



Creating a Dedicated SQL Pool and External Tables

A **Dedicated SQL Pool** in Azure Synapse Analytics (formerly Azure SQL Data Warehouse) is a resource that provides scalable, high-performance data warehousing capabilities within the Azure Synapse ecosystem. It is specifically designed for large-scale, high-volume analytical workloads.

Key Features of Dedicated SQL Pool:

1. Massively Parallel Processing (MPP):

- The architecture leverages MPP, distributing data and queries across multiple nodes for high-performance parallel processing.
- This enables the pool to handle large datasets efficiently.

2. Scalability:

- Compute resources can be scaled up or down based on your workload requirements.
- You pay only for the resources you use, and scaling can often be done without interrupting operations.

3. Columnar Storage:

- Data is stored in a columnar format, which is highly optimized for analytical queries and reduces storage costs through compression.

4. Data Distribution:

- Data is distributed across nodes using one of three strategies:
 - **Hash distribution:** Distributes rows based on a key column, useful for minimizing data movement during queries.
 - **Round-robin distribution:** Evenly distributes rows across nodes without a specific pattern.
 - **Replicated distribution:** Copies the entire table to each node, ideal for small tables used in joins.

5. Integrations:

- Seamlessly integrates with other Azure services like Azure Data Factory, Azure Data Lake Storage, and Power BI.
- Supports integration with on-premises data sources and other cloud services.

6. Advanced Query Capabilities:

- Supports T-SQL for querying and managing data.

- Includes advanced features like partitioning, indexing, and statistics to optimize query performance.

7. Security:

- Provides enterprise-grade security, including role-based access control (RBAC), auditing, and encryption at rest and in transit.

8. Concurrency Management:

- Optimized for managing multiple users and queries, balancing performance across concurrent operations.

Common Use Cases:

- **Enterprise Data Warehousing:** Centralized storage and analysis of organizational data for reporting and analytics.
- **Big Data Analytics:** Analyzing terabytes or petabytes of data with high query performance.
- **ETL and ELT Workflows:** Efficiently loading, transforming, and querying large datasets for data integration.

Comparison with Serverless SQL Pool:

- **Dedicated SQL Pool:**

- Pre-allocated compute resources.
- Best suited for predictable, high-volume workloads.
- Charges based on provisioned resources, even when idle.

- **Serverless SQL Pool:**

- On-demand querying with no pre-allocation.
- Ideal for exploratory analytics or infrequent queries.
- Charges based on data processed per query.

By utilizing Dedicated SQL Pools, organizations can run large-scale analytical queries efficiently, enabling faster decision-making and deeper insights from their data.



Difference Between Dedicated SQL Pool and Serverless SQL Pool

The key differences between **Dedicated SQL Pool** and **Serverless SQL Pool** in Azure Synapse Analytics revolve around their architecture, use cases, pricing models, and operational approaches. Here's a detailed comparison:

1. Provisioning and Architecture

Feature	Dedicated SQL Pool	Serverless SQL Pool
Provisioning	Pre-allocated compute resources (fixed cluster size).	No need for pre-allocation; runs on-demand.
Storage	Data is stored within the SQL pool (provisioned storage).	Queries external data directly from data lakes (e.g., Azure Data Lake).

2. Pricing Model

Feature	Dedicated SQL Pool	Serverless SQL Pool
Billing	Based on provisioned resources (compute and storage).	Based on the amount of data processed per query.
Idle Cost	Incurs charges even when not in use.	No charges when not in use.

3. Performance

Feature	Dedicated SQL Pool	Serverless SQL Pool
Workload	Designed for high-performance, predictable workloads.	Suited for lightweight, exploratory workloads.
Scalability	Scale up or down manually or programmatically.	Automatically handles workload with no scaling required.

4. Data Access and Querying

Feature	Dedicated SQL Pool	Serverless SQL Pool
Data Storage	Stores data in a structured, columnar format inside the pool.	Queries data directly from external sources (e.g., Parquet, CSV).
Use Case	For structured, preloaded datasets.	For ad-hoc querying of external datasets without loading them.

5. Use Cases

Feature	Dedicated SQL Pool	Serverless SQL Pool
Best for	Enterprise-grade data warehouses and predictable analytics.	Ad-hoc analytics, data exploration, or lightweight workloads.
Integration	Optimized for structured, relational data in Synapse.	Ideal for querying data stored in Azure Data Lake Storage.

6. Query Optimization

Feature	Dedicated SQL Pool	Serverless SQL Pool
Performance Tuning	Requires optimization like indexing, partitioning, and distribution strategies.	Automatically optimized for on-demand queries.
Concurrency	Can manage multiple concurrent queries with workload isolation.	Supports concurrent ad-hoc queries but is not designed for heavy concurrency.

Comparison Table Summary

Feature	Dedicated SQL Pool	Serverless SQL Pool
Provisioning	Fixed compute resources	No provisioning required
Billing	Based on provisioned resources	Based on data processed per query
Idle Cost	Charges even when idle	No charges when idle
Use Case	Large-scale, predictable workloads	Ad-hoc queries on external datasets
Data Location	Stored within the pool	External sources like Azure Data Lake

When to Choose:

1. Dedicated SQL Pool:

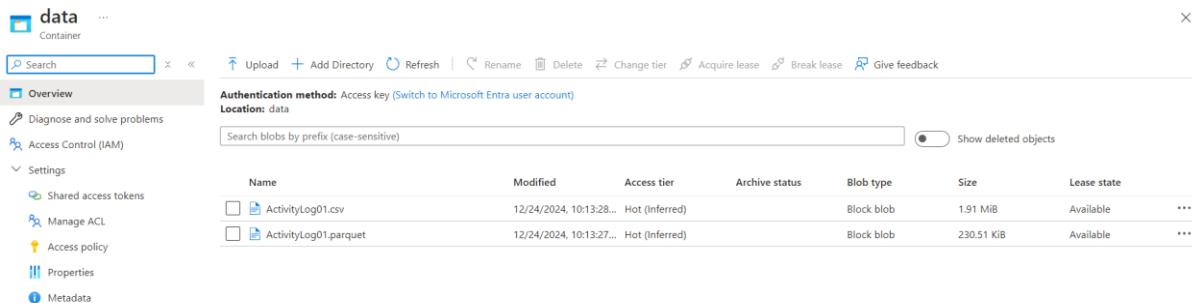
- Use for large-scale, high-performance workloads.
- Ideal for scenarios where you need consistent, fast response times and handle structured data.

2. Serverless SQL Pool:

- Use for exploratory data analysis or querying external data directly without data movement.
- Ideal for infrequent or lightweight workloads with unpredictable demand.

To begin with this Lab:

1. In this lab, first we will create a dedicated SQL Pool in Azure Synapse, also known as a data warehouse. Then we will create some external and persisted tables in which we will load the data from our data lake or Gen 2 Storage account.
2. Here you can see that we have the storage account or data lake in which we have the parquet and CSV file with us.

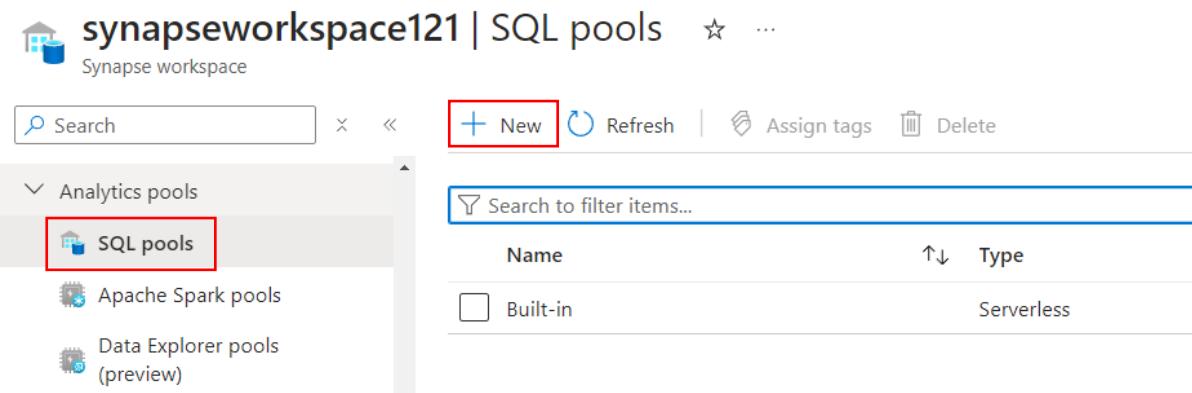


The screenshot shows the Azure Storage Explorer interface. A container named 'data' is selected. The left sidebar includes options like Overview, Diagnose and solve problems, Access Control (IAM), Settings, Shared access tokens, Manage ACL, Access policy, Properties, and Metadata. The main pane displays a list of blobs with the following details:

Name	Modified	Access tier	Archive status	Blob type	Size	Lease state
ActivityLog01.csv	12/24/2024, 10:13:28...	Hot (Inferred)		Block blob	1.91 MiB	Available
ActivityLog01.parquet	12/24/2024, 10:13:27...	Hot (Inferred)		Block blob	230.51 KiB	Available

Creating External Table CSV

3. To create a dedicated SQL Pool, you must complete the previous 4 labs. Now open your Synapse Analytics and from the left pane come to Analytics Pools then choose SQL pools. Click on the new button to create a dedicated SQL Pool.



The screenshot shows the Azure Synapse Analytics workspace titled 'synapseworkspace121 | SQL pools'. The left sidebar lists Analytics pools, with 'SQL pools' highlighted and a red box around it. The main pane shows a table of existing SQL pools:

Name	Type
Built-in	Serverless

4. Give a name to your dedicated SQL Pool and choose the lowest Performance level which is DW100c and move to the Review page then create your SQL Pool or Data warehouse.

New dedicated SQL pool ...

* Basics * Additional settings Tags Review + create

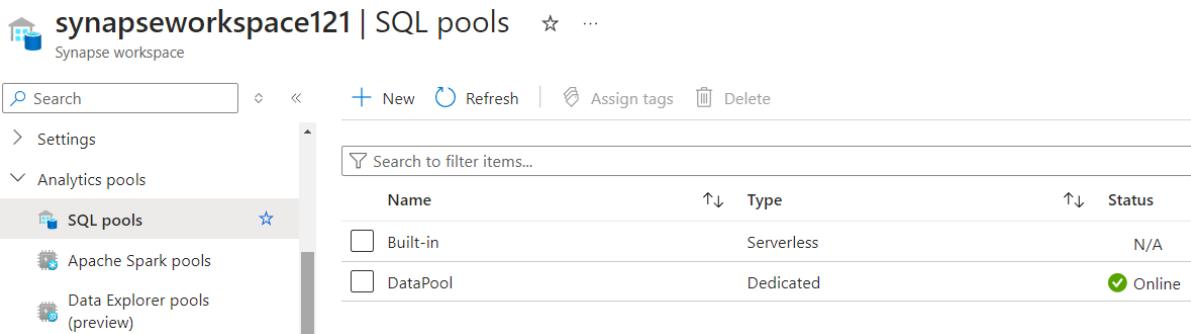
Create a dedicated SQL pool with your preferred configurations. Complete the Basics tab then go to Review + Create to provision with smart defaults, or visit each tab to customize. [Learn more](#)

Dedicated SQL pool details

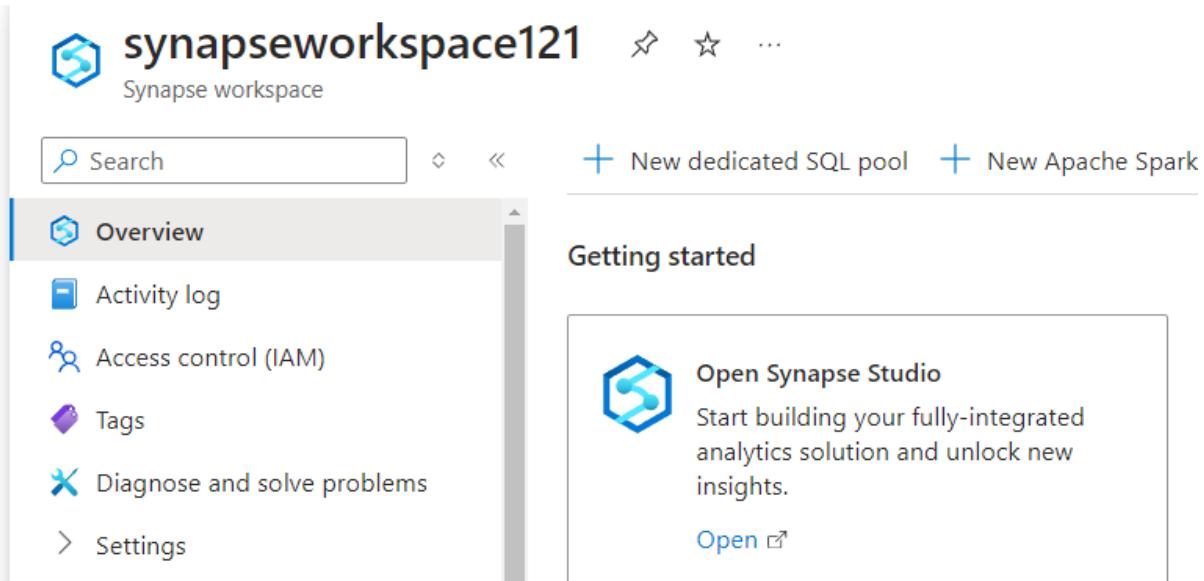
Name your dedicated SQL pool and choose its initial settings.

Dedicated SQL pool name *	DataPool
Performance level ⓘ	<input type="range" value="100"/> DW100c
Estimated price ⓘ	Est. Cost Per Hour 115.64 INR View pricing details

- Now wait until the deployment is completed and you will see that in the SQL pools. Once it is done then open the Synapse Studio from the overview page.



The screenshot shows the 'synapseworkspace121 | SQL pools' page. On the left, there's a sidebar with options like 'Settings', 'Analytics pools', 'SQL pools' (which is selected and highlighted in blue), 'Apache Spark pools', and 'Data Explorer pools (preview)'. The main area has a search bar and a table with columns 'Name', 'Type', and 'Status'. It lists two pools: 'Built-in' (Serverless, N/A status) and 'DataPool' (Dedicated, Online status).

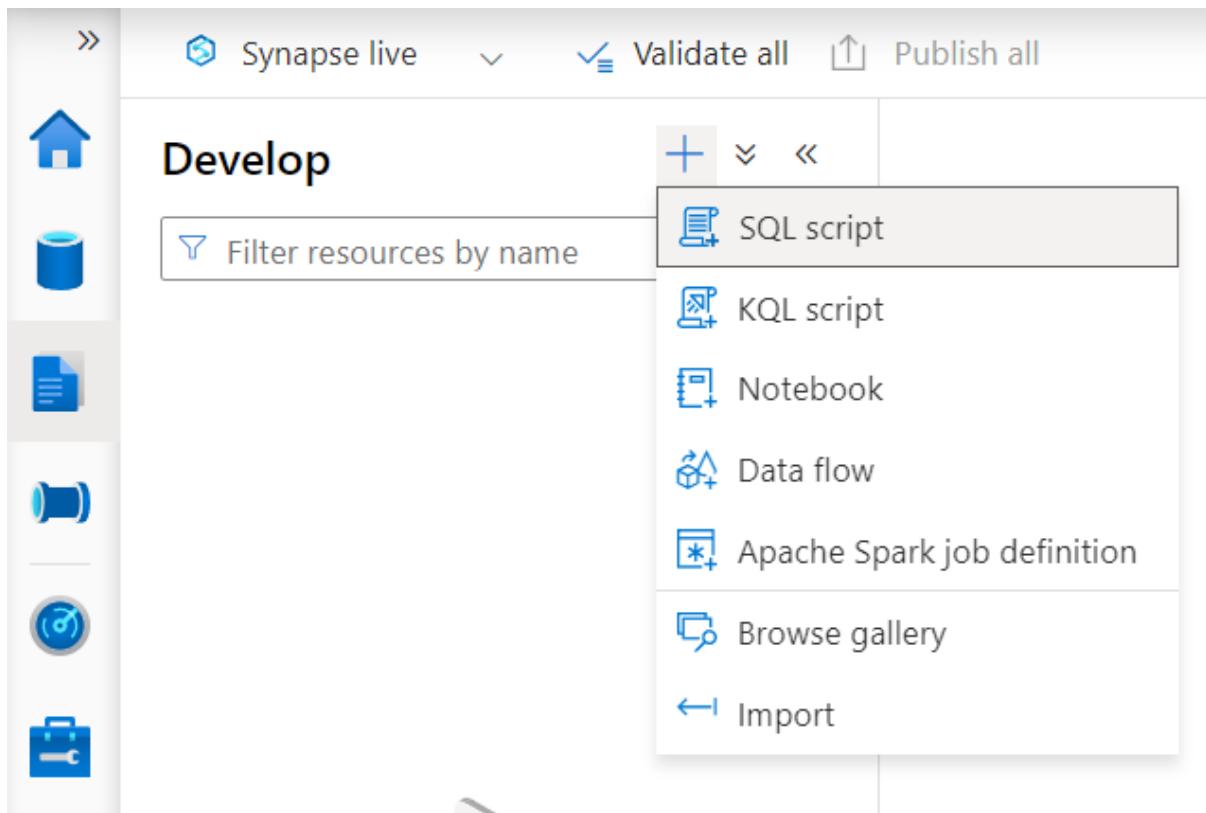


The screenshot shows the 'synapseworkspace121' workspace overview. The left sidebar includes 'Overview' (selected and highlighted in blue), 'Activity log', 'Access control (IAM)', 'Tags', 'Diagnose and solve problems', and 'Settings'. The main area features a 'Getting started' section with a button to 'Open Synapse Studio'. It also includes a 'New dedicated SQL pool' and 'New Apache Spark' button.

6. On Synapse Studio, go to the Data tab and in the workspace, you will see the dedicated SQL Pool as your Database here.

The screenshot shows the Synapse Studio interface with the 'Data' tab selected. The left sidebar has icons for Home, Workspace, Datasets, Pipelines, and Scripts. The main area is titled 'Data' and shows a 'Workspace' tab selected. Below it is a search bar with the placeholder 'Filter resources by name'. Under the 'Workspace' tab, there is a section for 'SQL database' which contains one item: 'DataPool (SQL)'. This item has several sub-folders: Tables, External tables, External resources, Views, Programmability, Schemas, and Security. The 'Linked' tab is also visible but not selected.

7. Now open the Develop Tab and click the Plus icon to create a new SQL Script.



- From the top you need to change the connection to the dedicated SQL Pool as you can see below in the snapshot.

- Now we will create an External table for CSV file inside our dedicated SQL Pool. First, we will start with creating the master key for the SAS Token and then go to the Storage account where your files have been stored. Then from the left pane choose Shared Access Signature and create an SAS Token for your code by choosing the allowed service, resource type, and permission.

```

1 CREATE MASTER KEY ENCRYPTION BY PASSWORD = 'P@ssword@123'
2
3 CREATE DATABASE SCOPED CREDENTIAL sasToken
4 WITH IDENTITY ='SHARED ACCESS SIGNATURE',
5 SECRET='sv=2022-11-02&ss=b&srt=sco&sp=rl&se=2024-12-24T14:53:02Z&st=2024-12-24T06:53:02Z&spr=https://&sig=...'
```

A shared access signature (SAS) is a URI that grants restricted access rights to Azure Storage resources. You can provide a shared access signature to clients who should not be trusted with your storage account key but whom you wish to delegate access to certain storage account resources. By distributing a shared access signature URI to these clients, you grant them access to a resource for a specified period of time.

An account-level SAS can delegate access to multiple storage services (i.e. blob, file, queue, table). Note that stored access policies are currently not supported for an account-level SAS.

[Learn more about creating an account SAS](#)

Allowed services ⓘ
 Blob File Queue Table

Allowed resource types ⓘ
 Service Container Object

Allowed permissions ⓘ
 Read Write Delete List Add Create Update Process Immutable storage Permanent delete

Allowed protocols ⓘ
 HTTPS only HTTPS and HTTP

Preferred routing tier ⓘ
 Basic (default) Microsoft network routing Internet routing

Some routing options are disabled because the endpoints are not published.

Signing key ⓘ
 key1 ▾

Generate SAS and connection string

Connection string
 BlobEndpoint=https://thestorageaccount1201.blob.core.windows.net/QueueEndpoint=https://thestorageaccount1201.queue.core.windows.net/FileEndpoint=https://thestorageaccount1201.file.core.windows.net/

SAS token ⓘ
 sv=2022-11-02&ss=b&srt=sco&sp=rl&se=2024-12-24T14:53:02Z&st=2024-12-24T06:53:02Z&spr=https&sig=j6mLaO5KW%2FcCcFEHLme1twsvkYPPp6Rtv... ▾

Blob service SAS URL
 https://thestorageaccount1201.blob.core.windows.net/?sv=2022-11-02&ss=b&srt=sco&sp=rl&se=2024-12-24T14:53:02Z&st=2024-12-24T06:53:02Z&spr=https&sig=j6mLaO5KW%2FcCcFEHLme1twsvkYPPp...

10. Then we need to create the external data source for our table and here you can see that we are using different way to give the location type which is Hadoop, so use the same method first give the container name then follow with the @ and the storage account name. After that, we have to create an external file format as you can see below.

```

9   CREATE EXTERNAL DATA SOURCE srcActivityLog
10  WITH
11  (
12      LOCATION='abfss://data@thestorageaccount1201.blob.core.windows.net',
13      TYPE = HADOOP,
14      CREDENTIAL=sasToken
15  )

17  CREATE EXTERNAL FILE FORMAT delimitedTxtFileFormat WITH
18  (
19      FORMAT_TYPE=DELIMITEDTEXT,
20      FORMAT_OPTIONS(
21          FIELD_TERMINATOR=',',
22          FIRST_ROW=2
23      )
24  )

```

11. Now we are going to create our external table in which the data will be fetched from the data lake.

```

26  CREATE EXTERNAL TABLE ActivityLog
27  (
28      [Correlationid] varchar(200),
29      [Operationname] varchar(300),
30      [Status] varchar(100),
31      [Eventcategory] varchar(100),
32      [Level] varchar(100),
33      [Time] varchar(100),
34      [Subscription] varchar(200),
35      [Eventinitiatedby] varchar(1000),
36      [Resourcetype] varchar(300),
37      [Resourcegroup] varchar(1000),
38      [Resource] varchar(2000))
39  WITH (
40      LOCATION='/ActivityLog01.csv',
41      DATA_SOURCE=srcActivityLog,
42      FILE_FORMAT=delimitedTxtFileFormat
43  )

```

12. Below you can see that we have the data with us. Once you are done then publish your changes to save everything.

45 SELECT * FROM ActivityLog;

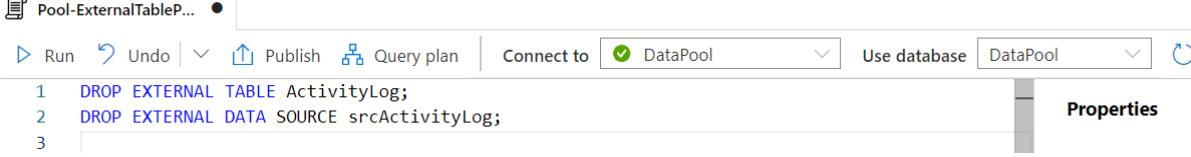
The screenshot shows a SQL query results table with the following data:

Correlationid	Operationname	Status	Eventcategory	Level	Time
0e67cb2c-9ae4...	Delete resource...	Started	Administrative	Informational	2024-05-10T07:...
a327510b-5407...	'audit' Policy ac...	Succeeded	Policy	Warning	2024-04-23T05:...
42717fb3-4419...	Create or Upda...	Started	Administrative	Informational	2024-04-23T03:...
6b81ca7f-79e2...	Delete website	Succeeded	Administrative	Informational	2024-04-23T03:...
caf1a463-d664-...	Delete Web Ap...	Started	Administrative	Informational	2024-04-22T16:...
8a358d1a-770a...	SwapWebSite	Succeeded	Administrative	Informational	2024-04-22T09:...

Creating External Table Parquet

13. Now we will create the external table for the Parquet file, for that first we need to create a new SQL Script.

14. First, we will drop our external table and external data source and change the database from serverless to a dedicated SQL Pool.



```
1  DROP EXTERNAL TABLE ActivityLog;
2  DROP EXTERNAL DATA SOURCE srcActivityLog;
3
```

15. Then we will create the external data source for our parquet file, after that we will create the external file format.

```
5  CREATE EXTERNAL DATA SOURCE srcActivityLog
6  WITH
7  (
8      LOCATION='https://thestorageaccount1201.blob.core.windows.net/data',
9      CREDENTIAL=sasToken
10 )

12  CREATE EXTERNAL FILE FORMAT parquetFileFormat WITH
13  (
14      FORMAT_TYPE=PARQUET,
15      DATA_COMPRESSION='org.apache.hadoop.io.compress.SnappyCodec'
16 )
```

16. Here is our external parquet table which we need to create next.

```
18  CREATE EXTERNAL TABLE ActivityLog
19  (
20      [Correlationid] varchar(200),
21      [Operationname] varchar(300),
22      [Status] varchar(100),
23      [Eventcategory] varchar(100),
24      [Level] varchar(100),
25      [Time] varchar(100),
26      [Subscription] varchar(200),
27      [Eventinitiatedby] varchar(1000),
28      [Resourcetype] varchar(300),
29      [Resourcegroup] varchar(1000),
30      [Resource] varchar(2000))
31  WITH (
32      LOCATION='/ActivityLog01.parquet',
33      DATA_SOURCE=srcActivityLog,
34      FILE_FORMAT=parquetFileFormat
35 )
```

17. As you can see below we have the data inside our tables.

```
37  SELECT * FROM ActivityLog;
38
```

Results Messages

View

Table

Chart

Export results

Search

Correlationid	Operationname	Status	Eventcategory	Level	Time
e6c77441-737b...	'auditIfNotExist...	Succeeded	Policy	Informational	2024-05-15T11:...
b30f69ec-7203...	List Storage Ac...	Started	Administrative	Informational	2024-05-15T06:...
2274de78-8494...	Validate Deploy...	Started	Administrative	Informational	2024-05-10T07:...
f469ad59-1a5f...	'audit' Policy ac...	Succeeded	Policy	Warning	2024-05-10T06:...

00:00:04 Query executed successfully.