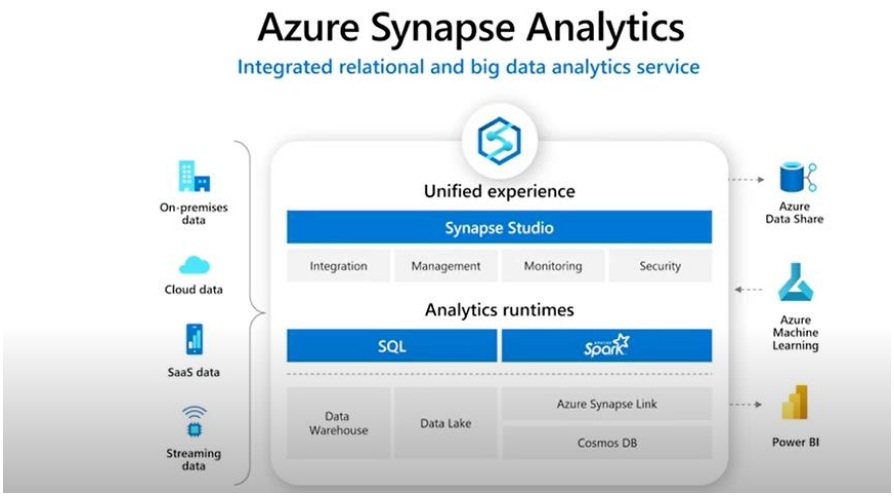
# Azure Synapse Analytics

**What is Azure Synapse Analytics?**

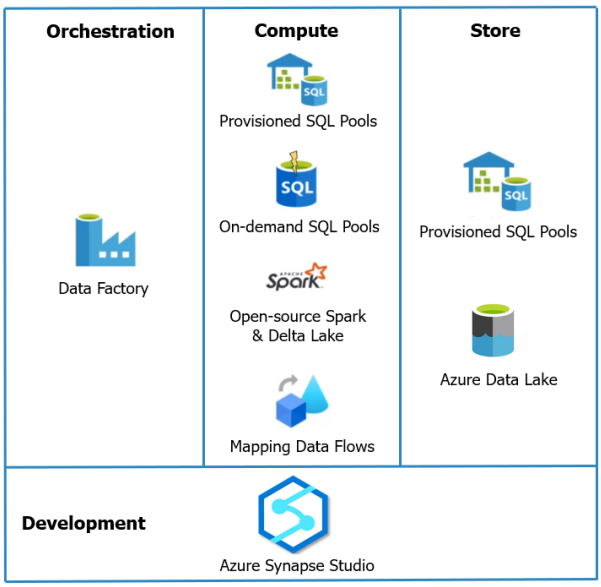
Azure Synapse Analytics is an integrated analytics platform, which combines **data warehousing, big data analytics, data integration**, and **visualization** into a single environment.



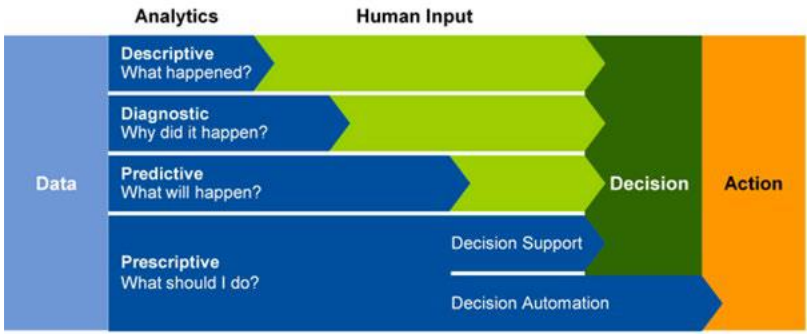

- Azure Synapse Analytics brings together **data warehouse** and **big data analytics** and **Data integration** into a single and unified space workspace.
- It allows customers to build end-to-end analytics solutions and **perform data ingestion, data exploration, data warehousing, big data analytics, and machine learning tasks from a single, unified environment**.
- The advantage of having a single integrated data service is that, for enterprises, it accelerates the delivery of BI, AI, machine learning, Internet of Things, and intelligent applications and Data professionals of all types can collaborate, manage, and analyse their most important data efficiently—all within the same service
- Azure Synapse Analytics is **deeply integrated with Power BI and Azure Machine Learning** to greatly expand the discovery of insights from all your data and apply machine learning models to all your intelligent apps.
- It provides deep integration of Apace spark and SQL Engine.
- **Synapse SQL** is a distributed query system for T-SQL and offers **serverless** and **dedicated** resource models
- **Apache Spark for Azure Synapse is** used for data preparation, data engineering, ETL, and machine learning.

- **Data Integration engine** provides experiences as Azure Data Factory, allowing you to create rich at-scale ETL pipelines without leaving Azure Synapse Analytics.
- It provides **Unified management, monitoring, and security**.

Big Data Services By Synapse:




**Type of Analytics supported by Synapse:**




**Descriptive analytics:**

**"What is happening in my business?"**

- Azure Synapse Analytics leverages the **dedicated SQL pool capability** (Data Warehouse) that enables you to create a persisted data warehouse to perform this type of analysis.
- You can also make use of the **serverless SQL pool** to prepare data from files stored in a data lake to create a data warehouse interactively.

**Diagnostic analytics**

**"Why is it happening?".**

- This may involve exploring information that already exists in a data warehouse. But more wider data exploration can be done by interactively explore data within a data lake.
- **Serverless SQL pools** can quickly enable a user to search for additional data.

**Predictive analytics**

**"What is likely to happen in the future based on previous trends and patterns.**

- Azure Synapse Spark pools can be used with other services such as Azure Machine Learning Services, or Azure Databricks, enables you to answer the question

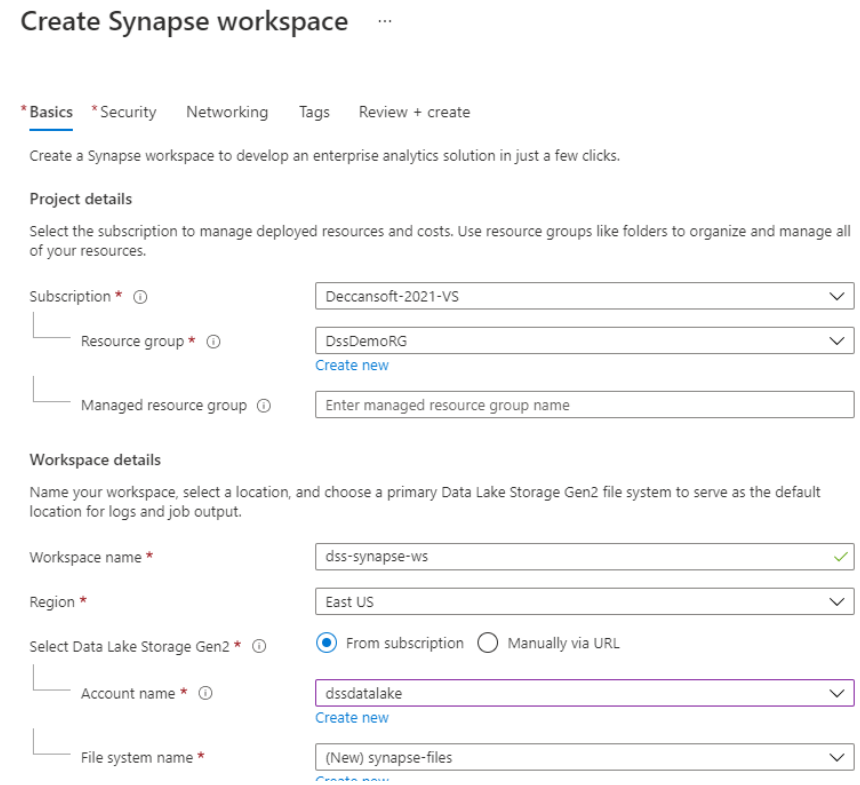
**Prescriptive analytics**

- This type of analytics looks at **executing actions based on real-time** or near real-time analysis of data, using predictive analytics.
- Azure Synapse Analytics provides this capability through both Apache Spark, **Azure Synapse Link**, and by integrating streaming technologies such as **Azure Stream Analytics**.
- Azure Synapse Analytics brings these two worlds together with a unified data integration experience to ingest, prepare, manage, and serve data using Azure Synapse Pipelines. In addition, you can visualize the data in the form of dashboards and reports for immediate analysis using Power BI, which is integrated into the service too.

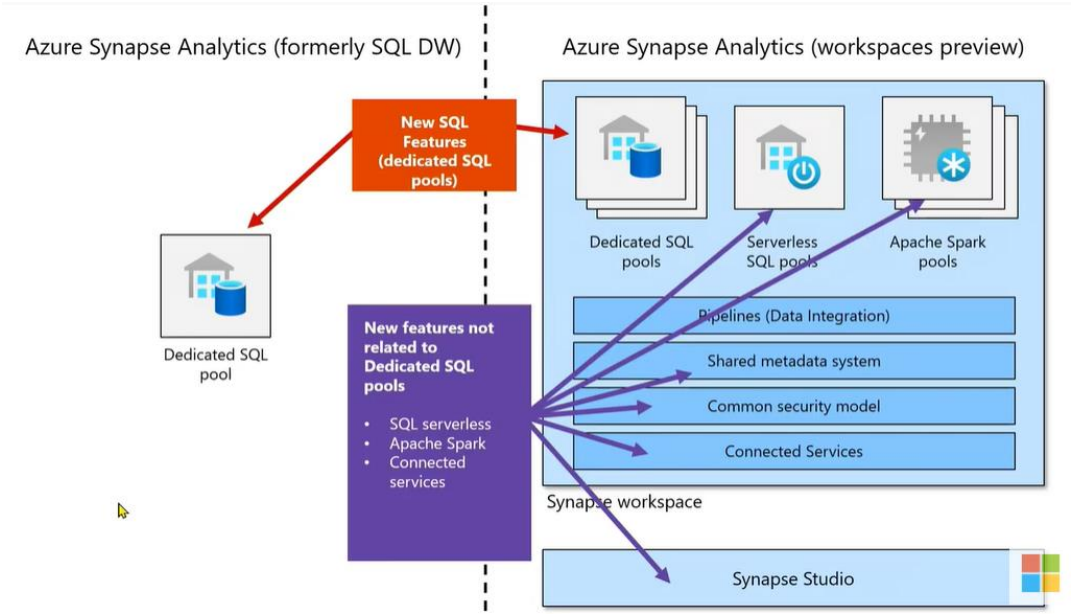
## Azure Synapse Analytics workspace

**Lab 1: Create Synapse Analytics Workspace**

Search→synapse→Azure synapse Analytics→Create→




Security Tab→Provide password for administrator access to the workspace's SQL pools.




- A workspace is the top-level resource and comprises your analytics solution
- Synapse SQL offers both **serverless** and **dedicated** resource models. Both supports Data Warehousing and Data Lake
- It has one default serverless SQL Pool which maps to distributed query service.
- There can be any number of dedicated SQL Pools and any number of Apache Spark Pools
- Pipeline Provides Data integration, Orchestration and Data Movement.

- Shared metadata system makes it easy to share tabular data between SQL and Spark.
- Entire workspace, all resources, all pools are governed by common security model, which makes it easy to manage.
- There are series of connected services which expands the reach of synapse in other services.
- Synapse Studio is one stop shop for data engineers to code, monitor, manage, debug, secure.

**Dedicated SQL Pool:**

- Dedicated SQL pool (formerly SQL DW) represents a collection of analytic resources that are provisioned when using Synapse SQL.
- The size of a dedicated SQL pool is determined by Data Warehousing Units (DWU).
- Dedicated SQL pool uses PolyBase to query the big data stores. PolyBase uses standard T-SQL queries to bring the data into dedicated SQL pool (formerly SQL DW) tables.
- Dedicated SQL pool (formerly SQL DW) stores data in relational tables with columnar storage.

**Serverless SQL Pool:**

- Serverless SQL pool is a query service over the data in your data lake.
- Serverless SQL pool is a distributed data processing system, built for large-scale data and computational functions. Serverless SQL pool enables you to analyse your Big Data in seconds to minutes, depending on the workload.
- Serverless SQL pool is serverless, hence there's no infrastructure to setup or clusters to maintain.
- There is no charge for resources reserved, you are only being charged for the data processed by queries you run, hence this model is a true pay-per-use model.
- You can use following tools for querying Data: Azure Synapse Studio, Azure Data Studio, SSMS

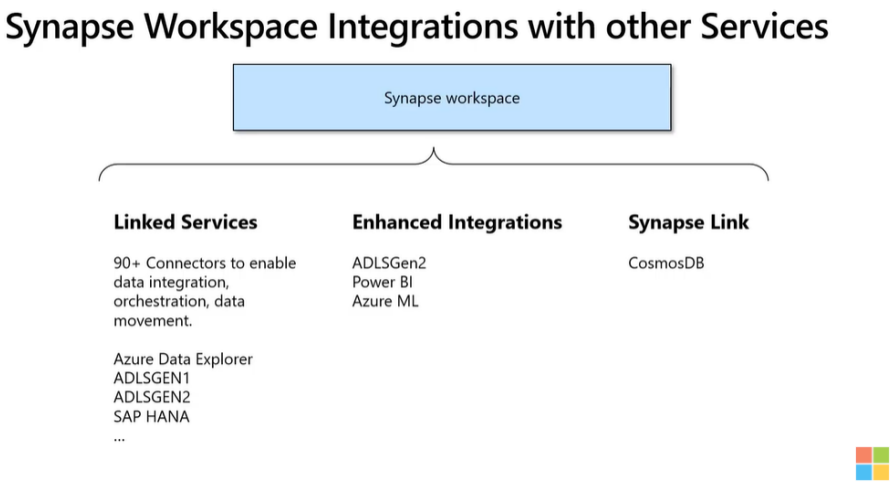
**Note:**

*For predictable performance and cost, create dedicated SQL pools to reserve processing power for data stored in SQL tables.*

*For unplanned or ad-hoc workloads, use the always-available, serverless SQL endpoint.*

**Spark Pool:**

- Spark pools in Azure Synapse offer a fully managed Spark service.
- Apache Spark is a parallel processing framework that supports in-memory processing to boost the performance of big-data analytic applications.
- Apache Spark pool includes many language (Scala, Python, SparkSQL, and C#) features to support preparation and processing of large volumes of data so that it can be made more valuable and then consumed by other services within Azure Synapse Analytics.
- For machine learning workloads, you can use SparkML algorithms and AzureML integration for Apache Spark 2.4 with built-in support for Linux Foundation Delta Lake.




**Azure Synapse Pipelines:**

- Azure Synapse Pipelines **leverages the capabilities of Azure Data Factory** and is the cloud-based ETL and data integration service that allows you to create data-driven workflows for orchestrating data movement and transforming data at scale.
- Using Azure Synapse Pipelines, you can create and schedule data-driven workflows (called pipelines) that can ingest data from disparate data stores. You can build complex ETL processes that **transform data visually with data flows** or by using compute services such as Azure Databricks.

**Azure Synapse Link:**

**(Perform operational analytics with near real-time hybrid transactional and analytical processing)**

- Azure Synapse Analytics enables you to reach out to operational data using Azure Synapse Link, and is achieved without impacting the performance of the transactional data store. For

this to happen, you have to enable the feature within both Azure Synapse Analytics, and within the data store to which Azure

- Synapse Analytics will connect, such as Azure Cosmos DB.
- In the case of Azure Cosmos DB, this will create an analytical data store. As data changes in the transactional system, the changed data is fed to the analytical store in a Column store format from which Azure Synapse Link can query with no disruption to the source system

**Azure Synapse Studio:**

- Synapse Studio features a user-friendly, **web-based interface** that provides an integrated workspace and development experience.
- This allows data engineers to build end-to-end analytics solutions (ingest, explore, prepare, orchestrate, visualize) by performing everything they need within a single environment

## Explore Azure Synapse Studio

**Lab2 :Launch and Explore synapse Studio:**

Option 1: Workspace→Open Synapse Studio→Open

Option 2:https://web.azuresynapse.net/

Azure Synapse Studio is a single web UI that allows you to:

- Explore your data estate.
- Develop TSQL scripts and notebooks to interact with the analytical engines.
- Build data integration pipelines for managing data movement.
- Monitor the workloads within the service.
- Manage the components of the service.

**Data:**

- You can create Database and Database objects
- You can create linked Database to access data from external Repositories.
- By default, the Azure Data Lake Storage Gen2 account, which is provided during the creation of the Synapse workspace is linked and shown here.

- Based on the repository, different options can be seen on the toolbar like creating a new SQL script, new notebook, new data flow, new dataset, as well as file-based operations like creating or deleting a new file or folder

**Develop:**

- It provides options to create new artifacts like SQL script, Notebook, Data flow, etc.

**Integrate:**

- We can create data pipelines, jump directly to the Copy tool which allows us to create data pipelines step by step using a wizard, or browse a gallery of samples or previously created data pipelines to reuse the same for integrating data.

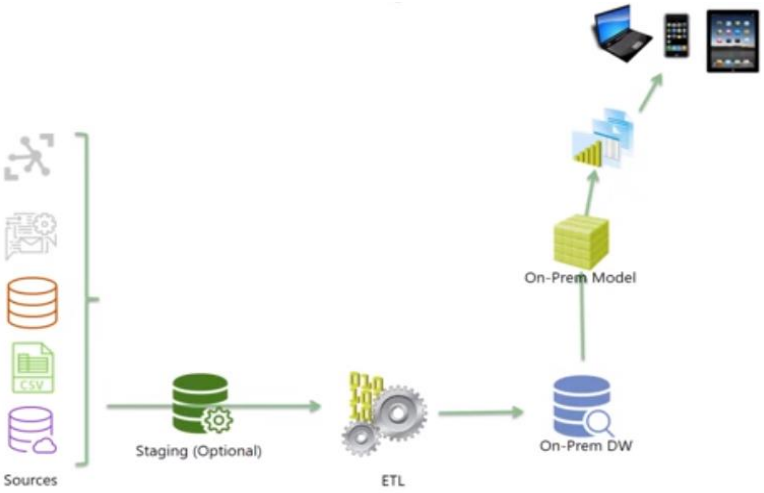
**Monitor:**

- Synapse Studio is not only a developer console but also an administrative console as well. With Monitor view you can monitor the pipeline executions, triggers that initiated a pipeline execution, and different integration runtimes.
- It also provides different options to monitor spark applications and those job executions that are generated from those applications, ad-hoc SQL queries or requests that are executed, as well as options to debug a data flow as well.

**Manage:**

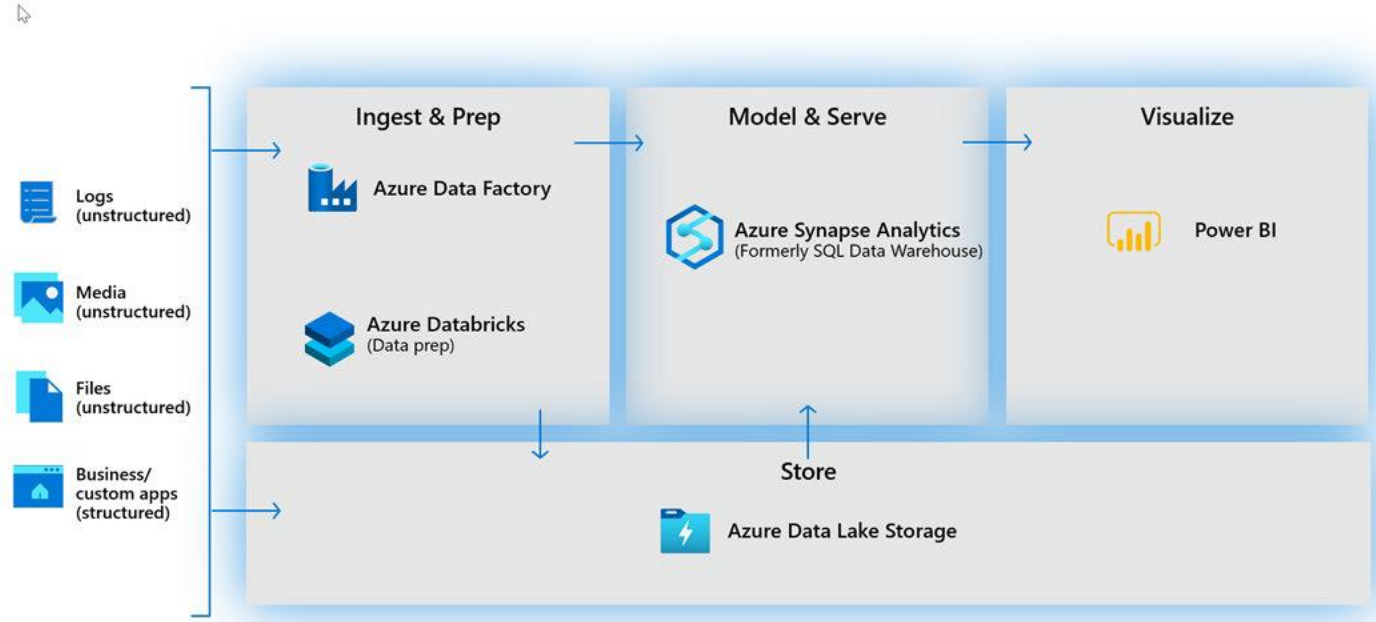
- In Analytics pools section, you can see built in serverless pool and create new SQL Pool.
- One can create linked services to register external data repositories in the external connections section.
- In the Integration section triggers and Integration runtime can be registered.
- In the Security section, one can configure access control to this environment to different users and group, modify the credentials that we configured for administrative access, and manage any private endpoints for secure network connectivity (if any).

**Traditional Data Warehousing:**




Modern Data Warehouse Architecture

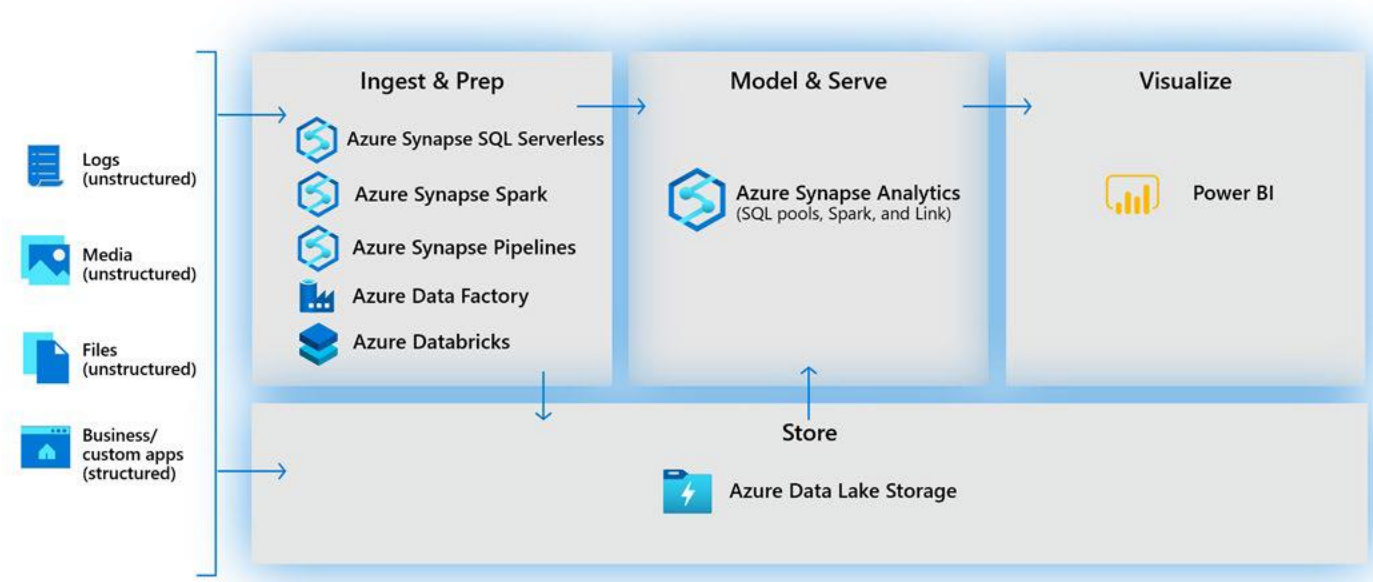
The process of building a modern data warehouse typically consists of:




1. Data Ingestion and Preparation.
    - Whether the data is an on-premises data sources, other Azure services, or other cloud services, customers can seamlessly author, monitor, and manage their big data pipelines with a visual environment that is easy to use.
    - Another option for data preparation is Azure Databricks - to shape the data formats and prep it using a Notebook making internal collaboration on data more streamlined and efficient
2. Making the data ready for consumption by analytical tools.
    - Data Analytics.

3. Providing access to the data, in a shaped format so that it can easily be consumed by data visualization tools.
    - Power BI Supports enormous set of data sources.
    - It can be used to build visualizations on massive amounts of data.
    - Data can be queried live or can be modelled and ingested for analysis and visualization
    - it's a powerful tool to build and deploy dashboards in the enterprise, through rich visualizations, and features like natural language querying

**Modern Datawarehouse using Synapse Analytics:**

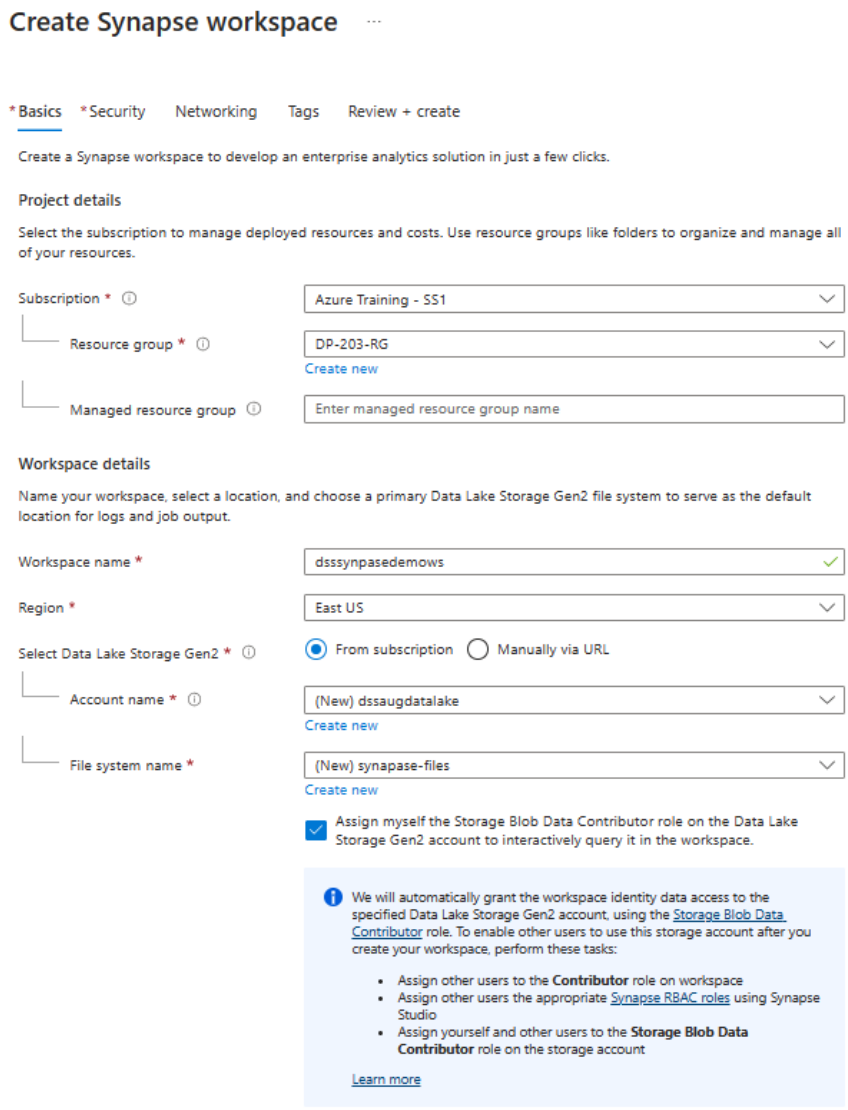



**Lab: Explore Azure Synapase Analytics**

**Prerequisite:**In Blob storage create container "orders" and Load file orders-2016.txt

**Provision Synapase Workspace**

1. Azure Portal→+Create Resource→Search(Azure Synapase Analytics)→Create synapse workspace




2. **For Data Lake Storage Gen2**

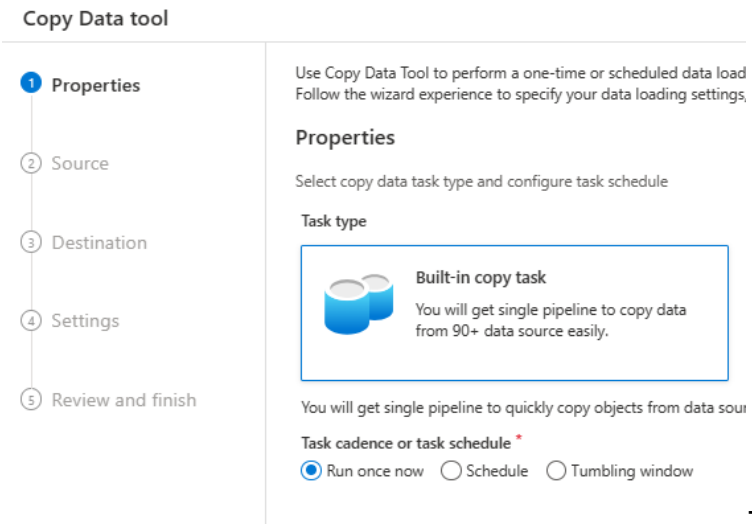
   Account name: Either use existing data lake account OR Create new

   File system name:Create new

3. Review+Create

**Ingest Orders data from blob to Data Lake Account**

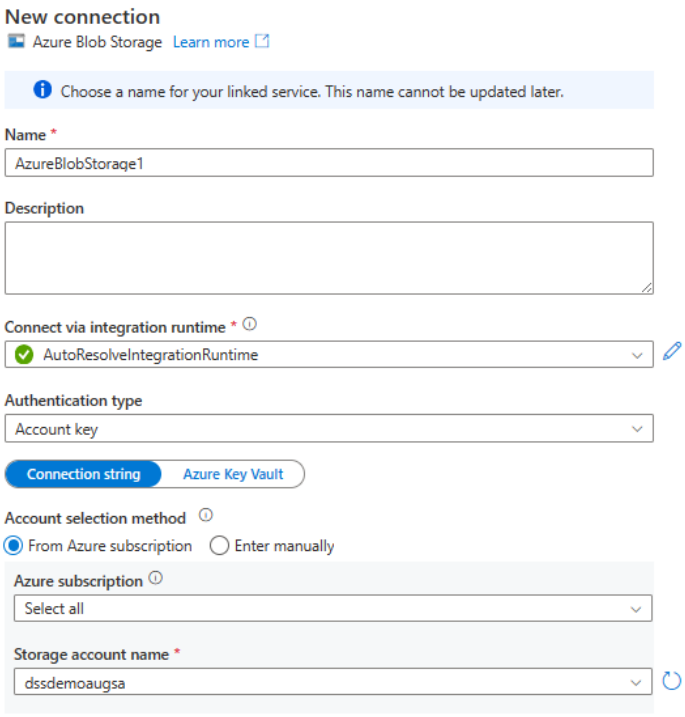
1. Go to your Synapase workspace→Overview→Open Synapase Studio→Open
2. **Synapase studio→Home** page→ select **Ingest**




→Next

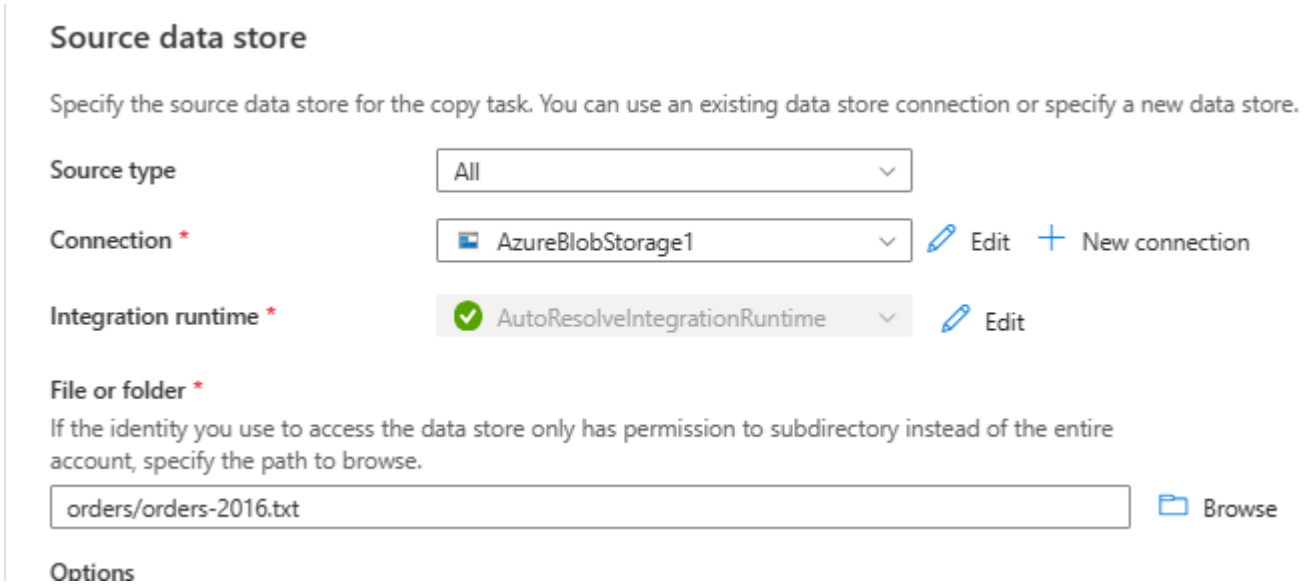
3. **Source** step→ **Dataset** substep, select the following settings:

- **Source type**: All
- **Connection**: *+New Connection →Select Azure blob storage →Continue*




→Test Connction and Create

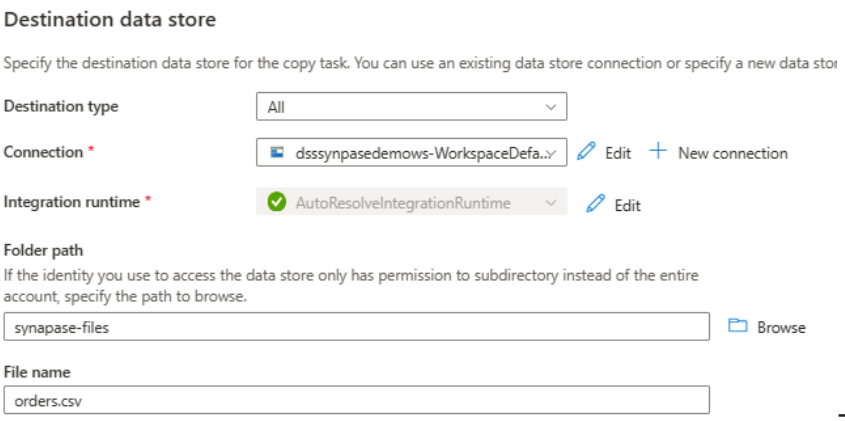
4. Browse and select your orders2016.txt file



→Next→Preview data→Next

5. Destination step→ **Dataset** substep, select the following settings

- **Destination type**: All
- **Connection**: *Select existing workspace connection*



→Next→Next→Next

6. Select Monitor and observe the pipeline status as Succeded

**Use Serverless Pool to Analyse Data**

1. Go to Data Hub→Linked section and observ newly copied orders.csv file→Right click→New SQL script→Select TOP 100 rows
2. Modify the script with HEADER_ROW = TRUE as shown below so that the result set shows column names also

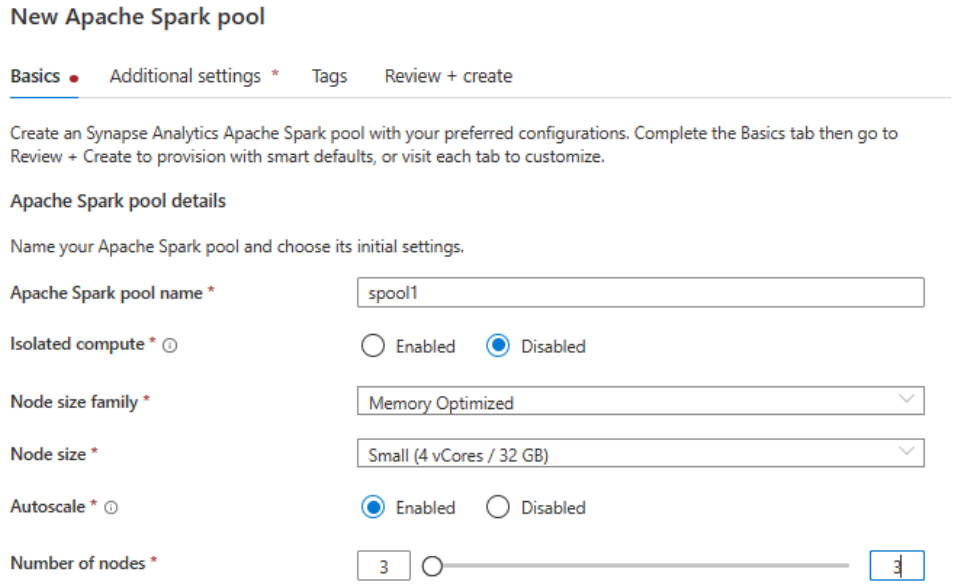
```
SELECT
    TOP 100 *
```

```
FROM
    OPENROWSET(
        BULK 'https://dssaugdatalake.dfs.core.windows.net/synapase-
files/orders.csv',
        FORMAT = 'CSV',
        PARSER_VERSION = '2.0',
        HEADER_ROW = TRUE
    ) AS [result]
```

3. You can also use Chart view

**Use Spark Pool to Analyse Data**

1. Manage Hub→Apache spark Pool→+New




**→Review+Create→Create**

2. Data Page→Go to orders.csv→Right click→New Notebook→Load to data frame→
3. Observer the Notebook cell

```
%%pyspark
df = spark.read.load('abfss://synapase-
files@dssaugdatalake.dfs.core.windows.net/orders.csv', format='csv'
## If header exists uncomment line below
##, header=True
)
display(df.limit(10))
```

4. Attach notebook to your spark pool→Run
5. Uncomment header=True→Run cell again
6. Get number of orders placed by Customer

```
from pyspark.sql.functions import countDistinct
```

```
df_counts = df.groupBy(df.CustomerName).agg(countDistinct('OrderID'))
display(df_counts.limit(30))
```