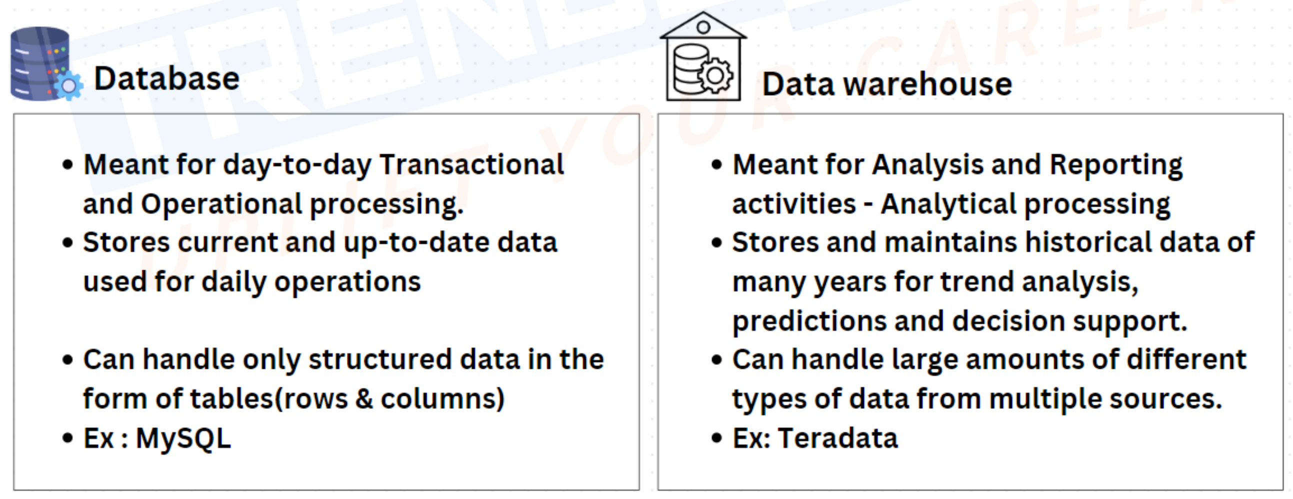
Azure Synapse Analytics

Need for Data Warehouse

- Databases are used for Day-to-day transactions
- Databases can be used for Reporting and Analysis tasks but not the preferred choice.

  Why are Databases not the preferred choice for performing analysis and reporting?

  1. Analysis of data would require a good amount of resources for performing data crunching activities which can overburden the database and slow down the day-to-day operations.
  2. Databases are structurally designed to handle predominantly day-to-day transactions efficiently.
  3. Data warehouse is specially built in terms of architecture and infrastructure to perform data analysis activities.
  4. Data Analysis involves handling huge volumes of historical data and storing this data in databases would be a big challenge. Data warehouses are a best fit to store historical data in a more cost-effective way.



Database

- Meant for day-to-day Transactional and Operational processing.
- Stores current and up-to-date data used for daily operations
- Can handle only structured data in the form of tables(rows & columns)
- Ex : MySQL

Data warehouse

- Meant for Analysis and Reporting activities - Analytical processing
- Stores and maintains historical data of many years for trend analysis, predictions and decision support.
- Can handle large amounts of different types of data from multiple sources.
- Ex: Teradata

Azure Synapse Analytics

- Is a Data warehouse optimized for Data Analysis. However, Synapse is used to perform several other operations making it much more than just a Data warehouse tool.
- Every azure synapse workspace should be associated with a storage account (like - ADLS Gen2)

Creating an Azure Synapse Analytics Workspace (Synapse Studio)

Step 1 : Create a Resource Group in the Azure Portal

Step 2 : Search for the service -> Azure Synapse Analytics -> Create Synapse Workspace.

Step 3 : Fill in the necessary fields like the Resource Group, Workspace Name, Region, Storage Account, File System Name(Container Name), Managed Resource Group where the tools used internally by Synapse are maintained.

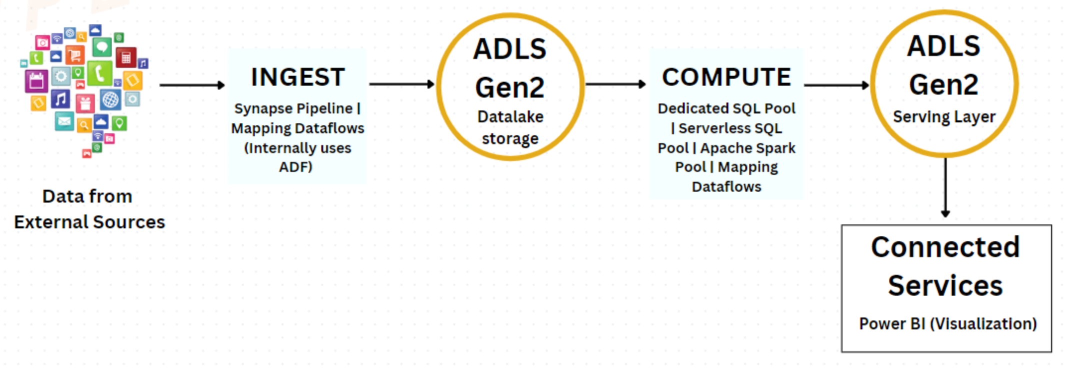
Step 4 : Tick the checkbox to assign the necessary permission for Synapse to access the storage account -> Review + Create.

Once the Synapse Workspace is deployed, launch the Synapse Studio that consists of the following options - **Home, Data, Develop, Integrate, Monitor & Manage.**

What is Azure Synapse Analytics ?

It is a Unified Analytics Service that brings together many capabilities like

1. Data Integration - Synapse Pipeline, Mapping Dataflow (Internally uses Data Factory to perform data ingestion)


Data from External Sources
INGEST
Synapse Pipeline | Mapping Dataflows (Internally uses ADF)
ADLS Gen2
Datalake storage
COMPUTE
Dedicated SQL Pool | Serverless SQL Pool | Apache Spark Pool | Mapping Dataflows
ADLS Gen2
Serving Layer
Connected Services
Power BI (Visualization)

2. Enterprise Level Data Warehousing
3. Big Data Analytics

3 Most Important Compute Services

1. Serverless SQL Pool (~ Athena in AWS)
2. Dedicated SQL Pool (~ Redshift in AWS)
3. Apache Spark Pool

Serverless SQL Pool

-Serverless SQL pool is a query service over the data in your data lake. It enables you to access your data through T-SQL support.

-Serverless SQL Pool doesn't have any dedicated storage provisioned in the beginning. Since it is serverless, there is no infrastructure to setup or cluster to maintain.

-Every Azure Synapse Analytics workspace comes with Serverless SQL Pool endpoints that you can use to query data in the Azure Data Lake.

Steps to work on data using Serverless SQL Pool

Step 1 - **Uploading a file using Synapse Studio to ADLS Gen2 container storage.** Since every Azure Synapse Workspace is hooked to a storage in datalake (like ADLS Gen2), by default a container is created in the storage account.

Step 2 - **Querying the file using OPENROWSET.** Right Click on the file reflecting in the Azure Synapse Workspace (stored in the ADLS Gen2 storage container) -> New SQL Script. A TSQL script gets created automatically.

Step 3 - Publish & Run the script to get the required results.

Important Note : The dataset file can be queried without having to create an entity like a table on-top of the data. This is achievable because of the **OPENROWSET** functionality.

External table Vs Normal/ Managed table

- External table is where the metadata is stored in the DWH and the data is stored externally in a Datalake storage account (Like - ADLS Gen2)
- Normal table is where the DWH stores both the metadata and the actual data.

In the case of Serverless SQL Pool, only external table creation is possible. Since the resources are not provisioned, there is no possibility of Normal table creation to store the data. Only compute power is made available on demand.

Creating an External Table in Azure Synapse Workspace

Step 1 : Create External Data Source (~ Linked Service)

Creating a Database

Creating Master Key Encryption

```
CREATE MASTER KEY ENCRYPTION BY PASSWORD =
‘<PASSWORD>’
```

Creating a Sas Token

```
CREATE DATABASE SCOPED CREDENTIALS SasToken
WITH IDENTITY = ‘SHARED ACCESS SIGNATURE’,
SECRET =
‘<Sas-token-generated-using-shared-access-signature-in-storage-account>’
```

Creating data source

```
CREATE EXTERNAL DATA SOURCE <datasource-name>
WITH ( LOCATION = ‘<path-to-data-container>’,
       CREDENTIAL = <SasToken>
)
```

Step 2 : Create File Format (~ Dataset)

```
CREATE EXTERNAL FILE FORMAT <file-format-name>
WITH ( FORMAT_TYPE = DELIMITEDTEXT,
        FORMAT_OPTIONS (
                FIELD_TERMINATOR = ‘,’,
                FIRST_ROW = 2)
)
```

Step 3 : Create External Table

```
CREATE EXTERNAL TABLE <table-name> (
<column_name_1> <data-type_1>,
<column_name_n> <data-type_n>
)
WITH (
LOCATION = ‘<path-to-data>’,
DATA_SOURCE = <datasource-name>,
FILE_FORMAT = <file-format-name>
)
GO
```

(Note : datasource-name and file-format-names as generated in the first two steps)

- We can execute queries on the external table created. The Monitor section provides information of the sql queries executed and the amount charged based on the data scanned.

Viewing the External Table created

- Once the external table is created, it will be listed in the **Data** section -> **Workspace** tab -> **External Table**
- To delete the external table : Right Click on the table -> New SQL Script -> execute the query **drop EXTERNAL TABLE <table-name>** (with this only the metadata/ table schema is deleted, the actual data is not deleted.
- To delete the Data Source and the File Formats : Right Click on the Data Source / File Format -> New SQL Script -> execute the query **drop EXTERNAL DATA SOURCE <datasource-name>**

  drop EXTERNAL FILE FORMAT <file-format-name>

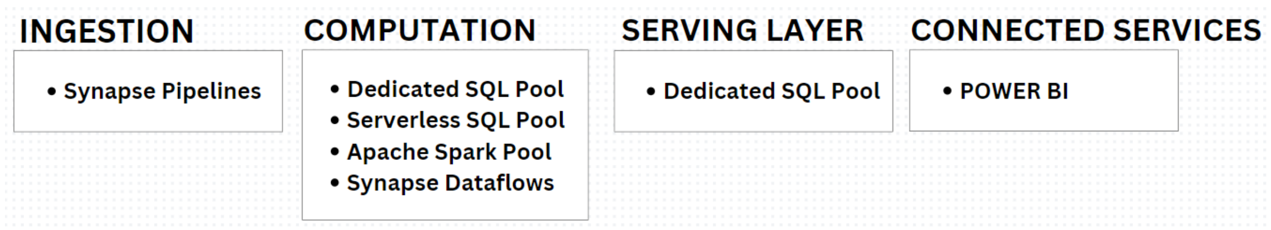
(Note : Shortcut to automatically generate the script to create an external table Right Click on the File -> New SQL Script -> Create External Table)

Use cases of Serverless SQL Pool

- For Adhoc Data Exploration.
- Logical Data Warehouse that requires Faster Processing without the requirement of infrastructure setup or resource provisioning.
- For Querying the data without the movement of data(No ETL) present in different sources using **Federated Query Execution** with T-SQL.
- For querying the data without having to create entities-tables on-top of data using **OPENROWSET**.
- For a highly Cost-effective way of processing data.

Key Points

- Azure Synapse is used to perform the following



INGESTION	COMPUTATION	SERVING LAYER	CONNECTED SERVICES
• Synapse Pipelines	• Dedicated SQL Pool • Serverless SQL Pool • Apache Spark Pool • Synapse Dataflows	• Dedicated SQL Pool	• POWER BI

- Two ways of querying the data present in Datalake (ADLS Gen2)

1. OPENROWSET
2. External Tables

- Use-cases of Serverless SQL Pool

1. Faster Data Exploration
2. Logical Data warehouse Creation
3. Cost-Effective

Data Lakehouse Architecture

Referred to as Modern Datawarehouse Architecture.

Datawarehouse

Advantages

- Consists of highly curated and structured data
- Involves a lot of policies and rules for data governance, ensuring that the data is governed well.
- High Security Aspects.

Challenges

- Only a small percentage of data is structured in the industry right now. There are huge volumes of Unstructured and Semi-structured data. Datawarehouses supports only structured data.
- Machine Learning algorithms are not supported in case of Datawarehouses.

DataLake

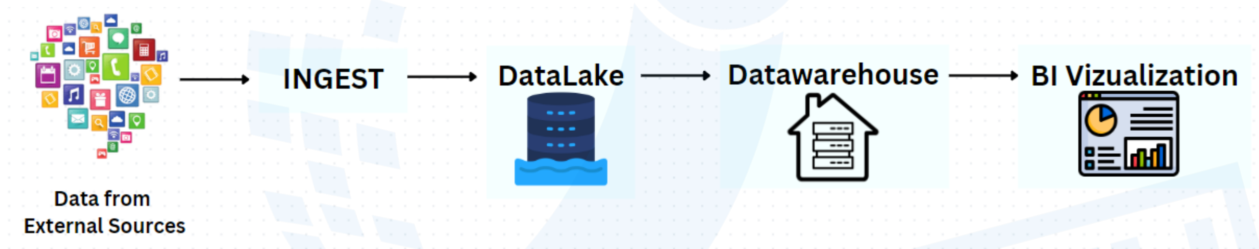
Advantages

- Can store data in open file formats (Parquet, ORC, Avro).
- Datalakes are highly scalable.
- Datalakes are an inexpensive solution for storing huge volumes of data.
- Support Machine Learning and Data Science.

Challenges

- Data Governance and reliability is compromised.
- Granular level of access control cannot be set.
- Not suitable for BI and Visualization workloads.

Better Solution is a combination of DATALAKE + DATAWAREHOUSE



Data Lakehouse

Best of Datalake and data warehouse as a single component.

Objectives / Requirements of an ideal Data Lakehouse

- Quick Data Discovery
- Should handle all types of data (Structured, Semi-strucutred, Unstructured)
- Reduce the ETL activity
- There shouldn't be multiple copies.
- Data should be in open file format.
- Storage and compute should be decoupled for scalability.
- Integrated security and governance
- Should be able to handle BI, ML and other usecases.
- Should be cost effective.
- Should support ACID transactions (delta format)

- Single solution which can handle computational needs and can be acting as a serving layer.

[Companies that are trying to build an ideal Datalakehouse - Databricks, Snowflake, Redshift, Synapse, Google Big Query]

Dedicated SQL Pool

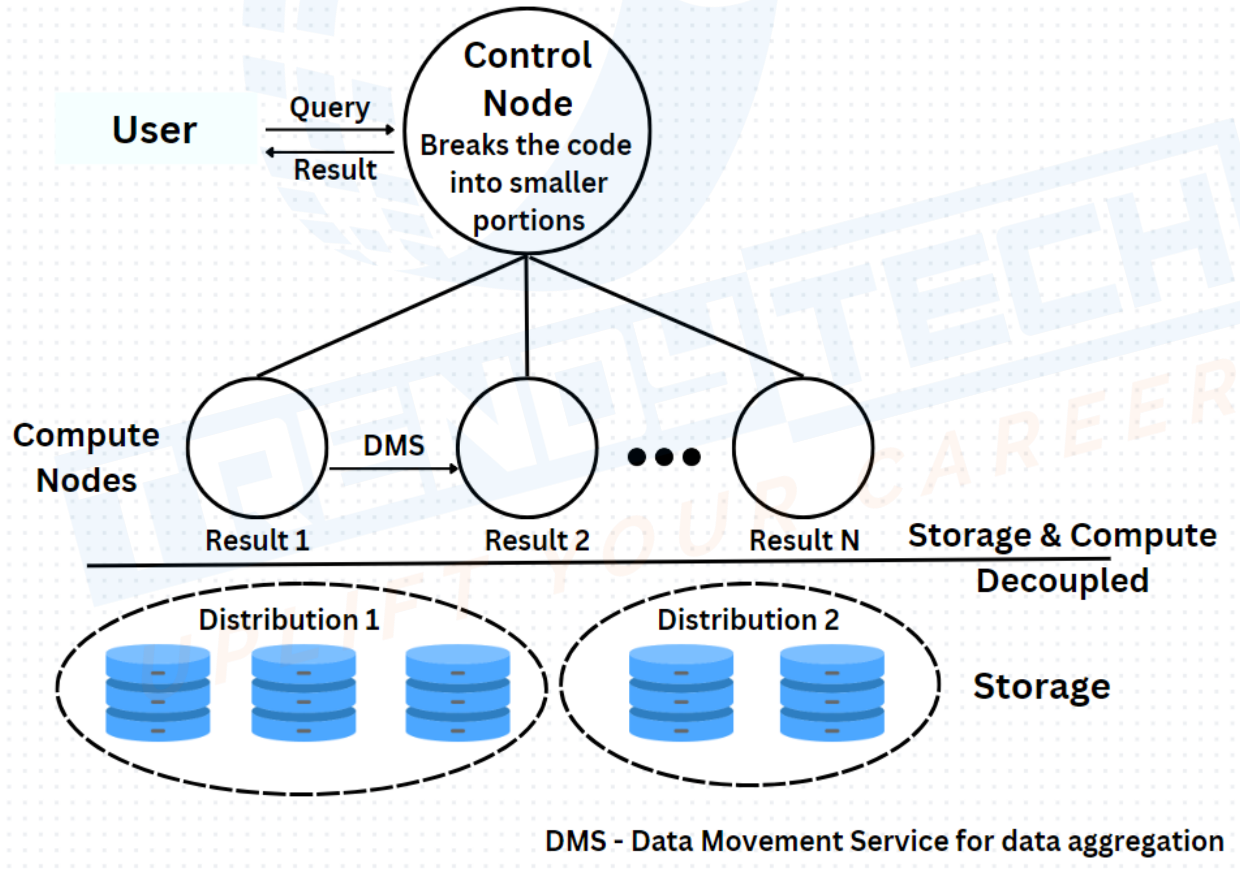
Was also known as a SQL Datawarehouse.

- Internally, a dedicated SQL pool uses a distributed query engine.

  Use-cases of dedicated SQL pool - It can be used as both a processing / compute layer and as a serving layer.

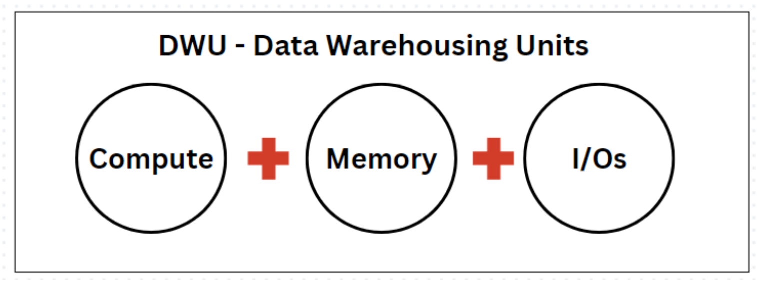
- It internally uses a **MPP Engine (Massively Parallel Processing)**

Architecture


User
Query
Result
Control Node
Breaks the code into smaller portions
Compute Nodes
DMS
Result 1
Result 2
Result N
Storage & Compute Decoupled
Distribution 1
Distribution 2
Storage
DMS - Data Movement Service for data aggregation

- Dedicated datawarehouse is an expensive service.
- The architecture of a Dedicated SQL Pool : One Control Node and multiple Compute Nodes

- The underlying data is distributed in various distributions (default : 60 distributions)



DW100C - 1 compute node, 60 GB RAM

DW200C - 1 compute node, 120 GB RAM

DW300C - 1 compute node,180 GB RAM

DW400C - 1 compute node, 240 GB RAM

DW500C - 1 compute node, 300 GB RAM

DW1000C - 2 compute node, 600 GB RAM

DW2000C - 3 compute node, 1200 GB RAM

DW30000C - 60 compute node, 18000 GB RAM

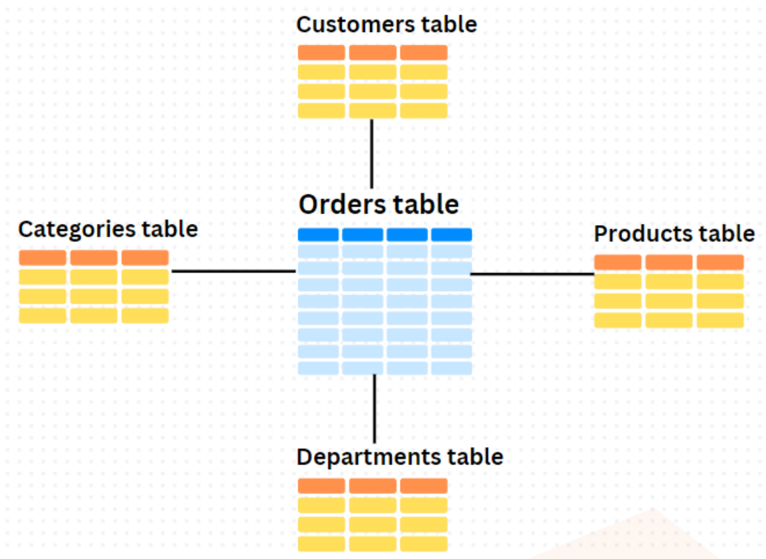
- Maximum parallelism is achieved with 60 compute nodes handling 1 distribution each.
- Dedicated SQL Pool is more like a traditional datawarehouse. Modelled as Fact and Dimension tables

Ex :

Orders table - the Fact table as it is a huge table that keeps on increasing

Customers and Products tables - the supporting small Dimension tables enriching the Fact table.

- Star Schema



No concept of Primary and Foreign keys

Table Distributions

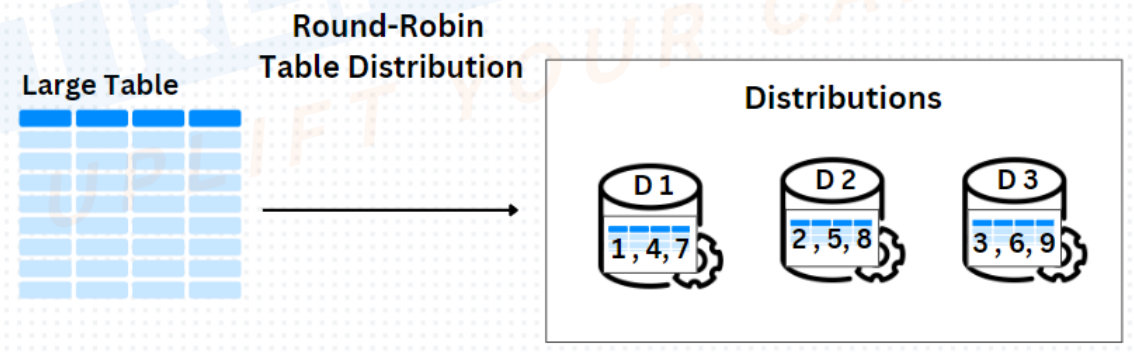
There are 3 types of table distributions

1. **Round Robin :**

   -Distribution of data is easy. It is the default distribution style.

   -But running the query requires shuffling if aggregations / joins are required as the data is not co-located.

   -Used in cases of Staging tables / Intermediate data.



2. **Hash :**

-Distributing the data can be slower in case of hash table distribution style.

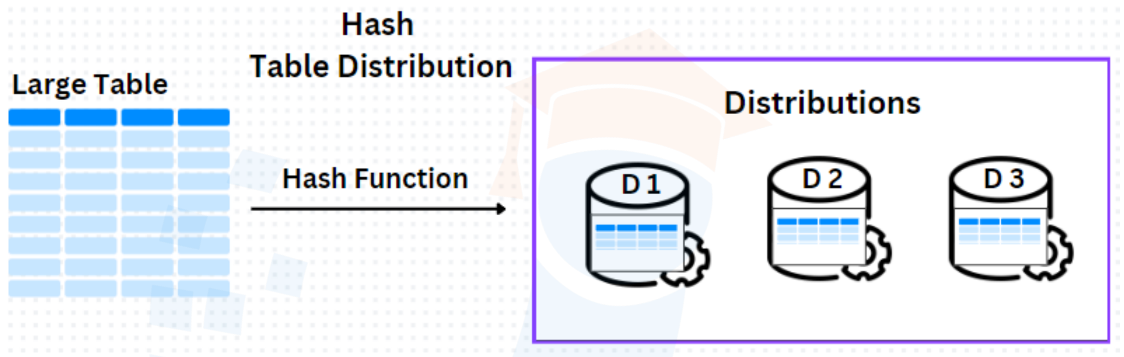
-Same keys go to the same distribution (deterministic hash)

-However, running queries are faster due to co-located data.

-hash(<column-name>)

Column name on which you want to hash.

Ex : hash(customer_id)



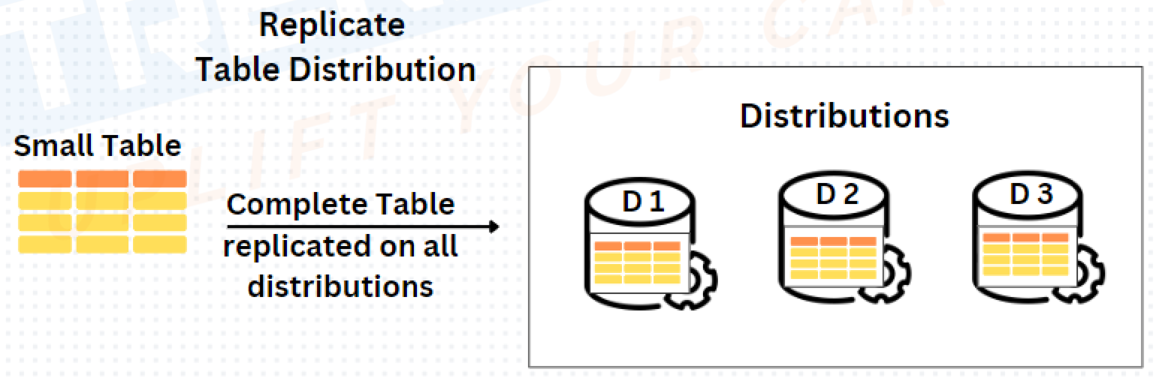
3. **Replicate :**

-Replicate is used for small dimension tables

-It is similar to Broadcast / Mapside join

-The smaller dimension table is distributed on all the distributions

-No shuffling is needed incase of Replicate



Example Use-case

1. Upload the data in ADLS Gen2 (orders)
2. Create an external table in the Dedicated SQL Pool.
3. Load the data into the external table from the Datalake.

2 ways to load the data from Datalake to Dedicated SQL Pool

a. **Polybase**

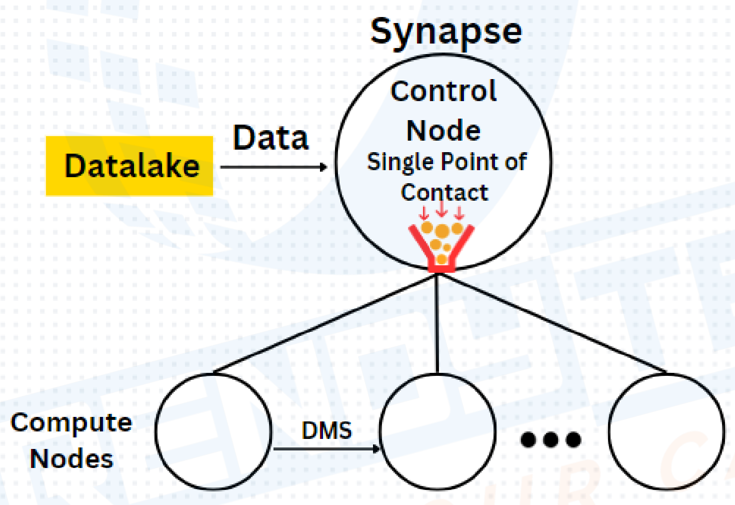
-Uses MPP(Massive Parallel Processing) Architecture

-In case of polybase, we can move data in 2 ways :

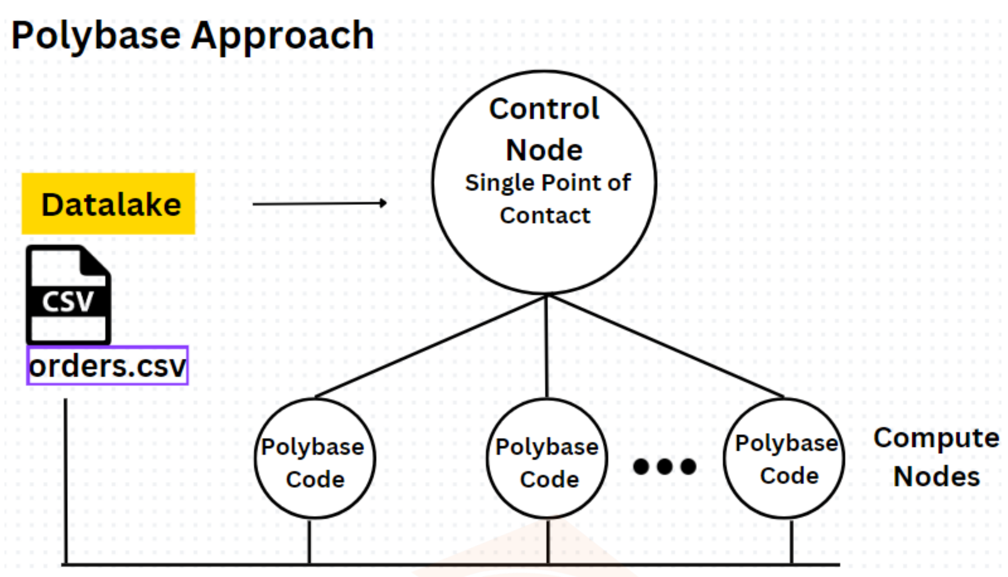
Dedicated SQL Pool -> DataLake (CETAS - Create External Table AS)

Datalake -> Dedicated SQL Pool (CTAS - Create Table AS)

Problem



Solution : Control node gives the control to Compute nodes to directly interact with the data files



STEPS

1. Create an external table (orders) in the dedicated SQL Pool

Data Source -> External File Format -> External Table

2. CTAS

```
create table internal_table as
select * from external_table
```

Example : Using Round Robin Distribution

```
CREATE table orders_internal_roundrobin
WITH
(
        DISTRIBUTION = ROUND_ROBIN
)
AS
select * from orders_ext
```

Checking the statistics on how data is distributed

```
DBCC PDW_SHOWSPACEUSED('orders_internal_roundrobin')
```

Example : Using Hash Distribution

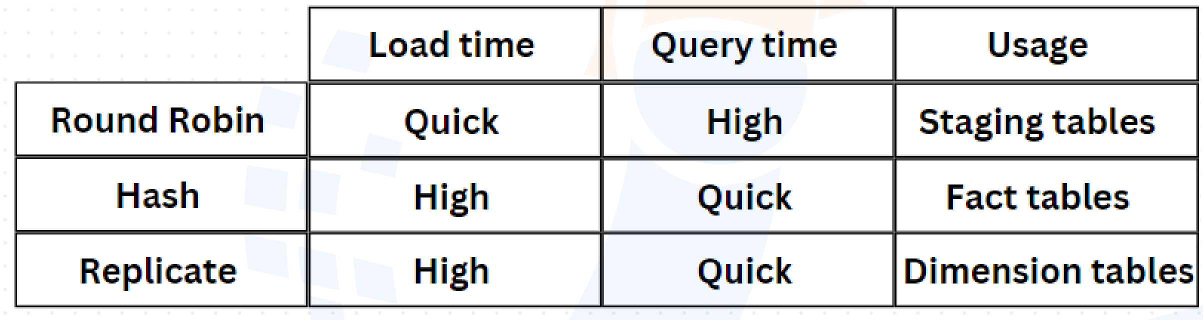
```
CREATE table orders_internal_hash
```

```
WITH
(
        DISTRIBUTION = HASH
)
AS
select * from orders_ext
```

Checking the statistics on how data is distributed

```
DBCC PDW_SHOWSPACEUSED(‘orders_internal_hash’)
```

[In case of Hash, you can see that there is no shuffling of data involved as the data is colocated.]



	Load time	Query time	Usage
Round Robin	Quick	High	Staging tables
Hash	High	Quick	Fact tables
Replicate	High	Quick	Dimension tables

b. **Copy Command**

-Uses MPP(Massive Parallel Processing) Architecture.

-Can Perform better than Polybase

-No external objects required

-Wild card characters are allowed in the file path

Spark Pool

- Majorly used for computation and is very fast due to in-memory computation.
- Spark pool is the best choice for complex computations.

Creating an Apache Spark Pool :

In the Synapse Workspace -> Manage section -> Apache Spark Pools -> New (Enter the Name -> Enter the Number of Nodes -> Enter the Node Size) -> default settings -> Review&Create

Spark Table

- Table comprises of **Data** & **Metadata**
- Since Spark is purely a compute engine, it doesn't handle both data and metadata.
- Command to create a Spark Table

```
%%pyspark
df = spark.read.load('<adls-storage-file-path>', format = 'csv', header =
True
)
df.write.mode("overwrite").saveAsTable("<table-name>")
```

- Spark Tables are created under the **Lake database** in the synapse studio (Data -> Workspace)

Reading / Processing the data present in Dedicated SQL Pool using Spark :

Data section(In Synapse Studio) -> Workspace -> SQL Database -> retaildb (dedicated SQL Pool) -> Tables -> select table (ex: dbo.orders) -> Right click -> New Notebook -> Load to Dataframe -> Basic template code generated in the Notebook ->

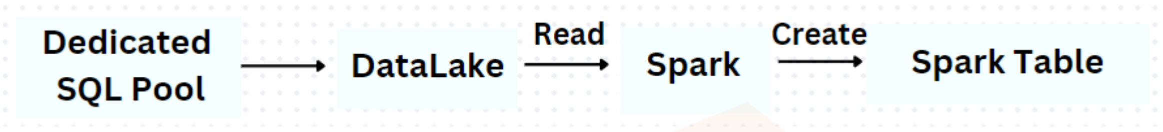
```
%%spark
val df = spark.read.sqlanalytics("retaildb.dbo.orders")
df.write.mode("overwrite").saveAsTable("default.t1")
```

Key Points :

- **Taking data from dedicated SQL Pool -> Create a Spark Dataframe and loading the data into the Dataframe -> Performing some processing -> Creating the Spark table on this processed Data**

- **Spark cluster takes the data from a dedicated SQL Pool using the polybase approach.**

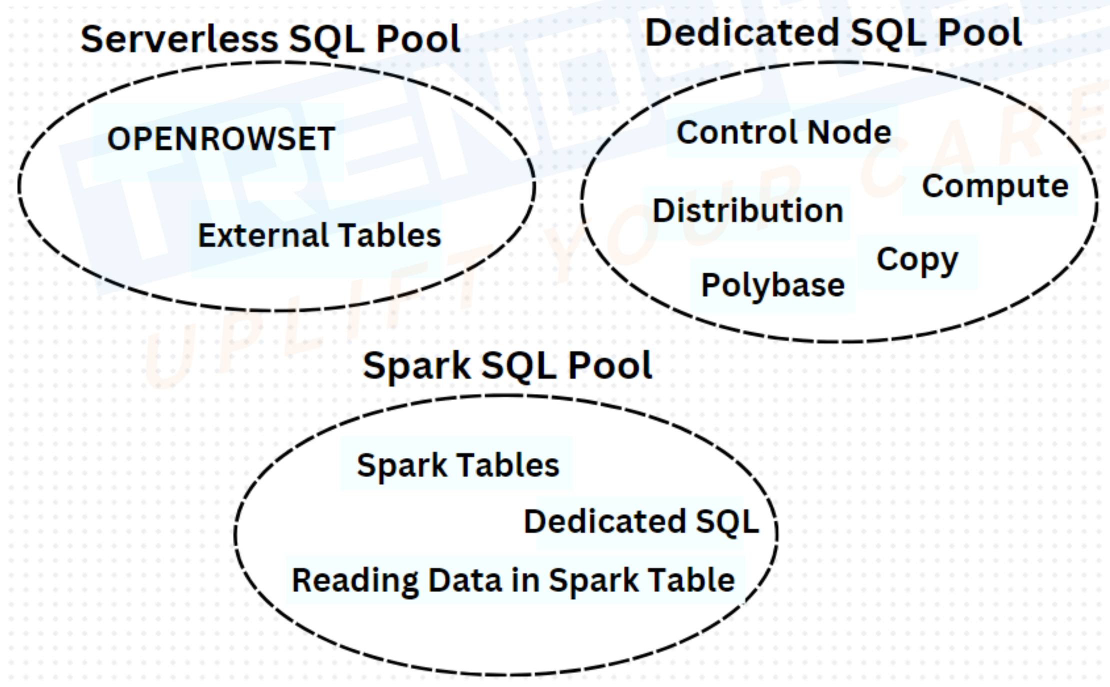
  Driver requests the control node for Data
  |
  Control Node passes the instructions to the compute node
  |
  Compute Node writes the data in parallel (MPP) to Datalake
  |
  Spark fetches the data from the Datalake



Reading / Processing Spark Table through Serverless SQL Pool :

- Serverless SQL Pool cannot access the hive metastore for metadata.
- Therefore a copy of this metadata is created at the background (synchronous activity) when a Spark table is created.
- Spark Tables are present under the Lake Database
- Select a Spark table -> Right Click -> New SQL Script -> select top 100 -> connects to a built-in serverless SQL Pool

Synapse Summary :


Serverless SQL Pool
OPENROWSET
External Tables
Dedicated SQL Pool
Control Node
Compute
Distribution
Copy
Polybase
Spark SQL Pool
Spark Tables
Dedicated SQL
Reading Data in Spark Table