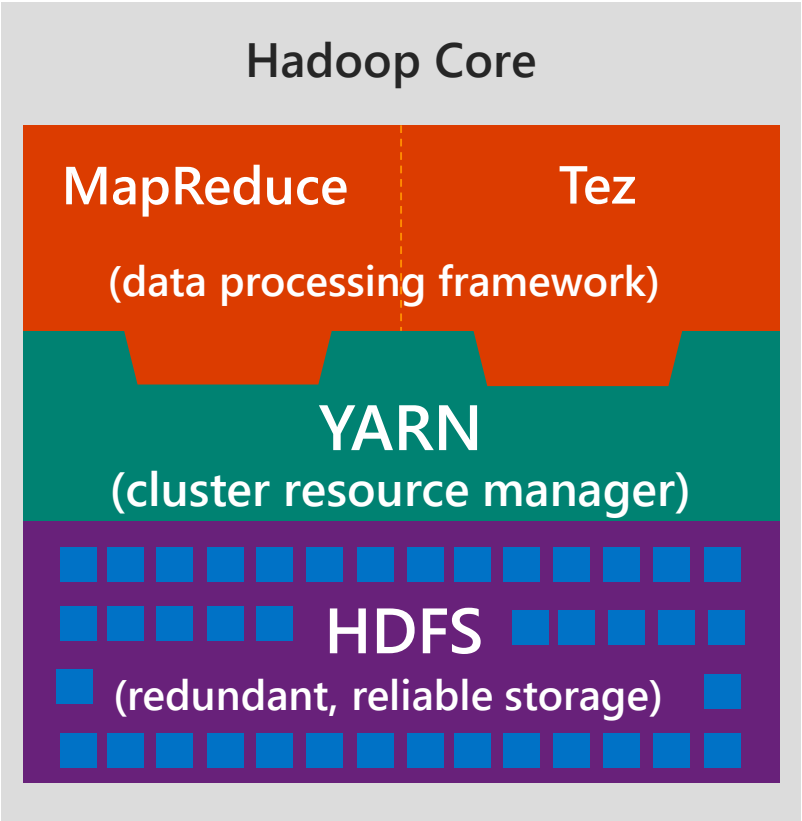
Hadoop – What is it?

A highly reliable, distributed and parallel programming framework for analyzing big data

- An Java-based, open sourced, Apache project
- Capable of running on variety of hardware platforms, including clusters of commodity hardware
  - Is tolerant to failures of nodes, software components, network
  - Scales with the cluster
- The Hadoop core consists of:
  - A scalable, reliable file system (HDFS)
  - A framework that enables development of programs based on MapReduce (MR) or Directed Acyclic Graph (DAG) model
  - YARN, a distributed resource manager that allocates and controls access to the resource of the cluster manager
- In addition to the core Hadoop has a rich ecosystem that supports SQL/NoSQL, Streaming, Real-time and Interactive applications.



HDFS – What is it?

A scalable, reliable and highly distributed file system to store structured and unstructured data

- A open-source Apache Project
- Optimized for high throughput of data access rather than low latency of data access.
- Data is organized into files and directories
- Files are divided into uniform size block and divided across cluster nodes
- Data blocks are re-balanced (using different models)
- Block placement is exposed to application so that computation can be place close to the data (enables the “Move the Computation to the Data” philosophy”)
- Blocks are replicated (default 3) to handle failures and for performance
- Checksums for corruption detection and recovery
- POSIX-style permissions and access control model

MapReduce Component: Job Tracker

The JobTracker service farms out MapReduce tasks to specific nodes in the cluster (ideally the nodes that have the data, or at least are in the same rack).

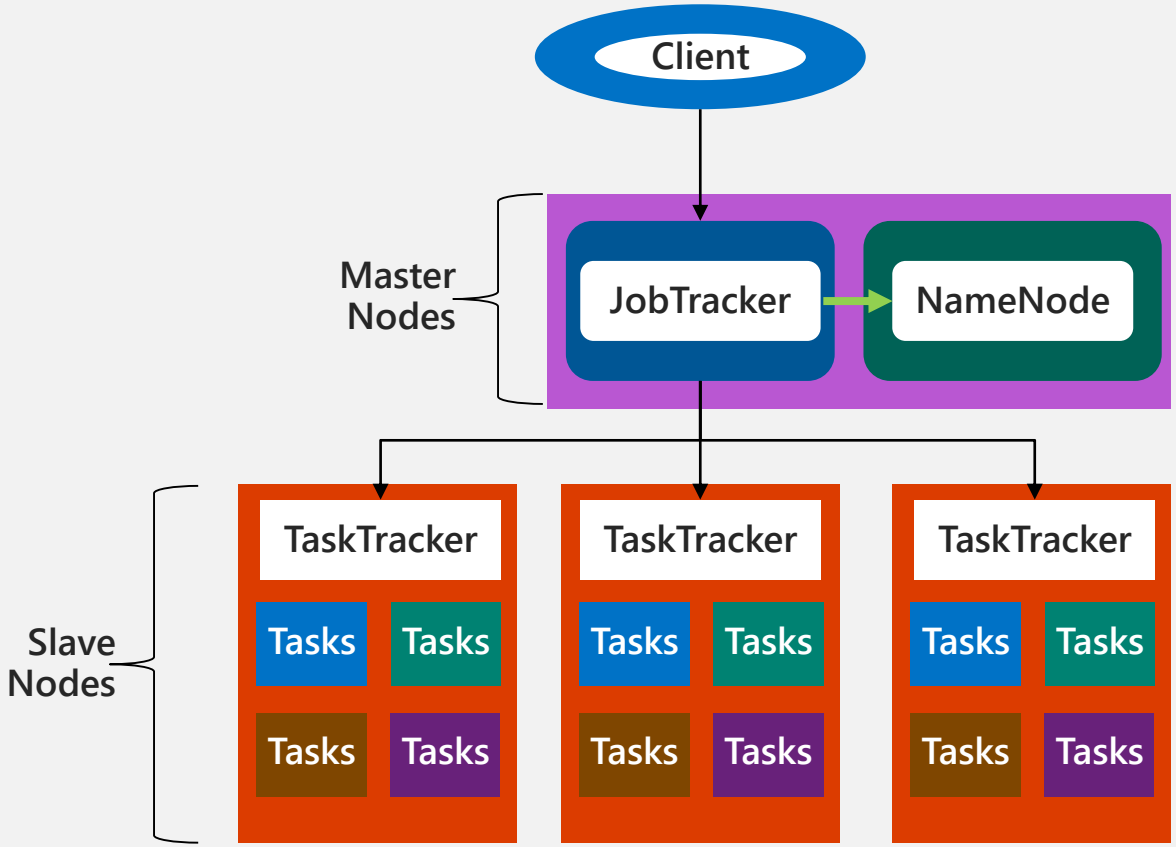
1. Client applications submit jobs to the Job tracker.
2. The JobTracker talks to the NameNode to determine the location of the data
3. The JobTracker locates TaskTracker nodes with available slots at or near the data
4. The JobTracker submits the work to the chosen TaskTracker nodes.
5. When the work is completed, the JobTracker updates its status.

TaskTracker nodes are monitored.

- If they do not submit heartbeat signals often enough, they are deemed to have failed and the work is scheduled on a different TaskTracker.

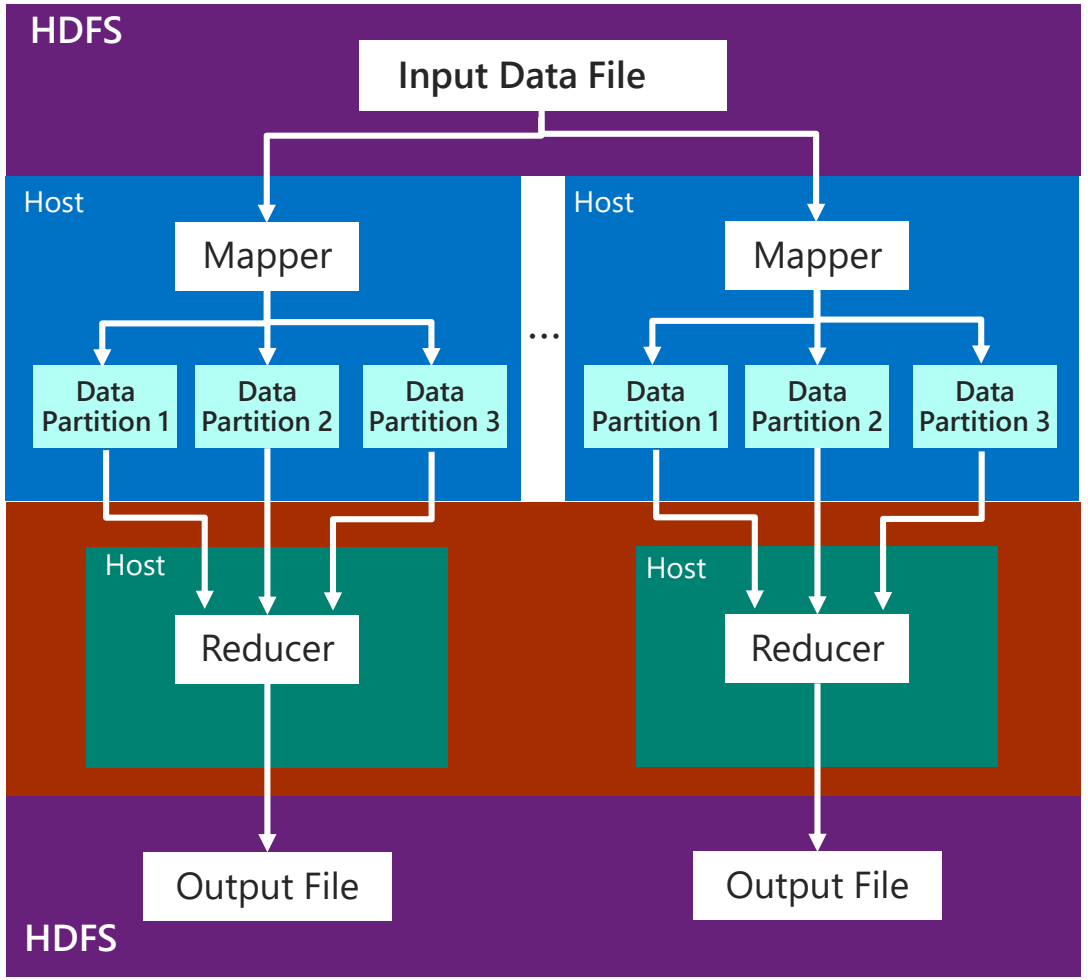
A TaskTracker will notify the JobTracker when a task fails

- The Job Tracker may resubmit the job elsewhere, mark that specific record as something to avoid and even blacklist the TaskTracker as unreliable.

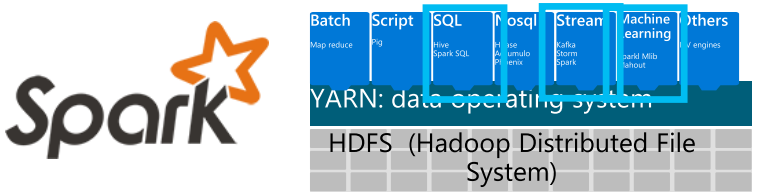


MapReduce: How it works

- Hadoop divides the input file into splits and assign every split to different mapper.
- Locally Hadoop reads the split line-by-line and call map() for very line, passing it as key/value parameters
- The mapper emits other intermediate key/value pairs
- All the intermediate values for a given intermediate key are combined together into a list.
- The list is given to one or more Reducers.
  - All values associated with a particular intermediate key are guaranteed to go to the same Reducer
  - The intermediate keys, and their value lists, are passed to the Reducer in sorted key order – This step is known as the 'shuffle and sort'
  - Hadoop calls reduce() on every line of the input
  - The Reducer outputs zero or more final key/value pairs. These are written to HDFS



Spark



Massive data processing framework built on in-memory

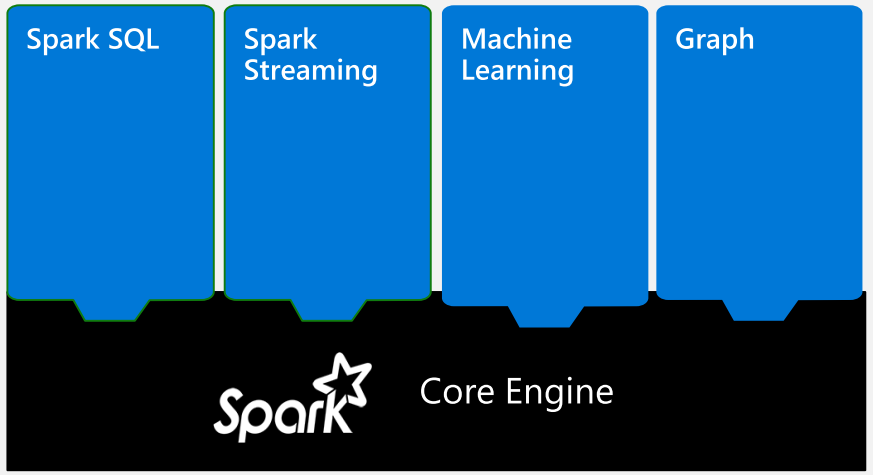
Single execution model for multiple tasks

Processing up to 100x faster performance

Developer friendly (Java, Python, Scala)

BI tool of choice (Power BI, Tabelau, Qlik, SAP)

Notebook experience (Jupyter & Zeppelin)

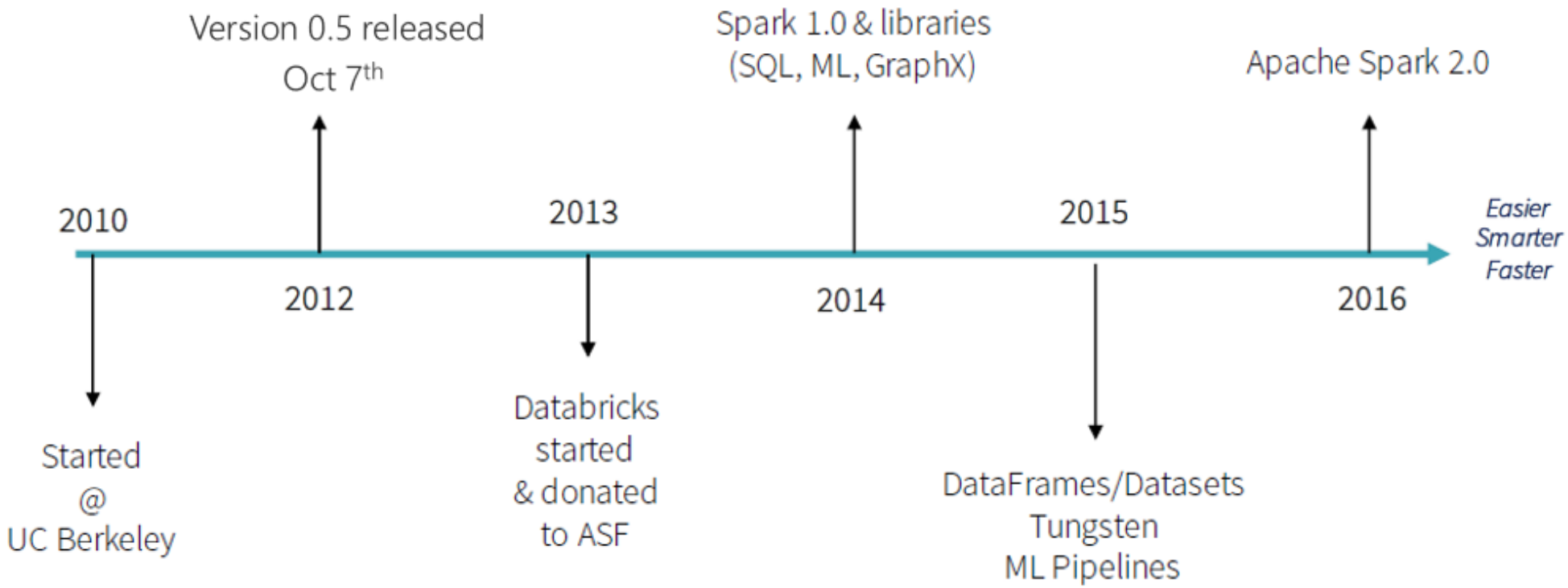



Microsoft
Azure Databricks
A Technical Overview

TOC

- Spark Overview
- Azure Databricks
  - Overview of the offering
  - Core Concepts
- Secure Collaboration
  - Azure Active Directory Integration
  - Fine Grained Permission and Access Control
- Core Artifacts
  - Clusters, Jobs, Notebooks, Libraries, Workspaces, Folders
- Spark Application Workloads
  - Data Analytics, Stream Analytics, Machine Learning, Graph Processing
- Performance
- CLI and Rest APIs

SPARK: A BRIEF HISTORY

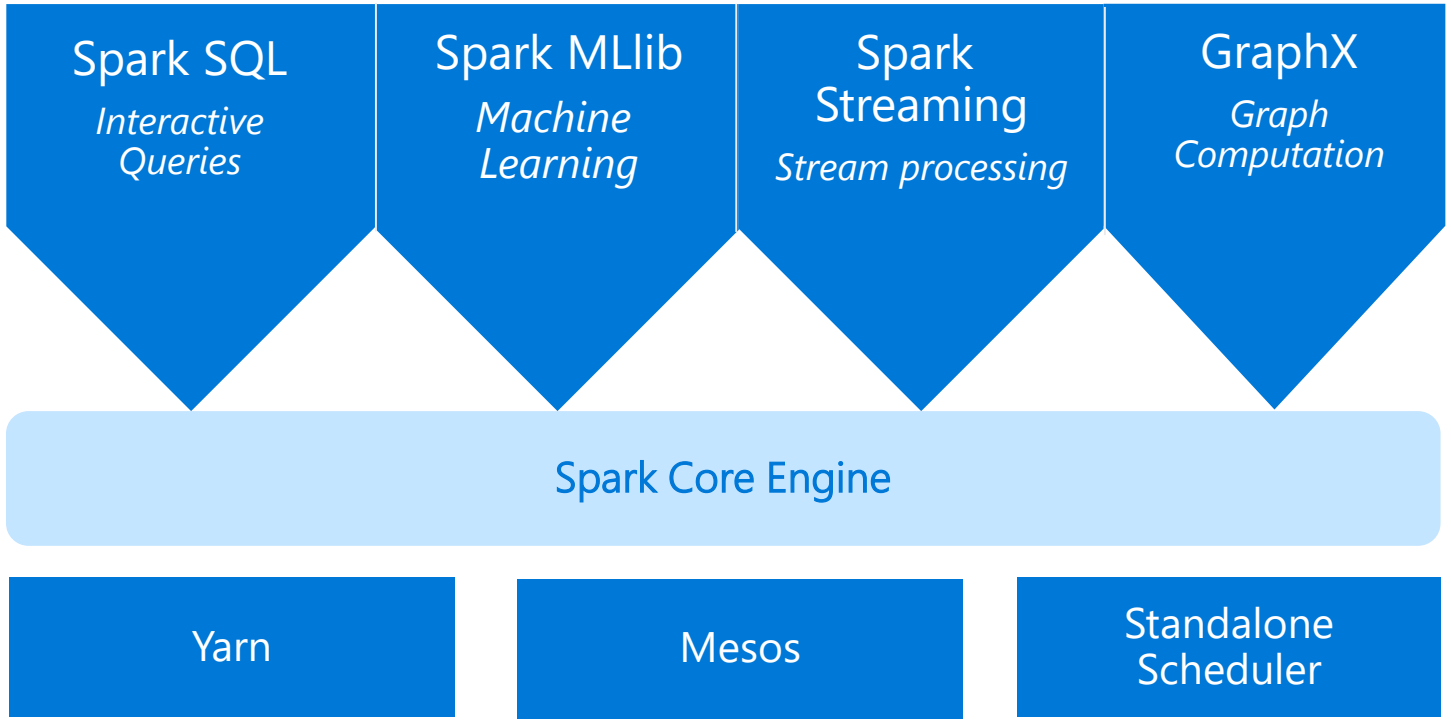


APACHE SPARK

An unified, open source, parallel, data processing framework for Big Data Analytics

Spark Unifies:

- Batch Processing
- Interactive SQL
- Real-time processing
- Machine Learning
- Deep Learning
- Graph Processing



SPARK - BENEFITS

Performance

Using in-memory computing, Spark is considerably faster than Hadoop (100x in some tests).

Can be used for batch and real-time data processing.

Developer Productivity

Easy-to-use APIs for processing large datasets.

Includes 100+ operators for transforming.

Unified Engine

Integrated framework includes higher-level libraries for interactive SQL queries, Stream Analytics, ML and graph processing.

A single application can combine all types of processing

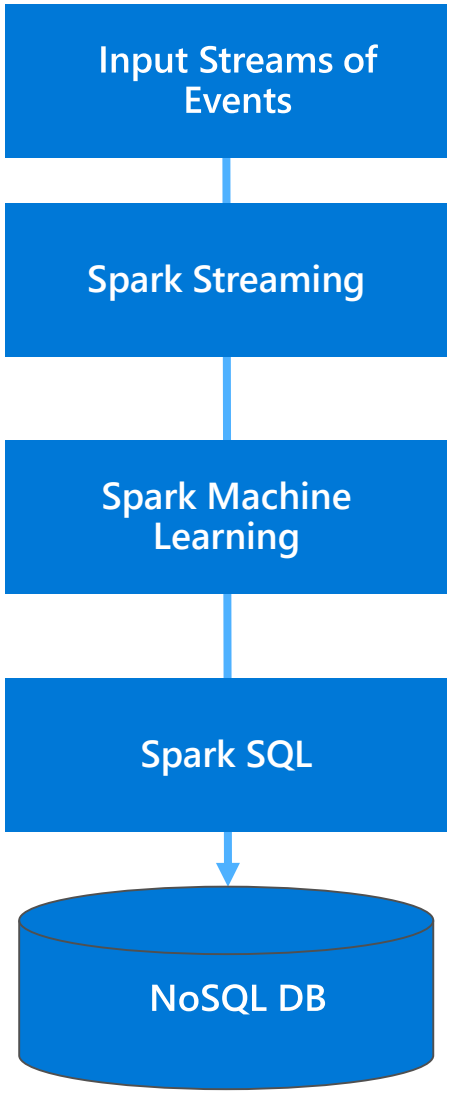
Ecosystem

Spark has built-in support for many data sources, rich ecosystem of ISV applications and a large dev community.

Available on multiple public clouds (AWS, Google and Azure) and multiple on-premises distributors

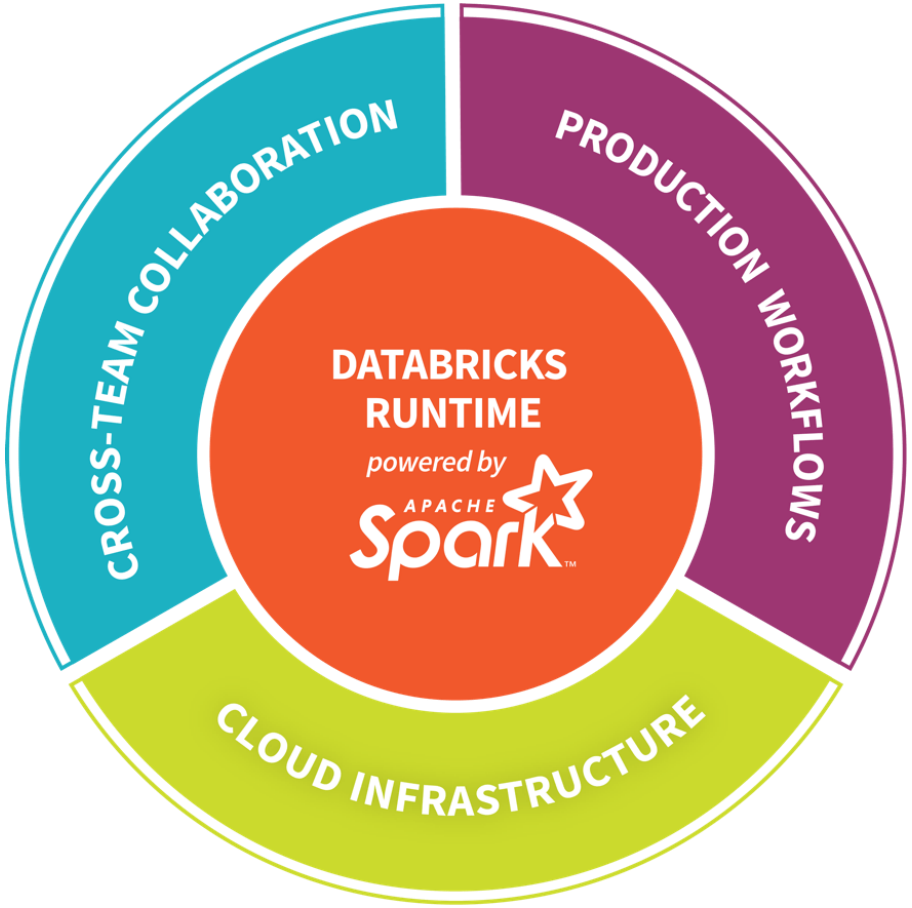
ADVANTAGES OF A UNIFIED PLATFORM

- Improves developer productivity—a single consistent set of APIs
- All different systems in Spark share the same abstraction – RDDs (Resilient Distributed Datasets)
- Developers can mix and match different kind of processing in the same application. This is a common requirement for many big data pipelines.
- Performance improves because unnecessary movement of data across engines is eliminated. In many pipelines, data exchange between engines is the dominant cost



DATABRICKS - COMPANY OVERVIEW

- Founded in late 2013
- By the creators of Apache Spark, original team from UC Berkeley AMPLab
- Largest code contributor code to Apache Spark
- Level 2/3 support partnership with
  - Hortonworks
  - MapR
  - DataStax
- Provides certifications such as Databricks Certified Application, Databricks Certified Distribution and Databricks Certified Developer
- Main Product: The Unified Analytics Platform
- In Oct 2017, introduced Databricks Delta (currently in private preview).



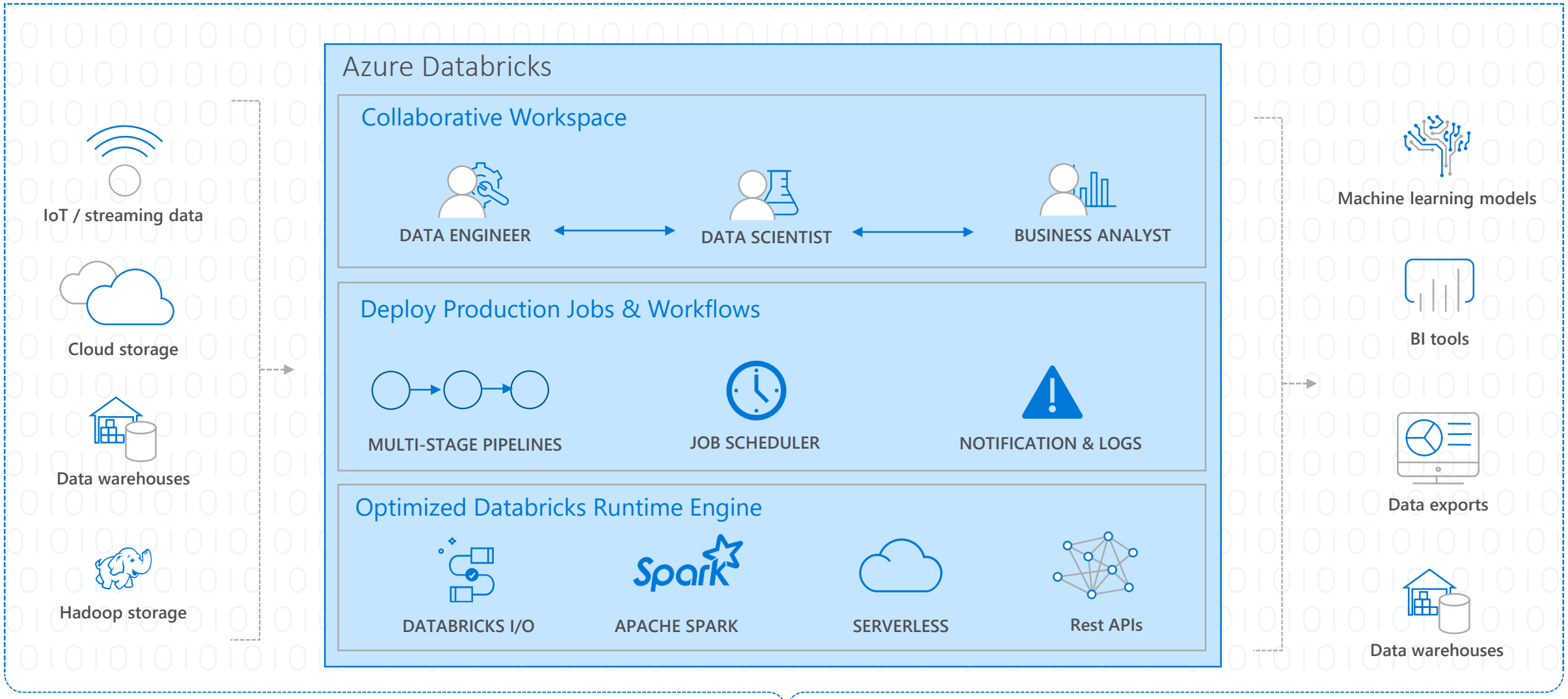
Azure Databricks

Databricks Spark as a managed service on Azure

AZURE DATABRICKS

- Azure Databricks is a **first party** service on Azure.
  - Unlike with other clouds, it is not an Azure Marketplace or a 3rd party hosted service.
- Azure Databricks is integrated seamlessly with Azure services:
  - Azure Portal: Service an be launched directly from Azure Portal
  - Azure Storage Services: Directly access data in Azure Blob Storage and Azure Data Lake Store
  - Azure Active Directory: For user authentication, eliminating the need to maintain two separate sets of uses in Databricks and Azure.
  - Azure SQL DW and Azure Cosmos DB: Enables you to combine structured and unstructured data for analytics
  - Apache Kafka for HDInsight: Enables you to use Kafka as a streaming data source or sink
  - Azure Billing: You get a single bill from Azure
  - Azure Power BI: For rich data visualization
- Eliminates need to create a separate account with Databricks.

AZURE DATABRICKS

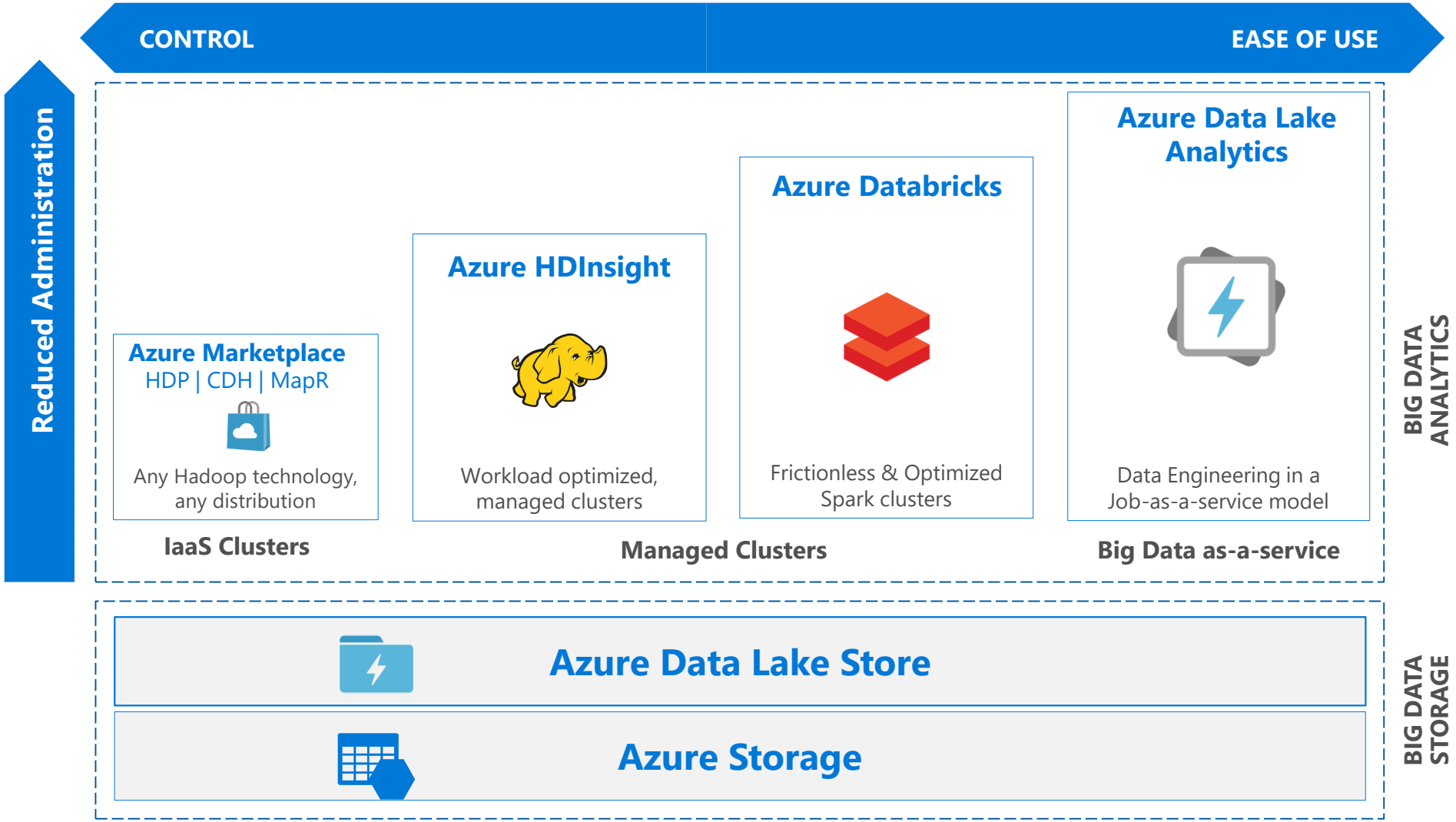


Enhance Productivity

Build on secure & trusted cloud

Scale without limits

KNOWING THE VARIOUS BIG DATA SOLUTIONS



LOOKING ACROSS THE OFFERINGS

Azure HDInsight

What It Is

- Hortonworks distribution as a first party service on Azure
- Big Data engines support – Hadoop Projects, Hive on Tez, Hive LLAP, Spark, HBase, Storm, Kafka, R Server
- Best-in-class developer tooling and Monitoring capabilities
- **Enterprise Features**
  - VNET support (join existing VNETs)
  - Ranger support (Kerberos based Security)
  - Log Analytics via OMS
  - Orchestration via Azure Data Factory
  - Available in most Azure Regions (27) including Gov Cloud and Federal Clouds

Guidance

- Customer needs Hadoop technologies other than, or in addition to Spark
- Customer prefers Hortonworks Spark distribution to stay closer to OSS codebase and/or 'Lift and Shift' from on-premises deployments
- Customer has specific project requirements that are only available on HDInsight

Azure Databricks

What It Is

- Databricks' Spark service as a first party service on Azure
- Single engine for Batch, Streaming, ML and Graph
- Best-in-class notebooks experience for optimal productivity and collaboration
- **Enterprise Features**
- Native Integration with Azure for Security via AAD (OAuth)
- Optimized engine for better performance and scalability
- RBAC for Notebooks and APIs
- Auto-scaling and cluster termination capabilities
- Native integration with SQL DW and other Azure services
- Serverless pools for easier management of resources

Guidance

- Customer needs the best option for Spark on Azure
- Customer teams are comfortable with notebooks and Spark
- Customers need Auto-scaling and
- Customer needs to build integrated and performant data pipelines
- Customer is comfortable with limited regional availability (3 in preview, 8 by GA)

Azure ML

What It Is

- Azure first party service for Machine Learning
- Leverage existing ML libraries or extend with Python and R
- Targets emerging data scientists with drag & drop offering
- Targets professional data scientists with
  - Experimentation service
  - Model management service
  - Works with customers IDE of choice

Guidance

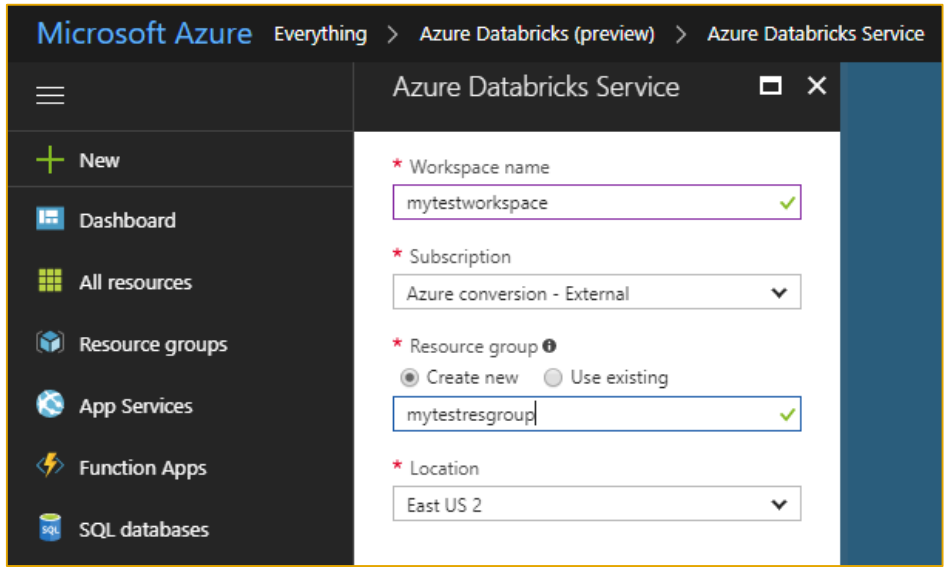
- Azure Machine Learning Studio is a GUI based ML tool for emerging Data Scientists to experiment and operationalize with least friction
- Azure Machine Learning Workbench is not a compute engine & uses external engines for Compute, including SQL Server and Spark
- AML deploys models to HDI Spark currently
- AML should be able to deploy Azure Databricks in the near future

Azure Databricks

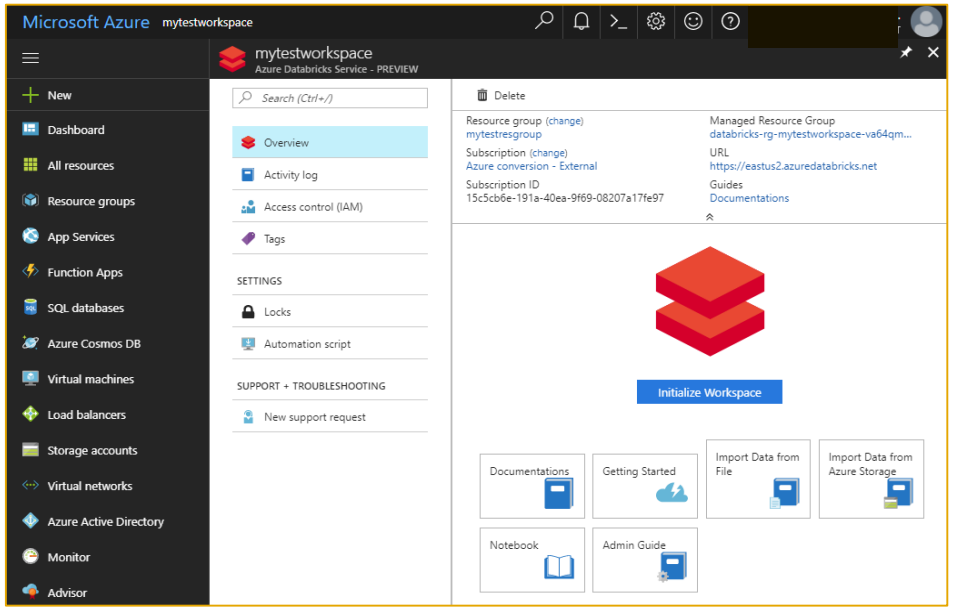
Core Concepts

PROVISIONING AZURE DATABRICKS WORKSPACE

- Azure Databricks is provisioned directly from the Azure Portal like any other Azure service
  - In contrast, with other clouds, it has to be provisioned through the Databricks portal.
  - With Azure Databricks, the Azure Portal offers a unified portal to provision and administer Azure Databricks as well as other Azure services.
- Any Azure user with the appropriate subscription and authorization can provision Azure Databricks service*.
  - There is no need for a separate Databricks account



Provisioning the Azure Databricks Service

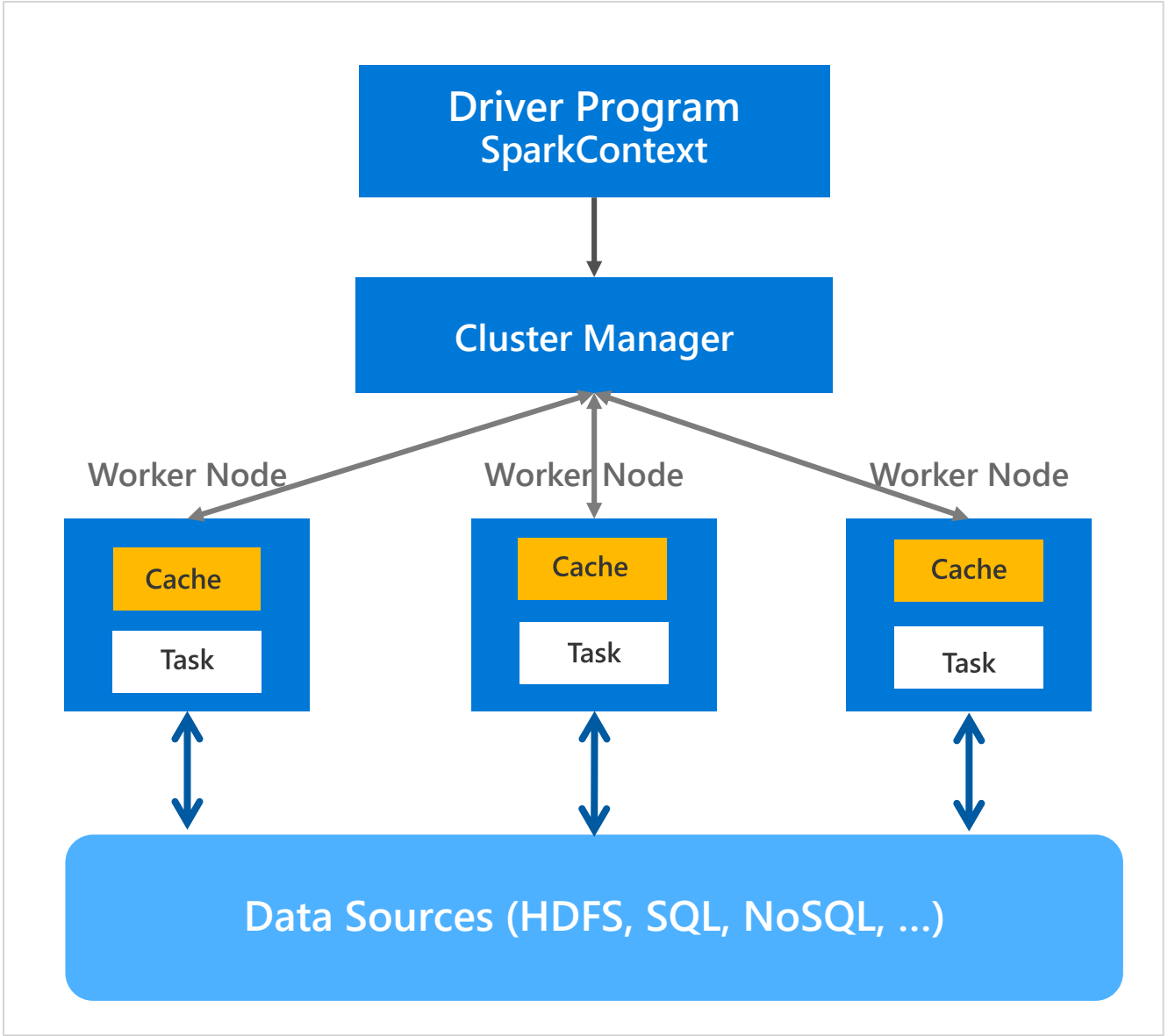


After provisioning the is complete

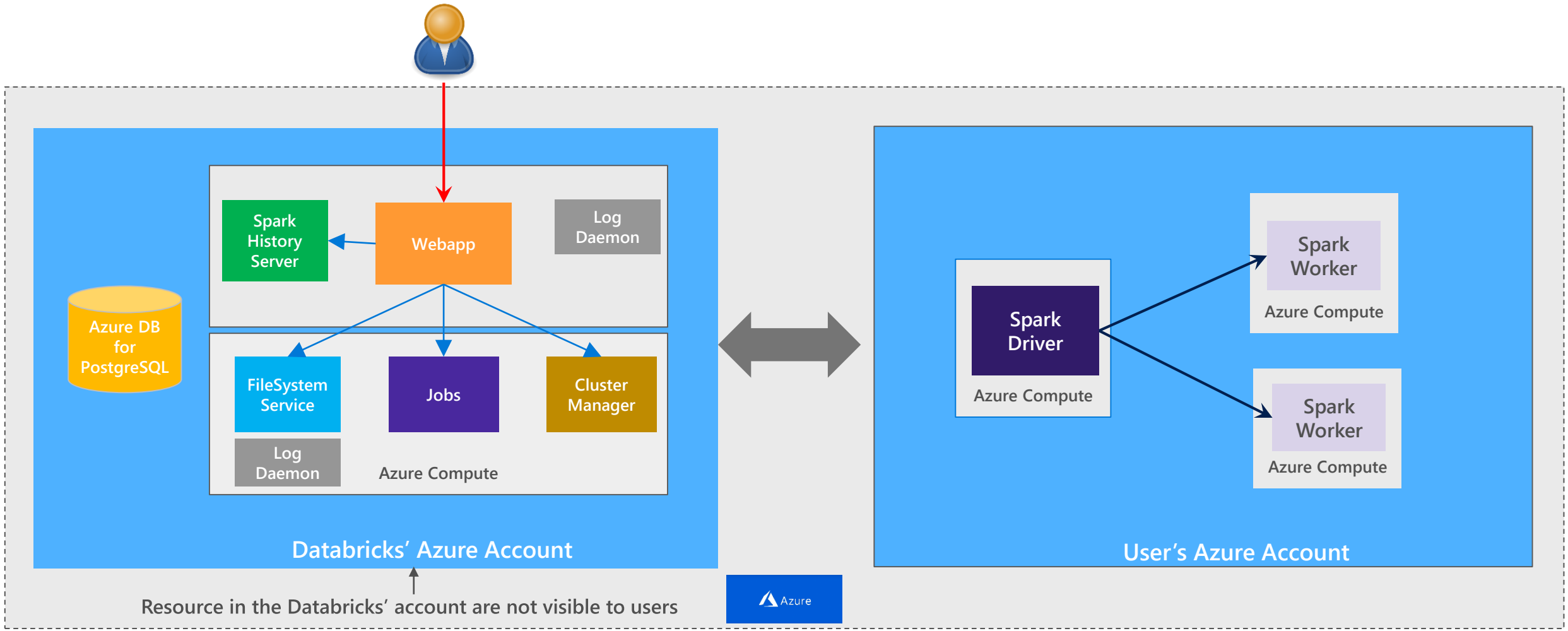
** During the current preview phase, the subscription has to be whitelisted.*

GENERAL SPARK CLUSTER ARCHITECTURE

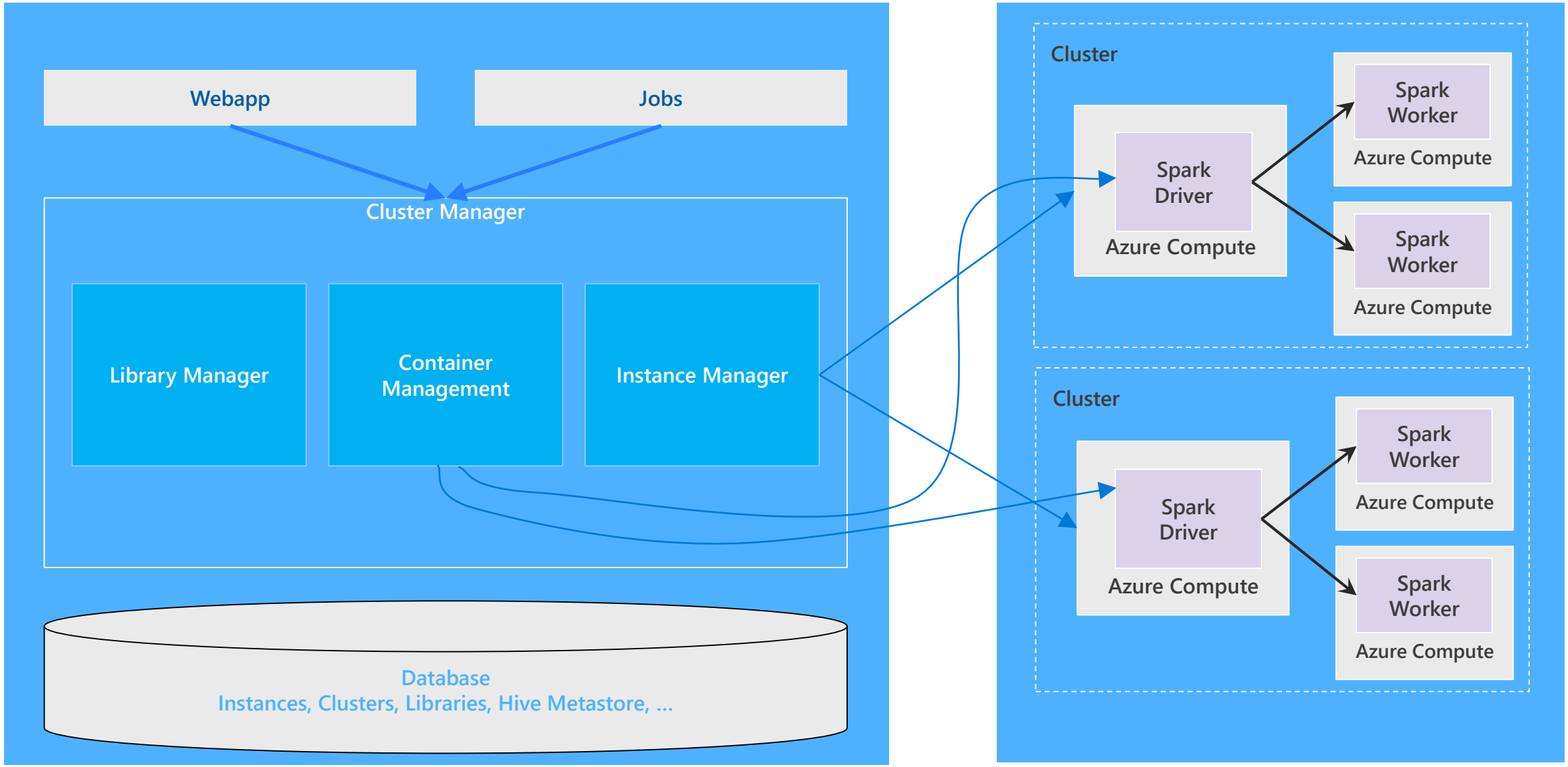
- 'Driver' runs the user's 'main' function and executes the various parallel operations on the worker nodes.
- The results of the operations are collected by the driver
- The worker nodes read and write data from/to Data Sources including HDFS.
- Worker node also cache transformed data in memory as RDDs (Resilient Data Sets).
- Worker nodes and the Driver Node execute as VMs in public clouds (AWS, Google and Azure).



AZURE DATABRICKS CLUSTER ARCHITECTURE



CLUSTER MANAGER ARCHITECTURE



Secure Collaboration

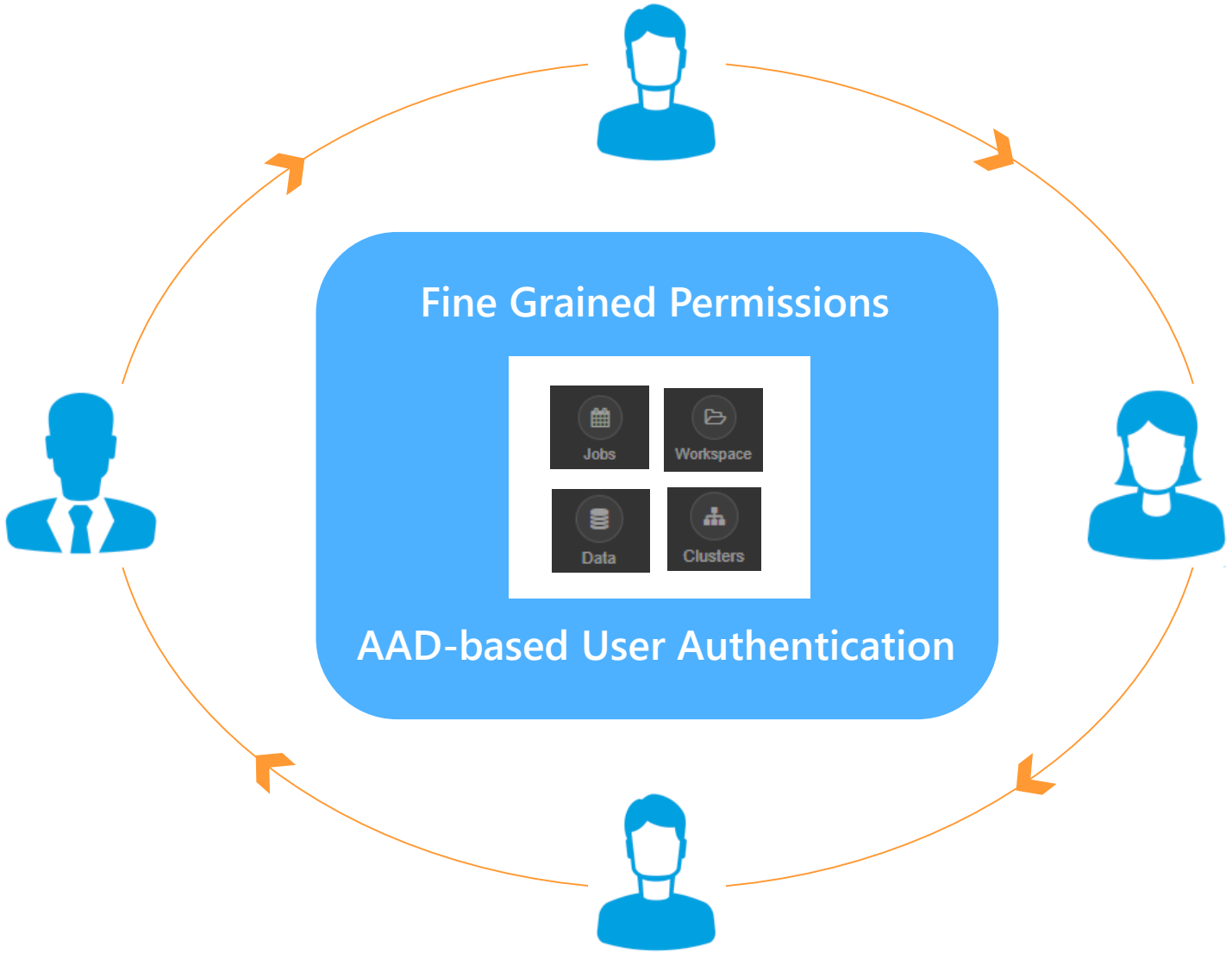
SECURE COLLABORATION

Azure Databricks enables *secure* collaboration between colleagues

- With Azure Databricks colleagues can *securely share* key artifacts such as Clusters, Notebooks, Jobs and Workspaces
- Secure collaboration is enabled through a combination of:

Fine grained permissions: Defines who can do what on which artifacts (access control)

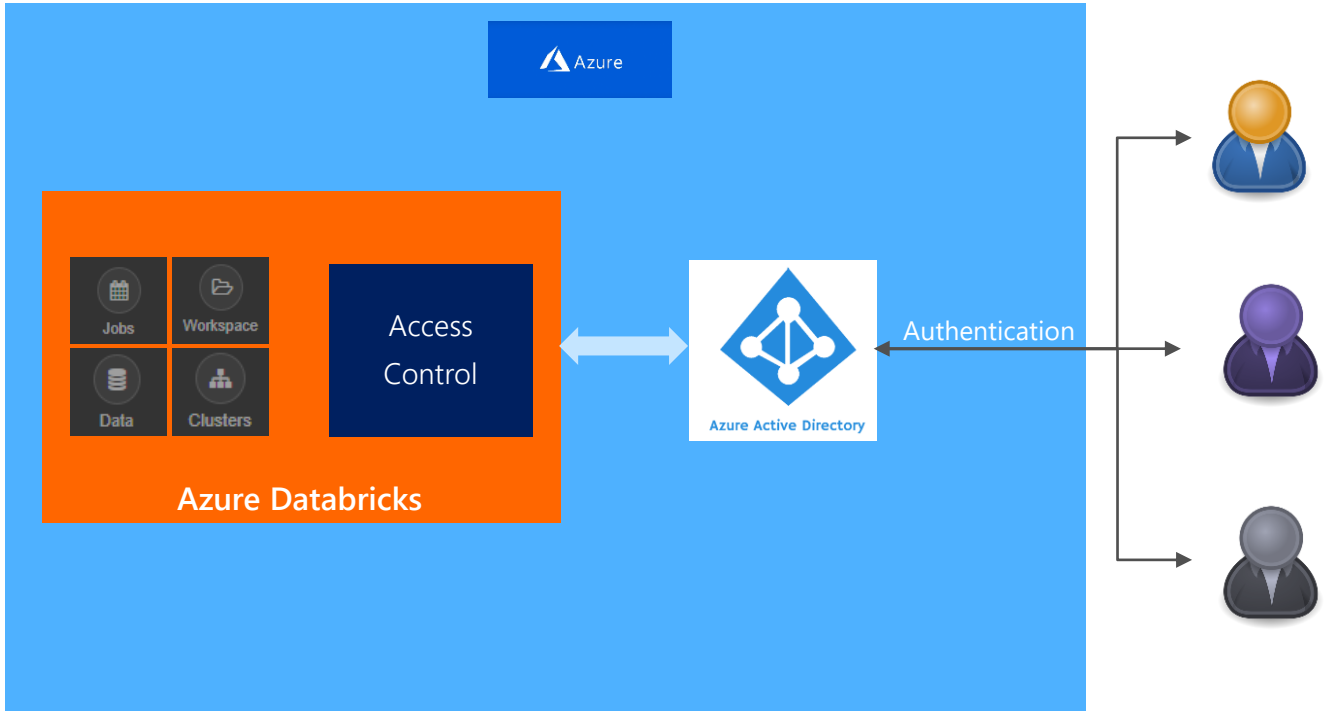
AAD-based authentication: Ensures that users are actually who they claim to be



AZURE DATABRICKS INTEGRATION WITH AAD

Azure Databricks is integrated with AAD—so Azure Databricks users are just regular AAD users

- There is no need to define users—and their access control—separately in Databricks.
- AAD users can be used directly in Azure Databricks for all user-based access control (Clusters, Jobs, Notebooks etc.).
- Databricks has delegated user authentication to AAD enabling single-sign on (SSO) and unified authentication.
- *Notebooks, and their outputs, are stored in the Databricks account*. However, AAD-based access-control ensures that only authorized users can access them.



DATABRICKS ACCESS CONTROL

Access control can be defined at the user level via the Admin Console

Access Control can be defined for Workspaces, Clusters, Jobs and REST APIs

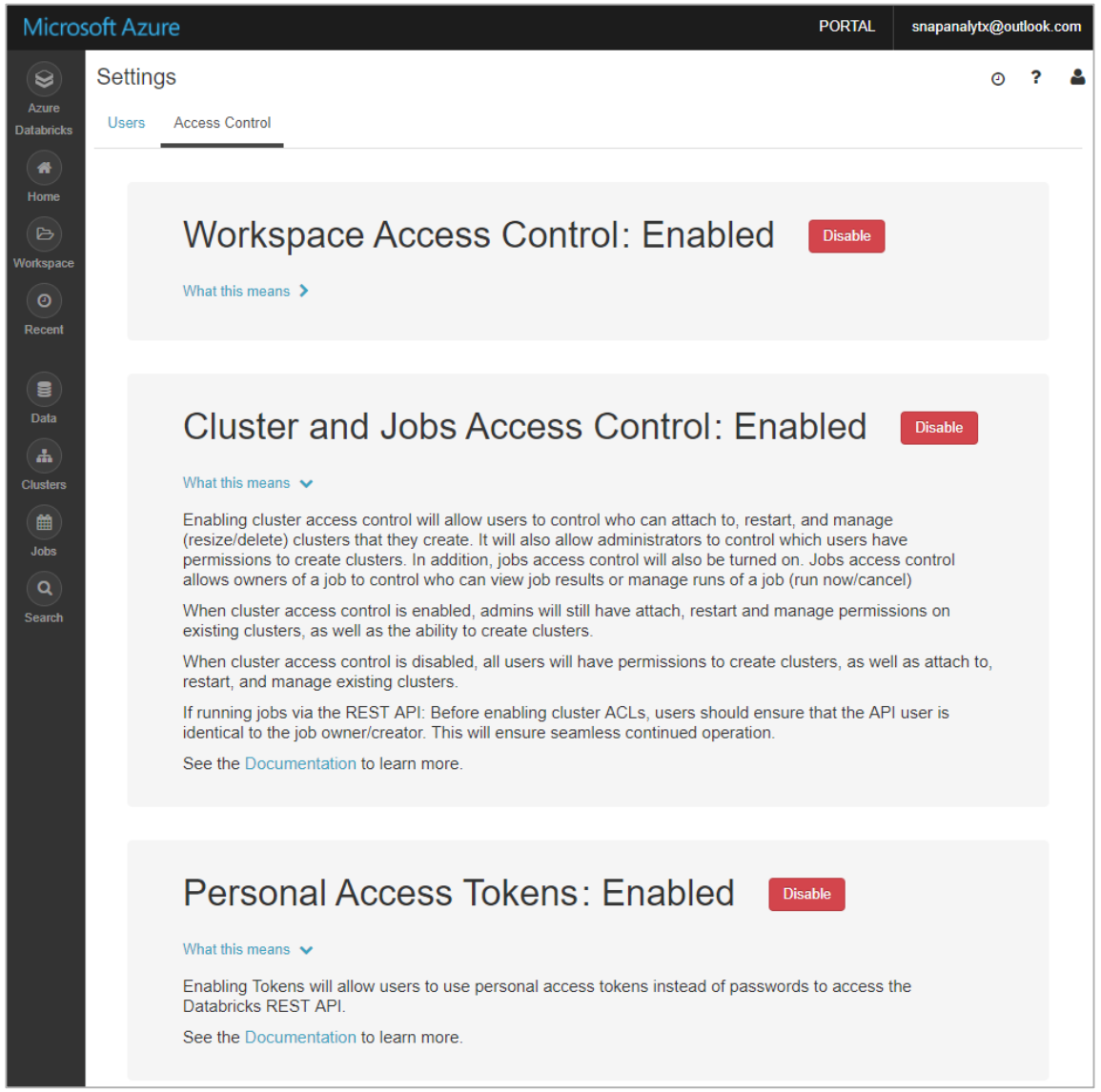
Databricks Access Control

Workspace Access Control	Defines who can who can view, edit, and run notebooks in their workspace
Cluster Access Control	Allows users to who can attach to, restart, and manage (resize/delete) clusters. Allows Admins to specify which users have permissions to create clusters
Jobs Access Control	Allows owners of a job to control who can view job results or manage runs of a job (run now/cancel)
REST API Tokens	Allows users to use personal access tokens instead of passwords to access the Databricks REST API

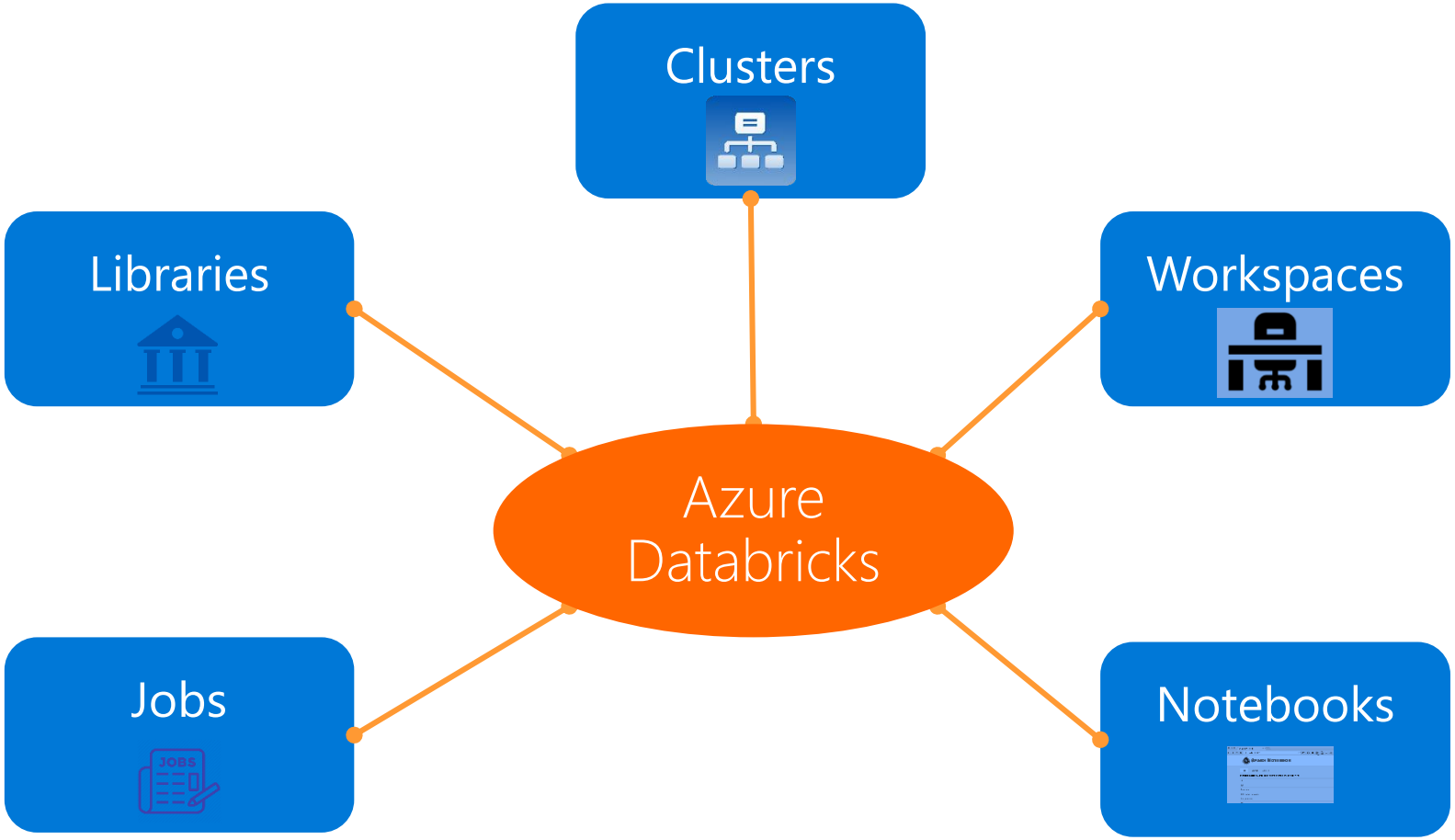
ENABLE/DISABLE ACCESS CONTROL

Access Control can be selectively enabled or disabled for:

- Workspaces,
- Clusters,
- Jobs
- REST APIs



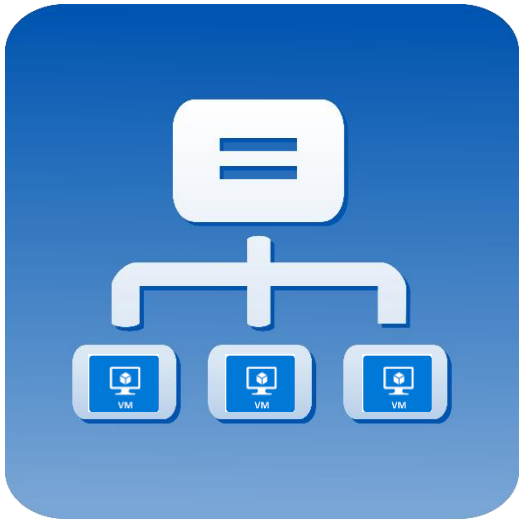
AZURE DATABRICKS CORE ARTIFACTS



Clusters

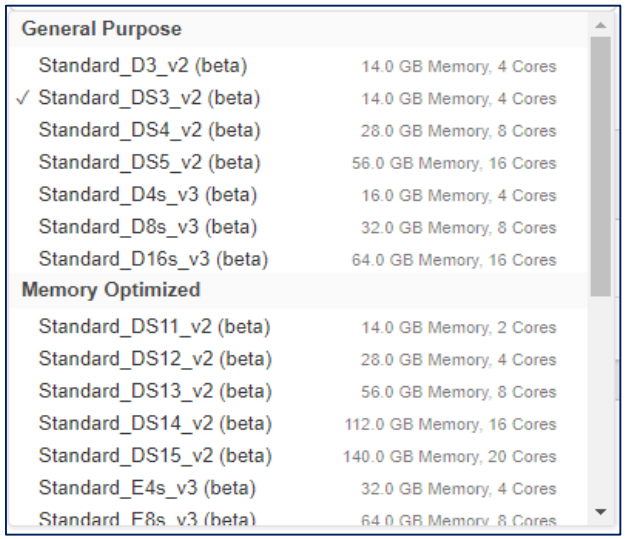
CLUSTERS

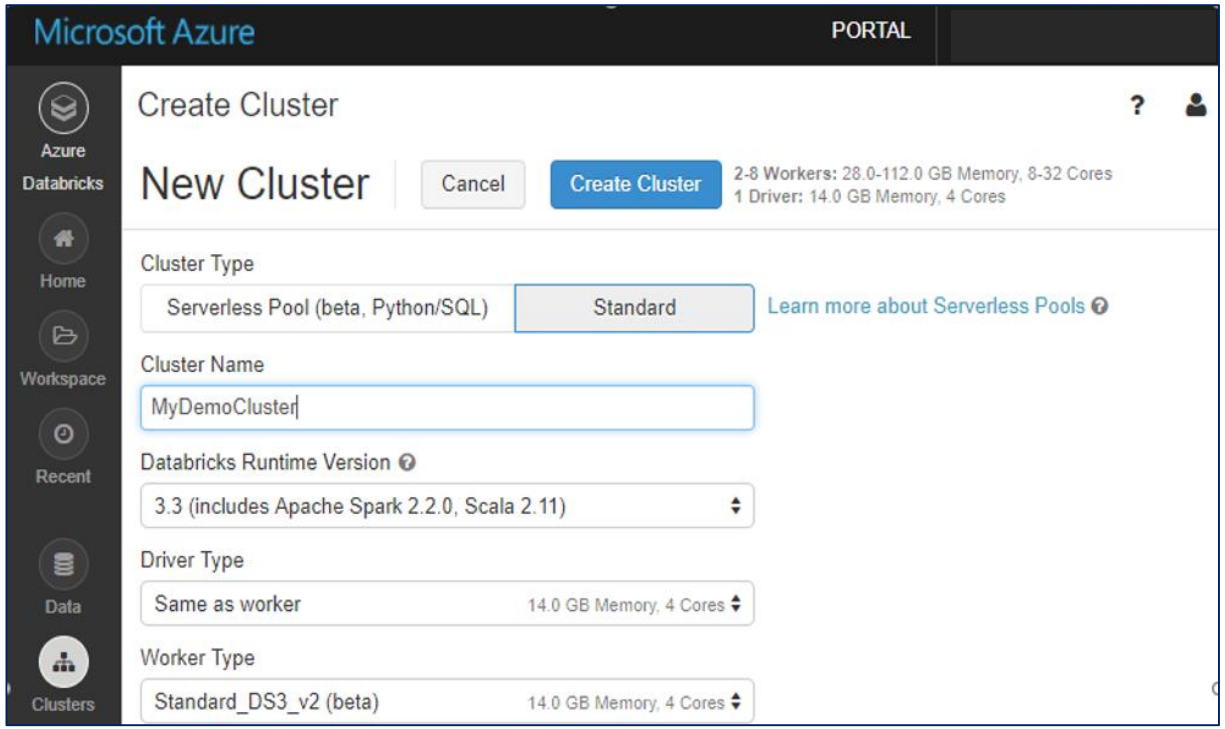
- Azure Databricks clusters are the set of Azure Linux VMs that host the Spark Worker and Driver Nodes
- Your Spark application code (i.e. Jobs) runs on the provisioned clusters.
- Azure Databricks clusters are launched in your subscription—but are managed through the Azure Databricks portal.
- Azure Databricks provides a comprehensive set of graphical wizards to manage the complete lifecycle of clusters—from creation to termination.



CLUSTER CREATION

- You can create two types of clusters – *Standard*
- While creating a cluster you can specify:
  - Number of nodes
  - Autoscaling and Auto Termination policy
  - Auto Termination policy
  - Spark Configuration details
  - The Azure VM instance types for the Driver and Worker Nodes





Graphical wizard in the Azure Databricks portal to create a Standard Cluster

CLUSTERS: AUTO SCALING AND AUTO TERMINATION

Simplifies cluster management and reduces costs by eliminating wastage

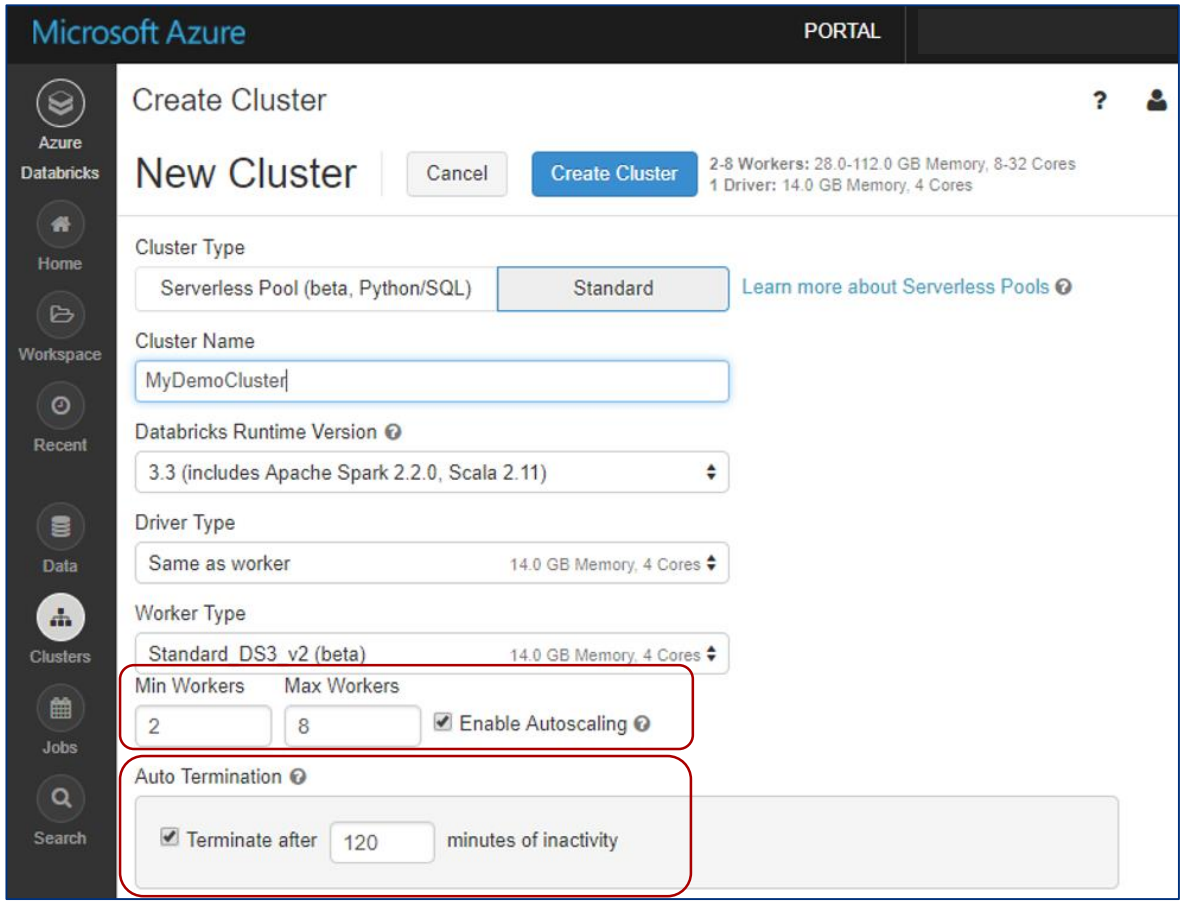
When creating Azure Databricks clusters you can choose Autoscaling and Auto Termination options.

Autoscaling: Just specify the min and max number of clusters. Azure Databricks automatically scales up or down based on load.

Auto Termination: After the specified minutes of inactivity the cluster is automatically terminated.

Benefits:

- You do not have to guess, or determine by trial and error, the correct number of nodes for the cluster
- As the workload changes you do not have to manually tweak the number of nodes
- You do not have to worry about wasting resources when the cluster is idle. You only pay for resource when they are actually being used
- You do not have to wait and watch for jobs to complete just so you can shutdown the clusters



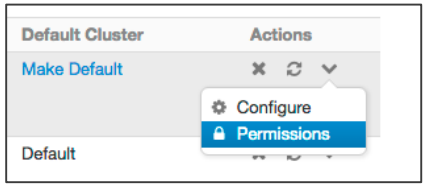
CLUSTER ACCESS CONTROL

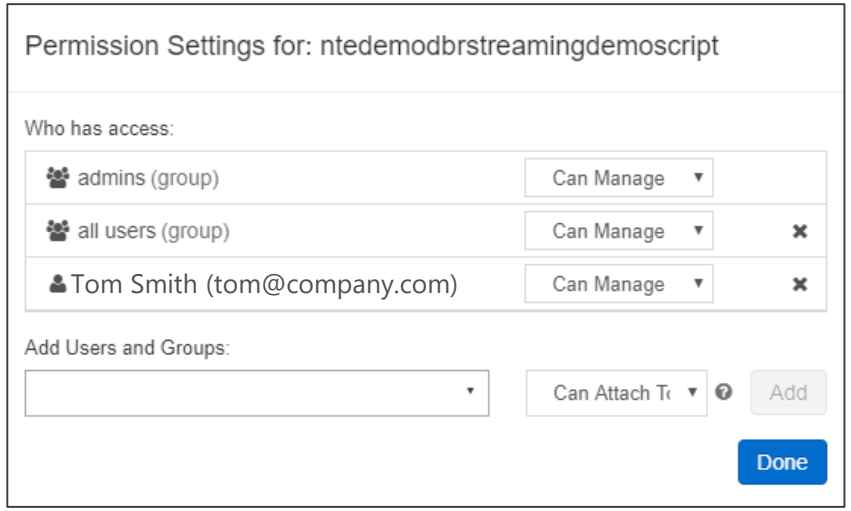
- There are two configurable types of permissions for Cluster Access Control:
  - *Individual Cluster Permissions* - This controls a user's ability to attach notebooks to a cluster, as well as to restart/resize/terminate/start clusters.
  - *Cluster Creation Permissions* - This controls a user's ability to create clusters

- Individual permissions can be configured on the Clusters Page by clicking on Permissions under the 'More Actions' icon of an existing cluster

- There are 4 different individual cluster permission levels: *No Permissions*, *Can Attach To*, *Can Restart*, *and Can Manage*. Privileges are shown below

Abilities	No Permissions	Can Attach To	Can Restart	Can Manage
Attach notebooks to cluster		x	x	x
View Spark UI		x	x	x
View cluster metrics (Ganglia)		x	x	x
Terminate cluster			x	x
Start cluster			x	x
Restart cluster			x	x
Resize cluster				x
Modify permissions				x





Jobs

J O B S

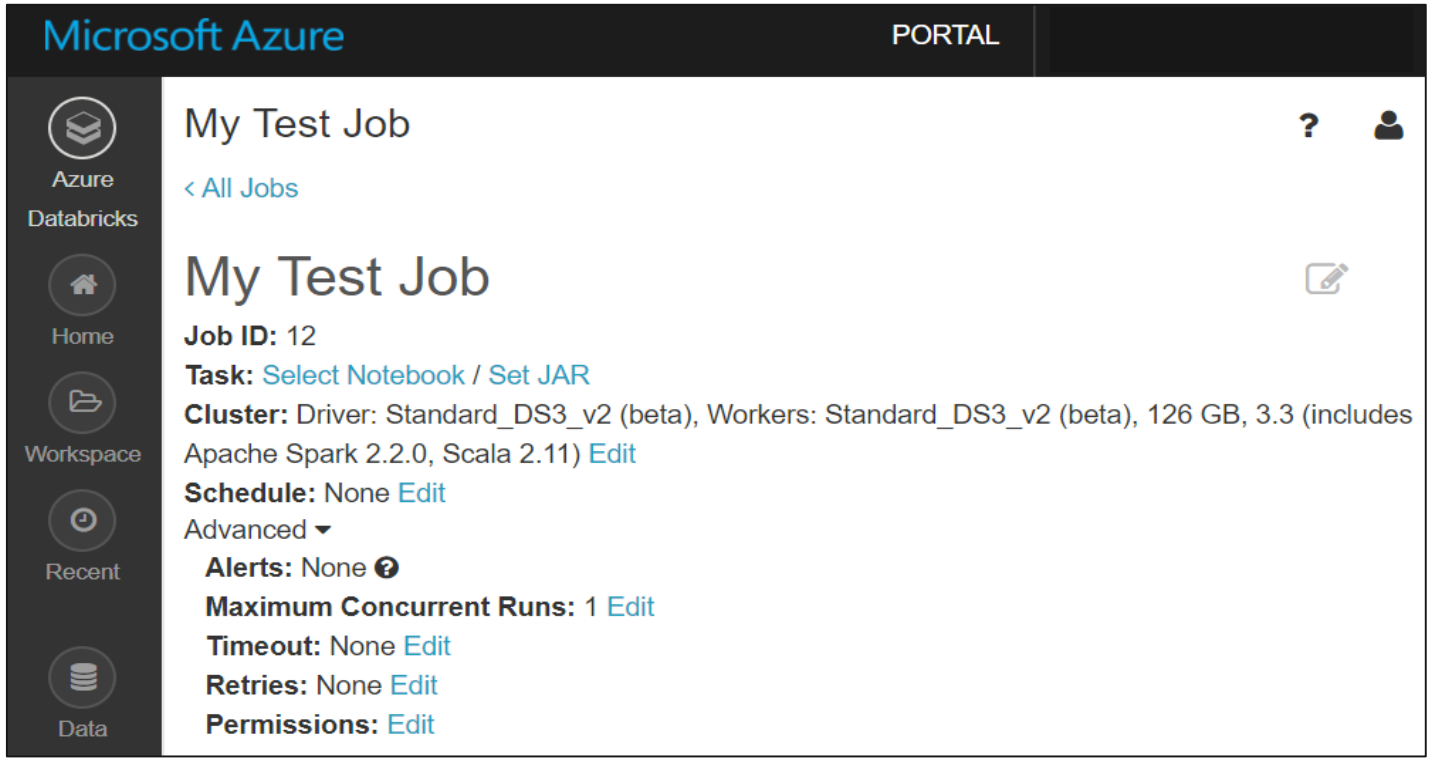
Jobs are the mechanism to submit Spark application code for execution on the Databricks clusters

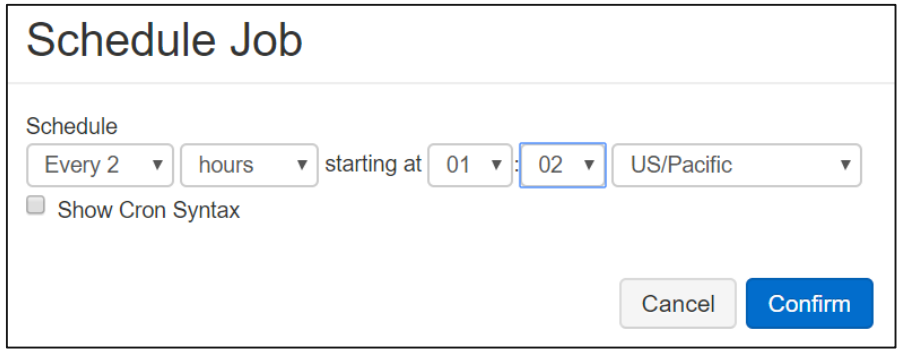
- Spark application code is submitted as a ‘Job’ for execution on Azure Databricks clusters
- Jobs execute either ‘Notebooks’ or ‘Jars’
- Azure Databricks provide a comprehensive set of graphical tools to create, manage and monitor Jobs.

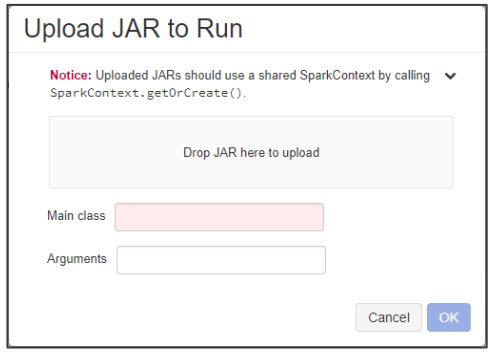
CREATING AND RUNNING JOBS (1 OF 2)

When you create a new Job you have to specify:

- The Notebook or Jar to execute
- Cluster: The cluster on which the Job execute. This could be an exiting or new cluster.
- Schedule i.e. how often the Job runs. Jobs can also be run one time right away.



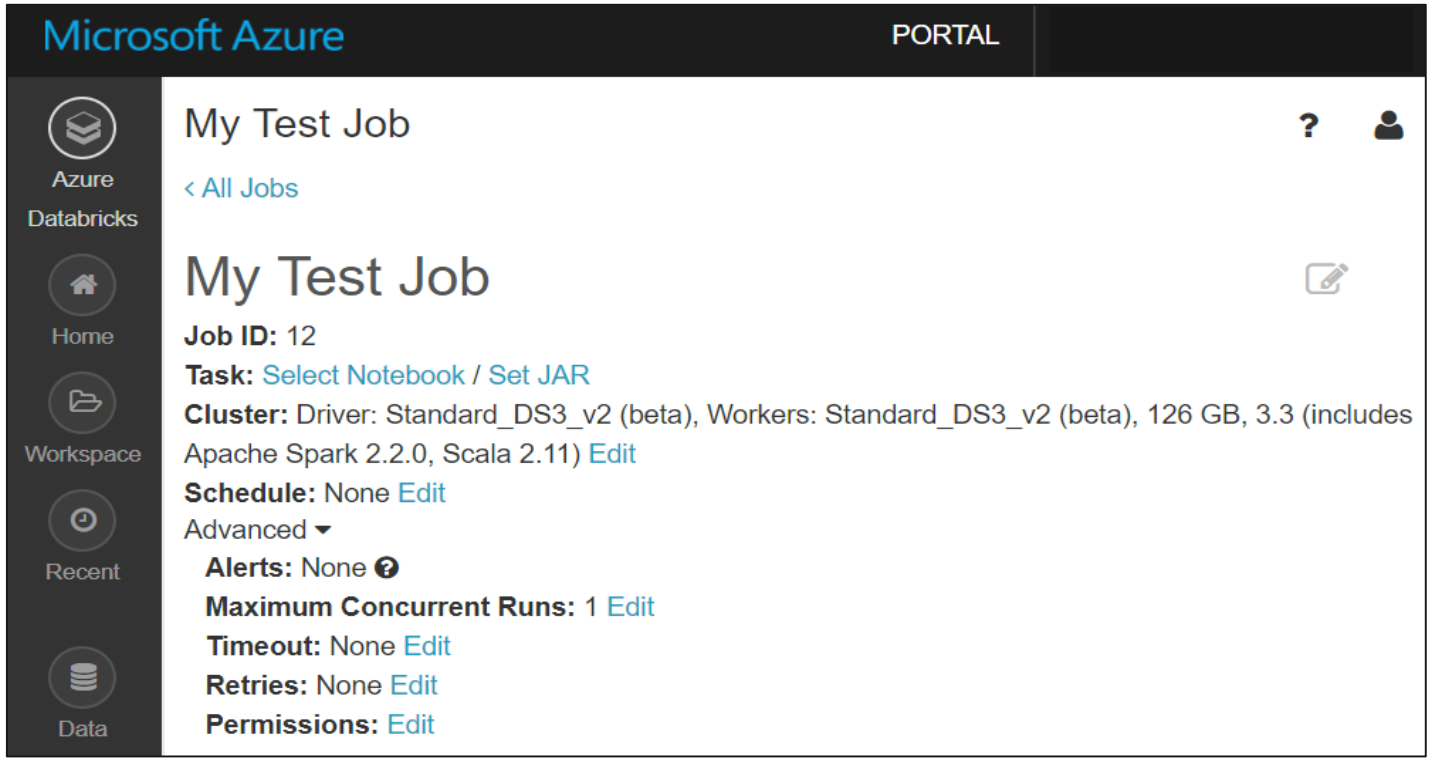


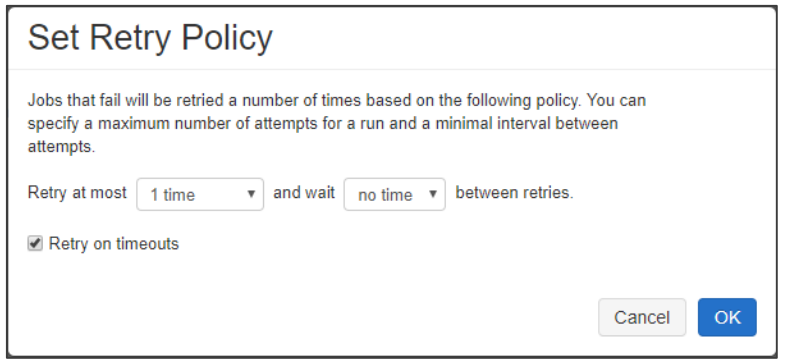


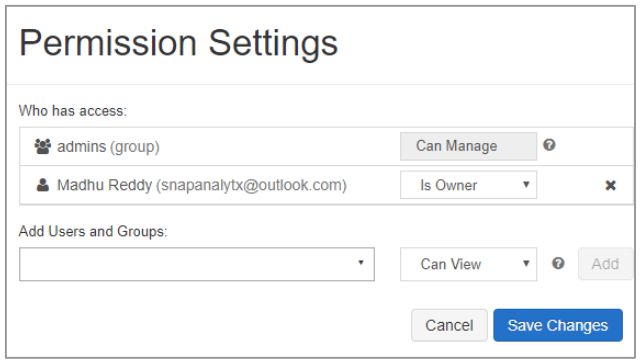
CREATING AND RUNNING JOBS (2 OF 2)

When you create a new job you can optionally specify advanced options:

- Maximum number of concurrent runs of the Job
- Timeout: Jobs still running beyond the specified duration are automatically killed
- Retry Policy: Specifies if—and when—failed jobs will be retried
- Permissions: Who can do what with jobs. This allows for Job definition and management to be *securely shared* with others (see next slide)







JOB ACCESS CONTROL

Enables job owners and administrators to grant fine grained permissions on their jobs

- With Jobs Access Controls job owners can choose which other users or groups can view results of the job.
- Owners can also choose who can manage runs of their job (i.e. invoke run now and cancel.)
- There are 5 different permission levels for jobs:
  - No Permissions
  - Can View
  - Can Manage Run
  - Is Owner and
  - Can Manage

Abilities	No Permissions	Can View	Can Manage Run	Is Owner	Can Manage (admin)
View job details and settings	Yes	Yes	Yes	Yes	Yes
View results, Spark UI, logs of a job run		Yes	Yes	Yes	Yes
Run now			Yes	Yes	Yes
Cancel run			Yes	Yes	Yes
Edit job settings				Yes	Yes
Modify permissions				Yes	Yes
Delete job				Yes	Yes
Change owner					Yes

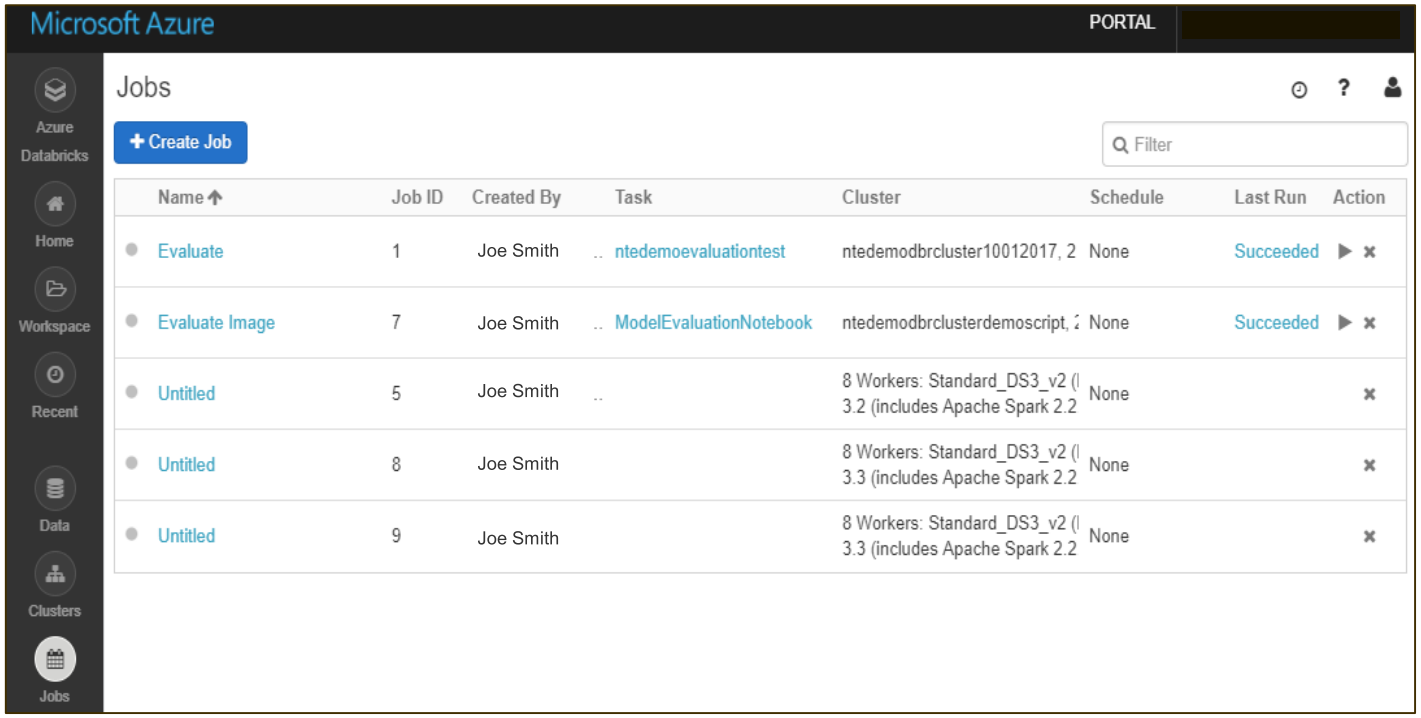
Note: 'Can Manage' permission is reserved for administrators.

VIEWING LIST OF JOBS

In the Portal you can view the list of all jobs you have access to

You can click on “Run Now” icon ▶ to run the job right away

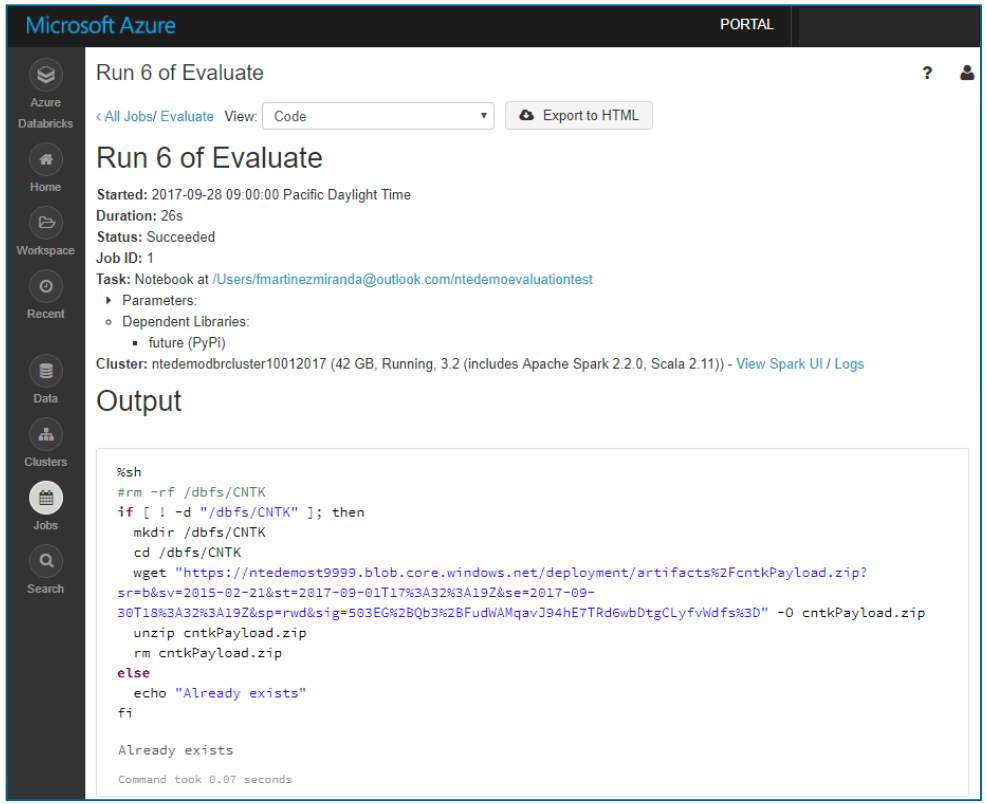
You can also delete a job from the list

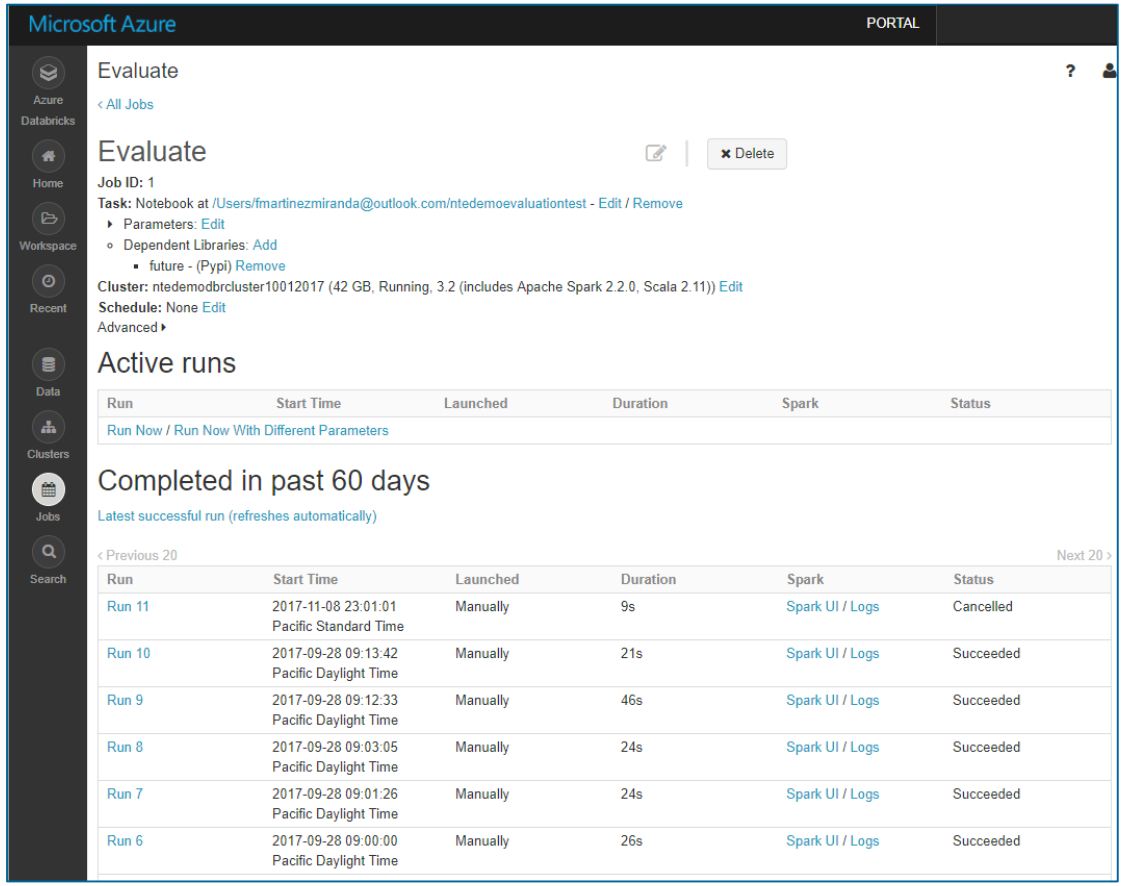


VIEWING JOBS HISTORY

In the Azure Databricks Jobs Portal you can view:

- The list of currently running (Active) Jobs
- History of old Job runs (for up to 60 days)
- The output of a particular Job run (including standard error, standard output, Spark UI logs)



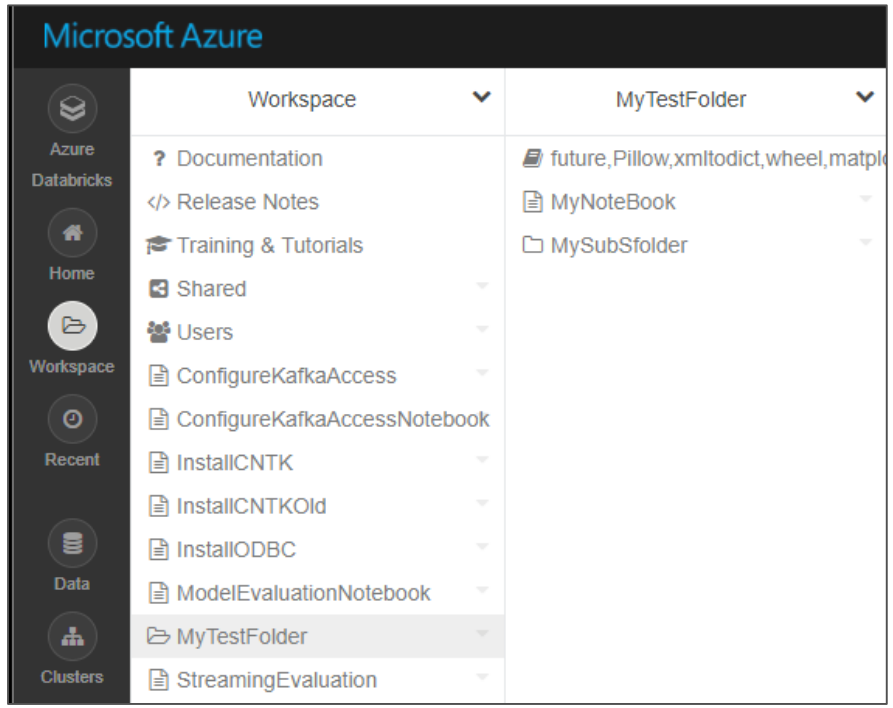


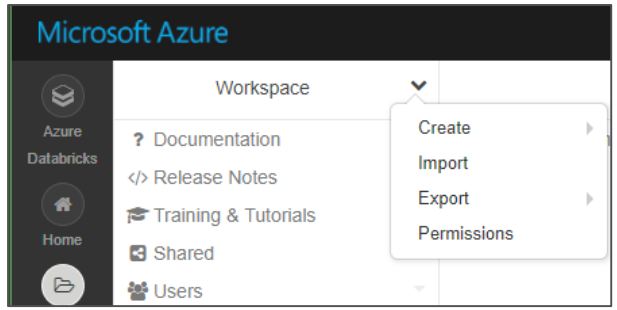
Workspaces & Folders

WORKSPACES

Workspaces enables users to organize—and share—their Notebooks, Libraries and Dashboards

- Workspaces—sort of like Directories— are a convenient way to organize an user's Notebook, Libraries and Dashboards.
- Everything in a workspace is organized into hierarchical folders. Folders can hold Libraries, Notebooks, Dashboard or more (sub) folders.
  - Icons indicate the type of the object contained in a folder
- Every user has one directory that is private and unshared.
  - By default, the workspace and all its contents are available to users.
- Fine grained access control can be defined on workspaces (next slide) to enable *secure collaboration with colleagues*.



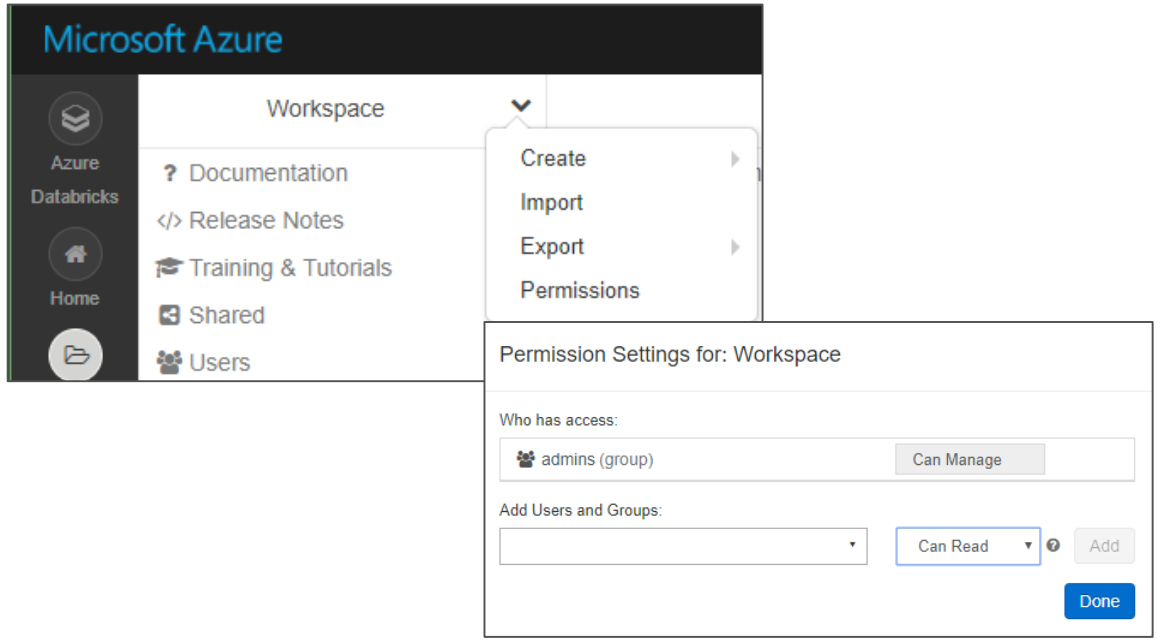


WORKSPACE OPERATIONS

You can search the entire Databricks workspace

In the Azure Databricks Portal, via the Workspaces drop down menu, you can:

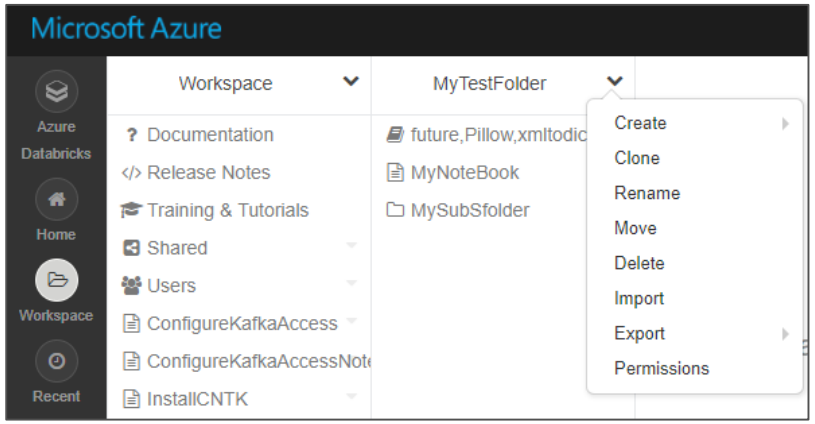
- Create Folders, Notebooks and Libraries
- Import Notebooks into the Workspace
- Export the Workspace to a database archive
- Set Permissions. You can grant 4 levels of permissions
  - Can Manage
  - Can Read
  - Can Edit
  - Can Run



FOLDER OPERATIONS AND ACCESS CONTROL

In the Azure Databricks Portal, via the Folder drop down menu, you can:

- Create Folders, Notebooks and Libraries within the folder
- Clone the folder to create a deep copy of the folder
- Rename or delete the folder
- Move the folder to another location
- Export a folder to save it and its contents as a Databricks archive
- Import a saved Databricks archive into the selected folder
- Set Permissions for the folder. As with Workspaces you can set 5 levels of permissions: *No Permissions, Can Manage, Can Read, Can Edit, Can Run*



Abilities	No Permissions	Read	Run	Edit	Manage
Create items					✅
Delete items					✅
Move/rename items					✅
Change permissions					✅

Abilities associated with each permission level

Notebooks, Libraries, Visualization

AZURE DATABRICKS NOTEBOOKS OVERVIEW

Notebooks are a popular way to develop, and run, Spark Applications

- Notebooks are not only for authoring Spark applications but can be *run/executed directly* on clusters
  - Shift+Enter
  - click the ▶ at the top right of the cell in a notebook
  - Submit via Job
- Notebooks support fine grained permissions—so they can be *securely shared* with colleagues for collaboration (see following slide for details on permissions and abilities)
- Notebooks are well-suited for prototyping, rapid development, exploration, discovery and iterative development

Notebooks typically consist of code, data, visualization, comments and notes

MIXING LANGUAGES IN NOTEBOOKS

You can mix multiple languages in the same notebook

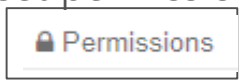
Normally a notebook is associated with a specific language. However, with Azure Databricks notebooks, you can mix multiple languages in the same notebook. This is done using the language magic command:

- %python Allows you to execute python code in a notebook (even if that notebook is not python)
- %sql Allows you to execute sql code in a notebook (even if that notebook is not sql).
- %r Allows you to execute r code in a notebook (even if that notebook is not r).
- %scala Allows you to execute scala code in a notebook (even if that notebook is not scala).
- %sh Allows you to execute shell code in your notebook.
- %fs Allows you to use Databricks Utilities - dbutils filesystem commands.
- %md To include rendered markdown

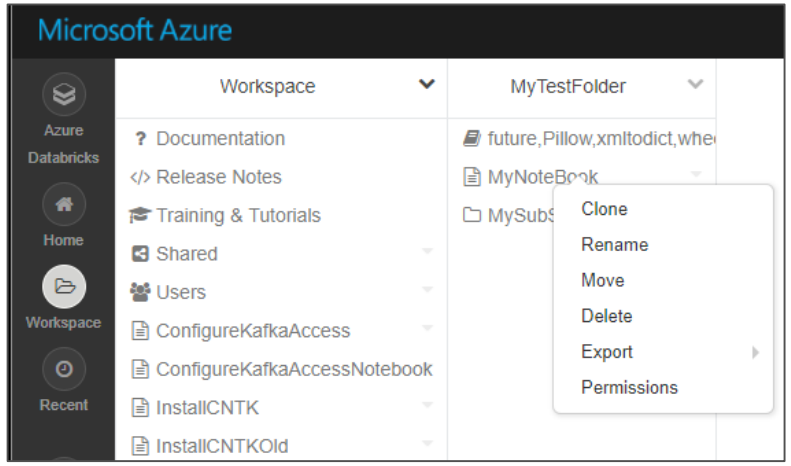
NOTEBOOK OPERATIONS AND ACCESS CONTROL

You can create a new notebook from the Workspace or the folder drop down menu (see previous slides)

From a notebook's drop down menu you can:

- Clone the notebook
- Rename or delete the notebook
- Move the notebook to another location
- Export a notebook to save it and its contents as a Databricks archive or IPython notebook or HTML or source code file.
- Set Permissions for the notebook As with Workspaces you can set 5 levels of permissions: *No Permissions, Can Manage, Can Read, Can Edit, Can Run*
- You can also set permissions from notebook UI itself by selecting the 

menu option.



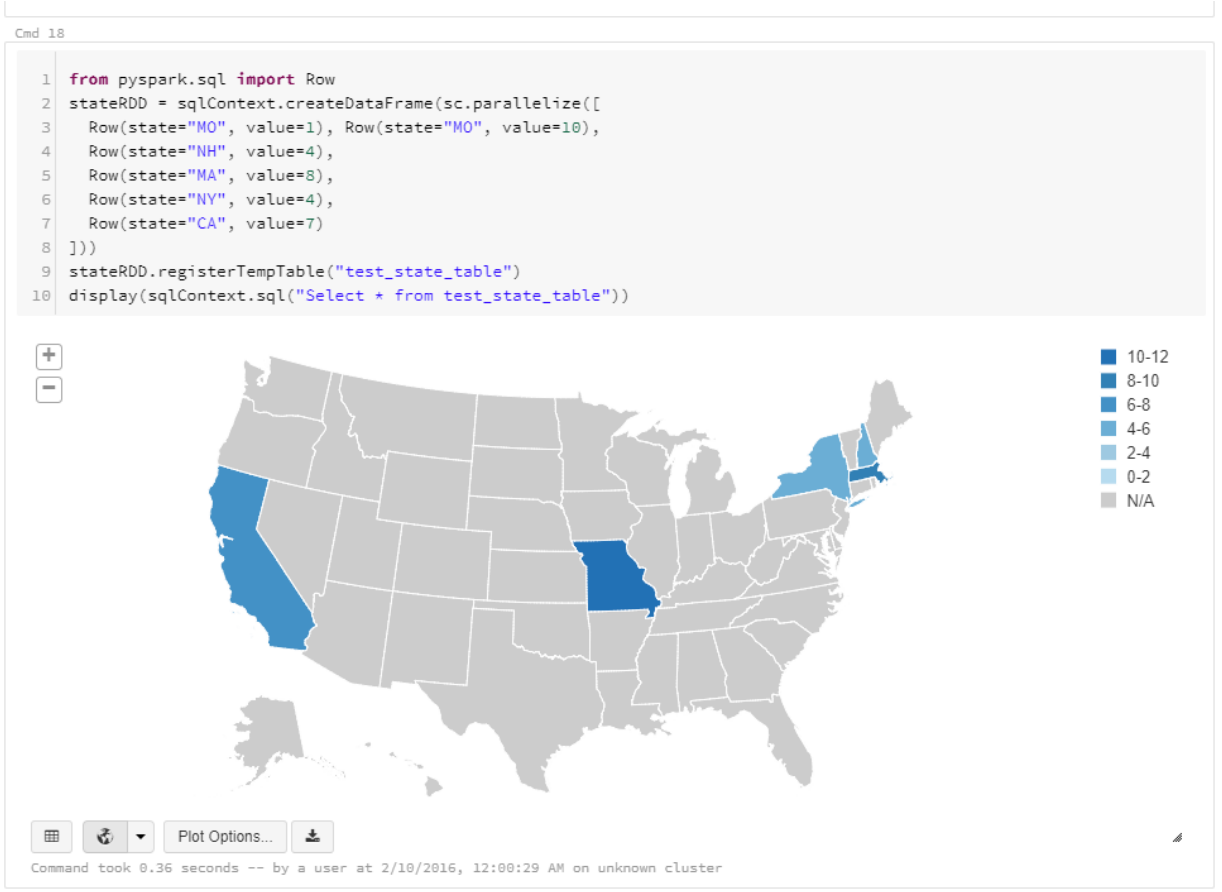
Abilities	No Permissions	Read	Run	Edit	Manage
View cells		✅	✅	✅	✅
Comment		✅	✅	✅	✅
Run Commands			✅	✅	✅
Attach/detach notebooks			✅	✅	✅
Edit cells				✅	✅
Change permissions					✅

Abilities associated with each permission level

VISUALIZATION

Azure Databricks supports a number of visualization plots out of the box

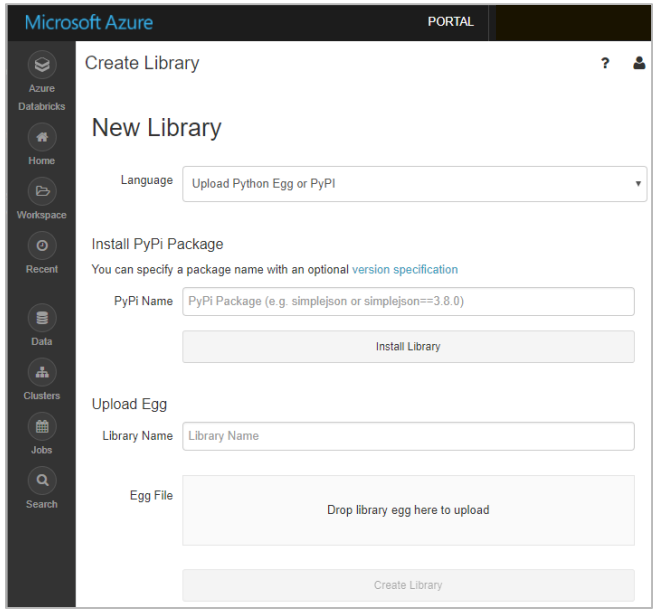
- All notebooks, *regardless of their language*, support Databricks visualizations.
- When you run the notebook the visualizations are rendered inside the notebook in-place
- The visualizations are written in HTML.
  - You can save the HTML of the entire notebook by exporting to HTML.
  - If you use Matplotlib, the plots are rendered as images so you can just right click and download the image
- You can change the plot type just by picking from the selection

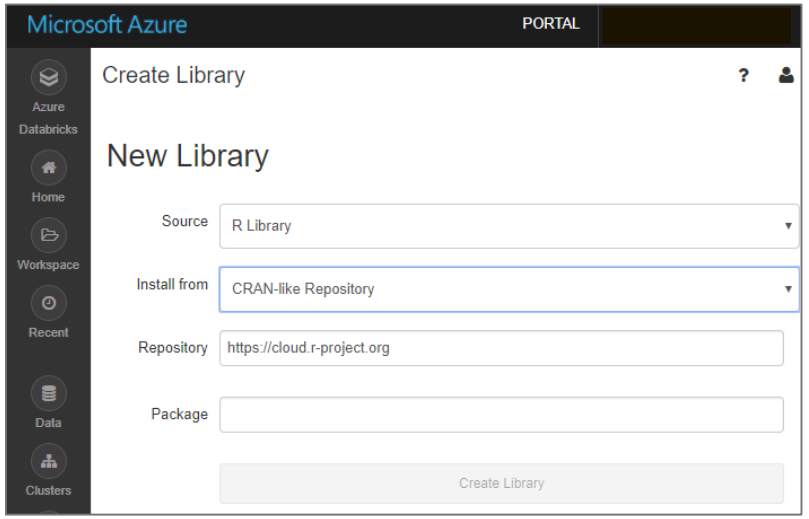


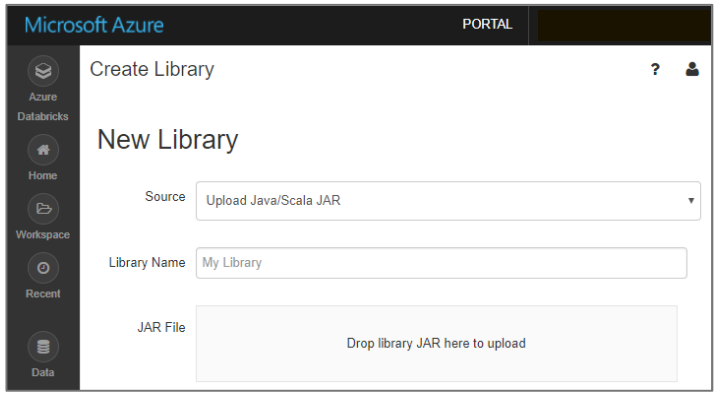
LIBRARIES OVERVIEW

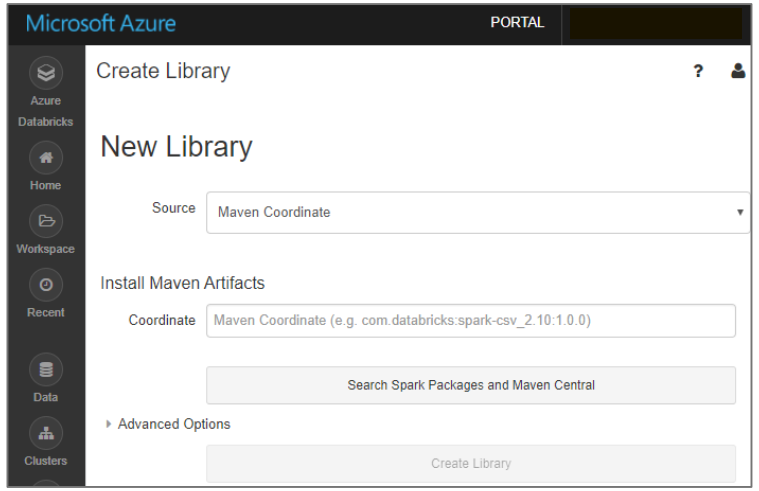
Enables external code to be imported and stored into a Workspace

- Libraries are containers to hold all your *Python, R, Java/Scala* libraries.
- Libraries resides within workspaces or folders.
- Libraries are created by importing the source code
- After importing libraries are immutable—can be deleted or overwritten only.
- You can customize installation of libraries via Init Scripts by writing custom UNIX scripts
- Libraries can also be managed via the Library API





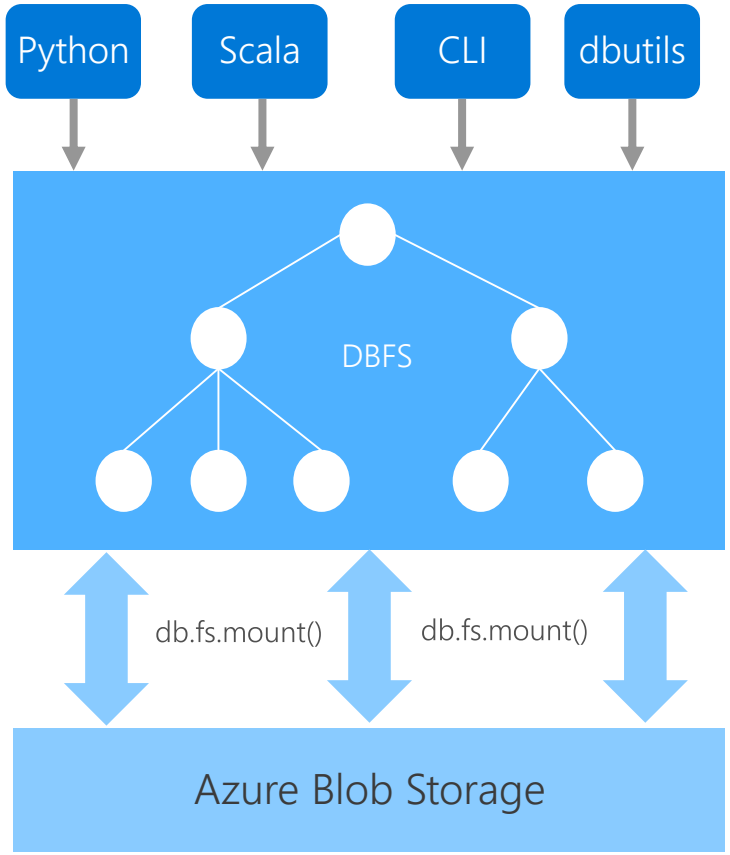




DATABRICKS FILE SYSTEM (DBFS)

Is a distributed File System (DBFS) that is a layer over Azure Blob Storage

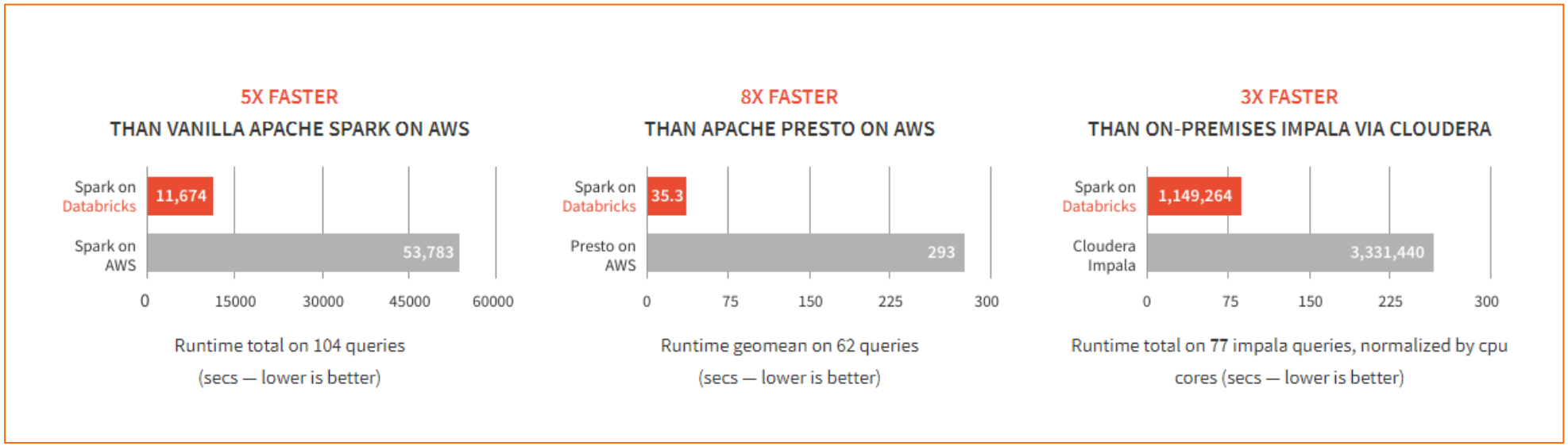
- Azure Storage buckets can be mounted in DBFS so that users can directly access them without specifying the storage keys
- DBFS mounts are created using *dbutils.fs.mount()*
- Azure Storage data can be cached locally on the SSD of the worker nodes
- Available in both Python and Scala and accessible via a DBFS CLI
- Data persist in Azure Blob Storage – is not lost even after cluster termination
- Comes pre-installed on Spark clusters in Databricks



Azure Databricks Performance

DATABRICKS SPARK IS FAST

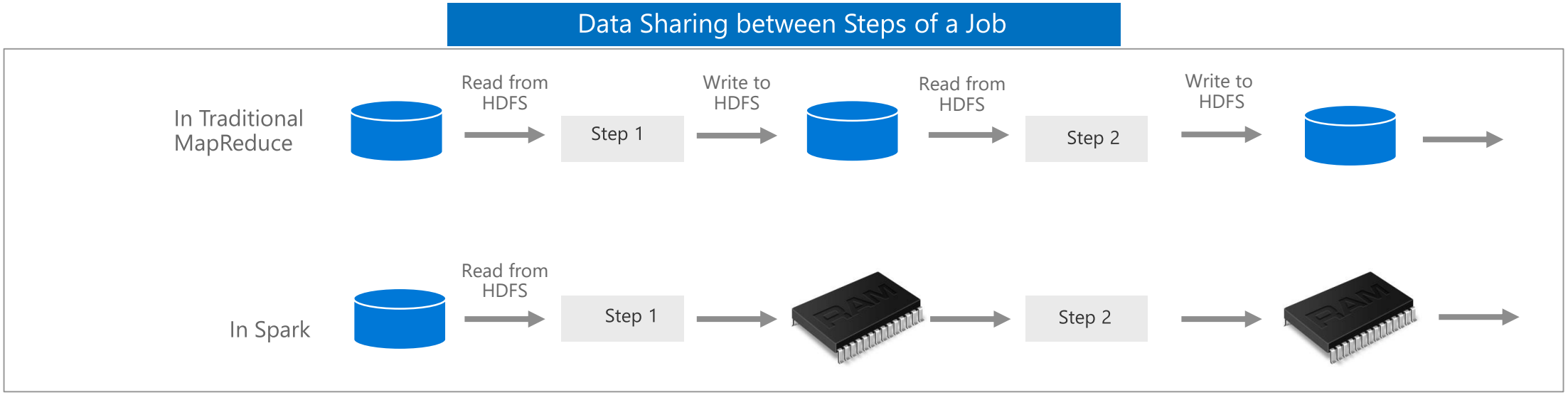
Benchmarks have shown Databricks to often have better performance than alternatives



SOURCE: Benchmarking Big Data SQL Platforms in the Cloud

WHAT MAKES SPARK FAST?(1OF 2)

- **In-memory cluster computing:** Spark provides primitives for *in-memory* cluster computing. A Spark job can *load and cache* data into memory and query it repeatedly (iteratively) much quicker than disk-based systems.
- **Scala Integration:** Spark integrates into the Scala programming language, letting you manipulate distributed datasets like local collections. No need to structure everything as map and reduce operations
- **Faster Data-sharing:** Data-sharing between operations is faster as data is in-memory:
  - In (traditional) Hadoop data is shared through HDFS which is expensive. HDFS maintains three replicas.
  - Spark stores data in-memory *without any replication*.



WHAT MAKES SPARK FAST? (2 OF 2)

Databricks IO Cache automatically caches 'remote' data on 'local nodes' to accelerate data reads

- A copy of the remote file is created in the node's local storage
  - Local data is stored in a fast intermediate format
  - Currently *Parquet* file format is supported
- Remote data is cached automatically
- Supports *DBFS, HDFS, Azure Blob Storage and Azure Data Lake store*
- DBIO Cache lets you"
  - Enable or disable caching at anytime
  - Cache only a select subset of the data
- DBIO Cache has to be configured during cluster creation. The '*max disk space per node reserved for cached data*' must be specified during cluster creation

You can Monitor the state of the DBIO cache in the Portal

Storage

Parquet IO Cache

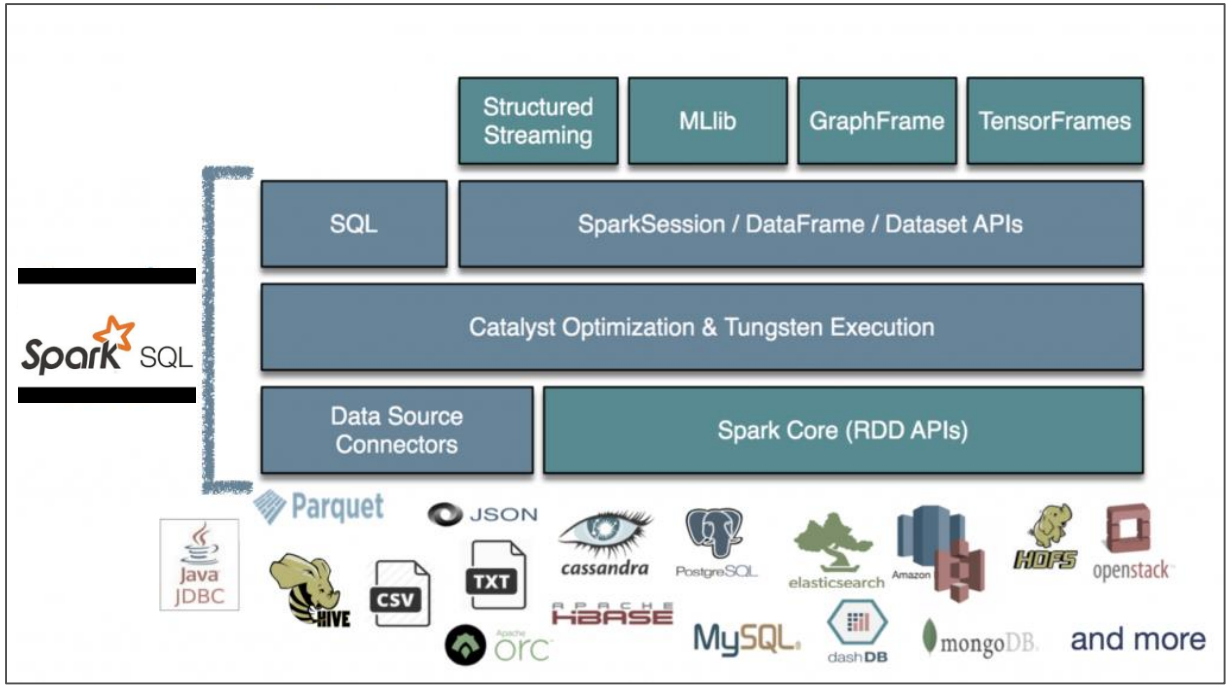
Host	Disk Usage	Max Disk Usage Limit	Percent Disk Usage	Metadata Cache Size	Max Metadata Cache Size Limit	Percent Metadata Usage
10.0.185.226	8.3 GB	442.4 GB	1 %	6.8 MB	8.8 GB	0 %
10.0.194.201	8.2 GB	442.4 GB	1 %	6.8 MB	8.8 GB	0 %
10.0.199.229	8.2 GB	442.4 GB	1 %	6.9 MB	8.8 GB	0 %
10.0.215.147	8.1 GB	442.4 GB	1 %	7.0 MB	8.8 GB	0 %
Total	32.8 GB	1769.5 GB	1 %	27.4 MB	35.4 GB	0 %

Data Analytics

SPARK SQL OVERVIEW

Spark SQL is a distributed SQL query engine for processing structured data

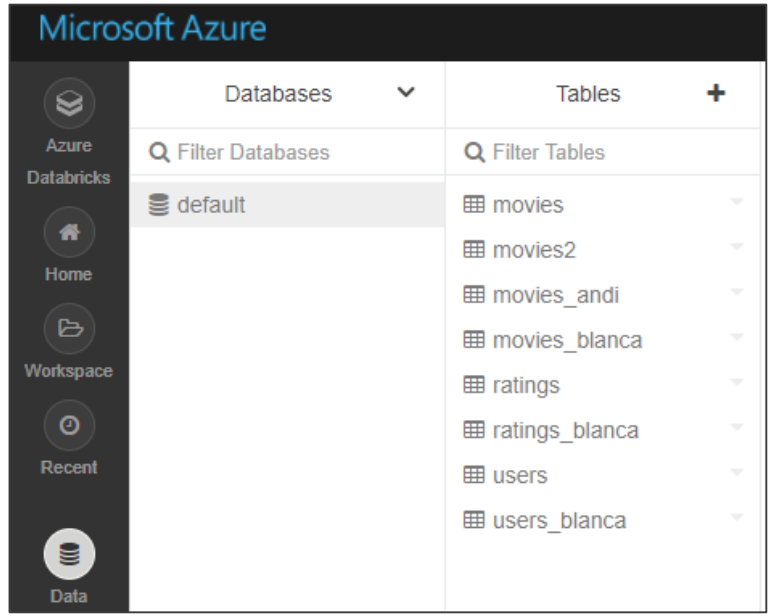
- Can query data stored in wide variety of data sources—external databases, structured data files, Hive tables and more.
- Data can be queried using either SQL or HiveQL
- Has bindings in Python, Scala and Java
- Has built-in support for structured streaming.
- Built using the Catalyst optimizer and Tungsten execution



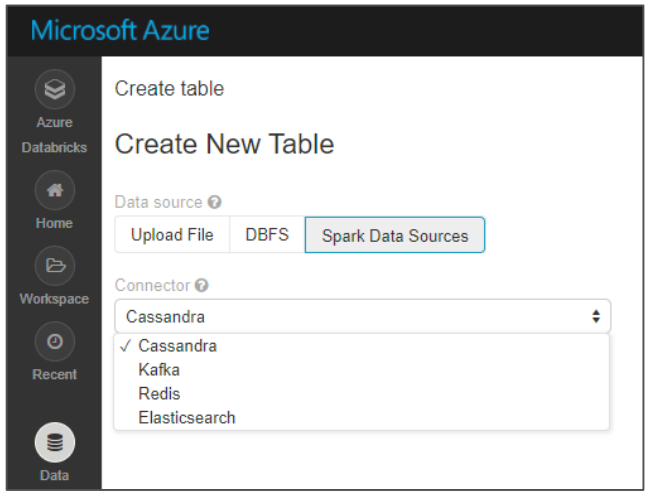
DATABASES AND TABLES OVERVIEW

Tables enable data to be structured and queried using Spark SQL or any of the Spark's language APIs

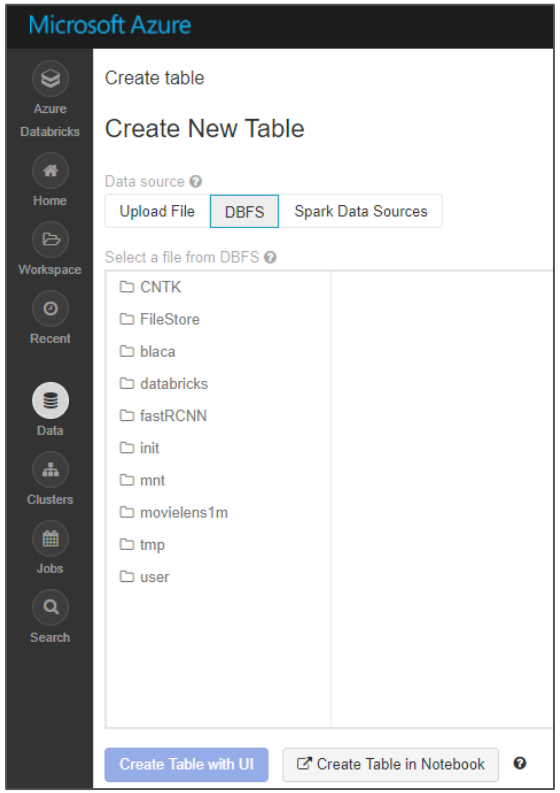
- Databases are a collection of related tables
- Tables are defined using the GUI in the console or programmatically using APIs or Notebooks
- Databricks uses the Hive metastore to manage tables, and supports all file formats and Hive data sources.
- There are multiple ways to create tables (see next slide).
- Like Apache Spark DataFrames, any Spark operation can be applied to Tables (including caching, filtering).
- Partitioned Tables and Partition Pruning: Spark SQL is able to dynamically generate partitions at the file storage level to provide partition columns for tables. When the table is scanned, Spark pushes down the filter predicates involving the partitionBy keys for partition pruning.



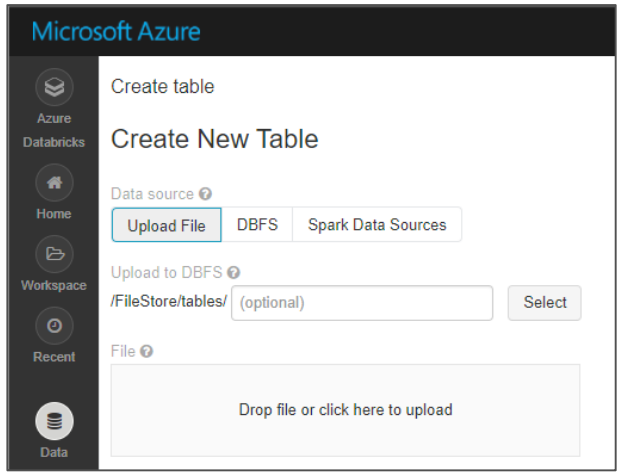
WAYS TO CREATE TABLES



From Spark Data Sources



From data in DBFS



From local files (in CSV, JSON or Avro formats)

Note: You can also create tables programmatically (CREATE TABLE tablename ...)

TABLE OPERATIONS

Azure Databricks tables support the following operations

- **Listing** database and tables
- **Viewing** table details including its schema and sample data
- **Reading** from tables
- **Updating** tables: Table schema is immutable. However, a user can update table data by changing the underlying files.
- **Deleting** tables: A user can delete tables either through the UI or programmatically

From SQL:

```
SELECT * FROM diamonds
```

From Python, use one of these examples:

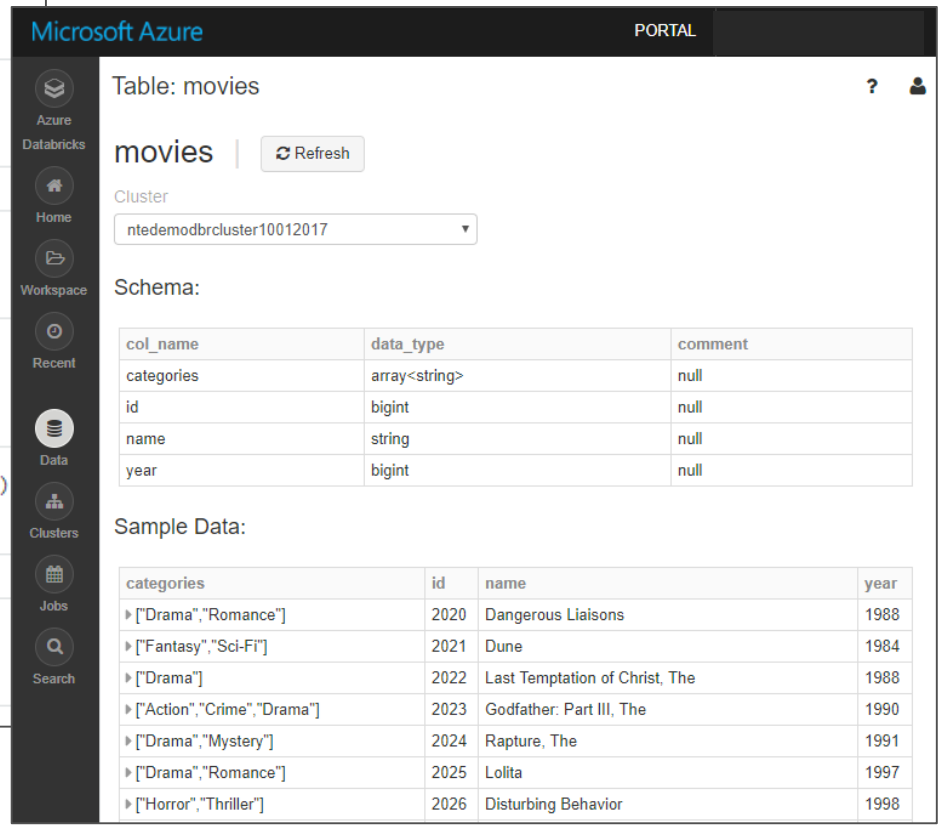
```
diamonds = spark.sql("SELECT * FROM diamonds")
display(diamonds.select("*"))
```

```
diamonds = spark.table("diamonds")
display(diamonds.select("*"))
```

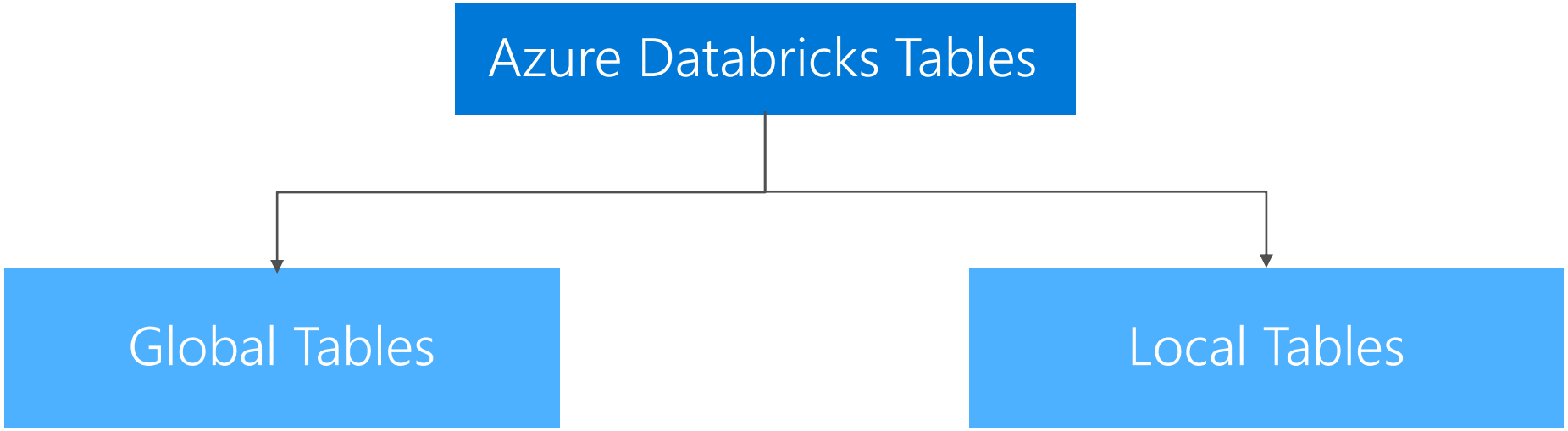
From Scala, use one of these examples:

```
val diamonds = spark.sql("SELECT * FROM diamonds")
display(diamonds.select("*"))
```

```
val diamonds = spark.table("diamonds")
display(diamonds.select("*"))
```



LOCAL AND GLOBAL TABLES



Databricks registers global tables to the Hive metastore and makes them available across all clusters.

Only global tables are visible in the Tables pane

Databricks does not registers local tables in the Hive metastore and are only available within one cluster.

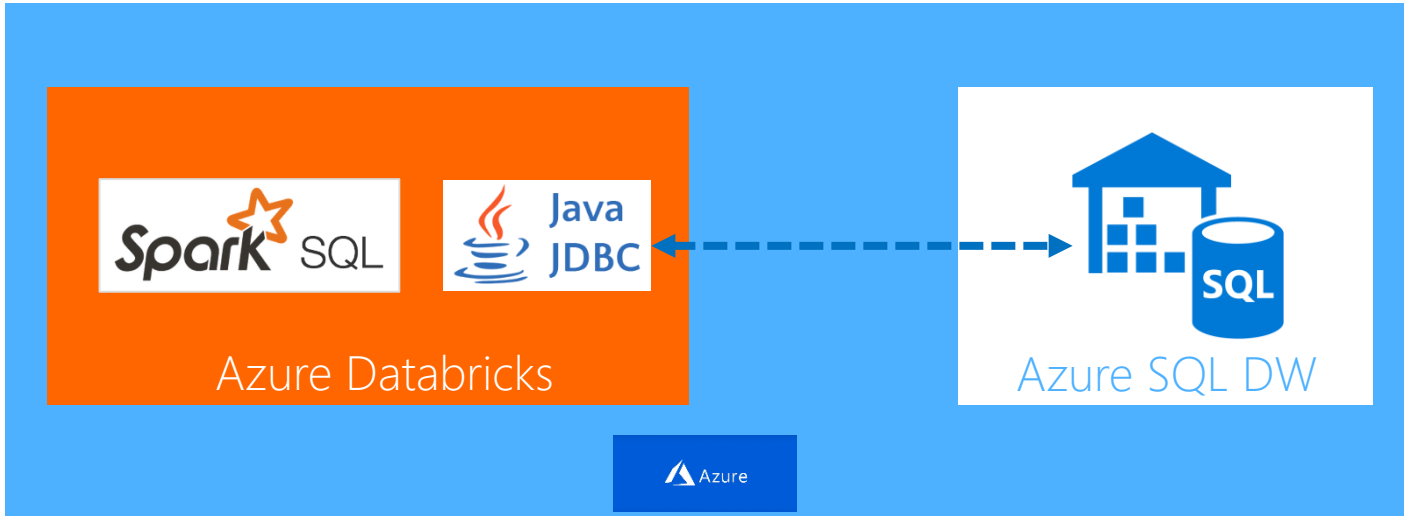
Also known as *temporary* tables

AZURE SQL DW INTEGRATION

Integration enables structured data from SQL DW to be included in Spark Analytics

Azure SQL Data Warehouse is a SQL-based fully managed, petabyte-scale cloud solution for data warehousing

- You can bring in data from Azure SQL DW to perform advanced analytics that require both structured and unstructured data.
- Currently you can access data in Azure SQL DW via the JDBC driver. From within your spark code you can access just like any other JDBC data source.
- If Azure SQL DW is authenticated via AAD then Azure Databricks user can seamlessly access Azure SQL DW.



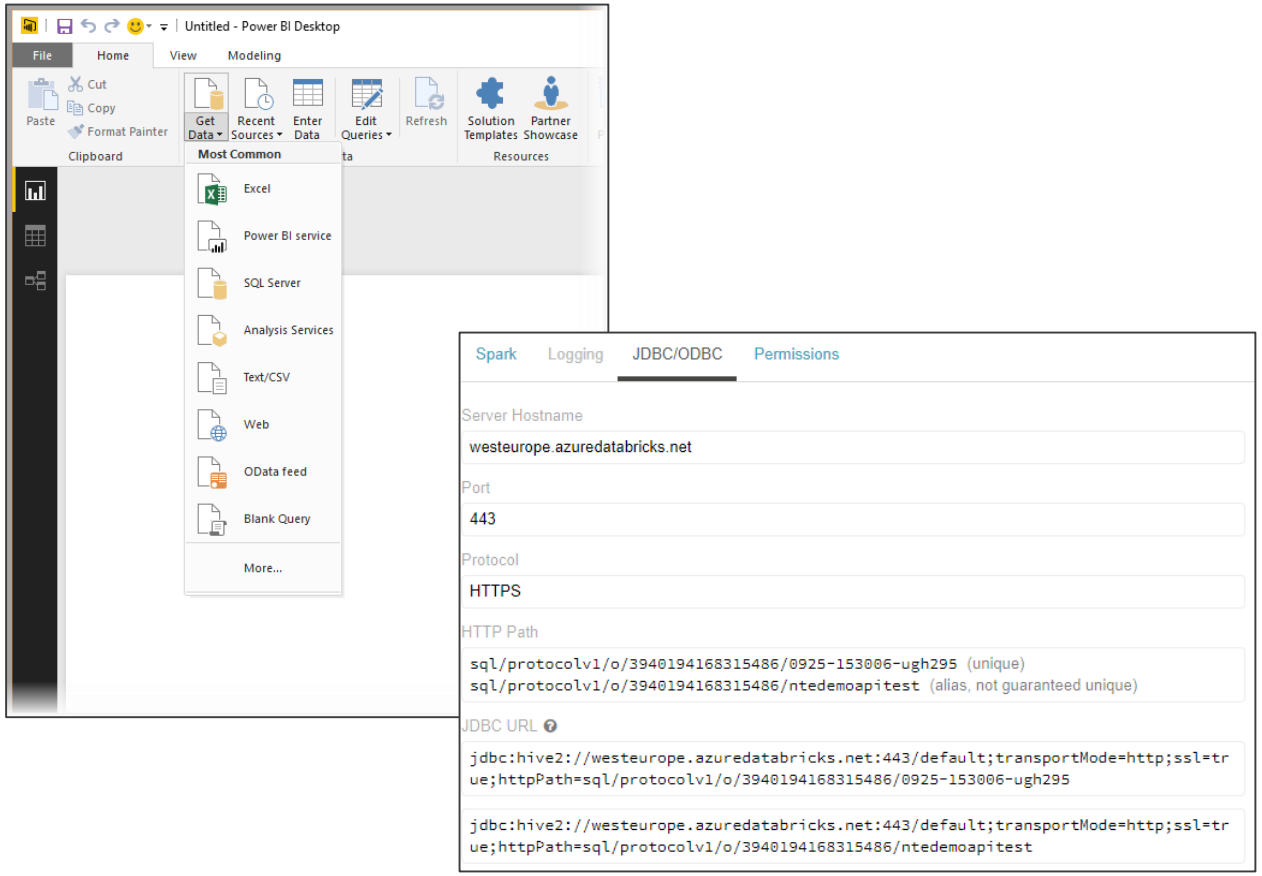
POWER BI INTEGRATION

Enables powerful visualization of data in Spark with Power BI

Power BI is a business analytics tool that provides data Visualization, Report and Dashboard throughout an organization

Power BI Desktop can connect to Azure Databricks clusters to query data using JDBC/ODBC server that runs on the driver node.

- This server listens on port 10000 and it is not accessible outside the subnet where the cluster is running.
- Azure Databricks uses a public HTTPS gateway
- The JDBC/ODBC connection information can be obtained from the Cluster UI directly as shown in the figure.
- When establishing the connection, you can use a Personal Access Token to authenticate to the cluster gateway. Only users who have attach permissions can access the cluster via the JDBC/ ODBC endpoint.
- In Power BI desktop you can setup the connection by choosing the ODBC data source in the “Get Data” option.



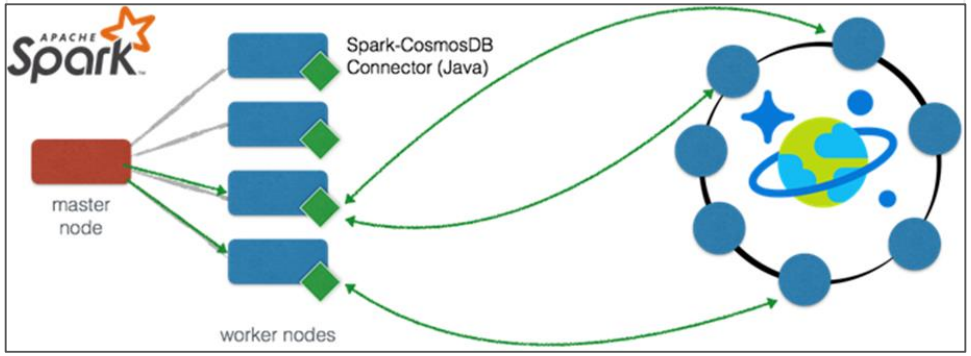
COSMOS DB INTEGRATION

The Spark connector enables real-time analytics over globally distributed data in Azure Cosmos DB

Azure Cosmos DB is Microsoft's globally distributed, multi-model database service for mission-critical applications

- With Spark connector for Azure Cosmos DB, Apache Spark can now interact with all Azure Cosmos DB data models: *Documents, Tables, and Graphs*.
  - efficiently exploits the native Azure Cosmos DB managed indexes and enables updateable columns when performing analytics.
  - utilizes push-down predicate filtering against fast-changing globally-distributed data
- Some use-cases for Azure Cosmos DB + Spark include:
  - Streaming Extract, Transformation, and Loading of data (ETL)
  - Data enrichment
  - Trigger event detection
  - Complex session analysis and personalization
  - Visual data exploration and interactive analysis
  - Notebook experience for data exploration, information sharing, and collaboration

The connector uses the Azure DocumentDB Java SDK and moves data directly between Spark worker nodes and Cosmos DB data nodes



AZURE BLOB STORAGE INTEGRATION

Data can be read from Azure Blob Storage using the Hadoop FileSystem interface. Data can be read from public storage accounts without any additional settings. To read data from a private storage account, you need to set an account key or a Shared Access Signature (SAS) in your notebook

Setting up an account key

```
spark.conf.set ( "fs.azure.account.key.{Your Storage Account Name}.blob.core.windows.net", "{Your Storage Account Access Key}")
```

Setting up a SAS for a given container:

```
spark.conf.set( "fs.azure.sas.{Your Container Name}.{Your Storage Account Name}.blob.core.windows.net", "{Your SAS For The Given Container}")
```

Once an account key or a SAS is setup, you can use standard Spark and Databricks APIs to read from the storage account:

```
val df = spark.read.parquet("wasbs://{Your Container Name}@m{Your Storage Account  name}.blob.core.windows.net/{Your Directory Name}")
dbutils.fs.ls("wasbs://{Your ntainer Name}@{Your Storage Account Name}.blob.core.windows.net/{Your Directory Name}")
```

AZURE DATA LAKE INTEGRATION

To read from your Data Lake Store account, you can configure Spark to use service credentials with the following snippet in your notebook

```
spark.conf.set("dfs.adls.oauth2.access.token.provider.type", "ClientCredential")
spark.conf.set("dfs.adls.oauth2.client.id", "{YOUR SERVICE CLIENT ID}")
spark.conf.set("dfs.adls.oauth2.credential", "{YOUR SERVICE CREDENTIALS}")
spark.conf.set("dfs.adls.oauth2.refresh.url", "https://login.windows.net/{YOUR DIRECTORY ID}/oauth2/token")
```

After providing credentials, you can read from Data Lake Store using standard APIs:

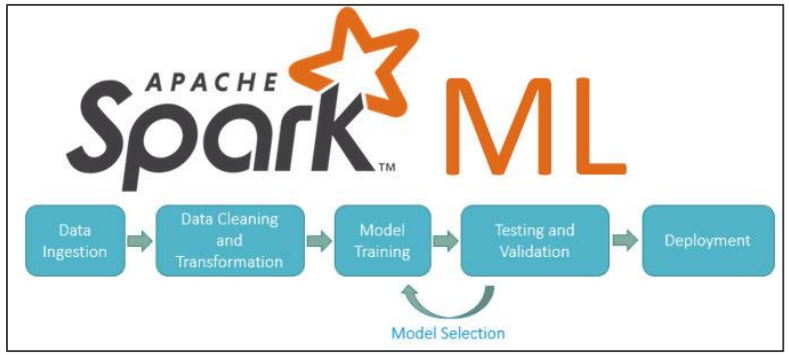
```
val df = spark.read.parquet("adl://{YOUR DATA LAKE STORE ACCOUNT NAME}.azuredatalakestore.net/{YOUR DIRECTORY NAME}")
dbutils.fs.list("adl://{YOUR DATA LAKE STORE ACCOUNT NAME}.azuredatalakestore.net/{YOUR DIRECTORY NAME}")
```

Machine Learning and Deep Learning

SPARK MACHINE LEARNING(ML) OVERVIEW

Enables Parallel, Distributed ML for large datasets on Spark Clusters

- Offers a set of parallelized machine learning algorithms (see next slide)
- Supports Model Selection (hyperparameter tuning) using Cross Validation and Train-Validation Split.
- Supports Java, Scala or Python apps using DataFrame-based API (as of Spark 2.0). Benefits include:
  - An uniform API across ML algorithms and across multiple languages
  - Facilitates ML pipelines (enables combining multiple algorithms into a single pipeline).
  - Optimizations through Tungsten and Catalyst
- Spark MLlib comes pre-installed on Azure Databricks
- 3rd Party libraries supported include: H20 Sparkling Water, SciKit-learn and XGBoost

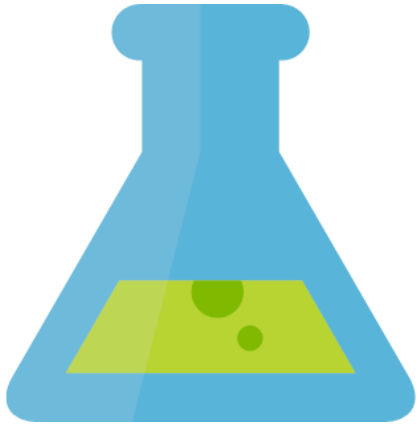


MMLSPARK

Microsoft Machine Learning Library for Apache Spark (MMLSpark) lets you easily create scalable machine learning models for large datasets.

It includes integration of SparkML pipelines with the Microsoft Cognitive Toolkit and OpenCV, enabling you to:

- Ingress and pre-process image data
- Featurize images and text using pre-trained deep learning models
- Train and score classification and regression models using implicit featurization



SPARK ML ALGORITHMS

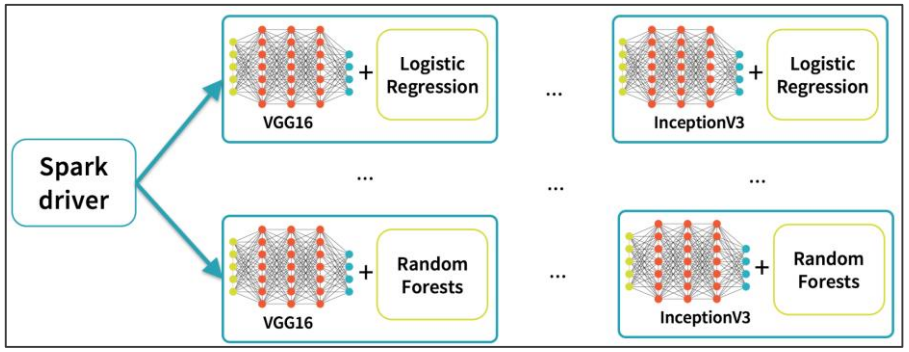
Spark ML Algorithms

Classification and Regression	• Linear Models (SVMs, logistic regression, linear regression) • Naïve Bayes • Decision Trees • Ensembles of trees (Random Forest, Gradient-Boosted Trees) • Isotonic regression
Clustering	• k-means and streaming k-means • Gaussian mixture • Power iteration clustering (PIC) • Latent Dirichlet allocation (LDA)
Collaborative Filtering	• Alternating least squares (ALS)
Dimensionality Reduction	• SVD • PCA
Frequent Pattern Mining	• FP-growth • Association rules
Basic Statistics	• Summary statistics • Correlations • Stratified sampling • Hypothesis testing • Random data generation

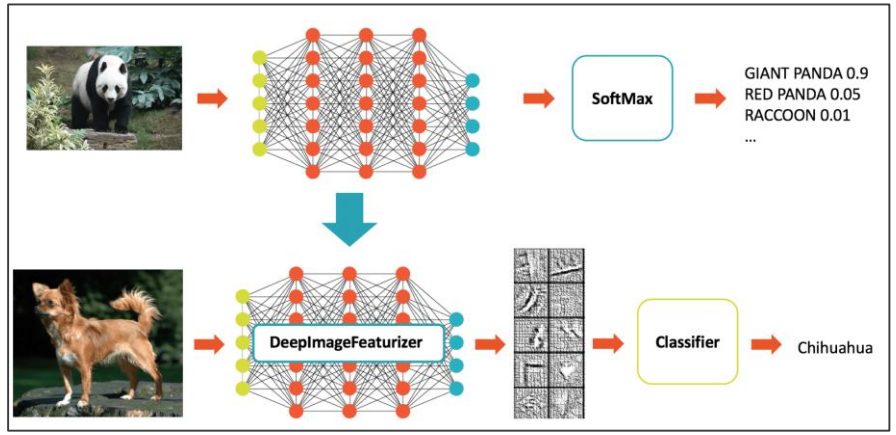
DEEP LEARNING

Azure Databricks supports and integrates with a number of Deep Learning libraries and frameworks to make it easy to build and deploy Deep Learning applications

- Supports Deep Learning Libraries/frameworks including:
  - Microsoft Cognitive Toolkit (CNTK).
    - Article explains how to install CNTK on Azure Databricks.
  - TensorFlowOnSpark
  - BigDL
- Offers Spark Deep Learning Pipelines, a suite of tools for working with and processing images using deep learning using transfer learning. It includes high-level APIs for common aspects of deep learning so they can be done efficiently in a few lines of code:
  - Image loading
  - Applying pre-trained models as transformers in a Spark ML pipeline
  - Transfer learning
  - Distributed hyperparameter tuning
  - Deploying models in DataFrames and SQL



Distributed Hyperparameter Tuning

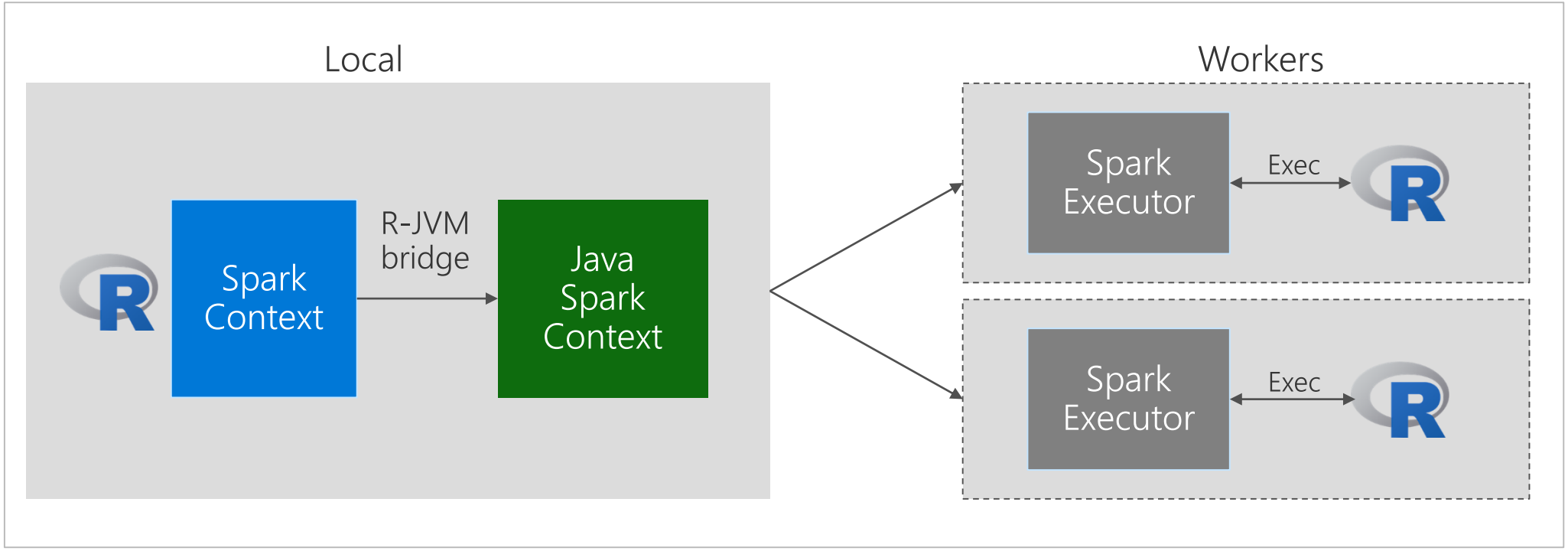


Transfer Learning

SPARKR OVERVIEW

An R package that provides a light-weight frontend to use Apache Spark from R

- Provides a distributed DataFrame implementation that supports operations like selection, filtering, aggregation etc (similar to R data frames, dplyr)
- Supports distributed machine learning using Spark MLlib.
- R programs can connect to a Spark cluster from RStudio, R shell, Rscript or other R IDEs.

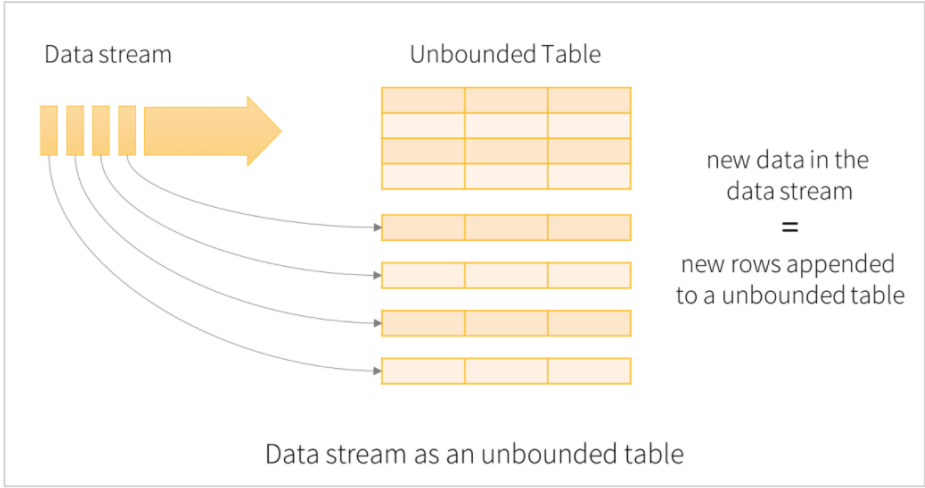


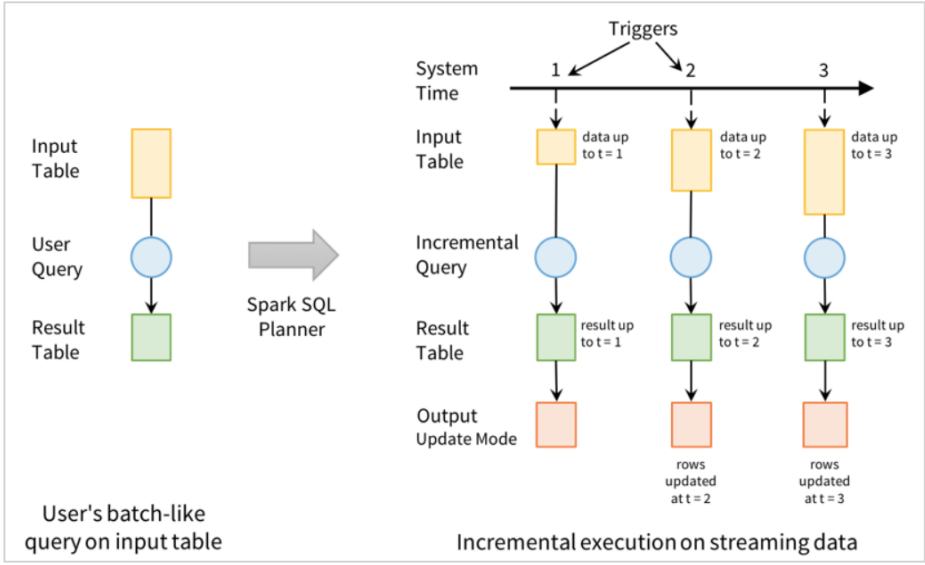
Stream Analytics & Graph Processing

SPARK STRUCTURED STREAMING OVERVIEW

A unified system for end-to-end fault-tolerant, exactly-once stateful stream processing

- Unifies streaming, interactive and batch queries—a single API for both static bounded data and streaming unbounded data.
- Runs on Spark SQL. Uses the Spark SQL Dataset/DataFrame API used for batch processing of static data.
- Runs incrementally and continuously and updates the results as data streams in.
- Supports app development in Scala, Java, Python and R.
- Supports streaming aggregations, event-time windows, windowed grouped aggregation, stream-to-batch joins.
- Features streaming deduplication, multiple output modes and APIs for managing/monitoring streaming queries.
- Built-in sources: Kafka, File source (json, csv, text, parquet)

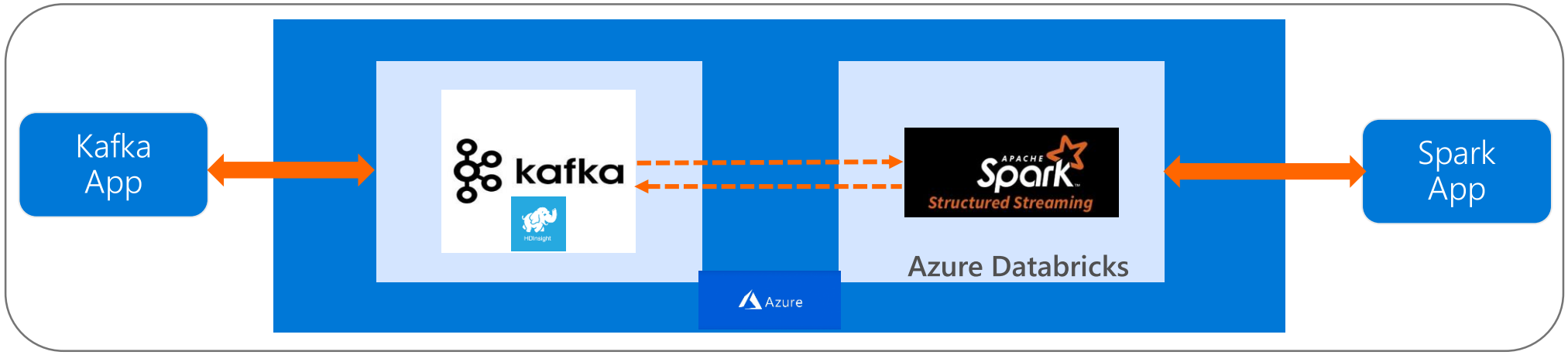




APACHE KAFKA FOR HDINSIGHT INTEGRATION

Azure Databricks Structured Streaming integrates with Apache Kafka for HDInsight

- Apache Kafka for Azure HDInsight is an enterprise grade streaming ingestion service running in Azure.
- Azure Databricks Structured Streaming applications can use Apache Kafka for HDInsight as a data source or sink.
- No additional software (gateways or connectors) are required.
- Setup: Apache Kafka on HDInsight does not provide access to the Kafka brokers over the public internet. So the Kafka clusters and the Azure Databricks cluster must be located in the same Azure Virtual Network.



Note: Azure Databricks Structured Streaming integration with **Azure Event Hubs** is forthcoming

SPARK GRAPHX OVERVIEW

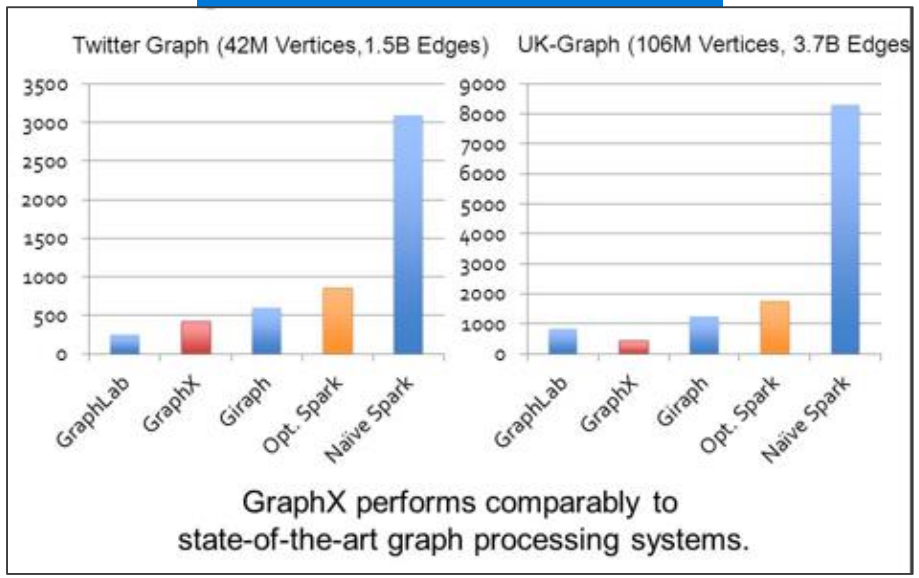
A set of APIs for graph and graph-parallel computation.

- Unifies ETL, exploratory analysis, and iterative graph computation within a single system.
- Developers can:
  - view the same data as both graphs and collections,
  - transform and join graphs with RDDs, and
  - write custom iterative graph algorithms using the Pregel API.
- Currently only supports using the Scala and RDD APIs.

Algorithms

- PageRank
- Connected components
- Label propagation
- SVD++
- Strongly connected components
- Triangle count

PageRank Benchmark

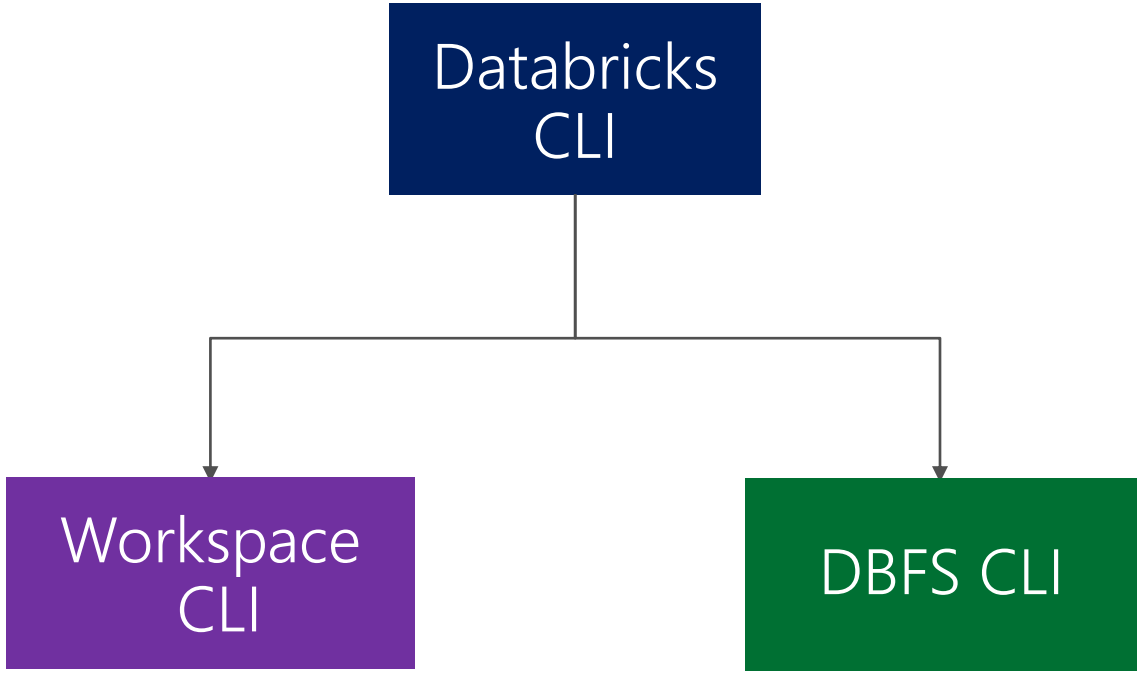


Source: AMPLab

CLI and REST APIs

DATABRICKS CLI

An easy to use interface built on top of the Databricks REST API



Currently, the CLI fully implements the DBFS API and the Workspace API

DATABRICKS WORKSPACE CLI

Databricks Workspace CLI Commands

delete	Deletes objects from the Databricks...
export	Exports a file from the Databricks workspace...
export_dir	Recursively exports a directory from the...
import	Imports a file from local to the Databricks...
import-dir	Recursively imports a directory from local to...
list / ls	List objects in the Databricks Workspace
mkdirs	Make directories in the Databricks Workspace
rm	Deletes objects from the Databricks...

Workspace CLI Example

```
$ Databricks workspace ls /Users/example@Databricks.com/example -l
NOTEBOOK  a  PYTHON
NOTEBOOK  b  SCALA
NOTEBOOK  c  SQL
NOTEBOOK  d  R
DIRECTORY  e
```

DBFS CLI

Leverages the DBFS API to provide an easy Command Line Interface to DBFS

DBFS CLI Commands:

cp	Copy files to and from DBFS.
ls	List files in DBFS.
mkdirs	Make directories in DBFS.
mv	Moves a file between two DBFS paths.
rm	Remove files from dbfs.

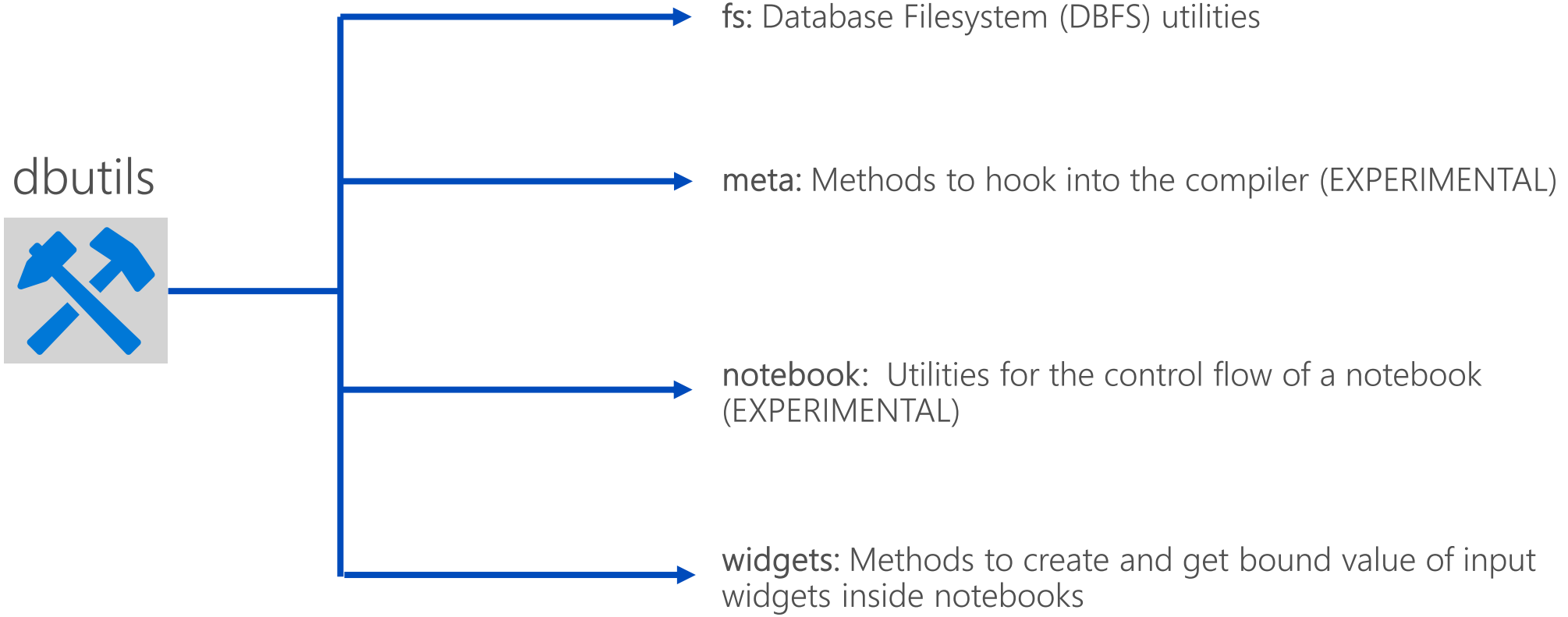
DBFS CLI examples

```
# List files in DBFS
dbfs ls

# Put local file ./foo.txt to dbfs:/foo.txt
dbfs cp ./foo.txt dbfs:/foo.txt

# Get dbfs:/foo.txt and save to local file ./foo.txt
dbfs cp dbfs:/foo.txt ./foo.txt

# Recursively put local dir ./foo to dbfs:/foo
dbfs cp -r ./foo dbfs:/foo
```

Note: All dbfs paths should be prefixed with dbfs://

DATABRICKS UTILITIES (DBUTILS)

Set of tools that make it easy to perform combinations of tasks



DATABRICKS REST API

Databricks REST API

Cluster API	Create/edit/delete clusters
DBFS API	Interact with the Databricks File System
Groups API	Manage groups of users
Instance Profile API	Allows admins to add, list, and remove instances profiles that users can launch clusters with
Job API	Create/edit/delete jobs
Library API	Create/edit/delete libraries
Workspace API	List/import/export/delete notebooks/folders

DATABRICKS API - AUTHENTICATION

Personal access tokens or passwords can be used to authenticate and access Databricks REST APIs

- Tokens can be *generated* and *revoked* from the Databricks Portal Token Management Page.
- Tokens have an expiration time
- In the REST call, the token is placed in the header as

-H "Authorization: Bearer TOKEN_VALUE"