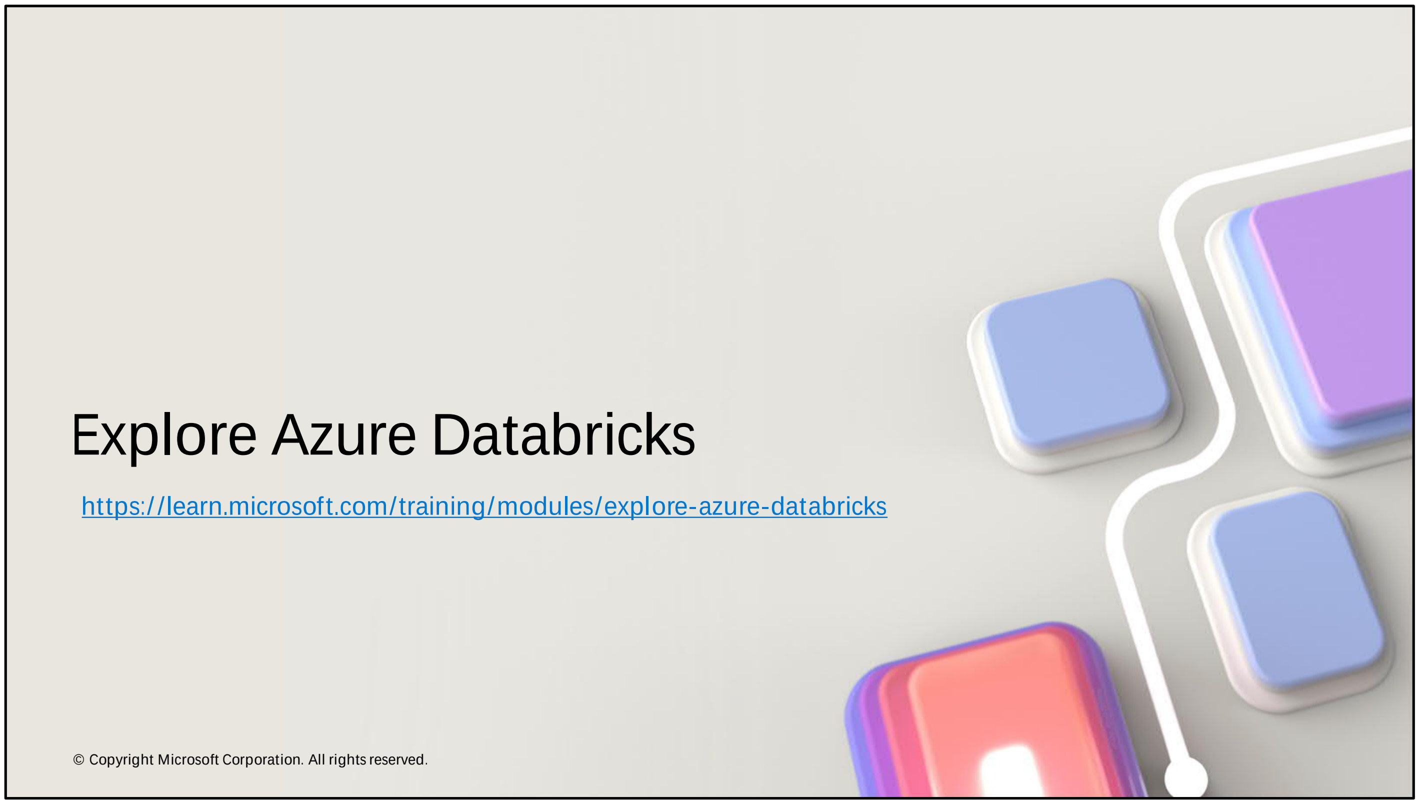
Microsoft

Implementing a Data Analytics Solution with Azure Databricks

Course timing note for instructors

- Explore Azure Databricks
- Use Apache Spark in Azure Databricks
- Use Delta Lake in Azure Databricks
- Use SQL Warehouses in Azure Databricks



Explore Azure Databricks

https://learn.microsoft.com/training/modules/explore-azure-databricks

Azure Databricks is a fully managed, cloud-based data analytics platform, which empowers developers to accelerate AI and innovation by simplifying the process of building enterprise-grade data applications. Built as a joint effort by Microsoft and the team that started Apache Spark, Azure Databricks provides data science, engineering, and analytical teams with a single platform for big data processing and machine learning.

By combining the power of Databricks, an end-to-end, managed Apache Spark platform optimized for the cloud, with the enterprise scale and security of Microsoft's Azure platform, Azure Databricks makes it simple to run large-scale Spark workloads.

Learning objectives

In this module, you'll learn how to:

Provision an Azure Databricks workspace.

Identify core workloads and personas for Azure Databricks.

Describe key concepts of an Azure Databricks solution.

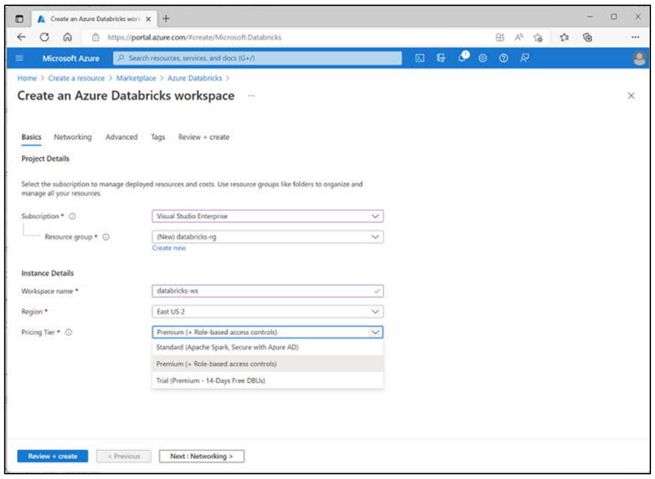
Prerequisites

Before starting this module, you should have a fundamental knowledge of data analytics concepts.

Provision an Azure Databricks workspace

Choose an appropriate tier:

- Standard - Core Apache Spark capabilities with Azure authentication
- Premium - Role-based access controls and other enterprise-level features
- Trial - A 14-day free trial of a premium-level workspace


Create an Azure Databricks wor
https://portal.azure.com/#create/Microsoft.Databricks
Microsoft Azure
Search resources, services, and docs (G+/)
Home > Create a resource > Marketplace > Azure Databricks >

Create an Azure Databricks workspace

Basics | Networking | Advanced | Tags | Review + create

Project Details

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription *: Visual Studio Enterprise

Resource group *: (New) databricks-rg

Create new

Instance Details

Workspace name *: databricks-ws

Region *: East US 2

Pricing Tier *: Premium (+ Role-based access controls)

- Standard (Apache Spark, Secure with Azure AD)
- Premium (+ Role-based access controls)
- Trial (Premium - 14-Days Free DBUs)

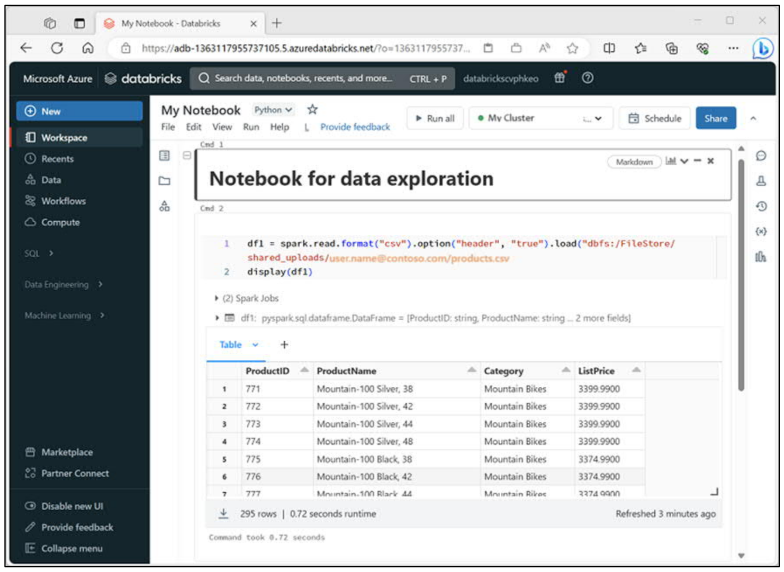
To use Azure Databricks, you must create an Azure Databricks workspace in your Azure subscription. You can accomplish this by:
Using the Azure portal user interface
Using an Azure Resource Manager (ARM) or Bicep template
Using the New-AzDatabricksWorkspace Azure PowerShell cmdlet
Using the az databricks workspace create Azure command line interface (CLI) command

When you create a workspace, you must specify one of the following pricing tiers:
Standard - Core Apache Spark capabilities with Azure AD integration
Premium - Role-based access controls and other enterprise-level features
Trial - A 14-day free trial of a premium-level workspace

Use the Azure Databricks portal

Web-based user interface to

- Create and manage workspace resources
- Work with data in files and tables
- Use notebooks to run Spark code



```
df1 = spark.read.format("csv").option("header", "true").load("dbfs:/FileStore/
shared_uploads/user.name@contoso.com/products.csv
display(df1)
```

(2) Spark Jobs

df1: pyspark.sql.dataframe.DataFrame = [ProductID: string, ProductName: string ... 2 more fields]

Table

	ProductID	ProductName	Category	ListPrice
1	771	Mountain-100 Silver, 38	Mountain Bikes	3399.9900
2	772	Mountain-100 Silver, 42	Mountain Bikes	3399.9900
3	773	Mountain-100 Silver, 44	Mountain Bikes	3399.9900
4	774	Mountain-100 Silver, 48	Mountain Bikes	3399.9900
5	775	Mountain-100 Black, 38	Mountain Bikes	3374.9900
6	776	Mountain-100 Black, 42	Mountain Bikes	3374.9900
7	777	Mountain-100 Black, 44	Mountain Bikes	3374.9900

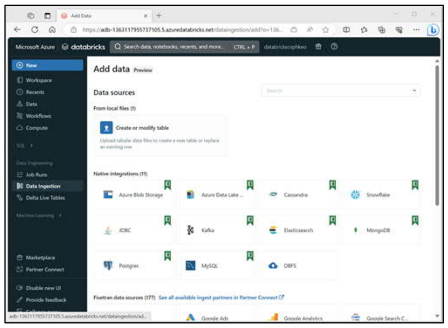
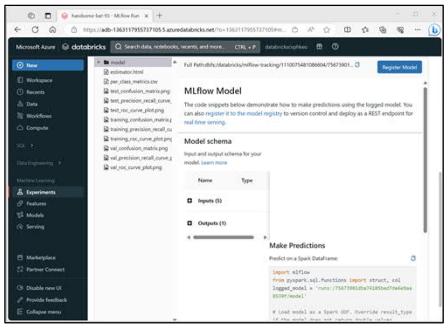
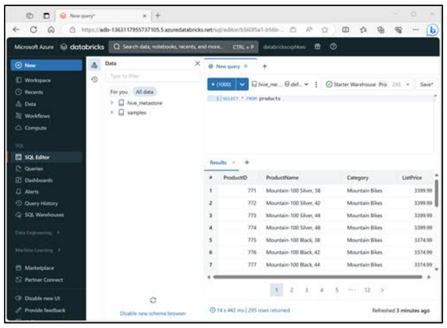
295 rows | 0.72 seconds runtime

Refreshed 3 minutes ago

```
Command took 0.72 seconds
```

After you've provisioned an Azure Databricks workspace, you can use the Azure Databricks portal to work with data and compute resources. The Azure Databricks portal is a web-based user interface through which you can create and manage workspace resources (such as Spark clusters) and work with data in files and tables.

Azure Databricks workloads

Data Science and Engineering	Machine Learning	SQL
		

Azure Databricks is a comprehensive platform that offers many data processing capabilities. While you can use the service to support any workload that requires scalable data processing, Azure Databricks is optimized for three specific types of data workload:

Data Science and Engineering

Machine Learning

SQL (only available in premium tier workspaces.)

The Azure Databricks user interface supports three corresponding persona views that you can switch between depending on the workload you're implementing.

Data Science and Engineering

Azure Databricks provides Apache Spark based processing and analysis of large volumes of data in a data lake. Data engineers, data scientists, and data analysts can use interactive notebooks to run code in Python, Scala, SparkSQL, or other languages to cleanse, transform, aggregate, and analyze data.

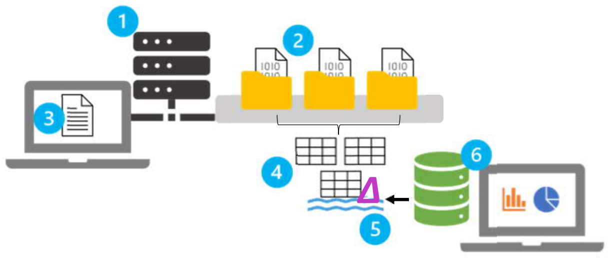
Machine Learning

Azure Databricks supports machine learning workloads that involve data exploration and preparation, training and evaluating machine learning models, and serving models to generate predictions for applications and analyses. Data scientists and ML engineers can use AutoML to quickly train predictive models, or apply their skills with common machine learning frameworks such as SparkML, Scikit-Learn, PyTorch, and Tensorflow. They can also manage the end-to-end machine learning lifecycle with MLFlow.

SQL

Azure Databricks supports SQL-based querying for data stored in tables in a SQL Warehouse. This capability enables data analysts to query, aggregate, summarize, and visualize data using familiar SQL syntax and a wide range of SQL-based data analytical tools.

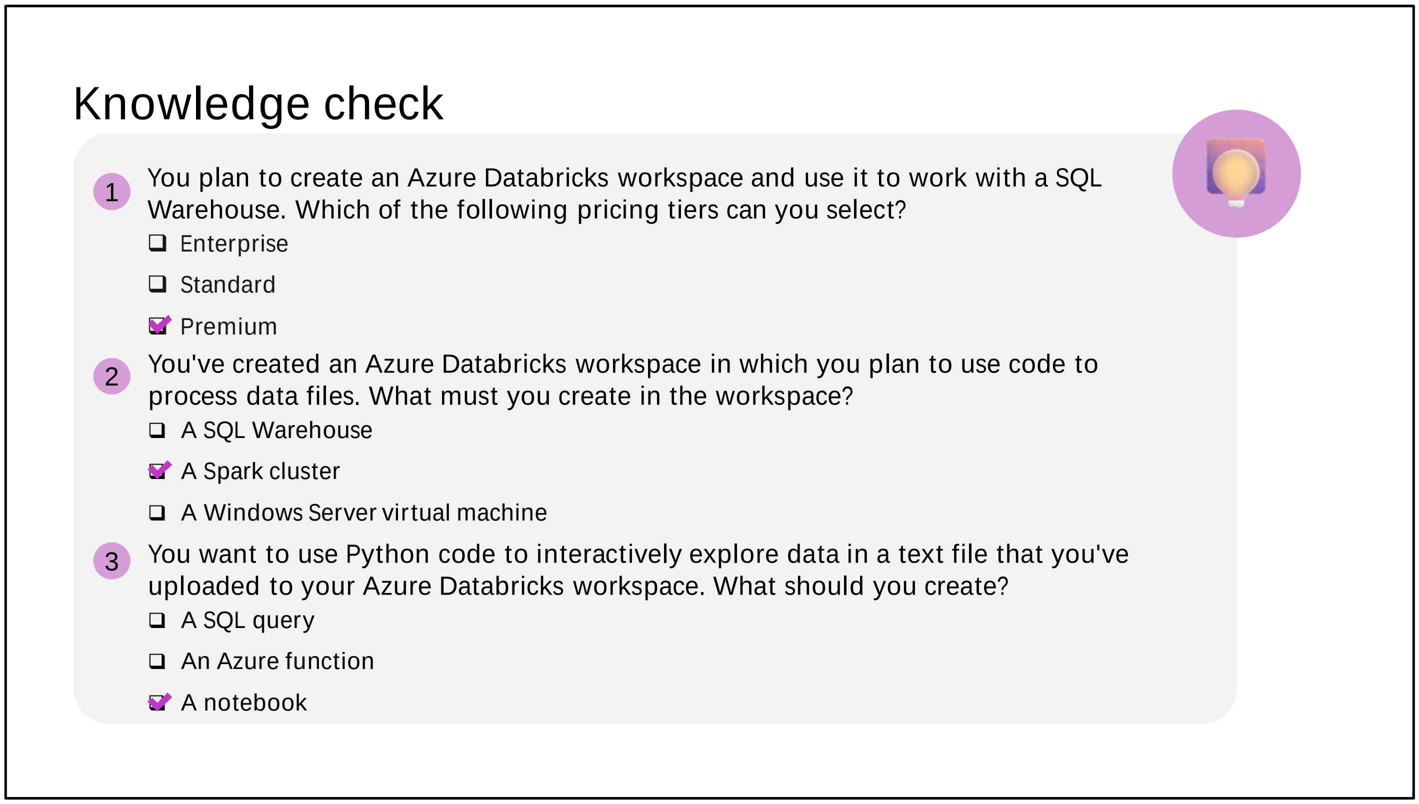
Key concepts



1. Apache Spark clusters provide highly scalable parallel compute for distributed data processing
2. Databricks File System (DBFS) provides distributed shared storage for data lakes
3. Notebooks provide an interactive environment for combining code, notes, and images
4. Metastore provides a relational abstraction layer, enabling you to define tables based on data in files
5. Delta Lake builds on the metastore to enable common relational database capabilities
6. SQL Warehouses provide relational compute endpoints for querying data in tables

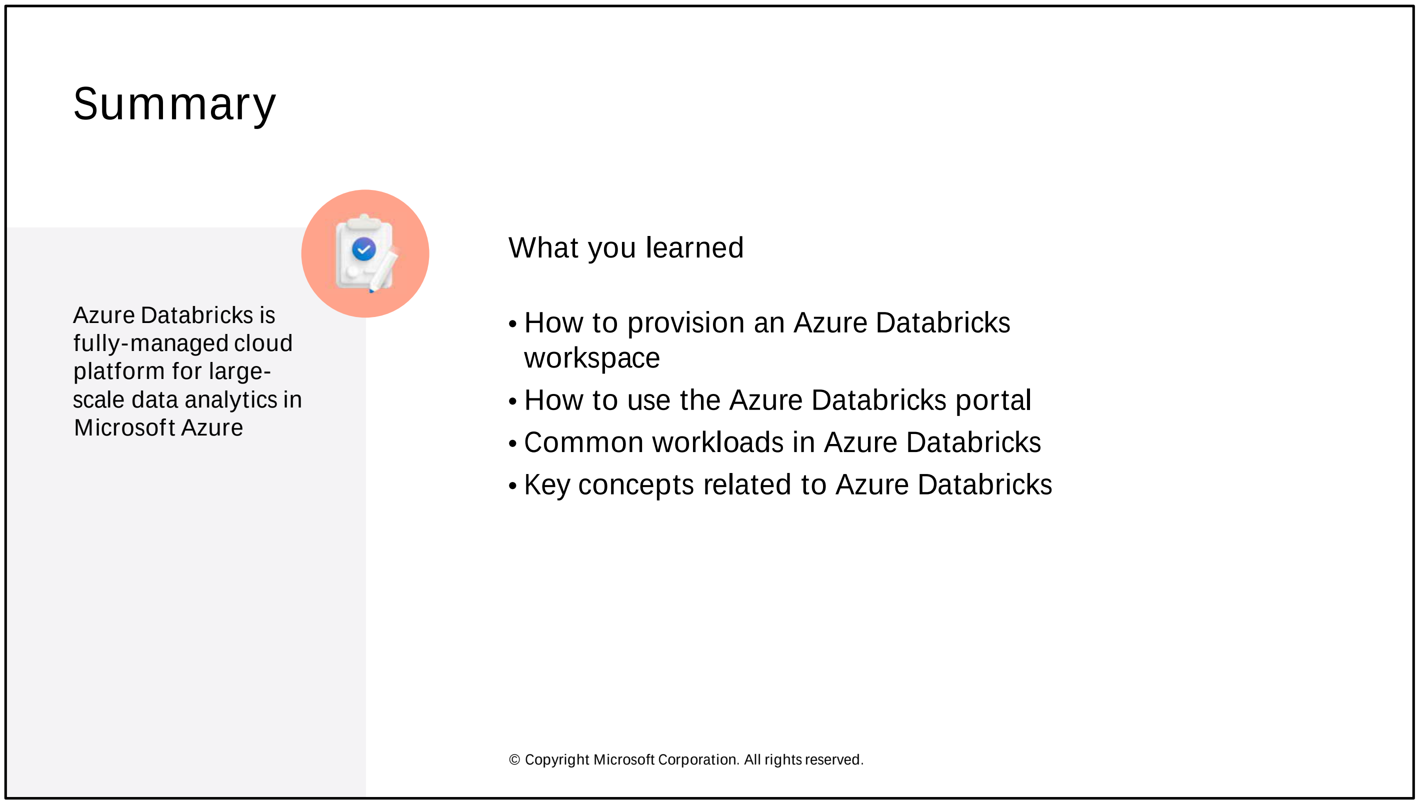
Many of the same concepts are relevant in Azure Synapse Analytics and Microsoft Fabric – particularly when applying Spark workloads. Clusters in Azure Databricks are the equivalent of Spark pools in Azure Synapse Analytics and Fabric. The DBFS file system in Azure Databricks performs the same distributed data storage role as data lake storage linked services in Azure Synapse Analytics and the OneLake storage tier in Fabric.

Notebooks in Azure Databricks share many similarities with notebooks in Azure Synapse Analytics and Fabric. The metastore is a standard Spark construct in Azure Databricks, Azure Synapse Analytics, and Fabric; and enables Spark SQL to overlay relational table schema onto data in files and to support Delta Lake tables. One significant difference between Azure Databricks and Azure Synapse Analytics is the implementation of SQL Warehouses; in Azure Databricks these are more similar to Lake Databases in a serverless SQL pool than a data warehouse in a dedicated pool in that the data is not stored in relational storage, but rather as files in the DBFS file system. SQL Warehouses offer a SQL-optimized compute engine with support for standard SQL processing semantics; but maintains a separation between query processing and data storage in a similar way to data warehouses in Microsoft Fabric.



Knowledge check

1. You plan to create an Azure Databricks workspace and use it to work with a SQL Warehouse. Which of the following pricing tiers can you select?
   - ❑ Enterprise
   - ❑ Standard
   - ✓ Premium
2. You've created an Azure Databricks workspace in which you plan to use code to process data files. What must you create in the workspace?
   - ❑ A SQL Warehouse
   - ✓ A Spark cluster
   - ❑ A Windows Server virtual machine
3. You want to use Python code to interactively explore data in a text file that you've uploaded to your Azure Databricks workspace. What should you create?
   - ❑ A SQL query
   - ❑ An Azure function
   - ✓ A notebook

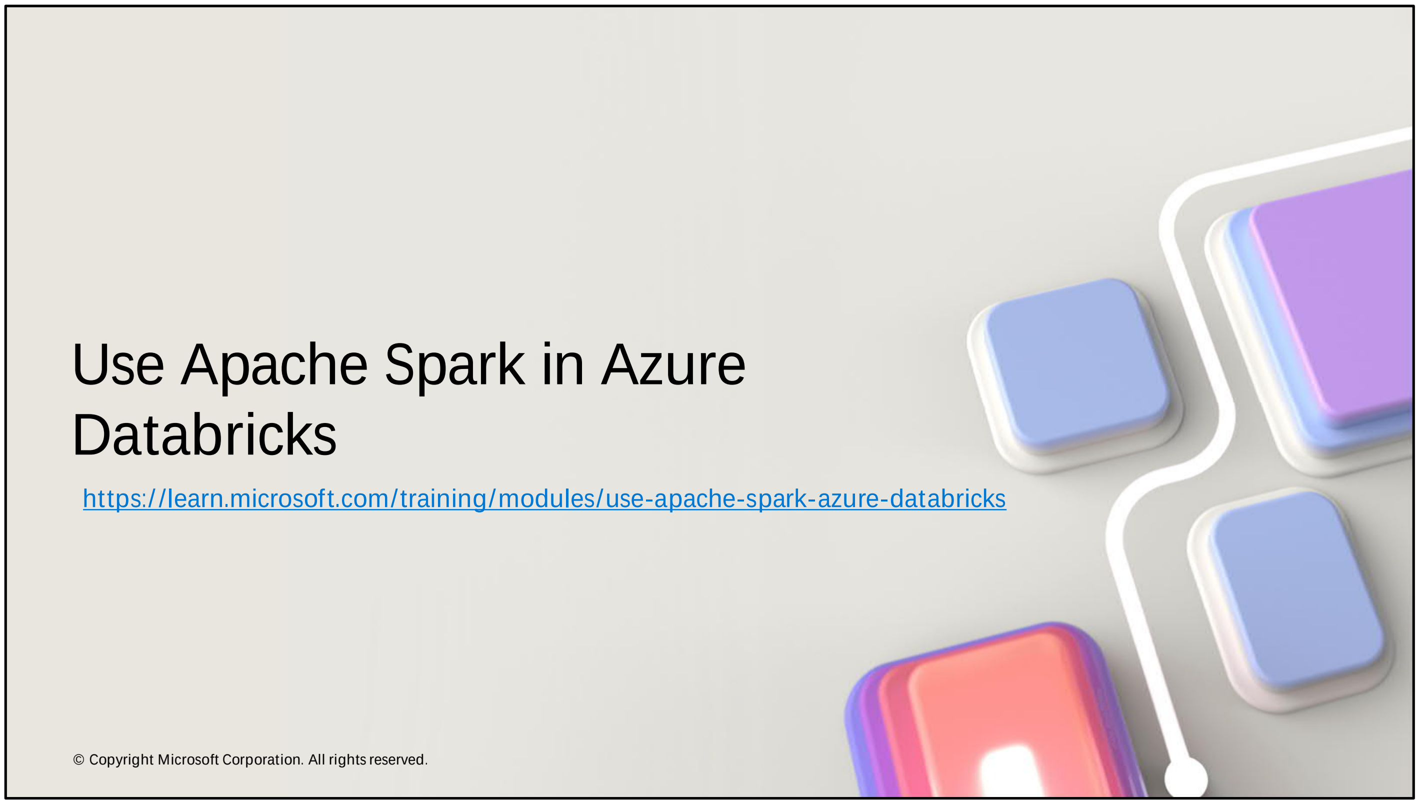


Summary

Azure Databricks is fully-managed cloud platform for large-scale data analytics in Microsoft Azure

What you learned

- How to provision an Azure Databricks workspace
- How to use the Azure Databricks portal
- Common workloads in Azure Databricks
- Key concepts related to Azure Databricks



Use Apache Spark in Azure Databricks

https://learn.microsoft.com/training/modules/use-apache-spark-azure-databricks

Azure Databricks offers a highly scalable platform for data analytics and processing using Apache Spark.

Spark is a flexible platform that supports many different programming languages and APIs. Most data processing and analytics tasks can be accomplished using the Dataframe API, which is what we'll focus on in this module.

In this module, you'll learn how to:

Describe key elements of the Apache Spark architecture.

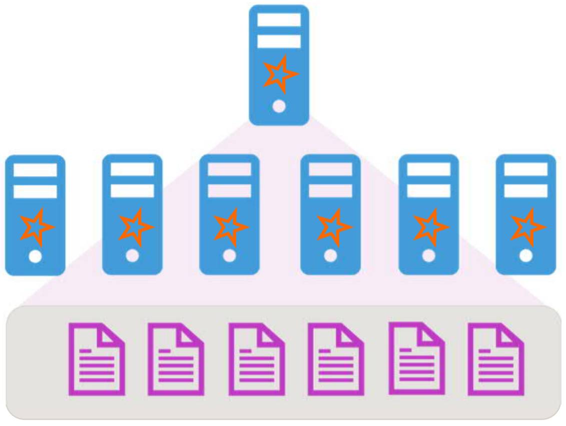
Create and configure a Spark cluster.

Use Spark to process and analyze data stored in files.

Use Spark to visualize data.

Get to know Apache Spark

- Open source parallel processing framework
- Designed for large-scale data processing an analytics
- Cluster architecture consists of *driver* and *worker* nodes:
  - Driver program coordinates the overall distributed work using a *SparkSession* object
  - Workers process subsets of the data in parallel
  - Jobs are performed in stages to ensure consistency of collated results



To gain a better understanding of how to process and analyze data with Apache Spark in Azure Databricks, it's important to understand the underlying architecture.

High-level overview

From a high level, the Azure Databricks service launches and manages Apache Spark clusters within your Azure subscription. Apache Spark clusters are groups of computers that are treated as a single computer and handle the execution of commands issued from notebooks. Clusters enable processing of data to be parallelized across many computers to improve scale and performance. They consist of a Spark driver and worker nodes. The driver node sends work to the worker nodes and instructs them to pull data from a specified data source.

In Databricks, the notebook interface is typically the driver program. This driver program contains the main loop for the program and creates distributed datasets on the cluster, then applies operations to those datasets. Driver programs access Apache Spark through a SparkSession object regardless of deployment location.

Microsoft Azure manages the cluster, and auto-scales it as needed based on your usage and the setting used when configuring the cluster. Auto-termination can also be enabled, which allows Azure to terminate the cluster after a specified number of minutes of inactivity.

Spark jobs in detail

Work submitted to the cluster is split into as many independent jobs as needed. This is how work is distributed across the cluster's nodes. Jobs are further subdivided into tasks. The input to a job is partitioned into one or more partitions. These partitions are the unit of work for each slot. In between tasks, partitions may need to be reorganized and shared over the network.

The secret to Spark's high performance is parallelism. Scaling vertically (by adding resources to a single computer) is limited to a finite amount of RAM, Threads and CPU speeds; but clusters scale horizontally, adding new nodes to the cluster as needed.

Spark parallelizes jobs at two levels:

The first level of parallelization is the executor - a Java virtual machine (JVM) running on a worker node, typically, one instance per node.

The second level of parallelization is the slot - the number of which is determined by the number of cores and CPUs of each node.

Each executor has multiple slots to which parallelized tasks can be assigned.

The JVM is naturally multi-threaded, but a single JVM, such as the one coordinating the work on the driver, has a finite upper limit. By splitting the work into tasks, the driver can assign units of work to *slots in the executors on worker nodes for parallel execution. Additionally, the driver determines how to partition the data so that it can be distributed for parallel processing. So, the driver assigns a partition of data to each task so that each task knows which piece of data it is to process. Once started, each task will fetch the partition of data assigned to it.

Jobs and stages

Depending on the work being performed, multiple parallelized jobs may be required. Each job is broken down into stages. A useful analogy is to imagine that the job is to build a house:

The first stage would be to lay the foundation.

The second stage would be to erect the walls.

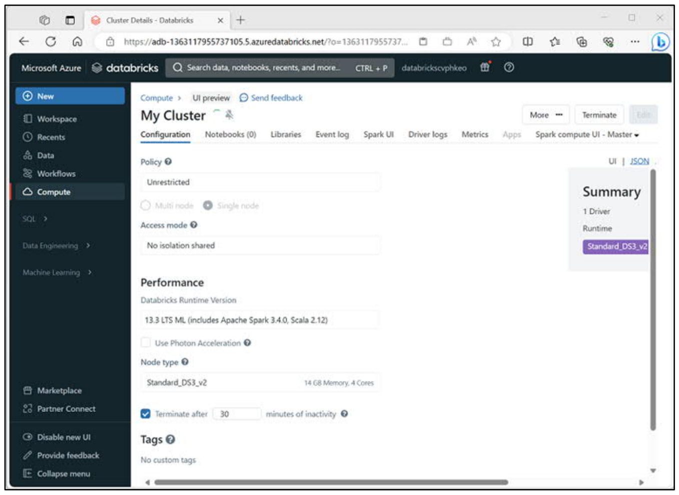
The third stage would be to add the roof.

Attempting to do any of these steps out of order just doesn't make sense, and may in fact be impossible. Similarly, Spark breaks each job into stages to ensure everything is done in the right order.

Create a Spark cluster

Create a cluster in the Azure Databricks portal, specifying:

- Cluster name
- Cluster mode (standard, high-concurrency, or single-node)
- Databricks Runtime version
- Worker and driver node VM configuration
- Autoscaling and automatic shutdown



My Cluster

Policy: Unrestricted

Multi node [✓ Single node]

Access mode: No isolation shared

Performance

Databricks Runtime Version: 13.3 LTS ML (includes Apache Spark 3.4.0, Scala 2.12)

☐ Use Photon Acceleration

Node type: Standard_DS3_v2 14 GB Memory, 4 Cores

☑ Terminate after 30 minutes of inactivity

Tags

No custom tags

You can create one or more clusters in your Azure Databricks workspace by using the Azure Databricks portal.

When creating the cluster, you can specify configuration settings, including:

•A name for the cluster.

•A *cluster mode*, which can be:

- *Standard*: Suitable for single-user workloads that require multiple worker nodes.
- *High Concurrency*: Suitable for workloads where multiple users will be using the cluster concurrently.
- *Single Node*: Suitable for small workloads or testing, where only a single worker node is required.

•The version of the *Databricks Runtime* to be used in the cluster; which dictates the version of Spark and individual components such as Python, Scala, and others that get installed.

•The type of virtual machine (VM) used for the worker nodes in the cluster.

•The minimum and maximum number of worker nodes in the cluster.

•The type of VM used for the driver node in the cluster.

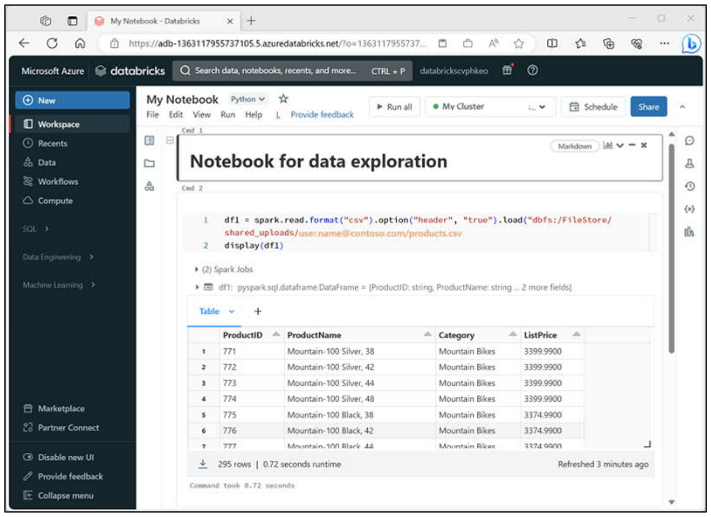
•Whether the cluster supports *autoscaling* to dynamically resize the cluster.

•How long the cluster can remain idle before being shut down automatically.

Use Spark in notebooks

Interactive notebooks

- Syntax highlighting and error support
- Code auto-completion
- Interactive data visualizations
- The ability to export results



Notebook for data exploration

```
df1 = spark.read.format("csv").option("header", "true").load("dbfs:/FileStore/shared_uploads/user.name@contoso.com/products.csv
display(df1)
```

(2) Spark Jobs

df1: pyspark.sql.dataframe.DataFrame = [ProductID: string, ProductName: string ... 2 more fields]

	ProductID	ProductName	Category	ListPrice
1	771	Mountain-100 Silver, 38	Mountain Bikes	3399.9900
2	772	Mountain-100 Silver, 42	Mountain Bikes	3399.9900
3	773	Mountain-100 Silver, 44	Mountain Bikes	3399.9900
4	774	Mountain-100 Silver, 48	Mountain Bikes	3399.9900
5	775	Mountain-100 Black, 38	Mountain Bikes	3374.9900
6	776	Mountain-100 Black, 42	Mountain Bikes	3374.9900
7	777	Mountain-100 Black, 44	Mountain Bikes	3374.9900

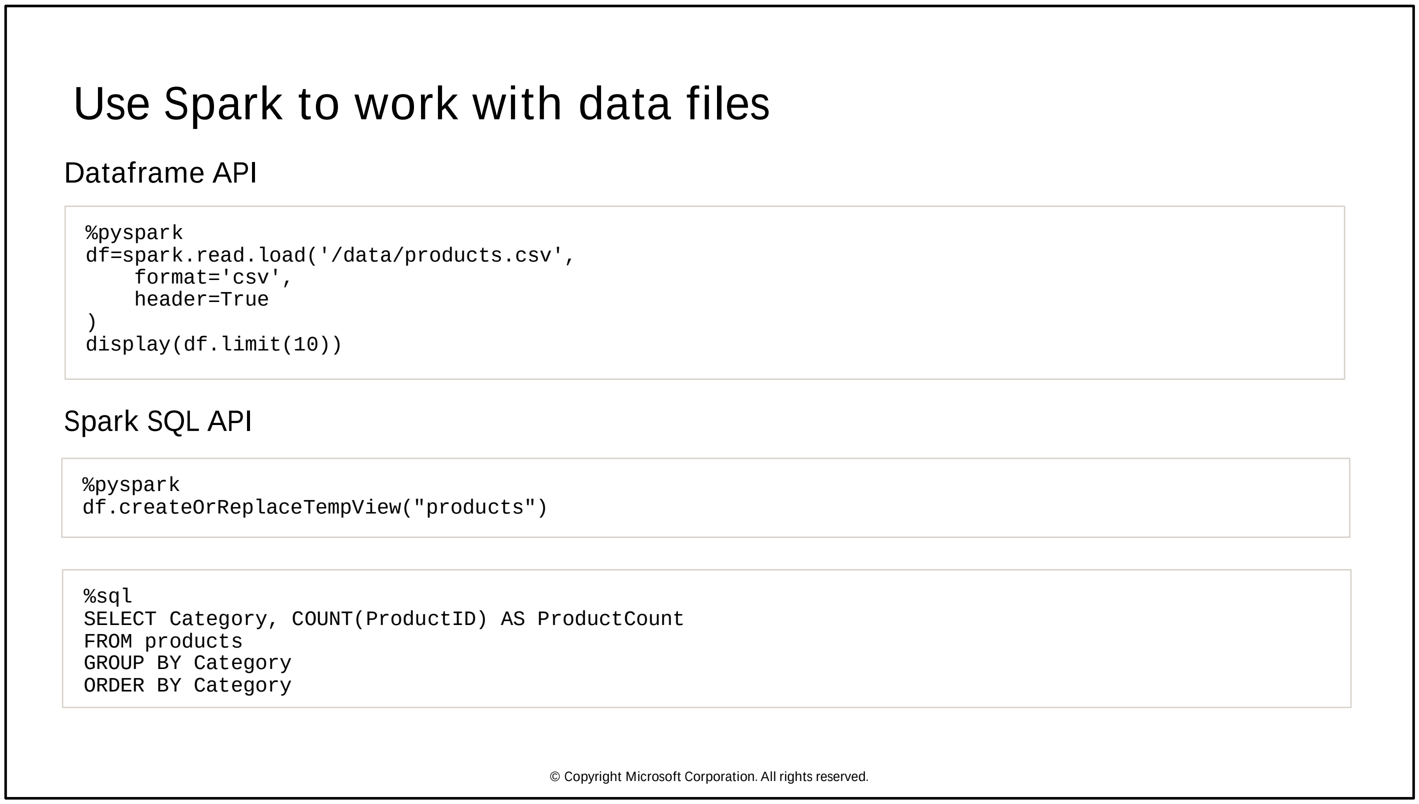
295 rows | 0.72 seconds runtime   Refreshed 3 minutes ago

```
Command took 0.72 seconds
```

Azure Databricks includes an integrated notebook interface for working with Spark. Notebooks provide an intuitive way to combine code with Markdown notes, commonly used by data scientists and data analysts. The look and feel of the integrated notebook experience within Azure Databricks is similar to that of Jupyter notebooks - a popular open source notebook platform.

Notebooks consist of one or more cells, each containing either code or markdown. Code cells in notebooks have some features that can help you be more productive, including:

Syntax highlighting and error support.
Code auto-completion.
Interactive data visualizations.
The ability to export results.



Use Spark to work with data files

Dataframe API

```
%pyspark
df=spark.read.load('/data/products.csv',
    format='csv',
    header=True
)
display(df.limit(10))
```

Spark SQL API

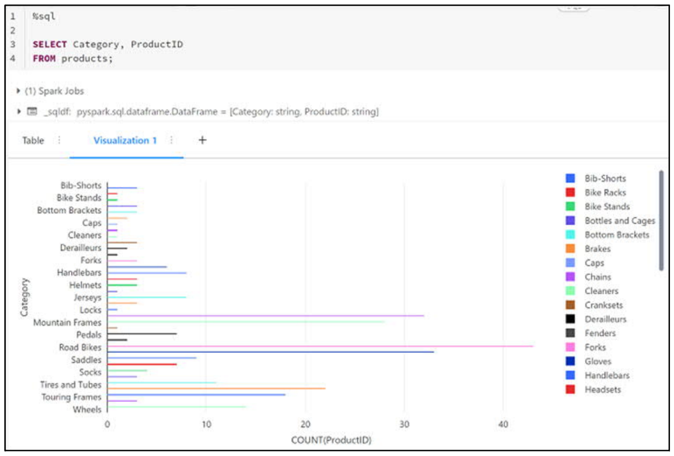
```
%pyspark
df.createOrReplaceTempView("products")
```

```
%sql
SELECT Category, COUNT(ProductID) AS ProductCount
FROM products
GROUP BY Category
ORDER BY Category
```

Natively, Spark uses a data structure called a resilient distributed dataset (RDD); but while you can write code that works directly with RDDs, the most commonly used data structure for working with structured data in Spark is the dataframe, which is provided as part of the Spark SQL library. Dataframes in Spark are similar to those in the ubiquitous Pandas Python library, but optimized to work in Spark's distributed processing environment.

Visualize data

Built-in charts



```
%sql

SELECT Category, ProductID
FROM products;
```

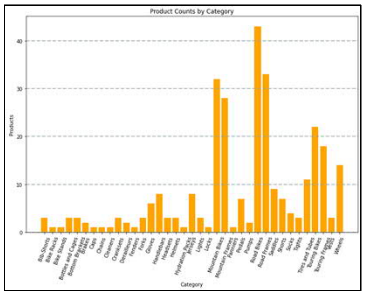
(1) Spark Jobs

_sqldf: pyspark.sql.dataframe.DataFrame = [Category: string, ProductID: string]

Graphics packages

```
from matplotlib import pyplot as plt

fig = plt.figure(figsize=(12,8))
plt.bar(x=data['Category'],
        height=data['ProductCount'],
        color='orange’)
plt.show()
```



One of the most intuitive ways to analyze the results of data queries is to visualize them as charts. Notebooks in Azure Databricks provide charting capabilities in the user interface, and when that functionality doesn't provide what you need, you can use one of the many Python graphics libraries to create and display data visualizations in the notebook.

Knowledge check

1 Which definition best describes Apache Spark?

- ☐ A highly scalable relational database management system
- ☐ A virtual server with a Python runtime
- [✓ A distributed platform for parallel data processing using multiple languages]

2 You need to use Spark to analyze data in a parquet file. What should you do?

- [✓ Load the parquet file into a dataframe]
- ☐ Import the data into a table in a serverless SQL pool
- ☐ Convert the data to CSV format

3 You want to write code in a notebook cell that uses a SQL query to retrieve data from a view in the Spark catalog. Which magic should you use?

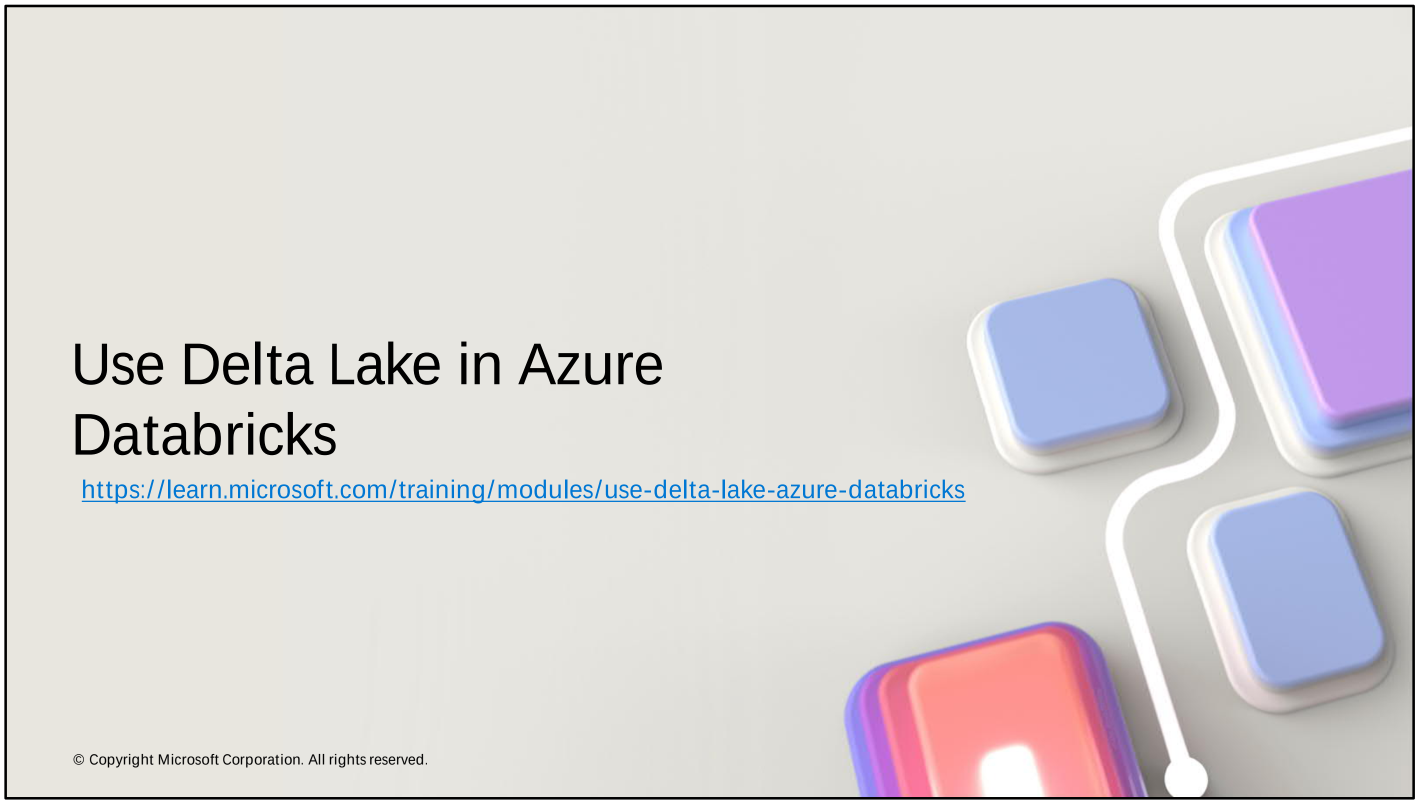
- ☐ %spark
- ☐ %pyspark
- [✓ %sql]

Summary

Azure Databricks is built on Apache Spark and enables data engineers and analysts to run Spark jobs to transform, analyze and visualize data at scale.

What you learned

- Describe key elements of the Apache Spark architecture
- Create and configure a Spark cluster
- Use Spark to process and analyze data stored in files
- Use Spark to visualize data



Use Delta Lake in Azure Databricks

https://learn.microsoft.com/training/modules/use-delta-lake-azure-databricks

Delta Lake is an open source relational storage area for Spark that you can use to implement a data lakehouse architecture in Azure Databricks.

Learning objectives

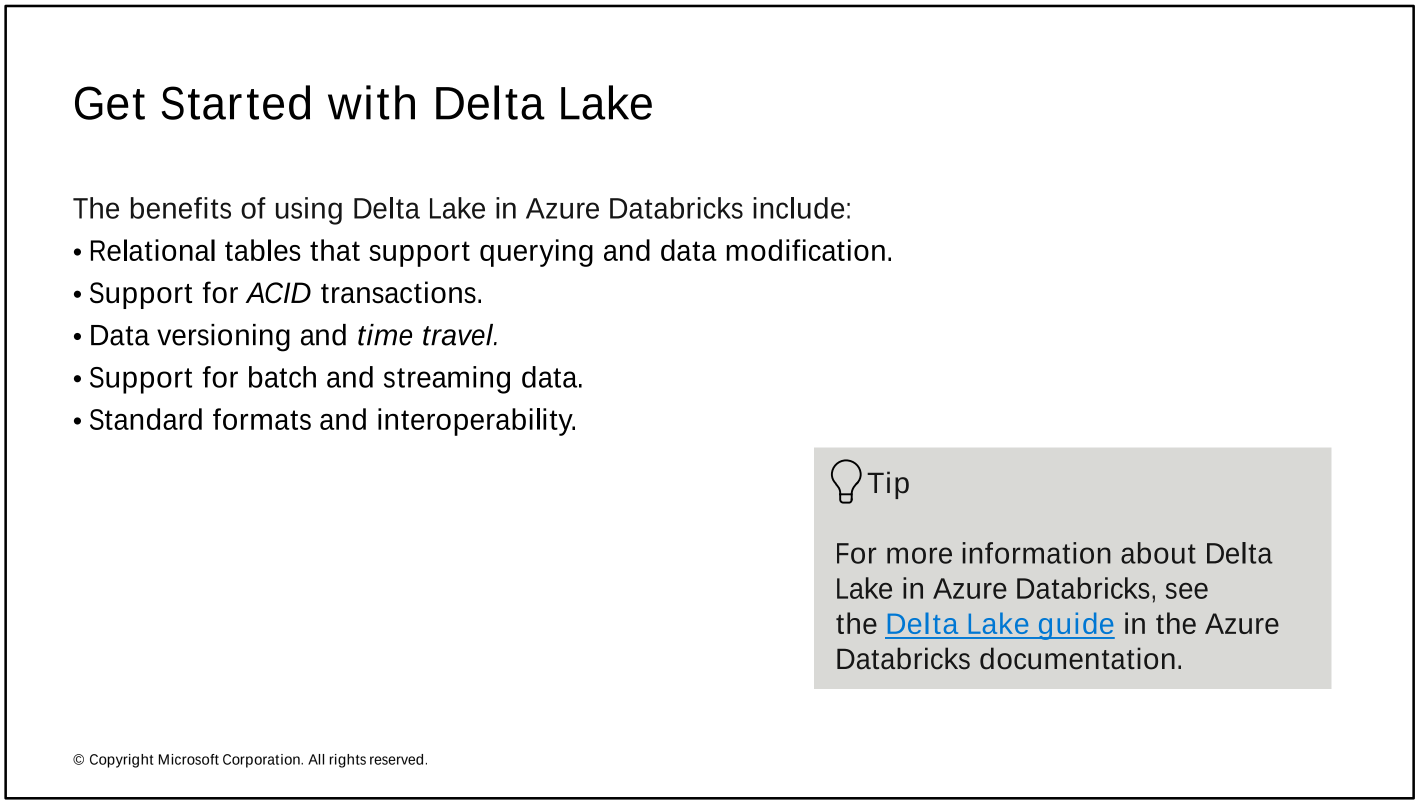
In this module, you'll learn how to:

Describe core features and capabilities of Delta Lake.

Create and use Delta Lake tables in Azure Databricks.

Create Spark catalog tables for Delta Lake data.

Use Delta Lake tables for streaming data.



Get Started with Delta Lake

The benefits of using Delta Lake in Azure Databricks include:

- Relational tables that support querying and data modification.
- Support for *ACID* transactions.
- Data versioning and *time travel.*
- Support for batch and streaming data.
- Standard formats and interoperability.

Tip

For more information about Delta Lake in Azure Databricks, see the Delta Lake guide in the Azure Databricks documentation.

https://learn.microsoft.com/training/modules/use-delta-lake-azure-databricks/02-understand-delta-lake

Delta Lake is an open-source storage layer that adds relational database semantics to Spark-based data lake processing. Delta Lake is supported in Azure Synapse Analytics Spark pools for PySpark, Scala, and .NET code. The benefits of using Delta Lake in Azure Databricks include:

Relational tables that support querying and data modification.

Support for ACID transactions.

Data versioning and time travel.

Support for batch and streaming data.

Standard formats and interoperability.

Create Delta Lake tables

- Creating a Delta Lake table from a dataframe
- Making conditional updates
- Querying a previous version of a table

Tip

For more information about using the Data Lake API, see the Delta Lake API documentation.

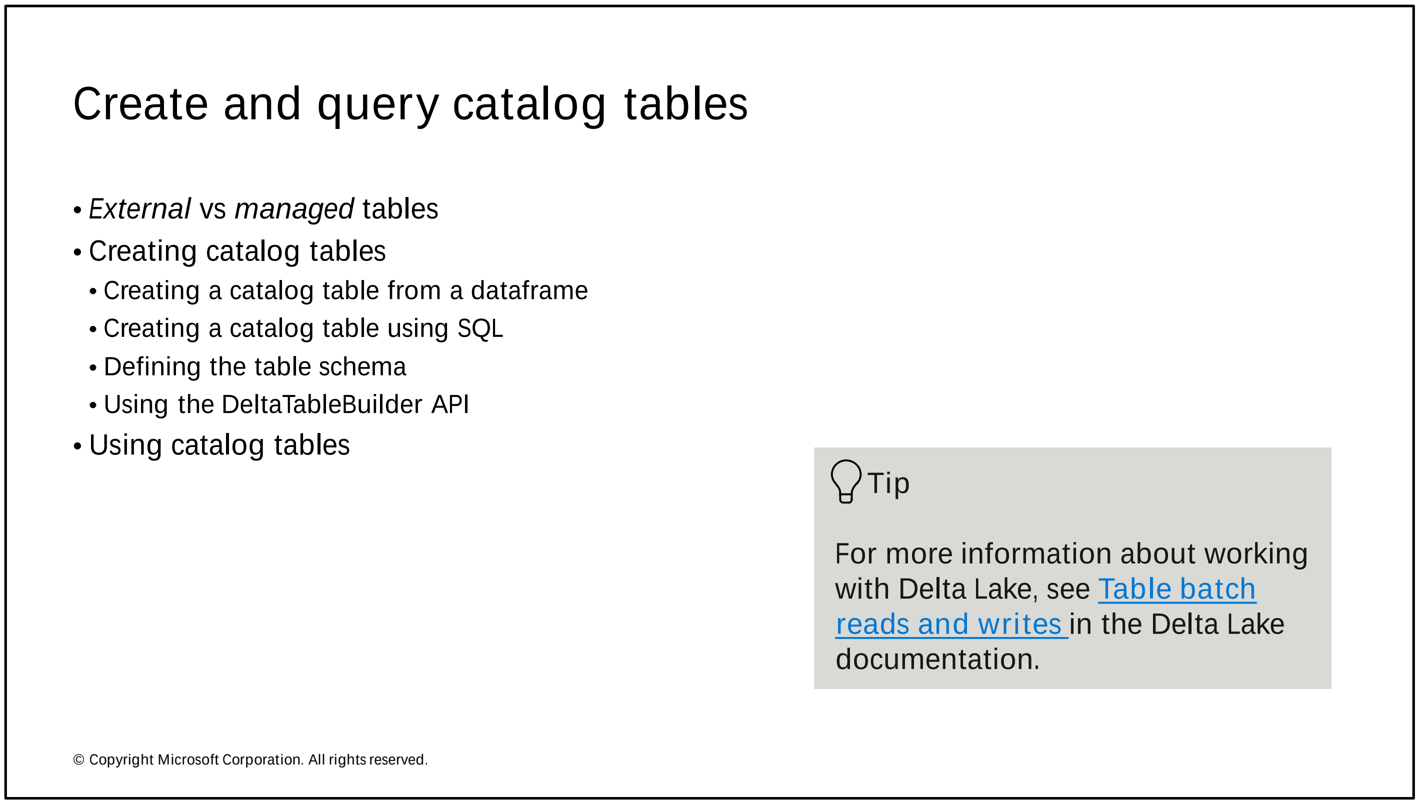
https://learn.microsoft.com/training/modules/use-delta-lake-azure-databricks/03-create-delta-tables

Delta lake is built on tables, which provide a relational storage abstraction over files in a data lake.

One of the easiest ways to create a Delta Lake table is to save a dataframe in the delta format, specifying a path where the data files and related metadata information for the table should be stored.

While you can make data modifications in a dataframe and then replace a Delta Lake table by overwriting it, a more common pattern in a database is to insert, update or delete rows in an existing table as discrete transactional operations. To make such modifications to a Delta Lake table, you can use the DeltaTable object in the Delta Lake API, which supports update, delete, and merge operations.

Delta Lake tables support versioning through the transaction log. The transaction log records modifications made to the table, noting the timestamp and version number for each transaction. You can use this logged version data to view previous versions of the table - a feature known as time travel.



Create and query catalog tables

- *External* vs *managed* tables
- Creating catalog tables
  - Creating a catalog table from a dataframe
  - Creating a catalog table using SQL
  - Defining the table schema
  - Using the DeltaTableBuilder API
- Using catalog tables

Tip

For more information about working with Delta Lake, see [Table batch reads and writes](https://docs.delta.io/latest/delta-batch.html) in the Delta Lake documentation.

https://learn.microsoft.com/training/modules/use-delta-lake-azure-databricks/04-catalog-tables

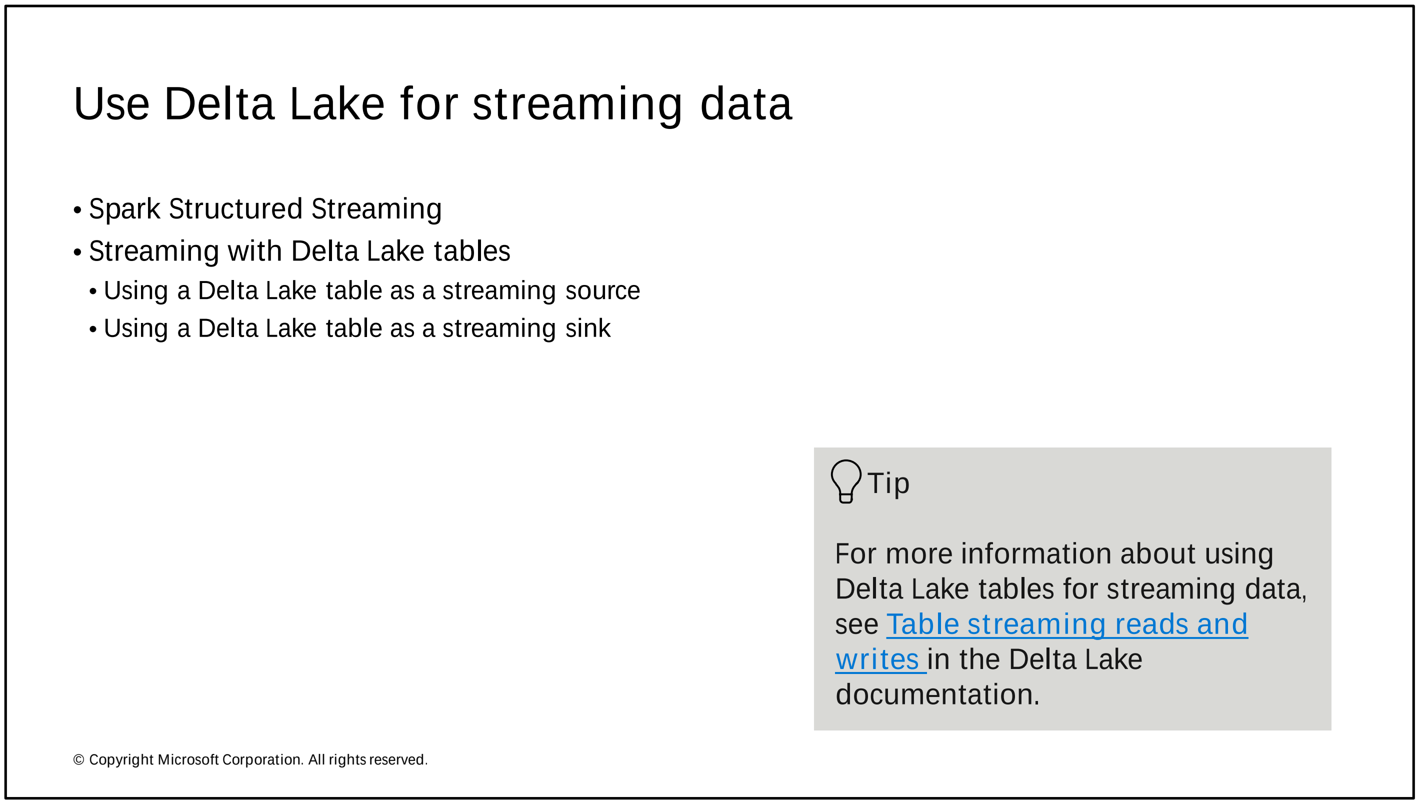
https://docs.delta.io/latest/delta-batch.html

External vs managed tables

Tables in a Spark catalog, including Delta Lake tables, can be managed or external; and it's important to understand the distinction between these kinds of table.

A managed table is defined without a specified location, and the data files are stored within the storage used by the metastore. Dropping the table not only removes its metadata from the catalog, but also deletes the folder in which its data files are stored.

An external table is defined for a custom file location, where the data for the table is stored. The metadata for the table is defined in the Spark catalog. Dropping the table deletes the metadata from the catalog, but doesn't affect the data files.



Use Delta Lake for streaming data

- Spark Structured Streaming
- Streaming with Delta Lake tables
  - Using a Delta Lake table as a streaming source
  - Using a Delta Lake table as a streaming sink

Tip

For more information about using Delta Lake tables for streaming data, see Table streaming reads and writes in the Delta Lake documentation.

https://learn.microsoft.com/training/modules/use-delta-lake-azure-databricks/05-use-delta-lake-streaming-data

https://docs.delta.io/latest/delta-streaming.html

All of the data we've explored up to this point has been static data in files. However, many data analytics scenarios involve streaming data that must be processed in near real time. For example, you might need to capture readings emitted by internet-of-things (IoT) devices and store them in a table as they occur.

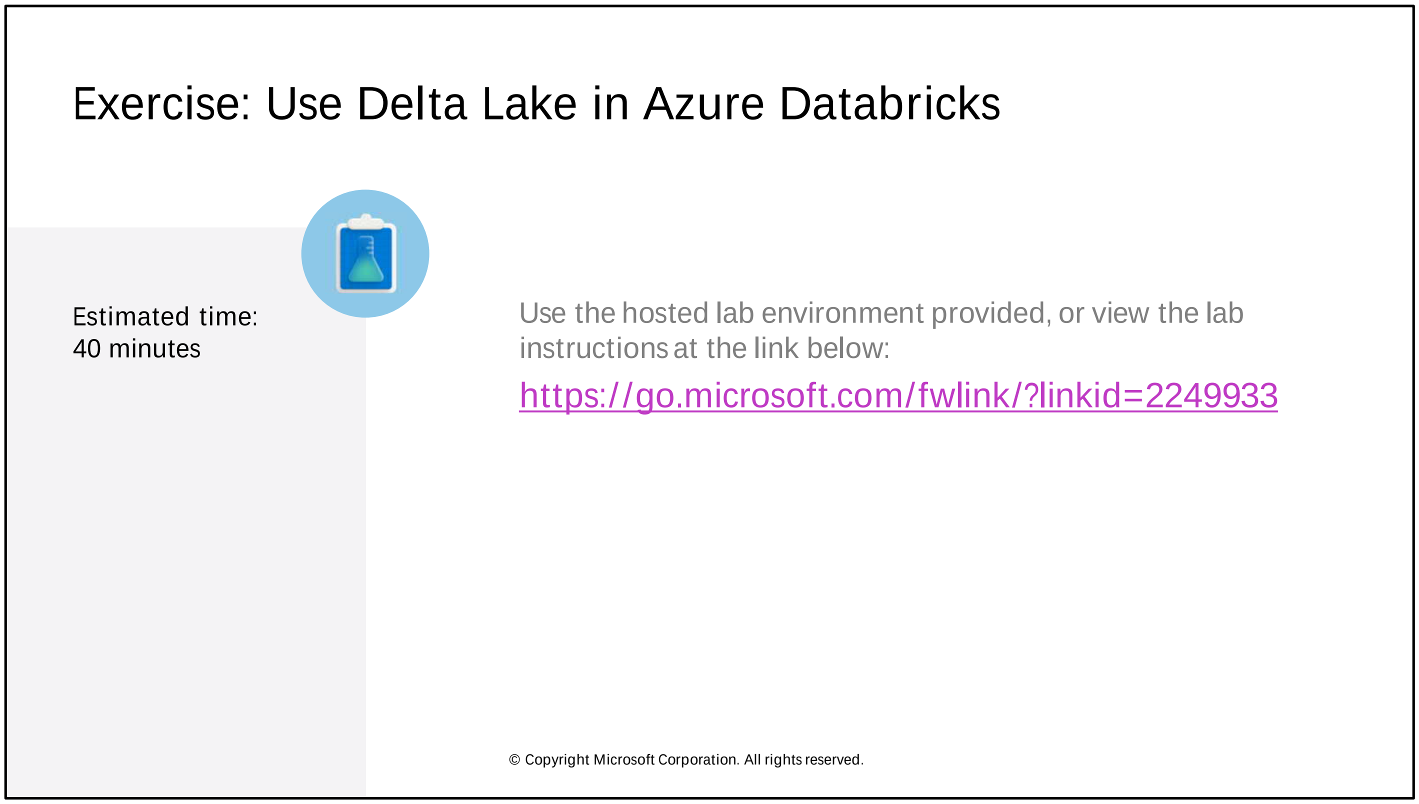
Spark Structured Streaming

A typical stream processing solution involves constantly reading a stream of data from a source, optionally processing it to select specific fields, aggregate and group values, or otherwise manipulate the data, and writing the results to a sink.

Spark includes native support for streaming data through Spark Structured Streaming, an API that is based on a boundless dataframe in which streaming data is captured for processing. A Spark Structured Streaming dataframe can read data from many different kinds of streaming source, including network ports, real time message brokering services such as Azure Event Hubs or Kafka, or file system locations.

You can use a Delta Lake table as a source or a sink for Spark Structured Streaming. For example, you could capture a stream of real time data from an IoT device and write the stream directly to a Delta Lake table as a sink - enabling you to query the table to see the latest streamed data. Or, you could read a Delta Table as a streaming source, enabling you to constantly report new data as it is added to the table.

After reading the data from the Delta Lake table into a streaming dataframe, you can use the Spark Structured Streaming API to process it.



Exercise: Use Delta Lake in Azure Databricks

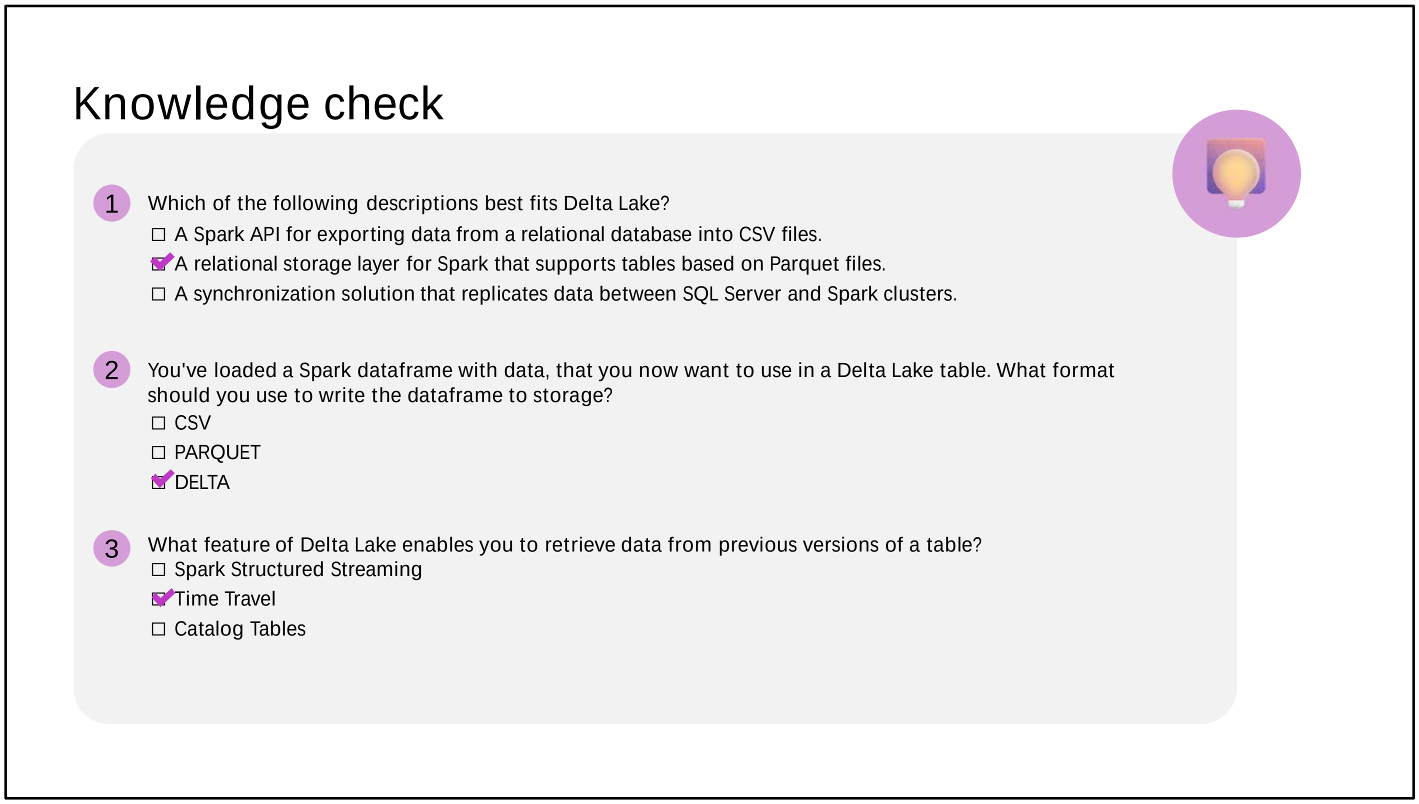
Estimated time:
40 minutes

Use the hosted lab environment provided, or view the lab instructions at the link below:

https://go.microsoft.com/fwlink/?linkid=2249933

Modify the slide as necessary to provide the students with any lab environment details that are specific to your delivery. For example:

- Where to go to launch a hosted lab environment (if used)
- When to find Azure credentials to use (if provided)



Knowledge check

1 Which of the following descriptions best fits Delta Lake?
- ☐ A Spark API for exporting data from a relational database into CSV files.
- ✓ A relational storage layer for Spark that supports tables based on Parquet files.
- ☐ A synchronization solution that replicates data between SQL Server and Spark clusters.

2 You've loaded a Spark dataframe with data, that you now want to use in a Delta Lake table. What format should you use to write the dataframe to storage?
- ☐ CSV
- ☐ PARQUET
- ✓ DELTA

3 What feature of Delta Lake enables you to retrieve data from previous versions of a table?
- ☐ Spark Structured Streaming
- ✓ Time Travel
- ☐ Catalog Tables

1. Which of the following descriptions best fits Delta Lake?

A Spark API for exporting data from a relational database into CSV files.

A relational storage layer for Spark that supports tables based on Parquet files.

Correct. Delta Lake provides a relational storage layer in which you can create tables based on Parquet files in a data lake.

A synchronization solution that replicates data between SQL Server and Spark clusters.

2. You've loaded a Spark dataframe with data, that you now want to use in a Delta Lake table. What format should you use to write the dataframe to storage?

CSV

PARQUET

DELTA

Correct. Storing a dataframe in DELTA format creates parquet files for the data and the transaction log metadata necessary for Delta Lake tables.

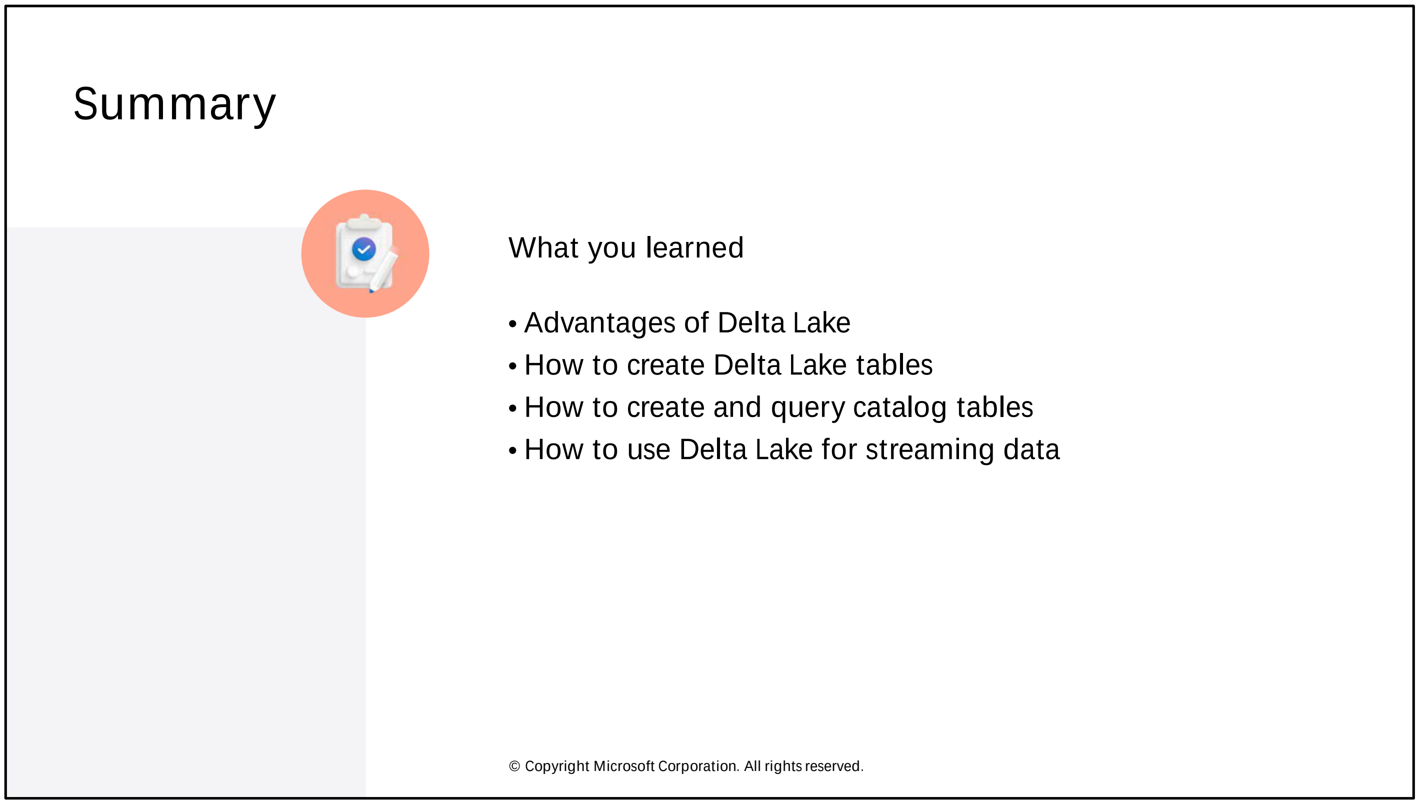
3. What feature of Delta Lake enables you to retrieve data from previous versions of a table?

Spark Structured Streaming

Time Travel

Correct. The Time Travel feature is based on the transaction log, which enables you to specify a version number or timestamp for the data you want to retrieve.

Catalog Tables



Summary

What you learned

- Advantages of Delta Lake
- How to create Delta Lake tables
- How to create and query catalog tables
- How to use Delta Lake for streaming data

1. Which of the following descriptions best fits Delta Lake?

A Spark API for exporting data from a relational database into CSV files.

A relational storage layer for Spark that supports tables based on Parquet files.

Correct. Delta Lake provides a relational storage layer in which you can create tables based on Parquet files in a data lake.

A synchronization solution that replicates data between SQL Server and Spark clusters.

2. You've loaded a Spark dataframe with data, that you now want to use in a Delta Lake table. What format should you use to write the dataframe to storage?

CSV

PARQUET

DELTA

Correct. Storing a dataframe in DELTA format creates parquet files for the data and the transaction log metadata necessary for Delta Lake tables.

3. What feature of Delta Lake enables you to retrieve data from previous versions of a table?

Spark Structured Streaming

Time Travel

Correct. The Time Travel feature is based on the transaction log, which enables you to specify a version number or timestamp for the data you want to retrieve.

Catalog Tables

Use SQL Warehouses in Azure Databricks

https://learn.microsoft.com/training/modules/use-sql-warehouses-azure-databricks

Azure Virtual Desktop is a desktop and application virtualization service that runs in the Azure cloud. Azure Virtual Desktop works across devices (Windows, Mac, iOS, Android, and Linux) with apps that you can use to access remote desktops and apps.

This section helps Desktop Infrastructure Architects, Cloud Architects, Desktop Administrators, or System Administrators explore Azure Virtual Desktop and build virtualized desktop infrastructure (VDI) solutions at enterprise scale. Enterprise-scale solutions generally cover 1,000 virtual desktops and above.

Learning objectives

After completing this section, you'll be able to:

•Compare Azure Stack Hub, Azure Stack HCI, and Azure Stack Edge.

•Explain how to integrate hybrid cloud scenarios using Azure Stack Hub.

•Provide an overview of the Azure Stack Hub systems.

•Explain how Azure Stack Hub is managed.

•Identify the key resource providers for Azure Stack Hub.

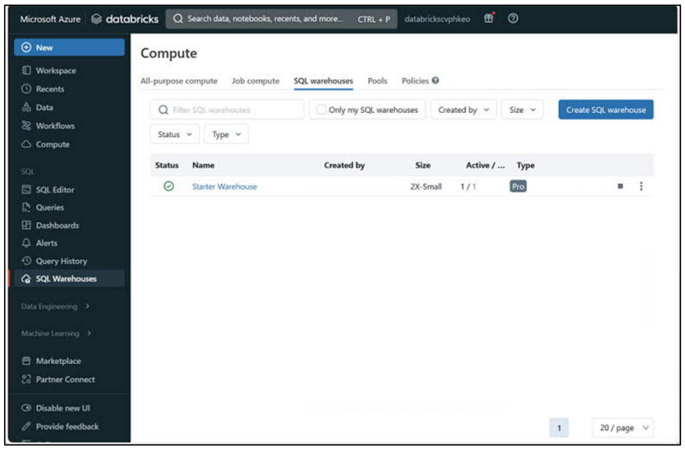
Prerequisites

•Conceptual knowledge of Azure compute solutions.

•Working experience with virtual machines, containers, and app service.

Get started with SQL Warehouses

- Creating a SQL Warehouse
- SQL Warehouse configuration settings



Status	Name	Created by	Size	Active / ...	Type
✓	Starter Warehouse		2X-Small	1 / 1	Pro

https://learn.microsoft.com/training/modules/use-sql-warehouses-azure-databricks/02-sql-warehouses

SQL Warehouses (formerly known as SQL Endpoints) provide a relational database interface for data in Azure Databricks. The data is stored in files that are abstracted by Delta tables in a hive metastore, but from the perspective of the user or client application, the SQL Warehouse behaves like a relational database.

When you create a premium-tier Azure Databricks workspace, it includes a default SQL Warehouse named Starter Warehouse, which you can use to explore sample data and get started with SQL-based data analytics in Azure Databricks. You can modify the configuration of the default SQL Warehouse to suit your needs, or you can create more SQL Warehouses in your workspace.

You can manage the SQL Warehouses in your Azure Databricks workspace by using the Azure Databricks portal in the SQL persona view.

When you create or configure a SQL Warehouse, you can specify the following settings:

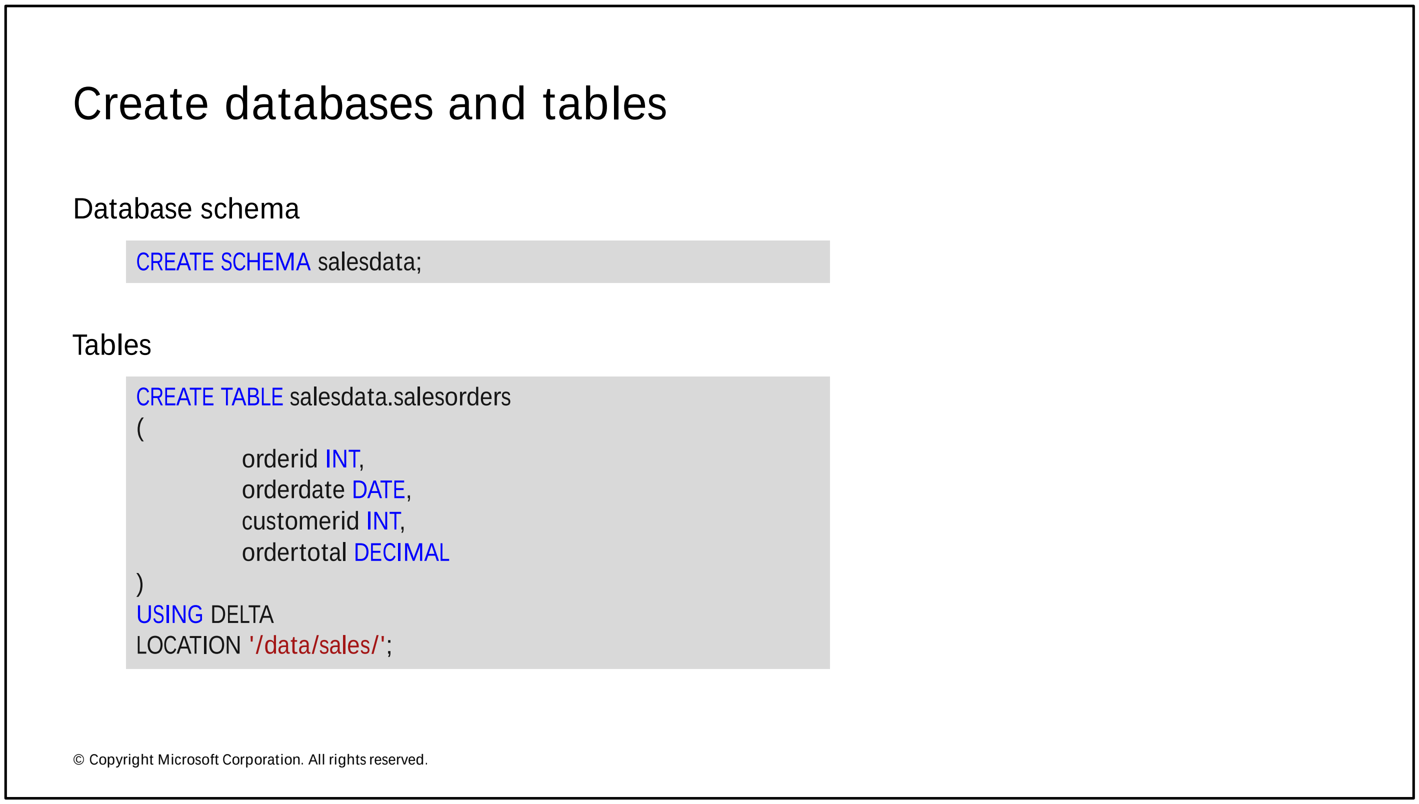
Name: A name used to identify the SQL Warehouse.

Cluster size: Choose from a range of standard sizes to control the number and size of compute resources used to support the SQL Warehouse. Available sizes range from 2X-Small (a single worker node) to 4X-Large (256 worker nodes). For more information, see Cluster size in the Azure Databricks documentation.

Auto Stop: The amount of time the cluster will remain running when idle before being stopped. Idle clusters continue to incur charges when running.

Scaling: The minimum and maximum number of clusters used to distribute query processing.

Type: You can create a SQL Warehouse that uses serverless compute for fast, cost-effective on-demand provisioning. Alternatively, you can create a Pro or Classic SQL warehouse.



Create databases and tables

Database schema

```
CREATE SCHEMA salesdata;
```

Tables

```
CREATE TABLE salesdata.salesorders
(
		orderid INT,
		orderdate DATE,
		customerid INT,
		ordertotal DECIMAL
)
USING DELTA
LOCATION '/data/sales/';
```

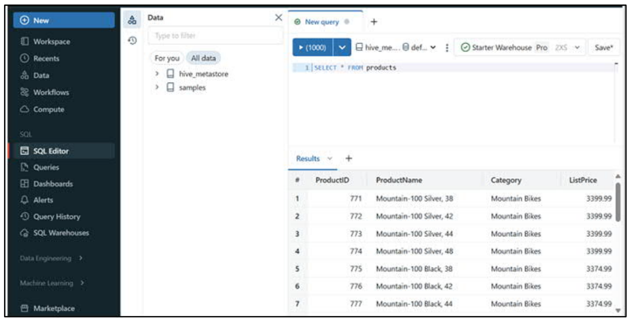
https://learn.microsoft.com/training/modules/use-sql-warehouses-azure-databricks/03-databases-tables

All SQL Warehouses contain a default database schema named default. You can use create tables in this schema in order to analyze data. However, if you need to work with multiple tables in a relational schema, or you have multiple analytical workloads where you want to manage the data (and access to it) separately, you can create custom database schema. To create a database, use the SQL editor to run a CREATE DATABASE or CREATE SCHEMA SQL statement.

You can use the user interface in the Azure Databricks portal to upload delimited data, or import data from a wide range of common data sources. The imported data is stored in files in Databricks File System (DBFS) storage, and a Delta table is defined for it in the Hive metastore.

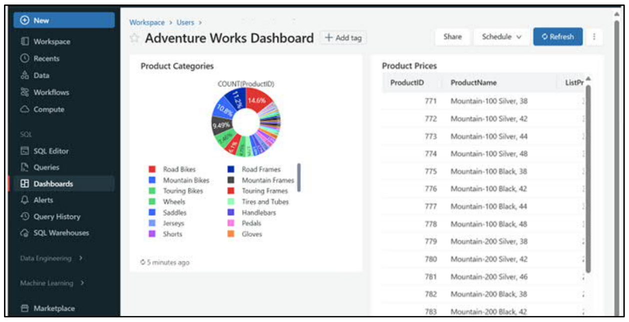
If the data files already exist in storage, or you need to define an explicit schema for the table, you can use a CREATE TABLE SQL statement. For example, the following code creates a table named salesorders in the salesdata database, based on the /data/sales/ folder in DBFS storage.

Create queries and dashboards



```
SELECT * FROM products
```

#	ProductID	ProductName	Category	ListPrice
1	771	Mountain-100 Silver, 38	Mountain Bikes	3399.99
2	772	Mountain-100 Silver, 42	Mountain Bikes	3399.99
3	773	Mountain-100 Silver, 44	Mountain Bikes	3399.99
4	774	Mountain-100 Silver, 48	Mountain Bikes	3399.99
5	775	Mountain-100 Black, 38	Mountain Bikes	3374.99
6	776	Mountain-100 Black, 42	Mountain Bikes	3374.99
7	777	Mountain-100 Black, 44	Mountain Bikes	3374.99



ProductID	ProductName	ListPr
771	Mountain-100 Silver, 38	
772	Mountain-100 Silver, 42	
773	Mountain-100 Silver, 44	
774	Mountain-100 Silver, 48	
775	Mountain-100 Black, 38	
776	Mountain-100 Black, 42	
777	Mountain-100 Black, 44	
778	Mountain-100 Black, 48	
779	Mountain-200 Silver, 38	
780	Mountain-200 Silver, 42	
781	Mountain-200 Silver, 46	
782	Mountain-200 Black, 38	
783	Mountain-200 Black, 42	

Queries

You can use the SQL Editor in the Azure Databricks portal to create a query based on any valid SQL SELECT statement.

Dashboards

Dashboards enable you to display the results of queries, either as tables of data or as graphical visualizations.

https://learn.microsoft.com/training/modules/use-sql-warehouses-azure-databricks/04-queries-dashboards

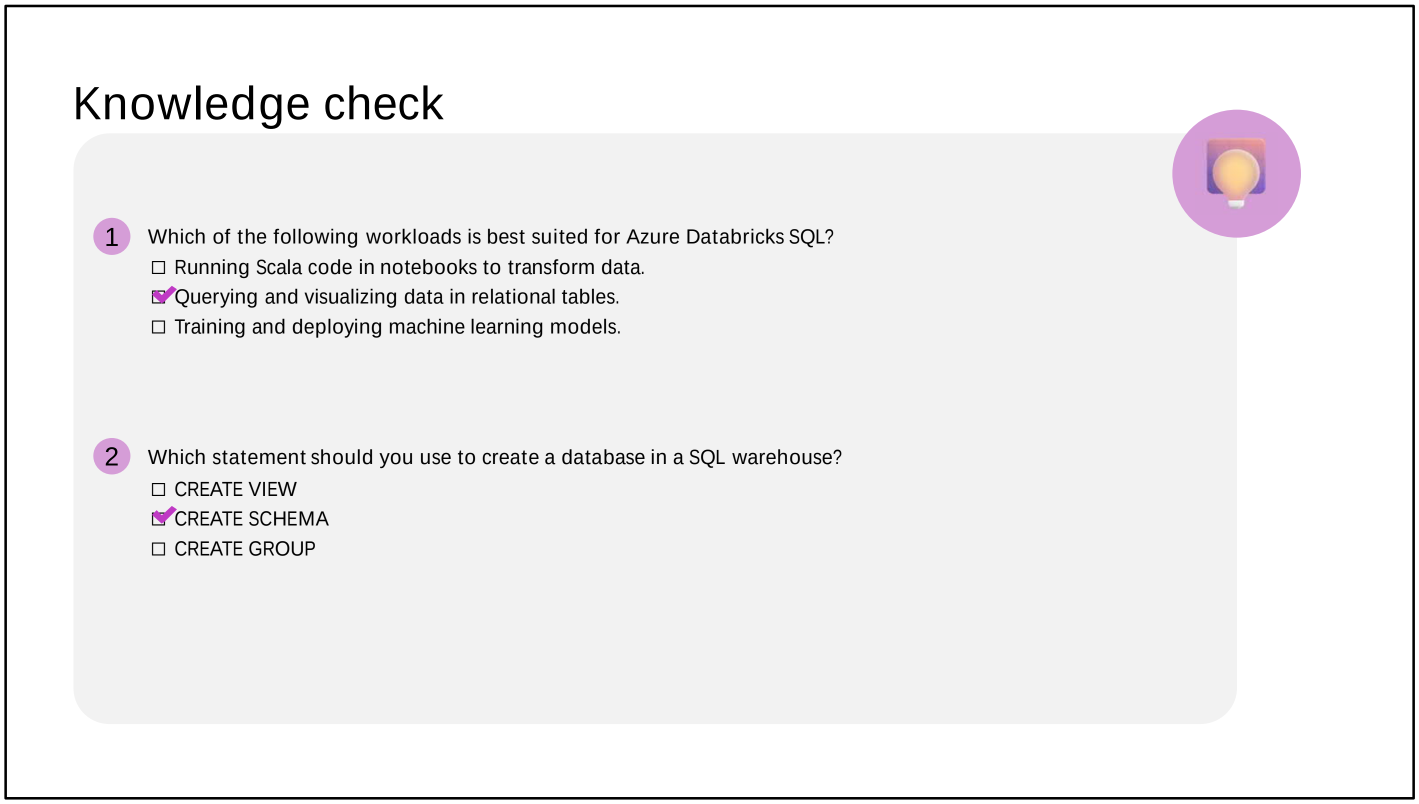
Azure Databricks SQL is primarily designed for data analytics and visualization workloads. To support these workloads, users can create queries to retrieve and summarize data from tables, and dashboards to share visualizations of the data.

You can use the SQL Editor in the Azure Databricks portal to create a query based on any valid SQL SELECT statement, and then save the query with a meaningful name to be retrieved and run later.

After saving the query, you can schedule it to be run automatically at regular intervals to refresh the data, or you can open it and run it interactively.

Dashboards enable you to display the results of queries, either as tables of data or as graphical visualizations.

You can create multiple visualizations in a dashboard and share it with users in your organization. As with individual queries, you can schedule the dashboard to refresh is data periodically, and notify subscribers by email that new data is available.



Knowledge check

1 Which of the following workloads is best suited for Azure Databricks SQL?

- ☐ Running Scala code in notebooks to transform data.
- [✓ Querying and visualizing data in relational tables.]
- ☐ Training and deploying machine learning models.

2 Which statement should you use to create a database in a SQL warehouse?

- ☐ CREATE VIEW
- [✓ CREATE SCHEMA]
- ☐ CREATE GROUP

1. Which of the following workloads is best suited for Azure Databricks SQL?

Running Scala code in notebooks to transform data.
Querying and visualizing data in relational tables.
Correct. Azure Databricks SQL is optimized for SQL-based querying and data visualization.
Training and deploying machine learning models.

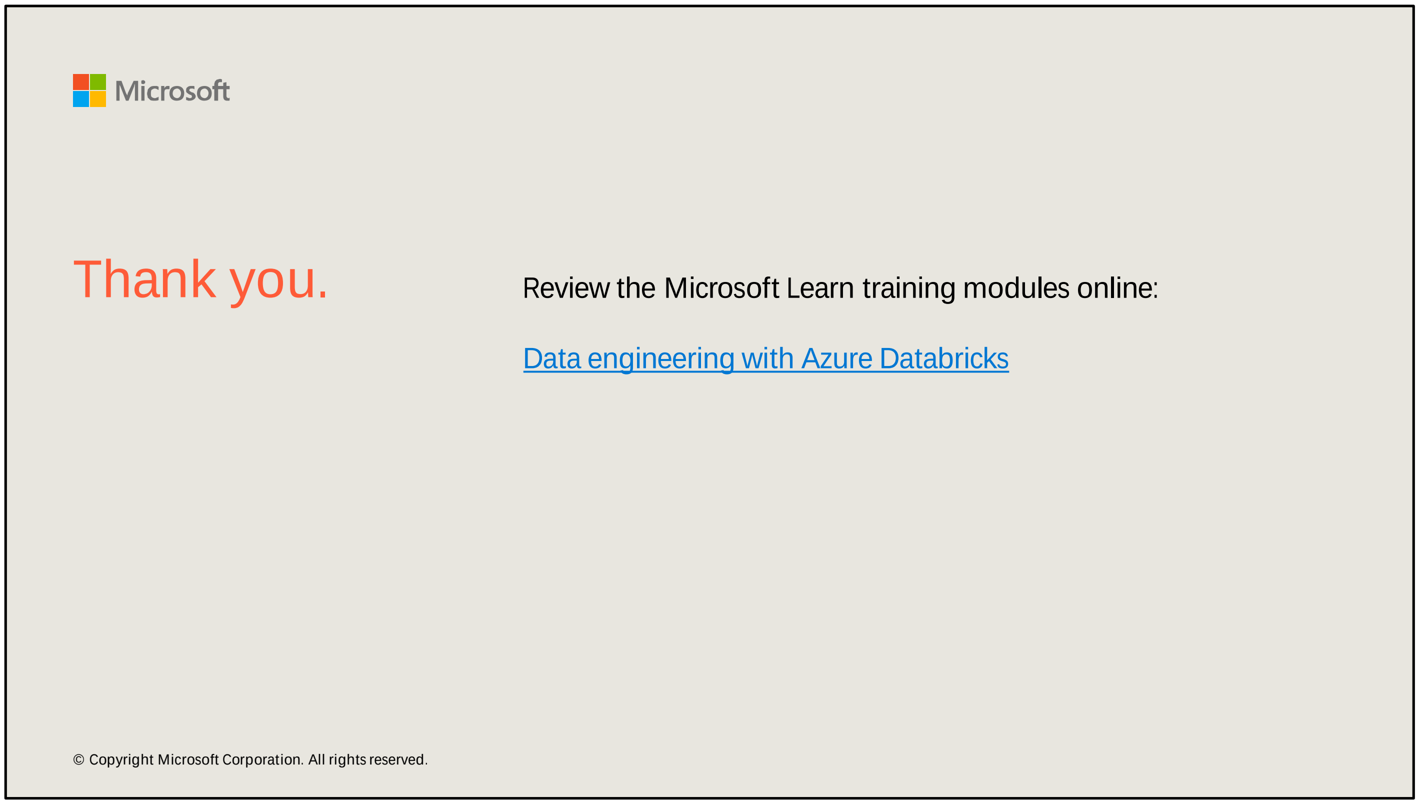
2. Which statement should you use to create a database in a SQL warehouse?

CREATE VIEW
CREATE SCHEMA
Correct. The CREATE SCHEMA statement is used to create a database.
CREATE GROUP

Summary

What you learned

- How to get started with SQL Warehouses
- How to create databases and tables
- How to create queries and dashboards



Microsoft

Thank you.

Review the Microsoft Learn training modules online:

[Data engineering with Azure Databricks]