Microsoft AI

Azure Databricks

Microsoft Customer Engineer
Maizie Ku

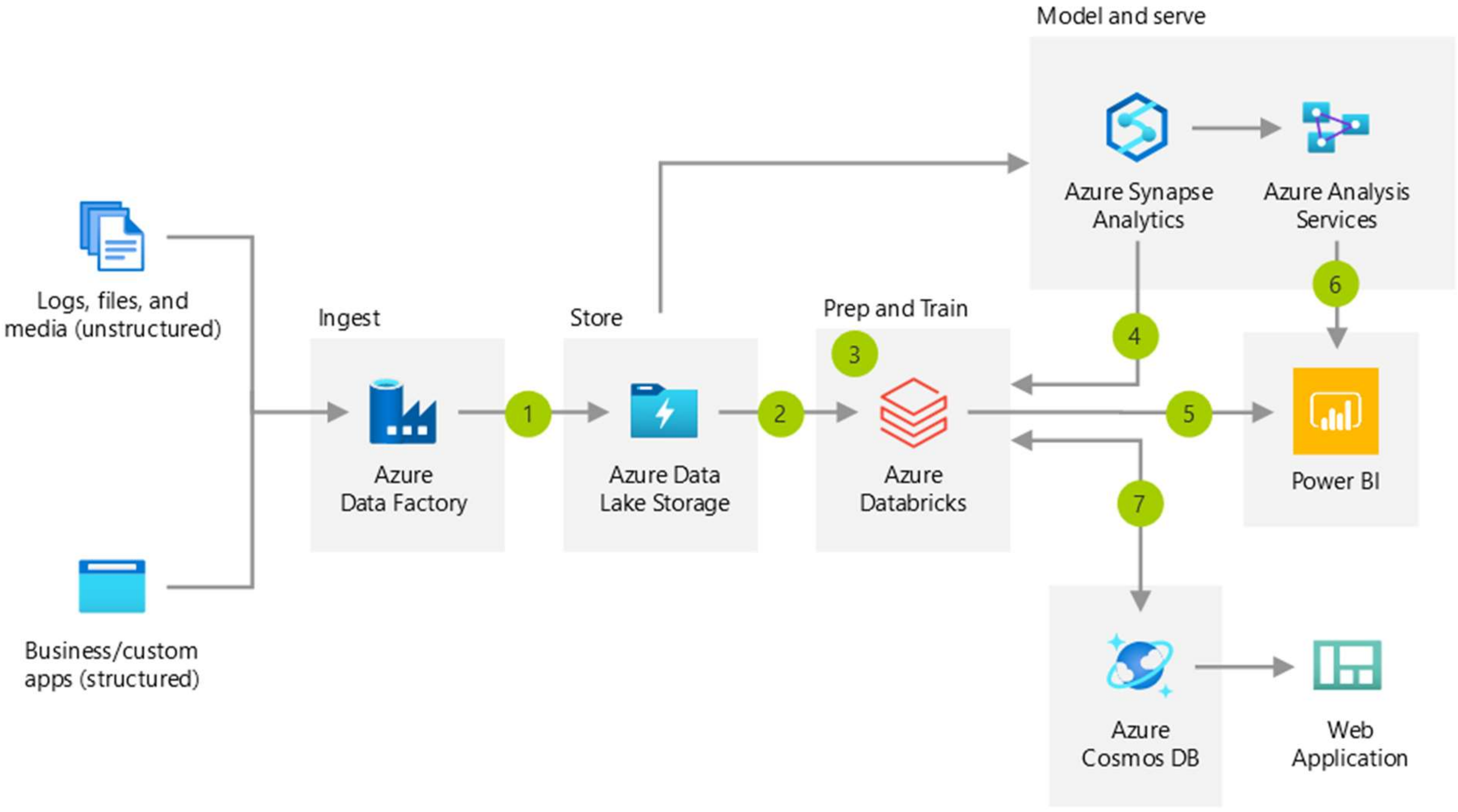
Speaker Introduction 講者介紹

Maizie Ku

Microsoft Customer Engineer

經歷 :	7+ Advanced Analytics
現任:	Microsoft Data & AI Customer Engineer
認證 :	Perform Cloud Data Science with Azure Machine Learning
	Analyzing Big Data with Microsoft R
	Power BI Microsoft Certified Professional
課程 :	10+ Azure Machine Learning Workshop
	10+ Azure Databricks Workshop
	10+ BI Lecture/Workshop

Big Data Advanced Analytics



Rooted in open source

APACHE Spark™

DELTA LAKE

mlflow™

Azure Machine Learning

Enterprise cloud service

Why Spark?

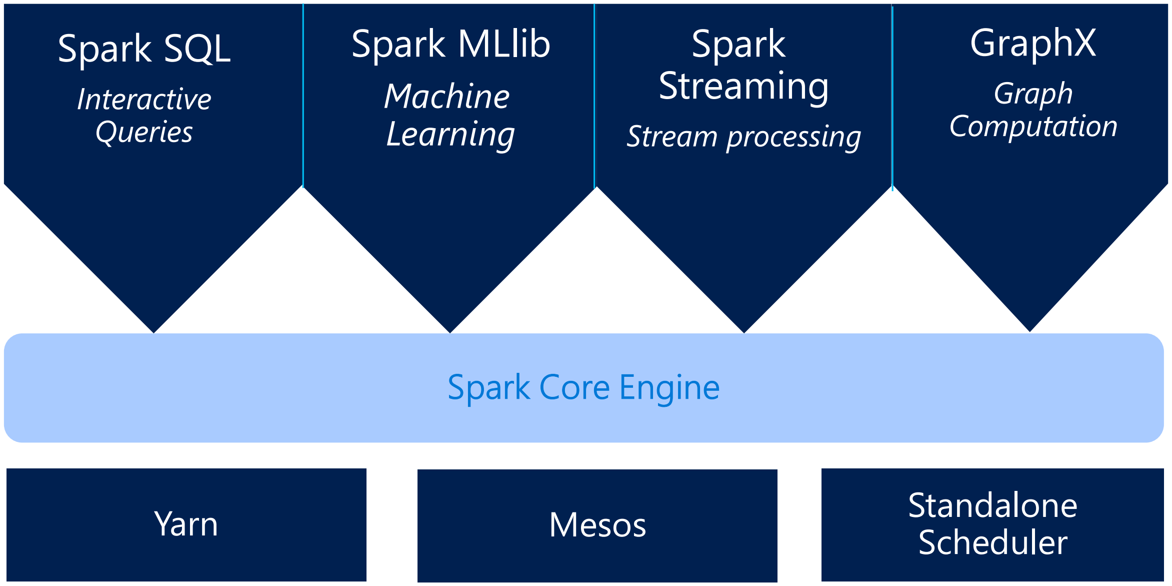
- Open-source data processing engine built around speed, ease of use, and sophisticated analytics
- In memory engine that is up to 100 times faster than Hadoop
- Largest open-source data project with 1000+ contributors
- Highly extensible with support for Scala, Java and Python alongside Spark SQL, GraphX, Streaming and Machine Learning Library (Mllib)

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 SQL
Interactive Queries

Spark MLlib
Machine Learning

Spark Streaming
Stream processing

GraphX
Graph Computation

Spark Core Engine

Yarn

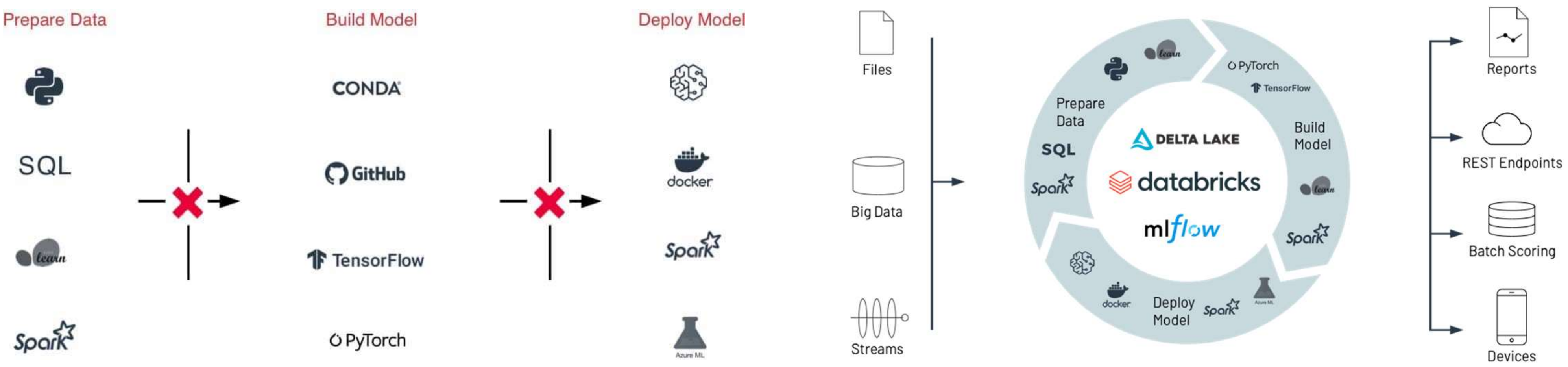
Mesos

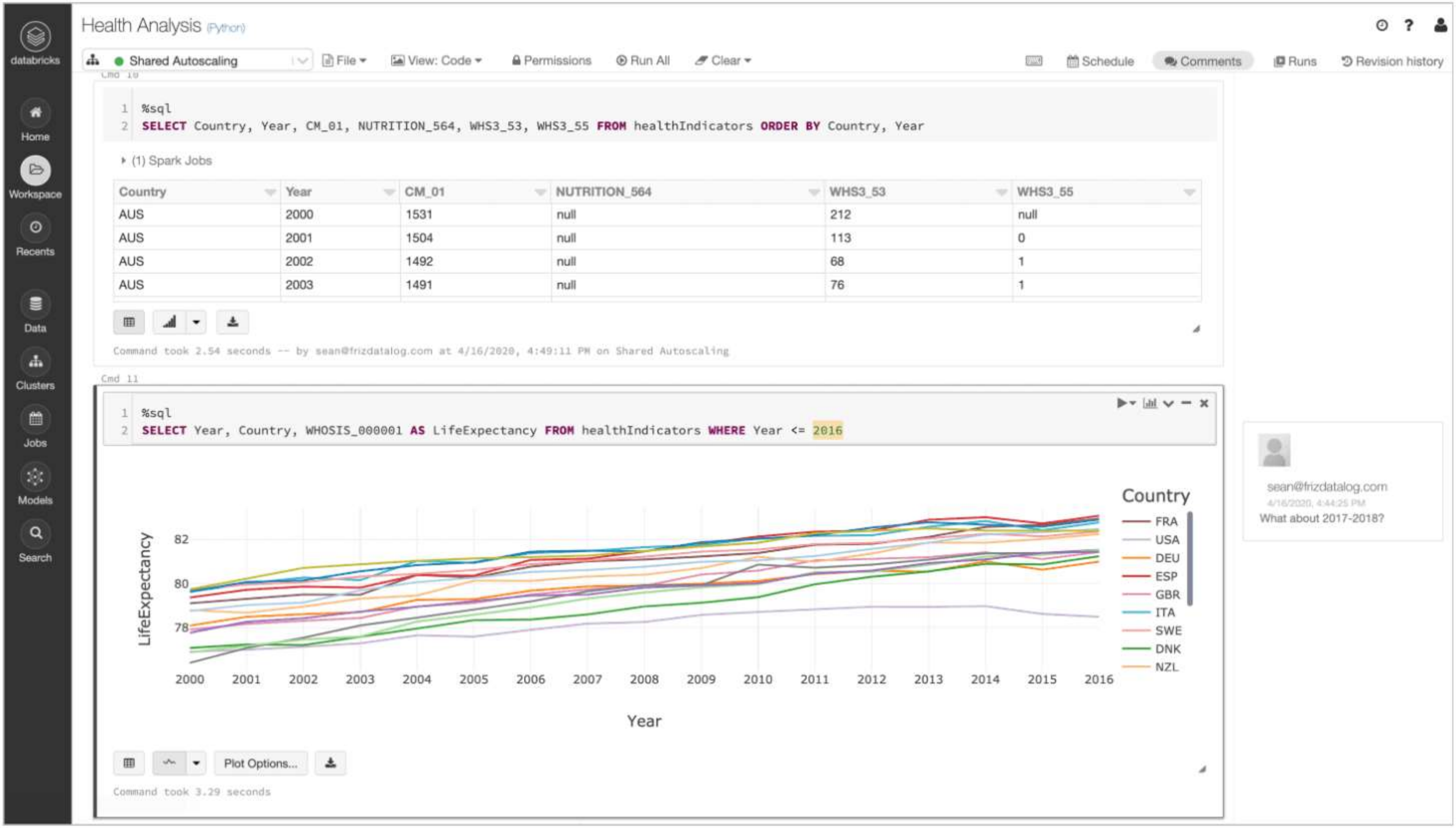
Standalone Scheduler

Databricks

- It’s a managed platform for running Apache Spark
- No cluster management
- No tedious maintenance tasks
- Point and click platform for those that prefer a user interface
- Capabilities to automate aspects of data workloads with automated jobs

Simplify the Machine Learning Lifecycle





```
%sql
SELECT Country, Year, CM_01, NUTRITION_564, WHS3_53, WHS3_55 FROM healthIndicators ORDER BY Country, Year
```

Country	Year	CM_01	NUTRITION_564	WHS3_53	WHS3_55
AUS	2000	1531	null	212	null
AUS	2001	1504	null	113	0
AUS	2002	1492	null	68	1
AUS	2003	1491	null	76	1

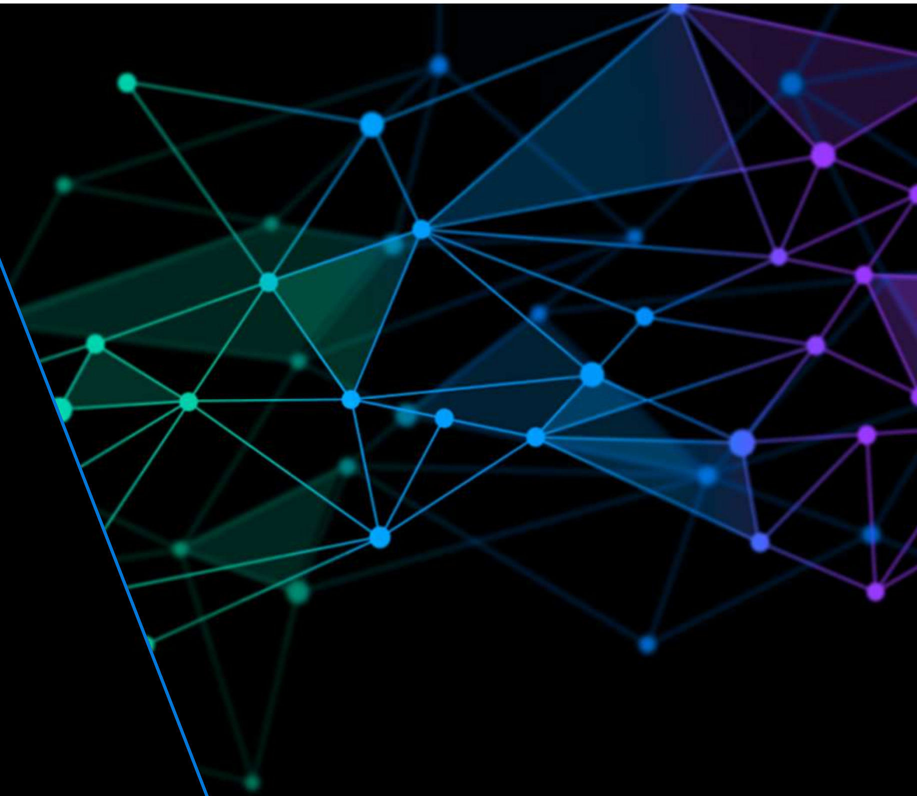
```
Command took 2.54 seconds -- by sean@frizdatalog.com at 4/16/2020, 4:49:11 PM on Shared Autoscaling
```

```
%sql
SELECT Year, Country, WHOSIS_000001 AS LifeExpectancy FROM healthIndicators WHERE Year <= 2016
```

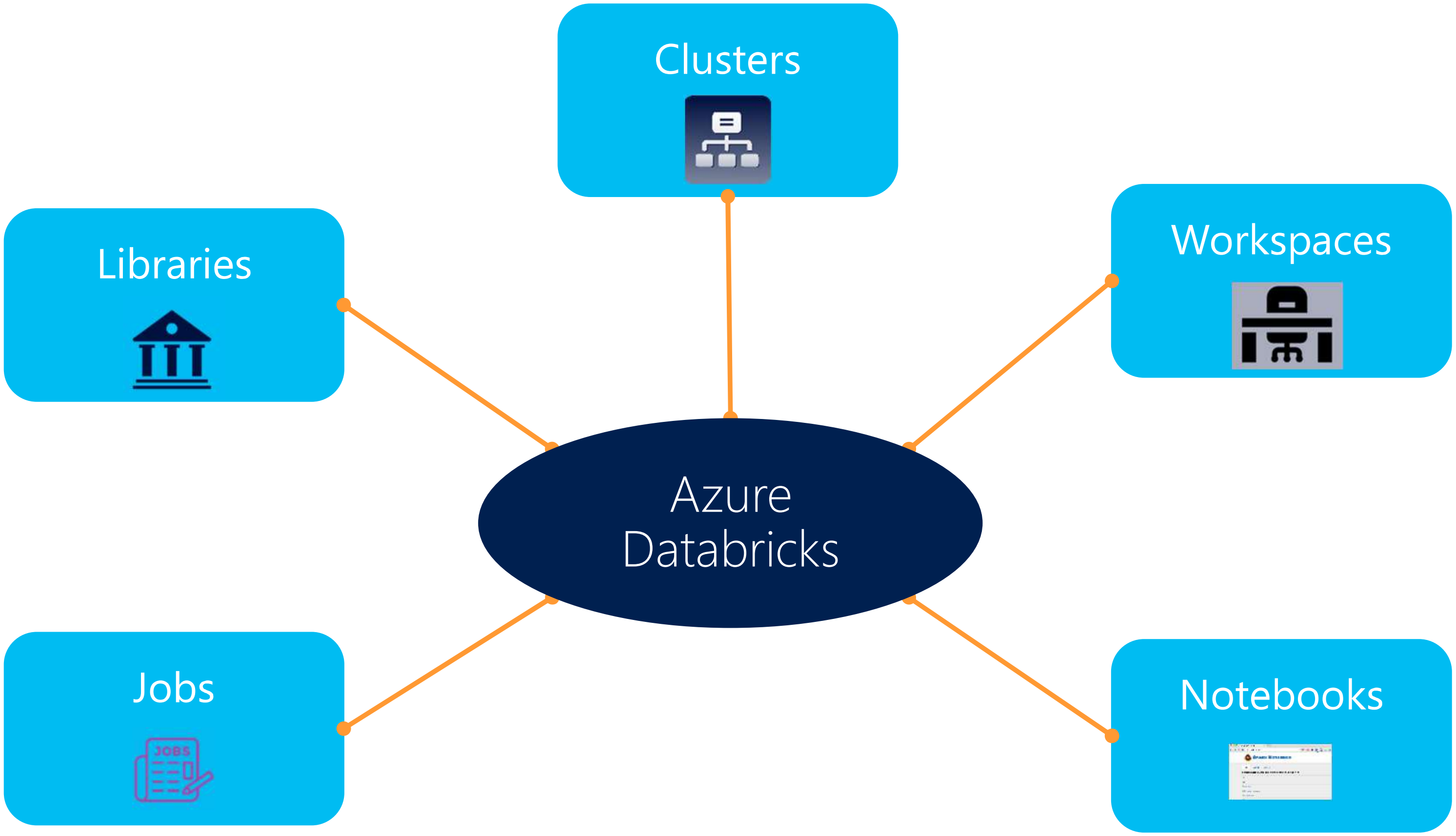
```
Command took 3.29 seconds
```

sean@frizdatalog.com
4/16/2020, 4:44:25 PM
What about 2017-2018?

Databricks Core Artifacts



Core Artifacts

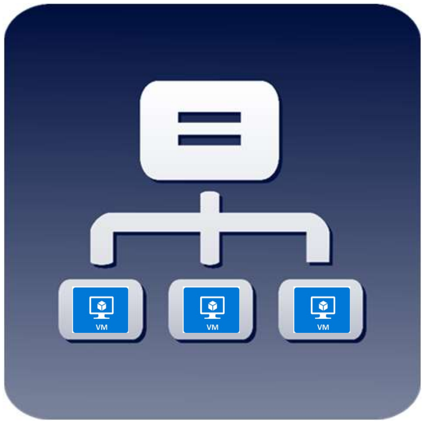


Cluster

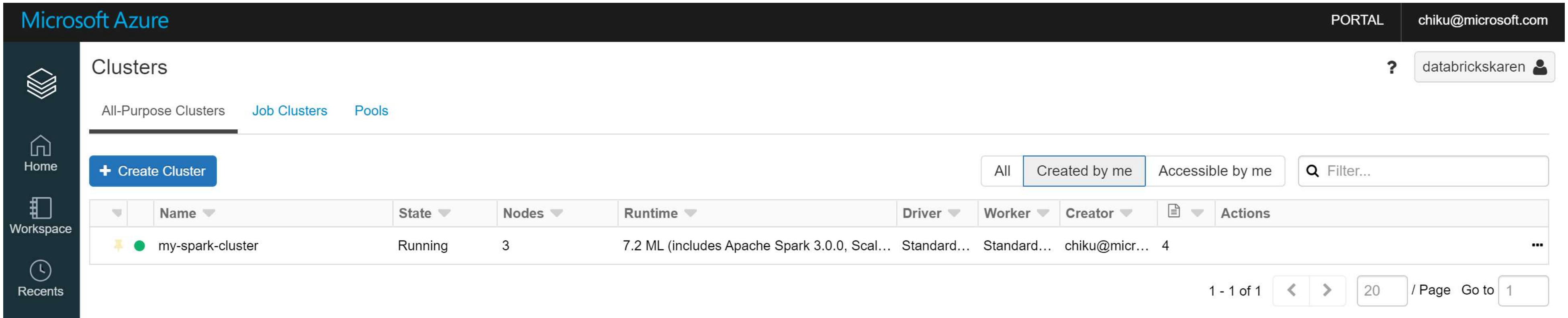
- Clusters are the set of Azure Linux VMs that host the Spark Worker and Driver Nodes
- Spark application code (i.e. Jobs) runs on the provisioned clusters.
- Clusters are launched in your subscription, but are managed through the Azure Databricks portal.

Cluster

- Azure Databricks provides a comprehensive set of graphical wizards to manage the complete lifecycle of clusters—from creation to termination.
- Types of Cluster:
  - Interactive Clusters: are used to analyze data collaboratively with interactive notebooks.
  - Job Clusters: are used to run fast and robust automated workloads using the UI or API.



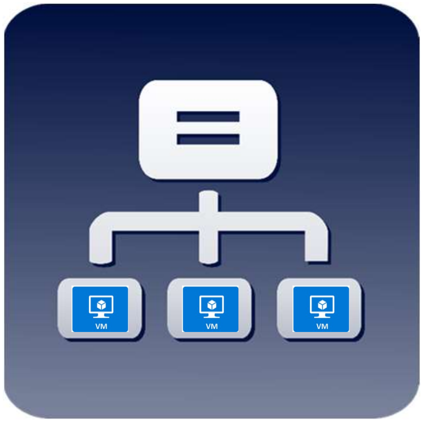
Clusters



Clusters

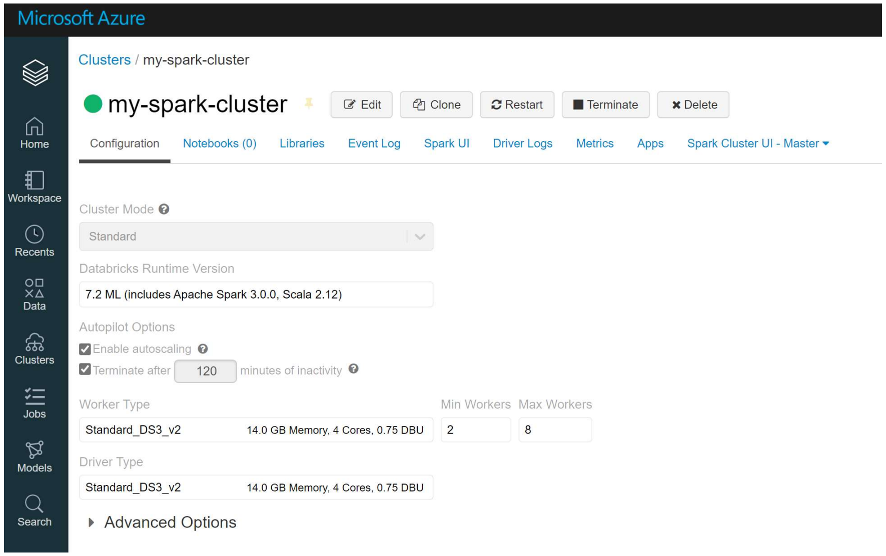
	Name	State	Nodes	Runtime	Driver	Worker	Creator		Actions
●	my-spark-cluster	Running	3	7.2 ML (includes Apache Spark 3.0.0, Scal…	Standard…	Standard…	chiku@micr…	4	•••

1 - 1 of 1   20 / Page   Go to 1



Driver : 1 node

Workers : 2~8 nodes



Clusters / my-spark-cluster

● my-spark-cluster

Cluster Mode: Standard

Databricks Runtime Version: 7.2 ML (includes Apache Spark 3.0.0, Scala 2.12)

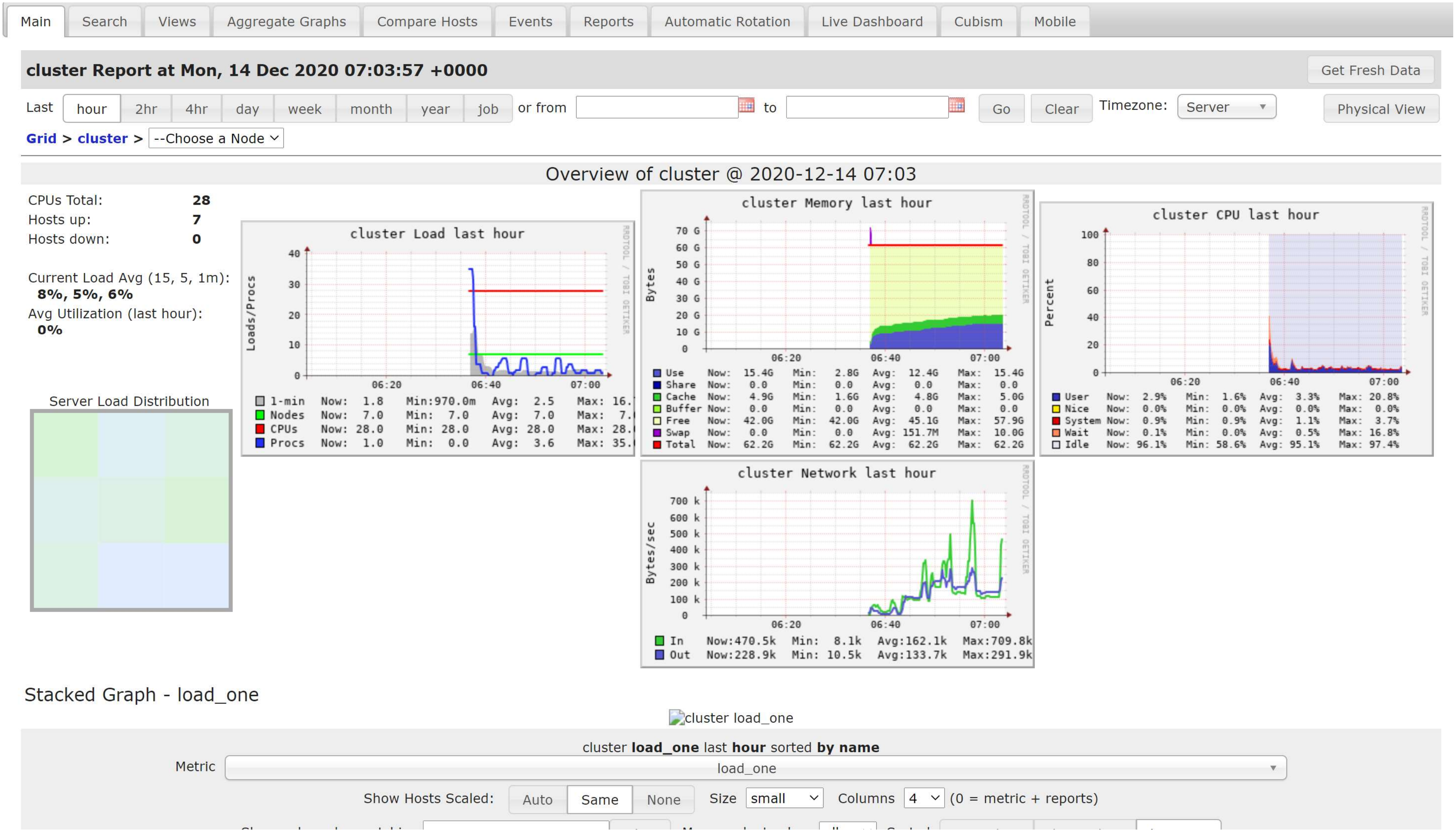
Autopilot Options

☑ Enable autoscaling

☑ Terminate after 120 minutes of inactivity

Worker Type: Standard_DS3_v2 14.0 GB Memory, 4 Cores, 0.75 DBU   Min Workers: 2   Max Workers: 8

Driver Type: Standard_DS3_v2 14.0 GB Memory, 4 Cores, 0.75 DBU



cluster Report at Mon, 14 Dec 2020 07:03:57 +0000

Last hour 2hr 4hr day week month year job or from ____ to ____ Go Clear Timezone: Server

Grid > cluster > --Choose a Node

Overview of cluster @ 2020-12-14 07:03

CPUs Total:	28
Hosts up:	7
Hosts down:	0

Current Load Avg (15, 5, 1m):
8%, 5%, 6%

Avg Utilization (last hour):
0%

Server Load Distribution

```
■ 1-min   Now:  1.8   Min:970.0m  Avg:  2.5   Max: 16.
■ Nodes   Now:  7.0   Min:  7.0   Avg:  7.0   Max:  7.
■ CPUs    Now: 28.0   Min: 28.0   Avg: 28.0   Max: 28.
■ Procs   Now:  1.0   Min:  0.0   Avg:  3.6   Max: 35.
```

```
■ Use    Now:  15.4G   Min:   2.8G   Avg:  12.4G   Max:  15.4G
■ Share  Now:   0.0    Min:   0.0    Avg:   0.0    Max:   0.0
■ Cache  Now:   4.9G   Min:   1.6G   Avg:   4.8G   Max:   5.0G
■ Buffer Now:   0.0    Min:   0.0    Avg:   0.0    Max:   0.0
□ Free   Now:  42.0G   Min:  42.0G   Avg:  45.1G   Max:  57.9G
■ Swap   Now:   0.0    Min:   0.0    Avg: 151.7M   Max:  10.0G
■ Total  Now:  62.2G   Min:  62.2G   Avg:  62.2G   Max:  62.2G
```

```
■ User   Now:  2.9%   Min:  1.6%   Avg:  3.3%   Max: 20.8%
■ Nice   Now:  0.0%   Min:  0.0%   Avg:  0.0%   Max:  0.0%
■ System Now:  0.9%   Min:  0.9%   Avg:  1.1%   Max:  3.7%
■ Wait   Now:  0.1%   Min:  0.0%   Avg:  0.5%   Max: 16.8%
□ Idle   Now: 96.1%   Min: 58.6%   Avg: 95.1%   Max: 97.4%
```

```
■ In   Now:470.5k  Min:  8.1k  Avg:162.1k  Max:709.8k
■ Out  Now:228.9k  Min: 10.5k  Avg:133.7k  Max:291.9k
```

Stacked Graph - load_one

cluster load_one

cluster **load_one** last **hour** sorted **by name**

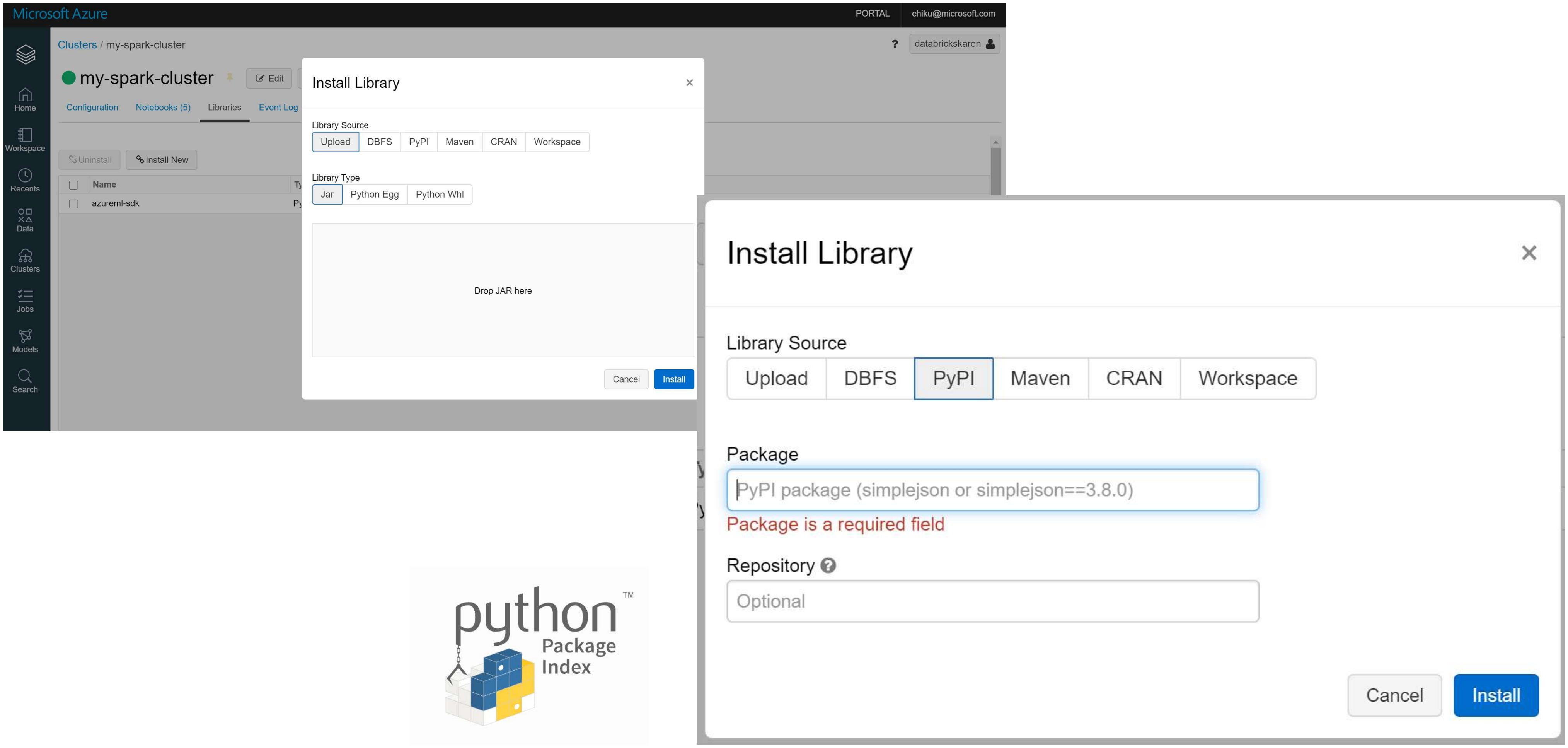
Metric load_one

Show Hosts Scaled: Auto Same None Size small Columns 4 (0 = metric + reports)

AutoScaling and AutoTermination Benefits

- Need not worry about # of nodes
  - You don’t need to guess or determine by trial and error, the correct number of nodes for the cluster.
- Dynamic Scaling
  - As the workload changes you do not have to manually tweak the number of nodes
- It’s pay-per-use!
  - You do not have to worry about wasting resources when the cluster is idle.
- Easy management
  - You do not have to wait and watch for jobs to complete just so you can shutdown the clusters.

How to install new packages



my-spark-cluster

Name
azureml-sdk

Install Library

Library Source: Upload

Library Type: Jar

Drop JAR here

Install Library

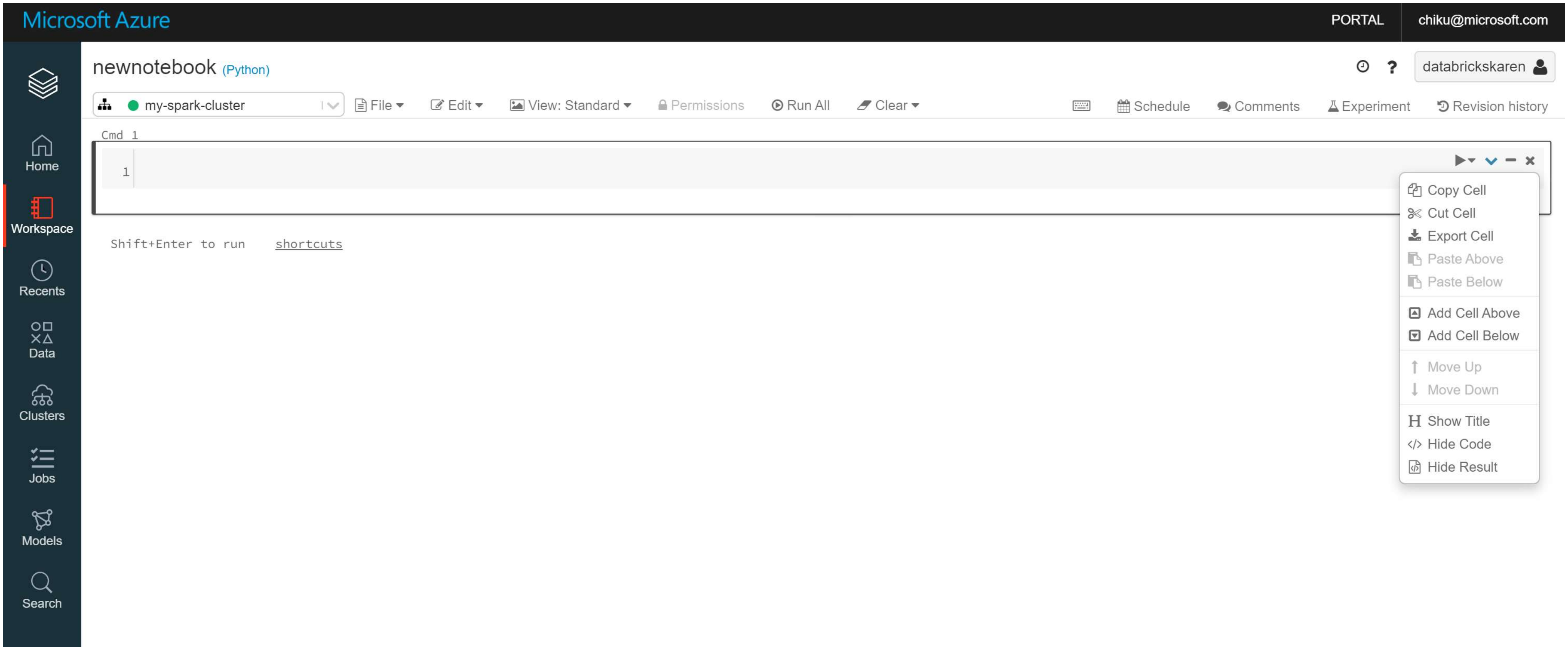
Library Source: PyPI

Package: PyPI package (simplejson or simplejson==3.8.0)

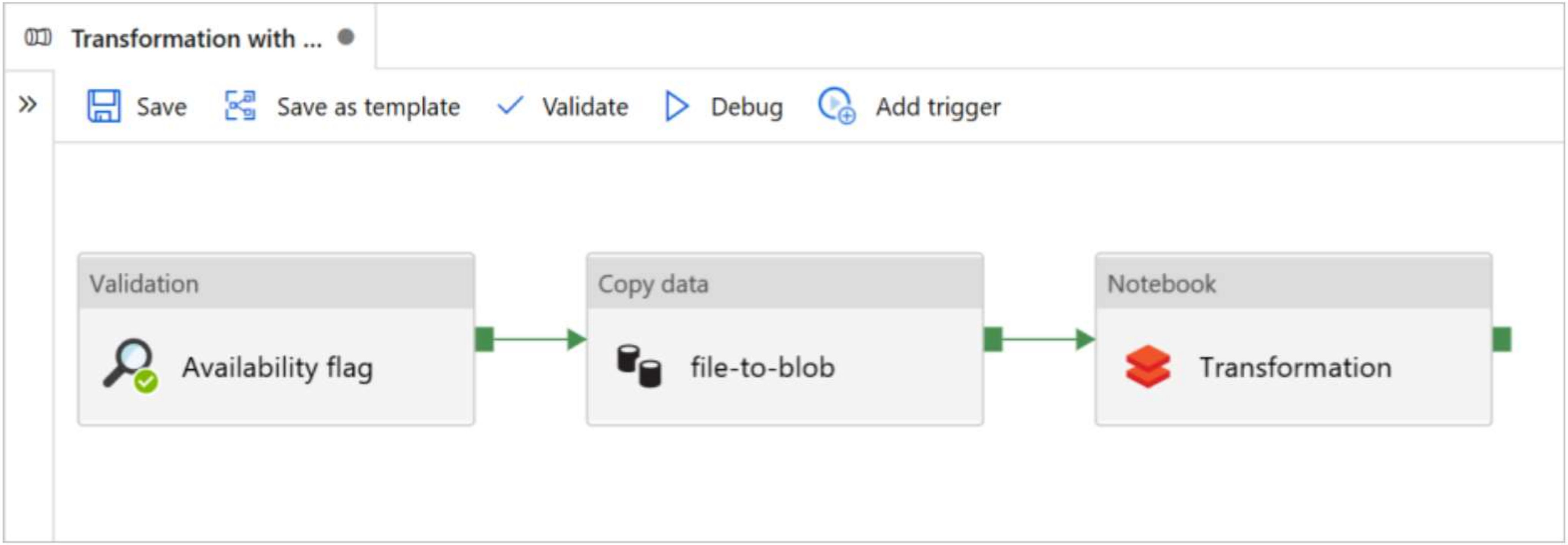
Package is a required field

Repository: Optional

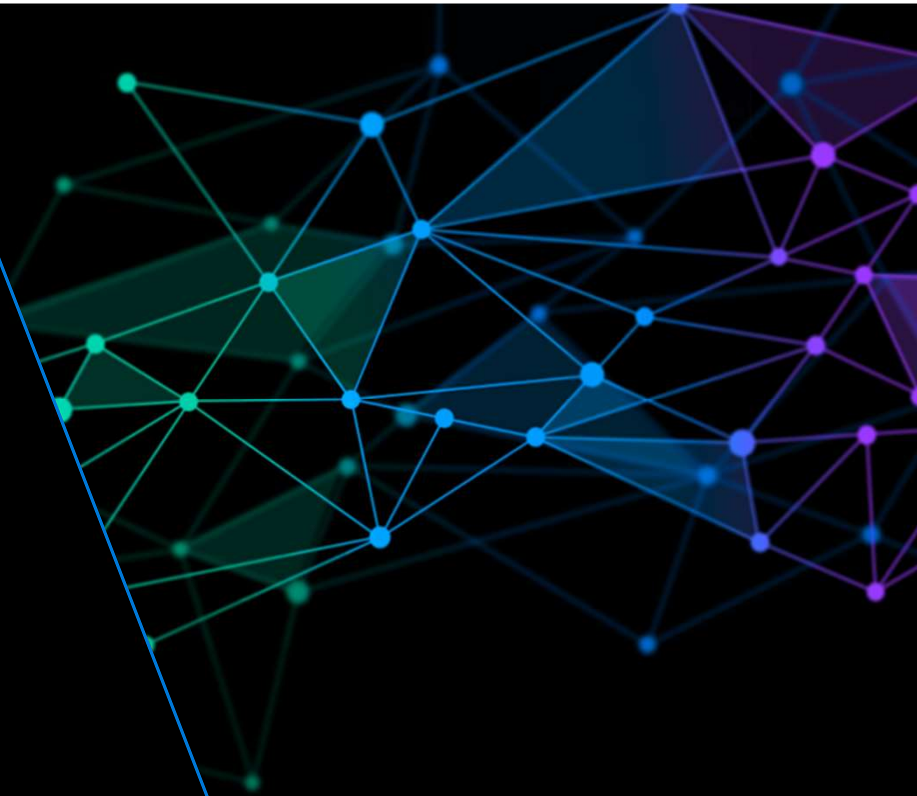
Notebook


Microsoft Azure
PORTAL
chiku@microsoft.com
newnotebook (Python)
databrickskaren
Home
Workspace
Recents
Data
Clusters
Jobs
Models
Search
my-spark-cluster
File
Edit
View: Standard
Permissions
Run All
Clear
Schedule
Comments
Experiment
Revision history
Cmd 1
Copy Cell
Cut Cell
Export Cell
Paste Above
Paste Below
Add Cell Above
Add Cell Below
Move Up
Move Down
Show Title
Hide Code
Hide Result
Shift+Enter to run
shortcuts

Notebook with Azure Data Factory

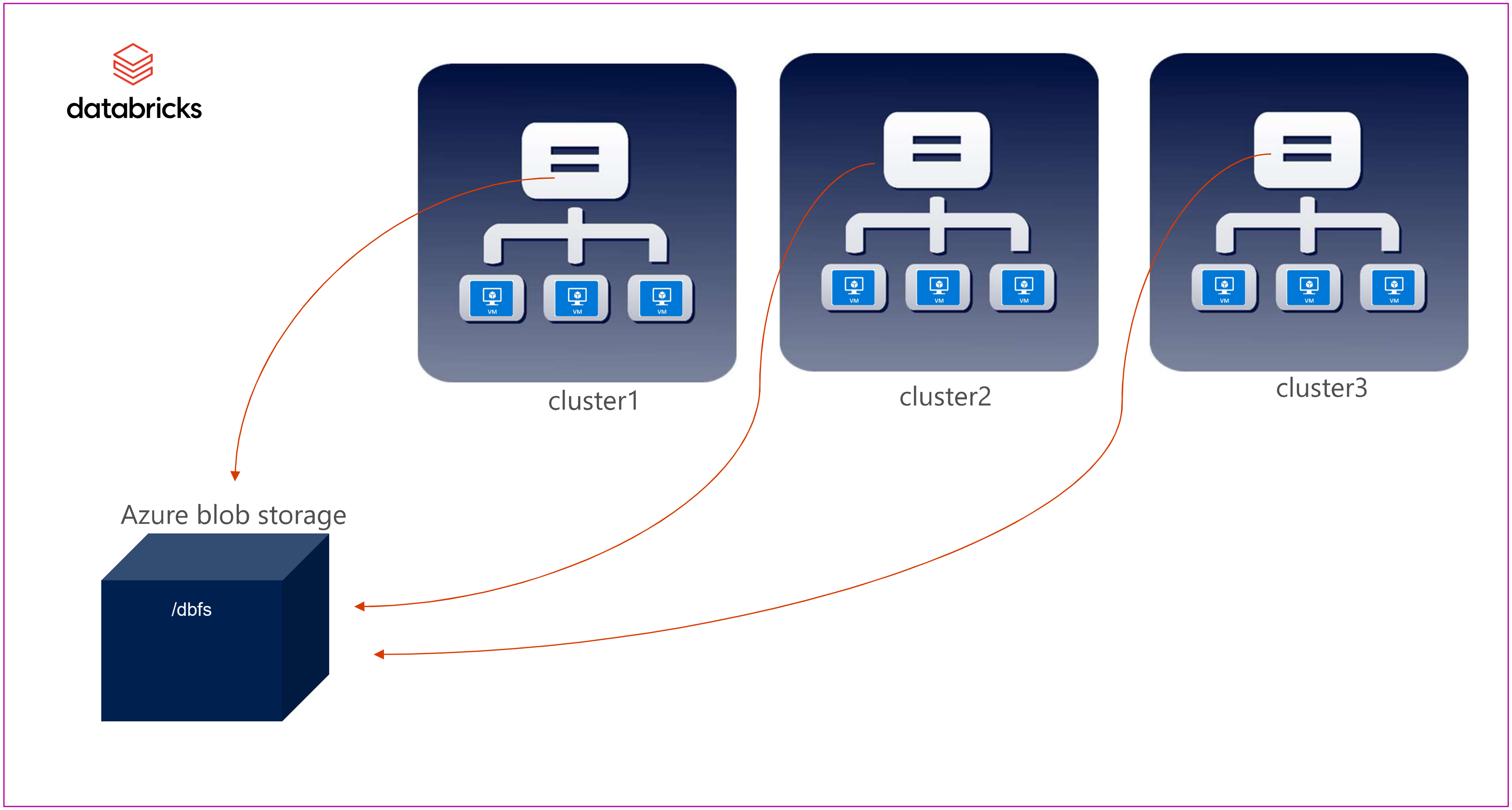


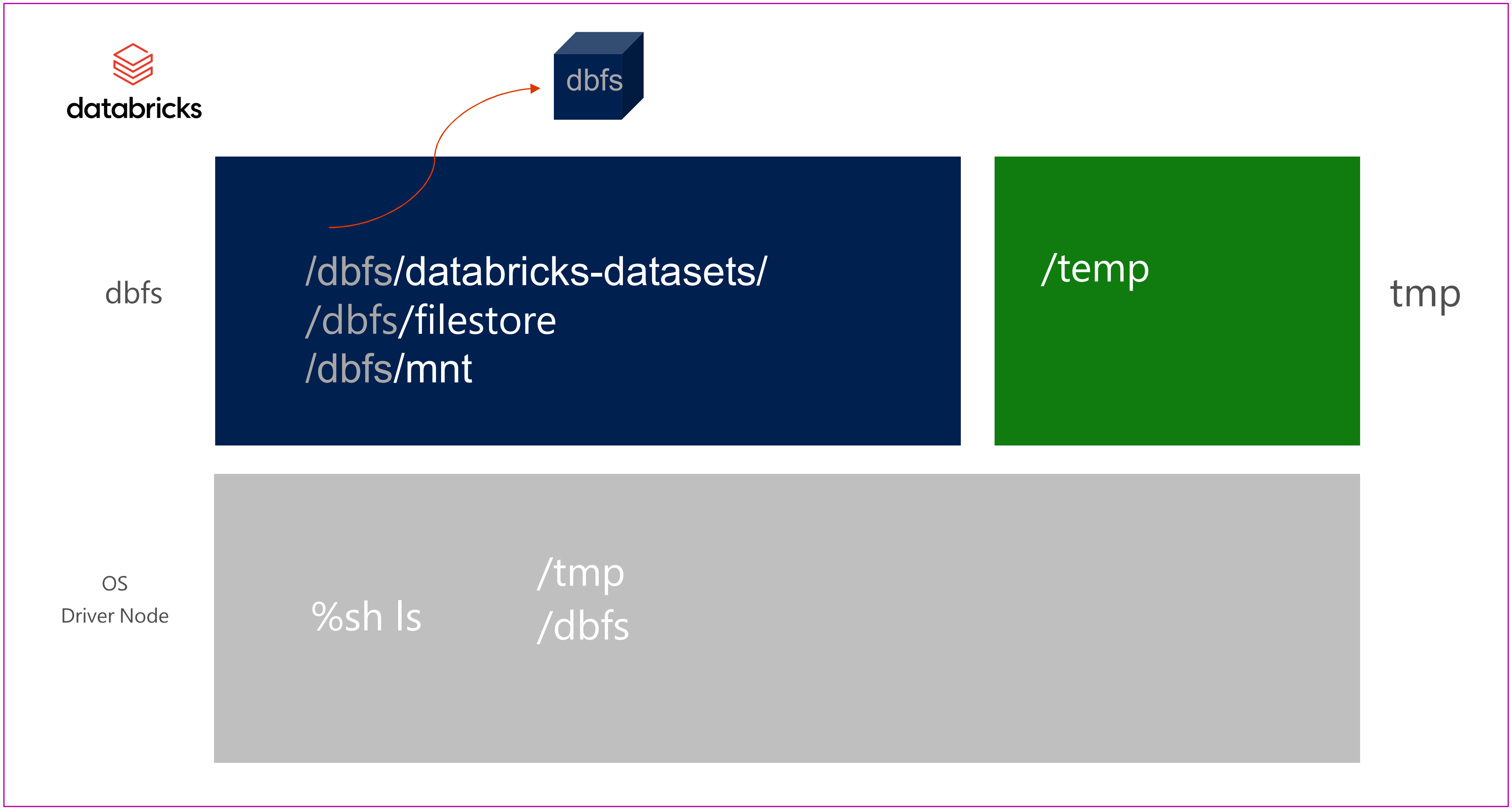
Databricks DBFS



What is DBFS?

- DBFS or Databricks File System is the distributed file system that is on all Databricks Runtime Clusters
- DBFS uses Azure Blob Storage on the backend to persist the data
- You can store your tables, data files, or logs in this system and access it via tools like the Databricks CLI, DBFS API, dbutils, Spark APIs, and even local file APIs


databricks
cluster1
cluster2
cluster3
VM
Azure blob storage
/dbfs



databricks

dbfs

dbfs

/dbfs/databricks-datasets/
/dbfs/filestore
/dbfs/mnt

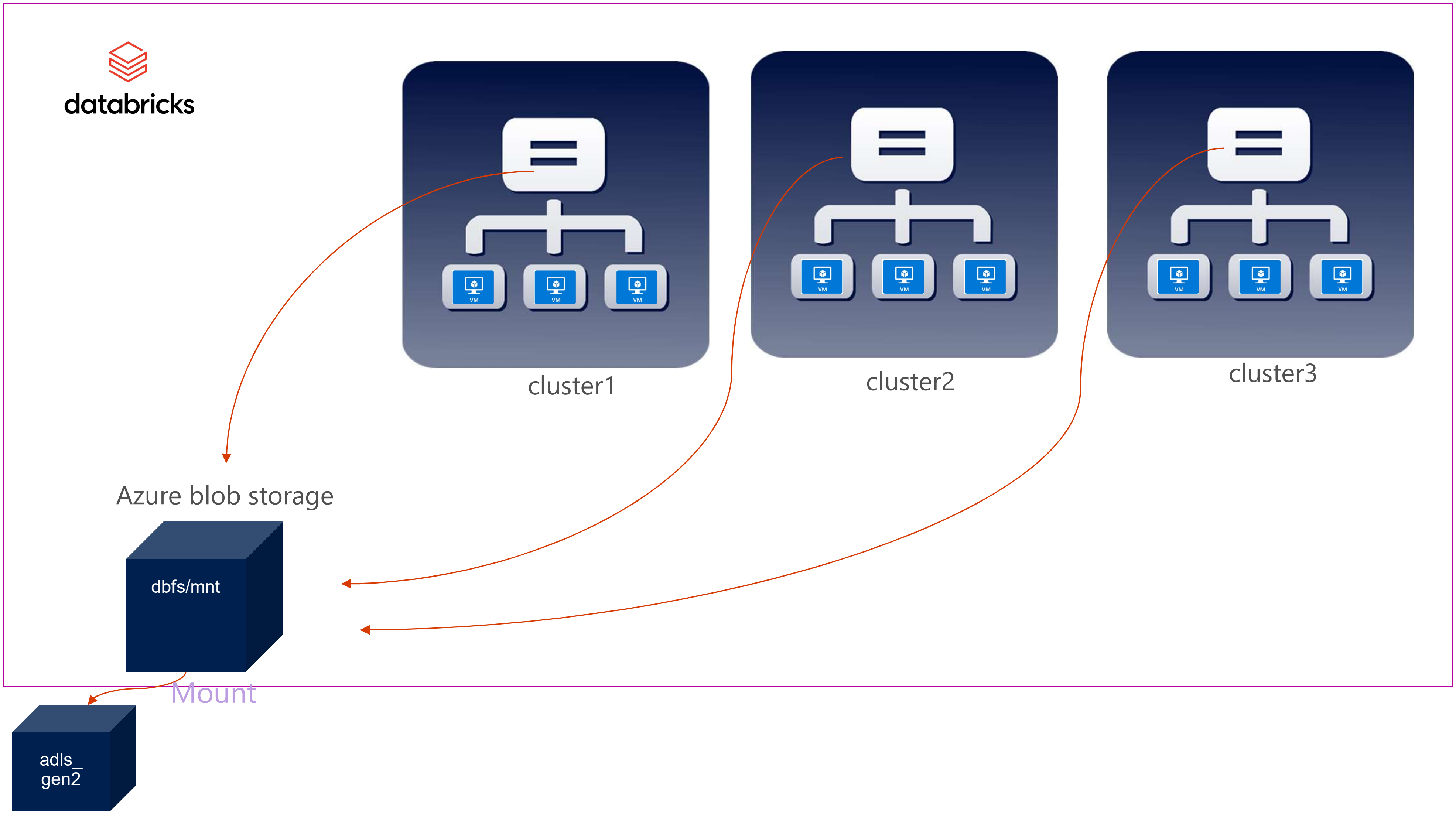
/temp

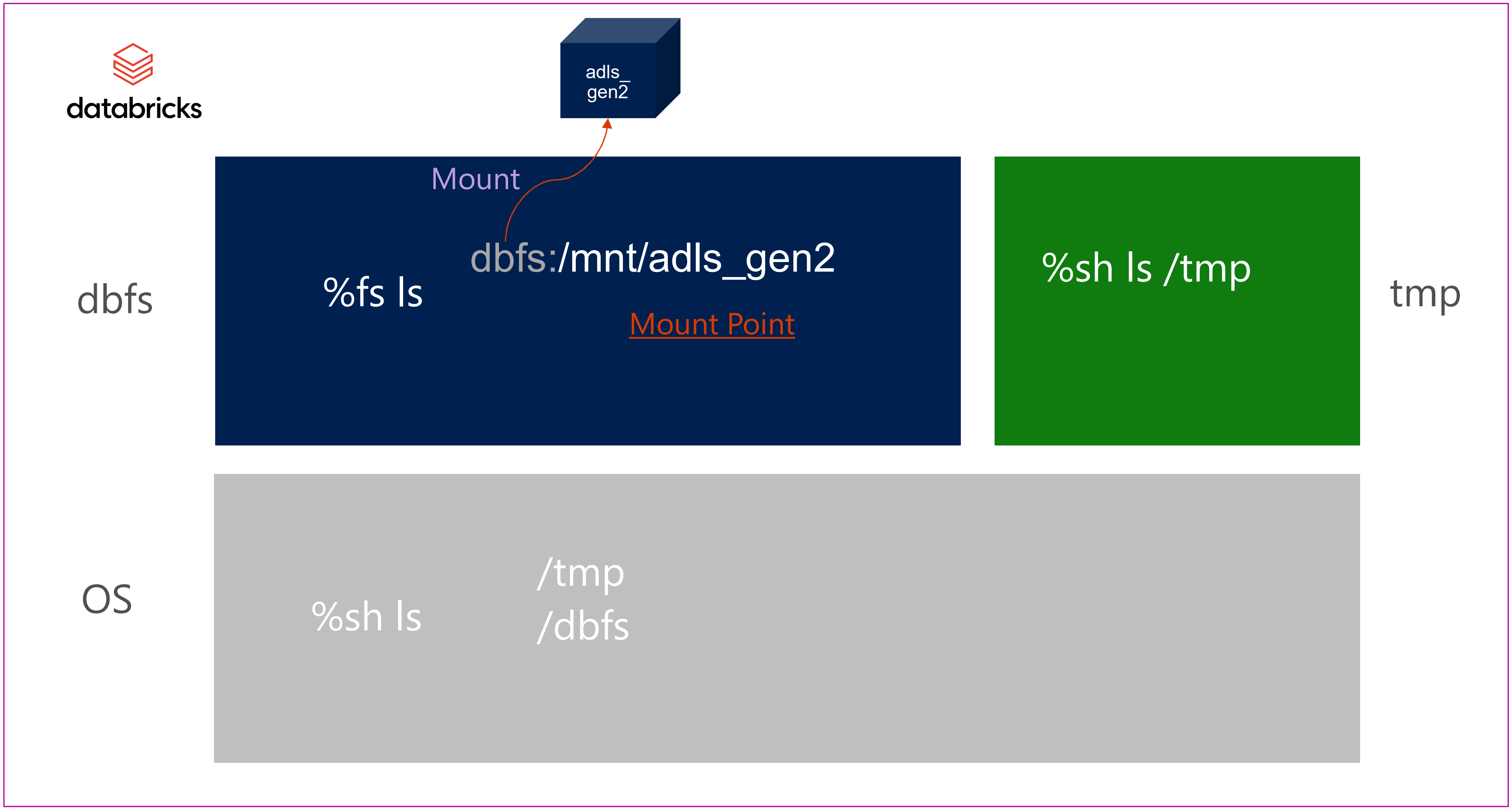
tmp

OS
Driver Node

%sh ls

/tmp
/dbfs


databricks
cluster1
cluster2
cluster3
Azure blob storage
dbfs/mnt
Mount
adls_
gen2



databricks

adls_
gen2

Mount

dbfs

%fs ls

dbfs:/mnt/adls_gen2

Mount Point

%sh ls /tmp

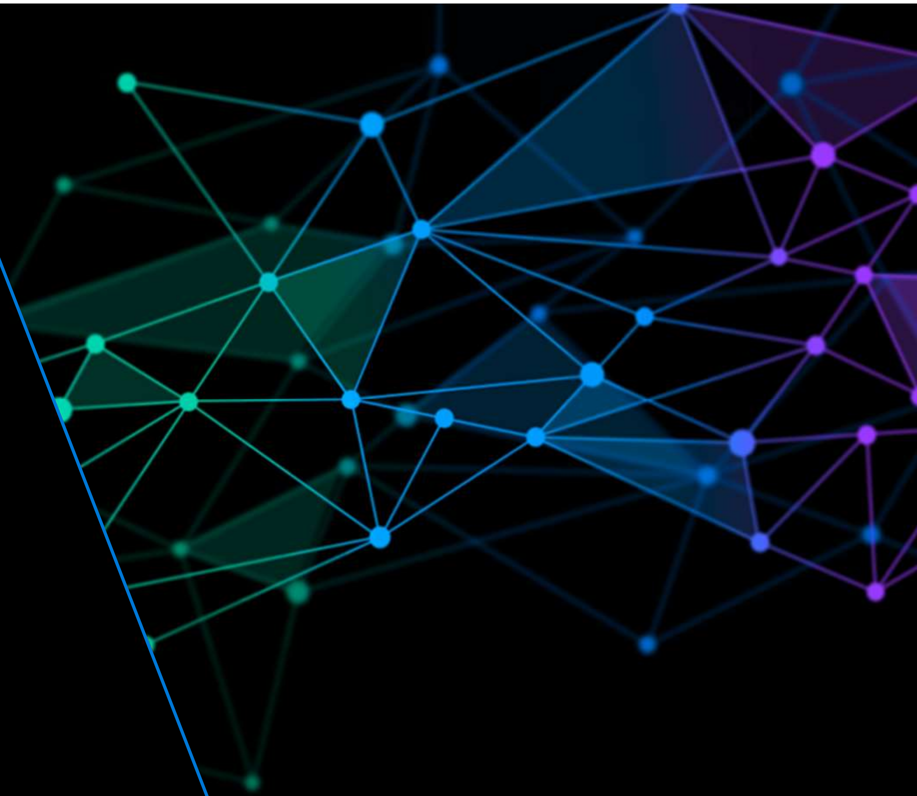
tmp

OS

%sh ls

/tmp
/dbfs

Customer Story



The Problem

Limited Local Resource

Data Everywhere

Big Amount of Data Quantity

Slow ELT/ Data Science

Platform Build Goals

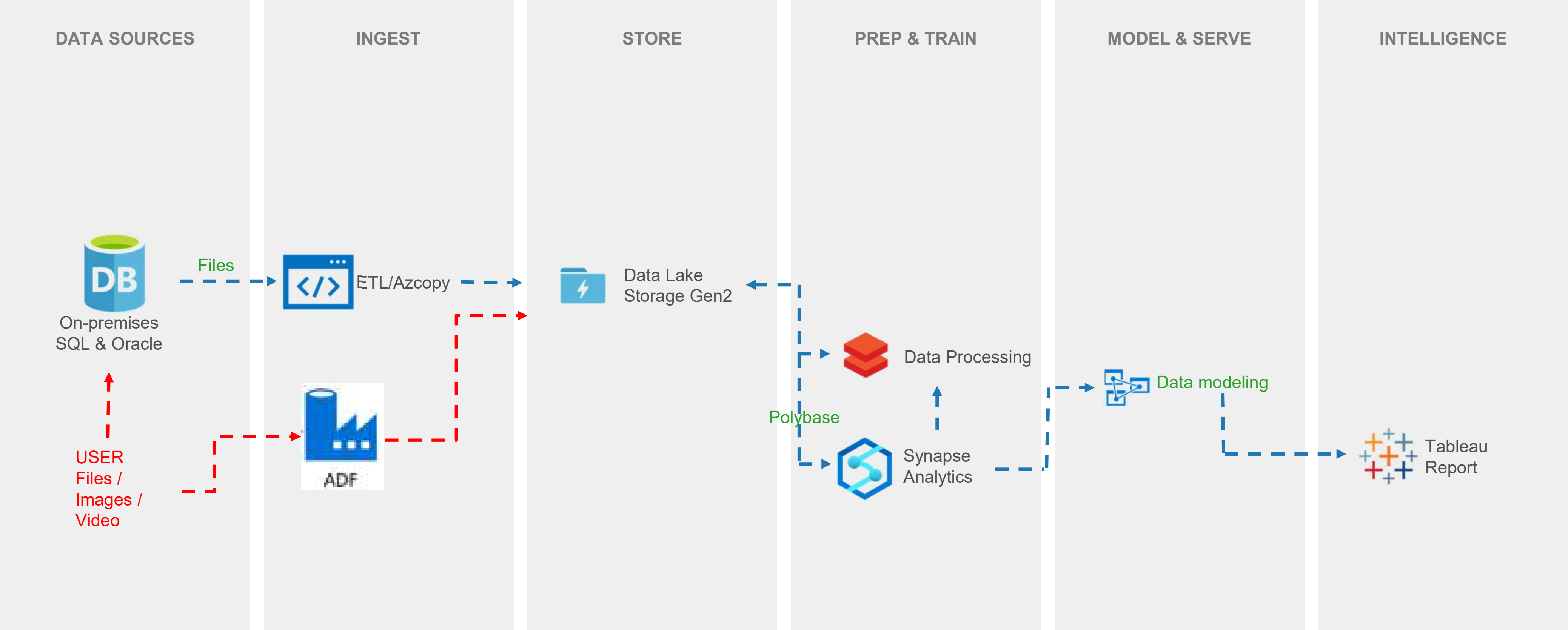
Cloud Focus

Centralize Data

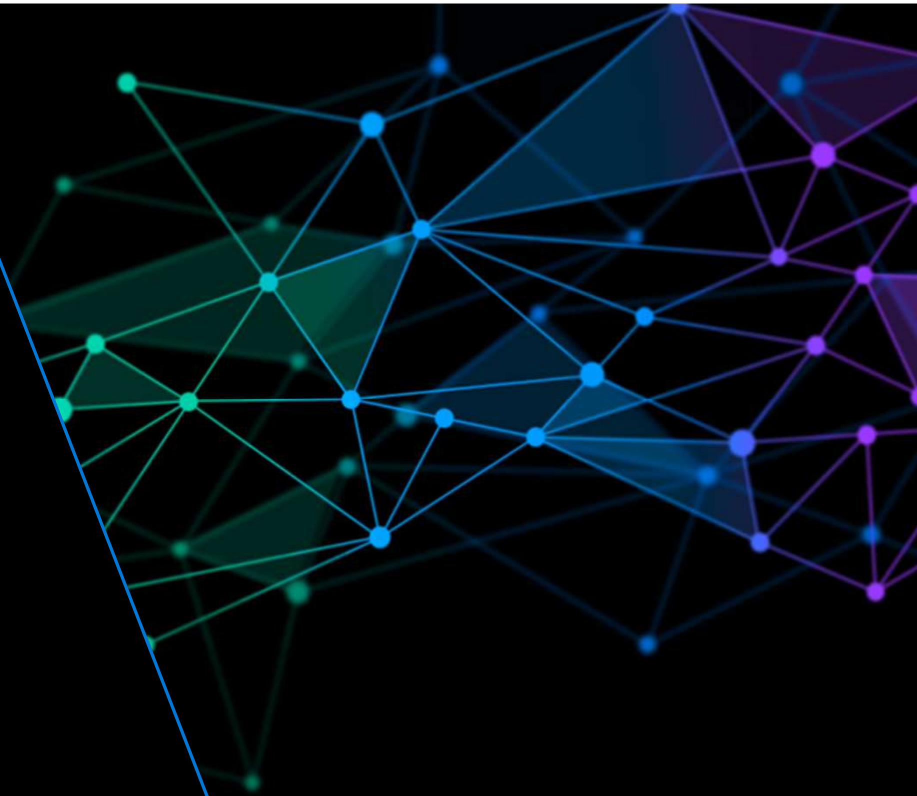
Tools for data science and analysis

Increase Productivity

Data Transfer End to End Workflow

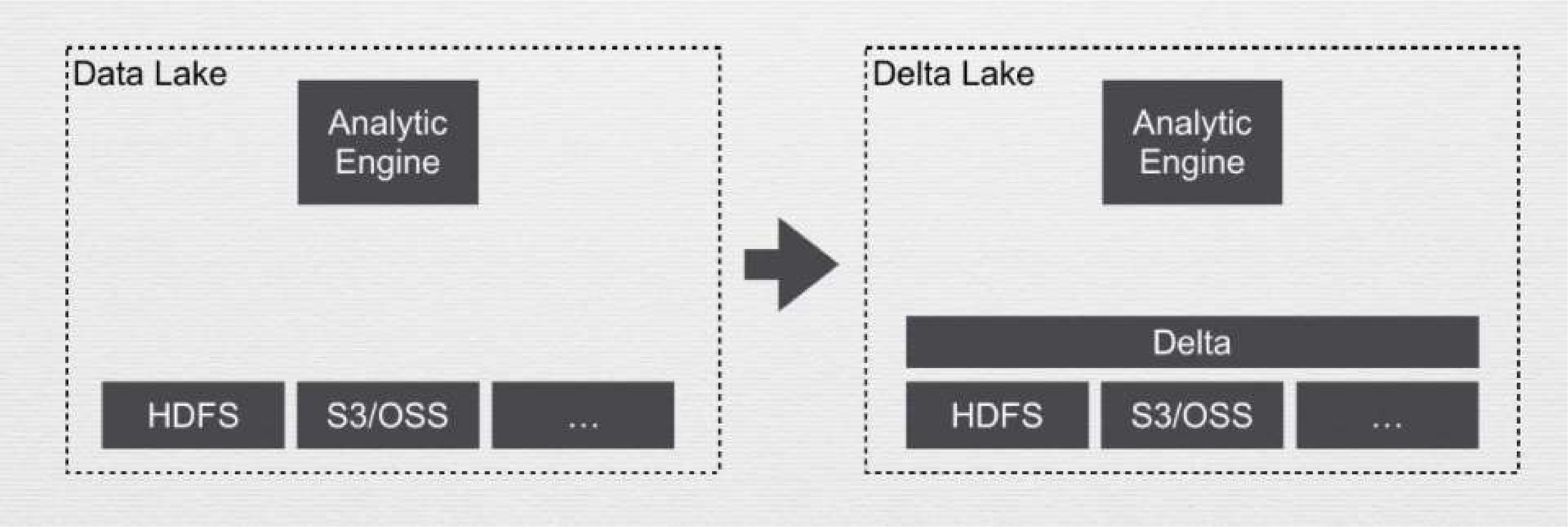


Delta Table



Delta Lake

- Delta Lake is a storage layer that brings scalable, ACID transactions to Apache Spark and other big-data engines.



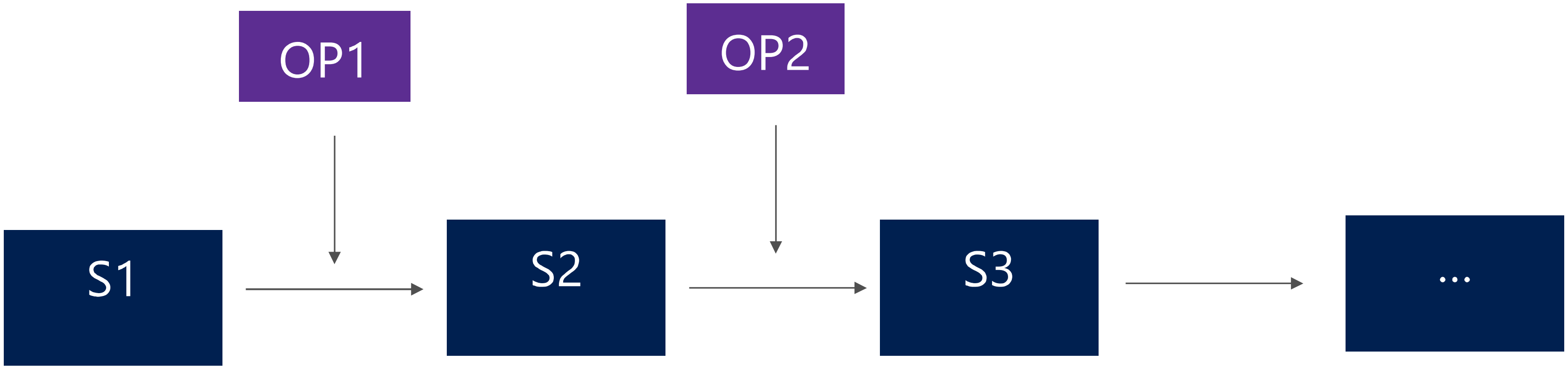
Delta Lake

- Data Lake
  - 讀寫併發, 數據不可靠
  - 數量質量較低
  - 性能較低
  - 無法更新/刪除數據

- Delta Table
  - ACID Transaction
  - Schema管理
  - Data Skipping, Z-ordering, Compaction
  - 資料更新和刪除
  - 可伸縮的中繼資料處理
  - 資料版本控制和時間旅行
  - 串流與批次統一

Delta Table 原理

- 紀錄每一個文件變動, 形成新的快照版本
- 歷史是線性的
- 歷史可回溯



寫入模式

使用 Append 模式，可以自動將新資料追加到現有 Delta Lake 表：

```
df.write.format("delta").mode("append").save("/delta/events")
```

替換表中的所有資料，可以使用 overwrite 模式：

```
df.write.format("delta").mode("overwrite").save("/delta/events")
```

將1月份替換為df中的資料：

```
df.write .format("delta") .mode("overwrite")
.option("replaceWhere", "date >= '2017-01-01' AND date <= '2017-
01-31'") .save("/delta/events")
```

表格Delete, Update and Upsert

Delta Lake 支援數個語句，有助於刪除差異資料表中的資料，以及更新資料。

```
%sql
MERGE INTO events
USING updates
ON events.eventId = updates.eventId
WHEN MATCHED THEN
  UPDATE SET
    events.data = updates.data
WHEN NOT MATCHED
  THEN INSERT (date, eventId, data) VALUES (date, eventId, data)
```

資料版本控制和時間旅行

Delta Lake 提供差異資料表會保留30天的歷程記錄。 這表示您可以指定30天前的版本。

```
%sql
DESCRIBE HISTORY eventsTable
```

Delta Lake 表創建一個DataFrame 關聯到表的特定版本，可以使用如下兩種方式：

```
%sql
SELECT * FROM events TIMESTAMP AS OF timestamp_expression
SELECT * FROM events VERSION AS OF version
```

Delta Cache

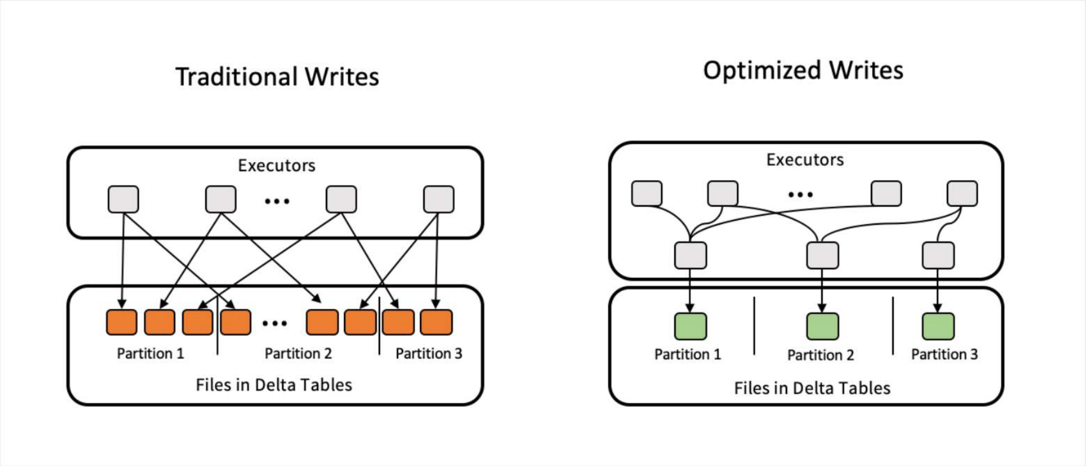
透過 Delta Cache來改善查詢效率, 任何Parquet table stored on WASB and other file system都可以創建Cache

```
%sql
DROP TABLE IF EXISTS TABLE
CACHE TABLE
SELECT column_name
FROM [db_name.]table_name
[ WHERE boolean_expression ]
```

```
%sql
Clear Cache
```

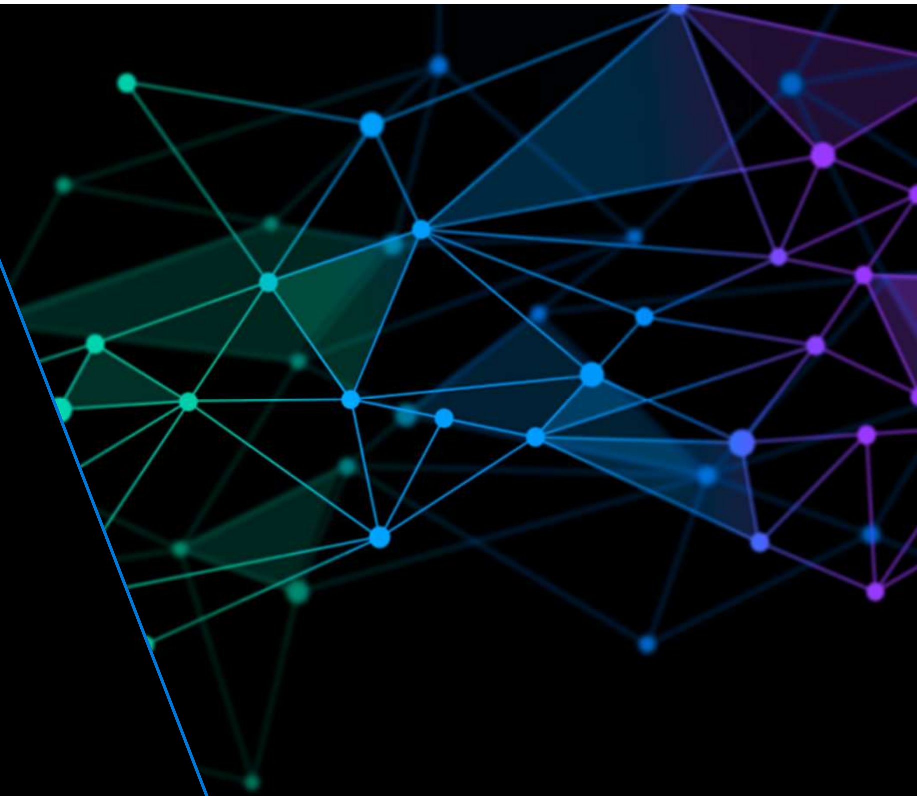
Optimize performance with file managemer

- Compaction (bin-packing)
  多的小檔案合併為一個128左右大小的檔案來改善讀取的效率



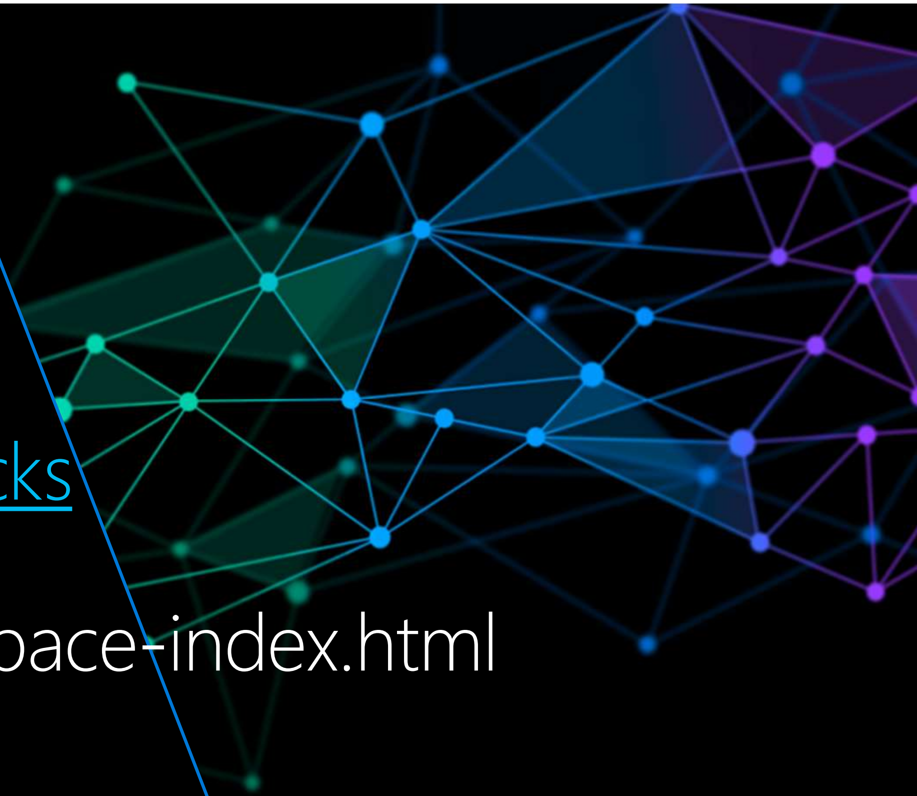
- Data skipping
- Z-Ordering (multi-dimensional clustering)

Demo: COVID-19



Reference Data

https://databricks.com/try-databricks
https://academy.databricks.com/
https://docs.databricks.com/workspace-index.html



Q & A