL)	(NULL)	(NULL)	[illegible]	(NULL)	True
(NULL)	(NULL)	(NULL)	(NULL)	(NULL)	[illegible]	(NULL)	True
(NULL)	(NULL)	(NULL)	(NULL)	(NULL)	[illegible]	(NULL)	True
(NULL)	(NULL)	(NULL)	(NULL)	(NULL)	[illegible]	(NULL)	True
(NULL)	(NULL)	(NULL)	(NULL)	(NULL)	[illegible]	(NULL)	True
(NULL)	(NULL)	(NULL)	(NULL)	(NULL)	[illegible]	(NULL)	True
(NULL)	(NULL)	(NULL)	(NULL)	(NULL)	[illegible]	(NULL)	True
(NULL)	(NULL)	(NULL)	(NULL)	(NULL)	[illegible]	(NULL)	True
(NULL)	(NULL)	(NULL)	(NULL)	(NULL)	[illegible]	(NULL)	True
(NULL)	(NULL)	(NULL)	(NULL)	(NULL)	[illegible]	(NULL)	True
(NULL)	(NULL)	(NULL)	(NULL)	(NULL)	[illegible]	(NULL)	True

Did some transformation using where clause and selected few columns only

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

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



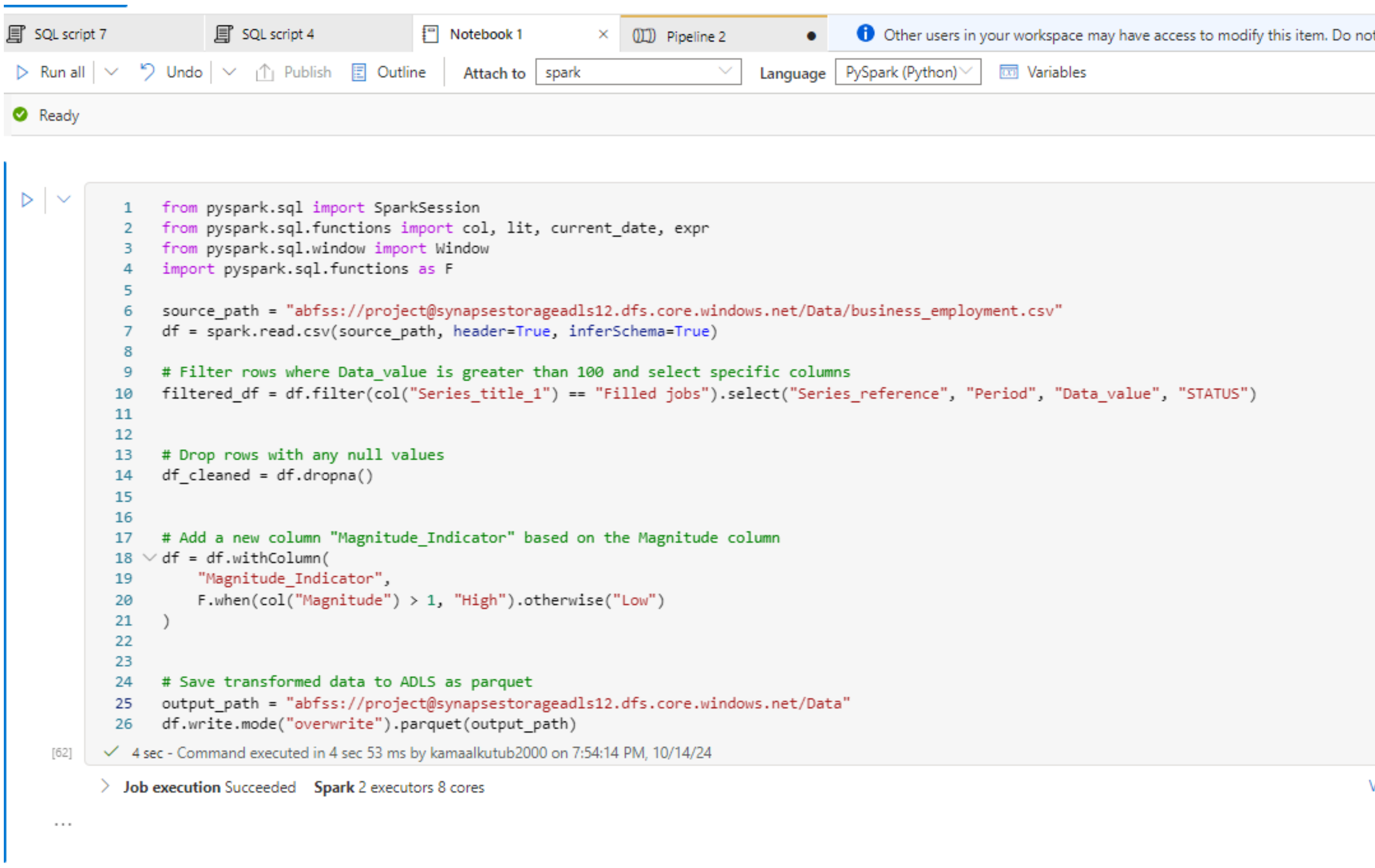
Series_reference	Period	Data_value	Series_title_1	Series_title_2
BDCQ.SEA1AA	2011.06	80078	Filled jobs	Agriculture, Forestry and Fishing
BDCQ.SEA1AA	2016.03	99291	Filled jobs	Agriculture, Forestry and Fishing
BDCQ.SEA1AA	2020.12	103593	Filled jobs	Agriculture, Forestry and Fishing
BDCQ.SEA1AA	2011.09	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

<u>Apache Spark Pool:</u> 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

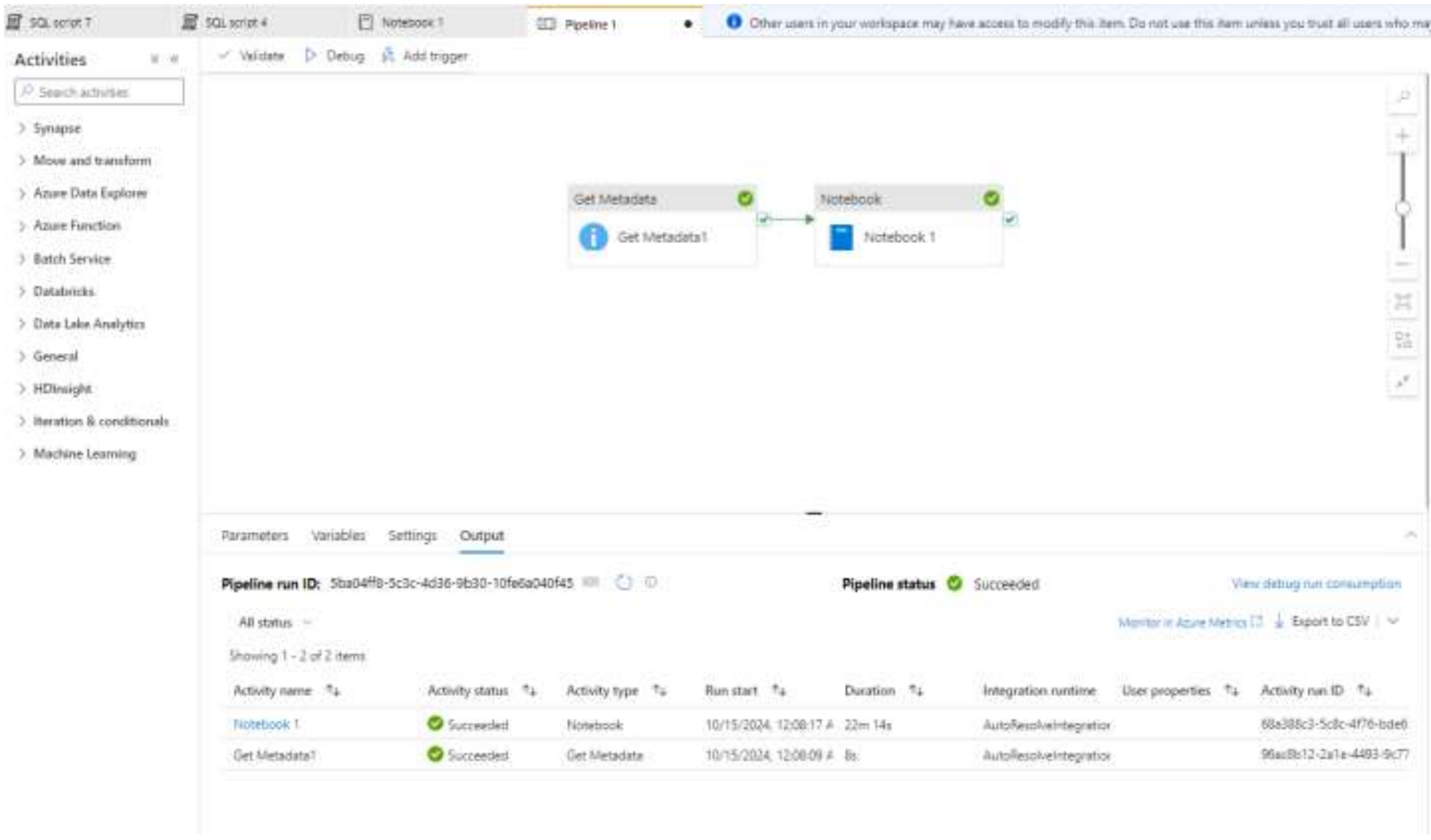

SQL script 7
SQL script 4
Notebook 1
Pipeline 2
Other users in your workspace may have access to modify this item. Do not
Run all
Undo
Publish
Outline
Attach to spark
Language PySpark (Python)
Variables
Ready

```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

View debug run consumption

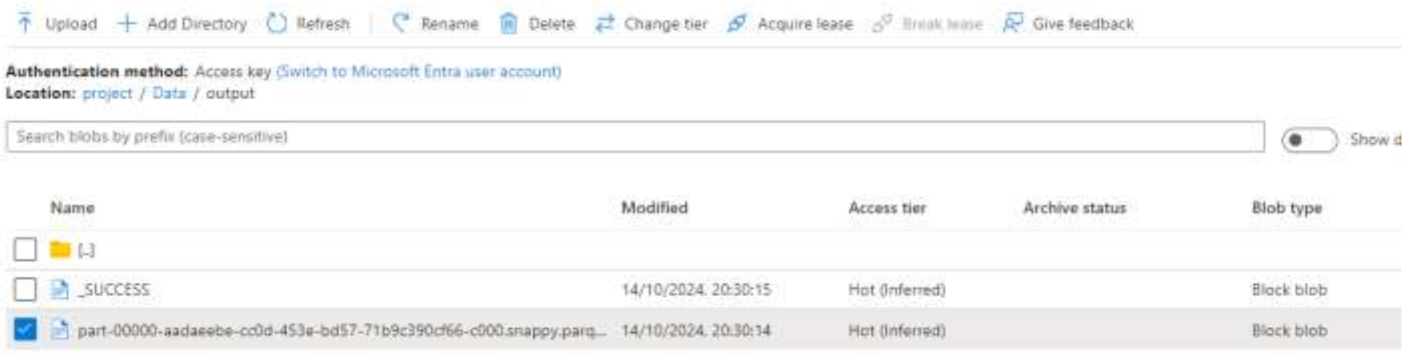
All status

Monitor in Azure Metrics Export to CSV

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

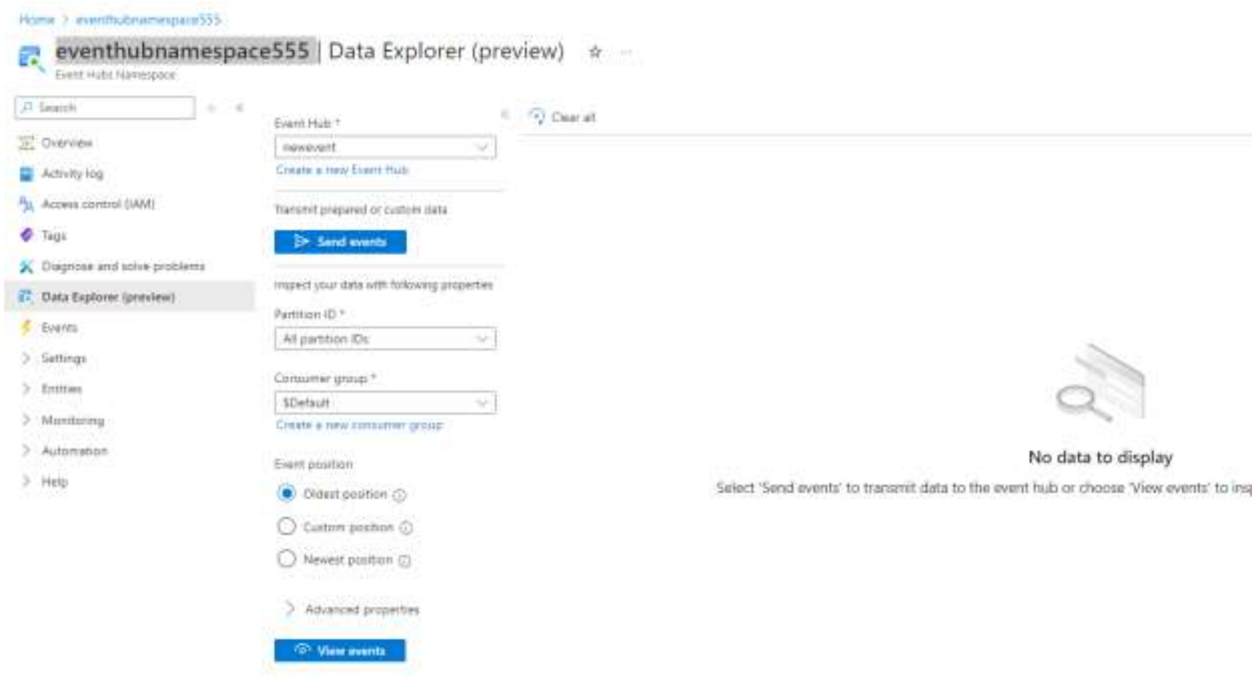
Search blobs by prefix (case-sensitive)

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

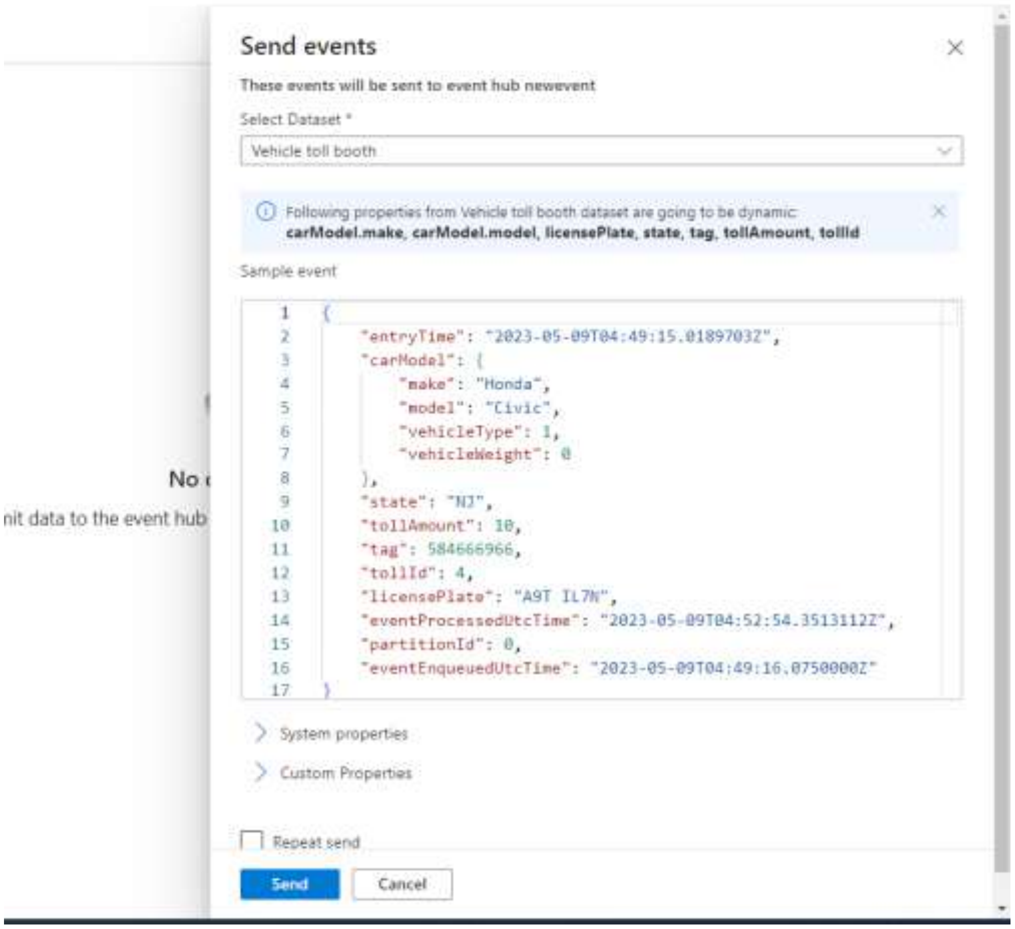
Partition ID *: All partition IDs

Consumer group *: $Default

Event position: [✓ Oldest position] Custom position Newest position

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:



No

nit data to the event hub

Send events

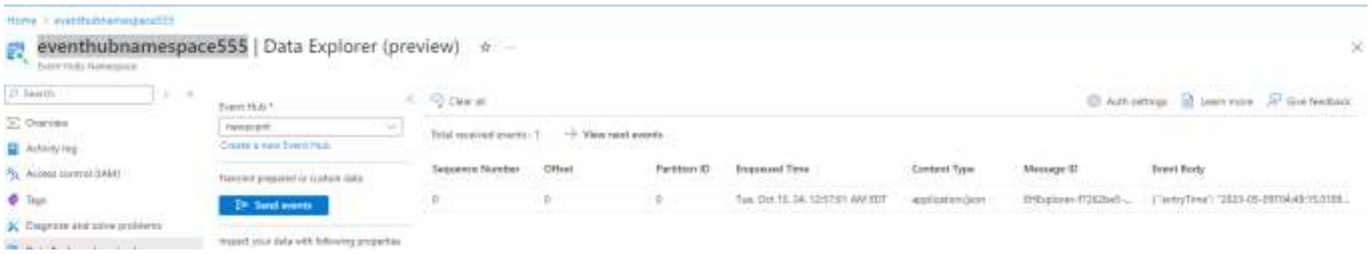
These events will be sent to event hub newevent

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"
}
```



Home > eventhubnamespace555

eventhubnamespace555 | Data Explorer (preview)

Event Hubs Namespace

Event Hub *: newevent

Create a new Event Hub

Send prepared or custom data

Inspect your data with following properties

Total received events : 1 → View next events

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-f7262be0-...	{"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.

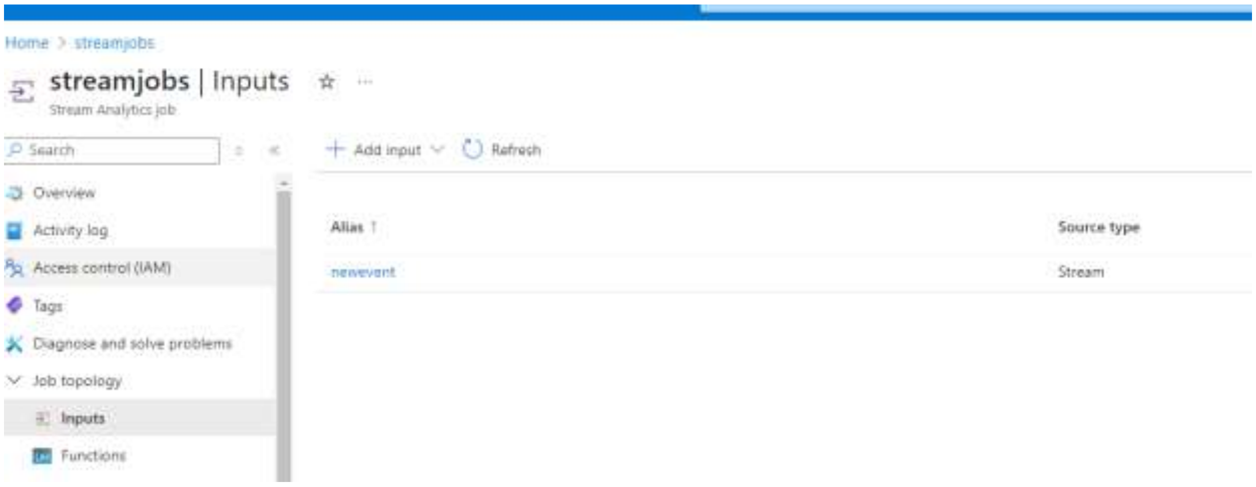
Created a stream job for the stream analytics real time data transformation

Created the below for stream jobs:

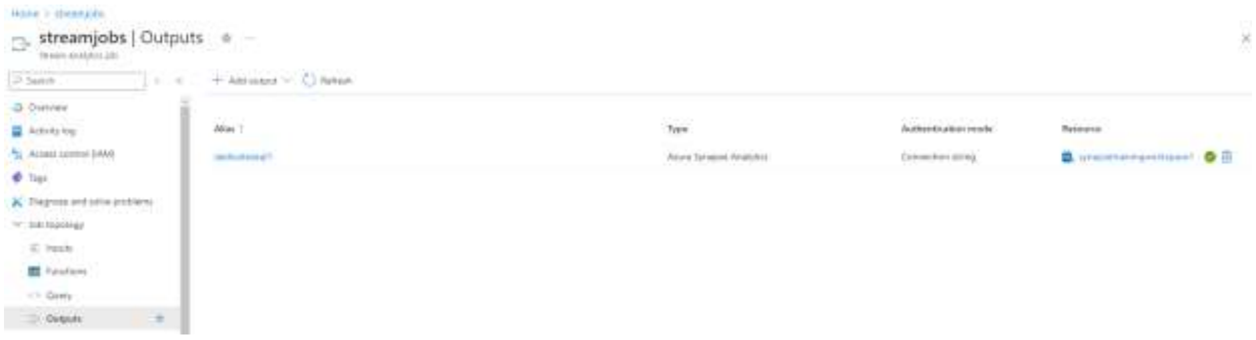
Input – to fetch from our Event Hubs created event

Output – To our dedicated sql Pool

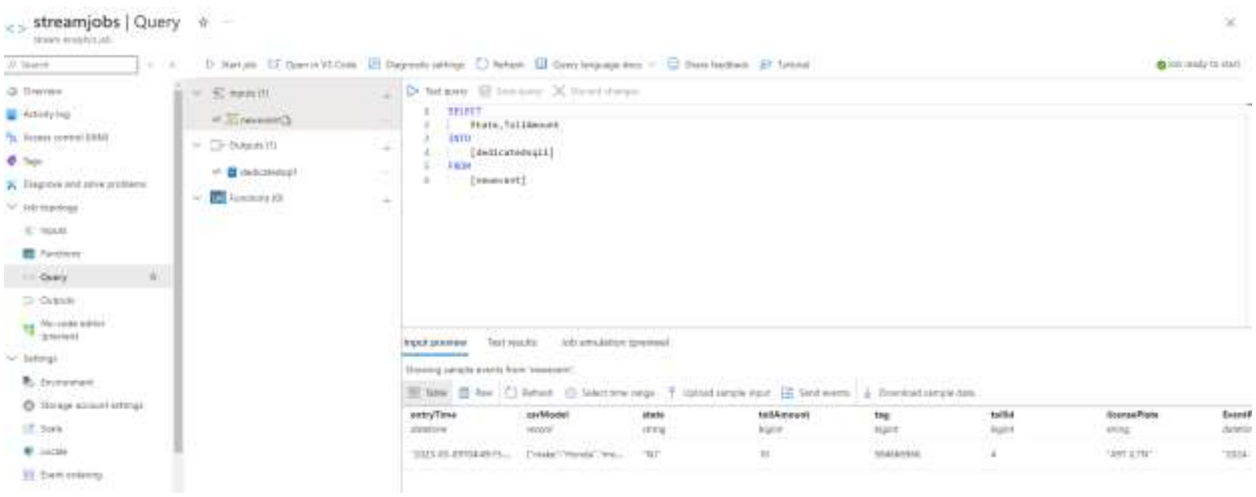
Query – Gave the query to run the stream jobs for gathering the data only for two columns.



Alias	Source type
newevent	Stream



Alias	Type	Authentication mode	Resource
[illegible]	Azure Synapse Analytics	Connection string	[illegible]

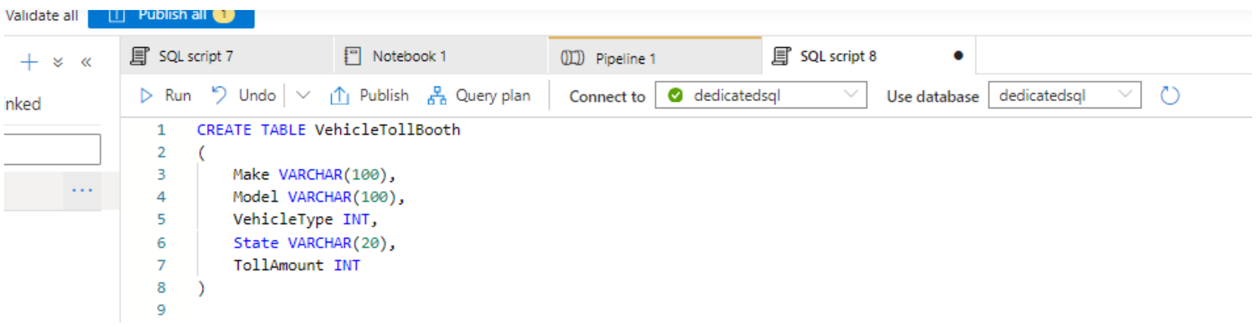


streamjobs | Query

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

entryTime	carModel	state	tollAmount	tag	tollId	licensePlate	EventP
datetime	record	string	bigint	bigint	bigint	string	[illegible]
"2025-05-03T04:49:15...	{"make":"Honda","mo...	"NJ"	10	[illegible]	4	[illegible]	[illegible]

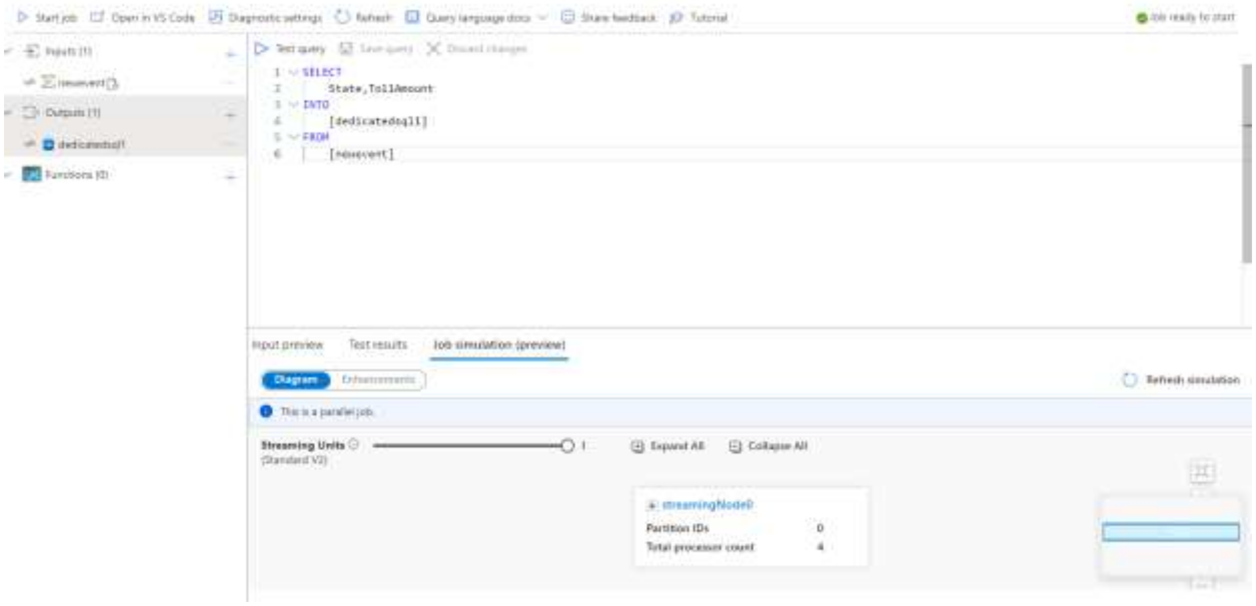
Also created a table 'vehicletollbooth' in dedicated pool:



Connect to: dedicatedsql — Use database: dedicatedsql

```
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

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

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.