**Process to Export tables from an external source and store those into adls gen2 linked to our Azure synapse**

**Steps:**

* At first step, we have to create a linking for external table in our Azure synapse.
* At second step we have to create CETAS in serverless SQL pool.
* At the end we would have our tables exported from external source and being stored in Adls Gen 2 linked to our serverless in our desired Azure synapse.

**Description:**

1. **External tables in dedicated SQL pool and serverless SQL pool**

We can create external tables in Synapse SQL pools via the following steps:

1. CREATE EXTERNAL DATA SOURCE to reference an external Azure storage and specify the credential that should be used to access the storage.
2. CREATE EXTERNAL FILE FORMAT to describe format of CSV or Parquet files.
3. CREATE EXTERNAL TABLE on top of the files placed on the data source with the same file format.

**1)Syntax for CREATE EXTERNAL DATA SOURCE**

CREATE EXTERNAL DATA SOURCE <data\_source\_name>

WITH

( LOCATION = '<prefix>://<path>'

[, CREDENTIAL = <database scoped credential> ]

)

[;]

**Where,**

**data\_source\_name**

Specifies the user-defined name for the data source. The name must be unique within the database.

**Location**

LOCATION = '<prefix>://<path>' - Provides the connectivity protocol and path to the external data source.

**Credential**

CREDENTIAL = <database scoped credential> is optional credential that will be used to authenticate on Azure storage.

**Example for CREATE EXTERNAL DATA SOURCE**

CREATE DATABASE SCOPED CREDENTIAL [sqlondemand]

WITH IDENTITY='SHARED ACCESS SIGNATURE',

SECRET = 'sv=2018-03-28&ss=bf&srt=sco&sp=rl&st=2019-10-14T12%3A10%3A25Z&se=2061-12-31T12%3A10%3A00Z&sig=KlSU2ullCscyTS0An0nozEpo4tO5JAgGBvw%2FJX2lguw%3D'

GO

CREATE EXTERNAL DATA SOURCE SqlOnDemandDemo WITH (

LOCATION = 'https://sqlondemandstorage.blob.core.windows.net',

CREDENTIAL = sqlondemand

);

**2)Syntax for CREATE EXTERNAL FILE FORMAT**

**2.1)Create an external file format for PARQUET files**.

CREATE EXTERNAL FILE FORMAT file\_format\_name

WITH (

FORMAT\_TYPE = PARQUET

[ , DATA\_COMPRESSION = {

'org.apache.hadoop.io.compress.SnappyCodec'

| 'org.apache.hadoop.io.compress.GzipCodec' }

]);

**2.2)Create an external file format for DELIMITED TEXT files**

CREATE EXTERNAL FILE FORMAT file\_format\_name

WITH (

FORMAT\_TYPE = DELIMITEDTEXT

[ , DATA\_COMPRESSION = 'org.apache.hadoop.io.compress.GzipCodec' ]

[ , FORMAT\_OPTIONS ( <format\_options> [ ,...n ] ) ]

);

<format\_options> ::=

{

FIELD\_TERMINATOR = field\_terminator

| STRING\_DELIMITER = string\_delimiter

| FIRST\_ROW = integer

| USE\_TYPE\_DEFAULT = { TRUE | FALSE }

| ENCODING = {'UTF8' | 'UTF16'}

| PARSER\_VERSION = {'parser\_version'}

}

**Where,**

**file\_format\_name-** Specifies a name for the external file format.

**FORMAT\_TYPE = [ PARQUET | DELIMITEDTEXT]-** Specifies the format of the external data.

* PARQUET - Specifies a Parquet format.
* DELIMITEDTEXT - Specifies a text format with column delimiters, also called field terminators.

**FIELD\_TERMINATOR = *field\_terminator* -** Applies only to delimited text files. The field terminator specifies one or more characters that mark the end of each field (column) in the text-delimited file. The default is the pipe character (ꞌ|ꞌ).

**STRING\_DELIMITER = *string\_delimiter* -** Specifies the field terminator for data of type string in the text-delimited file. The string delimiter is one or more characters in length and is enclosed with single quotes. The default is the empty string ("").

**FIRST\_ROW = *First\_row\_int* -** Specifies the row number that is read first and applies to all files.

**USE\_TYPE\_DEFAULT = { TRUE | FALSE } -** Specifies how to handle missing values in delimited text files when retrieving data from the text file.

TRUE - If you're retrieving data from the text file, store each missing value by using the default value's data type for the corresponding column in the external table definition

FALSE - Store all missing values as NULL.

**Encoding = {'UTF8' | 'UTF16'} -** Serverless SQL pool can read UTF8 and UTF16 encoded delimited text files.

**DATA\_COMPRESSION = *data\_compression\_method* -** This argument specifies the data compression method for the external data.

**PARSER\_VERSION = 'parser\_version'** Specifies parser version to be used when reading CSV files.

1. **CETAS in serverless SQL pool**

When using serverless SQL pool, CETAS is used to create an external table and export query results to Azure Storage Blob or Azure Data Lake Storage Gen2.

**Syntax**

CREATE EXTERNAL TABLE [ [database\_name . [ schema\_name ] . ] | schema\_name . ] table\_name

WITH (

LOCATION = 'path\_to\_folder',

DATA\_SOURCE = external\_data\_source\_name,

FILE\_FORMAT = external\_file\_format\_name

)

AS <select\_statement>

[;]

<select\_statement> ::=

[ WITH <common\_table\_expression> [ ,...n ] ]

SELECT <select\_criteria>

Ex:

CREATE EXTERNAL TABLE NYCTaxiPassengersCountStats

WITH

( LOCATION - 'synapsedemo/NYCTAX1/Aggdata/',

DATA\_SOURCE - demoDataSource,

FILE\_FORMAT - ParquetFileFormat

) I

AS

SELECT PassengerCount,

SUM(TripDistanceMiles) as SumTripDistance,

AVG(TripDistanceMiles) as AvgTripDistance

FROM

OPENROWSET

(

BULK 'https://maheeradlsgen2.dfs.core.windows.net/synapsedemo/data/NYCTripSmall.parquet',

FORMAT - 'PARQUET'

As [rows]

WHERE TripDistanceMiles > e AND PassengerCount > e

GROUP BY PassengerCount

GO

**Arguments:**

***[ [ database\_name . [ schema\_name ] . ] | schema\_name . ] table\_name***

The one to three-part name of the table to create. For an external table, serverless SQL pool stores only the table metadata. No actual data is moved or stored in serverless SQL pool.

**LOCATION = *'path\_to\_folder'***

Specifies where to write the results of the SELECT statement on the external data source.

**DATA\_SOURCE = *external\_data\_source\_name***

Specifies the name of the external data source object that contains the location where the external data will be stored.

**FILE\_FORMAT = *external\_file\_format\_name***

Specifies the name of the external file format object that contains the format for the external data file. Only external file formats with FORMAT\_TYPE=PARQUET and FORMAT\_TYPE=DELIMITEDTEXT are currently supported.

**WITH *<common\_table\_expression>***

Specifies a temporary named result set, known as a common table expression **(CTE**

**SELECT <select\_criteria>**

Populates the new table with the results from a SELECT statement. *select\_criteria* is the body of the SELECT statement that determines which data to copy to the new table.

----------------CTAS: Dedicated pool only-----------

CREATE TABLE [dbo].[Customers]

 (

    CustomerID int NOT NULL,

    CustomerName NVARCHAR (50),

    ContactName NVARCHAR (10)

)

WITH

(

    DISTRIBUTION = HASH (CustomerID),

    CLUSTERED COLUMNSTORE INDEX

)

GO

------------------------

INSERT INTO dbo.Customers VALUES (1, 'Maheer', 'dharmesh')

INSERT INTO dbo.Customers VALUES (2, 'Asi', 'suresh')

INSERT INTO dbo.Customers VALUES(3, 'Wafa', 'vinay')

----------------------

CREATE TABLE [dbo].[Orders]

 (

    OrderID int NOT NULL,

    CustomerID int,

    EmployeeID int

)

WITH

(

    DISTRIBUTION = HASH (OrderID),

    CLUSTERED COLUMNSTORE INDEX

)

GO

-------------------

INSERT INTO dbo.Orders VALUES (10308, 2, 2)

INSERT INTO dbo.Orders VALUES (10309, 37, 4)

INSERT INTO dbo.Orders VALUES(10310, 77, 7)

---------

SELECT \* FROM dbo.Customers

SELECT \* FROM dbo.Orders

SELECT \* FROM dbo.employeesNew1

---------------

CREATE TABLE dbo.employeesNew1

WITH

(

    DISTRIBUTION = ROUND\_ROBIN,

    CLUSTERED COLUMNSTORE INDEX

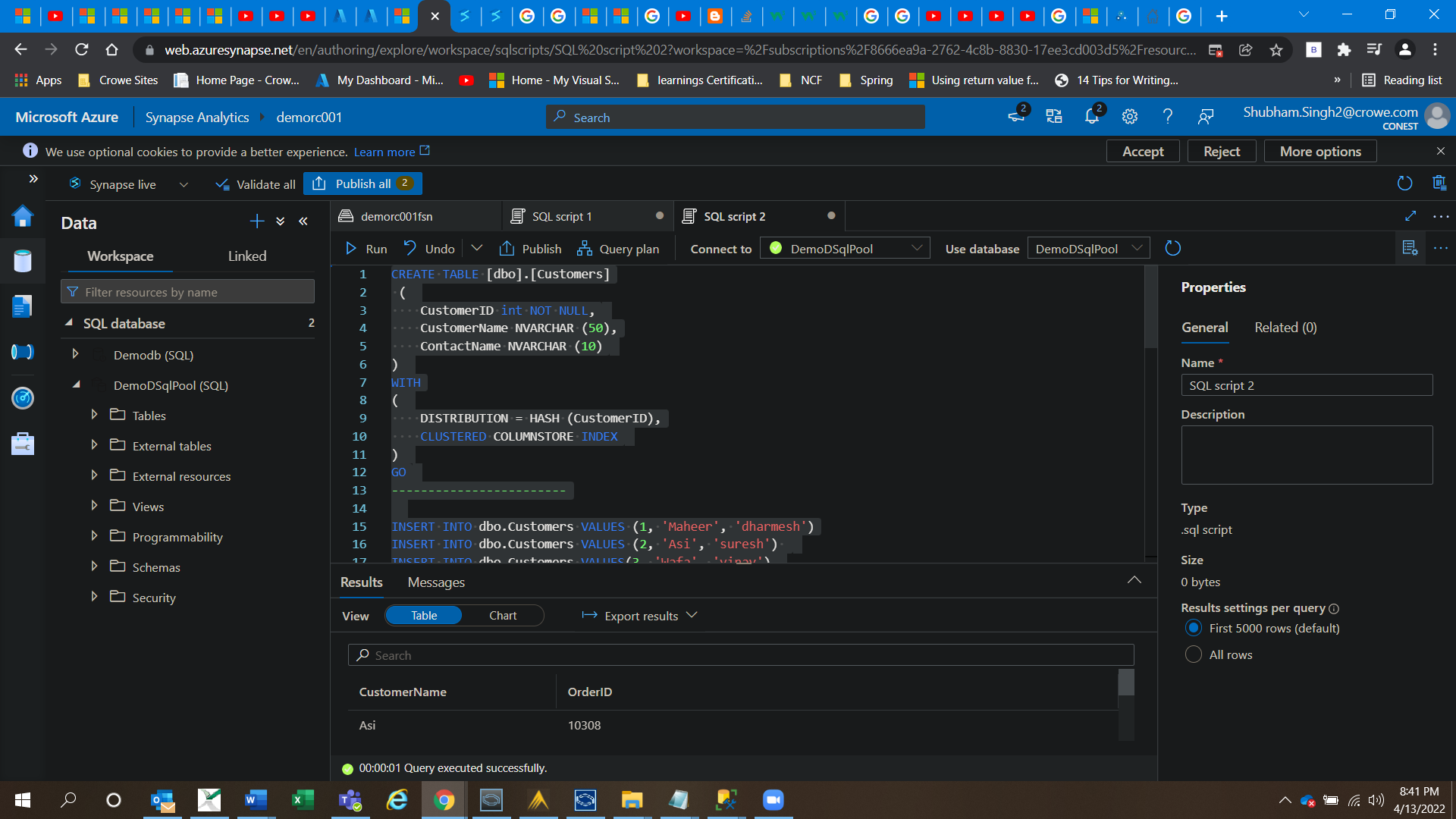
)

AS

SELECT Customers.CustomerName, Orders.OrderID

FROM Customers

FULL OUTER JOIN Orders ON Customers.CustomerID=Orders.CustomerID



Utilize ADLS Gen2 to create EDW tables rather than using temporary tables.

how we can leverage ADLS to create EDW tables.

EDW Tables: These tables are created by joining different base AX tables.