

Custom Learnings

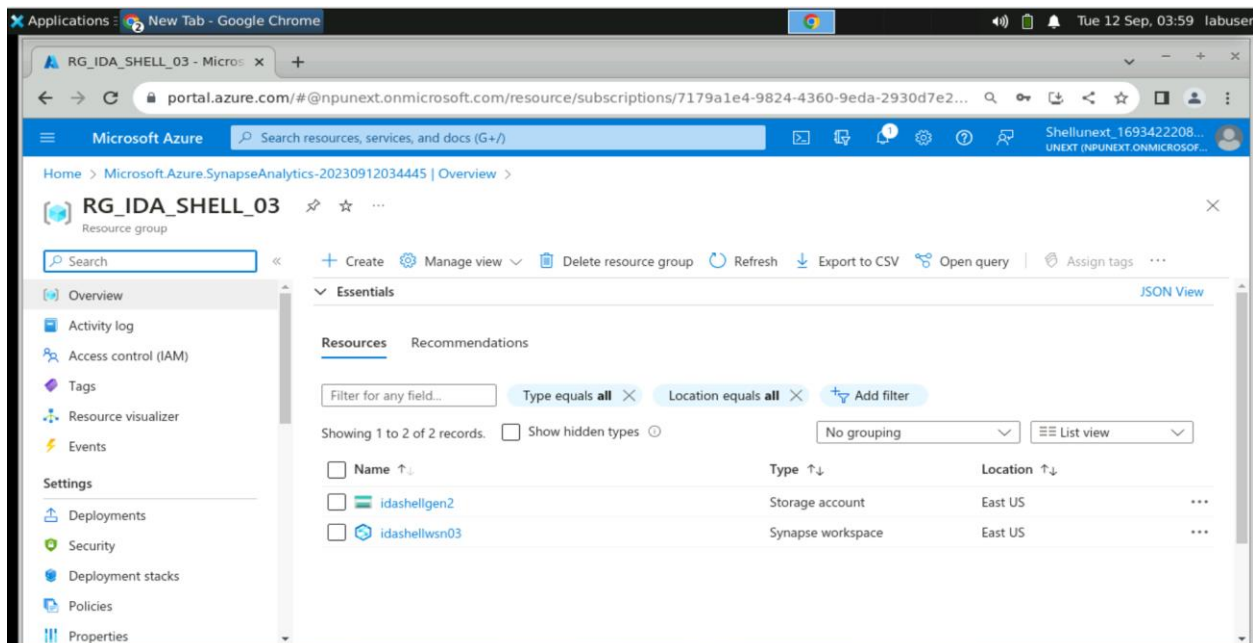
Day 10

Synapse Analytics:

It is used for

- ⇒ Process data
- ⇒ Create own pipeline
- ⇒ Tools present in Synapse Analytics
 - ADF
 - Spark Notebook
 - SQL DB
- ⇒ Managed Resource Group
 - Additional resource group sits in this
 - Not mandatory
 - It's recommended to create one as some naming convention need to be followed
 - If no value provided, a default one is created by Synapse
- ⇒ Create a new Synapse account
 - contributor
 - Account name – give a new account
 - File system name – input
 - Authentication method: use both local and azure active directory
 - No changes in networking
 - Synapse is very costly.

Creation of Azure Synapse Analytics:



⇒ Open Synapse Studio

- Develop option: create notebooks that support spark, sql
- Monitoring: monitor pipelines

*** Synapse does not support Blob Storage Account

⇒ Pool

- Serverless
 - Per TB of data the costing is done
 - To avail this service, a cost needs to be paid
- Dedicated pool
 - Very costly
 - Provides additional functionalities
 - It allows creation of tables, and other resources which are not present in serverless
- Spark Pool
 - Costly
 - For Big Data mgmt.
- SQL Pool
 - For relational data in lesser volume
- Difference between serverless and dedicated pool
 - table supported
 - partition
 - external tables
 - table index
 - table distribution

DIU (Data Integration Unit):

- ⇒ A Data Integration Unit (DIU) is a measure that represents the power of a single unit in Azure Data Factory and Synapse pipelines. Power is a combination of CPU, memory, and network resource allocation.

DWC (Data Warehousing Capacity) – 100 is the min value

⇒ SQL query to read a csv file

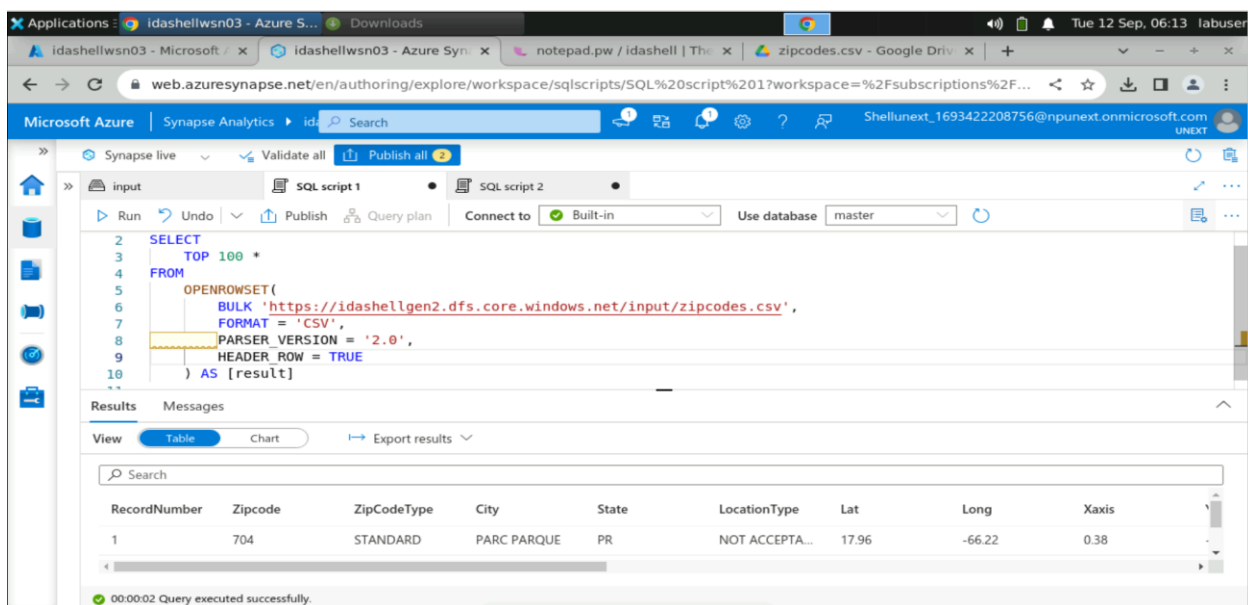
- Parser_version: defines how the csv file is read. Should always be the latest
- Give header = true to convert the first line of the csv file into the column names
- Give header_row = true
- To carry out fine-grained segregation in Azure DB
 - Use schemas to make segregation
- How to create a new database in Synapse
 - Use create database command in workspace
 - Or use UI to create a database
 - Using UI choose the type of database needed.
 - If serverless is chosen then tables option is not seen
- To accommodate special characters, UTF-8 encoding is required

- Create database with UTF-8 enabled.
- To do the above:
 - create database idashell collate Latin1_general_100_BIN2_UTF8
- SSMS: SQL Server Management Studio
 - It is a user-friendly tool that allows connection to multiple sql servers in different tabs
 - It is like query editor but more robust and this is where the data engineers mostly work upon and not other IDEs like query editor in Azure SQL DB, or Azure Synapse query script
 - Go to connect
 - Db regime
 - Server name: give the endpoint copied from the synapse home tab
 - Authentication: give user and password
 - Connect
- ⇒ Only the users mentioned in the Users tab can access the SQL DB
- ⇒ Create login loginname with password pass
- ⇒ Create user username from loginname
- ⇒ Each application that needs to access this Synapse data needs to have a user named after it
- ⇒ The same user is used to get the data from the sql db in synapse

If collation is not used before the database is created and collation is to be made during production then

- ⇒ The user altering the db should lock the database for themselves
- ⇒ No attempt should be made to alter the db while in production and available to others

Perform some sql queries :



The screenshot shows the Microsoft Azure Synapse Analytics interface. The top navigation bar includes 'Synapse live', 'Validate all', and 'Publish all'. The main area displays a SQL script in 'SQL script 1' with the following code:

```

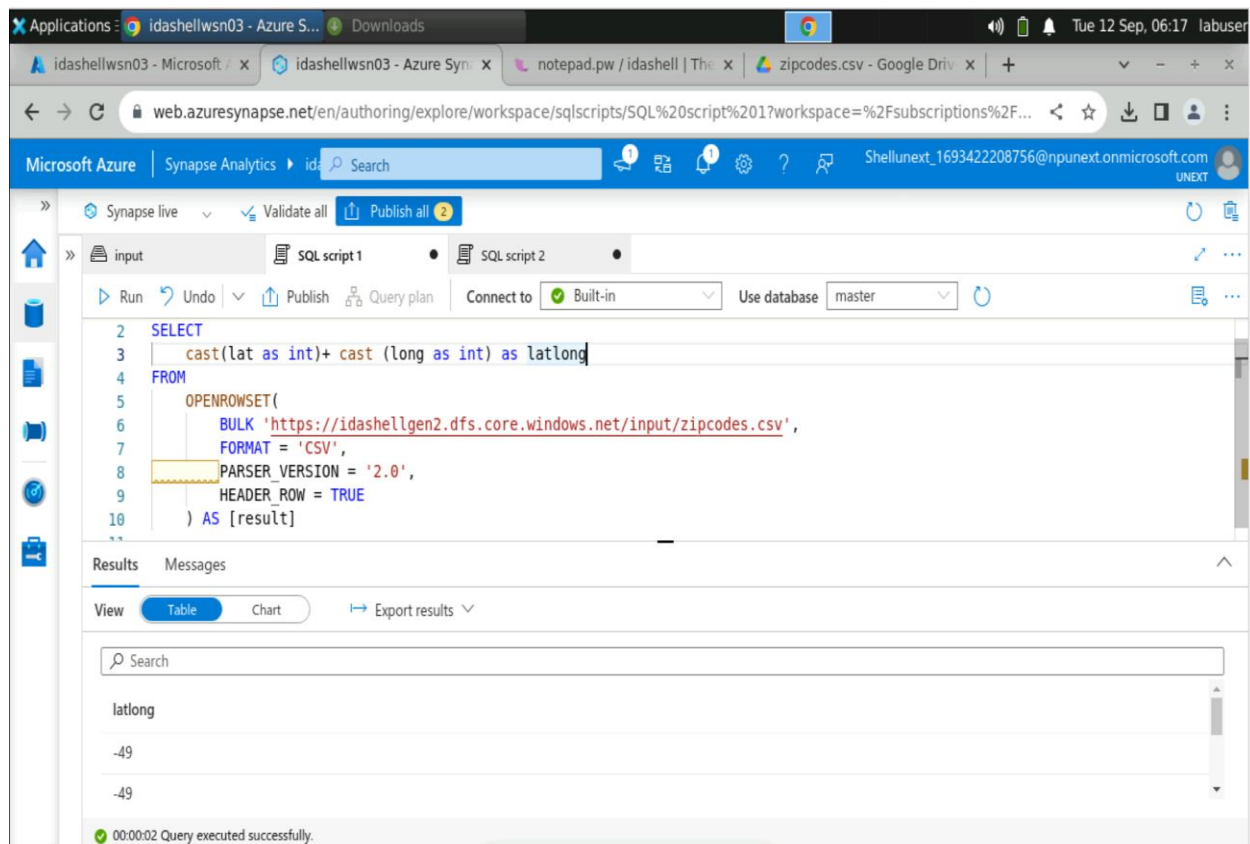
2 SELECT
3 TOP 100 *
4 FROM
5 OPENROWSET(
6 BULK 'https://idashellgen2.dfs.core.windows.net/input/zipcodes.csv',
7 FORMAT = 'CSV',
8 PARSER_VERSION = '2.0',
9 HEADER_ROW = TRUE
10 ) AS [result]

```

Below the script, the 'Results' tab is active, showing a table view of the query output. The table has the following columns: RecordNumber, Zipcode, ZipCodeType, City, State, LocationType, Lat, Long, and Xaxis. The first row of data is visible:

RecordNumber	Zipcode	ZipCodeType	City	State	LocationType	Lat	Long	Xaxis
1	704	STANDARD	PARC PARQUE	PR	NOT ACCEPTA...	17.96	-66.22	0.38

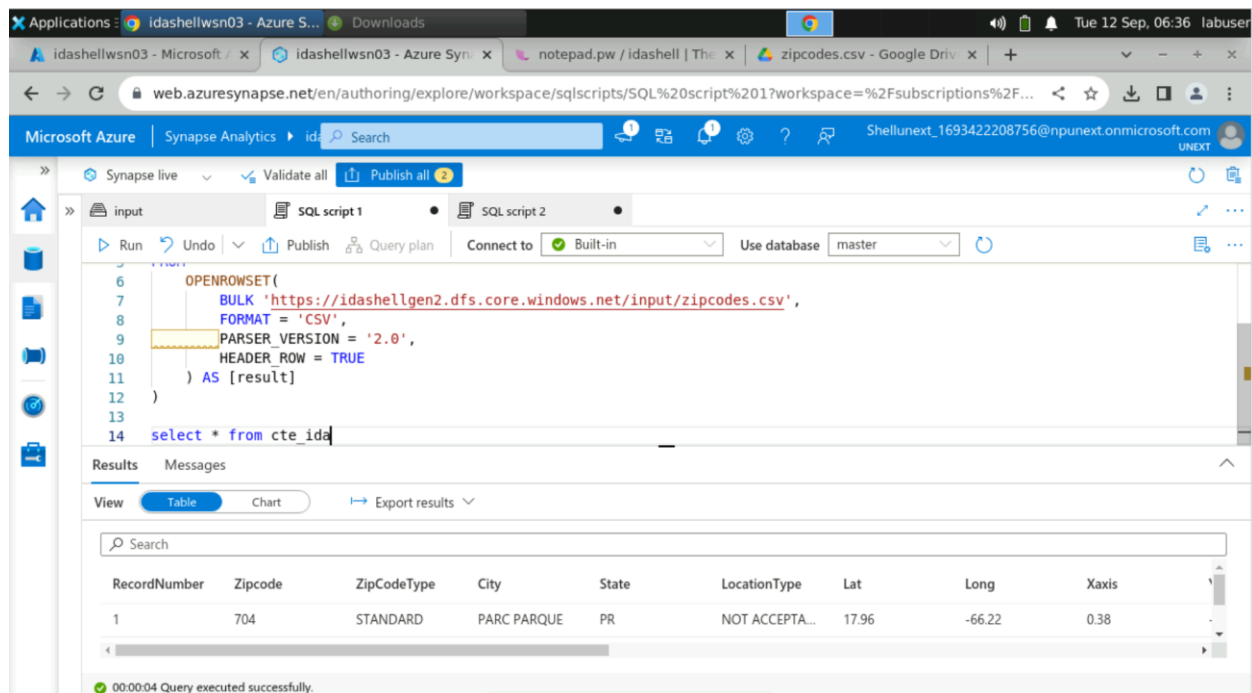
At the bottom, a status message indicates: '00:00:02 Query executed successfully.'



Common Table Expression (CTE)

- ⇒ It is used to carry out data transformation using temporary storage
- ⇒ CTE queries can't be run separately as the storage is temporary
- ⇒ Once CTE variable is created, rest of the queries must also run at the same time
- ⇒ A separate temp table can't be used to store the CTE result and run queries separately. as the data is stored in a temp DB. Since temp DBs do not have high storage and storing large volumes will incur more cost
- ⇒ It is thus advised to use such temp tables only for small amounts of data otherwise to go with CTE

Perform the Common table expression (cte) :



Serverless View Creation

- ⇒ Once CTE runs, create a view using 'create view view_name (CTE)' command
- ⇒ Views are used so that visualization can be done in Power BI
- ⇒ Power BI supports a lot of connectors for views
- ⇒ To generate a report from Power BI, the serverless should be running 24 hr. This will incur a lot of cost if instead of serverless dedicated is used. Hence, serverless is always used for visualizing the data

DEDICATED POOL

Synapse Analytics

- ⇒ It is a one stop shop for storage, SQL, analytics, etc..
- ⇒ A lot of resources are present under the same roof
- ⇒ EDW (Enterprise Data Warehousing)
- ⇒ Data integration, transformation, analytics all can be done under Synapse Analytics
- ⇒ MPP (Massive Parallel Processing)
- ⇒ Control and compute node (Master and Worker)
- ⇒ Control Node
 - For managing the compute nodes

⇒ Compute node:

- It is used to carry out data transformations and analytics

⇒ Storage node

- Decoupled node. It is not a part of the compute node

⇒ Data Movement service

- Movement of data from one compute node to another
- By default there are 60 compute nodes that run parallelly

⇒ Data Distributions

- Replication tables
 - It is for small tables
 - For 60 compute nodes, if all the tables are replicated then it will consume a lot of memory
- Round Robin
 - It is good for temporary/staging table
 - There are no specific keys for data distribution
- Hash Distributed Tables
 - Hash values are created for all the rows
 - It helps in determining if the data has changed in the table or not
 - Use a hash function to take the columns and create a hash value for the table rows
 - Hash functions can use two columns for example to create a hash value
 - If there are multiple rows that have the same values for these columns then these records will have the same hash values

⇒ Data Partitioning

- Note: in case of high granularity, a lot of different partitions will be created
- To avoid this, choose columns that provide the least number of partitions

⇒ Indexing

- Clustered Columnstore
 - Updateable primary storage
 - Good for read-only
- Clustered Index
 - Index is stored in the same order as the data being indexed
- Heap
 - Data is not in any particular order

⇒ Apache spark pool

- Unlike dedicated pool, the number of worker nodes can be given by the user and is not 60 by default

- It also provides a scaling option that allows the user to provide a range of worker nodes for the processing for different loads that is selected automatically
- Configure the number of worker nodes as well as the size of them
- Python, SQL, Scala, R are the languages supported in Spark Pool Notebook
 - Use Dataframe in Spark
 - The read command for a file in Spark
 - Spark read format option load
 - Any spark db comes under lake database

The screenshot shows the Microsoft Azure Synapse Analytics interface. On the left, the 'Data' pane shows the workspace 'idashellwsn03' with a linked resource 'input (Primary)' under 'Azure Data Lake Storage Gen2'. The main pane displays a Spark notebook with the following code:

```
1 %%pyspark
2 df = spark.read.load('abfss://input@idashellgen2.dfs.core.windows.net/zipcodes.csv'
3   ## If header exists uncomment line below
4   ##, header=True
5   )
6 display(df.limit(10))
```

The execution status is 'Ready'. Below the code, a message indicates: '[1] ✓ 4 min 59 sec session started in 4 min 33 sec 566 ms. Command executed in 25 sec 327 ms by Shellunext_1693422208756 on 10...'. The job execution succeeded with 'Spark 2 executors 8 cores'. Below this, a table view shows the first 10 rows of data:

_c0	_c1	_c2	_c3
RecordNumber	Zipcode	ZipCodeType	City
1	704	STANDARD	PARC PARQU

⇒ Dedicated pool

- Default distribution type is "Hash Values"
- Indexing: clustered
- To insert data
 - Copy command
 - Copy data from the file and load into the table
 -
 - To load the data
 - Use the concept of bulk load
 - Infer schema option allows the conversion of column data types as present in the table

Applications: idashellwsn03 - Azure S... Tue 12 Sep, 10:46 labuser

zipcodes.csv - Microsoft A... idashellwsn03 - Azure Syn... notepad.pw / idashell | The... zipcodes.csv - Google Driv... +

web.azure.synapse.net/en/authoring/explore/workspace/sqlscripts/SQL%20script%204?workspace=%2Fsubscriptions%2F7179...

Microsoft Azure | Synapse Analytics | idashell Search

Synapse live Validate all Publish all 4

SQL script 1 SQL script 2 Notebook 1 Notebook 2 SQL script 3 SQL script 4

Run Undo Publish Query plan Connect to DSP_03 Use database DSP_03

```
1 IF NOT EXISTS (SELECT * FROM sys.objects O JOIN sys.schemas S ON O.schema_id = S.schema_id WHERE O.NAME = '[test]' AND O.TYPE = 'U')
2 CREATE TABLE [dbo].[test]
3 (
4     [RecordNumber] bigint,
5     [Zipcode] bigint,
6     [ZipCodeType] nvarchar(4000),
7     [City] nvarchar(4000),
8     [State] nvarchar(4000),
```

Results Messages

View Table Chart Export results

RecordNumber	Zipcode	ZipCodeType	City	State	LocationType	Lat	Long	Xaxis
10	708	STANDARD	BDA SAN LUIS	PR	NOT ACCEPTA...	18.14	-66.26	0.38
49348	34487	PO BOX	HOMOSASSA	FL	PRIMARY	28.78	-82.61	0.11

00:00:08 Query executed successfully.