

# Azure Databricks

---

Data Engineering & Analytics on Databricks Platform

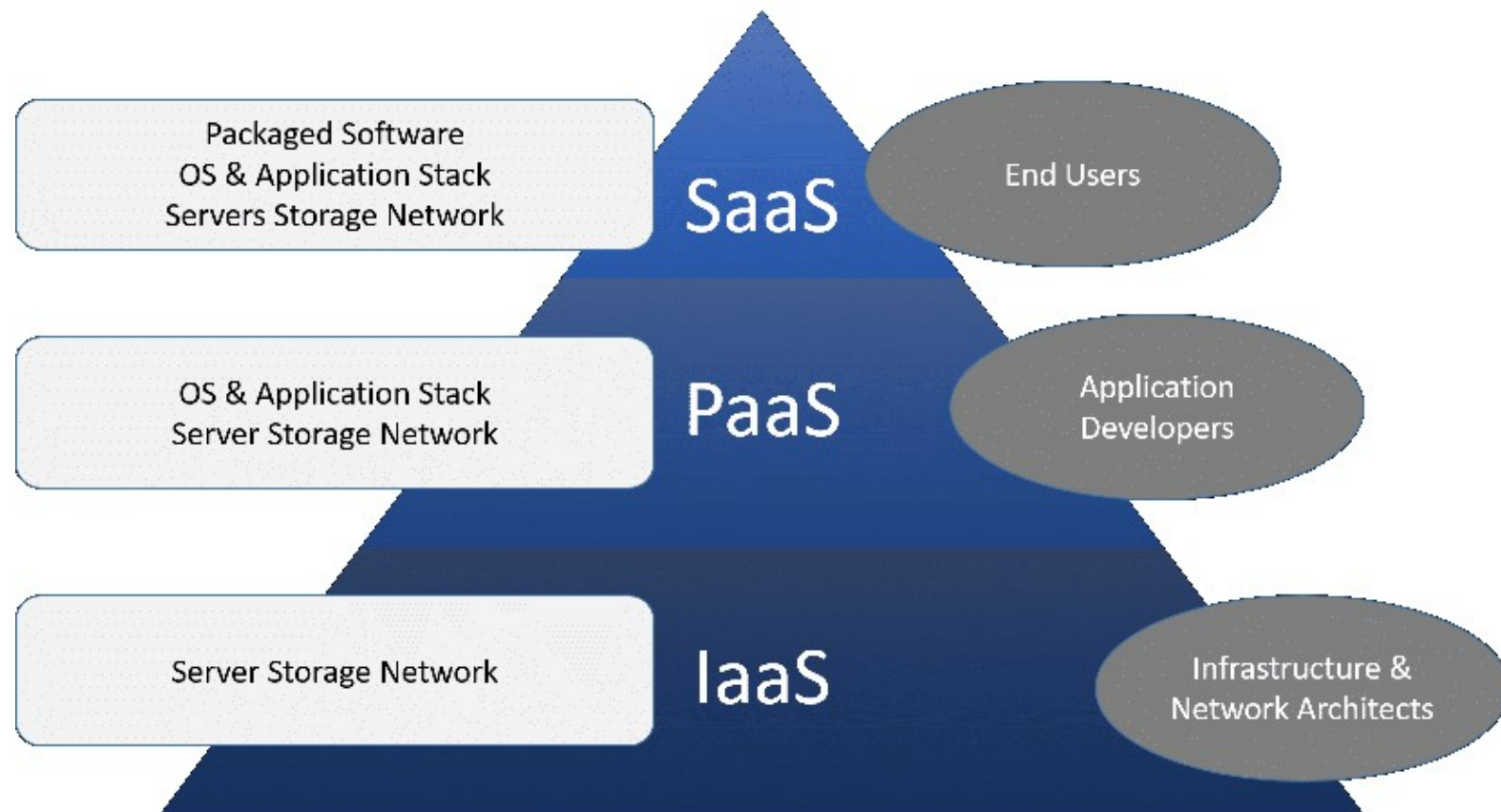
# Cloud computing

- 'Cloud computing' is a term that indicates delivery of computing resources such as servers, storage, network etc. over the internet by a service provider and is accessible from anywhere and at any time.



# Cloud computing service models

---





Microsoft Azure

# Microsoft Azure

---

- Microsoft Azure is a cloud computing platform run by Microsoft, which offers access, management, and development of applications and services through global data centers.
- It provides a range of capabilities, including software as a service (SaaS), platform as a service (PaaS), and infrastructure as a service (IaaS).
- Microsoft Azure supports many programming languages, tools, and frameworks, including Microsoft-specific and third-party software and systems.

# Microsoft Azure benefits

---

- Flexible
  - Move compute resources up and down as needed
- Open
  - Supports almost any OS, language, tool, or framework
- Reliable
  - 99.95% availability SLA and 24x7 tech support
- Global
  - Data housed in geo-synchronous data centers
- Economical
  - Only pay for what you use

# Azure Pricing

---

- Azure pricing depends on many factors, including the type of service, the capacity required, the location, and the level of management.
- Azure offers a free tier, which provides free use of certain services for the first 12 months, and free forever usage of specific services.
- The majority of Azure services can be purchased under the “pay as you go” pricing model, which charges users based on actual usage. Additionally, Azure offers significant discounts for reserved instances (which require commitment of 1 or 3 years), and spot instances (virtual machines from Azure’s spare capacity, which can be terminated at short notice).

# Azure Pricing – Free Tier

---

- Azure offers a free tier that lets companies use a variety of services free for 12 months (with a limited allowance), provides a credit for additional services in the first 30 days, and provides several services for free on an ongoing basis.
- **12 Months Free Use**
  - Azure provides the following services free for the first 12 months after creating an Azure account. Each service has a usage limit—for example, you can use a Windows Virtual Machine for 750 hours.

<b>Compute Services</b> Linux Virtual Machines Windows Virtual Machines	<b>Storage Services</b> Azure Managed Disks Azure Blob Storage Azure Files
<b>Database Services</b> Azure SQL Database (MS SQL Server) Cosmos DB (NoSQL)	<b>AI and Analytics Services</b> Computer Vision Text Analytics Translator Personalizer Language Understanding



# Azure Pricing – Free Tier

---

## **Credit for first 30 days**

- If you want to use a service that is not included in the free services, or if you exceed the service limits in the free tier, Azure provides a credit of \$200 which you can deduct from your first bill, during the first 30 days of usage.
- Beyond that, any further use of Azure services will be billed.

## **Azure charges after the free trial**

- Certain Azure services are available free, within certain limitations, during the first 12 months.
- Azure also provides a \$200 credit for new customers. When the first 12 months end and/or when you finish using the \$200 credit, you will be charged for all Azure services you use.
- The only exception is services that are provided free forever

# Free Azure Services

---

- The following services are always free, even after the first 12 months of use.
- Keep in mind that you are billed for use of other Azure resources you consume during the use of these services.
  - For example, when deploying containers using Azure Kubernetes Service (AKS), you are billed for the VMs/containers that are deployed by the service. But there is no charge for AKS itself.

<b>Development Services</b> Azure App Service DevTest Labs Azure DevOps	<b>Serverless and Containers</b> Azure Functions (free up to 1 million requests) Azure Kubernetes Service (AKS) Azure Container Instances Service Fabric
<b>Messaging, Routing and Automation</b> Event Grid Load Balancer Azure Automation	<b>Networking</b> Virtual Networks (VNets) Data transfer between VNets Inbound data transfer (unlimited) Outbound data transfer (up to 15GB)
<b>Data Management and Search</b> Data Factory Data Catalog Cognitive Search	<b>Other Services</b> Active Directory B2C Azure Security Center Azure Advisor

# Azure pricing models

---

Microsoft offers three ways to pay for Azure VMs and other cloud resources:

- Pay as you go
  - Pay for services according to actual usage, billed per second, with no long-term commitment or upfront payments. You get complete flexibility to increase or decrease resources as needed. Azure VMs can be automatically scaled up and down using Azure's auto-scaling features.
  - Suitable for users who prefer flexibility and applications with volatile or short-term workloads.
- Reserved instances
  - Azure provides Reserved Virtual Machine Instances (RVMI)—virtual machines that are pre-purchased for one or three years in a specific region. Committing to reserved instances in advance grants a discount of up to 72% compared to pay-as-you-go prices.
- Spot instances
  - You can buy unused computing power at a discount of up to 90% compared to pay as you go prices. However, spot instances can be interrupted on short notice, so they are considered to be suitable only for workloads that can tolerate disruptions.
  - Suitable for distributed, fault tolerant applications, stateless applications, workloads that are not urgent, or are heavily parallelized.



# Apache Spark

# Apache Spark

---

- Apache Spark is an open-source, distributed processing system used for big data workloads.
- Apache Spark utilizes in-memory caching, and optimized query execution for fast analytic queries against data of any size.
- Apache Spark provides development APIs in Java, Scala, Python and R, and supports code reuse across multiple workloads—batch processing, interactive queries, real-time analytics, machine learning, and graph processing.



# Apache Spark

---

- Hadoop MapReduce is a programming model for processing big data sets with a parallel, distributed algorithm. MapReduce uses sequential multi-step process to run a job. Because each step requires a disk read, and write, MapReduce jobs are slower due to the latency of disk I/O.
- Spark was created to address the limitations to MapReduce, by doing processing in-memory, reducing the number of steps in a job, and by reusing data across multiple parallel operations.
- With Spark, only one-step is needed where data is read into memory, operations performed, and the results written back—resulting in a much faster execution.

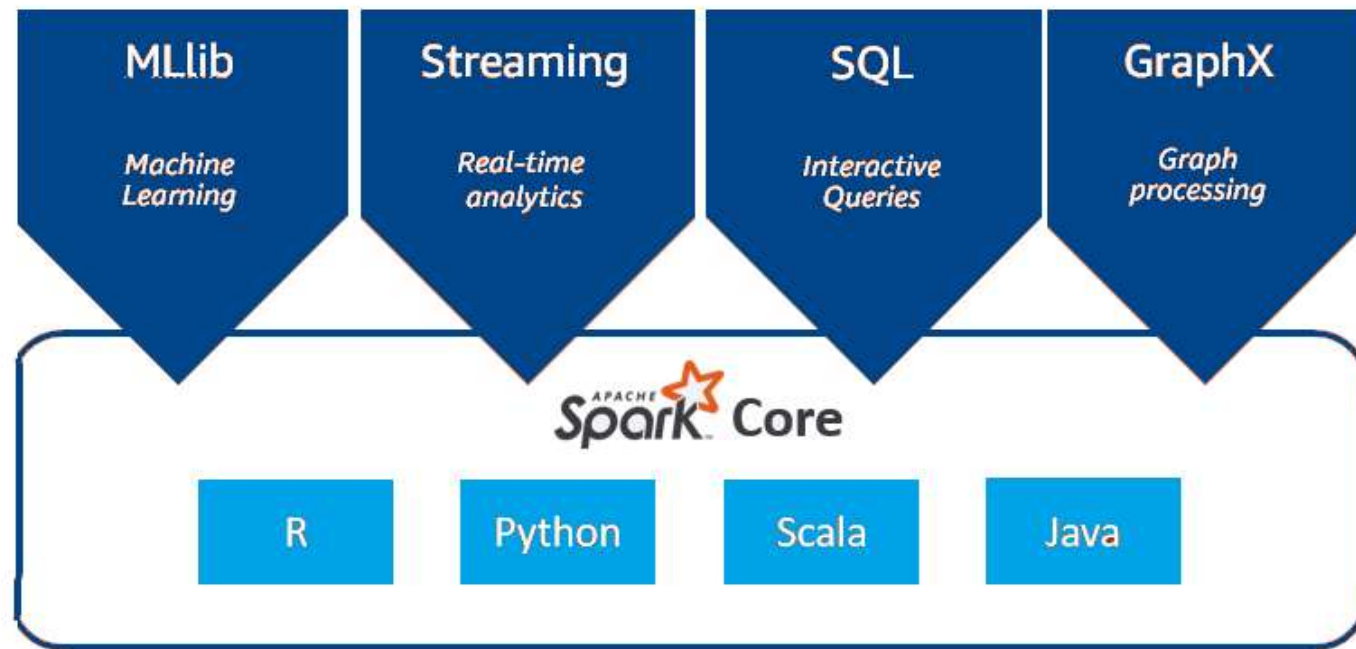
# Apache Spark

---

- Spark also reuses data by using an in-memory cache to greatly speed up machine learning algorithms that repeatedly call a function on the same dataset.
- Data re-use is accomplished through the creation of DataFrames, an abstraction over RDD, which is a collection of objects that is cached in memory, and reused in multiple Spark operations. This dramatically lowers the latency making Spark multiple times faster than MapReduce, especially when doing machine learning, and interactive analytics.

# Apache Spark Workloads

---



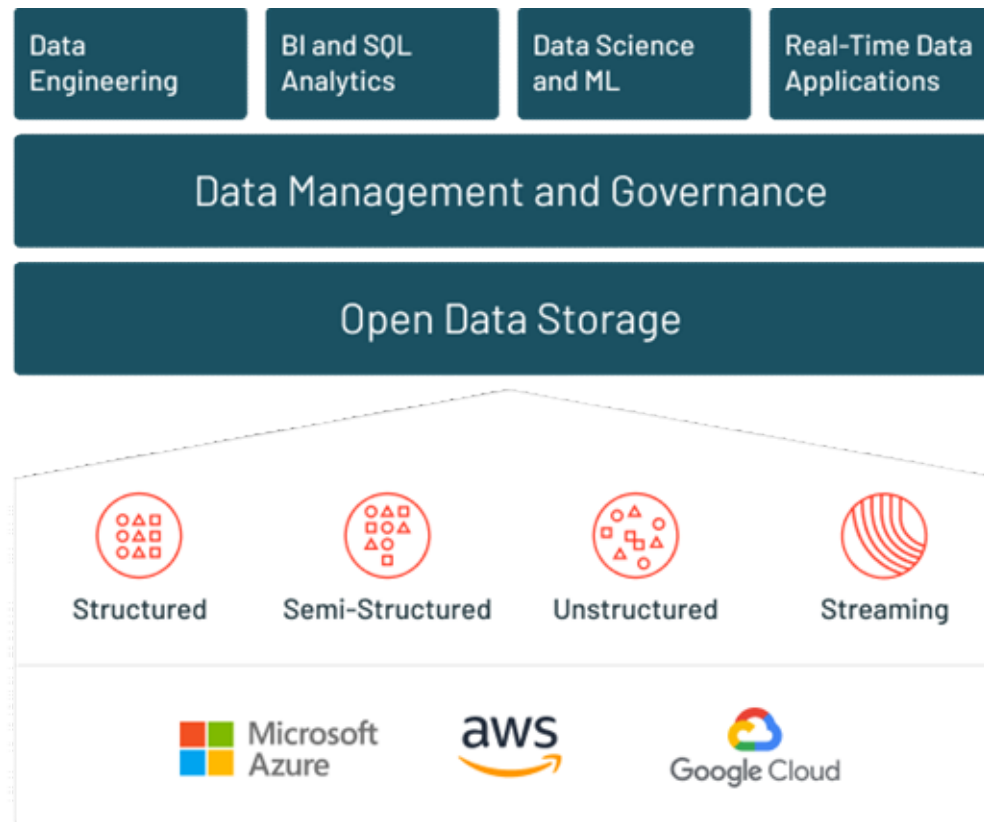




# Azure Databricks

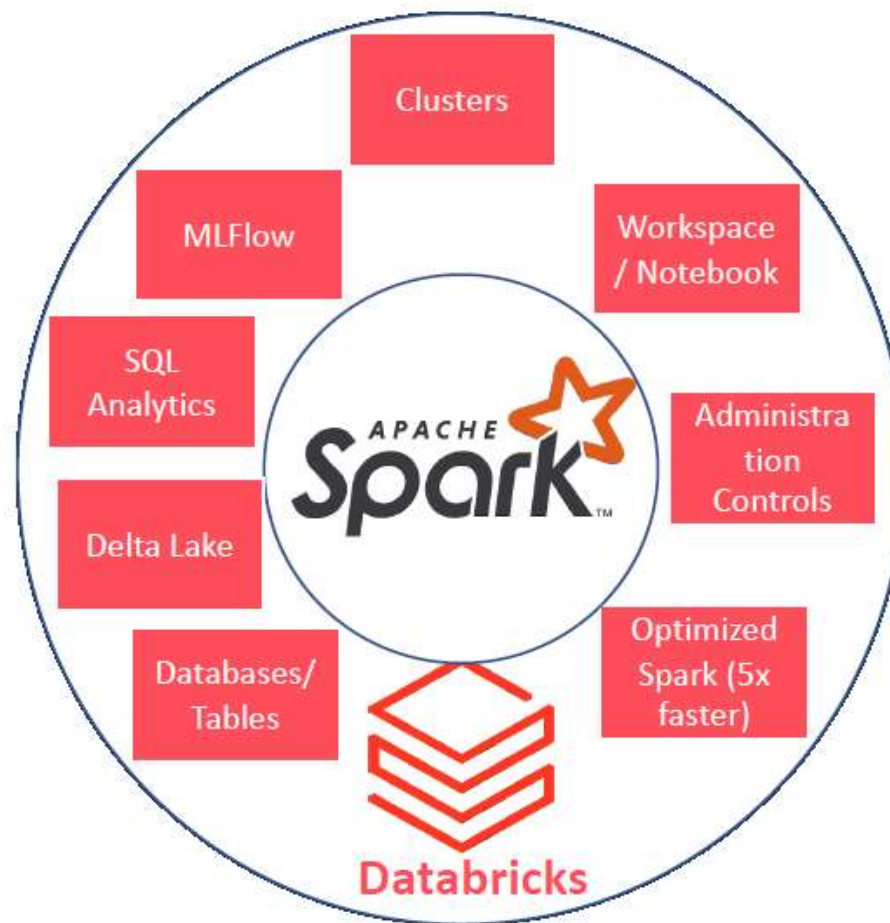
# What is Databricks ?

- **Databricks** is a cloud-based platform for managing and analyzing large datasets using Apache Spark.



# What is Databricks ?

---



# Databricks features

---

- **Unified Workspace**

- Databricks provides a single platform for data scientists, engineers, and business analysts to work together and collaborate on data projects.

- **Scalability and Flexibility**

- Databricks is designed to be highly scalable so that they can easily handle large amounts of data.
- It can also be flexibly configured to support different workloads, such as batch processing, real-time streaming, or machine learning.
- This makes it a good choice for organizations that need to process data at different scales, or that have complex data pipelines.

- **Integrated Tools and Services**

- Databricks comes with a range of tools and services for working with big data, including support for various data formats, integration with popular data science libraries and frameworks, and tools for data preparation and analysis.
- This makes it easier to build and deploy data-driven applications, without having to worry about setting up and managing complex infrastructure.

- **Security and Compliance**

- Databricks offers a range of features to help ensure that data is handled securely and in compliance with relevant regulations.
- This includes support for encryption, role-based access control, and auditing, as well as integration with popular security and compliance tools.

# Databricks Use Cases

---

- **Data Warehousing**

- Databricks can be used to store and manage large amounts of data from multiple sources, and provide fast and efficient access to the data for analysis and other purposes.

- **Data Preparation**

- Databricks provides tools and services for cleaning, transforming, and enriching data, making it easier to prepare data for analysis or other uses.

- **Data Analysis**

- Databricks offers a range of tools and services for exploring, visualizing, and analyzing data, so that users can gain insights and identify trends in the data.

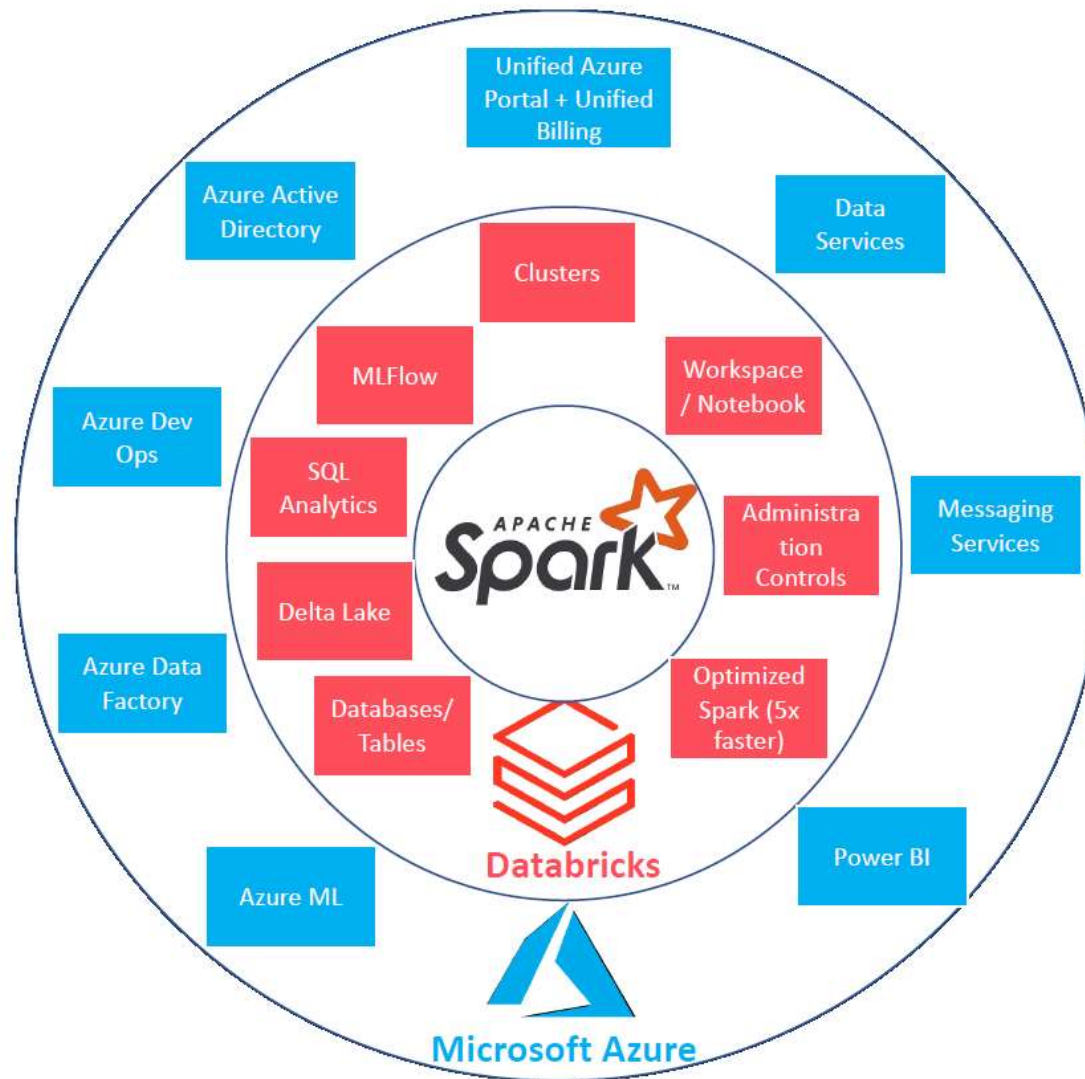
- **Machine Learning**

- Databricks supports popular machine learning libraries and frameworks, such as TensorFlow, Keras, and PyTorch, making it easier to build and train ML models on large datasets.

- **Real-time Data Processing**

- Databricks can be used to process streaming data in real-time so that users can take action based on the data as it arrives.

# Azure Databricks

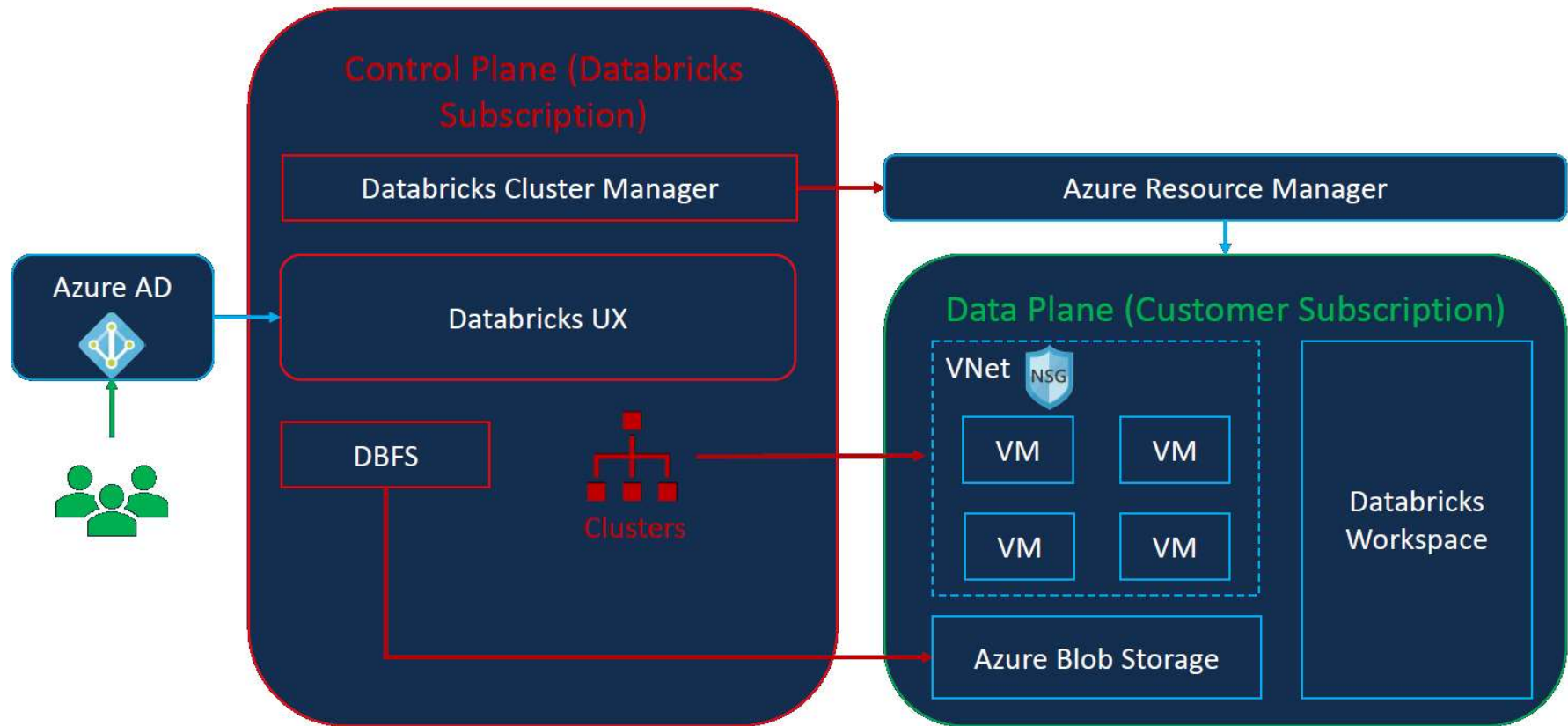


# Azure Databricks architecture

---

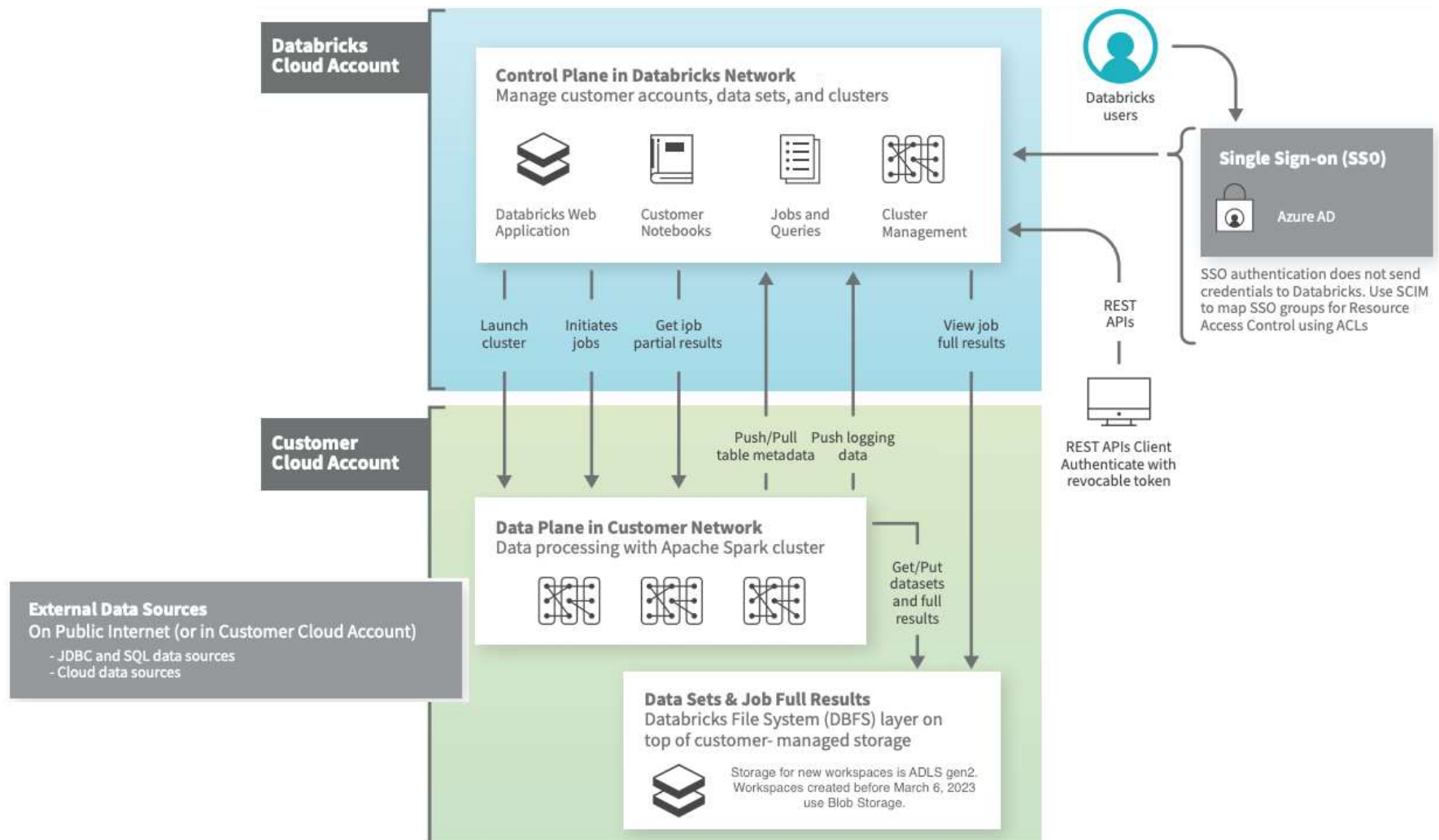
- Azure Databricks is structured to enable secure cross-functional team collaboration while keeping a significant amount of backend services managed by Azure Databricks so you can stay focused on your data science, data analytics, and data engineering tasks.
- Azure Databricks operates out of a ***control plane*** and a ***data plane***.
- - **Control plane**
    - The control plane includes the backend services that Azure Databricks manages in its own Azure account. Notebook commands and many other workspace configurations are stored in the control plane and encrypted at rest.
  - **Data plane**
    - Your Azure account manages the data plane, and is where your data resides. This is also where data is processed. Use Azure Databricks connectors to connect clusters to external data sources outside of your Azure account to ingest data, or for storage. You can also ingest data from external streaming data sources, such as events data, streaming data, IoT data, and more.

# Azure Databricks architecture





# Azure Databricks architecture



# Getting started with Azure Databricks

---

Before you begin, do the following:

- You must have an Azure subscription that isn't a Free Trial Subscription. If you have a free account, complete the following steps:
  - Go to your profile and change your subscription to pay-as-you-go.
  - Remove the spending limit.
  - Request a quota increase for vCPUs in your region.
- Sign in to the Azure portal.
- You must be an Azure Contributor or Owner.

Reference URL: <https://learn.microsoft.com/en-us/azure/databricks/getting-started/>

# Azure Databricks Workspace

---

- A workspace is an environment for accessing all of your Azure Databricks assets. A workspace organizes objects (notebooks, libraries, dashboards, and experiments) into folders and provides access to data objects and computational resources.
- Databricks recommends you deploy your first Azure Databricks workspace using the Azure portal. You can also deploy Azure Databricks with one of the following options:
  - Azure CLI
  - Powershell
  - Azure Resource Manager (ARM) template
    - An ARM template is a JavaScript Object Notation (JSON) file that defines the infrastructure and configuration for your project. The template uses declarative syntax, which lets you state what you intend to deploy without having to write the sequence of programming commands to create it.
  - Bicep
    - Bicep is a domain-specific language (DSL) that uses declarative syntax to deploy Azure resources. It provides concise syntax, reliable type safety, and support for code reuse. Bicep offers the best authoring experience for your infrastructure-as-code solutions in Azure.

# Lab 1 – Setup Azure account

---

Instruction document: L01-Setup-Azure-Account.txt

In this lab we do the following:

- Create a 'Pay as you go' Azure account.
- Login to Azure portal with your account.
- Check your subscription and register for a few 'Resource providers'
- Update the quotas for the compute service in 'East US' region

# Lab 2 - Getting started with Azure Databricks

---

Instruction document: [L02-Getting-Started-With-AzureDatabricks.txt](#)

In this lab we do the following:

- Create and launch 'Azure Databricks' workspace
- Create a single node Databricks cluster
- Upload data files
- Create a new Databricks Notebook
- Create a Spark application using Azure Databricks Notebook
- Export and Import of Azure Databricks Notebooks
- Delete the Databricks cluster
- Delete Azure Databricks Workspace

# Azure Resource Group

---

- Resource group is a container that holds resources for an Azure solution.
  - The resource group can include all the resources for the solution, or only those resources that you want to manage as a group.
  - You decide how you want to allocate resources to resource groups based on what makes the most sense for your organization.
  - Generally, add resources that share the same lifecycle to the same resource group so you can easily deploy, update, and delete them as a group.
- The resource group stores metadata about the resources.
  - Specifying a location for the resource group means specifying where that metadata is stored.
  - For compliance reasons, you may need to ensure that your data is stored in a particular region.

# Azure Storage Account

---

- An Azure storage account contains all of your Azure Storage data objects: blobs, files, queues, and tables. It provides a unique namespace for your Azure Storage data that's accessible from anywhere in the world over HTTP or HTTPS.
- Data in your storage account is durable and highly available, secure, and massively scalable.
- Types of storage accounts
  - Standard general-purpose v2
  - Premium block blobs
  - Premium file shares
  - Premium page blobs

**For more details:** <https://learn.microsoft.com/en-us/azure/storage/common/storage-account-overview#types-of-storage-accounts>

# Databricks Cluster

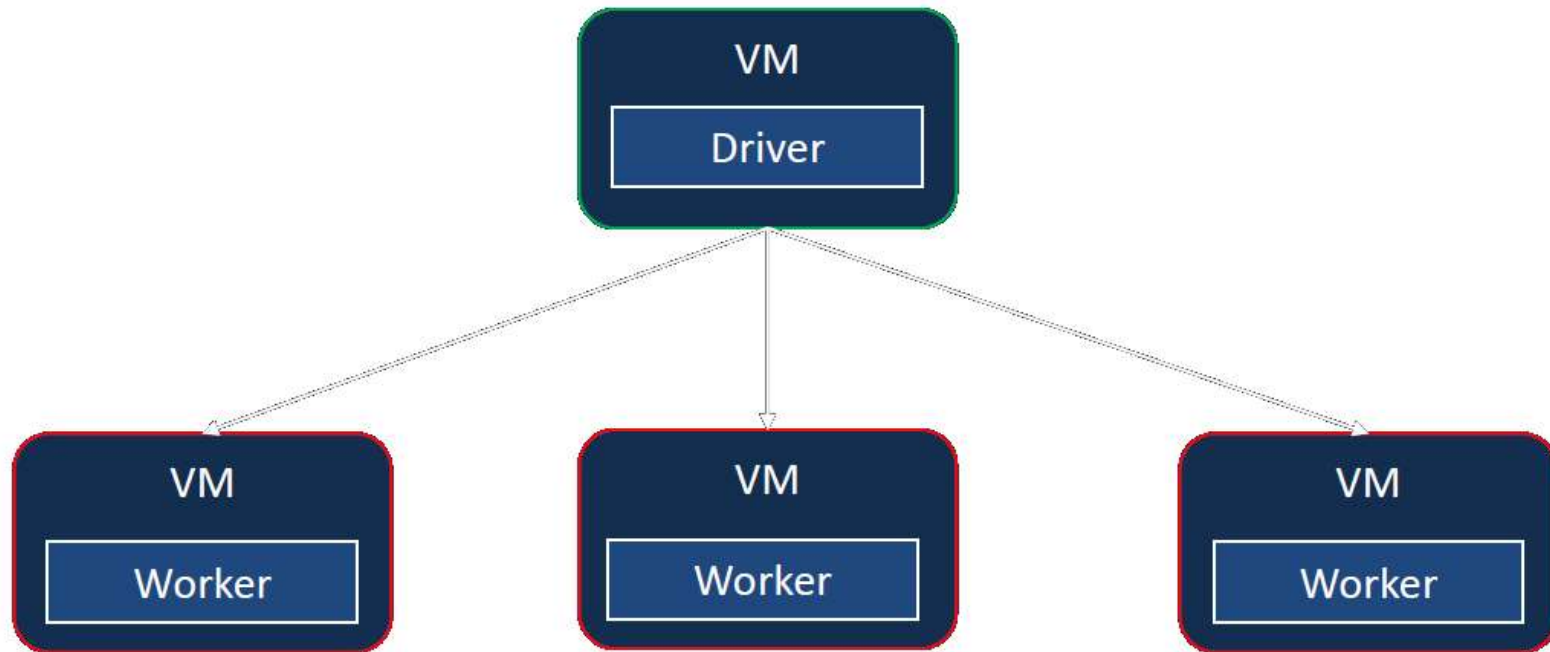
---

- A Databricks cluster is a set of computation resources and configurations on which you run data engineering, data science, and data analytics workloads.
- You run these workloads as a set of commands in a **notebook** or as an **automated job**.
  - Databricks makes a distinction between all-purpose clusters and job clusters.
  - All-purpose clusters are used to analyze data collaboratively using interactive notebooks.
  - Job clusters are used to run fast and robust automated jobs.



# Databricks Cluster

---



Multi-node Databricks Cluster

# Databricks cluster types

---

There are two types of clusters in Databricks:

All Purpose	Job Cluster
Created manually	Created by Jobs
Persistent	Terminated at the end of the job
Suitable for interactive workloads	Suitable for automated workloads
Shared among many users	Isolated just for the job
Expensive to run	Cheaper to run

# Databricks cluster access modes

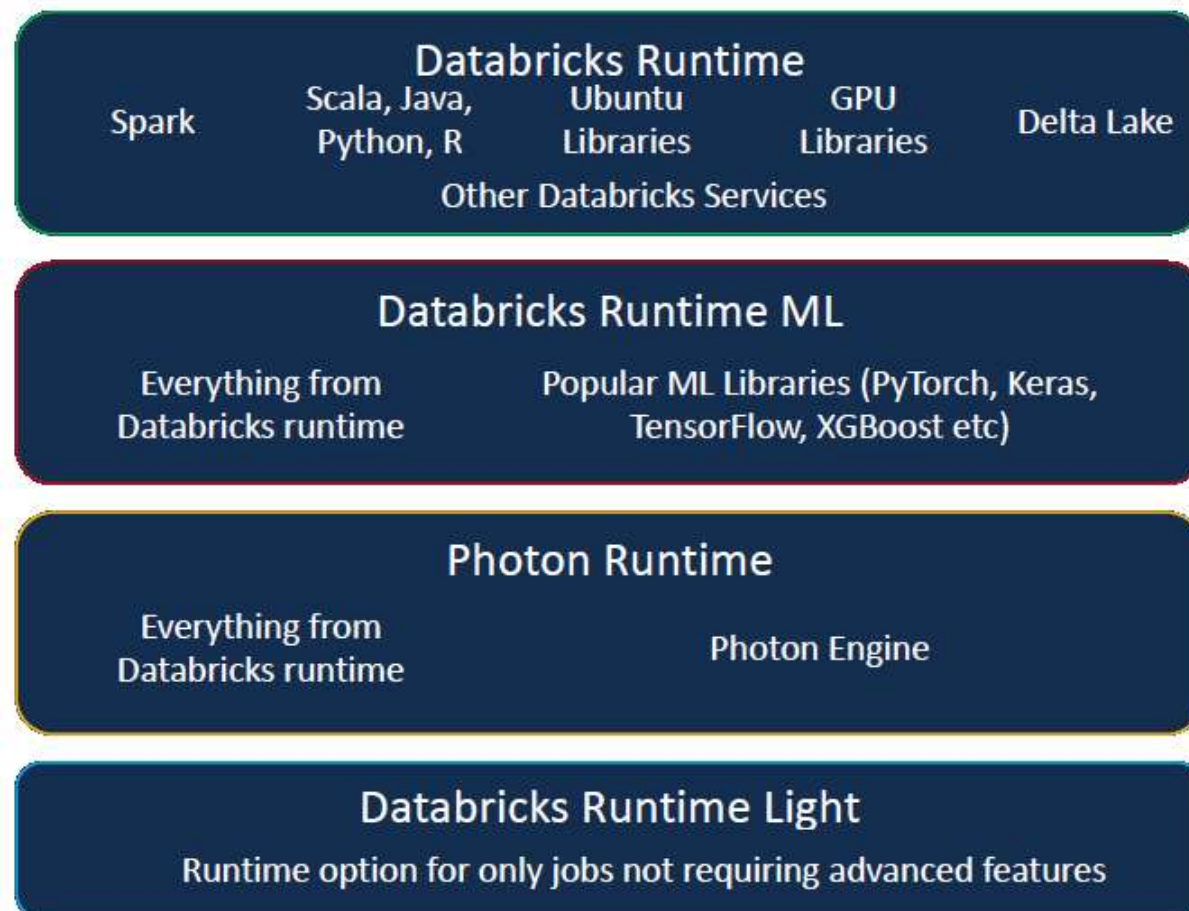
**Cluster access mode** is a security feature that determines who can use a cluster and what data they can access via the cluster.

Access Mode	Visible to user	UC Support	Supported Languages	Notes
Single user	Always	Yes	Python, SQL, Scala, R	Can be assigned to and used by a single user. See <a href="#">single user limitations</a> .
Shared	Always (Premium plan required)	Yes	Python (on Databricks Runtime 11.1 and above), SQL	Can be used by multiple users with data isolation among users. See <a href="#">shared limitations</a> .
No Isolation Shared	Admins can hide this cluster type by <a href="#">enforcing user isolation</a> in the admin settings page.	No	Python, SQL, Scala, R	There is a <a href="#">related account-level setting</a> for No Isolation Shared clusters.
Custom	Hidden (For all new clusters)	No	Python, SQL, Scala, R	This option is shown only if you have existing clusters without a specified access mode.

# Databricks runtime options

---

Databricks Runtime is the set of software artifacts that run on the clusters of machines managed by Databricks.



# Compute policy

A compute policy defines limits on the attributes available during compute creation

Policy ?

Unrestricted | ▼

Unrestricted

Personal Compute

Power User Compute

Legacy Shared Compute

Performance

Databricks runtime version ?

Runtime: 13.3 LTS (Scala 2.12, Spark 3.4.1) | ▼

All-purpose compute

Job compute

SQL warehouses

Pools

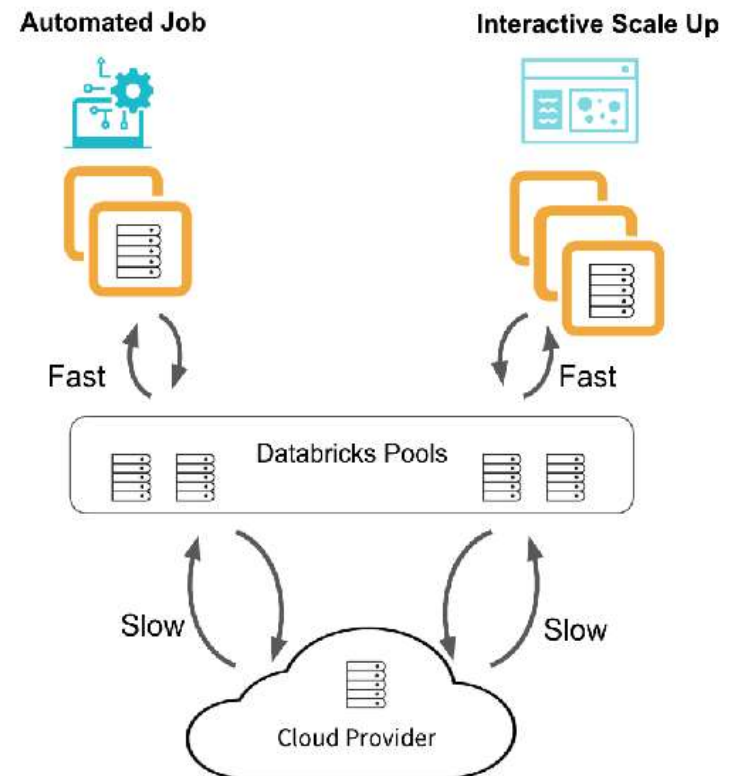
**Policies ?**

Q Filter policies

Name ≡↑	Definitions	Policy Family
Job Compute	9	Job Compute
Legacy Shared Compute	13	Legacy Shared Compute
Personal Compute	11	Personal Compute
Power User Compute	12	Power User Compute
Shared Compute ⓘ	13	Shared Compute

# Cluster pools

- Azure Databricks cluster pools are a set of idle, ready-to-use instances.
- When cluster nodes are created using the idle instances, cluster start and auto-scaling times are reduced.
- If the pool has no idle instances, the pool expands by allocating a new instance from the instance provider in order to accommodate the cluster's request.
- When a cluster releases an instance, it returns to the pool and is free for another cluster to use. Only clusters attached to a pool can use that pool's idle instances.



# Create pools based on workloads

---

- If your driver node and worker nodes have different requirements, create a different pool for each.
- You can minimize instance acquisition time by creating a pool for each instance type and Azure Databricks runtime your organization commonly uses.
  - For example, if most data engineering clusters use instance type A, data science clusters use instance type B, and analytics clusters use instance type C, create a pool with each instance type.
- Configure pools to use on-demand instances for jobs with short execution times and strict execution time requirements. Use on-demand instances to prevent acquired instances from being lost to a higher bidder on the spot market.
- Configure pools to use spot instances for clusters that support interactive development or jobs that prioritize cost savings over reliability.

# Using the pool for your cluster

Clusters are attached to the pool. You can specify a different pool for the driver node and worker nodes, or use the same pool for both.

☒ Multi node ☐ Single node

Access mode ?

Single user access ?

Single user

Kanakaraju Yarakaraju (kanakaraj...)

## Performance

Databricks runtime version ?

Runtime: 12.2 LTS (Scala 2.12, Spark 3.3.2)

☐ Use Photon Acceleration ?

Worker type ?

Min workers

Max workers

CTS Demo Pool

Standard\_F4

2

8



# Lab 3 – All purpose cluster, job cluster & instance pool

---

Instruction document: [L03-Allpurpose-and-Job-Clusters-Pools.txt](#)

In this lab we do the following:

- Spin up an all-purpose compute cluster
- Create a notebook
- Create a job and attach the notebook to it
- Run the job using the all-purpose compute cluster
- Modify the job to run it using job cluster
- Create an instance pool
- Create an all-purpose compute cluster in the pool
- Run the job using the all-purpose compute using the pool.

# Databricks Databases & Tables

---

- Databricks Database is a collection of tables.
- Databricks Table is a collection of structured data.
  - We can cache, filter, and perform any operations supported by Spark DataFrames on Azure Databricks tables and query tables with Spark APIs and Spark SQL.
  - There are two types of tables.
    1. Global Table
      - A global table is available across all clusters.
      - Databricks registers global tables to either the Hive metastore or an external metastore.
      - These are persistent tables (i.e. not temporary tables)
    2. Local Table
      - A local table is not accessible from other clusters
      - Not registered in the Hive metastore. (i.e. not created in any database)
      - This is also known as a temporary view.

# Lab 4 – Working with databases & tables

---

Instruction document: L04-Working-With-Databases-Tables.txt

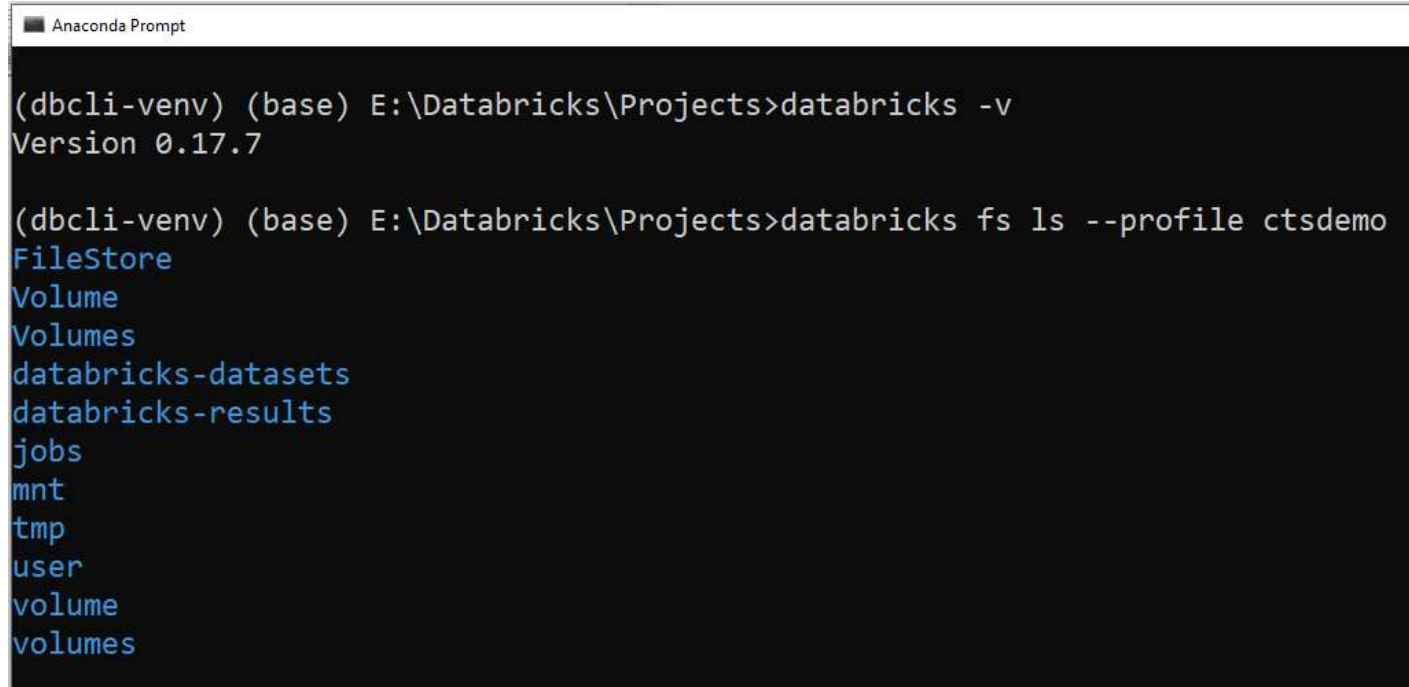
In this lab we do the following:

- Understand the default hive warehouse structure.
- Perform database operations – create, describe, drop database.
- Perform basic table operations – create, load, insert, describe, select etc.
- Work with External tables
- Work with Partitioned tables
- Work with local tables (Temporary views)

# Databricks CLI

---

- The Databricks command-line interface (Databricks CLI) utility provides an easy-to-use interface to automate the Databricks platform from your terminal, command prompt, or automation scripts.



```
Anaconda Prompt
(dbcli-venv) (base) E:\Databricks\Projects>databricks -v
Version 0.17.7

(dbcli-venv) (base) E:\Databricks\Projects>databricks fs ls --profile ctsdemo
FileStore
Volume
Volumes
databricks-datasets
databricks-results
jobs
mnt
tmp
user
volume
volumes
```

# Databricks File System (DBFS)

---

- DBFS is a distributed file system mounted into a Databricks workspace and available on Databricks clusters.
- DBFS is an abstraction on top of scalable object storage that maps file-system calls to native cloud storage API calls.
- DBFS Root is the default storage for a Databricks workspace created during workspace deployment.
  - DBFS Root contains a folder called “**FileStore**” into which we can upload files from Databricks UI.
  - DBFS Root contains two folders
    - `dbfs:/databricks-datasets` → Has sample datasets
    - `dbfs:/databricks-results` → Stores results of the queries executed.
  - DBFS Root is deleted when the workspace is deleted. Hence, it is advised not to store any customer data here. Instead, store customer data in Azure Blob storage or ADLS.

# What can you do with DBFS?

---

- DBFS provides convenience by mapping cloud object storage URIs to relative paths.
- Allows you to interact with object storage using directory and file semantics instead of cloud-specific API commands.
- Allows you to mount cloud object storage locations so that you can map storage credentials to paths in the Databricks workspace.
- Simplifies the process of persisting files to object storage, allowing virtual machines and attached volume storage to be safely deleted on cluster termination.
- Provides a convenient location for storing init scripts, JARs, libraries, and configurations for cluster initialization.

# Lab 5 – Setup Databricks CLI

---

Instruction document: L05-Setup-Databricks-CLI-Python-Virtual-Env.txt

In this lab we do the following:

- Create a Python Virtual Environment
- Install Databricks CLI in the Virtual environment
- Validate the installation
- Configure a profile with Databricks CLI for Azure Databricks workspace

# Lab 6 – Working with DBFS

---

## Instruction documents:

- L06a-Working-with-DBFS-using-Databricks-UI.txt
- L06b-Working-with-DBFS-using-fs-magic-command.txt
- L06c-Working-with-DBFS-using-dbutils.txt
- L06d-Working-with-DBFS-using-Databricks-CLI.txt

In this lab we do the following:

- Perform DBFS file operations Databricks UI
- Perform DBFS file operations using **%fs** magic command in a notebook.
- Perform DBFS file operations using **dbutils** package in a python notebook.
- Perform DBFS file operations from Databricks CLI.



# Azure CLI

---

- The Azure Command-Line Interface (CLI) is a cross-platform command-line tool to connect to Azure and execute administrative commands on Azure resources.
- It allows the execution of commands through a terminal using interactive command-line prompts or a script.
- For interactive use, you first launch a shell such as `cmd.exe` on Windows, or `Bash` on Linux or Mac OS, and then issue a command at the shell prompt. To automate repetitive tasks, you assemble the CLI commands into a shell script.
- You can install the Azure CLI locally on Linux, Mac, or Windows computers. It can also be used from a browser through the Azure Cloud Shell.

# Lab 7 – Setup Azure CLI on Windows

---

Instruction document: L07-Setup-Azure-CLI-on-Windows.txt

In this lab we do the following:

- Install Azure CLI on Windows.
- Validate Azure CLI on Windows.
- Perform a few basic CLI operations.

# Lab 8 – Working with Storage Account using Azure CLI

---

Instruction document: L08-Azure-CLI-Storage-Account.txt

In this lab we do the following:

- Create a new Resource Group
- Create a Storage Account in the Resource Group
- Add ADLS file system in the Storage Account
- Load files into the ADLS file system (containers)
- Cleanup the resources

# Azure Blob Storage

---

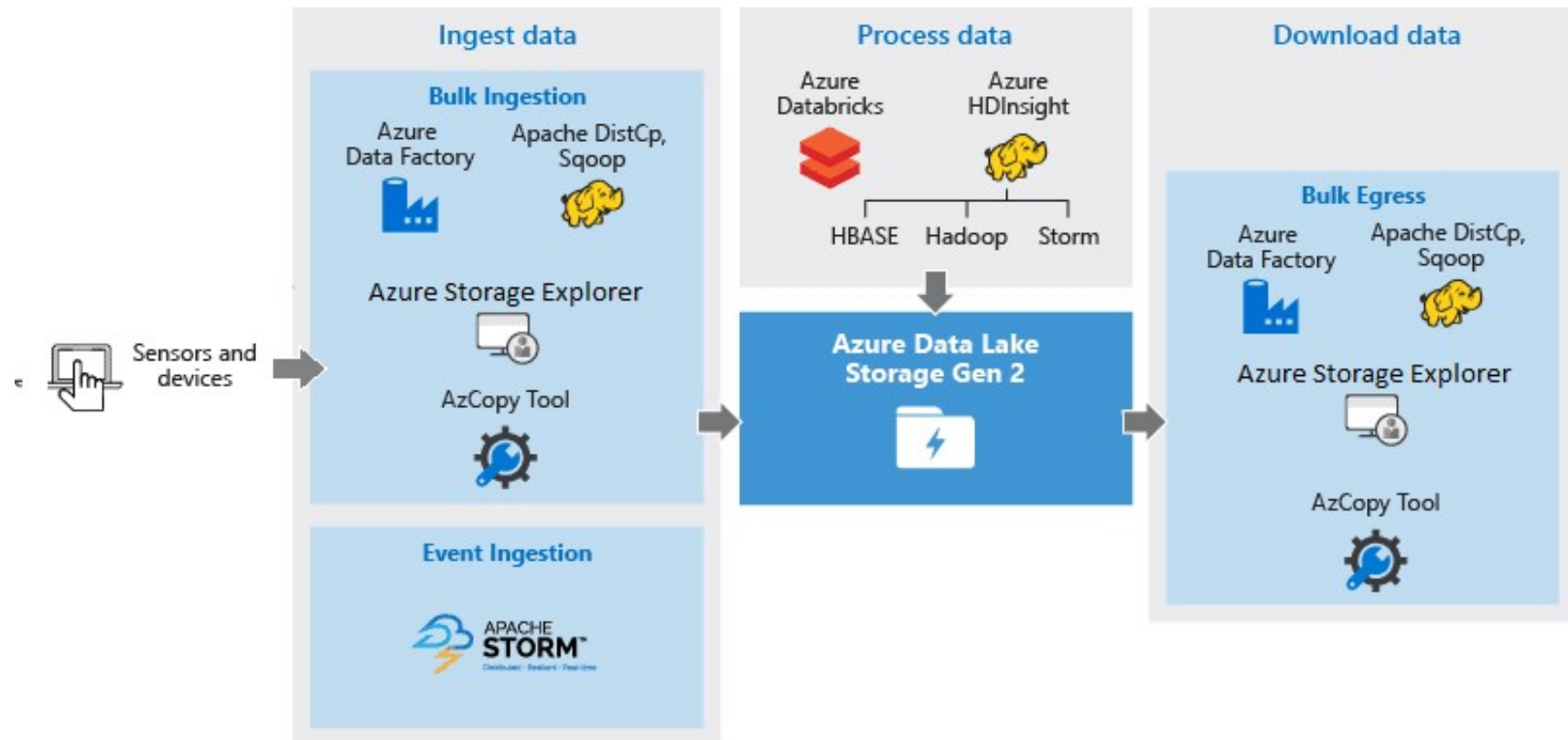
- **Azure Blob Storage** is an **object storage solution for the cloud**. It is optimized for storing massive amounts of unstructured data, such as text or binary data in a single hierarchy, also known as a flat namespace.
- Blob Storage can manage thousands of simultaneous uploads, enormous amounts of video data, constantly growing log files, and can be reached from anywhere with an internet connection via HTTP/HTTPS.
- Blobs aren't limited to common file formats. A blob could contain gigabytes of binary data streamed from a scientific instrument, an encrypted message for another application, or data in a custom format for an application. Azure takes care of the physical storage needs on your behalf.

# Azure Data Lake Storage (ADLS)

---

- ADLS is a comprehensive, scalable, and cost-effective data lake solution for high-performance big data analytics built into Azure.
- ADLS Gen1 is an enterprise-wide hyper-scale repository for big data analytic workloads. It enables you to capture data of any size, type, and ingestion speed in one single place for operational and exploratory analytics.
- ADLS Gen2 is a set of capabilities dedicated to big data analytics, built on Azure Blob. It converges the capabilities of ADLS Gen1 with Azure Blob storage.
  - **ADLS Gen2 = Azure Blob Storage + ADLS Gen1**
- ADLS Gen2 provides file system semantics, file-level security, and scale, which are inherited from ADLS Gen1. All these capabilities are built on Blob storage resulting in low cost, tiered access, high security, high availability, and durability.

# Azure Data Lake Storage (ADLS)



# Accessing ADLS from Databricks

---

There are different authentication methods to access the data in the ADLS from Databricks workspace.

- Access Data Lake using Access Keys
- Access Data Lake using Shared Access Signature (SAS) token
- Access Data Lake using Service Principal
- Access Data Lake using Cluster scoped authentication
- Access Data Lake using Azure AD credential pass-through

Reference: <https://learn.microsoft.com/en-us/azure/databricks/storage/azure-storage>

# Lab 9a – Access ADLS using access token

---

Instruction document: L09-ADLS-Gen2-Access-Patterns.txt

In this lab we do the following:

- Open the storage account and upload sample data into a container
- Disable 'soft delete' for blobs and containers
- Spin an 'All purpose' cluster in your workspace
- Get the 'Access key' of your storage account.
- Set the spark configuration for access token
- Access the ADLS container files from notebook using access keys



# Lab 9b – Access ADLS using SAS token

---

Instruction document: L09-ADLS-Gen2-Access-Patterns.txt

In this lab we do the following:

- Open the storage account and upload sample data into a container
- Disable 'soft delete' for blobs and containers
- Spin an 'All purpose' cluster in your workspace
- Generate the SAS (shared access signature) token for your container
- Set the spark configuration for SAS token
- Access the ADLS container files from notebook using SAS token

# Lab 9c – Access ADLS using service principal

---

Instruction document: L09-ADLS-Gen2-Access-Patterns.txt

In this lab we do the following:

- Open the storage account and upload sample data into a container
- Disable 'soft delete' for blobs and containers
- Spin an 'All purpose' cluster in your workspace
- Register Microsoft Entra ID Application / Service Principal
- Generate a secret/password for the Application
- Assign Role 'Storage Blob Data Contributor' to the Data Lake.
- Access the ADLS container files from notebook using Service principal

# Lab 9d – Access ADLS using cluster scoped credentials

---

Instruction document: L09-ADLS-Gen2-Access-Patterns.txt

In this lab we do the following:

- Open the storage account and upload sample data into a container
- Disable 'soft delete' for blobs and containers
- Spin an 'All purpose' cluster in your workspace
- Edit the cluster to add the access-key in Spark configurations.
- Access the ADLS container files without requiring any access configurations in the notebook.

# Databricks Secret Scope

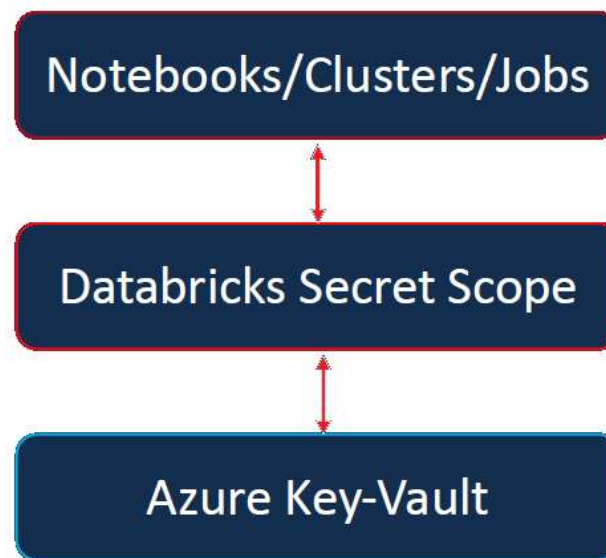
---

- A secret scope is collection of secrets identified by a name.
  - Managing secrets begins with creating a secret scope.
- Secret scopes help store the credentials securely and reference them in notebooks, clusters and jobs when required.
- Two Types:
  - Databricks backed secret scope
  - Azure key-vault backed secret scope
- A Databricks-backed secret scope is stored in an encrypted database owned and managed by Databricks.
  - This can be used only using Databricks CLI or Secret API

# Azure Key Vault

---

- Azure Key Vault is a cloud service for securely storing and accessing secrets.
  - A secret is anything that you want to tightly control access to, such as API keys, passwords, certificates, or cryptographic keys.



# Lab 10 – Secure ASDL access using Secret scope

---

Instruction document: [L10-Securing-secrets-using-Secret-Scope.txt](#)

In this lab we do the following:

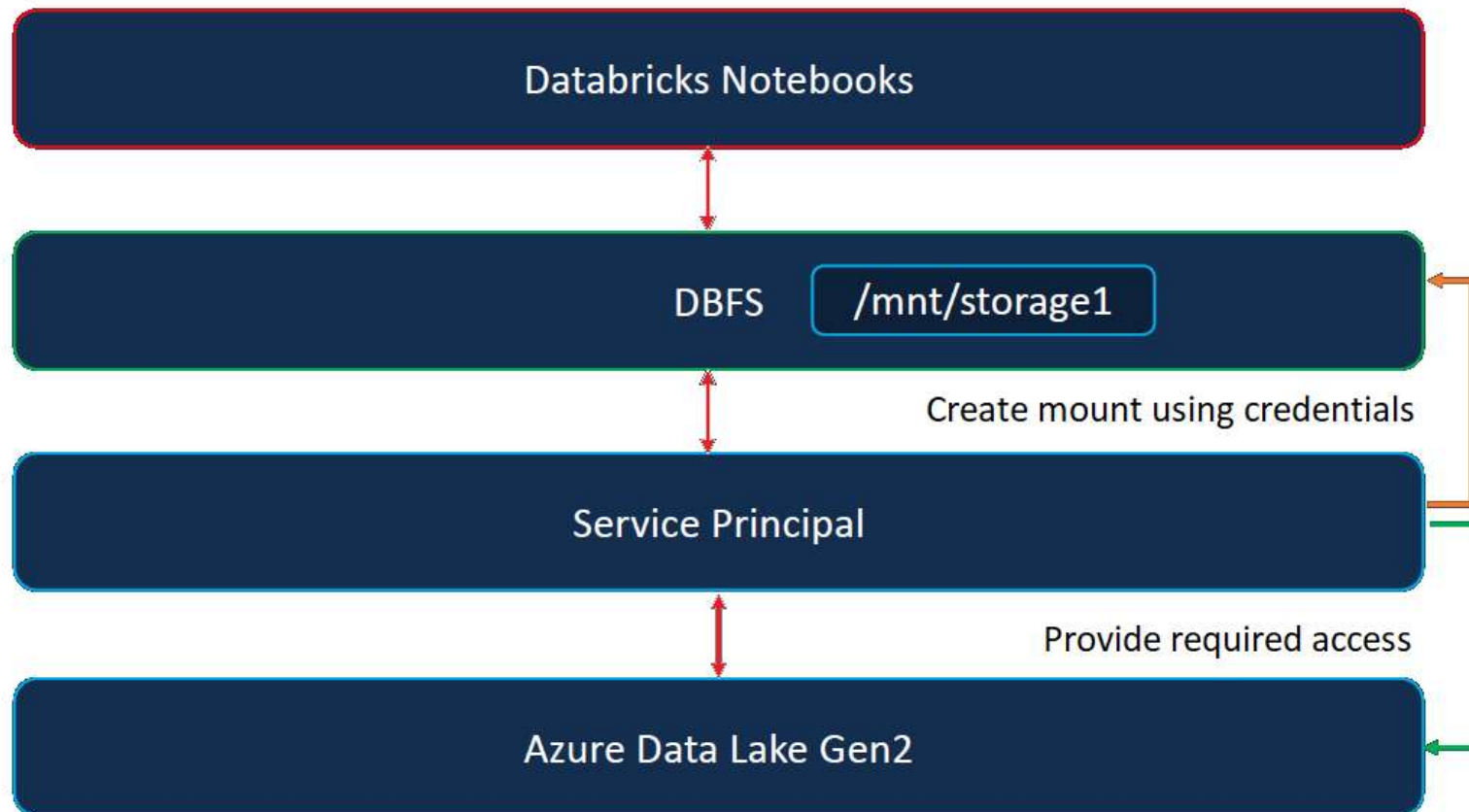
- Understand Azure Key Vault service to store secrets
- Create a secret for access-key in Azure Key vault
- Create a Databricks secret scope
- Link the secret scope to the key vault
- Understand **dbutils.secrets** utility
- Refactor the Lab-9 code to replace the hard-coded secret values using secrets.

# Databricks Mounts

---

- Databricks allows you to mount Azure Blob storage or ADLS as a mount point (/mnt) in DBFS by specifying the required credentials.
  - Once mounted, we can access the data without requiring credentials.
  - We can access files using file semantics rather than storage URLs (e.g. /mnt/storage1)
  - As the files are stored to object storage (e.g. Azure Blob), you get all the benefits of cloud object storage.
- Databricks Mount is the recommended solution for accessing Azure Storage until the introduction of Unity Catalog.

# Databricks Mounts





# Lab 11 – Mount ADLS on Azure Databricks

---

Instruction document: [L11-Mount-ADLS-using-Service-Principal.txt](#)

In this lab we do the following:

- Launch a new Databricks Workspace
- Register an Azure Active Directory Application
- Create Databricks Secret for AD Application Client Secret
- Create ADLS storage account
- Assign IAM Role on Storage Account to Azure AD Application
- Create 'ADLS File System' and upload data
- Mount ADLS Storage Account on to Azure Databricks cluster
- Run PySpark code on the mounted datasets
- Un-mount the mount point

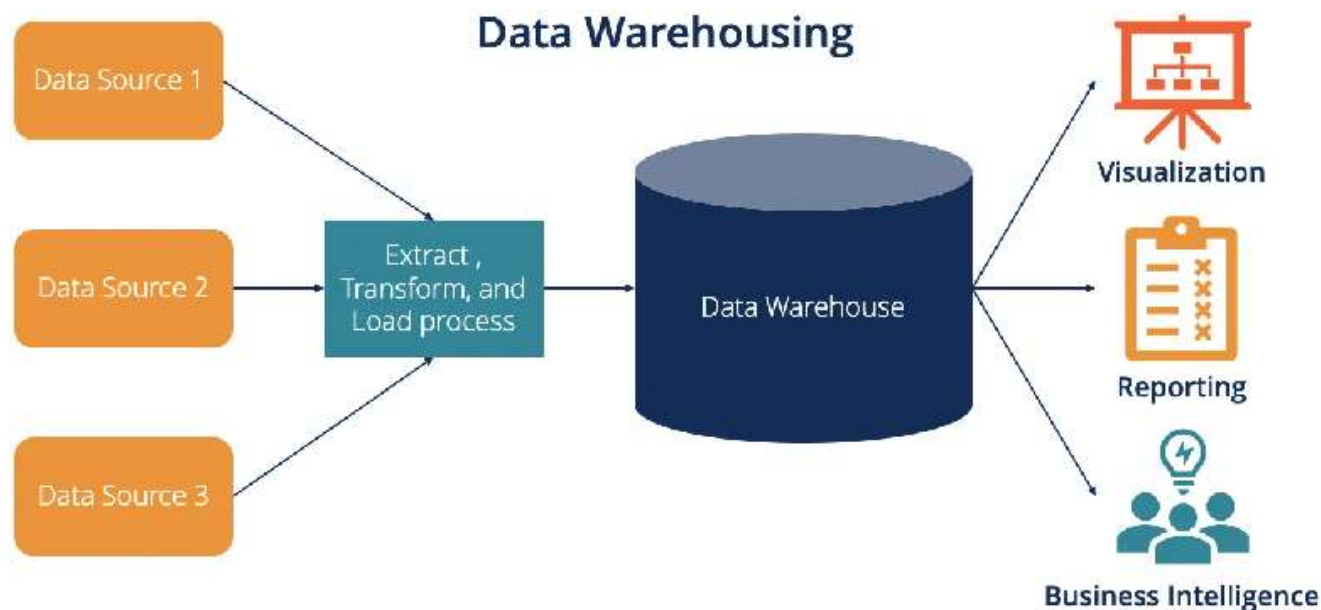


# Big Data Storage Patterns

Data Warehouse, Data Lake & Data Lakehouse

# Data Warehouse

A **data warehouse** is a **unified data repository** for storing large amounts of information from multiple sources within an organization. A data warehouse represents a single source of “data truth” in an organization and serves as a core reporting and business analytics component.



# Data Warehouse

---

## Pros:

- Improving data standardization, quality, and consistency
- Delivering enhanced business intelligence
- Increasing the power and speed of data analytics and business intelligence workloads
- Improving the overall decision-making process

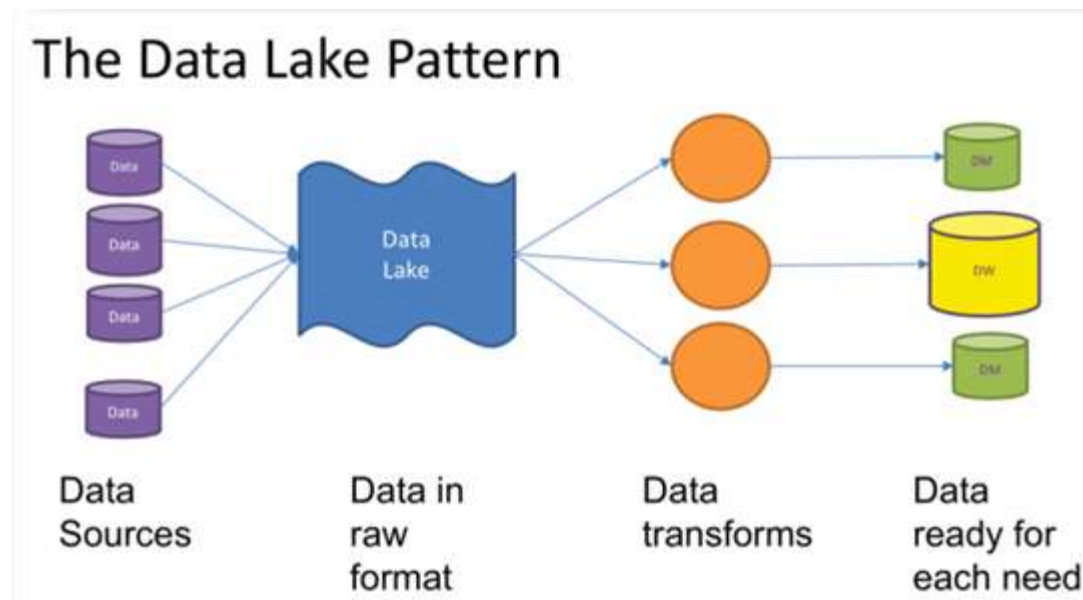
## Cons:

- Lack of data flexibility
- High implementation and maintenance costs

# Data Lake

A data lake is a centralized, flexible storage repository that stores large amounts of structured and unstructured data in its raw, original, and unformatted form.

- Data lake stores data using a flat architecture and object storage in its raw form.
- Data lakes are flexible, durable, and cost-effective and enable organizations to gain advanced insight from unstructured data, unlike data warehouses that struggle with data in this format.



# Data Lake

---

## **Pros:**

- Data consolidation
- Data flexibility
- Cost savings
- Support for a wide variety of data science and ML use cases

## **Cons:**

- Poor performance for BI and data analytics use cases
- Lack of data reliability and security

# Data Lakehouse

---

- A data lakehouse is a new, big-data storage architecture that combines the best features of both data warehouses and data lakes.
- A data lakehouse enables a single repository for all your data (structured, semi-structured, and unstructured) while enabling best-in-class machine learning, business intelligence, and streaming capabilities.
- Data lakehouses usually start as data lakes containing all data types; the data is then converted to Delta Lake format. Delta lakes enable ACID transactional processes from traditional data warehouses on data lakes.

# Data Lakehouse

---





# Data Warehouse, Data Lake & Lakehouse

	Data Warehouse	Data Lake	Data Lakehouse
Storage Data Type	Works well with structured data	Works well with semi-structured and unstructured data	Can handle structured, semi-structured, and unstructured data
Purpose	Optimal for data analytics and business intelligence (BI) use-cases	Suitable for machine learning (ML) and artificial intelligence (AI) workloads	Suitable for both data analytics and machine learning workloads
Cost	Storage is costly and time-consuming	Storage is cost-effective, fast, and flexible	Storage is cost-effective, fast, and flexible
ACID Compliance	Records data in an ACID-compliant manner to ensure the highest levels of integrity	Non-ACID compliance: updates and deletes are complex operations	ACID-compliant to ensure consistency as multiple parties concurrently read or write data

# Which one is for me?

---

- A data warehouse is a good choice for companies seeking a mature, structured data solution that focuses on business intelligence and data analytics use cases.
- However, data lakes are suitable for organizations seeking a flexible, low-cost, big-data solution to drive machine learning and data science workloads on unstructured data.
- Suppose the data warehouse and data lake approaches aren't meeting your company's data demands, or you're looking for ways to implement both advanced analytics and machine learning workloads on your data. In that case, a data lakehouse is a reasonable choice.



Delta Lake

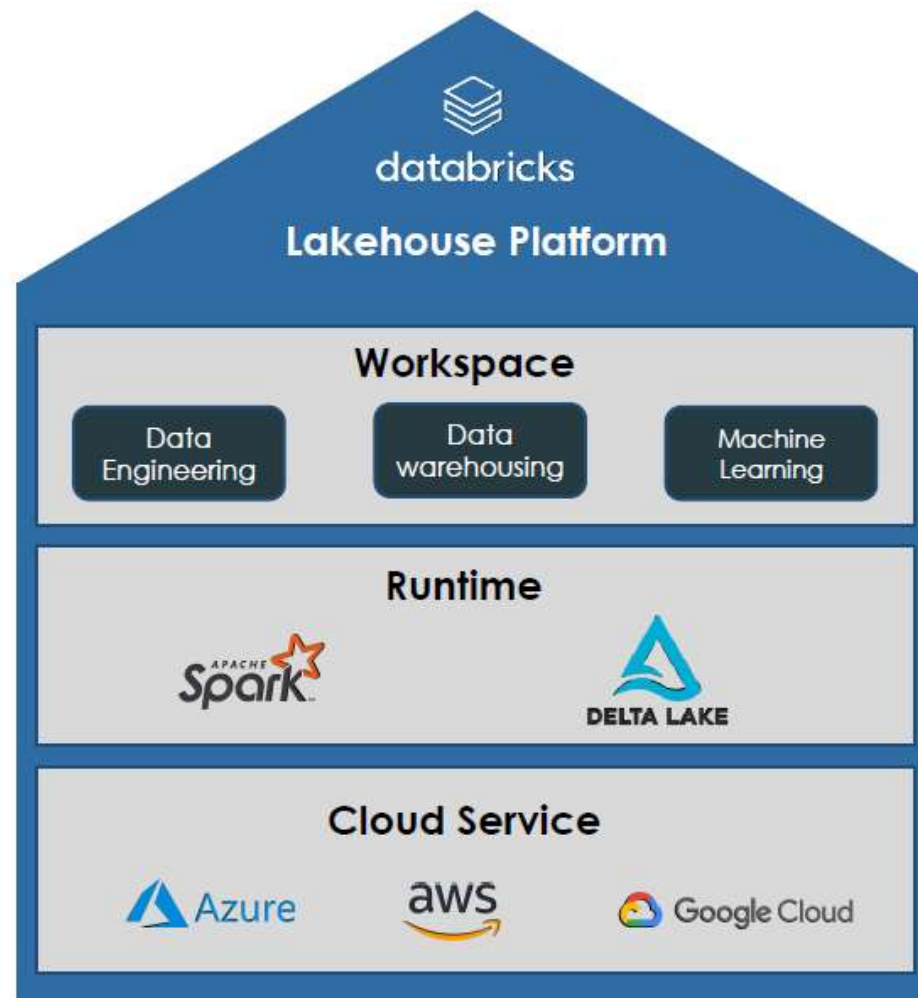
# Delta Lake

---

- Delta Lake (*delta.io*) is an open-source storage framework that enables building a Lakehouse architecture with compute engines including Spark, PrestoDB, Flink, Trino, and Hive and APIs for Scala, Java, Rust, and Python.
- Delta Lake is a file-based metadata, caching and indexing layer on top of a data lake storage that provides an abstraction to serve ACID transactions (CRUD operations – INSERT, UPDATE, DELETE and MERGE/UPSERT).
- Delta Lake provides ACID transactions, scalable meta data handling, a unified process model that spans batch and streaming, full audit history, and support for SQL DML statements.
- It can run on existing data lakes and is fully compatible with several processing engines, including Apache Spark.

# Delta Lake

---



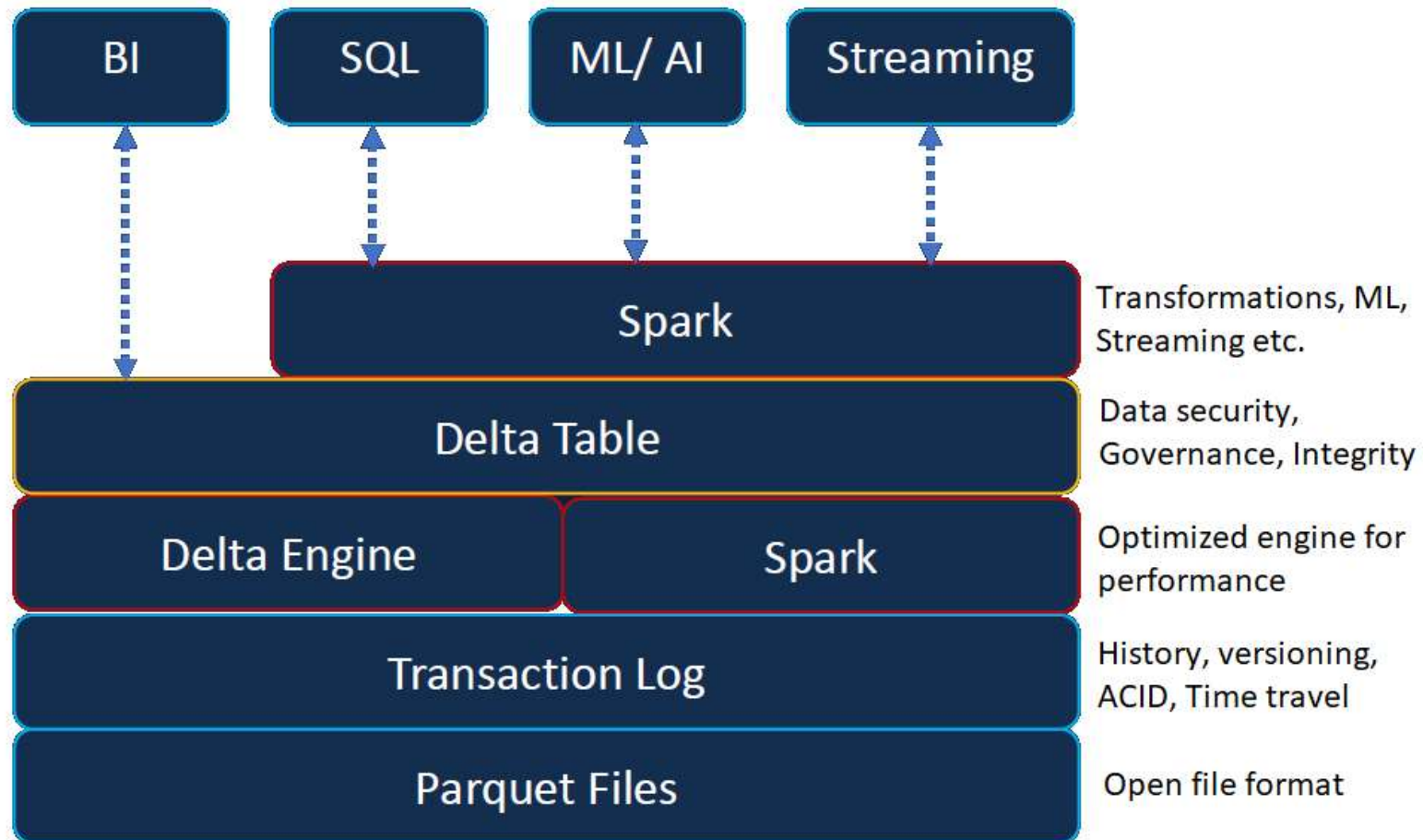
# Delta Lake Features

---

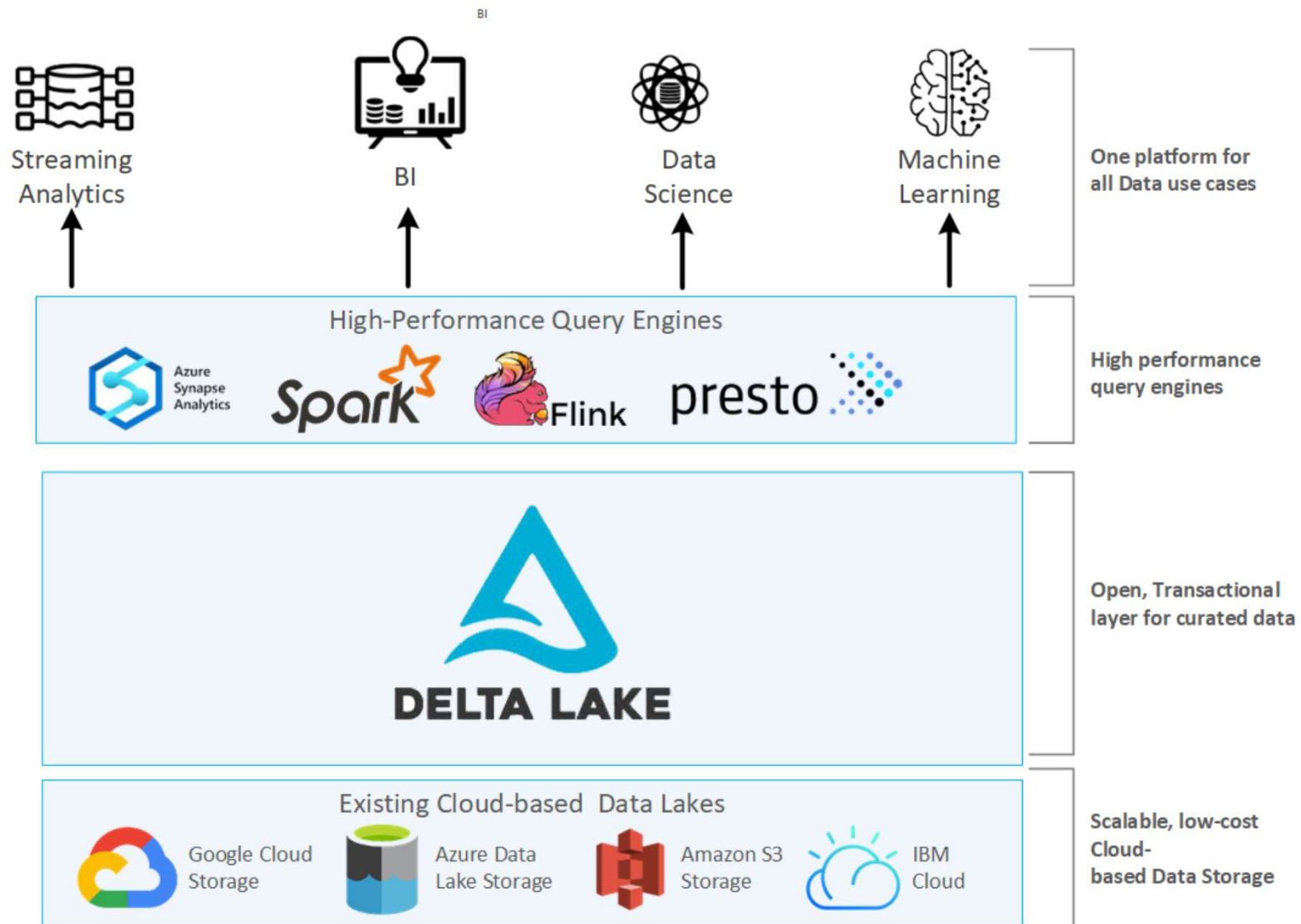
Features of Delta lake are:

- Transactional ACID guarantees
- Full DML support
- Audit History
- Unification of batch and streaming into one processing model
- Schema enforcement and evolution
- Rich metadata support and scaling

# How Delta Lake works?



# Delta Lake Architecture





# Databricks Delta Tables

---

- A Delta table is a managed table that uses the Databricks Delta Optimistic Concurrency Control feature to provide full ACID transactions.
- Databricks Delta tables support all common operations, such as:
  - CRUD (create, read, update, delete)
  - Upsert (update or insert)
  - Merge (upsert a set of records)
  - Delete from the table where ... (delete based on predicate)
- In addition, Databricks Delta tables support the following:
  - Time travel – rollback data as of any point in time (based on timestamp or version)
  - Uncommitted reads - see changes that have not yet been committed
  - Optimistic concurrency control - prevents dirty reads and writes

# Lab 12 – CRUD operations using Delta Tables

---

Instruction document: `L12-Perform-CRUD-using-Delta-Tables.txt`

In this lab we do the following:

- Create source DataFrames from well-formed JSON strings
- Save the DataFrame into a directory in delta format
- Create DataFrame from the delta files
- Create a DeltaTable reading from delta file path
- Perform an UPDATE using the DeltaTable
- Perform a DELETE using the DeltaTable
- Perform MERGE operation using the DeltaTable

# Lab 13 – Restore a Delta Table to an old version

---

Instruction document: L13-Restore-Delta-Tables.txt

In this lab we do the following:

- Examine the history of delta table
- Perform 'Restore to version' on delta table
- Perform 'Restore to timestamp' on delta table

- You can restore a Delta table to its earlier state by using the RESTORE command.
- A Delta table internally maintains historic versions of the table that enable it to be restored to an earlier state. A version corresponding to the earlier state or a timestamp of when the earlier state was created are supported as options by the RESTORE command.

## Reference:

<https://docs.delta.io/latest/delta-utility.html#restore-a-delta-table-to-an-earlier-state>

# Lab 14 – Vacuum a Delta Table

---

Instruction document: L14-Vacuum-Delta-Tables.txt

In this lab we do the following:

- Examine the history of Delta table
- Perform 'vacuum' on delta table

- You can remove files no longer referenced by a Delta table and are older than the retention threshold by running the ***vacuum*** command on the table.
- ***vacuum*** is not triggered automatically.
- The default retention threshold for the files is 7 days.

## Reference:

<https://docs.delta.io/latest/delta-utility.html#remove-files-no-longer-referenced-by-a-delta-table>

# Lab 15 – Delta Table Compaction

---

Instruction document: L15-Compacting-Delta-Tables.txt

In this lab we do the following:

- Examine the history of Delta table
- Perform compaction on delta table using repartition with overwrite mode.

- **Compaction is a process of merging too many small files into fewer large files.**

## **Reference:**

<https://docs.delta.io/latest/best-practices.html#-delta-compact-files>

**Thank You**