Azure Advanced Analytics engine for Data Science

Adrián J. Fernández Zenteno

Cloud Solution Architect - Azure Advanced Analytics & Data Science

Email: adrian.fernandez@microsoft.com

Twitter: @AdrianFZ10

Microsoft

Agenda

- Azure Databricks
- Spark Summit 2020
- Synapse Analytics Studio
- Execution Pools engine:
  - SQL Analytics / On-Demand
  - Apache Spark
- Synapse and Azure ML integration

Machine Learning on Azure

Domain specific pretrained models

To simplify solution development

- Vision
- Speech
- Language
- Search

Familiar data science tools

To simplify model development

- Visual Studio Code
- Azure Notebooks
- Jupyter
- Command line

Popular frameworks

To build advanced deep learning solutions

- PyTorch
- TensorFlow
- Scikit-Learn
- ONNX

Productive services

To empower data science and development teams

- Azure Machine Learning
- Azure Databricks
- Synapse Analytics
- ML VMs

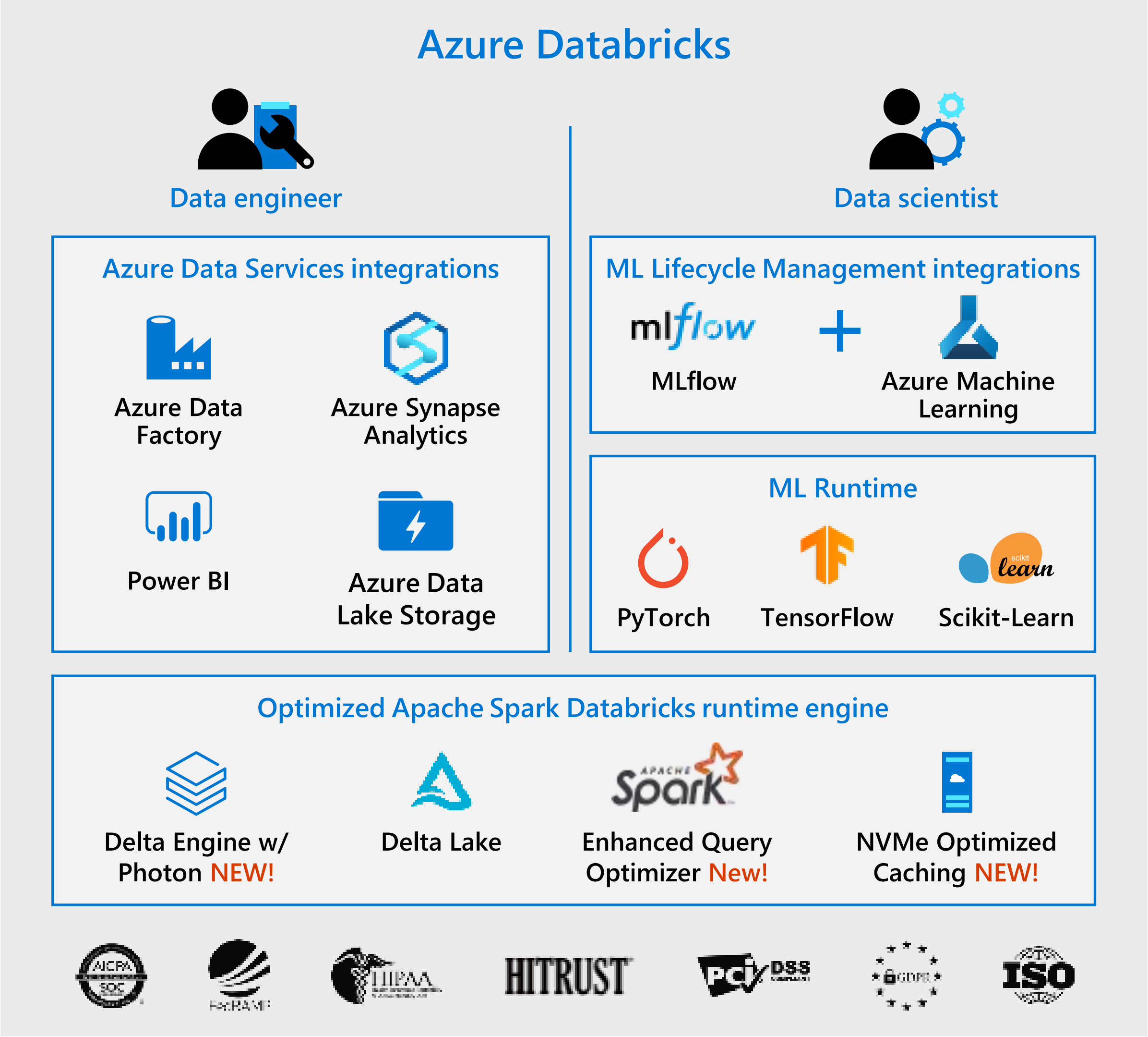
Powerful infrastructure

To accelerate deep learning

- CPU
- GPU
- FPGA

From the intelligent Cloud to the intelligent Edge

Azure Databricks – Introduction



Azure Databricks

Data engineer

Data scientist

Azure Data Services integrations

Azure Data Factory

Azure Synapse Analytics

Power BI

Azure Data Lake Storage

ML Lifecycle Management integrations

MLflow

Azure Machine Learning

ML Runtime

PyTorch

TensorFlow

Scikit-Learn

Optimized Apache Spark Databricks runtime engine

Delta Engine w/ Photon NEW!

Delta Lake

Enhanced Query Optimizer New!

NVMe Optimized Caching NEW!

Collaborative

Workspaces for data teams across the full lifecycle

Connected

Native integration with the entire Azure Portfolio

Leverage the most popular open source tools

Fast

Scalable and reliable data powered by the fastest Spark Engine on the market

Secure

Azure Active Directory Single Sign-On

Azure Databricks – Top Announcements 📢

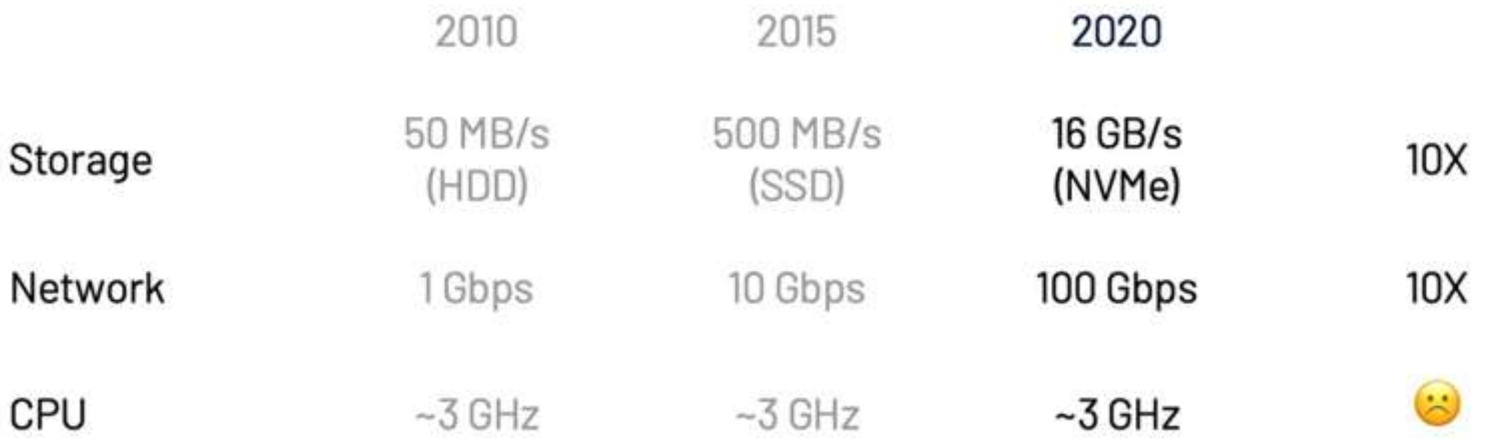
- Databricks Runtime 7.x with Apache Spark 3.0!
- Delta Engine with Photon!
- Koalas
- Redash
- Workspace 2.0
- Azure US Gov Preview with FedRAMP High Certification
- Spark AI Summit

https://databricks.com/sparkaisummit/north-america-2020

Azure Databricks - Delta Engine

Azure Databricks – Delta Engine Motivation

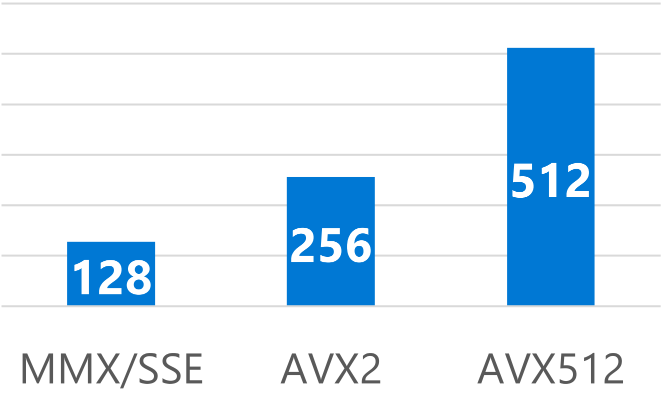
Hardware Trends



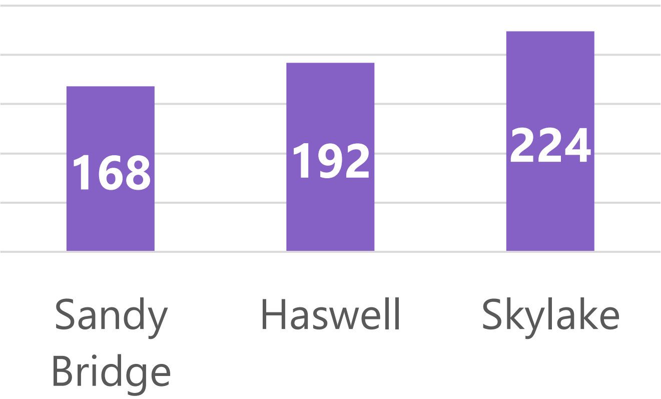
	2010	2015	2020	
Storage	50 MB/s (HDD)	500 MB/s (SSD)	16 GB/s (NVMe)	10X
Network	1 Gbps	10 Gbps	100 Gbps	10X
CPU	~3 GHz	~3 GHz	~3 GHz	🙁

So where are CPU innovations going?

Data level parallelism (SIMD register width)

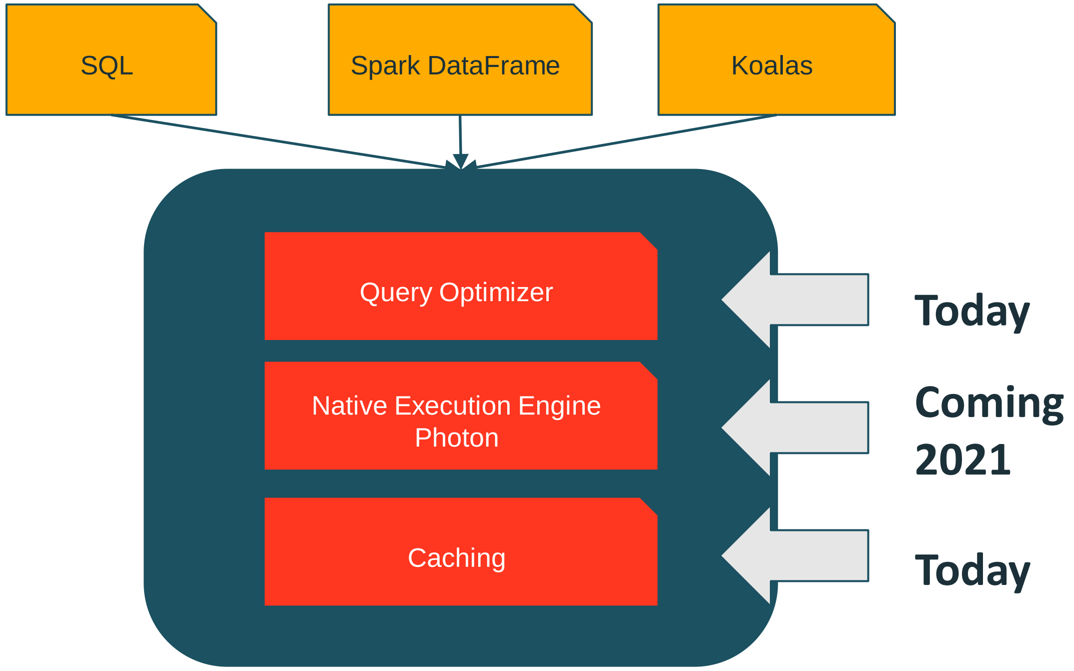


Instruction level parallelism:



Azure Databricks – Delta Engine

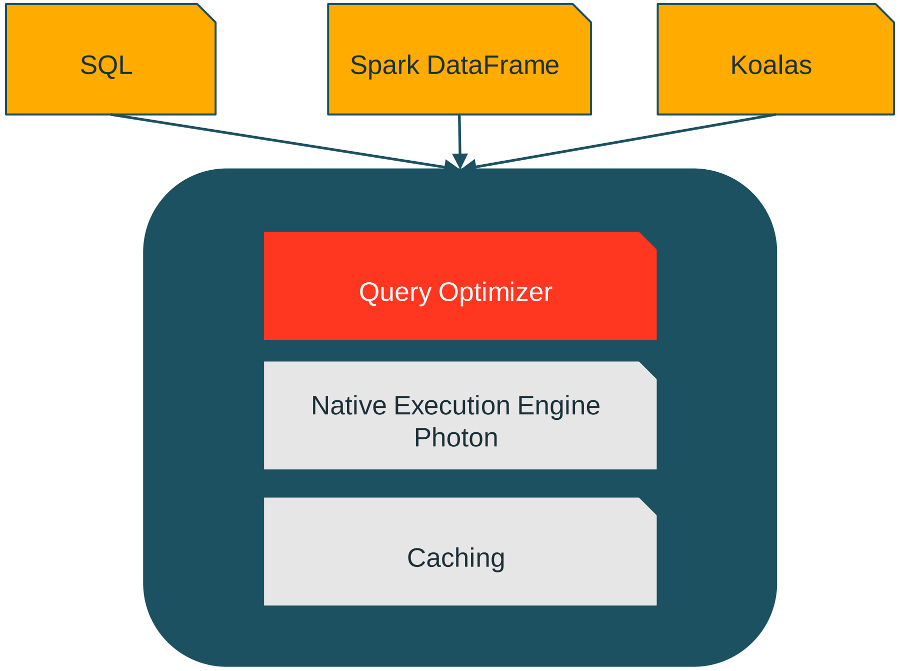
- Builds On Apache Spark 3.0
- 100% Spark Compatibility
- Fully API compatible
- Accelerates SQL and DataFrame workloads with:
  - Improved query optimizer
  - Native vectorized execution engine
  - Caching



Delta Engine

Delta Engine’s Improved Query Optimizer

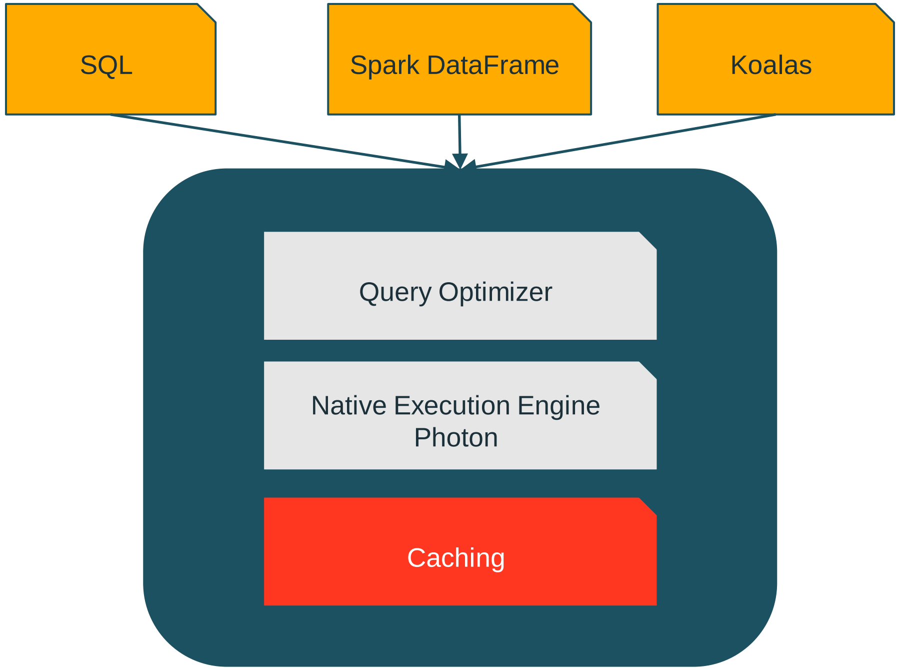
- Extends Spark’s cost-based optimizer and adaptive query execution with advanced stats
- Up to 18x performance increase for star schema workloads



Delta Engine

Delta Engine’s Caching

- Automatically caches input
- Transcodes data into a more CPU-efficient format fully leveraging NVMe SSDs
- Up to 5x scan performance increase

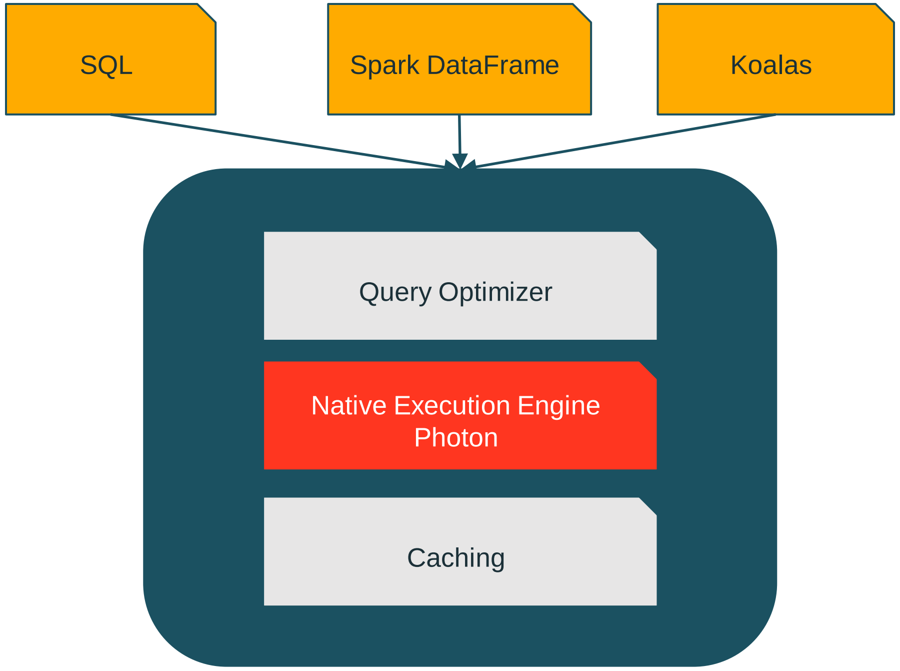


Delta Engine

Delta Engine Photon

- New execution engine for Delta Engine to accelerate Spark SQL

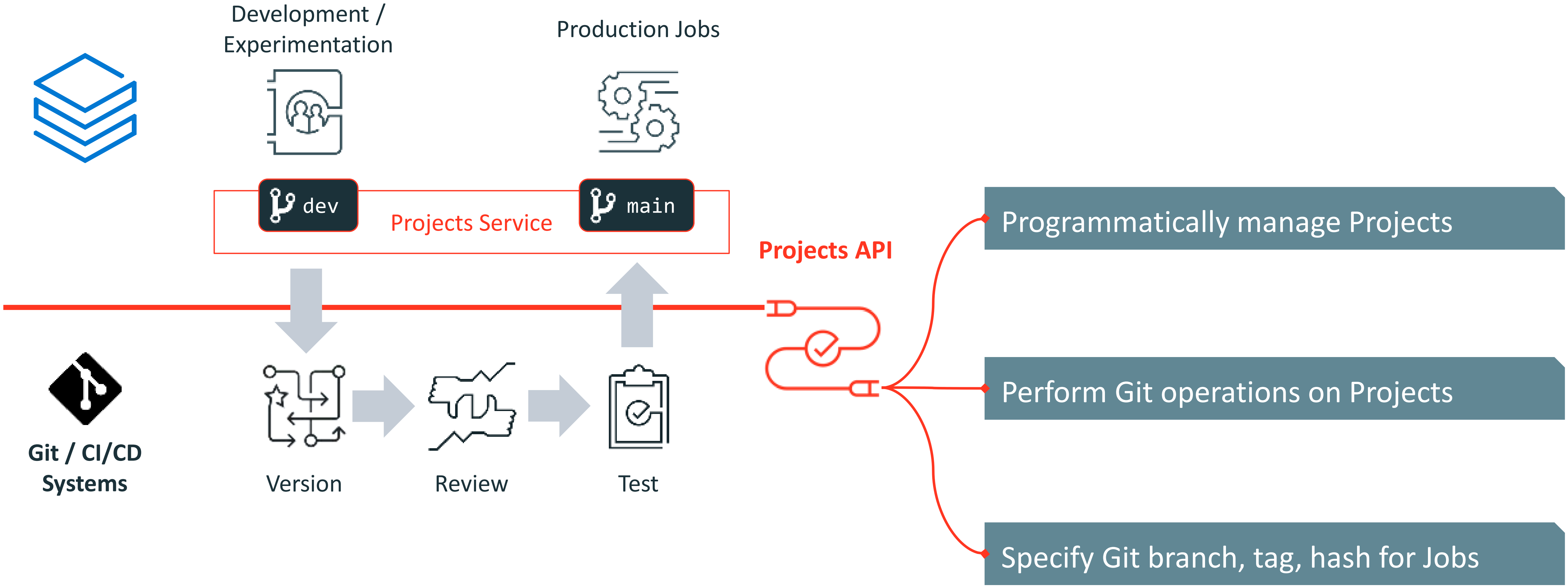
- Built from scratch in C++, for performance:
  - Vectorization: exploit data-level parallelism and instruction-level parallelism
  - Optimize for modern structured and semi-structured workloads



Delta Engine

Azure Databricks – Workspace 2.0

Azure Databricks – Projects API for CI/CD



Development / Experimentation

Production Jobs

dev

Projects Service

main

Projects API

Programmatically manage Projects

Perform Git operations on Projects

Specify Git branch, tag, hash for Jobs

Git / CI/CD Systems

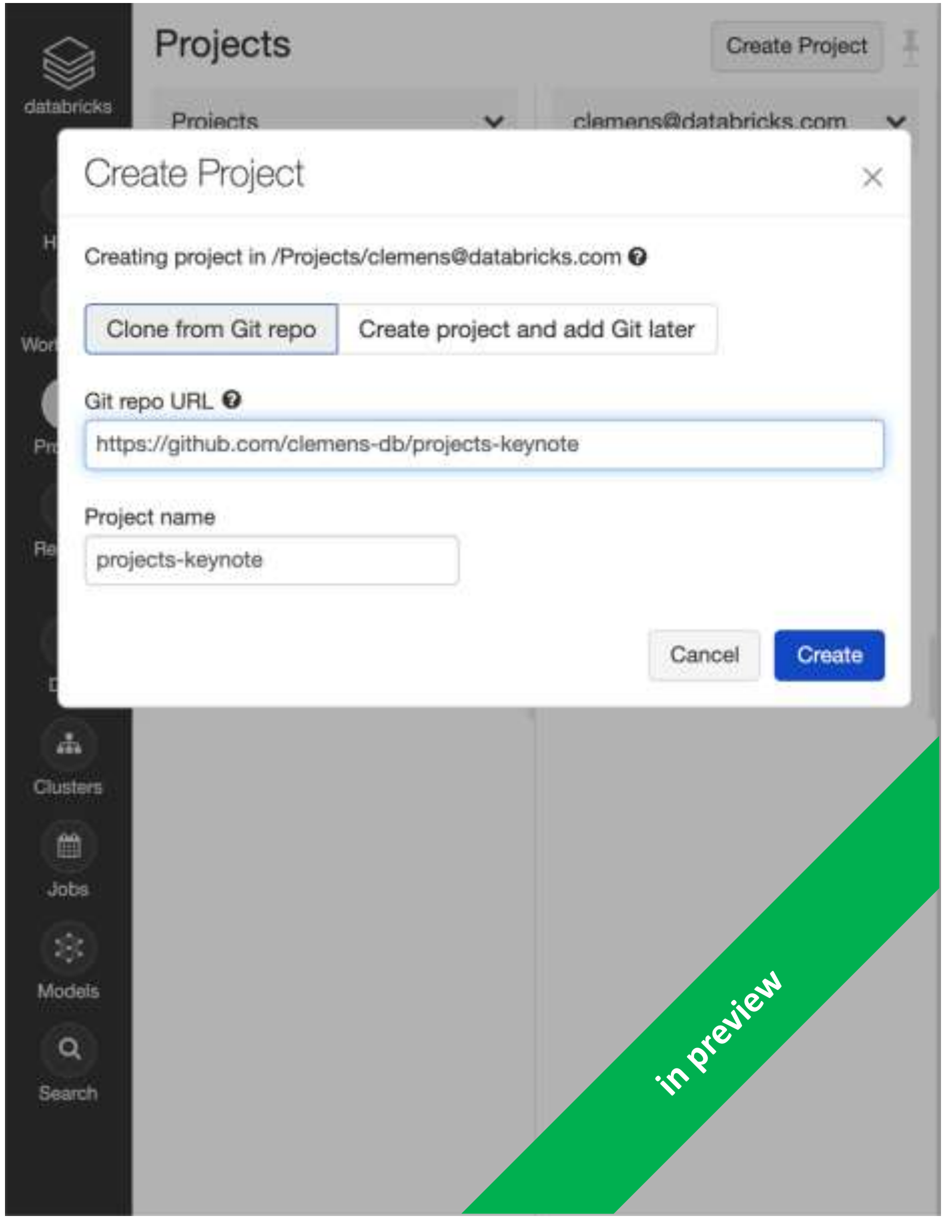
Version

Review

Test

Azure DevOps Bitbucket GitLab GitHub

Repository-level git integration

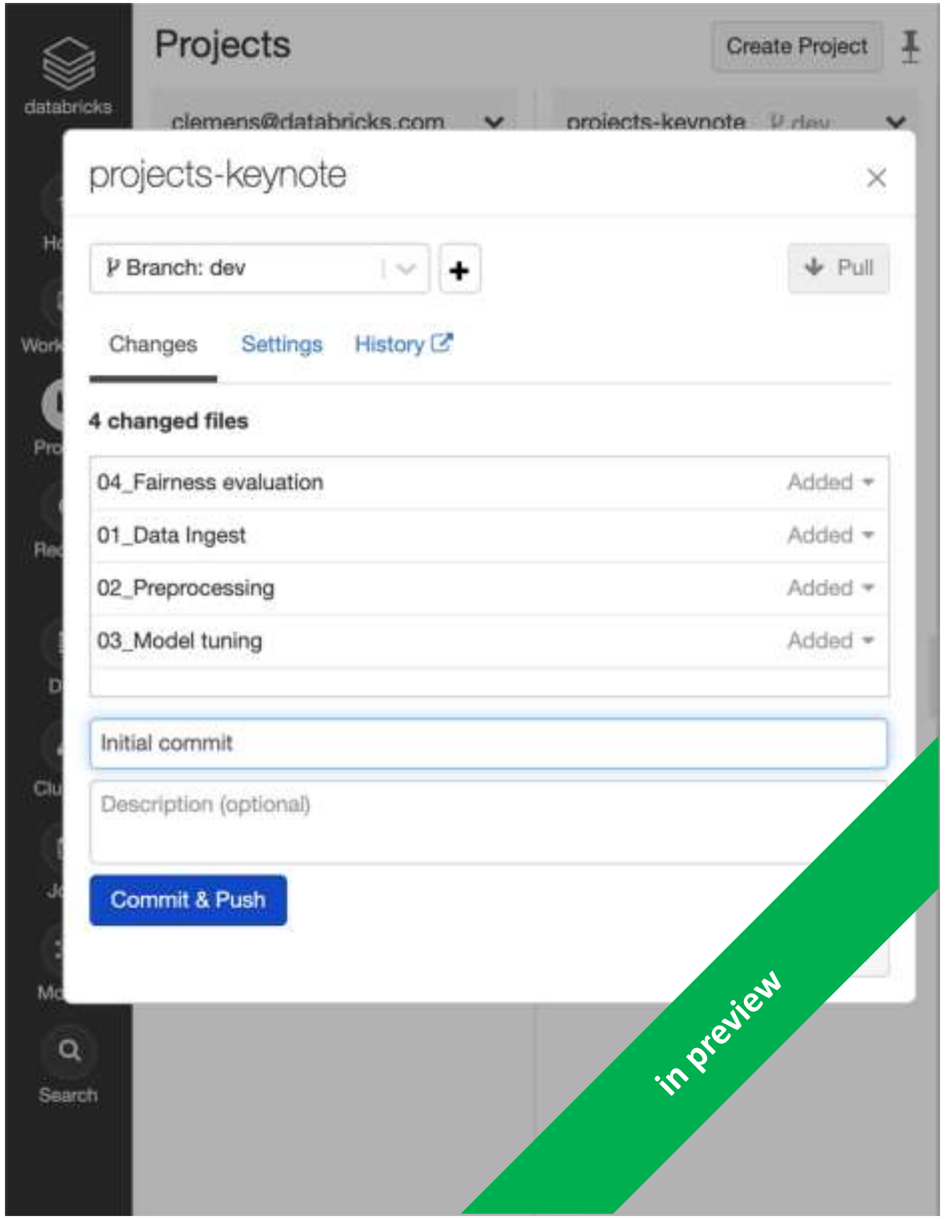


Create Project

Creating project in /Projects/clemens@databricks.com

Git repo URL: https://github.com/clemens-db/projects-keynote

Project name: projects-keynote



projects-keynote

4 changed files

04_Fairness evaluation	Added
01_Data Ingest	Added
02_Preprocessing	Added
03_Model tuning	Added

Initial commit

Description (optional)

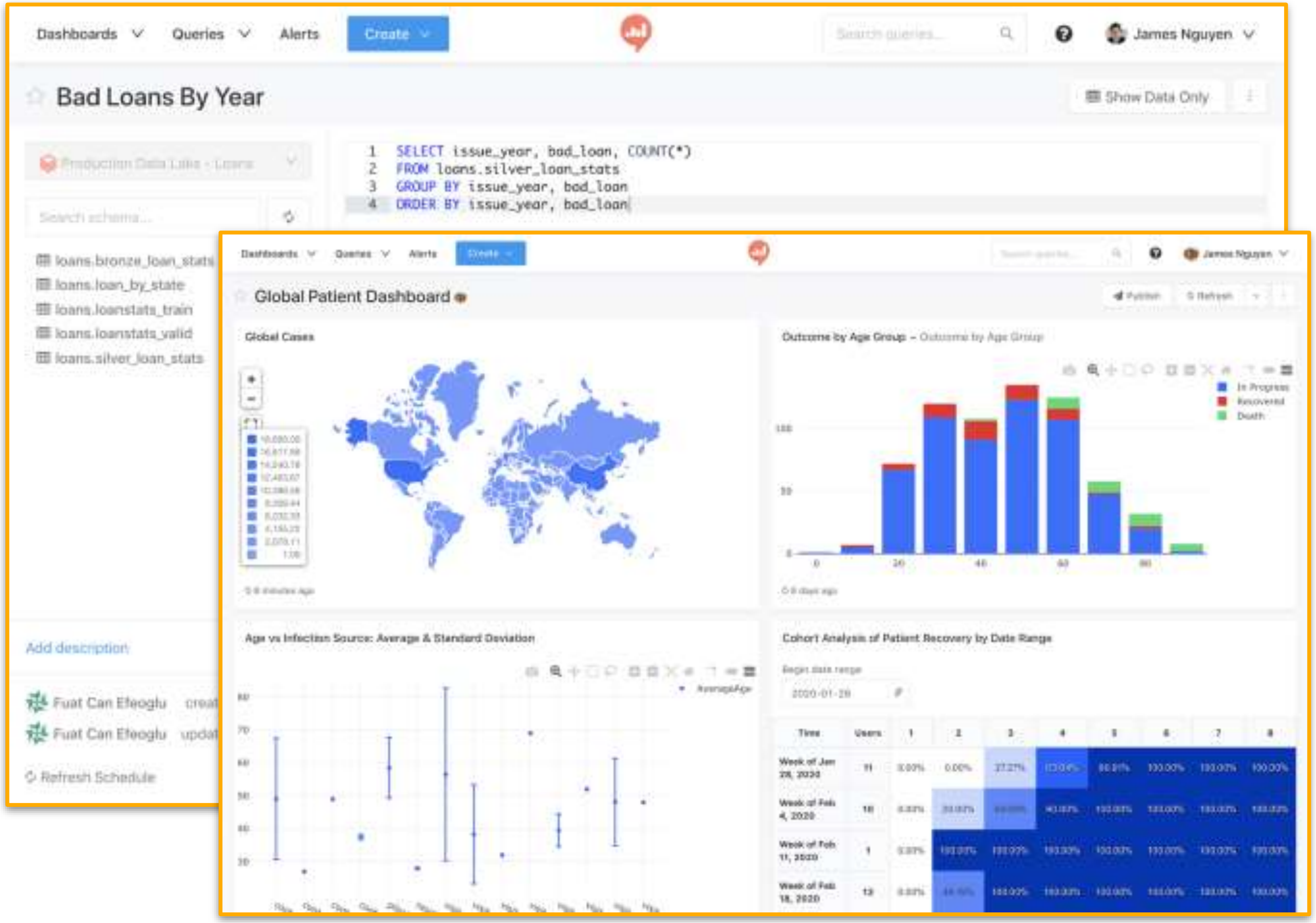
Azure Databricks - Redash

Azure Databricks – Redash: A home for SQL Users

Self-serve on the Data Lake!
Collaborative queries, dashboards & alerts on your data lake

Simple, SQL oriented UX
Analysts don’t need to understand or get exposed to notebooks or jobs

Ready-to-go
Tightly integrated with Databricks compute & security



Bad Loans By Year

```
SELECT issue_year, bad_loan, COUNT(*)
FROM loans.silver_loan_stats
GROUP BY issue_year, bad_loan
ORDER BY issue_year, bad_loan
```

Global Patient Dashboard

Age vs Infection Source: Average & Standard Deviation

Cohort Analysis of Patient Recovery by Date Range

Time	Users	1	2	3	4	5	6	7	8
Week of Jan 28, 2020	11	0.00%	0.00%	27.27%	[illegible]	[illegible]	100.00%	100.00%	100.00%
Week of Feb 4, 2020	10	0.00%	[illegible]	[illegible]	[illegible]	100.00%	100.00%	100.00%	100.00%
Week of Feb 11, 2020	1	0.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%
Week of Feb 18, 2020	13	0.00%	[illegible]	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%

Spark Interfaces

Resilient Distributed Dataset (RDD)

Spark RDD is a resilient, partitioned, distributed and immutable collection of data.

DataFrame

Distributed collection of data organized into named columns. It is conceptually equivalent to a table in a relational database or a data frame in R/Python, but with richer optimizations under the hood

Dataset

A Dataset is a strongly-typed, immutable collection of objects that are mapped to a relational schema.

An extension of the DataFrame API that provides a *type-safe, object-oriented programming interface*.

DataFrames

DataFrame is a distributed collection of data organized into named columns.

Conceptually equivalent to a table in a relational database or a data frame in R/Python, but with richer optimizations.

DataFrames can be constructed from a wide array of sources such as: structured data files, tables in Hive, external databases, or existing RDDs.

DataFrames are evaluated lazily, meaning, computation only happens when an action (e.g. display result, save output) is required.

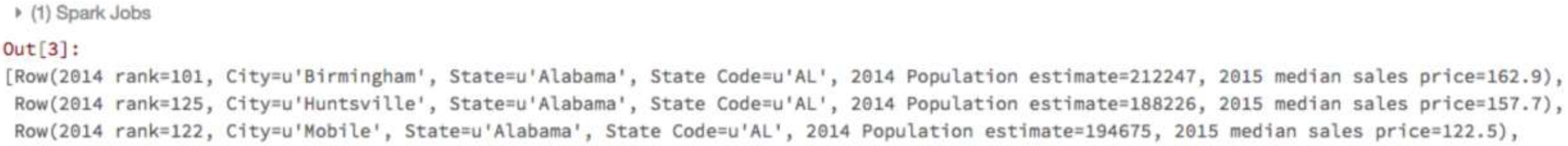
Loading Data in DataFrames

```
%python
# Use the Spark CSV datasource with options specifying:
# - First line of file is a header
# - Automatically infer the schema of the data

data = spark.read.format("csv") \
  .option("header", "true") \
  .option("inferSchema", "true") \
  .load("/databricks-datasets/samples/population-vs-price/data_geo.csv")

data.cache() # Cache data for faster reuse
data = data.dropna() # drop rows with missing values
```

Viewing DataFrames

Using Spark Command **take()** to view raw records

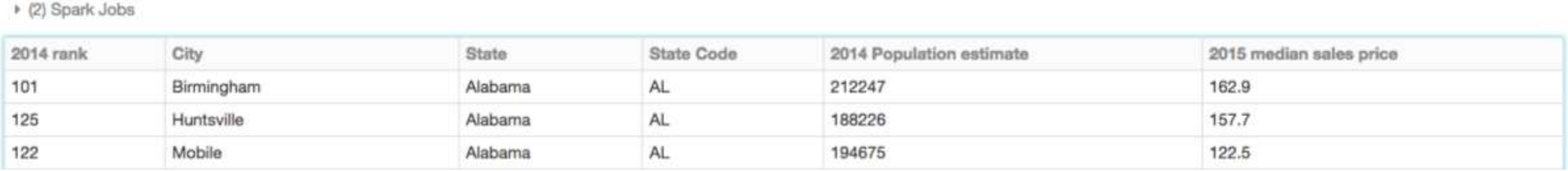
```
%python data.take(10) #view 10 records of DataFrame
```



```
Out[3]:
[Row(2014 rank=101, City=u'Birmingham', State=u'Alabama', State Code=u'AL', 2014 Population estimate=212247, 2015 median sales price=162.9),
 Row(2014 rank=125, City=u'Huntsville', State=u'Alabama', State Code=u'AL', 2014 Population estimate=188226, 2015 median sales price=157.7),
 Row(2014 rank=122, City=u'Mobile', State=u'Alabama', State Code=u'AL', 2014 Population estimate=194675, 2015 median sales price=122.5),
```

Using **display()** to view in tabular mode

```
%python display(data)
```



2014 rank	City	State	State Code	2014 Population estimate	2015 median sales price
101	Birmingham	Alabama	AL	212247	162.9
125	Huntsville	Alabama	AL	188226	157.7
122	Mobile	Alabama	AL	194675	122.5

Datasets

The Apache Spark Dataset API provides a type-safe, object-oriented programming interface

DataFrame is an alias for an untyped **Dataset [Row]**

Datasets provide compile-time type safety

The Dataset API also offers high-level domain-specific language operations

Load Sample Data in Dataset

Read a data file from an external data source.

```
val df = spark.read.json("/databricks-datasets/samples/people/people.json")
```

At the time of reading the JSON file, Spark does not know the structure of your data.

It doesn’t know how you want to organize your data into a typed-specific JVM object.

It attempts to infer the schema from the JSON file

This creates a `DataFrame = Dataset[Row]` of generic Row objects.

Viewing Dataset

Viewing data in tabular mode using display()

```
// display the dataset table just read in from the JSON file display(ds)
```

Using standard Spark commands like take(), foreach() and println() API calls

```
// Using the standard Spark commands, take() and foreach(), print the first
// 10 rows of the Datasets.
ds.take(10).foreach(println(_))
```

Learn More

Azure Databricks Overview -
https://aka.ms/LearnAzureDatabricks

Delta Engine Documentation -
https://aka.ms/DeltaEngineDocs

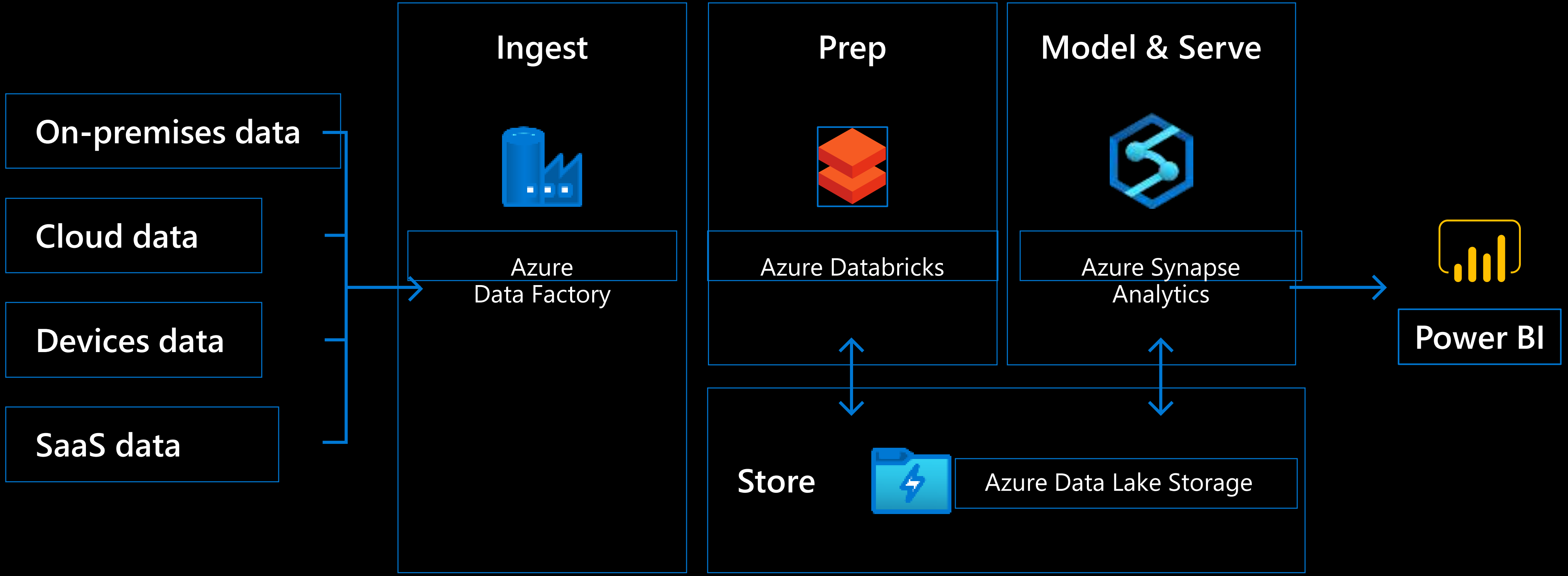
Next Gen Data Science Workspace -
https://aka.ms/AzureDatabricks_DataScienceWorkspace

Spark and AI Summit 2020 content -
https://aka.ms/SparkAISummit2020

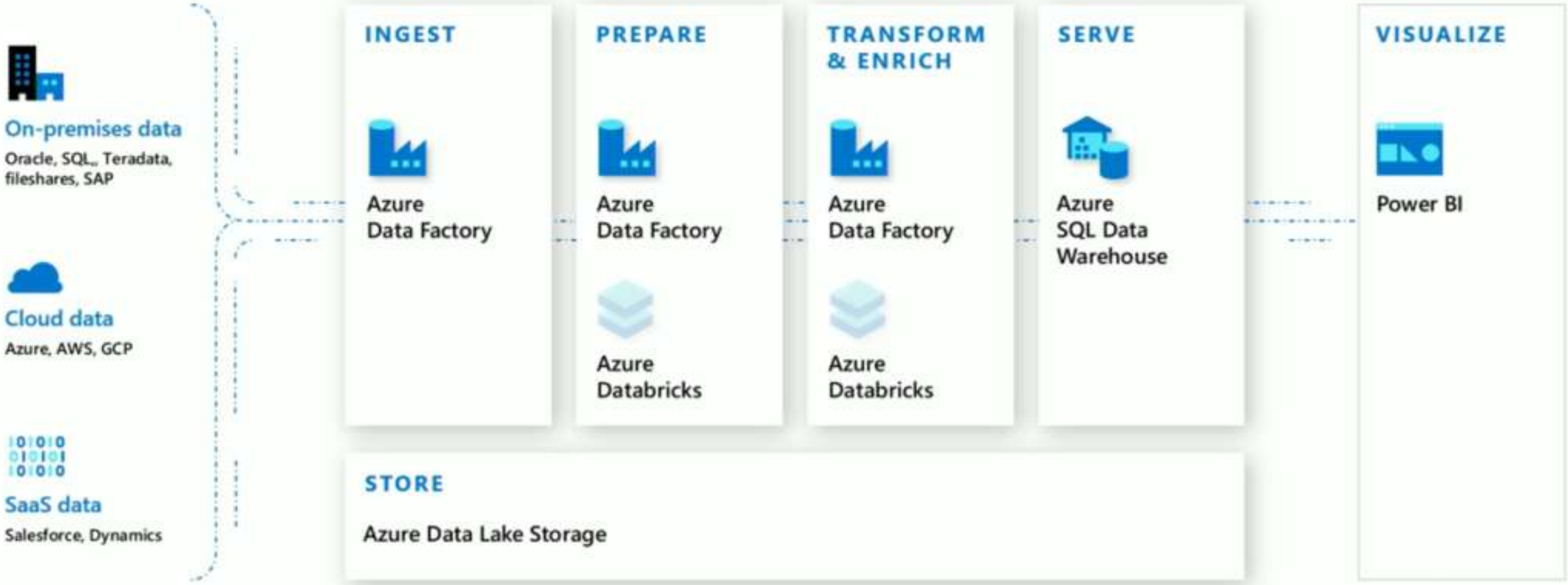
Microsoft

Azure Synapse Analytics

Azure Analytics



Modern Data Warehouse



On-premises data

Oracle, SQL., Teradata,
fileshares, SAP

Cloud data

Azure, AWS, GCP

SaaS data

Salesforce, Dynamics

INGEST

Azure
Data Factory

PREPARE

Azure
Data Factory

Azure
Databricks

TRANSFORM
& ENRICH

Azure
Data Factory

Azure
Databricks

SERVE

Azure
SQL Data
Warehouse

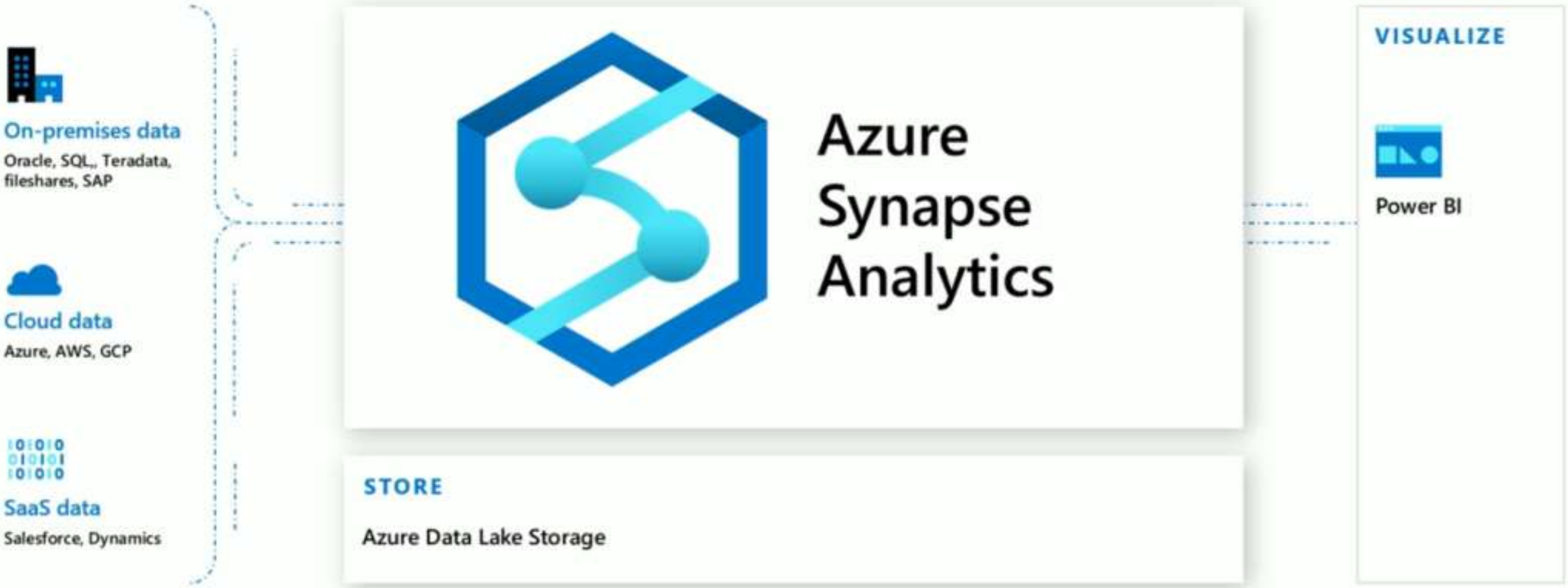
VISUALIZE

Power BI

STORE

Azure Data Lake Storage

Azure Synapse Analytics - Data Lakehouse



On-premises data

Oracle, SQL,, Teradata, fileshares, SAP

Cloud data

Azure, AWS, GCP

SaaS data

Salesforce, Dynamics

Azure Synapse Analytics

STORE

Azure Data Lake Storage

VISUALIZE

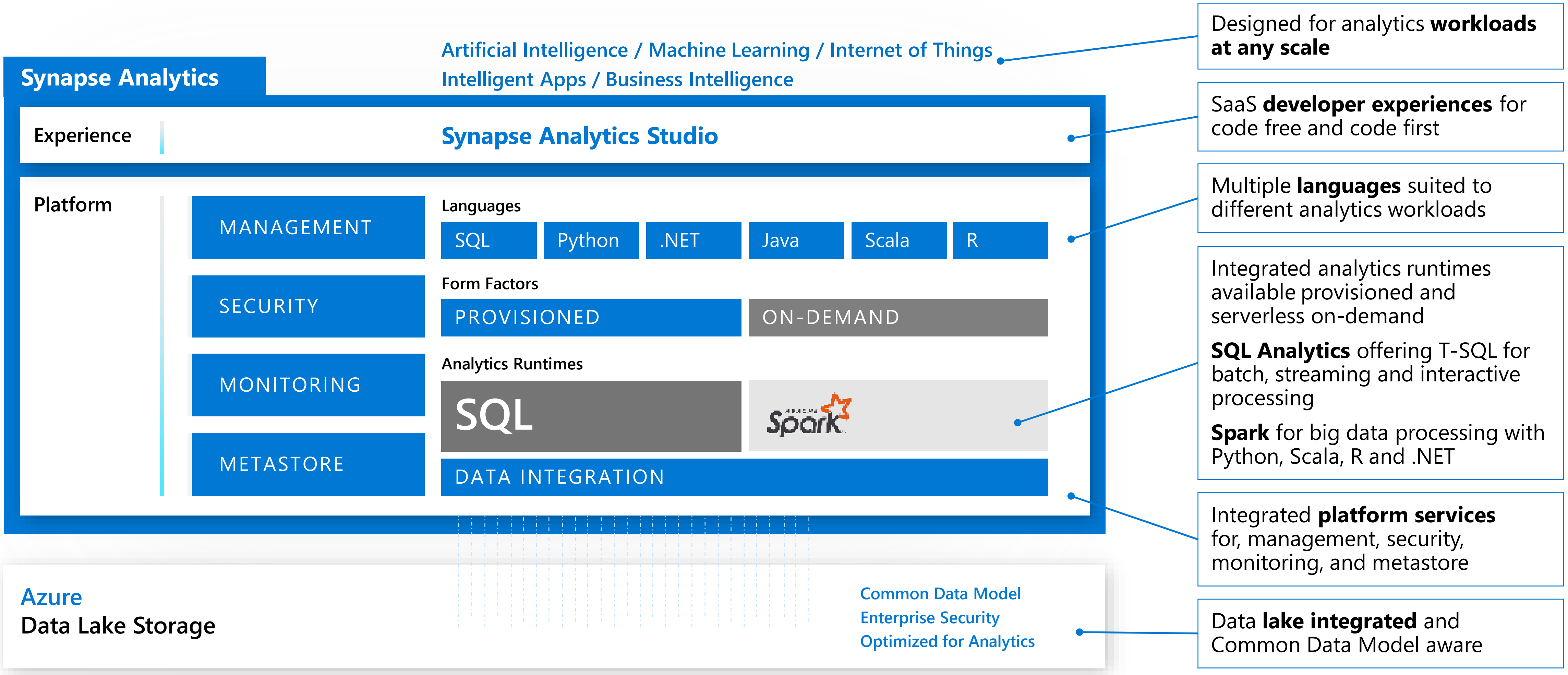
Power BI

Microsoft

Azure Synapse Analytics

Azure Synapse Analytics

Integrated data platform for BI, AI and continuous intelligence



Designed for analytics **workloads at any scale**

SaaS **developer experiences** for code free and code first

Multiple **languages** suited to different analytics workloads

Integrated analytics runtimes available provisioned and serverless on-demand

SQL Analytics offering T-SQL for batch, streaming and interactive processing

Spark for big data processing with Python, Scala, R and .NET

Integrated **platform services** for, management, security, monitoring, and metastore

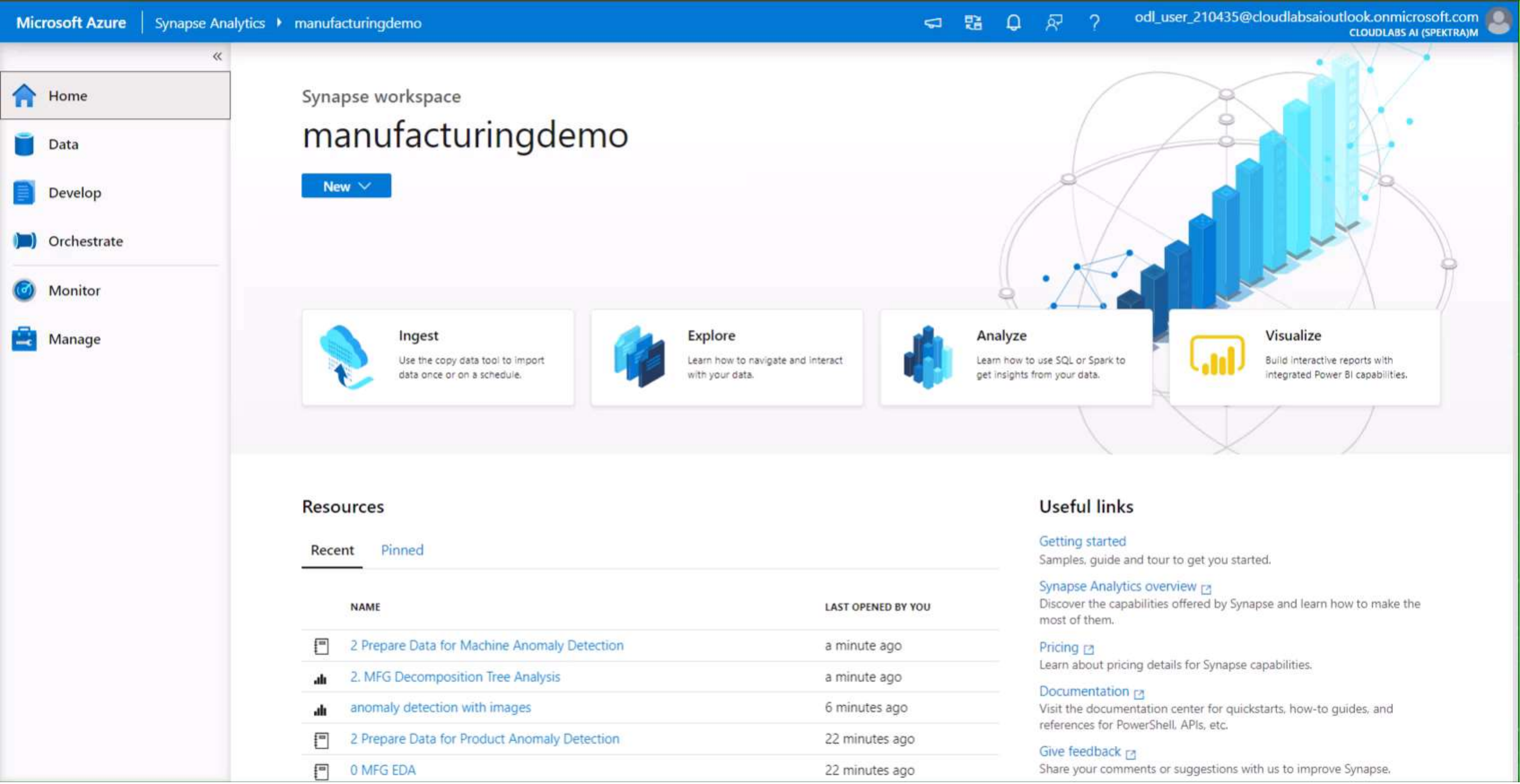
Data **lake integrated** and Common Data Model aware

Microsoft

Azure Synapse Analytics

Studio

DEMO



Synapse workspace

manufacturingdemo

Ingest

Use the copy data tool to import data once or on a schedule.

Explore

Learn how to navigate and interact with your data.

Analyze

Learn how to use SQL or Spark to get insights from your data.

Visualize

Build interactive reports with integrated Power BI capabilities.

Resources

Recent | Pinned

NAME	LAST OPENED BY YOU
2 Prepare Data for Machine Anomaly Detection	a minute ago
2. MFG Decomposition Tree Analysis	a minute ago
anomaly detection with images	6 minutes ago
2 Prepare Data for Product Anomaly Detection	22 minutes ago
0 MFG EDA	22 minutes ago

Useful links

Getting started

Samples, guide and tour to get you started.

Synapse Analytics overview

Discover the capabilities offered by Synapse and learn how to make the most of them.

Pricing

Learn about pricing details for Synapse capabilities.

Documentation

Visit the documentation center for quickstarts, how-to guides, and references for PowerShell, APIs, etc.

Give feedback

Share your comments or suggestions with us to improve Synapse.

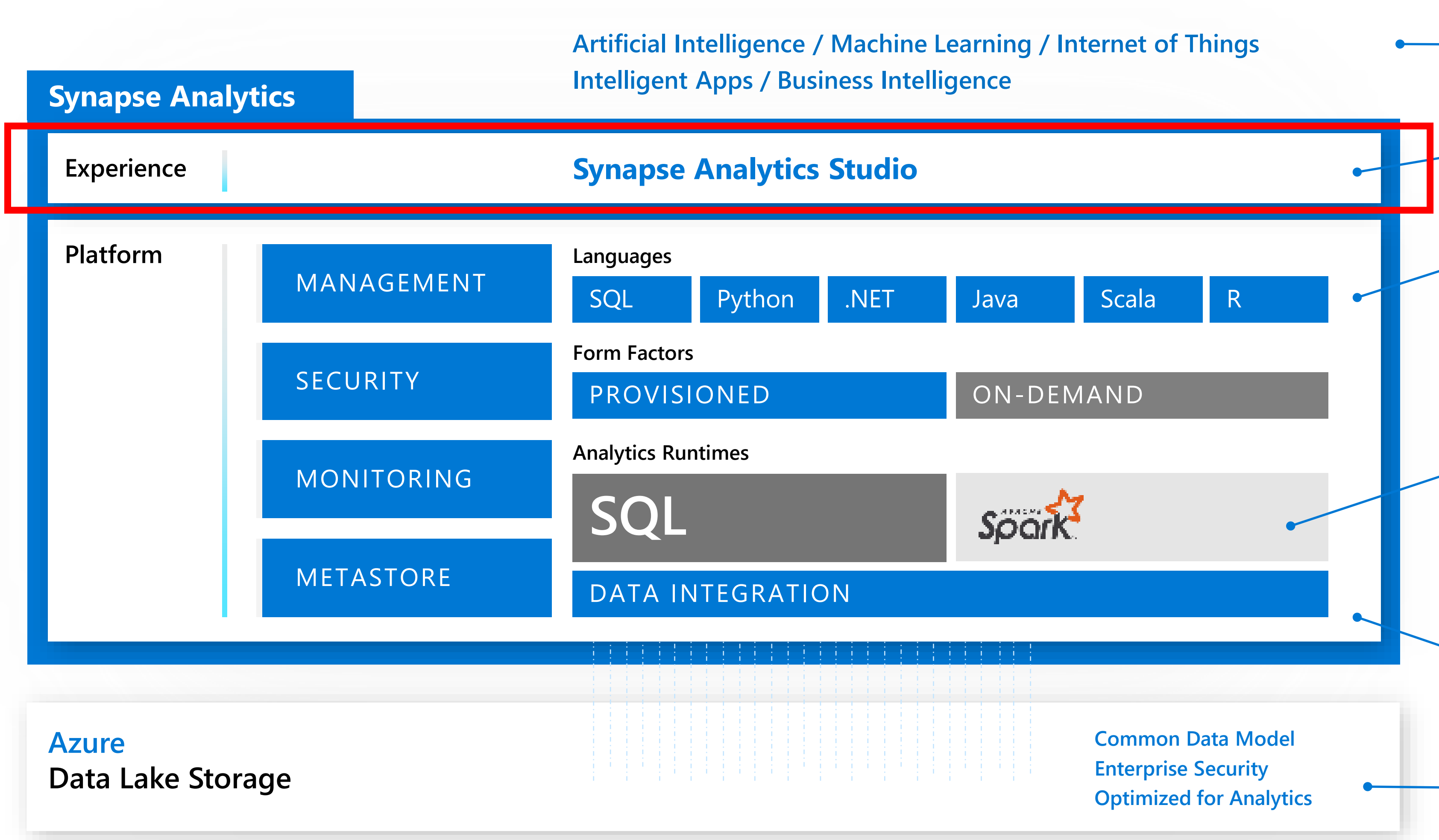
Microsoft

Azure Synapse Analytics

Studio

Azure Synapse Analytics

Integrated data platform for BI, AI and continuous intelligence



Designed for analytics **workloads at any scale**

SaaS **developer experiences** for code free and code first

Multiple **languages** suited to different analytics workloads

Integrated analytics runtimes available provisioned and serverless on-demand

SQL Analytics offering T-SQL for batch, streaming and interactive processing

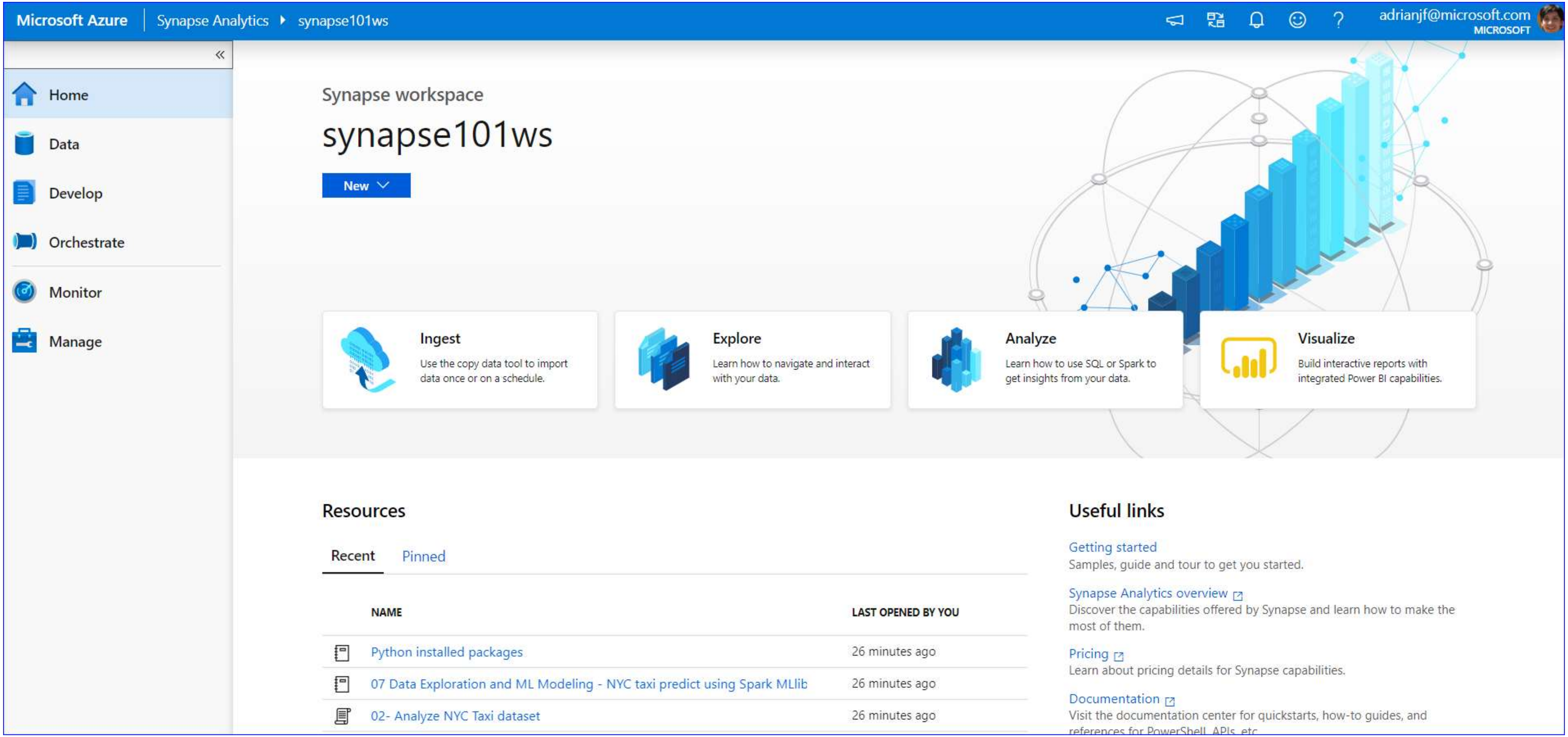
Spark for big data processing with Python, Scala, R and .NET

Integrated **platform services** for, management, security, monitoring, and metastore

Data **lake integrated** and Common Data Model aware

Synapse Studio

https://web.azuresynapse.net



Microsoft Azure | Synapse Analytics ▸ synapse101ws

adrianjf@microsoft.com
MICROSOFT

«

- Home
- Data
- Develop
- Orchestrate
- Monitor
- Manage

Synapse workspace

synapse101ws

New ⌄

Ingest
Use the copy data tool to import data once or on a schedule.

Explore
Learn how to navigate and interact with your data.

Analyze
Learn how to use SQL or Spark to get insights from your data.

Visualize
Build interactive reports with integrated Power BI capabilities.

Resources

Recent | Pinned

NAME	LAST OPENED BY YOU
Python installed packages	26 minutes ago
07 Data Exploration and ML Modeling - NYC taxi predict using Spark MLlib	26 minutes ago
02- Analyze NYC Taxi dataset	26 minutes ago

Useful links

Getting started
Samples, guide and tour to get you started.

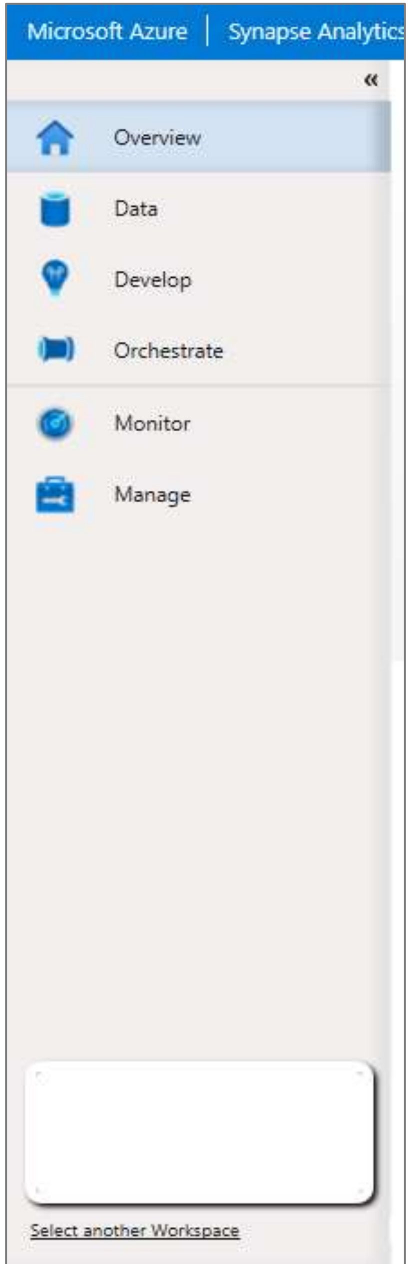
Synapse Analytics overview
Discover the capabilities offered by Synapse and learn how to make the most of them.

Pricing
Learn about pricing details for Synapse capabilities.

Documentation
Visit the documentation center for quickstarts, how-to guides, and references for PowerShell, APIs, etc.

Synapse Studio

https://web.azuresynapse.net



- **Data**: Shows the available data sources available to the workspace. These can exist internally in the workspace (such as a SQL compute database or a Spark database), or externally (such as a Data Lake Store Gen2, or Azure Blob Storage account)
- **Develop**: Shows the different objects used to query or operate with the data, such as SQL scripts, notebooks, data flows, Spark job definitions, Power BI, etc
- **Orchestrate**: Shows the objects used to automate analytics processes (such as pipelines, datasets, etc.)
- **Monitor**: Shows metrics for pipeline runs, trigger runs, integration runtimes, and spark applications
- **Manage**: Create linked services, pipeline triggers, integration runtimes, and manage access to Synapse

Microsoft

Azure Synapse Analytics

Spark

Azure Synapse Apache Spark - Summary

- Apache Spark 2.4 derivation
  - Linux Foundation Delta Lake 0.6.0 support
  - Apache Spark in Azure Synapse includes .NET Core 3.1
  - Python 3.6.1 + Anacondas support
- Operating System version
  - Apache Spark in Azure Synapse runs on Ubuntu 16.04.
- Tightly coupled to other Azure Synapse services
  - Integrated security and sign on
  - Integrated Metadata
  - Integrated and simplified provisioning
  - Integrated UX including nteract based notebooks
  - Fast load of SQL Analytics pools

- Core scenarios
  - Data Prep/Data Engineering/ETL
  - Machine Learning via Spark ML and Azure ML integration
  - Extensible through library management
- Efficient resource utilization
  - Fast Start
  - Auto scale (up and down)
  - Auto pause
  - Min cluster size of 3 nodes
  - Max cluster size 200 nodes
- Multi Language Support
  - .Net (C#), PySpark, Scala, Spark SQL, Java

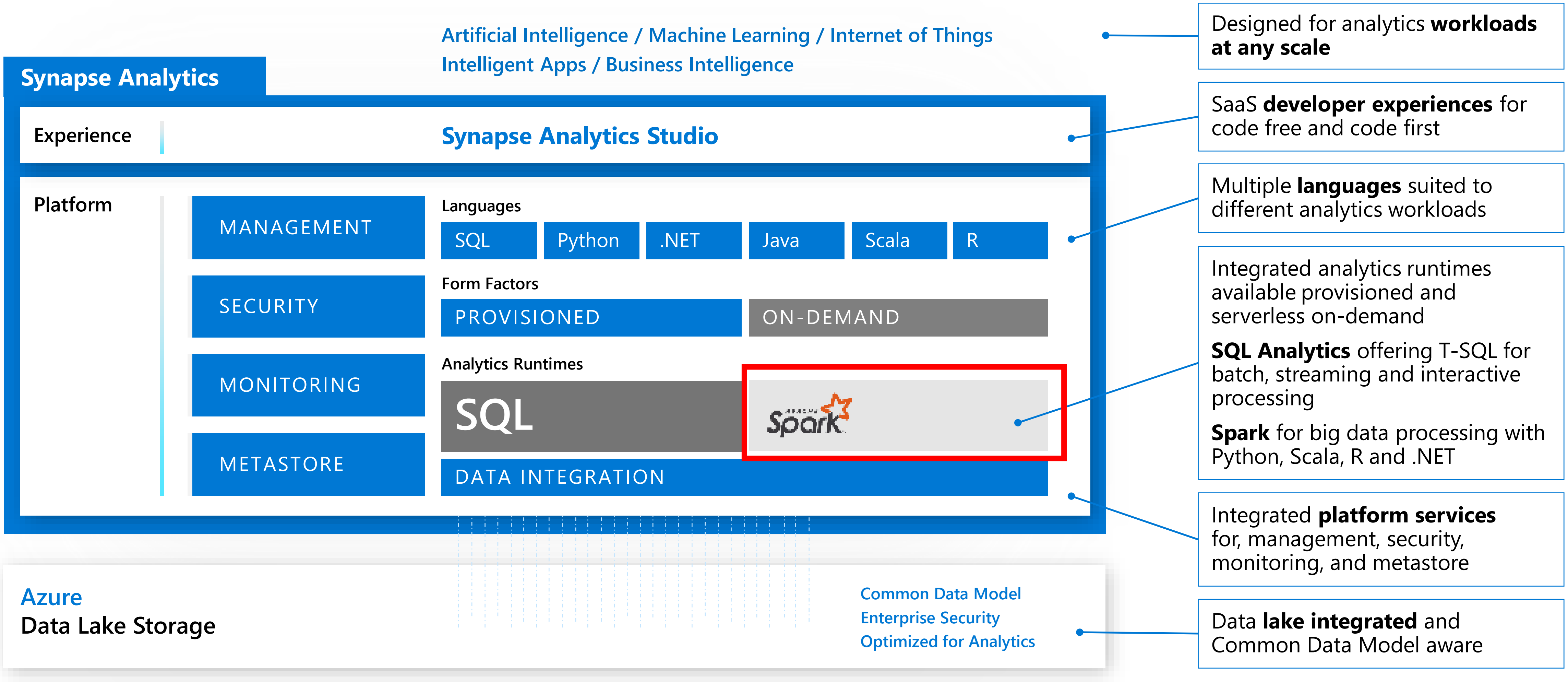
https://docs.microsoft.com/en-us/azure/synapse-analytics/spark/apache-spark-version-support

What is Delta Lake?

- OSS storage layer for Spark
- Provides:
  - ACID transactions
  - History of changed
  - Time travel in data history
  - Schema evolution
  - …

Azure Synapse Analytics

Integrated data platform for BI, AI and continuous intelligence



Artificial Intelligence / Machine Learning / Internet of Things
Intelligent Apps / Business Intelligence

Synapse Analytics

Experience

Synapse Analytics Studio

Platform

MANAGEMENT

SECURITY

MONITORING

METASTORE

Languages

SQL | Python | .NET | Java | Scala | R

Form Factors

PROVISIONED | ON-DEMAND

Analytics Runtimes

SQL | Spark

DATA INTEGRATION

Azure
Data Lake Storage

Common Data Model
Enterprise Security
Optimized for Analytics

Designed for analytics **workloads at any scale**

SaaS **developer experiences** for code free and code first

Multiple **languages** suited to different analytics workloads

Integrated analytics runtimes available provisioned and serverless on-demand

SQL Analytics offering T-SQL for batch, streaming and interactive processing

Spark for big data processing with Python, Scala, R and .NET

Integrated **platform services** for, management, security, monitoring, and metastore

Data **lake integrated** and Common Data Model aware

Languages

Overview

Supports multiple languages to develop notebook

- pySpark (Python)
- Spark (Scala)
- SparkSQL
- .NET for Apache Spark (C#)

Benefits

Allows to write multiple languages in one notebook

Using Magic command **%%** <language>

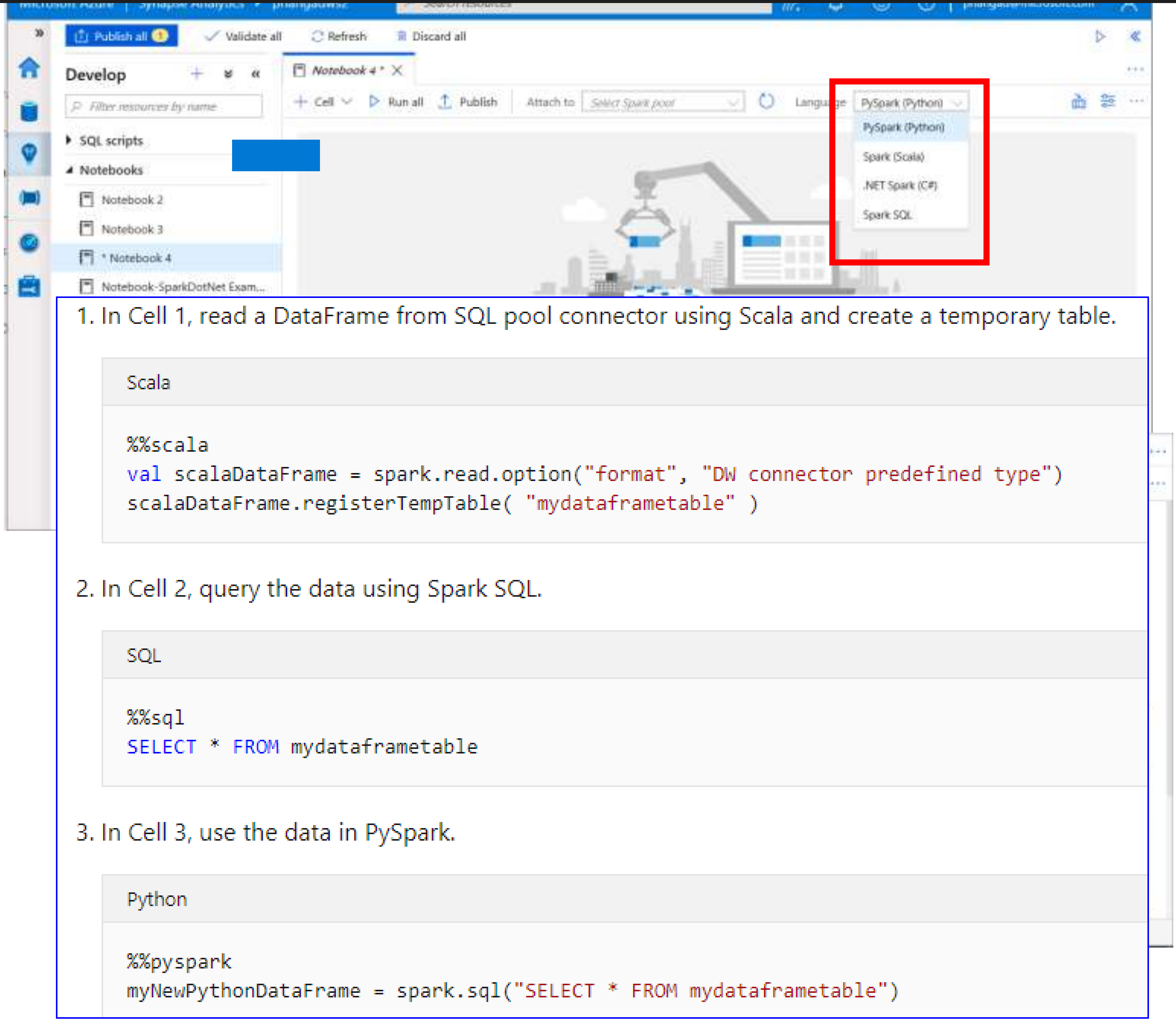
%%pyspark

%%spark

%%sql

%%csharp

use of **temporary tables** across **languages**



1. In Cell 1, read a DataFrame from SQL pool connector using Scala and create a temporary table.

Scala

```scala
%%scala
val scalaDataFrame = spark.read.option("format", "DW connector predefined type")
scalaDataFrame.registerTempTable( "mydataframetable" )
```

2. In Cell 2, query the data using Spark SQL.

SQL

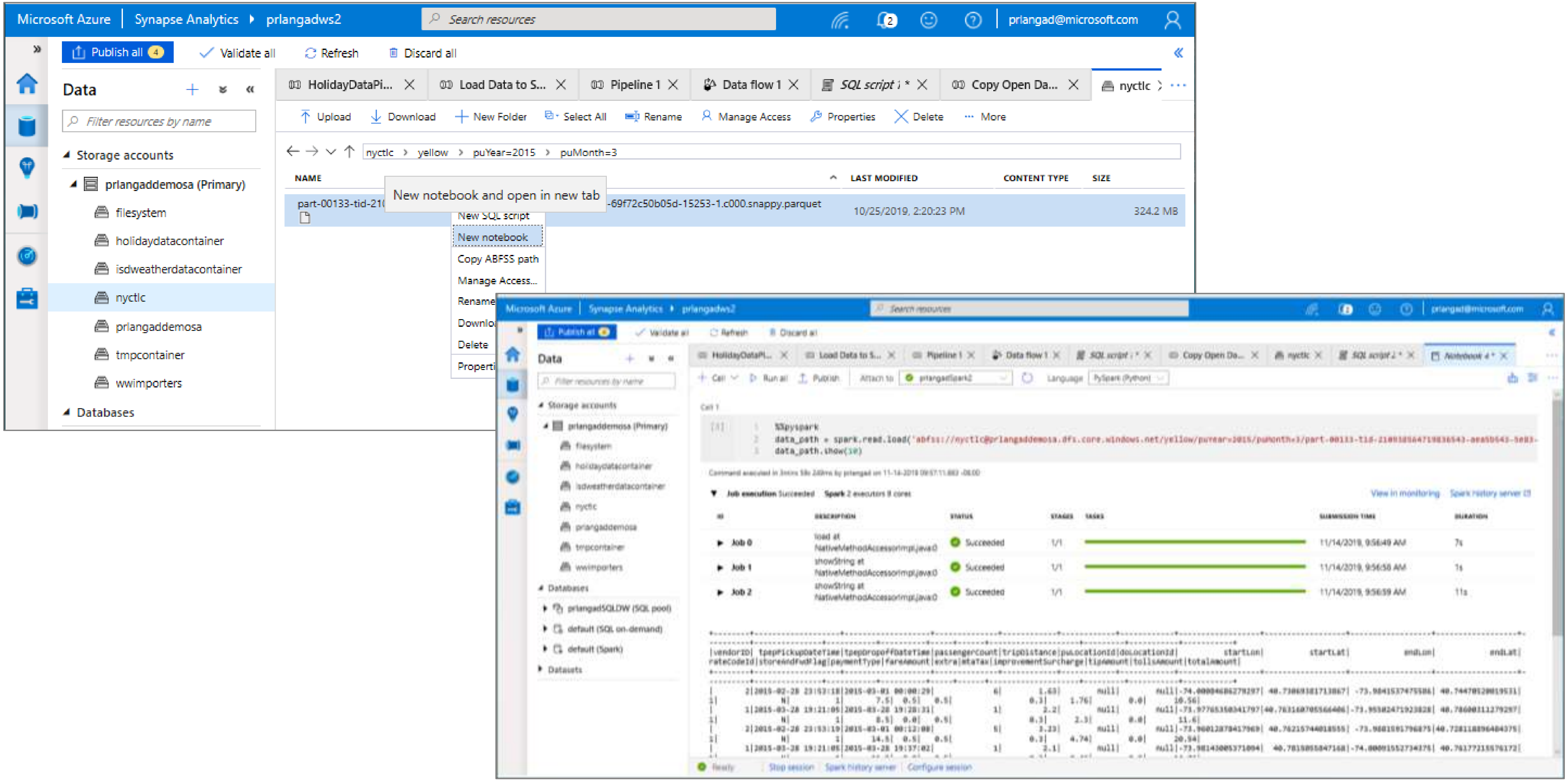
```sql
%%sql
SELECT * FROM mydataframetable
```

3. In Cell 3, use the data in PySpark.

Python

```python
%%pyspark
myNewPythonDataFrame = spark.sql("SELECT * FROM mydataframetable")
```

Create Notebook on files in storage



```
%%pyspark
data_path = spark.read.load('abfss://nyctlc@prlangaddemosa.dfs.core.windows.net/yellow/puYear=2015/puMonth=3/part-00133-tid-2109385647198836543-aea5b543-5e83-
data_path.show(10)
```

Command executed in 3mins 58s 248ms by prlangad on 11-14-2019 09:57:11.883 -08:00

Job execution Succeeded   Spark 2 executors 8 cores   View in monitoring   Spark history server

ID	DESCRIPTION	STATUS	STAGES	TASKS	SUBMISSION TIME	DURATION
Job 0	load at NativeMethodAccessorImpl.java:0	Succeeded	1/1		11/14/2019, 9:56:49 AM	7s
Job 1	showString at NativeMethodAccessorImpl.java:0	Succeeded	1/1		11/14/2019, 9:56:56 AM	1s
Job 2	showString at NativeMethodAccessorImpl.java:0	Succeeded	1/1		11/14/2019, 9:56:59 AM	11s

```
|vendorID| tpepPickupDateTime|tpepDropoffDateTime|passengerCount|tripDistance|puLocationId|doLocationId|          startLon|          startLat|            endLon|            endLat|
rateCodeId|storeAndFwdFlag|paymentType|fareAmount|extra|mtaTax|improvementSurcharge|tipAmount|tollsAmount|totalAmount|
|       2|2015-02-28 23:53:18|2015-03-01 00:00:29|             6|        1.63|        null|        null|-74.00004686279297| 40.73069381713867| -73.98415374755860| 40.74470520019531|
1|              N|          1|       7.5|  0.5|   0.5|                 0.3|     1.76|        0.0|      10.56|
|       1|2015-03-28 19:21:05|2015-03-28 19:28:31|             1|         2.2|        null|        null|-73.97765350341797|40.76316070556640|-73.95502471923828| 40.78600311279297|
1|              N|          1|       8.5|  0.0|   0.5|                 0.3|      2.3|        0.0|       11.6|
|       2|2015-02-28 23:53:19|2015-03-01 00:12:08|             5|        3.23|        null|        null|-73.96012878417969| 40.76215744018555| -73.98815917968750|40.72811889648437|
1|              N|          1|      14.5|  0.5|   0.5|                 0.3|     4.74|        0.0|      20.54|
|       1|2015-03-28 19:21:05|2015-03-28 19:37:02|             1|         2.1|        null|        null|-73.98143005371094| 40.78150558471680|-74.00091552734375| 40.76177215576172|
```

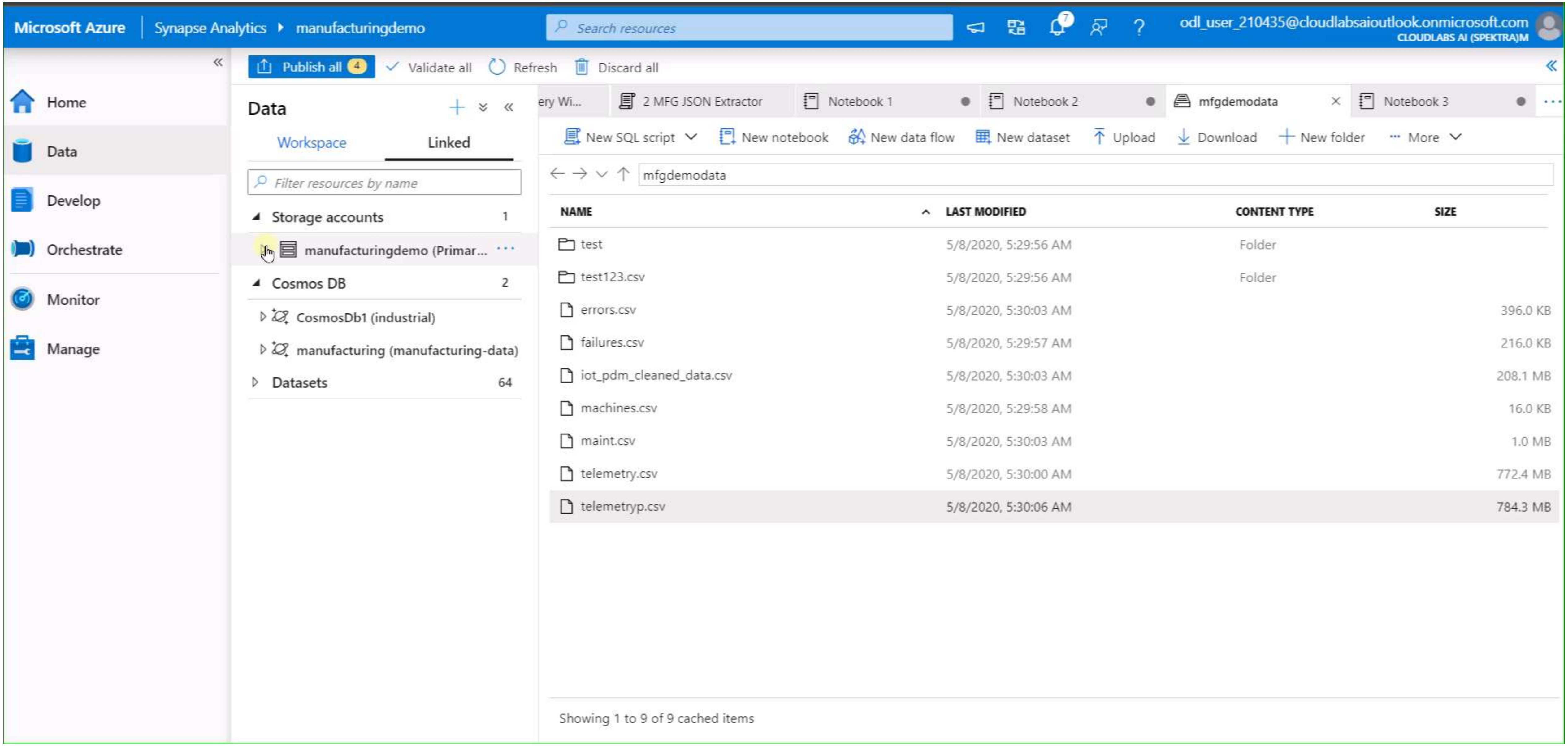
Microsoft

Azure Synapse Analytics

Notebook on files in storage (CSV, Parket, JSON)

Demo

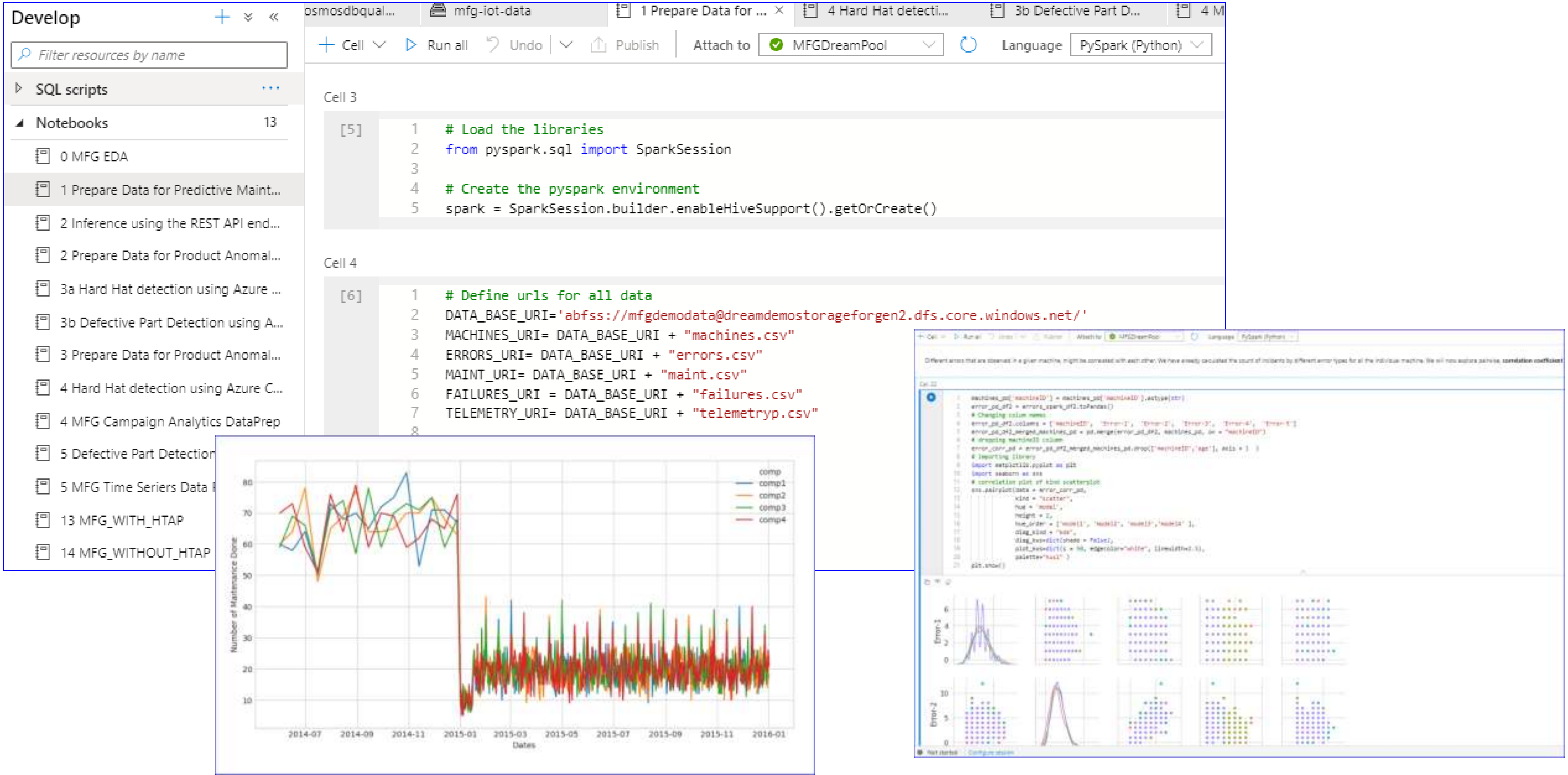
Create Notebook on files in storage



NAME	LAST MODIFIED	CONTENT TYPE	SIZE
test	5/8/2020, 5:29:56 AM	Folder	
test123.csv	5/8/2020, 5:29:56 AM	Folder	
errors.csv	5/8/2020, 5:30:03 AM		396.0 KB
failures.csv	5/8/2020, 5:29:57 AM		216.0 KB
iot_pdm_cleaned_data.csv	5/8/2020, 5:30:03 AM		208.1 MB
machines.csv	5/8/2020, 5:29:58 AM		16.0 KB
maint.csv	5/8/2020, 5:30:03 AM		1.0 MB
telemetry.csv	5/8/2020, 5:30:00 AM		772.4 MB
telemetryp.csv	5/8/2020, 5:30:06 AM		784.3 MB

Showing 1 to 9 of 9 cached items

Languages – PySpark (Python)



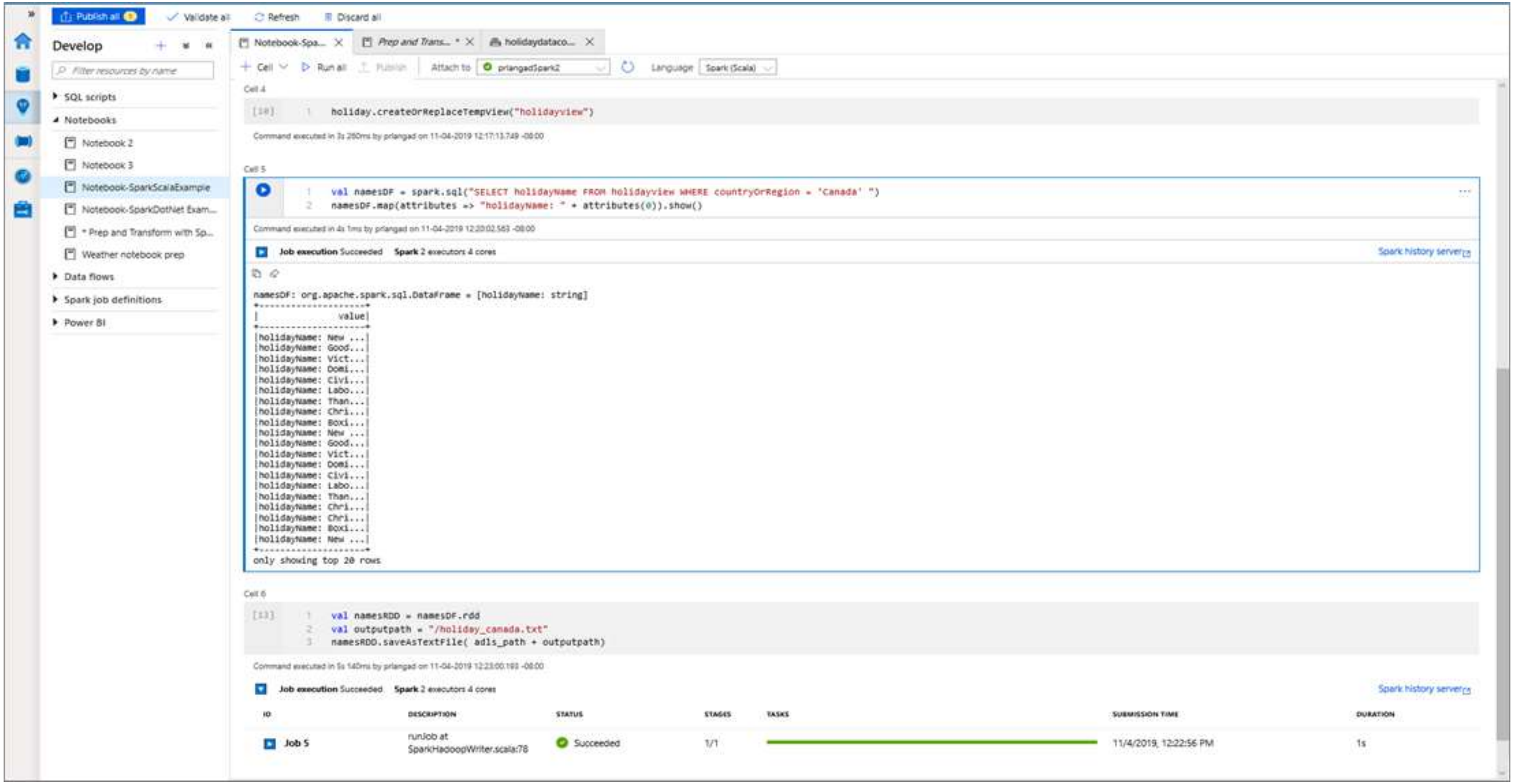
```python
# Load the libraries
from pyspark.sql import SparkSession

# Create the pyspark environment
spark = SparkSession.builder.enableHiveSupport().getOrCreate()
```

```python
# Define urls for all data
DATA_BASE_URI='abfss://mfgdemodata@dreamdemostorageforgen2.dfs.core.windows.net/'
MACHINES_URI= DATA_BASE_URI + "machines.csv"
ERRORS_URI= DATA_BASE_URI + "errors.csv"
MAINT_URI= DATA_BASE_URI + "maint.csv"
FAILURES_URI = DATA_BASE_URI + "failures.csv"
TELEMETRY_URI= DATA_BASE_URI + "telemetryp.csv"
```

Different errors that are observed in a given machine, might be correlated with each other. We have already calculated the count of incidents by different error types for all the individual machine. We will now explore pairwise **correlation coefficient**

```python
machines_pd['machineID'] = machines_pd['machineID'].astype(str)
error_pd_df2 = errors_spark_df2.toPandas()
# Changing column names
error_pd_df2.columns = ['machineID', 'Error-1', 'Error-2', 'Error-3', 'Error-4', 'Error-5']
error_pd_df2_merged_machines_pd = pd.merge(error_pd_df2, machines_pd, on = "machineID")
# dropping machineID column
error_corr_pd = error_pd_df2_merged_machines_pd.drop(['machineID','age'], axis = 1 )
# importing library
import matplotlib.pyplot as plt
import seaborn as sns
# correlation plot of kind scatterplot
sns.pairplot(data = error_corr_pd,
             kind = "scatter",
             hue = 'model',
             height = 2,
             hue_order = ['model1', 'model2', 'model3','model4' ],
             diag_kind = 'kde',
             diag_kws=dict(shade = False),
             plot_kws=dict(s = 50, edgecolor="white", linewidth=2.5),
             palette='husl' )
plt.show()
```

Languages – Spark (Scala)



Attach to: priangadSpark2

Language: Spark (Scala)

Cell 4

```
[10]  holiday.createOrReplaceTempView("holidayview")
```

Command executed in 3s 280ms by priangad on 11-04-2019 12:17:13.749 -08:00

Cell 5

```
val namesDF = spark.sql("SELECT holidayName FROM holidayview WHERE countryOrRegion = 'Canada' ")
namesDF.map(attributes => "holidayName: " + attributes(0)).show()
```

Command executed in 4s 1ms by priangad on 11-04-2019 12:20:02.563 -08:00

Job execution Succeeded **Spark** 2 executors 4 cores

```
namesDF: org.apache.spark.sql.DataFrame = [holidayName: string]
+--------------------+
|               value|
+--------------------+
|holidayName: New ...|
|holidayName: Good...|
|holidayName: Vict...|
|holidayName: Domi...|
|holidayName: Civi...|
|holidayName: Labo...|
|holidayName: Than...|
|holidayName: Chri...|
|holidayName: Boxi...|
|holidayName: New ...|
|holidayName: Good...|
|holidayName: Vict...|
|holidayName: Domi...|
|holidayName: Civi...|
|holidayName: Labo...|
|holidayName: Than...|
|holidayName: Chri...|
|holidayName: Chri...|
|holidayName: Boxi...|
|holidayName: New ...|
+--------------------+
only showing top 20 rows
```

Cell 6

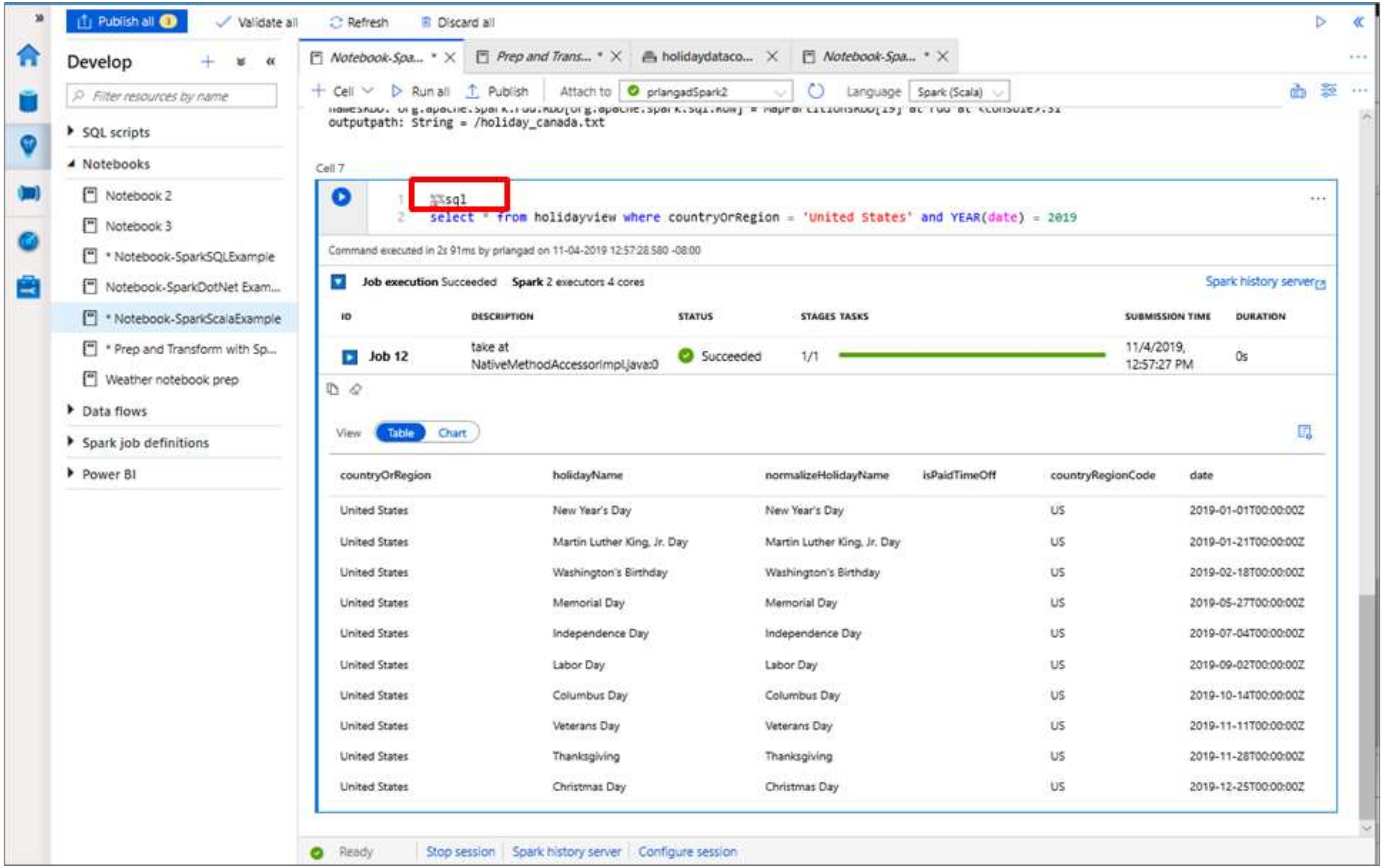
```
[13]  val namesRDD = namesDF.rdd
      val outputpath = "/holiday_canada.txt"
      namesRDD.saveAsTextFile( adls_path + outputpath)
```

Command executed in 5s 140ms by priangad on 11-04-2019 12:23:00.193 -08:00

Job execution Succeeded **Spark** 2 executors 4 cores

ID	DESCRIPTION	STATUS	STAGES	TASKS	SUBMISSION TIME	DURATION
Job 5	runJob at SparkHadoopWriter.scala:78	Succeeded	1/1		11/4/2019, 12:22:56 PM	1s

Languages – Spark SQL



```
namesRDD: org.apache.spark.rdd.RDD[org.apache.spark.sql.Row] = MapPartitionsRDD[19] at rdd at <console>:31
outputpath: String = /holiday_canada.txt
```

Cell 7

```
%%sql
select * from holidayview where countryOrRegion = 'United States' and YEAR(date) = 2019
```

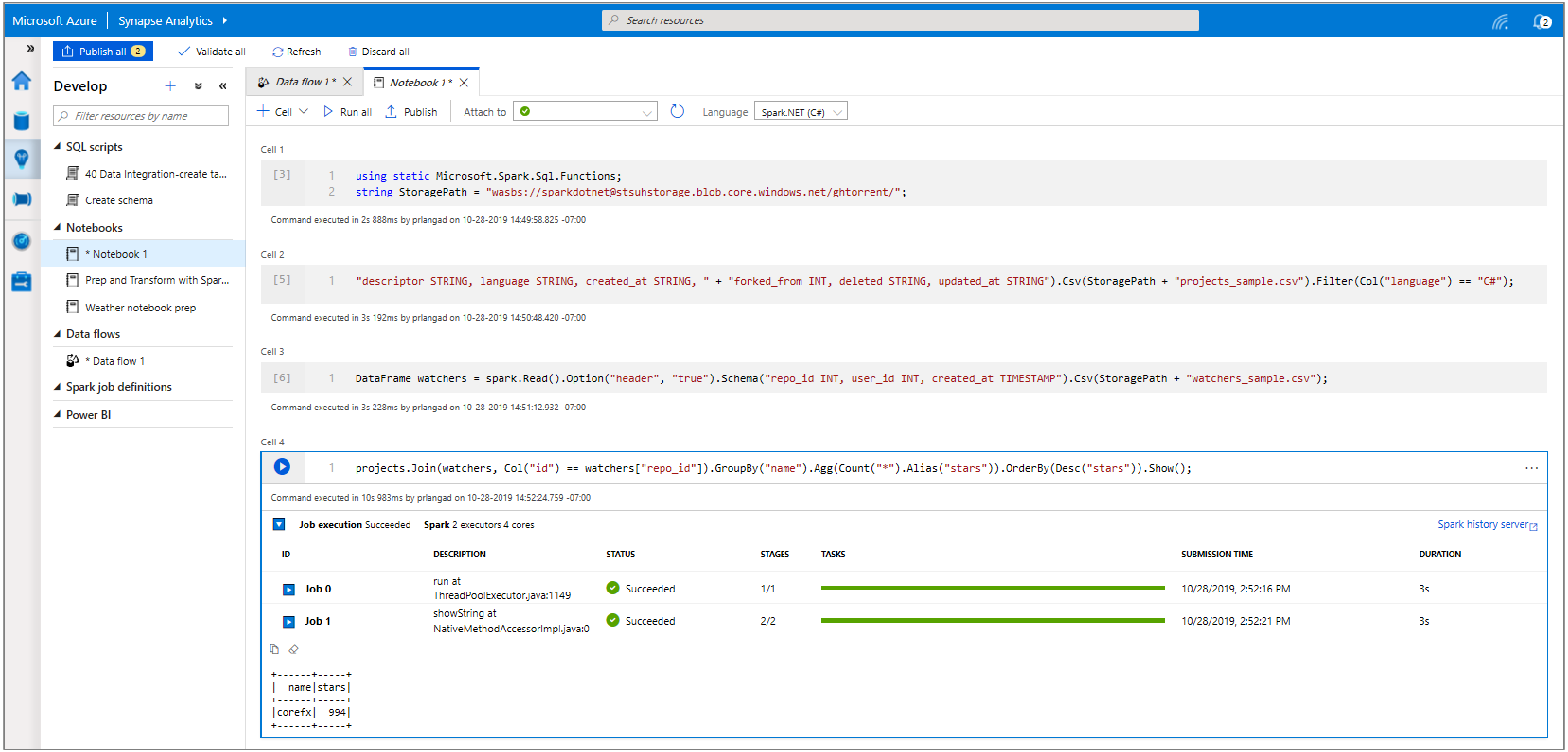
Command executed in 2s 91ms by prlangad on 11-04-2019 12:57:28.580 -08:00

Job execution Succeeded **Spark** 2 executors 4 cores Spark history server

ID	DESCRIPTION	STATUS	STAGES TASKS	SUBMISSION TIME	DURATION
Job 12	take at NativeMethodAccessorImpl.java:0	Succeeded	1/1	11/4/2019, 12:57:27 PM	0s

countryOrRegion	holidayName	normalizeHolidayName	isPaidTimeOff	countryRegionCode	date
United States	New Year's Day	New Year's Day		US	2019-01-01T00:00:00Z
United States	Martin Luther King, Jr. Day	Martin Luther King, Jr. Day		US	2019-01-21T00:00:00Z
United States	Washington's Birthday	Washington's Birthday		US	2019-02-18T00:00:00Z
United States	Memorial Day	Memorial Day		US	2019-05-27T00:00:00Z
United States	Independence Day	Independence Day		US	2019-07-04T00:00:00Z
United States	Labor Day	Labor Day		US	2019-09-02T00:00:00Z
United States	Columbus Day	Columbus Day		US	2019-10-14T00:00:00Z
United States	Veterans Day	Veterans Day		US	2019-11-11T00:00:00Z
United States	Thanksgiving	Thanksgiving		US	2019-11-28T00:00:00Z
United States	Christmas Day	Christmas Day		US	2019-12-25T00:00:00Z

Languages – Spark.NET (C#)



Cell 1

```
[3]  using static Microsoft.Spark.Sql.Functions;
     string StoragePath = "wasbs://sparkdotnet@stsuhstorage.blob.core.windows.net/ghtorrent/";
```

Command executed in 2s 888ms by prlangad on 10-28-2019 14:49:58.825 -07:00

Cell 2

```
[5]  "descriptor STRING, language STRING, created_at STRING, " + "forked_from INT, deleted STRING, updated_at STRING").Csv(StoragePath + "projects_sample.csv").Filter(Col("language") == "C#");
```

Command executed in 3s 192ms by prlangad on 10-28-2019 14:50:48.420 -07:00

Cell 3

```
[6]  DataFrame watchers = spark.Read().Option("header", "true").Schema("repo_id INT, user_id INT, created_at TIMESTAMP").Csv(StoragePath + "watchers_sample.csv");
```

Command executed in 3s 228ms by prlangad on 10-28-2019 14:51:12.932 -07:00

Cell 4

```
projects.Join(watchers, Col("id") == watchers["repo_id"]).GroupBy("name").Agg(Count("*").Alias("stars")).OrderBy(Desc("stars")).Show();
```

Command executed in 10s 983ms by prlangad on 10-28-2019 14:52:24.759 -07:00

Job execution Succeeded **Spark** 2 executors 4 cores

ID	DESCRIPTION	STATUS	STAGES	TASKS	SUBMISSION TIME	DURATION
Job 0	run at ThreadPoolExecutor.java:1149	✓ Succeeded	1/1	🟩	10/28/2019, 2:52:16 PM	3s
Job 1	showString at NativeMethodAccessorImpl.java:0	✓ Succeeded	2/2	🟩	10/28/2019, 2:52:21 PM	3s

```
+------+-----+
|  name|stars|
+------+-----+
|corefx|  994|
+------+-----+
```

Library Management - Python

Overview

Customers can add new python libraries at Spark pool level

Benefits

Input requirements.txt in simple pip freeze format

Add new libraries to your cluster

Update versions of existing libraries on your cluster

Libraries will get installed for your Spark pool during cluster creation

Ability to specify different requirements file for different pools within the same workspace

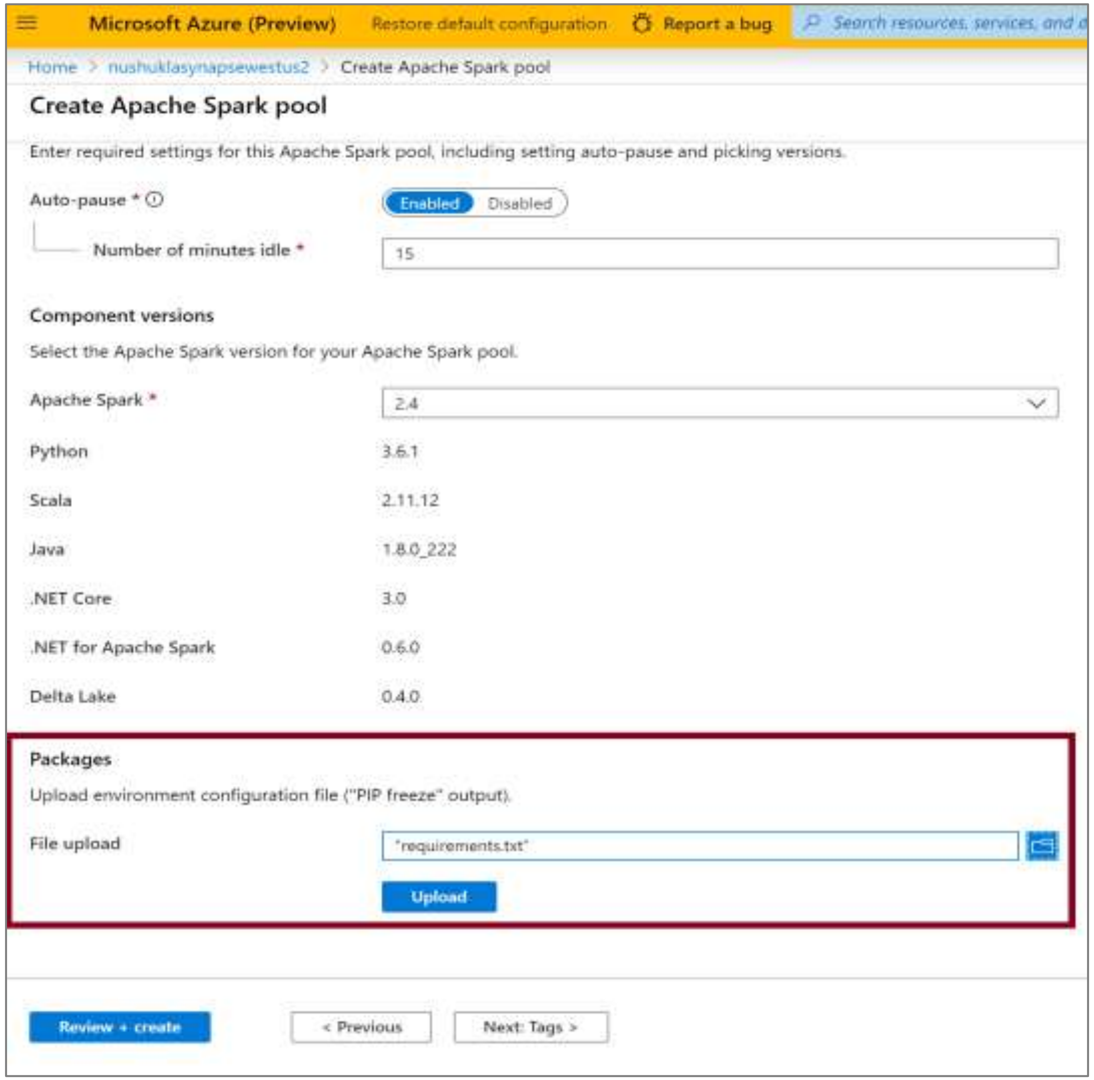
Constraints

The library version must exist on PyPI repository

Version downgrade of an existing library not allowed

In the Portal

Specify the new requirements while creating Spark Pool in Additional Settings blade


Microsoft Azure (Preview)
Restore default configuration
Report a bug
Search resources, services, and d
Home > nushuklasynapsewestus2 > Create Apache Spark pool
Upload
Review + create
< Previous
Next: Tags >

Create Apache Spark pool

Enter required settings for this Apache Spark pool, including setting auto-pause and picking versions.

Auto-pause * [✓ Enabled] Disabled

Number of minutes idle * 15

Component versions

Select the Apache Spark version for your Apache Spark pool.

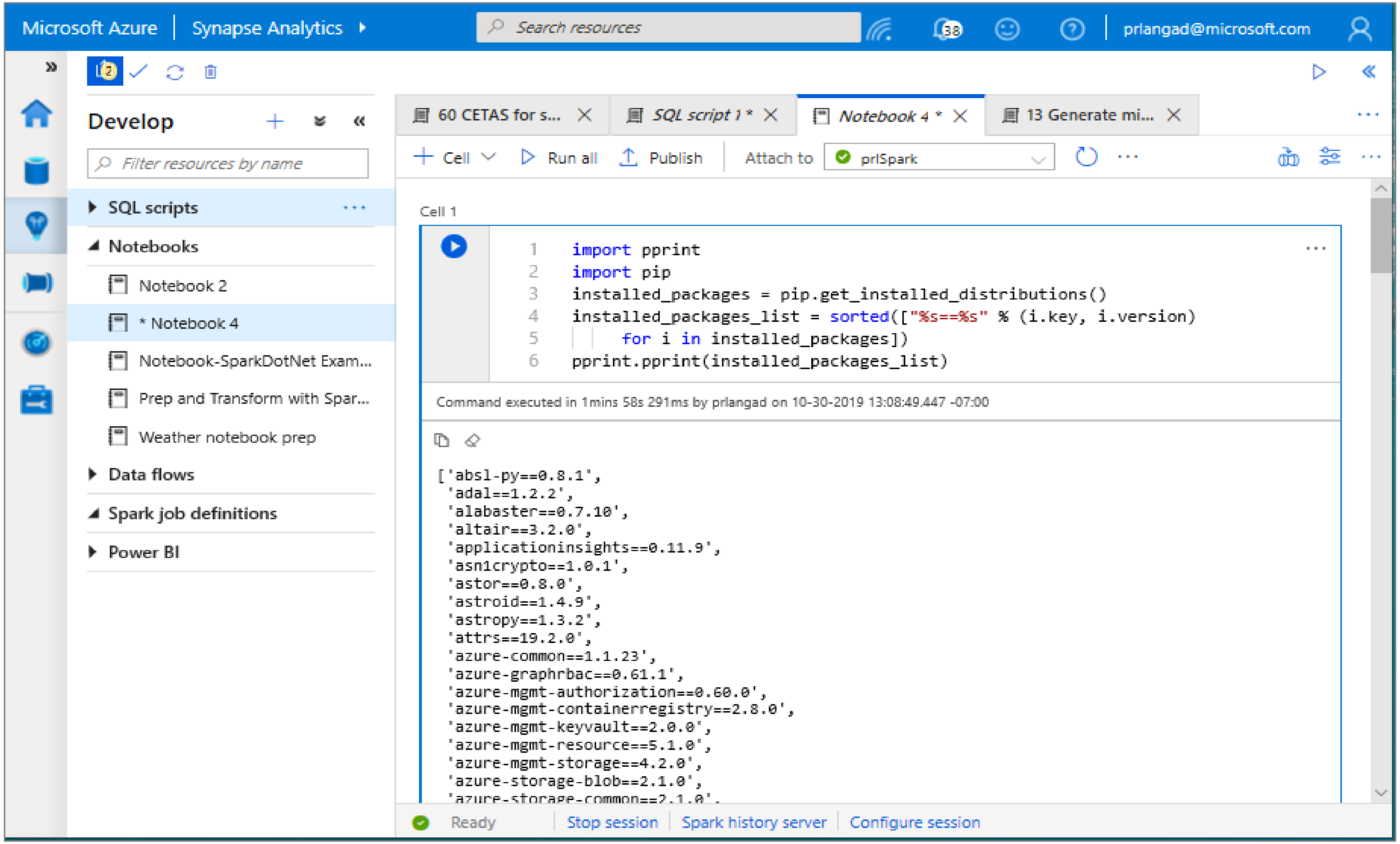
Apache Spark *	2.4
Python	3.6.1
Scala	2.11.12
Java	1.8.0_222
.NET Core	3.0
.NET for Apache Spark	0.6.0
Delta Lake	0.4.0

Packages

Upload environment configuration file ("PIP freeze" output).

File upload "requirements.txt"

Library Management - Python



```python
import pprint
import pip
installed_packages = pip.get_installed_distributions()
installed_packages_list = sorted(["%s==%s" % (i.key, i.version)
    for i in installed_packages])
pprint.pprint(installed_packages_list)
```

Command executed in 1mins 58s 291ms by prlangad on 10-30-2019 13:08:49.447 -07:00

```
['absl-py==0.8.1',
 'adal==1.2.2',
 'alabaster==0.7.10',
 'altair==3.2.0',
 'applicationinsights==0.11.9',
 'asn1crypto==1.0.1',
 'astor==0.8.0',
 'astroid==1.4.9',
 'astropy==1.3.2',
 'attrs==19.2.0',
 'azure-common==1.1.23',
 'azure-graphrbac==0.61.1',
 'azure-mgmt-authorization==0.60.0',
 'azure-mgmt-containerregistry==2.8.0',
 'azure-mgmt-keyvault==2.0.0',
 'azure-mgmt-resource==5.1.0',
 'azure-mgmt-storage==4.2.0',
 'azure-storage-blob==2.1.0',
 'azure-storage-common==2.1.0',
```

Microsoft

Azure Synapse Analytics

Foundation

Manage – Access Control

Overview

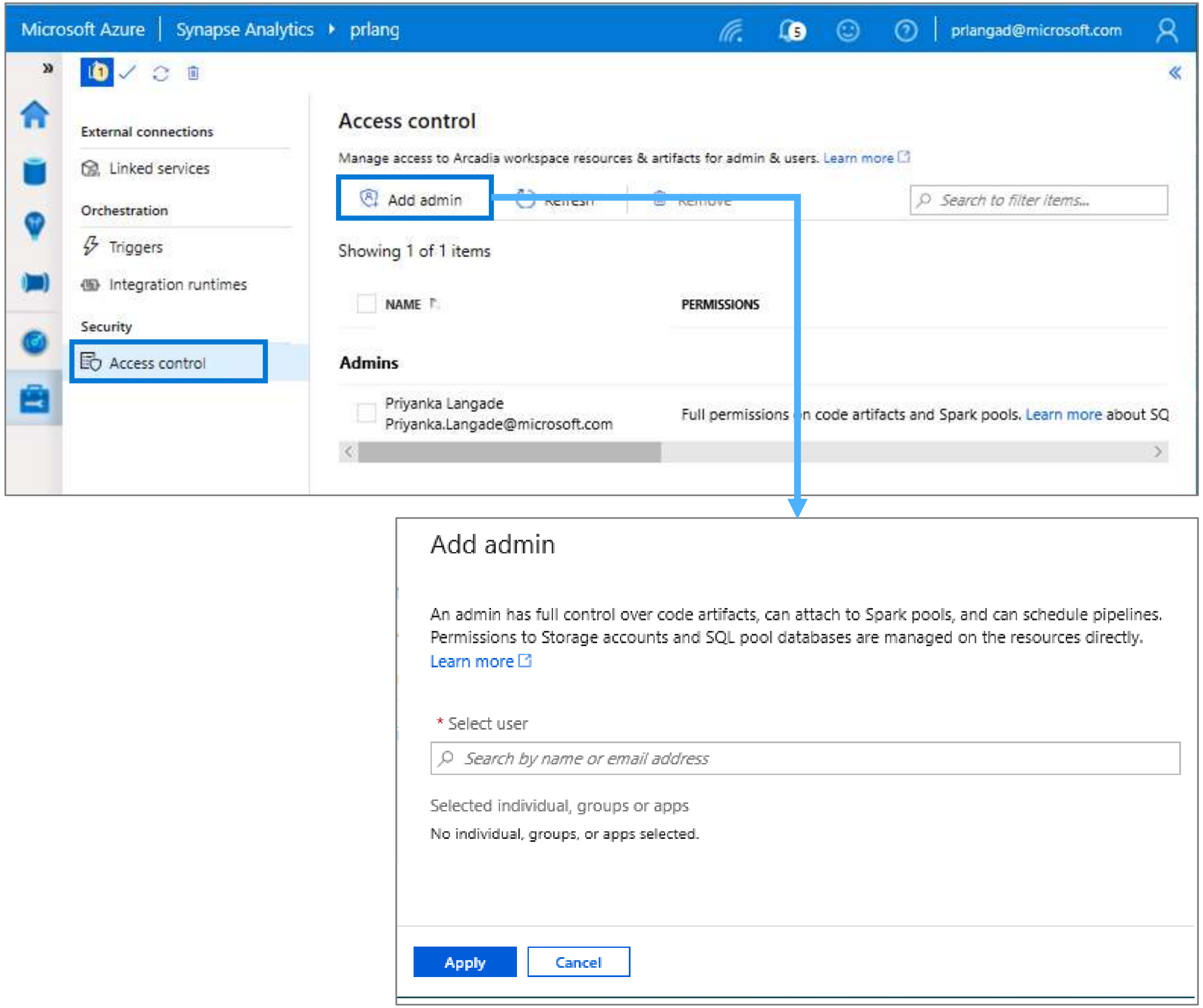
It provides access control management to workspace resources and artifacts for admin and users

Benefits

Share workspace with the team

Increases productivity

Manage permissions on code artifacts and Spark pools



Access control

Manage access to Arcadia workspace resources & artifacts for admin & users. Learn more

Showing 1 of 1 items

NAME	PERMISSIONS
Priyanka Langade Priyanka.Langade@microsoft.com	Full permissions on code artifacts and Spark pools. Learn more about SQ

An admin has full control over code artifacts, can attach to Spark pools, and can schedule pipelines. Permissions to Storage accounts and SQL pool databases are managed on the resources directly. Learn more

Selected individual, groups or apps

No individual, groups, or apps selected.

Spark Monitoring

Overview

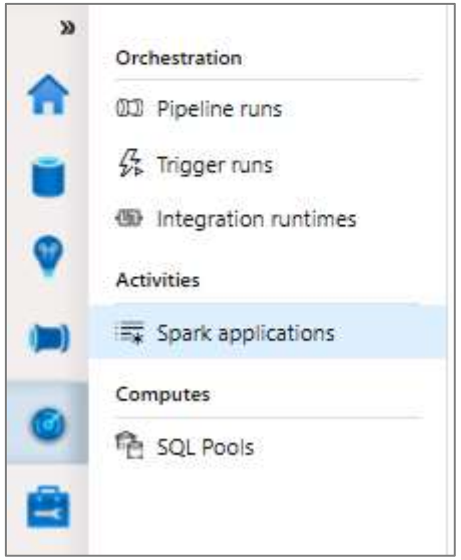
Monitor Spark pools, Spark applications for the progress and status of activities

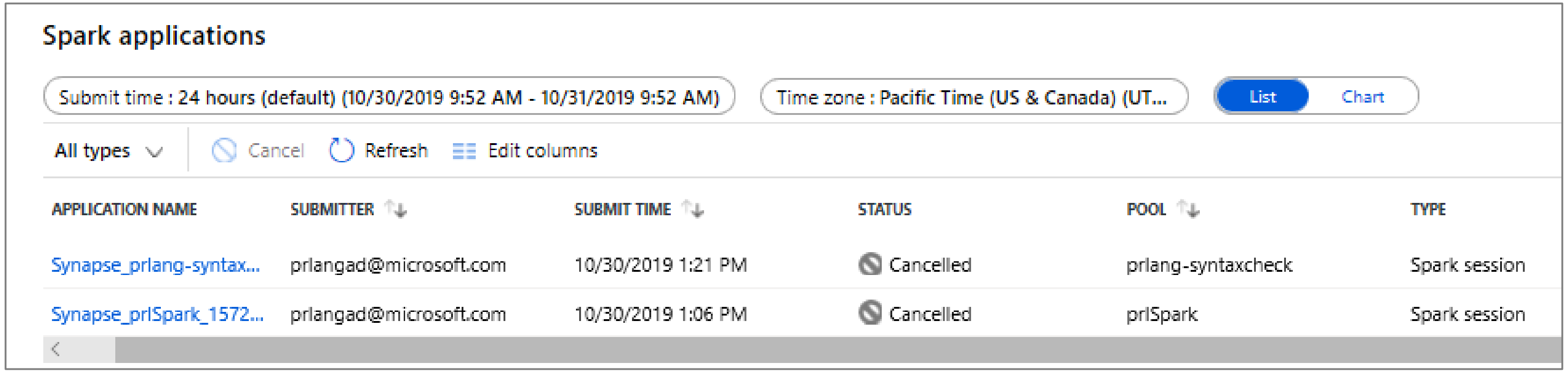
Benefits

Monitor Spark pools for the status as paused, active, resume, scaling and upgrading

Build a dashboard to monitor performance

Track the usage of resources





Spark applications

Submit time : 24 hours (default) (10/30/2019 9:52 AM - 10/31/2019 9:52 AM)

Time zone : Pacific Time (US & Canada) (UT...

APPLICATION NAME	SUBMITTER	SUBMIT TIME	STATUS	POOL	TYPE
Synapse_prlang-syntax...	prlangad@microsoft.com	10/30/2019 1:21 PM	Cancelled	prlang-syntaxcheck	Spark session
Synapse_prlSpark_1572...	prlangad@microsoft.com	10/30/2019 1:06 PM	Cancelled	prlSpark	Spark session

SQL Monitoring

Overview

Monitor SQL Pool in Azure Portal for overall usage and query activities.

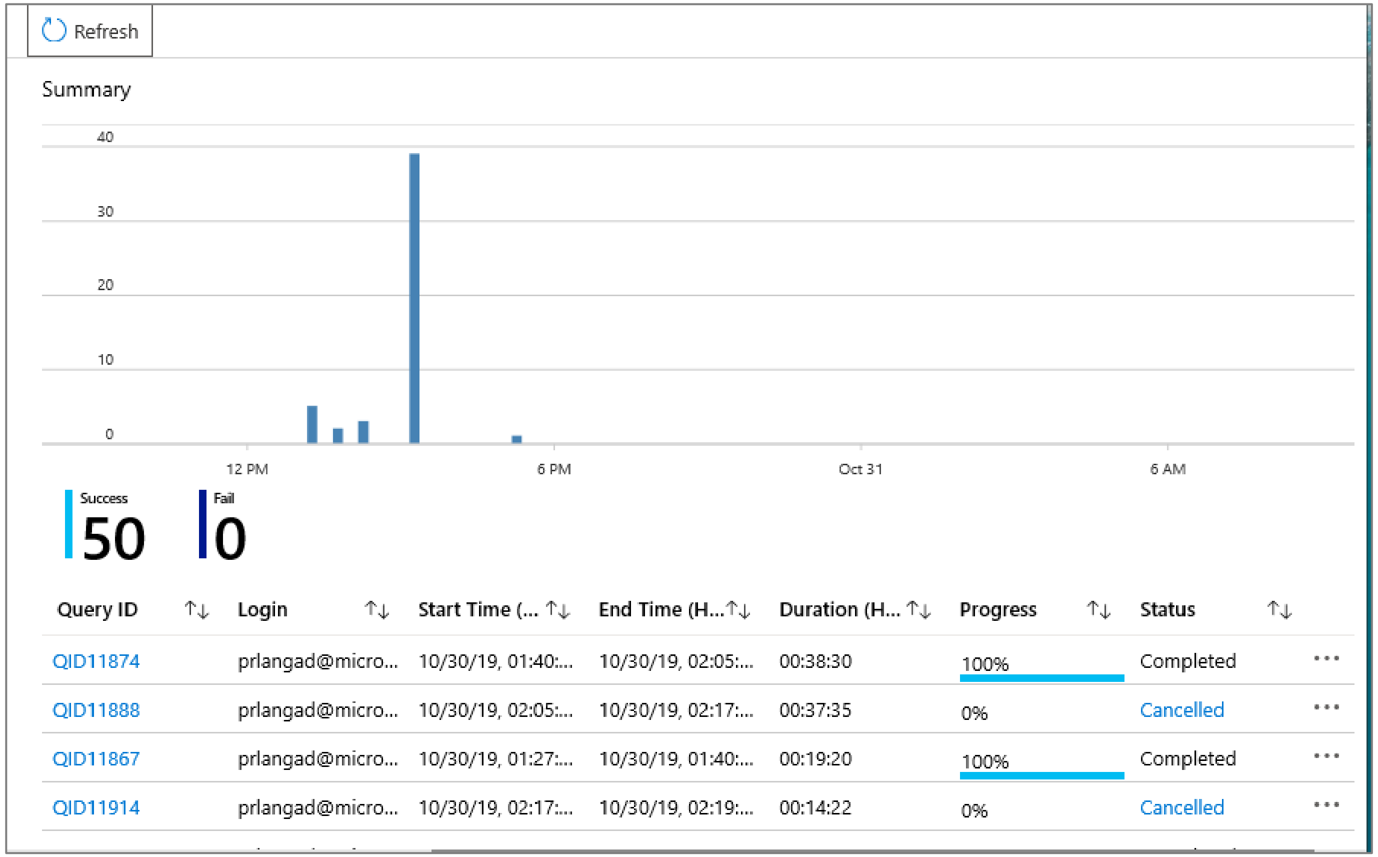
Benefits

Access SQL Audit Logs for my SQL computes

Monitor status and progress of all/specific activities

Dashboard view to monitor performance

Get to know scale of SQL compute resource



Query ID	Login	Start Time (...	End Time (H...	Duration (H...	Progress	Status	
QID11874	prlangad@micro...	10/30/19, 01:40:...	10/30/19, 02:05:...	00:38:30	100%	Completed	···
QID11888	prlangad@micro...	10/30/19, 02:05:...	10/30/19, 02:17:...	00:37:35	0%	Cancelled	···
QID11867	prlangad@micro...	10/30/19, 01:27:...	10/30/19, 01:40:...	00:19:20	100%	Completed	···
QID11914	prlangad@micro...	10/30/19, 02:17:...	10/30/19, 02:19:...	00:14:22	0%	Cancelled	···

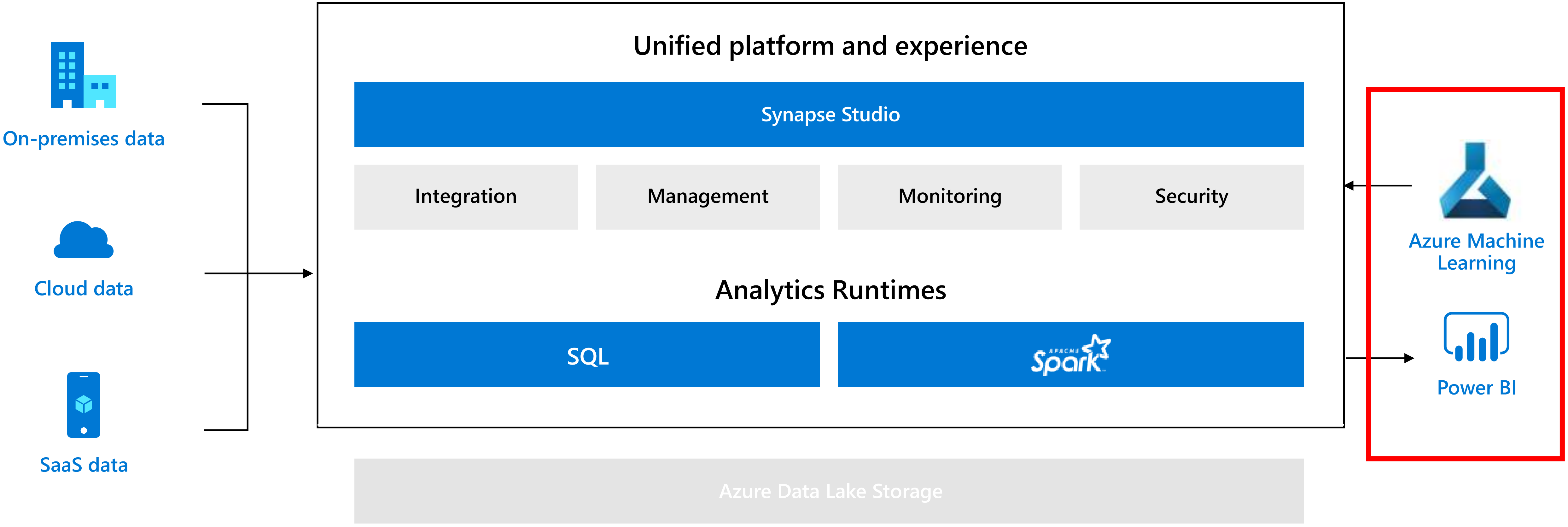
Microsoft

Azure Synapse Analytics

Connected Services

Azure Synapse Analytics

Limitless analytics service with unmatched time to insight



Azure Machine Learning Services

Overview

Data Scientists can use Azure ML notebooks to do (distributed) data preparation on Synapse Spark compute.

Benefits

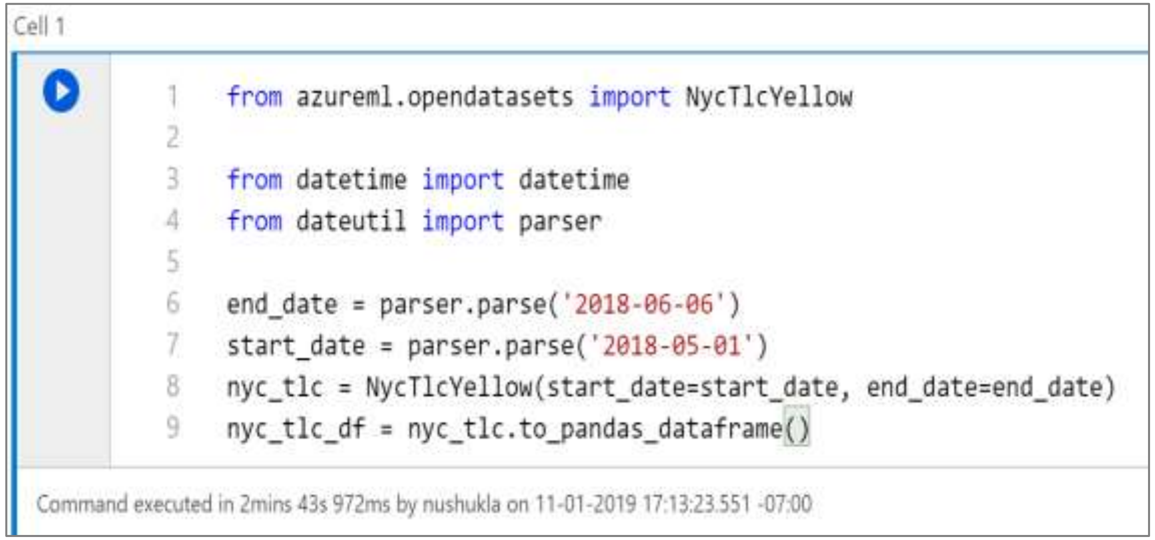
Connect to your existing Azure ML workspace and project

Use the AutoML Classifier for classification or regression problem

Train the model

Access open datasets

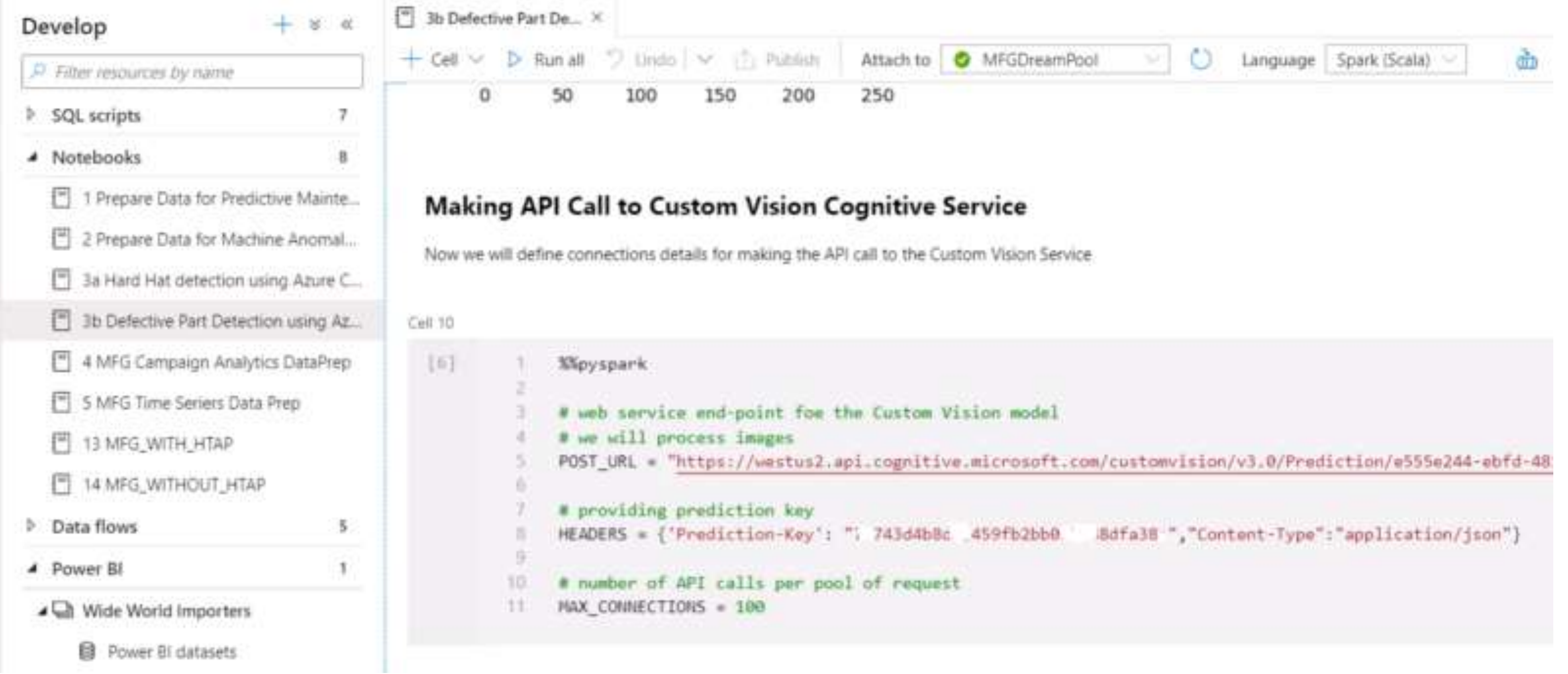
API Azure Cognitive Services



Cell 1

```
from azureml.opendatasets import NycTlcYellow

from datetime import datetime
from dateutil import parser

end_date = parser.parse('2018-06-06')
start_date = parser.parse('2018-05-01')
nyc_tlc = NycTlcYellow(start_date=start_date, end_date=end_date)
nyc_tlc_df = nyc_tlc.to_pandas_dataframe()
```

Command executed in 2mins 43s 972ms by nushukla on 11-01-2019 17:13:23.551 -07:00



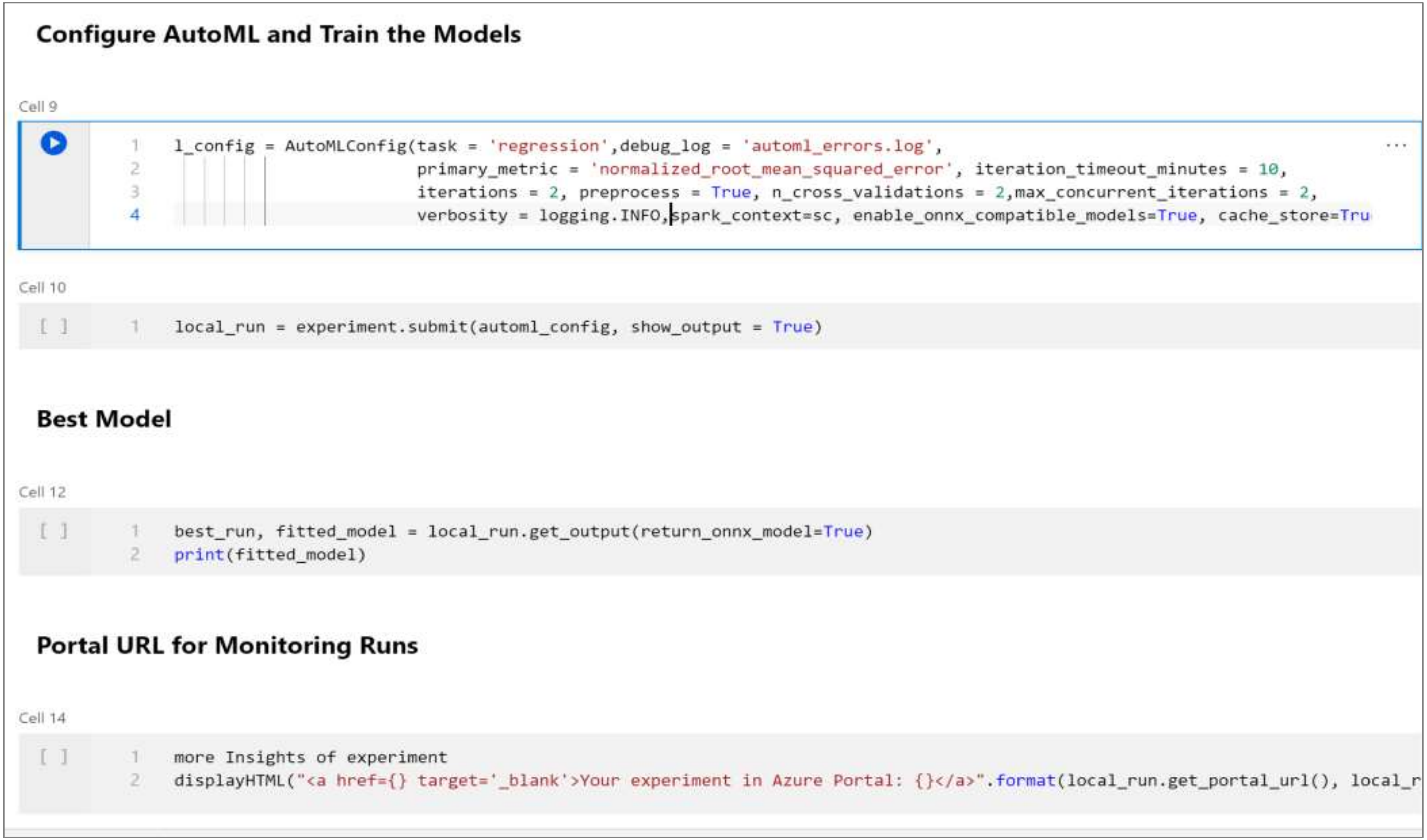
Making API Call to Custom Vision Cognitive Service

Now we will define connections details for making the API call to the Custom Vision Service

Cell 10

```
[6]  %%pyspark

     # web service end-point foe the Custom Vision model
     # we will process images
     POST_URL = "https://westus2.api.cognitive.microsoft.com/customvision/v3.0/Prediction/e555e244-ebfd-48

     # providing prediction key
     HEADERS = {'Prediction-Key': "[redacted] 743d4b8c [redacted] 459fb2bb0 [redacted] 8dfa38 ","Content-Type":"application/json"}

     # number of API calls per pool of request
     MAX_CONNECTIONS = 100
```

Azure Machine Learning Services (continued)



Configure AutoML and Train the Models

Cell 9

```
l_config = AutoMLConfig(task = 'regression',debug_log = 'automl_errors.log',
                        primary_metric = 'normalized_root_mean_squared_error', iteration_timeout_minutes = 10,
                        iterations = 2, preprocess = True, n_cross_validations = 2,max_concurrent_iterations = 2,
                        verbosity = logging.INFO,spark_context=sc, enable_onnx_compatible_models=True, cache_store=Tru
```

Cell 10

```
local_run = experiment.submit(automl_config, show_output = True)
```

Best Model

Cell 12

```
best_run, fitted_model = local_run.get_output(return_onnx_model=True)
print(fitted_model)
```

Portal URL for Monitoring Runs

Cell 14

```
more Insights of experiment
displayHTML("<a href={} target='_blank'>Your experiment in Azure Portal: {}</a>".format(local_run.get_portal_url(), local_r
```

Microsoft

Azure Synapse Analytics

Spark to Cosmos DB Connector

Spark to Cosmos DB Connector

Overview

Spark to Cosmos DB Connector

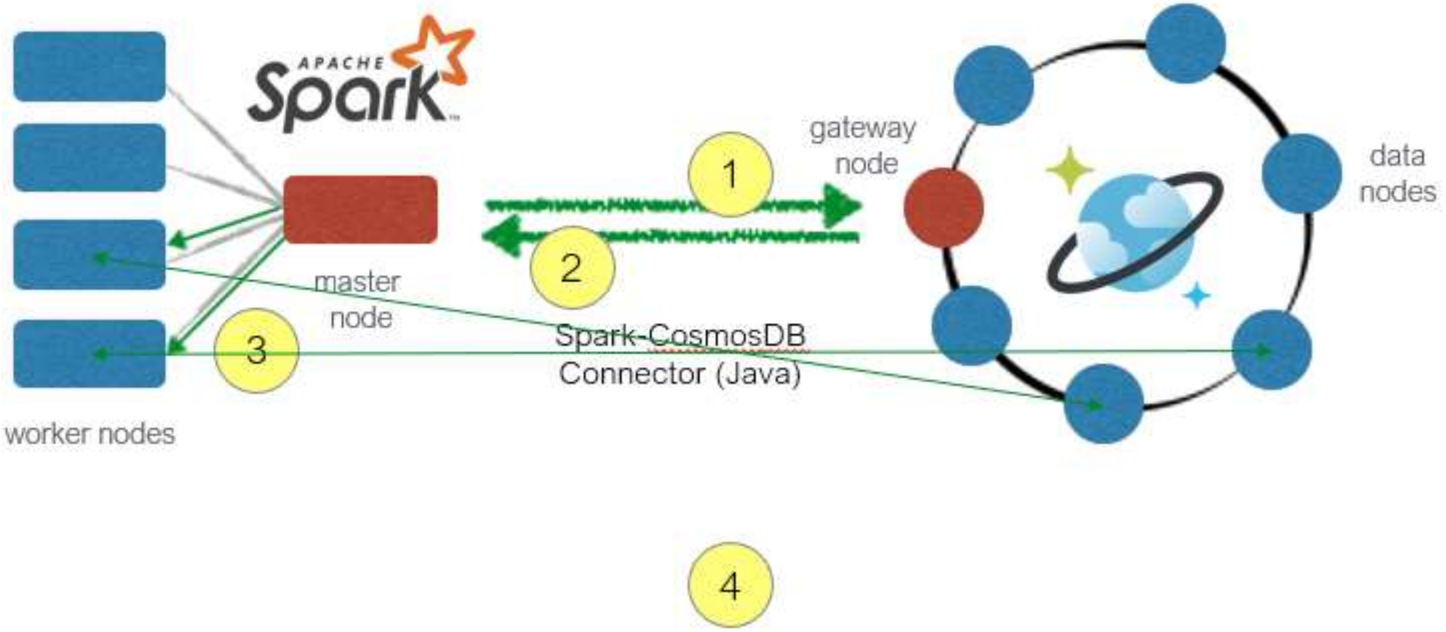
Benefits

1. Connection is made between Spark master node and Cosmos DB gateway node.

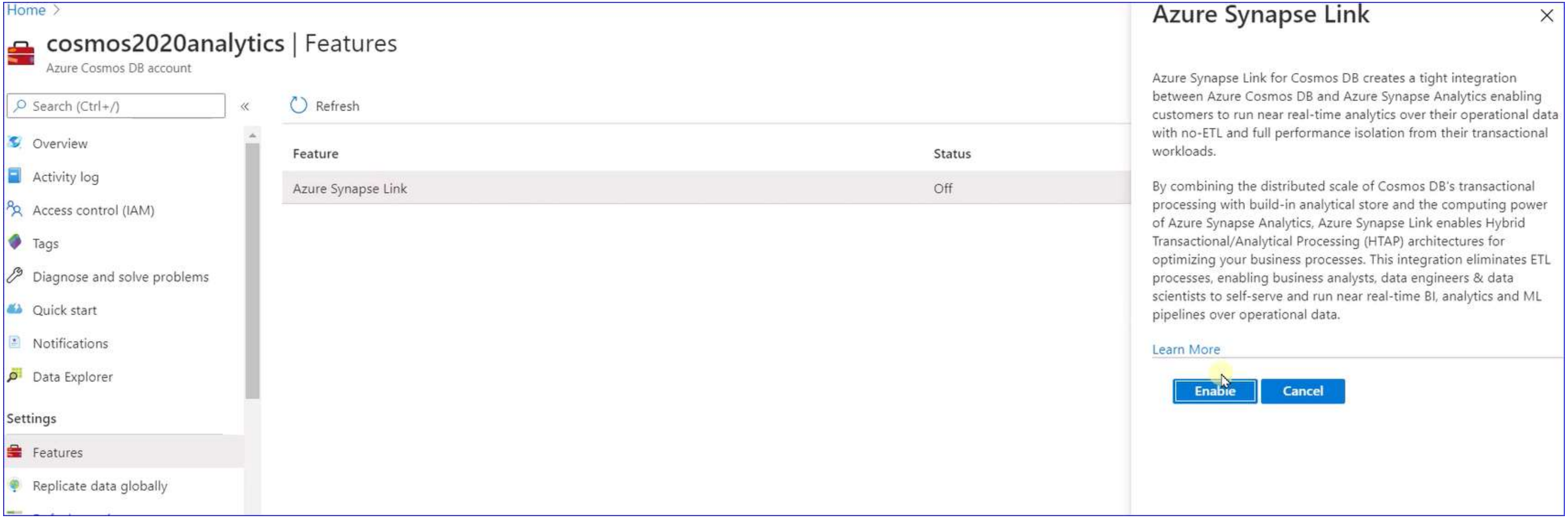
2. Partition map data is transmitted back to Spark master node.

3. Query is submitted from Spark worker nodes to

4. Cosmos DB data nodes and the data is transmitted back to Spark worker nodes for further processing



Azure Synapse Link for Cosmos DB



Feature	Status
Azure Synapse Link	Off

Azure Synapse Link

Azure Synapse Link for Cosmos DB creates a tight integration between Azure Cosmos DB and Azure Synapse Analytics enabling customers to run near real-time analytics over their operational data with no-ETL and full performance isolation from their transactional workloads.

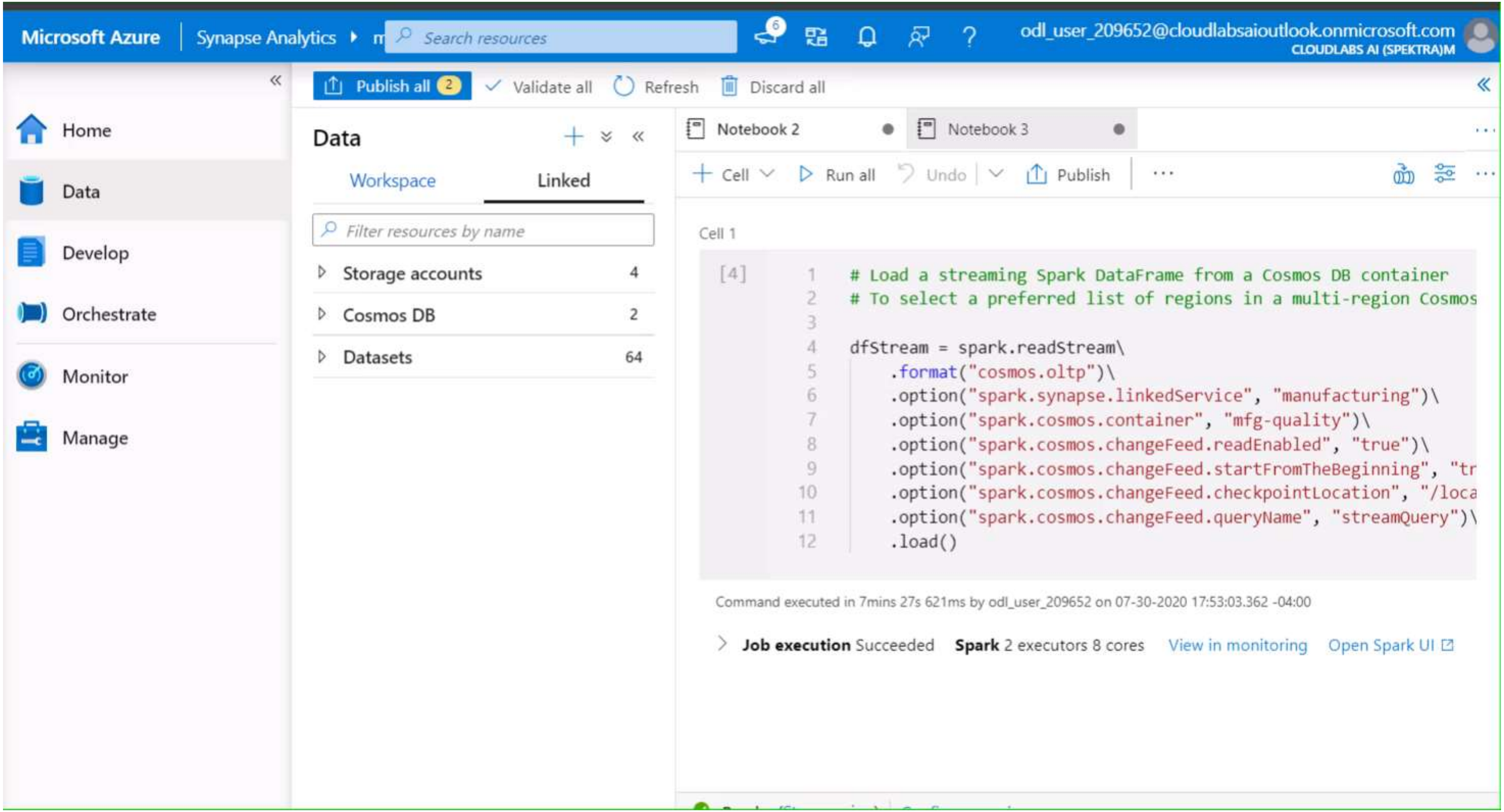
By combining the distributed scale of Cosmos DB's transactional processing with build-in analytical store and the computing power of Azure Synapse Analytics, Azure Synapse Link enables Hybrid Transactional/Analytical Processing (HTAP) architectures for optimizing your business processes. This integration eliminates ETL processes, enabling business analysts, data engineers & data scientists to self-serve and run near real-time BI, analytics and ML pipelines over operational data.

Microsoft

Azure Synapse Analytics

Spark to Cosmos DB Connector

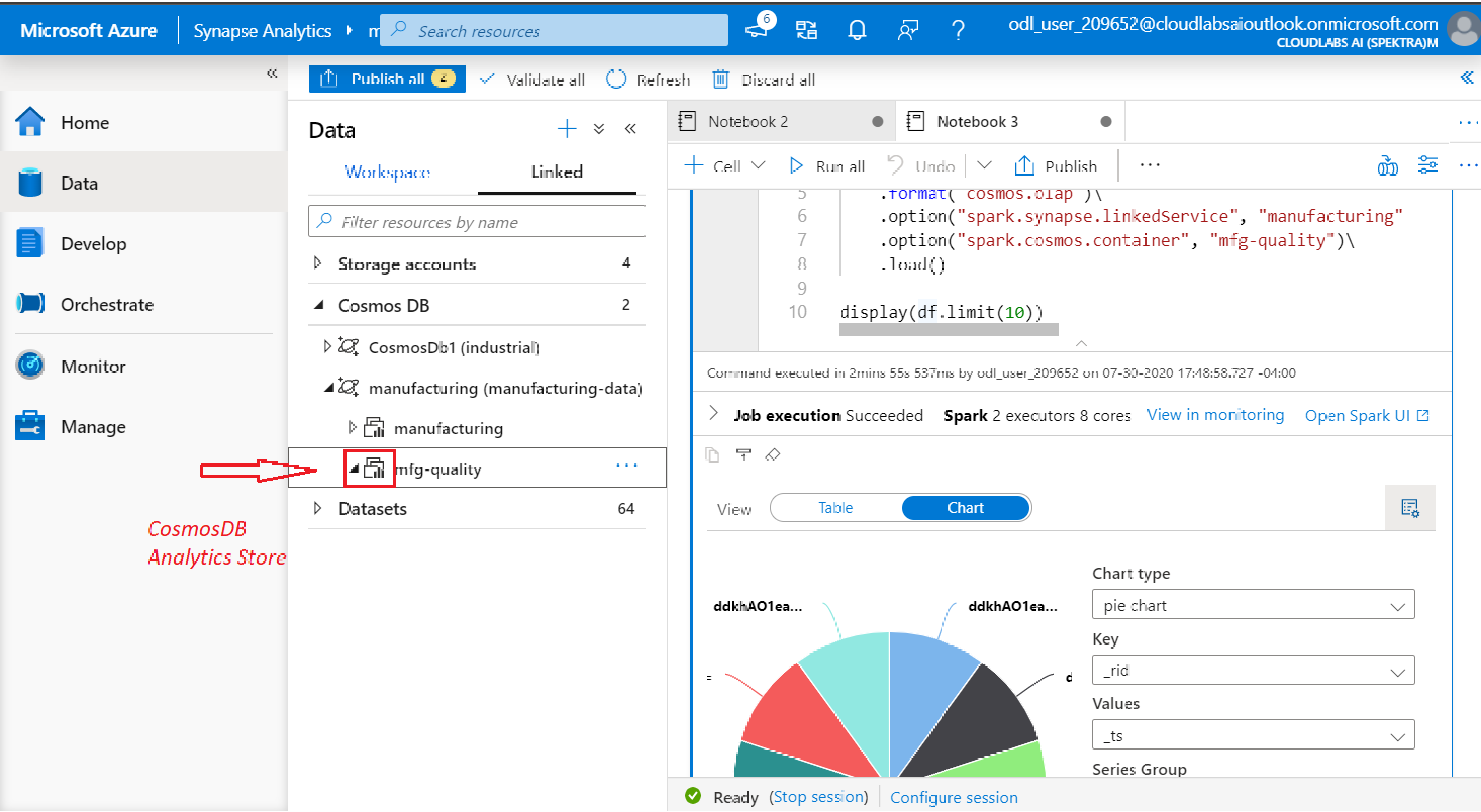
Demo



```
# Load a streaming Spark DataFrame from a Cosmos DB container
# To select a preferred list of regions in a multi-region Cosmos

dfStream = spark.readStream\
    .format("cosmos.oltp")\
    .option("spark.synapse.linkedService", "manufacturing")\
    .option("spark.cosmos.container", "mfg-quality")\
    .option("spark.cosmos.changeFeed.readEnabled", "true")\
    .option("spark.cosmos.changeFeed.startFromTheBeginning", "tr
    .option("spark.cosmos.changeFeed.checkpointLocation", "/loca
    .option("spark.cosmos.changeFeed.queryName", "streamQuery")\
    .load()
```

Command executed in 7mins 27s 621ms by odl_user_209652 on 07-30-2020 17:53:03.362 -04:00

Job execution Succeeded **Spark** 2 executors 8 cores View in monitoring Open Spark UI



CosmosDB Analytics Store

```
    .format("cosmos.olap")\
    .option("spark.synapse.linkedService", "manufacturing")\
    .option("spark.cosmos.container", "mfg-quality")\
    .load()

display(df.limit(10))
```

Command executed in 2mins 55s 537ms by odl_user_209652 on 07-30-2020 17:48:58.727 -04:00

Job execution Succeeded Spark 2 executors 8 cores View in monitoring Open Spark UI

View: Table | Chart

Chart type: pie chart

Key: _rid

Values: _ts

Series Group

Microsoft

Azure Synapse Analytics

Power BI

Power BI

Overview

Power BI is a business analytics service that delivers insights to enable fast, informed decisions

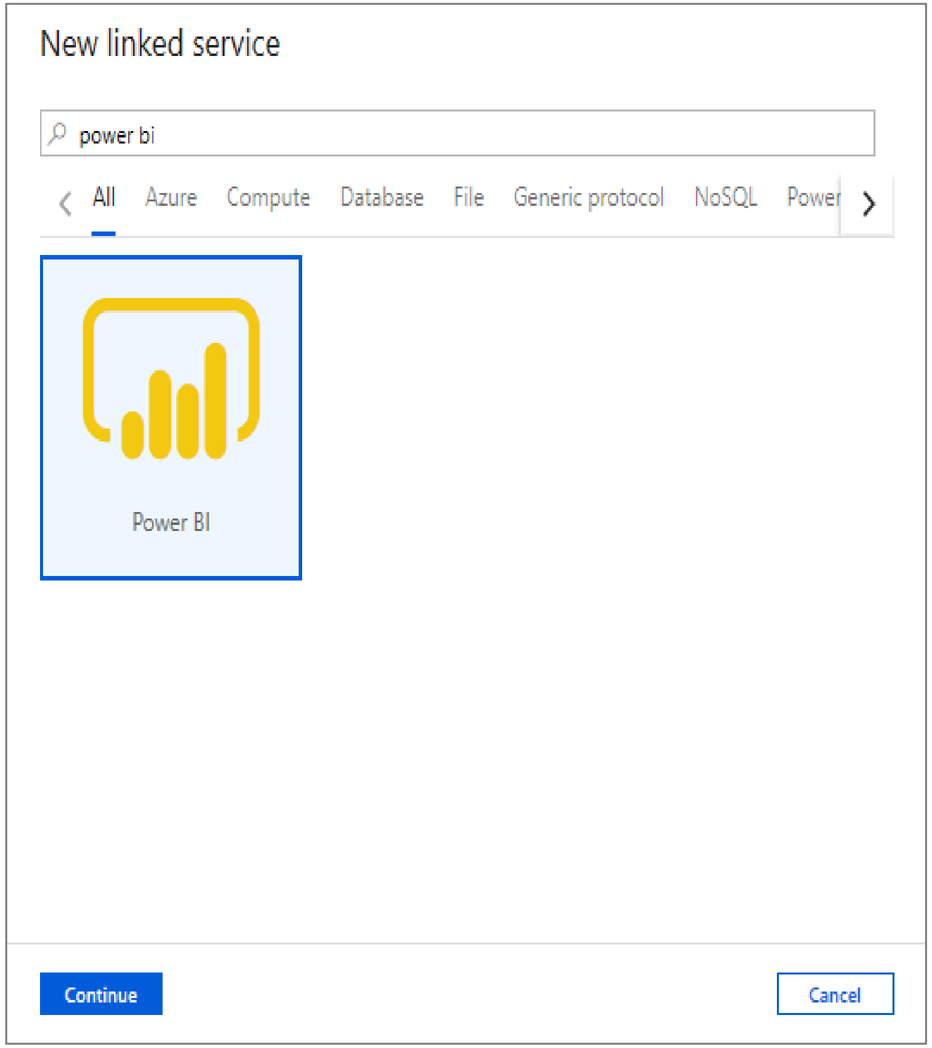
Benefits

Create Power BI reports in the workspace

Have access to published reports in workspace

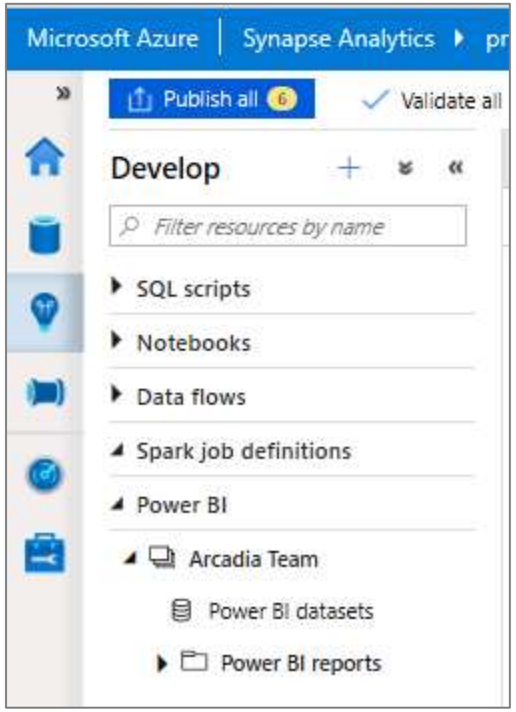
Update reports real time from Synapse workspace to get it reflected on Power BI service

Visually explore and analyze data

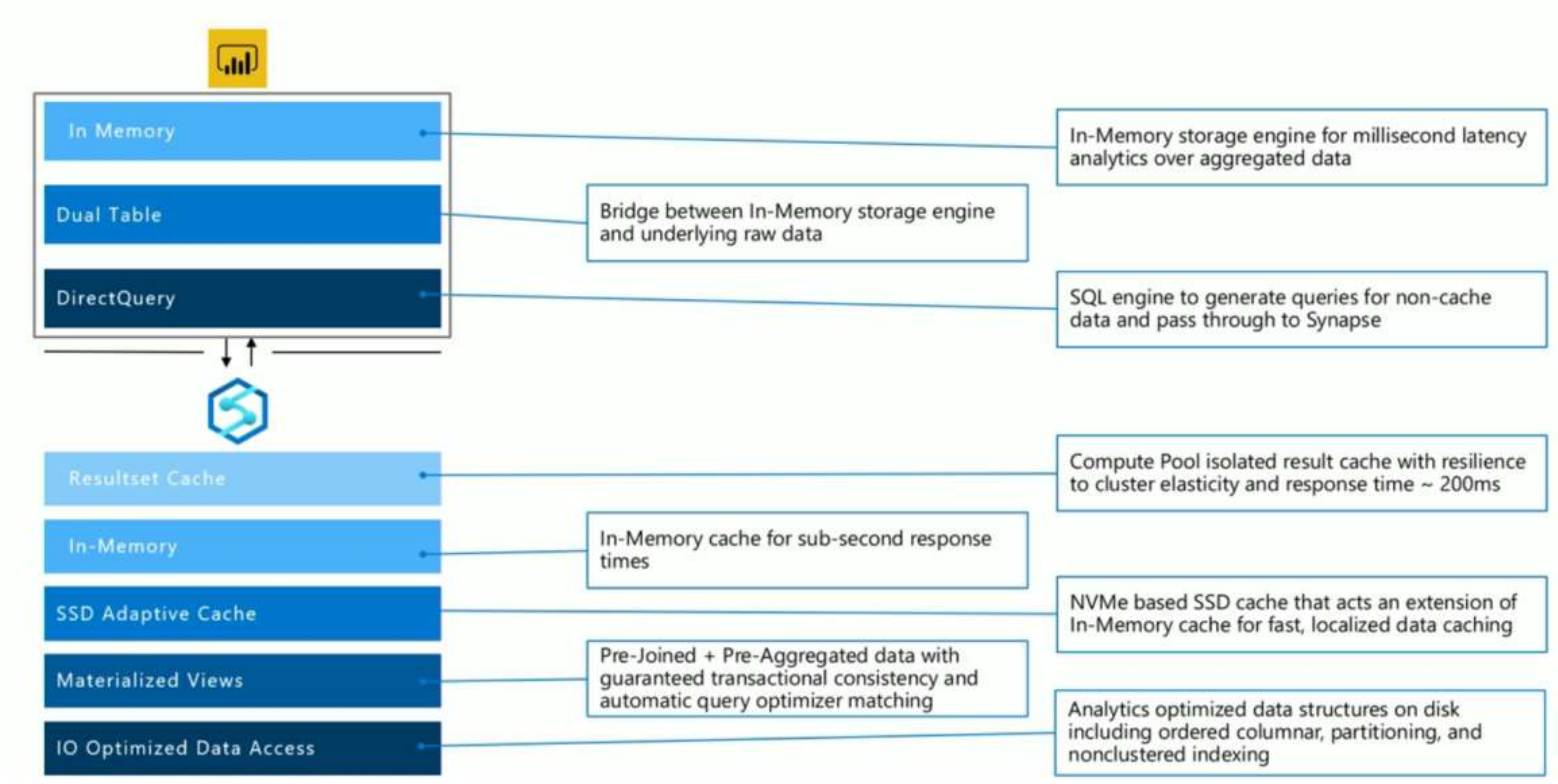


New linked service

Search: power bi



Power BI Aggregations and Synapse query performance



In Memory

In-Memory storage engine for millisecond latency analytics over aggregated data

Dual Table

Bridge between In-Memory storage engine and underlying raw data

DirectQuery

SQL engine to generate queries for non-cache data and pass through to Synapse

Resultset Cache

Compute Pool isolated result cache with resilience to cluster elasticity and response time ~ 200ms

In-Memory

In-Memory cache for sub-second response times

SSD Adaptive Cache

NVMe based SSD cache that acts an extension of In-Memory cache for fast, localized data caching

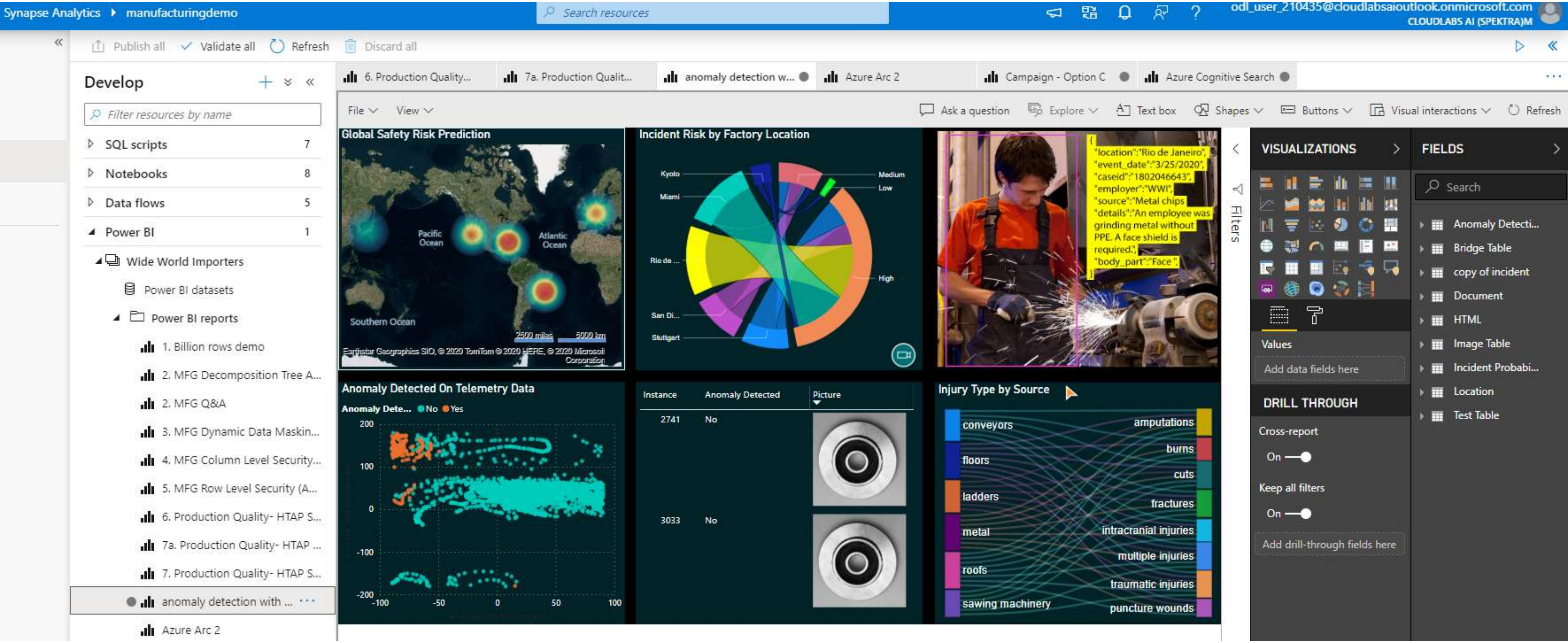
Materialized Views

Pre-Joined + Pre-Aggregated data with guaranteed transactional consistency and automatic query optimizer matching

IO Optimized Data Access

Analytics optimized data structures on disk including ordered columnar, partitioning, and nonclustered indexing

Power BI visualization end to end integration with Synapse Analytics



Develop

Filter resources by name

SQL scripts	7
Notebooks	8
Data flows	5
Power BI	1

- Wide World Importers
  - Power BI datasets
  - Power BI reports
    - 1. Billion rows demo
    - 2. MFG Decomposition Tree A...
    - 2. MFG Q&A
    - 3. MFG Dynamic Data Maskin...
    - 4. MFG Column Level Security...
    - 5. MFG Row Level Security (A...
    - 6. Production Quality- HTAP S...
    - 7a. Production Quality- HTAP ...
    - 7. Production Quality- HTAP S...
    - anomaly detection with ...
    - Azure Arc 2

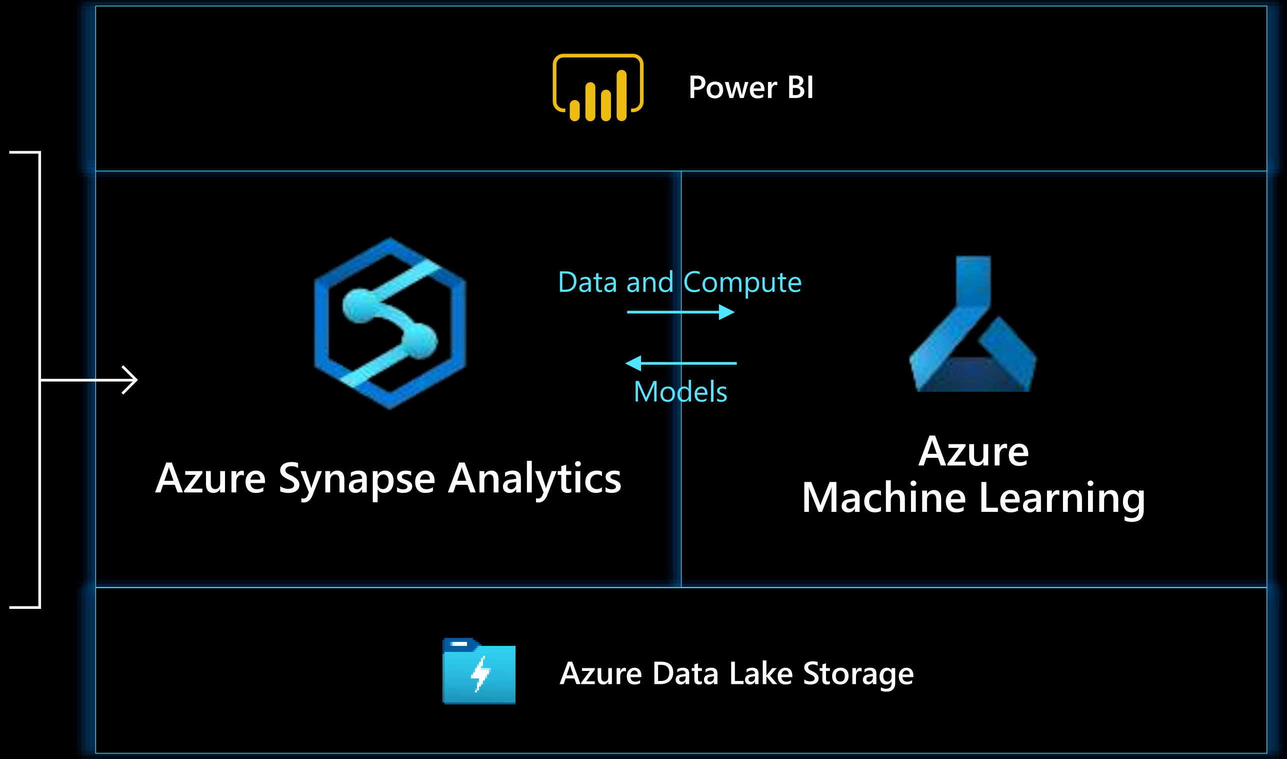
{
"location":"Rio de Janeiro",
"event_date":"3/25/2020",
"caseid":"1802046643",
"employer":"WWI",
"source":"Metal chips
"details":"An employee was grinding metal without PPE. A face shield is required.",
"body_part":"Face ",
}

Instance	Anomaly Detected	Picture
2741	No	
3033	No	

FIELDS:

- Anomaly Detecti...
- Bridge Table
- copy of incident
- Document
- HTML
- Image Table
- Incident Probabi...
- Location
- Test Table

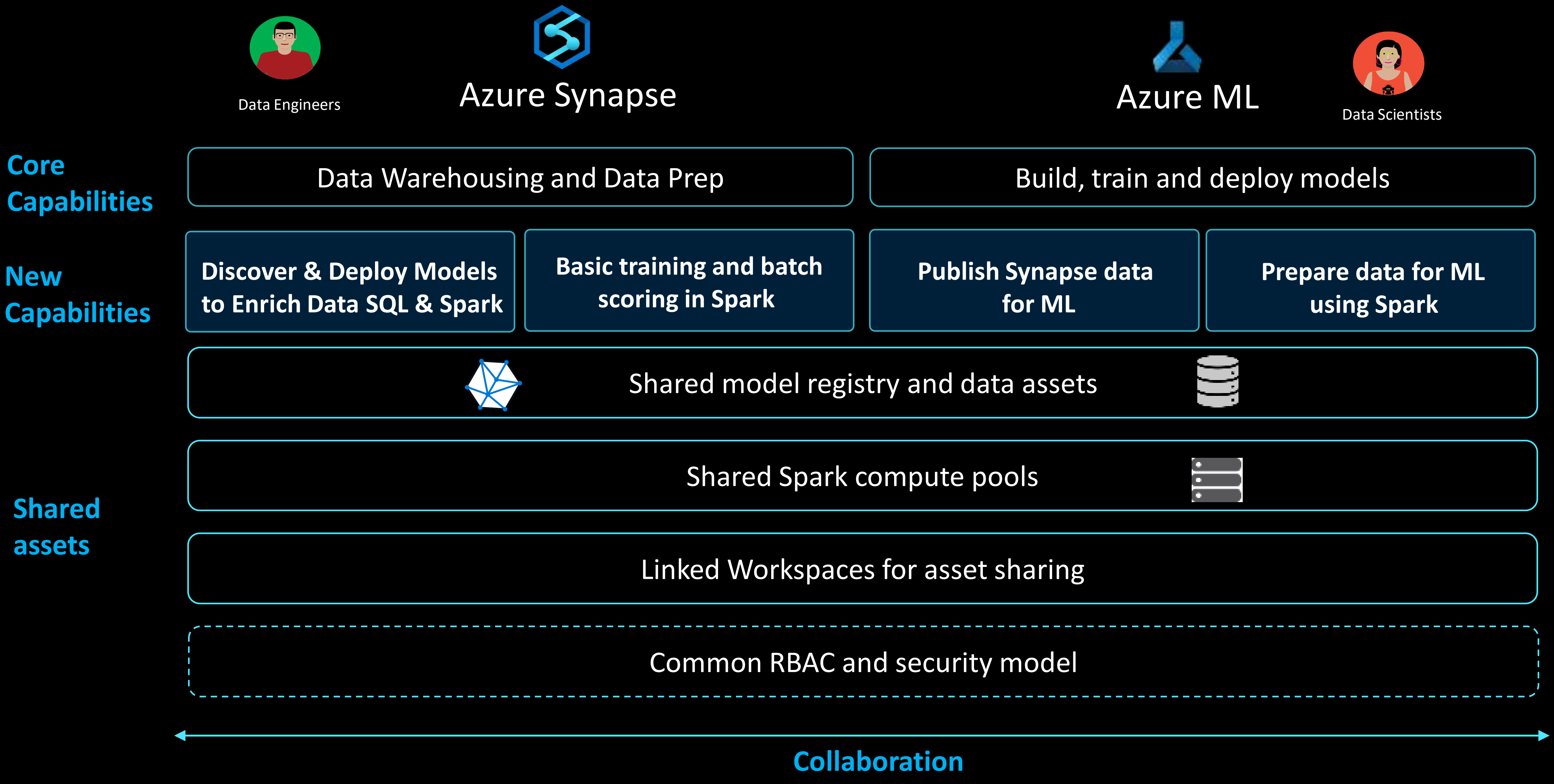
Analytics + Machine Learning



Batch Data Sources

- On-premises data
- Cloud data
- Devices data
- SaaS data

Augment analytics with Azure Machine Learning
(coming)



Data Engineers

Azure Synapse

Azure ML

Data Scientists

Core Capabilities

Data Warehousing and Data Prep

Build, train and deploy models

New Capabilities

Discover & Deploy Models to Enrich Data SQL & Spark

Basic training and batch scoring in Spark

Publish Synapse data for ML

Prepare data for ML using Spark

Shared assets

Shared model registry and data assets

Shared Spark compute pools

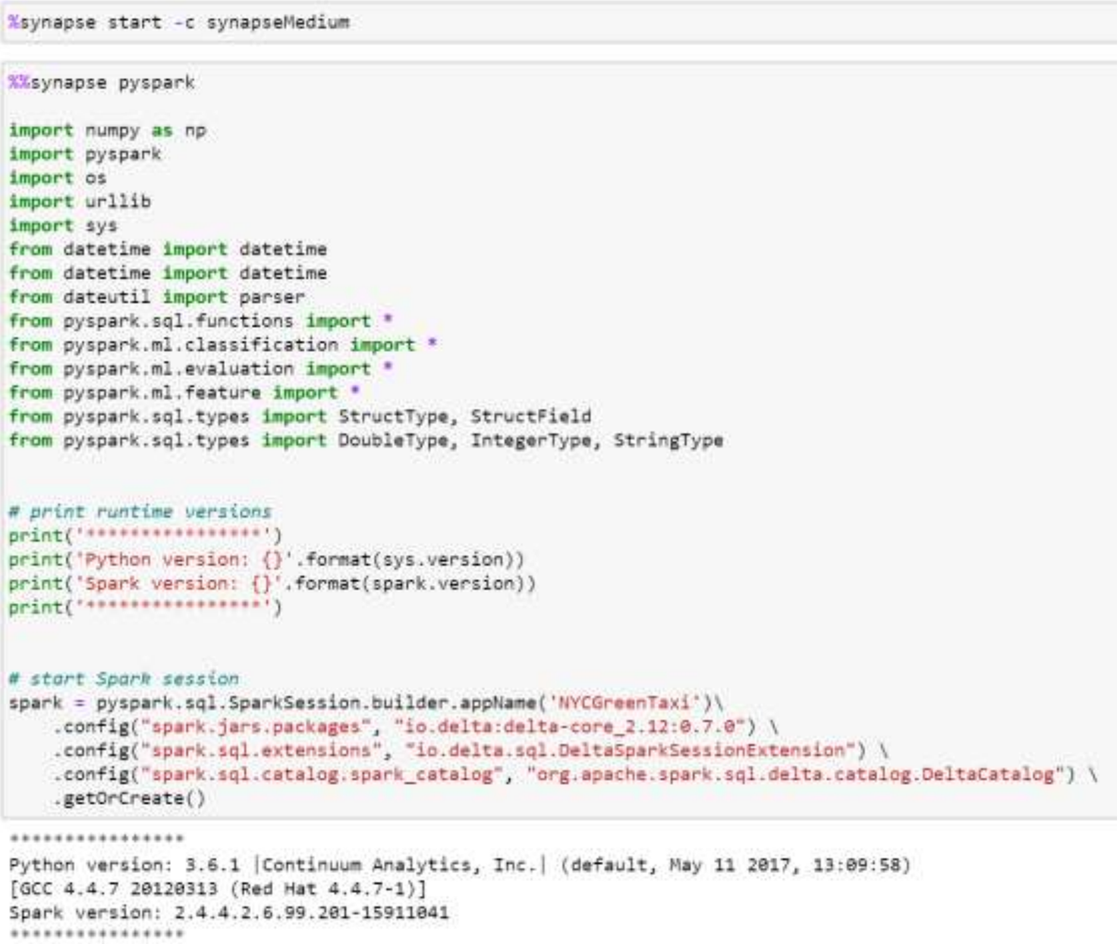
Linked Workspaces for asset sharing

Common RBAC and security model

Collaboration

Synapse Spark in Azure ML private preview experience – interactively run data processing on Spark in notebook

Step1: Start Spark session

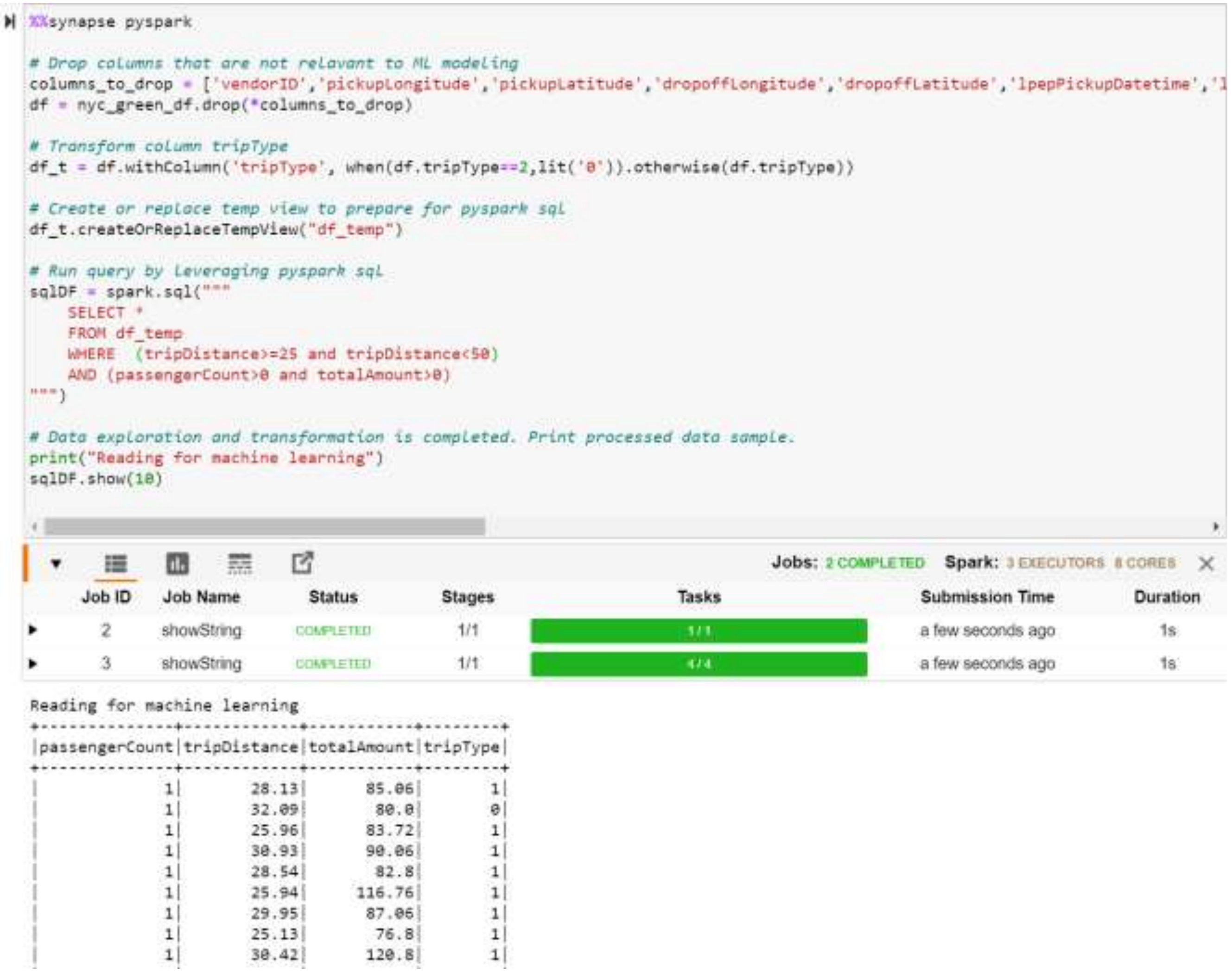


```
%synapse start -c synapseMedium
```

```
%%synapse pyspark

import numpy as np
import pyspark
import os
import urllib
import sys
from datetime import datetime
from datetime import datetime
from dateutil import parser
from pyspark.sql.functions import *
from pyspark.ml.classification import *
from pyspark.ml.evaluation import *
from pyspark.ml.feature import *
from pyspark.sql.types import StructType, StructField
from pyspark.sql.types import DoubleType, IntegerType, StringType


# print runtime versions
print('****************')
print('Python version: {}'.format(sys.version))
print('Spark version: {}'.format(spark.version))
print('****************')


# start Spark session
spark = pyspark.sql.SparkSession.builder.appName('NYCGreenTaxi')\
    .config("spark.jars.packages", "io.delta:delta-core_2.12:0.7.0") \
    .config("spark.sql.extensions", "io.delta.sql.DeltaSparkSessionExtension") \
    .config("spark.sql.catalog.spark_catalog", "org.apache.spark.sql.delta.catalog.DeltaCatalog") \
    .getOrCreate()
```

```
****************
Python version: 3.6.1 |Continuum Analytics, Inc.| (default, May 11 2017, 13:09:58)
[GCC 4.4.7 20120313 (Red Hat 4.4.7-1)]
Spark version: 2.4.4.2.6.99.201-15911041
****************
```

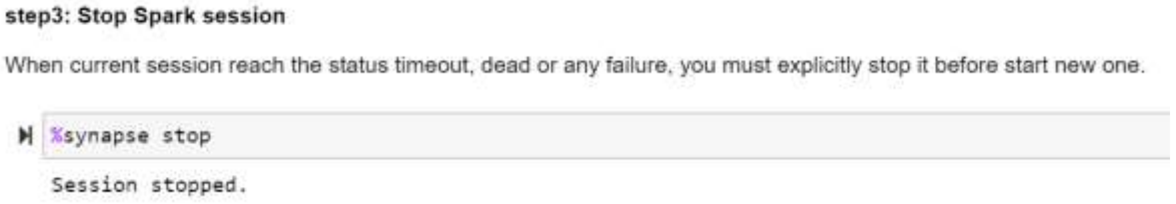
Step2: data exploration and transformation



```
%%synapse pyspark

# Drop columns that are not relavant to ML modeling
columns_to_drop = ['vendorID','pickupLongitude','pickupLatitude','dropoffLongitude','dropoffLatitude','lpepPickupDatetime','l
df = nyc_green_df.drop(*columns_to_drop)

# Transform column tripType
df_t = df.withColumn('tripType', when(df.tripType==2,lit('0')).otherwise(df.tripType))

# Create or replace temp view to prepare for pyspark sql
df_t.createOrReplaceTempView("df_temp")

# Run query by leveraging pyspark sql
sqlDF = spark.sql("""
    SELECT *
    FROM df_temp
    WHERE  (tripDistance>=25 and tripDistance<50)
    AND (passengerCount>0 and totalAmount>0)
""")

# Data exploration and transformation is completed. Print processed data sample.
print("Reading for machine learning")
sqlDF.show(10)
```

Job ID	Job Name	Status	Stages	Tasks	Submission Time	Duration
2	showString	COMPLETED	1/1	1/1	a few seconds ago	1s
3	showString	COMPLETED	1/1	4/4	a few seconds ago	1s

```
Reading for machine learning
+--------------+------------+-----------+--------+
|passengerCount|tripDistance|totalAmount|tripType|
+--------------+------------+-----------+--------+
|             1|       28.13|      85.06|       1|
|             1|       32.09|       80.0|       0|
|             1|       25.96|      83.72|       1|
|             1|       30.93|      90.06|       1|
|             1|       28.54|       82.8|       1|
|             1|       25.94|     116.76|       1|
|             1|       29.95|      87.06|       1|
|             1|       25.13|       76.8|       1|
|             1|       30.42|      120.8|       1|
```



step3: Stop Spark session

When current session reach the status timeout, dead or any failure, you must explicitly stop it before start new one.

```
%synapse stop
```

```
Session stopped.
```

Learn More

- What is SQL on-demand?: link
- What is Apache Spark in Azure Synapse Analytics?: link
- Best practices for SQL pool in Azure Synapse Analytics: link
- Best practices for SQL on-demand in Azure Synapse Analytics: link
- Azure Synapse Analytics shared metadata: link
- Use maintenance schedules to manage service updates and maintenance: link
- Cheat sheet for Azure Synapse Analytics (formerly SQL DW): link
- Best practices for SQL Analytics in Azure Synapse Analytics (formerly SQL DW): link
- Synapse Analytics documentation is here: aka.ms/SynapseDocs

Microsoft

Q&A

Adrián J. Fernández Zenteno

Sr. Cloud Solution Architect - Azure Data & AI

Adrian.Fernandez@microsoft.com

Microsoft

Muchas gracias