Azure Databricks

Microsoft Data & AI MU - ATCI

Agenda

Azure Databricks – Introduction

Azure Databricks – Benefits

Azure Databricks - Core Artifacts

Azure Databricks Delta

avanade

Azure Databricks - Introduction

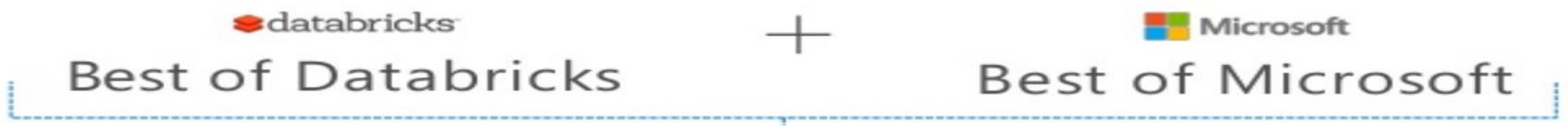
Apache Spark is an open source standard for advanced analytics, machine learning, AI and Big data

Databricks is extension of power of Spark with additional features

Azure Databricks Microsoft aims to help businesses work with data in a faster, easier and more collaborative way.

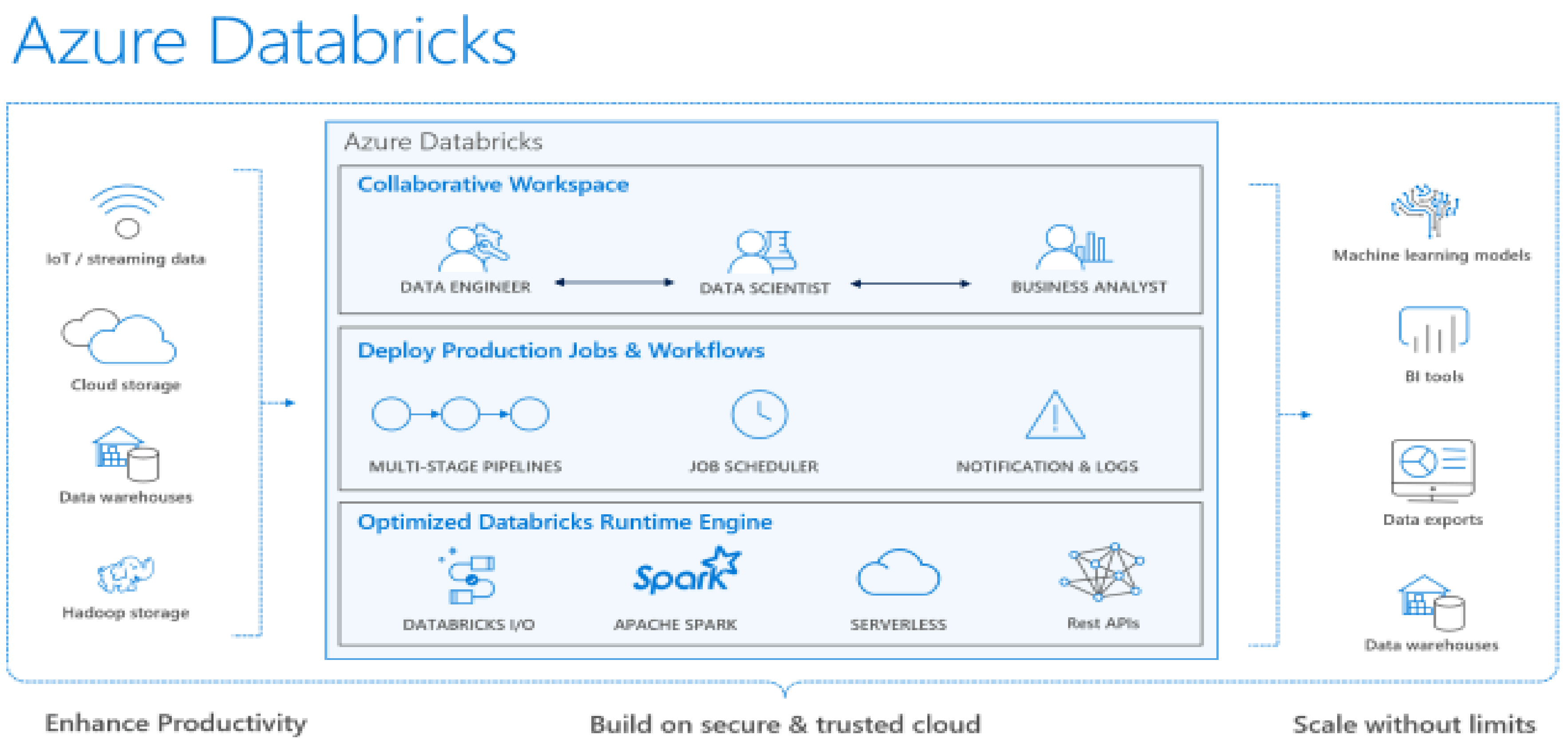
Using Databricks customers can accelerate innovation with one click setup, streamlined workflows and an interactive workspace that enables easy collaboration between data scientists, engineers and business analysts.

Azure Databricks - Introduction



Azure Databricks provides a unified platform to manage clusters for various use cases such as running production ETL pipelines, streaming analytics and ad-hoc analytics.

Azure Databricks - Introduction



Azure Databricks – Key benefits

Enhance productivity

❖Get started quickly by launching your spark environment with one click.

❖Integrate effortlessly with a wide variety of data stores and services such as Azure Synapse, Azure Cosmos DB, Azure Data Lake Store, Azure Blob storage, and Azure Event Hub. Add artificial intelligence (AI) capabilities instantly and share insights through rich integration with Power BI.

❖Use Databricks notebook to unify the process and instantly deploy into production. Improves collaboration amongst various teams through a unified workspace. Notebooks supports for multiple programming languages like R, Python, Scala and SQL.

Azure Databricks – Key benefits

Scale without limits

❖Operate at massive scale without limits globally. Databricks server less and highly elastic cloud service is designed to remove operational complexity.

❖Minimize costs by Auto scaling and Auto termination of Spark Clusters.

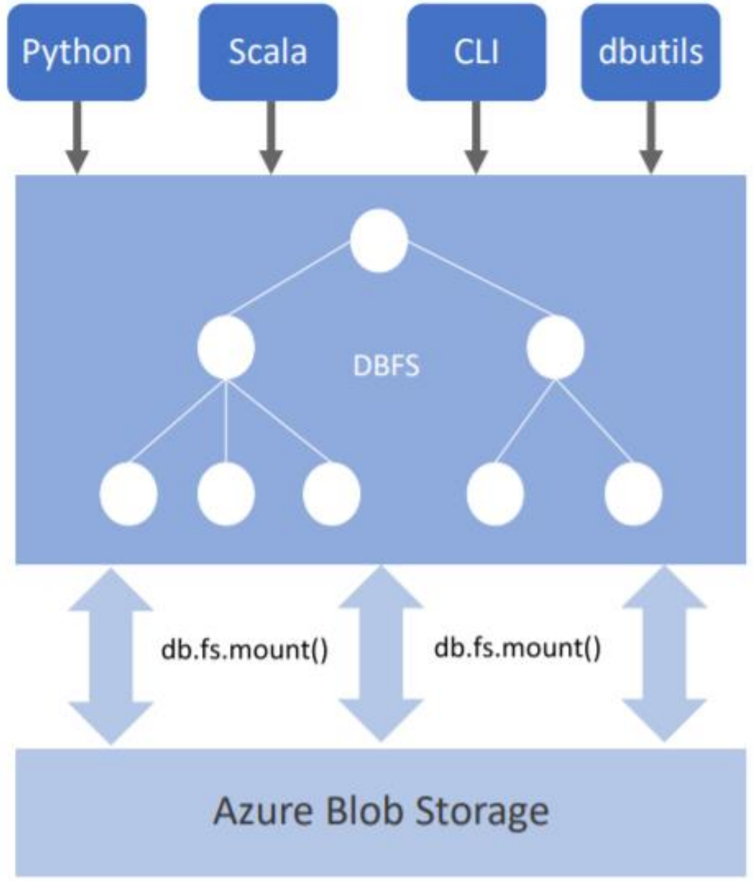
❖Accelerate data processing with the fastest Spark engine.

❖High-speed connectors to Azure storage services such as Azure Blob Store and Azure Data Lake, developed together with the Microsoft teams behind these services.

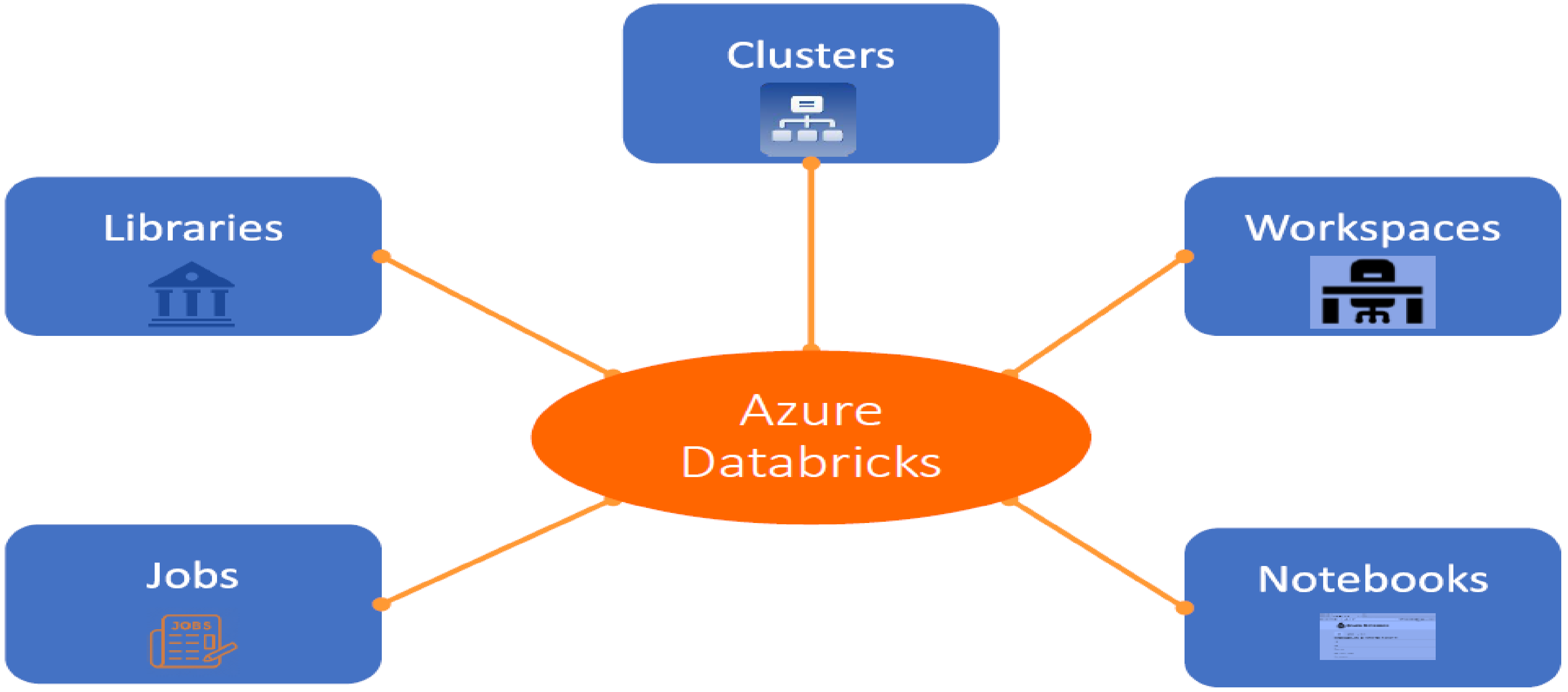
DATABRICKS FILE SYSTEM (DBFS)

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

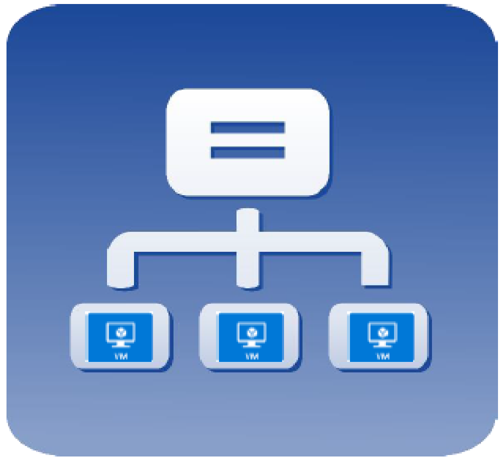


Azure Databricks – Core Artifacts



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



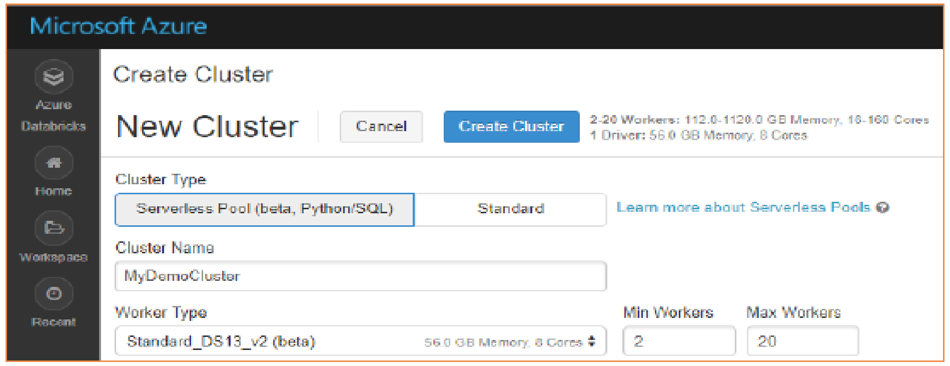
avanade

Azure Databricks – Clusters

SERVERLESS POOL (BETA)

A self-managed pool of cloud resources, auto-configured for interactive Spark workloads

- You specify only the **minimum** and **maximum** number of nodes in the cluster—Azure Databricks provisions and adjusts the compute and local storage based on your usage.
- Limitation: Currently works only for SQL and Python.



Create Cluster

New Cluster

2-20 Workers: 112.0-1120.0 GB Memory, 16-160 Cores
1 Driver: 56.0 GB Memory, 8 Cores

Cluster Type: [✓ Serverless Pool (beta, Python/SQL)] Standard

Cluster Name: MyDemoCluster

Worker Type: Standard_DS13_v2 (beta) 56.0 GB Memory, 8 Cores

Min Workers: 2

Max Workers: 20

Benefits of Serverless Pool	
Auto-Configuration	▪ Databricks chooses the best configuration for Spark to get the best performance ▪ Users don't need to worry about providing any of the Databricks runtime version or Spark configuration. ▪ Databricks also chooses the best cluster parameters to save cost on infrastructure
Elasticity	▪ Automatically scales the compute and local storage, independently, based on usage
Fine grained Sharing	▪ Offers maximum resource utilization and minimum query latencies • *Preemption:* Databricks proactively preempts Spark tasks from over-committed users to ensure all users get their fair share of cluster time and their jobs complete in a timely manner even when contending with dozens of other users. Uses the "Task Preemption for High Concurrency" feature of Spark in Databricks. • *Fault isolation:* Databricks sandboxes the environments belonging to different notebooks from one another.

Azure Databricks – *Jobs*

JOBS

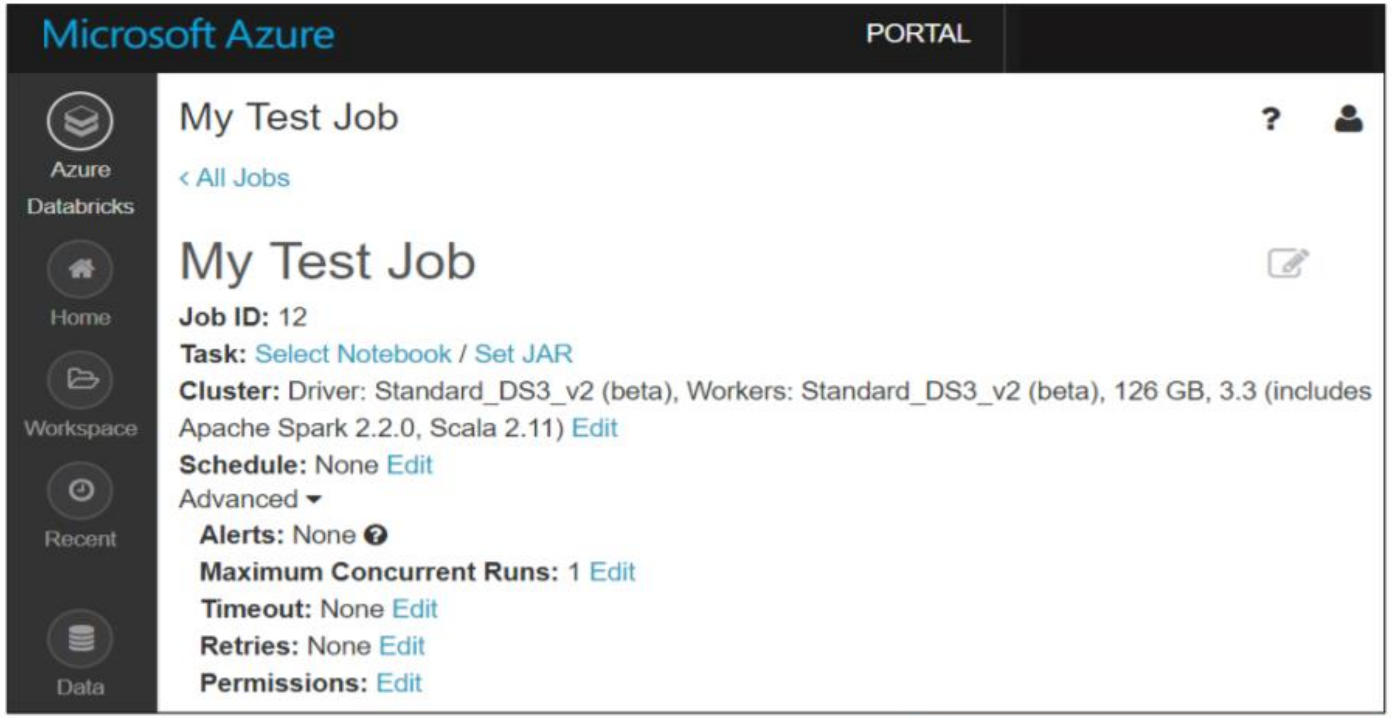
Jobs are the mechanism to submit Spark application code for execution on the Databricks clusters

- 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.



My Test Job

‹ All Jobs

My Test Job

Job ID: 12

Task: Select Notebook / Set JAR

Cluster: Driver: Standard_DS3_v2 (beta), Workers: Standard_DS3_v2 (beta), 126 GB, 3.3 (includes Apache Spark 2.2.0, Scala 2.11) Edit

Schedule: None Edit

Advanced

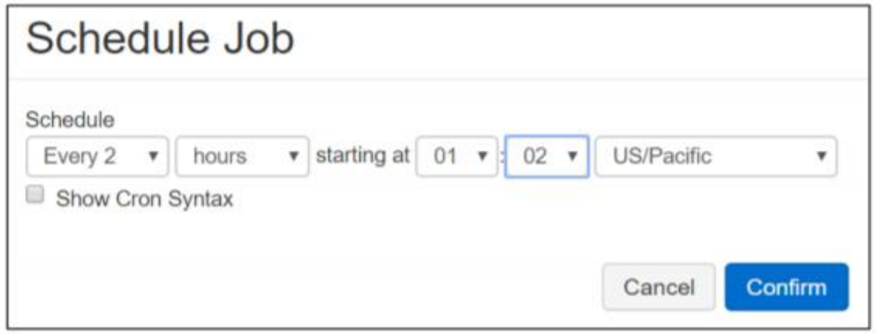
Alerts: None

Maximum Concurrent Runs: 1 Edit

Timeout: None Edit

Retries: None Edit

Permissions: Edit

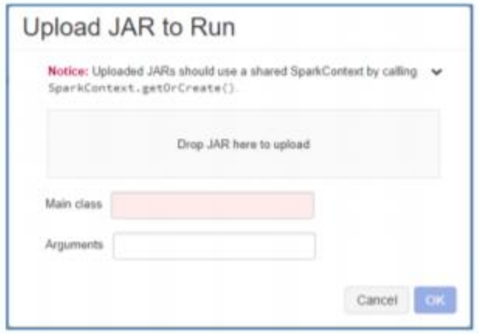


Schedule Job

Schedule

Every 2 | hours | starting at 01 : 02 | US/Pacific

Show Cron Syntax



Upload JAR to Run

Notice: Uploaded JARs should use a shared SparkContext by calling `SparkContext.getOrCreate()`.

Drop JAR here to upload

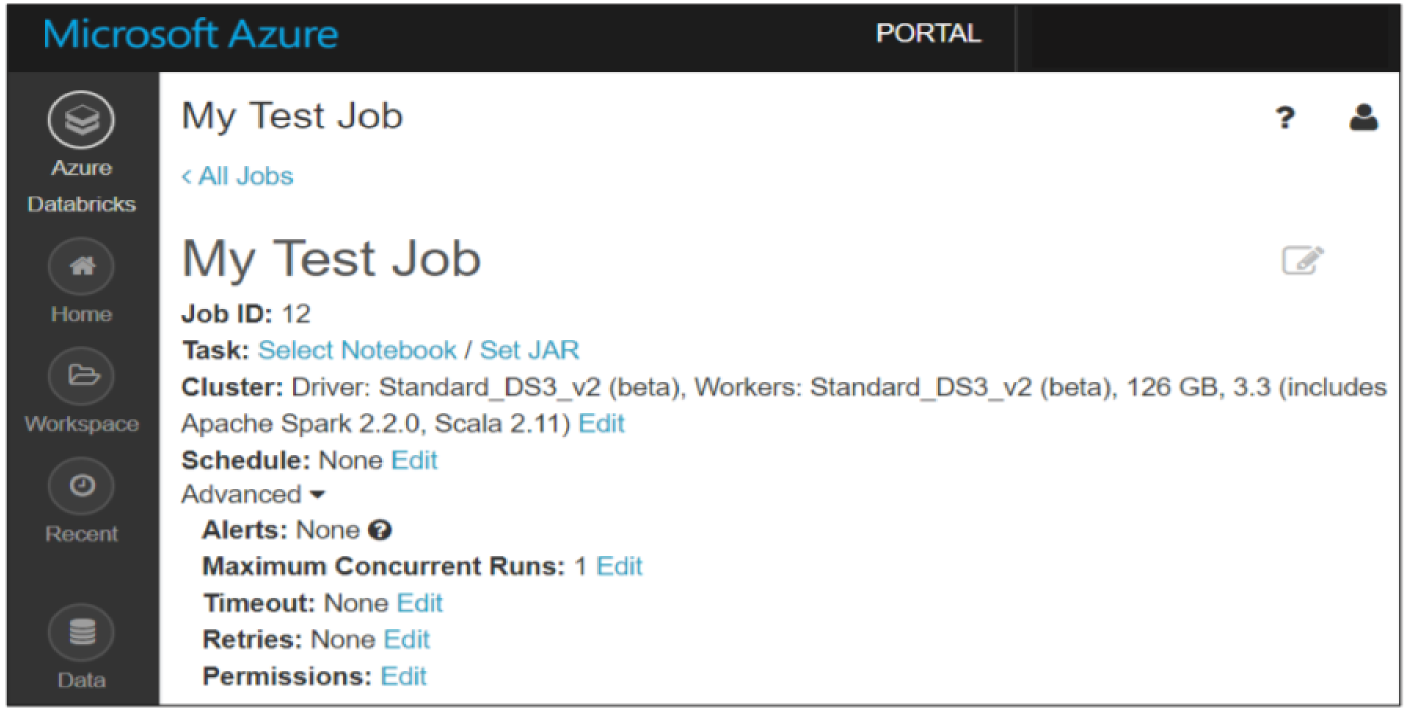
Main class

Arguments

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)

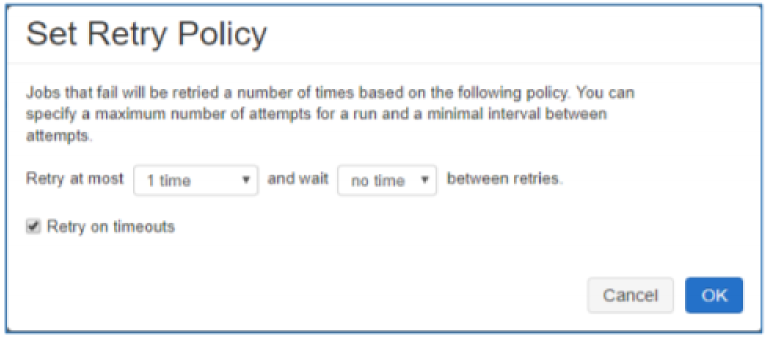


My Test Job

< All Jobs

My Test Job

Job ID: 12
Task: Select Notebook / Set JAR
Cluster: Driver: Standard_DS3_v2 (beta), Workers: Standard_DS3_v2 (beta), 126 GB, 3.3 (includes Apache Spark 2.2.0, Scala 2.11) Edit
Schedule: None Edit
Advanced
Alerts: None
Maximum Concurrent Runs: 1 Edit
Timeout: None Edit
Retries: None Edit
Permissions: Edit

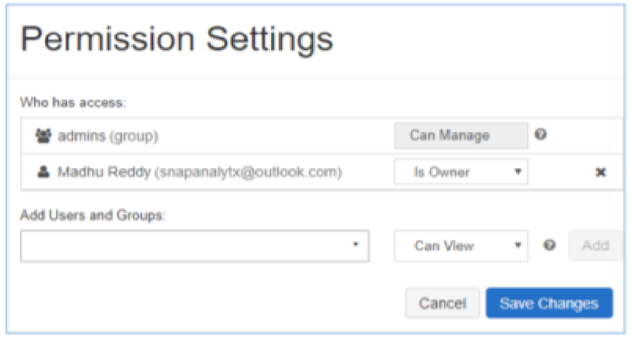


Set Retry Policy

Jobs that fail will be retried a number of times based on the following policy. You can specify a maximum number of attempts for a run and a minimal interval between attempts.

Retry at most 1 time and wait no time between retries.

[✓] Retry on timeouts



Permission Settings

Who has access:

admins (group)	Can Manage
Madhu Reddy (snapanalytx@outlook.com)	Is Owner

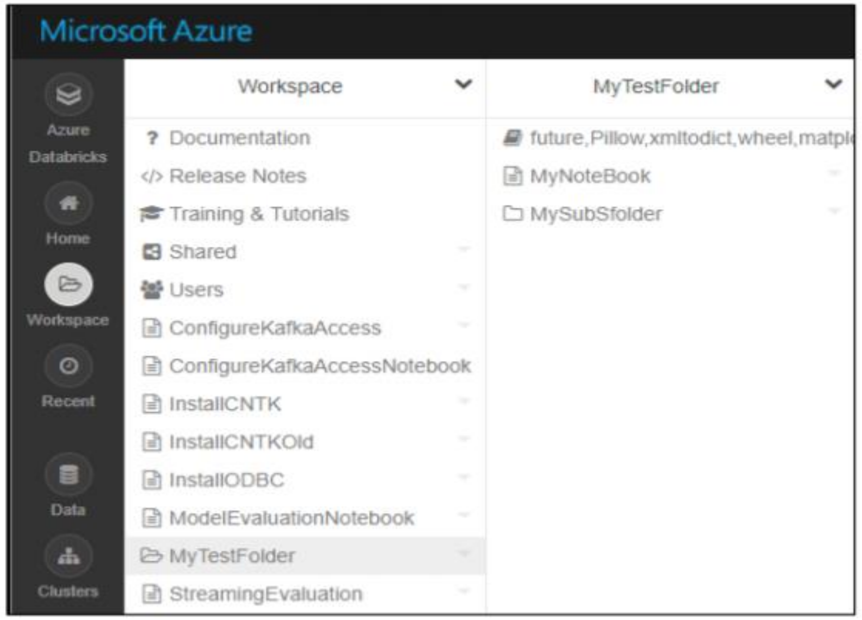
Add Users and Groups:

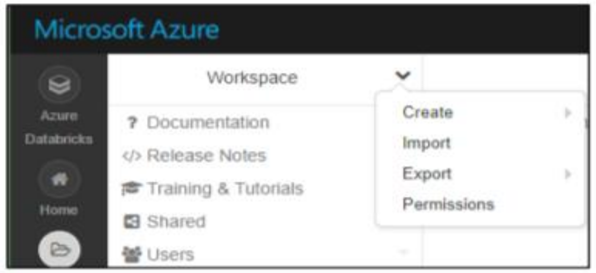
Can View

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.*



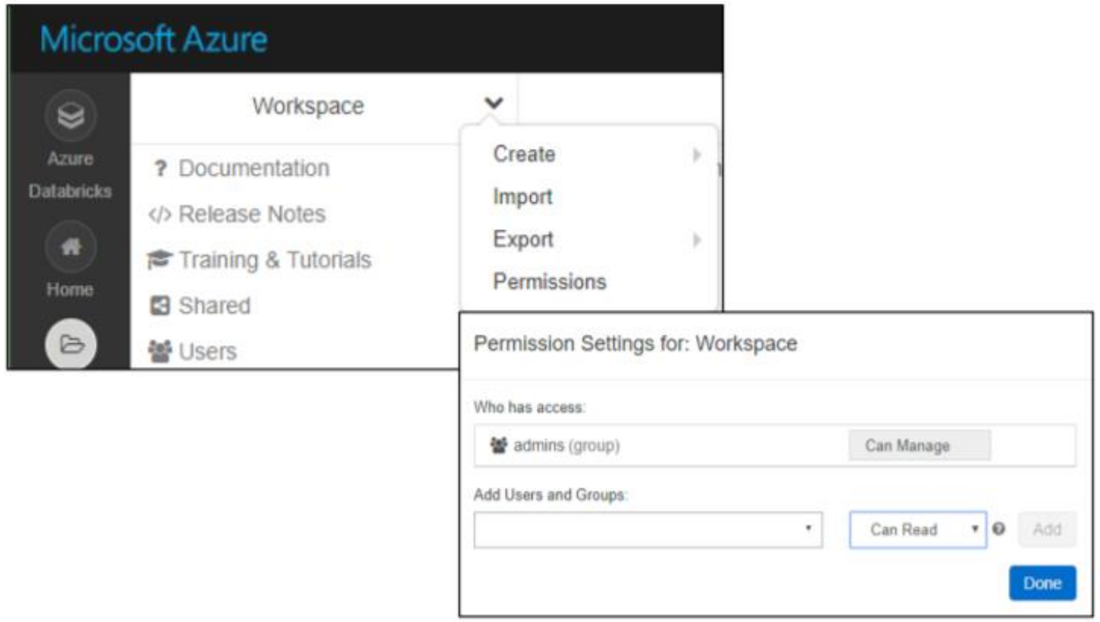


WORKSPACE OPERATIONS

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



Permission Settings for: Workspace

Who has access:

admins (group) Can Manage

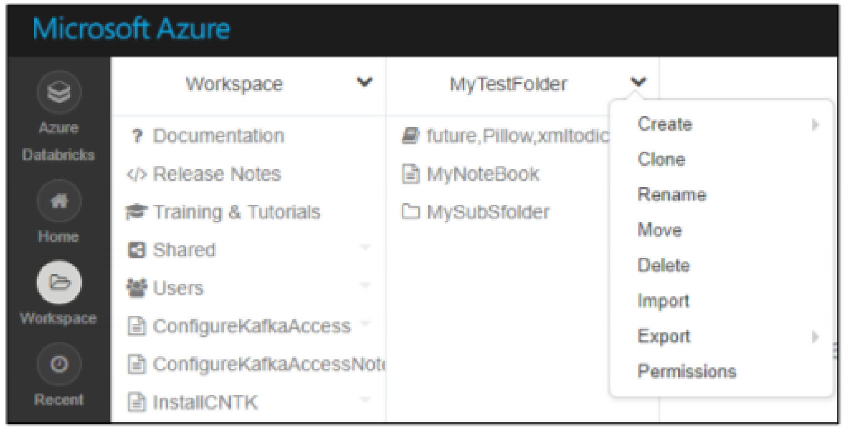
Add Users and Groups:

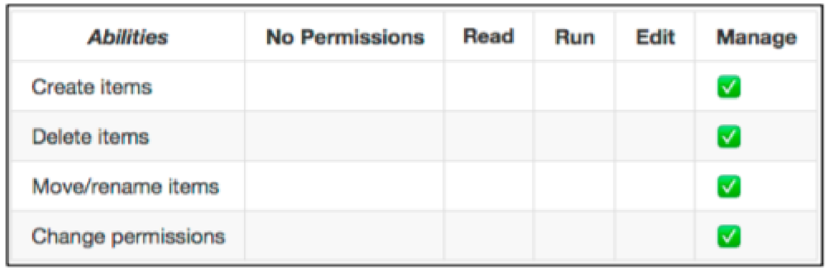
Can Read

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*





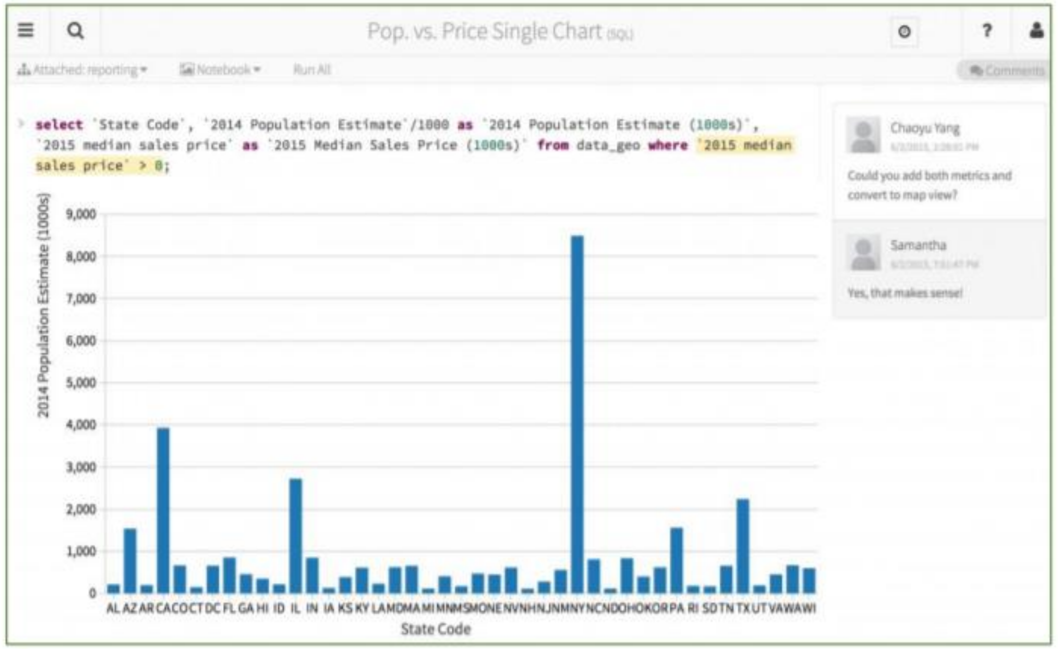
Abilities	No Permissions	Read	Run	Edit	Manage
Create items					✓
Delete items					✓
Move/rename items					✓
Change permissions					✓

Abilities associated with each permission level

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


Pop. vs. Price Single Chart (SQL)
Attached: reporting
Notebook
Run All
Comments
2014 Population Estimate (1000s)
9,000
8,000
7,000
6,000
5,000
4,000
3,000
2,000
1,000
0
AL AZ AR CA CO CT DC FL GA HI ID IL IN IA KS KY LA MD MA MI MN MS MO NE NV NH NJ NM NY NC ND OH OK OR PA RI SD TN TX UT VA WA WI
State Code

```
select `State Code`, `2014 Population Estimate`/1000 as `2014 Population Estimate (1000s)`,
`2015 median sales price` as `2015 Median Sales Price (1000s)` from data_geo where `2015 median
sales price` > 0;
```

Chaoyu Yang

Could you add both metrics and convert to map view?

Samantha

Yes, that makes sense!

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:

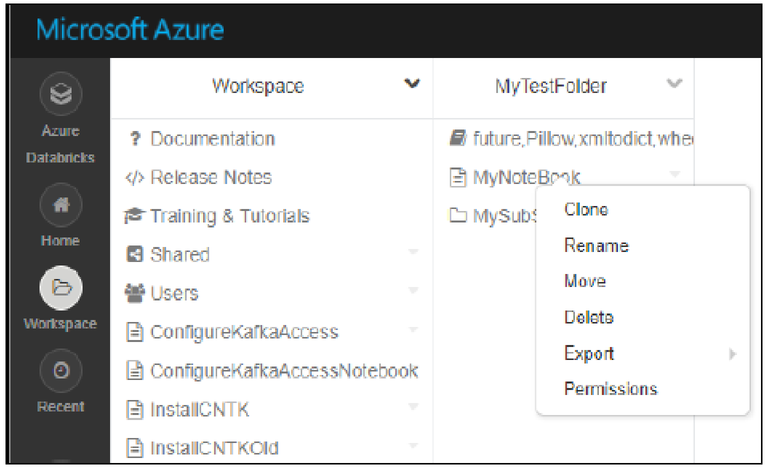
• %python	Allows you to execute python code in a notebook (even if that notebook is not python)
• %sql	Allows you to execute sql code in a notebook (even if that notebook is not sql).
• %r	Allows you to execute r code in a notebook (even if that notebook is not r).
• %scala	Allows you to execute scala 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

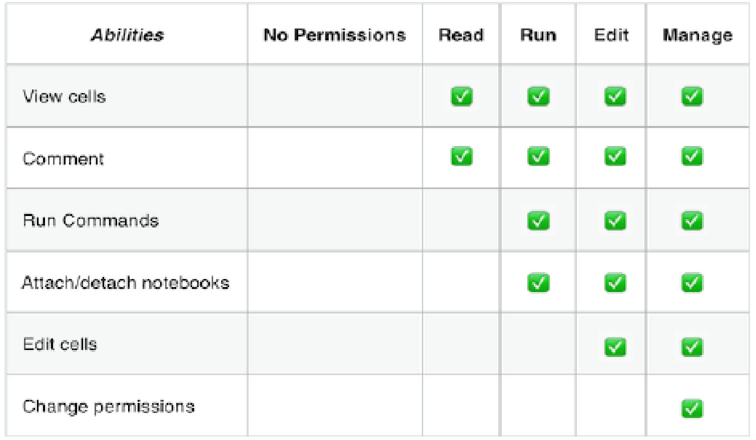
NOTEBOOK OPERATIONS AND ACCESS CONTROL

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 Permissions menu option.





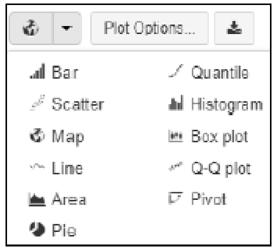
Abilities	No Permissions	Read	Run	Edit	Manage
View cells		✓	✓	✓	✓
Comment		✓	✓	✓	✓
Run Commands			✓	✓	✓
Attach/detach notebooks			✓	✓	✓
Edit cells				✓	✓
Change permissions					✓

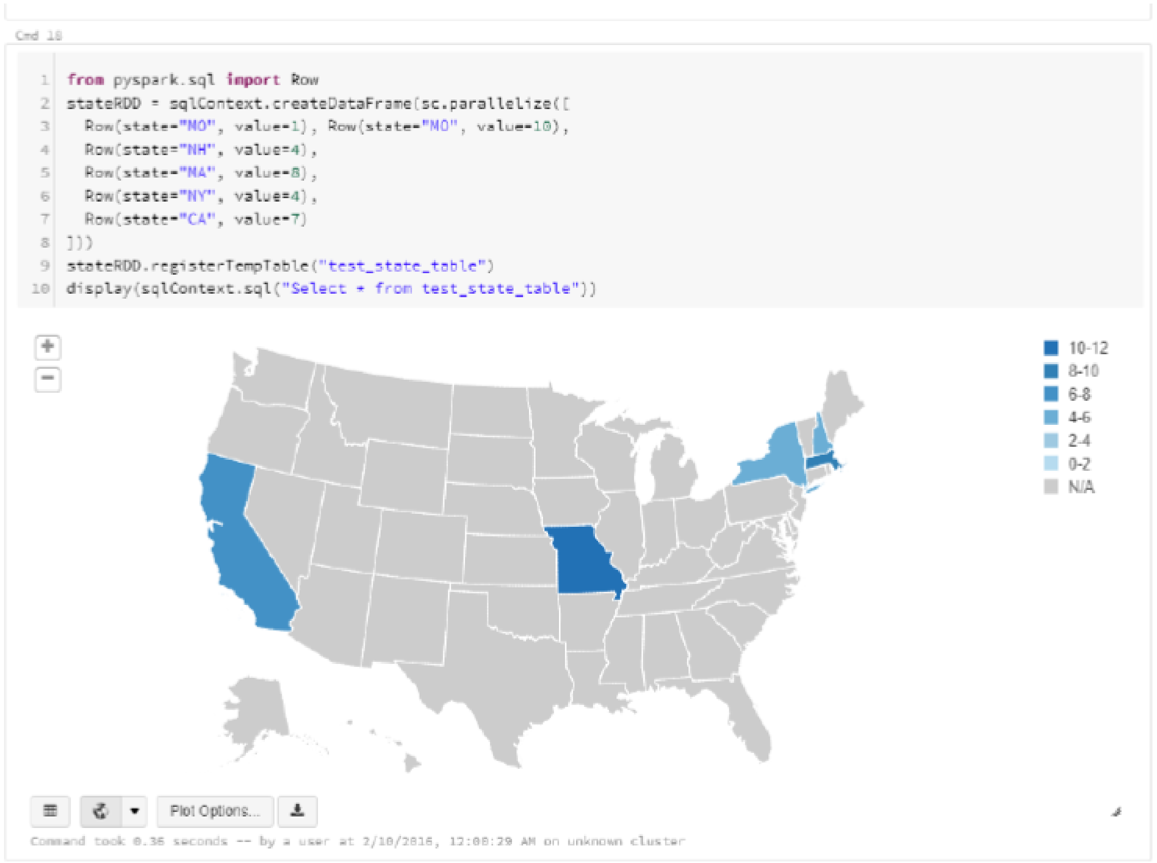
Abilities associated with each permission level

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

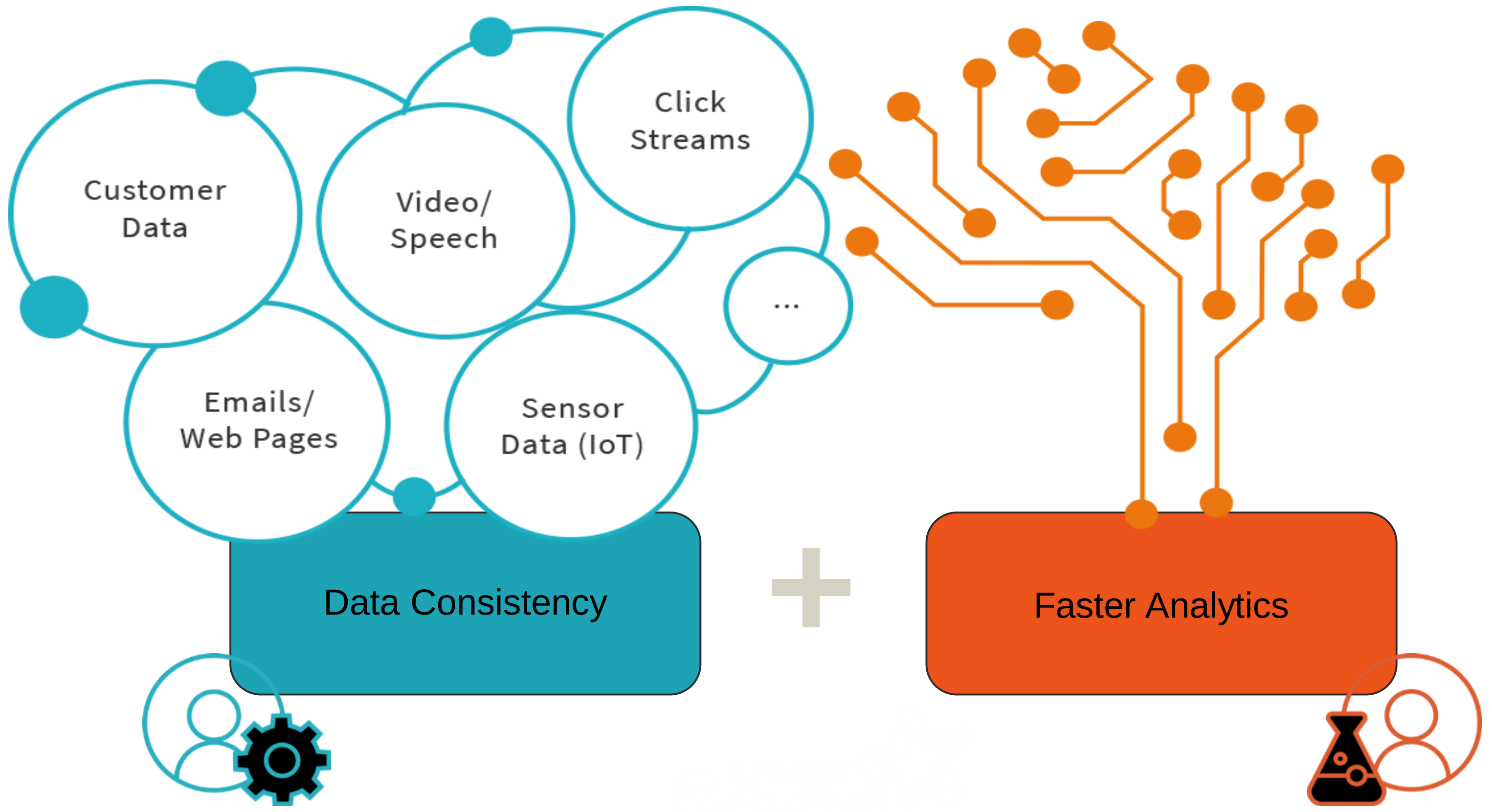



Cmd 18

```
from pyspark.sql import Row
stateRDD = sqlContext.createDataFrame(sc.parallelize([
  Row(state="MO", value=1), Row(state="MO", value=10),
  Row(state="NH", value=4),
  Row(state="MA", value=8),
  Row(state="NY", value=4),
  Row(state="CA", value=7)
]))
stateRDD.registerTempTable("test_state_table")
display(sqlContext.sql("Select * from test_state_table"))
```

```
Command took 0.36 seconds -- by a user at 2/10/2016, 12:00:29 AM on unknown cluster
```

Azure Databricks Delta



Azure Databricks - Delta

Databricks Delta delivers a powerful transactional storage layer by harnessing the power of Apache Spark and Databricks DBFS (Data bricks file system). The core abstraction of Databricks Delta is an optimized Spark table that

- ➢ Stores data as Parquet files in DBFS.
- ➢ Maintains a *transaction log* that efficiently tracks changes to the table.
- ➢ Improves performance by organizing data and creating metadata.
  - ✓ Manages optimal file sizes.
  - ✓ Stores statistics that enable data skipping.
  - ✓ Creates and maintains indexes.
- ➢ Improves data reliability by adding data management capabilities.
  - ✓ Schema enforcement ensures data is not corrupted.
  - ✓ Upserts makes fixing bad data simple.

Azure Databricks Delta Architecture

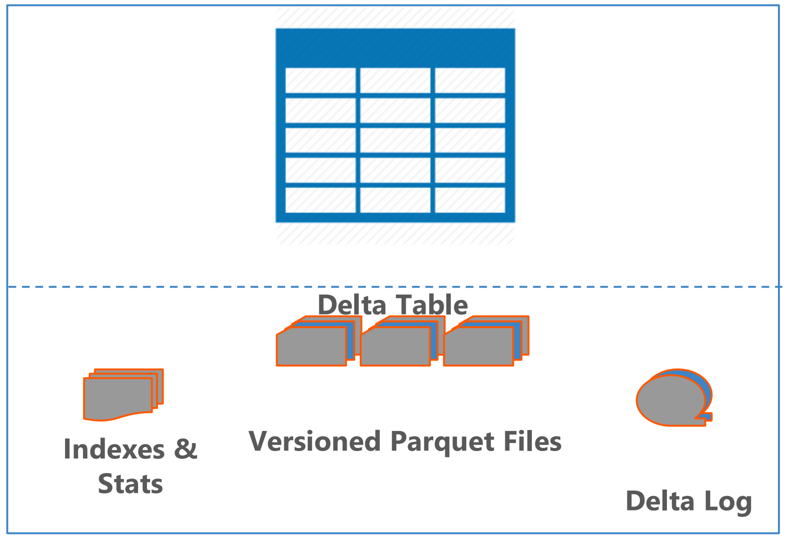
Delta Table = Parquet + Transaction Log

Linear history of atomic changes

Optimistic Concurrency Control

Log checkpoint is stored as Parquet

Delta Table



Azure Databricks Vs Spark

	APACHE Spark	databricks
DATABRICKS RUNTIME *Built on Apache Spark and optimized for performance*	✗	✓
MANAGED DELTA LAKE *Reliable and Performant Data Lakes*	✗	✓
INTEGRATED WORKSPACE *Interactive Data Science and Collaboration*	✗	✓
PRODUCTION JOBS AND WORKFLOWS *Data Pipelines and Workflow Automation*	✗	✓
ENTERPRISE SECURITY *End-to-End Data Security and Compliance* Learn More	✗	✓
INTEGRATIONS *Compatible with Common Tools in the Ecosystem*	✗	✓
EXPERT SUPPORT *Unparalled Support by the Leading Committers of Apache Spark*	✗	✓

Resources

[Azure Databricks Delta Guide]

[QuickStart notebook]

[Performance optimization]