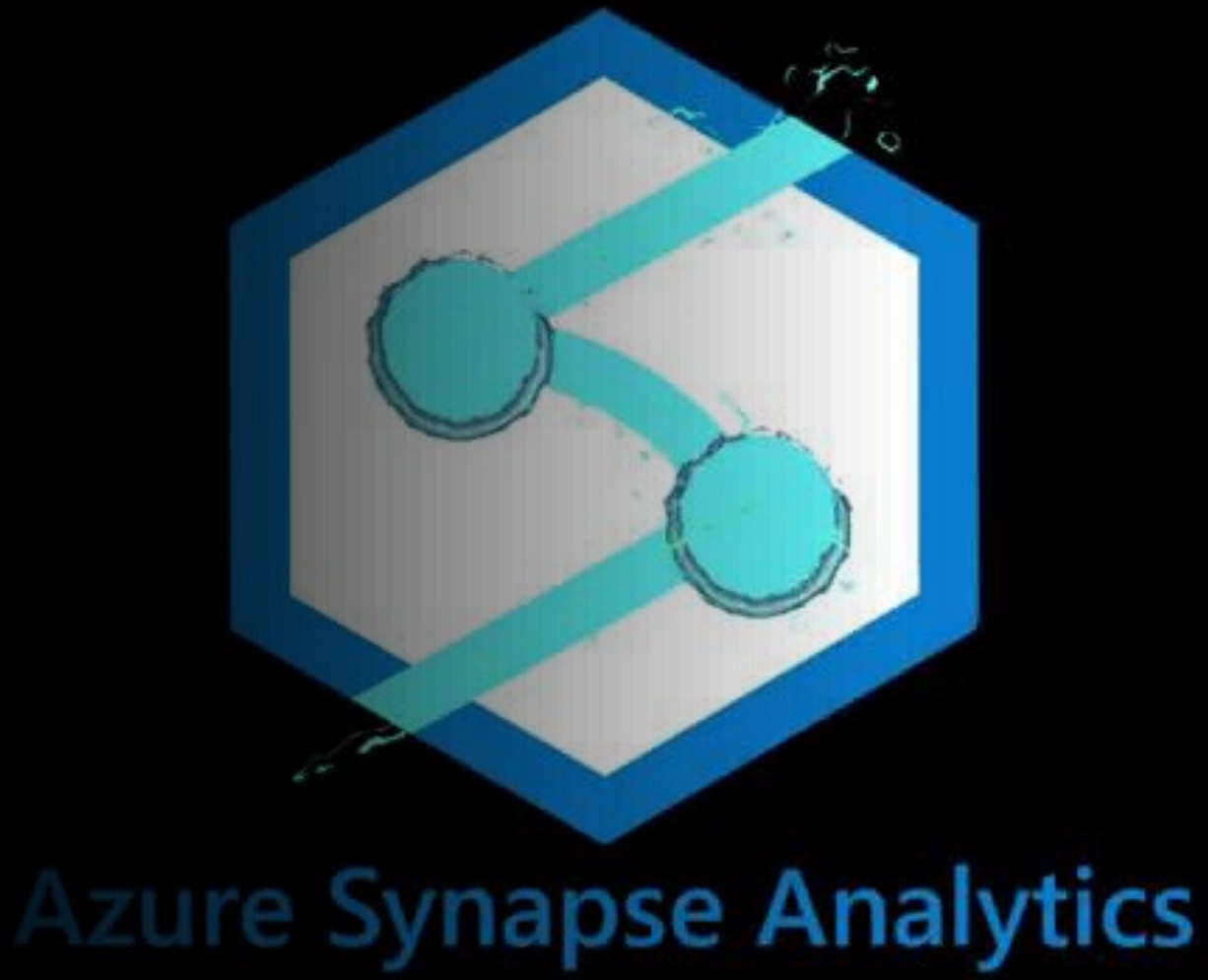
Modern Data Warehousing and Advanced Analysis with Azure Synapse

Poll Question

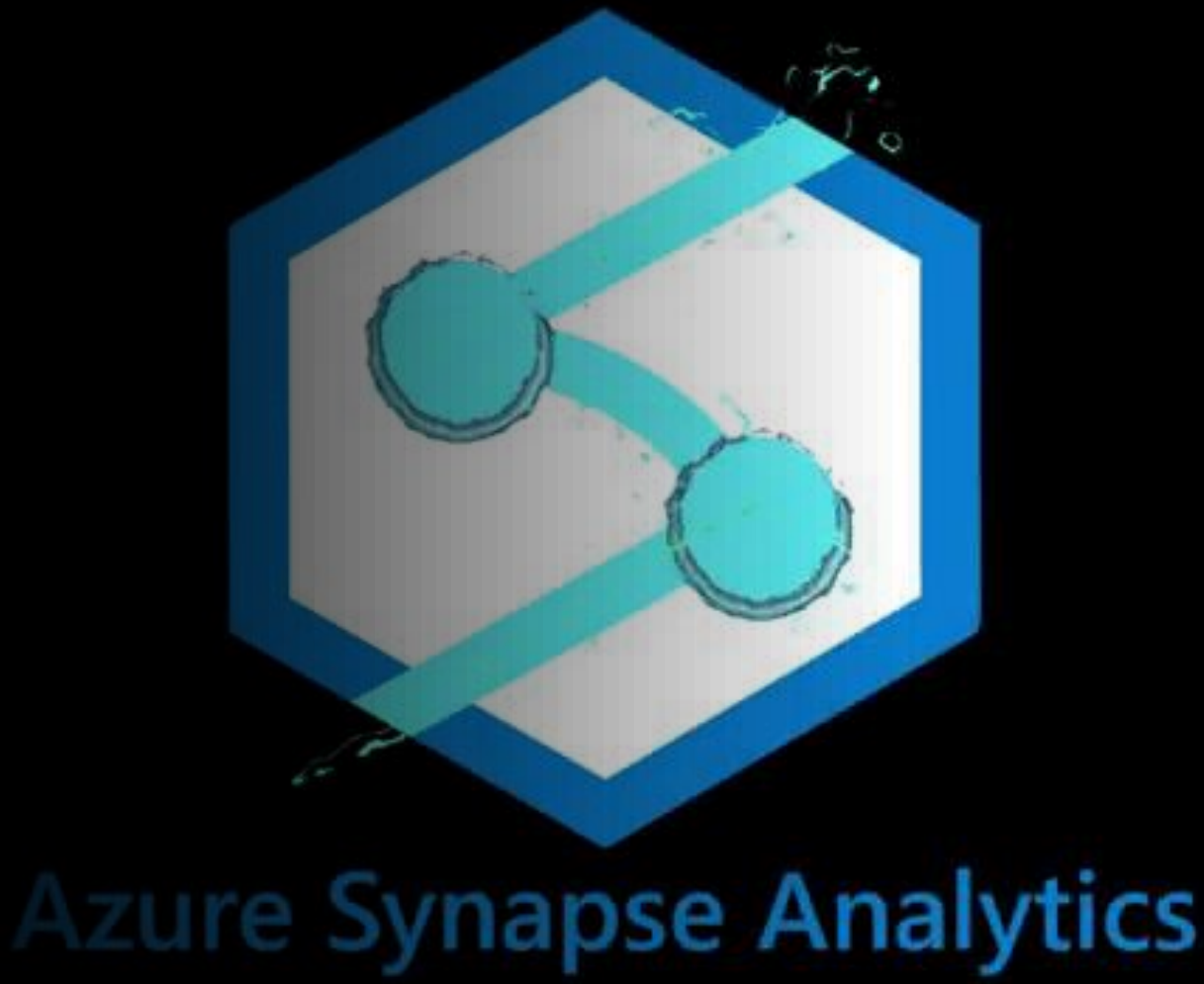
https://geni.us/SynapsePrecon1



Modern Data Warehousing and Advanced Analysis with Azure Synapse

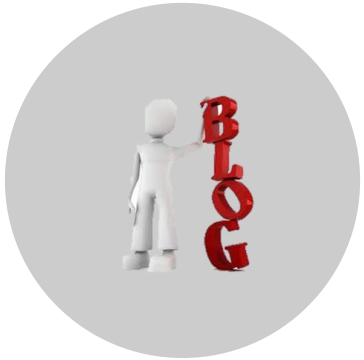
Ginger Grant

November 10, 2020



About Me – Ginger Grant

www.desertislesql.com

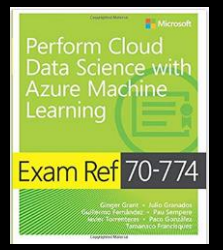


MVP Microsoft Most Valuable Professional

Microsoft CERTIFIED Trainer

ginger.grant@DesertIsleSQL.com

@DesertIsleSql



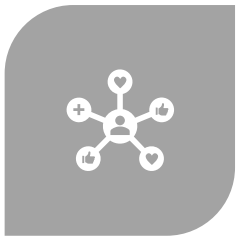
PASS LOCAL GROUP LEADER

Agenda

- Introduction to Azure Synapse
- SQL Serverless
- Data Lakehouse
- Power BI Integration
- Orchestration
- Data Warehousing
- Spark and Machine Learning
- Security

What is Synapse?

NEW VERSION OF SQL DW



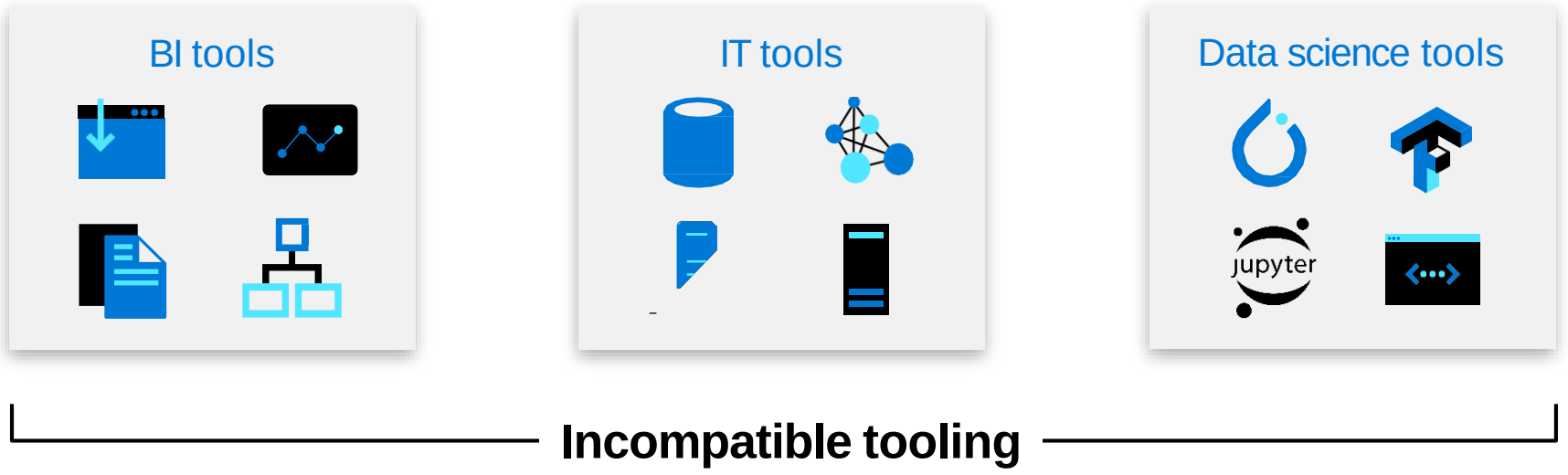
ANOTHER PLACE TO DO ADF

SPARK IMPLEMENTATION

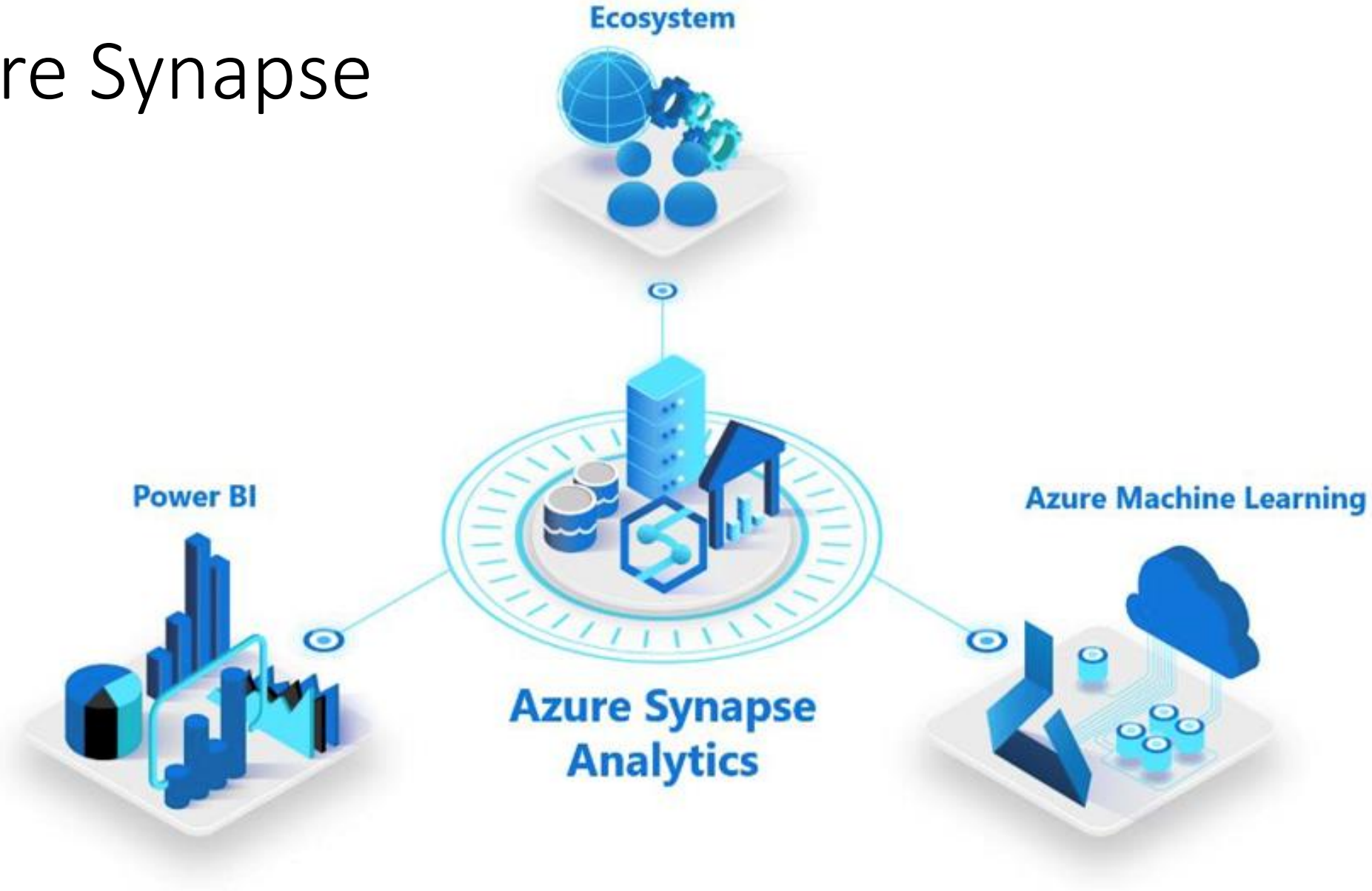
PLACE TO WRITE SQL AGAINST BLOB STORAGE

ANOTHER WORD PEOPLE SAY DIFFERENTLY IN DIFFERENT COUNTRIES

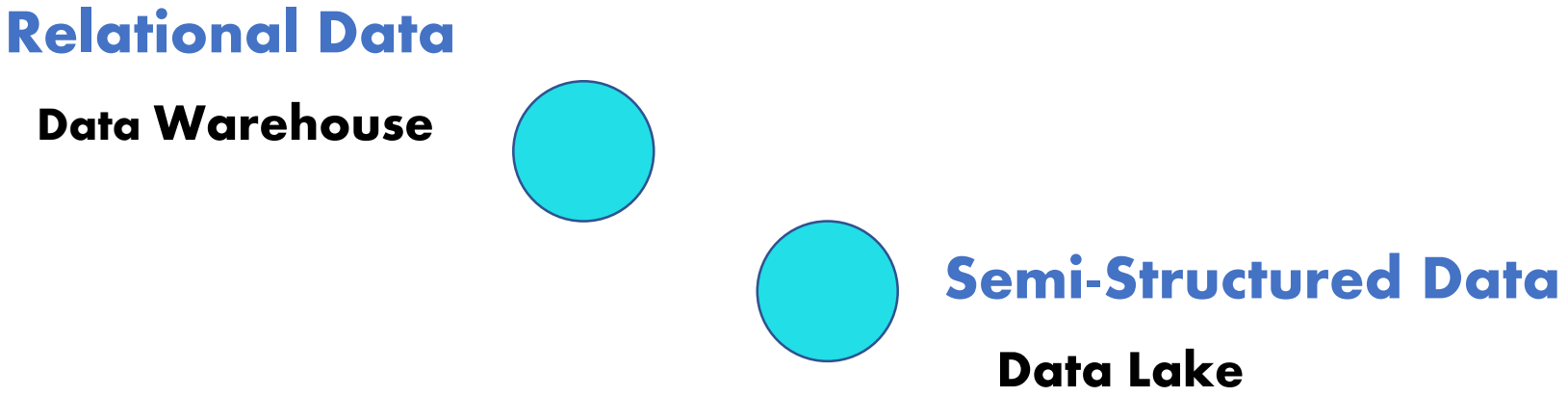
Data silos and incompatible tooling inhibit collaboration



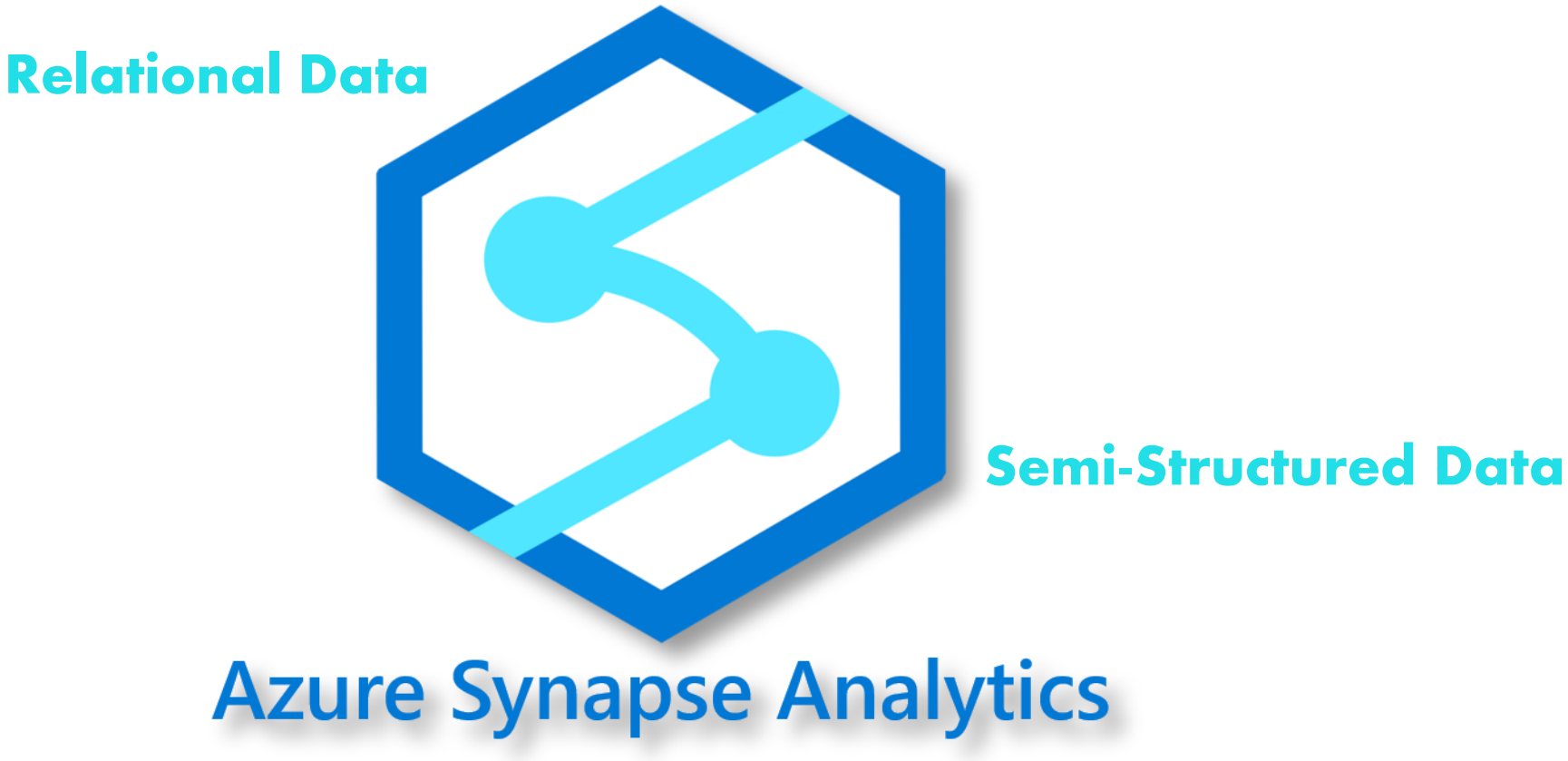
Azure Synapse



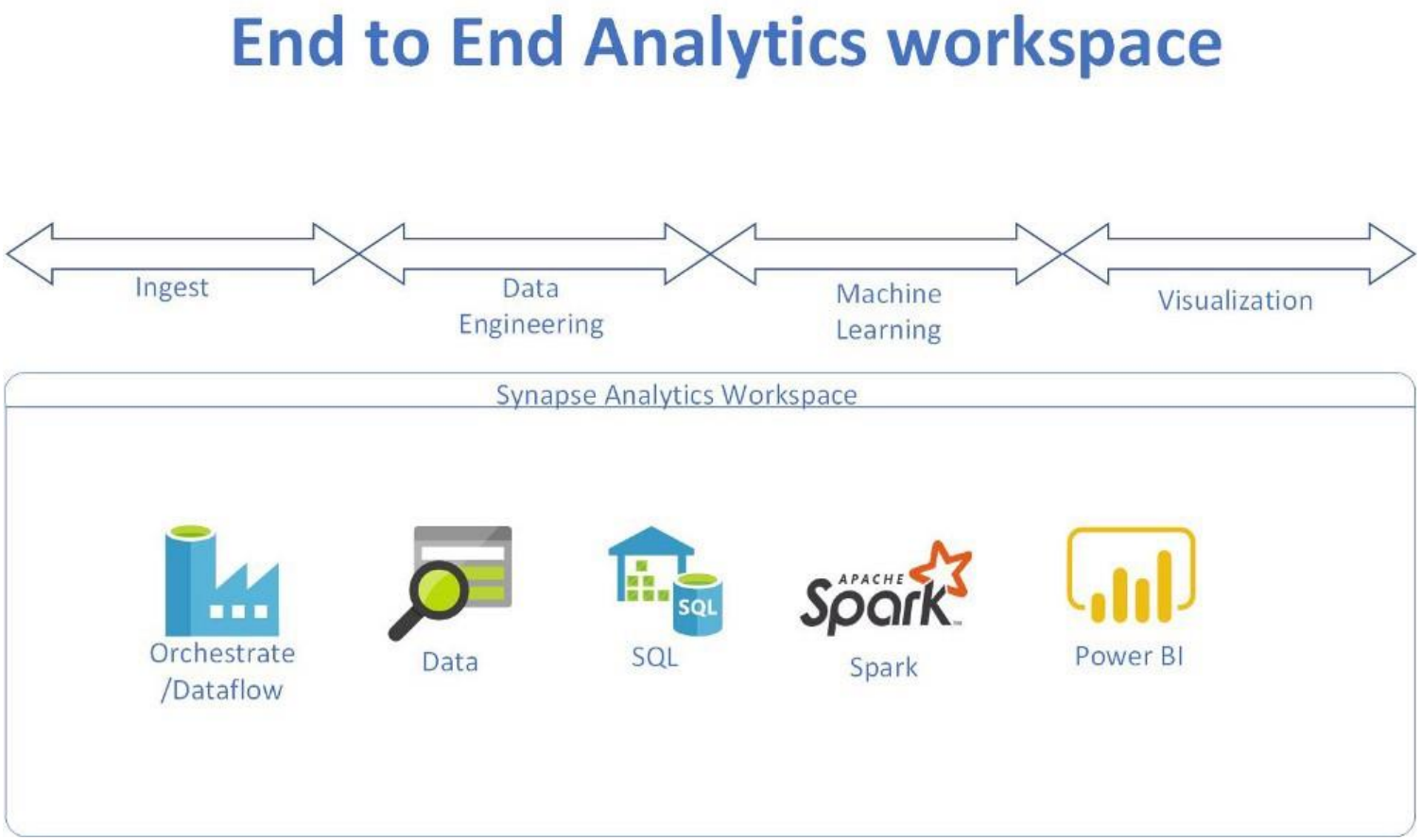
Azure Synapse Analytics



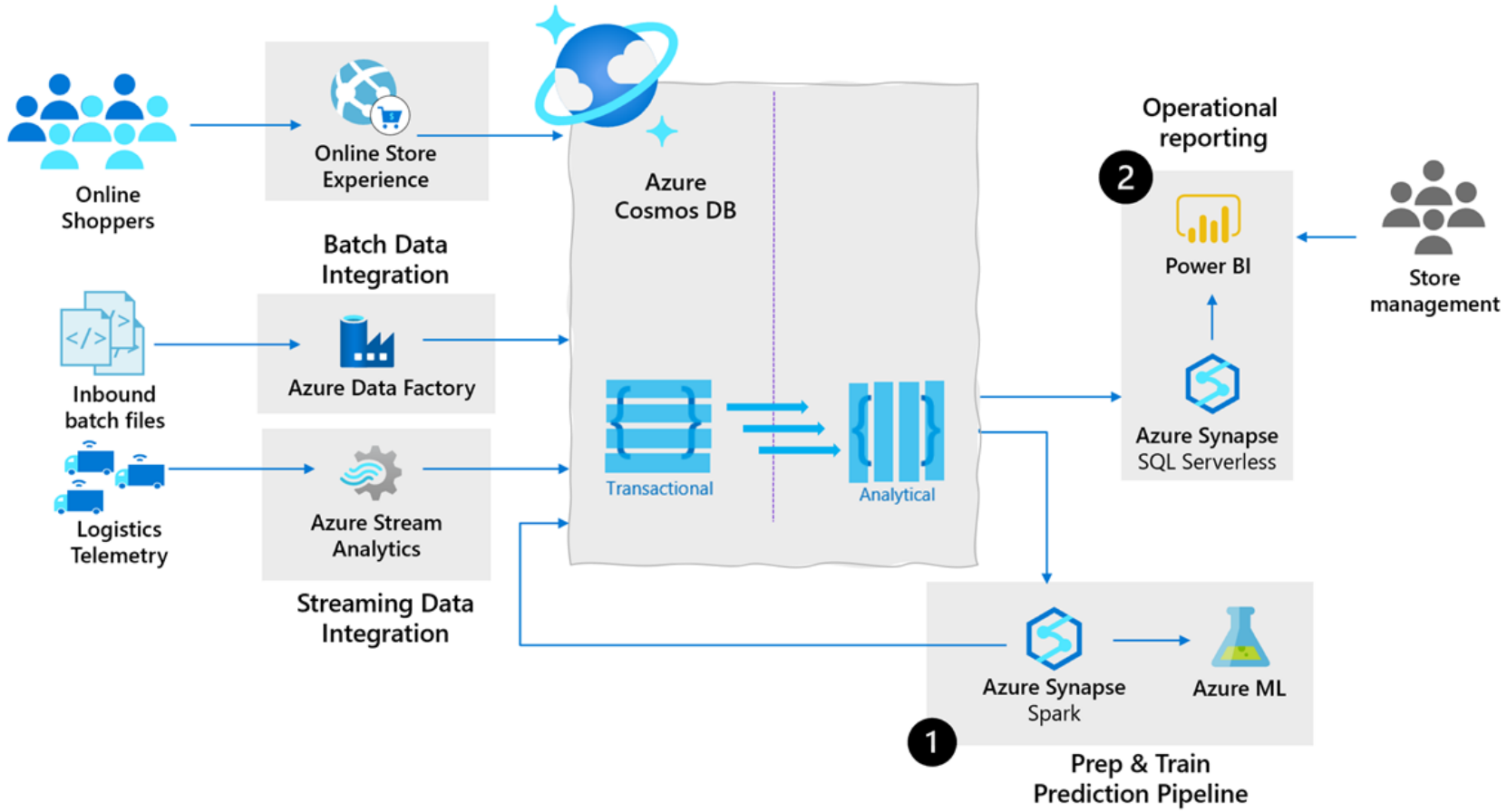
Azure Synapse Analytics



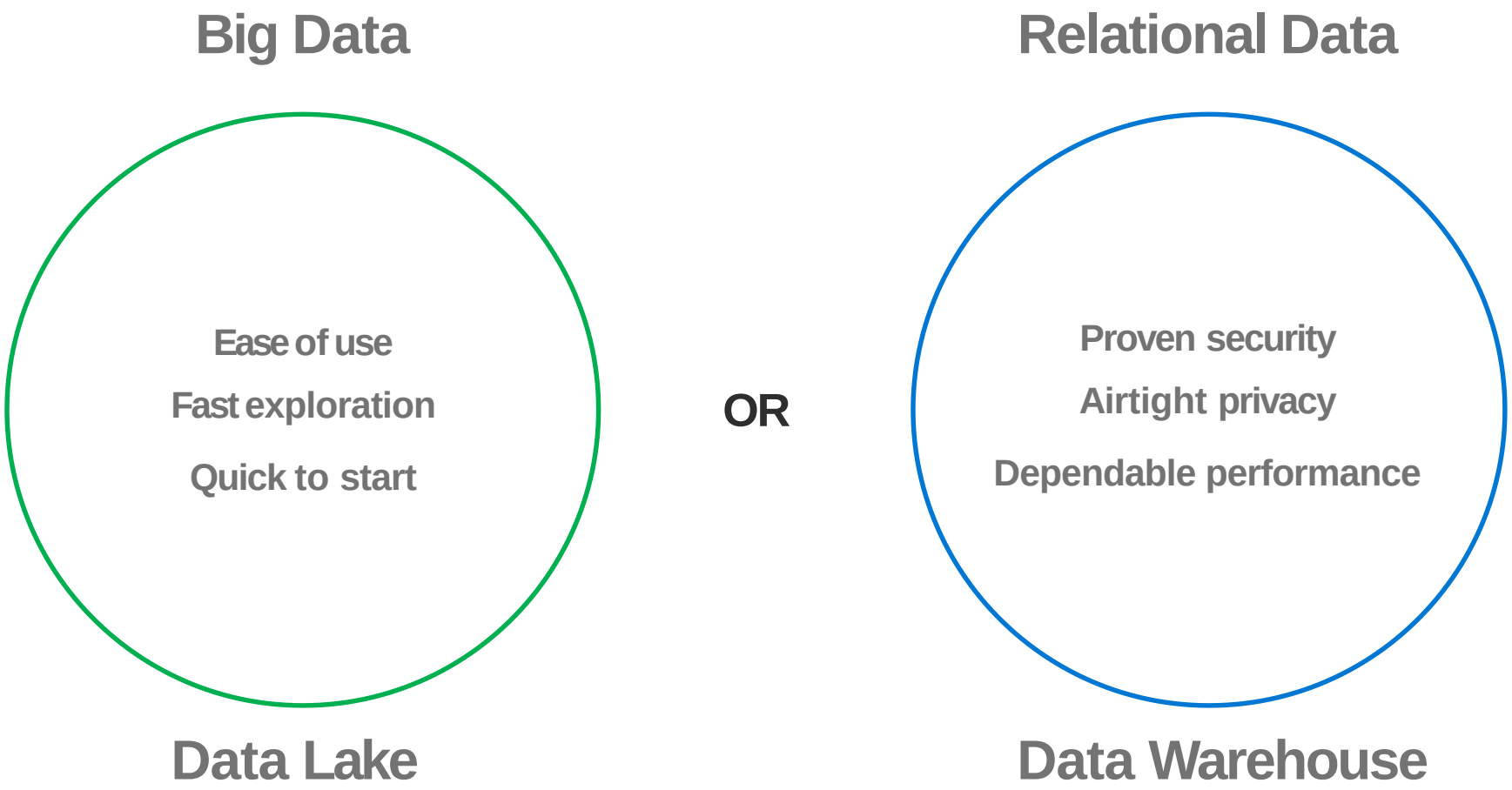
All encompassing Environment



Azure Synapse Use Case

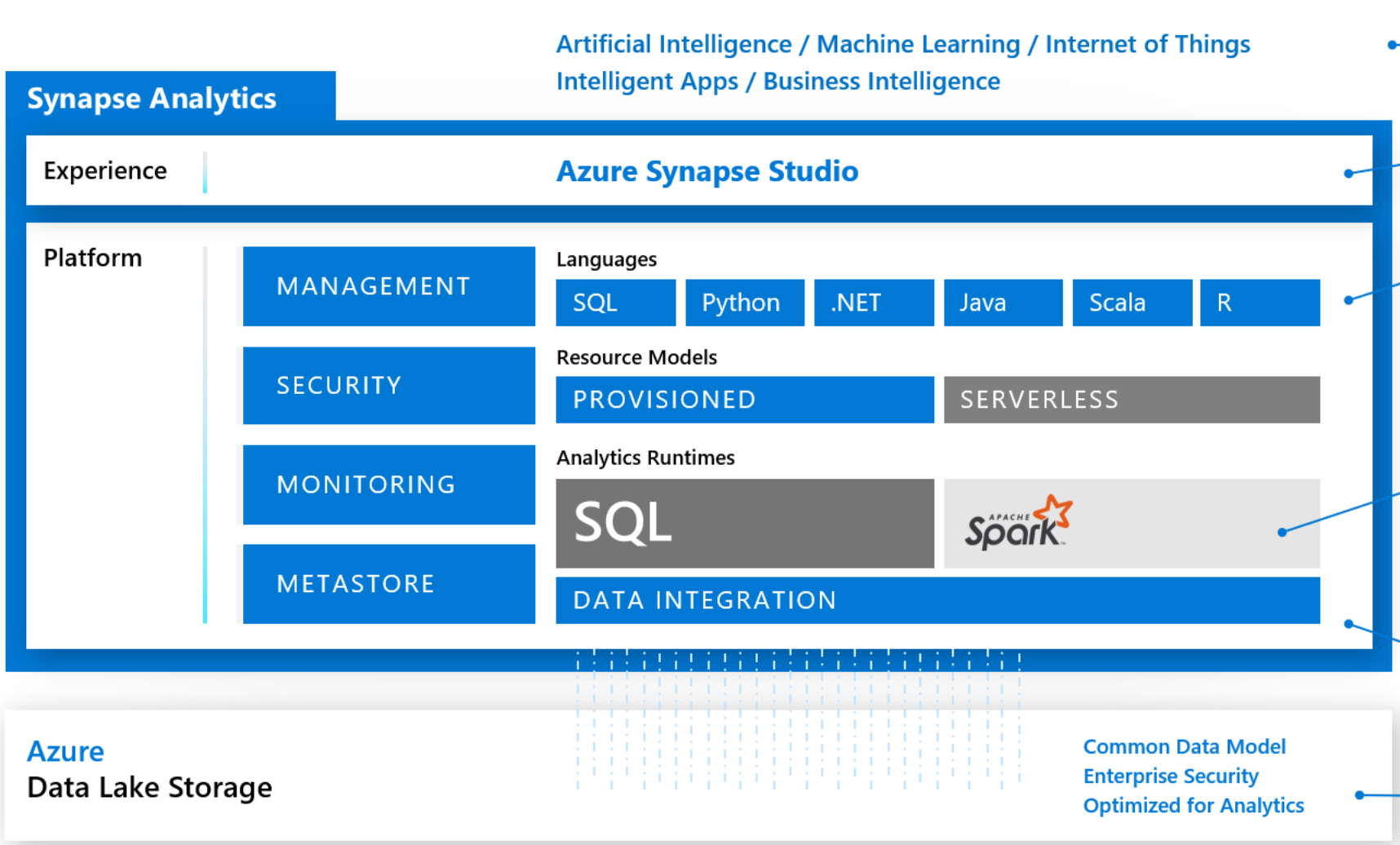


Designed to Bring this silos together



Azure Synapse Analytics

Limitless analytics service with unmatched time to insight



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

Preview

Synapse Workspaces

Controlled Access to Data	Collaboration Container	Integrates with Power BI
Azure Data Lake Gen 2	Control access to data	SQL Pools
SQL On-Demand	Spark	Notebook

Azure Data Lake Gen 2

Hierarchical Name Spaces

File System Name is the name of the default directory

Hadoop Access

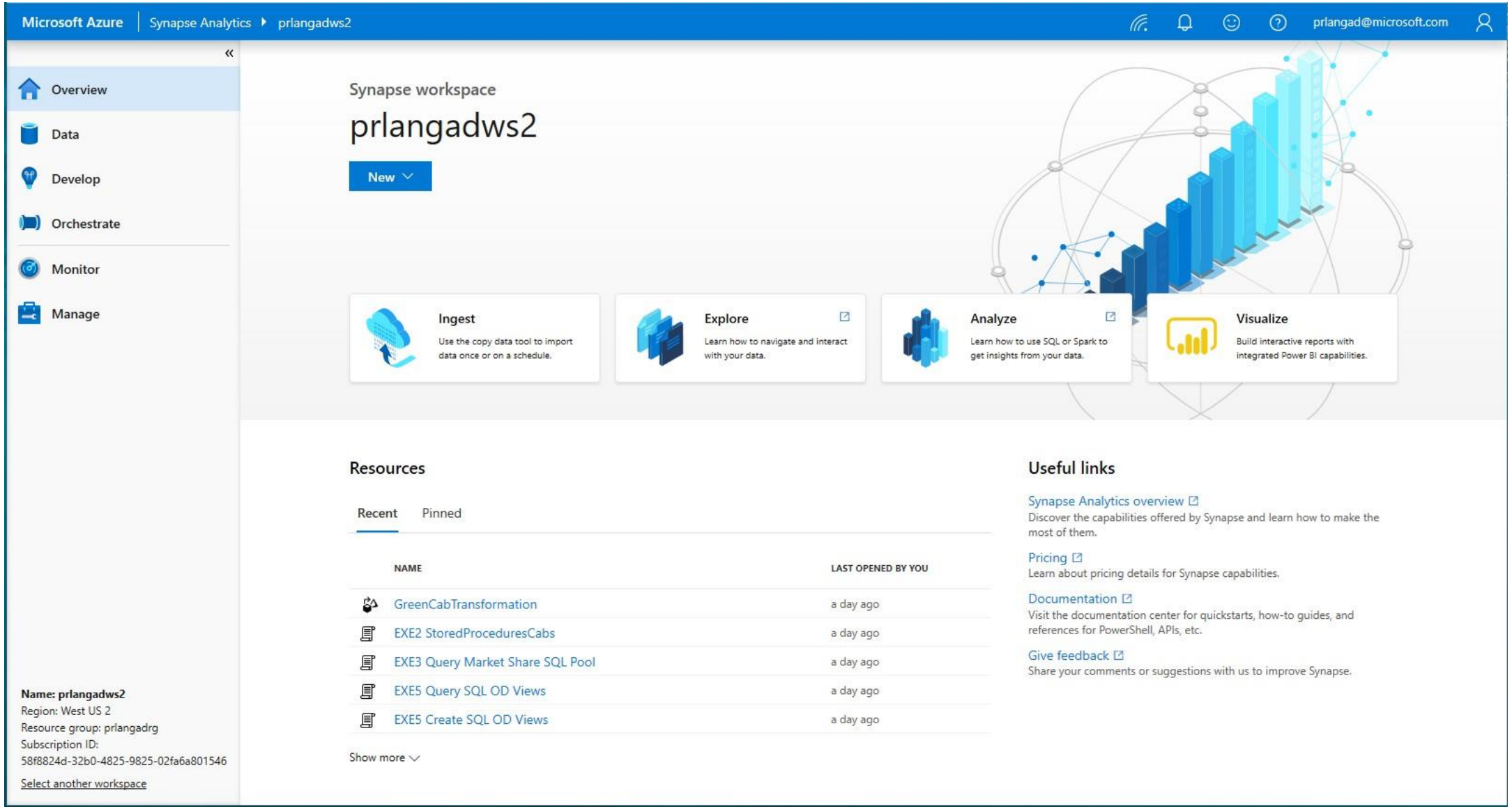
Logging

Synapse Studio

- Preview Feature
- Create a workspace first
- Dev Environment
- Create New Storage
- Manage Pools
- Notebooks

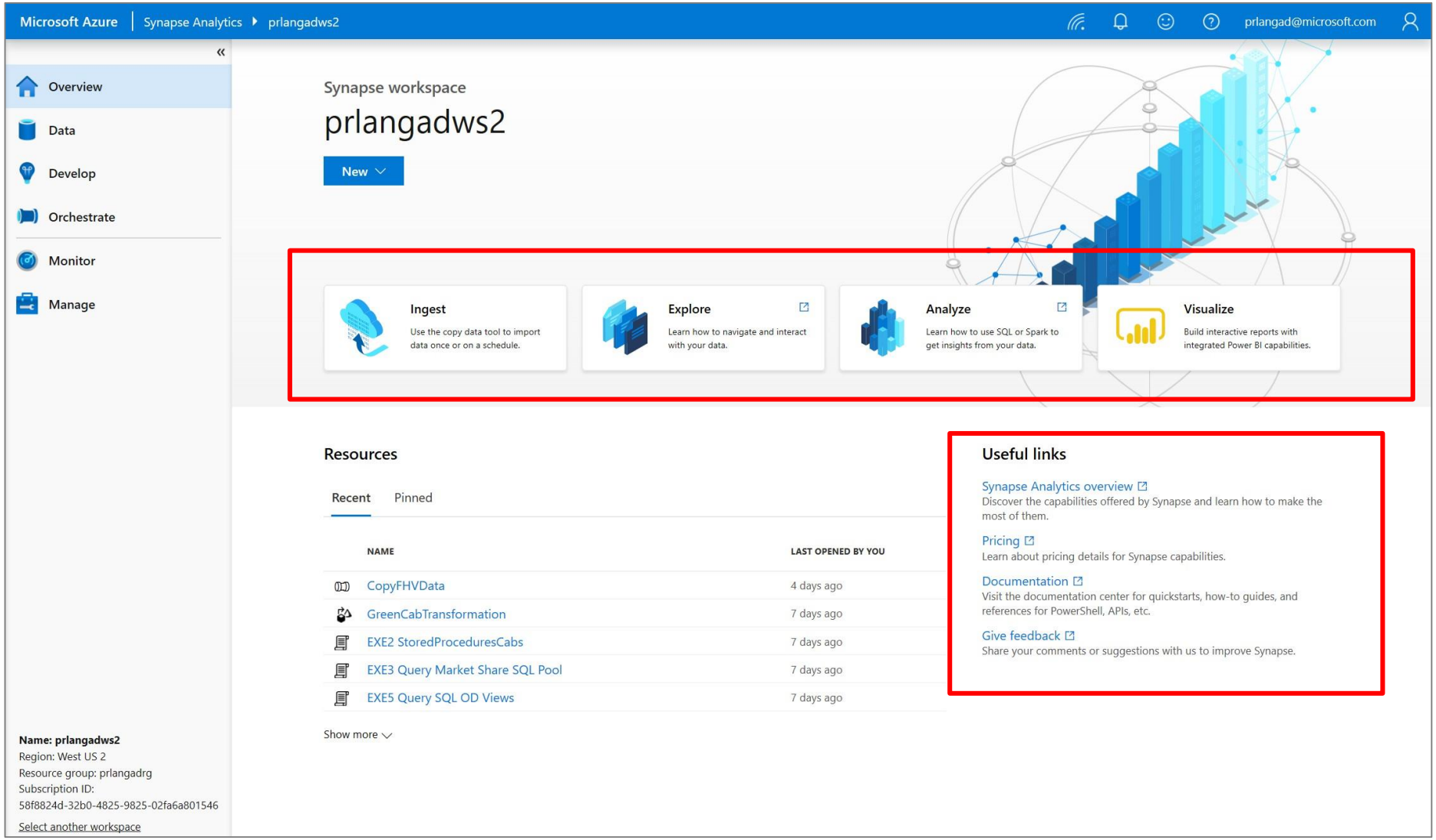
Studio

A single place for Data Engineers, Data Scientists, and IT Pros to collaborate on enterprise analytics



Azure Studio Overview

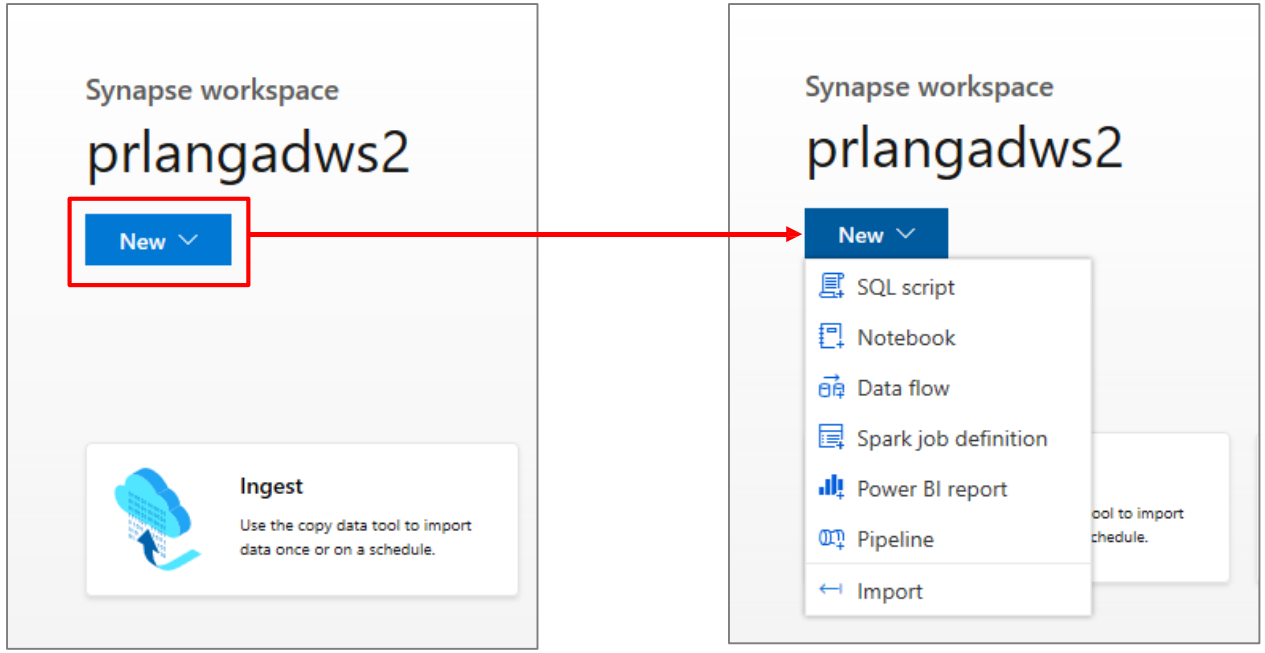
It is a starting point for the activities with key links to tasks, artifacts and documentation



Overview Hub

New dropdown – offers quickly start work item

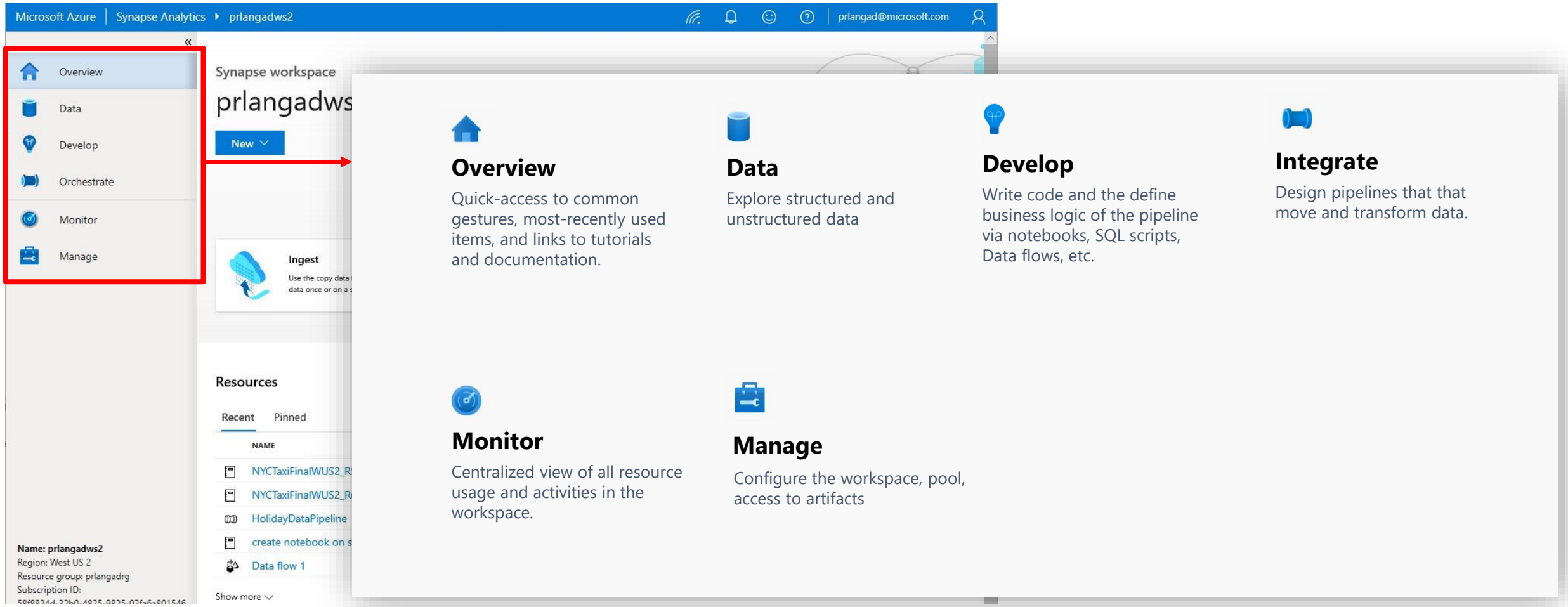
Recent & Pinned – Lists recently opened code artifacts. Pin selected ones for quick access



Synapse Studio

Synapse Studio divided into **Activity hubs**

These organize the tasks needed for building analytics solutions



Demonstration

Azure Synapse Workspace

Diving into Data Tasks

- Writing T-SQL
- Ingestion
- Spark
- Provisioned

Synapse Pools

- Serverless
- SQL
- Apache Spark
- Integration Runtimes

Serverless Pools

Auto Created by Synapse

Fully TSQL compliant

Flat file Examination

Charged $5 per Terabyte

Management of Serverless Pools

- Currently No Options for Changing Pool Size
- Not charged when running, only querying

SQL Pools

USED FOR DATA WAREHOUSES

CHARGED FOR EVERY MINUTE IN USE

SCALABLE AND SEPARATE FROM DATA

CAN CHANGE CONFIGURATION QUICKLY

Decreasing Azure Spend

- Create Pools for different tasks
- Change Pool Size often
- Pause when not in use
- Evaluate use cases

Apache Spark

- Used for Machine Learning or ETL
- Python, .Net or Scala
- Similar functionality to Databricks
- Notebook based
- Supports Spark version 2.4

Manage the Pool sizes

Evaulate pool sizes

Shorten the time to autostop

Cost Control for Spark

Integration Runtimes

- Spark Pool
- AutoResolveIntegrationRuntime
- Required for Orchestration
- Generate Others

Minimizing Azure Spend

Monitor Pipelines

Look at the Active flow Debug sessions

Create Multiple Integration Runtimes

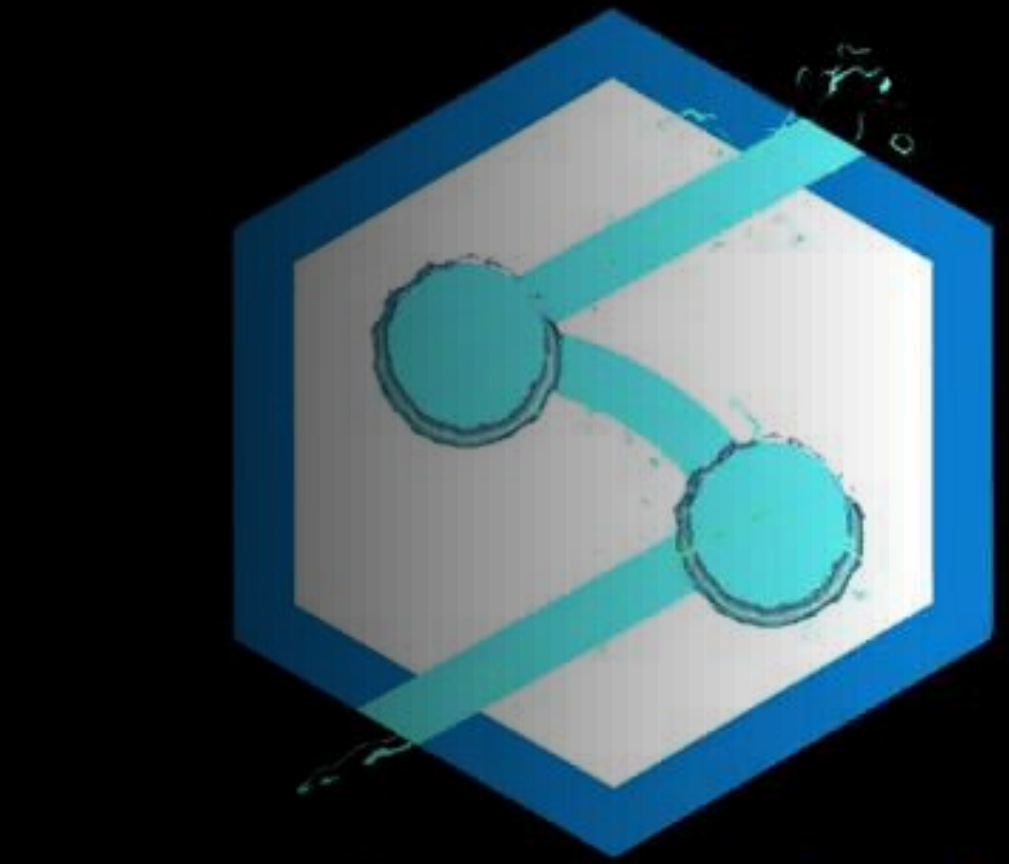
Demonstration

Creating and Examining Pools

Quiz!

Poll Question

https://geni.us/SynapsePrecon2



Azure Synapse Analytics

Analyzing Data with SQL Serverless

Data Warehousing in Synapse

1

Serverless Pools for a Data Lakehouse

2

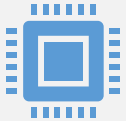
SQL Pools for Traditional Data Warehouse

Ingest and Store – Loading staging tables

Indexing

Use Heap tables

Speed load performance by staging data in heap tables and temporary tables prior to running transformations.



Only load to a CCI table if the test requires a load to a single table, then complex end-user queries against that table.

as

SQL Serverless

(aka SQL On-Demand)

An interactive query service that provides T-SQL queries over high scale data in Azure Storage.

Benefits

Serverless

No infrastructure

Pay only for query execution

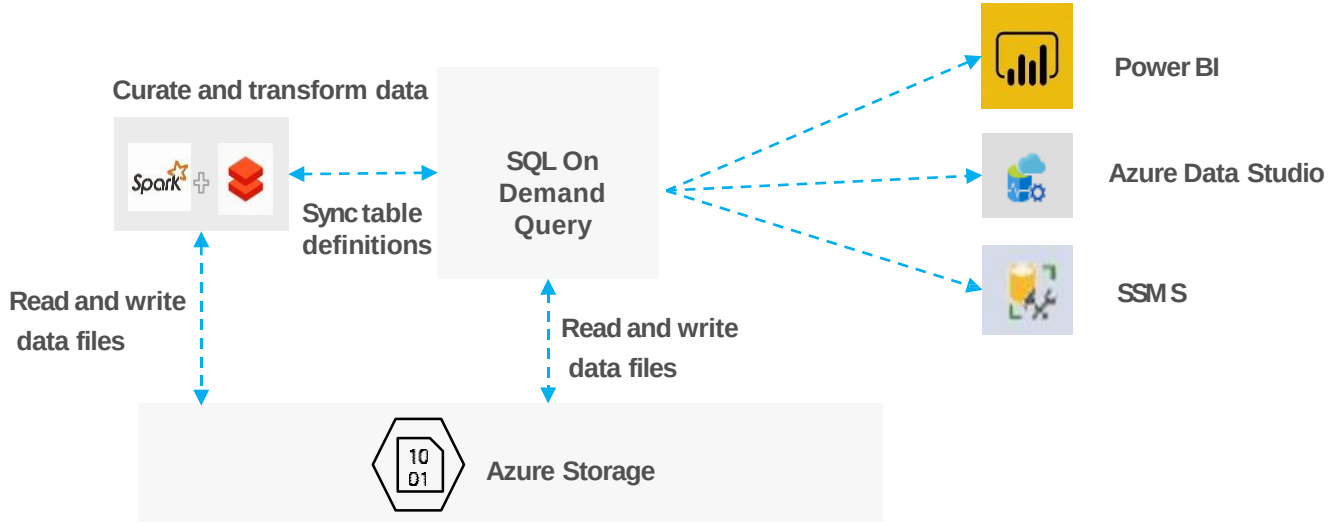
No ETL

Offers security

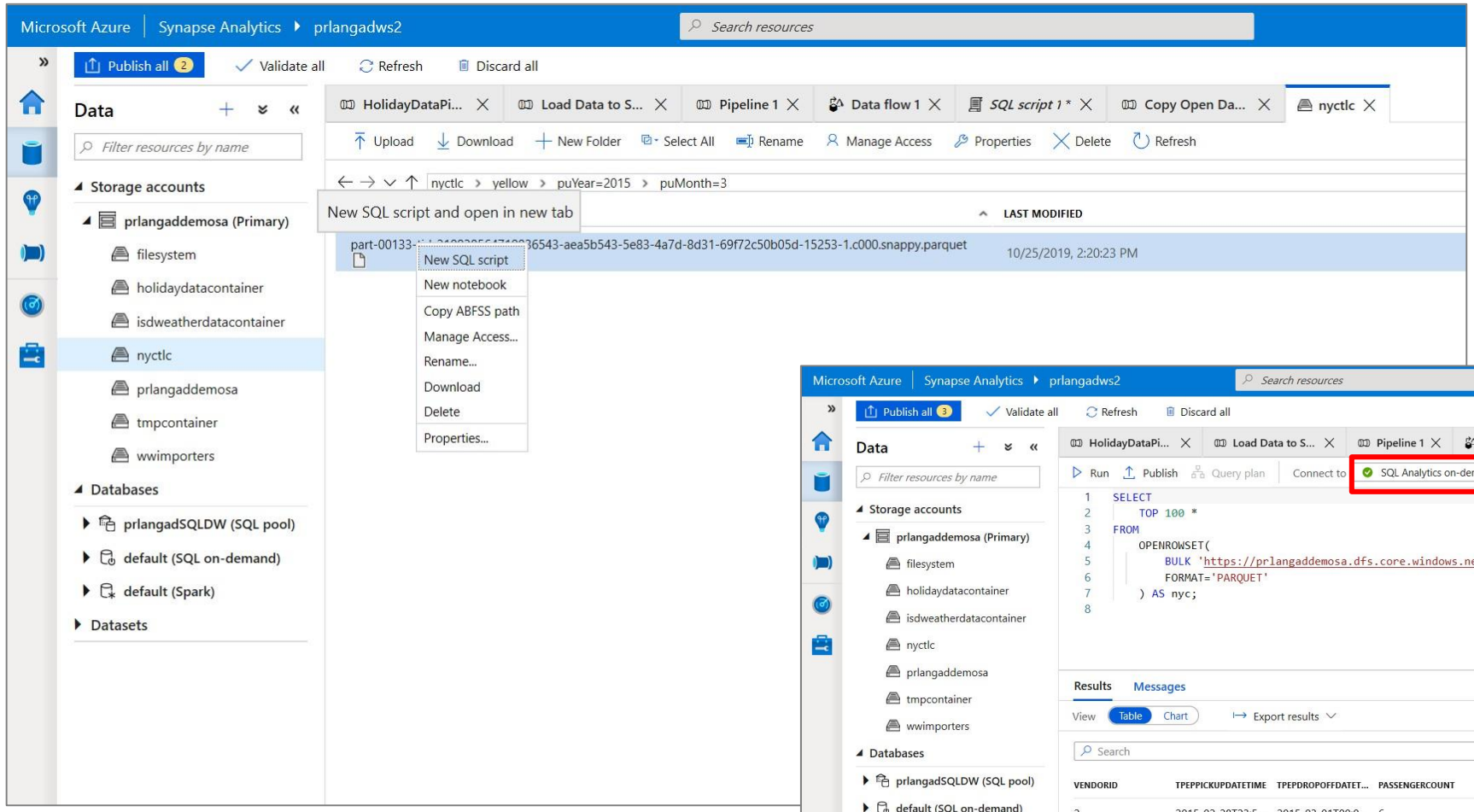
Data integration with Databricks, HDInsight T-SQL syntax to query data

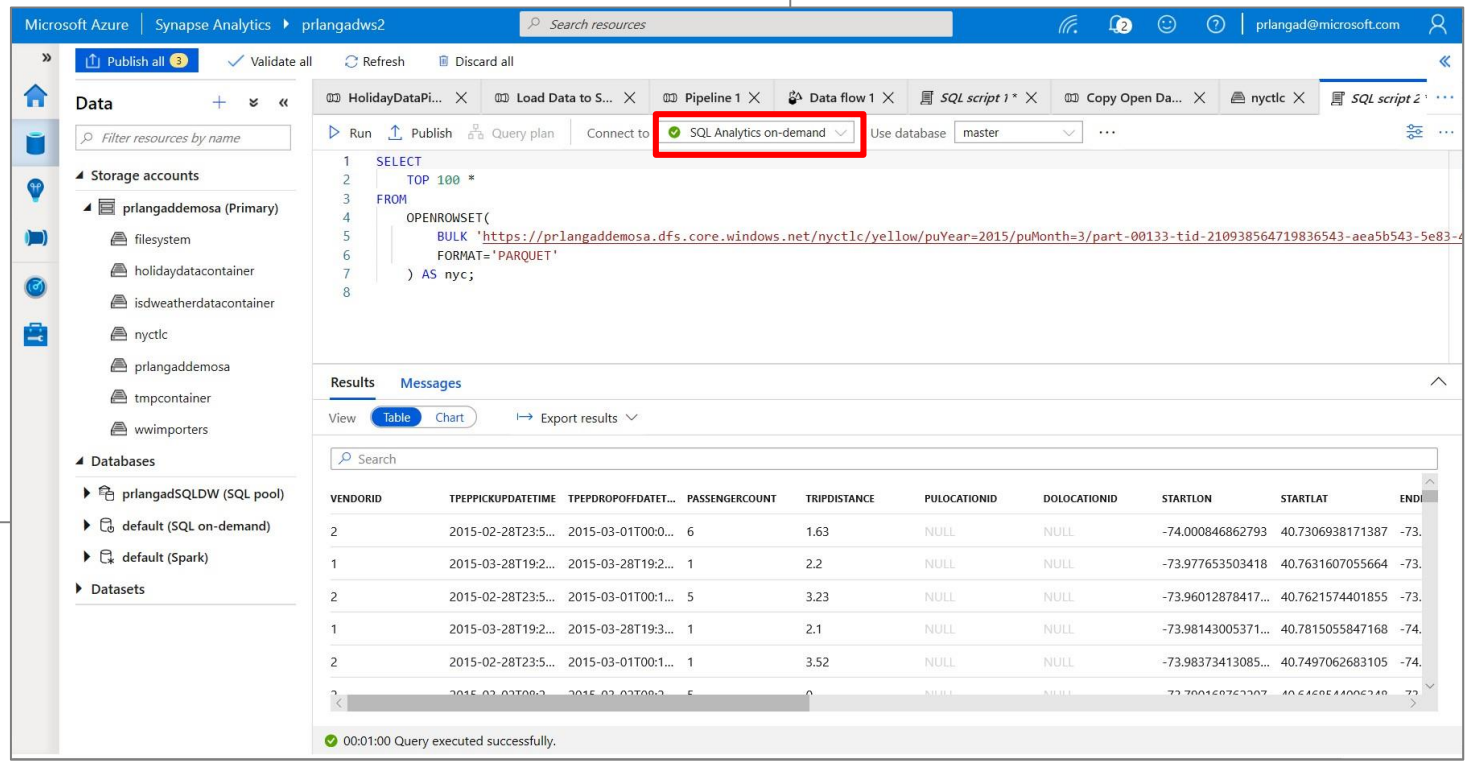
Supports data in various formats (Parquet, CSV, JSON)

Support for BI ecosystem



SQL Serverless – Querying on storage





SQL Serverless – Querying CSV File

Uses OPENROWSET function to access data

Benefits

- **Ability to read CSV File with no header row, Windows style new line**
- **No header row, Unix-style new line**
- **Header row, Unix-style new line**
- **Header row, Unix-style new line, quoted**
- **Header row, Unix-style new line, escape**
- **Header row, Unix-style new line, tab-delimited**
- **without specifying all columns**

```
SELECT *
FROM OPENROWSET(
      BULK 'https://XXX.blob.core.windows.net/csv/population/population.csv',
      FORMAT = 'CSV',
      FIELDTERMINATOR =',',
      ROWTERMINATOR = '\n'
   )
WITH (
   [country_code] VARCHAR (5) COLLATE Latin1_General_BIN2,
   [country_name] VARCHAR (100) COLLATE Latin1_General_BIN2,
   [year] smallint,
   [population] bigint
) AS [r]
WHERE
   country_name = 'Luxembourg'
   AND year = 2017
```

	country_code	country_name	year	population
1	LU	Luxembourg	2017	594130

SQL Serverless – Querying CSV Files

Read CSV file - header row, Unix-style new line

```
SELECT *
FROM OPENROWSET(
      BULK 'https://XXX.blob.core.windows.net/csv/population-unix-hdr/population.csv',
      FORMAT = 'CSV',
      FIELDTERMINATOR =',',
      ROWTERMINATOR = '0x0a',
      FIRSTROW = 2
   )
   WITH (
      [country_code] VARCHAR (5) COLLATE Latin1_General_BIN2,
      [country_name] VARCHAR (100) COLLATE Latin1_General_BIN2,
      [year] smallint,
      [population] bigint
   ) AS [r]
WHERE
   country_name = 'Luxembourg'
   AND year = 2017
```

	country_code	country_name	year	population
1	LU	Luxembourg	2017	594130

Read CSV file - without specifying all columns

```
SELECT
   COUNT(DISTINCT country_name) AS countries
FROM OPENROWSET(
      BULK 'https://XXX.blob.core.windows.net/csv/population/population.csv
',
      FORMAT = 'CSV',
      FIELDTERMINATOR =',',
      ROWTERMINATOR = '\n'
   )
WITH (
      [country_name] VARCHAR (100) COLLATE Latin1_General_BIN2 2
) AS [r]
```

	countries
1	228

SQL Serverless– Querying folders

Uses OPENROWSET function to access data from multiple files or folders

Benefits

- Offers reading multiple files/folders through usage of wildcards
- Offers reading specific file/folder
- Supports use of multiple wildcards

```
SELECT YEAR(pickup_datetime) as [year], SUM(passenger_count) AS passengers_total,
COUNT(*) AS [rides_total]
FROM OPENROWSET(
BULK 'https://XXX.blob.core.windows.net/csv/taxi/*.*',
FORMAT = 'CSV'
, FIRSTROW = 2 )
WITH (
        vendor_id VARCHAR(100) COLLATE Latin1_General_BIN2,
        pickup_datetime DATETIME2,
        dropoff_datetime DATETIME2,
        passenger_count INT,
        trip_distance FLOAT, rate_code
        INT,
        store_and_fwd_flag VARCHAR(100) COLLATE Latin1_General_BIN2,
        pickup_location_id INT,
        dropoff_location_id INT,
        payment_type INT,
        fare_amount FLOAT,
        extra FLOAT, mta_tax FLOAT,
        tip_amount FLOAT,
        tolls_amount FLOAT,
        improvement_surcharge FLOAT,
        total_amount FLOAT
        ) AS nyc
GROUP BY YEAR(pickup_datetime)
ORDER BY YEAR(pickup_datetime)
```

	year	passengers_total	rides_total
1	2001	14	10
2	2002	29	16
3	2003	22	16
4	2008	378	188
5	2009	594	353
6	2016	102093687	61758523
7	2017	184464988	113496932
8	2018	86272771	53925040
9	2019	37	29
…	2020	6	6

SQL Serverless– Querying folders

Read all files from multiple folders

```
SELECT YEAR(pickup_datetime) as [year],
    SUM(passenger_count) AS passengers_total,
    COUNT(*) AS [rides_total]
FROM OPENROWSET(
    BULK 'https://XXX.blob.core.windows.net/csv/t*i/',
        FORMAT = 'CSV',
        FIRSTROW = 2    )
    WITH (
        vendor_id VARCHAR(100) COLLATE Latin1_General_BIN2,
        pickup_datetime DATETIME2,
        dropoff_datetime DATETIME2,
        passenger_count INT,
        trip_distance FLOAT,
        <… columns>
    ) AS nyc
GROUP BY YEAR(pickup_datetime)
ORDER BY YEAR(pickup_datetime)
```

	year	passengers_total	rides_total
1	2001	14	10
2	2002	29	16
3	2003	22	16
4	2008	378	188
5	2009	594	353
6	2016	102093687	61758523
7	2017	184464988	113496932
8	2018	86272771	53925040
9	2019	37	29
…	2020	6	6

Read subset of files in folder

```
SELECT
    payment_type,
    SUM(fare_amount) AS fare_total
FROM OPENROWSET(
    BULK 'https://XXX.blob.core.windows.net/csv/taxi/yellow_tripdata_2017-*.csv',
        FORMAT = 'CSV',
        FIRSTROW = 2    )
    WITH (
        vendor_id VARCHAR(100) COLLATE Latin1_General_BIN2,
        pickup_datetime DATETIME2,
        dropoff_datetime DATETIME2,
        passenger_count INT,
        trip_distance FLOAT,
        <…columns>
    ) AS nyc
GROUP BY payment_type
ORDER BY payment_type
```

	payment_type	fare_total
1	1	1026072325.579…
2	2	441093322.8000…
3	3	10435183.04
4	4	3304550.99
5	5	14

SQL Serverless– Querying specific files

Overview

filename – Provides file name that originates row result

filepath – Provides full path when no parameter is passed or part of path when parameter is passed that originates result

Benefits

Provides source name/path of file/folder for row result set

Example of filename function

```
SELECT
    r.filename() AS [filename]
    ,COUNT_BIG(*) AS [rows]
FROM OPENROWSET(
        BULK 'https://XXX.blob.core.windows.net/csv/taxi/yellow_tripdata_2017-1*.csv',
        FORMAT = 'CSV',
        FIRSTROW = 2
    )
    WITH (
        vendor_id INT,
        pickup_datetime DATETIME2,
        dropoff_datetime DATETIME2,
        passenger_count SMALLINT,
        trip_distance FLOAT,
        <...columns>
    ) AS [r]

GROUP BY r.filename()

ORDER BY [filename]
```

	filename	rows
1	yellow_tripdata_2017-10.csv	9768815
2	yellow_tripdata_2017-11.csv	9284803
3	yellow_tripdata_2017-12.csv	9508276

SQL Serverless– Querying specific files

Example of filepath function

```
SELECT
    r.filepath() AS filepath
    ,r.filepath(1) AS [year]
    ,r.filepath(2) AS [month]
    ,COUNT_BIG(*) AS [rows]
FROM OPENROWSET(
        BULK 'https://XXX.blob.core.windows.net/csv/taxi/yellow_tripdata_*-*.csv',
        FORMAT = 'CSV',
        FIRSTROW = 2    )
WITH (
    vendor_id INT,
    pickup_datetime DATETIME2,
    dropoff_datetime DATETIME2,
    passenger_count SMALLINT,
    trip_distance FLOAT,
   <... columns>
) AS [r]

WHERE r.filepath(1) IN ('2017')
    AND r.filepath(2) IN ('10', '11', '12')

GROUP BY    r.filepath() ,r.filepath(1) ,r.filepath(2)
ORDER BY    filepath
```

filepath	year	month	rows
https://XXX.blob.core.windows.net/csv/taxi/yellow_tripdata_2017-10.csv	2017	10	9768815
https://XXX.blob.core.windows.net/csv/taxi/yellow_tripdata_2017-11.csv	2017	11	9284803
https://XXX.blob.core.windows.net/csv/taxi/yellow_tripdata_2017-12.csv	2017	12	9508276

SQL Serverless– Querying Parquetfiles

Overview

Uses OPENROWSET function to access data

Benefits

Ability to specify column names of interest

Offers auto reading of column names and data types

Provides target specific partitions using filepath function

```
SELECT
      YEAR(pickup_datetime),
      passenger_count,
      COUNT(*) AS cnt
FROM
   OPENROWSET(
      BULK 'https://XXX.blob.core.windows.net/parquet/taxi/*/*/*',
      FORMAT='PARQUET'
   ) WITH (
      pickup_datetime DATETIME2,
      passenger_count INT
   ) AS nyc
GROUP BY
   passenger_count,
   YEAR(pickup_datetime)
ORDER BY
   YEAR(pickup_datetime),
   passenger_count
```

	(No column name)	passenger_count	cnt
1	2016	0	2557
2	2016	1	43735845
3	2016	2	9056714
4	2016	3	2610541
5	2016	4	1309639
6	2016	5	3086097
7	2016	6	1956607

SQL Serverless – Creating views

Overview

Create views using SQL On Demand queries

Benefits

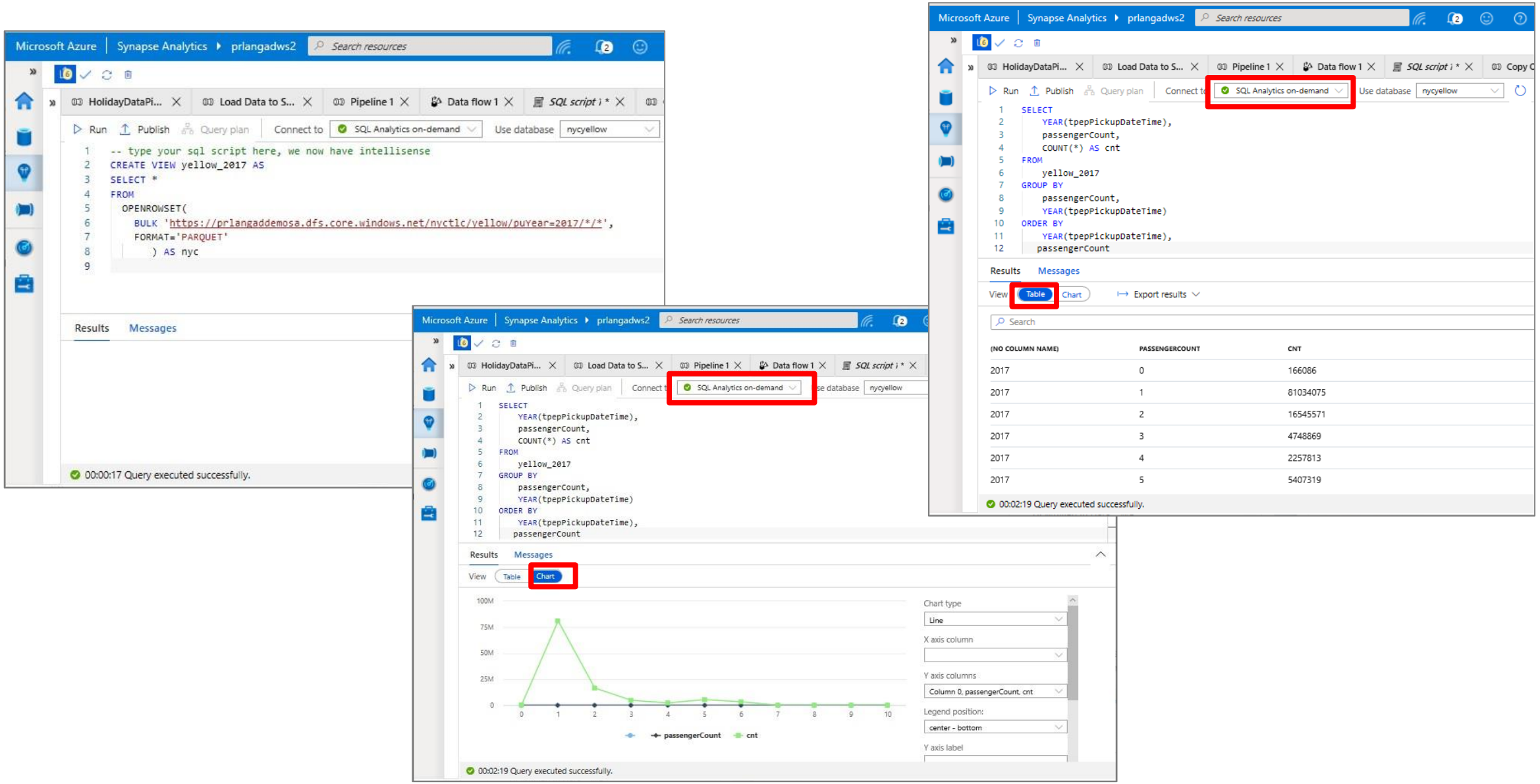
Works same as standard views

```
USE [mydbname]
GO

IF EXISTS(select * FROM sys.views where name = 'populationView')
DROP VIEW populationView
GO

CREATE VIEW populationView AS
SELECT *
FROM OPENROWSET(
      BULK 'https://XXX.blob.core.windows.net/csv/population/population.csv',
      FORMAT = 'CSV',
      FIELDTERMINATOR =',',
      ROWTERMINATOR = '\n'
   )
WITH (
   [country_code] VARCHAR (5) COLLATE Latin1_General_BIN2,
   [country_name] VARCHAR (100) COLLATE Latin1_General_BIN2,
   [year] smallint,
   [population] bigint
) AS [r]
```

```
SELECT
   country_name, population
FROM populationView
WHERE
   [year] = 2019
ORDER BY
   [population] DESC
```

	country_name	population
1	China	1389618778
2	India	1311559204
3	United States	331883986
4	Indonesia	264935824
5	Pakistan	210797836
6	Brazil	210301591
7	Nigeria	208679114
8	Bangladesh	161062905
9	Russia	141944641
10	Mexico	127318112

SQL Serverless – Creating views



SQL Serverless– Querying JSON files

Read JSON files and provides data in tabular format

Benefits

Supports OPENJSON, JSON_VALUE and JSON_QUERY functions

```
SELECT *
FROM
   OPENROWSET(
      BULK 'https://XXX.blob.core.windows.net/json/books/book1.json',
      FORMAT='CSV',
      FIELDTERMINATOR ='0x0b',
      FIELDQUOTE = '0x0b',
      ROWTERMINATOR = '0x0b'
   )
   WITH (
      jsonContent varchar(8000)
   ) AS [r]
```

	jsonContent
1	{"_id": "kim95", "type": "Book", "title": "Modern Databas…

SQL Serverless – Querying JSON files

Example of JSON_VALUE function

```
SELECT
   JSON_VALUE(jsonContent, '$.title') AS title,
   JSON_VALUE(jsonContent, '$.publisher') as publisher,
   jsonContent
FROM
   OPENROWSET(
      BULK 'https://XXX.blob.core.windows.net/json/books/*.json',
      FORMAT='CSV',
      FIELDTERMINATOR ='0x0b',
      FIELDQUOTE = '0x0b',
      ROWTERMINATOR = '0x0b'
   )
   WITH (
      jsonContent varchar(8000)
   ) AS [r]
WHERE
   JSON_VALUE(jsonContent, '$.title') = 'Probabilistic and Statistical Methods in Cryptology, A
n Introduction by Selected Topics'
```

	title	publisher	jsonContent
1	Probabilistic and Statistical Methods in Cryptology, An Int…	Springer	{"_id": "neuen…

Example of JSON_QUERY function

```
SELECT
   JSON_QUERY(jsonContent, '$.authors') AS authors,
   jsonContent
FROM
   OPENROWSET(
      BULK 'https://XXX.blob.core.windows.net/json/books/*.json',
      FORMAT='CSV',
      FIELDTERMINATOR ='0x0b',
      FIELDQUOTE = '0x0b',
      ROWTERMINATOR = '0x0b'
   )
   WITH (
      jsonContent varchar(8000)
   ) AS [r]
WHERE
   JSON_VALUE(jsonContent, '$.title') = 'Probabilistic and Statistical Methods in Cryptology,
An Introduction by Selected Topics'
```

	authors	jsonContent
1	["Daniel Neuenschwander"]	{"_id": "neuenschwander04", "type": "Book", "title": "Probabi…

Create External Table As Select

Creates an external table and then exports results of the Select statement.

These operations will import data into the database for the duration of the query

Steps:

1. **Create Master Key**
2. **Create Credentials**
3. **Create External Data Source**
4. **Create External Data Format**
5. **Create External Table**

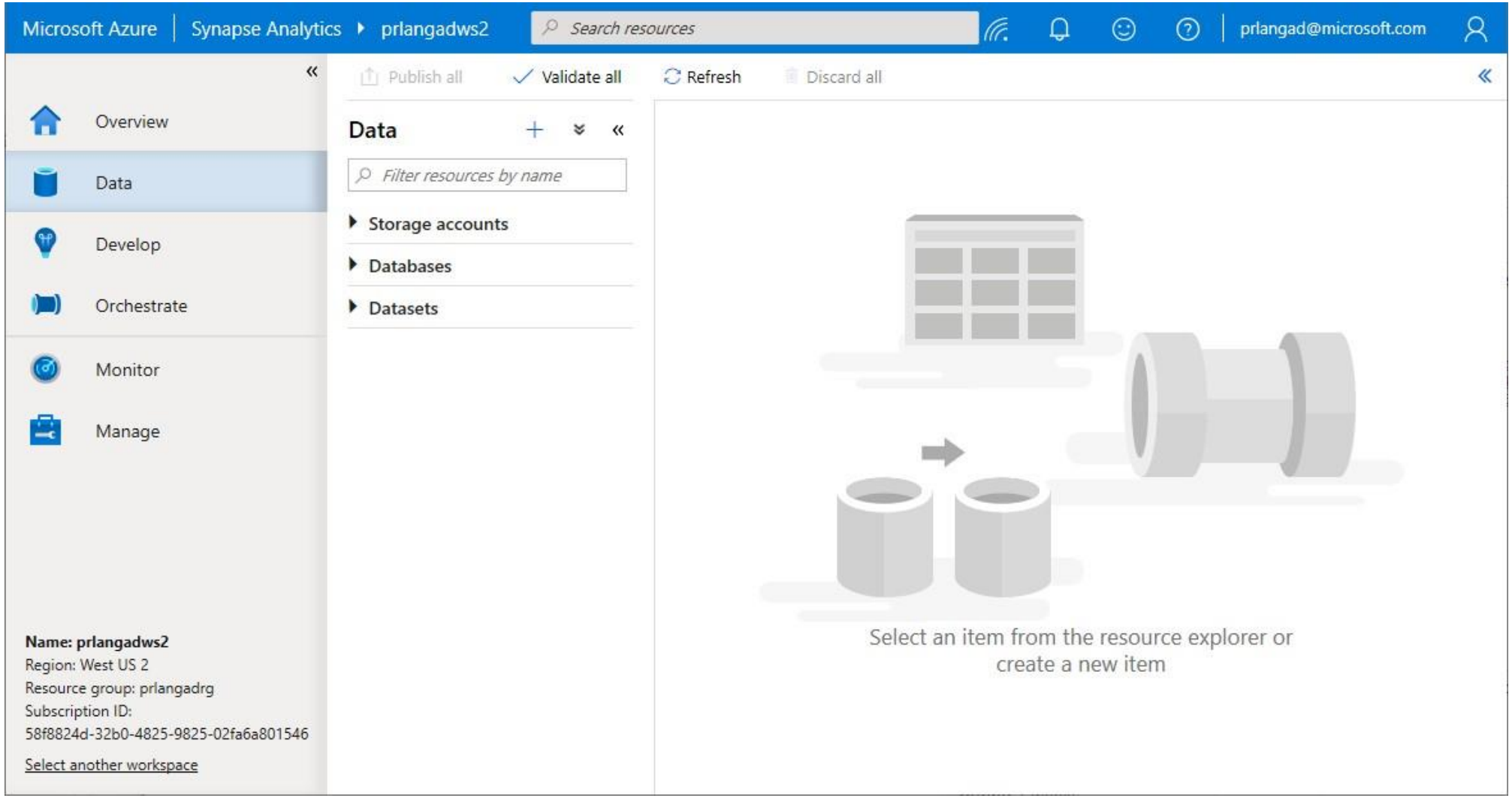
```
-- Create a database master key if one does not already exist
CREATE MASTER KEY ENCRYPTION BY PASSWORD = 'S0me!nfo'
;
-- Create a database scoped credential with Azure storage account key as the secret.
CREATE DATABASE SCOPED CREDENTIAL AzureStorageCredential
WITH
    IDENTITY = '<my_account>'
,   SECRET     = '<azure_storage_account_key>'
;
-- Create an external data source with CREDENTIAL option.
CREATE EXTERNAL DATA SOURCE  MyAzureStorage
WITH
(   LOCATION = 'wasbs://daily@logs.blob.core.windows.net/'
,   CREDENTIAL = AzureStorageCredential
,   TYPE       = HADOOP
)
-- Create an external file format
CREATE EXTERNAL FILE FORMAT MyAzureCSVFormat
WITH (FORMAT_TYPE = DELIMITEDTEXT,
     FORMAT_OPTIONS(
        FIELD_TERMINATOR = ',',
        FIRST_ROW = 2)
--Create an external table
CREATE EXTERNAL TABLE dbo.FactInternetSalesNew
WITH(
       LOCATION = '/files/Customer',
       DATA_SOURCE = MyAzureStorage,
       FILE_FORMAT = MyAzureCSVFormat
   )
AS SELECT T1.* FROM dbo.FactInternetSales T1 JOIN dbo.DimCustomer T2
ON ( T1.CustomerKey = T2.CustomerKey )
OPTION ( HASH JOIN );
```

SQL Serverless – How and Where

- Azure Data Lake G2
- Built in Serverless Pool
- Some of the code is generated for you

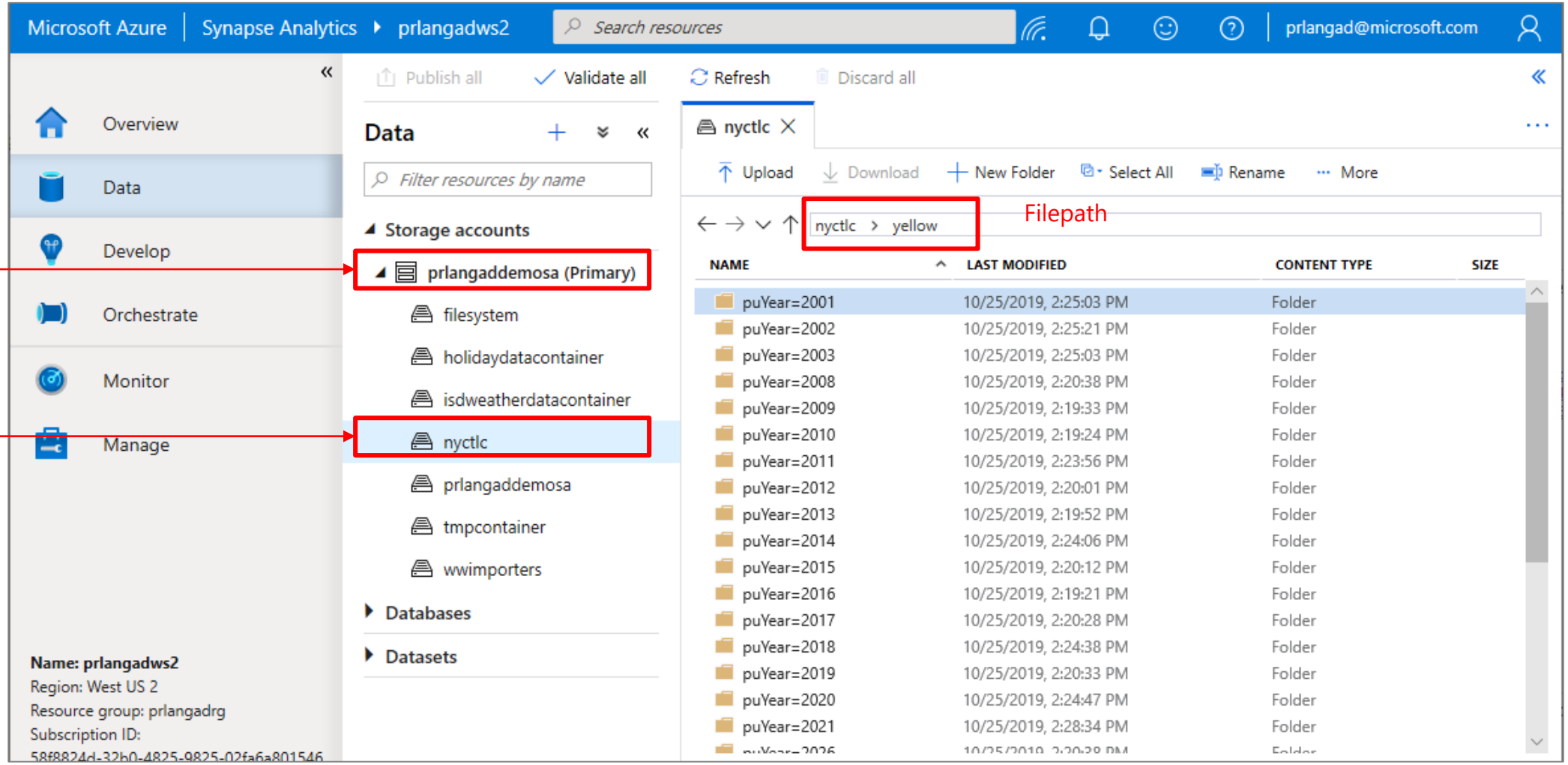
Data Hub

Explore data inside the workspace and in linked storage accounts



Data Hub –Storage Accounts

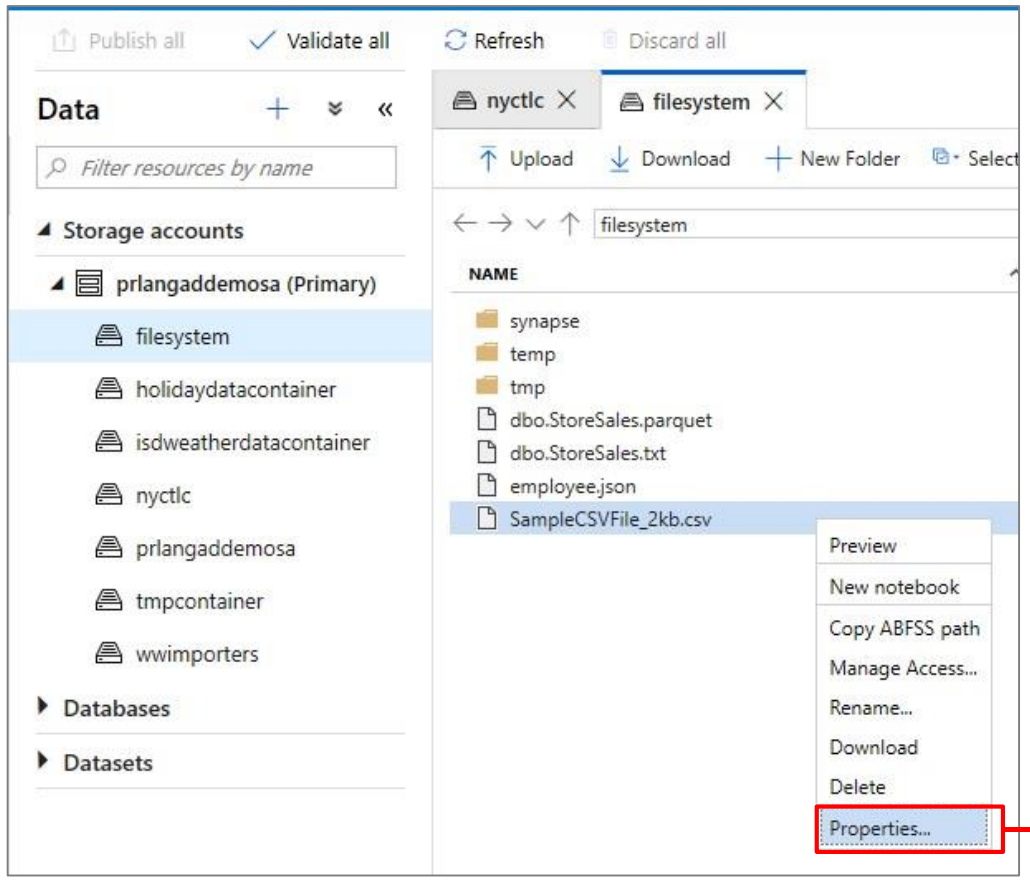
Browse Azure Data Lake Storage Gen2 accounts and filesystems
Navigate through folders to see data

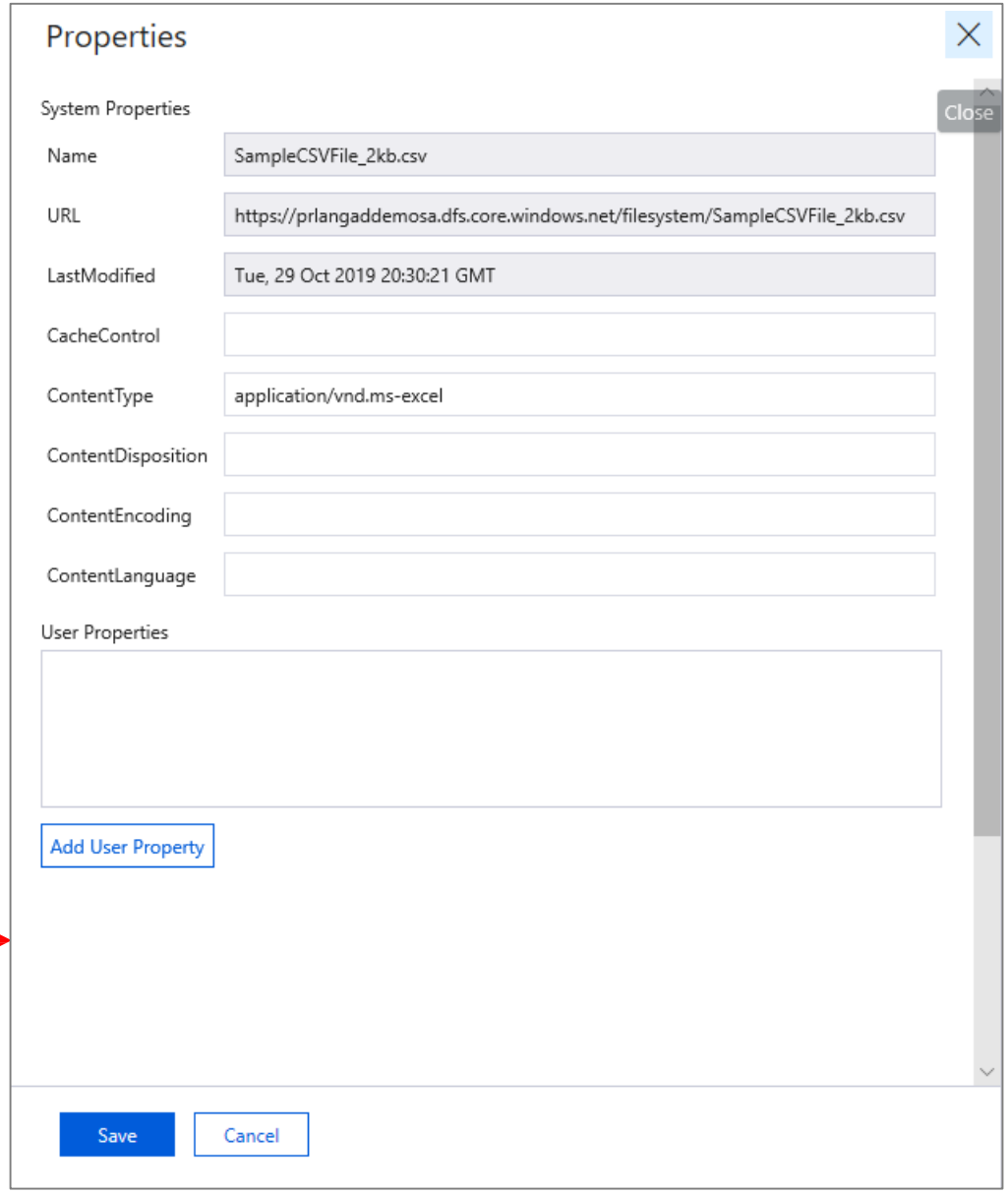


ADLS Gen2 Account

Container (filesystem)

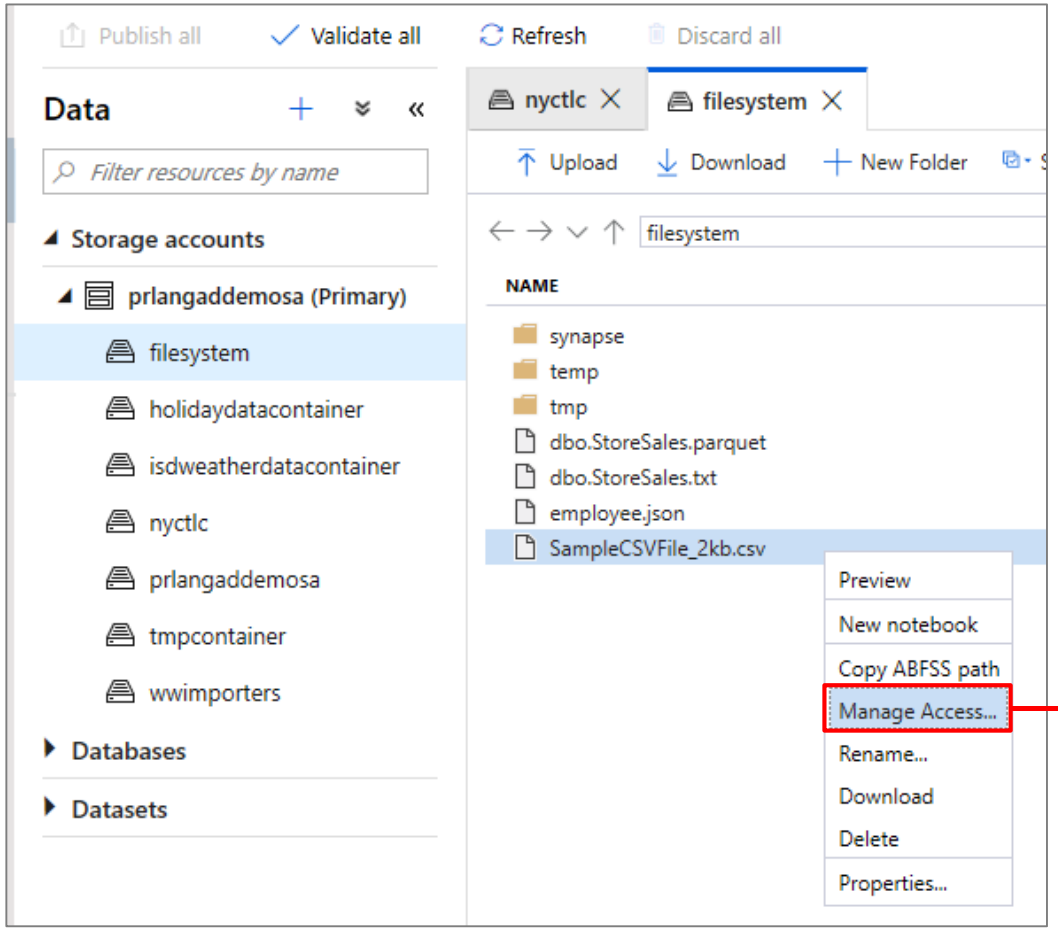
Data Hub – Storage accounts

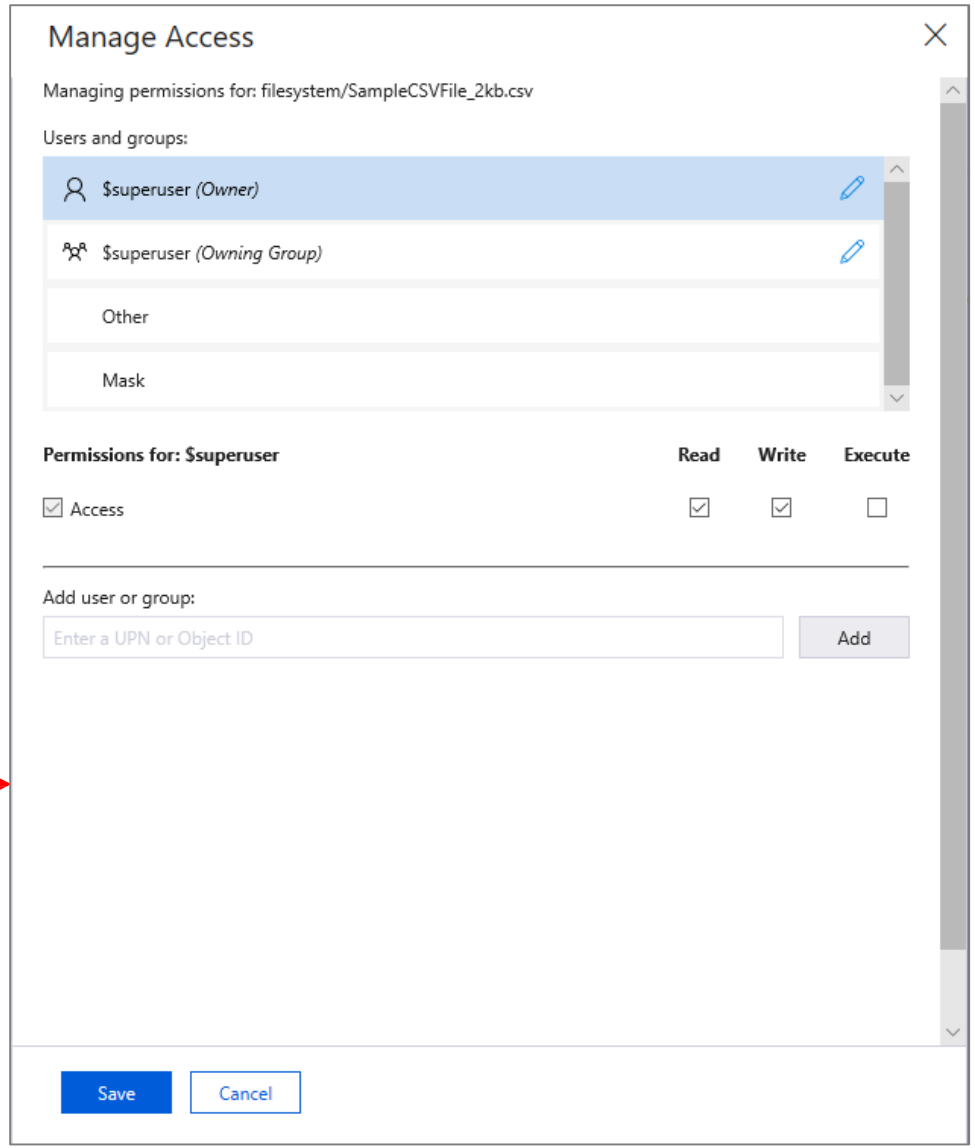




Data Hub – Storage accounts

Manage Access - Configure standard POSIX ACLs on files and folders

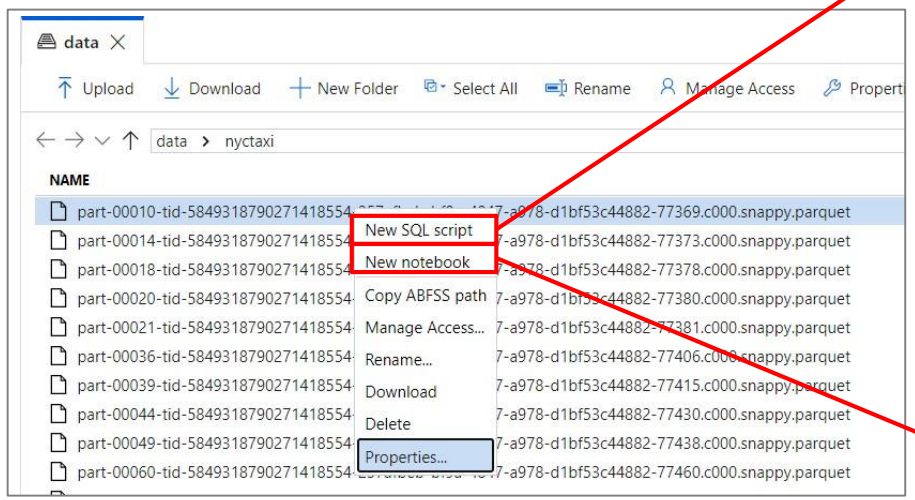


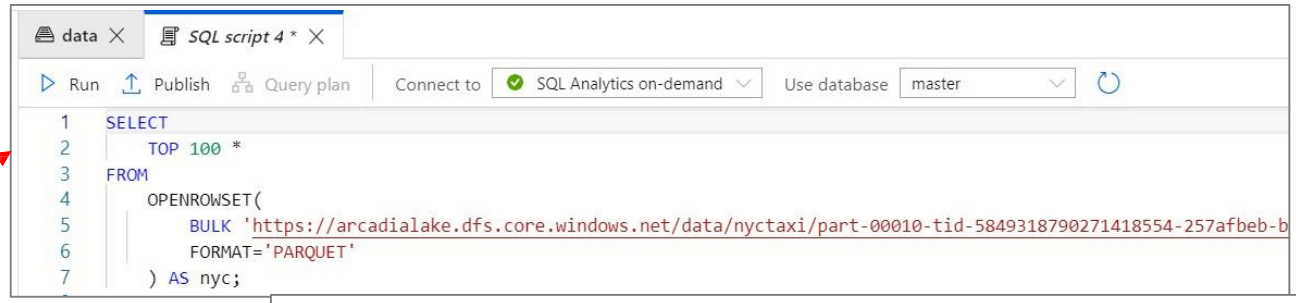


Data Hub –Storage accounts

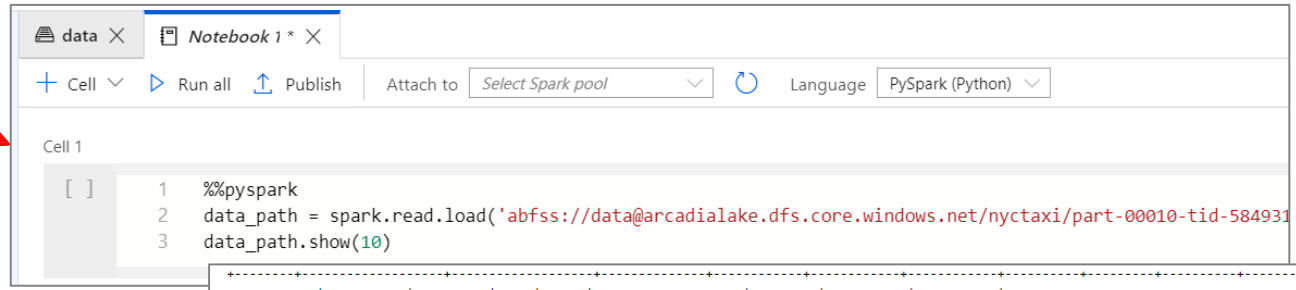
Two simple gestures to start analyzing with SQL scripts or with notebooks.

Auto-generates T-SQL or PySpark





STARTLON	STARTLAT	ENDLON	ENDLAT	RATECODEID	STOREANDFWDFLAG	PAYMENTTYPE	FAREAMOU
-73.95661	40.783125	-73.962751	40.772417	1	N	CSH	5.3
-73.998248	40.724852	-73.991723	40.715288	1	NULL	CRD	6.1
-73.961869	40.811094	-73.9683	40.78687	1	N	CSH	8.1
-74.000057	40.717537	-73.99465	40.698172	1	NULL	CSH	8.1
-73.967002	40.757073	-73.965254	40.766673	1	N	CSH	4.5

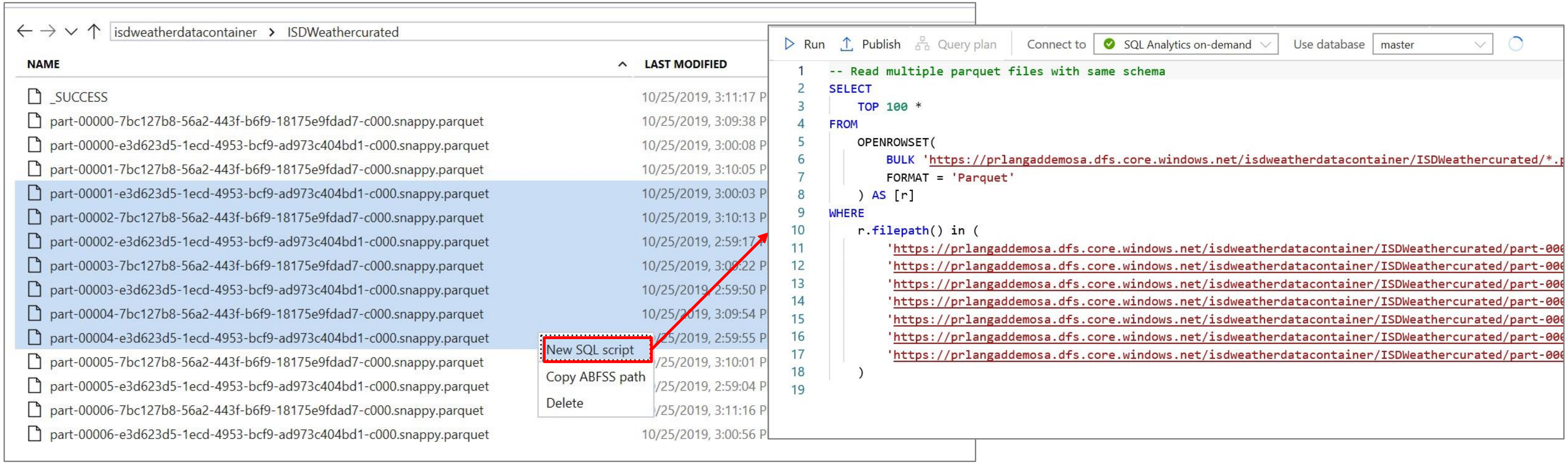



```
+--------+--------------------+--------------------+---------------+------------+------------+------------+----------+----------+----------+----------+-----------+
|vendorID| tpepPickupDateTime|tpepDropoffDateTime|passengerCount|tripDistance|puLocationId|doLocationId|  startLon|  startLat|    endLon|    endLat|rateCodeId|
storeAndFwdFlag|paymentType|fareAmount|extra|mtaTax|improvementSurcharge|tipAmount|tollsAmount|totalAmount|
+--------+--------------------+--------------------+---------------+------------+------------+------------+----------+----------+----------+----------+----------+
|     VTS|2011-04-30 23:57:00|2011-05-01 00:08:00|              3|        1.37|        null|        null| -73.99358|40.735863|-73.974242| 40.73697|         1|
null|         CRD|       7.3|  0.5|   0.5|                null|       1.5|       0.0|       9.8|
|     CMT|2011-05-04 05:29:10|2011-05-04 05:52:56|              1|         8.8|        null|        null|-73.973757| 40.78716|-73.967875| 40.69967|         1|
N|         CRD|      22.5|  0.5|   0.5|                null|       4.7|       0.0|      28.2|
|     VTS|2011-04-30 23:53:00|2011-05-01 00:17:00|              1|        5.45|        null|        null|-73.952017|40.778407|-74.005777|40.726028|         1|
null|         CSH|      16.9|  0.5|   0.5|                null|       0.0|       0.0|      17.9|
|     CMT|2011-05-04 05:38:52|2011-05-04 05:43:09|              1|         1.1|        null|        null|-73.993478|40.741549|-73.981673|40.750561|         1|
N|         CRD|       5.3|  0.5|   0.5|                null|       1.0|       0.0|       7.3|
|     VTS|2011-04-30 23:55:00|2011-05-01 00:09:00|              1|        6.88|        null|        null|-73.782203|40.644765|-73.718137|40.636435|         4|
null|         CSH|      26.1|  0.5|   0.5|                null|       0.0|       0.0|      27.1|
```

Data Hub – Storage accounts

SQL Script from Multiple files

Multi-select files generates a SQL script that analyzes all the files together



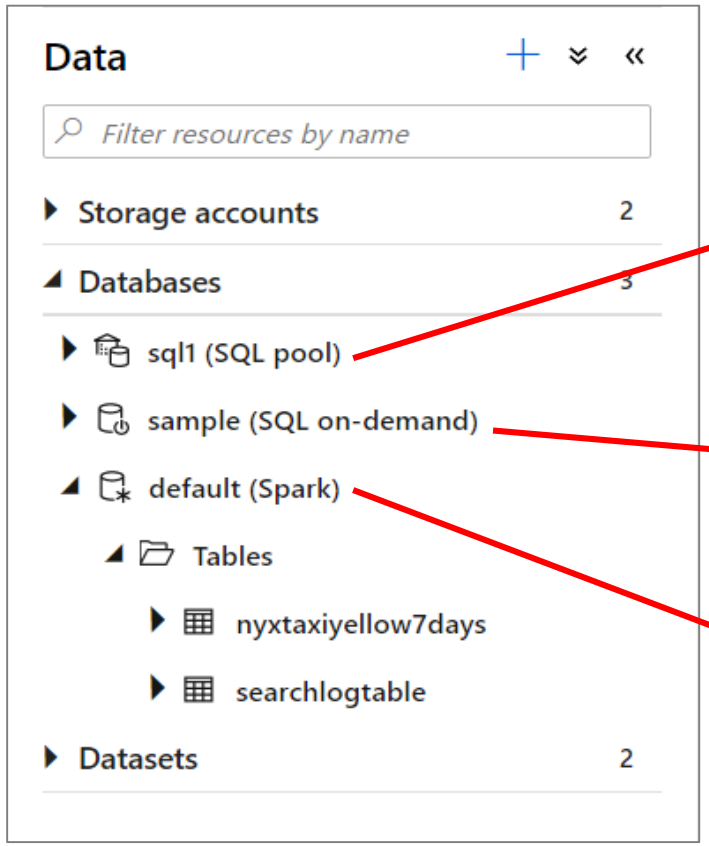
Data Hub –Databases

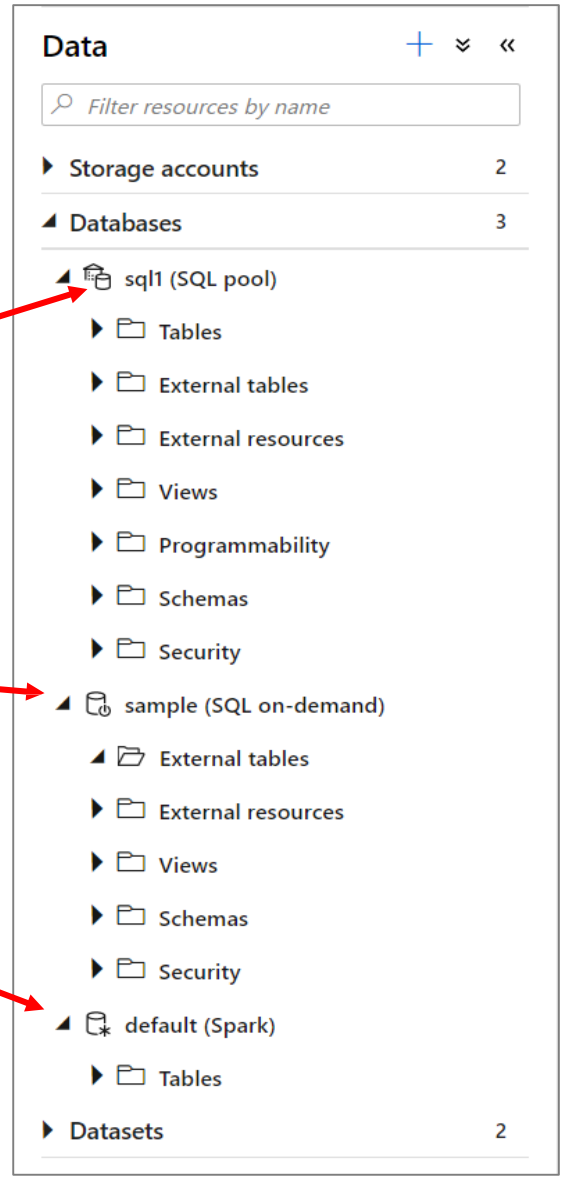
Explore the different kinds of databases that exist in a workspace.

SQL pool

SQL on-demand

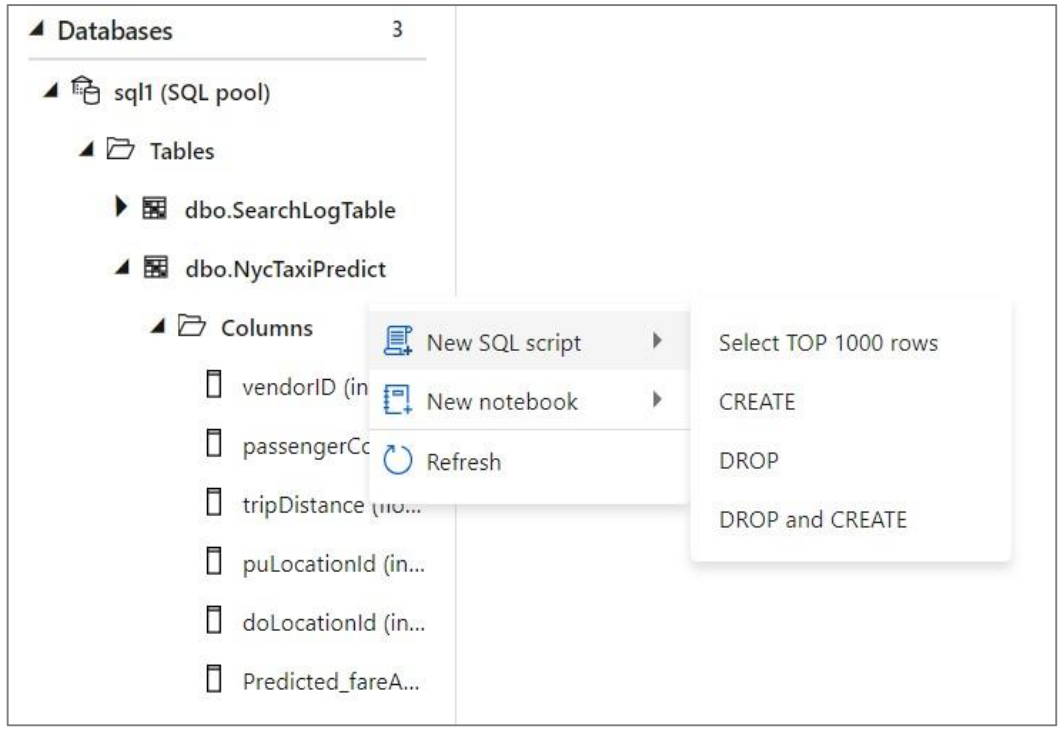
Spark



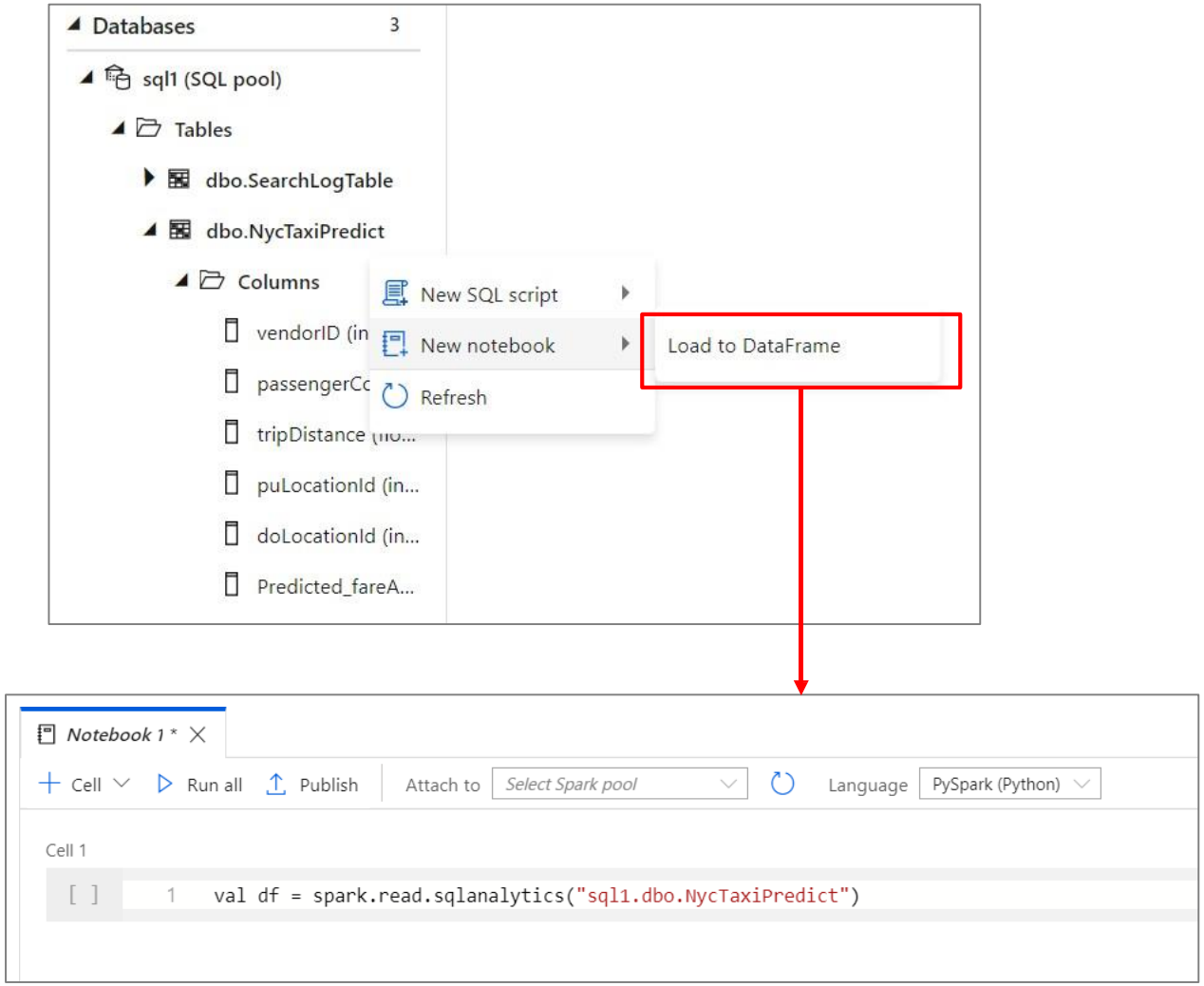


Data Hub – Databases

Familiar gesture to generate T-SQL scripts from SQL metadata objects such as tables.



Starting from a table, auto-generate a single line of PySpark code that makes it easy to load a SQL table into a Spark dataframe



Demonstration

SQL Serverless

Data Lakehouse

What is a Data Lake

Cheap data storage location

Able to handle any kind of data

Integrates with many services and sources

Scalability at a low cost

Schema on Read

Beware the Data Swamp as data has no enforced organization

What is a Data Lakehouse?

- Using a data lake as a data warehouse
- Query flat file data as if it was in a warehouse
- Decreased cost over Data Warehouse
- Ideal for Power BI
- Needs to be part of a well organized Data alkae

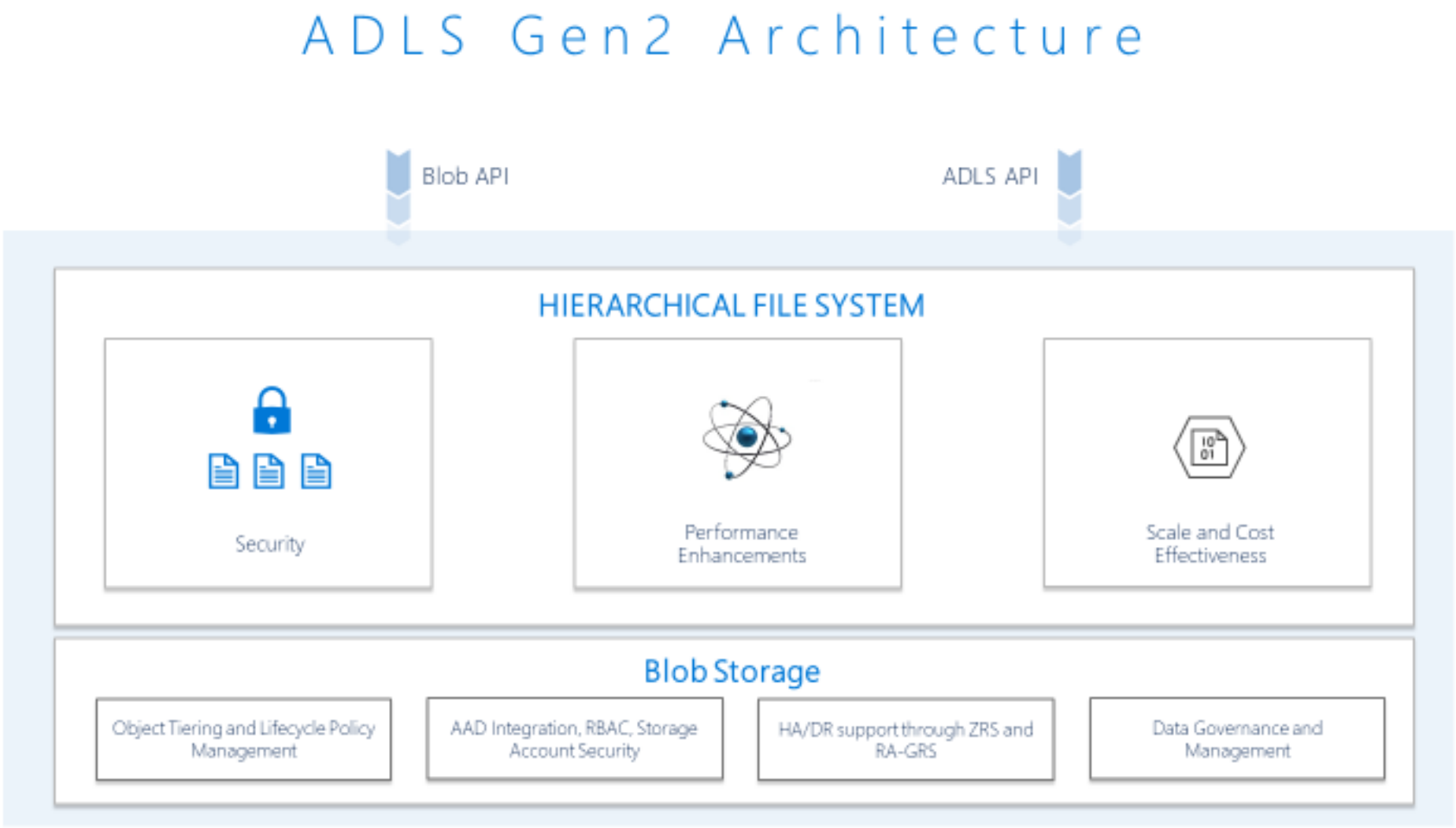
Data Lakehouse Considerations

- Slower than a data warehouse
- Needs to be well organized
- Does the query speed matter?

Azure Data Lake Gen 2

- Improved Integration
- Hierarchy
- Performance
- Improvements
- Flexible Pricing

Underlying Architecture



Structuring ADLS Gen2

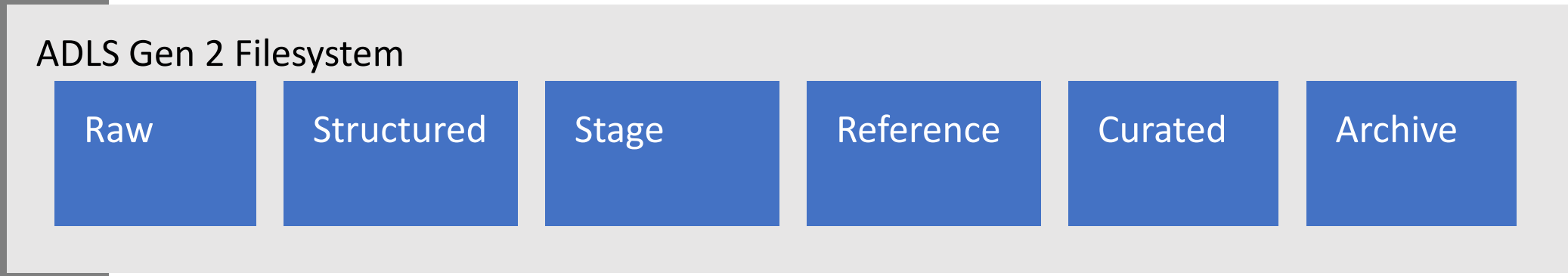
Separate storage accounts for each environment: dev, test, & production.

Use a common folder structure to organize data by degree of refinement.

Zone Development

Greater Flexiblity

Use a common folder structure to organize data by degree of refinement.



Optional Organizational Subareas

Subsystem

Data Sources

Data Feeds

Partitions

Used for data received incrementally Big Data

Users

Sand box for development

Data Sources

Batch

Microbatch

Stream

Data Lake Elements

Structured

Tabular Data

CSV

Txt

XML

JSON

Unstructured

Images

Sound

PDF

DOC

Data Lake Security

Use Azure Role Assignments

Limit the number of Storage Blob Data Owners

Use AD groups to assign permissions based on folders

Provide access to service principals

Storage Partitions

\SubjectArea\DataSource\YYYY\MM\DD\FileData_YYYY_MM_DD.csv

Optimum for folder security

Commonly used in Raw Zones

May also be in Curated or Staged Zones

File Formats

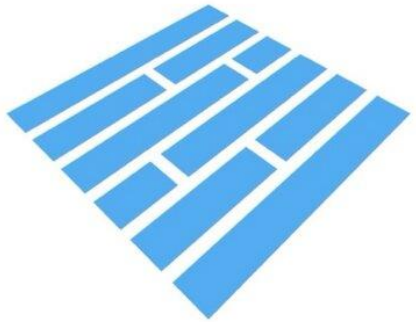
Common storage formats

CSV

Parquet

Delta Lake

Parquet Files



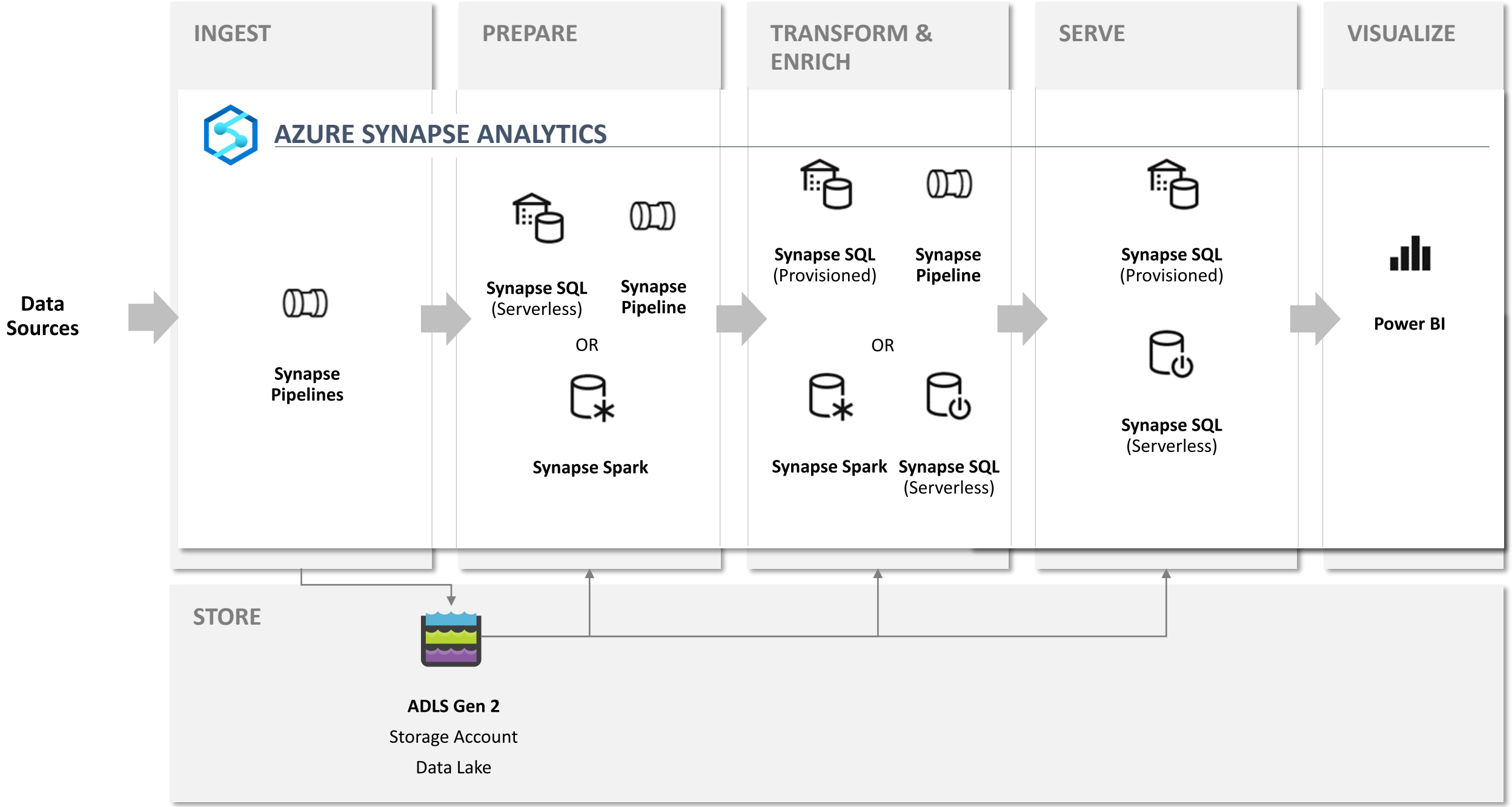
Optimally designed for Spark Processing

Part of Apache Hadoop Collective

Supports nested data structures

Columnar Storage compression options

Compression can be huge resulting in decreased query time and cost to store compared to CSV


INGEST
PREPARE
TRANSFORM & ENRICH
SERVE
VISUALIZE
AZURE SYNAPSE ANALYTICS
Data Sources
Synapse Pipelines
Synapse SQL (Serverless)
Synapse Pipeline
OR
Synapse Spark
Synapse SQL (Provisioned)
Synapse Pipeline
OR
Synapse Spark
Synapse SQL (Serverless)
Synapse SQL (Provisioned)
Synapse SQL (Serverless)
Power BI
STORE
ADLS Gen 2
Storage Account
Data Lake

Integrating Synapse with Power BI

Customers are searching for ways to level-up their data

Dashboards

Ad-hoc Reporting

Advanced Visualization

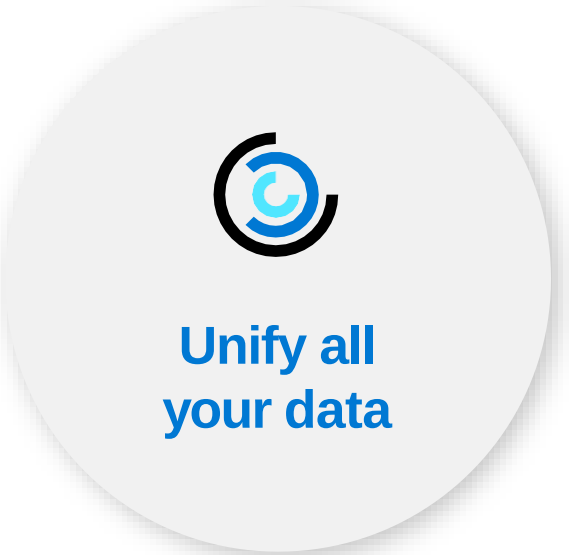
However, their data remains…

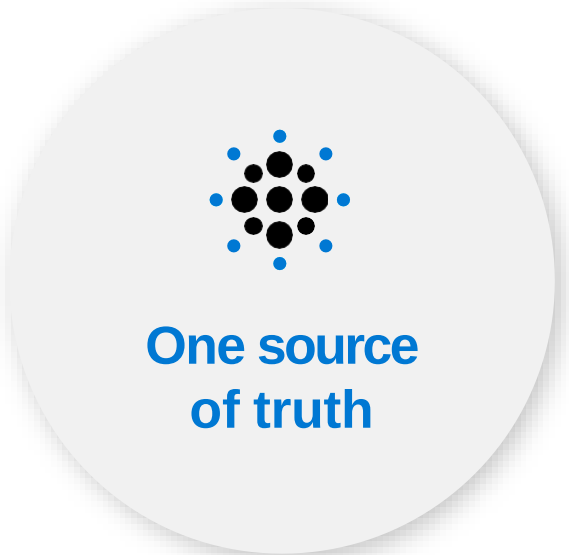
Siloed

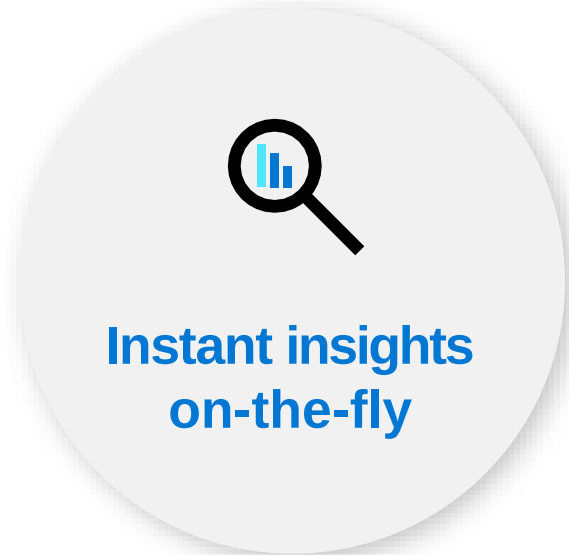
Under-utilized

Slow to provide insights

Combining BI tools and data warehouses expands analytics capabilities

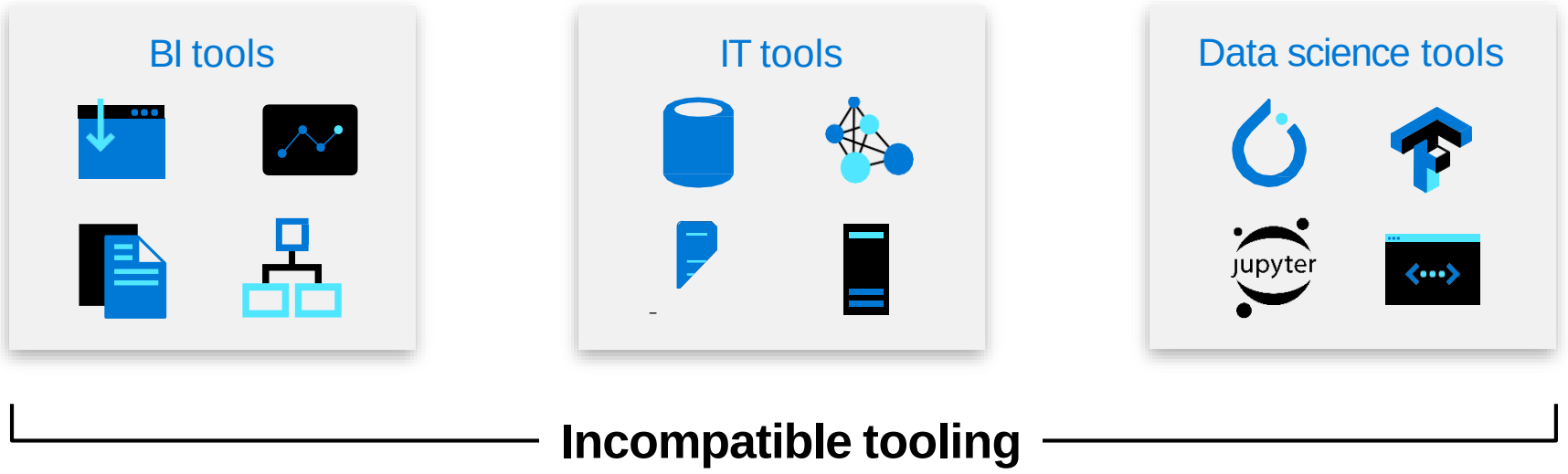






BI tools + Azure Synapse Analytics

Data silos and incompatible tooling inhibit collaboration



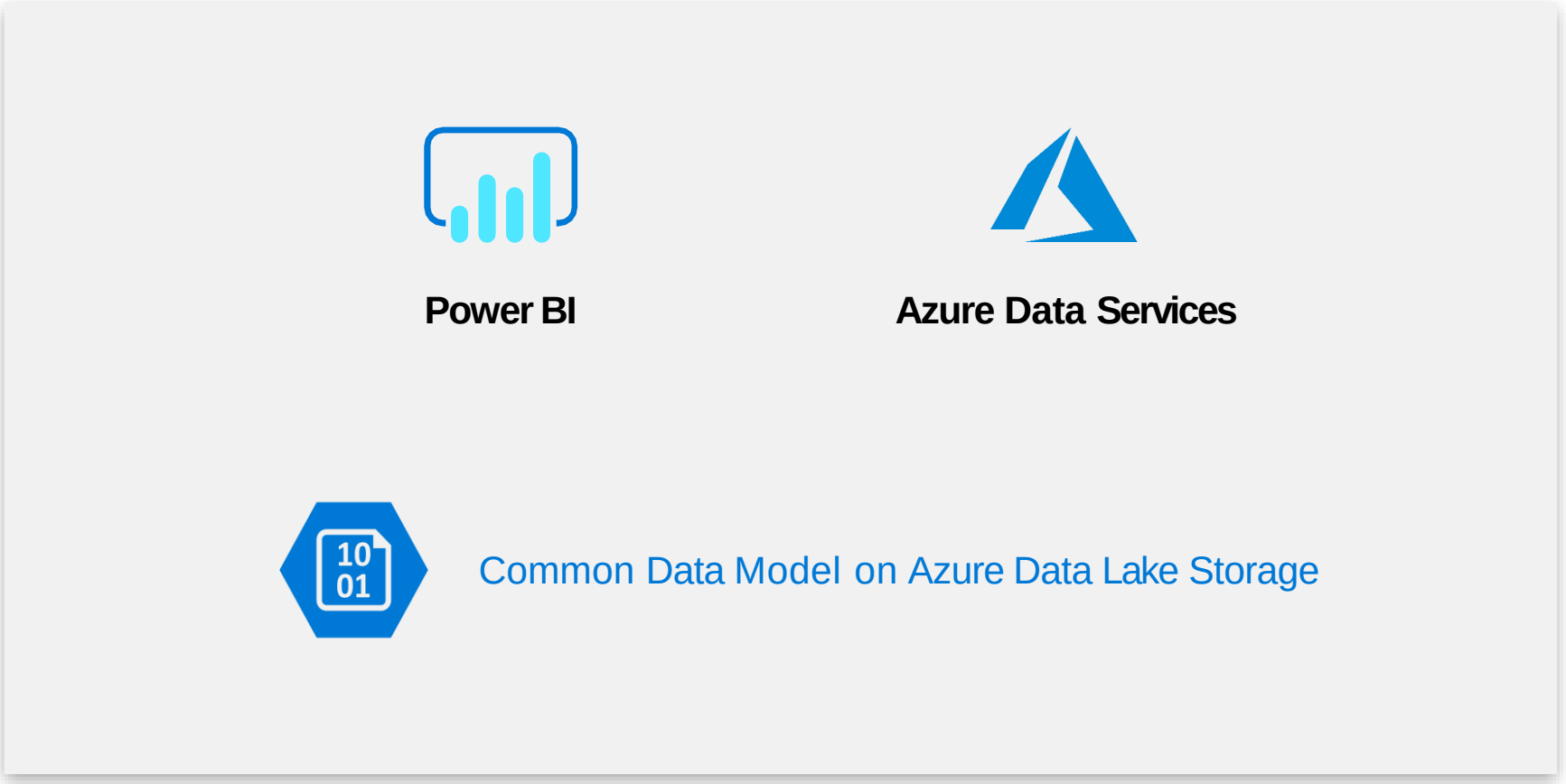
Unify your organization

Power BI

Azure Data Services

Common Data Model on Azure Data Lake Gen 2 Storage

Unify your organization



Unleash data value with Power BI and Azure Data Services

Business value

Unmatched capabilities

Powerful visualization and reporting

Petabyte-scale analytics

Advanced analytics and AI

Frictionless collaboration

Business analysts ⟷ IT professionals ⟷ Data scientists

Powerful and integrated tooling

Power BI

Azure Data Services

Unified data

Common Data Model on Azure Data Lake Storage

Maximize the value of your data

Powerful visualization and reporting

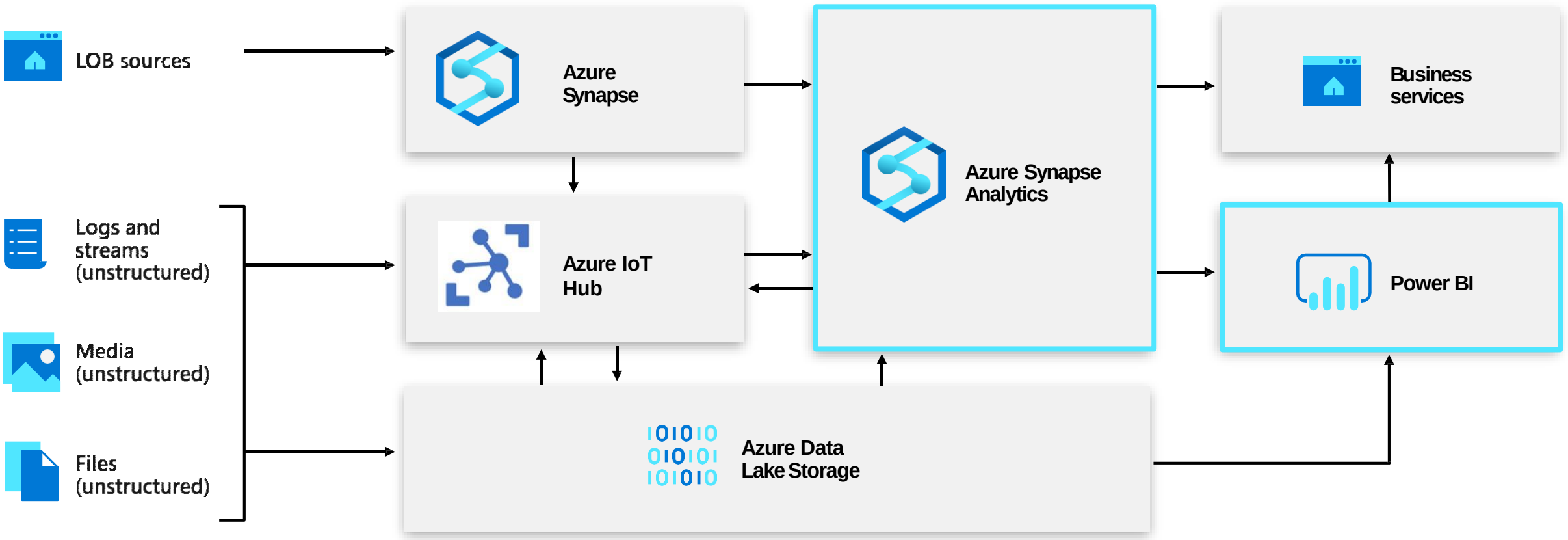
+

Petabyte-scale analytics

+

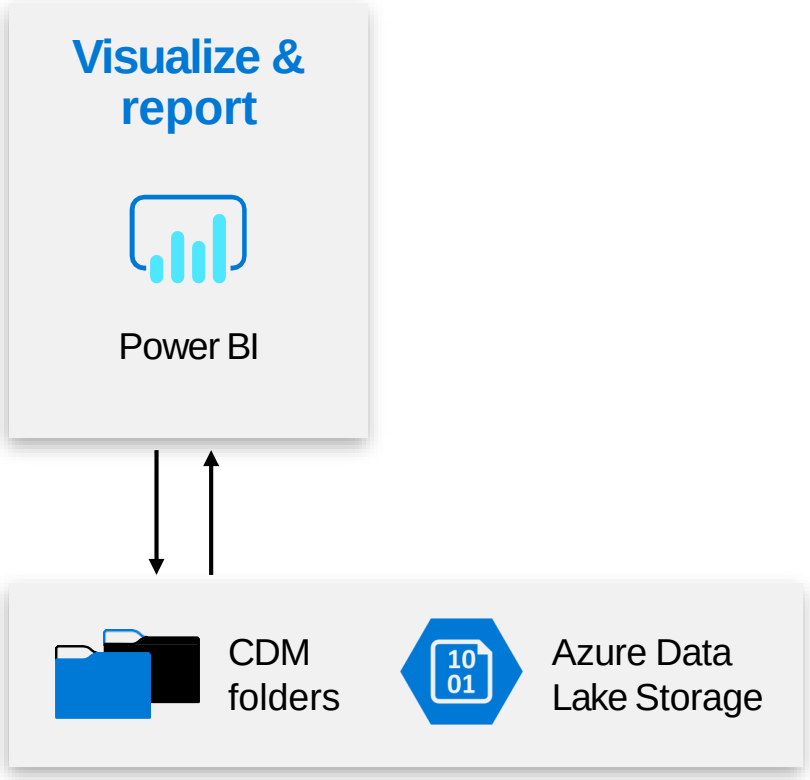
Advanced analytics and AI

The data warehouse + Power BI in the data-driven business



Leverage powerful visualization and reporting

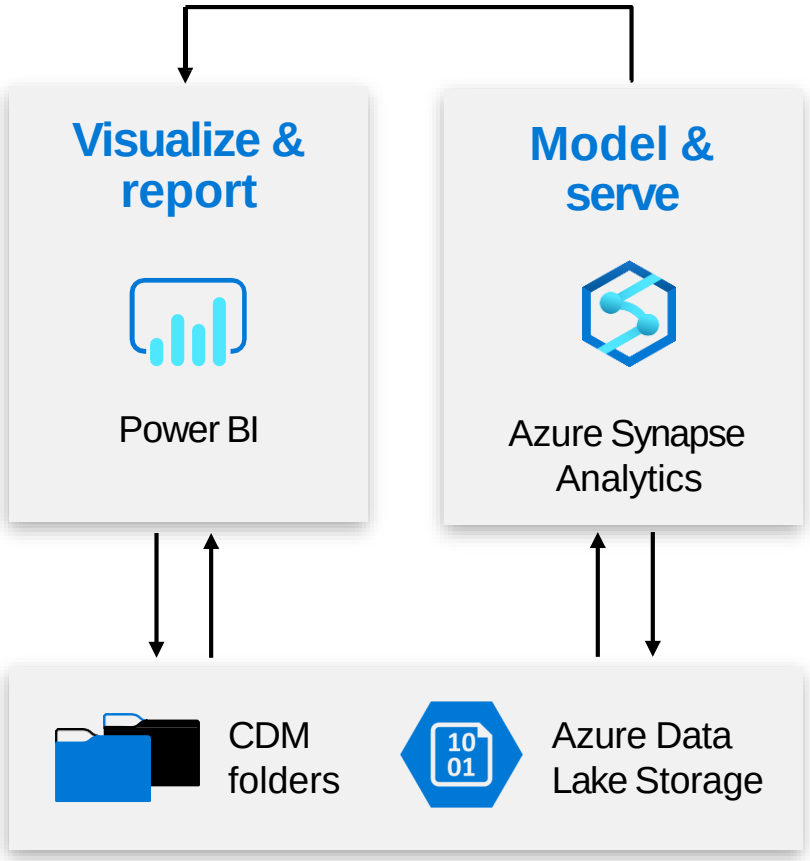
Power BI



- **Scale and govern efficiently**
- **Semantic modeling**
- **Improve management**
- **Open-platform connectivity**

Deliver petabyte-scale analytics

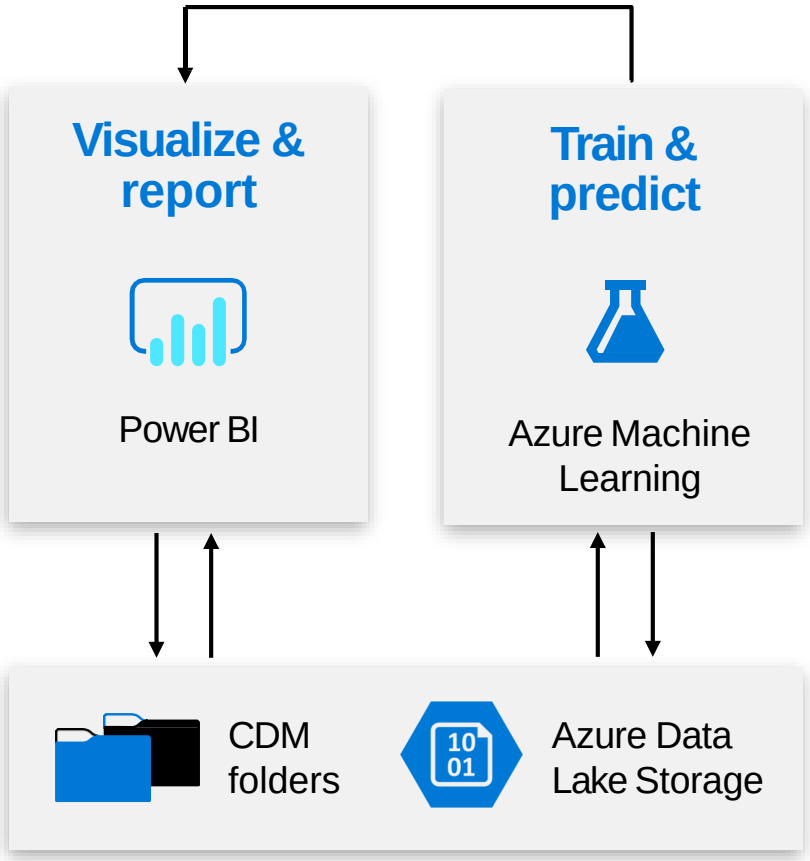
Power BI + Azure SQL Data Warehouse



- **Power BI Aggregations**
- **DirectQuery**
- **14x faster and 94% cheaper**
- **One-click connection**

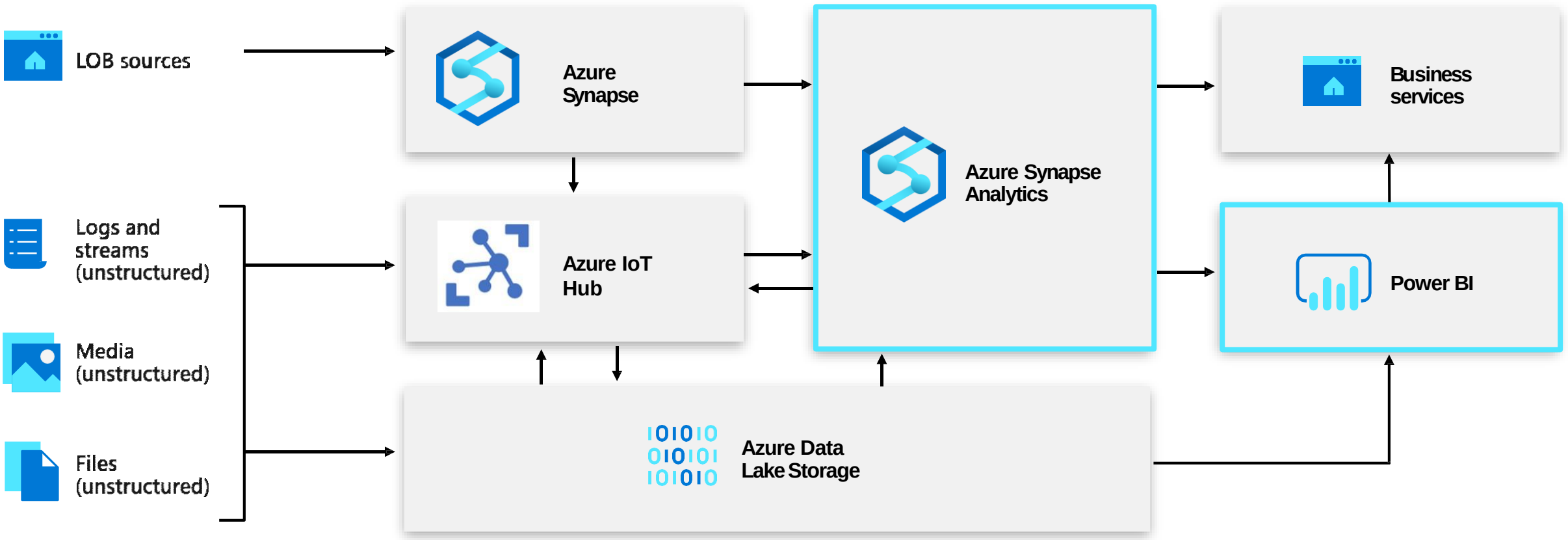
Gain deep insights with advanced analytics and AI

Power BI + Azure Machine Learning



- **Solve complex challenges**
- **Reduce time-to-insight**
- **Start with no-code ML**
- **Extend with Azure Machine Learning**

The data warehouse + Power BI in the data-driven business



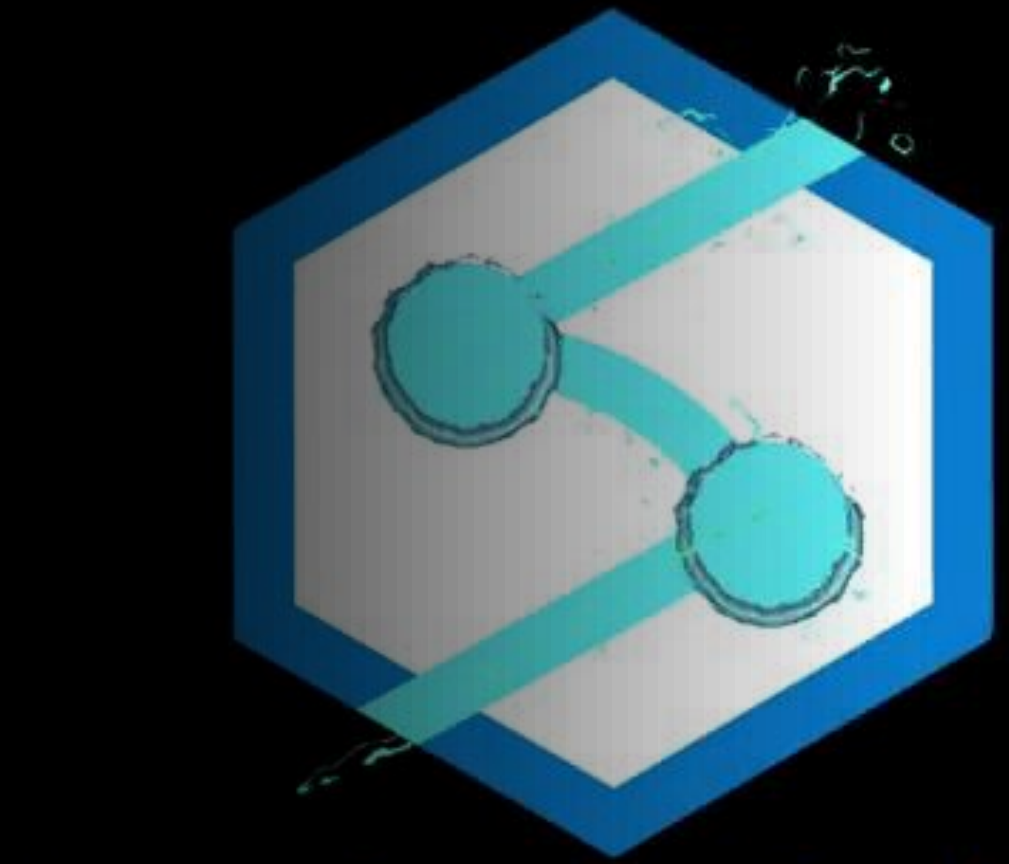
Demonstration

Power BI

Quiz!

Poll Question

https://geni.us/SynapsePrecon3



Azure Synapse Analytics

Integration

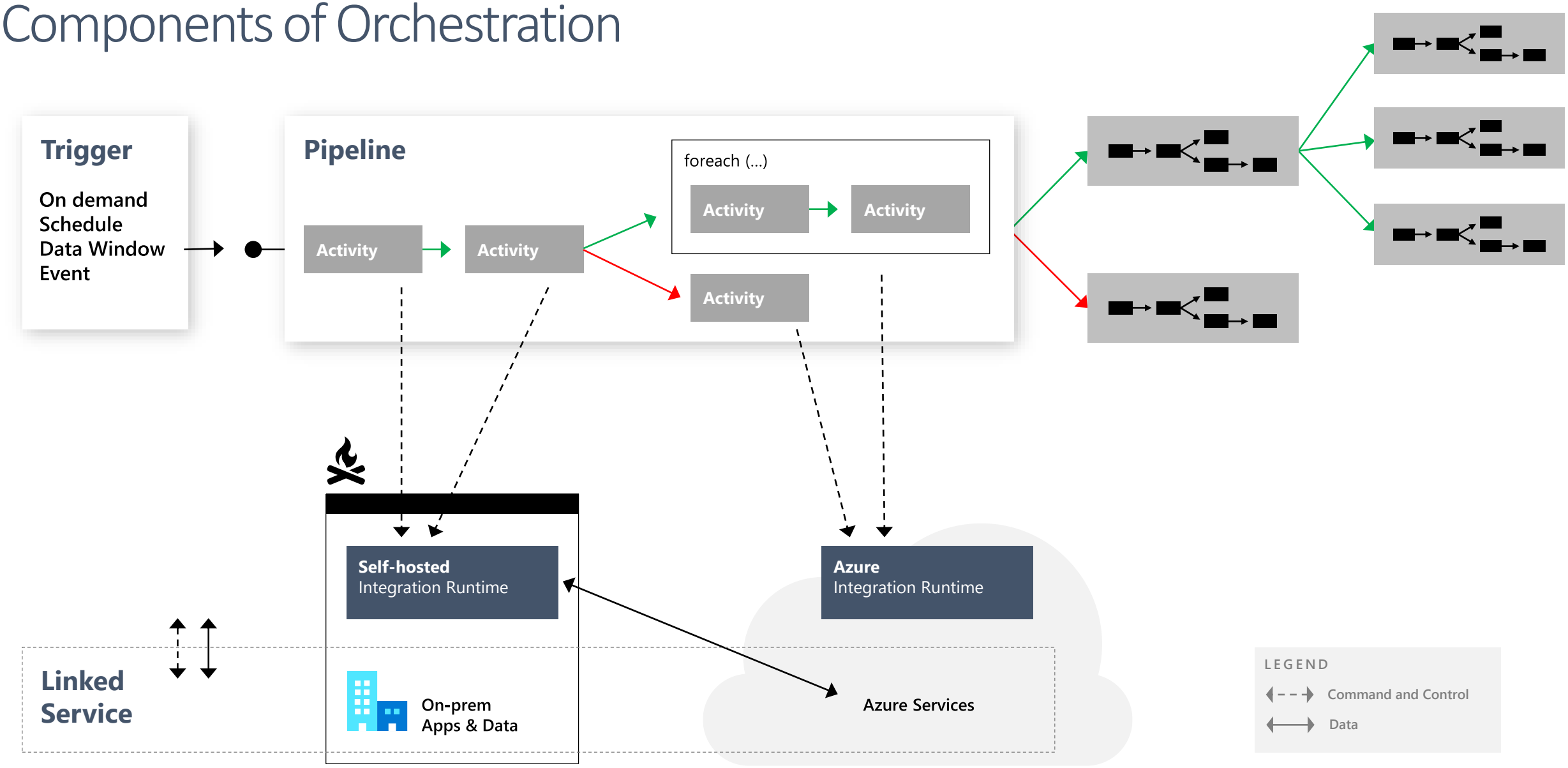
Migrating data

Can choose to migrate it all in Spark

Use predesigned components from Azure Data Factory

Same code base and cost

Components of Orchestration



Fastest is done by batch:	Extract from data source to multiple CSV/Parquet files
	Use AzCopy to upload to ADLS
Alternative is query-insert:	Set up SSIS self-hosted integration runtime on-premises
	Use Synapse Pipeline to extract/copy
	Use Synapse Pipeline to execute load procedure
Large Migrations:	Use Azure Data Box where available

Options:

- Extract using Synapse Pipelines
- Write to ADLS as Parquet files
- AzCopy is a fast move for files from S3 to ADLS

File format challenges

Invalid file format

- Multiple row types
- Ragged columns
- Row size > 1Mb
- Datetime format/s (e.g., use of nanosecond date time)
- NULL value literal/s
- Free form text
- Parquet partitions
- XML data
- Use of non-standard line delimiters (e.g., CR)

Solutions

- Use Spark to pre-process and fix data errors
- Flatten and parse XML in Spark
- Use COPY to ingest complex CSV instead of Polybase

Ingest and Store – Formats

For batch flat files, Azure Synapse Analytics supports
CSV, Parquet, ORC, and JSON formats.

Ingest streaming data messages/events via Event Hub or IoT Hub.

Parquet format recommended for storing ingested data at various levels of refinement.

Ingest - When to BCP / Bulk Copy

Green fields: Never

- Network unreliability, no retries
- Needs VM in cloud, performance dependent on VM configuration
- Doesn’t support ADLS
- Reduces concurrency
- Control-gated performance limitation, can not scale with DWU

Migrations:

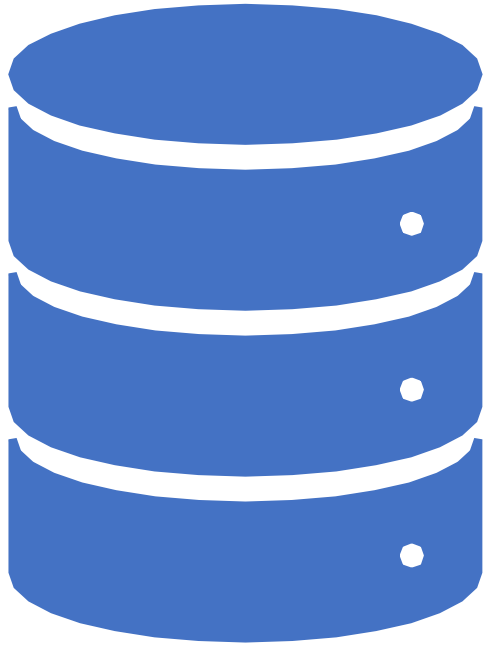
- Use Synapse Pipeline or AzCopy
- Bulk Copy will work, but it will be slower than other methods

Ingest – Synapse Pipelines

Un-check USE TYPE DEFAULT, it is not a best practice.

Land data in ADLS Gen2, then ingest using Polybase / COPY.

Ingest Flat files to tables



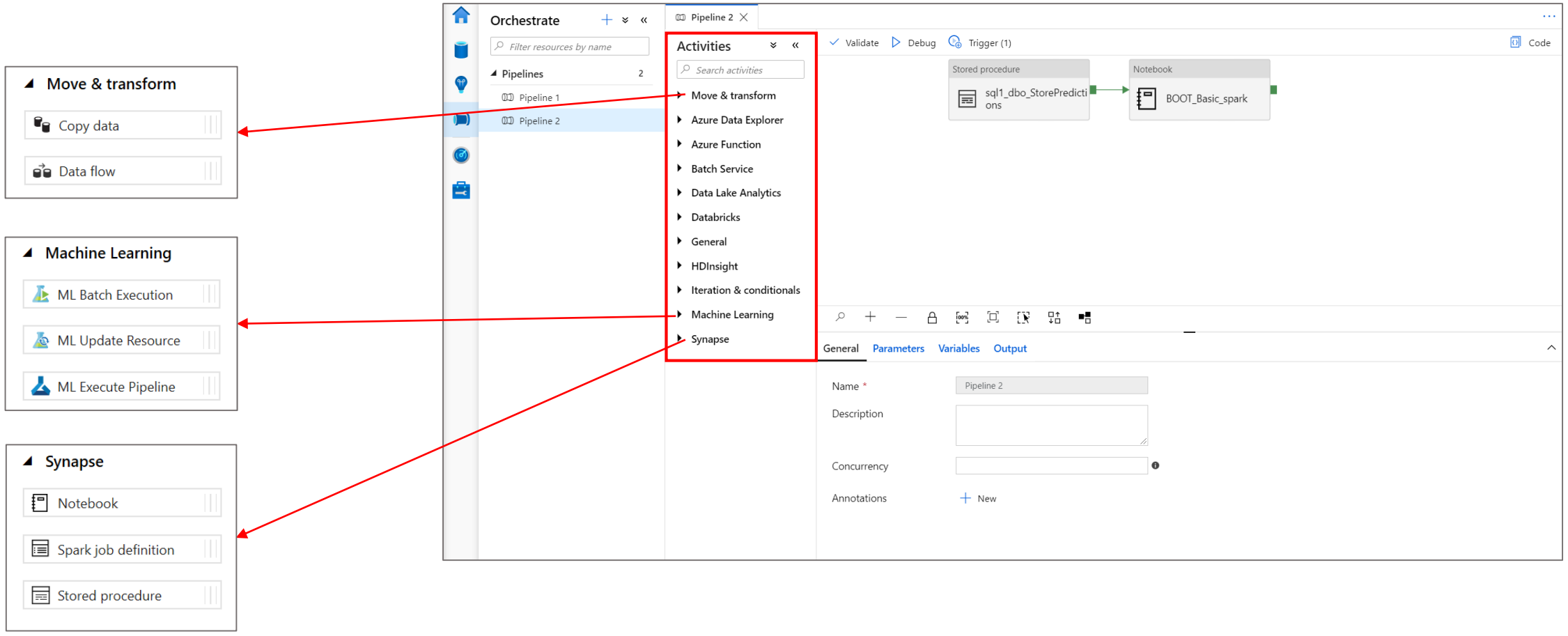
- Ingest flat file data into Azure Storage (Azure Data Lake Store Gen2)
  - When your data sources are on-premises, you need to move the data to Azure Storage before ingestion.
  - Data in other cloud platforms needs to be moved to Azure Storage before ingestion.

- Load from flat files as relational tables within the data warehouse

Pipelines

Create pipelines to ingest, transform and load data with 90+ inbuilt connectors.

Offers a wide range of activities that a pipeline can perform.



Pipelines

Overview

Provide ability to load data from storage account to desired linked service.

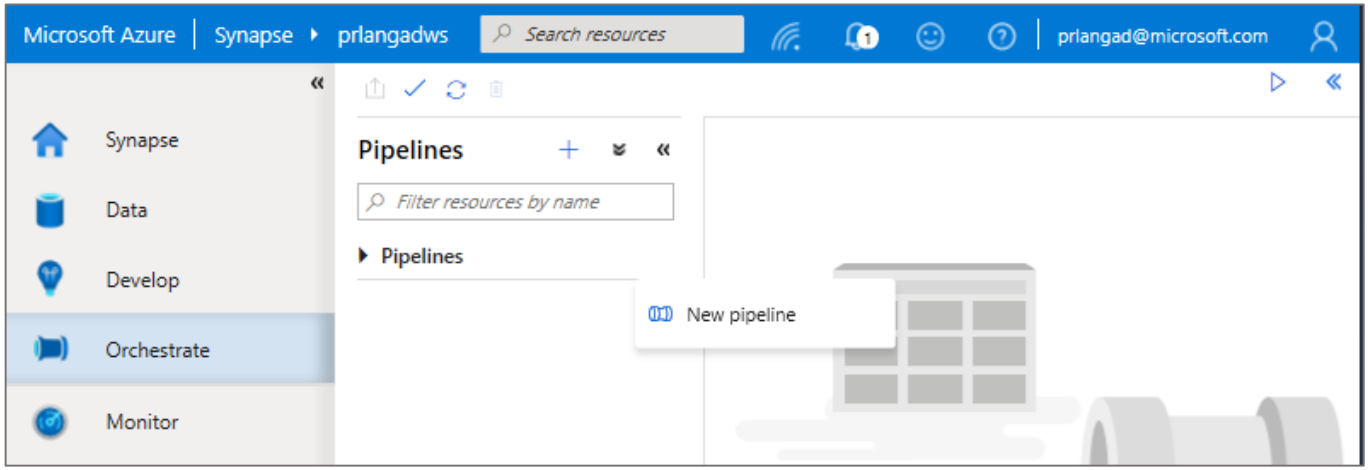
Load data by manual execution of pipeline or by orchestration.

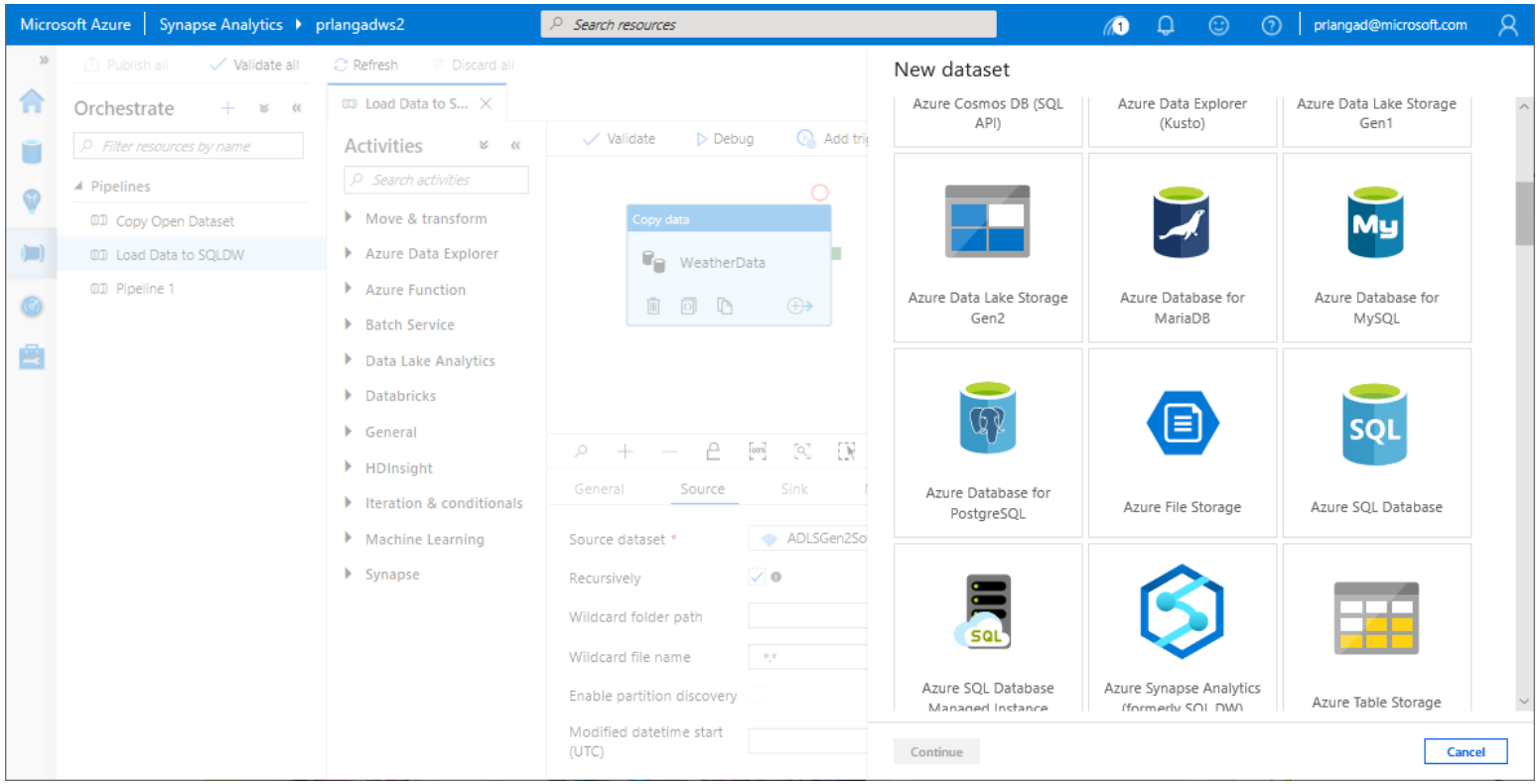
Benefits

Supports common loading patterns.

Fully parallel loading into data lake or SQL tables.

Graphical development experience.





Integration runtimes

Overview

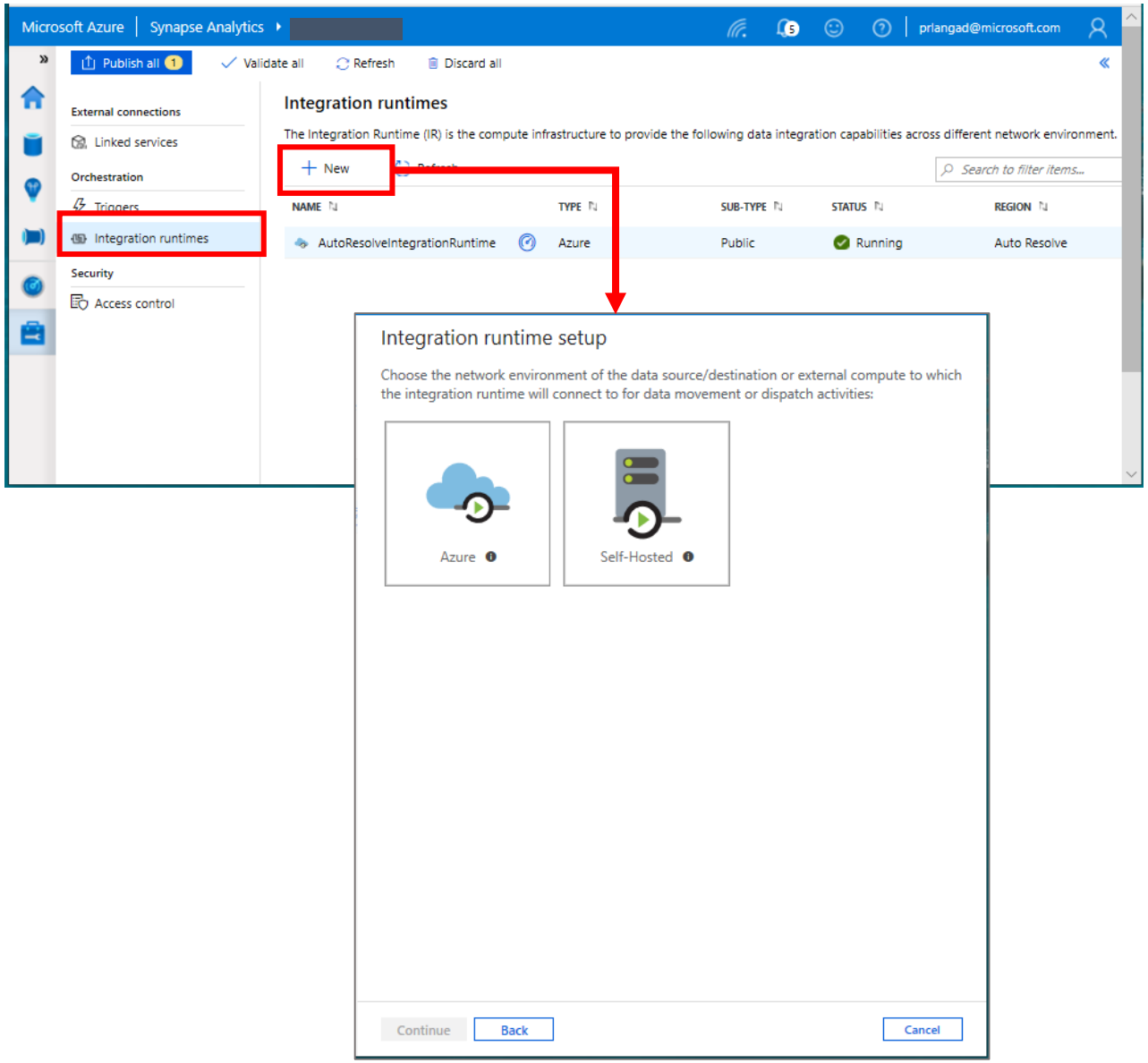
Integration runtimes are the compute infrastructure used by Pipelines to provide the data integration capabilities across different network environments. An integration runtime provides the bridge between the activity and linked services.

Benefits

Offers Azure Integration Runtime or Self-Hosted Integration Runtime

Azure Integration Runtime – provides fully managed, serverless compute in Azure

Self-Hosted Integration Runtime – use compute resources in on-premises machine or a VM inside private network



Linked services

Overview

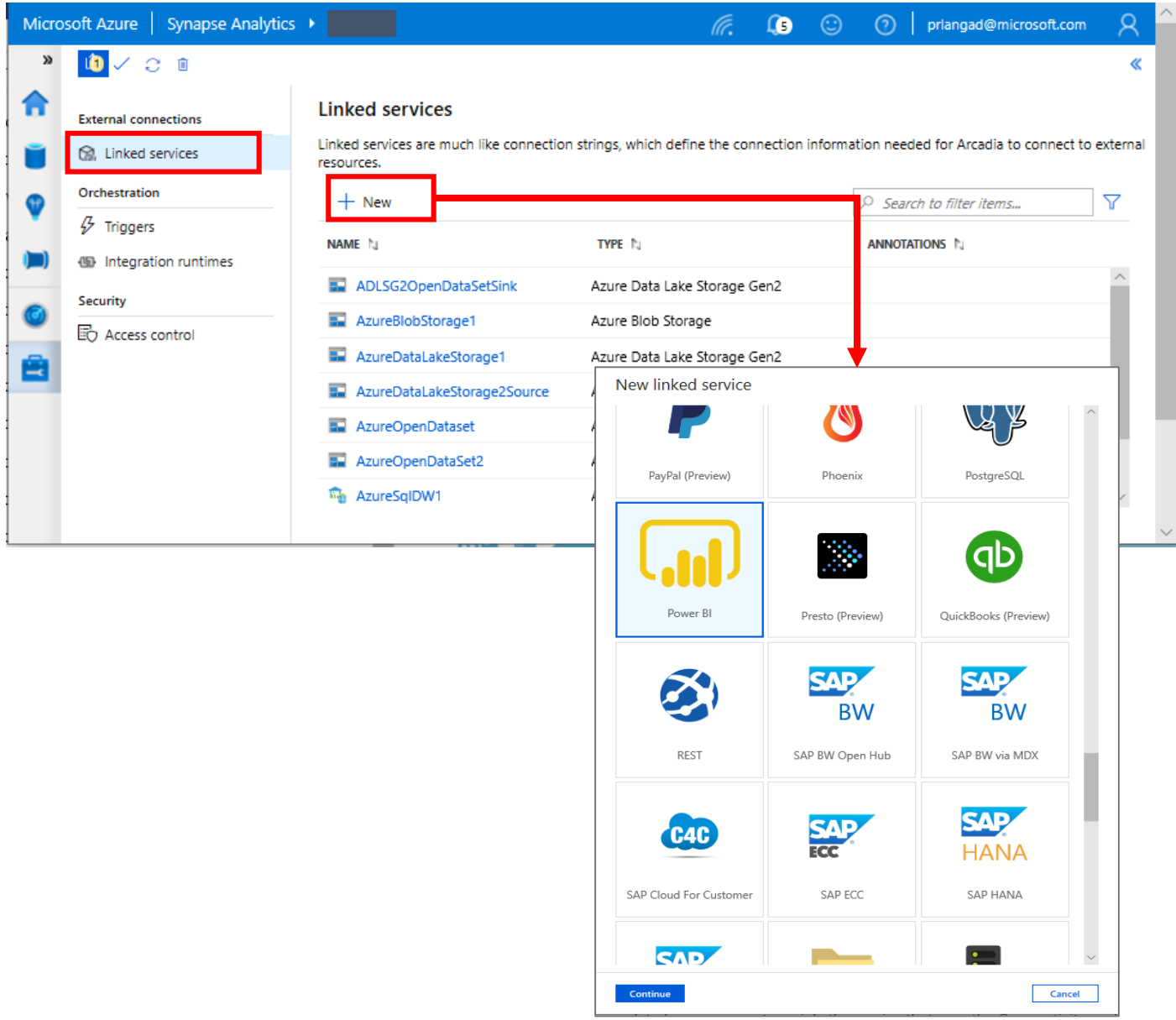
Linked services define the connection information needed to connect to external resources.

Benefits

Offers pre-build 90+ connectors

Easy cross platform data migration

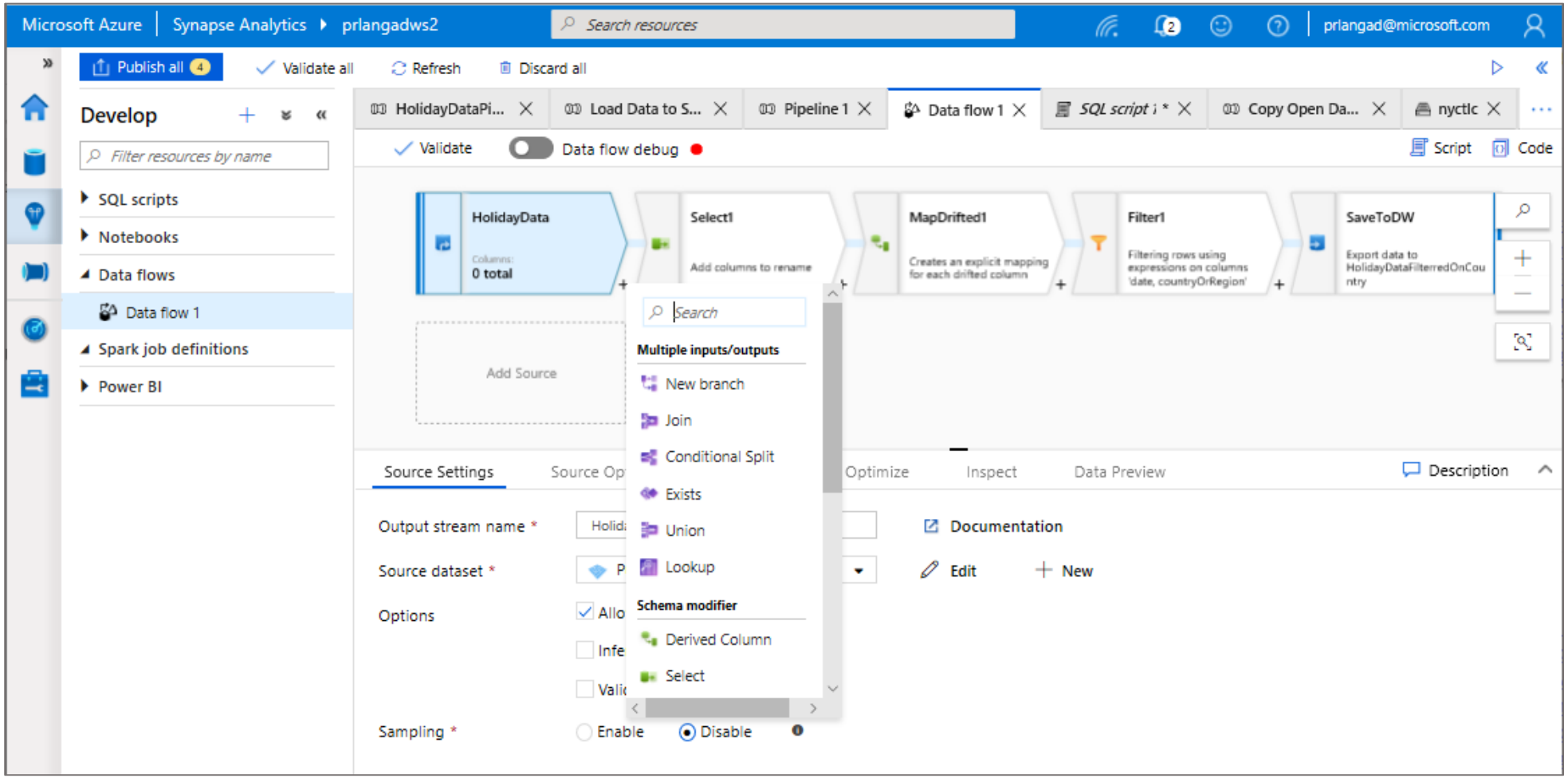
Represents data store *or* compute resources



Data Flows

Data flows are pipeline activities providing a visual way of specifying how to transform data.

Provides a code-free experience.



Dataflow Capabilities

Handle upserts, updates, deletes on sql sinks

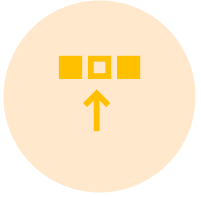
Add new partition methods

Add schema drift support

Add file handling (move files after read, write files to file names described in rows etc)

New inventory of functions (for e.g Hash functions for row comparison)

Commonly used ETL patterns(Sequence generator/Lookup transformation/SCD…)



Data lineage – Capturing sink column lineage & impact analysis(invaluable if this is for enterprise deployment)

Implement commonly used ETL patterns as templates(SCD Type1, Type2, Data Vault)

Triggers

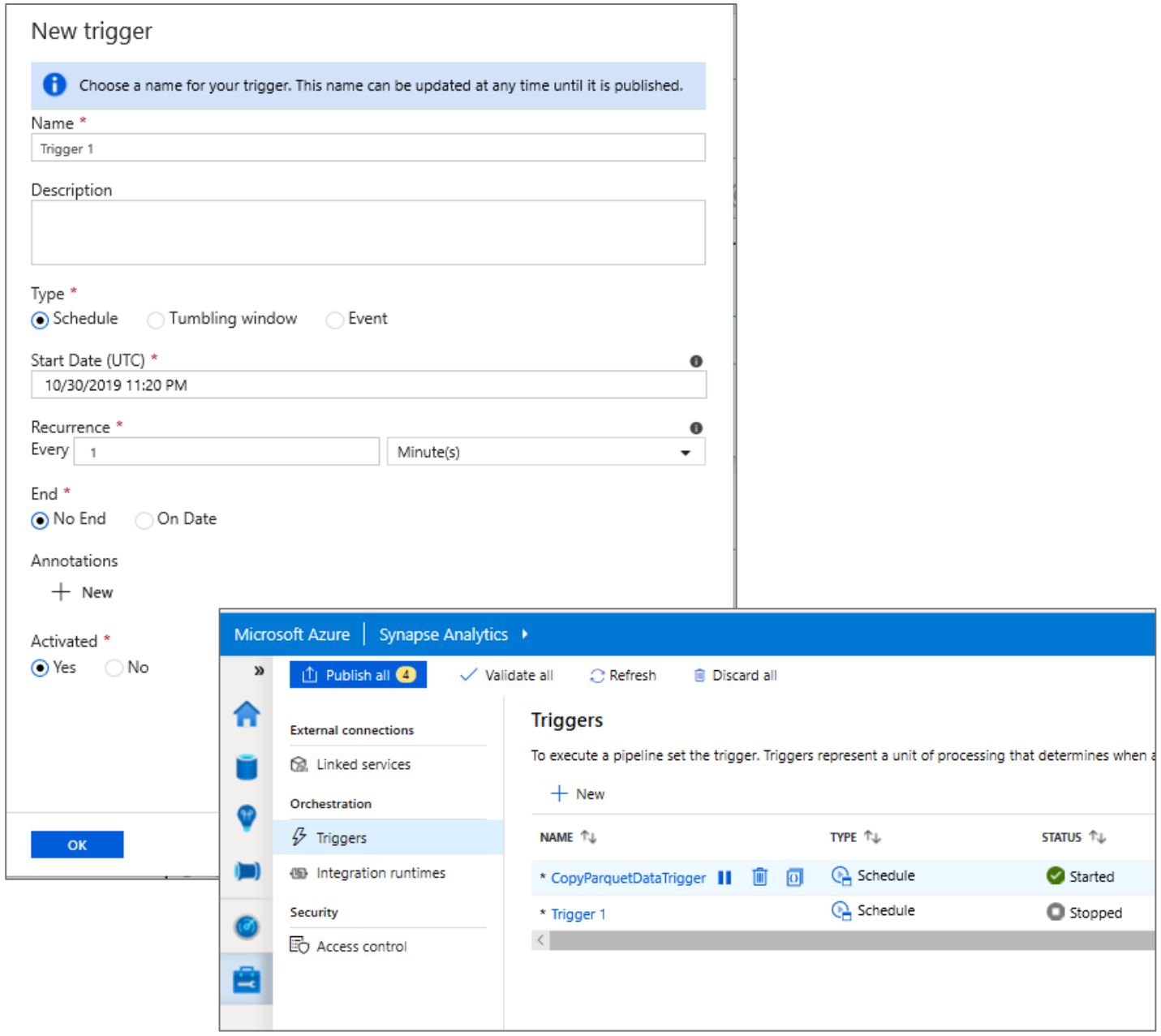
Overview

Triggers represent a unit of processing that determines when a pipeline execution needs to be kicked off.

Data Integration offers 3 trigger types as –

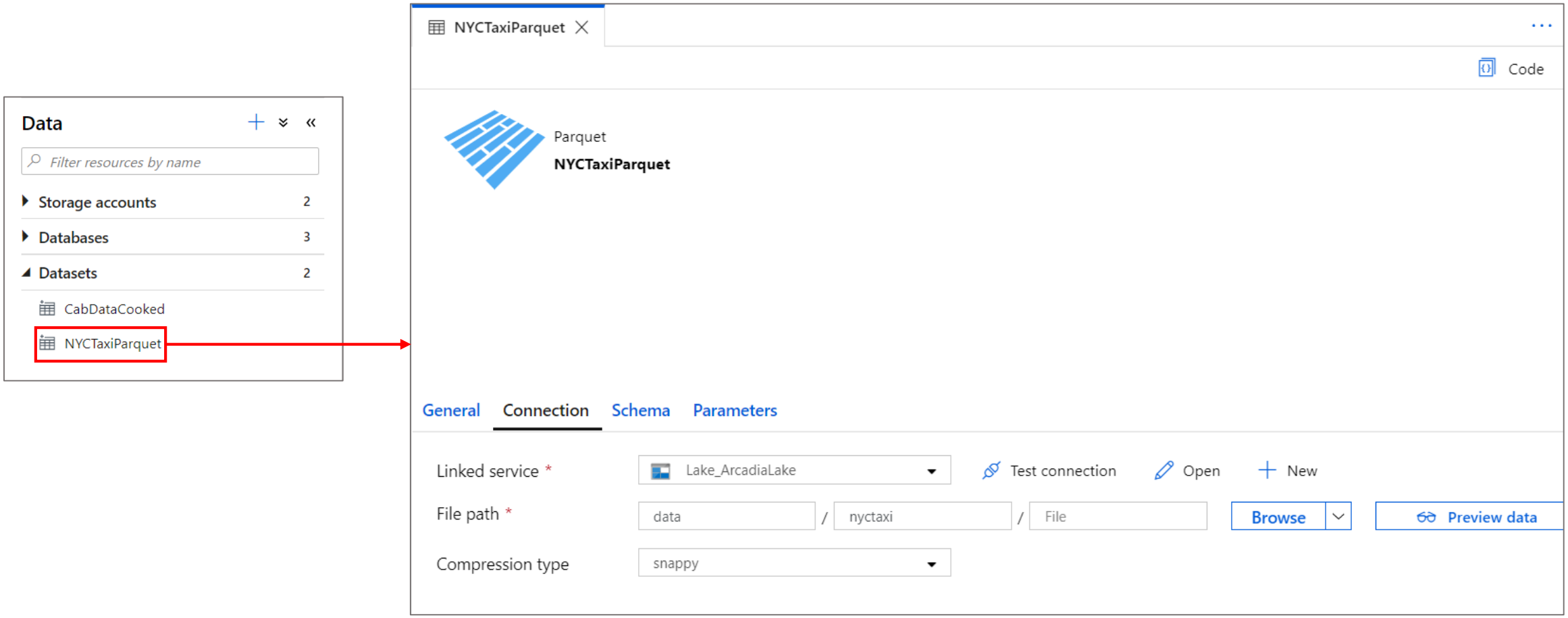
1. Schedule – gets fired at a schedule with information of start date, recurrence, end date
2. Event – gets fired on specified event
3. Tumbling window – gets fired at a periodic time interval from a specified start date, while retaining state

It also provides ability to monitor pipeline runs and control trigger execution.



Datasets

Orchestration datasets describe data that is persisted. Once a dataset is defined, it can be used in pipelines and sources of data or as sinks of data.



Data Movement

Scalable

per job elasticity

Up to 4 GB/s

Simple

Visually author or via code (Python, .Net, etc.)

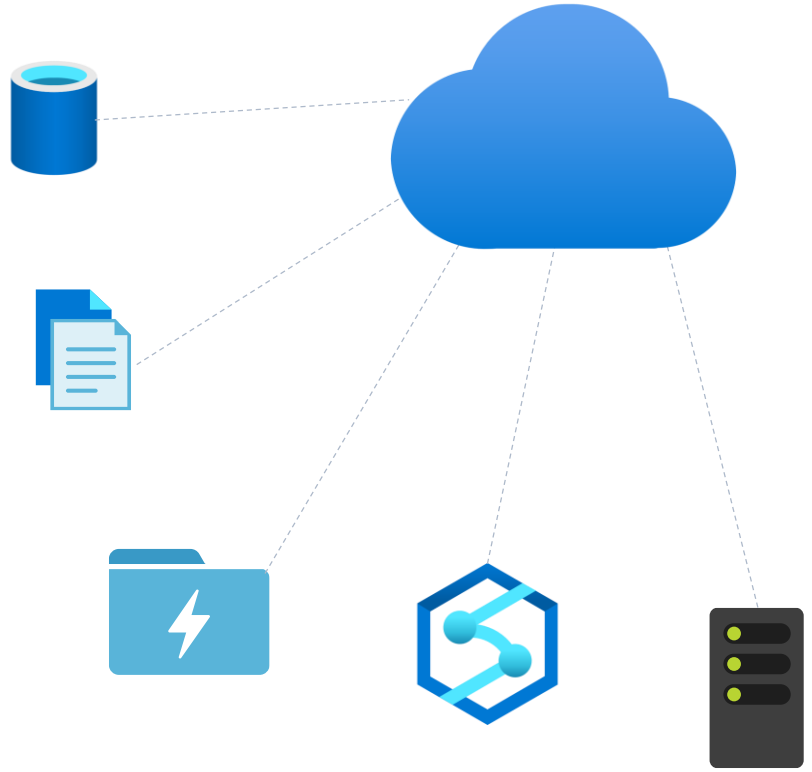
Serverless, no infrastructure to manage

Access all your data

90+ connectors provided and growing (cloud, on premises, SaaS)

Data Movement as a Service: 25 points of presence worldwide

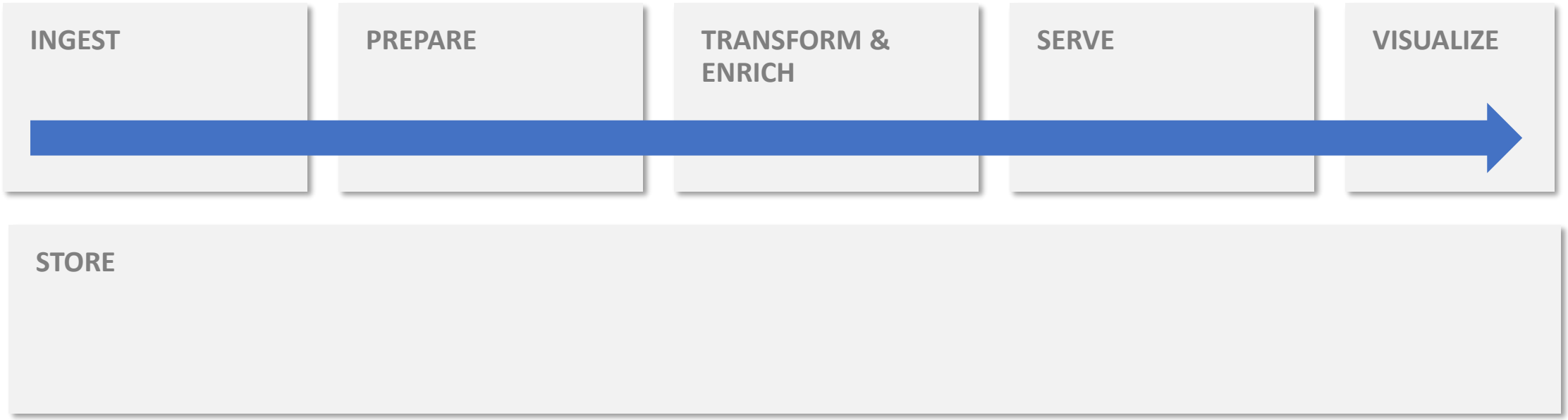
Self-hostable Integration Runtime for hybrid movement



90+ Connectors out of the box

Azure (15)	Database & DW (26)		File Storage (6)	File Formats(6)	NoSQL (3)	Services and App (28)		Generic (4)
Blob storage	Amazon Redshift	Oracle	Amazon S3	AVRO	Cassandra	Amazon MWS	Oracle Service Cloud	Generic HTTP
Cosmos DB - SQL API	DB2	Phoenix	File system	Binary	Couchbase	CDS for Apps	PayPal	Generic OData
Cosmos DB - MongoDB API	Drill	PostgreSQL	FTP	Delimited Text	MongoDB	Concur	QuickBooks	Generic ODBC
Data Explorer	Google BigQuery	Presto	Google Cloud Storage	JSON		Dynamics 365	Salesforce	Generic REST
Data Lake Storage Gen1	Greenplum	SAP BW Open Hub	HDFS	ORC		Dynamics AX	SF Service Cloud	
Data Lake Storage Gen2	HBase	SAP BW via MDX	SFTP	Parquet		Dynamics CRM	SF Marketing Cloud	
Database for MariaDB	Hive	SAP HANA				Google AdWords	SAP C4C	
Database for MySQL	Apache Impala	SAP table				HubSpot	SAP ECC	
Database for PostgreSQL	Informix	Spark				Jira	ServiceNow	
File Storage	MariaDB	SQL Server				Magento	Shopify	
SQL Database	Microsoft Access	Sybase				Marketo	Square	
SQL Database MI	MySQL	Teradata				Office 365	Web table	
SQL Data Warehouse	Netezza	Vertica				Oracle Eloqua	Xero	
Search index						Oracle Responsys	Zoho	
Table storage								

Modern Data Warehouse



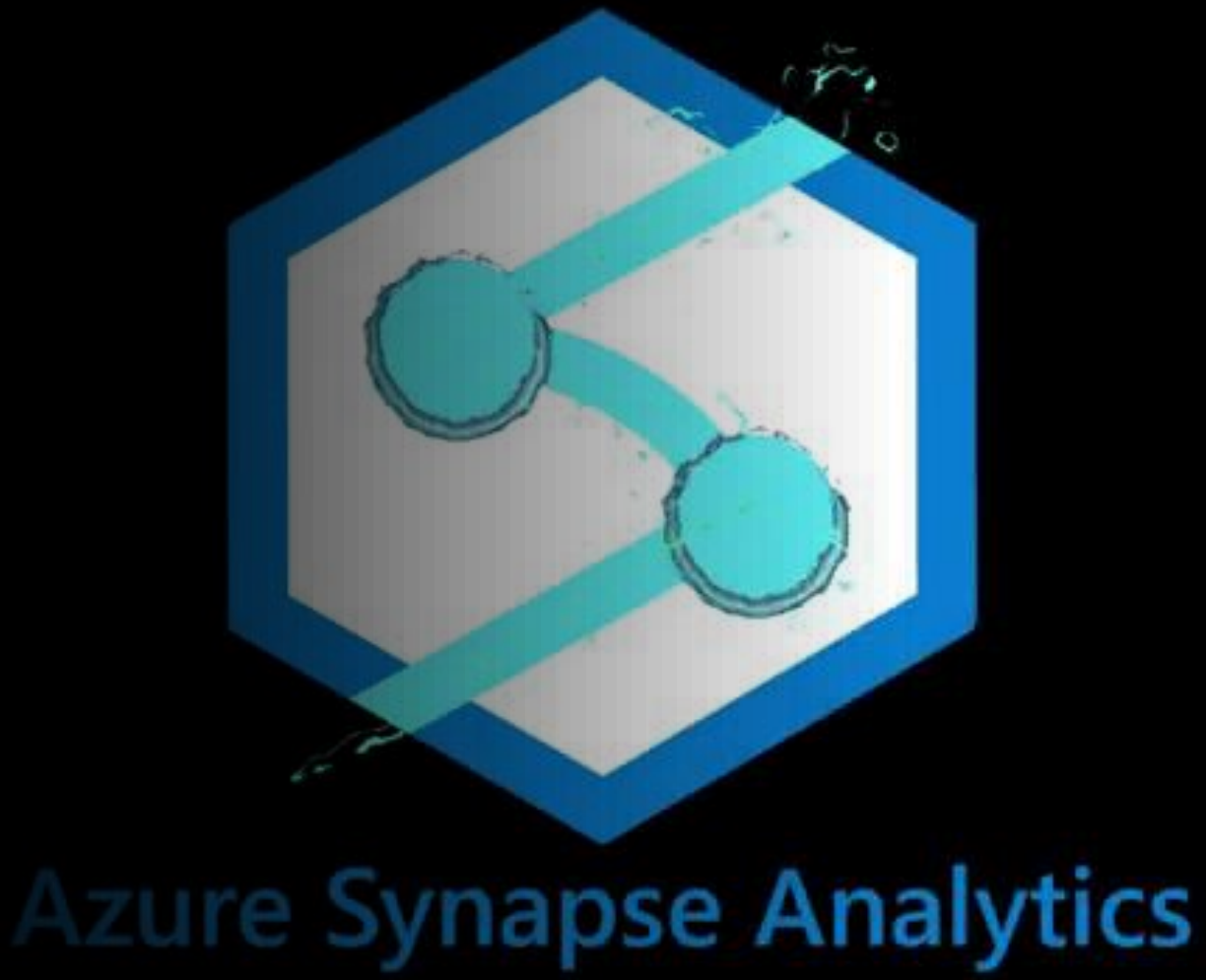
Demonstration

Integration

Quiz!

Integration

https://geni.us/SynapsePrecon4



Types of SQL Pools

SQL on-Demand

- Synapse Automatically creates
- Provides the ability to connect to Azure Data Studio and SSMS
- Can be used for unstructured data

SQL Pools

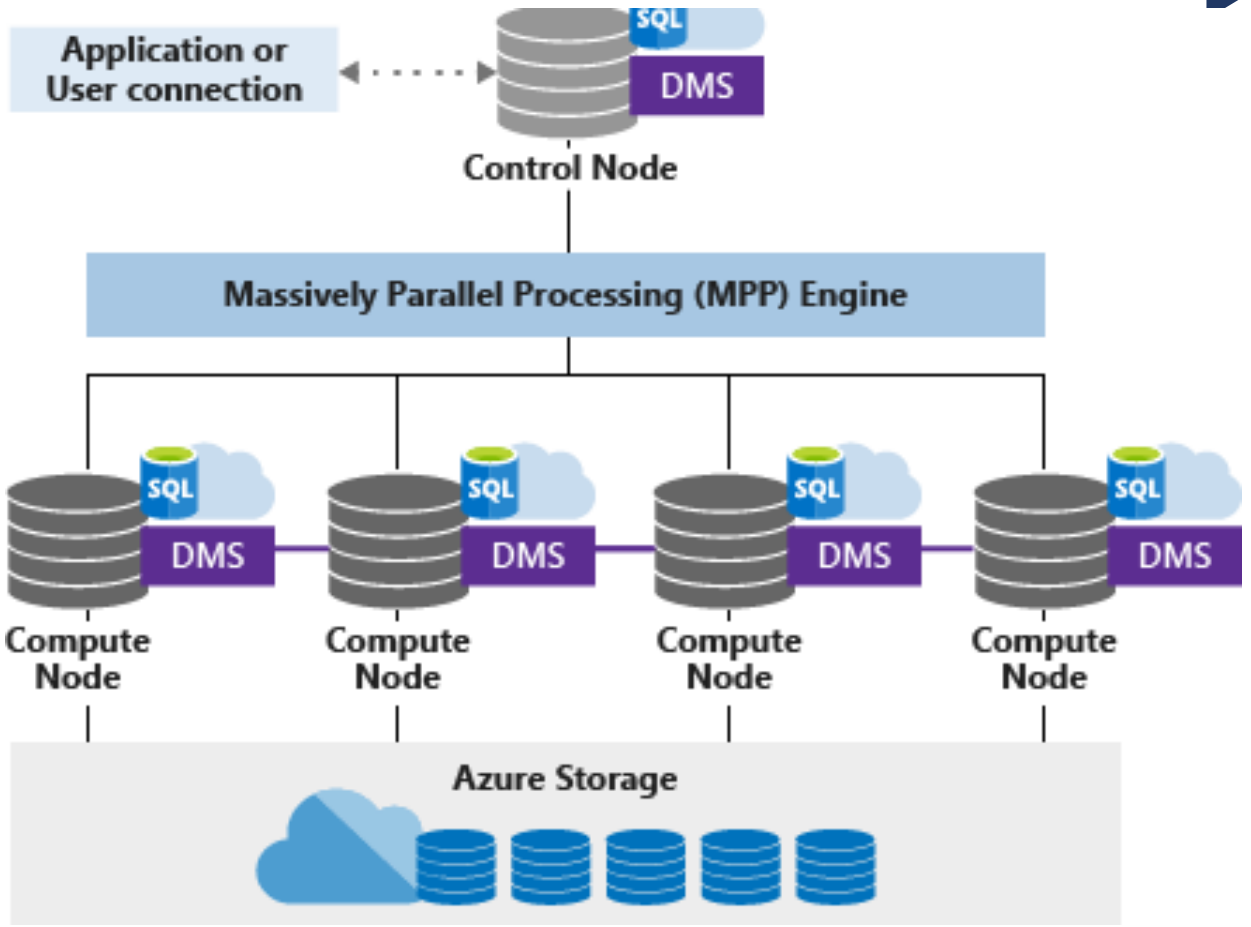
- Manually Created
- SQL DW Replacement
- Provides the ability to connect to Azure Data Studio and SSMS
- Uses MPP to run High-performing queries
- Uses Columnar Storage

SQL Pools

- Data Warehousing part of Synapse
- Resource Collection
- Sized by Data Warehousing Units (DTUs)
- Import data with Polybase
- Paritioning data still important to improve Performance

Synapse SQL MPP Architecture

- Scales across Nodes
- Compute is separate from storage
- Node Based Architecture
- Grow or shrink Compute
- Pause when not in use to save cash



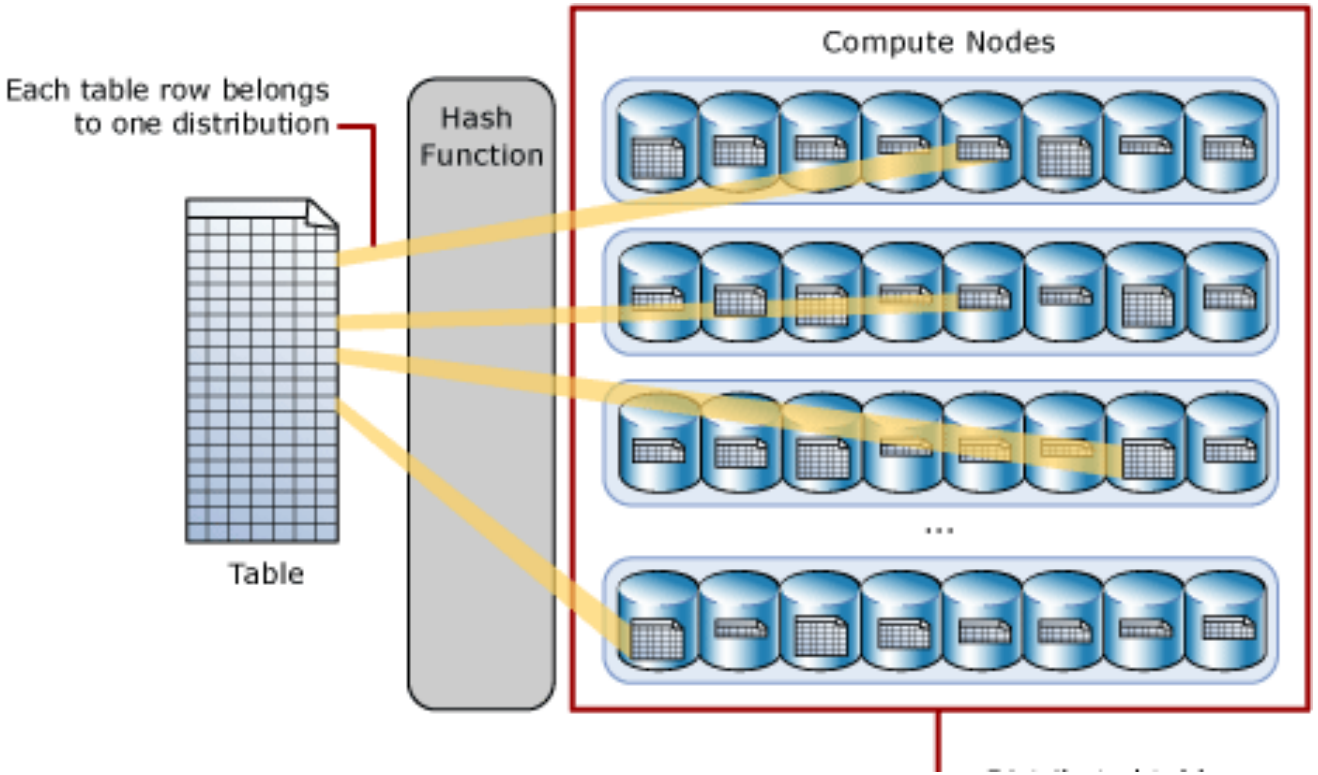
Data Storage Sharding Options

Hash

Round Robin

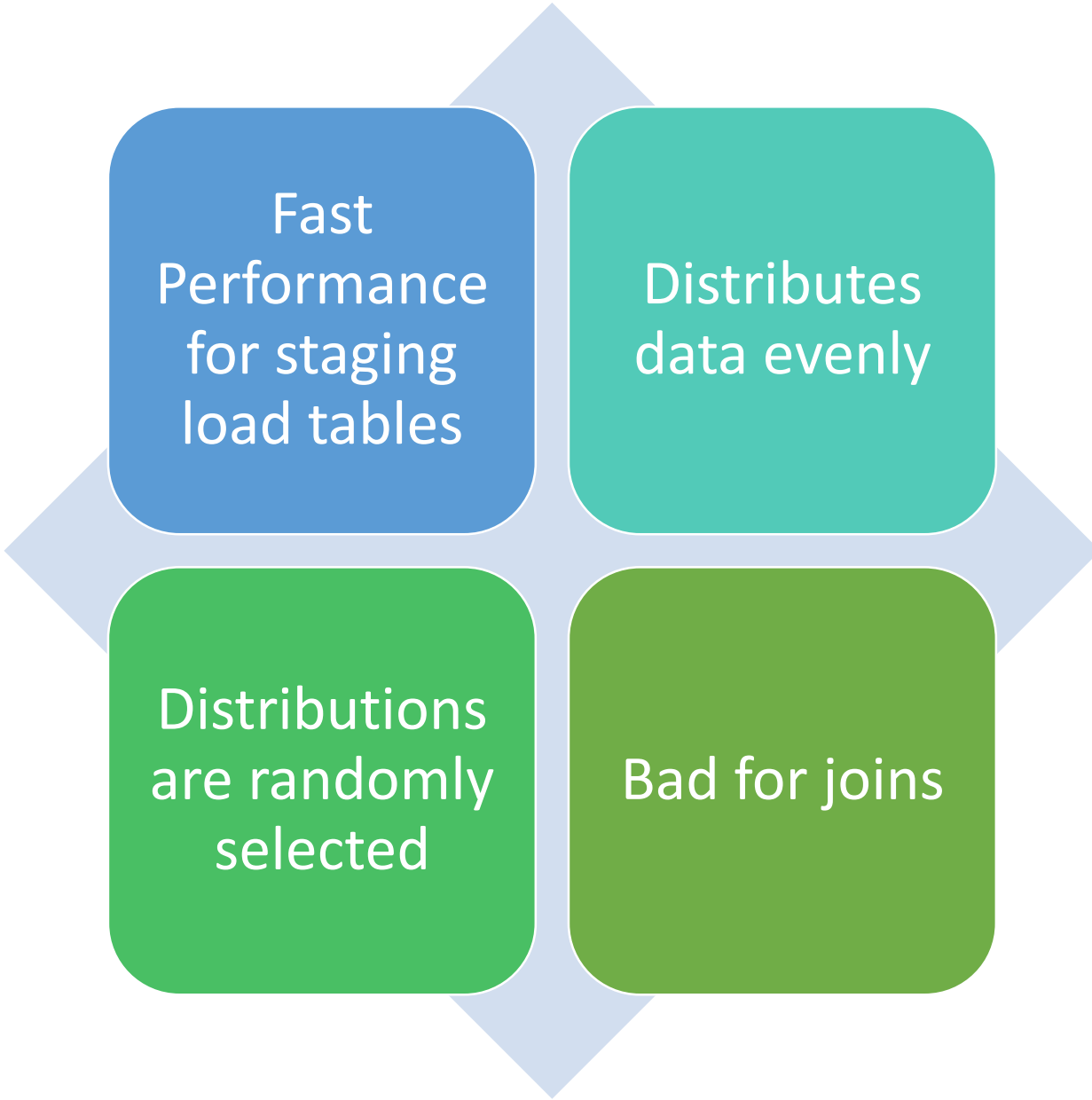
Replicate

Hash-distributed tables

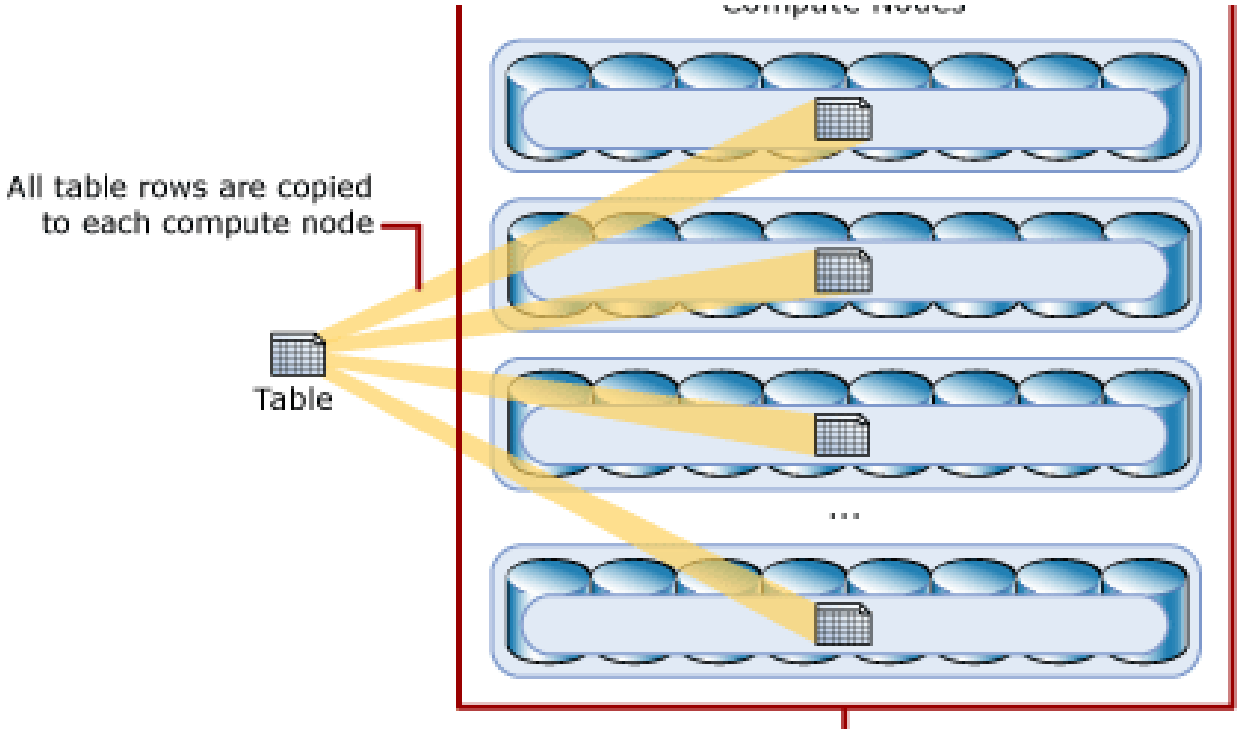


- Good for joins and large table aggregations
- Uses a hash to assign rows to distributions
- Each row belongs to one distribution.
- The number of table rows per distribution varies

Round Robin


Fast Performance for staging load tables
Distributes data evenly
Distributions are randomly selected
Bad for joins

Replicated



Good Performance for small tables

Caches a full copy

Data is not transferred among nodes, Good for Joins

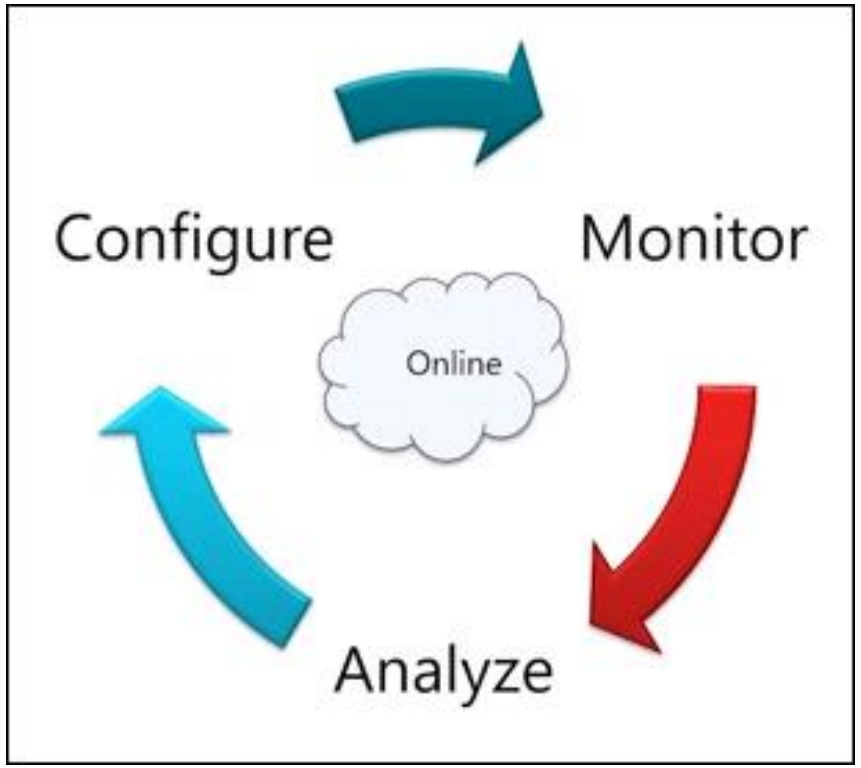
Extra storage is required for large tables

Workload Management

Classification

Importance

Isolation



Workload Classification

GroupName

Members

Importance

```
CREATE WORKLOAD CLASSIFIER
BoardReports WITH
  (WORKLOAD_GROUP ='HIBoardReport'
,MEMBERNAME = 'jsmith'
,IMPORTANCE = HIGH);
```

Workload Classifier

1. User
2. Role
3. Label
4. Context
5. Time

Workload Importance

Levels

5 different Levels

FIFO

Implemented after levels

Workload Groups

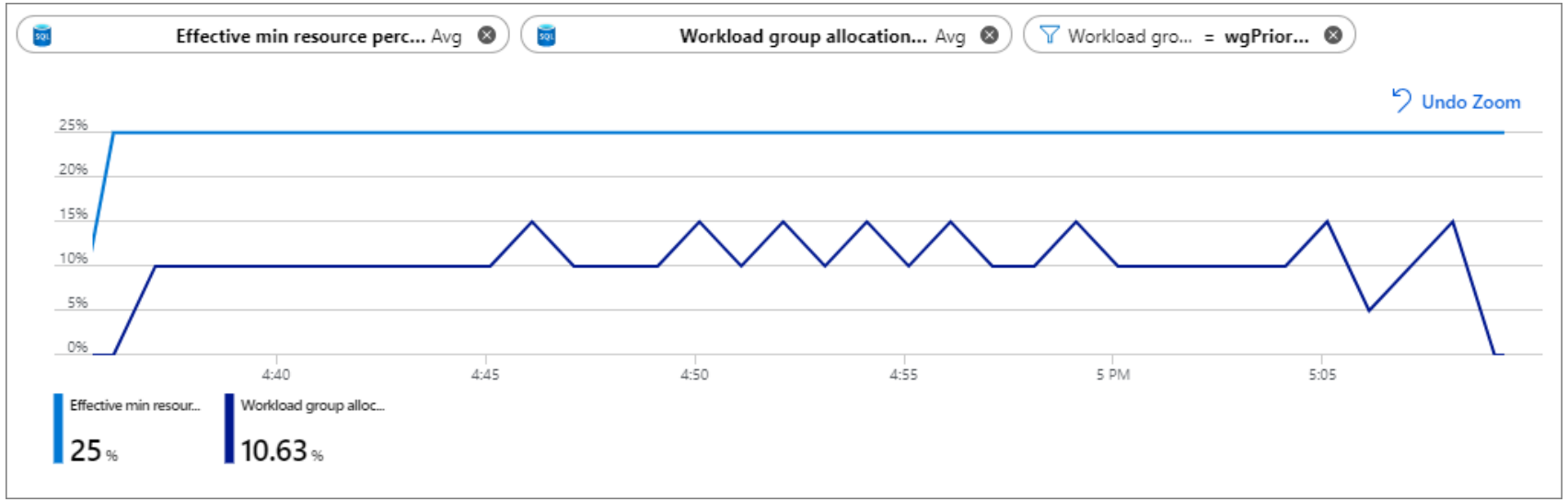
Further Specify Priority

Workload Isolation

Dedicating resources to workloads

```
CREATE WORKLOAD GROUP ELT
WITH ( MIN_PERCENTAGE_RESOURCE = 20
       ,CAP_PERCENTAGE_RESOURCE = 100
,REQUEST_MIN_RESOURCE_GRANT_PERCENT = 5)
```

Workload Monitoring



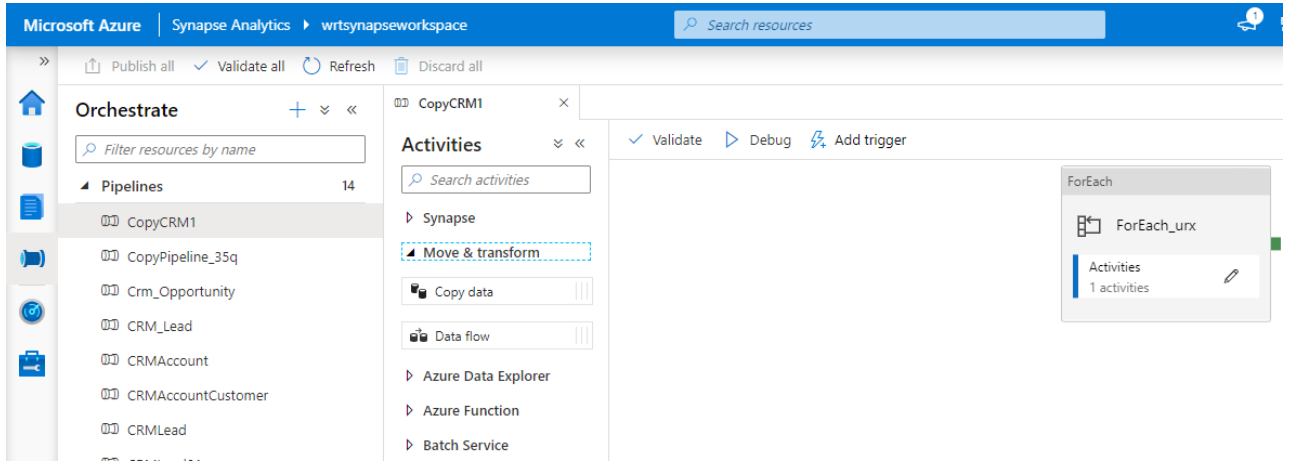
Loading data into SQL Pools

Integrate

Shares the same code with Data Factory

Create Pipelines to import data

Automated Loading Methods



Partitioning Data

Partition Table based

Works on all Distributions

Improves query performance

Not the same as SQL Server Partitions

Can switch Partitions

```SQL
CREATE TABLE [dbo].[FactInternetSales]
(
    [ProductKey]            int          NOT NULL
,   [OrderDateKey]          int          NOT NULL
,   [CustomerKey]           int          NOT NULL
,   [PromotionKey]          int          NOT NULL
,   [SalesOrderNumber]      nvarchar(20) NOT NULL
,   [OrderQuantity]         smallint     NOT NULL
,   [UnitPrice]             money        NOT NULL
,   [SalesAmount]           money        NOT NULL
)
WITH
(   CLUSTERED COLUMNSTORE INDEX
,   DISTRIBUTION = HASH([ProductKey])
,   PARTITION   (   [OrderDateKey] RANGE RIGHT FOR VALUES
                    (20000101,20010101,20020101
                    ,20030101,20040101,20050101
                    )
                )
)
;
```

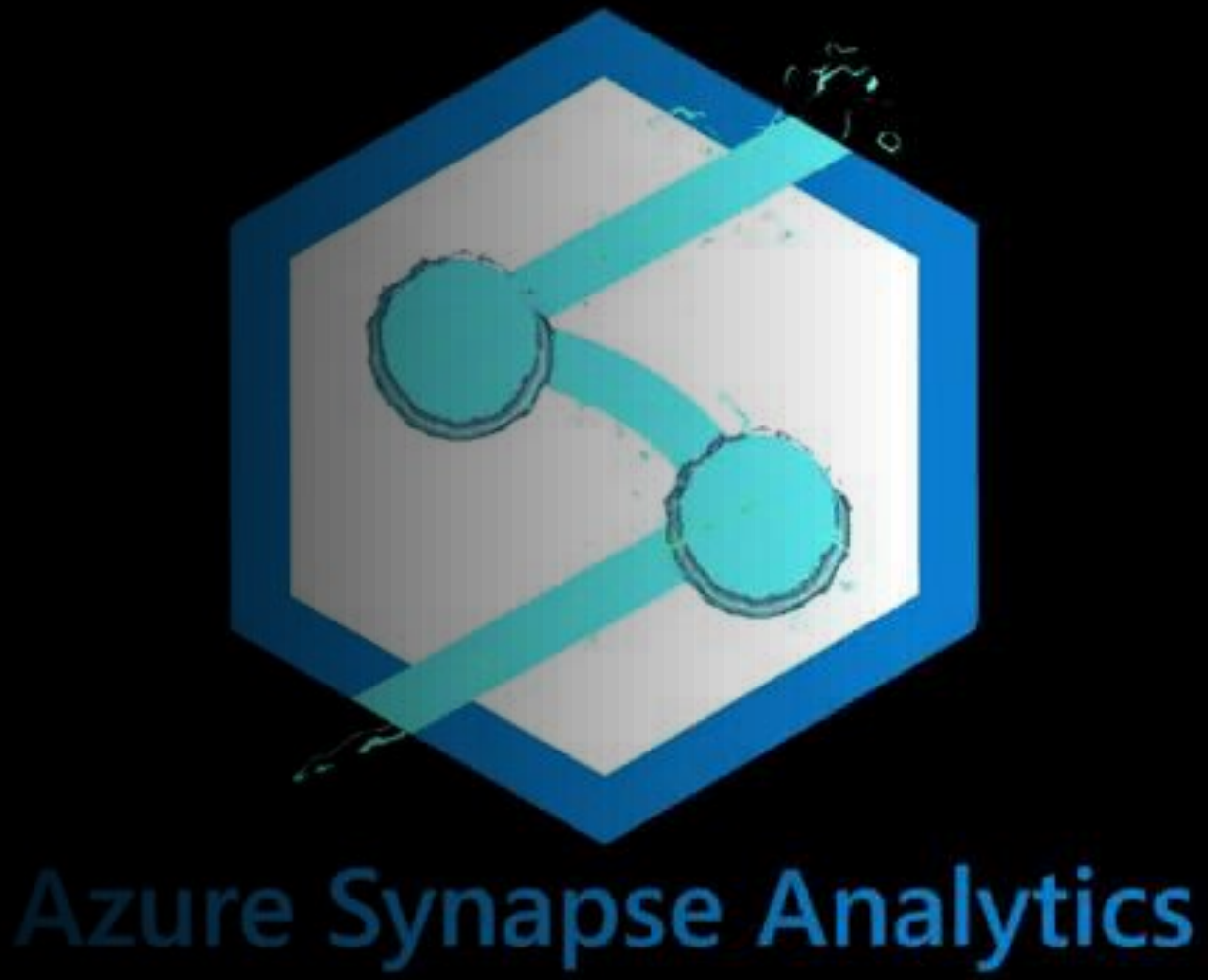
Demonstration

SQL Pools

Quiz!

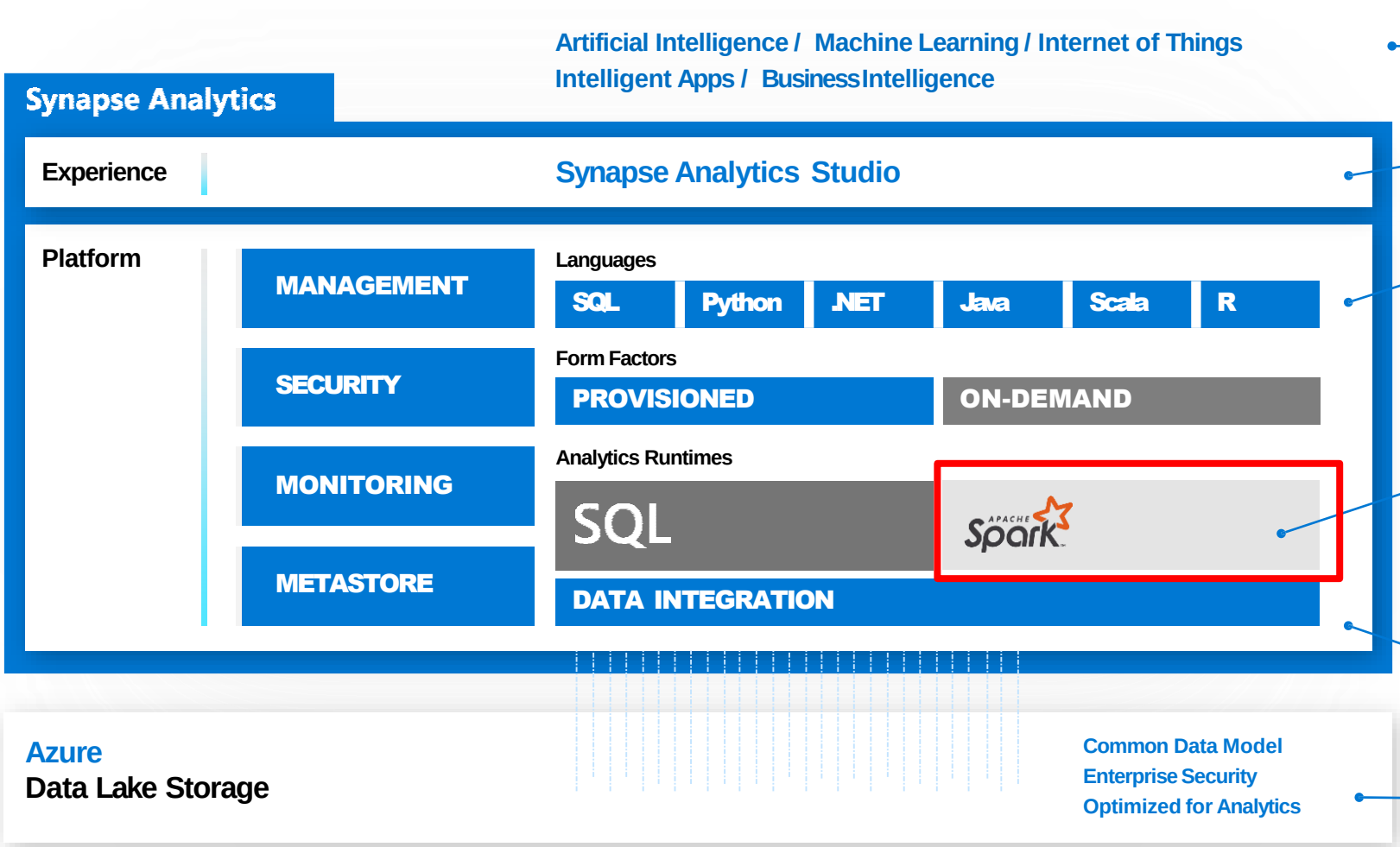
Poll Question

https://geni.us/SynapsePrecon5



Apache Spark

Azure Synapse 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

Apache Spark Pools

- Spark Clusters are also used in Databricks
- Open Source Hadoop Framework In-Memory Processing
- Use data from Azure Data Lake Gen 2
- Process data in HDFS formats including Parquet
- Preloaded with Anaconda
- Integrates with Jetbrains' IntelliJ IDEA to create Spark Pool apps

Summary Azure Synapse Spark

Apache Spark 2.4 derivation

- Linux Foundation Delta Lake 0.4 support
- .Net Core 3.0 support
- Python 3.6 + Anacondas support

Core scenarios

- Data Prep/Data Engineering/ETL
- Machine Learning via Spark ML and Azure ML integration
- Extensible through library management

Tightly coupled to other Azure Synapse services

- Integrated security and sign on
- Integrated Metadata
- Integrated and simplified provisioning
- Integrated UX including Notebooks
- Fast load of SQL Analytics pools

Multi Language Support

.Net (C#),
Python
Scala
SQL
Java

Efficient resource utilization

Fast Start
Auto scale (up and down)
Auto pause
Min cluster size of 3 nodes

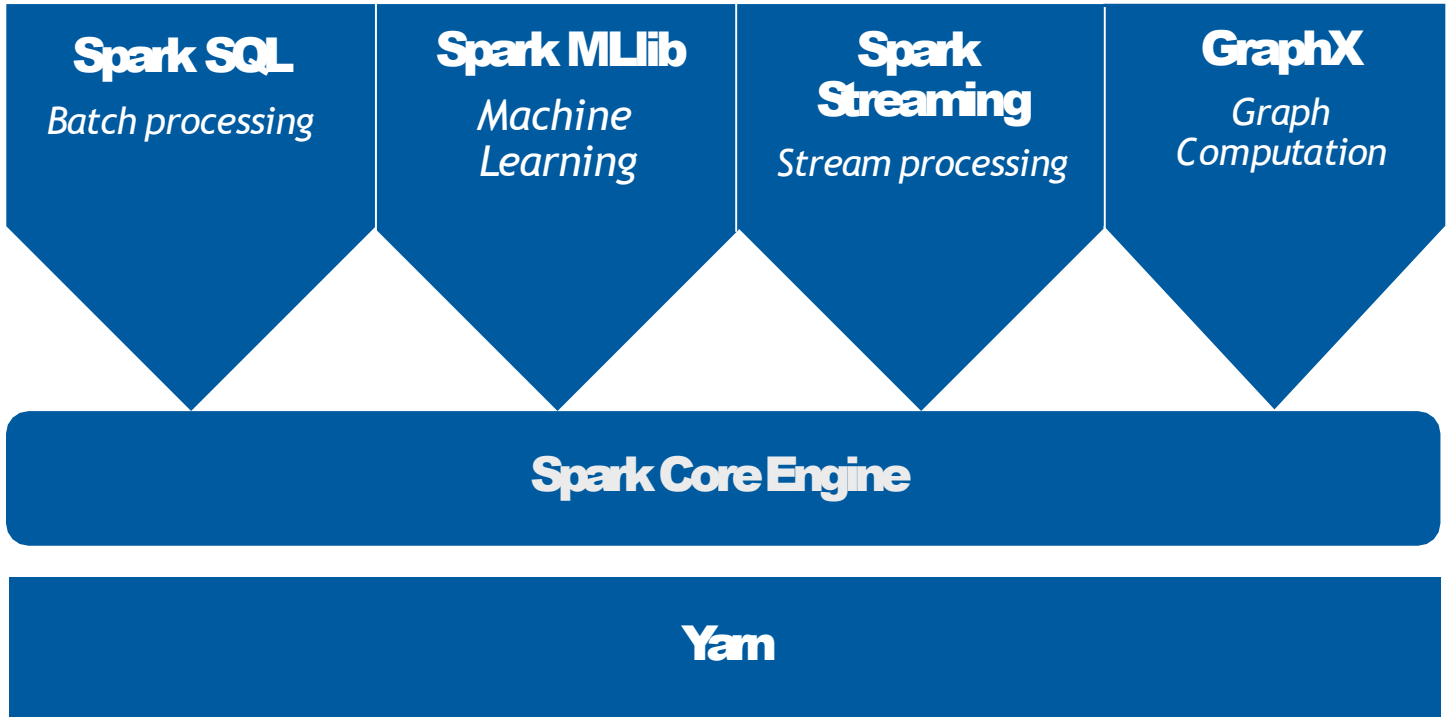
Apache Spark

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

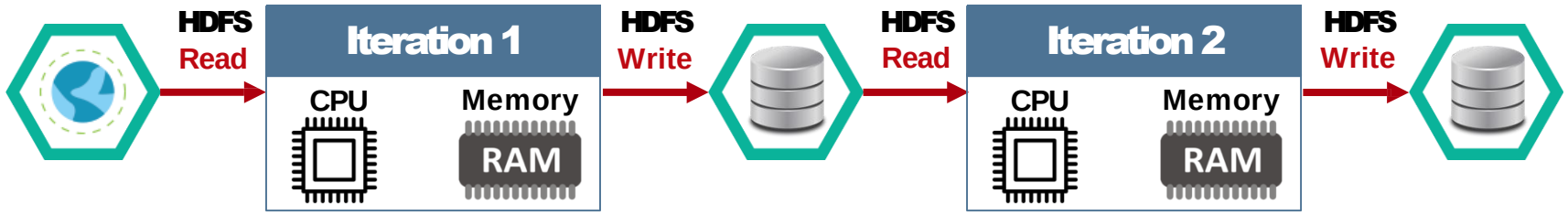
Graph Processing



http://spark.apache.org

Motivation for ApacheSpark

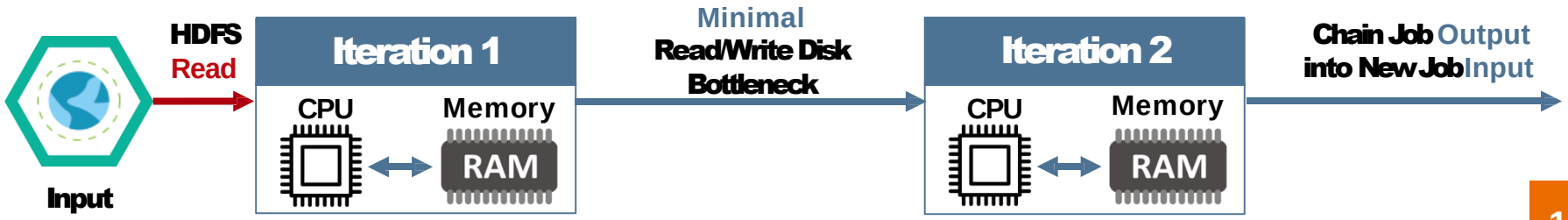
Traditional Approach: MapReduce jobs for complex jobs, interactive query, and online event-hub processing involves lots of (slow) disk I/O



Motivation for ApacheSpark

Traditional Approach: MapReduce jobs for complex jobs, interactive query, and online event-hub processing involves lots of (slow) disk I/O

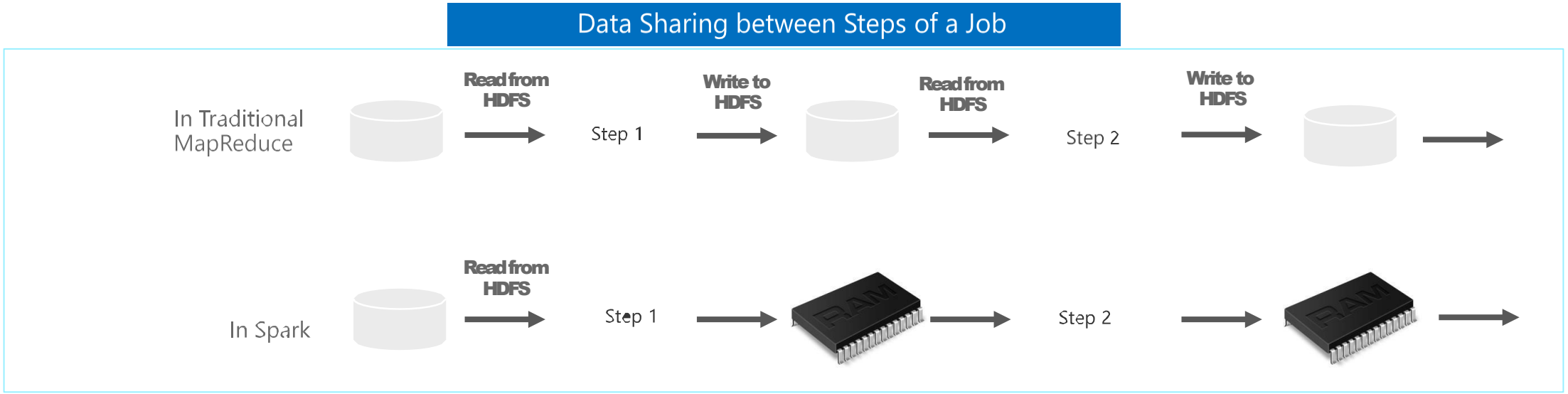
Solution: Keep data in-memory with a new distributed execution engine



10–100x faster than network & disk

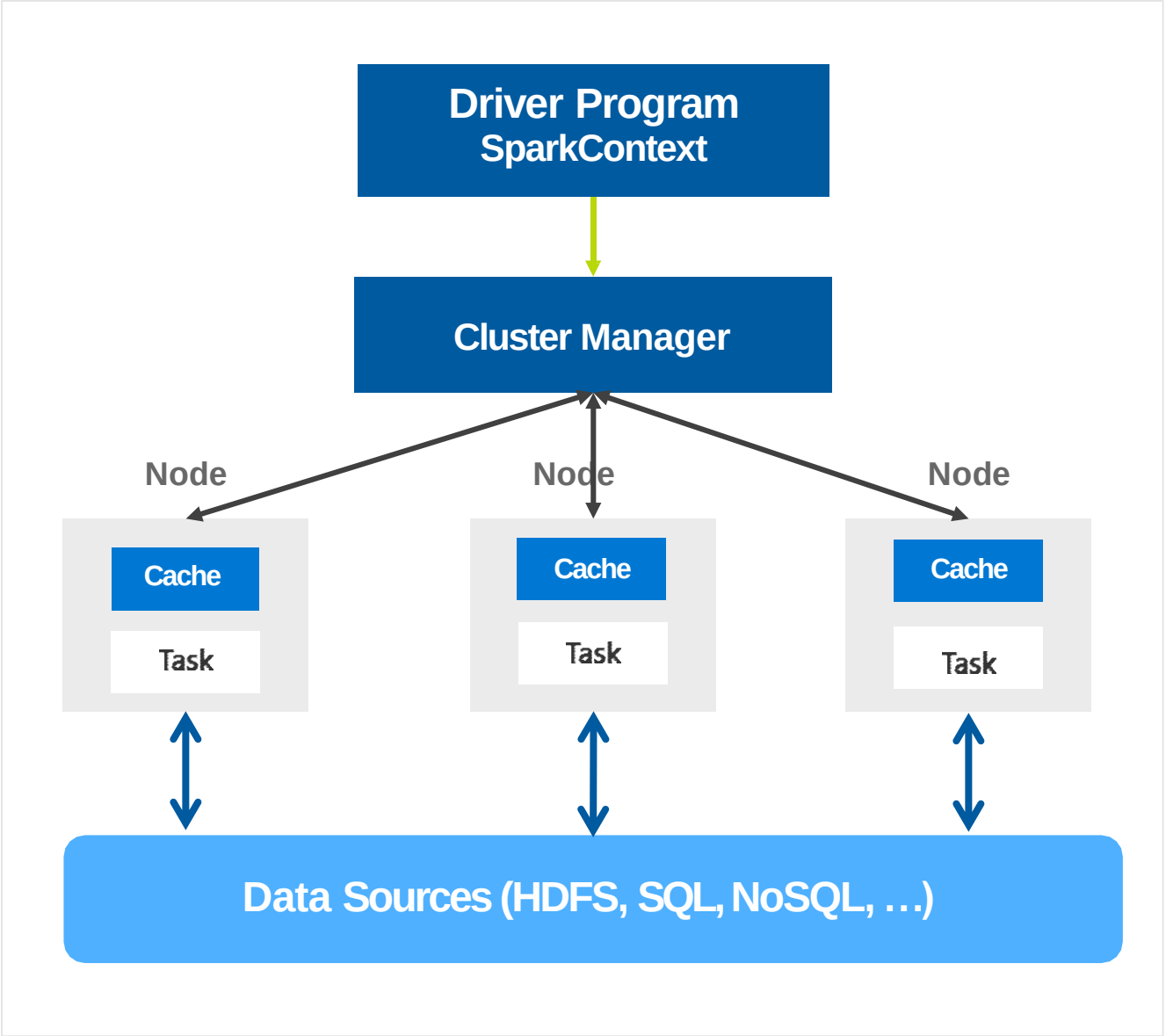
What makes Spark fast

- **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*.



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).



Spark Component Features

Spark SQL

- Unified data access: Query structured data sets with SQL or DataFrame APIs
- Fast, familiar query language across all your enterprise data
- Use BI tools to connect and query via JDBC or ODBC drivers

Mllib/SparkML

- Predictive and prescriptive analytics
- Machine learning algorithms for:
  - Clustering
  - Classification
  - Regression
  - etc.
- Smart application design from pre-built, out-of-the-box statistical and algorithmic models

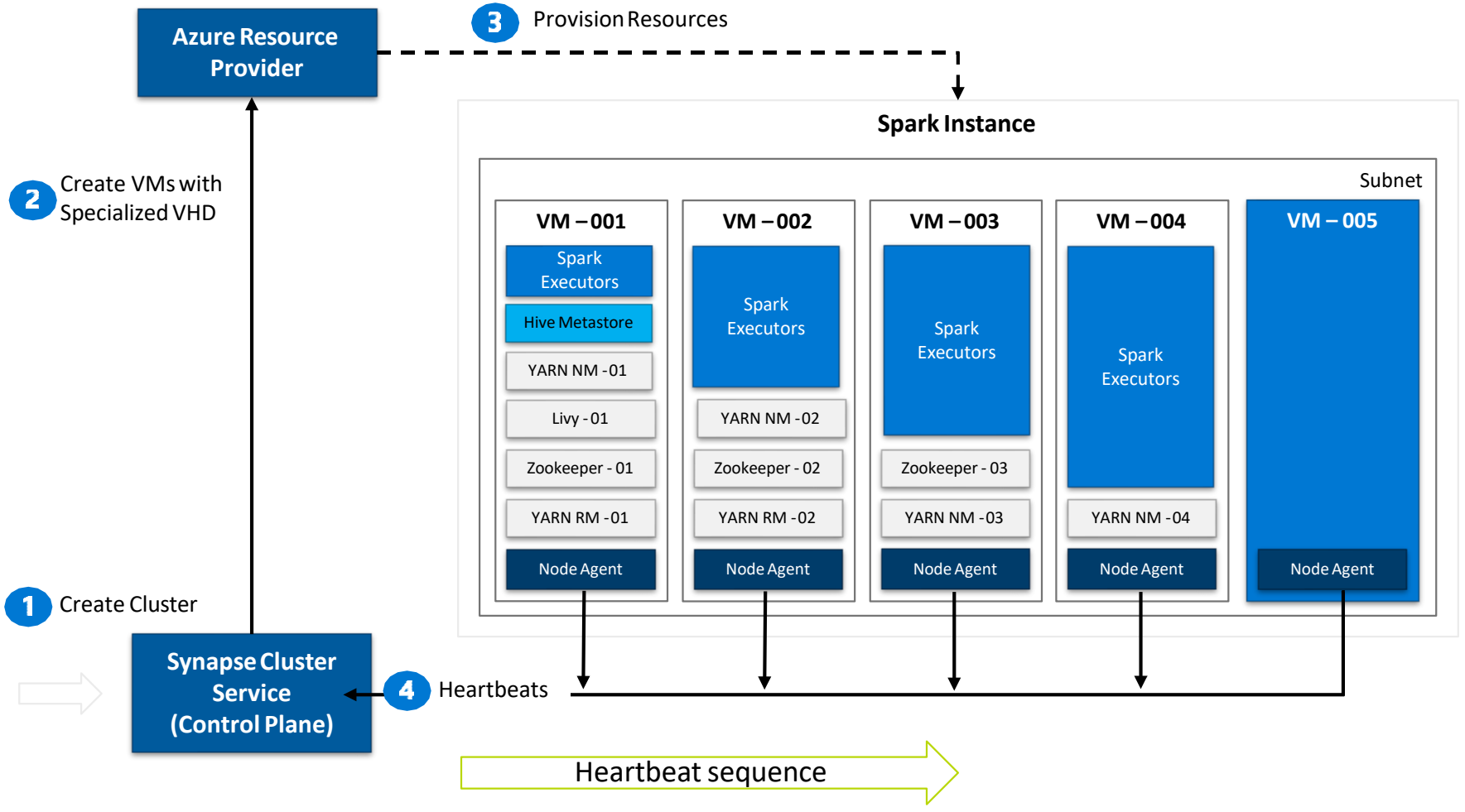
Spark Streaming

- Micro-batch event processing for near-real time analytics
- e.g. Internet of Things (IoT) devices, Twitter feeds, Kafka (event hub), etc.
- Spark's engine drives some action or outputs data in batches to various data stores

GraphX

- Represent and analyze systems represented by graph nodes
- Trace interconnections between graph nodes
- Applicable to use cases in transportation, telecommunications, road networks, modeling personal relationships, social media, etc.

Synapse Spark Instances



1. Synapse Job Service sends request to Cluster Service for creating BBC clusters per the description in the associated Spark pool.
2. Cluster Service sends request to Azure using Azure SDK to create VMs (required plus additional) with specialized VHD.
3. The specialized VHD contains bits for all the services that are required by the Cluster type (for e.g. Spark) with prefetch instrumentation.
4. Once VM boots up, the Node Agent sends heartbeat to Cluster Service for getting node configuration.
5. The nodes are initialized and assigned roles based on their first heartbeat.
6. Extra nodes get deleted on first heartbeat.
7. After Cluster Service considers the cluster ready, it returns the Livy end-point to the Job Service.

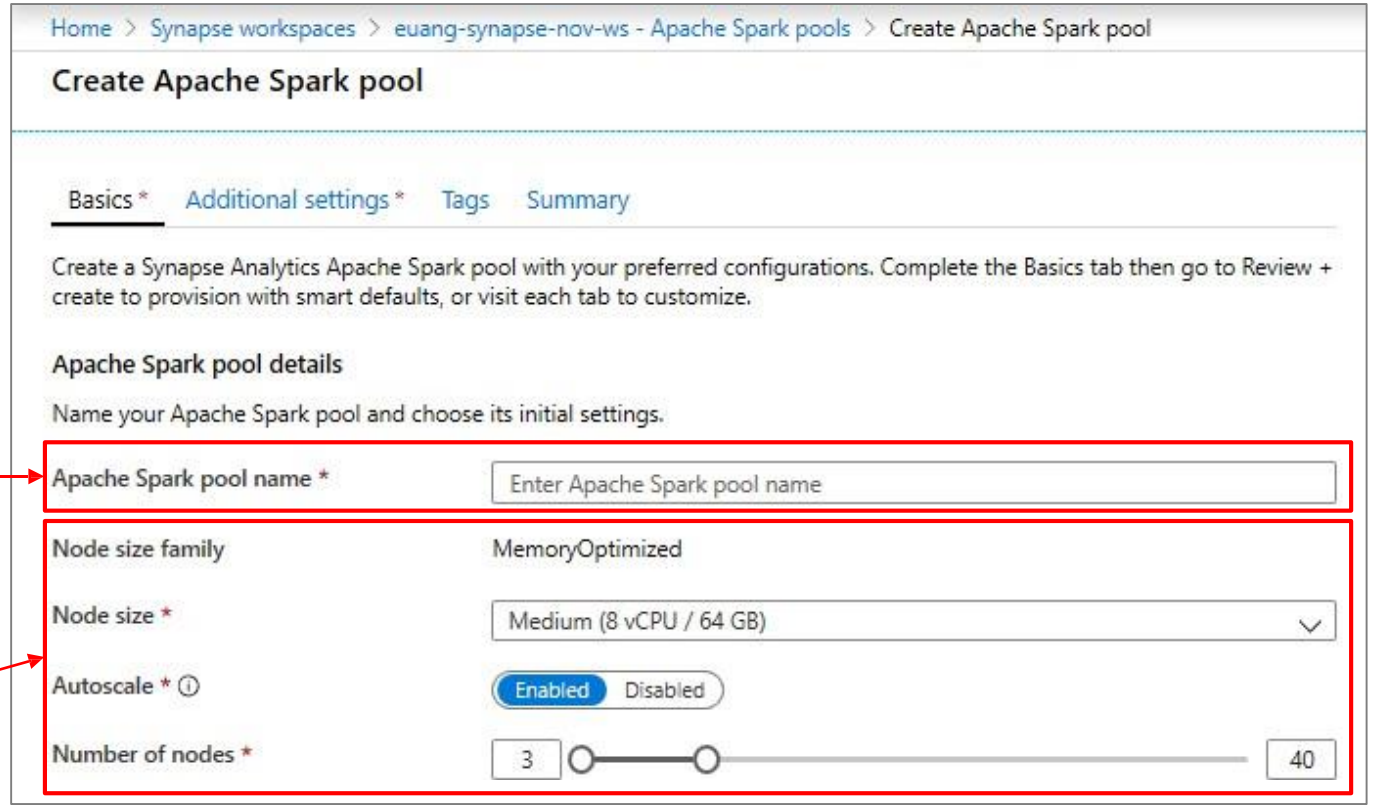
Creating a Spark pool

Provision Spark Pool through Azure Portal with default settings or per requirements

Basic Settings – Minimum details required from user

Only required field from user

Default Settings

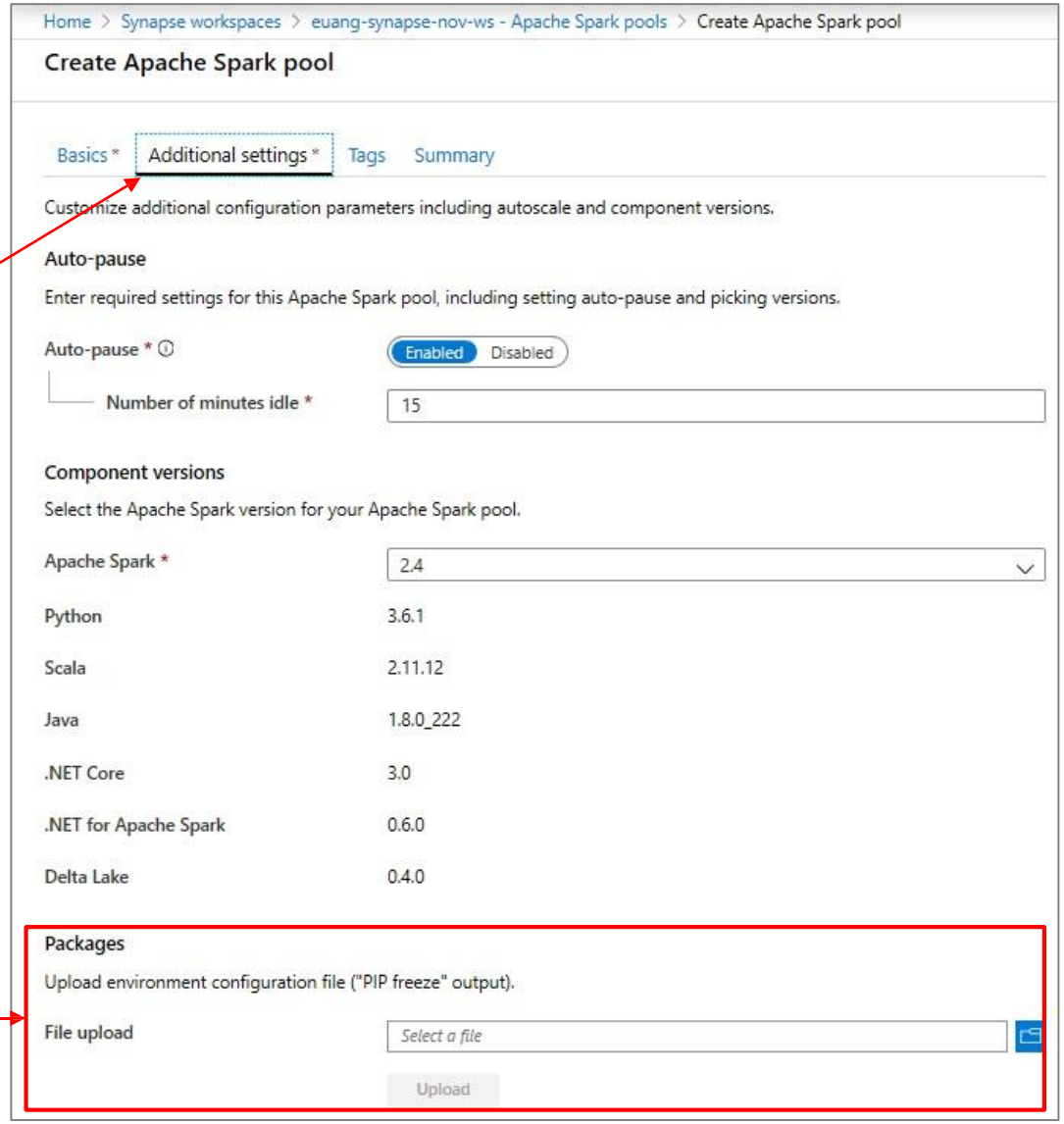


Optional settings for Creating a Spark pool

Additional Settings offer optional settings to customize Spark pool

Customize component versions, auto-pause

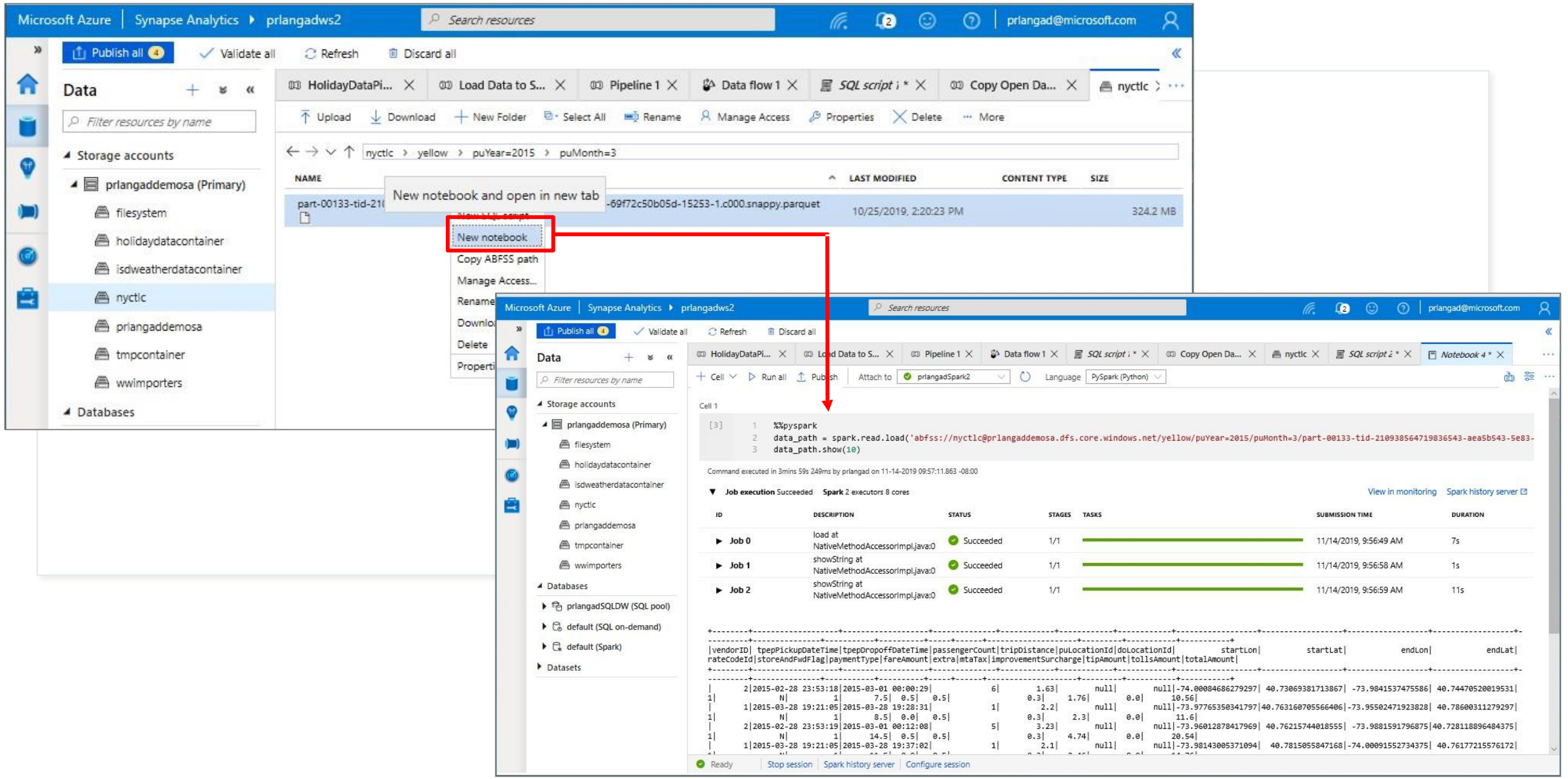
Import libraries by providing text file containing library name and version



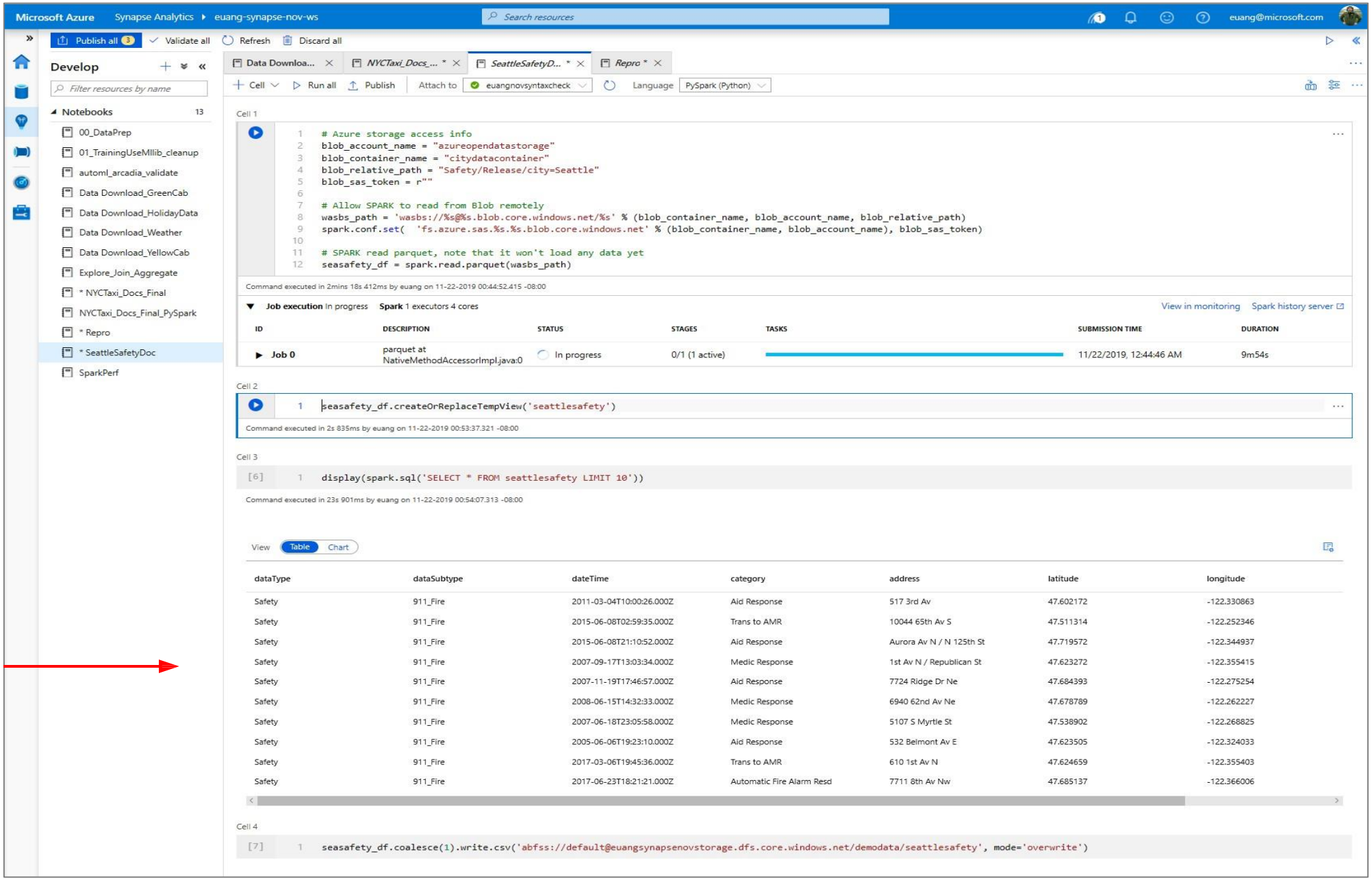
Demonstration

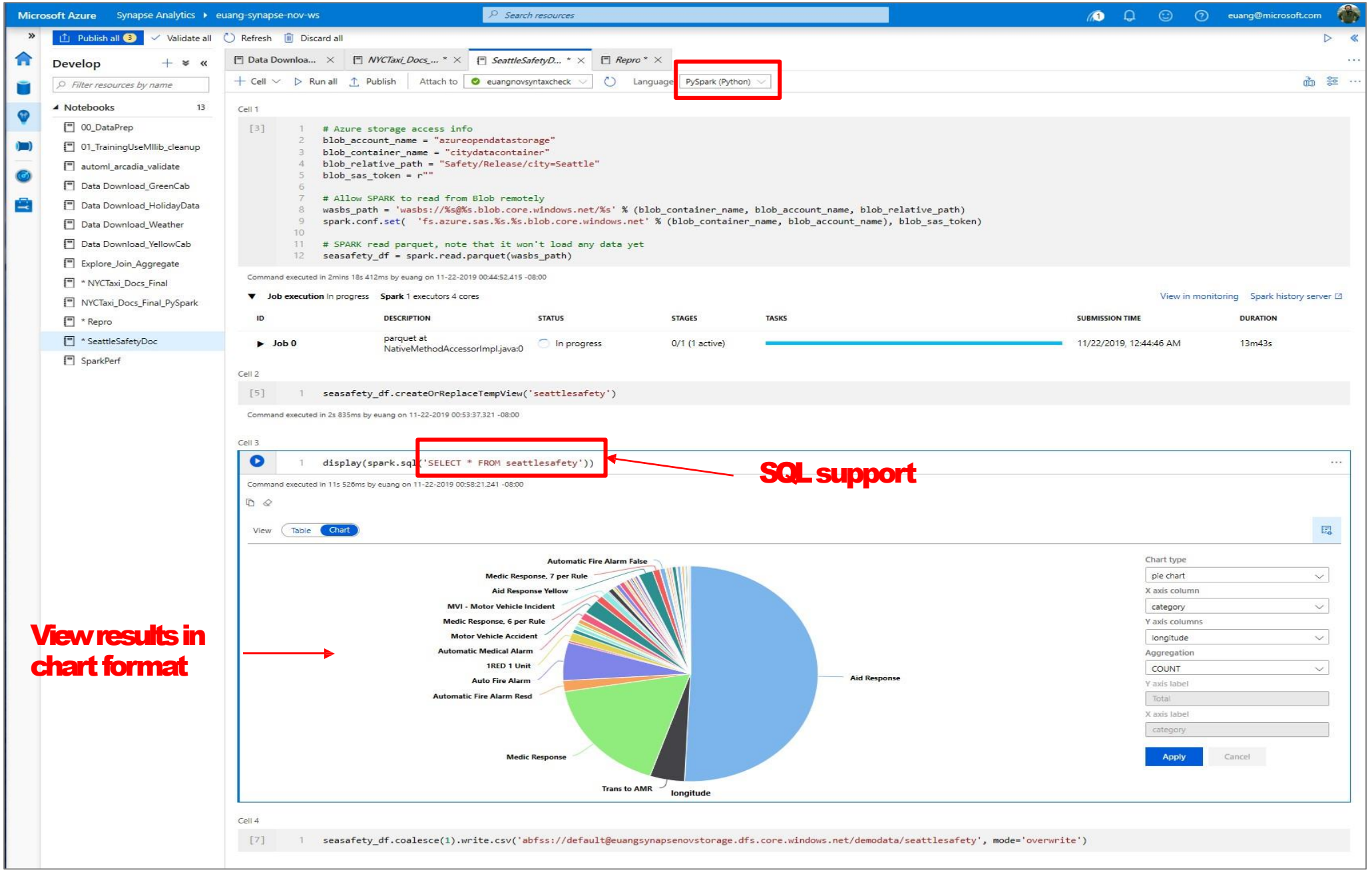
Spark Pools

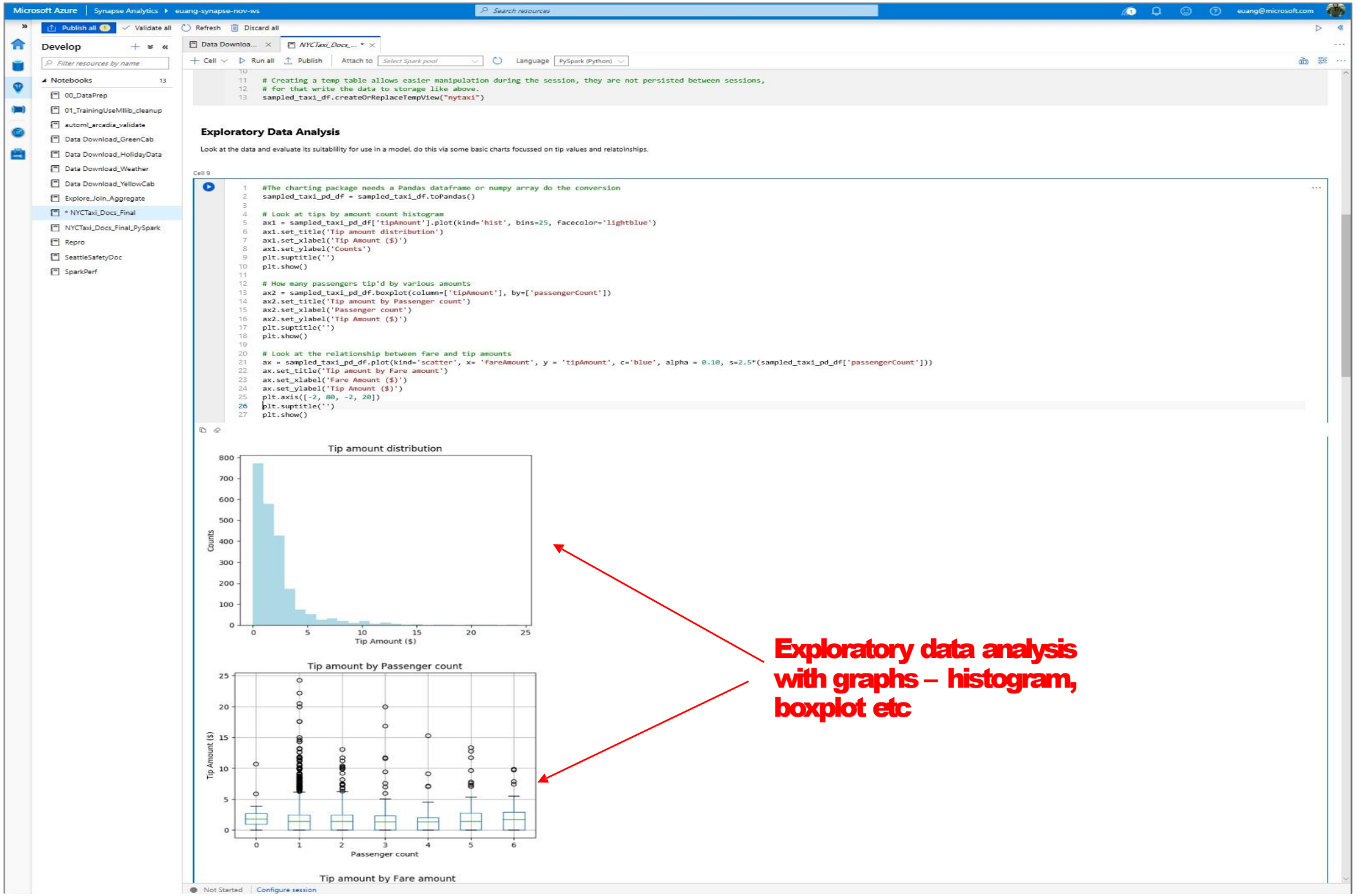
Create Notebook on files in storage



View results in table format




Microsoft Azure
Synapse Analytics
euang-synapse-nov-ws
Search resources
euang@microsoft.com
Publish all 3
Validate all
Refresh
Discard all
Develop
Filter resources by name
Notebooks 13
00_DataPrep
01_TrainingUseMllib_cleanup
automl_arcadia_validate
Data Download_GreenCab
Data Download_HolidayData
Data Download_Weather
Data Download_YellowCab
Explore_Join_Aggregate
* NYCTaxi_Docs_Final
NYCTaxi_Docs_Final_PySpark
* Repro
* SeattleSafetyDoc
SparkPerf
Data Downloa...
NYCTaxi_Docs_... *
SeattleSafetyD... *
Repro *
Cell
Run all
Publish
Attach to
euangnovsyntaxcheck
Language
PySpark (Python)
Cell 1
[3]
# Azure storage access info
blob_account_name = "azureopendatastorage"
blob_container_name = "citydatacontainer"
blob_relative_path = "Safety/Release/city=Seattle"
blob_sas_token = r""
# Allow SPARK to read from Blob remotely
wasbs_path = 'wasbs://%s@%s.blob.core.windows.net/%s' % (blob_container_name, blob_account_name, blob_relative_path)
spark.conf.set('fs.azure.sas.%s.%s.blob.core.windows.net' % (blob_container_name, blob_account_name), blob_sas_token)
# SPARK read parquet, note that it won't load any data yet
seasafety_df = spark.read.parquet(wasbs_path)
Command executed in 2mins 18s 412ms by euang on 11-22-2019 00:44:52.415 -08:00
Job execution In progress
Spark 1 executors 4 cores
View in monitoring
Spark history server
ID
DESCRIPTION
STATUS
STAGES
TASKS
SUBMISSION TIME
DURATION
Job 0
parquet at NativeMethodAccessorImpl.java:0
In progress
0/1 (1 active)
11/22/2019, 12:44:46 AM
13m43s
Cell 2
[5]
seasafety_df.createOrReplaceTempView('seattlesafety')
Command executed in 2s 835ms by euang on 11-22-2019 00:53:37.321 -08:00
Cell 3
display(spark.sql('SELECT * FROM seattlesafety'))
Command executed in 11s 526ms by euang on 11-22-2019 00:58:21.241 -08:00
View
Table
Chart
Automatic Fire Alarm False
Medic Response, 7 per Rule
Aid Response Yellow
MVI - Motor Vehicle Incident
Medic Response, 6 per Rule
Motor Vehicle Accident
Automatic Medical Alarm
1RED 1 Unit
Auto Fire Alarm
Automatic Fire Alarm Resd
Medic Response
Trans to AMR
Aid Response
longitude
Chart type
pie chart
X axis column
category
Y axis columns
longitude
Aggregation
COUNT
Y axis label
Total
X axis label
category
Apply
Cancel
Cell 4
[7]
seasafety_df.coalesce(1).write.csv('abfss://default@euangsynapsenovstorage.dfs.core.windows.net/demodata/seattlesafety', mode='overwrite')
SQL support
View results in chart format



Exploratory data analysis with graphs – histogram, boxplot etc

Library Management - Python

Customers can add new python libraries at Spark pool level

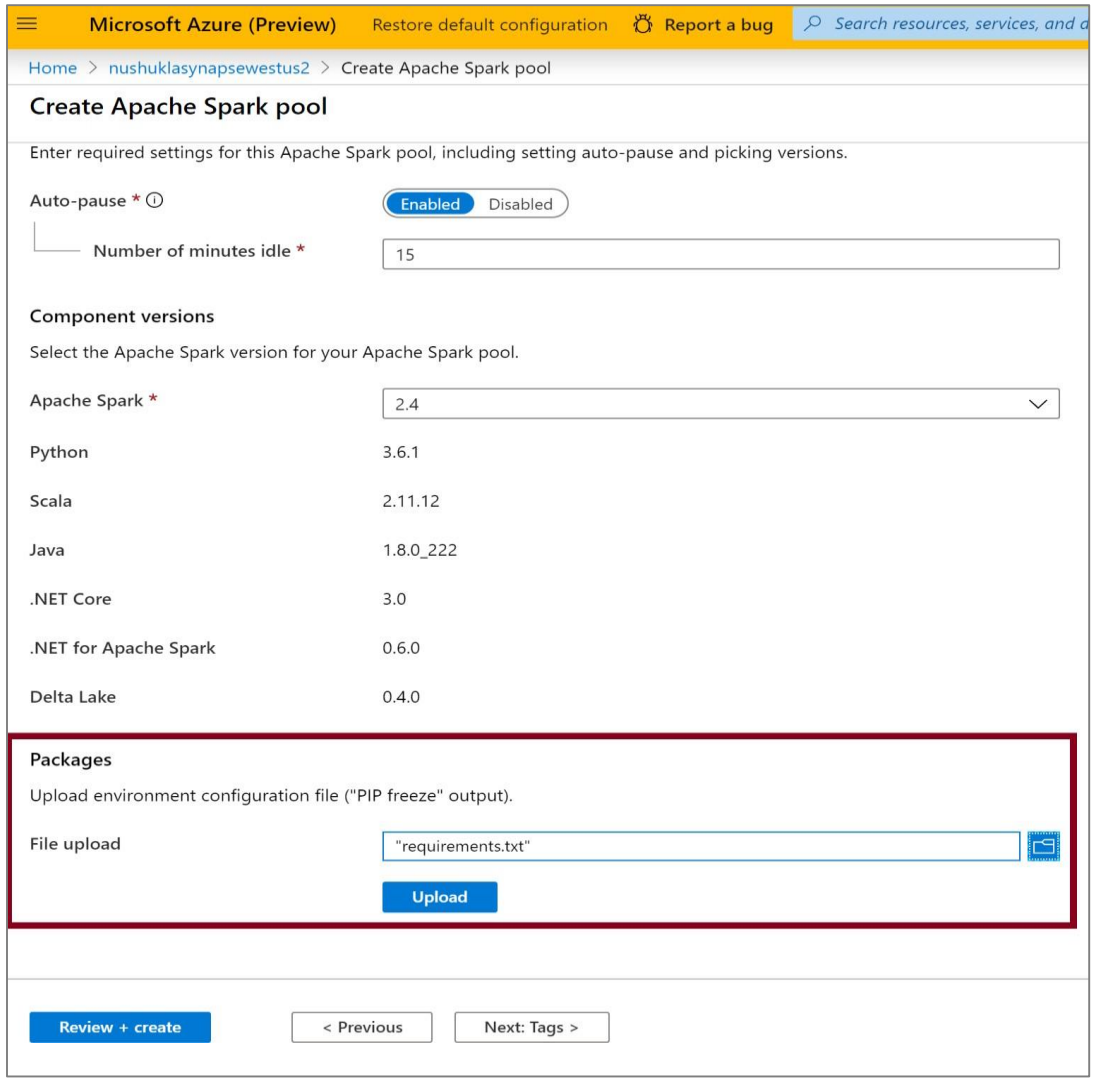
Benefits

- 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

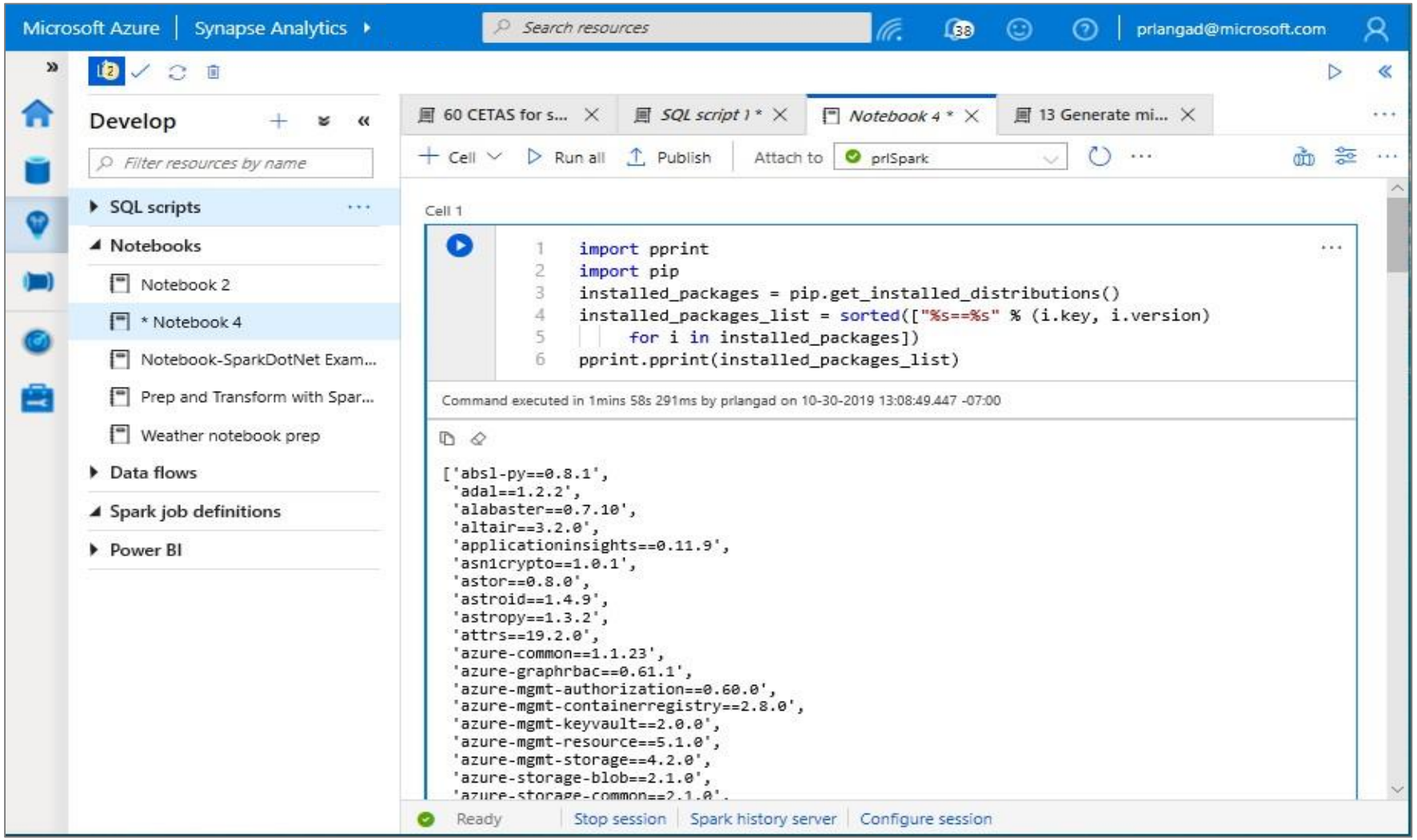
The library version must exist on PyPI repository Version downgrade of an existing library not allowed

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



Library Management - Python

Get list of installed libraries with version information

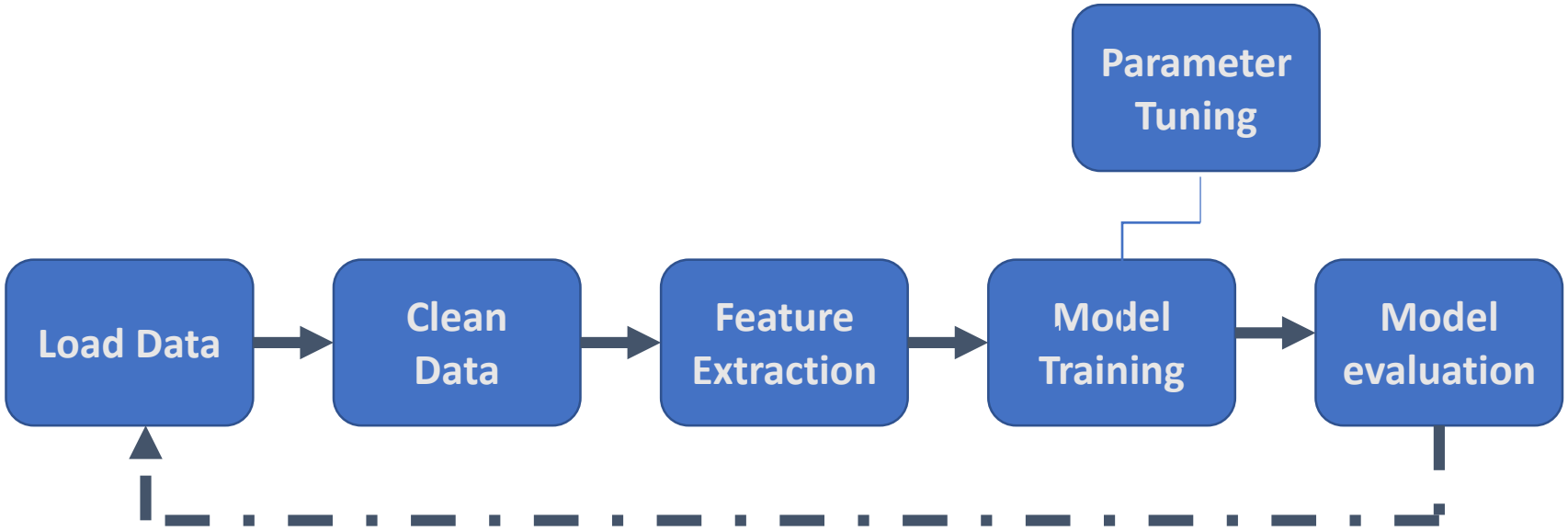


Machine Learning with Spark

- Open Source Apache Hadoop project
- Resilient Distributed Datasets
- MLLib

Machine Learning Pipeline

Standard Workflow



What is inside of Mllib?

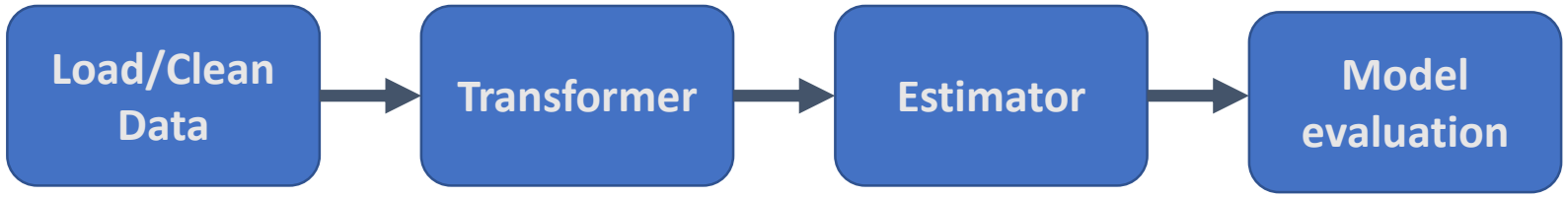
- Algorithms
- Featurization
- Utilities
- Pipeline

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

ML Lib Pipeline

Spark



Sample MLlib Pipeline Python Code

```
from pyspark.ml import Pipeline
from pyspark.ml.feature import HashingTF, Tokenizer
from pyspark.ml.classification import LogisticRegression


# Configure an ML pipeline, which consists of three stages: tokenizer, hashingTF, and
lr.
tokenizer = Tokenizer(inputCol="text", outputCol="words")
hashingTF = HashingTF(inputCol=tokenizer.getOutputCol(),
outputCol="features")
lr = LogisticRegression(maxIter=10, regParam=0.001)
pipeline = Pipeline(stages=[tokenizer, hashingTF, lr])


# Fit the pipeline to training documents.
model = pipeline.fit(training)


# Make predictions
prediction = model.transform(test)
```

Pipeline Parameter Tuning

- Automate model tuning in the Pipeline
- MLLib will select the best hyperparamters for the modle
- Use the individual algorithms in the Estimator or the whole Pipeline
- Data is evaluated to determine the best result for you

Python Code for Pipeline Parameter tuning

```
// Pipeline is treated as an Estimator, wrapping it in a CrossValidator
instance.
// This will allow us to jointly choose parameters for all Pipeline stages.

val cv = new CrossValidator()
  .setEstimator(pipeline)
  .setEvaluator(new BinaryClassificationEvaluator)
  .setEstimatorParamMaps(paramGrid)
  .setNumFolds(2)  // Use 3+ in practice
  .setParallelism(2)  // Evaluate up to 2 parameter settings in
parallel
```

Delta Lake

New project announced in October 2017, released April 2019

Apache Open Source

Meant to address problems with Data Lakes

Based on Parquet

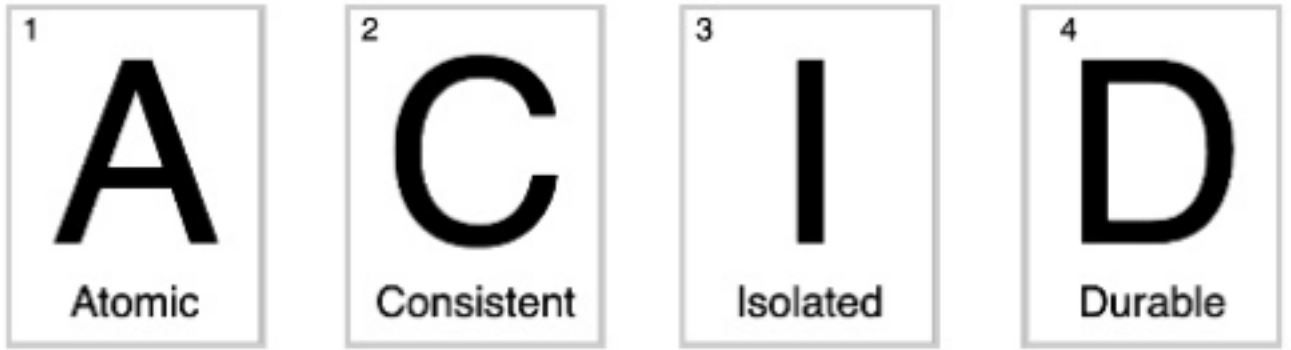
Includes Indexing and Timetravel

Delta Lake Feature List

ACID Transactional reliability

Enforces Schema on write

File Compression



Delta Lake Ensures Reliability

Enforces the schema

Contains Unified Batch and Streaming as they can be the same table

Data Snapshots

Unified Data Management

Performance Tuning Delta Lake

Partitioning (filter part of the data)

```
OPTIMIZE <table> [WHERE <partition_filter>]
```

Compaction

```
OPTIMIZE <table> [WHERE <partition_filter>]
ZORDER BY (<column>[, …])
```

Z-Ordering (very Index like)

How Data Skipping Works internally

```
SELECT input_file_name() as "file_name",
       min(col) AS "col_min",
       max(col) AS "col_max"
FROM table
GROUP BY input_file_name()
```

file_name	col_min	col_max
data_file_1	6	8
data_file_2	3	10
data_file_3	1	4

Data Skipping means that Delta Lakes knows to only look at data_file_3 and data_file 2

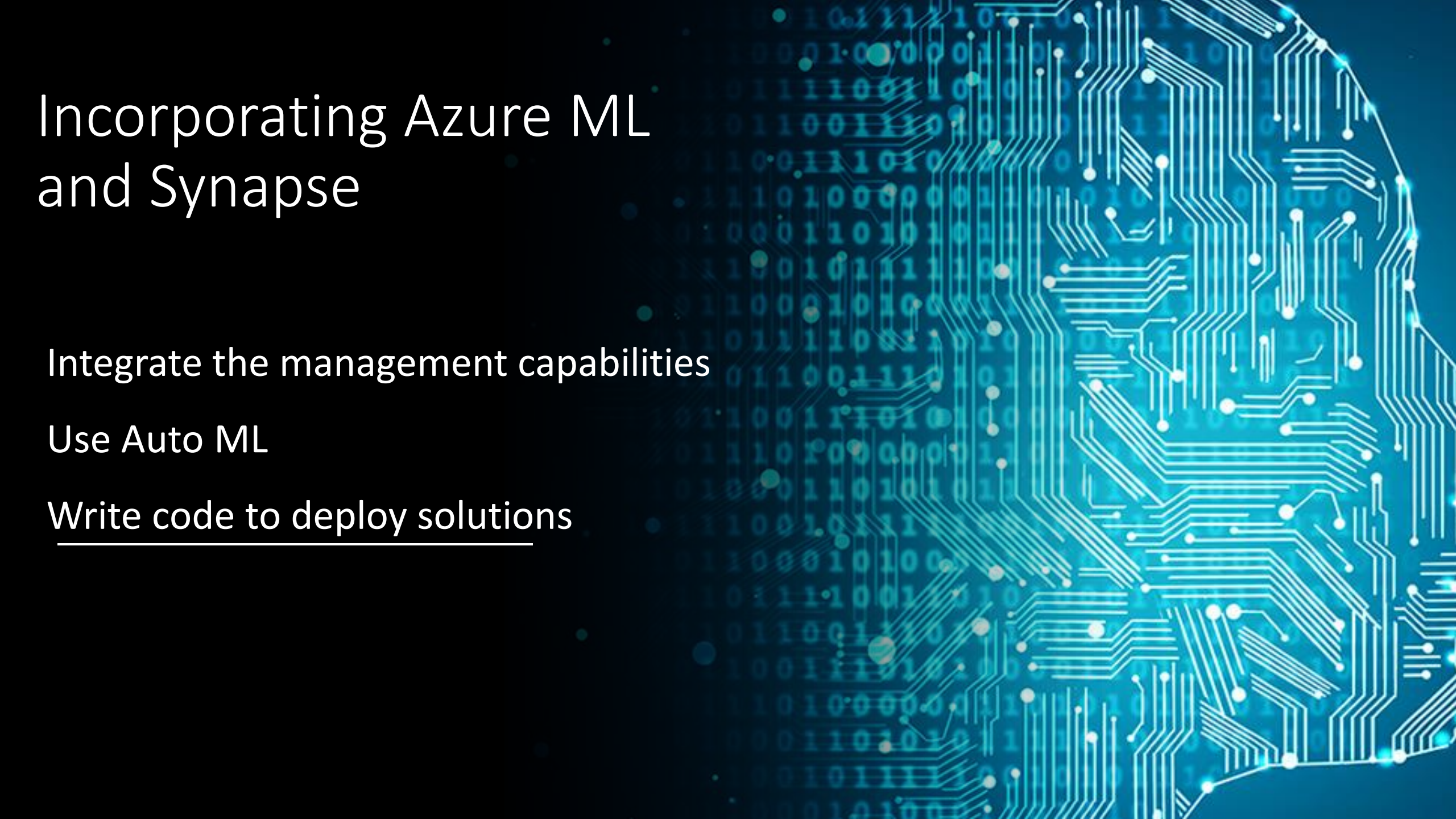
```
Select * from table where column <= 5
```

Incorporating Azure ML and Synapse

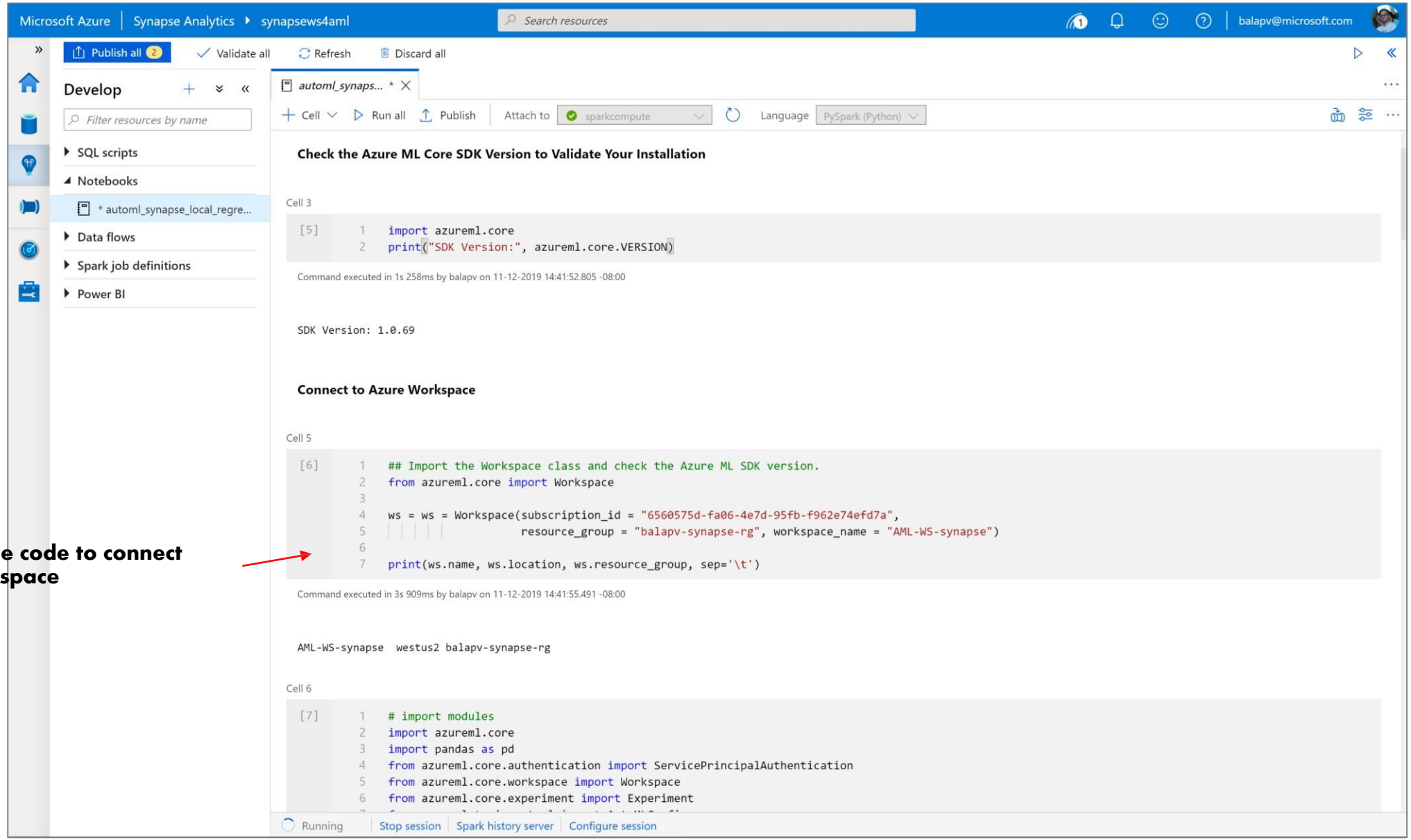
Integrate the management capabilities

Use Auto ML

Write code to deploy solutions

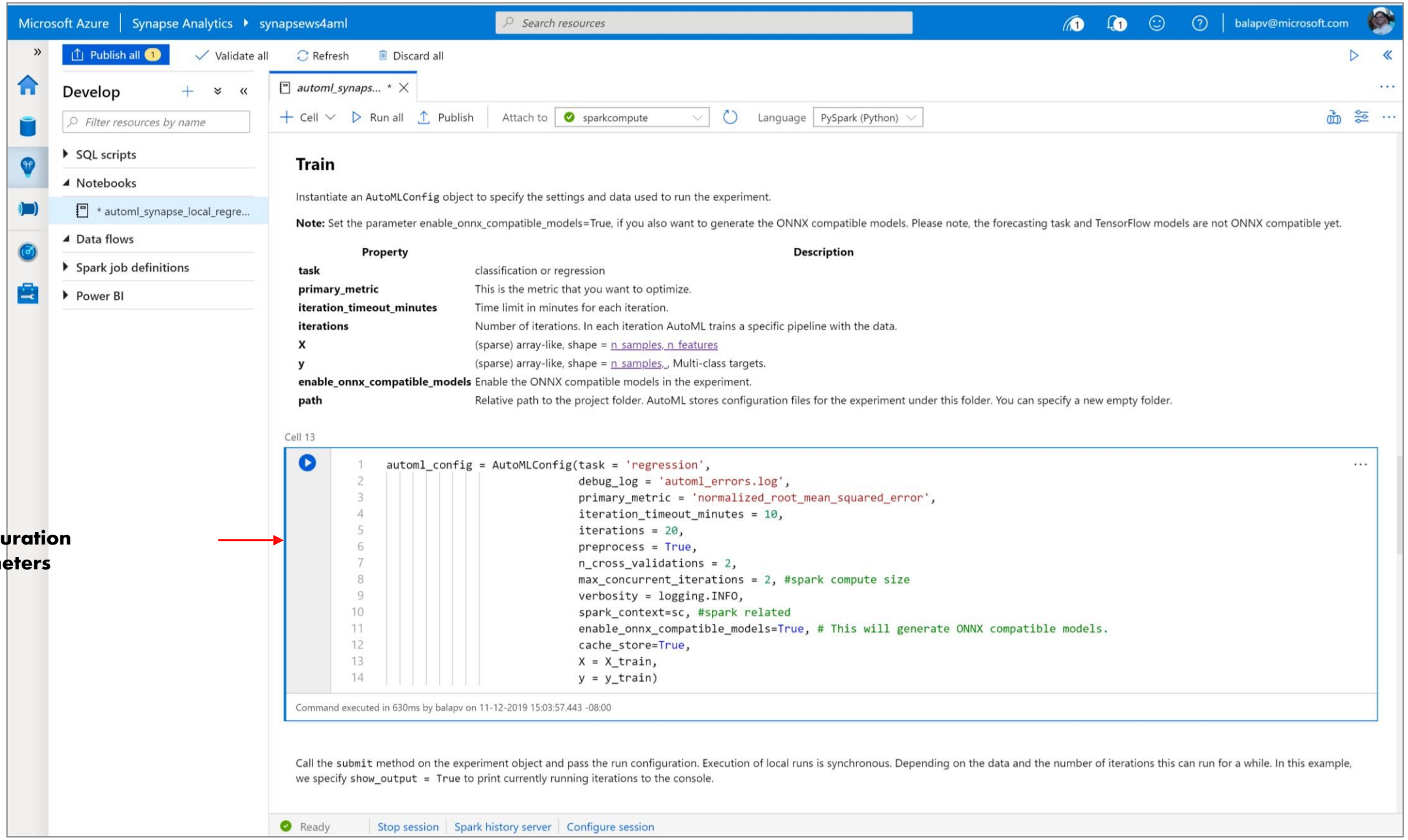


Synapse Notebook: Connect to AML workspace



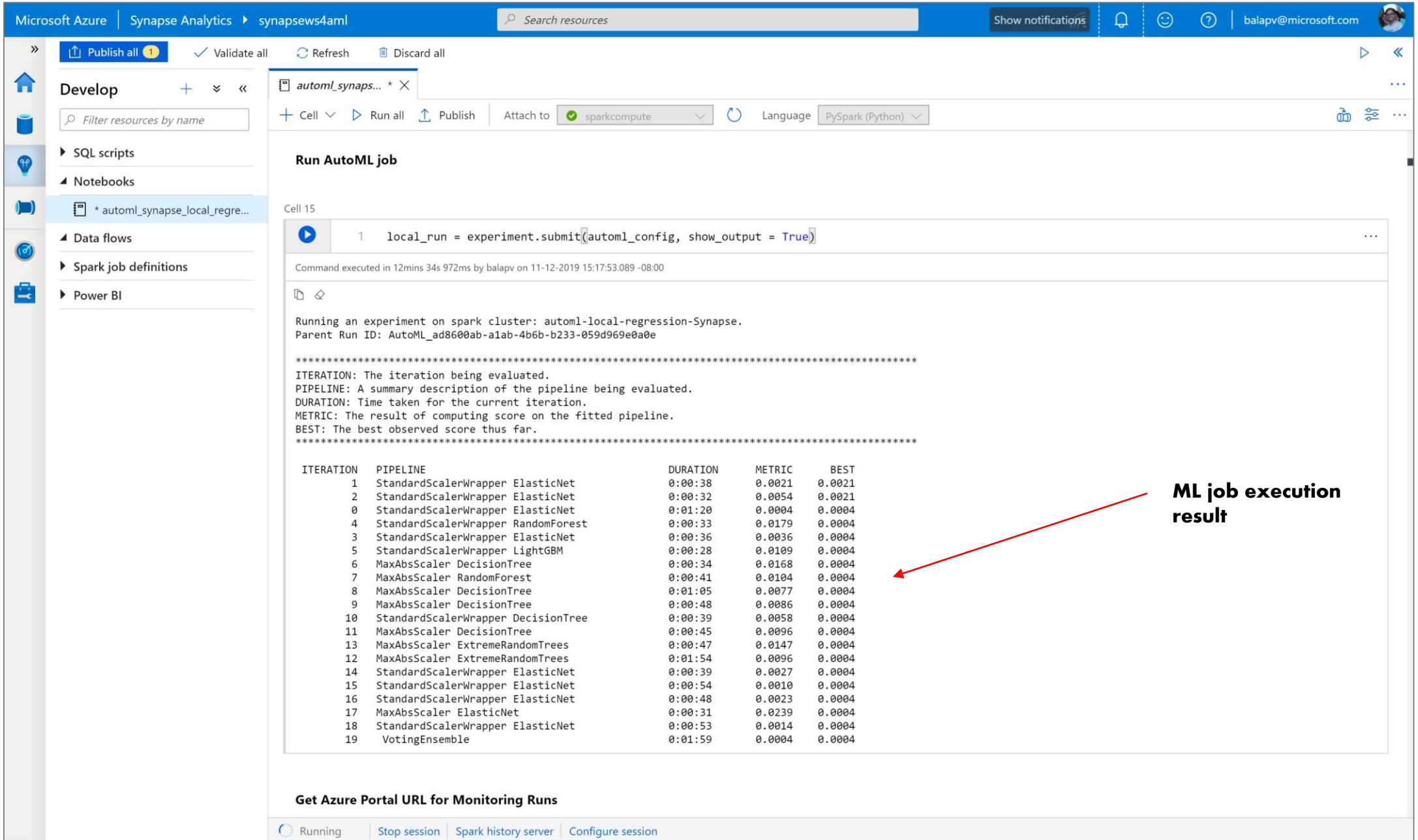
Simple code to connect workspace

Synapse Notebook: Configure AML job to run on Synapse



Configuration parameters

Synapse Notebook: Run AML job



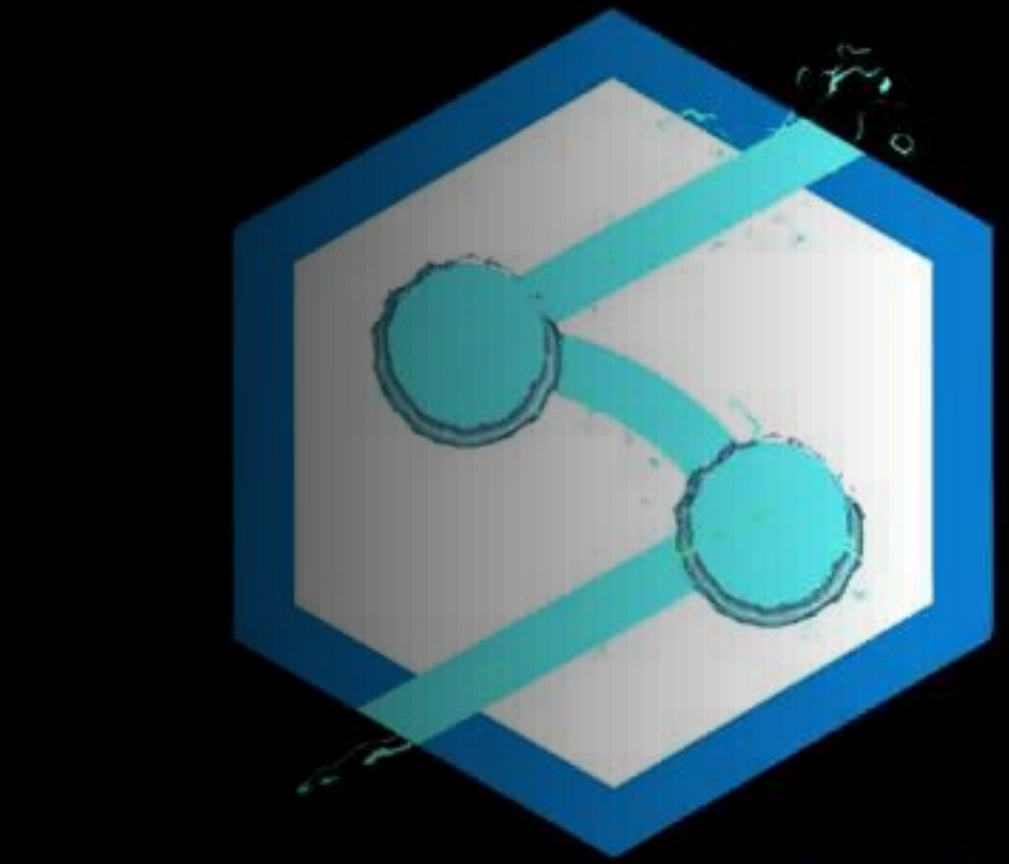
Demonstration

Machine Learning in Synapse

Quiz!

Poll Question

https://geni.us/SynapsePrecon3



Azure Synapse Analytics

Databricks differences

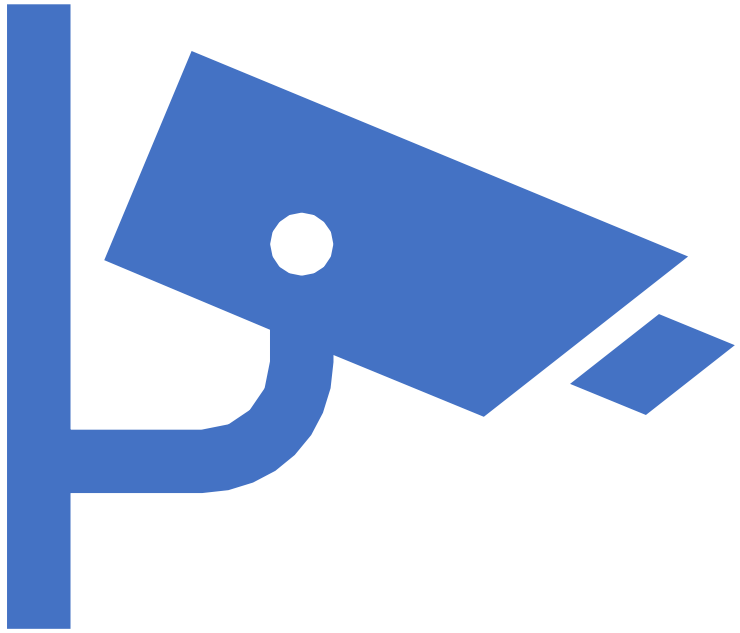
PROPRIETARY VERSION OF SPARK

OPTIMIZED FOR SPEED

SOURCE CONTROL

CAN WRITE SPARK R CODE

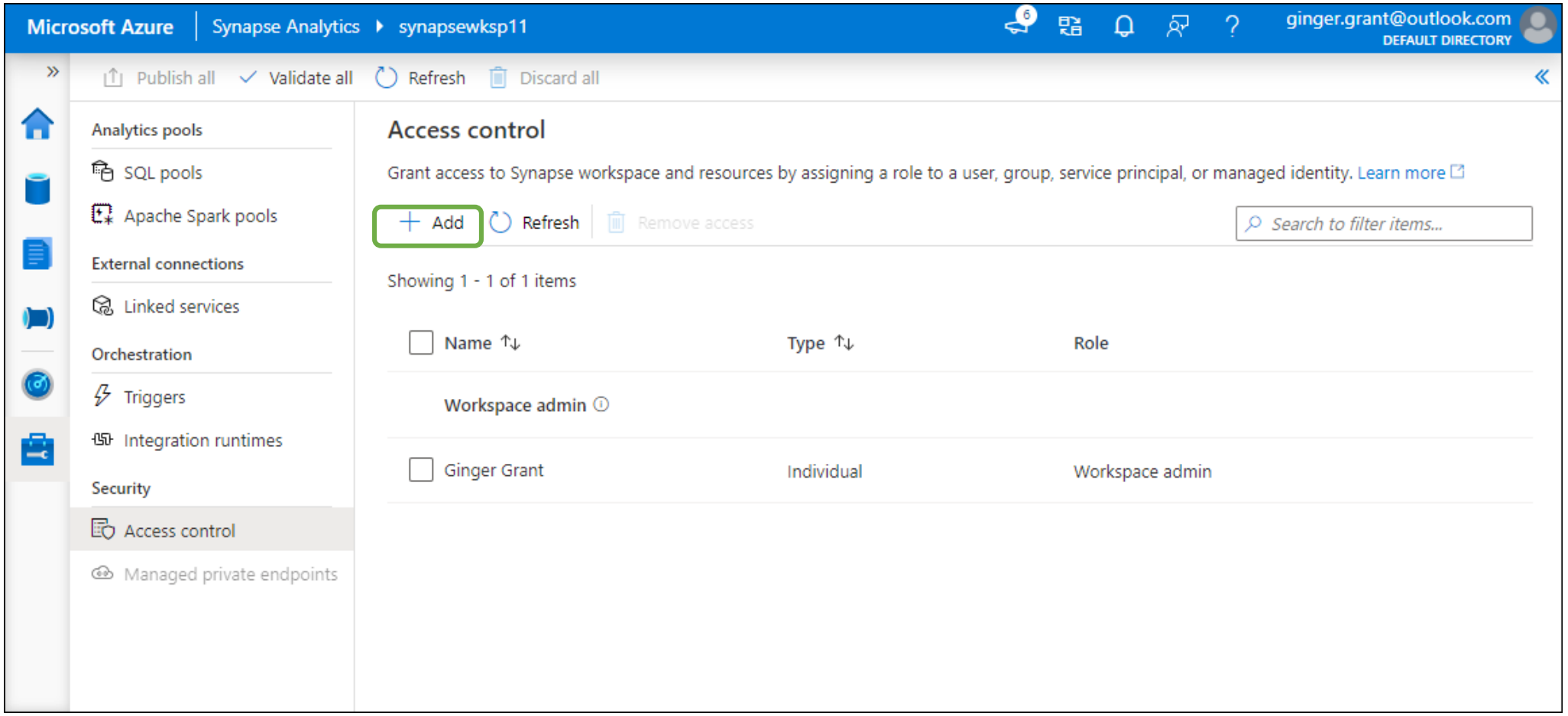
FOR MORE DETAILS, COME TO MY SESSION TOMORROW



Azure Synapse

Security

Azure Synapse Workspace Security



Enterprise-grade security

VNet

Azure Synapse Analytics

Data Protection

Access Control

Authentication

Network Security

Threat Protection

Defense-in-Depth

Industry-leading compliance

ISO 27001

SOC 1 Type 2

SOC 2 Type 2

PCI DSS Level 1

Cloud Controls Matrix

ISO 27018

Content Delivery and Security Association

Shared Assessments

FedRAMP JAB P-ATO

HIPAA / HITECH

FIPS 140-2

21 CFR Part 11

FERPA

DISA Level 2

CJIS

IRS 1075

ITAR-ready

Section 508 VPAT

European Union Model Clauses

EU Safe Harbor

United Kingdom G-Cloud

China Multi Layer Protection Scheme

China GB 18030

China CCCPPF

Singapore MTCS Level 3

Australian Signals Directorate

New Zealand GCIO

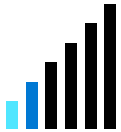
Japan Financial Services

ENISA IAF

Threat Protection - Business requirements

How do we enumerate and track potential SQL vulnerabilities?

To mitigate any security misconfigurations before they become a serious issue.



How do we discover and alert on suspicious database activity?

To detect and resolve any data exfiltration or SQL injection attacks.

Customer Data

Data Protection

Access Control

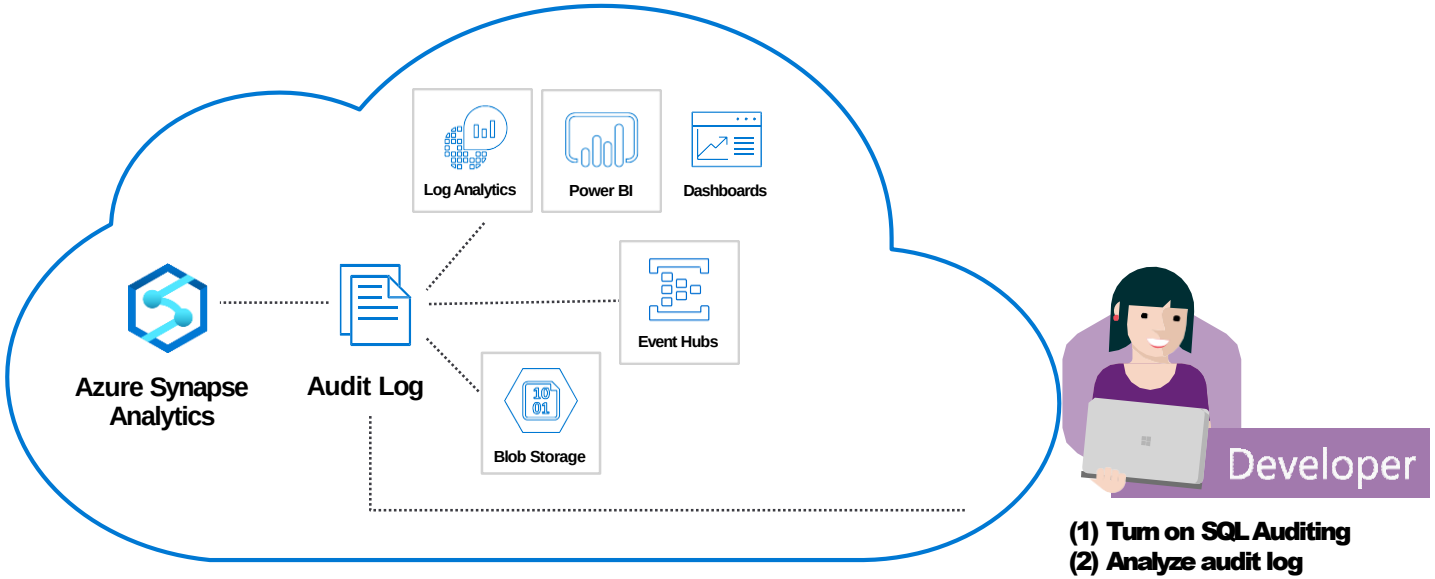
Authentication

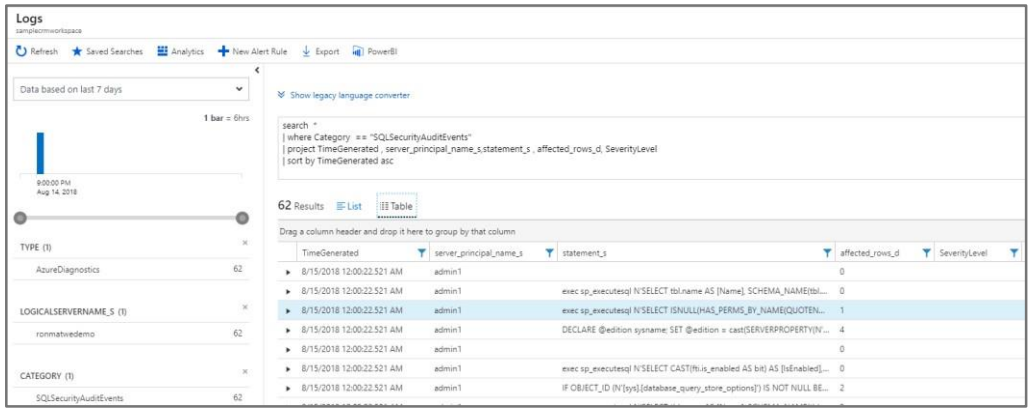
Network Security

Threat Protection

SQL Auditing in Azure Log Analytics and EventHubs

Gain insight into database audit log

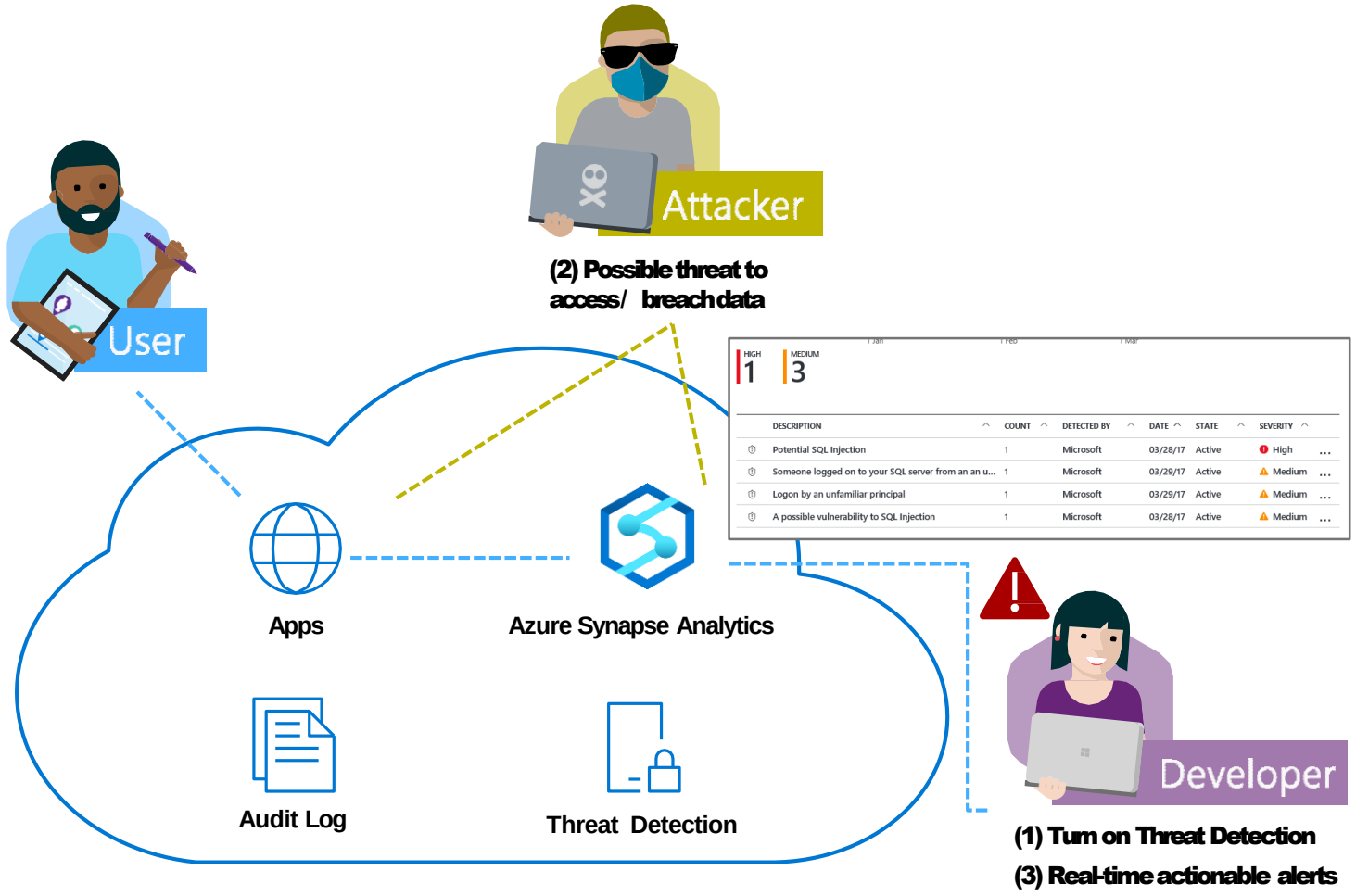




- ✓ **Configurable via audit policy**
- ✓ **SQL audit logs can reside in**
  - **Azure Storage account**
  - **Azure Log Analytics**
  - **Azure Event Hubs**
- ✓ **Rich set of tools for**
  - **Investigating security alerts**
  - **Tracking access to sensitive data**

SQL threat detection

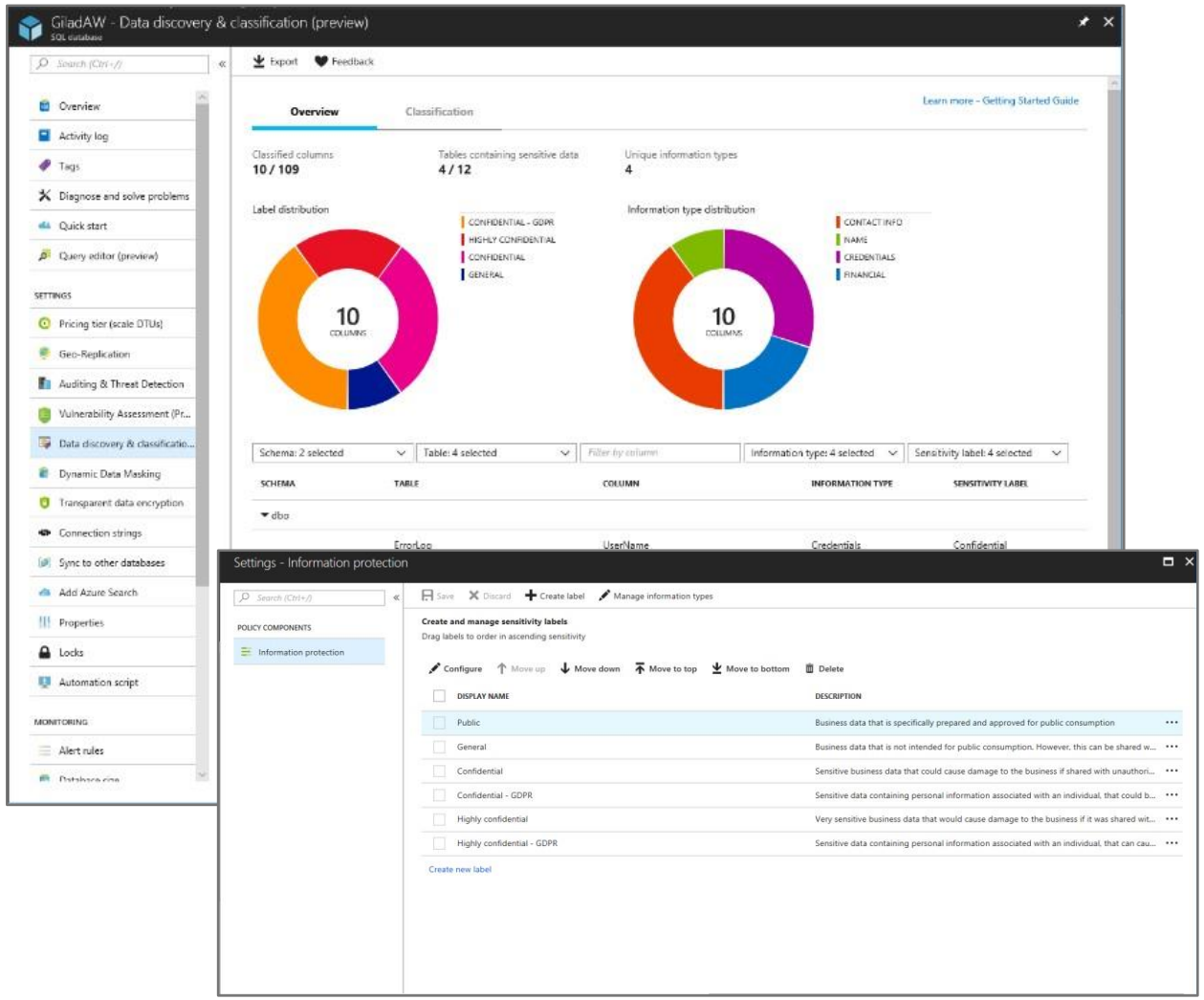
Detect and investigate anomalous database activity



- ✓ **Detects potential SQL injection attacks**
- ✓ **Detects unusual access & data exfiltration activities**
- ✓ **Actionable alerts to investigate & remediate**
- ✓ **View alerts for your entire Azure tenant using Azure Security Center**

SQL Data Discovery & Classification

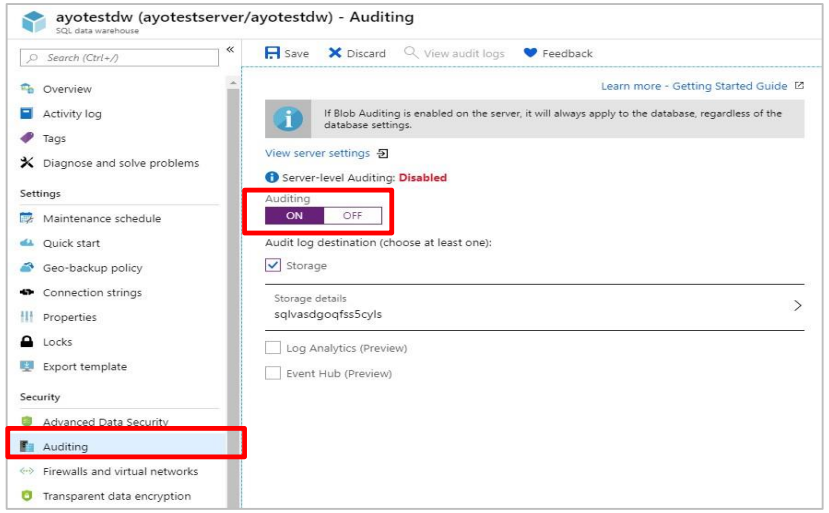
Discover, classify, protect and track access to sensitive data



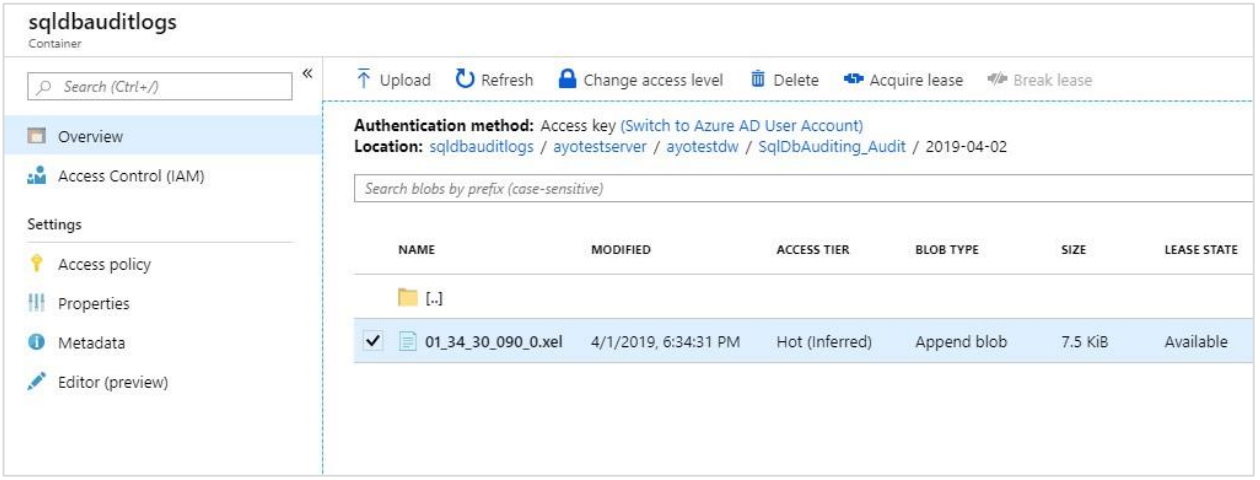
- ✓ **Automatic discovery of columns with sensitive data**
- ✓ **Add persistent sensitive data labels**
- ✓ **Audit and detect access to the sensitive data**
- ✓ **Manage labels for your entire Azure tenant using Azure Security Center**

SQL Data Discovery & Classification – audit sensitive data access

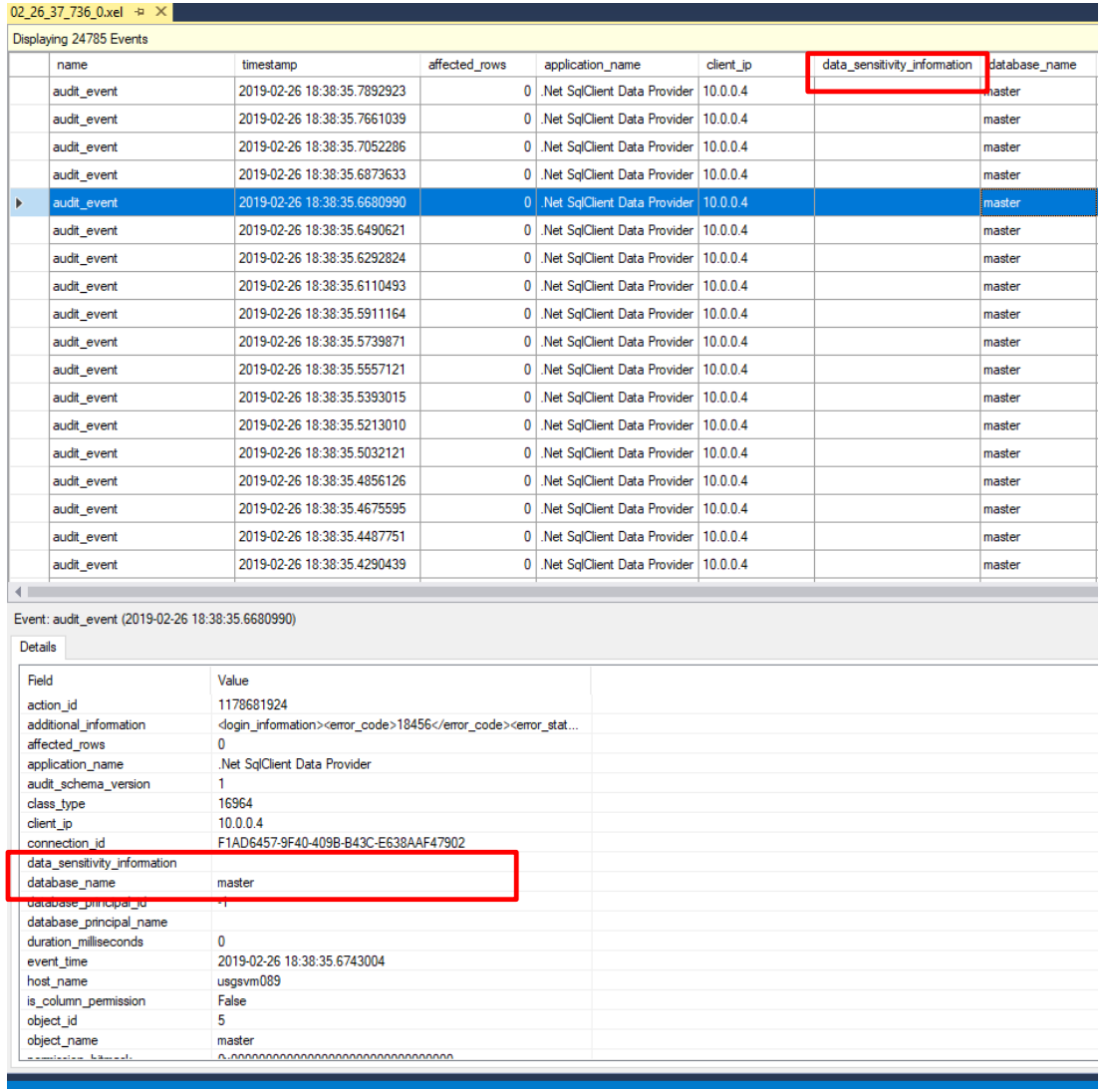
Step 1: Configure auditing for your target Data warehouse. This can be configured for just a single data warehouse or all databases on a server.



Step 2: Navigate to audit logs in storage account and download 'xel' log files to local machine.



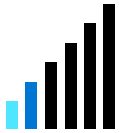
Step 3: Open logs using extended events viewer in SSMS. Configure viewer to include 'data_sensitivity_information' column



Network Security - Business requirements

How do we implement network isolation?

Data at different levels of security needs to be accessed from different locations.



How do we achieve separation?

Disallowing access to entities outside the company's network security boundary.

Customer Data

Data Protection

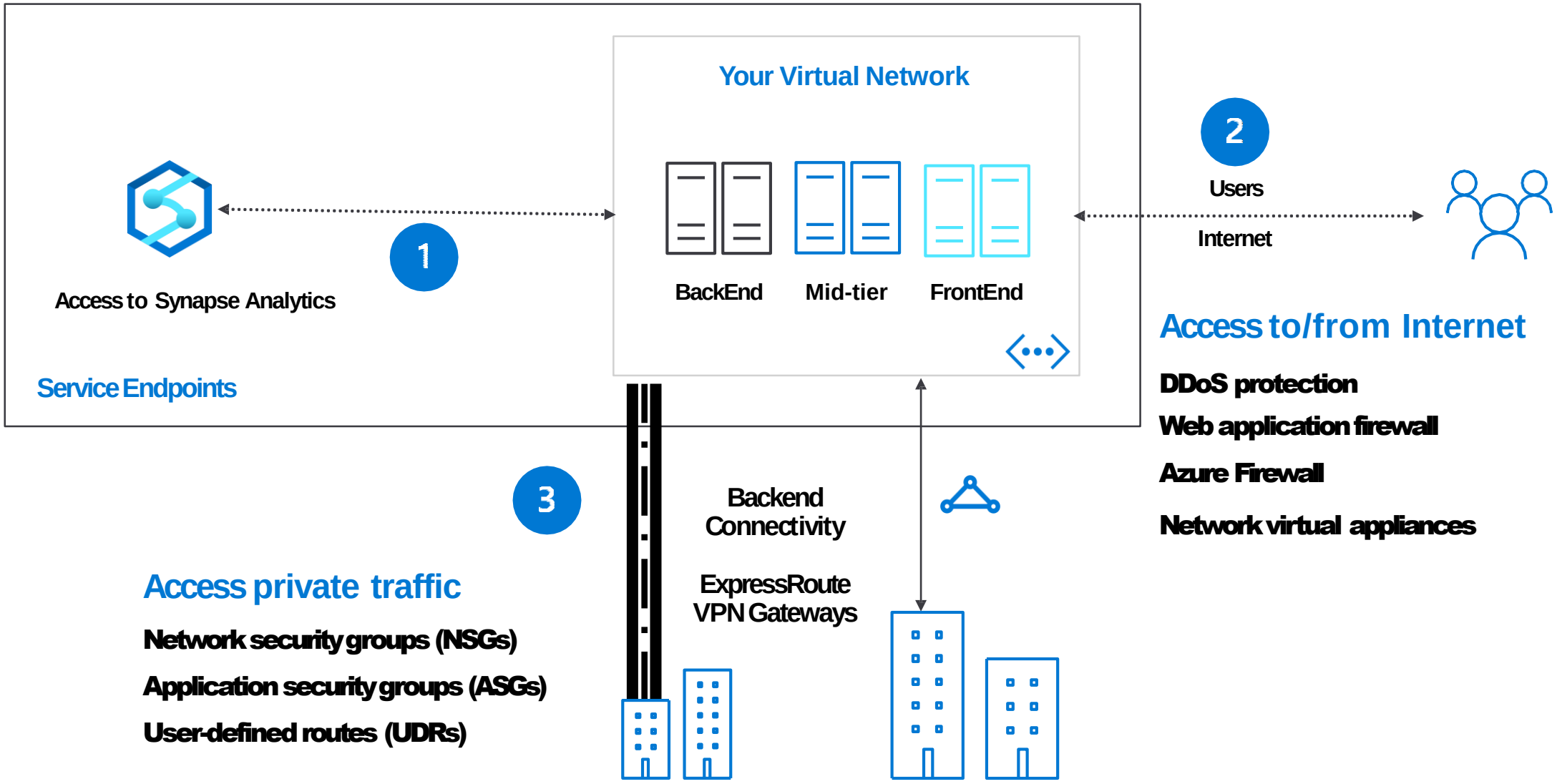
Access Control

Authentication

Network Security

Threat Protection

Azure networking: application-access patterns



Securing with firewalls

Overview

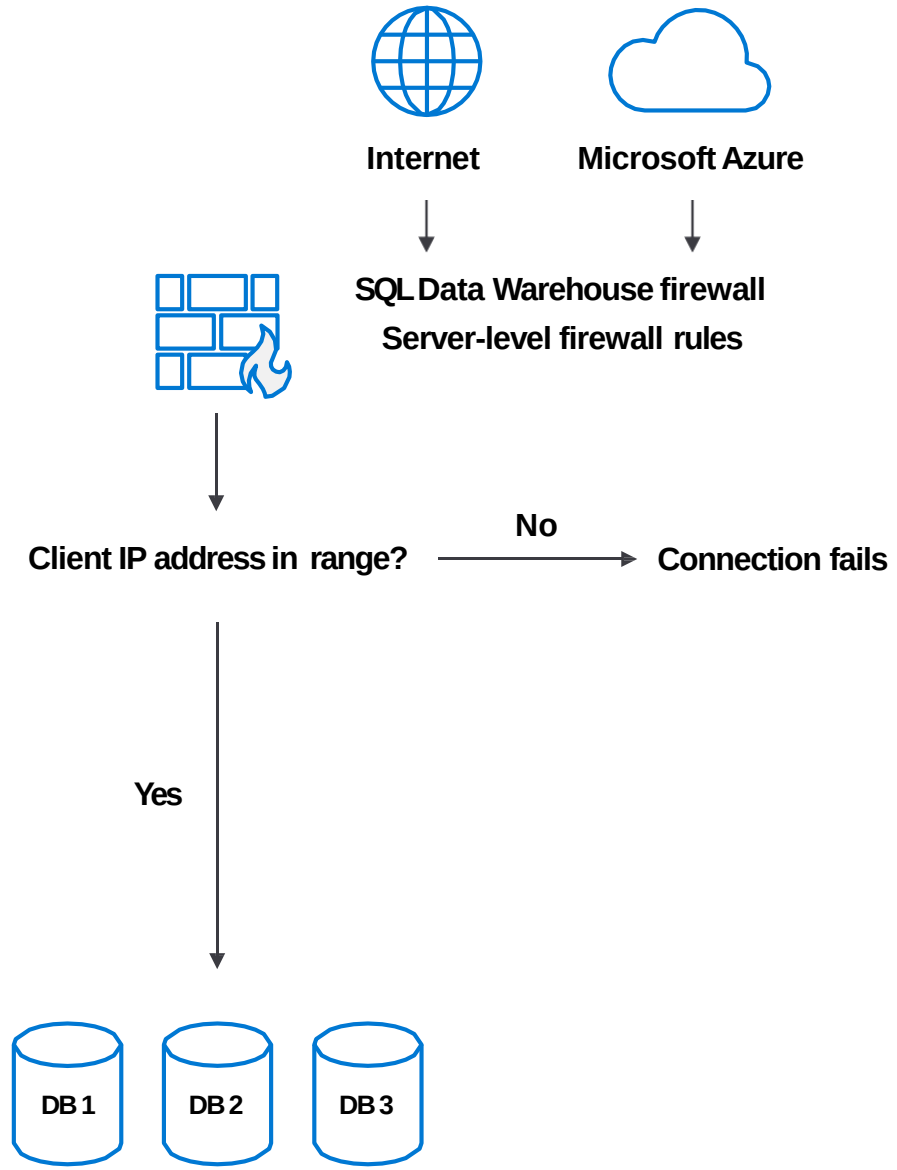
By default, all access to your Azure Synapse Analytics is blocked by the firewall.

Firewall also manages virtual network rules that are based on virtual network service endpoints.

Rules

Allow specific or range of whitelisted IP addresses.

Allow Azure applications to connect.

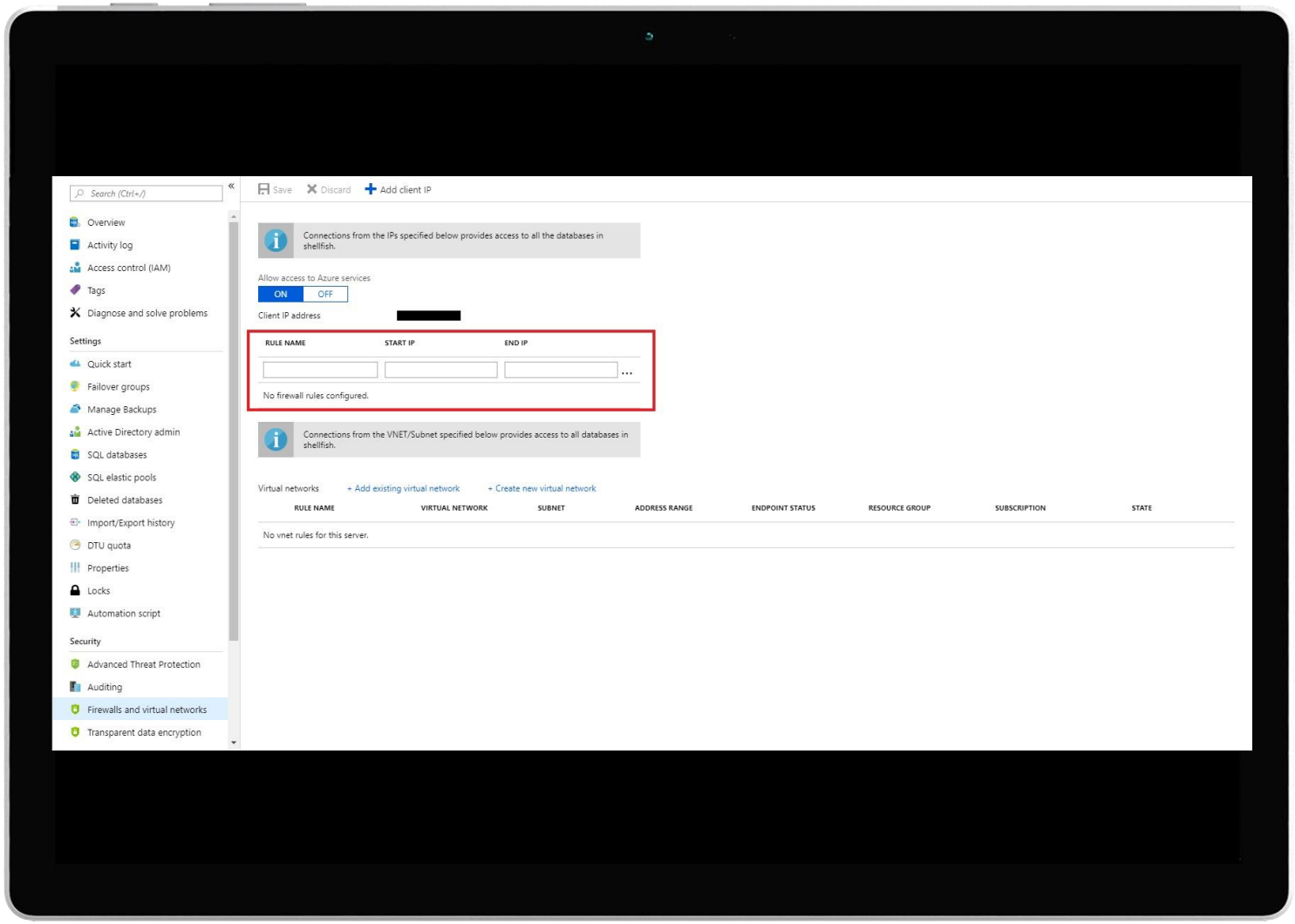


Firewall configuration on the portal

By default, Azure blocks all external connections to port 1433

Configure with the following steps:

Azure Synapse Analytics Resource:
Server name > Firewalls and virtual networks



Firewall configuration using REST API

Managing firewall rules through REST API must be authenticated.

For information, see Authenticating Service Management Requests.

Server-level rules can be created, updated, or deleted using REST API.

To create or update a server-level firewall rule, execute the PUT method.

To remove an existing server-level firewall rule, execute the DELETE method.

To list firewall rules, execute the GET.

```
PUT
https://management.azure.com/subscriptions/{subscriptionId}/resourceGroups/{resourceGroupName}/providers/Microsoft.Sql/servers/{serverName}/firewallRules/{firewallRuleName}?api-version=2014-04-01REQUEST BODY
{
  "properties": {
    "startIpAddress": "0.0.0.3",
    "endIpAddress": "0.0.0.3"
  }
}

DELETE
https://management.azure.com/subscriptions/{subscriptionId}/resourceGroups/{resourceGroupName}/providers/Microsoft.Sql/servers/{serverName}/firewallRules/{firewallRuleName}?api-version=2014-04-01

GET
https://management.azure.com/subscriptions/{subscriptionId}/resourceGroups/{resourceGroupName}/providers/Microsoft.Sql/servers/{serverName}/firewallRules/{firewallRuleName}?api-version=2014-04-01
```

Firewall configuration using PowerShell/T-SQL

Windows PowerShell Azure cmdlets

New-AzureRmSqlServerFirewallRule

Get-AzureRmSqlServerFirewallRule

Set-AzureRmSqlServerFirewallRule

Transact SQL

sp_set_firewall_rule

sp_delete_firewall_rule

```
# PSAllow external IP access to SQLDWPS
C:\> New-AzureRmSqlServerFirewallRule
            -ResourceGroupName "myResourceGroup" `
            -ServerName $servername `
            -FirewallRuleName "AllowSome"
            -StartIpAddress "0.0.0.0"
            -EndIpAddress "0.0.0.0“

-- T-SQLAllow external IP access to SQLDW
EXECUTE sp_set_firewall_rule
            @name= N'ContosoFirewallRule’,
            @start_ip_address = '192.168.1.1’,
            @end_ip_address = '192.168.1.10'
```

VNET configuration on Azure portal

Configure with the following steps:

Azure Synapse Analytics Resource:

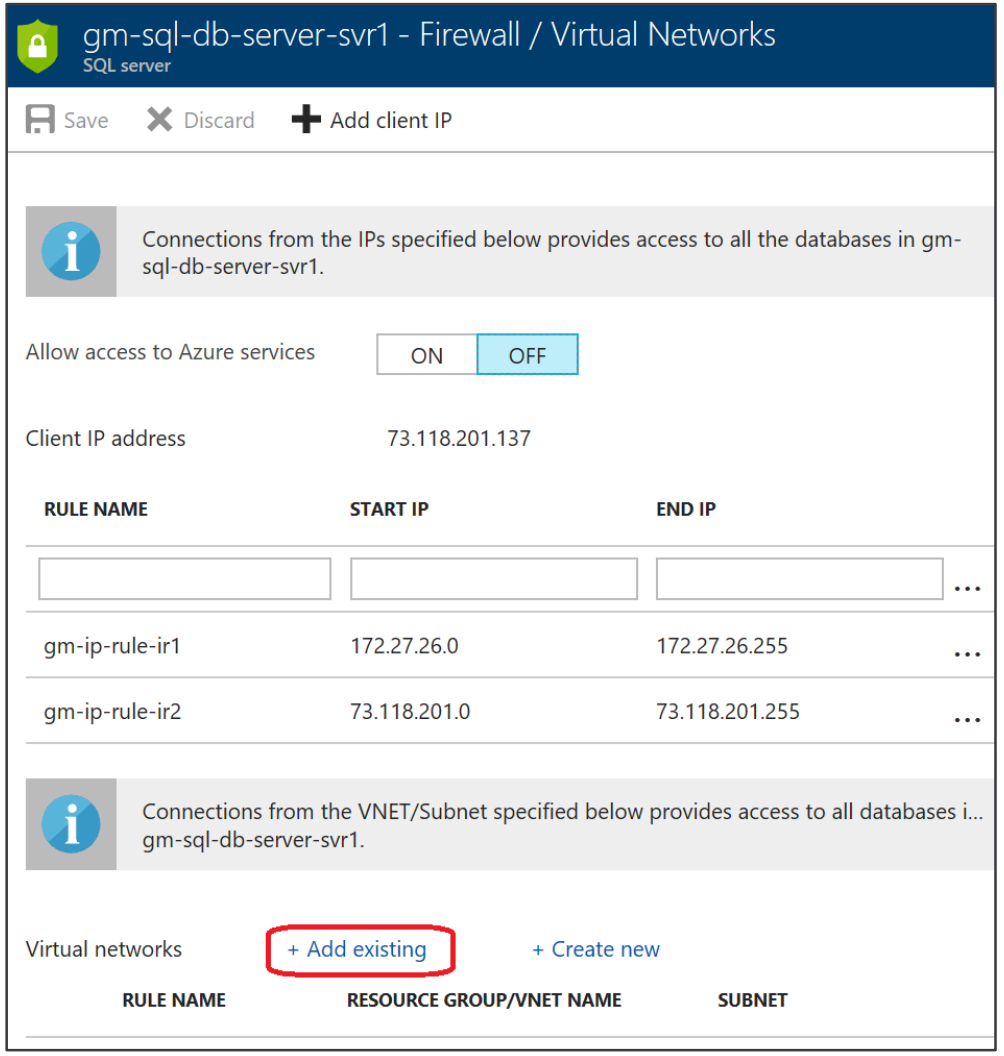
Server name > Firewalls and virtual networks

REST API and PowerShell alternatives available

Note:

By default, VMs on your subnets cannot communicate with your SQL Data Warehouse.

There must first be a virtual network service endpoint for the rule to reference.



Authentication - Business requirements

How do I configure Azure Active Directory with Azure Synapse Analytics?

Azure Active Directory Authenitcation

How do I allow non-Microsoft accounts to be able to authenticate?

Identities, SQL Authentication

Customer Data

Data Protection

Access Control

Authentication

Network Security

Threat Protection

Azure Active Directory authentication

Overview

Manage user identities in one location.

Enable access to Azure Synapse Analytics and other Microsoft services with Azure Active Directory user identities and groups.

Benefits

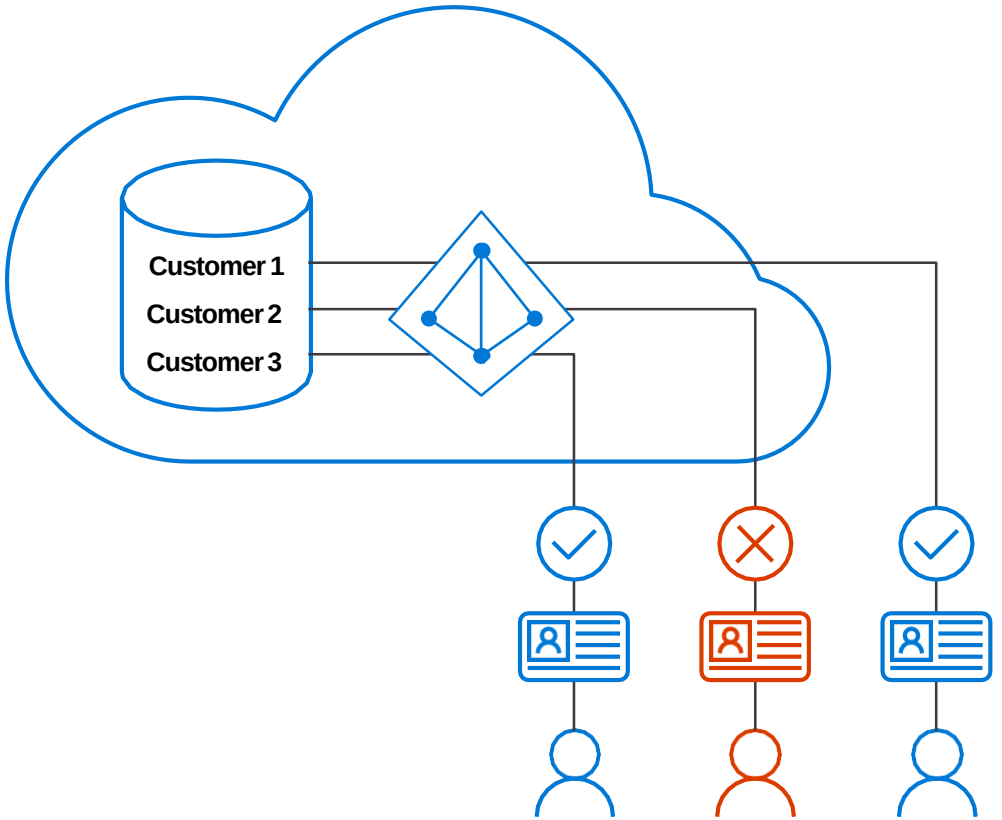
Alternative to SQL Server authentication

Limits proliferation of user identities across databases

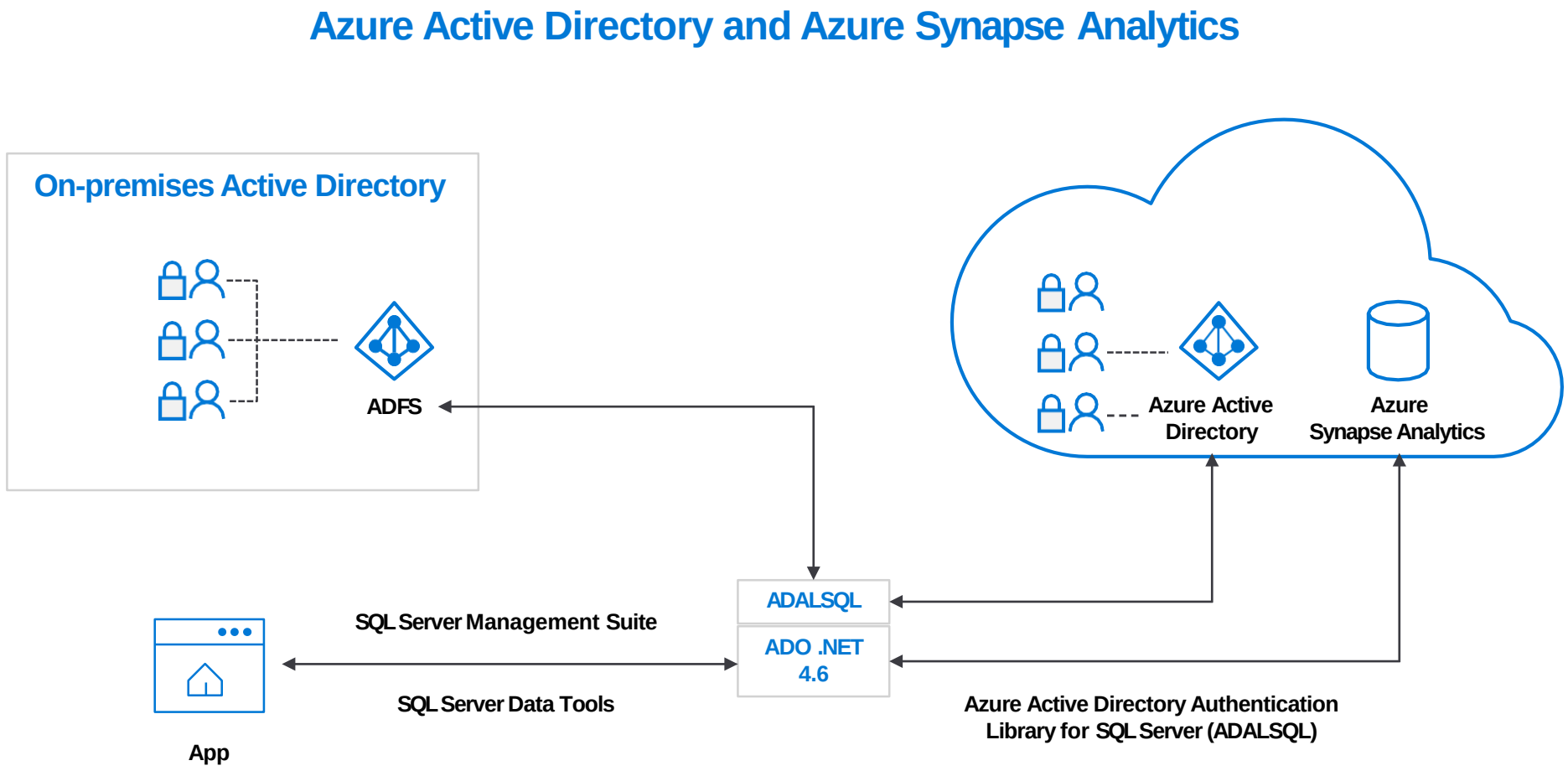
Allows password rotation in a single place

Enables management of database permissions by using external Azure Active Directory groups

Eliminates the need to store passwords



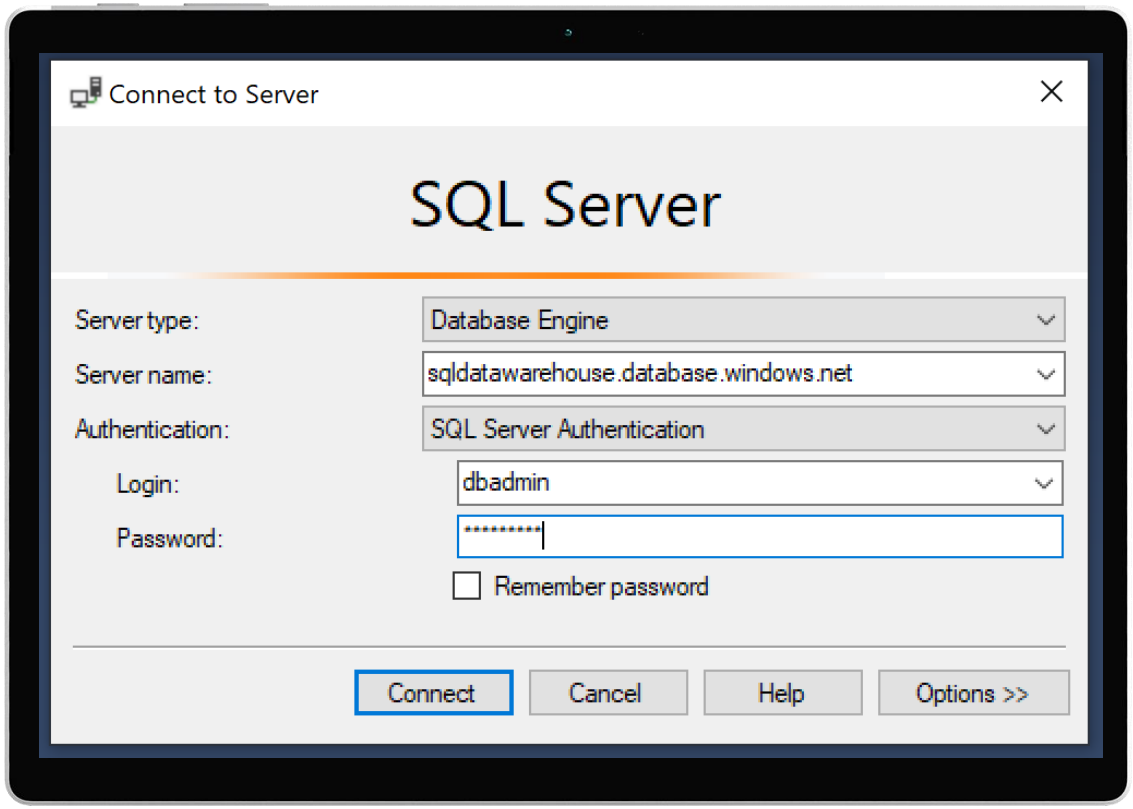
Azure Active Directory trust architecture



SQL authentication

Overview

- This authentication method uses a username and password.
- When you created the logical server for your data warehouse, you specified a "server admin" login with a username and password.
- Using these credentials, you can authenticate to any database on that server as the database owner.
- Furthermore, you can create user logins and roles with familiar SQL Syntax.




```
-- Connect to master database and create a login
CREATE LOGIN ApplicationLogin WITH PASSWORD = 'Str0ng_password';
CREATE USER ApplicationUser FOR LOGIN ApplicationLogin;

-- Connect to SQLDW database and create a database user
CREATE USER DatabaseUser FOR LOGIN ApplicationLogin;
```

Access Control - Business requirements

How do I restrict access to sensitive data to specific database users?

We will take a look at several methods for this

How do I ensure users only have access to relevant data?

For example, in a hospital only medical staff should be allowed to see patient data that is relevant to them—and not every patient's data.

Customer Data

Data Protection

Access Control

Authentication

Network Security

Threat Protection

Object-level security (tables, views, and more)

Overview

GRANT controls permissions on designated tables, views, stored procedures, and functions.

Prevent unauthorized queries against certain tables.

Simplifies design and implementation of security at the database level as opposed to application level.

```
-- Grant SELECT permission to user RosaQdM on table Person.Address in the AdventureWorks2012 database
GRANT SELECT ON OBJECT::Person.Address TO RosaQdM; GO
-- Grant REFERENCES permission on column BusinessEntityID in view HumanResources.vEmployee to user Wanida
GRANT REFERENCES(BusinessEntityID) ON OBJECT::HumanResources.vEmployee to Wanida with GRANT OPTION;
GO
-- Grant EXECUTE permission on stored procedure HumanResources.uspUpdateEmployeeHireInfo to an application role called Recruiting11
USE AdventureWorks2012;
GRANT EXECUTE ON OBJECT::HumanResources.uspUpdateEmployeeHireInfo TO RECRUITING 11; GO
```

Row-level security (RLS)

Overview

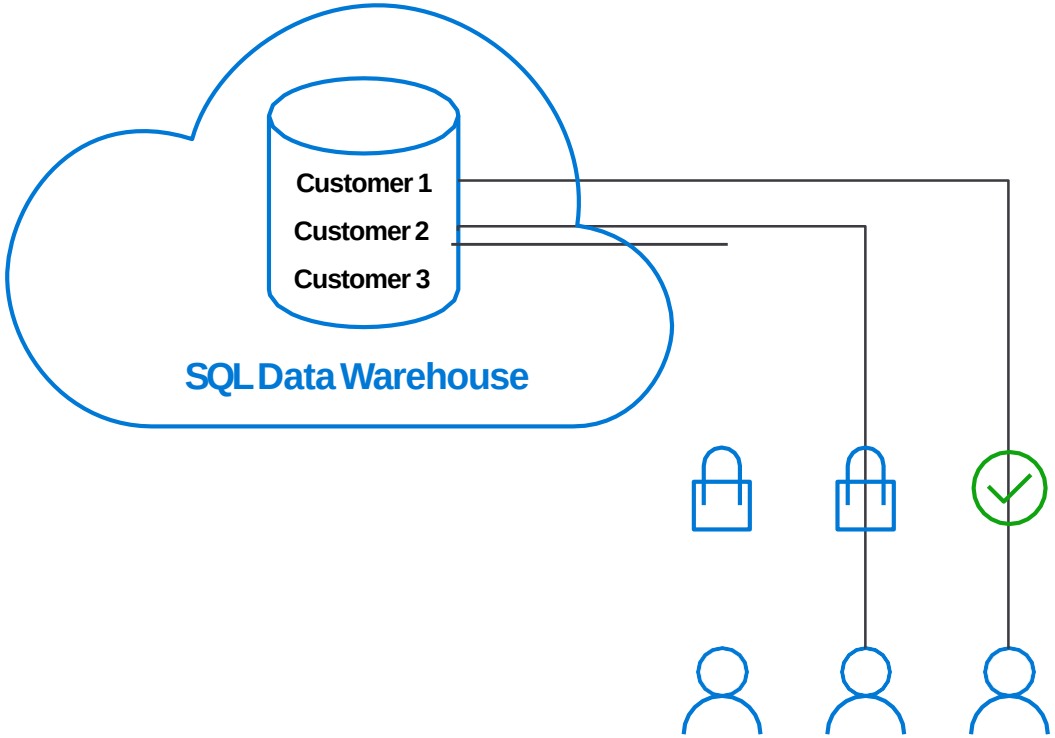
Fine grained access control of specific rows in a database table.

Help prevent unauthorized access when multiple users share the same tables.

Eliminates need to implement connection filtering in multi-tenant applications.

Administer via SQL Server Management Studio or SQL Server Data Tools.

Easily locate enforcement logic inside the database and schema bound to the table.



Row-level security

Creating policies

Filter predicates silently filter the rows available to read operations (SELECT, UPDATE, and DELETE).

The following examples demonstrate the use of the CREATE SECURITY POLICY syntax

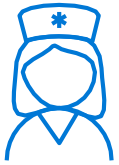
```
-- The following syntax creates a security policy with a filter predicate for the
Customer table
CREATE SECURITY POLICY [FederatedSecurityPolicy]
ADD FILTER PREDICATE [rls].[fn_securitypredicate]([CustomerId])
ON [dbo].[Customer];


-- Create a new schema and predicate function, which will use the application user ID
stored in CONTEXT_INFO to filter rows.
CREATE FUNCTION rls.fn_securitypredicate (@AppUserId int)
RETURNS TABLE
WITH SCHEMABINDING
AS
RETURN (
SELECT 1 AS fn_securitypredicate_result
WHERE
DATABASE_PRINCIPAL_ID() = DATABASE_PRINCIPAL_ID('dbo') -- application context
AND CONTEXT_INFO() = CONVERT(VARBINARY(128), @AppUserId));
GO
```

Row-level security

Three steps:

1. **Policy manager creates filter predicate and security policy in T-SQL, binding the predicate to the patients table.**
2. **App user (e.g., nurse) selects from Patients table.**
3. **Security policy transparently rewrites query to apply filter predicate.**

Policy manager



Nurse

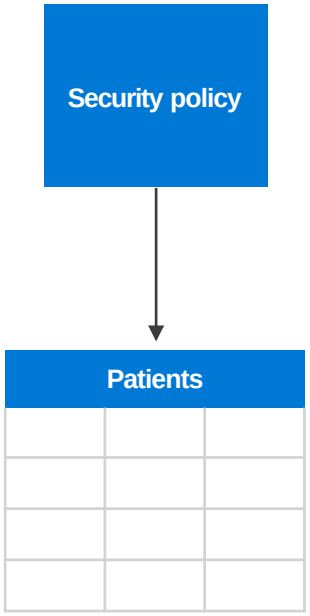
Application

```
SELECT * FROM Patients
```

Database

Security policy

Patients



Filter Predicate: INNER JOIN…

```
CREATE FUNCTION dbo.fn_securitypredicate(@wing int)
    RETURNS TABLE WITH SCHEMABINDING AS
    return SELECT 1 as [fn_securitypredicate_result] FROM
        StaffDuties d INNER JOIN Employees e
        ON(d.EmpId = e.EmpId)
        WHERE e.UserSID = SUSER_SID() AND @wing = d.Wing;

CREATE SECURITY POLICY dbo.SecPol
    ADD FILTER PREDICATE dbo.fn_securitypredicate(Wing) ON Patients
    WITH (STATE = ON)
```

```
SELECT * FROM Patients
    SEMI JOIN APPLY dbo.fn_securitypredicate(patients.Wing);
```

```
SELECT Patients.* FROM Patients,
    StaffDuties d INNER JOIN Employees e ON(d.EmpId = e.EmpId)
    WHERE e.UserSID = SUSER_SID() AND Patients.wing = d.Wing;
```

Column-level security

Overview

Control access of specific columns in a database table based on customer's group membership or execution context.

Simplifies the design and implementation of security by putting restriction logic in database tier as opposed to application tier.

Administer via GRANT T-SQL statement.

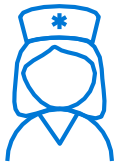
Both Azure Active Directory (AAD) and SQL authentication are supported.

Column-level security

Three steps:

1. **Policy manager creates permission policy in T-SQL, binding the policy to the Patients table on a specific group.**
2. **App user (for example, a nurse) selects from Patients table.**
3. **Permission policy prevents access on sensitive data.**

Policy manager



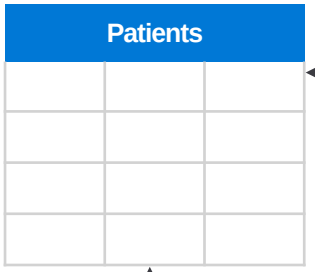
Nurse

Application

```
SELECT * FROM Membership;

Msg 230, Level 14, State 1, Line 12
The SELECT permission was denied on the column
'SSN' of the object 'Membership', database
'CLS_TestDW', schema 'dbo'.
```

Queries executed as 'Nurse' will fail if they include the SSN column

Database



Permission policy

```
CREATE TABLE Patients (
    PatientID int IDENTITY,
    FirstName varchar(100) NULL,
    SSN char(9) NOT NULL,
    LastName varchar(100) NOT NULL,
    Phone varchar(12) NULL,
    Email varchar(100) NULL
);
```

```
GRANT SELECT ON Patients (
    PatientID, FirstName, LastName, Phone, Email
) TO Nurse;
```

Allow 'Nurse' to access all columns except for sensitive SSN column

Data Protection - Business requirements

How do I protect sensitive data against unauthorized (high-privileged) users?

What key management options do I have?

Customer Data

Data Protection

Access Control

Authentication

Network Security

Threat Protection

Dynamic Data Masking

Overview

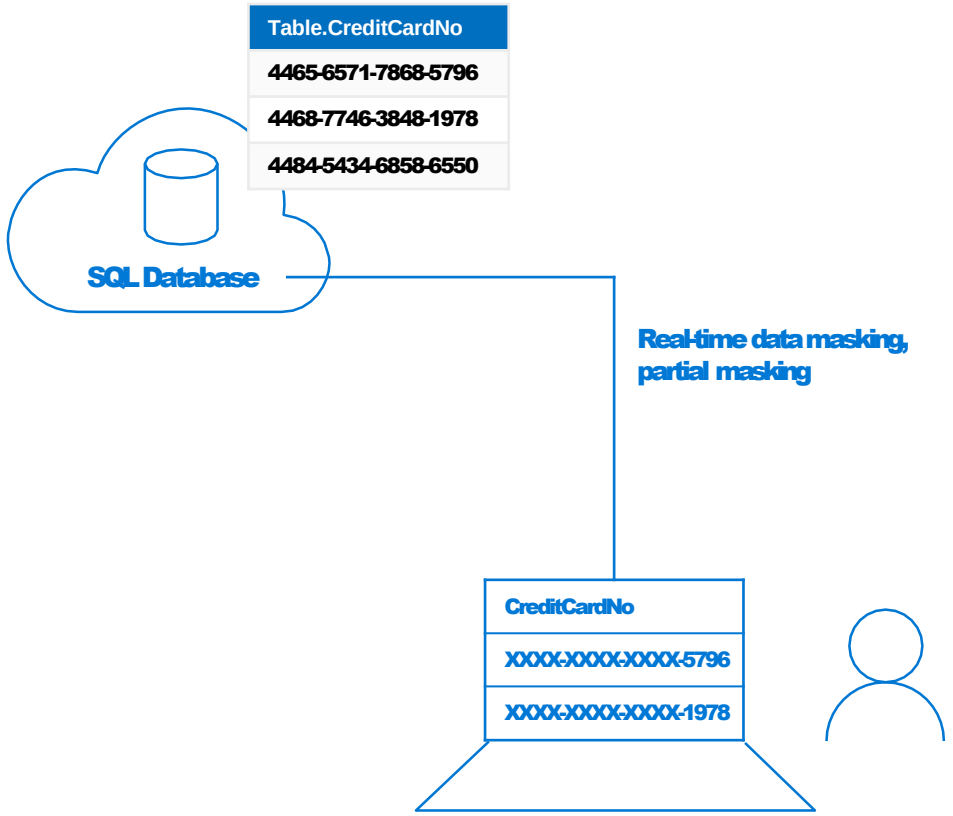
Prevent abuse of sensitive data by hiding it from users

Easy configuration in new Azure Portal

Policy-driven at table and column level, for a defined set of users

Data masking applied in real-time to query results based on policy

Multiple masking functions available, such as full or partial, for various sensitive data categories (credit card numbers, SSN, etc.)



Dynamic Data Masking

Three steps

1.Security officer defines dynamic data masking policy in T-SQL over sensitive data in the Employee table. The security officer uses the built-in masking functions (default, email, random)

2. The app-user selects from the Employee table

3.The dynamic data masking policy obfuscates the sensitive data in the query results for non-privileged users

1

```
ALTER TABLE [Employee]
ALTER COLUMN [SocialSecurityNumber]
ADD MASKED WITH (FUNCTION = 'DEFAULT()')

ALTER TABLE [Employee]
ALTER COLUMN [Email]
ADD MASKED WITH (FUNCTION = 'EMAIL()')

ALTER TABLE [Employee]
ALTER COLUMN [Salary]
ADD MASKED WITH (FUNCTION = 'RANDOM(1,20000)')

GRANT UNMASK to admin1
```

2


```
SELECT [First Name],
       [Social Security Number],
       [Email],
       [Salary]
FROM   [Employee]
```

3

Non-masked data (admin login)

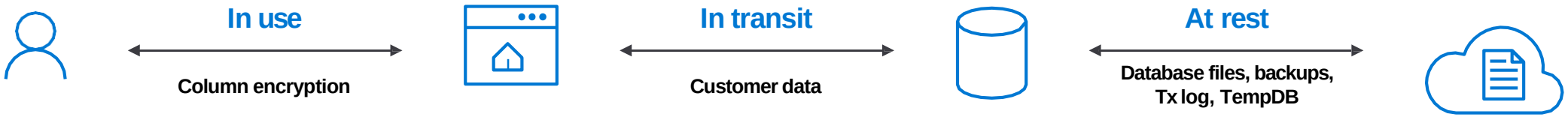
	First Name	Social Security Num...	Email	Salary
1	LILA	758-10-9637	lila.barnett@comcast.net	1012794
2	JAMIE	113-29-4314	jamie.brown@ntlworld.com	1025713
3	SHELLEY	550-72-2028	shelley.lynn@charter.net	1040131
4	MARCELLA	903-94-5665	marcella.estrada@comcast.net	1040753
5	GILBERT	376-79-4787	gilbert.juarez@verizon.net	1041308

Masked data (admin1 login)

	First Name	Social Security Number	Email	Salary
1	LILA	XXX-XX-XX37	lXX@XXXX.net	8940
2	JAMIE	XXX-XX-XX14	jXX@XXXX.com	19582
3	SHELLEY	XXX-XX-XX28	sXX@XXXX.net	3713
4	MARCELLA	XXX-XX-XX65	mXX@XXXX.net	11572
5	GILBERT	XXX-XX-XX87	gXX@XXXX.net	4487

Types of data encryption

Data Encryption	Encryption Technology	Customer Value
In transit	**Transport Layer Security (TLS) from the client to the server** **TLS 1.2**	**Protects data between client and server against snooping and man-in-the-middle attacks**
At rest	**Transparent Data Encryption (TDE) for Azure Synapse Analytics**	**Protects data on the disk** **User or Service Managed key management is handled by Azure, which makes it easier to obtain compliance**



Transparent data encryption (TDE)

- **Overview**

- **All customer data encrypted at rest**

- **TDE performs real-time I/O encryption and decryption of the data and log files.**

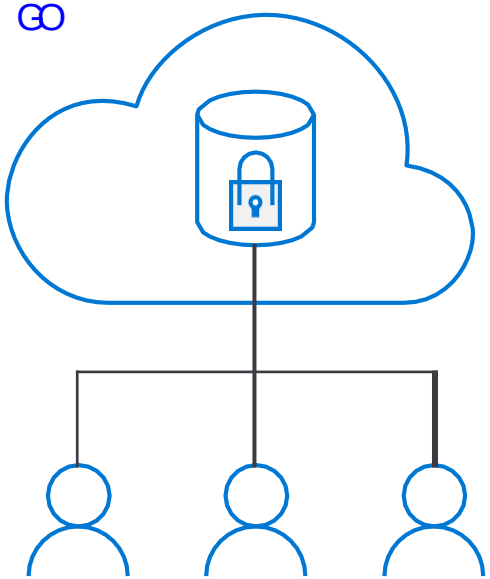
- **Service OR User managed keys.**

- **Application changes kept to a minimum.**

- **Transparent encryption/decryption of data in a TDE-enabled client driver.**

- **Compliant with many laws, regulations, and guidelines established across various industries.**

```
USE master;
GO
CREATE MASTER KEY ENCRYPTION BY PASSWORD = '<UseStrongPasswordHere>';
go
CREATE CERTIFICATE MyServerCert WITH SUBJECT = 'My DEK Certificate';  go
USE MyDatabase;
GO
CREATE DATABASE ENCRYPTION KEY
WITH ALGORITHM = AES_128
ENCRYPTION BY SERVER CERTIFICATE MyServerCert;  GO
ALTER DATABASE MyDatabase
SET ENCRYPTION ON;


GO
```



Transparent data encryption (TDE)

Key Vault

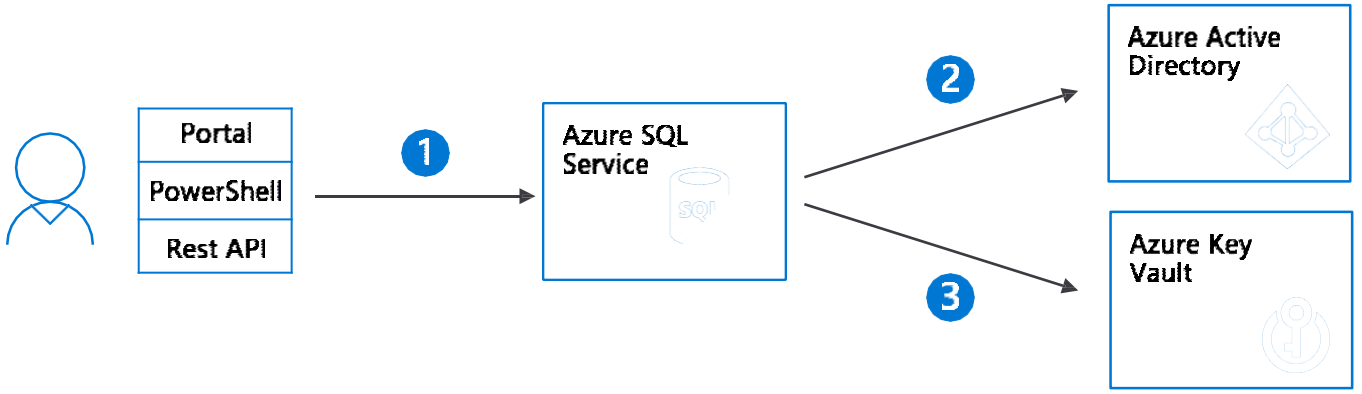
Benefits with User Managed Keys

Assume more control over who has access to your data and when.

Highly available and scalable cloud-based key store.

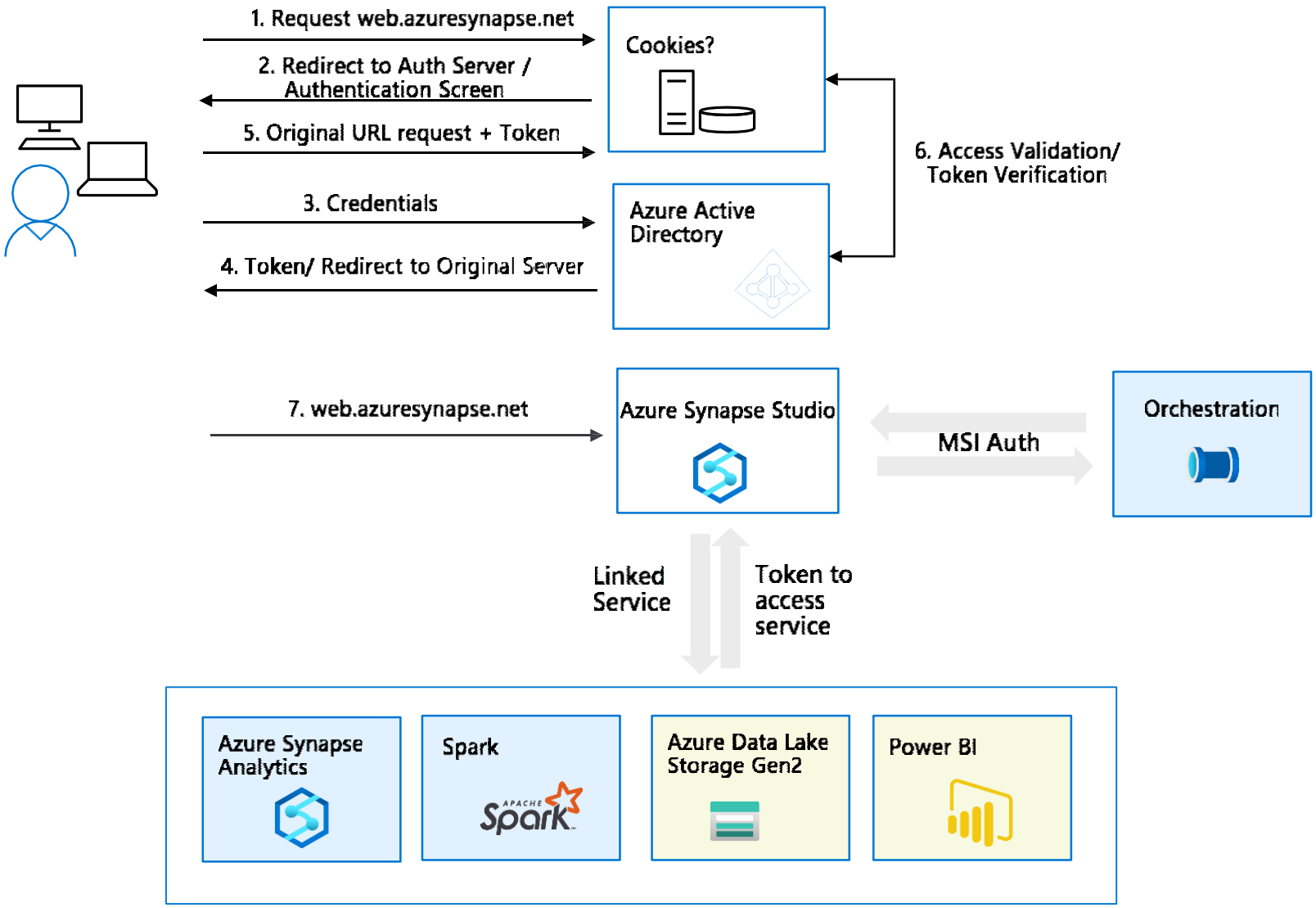
Central key management that allows separation of key management and data.

Configurable via Azure Portal, PowerShell, and REST API.



1. The Key Vault admin grants vault access to the SQL Database server using its unique Azure Active Directory (AD) identity
2. The server uses its Azure AD identity to authenticate with Azure AD for access to your Key Vault
3. The server sends get, wrap key, and unwrap key request to the asymmetric key in key Vault for database encryption key protection.

Single Sign-On



Implicit authentication - **Userprovides login credentials once to accessAzure Synapse Workspace**

AAD authentication - **Azure Synapse Studio will request token to access each linked services as user. A separate token is acquired for each of the below services:**

1. **ADLS Gen2**
2. **Azure Synapse Analytics**
3. **Power BI**
4. **Spark – Spark Livy API**
5. **management.azure.com – resource provisioning**
6. **Develop artifacts – dev.workspace.net**
7. **Graph endpoints**

MSI authentication - **Orchestration uses MSI auth for automation**

Summary

- Introduction to Azure Synapse
- Different Pools
- Data Lakehouse
- Power BI Integration
- Orchestration
- Data Warehousing
- Spark and Machine Learning
- Security

Thanks for the help

Rie Irish

Kaiser Larsen

Euan Garden

Matt Usher

Questions?

www.desertislesql.com