Azure Synapse

Need to have a Data Warehouse – It helps us do analytics on the data that we have. The data is stored in a way where it is made to process high volumes of read requests.

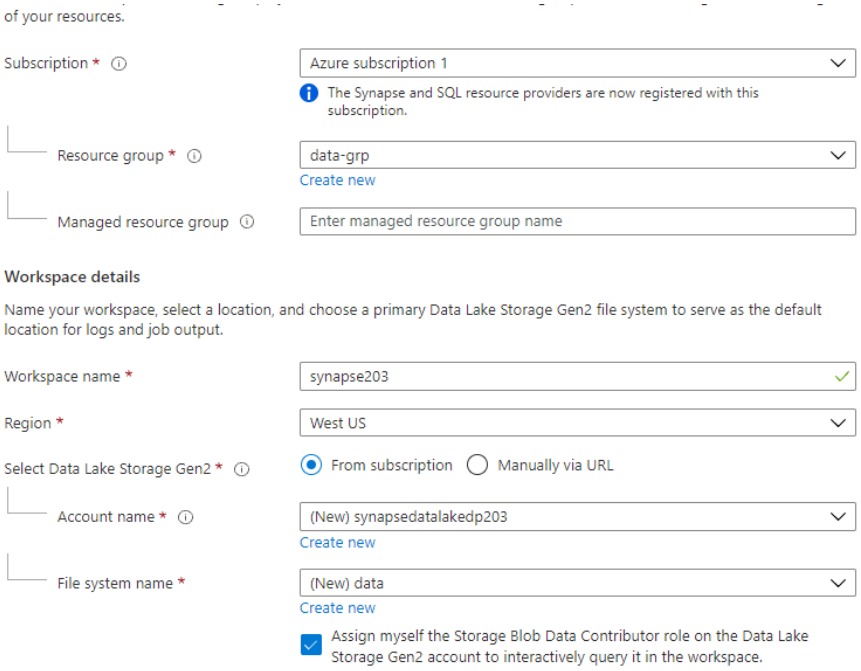
Synapse initially was just a data warehouse but now it is known as Azure Synapse Analytics. Now we can create warehouses with the help of SQL, integrate the data using pipelines and also use data from data lakes.

We can also use Spark for processing and the data and services like Azure monitor and Azure Active Directory with synapse.

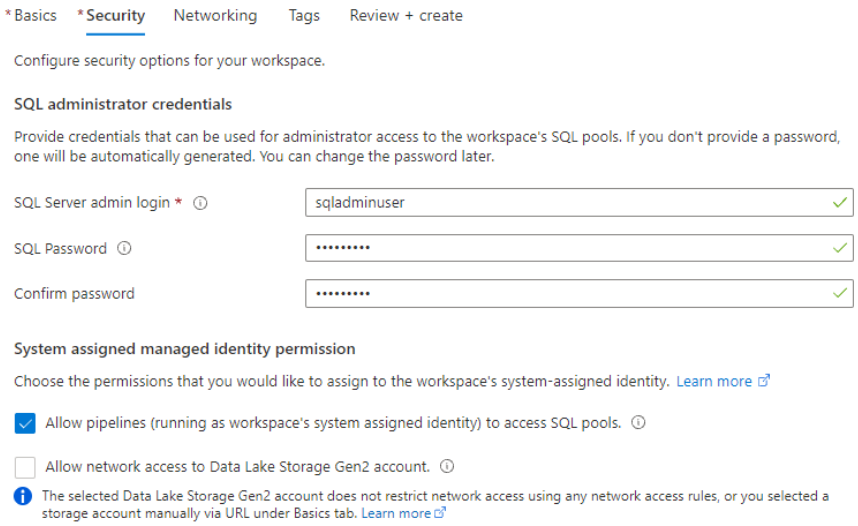
Creating an Azure Synapse Workspace

Go to your home screen > Create a Resource > Search for Azure Synapse Analytics > Create.

Enter : Your subscription, your resource group, *unique* workspace name, region, data lake gen2 details (new or old)



On the next screen give the password for your SQL Administrator Credentials, make sure “allow pipelines” checkbox is ticked.



Leave everything else as is and create.

Synapse Compute Options

There are different compute options – Serverless SQL pool and SQL pool.

Serverless SQL Pool

- You can user this option to perform quick adhoc analysis of data
- Can use T-SQL
- Can only create external tables but cannot persist the data
- Charged based on how much you use the service and how much data your process

SQL Pool

- User to build your warehouse
- Can use T-SQL
- Used if you want to persist the data
- Charged based on the data warehousing units (which includes things like compute, memory, etc.)

External tables - Can be defined in the Serverless pool and the dedicated SQL pool. We use external tables when the table data is lying in an external source, but the table definition is lying in Azure synapse. This is useful when you don't want to load the table on to the server itself.

For example if there are tables that exist on an external source and there's data on the sql server, then to perform a join operation between the two, an external table can be used.

There are a few important checks that need to be done in order to access the external data:

- We first need to have authorization to use the external source of data
- We then need to define the format of the external file that we want to use as an external table
- Finally, create the external table

Using External Tables

Open up your synapse dashboard and click on Open Synapse Studio. In the synapse studio you can use SQL commands against your Serverless SQL pool as well as dedicated SQL pool. Can create pipelines to integrate your data and just view your data as well.

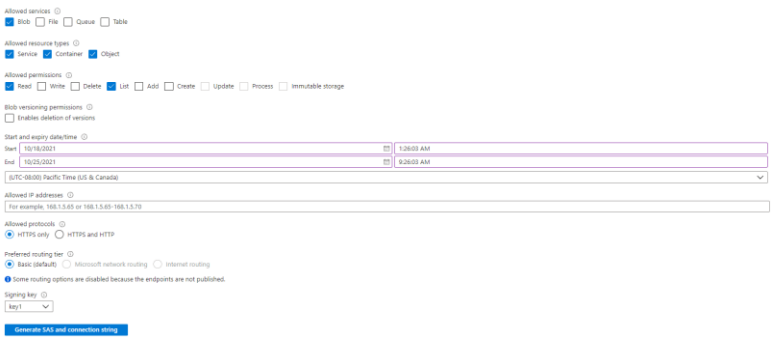
Executing a script on Azure Synapse Studio to create External tables

In the left-hand menu, click on develop, click on the plus icon in the develop screen and select SQL Script out of the options given.

Name the script on the left and copy and paste this SQL script on to the editor.

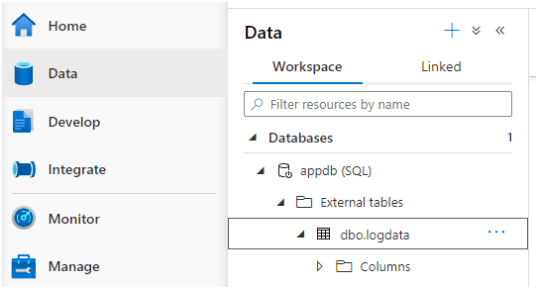
Now, we would be running a series of commands –

- Firstly, we would run the create data base command to create a database in the serverless pool
- Change the database from the top right of the editor where master is selected (refresh if the newly created database is not showing)
- Next, we create a master key that would be used to encrypt the database scope credentials which will allow ourselves to use the file that we would be using in our Data Lake Gen2 account.
- Now to create the scope credentials we need to get the shared access signature like follows and copy the SAS token –



- Paste it in the SECRET variable and remove the '?' from the front. Run the command for creating scoped credential.
- Next, define the location of your data by giving the location of your file in the LOCATION variable like this : https://<datalake_name>.dfs.core.windows.net/<your_container_name>
- Run the create external data source command with the new location that you entered
- Next, we are giving the format of your file. We name the file format as TextFileFormat, and start reading from the second row as the first row is headers
- Next, we create the external table. We give the column name and the types that we want as all the data would be coming in the string format. Remove all the NULLs from the command and execute the script.
- Now you can run the select * command to see all your data
- To save this script for future use you can click on Publish all button at the top of the editor and it would be available in your Synapse Studio

Now if we click on Data in Synapse Studio, we would be able to see our external table.



Creating a dedicated SQL pool

One of the main differences between the Serverless SQL pools and a Dedicated SQL pool is the ability to persist your data.

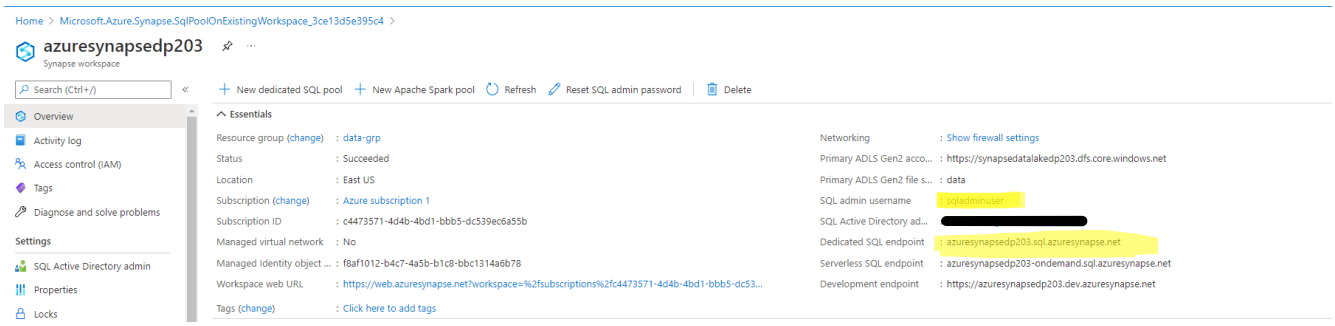
Go to Synapse workspace, from the left-hand menu select SQL pools. Click New and give a name, choose the performance level (for learning purposes, the lowest level is fine). Leave everything else as is and create the pool.

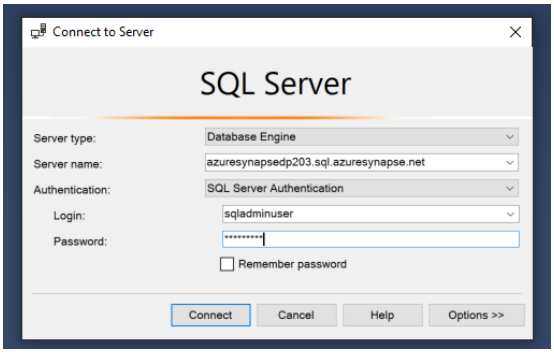
Once created this pool would also show up on synapse studio under databases.

Creating an external table in the Dedicated SQL pool using the Microsoft SQL Server Management Studio

A reason why we are using the SSMS is we get to log in as different users where as when we are using Synapse Studio, we are logged in with the admin user. So probably in a professional environment we are much more likely to work with different permissions compared to just working as the admin user.

Copy the Dedicated SQL Endpoint from your Synapse Workspace and paste it in the dialog box for a new connection in SSMS. For the login, cop the SQL admin username on the Synapse Workspace as well and give the password that you gave at the creation of the Synapse Workspace for the SQL admin.





Upon logging in, it would show your dedicated SQL pool. Right click on it and new query. Copy this script onto it.

We again go through the similar process of creating an external table:

- Select the first command to create a master key and execute it
- Copy your key for the Data lake gen2, and paste it in the SECRET variable. Execute it.
- Replace the name in the location variable with the name of your data lake and here we are specifying the driver (Hadoop) to source the external data. Execute it.
- Execute the external file format command to tell the format of external table
- Create the external table after removing all the NULLs

Now when we try to do a select * on logdata, it gives an error about converting varchar to datetime.

This error occurs because the date that we have in our file is not in the order in which Hadoop could infer it. So we would either have to change the format of the data in the file or we could clean the data or use a different driver to read the data.

After having cleaned the file, we again try to read the file. Here is the clean version of the file to skip the cleansing.

- Drop the table and create the table again with the new csv file

Upon doing the select * statement we can see the table contents now.

Creating an external table based on parquet files

The main advantages that I see while using parquet file is that they have much better compression and have faster read / access times.

Go to your data lake gen2 account, create a new folder inside your container and upload these files to it.

Go to your SSMS and copy this script on to the new query for your dedicated sql pool.

The same process is followed as before while creating the external table using the csv file. We first create a master key, then scoped credentials, the external data source, create the file format of the source of the data and then finally create the table.

Loading Data into a SQL Data Warehouse

There are different ways of loading the data-

- Using the copy statement – using T-SQL you can transfer data into a table in a SQL pool
- Azure Synapse pipeline – Here you can also perform transformation on your data
- Using Polybase to define external tables – here the data can be in an external data store but you can access it via external tables

Loading data into the SQL Warehouse using the Copy command

When loading the data into a sql warehouse we should not use the admin account. A separate user should be created for performing the load operations. After creating a new user, we can create a work group to allocate the right number of resources for the load operation.

Note – when you drop a table, it also drop the user created for the table (that's what I think, what happens if there are multiple tables associated with a user?)

Open up the SSMS and query the master database under system databases and not your dedicated sql pool that we had been using earlier.

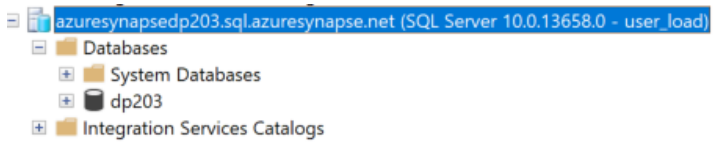
- Create a new user with the scrip present in this file.

Now we create a database user based on this new log in. But this new user would be in the context of the dedicated sql pool we have created so we query that for the purpose.

- Execute the create user query on the dedicated sql pool
- Execute the query to assign workload to the newly created user
- Drop the external table before recreating it with the create table command
- Grant the required privileges to the user

Now we can connect as the user that we just created. Click on connect on the left hand menu. Give the username password, click on options and go to the Additional Connection Parameters tab and enter database=<your dedicated sql pool name>

Upon connecting it would show up as –



When you query your sql pool, for select * from [logdata] it would show up as an empty table. Now we copy the log.csv data using the copy into command. In the command replace the name of the data lake with your own data lake and make sure the log.csv file is correctly addressed.

After executing the command the data would show up in the select * command.

Loading data into the SQL Warehouse using the Copy command for Parquet files

Begin by deleting the data from the existing logdata table with the command – *delete from [logdata]* as the sqladminuser since the newly created user doesn't have the permission to delete tables.

Then copy the data into the logdata table from the parquet files present in your own data lake with the following command. You would have to generate a new SAS token for your data lake where the parquet files are stored (remember to remove the ? from the front of the token).

```
COPY INTO [logdata] FROM
'https://datalake203prep.blob.core.windows.net/data/raw/parquet/*.parquet'
WITH
(
FILE_TYPE='PARQUET',
CREDENTIAL=(IDENTITY= 'Shared Access Signature', SECRET=' sv=2020-08-
04&ss=b&srt=sco&sp=rlcx&se=2021-10-20T07:27:55Z&st=2021-10-
19T23:27:55Z&spr=https&sig=7KynsIU4gh6aasv8h%2BvZLwUfcjkD7xrGlgtRu8qEocg%3D')
)
```

Upon running the select * command you can see the data.

Loading data using PolyBase

For this task, we are using this script.

To see your scoped credentials, you can issue the command :

```
SELECT * FROM sys.database_scoped_credentials
```

To see your external data sources you can use the command:

```
SELECT * FROM sys.external_data_sources
```

To see your external file formats in place :

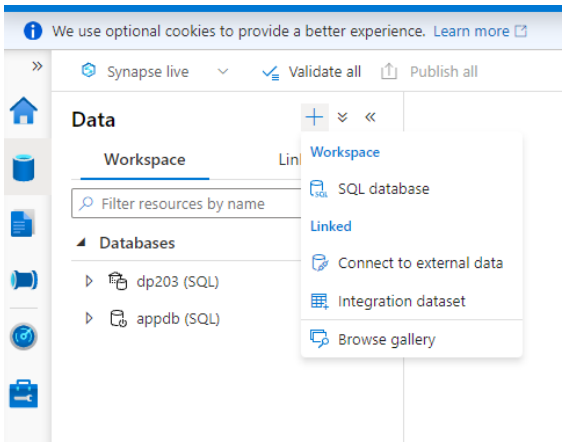
```
SELECT * FROM sys.external_file_formats
```

Proceed with the following steps:

- Begin by dropping the existing logdata table.
- Create an external table with your location, data source and the file format
- Do a select * on the table to see if the data is there in the table or not
- Now we use a create table that would use data from the external table and use that to create a table on the dedicated sql pool

Bulk insert into Dedicated SQL pool

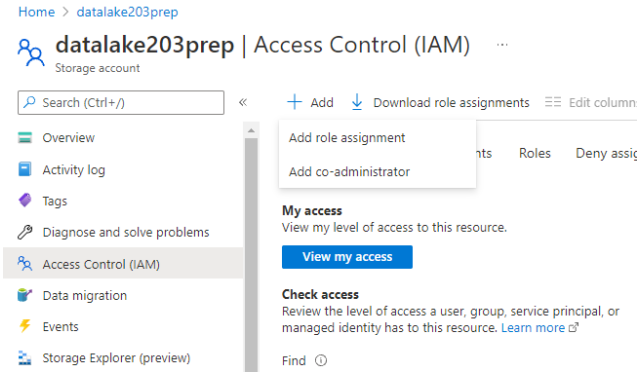
Open Synapse Studio and click on the data button on the left. Click on " + " to add an external source.



Select Azure Data Lake Storage Gen2. This would be a new linked service which is just like a connection onto a data store.

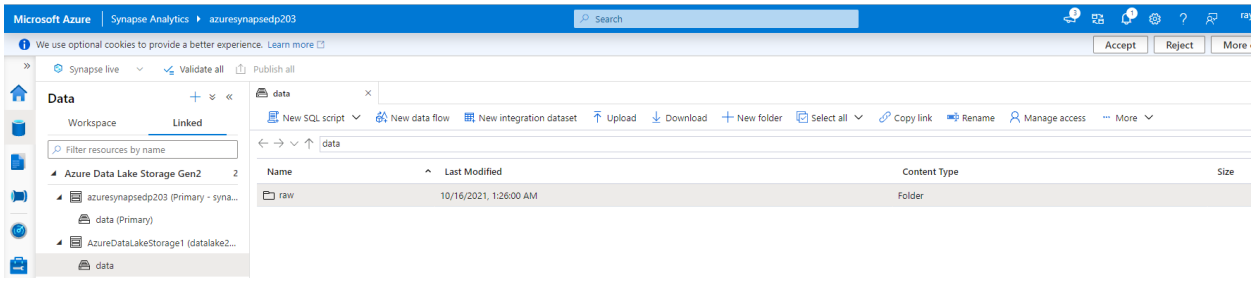
On the next screen give the linked service a name, select your subscription and select your storage account and hit create.

But before we can access data here as a linked service we would have to give permissions for ourselves for reading or writing. To do that, head over to the gen2 storage account, go to access control from the left-hand menu. Click on add and click on role assignment.



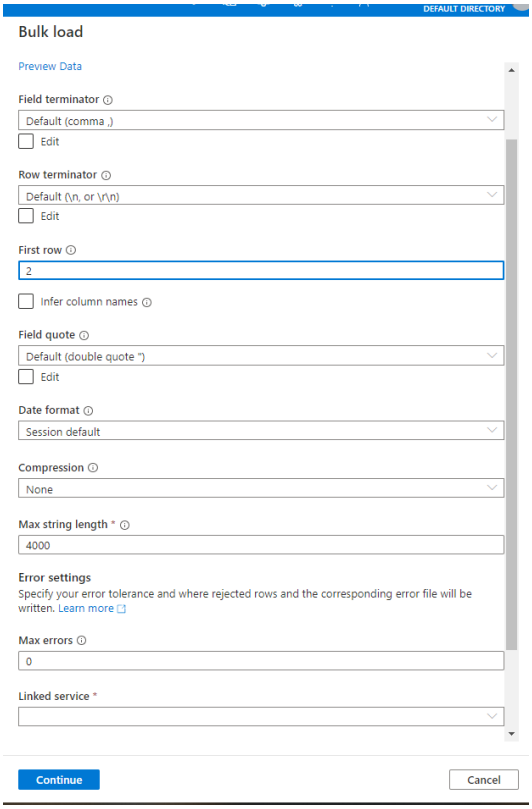
On the next screen, search and select Blob Data Contributor and hit next. In the next screen, click select members and yourself. Create the role assignment.

Now if you go back to the Azure Data Lake Gen2 account on Synapse linked storage and refresh, then you can see your gen2 storage account and its contents.

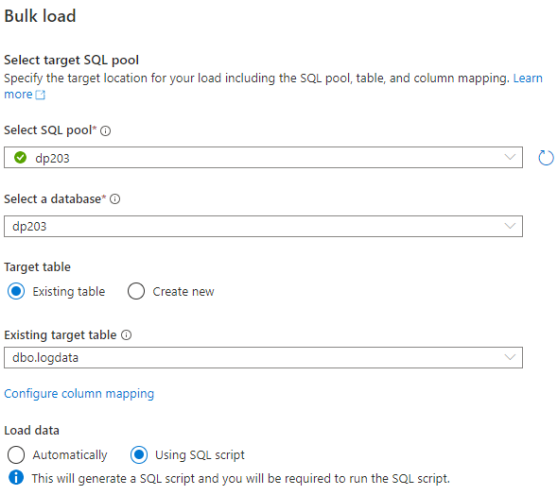


Go to your Log.csv file. Now if we right click on the Log.csv file, then we can have a preview of the data. **But** more importantly, we can also do a bulk load from the option *new sql script*.

An side window would pop up with options asking about field terminator, row terminator, first row (2 in our case), etc.



On the next screen click on existing table and select the logdata table and click on the Open Script button.



This would open up the script to load the data directly onto your table. Generates the copy statements for you.

```
--BEGIN
COPY INTO dbo.logdata
(Id 1, Correlationid 2, Operationname 3, Status 4, Eventcategory 5, Level 6, Time 7, Subscription 8, Eventinitiatedby 9, Resourcetype 10, Resourcegroup
FROM 'https://datalake203prep.dfs.core.windows.net/data/raw/Log.csv'
WITH
(
    FILE_TYPE = 'CSV'
    ,MAXERRORS = 0
    ,FIRSTROW = 2
    ,ERRORFILE = 'https://datalake203prep.dfs.core.windows.net/data/'
)
--END
GO

SELECT TOP 100 * FROM dbo.logdata
GO
```