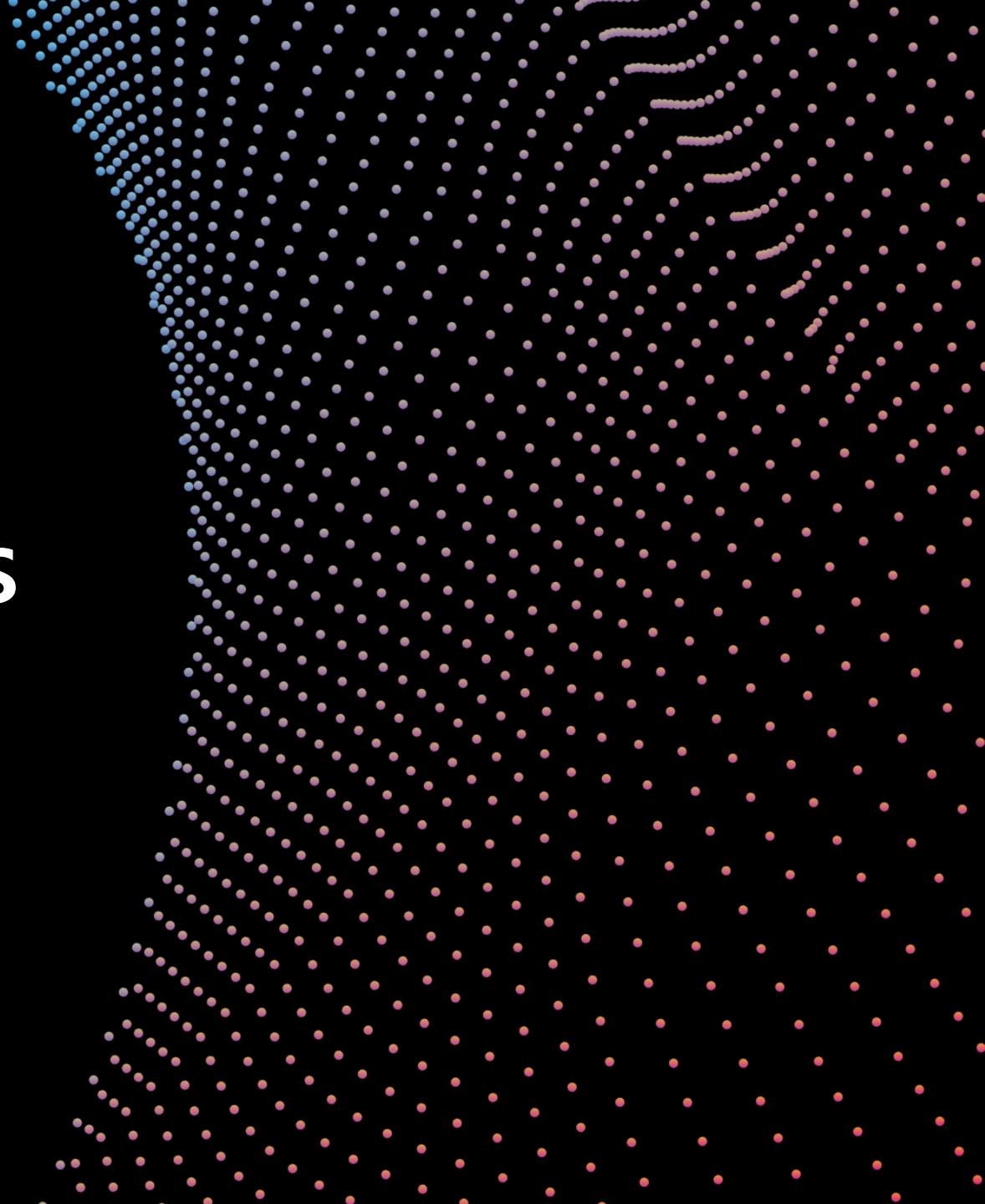


Azure Synapse Analytics

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	
Transform	9:45-10:30	Build Hands-on: Data Integration Part 1	Main Call
	10:30-11:30	Break	
	11:30-12:15	Data Transformations	
	12:15-12:30	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

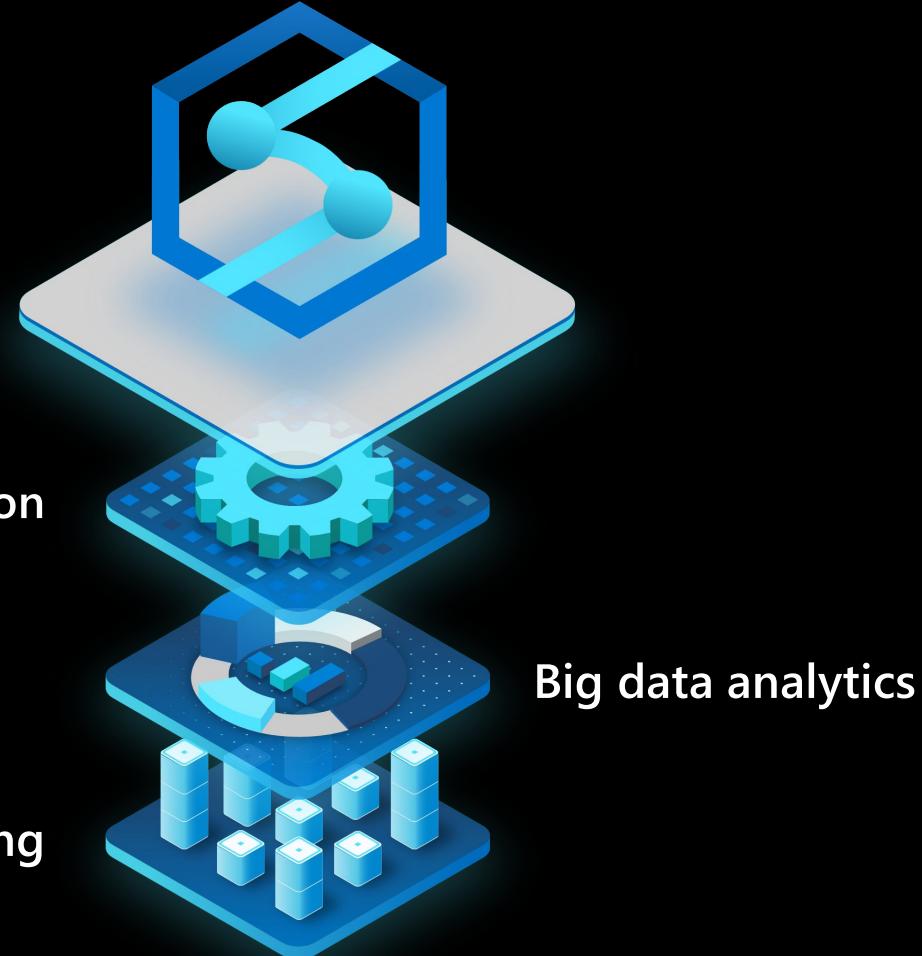
The first unified, cloud native platform for converged analytics

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

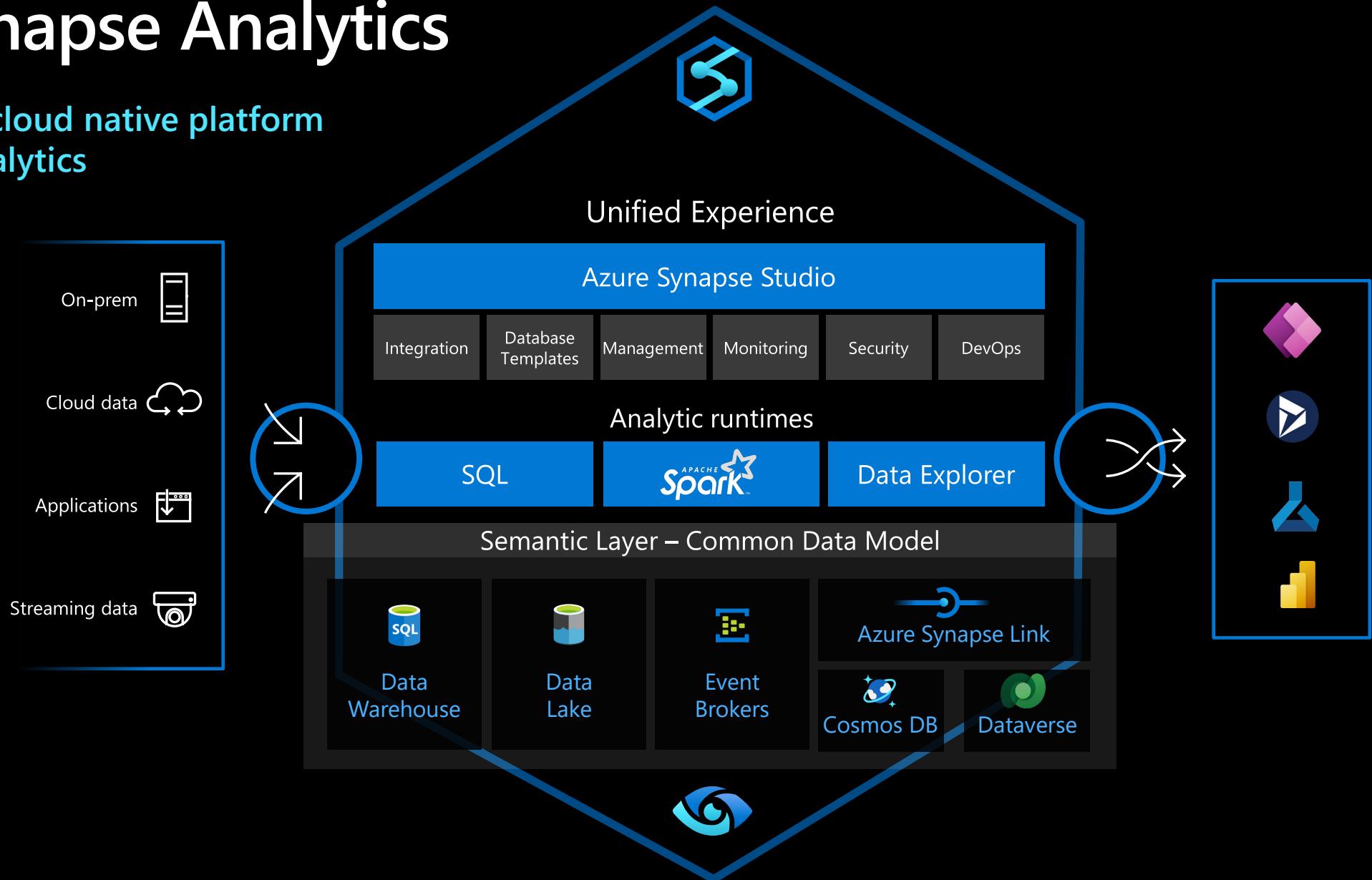
Data warehousing

Big data analytics

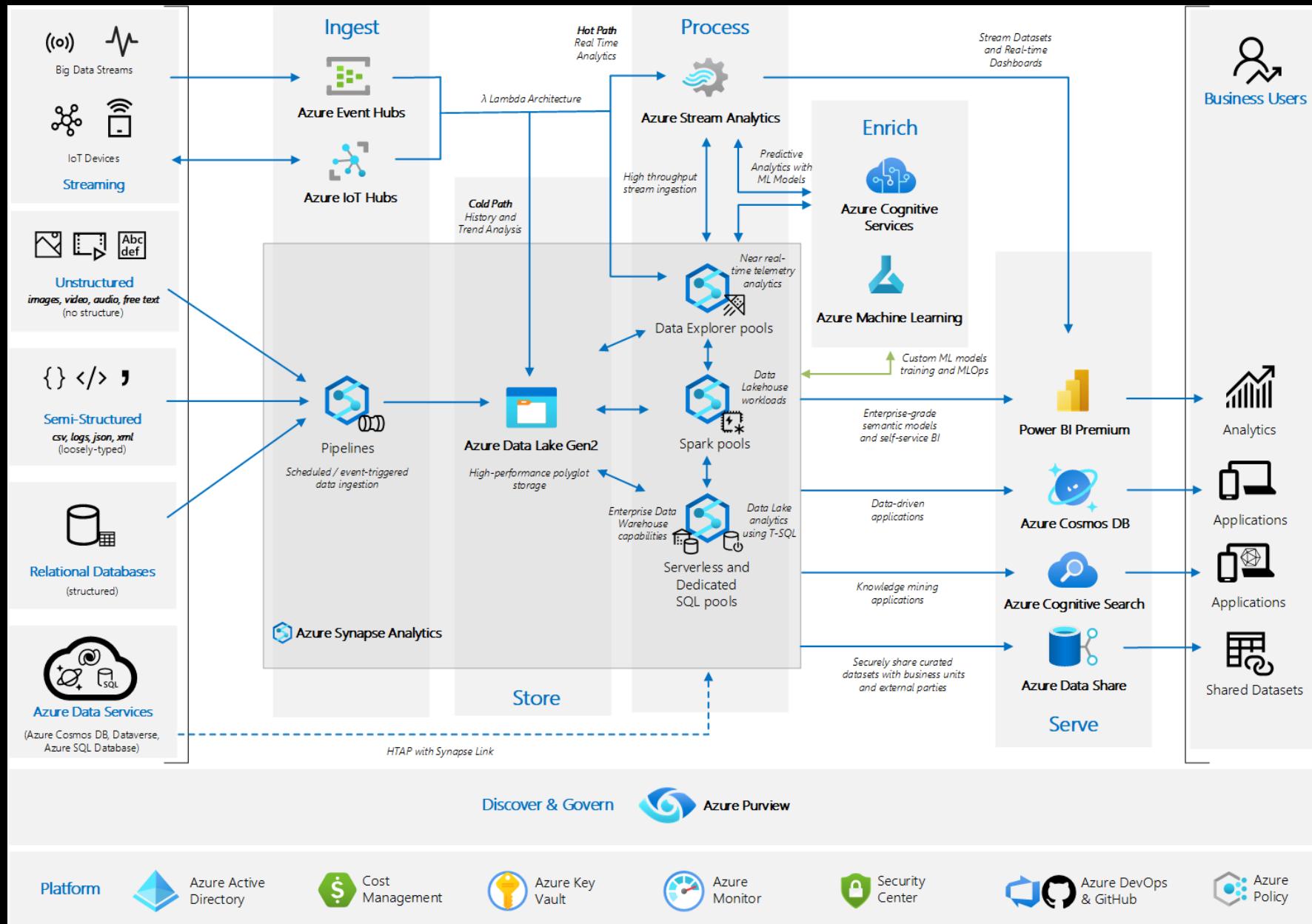


Azure Synapse Analytics

The first unified, cloud native platform
for converged analytics



Azure Synapse Analytics Reference Architecture



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. At the top, the navigation bar includes 'Microsoft Azure', 'Synapse Analytics', and 'wsazuresynapseanalytics'. A search bar labeled 'Search resources' is also present. The main area is titled 'Data' and has tabs for 'Workspace' and 'Linked'. A sidebar on the left lists various database objects: 'wwi.DimStockItem', 'wwi.DimSupplier', 'wwi.DimTransactionType', 'wwi.EmployeePIIData', 'wwi.FactMovement', 'wwi.FactOrder', and 'Columns' (with sub-items like 'OrderKey', 'CityKey', etc.). On the right, there's an 'SQL script 5' editor with 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 editor, there are tabs for 'Results' and 'Messages', with 'Results' selected. Under 'View', 'Table' is chosen, and there's a 'Chart' button and a 'Save as image' option. A bar chart titled 'Quantity' is displayed, showing data for various cities. The x-axis lists cities: Sekiu, Vennerborg, Harbour Pointe, Lake Stevens, Koontzville, Malott, College Place, Upper Preston, Point Roberts, and Trentwood. The y-axis ranges from 0 to 20k, with bars reaching approximately 18k for Sekiu and 15k for Vennerborg.

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, a vertical sidebar menu is highlighted with a red border, containing the following items: Home, Data, Develop, Integrate, Monitor, and Manage. A red arrow points from the 'Integrate' item in this menu to the 'Integrate' hub icon on the main page. The main content area is titled 'Synapse workspace wsazuresy' and features six activity hubs arranged in a grid:

- 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.

The top navigation bar includes 'Microsoft Azure', 'Synapse Analytics', 'wsazuresynapseanalytics', user information 'someone@microsoft.com', and the 'MICROSOFT' logo.

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 home page. 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. The main area features three main sections: "Ingest" (cloud icon), "Explore and analyze" (bar chart icon), and "Visualize" (3D bar chart icon). Red callout boxes and arrows highlight several features: "Updates" (near the workspace name), "Switch Workspaces" (near the "New" button), "Language Settings" (inside the "Visualize" section), "Help Knowledge Center" (inside the "Visualize" section), and "Feedback" (near the top right). The bottom of the page includes sections for "Discover more" (Knowledge center, Browse partners) and "Recent resources" (DW Load User Creation, DW Creation Script).

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

Synapse Analytics workspace
asa-[REDACTED]-POC

New

Updates

Switch Workspaces

Language Settings

Help Knowledge Center

Feedback

Ingest

Explore and analyze

Visualize

Discover more

Knowledge center

Browse partners

Recent resources

Name	Last opened by you
DW Load User Creation	6 days ago
DW Creation Script	6 days ago

Home hub

Overview

New dropdown – offers quickly start work item

Recent resources – Lists recently opened workspace artifacts for quick access

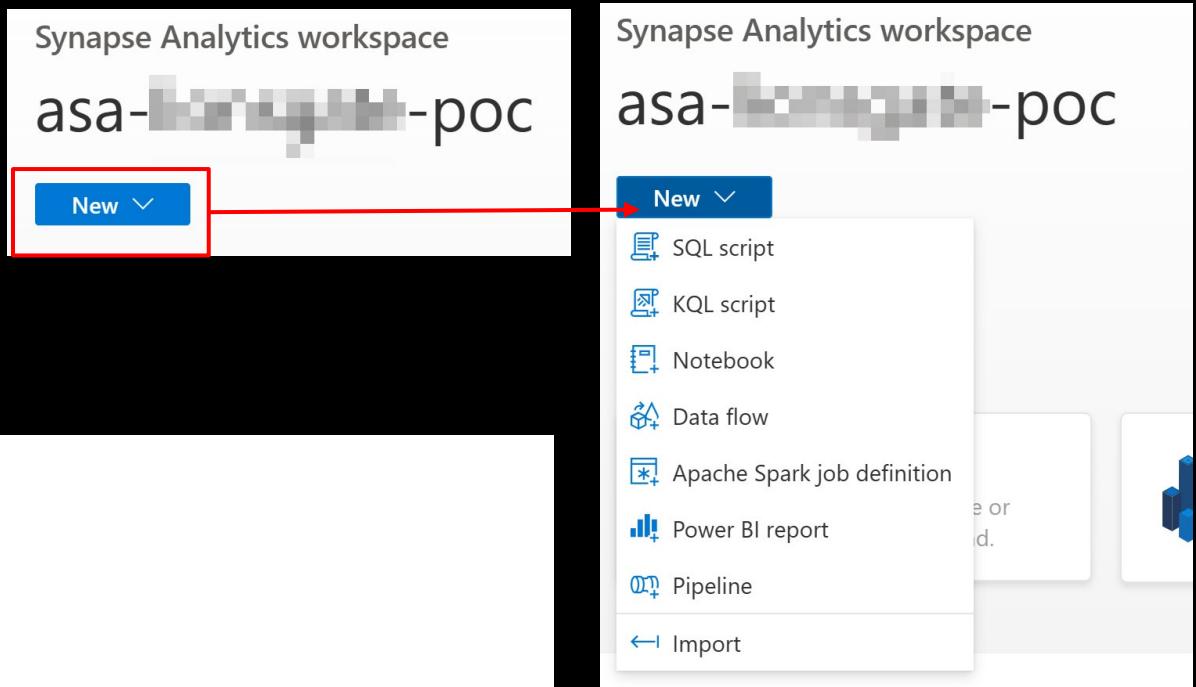
Discover more

 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 ▾



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

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

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

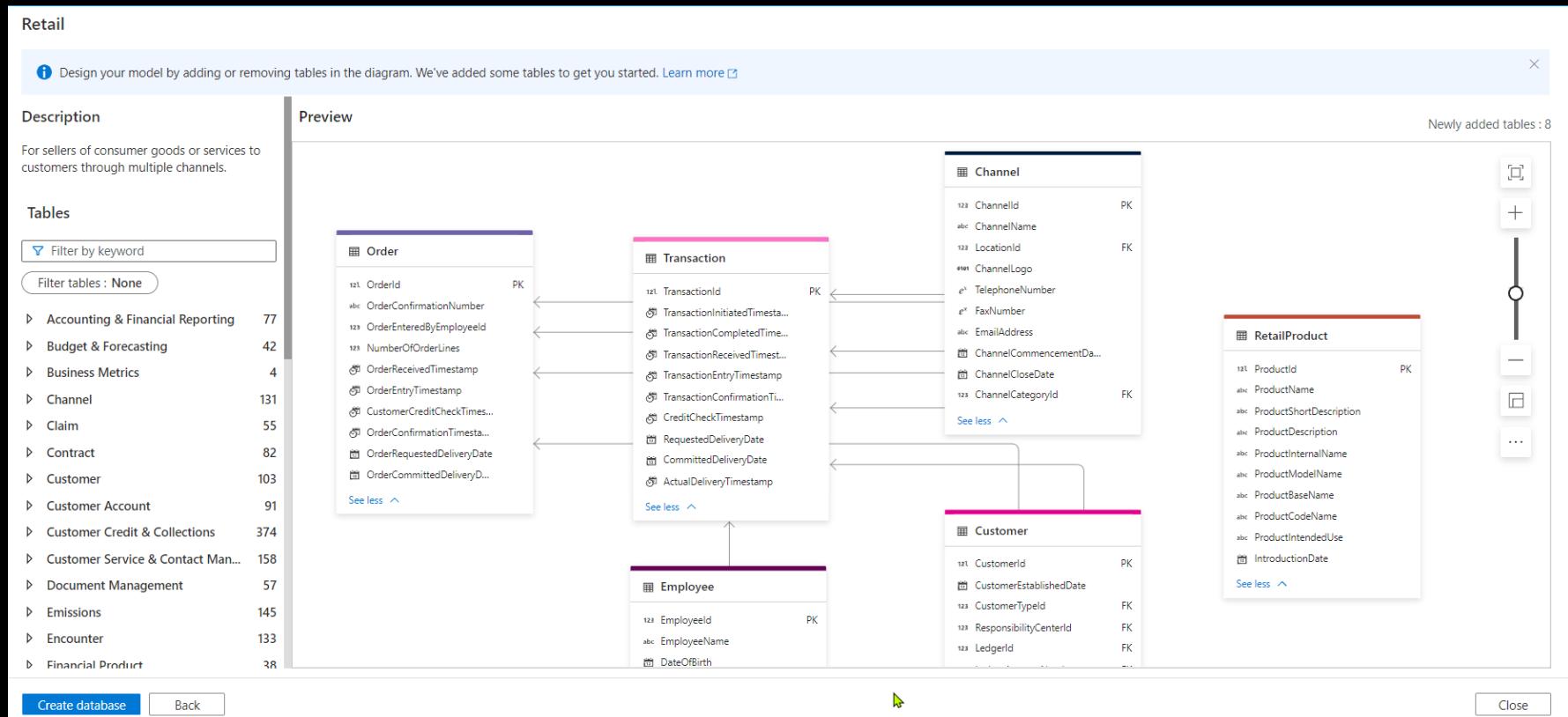
Knowledge center – database templates

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 hub interface. The left sidebar has icons for Home, Data, Develop, Integrate, Monitor, and Manage. The main area is titled 'Data' and has tabs for 'Workspace' (which is 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 hub interface. The left sidebar has icons for Home, Data, Develop, Integrate, Monitor, and Manage. The main area is titled 'Data' and has tabs for 'Workspace' and 'Linked' (which is 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

Linked Cosmos DB Analytical Store

Linked Azure Data Explorer

Linked ADLS Gen2 Account

Container (filesystem)

The screenshot shows the Microsoft Azure Synapse Analytics Data hub interface. The left sidebar displays 'Data' resources, categorized under 'Linked'. It lists four items: 'Azure Blob Storage' (3 items), 'Azure Cosmos DB' (1 item), 'Azure Data Explorer' (2 items), and 'Azure Data Lake Storage Gen2' (1 item). The 'Azure Data Lake Storage Gen2' item is expanded, showing a sub-item 'wsazuresynapseanalytics (Primary...)' which is also expanded, revealing 'rawdata' and 'staging' containers. Red boxes highlight the 'Azure Cosmos DB' (item 1), 'Azure Data Explorer' (item 2), and the 'rawdata' container. A red arrow points from the 'Linked Cosmos DB Analytical Store' text to the 'Azure Cosmos DB' item. Another red arrow points from the 'Linked Azure Data Explorer' text to the 'Azure Data Explorer' item. A third red arrow points from the 'Container (filesystem)' text to the 'rawdata' container. The main workspace area shows a file browser titled 'rawdata' with a path 'rawdata > taxidata'. The 'File path' bar shows 'rawdata' and 'taxidata'. Below the path is a table listing files in 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.parq...	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.parq...	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.parq...	8/27/2020, 12:32:21 AM		966.1 MB

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

Data hub – Storage accounts

Preview a sample of your data

The screenshot shows the Azure Data Explorer interface with three main panes:

- Left Pane (Data):** A navigation tree under the "Linked" workspace. It includes sections for Azure Blob Storage (4), Azure Cosmos DB (2), Azure Data Explorer (1), Azure Data Lake Storage Gen2 (8), and two specific data湖 (asaworkspace01 and asadatalake01) with their own sub-folders like dev, staging, tempdata, trigger, and wwi.
- Middle Pane:** A workspace named "wwi-02". It contains a breadcrumb trail: "wwi-02 > sale-csv". Below it is a table with a single row labeled "wwi-factsale.csv". A context menu is open over this row, with the "Preview" option highlighted by a red box and a red arrow pointing to it.
- Right Pane:** A preview of the "Products.csv" file. It shows the following details:
 - Path:** <https://azuresynapsesa.dfs.core.windows.net/rawdata/sample csv files/Products.csv>
 - Modified:** 10/27/2020, 8:38:51 PM
 - With column header:** On (with a toggle switch)A preview table is displayed with 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

OK button is at the bottom right of the preview pane.

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: The "Data" sidebar shows a tree structure of data resources. The "Linked" tab is selected, displaying "Azure Blob Storage" (4), "Azure Cosmos DB" (2), "Azure Data Explorer" (1), "Azure Data Lake Storage Gen2" (8), and several workspace and data lake nodes under "asadata01" and "asadatalake01". The node "wwi-02" is highlighted in grey.

Middle Panel: The main workspace shows a folder structure: "wwi-02" > "sale-csv". A file named "wwi-factsale.csv" is selected. A context menu is open, listing options: Preview, New SQL script, New notebook, New data flow, New integration dataset, Manage access..., Rename..., Download, Delete, and Properties... (the last option is highlighted with a red box).

Right Panel: The "Properties" dialog is displayed, containing the following fields:

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

Data

+

Workspace **Linked**

Filter resources by name

► Azure Blob Storage 4

► Azure Cosmos DB 2

► Azure Data Explorer 1

▲ Azure Data Lake Storage Gen2 8

► asaworkspace01 (Primary - asadata...)

▲ asadatalake01 (asadatalake01)

- dev
- staging
- tempdata
- trigger
- wwi

wwi-02

New SQL script New notebook New c

← → ↘ ↙ ↗ wwi-02 > sale-csv

Name

wwi-factsale.csv

Preview

New SQL script

New notebook

New data flow

New integration dataset

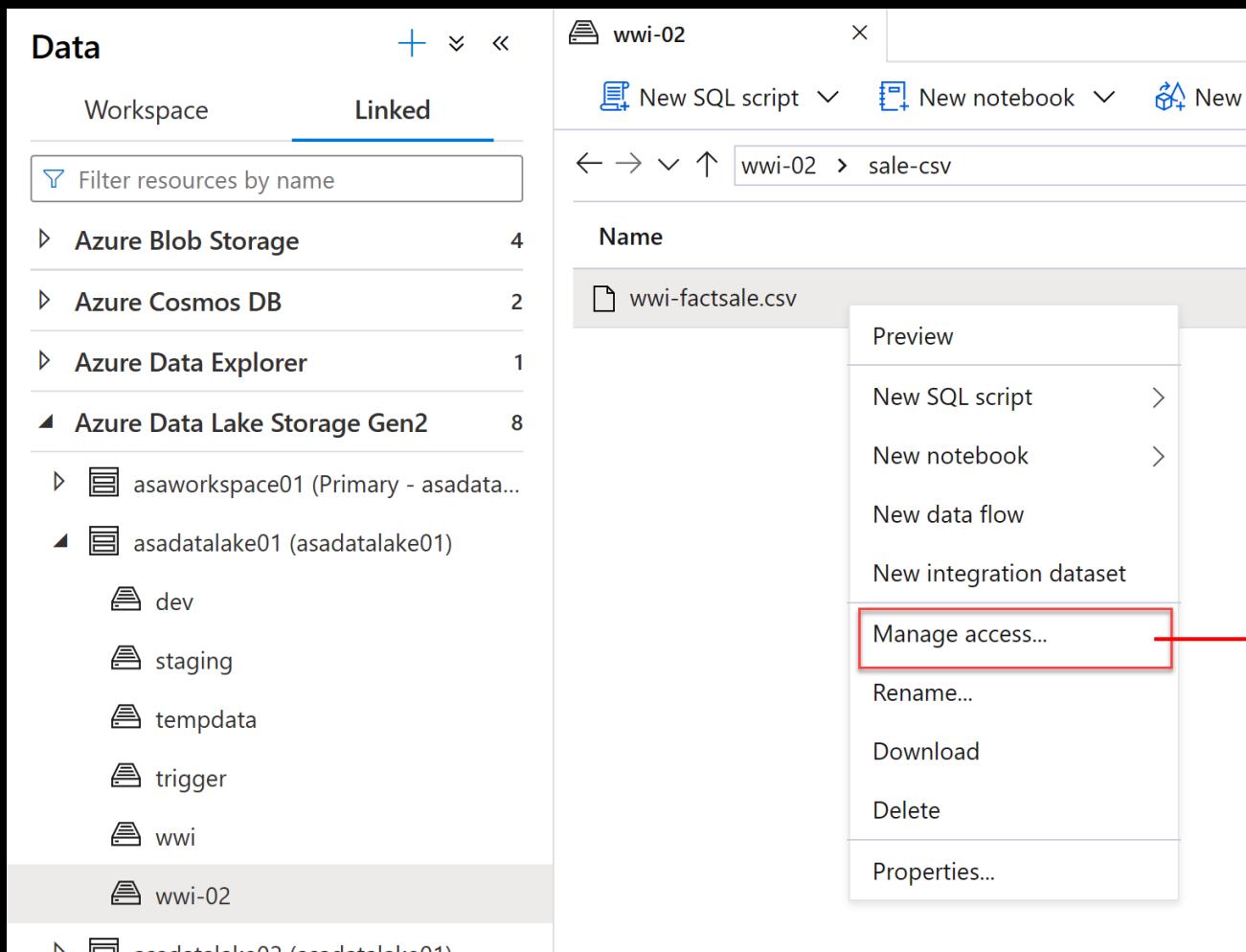
Manage access...

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

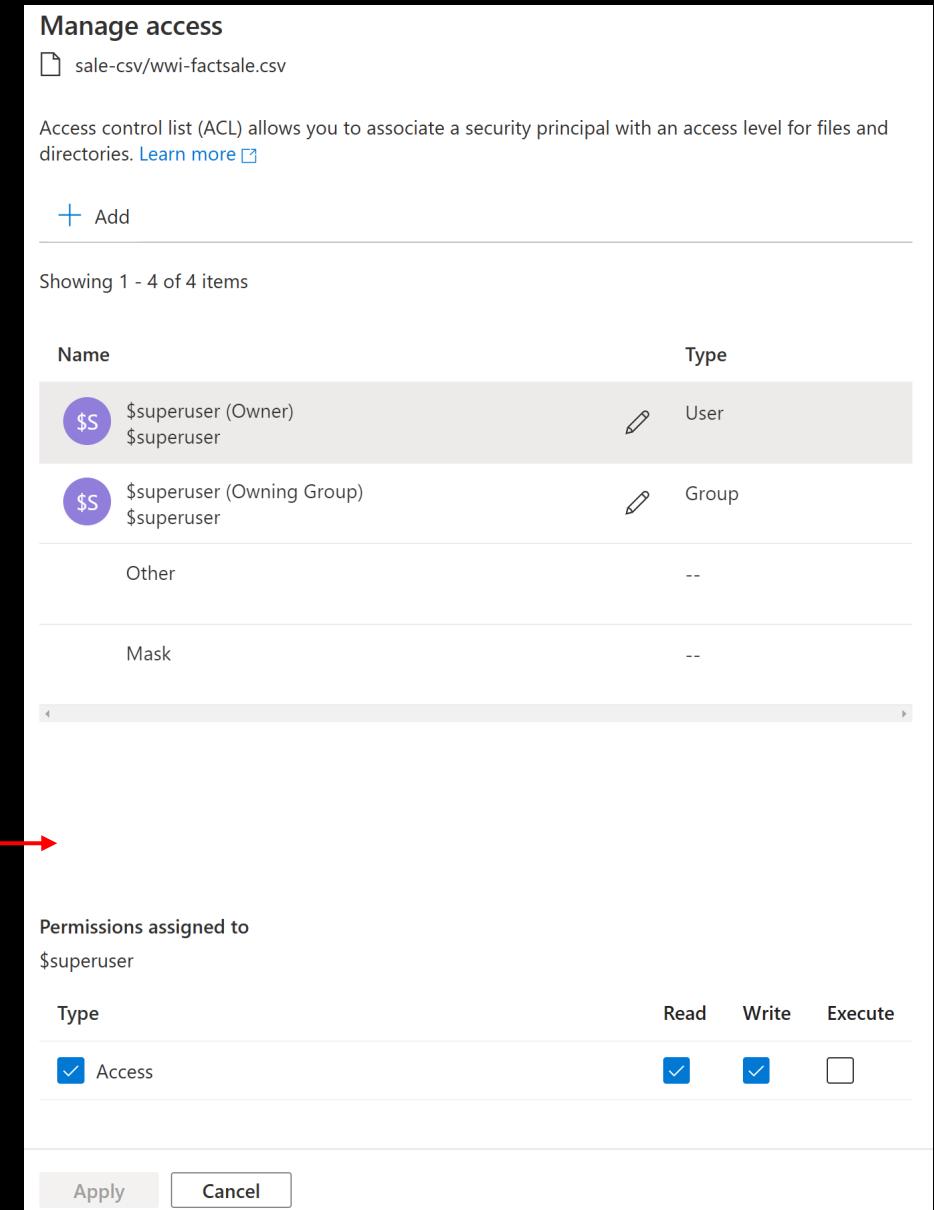
Name	Type
\$superuser (Owner) \$superuser	User
\$superuser (Owning Group) \$superuser	Group
Other	--

Mask

Permissions assigned to \$superuser

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

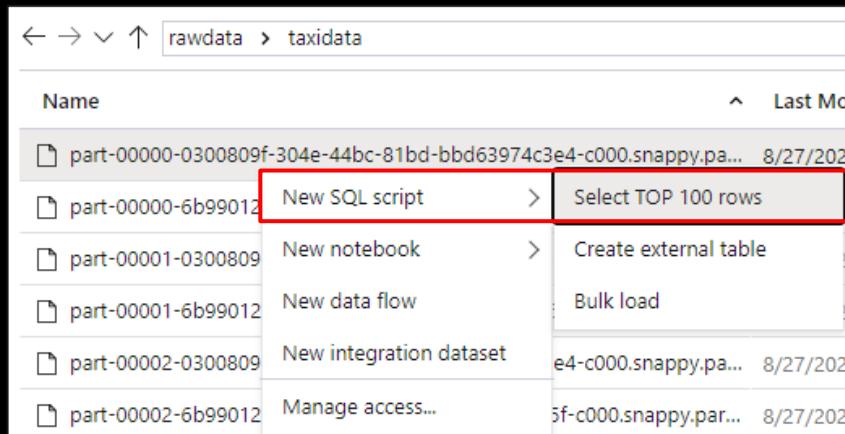
Apply Cancel



Data hub – storage accounts

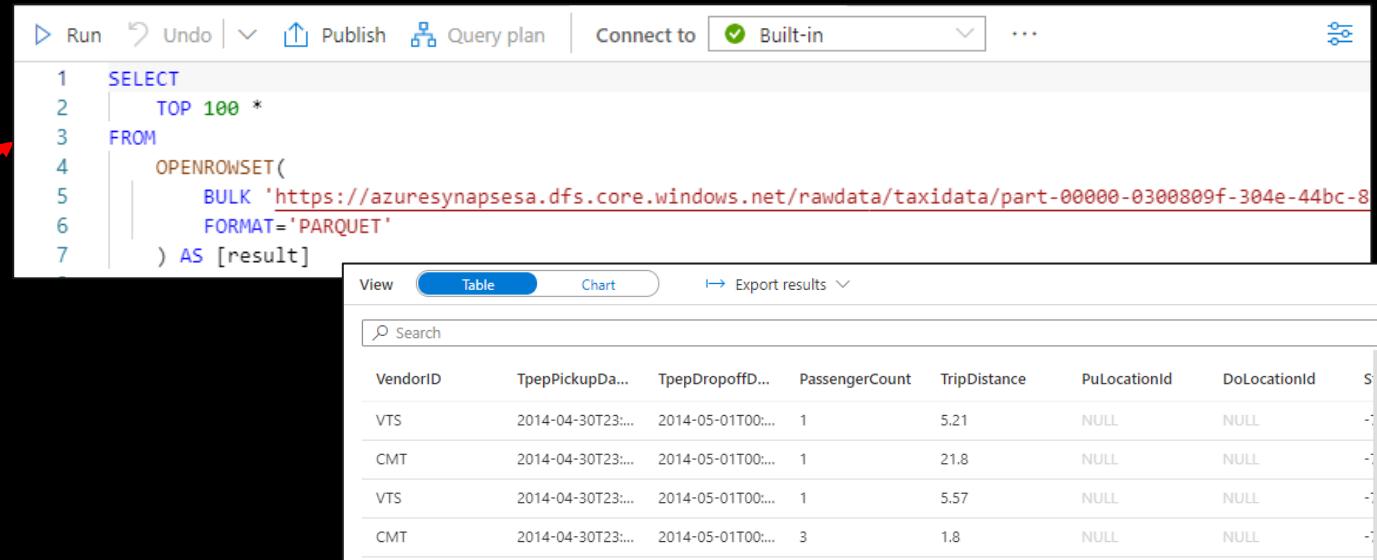
Two simple gestures to start analyzing with SQL scripts or with notebooks.

T-SQL or PySpark auto-generated.



rawdata > taxidata

Name	Last Modified
part-00000-0300809f-304e-44bc-81bd-bbd63974c3e4-c000.snappy.parquet	8/27/2022
part-00000-6b99012	New SQL script > Select TOP 100 rows
part-00001-0300809	New notebook > Create external table
part-00001-6b99012	New data flow Bulk load
part-00002-0300809	New integration dataset e4-c000.snappy.parquet 8/27/2022
part-00002-6b99012	Manage access... 5f-c000.snappy.parquet 8/27/2022

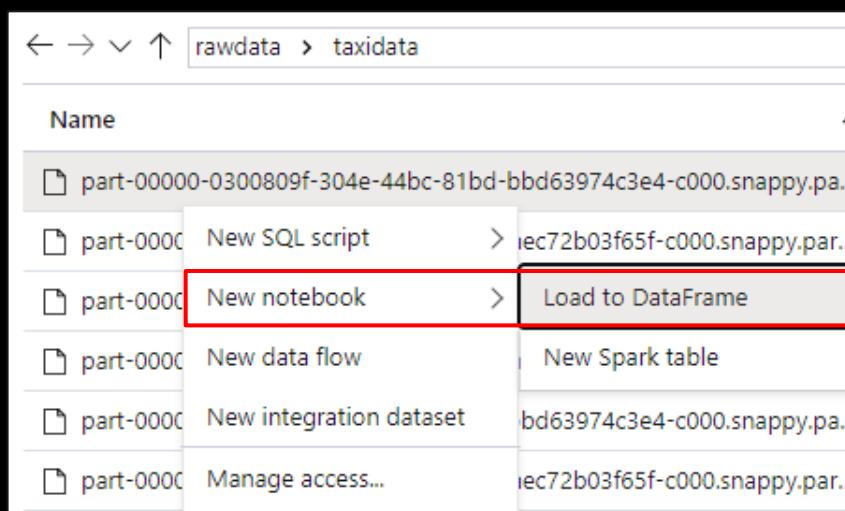


Run Undo Publish Query plan Connect to Built-in ...

```
1 SELECT
2     TOP 100 *
3     FROM
4     OPENROWSET(
5         BULK 'https://azuresynapsesa.dfs.core.windows.net/rawdata/taxidata/part-00000-0300809f-304e-44bc-8
6         FORMAT='PARQUET'
7     ) AS [result]
```

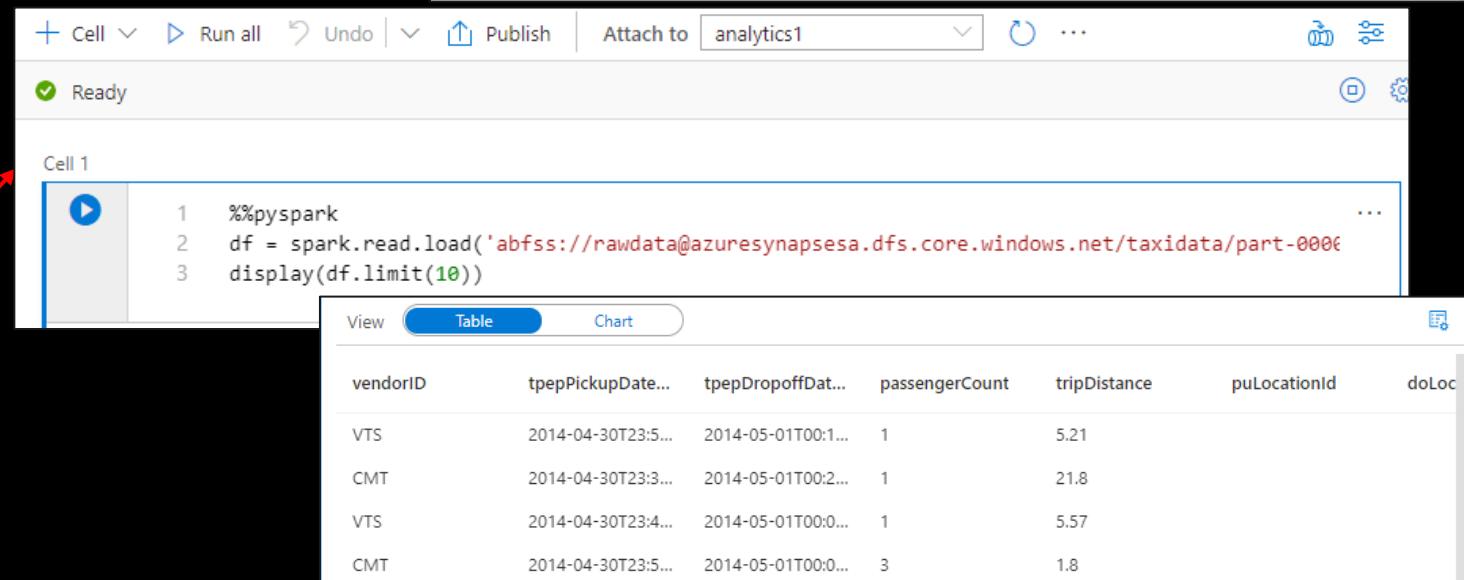
View Table Chart Export results

VendorID	TpepPickupDate	TpepDropoffDate	PassengerCount	TripDistance	PuLocationId	DoLocationId	...
VTS	2014-04-30T23:5...	2014-05-01T00:1...	1	5.21	NULL	NULL	-
CMT	2014-04-30T23:3...	2014-05-01T00:2...	1	21.8	NULL	NULL	-
VTS	2014-04-30T23:...	2014-05-01T00:...	1	5.57	NULL	NULL	-
CMT	2014-04-30T23:...	2014-05-01T00:...	3	1.8	NULL	NULL	-



rawdata > taxidata

Name	
part-00000-0300809f-304e-44bc-81bd-bbd63974c3e4-c000.snappy.parquet	
part-00000-6b99012	New SQL script > iec72b03f65f-c000.snappy.parquet
part-00000-6b99012	New notebook > Load to DataFrame
part-00000-6b99012	New data flow New Spark table
part-00000-6b99012	New integration dataset bd63974c3e4-c000.snappy.parquet
part-00000-6b99012	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	2014-04-30T23:5...	2014-05-01T00:1...	1	5.21			-
CMT	2014-04-30T23:3...	2014-05-01T00:2...	1	21.8			-
VTS	2014-04-30T23:...	2014-05-01T00:...	1	5.57			-
CMT	2014-04-30T23:5...	2014-05-01T00:0...	3	1.8			-

Data hub – databases

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

Lake database

Dedicated SQL pool

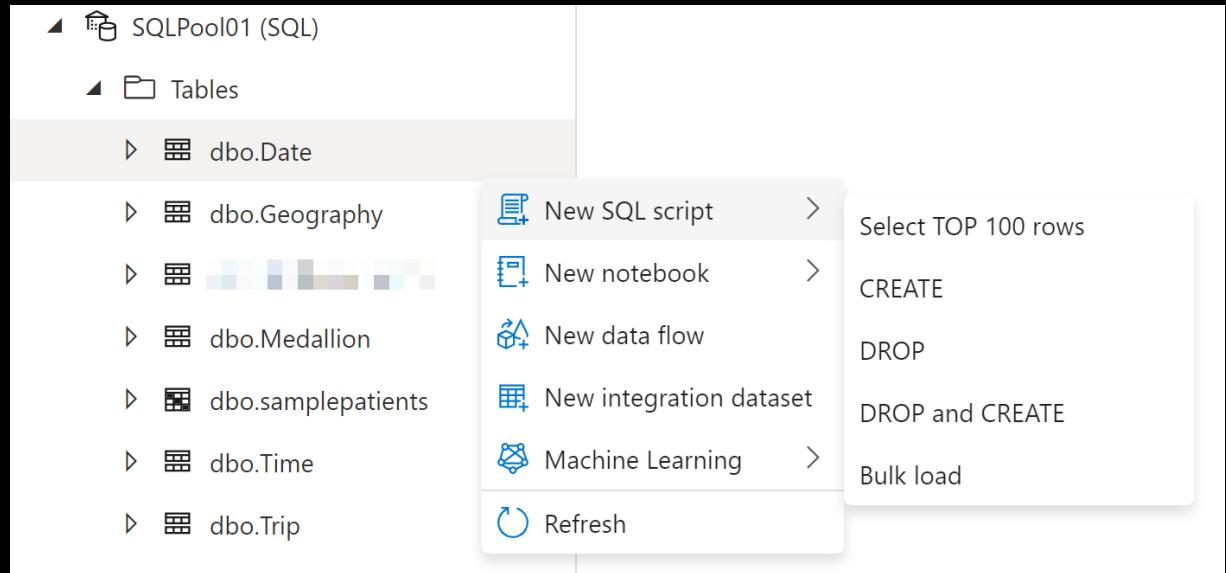
Serverless SQL pool

The screenshot shows the 'Data' hub interface with the 'Workspace' tab selected. A search bar at the top allows filtering resources by name. Below it, two main categories are listed: 'Lake database' and 'SQL database'. The 'Lake database' section contains two items: 'asa_db01' and 'default'. The 'SQL database' section contains seven items: 'SQLPool01 (SQL)', 'SQLPool02 (SQL)', 'SQLPool03 (SQL)', 'SQLPool04 (SQL)', and three unnamed items represented by small grey icons. Three red arrows point from the text labels on the left to the corresponding database entries in the workspace view: one arrow points to 'asa_db01' under 'Lake database', another points to 'SQLPool01 (SQL)' under 'SQL database', and a third points to 'SQLServerlessDB (SQL)' under 'SQL database'.

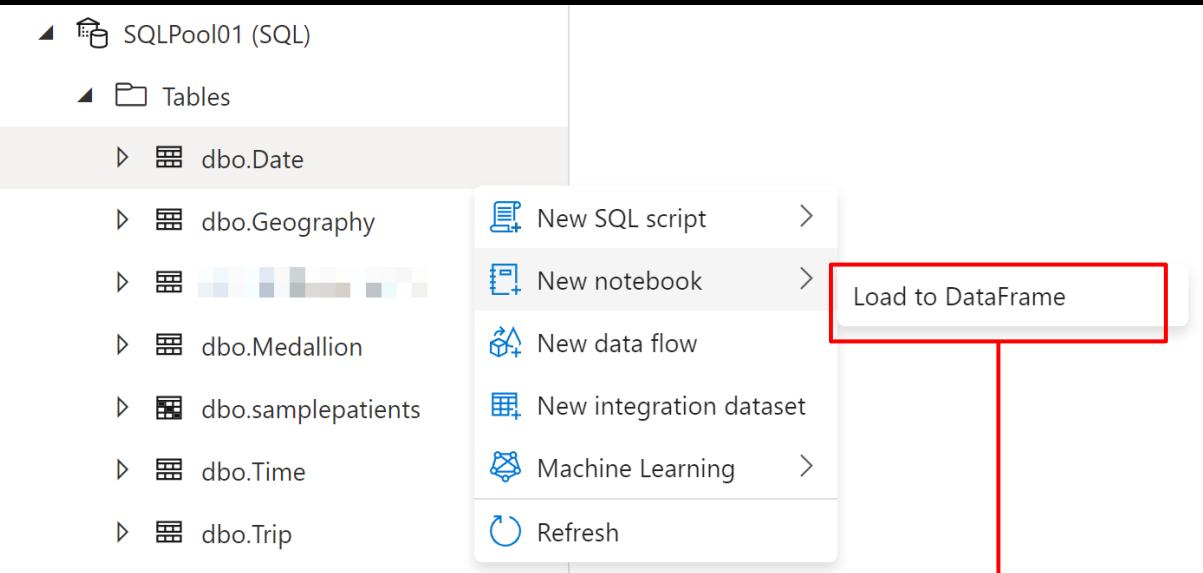
The screenshot shows the 'Data' hub interface with the 'Linked' tab selected. A search bar at the top allows filtering resources by name. Below it, two main categories are listed: 'Lake database' and 'SQL database'. The 'Lake database' section contains two items: 'asa_db01' and 'default'. The 'SQL database' section contains seven items: 'SQLPool01 (SQL)', 'SQLPool02 (SQL)', 'SQLPool03 (SQL)', 'SQLPool04 (SQL)', and three unnamed items represented by small grey icons. Three red arrows point from the text labels on the left to the corresponding database entries in the linked view: one arrow points to 'asa_db01' under 'Lake database', another points to 'SQLPool01 (SQL)' under 'SQL database', and a third points to 'SQLServerlessDB (SQL)' under 'SQL database'.

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 SparkPool01', 'Language (Spark (Scala))', and 'Variables'. The notebook content shows a cell with the following PySpark code:

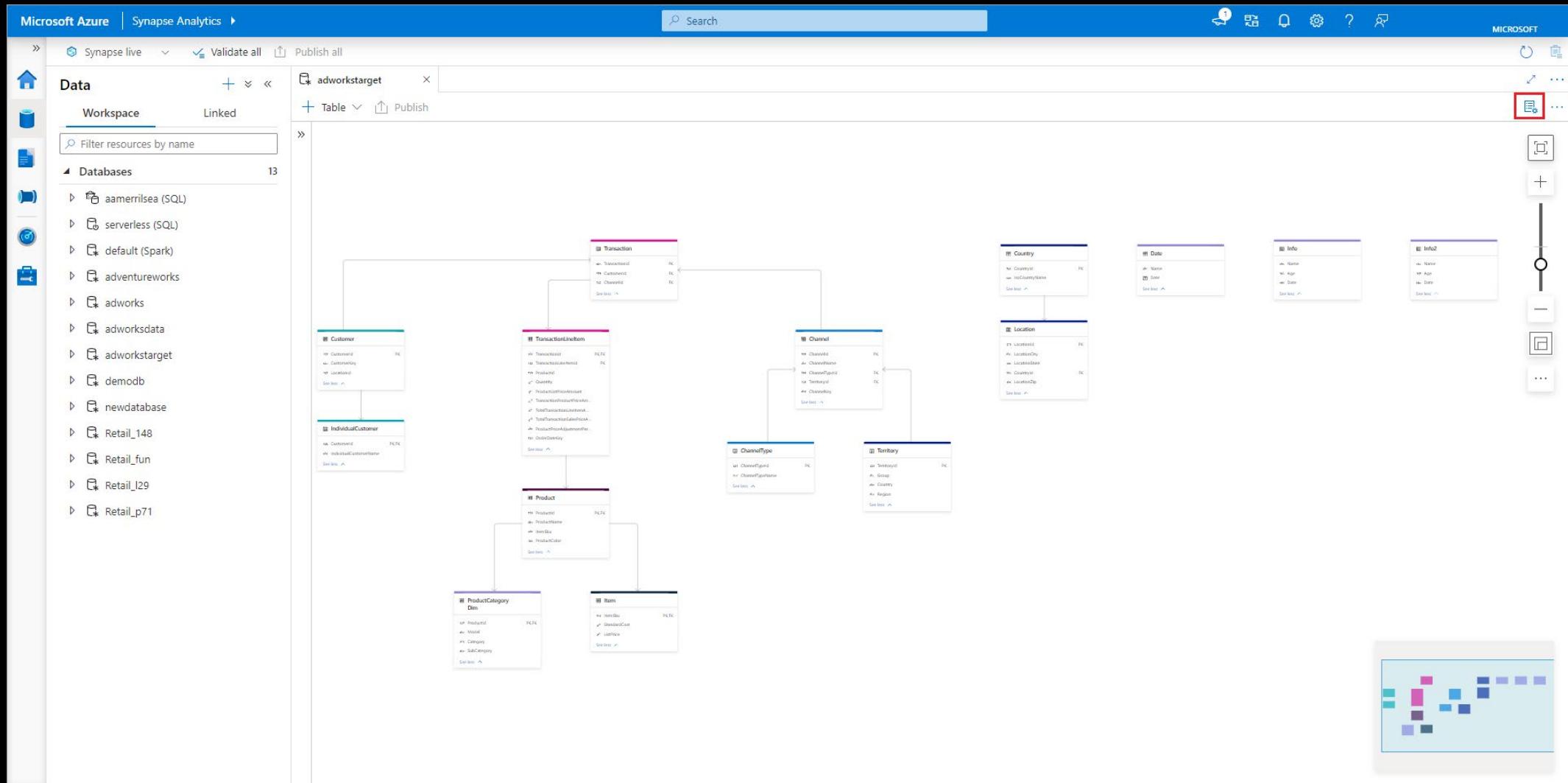
```
1 %%spark
2 val df = spark.read.sqlAnalytics("SQLPool01.dbo.Date")
3 df.write.mode("overwrite").saveAsTable("default.t1")
```

The cell status is 'Not started'. Below the cell, a note says 'Press shift + enter to execute cells'. At the bottom right are buttons for '+ Code' and '+ Markdown'.

Data hub – databases

Lake database designer (in preview)

Visually compose database tables, columns, and relationships



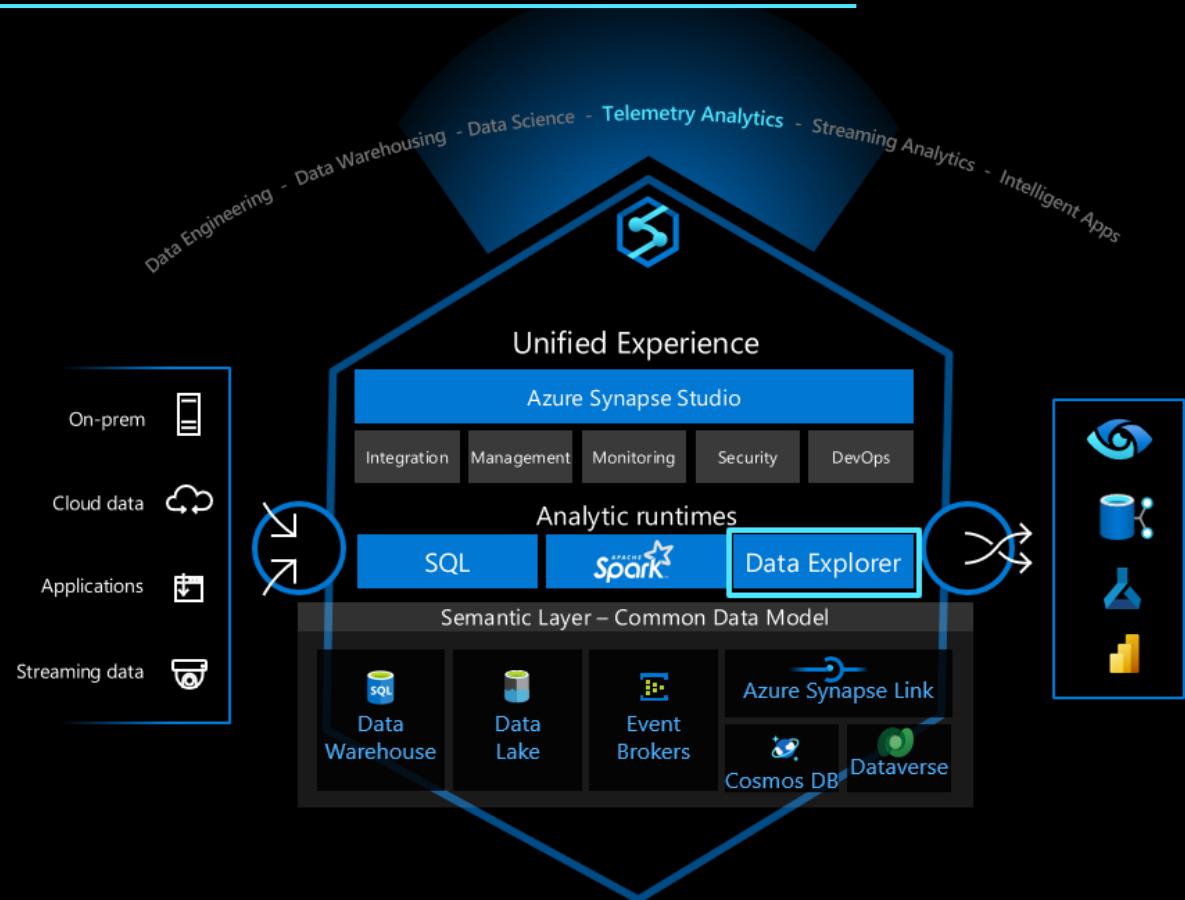
Data Hub - Azure Synapse data explorer

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 shows the Azure Data Hub datasets interface. On the left, a sidebar titled 'Data' has tabs for 'Workspace' and 'Linked'. The 'Linked' tab is selected, showing a list of resources under 'Filter resources by name'. Below this, categories like 'Azure Blob Storage', 'Azure Cosmos DB', 'Azure Data Explorer', 'Azure Data Lake Storage Gen2', and 'Integration datasets' are listed. A red box highlights the 'Integration datasets' section, and a red arrow points from this box to the 'asal400_sales_adlsgen2' dataset on the main canvas.

The main canvas displays the 'asal400_sales_adlsgen2' dataset details. At the top, there's a preview icon (blue grid) and the file format 'Parquet'. The dataset name is 'asal400_sales_adlsgen2'. Below the preview, three tabs are visible: 'Connection', 'Schema', and 'Parameters'. The 'Connection' tab is active, showing a dropdown for 'Linked service' set to 'asadatalake01'. Other fields include 'Integration runtime' (set to 'LargeAzureComputeOptimizedIntegr...'), 'File path' ('wwi-02 / sale / File'), and 'Compression type' ('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 'Workspace' tab is selected, showing a list of datasets: wwi.DimStockItem, wwi.DimSupplier, wwi.DimTransactionType, wwi.EmployeePIIData, wwi.FactMovement, wwi.FactOrder, and Columns. Below this is a search bar labeled 'Filter resources by name'. On the right, a central area contains an 'SQL script 5' tab with the following T-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' being active. Under 'Results', there are buttons for 'View' (Table, Chart), 'Chart' (selected), and 'Save as image'. A bar chart titled 'Quantity' is displayed, showing the total quantity for various cities. The x-axis lists cities: Sekiu, Venerborg, Harbour Pointe, Lake Stevens, Koontzville, Malott, College Place, Upper Preston, Point Roberts, and Trentwood. The y-axis shows values from 0 to 20k, with Sekiu having the highest value around 18k.

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]
9
```

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]
9
```

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:

The chart displays a fluctuating line graph representing the total amount over time. The x-axis shows dates from 2019-01-01 to 2019-12-31, and the y-axis ranges from 0 to 60. The data shows significant weekly volatility with some seasonal trends.

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

Export results

CSV
JSON
XML

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.

The screenshot shows the Azure Data Studio interface for a notebook named "Notebook 3". The top bar includes "Run all", "Undo", "Publish", "Outline", "Attach to" (set to "SparkPool01"), "Language" (set to "PySpark (Python)"), "Variables", and a gear icon. A red box highlights the "Language" dropdown menu, which lists "PySpark (Python)", "PySpark (Python)", "Spark (Scala)", ".NET Spark (C#)", and "Spark SQL".

The code cell contains:

```
1 df = spark.read.load('abfss://wwi-02@asadatalake01.dfs.core.windows.net/sale/Year'
2 display(df.limit(10))
```

The output shows the results of the PySpark command:

TransactionId	CustomerId	ProductId	Quantity	Price	TotalAmount
9453b35d-3f0d-44c8-bc5d-482a...	5	1588	1	20.190000000000000	20.190000000000000
9453b35d-3f0d-44c8-bc5d-482a...	5	4882	2	29.370000000000000	58.740000000000000
9453b35d-3f0d-44c8-bc5d-482a...	5	3283	2	28.330000000000000	56.660000000000000
9453b35d-3f0d-44c8-bc5d-482a...	5	4777	2	28.350000000000000	56.700000000000000
9453b35d-3f0d-44c8-bc5d-482a...	5	4908	2	35.400000000000000	70.800000000000000
9453b35d-3f0d-44c8-bc5d-482a...	5	4479	4	30.420000000000000	121.680000000000000
9453b35d-3f0d-44c8-bc5d-482a...	5	2779	2	14.500000000000000	29.000000000000000
9453b35d-3f0d-44c8-bc5d-482a...	5	1917	3	26.670000000000000	80.010000000000000
9453b35d-3f0d-44c8-bc5d-482a...	5	1382	3	26.770000000000000	80.310000000000000
9453b35d-3f0d-44c8-bc5d-482a...	5	1664	3	24.990000000000000	74.970000000000000

Below the table are buttons for "+ Code" and "+ Markdown".

This screenshot shows the same notebook interface as above, but the "Export results" button is highlighted with a red box. It displays options for CSV, JSON, and XML.

The code cell is identical to the one above:

```
1 df = spark.read.load('abfss://wwi-02@asadatalake01.dfs.core.windows.net/sale/Year'
2 display(df.limit(10))
```

The output shows the results of the PySpark command:

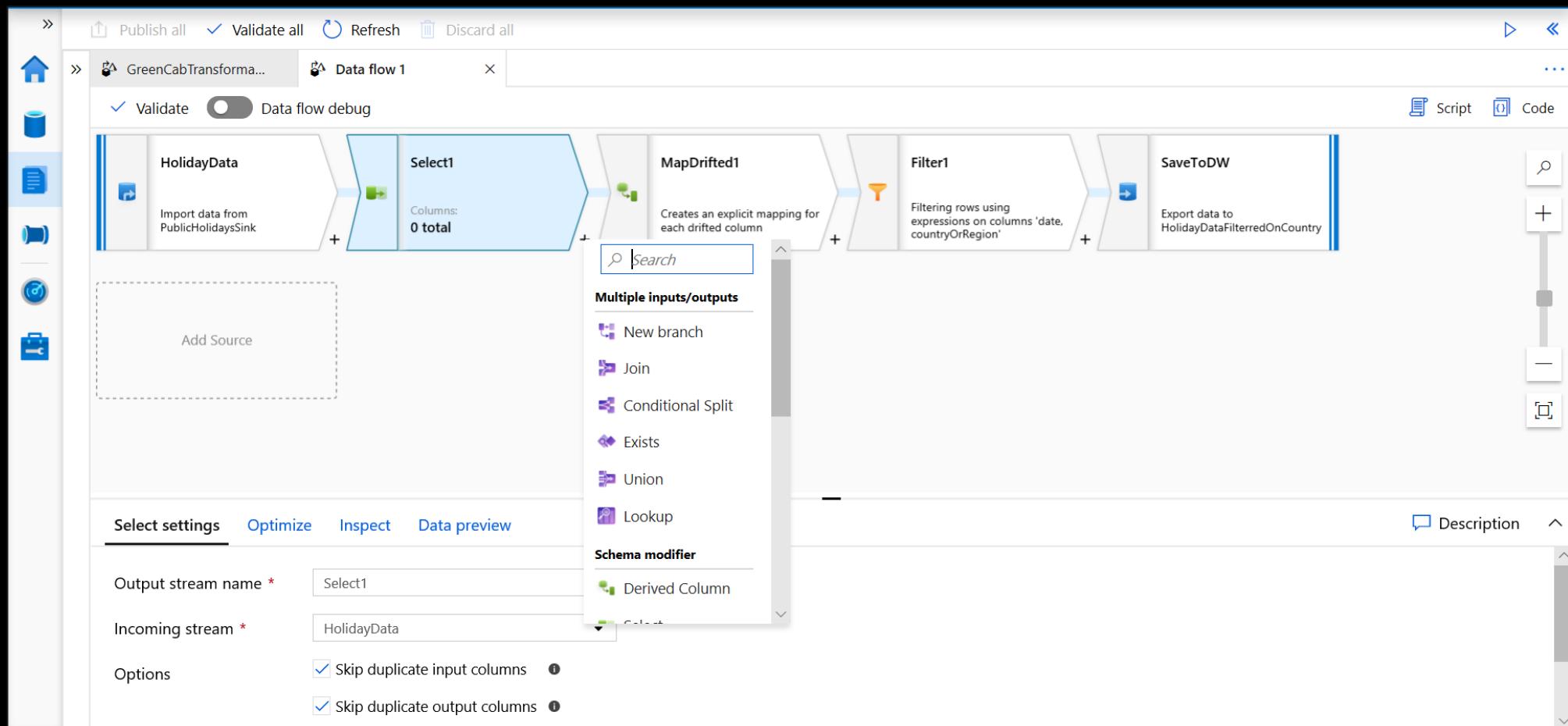
TransactionId	CustomerId	ProductId
9453b35d-3f0d-44c8-bc5d-482a...	5	1588
9453b35d-3f0d-44c8-bc5d-482a...	5	4882

The "Export results" button is highlighted with a red box, showing options for CSV, JSON, and XML.

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

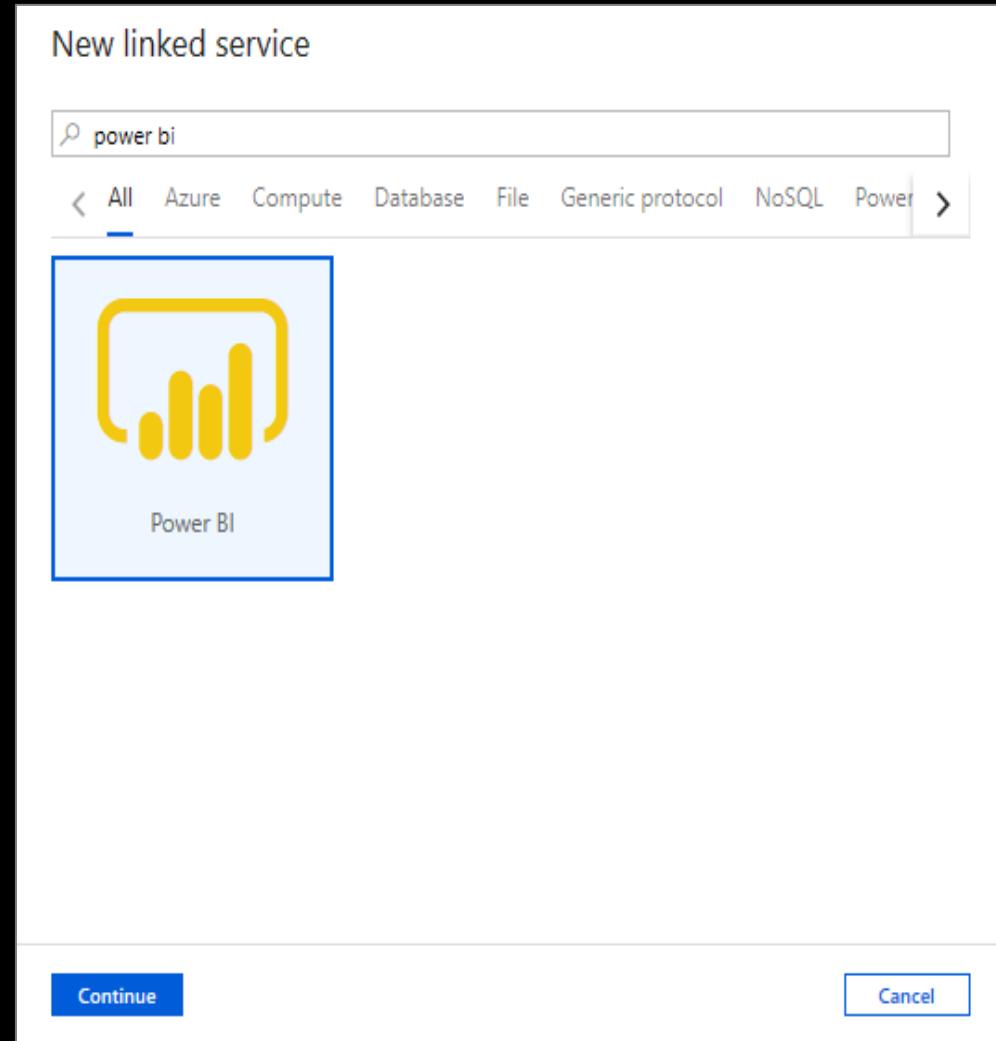
Overview

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



A screenshot of the Microsoft Azure Synapse Analytics 'Develop' hub. The top navigation bar shows 'Microsoft Azure | Synapse Analytics'. On the left, there is a sidebar with icons for Home, Database, Notebook, Data Flow, Spark Job Definition, Power BI, and a folder icon. The 'Power BI' icon is highlighted. The main area has a search bar with the placeholder 'Filter resources by name'. Below the search bar, there is a list of resources: 'SQL scripts' (27), 'Notebooks' (11), 'Data flows' (2), 'Spark job definitions' (1), 'Power BI' (1), 'Demo Synapse' (with 'Power BI datasets' and 'Power BI reports' listed under it). There are also 'Publish all' and 'Validate all' buttons at the top right.

Develop hub – Power BI

View published reports in Power BI workspace

Microsoft Azure | Synapse Analytics > gamingtelemetry

Search resources

PUBLISH ALL VALIDATE ALL REFRESH DISCARD ALL

Develop Report

Ask a question EXPLORE TEXT BOX SHAPES BUTTONS VISUAL INTERACTIONS

prlangad@microsoft.com MICROSOFT

Filter resources by name

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

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 Sep 2019 Oct 2019 Nov 2019

Tabular Map Forecast Extra Users

VISUALIZATIONS FIELDS

Search

agegroup forecast historical platform predictions realtime regions scenario weekdays

Values Add data fields here

DRILL THROUGH

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

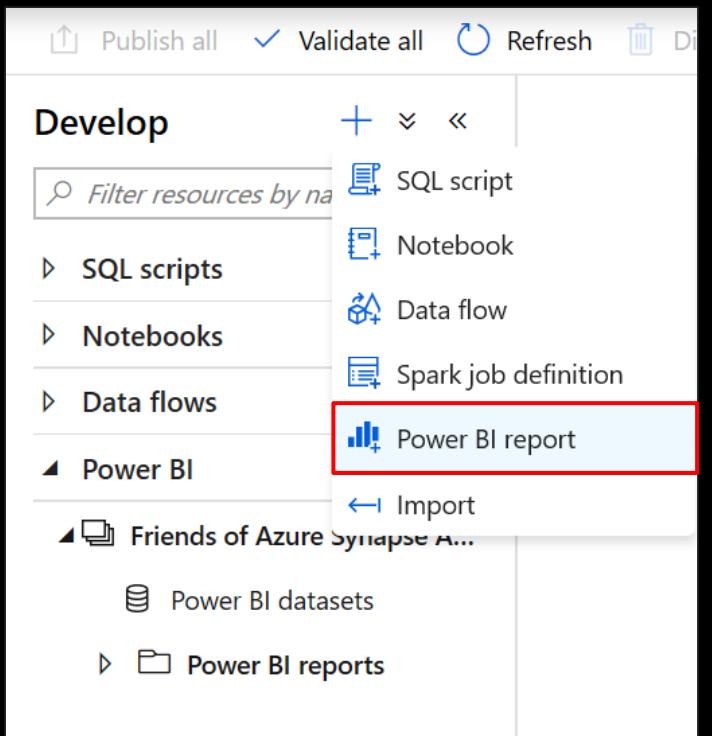
The screenshot shows the Microsoft Power BI Develop hub interface. The main area displays a published report titled "GAME STUDIO". The report includes a "What If" analysis section where users can increase free game addons by a percentage, showing a forecast of 7,361,707 users. It also features a chart titled "Total Users vs 'What If' Analysis" comparing actual users (24.5M) with a forecast (7,346.3K). A "What If" Analysis Forecast chart tracks user growth from August 2019 to November 2019. The sidebar on the left lists various Power BI resources, and the right side shows the "VISUALIZATIONS" and "FIELDS" panes.

Develop Hub – Power BI

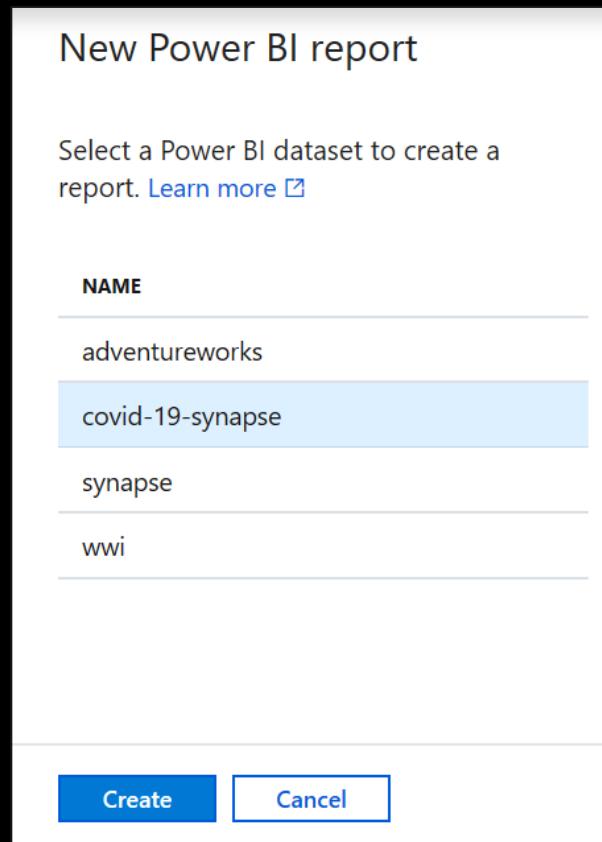
Create new reports from existing published Power BI datasets

Create new Power BI datasets

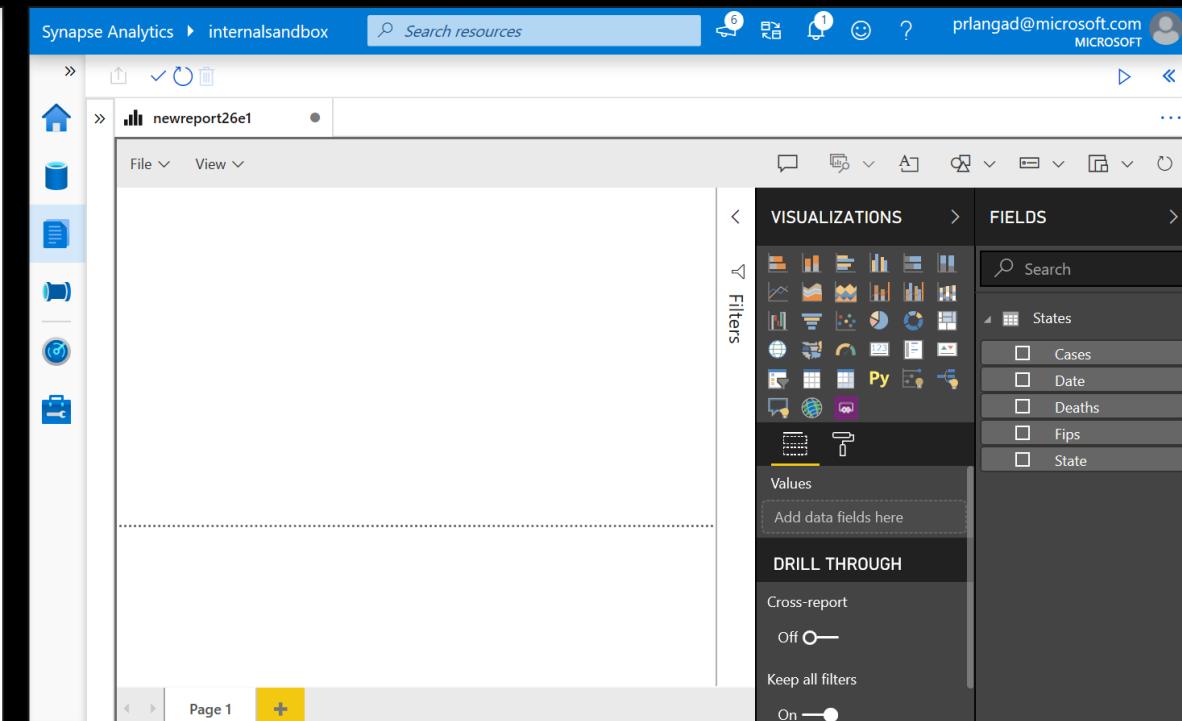
1



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 Report

Filter resources by name

SQL scripts 9
Notebooks 6
Data flows 1
Power BI 1
gaming-telemetry
Power BI datasets
Power BI reports 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, Sep 2019, Oct 2019, Nov 2019

Tabular Map Forecast Extra Users

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

Historical Forecast Predictions +

The screenshot shows a detailed Power BI report titled "GAME STUDIO" within the "gamingtelemetry" workspace in Microsoft Azure. The report includes a main visual showing a white Xbox controller on a console, a "Total Users" summary of 24.5M, and a "What If..." analysis section where users can increase free game addons by 1. It also features a "Total Users vs 'What If' Analysis" table and a "What If" Analysis Forecast chart showing user growth from August to November 2019. The interface includes a sidebar for developing the report and a right-hand pane for managing 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 selected, and 'Report' is highlighted. A red box and arrow point to the 'Save' button in the top right corner of the main report area, which says 'Save this report'. A large callout box at the bottom left contains the text 'Publish changes by simply saving the report in workspace'. The main report area displays a 'GAME STUDIO' dashboard with a 'What If...' analysis section, a table of 'Total Users vs "What If" Analysis', and a line chart of 'What If' Analysis Forecast. The right side of the screen shows the 'VISUALIZATIONS' and 'FIELDS' panes.

Microsoft Azure | Synapse Analytics > gamingtelemetry

Develop

Save this report

What If...

Users (Forecast) 7,613,619

Extra Users 504.8K

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

Historical Forecast Predictions

Visualizations Fields

VISUALIZATIONS FIELDS

Values

Add data fields here

DRILL THROUGH

Cross-report Off Keep all filters On

Add drill-through fields here

agegroup forecast historical platform predictions realtime regions scenario weekdays

Historical Forecast Predictions

Forecast Extra Users

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

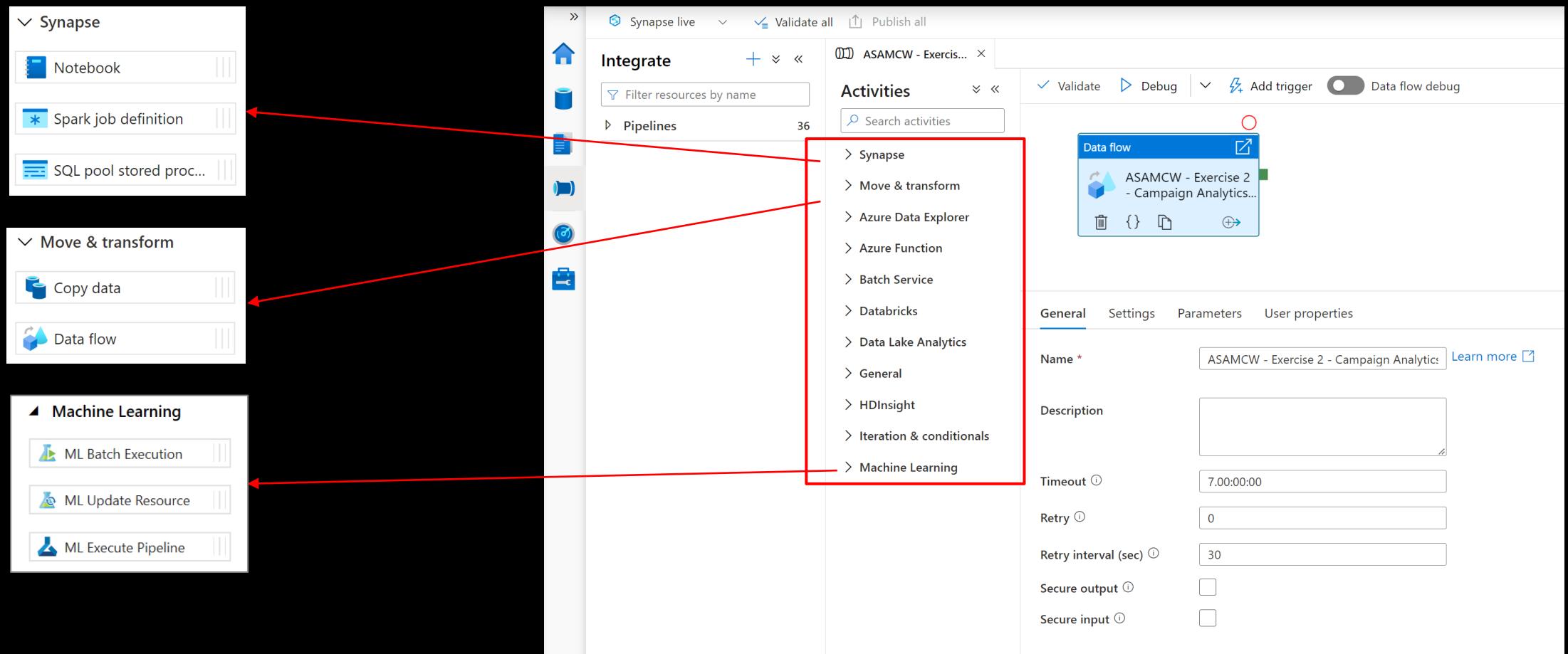
Usage

```
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 lists various management categories: Analytics pools (selected), SQL pools, 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, and Git configuration. The main content area is titled "SQL pools" and displays information about serverless and dedicated SQL pools. A message states: "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." Below this is a "New" button and a "Refresh" button. A "Filter by name" input field is present. A table lists five items, showing details such as Name, Type, Status, and Size. The table data is as follows:

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

Continue Cancel

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

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 interface. On the left, there's a navigation sidebar with icons for Analytics pools, SQL pools, Apache Spark pools, Data Explorer pools (preview), External connections, Linked services, Azure Purview, Integration, Triggers (which is highlighted with a red box), Integration runtimes, and Security. The main area is titled 'Triggers' with the sub-instruction 'To execute a pipeline set the trigger. Triggers'. Below this, there's a 'New' button (also highlighted with a red box) and a 'Filter by name' input field. A table lists two triggers: 'Blob-storage-trigger-01' (Storage events, Stopped) and 'cdm-trigger' (Storage events, Started). To the right of the main area, a large modal window titled 'New trigger' is open, containing 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. An arrow points from the 'New' button in the main interface to the 'New trigger' modal.

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 main menu to two separate 'Add role assignment' dialog boxes.

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

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

Source control

Git configuration

Microsoft Azure | Synapse Analytics > wsazuresynapseanalytics

main branch

Filter...

dev branch

main branch

workspace_publish branch

Create pull request [Alt+P]

New branch [Alt+N]

Switch to live mode

Validate all Commit all Publish

Configure a repository

Connect your workspace with your Git repository just within please view document here.

Setting Disconnect

Repository type GitHub

GitHub account SynapseTestDemo

Git repository name synapsetestdemo-ws-01

Collaboration branch dev

Publish branch main

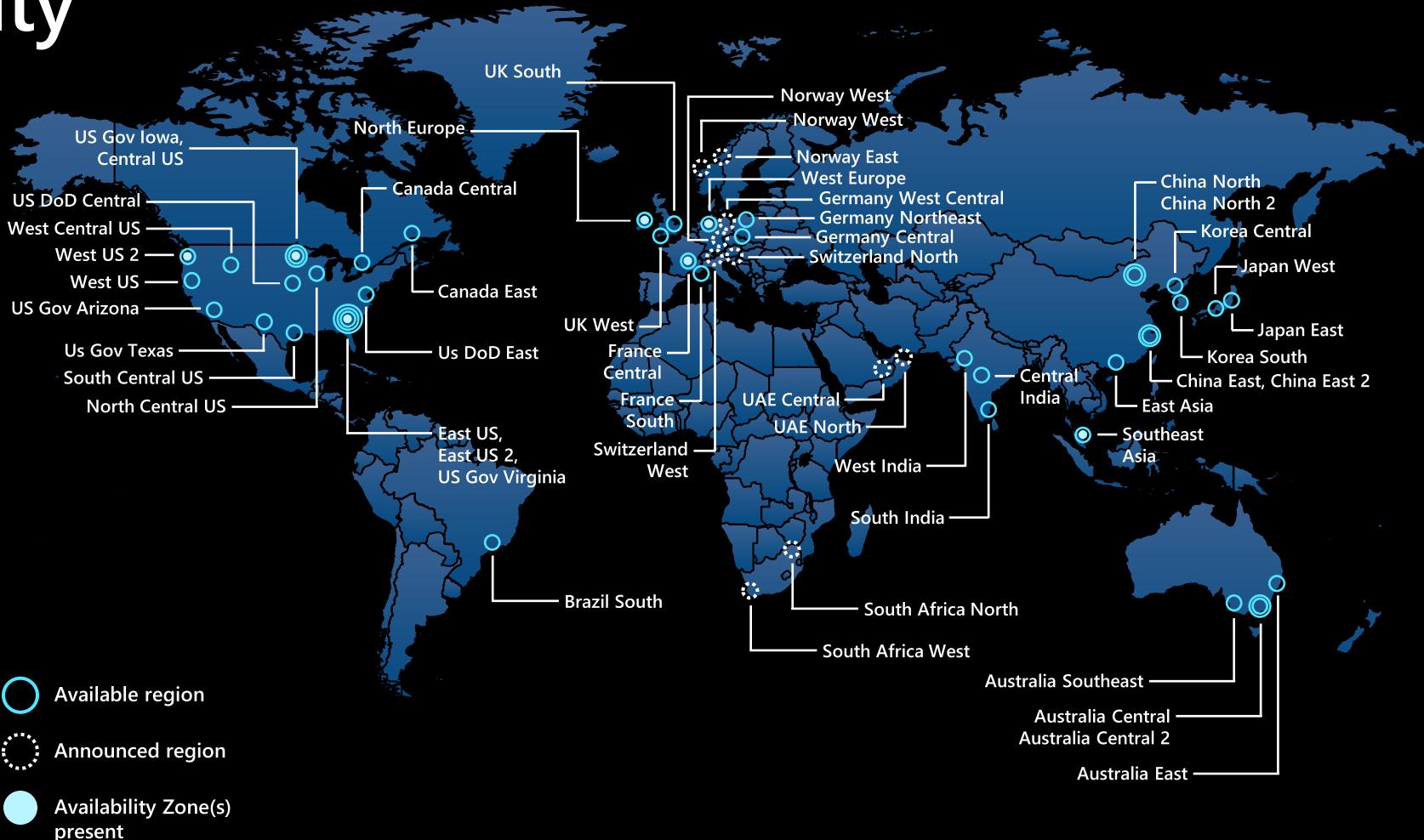
Root folder /

Source control

Git configuration

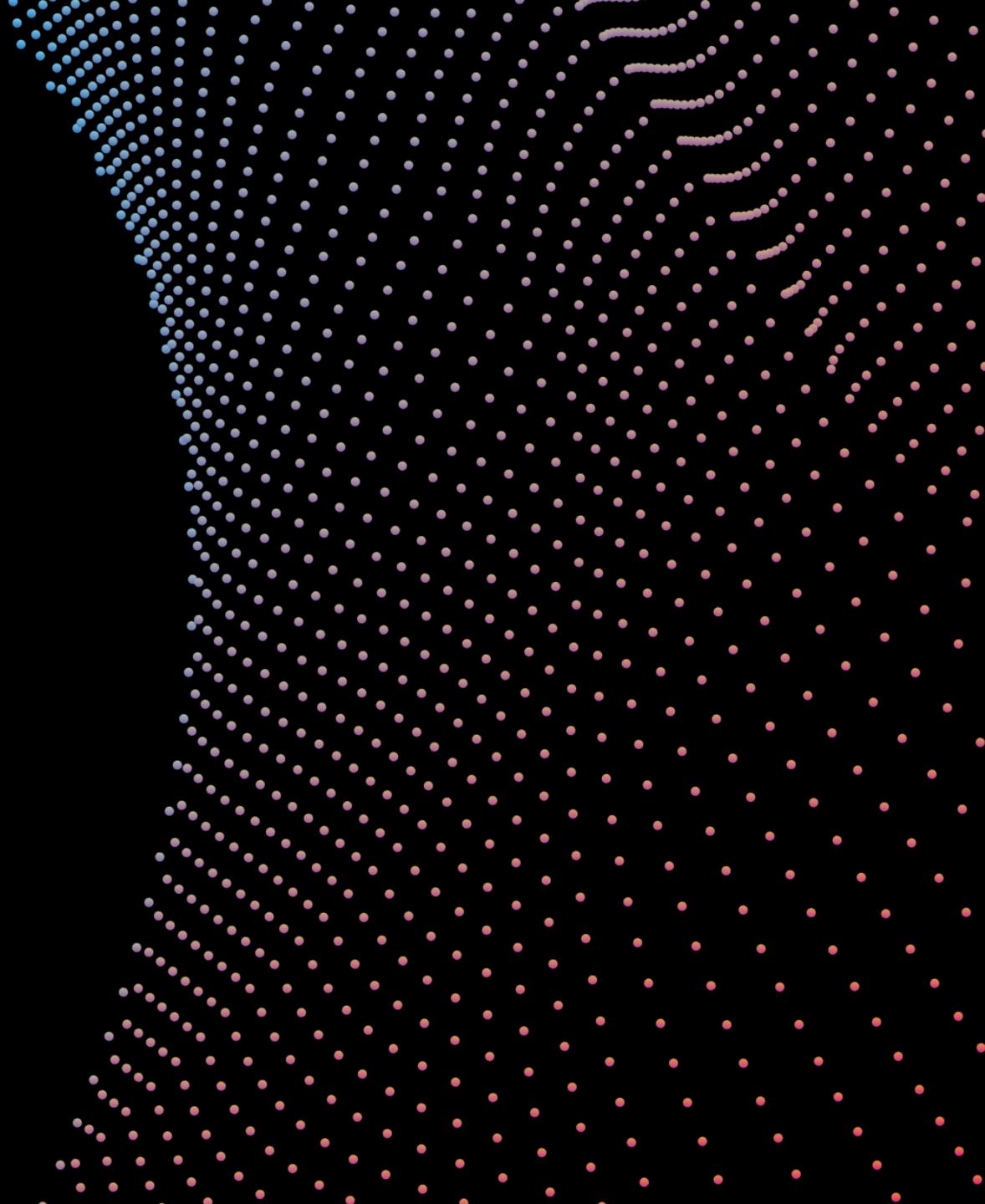
Azure Synapse regional availability

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



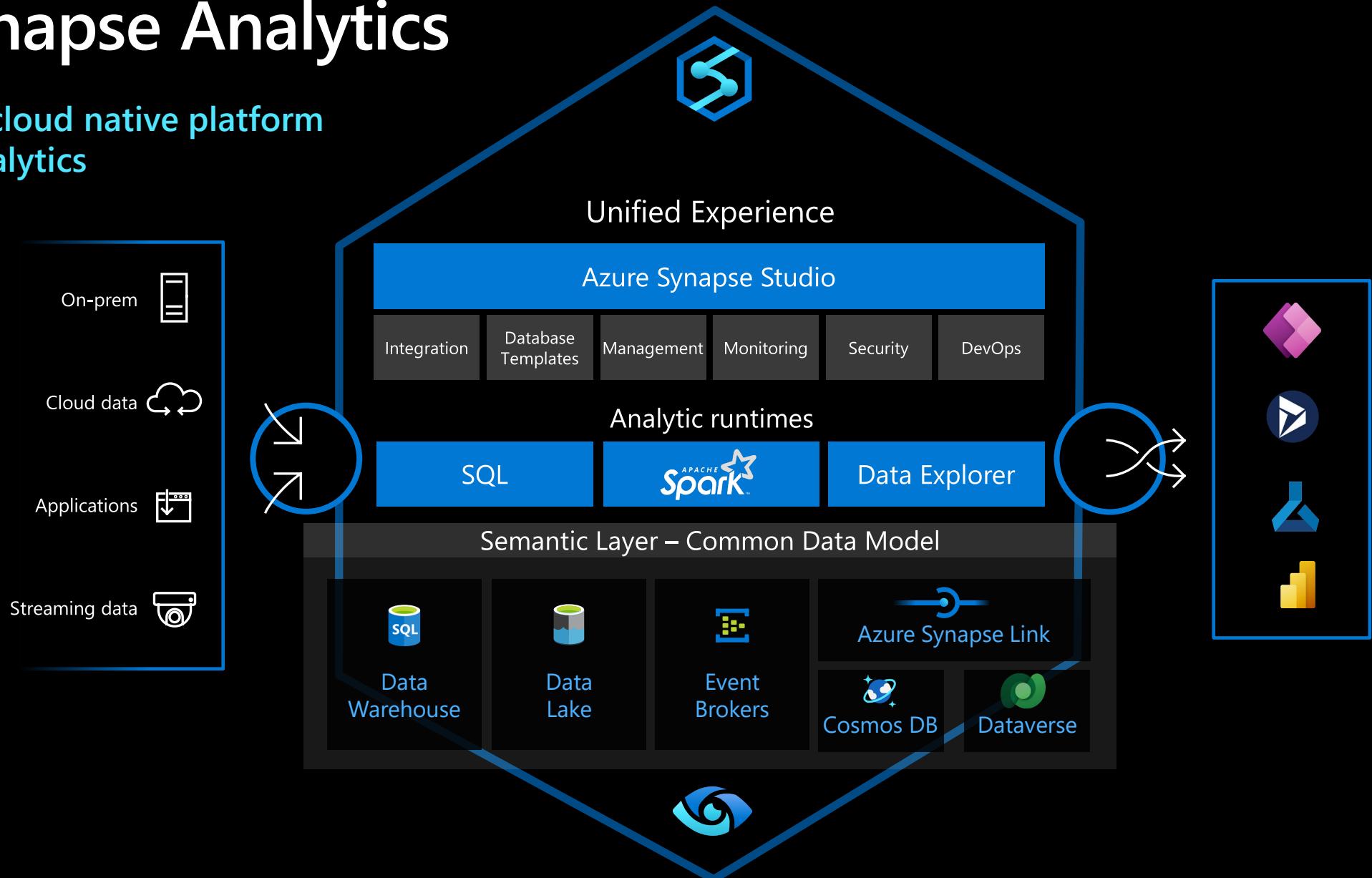
Agenda

- 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

Azure Data Factory and Azure Synapse Capabilities

Category	Feature	Azure Data Factory	Azure Synapse Analytics
Integration Runtime	Using SSIS and SSIS Integration Runtime	✓	✓ Public preview
	Support for Cross-region Integration Runtime (Data Flows)	✓	X
	Integration Runtime Sharing	✓ Can be shared across different data factories	X
Pipelines Activities	SSIS Package Activity	✓	✓ Public preview
	Support for Power Query Activity	✓	X
Template Gallery and Knowledge center	Solution Templates	✓ Azure Data Factory Template Gallery	✓ Synapse Workspace Knowledge center
GIT Repository Integration	GIT Integration	✓	✓
Monitoring	Monitoring of Spark Jobs for Data Flow	X	✓ Leverage the Synapse Spark pools

Linked services

Overview

It defines the connection information needed to connect to external resources.

Benefits

Offers pre-build 100+ 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 connection information needed to connect to external resources.

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	

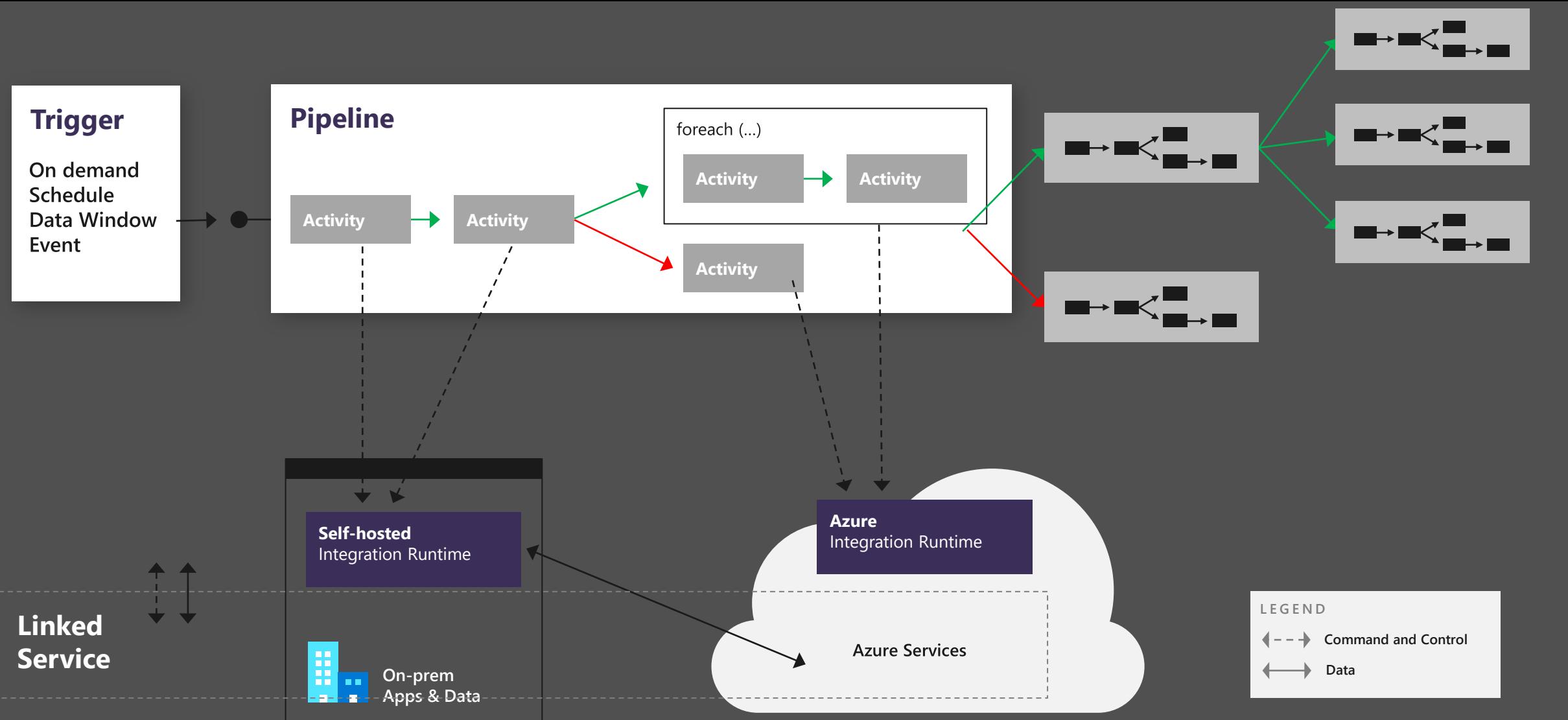
Integration dataset

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. On the left, the 'Linked' tab is selected under the 'Workspace' section. A search bar labeled 'Filter resources by name' is present. The main list displays various Azure services and integration datasets. Under 'Integration datasets', the 'asal400_december_sales' dataset is selected, highlighted with a grey background. To the right, the details for this dataset are shown in a larger pane. The dataset is named 'asal400_december_sales' and is of type 'Parquet'. Below this, the 'Connection' tab is selected, showing the 'Linked service' dropdown set to 'asadatalake01', the 'Integration runtime' dropdown set to 'LargeAzureComputeOptimizedIntegr...', and the 'File path' input field containing 'wwi-02 / campaign-analytics / large-sale-december20'. Other tabs like 'Schema' and 'Parameters' are also visible.

Components of orchestration

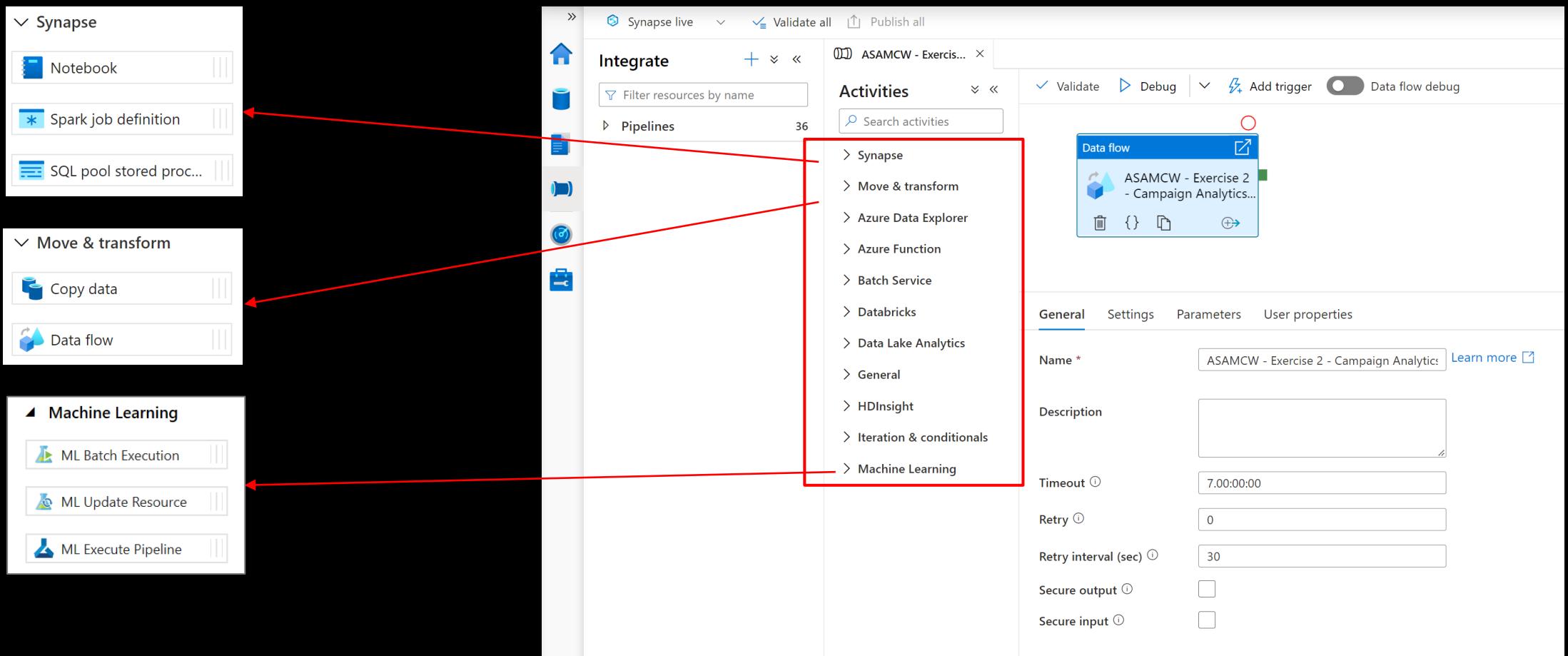


Synapse Pipelines shares codebase with Azure Data Factory

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

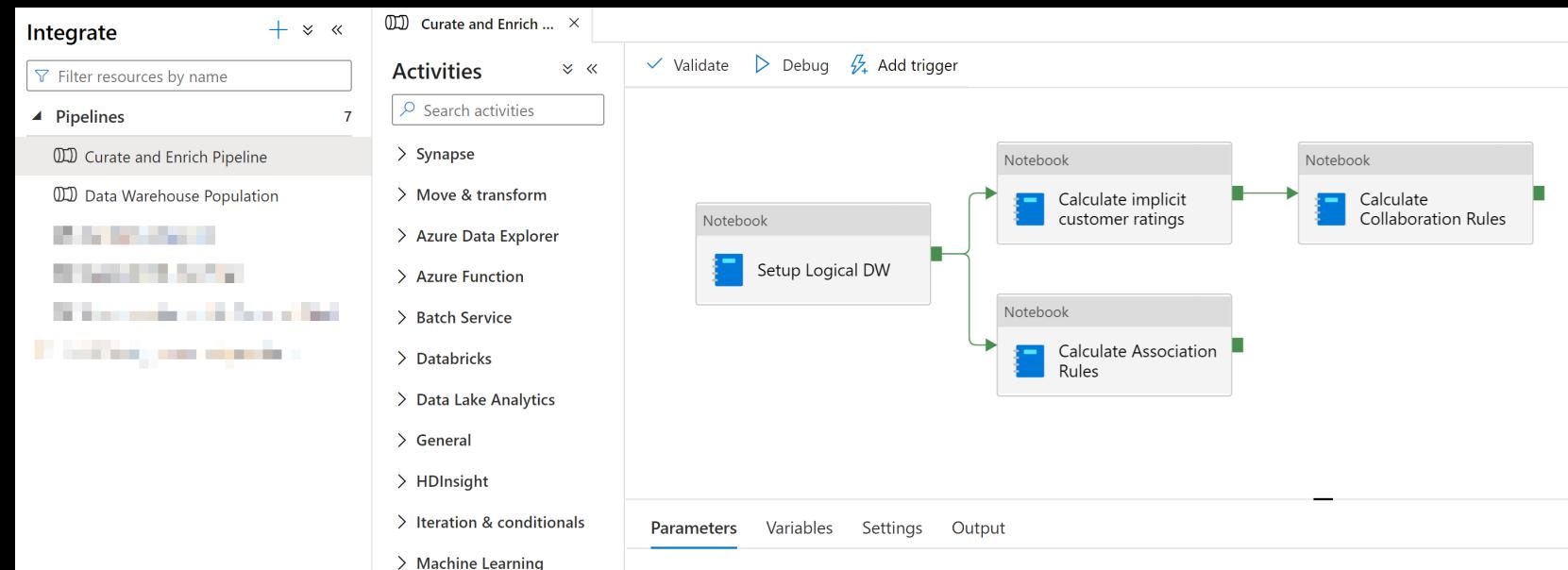
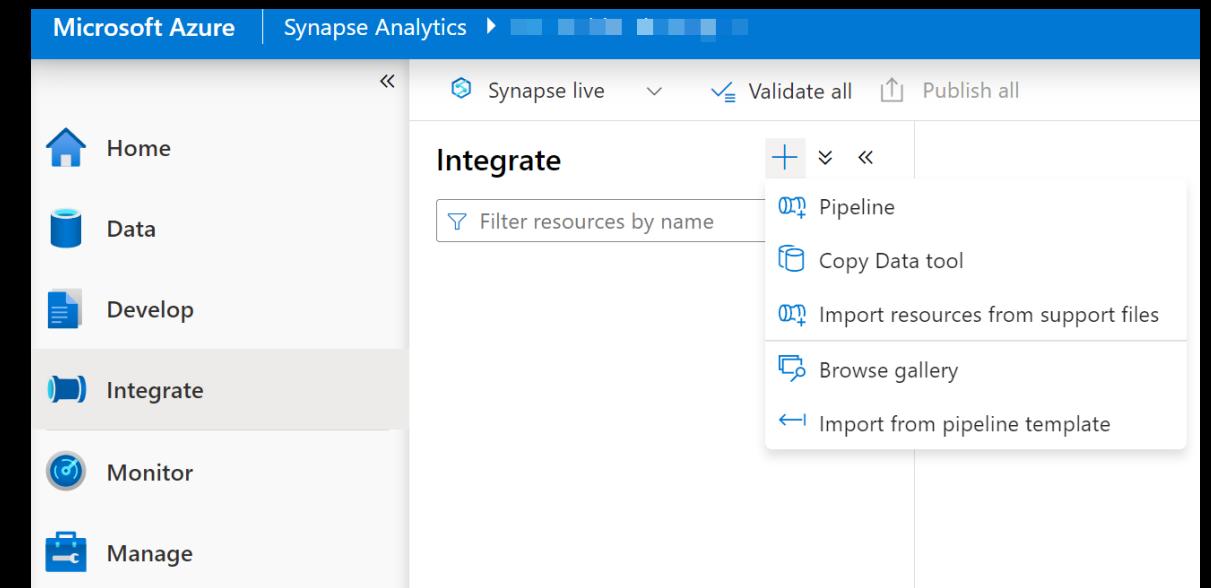
Overview

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

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

Benefits

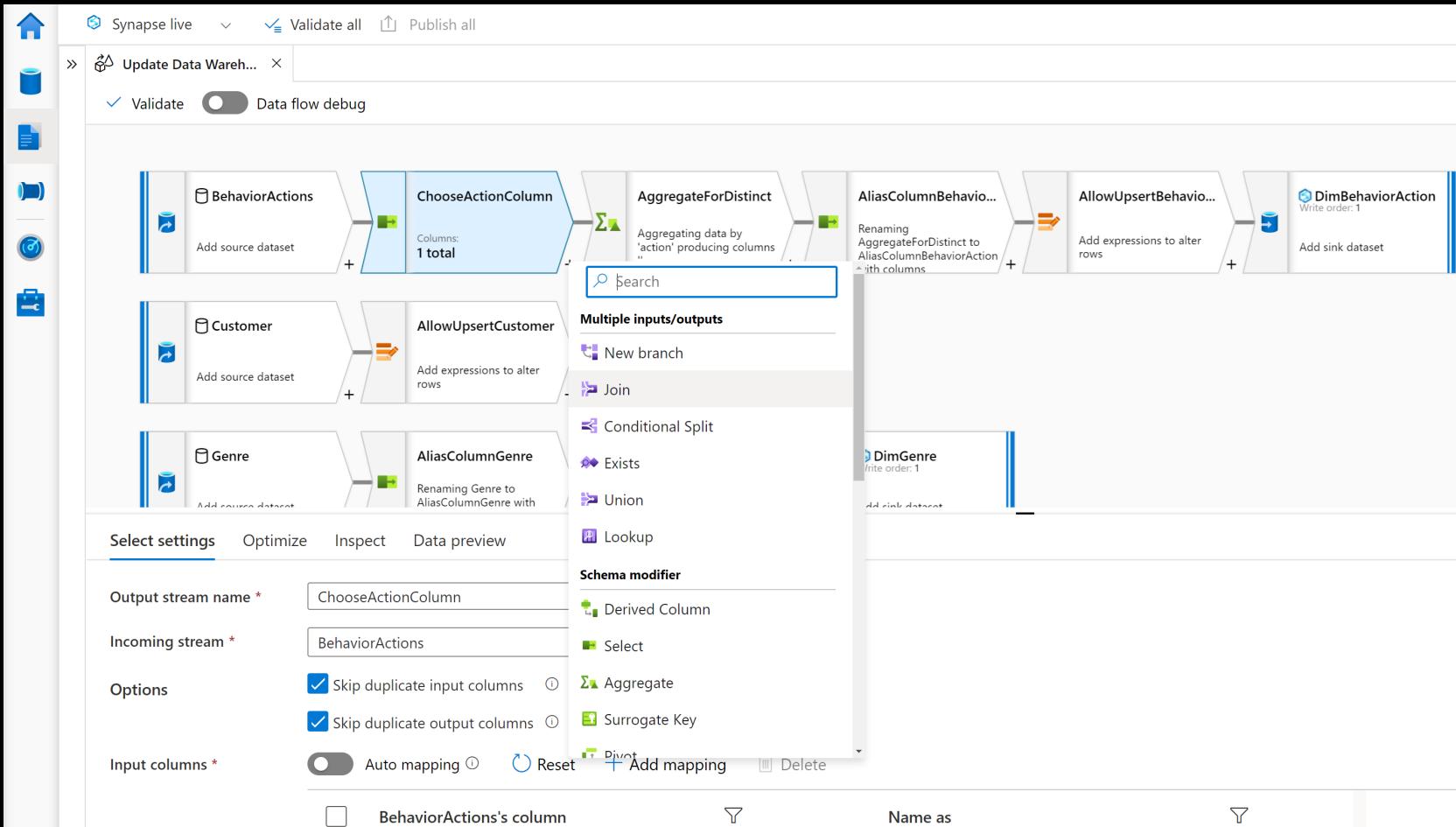
- Supports common loading patterns.
- Fully parallel loading into data lake or SQL tables.
- Graphical development experience.



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



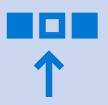
Add file handling (move files after read, write files to file names described in rows etc)



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 is a 'New' button, also highlighted with a red box. To the right of the 'New' button is a detailed configuration panel for a new trigger named 'Trigger 1'. The configuration 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. There's also a checkbox for 'Specify an end date' which is unchecked. At the bottom of the configuration panel, there's a 'Start trigger' section with a checkbox for 'Start trigger on creation' which is also unchecked. The status bar at the bottom right shows two triggers: 'Blob-storage-trigger-01' (Storage events, Status: Stopped) and 'cdm-trigger' (Storage events, Status: Started).

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.

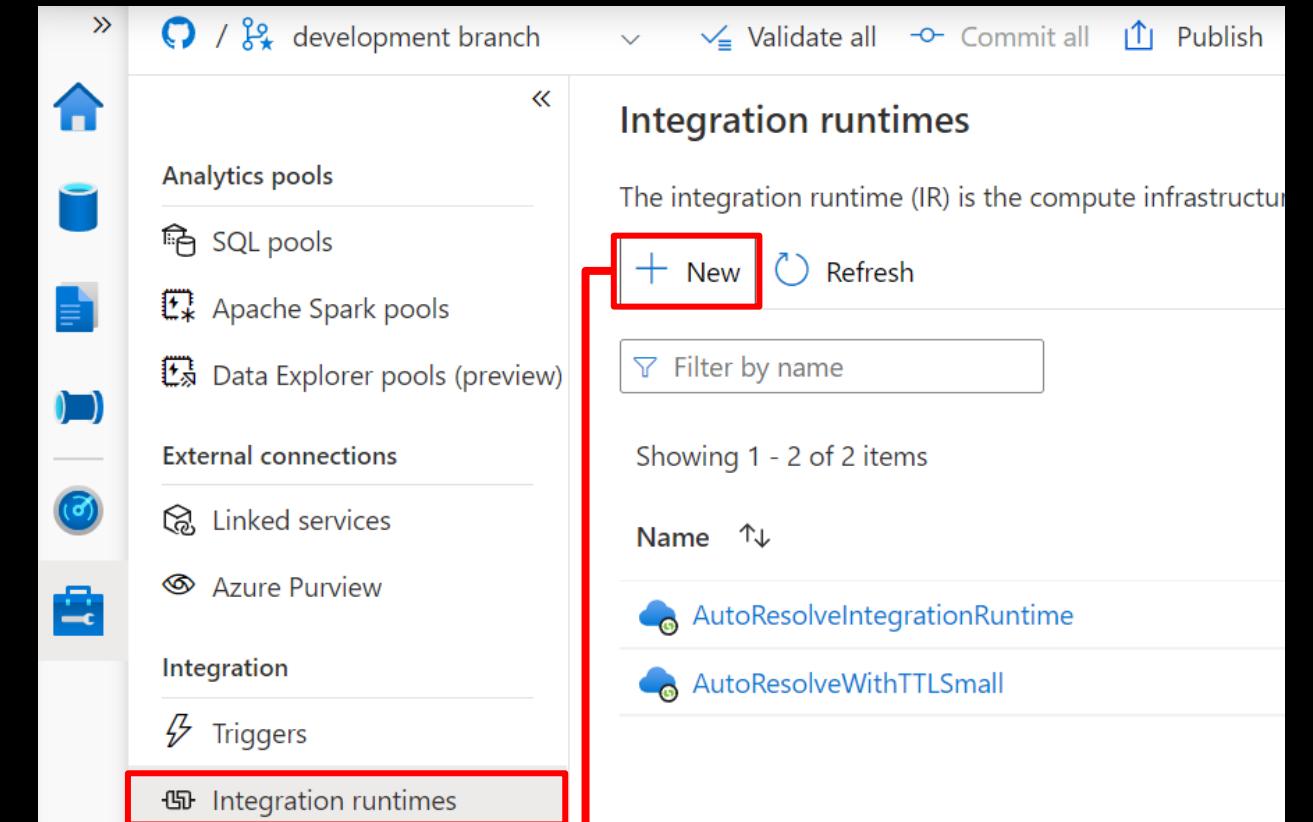
Benefits

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 Data Factory interface. At the top right are buttons for 'Validate all', 'Commit all', and 'Publish'. Below them is a title 'Integration runtimes' with a description: 'The integration runtime (IR) is 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.' A red box highlights the 'New' button. A red arrow points from the 'Integration runtimes' section in the main list down to the 'Integration runtime setup' dialog.

Integration runtimes

The integration runtime (IR) is 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.

New Refresh

Filter by name

Showing 1 - 2 of 2 items

Name ↑

AutoResolveIntegrationRuntime

AutoResolveWithTTLSmall

Integration runtime setup

Integration Runtime is the native compute used to execute or dispatch activities. Choose what integration runtime to create based on required capabilities. [Learn more](#)

Azure, Self-Hosted

Perform data flows, data movement and dispatch activities to external compute.

Azure-SSIS (preview)

Lift-and-shift existing SSIS packages to execute in Azure.

Integration runtime setup

Network environment:

Choose the network environment of the data source / destination or external compute to which the integration runtime will connect to for data flows, data movement or dispatch activities:

Azure

Use this for running data flows, data movement, external and pipeline activities in a fully managed, serverless compute in Azure.

Self-Hosted

Use this for running activities in an on-premises / private network

[View more](#)

Data movement with integration runtimes

Scalable

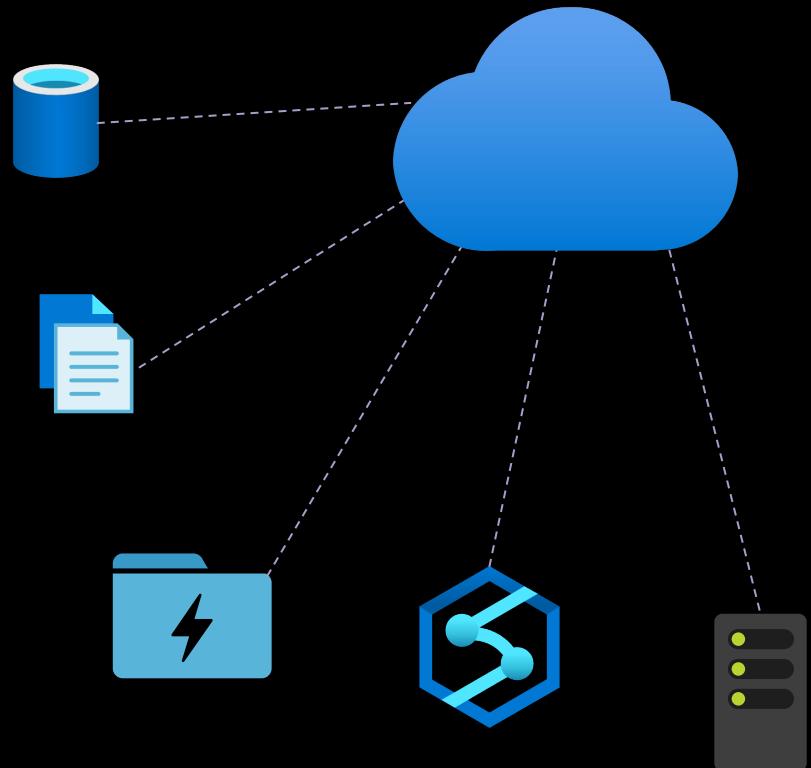
- per job elasticity
- Up to 4 GB/s

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

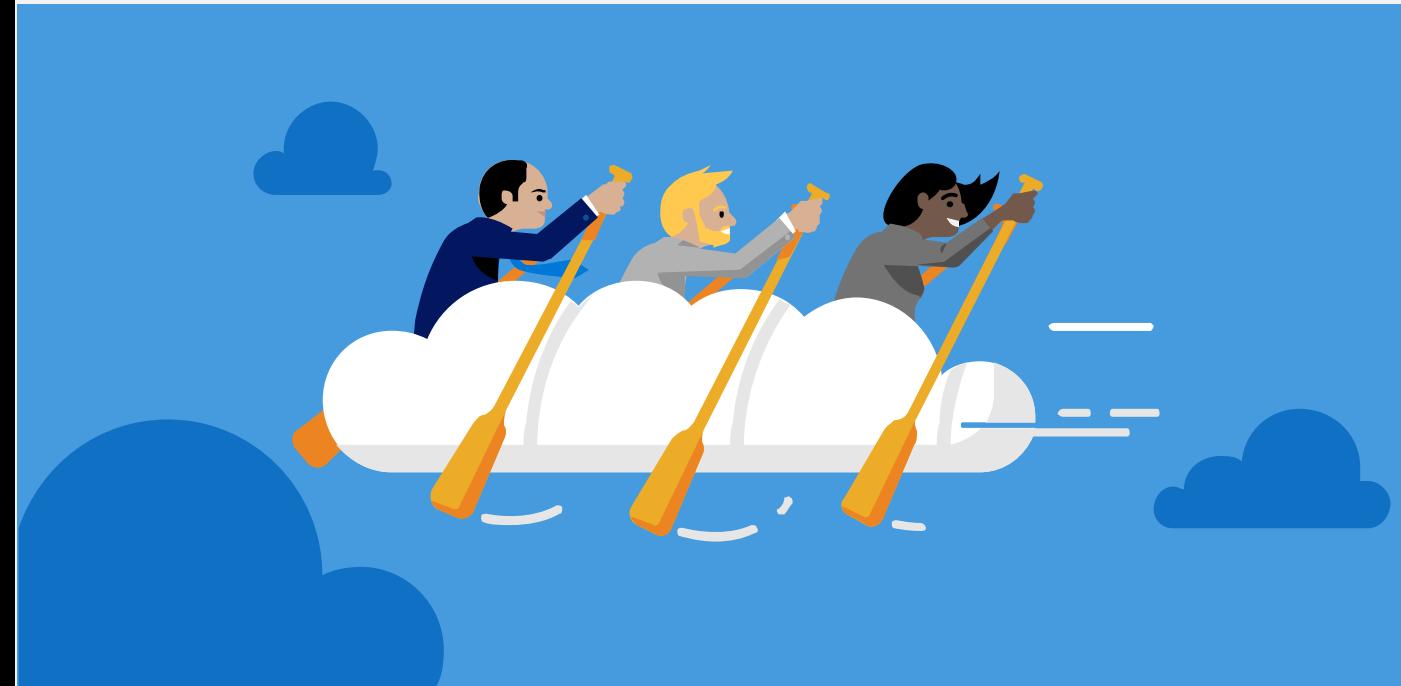
Which one of these is NOT a component of a Synapse pipeline?

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 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

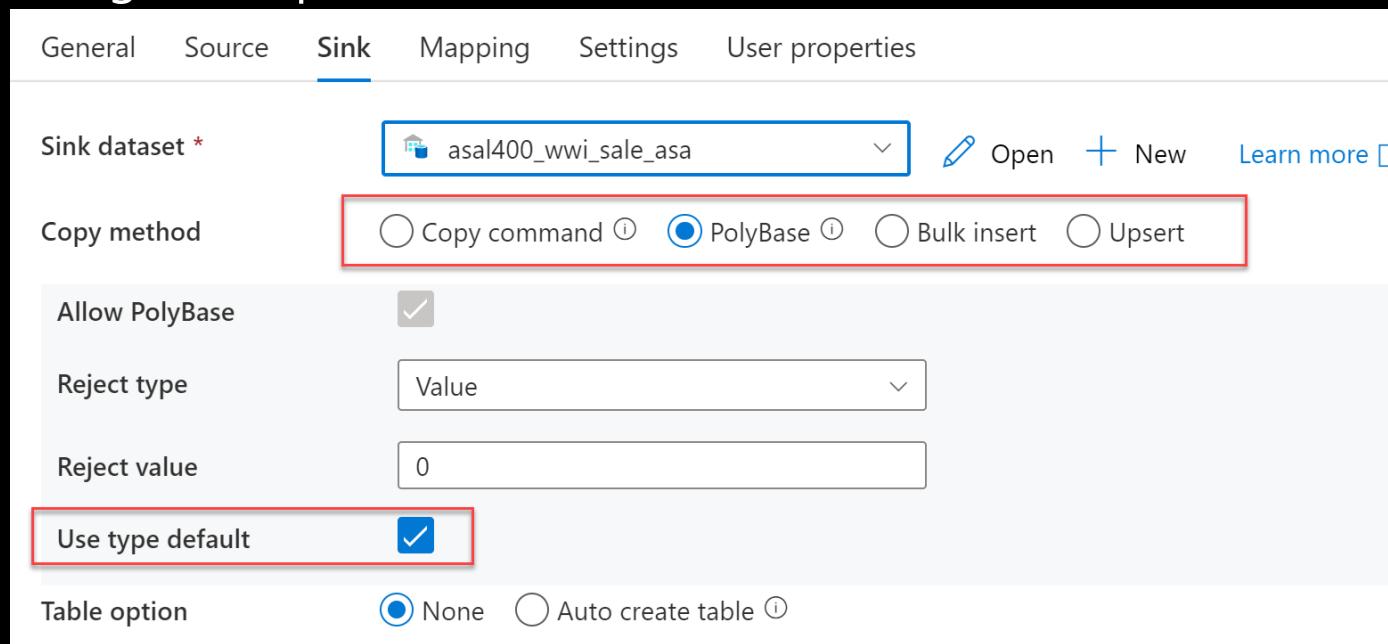
- Network unreliability, no retries
- Needs VM in cloud, performance dependent on VM configuration
- Doesn't support ADLS
- Reduces concurrency
- Control-gated performance limitation, can not scale with DWU

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 – DWU3000c

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.

When leveraging CETAS using the PARQUET file type, statistics will be automatically created when the first query targets this external table, resulting in improved performance.

Recommendations and limitations

- Polybase can't load rows that have more than 1,000,000 bytes of data, this is regardless of table schema
- For fastest load, use compressed delimited text files, the difference between UTF8 and UTF16 performance is minimal
- Split large compressed files into smaller compressed files
- When splitting delimited uncompressed text files the guidance is that as long as the file size is over 2 GB and count of file is not reaching high 100s(like 1000) it doesn't really matter if the file size is 50 GB, 100 GB, etc.
- If you have multiple files in a folder, create external table at the folder level there is no need to do this at individual file level, Polybase automatically has parallelization built-in.

Ingest Azure Synapse Data Explorer (Preview)

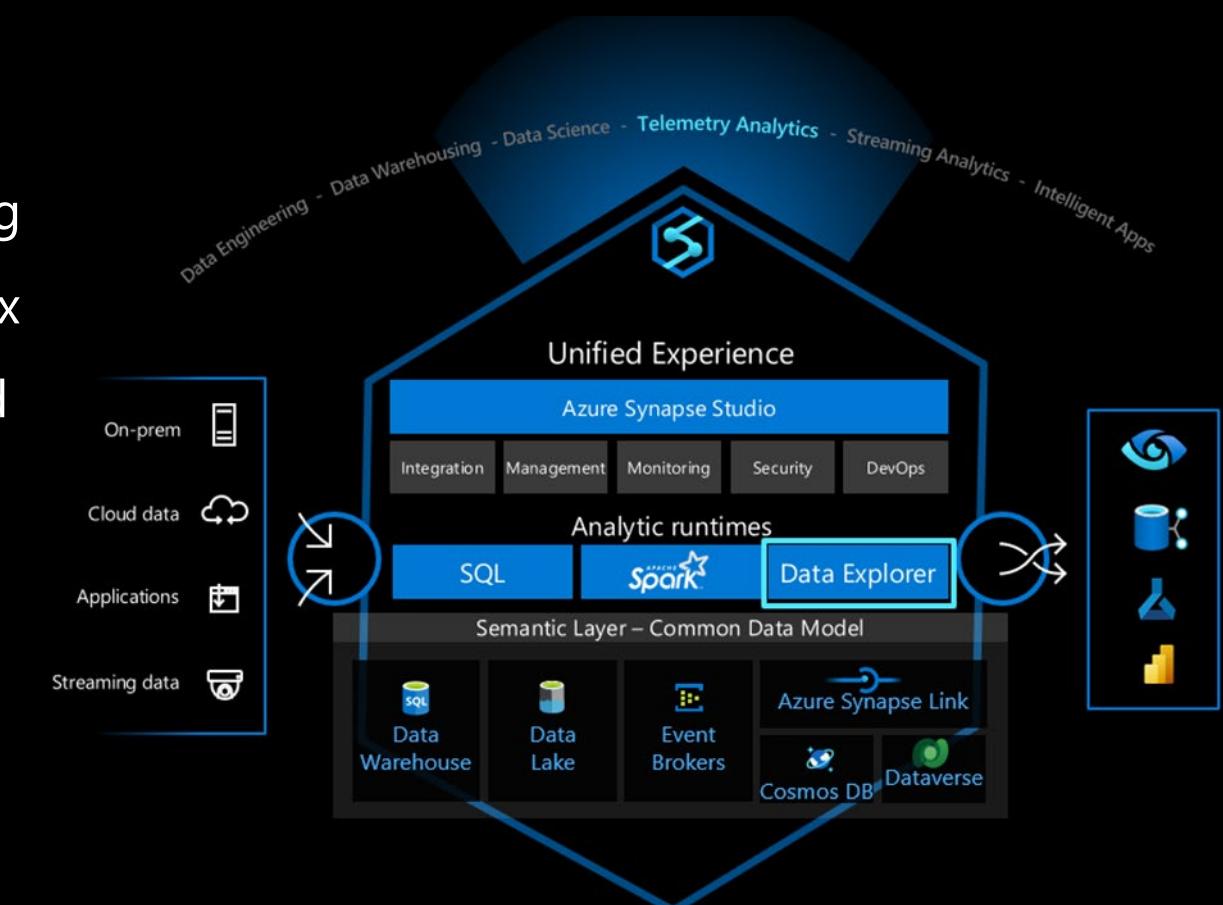
Synapse Data Explorer

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.

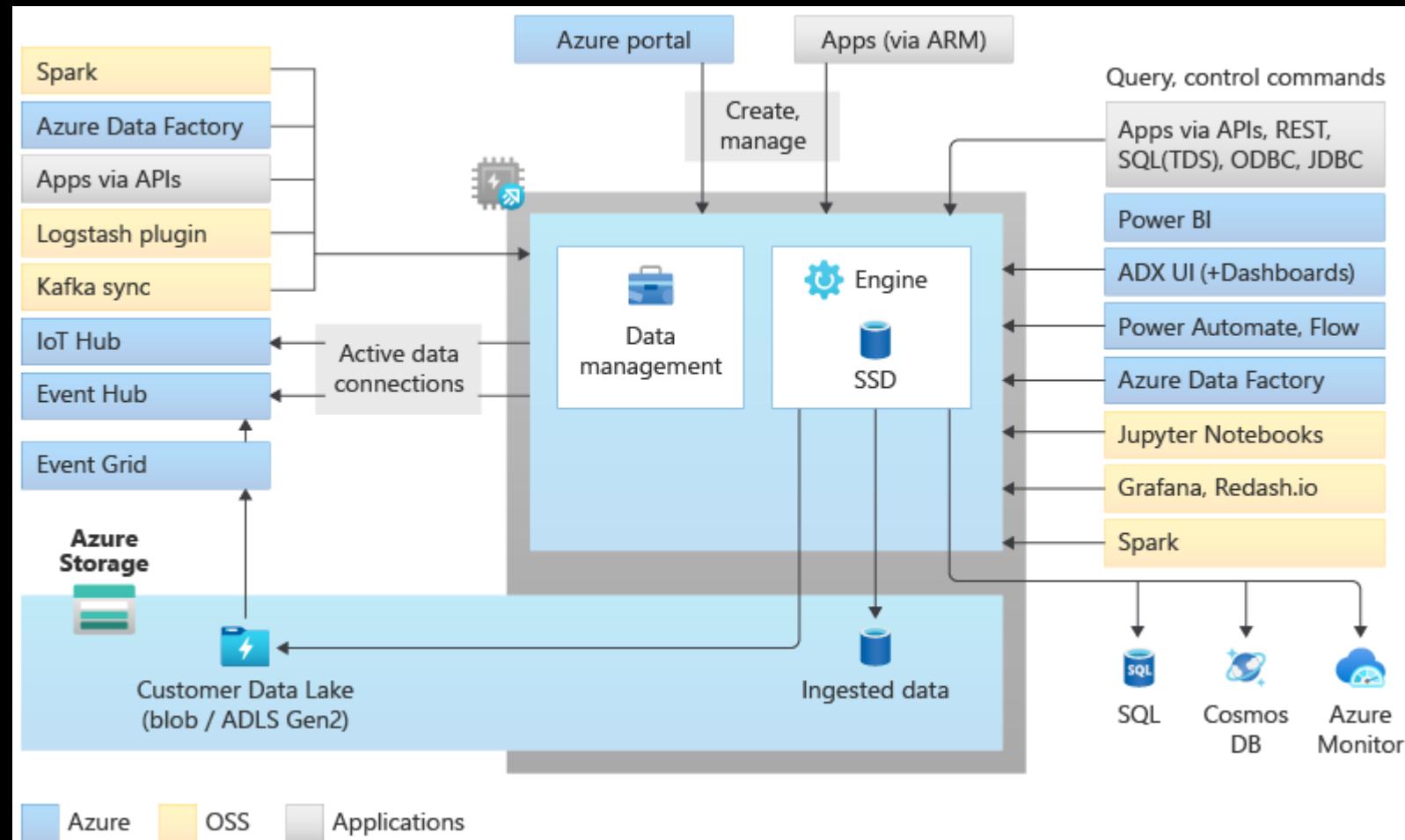
Benefits

- 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



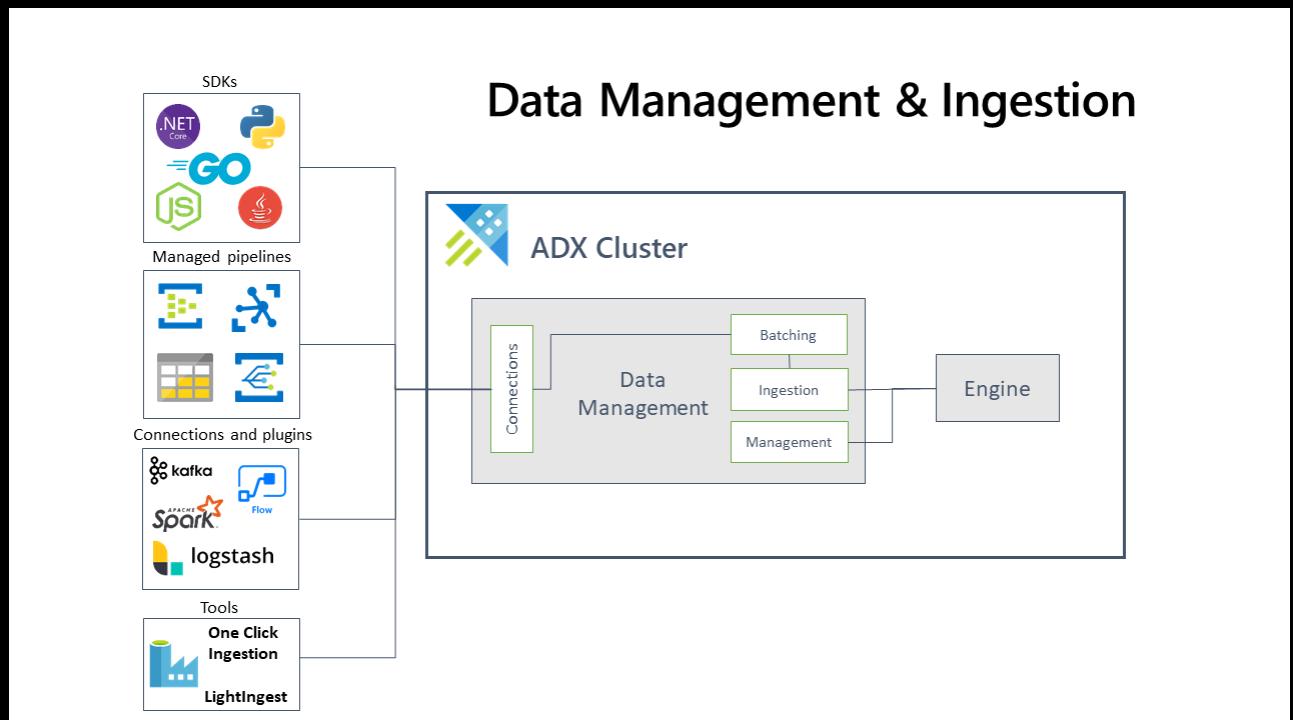
Data ingestion overview

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

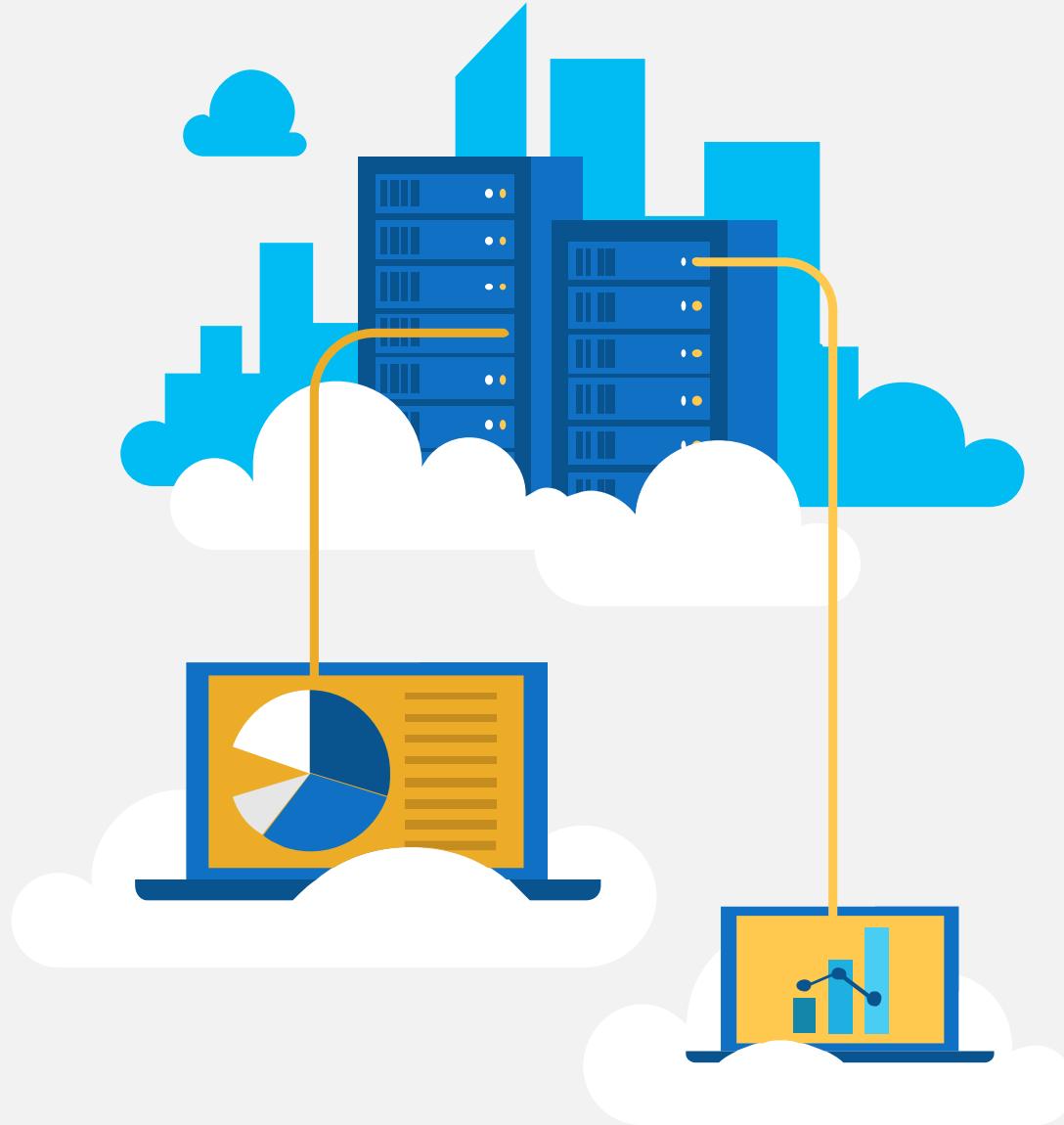


Pop Quiz 2

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

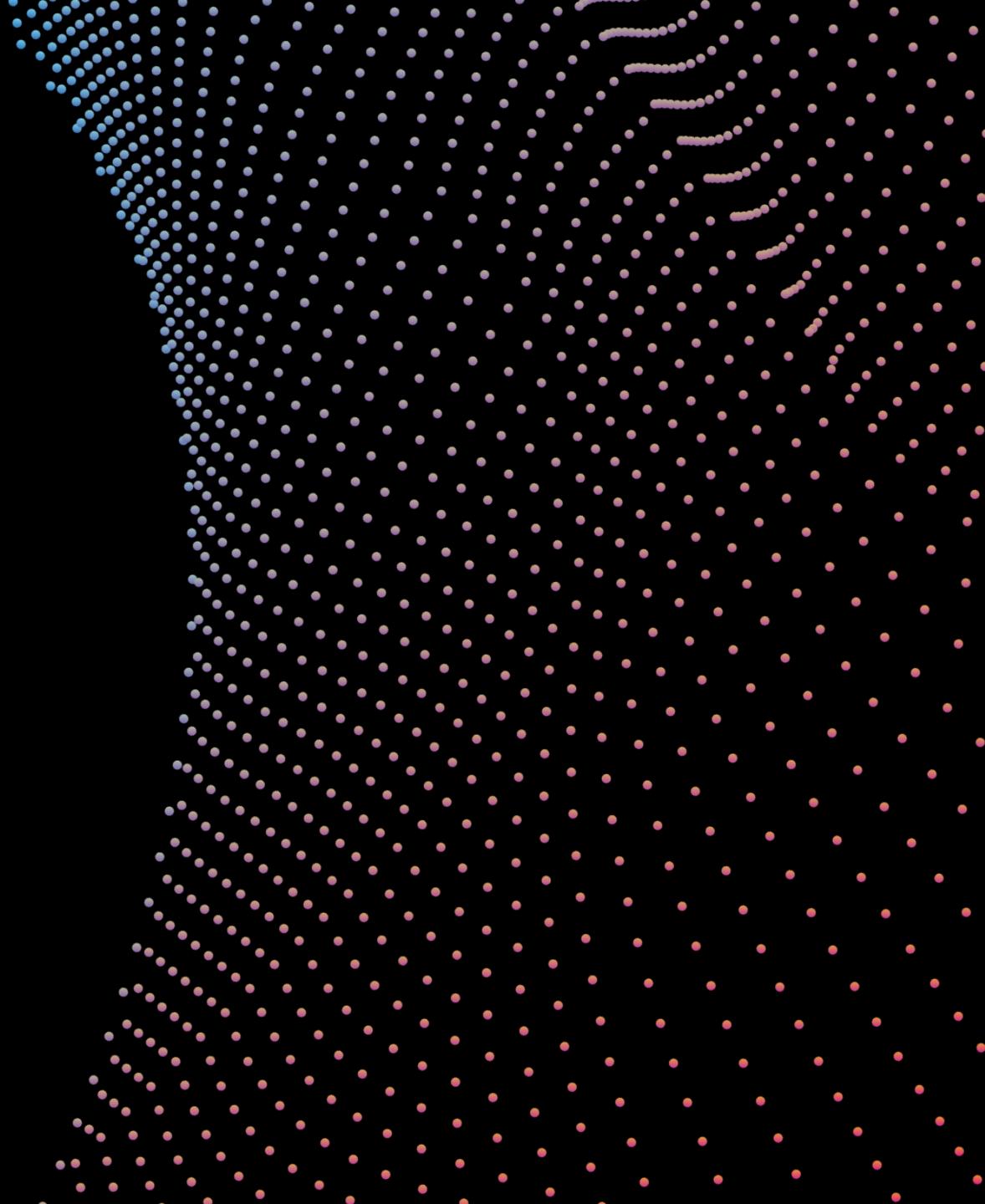
FALSE





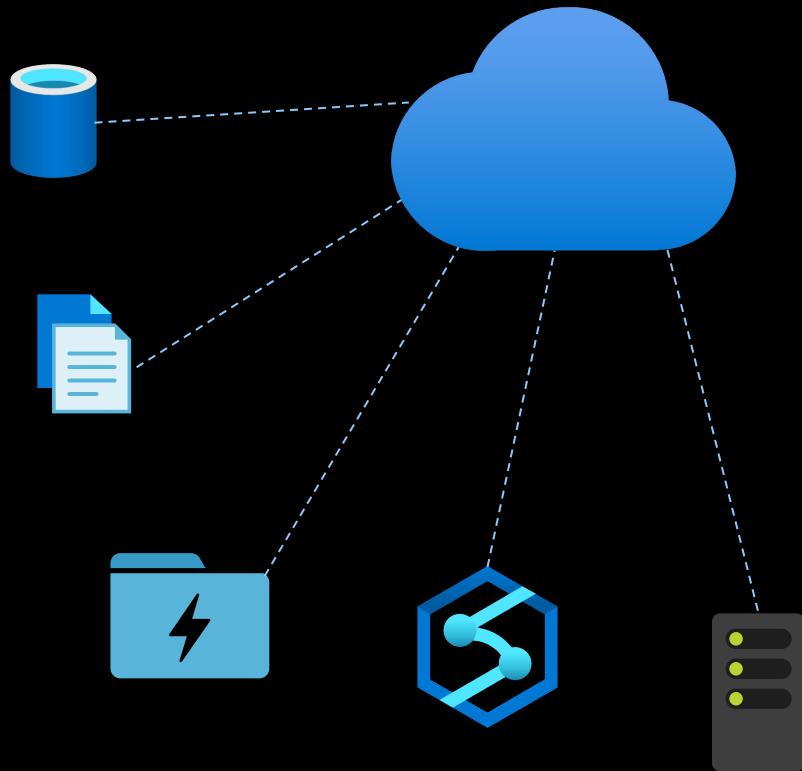
Thank you

Data Transformations



Agenda

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



Typical data transformations

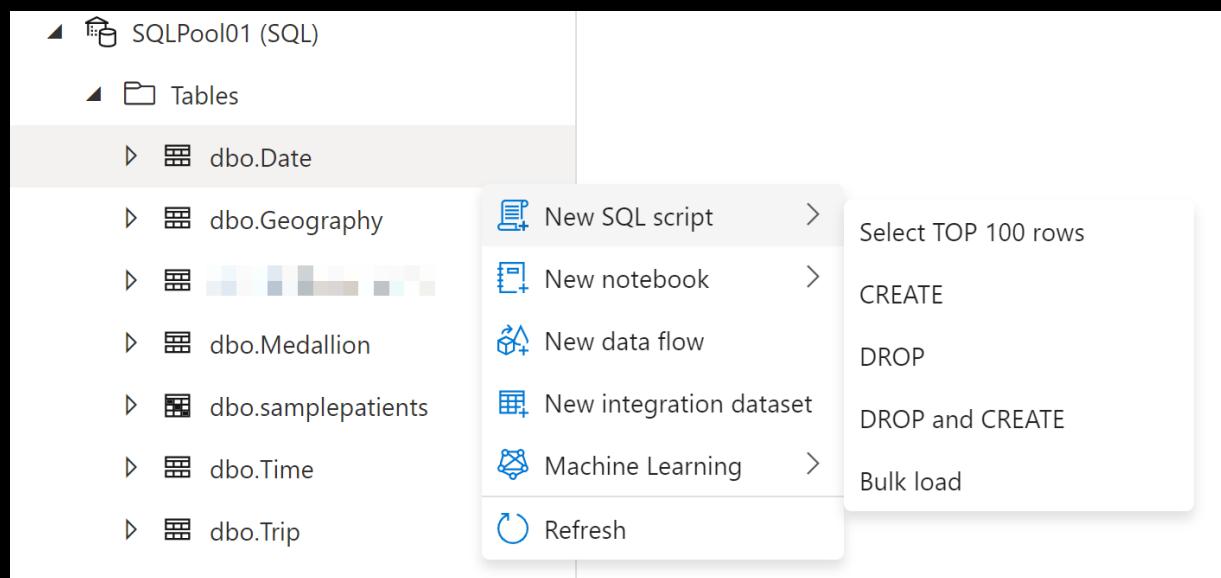
- Create persistent staging area / data vault
- Standardize data from different sources
- Remove duplicate rows
- Impute missing values
- Calculate derived values
- Prepare data for facts and dimensions

Transform

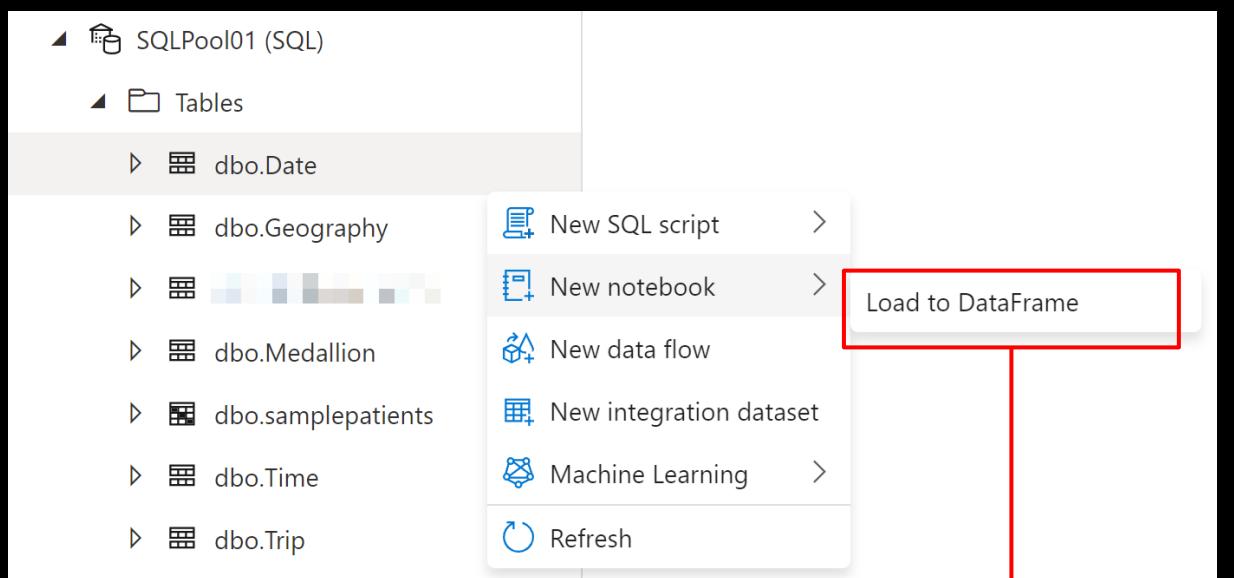
Applying transformations

Code based transformations

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



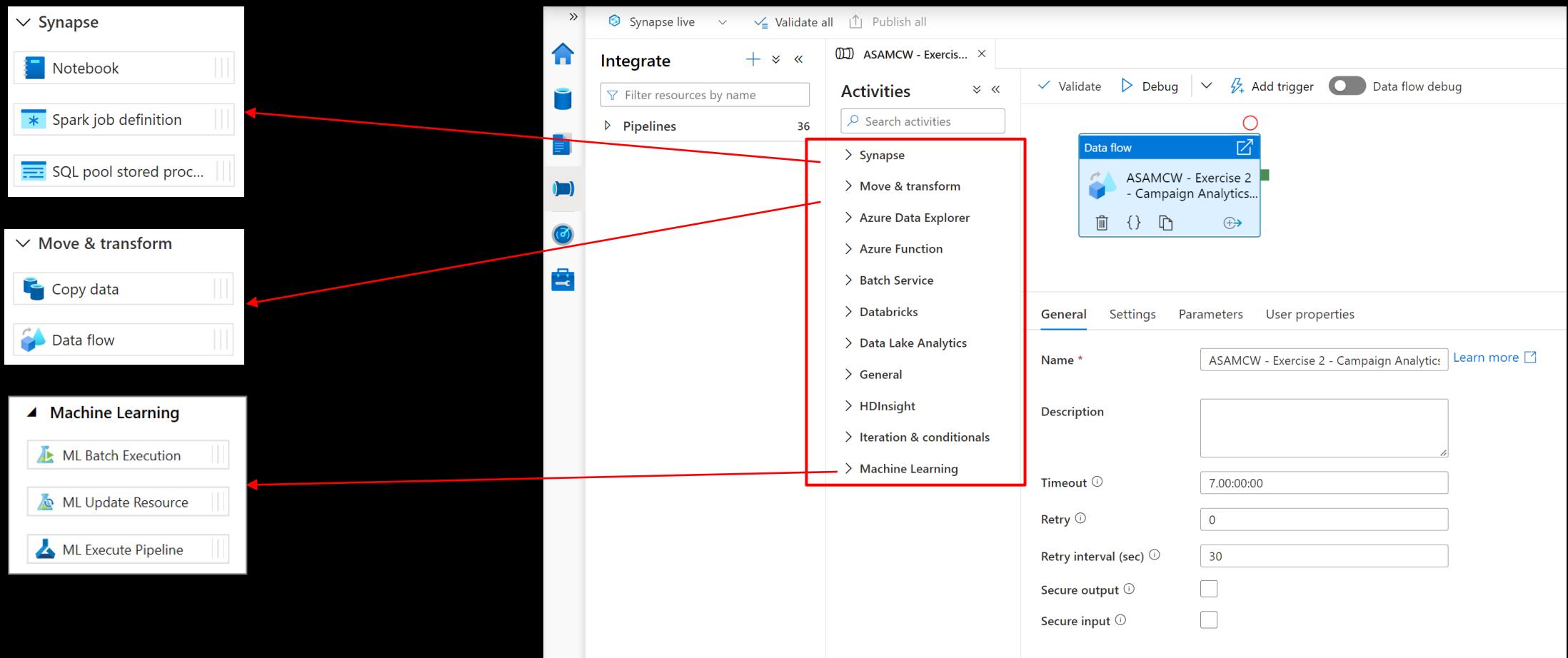
Starting from a table, auto-generate a single line of PySpark code that makes it easy to load a SQL table into a Spark dataframe and author transforms in a notebook.



A screenshot of a Jupyter Notebook cell titled 'Notebook 3'. The 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 cell has a status bar indicating '[]' and a note 'Press shift + enter to execute cells'. At the bottom of the screen, there are buttons for '+ Code' and '+ Markdown'.

Transform with pipelines

Orchestrate transformations with Synapse Pipelines.



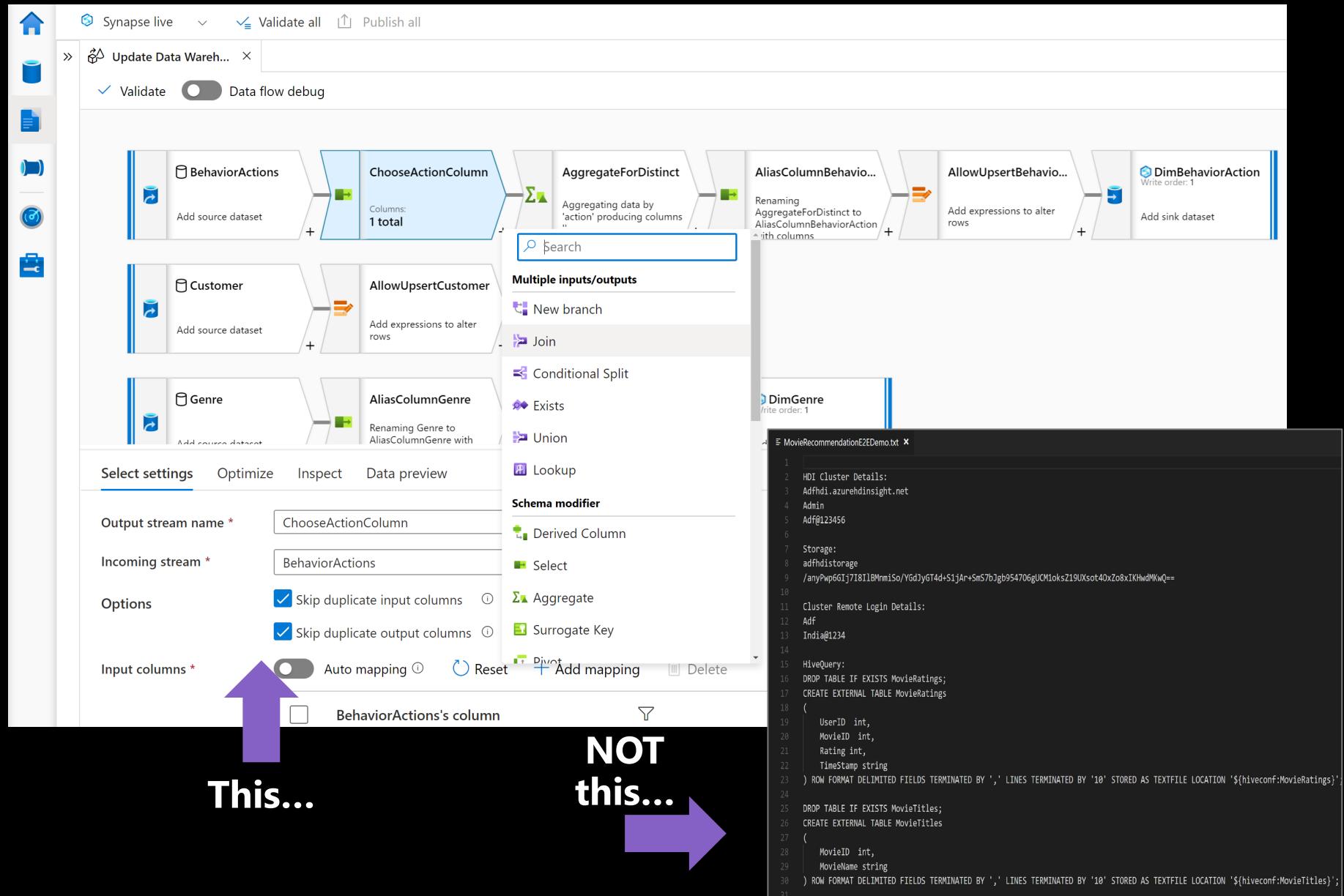
No code transform with data flows

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

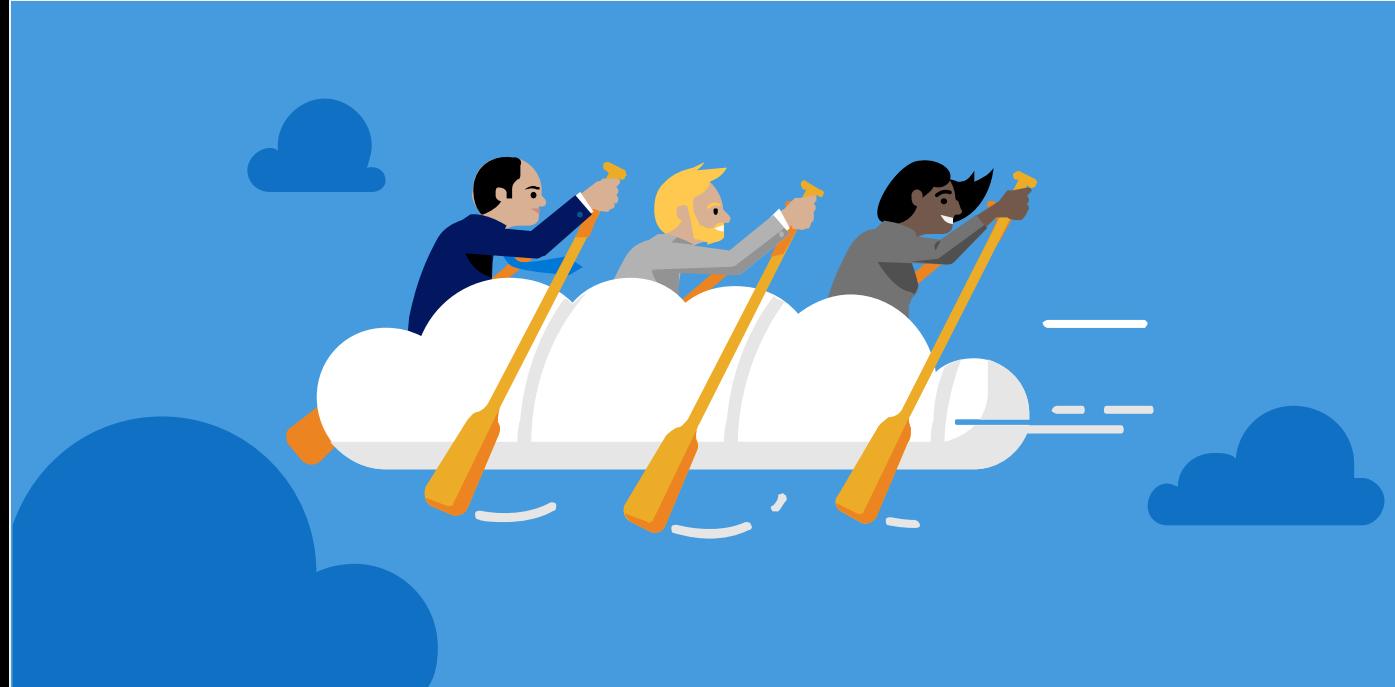
Pop Quiz 1

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

A)
100TB

B)
1PB

C)
10PB



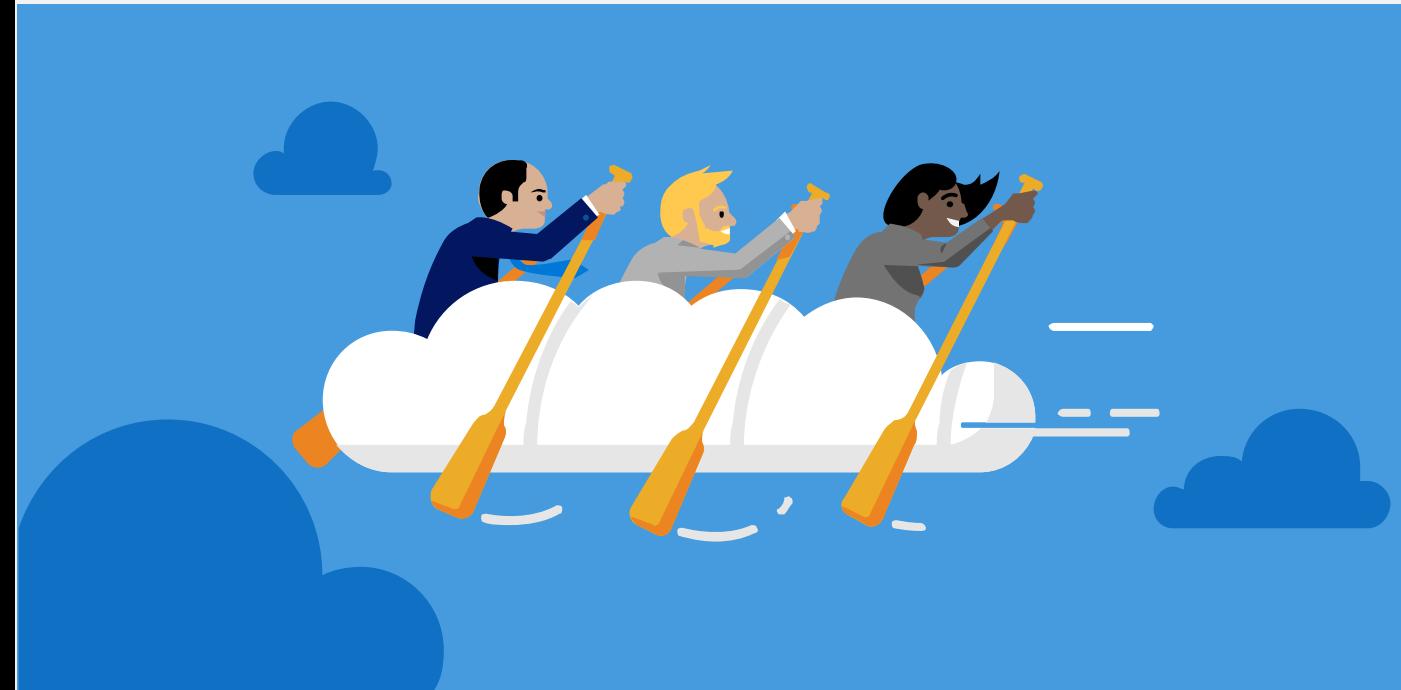
Pop Quiz 1

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

A)
100TB

**B)
1PB**

C)
10PB



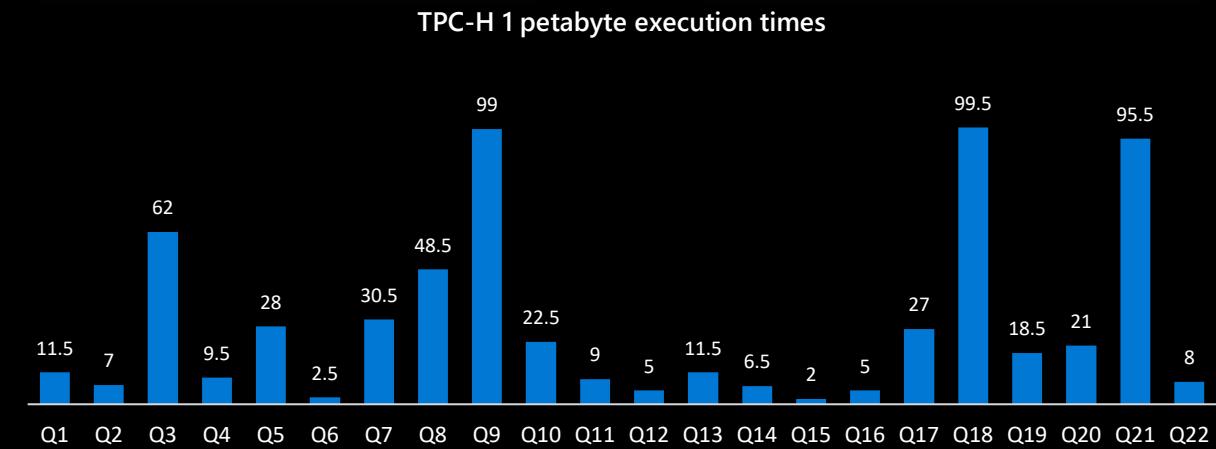
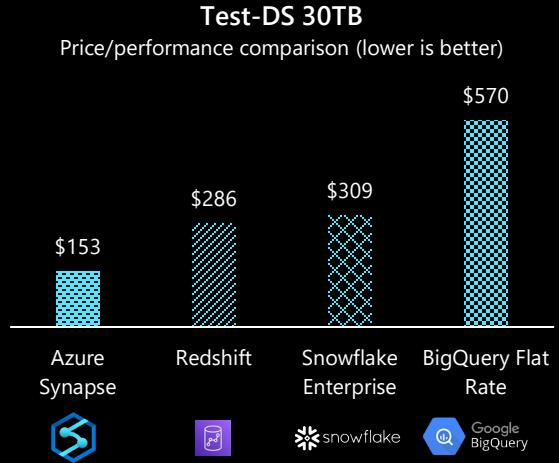
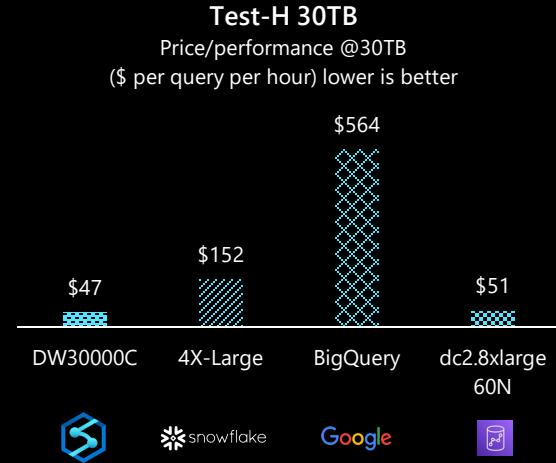
Industry leading SQL performance and scalability

TPC-H and TPC-DS Leader

Price/performance leadership relative to other cloud data warehouses

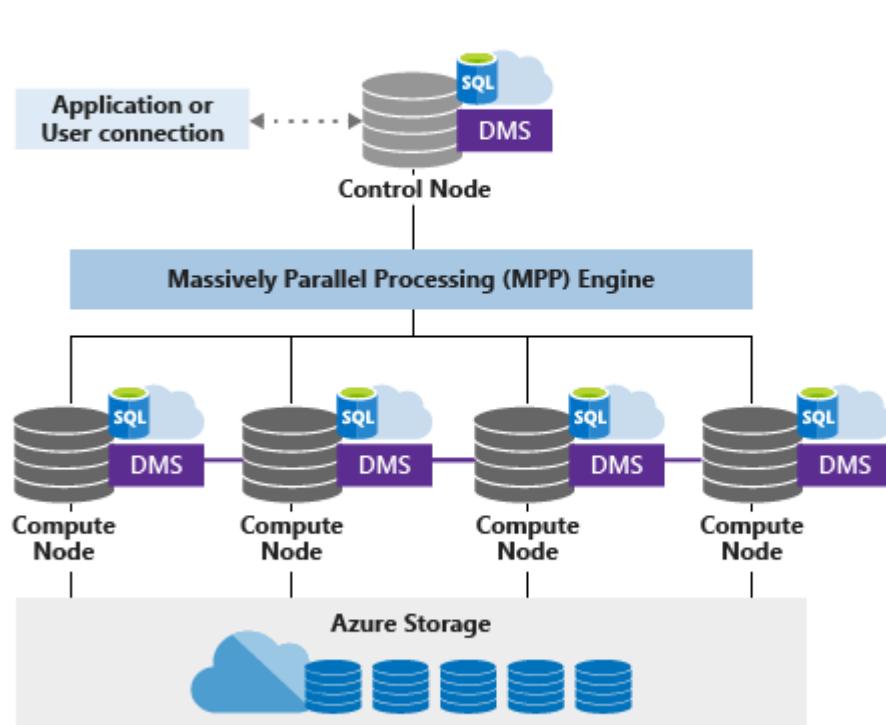
"Polaris" is the only query engine to successfully complete TPC-H at 1PB scale

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

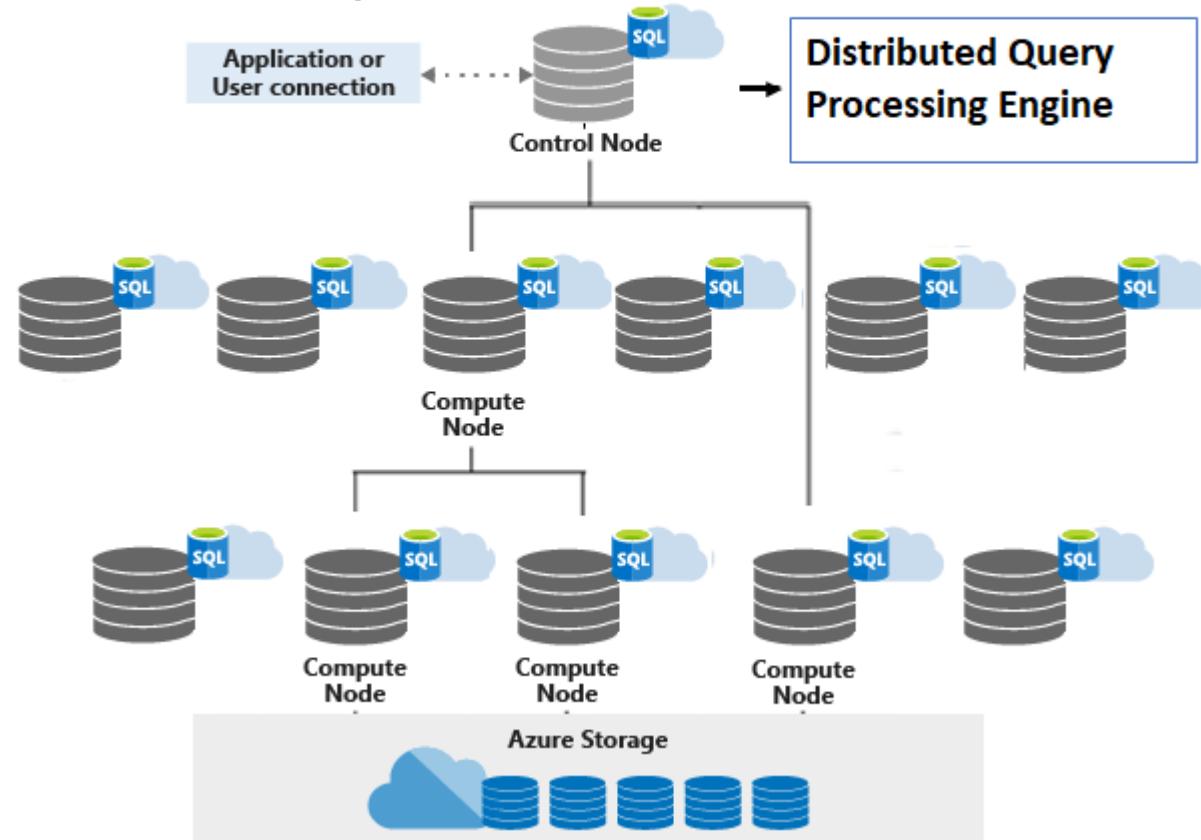


SQL pool infrastructure

Dedicated SQL pool



Serverless SQL pool



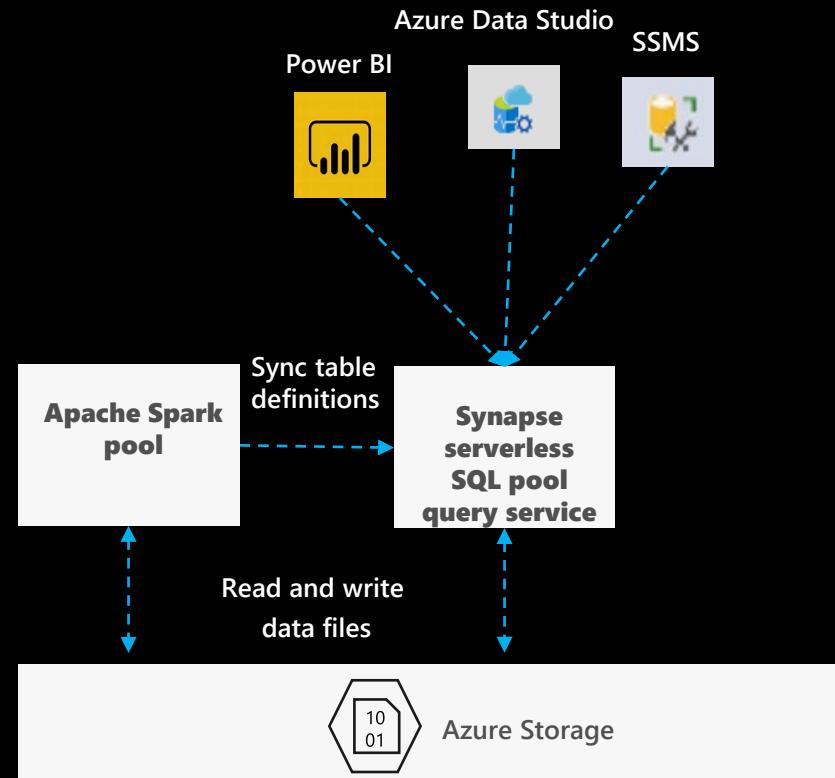
Serverless SQL pool

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

The screenshot illustrates the ease of exploring files stored in Azure Data Lake Storage. On the left, a file named "sale-small-20191201-snappy.parquet" is selected in a list, and a context menu is open. The "New SQL script" option is highlighted with a red box, and the "Select TOP 100 rows" option is also highlighted with a red box. A red arrow points from the "Select TOP 100 rows" option to the right-hand query editor window. The query editor shows an auto-generated T-SQL script for querying the parquet file:

```
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/Month=12/Day=20191201/sale-small-20191201-snappy.parquet',
7         FORMAT = 'PARQUET'
8     ) AS [result]
```

The "Connect to" dropdown in the top bar is set to "Built-in". The results pane displays the first few rows of the query output:

TransactionId	CustomerId	ProductId	Quantity	Price	TotalAmount	TransactionDate	ProfitAmount	Ho
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 the query was 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

OPENROWSET

Overview

OPENROWSET will automatically determine columns of data stored in external files.

Benefits

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

Automatic Schema Inference with Parquet files,
OPENROWSET identifies columns and their types based on underlying file metadata.

Perfect solution for data exploration where schema is unknown.

```
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

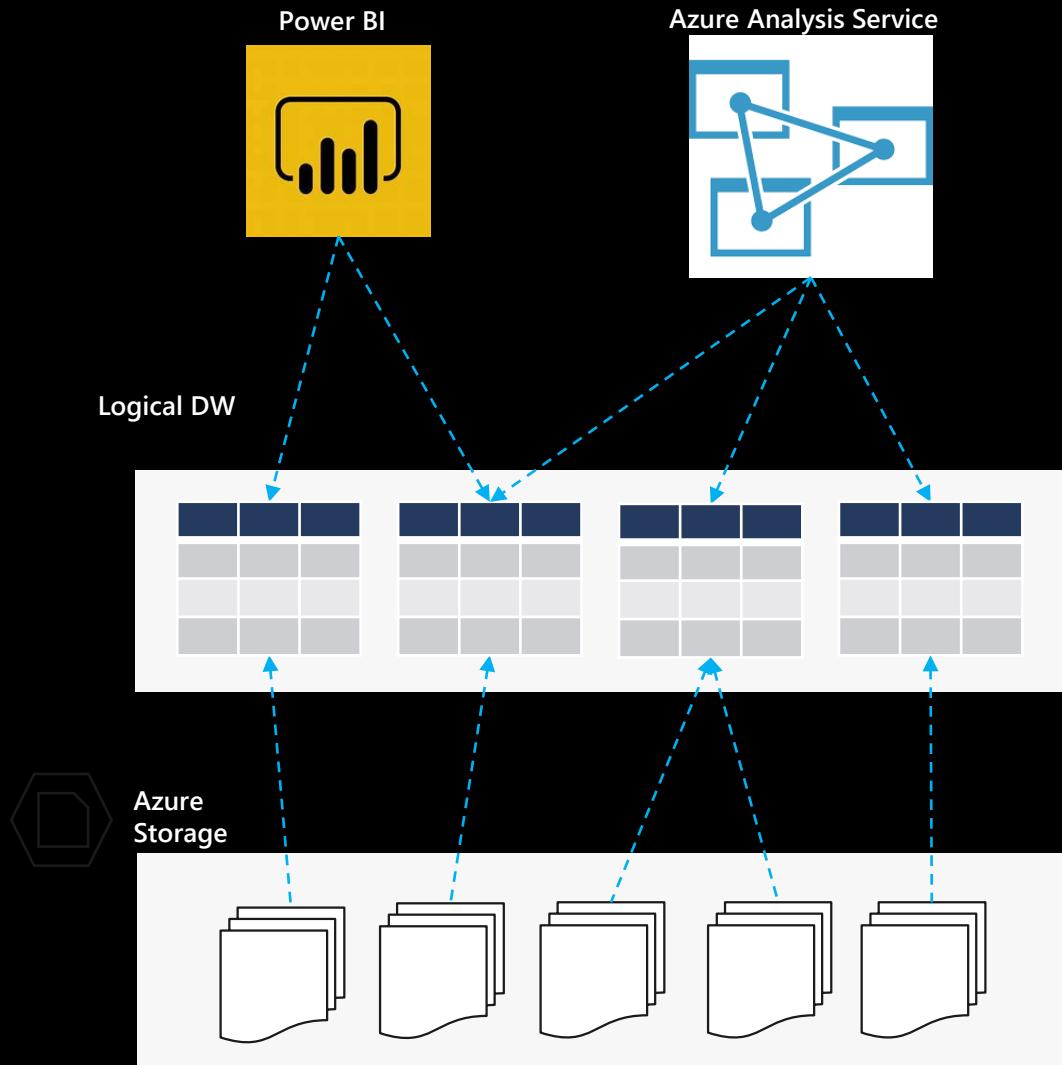
Leverage serverless SQL pool as a logical data warehouse

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

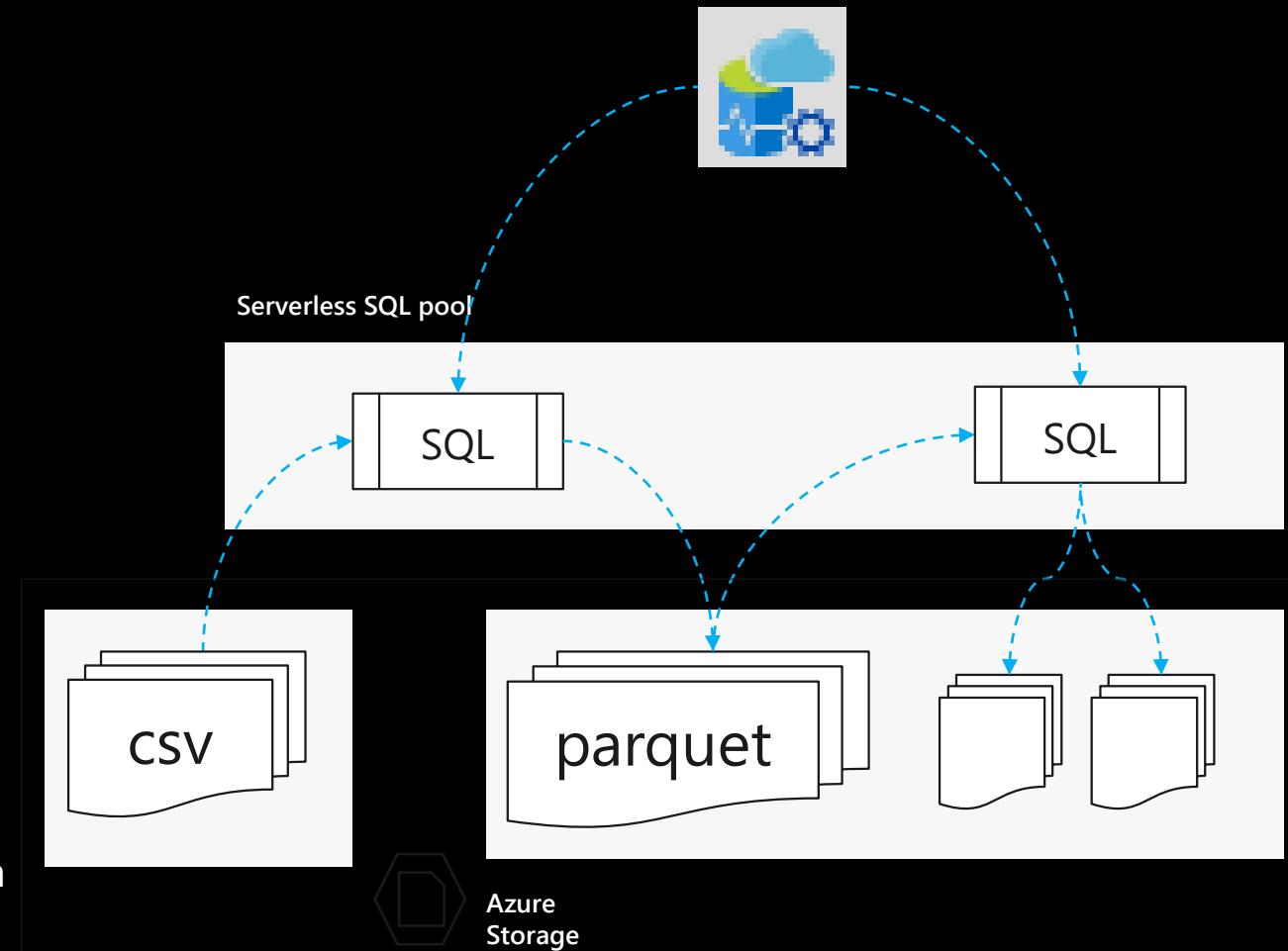
Benefits

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 the desired format (Parquet is recommended)

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
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
```

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

IF NOT EXISTS (SELECT * FROM sys.external_data_sources WHERE name = 'default_azureSynapseA_DFS_Core_Windows_Net')
CREATE EXTERNAL DATA SOURCE [default_azureSynapseA_DFS_Core_Windows_Net]
WITH (
    LOCATION = 'https://azuresynapsesa.dfs.core.windows.net/default',
)
GO

CREATE EXTERNAL TABLE adls.retailsale (
    [storeId] varchar(8000),
    [productCode] varchar(8000),
    [quantity] varchar(8000),
    [logQuantity] varchar(8000),
    [advertising] varchar(8000),
    [price] varchar(8000),
    [weekStarting] varchar(8000),
    [id] varchar(8000)
)
WITH (
    LOCATION = 'Parquet/part-00000-5ae12a71-d27d-4e3a-a686-3bfb7d67c2c9-c000.snappy.parquet',
    DATA_SOURCE = [default_azureSynapseA_DFS_Core_Windows_Net],
    FILE_FORMAT = [SynapseParquetFormat]
)
GO

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

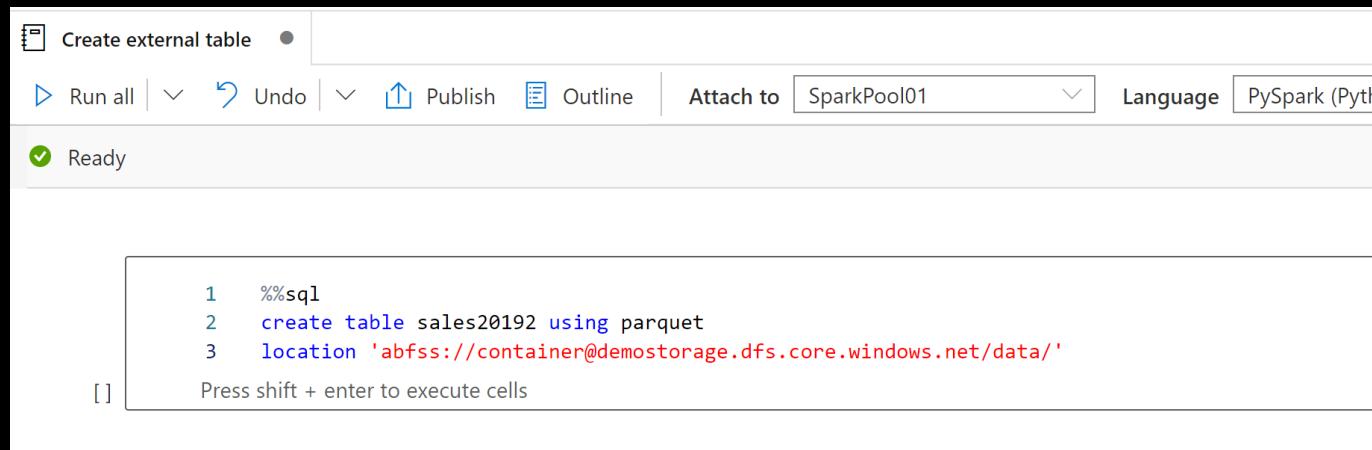
Tables designed using Spark languages are immediately available in serverless SQL pool.

Schema definition matches original

Spark table updates are applied in serverless SQL pool

No need to manually create SQL tables that match Spark tables

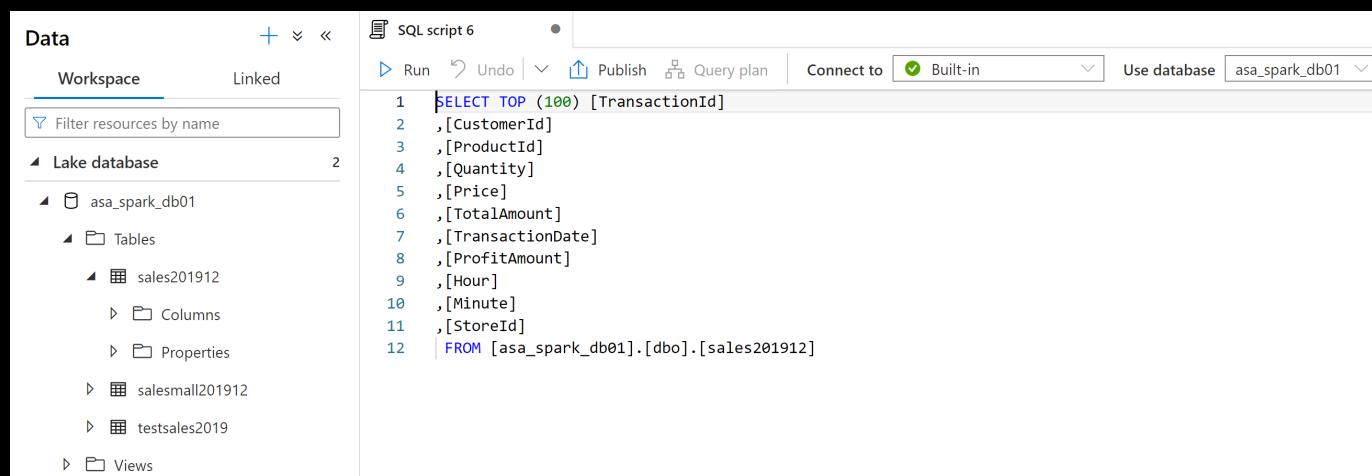
Spark and serverless SQL pool tables reference the same external files.



The screenshot shows the Azure Data Studio interface. At the top, there's a toolbar with 'Create external table' (highlighted), 'Run all', 'Undo', 'Publish', 'Outline', 'Attach to' (set to 'SparkPool01'), 'Language' (set to 'PySpark (Python)'), and 'PySpark (Python)'. Below the toolbar, a status bar says 'Ready'. In the main area, a code editor window contains the following SQL-like code:

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

Below the code, a note says 'Press shift + enter to execute cells'.



The screenshot shows the Azure Data Studio interface with two windows. On the left is the 'Data' window, specifically the 'Workspace' tab, showing a tree view of a 'Lake database' named 'asa_spark_db01'. Underneath, it lists 'Tables' containing 'sales201912', 'salesmall201912', 'testsales2019', and 'Views'. On the right is a 'SQL script 6' window with the following SQL query:

```
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.synapsesql("sql1.dbo.Tbl2")
```

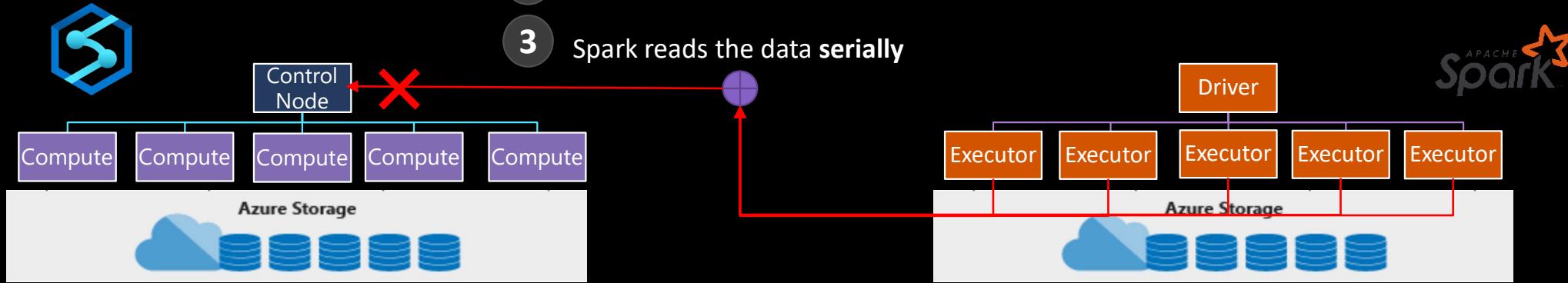
Using Python

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

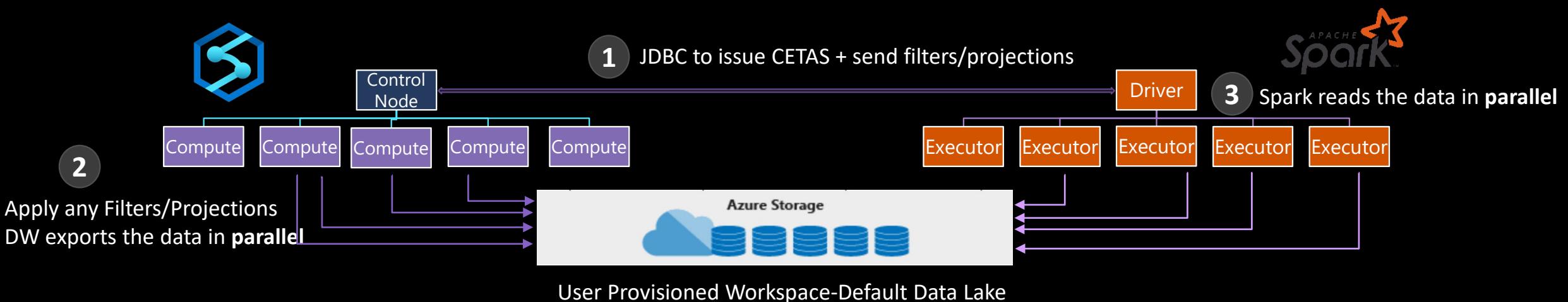
```
%pyspark  
sample = spark.sql("SELECT * FROM tbl1")  
sample.createOrReplaceTempView("tblnew")
```

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

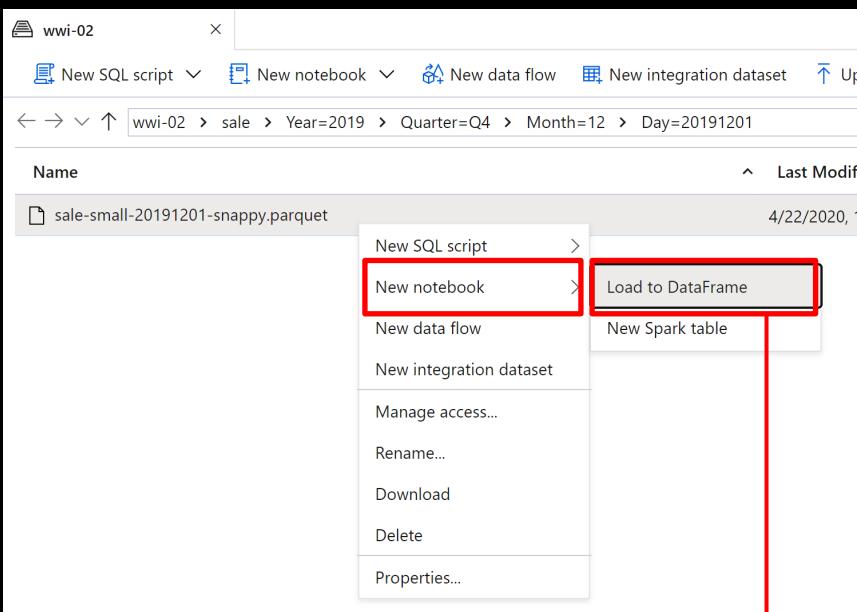
Existing Approach: JDBC



New Approach: JDBC and Polybase



Create a notebook from files in storage



A screenshot of the Azure Notebook interface. The title bar says 'wwi-02' and 'Notebook 3'. The toolbar includes '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 PySpark code:

```
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, it says 'Press shift + enter to execute cells'. At the bottom of the notebook interface are buttons for '+ Code' and '+ Markdown'.

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

Search resources

Publish all Validate all Refresh Discard all

Develop

Notebooks 13

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

NYCTaxi_Docs_Final * SeattleSafetyDoc * Repro

Cell 1

```
1 # Azure storage access info
2 blob_account_name = "azureopendatastorage"
3 blob_container_name = "citydatacontainer"
4 blob_relative_path = "Safety/Release/city=Seattle"
5 blob_sas_token = r""
6
7 # Allow SPARK to read from Blob remotely
8 wasbs_path = 'wasbs://s@blob.core.windows.net/%s' % (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	9m54s

View in monitoring Spark history server

Cell 2

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

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

Cell 3

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

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

View Table Chart

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

Cell 4

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

View results in table format



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

Search resources

Publish all Validate all Refresh Discard all

Develop Notebooks

NYCTaxi_Docs_Final * SeattleSafetyDoc * Repo * 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

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

Apply Cancel

Aid Response

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

Cell 4 [7]

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

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

Develop + <<

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

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

Exploratory Data Analysis

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

Cell 9

```
1 #The charting package needs a Pandas dataframe or numpy array do the conversion
2 sampled_taxi_pd_df = sampled_taxi_df.toPandas()
3
4 # Look at tips by amount count histogram
5 ax1 = sampled_taxi_pd_df['tipAmount'].plot(kind='hist', bins=25, facecolor='lightblue')
6 ax1.set_title('Tip amount distribution')
7 ax1.set_xlabel('Tip Amount ($)')
8 ax1.set_ylabel('Counts')
9 plt.suptitle('')
10 plt.show()
11
12 # How many passengers tip'd by various amounts
13 ax2 = sampled_taxi_pd_df.boxplot(column=['tipAmount'], by=['passengerCount'])
14 ax2.set_title('Tip amount by Passenger count')
15 ax2.set_xlabel('Passenger count')
16 ax2.set_ylabel('Tip Amount ($)')
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

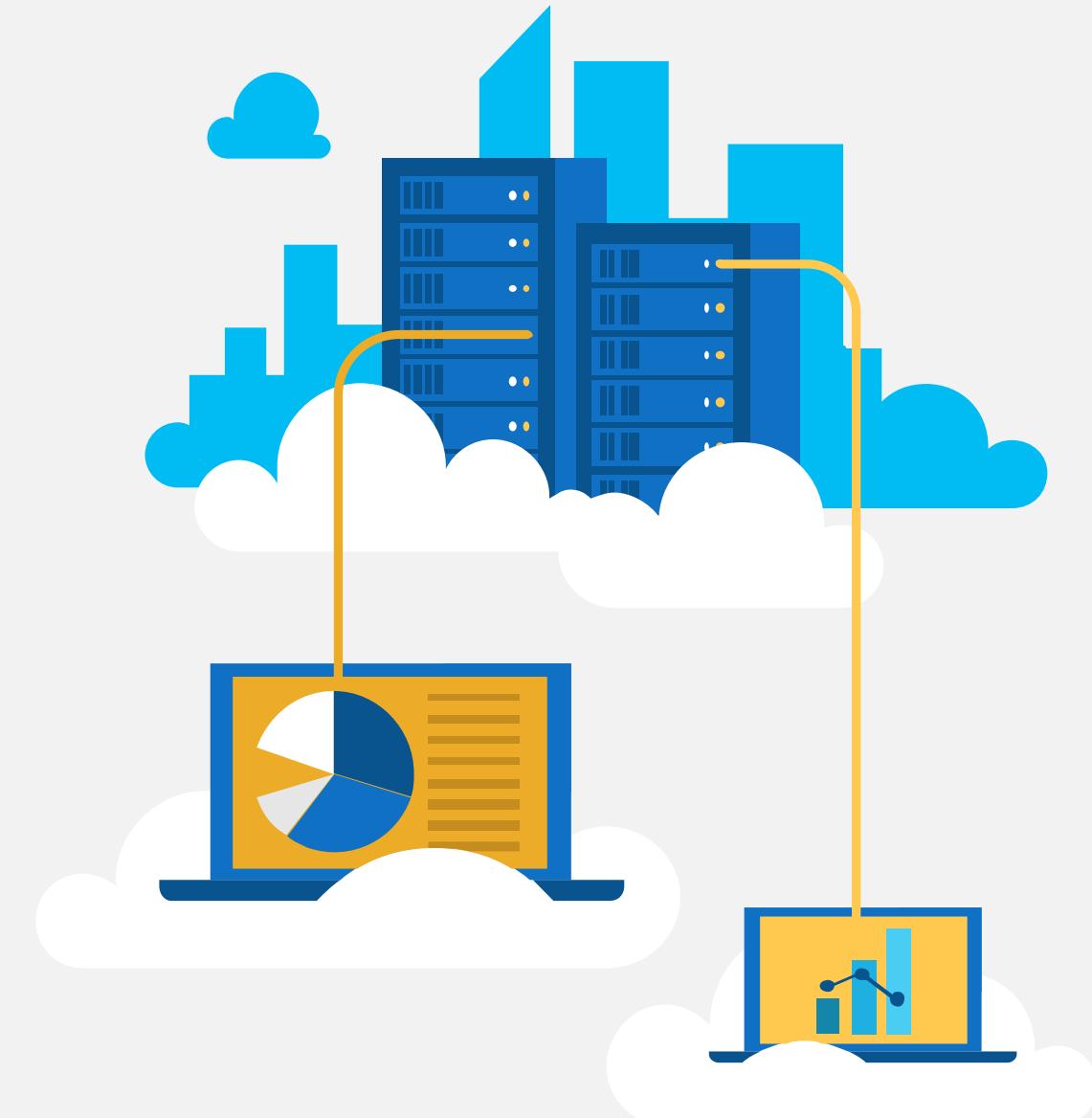
Pop Quiz 2

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



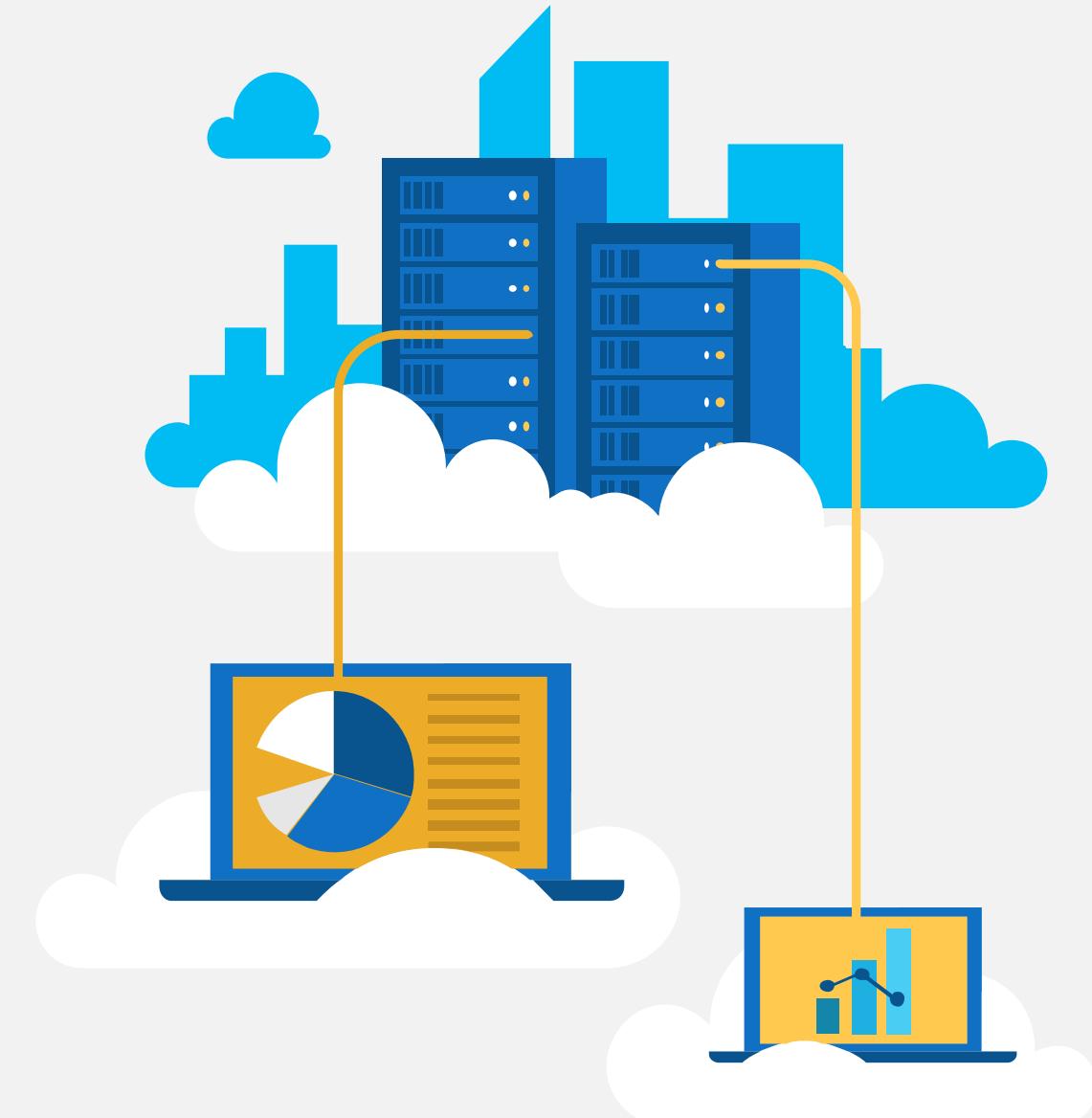
Pop Quiz 2

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 and use code

6142 6491



