



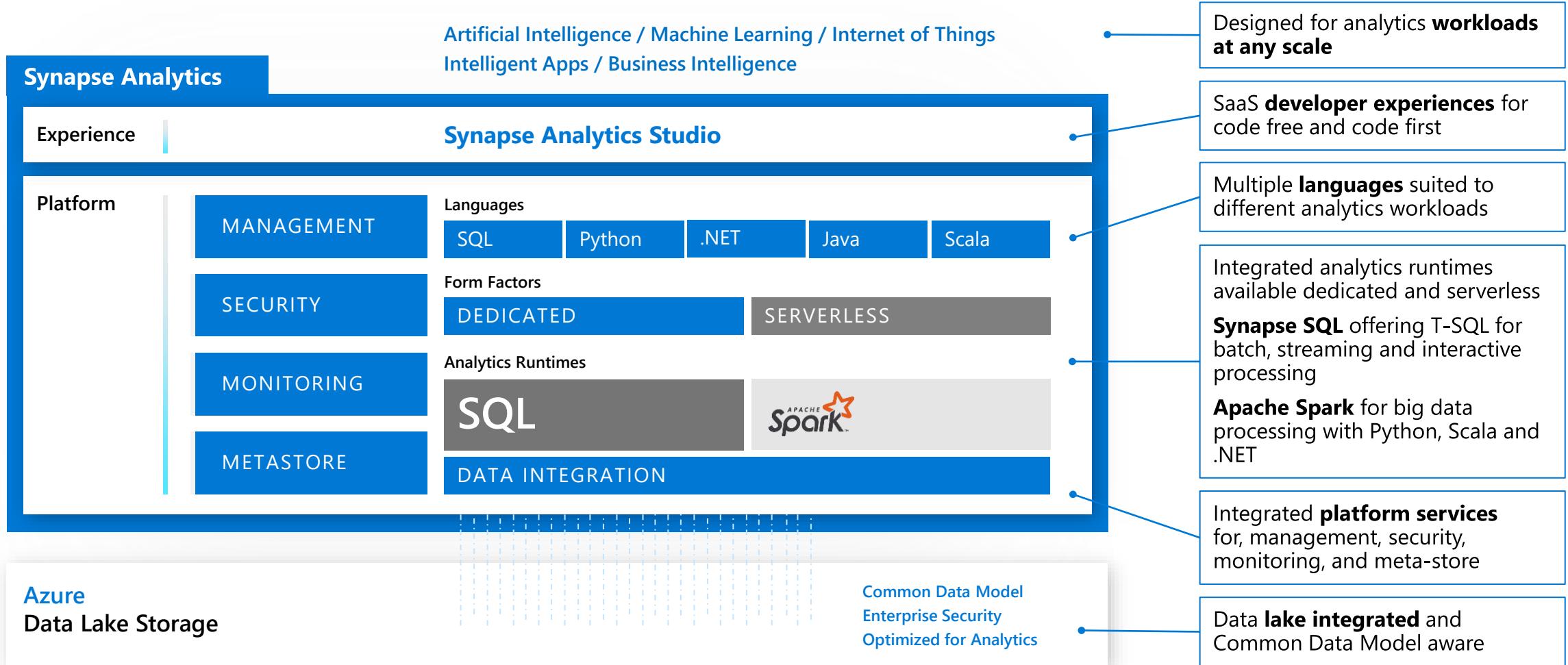
# Azure Synapse Analytics

Gary Lee

Cloud Solution Architect (Data / AI / Blockchain)

# Azure Synapse Analytics

Limitless analytics service with unmatched time to insight



# Sections



- [Studio](#)
- [Integrate](#)
- [Synapse SQL](#)
  - [Dedicated SQL Pool](#)
  - [Serverless SQL Pool](#)
- [Spark Analytics](#)
- [Security](#)
- [Monitor](#)
- [Management](#)
- [Metastore](#)
- [Linked services](#)
- [Synapse Link](#)



# Azure Synapse Analytics Studio

# Studio

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

The screenshot shows the Microsoft Azure Synapse Analytics Studio interface. The top navigation bar includes 'Microsoft Azure' and 'Synapse Analytics' with a workspace name 'wsazuresynapseanalytics'. The left sidebar has a 'Home' icon selected, along with 'Data', 'Develop', 'Integrate', 'Monitor', and 'Manage' options. The main content area is titled 'Synapse workspace' and 'wsazuresynapseanalytics'. It features a 'New' button and four primary action cards: 'Ingest' (cloud icon), 'Explore and analyze' (3D bar chart icon), 'Visualize' (bar chart icon), and 'Learn' (book icon). A large circular graphic in the background illustrates data flow and analysis. Below these cards is a 'Recent resources' section with a table:

Name	Last opened by you
05 Sentiment_Analysis_Cognitive_Services	4 hours ago
Predict NYCTaxi Trip Amount	4 hours ago
001 SQL Pool Security RLS DDM CLE	5 hours ago
005 Predict In-Engine Scoring	a day ago
05 Anomaly_Detection_Cognitive_Services	a day ago

At the bottom left of the recent resources section is a 'Show more' link.

# Synapse Studio

Synapse Studio divided into **Activity hubs**.

These organize the tasks needed for building analytics solution.

The screenshot shows the Microsoft Azure Synapse Studio interface. On the left, there is a vertical sidebar with a red border around the first five items: Home, Data, Develop, Integrate, and Monitor. A red arrow points from the 'Integrate' item in this sidebar to the 'Integrate' hub icon on the main page. The main page has a header with 'Microsoft Azure | Synapse Analytics > wsazuresynapseanalytics' and a user 'someone@microsoft.com'. Below the header, there's a 'Synapse workspace' section with a 'wsazuresy' workspace card and a 'New' button. The main content area is divided into six activity hubs: Home, Data, Develop, Integrate, Monitor, and Manage. Each hub has an icon and a brief description.

- Home**: 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, linked service, access to artifacts.

Recent resources are listed on the left under 'Recent resources'.

# Home Hub

Ease of access to get updates, to switch workspace, to get notifications and to provide feedback

Microsoft Azure | Synapse Analytics > wsazuresynapseanalytics

someone@microsoft.com MICROSOFT

Home Data Develop Integrate Monitor Manage

Synapse workspace  
wsazuresynapseanalytics

New ▾

Updates Switch Workspaces Notifications Feedback

Ingest Explore and analyze Visualize Learn

Recent resources

Name	Last opened by you
05 Sentiment_Analysis_Cognitive_Services	4 hours ago
Predict NYCTaxi Trip Amount	4 hours ago
001 SQL Pool Security RLS DDM CLE	5 hours ago
005 Predict In-Engine Scoring	a day ago



# Synapse Studio Home hub

# Home Hub

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

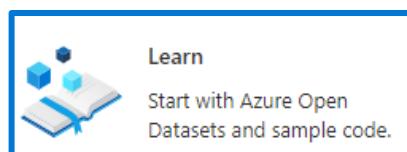
The screenshot shows the Microsoft Azure Synapse Analytics Home Hub. The top navigation bar includes 'Microsoft Azure' and 'Synapse Analytics' with the workspace name 'wsazuresynapseanalytics'. The left sidebar has a 'Home' tab selected, along with 'Data', 'Develop', 'Integrate', 'Monitor', and 'Manage' tabs. The main content area is titled 'Synapse workspace' and 'wsazuresynapseanalytics'. It features a 'New' button and four activity cards: 'Ingest' (Perform a one-time or scheduled data load.), 'Explore and analyze' (Learn how to get insights from your data.), 'Visualize' (Build interactive reports with Power BI capabilities.), and 'Learn' (Start with Azure Open Datasets and sample code.). A red box highlights the 'Ingest', 'Explore and analyze', 'Visualize', and 'Learn' cards. Below this is a section titled 'Recent resources' with a table:

Name	Last opened by you
05 Sentiment_Analysis_Cognitive_Services	4 hours ago
Predict NYCTaxi Trip Amount	4 hours ago
001 SQL Pool Security RLS DDM CLE	5 hours ago
005 Predict In-Engine Scoring	a day ago
05 Anomaly_Detection_Cognitive_Services	a day ago

A 'Show more' button is at the bottom of the recent resources list.

# Home Hub - Learn

Knowledge center offers one stop for learning with samples, artifacts and Synapse studio tour



Microsoft Azure | Synapse Analytics > wsazuresynapseanalytics

Home > Knowledge center

## Knowledge center

Get started with Azure Open Datasets and sample code. Return to the Knowledge center periodically as we provide updated content.



Experience limitless scale

Deliver insights from all your data, across data warehouses and big data analytics systems, with blazing speed.

Use samples immediately

Click once and we'll create everything you need, from scripts and notebooks to pools and data.

Browse gallery

Select from sample code and Azure Open Datasets to quickly get started in your workspace.

Tour Synapse studio

Familiarize yourself with key features of Synapse Studio. Start by taking a tour of the homepage.

# Knowledge center

Knowledge center offers open datasets, sample notebooks, SQL scripts and pipeline templates for easy start and learning

**Use samples immediately**

Create everything you need in just one click.

**Explore sample data with Spark**  
 Includes a sample script. If you have permissions, we'll create a new pool for you; otherwise, you can use an existing pool.  
 Name SampleSpark  
 Size Medium (8 vCores / 64 GB) - 3 nodes

**Query data with SQL**  
 Includes a sample script and serverless SQL pool - Built-in (included with your workspace).

**Create external table with SQL**  
 Includes a sample script. You can use serverless SQL pool - Built-in (included with your workspace) or a dedicated SQL pool. We will create a table for you called SampleTable.  
 Create a pool  Select an existing pool  
 Name SampleSQL  
 Size DW100c

[Use sample](#) [Cancel](#)

Microsoft Azure | Synapse Analytics > wsazuresynapseanalytics Home Help Feedback ? someone@microsoft.com MICROSOFT

**Gallery**

Datasets Notebooks SQL scripts Pipelines

Filter by keyword Tags : All

 <b>Bing COVID-19 Data</b> Bing COVID-19 data includes confirmed, fatal, and recovered cases from all regions, updated da... ID: bing-covid-19-data Sample	 <b>Boston Safety Data</b> Read data about 311 calls reported to the city of Boston. This dataset is stored in Parquet format and is up... ID: city_safety_boston Sample	 <b>COVID Tracking Project</b> The COVID Tracking Project dataset provides the latest numbers on tests, confirmed cases, hospitalizat... ID: covid-tracking Sample	 <b>Chicago Safety Data</b> Read data about 311 calls reported to the city of Chicago. This dataset is stored in Parquet format and is ... ID: city_safety_chicago Sample
 <b>European Centre for Disease Prevention and Control (ECDC) Covid-19 Cases</b> The latest available public data on... ID: ecdc-covid-19-cases Sample	 <b>NOAA Integrated Surface Data (ISD)</b> NOAA Integrated Surface Data (ISD) provides Worldwide hourly weath... ID: isd Sample	 <b>NYC Taxi &amp; Limousine Commission - For-Hire Vehicle (FHV) trip records</b> The For-Hire Vehicle trip records i... ID: nyc_tlc_fhv Sample	 <b>NYC Taxi &amp; Limousine Commission - green taxi trip records</b> The green taxi trip records include... ID: nyc_tlc_green Sample

[Continue](#) [Close](#)

# Home Hub

It provides links to feature announcements, documents and community center for reference

### Feature showcase



Get started with Azure Synapse Analytics

Take a step-by-step tour of Synapse exploring the breadth of capabilities offered.

[Learn more](#)

### Community

**Azure HDInsight Spark Language Runtime Comparison - Lower is better**



Query	.NET	Python	Scala	Java
1. Numerics	~100	~100	~100	~100
2. Map	~100	~100	~100	~100
3. Reduce	~100	~100	~100	~100
4. Join	~100	~100	~100	~100
5. GroupBy	~100	~100	~100	~100
6. Window	~100	~100	~100	~100
7. Sort	~100	~100	~100	~100
8. Filter	~100	~100	~100	~100
9. MapPartitions	~100	~100	~100	~100
10. ReducePartitions	~100	~100	~100	~100
11. GroupByKey	~100	~100	~100	~100
12. WindowPartitions	~100	~100	~100	~100
13. SortPartitions	~100	~100	~100	~100
14. FilterPartitions	~100	~100	~100	~100
15. MapWithIndex	~100	~100	~100	~100
16. ReduceWithIndex	~100	~100	~100	~100
17. GroupByKeyWithIndex	~100	~100	~100	~100
18. WindowWithIndex	~100	~100	~100	~100
19. SortWithIndex	~100	~100	~100	~100
20. FilterWithIndex	~100	~100	~100	~100
21. MapWithIndexPartitions	~100	~100	~100	~100
22. ReduceWithIndexPartitions	~100	~100	~100	~100

**Microsoft and the .NET Foundation announce the release of version 1....**

.NET for Apache Spark is now available in Synapse Analytics.

Oct 26

**Synapse SQL: SQL permissions**

**ADLS: ACL permissions**

**Storage files**

Securing access to ADLS files using Synapse SQL permission model

Define serverless SQL permissions using SQL runtime permission model.

Oct 19

**Analyze CosmosDB data using Synapse Link and Transact-SQL Ia...**

Analyze data and create reports from Cosmos DB.

Oct 16

**Quickly Get Started with Samples in Azure Synapse Analytics**

Be even more productive with the knowledge center in Synapse Studio.

Sep 24

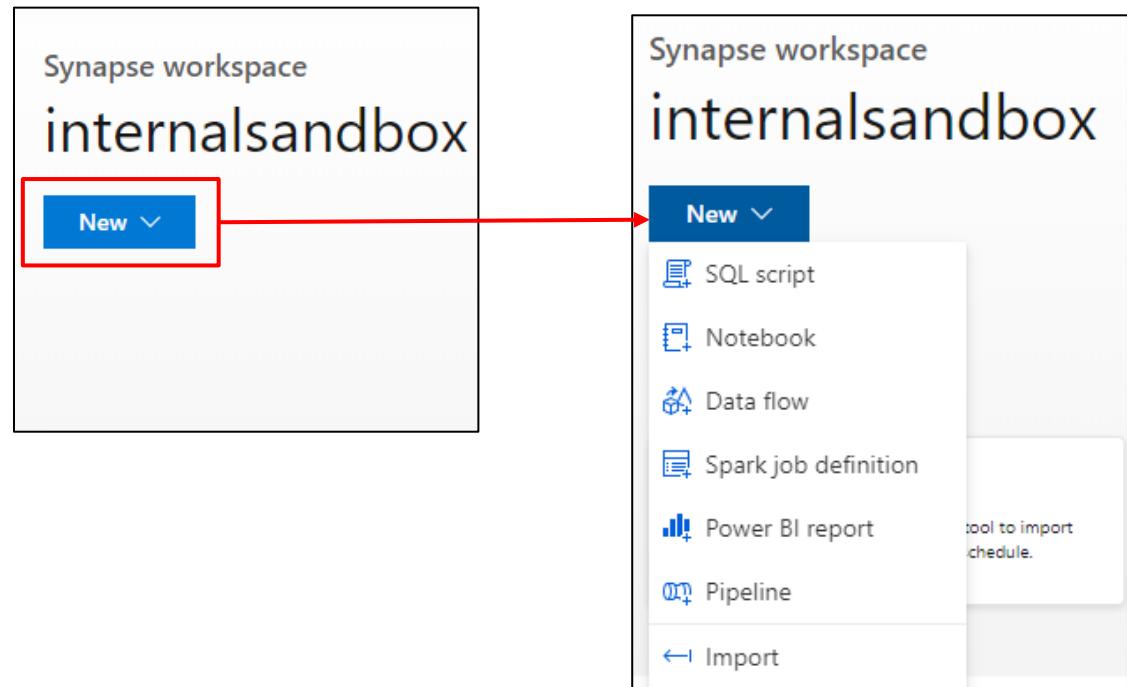
[Visit the community center](#)

# Home Hub

# Overview

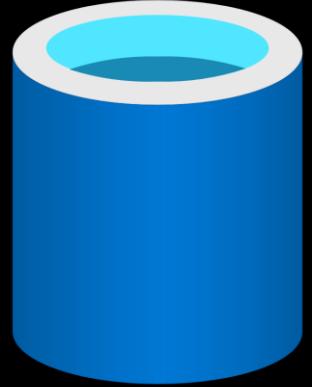
**New dropdown** – offers quickly start work item

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



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

NAME	LAST OPENED BY YOU
 NYCTAx1	6 hours ago



# Synapse Studio **Data hub**

# Data Hub

Explore data inside the workspace and in linked storage accounts

This screenshot shows the Microsoft Azure Synapse Analytics Data Hub interface. The left sidebar includes Home, Data (selected), Develop, Integrate, Monitor, and Manage. The main area has tabs for Data, Workspace (selected and highlighted with a red box), and Linked. A search bar at the top says "Filter resources by name". Below are sections for Databases (10 items) and Integration (24 items). The Databases section lists newpoll (SQL), NYCTaxi\_Pool (SQL), Predict\_Pool (SQL), Streaming\_Pool (SQL), WWI\_Pool (SQL), NYT2020 (SQL), SQLServerlessDB (SQL), and default (Spark).

- Databases (10)
  - newpoll (SQL)
  - NYCTaxi\_Pool (SQL)
  - Predict\_Pool (SQL)
  - Streaming\_Pool (SQL)
  - WWI\_Pool (SQL)
  - NYT2020 (SQL)
  - SQLServerlessDB (SQL)
  - default (Spark)

- Integration (24)
  - (Attached Containers)
  - wsazuresynapseanalytics (Primary...)
  - Azure Data Lake Storage Gen2
  - Azure Data Explorer
  - Azure Cosmos DB
  - Azure Blob Storage

This screenshot shows the Microsoft Azure Synapse Analytics Data Hub interface with the Linked tab selected (highlighted with a red box). The left sidebar is identical to the first screenshot. The main area shows a list of linked storage accounts and datasets. The list includes Azure Blob Storage (3), Azure Cosmos DB (1), Azure Data Explorer (2), Azure Data Lake Storage Gen2 (2), wsazuresynapseanalytics (Primary...) (24), (Attached Containers) (24), and Integration datasets (24).

- Azure Data Lake Storage Gen2 (2)
  - wsazuresynapseanalytics (Primary...)
- Integration datasets (24)
- (Attached Containers) (24)
- Azure Data Explorer (2)
- Azure Cosmos DB (1)
- Azure Blob Storage (3)

# Data Hub – Linked Storage

Browse Azure Data Lake Storage Gen2 accounts – filesystems, Azure Data Explorer – clusters, Azure Cosmos DB -containers

The screenshot shows the Microsoft Azure Synapse Analytics Studio interface. On the left, the 'Data' sidebar is open, showing 'Linked' storage resources:

- Linked Cosmos DB Analytical Store**: Points to the 'Azure Cosmos DB' item.
- Linked Azure Data Explorer**: Points to the 'Azure Data Explorer' item.
- Linked ADLS Gen2 Account**: Points to the 'wsazuresynapseanalytics (Primary...)' item.
- Container (filesystem)**: Points to the 'rawdata' item under the 'wsazuresynapseanalytics' account.

The main workspace shows a file path 'rawdata > taxidata'. Below it is a table of files:

Name	Last Modified	Content Type	Size
part-00000-0300809f-304e-44bc-81bd-bbd63974c3e4-c000.snappy.parq...	8/27/2020, 12:32:19 AM		121.9 MB
part-00000-6b990121-0341-456c-8723-aec72b03f65f-c000.snappy.parqu...	8/27/2020, 12:32:25 AM		535.4 MB
part-00001-0300809f-304e-44bc-81bd-bbd63974c3e4-c000.snappy.parq...	8/27/2020, 12:32:20 AM		124.5 MB
part-00001-6b990121-0341-456c-8723-aec72b03f65f-c000.snappy.parqu...	8/27/2020, 12:32:23 AM		983.7 MB
part-00002-0300809f-304e-44bc-81bd-bbd63974c3e4-c000.snappy.parq...	8/27/2020, 12:32:19 AM		123.7 MB
part-00002-6b990121-0341-456c-8723-aec72b03f65f-c000.snappy.parqu...	8/27/2020, 12:32:21 AM		966.1 MB

At the bottom, it says 'Showing 1 to 6 of 6 cached items'.

# Data Hub – Storage accounts

Preview a sample of your data

The screenshot illustrates the process of previewing data from a storage account within the Azure Synapse Analytics Studio.

**Left Window:** Shows the "Data" workspace with a list of linked services including Azure Blob Storage, Azure Cosmos DB, Azure Data Explorer, Azure Data Lake Storage Gen2, and a primary storage account named "wsazuresynapseanalytics".

**Middle Window:** A context menu is open for a file named "Products.csv" located in the "rawdata" folder under "sample csv files". The "Preview" option is highlighted with a red box and a red arrow points from it to the preview window on the right.

**Right Window:** A preview of the "Products.csv" file is shown. The file path is <https://azuresynapsesa.dfs.core.windows.net/rawdata/sample csv files/Products.csv>. The file was modified on 10/27/2020, 8:38:51 PM. The "With column header" toggle is turned on. The preview shows the following data:

PRODUCTID	PRODUCTNAME	PRODUCTCATEGORY	UNITPRICE
406032	Apple	100	2.48
406064	Banana	100	1.49
406096	Avocado	100	3.49
406128	Oranges	100	2.99
406160	Onion	100	3.49
406192	Potato	100	5.49
406224	Broccoli	100	6.49
406256	Beaf	100	10.49
406288	Chicken	100	20.49

An "OK" button is visible at the bottom of the preview window.

# Data Hub – Storage accounts

See basic file properties

The screenshot illustrates the Azure Synapse Studio interface for managing storage accounts. On the left, the 'Data' hub shows a list of resources under 'Linked'. A red arrow points from the 'Properties...' button in the context menu of a 'Products.csv' file in the 'rawdata' folder to the 'Properties' dialog box on the right.

**Properties Dialog Box:**

- Name: sample csv files/Products.csv
- URL: https://azuresynapsesa.dfs.core.windows.net/rawdata/sample csv files/Products.csv
- ABFSS Path: abfss://rawdata@azuresynapsesa.dfs.core.windows.net/sample csv files/Products.csv
- Last modified: 10/27/2020, 8:38:51 PM
- Cache Control: max-age=0
- Content Type: application/octet-stream
- Content Disposition:
- Content Encoding:
- Content Language:
- User Properties:

Buttons at the bottom: Apply, Cancel

# Data Hub – Storage accounts

Manage Access - Configure standard POSIX ACLs on files and folders

The screenshot illustrates the process of managing access to a file in Azure Synapse Studio. On the left, the 'Data' workspace shows a tree view of resources, including 'rawdata' under 'wsazuresynapseanalytics'. In the center, a file browser window displays 'Products.csv' in the 'rawdata' folder. A context menu is open over the file, with the 'Manage access...' option highlighted by a red box and a red arrow pointing to the 'Manage Access' dialog on the right.

**Manage Access**

Users, groups, and service principals:

- \$superuser (Owner)
- \$superuser (Owning Group)
- Other
- Mask

Permissions for: \$superuser

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

Add user, group, or service principal:  
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.

T-SQL or PySpark auto-generated.

rawdata > taxidata

Name

- part-00000-0300809f-304e-44bc-81bd-bbd63974c3e4-c000.snappy.parquet 8/27/2022
- New SQL script > Select TOP 100 rows
- New notebook > Create external table
- New data flow Bulk load
- New integration dataset e4-c000.snappy.parquet 8/27/2022
- Manage access... 5f-c000.snappy.parquet 8/27/2022

```

1 SELECT
2     TOP 100 *
3     FROM
4     OPENROWSET(
5         BULK 'https://azuresynapsesa.dfs.core.windows.net/rawdata/taxidata/part-00000-0300809f-304e-44bc-81bd-bbd63974c3e4-c000.snappy.parquet',
6         FORMAT='PARQUET'
7     ) AS [result]

```

View Table Chart Export results

VendorID	TpepPickupDate	TpepDropoffDate	PassengerCount	TripDistance	PuLocationId	DoLocationId	...
VTS	2014-04-30T23:59:59.998Z	2014-05-01T00:00:00.000Z	1	5.21	NULL	NULL	-
CMT	2014-04-30T23:59:59.998Z	2014-05-01T00:00:00.000Z	1	21.8	NULL	NULL	-
VTS	2014-04-30T23:59:59.998Z	2014-05-01T00:00:00.000Z	1	5.57	NULL	NULL	-
CMT	2014-04-30T23:59:59.998Z	2014-05-01T00:00:00.000Z	3	1.8	NULL	NULL	-

rawdata > taxidata

Name

- part-00000-0300809f-304e-44bc-81bd-bbd63974c3e4-c000.snappy.parquet
- New SQL script > iec72b03f65f-c000.snappy.parquet
- New notebook > Load to DataFrame
- New data flow New Spark table
- New integration dataset bd63974c3e4-c000.snappy.parquet
- Manage access... iec72b03f65f-c000.snappy.parquet

+ Cell ▾ Run all Undo Publish Attach to analytics1 ...

Ready

Cell 1

```

1 %%pyspark
2 df = spark.read.load('abfss://rawdata@azuresynapsesa.dfs.core.windows.net/taxidata/part-00000-0300809f-304e-44bc-81bd-bbd63974c3e4-c000.snappy.parquet')
3 display(df.limit(10))

```

View Table Chart

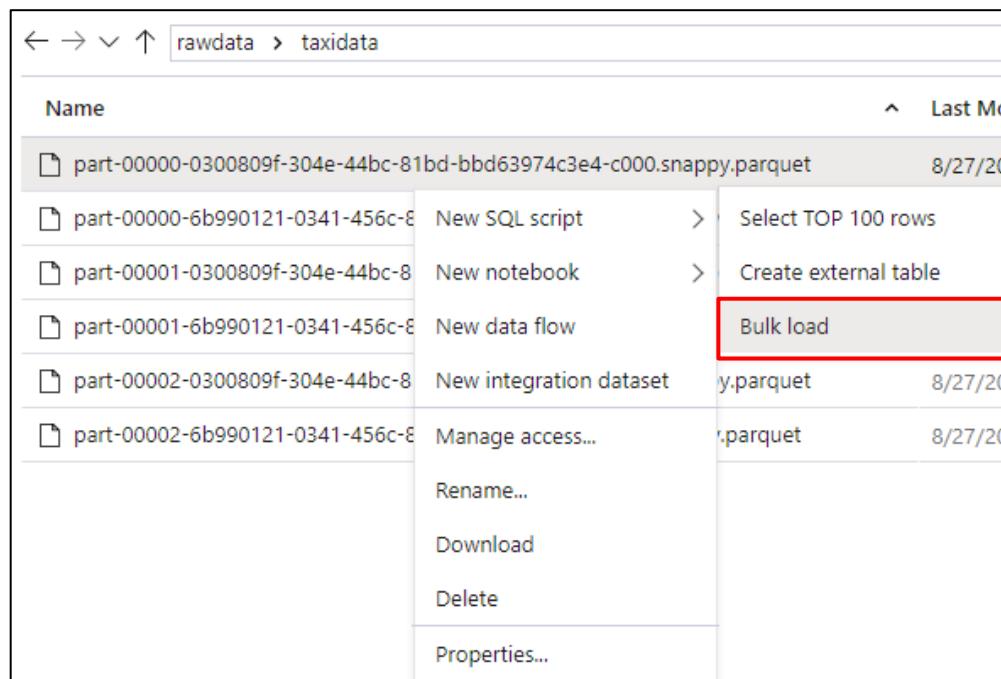
vendorID	tpepPickupDate	tpepDropoffDate	passengerCount	tripDistance	puLocationId	doLocationId	...
VTS	2014-04-30T23:59:59.998Z	2014-05-01T00:00:00.000Z	1	5.21			-
CMT	2014-04-30T23:59:59.998Z	2014-05-01T00:00:00.000Z	1	21.8			-
VTS	2014-04-30T23:59:59.998Z	2014-05-01T00:00:00.000Z	1	5.57			-
CMT	2014-04-30T23:59:59.998Z	2014-05-01T00:00:00.000Z	3	1.8			-

# Data Hub – Bulk load wizard

Bulk load data from the storage container or storage files with Bulk load wizard.

Auto create T-SQL scripts to load data

Offers creation of stored procedure to enable integration with pipeline



**Bulk load**

Source storage location  
Select the storage location where the files containing the data is staged. Azure Data Lake Storage (ADLS) Gen2 and Azure Blob Storage are supported. [Learn more](#)

Storage account  [Edit](#)

Connect via integration runtime \*  [Edit](#)

Input file or folder  [Browse](#)

```

1 --Uncomment the 4 lines below to create a stored procedure for da
2 --CREATE PROC bulk_load_test
3 --AS
4 --BEGIN
5 COPY INTO dbo.test
6 (vendorID 1, tpepPickupDateTime 2, tpepDropoffDateTime 3, passeng
7 FROM 'https://azuresynapsesa.dfs.core.windows.net/rawdata/taxidat
8 WITH
9 (
10     FILE_TYPE = 'PARQUET'
11     ,MAXERRORS = 0
12     ,IDENTITY_INSERT = 'OFF'
13 )
14 --END
15 GO
16
17 SELECT TOP 100 * FROM test
18 GO

```

[Cancel](#)

# Data Hub – Storage accounts

Increased productivity by automatic creation of external file format, external data source and external table.

It also offers generating scripts to create external file format, external data source and external table for manual execution.

A screenshot of the Azure Synapse Analytics Studio interface. On the left, there is a list of files in a data lake, including various parquet files and a py.parquet file. A context menu is open over one of the parquet files, with the 'Create external table' option highlighted by a red box and a red arrow pointing to the 'Create external table' dialog box on the right.

### Create external table

part-00000-0300809f-304e-44bc-81bd-bbd63974c3e4-c000.snappy.parquet

External tables provide a convenient way to persist the schema of data residing in your data lake which can be reused for future adhoc analytics. [Learn more](#)

Select SQL pool \* ①

NYCTaxi\_Pool

Select a database \* ①

NYCTaxi\_Pool

External table name \* ①

dbo.ast\_nyc

Create external table \*

Automatically

Using SQL script

This will automatically create the external table in your database where you can quickly SELECT Top 100 in your SQL script

**Create** **Cancel**

The screenshot shows the Azure Synapse Analytics Studio query editor with two tabs of generated SQL code. The top tab shows a simple query:

```
SELECT TOP 100 * FROM dbo.ast_nyc
GO
```

The bottom tab shows the full generated script:

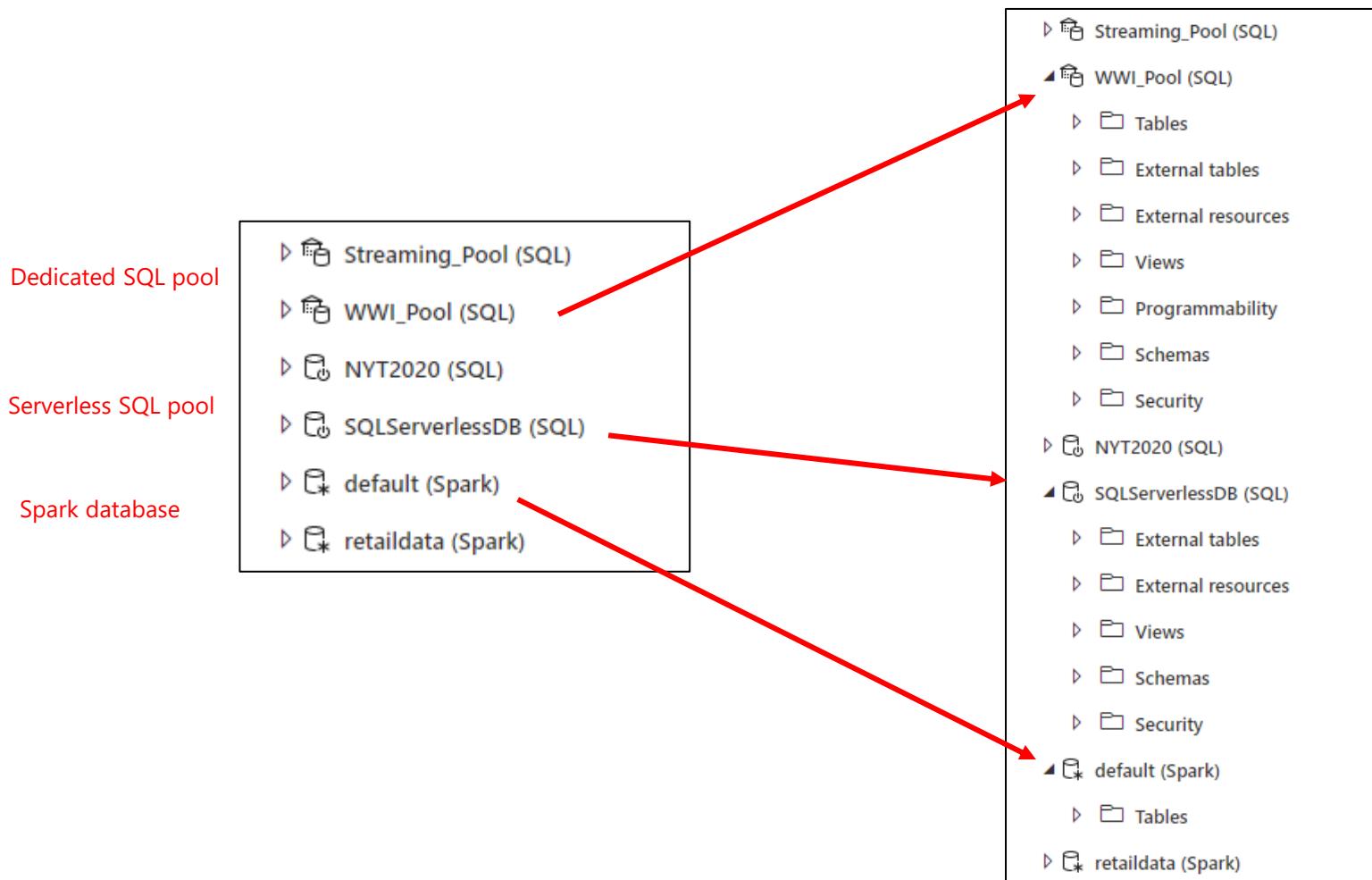
```
IF NOT EXISTS (SELECT * FROM sys.external_file_formats WHERE name = 'SynapseParquetFormat')
CREATE EXTERNAL FILE FORMAT [SynapseParquetFormat]
WITH ( FORMAT_TYPE = PARQUET)
GO

IF NOT EXISTS (SELECT * FROM sys.external_data_sources WHERE name = 'rawdata_azureSynapseA_dfs_core_windows_net')
CREATE EXTERNAL DATA SOURCE [rawdata_azureSynapseA_dfs_core_windows_net]
WITH (
    LOCATION = 'abfss://rawdata@azuresynapsesa.dfs.core.windows.net',
    TYPE = HADOOP
)
GO

CREATE EXTERNAL TABLE dbo.ast_nyc (
    [vendorID] varchar(8000),
    [tpepPickupDateTime] datetime2(7),
    [tpepDropoffDateTime] datetime2(7),
    [passengerCount] int,
    [tripDistance] float,
    [puLocationId] varchar(8000),
    [doLocationId] varchar(8000),
    [startLon] float,
    [startLat] float,
    [endLon] float,
    [endLat] float,
    [rateCodeId] int,
    [storeAndFwdFlag] varchar(8000),
    [paymentType] varchar(8000),
    [fareAmount] float,
    [extra] float,
    [mtaTax] float,
    [improvementSurcharge] varchar(8000),
    [tipAmount] float,
    [tollsAmount] float,
    [totalAmount] float,
    [puYear] int,
    [puMonth] int
)
WITH (
    LOCATION = 'taxidata/part-00000-0300809f-304e-44bc-81bd-bbd63974c3e4-c000.snappy.parquet',
    DATA_SOURCE = [rawdata_azureSynapseA_dfs_core_windows_net],
    FILE_FORMAT = [SynapseParquetFormat],
    REJECT_TYPE = VALUE,
```

# Data Hub – Databases

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



# Data Hub – Databases

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

Databases 3

- sql1 (SQL pool)
  - Tables
    - dbo.SearchLogTable
    - dbo.NycTaxiPredict
      - Columns
        - New SQL script
        - New notebook
        - Select TOP 1000 rows
        - CREATE
        - Drop
        - 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

Databases 3

- sql1 (SQL pool)
  - Tables
    - dbo.SearchLogTable
    - dbo.NycTaxiPredict
      - Columns
        - New SQL script
        - New notebook
        - Refresh
        - Load to DataFrame

Notebook 1 \* X

+ Cell ▾ Run all Publish Attach to Select Spark pool Language PySpark (Python) ▾

Cell 1

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

# Data Hub – 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 Synapse Analytics Studio interface for managing datasets. On the left, a sidebar titled 'Data' lists resources: Storage accounts (2), Databases (3), and Datasets (2). The 'NYCTaxiParquet' dataset is highlighted with a red box and has a red arrow pointing from the sidebar to its main configuration page on the right.

The main page title is 'NYCTaxiParquet X'. It features a Parquet file icon and the dataset name 'NYCTaxiParquet'. Below this, there are tabs for General, Connection, Schema, and Parameters. The Connection tab is selected, showing the following details:

- Linked service: Lake\_ArcadiaLake (dropdown menu)
- File path: data / nyctaxi / File (input fields)
- Compression type: snappy (dropdown menu)
- Action buttons: Test connection, Open, New, Browse, Preview data



# Synapse Studio

## Develop hub

# Develop Hub

## Overview

It provides development experience to query, analyze, model data

## Benefits

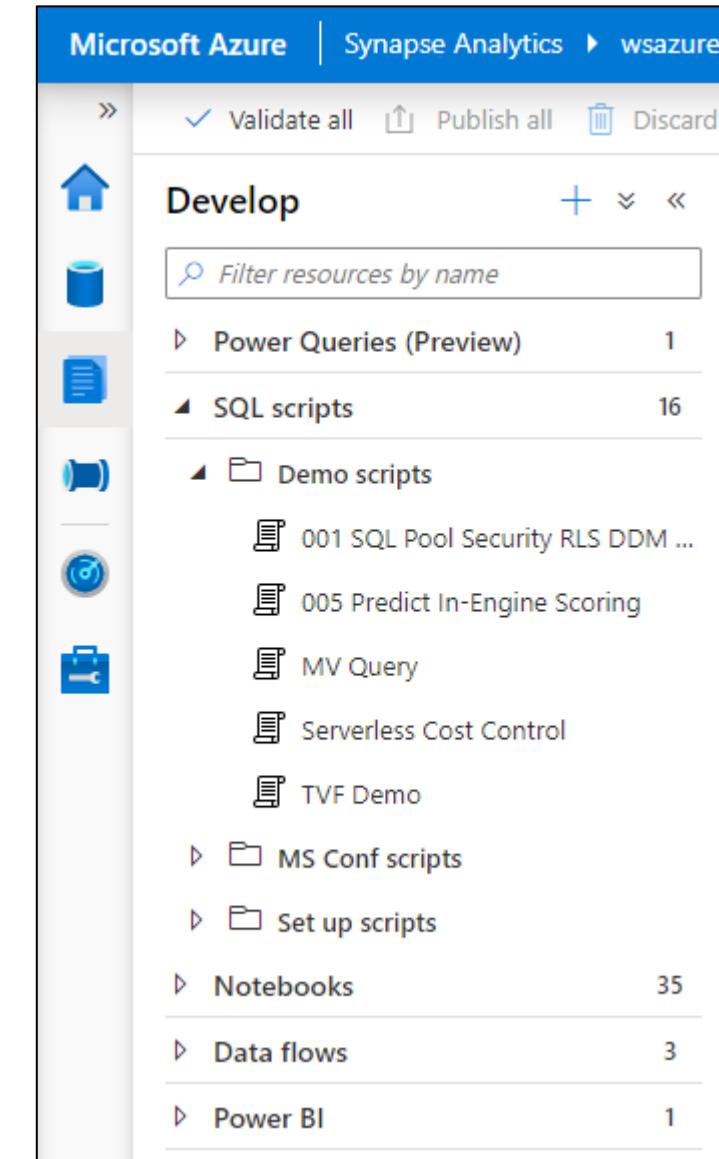
Multiple languages to analyze data under one umbrella

Switch over notebooks and scripts without loosing content

Code intellisense offers reliable code development

Create insightful visualizations

Organize artifacts in folders and sub-folders



# Develop Hub - SQL scripts

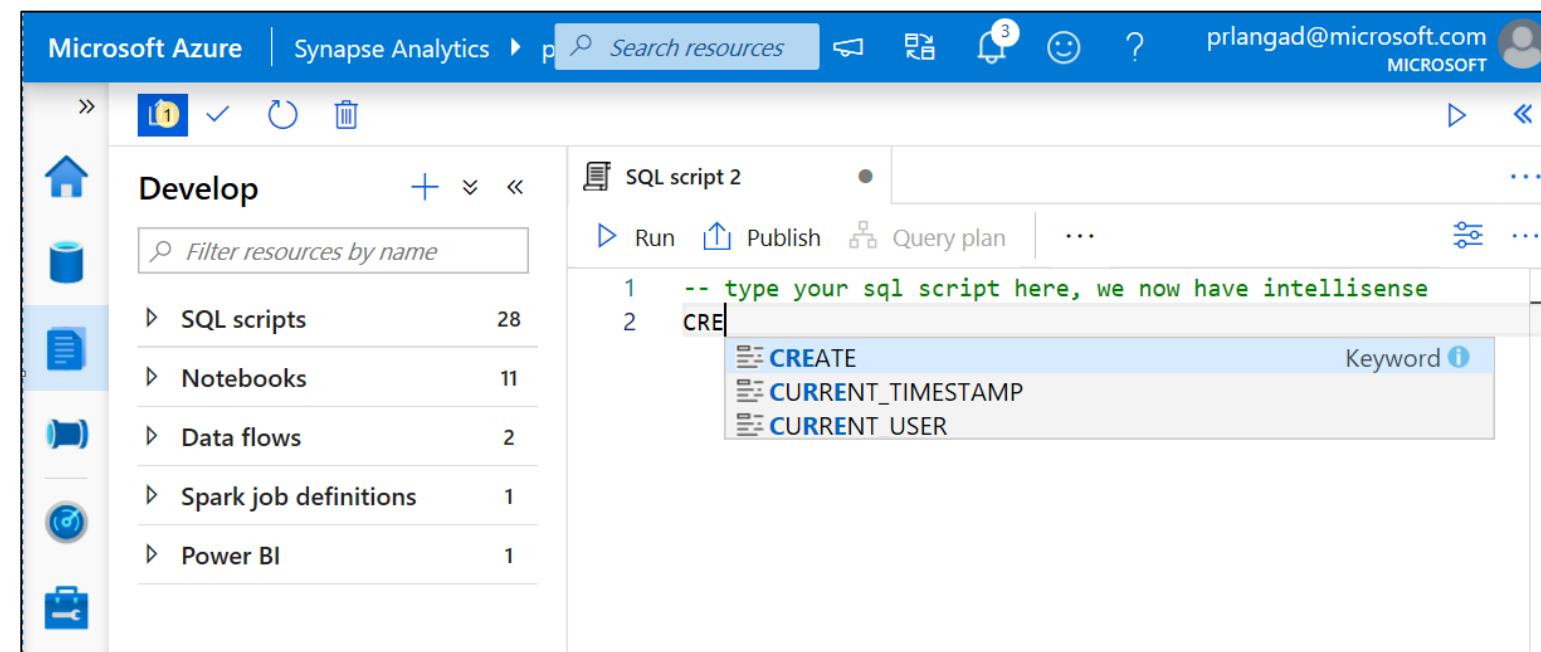
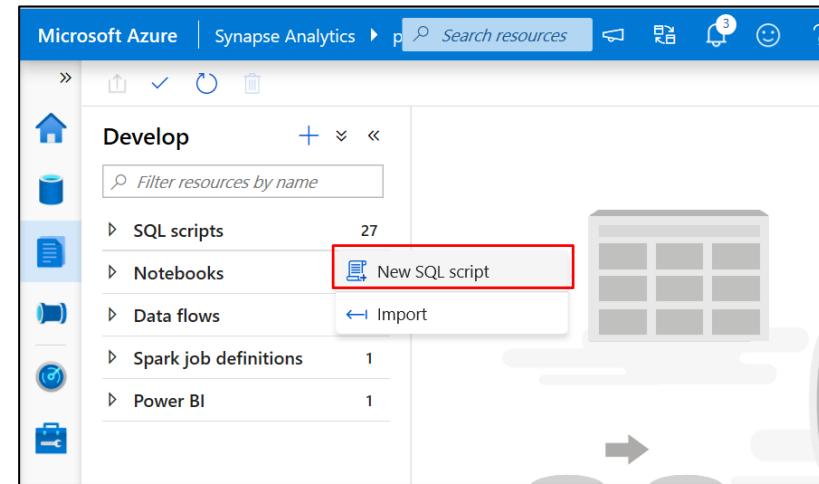
## SQL Script

### Authoring SQL Scripts

Execute SQL script on dedicated SQL pool or serverless SQL pool

Commit individual SQL script or multiple SQL scripts through Commit all feature

Language support and intellisense



# Develop Hub - SQL scripts

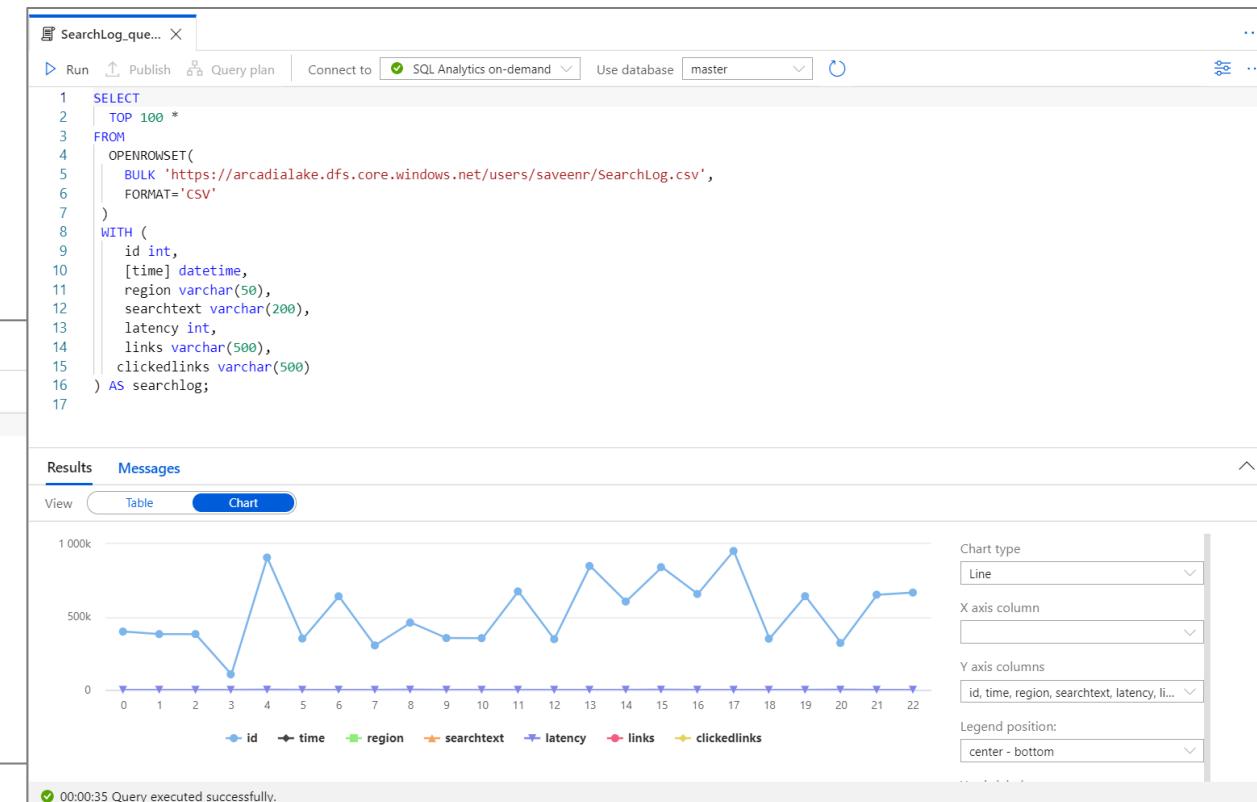
## SQL Script

View results in Table or Chart form and export results in several popular formats

The screenshot shows the Azure Synapse Analytics Develop Hub interface. On the left, a code editor displays a T-SQL script for reading CSV data from a blob storage location and creating a temporary table named 'searchlog'. The script uses OPENROWSET and BULK options to read the data.

Below the code editor, the results pane is visible. It has tabs for 'Results' (highlighted with a red box) and 'Messages'. Under 'Results', there are 'Table' and 'Chart' views. The 'Table' view is selected, showing a list of search log entries with columns ID, TIME, and REGION. The 'Chart' view shows a line chart of the data over time. A red arrow points from the 'Table' tab in the results pane to the 'Table' tab in the chart pane.

On the right, a separate window titled 'SearchLog\_que...' shows the same T-SQL script and its execution results. The results are displayed in a 'Table' view, showing the same three columns: ID, TIME, and REGION. Below the table, a red box highlights the 'Export results' dropdown menu, which lists four options: CSV, Excel, JSON, and XML.



# Develop Hub - Notebooks

## Notebooks

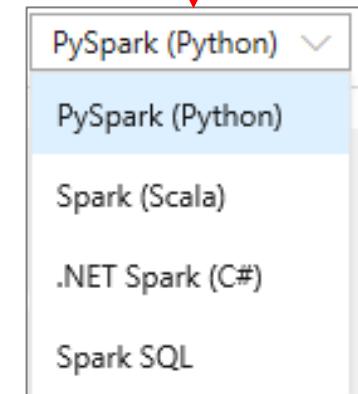
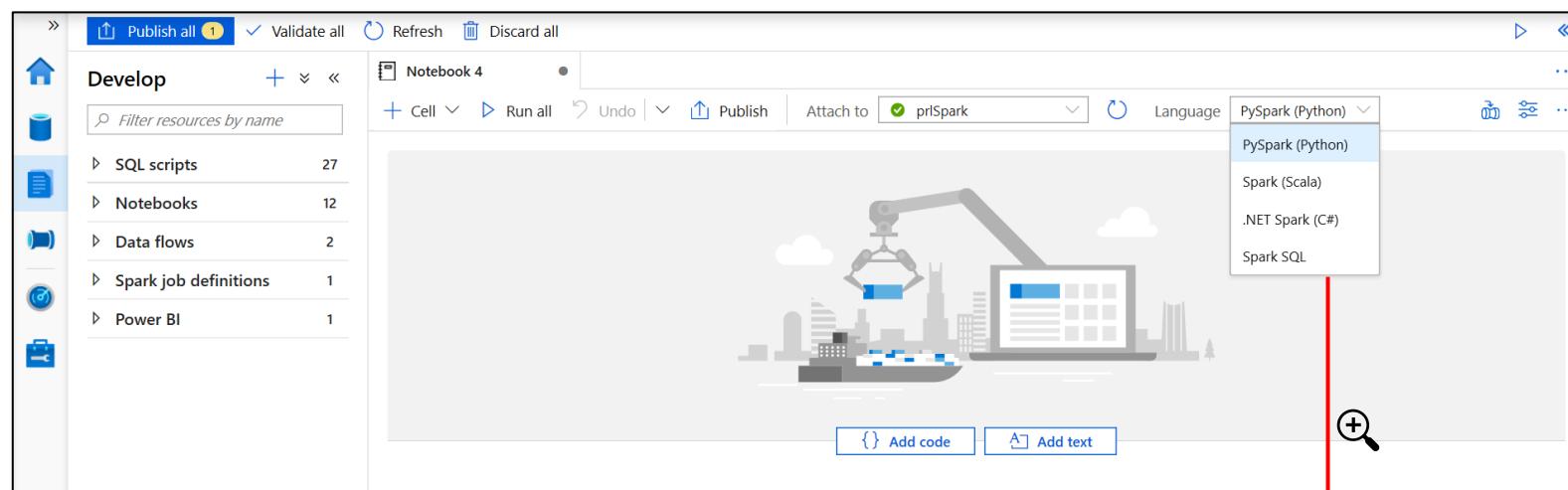
Allows to write multiple languages in one notebook

`%%<Name of language>`

Offers use of temporary tables across languages

Language support for Syntax highlight, syntax error, syntax code completion, smart indent, code folding

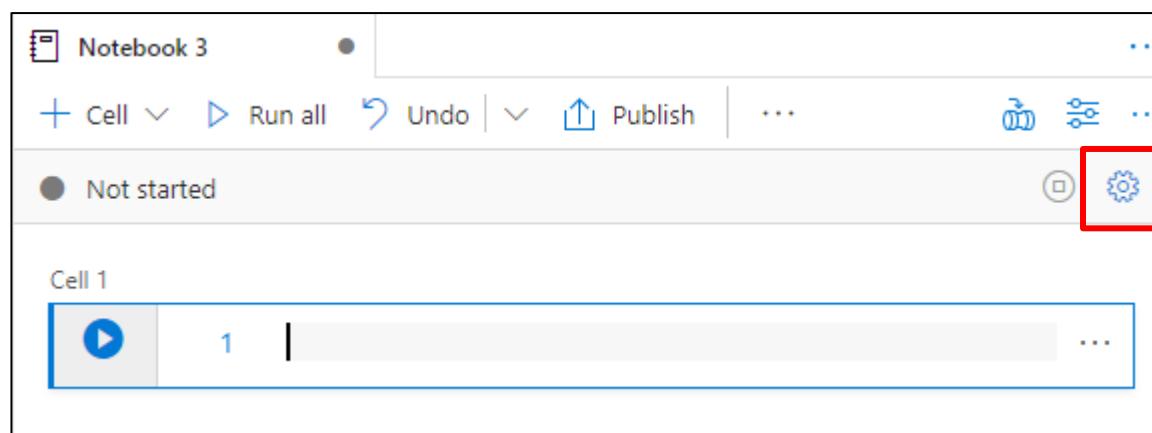
Export results



# Develop Hub - Notebooks

Configure session allows developers to control how many resources are devoted to running their notebook.

Provides quick links to monitor session and Spark history server



### Configure session

Livy session ID

Status  
Not started

Attach to \* ⓘ  
analytics1

analytics1  
Refresh at 12:04:28 AM

Medium (8 vCores / 56 GB) 3 - 10 nodes  
0.00% utilized

Available session sizes ⓘ

Small	19 executors	<a href="#">Use</a>
Medium	9 executors	<a href="#">Use</a>

Executor size \* ⓘ  
Small (4 vCores, 28GB memory)

Executors \* ⓘ  
2

Driver size \* ⓘ  
Small (4 vCores, 28GB memory)

Session timeout (minutes) \* ⓘ  
30

[Apply](#) [Cancel](#)

# Develop Hub - Notebooks

As notebook cells run, the underlying Spark application status is shown. Providing immediate feedback and progress tracking.

The screenshot shows the Microsoft Azure Synapse Analytics Develop Hub - Notebooks interface. At the top, there's a navigation bar with 'Microsoft Azure' and 'Synapse Analytics'. A search bar says 'Search resources' and there are icons for notifications and help. On the right, it shows the email 'prlangad@microsoft.com' and the 'MICROSOFT' logo.

In the main area, there's a toolbar with 'Publish all', 'Validate all', 'Refresh', and 'Discard all'. Below that, a breadcrumb trail shows 'opendataset' and 'Notebook 1'. The toolbar also includes 'Cell', 'Run all', 'Undo', 'Publish', 'Attach to' (set to 'SparkPoolDef'), 'Language' (set to 'PySpark (Python)'), and other options.

The main content area is titled 'Cell 1'. It contains a code cell with the following Python code:

```
%pyspark
data_path = spark.read.load('abfss://opendataset@internalsandboxwe.dfs.core.windows.net/holidays/part-00000-bd1ab'
                            ...
data_path.show(100)
```

Below the code, a message says 'Command executed in 2mins 44s 998ms by prlangad on 03-19-2020 11:31:56.458 -07:00'.

Underneath the code cell, there's a section titled 'Job execution Succeeded' for 'Spark 2 executors 8 cores'. It lists three jobs:

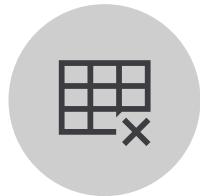
ID	DESCRIPTION	STATUS	STAGES	TASKS	SUBMISSION TIME	DURATION
Job 0	load at NativeMethodAccessImpl.java:0	<span style="color: green;">✓ Succeeded</span>	1/1	<div style="width: 100%; background-color: #2e8b57;"></div>	3/19/2020, 11:31:35 AM	6s
Job 1	showString at NativeMethodAccessImpl.java:0	<span style="color: green;">✓ Succeeded</span>	1/1	<div style="width: 100%; background-color: #2e8b57;"></div>	3/19/2020, 11:31:43 AM	1s
Job 2	showString at NativeMethodAccessImpl.java:0	<span style="color: green;">✓ Succeeded</span>	1/1	<div style="width: 100%; background-color: #2e8b57;"></div>	3/19/2020, 11:31:45 AM	9s

At the bottom of the cell, there's a preview of the data:

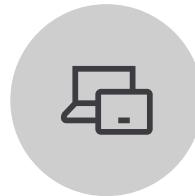
```
+---+-----+-----+-----+-----+-----+-----+-----+
|vendorID| tpepPickupDateTime|tpepDropoffDateTime|passengerCount|tripDistance|puLocationId|doLocationId| startLon| startLat|
|endLon| endLat|rateCodeId|storeAndFwdFlag|paymentType|fareAmount|extra|mtaTax|improvementSurcharge|tipAmount|tollsAmount|
|totalAmount|
+---+-----+-----+-----+-----+-----+-----+-----+
|      CMT|2009-04-30 23:59:52|2009-05-01 00:11:14|          1|        1.9|       null|      null|-73.984708|      null|     1.8| | |
|40.760237|-73.960426|40.761527|         null|        0|        Credit|        8.5|       0.0|      null|      null|      null|           |
|          0.0|        10.3|          null|          null|        0|          null|        null|       null|      null|      null|           |
|      CMT|2009-05-07 01:03:26|2009-05-07 01:14:11|          1|        3.4|       null|      null|-73.956527|      null|     2.55|
|40.771307|-73.941002|40.80763|         null|        0|        Credit|        9.7|       0.0|      null|      null|      null|           |
|          0.0|        12.25|          null|          null|        1|          null|        2.2|       null|      null|      null|           |
|      CMT|2009-04-30 23:50:42|2009-05-01 00:06:43|          1|        2.2|       null|      null|-74.009102|      null|           |
+---+-----+-----+-----+-----+-----+-----+-----+
```

At the very bottom, there are buttons for 'Ready' (with a checkmark), 'Stop session', and 'Configure session'.

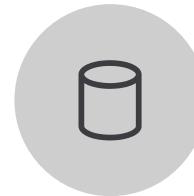
# Dataflow Capabilities



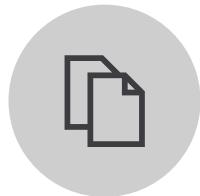
Handle upserts, updates, deletes on sql sinks



Add new partition methods



Add schema drift support



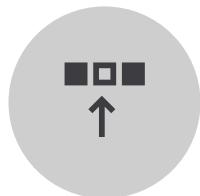
Add file handling (move files after read, write files to file names described in rows etc)



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)

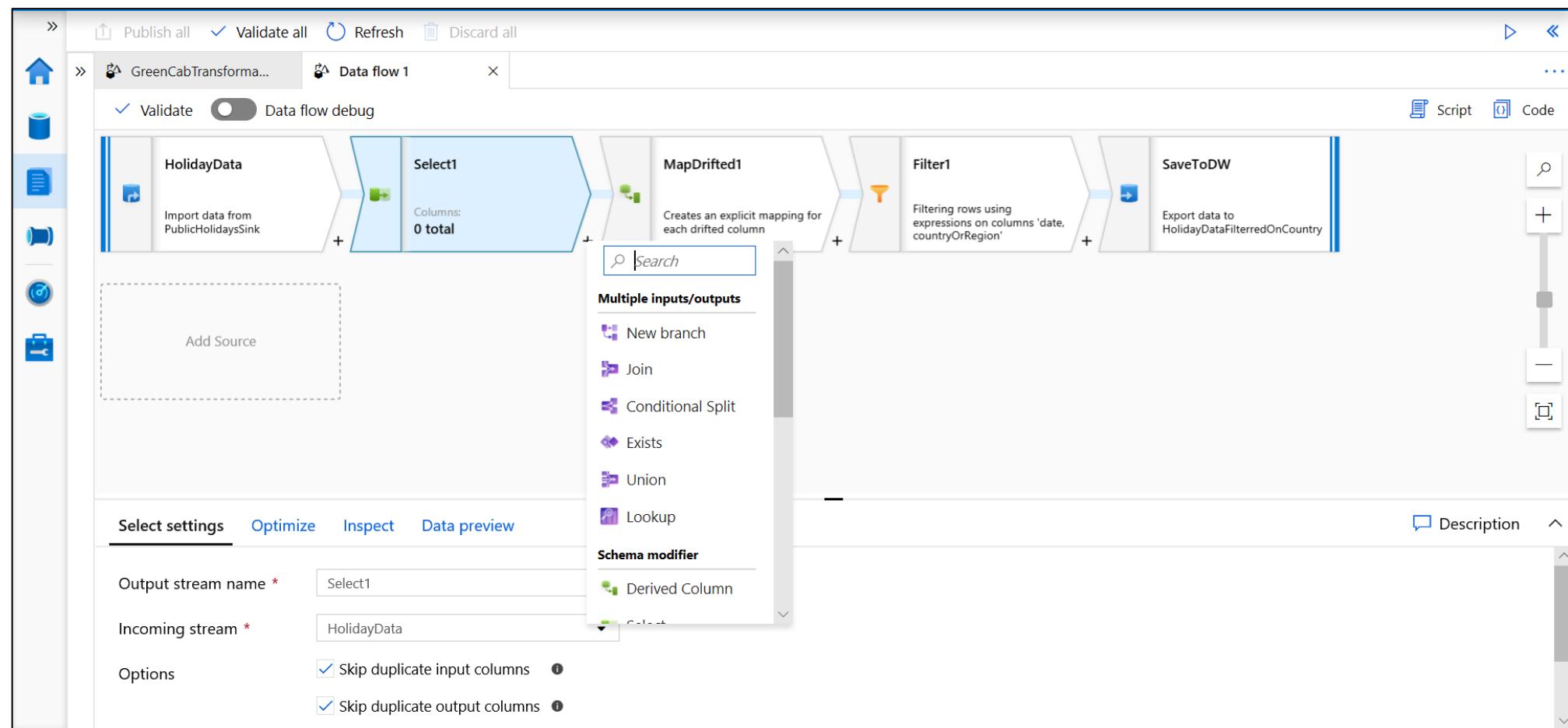


Implement commonly used ETL patterns as templates(SCD Type1, Type2, Data Vault)

# Develop Hub - Data Flows

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

Provides a code-free experience.



# Develop Hub – Power BI

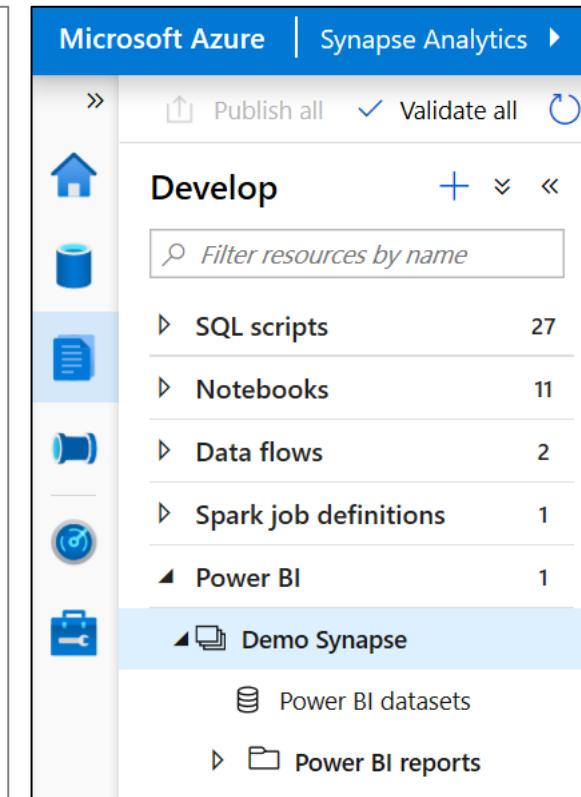
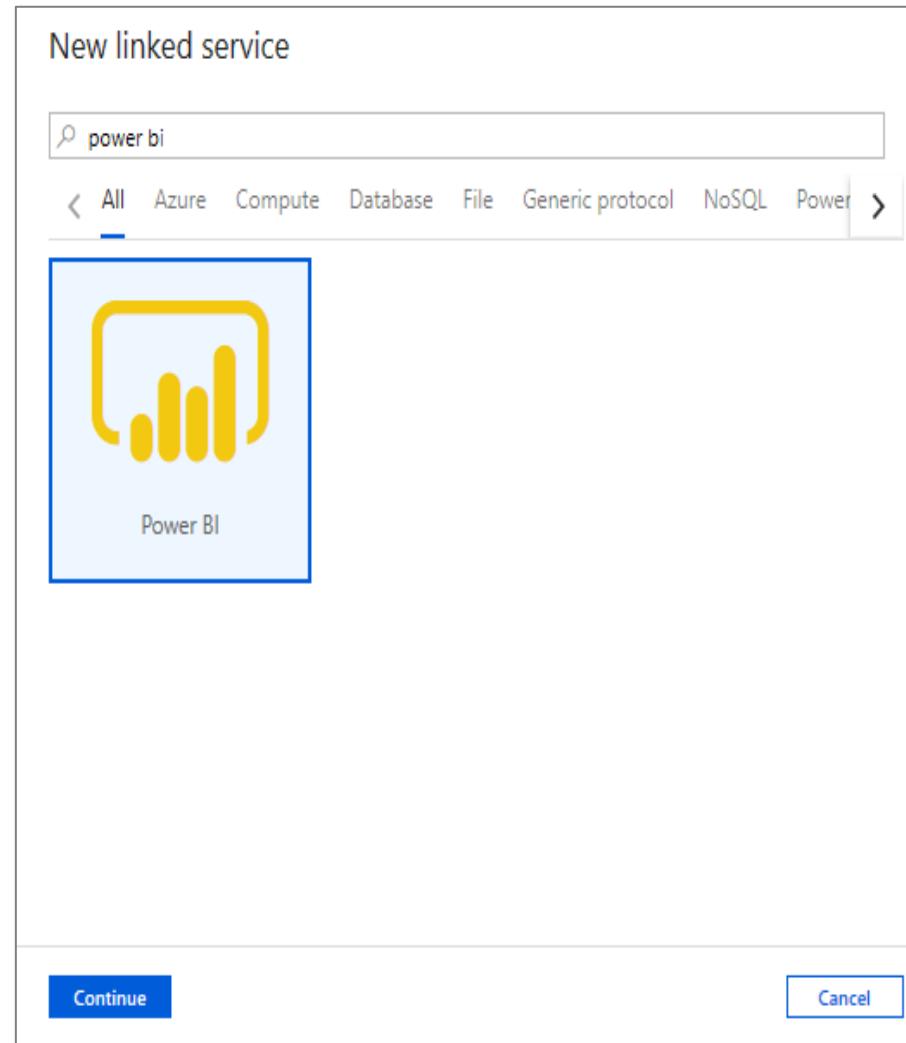
## Overview

Create Power BI reports in the workspace

Provides access to published reports in the workspace

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

Visually explore and analyze data



# Develop Hub – Power BI

View published reports in Power BI workspace

The screenshot shows the Microsoft Azure Power BI workspace interface. The left sidebar displays the 'Develop' section with a list of resources: SQL scripts (9), Notebooks (6), Data flows (1), Power BI (1), and a folder named 'gaming-telemetry' containing Power BI datasets and reports. The 'Report' item under 'Power BI reports' is selected.

The main content area displays a report titled 'GAME STUDIO'. The report features a header with a game console and controller image, followed by a 'What If...' section asking 'We increase free game addons by: 1'. It includes two large callout boxes: 'Users (Forecast) 7,361,707' (7,346,291 last month) and 'Extra Users 252.8K' (+3.4% Users Increase). Below these are two visualizations: a 'Total Users vs "What If" Analysis' table and a 'What If' Analysis Forecast' line chart.

The 'Total Users vs "What If" Analysis' table provides detailed user data by region and age group:

Region	Users	Forecast	Extra Users
APAC	1,268.5K	1,273.7K	45.4K
18-22	96.8K	97.7K	4.0K
22-26	436.0K	435.5K	13.4K
26-30	462.9K	464.0K	15.6K
30-34	75.0K	76.3K	3.4K
34-40	24.0K	24.2K	1.1K
41-60	27.1K	27.5K	1.3K
>60	146.7K	148.5K	6.7K
EMEA	844.9K	846.5K	30.4K
18-22	66.8K	67.5K	2.7K
22-26	291.8K	290.7K	9.1K
26-30	306.9K	307.1K	10.4K
30-34	50.4K	50.9K	2.3K
34-40	16.3K	16.4K	0.7K
41-60	18.5K	18.7K	0.9K
Total	7,346.3K	7,361.7K	252.8K

The 'What If' Analysis Forecast' chart shows the projected growth of users over time from August 2019 to November 2019, with a purple line representing the forecast and a grey shaded area representing the confidence interval.

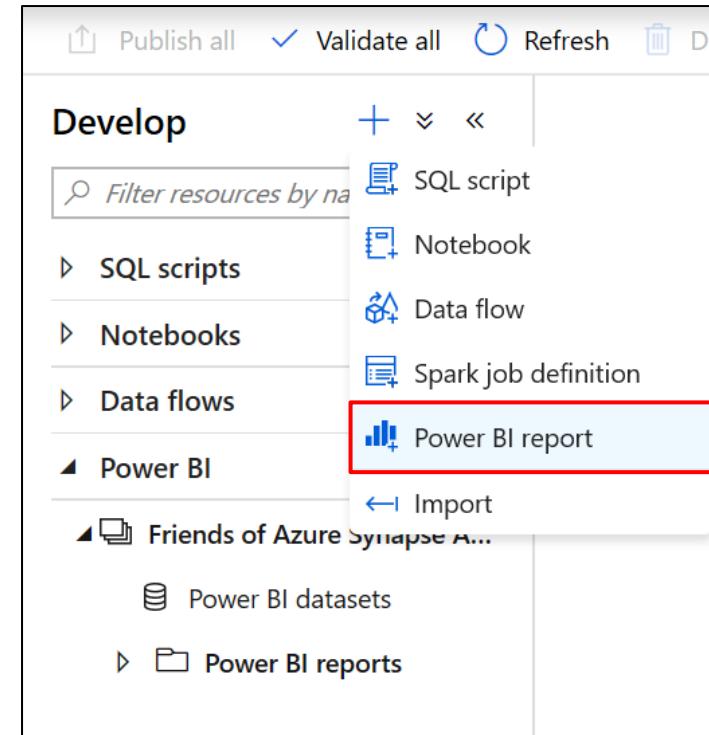
The bottom navigation bar includes tabs for 'Historical', 'Forecast', 'Predictions', and a plus sign icon. The right sidebar contains sections for 'VISUALIZATIONS' (with icons for various chart types) and 'FIELDS' (with a search bar and a list of fields like agegroup, forecast, historical, platform, predictions, realtime, regions, scenario, and weekdays).

# Develop Hub – Power BI

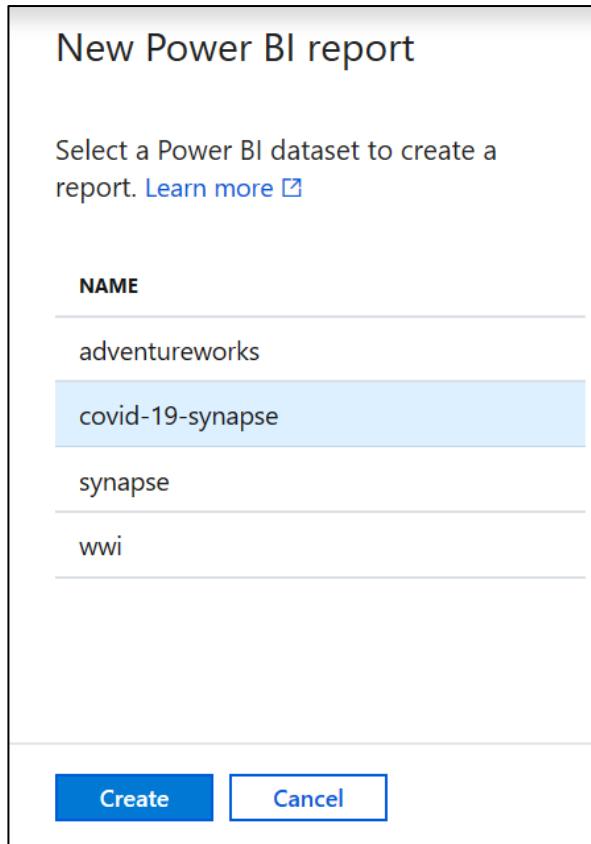
Create new reports from existing published Power BI datasets

Create new Power BI datasets

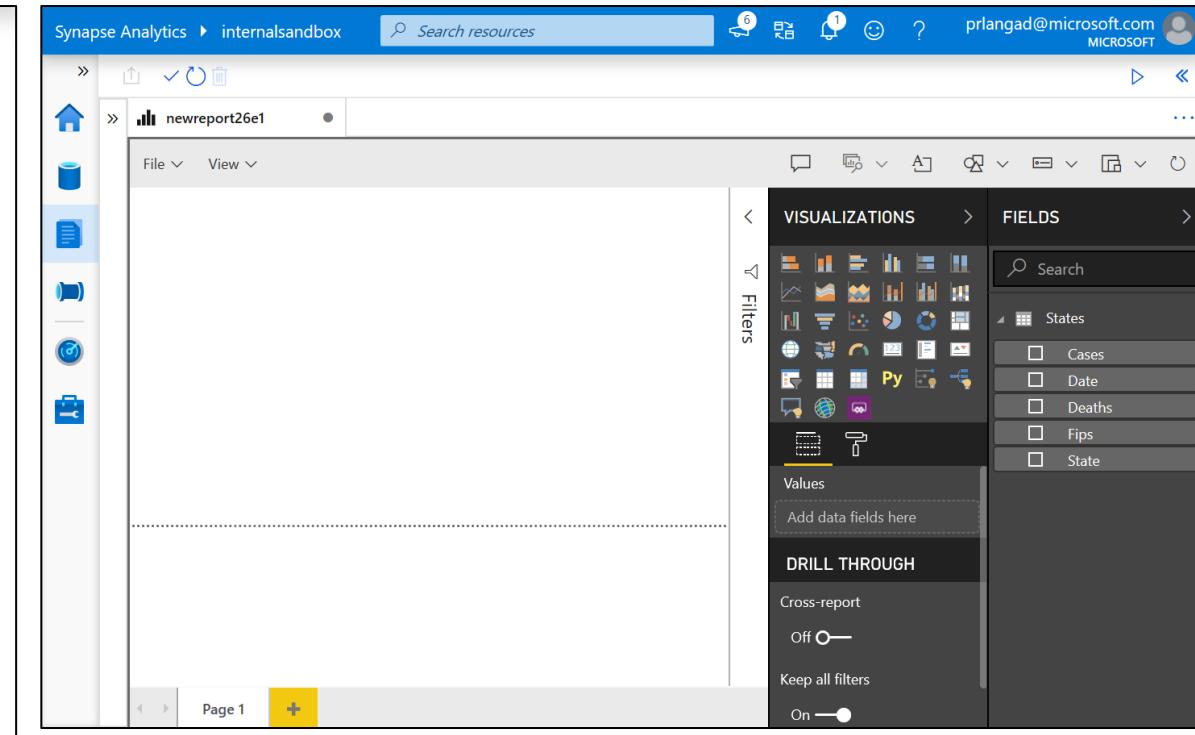
1



2



3



# Develop Hub – Power BI

Edit reports in Synapse workspace

The screenshot shows the Microsoft Azure Synapse Analytics Develop Hub interface. On the left, the navigation pane is open with the 'Develop' section selected. Under 'Power BI', the 'Report' item is highlighted. The main area displays a Power BI report titled 'GAME STUDIO'. The report features a central image of a white Xbox controller on a console. Key statistics are displayed: 'Total Users' at 24.5M, 'Users (Forecast)' at 7,361,707, and 'Extra Users' at 252.8K. A 'What If...' card shows a slider for increasing free game addons by 1. Below these are two tables: 'Total Users vs "What If" Analysis' and 'What If" Analysis Forecast'. The forecast table shows user counts for various regions and age groups. The forecast chart tracks user growth from August 2019 to November 2019. The right side of the screen contains the 'VISUALIZATIONS' and 'FIELDS' panes, which provide options for modifying the report's components and data fields.

**GAME STUDIO**

**What If...**  
We increase free game addons by:  
1

**Total Users vs "What If" Analysis**

Region	Users	Forecast	Extra Users
APAC	1,268.5K	1,273.7K	45.4K
18-22	96.8K	97.7K	4.0K
22-26	436.0K	435.5K	13.4K
26-30	462.9K	464.0K	15.6K
30-34	75.0K	76.3K	3.4K
34-40	24.0K	24.2K	1.1K
41-60	27.1K	27.5K	1.3K
>60	146.7K	148.5K	6.7K
EMEA	844.9K	846.5K	30.4K
18-22	66.8K	67.5K	2.7K
22-26	291.8K	290.7K	9.1K
26-30	306.9K	307.1K	10.4K
30-34	50.4K	50.9K	2.3K
34-40	16.3K	16.4K	0.7K
41-60	18.5K	18.7K	0.9K
<b>Total</b>	<b>7,346.3K</b>	<b>7,361.7K</b>	<b>252.8K</b>

**"What If" Analysis Forecast**

Users Forecast

Date

Historical Forecast Predictions

Visualizations Fields

# Develop Hub – Power BI

Publish edited reports in Synapse workspace to Power BI workspace

The screenshot shows the Microsoft Azure Synapse Analytics Develop Hub interface. On the left, a sidebar lists resources under 'Develop': SQL scripts (9), Notebooks (6), Data flows (1), Power BI (1), and Power BI datasets, Power BI reports, and Reports (selected). A red arrow points from the text 'Publish changes by simple save report in workspace' to the 'Save this report' button in the top navigation bar, which is highlighted with a red box.

The main area displays a Power BI report titled 'GAME STUDIO'. The report features a large image of an Xbox console and controller. Key statistics include 'Total Users 24.5M' and a 'What If...' analysis showing 'Users (Forecast) 7,613,619' and 'Extra Users 504.8K'. The report includes a table of 'Total Users vs "What If" Analysis' and a line chart of 'What If' Analysis Forecast over time from August 2019 to November 2019.

On the right, the 'VISUALIZATIONS' and 'FIELDS' panes are open, showing various chart and field options. The 'VALUES' section has 'Add data fields here' and 'DRILL THROUGH' has 'Cross-report Off' and 'Keep all filters On'.

Publish changes by simple save  
report in workspace



File ▾ View ▾ Edit report ▾ Explore ▾ Refresh ▾ Pin a live Page

 Reset to default

Comments

Bookmarks ▾

Usage metrics

View related

Favorite

Subscribe

Share



Home

Favorites

Recent

Apps

Shared with me

Workspaces

gaming-telemetry

Real-time publish on save



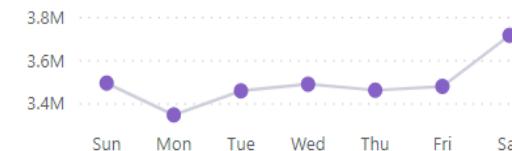
Historical

Forecast

Predictions

Select a Platform:

Console

Total Users  
**24.5M**

## GAME STUDIO

### What If...

We increase free game addons by:

2

Users (Forecast)

**7,613,619**7,346,291  
Last month

Extra Users

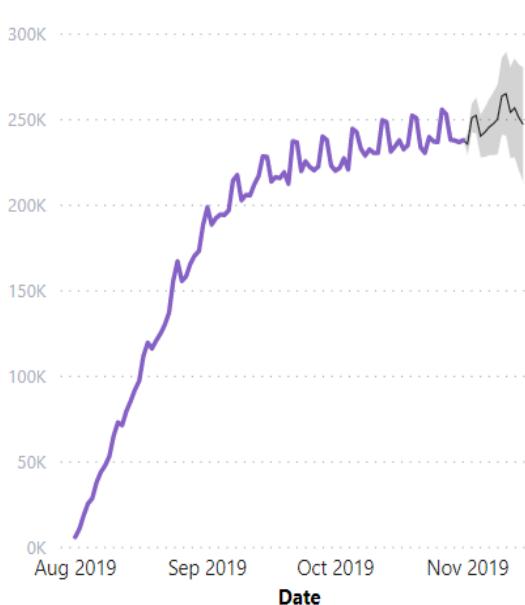
**504.8K**+6.9%  
Users Increase

### Total Users vs "What If" Analysis

Region	Users	Forecast	Extra Users
APAC	<b>1,268.5K</b>	<b>1,319.0K</b>	<b>90.7K</b>
	18-22 96.8K	101.8K	8.1K
	22-26 436.0K	448.9K	26.7K
	26-30 462.9K	479.5K	31.1K
	30-34 75.0K	79.7K	6.7K
	34-40 24.0K	25.3K	2.2K
	41-60 27.1K	28.8K	2.5K
EMEA	<b>844.9K</b>	<b>876.7K</b>	<b>60.7K</b>
	18-22 66.8K	70.2K	5.5K
	22-26 291.8K	299.6K	18.0K
	26-30 306.9K	317.5K	20.9K
	30-34 50.4K	53.2K	4.5K
	34-40 16.3K	17.2K	1.5K
	41-60 18.5K	19.6K	1.8K
Total	<b>7,346.3K</b>	<b>7,613.6K</b>	<b>504.8K</b>

### "What If" Analysis Forecast

● Users ● Forecast



Tabular

Map

Forecast

Extra Users

# Develop – CI/CD

Commit artifacts to source-controlled repository and operationalize release pipelines with Synapse deployment task

The screenshot illustrates the integration of Azure Synapse Analytics with CI/CD pipelines. On the left, a GitHub repository named 'synapsetestdemo-ws-01' is shown with its commit history. On the right, an Azure DevOps pipeline interface displays a 'Synapse deployment v2' release stage.

**GitHub Repository:**

- credential: Adding linkedService: synapsedemosws-WorkspaceDefaultStorage
- dataflow: Updating integrationRuntime: AutoResolveIntegrationRuntime
- dataset: Updating integrationRuntime: AutoResolveIntegrationRuntime
- integrationRuntime: Adding linkedService: synapsedemosws-WorkspaceDefaultStorage
- linkedService: Updating integrationRuntime: AutoResolveIntegrationRuntime
- notebook: Updating integrationRuntime: AutoResolveIntegrationRuntime
- pipeline: Updating integrationRuntime: AutoResolveIntegrationRuntime
- sparkJobDefinition: Adding linkedService: synapsedemosws-WorkspaceDefaultStorage
- sqlscript: Updating integrationRuntime: AutoResolveIntegrationRuntime
- README.md: Initial commit

**Azure DevOps Pipeline:**

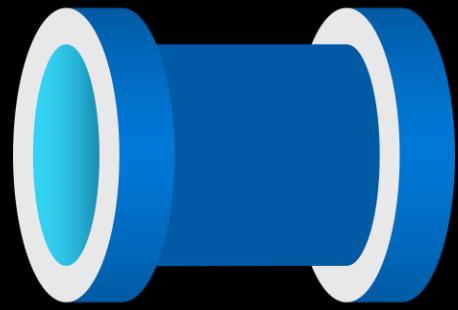
**Synapse deployment v2 > Release-13**

- Pipeline:** Pipeline tab selected.
- Variables:** Variables tab.
- History:** History tab.
- Deploy:** Deploy button.
- Cancel:** Cancel button.
- Refresh:** Refresh button.
- Edit:** Edit button.
- ...** More options.

**Release:** Manually triggered by Priyanka Langade on 11/16/2020, 11:02 PM.

**Artifacts:** azuresynapsesdev 0c8b1a872 branch-retail-12

**Stages:** Load to Prod Succeeded on 11/17/2020, 4:54 PM.



# Synapse Studio

## Integrate hub

# Integrate Hub

It provides ability to create pipelines to ingest, transform and load data with 90+ inbuilt connectors.

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

The screenshot shows the Azure Synapse Integrate Hub interface. On the left, there are three collapsed sections: 'Synapse' (Notebook, Spark job definition, Stored procedure), 'Move & transform' (Copy data, Data flow), and 'Machine Learning' (ML Batch Execution, ML Update Resource, ML Execute Pipeline). Red arrows point from each of these sections to a red-bordered list of activities on the right. This list includes: Synapse, Move & transform, Azure Data Explorer, Azure Function, Batch Service, Databricks, Data Lake Analytics, General, HDInsight, Iteration & conditionals, and Machine Learning. The 'Move & transform' section is currently expanded, showing two 'Copy data' activities: 'GreenTaxi' and 'YellowTaxi'. The 'General' tab is selected in the pipeline editor on the right, where a new pipeline named 'Copy Open Dataset' is being configured. Other tabs include Parameters, Variables, and Output.



# Synapse Studio Monitor hub

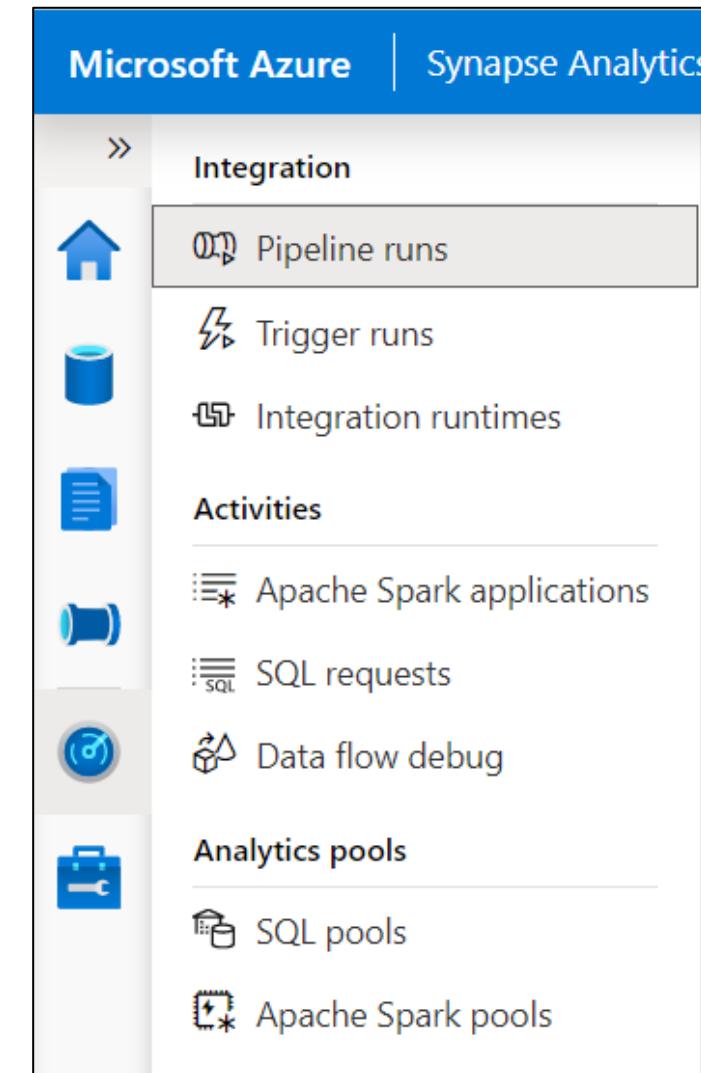
# Monitor Hub

## Overview

This feature provides single pane of glass to monitor orchestration, activities for Apache Spark Application and SQL requests.

## Benefits

Offers additional filters to monitor specific activities or orchestration



# Monitor Hub - Integration

## Overview

Monitor orchestration in the Synapse workspace for the progress and status of pipeline

Pipeline runs				
		Time : Last week (10/24/2019 9:44 AM - 10/31/2019 9:44 AM)	Time zone : Pacific Time (US & Canada) (UT...)	Runs : Latest runs
All status		Rerun	Cancel	Refresh
Pipeline Name	Run Start	DURATION	Triggered By	Status
Load Data to SQLDW	10/25/2019, 3:49:42 PM	00:10:55	Manual trigger	<span style="color: green;">✓ Succeeded</span>
Copy Open Dataset	10/25/2019, 2:17:54 PM	00:14:12	Manual trigger	<span style="color: green;">✓ Succeeded</span>
Pipeline 1	10/24/2019, 1:23:43 PM	00:00:08	Manual trigger	<span style="color: green;">✓ Succeeded</span>

## Benefits

Track all/specific pipelines

Monitor pipeline run and activity run details

Find the root cause of pipeline failure or activity failure

Trigger runs			
		Time : Last 30 days (3/17/20 11:48 PM - 4/16/20 11:48 PM)	Time zone : Pacific Time (US & Canada) (UT...)
All status		Refresh	Edit columns
Showing 1 - 1 items			
Trigger Name	Trigger Type	Trigger Time	Status
TriggerCopy_csvdata100	ScheduleTrigger	4/10/20, 12:14:00 AM	<span style="color: green;">✓ Succeeded</span>

# Monitor Hub - Spark applications

## Overview

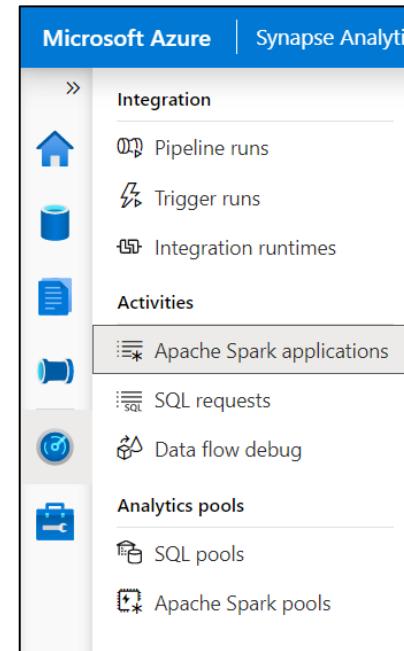
Monitor Spark pools, Spark applications for the status of activities

## Benefits

Apply filter for pool to get Apache Spark Applications per pool

Additional available filters include

1. Application Name
2. Livy ID
3. Status
4. End time



Application name	Submitter	Submit time	Status	Pool
Synapse_automlpool_1...	negust@microsoft.com	11/30/20, 7:23:26 PM	Stopped	automlpool
Synapse_automlpool_1...	negust@microsoft.com	11/30/20, 7:14:00 PM	Stopped	automlpool
Synapse_automlpool_1...	negust@microsoft.com	11/30/20, 5:09:00 PM	Stopped (session timed ou...	automlpool
Notebook 3_analyticscp...	negust@microsoft.com	11/30/20, 12:52:51 PM	Stopped	analyticspool
Notebook 4_analytics1...	prlangad@microsoft.com	11/24/20, 6:29:41 PM	Stopped	analytics1
Synapse_hbpool_16059...	charlesf@microsoft.com	11/20/20, 2:30:44 PM	Stopped	hbpool
Synapse_analyticspool_...	negust@microsoft.com	11/20/20, 11:20:30 AM	Stopped	analyticspool

# Monitor Hub – SQL requests

## Overview

Monitor SQL requests for the progress and status of activities

## Benefits

Apply filter for pool to get SQL requests per compute pool

Validate query text

Additional available filters include

1. Start time
2. End time
3. Request ID
4. Session ID
5. Submitter
6. Workload group

SQL requests						
Request ID ↑↓		Request content ↑↓	Submit time ↑↓	Duration	Submitter ↑↓	Status ↑↓
Showing 1 - 100 of 248 items						
QID125878	USE [DWShellDb]	12/1/20, 12:27:53 AM	0s	System	Completed	0s
QID125879	--Backing up Logical Azure Data	12/1/20, 12:27:53 AM	20s	System	Completed	0s
QID125637	USE [DWShellDb]	11/30/20, 8:27:53 PM	0s	System	Completed	0s
QID125638	--Backing up Logical Azure Data	11/30/20, 8:27:53 PM	15s	System	Completed	0s
QID125529	USE [Predict_Pool]	11/30/20, 6:41:15 PM	0s	anrampal@microsoft.com	Completed	0s
QID125530	SELECT s.NAME AS SchemaName	11/30/20, 6:41:15 PM	0s	anrampal@microsoft.com	Completed	0s
QID125421	USE [Predict_Pool]	11/30/20, 4:54:56 PM	0s	negust@microsoft.com	Completed	0s

Microsoft Azure   Synapse Analytics > wsazuresynapseanalytics					
SQL requests					
Request ID ↑↓		Request content ↑↓	Submit time ↑↓	Duration	Data processed
Showing 1 - 100 of 279 items					
12162198	SELECT TOP 100* FROM OPENROWSET (	11/30/20, 8:57:12 PM	1s	1 MiB	
11573006	SELECT TOP 100* FROM OPENROWSET (	11/30/20, 6:23:32 PM	6s	1 MiB	
9079654	SELECT product = ISNULL(p.pro	11/30/20, 7:28:38 AM	6s	1 MiB	
9066730	SELECT product = ISNULL(p.pro	11/30/20, 7:26:17 AM	5s	1 MiB	
9065769	SELECT * FROM OPENROWSET (	11/30/20, 7:25:47 AM	2s	1 MiB	
9062482	SELECT * FROM OPENROWSET (	11/30/20, 7:25:17 AM	18s	18 MiB	

# Monitor Hub – Analytics pools

**SQL pools**

Refresh Edit columns

Pool : All

Showing 1 - 6 of 6 items

Pool name ↑↓	Type ↑↓	Status ↑↓	Size ↑↓	CPU utilizati... ⓘ ↑↓	Memory utili... ⓘ ↑↓	Created on ↑↓
Built-in	Serverless	✓ Online	Auto	N/A	N/A	N/A
newpoll	Dedicated	● Paused	DW200c	-	-	11/5/2020 5:42:52 AM
NYCTaxi_Pool	Dedicated	✓ Online	DW100c	1.1	23	8/27/2020 6:03:32 AM
Predict_Pool	Dedicated	✓ Online	DW1000c	-	5.33	9/9/2020 1:13:26 AM
Streaming_Pool	Dedicated	● Paused	DW2000c	-	-	8/27/2020 4:04:18 AM
WWI_Pool	Dedicated	✓ Online	DW100c	0.12	20.22	8/26/2020 7:18:23 PM

**Apache Spark pools**

Refresh Edit columns

Pool : analytics

Showing 1 - 4 of 4 items

Pool name ↑↓	Size	Active users	Allocated vCores	Allocated memory (GB)	Created on
analytics2	Medium (8 vCores / 64 GB) - 10 nodes	0	0	0	10/30/20, 2:47:16 PM
analytics1	Medium (8 vCores / 64 GB) - 3 to 10 nodes	0	0	0	10/28/20, 3:36:18 AM
AnalyticsPool99	Medium (8 vCores / 64 GB) - 3 to 10 nodes	0	0	0	9/10/20, 7:17:53 AM
analyticspool	Medium (8 vCores / 64 GB) - 3 to 15 nodes	0	0	0	8/26/20, 7:13:44 PM



# Synapse Studio Manage hub

# Manage Hub

## Overview

This feature provides ability to manage Analytics pools, Linked Services, Integration, Security and Source Control.

The screenshot shows the Microsoft Azure Synapse Analytics Manage Hub interface. The left sidebar has a 'Manage' section selected. The main area shows the 'SQL pools' section, which lists six pools: 'Built-in' (Serverless, Online, Auto), 'newpool' (Dedicated, Paused, DW200c), 'NYCTaxi\_Pool' (Dedicated, Online, DW100c), 'Predict\_Pool' (Dedicated, Online, DW1000c), 'Streaming\_Pool' (Dedicated, Paused, DW2000c), and 'WWI\_Pool' (Dedicated, Online, DW100c). There are buttons for '+ New', 'Refresh', and 'System assigned managed identity'.

Name	Type	Status	Size
Built-in	Serverless	Online	Auto
newpool	Dedicated	Paused	DW200c
NYCTaxi_Pool	Dedicated	Online	DW100c
Predict_Pool	Dedicated	Online	DW1000c
Streaming_Pool	Dedicated	Paused	DW2000c
WWI_Pool	Dedicated	Online	DW100c

# Manage – dedicated SQL pools

## Overview

Provides ability to Pause and Resume, change Scale, Assign Tags from Studio.

**Microsoft Azure | Synapse Analytics > wsazuresynapseanalytics**

Synapse live ▾ Validate all Publish all 1

- Home
- Data
- Develop
- Integrate
- Monitor
- Manage**

Analytics pools
 

- SQL pools**
- Apache Spark pools
- External connections
- Linked services

Integration
 

- Triggers
- Integration runtimes
- Security
- Access control
- Credentials
- Managed private endpoints

Source control
 

- Git configuration

**SQL pools**

Serverless SQL pool is immediately available for your workspace. Dedicated SQL pools can be scaled up or down based on your needs.

+ New Refresh System assigned managed identity

Showing 1-6 of 6 items (1 Serverless, 5 Dedicated)

Name	Type
Built-in	Serverless
newpool	Dedicated
NYCTaxi_Pool	Dedicated
Predict_Pool	Pause
Streaming_Pool	Scale
WWI_Pool	Assign tags

**Scale**

NYCTaxi\_Pool

Scaling can impact workload management settings. Consider using the [workload management scale experience](#) in the Azure portal to configure the settings that best align to your workload needs. [Learn more about performance levels](#)

Performance level  DW500c

Estimated price ⓘ

Est. cost per hour  
6.00 USD

Apply Cancel

# Manage – serverless SQL pools

## Overview

It enables to set the budget for amount of data processed within Synapse Studio or T-SQL.

The screenshot shows the Microsoft Azure Synapse Analytics studio interface. On the left, there's a navigation sidebar with Home, Data, Develop, Integrate, Monitor, and Manage sections. Under the Manage section, 'Analytics pools' is selected, which further branches into 'SQL pools' and 'Apache Spark pools'. Below this is an 'External connections' section. The main content area is titled 'SQL pools' and contains a message: 'Serverless SQL pool is immediately available for your workspace. Dedicated SQL pools can constraints. Learn more'. It has a 'New' button, a 'Refresh' button, and a toggle switch for 'System assigned managed identity'. A table lists 1-6 items (1 Serverless, 5 Dedicated). The first item is 'Built-in' (Serverless) and the second is 'newpool' (Cost Control). A red box highlights the 'Cost Control' link for 'newpool'. At the bottom, there's a T-SQL query editor window with the following code:

```

1 sp_set_data_processed_limit
2     @type = N'daily',
3     @limit_tb = 10
4
5 sp_set_data_processed_limit
6     @type= N'weekly',
7     @limit_tb = 100
8
9 sp_set_data_processed_limit
10    @type= N'monthly',
11    @limit_tb = 1000

```

### Cost Control

Workspace Budget limit for a period. [Learn more](#)

**Daily limit** ⓘ

Enable  Disable

**Data used today**  
2 MB

10	TB
----	----

**Weekly limit** ⓘ

Enable  Disable

**Data used this week**  
23 MB

100	TB
-----	----

**Monthly limit** ⓘ

Enable  Disable

**Data used this month**  
2 MB

0	TB
---	----

**Apply** **Cancel**

# Manage – Apache Spark pools

## Overview

Provides ability to Pause, Scale, Assign Tags, upload Spark configuration, packages from Studio and validate role assignments on respective Spark pool

The screenshot shows the Microsoft Azure Synapse Analytics studio interface. On the left, there's a navigation sidebar with icons for Home, Data, Develop, Integrate, Monitor, and Manage. Under the Manage section, 'Analytics pools' is selected, which is highlighted in blue. The main content area is titled 'Apache Spark pool'. It displays a table with four items: 'analyticspool', 'AnalyticsPool99', 'analytics1', and 'analytics2'. Each item has a '...' button to its right, which is highlighted with a red box. A dropdown menu from this button contains several options: 'Auto-pause settings', 'Autoscale settings', 'Packages', 'Spark configuration', 'Assign tags', 'View role assignments', and 'Delete'. The 'Packages' option is also highlighted with a dashed blue box.

This screenshot shows a 'Manage packages' dialog box. At the top, it says 'Manage packages' and 'analyticspool'. Below that is a table with one item: 'requirements.txt'. The 'NAME' column shows 'requirements.txt', the 'SIZE' column shows '73B', and the 'DATE' column shows '9/9/2020, 8:39:40 PM'. There are download and delete icons next to the file name. At the bottom of the dialog are 'Apply' and 'Cancel' buttons.

This screenshot shows a 'View role assignment' dialog box. It starts with a header 'View role assignment' and a 'analyticspool' icon. Below that is a message: 'To manage these pools, users need sufficient Azure RBAC permissions on this workspace, such as the Owner or Contributor role.' A 'Learn more' link is provided. The main content area is titled 'Synapse role assignments on this Apache Spark pool allow users to submit jobs. All Workspace users can view this pool.' Another 'Learn more' link is shown. Below this are filter buttons: 'Filter by name', 'Type : All', and 'Role : All'. The table lists four entries:

Name	Type	Scope
Synapse Administrator	Individual	Workspace (Inherited)
Priyanka Langade	Individual	Workspace (Inherited)
SynapseWsAdmin	Group	Workspace (Inherited)
SynapseWsAdmin@serv		
Charles Feddersen	Individual	Workspace (Inherited)
wsazuresynapseanalytics	Service Principal	Workspace (Inherited)

At the bottom right of the dialog is a 'Manage role assignments (Access control)' link.

# Manage – Linked services

## Overview

It defines 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 > wsazuresynapseanalytics**

Synapse live | Validate all | Publish all

**Linked services**

Linked services are much like connection strings, which define the connection to external resources. [Learn more](#)

**New**

**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

Showing 1 - 15 of 15

Name ↑

- AzureDataExplor
- AzureDataExplor
- AzureMLService1
- AzureMLServiceN
- bing-covid-19-d
- Nellies\_Keyvault

Continue Cancel

# Manage – Triggers

## Overview

It defines a unit of processing that determines when a pipeline execution needs to be kicked off.

## Benefits

### Create and manage

- Schedule trigger
- Tumbling window trigger
- Event trigger

### Control pipeline execution

**Triggers**

To execute a pipeline set the trigger to be kicked off.

+ New

Filter by keyword

Showing 0 - 0 of 0 items

Name ↑↓	Type ↑↓	Status ↑↓	Pipelines ↑↓
---------	---------	-----------	--------------

New trigger

Choose a name for your trigger. This name can be updated at any time until it is published.

Name \*

Description

Type \*

Schedule    Tumbling window    Event

Start Date (UTC) \*

10/29/2019 9:46 PM

Recurrence \*

Every 1 Minute(s)

End \*

No End    On Date

Annotations

+ New

Activated \*

Yes    No

OK   Cancel

# Manage – 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

The screenshot shows the 'Integration runtimes' blade in the Azure portal. On the left, a navigation menu lists 'Analytics pools', 'SQL pools', 'Apache Spark pools', 'External connections', 'Linked services', 'Integration', 'Triggers', 'Integration runtimes' (which is selected and highlighted with a red box), 'Security', 'Access control', and 'Credentials'. At the top right, there are buttons for 'Validate all', 'Publish all', '+ New' (also highlighted with a red box), and 'Refresh'. A search bar labeled 'Filter by keyword' is present. Below the search bar, it says 'Showing 1 - 1 of 1 items'. To the right, a table header includes columns for 'Name' (sorted by Type), 'Type' (sorted by Type), 'Sub-type' (sorted by Sub-type), and 'Status' (sorted by Status). A modal window titled 'Integration runtime setup' is open, asking 'Choose the network environment of the data source/destination or external compute to which the integration runtime will connect to for data movement or dispatch activities'. It shows two options: 'Azure' (represented by a cloud icon) and 'Self-Hosted' (represented by a server icon). At the bottom of the modal are 'Continue', 'Back', and 'Cancel' buttons.

# Manage – Access Control

## Overview

It provides access control management to workspace resources and artifacts for admins

## Benefits

Share workspace with the team

Increases productivity

Assign granular level permissions

Manage permissions on Spark pools,  
Integration Runtimes, Linked services,  
Credentials

The screenshot shows the Microsoft Azure Synapse Analytics Access Control interface. On the left, there's a sidebar with icons for Analytics pools, SQL pools, Apache Spark pools, External connections, Linked services, Orchestration, and Triggers. The main area is titled 'Access control' and shows a table of existing role assignments:

NAME	TYPE	ROLE
soft.com	Individual	Workspace admin

A red box highlights the '+ Add' button, and a red arrow points from it down to the 'Add role assignment' dialog box. Another red arrow points from the 'soft.com' row in the table to the 'Add role assignment' dialog box.

**Add role assignment**

Grant others access to this workspace by assigning roles to users, groups, and/or service principals. [Learn more](#)

**Scope \***  Workspace  Workspace item

**Role \***  Select a role  
 Filter...

- Synapse Administrator
- Synapse SQL Administrator
- Synapse Apache Spark Administrator
- Synapse Contributor (preview)
- Synapse Artifact Publisher (preview)
- Synapse Artifact User (preview)
- Synapse Compute Operator (preview)
- Synapse Credential User (preview)

**Add role assignment**

Grant others access to this workspace by assigning roles to users, groups, and/or service principals. [Learn more](#)

**Scope \***  Workspace  Workspace item

**Item type \***  Credentials

**Item \***  WorkspaceSystemIdentity

**Role \***  Synapse Administrator

**Select user \***  Search by name or email address

**Selected user(s), group(s), or service principal(s)**  
No users, groups, or apps selected.

# Manage – Source Control

## Overview

Associate Synapse workspace with a Git repository, Azure DevOps, or GitHub

**Configure a repository**

SynapseTestDemo

Specify the settings that you want to use when connecting to your repository.

Enter manually  Use repository link

**Git repository name \***  
synapsetestdemo-ws-01

**Collaboration branch \***  
dev

**Publish branch \***  
main

**Root folder \***  
/

**Import existing resource**  
 Import existing resources to repository

**Import resource into this branch**

**Source control**

**Git configuration**

**Apply** **Back** **Cancel**

**Microsoft Azure | Synapse Analytics**

wsazuresynapseanalytics

**Configure a repository**

main branch

Filter...

dev branch  
main branch  
workspace\_publish branch  
Create pull request [Alt+P]  
New branch [Alt+N]  
Switch to live mode

**Repository type** GitHub

**GitHub account** SynapseTestDemo

**Git repository name** synapsetestdemo-ws-01

**Collaboration branch** dev

**Publish branch** main

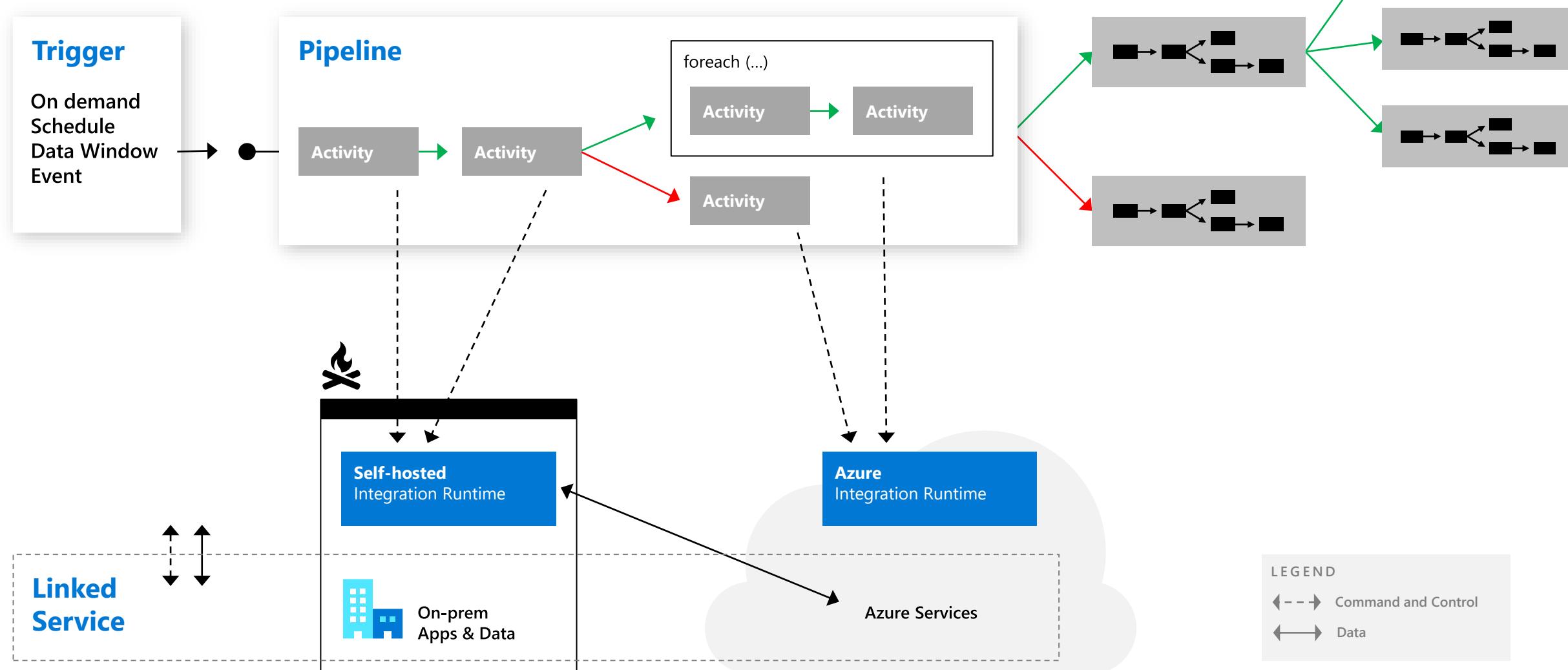
**Root folder** /



# Azure Synapse Analytics

## Integrate

# Orchestration @ Scale



# Data Movement

## Scalable

per job elasticity

Up to 4 GB/s

## Simple

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

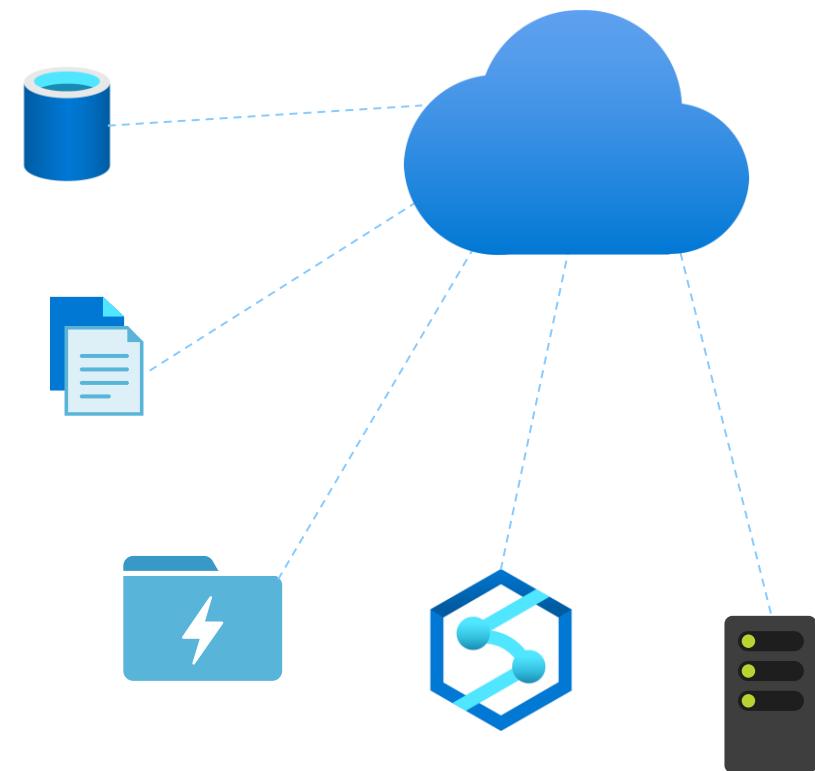
Serverless, no infrastructure to manage

## 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

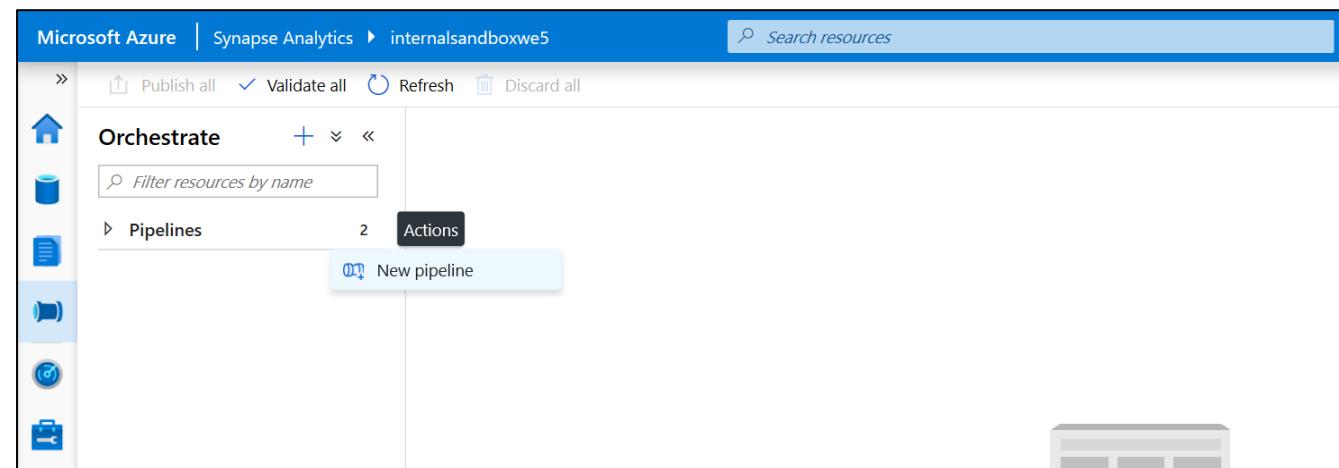


# 90+ Connectors out of the box

# Pipelines

## Overview

It provides ability to load data from storage account to desired linked service. Load data by manual execution of pipeline or by orchestration

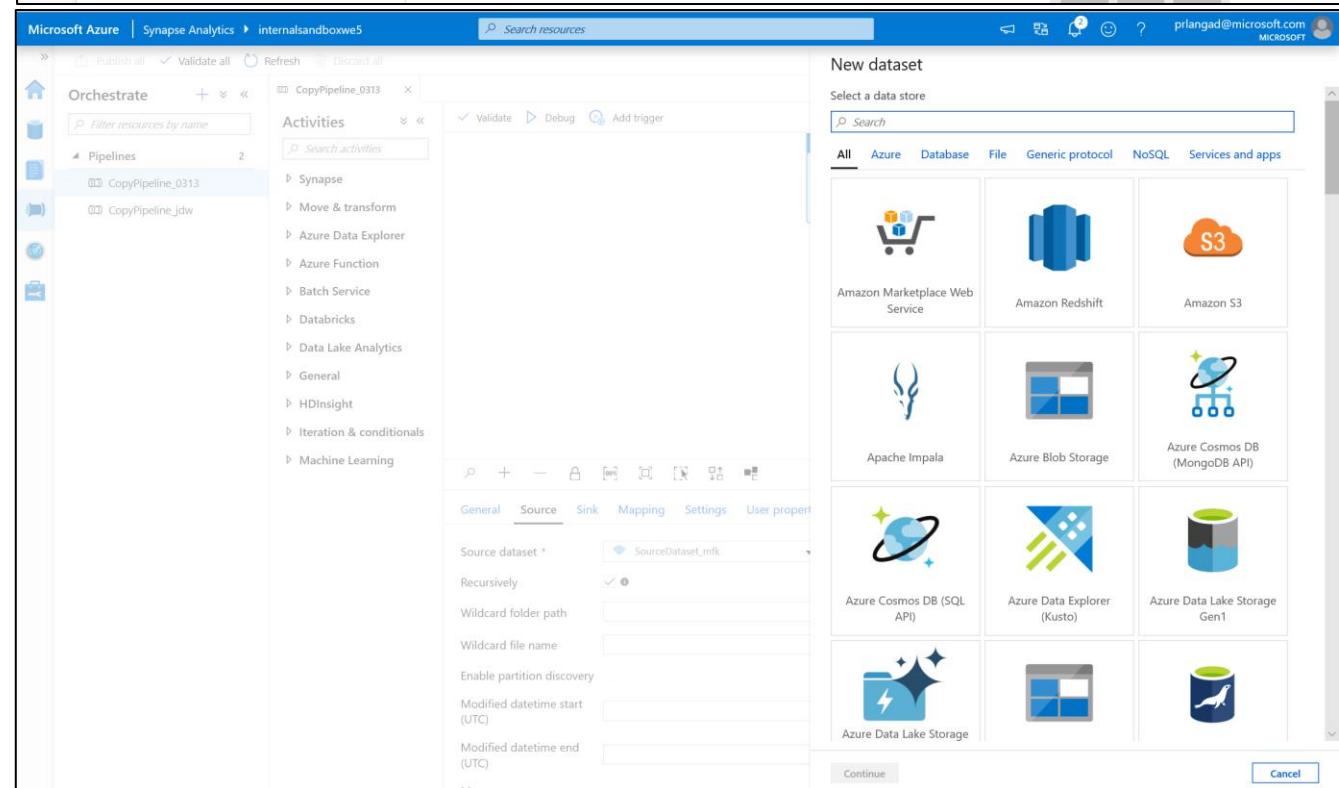


## Benefits

Supports common loading patterns

Fully parallel loading into data lake or SQL tables

Graphical development experience



# Prep & Transform Data

## Overview

It offers data cleansing, transformation, aggregation, conversion, etc.

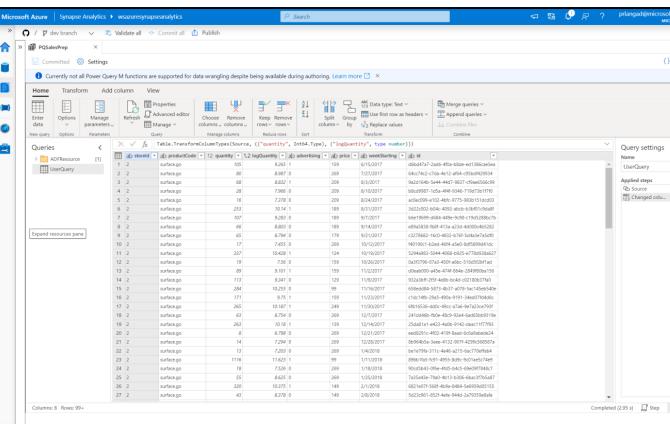
## Benefits

Cloud scale via Spark execution

Guided experience to easily build resilient data flows

Flexibility to transform data per user's comfort

Monitor and manage dataflows from a single pane of glass



...  
not

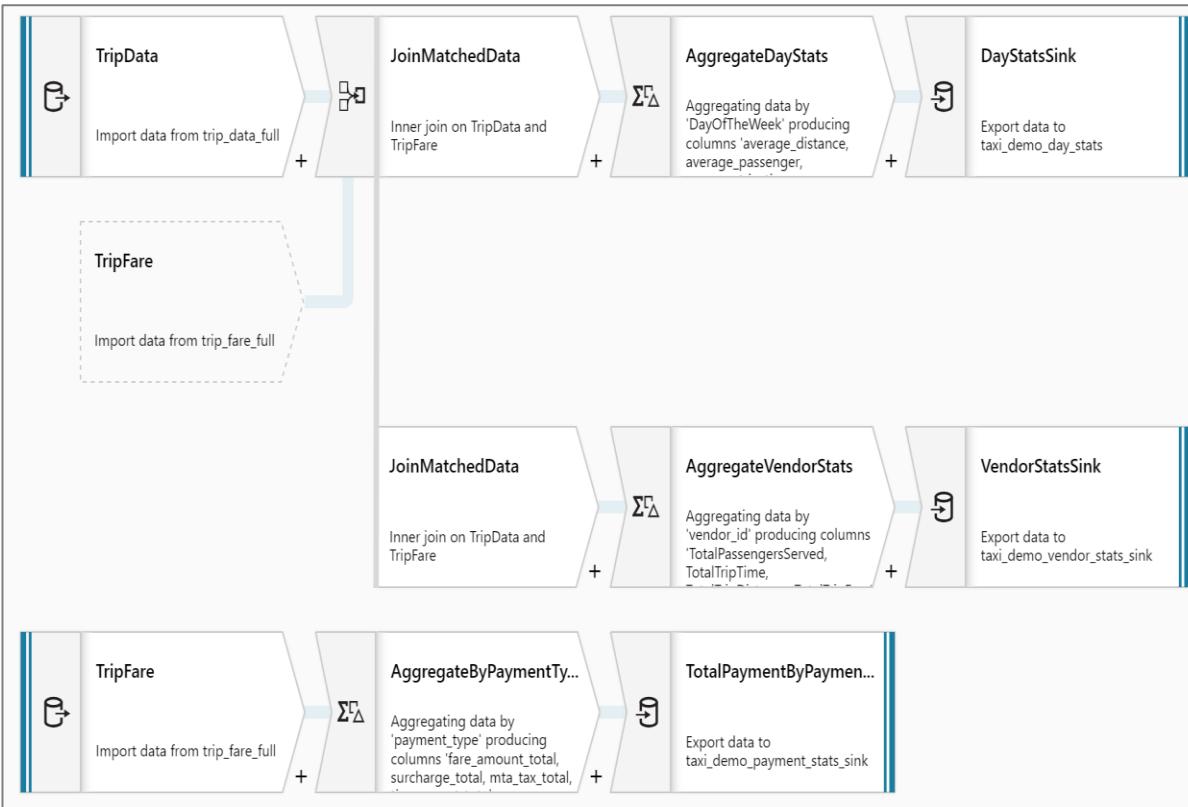
```

// MovieRecommendationE2EDemo.txt
1 // Hadoop Cluster Details:
2   adfhdi.azurehdinsight.net
3   Admin
4   Adfhdi@123456
5
6 Storage:
7   adfhdi/storage
8   /anyPw6G1j7tE1l0Mm1So/Vg2)gT4d+51Ar+vSn7bJg95476gjCHloksI9Uxscf40xZob1kWdWQ==
9
10 Cluster Remote Login Details:
11   Adfhdi
12   Adfhdi@123456
13
14 HiveQuery:
15   DROP TABLE IF EXISTS MovieRatings;
16   CREATE EXTERNAL TABLE MovieRatings
17   (
18     UserID int,
19     MovieID int,
20     Rating int,
21    TimeStamp string
22   ) ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' LINES TERMINATED BY '\n' STORED AS TEXTFILE LOCATION '${hiveconf:MovieRatings}';
23
24 DROP TABLE IF EXISTS MovieTitles;
25 CREATE EXTERNAL TABLE MovieTitles
26   (
27     MovieID int,
28     MovieTitle string
29   ) ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' LINES TERMINATED BY '\n' STORED AS TEXTFILE LOCATION '${hiveconf:MovieTitles}';
  
```

# Prep & Transform Data

## Mapping Dataflow

Code free data transformation @scale



## Power Query

Code free data preparation @scale

The screenshot shows the Microsoft Power Query Editor interface with the following details:

- Home Tab:** Shows the current file path: Microsoft Azure | Synapse Analytics > wsasuresynapseanalytics.
- Transform Tab:** Contains various data wrangling tools like Enter data, Options, Manage parameters, Refresh, Advanced editor, Properties, and a large ribbon of transformation functions.
- Queries List:** Displays a list of queries:
  - ab\_storeid (ADFResource) [1]: Contains 2 rows.
  - UserQuery: Contains 27 rows of data, each with columns: ab\_productCode, quantity, logQuantity, advertising, price, weekStarting, id, and date.
- Applied Steps Panel:** Shows the history of changes made to the query, including "Changed colu...".
- Query settings Panel:** Shows the name "UserQuery" and the applied step "Changed colu...".

ab_productCode	quantity	logQuantity	advertising	price	weekStarting	id	date
surface.go	105	9.365	1	159	6/15/2017	d6bd47a7-2ad5-4f0a-b0de-ed1306ca5ea	04c74c2-c7d4e12-a64-c9db0d429334
surface.go	80	8.967	0	269	7/27/2017	9a2164b5-e54d-44d7-9837-9a6e566c599	b8d99987-1d5a-4ff9-9346-719d73b17f70
surface.go	68	8.832	1	209	8/3/2017	9a2164b5-e54d-44d7-9837-9a6e566c599	9e0c0994-e102-4bf6-9775-983b151dc0f3
surface.go	28	7.966	0	209	8/10/2017	9a2164b5-e54d-44d7-9837-9a6e566c599	3d22c002-3f04c-4092-abcb-b3bf01c5da8f
surface.go	16	7.378	0	209	8/24/2017	9a2164b5-e54d-44d7-9837-9a6e566c599	b6e19699-d684-449e-9c9-c19d5288bc7b
surface.go	253	10.14	1	189	8/31/2017	9a2164b5-e54d-44d7-9837-9a6e566c599	899a5838-f68f-413a-a23d-4d0000bc5282
surface.go	107	9.283	0	189	9/7/2017	9a2164b5-e54d-44d7-9837-9a6e566c599	923a3b7-2f51-4eb9-bc4d-c02180379f0
surface.go	66	8.803	0	189	9/14/2017	9a2164b5-e54d-44d7-9837-9a6e566c599	658edd84-5873-4b37-a078-3ac145eb540e
surface.go	65	8.794	0	179	9/21/2017	9a2164b5-e54d-44d7-9837-9a6e566c599	c3278863-16cd-4832-4767-5d4a3e7a5d0
surface.go	17	7.455	0	269	10/12/2017	9a2164b5-e54d-44d7-9837-9a6e566c599	140f90c1-22ed-4f64-a5e0-8d58994a1dc
surface.go	337	10.428	1	124	10/19/2017	9a2164b5-e54d-44d7-9837-9a6e566c599	525a983-3044-4068-c923-e778938a627
surface.go	19	7.56	0	159	10/26/2017	9a2164b5-e54d-44d7-9837-9a6e566c599	0a3f3796-07a3-450f-abcb-516d52b1fad
surface.go	89	9.101	1	159	11/2/2017	9a2164b5-e54d-44d7-9837-9a6e566c599	9e0ab000-45e-474f-854e-2849b80ba156
surface.go	173	9.347	0	129	11/9/2017	9a2164b5-e54d-44d7-9837-9a6e566c599	899a5838-f68f-413a-a23d-4d0000bc5282
surface.go	284	10.255	0	99	11/16/2017	9a2164b5-e54d-44d7-9837-9a6e566c599	658edd84-5873-4b37-a078-3ac145eb540e
surface.go	171	9.75	1	159	11/23/2017	9a2164b5-e54d-44d7-9837-9a6e566c599	c1d149b-2959-490a-9191-34ed07f046fc
surface.go	265	10.187	1	249	11/30/2017	9a2164b5-e54d-44d7-9837-9a6e566c599	6fb16536-0d0c-49cc-97a6-9e7a3ce793f
surface.go	63	8.754	0	269	12/7/2017	9a2164b5-e54d-44d7-9837-9a6e566c599	241dd46b-fb0e-4b9c-92e4-6ad63bb9319e
surface.go	263	10.18	1	139	12/14/2017	9a2164b5-e54d-44d7-9837-9a6e566c599	25d8181-423-40b-9142-deaf11f77f93
surface.go	8	6.798	0	269	12/21/2017	9a2164b5-e54d-44d7-9837-9a6e566c599	e0d291c4-0202-419f-8aa-6ca9abed2e4
surface.go	14	7.294	0	269	12/28/2017	9a2164b5-e54d-44d7-9837-9a6e566c599	899a5838-f68f-413a-a23d-4d0000bc5282
surface.go	13	7.203	0	269	1/4/2018	9a2164b5-e54d-44d7-9837-9a6e566c599	be1e797a-311c-4e46-4215-6a770efab4
surface.go	1116	11.623	1	99	1/11/2018	9a2164b5-e54d-44d7-9837-9a6e566c599	899a5838-f68f-413a-a23d-4d0000bc5282
surface.go	18	7.526	0	269	1/18/2018	9a2164b5-e54d-44d7-9837-9a6e566c599	900d5b43-0f6e-4f05-b4c5-69e097f84c7
surface.go	55	8.625	0	269	1/25/2018	9a2164b5-e54d-44d7-9837-9a6e566c599	735a493e-799d-4b13-e306-9eac37f5a87
surface.go	320	10.375	1	149	2/1/2018	9a2164b5-e54d-44d7-9837-9a6e566c599	6821e07f-568f-409a-9484-5e6959d05155
surface.go	43	8.378	0	149	2/8/2018	9a2164b5-e54d-44d7-9837-9a6e566c599	5d23d961-852f-4fe-944d-2a79359eafe

# 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 the Azure Synapse Analytics Data Integration interface. On the left, there's a navigation sidebar with options like 'Analytics pools', 'SQL pools', 'Apache Spark pools', 'External connections', 'Linked services', 'Orchestration', 'Triggers' (which is selected and highlighted in blue), 'Integration runtimes', 'Security', 'Access control', and 'Managed Virtual Networks'. The main area is titled 'Triggers' with the sub-instruction: 'To execute a pipeline set the trigger. Triggers represent a unit of processing that determines when a pipeline execution needs to be kicked off.' Below this, there's a search bar with 'Search to filter items...' and a table showing one item: 'HolidayUpdateTrigger' (Type: Schedule, Status: Started). At the top, there's a 'New trigger' dialog box with fields for Name (set to 'Trigger 1'), Description, Type (selected 'Schedule'), Start Date (set to '10/30/2019 11:20 PM'), Recurrence (set to 'Every 1 Minute(s)'), End (set to 'No End'), Annotations (with a '+ New' button), and Activated (set to 'Yes').



# Azure Synapse Analytics

## Synapse SQL

# Key features

## Rich surface area

- T-SQL language for data analytics
- Supporting large number of languages and tools
- Enterprise-grade security

## dedicated SQL pool

- Modern Data Warehouse
- Indexing and caching
- Import and query external data
- Workload management

## serverless SQL pool

- Querying external data
- Model raw files as virtual tables and views
- Easy data transformation

# Synapse SQL - clients and tools

## Tools

Web IDE - Synapse Studio

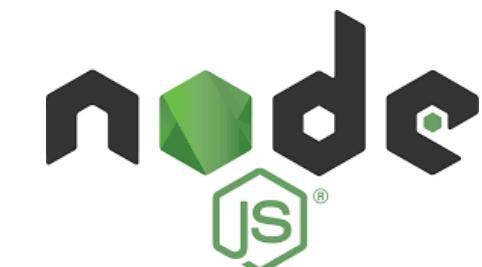
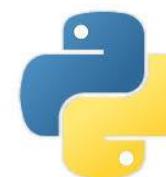
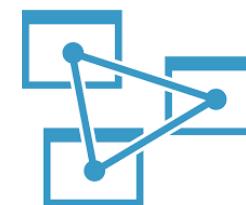


Client tools - Azure Data Studio, SSMS,



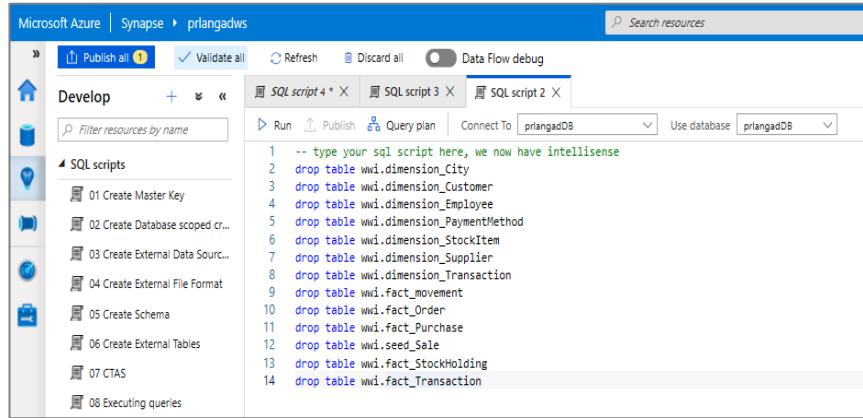
Any tool/library that uses standard SQL can access serverless SQL pool

- PowerBI
- Azure Analytic Services
- Client languages and drivers that works with Azure SQL can be used to access serverless SQL pool

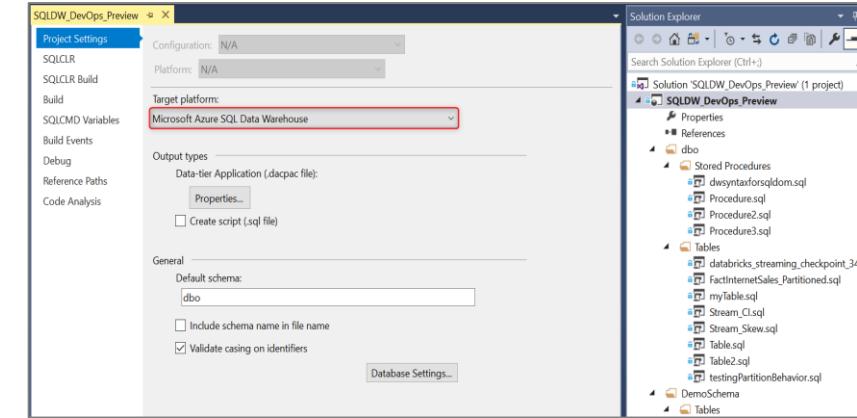


# Developer Tools

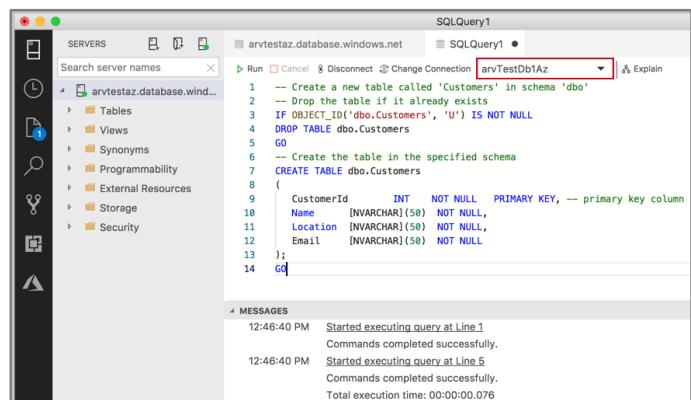
## Azure Synapse Analytics



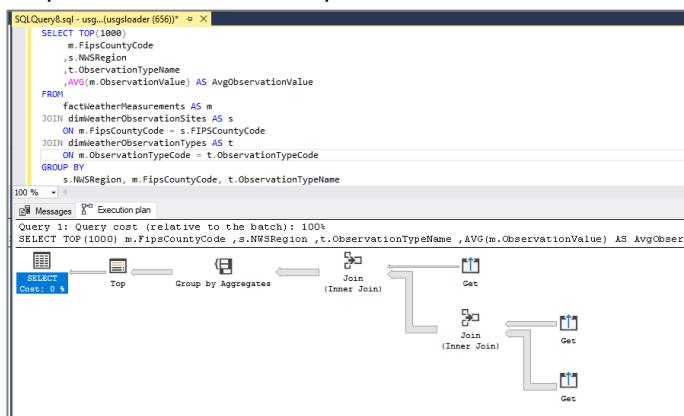
## Visual Studio - SSDT database projects



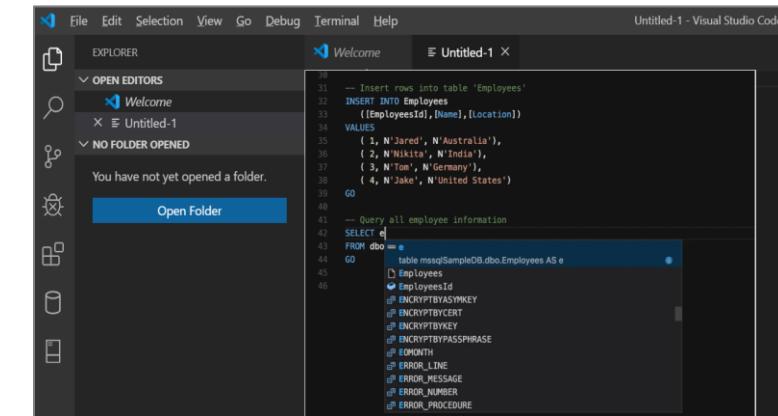
## Azure Data Studio (queries, extensions etc.)



## SQL Server Management Studio (queries, execution plans etc.)



## Visual Studio Code

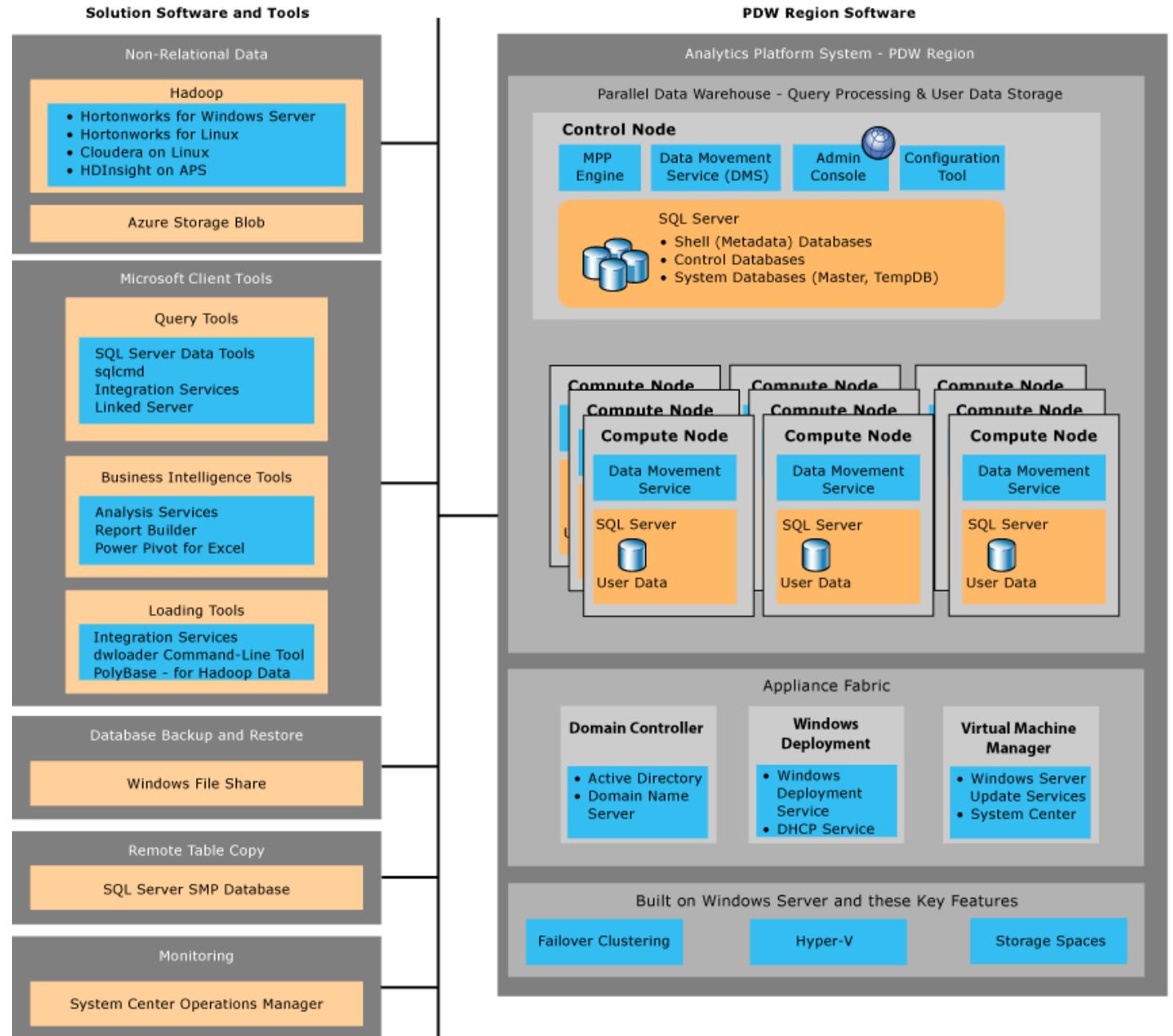




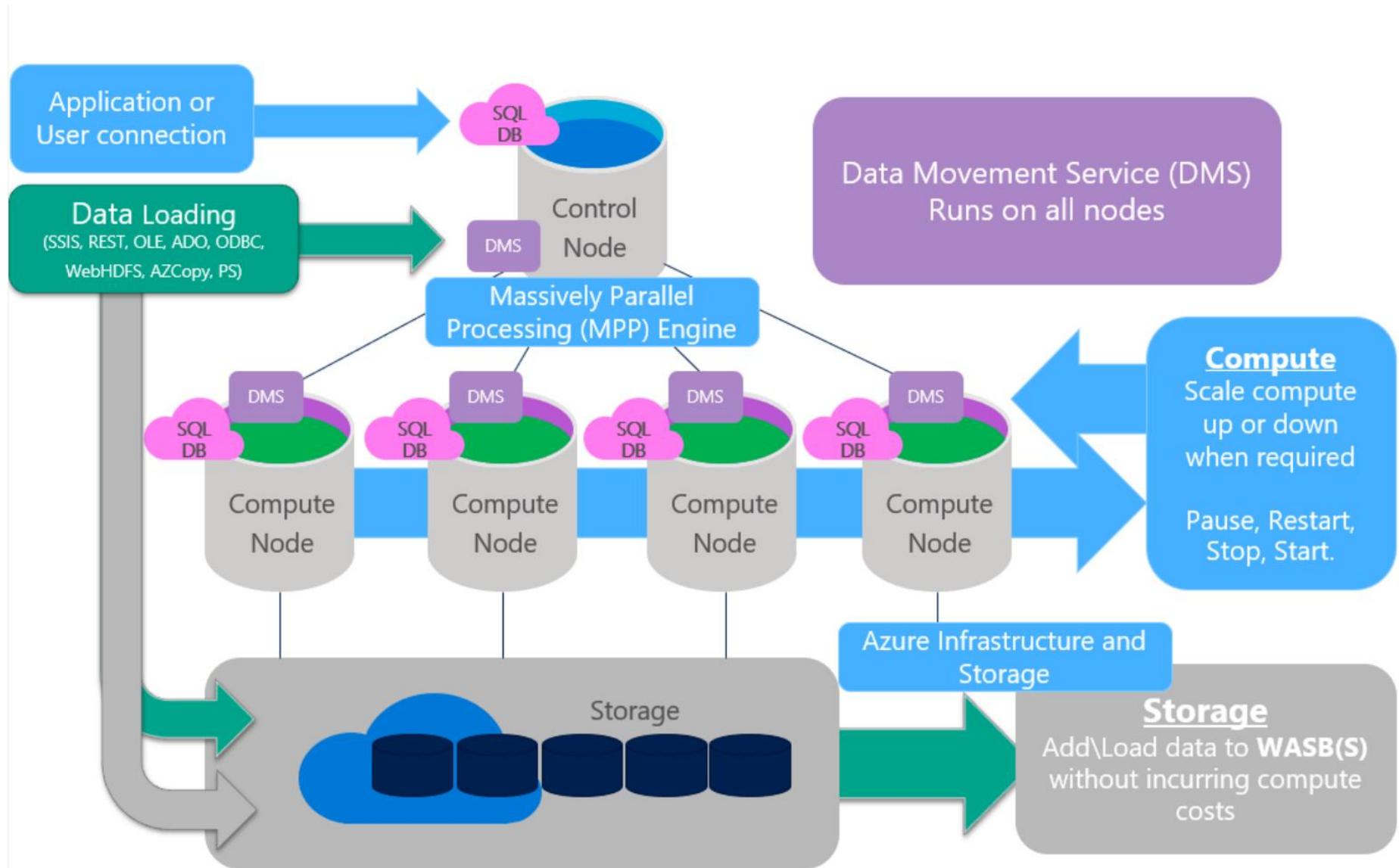
# Azure Synapse Analytics

## Synapse dedicated SQL pool

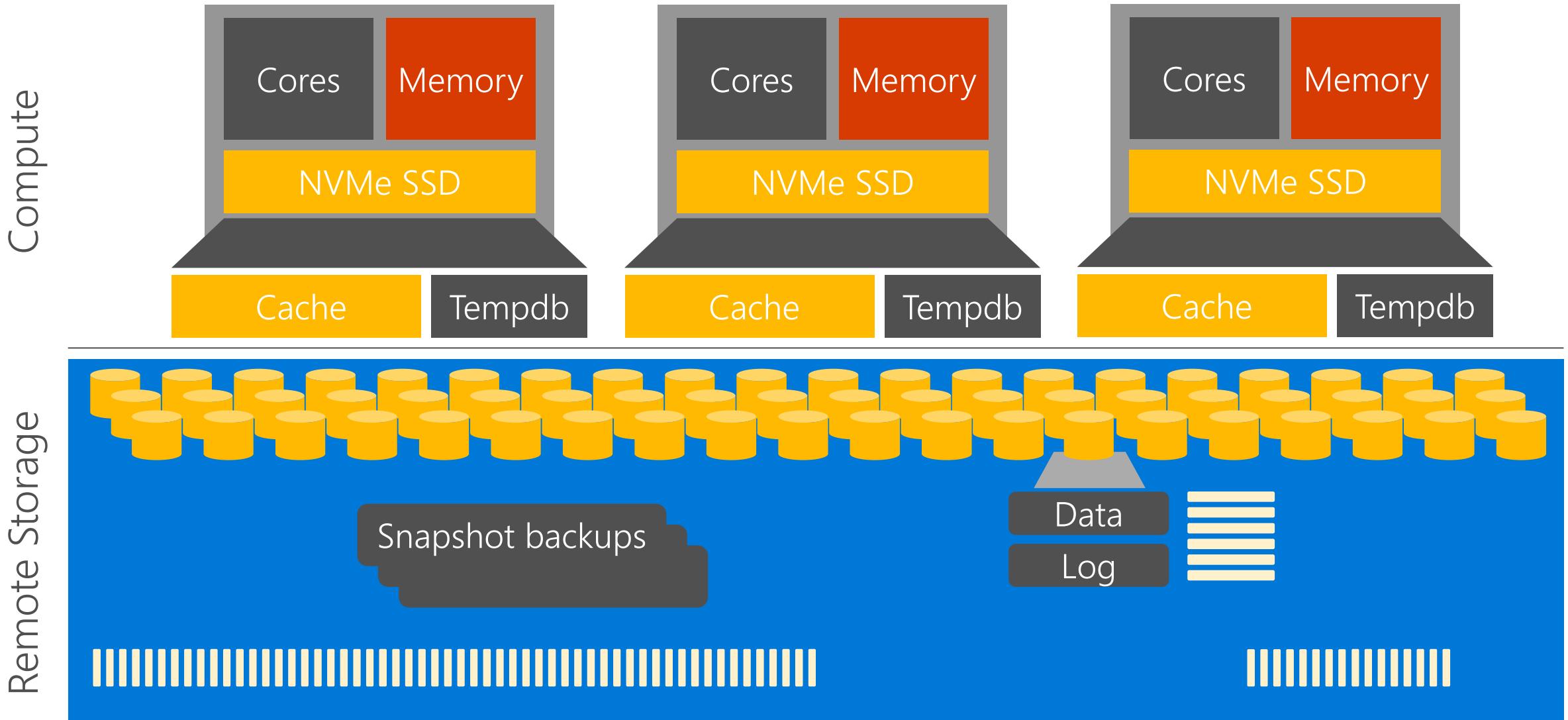
# PDW (Parallel Data Warehouse) Architecture



# Synapse SQL Pool Architecture

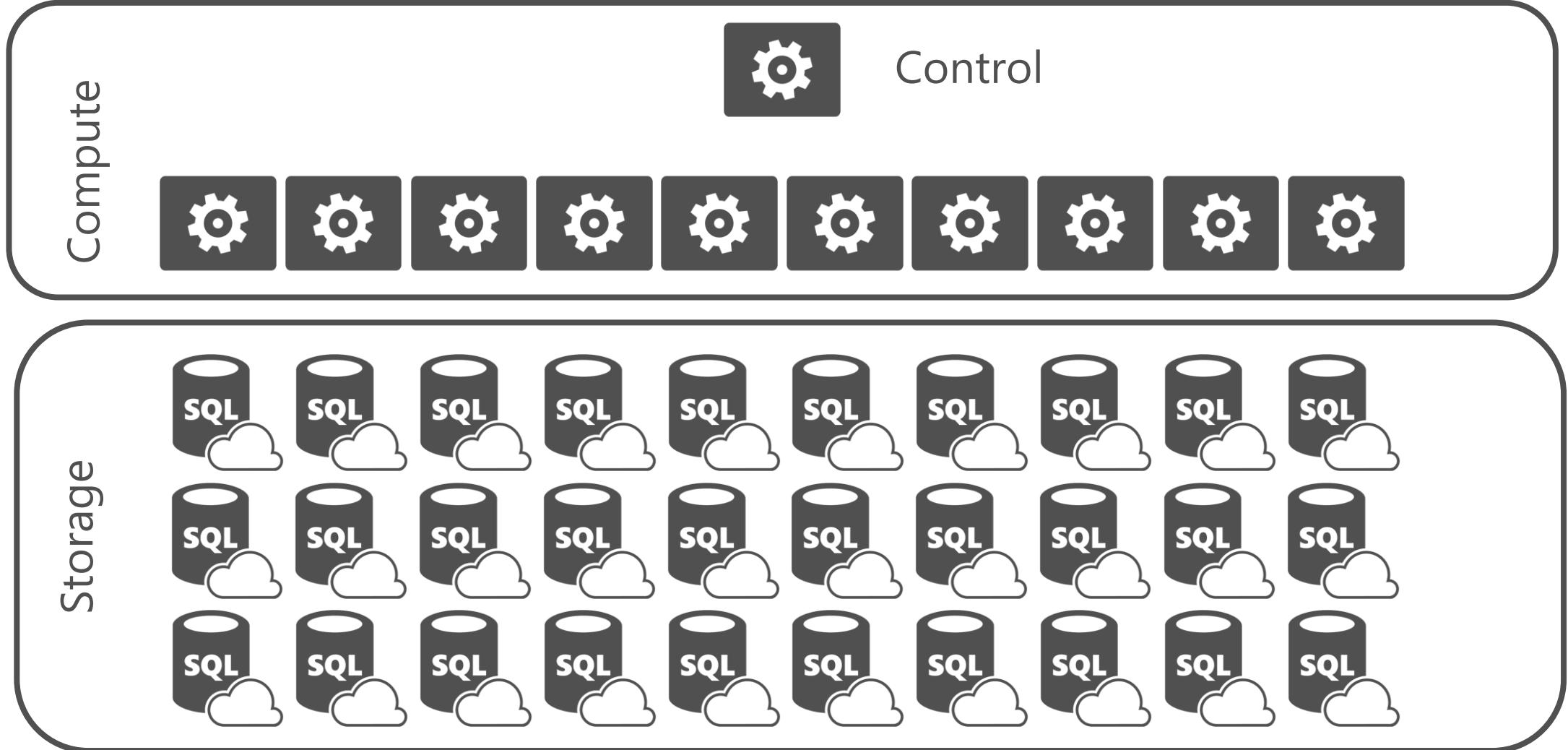


# TIERED STORAGE MODEL AUTOMATES DATA TEMPERATURE

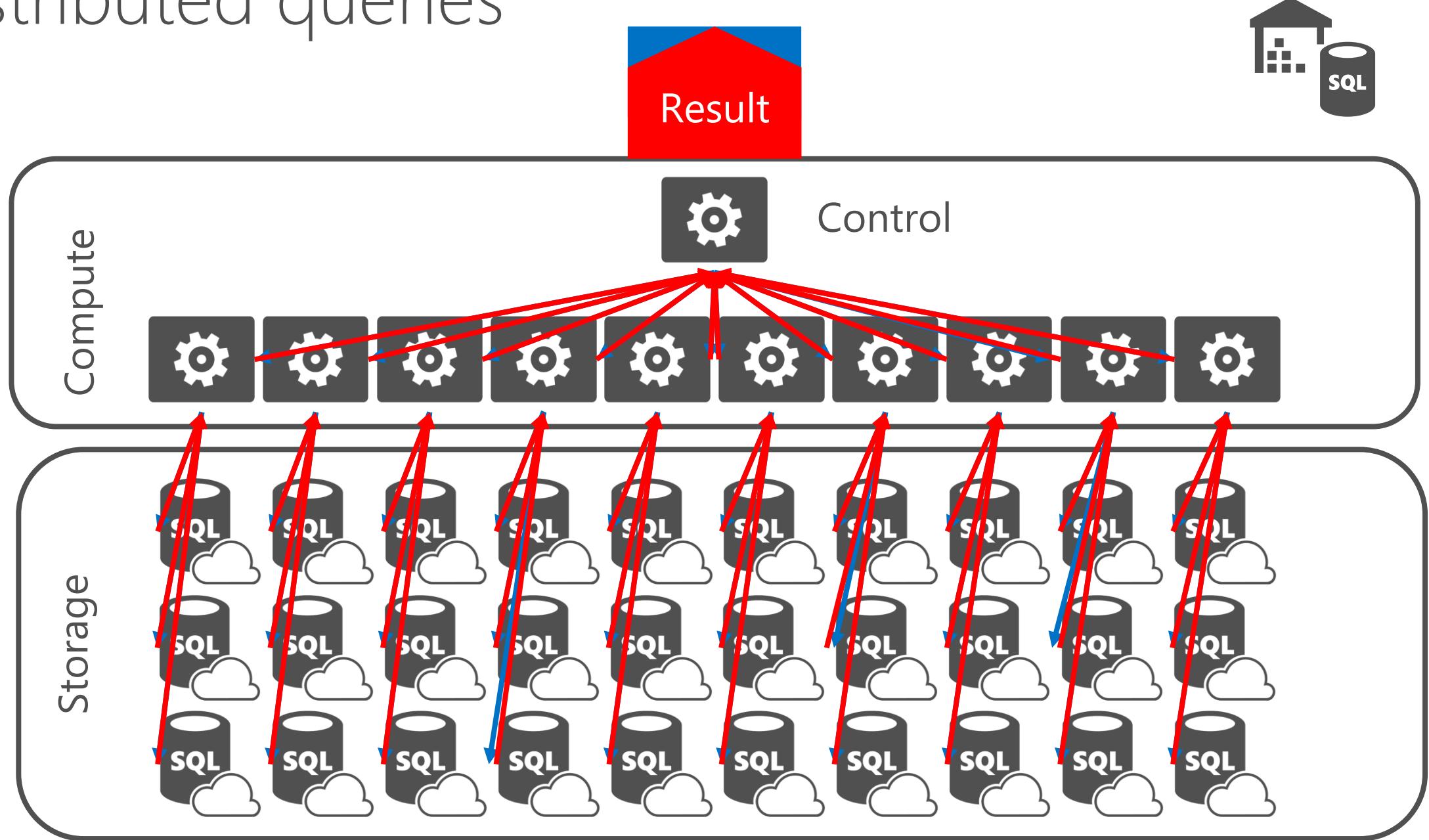


# Synapse Analytics Architectural overview

# Logical overview



# Distributed queries



# Simple example

```
SELECT COUNT_BIG(*)  
FROM dbo.[FactInternetSales]  
;
```



```
SELECT SUM(*)  
FROM dbo.[FactInternetSales]  
;
```



Control

Compute

```
SELECT COUNT_BIG(*)  
FROM dbo.[FactInternetSales]  
;
```



```
SELECT COUNT_BIG(*)  
FROM dbo.[FactInternetSales]  
;
```



```
SELECT COUNT_BIG(*)  
FROM dbo.[FactInternetSales]  
;
```



```
SELECT COUNT_BIG(*)  
FROM dbo.[FactInternetSales]  
;
```

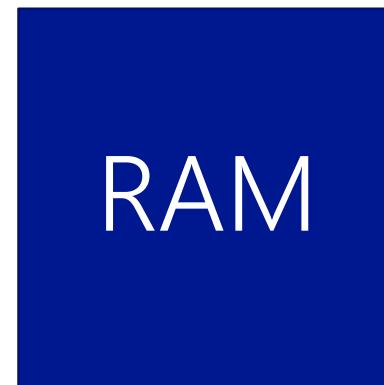
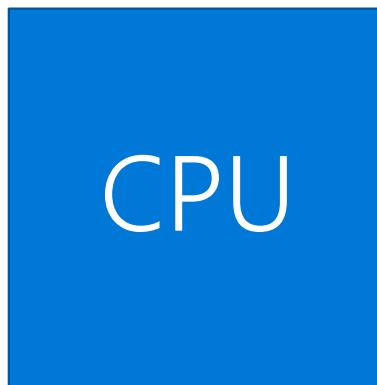


# Scale operations

# Data Warehouse Units

Normalized amount of compute

Converts to billing units i.e. what you pay



DWUc
100
200
300
400
500
600
1000
1200
1500
2000
3000
6000
...
30000

# Management Interfaces

	<b>Portal</b>	<b>PowerShell</b>	<b>T-SQL</b>
Create			
Pause and resume			
Scale			

# Scaling via T-SQL and PowerShell

T-SQL

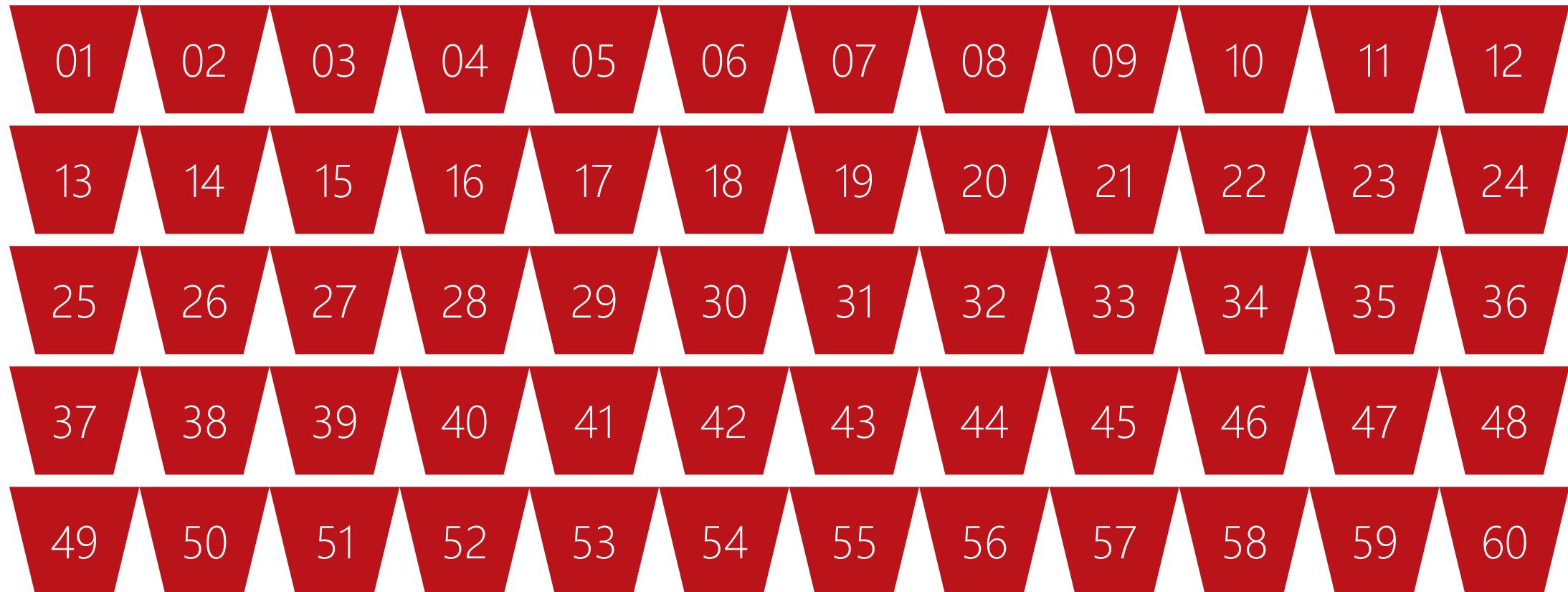
```
ALTER DATABASE ContosoRetailDW  
MODIFY (service_objective = 'DW100');
```

PowerShell

```
Set-AzureRmSqlDatabase  
-ResourceGroupName "RG_name"  
-ServerName "SRV_name"  
-DatabaseName "DB_name"  
-RequestedServiceObjectiveName "Dw1000"
```

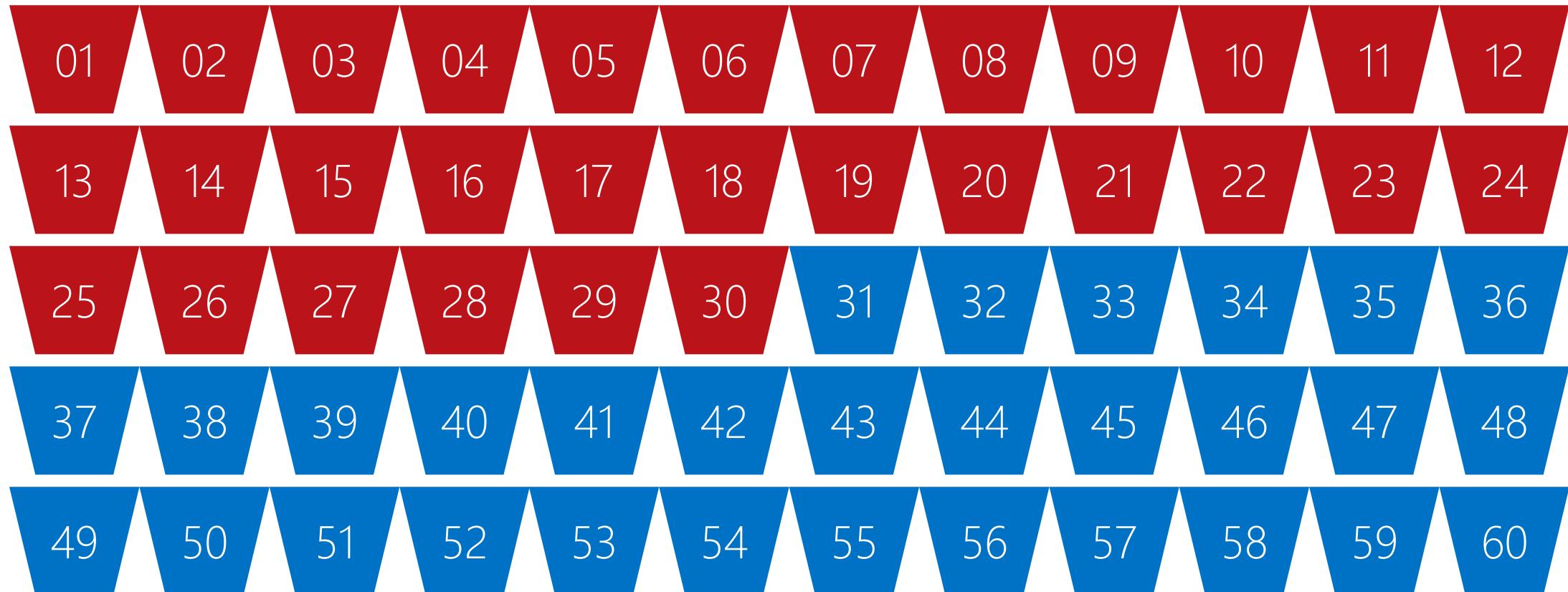
# Mapping Compute in S QLDW

DW100



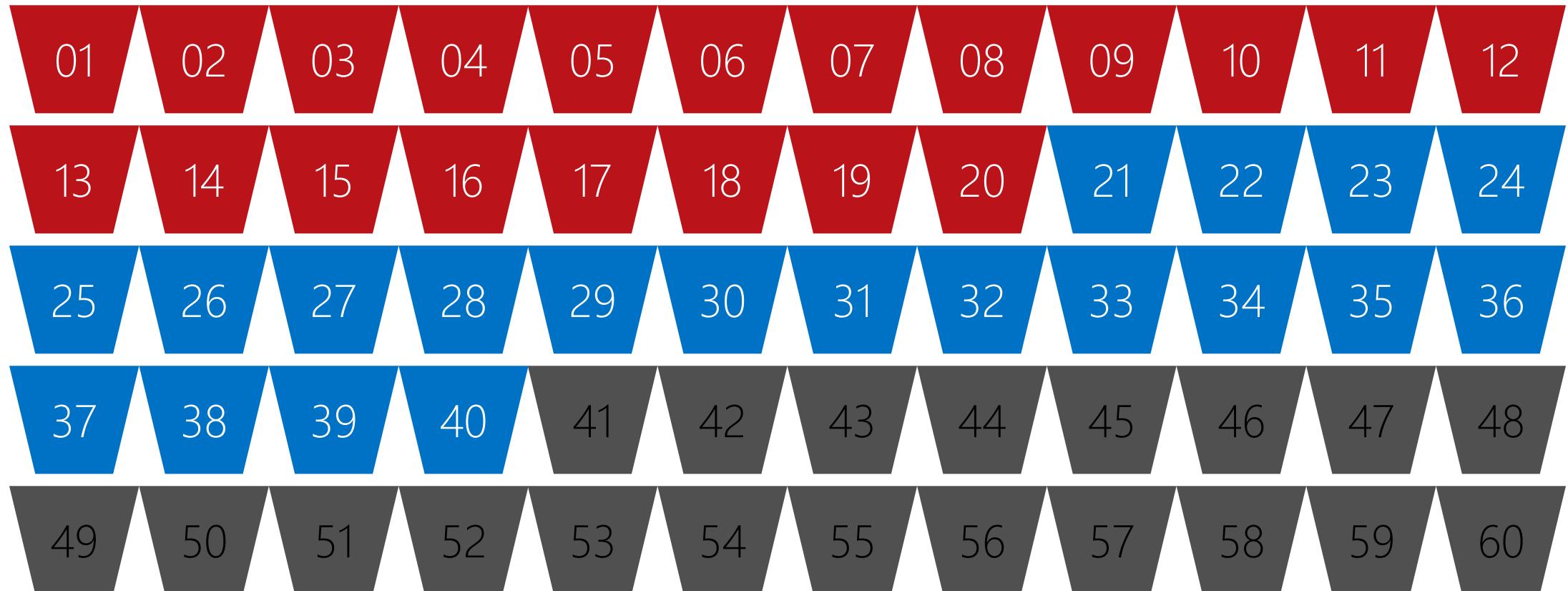
# Mapping Compute in SQLDW

DW200



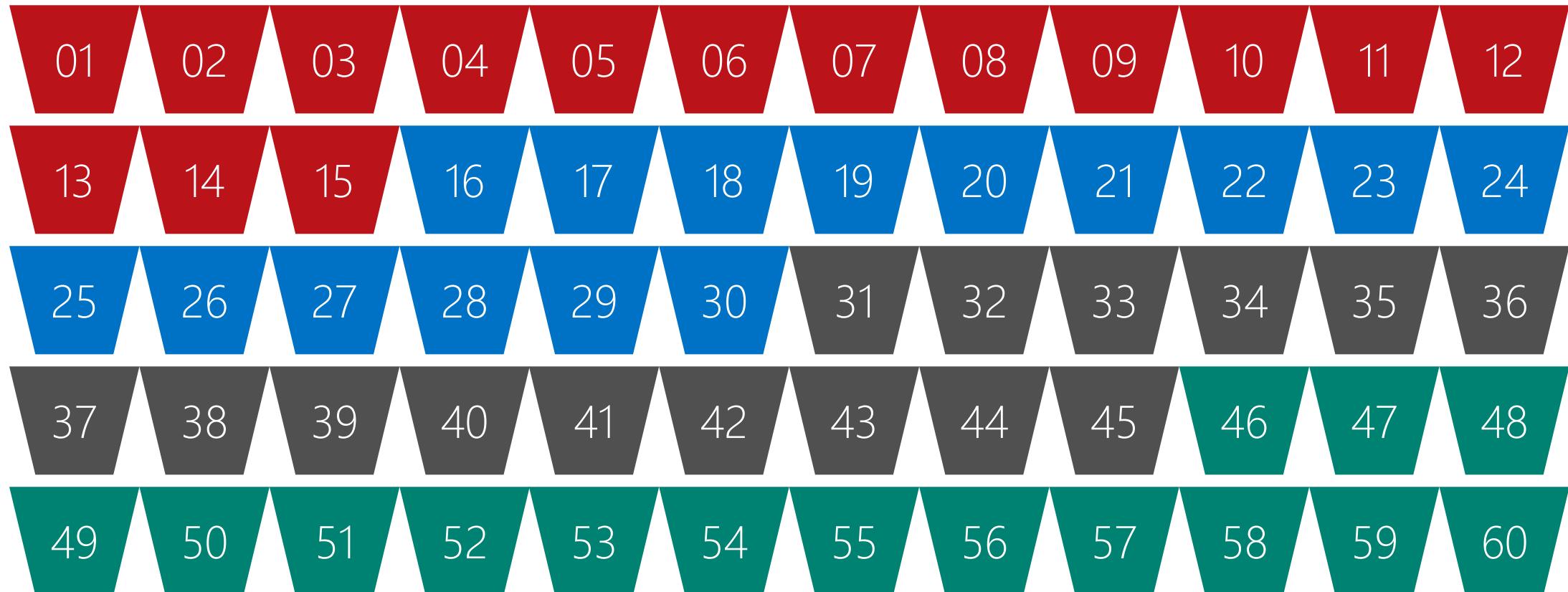
# Mapping Compute in SQLDW

DW300



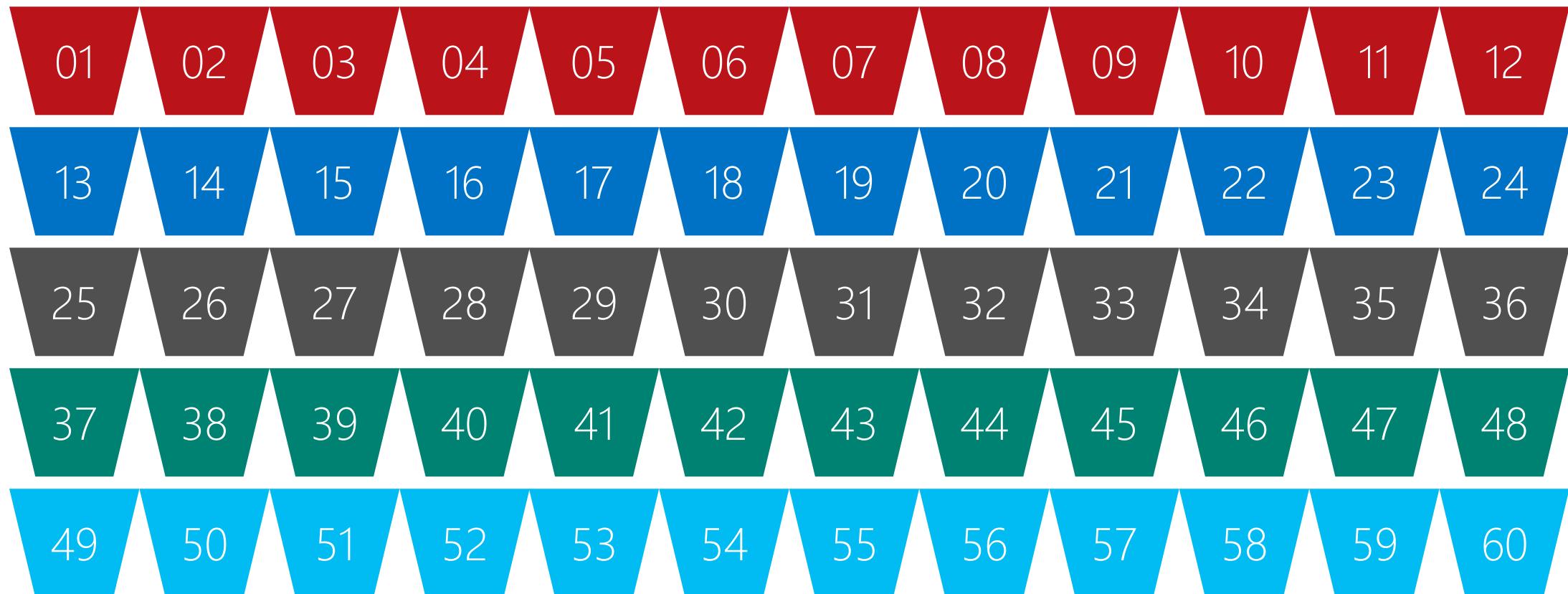
# Mapping Compute in SQLDW

DW400



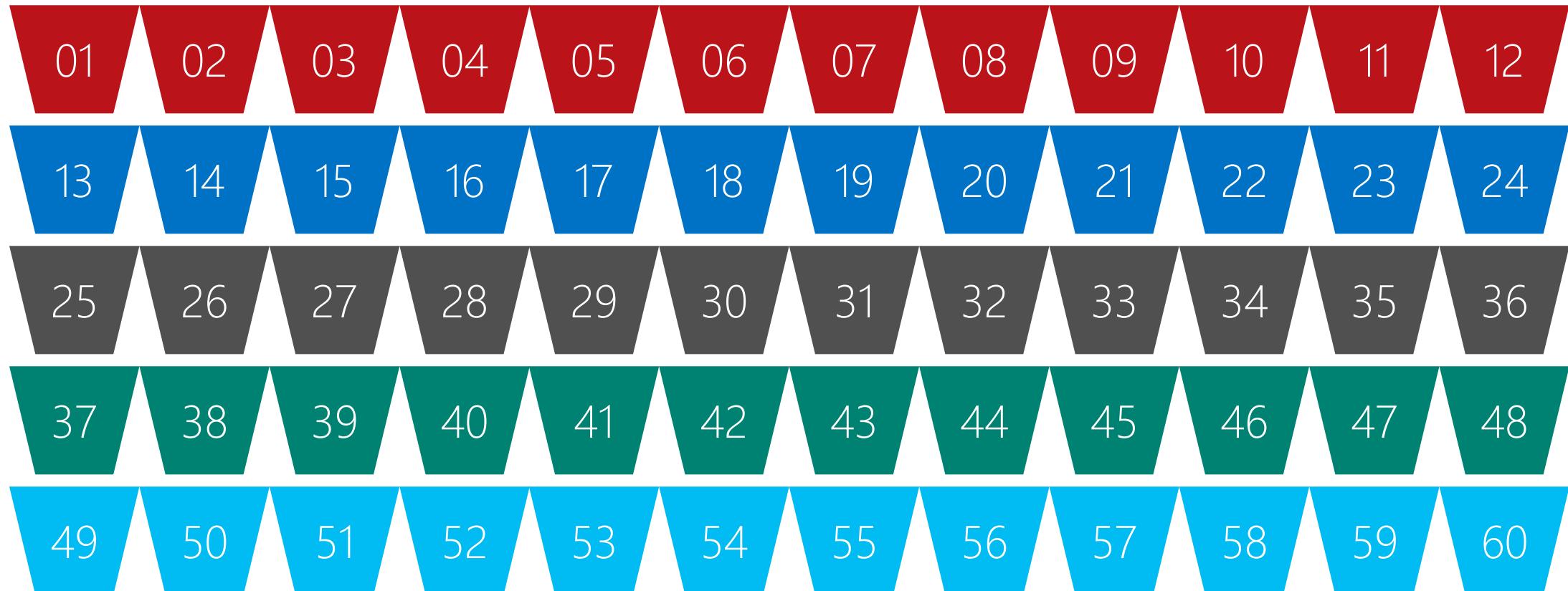
# Mapping Compute in SQLDW

DW500



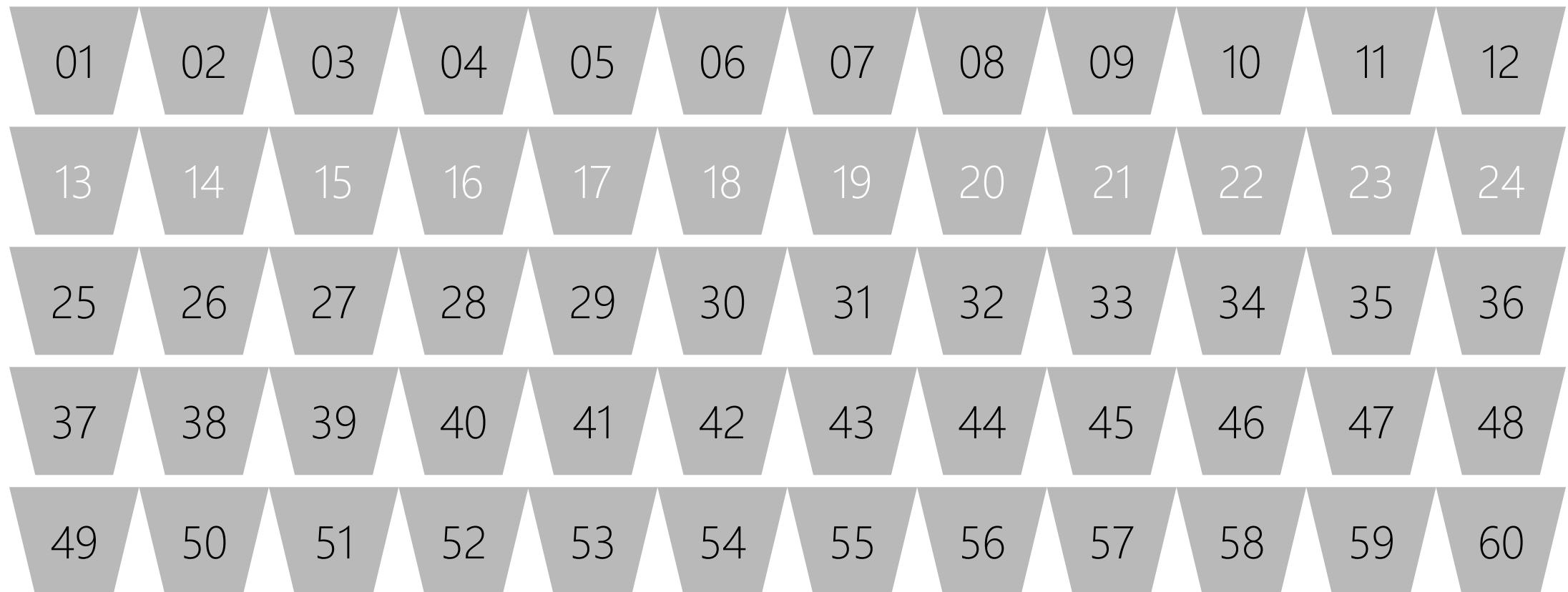
# Pausing compute in SQLDW

DW500



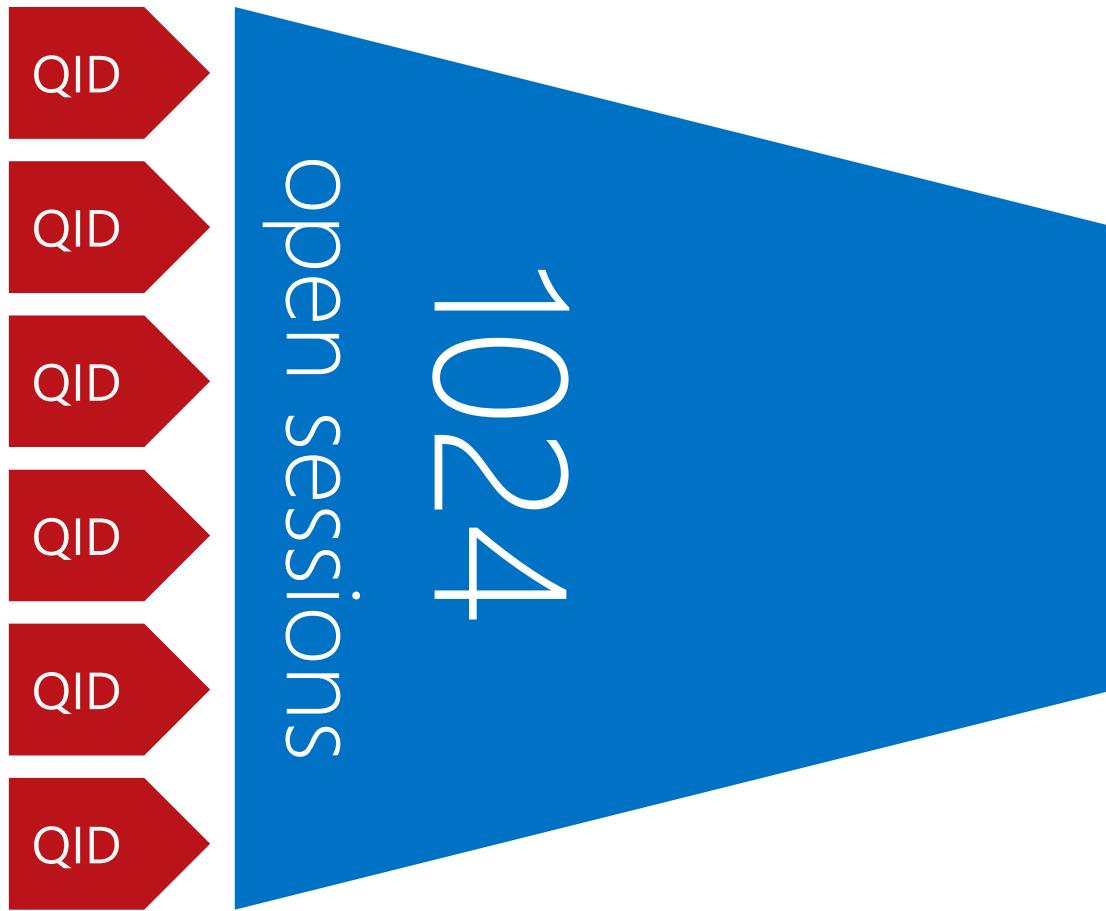
# Resuming compute in SQLDW

DW500



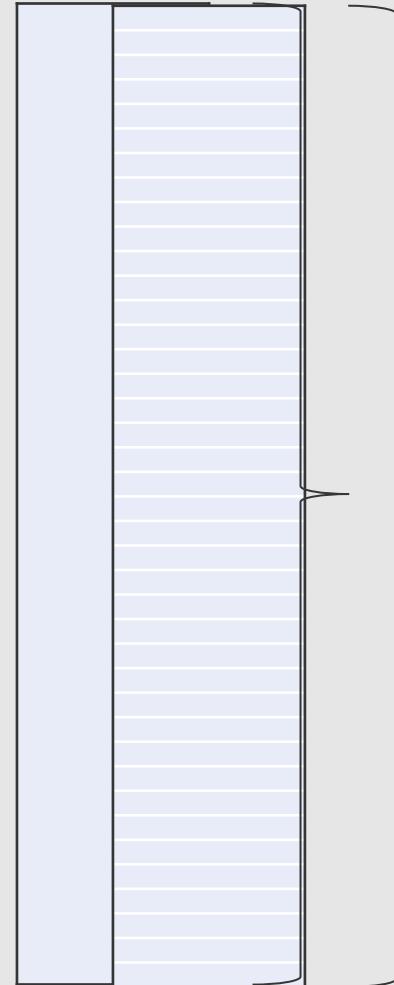
# Concurrency

# Concurrency: queries



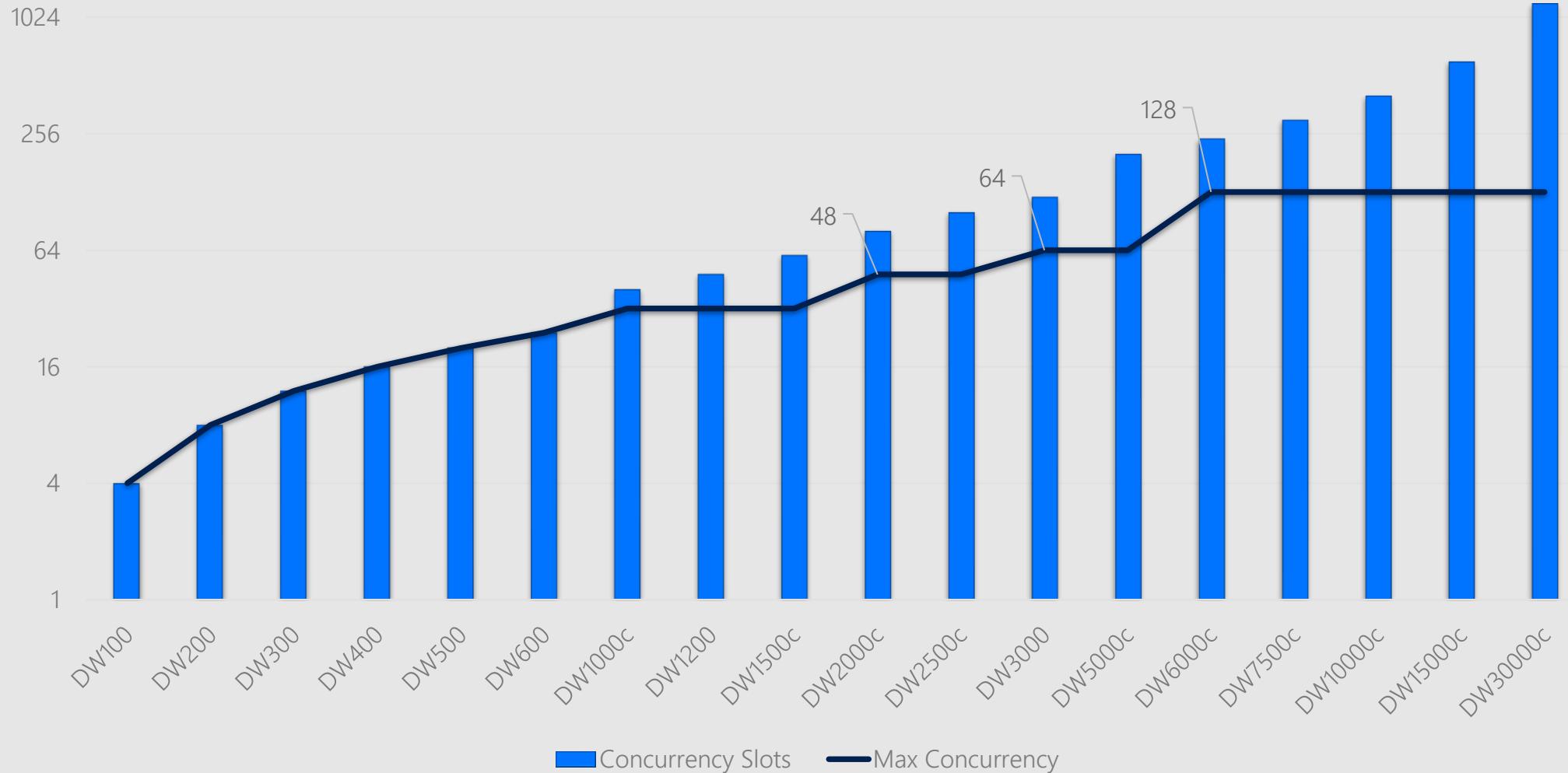
128  
active queries

# Concurrency slots



Concurrency Slots  
Total system available memory  
20 per DW500c: 250MB/slot  
4 per DW100: 100MB/slot

# Concurrent Query and Slots



# Memory & Concurrency Limits

<https://docs.microsoft.com/en-us/azure/synapse-analytics/sql-data-warehouse/memory-concurrency-limits>

## Concurrency maximums for workload groups

With the introduction of [workload groups](#), the concept of concurrency slots no longer applies. Resources per request are allocated on a percentage basis and specified in the workload group definition. However, even with the removal of concurrency slots, there are minimum amounts of resources needed per queries based on the service level. The below table defined the minimum amount of resources needed per query across service levels and the associated concurrency that can be achieved.

Service Level	Maximum concurrent queries	Min % supported for REQUEST_MIN_RESOURCE_GRANT_PERCENT
DW100c	4	25%
DW200c	8	12.5%
DW300c	12	8%
DW400c	16	6.25%
DW500c	20	5%
DW1000c	32	3%
DW1500c	32	3%
DW2000c	48	2%
DW2500c	48	2%
DW3000c	64	1.5%
DW5000c	64	1.5%
DW6000c	128	0.75%
DW7500c	128	0.75%
DW10000c	128	0.75%
DW15000c	128	0.75%
DW30000c	128	0.75%

# Resource Class

<https://docs.microsoft.com/en-us/azure/synapse-analytics/data-warehouse/resource-classes-for-workload-management>

## Static resource classes

Static resource classes allocate the same amount of memory regardless of the current performance level, which is measured in [data warehouse units](#). Since queries get the same memory allocation regardless of the performance level, [scaling out the data warehouse](#) allows more queries to run within a resource class. Static resource classes are ideal if the data volume is known and constant.

The static resource classes are implemented with these pre-defined database roles:

- staticrc10
- staticrc20
- staticrc30
- staticrc40
- staticrc50
- staticrc60
- staticrc70
- staticrc80

## Dynamic resource classes

Dynamic Resource Classes allocate a variable amount of memory depending on the current service level. While static resource classes are beneficial for higher concurrency and static data volumes, dynamic resource classes are better suited for a growing or variable amount of data. When you scale up to a larger service level, your queries automatically get more memory.

The dynamic resource classes are implemented with these pre-defined database roles:

- smallrc
- mediumrc
- largerc
- xlargerc

The memory allocation for each resource class is as follows.

Service Level	smallrc	mediumrc	largerc	xlargerc
DW100c	25%	25%	25%	70%
DW200c	12.5%	12.5%	22%	70%
DW300c	8%	10%	22%	70%
DW400c	6.25%	10%	22%	70%
DW500c	5%	10%	22%	70%
DW1000c to DW30000c	3%	10%	22%	70%

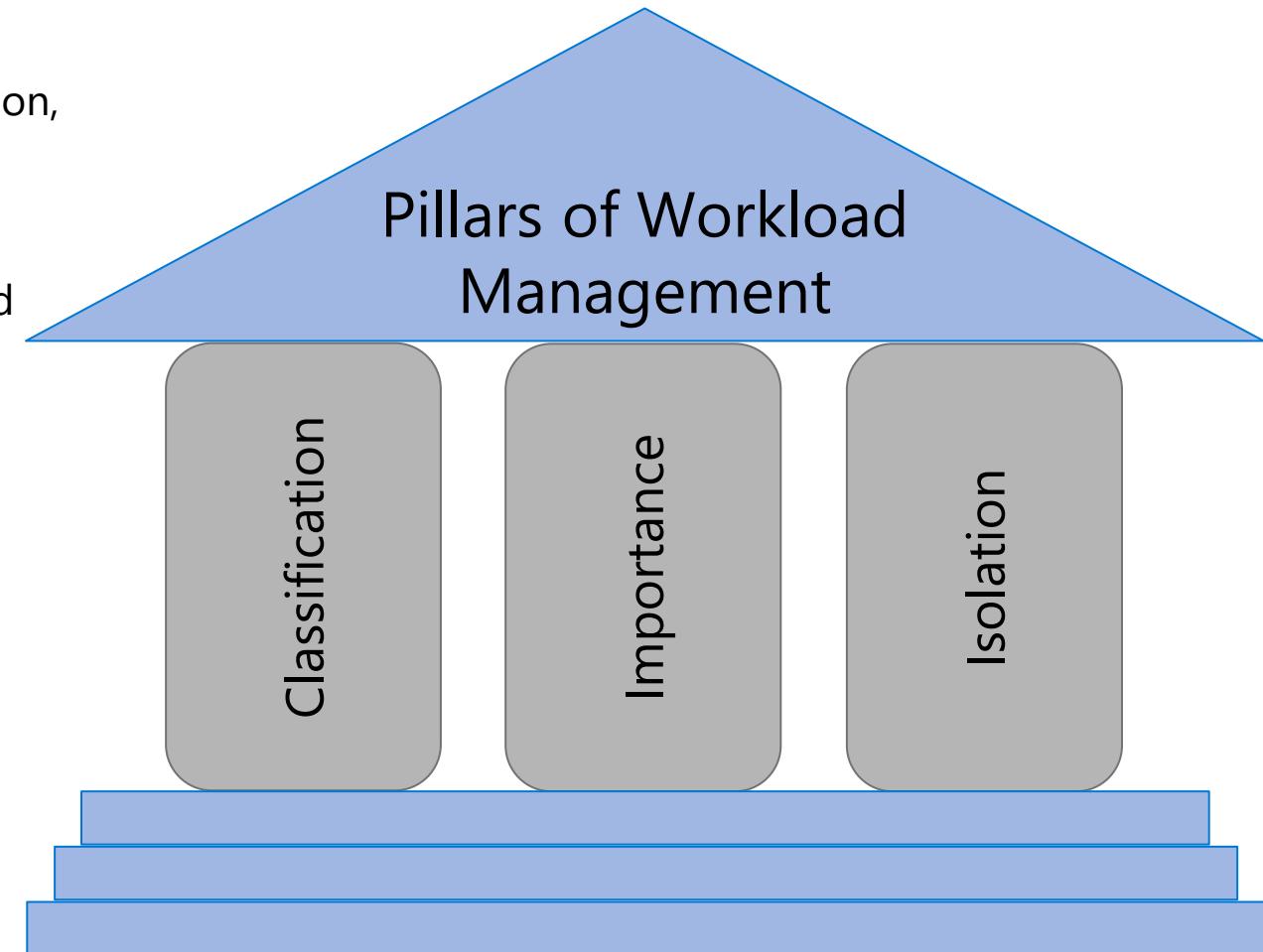
# Workload Management

## Overview

It manages resources, ensures highly efficient resource utilization, and maximizes return on investment (ROI).

The three pillars of workload management are

1. Workload Classification – To assign a request to a workload group and setting importance levels.
2. Workload Importance – To influence the order in which a request gets access to resources.
3. Workload Isolation – To reserve resources for a workload group.



# Workload classification

## Overview

Map queries to allocations of resources via pre-determined rules.

Use with workload importance to effectively share resources across different workload types.

If a query request is not matched to a classifier, it is assigned to the default workload group.

## Benefits

Map queries to both Resource Management and Workload Isolation concepts.

## Monitoring DMVs

[sys.workload\\_management\\_workload\\_classifiers](#)

[sys.workload\\_management\\_workload\\_classifier\\_details](#)

Query DMVs to view details about all active workload classifiers.

```
CREATE WORKLOAD CLASSIFIER classifier_name
WITH
(
    WORKLOAD_GROUP = 'name'
    , MEMBERNAME = 'security_account'
    [ [,] IMPORTANCE = {LOW|BELOW_NORMAL|NORMAL|ABOVE_NORMAL|HIGH} ]
    [ [,] WLM_LABEL = 'label' ]
    [ [,] WLM_CONTEXT = 'name' ]
    [ [,] START_TIME = 'start_time' ]
    [ [,] END_TIME = 'end_time' ]
)[ ; ]
```

*WORKLOAD\_GROUP: maps to an existing resource class*

*IMPORTANCE: specifies relative importance of request*

*MEMBERNAME: database user, role, AAD login or AAD group*

# Workload importance

## Overview

Queries past the concurrency limit enter a FiFo queue

By default, queries are released from the queue on a first-in, first-out basis as resources become available

Workload importance allows higher priority queries to receive resources immediately regardless of queue

## Example Video

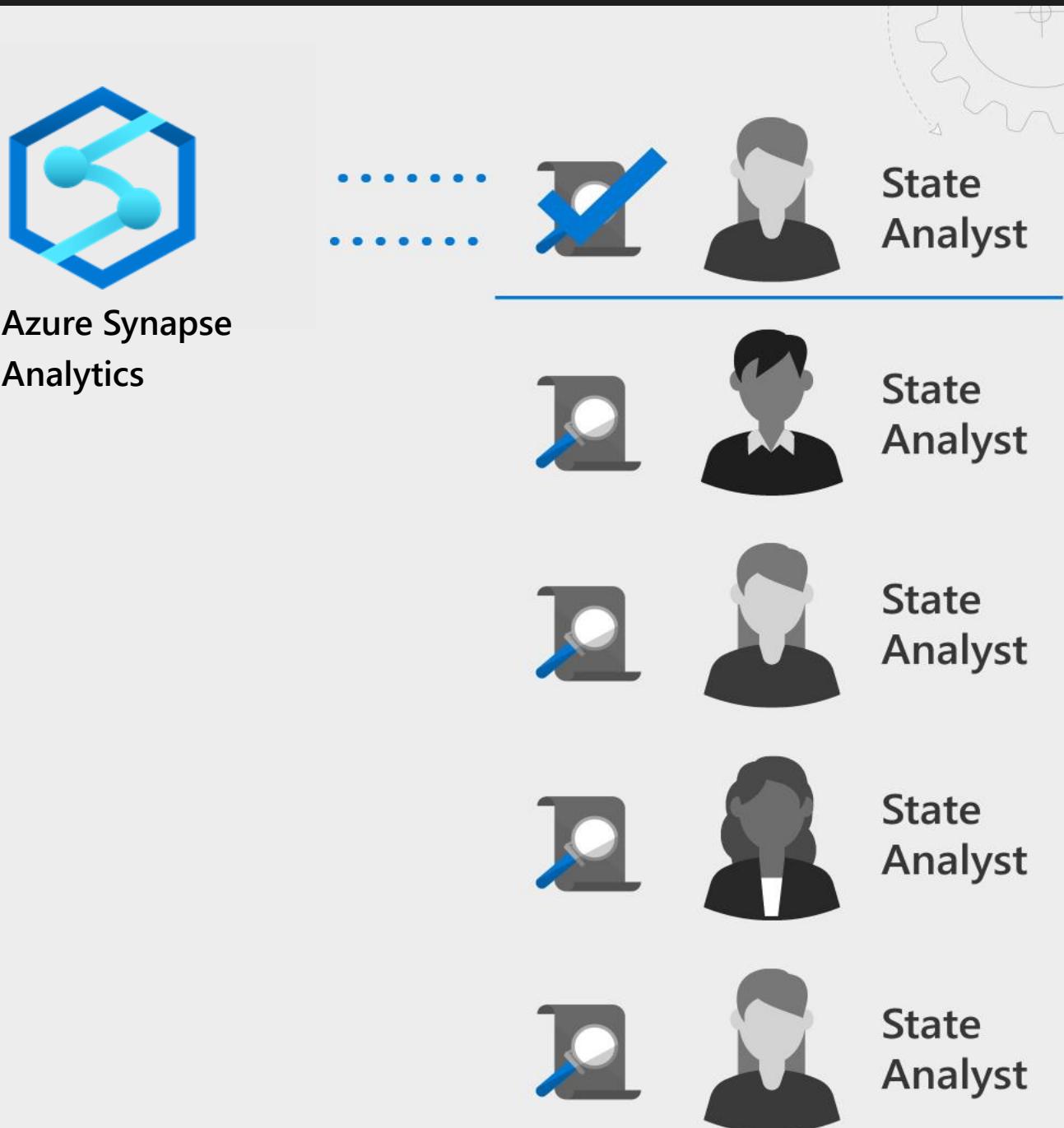
State analysts have normal importance.

National analyst is assigned high importance.

State analyst queries execute in order of arrival

When the national analyst's query arrives, it jumps to the top of the queue

```
CREATE WORKLOAD CLASSIFIER National_Analyst
WITH
(
    WORKLOAD_GROUP = 'analyst'
    ,IMPORTANCE = HIGH
    ,MEMBERNAME = 'National_Analyst_Login')
```



# Workload Isolation

## Overview

Allocate fixed resources to workload group.

Assign maximum and minimum usage for varying resources under load. These adjustments can be done live without having to Synapse SQL (provisioned) offline.

## Benefits

Reserve resources for a group of requests

Limit the amount of resources a group of requests can consume

Shared resources accessed based on importance level

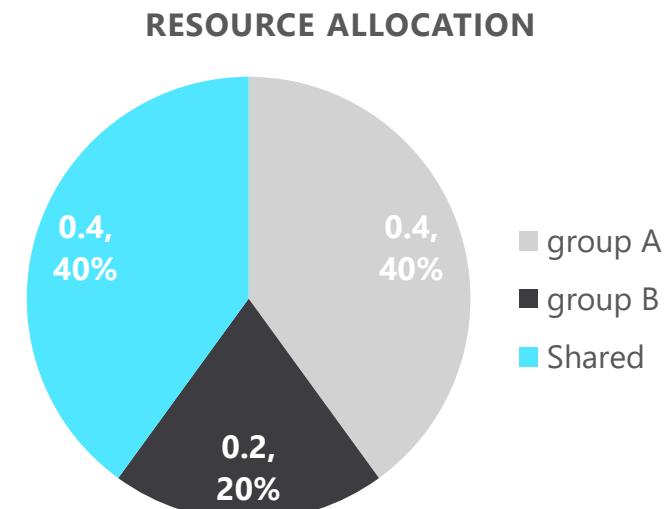
Set Query timeout value. Get DBAs out of the business of killing runaway queries

## Monitoring DMVs

[sys.workload\\_management\\_workload\\_groups](#)

Query to view configured workload group.

```
CREATE WORKLOAD GROUP group_name
WITH
(
    MIN_PERCENTAGE_RESOURCE = value
    , CAP_PERCENTAGE_RESOURCE = value
    , REQUEST_MIN_RESOURCE_GRANT_PERCENT = value
    [[,] REQUEST_MAX_RESOURCE_GRANT_PERCENT = value ]
    [[,] IMPORTANCE = {LOW | BELOW_NORMAL | NORMAL | ABOVE_NORMAL | HIGH} ]
    [[,] QUERY_EXECUTION_TIMEOUT_SEC = value ]
)[;]
```



# User & Authentication

-- Master Database

```
CREATE LOGIN MedRCLLogin WITH PASSWORD = 'rladydtjs$1971';
```

```
CREATE USER LoadingUser FOR LOGIN MedRCLLogin;
```

-- Master Database

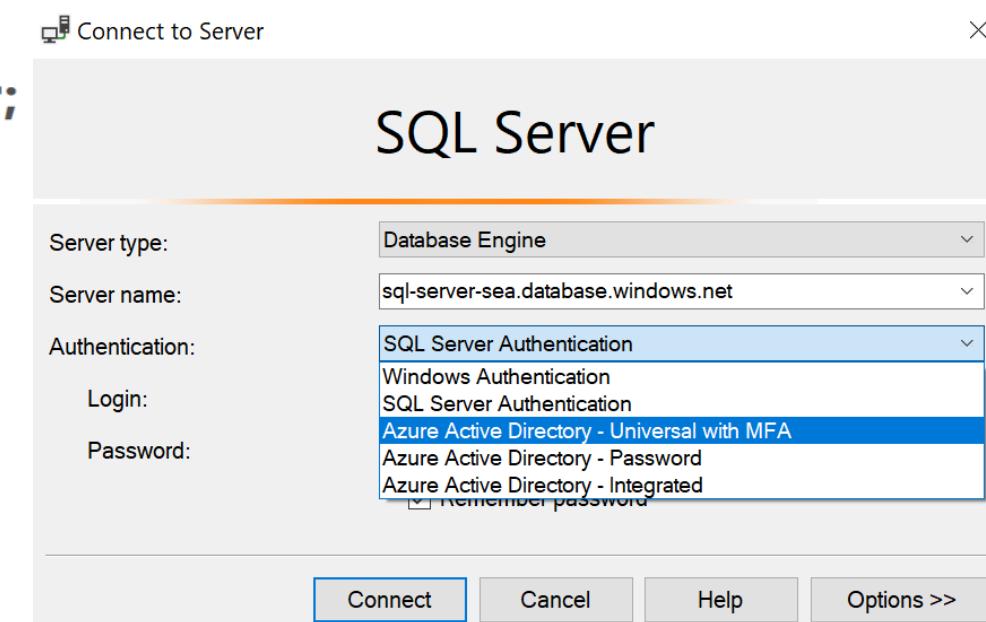
```
CREATE USER LoadingUser FOR LOGIN MedRCLLogin;
```

```
GRANT CONTROL ON DATABASE:[yongdw] to LoadingUser;
```

```
EXEC sp_addrolemember 'mediumrc', 'LoadingUser';
```

```
Create Login Mary@domainname.net From EXTERNAL PROVIDER;  

Create User Mary From Login Mary@domainname.net;
```



# Tables & Distribution

# Table Distribution Options

## Hash Distributed

Data divided across nodes based on hashing algorithm

Same value will always hash to same distribution

Single column only

Check for Data Skew,  
NULLS, -1

## Round Robin (Default)

Data distributed evenly across nodes

Easy place to start, don't need to know anything about the data

Simplicity at a cost

Will incur more data movement at query time

## Replicated

Data repeated on every node

Simplifies many query plans and reduces data movement

Best with joining hash table

Consumes more space  
Joining two Replicated Table runs on one node

# Creating tables

```
CREATE TABLE [build].[FactOnlineSales]
```

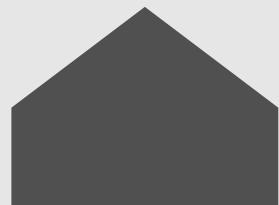
```
(  
    [OnlineSalesKey] int NOT NULL  
, [DateKey] datetime NOT NULL  
, [StoreKey] int NOT NULL  
, [ProductKey] int NOT NULL  
, [PromotionKey] int NOT NULL  
, [CurrencyKey] int NOT NULL  
, [CustomerKey] int NOT NULL  
, [SalesOrderNumber] nvarchar(20) NOT NULL  
, [SalesOrderLineNumber] int NULL  
, [SalesQuantity] int NOT NULL  
, [SalesAmount] money NOT NULL  
)
```

```
WITH
```

```
( CLUSTERED COLUMNSTORE INDEX  
, DISTRIBUTION = ROUND_ROBIN
```

```
)
```

```
;
```



```
CREATE TABLE [build].[FactOnlineSales]
```

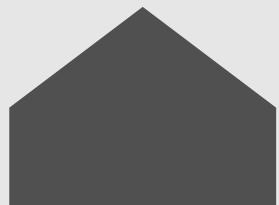
```
(  
    [OnlineSalesKey] int NOT NULL  
, [DateKey] datetime NOT NULL  
, [StoreKey] int NOT NULL  
, [ProductKey] int NOT NULL  
, [PromotionKey] int NOT NULL  
, [CurrencyKey] int NOT NULL  
, [CustomerKey] int NOT NULL  
, [SalesOrderNumber] nvarchar(20) NOT NULL  
, [SalesOrderLineNumber] int NULL  
, [SalesQuantity] int NOT NULL  
, [SalesAmount] money NOT NULL  
)
```

```
WITH
```

```
( CLUSTERED COLUMNSTORE INDEX  
, DISTRIBUTION = HASH([ProductKey])
```

```
)
```

```
;
```



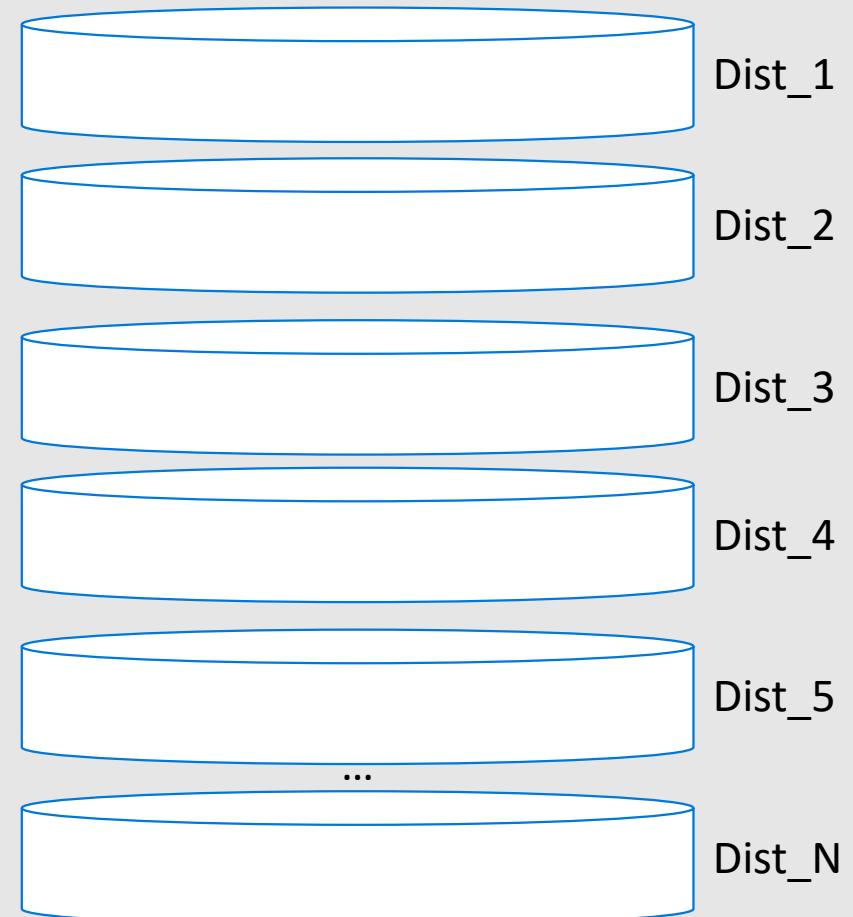
# Hash Distributed

```
CREATE TABLE ProductSales  
WITH (DISTRIBUTION=HASH(AccountID))  
AS ...
```

ProductSales – Raw Data

Account	SalesAmt	...
47	\$1,234.36	...
36	\$2,345.47	...
14	\$3,456.58	...
25	\$4,567.69	...
48	\$5,678.70	...
37	\$6,789.81	...
51	\$6,789.81	...
...	...	...

Hash(40)

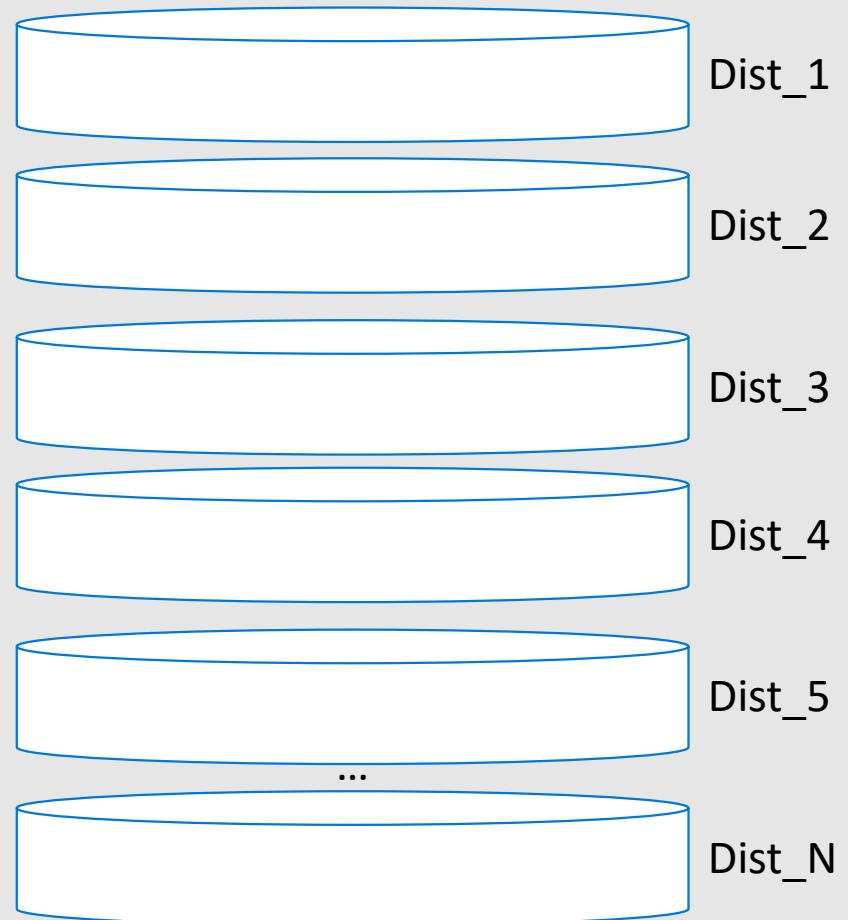


# Round Robin Distributed

```
CREATE TABLE ProductSales  
WITH (DISTRIBUTION = ROUND_ROBIN)  
AS ...
```

ProductSales – Raw Data

AccountID	SalesAmt	...
47	\$1,234.36	...
36	\$2,345.47	...
14	\$3,456.58	...
25	\$4,567.69	...
47	\$5,678.70	...
37	\$6,789.81	...
42	\$1,632.25	...
88	\$4,453.21	...
52	\$7,892.81	...
91	\$9,549.64	...
88	\$2,498.14	...
42	\$3,145.99	...
23	\$3,145.99	...



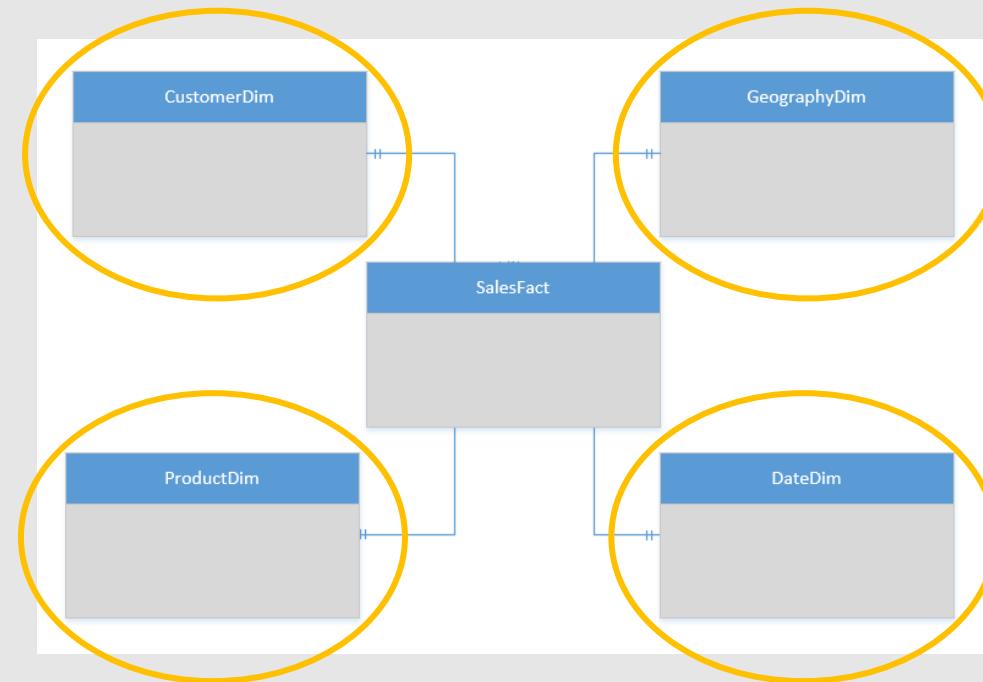
# Replicated Table Scenarios

Scenarios to consider using Replicated tables:

Star schema reporting

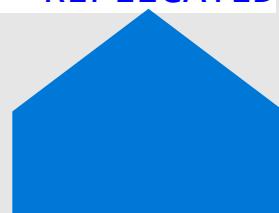
Dimensions – descriptive entities about fact data

ETL master data, common domain data used during transaction loading



# T-SQL: Create a Replicated table

```
CREATE TABLE dbo.DimCustomer
(
    CustomerKey          int           NOT NULL
,   GeographyKey        int           NULL
,   CustomerAlternateKey nvarchar(15) NOT NULL
,   Title                nvarchar(8)  NULL
,   FirstName             nvarchar(50) NULL
,   LastName              nvarchar(50) NULL
,   BirthDate            date         NULL
,   Gender                nvarchar(1)  NULL
,   EmailAddress          nvarchar(50) NULL
,   YearlyIncome          money        NULL
,   DateFirstPurchase     date         NULL
)
WITH
(
    CLUSTERED COLUMNSTORE INDEX
,   DISTRIBUTION = REPLICATED
)
```



# Why data moves

Data has to be co-located to operate on...

Common reasons:

Incompatible join

Incompatible aggregation

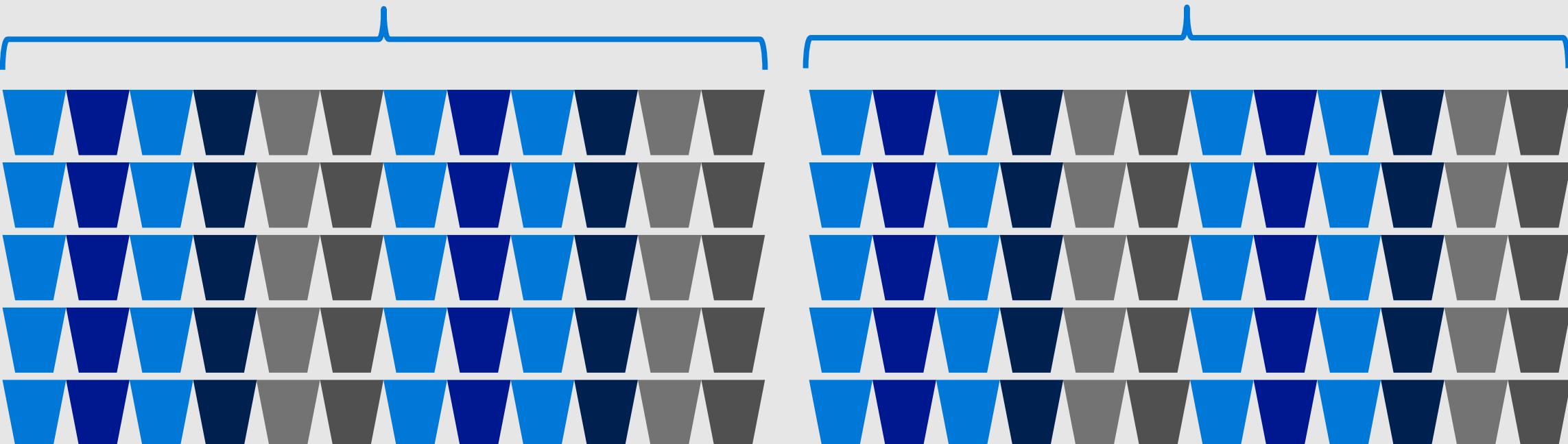
# Data Movement Types for a Query

DMS Operation	Description
ShuffleMoveOperation	Distribution → Hash algorithm → New distribution Changing the distribution column in preparation for join.
PartitionMoveOperation	Distribution → Control Node Aggregations - count(*) is count on nodes, sum of count
BroadcastMoveOperation	Distribution → Copy to all distributions Changes distributed table to replicated table for join.
TrimMoveOperation	Replicated table → Hash algorithm → Distribution When a replicated table needs to become distributed. Needed for outer joins.
MoveOperation	Control Node → Copy to all distributions Data moved from Control Node back to Compute Nodes resulting in a replicated table for further processing.

# Joining HASH tables

Store\_Sales HASH([ProductKey])  
[ProductKey] INT NULL

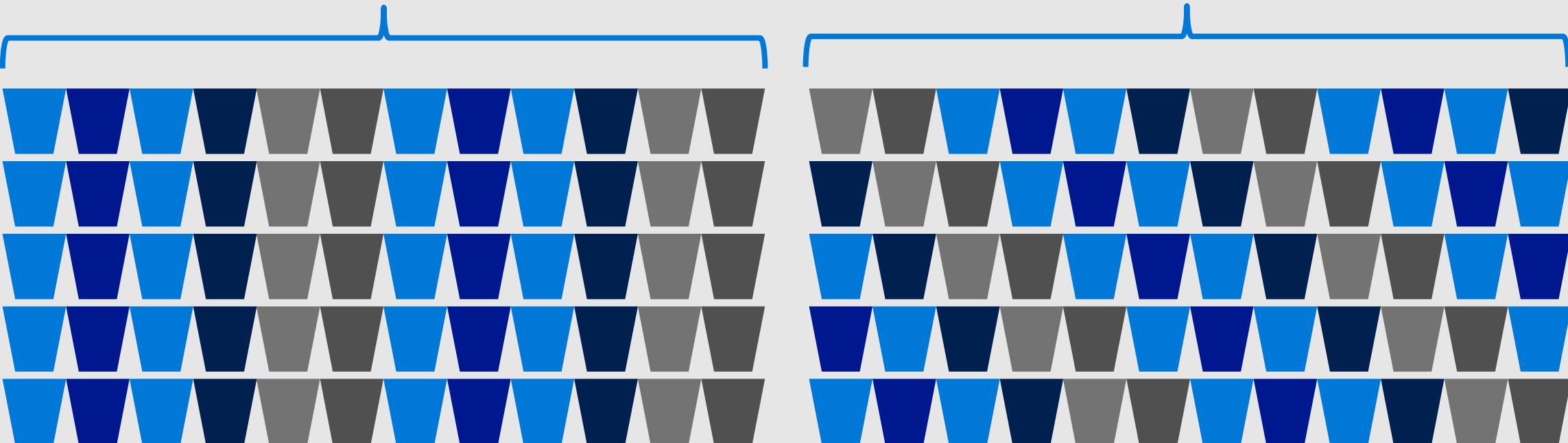
Web\_Sales HASH([ProductKey])  
[ProductKey] INT NULL



# Joining HASH tables

Store\_Sales HASH([ProductKey])  
[ProductKey] INT NULL

Web\_Sales HASH([ProductKey])  
[ProductKey] **BIGINT** NULL



# Aggregation - compatible

Resolved completely on each compute node

## No Data Movement

1. Hash Distribution Key is contained in the group by keys
2. Count Distinct on distribution key

```
-- FactOnlineSales distributed on ProductKey
```

```
SELECT COUNT_BIG(*)  
FROM [cso].[FactOnlineSales]  
GROUP BY [ProductKey]
```

```
SELECT COUNT_BIG(DISTINCT ([ProductKey]))  
FROM [cso].[FactOnlineSales]
```

# Aggregation - Incompatible

Partially aggregated on each node

Shuffle move co-locates rows with same group by key

1. Table is round robin distributed
2. Hash Distribution key is not contained in group by keys
3. Count Distinct on non-distribution key or on round robin table

```
-- FactOnlineSales distributed on ProductKey
```

```
SELECT COUNT_BIG(*)  
FROM [cso].[FactOnlineSales]  
GROUP BY [StoreKey]
```

```
SELECT COUNT_BIG(DISTINCT [DateKey])  
FROM [cso].[FactOnlineSales]
```

# Common table distribution methods

Table Category	Recommended Distribution Option
Fact	<p>Use hash-distribution with clustered columnstore index. Performance improves because hashing enables the platform to localize certain operations within the node itself during query execution.</p> <p>Operations that benefit:</p> <p>COUNT(DISTINCT( &lt;hashed_key&gt; ))</p> <p>OVER PARTITION BY &lt;hashed_key&gt;</p> <p>most JOIN &lt;table_name&gt; ON &lt;hashed_key&gt;</p> <p>GROUP BY &lt;hashed_key&gt;</p>
Dimension	Use replicated for smaller tables. If tables are too large to store on each Compute node, use hash-distributed.
Staging	Use round-robin for the staging table. The load with CTAS is faster. Once the data is in the staging table, use INSERT...SELECT to move the data to production tables.

# Distribution Guidance

For large fact tables, best option is to Hash Distribute

- Clustered Columnstore
- Distribute on column that is joined to other fact tables or large dimensions
- Primary or surrogate key maybe a good choice for distribution

However, be mindful of ...

- Hash column should have highly distinct values (Minimum 600 distinct values)
- Avoid distributing on a date column
- Avoid distributing on column with high frequency of NULLs and default values (e.g. -1)
- Distribution column is NOT updatable
- For compatible joins use the same data types for two distributed tables

If there are no distribution columns that make sense, then use Round Robin as last resort

# Skewed Distribution



# Finding Skew

## DBCC PDW\_SHOWSPACEUSED

Not a programmatic interface

DMV gives more detail and control

sys.dm\_pdw\_nodes\_db\_partition\_stats

Refer to the view dbo.vTableSizes provided in docs ([Table Size Queries](#))

```
select distribution_id, SUM(row_count) as total_distribution_row_count
from dbo.vTableSizes
where schema_name = 'Fact' and table_name = 'Flights'
group by distribution_id
order by total_distribution_row_count;
```

# Tables – Partitions

## Overview

Table partitions divide data into smaller groups

In most cases, partitions are created on a date column

Supported on all table types

RANGE RIGHT – Used for time partitions

RANGE LEFT – Used for number partitions

## Benefits

Improves efficiency and performance of loading and querying by limiting the scope to subset of data.

Offers significant query performance enhancements where filtering on the partition key can eliminate unnecessary scans and eliminate IO.

```
CREATE TABLE partitionedOrderTable
(
    OrderId INT NOT NULL,
    Date DATE NOT NULL,
    Name VARCHAR(2),
    Country VARCHAR(2)
)
WITH
(
    CLUSTERED COLUMNSTORE INDEX,
    DISTRIBUTION = HASH([OrderId]),
    PARTITION (
        [Date] RANGE RIGHT FOR VALUES (
            '2000-01-01', '2001-01-01', '2002-01-01',
            '2003-01-01', '2004-01-01', '2005-01-01'
        )
    )
);
```

# Tables – DW Distributions & Partitions Illustrated

Logical table structure

OrderId	Date	Name	Country
85016	11-2-2018	V	UK
85018	11-2-2018	Q	SP
85216	11-2-2018	Q	DE
85395	11-2-2018	V	NL
82147	11-2-2018	Q	FR
86881	11-2-2018	D	UK
93080	11-3-2018	R	UK
94156	11-3-2018	S	FR
96250	11-3-2018	Q	NL
98799	11-3-2018	R	NL
98015	11-3-2018	T	UK
98310	11-3-2018	D	DE
98979	11-3-2018	Z	DE
98137	11-3-2018	T	FR
...	...	...	...

## Physical data distribution

( Hash distribution (OrderId), Date partitions )

### Distribution1 (OrderId 80,000 – 100,000)

#### 11-2-2018 partition

OrderId	Date	Name	Country
85016	11-2-2018	V	UK
85018	11-2-2018	Q	SP
85216	11-2-2018	Q	DE
85395	11-2-2018	V	NL
82147	11-2-2018	Q	FR
86881	11-2-2018	D	UK
...	...	...	...

#### 11-3-2018 partition

OrderId	Date	Name	Country
93080	11-3-2018	R	UK
94156	11-3-2018	S	FR
96250	11-3-2018	Q	NL
98799	11-3-2018	R	NL
98015	11-3-2018	T	UK
98310	11-3-2018	D	DE
98979	11-3-2018	Z	DE
98137	11-3-2018	T	FR
...	...	...	...

...

x 60 distributions (shards)

- Each shard is partitioned with the same date partitions
- A minimum of 1 million rows per distribution and partition is needed for optimal compression and performance of clustered Columnstore tables

# Tables – Partitions

	logical_table_name	row_group_id	state	state_desc	total_rows	trim_reason_desc	physical_name
1	FactSalesQuotaCCI	3	3	COMPRESSED	514925	BULKLOAD	Table_0916c48a65e14db285dfed9c57386023_1
2	FactSalesQuotaCCI	3	3	COMPRESSED	512875	BULKLOAD	Table_0916c48a65e14db285dfed9c57386023_2
3	FactSalesQuotaCCI	3	3	COMPRESSED	511350	BULKLOAD	Table_0916c48a65e14db285dfed9c57386023_3
4	FactSalesQuotaCCI	3	3	COMPRESSED	509161	BULKLOAD	Table_0916c48a65e14db285dfed9c57386023_4
5	FactSalesQuotaCCI	3	3	COMPRESSED	512640	BULKLOAD	Table_0916c48a65e14db285dfed9c57386023_5
6	FactSalesQuotaCCI	3	3	COMPRESSED	512640	BULKLOAD	Table_0916c48a65e14db285dfed9c57386023_6
7	FactSalesQuotaCCI	3	3	COMPRESSED	512638	BULKLOAD	Table_0916c48a65e14db285dfed9c57386023_7
8	FactSalesQuotaCCI	3	3	COMPRESSED	511216	BULKLOAD	Table_0916c48a65e14db285dfed9c57386023_8
9	FactSalesQuotaCCI	3	3	COMPRESSED	511216	BULKLOAD	Table_0916c48a65e14db285dfed9c57386023_9
10	FactSalesQuotaCCI	3	3	COMPRESSED	512360	BULKLOAD	Table_0916c48a65e14db285dfed9c57386023_10
11	FactSalesQuotaCCI	3	3	COMPRESSED	511216	BULKLOAD	Table_0916c48a65e14db285dfed9c57386023_11
12	FactSalesQuotaCCI	3	3	COMPRESSED	510102	BULKLOAD	Table_0916c48a65e14db285dfed9c57386023_12
13	FactSalesQuotaCCI	3	3	COMPRESSED	509792	BULKLOAD	Table_0916c48a65e14db285dfed9c57386023_13
14	FactSalesQuotaCCI	3	3	COMPRESSED	512640	BULKLOAD	Table_0916c48a65e14db285dfed9c57386023_14
15	FactSalesQuotaCCI	3	3	COMPRESSED	512388	BULKLOAD	Table_0916c48a65e14db285dfed9c57386023_15
16	FactSalesQuotaCCI	3	3	COMPRESSED	513046	BULKLOAD	Table_0916c48a65e14db285dfed9c57386023_16
17	FactSalesQuotaCCI	3	3	COMPRESSED	513105	BULKLOAD	Table_0916c48a65e14db285dfed9c57386023_17
18	FactSalesQuotaCCI	3	3	COMPRESSED	512640	BULKLOAD	Table_0916c48a65e14db285dfed9c57386023_18
19	FactSalesQuotaCCI	3	3	COMPRESSED	512607	BULKLOAD	Table_0916c48a65e14db285dfed9c57386023_19
20	FactSalesQuotaCCI	3	3	COMPRESSED	511583	BULKLOAD	Table_0916c48a65e14db285dfed9c57386023_20
21	FactSalesQuotaCCI	3	3	COMPRESSED	512156	BULKLOAD	Table_0916c48a65e14db285dfed9c57386023_21
22	FactSalesQuotaCCI	3	3	COMPRESSED	512479	BULKLOAD	Table_0916c48a65e14db285dfed9c57386023_22
23	FactSalesQuotaCCI	3	3	COMPRESSED	511480	BULKLOAD	Table_0916c48a65e14db285dfed9c57386023_23
24	FactSalesQuotaCCI	3	3	COMPRESSED	512171	BULKLOAD	Table_0916c48a65e14db285dfed9c57386023_24
25	FactSalesQuotaCCI	3	3	COMPRESSED	512753	BULKLOAD	Table_0916c48a65e14db285dfed9c57386023_25
26	FactSalesQuotaCCI	3	3	COMPRESSED	511216	BULKLOAD	Table_0916c48a65e14db285dfed9c57386023_26
27	FactSalesQuotaCCI	3	3	COMPRESSED	509843	BULKLOAD	Table_0916c48a65e14db285dfed9c57386023_27
28	FactSalesQuotaCCI	3	3	COMPRESSED	511065	BULKLOAD	Table_0916c48a65e14db285dfed9c57386023_28
29	FactSalesQuotaCCI	3	3	COMPRESSED	510639	BULKLOAD	Table_0916c48a65e14db285dfed9c57386023_29
30	FactSalesQuotaCCI	3	3	COMPRESSED	510366	BULKLOAD	Table_0916c48a65e14db285dfed9c57386023_30
31	FactSalesQuotaCCI	3	3	COMPRESSED	511247	BULKLOAD	Table_0916c48a65e14db285dfed9c57386023_31
32	FactSalesQuotaCCI	3	3	COMPRESSED	511917	BULKLOAD	Table_0916c48a65e14db285dfed9c57386023_32

# Tables – Indexes

## Clustered Columnstore index (Default Primary)

Highest level of data compression

Best overall query performance

## Clustered index (Primary)

Performant for looking up a single to few rows

## Heap (Primary)

Faster loading and landing temporary data

Best for small lookup tables

## Nonclustered indexes (Secondary)

Enable ordering of multiple columns in a table

Allows multiple nonclustered on a single table

Can be created on any of the above primary indexes

More performant lookup queries

-- Create table with index

```
CREATE TABLE orderTable
```

```
(
```

```
    OrderId INT NOT NULL,
```

```
    Date DATE NOT NULL,
```

```
    Name VARCHAR(2),
```

```
    Country VARCHAR(2)
```

```
)
```

```
WITH
```

```
(
```

```
    CLUSTERED COLUMNSTORE INDEX |
```

```
    HEAP |
```

```
    CLUSTERED INDEX (OrderId)
```

```
);
```

-- Add non-clustered index to table

```
CREATE INDEX NameIndex ON orderTable (Name);
```

# Creating tables

```
CREATE TABLE [dbo].[DimStore]
(
    [StoreKey]           int          NOT NULL
    ,[GeographyKey]      int          NOT NULL
    ,[StoreName]          nvarchar(100) NOT NULL
    ,[StoreType]          nvarchar(15)   NULL
    ,[StoreDescription]  nvarchar(300) NOT NULL
    ,[Status]             nvarchar(20)  NOT NULL
    ,[OpenDate]           datetime     NOT NULL
    ,[CloseDate]          datetime     NULL
    ,[ETLLoadID]          int          NULL
    ,[LoadDate]            datetime     NULL
    ,[UpdateDate]         datetime     NULL
)
WITH
(
    CLUSTERED INDEX([StoreKey])
    DISTRIBUTION = ROUND_ROBIN
);
```

Row

Distribution

```
CREATE TABLE [dbo].[FactOnlineSales]
(
    [OnlineSalesKey]      int          NOT NULL
    ,[DateKey]             datetime     NOT NULL
    ,[StoreKey]            int          NOT NULL
    ,[ProductKey]          int          NOT NULL
    ,[PromotionKey]        int          NOT NULL
    ,[CurrencyKey]         int          NOT NULL
    ,[CustomerKey]         int          NOT NULL
    ,[SalesOrderNumber]    nvarchar(20) NOT NULL
    ,[SalesOrderLineNumber] int          NULL
    ,[SalesQuantity]       int          NOT NULL
    ,[SalesAmount]          money        NOT NULL
)
)
```

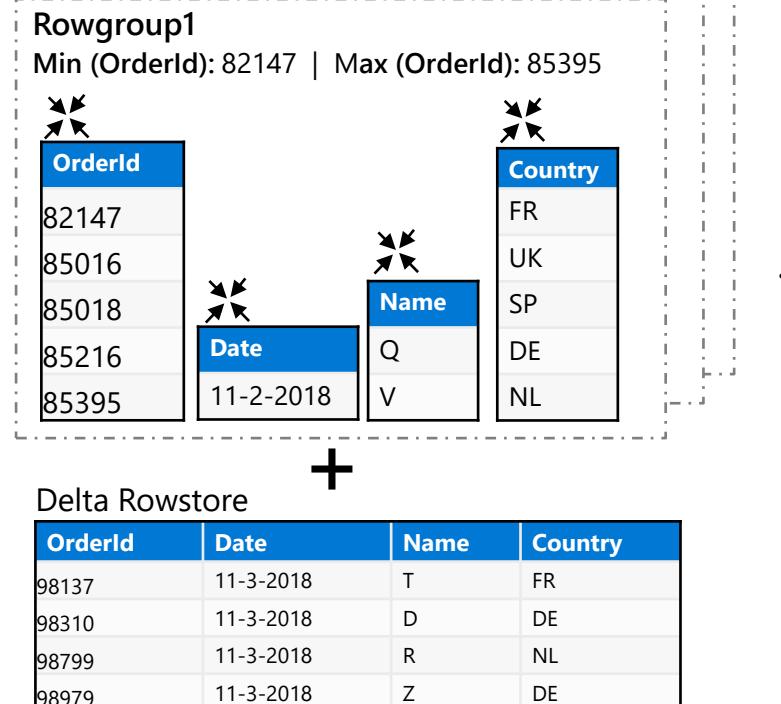
Column

# Synapse SQL (provisioned) Columnstore Tables

## Logical table structure

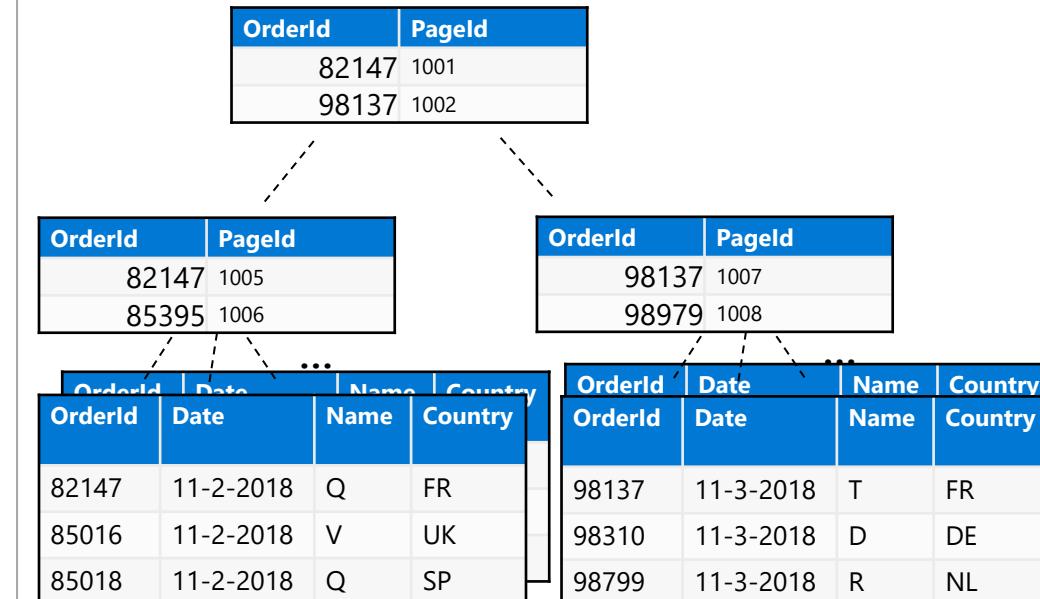
OrderId	Date	Name	Country
85016	11-2-2018	V	UK
85018	11-2-2018	Q	SP
85216	11-2-2018	Q	DE
85395	11-2-2018	V	NL
82147	11-2-2018	Q	FR
86881	11-2-2018	D	UK
93080	11-3-2018	R	UK
94156	11-3-2018	S	FR
96250	11-3-2018	Q	NL
98799	11-3-2018	R	NL
98015	11-3-2018	T	UK
98310	11-3-2018	D	DE
98979	11-3-2018	Z	DE
98137	11-3-2018	T	FR
...	...	...	...

## Clustered columnstore index (OrderId)



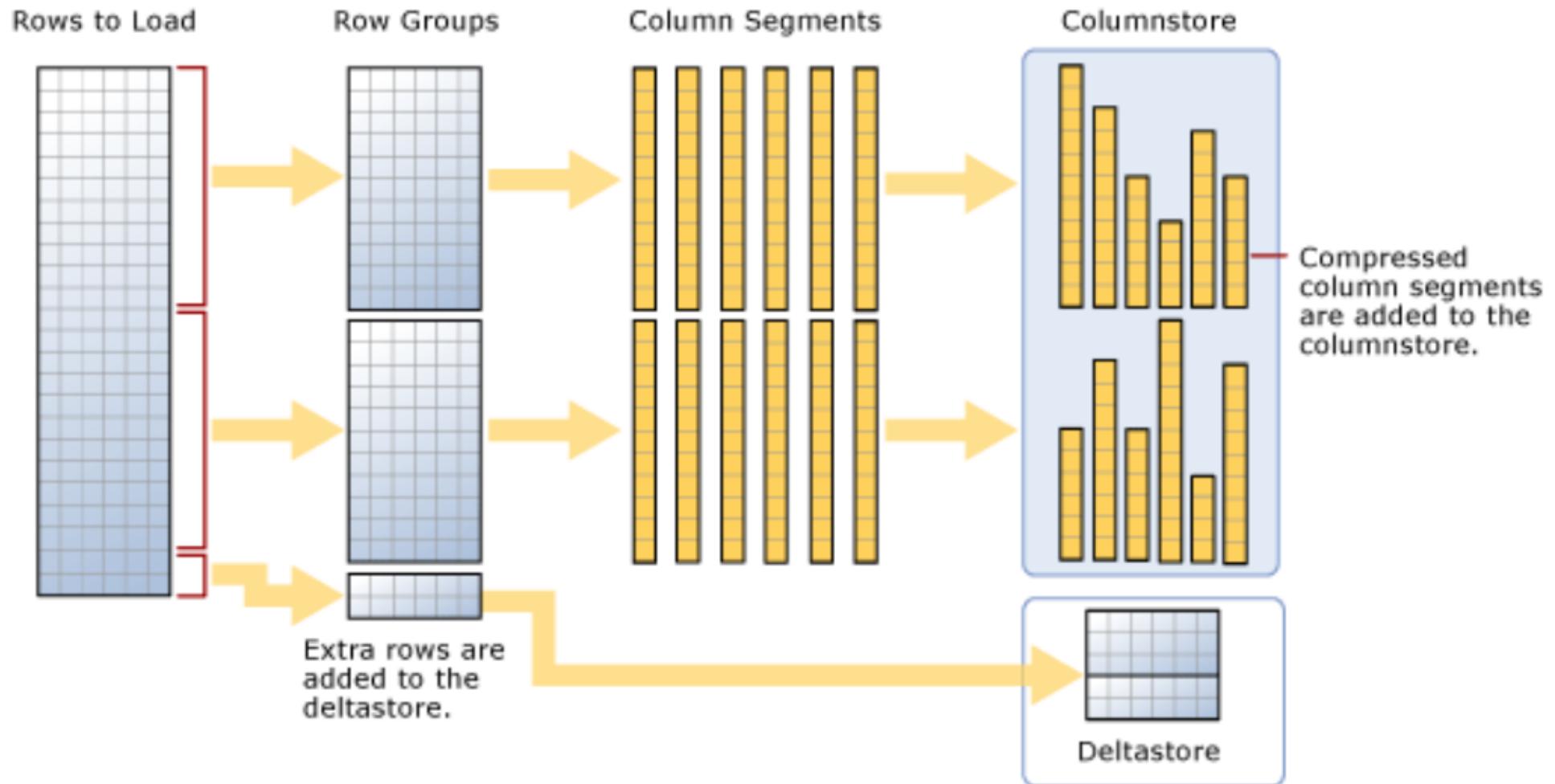
- Data stored in compressed columnstore segments after being sliced into groups of rows (rowgroups/micro-partitions) for maximum compression
- Rows are stored in the delta rowstore until the number of rows is large enough to be compressed into a columnstore

## Clustered/Non-clustered rowstore index (OrderId)



- Data is stored in a B-tree index structure for performant lookup queries for particular rows.
- Clustered rowstore index: The leaf nodes in the structure store the data values in a row (as pictured above)
- Non-clustered (secondary) rowstore index: The leaf nodes store pointers to the data values, not the values themselves

# Columnstore Storage Model



# Columnstore Storage Model

## ROW STORE

```
SQLQuery1.sql - D...(DCAC\monica (65))* → X
Editor Results Messages Execution plan
SQL Server parse and compile time:
    CPU time = 405 ms, elapsed time = 589 ms.

(395 rows affected)
Table 'Worktable'. Scan count 0, logical reads 0, physical reads 0, read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob read-ahead reads 0.
Table 'Workfile'. Scan count 0, logical reads 0, physical reads 0, read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob read-ahead reads 0.
Table 'FactResellerSalesXL'. Scan count 5, logical reads 318076, physical reads 2, read-ahead reads 291341, lob logical reads 0, lob physical reads 0, lob read-ahead reads 0.

(1 row affected)

SQL Server Execution Times:
    CPU time = 8233 ms, elapsed time = 5008 ms.
SQL Server parse and compile time:
    CPU time = 0 ms, elapsed time = 0 ms.

SQL Server Execution Times:
    CPU time = 0 ms, elapsed time = 0 ms.

SQL Server Execution Times:
    CPU time = 0 ms, elapsed time = 0 ms.
```

## COLUMNSTORE

```
Editor Results Messages Execution plan
SQL Server parse and compile time:
    CPU time = 0 ms, elapsed time = 0 ms.

SQL Server Execution Times:
    CPU time = 0 ms, elapsed time = 0 ms.
SQL Server parse and compile time:
    CPU time = 0 ms, elapsed time = 31 ms.

(395 rows affected)
Table 'FactResellerSalesXL_CCI'. Scan count 4, logical reads 0, physical reads 0, read-ahead reads 0, lob logical reads 8208, lob physical reads 1, lob read-ahead reads 22486.
Table 'FactResellerSalesXL_CCI'. Segment reads 12, segment skipped 0.
Table 'Worktable'. Scan count 0, logical reads 0, physical reads 0, read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob read-ahead reads 0.

(1 row affected)

SQL Server Execution Times:
    CPU time = 391 ms, elapsed time = 442 ms.
SQL Server parse and compile time:
    CPU time = 0 ms, elapsed time = 0 ms.

SQL Server Execution Times:
    CPU time = 0 ms, elapsed time = 0 ms.
```

# Columnstore Storage Model

## Clustered Index Scan (Clustered)

Scanning a clustered index, entirely or only a range.

Physical Operation	Clustered Index Scan
Logical Operation	Clustered Index Scan
Actual Execution Mode	Row
Estimated Execution Mode	Row
Storage	RowStore
Number of Rows Read	11669638
Actual Number of Rows	11669638
Actual Number of Batches	0
Estimated Operator Cost	240.048 (90%)
Estimated I/O Cost	233.63
Estimated CPU Cost	6.41838
Estimated Subtree Cost	240.048
Number of Executions	4
Estimated Number of Executions	1
Estimated Number of Rows to be Read	11669600
Estimated Number of Rows	11669600
Estimated Row Size	21 B
Actual Rebinds	0
Actual Rewinds	0
Ordered	False
Node ID	7

### Object

[AdventureworksDW2016CTP3].[dbo].[FactResellerSalesXL].  
[PK\_FactResellerSalesXL\_SalesOrderNumber\_SalesOrderLineNum  
ber]

### Output List

[AdventureworksDW2016CTP3].[dbo].  
[FactResellerSalesXL].ProductKey, [AdventureworksDW2016CTP3].  
[dbo].[FactResellerSalesXL].OrderQuantity,  
[AdventureworksDW2016CTP3].[dbo].  
[FactResellerSalesXL].SalesAmount

## Columnstore Index Scan (Clustered)

Scan a columnstore index, entirely or only a range.

Physical Operation	Columnstore Index Scan
Logical Operation	Clustered Index Scan
Actual Execution Mode	Batch
Estimated Execution Mode	Batch
Storage	ColumnStore
Actual Number of Rows	0
Actual Number of Batches	0
Estimated Operator Cost	1.58719 (36%)
Estimated I/O Cost	0.945347
Estimated CPU Cost	0.641838
Estimated Subtree Cost	1.58719
Number of Executions	4
Estimated Number of Executions	1
Estimated Number of Rows	11669600
Estimated Number of Rows to be Read	11669600
Estimated Row Size	21 B
Actual Rebinds	0
Actual Rewinds	0
Ordered	False
Actual Number of Locally Aggregated Rows	11669638
Node ID	3

### Object

[AdventureworksDW2016CTP3].[dbo].[FactResellerSalesXL\_CCI].  
[IndFactResellerSalesXL\_CCI]

### Output List

[AdventureworksDW2016CTP3].[dbo].[FactResellerSalesXL\_CCI].ProductKey,  
[AdventureworksDW2016CTP3].[dbo].  
[FactResellerSalesXL\_CCI].OrderQuantity, [AdventureworksDW2016CTP3].  
[dbo].[FactResellerSalesXL\_CCI].SalesAmount

# Ordered Clustered Columnstore Indexes

## Overview

Queries against tables with ordered columnstore segments can take advantage of improved segment elimination to drastically reduce the time needed to service a query.

### -- Create Table with Ordered Columnstore Index

```
CREATE TABLE sortedOrderTable
```

```
(  
    OrderId INT NOT NULL,  
    Date DATE NOT NULL,  
    Name VARCHAR(2),  
    Country VARCHAR(2)
```

```
)  
WITH  
(  
    CLUSTERED COLUMNSTORE INDEX ORDER (OrderId)
```

### -- Create Clustered Columnstore Index on existing table

```
CREATE CLUSTERED COLUMNSTORE INDEX cciOrderId  
ON dbo.OrderTable ORDER (OrderId)
```

```
-- Insert data into table with ordered columnstore index
```

```
INSERT INTO sortedOrderTable
```

```
VALUES (1, '01-01-2019','Dave', 'UK')
```

# Statistics

Database Properties - yongsynapse

Select a page: General, Options, Permissions

Script Help

Collation: SQL\_Latin1\_General\_CI\_AS

Compatibility level: SQL Server 2016 (130)

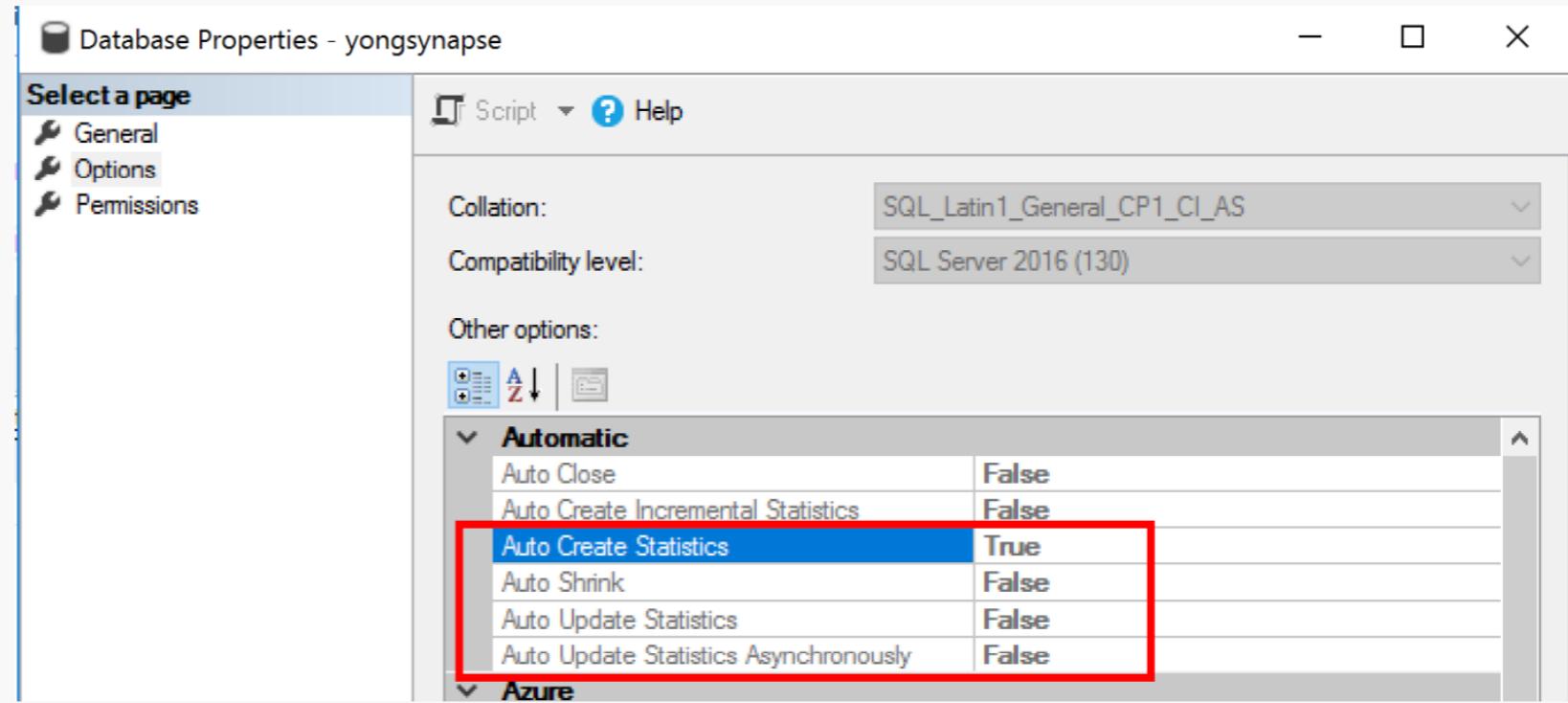
Other options:

Auto Close: False  
Auto Create Incremental Statistics: False  
**Auto Create Statistics: True**  
Auto Shrink: False  
Auto Update Statistics: False  
Auto Update Statistics Asynchronously: False

CREATE STATISTICS [statistics\_name] ON [schema\_name].[table\_name]([column\_name]) WITH FULLSCAN;

CREATE STATISTICS col1\_stats ON dbo.table1 (col1) WITH SAMPLE = 50 PERCENT;

UPDATE STATISTICS [dbo].[table1] ([stats\_col1]);



'Optimized for Elasticity'  
Sizing & Storage tiers

# Sizing factors

Database capacity

Tempdb

Concurrency & Memory

Load

Transaction size

Memory management



# Sizing Node Count

## As Is System's Node Count

What is Hadoop Worker Node Count?

What is Greenplum DB Total Node Count?

Performance level	Compute nodes
DW100c	1
DW200c	1
DW300c	1
DW400c	1
DW500c	1
DW1000c	2
DW1500c	3
DW2000c	4
DW2500c	5
DW3000c	6
DW5000c	10
DW6000c	12
DW7500c	15
DW10000c	20
DW15000c	30
DW30000c	60

# Sizing CPU & Memory

## CPU (Hyperthreaded vCore)

Convert to DTU = DWU \* 9

vCore = DTU / 100 (DW6000 \* 9 = 54,000 DTU) = 540 vCore

## Memory

DWU \* 60

(DW6000c \* 60 = 3.6 Tb)

Performance level	Memory per data warehouse (GB)
DW100c	60
DW200c	120
DW300c	180
DW400c	240
DW500c	300
DW1000c	600
DW1500c	900
DW2000c	1,200
DW2500c	1,500
DW3000c	1,800
DW5000c	3,000
DW6000c	3,600
DW7500c	4,500
DW10000c	6,000
DW15000c	9,000
DW30000c	18,000

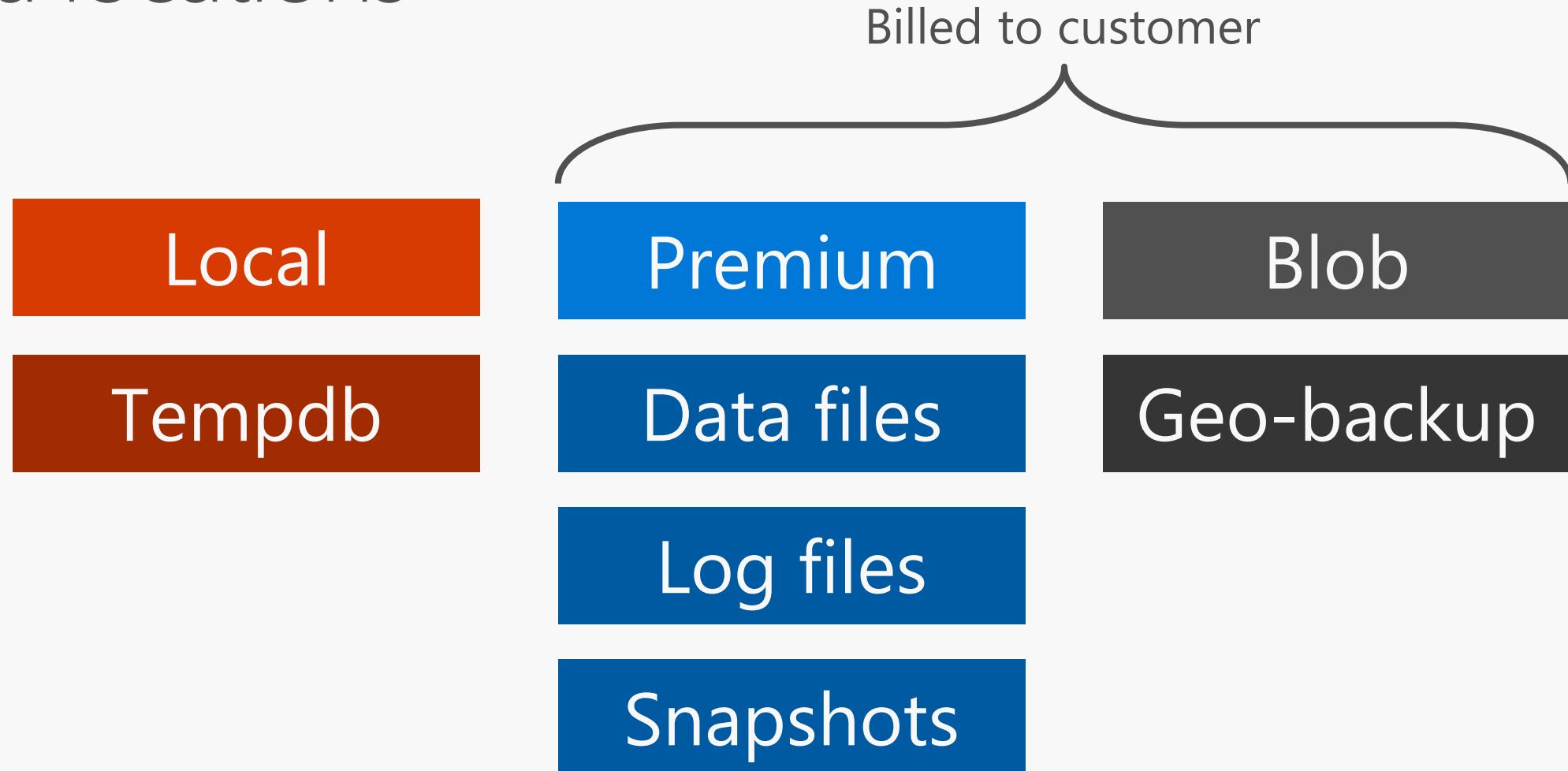
# Storage tiers

Local storage

Premium storage (remote)

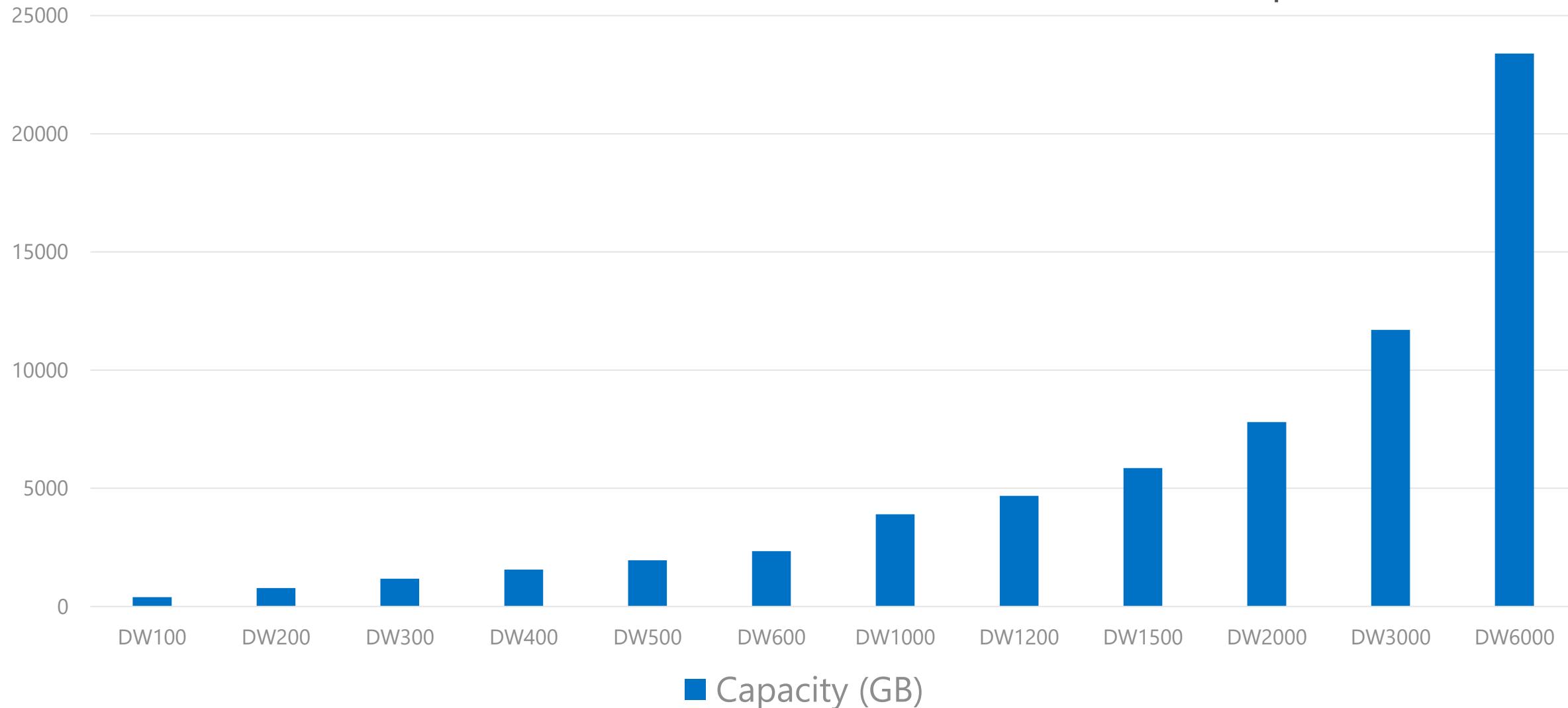
Blob storage (remote and geo redundant)

# Data locations



# Local Storage: Tempdb sizing

~399GB  
per DW100



# Premium Storage: Capacity limits

240TB

File capacity

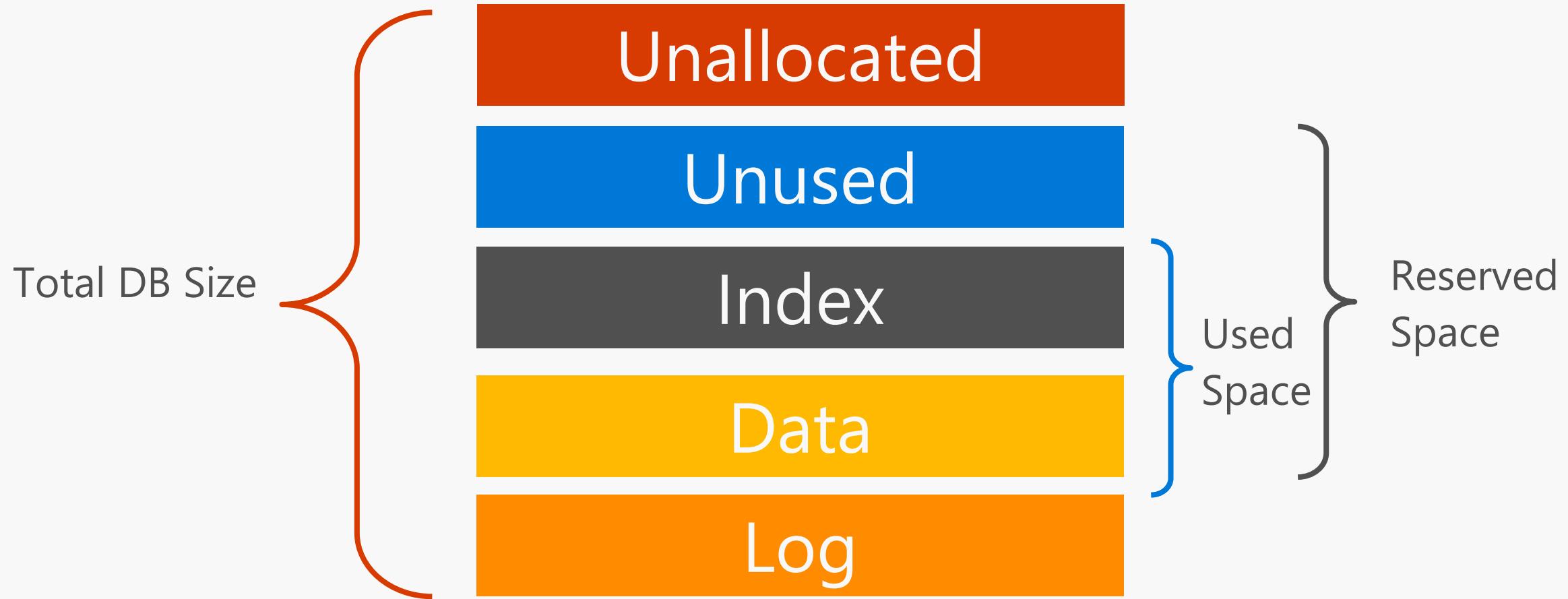
5x

CCI compression

> 1 PB

Db capacity

# Premium Storage: Database Size



# Premium Storage: Snapshots

Frequency

Retention

4

hours

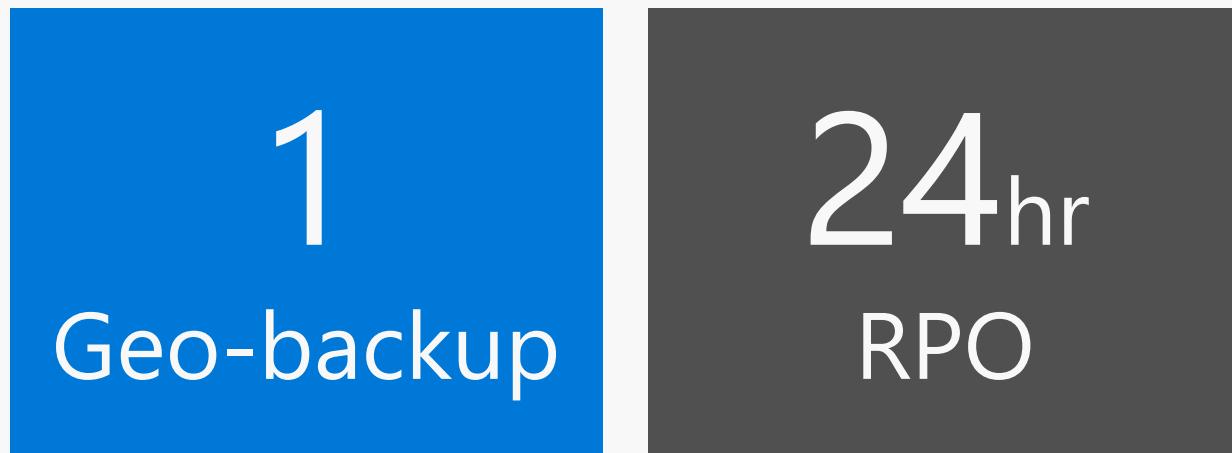
7

days

RPO : 8 Hours

# Blob Storage: Geo-redundant backups

## Frequency and Retention



Geo-backup policy

Save Discard Feedback



Geo-backup copies one of the automatic snapshots each day to RA-GRS storage. This can be used in an event of a disaster to recover your data warehouse to a new region.

[Learn more](#)

Geo-backup policy

Enabled Disabled

---

Most recent Geo-backup time

✓ 2016-07-25T06:25:39Z

# Loading

# Sizing for the data load

Delimited text guidance.

- Evenly split the data into multiple files.
- One file per reader.
- Delimited text is the fastest.

Compressed text limits concurrent access to text files

Split data across files  
OR  
Use different file format

DWU	Readers	Writers
DW100	8	60
DW200	16	60
DW300	24	60
DW400	32	60
DW500	40	60
DW600	48	60
DW1000	60	60

# Data loading

## Exception

Target Table = CI or NCI

Load user is defaultrc

## 60 Writers

## TAKE AWAY

Use mediumrc+ for  
high DWU loads

DWU	Max External Readers	Max Writers
DW100	8	60
DW200	16	60
DW300	24	60
DW400	32	60
DW500	40	60
DW600	48	60
DW1000	80	80
DW1200	96	96
DW1500	120	120
DW2000	160	160
DW3000	240	240
DW6000	480	480

# Loading Option

PolyBase	BCP	SSIS	ADF
<p>Fastest and preferred load option.</p> <p>Use CTAS for initial load.</p> <p>Use INSERT/INTO for incremental load or CTAS into stage table and partition switch into final table.</p>	<p>Use only for small files &lt; <b>10 GB</b>.</p> <p>Limited retry logic.</p> <p>Does not scale as you increase DWU (single thread, single CPU on client).</p> <p>Increase parallel threads to improve performance.</p>	<p>Increase client timeout at least 10 min, default 30 sec.</p> <p>Increase parallel threads to improve performance.</p> <p>Slight performance improvement &amp; greater reliability if run on VM.</p>	<p>A simpler way to use PolyBase.</p> <p>Quick to configure since you don't need to define the T-SQL objects.</p>

# Polybase Sample

```
CREATE MASTER KEY;

CREATE DATABASE SCOPED CREDENTIAL AzureStorageCredential
WITH
    IDENTITY = 'user',
    SECRET =
'dIzoQupXULTk/Dy09gXWbCfP2hD6TJ8H9Lf0yp0T+ubQ3uy//PB9tNxbt04WN47dT0strIHbw83zAkB1bm+Xiw=='
;

CREATE EXTERNAL DATA SOURCE AzureStorage
WITH (
    TYPE = HADOOP,
    LOCATION = 'wasbs://datacontainer@yongstorage.blob.core.windows.net',
    CREDENTIAL = AzureStorageCredential
);
CREATE EXTERNAL FILE FORMAT TextFile
WITH (
    FORMAT_TYPE = DelimitedText,
    FORMAT_OPTIONS (FIELD_TERMINATOR = ',')
);
```

# Polybase Sample

```
CREATE EXTERNAL TABLE dbo.DimDate2External (
    DateId INT NOT NULL,
    CalendarQuarter TINYINT NOT NULL,
    FiscalQuarter TINYINT NOT NULL
)
WITH (
    LOCATION='/datedimension/',
    DATA_SOURCE=AzureStorage,
    FILE_FORMAT=TextFile
);

CREATE TABLE dbo.DimDate2
WITH
(
    CLUSTERED COLUMNSTORE INDEX,
    DISTRIBUTION = ROUND_ROBIN
)
AS
SELECT * FROM [dbo].[DimDate2External];
```

# Polybase Sample

```
CREATE EXTERNAL TABLE dbo.DimDate2External (
    DateId INT NOT NULL,
    CalendarQuarter TINYINT NOT NULL,
    FiscalQuarter TINYINT NOT NULL
)
WITH (
    LOCATION='/datedimension/',
    DATA_SOURCE=AzureStorage,
    FILE_FORMAT=TextFile
);

CREATE TABLE dbo.DimDate2
WITH
(
    CLUSTERED COLUMNSTORE INDEX,
    DISTRIBUTION = ROUND_ROBIN
)
AS
SELECT * FROM [dbo].[DimDate2External];
```

# Monitoring

# Azure Synapse Monitoring Tools

- Azure Portal
- TSQL and Data Management Views (DMVs)
- Azure Monitor
- Azure Advisors
- SQL Server Management Studio
- Azure Data Studio
- Visual Studio Data Tools

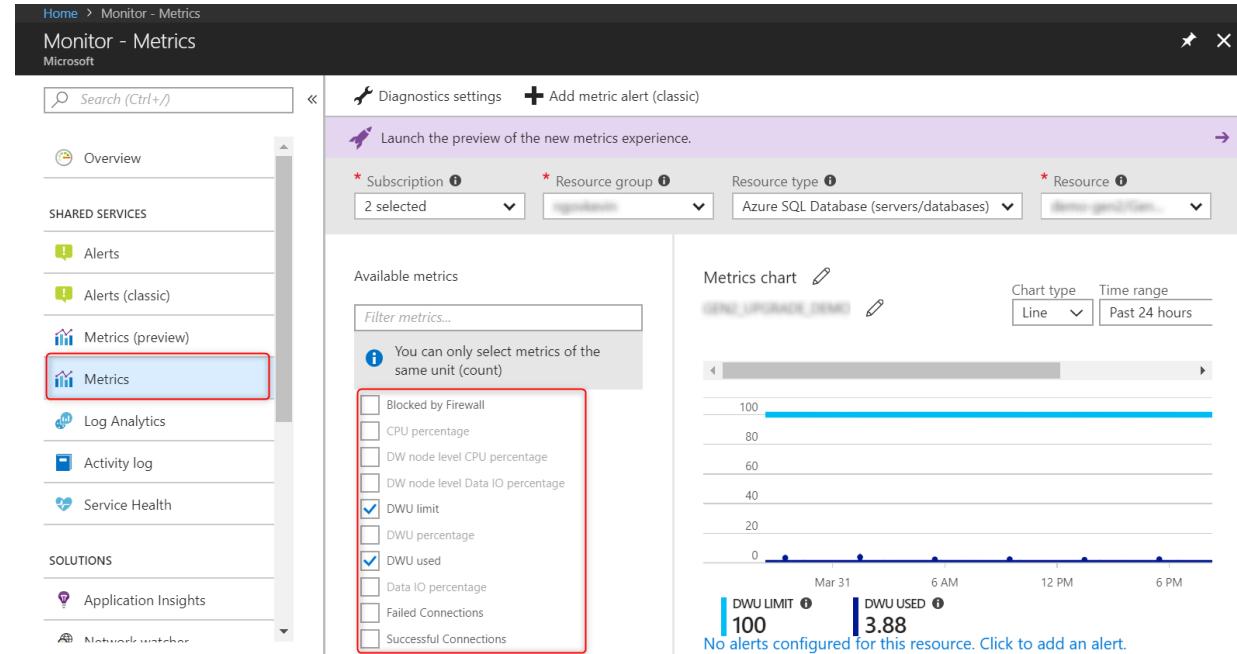
# Monitoring using the Portal

## • Overview

- Metrics surfaced through Azure Monitor
- Tracks metrics related to utilization and security
- Export to Log Analytics for further analysis
- No cost for first 31 days of each GB of data

## • Benefits

- Holistic monitoring across Azure analytics platform
- Customize charts and create enriched dashboards
- Identify under or over-utilization
- Make informed scaling decisions



## Metrics

CPU Percentage

Data IO Percentage

Successful Connections

Failed Connections

Blocked Connections

DWU (Compute) Limit

DWU Percentage (Used)

Cache Hit Percentage

Cache Used Percentage

Local tempdb Percentage

# Azure Monitor - Alerts

## Overview

Set alerts on metrics or activity logs.

Define metric thresholds.

Track individual or count of log events.

Send emails or call webhooks on triggers.

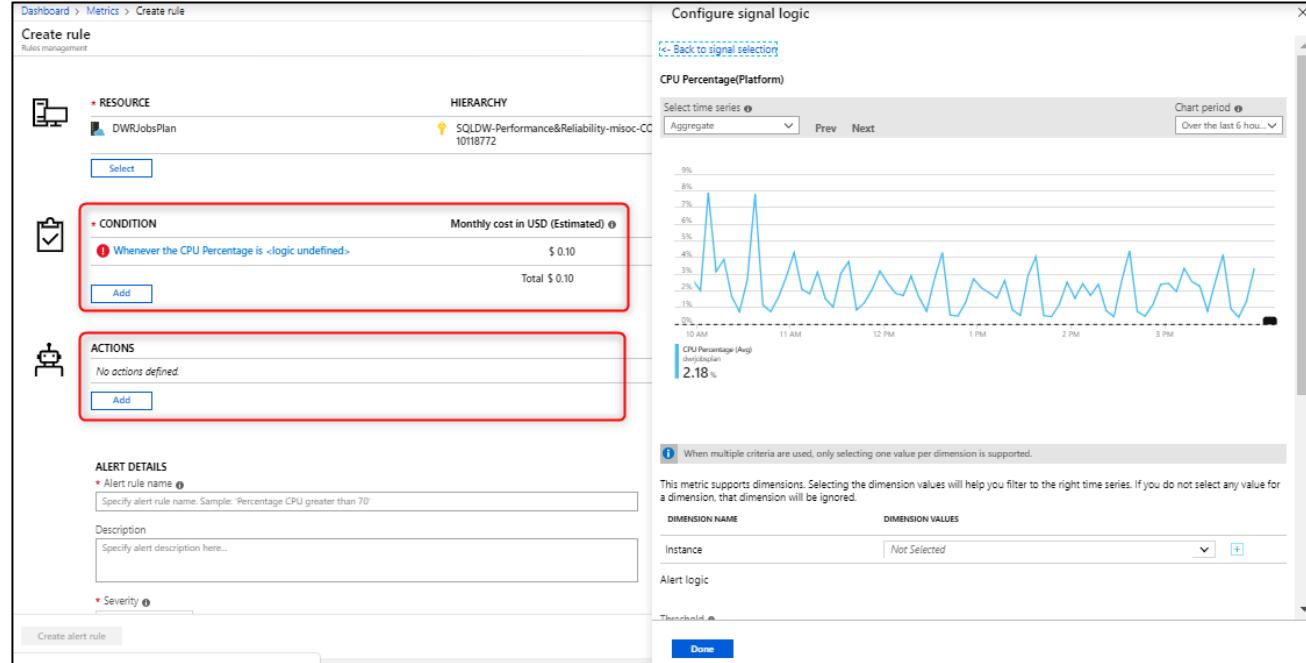
Manage with Azure portal, PowerShell, CLI, and REST APIs.

## Benefits

Automatically track data warehouse state.

Customizable triggers for immediate action.

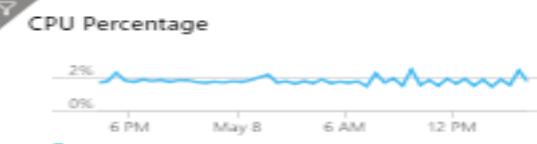
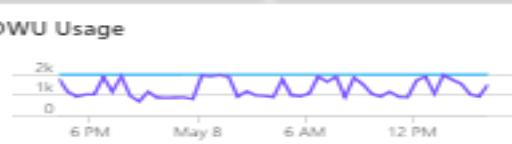
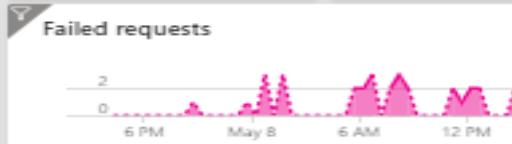
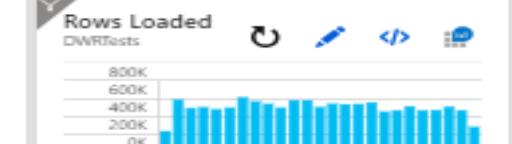
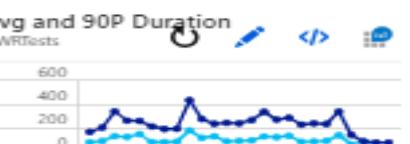
Integration with SQL Data Warehouse monitoring.



# Azure Dashboards

DW Reliability Pipeline ▾ + New dashboard ⚡ Upload ⚡ Download Edit Unshare Full screen Clone Delete

UTC Time : Past 24 hours

<b>dwareliability</b> SQL server Available	<b>CPU Percentage</b>  CPU Percentage (Avg) DWJobsPlan <b>1.85 %</b>	<b>Memory Percentage</b>  Memory Percentage (Avg) DWJobsPlan <b>39.35 %</b>	<b>README</b> Links & Various <a href="#">Edit</a>  Lists resources for both test and production (aka jobs) environments for the pipeline. <ul style="list-style-type: none"><li>• Pipeline Monitoring Report</li><li>• Git Repository</li></ul>			
<b>DWRJobs</b> SQL Data Warehouse Online	<b>DWRJobs</b> Function App Running	<b>DWRJobs</b> Application Insights 	<b>dwrjobsstorage</b> Available	<b>DWRTagging</b> Logic app 	<b>DWRAlerting</b> Logic app 	<b>Live Stream</b> DWRJobs <b>1 servers</b>
<b>DWU Usage</b>  DWU limit (Max) dwareliability/dwreliability <b>1.99 k</b>	<b>Failed requests</b>  Failed requests (Sum) dwrjobs <b>35</b>	<b>Rows Loaded</b> DWJobs  DWJobs <b>15M</b>	<b>Avg and 90P Duration</b> DWJobs  DWJobs <b>1.5K</b>			
<b>Test</b> SQL Data Warehouse Online	<b>DWRTests</b> Function App Running	<b>DWRTests</b> Application Insights 	<b>dwrtestsstorage</b> Available	<b>DWRTTestTagging</b> Logic app 	<b>DWRTTestAlerting</b> Logic app 	<b>Live Stream</b> DWRTests <b>1 servers</b>
<b>DWU Usage</b>  DWU limit (Max) dwareliability/test <b>100</b>	<b>Failed requests</b>  Failed requests (Sum) dwrtests <b>52</b>	<b>Rows Loaded</b> DWRTTests  DWRTTests <b>800K</b>	<b>Avg and 90P Duration</b> DWRTTests  DWRTTests <b>600</b>			

# SQL Pool – Database Auditing

## Configure Auditing

- Server-level vs. database-level auditing
- Default auditing policy includes all actions plus
  - BATCH\_COMPLETED\_GROUP
  - SUCCESSFUL\_DATABASE\_AUTHENTICATION\_GROUP
  - FAILED\_DATABASE\_AUTHENTICATION\_GROUP
- Use [PowerShell](#) or [RestAPI](#) to customize the audited events

## Analyze Audit Logs and reports

- **Azure Monitor Logs**
  - Auditing blade – View audit logs
  - Open in OMS
  - Log Analytics blade
- **Event Hub**
- **Azure Storage Account**
  - Azure Storage Explorer
  - Auditing blade – View audit logs
  - System function sys.fn\_get\_audit\_file
  - SSMS
  - PowerBI
  - PowerShell (query extended events files)

# Dynamic Management Views (DMVs)

## Overview

Dynamic Management Views (DMV) are queries that return information about model objects, server operations, and server health.

## Benefits:

Simple SQL syntax

Returns result in table format

Easier to read and copy result

# SQL Monitor with DMVs

## Overview

Offers monitoring of

- all open, closed sessions
- count sessions by user
- count completed queries by user
- all active, complete queries
- longest running queries
- memory consumption

Count sessions by user

--count sessions by user

```
SELECT login_name, COUNT(*) as session_count FROM sys.dm_pdw_exec_sessions  
where status = 'Closed' and session_id <> session_id() GROUP BY login_name;
```

List all open sessions

-- List all open sessions

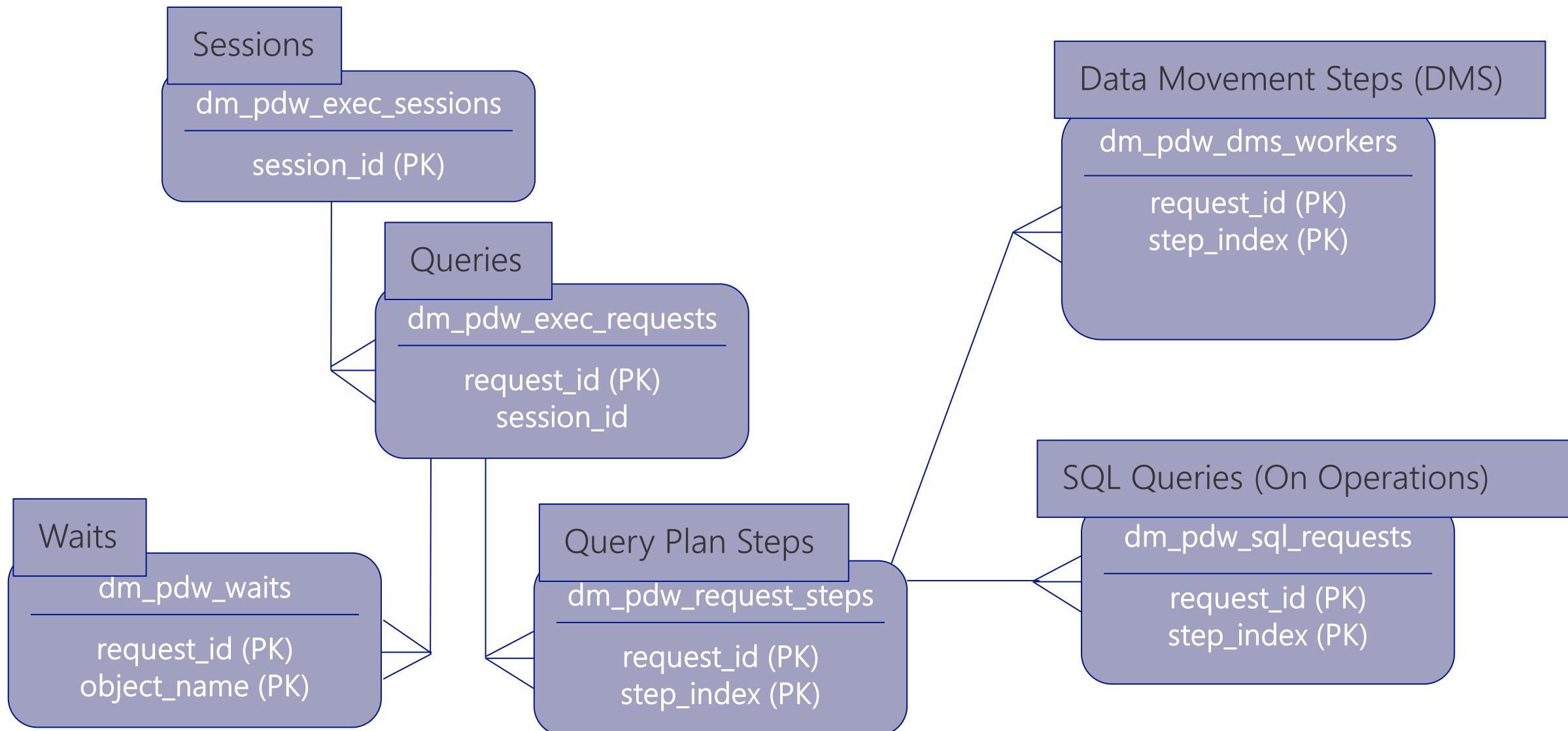
```
SELECT * FROM sys.dm_pdw_exec_sessions where status <> 'Closed' and session_id <>  
session_id();
```

List all active queries

-- List all active queries

```
SELECT * FROM sys.dm_pdw_exec_requests WHERE status not in  
('Completed', 'Failed', 'Cancelled') AND session_id <> session_id() ORDER BY submit_time  
DESC;
```

# Monitoring using DMVs



# Query Tuning Point Explorations (Session)

```
SELECT * FROM sys.dm_pdw_exec_sessions where status <> 'Closed' and session_id <> session_id();
```

	session_id	status	request_id	security_id	login_name	login_time	query_count
1	SID198	Idle	NULL	NULL	MedRCLLogin	2017-11-03 08:02:57.973	1
2	SID196	Idle	NULL	NULL	yongkim	2017-11-03 08:01:24.893	1
3	SID158	Idle	NULL	NULL	yongkim	2017-11-03 06:18:31.310	20
4	SID178	Idle	NULL	NULL	yongkim	2017-11-03 07:15:56.457	12

is_transactional	client_id	app_name	sql_spid
0	220.88.36.134:6329	Microsoft SQL Server Management Studio	123
0	220.88.36.134:47048	Microsoft SQL Server Management Studio	116
0	220.88.36.134:6776	Microsoft SQL Server Management Studio - Query	134
0	220.88.36.134:30120	Microsoft SQL Server Management Studio - Query	183

# Query Tuning Point Exploration (Long Running Query)

```
SELECT TOP 10 * FROM sys.dm_pdw_exec_requests ORDER BY total_elapsed_time DESC;
```

	request_id	session_id	status	submit_time	start_time	end_compile_time	end_time
1	QID1505	SID178	Completed	2017-11-03 07:19:35.193	2017-11-03 07:19:35.380	2017-11-03 07:19:35.380	2017-11-03 07:19:43.270
2	QID1507	SID178	Completed	2017-11-03 07:20:23.543	2017-11-03 07:20:24.107	2017-11-03 07:20:24.090	2017-11-03 07:20:27.923
3	QID1439	SID157	Completed	2017-11-03 06:18:14.437	2017-11-03 06:18:14.873	2017-11-03 06:18:14.873	2017-11-03 06:18:17.437
4	QID1468	SID158	Completed	2017-11-03 06:30:51.940	2017-11-03 06:30:54.110	2017-11-03 06:30:54.110	2017-11-03 06:30:54.203
5	QID1440	SID157	Completed	2017-11-03 06:18:14.967	2017-11-03 06:18:16.703	2017-11-03 06:18:16.670	2017-11-03 06:18:17.093
6	QID1560	SID194	Failed	2017-11-03 08:08:48.967	NULL	NULL	2017-11-03 08:08:49.720
7	QID1459	SID158	Completed	2017-11-03 06:18:35.593	2017-11-03 06:18:36.030	2017-11-03 06:18:36.030	2017-11-03 06:18:36.280
8	QID1553	SID194	Completed	2017-11-03 08:05:06.300	2017-11-03 08:05:06.377	2017-11-03 08:05:06.377	2017-11-03 08:05:06.753
9	QID1453	SID159	Completed	2017-11-03 06:18:32.390	2017-11-03 06:18:32.390	2017-11-03 06:18:32.390	2017-11-03 06:18:32.810
10	QID1542	SID194	Completed	2017-11-03 08:02:17.863	2017-11-03 08:02:17.910	2017-11-03 08:02:17.910	2017-11-03 08:02:18.223

total_elap...	label	error_id	database_id	command	resource_class
8078	NULL	NULL	9	CREATE TABLE myTable ( id int NOT NULL, ...	NULL
4379	NULL	NULL	9	CREATE INDEX IX_myTable ON myTAble (zipCode)	smallrc
2999	NULL	NULL	9	DECLARE @edition sysname; SET @edition = cast(SERV...	NULL
2265	NULL	NULL	9	DBCC SHOW_STATISTICS (dbo.DimDate2,'DatelD')	NULL
2125	NULL	NULL	9	SELECT (SERVERPROPERTY('N'EDITION'))	NULL
750	NULL	9c9deb58-aad...	9	status not in ('Completed','Failed','Cancelled') AND	NULL
687	NULL	NULL	9	SELECT sm.[name] AS [schema_name], tb.[name] A...	NULL
453	NULL	NULL	9	-- Other Active Connections SELECT * FROM sys.dm_pd...	NULL
421	NULL	NULL	9	DECLARE @edition sysname; SET @edition = cast(SERV...	NULL
359	NULL	NULL	9	CREATE USER viewuser FOR LOGIN viewlogin; GRANT...	NULL

# Query Tuning Point Exploration (Request Steps and SQL Request)

```
SELECT * FROM sys.dm_pdw_request_steps WHERE request_id = 'QID1592' ORDER BY step_index;
```

	request_id	step_index	operation_type	distribution_type	location_type	status	error_id	start_time	end_time
1	QID1592	0	ReturnOperation	AllDistributions	Compute	Complete	NULL	2017-11-03 08:17:36.883	2017-11-03 08:17:36.960

total_elapsed_time	row_count	command
78	-1	SELECT [T1_1].[Dateld] AS [Dateld], [T1_1].[CalendarQuarter] AS [CalendarQuarter],...

```
SELECT * FROM sys.dm_pdw_sql_requests WHERE request_id = 'QID1592' AND step_index = 0;
```

request_id	step_index	pdw_node_id	distribution_id	status	error_id	start_time	end_time	total_elapsed_time	row_count	spid	command	
1	QID1592	0	1	1	Complete	NULL	2017-11-03 08:17:36.947	2017-11-03 08:17:36.947	0	-1	2540	SELECT [T1_1].[Dateld] AS [Dateld], [T1_1].[CalendarQuarter] AS [CalendarQuarter], [T1...
2	QID1592	0	1	2	Complete	NULL	2017-11-03 08:17:36.947	2017-11-03 08:17:36.947	0	-1	2632	SELECT [T1_1].[Dateld] AS [Dateld], [T1_1].[CalendarQuarter] AS [CalendarQuarter], [T1...
3	QID1592	0	1	3	Complete	NULL	2017-11-03 08:17:36.947	2017-11-03 08:17:36.947	0	-1	2539	SELECT [T1_1].[Dateld] AS [Dateld], [T1_1].[CalendarQuarter] AS [CalendarQuarter], [T1...
4	QID1592	0	1	4	Complete	NULL	2017-11-03 08:17:36.947	2017-11-03 08:17:36.947	0	-1	2618	SELECT [T1_1].[Dateld] AS [Dateld], [T1_1].[CalendarQuarter] AS [CalendarQuarter], [T1...
5	QID1592	0	1	5	Complete	NULL	2017-11-03 08:17:36.947	2017-11-03 08:17:36.947	0	-1	2536	SELECT [T1_1].[Dateld] AS [Dateld], [T1_1].[CalendarQuarter] AS [CalendarQuarter], [T1...
6	QID1592	0	1	6	Complete	NULL	2017-11-03 08:17:36.947	2017-11-03 08:17:36.947	0	-1	2627	SELECT [T1_1].[Dateld] AS [Dateld], [T1_1].[CalendarQuarter] AS [CalendarQuarter], [T1...
7	QID1592	0	1	7	Complete	NULL	2017-11-03 08:17:36.947	2017-11-03 08:17:36.947	0	-1	2616	SELECT [T1_1].[Dateld] AS [Dateld], [T1_1].[CalendarQuarter] AS [CalendarQuarter], [T1...
8	QID1592	0	1	8	Complete	NULL	2017-11-03 08:17:36.947	2017-11-03 08:17:36.947	0	-1	2615	SELECT [T1_1].[Dateld] AS [Dateld], [T1_1].[CalendarQuarter] AS [CalendarQuarter], [T1...
9	QID1592	0	1	9	Complete	NULL	2017-11-03 08:17:36.947	2017-11-03 08:17:36.947	0	-1	2591	SELECT [T1_1].[Dateld] AS [Dateld], [T1_1].[CalendarQuarter] AS [CalendarQuarter], [T1...
10	QID1592	0	1	10	Complete	NULL	2017-11-03 08:17:36.947	2017-11-03 08:17:36.947	0	-1	2624	SELECT [T1_1].[Dateld] AS [Dateld], [T1_1].[CalendarQuarter] AS [CalendarQuarter], [T1...
11	QID1592	0	1	11	Complete	NULL	2017-11-03 08:17:36.947	2017-11-03 08:17:36.947	0	-1	2458	SELECT [T1_1].[Dateld] AS [Dateld], [T1_1].[CalendarQuarter] AS [CalendarQuarter], [T1...
12	QID1592	0	1	12	Complete	NULL	2017-11-03 08:17:36.947	2017-11-03 08:17:36.947	0	-1	2629	SELECT [T1_1].[Dateld] AS [Dateld], [T1_1].[CalendarQuarter] AS [CalendarQuarter], [T1...
13	QID1592	0	1	13	Complete	NULL	2017-11-03 08:17:36.947	2017-11-03 08:17:36.947	0	-1	2538	SELECT [T1_1].[Dateld] AS [Dateld], [T1_1].[CalendarQuarter] AS [CalendarQuarter], [T1...
14	QID1592	0	1	14	Complete	NULL	2017-11-03 08:17:36.947	2017-11-03 08:17:36.947	0	-1	2502	SELECT [T1_1].[Dateld] AS [Dateld], [T1_1].[CalendarQuarter] AS [CalendarQuarter], [T1...
15	QID1592	0	1	15	Complete	NULL	2017-11-03 08:17:36.947	2017-11-03 08:17:36.947	0	-1	2554	SELECT [T1_1].[Dateld] AS [Dateld], [T1_1].[CalendarQuarter] AS [CalendarQuarter], [T1...
16	QID1592	0	1	16	Complete	NULL	2017-11-03 08:17:36.947	2017-11-03 08:17:36.947	0	-1	2563	SELECT [T1_1].[Dateld] AS [Dateld], [T1_1].[CalendarQuarter] AS [CalendarQuarter], [T1...
17	QID1592	0	1	17	Complete	NULL	2017-11-03 08:17:36.947	2017-11-03 08:17:36.947	0	-1	2619	SELECT [T1_1].[Dateld] AS [Dateld], [T1_1].[CalendarQuarter] AS [CalendarQuarter], [T1...
18	QID1592	0	1	18	Complete	NULL	2017-11-03 08:17:36.947	2017-11-03 08:17:36.947	0	-1	2427	SELECT [T1_1].[Dateld] AS [Dateld], [T1_1].[CalendarQuarter] AS [CalendarQuarter], [T1...
19	QID1592	0	1	19	Complete	NULL	2017-11-03 08:17:36.947	2017-11-03 08:17:36.947	0	-1	2557	SELECT [T1_1].[Dateld] AS [Dateld], [T1_1].[CalendarQuarter] AS [CalendarQuarter], [T1...
20	QID1592	0	1	20	Complete	NULL	2017-11-03 08:17:36.947	2017-11-03 08:17:36.947	0	-1	2621	SELECT [T1_1].[Dateld] AS [Dateld], [T1_1].[CalendarQuarter] AS [CalendarQuarter], [T1...

# Query Plan

<https://www.sentryone.com/plan-explorer>

SentryOne Plan Explorer

File View Tools Window Help

Edit Connection Get Estimated Plan Get Actual Plan Stop Show Estimated Plan

Start Page Plan1 - yongguin.dlt...yongsynapse (yongguin)\* X

Command Text Results

Statement	Est Cos...	Compile T...	Duration	CPU	Est CPU Co...	Reads	Writes	Est IO Co...	Est Rows	Actual Rows
select EmployeeKey, count(*) from FactSalesQuotaCCI where salesQuotaKey = 97 group by EmployeeKey	100.0%	0						100.0%	17	

Text Data

```
select EmployeeKey, count(*) from FactSalesQuotaCCI where salesQuotaKey = 97 group by EmployeeKey
```

Text Data Plan XML Plan/Query Info

Plan Diagram

```
graph LR; SELECT[SELECT] --> Project1[Project]; Project1 --> Project2[Project]; Project2 --> GAgg1[GAgg]; GAgg1 --> Shuffle[Shuffle Move]; Shuffle -- 100.0% --> GAgg2[GAgg]; GAgg2 --> Filter[Filter]; Filter -- 378,199 --> Get[Get];
```

# Query Store Enable

Object Explorer

SQLQuery1.sql - yo...pse (yongkim (246))\*

```
ALTER DATABASE yongsynapse  
SET QUERY_STORE = ON
```

100 %

Messages

Commands completed successfully.

Regressed Queries..adventureWorks2012] x

TOP 25 REGRESSED QUERIES DURING THE LAST HOUR FOR DATABASE ADVENTUREWORKS2012

query id	query sql	Duration	regr perc recent	avg duration recent	avg duration hist	exec count recent	exec count hist	num plans
1	31	SELECT	0	26002	26002	1	1	1
2	29	(@_mspa	0	33652.15	33652.15	34	34	1
3	28	(@_mspa	0	14667.33	14667.33	3	3	1
4	27	(@_mspa	0	6000	6000	1	1	1
5	26	(@_msparam_0 nvarchar(4000), @_msparam_1 nvarc...	0	17992	17992	1	1	1
6	25	SUPERVISOR	0	20003	20003	1	1	1

Plan Summary For Query 31

plan id	exec count	min duration	max duration	avg duration	std dev duration	last duration	first exec time	last exec time	plan forced
1	31	1	26002	26002	0	26002	2014-10-09 21:27:15.6800000 +0:00	2014-10-09 21:27:15.6800000 +0:00	No

Plan 31 [Not Forced]

Query 1: Query cost (relative to the batch): 100%

```
SELECT TOP (10) WITH TIES pp.FirstName, pp.LastName, e.JobTitle, e.Gender, r.Rate FROM Person.Person AS pp INNER JOIN HumanResou...
```

100 %

Results

query_text_id	query_sql_text	plan_id	query_id	query_text_id	context_settings_id	object_id	batch_sql_handle	query_hash	is_internal
1	SELECT v.name AS [Name], SCHEMA_NAME(v.schema_id) ...	1	1	1	1	0	NULL	0x17A6013BAECEDEAF	0
2	SELECT t.name AS [Name], SCHEMA_NAME(t.schema_id) ...	2	2	2	1	0	NULL	0xE0B3ED43291AF5D3	0

# Grafana

Grafana Support



All dashboards » Synapse



Synapse by fg



Synapse dashboard from <https://github.com/matrix-org/synapse/blob/master/contrib/grafana/synapse.json>

Last updated: a year ago



Overview

Revisions

Reviews

# Window functions

## OVER clause

Defines a window or specified set of rows within a query result set

Computes a value for each row in the window

## Aggregate functions

COUNT, MAX, AVG, SUM, APPROX\_COUNT\_DISTINCT, MIN, STDEV, STDEVP, STRING\_AGG, VAR, VARP, GROUPING, GROUPING\_ID, COUNT\_BIG, CHECKSUM\_AGG

## Ranking functions

RANK, NTILE, DENSE\_RANK, ROW\_NUMBER

## Analytical functions

LAG, LEAD, FIRST\_VALUE, LAST\_VALUE, CUME\_DIST, PERCENTILE\_CONT, PERCENTILE\_DISC, PERCENT\_RANK

## ROWS | RANGE

PRECEDING, UNBOUNDED PRECEDING, CURRENT ROW, BETWEEN, FOLLOWING, UNBOUNDED FOLLOWING

`SELECT`

```
ROW_NUMBER() OVER(PARTITION BY PostalCode ORDER BY SalesYTD DESC) AS "Row Number",
LastName,
SalesYTD,
PostalCode
FROM Sales
WHERE SalesYTD <> 0
ORDER BY PostalCode;
```

Row Number	LastName	SalesYTD	PostalCode
1	Mitchell	4251368.5497	98027
2	Blythe	3763178.1787	98027
3	Carson	3189418.3662	98027
4	Reiter	2315185.611	98027
5	Vargas	1453719.4653	98027
6	Ansman-Wolfe	1352577.1325	98027
1	Pak	4116870.2277	98055
2	Varkey Chudukaktil	3121616.3202	98055
3	Saraiva	2604540.7172	98055
4	Ito	2458535.6169	98055
5	Valdez	1827066.7118	98055
6	Mensa-Annan	1576562.1966	98055
7	Campbell	1573012.9383	98055
8	Tsoflias	1421810.9242	98055

# Window Functions (continued)

## Analytical functions

LAG, LEAD, FIRST\_VALUE, LAST\_VALUE, CUME\_DIST, PERCENTILE\_CONT, PERCENTILE\_DISC, PERCENT\_RANK

-- PERCENTILE\_CONT, PERCENTILE\_DISC

```
SELECT DISTINCT Name AS DepartmentName
,PERCENTILE_CONT(0.5) WITHIN GROUP (ORDER BY ph.Rate)
    OVER (PARTITION BY Name) AS MedianCont
,PERCENTILE_DISC(0.5) WITHIN GROUP (ORDER BY ph.Rate)
    OVER (PARTITION BY Name) AS MedianDisc
FROM HumanResources.Department AS d
INNER JOIN HumanResources.EmployeeDepartmentHistory AS dh
    ON dh.DepartmentID = d.DepartmentID
INNER JOIN HumanResources.EmployeePayHistory AS ph
    ON ph.BusinessEntityID = dh.BusinessEntityID
WHERE dh.EndDate IS NULL;
```

DepartmentName	MedianCont	MedianDisc
Document Control	16.8269	16.8269
Engineering	34.375	32.6923
Executive	54.32695	48.5577
Human Resources	17.427850	16.5865

--LAG Function

```
SELECT BusinessEntityID,
YEAR(QuotaDate) AS SalesYear,
SalesQuota AS CurrentQuota,
LAG(SalesQuota, 1,0) OVER (ORDER BY YEAR(QuotaDate)) AS PreviousQuota
FROM Sales.SalesPersonQuotaHistory
WHERE BusinessEntityID = 275 and YEAR(QuotaDate) IN ('2005','2006');
```

BusinessEntityID	SalesYear	CurrentQuota	PreviousQuota
275	2005	367000.00	0.00
275	2005	556000.00	367000.00
275	2006	502000.00	556000.00
275	2006	550000.00	502000.00
275	2006	1429000.00	550000.00
275	2006	1324000.00	1429000.00

# Window Functions (continued)

## ROWS | RANGE

PRECEDING, UNBOUNDING PRECEDING, CURRENT ROW, BETWEEN, FOLLOWING, UNBOUNDED FOLLOWING

```
-- First_Value  
  
SELECT JobTitle, LastName, VacationHours AS VacHours,  
FIRST_VALUE(LastName) OVER (PARTITION BY JobTitle  
ORDER BY VacationHours ASC ROWS UNBOUNDED PRECEDING ) AS FewestVacHours  
FROM HumanResources.Employee AS e  
INNER JOIN Person.Person AS p  
ON e.BusinessEntityID = p.BusinessEntityID  
ORDER BY JobTitle;
```



JobTitle	FewestVacHours	LastName	VacHours
<hr/>			
<hr/>			
Accountant	Moreland	58	Moreland
Accountant	Seamans	59	Moreland
Accounts Manager	Liu	57	Liu
Accounts Payable Specialist	Tomic	63	Tomic
Accounts Payable Specialist	Sheperdigian	64	Tomic
Accounts Receivable Specialist	Poe	60	Poe
Accounts Receivable Specialist	Spoon	61	Poe
Accounts Receivable Specialist	Walton	62	Poe

# Group by options

## Group by with rollup

Creates a group for each combination of column expressions.

Rolls up the results into subtotals and grand totals

Calculate the aggregates of hierarchical data

-- GROUP BY ROLLUP Example --

```
SELECT Country,
Region,
SUM(Sales) AS TotalSales
FROM Sales
GROUP BY ROLLUP (Country, Region);
```

-- Results --

## Grouping sets

Combine multiple GROUP BY clauses into one GROUP BY CLAUSE.

Equivalent of UNION ALL of specified groups.

-- GROUP BY SETS Example --

```
SELECT Country,
SUM(Sales) AS TotalSales
FROM Sales
GROUP BY GROUPING SETS ( Country, () );
```



Country	Region	TotalSales
Canada	Alberta	100
Canada	British Columbia	500
Canada	NULL	600
United States	Montana	100
United States	NULL	100
NULL	NULL	700

# JSON data support – read JSON data

## Overview

Read JSON data stored in a string column with the following:

- **ISJSON** – verify if text is valid JSON
- **JSON\_VALUE** – extract a scalar value from a JSON string
- **JSON\_QUERY** – extract a JSON object or array from a JSON string

## Benefits

Ability to get standard columns as well as JSON column

Perform aggregation and filter on JSON values

-- Return all rows with valid JSON data

```
SELECT CustomerId, OrderDetails
FROM CustomerOrders
WHERE ISJSON(OrderDetails) > 0;
```

CustomerId	OrderDetails
101	N'[{ StoreId": "AW73565", "Order": { "Number": "SO43659", "Date": "2011-05-31T00:00:00" }, "Item": { "Price": 2024.40, "Quantity": 1 }}]'

-- Extract values from JSON string

```
SELECT CustomerId,
Country,
JSON_VALUE(OrderDetails,'$.StoreId') AS StoreId,
JSON_QUERY(OrderDetails,'$.Item') AS ItemDetails
FROM CustomerOrders;
```

CustomerId	Country	StoreId	ItemDetails
101	Bahrain	AW73565	{ "Price": 2024.40, "Quantity": 1 }

# Approximate execution

## HyperLogLog accuracy

Will return a result with a 2% accuracy of true cardinality on average.

e.g. COUNT (DISTINCT) returns 1,000,000, HyperLogLog will return a value in the range of 999,736 to 1,016,234.

## APPROX\_COUNT\_DISTINCT

Returns the approximate number of unique non-null values in a group.

## Use Case: Approximating web usage trend behavior

-- Syntax

```
APPROX_COUNT_DISTINCT( expression )
```

-- The approximate number of different order keys by order status from the orders table.

```
SELECT O_OrderStatus, APPROX_COUNT_DISTINCT(O_OrderKey) AS Approx_Distinct_OrderKey  
FROM dbo.Orders  
GROUP BY O_OrderStatus  
ORDER BY O_OrderStatus;
```

# Approximate execution

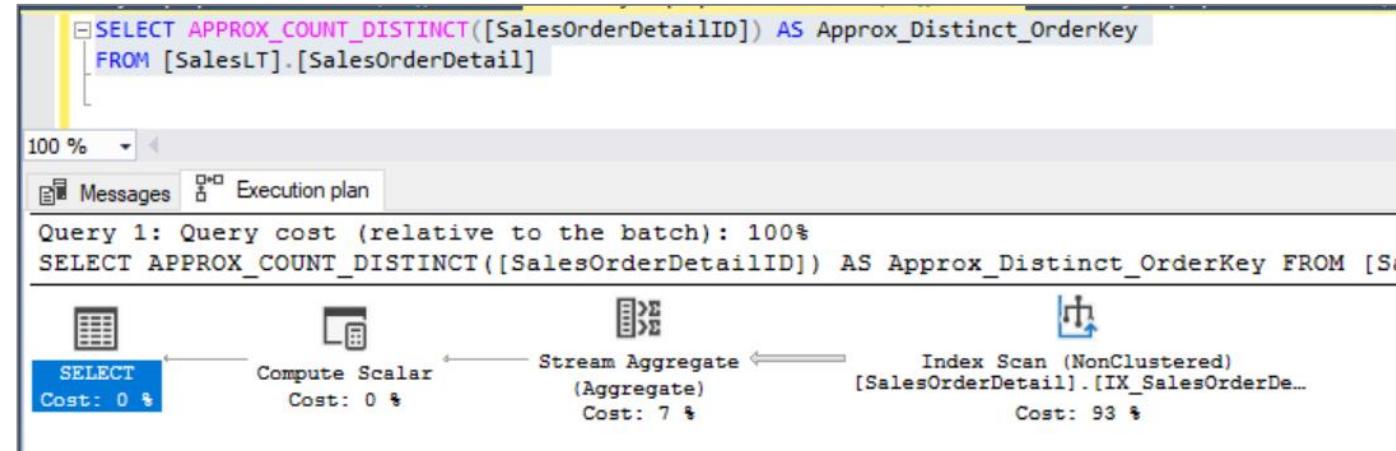
## APPROX\_COUNT\_DISTINCT

```
SELECT APPROX_COUNT_DISTINCT([SalesOrderDetailID]) AS Approx_Distinct_OrderKey
FROM [SalesLT].[SalesOrderDetail]
```

100 %

Results Messages

Approx_Distinct_OrderKey
540



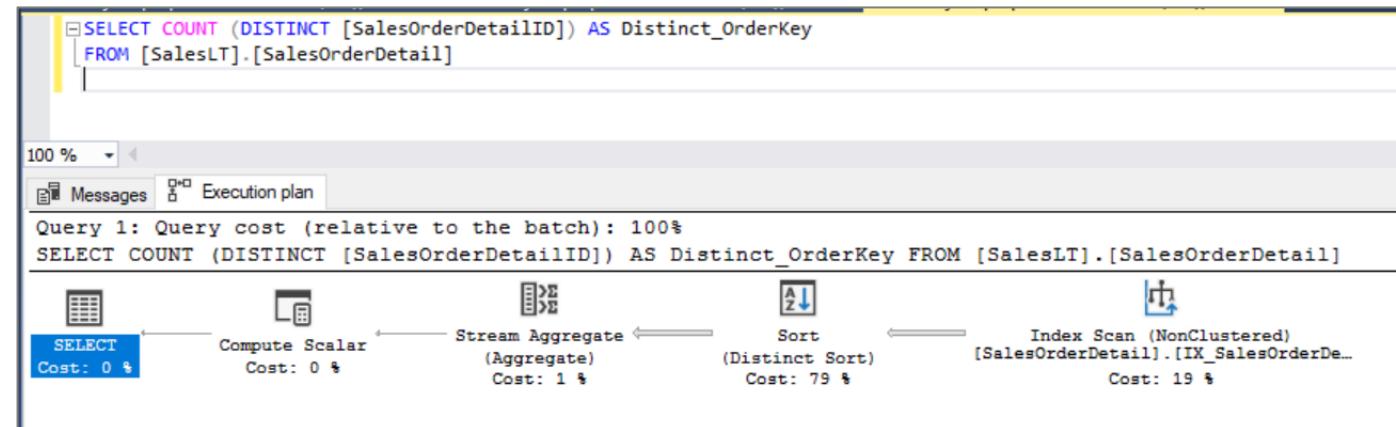
## COUNT DISTINCT

```
SELECT COUNT (DISTINCT [SalesOrderDetailID]) AS Distinct_OrderKey
FROM [SalesLT].[SalesOrderDetail]
```

100 %

Results Messages

Distinct_OrderKey
542



# Snapshot isolation

## Overview

Specifies that statements cannot read data that has been modified but not committed by other transactions.

This prevents dirty reads.

```
ALTER DATABASE MyDatabase  
SET ALLOW_SNAPSHOT_ISOLATION ON
```

```
ALTER DATABASE MyDatabase SET  
READ_COMMITTED_SNAPSHOT ON
```

## Isolation level

- READ COMMITTED
- REPEATABLE READ
- SNAPSHOT
- READ UNCOMMITTED
- SERIALIZABLE

## **READ\_COMMITTED\_SNAPSHOT**

**OFF** (Default) – Uses shared locks to prevent other transactions from modifying rows while running a read operation

**ON** – Uses row versioning to present each statement with a transactionally consistent snapshot of the data as it existed at the start of the statement. Locks are not used to protect the data from updates.

# JSON data support – insert JSON data

## Overview

The JSON format enables representation of complex or hierarchical data structures in tables.

JSON data is stored using standard NVARCHAR table columns.

## Benefits

Transform arrays of JSON objects into table format

Performance optimization using clustered columnstore indexes and memory optimized tables

```
-- Create Table with column for JSON string
CREATE TABLE CustomerOrders
(
    CustomerId BIGINT NOT NULL,
    Country NVARCHAR(150) NOT NULL,
    OrderDetails NVARCHAR(3000) NOT NULL -- NVARCHAR column for JSON
) WITH (DISTRIBUTION = ROUND_ROBIN)

-- Populate table with semi-structured data
INSERT INTO CustomerOrders
VALUES
( 101, -- CustomerId
'Bahrain', -- Country
N'[{ "StoreId": "AW73565",
    "Order": { "Number": "SO43659",
        "Date": "2011-05-31T00:00:00"
    },
    "Item": { "Price": 2024.40, "Quantity": 1 }
}]' -- OrderDetails
)
```

# JSON data support – modify and operate on JSON data

## Overview

Use standard table columns and values from JSON text in the same analytical query.

Modify JSON data with the following:

- **JSON\_MODIFY** – modifies a value in a JSON string
- **OPENJSON** – convert JSON collection to a set of rows and columns

## Benefits

Flexibility to update JSON string using T-SQL

Convert hierarchical data into flat tabular structure

```
-- Modify Item Quantity value
UPDATE CustomerOrders SET OrderDetails =
JSON_MODIFY(OrderDetails, '$.OrderDetails.Item.Quantity', 2)
```

### OrderDetails

```
N'[{"StoreId": "AW73565", "Order": { "Number": "SO43659",
"Date": "2011-05-31T00:00:00" }, "Item": { "Price": 2024.40, "Quantity": 2}}]'
```

```
-- Convert JSON collection to rows and columns
SELECT CustomerId,
StoreId,
OrderDetails.OrderDate,
OrderDetails.OrderPrice
FROM CustomerOrders
CROSS APPLY OPENJSON (CustomerOrders.OrderDetails)
WITH ( StoreId    VARCHAR(50) '$.StoreId',
OrderNumber  VARCHAR(100) '$.Order.Date',
OrderDate    DATETIME   '$.Order.Date',
OrderPrice   DECIMAL    '$.Item.Price',
OrderQuantity INT       '$.Item.Quantity'
) AS OrderDetails
```

CustomerId	StoreId	OrderDate	OrderPrice
101	AW73565	2011-05-31T00:00:00	2024.40

# Stored Procedures

## Overview

It is a group of one or more SQL statements or a reference to a Microsoft .NET Framework common language runtime (CLR) method.

Promotes flexibility and modularity.

Supports parameters and nesting.

## Benefits

Reduced server/client network traffic, improved performance

Stronger security

Easy maintenance

```
CREATE PROCEDURE HumanResources.uspGetAllEmployees
AS
    SET NOCOUNT ON;
    SELECT LastName, FirstName, JobTitle, Department
    FROM HumanResources.vEmployeeDepartment;
GO

-- Execute a stored procedures
EXECUTE HumanResources.uspGetAllEmployees;
GO
-- Or
EXEC HumanResources.uspGetAllEmployees;
GO
-- Or, if this procedure is the first statement within a batch:
HumanResources.uspGetAllEmployees;
```

# Materialized views

## Overview

A materialized view pre-computes, stores, and maintains its data like a table.

Materialized views are automatically updated when data in underlying tables are changed. This is a synchronous operation that occurs as soon as the data is changed.

The auto caching functionality allows Azure Synapse Analytics Query Optimizer to consider using indexed view even if the view is not referenced in the query.

Supported aggregations: MAX, MIN, AVG, COUNT, COUNT\_BIG, SUM, VAR, STDEV

## Benefits

Automatic and synchronous data refresh with data changes in base tables. No user action is required.

High availability and resiliency as regular tables

```
-- Create indexed view
CREATE MATERIALIZED VIEW Sales.vw_Orders
WITH
(
    DISTRIBUTION = ROUND_ROBIN |
    HASH(ProductID)
)
AS
    SELECT SUM(UnitPrice*OrderQty) AS Revenue,
        OrderDate,
        ProductID,
        COUNT_BIG(*) AS OrderCount
    FROM Sales.SalesOrderDetail
    GROUP BY OrderDate, ProductID;
GO

-- Disable index view and put it in suspended mode
ALTER INDEX ALL ON Sales.vw_Orders DISABLE;

-- Re-enable index view by rebuilding it
ALTER INDEX ALL ON Sales.vw_Orders REBUILD;
```

# Materialized views - example

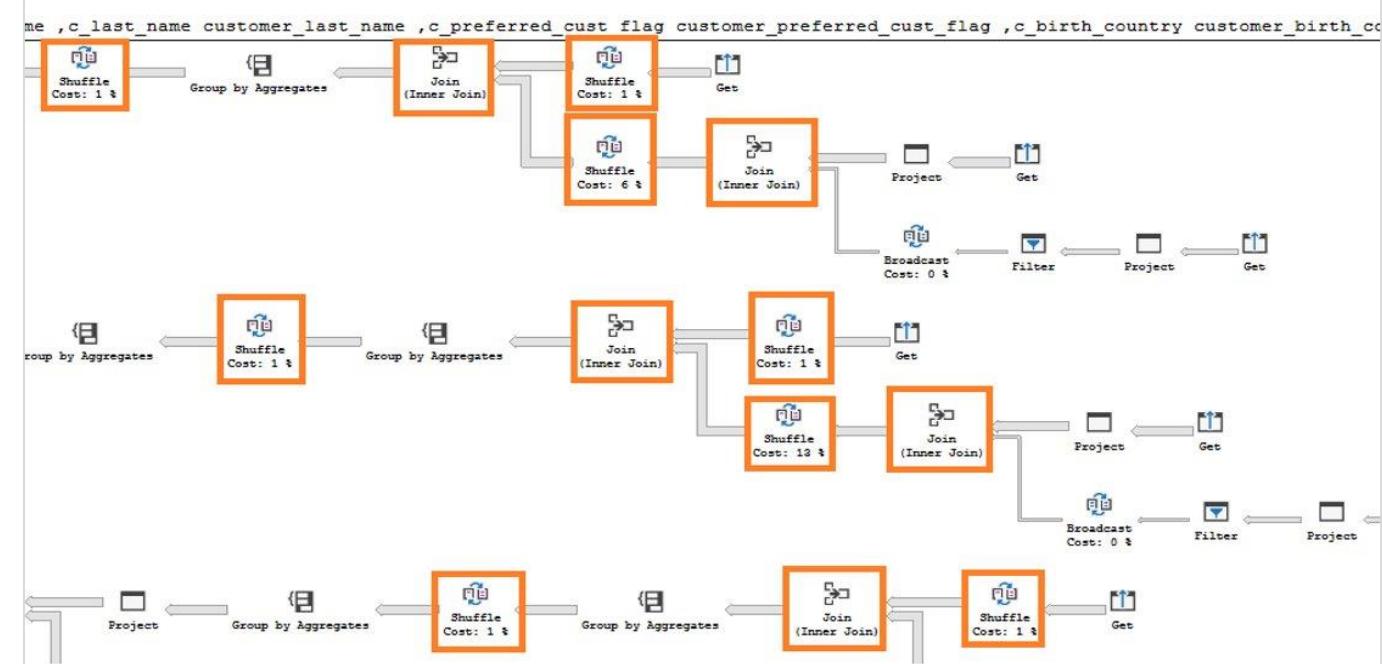
In this example, a query to get the year total sales per customer is shown to have a lot of data shuffles and joins that contribute to slow performance:

No relevant indexed views created on the data warehouse

```
-- Get year total sales per customer
(WITH year_total AS
    SELECT customer_id,
        first_name,
        last_name,
        birth_country,
        login,
        email_address,
        d_year,
        SUM(ISNULL(list_price - wholesale_cost -
            discount_amt + sales_price, 0)/2)year_total
    FROM customer cust
    JOIN catalog_sales sales ON cust.sk = sales.sk
    JOIN date_dim ON sales.sold_date = date_dim.date
    GROUP BY customer_id, first_name,
        last_name,birth_country,
        login,email_address ,d_year
)
SELECT TOP 100 ...
FROM year_total ...
WHERE ...
ORDER BY ...
```

**Execution time:** 103 seconds

Lots of data shuffles and joins needed to complete query



# Materialized views - example

Now, we add an indexed view to the data warehouse to increase the performance of the previous query. This view can be leveraged by the query even though it is not directly referenced.

Original query – get year total sales per customer

```
-- Get year total sales per customer
(WITH year_total AS
    SELECT customer_id,
           first_name,
           last_name,
           birth_country,
           login,
           email_address,
           d_year,
           SUM(ISNULL(list_price - wholesale_cost -
           discount_amt + sales_price, 0)/2)year_total
      FROM customer cust
     JOIN catalog_sales sales ON cust.sk = sales.sk
     JOIN date_dim ON sales.sold_date = date_dim.date
    GROUP BY customer_id, first_name,
             last_name,birth_country,
             login,email_address ,d_year
)
SELECT TOP 100 ...
   FROM year_total ...
  WHERE ...
 ORDER BY ...
```

Create indexed view with hash distribution on customer\_id column

```
-- Create indexed view for query
CREATE INDEXED VIEW nbViewCS WITH (DISTRIBUTION=HASH(customer_id)) AS
SELECT customer_id,
       first_name,
       last_name,
       birth_country,
       login,
       email_address,
       d_year,
       SUM(ISNULL(list_price - wholesale_cost - discount_amt +
       sales_price, 0)/2) AS year_total
  FROM customer cust
 JOIN catalog_sales sales ON cust.sk = sales.sk
 JOIN date_dim ON sales.sold_date = date_dim.date
 GROUP BY customer_id, first_name,
          last_name,birth_country,
          login, email_address, d_year
```

# Indexed (materialized) views - example

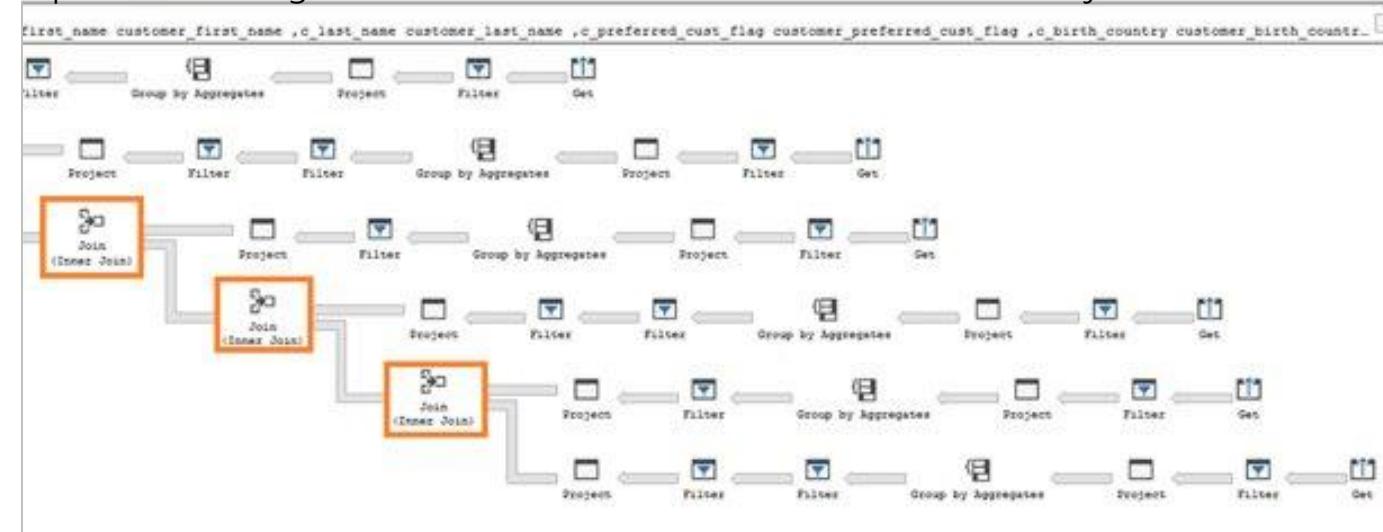
Synapse SQL (provisioned) query optimizer automatically leverages the indexed view to speed up the same query.  
 Notice that the query does not need to reference the view directly

Original query – no changes have been made to query

```
-- Get year total sales per customer
(WITH year_total AS
  SELECT customer_id,
         first_name,
         last_name,
         birth_country,
         login,
         email_address,
         d_year,
         SUM(ISNULL(list_price - wholesale_cost -
                    discount_amt + sales_price, 0)/2)year_total
    FROM customer cust
   JOIN catalog_sales sales ON cust.sk = sales.sk
   JOIN date_dim ON sales.sold_date = date_dim.date
 GROUP BY customer_id, first_name,
          last_name,birth_country,
          login,email_address ,d_year
)
SELECT TOP 100 ...
FROM year_total ...
WHERE ...
ORDER BY ...
```

**Execution time:** 6 seconds

Optimizer leverages materialized view to reduce data shuffles and joins needed



# Materialized views- Recommendations

**EXPLAIN** - provides query plan for SQL statement without running the statement; view estimated cost of the query operations.

**EXPLAIN WITH\_RECOMMENDATIONS** - provides query plan with recommendations to optimize the SQL statement performance.

```
EXPLAIN WITH_RECOMMENDATIONS
select count(*)
from (
    select distinct c_last_name, c_first_name, d_date
    from store_sales, date_dim, customer
    where store_sales.ss_sold_date_sk = date_dim.d_date_sk
    and store_sales.ss_customer_sk = customer.c_customer_sk
    and d_month_seq between 1194 and 1194+11)
except
    (select distinct c_last_name, c_first_name, d_date
    from catalog_sales, date_dim, customer
    where catalog_sales.cs_sold_date_sk = date_dim.d_date_sk
    and catalog_sales.cs_bill_customer_sk = customer.c_customer_sk and
    d_month_seq between 1194 and 1194+11)
) top_customers
```

# COPY command

## Overview

Copies data from source to destination

## Benefits

Retrieves data from all files from the folder and all its subfolders.

Supports multiple locations from the same storage account, separated by comma

Supports Azure Data Lake Storage (ADLS) Gen 2 and Azure Blob Storage.

Supports CSV, PARQUET, ORC file formats

```
COPY INTO test_1
FROM 'https://XXX.blob.core.windows.net/customerdatasets/test_1.txt'
WITH (
    FILE_TYPE = 'CSV',
    CREDENTIAL=(IDENTITY= 'Shared Access Signature',
SECRET='<Your_SAS_Token>'),
    FIELDQUOTE = """",
    FIELDTERMINATOR=';',
    ROWTERMINATOR='0XA',
    ENCODING = 'UTF8',
    DATEFORMAT = 'ymd',
    MAXERRORS = 10,
    ERRORFILE = '/errorsfolder/'--path starting from the storage container,
    IDENTITY_INSERT
)
```

```
COPY INTO test_parquet
FROM 'https://XXX.blob.core.windows.net/customerdatasets/test.parquet'
WITH (
    FILE_FORMAT = myFileFormat
    CREDENTIAL=(IDENTITY= 'Shared Access Signature',
SECRET='<Your_SAS_Token>')
)
```

# Create External Table As Select

## Overview

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 );
```

# Predict

## Overview

It provides ability to import existing machine learning models and score them within provisioned SQL. It takes ONNX (Open Neural Network Exchange) and data as inputs and generates prediction based on model.

## Benefits

1. It empowers data engineers to successfully deploy machine learning models with the familiar T-SQL interface
2. It offers seamless collaboration with data scientists
3. It generates new columns, but the number of columns and their data types depends on the type of model that was used for prediction.

### Syntax:

```
PREDICT
(
    MODEL = @model | model_literal,
    DATA = object AS <table_alias>
)
WITH ( <result_set_definition> )
<result_set_definition> ::= 
{
    { column_name
        data_type
    }
    [,...n]
}
MODEL = @model | model_literal
```

### Example:

```
DECLARE @model varbinary(max) = (SELECT Model FROM Models WHERE Id = <>);
SELECT d.*, p.Score
FROM PREDICT(MODEL = @model,
    DATA = dbo.mytable AS d) WITH (Score float) AS p;
```

# Result-set caching

## Overview

Cache the results of a query in provisioned SQL storage. This enables interactive response times for repetitive queries against tables with infrequent data changes.

The result-set cache persists even if dedicated SQL pool is paused and resumed later.

Query cache is invalidated and refreshed when underlying table data or query code changes.

Result cache is evicted regularly based on a time-aware least recently used algorithm (TLRU).

## Benefits

Enhances performance when same result is requested repetitively

Reduced load on server for repeated queries

Offers monitoring of query execution with a result cache hit or miss

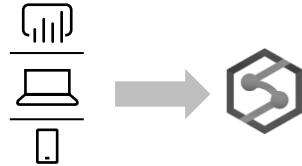
```
-- Turn on/off result-set caching for a database
-- Must be run on the MASTER database
ALTER DATABASE {database_name}
SET RESULT_SET_CACHING { ON | OFF }
```

```
-- Turn on/off result-set caching for a client session
-- Run on target Azure Synapse Analytics
SET RESULT_SET_CACHING {ON | OFF}
```

```
-- Check result-set caching setting for a database
-- Run on target Azure Synapse Analytics
SELECT is_result_set_caching_on
FROM sys.databases
WHERE name = {database_name}
```

```
-- Return all query requests with cache hits
-- Run on target data warehouse
SELECT *
FROM sys.dm_pdw_request_steps
WHERE command like '%DWResultCacheDb%'
    AND step_index = 0
```

# Result-set caching flow



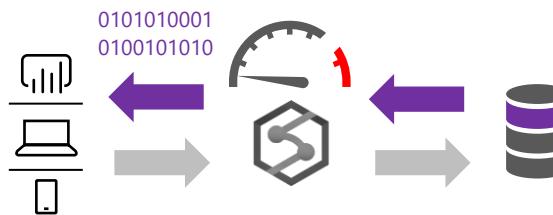
- 1 Client sends query to dedicated SQL pool



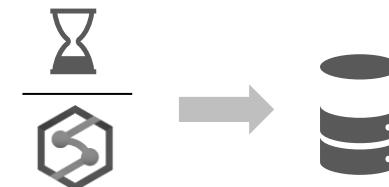
- 2 Query is processed using compute nodes which pull data from remote storage, process query and output back to client app



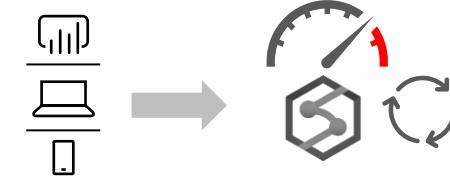
Query results are cached in remote storage so subsequent requests can be served immediately



- 3 Subsequent executions for the same query bypass compute nodes and can be fetched instantly from persistent cache in remote storage



- 4 Remote storage cache is evicted regularly based on time, cache usage, and any modifications to underlying table data.



- 5 Cache will need to be regenerated if query results have been evicted from cache

# Continuous integration and delivery (CI/CD)

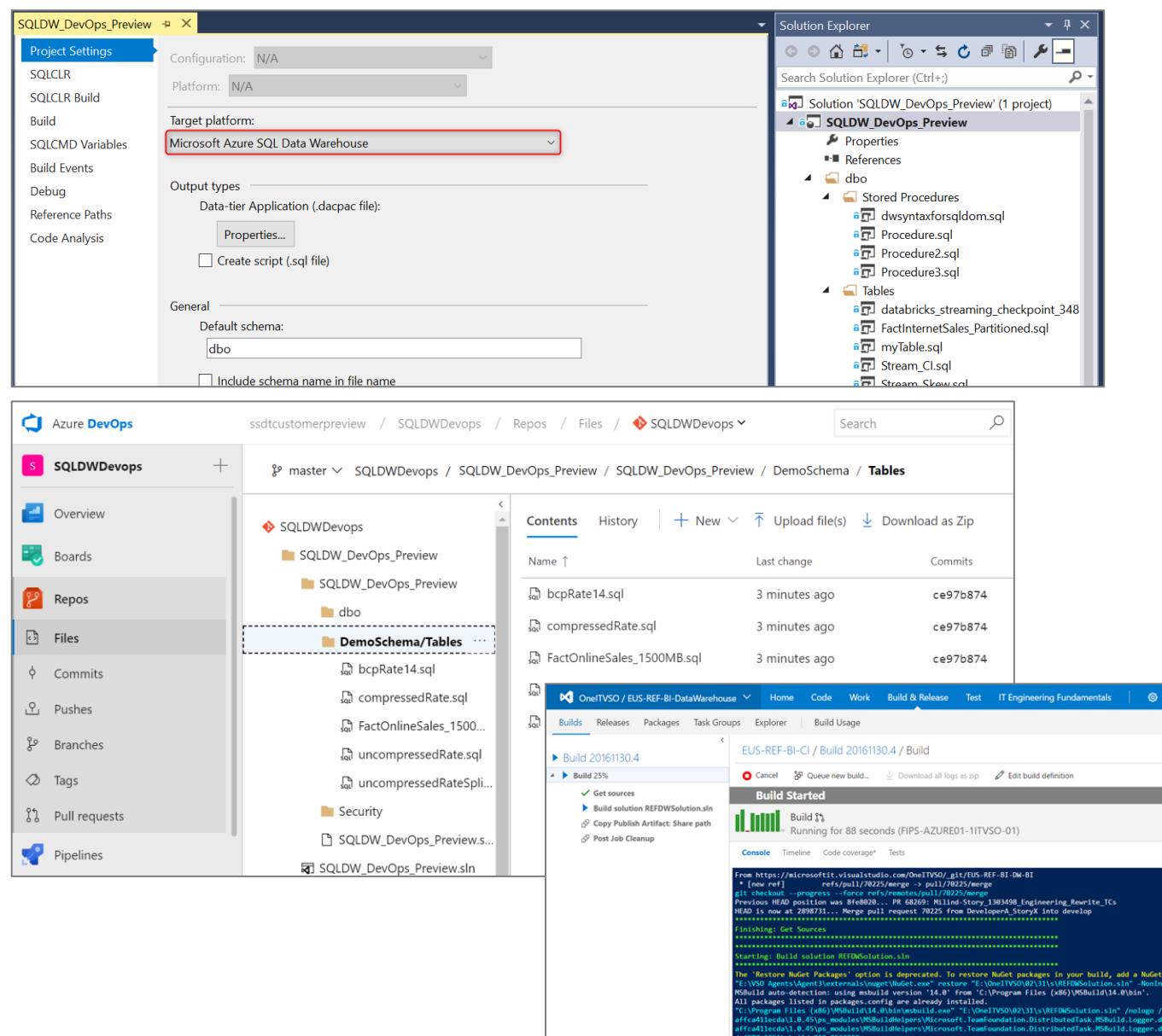
## Overview

Database project support in SQL Server Data Tools (SSDT) allows teams of developers to collaborate over a version-controlled Azure Synapse Analytics, and track, deploy and test schema changes.

## Benefits

Database project support includes first-class integration with Azure DevOps. This adds support for:

- Azure Pipelines** to run CI/CD workflows for any platform (Linux, macOS, and Windows)
- Azure Repos** to store project files in source control
- Azure Test Plans** to run automated check-in tests to verify schema updates and modifications
- Growing ecosystem of third-party integrations that can be used to complement existing workflows (Timetracker, Microsoft Teams, Slack, Jenkins, etc.)



# Azure Advisor recommendations

## Suboptimal Table Distribution

Reduce data movement by replicating tables

## Data Skew

Choose new hash-distribution key

Slowest distribution limits performance

## Cache Misses

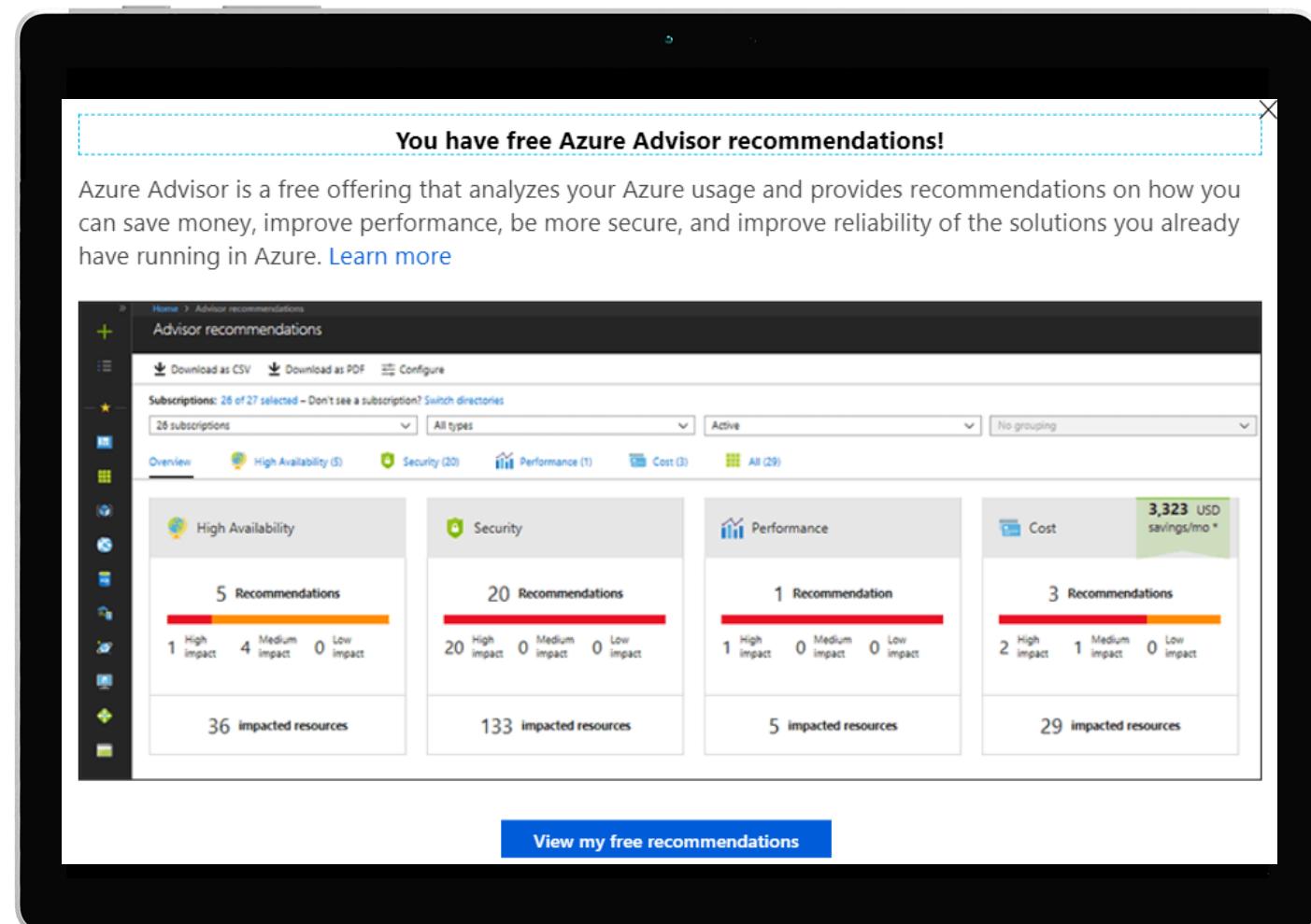
Provision additional capacity

## Tempdb Contention

Scale or update user resource class

## Suboptimal Plan Selection

Create or update table statistics



# Maintenance windows

## Overview

Choose a time window for your upgrades.

Select a primary and secondary window within a seven-day period.

Windows can be from 3 to 8 hours.

24-hour advance notification for maintenance events.

## Benefits

Ensure upgrades happen on your schedule.

Predictable planning for long-running jobs.

Stay informed of start and end of maintenance.

The screenshot shows the 'Maintenance Schedule (preview)' page in the Azure portal. At the top, there's a navigation bar with 'Home', 'maintenanceexamples', and 'Maintenance Schedule (preview)'. Below the navigation is a toolbar with 'Save', 'Discard', and 'Feedback' buttons. A sidebar on the left contains various icons for different Azure services. The main content area has an information icon with a message: 'Maintenance on your data warehouse could occur once a week within one of two maintenance windows. Choose the primary and secondary windows that best suit your operational needs. If you would like to use the maintenance windows already defined, no action is required.' Below this, there's a section titled 'Choose primary window' with radio buttons for 'Saturday - Sunday' (selected) and 'Tuesday - Thursday'. The 'Primary maintenance window' section shows 'Day' set to 'Saturday', 'Start time' at '03:00 UTC', and 'Time window' at '8 hours'. The 'Secondary maintenance window' section shows 'Day' set to 'Tuesday', 'Start time' at '13:00 UTC', and 'Time window' at '8 hours'. At the bottom, there's a 'Schedule summary' section with the same details for both windows.

Primary maintenance window	Secondary maintenance window
Day <a href="#">i</a> Saturday	Day <a href="#">i</a> Tuesday
Start time <a href="#">i</a> 03:00 UTC	Start time <a href="#">i</a> 13:00 UTC
Time window <a href="#">i</a> 8 hours	Time window <a href="#">i</a> 8 hours

Schedule summary	
Primary maintenance window Saturday 03:00 UTC (8 hours)	Secondary maintenance window Tuesday 13:00 UTC (8 hours)

# Automatic statistics management

## Overview

Statistics are automatically created and maintained for dedicated SQL pool. Incoming queries are analyzed, and individual column statistics are generated on the columns that improve cardinality estimates to enhance query performance.

Statistics are automatically updated as data modifications occur in underlying tables. By default, these updates are synchronous but can be configured to be asynchronous.

Statistics are considered out of date when:

- There was a data change on an empty table
- The number of rows in the table at time of statistics creation was 500 or less, and more than 500 rows have been updated
- The number of rows in the table at time of statistics creation was more than 500, and more than  $500 + 20\%$  of rows have been updated

-- Turn on/off auto-create statistics settings

```
ALTER DATABASE {database_name}
```

```
SET AUTO_CREATE_STATISTICS { ON | OFF }
```

-- Turn on/off auto-update statistics settings

```
ALTER DATABASE {database_name}
```

```
SET AUTO_UPDATE_STATISTICS { ON | OFF }
```

-- Configure synchronous/asynchronous update

```
ALTER DATABASE {database_name}
```

```
SET AUTO_UPDATE_STATISTICS_ASYNC { ON | OFF }
```

-- Check statistics settings for a database

```
SELECT      is_auto_create_stats_on,  
            is_auto_update_stats_on,  
            is_auto_update_stats_async_on  
FROM        sys.databases
```



# Azure Synapse Analytics

## Synapse serverless SQL pool

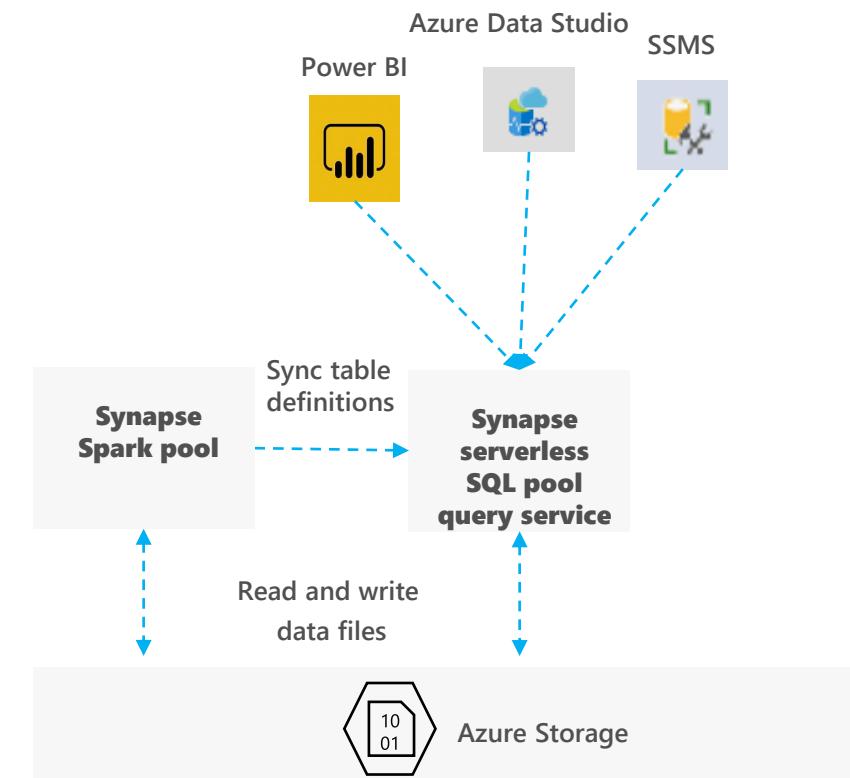
# serverless SQL pool

## Overview

An interactive query service that enables you to use standard T-SQL queries over files in Azure storage.

## Benefits

- Use SQL to work with files on Azure storage
  - Directly query files on Azure storage using T-SQL
  - Logical Data Warehouse on top of Azure storage
  - Easy data transformation of Azure storage files
- Supports any tool or library that uses T-SQL to query data
- Automatically synchronize tables from Spark
- Serverless
  - No infrastructure, no upfront cost, no resource reservation
  - Pay only for query execution (per data processed)



# Recommended usage scenarios

## Quick data exploration

- Easily explore schema and data in files on Azure storage
- Supports various file formats (Parquet, CSV, JSON)
- Direct connector to Azure storage for large BI ecosystem

## Logical Data Warehouse

- Model raw files as virtual tables and views
- Use any tool that works with SQL to analyze files
- Use enterprise-grade security model

## Easy data transformation

- Transform CSV to parquet format
- Move data between containers and accounts
- Save the results of queries on external storage

# Easily explore files on storage

The screenshot illustrates the integration of Azure Storage and Azure Synapse Analytics. On the left, the Azure portal navigation bar shows 'Microsoft Azure | Synapse Analytics > internalsandboxwe5'. The main area is divided into two panes:

- Left Pane (File Explorer):** Shows the 'Data' section with 'Storage accounts' (internalsandboxwe), 'Databases' (3), and 'Datasets' (5). The 'opendataset' folder under 'Storage accounts' contains several files, including '\_SUCCESS', 'part-00000...', 'part-00001...', 'part-00002...', and 'part-00003...'. A context menu is open over the 'New SQL script - Select TOP 100 rows' file, with options like 'New SQL script', 'New notebook', 'Copy ABFSS path', 'Manage Access...', 'Rename...', 'Download', 'Delete', and 'Properties...'.
- Right Pane (Query Editor):** Shows the 'opendataset' folder with a 'SQL script 1' tab. The script is set to 'Connect to: SQL on-demand'. The code is:

```

1 SELECT
2     TOP 100 *
3 FROM
4     OPENROWSET(
5         BULK 'https://internalsandboxwe.dfs.core.windows.net/opendataset/holidays/part-00001-bd1aba93-a85a-4909-8bf4-f79afb6c946f-c000.snappy.parquet',
6         FORMAT='PARQUET'
7     ) AS [r];

```

The results pane shows a table with columns: VENDORID, TPEPICKUPDATETIME, TPEPDROPOFFDATETIME, PASSENGERCOUNT, TRIPDISTANCE, PULOCATIONID, and DOLOCATIONID. The data is as follows:

VENDORID	TPEPICKUPDATETIME	TPEPDROPOFFDATETIME	PASSENGERCOUNT	TRIPDISTANCE	PULOCATIONID	DOLOCATIONID
VTS	2009-05-07T23:1...	2009-05-07T23:2...	1	2.94	NULL	NULL
VTS	2009-05-07T16:3...	2009-05-07T16:3...	5	0.73	NULL	NULL
VTS	2009-05-08T14:5...	2009-05-08T15:0...	3	0.55	NULL	NULL
VTS	2009-05-07T15:5...	2009-05-07T16:1...	1	2.5	NULL	NULL

A message at the bottom right indicates: '00:00:31 Query executed successfully.'

# Easily query files in various formats

## Overview

Use OPENROWSET function to access data stored in various file formats

## Benefits

Enables you to read CSV, parquet, and JSON files

Provides unified T-SQL interface for all file types

Use standard SQL language to transform and analyze returned data

- Use JSON functions to get the data from underlying files.
- Use JSON functions to get data from PARQUET nested types

```
SELECT TOP 10 *
FROM OPENROWSET(
    BULK 'https://XYZ.blob.core.windows.net/csv/taxi/*.csv',
    FORMAT = 'CSV')
WITH (
    country_code VARCHAR(4),
    country_name VARCHAR(50),
    year INT,
    population INT
) AS nyc
```

```
SELECT TOP 10 *
FROM OPENROWSET(
    BULK 'https://XYZ.blob.core.windows.net/csv/taxi/*.parquet',
    FORMAT = 'PARQUET') AS nyc
```

```
SELECT TOP 10 *
    JSON_VALUE(jsonContent, '$.countryCode') AS country_code,
    JSON_VALUE(jsonContent, '$.countryName') AS country_name,
    JSON_VALUE(jsonContent, '$.year') AS year
    JSON_VALUE(jsonContent, '$.population') AS population
FROM OPENROWSET(
    BULK 'https://XYZ.blob.core.windows.net/json/taxi/*.json',
    FORMAT='CSV',
    FIELDTERMINATOR = '0x0b',
    FIELDQUOTE = '0x0b',
    ROWTERMINATOR = '0x0b'
)
WITH ( jsonContent varchar(MAX) ) AS json_line
```

	country_code	country_name	year	population
1	LU	Luxembourg	2017	594130

# Automatic schema inference

## Overview

OPENROWSET will automatically determine columns and types of data stored in external file.

```
SELECT TOP 10 *
FROM OPENROWSET(
    BULK 'https://XYZ.blob.core.windows.net/csv/taxi/*.parquet',
    FORMAT = 'PARQUET') AS nyc
```

## Benefits

No need to up-front analyze file structure to query the file

OPENROWSET identifies columns and their types based on underlying file metadata.

Perfect solution for data exploration where schema is unknown.

The functionality is available for both parquet & CSV files.

	country_code	country_name	year	population
1	LU	Luxembourg	2017	594130

```
SELECT
    TOP 100 *
FROM
    OPENROWSET(
        BULK 'https://azuresynapsesa.dfs.core.windows.net/default/RetailData/StoreDemoGraphics.csv',
        FORMAT = 'CSV',
        PARSER_VERSION='2.0',
        HEADER_ROW = TRUE) AS [result]
```

StoreId	RatioAge60	CollegeRatio	Income	HighIncome15...	LargeHH	MinoritiesRatio	More1FullTime...	DistanceNeare...	SalesN
2	0.232864734	0.248934934	10.55320518	0.463887065	0.103953406	0.114279949	0.303585347	2.110122129	1.1428
5	0.117368032	0.32122573	10.92237097	0.535883355	0.103091585	0.053875277	0.410568032	3.801997814	0.6818

# Defined the query result schema inline

## Overview

Specify columns and types at query time.

## Benefits

Define result schema at query time in WITH clause.

No need for external format files.

Explicitly define exact return types, their sizes, and collations.

Improve performance by column elimination in parquet files.

```
SELECT TOP 10 *
FROM OPENROWSET(
    BULK 'https://XYZ.blob.core.windows.net/csv/taxi/*.csv',
    FORMAT - 'CSV')
WITH (
    country_code VARCHAR(4),
    country_name VARCHAR(50),
    year INT,
    population INT
) AS nyc
```

	country_code	country_name	year	population
1	LU	Luxembourg	2017	594130

# Customize the content parsing to fit your case

## Overview

Uses OPENROWSET function to access data from various types of CSV files.

## Benefits

Ability to read CSV files with custom format

- With or without header row
- Handle any new-line terminator (Windows or Unix style)
- Use custom field terminator and quote character
- Read UTF-8 and UTF-18 encoded files
- Use only a subset of columns by specifying column position after column types

```
SELECT *
FROM OPENROWSET(
    BULK 'https://XYZ.blob.core.windows.net/csv/population/population.csv',
    FORMAT = 'CSV',
    FIELDTERMINATOR = ',',
    ROWTERMINATOR = '\n'
)
WITH (
    [country_code] VARCHAR (5) 2,
    [country_name] VARCHAR (100) 4,
    [year] smallint 7,
    [population] bigint 9
) AS [r]
WHERE
    country_name = 'Luxembourg'
    AND year = 2017
```

Second, fourth, seventh and ninth columns are returned

	country_code	country_name	year	population
1	LU	Luxembourg	2017	594130

# Easily query multiple files, with wildcards

## Overview

Uses OPENROWSET function to access data from multiple files or folders using wildcards in path

## 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://XYZ.blob.core.windows.net/csv/taxi/year=*/month=1/*.parquet',  
    FORMAT = 'PARQUET') 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

# Query partitioned data, using the folder structure

## Overview

Uses OPENROWSET function to access data partitioned in sub-folders

## Benefits

Use filepath() function to access actual values from file paths.

Eliminate sub-folders/partitions before the query starts execution

Query Spark/Hive partitioned data sets

```
SELECT  
    r.filepath(1) AS [year]  
    ,r.filepath(2) AS [month]  
    ,COUNT_BIG(*) AS [rows]  
FROM OPENROWSET(  
    BULK 'https://XYZ.blob.core.windows.net/year=*/month=/*/*.parquet',  
    FORMAT = 'PARQUET') 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
```

year	month	rows
2017	10	9768815
2017	11	9284803
2017	12	9508276

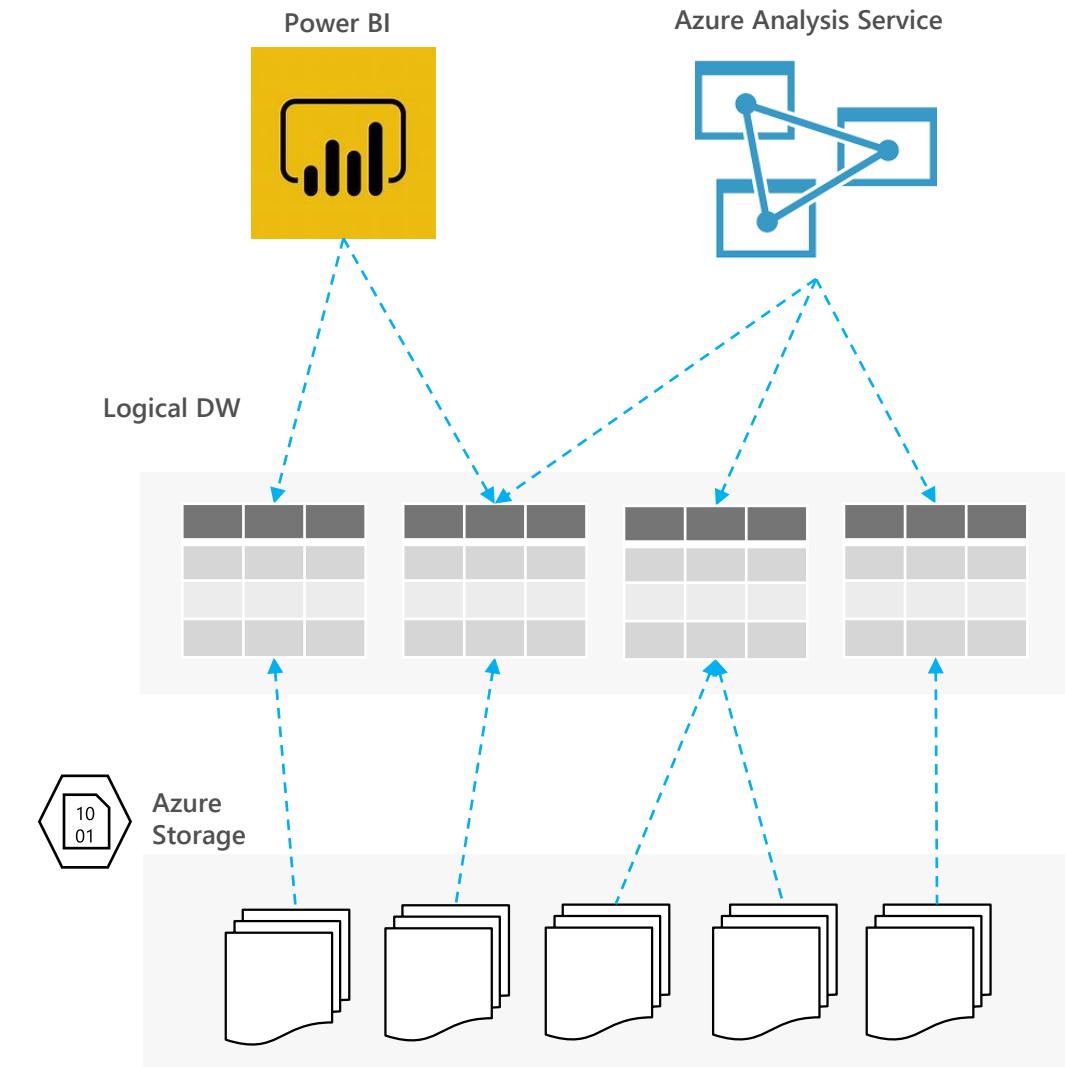
# Synapse serverless SQL pool as a logical data warehouse

## Overview

Logical relational layer on top of physical files in Azure Storage.

## Benefits

- Abstract physical storage and file formats using well understandable relational concepts such as tables and views.
- Direct connector to Azure storage for large ecosystem of BI tools
- BI tools that use SQL can work with files on storage
  - Analytic tools use external tables that represent proxy to actual files.
  - No need for custom connectors in BI tools.
- Provides complex data processing (joining and aggregation) on top of raw files.
- Apply enterprise-ready security model and access control using battle-tested SQL Server permission model on top of Azure storage files



# Logical Data Warehouse views

## Overview

serverless SQL pool logical data warehouse views are created on external files placed in customer Azure storage

## Benefits

Create SQL views on externally stored data

Access files using the view from various tools and language

Leverage rich T-SQL language to process and analyze data in external files exposed via views

Create PowerBI reports on the views created on external data

```
USE [mydbname]
GO

DROP VIEW IF EXISTS populationView
GO

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

# Creating views

Microsoft Azure | Synapse Analytics > internalsandbox... Search resources Publish all Validate all Refresh Discard all

Data + << Filter resources by name Storage accounts 1 internalsandboxwe (Primar... Databases 3 Datasets 5

```

CREATE VIEW yellow_2017 AS
Select *
FROM
OPENROWSET(
    BULK 'https://internalsandboxwe.dfs.core.windows.net/opendataset/nyctlc/yellow/puYear=2017/*/*',
    FORMAT='PARQUET'
) AS [r];

```

Microsoft Azure | Synapse Analytics > internalsandbox... Search resources Publish all Validate all Refresh Discard all

Develop + << Filter resources by name SQL scripts 3 opendataset SQL script 1 SQL script 2 SQL script 3

```

-- type your sql script here, we now have intellisense
SELECT
YEAR(tpepPickupDateTime),
passengerCount,
COUNT(*) AS cnt
FROM
yellow_2017
GROUP BY
passengerCount,
YEAR(tpepPickupDateTime)
ORDER BY
YEAR(tpepPickupDateTime),
passengerCount

```

Results Messages View Table Chart Export results

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

00:01:00 Query executed successfully.

Microsoft Azure | Synapse Analytics > internalsandbox... Search resources Publish all Validate all Refresh Discard all

Develop + << Filter resources by name SQL scripts 3 opendataset SQL script 1 SQL script 2 SQL script 3

```

-- type your sql script here, we now have intellisense
SELECT
YEAR(tpepPickupDateTime),
passengerCount,
COUNT(*) AS cnt
FROM
yellow_2017
GROUP BY
passengerCount,
YEAR(tpepPickupDateTime)
ORDER BY
YEAR(tpepPickupDateTime),
passengerCount

```

Results Messages View Table Chart Save as image

Chart type: Line  
Category column: (none)  
Legend (series) columns: Column 0, passengerCount, cnt  
Legend position: center - bottom  
Legend (series) label:

00:01:00 Query executed successfully.

# Logical Data Warehouse - tables

## Overview

Create external tables that reference external files in your serverless SQL pool logical data warehouse

## Benefits

Create external tables that reference set of files on Azure storage.

Join and transform multiple tables in the same query.

Enables you to analyze external files with the same experience that you have in classic databases.

Manage column statistics in external tables.

Manage access rights per table.

Create PowerBI reports on the views created on external data

```
USE [mydbname]
GO

DROP TABLE IF EXISTS dbo.Population
GO

CREATE EXTERNAL TABLE dbo.Population (
    country_code VARCHAR (5) COLLATE Latin1_General_BIN2,
    country_name VARCHAR (100) COLLATE Latin1_General_BIN2,
    year smallint,
    population bigint
)
WITH(
    LOCATION = '/csv/population/population-* .csv',
    DATA_SOURCE = MyAzureStorage,
    FILE_FORMAT = MyAzureCSVFormat
)
```

```
CREATE STATISTICS stat_country_name
ON dbo.Population(country_name);
```

```
SELECT
    country_name, population
FROM population
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

# Easy data transformation

## Overview

Easily perform data transformations of Azure Storage files using SQL queries

Optimize data pipeline - achieve more using serverless SQL pool

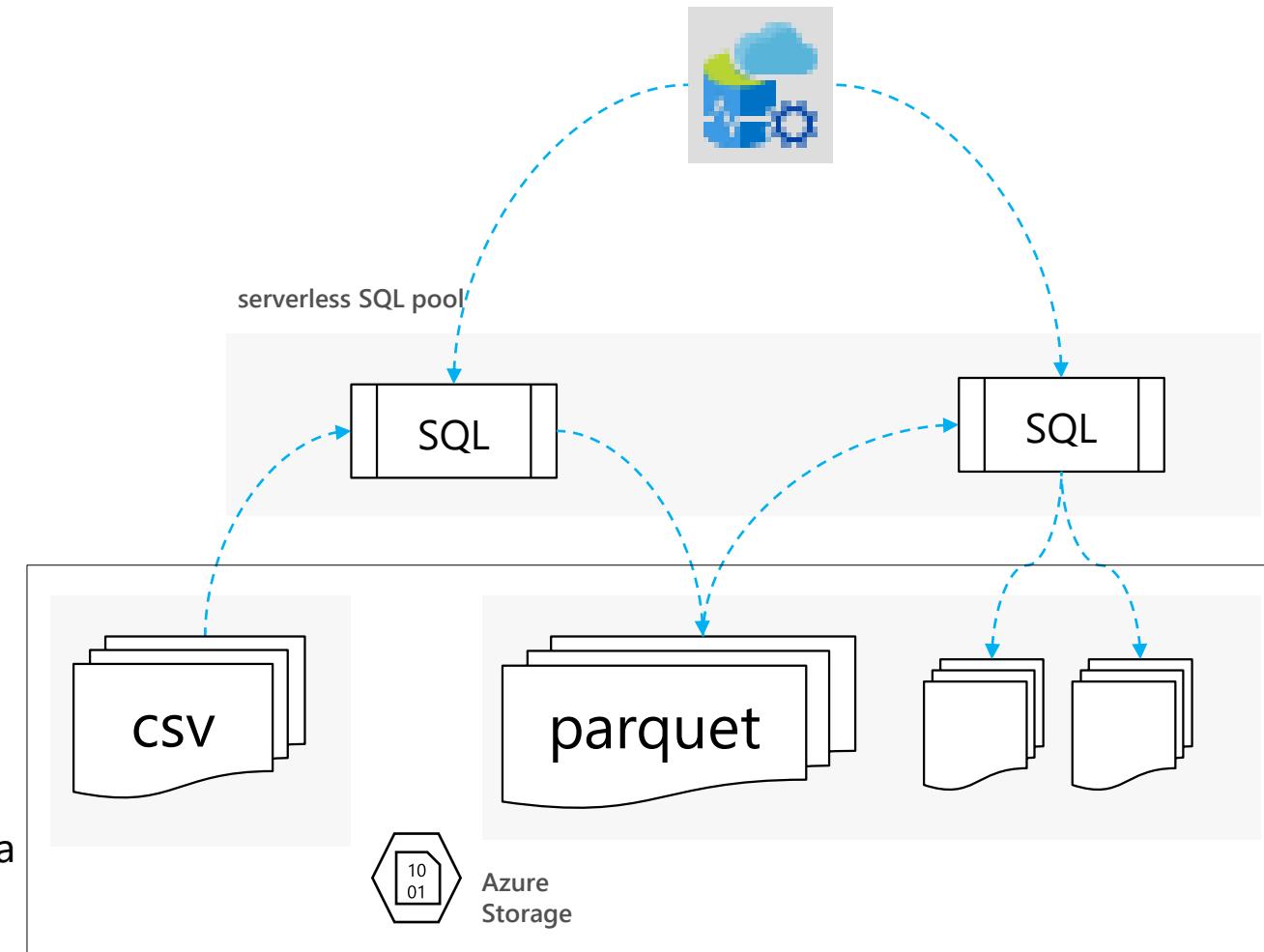
## Benefits

Single statement transformations:

- convert CSV or JSON files to Parquet
- copy files from one storage account to another
- re-partition data to new location(s)
- store results of your query on Azure Storage

SQL ETL pipelines

- Use SQL commands to transform data
- Chain SQL statement for build ETL process
- Materialize reports created on the current snapshot of data



# Easy data transformation with CETAS

## Overview

Create external tables as select (CETAS) enables you to easily transform data and store the results of query on Azure storage

## Benefits

- Select any data set and store it in parquet format.
- Pre-calculate and store results of query and store them permanently on Azure storage.
- Use saved data using external table.
- Improve performance of your reports by permanently storing the result based on current snapshot of data as parquet files.

```
-- copy CSV dataset into parquet data set
CREATE EXTERNAL TABLE parquet.Population
WITH(
    LOCATION = '/parquet/population',
    DATA_SOURCE = MyAzureStorage,
    FILE_FORMAT = MyAzureParquetFormat )
AS
SELECT *
FROM csv.Population

-- pre-create report using new parquet data-set
CREATE EXTERNAL TABLE parquet.PopulationByMonth2017
WITH(
    LOCATION = '/parquet/population/bymonth/2017',
    DATA_SOURCE = MyAzureStorage,
    FILE_FORMAT = MyAzureParquetFormat )
AS
SELECT month = p.month, population = COUNT ( p.population )
FROM parquet.Population p
WHERE p.year = 2017
GROUP BY p.month

-- Reporting tools can now directly read data from pre-created report
SELECT *
FROM parquet.PopulationByMonth2017
```

# UI based data transformation

Synapse live Validate all Publish all 1

SQL script 10 default

New SQL script New notebook New data flow New integration dataset Upload Download More

**default > Parquet**

Name	Last Modified	Content Type
_SUCCESS	11/16/2020, 4:49:15 PM	
part-00000-5ae12a71-d27d-4e3a-a686-3bfb7d67c2c9-c000.snappy.parquet	11/16/2020, 4:49:14 PM	

New SQL script > Select TOP 100 rows  
New notebook > **Create external table** (highlighted)  
New data flow  
New integration dataset  
Manage access...  
Rename...  
Download  
Delete  
Properties...

**Create external table**

part-00000-5ae12a71-d27d-4e3a-a686-3bfb7d67c2c9-c000.snappy.parquet

External tables provide a convenient way to persist the schema of data residing in your data lake which can be reused for future adhoc analytics. [Learn more](#)

Select SQL pool \*  Built-in

Select a database \*  SQLServerlessDB

External table name \*  adls.retailsales1 (highlighted)

Create external table \*  Using SQL script (highlighted)

**Automatically**   
**Using SQL script**

This will include the create external table definition and the SELECT Top 100 in your SQL script. You will be required to run the SQL script to create the external table

**Create** **Cancel** [join meetup](#)

```

1  SELECT TOP 100 * FROM adls.retailsale
2  GO
3
4

1  IF NOT EXISTS (SELECT * FROM sys.external_file_formats WHERE name = 'SynapseParquetFormat')
2      CREATE EXTERNAL FILE FORMAT [SynapseParquetFormat]
3          WITH ( FORMAT_TYPE = PARQUET)
4  GO
5
6  IF NOT EXISTS (SELECT * FROM sys.external_data_sources WHERE name = 'default_azureSynapseA_dfs_core_windows_net')
7      CREATE EXTERNAL DATA SOURCE [default_azureSynapseA_dfs_core_windows_net]
8          WITH (
9              LOCATION   = 'https://azuresynapsesa.dfs.core.windows.net/default',
10             )
11 Go
12
13 CREATE EXTERNAL TABLE adls.retailsale (
14     [storeId] varchar(8000),
15     [productCode] varchar(8000),
16     [quantity] varchar(8000),
17     [logQuantity] varchar(8000),
18     [advertising] varchar(8000),
19     [price] varchar(8000),
20     [weekStarting] varchar(8000),
21     [id] varchar(8000)
22     )
23     WITH (
24         LOCATION = 'Parquet/part-00000-5ae12a71-d27d-4e3a-a686-3bfb7d67c2c9-c000.snappy.parquet',
25         DATA_SOURCE = [default_azureSynapseA_dfs_core_windows_net],
26         FILE_FORMAT = [SynapseParquetFormat]
27     )
28 Go
29
30 SELECT TOP 100 * FROM adls.retailsale

```

# Automatic syncing of Spark tables

## Overview

Tables created in Spark pool are automatically created as external tables that reference external files in your serverless SQL pool logical data warehouse

## Benefits

Tables designed using Spark languages are immediately available in serverless SQL pool.

Schema definition matches original

Spark table updates are applied in serverless SQL pool

No need to manually create SQL tables that match Spark tables

Spark and serverless SQL pool tables references the same external files.

The screenshot shows the Azure Synapse Analytics studio interface. On the left is a dark sidebar with various icons. The main area has a top navigation bar with 'Create external table' and other options like 'Cell', 'Run all', 'Undo', 'Publish', and three dots. Below the navigation is a section labeled 'Cell 1' with a play button icon. Inside the cell, there is a code editor containing the following SQL:

```

1 %%sql
2 create table data1017 using parquet
3 location 'abfss://container@demostorage.dfs.core.windows.net/data/'

```

Below the code editor is a SQL query window titled 'SQLQuery\_1 - sqlikon...oud!SA'. It shows a SELECT statement:

```

1 SELECT TOP (10) [ExtractId]
2 , [DayOfWeekID]
3 , [DayOfWeekDescr]
4 , [DayOfWeekDescrShort]
5 , [ExtractDateTime]
6 , [LoadTS]
7 , [DeltaActionCode]
8 FROM [default]..[data1017]

```

On the right side of the interface, there are two tabs: 'Results' and 'Messages'. The 'Results' tab displays a table of data with columns: ExtractId, DayOfWeekID, DayOfWeekDescr, DayOfWeekDescrShort, ExtractDateTime, LoadTS, and DeltaActionCode. The data is as follows:

	ExtractId	DayOfWeekID	DayOfWeekDescr	DayOfWeekDescrShort	ExtractDateTime
1	6b86b273ff34fce19d6b804eff5a...	1	Sunday	Sun	2020-01-22 00:00:00.000
2	d4735e3a265e16eee03f5a718h9b...	2	Monday	Mon	2020-01-22 00:00:00.000
3	4e07408562bedb8b60c0531uct...	3	Tuesday	Tue	2020-01-22 00:00:00.000
4	4b22777d4dd1fc61c6f884f4864...	4	Wednesday	Wed	2020-01-22 00:00:00.000
5	ef2d127de37b942baad06145e54b...	5	Thursday	Thu	2020-01-22 00:00:00.000
6	e7f6c011776e8db7cd330b54174f...	6	Friday	Fri	2020-01-22 00:00:00.000
7	70000000-0000-0000-0000-000000000000	7	Saturday	Sat	2020-01-22 00:00:00.000



# Azure Synapse Analytics

## Apache Spark



# Azure Synapse Apache Spark - Summary

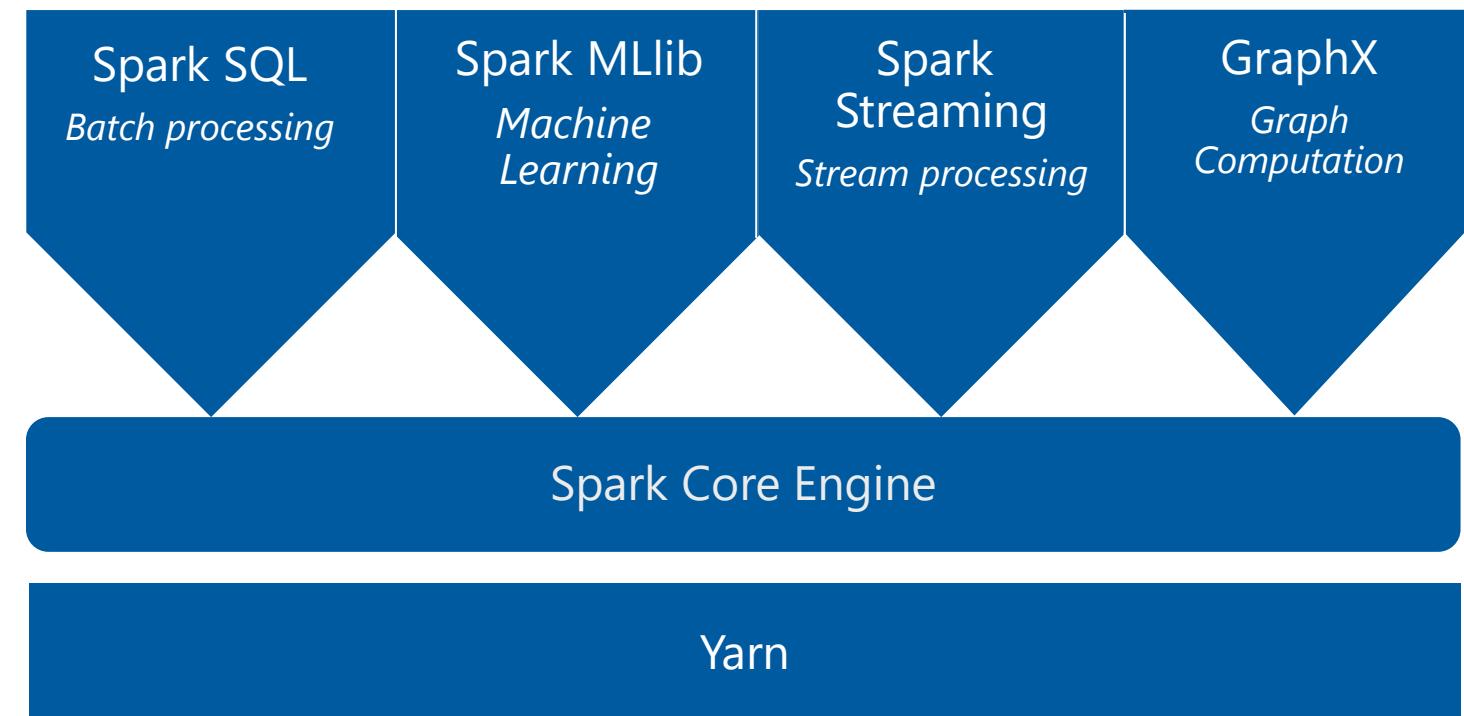
- **Apache Spark 2.4 derivation**
  - Linux Foundation Delta Lake 0.6 support
  - .Net Core 3.0 support
  - Python 3.6 + Anacondas support
- **Tightly coupled to other Azure Synapse services**
  - Integrated security and sign on
  - Integrated Metadata
  - Integrated and simplified provisioning
  - Integrated UX including interact based notebooks
  - Fast load of Synapse SQL (provisioned) pools
- **Core scenarios**
  - Data Prep/Data Engineering/ETL
  - Machine Learning via Spark ML and Azure ML integration
  - Extensible through library management
- **Efficient resource utilization**
  - Fast Start
  - Auto scale (up and down)
  - Auto pause
  - Min cluster size of 3 nodes
- **Multi Language Support**
  - .Net (C#), PySpark, Scala, Spark SQL, Java

# Apache Spark

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

## Spark Unifies:

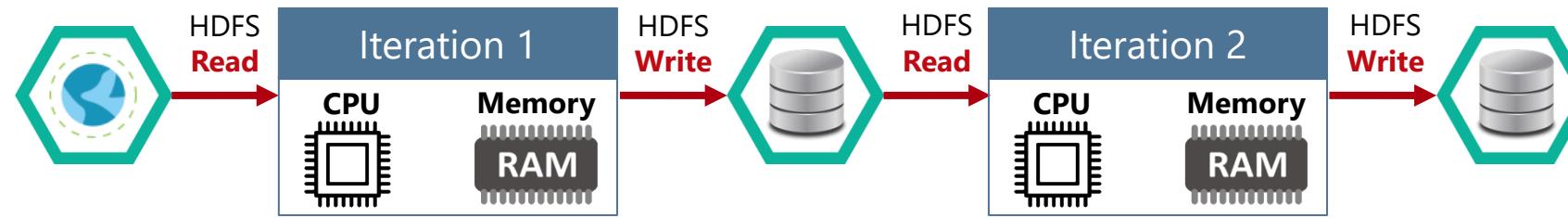
- Batch Processing
- Interactive SQL
- Real-time processing
- Machine Learning
- Deep Learning
- Graph Processing



<http://spark.apache.org>

# Motivation for Apache Spark

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

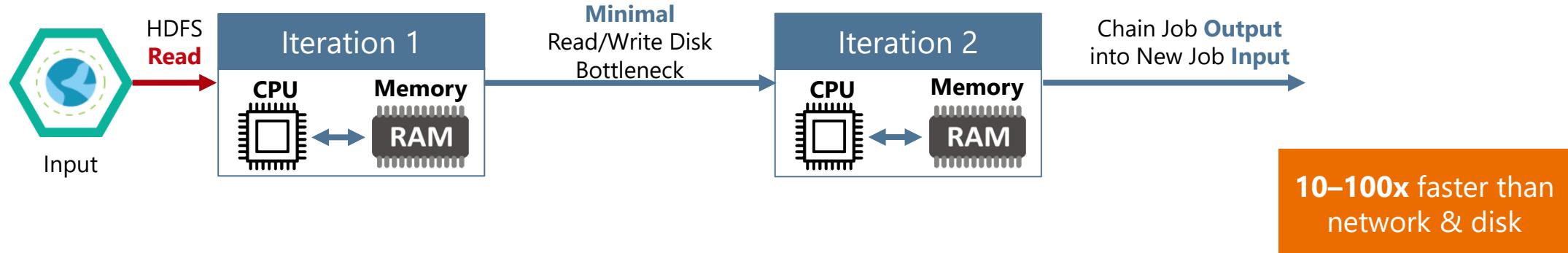


# Motivation for Apache Spark

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

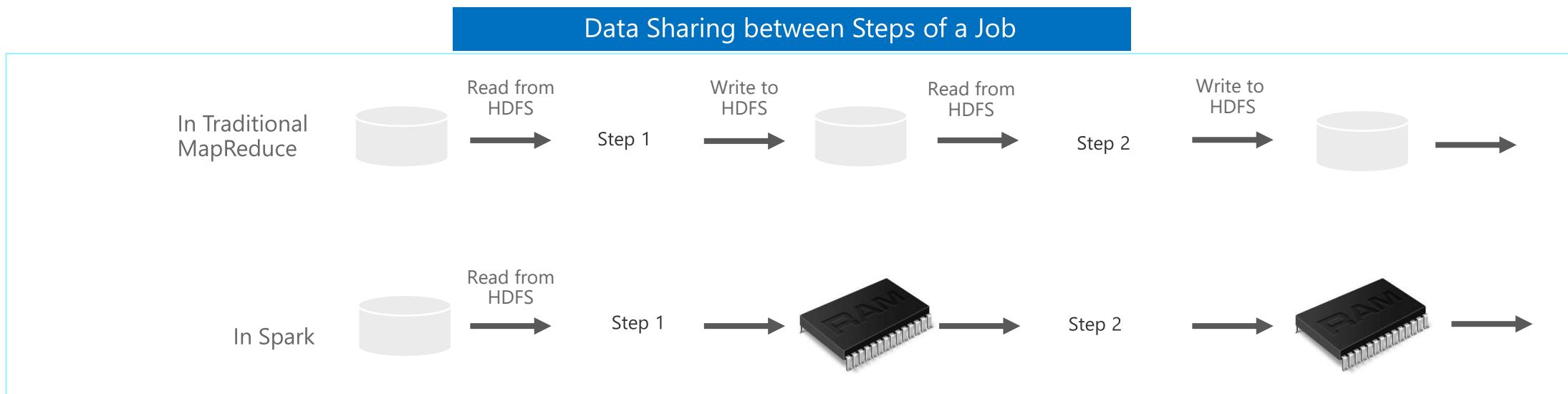


Solution: Keep data **in-memory** with a new distributed execution engine



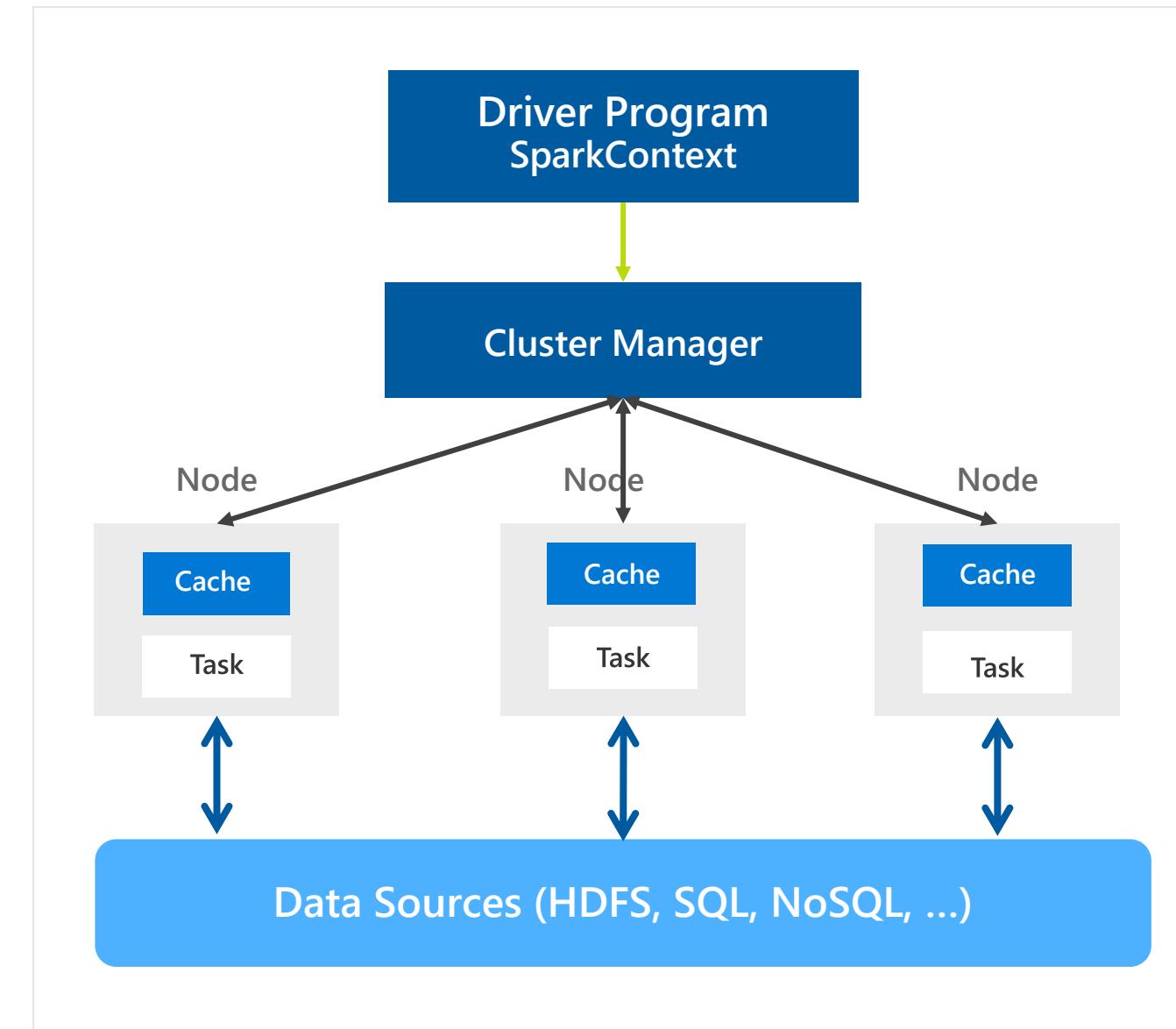
# What makes Spark fast

- **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

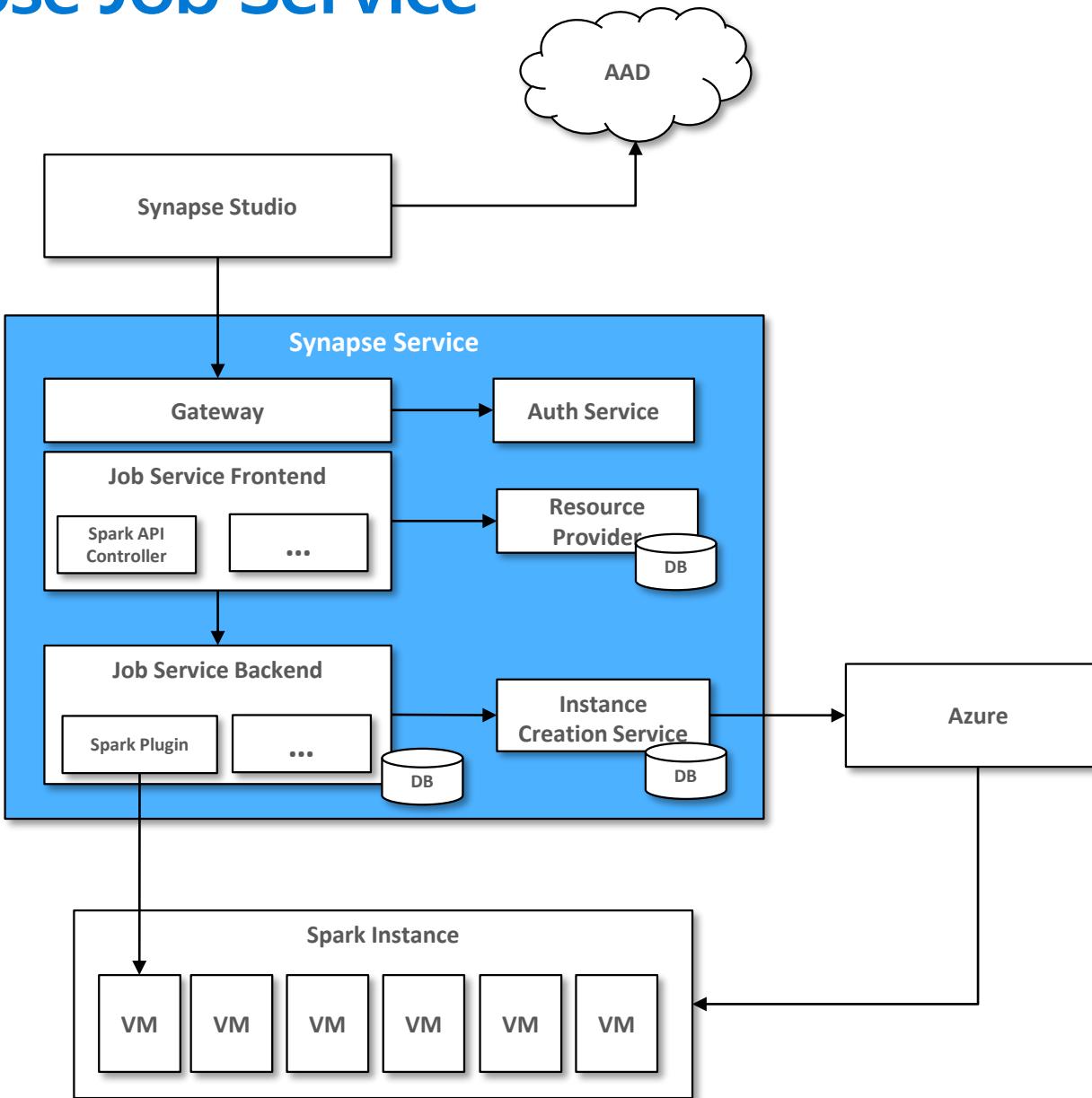
- Represent and analyze systems represented by graph nodes
- Trace interconnections between graph nodes
- Applicable to use cases in transportation, telecommunications, road networks, modeling personal relationships, social media, etc.



# Azure Synapse Apache Spark

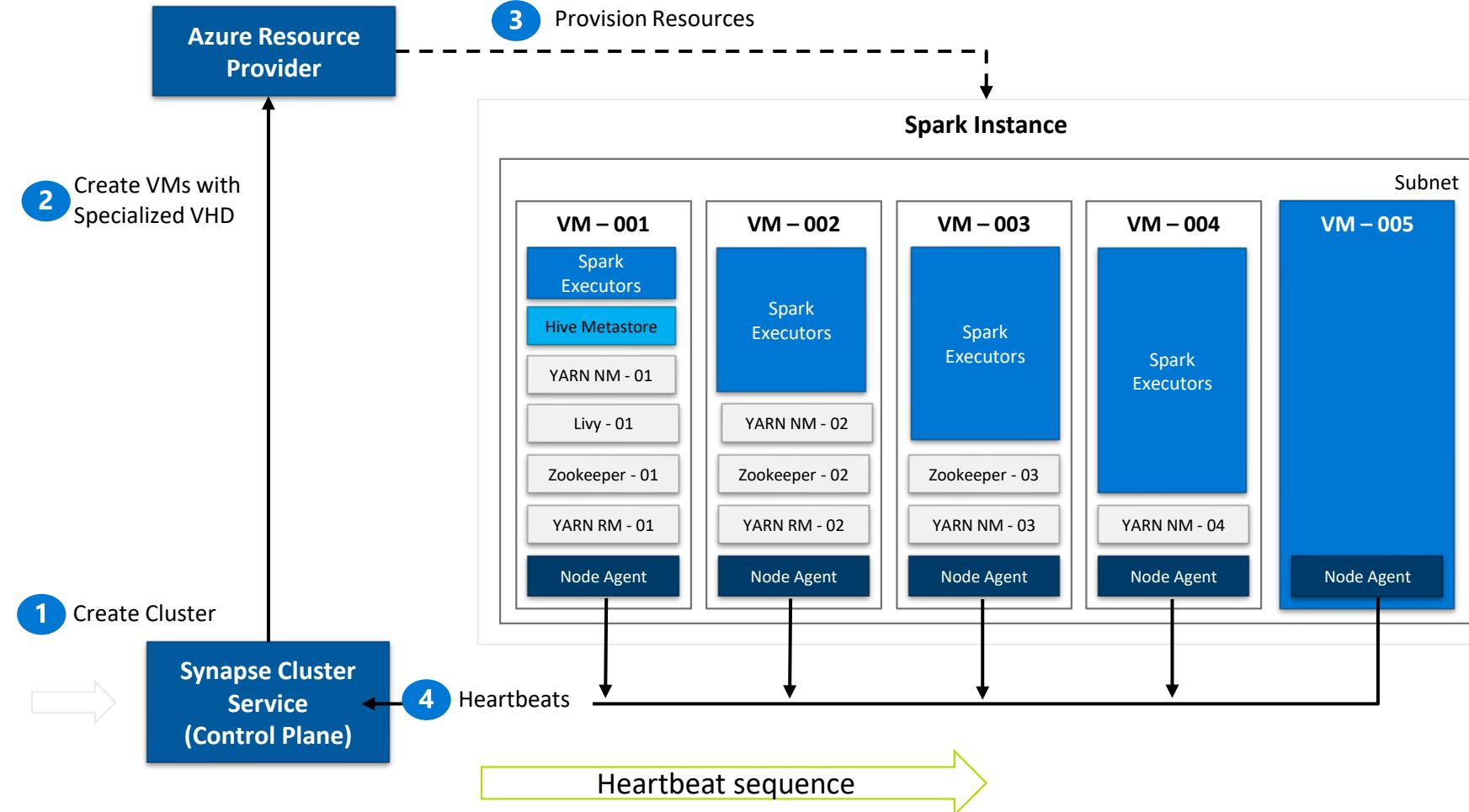
## Architecture Overview

# Synapse Job Service



- User creates Synapse Workspace and Spark pool and launches Synapse Studio.
- User attaches Notebook to Spark pool and enters one or more Spark statements (code blocks).
- The Notebook client gets user token from AAD and sends a Spark session create request to Synapse Gateway.
- Synapse Gateway authenticates the request and validates authorizations on the Workspace and Spark pool and forwards it to the Spark (Livy) controller hosted in Synapse Job Service frontend.
- The Job Service frontend forwards the request to Job Service backend that creates two jobs – one for creating the cluster and the other for creating the Spark session.
- The Job service backend contacts Synapse Resource Provider to obtain Workspace and Spark pool details and delegates the cluster creation request to Synapse Instance Service.
- Once the instance is created, the Job Service backend forwards the Spark session creation request to the Livy endpoint in the cluster.
- Once the Spark session is created the Notebook client sends Spark statements to the Job Service frontend.
- Job Service frontend obtains the actual Livy endpoint for the cluster created for the particular user from the backend and sends the statement directly to Livy for execution.

# 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 (1 of 2)

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

Basic Settings – Minimum details required from user

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  Disabled

Number of nodes \*

3  40

Only required field from user

Default Settings

# Creating a Spark pool (2 of 2) - optional

Additional Settings offer optional settings to customize Spark pool

Customize component versions, auto-pause

Home > prlangadws2 > 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 \* 15

**Component versions**

Select the Apache Spark version for your Apache Spark pool.

Component	Version
Apache Spark *	2.4
Python	3.6.1
Scala	2.11.12
Java	1.8.0_222
.NET Core	3.1
.NET for Apache Spark	0.10.0
Delta Lake	0.5.0

**Packages**

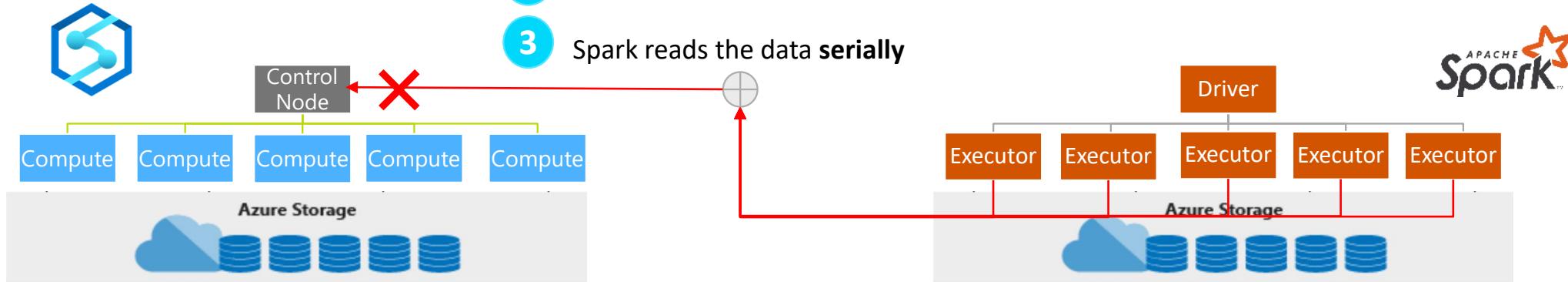
Upload environment configuration file ("PIP freeze" output).

File upload Select a file Upload

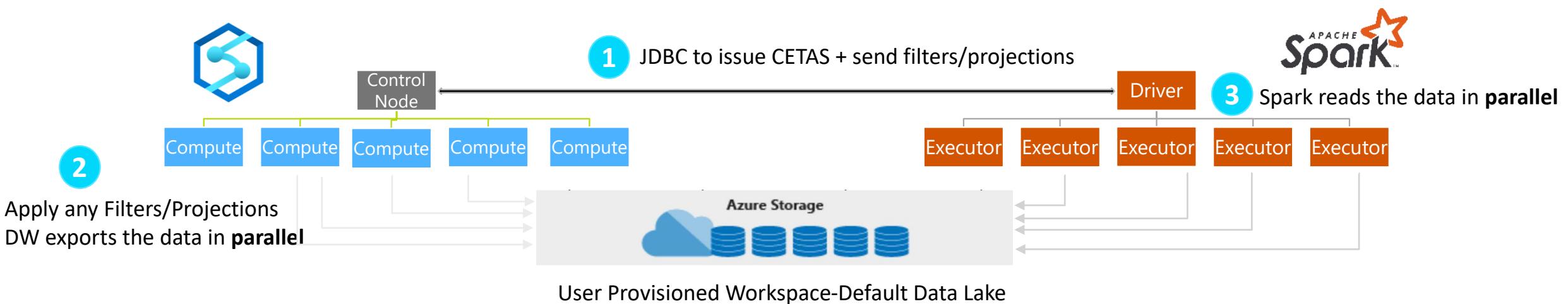
Review + create < Previous Next: Tags >

Import libraries by providing text file containing library name and version

## Existing Approach: JDBC



## New Approach: JDBC and Polybase



# Code-Behind Experience

## Existing Approach

```
val jdbcUsername = "<SQL DB ADMIN USER>"  
val jdbcPwd = "<SQL DB ADMIN PWD>"  
val jdbcHostname = "servername.database.windows.net"  
val jdbcPort = 1433  
val jdbcDatabase = "<AZURE SQL DB NAME>"  
  
val jdbc_url =  
  s"jdbc:sqlserver://${jdbcHostname}:${jdbcPort};database=${jdbcDatabase};"  
  encrypt=true;trustServerCertificate=false;hostNameInCertificate=*.databas  
e.windows.net;loginTimeout=60;"  
  
val connectionProperties = new Properties()  
  
connectionProperties.put("user", s"${jdbcUsername}")  
connectionProperties.put("password", s"${jdbcPwd}")  
  
val sqlTableDf = spark.read.jdbc(jdbc_url, "dbo.Tbl1", connectionProperties)
```

## New Approach

```
// Construct a Spark DataFrame from dedicated SQL pool  
var df = spark.read.sqlAnalytics("sql1.dbo.Tbl1")  
  
// Write the Spark DataFrame into dedicated SQL pool  
df.write.sqlAnalytics("sql1.dbo.Tbl2")
```

# Create Notebook on files in storage

The screenshot illustrates the process of creating a Notebook on files stored in Azure Storage. The left pane shows the Azure portal navigation bar and the Synapse Analytics workspace. The main area displays the Data blade for the 'nyctic' storage account. A red box highlights the 'New notebook' option in the context menu for a specific file ('part-00055'). A red arrow points from this menu to the bottom-right pane, which shows the execution of a PySpark job. The job's code reads data from an ABFS path and displays its schema and first few rows.

**Left Pane (Data Blade):**

- Home
- Data** (selected)
- Develop
- Orchestrate
- Monitor
- Manage

**Right Panes:**

- Top Right:** Shows the 'nyctic' storage account structure under 'green' folder, including 'puYear=2009' and 'puMonth=1' subfolders.
- Bottom Right:** Shows the execution of a PySpark job named 'Notebook 4'. The code reads data from the specified ABFS path and prints the schema and first few rows of the dataset.

```

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

ID	DESCRIPTION	STATUS	STAGES	TASKS	SUBMISSION TIME	DURATION
Job 0	load at NativeMethodAccessorImpl.java:0	Succeeded	1/1		11/14/2019, 9:56:49 AM	7s
Job 1	showString at NativeMethodAccessorImpl.java:0	Succeeded	1/1		11/14/2019, 9:56:58 AM	1s
Job 2	showString at NativeMethodAccessorImpl.java:0	Succeeded	1/1		11/14/2019, 9:56:59 AM	11s

```

+-----+-----+-----+-----+-----+-----+
|vendorID|tpepPickupDateTime|tpepDropoffDateTime|passengerCount|tripDistance|puLocationId|doLocationId|startLon|startLat|endLon|endLat|
+-----+-----+-----+-----+-----+-----+
| 2|2015-02-28 23:53:18|2015-03-01 00:00:29|       6|    1.63|      null| -74.00084686279297| 40.73069381713867| -73.9841537475586| 40.74470520019531| |
| 1|          N|           1|     7.5|   0.5|   0.5|      null|    0.0|    10.56|      null|      null|
| 1|2015-03-28 19:21:05|2015-03-28 19:28:31|       1|    2.2|      null| -73.97765350341797| 40.763160705566406| -73.95502471923828| 40.78600311279297|
| 1|          N|           1|     8.5|   0.0|   0.5|      null|    0.3|    2.3|      null|    11.6|
| 2|2015-02-28 23:53:19|2015-03-01 00:12:08|       5|    3.23|      null| -73.96012878417969| 40.76215744018555| -73.9881591796875| 40.72818896484375|
| 1|          N|           1|    14.5|   0.5|   0.5|      null|    0.3|    4.74|      null|    28.54|
| 1|2015-03-28 19:21:05|2015-03-28 19:37:02|       1|    2.1|      null| -73.98143005371094| 40.7815055847168| -74.000891552734375| 40.76177215576172|
+-----+-----+-----+-----+-----+-----+
  
```

Ready | Stop session | Spark history server | Configure session

View results in table format

The screenshot shows the Azure Synapse Analytics workspace interface. On the left, there's a sidebar with 'Develop' selected, showing a list of notebooks including '00\_DataPrep', '01\_TrainingUseMllib\_cleanup', 'automl\_arcaida\_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' (which is highlighted), and 'SparkPerf'. The main area has tabs for 'Data Download...', 'NYCTaxi\_Docs...', 'SeattleSafetyD...', and 'Repro'. Below these tabs, there are buttons for 'Cell', 'Run all', 'Publish', 'Attach to', 'Language' (set to PySpark (Python)), and 'PySpark (Python)' dropdown. The 'SeattleSafetyDoc' notebook is open, showing four cells. Cell 1 contains code for reading data from Azure Blob Storage. Cell 2 contains code for creating a temporary view named 'seattlesafety'. Cell 3 shows the resulting table output:

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 contains code for writing the data back to Azure Blob Storage.

View results in chart format

SQL support

The screenshot shows the Azure Synapse Analytics workspace interface. At the top, there's a navigation bar with 'Microsoft Azure', 'Synapse Analytics', and a user profile. Below it is a toolbar with 'Publish all', 'Validate all', 'Refresh', 'Discard all', and other options. A search bar labeled 'Search resources' is also present.

The main area contains several sections:

- Develop**: A sidebar showing a list of notebooks, with 'SeattleSafetyDoc' selected and highlighted in blue.
- Cell 1**: PySpark code for reading data from Azure Blob Storage and creating a DataFrame named 'seasafety\_df'. It includes a command history entry: "Command executed in 2mins 18s 412ms by euang on 11-22-2019 00:44:52.415 -08:00".
- Job execution**: A table showing a single job named 'Job 0' in progress, with 1 executor and 4 cores. The status is 'In progress' with 0/1 tasks active. Submission time was 11/22/2019, 12:44:46 AM, and duration was 13m43s.
- Cell 2**: PySpark code to create a temporary view named 'seattlesafety' from the DataFrame.
- Cell 3**: PySpark code to display the contents of the 'seattlesafety' view using the SQL command 'SELECT \* FROM seattlesafety'. A red box highlights this line of code. To the right, a pie chart is displayed with various categories labeled: 'Automatic Fire Alarm False', 'Medic Response, 7 per Rule', 'Aid Response Yellow', 'MVI - Motor Vehicle Incident', 'Medic Response, 6 per Rule', 'Motor Vehicle Accident', 'Automatic Medical Alarm', '1RED 1 Unit', 'Auto Fire Alarm', 'Automatic Fire Alarm Resd', 'Medic Response', 'Trans to AMR', 'longitude', and 'Aid Response'. A red arrow points from the 'View results in chart format' text to this chart. On the right side of the chart, there's a configuration panel for the pie chart.
- Cell 4**: PySpark code to write the DataFrame back to Azure Blob Storage as a CSV file.

Develop + Discard all

Data Download... NYCTaxi\_Docs... \* X

+ Cell 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 # for 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_title('Tip amount distribution')
7 ax1.set_xlabel('Tip Amount ($)')
8 ax1.set_ylabel('Counts')
9 plt.subtitle('')
10 plt.show()
11
12 # How many passengers tip'd by various amounts
13 ax2 = sampled_taxi_pd_df.boxplot(column=['tipAmount'], by=['passengerCount'])
14 ax2.set_title('Tip amount by Passenger count')
15 ax2.set_xlabel('Passenger count')
16 ax2.set_ylabel('Tip Amount ($)')
17 plt.subtitle('')
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_xlabel('Fare Amount ($)')
23 ax.set_ylabel('Tip Amount ($)')
24 plt.axis([-2, 80, -2, 20])
25 plt.subtitle('')
26 plt.show()
27
```

Tip amount distribution

Tip amount by Passenger count

Tip amount by Fare amount

Exploratory data analysis with graphs – histogram, boxplot etc

# Library Management - Python

## Overview

Customers can add new python libraries at Spark pool level

## Benefits

Input requirements.txt in simple pip freeze format

Add new libraries to your cluster

Update versions of existing libraries on your cluster

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

## Constraints

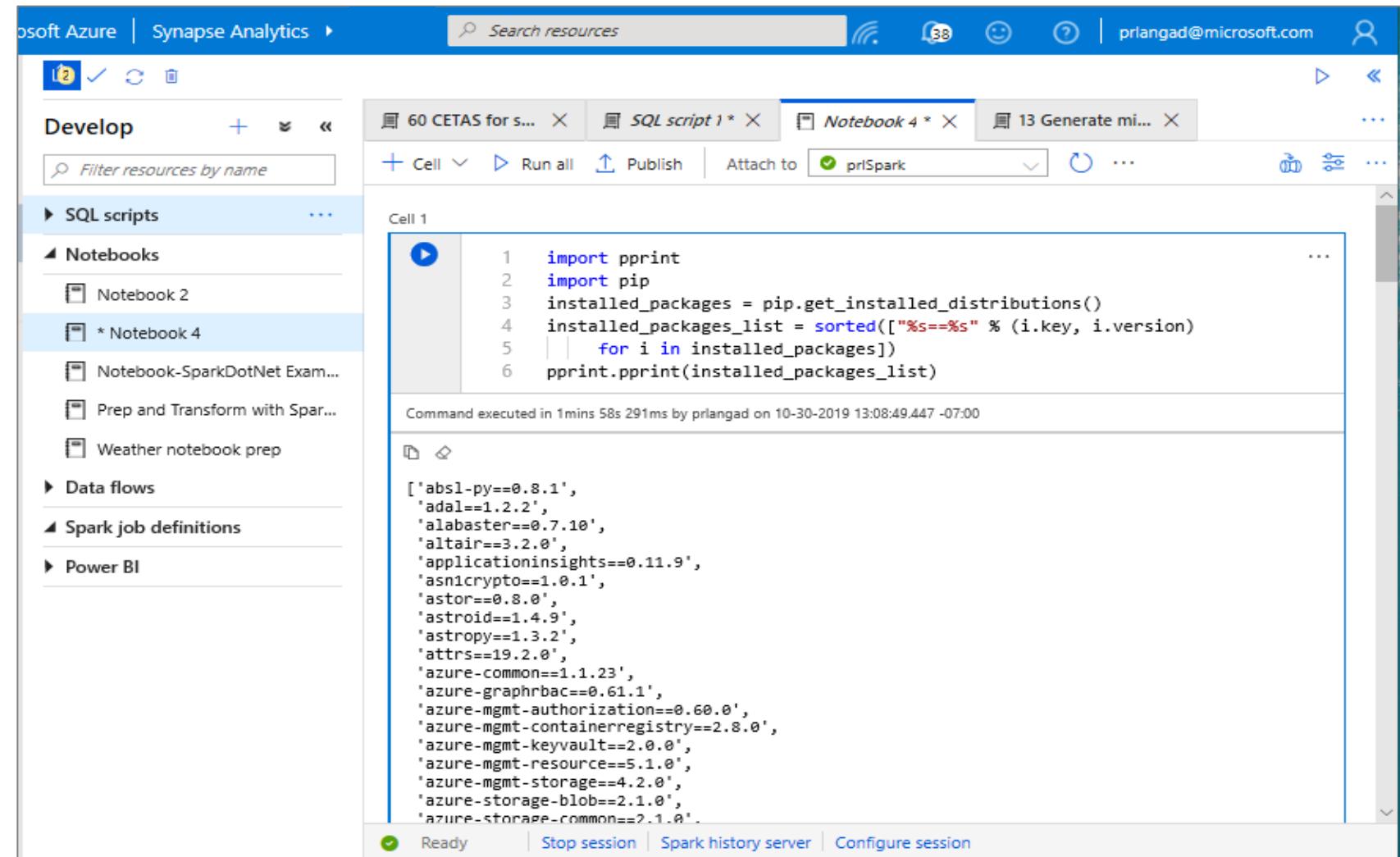
The library version must exist on PyPI repository

Version downgrade of an existing library not allowed

The screenshot shows the Microsoft Azure Synapse Analytics Library Management interface. The left sidebar includes options like Analytics pools, SQL pools, Apache Spark pools (which is selected), External connections, Linked services, Orchestration, Triggers, Integration runtimes, Security, Access control, and Managed private endpoints. The main area displays the 'Apache Spark pools' section, which lists three items: 'priLangadSpark2', 'priLang-syntaxcheck', and 'priSpark'. A red box highlights the 'Packages' section, which contains a link to 'Upload environment config file' and a 'Refresh' button. The right side shows the properties for the selected pool ('priSpark'), including its name, URL, creation date (10/30/2019, 12:50:37 PM), and sections for Configuration and Workspace.

# Library Management - Python

Get list of installed libraries with version information



The screenshot shows the Azure Synapse Analytics interface with the 'Develop' workspace selected. In the center, a notebook titled 'Notebook 4' is open. A code cell in this notebook contains the following Python script:

```
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)
```

When run, the cell outputs a long list of installed Python packages and their versions, such as:

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

The notebook also shows other tabs like 'SQL script 1' and 'Notebook 4'. The status bar at the bottom indicates the session is 'Ready'.

# Spark ML Algorithms

## Spark ML Algorithms

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

# Microsoft Machine Learning for Apache Spark

v1.0-rc

Microsoft's Open Source  
Contributions to Apache Spark



Distributed  
Machine Learning



Fast Model  
Deployment



Microservice  
Orchestration



Multilingual Binding  
Generation

[www.aka.ms/spark](http://www.aka.ms/spark)

 Azure/mmlspark

# Microsoft Spark Utilities

## Overview

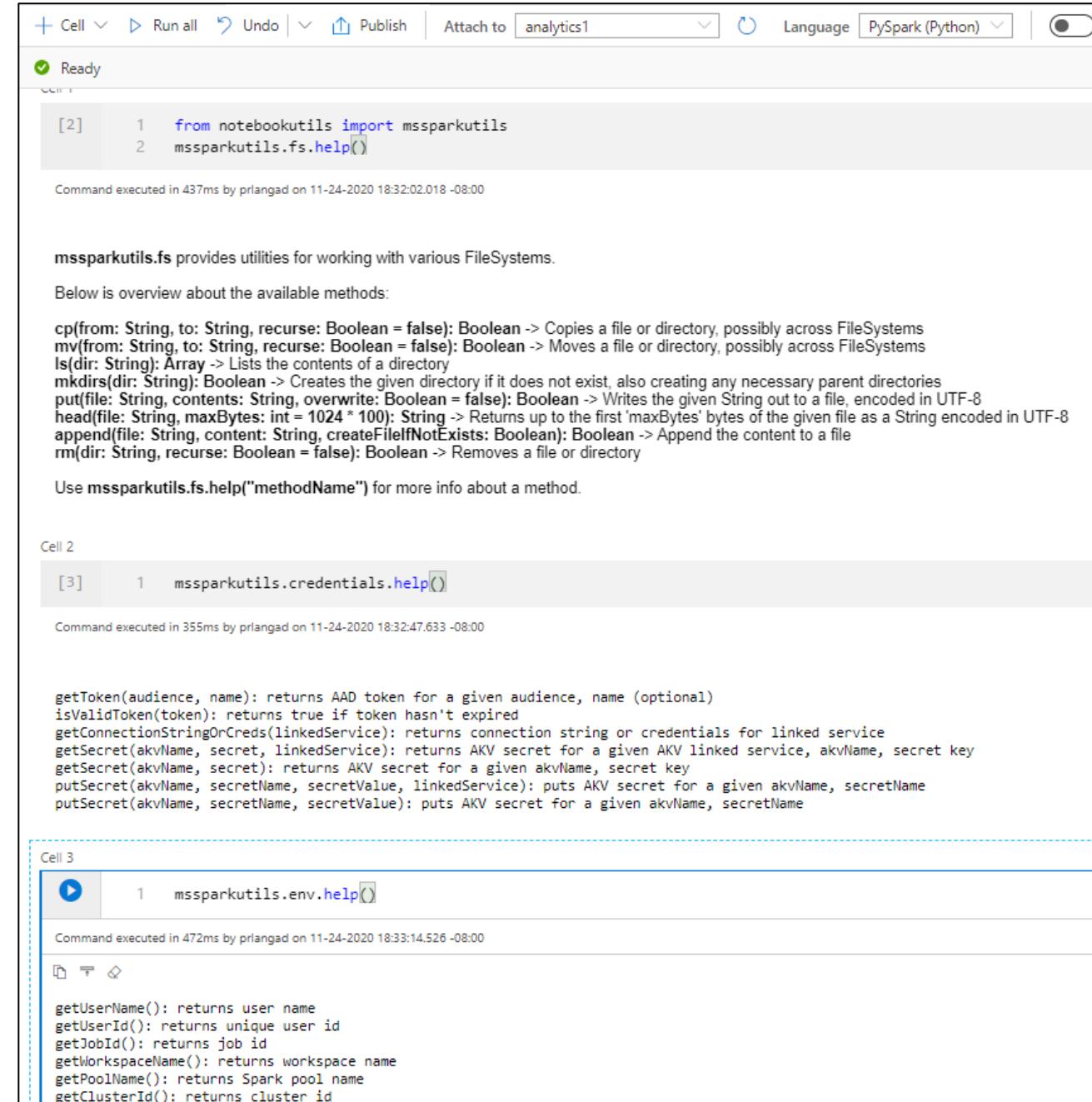
It provides utilities for working with file systems, including ADLS Gen2 and Azure Blob Storage.

## Benefits

It supports multiple methods for file systems such as List, Copy, Move, Write, Append, Delete file or directory, View file properties, Create new directory, Preview file content.

It supports environment utilities to get username, user id, job id, workspace name, pool name, cluster id.

It supports to get the access tokens of linked services and manage secrets in Azure Key Vault.



The screenshot shows a Jupyter Notebook interface with three code cells. Cell 1 shows the import of mssparkutils and its help function. Cell 2 shows the import of credentials and its help function. Cell 3 shows the import of env and its help function. Each cell includes the command, its execution time, and the resulting documentation for the specified method.

```
[2]: 1 from notebookutils import mssparkutils
      2 mssparkutils.fs.help()

Command executed in 437ms by prlangad on 11-24-2020 18:32:02.018 -08:00

mssparkutils.fs provides utilities for working with various FileSystems.

Below is overview about the available methods:

cp(from: String, to: String, recurse: Boolean = false): Boolean -> Copies a file or directory, possibly across FileSystems
mv(from: String, to: String, recurse: Boolean = false): Boolean -> Moves a file or directory, possibly across FileSystems
ls(dir: String): Array -> Lists the contents of a directory
mkdirs(dir: String): Boolean -> Creates the given directory if it does not exist, also creating any necessary parent directories
put(file: String, contents: String, overwrite: Boolean = false): Boolean -> Writes the given String out to a file, encoded in UTF-8
head(file: String, maxBytes: int = 1024 * 100): String -> Returns up to the first 'maxBytes' bytes of the given file as a String encoded in UTF-8
append(file: String, content: String, createFileIfNotExists: Boolean): Boolean -> Append the content to a file
rm(dir: String, recurse: Boolean = false): Boolean -> Removes a file or directory

Use mssparkutils.fs.help("methodName") for more info about a method.

Cell 2

[3]: 1 mssparkutils.credentials.help()

Command executed in 355ms by prlangad on 11-24-2020 18:32:47.633 -08:00

getToken(audience, name): returns AAD token for a given audience, name (optional)
isValidToken(token): returns true if token hasn't expired
getConnectionStringOrCreds(linkedService): returns connection string or credentials for linked service
getSecret(akvName, secret, linkedService): returns AKV secret for a given AKV linked service, akvName, secret key
getSecret(akvName, secret): returns AKV secret for a given akvName, secret key
putSecret(akvName, secretName, secretValue, linkedService): puts AKV secret for a given akvName, secretName
putSecret(akvName, secretName, secretValue): puts AKV secret for a given akvName, secretName

Cell 3

[4]: 1 mssparkutils.env.help()

Command executed in 472ms by prlangad on 11-24-2020 18:33:14.526 -08:00

getUser(): returns user name
getId(): returns unique user id
getJobId(): returns job id
getWorkspaceName(): returns workspace name
getPoolName(): returns Spark pool name
getClusterId(): returns cluster id
```

# Hypespace

## Overview

Hypespace introduces the ability for Apache Spark users to create indexes on their data

## Benefits

It helps accelerate your workloads or queries containing filters on predicates with high selectivity or a join that requires heavy shuffles.

Maintain the indexes through a multi-user concurrency model.

Leverage these indexes automatically, within your Spark workloads, without any changes to your application code for query/workload acceleration.

It supports index operations as create index, list index, restore index, delete index, vacuum index

Languages supported: Scala, Python, .NET

```

Create indexes
Cell 4
[ ] 1 # Create indexes from configurations
2
3
4 hyperspace.createIndex(emp_DF, emp_IndexConfig)
5 hyperspace.createIndex(dept_DF, dept_IndexConfig1)
6 hyperspace.createIndex(dept_DF, dept_IndexConfig2)

List indexes
Cell 6
[ ] 1 hyperspace.indexes().show()

Index usage
Cell 8
[ ] 1 # Enable Hypespace
2 Hyperspace.enable(spark)
3
4 emp_DF = spark.read.parquet(emp_Location)
5 dept_DF = spark.read.parquet(dept_Location)
6
7 emp_DF.show(5)
8 dept_DF.show(5)
9
10 # Filter with equality predicate
11 eqFilter = dept_DF.filter("deptId = 20").select("deptName")
12 eqFilter.show()
13
14 hyperspace.explain(eqFilter, True, displayHTML)

```

# Spark CDM connector

## Overview

It offers Spark dataframes to read and write entities in a CDM folder.

## Benefits

It supports use of Managed Identities for Azure resources to mediate access to the Azure datalake storage.

Writes from a Spark dataframe to an entity in a CDM folder based on dataframe schema or CDM entity definition.

Supports data in Apache Parquet format and CSV format.

The CDM connector is pre-installed and supports languages: Python, Scala

```

1 # Explicit write, creating an entity in a CDM folder based on a pre-defined model
2
3 # Case 1: Using an entity definition defined in the CDM Github repo
4
5 data = [
6     ["1", "2", "3", 4],
7     ["4", "5", "6", 8],
8     ["7", "8", "9", 4],
9     ["10", "11", "12", 8],
10    ["13", "14", "15", 4]
11 ]
12
13 schema = (StructType()
14     .add(StructField("teamMembershipId", StringType(), True))
15     .add(StructField("systemUserId", StringType(), True))
16     .add(StructField("teamId", StringType(), True))
17     .add(StructField("versionNumber", LongType(), True))
18 )
19
20 df = spark.createDataFrame(spark.sparkContext.parallelize(data,1), schema)
21
22 (df.write.format("com.microsoft.cdm")
23     .option("storage", storageAccountName)
24     .option("manifestPath", container + "/explicitTest/root.manifest.cdm.json")
25     .option("entity", "TeamMembership")
26     .option("entityDefinitionPath", "core/applicationCommon/TeamMembership.cdm.json/TeamMembership")
27     .option("useCdmStandardModelRoot", True) # sets the model root to the CDM CDN schema documents folder
28     .option("useSubManifest", True)
29     .mode("overwrite")
30     .save())
31
32 readDf = (spark.read.format("com.microsoft.cdm")
33     .option("storage", storageAccountName)
34     .option("manifestPath", container + "/explicitTest/root.manifest.cdm.json")
35     .option("entity", "TeamMembership")
36     .load())
37
38 readDf.select("*").show()

```

# IntelliJ IDE



Use the Azure Toolkit for IntelliJ plug-in to develop Apache Spark applications and submit applications to Spark pools.

The screenshot shows the IntelliJ IDEA interface with the following details:

- File Structure:** The left sidebar shows the project structure for "DataAnalysis\_v3\_Maven". It includes a "src" directory containing "main", "resources", and "scala" (with "DataAnalysis.scala" selected), and a "test" directory.
- Code Editor:** The main editor window displays the content of "DataAnalysis.scala". The code initializes a SparkSession and performs basic data processing.
- Run/Debug Configurations Dialog:** A modal dialog titled "Run/Debug Configurations" is open on the right. It shows a list of configurations:
  - "Apache Spark on Azure Synapse" (selected)
  - "DataAnalysisRunConfig"
 The configuration settings include:
  - Common Run Parameters:** Main class name is set to "DataAnalysis".
  - Locally Run:** Options for VM options, Program arguments, Working directory (set to "C:\Users\prlangad\IdeaProjects\DataAnalysis\_v3\_Maven"), Environment variables (set to "HADOOP\_HOME=C:\winutils"), and Use classpath of module (set to "DataAnalysis\_v3\_Maven").
  - Remotely Run in Cluster:** Options for Data repositories directory (set to "C:\Users\prlangad\IdeaProjects\DataAnalysis\_v3\_Maven\data"), Data default repo directory (set to "C:\Users\prlangad\IdeaProjects\DataAnalysis\_v3\_Maven\data\\_default\_"), Hadoop user default directory (set to "C:\Users\prlangad\IdeaProjects\DataAnalysis\_v3\_Maven\data\\_default\\_user\current"), and WINUTILS.exe location (set to "C:\winutils\bin\winutils.exe").
  - Enable parallel execution

## Synapse Notebook: Connect to AML workspace

The screenshot shows the Azure Synapse Notebook interface. The left sidebar lists resources under 'Develop': SQL scripts, Notebooks, Data flows, Spark job definitions, and Power BI. The 'Notebooks' section has a selected item: 'automl\_synapse\_local\_regr...'. The main area displays a notebook cell titled 'Check the Azure ML Core SDK Version to Validate Your Installation'. The code in Cell 3 is:

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

The output shows the command was executed in 1s 258ms by balapv on 11-12-2019 14:41:52.805 -08:00, and the result is 'SDK Version: 1.0.69'. Below this, a section titled 'Connect to Azure Workspace' is shown. Cell 5 contains the following code:

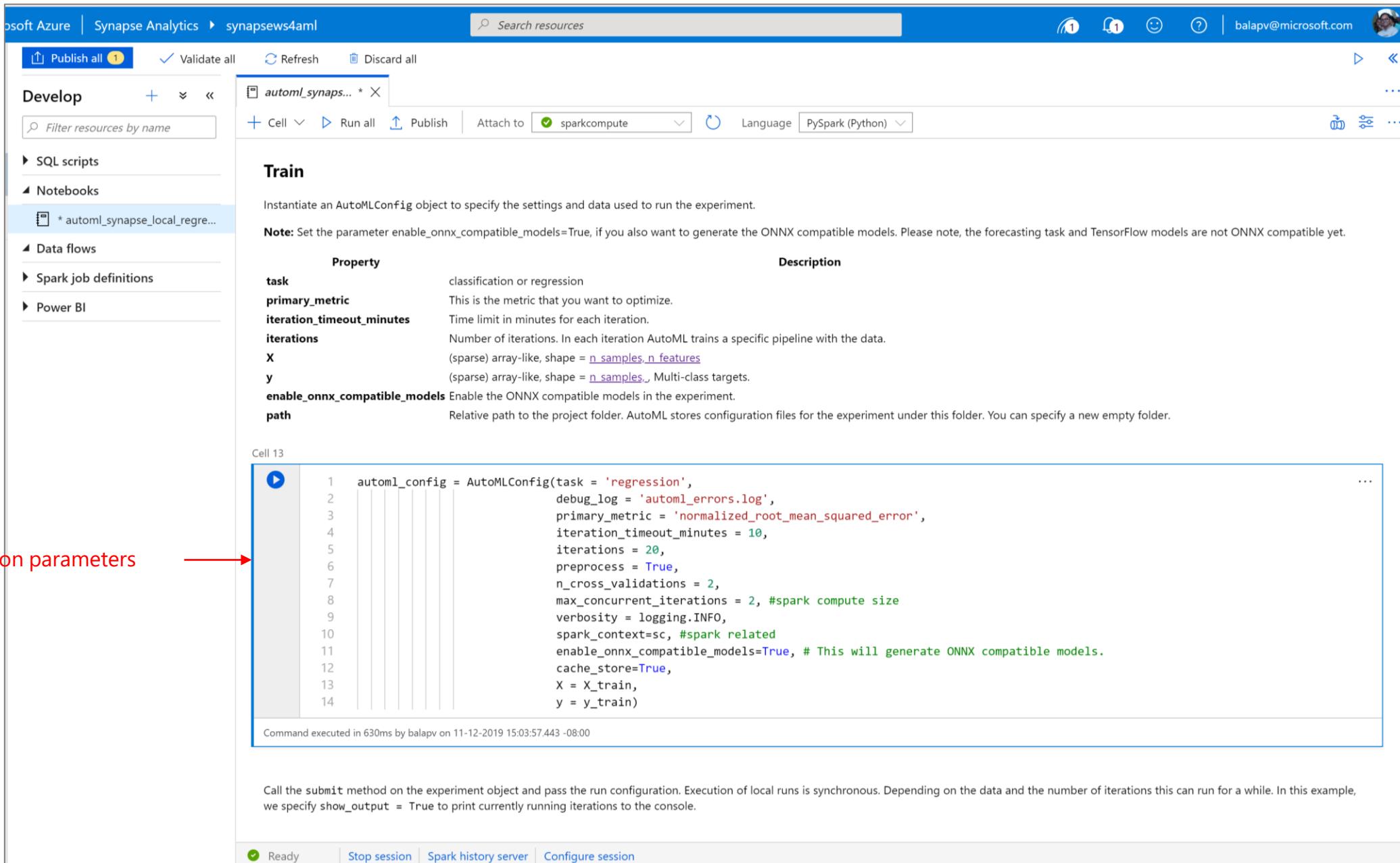
```
[6] 1 ## Import the Workspace class and check the Azure ML SDK version.  
2 from azureml.core import Workspace  
3  
4 ws = Workspace(subscription_id = "6560575d-fa06-4e7d-95fb-f962e74efd7a",  
5 | | | | | resource_group = "balapv-synapse-rg", workspace_name = "AML-WS-synapse")  
6  
7 print(ws.name,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

The output shows the command was executed in 3s 909ms by balapv on 11-12-2019 14:41:55.491 -08:00, and the result is 'AML-WS-synapse westus2 balapv-synapse-rg'. Cell 6 contains the following code:

```
[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
```

At the bottom of the notebook, there are buttons for 'Running', 'Stop session', 'Spark history server', and 'Configure session'. A red arrow points from the text 'Simple code to connect workspace' to the code in Cell 5.

# Synapse Notebook: Configure AML job to run on Synapse



The screenshot shows the Azure Synapse Analytics notebook interface. On the left, the sidebar lists resources under 'Develop': SQL scripts, Notebooks (with 'automl\_synapse\_local\_regr...' selected), Data flows, Spark job definitions, and Power BI. The main area displays the 'Train' section of the 'automl\_synapse\_local\_regr...' notebook. It includes a table of configuration parameters with their descriptions and a code cell showing the Python code for creating an AutoMLConfig object.

**Configuration parameters**

```

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)

```

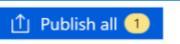
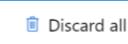
Call the `submit` method on the experiment object and pass the run configuration. Execution of local runs is synchronous. Depending on the data and the number of iterations this can run for a while. In this example, we specify `show_output = True` to print currently running iterations to the console.

Bottom navigation bar: Ready, Stop session, Spark history server, Configure session.

# Synapse Notebook: Run AML job

Microsoft Azure | Synapse Analytics > synapsews4aml

Search resources Show notifications    balapv@microsoft.com 

 Publish all  Validate all  Refresh  Discard all

**Develop**   Filter resources by name

 Run all  Publish Attach to  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

**ML job execution result** 

Get Azure Portal URL for Monitoring Runs

 Running  Stop session  Spark history server  Configure session



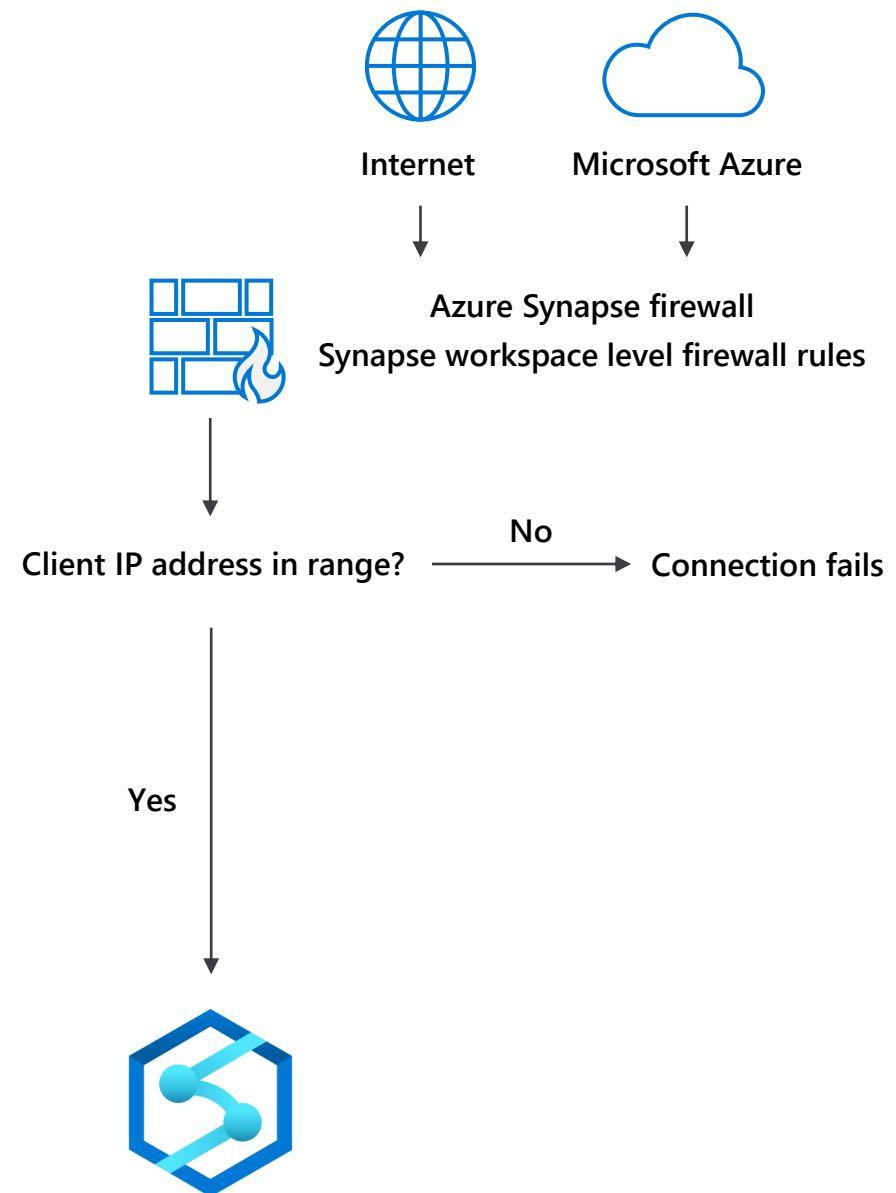
# Azure Synapse Analytics Security

# Securing with firewalls

## Overview

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.

# IP Firewall Rules

## Overview

IP firewall rules grant or deny access to user's Synapse workspace based on the originating IP address of each request.

IP firewall rules configured at the workspace level apply to all public endpoints of the workspace (dedicated SQL pool, serverless SQL pool, and Development).

## Key Points

- Customers can also use SQL Server Management Studio (SSMS) to connect to the SQL resources (dedicated SQL pool and serverless SQL pool) in their workspace.
- Customers must ensure that the firewall on the network and local computer allow outgoing communication on TCP ports 80, 443 and 1443 for Synapse Studio.
- Customers must also allow outgoing communication on UDP port 53 for Synapse Studio.
- To connect using tools such as SSMS and Power BI, user must allow outgoing communication on TCP port 1433.

# Managed VNet

## Overview

Creating a workspace with a Managed workspace VNet associated with it ensures that user's workspace is network isolated from other workspaces. The VNet associated with your workspace is managed by Azure Synapse. This VNet is called a Managed workspace VNet.

## Benefits

- With a Managed workspace customers can offload the burden of managing the VNet to Azure Synapse.
- Customers don't have to configure inbound NSG rules on their own VNets to allow Azure Synapse management traffic to enter their VNet.
- Customers don't need to create a subnet for your Spark clusters based on peak load.
- Managed workspace VNet along with Managed private endpoints protects against data exfiltration.

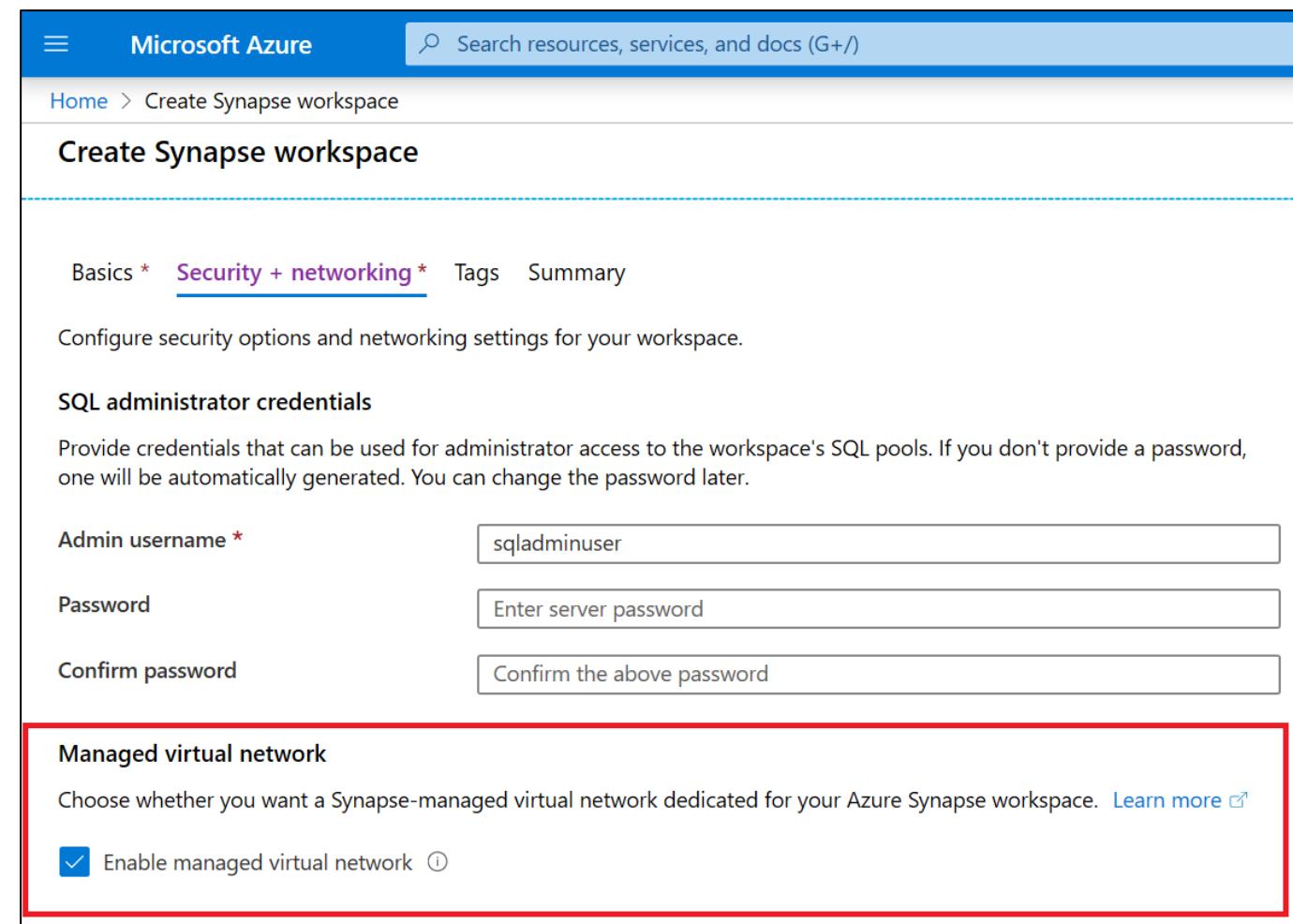
# Managed VNet

## Overview

During Azure Synapse workspace creation, user can choose to associate it to a managed VNet. User cannot change this workspace configuration after the workspace is created.

User cannot reconfigure a workspace that does not have a Managed workspace VNet associated with it and associate a VNet to it.

Private links are supported only in Synapse workspaces that have a managed VNet associated to it.



The screenshot shows the 'Create Synapse workspace' wizard in the Microsoft Azure portal. The 'Security + networking' tab is active. The 'SQL administrator credentials' section is filled with 'sqladminuser' for the admin username and 'Enter server password' for the password. Below this, the 'Managed virtual network' section is highlighted with a red border. It contains a note about choosing a Synapse-managed virtual network and a checked checkbox for 'Enable managed virtual network'.

Basics \* **Security + networking \*** Tags Summary

Configure security options and networking settings for your workspace.

**SQL administrator credentials**

Provide credentials that can be used for administrator access to the workspace's SQL pools. If you don't provide a password, one will be automatically generated. You can change the password later.

Admin username \*

Password

Confirm password

**Managed virtual network**

Choose whether you want a Synapse-managed virtual network dedicated for your Azure Synapse workspace. [Learn more](#)

Enable managed virtual network (i)

# Private Endpoints

## Overview

Managed private endpoints are private endpoints created in the Managed workspace VNet establishing a private link to Azure resources. Managed private endpoints are only supported in Azure Synapse workspaces with a Managed workspace VNet.

## Benefits

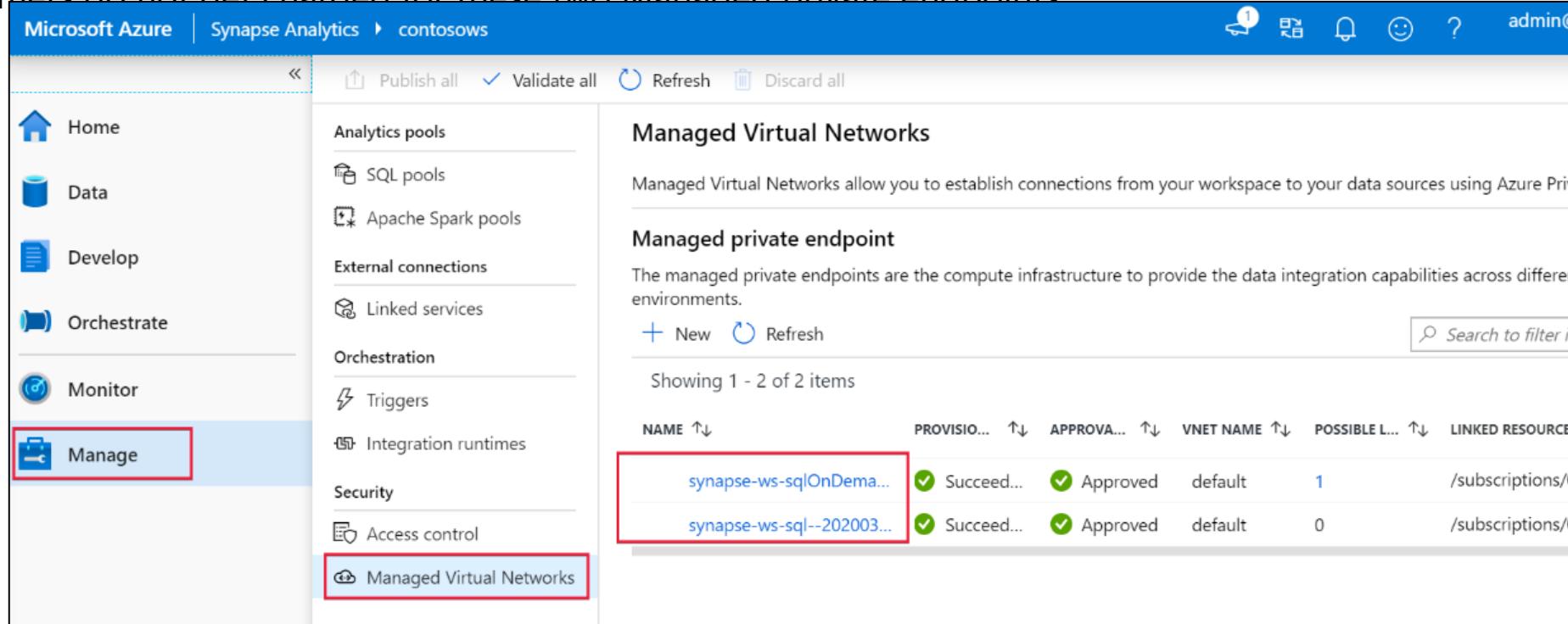
- Private link enables customers to access Azure services and Azure hosted customer/partner services from their Azure VNet securely.
- With use of private link, traffic between customer's VNet and workspace traverses entirely over the Microsoft backbone network.
- Private link protects against data exfiltration risks.
- Private endpoint uses a private IP address from customer's VNet to effectively bring the service into their VNet.
- Private endpoints are mapped to a specific resource in Azure and not the entire service.

# Private Endpoints for Synapse SQL (provisioned & serverless)

## Overview

Dedicated SQL pool and serverless SQL pool use multi-tenant infrastructure that is not deployed into the Managed workspace VNet.

Azure Synapse creates two managed private endpoints to dedicated SQL pool and serverless SQL pool in that workspace.  
~~Customers do not get charged for these two Managed private endpoints~~



The screenshot shows the Azure Synapse Analytics 'Manage' blade. The left sidebar has a 'Manage' item highlighted with a red box. At the bottom of the sidebar, there is another red box around the 'Managed Virtual Networks' link. The main content area is titled 'Managed Virtual Networks' and contains a table of managed private endpoints.

NAME ↑	PROVISIONING STATE ↑↓	APPROVAL STATE ↑↓	VNET NAME ↑↓	POSSIBLE LOCATIONS ↑↓	LINKED RESOURCE ID ↑↓
synapse-ws-sqlOnDemand... synapse-ws-sql--202003...	✓ Succeeded ✓ Succeeded	✓ Approved ✓ Approved	default	1 0	/subscriptions/0... /subscriptions/0...

# Data Exfiltration

Allow outbound data traffic over Synapse managed private endpoints to only approved tenants

**Create Synapse workspace** 

Configure networking settings for your workspace.

**Allow connections from all IP addresses**

 Azure Synapse Studio and other client tools will only be able to connect to the workspace endpoints if this setting is allowed. Connections from specific IP addresses or all Azure services can be allowed/disallowed after the workspace is provisioned.

Allow connections from all IP addresses to your workspace's endpoints. You can restrict this to just Azure datacenter IP addresses and/or specific IP address ranges after creating the workspace.

Allow connections from all IP addresses

**Managed virtual network**

Choose whether you want a Synapse-managed virtual network dedicated for your Azure Synapse workspace. [Learn more](#)

Enable managed virtual network 

Allow outbound data traffic only to approved targets 

Yes  No

 Private endpoints will be allowed to target resources in approved Azure AD tenants only. The Azure AD tenant of the current user will be included by default and is not listed below.

**Azure AD tenants**

 Add  Delete

Tenant name	Tenant id
No results to display	

**Review + create**   **Next: Tags >**

**Select Azure AD tenants** 

Select by Azure AD tenant name  
 Manually via tenant id

 The Azure AD tenant of the current user will be included by default and is not listed below.

**Tenants** 

tenant
AdventureWorks
Fabrikam
Tailspin Toys

Select all  
 AdventureWorks  
 Fabrikam  
 Tailspin Toys

**Select**

# Azure Active Directory authentication

## Overview

Manage user identities in one location.

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

## Benefits

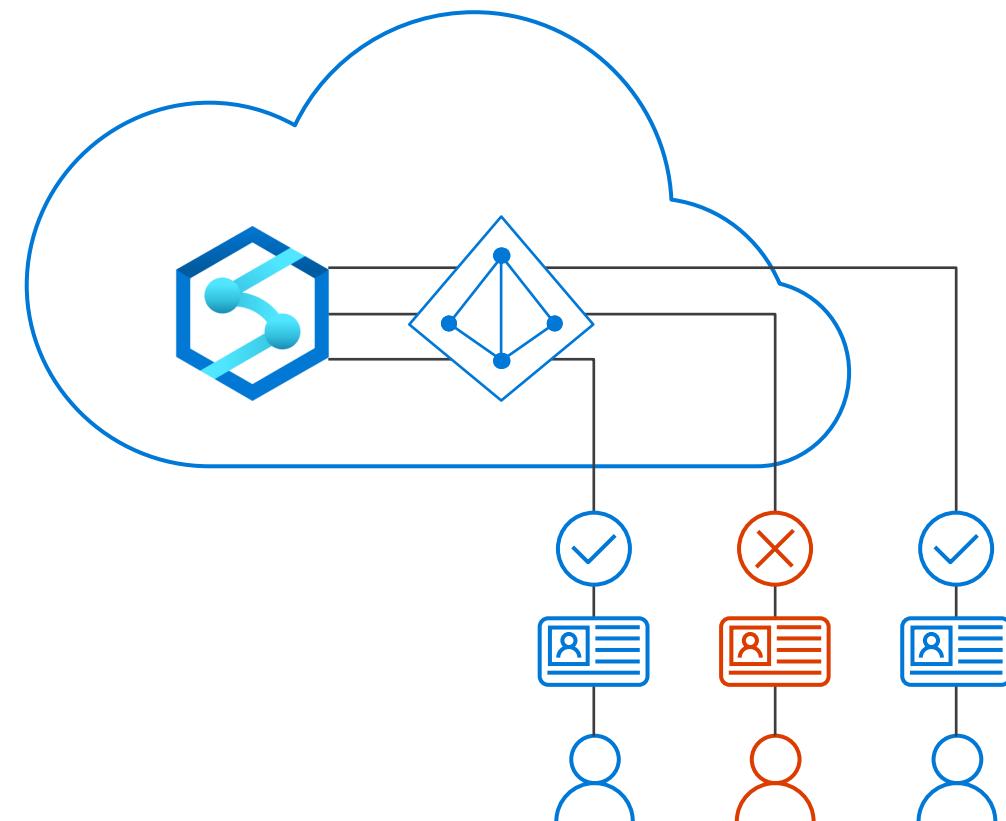
Enables management of database permissions by using external Azure Active Directory groups

Allows password rotation in a single place

Alternative to SQL Server authentication

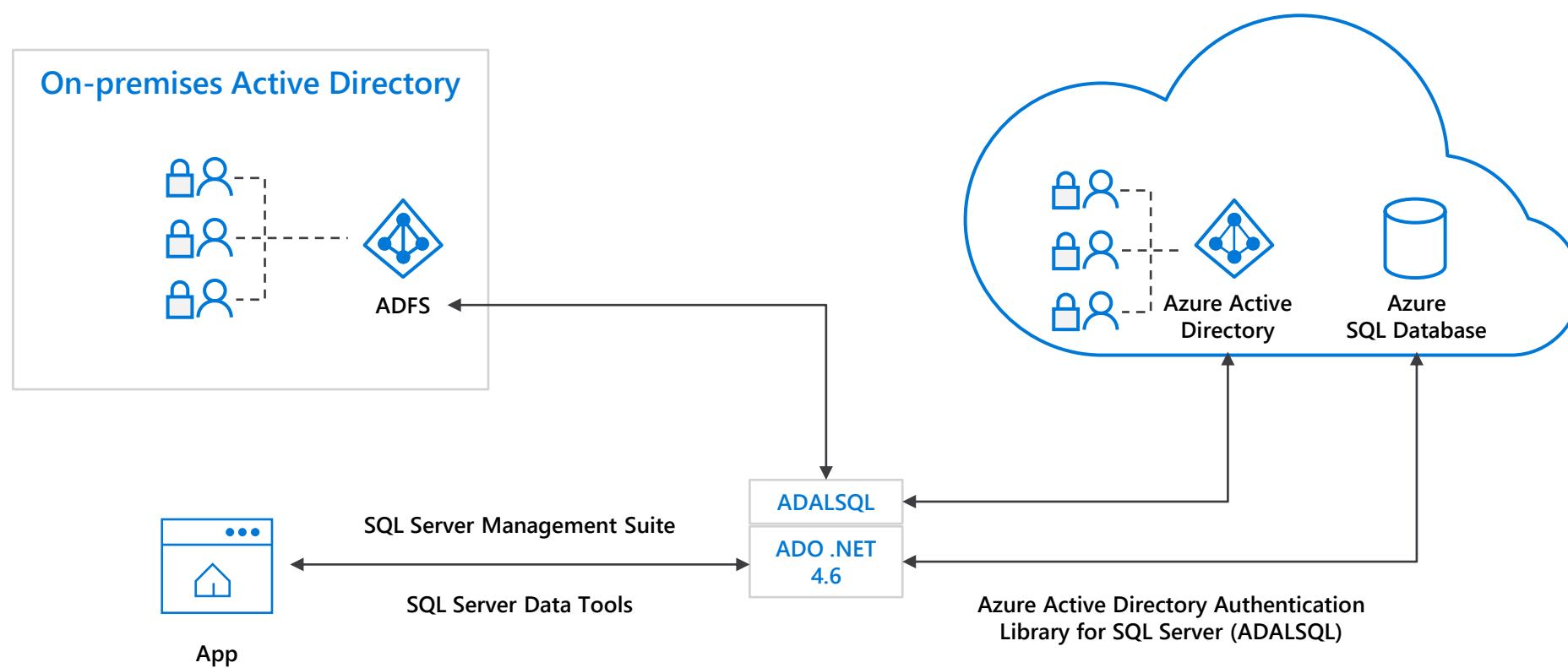
Eliminates the need to store passwords

## Azure Synapse Analytics



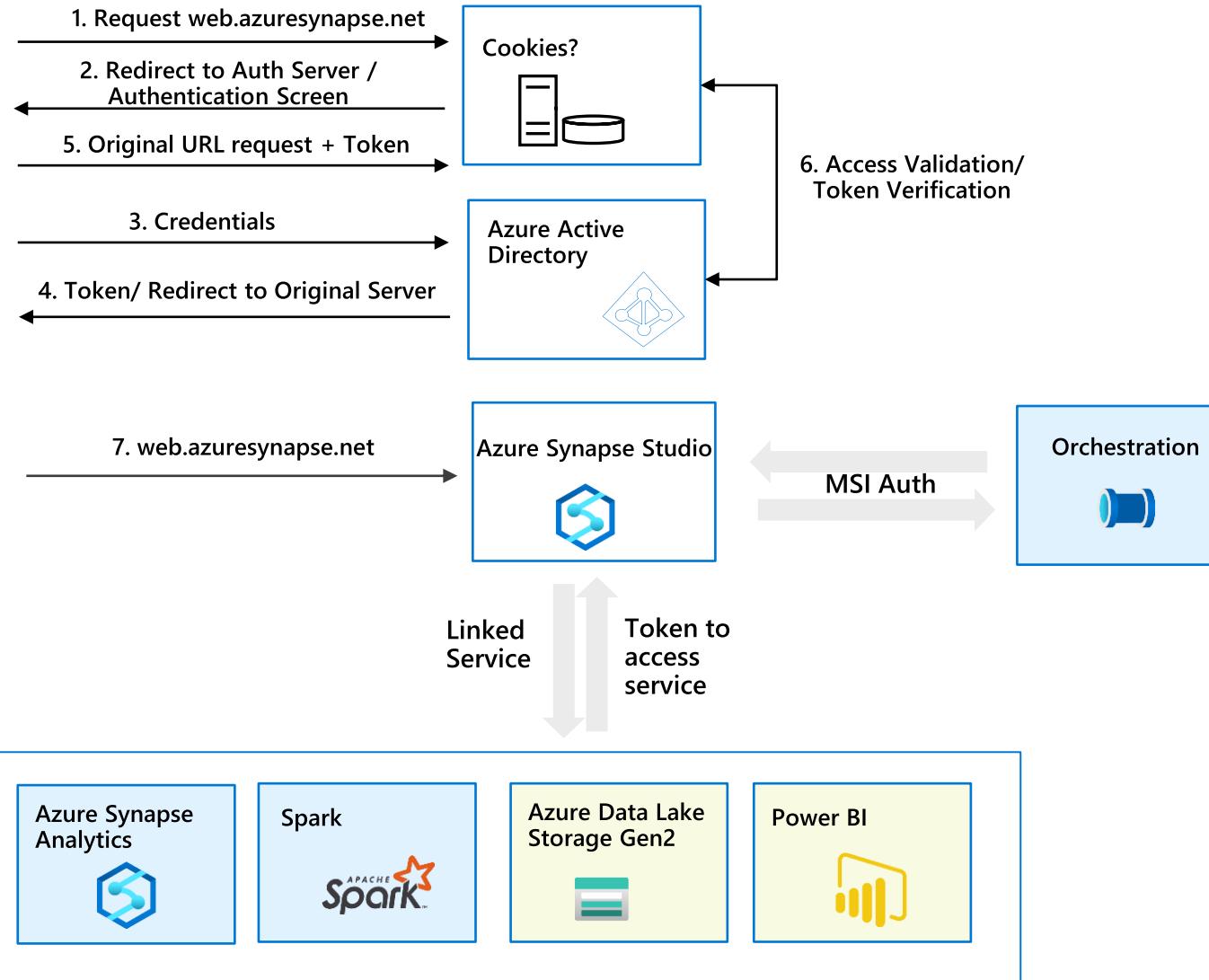
# Azure Active Directory trust architecture

## Azure Active Directory and Azure Synapse Analytics



# 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 workspace MSI auth for automation

# Access Control

## Overview

It provides access control management to workspace resources and artifacts

### Add role assignment

Grant others access to this workspace by assigning roles to users, groups, and/or service principals.  
[Learn more](#)

Scope \* ⓘ  
 Workspace  Workspace item

Role \* ⓘ  
Select a role  
 Filter...  
  
Synapse Administrator ⓘ  
Synapse SQL Administrator ⓘ  
Synapse Apache Spark Administrator ⓘ  
Synapse Contributor (preview) ⓘ  
Synapse Artifact Publisher (preview) ⓘ  
Synapse Artifact User (preview) ⓘ  
Synapse Compute Operator (preview) ⓘ  
Synapse Credential User (preview) ⓘ

Microsoft Azure | internalsandbox

» Publish all 1 ✓ Validate all ⏪ Refresh ⏹ Discard all

Analytics pools  
SQL pools  
Apache Spark pools  
External connections  
Linked services  
Orchestration  
Triggers  
Integration runtimes  
Security  
Access control

Access control  
Grant access to Synapse workspace and resources by assigning a role to a user, group, service principal, or application.

+ Add ⏪ Refresh ⏹ Remove access

Showing 1 - 3 of 3 items

NAME ↑↓	TYP ↑↓	ROLE
analytics1	Apache Spark pools	Synapse Compute Operator (preview)
analytics1	Apache Spark pools	Synapse Administrator
analytics1	Apache Spark pools	Synapse Contributor (preview)

**Add role assignment**

Grant others access to this workspace by assigning roles to users, groups, and/or service principals.  
[Learn more](#)

Scope \* ⓘ  
 Workspace  Workspace item

Item type \*  
Apache Spark pools

Item \*  
analytics1

Role \* ⓘ  
Synapse Compute Operator (preview)  
 Filter...  
Synapse Administrator ⓘ  
Synapse Contributor (preview) ⓘ  
Synapse Compute Operator (preview) ⓘ

# Synapse RBAC Roles and permitted actions

Action	Synapse Administrator	Synapse Contributor	Synapse Artifact Author	Synapse Artifact Reader	Synapse Compute Manager	Synapse Credential User	Synapse Managed Private Endpoint Administrator	Synapse Reader
workspaces/read	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
workspaces/roleAssignments/write, delete	Yes							
workspaces/managedPrivateEndpoint/write, delete	Yes							Yes
workspaces/bigDataPools/useCompute/action	Yes	Yes			Yes			
workspaces/bigDataPools/viewLogs/action	Yes	Yes			Yes			
workspaces/integrationRuntimes/useCompute/action	Yes	Yes			Yes			
workspaces/integrationRuntimes/viewLogs/action	Yes	Yes			Yes			
workspaces/artifacts/read	Yes	Yes	Yes	Yes				
workspaces/notebooks/write, delete	Yes	Yes	Yes					
workspaces/sparkJobDefinitions/write, delete	Yes	Yes	Yes					
workspaces/sqlScripts/write, delete	Yes	Yes	Yes					
workspaces/dataFlows/write, delete	Yes	Yes	Yes					
workspaces/pipelines/write, delete	Yes	Yes	Yes					
workspaces/triggers/write, delete	Yes	Yes	Yes					
workspaces/datasets/write, delete	Yes	Yes	Yes					
workspaces/libraries/write, delete	Yes	Yes	Yes					
workspaces/linkedServices/write, delete	Yes	Yes	Yes					
workspaces/credentials/write, delete	Yes	Yes	Yes					
workspaces/notebooks/viewOutputs/action	Yes	Yes						
workspaces/pipelines/viewOutputs/action	Yes	Yes						
workspaces/linkedServices/useSecret/action	Yes					Yes		
workspaces/credentials/useSecret/action	Yes					Yes		

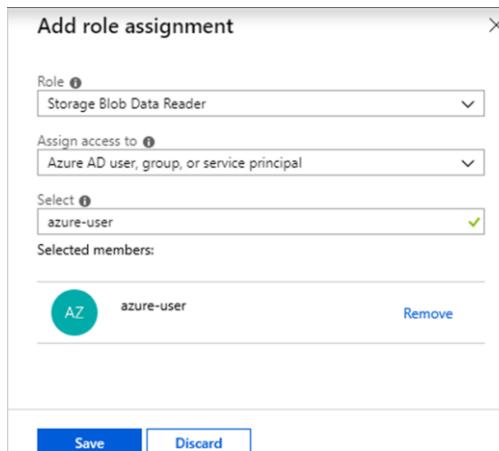
# Access control – serverless SQL pool

## Overview

Enterprise-grade security model enables you to control who can access data.

## Benefits

- Use Azure Active Directory users or native SQL logins.
- SAS tokens, AAD or workspace identity access
- Specify access methods in credential
- Grant access to storage by referencing storage credential
- Enable some logins to access external tables
- Add AAD role assignments directly on Azure storage.



```
--Create Login to a single serverless SQL pool database
CREATE LOGIN [alias@domain.com] FROM EXTERNAL PROVIDER;

-- create user under that login
use yourdb -- Use your DB name
go
CREATE USER alias FROM LOGIN [alias@domain.com];

To grant full access to a user to all serverless SQL pool databases
CREATE LOGIN [alias@domain.com] FROM EXTERNAL PROVIDER;
ALTER SERVER ROLE sysadmin ADD MEMBER [alias@domain.com];

-- enable impersonation using workspace Managed Identity
CREATE CREDENTIAL [ManagedIdentity]
WITH IDENTITY = 'Managed Identity'

-- enable access to specified storage using SAS token
CREATE CREDENTIAL [https://XXX.blob.core.windows.net/csv]
WITH IDENTITY = 'SHARED ACCESS SIGNATURE',
SECRET = 'sv=2014-02-
14&sr=b&si=TestPolicy&sig=o%2B5%2FOC%2BLm7tWWft'

-- grant login1 to use SAS token defined in credential for storage account
GRANT REFERENCES CREDENTIAL::[https://XXX.blob.core.windows.net/csv]
TO LOGIN = 'login1'

-- grant login2 to use Managed Identity
GRANT REFERENCES CREDENTIAL::[ManagedIdentity]
TO LOGIN = 'login2'

-- grant login2 to select external data via table
GRANT SELECT ON OBJECT::[dbo.population] TO LOGIN = 'login2'
```

**Industry-leading security  
and compliance**

# 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



CJIS



IRS 1075 / ITAR-ready



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

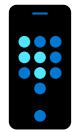


Japan Financial Services



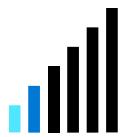
ENISA IAF

# Threat Protection - Business requirements



**How do we enumerate and track potential SQL vulnerabilities?**

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



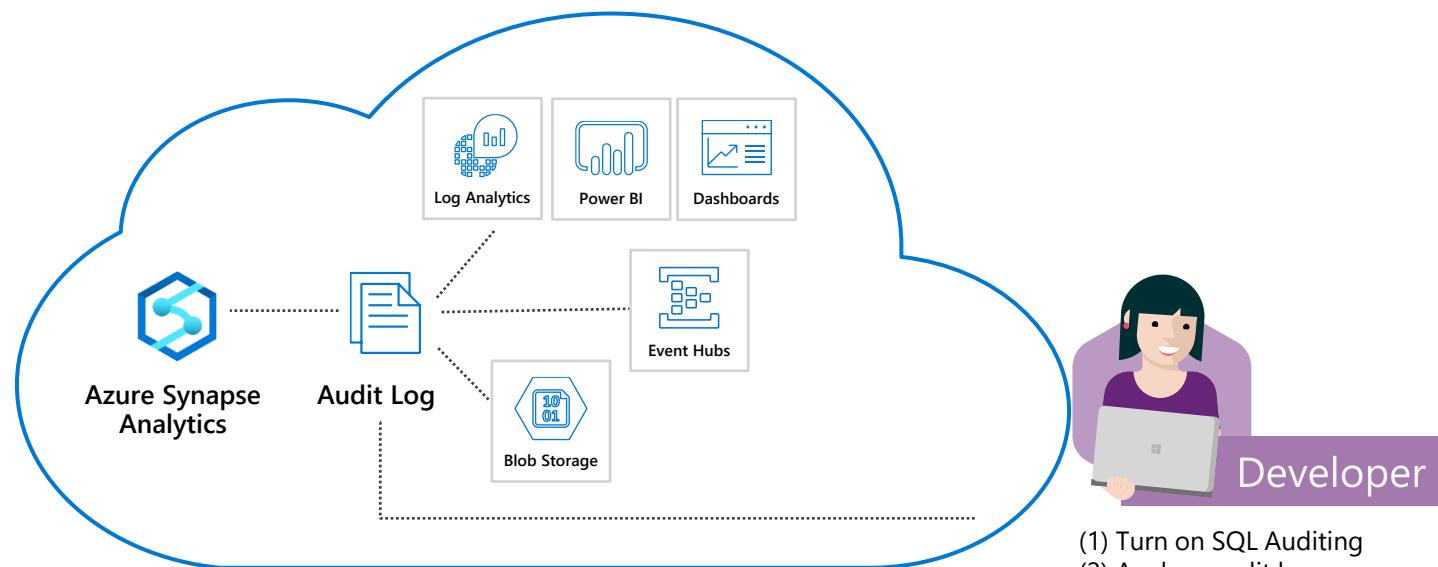
**How do we discover and alert on suspicious database activity?**

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



# SQL auditing in Azure Log Analytics and Event Hubs

Gain insight into database audit log



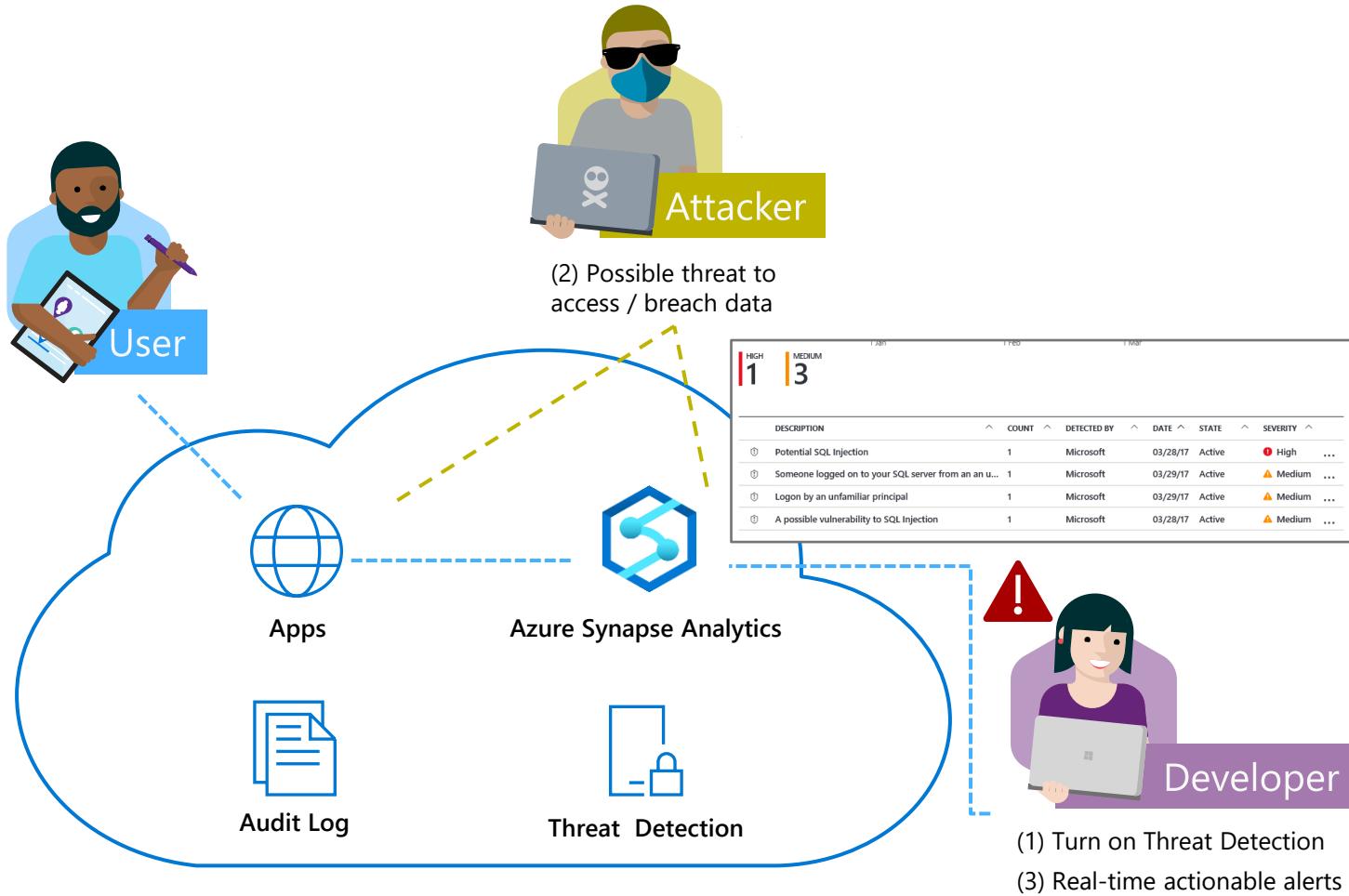
The screenshot shows the Azure Log Analytics interface with the search bar containing the query: `search * | where Category == "SQLSecurityAuditEvents" | project TimeGenerated, server_principal_name_s, statement_s, affected_rows_d, SeverityLevel | sort by TimeGenerated asc`. The results table displays 62 audit log entries. One specific entry is highlighted in blue:

TimeGenerated	server_principal_name_s	statement_s	affected_rows_d	SeverityLevel
8/15/2018 12:00:22.521 AM	admin1	exec sp_executesql N'SELECT t1.name AS [Name], SCHEMA_NAME(t1...	0	
8/15/2018 12:00:22.521 AM	admin1	exec sp_executesql N'SELECT ISNULL(HAS_PERMS_BY_NAME('QUOTEEN...)	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	exec sp_executesql N'SELECT CAST(t1.is_enabled AS bit) AS [isEnabled]...	0	
8/15/2018 12:00:22.521 AM	admin1	IF OBJECT_ID ('[sys].[database_query_store_options]') IS NOT NULL BE...	2	

- ✓ 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

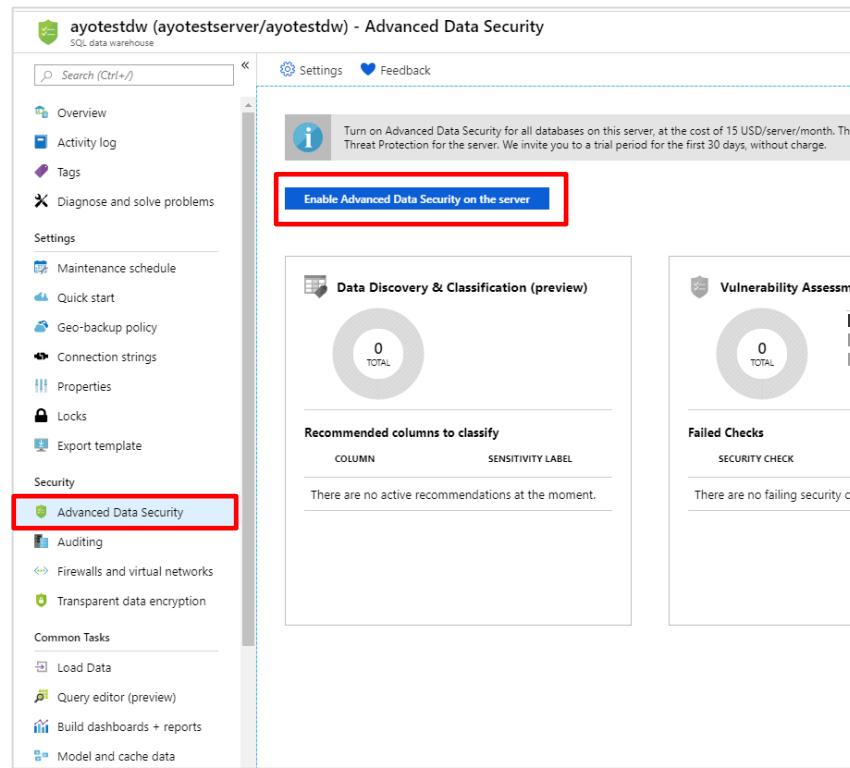
The screenshot displays two windows from the Azure portal:

- Top Window (Overview):** Shows summary statistics: Classified columns (10 / 109), Tables containing sensitive data (4 / 12), Unique information types (4). It includes donut charts for Label distribution (10 columns) and Information type distribution (CONTACT INFO: NAME, CREDENTIALS, FINANCIAL).
- Bottom Window (Settings - Information protection):** Shows a list of sensitivity labels with their descriptions. Labels include Public, General, Confidential, Confidential - GDPR, Highly confidential, and Highly confidential - GDPR.

- ✓ 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

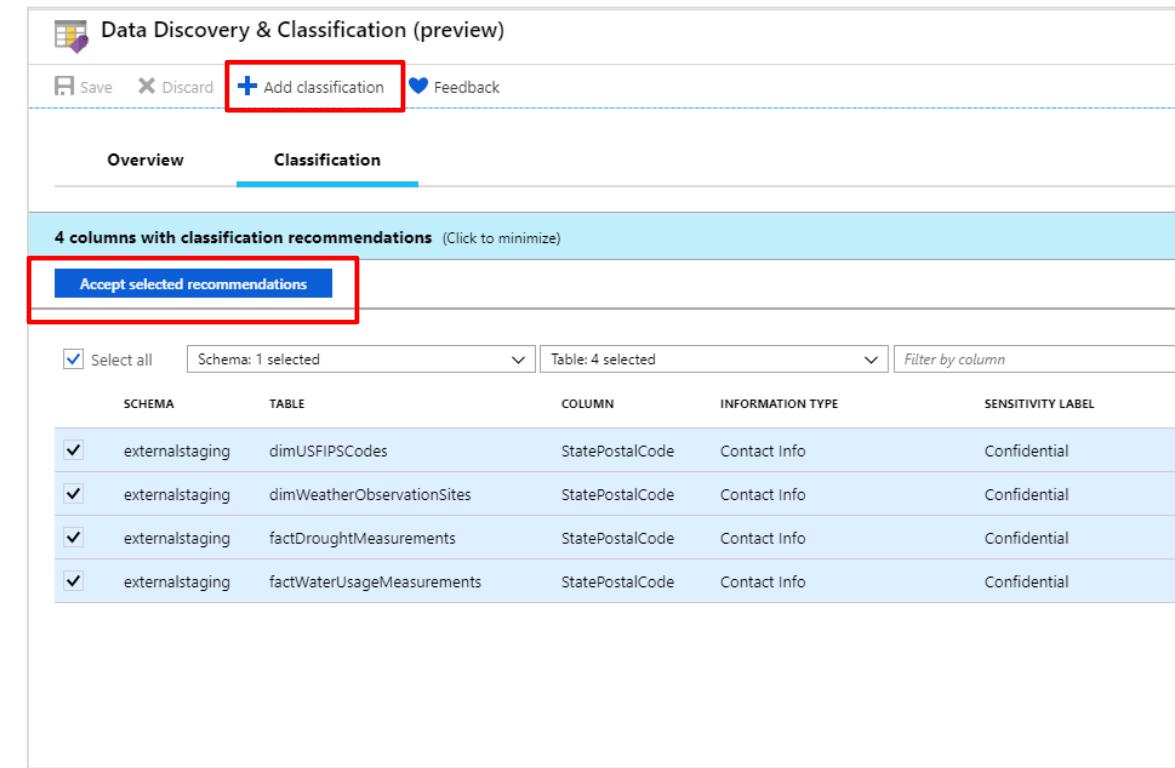
# SQL Data Discovery & Classification - setup

## Step 1: Enable Advanced Data Security on the logical SQL Server



The screenshot shows the 'ayotestdw (ayotestserver/ayotestdw) - Advanced Data Security' blade. On the left, there's a sidebar with various navigation items like Overview, Activity log, Tags, and Diagnose and solve problems. Under the 'Security' section, 'Advanced Data Security' is selected and highlighted with a red box. The main area has a heading 'Data Discovery & Classification (preview)' with a '0 TOTAL' count. Below it, there's a section for 'Recommended columns to classify' which says 'There are no active recommendations at the moment.' To the right, there's a 'Vulnerability Assessment' section with a '0 TOTAL' count and a note about failing security checks.

## Step 2: Use recommendations and/or manual classification to classify all the sensitive columns in your tables



The screenshot shows the 'Data Discovery & Classification (preview)' blade. At the top, there are 'Save' and 'Discard' buttons, and a prominent 'Add classification' button highlighted with a red box. Below that, there are two tabs: 'Overview' and 'Classification', with 'Classification' being the active tab. A message indicates '4 columns with classification recommendations' (Click to minimize). Below this, there's a large blue button labeled 'Accept selected recommendations' highlighted with a red box. At the bottom, there's a table with the following data:

SCHEMA	TABLE	COLUMN	INFORMATION TYPE	SENSITIVITY LABEL	
<input checked="" type="checkbox"/>	externalstaging	dimUSFIPSCodes	StatePostalCode	Contact Info	Confidential
<input checked="" type="checkbox"/>	externalstaging	dimWeatherObservationSites	StatePostalCode	Contact Info	Confidential
<input checked="" type="checkbox"/>	externalstaging	factDroughtMeasurements	StatePostalCode	Contact Info	Confidential
<input checked="" type="checkbox"/>	externalstaging	factWaterUsageMeasurements	StatePostalCode	Contact Info	Confidential

# 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 'ayotestdw (ayotestserver/ayotestdw) - Auditing' page. Under the 'Auditing' section, the 'Auditing' switch is set to 'ON'. Below it, under 'Audit log destination (choose at least one)', the 'Storage' checkbox is checked, and the storage details 'sqlvasdgoqfss5cyls' are listed. Other options like 'Log Analytics (Preview)' and 'Event Hub (Preview)' are shown but not selected.

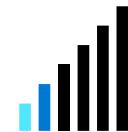
**Step 2:** Navigate to audit logs in storage account and download 'xel' log files to local machine.

The screenshot shows the 'sqldbauditlogs' container in the Storage account. It displays a list of blobs, with one blob named '01\_34\_30\_090\_0.xel' selected and highlighted with a red box. The blob's properties are shown below, including its name, modified date (4/1/2019, 6:34:31 PM), size (7.5 KiB), and lease state (Available).

**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 list of 24785 events. The columns include name, timestamp, affected\_rows, application\_name, client\_ip, data\_sensitivity\_information, and database\_name. One specific event is expanded, showing detailed information such as action\_id, additional\_information, affected\_rows, application\_name, audit\_schema\_version, class\_type, client\_ip, connection\_id, data\_sensitivity\_information, database\_name, database\_principal\_id, database\_principal\_name, duration\_milliseconds, event\_time, host\_name, is\_column\_permission, object\_id, and object\_name. The 'data\_sensitivity\_information' field is highlighted with a red box in both the list and the details view.

# Network Security - Business requirements

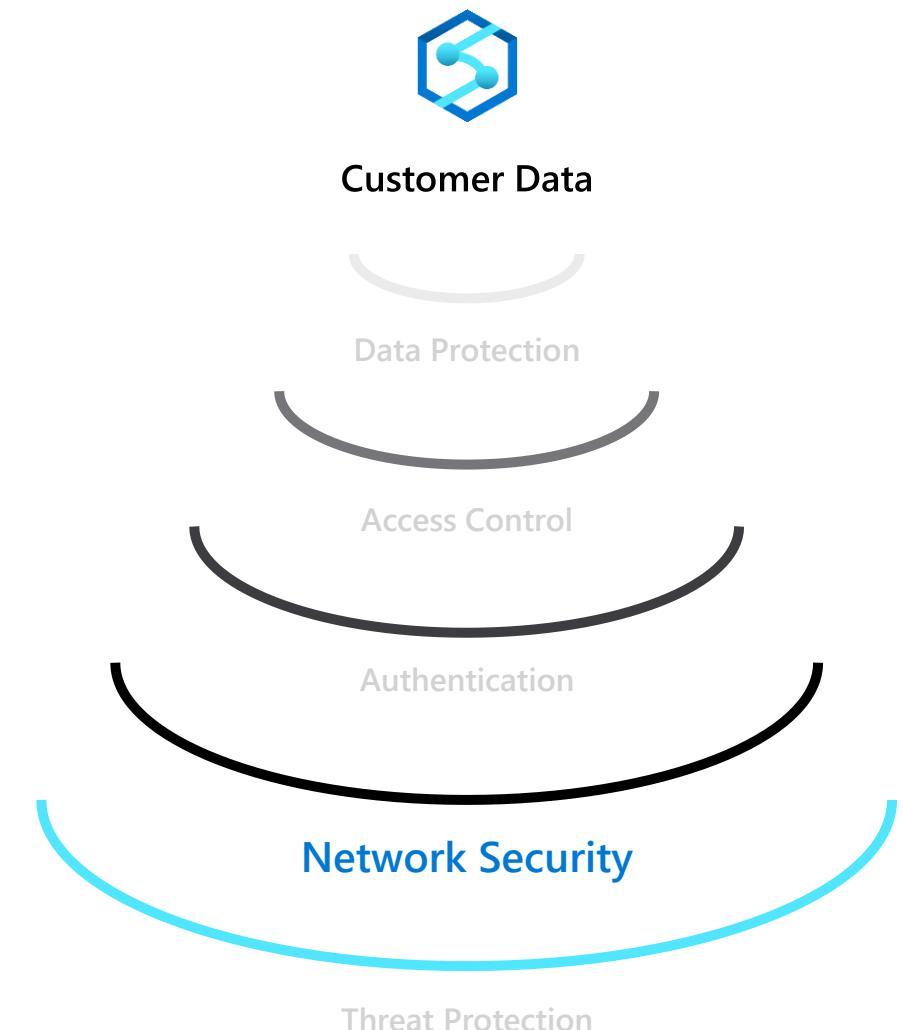


## How do we implement network isolation?

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.

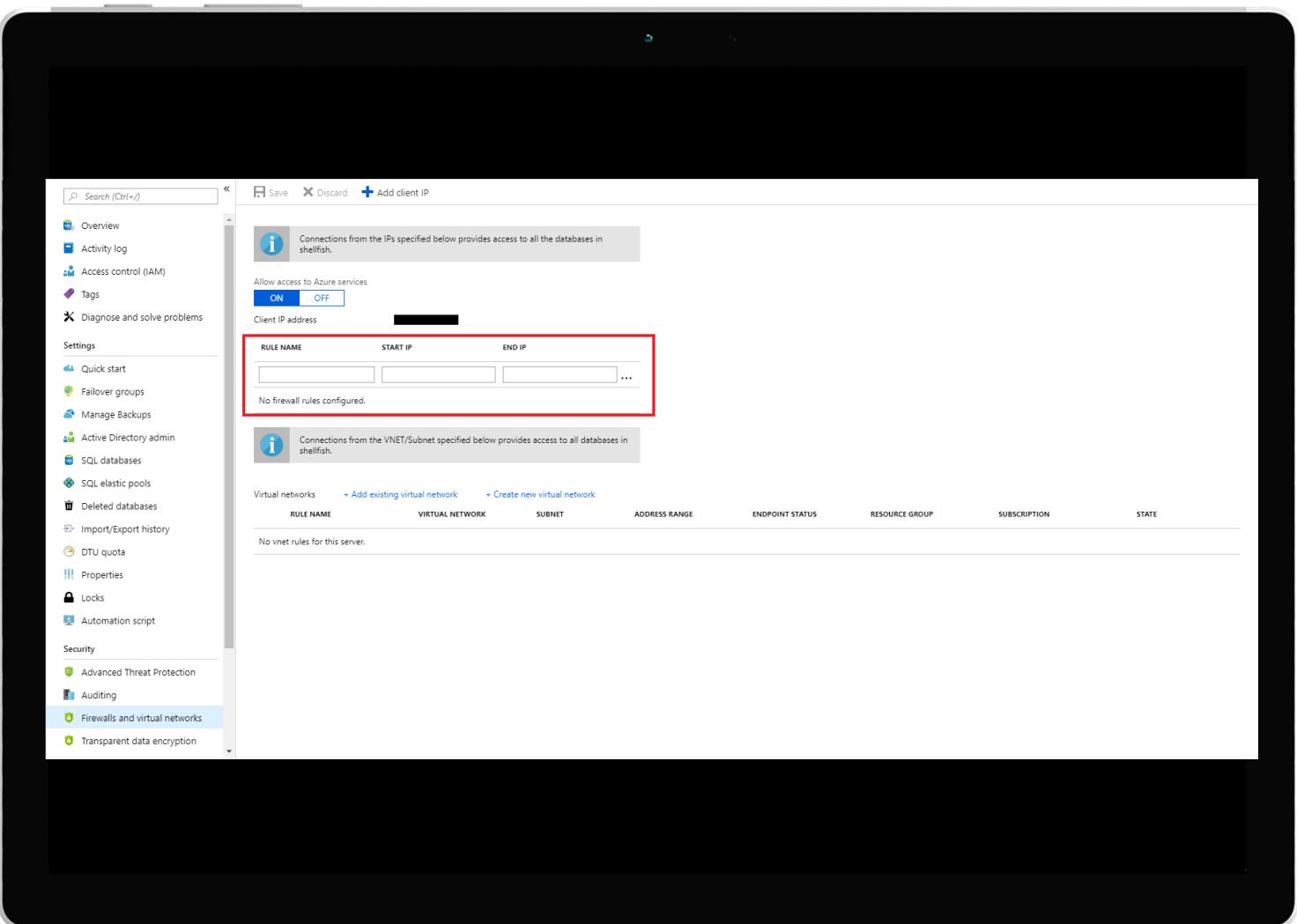


# Firewall configuration on the portal

By default, Azure blocks all external connections to port 1433

Configure with the following steps:

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



# Firewall configuration using REST API

Managing firewall rules through REST API must be authenticated.

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

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

To create or update a server-level firewall rule, execute the [PUT](#) method.

To remove an existing server-level firewall rule, execute the [DELETE](#) method.

To list firewall rules, execute the [GET](#).

PUT

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

DELETE

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

GET

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

# Firewall configuration using PowerShell/T-SQL

## Windows PowerShell Azure cmdlets

```
New-AzureRmSqlServerFirewallRule
```

```
Get-AzureRmSqlServerFirewallRule
```

```
Set-AzureRmSqlServerFirewallRule
```

## Transact SQL

```
sp_set_firewall_rule
```

```
sp_delete_firewall_rule
```

```
# PS Allow external IP access to SQL DW  
PS C:\> New-AzureRmSqlServerFirewallRule  
    -ResourceGroupName "myResourceGroup" `  
    -ServerName $servername `  
    -FirewallRuleName "AllowSome" `  
    -StartIpAddress "0.0.0.0" `  
    -EndIpAddress "0.0.0.0"  
  
-- T-SQL Allow external IP access to SQL DW  
EXECUTE sp_set_firewall_rule  
    @name = N'ContosoFirewallRule',  
    @start_ip_address = '192.168.1.1',  
    @end_ip_address = '192.168.1.10'
```

# Authentication - Business requirements

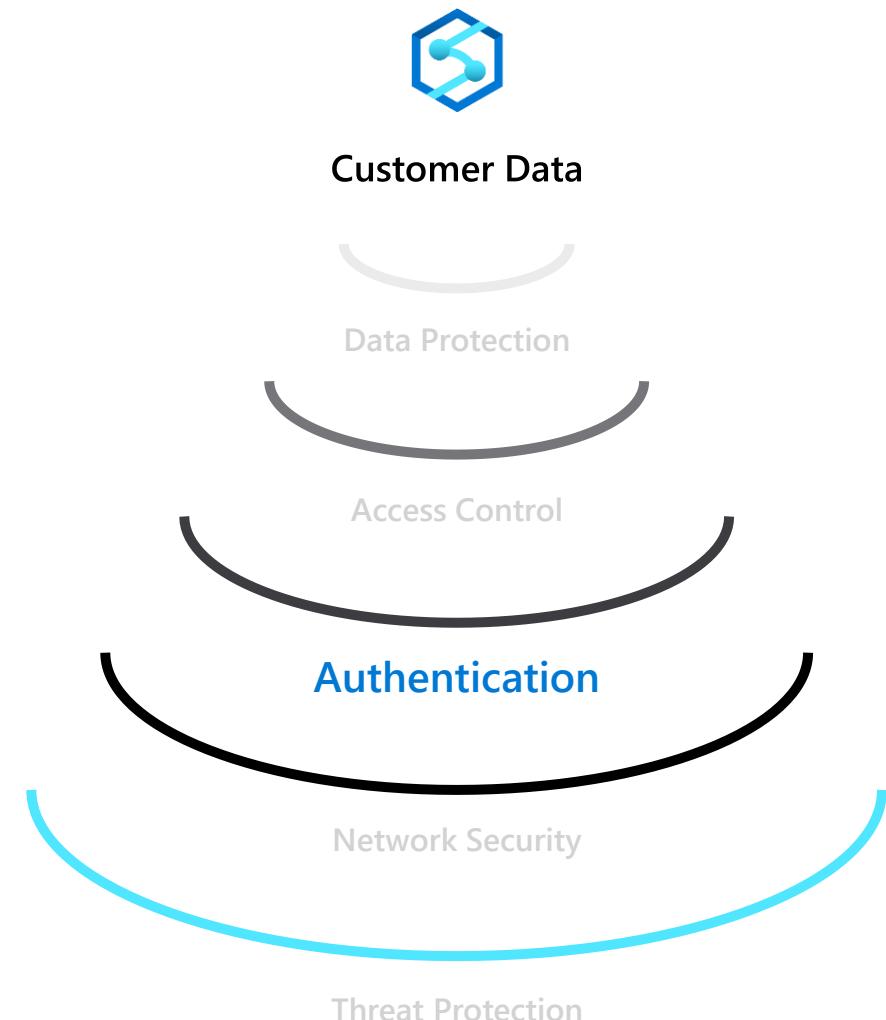


**How do I configure Azure Active Directory with Azure Synapse Analytics?**

I want additional control in the form of multi-factor authentication



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



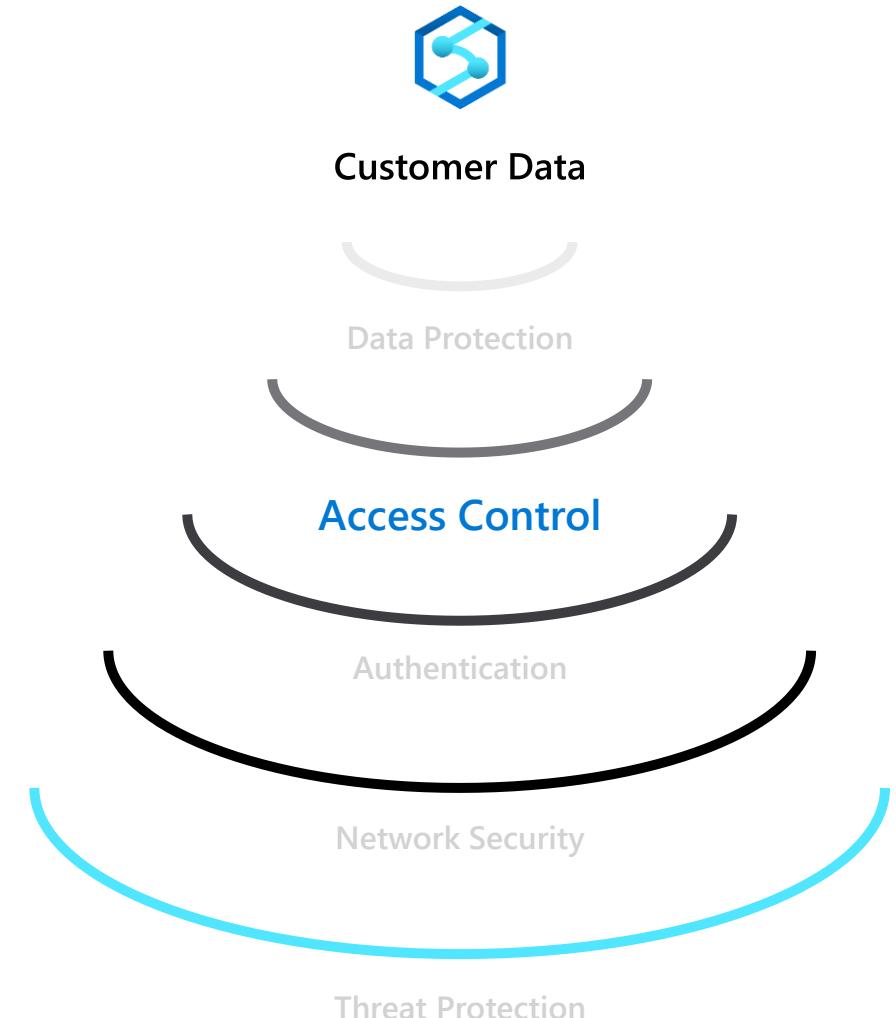
# Access Control - Business requirements



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

**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.



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

## Overview

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

Prevent unauthorized queries against certain tables.

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

```
-- Grant SELECT permission to user RosaQdM on table Person.Address in the AdventureWorks2012 database
GRANT SELECT ON OBJECT::Person.Address TO RosaQdM;
GO

-- Grant REFERENCES permission on column BusinessEntityID in view HumanResources.vEmployee to user Wanida
GRANT REFERENCES(BusinessEntityID) ON OBJECT::HumanResources.vEmployee TO Wanida WITH GRANT OPTION;
GO

-- Grant EXECUTE permission on stored procedure HumanResources.uspUpdateEmployeeHireInfo to an application role called Recruiting11
USE AdventureWorks2012;
GRANT EXECUTE ON OBJECT::HumanResources.uspUpdateEmployeeHireInfo TO RECRUITING 11;
GO
```

# Row-level security (RLS)

## Overview

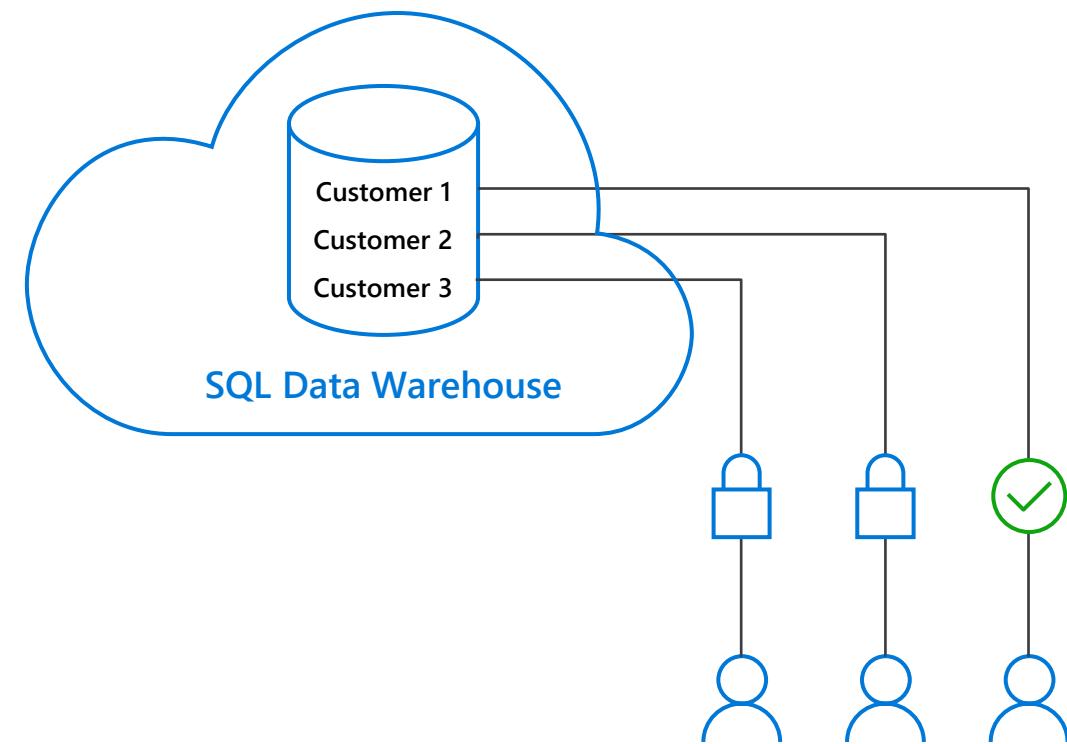
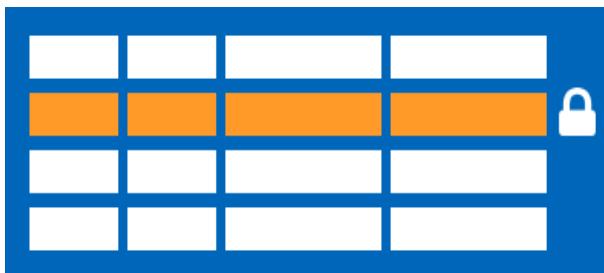
Fine grained access control of specific rows in a database table.

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

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

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

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



# Row-level security

## Creating policies

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

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

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

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

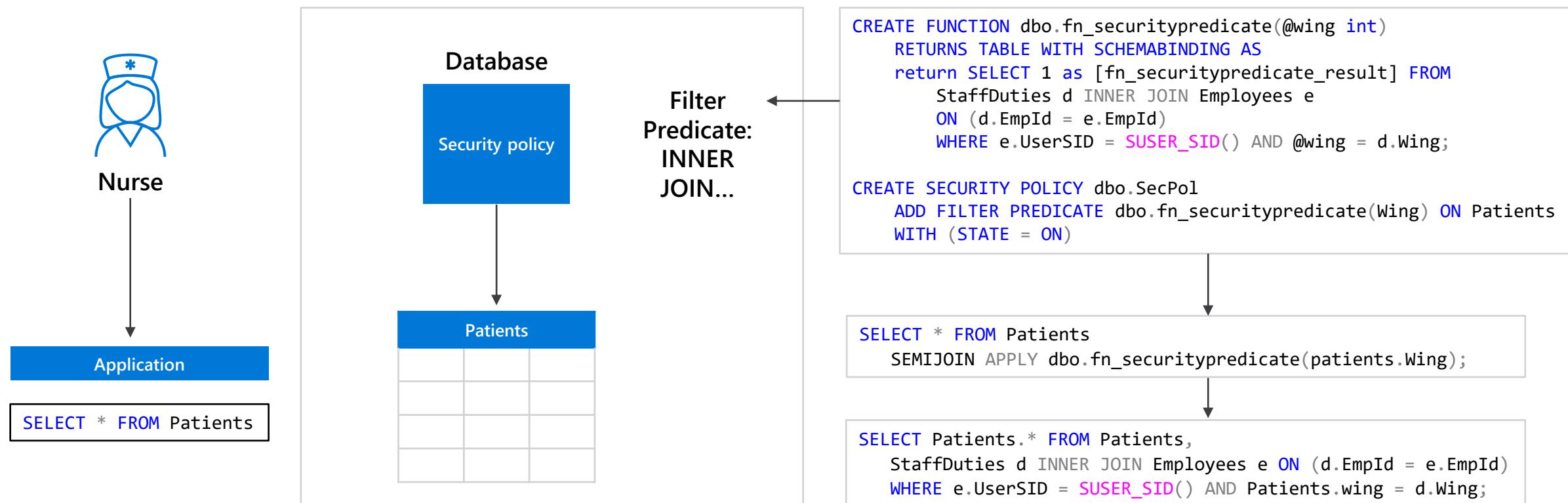
# Row-level security

## Three steps:

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



Policy manager



# Column-level security

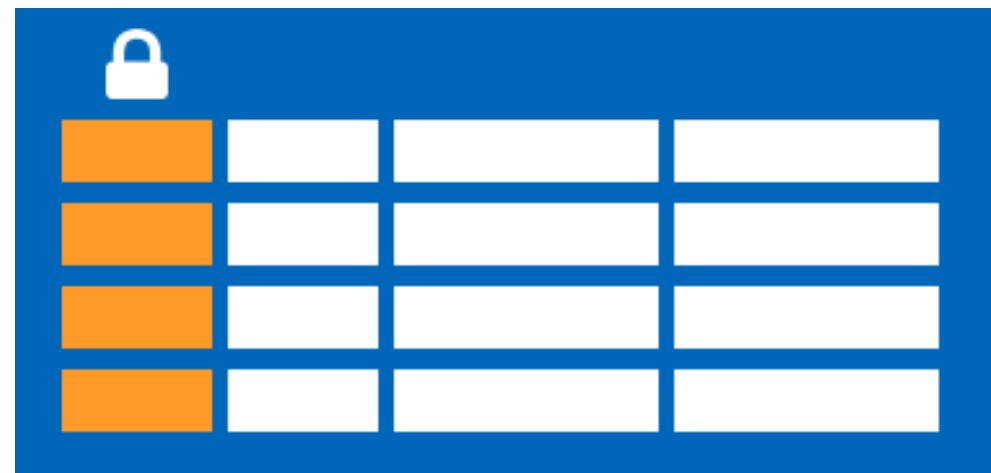
## Overview

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.

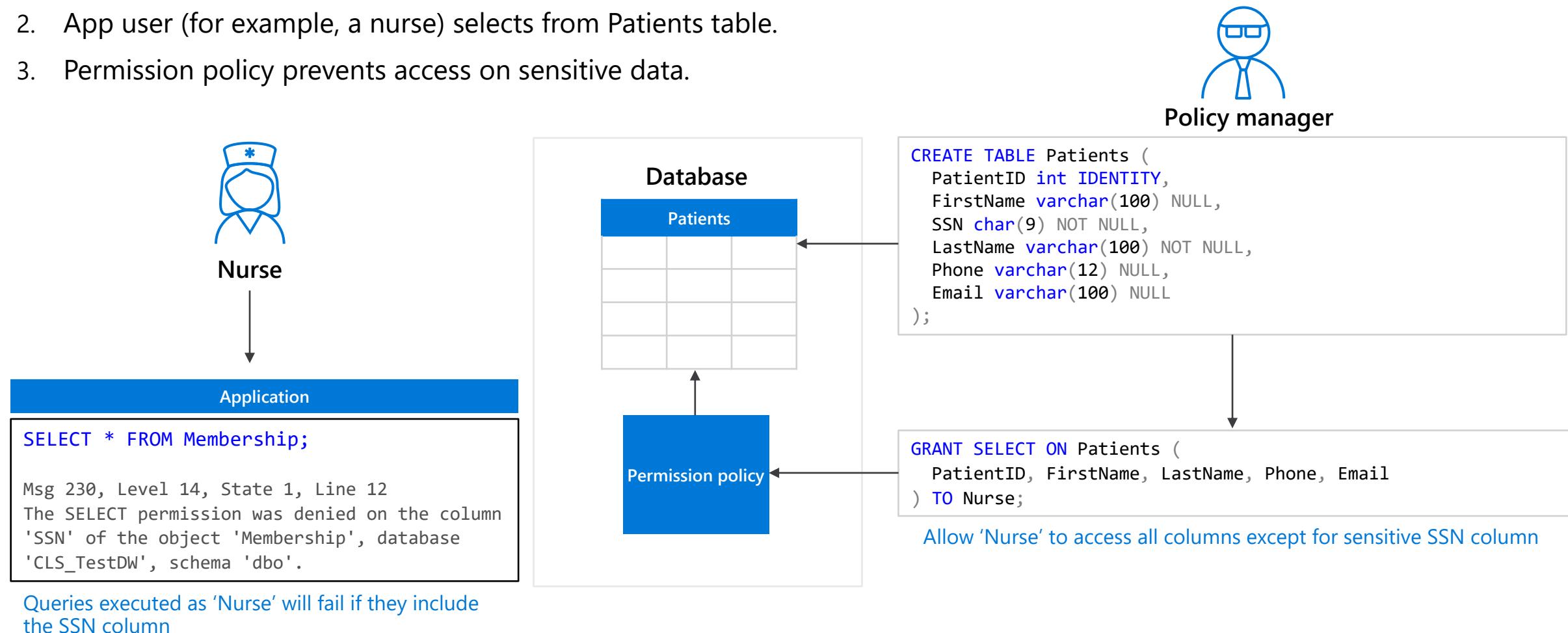
Both Azure Active Directory (AAD) and SQL authentication are supported.



# 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.



# Data Protection - Business requirements



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

What key management options do I have?



# Dynamic Data Masking

## Overview

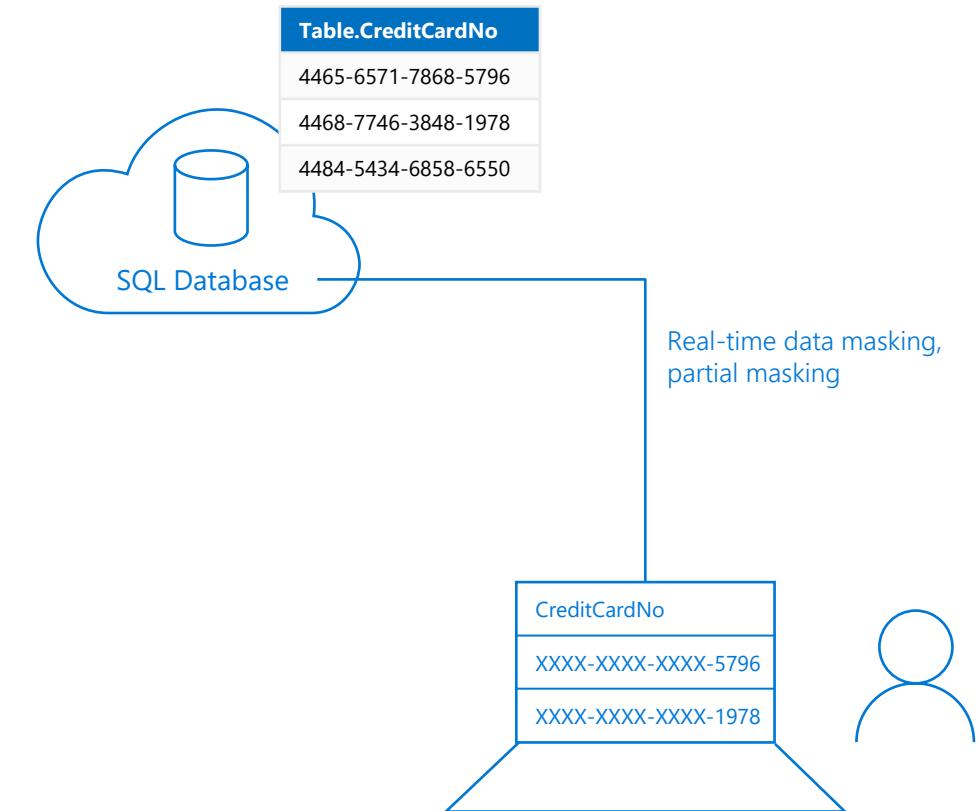
Prevent abuse of sensitive data by hiding it from users

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.)



# Column Level Encryption

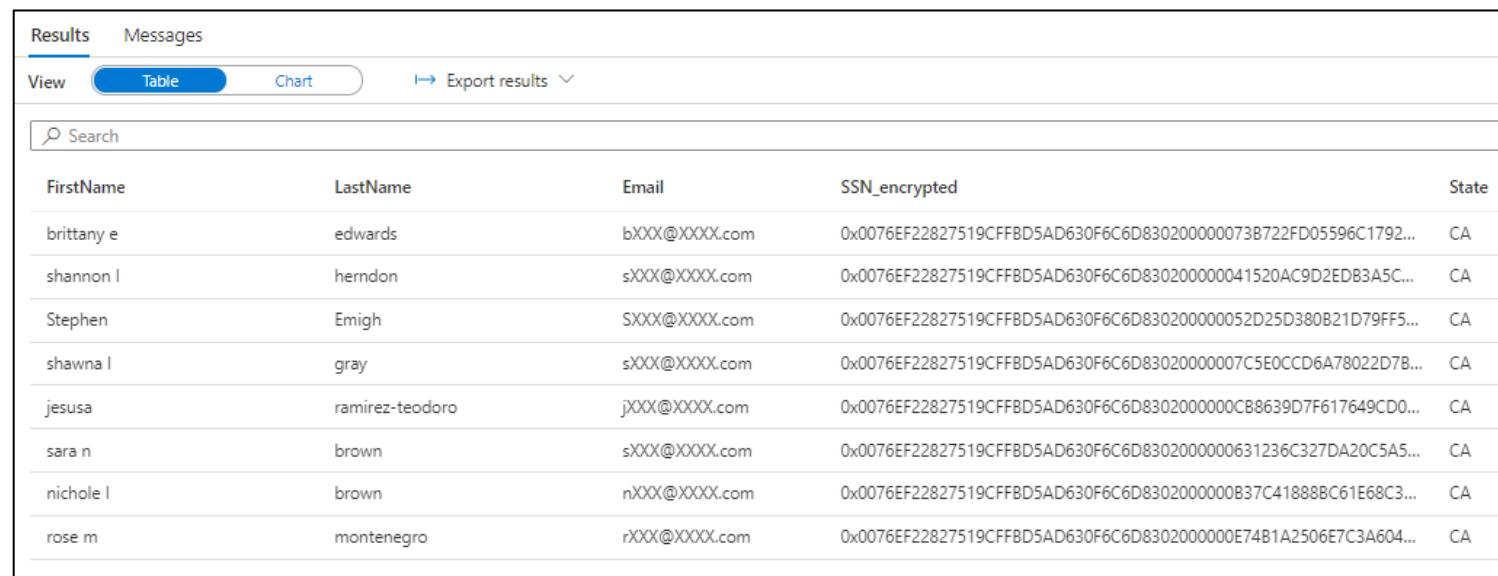
## Overview

It helps to implement fine-grained protection of sensitive data within a table in dedicated SQL pool.

The data in CLE enforced columns is encrypted on disk.  
User need to use DECRYPTBYKEY function to decrypt it.

5 step process to set up CLE

1. Create master key
2. Create certificate
3. Configure symmetric key for encryption
4. Encrypt the column data
5. Close symmetric key



The screenshot shows a SQL Server Management Studio (SSMS) results grid. The title bar says "Results" and "Messages". Below that is a toolbar with "View", "Table" (which is selected), and "Chart". There is also an "Export results" button. A search bar is present. The main area is a table with the following data:

FirstName	LastName	Email	SSN_encrypted	State
brittany e	edwards	bXXX@XXXX.com	0x0076EF22827519CFFBD5AD630F6C6D830200000073B722FD05596C1792...	CA
shannon l	herndon	sXXX@XXXX.com	0x0076EF22827519CFFBD5AD630F6C6D830200000041520AC9D2EDB3A5C...	CA
Stephen	Emigh	SXXX@XXXX.com	0x0076EF22827519CFFBD5AD630F6C6D830200000052D25D380B21D79FF5...	CA
shawna l	gray	sXXX@XXXX.com	0x0076EF22827519CFFBD5AD630F6C6D83020000007C5E0CCD6A78022D7B...	CA
jesusa	ramirez-teodoro	jXXX@XXXX.com	0x0076EF22827519CFFBD5AD630F6C6D8302000000CB8639D7F617649CD0...	CA
sara n	brown	sXXX@XXXX.com	0x0076EF22827519CFFBD5AD630F6C6D8302000000631236C327DA20C5A...	CA
nichole l	brown	nXXX@XXXX.com	0x0076EF22827519CFFBD5AD630F6C6D8302000000B37C41888BC61E68C3...	CA
rose m	montenegro	rXXX@XXXX.com	0x0076EF22827519CFFBD5AD630F6C6D8302000000E74B1A2506E7C3A604...	CA

# 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

```

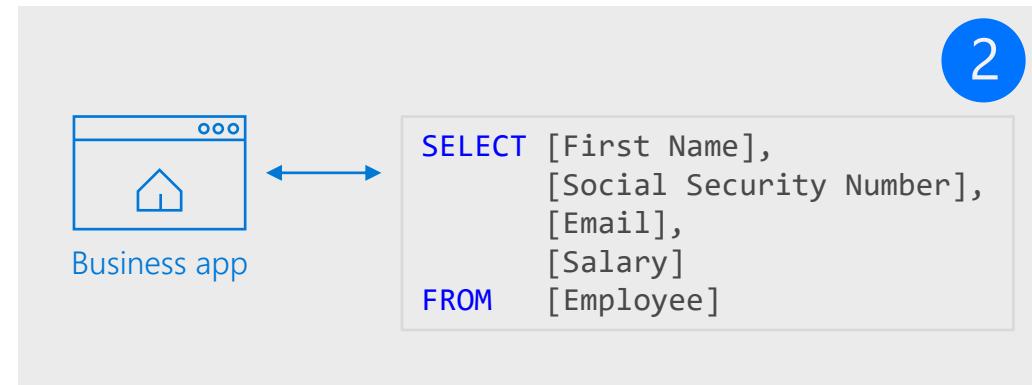
ALTER TABLE [Employee]
ALTER COLUMN [SocialSecurityNumber]
ADD MASKED WITH (FUNCTION = 'DEFAULT()')

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

Diagram illustrating Step 3:

	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

Non-masked data (admin login)

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

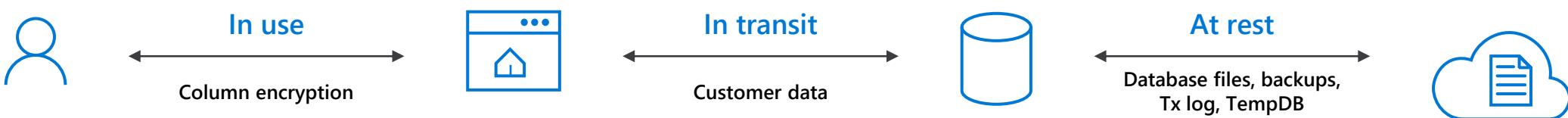
Masked data (admin1 login)

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

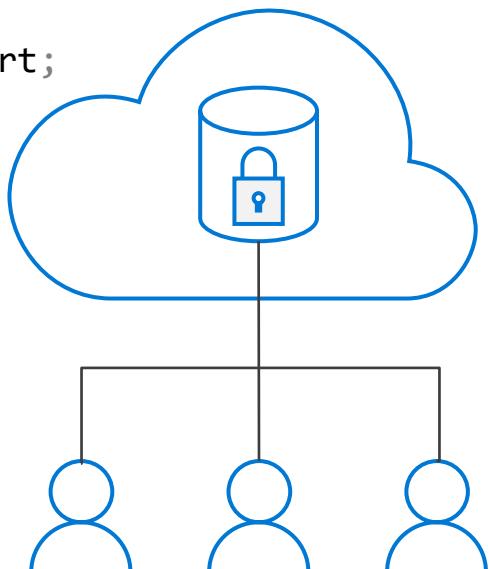
Service OR User managed keys.

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.

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



# Transparent data encryption (TDE)

## Key Vault

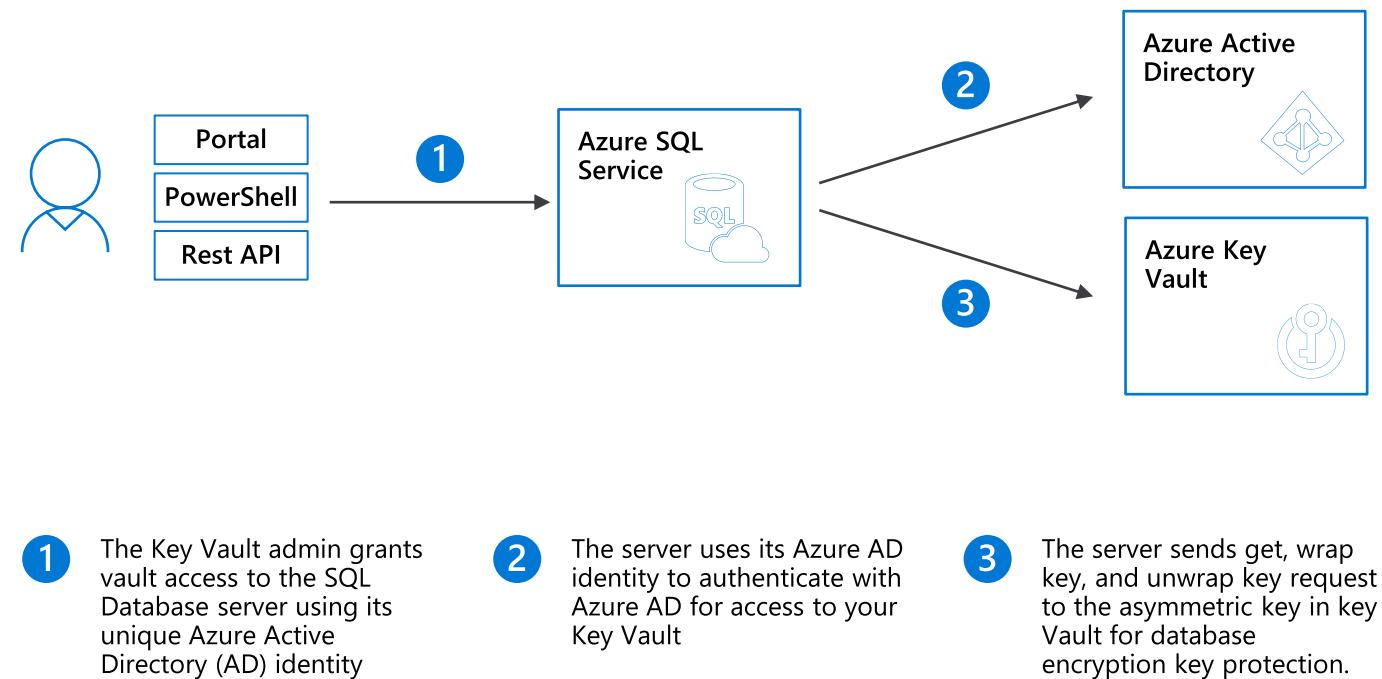
### Benefits with User Managed Keys

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

Highly available and scalable cloud-based key store.

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

Configurable via Azure Portal, PowerShell, and REST API.





# Azure Synapse Analytics

## Metastore

# Metastore

## Overview

It offers the different computational engines of a workspace to share databases and Parquet-backed tables between its Apache Spark pools, serverless SQL pool, and dedicated SQL pool.

## Benefits

- The shared metadata model supports the modern data warehouse pattern.
- The Spark created databases and all their tables become visible in any of the Azure Synapse workspace Spark pool instances and can be used from any of the Spark jobs provided necessary permissions are provided.
- Databases are created automatically in the serverless SQL pool metadata.
- The external and managed tables created by Spark job are made accessible as external tables in the serverless SQL pool metadata in the dbo schema of the corresponding database.
- Spark created databases and their Parquet-backed tables will be mapped into the SQL pools for which metadata synchronization enabled.



# Azure Synapse Analytics

## Linked Services



# Azure Machine Learning

## Overview

Native integration with Azure Machine Learning offers training model in code-first and code-free format. It enables ease of access to models from Machine Learning model registry for deployment in Synapse dedicated SQL pools and launch predictions to enrich the data.

## Benefits

Train models in code free manner with AutoML capabilities from Synapse.

Register models in Machine Learning model registry.

Deploy ONNX models in dedicated SQL pool to enrich data.

No data movement to get predictions.

Enables in-engine scoring the data and operationalize predictions.

The screenshot shows the Azure Machine Learning studio interface. On the left, there's a sidebar with 'default (Spark)' selected, showing tables like 'anomaly\_detector\_testing', 'fabrikam\_text\_analytics', 'nyc\_taxi', 'retail\_training\_data', and 'surfacesales\_featureddf'. A context menu is open over 'Machine Learning', with 'Enrich with existing model' and 'Enrich with new model' options highlighted. The main area has a modal dialog titled 'Enrich with new model' with 'retail\_training\_data' selected. Below it, a 'Choose a model type' section lists three options: 'Classification' (with a bar chart icon), 'Regression' (with a stack of bars icon), and 'Time series forecasting' (with a circular icon). Each option includes a brief description and an example. At the bottom of the dialog are 'Continue', 'Back', and 'Cancel' buttons.

default (Spark)

- Tables
  - anomaly\_detector\_testing
  - fabrikam\_text\_analytics
  - nyc\_taxi
  - retail\_training\_data
  - surfacesales\_featureddf

New SQL script >

New notebook >

Machine Learning > Enrich with existing model

Refresh Enrich with new model

Enrich with new model

retail\_training\_data

Choose a model type

Select the machine learning model type for the experiment based on the question you are trying to answer. Once you have selected the model type, you will be prompted with a few settings before the experiment run is created. [Learn more](#)

**Classification**  
Determine the likelihood of a specific outcome being achieved (binary classification) or identify the category an attribute belongs to (multiclass classification).  
Example: Predict if a customer will renew or cancel their subscription.

**Regression**  
Estimate a numeric value based on input variables.  
Example: Predict housing prices based on house size.

**Time series forecasting**  
Estimate values and trends based on historical data.  
Example: Predict stock market trends over the next year.

Continue Back Cancel



# Azure Machine Learning (Continued)

Deploy ONNX models in dedicated SQL pool to enrich data and operationalize with stored procedure

1

Predict\_Pool (SQL)  
Tables  
dbo.AMLModels  
dbo.RetailData  
dbo.retaildemo  
dbo.zretail\_scoring\_data  
dbo.zretail\_scoring\_data\_...  
External tables  
External resources  
Views  
Machine Learning > Enrich with existing model  
New SQL script  
New notebook  
New data flow  
New integration dataset  
Refresh

2

Enrich with existing model  
dbo.RetailData  
Select the model you want to use to enrich the selected dataset. [Learn more](#)  
Azure Machine Learning  
Azure Machine Learning workspace \* amlwsdemos  
Name Versi... Created Created By Fra...  
wsazuresynapseanalytics-retail\_training\_... 1 03:10:36 11/2... Nellie Gus... Cust...  
wsazuresynapseanalytics-retailsales-202... 1 11:10:24 11/1... Charles Fe... Cust...  
wsazuresynapseanalytics-retailsales-202... 1 10:58:55 11/1... Charles Fe... Cust...  
wsazuresynapseanalytics-retailsales-202... 1 10:24:16 11/1... Charles Fe... Cust...  
wsazuresynapseanalytics-retailsales-202... 1 10:21:17 11/1... Charles Fe... Cust...  
nyctaxi\_fareamount\_predict 1 02:34:37 11/1... Priyanka L... Cust...  
nyc\_taxi\_amount\_predict 6 02:33:48 11/1... Priyanka L... Cust...  
Continue Cancel

3

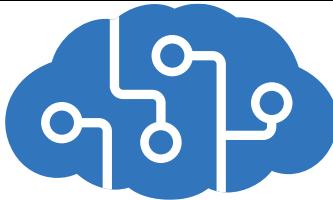
Enrich with existing model  
dbo.RetailData  
price → price real +   
advertising → advertising real +   
storeId → storeId real +   
weekStarting → weekStarting real +   
wholeSaleCost → wholeSaleCost real +   
basePrice → basePrice real +   
income → income real +   
minoritiesRatio → minoritiesRatio real +   
largeHH → largeHH real +   
highIncome150Ratio → highIncome150Ra... real +   
more1FullTimeEmplo → more1FullTimeEm... real +   
ratioAge60 → ratioAge60 real +   
salesNearestWarehou → salesNearestWare... real +   
salesNearest5StoresR → salesNearest5Stor... real +   
collegeRatio → collegeRatio real +   
avgDistanceNearest5 → avgDistanceNeare... real +   
distanceNearestWare → distanceNearestW... real +   
Output mapping \*  
Model Output Output Type  
quantity real +   
Continue Back Cancel

4

```

CREATE PROCEDURE dbo.test1
AS
BEGIN
SELECT *
FROM PREDICT (MODEL = (SELECT [model] FROM dbo.AMLModels WHERE [ID] = 'synaps...
    DATA = [dbo].[RetailData]) WITH ([quantity] [real])
END
GO
EXEC dbo.test1

```



# Azure Cognitive Services

## Overview

Azure Cognitive Services provides machine learning capabilities to solve general problems such as analyzing text for emotional sentiment or analyzing images to recognize objects or faces.

It enables adding cognitive features to analytics applications without having artificial intelligence (AI) or data science skills.

## Benefits:

Add Azure Cognitive Services as a Linked service in Azure Synapse and start enriching data in Azure Synapse with capabilities like anomaly detection or sentiment analysis.

No data movement to get predictions.

1

2

Enrich with existing model

fabrikam\_text\_analytics

This experience allows you to enrich the selected dataset with pre-trained [Azure Cognitive Services](#) models. [Learn more](#)

Azure Cognitive Services

---

Name

Anomaly Detector ⓘ

Text Analytics - Sentiment Analysis ⓘ

# Azure Cognitive Services (continued)

## Provide authentication details

## Enrich with existing model

fabrikam\_text\_analytics

3

Specify the Cognitive services account you want to connect to and configure which Azure key vault linked service to use for accessing secrets for authentication. [Learn more](#)

Azure subscription ⓘ

Cognitive Services account \* ⓘ

Azure Key Vault linked service \* ⓘ

Secret name \* ⓘ

## Configure sentiment analysis

**Enrich with existing model**

fabrikam\_text\_analytics

**Text Analytics - Sentiment Analysis**

Sentiment analysis evaluates the sentiment (positive/negative/neutral) of a text and also returns the probability (score) of the sentiment. [Learn more](#)

**Language \* ⓘ**  
English

**Text column \* ⓘ**  
comment (string)

**Open notebook** **Back** **Cancel**

# Open Notebook and run

The screenshot shows a Jupyter Notebook interface with the following details:

- Header:** Notebook 3, Run all, Undo, Publish, Attach to analyticspool, Language PySpark (Python), NextGen Notebooks (Preview).
- Code Block (Cell 1):** Contains Python code for sentiment analysis using PySpark. It loads data from a database, defines a pipeline for sentiment extraction, and displays the results.
- Output:** Command executed in 4mins 457ms by negatu on 11-17-2020 11:46:00.751 -0800
- Status Bar:** Job execution succeeded, Spark 2 executors 16 cores, View in monitoring, Open Spark UI.
- Data View:** A table showing the results of the sentiment analysis. The columns are comment, sentiment, statistics, documentScores, sentences, warnings, error-message, and error. The first few rows are:

comment	sentiment	statistics	documentScores	sentences	warnings	error-message	error
I ordered three w...	mixed					▶ "[{"sentiment": "neu", ""]"]	
Can't believe you...	negative					▶ "[{"sentiment": "neg", ""]"]	
Wow! I had no ...	mixed					▶ "[{"sentiment": "pos", ""]"]	
Jake, I've had sim...	mixed					▶ "[{"sentiment": "neu", ""]"]	
That hand model...	neutral					▶ "[{"sentiment": "neu", ""]"]	
I put my weddin...	negative					▶ "[{"sentiment": "neg", ""]"]	

# Azure Data Explorer

## Overview

Azure Data Explorer is a fast, fully managed data analytics service for real-time analysis on large volumes of Telemetry, Logs, Time Series data streaming from applications, websites, IoT devices, and more

The Azure Data Explore data connector for Azure Synapse Workspace is an extension of the Azure Data Explorer Spark connector that is now natively integrated into Azure Synapse Apache Spark pools

## Benefits

Use the continuous export feature to export data from Azure Data Explorer to Data Lake Store linked to Azure Synapse Workspace for archiving data in the data lake

Use the new ADX connector in Azure Synapse to query the cached and indexed data in Azure Data Explorer using Synapse Apache Spark

The screenshot shows the Azure Synapse Studio interface. At the top, there's a navigation bar with 'Synapse live', 'Validate all', and 'Publish all' buttons. Below it is a 'Data' section with tabs for 'Workspace' and 'Linked'. Under 'Linked', there's a list of resources: 'Azure Blob Storage' (3), 'Azure Cosmos DB' (1), 'Azure Data Explorer' (2), 'Azure Data Lake Storage Gen2' (1), and 'Integration datasets' (24). A context menu is open over the 'ML\_Models' item, with options: 'Read DataFrame from table', 'Write DataFrame to table', and 'Write streaming DataFrame to table'. A red box highlights these options. An arrow points down from this menu to a Python notebook titled 'Notebook 3' in the bottom half of the screen. The notebook contains the following code:

```

1 %%pyspark
2
3 # Read data from Azure Data Explorer table(s)
4 # Full Sample Code available at: https://github.com/Azure/azure-kusto-spark/blob/master/samples/src/main/python/SynapseSample.py
5
6 kustoDf = spark.read \
7   .format("com.microsoft.kusto.spark.synapse.datasource") \
8   .option("spark.synapse.linkedService", "AzureDataExplorerDemo") \
9   .option("kustoDatabase", "Occupancy") \
10  .option("kustoQuery", "Thermostats | take 10") \
11  .load()
12
13 display(kustoDf)
14

```

# Azure Key Vault



## Overview

Azure Key Vault integration offers increased security and control over keys and password.

It enables to securely store and tightly control access to tokens, passwords, secrets.

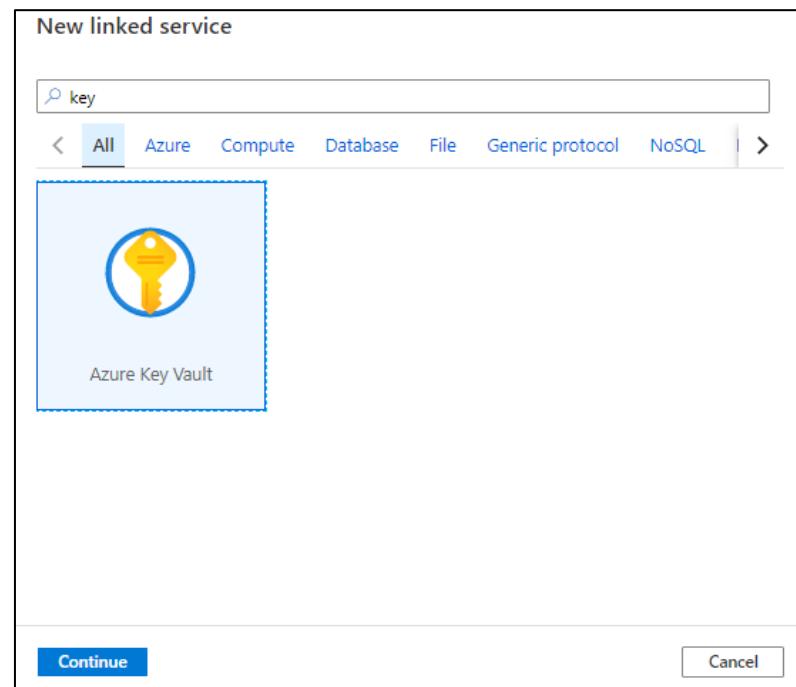
New linked service

key

All Azure Compute Database File Generic protocol NoSQL

Azure Key Vault

Continue Cancel



Linked services

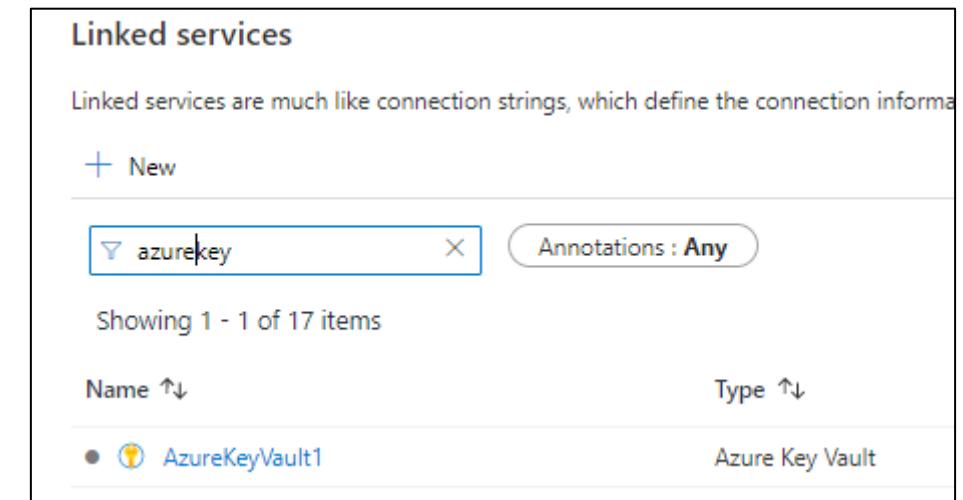
Linked services are much like connection strings, which define the connection information required by your data source.

+ New

azurekey Annotations : Any

Showing 1 - 1 of 17 items

Name ↑	Type ↑
● AzureKeyVault1	Azure Key Vault



# Azure Monitor



## Overview

Leverage Azure Monitor to obtain metrics, alerts, and logs.

Azure Synapse Analytics can write diagnostic logs in Azure Monitor.

Workspace level metrics, dedicated SQL pool level metrics, Apache Spark pool metrics are supported.

With Azure Monitor diagnostic settings, you can route diagnostic logs for analysis to multiple different targets.

With Azure Monitor diagnostic settings, you can route diagnostic logs for analysis to multiple different targets

- Storage Account
- Log Analytics workspace
- Event Hub

Home > wsazuresynapseanalytics > Diagnostic settings >

**Diagnostic setting**

Save Discard Delete Provide feedback

A diagnostic setting specifies a list of categories of platform logs and/or metrics that you want to collect from a subscription, and one or more destinations that you would stream them to. Normal usage charges for the destination will occur. [Learn more about the different log categories and contents of those logs](#)

Diagnostic setting name	TestAdministrativeDiagnosticLogs
Category details	<p><b>log</b></p> <ul style="list-style-type: none"> <li><input checked="" type="checkbox"/> Administrative</li> <li><input checked="" type="checkbox"/> Security</li> <li><input type="checkbox"/> ServiceHealth</li> <li><input checked="" type="checkbox"/> Alert</li> <li><input type="checkbox"/> Recommendation</li> <li><input type="checkbox"/> Policy</li> <li><input checked="" type="checkbox"/> Autoscale</li> <li><input type="checkbox"/> ResourceHealth</li> </ul>
Destination details	<p><input type="checkbox"/> Send to Log Analytics workspace</p> <p><input checked="" type="checkbox"/> Archive to a storage account</p> <p><b>Existing diagnostics will not appear in the portal if you change the storage account.</b></p> <p><b>Showing all storage accounts including classic storage accounts</b></p> <p>Location: All</p> <p>Subscription: <input type="text"/></p> <p>Storage account *: <input type="text"/></p> <p><input type="checkbox"/> Stream to an event hub</p>



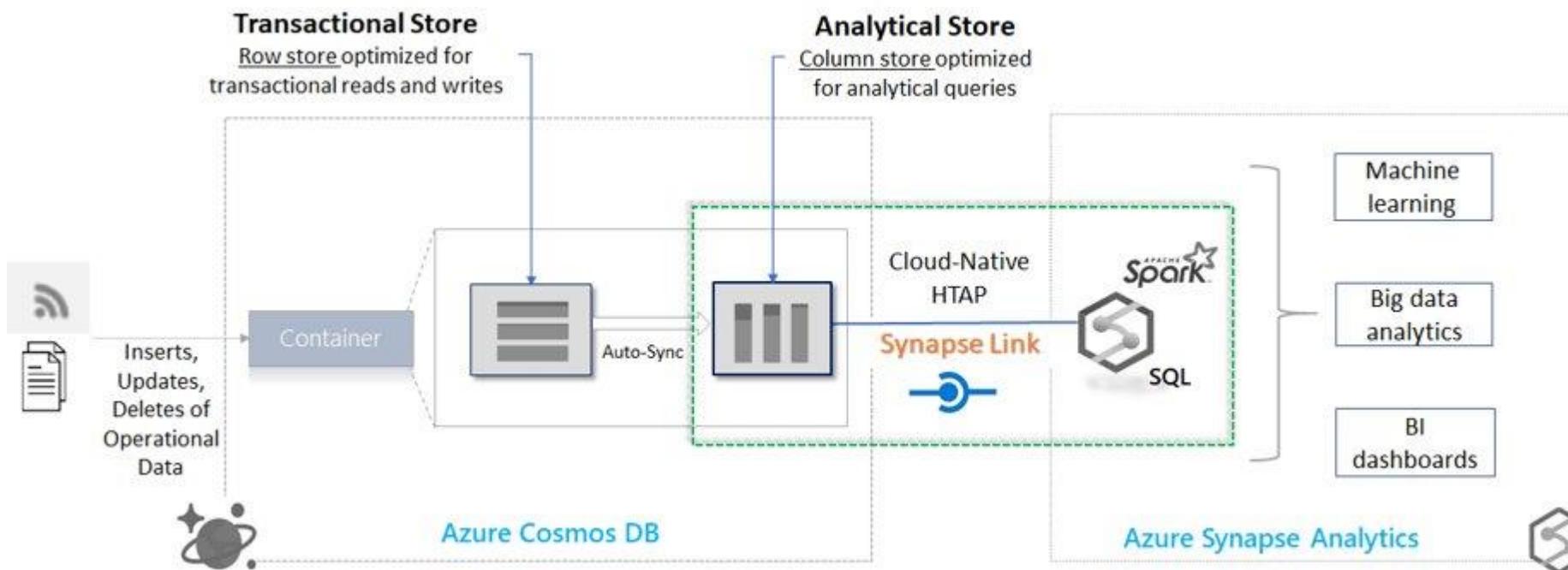
# Azure Synapse Analytics

## Synapse Link for Cosmos DB

# Synapse Link for Cosmos DB integration

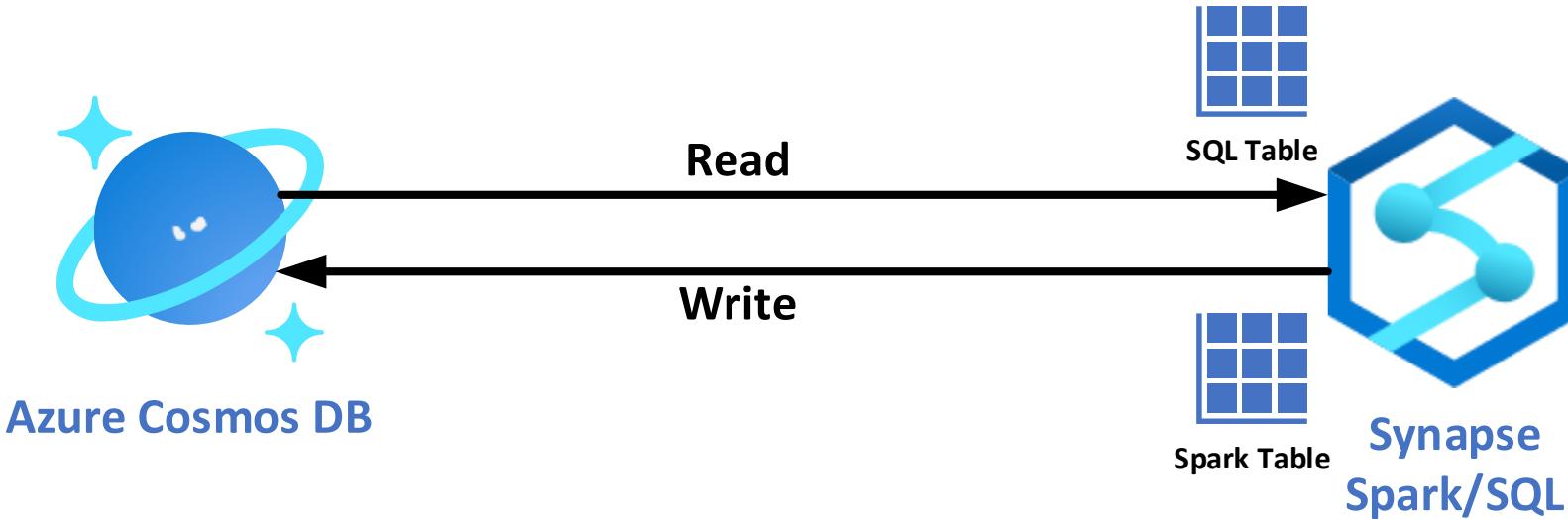
Cloud native hybrid transactional and analytical processing (HTAP) capability

In-place analytics over operational data in Azure Cosmos DB



# Modern Data Warehouse Architecture

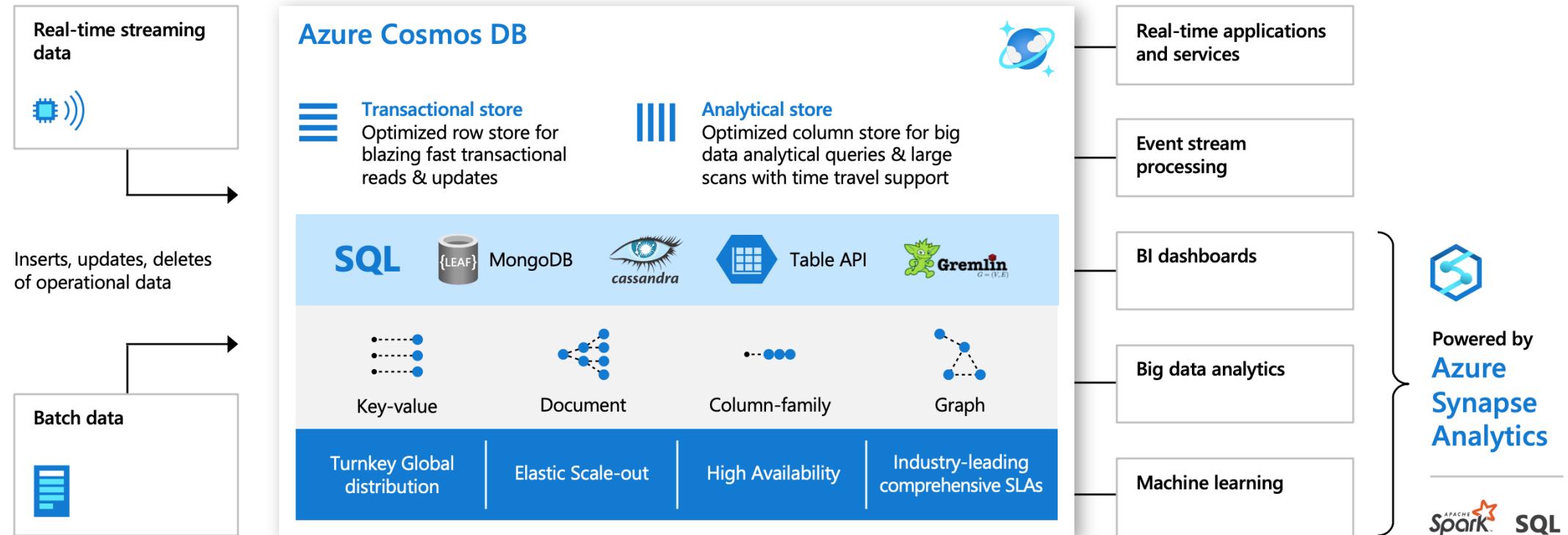
## *With Synapse Link for Cosmos DB*



HTAP reduces architectural complexity by collapsing the stack to:

- Accelerate new insight to perform near-real time analytics at the source with no impact on the transactional workload performance
- Deliver analytics and operational insights through Spark or SQL

# (Cosmos DB + Synapse) ❤ Analytics



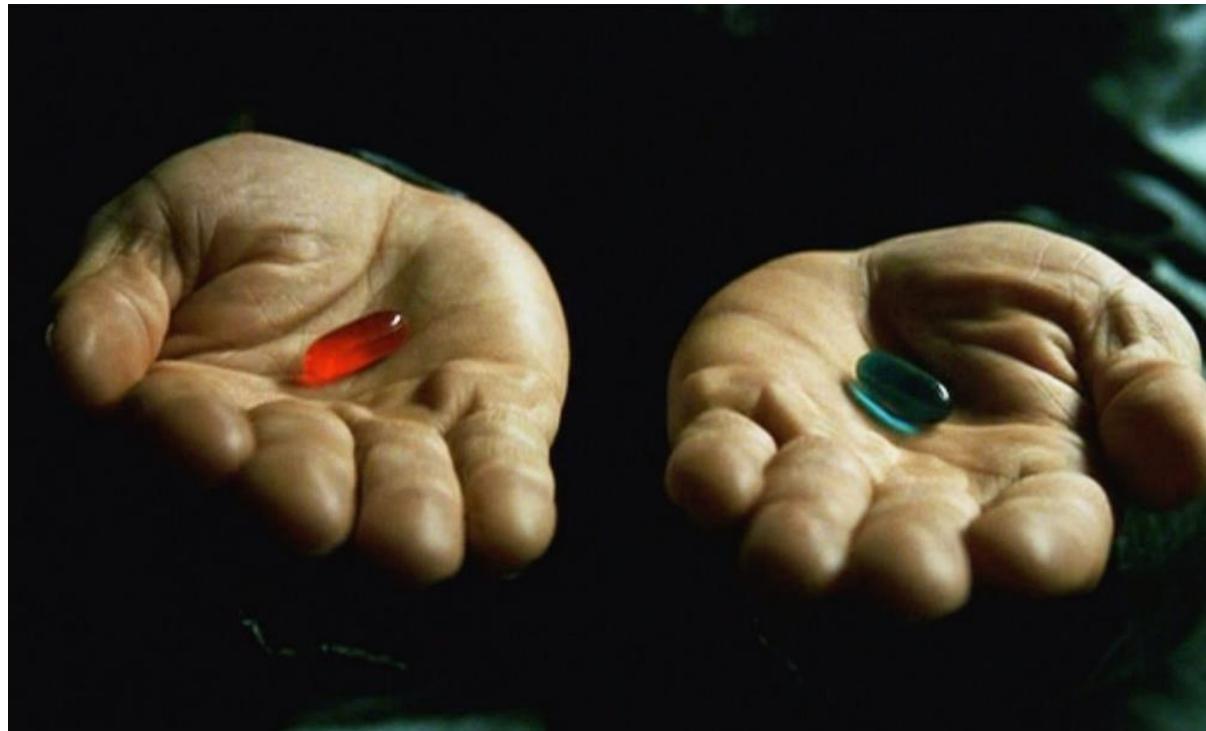
# Benefits of running HTAP in Cosmos DB and Synapse

## Operational Needs

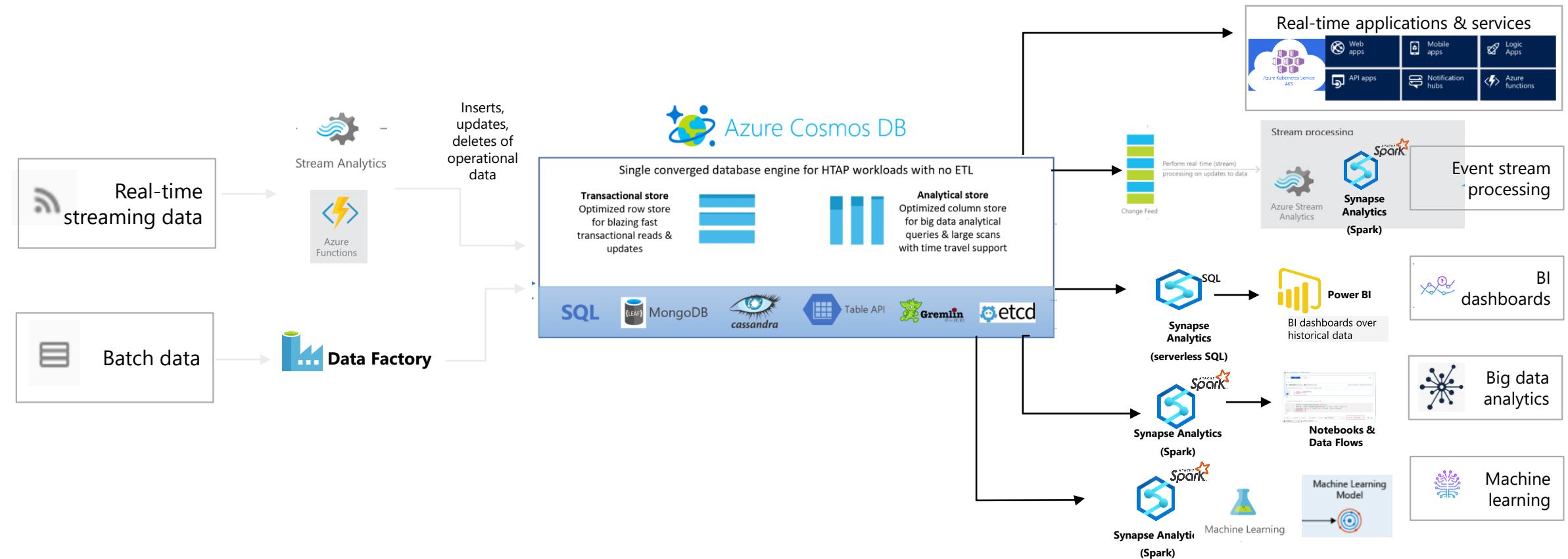
- High throughput ingestion of both batch & streaming feeds with real-time indexing
- Global distribution with active-active setup
- Apps to perform real-time CRUD & queries using developer friendly NoSQL APIs
- No downtime elastic scaling of database
- SLAs for latency/availability/consistency/throughput

## Analytics Needs

- Run analytics without impacting operational performance
- Low cost on storage and compute to run near-real time analytics workloads
- Analyze in-place data, eliminating ETL complexity, with Synapse Spark and SQL
- Rich BI ecosystem of SQL available for cheaper & faster queries over analytical storage
- Native integration of Azure Enterprise AI ecosystem for training and scoring ML models over analytical storage



# High level reference architecture for HTAP workloads





End