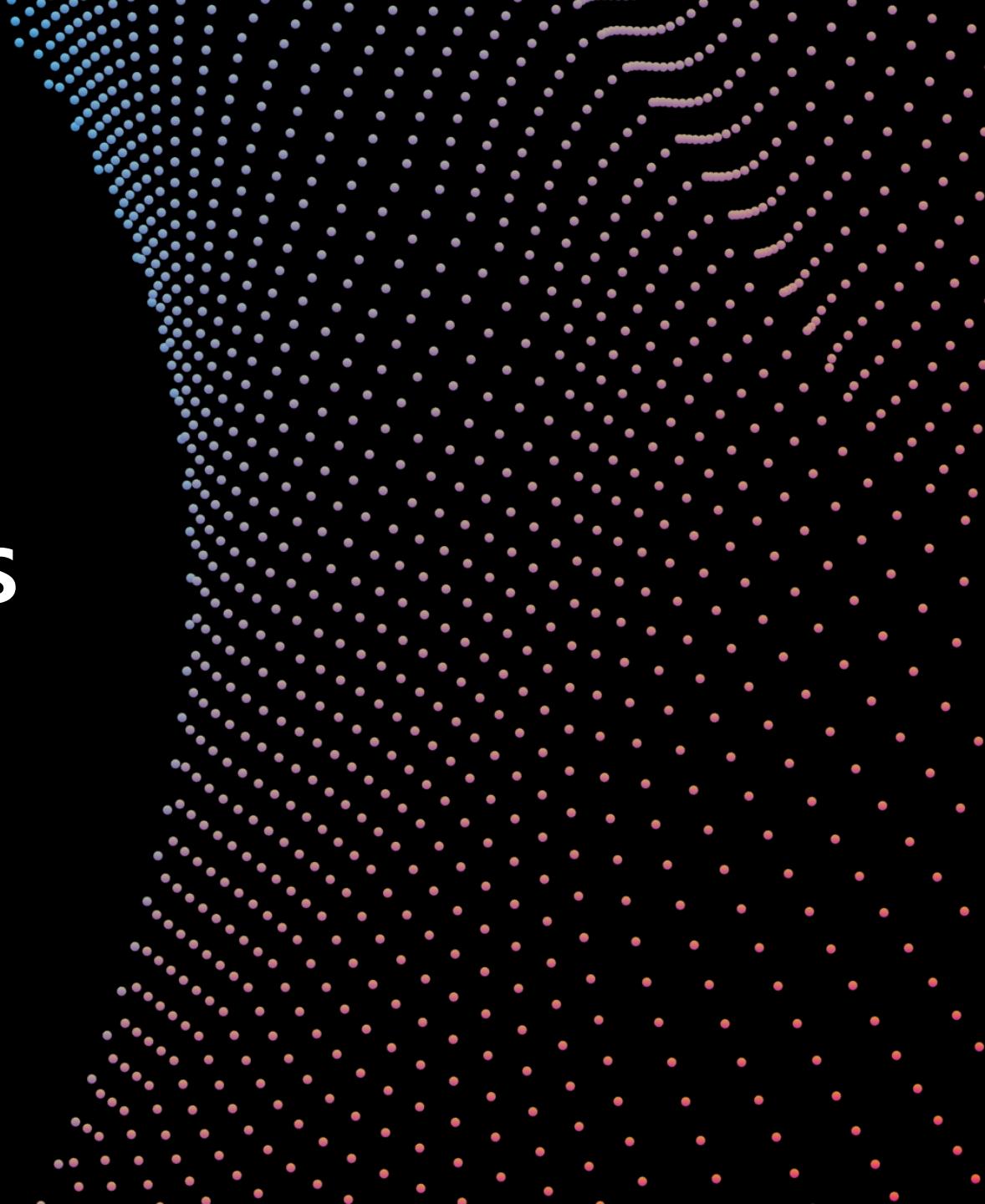


# Azure Synapse Analytics

## Technical Boot Camp

Day 1



# A look into Day 1

Kick-Off	6:30-6:35	Welcome	Main Call
	6:35-7:00	Azure Synapse Analytics 101	
	7:00-7:15	Demo Walkthrough	
Ingest	7:15-7:30	Break	Table Group Call
	7:30-8:30	Data Loading & Data Lake Organization	
	8:30-9:30	Activity: Data Lake Design & Security Considerations	
	9:30-9:45	Break	
	9:45-10:30	Build Hands-on: Data Integration Part 1	
Transform	10:30-11:30	Break	Main Call
	11:30-12:00	Data Transformations	
	12:00-12:30	Activity: Data Engineering Discussion	
	12:30-1:30	Build Hands-on: Data Integration Part 2	Table Group Call
		End of Day 1	

**Today we will be learning and collaborating across three spaces:**

- Main call (this meeting)
- Table Group channel within the event Team
- CloudLabs Learner Portal & Synapse environment

■ Presentation/  
Whole Group

■ Lab

■ Activity/ Discussion/  
Group Work

■ Announcements

# Azure Synapse Analytics 101

**Limitless analytics service with  
unmatched time to insight**



# Azure Synapse Analytics

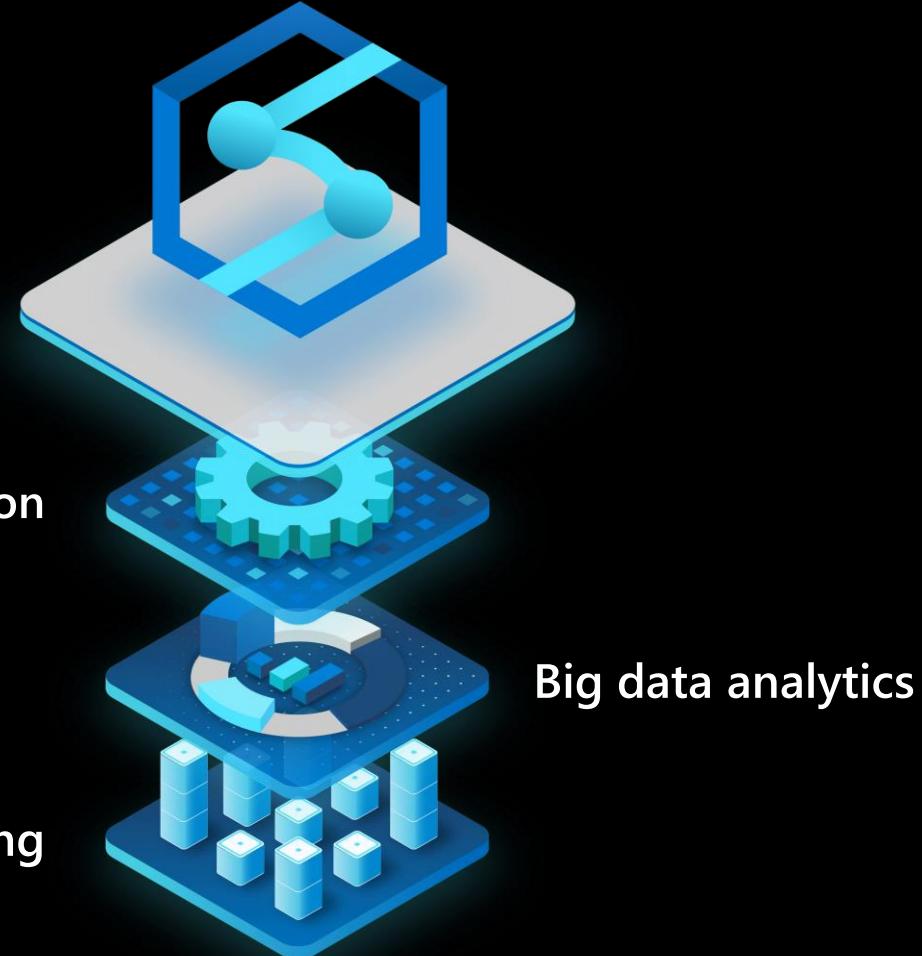
**The first unified, cloud native platform for converged analytics**

Azure Synapse is the only unified platform for analytics, blending big data, data warehousing, and data integration into a **single cloud native service** for end-to-end analytics at cloud scale.

Data integration

Data warehousing

Big data analytics



# Azure Synapse Analytics

---

Powered by a new cloud native  
distributed SQL engine



# Synapse Studio

## Unified workspace

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', 'Synapse Analytics', 'wsazuresynapseanalytics', a search bar, and notification icons. The left sidebar has a 'Data' section with 'Workspace' selected, showing a list of datasets like 'wwi.DimStockItem', 'wwi.DimSupplier', etc., and a 'Columns' section listing various columns. The main area has a 'SQL script 5' tab open with the following SQL query:

```
1 SELECT TOP 10
2     City,
3         SUM(Quantity) AS Quantity
4 FROM
5     wwi.FactOrder f
6 INNER JOIN wwi.DimCity d ON d.CityKey = f.CityKey
7 WHERE StateProvince = 'Washington'
8 GROUP BY
9     City
10 ORDER BY
11     Quantity DESC
12
13
```

Below the query, there are tabs for 'Results' and 'Messages', with 'Results' selected. The results are displayed as a bar chart titled 'Quantity' versus 'City'. The chart shows the total quantity for each city, with Sekiu having the highest quantity (approximately 19k) and Trentwood having the lowest (approximately 13k). The cities listed on the x-axis are: Sekiu, Venerborg, Harbour Pointe, Lake Stevens, Koontzville, Malott, College Place, Upper Preston, Point Roberts, and Trentwood.

# Synapse Studio

Synapse Studio divided into **Activity hubs**.

These organize the tasks needed for building analytics solution.

The screenshot shows the Microsoft Synapse Studio interface. On the left, there's a sidebar with a red border around the first five items: Home, Data, Develop, Integrate, and Monitor. A red arrow points from the 'Integrate' item to the 'Integrate' hub on the main page. The main area is titled 'Synapse workspace' and shows six activity hubs: Home, Data, Develop, Integrate, Monitor, and Manage. Each hub has a description and a small icon. The 'Integrate' hub is currently selected.

**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

# Home hub

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

The screenshot shows the Microsoft Azure Synapse Analytics workspace interface. At the top, there's a navigation bar with icons for back, forward, and search. Below it, the workspace name 'asa-[REDACTED]-POC' is displayed. A 'New' button is located near the workspace name. On the left, a vertical sidebar contains icons for Home, Ingest, Explore and analyze, Visualize, and other workspace management options.

Red callout boxes highlight several key features:

- Updates**: Points to the 'Ingest' section, which includes a sub-section for 'Explore and analyze'.
- Switch Workspaces**: Points to the 'Switch Workspaces' icon in the top right corner.
- Language Settings**: Points to the 'Visualize' section, which includes a sub-section for 'Explore and analyze'.
- Help Knowledge Center**: Points to the 'Help Knowledge Center' icon in the top right corner.
- Feedback**: Points to the 'Feedback' icon in the top right corner.

Below the main content area, there are sections for 'Discover more' (Knowledge center, Browse partners) and 'Recent resources' (DW Load User Creation, DW Creation Script). The 'Recent resources' table includes columns for Name, Last opened by you, and a timestamp (6 days ago).

# Home hub

## Overview

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

**Recent resources** – Lists recently opened workspace artifacts for quick access

Discover more

 Knowledge center  Browse partners

Recent resources

Name	Last opened by you
Column Level Security	2 minutes ago
DW Load User Creation	6 days ago
DW Creation Script	8 days ago
Raw Ingestion and Refinement Pipeline	
Update Data Warehouse Dimensions	

Show more ▾

Synapse Analytics workspace  
asa-[REDACTED]-poc

New ▾

Synapse Analytics workspace  
asa-[REDACTED]-poc

New ▾

-  SQL script
-  KQL script
-  Notebook
-  Data flow
-  Apache Spark job definition
-  Power BI report
-  Pipeline
-  Import

# Knowledge center

Knowledge center offers industry-specific database templates, 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

**Gallery**

Database templates Datasets Notebooks SQL scripts Pipelines

Filter by keyword

Database templates

Use database templates to quickstart creation of lake databases. [Learn more](#)

 Agriculture For companies engaged in growing crops, raising livestock and dairy production.	 Banking For companies who are analyzing banking data.	 Consumer Goods For manufacturers or producers of goods bought and used by consumers.	 Energy & Commodity Trading For traders of energy, commodities, or carbon credits.	 Freight & Logistics For companies providing freight and logistics services.	 Fund Management For companies managing investment funds on behalf of investors.
 Life Insurance & Annuities For companies who provide life insurance, sell annuities, or both.	 Oil & Gas For companies involved in various phases of the Oil & Gas value chain.	 Property & Casualty Insurance For companies who provide insurance against risks to property and various forms of liability coverage.	 Retail For sellers of consumer goods or services to customers through multiple channels.	 Utilities For gas, electric and water utilities and power generators and water desalination.	

AI solutions

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

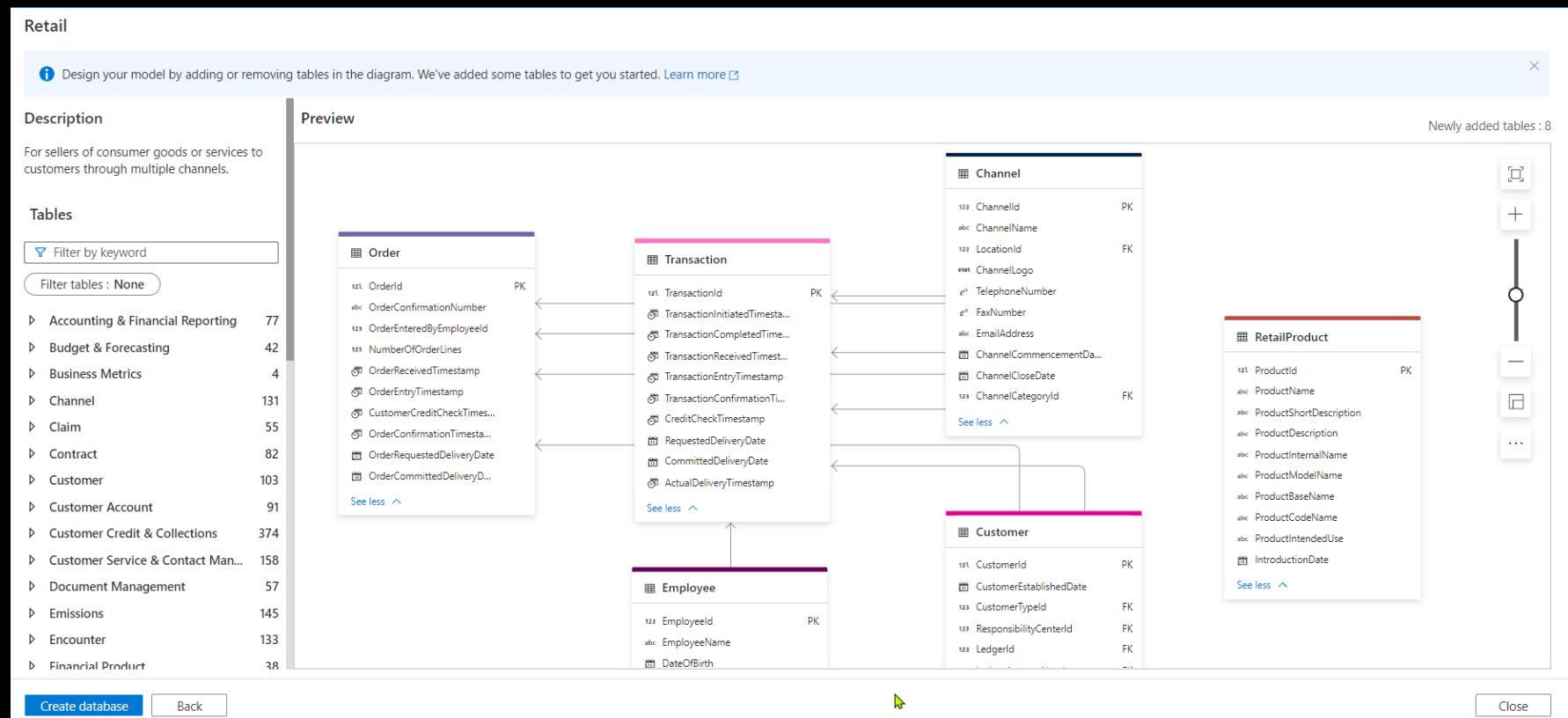
# Knowledge center – database templates

## Overview

Leverages decades of industry-specific experience to create an extensible schema to help you design your lake database. The database schemas can be expanded or derived from your own business terminology and taxonomies while Azure Synapse manages connections and storage of the schema.

## Templates available

- Retail
- Consumer Goods
- Banking
- Energy
- Freight & Logistics
- Agriculture
- Energy & Commodity Trading
- Freight & Logistics
- Oil & Gas
- Utilities



# Data hub

Explore data inside the workspace and in linked storage accounts

Microsoft Azure | Synapse Analytics > asa-[REDACTED]-poc

The screenshot shows the Microsoft Azure Synapse Analytics Data page. The left sidebar has icons for Home, Data, Develop, Integrate, Monitor, and Manage. The main area has a 'Data' tab with two sub-tabs: 'Workspace' (selected, highlighted with a red box) and 'Linked'. A search bar says 'Filter resources by name'. Below are sections for 'Lake database' (curated\_movies, default) and 'SQL database' (movies (SQL)).

Microsoft Azure | Synapse Analytics > wsazuresynapseanalytics

The screenshot shows the Microsoft Azure Synapse Analytics Data page. The left sidebar has icons for Home, Data, Develop, Integrate, Monitor, and Manage. The main area has a 'Data' tab with two sub-tabs: 'Workspace' and 'Linked' (selected, highlighted with a red box). A search bar says 'Filter resources by name'. Below are sections for 'Azure Blob Storage' (3), 'Azure Cosmos DB' (1), 'Azure Data Explorer' (2), 'Azure Data Lake Storage Gen2' (2), 'wsazuresynapseanalytics (Primary...)' (Attached Containers), and 'Integration datasets' (24).

# Data hub - Linked

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

The screenshot shows the Microsoft Azure Synapse Analytics Data hub interface. On the left, the 'Data' sidebar lists various linked resources under the 'Linked' tab:

- Linked Cosmos DB Analytical Store
- Linked Azure Data Explorer
- Linked ADLS Gen2 Account
- Container (filesystem)

Red arrows point from each of these labels to their corresponding entries in the sidebar. The 'Linked' tab is selected, showing a list of resources:

- Azure Blob Storage (3 items)
- Azure Cosmos DB (1 item)
- Azure Data Explorer (2 items)
- Azure Data Lake Storage Gen2 (1 item)
  - wsazuresynapseanalytics (Primary...)
    - default (Primary)
    - rawdata
    - staging
- Integration datasets (24 items)

The 'rawdata' folder under the ADLS Gen2 account is currently selected, as indicated by a red box around it. The breadcrumb navigation bar shows the path: rawdata > taxidata. The main pane displays the contents of the 'rawdata' folder:

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 of the main pane, it says 'Showing 1 to 6 of 6 cached items'.

# Data hub – Storage accounts

Preview a sample of your data

The screenshot illustrates the Azure Data Hub interface for previewing data from a storage account.

**Left Panel (Data):** Shows a list of linked workspaces. The "Linked" tab is selected, and the "Azure Blob Storage" workspace is currently active. A red arrow points to the "Preview" option in the context menu for the "wwi-factsale.csv" file.

**Middle Panel (wwi-02):** Displays a navigation tree under "wwi-02". The "sale-csv" folder is selected. A red arrow points to the "Preview" option in the context menu for the "wwi-factsale.csv" file.

**Right Panel (Products.csv):** Shows a preview of the "Products.csv" file. 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 table contains 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

**Buttons:** "OK" button at the bottom right of the preview modal.

# Data hub – storage accounts

See basic file properties

The screenshot illustrates the process of viewing basic file properties for a CSV file in Azure Data Studio.

**Left Panel (Data Explorer):** Shows the "Linked" workspace selected. The tree view includes:

- Azure Blob Storage (4 items)
- Azure Cosmos DB (2 items)
- Azure Data Explorer (1 item)
- Azure Data Lake Storage Gen2 (8 items):**
  - asaworkspace01 (Primary - asadata...)
  - asadatalake01 (asadatalake01)
    - dev
    - staging
    - tempdata
    - trigger
    - wwi
  - wwi-02
- asadatalake02 (asadatalake01)

**Middle Panel (File Explorer):** Displays the file structure under "wwi-02". The current path is "wwi-02 > sale-csv". A file named "wwi-factsale.csv" is selected. A context menu is open, with the "Properties..." option highlighted by a red box.

**Right Panel (Properties Dialog):** The "Properties" dialog is shown for the selected file. The "Name" field contains "sample csv files/Products.csv". The "URL" field shows the full Azure Data Lake Storage URL: <https://azuresynapsesa.dfs.core.windows.net/rawdata/sample csv files/Products.csv>. Other fields include:

- 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: (empty)
- Content Encoding: (empty)
- Content Language: (empty)
- User Properties: (empty)

At the bottom of the dialog are "Apply" and "Cancel" buttons.

# Data hub – storage accounts

## Manage Access - Configure standard POSIX ACLs on files and folders

The screenshot shows the Azure Data Explorer interface. On the left, the 'Data' sidebar is open with the 'Linked' tab selected. It lists various Azure services and their counts: Azure Blob Storage (4), Azure Cosmos DB (2), Azure Data Explorer (1), Azure Data Lake Storage Gen2 (8), and two specific data lake storage accounts: 'asadworkspace01' and 'asadatalake01'. The 'asadatalake01' account has five containers listed under it: dev, staging, tempdata, trigger, and wwi. The 'wwi' container is currently selected. On the right, the main workspace shows a dataset named 'wwi-02' with a path of 'wwi-02 > sale-csv'. A context menu is open for the 'wwi-factsale.csv' file, with the 'Manage access...' option highlighted by a red box.

- Workspace
- Linked

Filter resources by name

- ▶ Azure Blob Storage 4
- ▶ Azure Cosmos DB 2
- ▶ Azure Data Explorer 1
- ◀ Azure Data Lake Storage Gen2 8
  - ▶ asadworkspace01 (Primary - asadata...)
  - ◀ asadatalake01 (asadatalake01)
    - dev
    - staging
    - tempdata
    - trigger
    - wwi

wwi-02

New SQL script New notebook New ...

← → ↘ ↑ wwi-02 > sale-csv

Name

wwi-factsale.csv

- Preview
- New SQL script >
- New notebook >
- New data flow
- New integration dataset
- Manage access...
- Rename...
- Download
- Delete
- Properties...

## Manage access

sale-csv/wwi-factsale.csv

Access control list (ACL) allows you to associate a security principal with an access level for files and directories. [Learn more](#)

[+ Add](#)

Showing 1 - 4 of 4 items

Name	Type
\$superuser (Owner) \$superuser	<a href="#">Edit</a> User
\$superuser (Owning Group) \$superuser	<a href="#">Edit</a> Group
Other	--
Mask	--

Permissions assigned to  
\$superuser

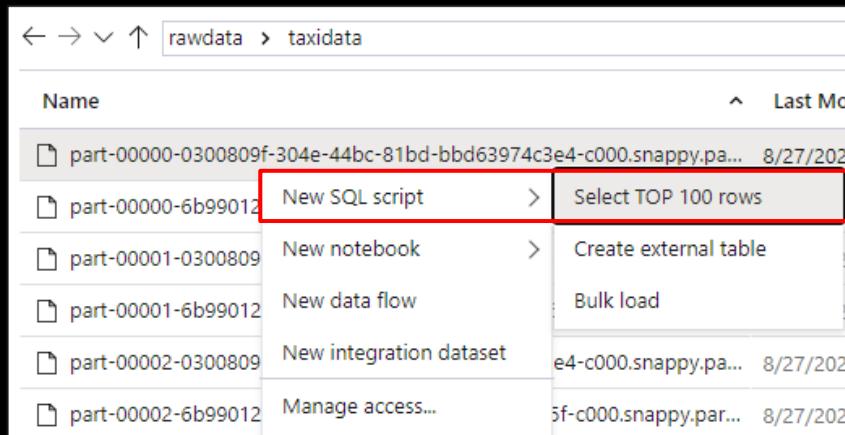
Type	Read	Write	Execute
Access	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>

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

# Data hub – storage accounts

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

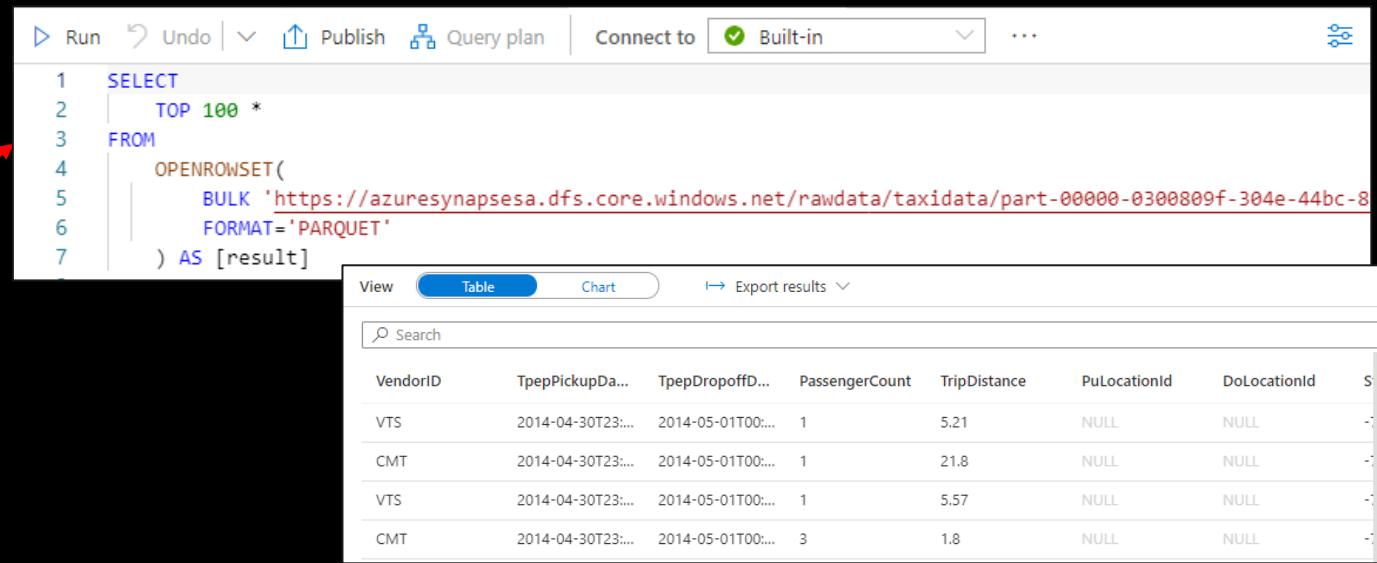
T-SQL or PySpark auto-generated.



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

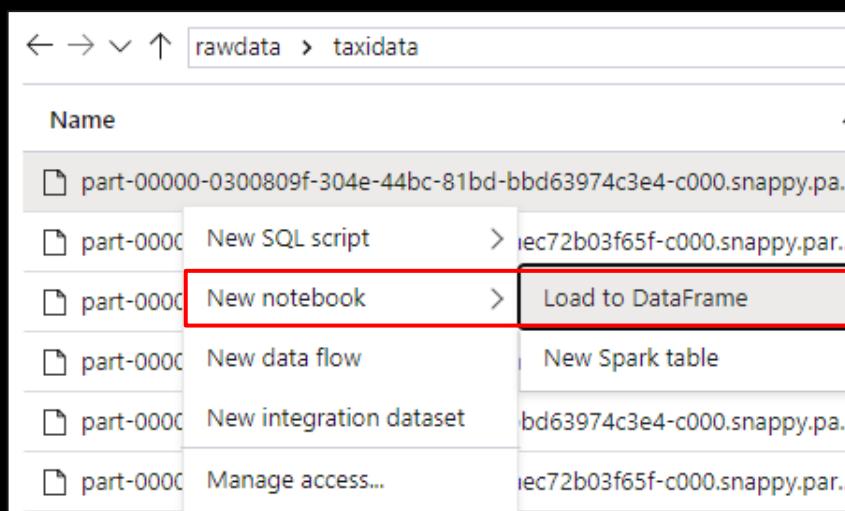


Run Undo Publish Query plan Connect to Built-in ...

```
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

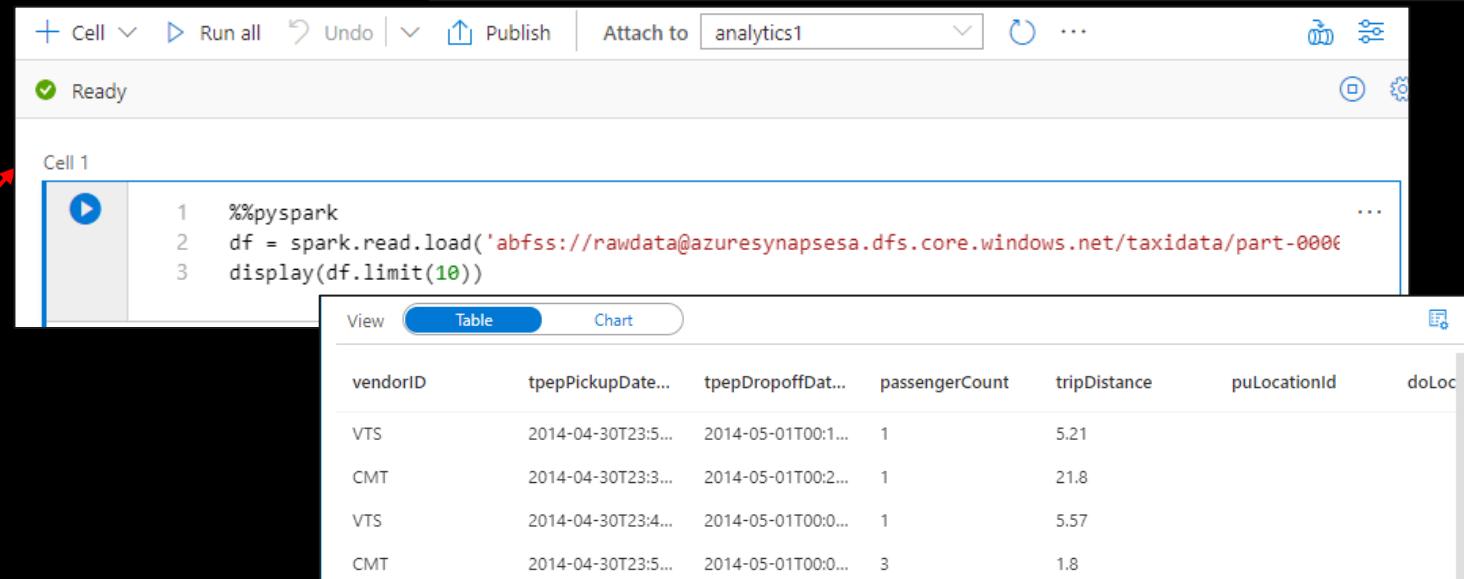
VendorID	TpepPickupDate	TpepDropoffDate	PassengerCount	TripDistance	PuLocationId	DoLocationId	...
VTS	2014-04-30T23:59:59.998Z	2014-05-01T00:00:00.000Z	1	5.21	NULL	NULL	-
CMT	2014-04-30T23:59:59.998Z	2014-05-01T00:00:00.000Z	1	21.8	NULL	NULL	-
VTS	2014-04-30T23:59:59.998Z	2014-05-01T00:00:00.000Z	1	5.57	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	2014-05-01T00:00:23.000Z	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 – databases

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

Lake database

Dedicated SQL pool

Serverless SQL pool

The screenshot shows the 'Data' hub interface with the 'Workspace' tab selected. It displays three main categories of databases:

- Lake database:** Contains 2 items: 'asa\_db01' and 'default'. A red arrow points from the 'asa\_db01' item to the corresponding item in the detailed view on the right.
- Dedicated SQL pool:** Contains 7 items: 'SQLPool01 (SQL)', 'SQLPool02 (SQL)', 'SQLPool03 (SQL)', 'SQLPool04 (SQL)', 'SQLPool05 (SQL)', 'SQLPool06 (SQL)', and 'SQLPool07 (SQL)'. A red arrow points from the 'SQLPool01 (SQL)' item to the corresponding item in the detailed view on the right.
- Serverless SQL pool:** Contains 1 item: 'SQLServerlessDB (SQL)'. A red arrow points from the 'SQLServerlessDB (SQL)' item to the corresponding item in the detailed view on the right.

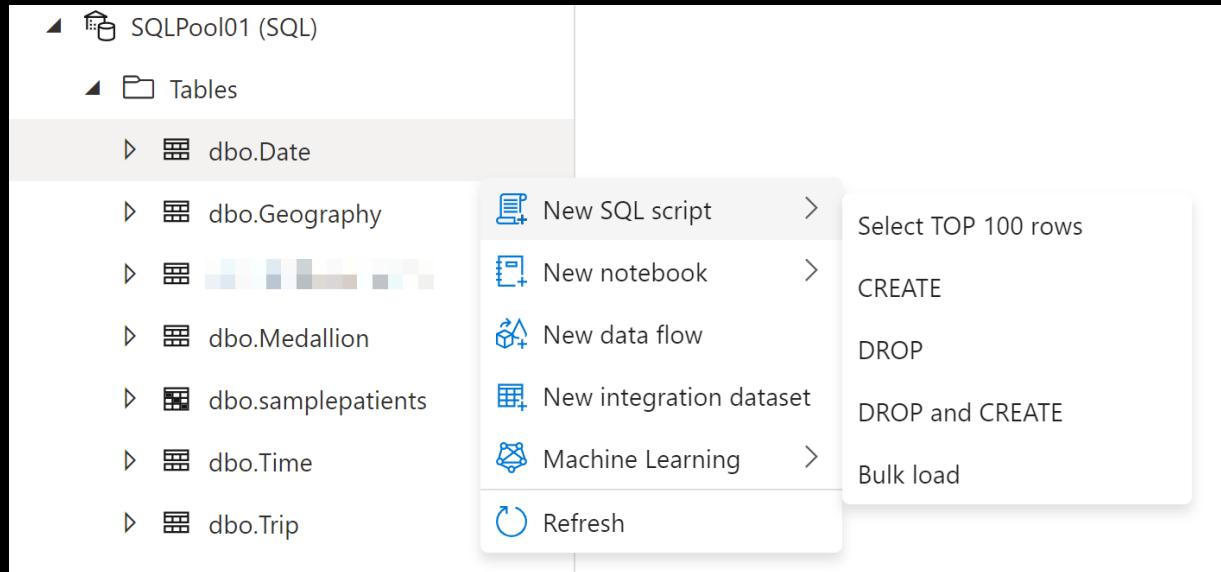
The screenshot shows the 'Data' hub interface with the 'Linked' tab selected. It displays three main categories of databases:

- Lake database:** Contains 2 items: 'asa\_db01' and 'default'. A red arrow points from the 'asa\_db01' item to the corresponding item in the workspace view on the left.
- SQL database:** Contains 7 items: 'SQLPool01 (SQL)', 'SQLPool02 (SQL)', 'SQLPool03 (SQL)', 'SQLPool04 (SQL)', 'SQLPool05 (SQL)', 'SQLPool06 (SQL)', and 'SQLPool07 (SQL)'. A red arrow points from the 'SQLPool01 (SQL)' item to the corresponding item in the workspace view on the left.
- Serverless SQL pool:** Contains 1 item: 'SQLServerlessDB (SQL)'. A red arrow points from the 'SQLServerlessDB (SQL)' item to the corresponding item in the workspace view on the left.

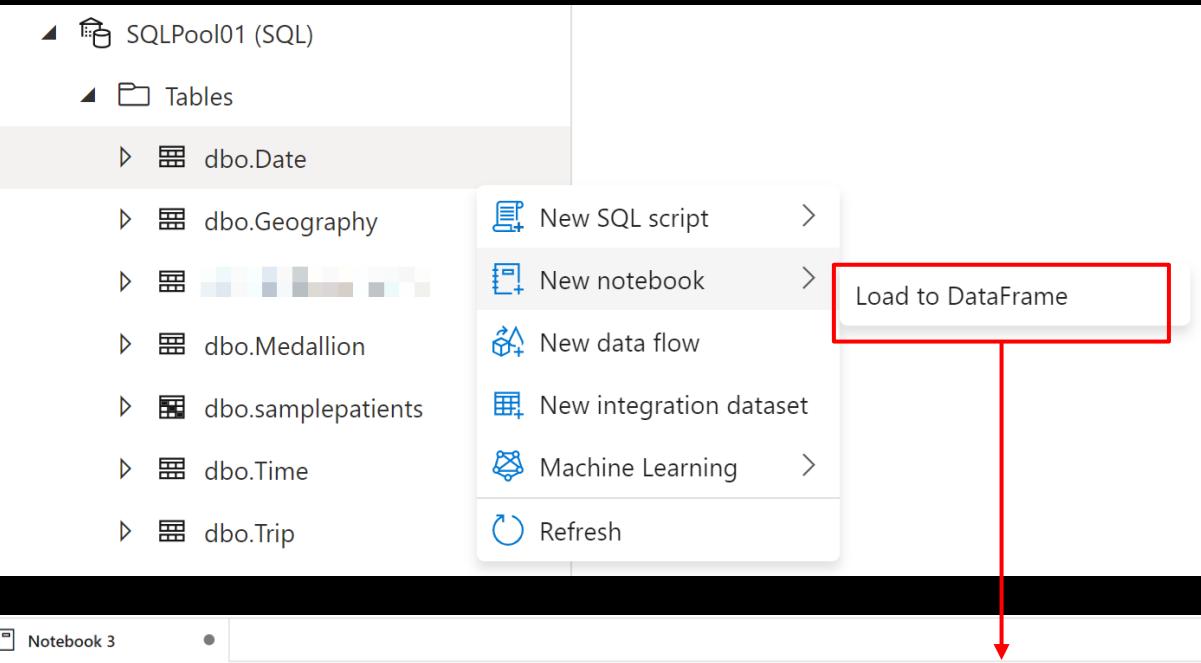
Each database item is expanded to show its sub-resources: Tables, Views, default, and Programmability, Schemas, Security, External tables, External resources, and Views.

# Data hub – databases

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



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



A screenshot of a Jupyter Notebook titled 'Notebook 3'. The toolbar at the top includes 'Run all', 'Undo', 'Publish', 'Outline', 'Attach to', 'Language', 'Spark (Scala)', and 'Variables'. The status bar shows 'Not started'. Below the toolbar, a code cell contains the following PySpark code:

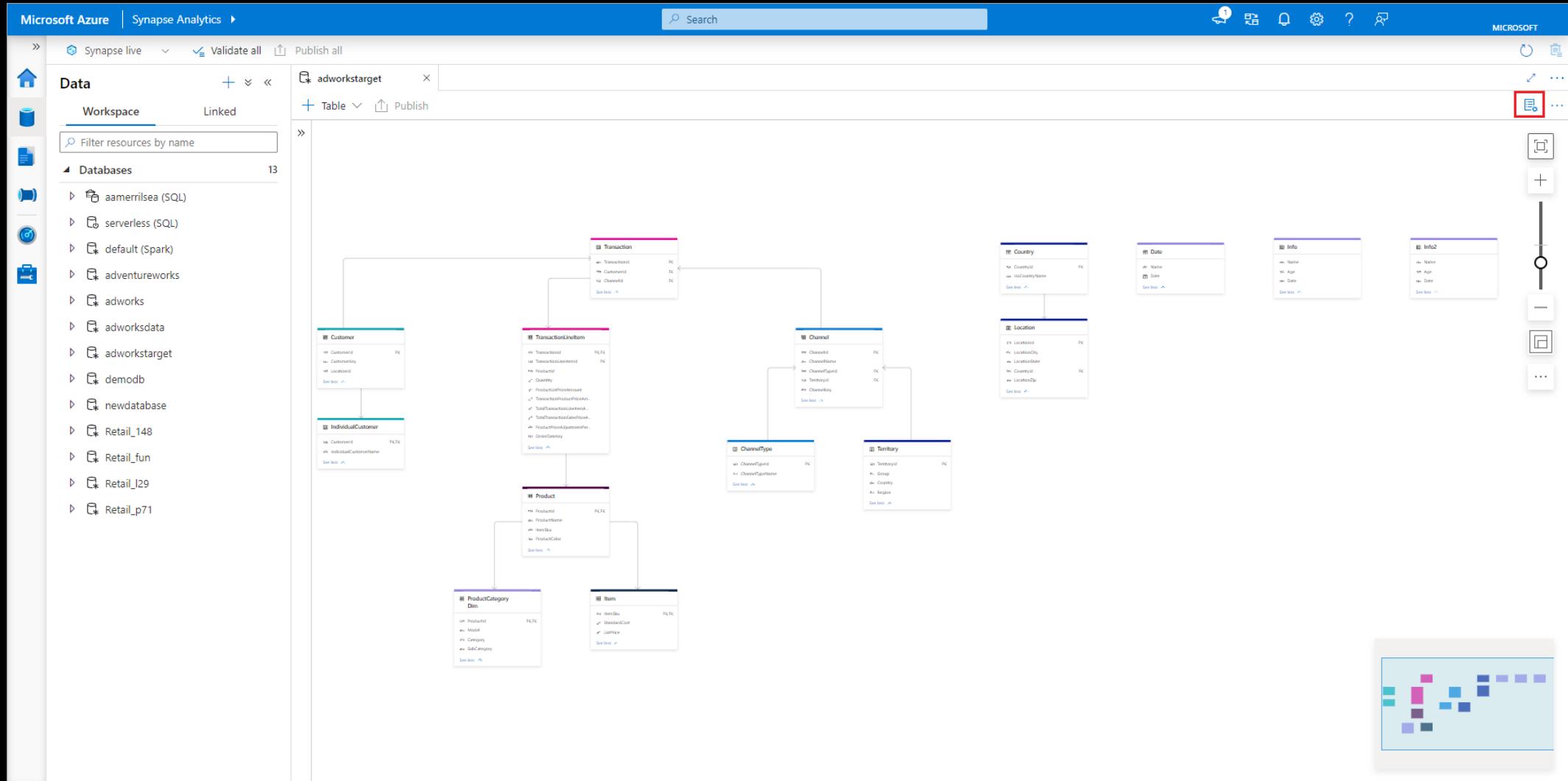
```
1 %%spark
2 val df = spark.read.sqlAnalytics("SQLPool01.dbo.Date")
3 df.write.mode("overwrite").saveAsTable("default.t1")
```

The code cell has a note at the bottom: 'Press shift + enter to execute cells'. At the bottom right of the notebook are buttons for '+ Code' and '+ Markdown'.

# Data hub – databases

Lake database designer (in preview)

Visually compose database tables, columns, and relationships



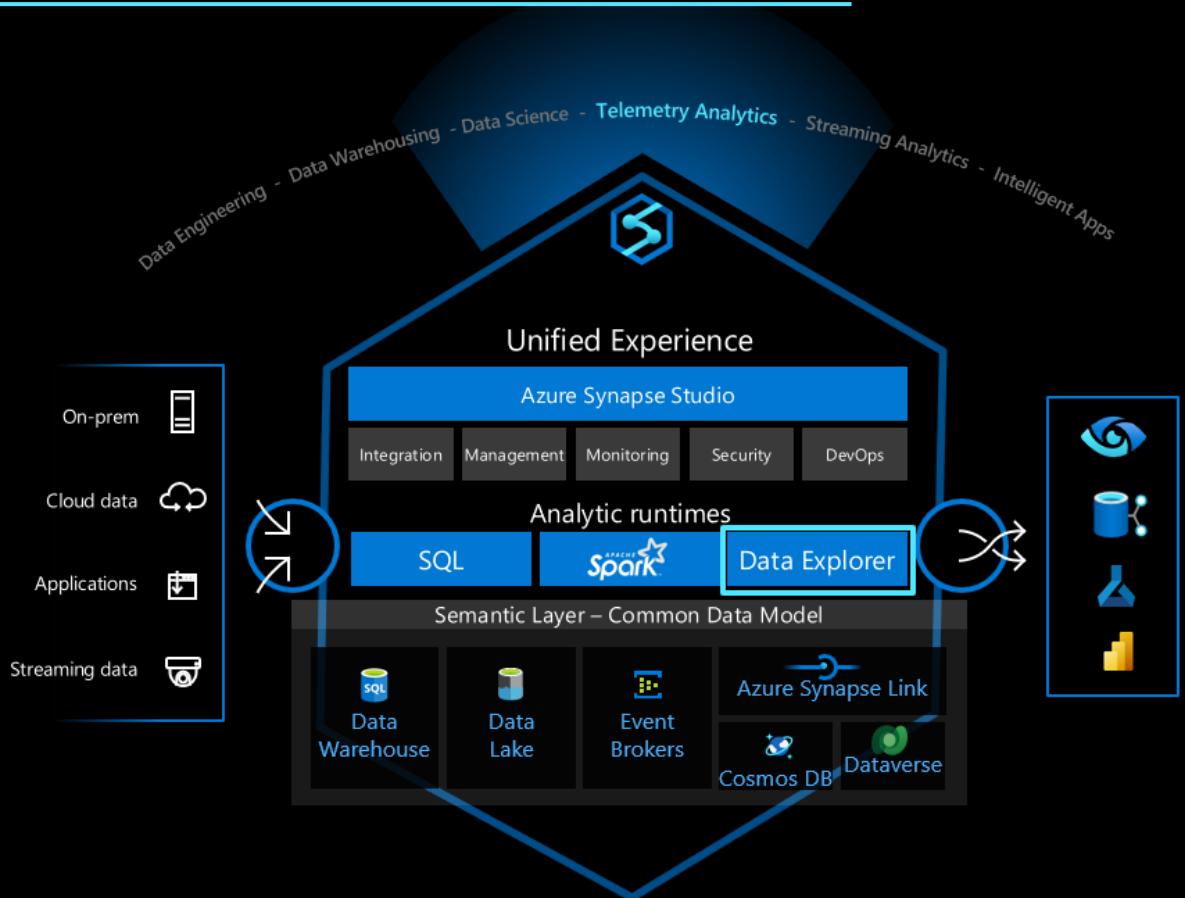
# Data Hub - Azure Synapse data explorer

Distributed storage & query engine  
for log & telemetry data

Optimized for telemetry analytics.

Interactively analyze massive amounts of text heavy log data in near real-time for scenarios such as IoT machine observability or web application telemetry.

Geospatial functionality in Azure Data Explorer provides options for rendering map data.



# Data hub – datasets

Integration 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 illustrates the process of creating an integration dataset in the Azure Data Hub. On the left, the 'Data' blade shows a list of resources under the 'Linked' tab, including 'Azure Blob Storage', 'Azure Cosmos DB', 'Azure Data Explorer', 'Azure Data Lake Storage Gen2', and 'Integration datasets'. A red box highlights the 'Integration datasets' item, and a red arrow points from this box to the 'asal400\_sales\_adlsgen2' dataset on the right. The right side shows the detailed configuration for the 'asal400\_sales\_adlsgen2' dataset, which is defined as a Parquet file type. The 'Connection' tab is selected, showing 'Linked service' set to 'asadatalake01' (highlighted by a blue box). The 'Integration runtime' is set to 'LargeAzureComputeOptimizedIntegr.v1'. The 'File path' is specified as 'wwi-02 / sale / File'. The 'Compression type' is set to 'snappy'.

# 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

The screenshot shows the Microsoft Azure Synapse Analytics Develop hub. The top navigation bar includes 'Microsoft Azure', 'Synapse Analytics', and a workspace name 'wsazures'. On the left is a vertical sidebar with icons for Home, Databases, Notebooks, Dataflows, and Power BI. The main area is titled 'Develop' and contains a search bar labeled 'Filter resources by name'. Below the search bar is a list of resource categories with counts: 'Power Queries (Preview)' (1), 'SQL scripts' (16), 'Demo scripts' (5 items: '001 SQL Pool Security RLS DDM ...', '005 Predict In-Engine Scoring', 'MV Query', 'Serverless Cost Control', 'TVF Demo'), 'MS Conf scripts' (1 item), 'Set up scripts' (1 item), 'Notebooks' (35), 'Data flows' (3), and 'Power BI' (1). Action buttons at the top right include 'Validate all', 'Publish all', and 'Discard all'.

# Develop hub – SQL scripts

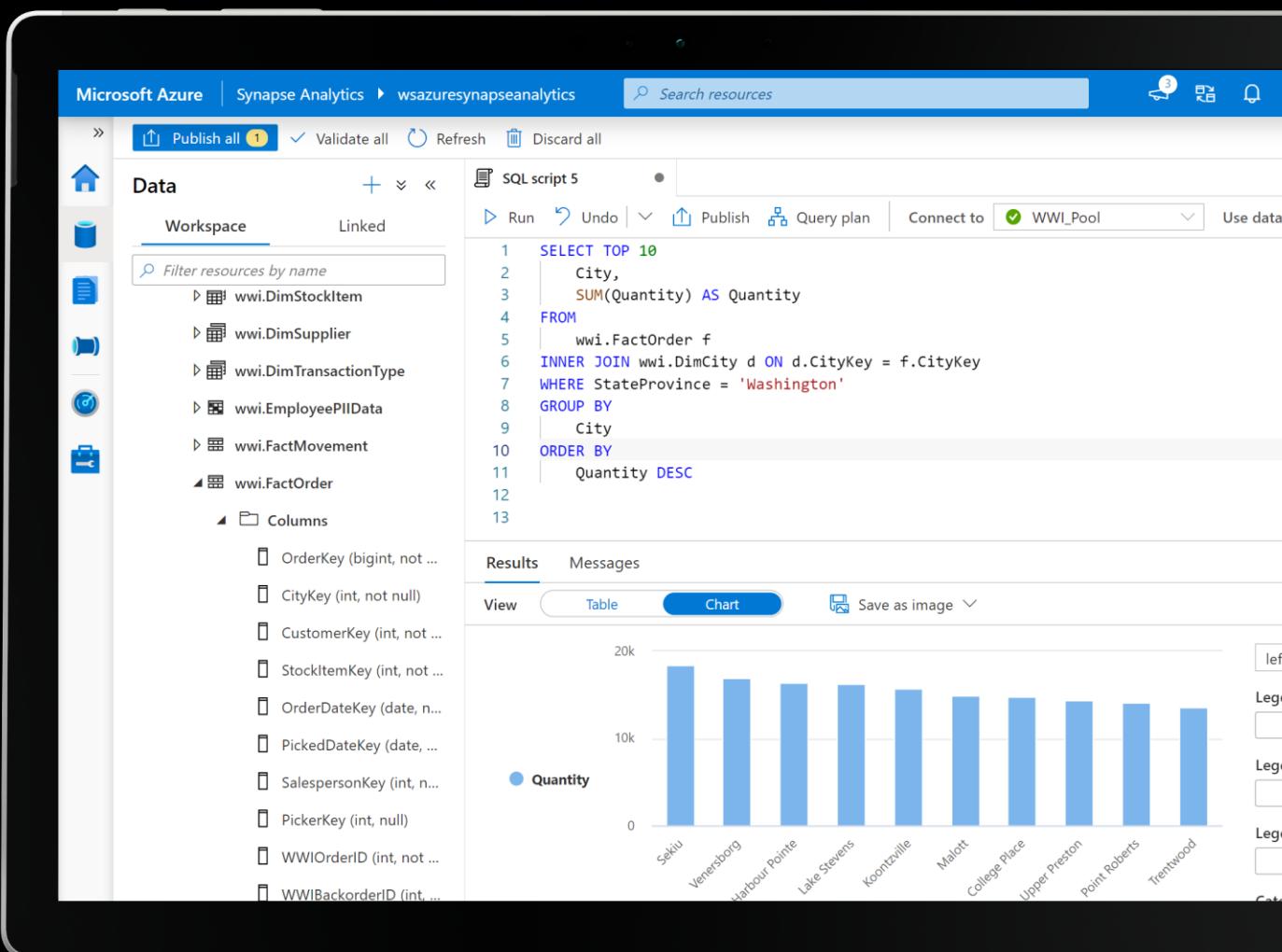
## SQL Editor

Automatic code completion (Intellisense)

Script collaboration within the Workspace

Built-in visualizations

Easily switch between clusters



The screenshot shows the Microsoft Azure Synapse Analytics Data workspace interface. On the left, there's a sidebar with icons for Home, Databases, Tables, Dimensions, Measures, and Workspaces. The 'Data' section is selected, showing a 'Workspace' view with a list of tables: wwi.DimStockItem, wwi.DimSupplier, wwi.DimTransactionType, wwi.EmployeePIIData, wwi.FactMovement, and wwi.FactOrder. Below this is a 'Columns' section listing various columns like OrderKey, CityKey, CustomerKey, StockItemKey, etc. On the right, a SQL script editor window titled 'SQL script 5' contains the following query:

```
1 SELECT TOP 10
2     City,
3     SUM(Quantity) AS Quantity
4 FROM
5     wwi.FactOrder f
6 INNER JOIN wwi.DimCity d ON d.CityKey = f.CityKey
7 WHERE StateProvince = 'Washington'
8 GROUP BY
9     City
10 ORDER BY
11     Quantity DESC
12
13
```

Below the script, there are tabs for 'Results' and 'Messages'. Under 'Results', a bar chart is displayed with the Y-axis labeled 'Quantity' ranging from 0 to 20k and the X-axis showing city names: Sekiu, Venerborg, Harbour Pointe, Lake Stevens, Koontzville, Malott, College Place, Upper Preston, Point Roberts, and Trentwood. The bars show varying quantities for each city.

# Develop hub – SQL scripts

## SQL Script

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

SQL script 6

Run Undo Publish Query plan Connect to Built-in Use database master

```
1 -- This is auto-generated code
2 SELECT
3     TOP 100 *
4 FROM
5     OPENROWSET(
6         BULK 'https://asadatalake01.dfs.core.windows.net/wwi-02/sale/Year=2019/Quarter=Q4/**',
7         FORMAT = 'PARQUET'
8     ) AS [result]
```

Results Messages

View Table Chart Export results ▾

Search

TransactionId	CustomerId	ProductId	Quantity
71ad8451-9ab7...	11	1043	3
71ad8451-9ab7...	11	2269	3
71ad8451-9ab7...	11	551	2
71ad8451-9ab7...	11	3998	2
71ad8451-9ab7...	11	2611	1
71ad8451-9ab7...	11	43	3

00:00:17 Query executed successfully.

SQL script 6

Run Undo Publish Query plan Connect to Built-in Use database master

```
1 -- This is auto-generated code
2 SELECT
3     TOP 100 *
4 FROM
5     OPENROWSET(
6         BULK 'https://asadatalake01.dfs.core.windows.net/wwi-02/sale/Year=2019/Quarter=Q4/**',
7         FORMAT = 'PARQUET'
8     ) AS [result]
```

Results Messages

View Table Chart Save as image ▾

Chart type: Line  
Category column: TotalAmount  
Legend (series) columns: ProfitAmount  
Legend position: bottom - center  
Legend (series) label:  
Category label:

Line chart showing the TotalAmount over time. The x-axis represents time in seconds, and the y-axis represents the TotalAmount. The data shows a highly volatile trend with frequent peaks and troughs, ranging from approximately 0 to 60.

Results Messages

View Table Export results ▾

Search

TransactionId	CustomerId	ProductId	Quantity
71ad8451-9ab7...	11	1043	3
71ad8451-9ab7...	11	2269	3
71ad8451-9ab7...	11	551	2
71ad8451-9ab7...	11	3998	2
71ad8451-9ab7...	11	2611	1
71ad8451-9ab7...	11	43	3

# Develop hub – Notebooks

The notebook experience supports PySpark, Spark, .NET Spark and SQL languages.

Output displays below the code cell with options to view data in tabular or chart format.

Multiple data export options are also available.

A screenshot of the Azure Data Studio Notebook interface. The top navigation bar shows "Notebook 3", "Run all", "Undo", "Publish", "Outline", "Attach to SparkPool01", "Language" (set to PySpark (Python)), "Variables", and a gear icon. A red box highlights the "Language" dropdown menu which includes PySpark (Python), PySpark (Python), Spark (Scala), .NET Spark (C#), and Spark SQL. Below the menu is a code cell with two lines of PySpark code: `df = spark.read.load('abfss://wwi-02@asadatalake01.dfs.core.windows.net/sale/YearMonth=12/Day=20191201/sale-small-20191201-snappy.parquet', format='parquet')` and `display(df.limit(10))`. A green checkmark indicates the command was executed in 1 second. The output area has tabs for "View", "Table" (selected), and "Chart". An "Export results" button is also present. The data is displayed in a table:

TransactionId	CustomerId	ProductId	Quantity	Price	TotalAmount
9453b35d-3f0d-44c8-bc5d-482a...	5	1588	1	20.1900000000000000	20.1900000000000000
9453b35d-3f0d-44c8-bc5d-482a...	5	4882	2	29.3700000000000000	58.7400000000000000
9453b35d-3f0d-44c8-bc5d-482a...	5	3283	2	28.3300000000000000	56.6600000000000000
9453b35d-3f0d-44c8-bc5d-482a...	5	4777	2	28.3500000000000000	56.7000000000000000
9453b35d-3f0d-44c8-bc5d-482a...	5	4908	2	35.4000000000000000	70.8000000000000000
9453b35d-3f0d-44c8-bc5d-482a...	5	4479	4	30.4200000000000000	121.6800000000000000
9453b35d-3f0d-44c8-bc5d-482a...	5	2779	2	14.5000000000000000	29.0000000000000000
9453b35d-3f0d-44c8-bc5d-482a...	5	1917	3	26.6700000000000000	80.0100000000000000
9453b35d-3f0d-44c8-bc5d-482a...	5	1382	3	26.7700000000000000	80.3100000000000000
9453b35d-3f0d-44c8-bc5d-482a...	5	1664	3	24.9900000000000000	74.9700000000000000

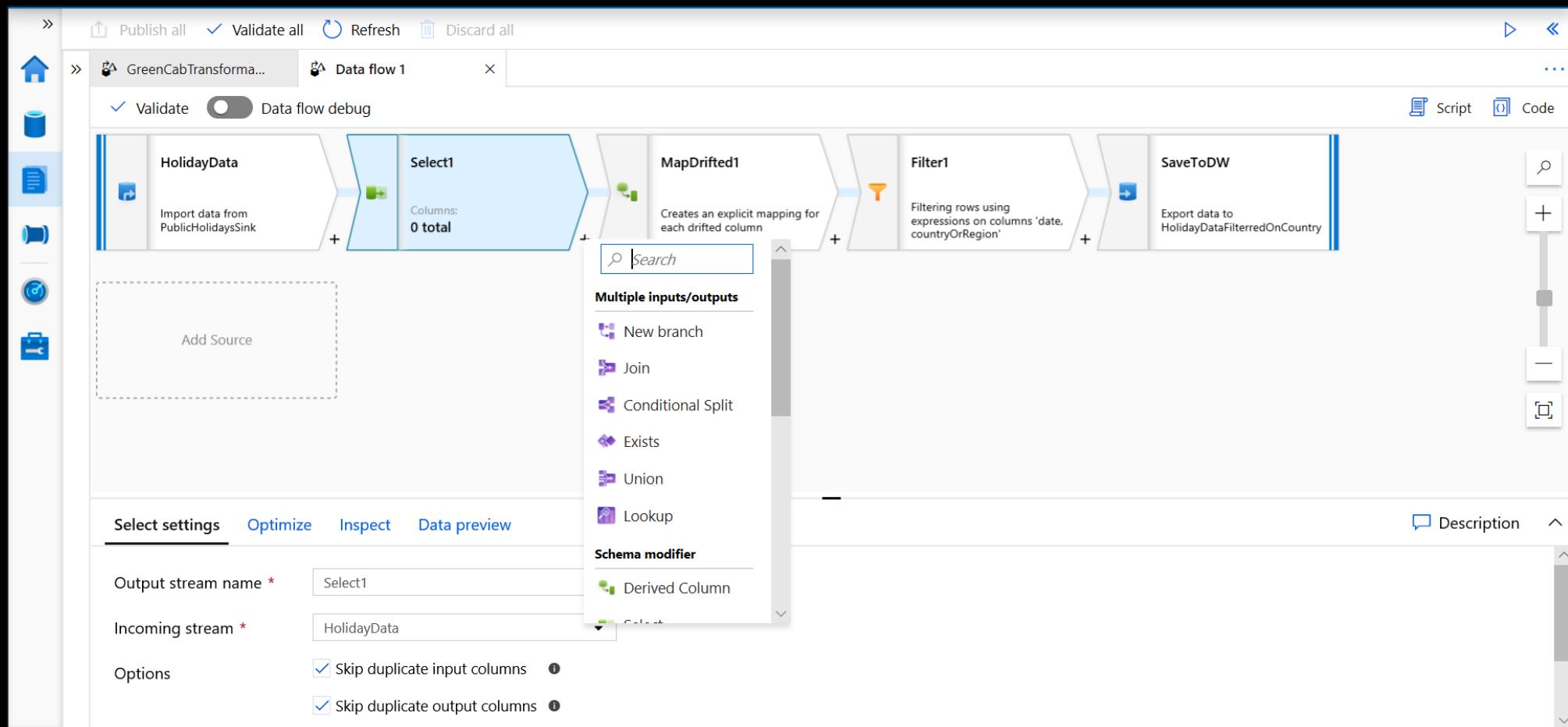
A screenshot of the Azure Data Studio Notebook interface showing the "Export results" dropdown menu. The menu includes options for CSV, JSON, and XML. The data is displayed in a table:

TransactionId	CustomerId	ProductId
9453b35d-3f0d-44c8-bc5d-482a...	5	1588
9453b35d-3f0d-44c8-bc5d-482a...	5	4882

# 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

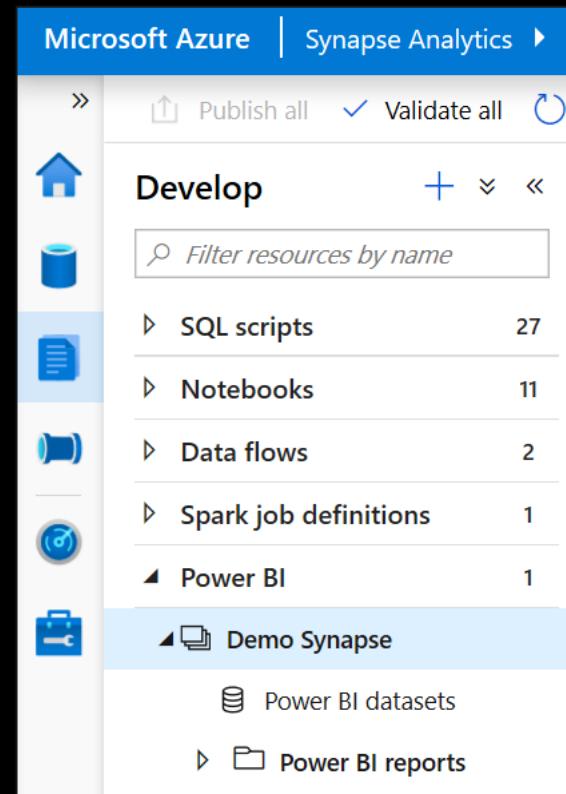
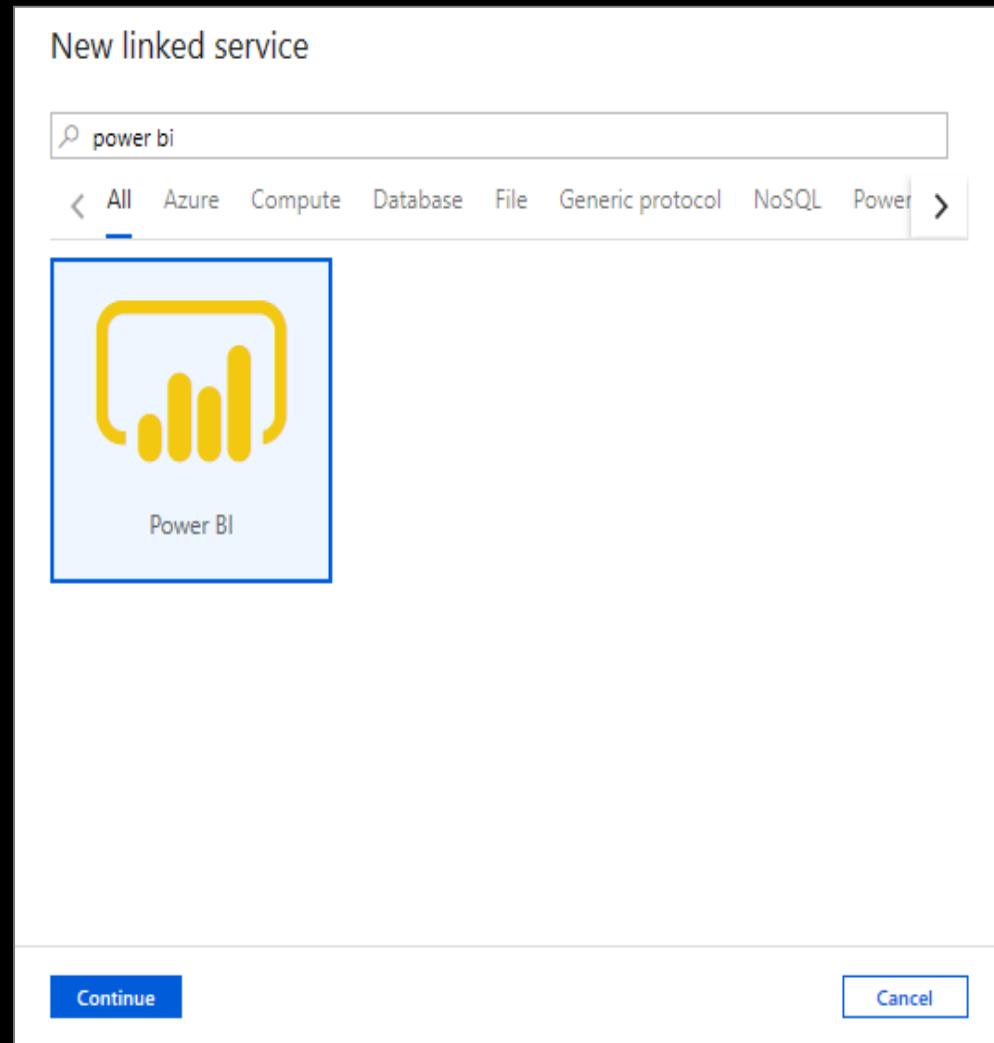
## Overview

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

Microsoft Azure | Synapse Analytics > gamingtelemetry

Search resources

prlangad@microsoft.com MICROSOFT

Publish all Validate all Refresh Discard all

Develop

Filter resources by name

- SQL scripts 9
- Notebooks 6
- Data flows 1
- Power BI 1
- gaming-telemetry
  - Power BI datasets
  - Power BI reports
    - Report

Report

File View Ask a question Explore Text box Shapes Buttons Visual interactions Refresh Duplicate this page Save

GAME STUDIO

Select a Platform: Console

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

Users (Forecast) 7,361,707 7,346,291 Last month

Extra Users 252.8K +3.4% Users Increase

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
Total	7,346.3K	7,361.7K	252.8K

"What If" Analysis Forecast

Users Forecast

Date: Aug 2019 - Nov 2019

Total Users 24.5M

Tabular Map Forecast Extra Users

Historical Forecast Predictions +

VISUALIZATIONS

FIELDS

Search

agegroup forecast historical platform predictions realtime regions scenario weekdays

Add data fields here

DRILL THROUGH

Cross-report Off On

Keep all filters On

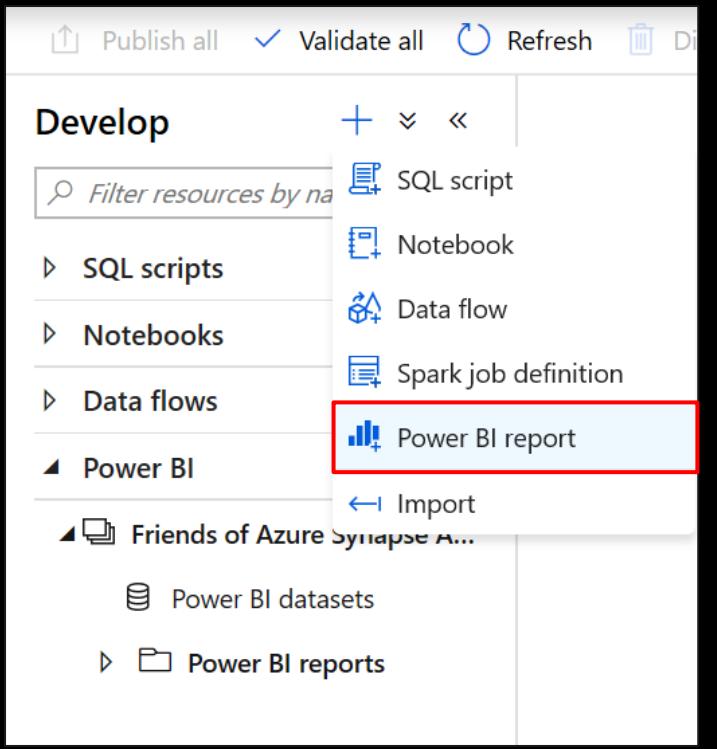
Add drill-through fields here

# Develop Hub – Power BI

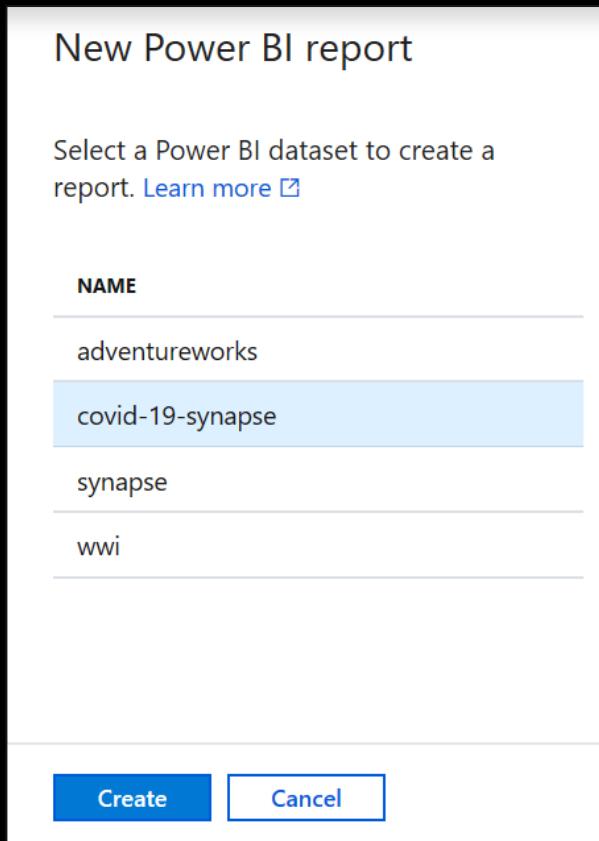
Create new reports from existing published Power BI datasets

Create new Power BI datasets

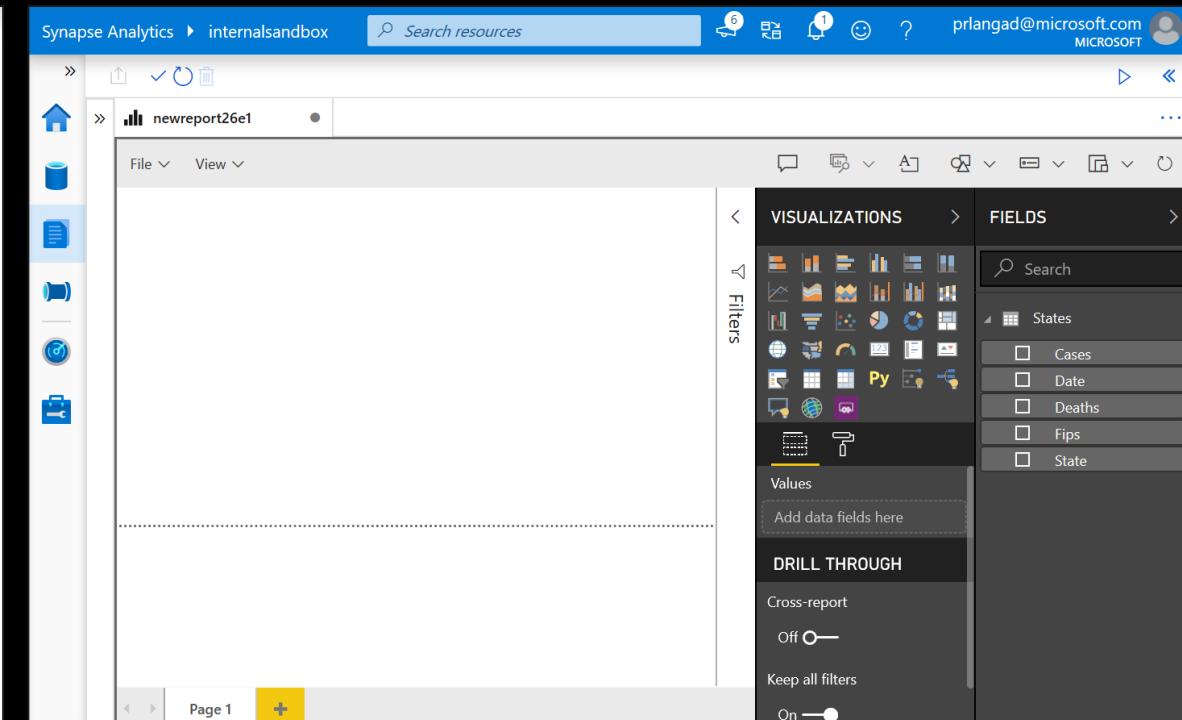
1



2



3



# Develop hub – Power BI

Edit reports in Synapse workspace

Microsoft Azure | Synapse Analytics > gamingtelemetry

Search resources

prlangad@microsoft.com MICROSOFT

Publish all Validate all Refresh Discard all

Develop

Filter resources by name

- SQL scripts 9
- Notebooks 6
- Data flows 1
- Power BI 1
- gaming-telemetry
  - Power BI datasets
  - Power BI reports
    - Report

Report

File View Ask a question Explore Text box Shapes Buttons Visual interactions Refresh Duplicate this page Save

### GAME STUDIO

Select a Platform: Console

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

Users (Forecast) 7,361,707 7,346,291 Last month

Extra Users 252.8K +3.4% Users Increase

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
Total	7,346.3K	7,361.7K	252.8K

"What If" Analysis Forecast

Users Forecast

Date: Aug 2019 - Nov 2019

Total Users 24.5M

Historical Forecast Predictions

VISUALIZATIONS

FIELDS

Filters

Values

Add data fields here

DRILL THROUGH

Cross-report Off

Keep all filters On

Add drill-through fields here

The screenshot shows the Microsoft Azure Synapse Analytics Develop hub interface. The main area displays a Power BI report titled "GAME STUDIO". The report includes a section for "What If..." analysis where users can increase free game addons by 1, showing a forecast of 7,361,707 users (7,346,291 last month) and an extra 252.8K users (+3.4% increase). It also features a "Total Users vs 'What If' Analysis" table and a "What If" Analysis Forecast chart showing user growth from August 2019 to November 2019. The left sidebar lists the report's structure, and the right sidebar provides options for visualizations and 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 like SQL scripts, Notebooks, Data flows, and Power BI datasets. Under Power BI, 'Power BI reports' is expanded, and 'Report' is selected. A red box highlights the 'Save' button in the top right corner of the main report area. A red arrow points from a callout box containing the text 'Publish changes by simply saving the report in workspace' to this 'Save' button.

**Save**  
Save this report

**PUBLISH CHANGES**  
Publish changes by simply saving the report in workspace

**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	1,268.5K	1,319.0K	90.7K
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
>60	146.7K	155.1K	13.3K
EMEA	844.9K	876.7K	60.7K
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	7,346.3K	7,613.6K	504.8K

**"What If" Analysis Forecast**

Users (purple line) and Forecast (grey line) over time from Aug 2019 to Nov 2019.

**VISUALIZATIONS > FIELDS >**

**VALUES**  
Add data fields here

**DRILL THROUGH**  
Cross-report  
Off —  
Keep all filters  
On —  
Add drill-through fields here

# CI/CD

Commit artifacts to source-controlled repository and operationalize release pipelines with Synapse deployment task in Azure DevOps or GitHub actions

→ ⏪ 🔒 <https://github.com/SynapseTestDemo/synapsetestdemo-ws-01/tree/dev>

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

README.md

**synapsetestdemo-ws-01**

↑ Synapse deployment v2 > Release-13

Pipeline Variables History + Deploy Cancel Refresh Edit ...

**Release**

Manually triggered by Priyanka Langade 11/16/2020, 11:02 PM

Artifacts

azuresynapsesdev  
0c8b1a872  
branch-retail-12

**Stages**

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

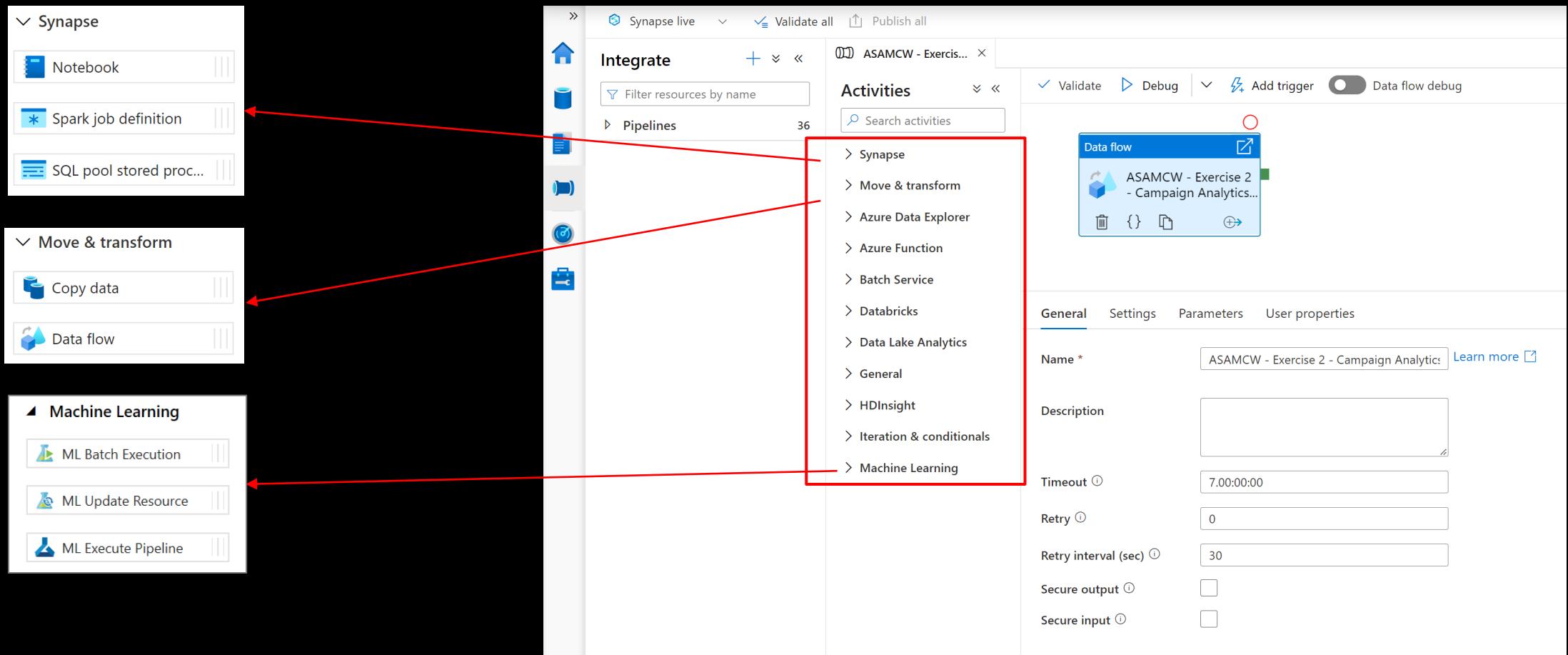
**Usage**

```
uses: Azure/synapse-workspace-deployment
with:
  TargetWorkspaceName: 'targetworkspace'
  TemplateFile: './TemplateForWorkspace.json'
  ParametersFile: './TemplateParametersForWorkspace.json'
  environment: 'Azure Public'
  resourceGroup: 'myresourcegroup'
  clientId: ${{ secrets.CLIENTID }}
  clientSecret: ${{ secrets.CLIENTSECRET }}
  subscriptionId: ${{ secrets.SUBID }}
  tenantId: ${{ secrets.TENANTID }}
```

# Integrate hub

Pipelines provide the ability to create pipelines to ingest, transform and load data with 100+ inbuilt connectors.

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



# 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

The screenshot shows the Microsoft Azure Synapse Analytics Monitor hub. The top navigation bar includes the Microsoft Azure logo and the text "Synapse Analytics". On the left, there's a vertical sidebar with icons for Analytics pools, SQL pools, Apache Spark pools, Data Explorer pools (preview), SQL requests, KQL requests, Apache Spark applications, Data flow debug, Pipeline runs, Trigger runs, and Integration runtimes. The "Pipeline runs" item is highlighted with a light gray background. The main content area is currently empty, showing a placeholder message: "Monitor your pipeline runs in one place. Click here to learn more about pipeline monitoring." At the bottom right, there are "Get started" and "View documentation" buttons.

# 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 contains a navigation menu with the following items:

- Synapse live
- Validate all
- Publish all (with a count of 1)
- Analytics pools (selected)
- SQL pools (highlighted in grey)
- Apache Spark pools
- Data Explorer pools (preview)
- External connections
- Linked services
- Azure Purview
- Integration
- Triggers
- Integration runtimes
- Security
- Access control
- Credentials
- Managed private endpoints
- Code libraries
- Workspace packages
- Source control
- Git configuration

The main content area is titled "SQL pools" and displays the following information:

The serverless SQL pool, Built-in, is immediately available for your workspace. Dedicated SQL pools can be configured to adapt to team or organization needs.

+ New    Refresh

Filter by name

Showing 1-5 of 5 items (1 Serverless, 4 Dedicated)

Name	Type	Status	Size
Built-in	Serverless	Online	Auto
SQLPool01	Dedicated	Online	DW300c
SQLPool02	Dedicated	Paused	DW500c
SQLPool03	Dedicated	Paused	DW500c
SQLPool04	Dedicated	Paused	DW300c

# Manage hub – 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 > asaworkspace01

Synapse live Validate all Publish all

Analytics pools SQL pools Apache Spark pools Data Explorer pools (preview)

External connections **Linked services** New

Filter by name Showing 1 - 32 of

Name	Icon
asaaml01	Power BI
asacosmosdk	Presto (Preview)
asacosmosdk	QuickBooks (Preview)
REST	SAP BW Open Hub
SAP Cloud For Customer	SAP ECC
SAP HANA	SAP HANA via MDX

Continue Cancel

Linked services are much like connection strings, which define the

# Manage hub – 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

The screenshot shows the Azure Synapse Analytics portal's 'Triggers' management screen. On the left, there's a sidebar with various options like 'Analytics pools', 'SQL pools', 'Apache Spark pools', etc., and a 'Triggers' link which is highlighted with a red box. In the main area, there's a heading 'Triggers' with the sub-instruction 'To execute a pipeline set the trigger. Triggers'. Below it is a '+ New' button, also highlighted with a red box. To the right of the '+ New' button is a detailed configuration panel titled 'New trigger' with fields for 'Name' (set to 'Trigger 1'), 'Description', 'Type' (set to 'Schedule'), 'Start date' (set to '01/24/2022 15:49:35'), 'Time zone' (set to 'Coordinated Universal Time (UTC)'), 'Recurrence' (set to 'Every 15 Minute(s)'), and other options like 'Specify an end date' and 'Annotations'. At the bottom of the main screen, there's a table listing two triggers: 'Blob-storage-trigger-01' (Storage events, Stopped) and 'cdm-trigger' (Storage events, Started).

Name	Type	Status
Blob-storage-trigger-01	Storage events	Stopped
cdm-trigger	Storage events	Started

# Manage hub – 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. At the top, there are buttons for 'Synapse live', 'Validate all', 'Publish all', 'Add' (highlighted with a red box), 'Refresh', and 'Remove access'. Below these are filters for 'Type : All', 'Role : All', and 'Scope : All'. A table lists 20 role assignments, showing columns for Name, Type, and a checkbox for each user. The users listed are Ciprian Jichici, Kyle Bunting, Zoiner Tejada, and Roxana Goidaci, all categorized as 'User'. Red arrows point from the 'Add' button and the 'Access control' link in the sidebar to two separate 'Add role assignment' modals.

Name	Type
Ciprian Jichici ciprian@solliance.net	User
Kyle Bunting kyle@solliance.net	User
Zoiner Tejada ZoinerTejada@solliance.net	User
Roxana Goidaci roxanag@solliance.net	User

### 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 \*

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

### 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 \*

Item \*

Role \*

Select user \*

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

# Manage hub – Source control

## Overview

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

Microsoft Azure | Synapse Analytics > wsazuresynapseanalytics

Synapse live Validate all Publish all

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

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

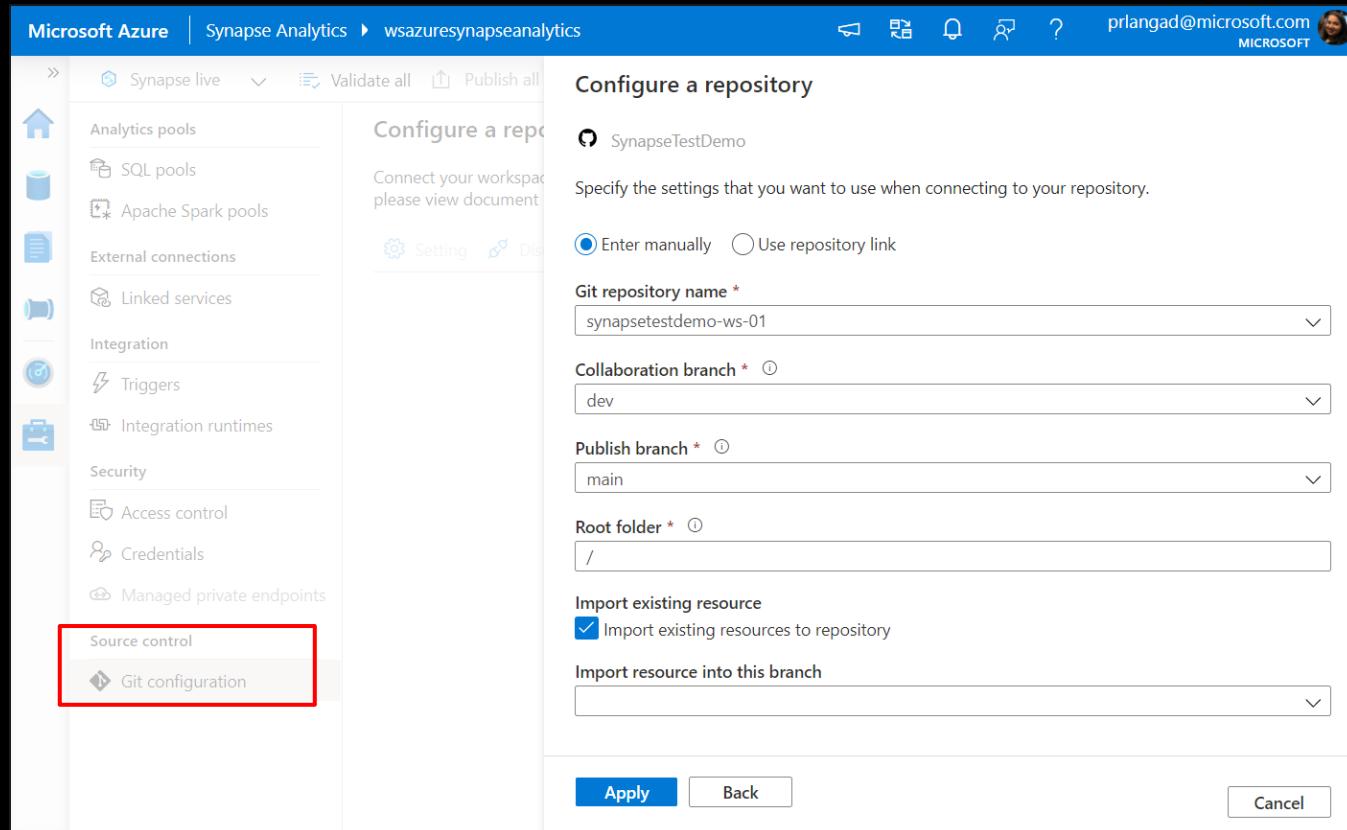
Publish branch \* main

Root folder \* /

Import existing resource Import existing resources to repository

Import resource into this branch

Apply Back Cancel



Microsoft Azure | Synapse Analytics > wsazuresynapseanalytics

Validate all Commit all Publish

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

Setting Disconnect

Repository type GitHub

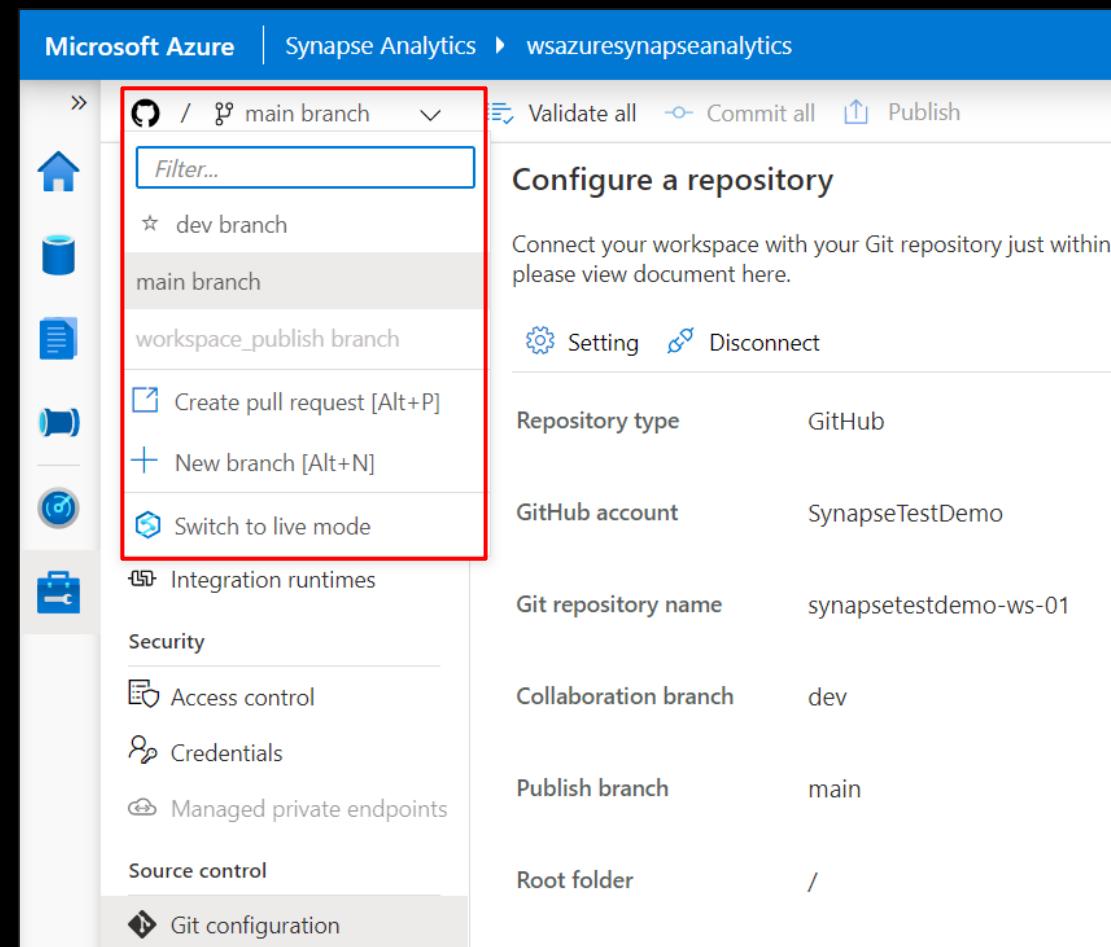
GitHub account SynapseTestDemo

Git repository name synapsetestdemo-ws-01

Collaboration branch dev

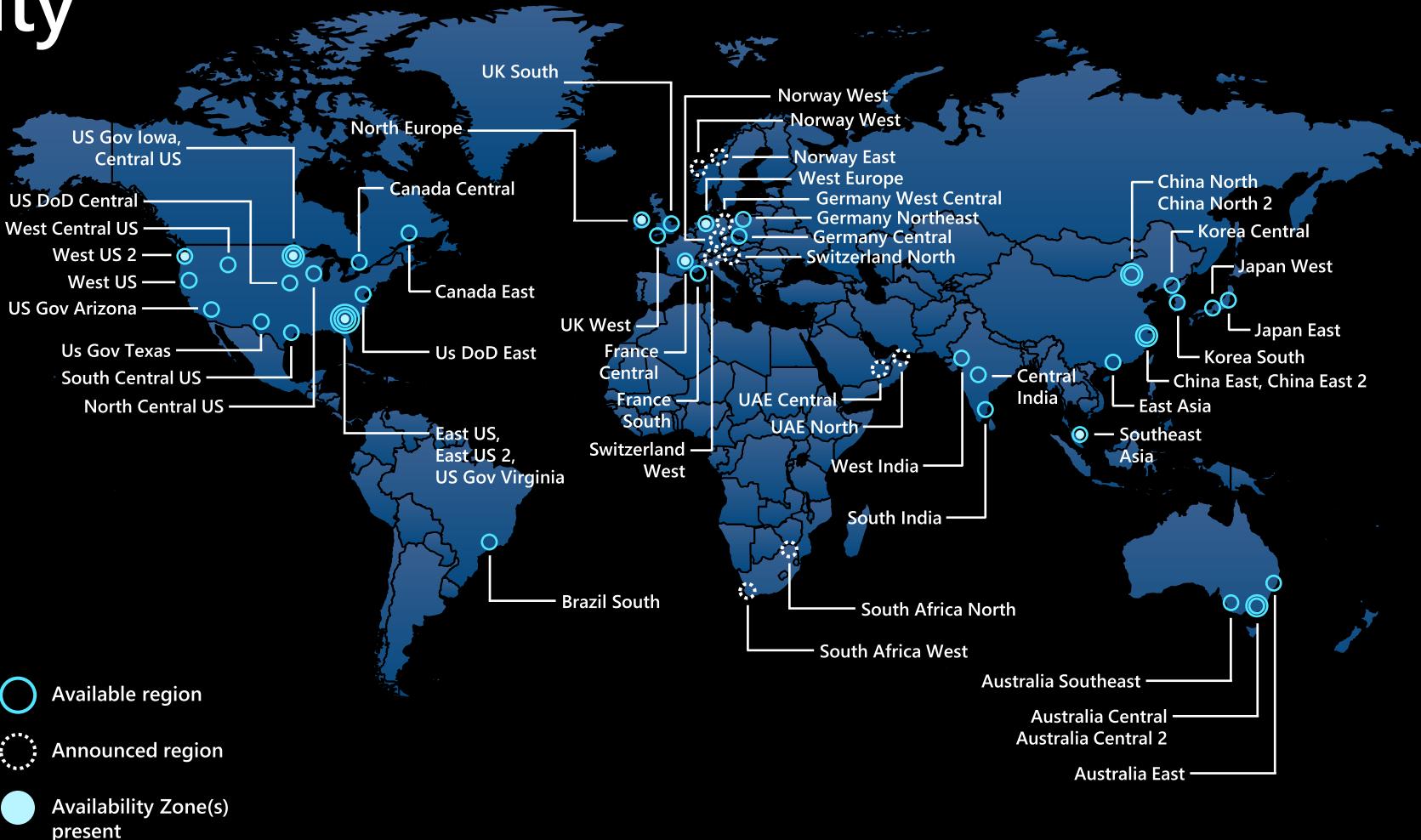
Publish branch main

Root folder /



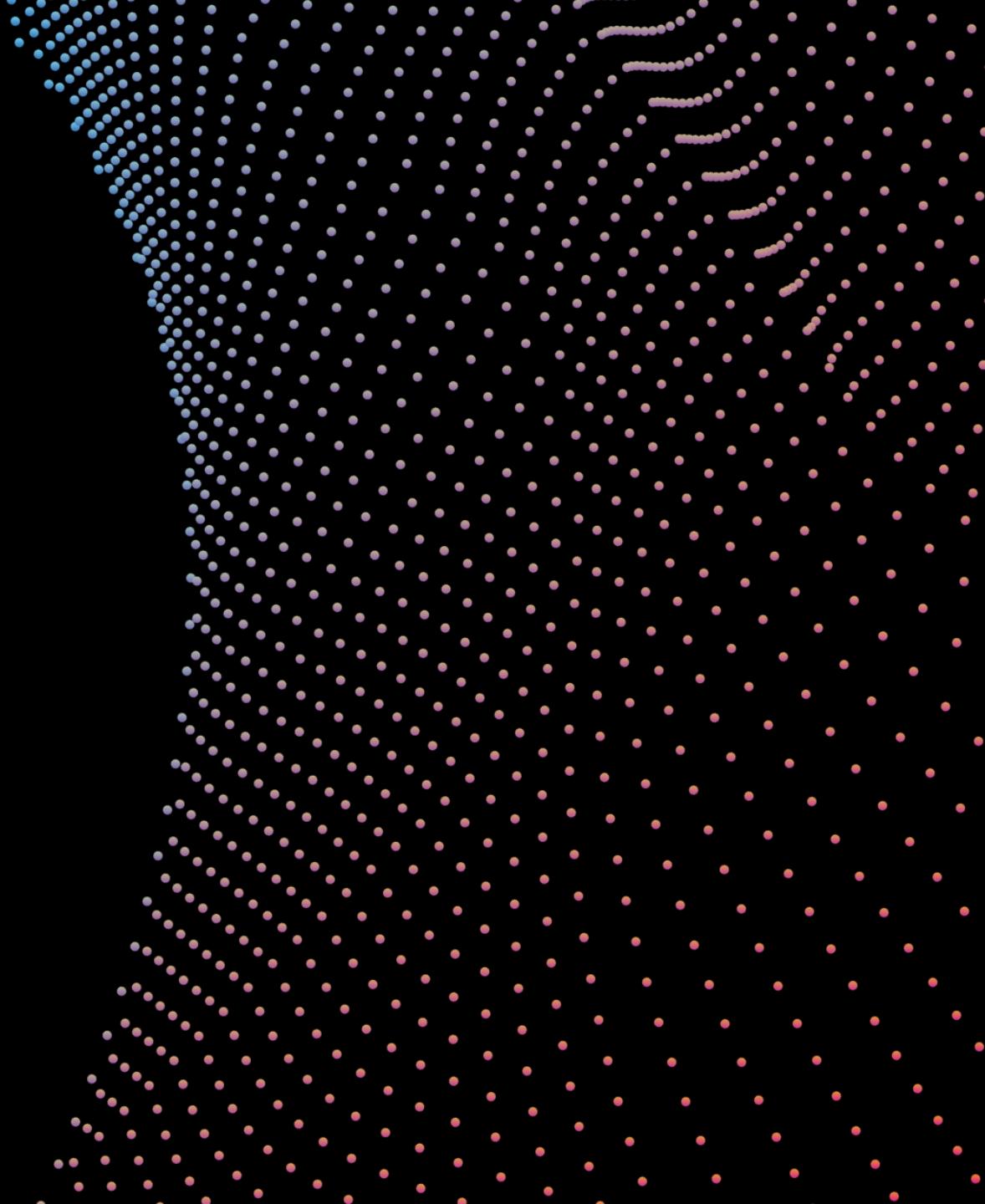
# Azure Synapse regional availability

Australia Southeast	Korea Central
Australia East	North Central US
Brazil South	North Europe
Canada Central	South Africa North
Canada East	South Central US
Central India	Southeast Asia
Central US	Switzerland North
East Asia	UK West
East US	UK South
East US 2	West Central US
France Central	West Europe
Germany West Central	West US
Japan East	West US 2
Japan West	



\* Region availability changes frequently, see documentation to confirm

# Data Loading & Data Lake Organization



# Agenda

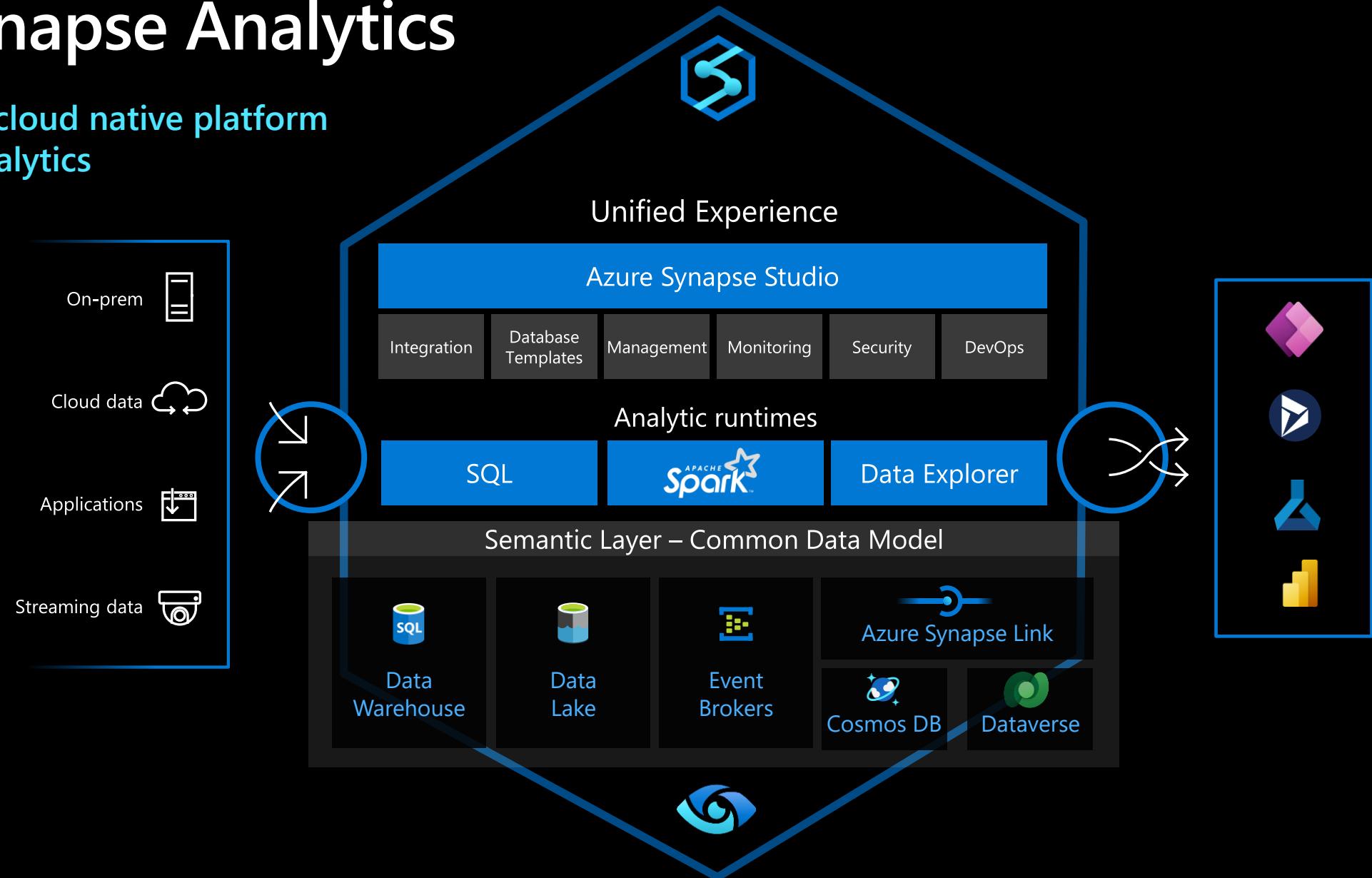
---

- 1) Integration
- 2) Ingest data to tables
- 3) Best practices



# Azure Synapse Analytics

The first **unified, cloud native platform** for converged analytics



# Analytical runtimes

Analytic runtimes

SQL



Data Explorer

## Serverless SQL Pool

- flat file data exploration
- single file querying
- multi-file (same schema) / nested folder queries
- well-formatted data
- JSON, CSV, Parquet, Delta (parquet)
- logical data warehouse (Lake Database), shared metadata with Spark pool
- bootstrap lake database creation with industry specific database templates

## Dedicated SQL Pool

- traditional data warehouse
- MPP
- ability to query file based storage, external tables (CSV, Parquet, ORC)

## Spark Pool

- process non-standard files (hula hula)
- complicated data cleaning, transformations
- data science libraries and other popular frameworks (pypi)
- logical data warehouse (Lake database), shared metadata with serverless SQL pool
- bootstrap lake database creation with industry specific database templates

## Azure Data Explorer Pool

- Log data, streaming data and time series analytics
- Auto organization, cleaning, conversions, indexing

# Ingest Integration with Pipelines

# 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 ▶ asaworkspace01

Synapse live Validate all Publish all

Analytics pools SQL pools Apache Spark pools Data Explorer pools (preview)

External connections **Linked services**

Linked services are much like connection strings, which define the

+ New

Filter by name

Showing 1 - 32 of

Name	Icon	Description
asaaml01	Power BI icon	Power BI (Preview)
asacosmosdk	Presto (Preview) icon	Presto (Preview)
asacosmosdk	QuickBooks (Preview) icon	QuickBooks (Preview)
REST	SAP BW Open Hub icon	SAP BW Open Hub
SAP Cloud For Customer	SAP ECC icon	SAP ECC
SAP HANA	SAP HANA icon	SAP HANA

Continue Cancel

New linked service

PayPal (Preview) Phoenix PostgreSQL

Power BI Presto (Preview) QuickBooks (Preview)

SAP BW SAP BW via MDX SAP Cloud For Customer SAP ECC SAP HANA

# 100+ Connectors out of the box

Azure (18)	Database & DW (30)		File Storage (9)	File Format (8)	NoSQL (4)	Services and App (30)		Generic (4)
Blob storage	Amazon Redshift	Oracle	Amazon S3	AVRO	Cassandra	Amazon MWS	Oracle Service Cloud	Generic HTTP
Cosmos DB - SQL & MongoDB	DB2	Phoenix	File system	Binary	Couchbase	CDS for Apps	PayPal	Generic OData
Cognitive Services	Drill	PostgreSQL	FTP	Delimited Text	MongoDB	Concur	QuickBooks	Generic ODBC
Data Explorer	Google BigQuery	Presto	Google Cloud Storage	JSON	MongoDB Atlas	Dynamics 365	Salesforce	Generic REST
Data Lake Storage Gen1 & Gen 2	Greenplum	SAP BW Open Hub	HDFS	ORC		Dynamics AX	SF Service Cloud	SharePoint Online List
Database for MariaDB	HBase	SAP BW via MDX	SFTP	Parquet		Dynamics CRM	SF Marketing Cloud	
Database for MySQL	Hive	SAP HANA	Amazon S3 Compatible	Excel		Google AdWords	SAP C4C	Compute (7)
Database for PostgreSQL	Apache Impala	SAP table	HTTP	XML		HubSpot	SAP ECC	Azure Batch
Databricks Delta Lake	Informix	Spark	Oracle Cloud Storage		Jira	ServiceNow		Azure Data Lake Analytics
File Storage	MariaDB	SQL Server			Magento	Shopify		Azure Databricks
SQL Database	Microsoft Access	Sybase			Marketo	Square		Azure Function
SQL Database MI	MySQL	Teradata			Office 365	Web table		Azure HDInsight
SQL Data Warehouse	Netezza	Vertica			Oracle Eloqua	Xero		Azure Machine Learning
Search index	Snowflake	Google AdWords			Oracle Responsys	Zoho		
Table storage	Amazon RDS for Oracle	SQL Server Database MI			Power BI	Dataverse		
Key Vault	Amazon RDS for SQL Server				GitHub	Snowflake		

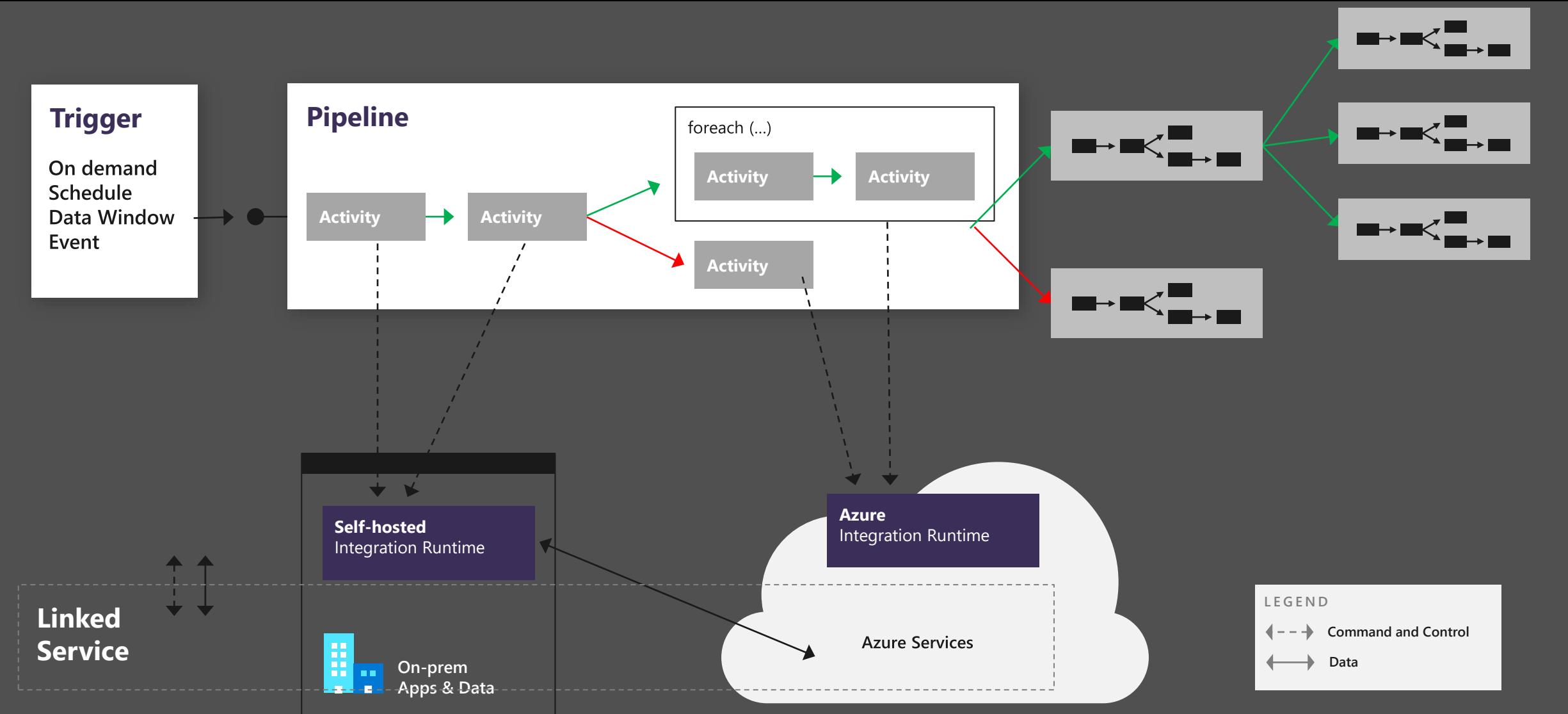
# Integration dataset

Integration 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 'Data' blade in the Azure portal. The left sidebar has tabs for 'Data' (selected), 'Workspace' (disabled), and 'Linked'. Under 'Data', there's a search bar 'Filter resources by name' and a list of data types: Azure Blob Storage (5), Azure Cosmos DB (2), Azure Data Explorer (1), Azure Data Lake Storage Gen2 (8), and Integration datasets (194). The 'Integration datasets' item is expanded, showing a list of datasets including 'AdeKustoSalesTelemetry', 'asal400\_campaign\_analytics\_source', 'asal400\_customerprofile\_cosmosdb', 'asal400\_december\_sales' (which is selected and highlighted in grey), 'asal400\_ecommerce\_userprofiles\_so...', 'asal400\_product\_blob', 'asal400\_saleheap\_asa', 'asal400\_sales\_adlsgen2', 'asal400\_userprofiles\_source', and 'asal400\_wwi02\_adls'. The main pane shows the details for the selected dataset 'asal400\_december\_sales'. It's a Parquet file located at 'asal400\_december\_sales'. Below this, the 'Connection' tab is selected, showing the 'Linked service' dropdown set to 'asadatalake01', a 'Test connection' button, and 'Edit', 'New', and 'Learn more' links. The 'Integration runtime' is set to 'LargeAzureComputeOptimizedIntegr...'. The 'File path' is 'wwi-02 / campaign-analytics / large-sale-december20', with 'Browse' and 'Preview data' buttons. The 'Compression type' is 'snappy'.

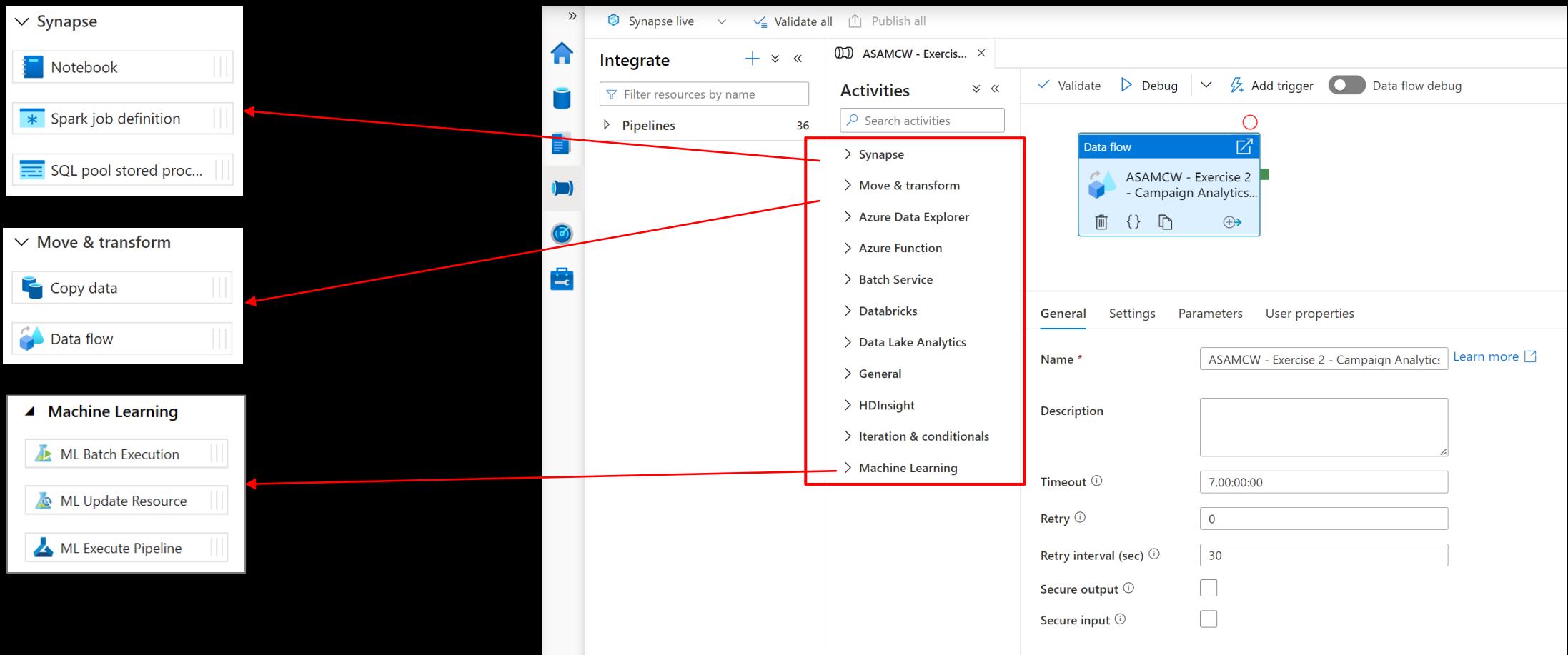
# Components of orchestration



# Pipelines

Create pipelines to ingest, transform and load data with 100+ inbuilt connectors.

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



# Pipelines

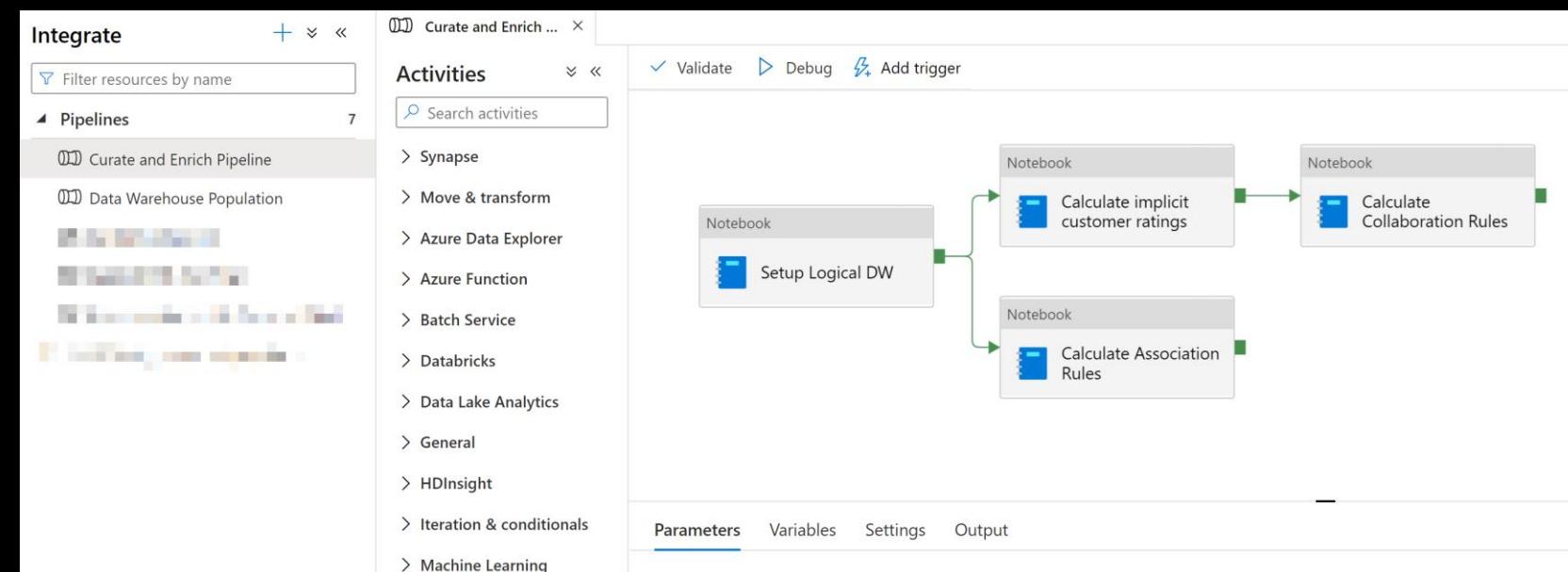
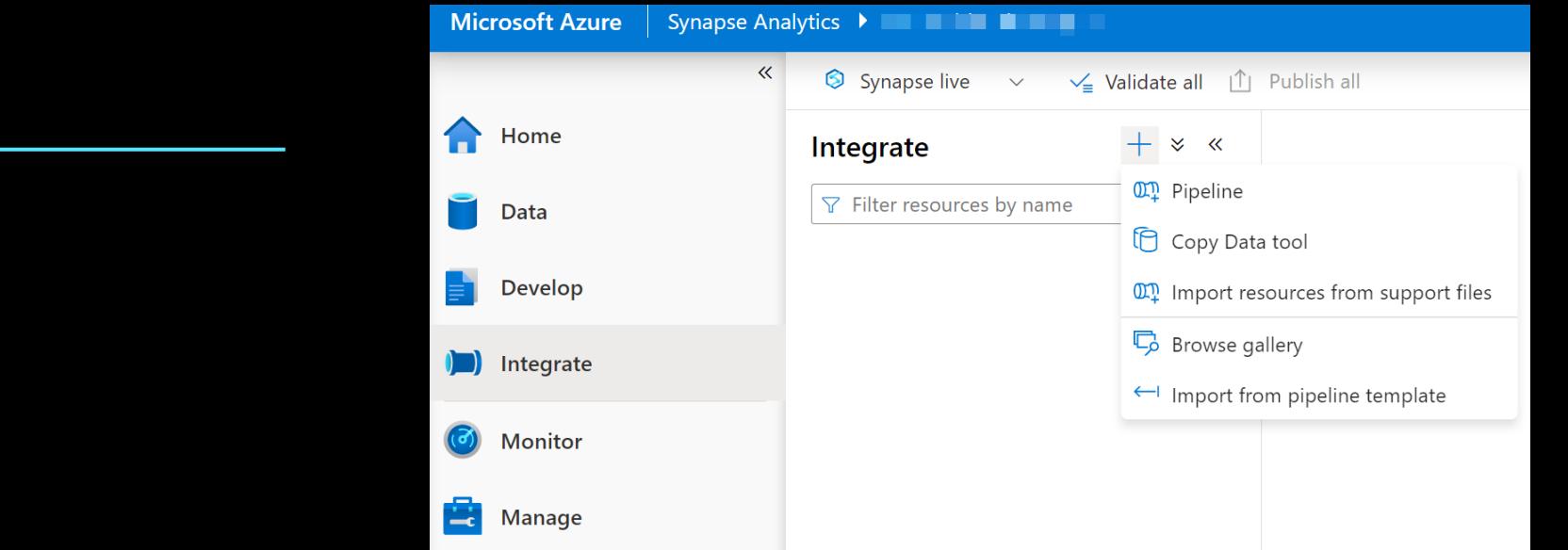
## Overview

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

- Load data by manual execution of pipeline or by orchestration.

## Benefits

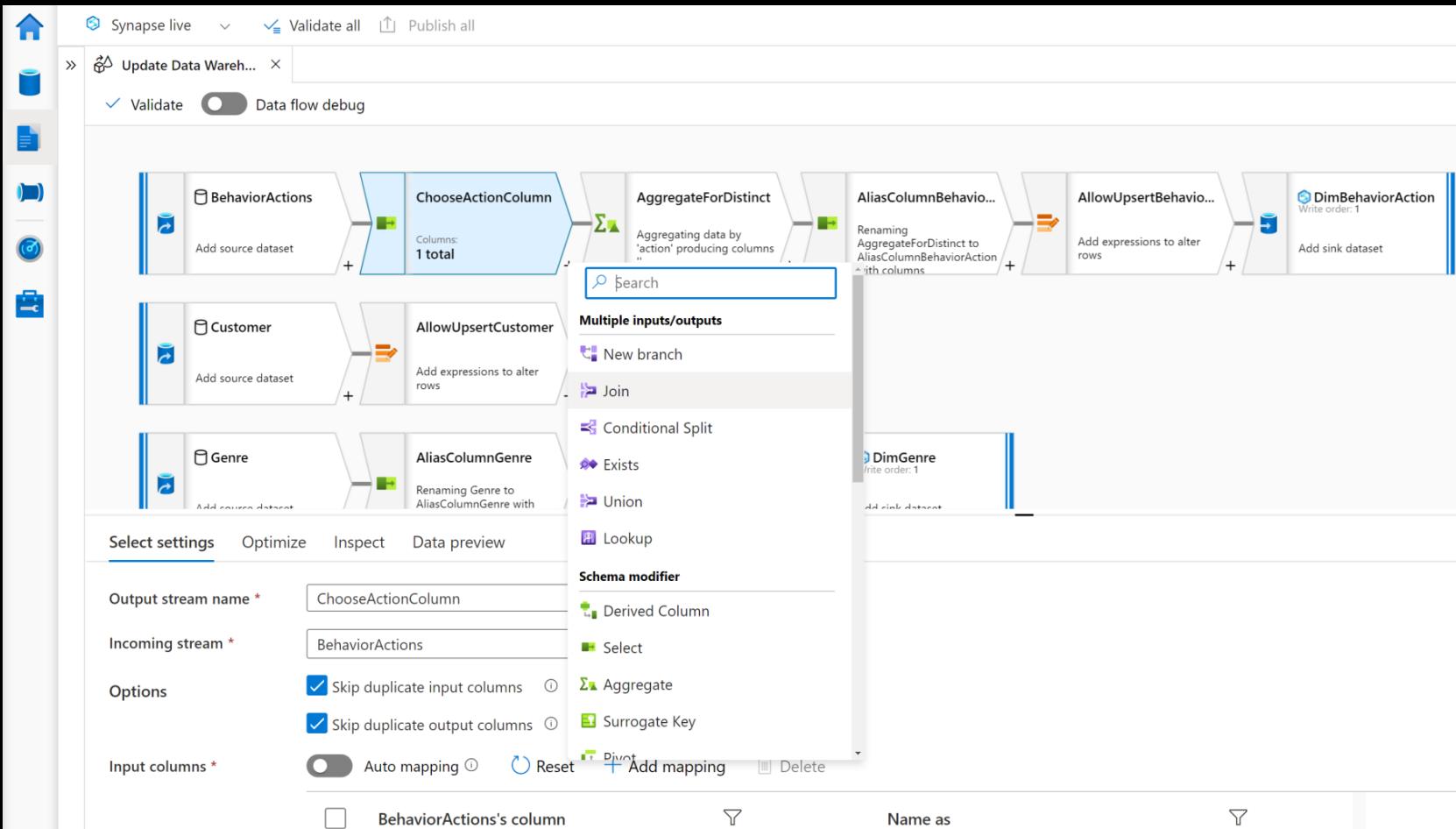
- Supports common loading patterns.
- Fully parallel loading into data lake or SQL tables.
- Graphical development experience.



# Data flows

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

Provides a code-free experience.



# Data flow capabilities



Handle upserts, updates, deletes on sql sinks



Add new partition methods



Add schema drift support



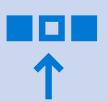
Add file handling (move files after read, write files to file names described in rows etc)



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



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



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



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

# Triggers

## Overview

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

Data Integration offers 4 trigger types as –

1. Schedule – gets fired at a schedule with information of start date, recurrence, end date
2. Event – gets fired on specified Storage event
3. Tumbling window – gets fired at a periodic time interval from a specified start date, while retaining state
4. Custom event – gets fired based on topics defined in Azure Event Grid

The screenshot shows the Azure Data Studio interface for managing triggers. On the left, there's a sidebar with various options like Analytics pools, SQL pools, Apache Spark pools, Data Explorer pools (preview), External connections, Linked services, Azure Purview, Integration, and Security. The 'Integration' section is expanded, and the 'Triggers' option is highlighted with a red box. In the main center area, there's a title 'Triggers' with the sub-instruction 'To execute a pipeline set the trigger'. Below this, there's a 'New' button, also highlighted with a red box. To the right of the 'New' button, there's a form for creating a new trigger, which includes fields for Name (Trigger 1), Description, Type (Schedule), Start date (01/24/2022 15:49:35), Time zone (Coordinated Universal Time (UTC)), Recurrence (Every 15 Minute(s)), and Annotations. At the bottom, there's a table listing existing triggers: 'Blob-storage-trigger-01' (Storage events, Status: Stopped) and 'cdm-trigger' (Storage events, Status: Started). A red arrow points from the 'New' button in the center to the 'New' button in the top-right trigger creation form.

Name	Type	Status
Blob-storage-trigger-01	Storage events	Stopped
cdm-trigger	Storage events	Started

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

# 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. Configurable TTL enables reuse of already provisioned resources.

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

In preview: Azure SSIS – lift and shift existing SSIS packages to execute in Azure

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', 'Data Explorer pools (preview)', 'External connections', 'Linked services', 'Azure Purview', 'Integration', 'Triggers', and 'Integration runtimes'. The 'Integration runtimes' item is highlighted with a red box. On the right, a list of existing runtimes is shown, with two items: 'AutoResolveIntegrationRuntime' and 'AutoResolveWithTTLSmall'. A red box highlights the '+ New' button. Below the list is a 'Filter by name' input field and a message indicating 'Showing 1 - 2 of 2 items'. A red arrow points from the 'Integration runtimes' menu item down to the 'Integration runtime setup' dialog. The 'Integration runtime setup' dialog has a title 'Integration runtime setup' and a sub-section 'Network environment'. It contains two options: 'Azure' (selected) and 'Self-Hosted'. A red box highlights the 'Azure' option. The 'Azure' section includes a description: 'Use this for running data flows, data movement, external and pipeline activities in a fully managed, serverless compute in Azure.' A red arrow points from the 'Azure' section to the 'Azure, Self-Hosted' option in the main list. The main list also includes 'Azure-SSIS (preview)' with a description: 'Lift-and-shift existing SSIS packages to execute in Azure.'

# Data movement with integration runtimes

## Scalable

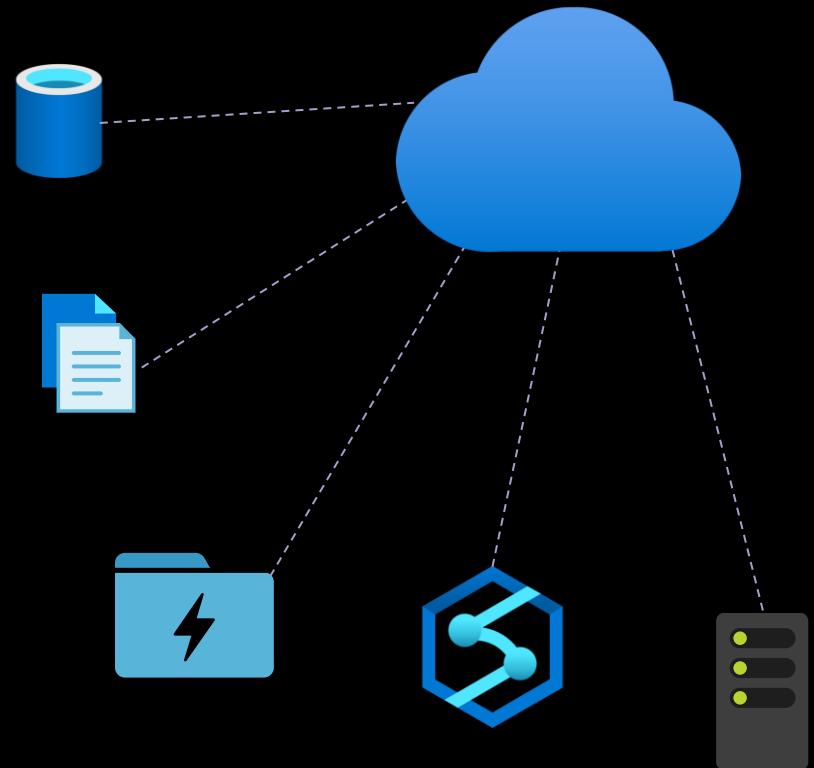
- per job elasticity
- Up to 4 GB/s

## Simple

- Visually author or via code (Python, .Net, etc.)
- Serverless, no infrastructure to manage

## Access all your data

- 100+ 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



# Pop Quiz 1

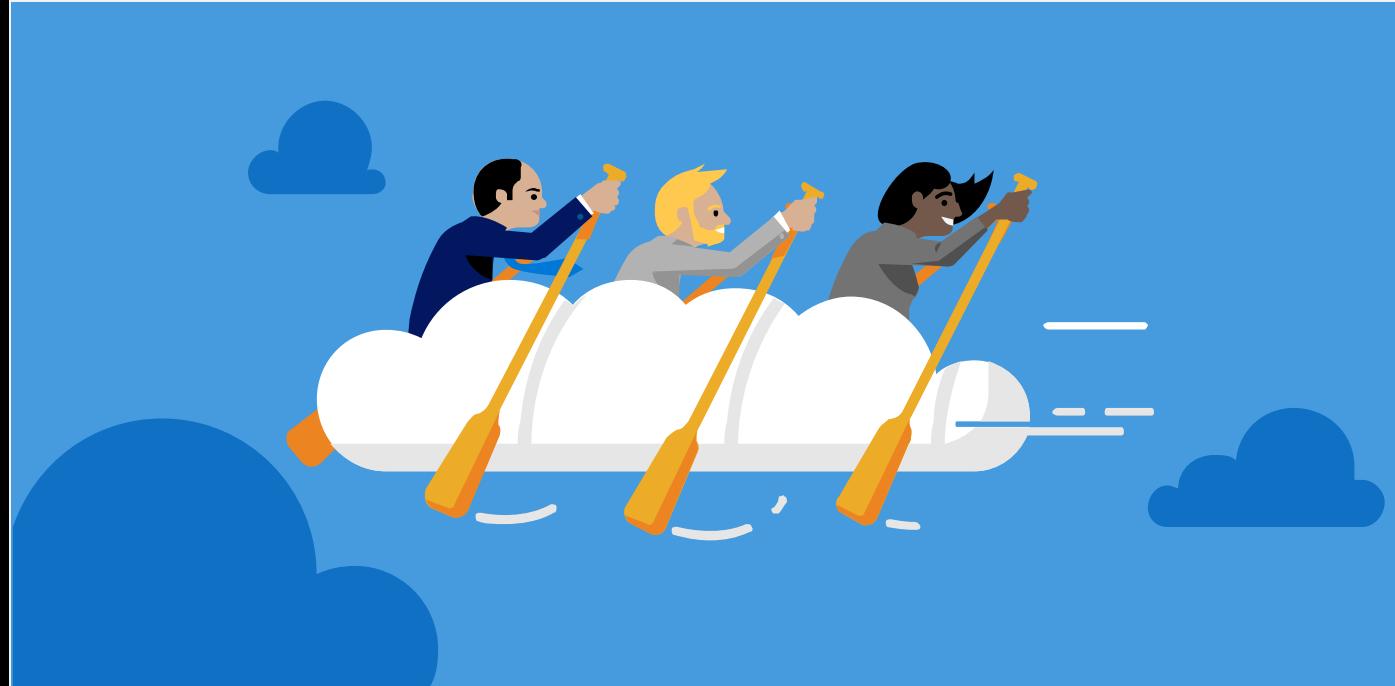
Which one of these is NOT a component of a Synapse pipeline?

A)  
I.R.

B)  
Linked  
Service

C)  
Table

D)  
Activity



# Pop Quiz 1

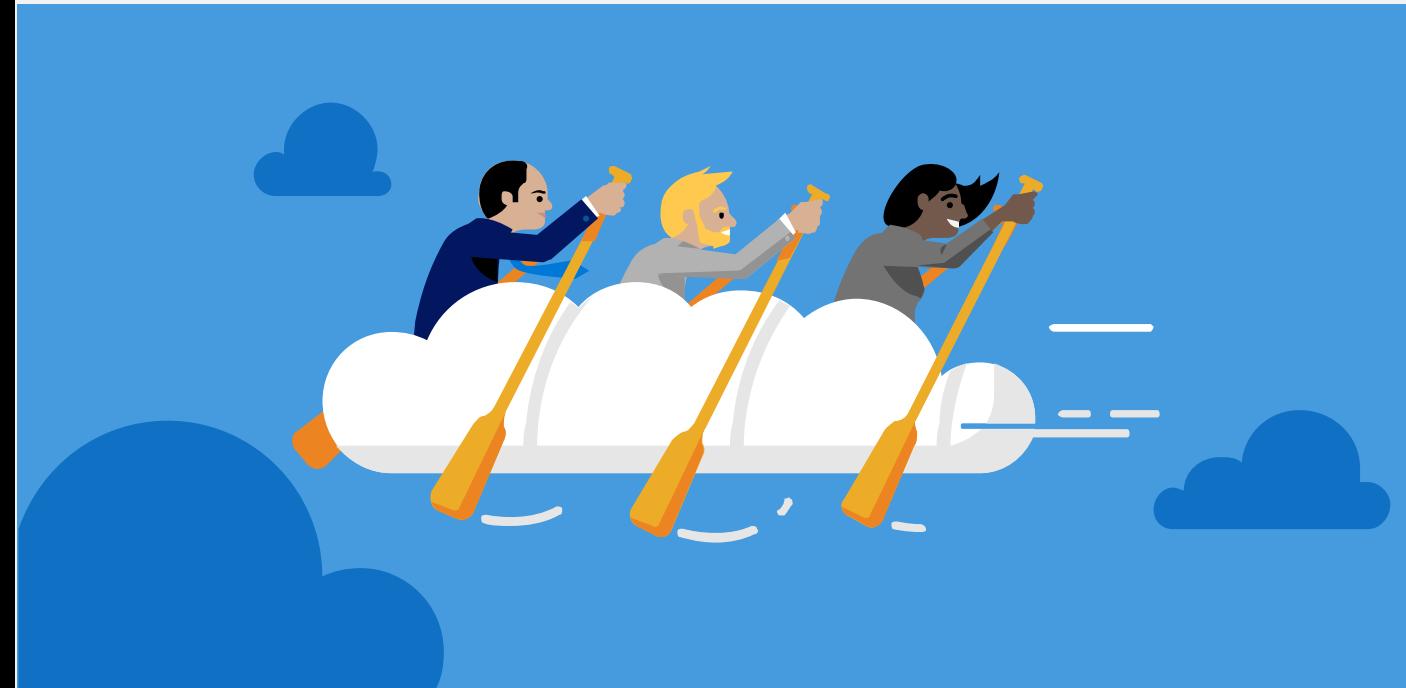
Which one of these is NOT a component of a Synapse pipeline?

A)  
I.R.

B)  
Linked  
Service

C)  
**Table**

D)  
Activity



# Ingest Files and Tables

# 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://XYZ.blob.core.windows.net/customerdatasets/test_1.txt'
WITH (
    FILE_TYPE = 'CSV',
    CREDENTIAL=(IDENTITY= 'Shared Access Signature',
SECRET='<Your_SAS_Token>'),
    FIELDQUOTE = """",
    FIELDTERMINATOR=';',
    ROWTERMINATOR='0X0A',
    ENCODING = 'UTF8',
    DATEFORMAT = 'ymd',
    MAXERRORS = 10,
    ERRORFILE = '/errorsfolder/'--path starting from the storage container,
    IDENTITY_INSERT
)
```

```
COPY INTO test_parquet
FROM 'https://XYZ.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 (Polybase)

## 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:

- Create Master Key
- Create Credentials
- Create External Data Source
- Create External Data Format
- 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 );
```

# Polybase vs COPY

## Polybase

- GA, stable
- Needs CONTROL permission
- Enables querying via external tables
- Challenges:
  - Row width (1 MB)
  - Delimiters in text
  - Fixed line delimiter
  - Code complexity

## Copy

- Relaxed permission
- No row width limit
- Supports delimiters in text
- Supports custom column and row delimiters

# Ingest Azure Synapse Data Explorer

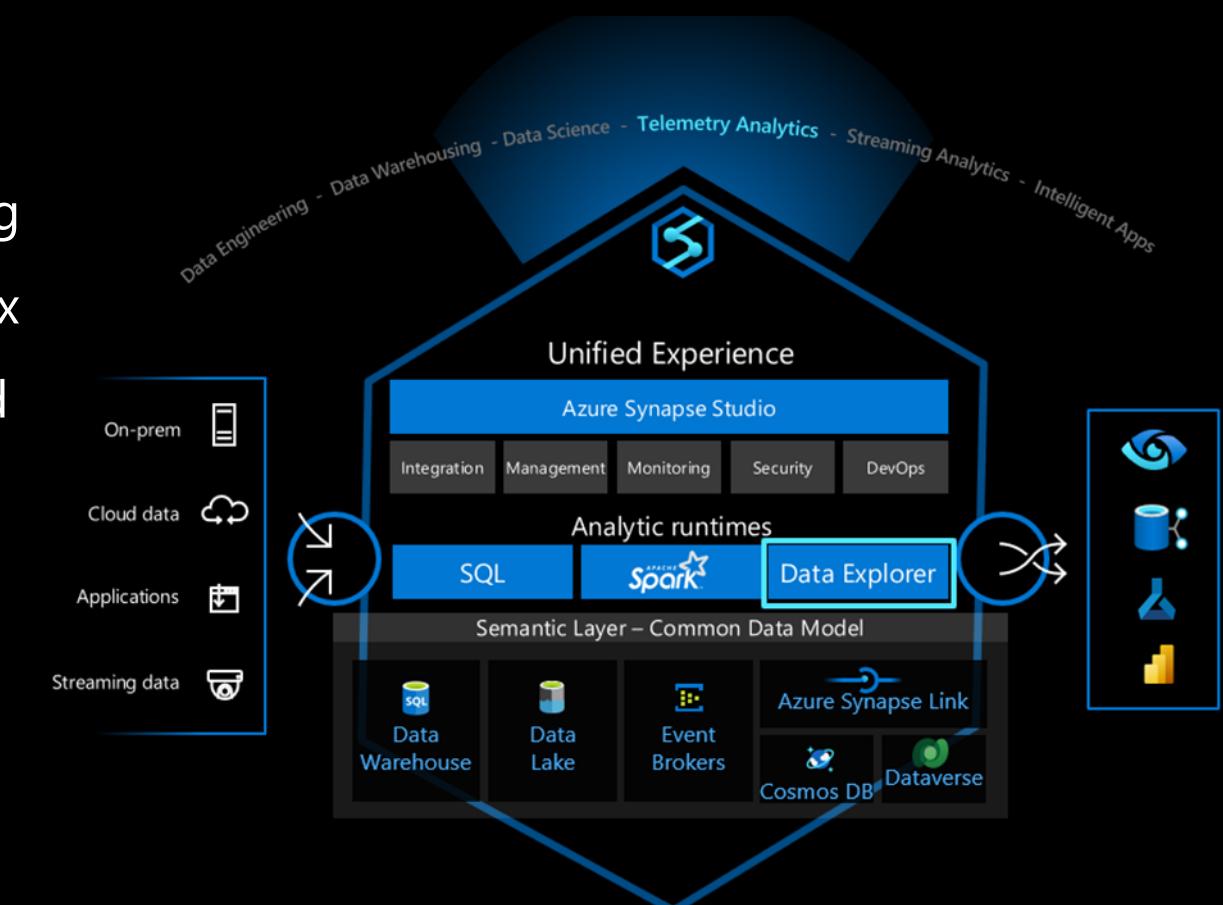
# Synapse Data Explorer

## Overview

Azure Synapse provides an interactive query experience optimized for efficient log analytics using powerful indexing technology to automatically index free-text and semi-structured data commonly found in telemetry data.

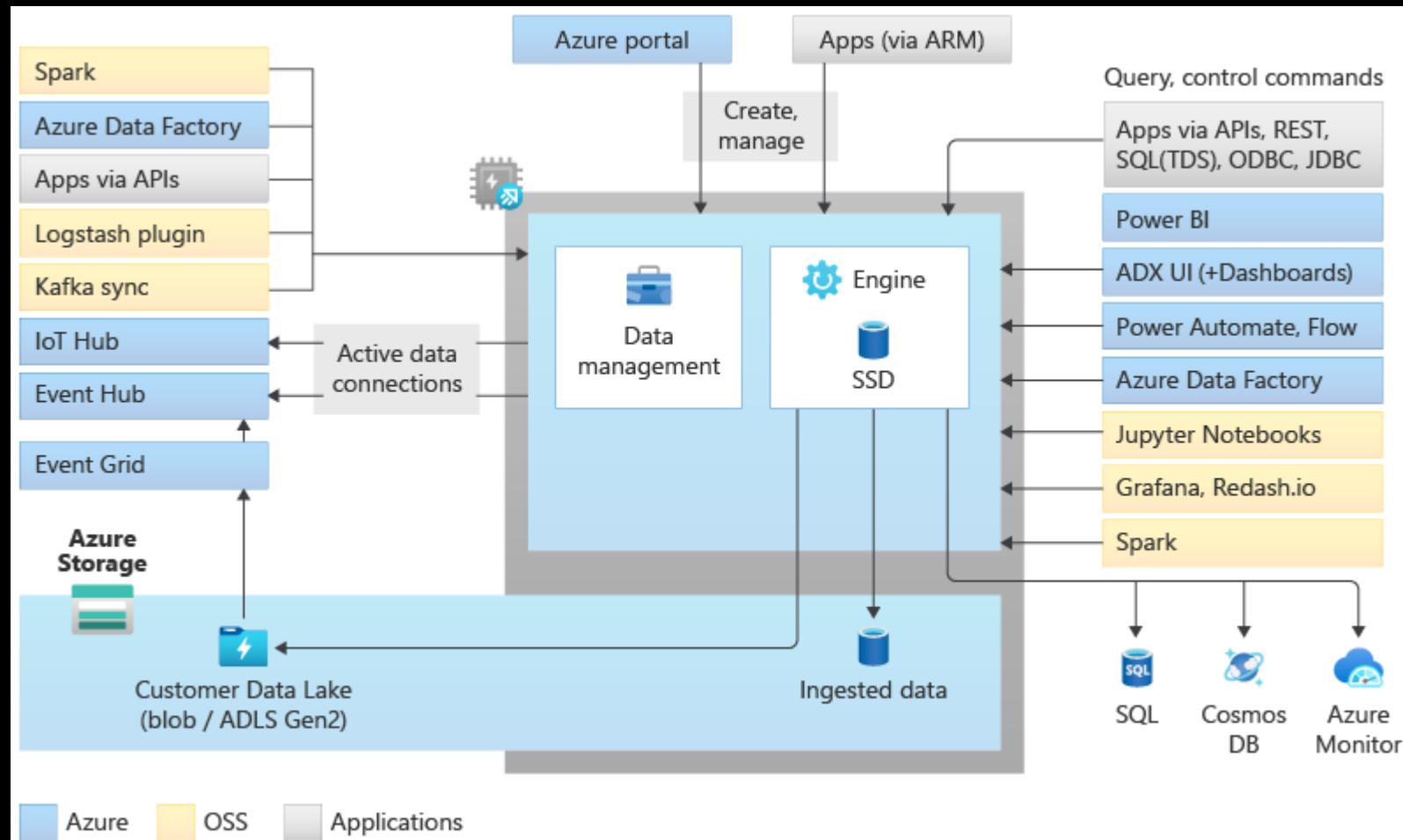
## Benefits

- Proven analytics on a petabyte scale
- Easy ingestion
- No index maintenance
- KQL (Kusto Query Language)
- Integrated in Synapse Studio



# Synapse Data Explorer pool architecture

- Storage and compute are separate and scale independently
- Separate compute to run background system jobs, managed, and queued data ingestion
- Persisted in blob storage accounts using a compressed columnar format



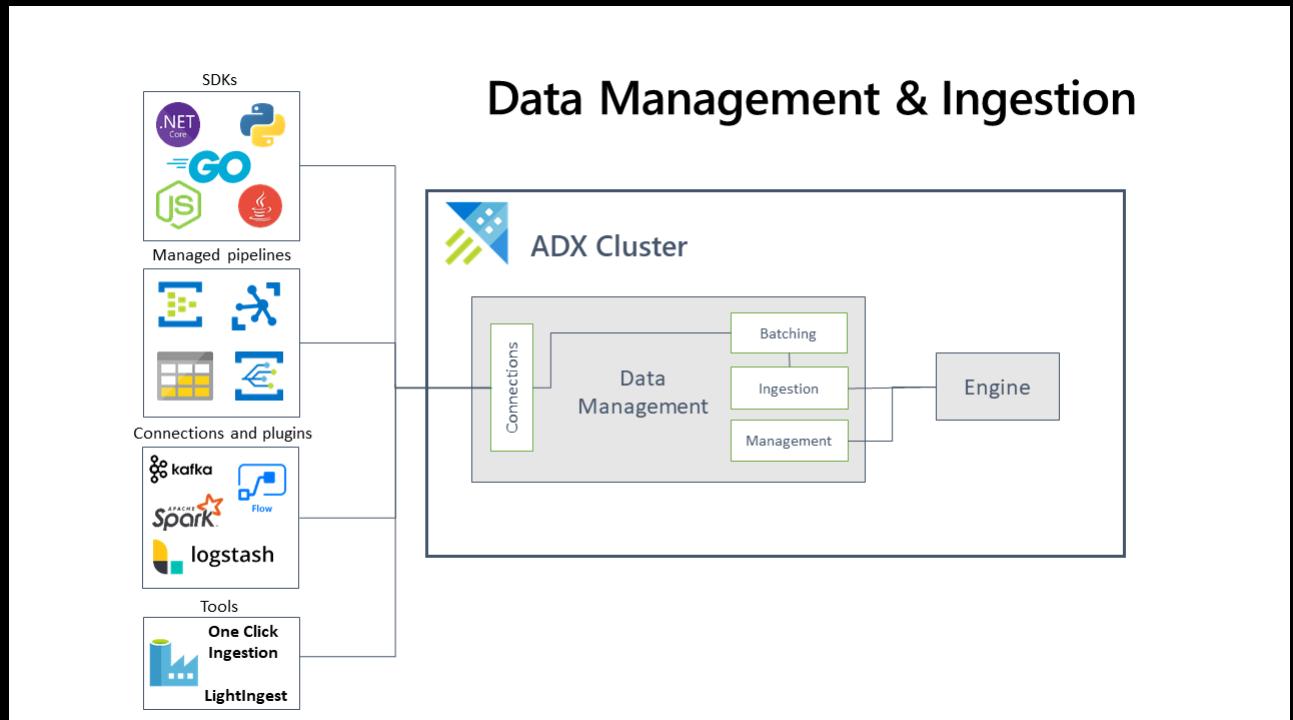
# Data ingestion overview

## Overview

The data management service pulls data from external sources, this can be via batch or streaming data. The DMS automatically indexes, organizes, encodes, and compresses data.

## Benefits

- Data validation and conversion
- Automatic indexing, organization, compression
- Ability to set a retention policy
- Once the DMS commits data into the engine, it's available for query



# Batch vs Streaming

---

## Batching

- Optimized for high ingestion throughput
- Most common type of ingestion
- Small batches are merged and optimized
- Configurable batching policy

## Streaming

- Near real-time latency
- Initially ingested to row store then moved to column store extents
- Initiate streaming ingestion via ADX client library or one of the supported data pipelines

# Data ingestion methods

## Overview

Azure Data Explorer supports several ingestion methods, each with its own target scenarios. These methods include ingestion tools, connectors and plugins to diverse services, managed pipelines, programmatic ingestion using SDKs, and direct access to ingestion.

## Ingestion methods

- Managed pipelines
- Connectors and plugins
- SDKs
- Tools



# Steps for ingestion

---

1. Set batching policy (optional)
2. Set retention policy
3. Create a table
4. Create schema mapping
5. Set update policy (optional)
6. Ingest data



# Ingest Best practices for Files and Tables

# Question... ?

How many different methods of loading ADLS can you think of?

What about a Synapse SQL Pool?



# Ingest flat files to tables

---

Ingest flat file data into Azure Storage (Azure Data Lake Store Gen2)

- When your data sources are on-premises, you need to move the data to Azure Storage before ingestion.
- Data in other cloud platforms needs to be moved to Azure Storage before ingestion.

Load from flat files as relational tables within the data warehouse

# Structuring Data in ADLS Gen 2

- Separate storage accounts for each environment: dev, test, & production.
- Use a common folder structure to organize data by degree of refinement.

ADLS Gen 2 Filesystem

Raw Data  
/bronze

Query Ready  
/silver

Report Ready  
/gold

# Ingest from on-premises data sources

---

## **Fastest is done by batch:**

- Extract from data source to multiple CSV/Parquet files
- Use AzCopy to upload to ADLS

## **Alternative is query-insert:**

- Set up SSIS self-hosted integration runtime on-premises
- Use Synapse Pipeline to extract/copy
- Use Synapse Pipeline to execute load procedure

## **Large Migrations:**

- Use Azure Data Box where available

# Ingest from cloud data sources

---

## Options:

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

# Ingest file data sources

---

## Look out for these file format challenges...

Invalid file format

- Multiple row types
- Ragged columns

Row size > 1Mb

Datetime format/s (e.g., use of nanosecond date time)

NULL value literal/s

Free form text

Parquet partitions

XML data

Use of non-standard line delimiters (e.g., CR)

## ...and try these Solutions

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

# Ingest and store - formats

---

For batch flat files, Azure Synapse Analytics supports Avro, Binary, Delimited Text, Excel, Parquet, ORC, JSON, and XML formats.

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

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

# Ingest – When to BCP / Bulk Copy

---

## **Green fields: Never**

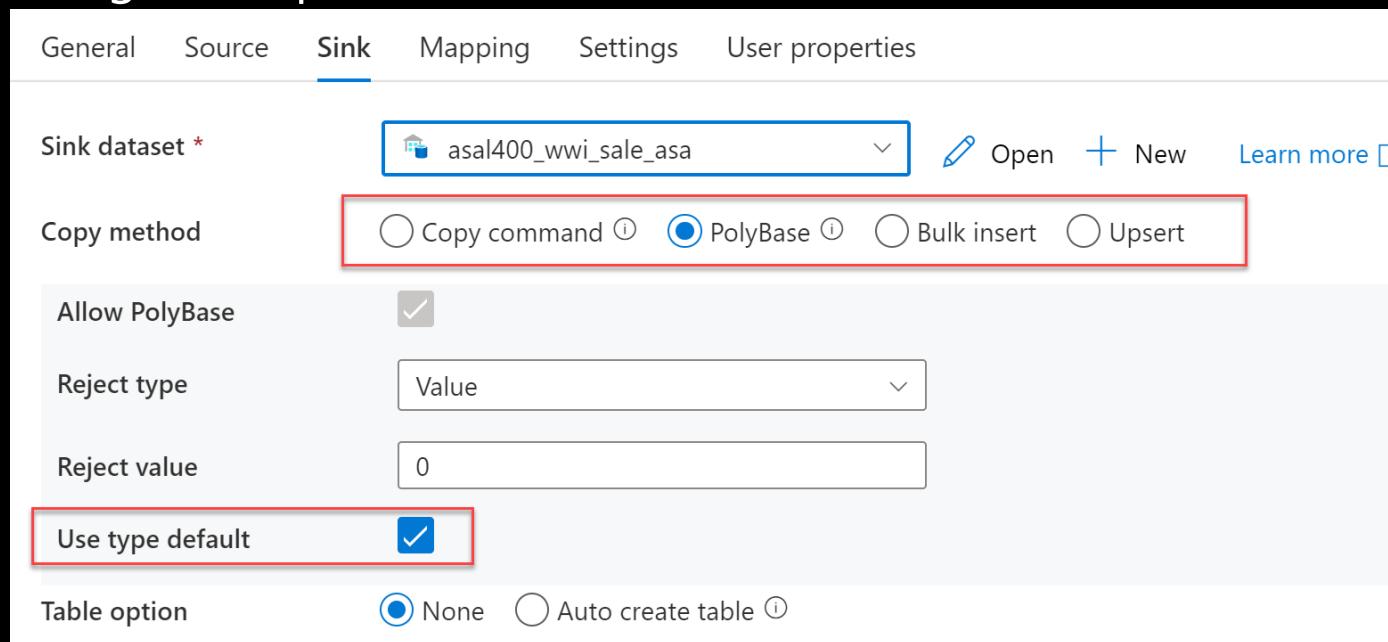
- Network unreliability, no retries
- Needs VM in cloud, performance dependent on VM configuration
- Doesn't support ADLS
- Reduces concurrency
- Control-gated performance limitation, can not scale with DWU

## **Migrations:**

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

# Ingest – Synapse pipelines

- Un-check USE TYPE DEFAULT, it is not a best practice.
- Land data in ADLS Gen2, then ingest using Polybase / COPY.
  - This means you can re-ingest the same data set without having to repeat extracts, and better demonstrate ingestion performance.



# Ingest and store – Loading staging tables

---

## Indexing

### Use Heap tables

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

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

# Ingest and store – Loading staging tables

---

## Distribution

Use Round Robin Distribution for:

- Potentially useful tables created from raw input.
- Temporary staging tables used in data preparation.

Other distribution considerations:

- Never load to a REPLICATED table
- Load to a ROUND\_ROBIN table if the test is ONLY raw ingestion performance, or if the table is very small
- Load to a HASH table if the task is a pipeline with subsequent transformations using the loaded table

# Ingest – Scaling to shorten duration

---

Ingestion duration is correlated with the number of DWU's allocated to the SQL Pool.

For every *doubling* of the DWU's you *halve* the ingestion time.

$$2d = t/2$$

d: DWU

T: ingestion time

Only applies from DWU500c – DWU30000c

# Export to files with CETAS

---

CETAS = parallel operation that creates external table metadata and exports the SELECT query results to a set of files in your storage account.

Store frequently used parts of queries, like joined reference tables, to a new set of files. You can then join to this single external table instead of repeating common joins in multiple queries.

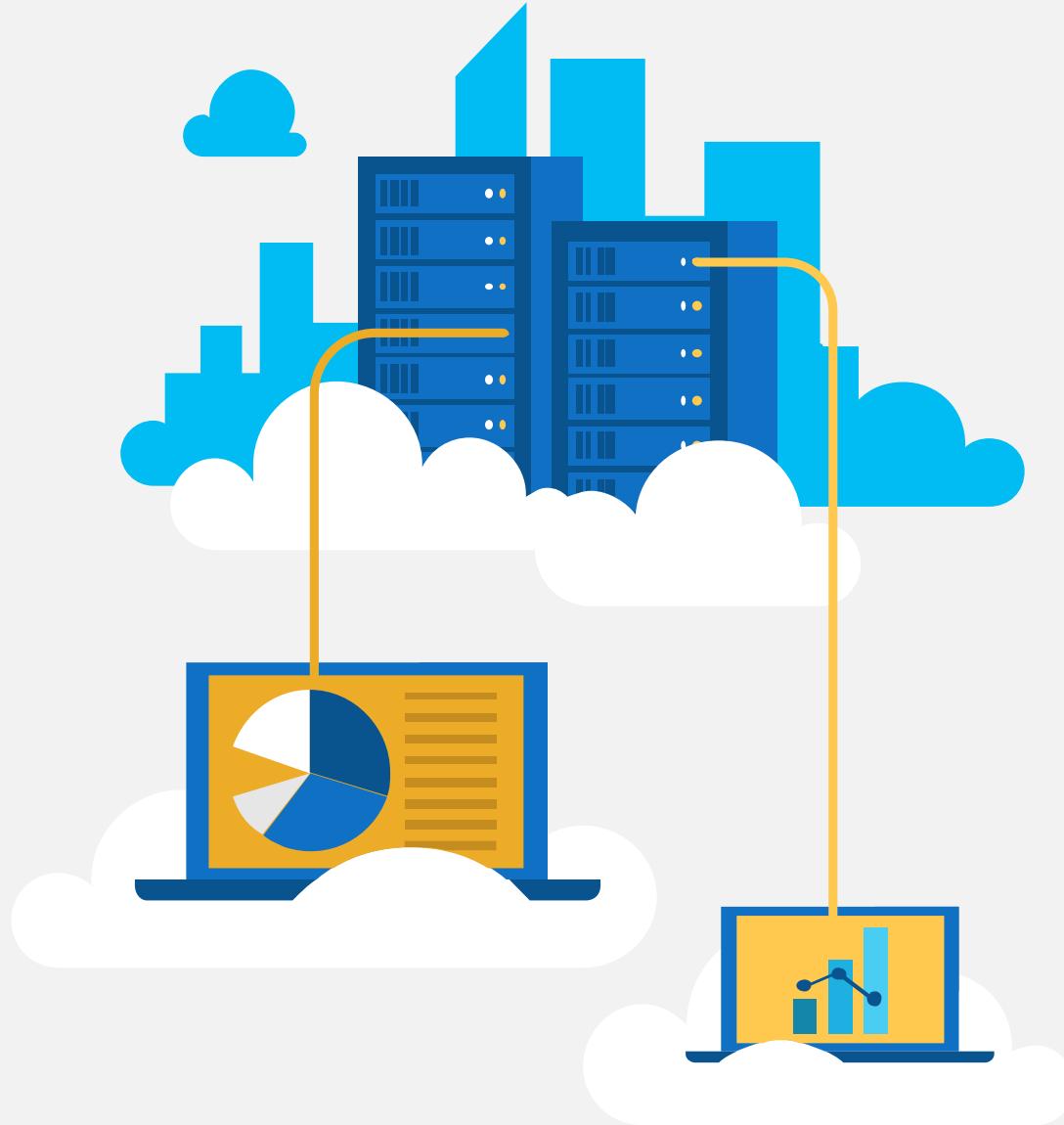
As CETAS generates Parquet files, statistics will be automatically created when the first query targets this external table, resulting in improved performance.

## Pop Quiz 2

True or False: Both COPY command AND Polybase require CONTROL permission

TRUE

FALSE

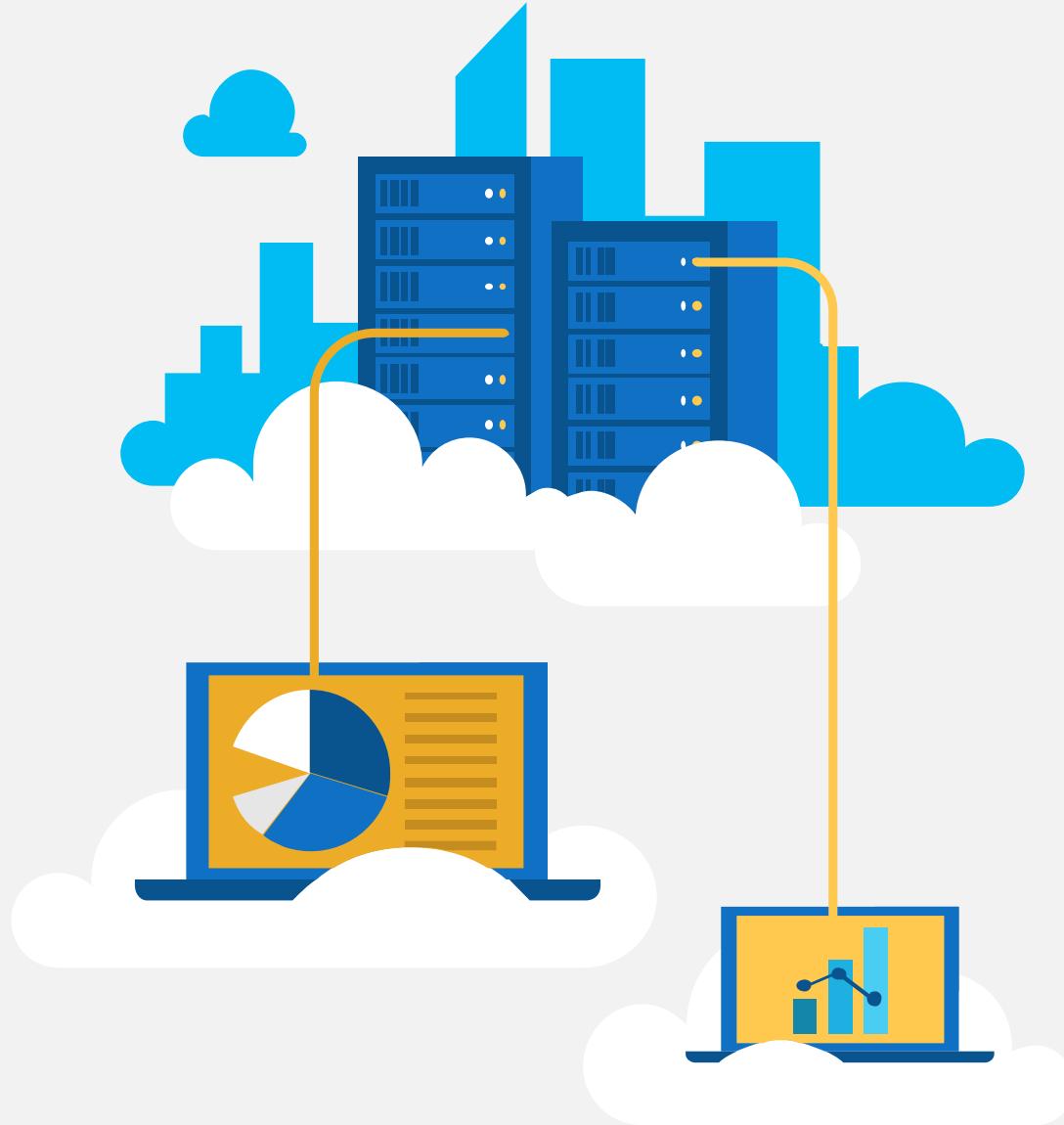


## Pop Quiz 2

True or False: Both COPY command AND Polybase require CONTROL permissions

TRUE

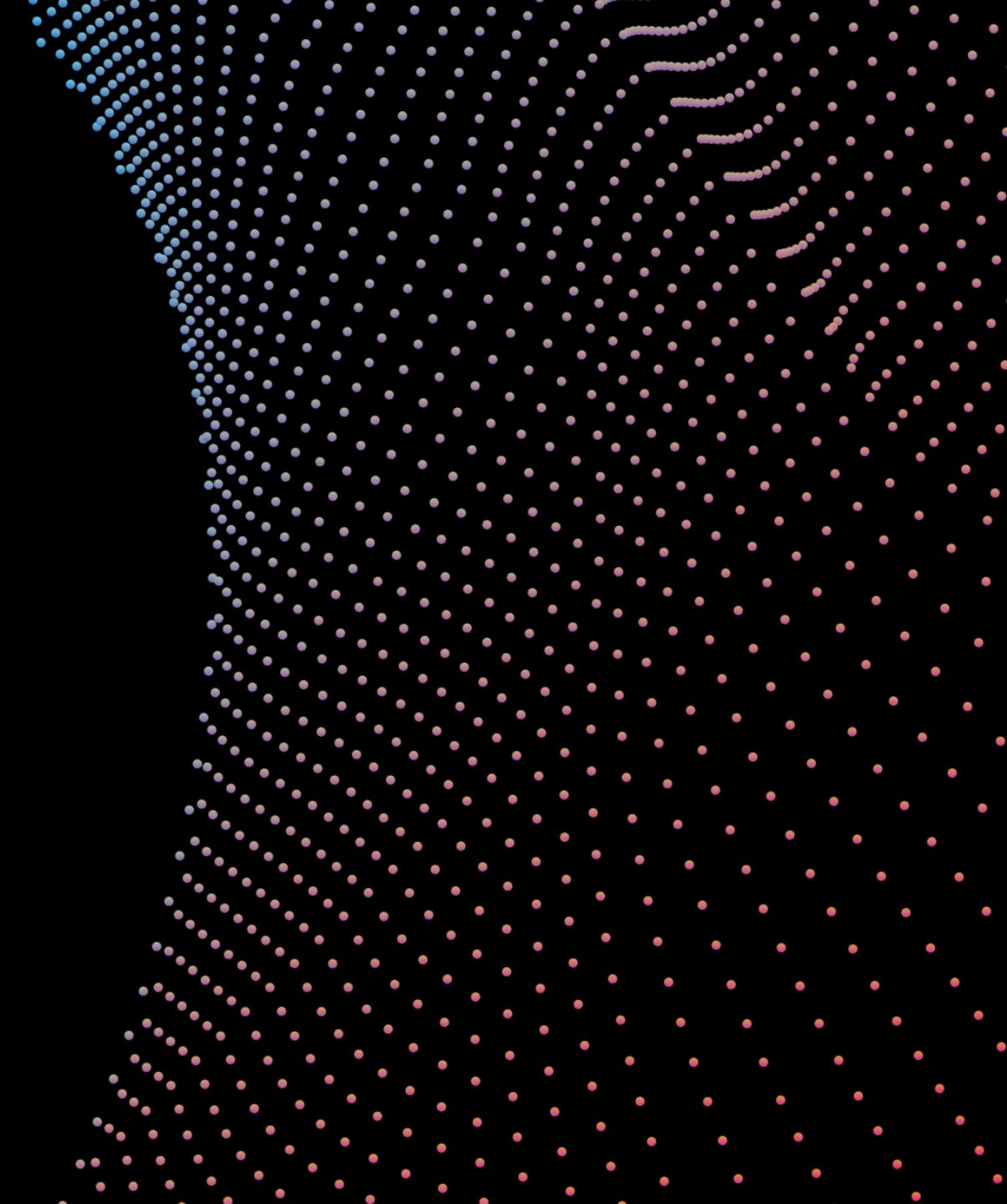
**FALSE**





# Thank you

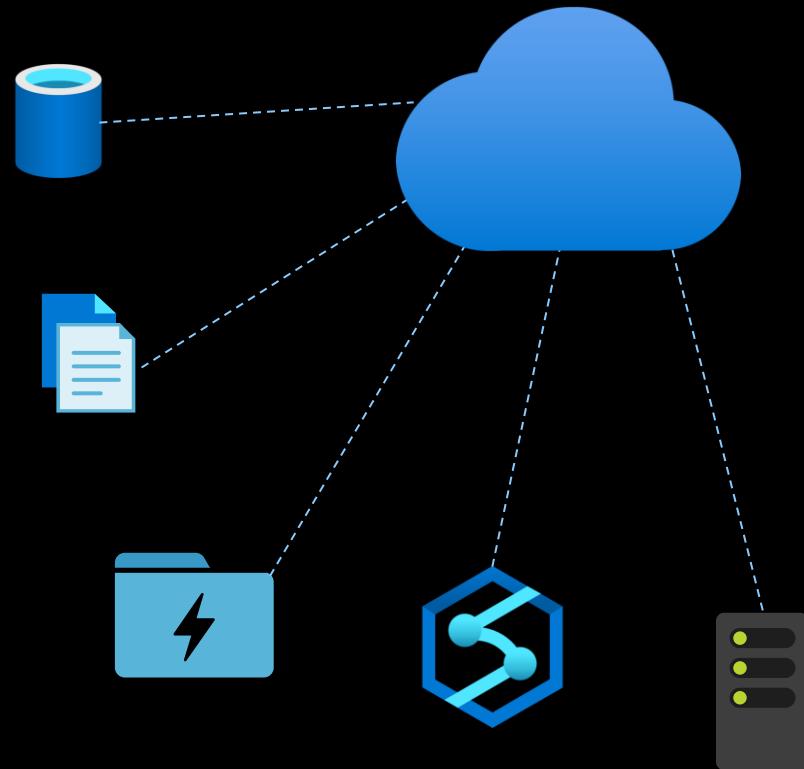
# Data Transformations



# Agenda

---

- 1) Preparing to transform
- 2) Apply transformations
- 3) Serverless transforms
- 4) Transform with Spark
- 5) Best practices



# Typical data transformations

---

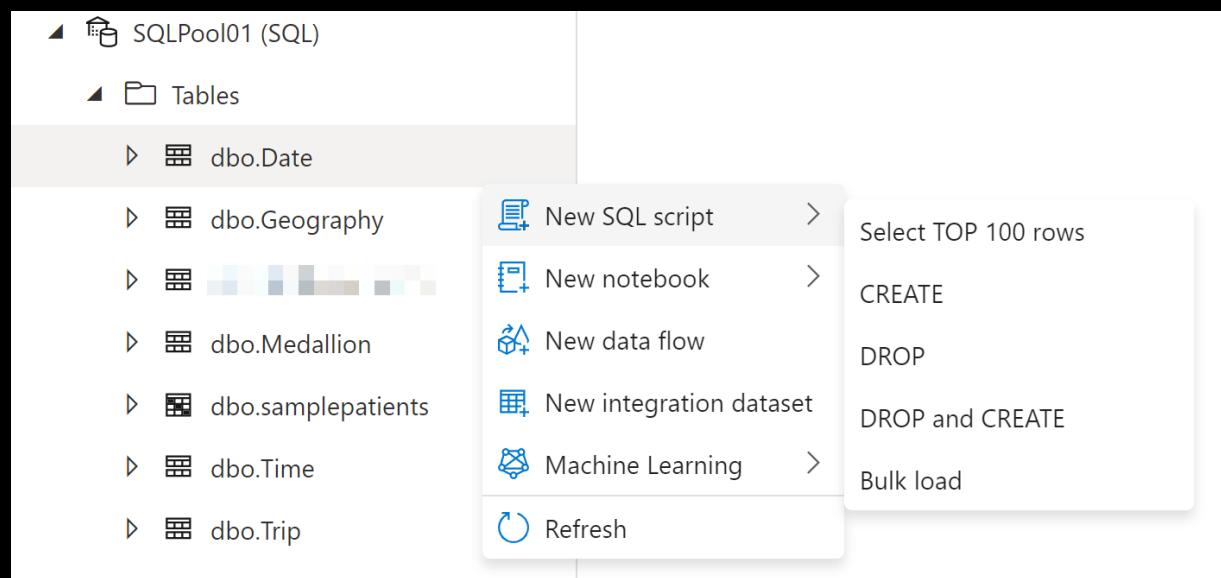
- Create persistent staging area / data vault
- Standardize data from different sources
- Remove duplicate rows
- Impute missing values
- Calculate derived values
- Prepare data for facts and dimensions

# Transform

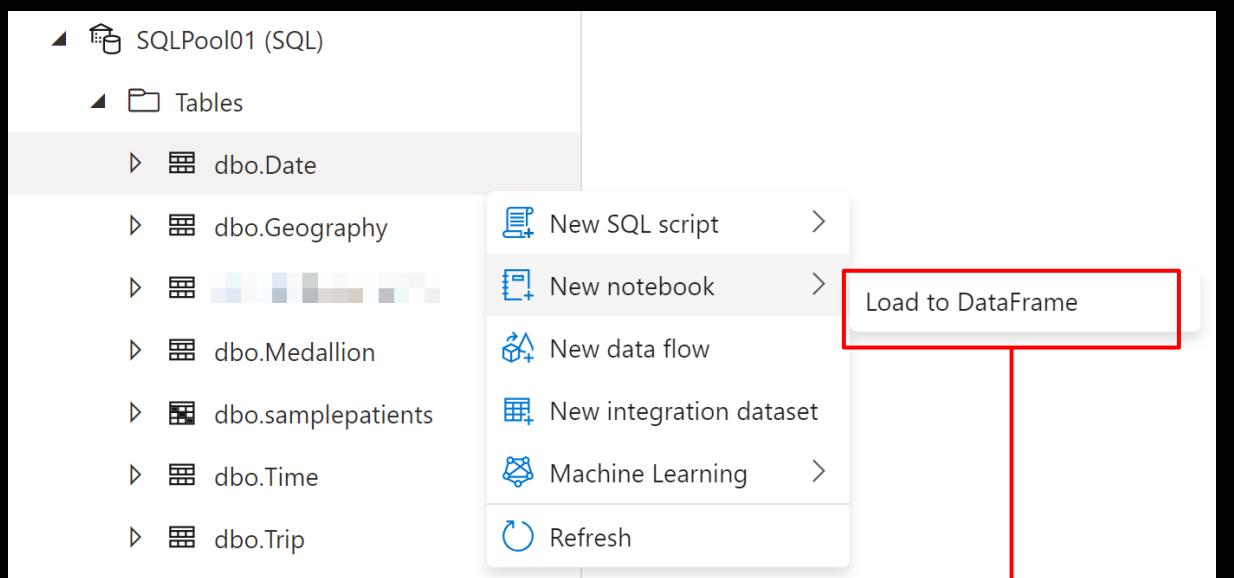
## Applying transformations

# Code based transformations

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



Starting from a table, auto-generate a single line of PySpark code that makes it easy to load a SQL table into a Spark dataframe and author transforms in a notebook.



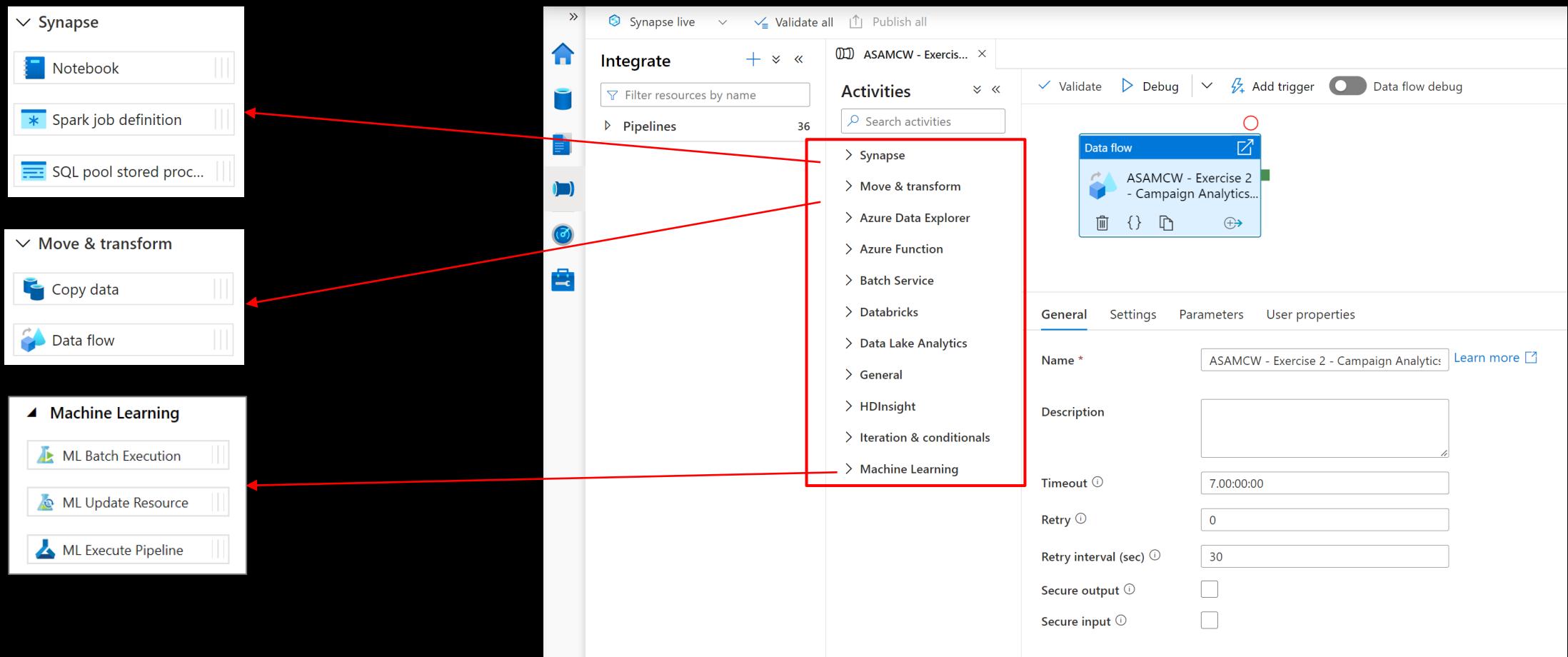
A screenshot of a Jupyter Notebook titled 'Notebook 3'. The top bar shows options like 'Run all', 'Undo', 'Publish', 'Outline', 'Attach to', 'Language', and 'Spark (Scala)'. The notebook content shows a cell with the following PySpark code:

```
1 %%spark
2 val df = spark.read.sqlAnalytics("SQLPool01.dbo.Date")
3 df.write.mode("overwrite").saveAsTable("default.t1")
```

The cell status is '[ ]' and the note 'Press shift + enter to execute cells' is displayed. Below the cell are buttons for '+ Code' and '+ Markdown'.

# Transform with pipelines

Orchestrate transformations with Synapse Pipelines.



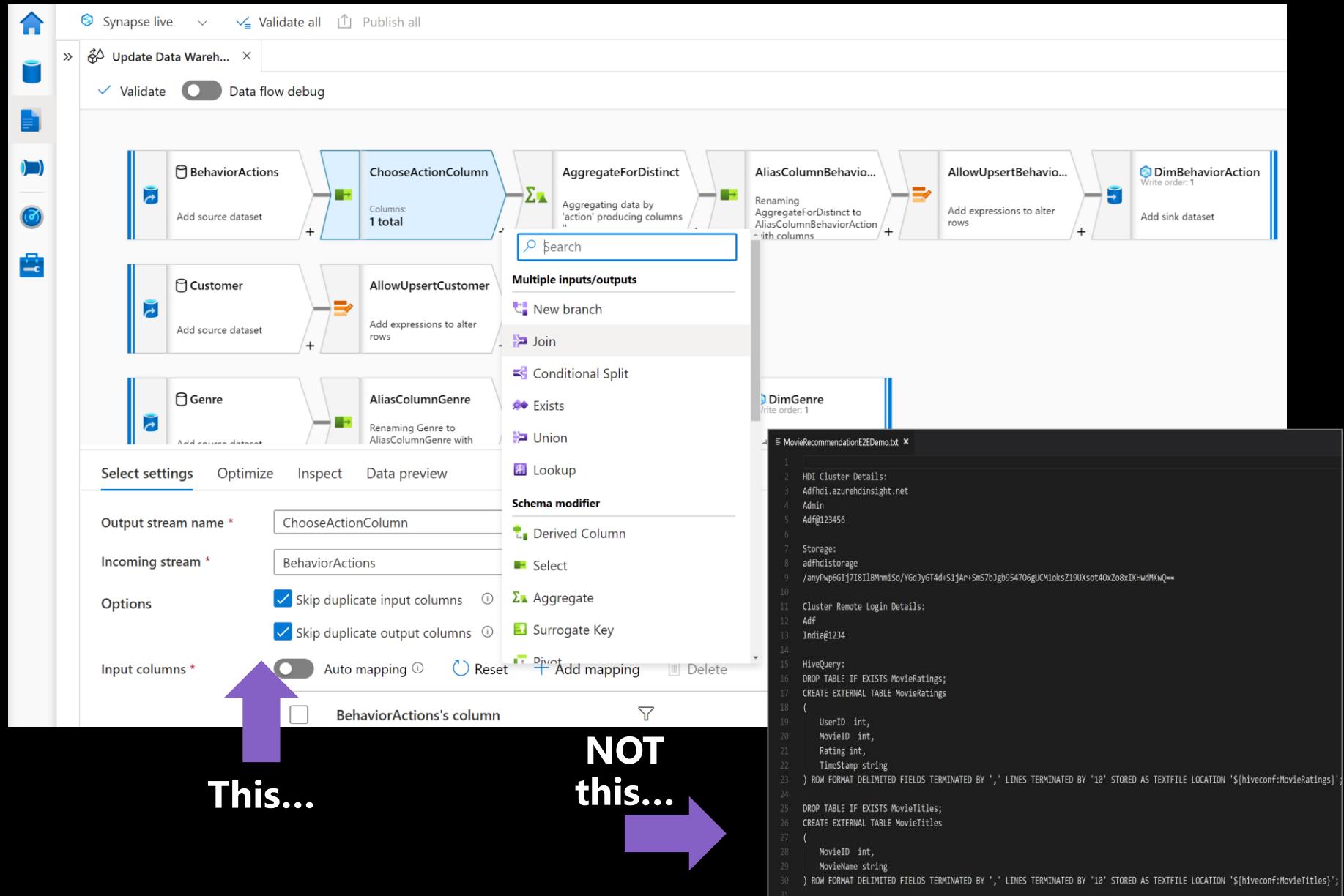
# No code transform with data flows

## 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





# Transform Transform with serverless SQL

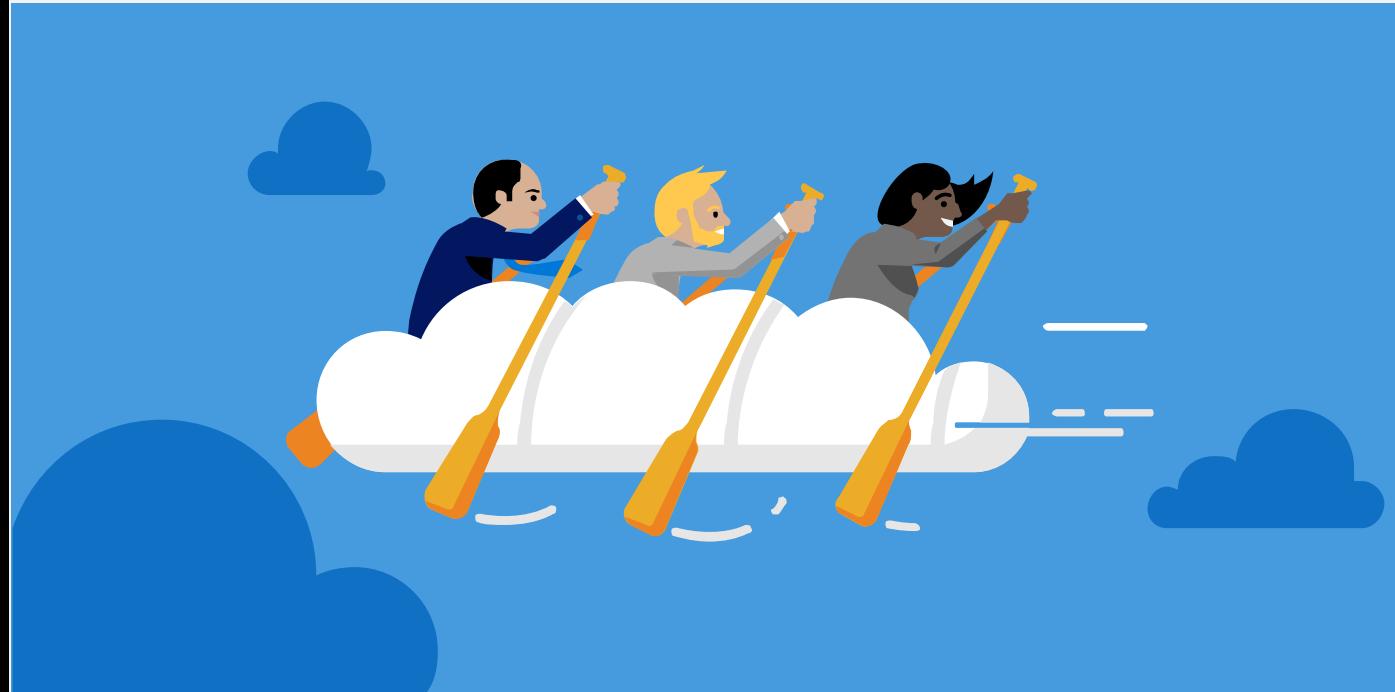
# Pop Quiz 1

What's the largest scale TPC-H workload serverless SQL has successfully run?

A)  
100TB

B)  
1PB

C)  
10PB



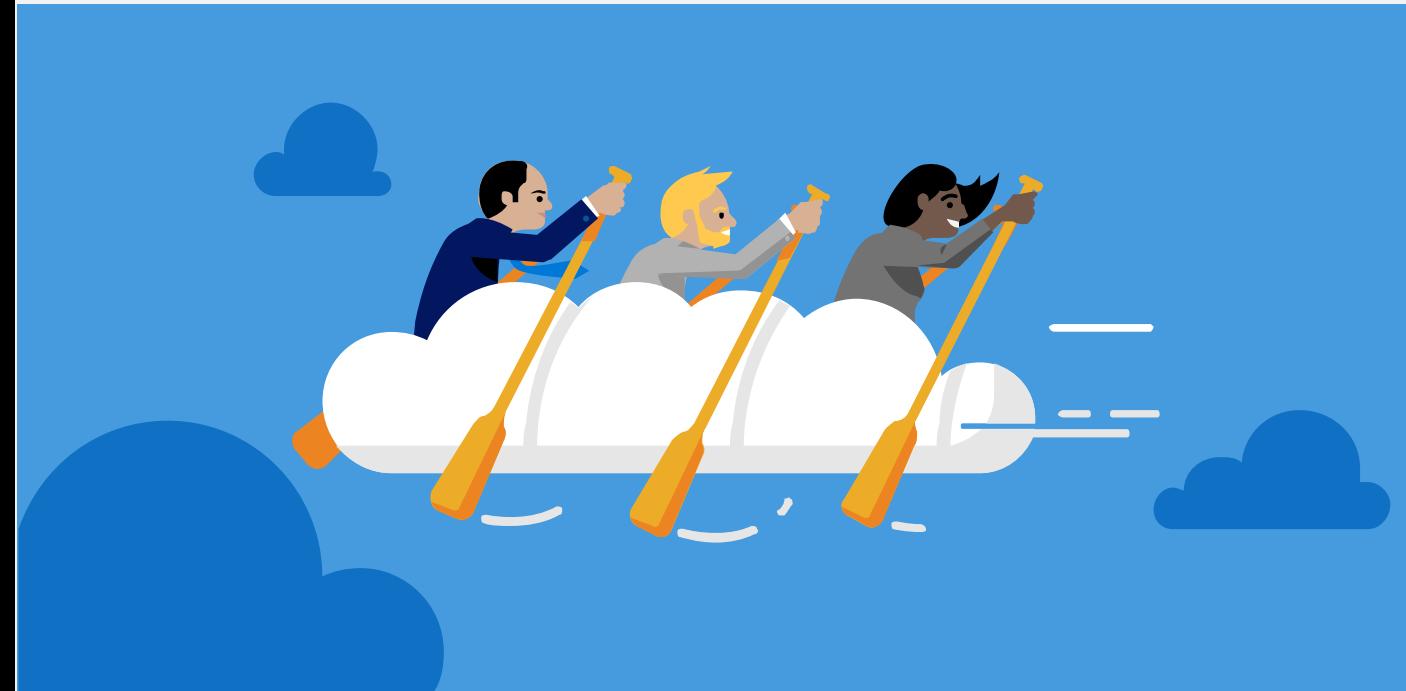
# Pop Quiz 1

What's the largest scale TPC-H workload a serverless SQL pool has successfully run?

A)  
100TB

**B)  
1PB**

C)  
10PB



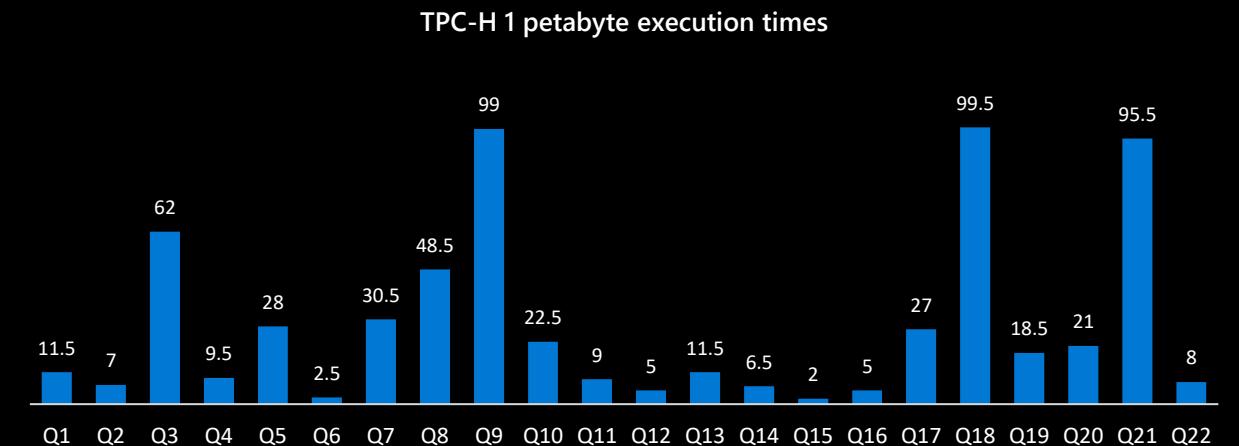
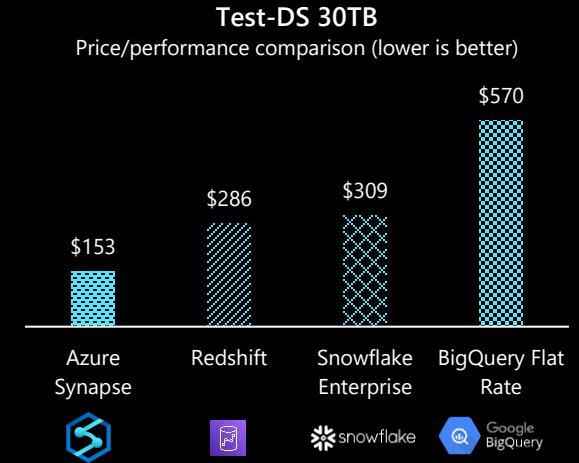
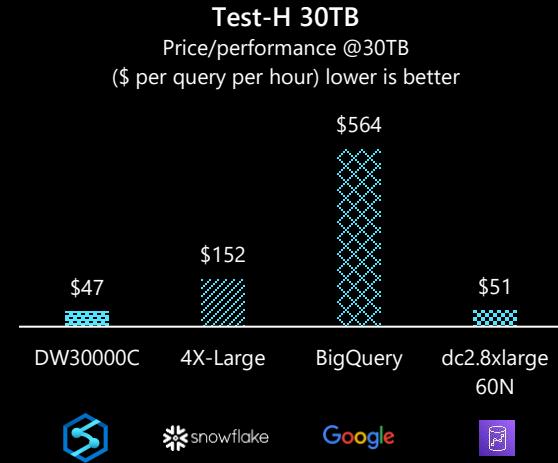
# Industry leading SQL performance and scalability

## TPC-H and TPC-DS Leader

Price/performance leadership relative to other cloud data warehouses

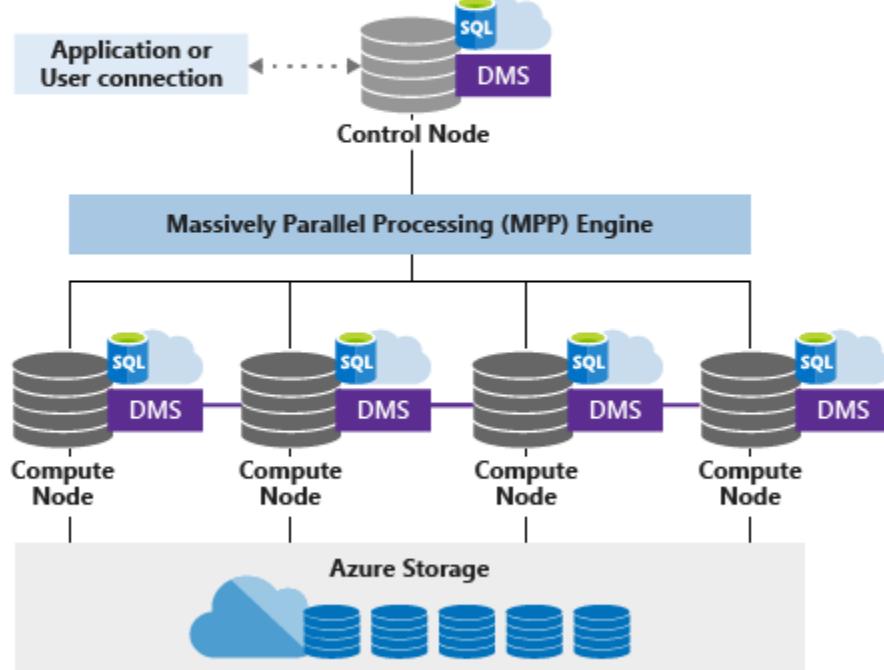
"Polaris" is the only query engine to successfully complete TPC-H at 1PB scale

<https://aka.ms/synapse-dqp>

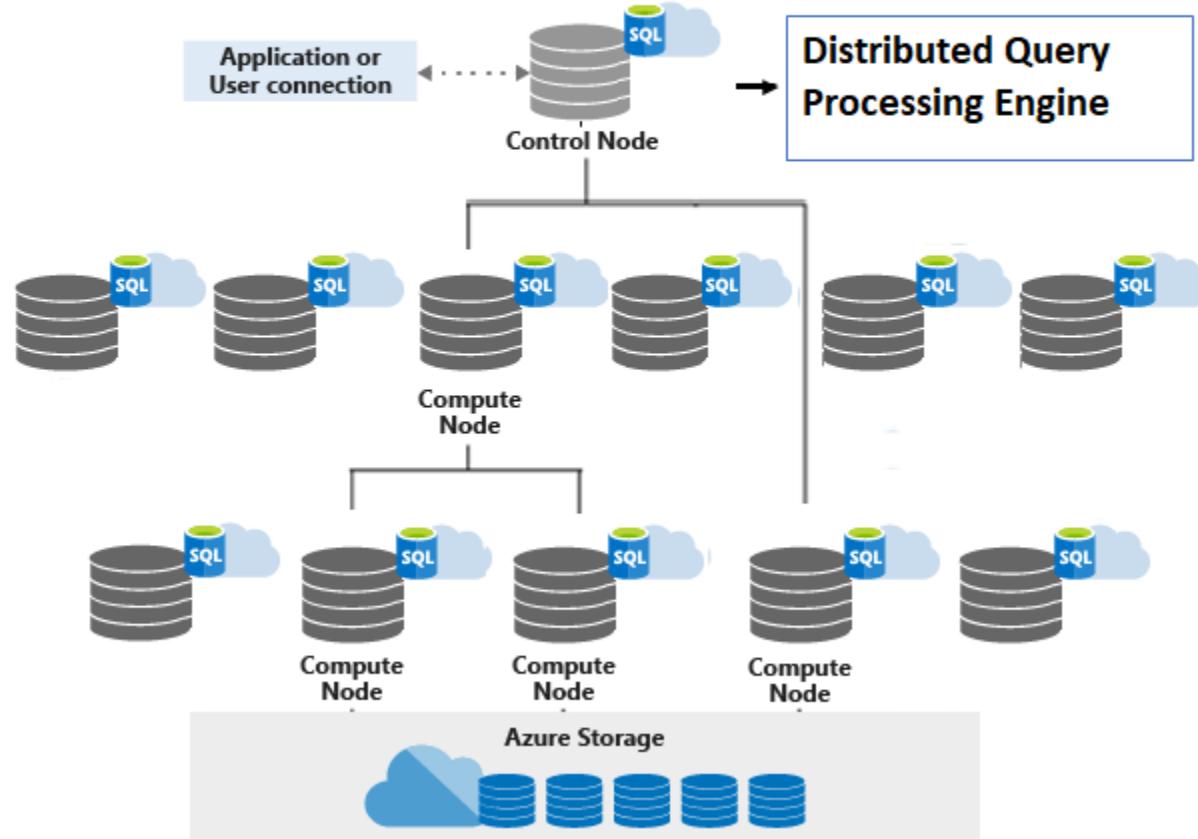


# SQL pool infrastructure

Dedicated SQL pool



Serverless SQL pool



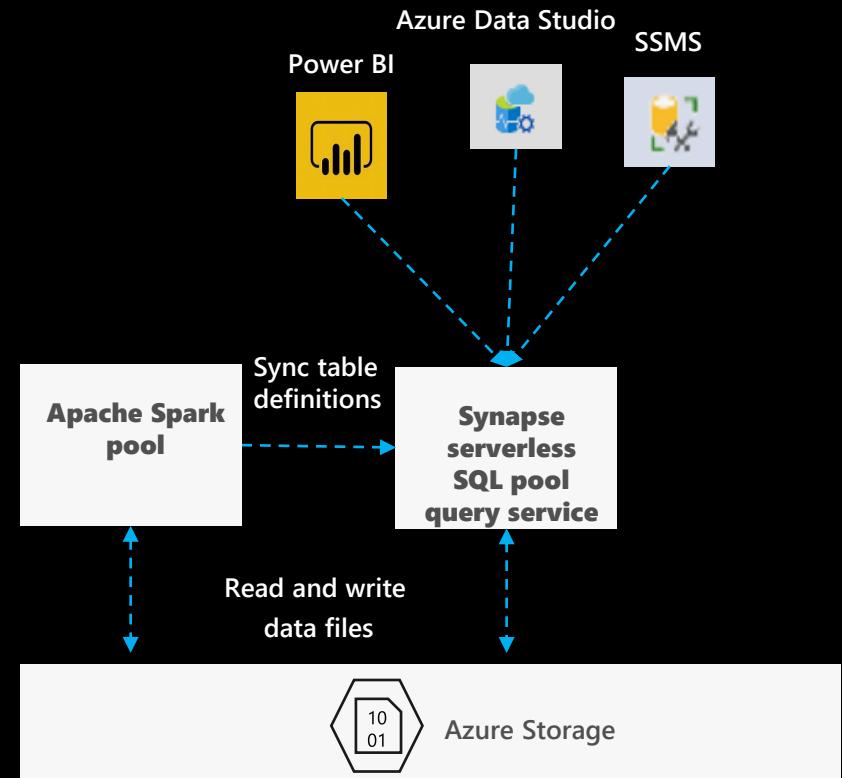
# Serverless SQL pool

## 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

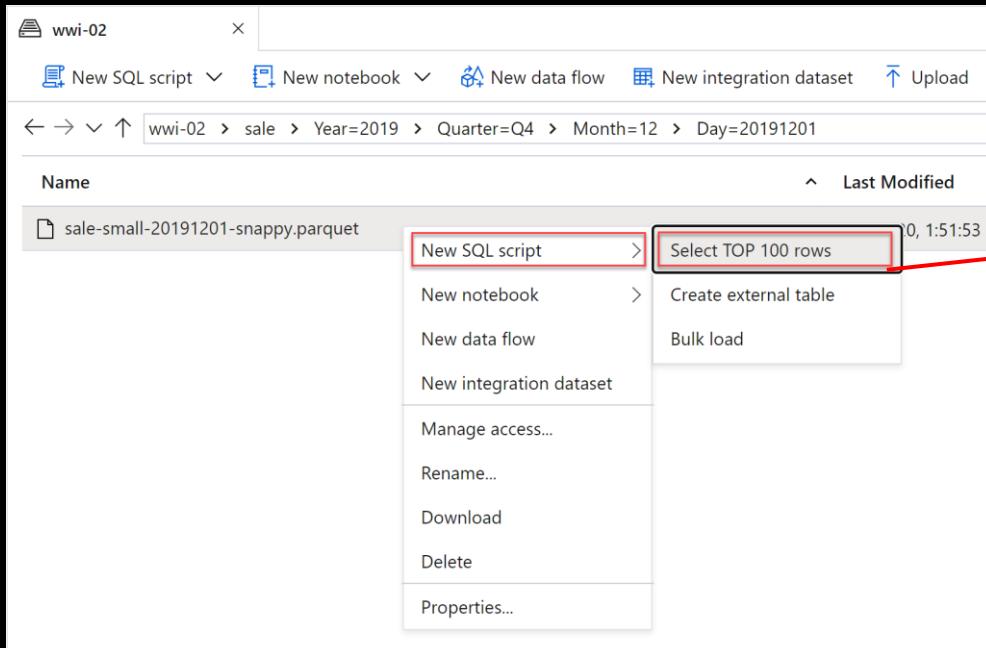
## Logical Data Warehouse

- 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



A screenshot of the Azure Data Explorer results pane. At the top, the connection dropdown is set to 'Built-in' (highlighted by a red box). The query plan shows an auto-generated T-SQL script for reading a parquet file from a blob storage location. The results pane displays a table of transaction data with columns: TransactionId, CustomerId, ProductId, Quantity, Price, TotalAmount, TransactionDate, ProfitAmount, and Hour. The first few rows of data are:

TransactionId	CustomerId	ProductId	Quantity	Price	TotalAmount	TransactionDate	ProfitAmount	Hour
9453b35d-3f0d...	5	1588	1	20.1900000000...	20.1900000000...	20191201	6.3300000000...	12
9453b35d-3f0d...	5	4882	2	29.3700000000...	58.7400000000...	20191201	19.3600000000...	12
9453b35d-3f0d...	5	3283	2	28.3300000000...	56.6600000000...	20191201	16.0800000000...	12
9453b35d-3f0d...	5	4777	2	28.3500000000...	56.7000000000...	20191201	17.3000000000...	12
9453b35d-3f0d...	5	4908	2	35.4000000000...	70.8000000000...	20191201	21.4200000000...	12

At the bottom of the results pane, a message indicates: '00:00:15 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/parquet/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.

## 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.

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

	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

# Inline schema definition

## 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

# Customized content parsing

## 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-16 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

# Using folder structure to query partitioned data

## 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

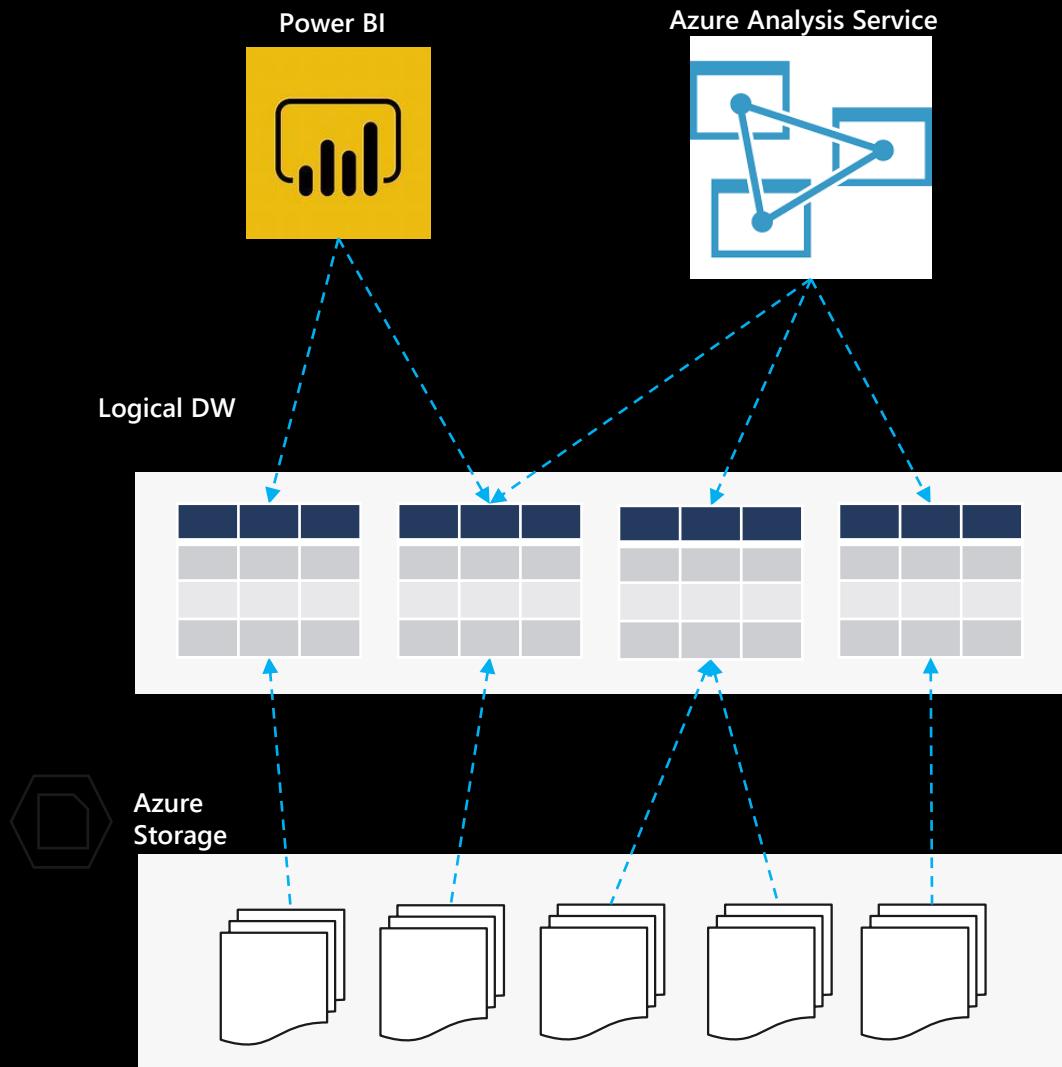
# Leverage serverless SQL pool as a logical data warehouse

## 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

# 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

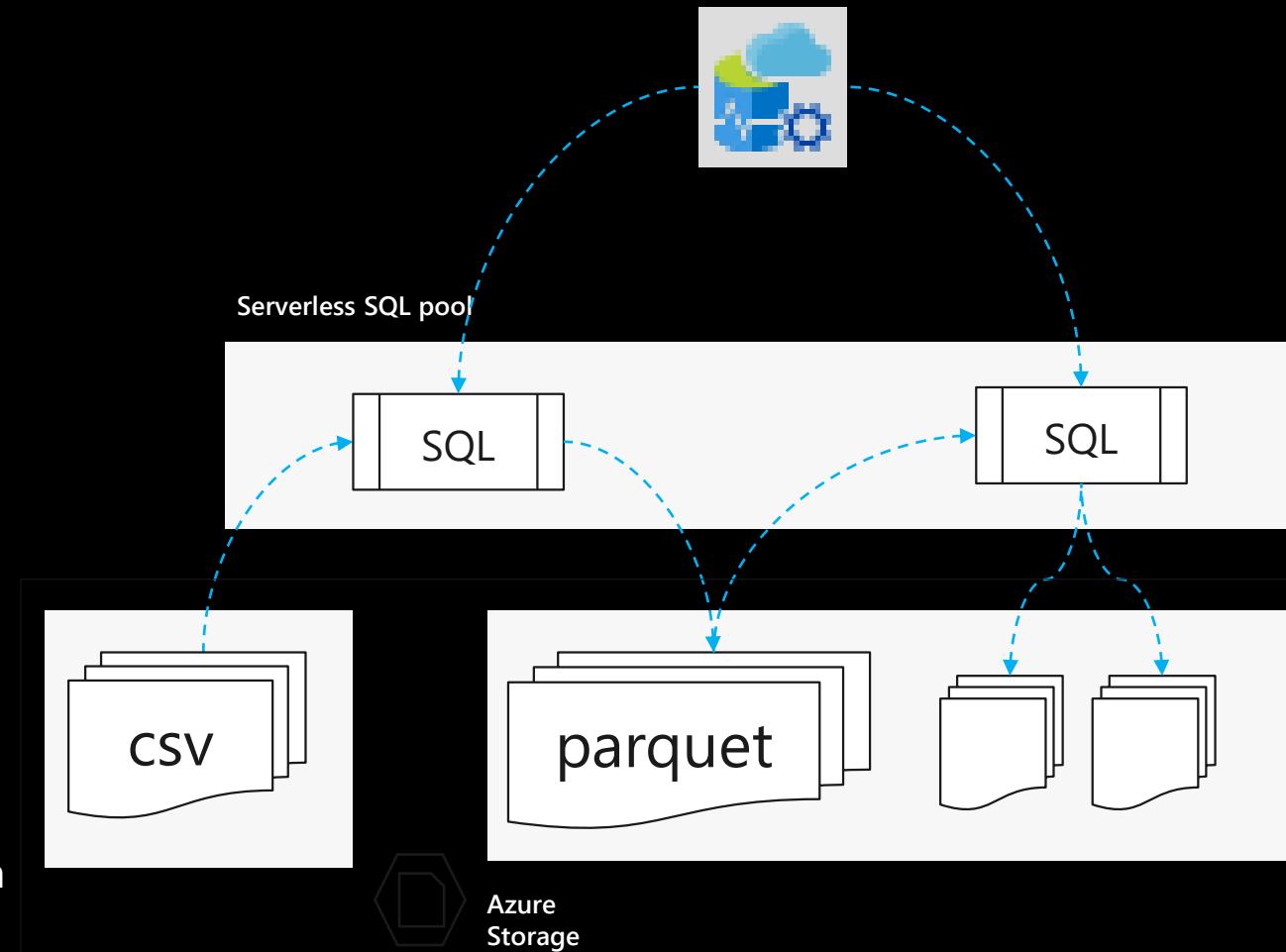
## Benefits

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 with CETAS

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  
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

Create external table \*  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
5
6  IF NOT EXISTS (SELECT * FROM sys.external_file_formats WHERE name = 'SynapseParquetFormat')
7    CREATE EXTERNAL FILE FORMAT [SynapseParquetFormat]
8      WITH ( FORMAT_TYPE = PARQUET)
9  GO
10
11 IF NOT EXISTS (SELECT * FROM sys.external_data_sources WHERE name = 'default_azureSynapseA_dfs_core_windows_net')
12   CREATE EXTERNAL DATA SOURCE [default_azureSynapseA_dfs_core_windows_net]
13   WITH (
14     LOCATION   = 'https://azuresynapsesa.dfs.core.windows.net/default',
15   )
16   Go
17
18 CREATE EXTERNAL TABLE adls.retailsale (
19   [storeId] varchar(8000),
20   [productCode] varchar(8000),
21   [quantity] varchar(8000),
22   [logQuantity] varchar(8000),
23   [advertising] varchar(8000),
24   [price] varchar(8000),
25   [weekStarting] varchar(8000),
26   [id] varchar(8000)
27 )
28 WITH (
29   LOCATION = 'Parquet/part-00000-5ae12a71-d27d-4e3a-a686-3bfb7d67c2c9-c000.snappy.parquet',
30   DATA_SOURCE = [default_azureSynapseA_dfs_core_windows_net],
31   FILE_FORMAT = [SynapseParquetFormat]
32 )
33 Go
34
35 SELECT TOP 100 * FROM adls.retailsale
```

# Lake databases

## Overview

A lake database is a database that uses Azure Data Lake as its backend platform.

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

## Benefits

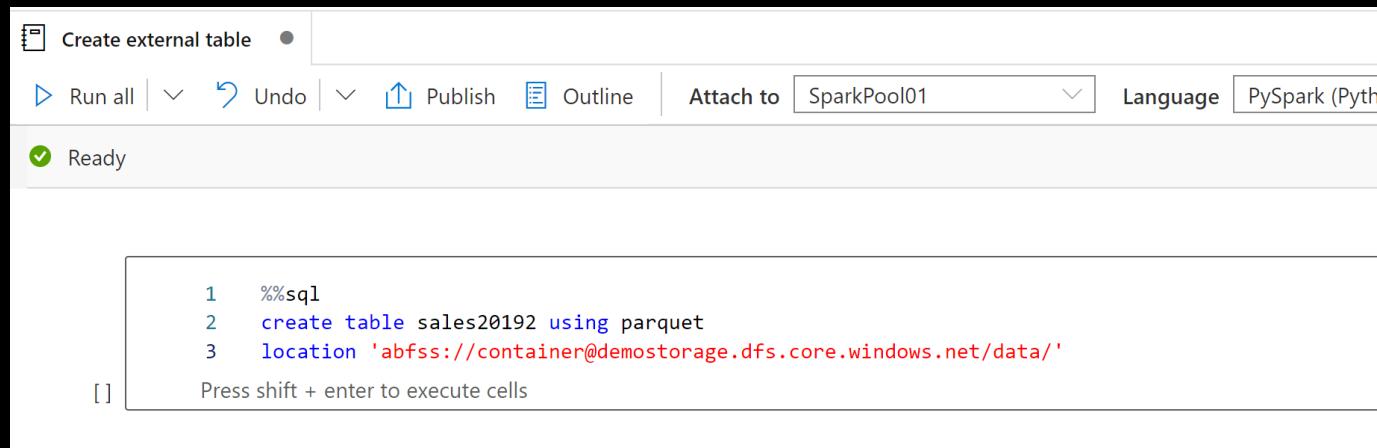
Tables designed using Spark languages are immediately available in serverless SQL pool.

Schema definition matches original

Spark table updates are applied in serverless SQL pool

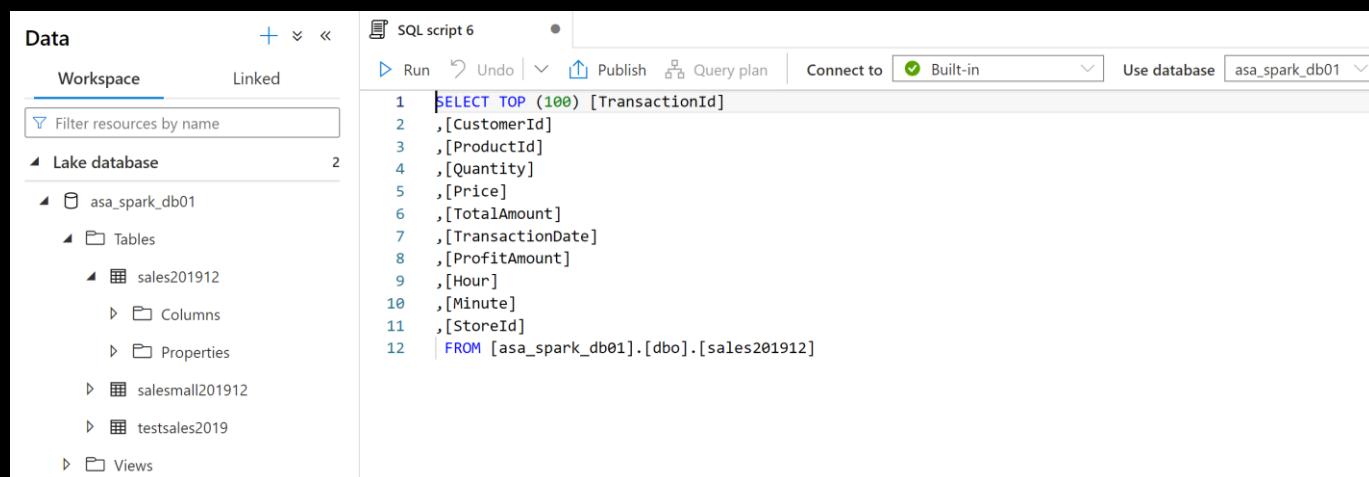
No need to manually create SQL tables that match Spark tables

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



```
1 %%sql
2 create table sales20192 using parquet
3 location 'abfss://container@demostorage.dfs.core.windows.net/data/'
```

Press shift + enter to execute cells



```
1 SELECT TOP (100) [TransactionId]
2 ,[CustomerId]
3 ,[ProductId]
4 ,[Quantity]
5 ,[Price]
6 ,[TotalAmount]
7 ,[TransactionDate]
8 ,[ProfitAmount]
9 ,[Hour]
10 ,[Minute]
11 ,[StoreId]
12 | FROM [asa_spark_db01].[dbo].[sales201912]
```

# Shared metadata tables

---

## Overview

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

## 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.

# Transform Transform with Spark

# Querying SQL pools

## 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=*.database.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 Using Scala

```
// Construct a Spark DataFrame from SQL Pool table  
var df = spark.read.synapsesql("sql1.dbo.Tbl1")  
  
// Write the Spark DataFrame into SQL Pool table  
df.write.sqlanalytics("sql1.dbo.Tbl2")
```

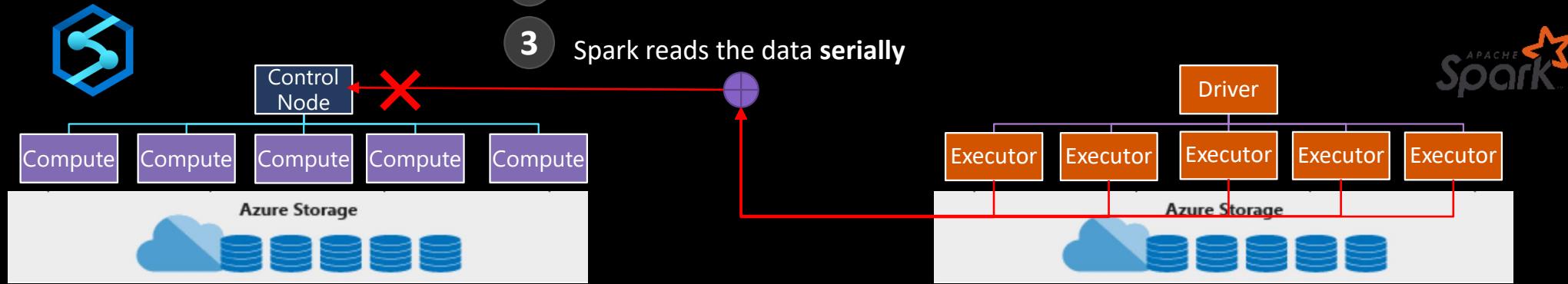
## Using Python

```
%spark  
var df = spark.read.synapsesql("sql1.dbo.Tbl1")  
df.createOrReplaceTempView("tbl1")
```

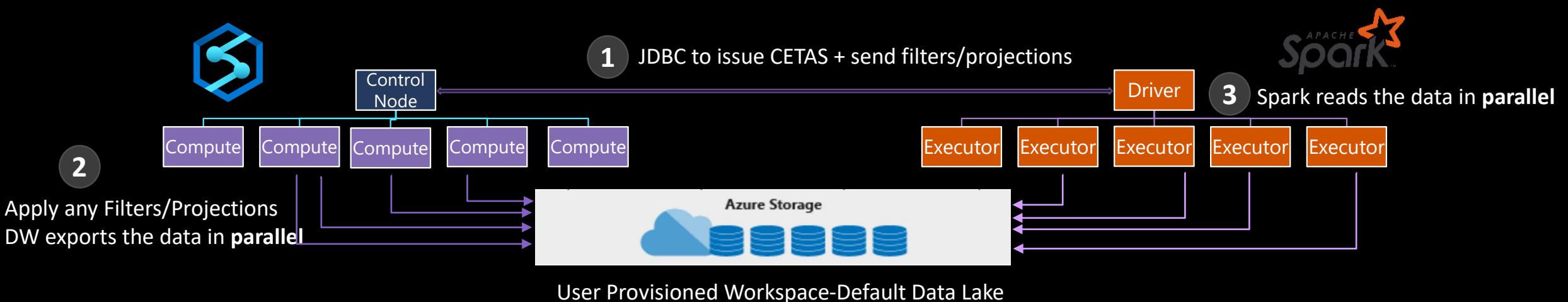
```
%pyspark  
sample = spark.sql("SELECT * FROM tbl1")  
sample.createOrReplaceTempView("tblnew")
```

```
%spark  
var df = spark.sql("SELECT * FROM tblnew")  
df.write.synapsesql("sql1.dbo.tb12",  
  Constants.INTERNAL)
```

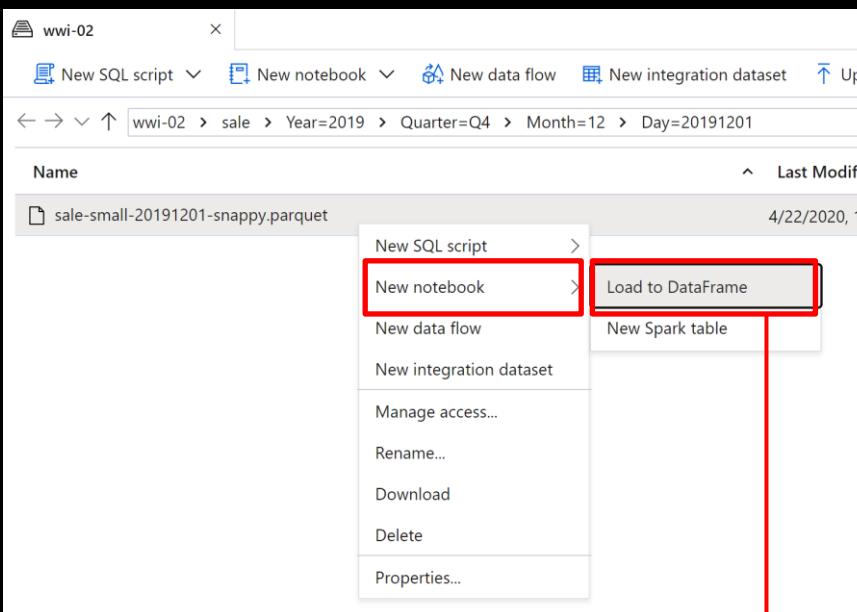
## Existing Approach: JDBC



## New Approach: JDBC and Polybase



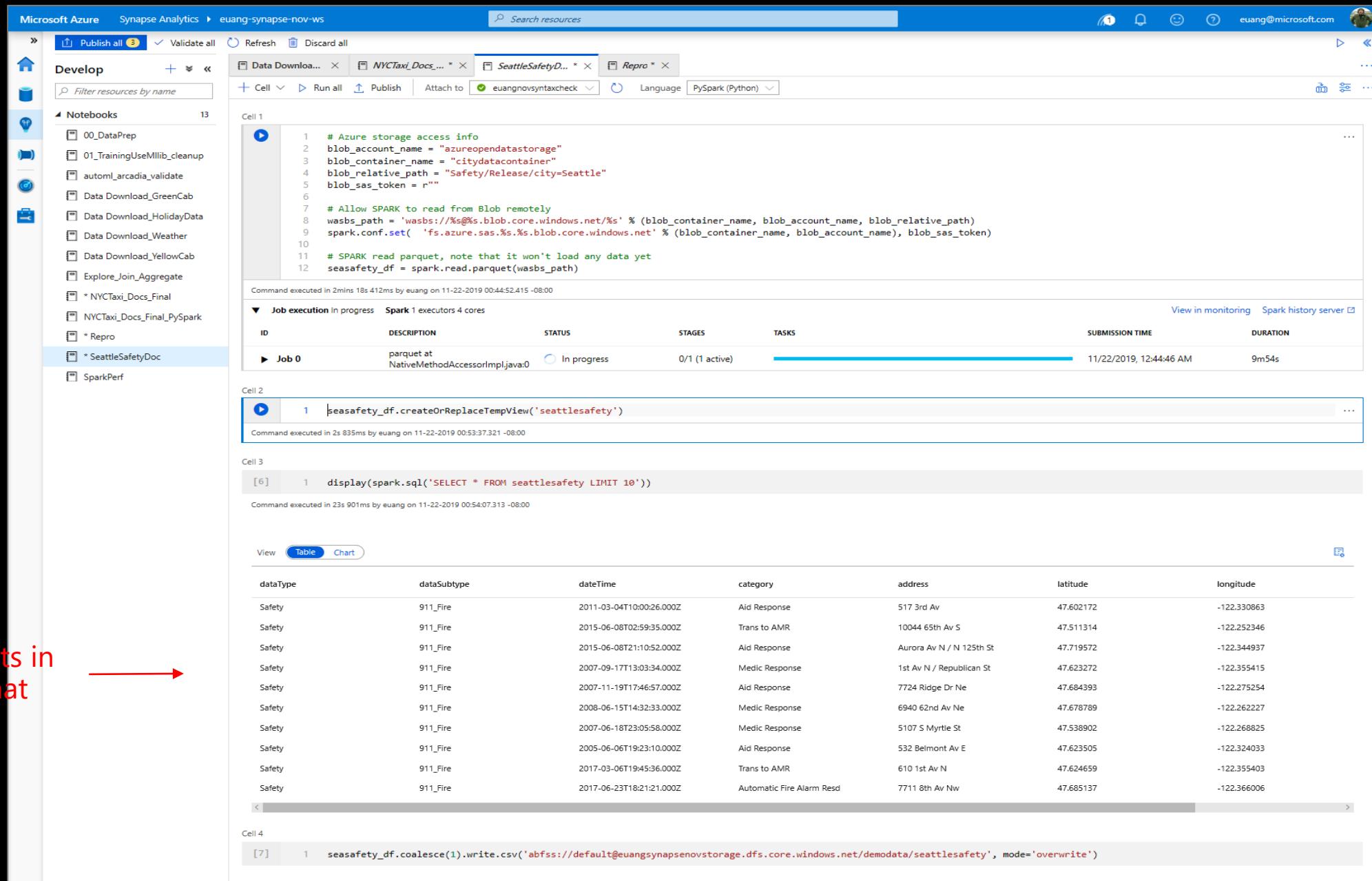
# Create a notebook from files in storage



A screenshot of a PySpark notebook titled 'Notebook 3'. The notebook interface includes a toolbar with 'Run all', 'Undo', 'Publish', 'Outline', 'Attach to' (set to 'SparkPool01'), 'Language' (set to 'PySpark (Python)'), and 'Variables'. The status bar at the bottom left says 'Not started'. The main area contains a code cell with the following content:

```
1  %%pyspark
2  df = spark.read.load('abfss://wwi-02@asadatalake01.dfs.core.windows.net/sale/Year=2019/Quarter=Q4/Month=12/Day=20191201/sale-small-20191201-snappy.parquet',
3      format='parquet')
4  display(df.limit(10))
```

Below the code cell, a note says 'Press shift + enter to execute cells'. At the bottom of the notebook interface are buttons for '+ Code' and '+ Markdown'.



**View results in  
table format**

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

Search resources

Publish all Validate all Refresh Discard all

Develop Notebooks 13

NYCTaxi\_Docs\_Final \* SeattleSafetyDoc \* Repro \* PySpark (Python)

Cell 1 [3]

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

Command executed in 2mins 18s 412ms by euang on 11-22-2019 00:44:52.415 -08:00

Job execution In progress Spark 1 executors 4 cores

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

View in monitoring Spark history server

Cell 2 [5]

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

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

Cell 3

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

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

SQL support

View Table Chart

Aid Response

Medic Response

Automatic Fire Alarm False

Medic Response, 7 per Rule

Aid Response Yellow

MVI - Motor Vehicle Incident

Medic Response, 6 per Rule

Motor Vehicle Accident

Automatic Medical Alarm

IRED 1 Unit

Auto Fire Alarm

Automatic Fire Alarm Resd

Trans to AMR

longitude

Chart type: pie chart

X axis column: category

Y axis columns: longitude

Aggregation: COUNT

Y axis label: Total

X axis label: category

Apply Cancel

Cell 4 [7]

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

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

Publish all Validate all Refresh Discard all

Develop + <<

Data Download... \* NYCTaxi\_Docs\_... \* Cell Run all Publish Attach to Select Spark pool Language PySpark (Python)

Filter resources by name

Notebooks 13

- 00\_DataPrep
- 01\_Training/useMllib\_cleanup
- automl\_Arcadia\_validate
- Data Download\_GreenCab
- Data Download\_HolidayData
- Data Download\_Weather
- Data Download\_YellowCab
- Explore\_Join\_Aggregate
- \* NYCTaxi\_Docs\_Final
- NYCTaxi\_Docs\_Final\_PySpark
- Repro
- SeattleSafetyDoc
- SparkPerf

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 ($)')
8 ax1.set_ylabel('Counts')
9 plt.suptitle('')
10 plt.show()
11
12 # How many passengers tip'd by various amounts
13 ax2 = sampled_taxi_pd_df.boxplot(column=['tipAmount'], by=['passengerCount'])
14 ax2.set_title('Tip amount by Passenger count')
15 ax2.set_xlabel('Passenger count')
16 ax2.set_ylabel('Tip Amount ($)')
17 plt.suptitle('')
18 plt.show()
19
20 # Look at the relationship between fare and tip amounts
21 ax = sampled_taxi_pd_df.plot(kind='scatter', x='fareAmount', y='tipAmount', c='blue', alpha = 0.10, s=2.5*(sampled_taxi_pd_df['passengerCount']))
22 ax.set_xlabel('Fare Amount ($)')
23 ax.set_ylabel('Tip Amount ($)')
24 plt.axis([-2, 80, -2, 20])
25 plt.suptitle('')
26 plt.show()
27
```

Tip amount distribution

Tip amount by Passenger count

Exploratory data analysis with graphs – histogram, boxplot etc

# Transform Best practices

# Serverless SQL pools

---

- Co-locate storage and serverless SQL pools (including Cosmos DB analytical stores)
- Consider Azure Storage throttling
- Prepare files for querying (CSV, JSON -> Parquet)
- Push wildcards to lower levels in the path
- Use appropriate data types and check inferred data types
- Use filename and filepath functions to target specific partitions
- Use PARSE VERSION 2.0 to query CSV files
- Use CETAS to enhance query performance and joins
- Choose SAS credentials over Azure AD pass-through (for now)

# CCI vs Heap

---

- Transformations using Heap tables are generally faster than CCI. This is because rows need to be assembled from column stores on read tables, and columnar compression is needed on targets.
- The wider the table, and the more text fields it contains, the faster Heap is over CCI.
- Use Heap tables at transformation layer, use CCI tables where appropriate at presentation layer

# CCI best practices

---

- MAX data types not supported
- At least 1 million rows \* 60 distributions \* number of partitions
- At least 100k rows per batch, up to 1million
- Load using at least LARGERC or STATICCRC60
  - Create a loading user
- Minimal UPDATE and DELETE (or REBUILD frequently)

# Dedicated SQL – 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
```

# Serverless SQL – automatic statistics management

---

- Automatic creation available only for Parquet (manual for CSV)
- Same goes for recreation of statistics
- Only single-column statistics are currently supported
- CSV sampling not supported yet (only FULLSCAN)

# CTAS vs Insert / Update / Delete / Merge

---

- Prefer CTAS when you update or delete more than 10% of rows
- Prefer CTAS when you are updating or deleting a clustered Columnstore index, and do not have time for an offline rebuild

# Simple is better than clever

---

- Persist standard columns early, to avoid calculations and functions in WHERE clause
- Unroll CTEs and JOIN sub-selects to transient / temporary tables to manage distribution
- Simple queries are easier to tune and debug

## Pop Quiz 2

What is the optimal size for a rowgroup in columnstore format in a Synapse SQL Pool?

A)  
99,999

B)  
60,000,000

C)  
1,048,576



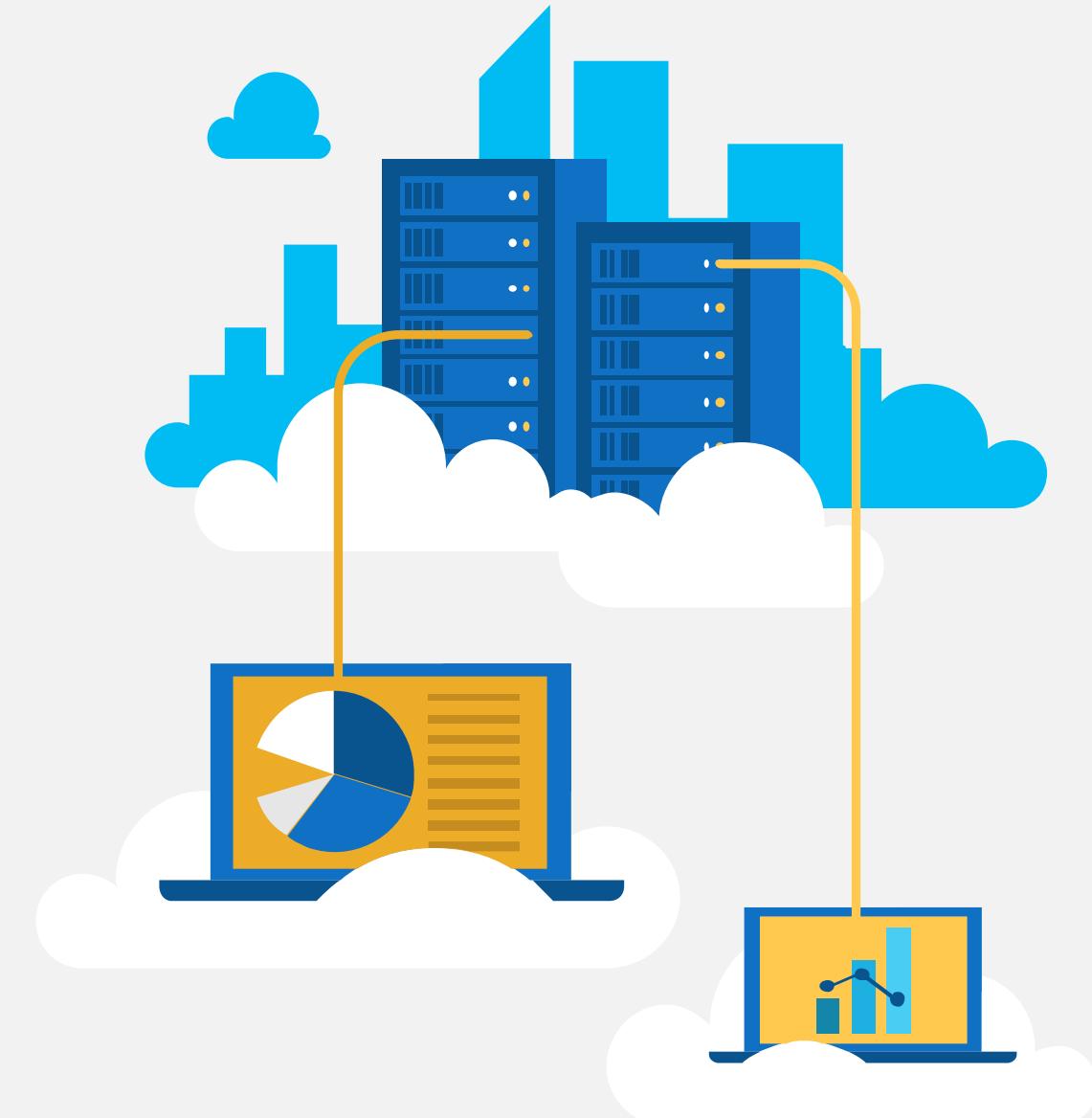
## Pop Quiz 2

What is the optimal size for a rowgroup in columnstore format in a Synapse SQL Pool?

A)  
99,999

B)  
60,000,000

C)  
**1,048,576**





# Thank you



# Full Group Activity: Data Engineering Discussion

**Objective:** As a result of participating in this activity, you will better be able to decide on which Azure Synapse Analytics component to use for specific data engineering scenarios.

**What you will be doing:** The facilitator will be posting questions to the entire group using an interactive tool called Mentimeter. The answers you post using Mentimeter will then drive the Data Engineering Discussion.

**Total Activity Time: 30 minutes**

## Mentimeter Poll

Scan QR Code

or

Go to [www.menti.com](http://www.menti.com) and use code

6142 6491



