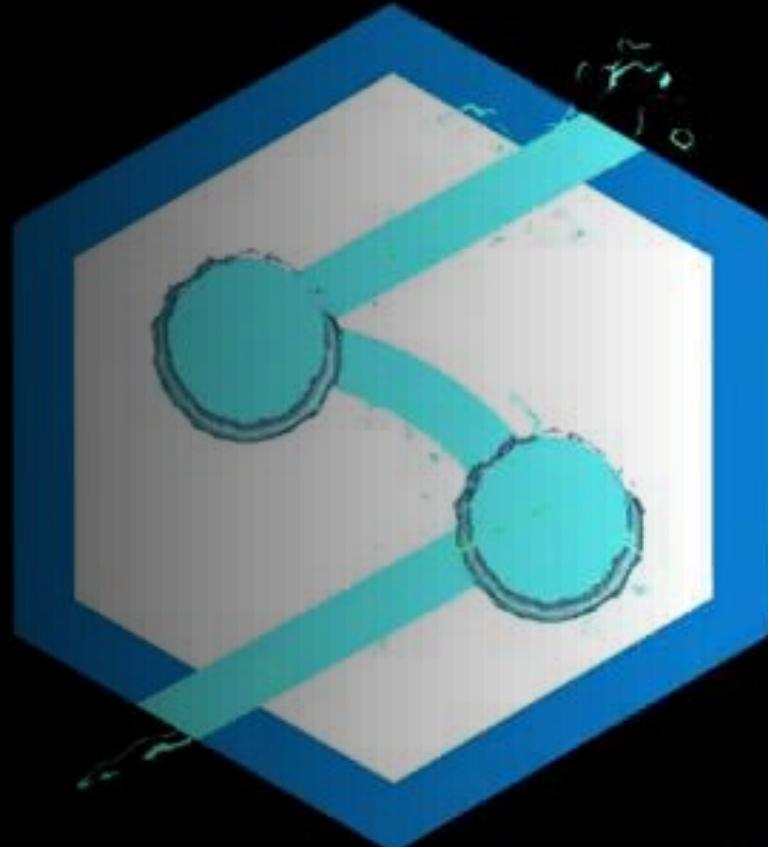


Modern Data Warehousing and Advanced Analysis with Azure Synapse

Poll Question

<https://geni.us/SynapsePrecon1>

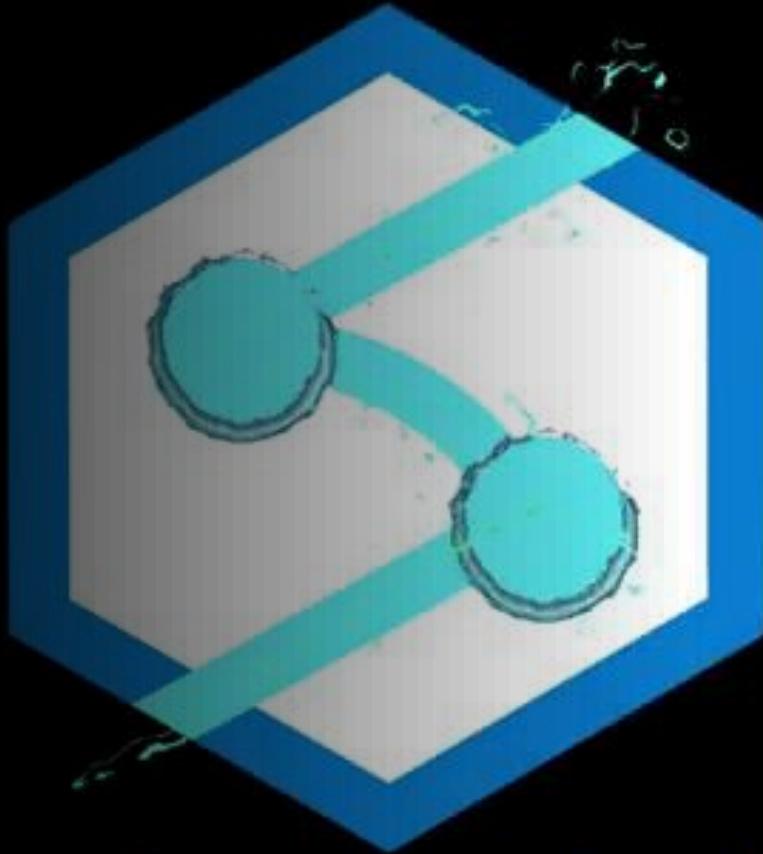


Azure Synapse Analytics

Modern Data Warehousing and Advanced Analysis with Azure Synapse

Ginger Grant

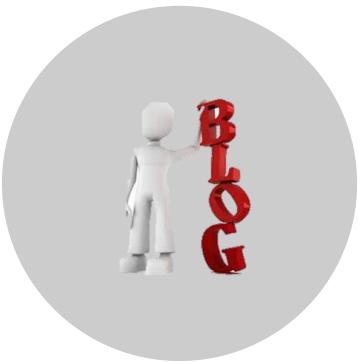
November 10, 2020



Azure Synapse Analytics

About Me – Ginger Grant

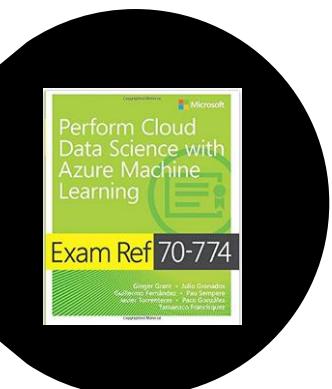
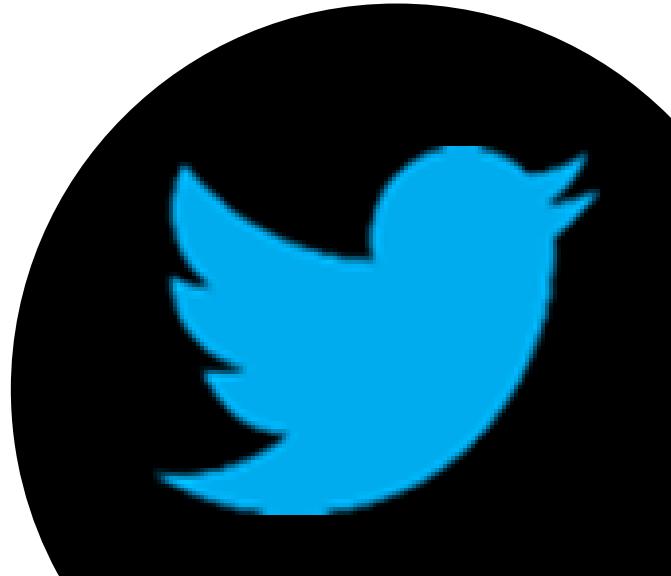
www.desertislesql.com



ginger.grant@DesertIsleSQL.com



@DesertIsleSql



Agenda

Introduction to Azure Synapse

SQL Serverless

Data Lakehouse

Power BI Integration

Orchestration

Data Warehousing

Spark and Machine Learning

Security

What is Synapse?



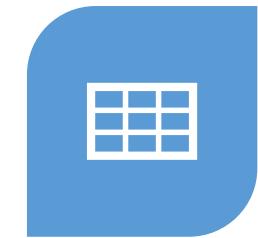
NEW VERSION
OF SQL DW



ANOTHER PLACE TO
DO ADF



SPARK
IMPLEMENTATION

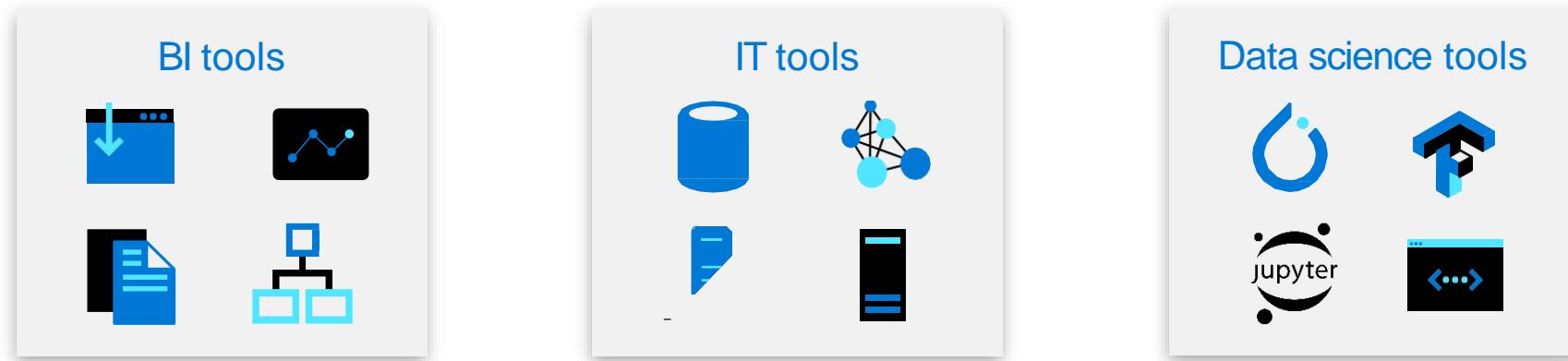


PLACE TO WRITE SQL
AGAINST BLOB
STORAGE



ANOTHER WORD
PEOPLE SAY
DIFFERENTLY IN
DIFFERENT COUNTRIES

Data silos and incompatible tooling inhibit collaboration



Incompatible tooling

Siloed data



Marketing



Sales



Product

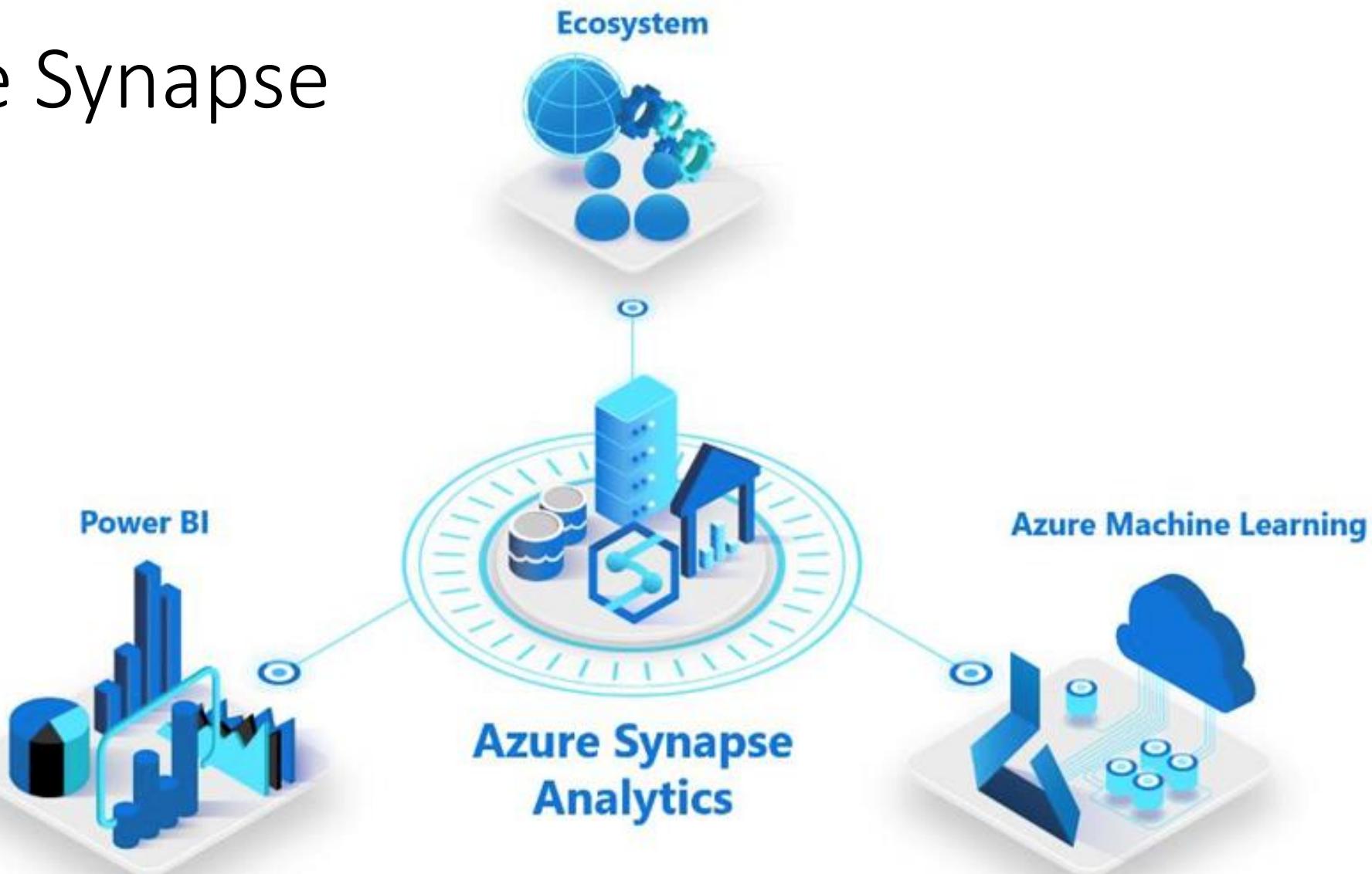


Ops



Finance

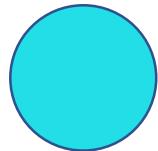
Azure Synapse



Azure Synapse Analytics

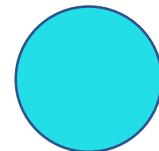
Relational Data

Data Warehouse

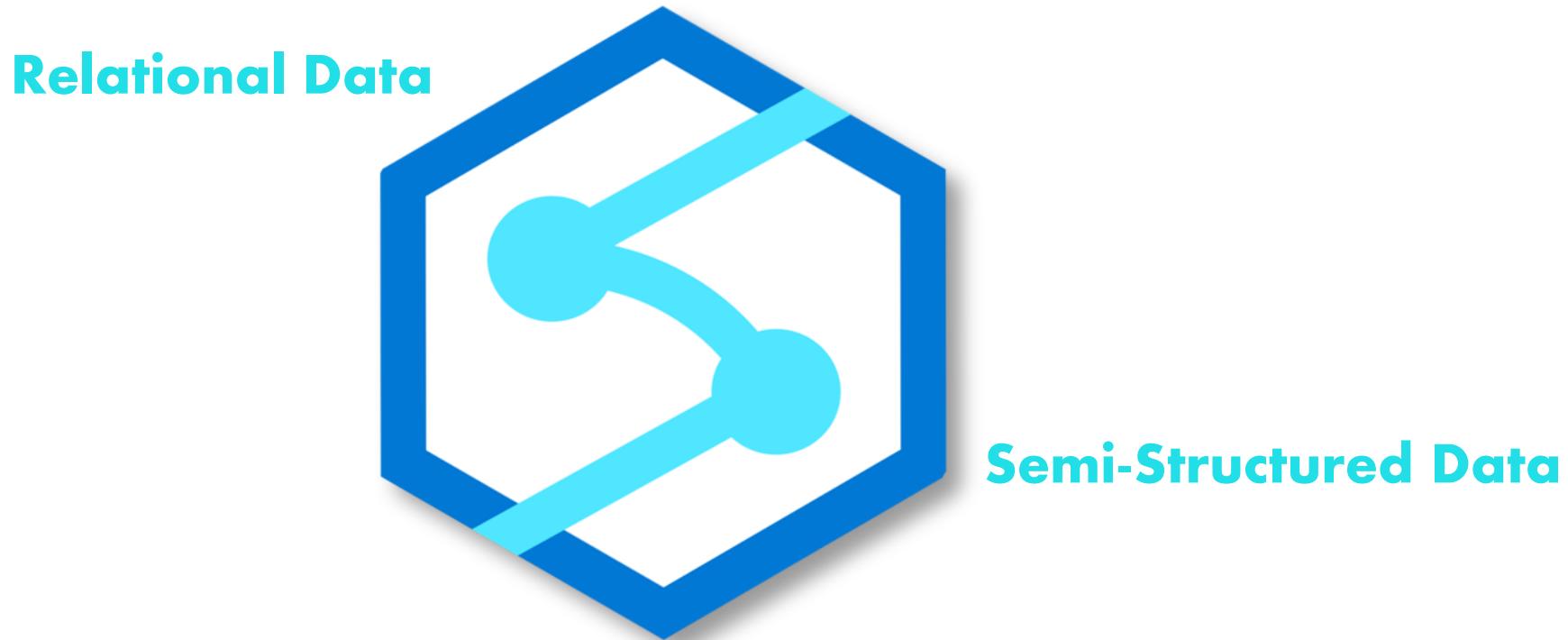


Semi-Structured Data

Data Lake



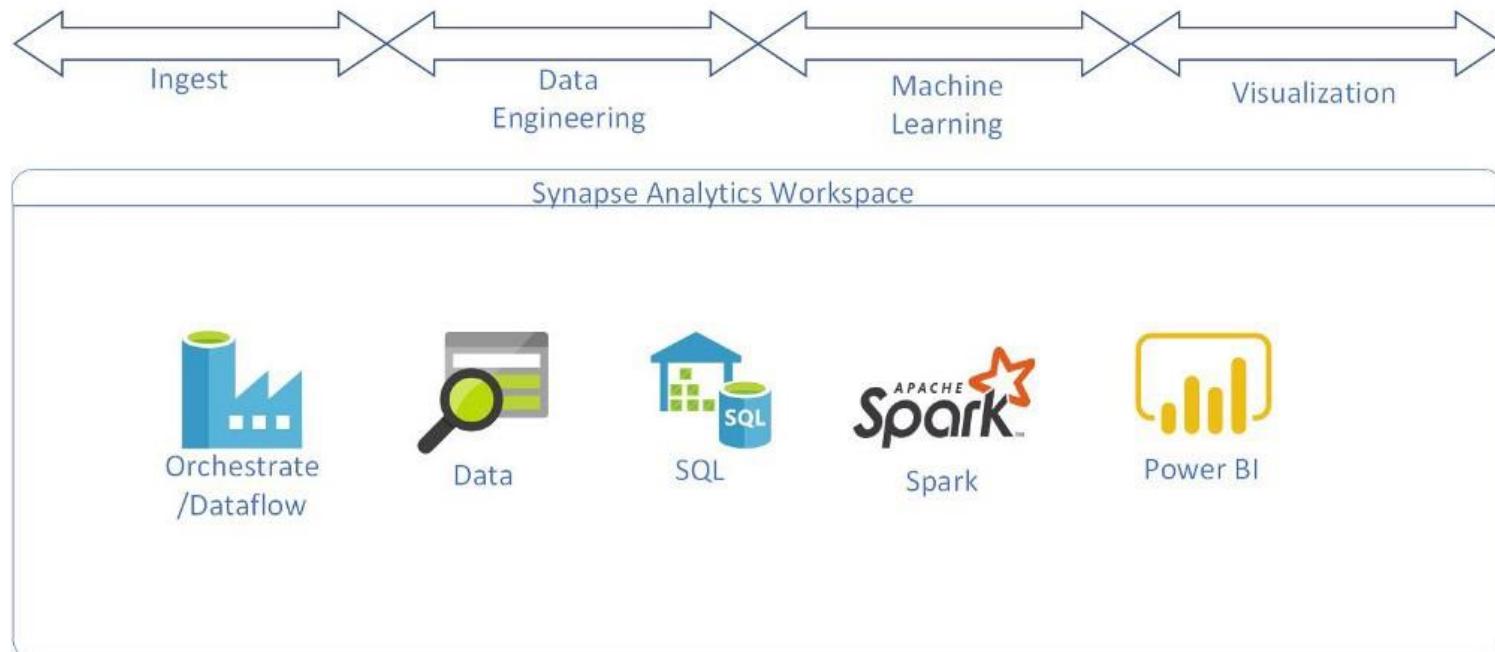
Azure Synapse Analytics



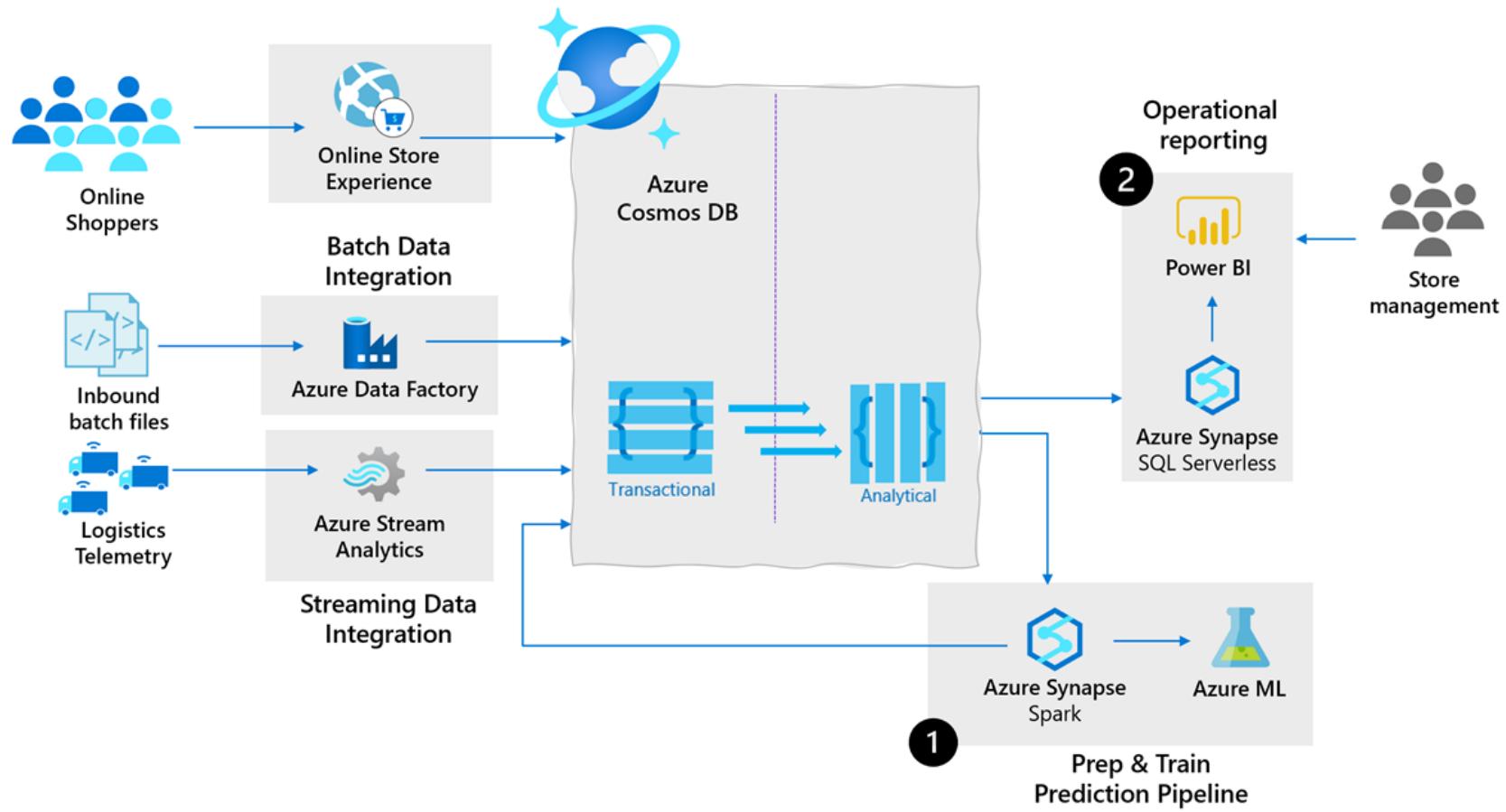
Azure Synapse Analytics

All encompassing Environment

End to End Analytics workspace



Azure Synapse Use Case



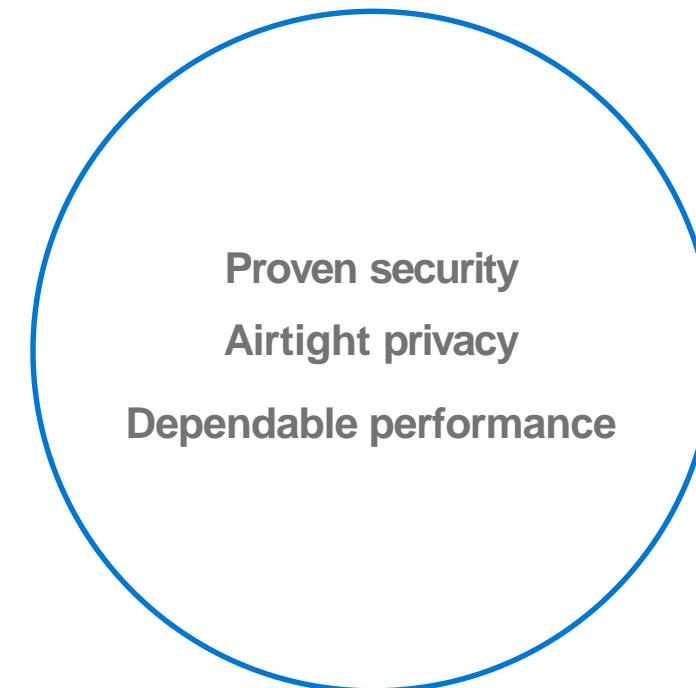
Designed to Bring this silos together

Big Data



Data Lake

Relational Data

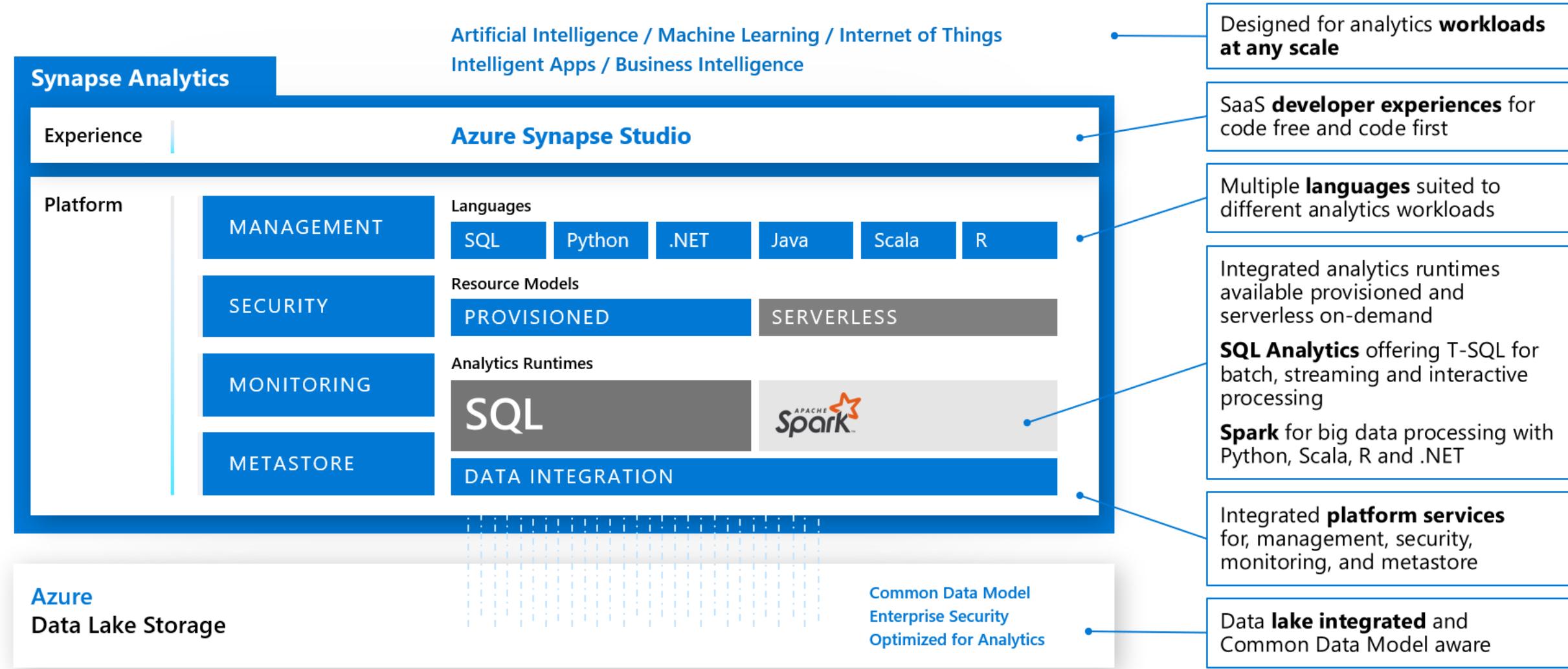


OR

Data Warehouse

Azure Synapse Analytics

Limitless analytics service with unmatched time to insight





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

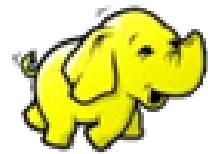
Azure Data Lake Gen 2



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

Microsoft Azure | Synapse Analytics > prlangadws2

Synapse workspace
prlangadws2

New ▾

Overview Data Develop Orchestrate Monitor Manage

Ingest Explore Analyze Visualize

Resources

Recent Pinned

NAME	LAST OPENED BY YOU
GreenCabTransformation	a day ago
EXE2 StoredProceduresCabs	a day ago
EXE3 Query Market Share SQL Pool	a day ago
EXE5 Query SQL OD Views	a day ago
EXE5 Create SQL OD Views	a day ago

Show more ▾

Name: prlangadws2
Region: West US 2
Resource group: prlangadrg
Subscription ID: 58f824d-32b0-4825-9825-02fa6a801546
[Select another workspace](#)

Useful links

[Synapse Analytics overview](#)
Discover the capabilities offered by Synapse and learn how to make the most of them.

[Pricing](#)
Learn about pricing details for Synapse capabilities.

[Documentation](#)
Visit the documentation center for quickstarts, how-to guides, and references for PowerShell, APIs, etc.

[Give feedback](#)
Share your comments or suggestions with us to improve Synapse.



Azure Studio Overview

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

The screenshot shows the Microsoft Azure Synapse Analytics workspace overview for the workspace 'prlangadws2'. The left sidebar includes links for Overview, Data, Develop, Orchestrate, Monitor, and Manage. The main area displays a 'Synapse workspace' titled 'prlangadws2' with a 'New' button. Below this are four cards: 'Ingest' (import data), 'Explore' (navigate data), 'Analyze' (use SQL or Spark), and 'Visualize' (build reports). A red box highlights these four cards. Further down are sections for 'Resources' (Recent and Pinned items) and 'Useful links' (Synapse Analytics overview, Pricing, Documentation, Give feedback). The bottom left shows workspace details: Name: prlangadws2, Region: West US 2, Resource group: prlangadrg, Subscription ID: 58f8824d-32b0-4825-9825-02fa6a801546, and a 'Select another workspace' link.

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

Explore
Learn how to navigate and interact with your data.

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

Visualize
Build interactive reports with integrated Power BI capabilities.

Resources

Recent Pinned

NAME	LAST OPENED BY YOU
CopyFHVData	4 days ago
GreenCabTransformation	7 days ago
EXE2 StoredProceduresCabs	7 days ago
EXE3 Query Market Share SQL Pool	7 days ago
EXE5 Query SQL OD Views	7 days ago

Show more ▾

Useful links

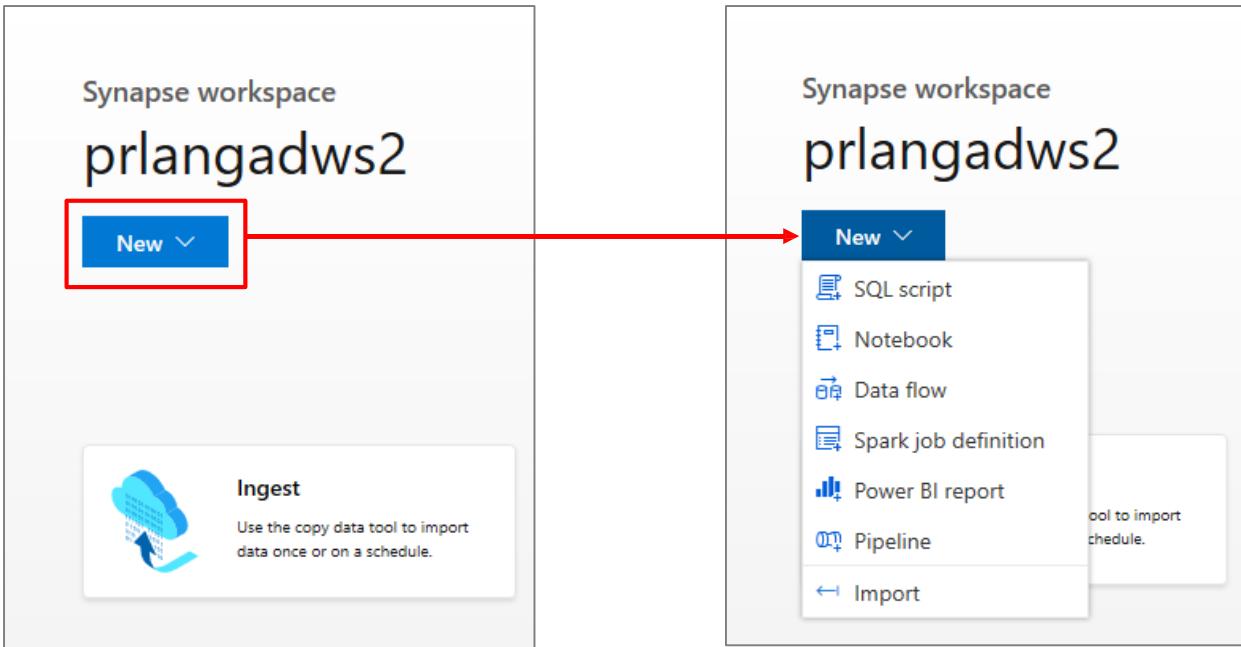
- [Synapse Analytics overview](#)
Discover the capabilities offered by Synapse and learn how to make the most of them.
- [Pricing](#)
Learn about pricing details for Synapse capabilities.
- [Documentation](#)
Visit the documentation center for quickstarts, how-to guides, and references for PowerShell, APIs, etc.
- [Give feedback](#)
Share your comments or suggestions with us to improve Synapse.

Name: prlangadws2
Region: West US 2
Resource group: prlangadrg
Subscription ID:
58f8824d-32b0-4825-9825-02fa6a801546
Select another workspace

Overview Hub

New dropdown – offers quickly start work item

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



Recent Pinned

NAME	LAST OPENED BY YOU
BOOT_AMLautoMLPredict	6 hours ago
SQLConnector	6 hours ago
TaxiCreateSparkTable	6 hours ago
Notebook 1	6 hours ago
NYCTAXI	6 hours ago

Show more ▾

Recent Pinned

NAME	LAST OPENED BY YOU
NYCTAXI	6 hours ago

Synapse Studio

Synapse Studio divided into **Activity hubs**

These organize the tasks needed for building analytics solutions

The screenshot shows the Microsoft Azure Synapse Analytics workspace interface for the workspace named "prlangadws". A red box highlights the left sidebar, which contains the following activity hubs:

- Overview
- Data
- Develop
- Orchestrate
- Monitor
- Manage

A red arrow points from the "New" button in the workspace header to the "Ingest" section under the "Overview" hub.

Overview
Quick-access to common gestures, most-recently used items, and links to tutorials and documentation.

Data
Explore structured and unstructured data

Develop
Write code and define business logic of the pipeline via notebooks, SQL scripts, Data flows, etc.

Integrate
Design pipelines that move and transform data.

Monitor
Centralized view of all resource usage and activities in the workspace.

Manage
Configure the workspace, pool, access to artifacts

Resources
Recent Pinned
NAME
NYCTaxiFinalWUS2_R
NYCTaxiFinalWUS2_R
HolidayDataPipeline
create notebook on s
Data flow 1

Name: prlangadws
Region: West US 2
Resource group: prlangadrg
Subscription ID: 5f9881a1-2361-4825-07f6-a8015a6

Demonstration

Azure Synapse Workspace



Diving into Data Tasks

Writing T-SQL

Ingestion

Spark

Provisioned

Synapse Pools

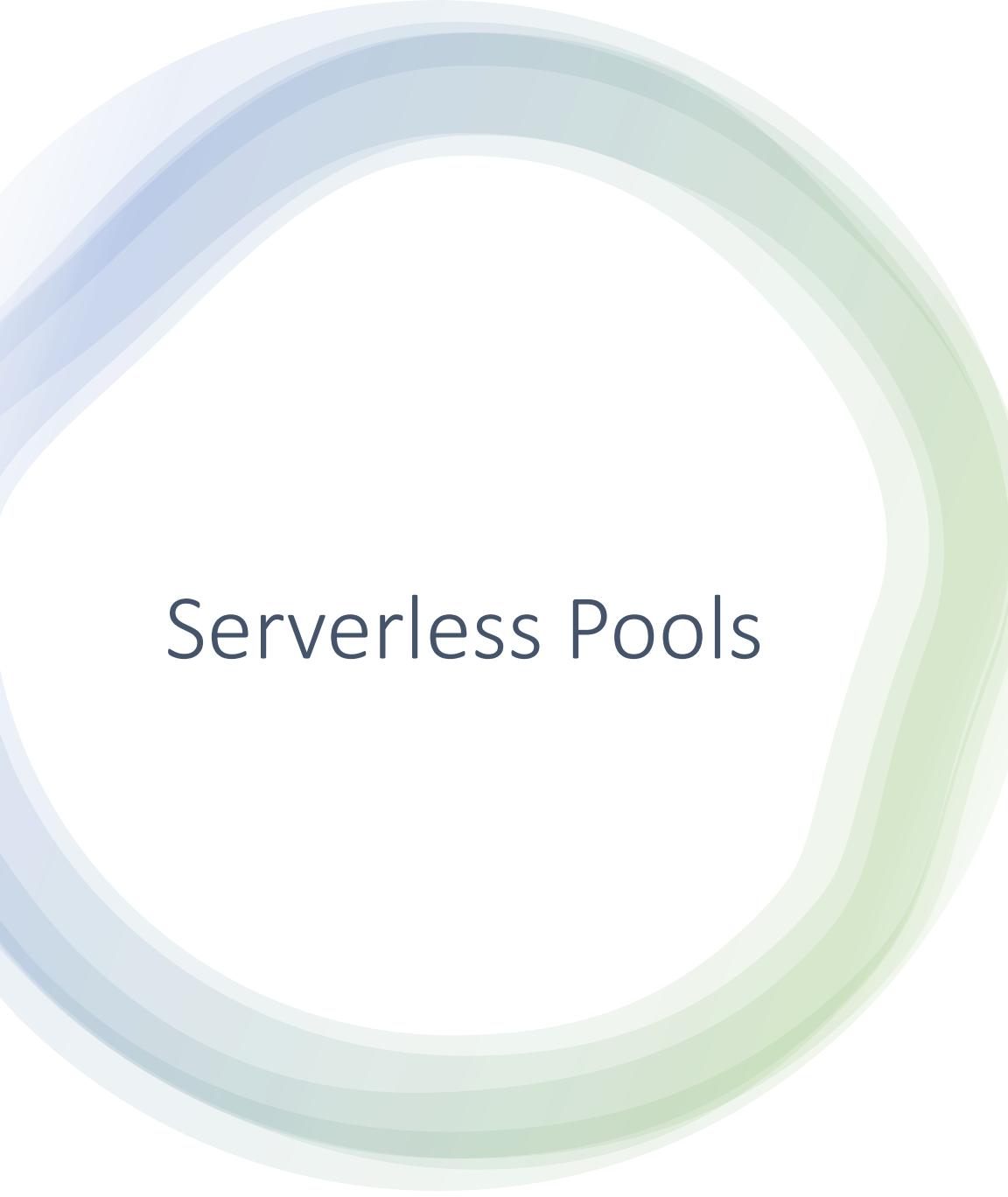
Serverless

SQL

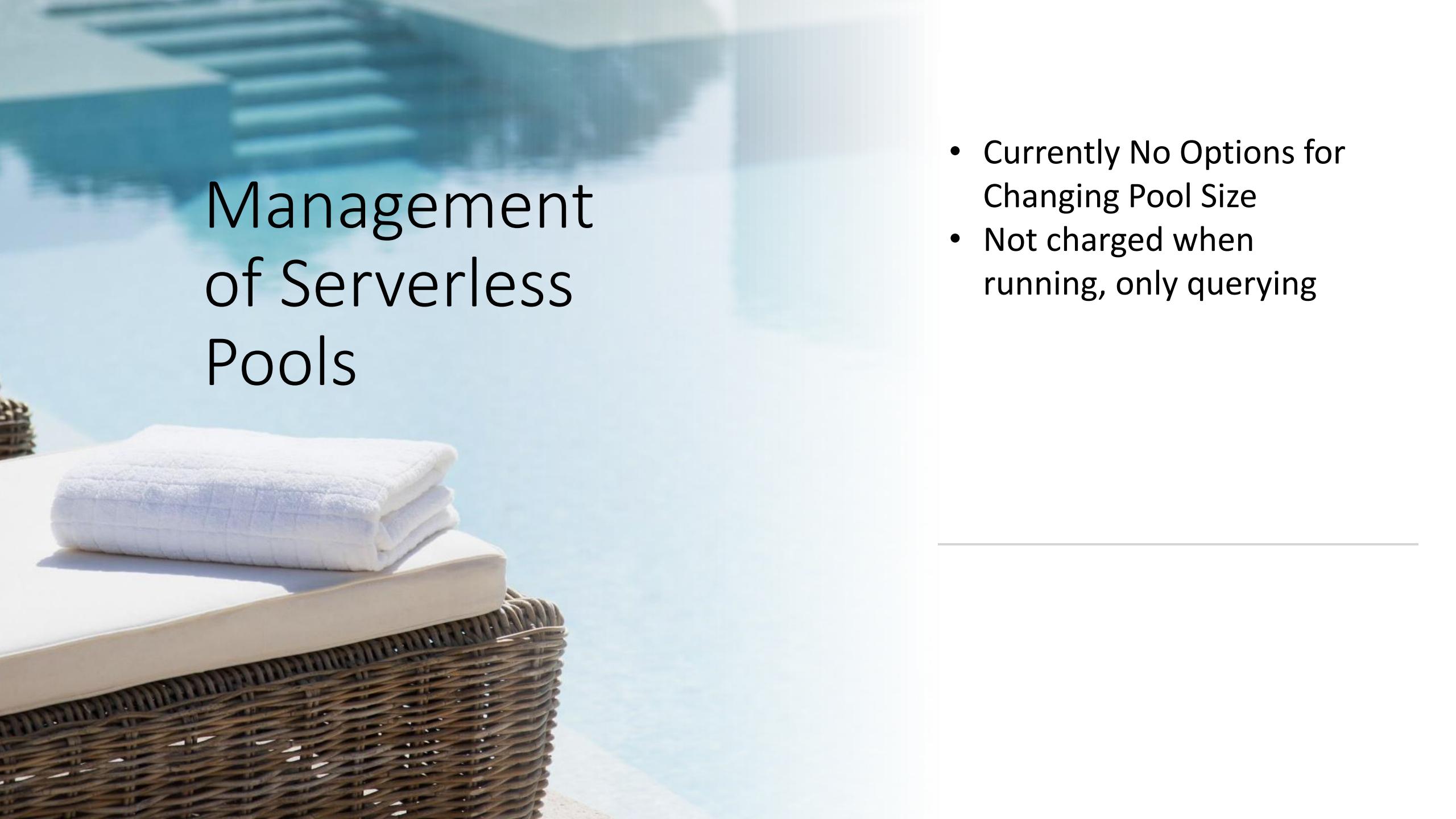
Apache Spark

Integration Runtimes

Serverless Pools



Auto Created by Synapse
Fully TSQL compliant
Flat file Examination
Charged \$5 per Terabyte

A photograph showing a close-up of a wicker chair's backrest and armrest, with two folded white towels resting on it. In the background, a large swimming pool with blue water and steps is visible under a clear sky.

Management of Serverless Pools

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

SQL Pools



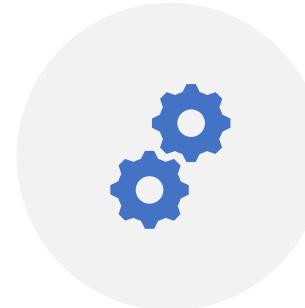
USED FOR DATA
WAREHOUSES



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

Evaluate 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

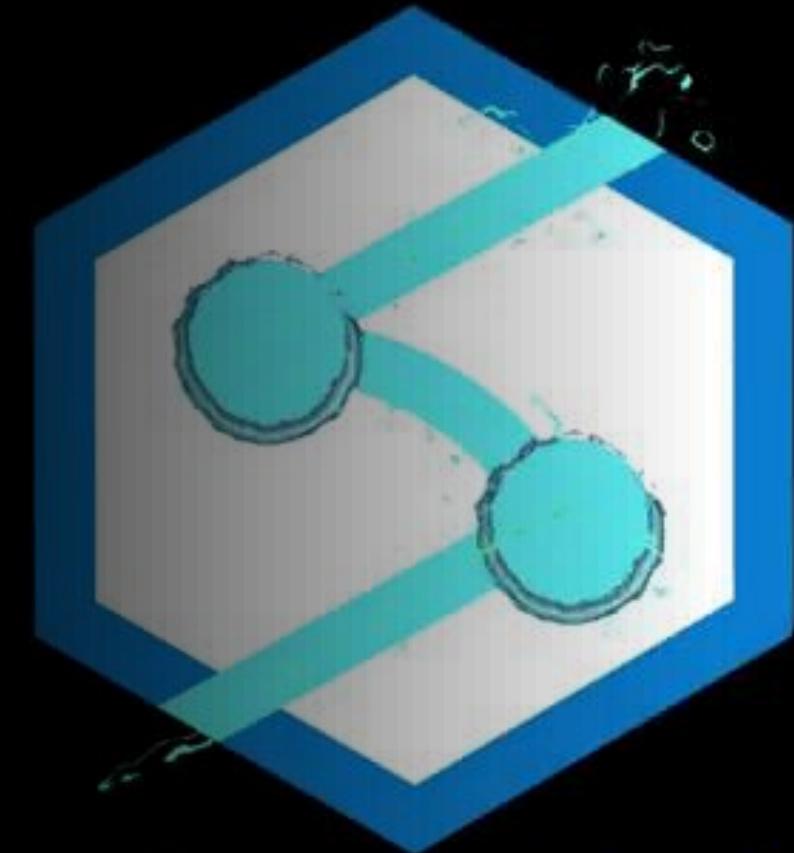




Quiz!

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



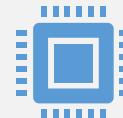
Indexing



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

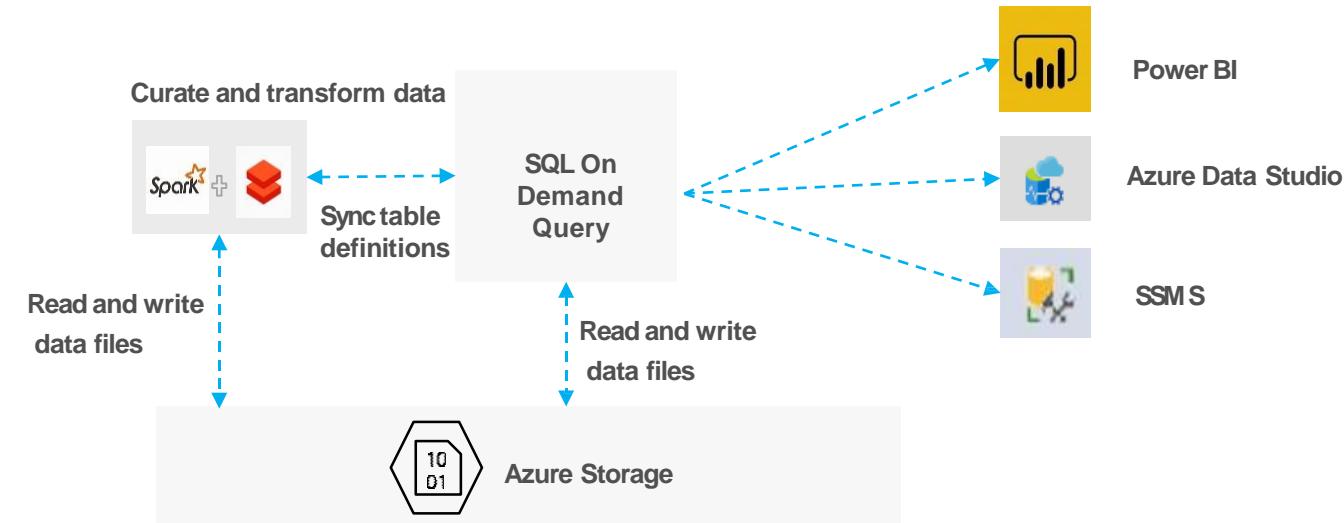
No ETL

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

The screenshot displays two side-by-side views of the Microsoft Azure Synapse Analytics interface.

Left View (Storage Browser): This view shows the storage account structure under the 'nyctlc' container. A context menu is open over a file named 'part-00133-tid-210938564719836543-aea5b543-5e83-4a7d-8d31-69f72c50b05d-15253-1.c000.snappy.parquet'. The menu options include 'New SQL script' (which is highlighted), 'New notebook', 'Copy ABFS path', 'Manage Access...', 'Rename...', 'Download', 'Delete', and 'Properties...'. The file was last modified on 10/25/2019, 2:20:23 PM.

Right View (Query Editor): This view shows a T-SQL query being run against the storage account. The 'SQL Analytics on-demand' option is selected in the 'Connect to' dropdown. The query itself is:

```
1 SELECT
2     TOP 100 *
3 FROM
4     OPENROWSET(
5         BULK 'https://prlangaddemosa.dfs.core.windows.net/nyctlc/yellow/puYear=2015/puMonth=3/part-00133-tid-210938564719836543-aea5b543-5e83-4a7d-8d31-69f72c50b05d-15253-1.c000.snappy.parquet'
6         FORMAT='PARQUET'
7     ) AS nyc;
```

The results pane shows a table with columns: VENDORID, TPEPICKUPDATETIME, TPEPDROPOFFDATETIME, PASSENGERCOUNT, TRIPDISTANCE, PULOCATIONID, DLOCATIONID, STARTLON, STARTLAT, ENDLON, and ENDLAT. The first few rows of data are:

VENDORID	TPEPICKUPDATETIME	TPEPDROPOFFDATETIME	PASSENGERCOUNT	TRIPDISTANCE	PULOCATIONID	DLOCATIONID	STARTLON	STARTLAT	ENDLON	ENDLAT
2	2015-02-28T23:5...	2015-03-01T00:0...	6	1.63	NULL	NULL	-74.000846862793	40.7306938171387	-73.	
1	2015-03-28T19:2...	2015-03-28T19:2...	1	2.2	NULL	NULL	-73.977653503418	40.7631607055664	-73.	
2	2015-02-28T23:5...	2015-03-01T00:1...	5	3.23	NULL	NULL	-73.96012878417...	40.7621574401855	-73.	
1	2015-03-28T19:2...	2015-03-28T19:3...	1	2.1	NULL	NULL	-73.98143005371...	40.7815055847168	-74.	
2	2015-02-28T23:5...	2015-03-01T00:1...	1	3.52	NULL	NULL	-73.98373413085...	40.7497062683105	-74.	

At the bottom of the results pane, a message indicates: '00:01:00 Query executed successfully.'

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
) 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]
```

	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	109768815
https://XXX.blob.core.windows.net/csv/taxi/yellow_tripdata_2017-11.csv	2017	11	119284803
https://XXX.blob.core.windows.net/csv/taxi/yellow_tripdata_2017-12.csv	2017	12	129508276

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

The image displays three screenshots of the Microsoft Azure Synapse Analytics portal. The top-left screenshot shows a SQL script window with the following code:

```
1 -- type your sql script here, we now have intellisense
2 CREATE VIEW yellow_2017 AS
3 SELECT *
4 FROM
5 OPENROWSET(
6      BULK 'https://prlangaddemoa.dfs.core.windows.net/nyctlc/yellow/puYear=2017/*/*',
7      FORMAT='PARQUET'
8 ) AS nyc
9
```

The bottom-left screenshot shows the results of the query execution, with a message indicating success: "00:00:17 Query executed successfully."

The middle-right screenshot shows the same SQL script window with the "SQL Analytics on-demand" checkbox selected (highlighted by a red box). The results tab is selected, displaying the following table:

(NO COLUMN NAME)	PASSENGERCOUNT	CNT
2017	0	166086
2017	1	81034075
2017	2	16545571
2017	3	4748869
2017	4	2257813
2017	5	5407319

The bottom-right screenshot shows the same results table, but the "Chart" tab is selected, displaying a line chart with two data series: "passengerCount" (blue line with diamond markers) and "cnt" (green line with square markers). The X-axis represents passenger count values from 0 to 10, and the Y-axis represents counts from 0 to 100M. The chart shows a peak at passenger count 1 with a value of approximately 80M.

This screenshot shows the Microsoft Azure Synapse Analytics interface with a query results page. The "SQL Analytics on-demand" checkbox is selected (highlighted by a red box). The results table is displayed in "Table" view, showing the following data:

(NO COLUMN NAME)	PASSENGERCOUNT	CNT
2017	0	166086
2017	1	81034075
2017	2	16545571
2017	3	4748869
2017	4	2257813
2017	5	5407319

A message at the bottom indicates "00:02:19 Query executed successfully."

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!Info'  
;  
-- 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

Microsoft Azure | Synapse Analytics > prlangadws2 | Search resources | Bind | Unbind | Feedback | Help | Sign in | prlangad@microsoft.com

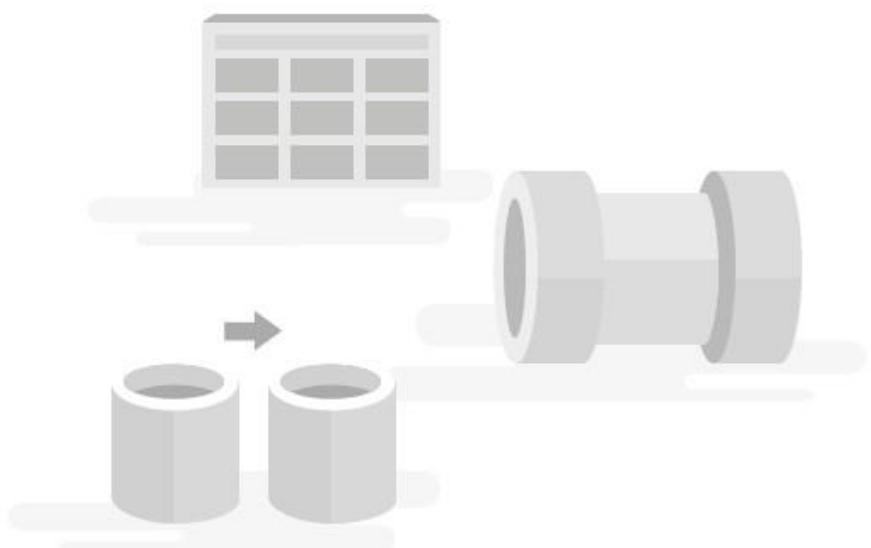
Overview | Data (selected) | Develop | Orchestrate | Monitor | Manage

Publish all | Validate all | Refresh | Discard all

Data | + Add | < Back

Filter resources by name

- ▶ Storage accounts
- ▶ Databases
- ▶ Datasets



Select an item from the resource explorer or create a new item

Name: prlangadws2
Region: West US 2
Resource group: prlangadrg
Subscription ID:
58f8824d-32b0-4825-9825-02fa6a801546
[Select another workspace](#)

Data Hub – Storage Accounts

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

ADLS Gen2 Account

Container (filesystem)

The screenshot shows the Microsoft Azure Synapse Analytics Data blade. On the left, the sidebar has 'Data' selected. The main area shows 'Storage accounts' with 'prlangaddemosa (Primary)' expanded, revealing containers like 'filesystem', 'holidaydatacontainer', 'isdweatherdatacontainer', and 'nyctlc'. The 'nyctlc' container is also expanded, showing sub-containers 'puYear=2001' through 'puYear=2026'. A red box highlights the 'nyctlc' container in the list and the 'Filepath' input field at the top right, which contains 'nyctlc > yellow'. The bottom of the blade displays account details: Name: prlangadws2, Region: West US 2, Resource group: prlangadrg, Subscription ID: 58f8824d-32b0-4825-9825-02fa6a801546.

NAME	LAST MODIFIED	CONTENT TYPE	SIZE
puYear=2001	10/25/2019, 2:25:03 PM	Folder	
puYear=2002	10/25/2019, 2:25:21 PM	Folder	
puYear=2003	10/25/2019, 2:25:03 PM	Folder	
puYear=2008	10/25/2019, 2:20:38 PM	Folder	
puYear=2009	10/25/2019, 2:19:33 PM	Folder	
puYear=2010	10/25/2019, 2:19:24 PM	Folder	
puYear=2011	10/25/2019, 2:23:56 PM	Folder	
puYear=2012	10/25/2019, 2:20:01 PM	Folder	
puYear=2013	10/25/2019, 2:19:52 PM	Folder	
puYear=2014	10/25/2019, 2:24:06 PM	Folder	
puYear=2015	10/25/2019, 2:20:12 PM	Folder	
puYear=2016	10/25/2019, 2:19:21 PM	Folder	
puYear=2017	10/25/2019, 2:20:28 PM	Folder	
puYear=2018	10/25/2019, 2:24:38 PM	Folder	
puYear=2019	10/25/2019, 2:20:33 PM	Folder	
puYear=2020	10/25/2019, 2:24:47 PM	Folder	
puYear=2021	10/25/2019, 2:28:34 PM	Folder	
puYear=2026	10/25/2019, 2:20:20 PM	Folder	

Data Hub – Storage accounts

The screenshot shows the Data Hub interface. On the left, there's a sidebar with sections for Storage accounts, Databases, and Datasets. Under Storage accounts, 'prlangaddemosa (Primary)' is expanded, showing its contents: filesystem, holidaydatacontainer, isdweatherdatacontainer, nyctlc, prlangaddemosa, tmpcontainer, and wwidporters. The 'filesystem' item is selected. The main area shows a 'Data' view with tabs for 'nyctlc' and 'filesystem'. The 'filesystem' tab is active, displaying a list of files and folders: synapse, temp, tmp, dbo.StoreSales.parquet, dbo.StoreSales.txt, employee.json, and SampleCSVFile_2kb.csv. A context menu is open over the 'SampleCSVFile_2kb.csv' file, with options like Preview, New notebook, Copy ABFSS path, Manage Access..., Rename..., Download, Delete, and Properties... (the last one is highlighted with a red box).

A 'Properties' dialog box is open, showing system properties for the file 'SampleCSVFile_2kb.csv'. The 'Name' field is 'SampleCSVFile_2kb.csv', 'URL' is 'https://prlangaddemosa.dfs.core.windows.net/filesystem/SampleCSVFile_2kb.csv', 'LastModified' is 'Tue, 29 Oct 2019 20:30:21 GMT', 'ContentType' is 'application/vnd.ms-excel', and 'ContentLanguage' is empty. There is also a section for 'User Properties' with a 'Add User Property' button. At the bottom are 'Save' and 'Cancel' buttons.

System Properties	
Name	SampleCSVFile_2kb.csv
URL	https://prlangaddemosa.dfs.core.windows.net/filesystem/SampleCSVFile_2kb.csv
LastModified	Tue, 29 Oct 2019 20:30:21 GMT
CacheControl	
ContentType	application/vnd.ms-excel
ContentDisposition	
ContentEncoding	
ContentLanguage	
User Properties	
Add User Property	

Data Hub – Storage accounts

Manage Access - Configure standard POSIX ACLs on files and folders

The screenshot shows the Data Hub interface with the 'Storage accounts' section selected. In the center, there's a file browser window titled 'nytclc' showing a 'filesystem' folder. Inside 'filesystem', there are several items: 'synapse', 'temp', 'tmp', 'dbo.StoreSales.parquet', 'dbo.StoreSales.txt', 'employee.json', and 'SampleCSVFile_2kb.csv'. A context menu is open over 'SampleCSVFile_2kb.csv', with the 'Manage Access...' option highlighted by a red box and a red arrow pointing to the corresponding button in the 'Manage Access' dialog.

The 'Manage Access' dialog is open, showing the permissions for the file 'SampleCSVFile_2kb.csv'. It lists '\$superuser' as the owner and the owning group. Under 'Permissions for: \$superuser', 'Access' is checked for Read, Write, and Execute. There is a field to 'Add user or group' with an 'Add' button.

User/Group	Read	Write	Execute
\$superuser (Owner)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
\$superuser (Owning Group)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Other	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Mask	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Permissions for: \$superuser

	Read	Write	Execute
Access	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>

Add user or group:

Enter a UPN or Object ID

Add

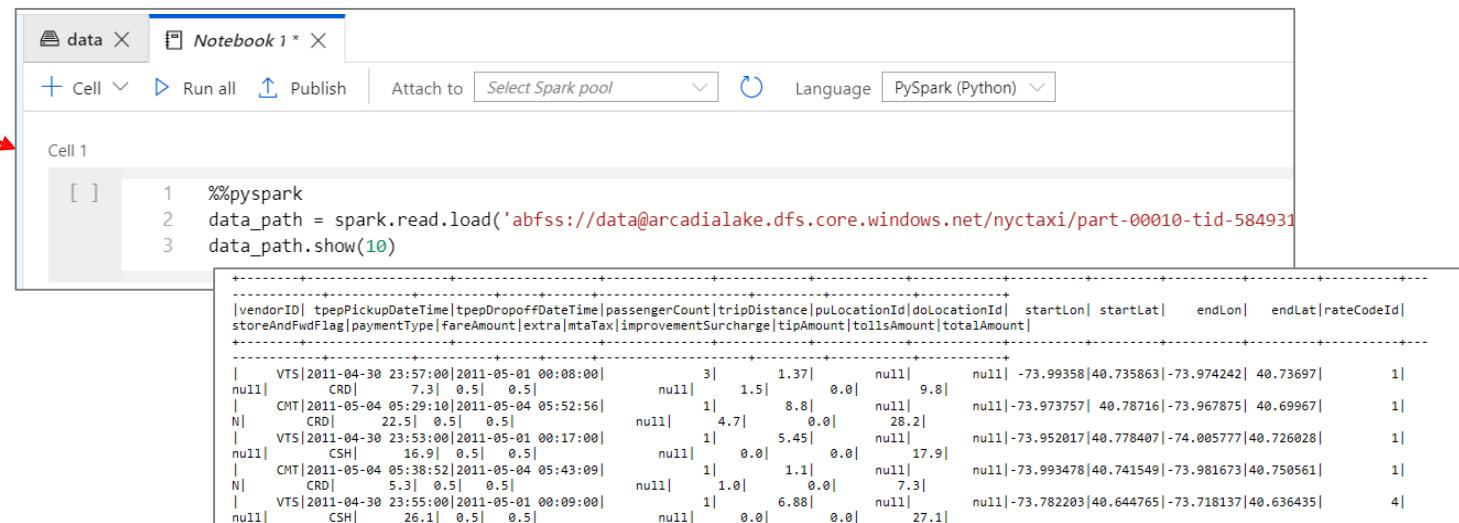
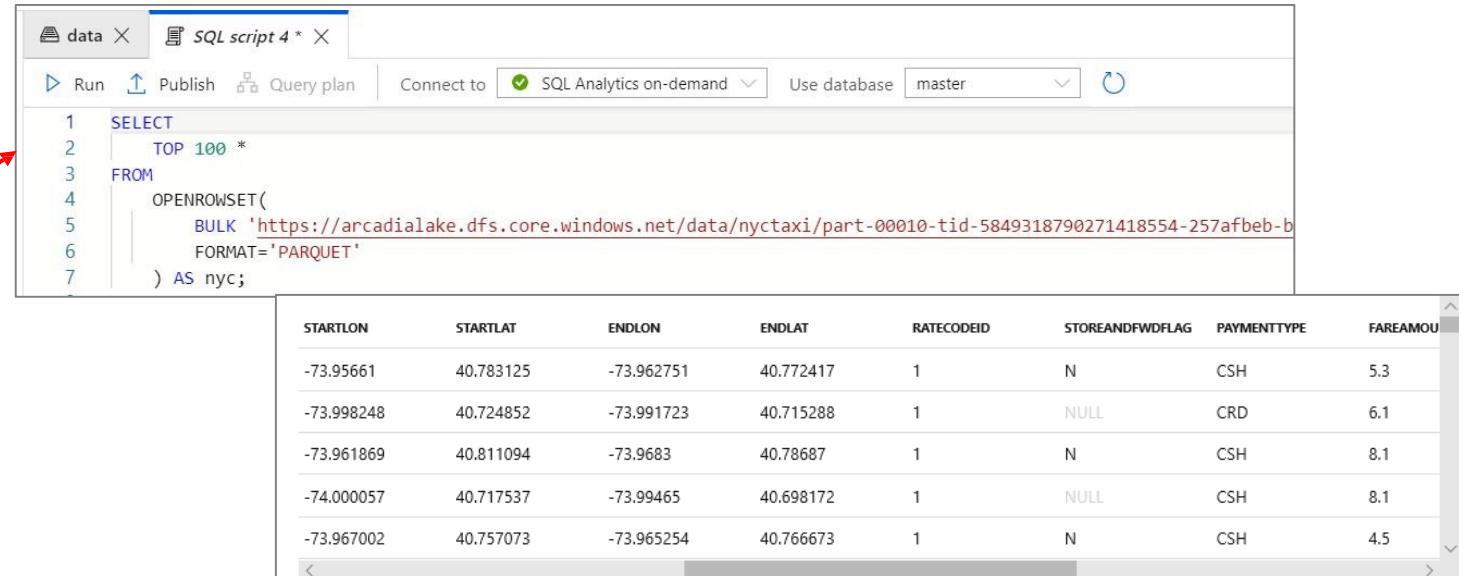
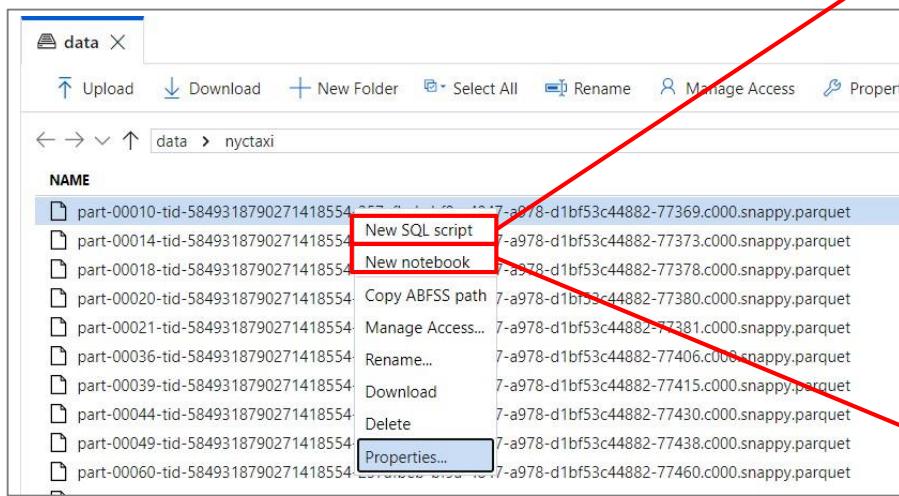
Save Cancel

Data Hub – Storage

accounts

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

Auto-generates T-SQL or PySpark



Data Hub – Storage accounts

SQL Script from Multiple files

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

The screenshot shows the Azure Data Explorer interface. On the left, there is a file browser window titled 'isdweatherdatacontainer > ISDWeathercurated'. It lists several Parquet files with their names and last modified times. A red arrow points from the 'New SQL script' button in the context menu of one of the files to the generated SQL script on the right. The SQL script is a T-SQL query that reads multiple Parquet files from a specified URL using OPENROWSET and BULK options.

NAME	LAST MODIFIED
_SUCCESS	10/25/2019, 3:11:17 P
part-00000-7bc127b8-56a2-443f-b6f9-18175e9fdad7-c000.snappy.parquet	10/25/2019, 3:09:38 P
part-00000-e3d623d5-1ecd-4953-bcf9-ad973c404bd1-c000.snappy.parquet	10/25/2019, 3:00:08 P
part-00001-7bc127b8-56a2-443f-b6f9-18175e9fdad7-c000.snappy.parquet	10/25/2019, 3:10:05 P
part-00001-e3d623d5-1ecd-4953-bcf9-ad973c404bd1-c000.snappy.parquet	10/25/2019, 3:00:03 P
part-00002-7bc127b8-56a2-443f-b6f9-18175e9fdad7-c000.snappy.parquet	10/25/2019, 3:10:13 P
part-00002-e3d623d5-1ecd-4953-bcf9-ad973c404bd1-c000.snappy.parquet	10/25/2019, 2:59:17 P
part-00003-7bc127b8-56a2-443f-b6f9-18175e9fdad7-c000.snappy.parquet	10/25/2019, 3:00:22 P
part-00003-e3d623d5-1ecd-4953-bcf9-ad973c404bd1-c000.snappy.parquet	10/25/2019, 2:59:50 P
part-00004-7bc127b8-56a2-443f-b6f9-18175e9fdad7-c000.snappy.parquet	10/25/2019, 3:09:54 P
part-00004-e3d623d5-1ecd-4953-bcf9-ad973c404bd1-c000.snappy.parquet	10/25/2019, 2:59:55 P
part-00005-7bc127b8-56a2-443f-b6f9-18175e9fdad7-c000.snappy.parquet	10/25/2019, 3:10:01 P
part-00005-e3d623d5-1ecd-4953-bcf9-ad973c404bd1-c000.snappy.parquet	10/25/2019, 2:59:04 P
part-00006-7bc127b8-56a2-443f-b6f9-18175e9fdad7-c000.snappy.parquet	10/25/2019, 3:11:16 P
part-00006-e3d623d5-1ecd-4953-bcf9-ad973c404bd1-c000.snappy.parquet	10/25/2019, 3:00:56 P

```
-- Read multiple parquet files with same schema
SELECT
    TOP 100 *
FROM
    OPENROWSET(
        BULK 'https://prlangaddemosa.dfs.core.windows.net/isdweatherdatacontainer/ISDWeathercurated/\*.parquet',
        FORMAT = 'Parquet'
    ) AS [r]
WHERE
    r.filepath() in (
        'https://prlangaddemosa.dfs.core.windows.net/isdweatherdatacontainer/ISDWeathercurated/part-00000',
        'https://prlangaddemosa.dfs.core.windows.net/isdweatherdatacontainer/ISDWeathercurated/part-00001',
        'https://prlangaddemosa.dfs.core.windows.net/isdweatherdatacontainer/ISDWeathercurated/part-00002',
        'https://prlangaddemosa.dfs.core.windows.net/isdweatherdatacontainer/ISDWeathercurated/part-00003',
        'https://prlangaddemosa.dfs.core.windows.net/isdweatherdatacontainer/ISDWeathercurated/part-00004',
        'https://prlangaddemosa.dfs.core.windows.net/isdweatherdatacontainer/ISDWeathercurated/part-00005',
        'https://prlangaddemosa.dfs.core.windows.net/isdweatherdatacontainer/ISDWeathercurated/part-00006',
        'https://prlangaddemosa.dfs.core.windows.net/isdweatherdatacontainer/ISDWeathercurated/part-00007'
    )
```

Data Hub – Databases

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

The screenshot displays two views of a workspace's database structure. On the left, a 'Data' pane shows a hierarchical list of resources. On the right, a detailed view of the 'Databases' section is shown. Red arrows point from specific items in the left pane to their corresponding expanded sections in the right pane.

Left Pane (Data View):

- Storage accounts:** 2 items
- Databases:** 3 items
 - sql1 (SQL pool)**
 - sample (SQL on-demand)**
 - default (Spark)**
 - Tables:** 2 items
 - nyntaxiyellow7days**
 - searchlogtable**
- Datasets:** 2 items

Right Pane (Detailed Database View):

- Storage accounts:** 2 items
- Databases:** 3 items
 - sql1 (SQL pool)**
 - Tables**
 - External tables**
 - External resources**
 - Views**
 - Programmability**
 - Schemas**
 - Security**
 - sample (SQL on-demand)**
 - External tables**
 - External resources**
 - Views**
 - Schemas**
 - Security**
 - default (Spark)**
 - Tables**
- Datasets:** 2 items

Data Hub – Databases

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

A screenshot of the Data Hub interface showing the 'Columns' section for the 'dbo.NycTaxiPredict' table. A context menu is open over the 'columns' row, showing options: 'New SQL script', 'New notebook', 'Refresh', 'Select TOP 1000 rows', 'CREATE', 'DROP', and 'DROP and CREATE'.

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

A screenshot of the Data Hub interface showing the 'Columns' section for the 'dbo.NycTaxiPredict' table. A context menu is open over the 'columns' row, with the 'Load to DataFrame' option highlighted with a red box and a red arrow pointing down to the generated PySpark code in the 'Notebook 1' window below.

The 'Notebook 1' window shows the following code:

```
Cell 1
[ ] 1 val df = spark.read.sqlanalytics("sql1.dbo.NycTaxiPredict")
```



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

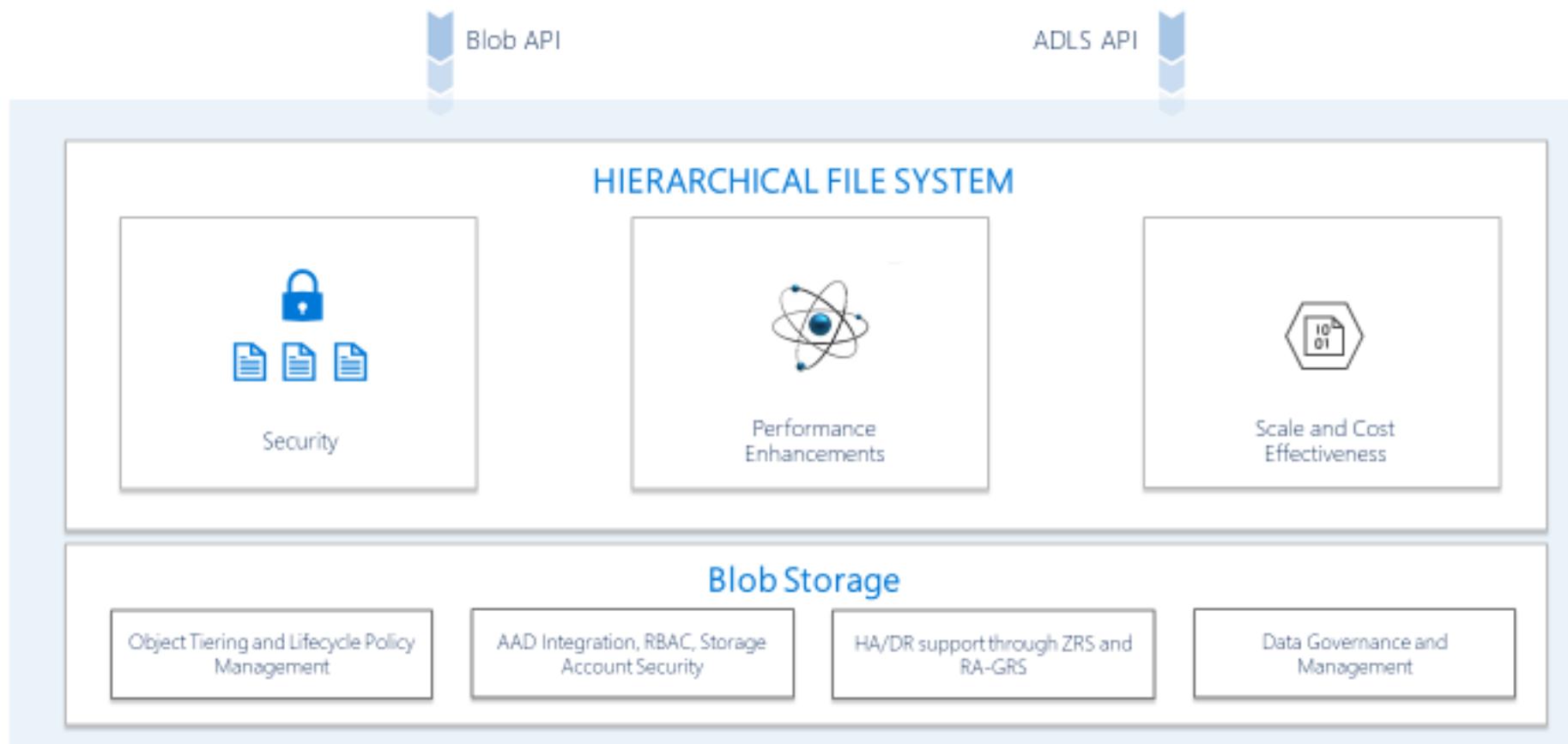
Performance

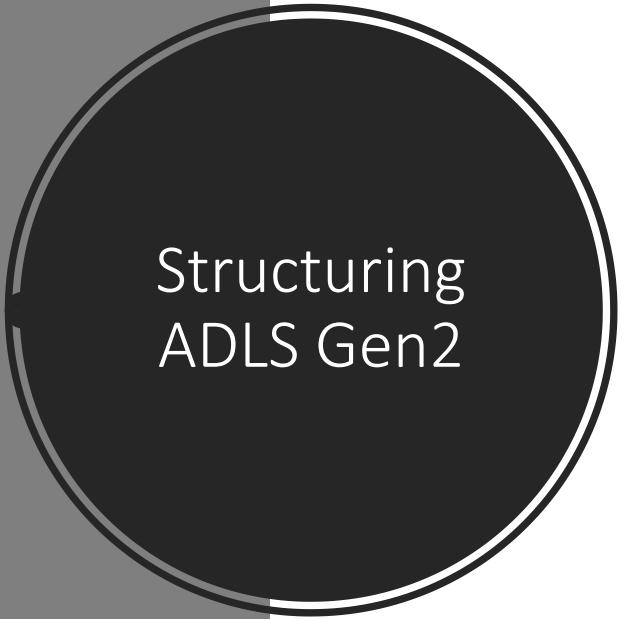
Improvements

Flexible Pricing

Underlying Architecture

ADLS Gen2 Architecture





Structuring ADLS Gen2

ADLS Gen 2 Filesystem

Raw Data
Bronze

Query Ready
Silver

Report Ready
Gold

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

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



Greater Flexibility

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

ADLS Gen 2 Filesystem

Raw

Structured

Stage

Reference

Curated

Archive

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\FileDialog_YYYY_MM_DD.csv

Optimum for folder security

Commonly used in Raw Zones

May also be in Curated or Staged Zones

File Formats

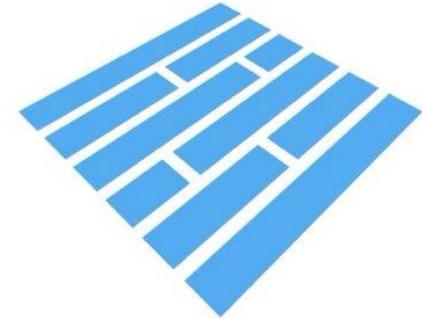
Common storage formats

CSV

Parquet

Delta Lake

Parquet Files



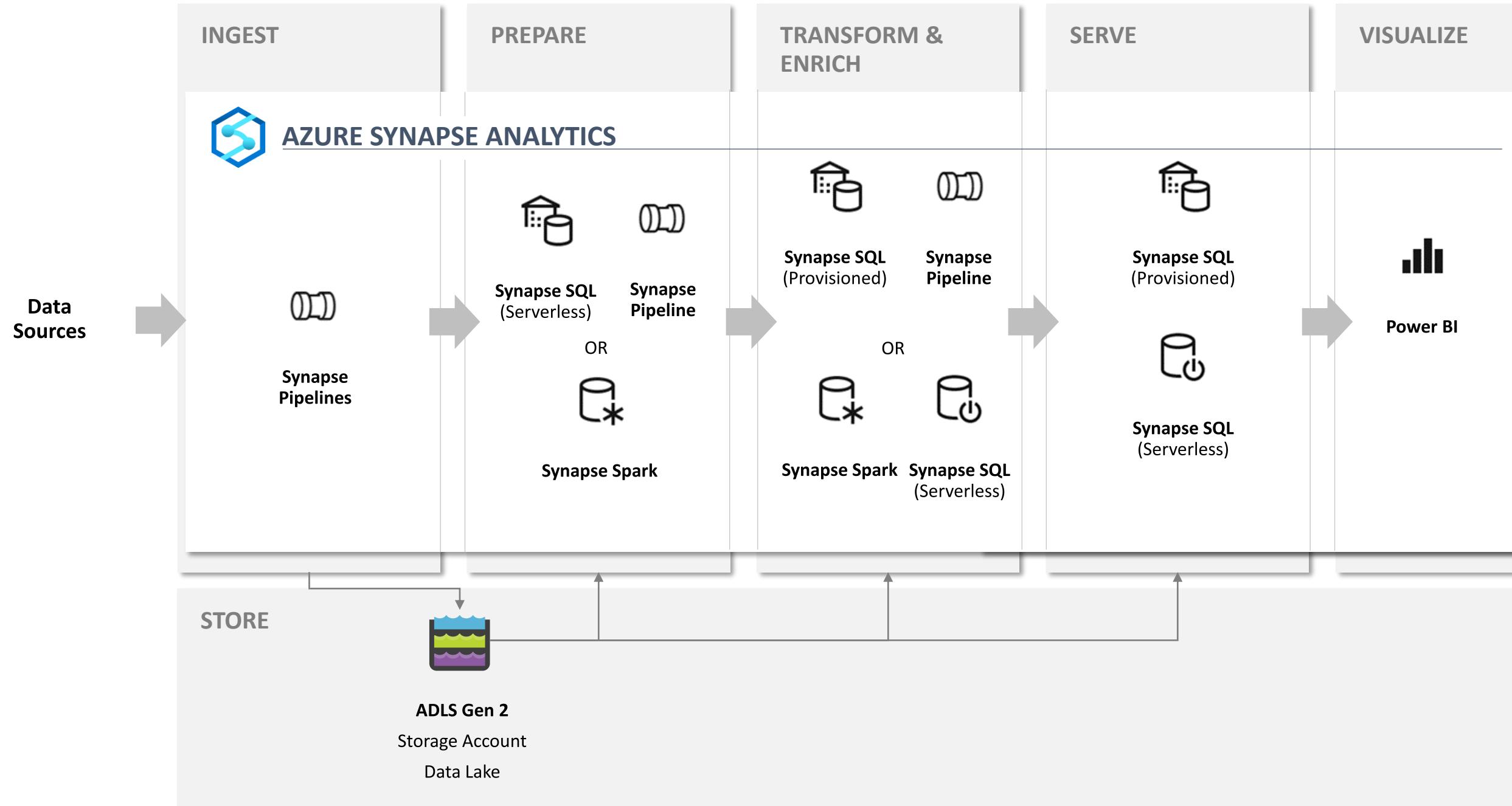
Optimally designed for Spark Processing

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



A yellow hexagonal graphic containing a white outline of a synapse or neural connection, which is part of the Microsoft Power BI logo.

Integrating Synapse with Power BI

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



Unify all
your data



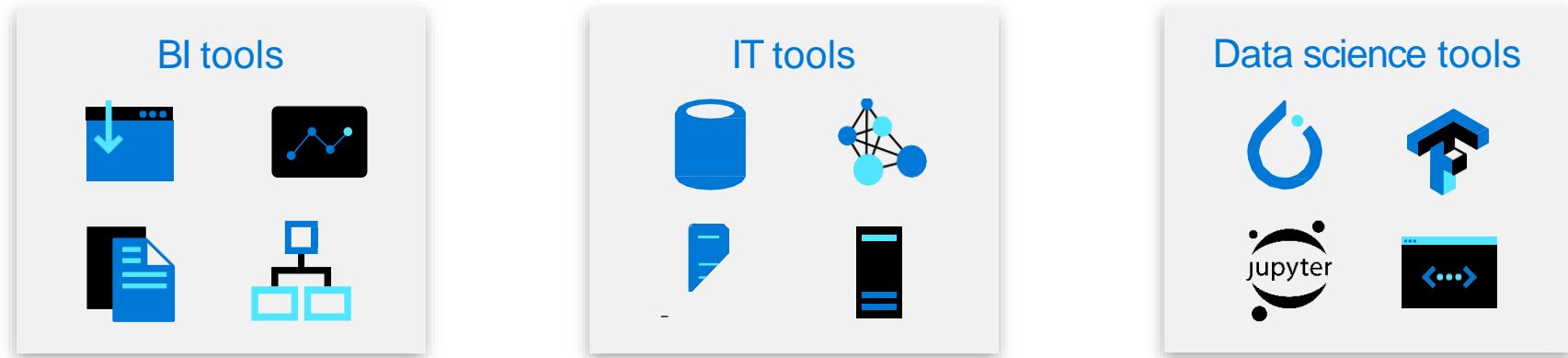
One source
of truth



Instant insights
on-the-fly

BI tools + Azure Synapse Analytics

Data silos and incompatible tooling inhibit collaboration



Incompatible tooling

Siloed data



Marketing



Sales



Product



Ops



Finance

Unify your organization



Power BI



Azure Data Services



Common Data Model on Azure Data Lake Gen 2 Storage

Unify your organization



Power BI

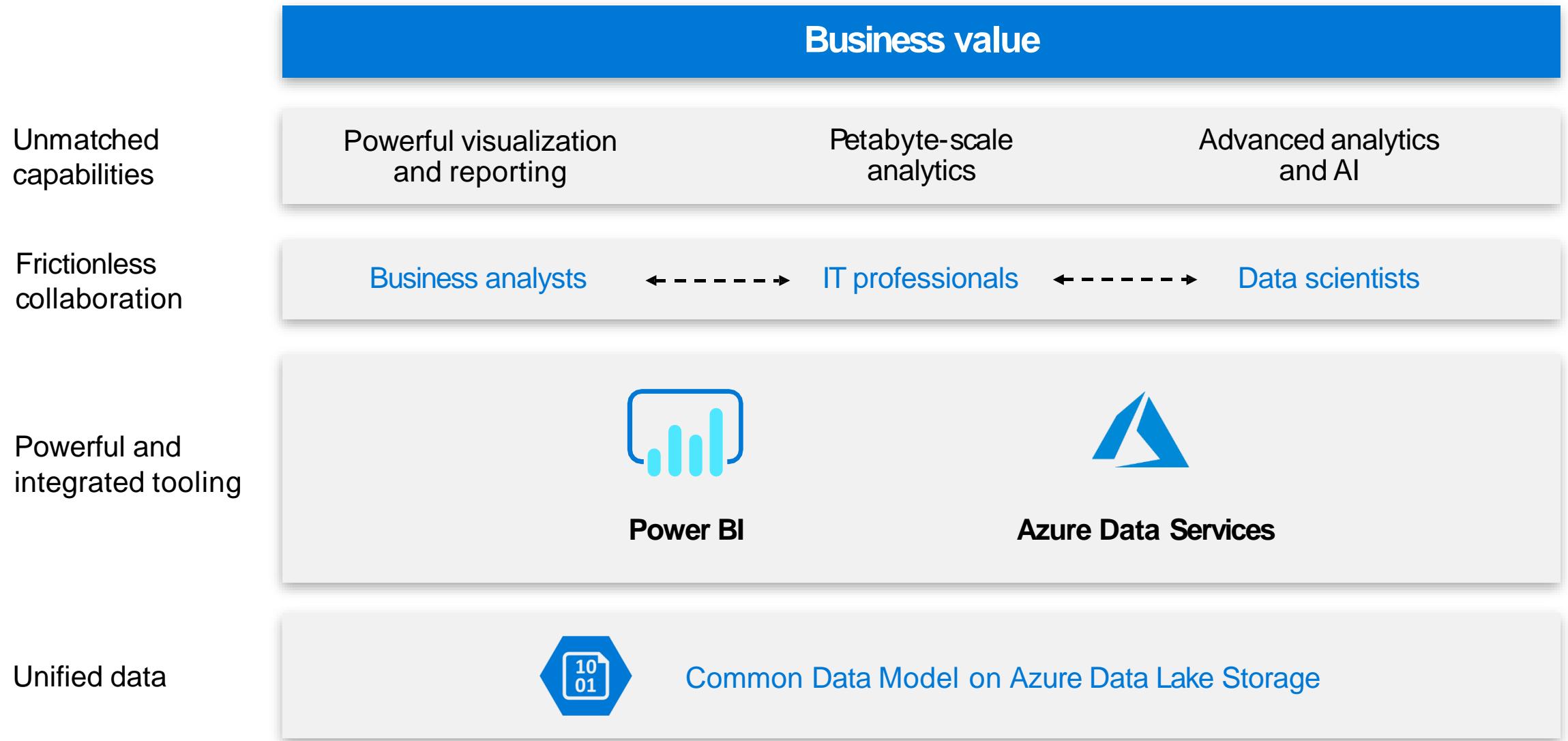


Azure Data Services



Common Data Model on Azure Data Lake Storage

Unleash data value with Power BI and Azure Data Services



Maximize the value of your data



+



+

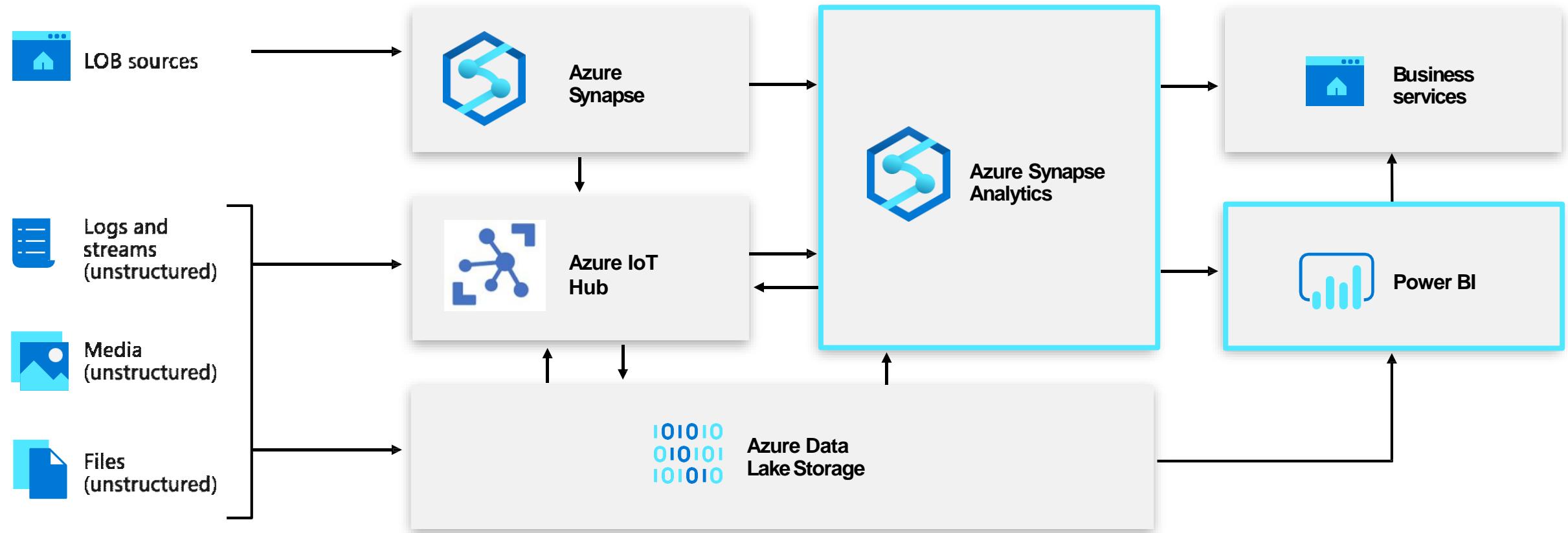


**Powerful visualization
and reporting**

**Petabyte-scale
analytics**

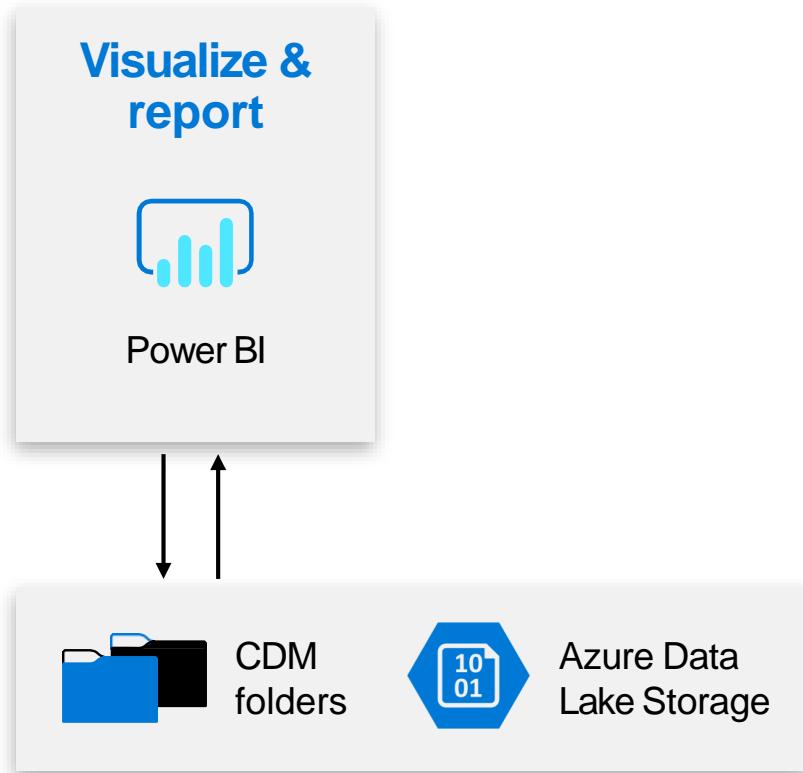
**Advanced
analytics and AI**

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



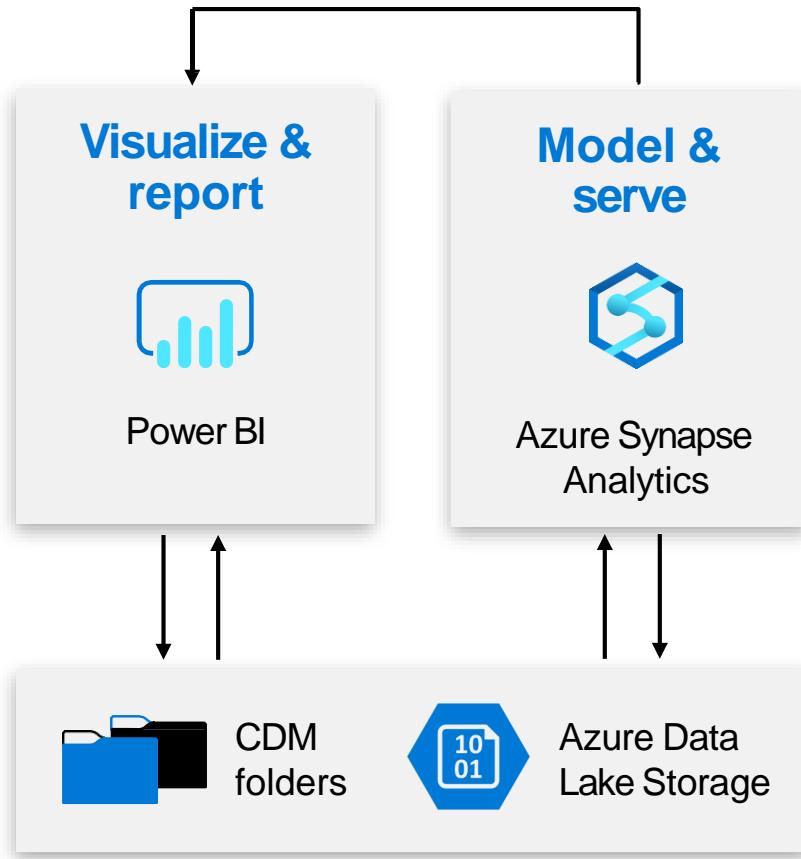
Leverage powerful visualization and reporting

Power BI



Deliver petabyte-scale analytics

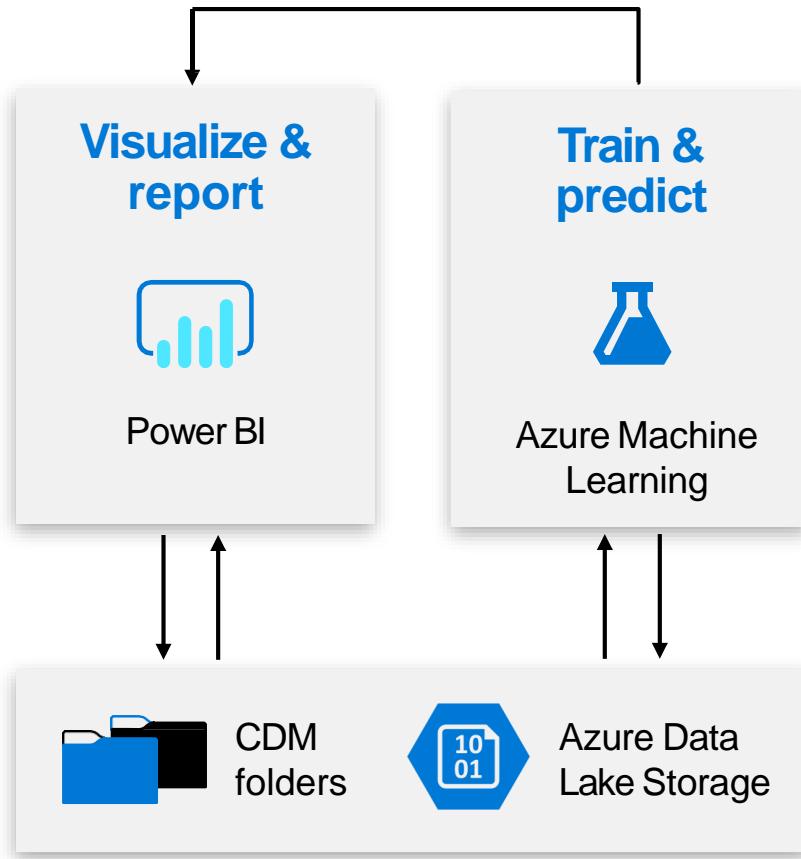
Power BI + Azure SQL Data Warehouse



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

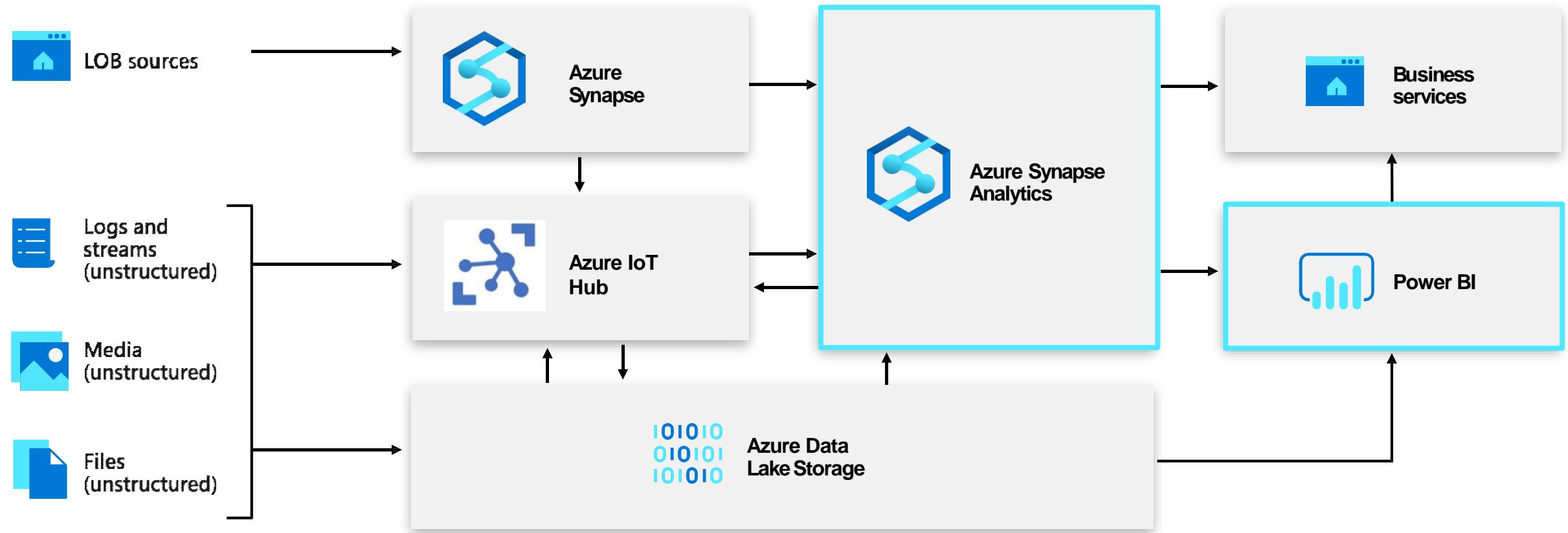
Gain deep insights with advanced analytics and AI

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

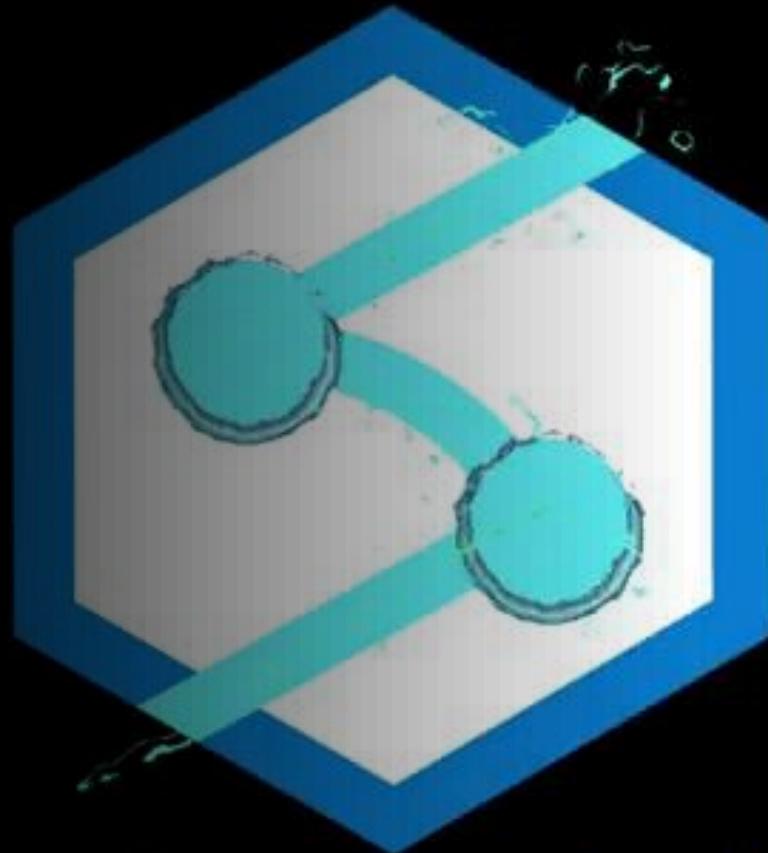
Power BI



Quiz!

Poll Question

<https://geni.us/SynapsePrecon3>



Azure Synapse Analytics



Integration



Composing



Orchestrating



Migrating data

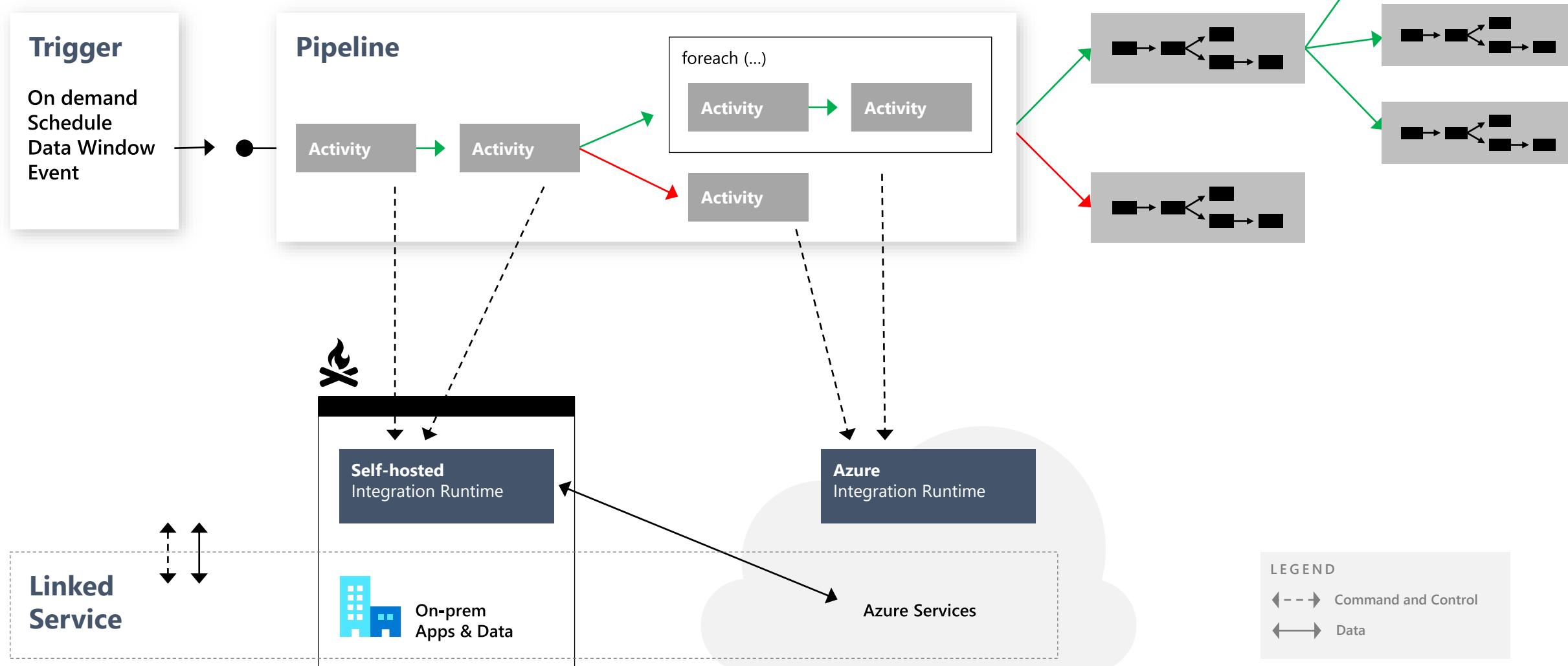
Can choose to migrate it all in Spark

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



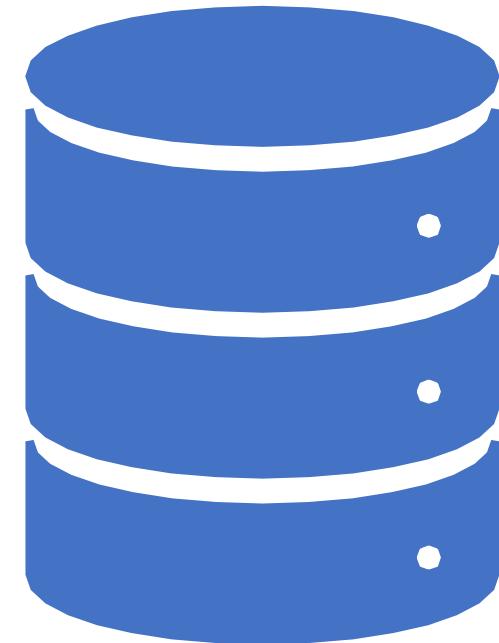
Un-check USE TYPE DEFAULT, it is not a best practice.



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

Ingest Flat files to tables

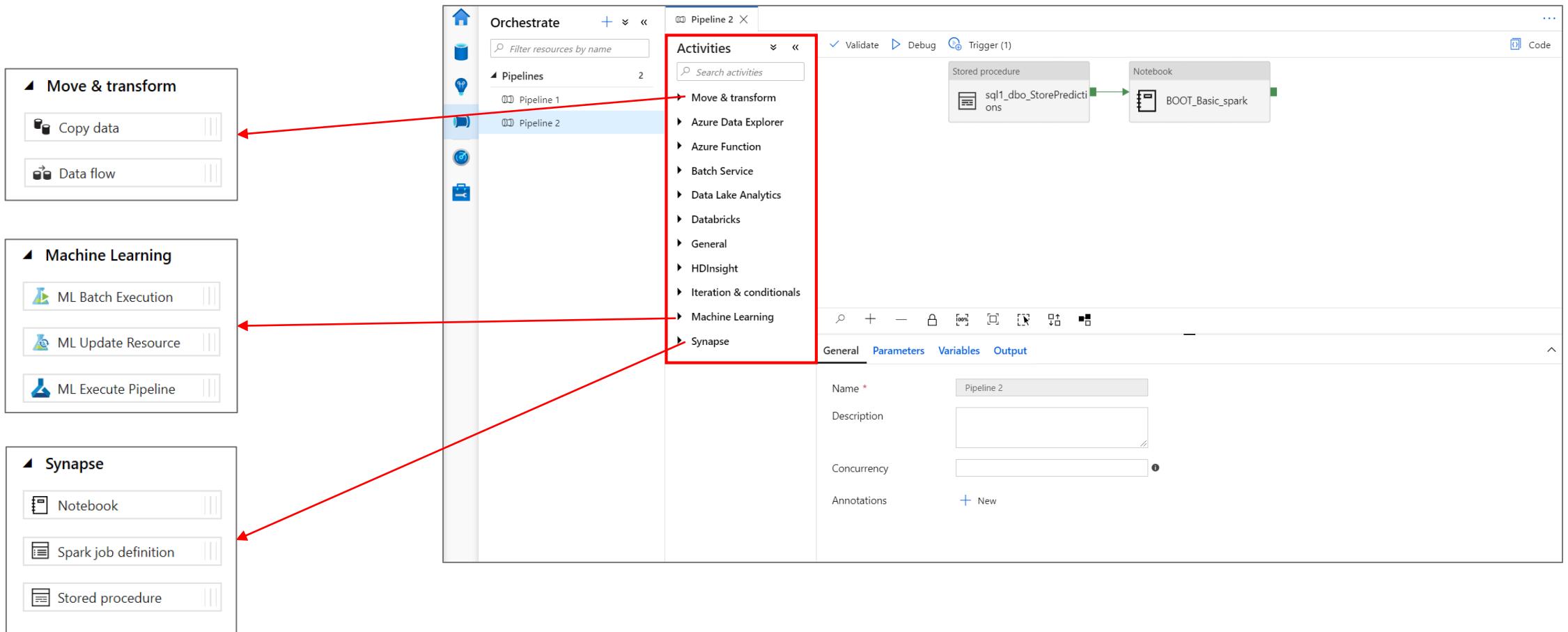
- Ingest flat file data into Azure Storage (Azure Data Lake Store Gen2)
 - When your data sources are on-premises, you need to move the data to Azure Storage before ingestion.
 - Data in other cloud platforms needs to be moved to Azure Storage before ingestion.
- Load from flat files as relational tables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

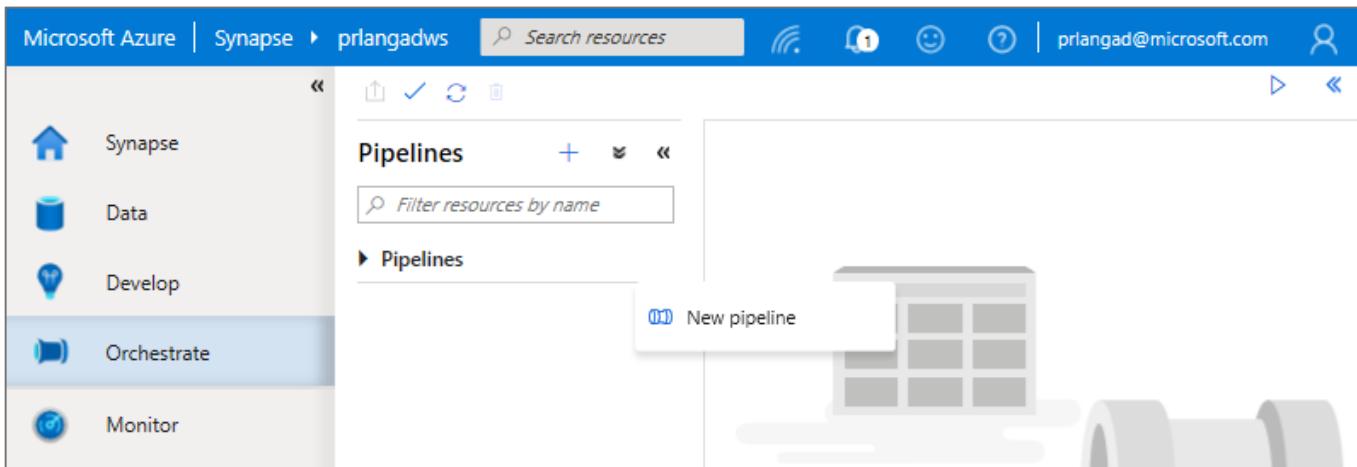
Load data by manual execution of pipeline or by orchestration.

Benefits

Supports common loading patterns.

Fully parallel loading into data lake or SQL tables.

Graphical development experience.



A screenshot of the Microsoft Azure Synapse Analytics Pipelines interface. The top navigation bar shows 'Microsoft Azure | Synapse Analytics > prlangadws2'. The left sidebar shows 'Orchestrate' selected, with 'Pipelines' expanded, showing 'Copy Open Dataset' and 'Load Data to SQLDW'. The main area shows a 'Load Data to S...' pipeline with a 'Copy data' activity named 'WeatherData'. The 'Source' tab of the activity configuration is visible, showing 'Source dataset' set to 'ADLSGen2So'. To the right, there is a grid of 'New dataset' options including Azure Cosmos DB, Azure Data Explorer, Azure Data Lake Storage Gen1, Azure Data Lake Storage Gen2, Azure Database for MariaDB, Azure Database for MySQL, Azure Database for PostgreSQL, Azure File Storage, Azure SQL Database, Azure SQL Database Managed Instance, Azure Synapse Analytics (formerly SQL DW), and Azure Table Storage.

Integration runtimes

Overview

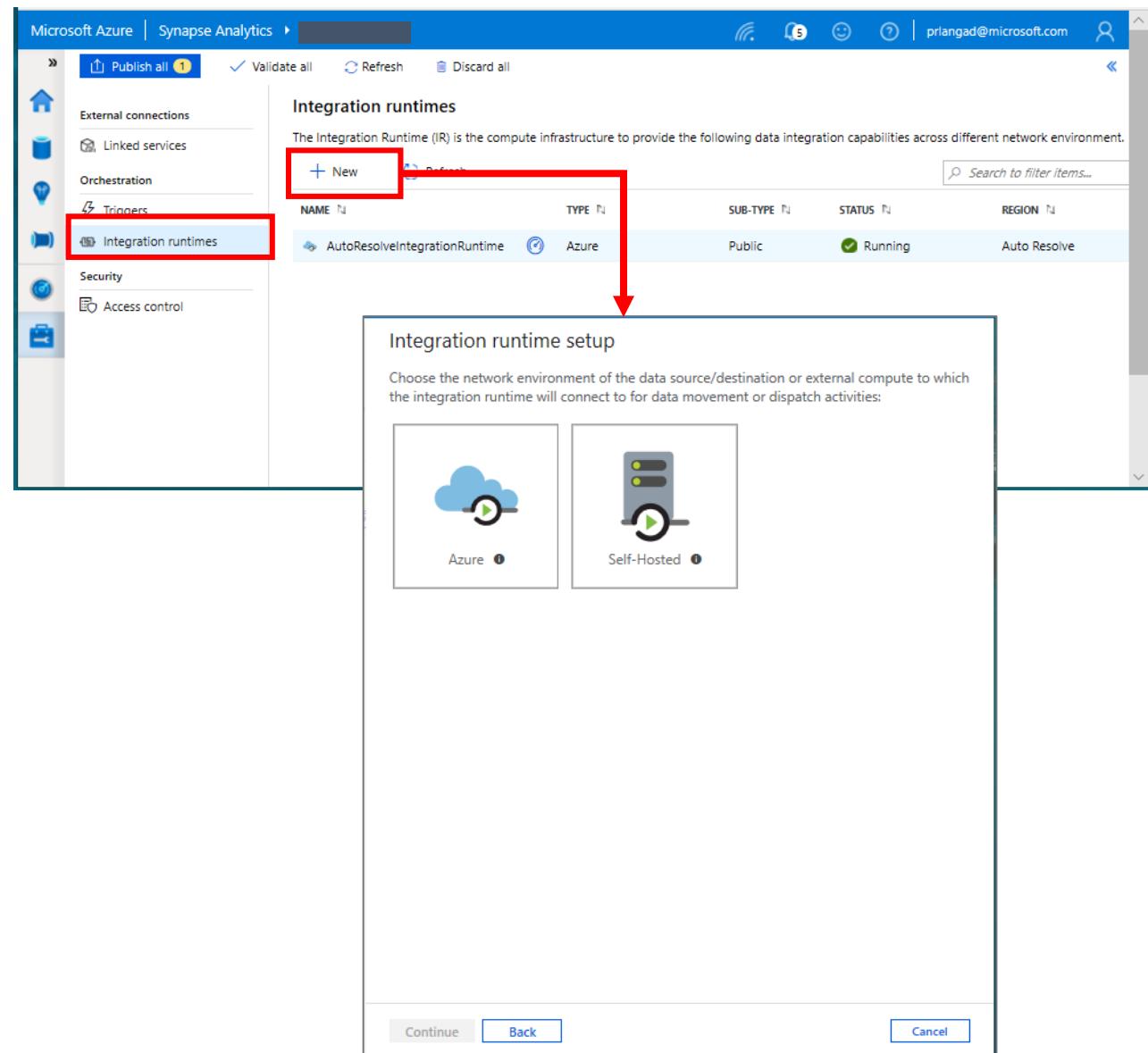
Integration runtimes are the compute infrastructure used by Pipelines to provide the data integration capabilities across different network environments. An integration runtime provides the bridge between the activity and linked services.

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

Microsoft Azure | Synapse Analytics

External connections

Linked services

Orchestration

Triggers

Integration runtimes

Security

Access control

Linked services

Linked services are much like connection strings, which define the connection information needed for Arcadia to connect to external resources.

+ New

Search to filter items...

NAME	TYPE	ANNOTATIONS
ADLSG2OpenDataSetSink	Azure Data Lake Storage Gen2	
AzureBlobStorage1	Azure Blob Storage	
AzureDataLakeStorage1	Azure Data Lake Storage Gen2	
AzureDataLakeStorage2Source		
AzureOpenDataset		
AzureOpenDataSet2		
AzureSqlDW1		

New linked service

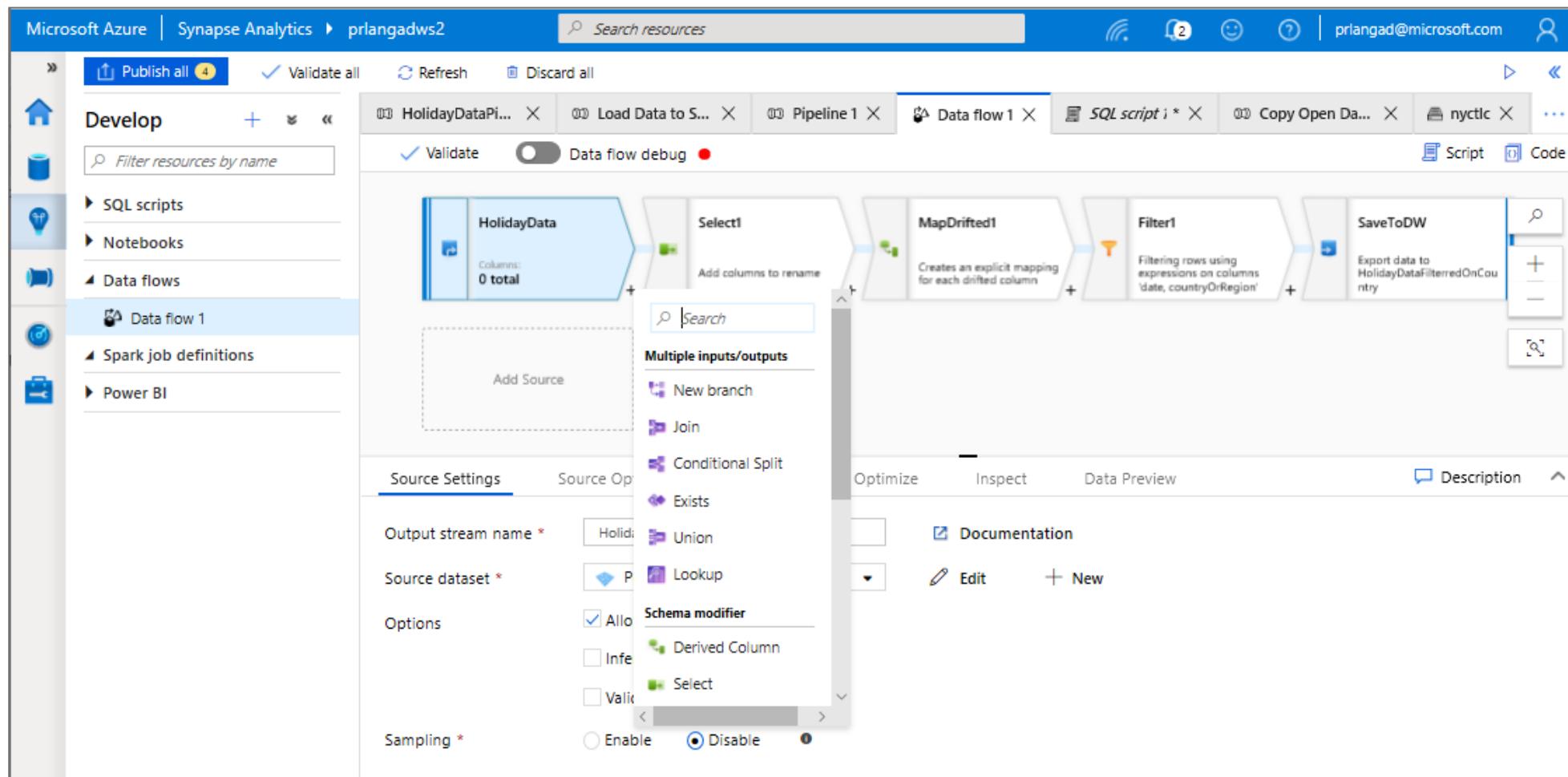
PayPal (Preview)	Phoenix	PostgreSQL
Power BI	Presto (Preview)	QuickBooks (Preview)
REST	SAP BW Open Hub	SAP BW via MDX
SAP Cloud For Customer	SAP ECC	SAP HANA
SAP	SAP	SAP

Continue Cancel

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.

The screenshot shows two windows from the Microsoft Azure Synapse Analytics portal. The top window is a 'New trigger' dialog with the following fields:

- Name: Trigger 1
- Description: (empty)
- Type: Schedule (radio button selected)
- Start Date (UTC): 10/30/2019 11:20 PM
- Recurrence: Every 1 Minute(s)
- End: No End (radio button selected)
- Annotations: + New
- Activated: Yes (radio button selected)

The bottom window shows the 'Triggers' management page with the following details:

NAME	TYPE	STATUS
* CopyParquetDataTrigger	Schedule	Started
* Trigger 1	Schedule	Stopped

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.

The screenshot shows the Azure Data Explorer interface. On the left, there is a sidebar titled "Data" with a search bar and a list of resources: "Storage accounts" (2), "Databases" (3), "Datasets" (2), "CabDataCooked", and "NYCTaxiParquet". The "NYCTaxiParquet" item is highlighted with a red box and has a red arrow pointing to the main content area. The main content area is titled "NYCTaxiParquet X" and contains a "Parquet" icon and the text "NYCTaxiParquet". Below this, there are tabs for "General", "Connection", "Schema", and "Parameters". The "Connection" tab is selected. It shows a "Linked service" dropdown set to "Lake_ArcadiaLake", a "Test connection" button, and "Open" and "New" buttons. The "File path" field is set to "data / nyctaxi / File", with "Browse" and "Preview data" buttons. The "Compression type" is set to "snappy".

Data Movement

Scalable

per job elasticity

Up to 4 GB/s

Simple

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

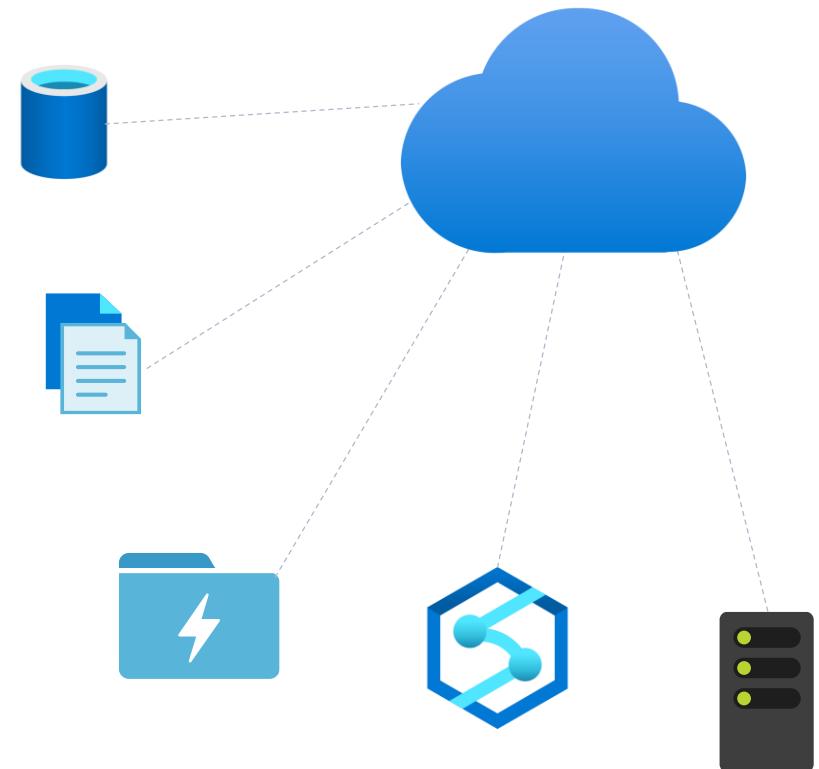
Serverless, no infrastructure to manage

Access all your data

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

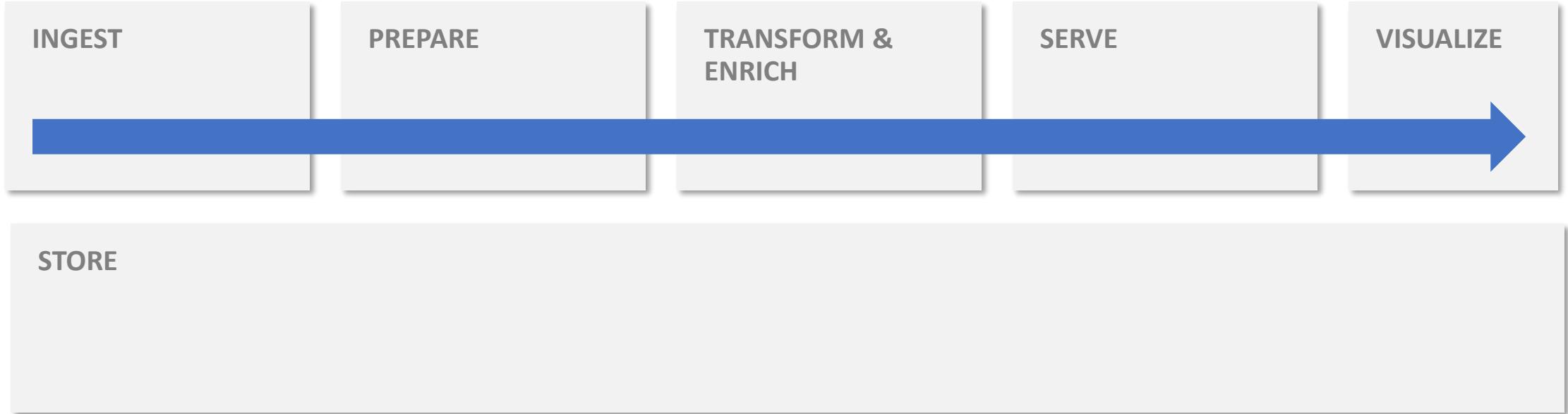
Data Movement as a Service: 25 points of presence worldwide

Self-hostable Integration Runtime for hybrid movement



90+ Connectors out of the box

Modern Data Warehouse



Demonstration

Integration

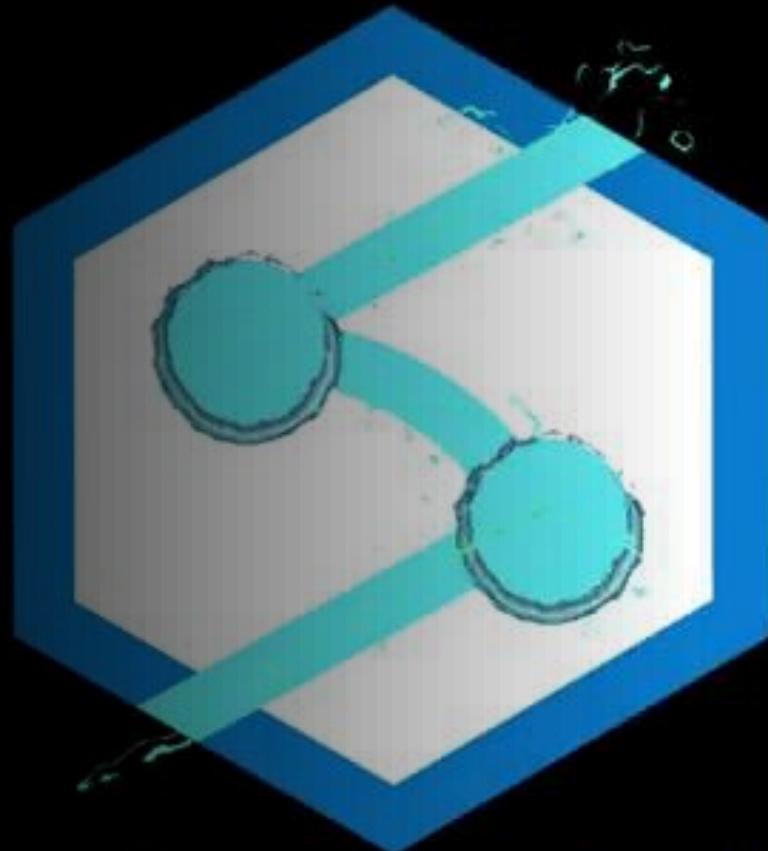




Quiz!

Integration

<https://geni.us/SynapsePrecon4>



Azure Synapse Analytics

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

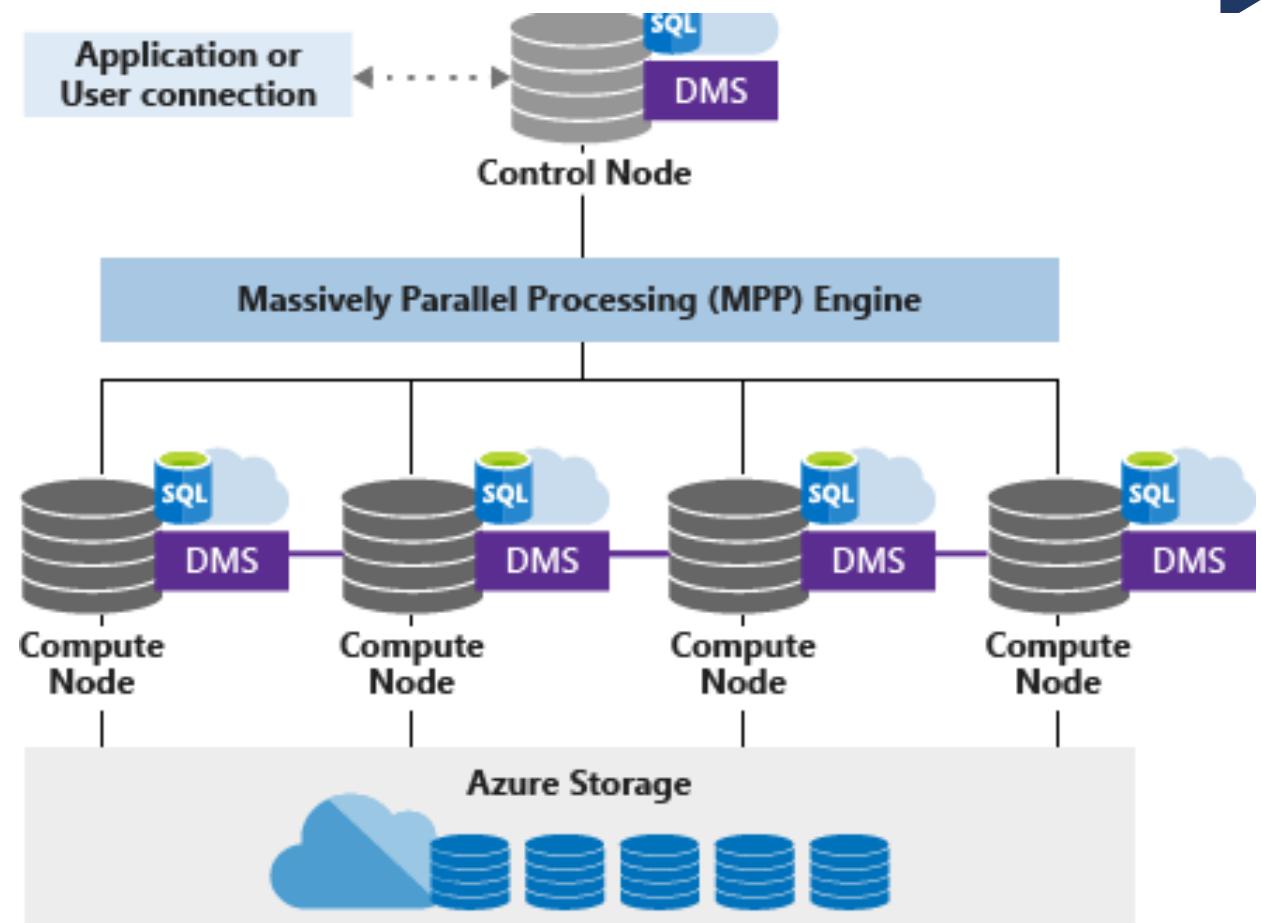
Sized by Data Warehousing Units (DTUs)

Import data with Polybase

Partitioning 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

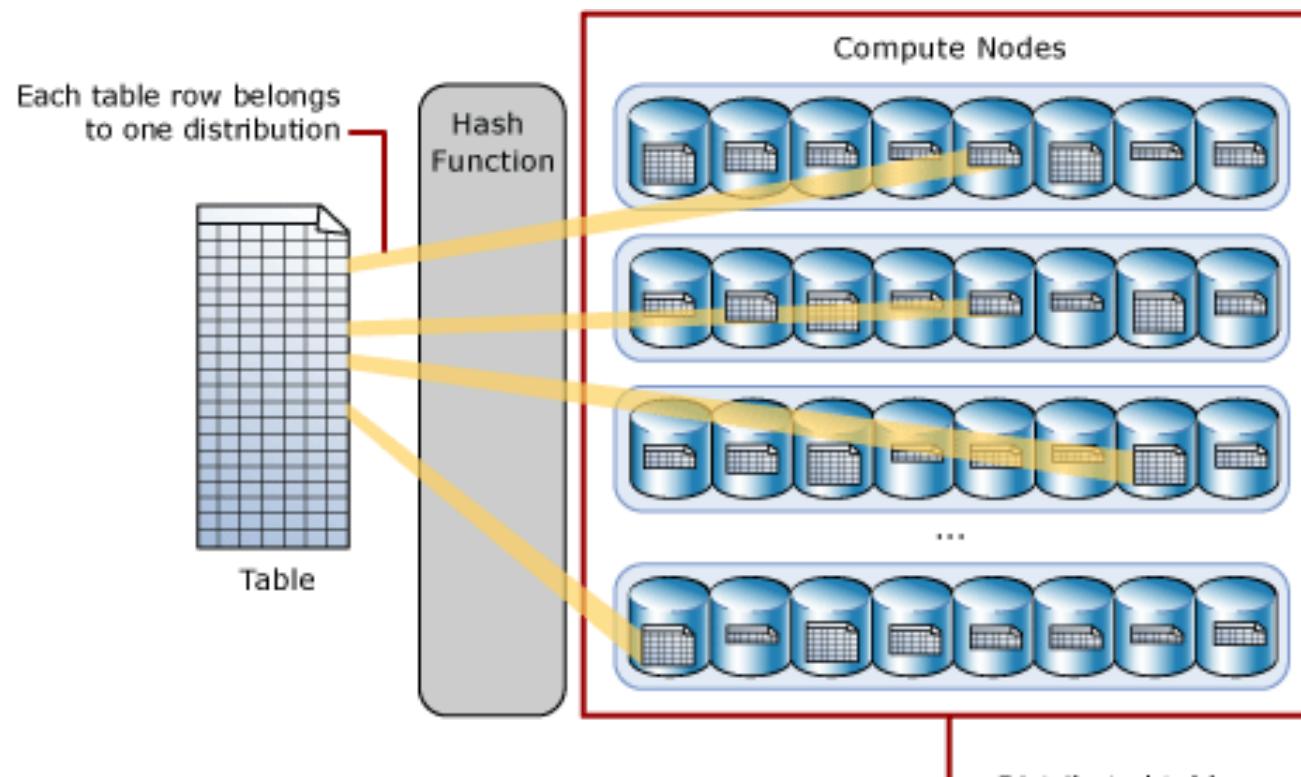


Round Robin



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

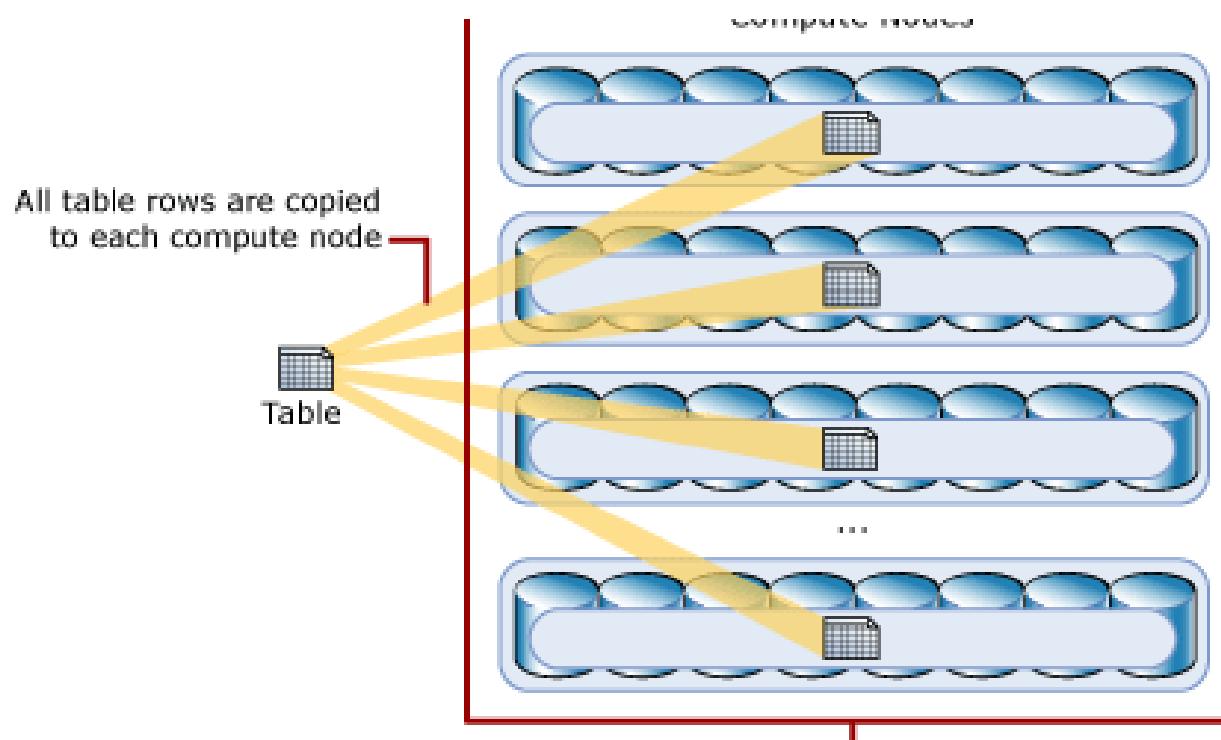
Fast
Performance
for staging
load tables

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

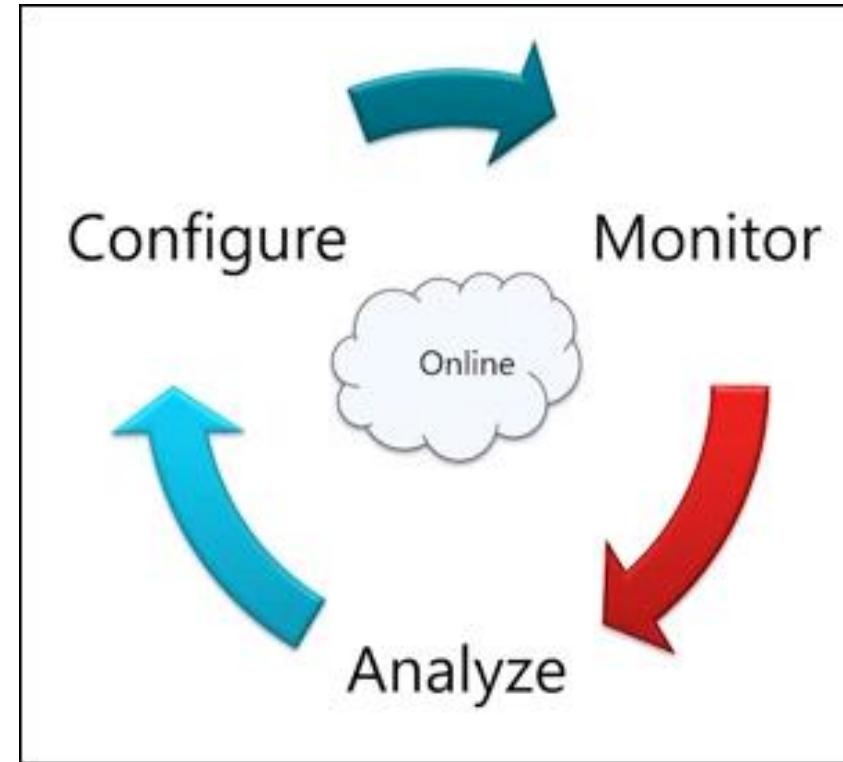
Extra storage is required for large tables

Workload Management

Classification

Importance

Isolation



Workload Classification

GroupName

Members

Importance

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

Workload Classifier



User



Role



Label



Context



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



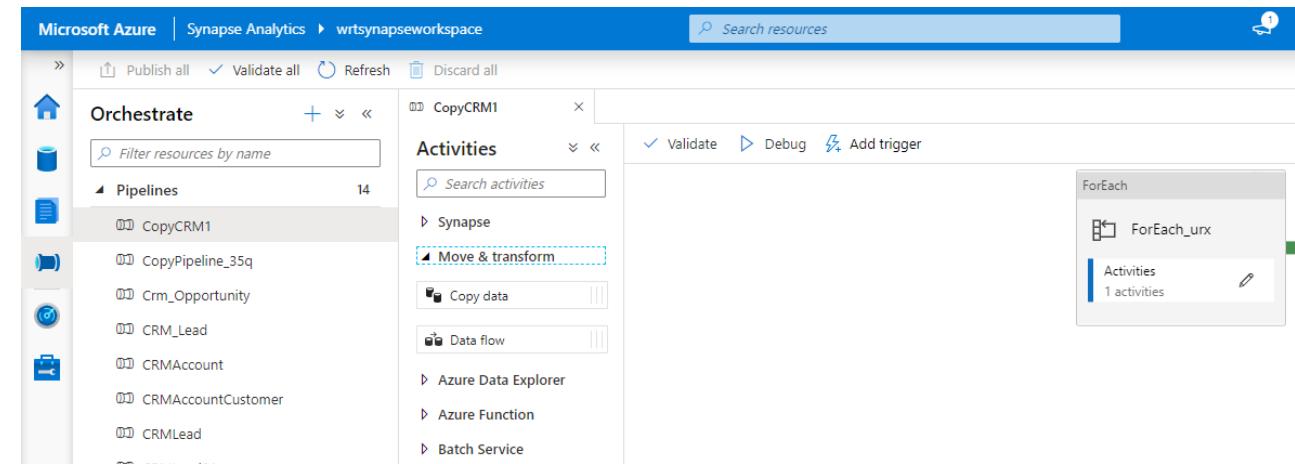
Loading data into SQL Pools

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

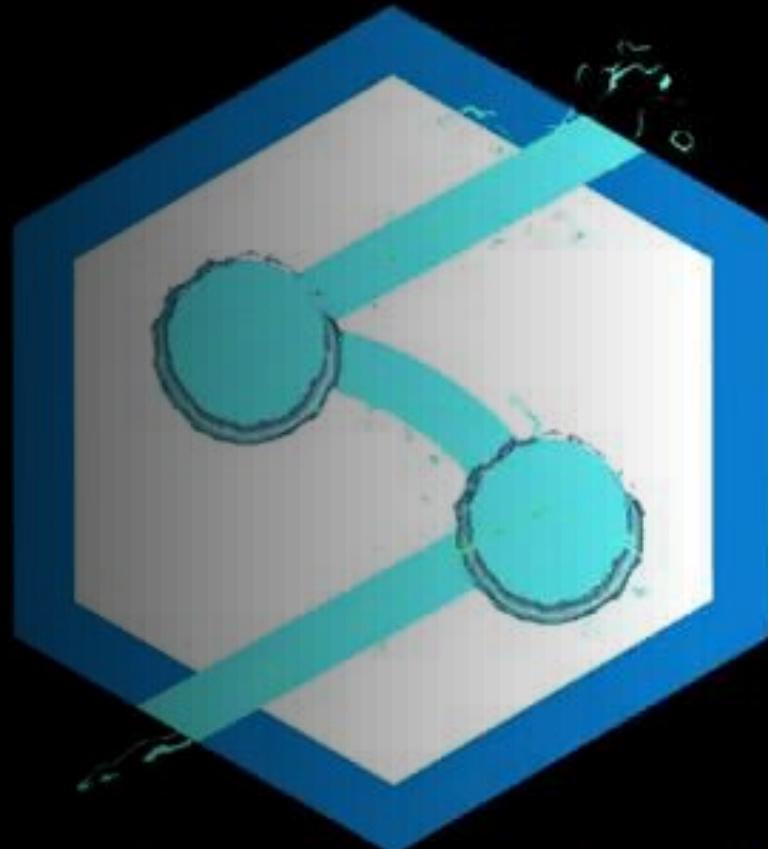
SQL Pools



Quiz!

Poll Question

<https://geni.us/SynapsePrecon5>



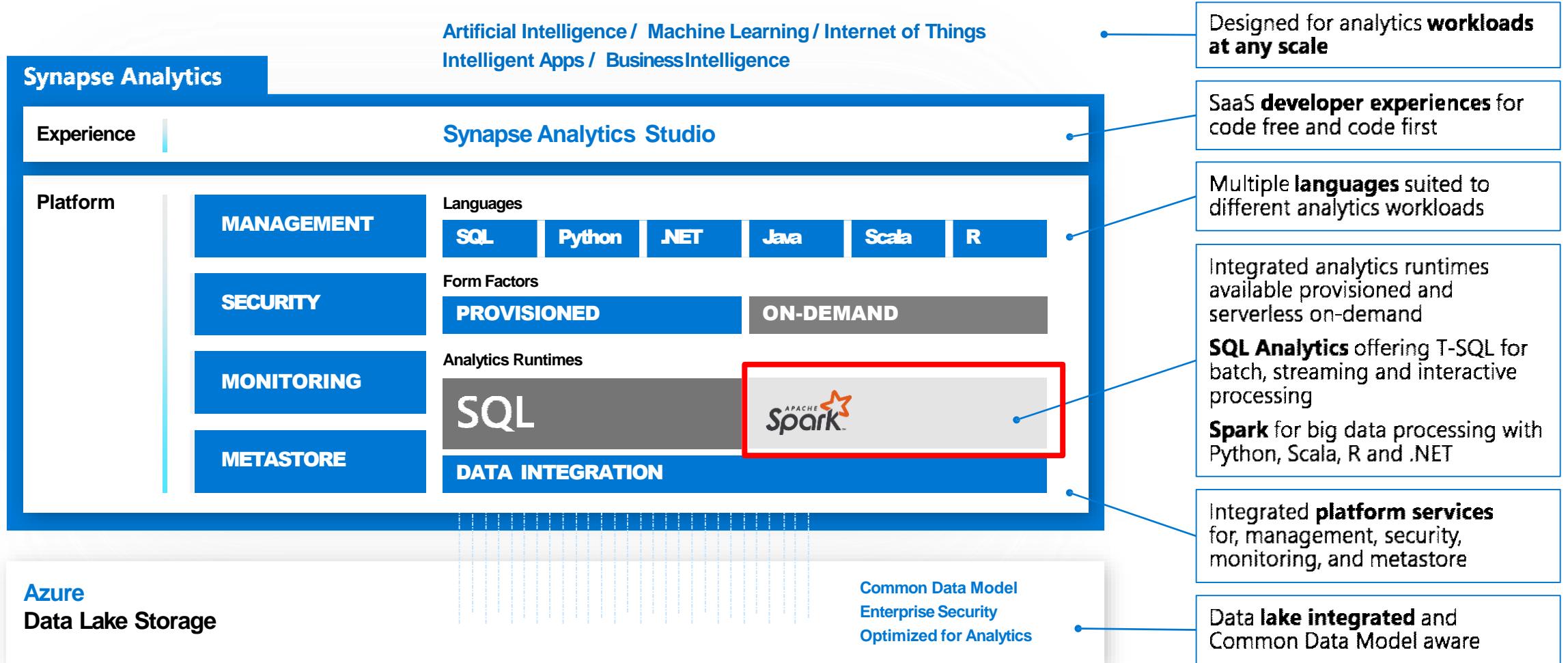
Azure Synapse Analytics



Apache Spark



Azure Synapse Analytics





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

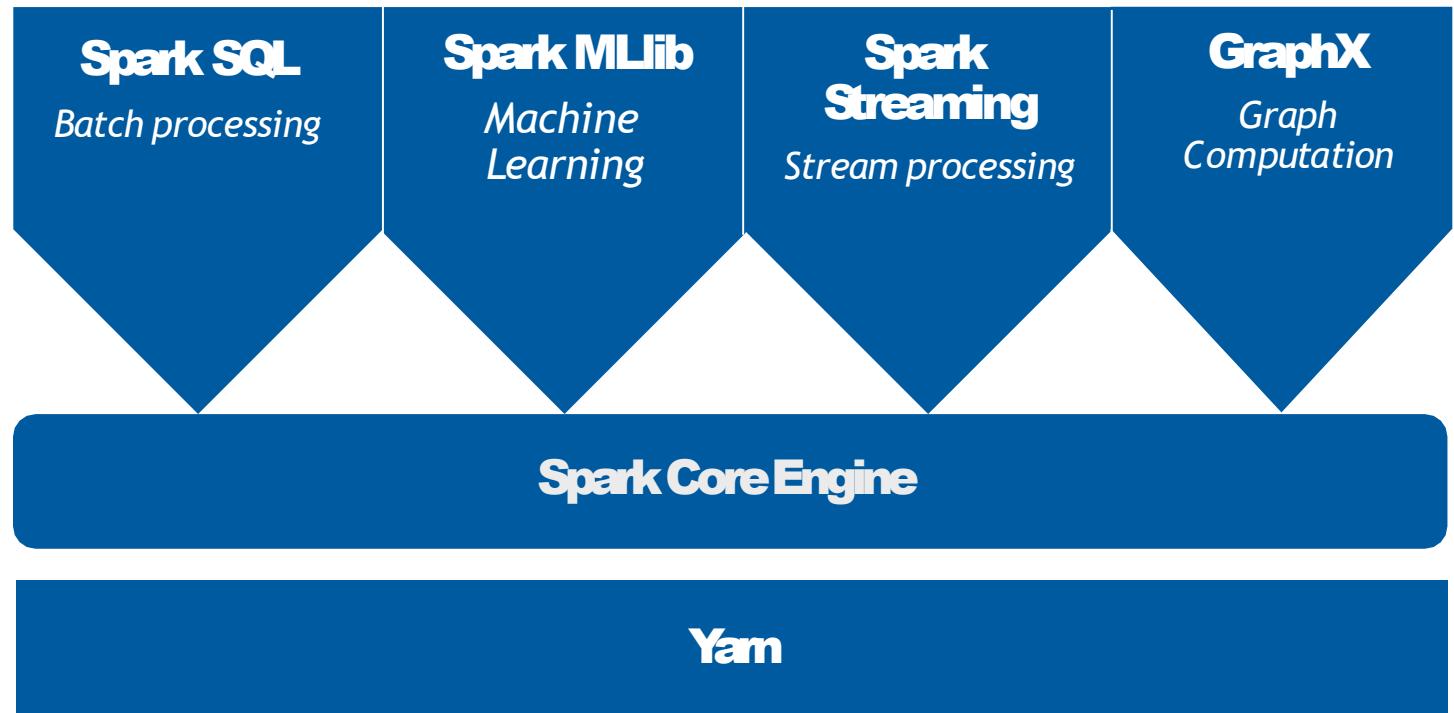
- Fast Start
- Auto scale (up and down)
- Auto pause
- Min cluster size of 3 nodes

Apache Spark

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

Spark Unifies:

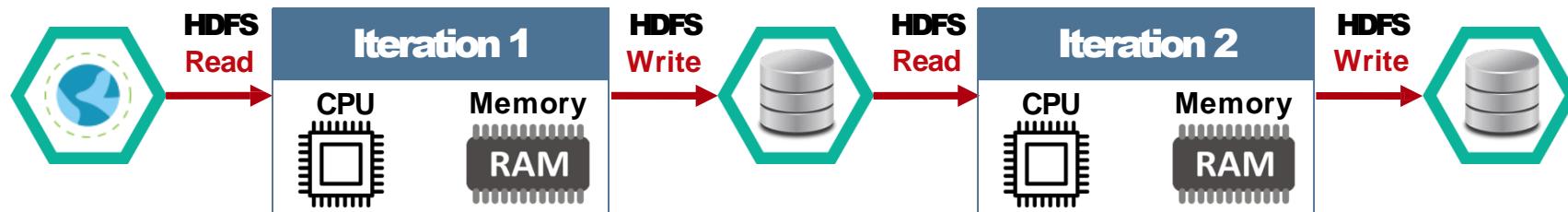
- Batch Processing
- Interactive SQL
- Real-time processing
- Machine Learning
- Deep Learning
- Graph Processing



<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

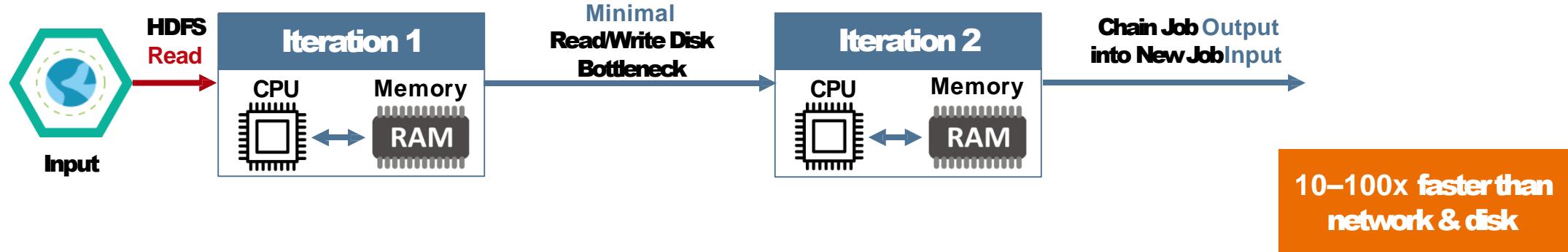


Motivation for ApacheSpark

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

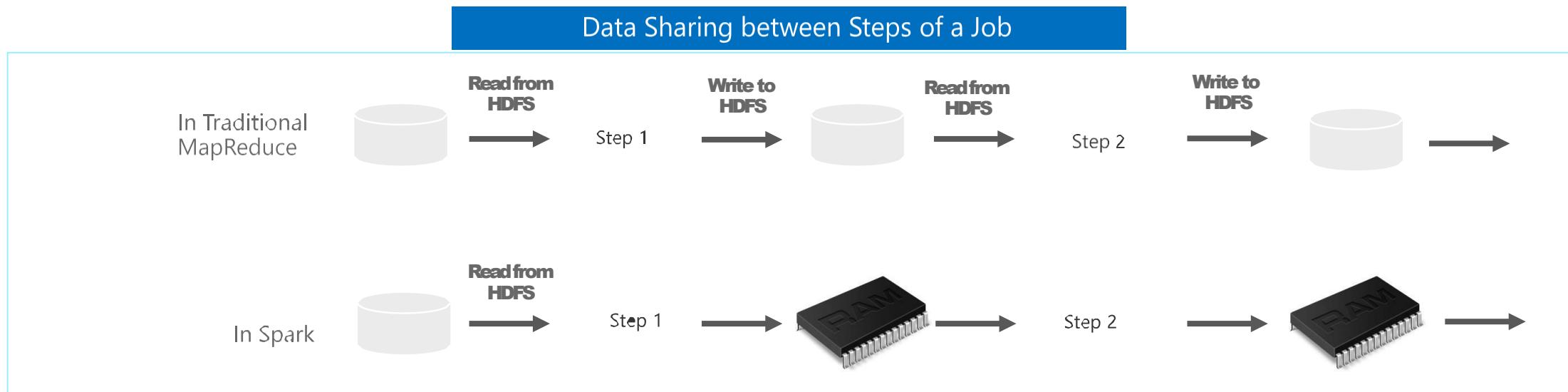


Solution: Keep data in-memory with a new distributed execution engine



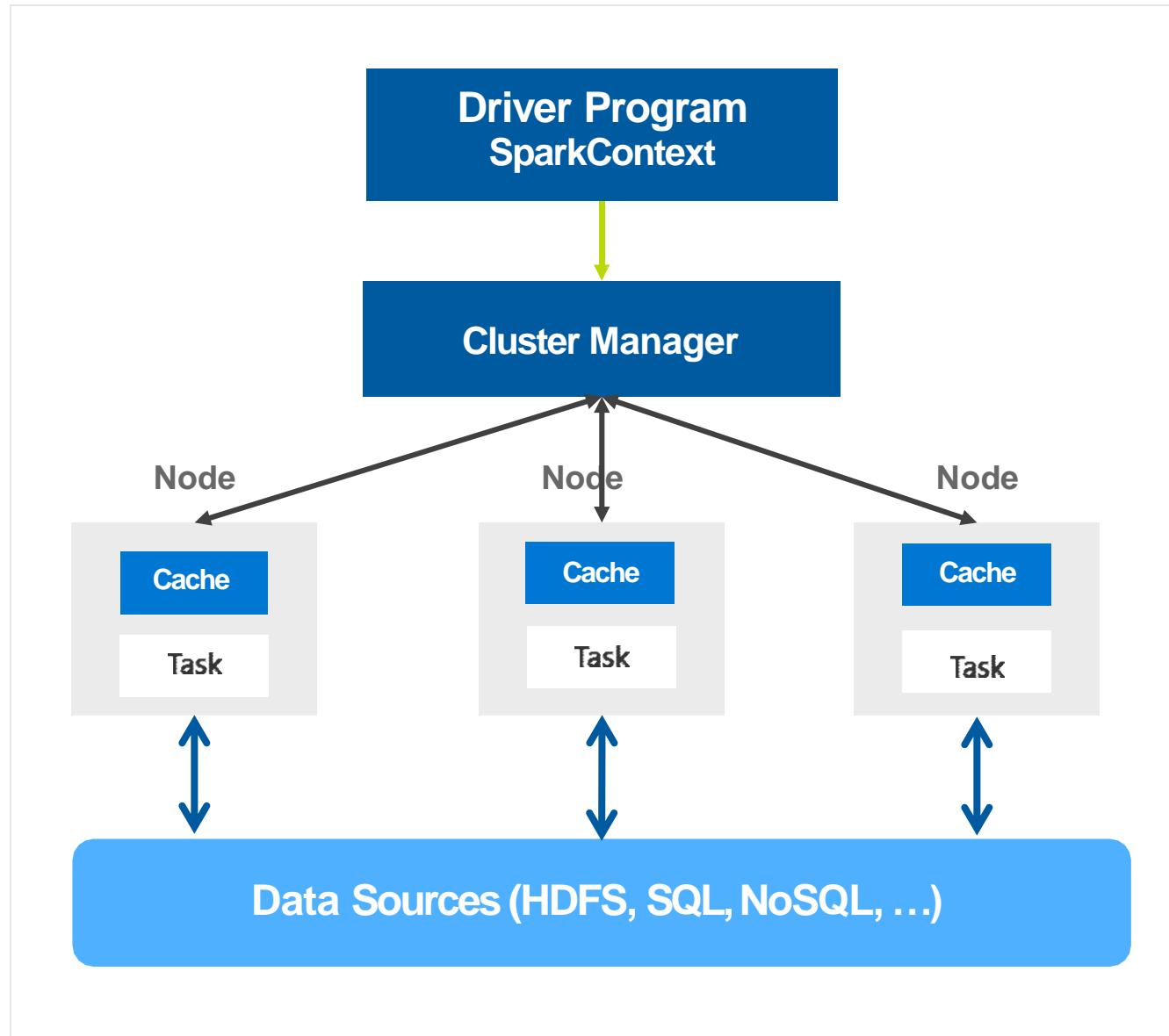
What makes Spark fast

- **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

Mlib/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

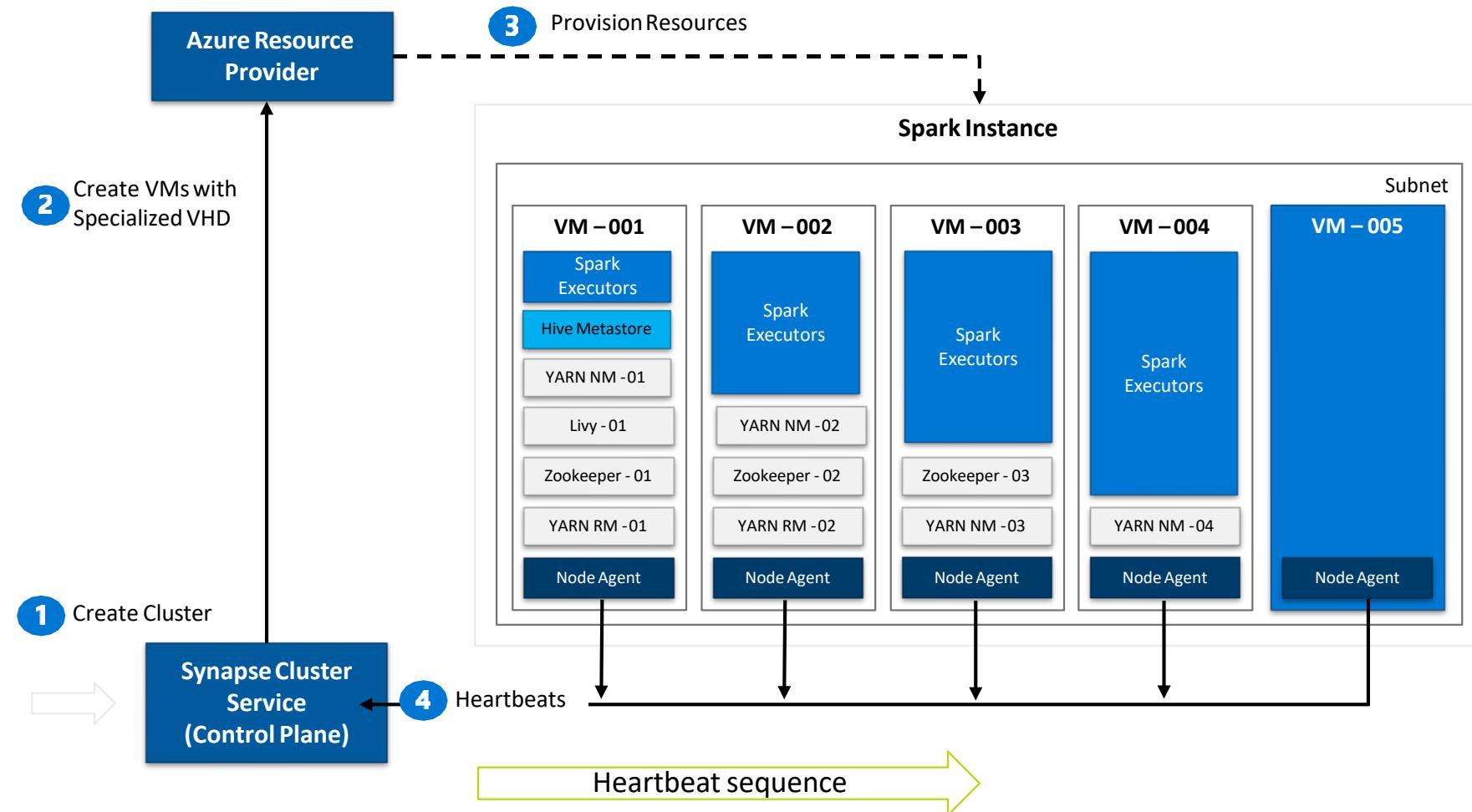
Spark Streaming

- 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 endpoint 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

Home > Synapse workspaces > euang-synapse-nov-ws - Apache Spark pools > Create Apache Spark pool

Create Apache Spark pool

Basics * Additional settings * Tags Summary

Create a Synapse Analytics Apache Spark pool with your preferred configurations. Complete the Basics tab then go to Review + create to provision with smart defaults, or visit each tab to customize.

Apache Spark pool details

Name your Apache Spark pool and choose its initial settings.

Apache Spark pool name *

Enter Apache Spark pool name

Node size family

MemoryOptimized

Node size *

Medium (8 vCPU / 64 GB)

Autoscale *

Enabled

Number of nodes *

3

40

Optional settings for Creating a Spark pool

Additional Settings offer optional settings to customize Spark pool

Customize component versions, auto-pause

Home > Synapse workspaces > euang-synapse-nov-ws - Apache Spark pools > Create Apache Spark pool

Create Apache Spark pool

Basics * Additional settings * Tags Summary

Customize additional configuration parameters including autoscale and component versions.

Auto-pause

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

Auto-pause * ⓘ Enabled Disabled

Number of minutes idle *

Component versions

Select the Apache Spark version for your Apache Spark pool.

Apache Spark *	<input type="text" value="2.4"/>
Python	3.6.1
Scala	2.11.12
Java	1.8.0_222
.NET Core	3.0
.NET for Apache Spark	0.6.0
Delta Lake	0.4.0

Packages

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

File upload

Import libraries by providing text file containing library name and version



Demonstration

Spark Pools

Create Notebook on files in storage

The screenshot illustrates the process of creating a Notebook on files stored in Azure Storage. The top navigation bar shows 'Microsoft Azure | Synapse Analytics' and the workspace name 'prlangadws2'. The left sidebar lists 'Storage accounts' (including 'priLangaddemosa (Primary)' and 'nyctic'), 'Databases', and 'Datasets'. The main area displays a file browser for 'nyctic' storage, specifically navigating to 'yellow/puYear=2015/puMonth=3/part-00133-tid-211'. A red box highlights the 'New notebook' option in the context menu for the selected file. An arrow points from this menu to the bottom-right panel, which shows a running PySpark notebook titled 'Notebook 4'. The notebook contains the following code:

```
%%pyspark
data_path = spark.read.load('abfss://nyctic@priLangaddemosa.dfs.core.windows.net/yellow/puYear=2015/puMonth=3/part-00133-tid-210938564719836543-aea5b543-5e83-')
data_path.show(10)
```

The notebook output shows the first 10 rows of a Parquet file:

vendorID	tpepPickupDateTime	tpepDropoffDateTime	passengerCount	tripDistance	puLocationId	doLocationId	startLon	startLat	endLon	endLat
1	2015-02-28 23:53:18	2015-03-01 00:00:29	6	1.63	null	null	-74.0008468279297	40.73069381713867	-73.9841537475586	40.74470520019531
1	N	1	7.5	0.5	0.5	1	0.3	1.76	0.0	10.56
1	2015-03-01 19:21:05	2015-03-28 19:28:31	1	2.2	null	null	-73.97765350341797	40.763160705566406	-73.95502471923828	40.7860031127927
1	N	1	8.5	0.0	0.5	1	0.3	2.3	0.0	11.6
1	2015-02-28 23:53:19	2015-03-01 00:12:08	5	3.23	null	null	-73.96012878417969	40.76215744018555	-73.9881591796875	40.72818896484375
1	N	1	14.5	0.5	0.5	1	0.3	4.74	0.0	20.54
1	2015-03-01 19:21:05	2015-03-28 19:37:02	1	2.1	null	null	-73.98143005371094	40.7815055847168	-74.000891552734375	40.76177215576172

Microsoft Azure Synapse Analytics > euang-synapse-nov-ws

Search resources

Publish all Validate all Refresh Discard all

Develop

Notebooks 13

- 00_DataPrep
- 01_TrainingUseMllib_cleanup
- automl_arclad_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 Download... * NYCTaxi_Docs... * SeattleSafetyD... * Repro *

Cell 1

```

1 # Azure storage access info
2 blob_account_name = "azureopendatastorage"
3 blob_container_name = "citydatacontainer"
4 blob_relative_path = "Safety/Release/city=Seattle"
5 blob_sas_token = r""
6
7 # Allow SPARK to read from Blob remotely
8 wasbs_path = 'wasbs://{}@{}.blob.core.windows.net/{}'.format(blob_container_name, blob_account_name, blob_relative_path)
9 spark.conf.set('fs.azure.sas.{}.blob.core.windows.net'.format(blob_container_name), blob_sas_token)
10
11 # SPARK read parquet, note that it won't load any data yet
12 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

ID	DESCRIPTION	STATUS	STAGES	TASKS	SUBMISSION TIME	DURATION
▶ Job 0	parquet at NativeMethodAccessImpl.java:0	In progress	0/1 (1 active)	<div style="width: 100%;"> </div>	11/22/2019, 12:44:46 AM	9m54s

View in monitoring Spark history server

Cell 2

```

1 seasafety_df.createOrReplaceTempView('seattlesafety')

```

Command executed in 2s 835ms by euang on 11-22-2019 00:53:37.321 -08:00

Cell 3

```

[6] 1 display(spark.sql("SELECT * FROM seattlesafety LIMIT 10"))

```

Command executed in 23s 901ms by euang on 11-22-2019 00:54:07.313 -08:00

View Table Chart

dataType	dataSubtype	dateTime	category	address	latitude	longitude
Safety	911_Fire	2011-03-04T10:00:26.000Z	Aid Response	517 3rd Av	47.602172	-122.330863
Safety	911_Fire	2015-06-08T02:59:35.000Z	Trans to AMR	10044 65th Av S	47.511314	-122.252346
Safety	911_Fire	2015-06-08T21:10:52.000Z	Aid Response	Aurora Av N / N 125th St	47.719572	-122.344937
Safety	911_Fire	2007-09-17T13:03:34.000Z	Medic Response	1st Av N / Republican St	47.623272	-122.355415
Safety	911_Fire	2007-11-19T17:46:57.000Z	Aid Response	7724 Ridge Dr Ne	47.684393	-122.275254
Safety	911_Fire	2008-06-15T14:32:33.000Z	Medic Response	6940 62nd Av Ne	47.678789	-122.262227
Safety	911_Fire	2007-06-18T23:05:58.000Z	Medic Response	5107 S Myrtle St	47.538902	-122.268825
Safety	911_Fire	2005-06-06T19:23:10.000Z	Aid Response	532 Belmont Av E	47.623505	-122.324033
Safety	911_Fire	2017-03-06T19:45:36.000Z	Trans to AMR	610 1st Av N	47.624659	-122.355403
Safety	911_Fire	2017-06-23T18:21:21.000Z	Automatic Fire Alarm Resd	7711 8th Av Nw	47.685137	-122.366006

Cell 4

```

[7] 1 seasafety_df.coalesce(1).write.csv('abfss://default@euangsynapsenovstorage.dfs.core.windows.net/demodata/seattlesafety', mode='overwrite')

```

View results in table format



Microsoft Azure Synapse Analytics > euang-synapse-nov-ws

Search resources

Publish all Validate all Refresh Discard all

Data Download... NYCTaxi_Docs_... * SeattleSafetyD... * Repro * PySpark (Python)

Cell 1 [3]

```
1 # Azure storage access info
2 blob_account_name = "azurereadystorage"
3 blob_container_name = "citydatacontainer"
4 blob_relative_path = "Safety/Release/city=Seattle"
5 blob_sas_token = r""
6
7 # Allow SPARK to read from Blob remotely
8 wasbs_path = 'wasbs://%' % (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)
10
11 # 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

ID	DESCRIPTION	STATUS	STAGES	TASKS	SUBMISSION TIME	DURATION
Job 0	parquet at NativeMethodAccessImpl.java:0	In progress	0/1 (1 active)		11/22/2019, 12:44:46 AM	13m43s

View in monitoring Spark history server

Cell 2 [5]

```
1 seasafety_df.createOrReplaceTempView('seattlesafety')
```

Command executed in 2s 835ms by euang on 11-22-2019 00:53:37.321 -08:00

Cell 3

```
1 display(spark.sql('SELECT * FROM seattlesafety'))
```

Command executed in 11s 526ms by euang on 11-22-2019 00:58:21.241 -08:00

SQL support

View Table Chart

Chart type pie chart X axis column category Y axis columns longitude Aggregation COUNT Y axis label Total X axis label category

Apply Cancel

Aid Response

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 IRED 1 Unit Auto Fire Alarm Automatic Fire Alarm Resd Medic Response Trans to AMR longitude

Cell 4 [7]

```
1 seasafety_df.coalesce(1).write.csv('abfss://default@euangsypnsestorage.dfs.core.windows.net/demodata/seattlesafety', mode='overwrite')
```

View results in chart format

Microsoft Azure | Synapse Analytics > euang-synapse-nov-ws

Search resources

Publish all Validate all Refresh Discard all

Develop + <>

Filter resources by name

Notebooks 13

- 00_DataPrep
- 01_TrainingUseMllib_cleanup
- automLarcada_validate
- Data Download_GreenCab
- Data Download_HolidayData
- Data Download_Weather
- Data Download_YellowCab
- ExploreJoinAggregate
- * NYCTaxi_Docs_Final
- NYCTaxi_Docs_Final_PySpark
- Repro
- SeattleSafetyDoc
- SparkPerf

Data Download... * NYCTaxi_Docs... * Run all Publish Attach to Select Spark pool Language PySpark (Python)

10
11 # Creating a temp table allows easier manipulation during the session, they are not persisted between sessions,
12 # that write the data to storage like above.
13 sampled_taxi_df.createOrReplaceTempView("nytaxi")

Exploratory Data Analysis

Look at the data and evaluate its suitability for use in a model, do this via some basic charts focussed on tip values and relationships.

Cell 9

```
1 #The charting package needs a Pandas dataframe or numpy array do the conversion
2 sampled_taxi_pd_df = sampled_taxi_df.toPandas()
3
4 # Look at tips by amount count histogram
5 ax1 = sampled_taxi_pd_df['tipAmount'].plot(kind='hist', bins=25, facecolor='lightblue')
6 ax1.set_xlabel('Tip amount distribution')
7 ax1.set_ylabel('Counts')
8 ax1.set_ylabel('Counts')
9 plt.suptitle('')
10 plt.show()
11
12 # How many passengers tip'd by various amounts
13 ax2 = sampled_taxi_pd_df.boxplot(column=['tipAmount'], by=['passengerCount'])
14 ax2.set_title('Tip amount by Passenger count')
15 ax2.set_xlabel('Passenger count')
16 ax2.set_ylabel('Tip Amount ($)')
17 plt.suptitle('')
18 plt.show()
19
20 # Look at the relationship between fare and tip amounts
21 ax = sampled_taxi_pd_df.plot(kind='scatter', x='fareAmount', y = 'tipAmount', c='blue', alpha = 0.10, s=2.5*(sampled_taxi_pd_df['passengerCount']))
22 ax.set_title('Tip amount by Fare amount')
23 ax.set_xlabel('Fare Amount ($)')
24 ax.set_ylabel('Tip Amount ($)')
25 plt.axis([-2, 80, -2, 20])
26 plt.suptitle('')
27 plt.show()
```

Tip amount distribution

Tip amount by Passenger count

Tip amount by Fare amount

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

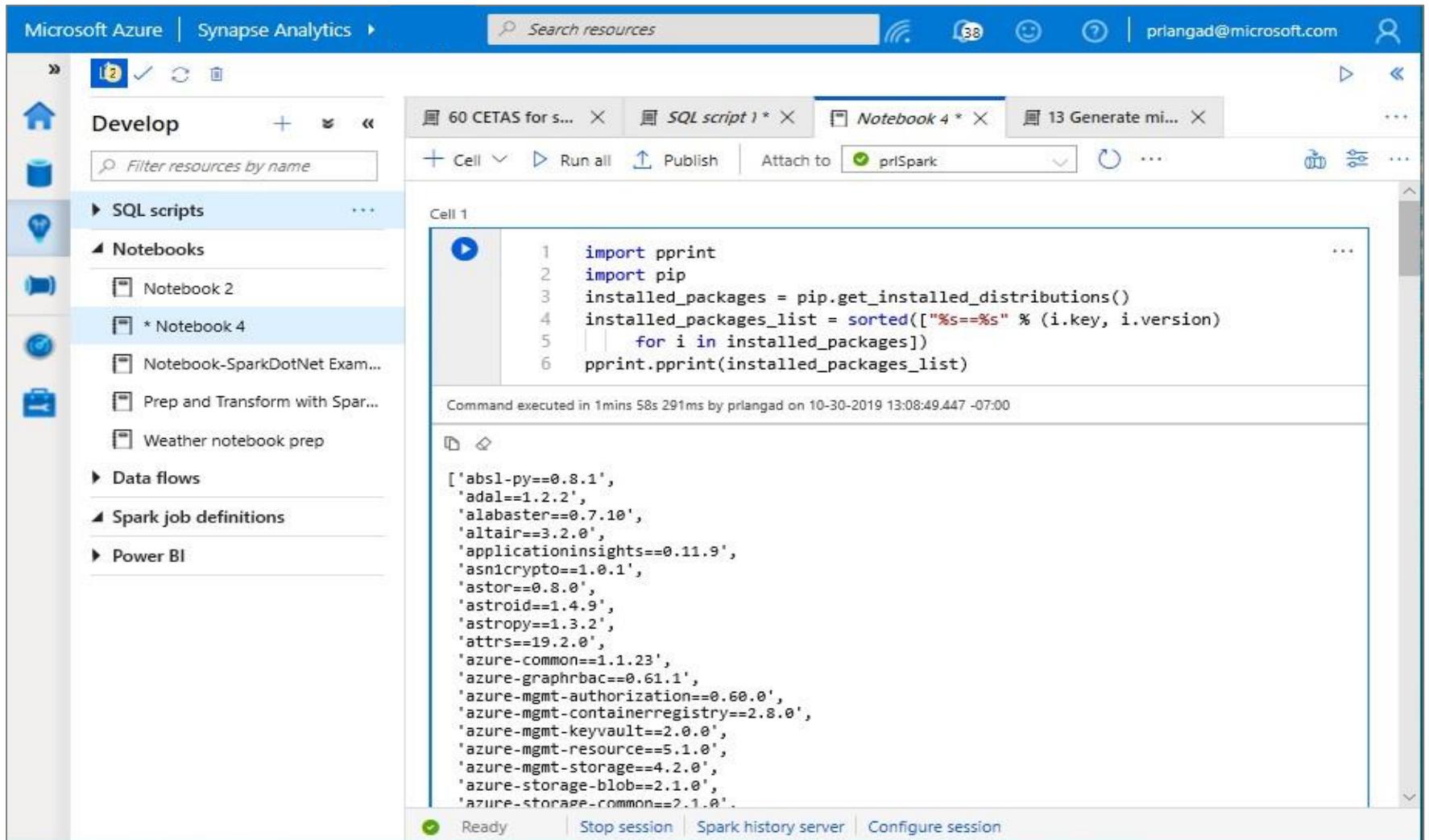
The library version must exist on PyPI repository Version downgrade of an existing library not allowed

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

The screenshot shows the 'Create Apache Spark pool' blade in the Microsoft Azure portal. At the top, there are tabs for 'Restore default configuration' and 'Report a bug'. Below that, the breadcrumb navigation shows 'Home > nushuklasynapsewestus2 > Create Apache Spark pool'. The main form has sections for 'Auto-pause' (set to 'Enabled'), 'Number of minutes idle' (set to 15), 'Component versions' (Apache Spark 2.4, Python 3.6.1, Scala 2.11.12, Java 1.8.0_222, .NET Core 3.0, .NET for Apache Spark 0.6.0, Delta Lake 0.4.0), and a 'Packages' section. The 'Packages' section is highlighted with a red border and contains a 'File upload' input field with the value 'requirements.txt' and a 'Upload' button. At the bottom, there are buttons for 'Review + create', '< Previous', and 'Next: Tags >'.

Library Management - Python

Get list of installed libraries with version information



The screenshot shows the Microsoft Azure Synapse Analytics workspace interface. On the left, the 'Develop' sidebar is open, showing a list of resources: SQL scripts, Notebooks, Data flows, Spark job definitions, and Power BI. Under 'Notebooks', 'Notebook 4' is selected. In the main area, a Python notebook is running. The code in Cell 1 is:

```
1 import pprint
2 import pip
3 installed_packages = pip.get_installed_distributions()
4 installed_packages_list = sorted(["%s==%s" % (i.key, i.version)
5 | | for i in installed_packages])
6 pprint.pprint(installed_packages_list)
```

The output of the code is displayed below the code block, showing a long list of installed Python packages and their versions. The command was executed in 1 minute 58 seconds on 10-30-2019 at 13:08:49.447 -07:00.

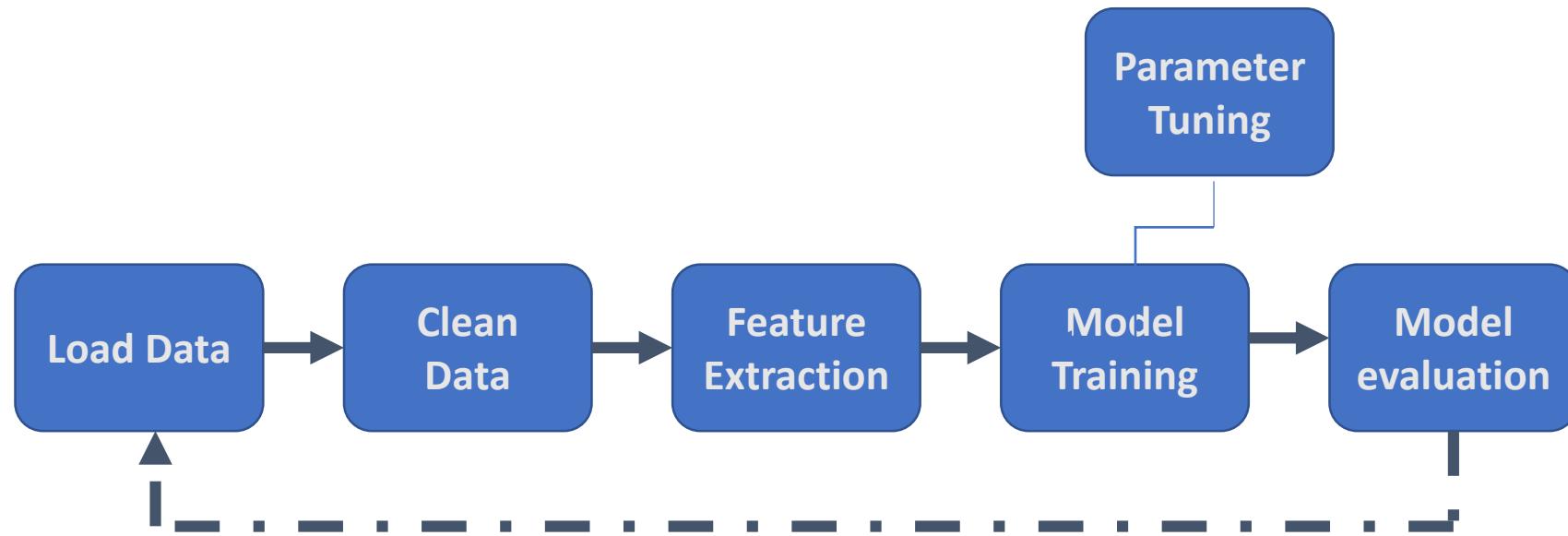
At the bottom of the notebook interface, there are buttons for 'Ready', 'Stop session', 'Spark history server', and 'Configure session'.

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

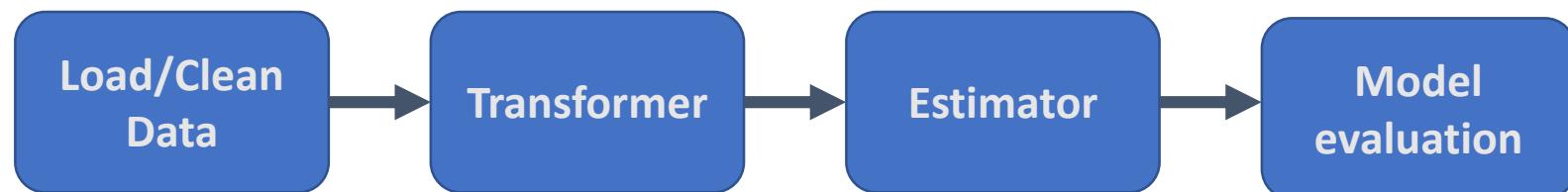
Spark ML Algorithms

Spark ML Algorithms

Classification and Regression	<ul style="list-style-type: none">• Linear Models (SVMs, logistic regression, linear regression)• Naïve Bayes• Decision Trees• Ensembles of trees (Random Forest, Gradient-Boosted Trees)• Isotonic regression
Clustering	<ul style="list-style-type: none">• k-means and streaming k-means• Gaussian mixture• Power iteration clustering (PIC)• Latent Dirichlet allocation (LDA)
Collaborative Filtering	<ul style="list-style-type: none">• Alternating least squares (ALS)
Dimensionality Reduction	<ul style="list-style-type: none">• SVD• PCA
Frequent Pattern Mining	<ul style="list-style-type: none">• FP-growth• Association rules
Basic Statistics	<ul style="list-style-type: none">• 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 hyperparameters for the model
- 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

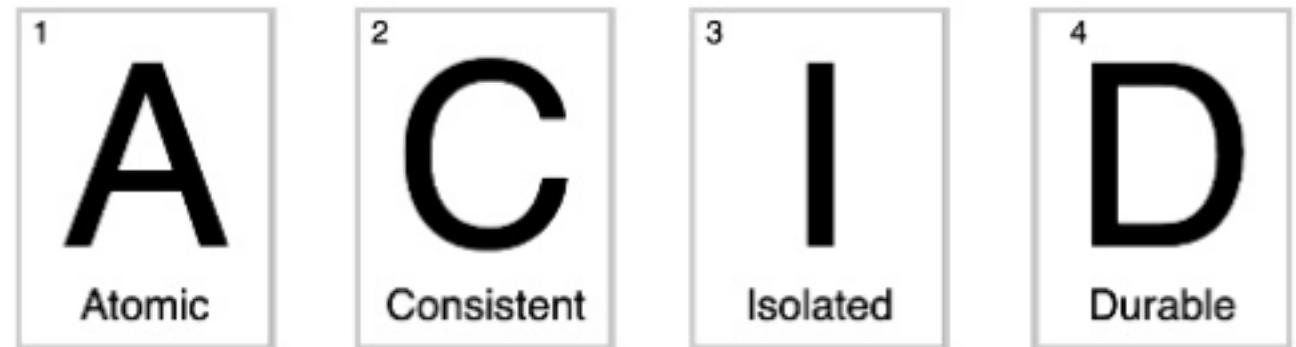


Delta Lake Feature List

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

The screenshot shows a Microsoft Azure Synapse Analytics notebook interface. The left sidebar is titled "Develop" and lists "SQL scripts", "Notebooks", "Data flows", "Spark job definitions", and "Power BI". A red arrow points from the text "Simple code to connect workspace" to the code in Cell 5.

Check the Azure ML Core SDK Version to Validate Your Installation

Cell 3

```
[5] 1 import azureml.core  
2 print("SDK Version:", azureml.core.VERSION)
```

Command executed in 1s 258ms by balapv on 11-12-2019 14:41:52.805 -08:00

SDK Version: 1.0.69

Connect to Azure Workspace

Cell 5

```
[6] 1 ## Import the Workspace class and check the Azure ML SDK version.  
2 from azureml.core import Workspace  
3  
4 ws = Workspace(subscription_id = "6560575d-fa06-4e7d-95fb-f962e74efd7a",  
5 | | | | | resource_group = "balapv-synapse-rg", workspace_name = "AML-WS-synapse")  
6  
7 print(ws.name, ws.location, ws.resource_group, sep='\t')
```

Command executed in 3s 909ms by balapv on 11-12-2019 14:41:55.491 -08:00

AML-WS-synapse westus2 balapv-synapse-rg

Cell 6

```
[7] 1 # import modules  
2 import azureml.core  
3 import pandas as pd  
4 from azureml.core.authentication import ServicePrincipalAuthentication  
5 from azureml.core.workspace import Workspace  
6 from azureml.core.experiment import Experiment
```

Running | Stop session | Spark history server | Configure session

Synapse Notebook: Configure AML job to run on Synapse

Microsoft Azure | Synapse Analytics > synapsews4aml | Search resources

Publish all (1) | Validate all | Refresh | Discard all

automl_synapse... * X

Develop | Filter resources by name | + | << | ...

+ Cell | Run all | Publish | Attach to: sparkcompute | Language: PySpark (Python)

Train

Instantiate an AutoMLConfig object to specify the settings and data used to run the experiment.

Note: Set the parameter enable_onnx_compatible_models=True, if you also want to generate the ONNX compatible models. Please note, the forecasting task and TensorFlow models are not ONNX compatible yet.

Property	Description
task	classification or regression
primary_metric	This is the metric that you want to optimize.
iteration_timeout_minutes	Time limit in minutes for each iteration.
iterations	Number of iterations. In each iteration AutoML trains a specific pipeline with the data.
X	(sparse) array-like, shape = <code>n_samples, n_features</code>
y	(sparse) array-like, shape = <code>n_samples, n_targets</code> . Multi-class targets.
enable_onnx_compatible_models	Enable the ONNX compatible models in the experiment.
path	Relative path to the project folder. AutoML stores configuration files for the experiment under this folder. You can specify a new empty folder.

Cell 13

```
1 automl_config = AutoMLConfig(task = 'regression',
2                               debug_log = 'automl_errors.log',
3                               primary_metric = 'normalized_root_mean_squared_error',
4                               iteration_timeout_minutes = 10,
5                               iterations = 20,
6                               preprocess = True,
7                               n_cross_validations = 2,
8                               max_concurrent_iterations = 2, #spark compute size
9                               verbosity = logging.INFO,
10                              spark_context=sc, #spark related
11                              enable_onnx_compatible_models=True, # This will generate ONNX compatible models.
12                              cache_store=True,
13                              X = X_train,
14                              y = y_train)
```

Command executed in 630ms by balapv on 11-12-2019 15:03:57.443 -08:00

Call the submit method on the experiment object and pass the run configuration. Execution of local runs is synchronous. Depending on the data and the number of iterations this can run for a while. In this example, we specify show_output = True to print currently running iterations to the console.

Configuration parameters



Synapse Notebook: Run AML job

Microsoft Azure | Synapse Analytics > synapsews4aml | Search resources | Show notifications | balapv@microsoft.com

Publish all 1 | Validate all | Refresh | Discard all

Develop | Filter resources by name

SQL scripts | Notebooks | * automl_synapse_local_regr...

Data flows | Spark job definitions | Power BI

automl_synapse_local_regr... * X | Run all | Publish | Attach to sparkcompute | Language PySpark (Python)

Run AutoML job

Cell 15

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

Command executed in 12mins 34s 972ms by balapv on 11-12-2019 15:17:53.089 -08:00

Running an experiment on spark cluster: automl-local-regression-Synapse.
Parent Run ID: AutoML_ad8600ab-a1ab-4b6b-b233-059d969e0a0e

ITERATION: The iteration being evaluated.
PIPELINE: A summary description of the pipeline being evaluated.
DURATION: Time taken for the current iteration.
METRIC: The result of computing score on the fitted pipeline.
BEST: The best observed score thus far.

ITERATION	PIPELINE	DURATION	METRIC	BEST
1	StandardScalerWrapper ElasticNet	0:00:38	0.0021	0.0021
2	StandardScalerWrapper ElasticNet	0:00:32	0.0054	0.0021
0	StandardScalerWrapper ElasticNet	0:01:20	0.0004	0.0004
4	StandardScalerWrapper RandomForest	0:00:33	0.0179	0.0004
3	StandardScalerWrapper ElasticNet	0:00:36	0.0036	0.0004
5	StandardScalerWrapper LightGBM	0:00:28	0.0109	0.0004
6	MaxAbsScaler DecisionTree	0:00:34	0.0168	0.0004
7	MaxAbsScaler RandomForest	0:00:41	0.0104	0.0004
8	MaxAbsScaler DecisionTree	0:01:05	0.0077	0.0004
9	MaxAbsScaler DecisionTree	0:00:48	0.0086	0.0004
10	StandardScalerWrapper DecisionTree	0:00:39	0.0058	0.0004
11	MaxAbsScaler DecisionTree	0:00:45	0.0096	0.0004
13	MaxAbsScaler ExtremeRandomTrees	0:00:47	0.0147	0.0004
12	MaxAbsScaler ExtremeRandomTrees	0:01:54	0.0096	0.0004
14	StandardScalerWrapper ElasticNet	0:00:39	0.0027	0.0004
15	StandardScalerWrapper ElasticNet	0:00:54	0.0010	0.0004
16	StandardScalerWrapper ElasticNet	0:00:48	0.0023	0.0004
17	MaxAbsScaler ElasticNet	0:00:31	0.0239	0.0004
18	StandardScalerWrapper ElasticNet	0:00:53	0.0014	0.0004
19	VotingEnsemble	0:01:59	0.0004	0.0004

Get Azure Portal URL for Monitoring Runs

Running | Stop session | Spark history server | Configure session

ML job execution result

Demonstration

Machine Learning in Synapse

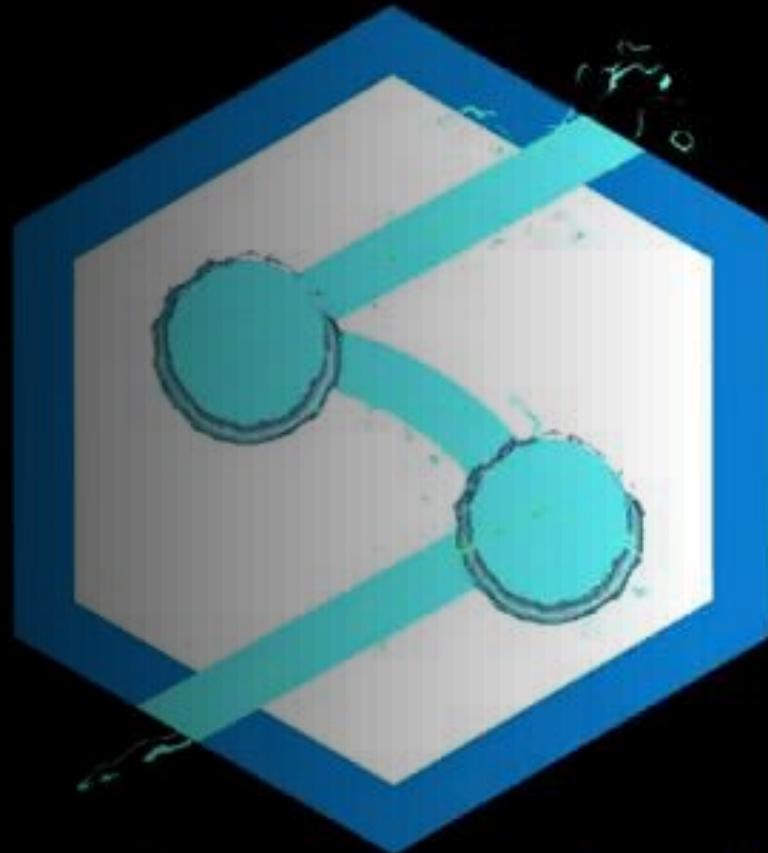




Quiz!

Poll Question

<https://geni.us/SynapsePrecon3>



Azure Synapse Analytics

Databricks differences



PROPRIETARY
VERSION OF SPARK



OPTIMIZED FOR
SPEED



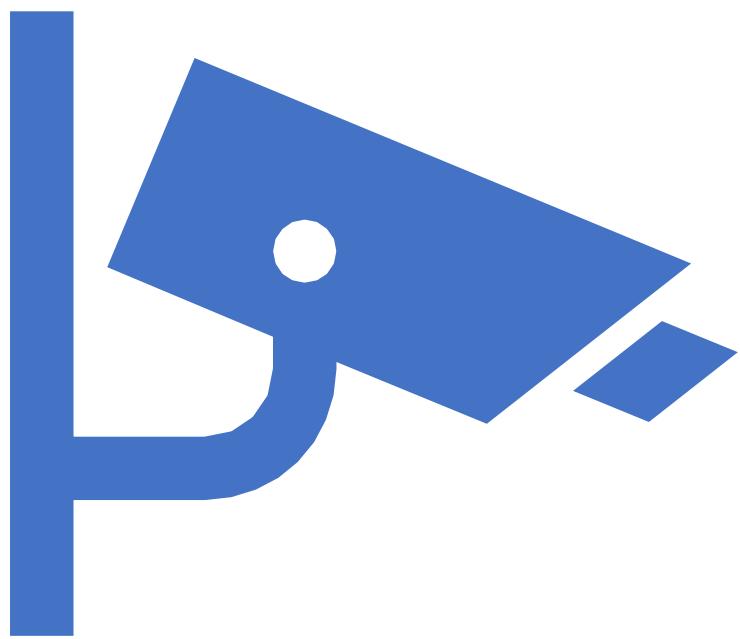
SOURCE CONTROL



CAN WRITE SPARK R
CODE



FOR MORE DETAILS,
COME TO MY SESSION
TOMORROW



Azure Synapse
Security

Azure Synapse Workspace Security

Microsoft Azure | Synapse Analytics > synapsewksp11

6 🔍 📈 🗃 ? ginger.grant@outlook.com DEFAULT DIRECTORY

> Publish all ✓ Validate all ⏪ Refresh ⏴ Discard all ⏵

Analytics pools

SQL pools

Apache Spark pools

External connections

Linked services

Orchestration

Triggers

Integration runtimes

Security

Access control

Managed private endpoints

Access control

Grant access to Synapse workspace and resources by assigning a role to a user, group, service principal, or managed identity. [Learn more](#)

[+ Add](#) ⏪ Refresh ⏴ Remove access Search to filter items...

Showing 1 - 1 of 1 items

<input type="checkbox"/> Name ↑↓	Type ↑↓	Role
Workspace admin ⓘ		
Ginger Grant	Individual	Workspace admin

Enterprise-grade security



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



CJS



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



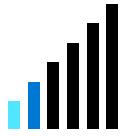
ENISA
IAF

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

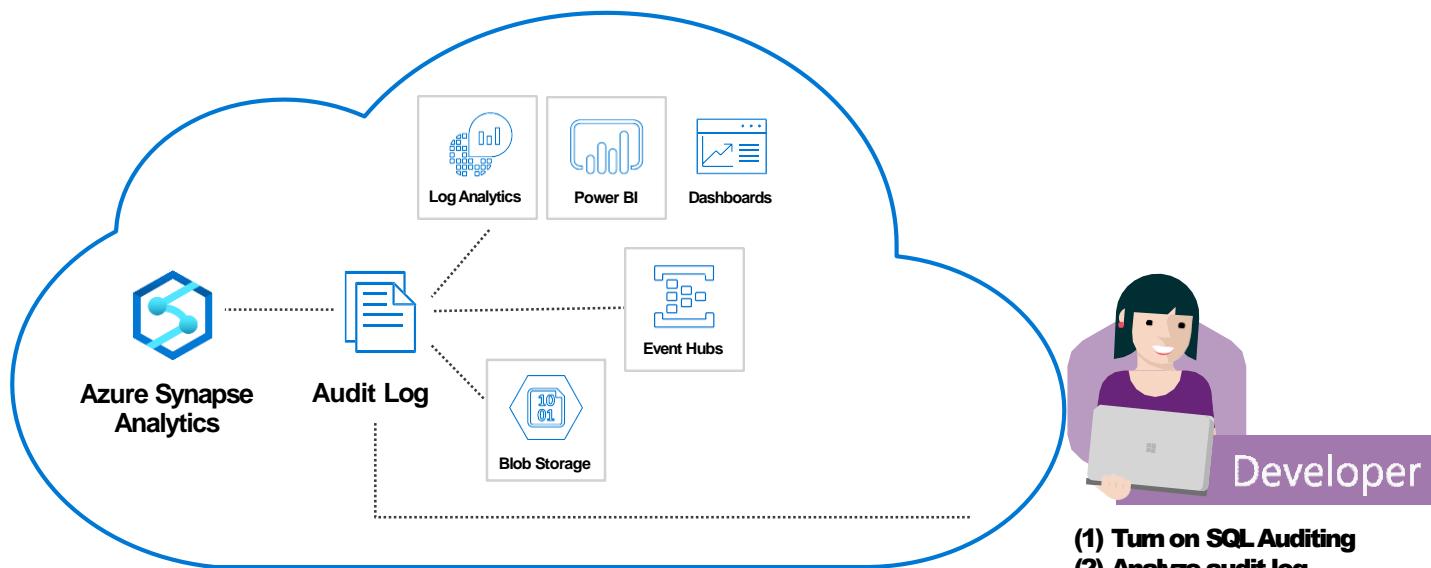
Authentication

Network Security

Threat Protection

SQL Auditing in Azure Log Analytics and EventHubs

Gain insight into database audit log



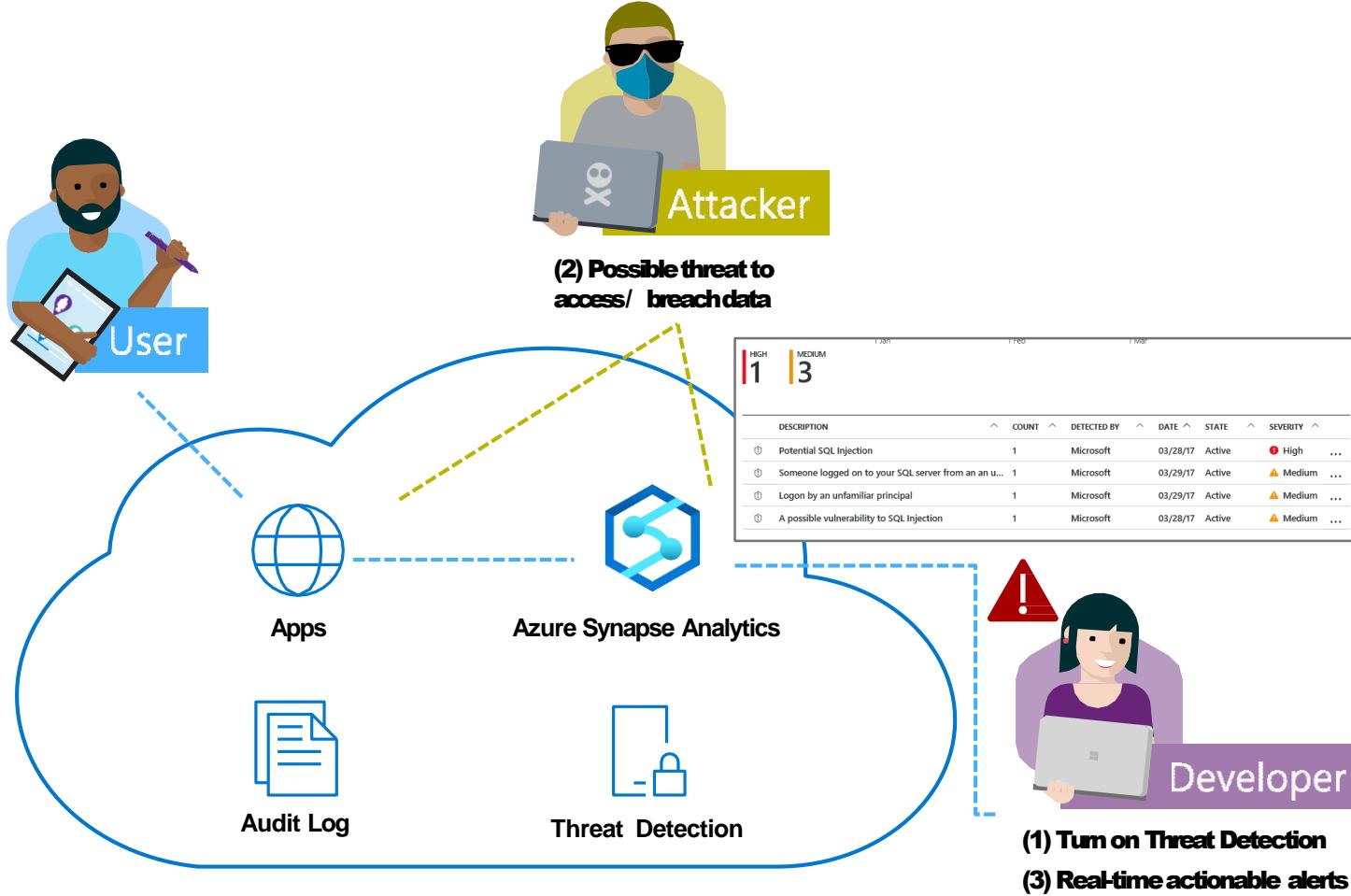
A screenshot of the Azure Log Analytics workspace titled 'Logs'. The search bar contains the query: 'search * | where Category == "SQLSecurityAuditEvents" | project TimeGenerated, server_principal_name_s, statement_s, affected_rows_d, SeverityLevel | sort by TimeGenerated asc'. The results table shows 62 entries. The columns are: TYPE (I), LOGICALSERVERNAME_S (I), CATEGORY (I), and statement_s. The first few rows of the table are:

TYPE (I)	LOGICALSERVERNAME_S (I)	CATEGORY (I)	statement_s
AzureDiagnostics	ronnmativedemo	SQLSecurityAuditEvents	8/15/2018 12:00:22.521 AM admin1 exec sp_executesql N'SELECT tb1.name AS [Name] SCHEMA_NAME(tb1... 0
			8/15/2018 12:00:22.521 AM admin1 exec sp_executesql N'SELECT ISNULL(HAS_PERMS_BY_NAME('QUOTE...' 1
			8/15/2018 12:00:22.521 AM admin1 DECLARE @edition sysname; SET @edition = cast(SERVERPROPERTY(N...' 4
			8/15/2018 12:00:22.521 AM admin1 0
			8/15/2018 12:00:22.521 AM admin1 exec sp_executesql N'SELECT CAST((hi.is_enabled AS bit) AS [isEnabled])... 0
			IF OBJECT_ID ('[sys].[database_query_store_options]') IS NOT NULL BEGIN

- ✓ 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

The screenshot shows the 'Data discovery & classification (preview)' blade in the Azure portal. It includes an 'Overview' section with a summary of 10 classified columns out of 109, 4 tables containing sensitive data, and 4 unique information types. Below this are two donut charts: one for 'Label distribution' showing 10 columns across Confidential, Highly Confidential, Confidential, and General categories, and another for 'Information type distribution' showing 10 columns across CONTACTINFO, NAME, CREDENTIALS, and FINANCIAL categories. At the bottom, there's a table view for 'Information protection' settings, showing columns for Schema, Table, Column, Information Type, and Sensitivity Label.

- ✓ 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.

The screenshot shows the 'Auditing' blade for the 'ayotestdw' database. Under the 'Auditing' section, the 'Auditing' switch is set to 'ON'. Below it, the 'Audit log destination' dropdown is set to 'Storage', and the storage details are listed as 'sqlvasdgoqfss5cyls'. A red box highlights the 'Auditing' switch and the 'Auditing' section in the left sidebar.

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

The screenshot shows the 'sqldbauditlogs' container in the Azure portal. In the 'Overview' section, there is a blob named '01_34_30_090_0.xel'. A red box highlights this blob.

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

The screenshot shows the 'Extended Events Viewer' in SSMS displaying a table of 24785 events. The columns are 'name', 'timestamp', 'affected_rows', 'application_name', 'client_ip', 'data_sensitivity_information', and 'database_name'. A red box highlights the 'data_sensitivity_information' column. Below the table, a detailed view of an 'audit_event' row is shown, with its details also highlighted by a red box.

name	timestamp	affected_rows	application_name	client_ip	data_sensitivity_information	database_name
audit_event	2019-02-26 18:38:35.7892923	0	Net SqlClient Data Provider	10.0.0.4		master
audit_event	2019-02-26 18:38:35.7661039	0	Net SqlClient Data Provider	10.0.0.4		master
audit_event	2019-02-26 18:38:35.7052286	0	Net SqlClient Data Provider	10.0.0.4		master
audit_event	2019-02-26 18:38:35.6873633	0	Net SqlClient Data Provider	10.0.0.4		master
audit_event	2019-02-26 18:38:35.6680990	0	Net SqlClient Data Provider	10.0.0.4		master
audit_event	2019-02-26 18:38:35.6490621	0	Net SqlClient Data Provider	10.0.0.4		master
audit_event	2019-02-26 18:38:35.6292824	0	Net SqlClient Data Provider	10.0.0.4		master
audit_event	2019-02-26 18:38:35.6110493	0	Net SqlClient Data Provider	10.0.0.4		master
audit_event	2019-02-26 18:38:35.5911164	0	Net SqlClient Data Provider	10.0.0.4		master
audit_event	2019-02-26 18:38:35.5739871	0	Net SqlClient Data Provider	10.0.0.4		master
audit_event	2019-02-26 18:38:35.5557121	0	Net SqlClient Data Provider	10.0.0.4		master
audit_event	2019-02-26 18:38:35.5393015	0	Net SqlClient Data Provider	10.0.0.4		master
audit_event	2019-02-26 18:38:35.5213010	0	Net SqlClient Data Provider	10.0.0.4		master
audit_event	2019-02-26 18:38:35.5032121	0	Net SqlClient Data Provider	10.0.0.4		master
audit_event	2019-02-26 18:38:35.4956126	0	Net SqlClient Data Provider	10.0.0.4		master
audit_event	2019-02-26 18:38:35.4675595	0	Net SqlClient Data Provider	10.0.0.4		master
audit_event	2019-02-26 18:38:35.4487751	0	Net SqlClient Data Provider	10.0.0.4		master
audit_event	2019-02-26 18:38:35.4290439	0	Net SqlClient Data Provider	10.0.0.4		master

Event: audit_event (2019-02-26 18:38:35.6680990)

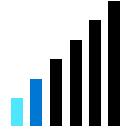
Field	Value
action_id	1176681924
additional_information	<login_information><error_code>18456</error_code><error_stat...
affected_rows	0
application_name	Net SqlClient Data Provider
audit_schema_version	1
class_type	16964
client_ip	10.0.0.4
connection_id	F1AD6457-9F40-409B-B43C-E638AAFA47902
data_sensitivity_information	
database_name	master
database_principal_id	-1
database_principal_name	
duration_milliseconds	0
event_time	2019-02-26 18:38:35.6743004
host_name	usgsvm099
is_column_permission	False
object_id	5
object_name	master

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

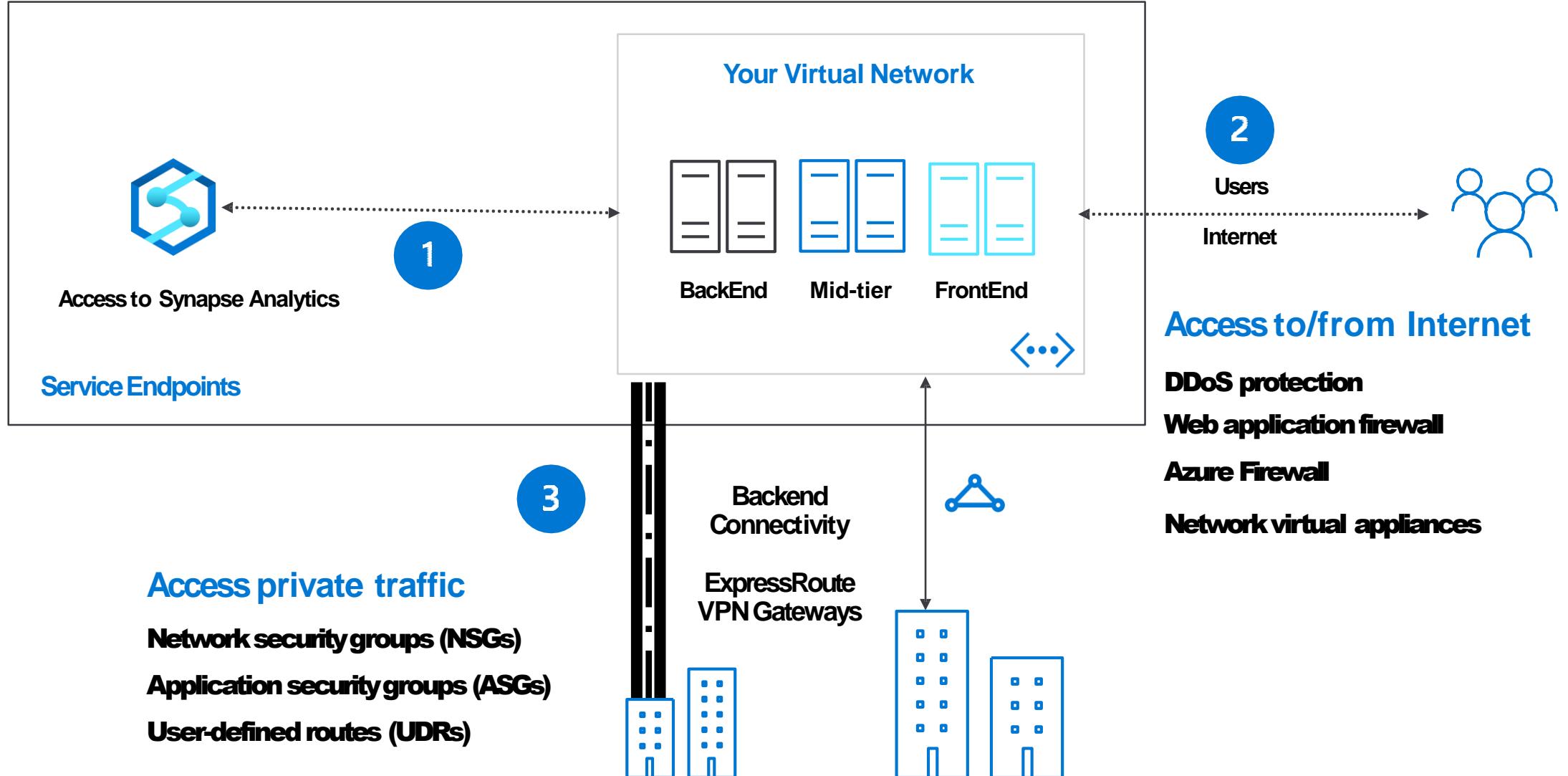
Access Control

Authentication

Network Security

Threat Protection

Azure networking: application-access patterns



Securing with firewalls

Overview

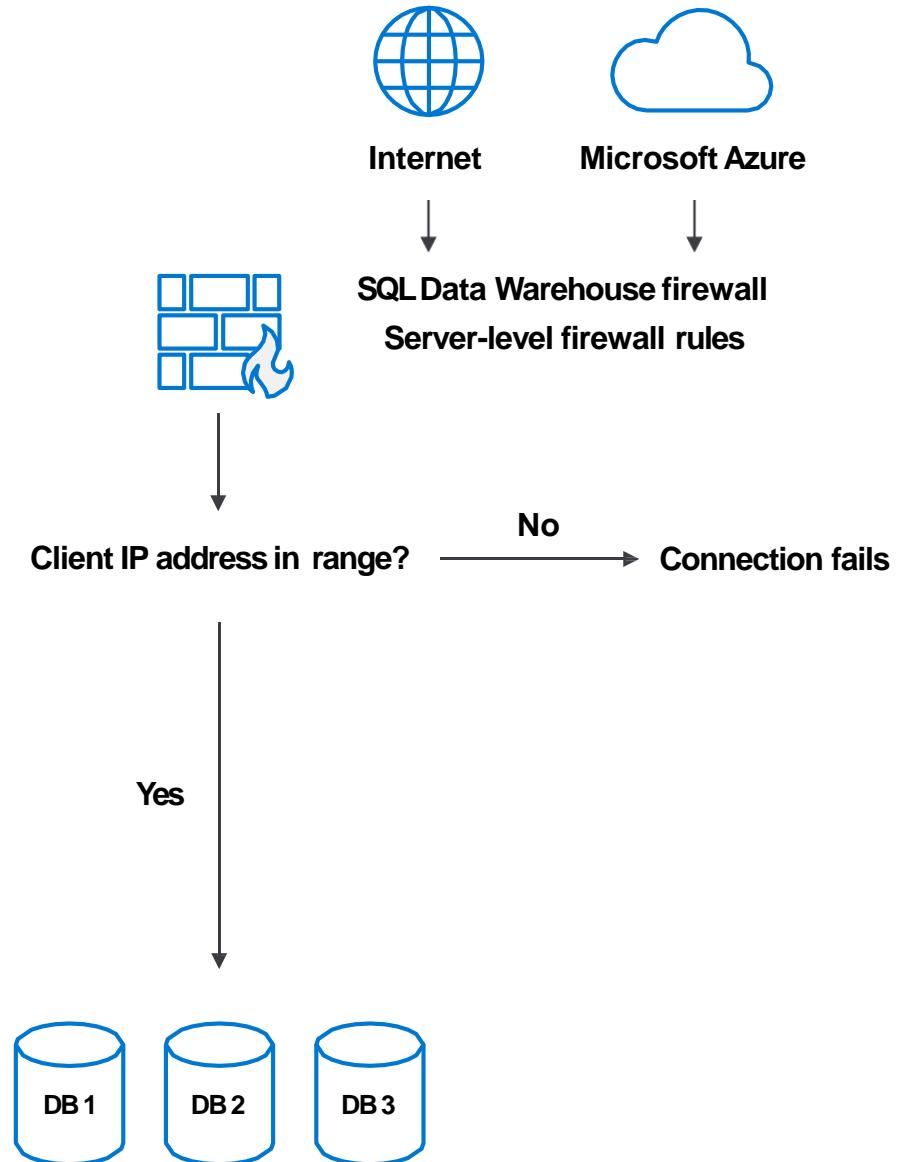
By default, all access to your Azure Synapse Analytics is blocked by the firewall.

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.

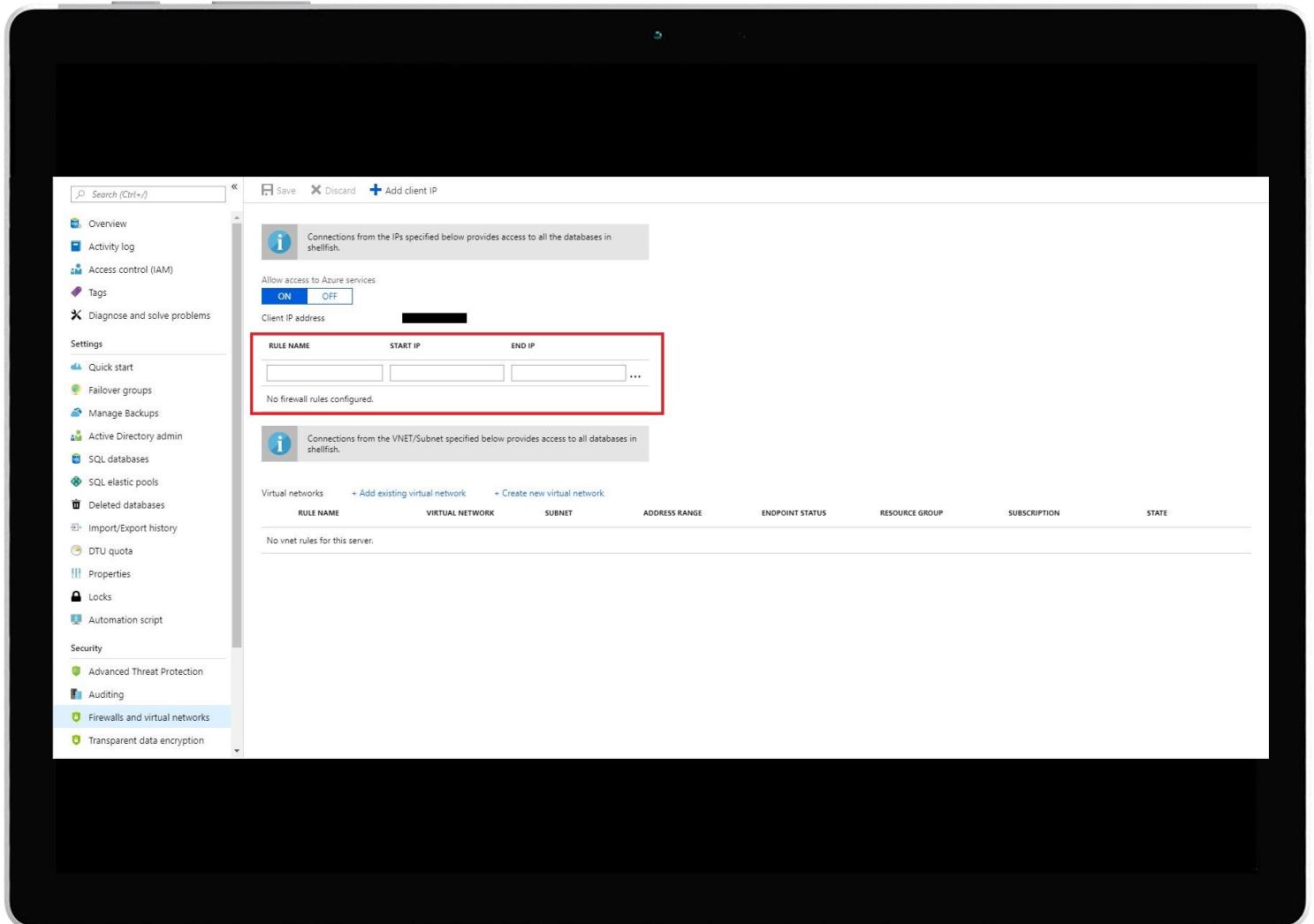


Firewall configuration on the portal

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 RESTAPI

Managing firewall rules through RESTAPI must be authenticated.

For information, see [Authenticating Service Management Requests](#).

Server-level rules can be created, updated, or deleted using [RESTAPI](#).

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.

The screenshot shows the 'Firewall / Virtual Networks' settings for a SQL server named 'gm-sql-db-server-svr1'. At the top, there are 'Save' and 'Discard' buttons, and a '+ Add client IP' button. Below this, a note states: 'Connections from the IPs specified below provides access to all the databases in gm-sql-db-server-svr1.' A toggle switch labeled 'Allow access to Azure services' is set to 'OFF'. The 'Client IP address' is listed as 73.118.201.137. The 'Virtual network rules' section contains two entries:

RULE NAME	START IP	END IP	Actions
gm-ip-rule-ir1	172.27.26.0	172.27.26.255	...
gm-ip-rule-ir2	73.118.201.0	73.118.201.255	...

Below this, another note states: 'Connections from the VNET/Subnet specified below provides access to all databases in gm-sql-db-server-svr1.' At the bottom, there are buttons for 'Virtual networks' (with '+ Add existing' highlighted by a red box), '+ Create new', and a table header row for 'RULE NAME', 'RESOURCE GROUP/VNET NAME', and 'SUBNET'.

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.

[Azure Synapse Analytics](#)

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

Benefits

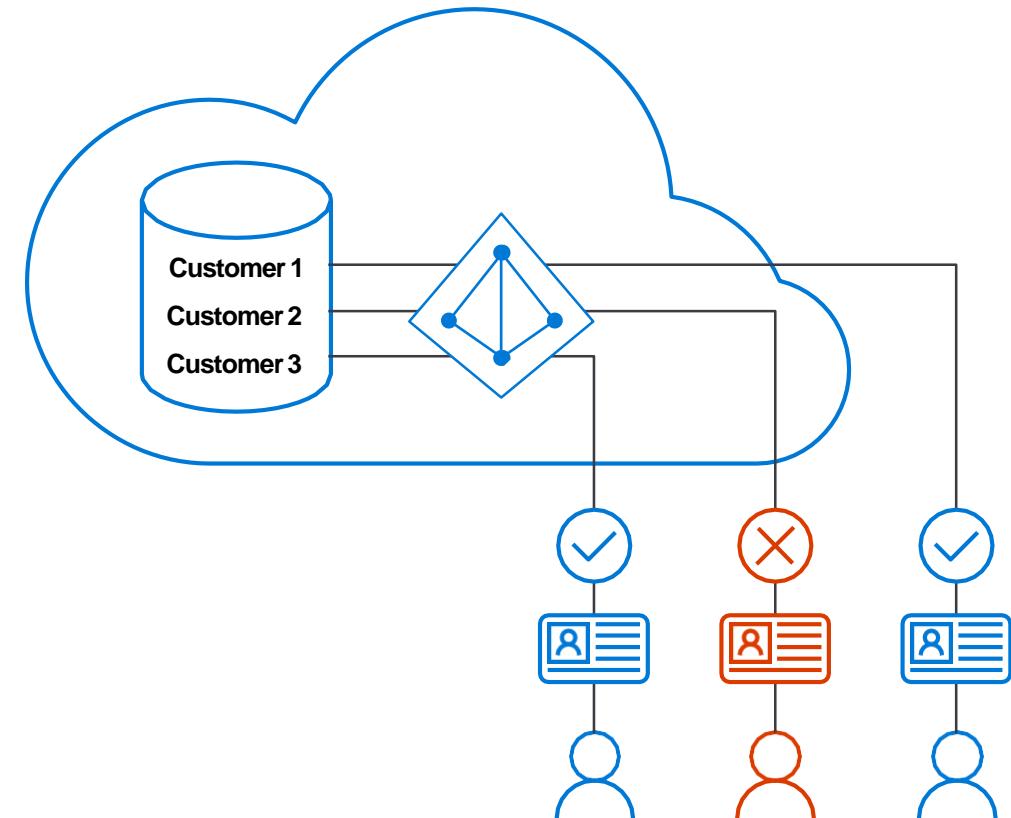
Alternative to SQL Server authentication

Limits proliferation of user identities across databases

Allows password rotation in a single place

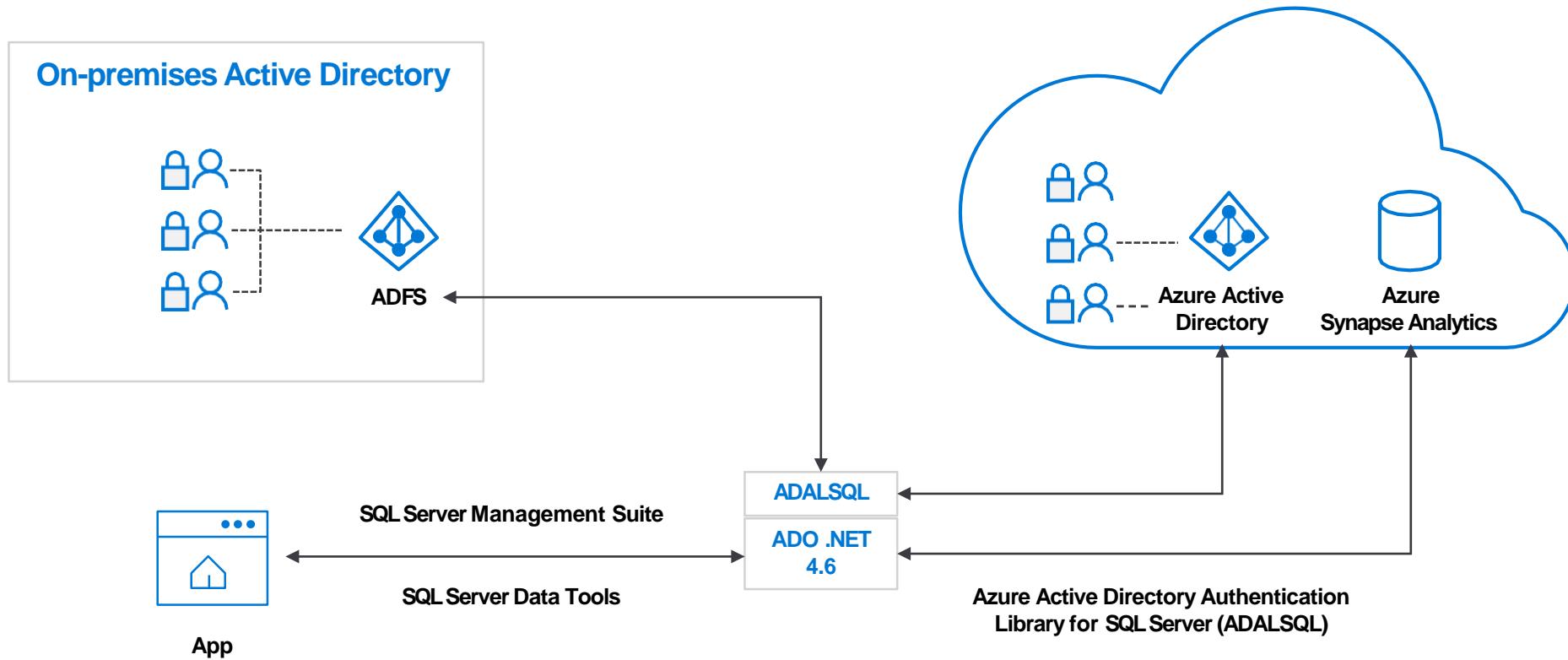
Enables management of database permissions by using external Azure Active Directory groups

Eliminates the need to store passwords



Azure Active Directory trust architecture

Azure Active Directory and Azure Synapse Analytics



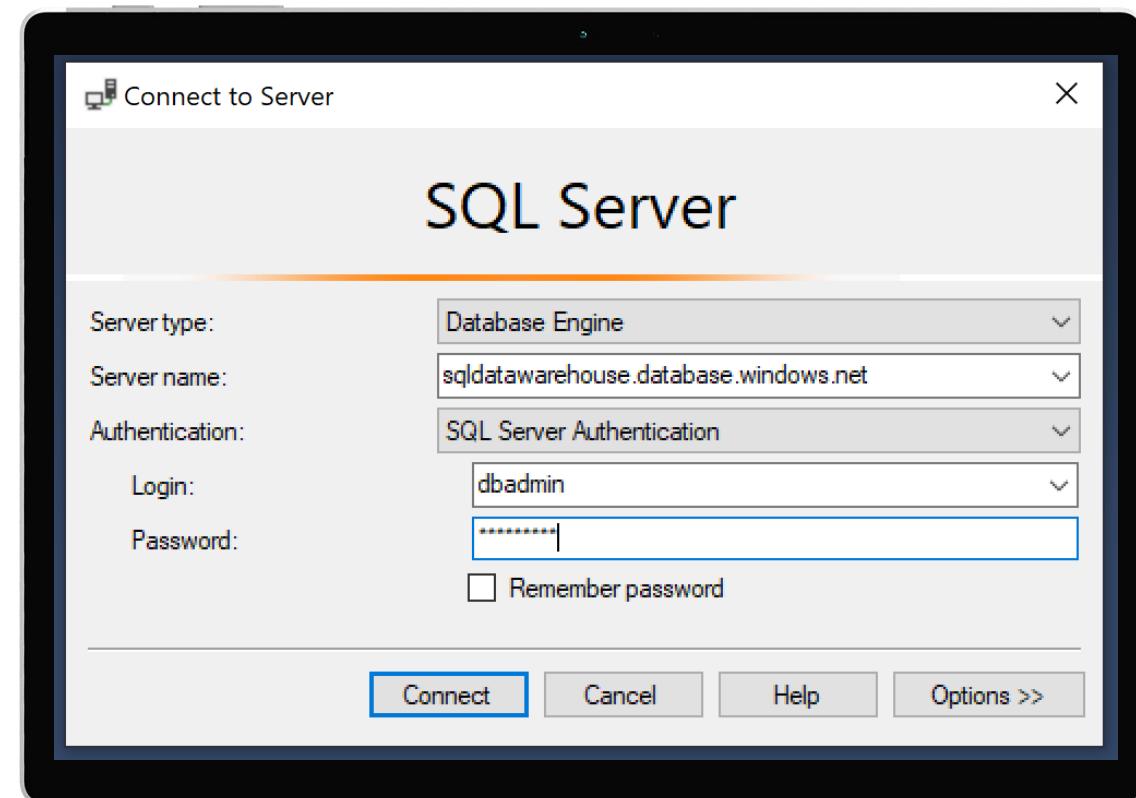
SQL authentication

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 RosaQdMon 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 GRANTOPTION;
GO

-- Grant EXECUTE permission on stored procedure HumanResources.uspUpdateEmployeeHireInfo to an application role called Recruiting11
USE AdventureWorks2012;
GRANT EXECUTE ON OBJECT::HumanResources.uspUpdateEmployeeHireInfo TO RECRUITING11; GO
```

Row-level security (RLS)

Overview

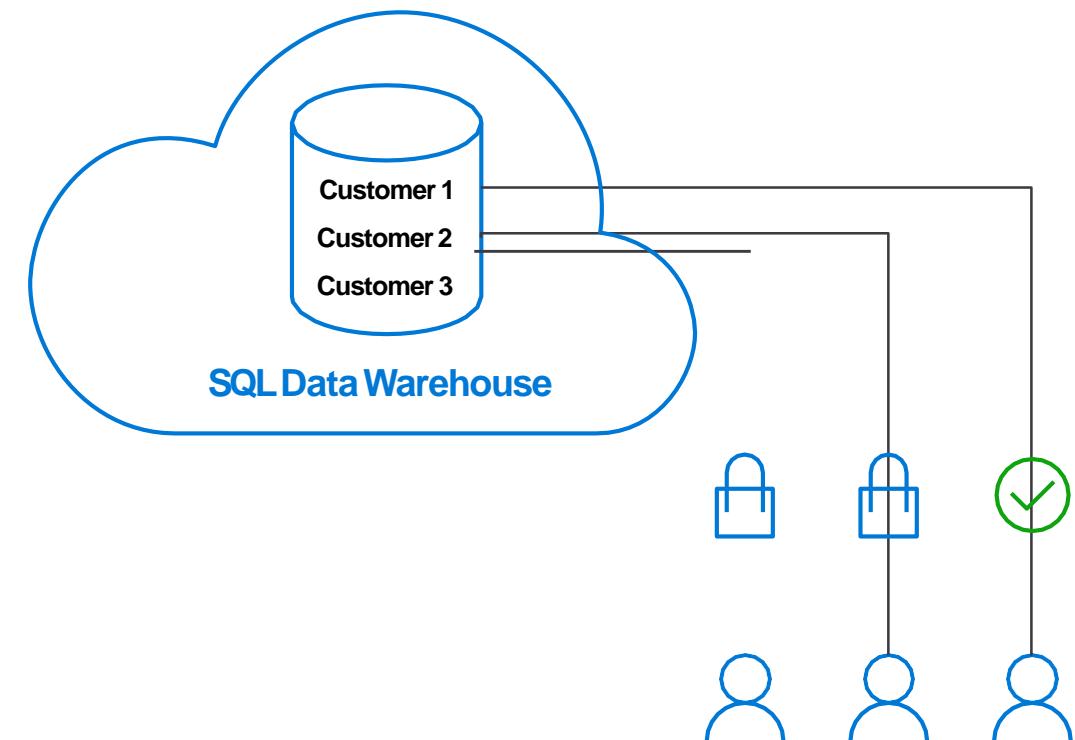
Fine grained access control of specific rows in a database table.

Avoid 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)
RETURN 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
```

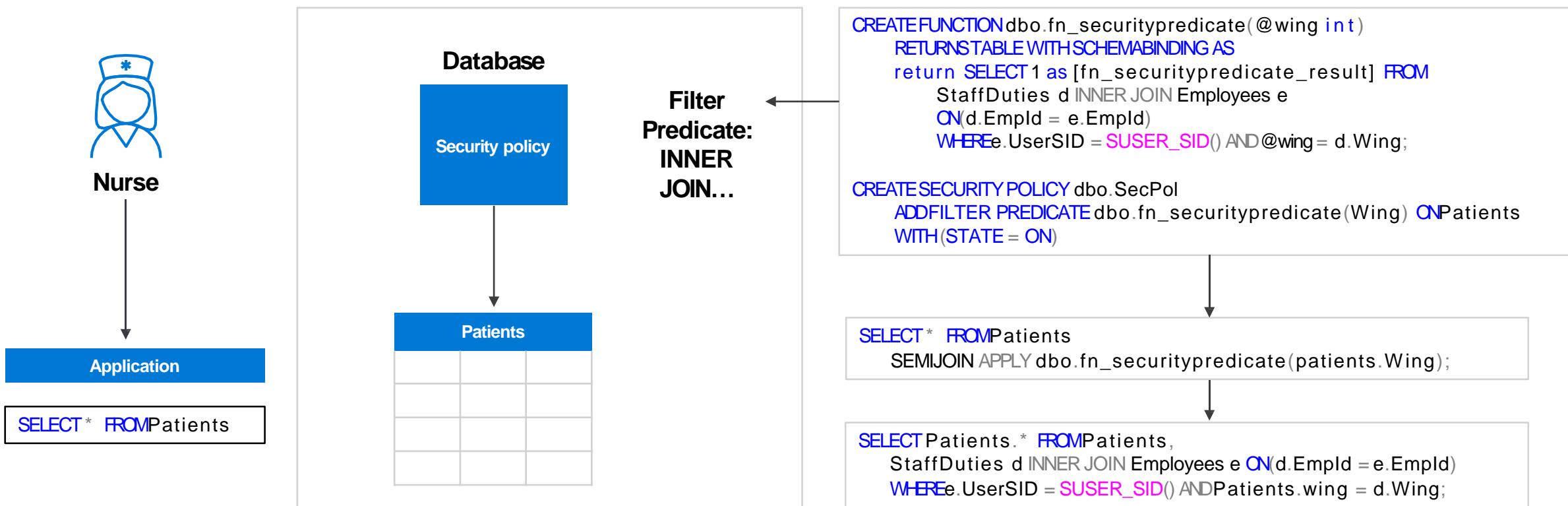
Row-level security

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



Column-level security

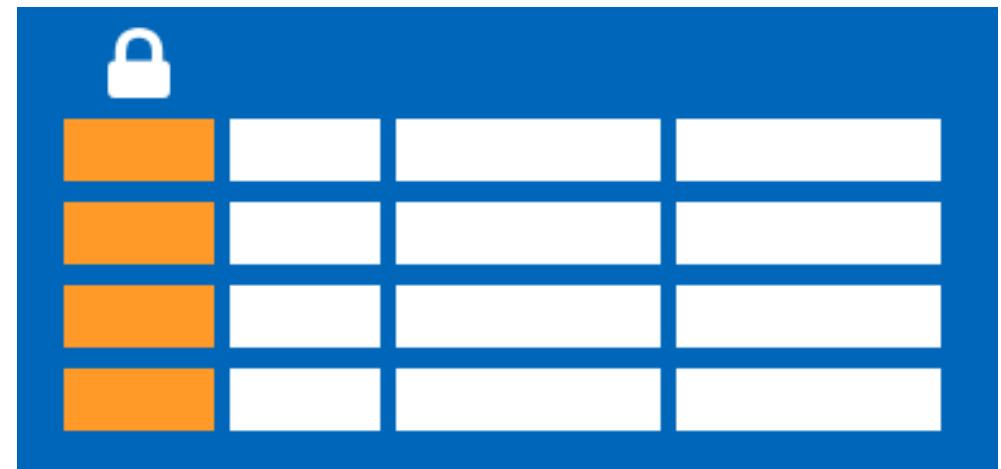
Overview

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.

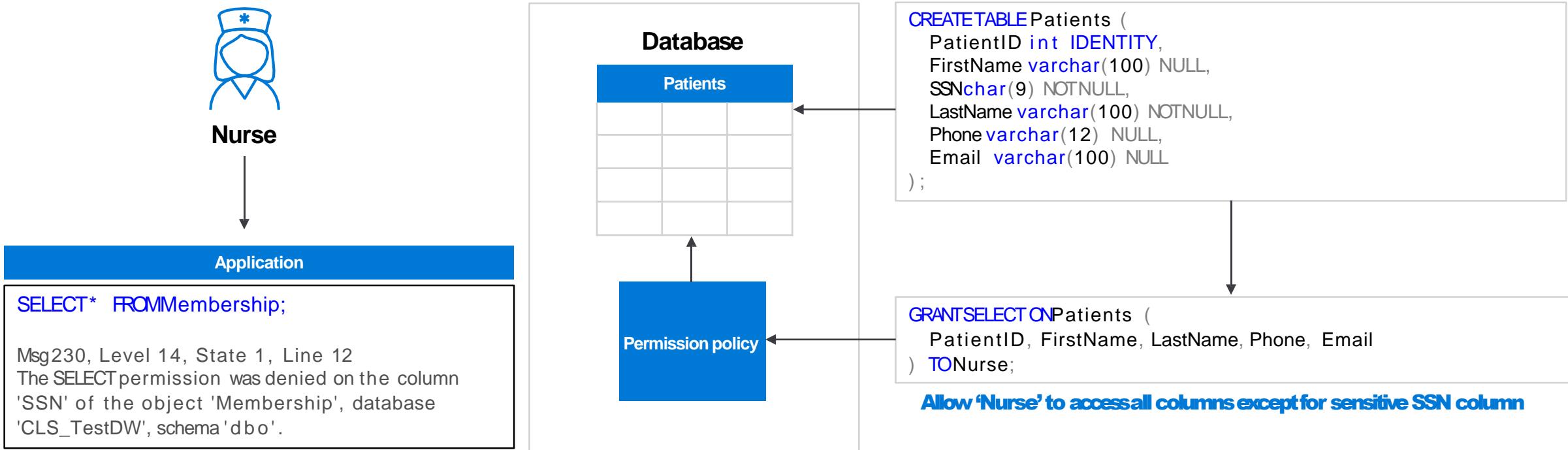
Both Azure Active Directory (AAD) and SQL authentication are supported.



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.



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

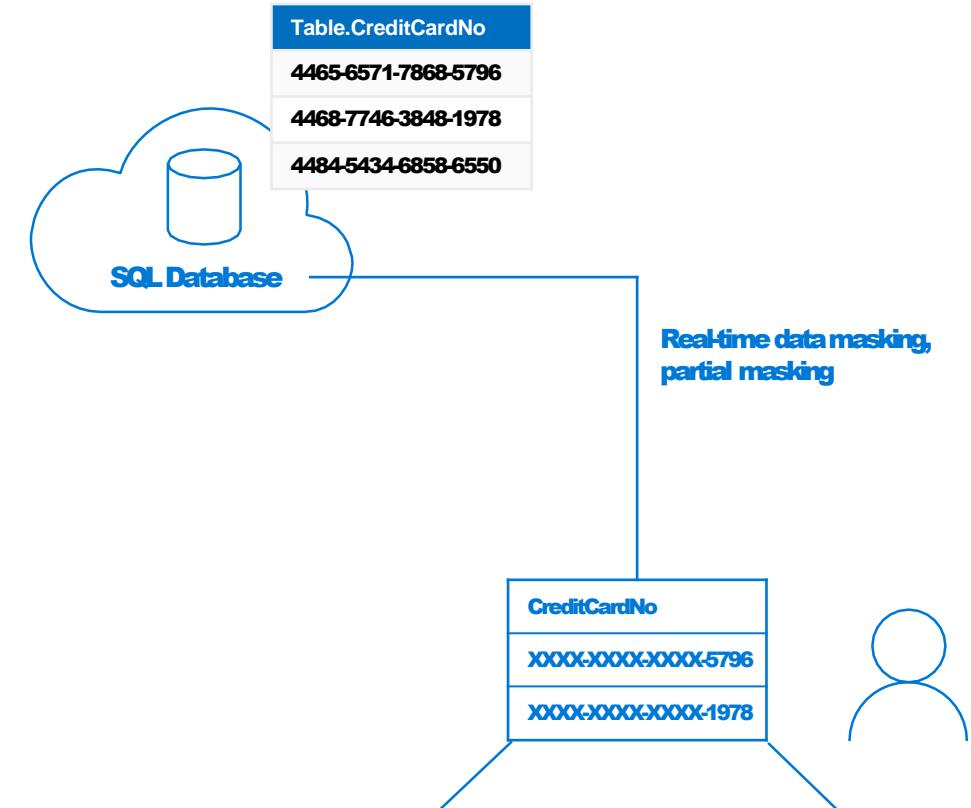
Prevent abuse of sensitive data by hiding it from users

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



Security officer

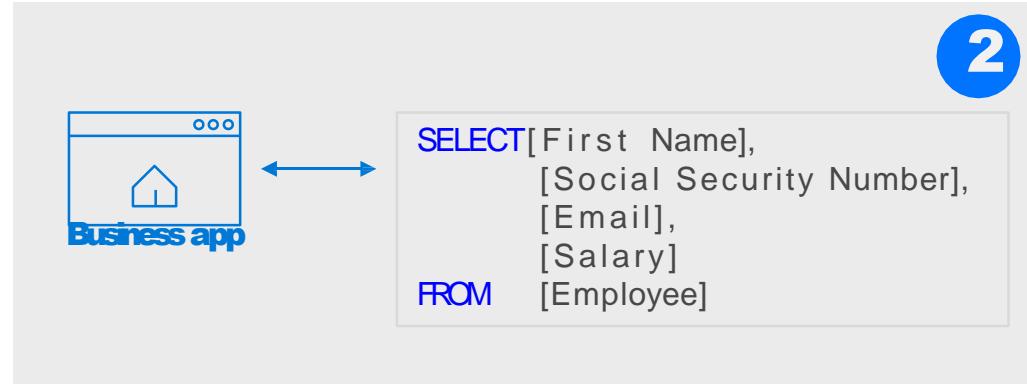
```
ALTER TABLE [Employee]
ALTER COLUMN [SocialSecurityNumber]
ADD MASKED WITH (FUNCTION = 'DEFAULT()')
```

```
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
```

1



2

The diagram shows two tables side-by-side. The top table is titled "Non-masked data (admin login)" and the bottom table is titled "Masked data (admin1 login)". Both tables have columns: First Name, Social Security Num..., Email, and Salary. The "Non-masked data" table shows real data for five employees. The "Masked data" table shows the same five employees, but their Social Security numbers, emails, and salaries are partially obscured by placeholder text ("XXX-XX-XX37", "jXX@XXXX.com", etc.).

	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

	First Name	Social Security Number	Email	Salary
1	LILA	XXX-XX-XX37	IXX@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

3

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

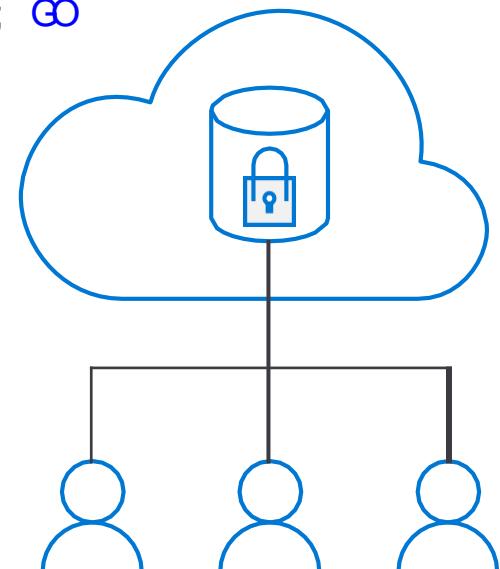


Transparent data encryption (TDE)

- 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.

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

GO
```



Transparent data encryption (TDE)

Key Vault

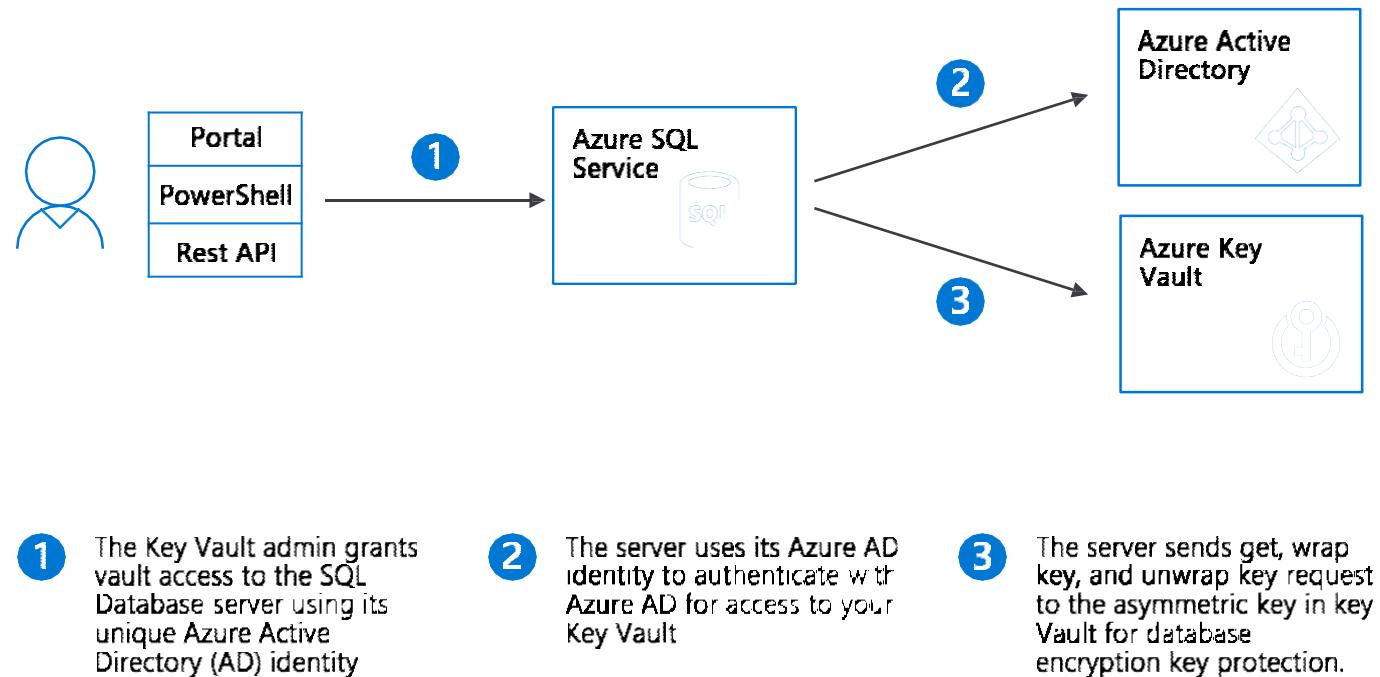
Benefits with User Managed Keys

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

Highly available and scalable cloud-based key store.

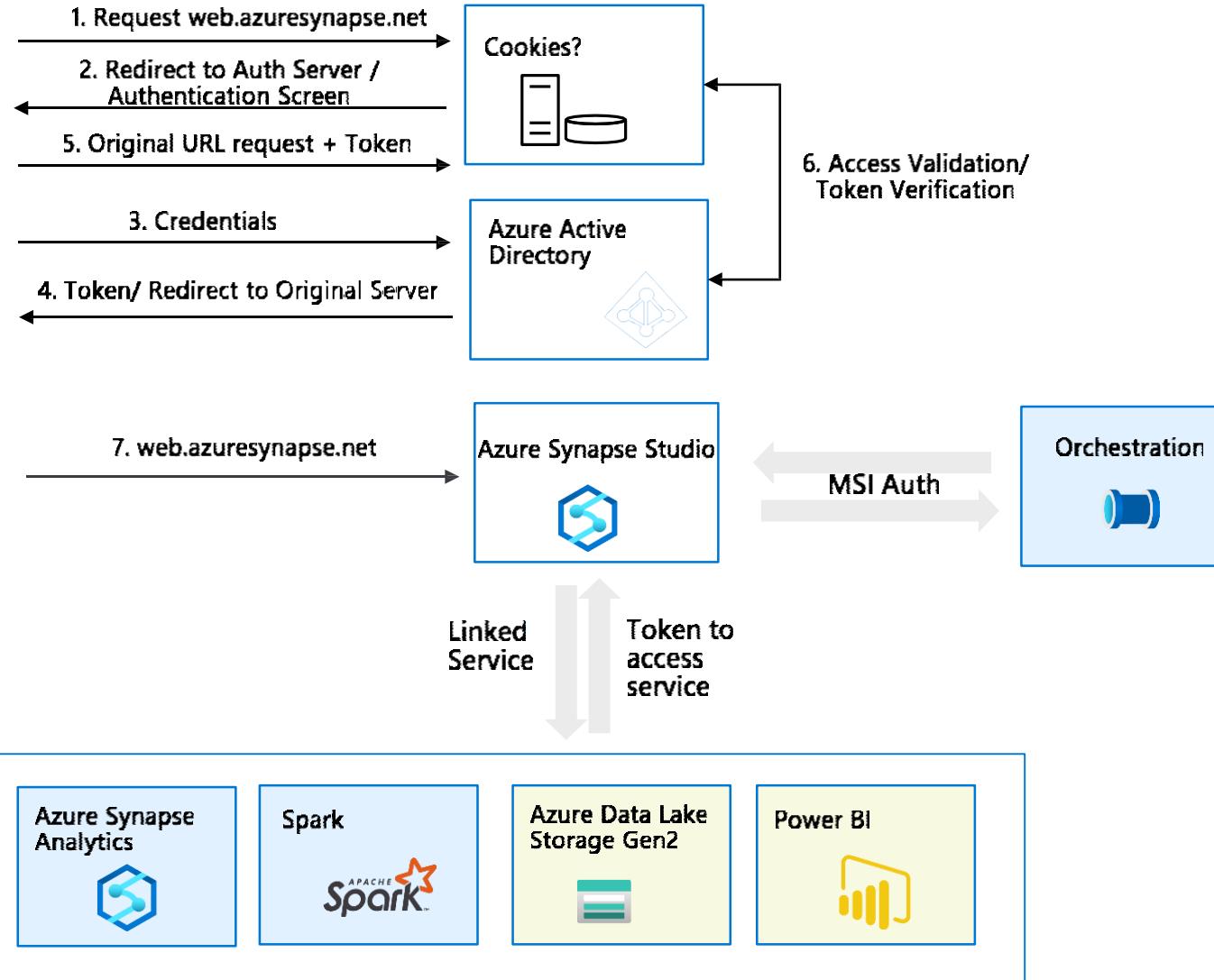
Central key management that allows separation of key management and data.

Configurable via Azure Portal, PowerShell, and REST API.



Single Sign-On

- Synapse Foundation Components
- Synapse Linked Services



Implicit authentication - User provides login credentials once to access Azure 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



Microsoft

The background of the slide features a clear blue sky with several wispy, white clouds scattered across it.

Questions?

www.desertislesql.com