



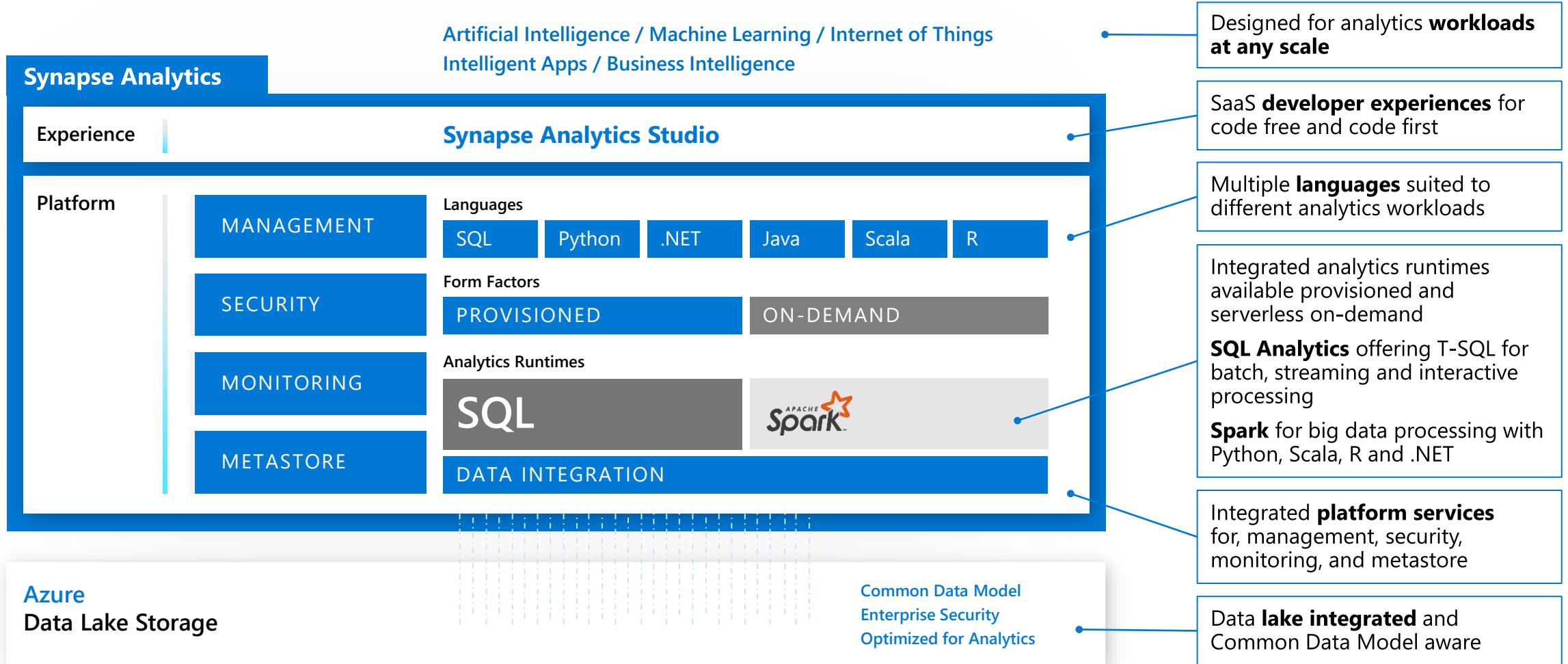
Azure Synapse Analytics

Gary Lee

Cloud Solution Architect (Data / AI / Blockchain)

Azure Synapse Analytics

Limitless analytics service with unmatched time to insight

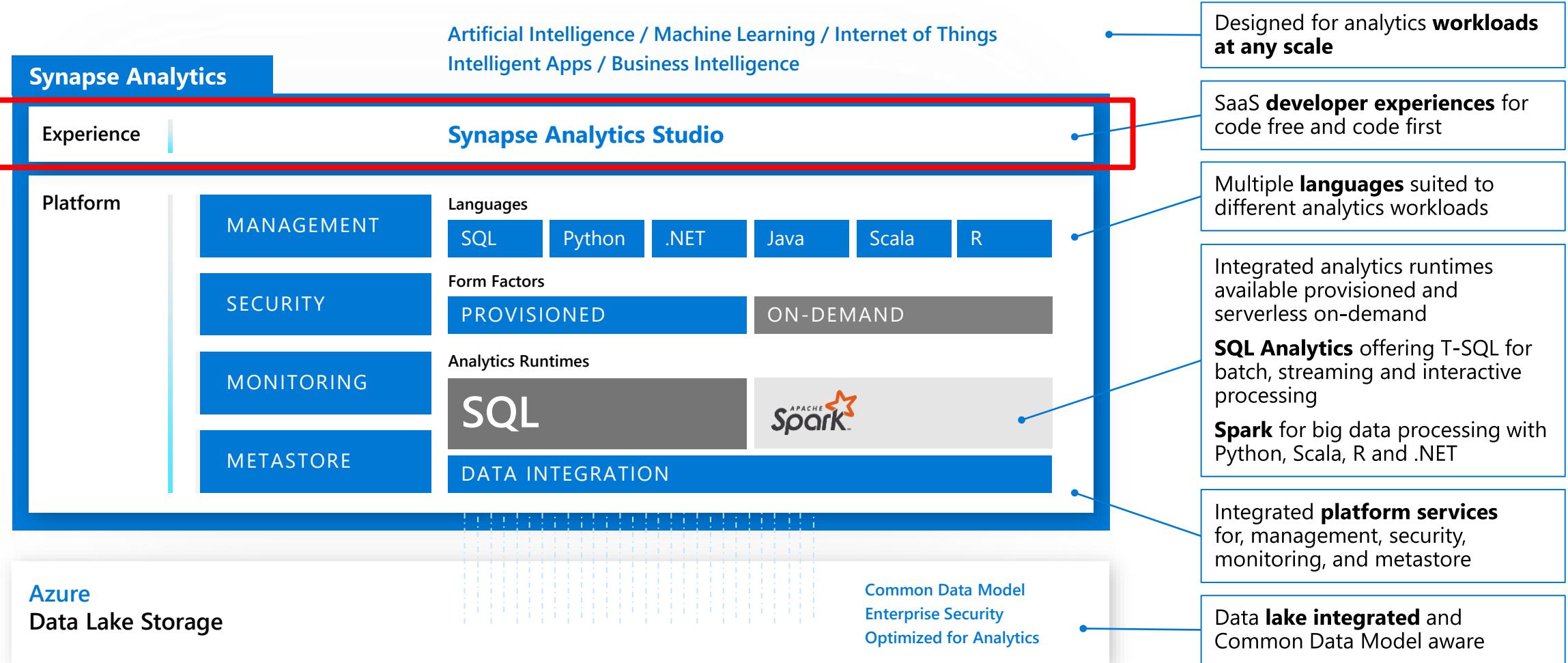




Azure Synapse Analytics Studio

Azure Synapse Analytics

Limitless analytics service with unmatched time to insight



Studio

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

Microsoft Azure | Synapse Analytics > prlangadws2

Synapse workspace
prlangadws2

New ▾

Overview Data Develop Orchestrate Monitor Manage

Ingest Explore Analyze Visualize

Resources

Recent Pinned

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

Show more ▾

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

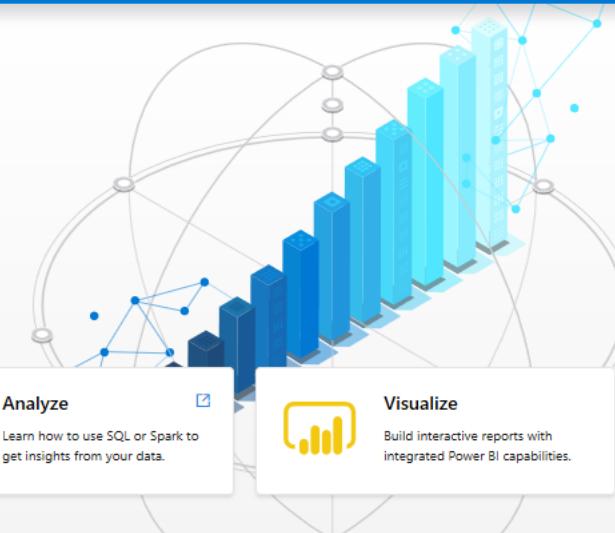
Useful links

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

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

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

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



Synapse Studio

Synapse Studio divided into **Activity hubs**.

These organize the tasks needed for building analytics solution.

The screenshot shows the Microsoft Azure Synapse Studio interface. At the top, there's a navigation bar with 'Microsoft Azure' and 'Synapse Analytics' followed by a workspace name 'prlangadws2'. On the far right of the bar are icons for notifications, a smiley face, a question mark, and a user profile. Below the bar is a header with 'Synapse workspace' and the workspace name 'prlangadws'. A red box highlights the left sidebar, which contains links for 'Overview', 'Data', 'Develop', 'Orchestrate', 'Monitor', and 'Manage'. A red arrow points from the 'New' button in the workspace header down to the 'Ingest' section in the sidebar. The main area is divided into six activity hubs: 'Overview', 'Data', 'Develop', 'Orchestrate', 'Monitor', and 'Manage'. Each hub has a title, a brief description, and a corresponding icon. The 'Overview' hub is currently selected.

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

Data
Explore structured and unstructured data

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

Orchestrate
Design pipelines that move and transform data.

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

Manage
Configure the workspace, pool, access to artifacts

Resources

Recent Pinned

NAME
NYCTaxiFinalWUS2_R
NYCTaxiFinalWUS2_R
HolidayDataPipeline
create notebook on s
Data flow 1

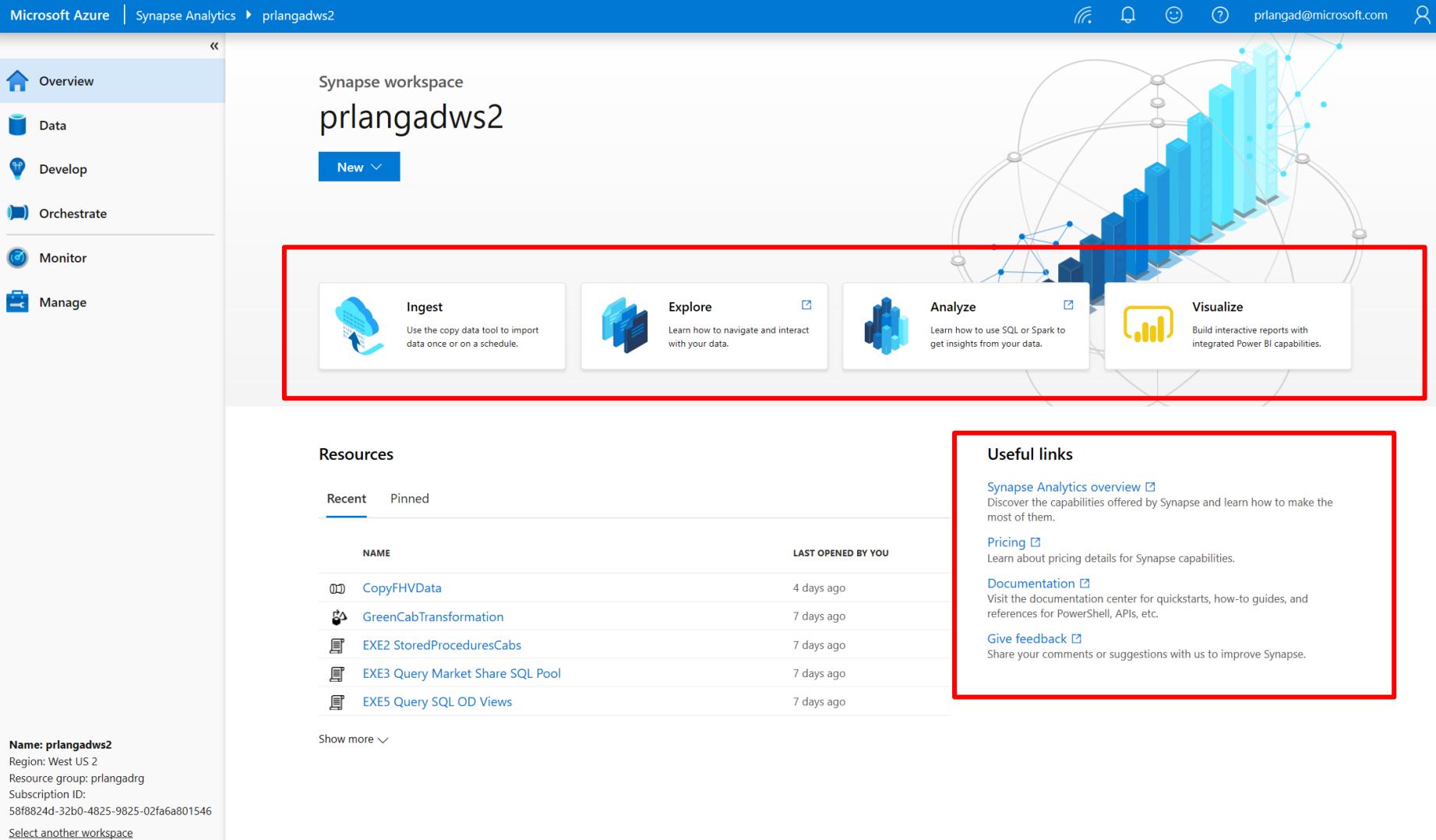
Name: prlangadws2
Region: West US 2
Resource group: prlangadrg
Subscription ID: 66666666-6666-6666-6666-666666666666
Show more ▾



Synapse Studio Overview hub

Overview Hub

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



The screenshot shows the Microsoft Azure Synapse Analytics Overview Hub. The left sidebar includes links for Overview, Data, Develop, Orchestrate, Monitor, and Manage. The main area displays a "Synapse workspace" named "prlangadws2". A large red box highlights the "Ingest", "Explore", "Analyze", and "Visualize" sections. Below these, the "Resources" section lists recent and pinned items, and the "Useful links" section provides links to Synapse Analytics overview, Pricing, Documentation, and Give feedback.

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

Explore
Learn how to navigate and interact with your data.

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

Visualize
Build interactive reports with integrated Power BI capabilities.

Resources

Recent Pinned

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

Show more ▾

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

Useful links

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

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

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

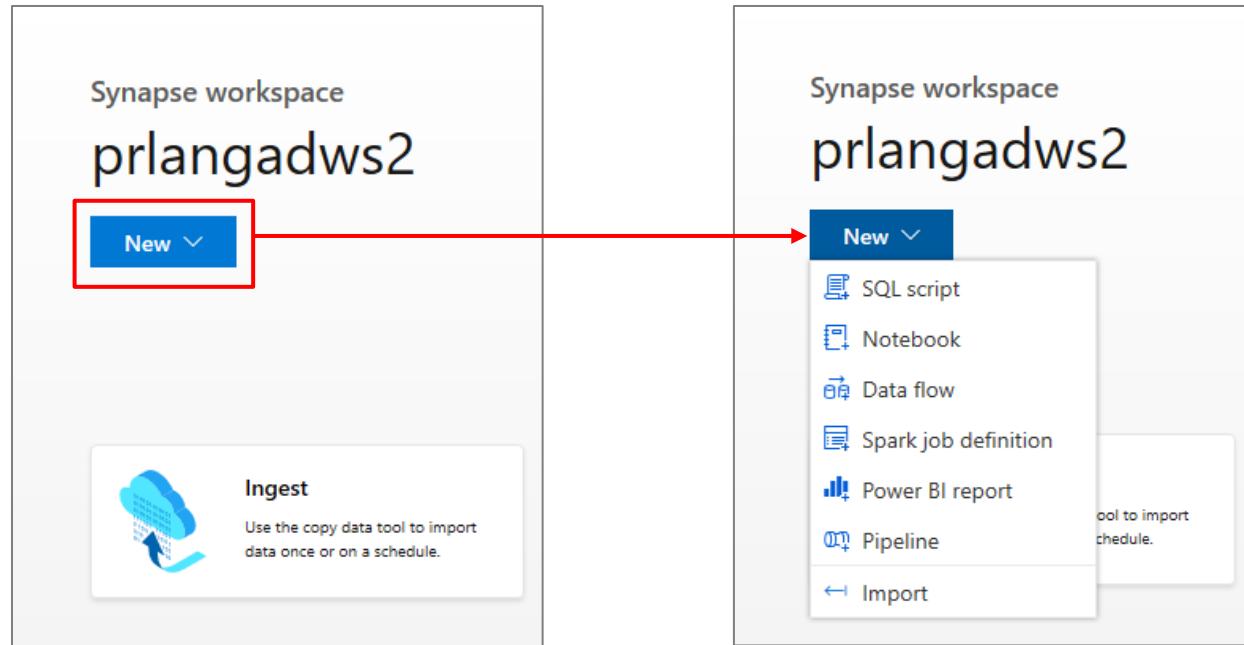
[Give feedback](#)
Share your comments or suggestions with us to improve Synapse.

Overview Hub

Overview

New dropdown – offers quickly start work item

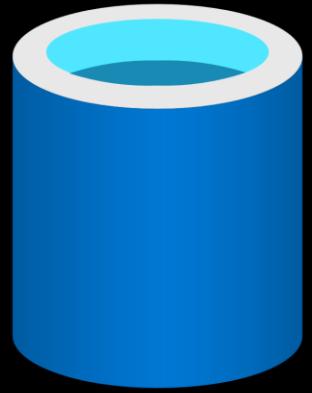
Recent & Pinned – Lists recently opened code artifacts. Pin selected ones for quick access



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

Show more ▾

Recent	Pinned
NAME	LAST OPENED BY YOU
NYCTAXI	6 hours ago



Synapse Studio **Data hub**

Data Hub

Explore data inside the workspace and in linked storage accounts

Microsoft Azure | Synapse Analytics > prlangadws2 | | | prlangad@microsoft.com |

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

Publish all Validate all Refresh Discard all

Data

Storage accounts

Databases

Datasets

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

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

Data Hub – Storage accounts

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

The screenshot shows the Microsoft Azure Data Hub interface for managing storage accounts and filesystems. On the left, a sidebar lists various account types: ADLS Gen2 Account, Container (filesystem), Develop, Orchestrate, Monitor, and Manage. The 'Data' section is selected. Below it, the 'Storage accounts' section lists 'prlangaddemosa (Primary)' and 'nyctlc'. The 'Manage' section lists 'filesystem', 'holidaydatacontainer', 'isdweatherdatacontainer', 'prlangaddemosa', 'tmpcontainer', and 'wwimporters'. The main pane displays the contents of the 'nyctlc' storage account under the 'Data' tab. The 'Filepath' bar shows 'nyctlc > yellow'. The file listing table has columns: NAME, LAST MODIFIED, CONTENT TYPE, and SIZE. The table lists 26 folder entries from 'puYear=2001' to 'puYear=2026', all of which are 'Folder' type and were modified on 10/25/2019.

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

Data Hub – Storage accounts

Preview a sample of your data

The screenshot shows the Azure Synapse Studio Data Hub interface. On the left, there's a sidebar with sections for Data, Storage accounts, Databases, and Datasets. Under Storage accounts, 'nyctlc' is selected, and under it, 'filesystem' is also selected. In the main area, a 'SampleCSVFile_2kb.csv' file is listed under 'nyctlc'. A red arrow points from the 'Preview' option in the context menu for this file to the preview window on the right.

SampleCSVFile_2kb.csv

Path https://prlangaddemosa.dfs.core.windows.net/filesystem/SampleCSVFile_2kb.csv
Modified 10/29/2019, 1:30:21 PM

With column header On

USER_ID	USERNAME	FIRST_NAME	LAST_NAME	GENDER	PASSWORD
1	rogers63	david	john	Female	e6a33eee18
2	mike28	rogers	paul	Male	2e7dc6b8a1
3	rivera92	david	john	Male	1c3a8e03f4
4	ross95	maria	sanders	Male	62f0a68a41
5	paul85	morris	miller	Female	61bd060b07
6	smith34	daniel	michael	Female	7055b3d9f5
7	james84	sanders	paul	Female	b7f72d6eb9
8	daniel53	mark	mike	Male	299cbf7171
9	brooks80	morgan	maria	Female	aa736a35dc
10	morgan65	paul	miller	Female	a28dca31f5

OK

Data Hub – Storage accounts

See basic file properties

The screenshot shows the Azure Synapse Studio Data Hub interface. On the left, there's a navigation sidebar with sections for Data, Storage accounts, Databases, and Datasets. Under Storage accounts, 'prlangaddemosa (Primary)' is expanded, showing its contents: filesystem, holidaydatacontainer, isdweatherdatacontainer, nyctlc, prlangaddemosa, tmpcontainer, and wwidporters. The 'filesystem' item is selected. In the main area, there are two tabs: 'nyctlc' and 'filesystem'. The 'filesystem' tab is active, showing a list of files and folders. A file named 'SampleCSVFile_2kb.csv' is selected and highlighted with a blue border. A context menu is open over this file, listing options: Preview, New notebook, Copy ABFSS path, Manage Access..., Rename..., Download, Delete, and Properties... (the last option is highlighted with a red box). At the top of the main area, there are buttons for Publish all, Validate all, Refresh, Discard all, Upload, Download, New Folder, and Select.

The screenshot shows the 'Properties' dialog box for the selected file. It has two main sections: System Properties and User Properties. The System Properties section includes fields for Name (SampleCSVFile_2kb.csv), URL (https://prlangaddemosa.dfs.core.windows.net/filesystem/SampleCSVFile_2kb.csv), LastModified (Tue, 29 Oct 2019 20:30:21 GMT), CacheControl, ContentType (application/vnd.ms-excel), ContentDisposition, ContentEncoding, and ContentLanguage. The User Properties section is currently empty. At the bottom, there are 'Save' and 'Cancel' buttons. A red arrow points from the 'Properties...' option in the context menu on the left to the 'Properties...' button in the dialog box on the right.

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

Data Hub – Storage accounts

Manage Access - Configure standard POSIX ACLs on files and folders

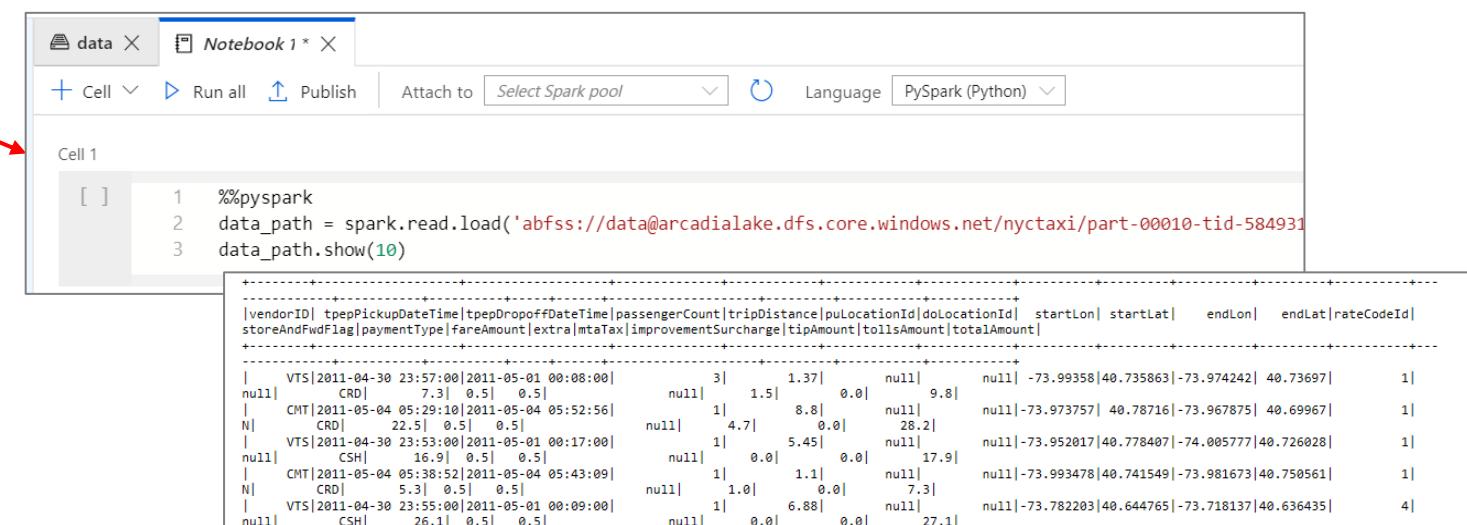
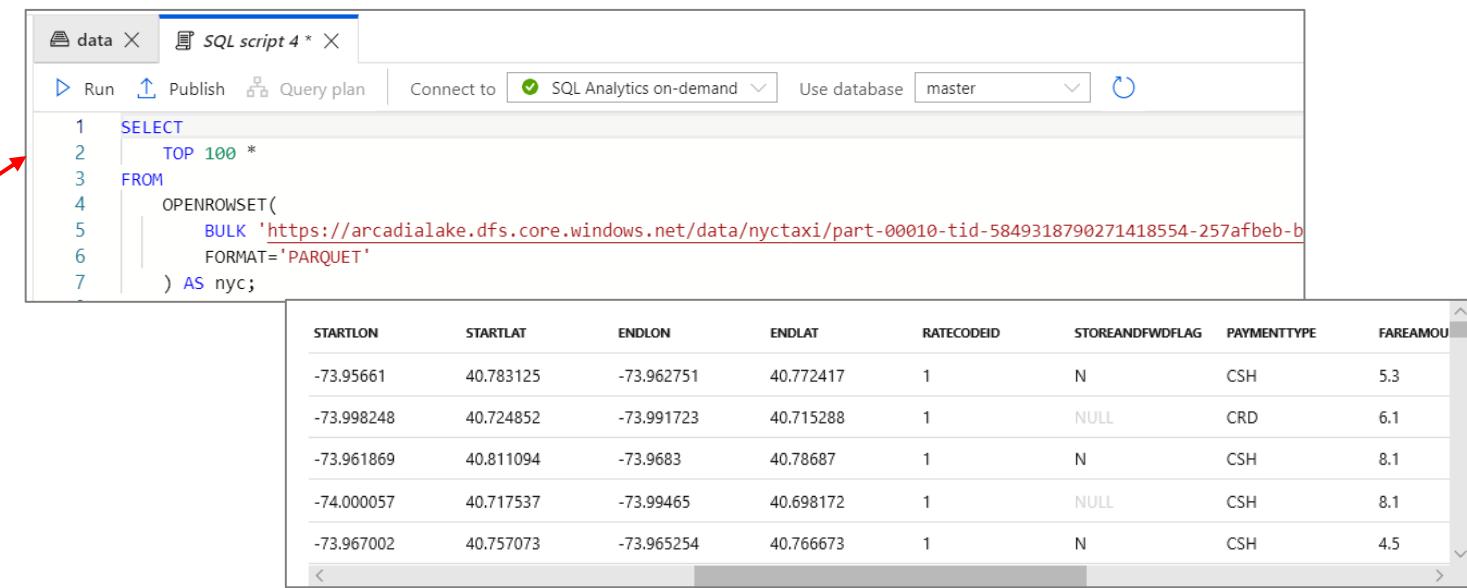
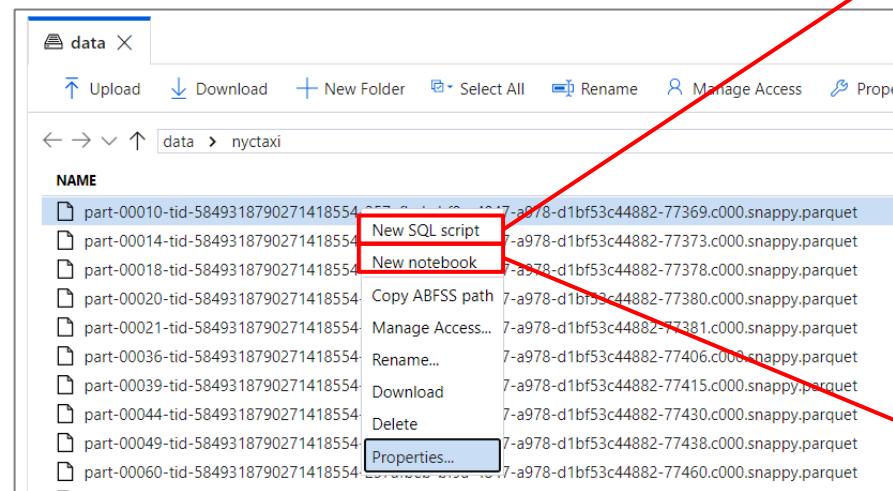
The screenshot shows the Azure Synapse Studio Data Hub interface. On the left, there's a navigation sidebar with sections for Data, Storage accounts, Databases, and Datasets. Under Storage accounts, 'prlangaddemo (Primary)' is selected, showing its contents: filesystem, holidaydatacontainer, isdweatherdatacontainer, nyctlc, prlangaddemo, tmpcontainer, and wwimporters. The 'filesystem' item is also selected. In the main workspace, there are two tabs: 'nyctlc' and 'filesystem'. The 'filesystem' tab is active, displaying a list of files and folders: synapse, temp, tmp, dbo.StoreSales.parquet, dbo.StoreSales.txt, employee.json, and SampleCSVFile_2kb.csv. A context menu is open over the 'SampleCSVFile_2kb.csv' file, listing options: Preview, New notebook, Copy ABFSS path, Manage Access..., Rename..., Download, Delete, and Properties... . The 'Manage Access...' option is highlighted with a red box and a red arrow pointing to the corresponding window on the right.

The screenshot shows the 'Manage Access' dialog for the file 'SampleCSVFile_2kb.csv'. The title bar says 'Manage Access' and the sub-header says 'Managing permissions for: filesystem/SampleCSVFile_2kb.csv'. It lists 'Users and groups': '\$superuser (Owner)' and '\$superuser (Owning Group)'. Below that are 'Other' and 'Mask' sections. Under 'Permissions for: \$superuser', there are checkboxes for 'Read', 'Write', and 'Execute', all of which are checked. There's also a section to 'Add user or group' with a text input field 'Enter a UPN or Object ID' and a 'Add' button. At the bottom are 'Save' and 'Cancel' buttons.

Data Hub – Storage accounts

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

T-SQL or PySpark auto-generated



Data Hub – Storage accounts

SQL Script from Multiple files

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

The screenshot shows the Azure Synapse Analytics Studio interface. On the left, there is a file browser window titled "isdweatherdatacontainer > ISDWeathercurated". It lists several Parquet files with their names and last modified dates. A red arrow points from the "New SQL script" option in a context menu (which is highlighted with a red box) to the generated SQL script on the right. The SQL script is a T-SQL query that reads multiple Parquet files from a specified URL using OPENROWSET and BULK options.

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

```

1 -- Read multiple parquet files with same schema
2 SELECT
3     TOP 100 *
4 FROM
5     OPENROWSET(
6         BULK 'https://prlangaddemosa.dfs.core.windows.net/isdweatherdatacontainer/ISDWeathercurated/\*.parquet',
7         FORMAT = 'Parquet'
8     ) AS [r]
9 WHERE
10    r.filepath() in (
11        'https://prlangaddemosa.dfs.core.windows.net/isdweatherdatacontainer/ISDWeathercurated/part-00000',
12        'https://prlangaddemosa.dfs.core.windows.net/isdweatherdatacontainer/ISDWeathercurated/part-00001',
13        'https://prlangaddemosa.dfs.core.windows.net/isdweatherdatacontainer/ISDWeathercurated/part-00002',
14        'https://prlangaddemosa.dfs.core.windows.net/isdweatherdatacontainer/ISDWeathercurated/part-00003',
15        'https://prlangaddemosa.dfs.core.windows.net/isdweatherdatacontainer/ISDWeathercurated/part-00004',
16        'https://prlangaddemosa.dfs.core.windows.net/isdweatherdatacontainer/ISDWeathercurated/part-00005',
17        'https://prlangaddemosa.dfs.core.windows.net/isdweatherdatacontainer/ISDWeathercurated/part-00006',
18        'https://prlangaddemosa.dfs.core.windows.net/isdweatherdatacontainer/ISDWeathercurated/part-00007'
19    )

```

Data Hub – Databases

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

The screenshot illustrates the Data Hub interface for exploring databases in a workspace. It shows two separate database hierarchies side-by-side.

Left Database Hierarchy:

- Storage accounts:** 2 items
- Databases:** 3 items
 - sql1 (SQL pool)**
 - sample (SQL on-demand)**
 - default (Spark)**
 - Tables:** 2 items
 - nytaxiyellow7days**
 - searchlogtable**
- Datasets:** 2 items

Right Database Hierarchy (Detailed View):

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

Data Hub – Databases

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

Databases 3

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

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

Databases 3

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

Notebook 1 * X

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

Cell 1

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

Data Hub – Datasets

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

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

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

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



Synapse Studio

Develop hub

Develop Hub

Overview

It provides development experience to query, analyze, model data

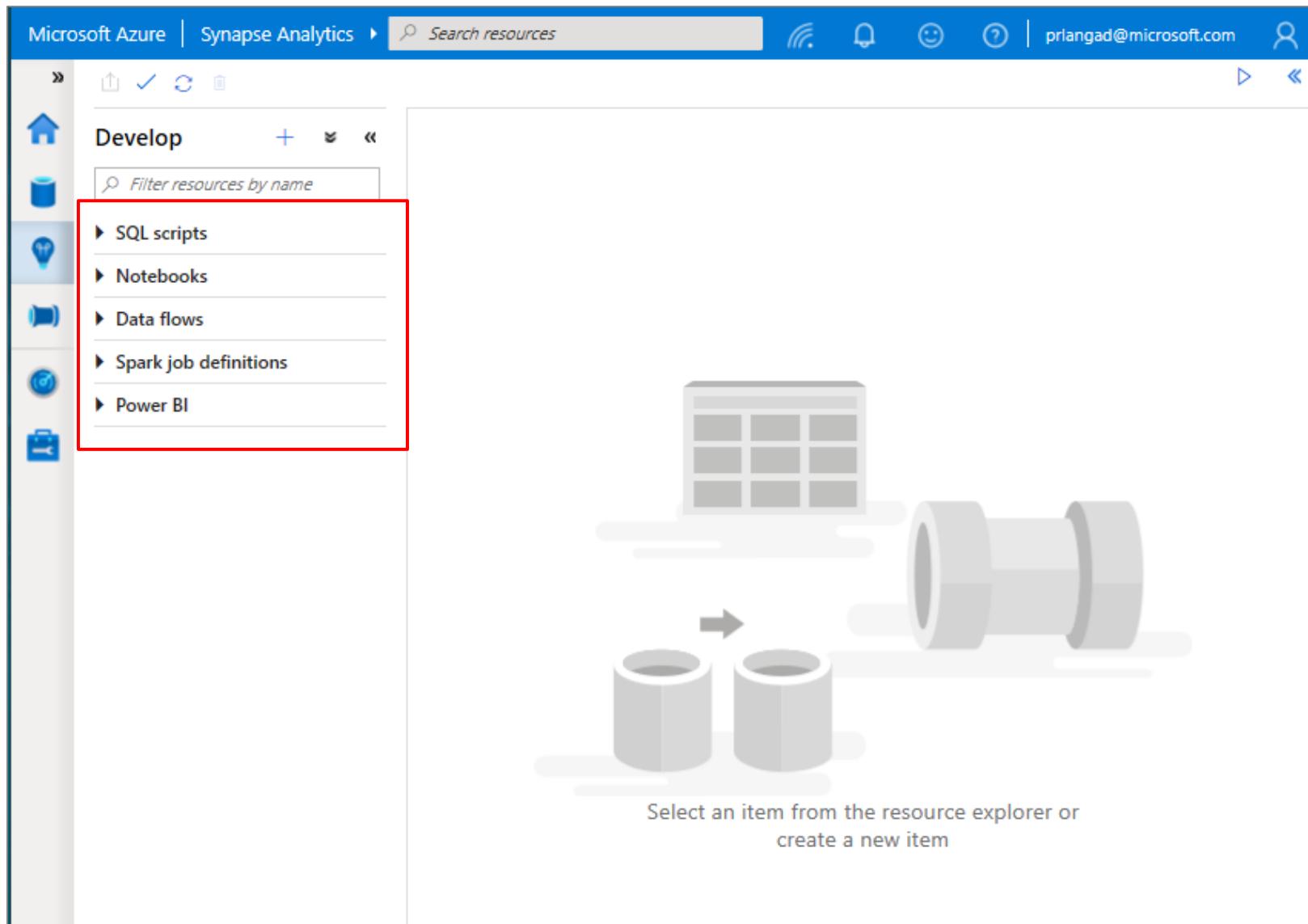
Benefits

Multiple languages to analyze data under one umbrella

Switch over notebooks and scripts without loosing content

Code intellisense offers reliable code development

Create insightful visualizations



Develop Hub - SQL scripts

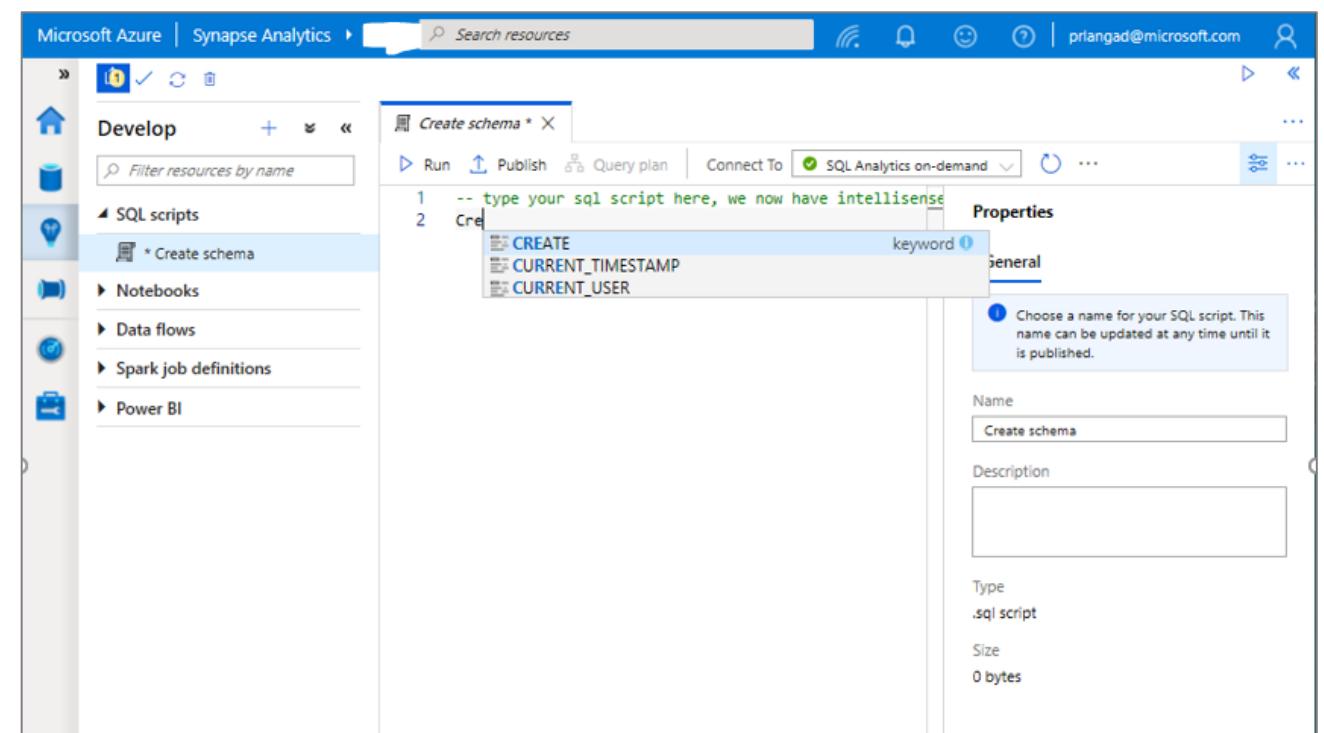
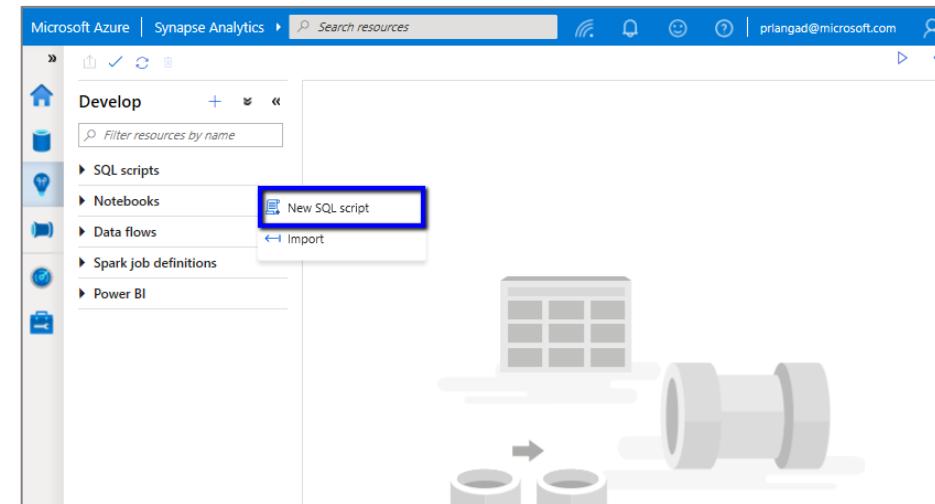
SQL Script

Authoring SQL Scripts

Execute SQL script on provisioned SQL Pool or SQL On-demand

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

Language support and intellisense



Develop Hub - SQL scripts

SQL Script

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

The screenshot shows the Azure Synapse Studio interface with the following components:

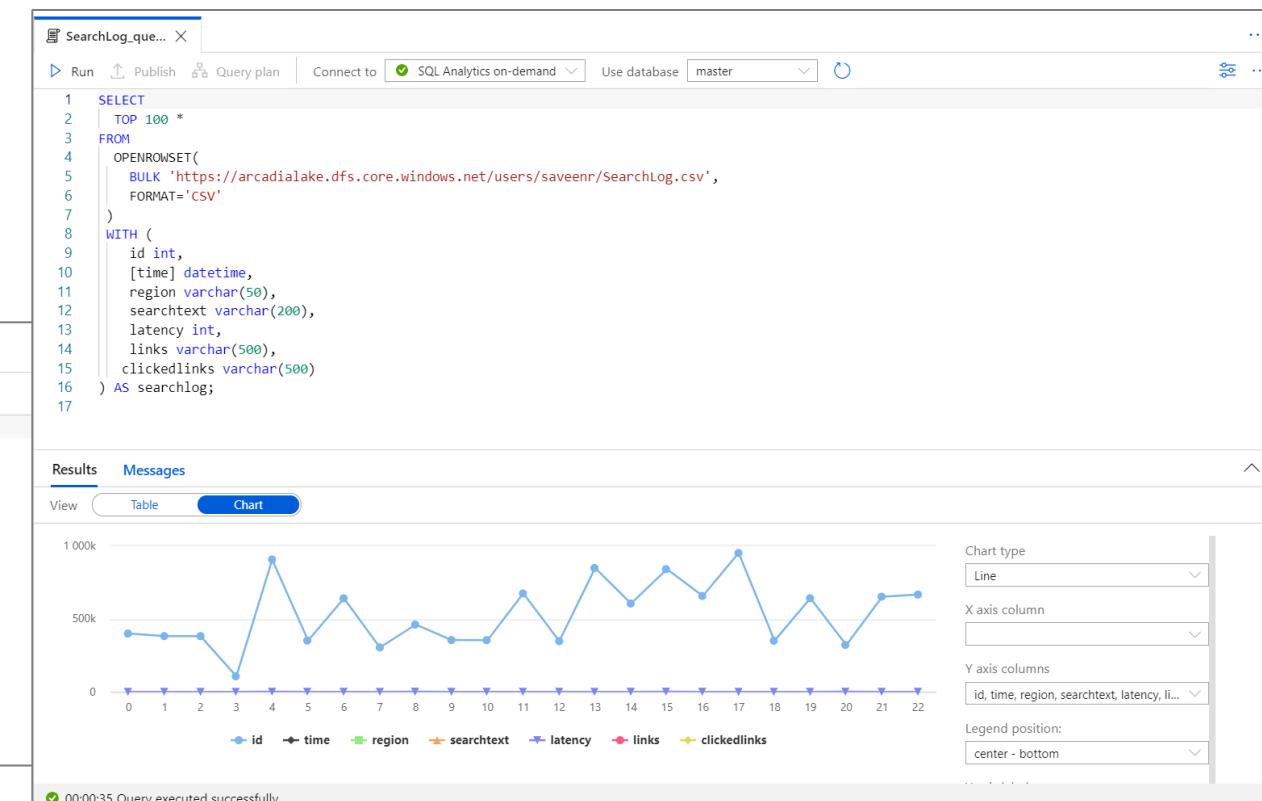
- Top Bar:** Shows "SearchLog_que..." in the title bar, "Run", "Publish", "Query plan", "Connect to SQL Analytics on-demand", "Use database master", and a refresh button.
- Script Editor:** Displays the following T-SQL script:

```

1 SELECT
2     TOP 100 *
3 FROM
4     OPENROWSET(
5         BULK 'https://arcadialake.dfs.core.windows.net/users/saveenr/SearchLog.csv',
6         FORMAT='CSV'
7     )
8     WITH (
9         id int,
10        [time] datetime,
11        region varchar(50),
12        searchtext varchar(200),
13        latency int,
14        links varchar(500),
15        clickedlinks varchar(500)
16    ) AS searchlog;
17

```
- Results View:** Contains a table with columns ID, TIME, and REGION. The data is:

ID	TIME	REGION
399266	2019-10-15T11:53:04.0000000	en-us
382045	2019-10-15T11:53:25.0000000	en-gb
382045	2019-10-16T11:53:42.0000000	en-gb
106479	2019-10-16T11:53:10.0000000	en-ca
906441	2019-10-16T11:54:18.0000000	en-us
- Export Options:** A red box highlights the "Export results" dropdown menu, which includes options: CSV, Excel, JSON, and XML.



Develop Hub - Notebooks

Notebooks

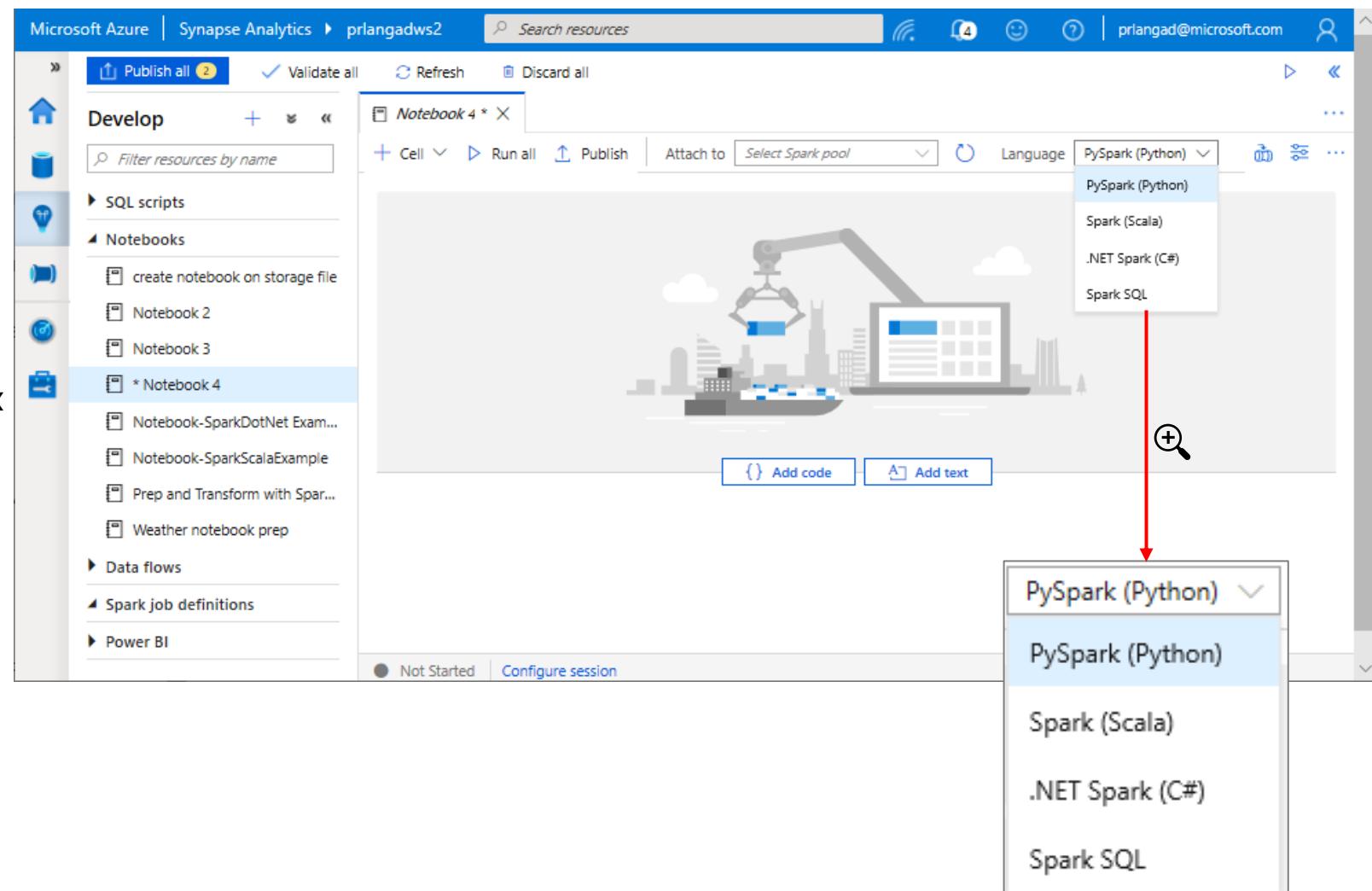
Allows to write multiple languages in one notebook

`%%<Name of language>`

Offers use of temporary tables across languages

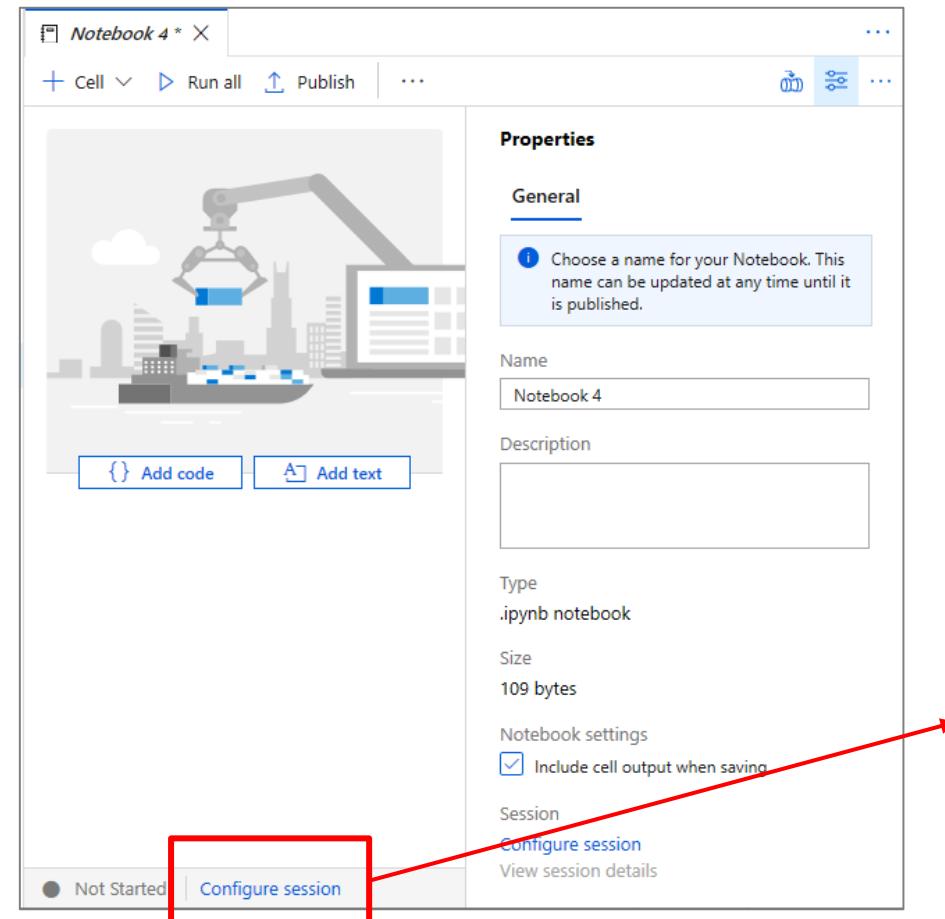
Language support for Syntax highlight, syntax error, syntax code completion, smart indent, code folding

Export results



Develop Hub - Notebooks

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



Configure session

Notebook 4

* Session timeout ⓘ

30

* Executors ⓘ

2

* Executor size ⓘ

Small (4 vCPU, 28GB memory)

* Driver size ⓘ

Small (4 vCPU, 28GB memory)

Apply

Cancel

Develop Hub - Notebooks

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

The screenshot shows the Azure Synapse Studio interface for developing notebooks. At the top, there's a toolbar with options like '+ Cell', 'Run all', 'Publish', 'Attach to' (set to 'spark1'), 'Language' (set to 'PySpark (Python)'), and a three-dot menu. Below the toolbar, a cell titled 'Cell 1' contains the following code:

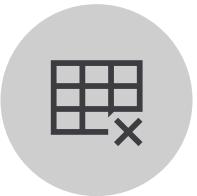
```
1 %%pyspark
2 data_path = spark.read.load('abfss://data@arcadialake.dfs.core.windows.net/nytaxi/part-00010-tid-584931879')
3 data_path.show(10)
```

Below the code, a message indicates the command was executed in 4mins 33s 99ms by saveenr on 11-16-2019 09:36:31.944 -08:00. A section titled 'Job execution' shows the status of three tasks: 'Job 0' (Succeeded), 'Job 1' (Succeeded), and 'Job 2' (Succeeded). Each task has a green progress bar indicating completion. To the right, there are links to 'View in monitoring' and 'Spark history server'. The main area then displays the data from the 'data_path.show(10)' command as a table:

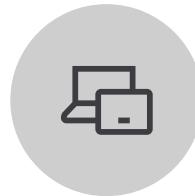
vendorID	tpepPickupDateTime	tpepDropoffDateTime	passengerCount	tripDistance	puLocationId	doLocationId	startLon	startLat	endLon	endLat	rateCodeId	lpep_pickup_dts	lpep_dropoff_dts	store_and_fwd_flag	pickup_type	payment_type	fare_amount	extra	mta_tax	tip_amount	total_amount	congestion_surcharge	surrogate_key
null	2 2017-03-09 21:30:11 2017-03-09 21:44:20											1	4.06	148	48	null	null	null					
null	1	N	1	14.0	0.5	0.5						0.3	3.06	0.0	18.36								
null	2 2017-03-09 21:47:00 2017-03-09 21:58:01											1	2.73	48	107	null	null	null					
null	1	N	2	11.5	0.5	0.5						0.3	0.0	0.0	12.8								
null	2 2017-03-09 22:01:08 2017-03-09 22:11:16											1	2.27	79	162	null	null	null					
null	1	N	1	10.0	0.5	0.5						0.3	2.82	0.0	14.12								
null	2 2017-03-09 22:16:05 2017-03-10 06:26:11											1	3.86	237	41	null	null	null					
null	1	N	1	12.0	0.5	0.5						0.3	3.99	0.0	17.29								
null	2 2017-03-31 06:31:53 2017-03-31 06:41:48											1	3.45	41	162	null	null	null					
null	1	N	2	12.0	0.5	0.5						0.3	0.0	0.0	13.3								
null	1 2017-03-01 00:00:00 2017-03-01 00:14:22											1	2.8	261	79	null	null	null					
null	1	N	1	12.5	0.5	0.5						0.3	1.0	0.0	14.8								
null	1 2017-03-01 00:00:00 2017-03-01 00:19:30											1	6.0	87	142	null	null	null					
null	1	N	1	19.5	0.5	0.5						0.3	3.5	0.0	24.3								

At the bottom, there are buttons for 'Ready' (checked), 'Stop session', 'Spark history server', and 'Configure session'.

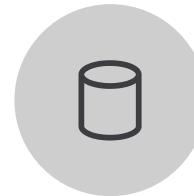
Dataflow Capabilities



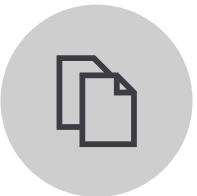
Handle upserts, updates, deletes on sql sinks



Add new partition methods



Add schema drift support



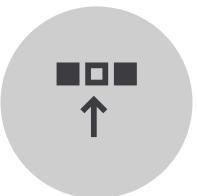
Add file handling (move files after read, write files to file names described in rows etc)



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



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



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

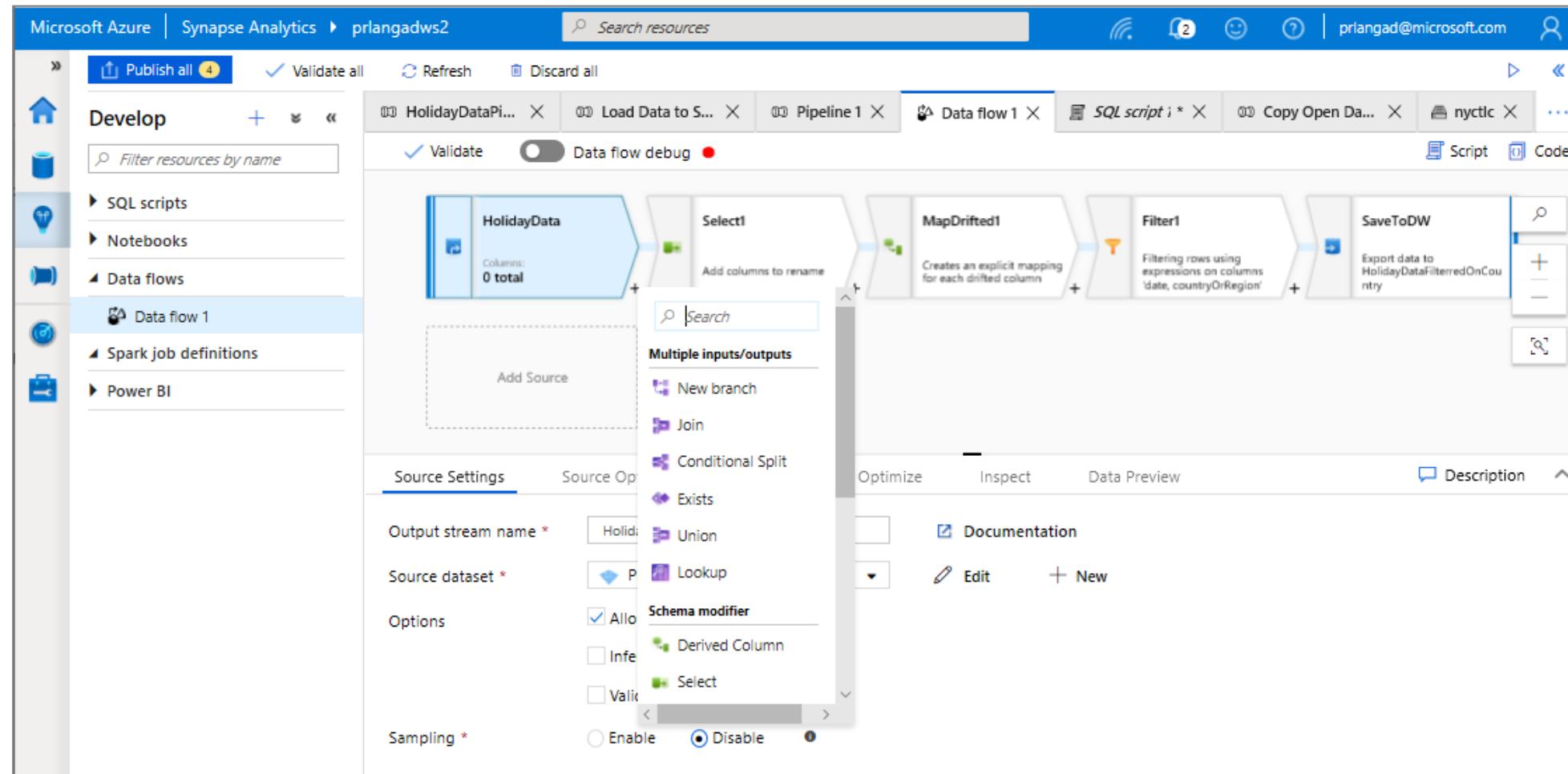


Implement commonly used ETL patterns as templates(SCD Type1, Type2, Data Vault)

Develop Hub - Data Flows

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

Provides a code-free experience.



Develop Hub – Power BI

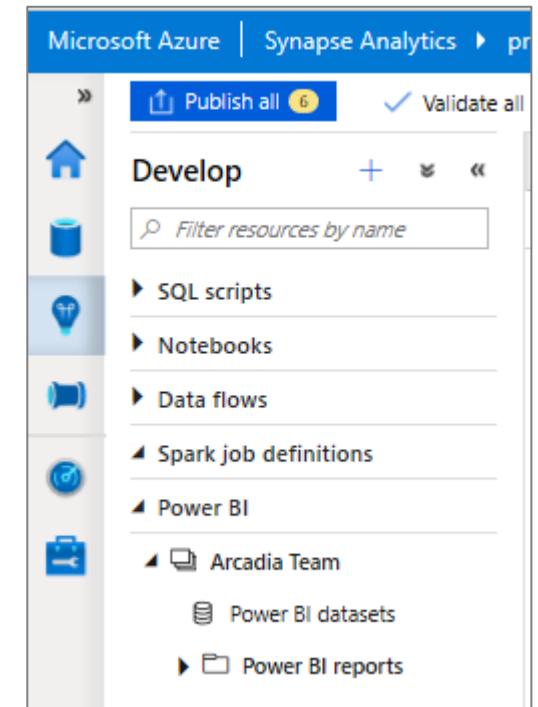
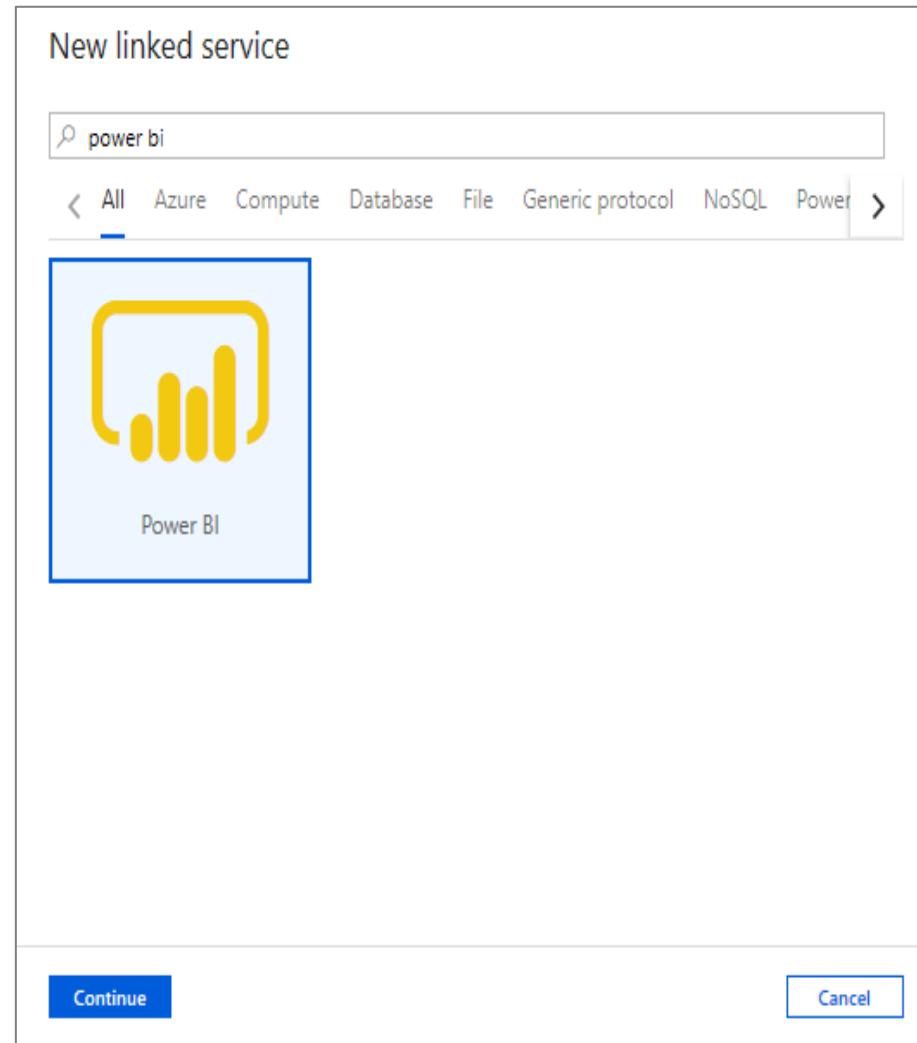
Overview

Create Power BI reports in the workspace

Provides access to published reports in the workspace

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

Visually explore and analyze data



Develop Hub – Power BI

View published reports in Power BI workspace

The screenshot shows the Azure Synapse Analytics Develop Hub interface for Power BI. On the left, a sidebar lists resources under 'Develop': SQL Scripts, YellowCabExploration_sqld, Notebooks (AMAutoMLPredict, AutoML, Data Download_Weather, * PrepareTaxiData, yellowcabprep, YellowCabPrepare), Data flows (PrepareCabDataFlow), Spark job definitions, Power BI (SynapseNYTaxiInsights, Power BI Datasets, Power BI Reports, SynapseNYIgnite2019, SynapseNYIgnite2019 (1)). The main area displays a line chart titled 'Rides vs Greenrides and Yellowrides by DatePickup' with three data series: Rides (blue), Greenrides (yellow), and Yellowrides (green). The chart spans from April 2015 to April 2016. To the right of the chart are several Power BI tool panels: 'Filters' (with 'Search' and 'Add data fields here' options for both 'Filters on this page' and 'Filters on all pages'), 'VISUALIZATIONS' (listing various chart icons like bar, line, pie, etc.), 'FIELDS' (listing fields such as dimHoliday, dimNYCLocations, Fhv, GreenCab, PredictedValues, vwFhvMarketShare, vwGrnCabMarketS..., vwMarketShareBy..., vwPredictedValues, vwYelCabMarketSh..., weather, YellowCab, YellowCabTripsHoli...), 'DRILLTHROUGH' (Cross-report set to Off, Keep all filters set to On), and 'Add drillthrough fields here'.

Develop Hub – Power BI

Edit reports in Synapse workspace

The screenshot shows the Azure Synapse Analytics Develop Hub interface for Power BI. On the left, the sidebar lists various resources: SQL Scripts, Notebooks (including AMLAutoMLPredict, AutoML, Data Download_Weather, * PrepareTaxiData, yellowcabprep, YellowCabPrepare), Data flows (PrepareCabDataFlow), Spark job definitions, and Power BI datasets (SynapseNYTaxiInsights, Power BI Reports). The Power BI Reports section is expanded, showing the report "SynapseNYIgnite2019" which is currently selected.

The main area displays a report titled "SynapseNYIgnite2019". It contains two visualizations: a line chart showing "Triphives, Greenrides and Yellowrides by DatePickup" from Jan 2015 to Jul 2017, and a bar chart showing "Trips by holidayName" for various holidays like Veterans Day, Thanksgiving, Martin Luther King Jr. Day, Columbus Day, and Memorial Day. The bar chart shows trip counts ranging from approximately 0.2M to 1.8M.

To the right of the report, the Power BI Visualizations pane is open, showing the following details for the selected visualization:

- Filters:** holidayName is (All)
- numTrips:** is (All)
- Add data fields here**
- Filters on this page:** Add data fields here
- Filters on all pages:** Add data fields here
- Axis:** holidayName
- Legend:** Add data fields here
- Value:** numTrips
- Toolips:** Add data fields here
- DRILLTHROUGH:** Add data fields here
- Cross-report:** Add data fields here

The "holidayName" field is highlighted in yellow in the Axis and Value sections. The "numTrips" field is also highlighted in yellow in the Value section.

Develop Hub – Power BI

Publish edited reports in Synapse workspace to Power BI workspace

The screenshot shows the Azure Synapse Analytics Develop Hub interface. On the left, there's a sidebar with icons for Home, Datasets, Notebooks, Data Flows, Spark job definitions, Power BI, and Power BI Reports. Under Power BI Reports, two items are listed: 'SynapseNYIgnite2019' and 'SynapseNYIgnite2019 (1)'. The 'SynapseNYIgnite2019' item is highlighted with a blue background. The main workspace shows a Power BI report with a chart and a bar chart. A red box highlights the 'Save' button in the top navigation bar. Below the workspace, the text 'Publish changes by simple save report in workspace' is displayed in red. The right side of the screen shows the Power BI visualizations and fields panes.

Publish changes by simple save report in workspace

Save
Save this report

File View

Develop

Publish all 2

Validate all

Refresh

Discard all

yellowcabprep X PrepareTaxiData * X 1 Marketshare X 2 MostTripsHo... X AutoML X SynapseNYIgnite2019 X ...

Filter resources by name

SQL script 1

YellowCabExploration_sqld

Notebooks

- AMLAutoMLPredict
- AutoML
- Data Download_Weather
- * PrepareTaxiData
- yellowcabprep
- YellowCabPrepare

Data flows

- PrepareCabDataFlow
- Spark job definitions
- Power BI
- SynapseNYTaxiInsights
- Power BI Datasets
- Power BI Reports
- SynapseNYIgnite2019
- SynapseNYIgnite2019 (1)

Page 1 +

Filters

Search

Filters on this visual

- holidayName is (All)
- numTrips is (All)

Add data fields here

Visualizations

Fields

Search

dimHoliday

dimNYCLocations

Fhv

GreenCab

PredictedValues

vwFhvMarketShare

vwGrnCabMarketSh...

vwMarketShareBy...

vwPredictedValues

vwYelCabMarketSh...

weather

YellowCab

YellowCabTripsHol...

date

holidayName

Σ numTrips

Σ year

Axis

holidayName

Legend

Add data fields here

Value

numTrips

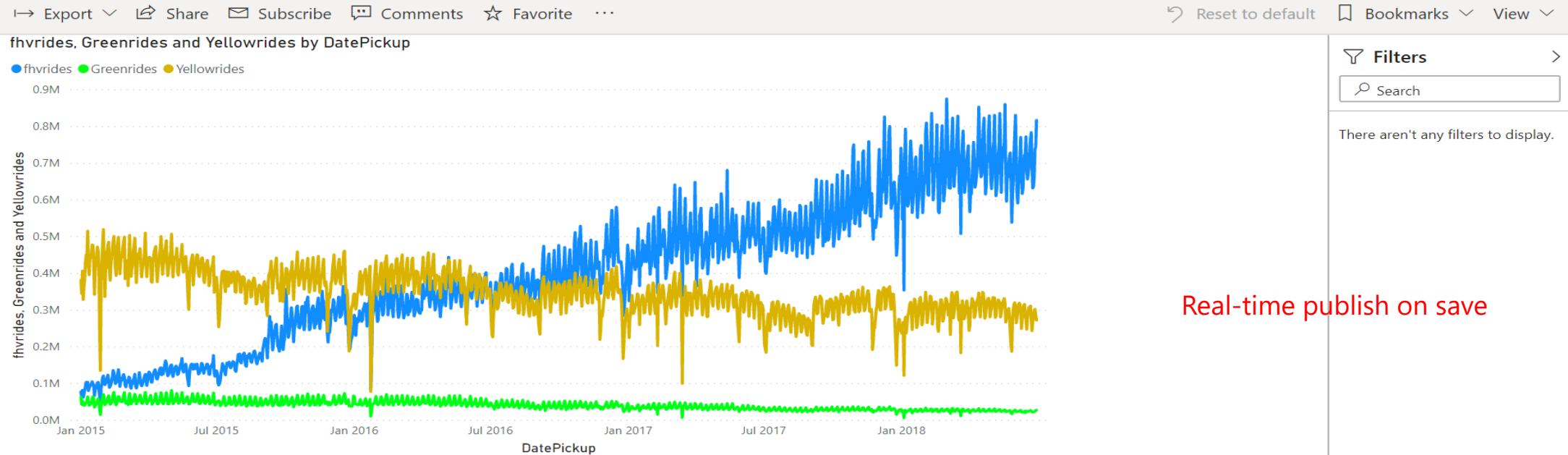
Tooltips

Add data fields here

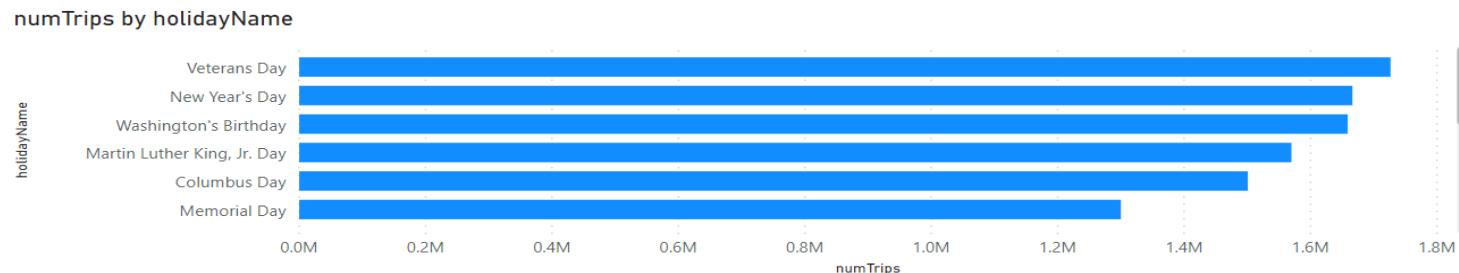
DRILLTHROUGH

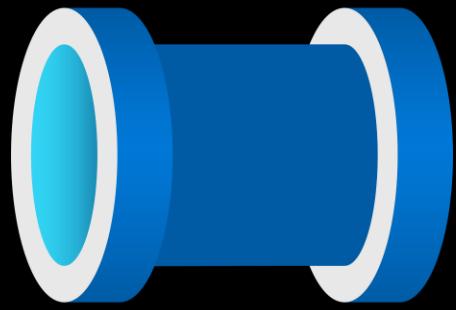
Cross-report

- Home
- Favorites
- Recent
- Apps
- Shared with me
- Workspaces
- SynapseNYTaxiInsi...



Real-time publish on save



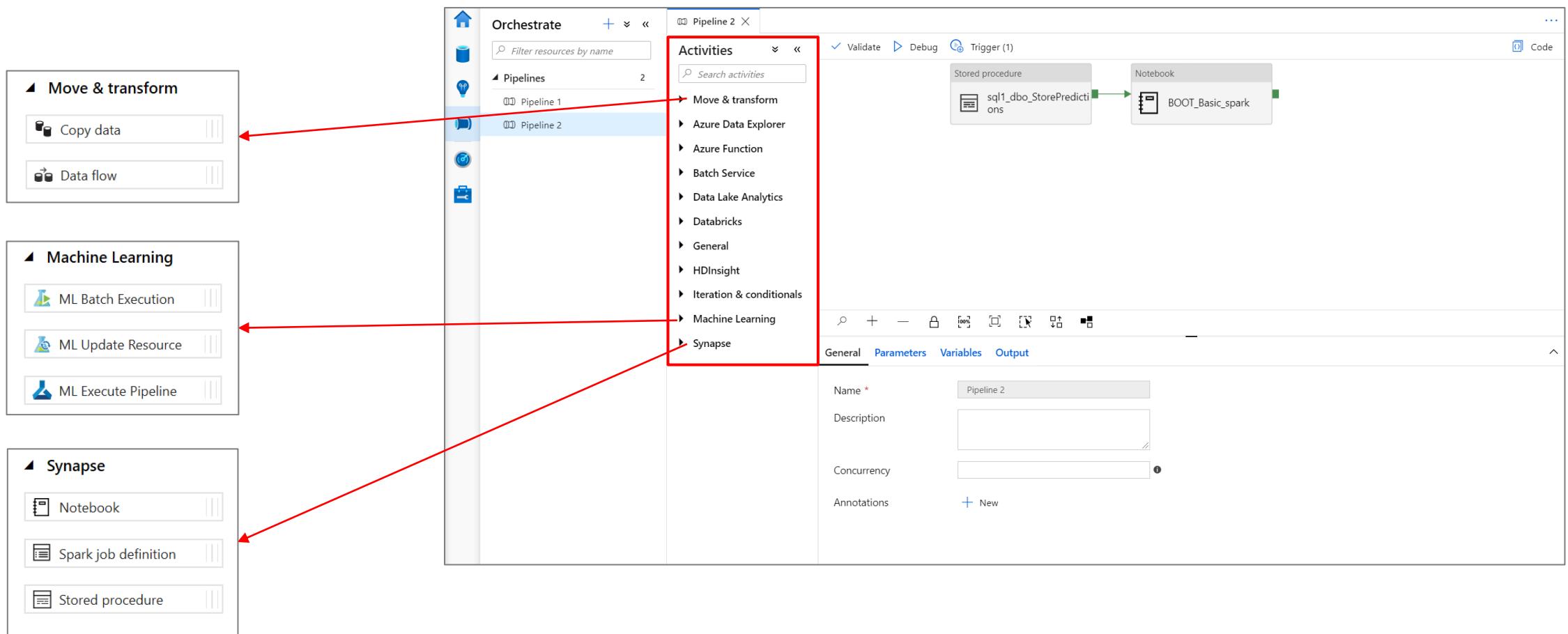


Synapse Studio Orchestrate hub

Orchestrate Hub

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

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





Synapse Studio Monitor hub

Monitor Hub

Overview

This feature provides ability to monitor orchestration, activities and compute resources.

The screenshot shows the Microsoft Azure Synapse Analytics Monitor Hub interface. The left sidebar has categories: Orchestration (selected), Pipeline runs, Trigger runs, Integration runtimes, Activities, Spark applications, Computes, and SQL Pools. The main area is titled "Pipeline runs" with filters: Time : 24 hours (default), Time zone : Pacific Time (US & Canada) (UT...), Runs : Latest runs, List (selected), Gantt, All status, Rerun, Cancel, Refresh, and Edit columns. It lists two pipeline runs:

Pipeline Name	Run Start	Duration	Triggered By	Status
Load Data to SQLDW	10/25/2019, 3:49:42 PM	00:10:55	Manual trigger	Succeeded
Copy Open Dataset	10/25/2019, 2:17:54 PM	00:14:12	Manual trigger	Succeeded

Monitoring Hub - Orchestration

Overview

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

Benefits

Track all/specific pipelines

Monitor pipeline run and activity run details

Find the root cause of pipeline failure or activity failure

Pipeline runs					
Time : Last week (10/24/2019 9:44 AM - 10/31/2019 9:44 AM)		Time zone : Pacific Time (US & Canada) (UT...)		Runs : Latest runs	
All status	Rerun	Cancel	Refresh	Edit columns	
<input type="checkbox"/> PIPELINE NAME	RUN START		DURATION	TRIGGERED BY	STATUS
Load Data to SQLDW	10/25/2019, 3:49:42 PM		00:10:55	Manual trigger	✓ Succeeded
Copy Open Dataset	10/25/2019, 2:17:54 PM		00:14:12	Manual trigger	✓ Succeeded
Pipeline 1	10/24/2019, 1:23:43 PM		00:00:08	Manual trigger	✓ Succeeded

Monitoring Hub - Spark applications

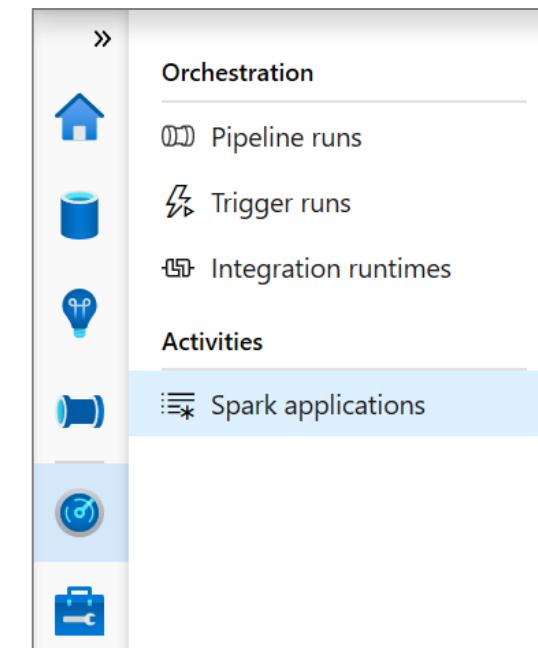
Overview

Monitor Spark pools, Spark applications for the progress and status of activities

Benefits

Monitor Spark pools for the status as paused, active, resume, scaling and upgrading

Track the usage of resources



Spark applications						
Submit time : 24 hours (default) (10/30/2019 9:52 AM - 10/31/2019 9:52 AM)		Time zone : Pacific Time (US & Canada) (UT...		List	Chart	
All types	Cancel	Refresh	Edit columns			
APPLICATION NAME	SUBMITTER	SUBMIT TIME	STATUS	POOL	TYPE	
Synapse_prlang-syntax...	prlangad@microsoft.com	10/30/2019 1:21 PM	Cancelled	prlang-syntaxcheck	Spark session	
Synapse_prlSpark_1572...	prlangad@microsoft.com	10/30/2019 1:06 PM	Cancelled	prlSpark	Spark session	



Synapse Studio Manage hub

Manage Hub

Overview

This feature provides ability to manage Linked Services, Orchestration and Security.

The screenshot shows the Microsoft Azure Synapse Analytics Studio Manage Hub. The left sidebar contains navigation links: External connections, **Linked services** (selected), Orchestration, Triggers, Integration runtimes, Security, and Access control. The top bar includes buttons for Publish all, Validate all, Refresh, Discard all, and user information (prlangad@microsoft.com). The main area is titled "Linked services" with a sub-instruction: "Linked services are much like connection strings, which define the connection information needed for Arcadia to connect to external resources." A "New" button is available. A search bar at the top right says "Search to filter items...". The table lists ten linked services:

NAME ↑↓	TYPE ↑↓	ANNOTATIONS ↑↓
ADLSG2OpenDataSetSink	Azure Data Lake Storage Gen2	
AzureBlobStorage1	Azure Blob Storage	
AzureDataLakeStorage1	Azure Data Lake Storage Gen2	
AzureDataLakeStorage2Source	Azure Data Lake Storage Gen2	
AzureOpenDataset	Azure Blob Storage	
AzureOpenDataSet2	Azure Blob Storage	
AzureSqlDW1	Azure Synapse Analytics (formerly SQL DW)	
AzureSynapseAnalytics1	Azure Synapse Analytics (formerly SQL DW)	
AzureSynapseAnalytics2	Azure Synapse Analytics (formerly SQL DW)	
PowerBIWorkspace1	Power BI	

Manage – Linked services

Overview

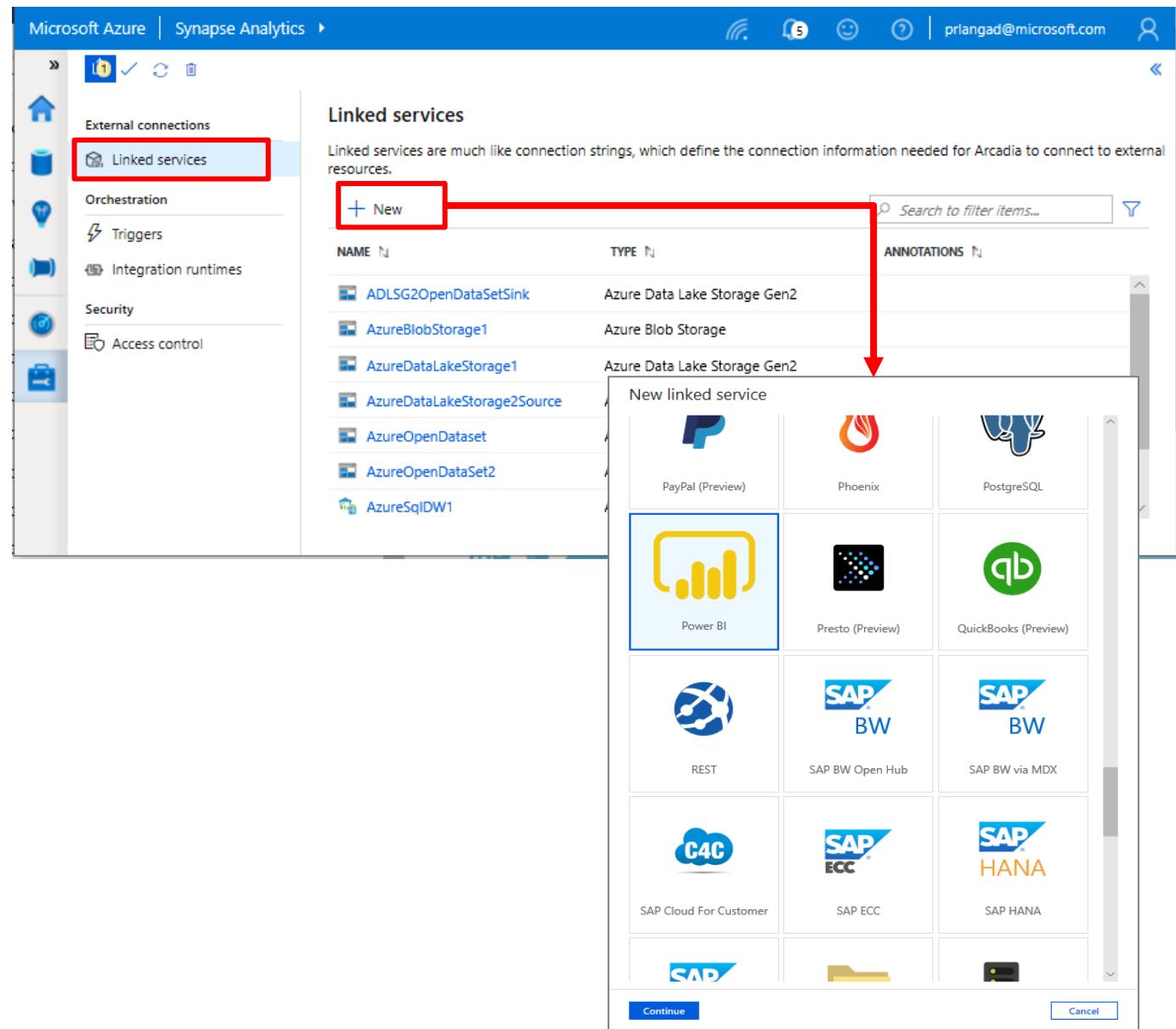
It defines the connection information needed to connect to external resources.

Benefits

Offers pre-build 90+ connectors

Easy cross platform data migration

Represents data store or compute resources



Manage – Access Control

Overview

It provides access control management to workspace resources and artifacts for admin and users

Benefits

Share workspace with the team

Increases productivity

Manage permissions on code artifacts and Spark pools

Microsoft Azure | Synapse Analytics > prlangad

External connections
Linked services
Orchestration
Triggers
Integration runtimes
Access control

Add admin Refresh Remove

Showing 1 of 1 items

NAME	PERMISSIONS
Priyanka Langade Priyanka.Langade@microsoft.com	Full permissions on code artifacts and Spark pools.

Add admin

An admin has full control over code artifacts, can attach to Spark pools, and can schedule pipelines. Permissions to Storage accounts and SQL pool databases are managed on the resources directly. [Learn more](#)

* Select user

Search by name or email address

Selected individual, groups or apps
No individual, groups, or apps selected.

Apply **Cancel**

Manage – Triggers

Overview

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

Benefits

Create and manage

- Schedule trigger
- Tumbling window trigger
- Event trigger

Control pipeline execution

The screenshot shows the Azure Synapse Analytics portal with the 'Triggers' section selected. A red box highlights the '+ New' button, which is also highlighted by a red arrow pointing to a detailed configuration dialog box titled 'New trigger'. The dialog box contains fields for Name (set to 'Trigger 2'), Description, Type (set to 'Schedule'), Start Date (UTC) (set to 10/29/2019 9:46 PM), Recurrence (Every 1 Minute(s)), End (No End), Annotations, and Activated (No). The main interface shows two existing triggers: 'CopyParquetDataTrigger' (Started, Schedule) and 'Trigger 1' (Stopped, Schedule).

NAME	TYPE	STATUS	NUMBER OF PIPELINES	ANNOTATIONS
* CopyParquetDataTrigger	Schedule	Started	1	
* Trigger 1	Schedule	Stopped	0	

Manage – Integration runtimes

Overview

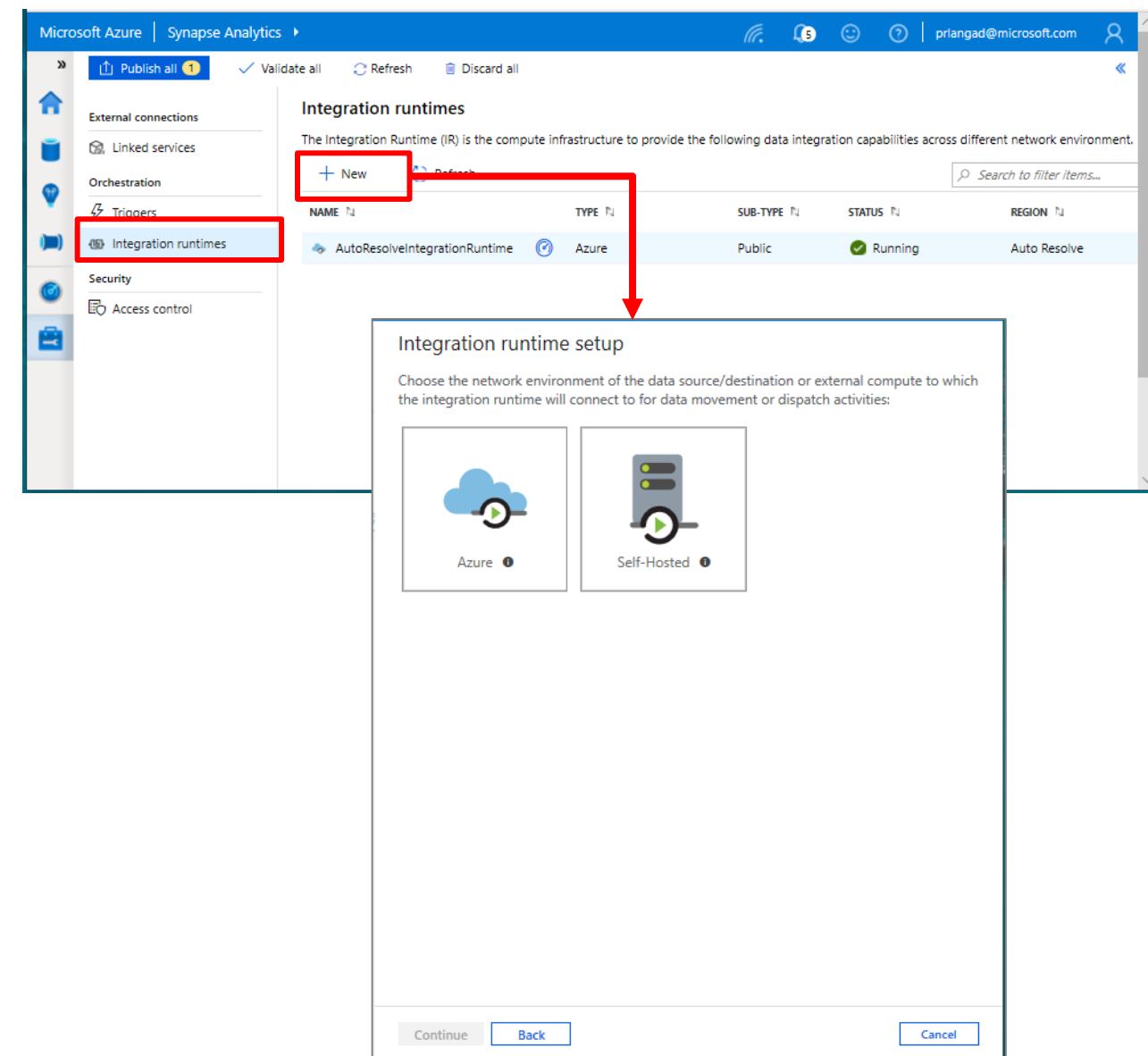
Integration runtimes are the compute infrastructure used by Pipelines to provide the data integration capabilities across different network environments. An integration runtime provides the bridge between the activity and linked services.

Benefits

Offers Azure Integration Runtime or Self-Hosted Integration Runtime

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

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



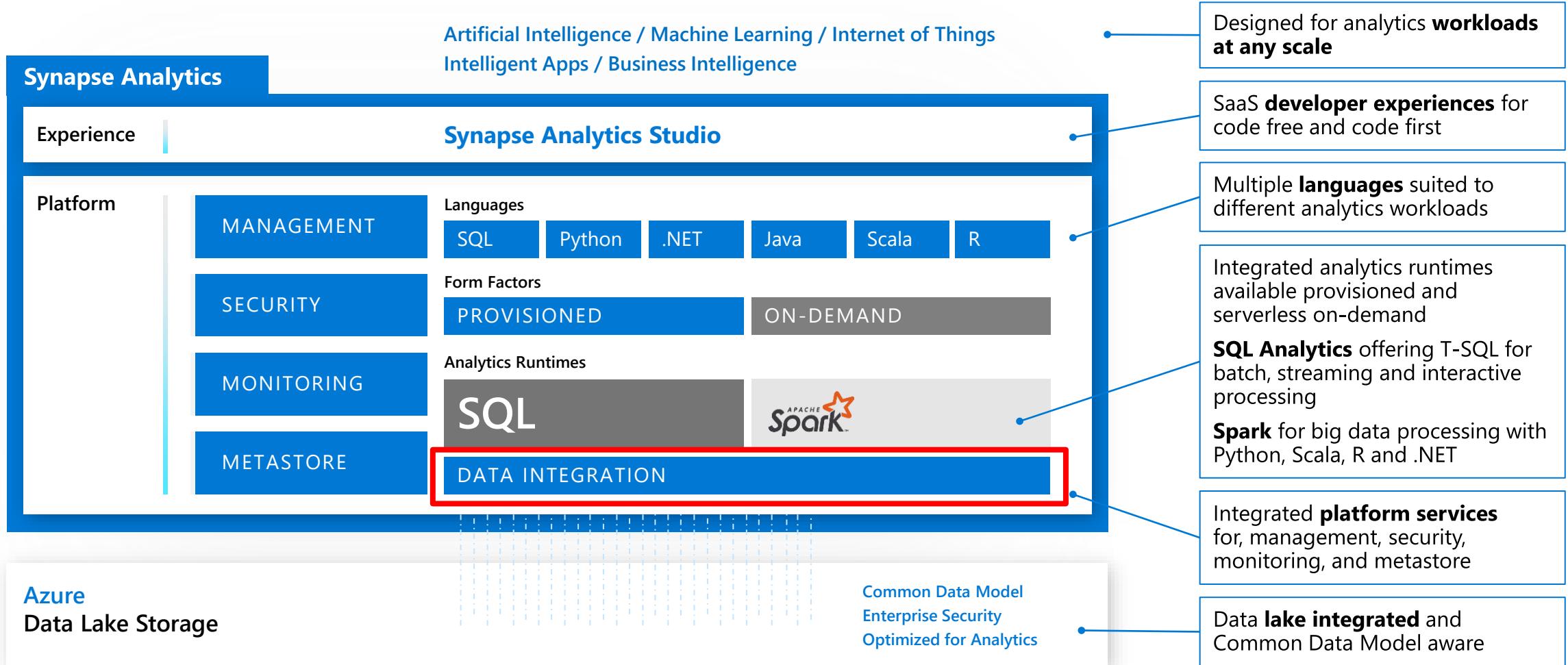


Azure Synapse Analytics

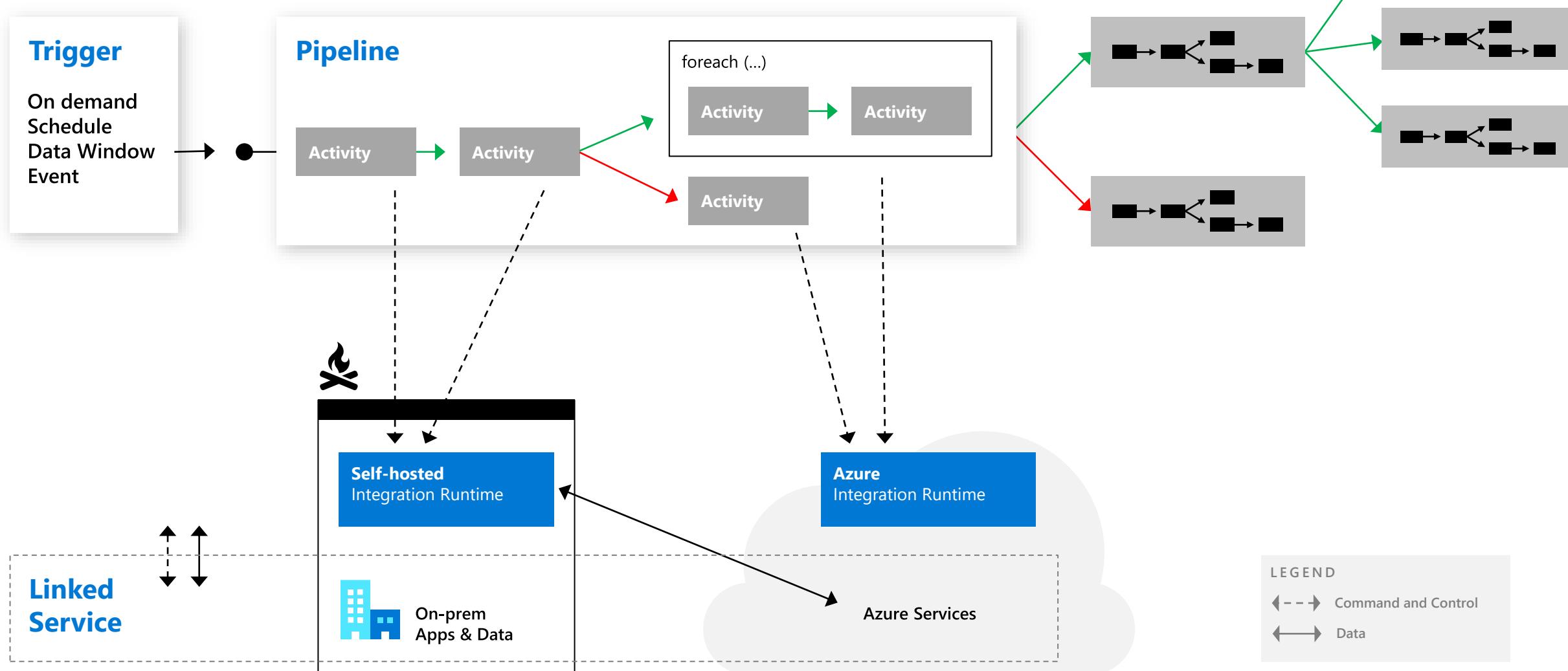
Data Integration

Azure Synapse Analytics

Limitless analytics service with unmatched time to insight



Orchestration @ Scale



Data Movement

Scalable

per job elasticity

Up to 4 GB/s

Simple

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

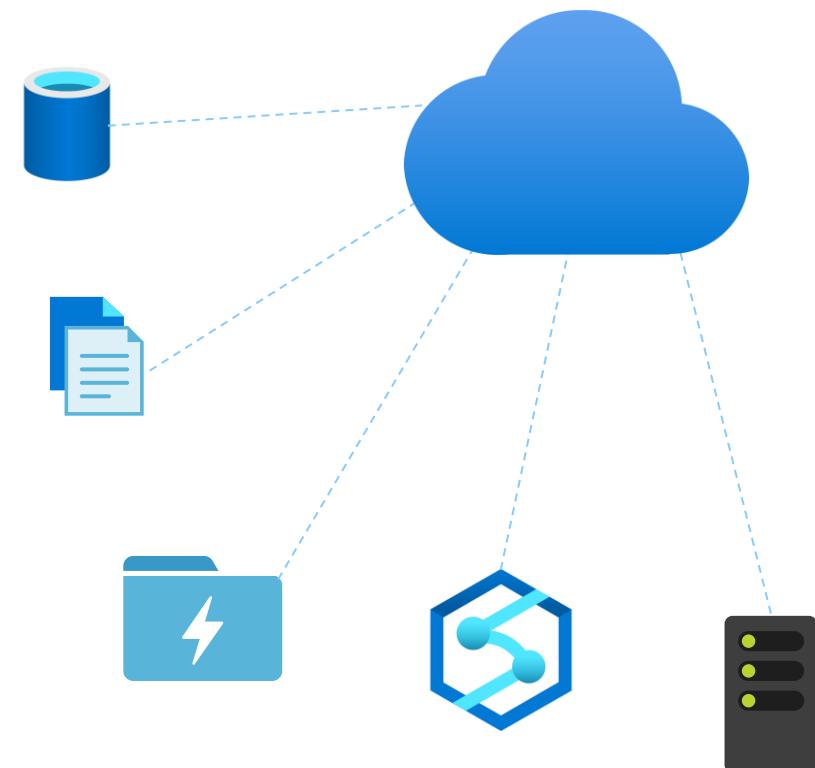
Serverless, no infrastructure to manage

Access all your data

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

Data Movement as a Service: 25 points of presence worldwide

Self-hostable Integration Runtime for hybrid movement

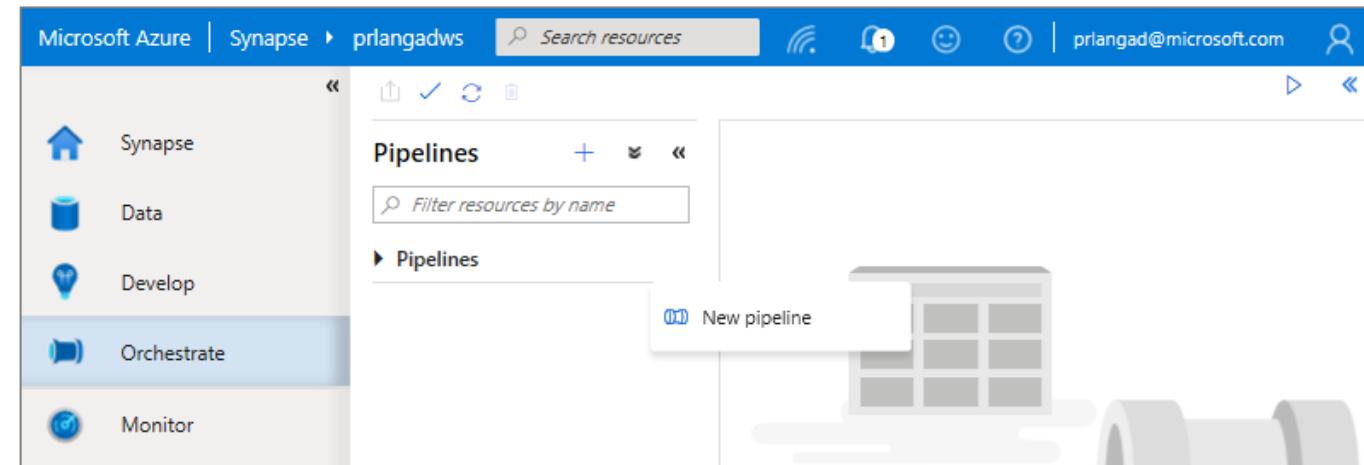


90+ Connectors out of the box

Pipelines

Overview

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



Benefits

Supports common loading patterns

Fully parallel loading into data lake or SQL tables

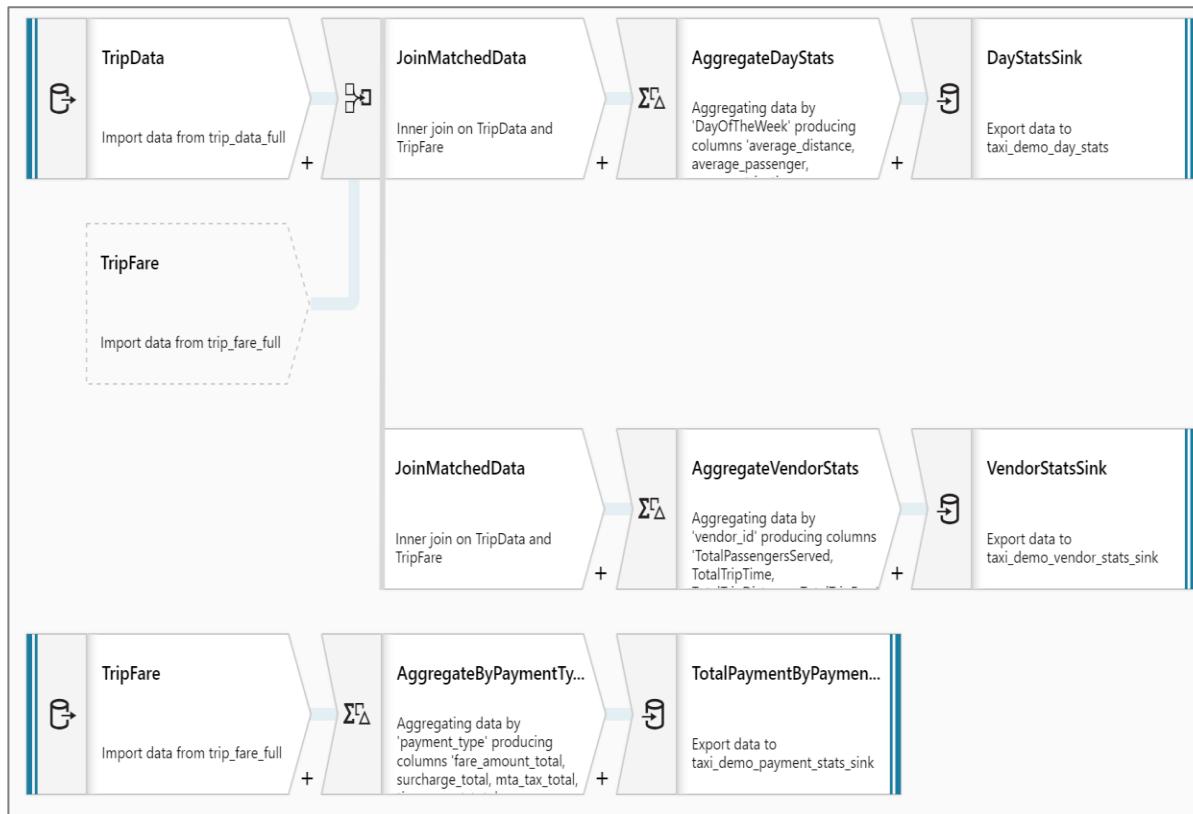
Graphical development experience

A screenshot of the Microsoft Azure Synapse Pipelines interface showing the configuration of a 'Load Data to SQLDW' activity. The left sidebar shows 'Orchestrate' selected. The main area shows a pipeline named 'Pipeline 1' with a 'Copy data' activity. The configuration pane shows the 'Source dataset' as 'ADLSGen2' and the 'Sink' as 'Azure Synapse Analytics Managed Instance'. On the right, there is a grid of 'New dataset' options including Azure Cosmos DB, Azure Data Explorer, Azure Data Lake Storage Gen1, Azure Data Lake Storage Gen2, Azure Database for MariaDB, Azure Database for MySQL, Azure Database for PostgreSQL, Azure File Storage, Azure SQL Database, Azure SQL Database Managed Instance, Azure Synapse Analytics (formerly SQL DW), and Azure Table Storage.

Prep & Transform Data

Mapping Dataflow

Code free data transformation @scale



Wrangling Dataflow

Code free data preparation @scale

CustId	FirstName	LastName	City	ZIP	Email	State	BasePay
1	'Harry'	'Potter'	'Bellevue'	'98004'	'harryk@fabrikam.com'	'WA'	90000
2	'Harry'	'Potter'	'Bellevue'	'98004'	'harryk@fabrikam.com'	'WA'	90000
3	'Hermione'	'Granger'	'Wilmington'	'19801'	'hermione@fabrikam.com'	'DE'	100000
4	'Hermione'	'Granger'	'Wilmington'	'19801'	'gamalloy@fabrikam.com'	'DE'	100000
5	'Lord'	'Voldemort'	'Billings'	'59115'	'lordc@fabrikam.com'	'MT'	110000
6	'Albus'	'Dumbledore'	'Newyork'	'12345'	'albusd@fabrikam.com'	'NY'	120000
7	'Severus'	'Snape'	'Columbus'	'56789'	'severus@fabrikam.com'	'OH'	130000
8	'Draco'	'Malfoy'	'Houston'	'91019'	'dracoh@fabrikam.com'	'TX'	140000
9	'Dobby'	'Elf'	'Salt Lake Ci...	'11128'	'dobbyz@fabrikam.com'	'UT'	150000
10	'Ron'	'Weasley'	'Las Vegas'	'51527'	'ronag@fabrikam.com'	'NV'	160000
11	'Sirius'	'Black'	'Providence'	'61623'	'hblack@fabrikam.com'	'RI'	170000
12	'Luna'	'Lovegood'	'Kansas City'	'68692'	'lunal@fabrikam.com'	'MO'	180000
13	'Rubeus'	'Hagrid'	'Boston'	'98052'	'gammaloy@fabrikam.com'	'Malfoy'	190000
14	'Bellatrix'	'Lestrange'	'Los Angeles'	'78965'	'mlestrange@fabrikam.com'	'CA'	200000
15	'Ginny'	'Weasley'	'Redmond'	'98052'	'ginnyw@fabrikam.com'	'WA'	210000
16	'Neville'	'Longbottom'	'Bothell'	'98053'	'nevilea@fabrikam.com'	'WA'	220000
17	'Alastor'	'Moody'	'Renton'	'98054'	'albusd@fabrikam.com'	'WA'	230000
18	'Lucius'	'Malfoy'	'Bellevue'	'98004'	'luciusmalfoy@fabrikam.co...	'WA'	240000
19	'Cedric'	'Diggory'	'Seattle'	'98989'	'cedricp@fabrikam.com'	'WA'	250000
20	'Argus'	'Filch'	'Salt Lake Ci...	'11128'	'argusm@fabrikam.com'	'UT'	260000
21	'Lord'	'Voldemort'	'Billings'	'59115'	'lordc@fabrikam.com'	'MT'	110000
22	'Albus'	'Dumbledore'	'Newyork'	'12345'	'albusd@fabrikam.com'	'NY'	120000
23	'Severus'	'Snape'	'Columbus'	'56789'	'severus@fabrikam.com'	'OH'	130000
24	'Mason'	'Bettie'	'Bellmore'	'98900A'	'masonm@fabrikam.com'	'MA'	00000

Triggers

Overview

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

Data Integration offers 3 trigger types as –

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

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

The screenshot shows the Azure Synapse Analytics Orchestration Triggers interface. At the top, a modal dialog titled "New trigger" is open, prompting the user to "Choose a name for your trigger. This name can be updated at any time until it is published." The "Name" field contains "Trigger 1". Below it is a "Description" field which is currently empty. The "Type" section has three options: "Schedule" (selected), "Tumbling window", and "Event". Under "Schedule", the "Start Date (UTC)" is set to "10/30/2019 11:20 PM". The "Recurrence" section shows "Every 1 Minute(s)". The "End" section has two options: "No End" (selected) and "On Date". The "Annotations" section includes a "+ New" button. The "Activated" section has two options: "Yes" (selected) and "No". At the bottom of the dialog is an "OK" button. Below the dialog, the main interface shows a navigation bar with "External connections", "Linked services", "Orchestration", "Triggers" (which is selected and highlighted in blue), and "Integration runtimes", "Security", and "Access control". The "Triggers" section contains a table with the following data:

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

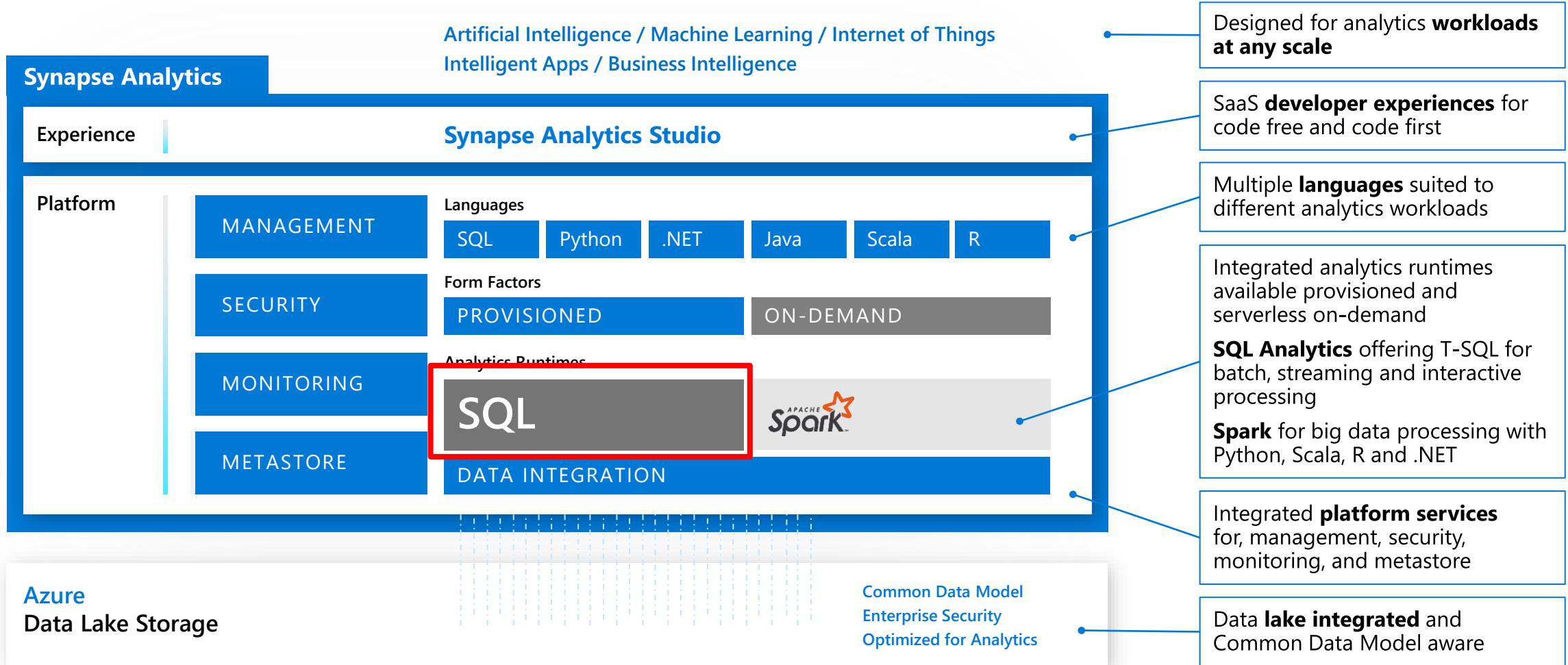


Azure Synapse Analytics

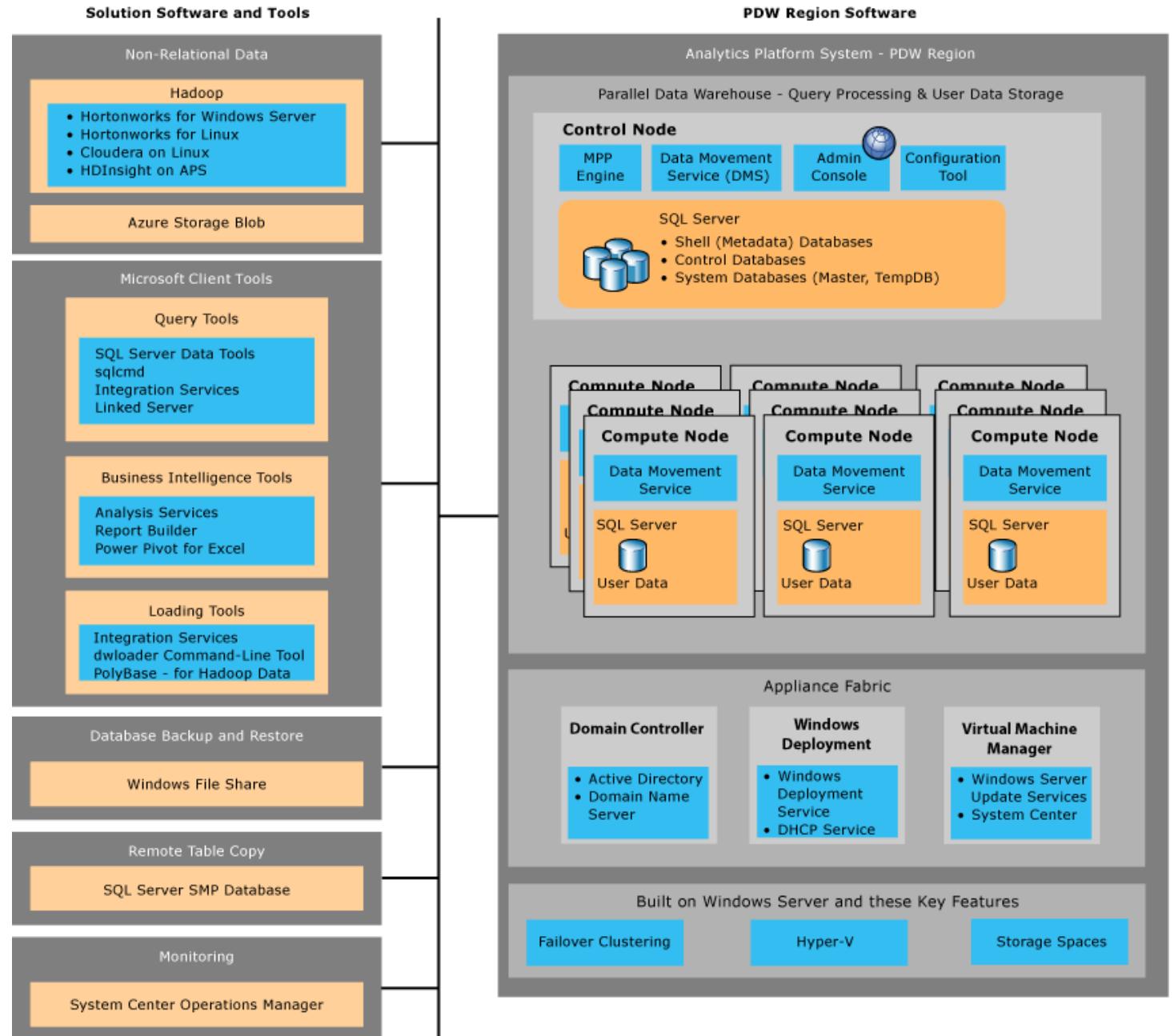
SQL Analytics

Azure Synapse Analytics

Limitless analytics service with unmatched time to insight

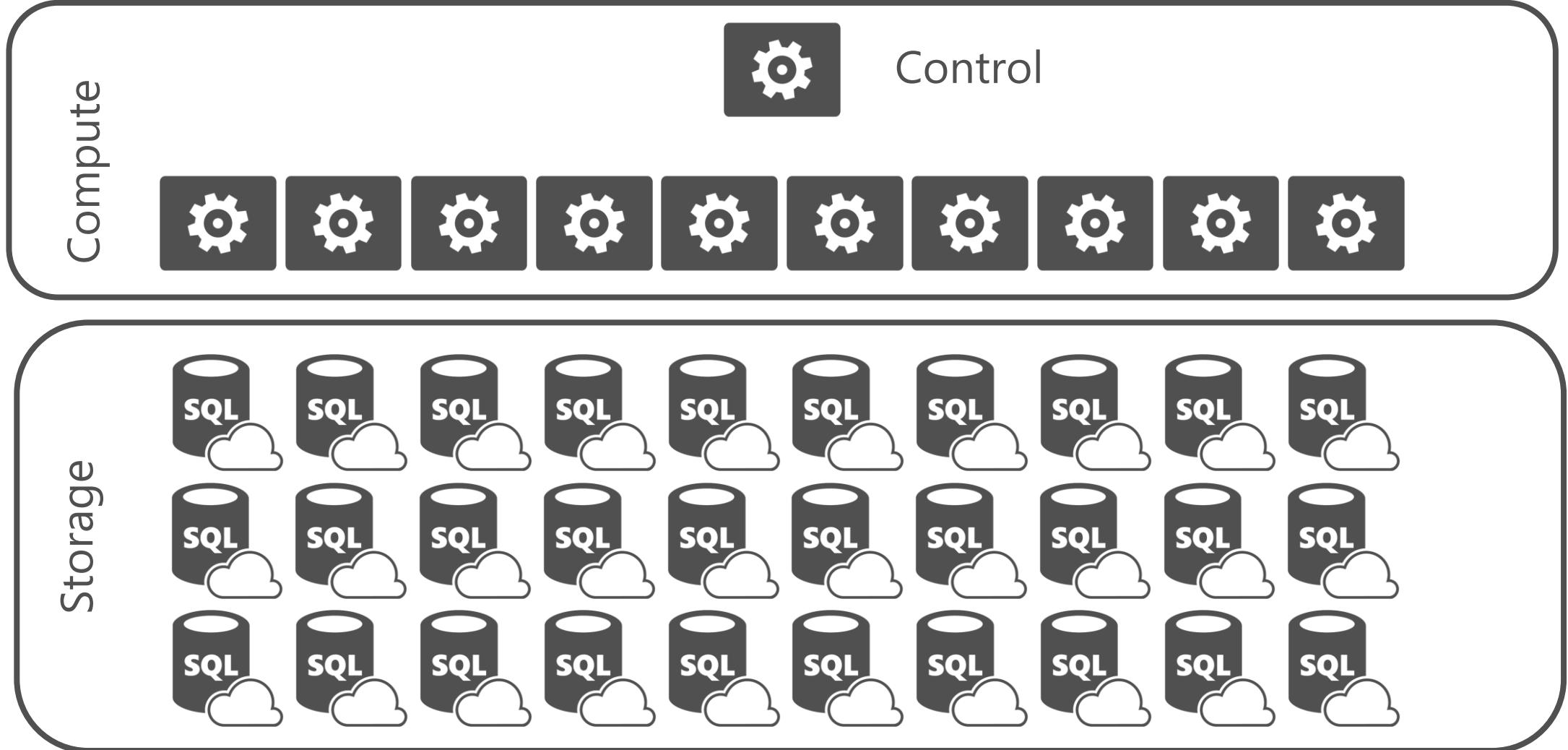


PDW (Parallel Data Warehouse) Architecture



Synapse Analytics Architectural overview

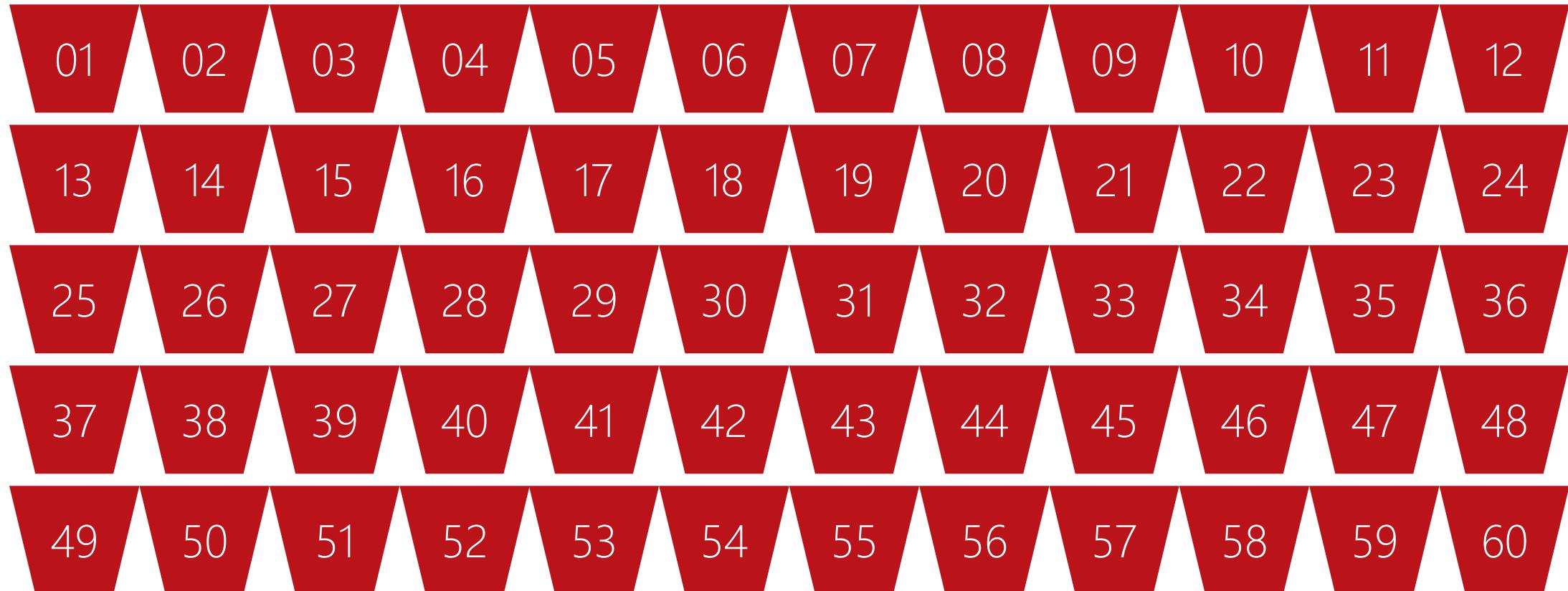
Logical overview



Scale operations

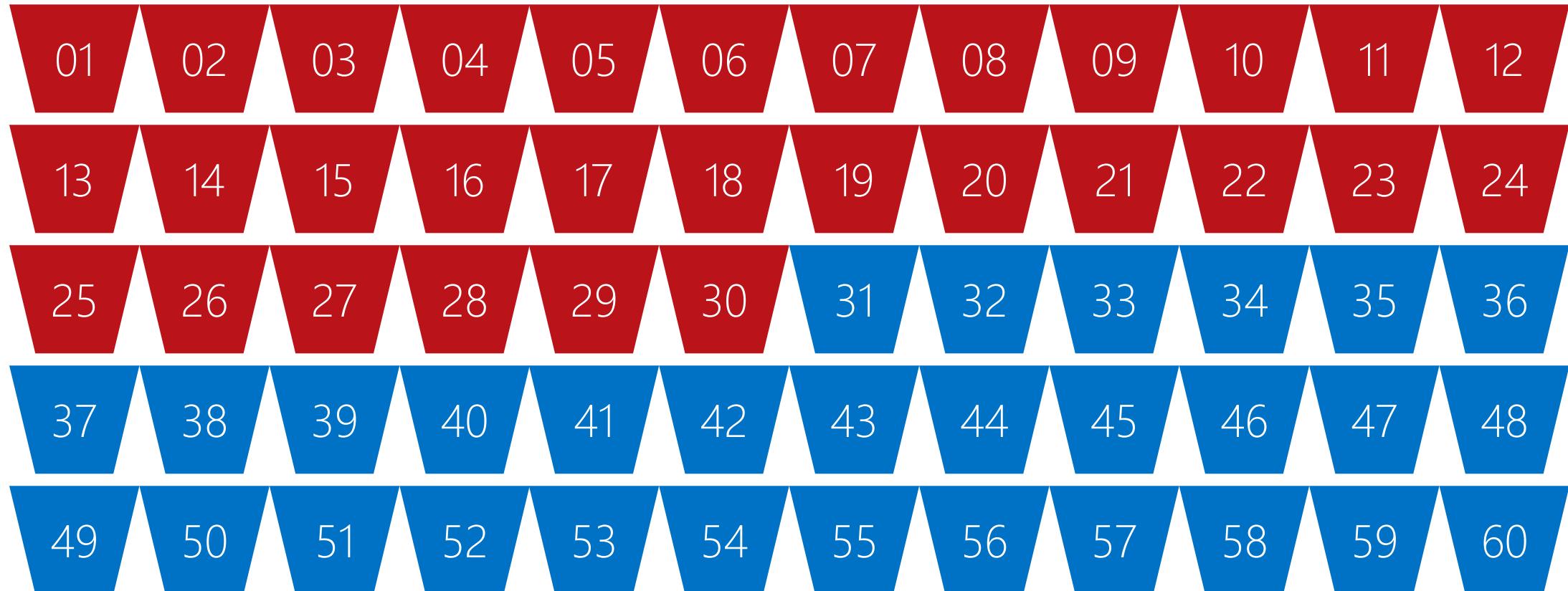
Mapping Compute in S QLDW

DW100



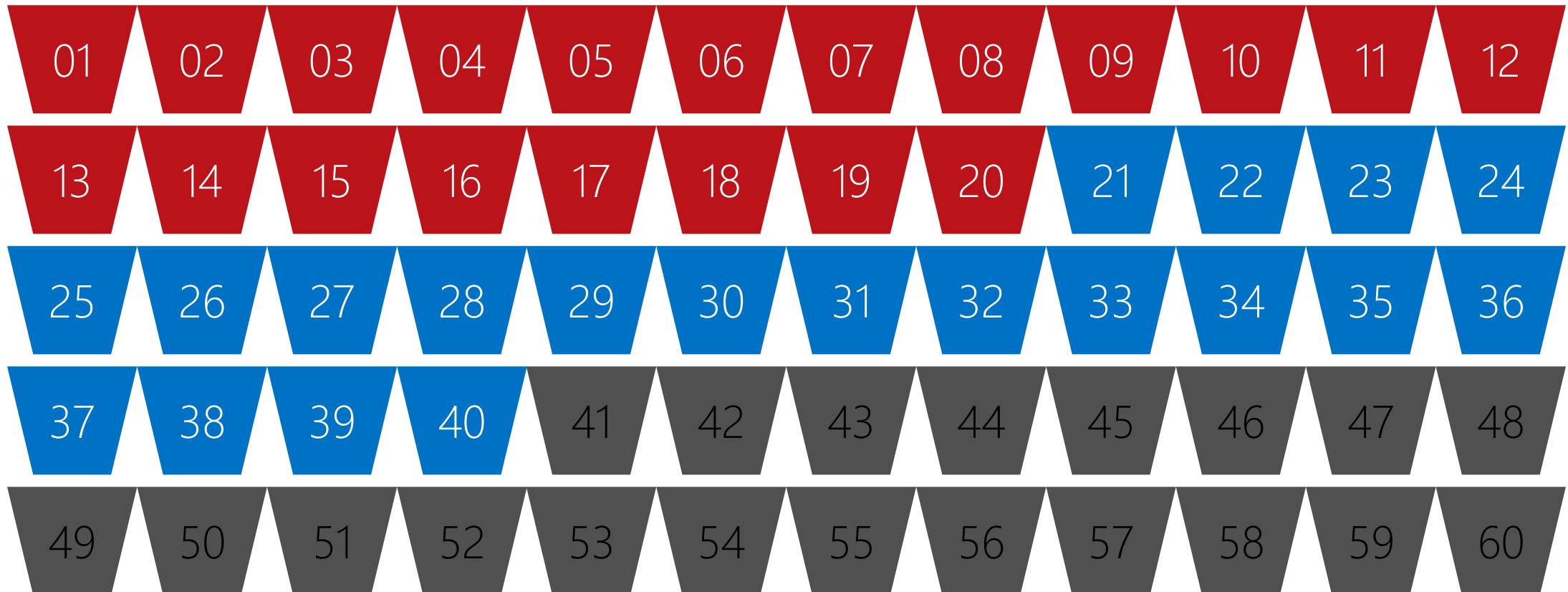
Mapping Compute in SQLDW

DW200



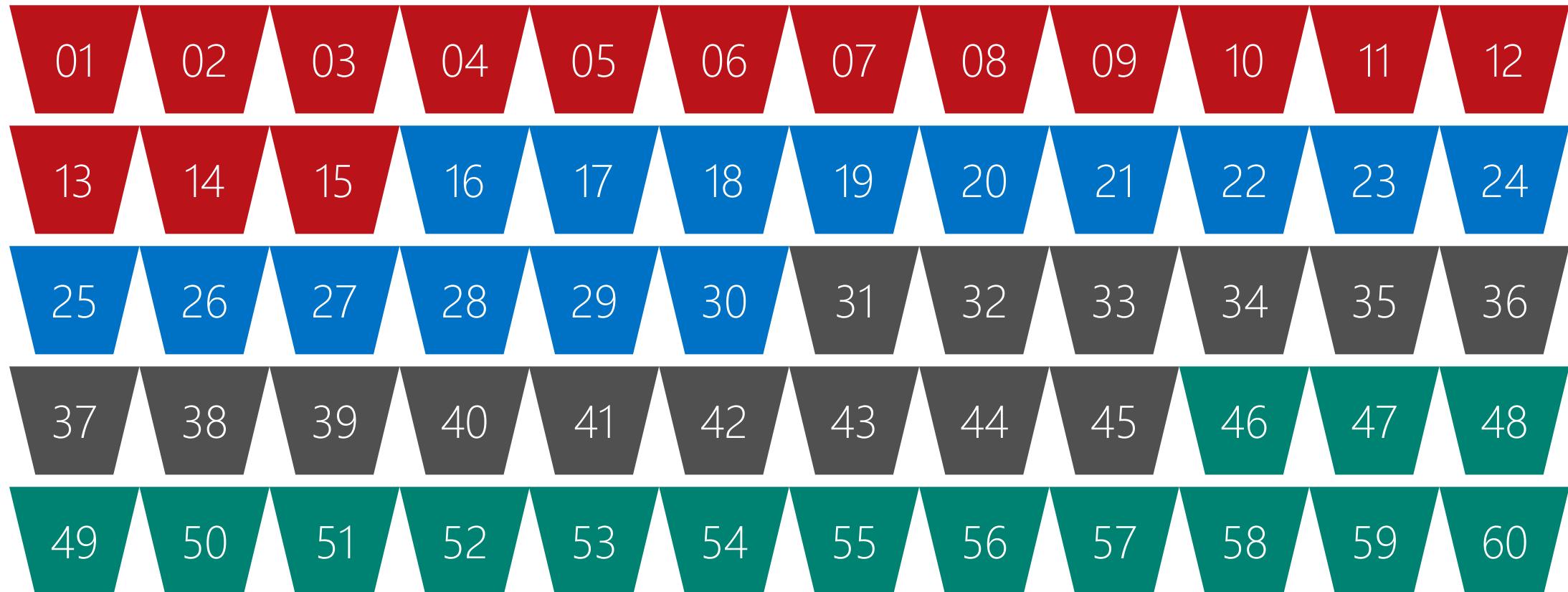
Mapping Compute in SQLDW

DW300



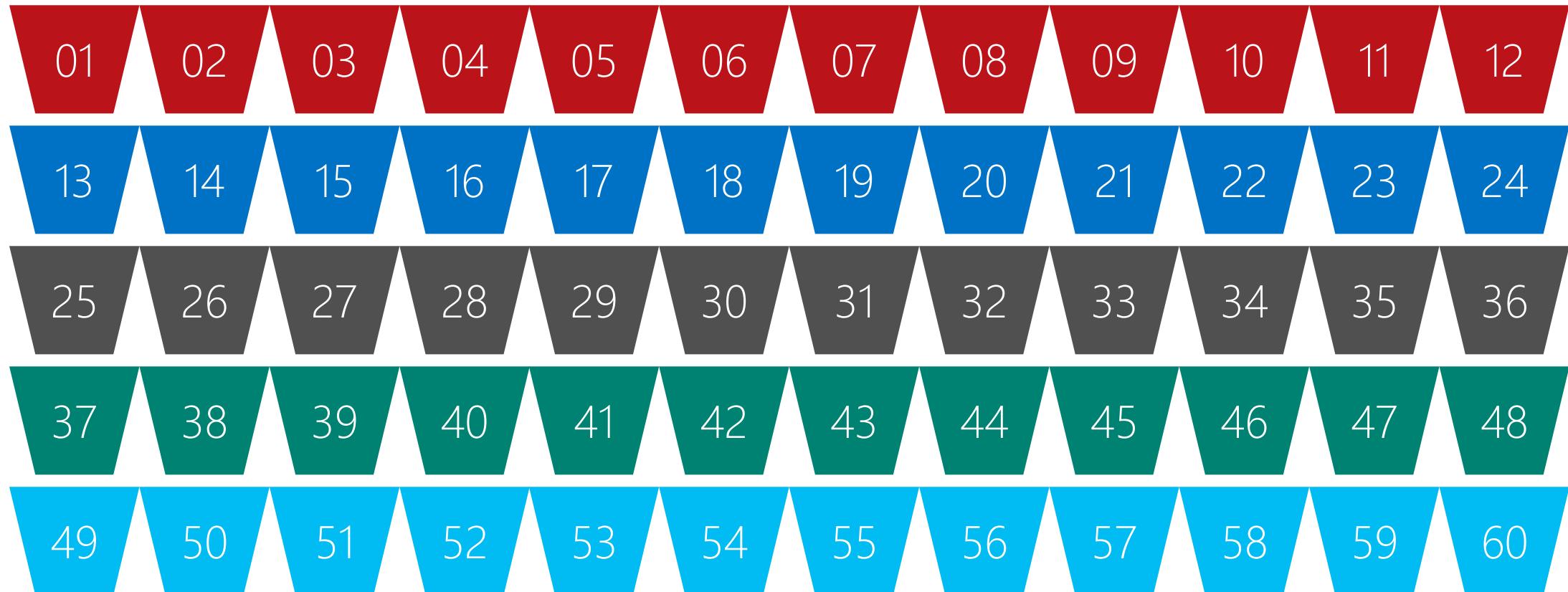
Mapping Compute in SQLDW

DW400



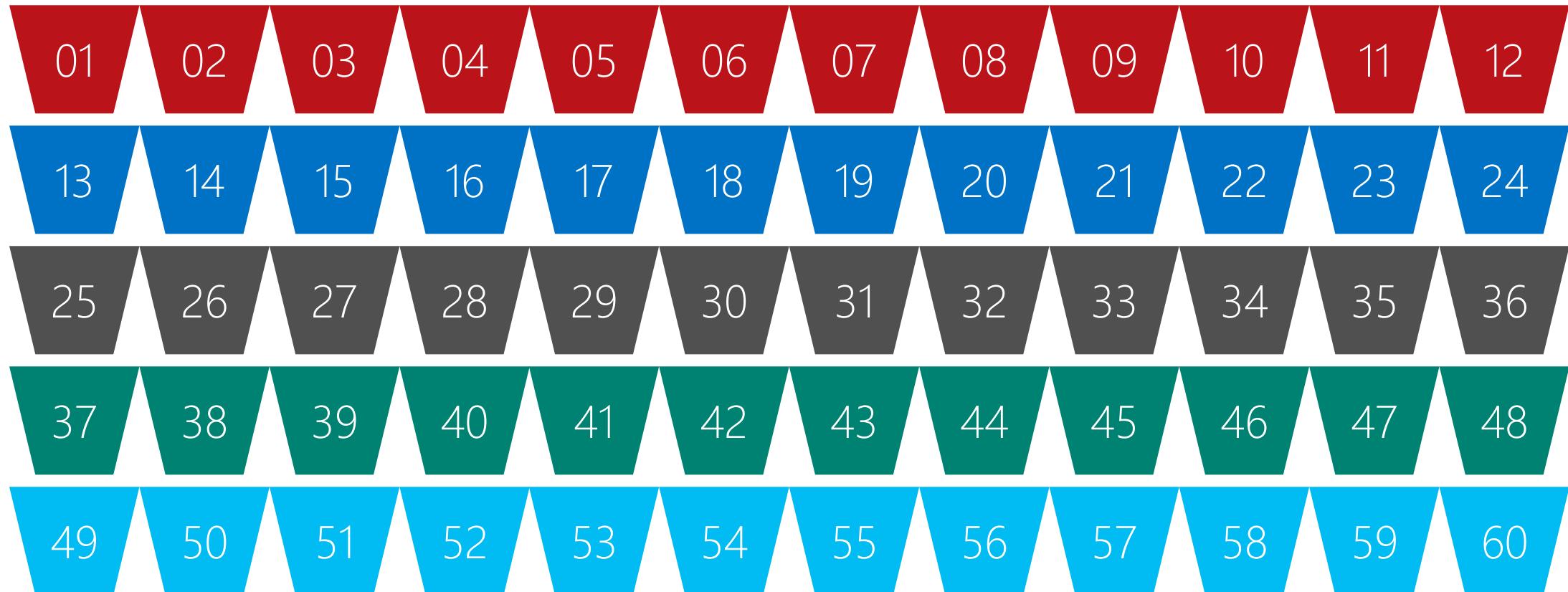
Mapping Compute in SQLDW

DW500



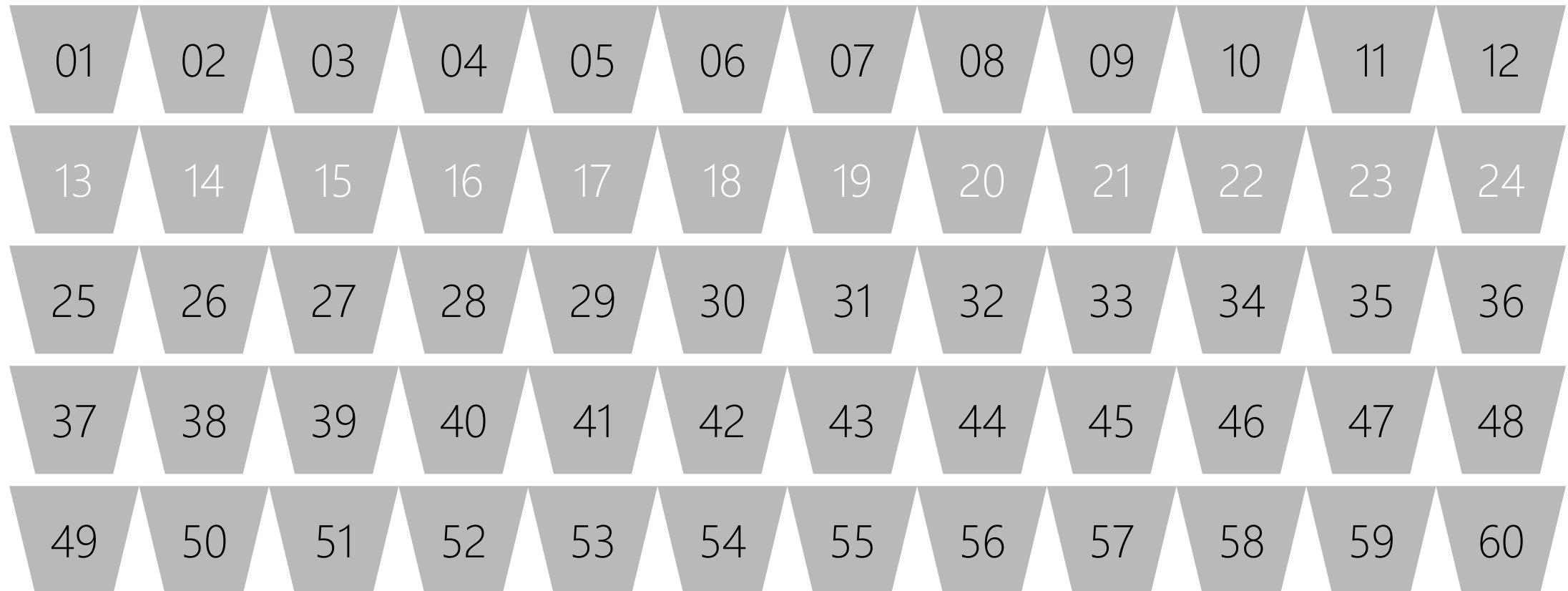
Pausing compute in SQLDW

DW500

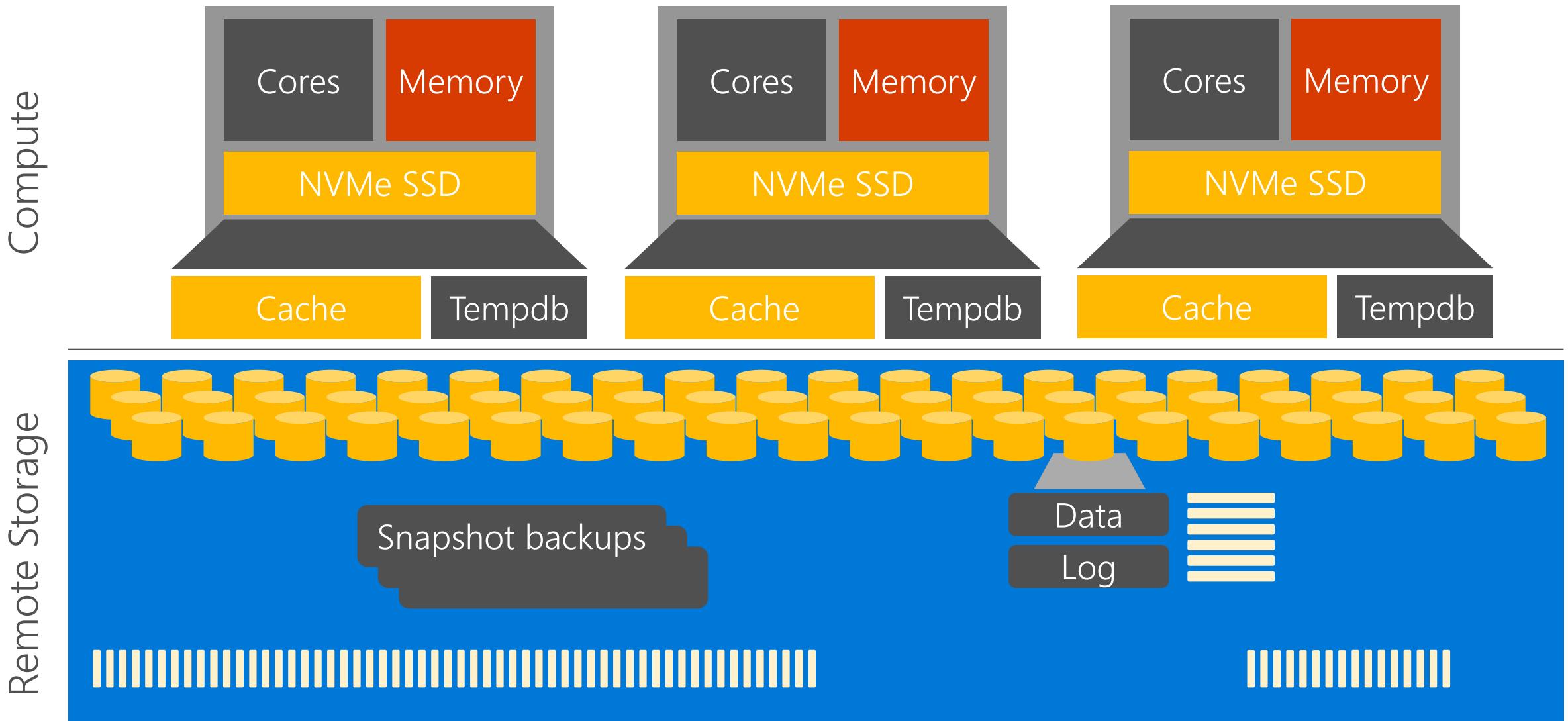


Resuming compute in SQLDW

DW500



TIERED STORAGE MODEL AUTOMATES DATA TEMPERATURE



Concurrency

Concurrency: queries



Memory & Concurrency Limits

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

Concurrency maximums for workload groups

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

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

Resource Class

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

Static resource classes

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

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

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

Dynamic resource classes

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

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

- smallrc
- mediumrc
- largerc
- xlargerc

The memory allocation for each resource class is as follows.

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

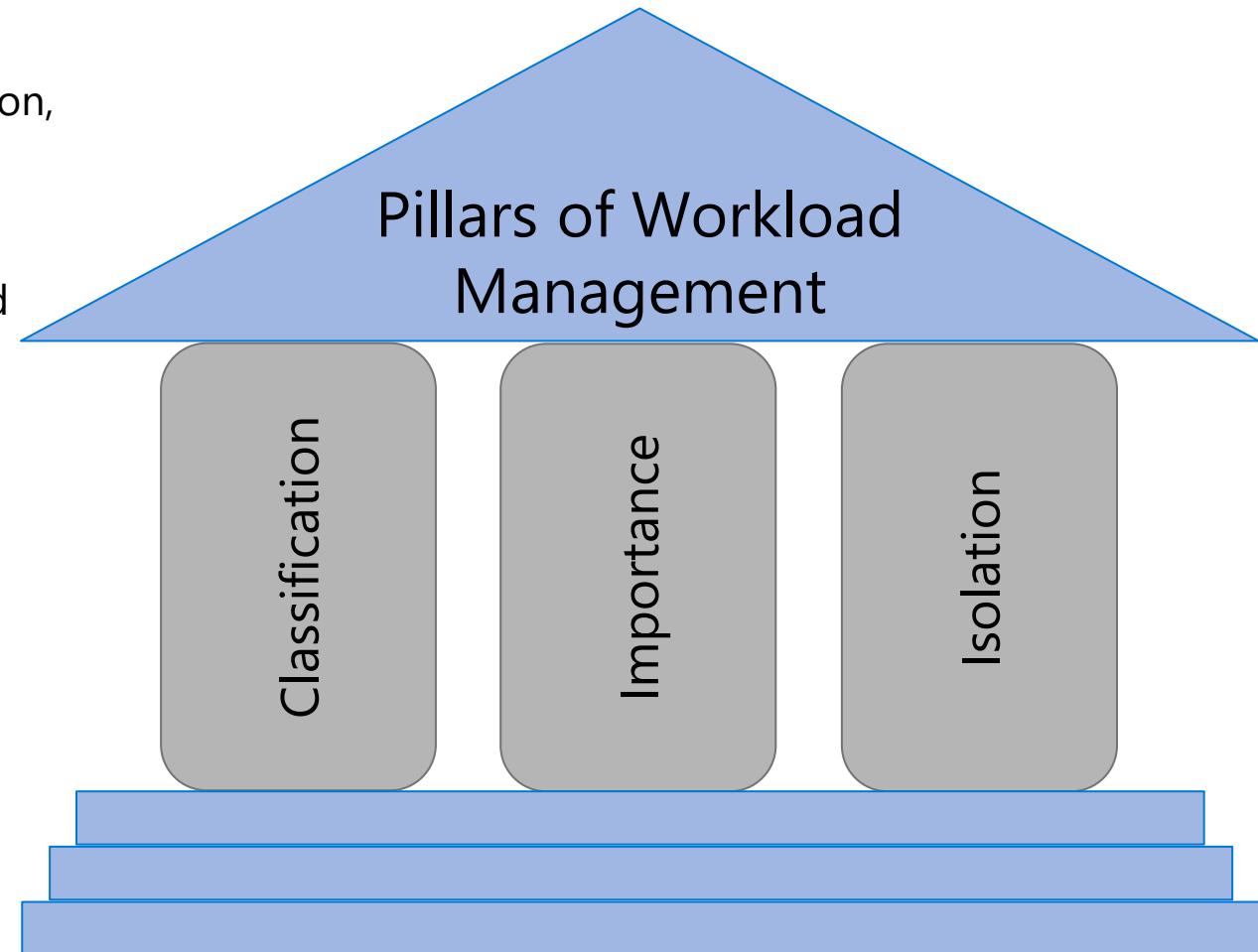
Workload Management

Overview

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

The three pillars of workload management are

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



Workload classification

Overview

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

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

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

Benefits

Map queries to both Resource Management and Workload Isolation concepts.

Monitoring DMVs

[sys.workload_management_workload_classifiers](#)

[sys.workload_management_workload_classifier_details](#)

Query DMVs to view details about all active workload classifiers.

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

WORKLOAD_GROUP: maps to an existing resource class

IMPORTANCE: specifies relative importance of request

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

Workload importance

Overview

Queries past the concurrency limit enter a FiFo queue

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

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

Example Video

State analysts have normal importance.

National analyst is assigned high importance.

State analyst queries execute in order of arrival

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

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



Azure Synapse
Analytics

.....
.....



State
Analyst



State
Analyst



State
Analyst



State
Analyst



State
Analyst

Workload Isolation

Overview

Allocate fixed resources to workload group.

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

Benefits

Reserve resources for a group of requests

Limit the amount of resources a group of requests can consume

Shared resources accessed based on importance level

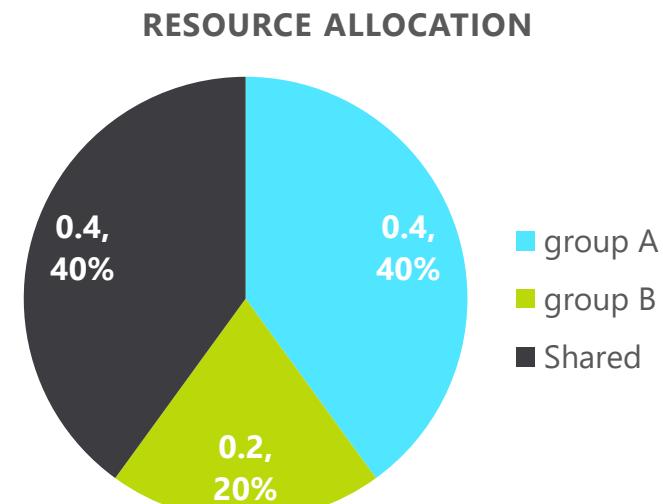
Set Query timeout value. Get DBAs out of the business of killing runaway queries

Monitoring DMVs

[sys.workload_management_workload_groups](#)

Query to view configured workload group.

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



User & Authentication

-- Master Database

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

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

-- Master Database

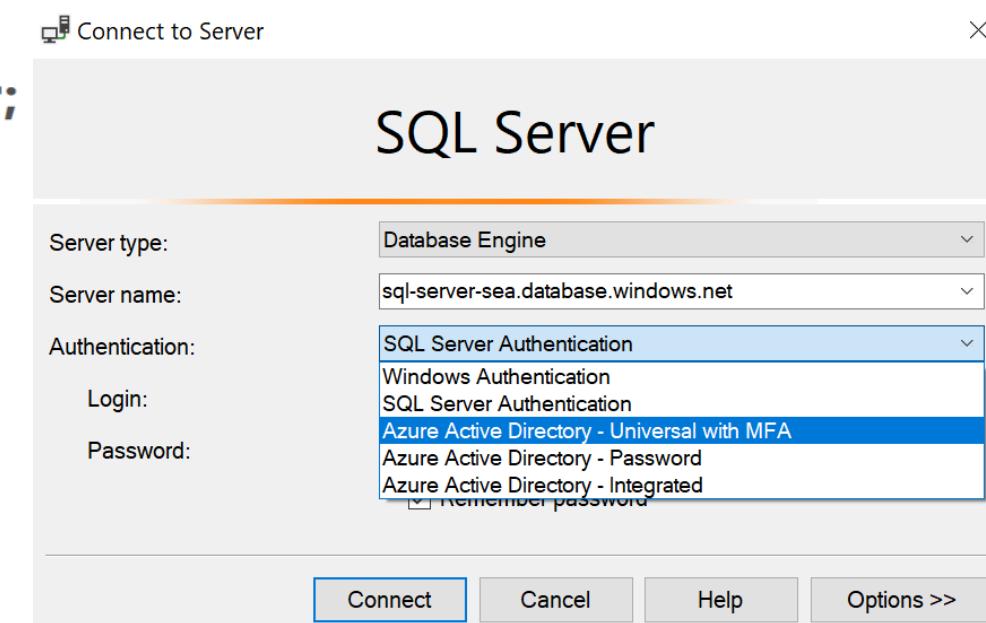
```
CREATE USER LoadingUser FOR LOGIN MedRCLLogin;
```

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

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

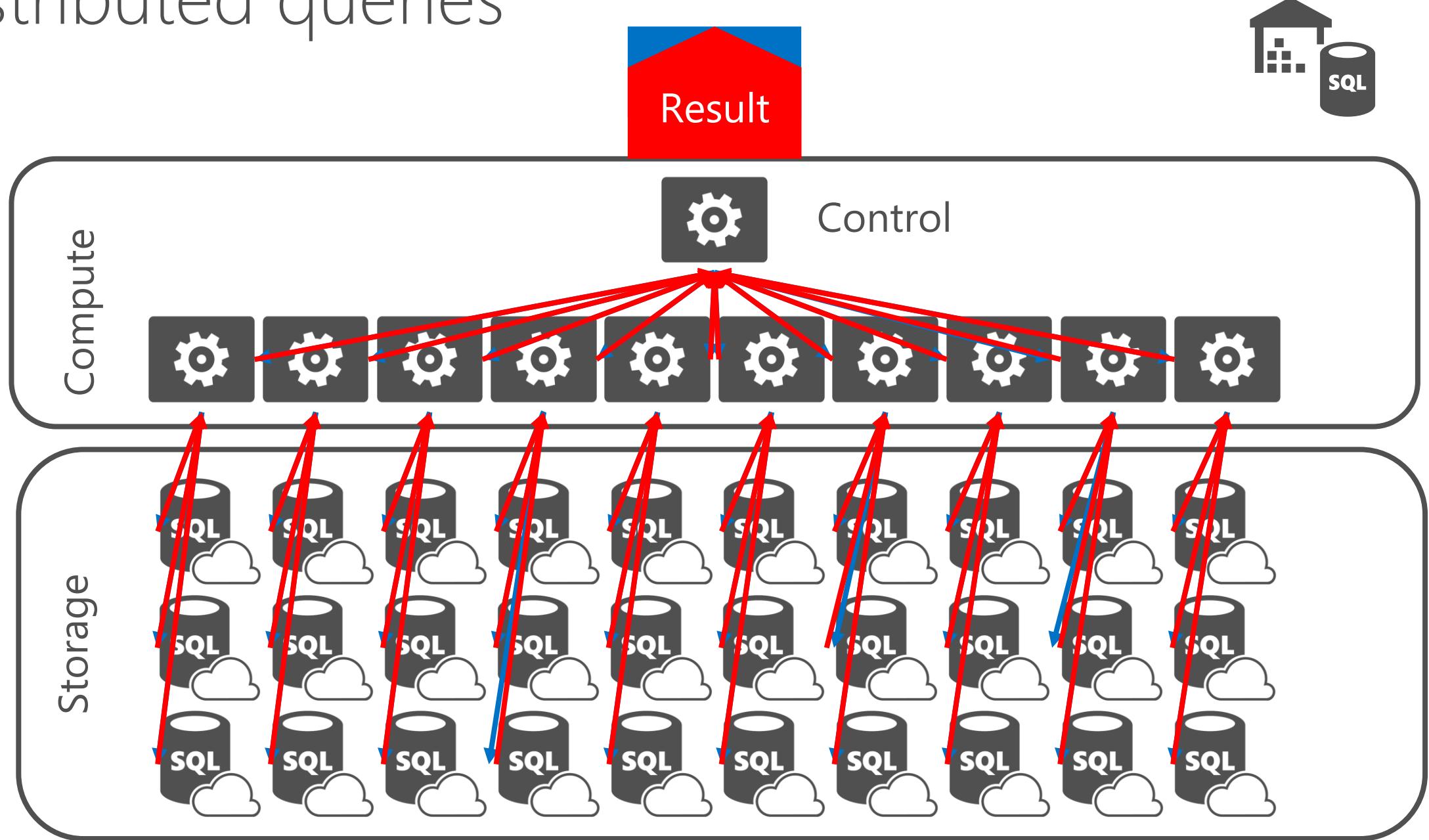
```
Create Login Mary@domainname.net From EXTERNAL PROVIDER;  

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



Tables & Distribution

Distributed queries



Simple example

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



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



Control

Compute

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



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



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

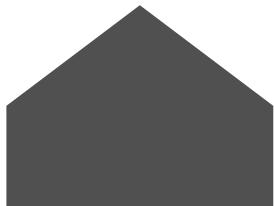


```
SELECT COUNT_BIG(*)  
FROM dbo.[FactInternetSales]  
;
```

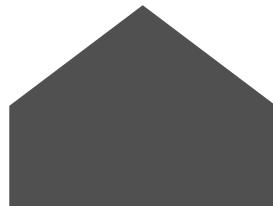


Creating tables

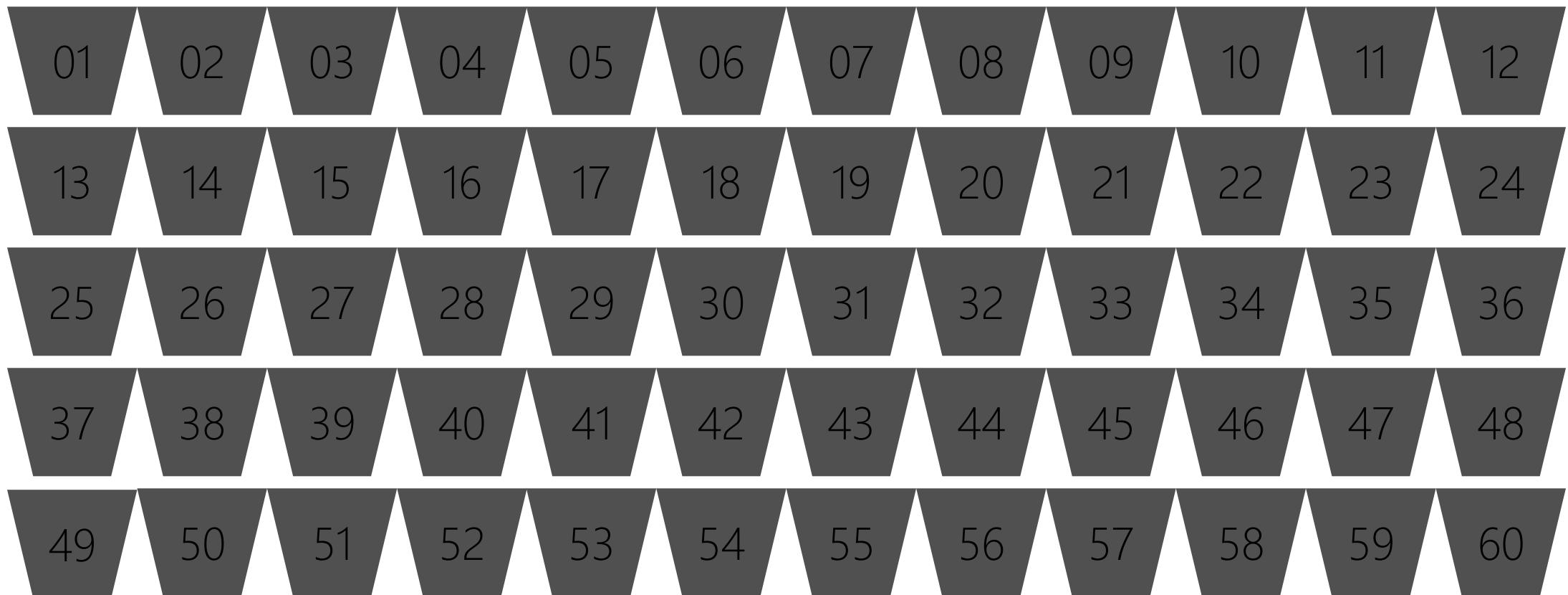
```
CREATE TABLE [build].[FactOnlineSales]
(
    [OnlineSalesKey]           int          NOT NULL
    , [DateKey]                datetime    NOT NULL
    , [StoreKey]               int          NOT NULL
    , [ProductKey]              int         NOT NULL
    , [PromotionKey]            int         NOT NULL
    , [CurrencyKey]             int         NOT NULL
    , [CustomerKey]             int         NOT NULL
    , [SalesOrderNumber]        nvarchar(20) NOT NULL
    , [SalesOrderLineNumber]   int          NULL
    , [SalesQuantity]            int         NOT NULL
    , [SalesAmount]              money       NOT NULL
)
WITH
(
    CLUSTERED COLUMNSTORE INDEX
    , DISTRIBUTION = ROUND_ROBIN
)
;
```



```
CREATE TABLE [build].[FactOnlineSales]
(
    [OnlineSalesKey]           int          NOT NULL
    , [DateKey]                datetime    NOT NULL
    , [StoreKey]               int          NOT NULL
    , [ProductKey]              int         NOT NULL
    , [PromotionKey]            int         NOT NULL
    , [CurrencyKey]             int         NOT NULL
    , [CustomerKey]             int         NOT NULL
    , [SalesOrderNumber]        nvarchar(20) NOT NULL
    , [SalesOrderLineNumber]   int          NULL
    , [SalesQuantity]            int         NOT NULL
    , [SalesAmount]              money       NOT NULL
)
WITH
(
    CLUSTERED COLUMNSTORE INDEX
    , DISTRIBUTION = HASH([ProductKey])
)
;
```

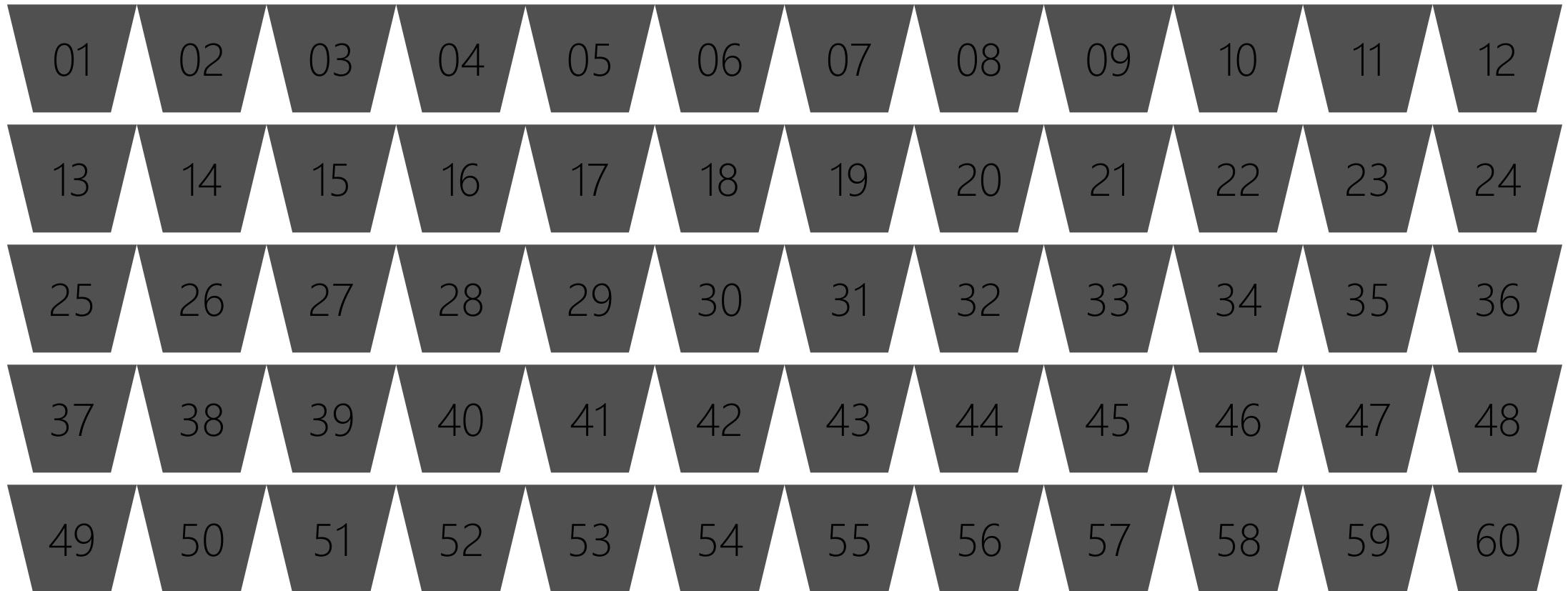


ROUND ROBIN DISTRIBUTION



HASH distribution

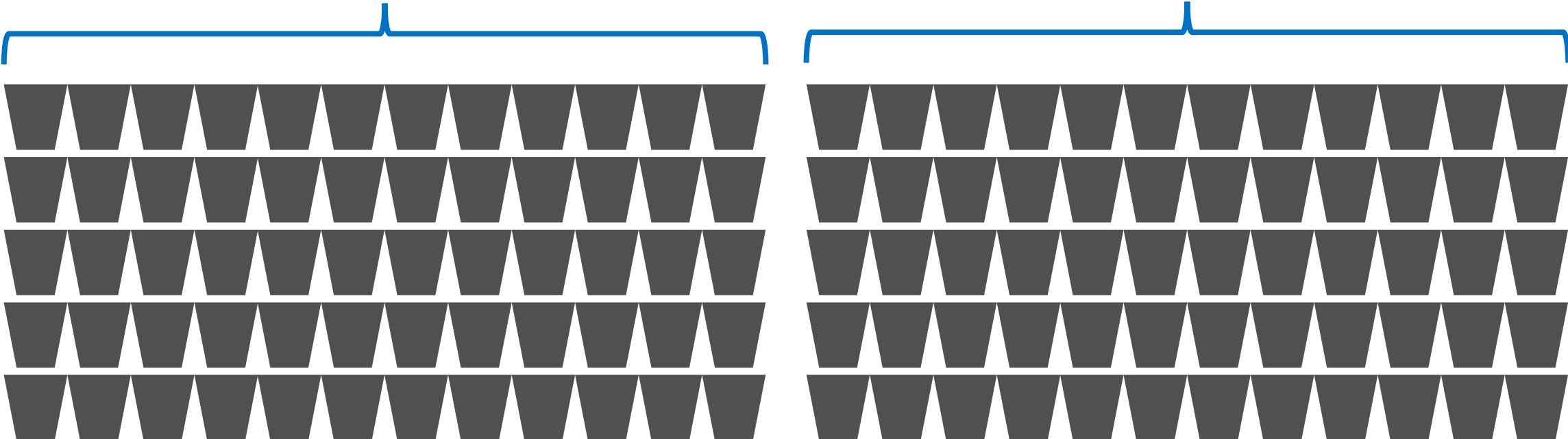
HASH (0B)



Joining HASH tables

Store_Sales HASH([ProductKey])

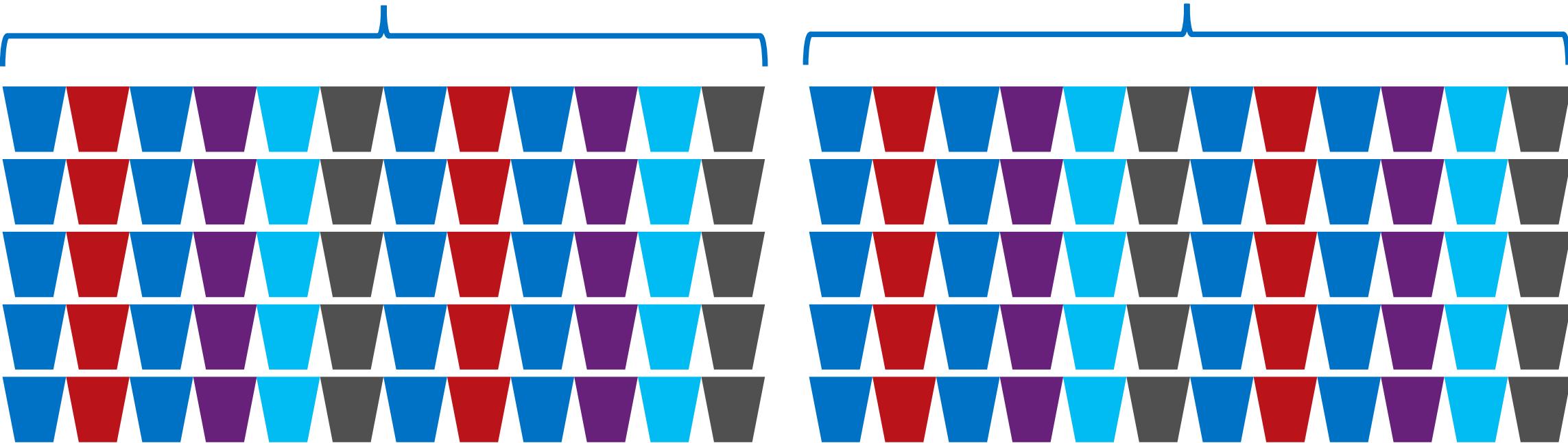
Web_Sales HASH([ProductKey])



Joining HASH tables

Store_Sales HASH([ProductKey])
[ProductKey] **INT** NULL

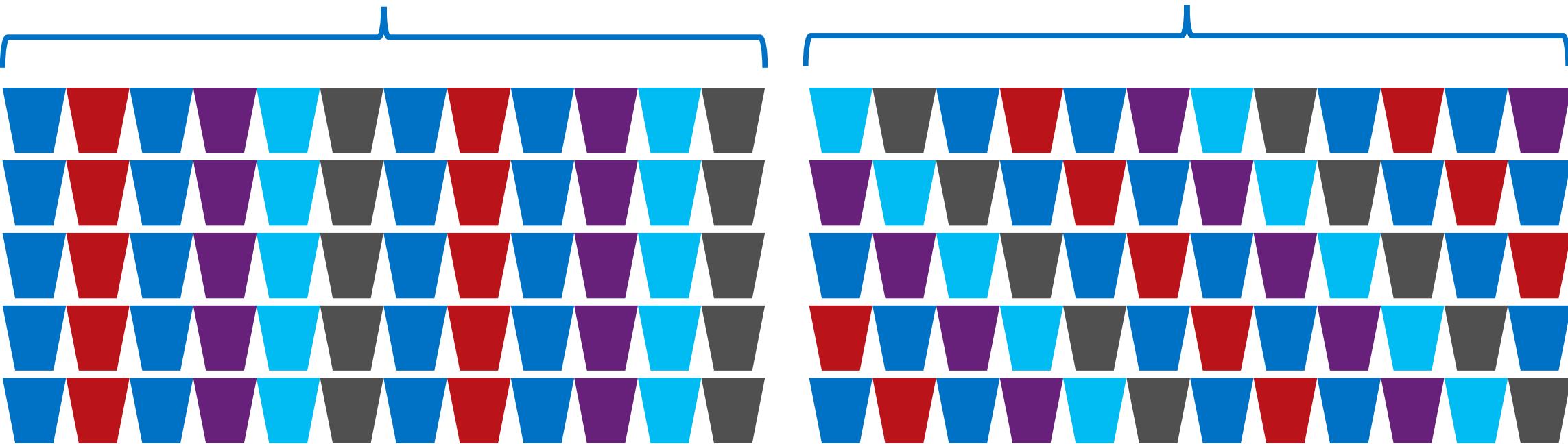
Web_Sales HASH([ProductKey])
[ProductKey] **INT** NULL



Joining HASH tables

Store_Sales HASH([ProductKey])
[ProductKey] **INT** NULL

Web_Sales HASH([ProductKey])
[ProductKey] **BIGINT** NULL



Tables – Indexes

Clustered Columnstore index (Default Primary)

Highest level of data compression

Best overall query performance

Clustered index (Primary)

Performant for looking up a single to few rows

Heap (Primary)

Faster loading and landing temporary data

Best for small lookup tables

Nonclustered indexes (Secondary)

Enable ordering of multiple columns in a table

Allows multiple nonclustered on a single table

Can be created on any of the above primary indexes

More performant lookup queries

-- Create table with index

```
CREATE TABLE orderTable
```

```
(
```

```
    OrderId INT NOT NULL,
```

```
    Date DATE NOT NULL,
```

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

```
    Country VARCHAR(2)
```

```
)
```

```
WITH
```

```
(
```

```
    CLUSTERED COLUMNSTORE INDEX |
```

```
    HEAP |
```

```
    CLUSTERED INDEX (OrderId)
```

```
);
```

-- Add non-clustered index to table

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

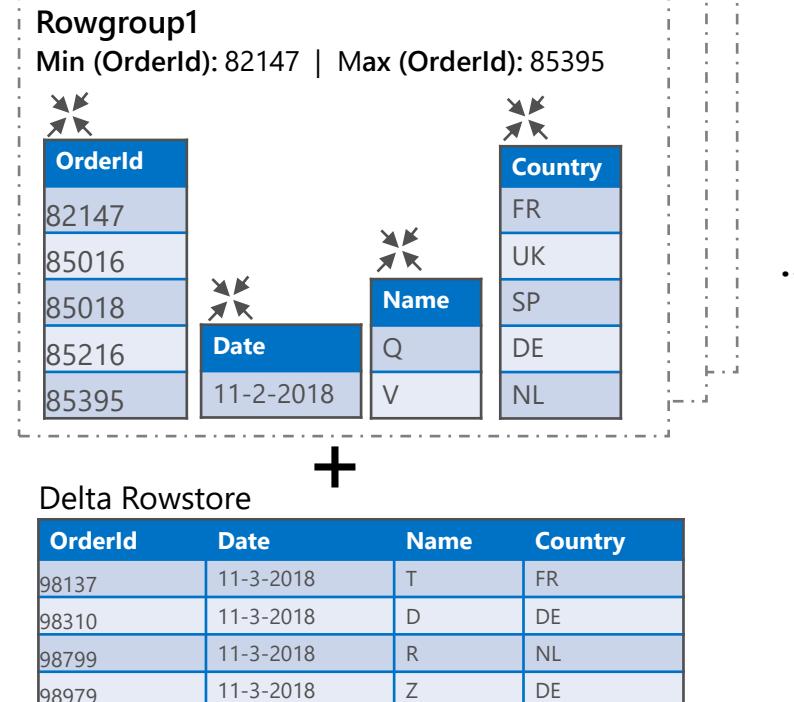
SQL Analytics Columnstore Tables

Logical table structure

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

Clustered columnstore index

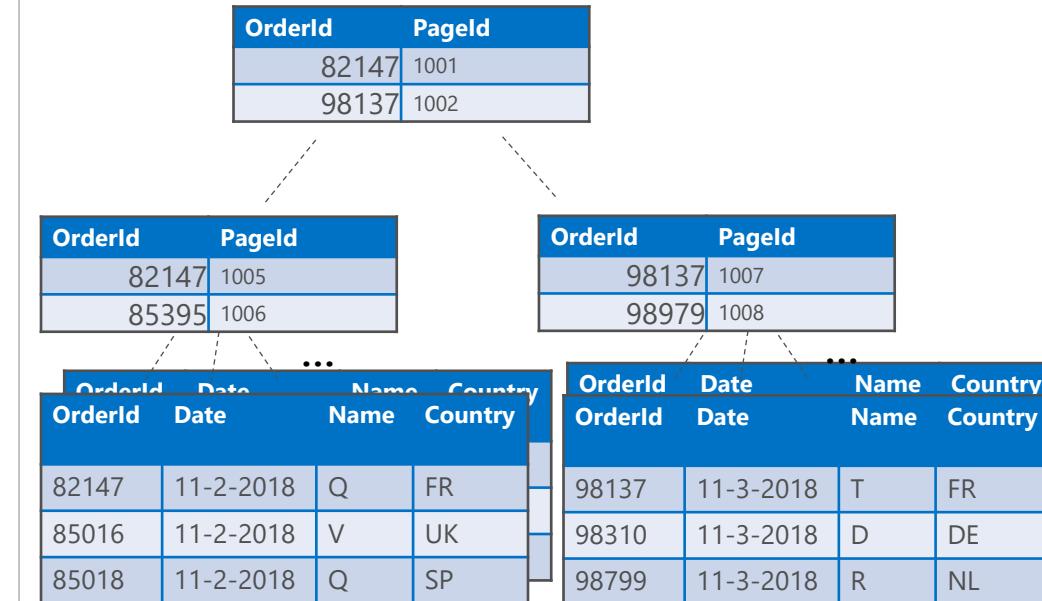
(OrderId)



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

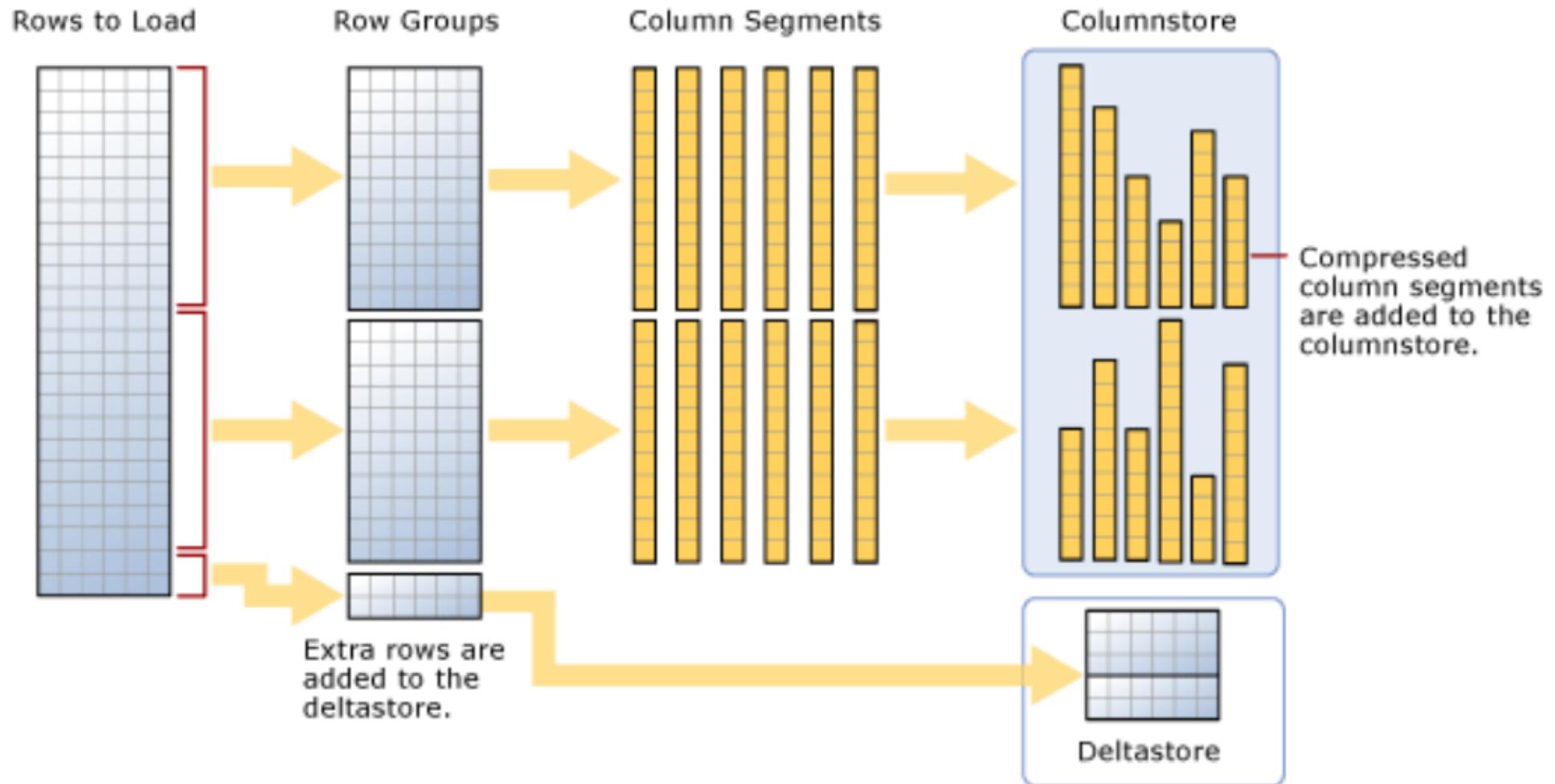
Clustered/Non-clustered rowstore index

(OrderId)



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

Columnstore Storage Model



Columnstore Storage Model

ROW STORE

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

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

(1 row affected)

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

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

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

COLUMNSTORE

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

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

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

(1 row affected)

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

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

Columnstore Storage Model

Clustered Index Scan (Clustered)

Scanning a clustered index, entirely or only a range.

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

Object

[AdventureworksDW2016CTP3].[dbo].[FactResellerSalesXL].
[PK_FactResellerSalesXL_SalesOrderNumber_SalesOrderLineNum
ber]

Output List

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

Columnstore Index Scan (Clustered)

Scan a columnstore index, entirely or only a range.

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

Object

[AdventureworksDW2016CTP3].[dbo].[FactResellerSalesXL_CCI].
[IndFactResellerSalesXL_CCI]

Output List

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

Ordered Clustered Columnstore Indexes

Overview

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

-- Create Table with Ordered Columnstore Index

```
CREATE TABLE sortedOrderTable
```

```
(
```

```
    OrderId INT NOT NULL,
```

```
    Date DATE NOT NULL,
```

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

```
    Country VARCHAR(2)
```

```
)
```

```
WITH
```

```
(
```

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

```
)
```

-- Create Clustered Columnstore Index on existing table

```
CREATE CLUSTERED COLUMNSTORE INDEX cciOrderId
```

```
ON dbo.OrderTable ORDER (OrderId)
```

-- Insert data into table with ordered columnstore index

```
INSERT INTO sortedOrderTable
```

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

Tables – Distributions

Round-robin distributed

Distributes table rows evenly across all distributions at random.

Hash distributed

Distributes table rows across the Compute nodes by using a deterministic hash function to assign each row to one distribution.

Replicated

Full copy of table accessible on each Compute node.

```
CREATE TABLE dbo.OrderTable
(
    OrderId INT NOT NULL,
    Date DATE NOT NULL,
    Name VARCHAR(2),
    Country VARCHAR(2)
)
WITH
(
    CLUSTERED COLUMNSTORE INDEX,
    DISTRIBUTION = HASH([OrderId]) |
        ROUND ROBIN |
        REPLICATED
);
```

Tables – Partitions

Overview

Table partitions divide data into smaller groups

In most cases, partitions are created on a date column

Supported on all table types

RANGE RIGHT – Used for time partitions

RANGE LEFT – Used for number partitions

Benefits

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

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

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

Tables – Partitions

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

Tables – Distributions & Partitions

Logical table structure

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

Physical data distribution

(Hash distribution (OrderId), Date partitions)

Distribution1

(OrderId 80,000 – 100,000)

11-2-2018 partition

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

11-3-2018 partition

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

...

x 60 distributions (shards)

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

Common table distribution methods

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

Data Movement

```
EXPLAIN SELECT vs_key, a.ord ,b.qty  
FROM vendor_sales a  
JOIN store_sales b ON a.vs_key = b.vid  
WHERE a.color = 'Red'
```

DMS Operation	Description
Shuffle Move	Data is Re-Distributed for Join or Aggregation
Partition Move	Data is collected from Control Node
Broadcast Move	Data is replicated to 60 distributions for Join

Statistics

Database Properties - yongsynapse

Select a page: General, Options, Permissions

Script Help

Collation: SQL_Latin1_General_CI_AS

Compatibility level: SQL Server 2016 (130)

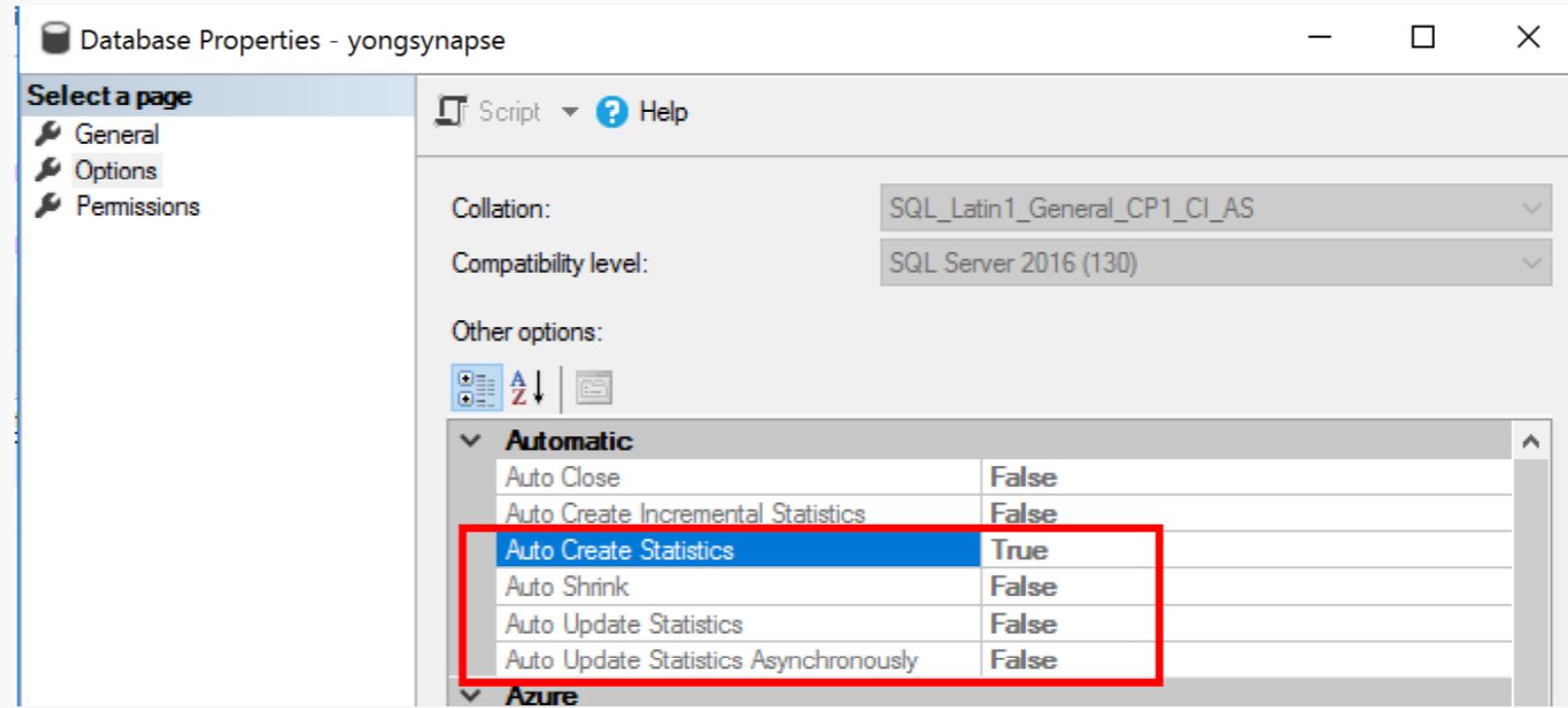
Other options:

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

CREATE STATISTICS [statistics_name] ON [schema_name].[table_name]([column_name]) WITH FULLSCAN;

CREATE STATISTICS col1_stats ON dbo.table1 (col1) WITH SAMPLE = 50 PERCENT;

UPDATE STATISTICS [dbo].[table1] ([stats_col1]);



'Optimized for Elasticity'
Sizing & Storage tiers

Sizing factors

Database capacity

Tempdb

Concurrency & Memory

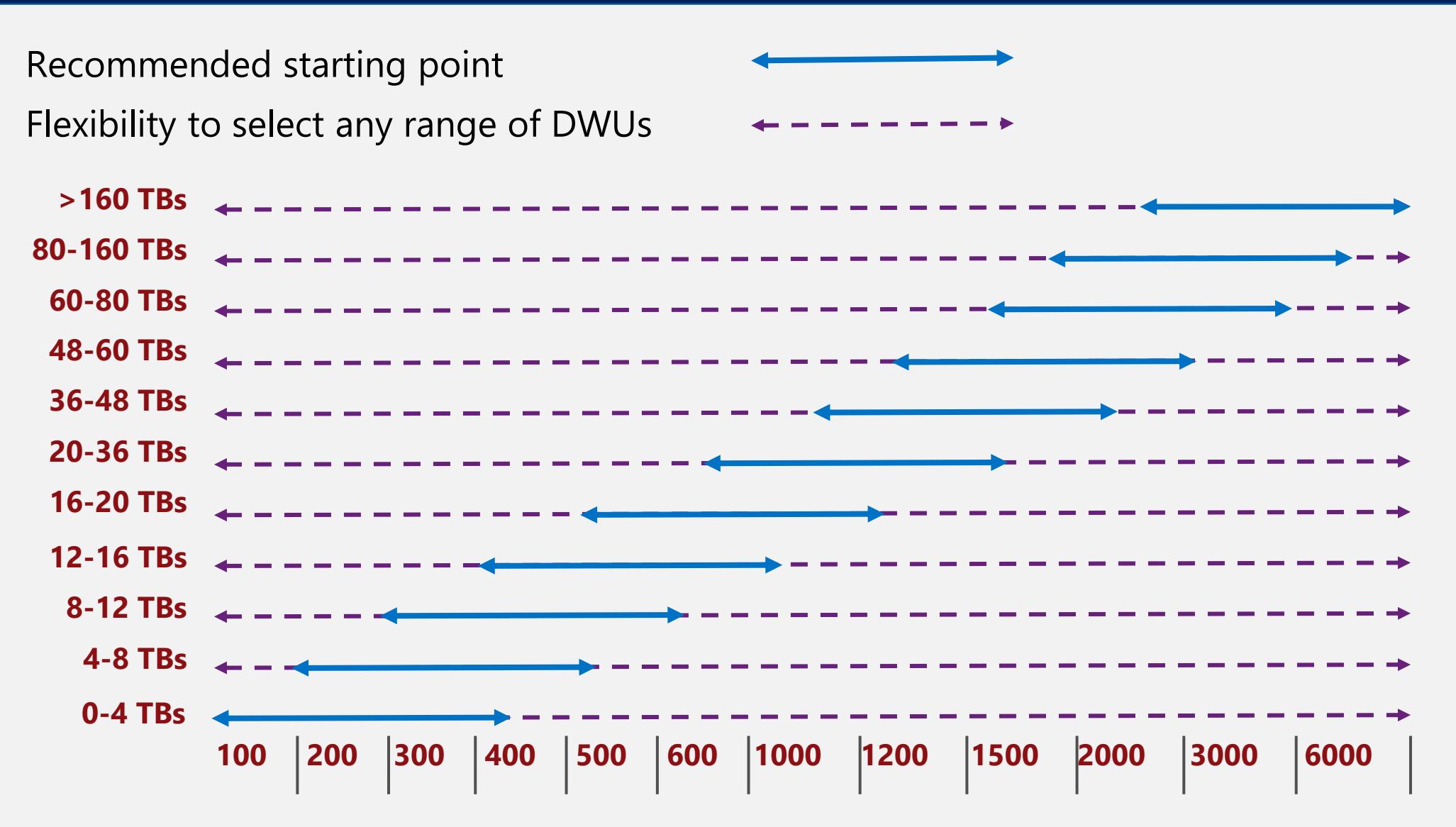
Load

Transaction size

Memory management



Starting point: Sizing by capacity



Sizing Node Count

As Is System's Node Count

What is Hadoop Worker Node Count?

What is Greenplum DB Total Node Count?

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

Sizing CPU & Memory

CPU (Hyperthreaded vCore)

Convert to DTU = DWU * 9

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

Memory

DWU * 60

(DW6000c * 60 = 3.6 Tb)

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

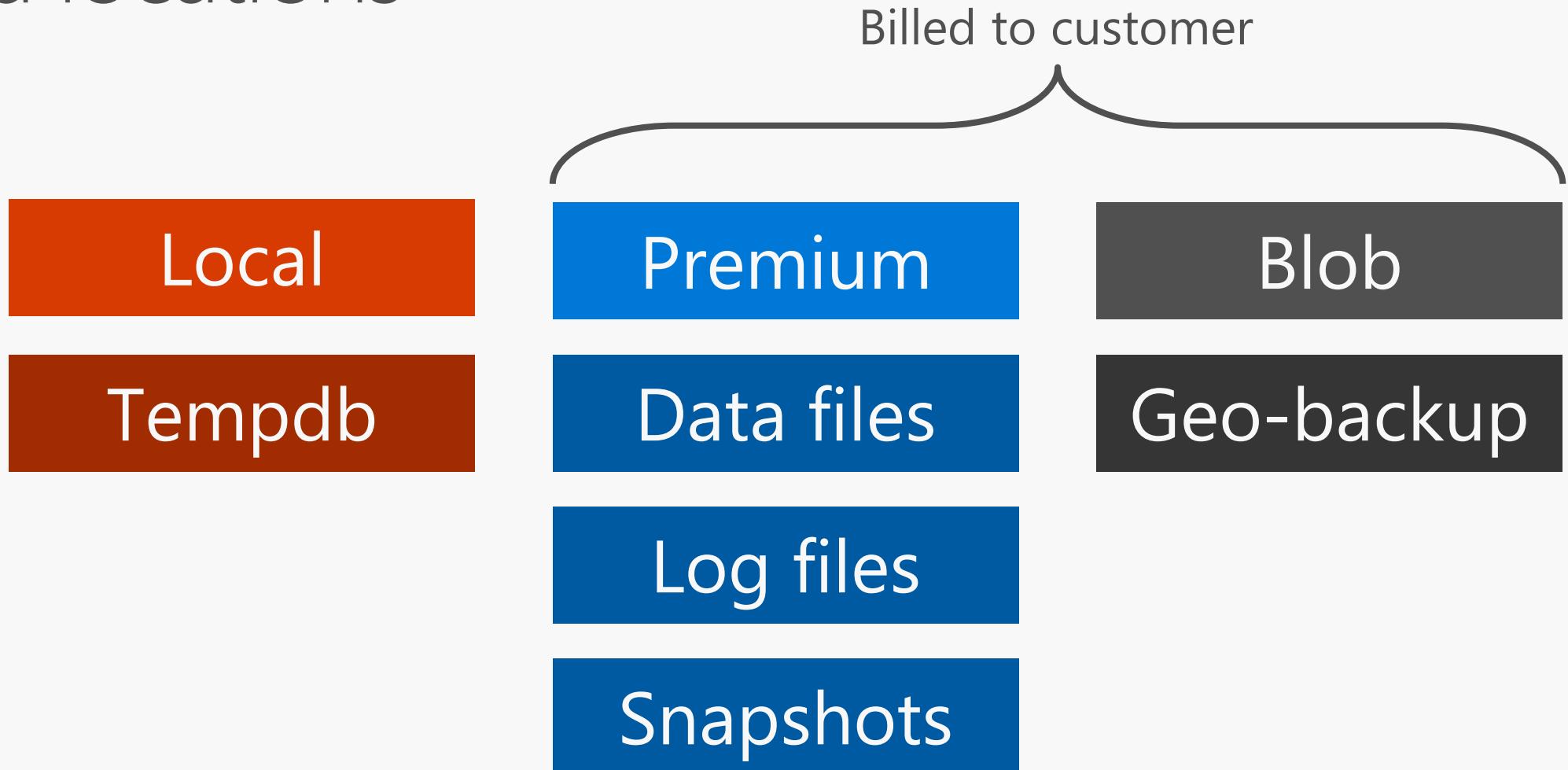
Storage tiers

Local storage

Premium storage (remote)

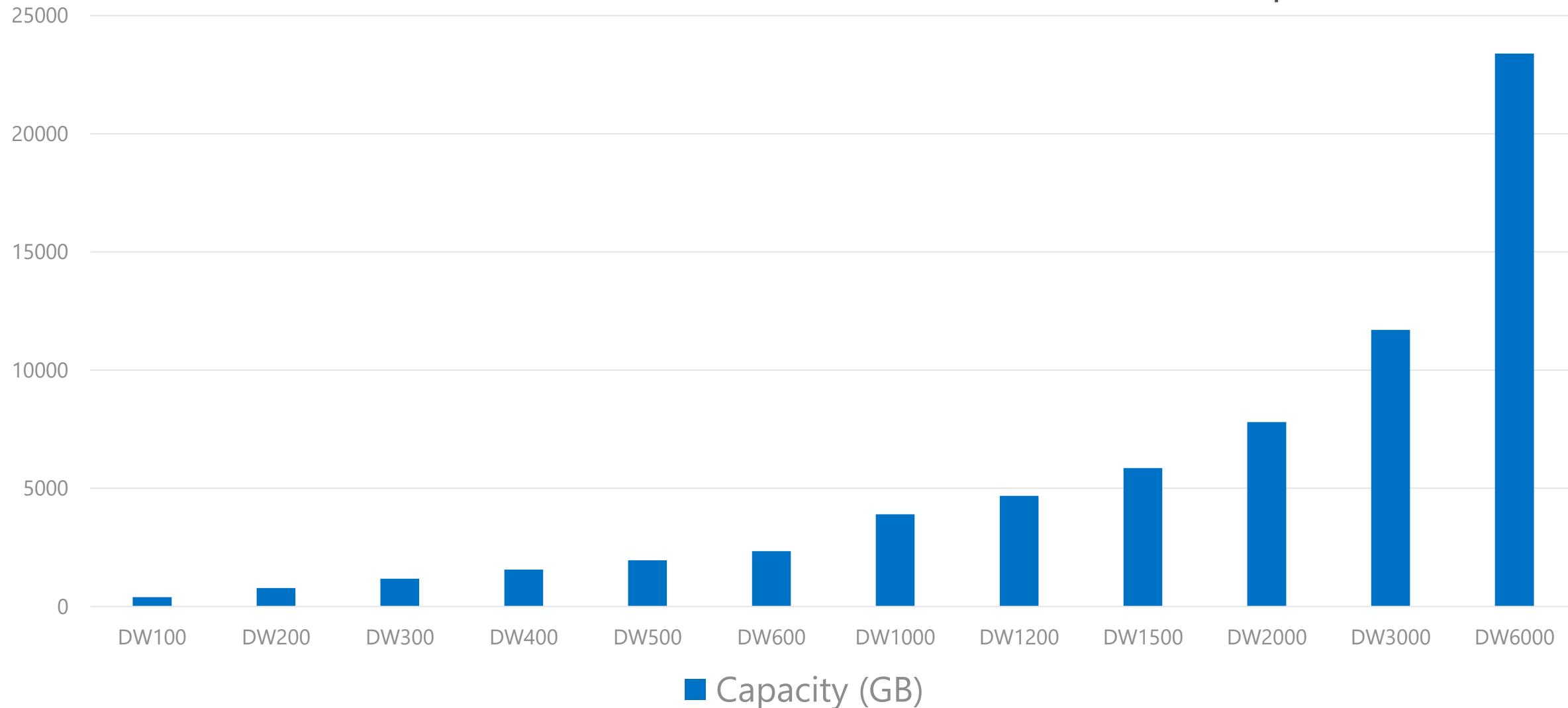
Blob storage (remote and geo redundant)

Data locations



Local Storage: Tempdb sizing

~399GB
per DW100



Premium Storage: Capacity limits

240TB

File capacity

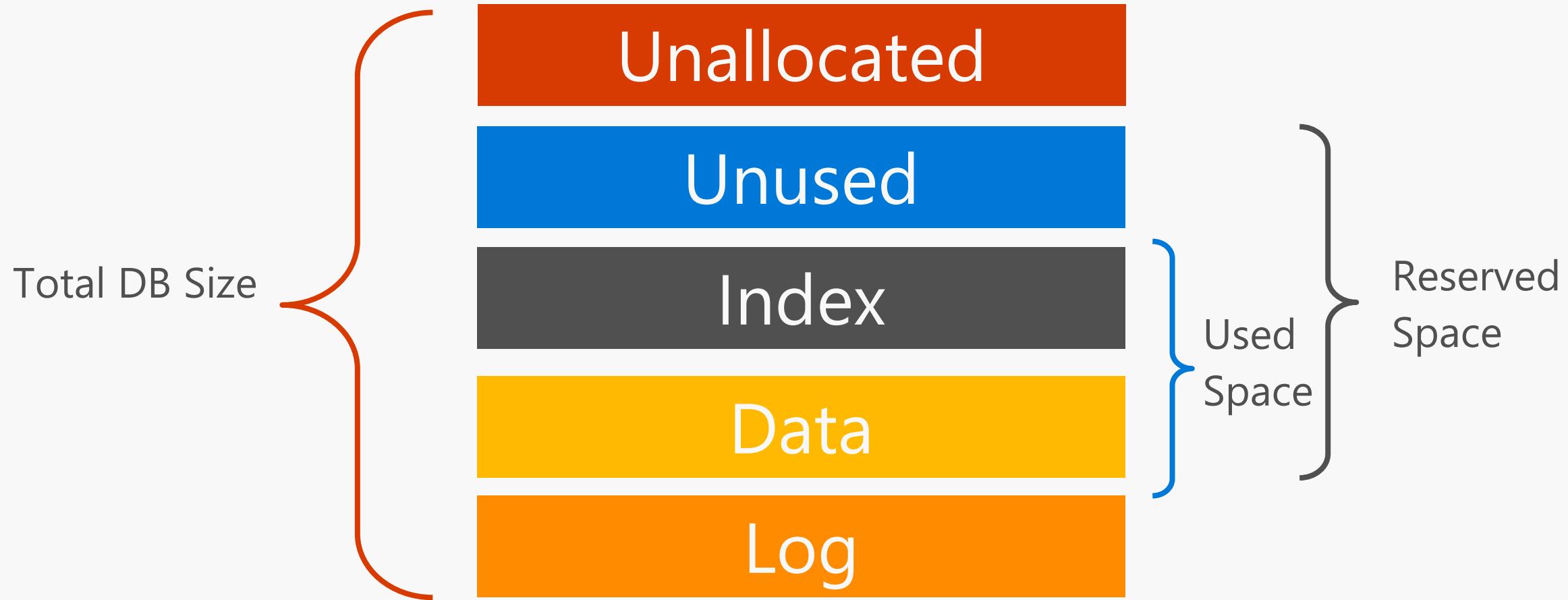
5x

CCI compression

> 1 PB

Db capacity

Premium Storage: Database Size



Premium Storage: Snapshots

Frequency

Retention

4

hours

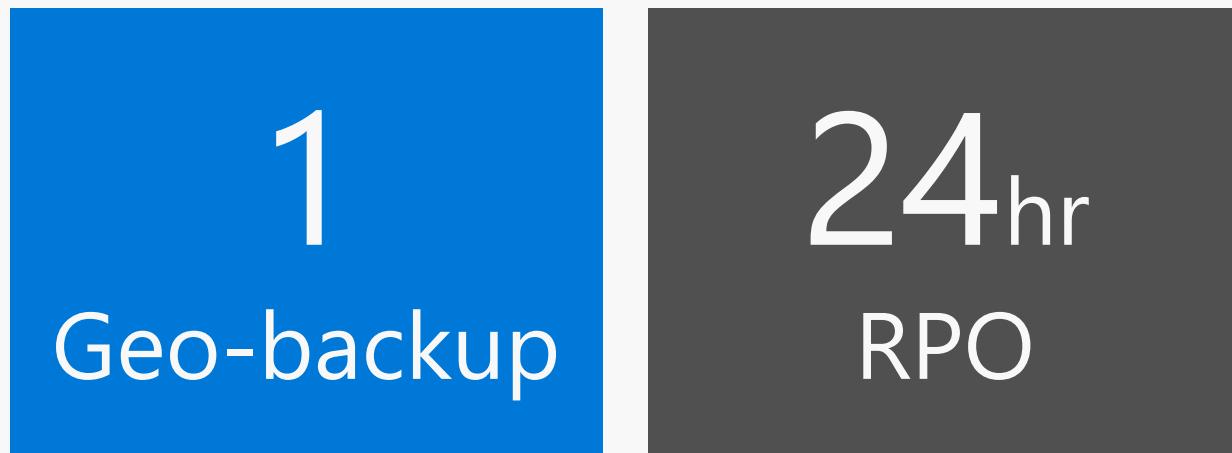
7

days

RPO : 8 Hours

Blob Storage: Geo-redundant backups

Frequency and Retention



Geo-backup policy

Save Discard Feedback



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

[Learn more](#)

Geo-backup policy

Enabled

Most recent Geo-backup time

✓ 2016-07-25T06:25:39Z

Loading

Sizing for the data load

Delimited text guidance.

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

Compressed text limits concurrent access to text files

Split data across files
OR
Use different file format

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

Loading delimited text

A compressed text file cannot be read in parallel

Splitting data across multiple files maximises load performance



Data loading

Exception

Target Table = CI or NCI

Load user is defaultrc

60 Writers

TAKE AWAY

Use mediumrc+ for
high DWU loads

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

Loading Option

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

Polybase Sample

```
CREATE MASTER KEY;

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

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

Polybase Sample

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

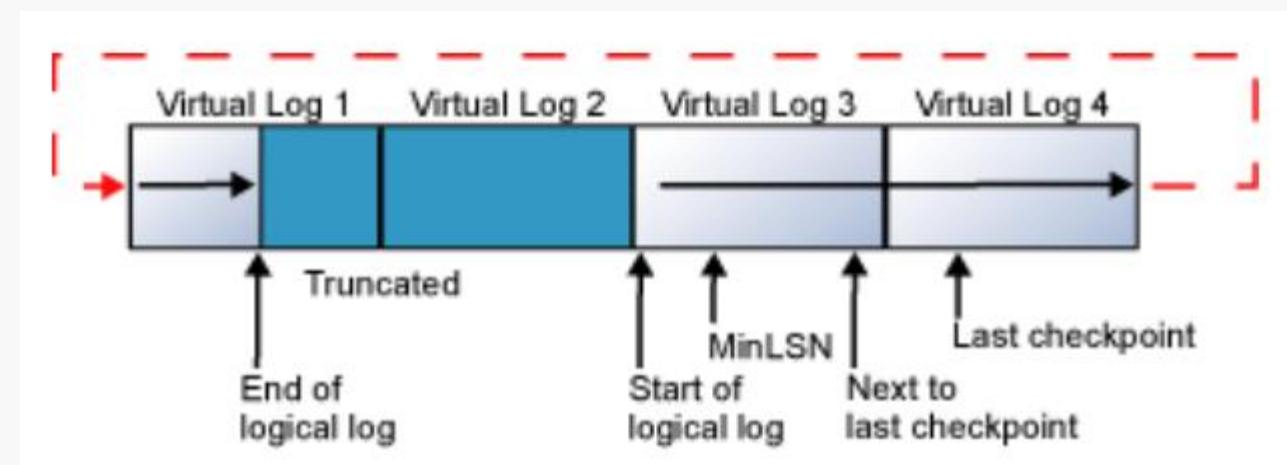
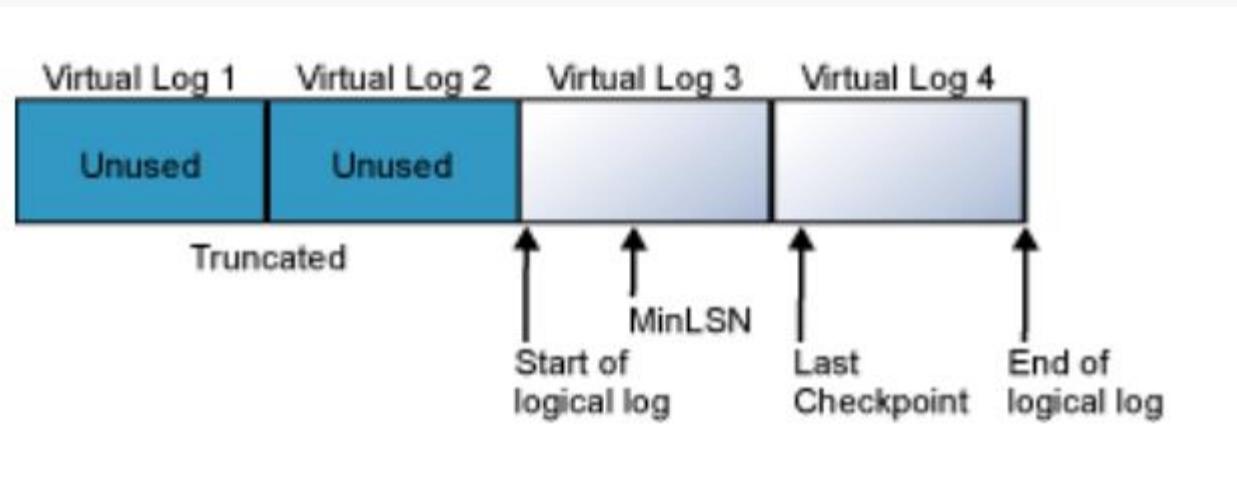
CREATE TABLE dbo.DimDate2
WITH
(
    CLUSTERED COLUMNSTORE INDEX,
    DISTRIBUTION = ROUND_ROBIN
)
AS
SELECT * FROM [dbo].[DimDate2External];
```

Polybase Sample

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

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

Transaction Logs



Transaction Logs

Transaction log truncation

Log truncation deletes inactive [virtual log files \(VLFs\)](#) from the logical transaction log of a SQL Server database, freeing space in the logical log for reuse by the Physical transaction log.

To avoid running out of space, unless log truncation is delayed for some reason, truncation occurs automatically after the following events:

- Under the simple recovery model, after a checkpoint.
- Under the full recovery model or bulk-logged recovery model, if a checkpoint has occurred since the previous backup, truncation occurs after a log backup (unless it is a copy-only log backup).

Factors that can delay log truncation

Really, Log truncation can be delayed by a variety of reasons. Learn what, if anything, is preventing your log truncation by querying the `log_reuse_wait` and `log_reuse_wait_desc` columns of the `sys.databases` catalog view.

Operations that can be minimally logged

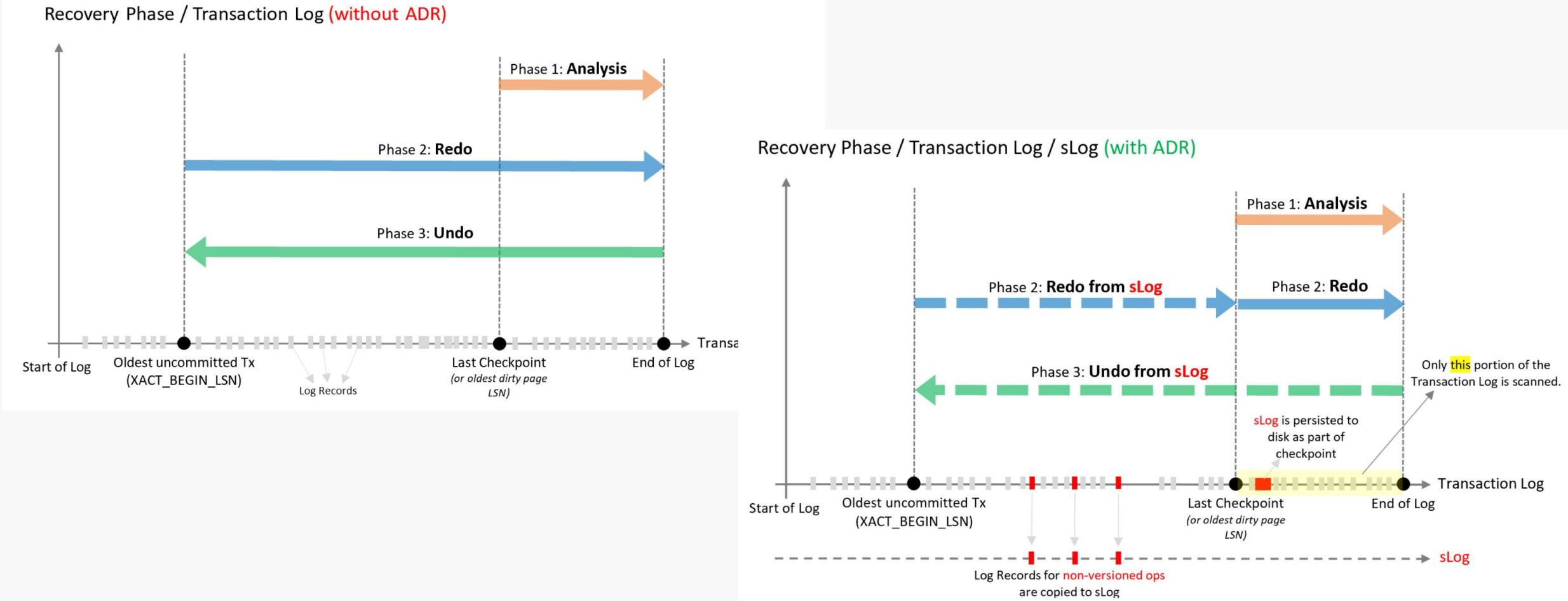
Under the full [recovery model](#), all bulk operations are fully logged. However, you can minimize logging for a set of bulk operations by switching the database to the bulk-logged recovery model temporarily for bulk operations.

Minimal logging is more efficient than full logging, and it reduces the possibility of a large-scale bulk operation filling the available transaction log space during a bulk transaction. However, if the database is damaged or lost when minimal logging is in effect, you cannot recover the database to the point of failure.

ADR (Accelerated database recovery)

Standard database recovery process

Database recovery follows the [ARIES](#) recovery model and consists of three phases, which are illustrated in the following diagram and explained in more detail following the diagram.



Monitoring

Best Practice – Query Performance

Check for SKEW (DBCC PDW_SHOWSPACEUSED)

Statistics

CTAS large return operation

De-normalize if needed

DSQL Query Plan

- Minimize data movement operations
 - Distribution & aggregation compatible
- Minimize size of data movement
 - Check for predicate pushdown. Rewrite query if needed
- Use higher **resource class** for memory intensive queries
- Load locally large external tables rather than querying directly

Dynamic Management Views (DMVs)

Overview

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

Benefits:

Simple SQL syntax

Returns result in table format

Easier to read and copy result

SQL Monitor with DMVs

Overview

Offers monitoring of

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

Count sessions by user

--count sessions by user

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

List all open sessions

-- List all open sessions

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

List all active queries

-- List all active queries

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

Query Tuning Point Explorations (Session)

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

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

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

Query Tuning Point Exploration (Long Running Query)

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

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

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

Query Tuning Point Exploration (Request Steps and SQL Request)

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

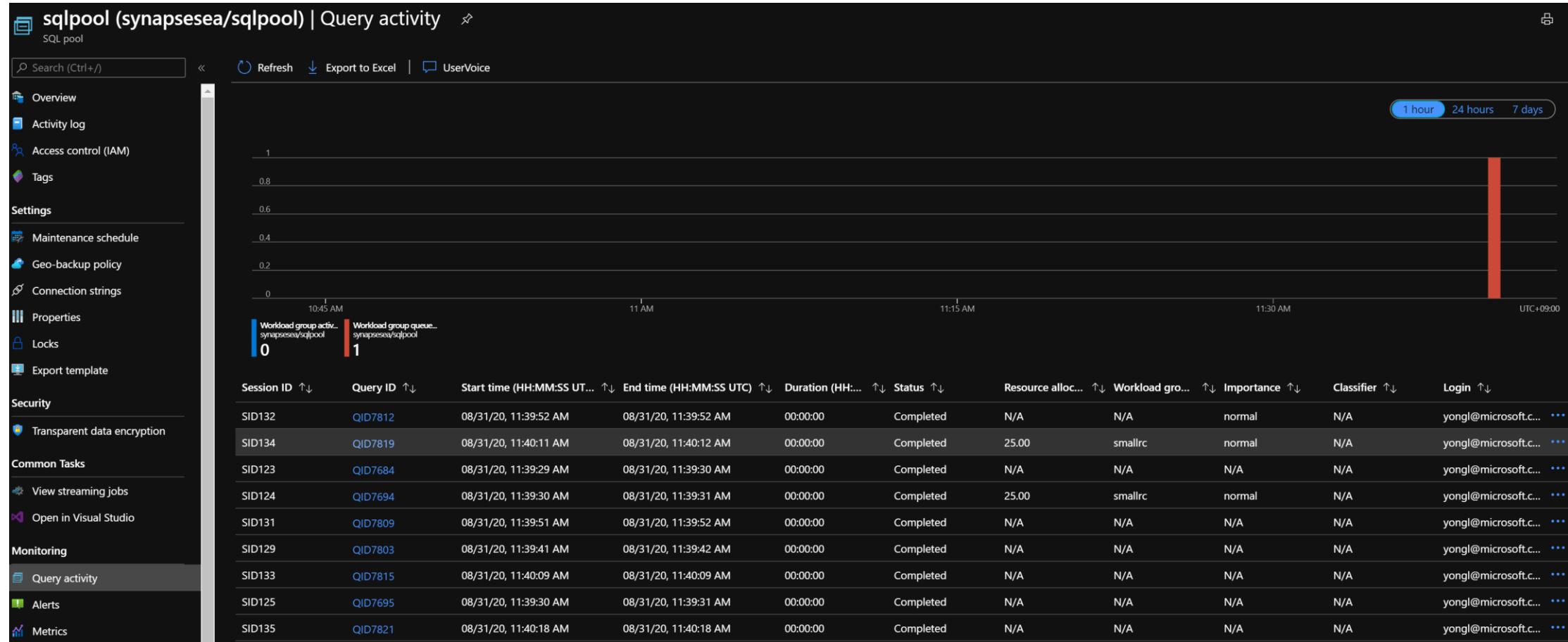
	request_id	step_index	operation_type	distribution_type	location_type	status	error_id	start_time	end_time
1	QID1592	0	ReturnOperation	AllDistributions	Compute	Complete	NULL	2017-11-03 08:17:36.883	2017-11-03 08:17:36.960

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

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

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

Query Tuning Point Explorations (Azure Portal)



Query Plan

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

SentryOne Plan Explorer

File View Tools Window Help

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

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

Command Text Results

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

Text Data

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

Text Data Plan XML Plan/Query Info

Plan Diagram

The diagram illustrates the query execution flow. It starts with a 'SELECT' node, followed by a 'Project' node. This is followed by two more 'Project' nodes, each with a 'GAgg' node. Then there is a 'Shuffle (Move)' node, which is highlighted with a red oval and shows 100.0% completion. After the shuffle, there are two more 'GAgg' nodes, each with a 'Filter' node. Finally, the process ends with a 'Get' node.

0.0 % 0.0 % 0.0 % 0.0 % 0.0 % 0.0 % 0.0 % 0.0 %
SELECT Project Project GAgg GAgg Shuffle (Move) GAgg Filter Get
17 17 17 17 17 378,199 61,423,600

Query Store Enable

Object Explorer

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

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

100 %

Messages

Commands completed successfully.

Regressed Queries..adventureWorks2012] x

TOP 25 REGRESSED QUERIES DURING THE LAST HOUR FOR DATABASE ADVENTUREWORKS2012

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

Plan Summary For Query 31

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

Plan 31 [Not Forced]

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

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

100 %

Results

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

Grafana

Grafana Support



All dashboards » Synapse



Synapse by fg



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

Last updated: a year ago



Overview

Revisions

Reviews

Comprehensive SQL functionality



Advanced storage system

- Columnstore Indexes
- Table partitions
- Distributed tables
- Isolation modes
- Materialized Views
- Nonclustered Indexes
- Result-set caching

T-SQL Querying

- Windowing aggregates
- Approximate execution (Hyperloglog)
- JSON data support

Complete SQL object model

- Tables
- Views
- Stored procedures
- Functions

Windowing functions

OVER clause

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

Computes a value for each row in the window

Aggregate functions

COUNT, MAX, AVG, SUM, APPROX_COUNT_DISTINCT, MIN, STDEV, STDEVP, STRING_AGG, VAR, VARP, GROUPING, GROUPING_ID, COUNT_BIG, CHECKSUM_AGG

Ranking functions

RANK, NTILE, DENSE_RANK, ROW_NUMBER

Analytical functions

LAG, LEAD, FIRST_VALUE, LAST_VALUE, CUME_DIST, PERCENTILE_CONT, PERCENTILE_DISC, PERCENT_RANK

ROWS | RANGE

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

`SELECT`

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

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

Windowing Functions (continued)

Analytical functions

LAG, LEAD, FIRST_VALUE, LAST_VALUE, CUME_DIST, PERCENTILE_CONT, PERCENTILE_DISC, PERCENT_RANK

-- PERCENTILE_CONT, PERCENTILE_DISC

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

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

--LAG Function

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

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

Windowing Functions (continued)

ROWS | RANGE

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

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



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

Approximate execution

HyperLogLog accuracy

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

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

APPROX_COUNT_DISTINCT

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

Use Case: Approximating web usage trend behavior

-- Syntax

```
APPROX_COUNT_DISTINCT( expression )
```

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

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

Approximate execution

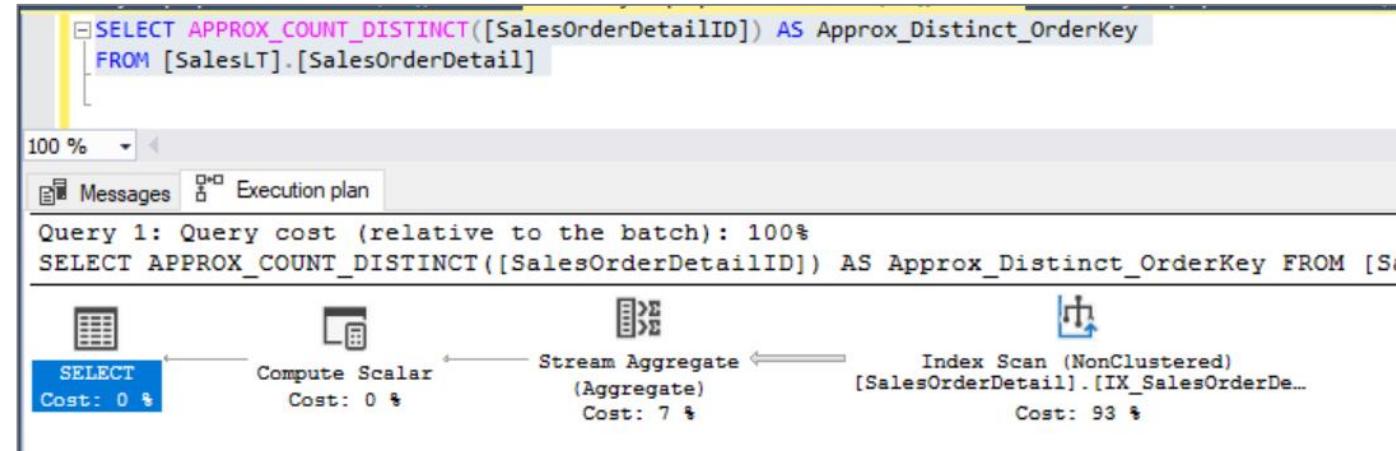
APPROX_COUNT_DISTINCT

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

100 %

Results Messages

Approx_Distinct_OrderKey
540



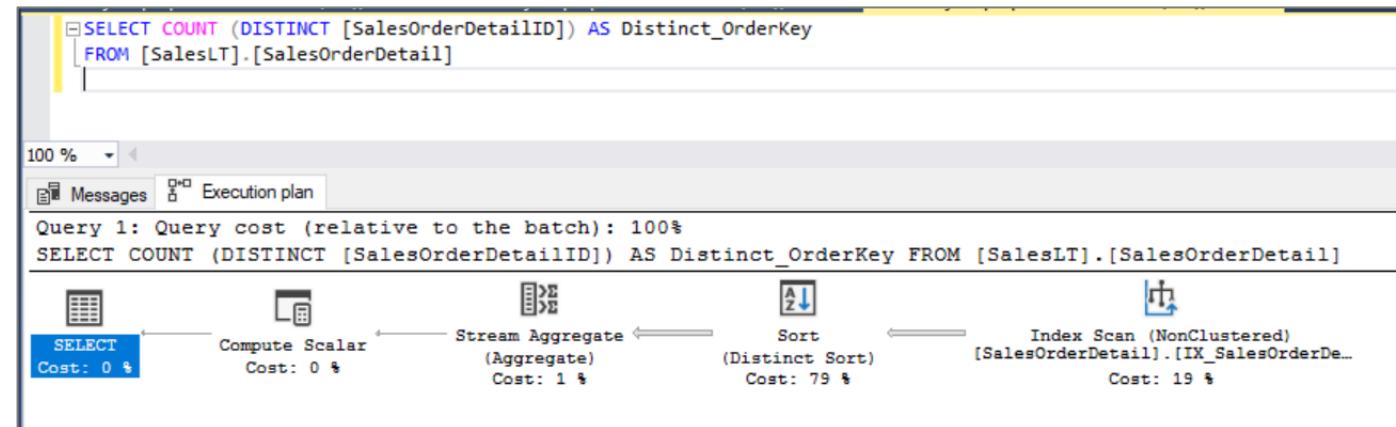
COUNT DISTINCT

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

100 %

Results Messages

Distinct_OrderKey
542



Group by options

Group by with rollup

Creates a group for each combination of column expressions.

Rolls up the results into subtotals and grand totals

Calculate the aggregates of hierarchical data

-- GROUP BY ROLLUP Example --

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

-- Results --

Grouping sets

Combine multiple GROUP BY clauses into one GROUP BY CLAUSE.

Equivalent of UNION ALL of specified groups.

-- GROUP BY SETS Example --

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



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

Snapshot isolation

Overview

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

This prevents dirty reads.

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

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

Isolation level

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

READ_COMMITTED_SNAPSHOT

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

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

JSON data support – insert JSON data

Overview

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

JSON data is stored using standard NVARCHAR table columns.

Benefits

Transform arrays of JSON objects into table format

Performance optimization using clustered columnstore indexes and memory optimized tables

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

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

JSON data support – read JSON data

Overview

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

- **ISJSON** – verify if text is valid JSON
- **JSON_VALUE** – extract a scalar value from a JSON string
- **JSON_QUERY** – extract a JSON object or array from a JSON string

Benefits

Ability to get standard columns as well as JSON column

Perform aggregation and filter on JSON values

-- Return all rows with valid JSON data

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

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

-- Extract values from JSON string

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

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

JSON data support – modify and operate on JSON data

Overview

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

Modify JSON data with the following:

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

Benefits

Flexibility to update JSON string using T-SQL

Convert hierarchical data into flat tabular structure

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

OrderDetails

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

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

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

Stored Procedures

Overview

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

Promotes flexibility and modularity.

Supports parameters and nesting.

Benefits

Reduced server/client network traffic, improved performance

Stronger security

Easy maintenance

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

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

Materialized views

Overview

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

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

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

Supported aggregations: MAX, MIN, AVG, COUNT, COUNT_BIG, SUM, VAR, STDEV

Benefits

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

High availability and resiliency as regular tables

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

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

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

Materialized views - example

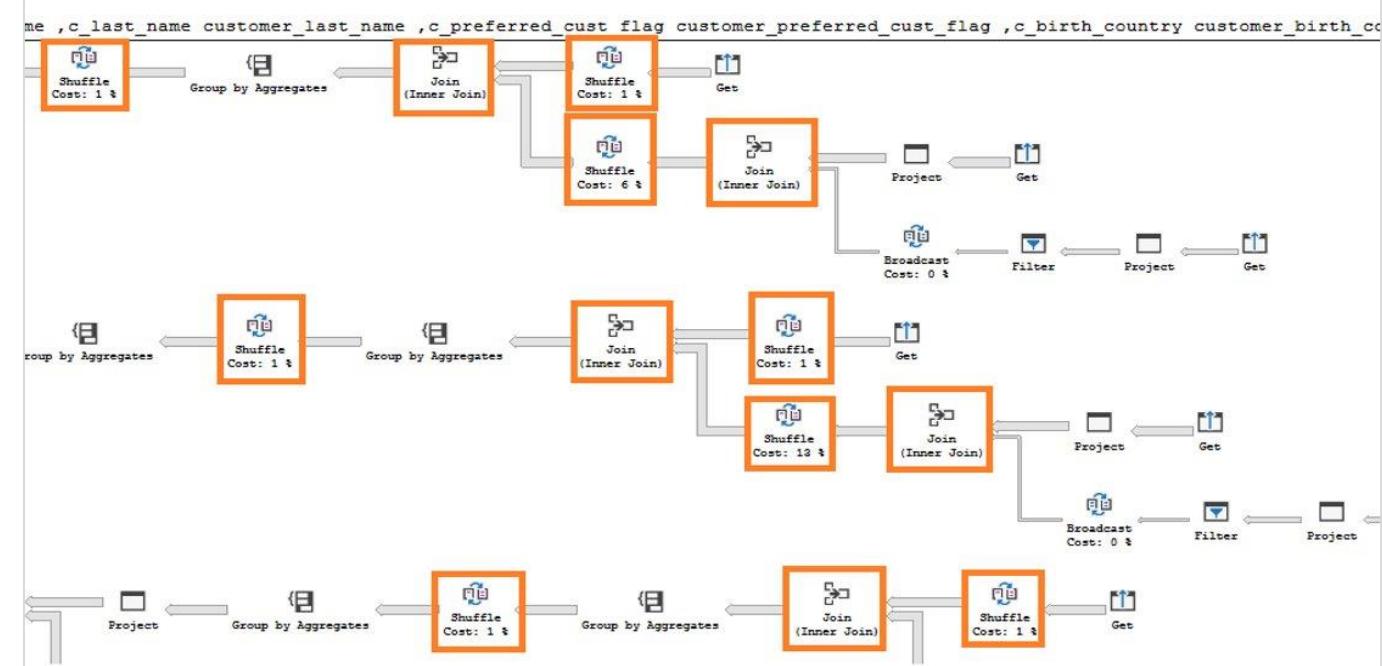
In this example, a query to get the year total sales per customer is shown to have a lot of data shuffles and joins that contribute to slow performance:

No relevant indexed views created on the data warehouse

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

Execution time: 103 seconds

Lots of data shuffles and joins needed to complete query



Materialized views - example

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

Original query – get year total sales per customer

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

Create indexed view with hash distribution on customer_id column

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

Indexed (materialized) views - example

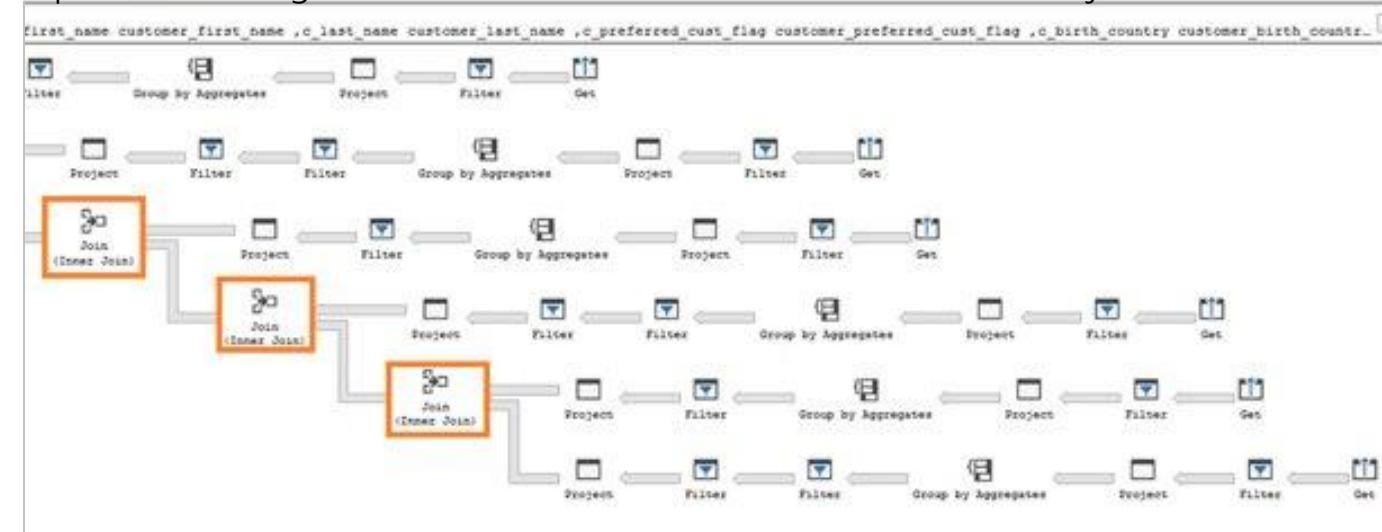
SQL pool query optimizer automatically leverages the indexed view to speed up the same query. Notice that the query does not need to reference the view directly

Original query – no changes have been made to query

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

Execution time: 6 seconds

Optimizer leverages materialized view to reduce data shuffles and joins needed



Materialized views- Recommendations

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

EXPLAIN WITH_RECOMMENDATIONS - provides query plan with recommendations to optimize the SQL statement performance.

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

COPY command

Overview

Copies data from source to destination

Benefits

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

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

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

Supports CSV, PARQUET, ORC file formats

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

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

Result-set caching

Overview

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

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

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

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

Benefits

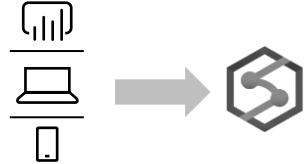
Enhances performance when same result is requested repetitively

Reduced load on server for repeated queries

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

```
-- Turn on/off result-set caching for a database  
-- Must be run on the MASTER database  
ALTER DATABASE {database_name}  
SET RESULT_SET_CACHING { ON | OFF }  
  
-- Turn on/off result-set caching for a client session  
-- Run on target Azure Synapse Analytics  
SET RESULT_SET_CACHING {ON | OFF}  
  
-- Check result-set caching setting for a database  
-- Run on target Azure Synapse Analytics  
SELECT is_result_set_caching_on  
FROM sys.databases  
WHERE name = {database_name}  
  
-- Return all query requests with cache hits  
-- Run on target data warehouse  
SELECT *  
FROM sys.dm_pdw_request_steps  
WHERE command like '%DWResultCacheDb%'  
    AND step_index = 0
```

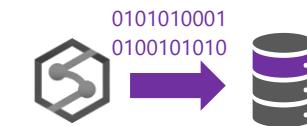
Result-set caching flow



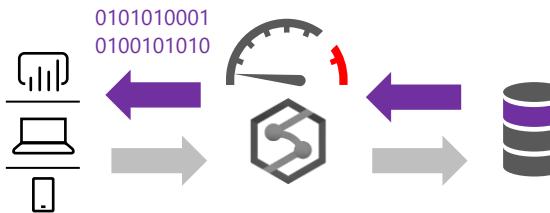
- 1 Client sends query to SQL pool



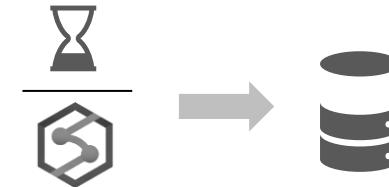
- 2 Query is processed using compute nodes which pull data from remote storage, process query and output back to client app



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



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



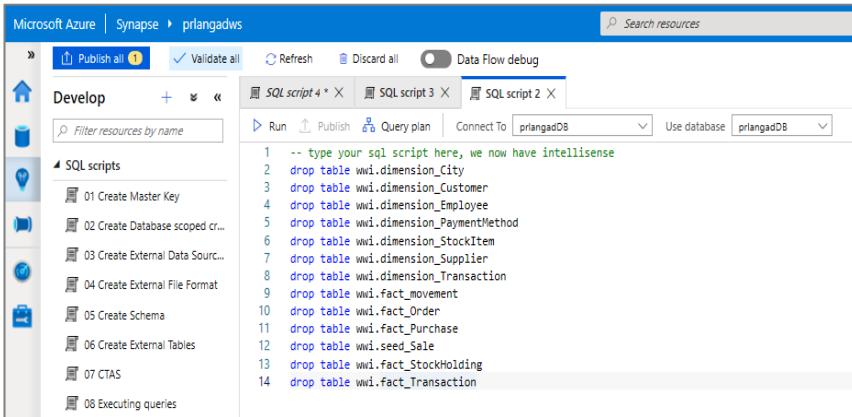
- 4 Remote storage cache is evicted regularly based on time, cache usage, and any modifications to underlying table data.



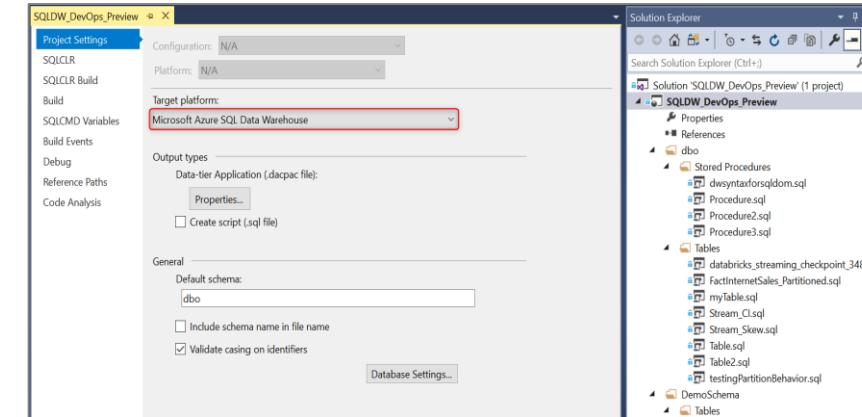
- 5 Cache will need to be regenerated if query results have been evicted from cache

Developer Tools

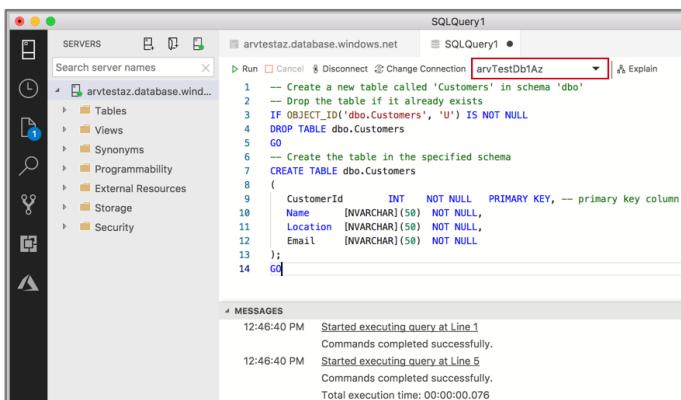
Azure Synapse Analytics



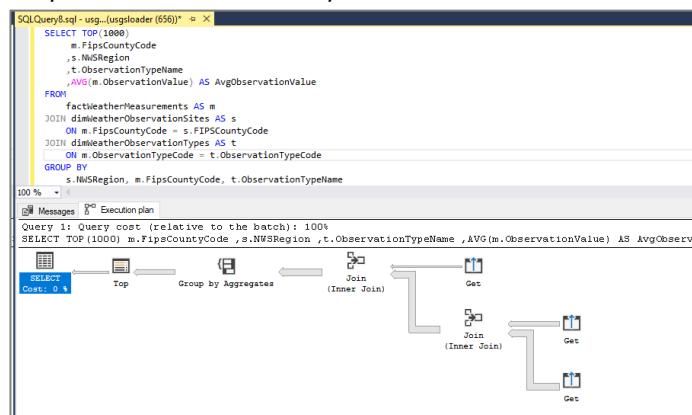
Visual Studio - SSDT database projects



Azure Data Studio (queries, extensions etc.)



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



Visual Studio Code

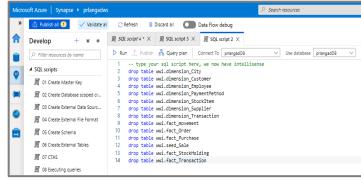
The screenshot shows the Visual Studio Code interface with the following details:

- Explorer Tab:** Shows 'OPEN EDITORS' with two items: 'Welcome' and 'Untitled-1'. It also indicates 'NO FOLDER OPENED'.
- Welcome Tab:** Displays a sample SQL script for inserting rows into the 'Employees' table and querying all employee information.
- Untitled-1 Tab:** Displays a partial SQL query starting with 'SELECT'.

```
39 -- Insert rows into table 'Employees'
40
41 INSERT INTO Employees
42     ([EmployeeId], [Name], [Location])
43 VALUES
44     ('1', 'N'Jared', 'N'Australia'),
45     ('2', 'N'Nikita', 'N'India'),
46     ('3', 'N'Tom', 'N'Germany'),
47     ('4', 'N'Jake', 'N'United States')
48 GO
49
50 -- Query all employee information
51
52 SELECT *
53 FROM dbo.dbo.Employees
54 GO
55
56 table msgsqlSampleDB.dbo.Employees AS e
57
58     Employees
59         EmployeeId
60             #ENCRYPTBYASYMMKEY
61             #ENCRYPTBYCERT
62             #ENCRYPTBYKEY
63             #ENCRYPTBYPASSPHRASE
64             #EMonth
65             #ErrorLine
66             #ErrorMessage
67             #ErrorNumber
68             #ErrorProcedure
```

Developer Tools

Azure Synapse Analytics

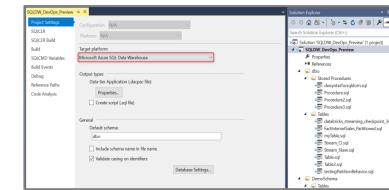


Azure Cloud Service

Offers end-to-end lifecycle for analytics

Connects to multiple services

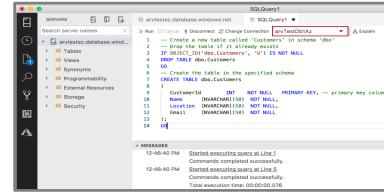
Visual Studio - SSDT database projects



Runs on Windows

Create, maintain database code, compile, code refactoring

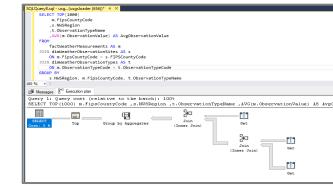
Azure Data Studio



Runs on Windows, Linux, macOS

Light weight editor, (queries and extensions)

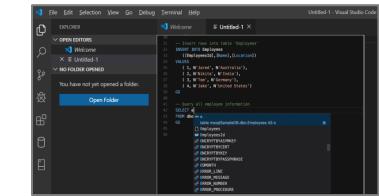
SQL Server Management Studio



Runs on Windows

Offers GUI support to query, design and manage

Visual Studio Code



Runs on Windows, Linux, macOS

Offers development experience with light-weight code editor

Continuous integration and delivery (CI/CD)

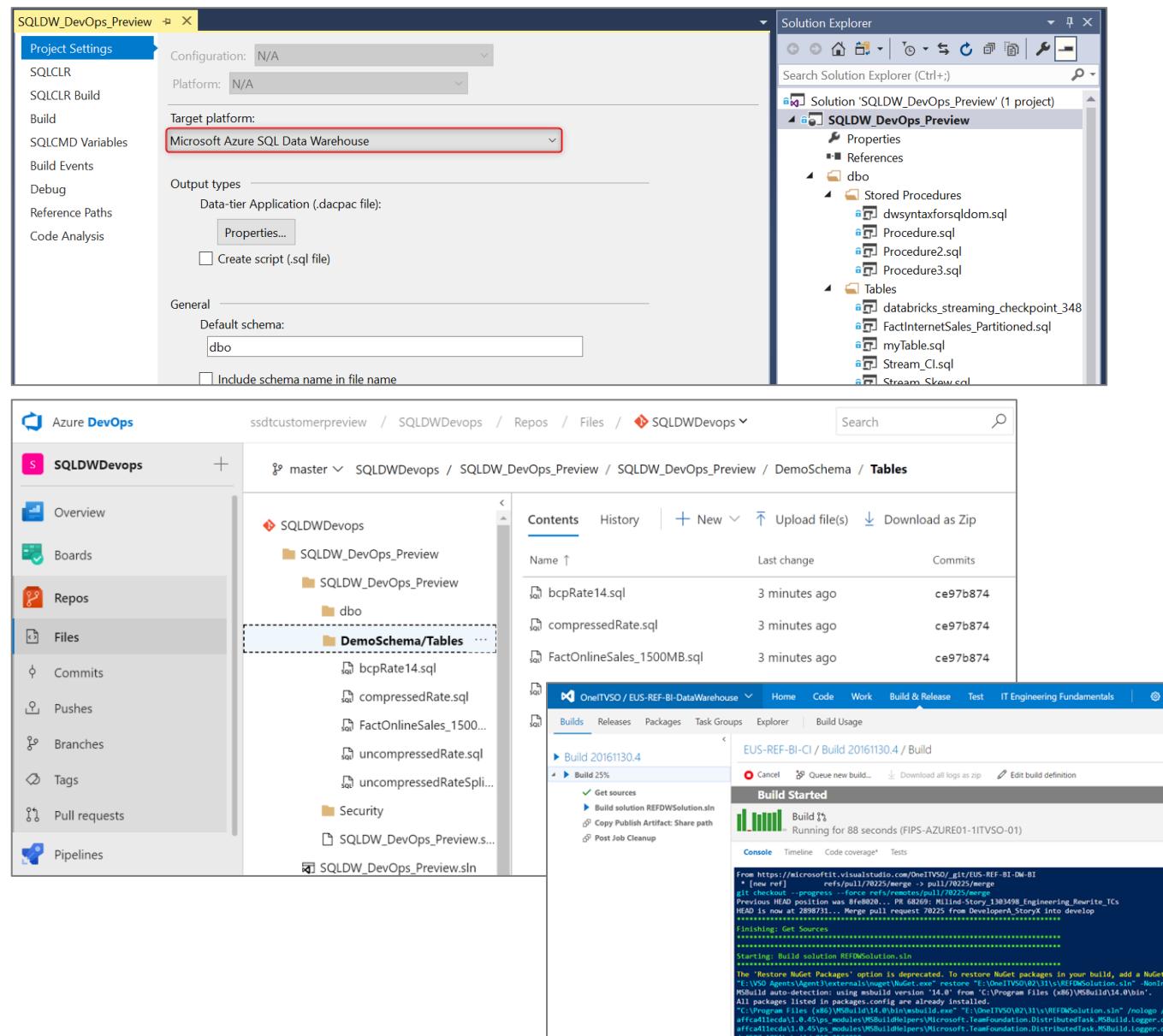
Overview

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

Benefits

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

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



Azure Advisor recommendations

Suboptimal Table Distribution

Reduce data movement by replicating tables

Data Skew

Choose new hash-distribution key

Slowest distribution limits performance

Cache Misses

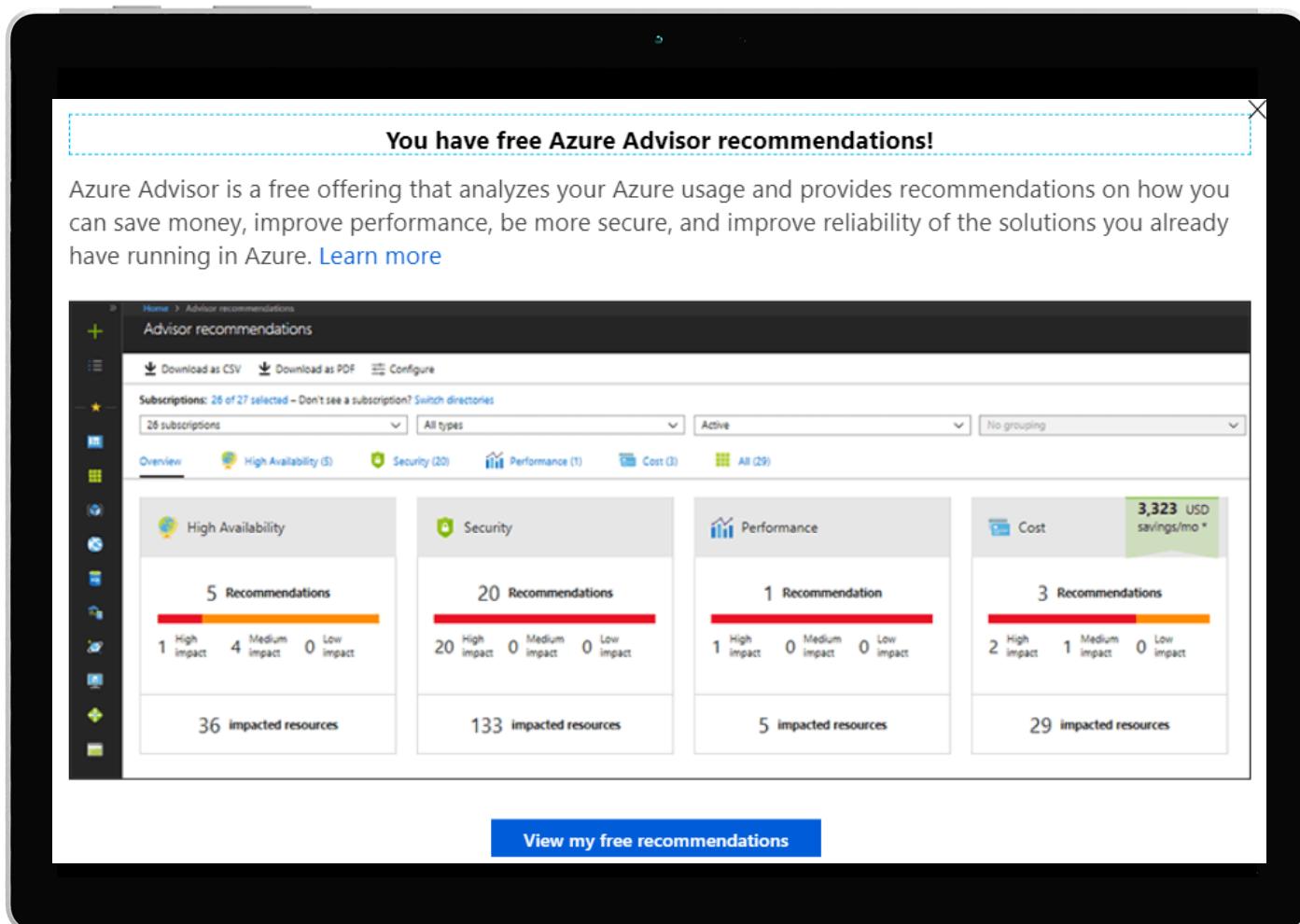
Provision additional capacity

Tempdb Contention

Scale or update user resource class

Suboptimal Plan Selection

Create or update table statistics



Maintenance windows

Overview

Choose a time window for your upgrades.

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

Windows can be from 3 to 8 hours.

24-hour advance notification for maintenance events.

Benefits

Ensure upgrades happen on your schedule.

Predictable planning for long-running jobs.

Stay informed of start and end of maintenance.

Maintenance Schedule (preview)

Maintenance on your data warehouse could occur once a week within one of two maintenance windows. Choose the primary and secondary windows that best suit your operational needs. If you would like to use the maintenance windows already defined, no action is required.

Maintenance will not take place outside these windows unless we notify you in advance.

Choose primary window

Saturday - Sunday Tuesday - Thursday

Primary maintenance window	Secondary maintenance window
Day i Saturday	Day i Tuesday
Start time i 03:00 UTC	Start time i 13:00 UTC
Time window i 8 hours	Time window i 8 hours

Schedule summary

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

Automatic statistics management

Overview

Statistics are automatically created and maintained for 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
```

SQL On-Demand
SQL On Demand

SQL On-Demand

Overview

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

Benefits

Serverless

No infrastructure

Pay only for query execution

No ETL

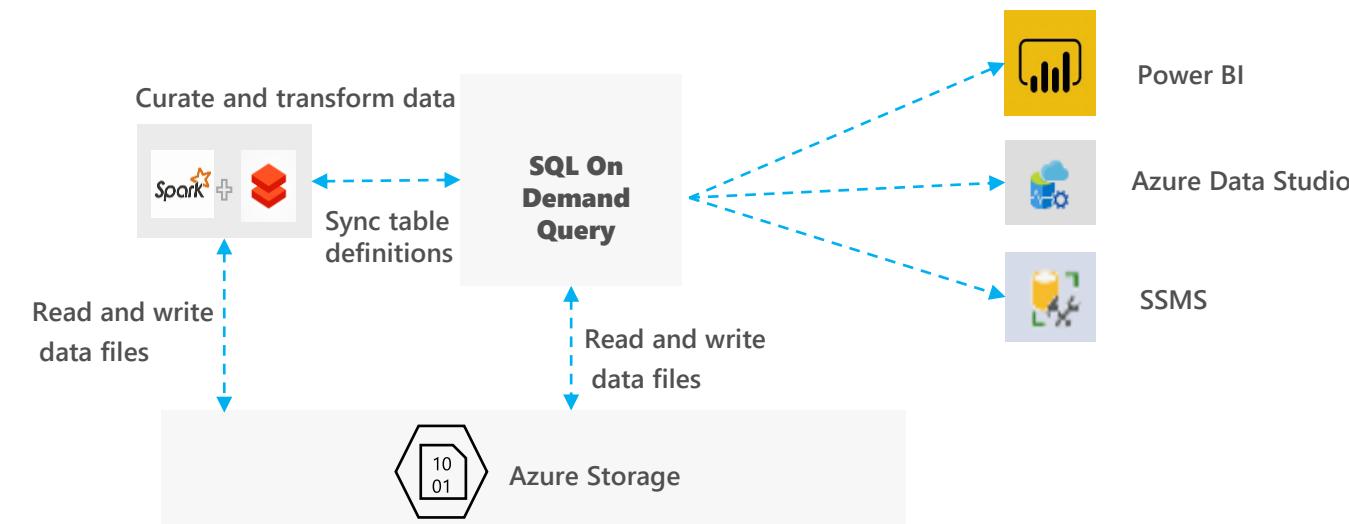
Offers security

Data integration with Databricks, HDInsight

T-SQL syntax to query data

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

Support for BI ecosystem



SQL On Demand – Querying on storage

Microsoft Azure | Synapse Analytics > prlangadws2

Search resources

Publish all 2 Validate all Refresh Discard all

Data HolidayDataPi... Load Data to S... Pipeline 1 Data flow 1 SQL script 1 Copy Open Da... nytlc

Upload Download New Folder Select All Rename Manage Access Properties Delete Refresh

Storage accounts prlangaddemosa (Primary)

- filesystem
- holidaydatacontainer
- isdweatherdatacontainer
- nytlc**
- prlangaddemosa
- tmpcontainer
- wwimporters

New SQL script and open in new tab

part-00133-120938564719836543-aea5b543-5e83-4a7d-8d31-69f72c50b05d-15253-1.c000.snappy.parquet

New SQL script New notebook Copy ABFS path Manage Access... Rename... Download Delete Properties...

LAST MODIFIED 10/25/2019, 2:20:23 PM

Microsoft Azure | Synapse Analytics > prlangadws2

Search resources

Publish all 3 Validate all Refresh Discard all

Data HolidayDataPi... Load Data to S... Pipeline 1 Data flow 1 SQL script 1 Copy Open Da... nytlc SQL script 2

Run Publish Query plan Connect to SQL Analytics on-demand Use database master

```

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

```

Results Messages

View Table Chart Export results

VENDORID	TPEPICKUPDATETIME	TPEPDROPOFFDATETIME	PASSENGERCOUNT	TRIPDISTANCE	PULOCATIONID	DOLOCATIONID	STARTLON	STARTLAT	ENDLON	ENDLAT
2	2015-02-28T23:5...	2015-03-01T00:0...	6	1.63	NULL	NULL	-74.000846862793	40.7306938171387	-73.	
1	2015-03-28T19:2...	2015-03-28T19:2...	1	2.2	NULL	NULL	-73.977653503418	40.7631607055664	-73.	
2	2015-02-28T23:5...	2015-03-01T00:1...	5	3.23	NULL	NULL	-73.96012878417...	40.7621574401855	-73.	
1	2015-03-28T19:2...	2015-03-28T19:3...	1	2.1	NULL	NULL	-73.98143005371...	40.7815055847168	-74.	
2	2015-02-28T23:5...	2015-03-01T00:1...	1	3.52	NULL	NULL	-73.98373413085...	40.7497062683105	-74.	
?	2015-02-28T00:0...	2015-02-28T00:0...	5	...	NULL	NULL	-73.9814077707	40.7497062683105	-74.	

00:01:00 Query executed successfully.

SQL On Demand – Querying CSV File

Overview

Uses OPENROWSET function to access data

Benefits

Ability to read CSV File with

- no header row, Windows style new line
- no header row, Unix-style new line
- header row, Unix-style new line
- header row, Unix-style new line, quoted
- header row, Unix-style new line, escape
- header row, Unix-style new line, tab-delimited
- without specifying all columns

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

	country_code	country_name	year	population
1	LU	Luxembourg	2017	594130

SQL On Demand – Querying CSV File

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

```

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

```

	country_code	country_name	year	population
1	LU	Luxembourg	2017	594130

Read CSV file - without specifying all columns

```

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

```

	countries
1	228

SQL On Demand – Querying folders

Overview

Uses OPENROWSET function to access data from multiple files or folders

Benefits

Offers reading multiple files/folders through usage of wildcards

Offers reading specific file/folder

Supports use of multiple wildcards

```

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

```

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

SQL On Demand – Querying folders

Read all files from multiple folders

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

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

Read subset of files in folder

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

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

SQL On Demand – Querying specific files

Overview

`filename` – Provides file name that originates row result

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

Benefits

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

Example of filename function

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

GROUP BY r.filename()

ORDER BY [filename]

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

SQL On Demand – Querying specific files

Example of filepath function

```

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

```

```

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

```

```

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

```

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

SQL On Demand – Querying Parquet files

Overview

Uses OPENROWSET function to access data

Benefits

Ability to specify column names of interest

Offers auto reading of column names and data types

Provides target specific partitions using filepath function

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

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

SQL On Demand – Creating views

Overview

Create views using SQL On Demand queries

Benefits

Works same as standard views

```
USE [mydbname]
GO

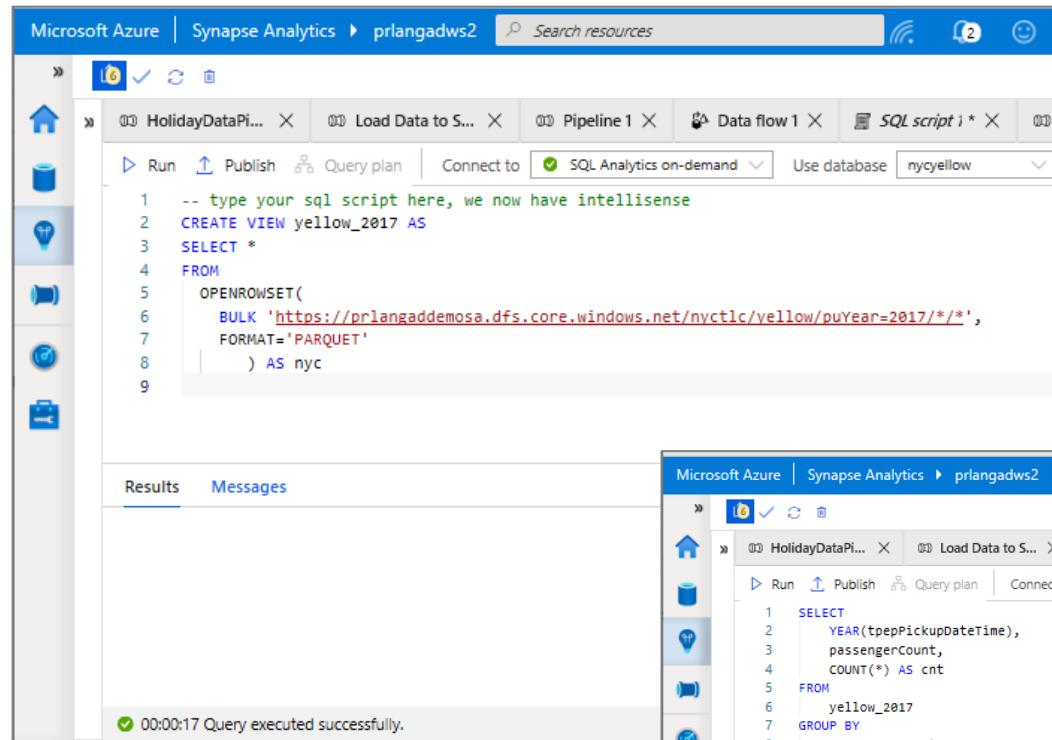
IF EXISTS(select * FROM sys.views where name = 'populationView')
DROP VIEW populationView
GO

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

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

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

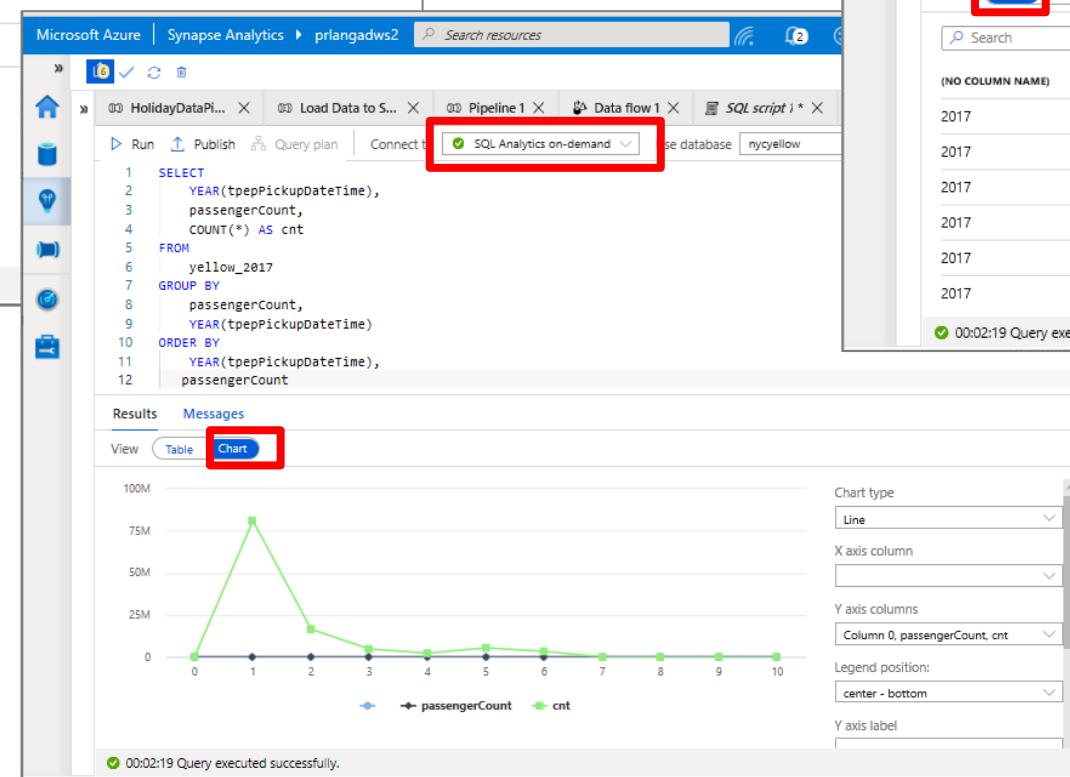
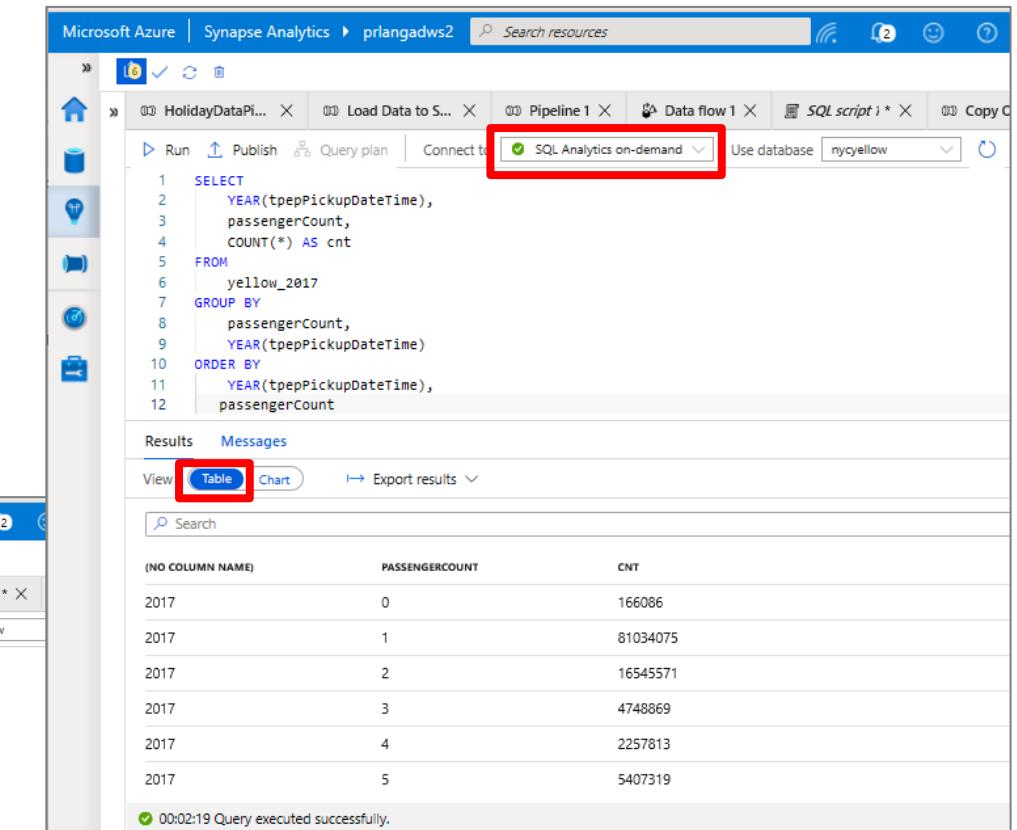
SQL On Demand – Creating views



```
-- type your sql script here, we now have intellisense
CREATE VIEW yellow_2017 AS
SELECT *
FROM
OPENROWSET(
    BULK 'https://prlangaddemosa.dfs.core.windows.net/nyctlc/yellow/puYear=2017/*/*',
    FORMAT='PARQUET'
) AS nyc
```

Results **Messages**

00:00:17 Query executed successfully.

```
SELECT
YEAR(tpepPickupDateTime),
passengerCount,
COUNT(*) AS cnt
FROM
yellow_2017
GROUP BY
passengerCount,
YEAR(tpepPickupDateTime)
ORDER BY
YEAR(tpepPickupDateTime),
passengerCount
```

Results **Messages**

View **Table** Chart Export results

00:02:19 Query executed successfully.

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

SQL On Demand – Querying JSON files

Overview

Read JSON files and provides data in tabular format

Benefits

Supports OPENJSON, JSON_VALUE and JSON_QUERY functions

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

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

SQL On Demand – Querying JSON files

Example of JSON_VALUE function

```

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

```

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

Example of JSON_QUERY function

```

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

```

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

Create External Table As Select

Overview

Creates an external table and then exports results of the Select statement. These operations will import data into the database for the duration of the query

Steps:

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

```
-- Create a database master key if one does not already exist
CREATE MASTER KEY ENCRYPTION BY PASSWORD = 'S0me!Info'
;

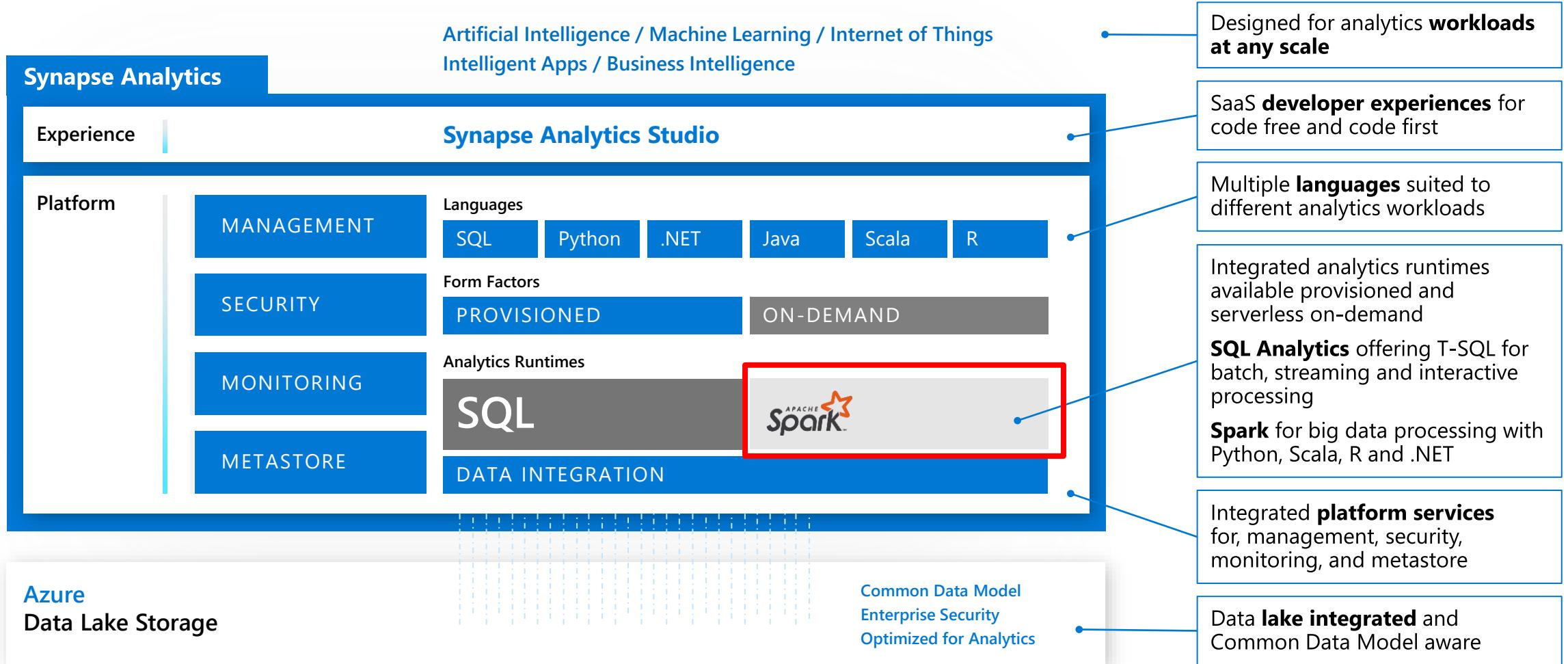
-- Create a database scoped credential with Azure storage account key as the secret.
CREATE DATABASE SCOPED CREDENTIAL AzureStorageCredential
WITH
    IDENTITY = '<my_account>'
, SECRET = '<azure_storage_account_key>'
;
-- Create an external data source with CREDENTIAL option.
CREATE EXTERNAL DATA SOURCE MyAzureStorage
WITH
(
    LOCATION = 'wasbs://daily@logs.blob.core.windows.net/'
, CREDENTIAL = AzureStorageCredential
, TYPE = HADOOP
)
-- Create an external file format
CREATE EXTERNAL FILE FORMAT MyAzureCSVFormat
WITH (FORMAT_TYPE = DELIMITEDTEXT,
      FORMAT_OPTIONS(
          FIELD_TERMINATOR = ',',
          FIRST_ROW = 2))
--Create an external table
CREATE EXTERNAL TABLE dbo.FactInternetSalesNew
WITH(
    LOCATION = '/files/Customer',
    DATA_SOURCE = MyAzureStorage,
    FILE_FORMAT = MyAzureCSVFormat
)
AS SELECT T1.* FROM dbo.FactInternetSales T1 JOIN dbo.DimCustomer T2
ON ( T1.CustomerKey = T2.CustomerKey )
OPTION ( HASH JOIN );
```



Azure Synapse Analytics Spark

Azure Synapse Analytics

Limitless analytics service with unmatched time to insight





Azure Synapse Apache Spark - Summary

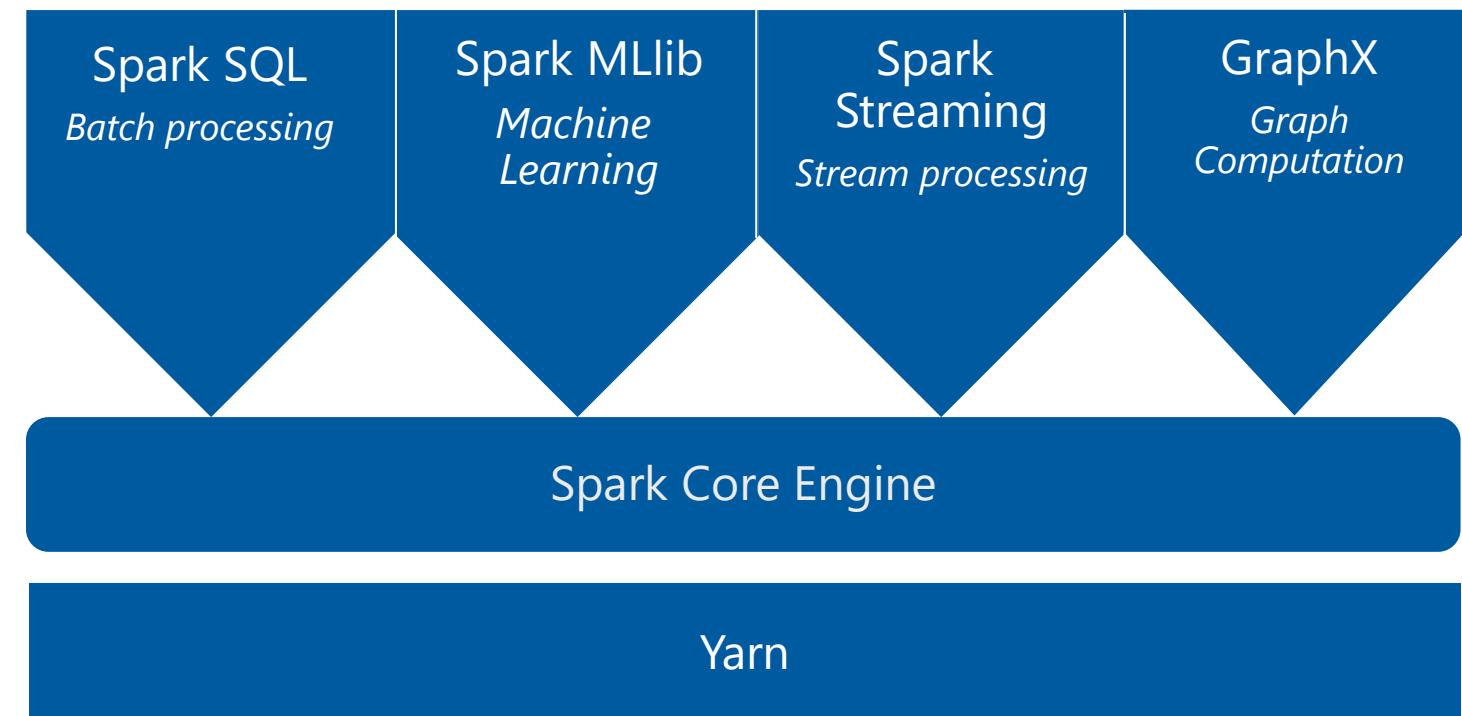
- **Apache Spark 2.4 derivation**
 - Linux Foundation Delta Lake 0.4 support
 - .Net Core 3.0 support
 - Python 3.6 + Anacondas support
- **Tightly coupled to other Azure Synapse services**
 - Integrated security and sign on
 - Integrated Metadata
 - Integrated and simplified provisioning
 - Integrated UX including interact based notebooks
 - Fast load of SQL Analytics pools
- **Core scenarios**
 - Data Prep/Data Engineering/ETL
 - Machine Learning via Spark ML and Azure ML integration
 - Extensible through library management
- **Efficient resource utilization**
 - Fast Start
 - Auto scale (up and down)
 - Auto pause
 - Min cluster size of 3 nodes
- **Multi Language Support**
 - .Net (C#), PySpark, Scala, Spark SQL, Java

Apache Spark

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

Spark Unifies:

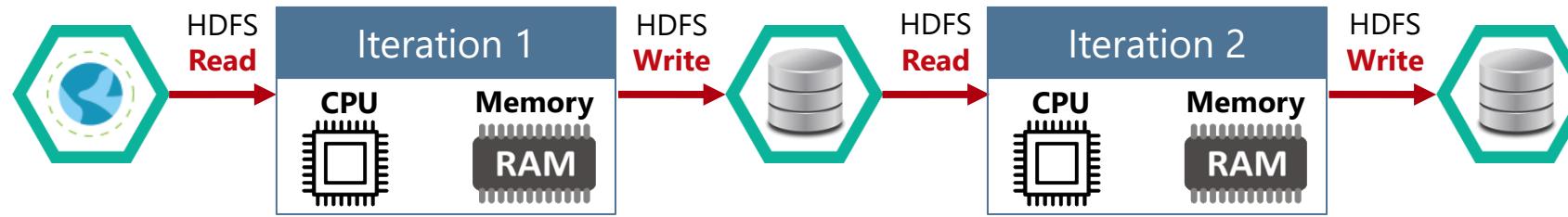
- Batch Processing
- Interactive SQL
- Real-time processing
- Machine Learning
- Deep Learning
- Graph Processing



<http://spark.apache.org>

Motivation for Apache Spark

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

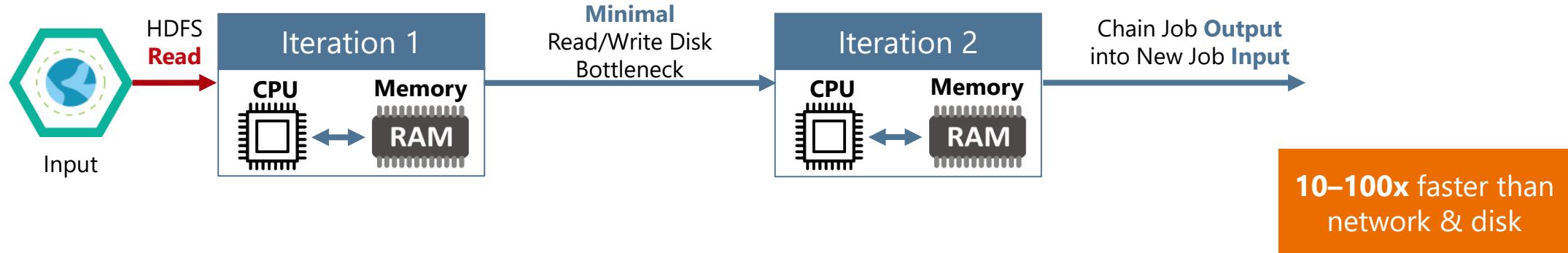


Motivation for Apache Spark

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

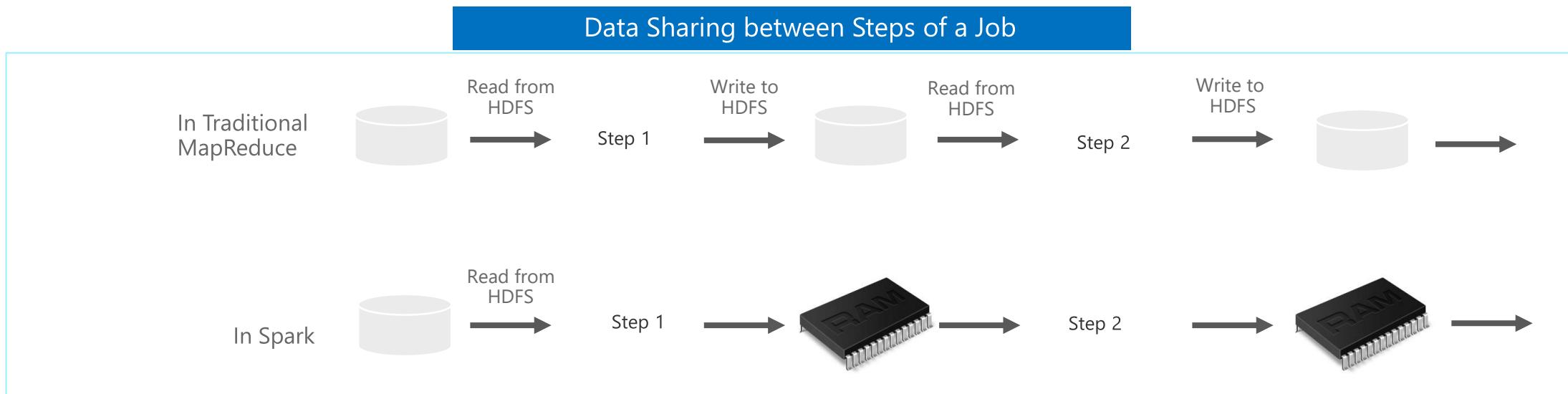


Solution: Keep data **in-memory** with a new distributed execution engine



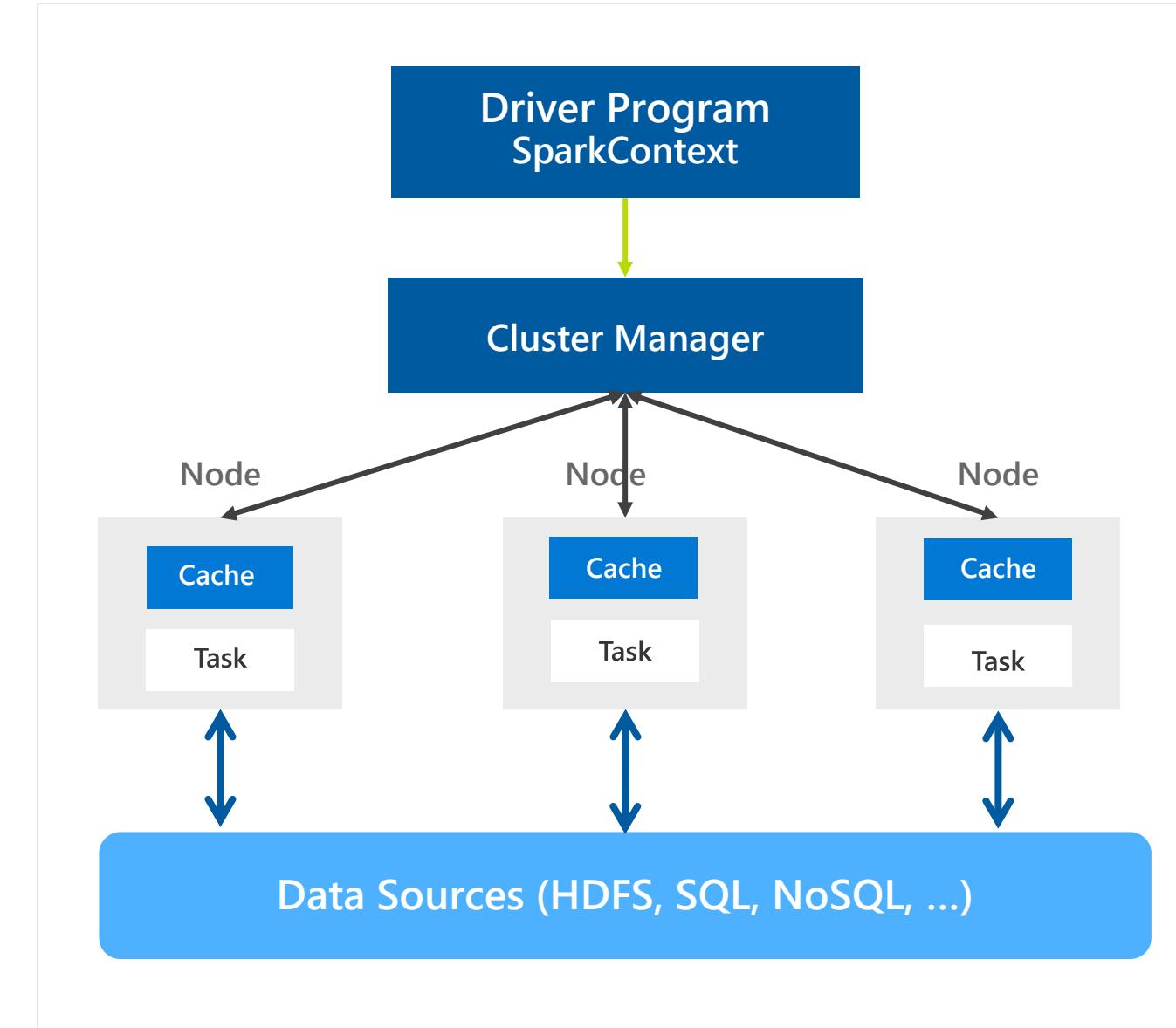
What makes Spark fast

- **In-memory cluster computing:** Spark provides primitives for *in-memory* cluster computing. A Spark job can *load and cache* data into memory and query it repeatedly (iteratively) much quicker than disk-based systems.
- **Scala Integration:** Spark integrates into the Scala programming language, letting you manipulate distributed datasets like local collections. No need to structure everything as map and reduce operations
- **Faster Data-sharing:** Data-sharing between operations is faster as data is in-memory:
 - In (traditional) Hadoop data is shared through HDFS which is expensive. HDFS maintains three replicas.
 - Spark stores data in-memory *without any replication*.



General Spark Cluster Architecture

- 'Driver' runs the user's 'main' function and executes the various parallel operations on the worker nodes.
- The results of the operations are collected by the driver
- The worker nodes read and write data from/to Data Sources including HDFS.
- Worker node also cache transformed data in memory as RDDs (Resilient Data Sets).
- Worker nodes and the Driver Node execute as VMs in public clouds (AWS, Google and Azure).



Spark Component Features

Spark SQL

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

Mlib/SparkML

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

Spark Streaming

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

GraphX

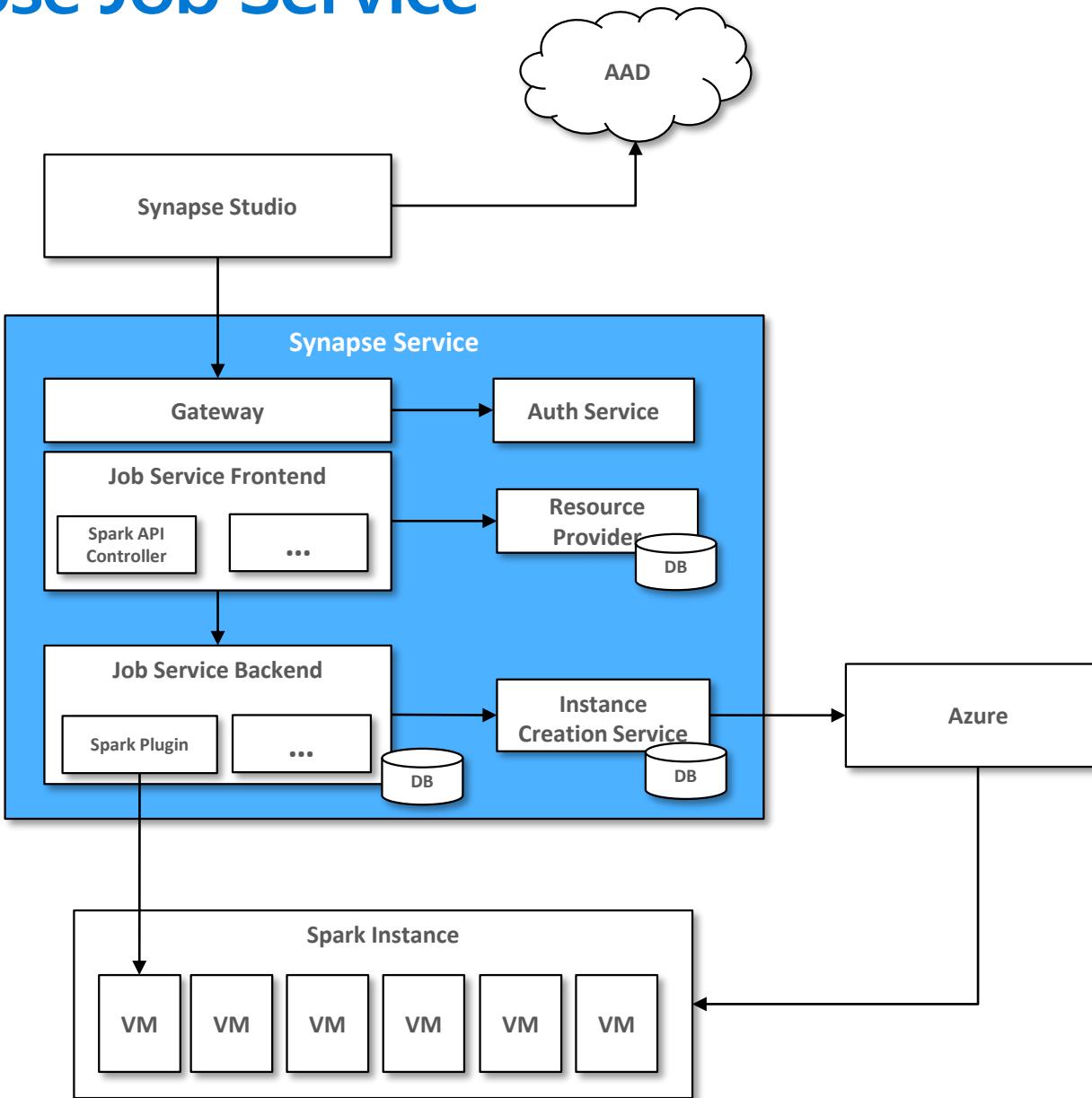
- Represent and analyze systems represented by graph nodes
- Trace interconnections between graph nodes
- Applicable to use cases in transportation, telecommunications, road networks, modeling personal relationships, social media, etc.



Azure Synapse Apache Spark

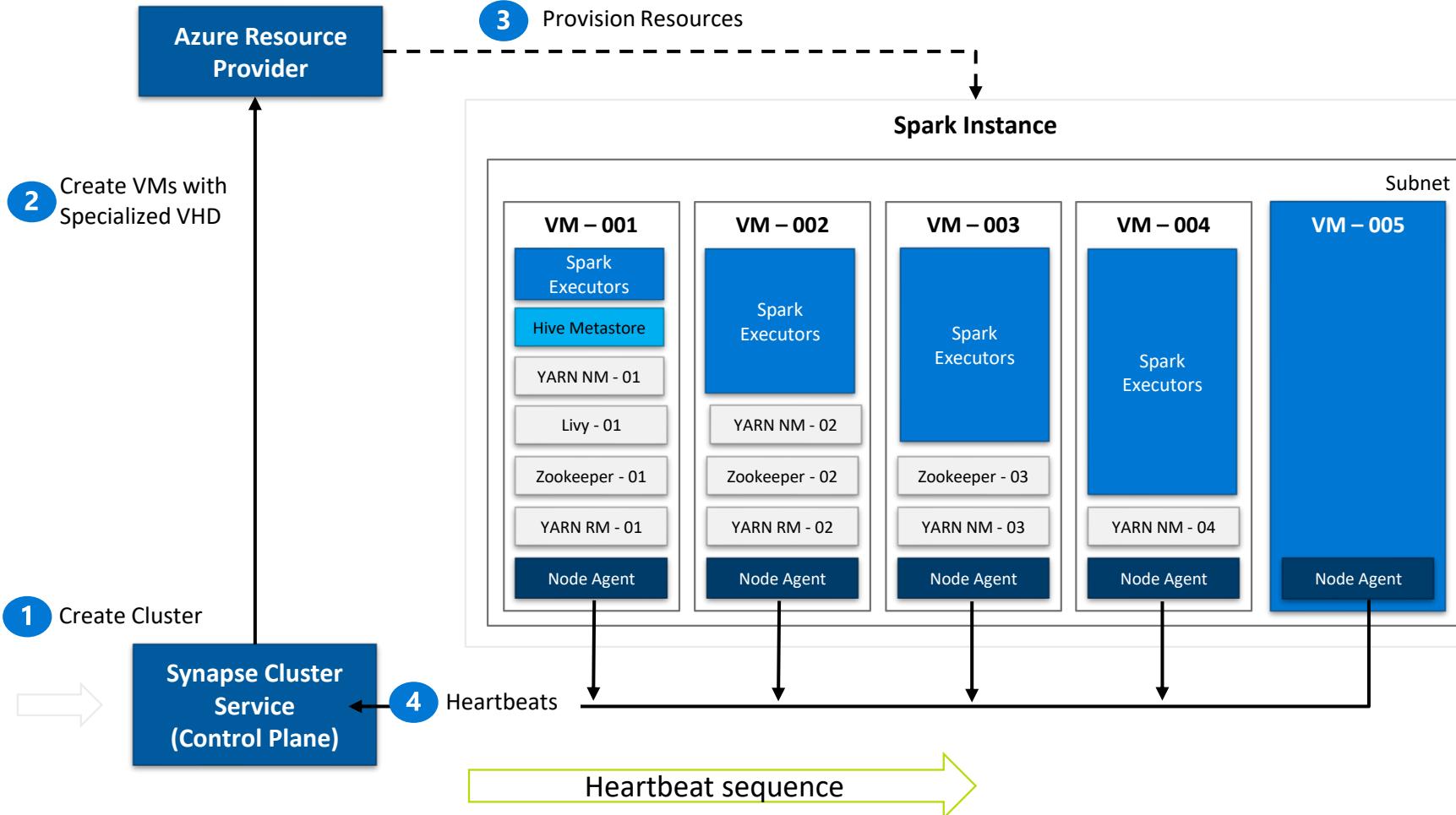
Architecture Overview

Synapse Job Service



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

Synapse Spark Instances



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

Creating a Spark pool (1 of 2)

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

Basic Settings – Minimum details required from user

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

Create Apache Spark pool

Basics * Additional settings * Tags Summary

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

Apache Spark pool details

Name your Apache Spark pool and choose its initial settings.

Apache Spark pool name *

Enter Apache Spark pool name

Node size family

MemoryOptimized

Node size *

Medium (8 vCPU / 64 GB)

Autoscale * ⓘ

Enabled Disabled

Number of nodes *

3 40

Only required field from user

Default Settings

Creating a Spark pool (2 of 2) - optional

Additional Settings offer optional settings to customize Spark pool

Customize component versions, auto-pause

Import libraries by providing text file containing library name and version

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

Create Apache Spark pool

Basics * Additional settings * Tags Summary

Customize additional configuration parameters including autoscale and component versions.

Auto-pause

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

Auto-pause * ⓘ Enabled Disabled

Number of minutes idle * 15

Component versions

Select the Apache Spark version for your Apache Spark pool.

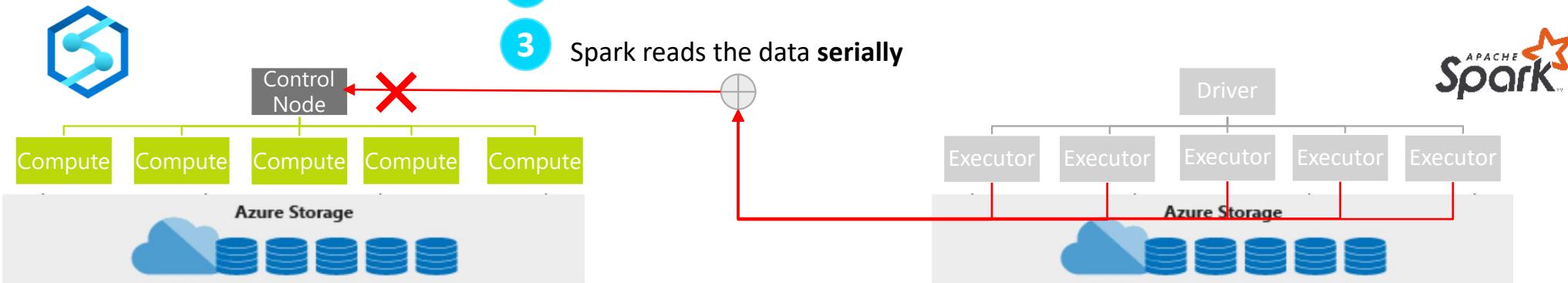
Apache Spark *	2.4
Python	3.6.1
Scala	2.11.12
Java	1.8.0_222
.NET Core	3.0
.NET for Apache Spark	0.6.0
Delta Lake	0.4.0

Packages

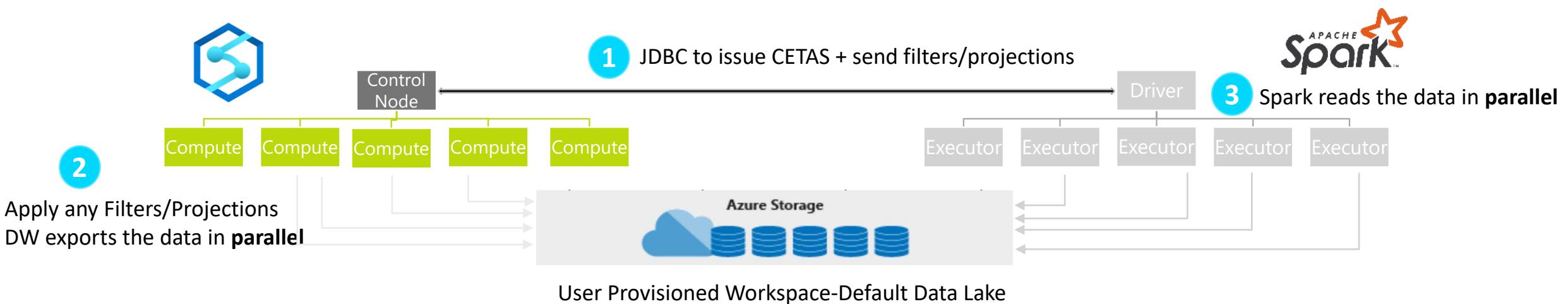
Upload environment configuration file ("PIP freeze" output).

File upload Upload

Existing Approach: JDBC



New Approach: JDBC and Polybase



Code-Behind Experience

Existing Approach

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

New Approach

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

Create Notebook on files in storage

The screenshot illustrates the process of creating a new notebook from a file stored in Azure Storage.

Left Panel (Storage View):

- Shows the Azure portal navigation bar: Microsoft Azure | Synapse Analytics > prlangadw2.
- The "Data" section is selected.
- Storage accounts listed: prlangaddemosa (Primary), nyctic, prlangaddemosa, tmpcontainer, wwidporters.
- A red box highlights the "New notebook" option in the context menu for a parquet file named "part-00133-tid-210938564719836543-aea5b543-5e83-46f7-869f72c50b05d-15253-1.c000.snappy.parquet".

Bottom Panel (Notebook View):

- The notebook interface shows a cell containing PySpark code to read the parquet file.
- The command output shows the job execution status: Succeeded (Spark 2 executors 8 cores).
- The data preview shows the schema and some rows of the parquet file.

```

%pyspark
data_path = spark.read.load('abfss://nyctic@prlangaddemosa.dfs.core.windows.net/yellow/puYear=2015/puMonth=3/part-00133-tid-210938564719836543-aea5b543-5e83-46f7-869f72c50b05d-15253-1.c000.snappy.parquet')
data_path.show(10)
    
```

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

```

+-----+-----+-----+-----+-----+-----+
| vendorID | tpepPickupDateTime | tpepDropoffDateTime | passengerCount | tripDistance | puLocationId | doLocationId | startLon | startLat | endLon | endLat |
+-----+-----+-----+-----+-----+-----+
| 2 | 2015-02-28 23:53:18 | 2015-03-01 00:00:29 | 6 | 1.63 | null | null | -74.00084686279297 | 40.73069381713867 | -73.9841537475586 | 40.74470520019531 |
| 1 | N | 1 | 7.5 | 0.5 | 0.5 | 0.3 | 1.76 | 0.0 | 10.56 |
| 1 | 2015-02-28 19:21:05 | 2015-03-28 19:28:31 | 1 | 2.2 | null | null | -73.97765350341797 | 40.763160705566406 | -73.95502471923828 | 40.7860031127927 |
| 1 | N | 1 | 8.5 | 0.0 | 0.5 | 0.3 | 2.3 | 0.0 | 11.6 |
| 2 | 2015-02-28 23:53:19 | 2015-03-01 00:12:08 | 5 | 3.23 | null | null | -73.96012878417969 | 40.76215744018555 | -73.9881591796875 | 40.72818896484375 |
| 1 | N | 1 | 14.5 | 0.5 | 0.5 | 0.3 | 4.74 | 0.0 | 20.54 |
| 1 | 2015-03-28 19:21:05 | 2015-03-28 19:37:02 | 1 | 2.1 | null | null | -73.98143005371094 | 40.7815055847168 | -74.000891552734375 | 40.76177215576172 |
+-----+-----+-----+-----+-----+-----+
    
```

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

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

Search resources

Publish all Validate all Refresh Discard all

Develop + <>

Data Download... * NYCTaxi_Docs... * SeattleSafetyD... * Repro *

Cell 1

```

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

```

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

Job execution In progress Spark 1 executors 4 cores

ID	DESCRIPTION	STATUS	STAGES	TASKS	SUBMISSION TIME	DURATION
Job 0	parquet at NativeMethodAccessImpl.java:0	In progress	0/1 (1 active)		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

Develop Publish all Validate all Refresh Discard all

NYCTaxi_Docs * SeattleSafetyD... * Repro * PySpark (Python)

Cell 1 [3]

```

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

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

Cell 4 [7]

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

View results in chart format

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

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

Library Management - Python

Overview

Customers can add new python libraries at Spark pool level

Benefits

Input requirements.txt in simple pip freeze format

Add new libraries to your cluster

Update versions of existing libraries on your cluster

Libraries will get installed for your Spark pool during cluster creation

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

Constraints

The library version must exist on PyPI repository

Version downgrade of an existing library not allowed

In the Portal

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

Microsoft Azure (Preview) Restore default configuration Report a bug Search resources, services, and data

Home > nushuklasynapsewestus2 > Create Apache Spark pool

Create Apache Spark pool

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

Auto-pause * Enabled Disabled
Number of minutes idle *

Component versions
Select the Apache Spark version for your Apache Spark pool.

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

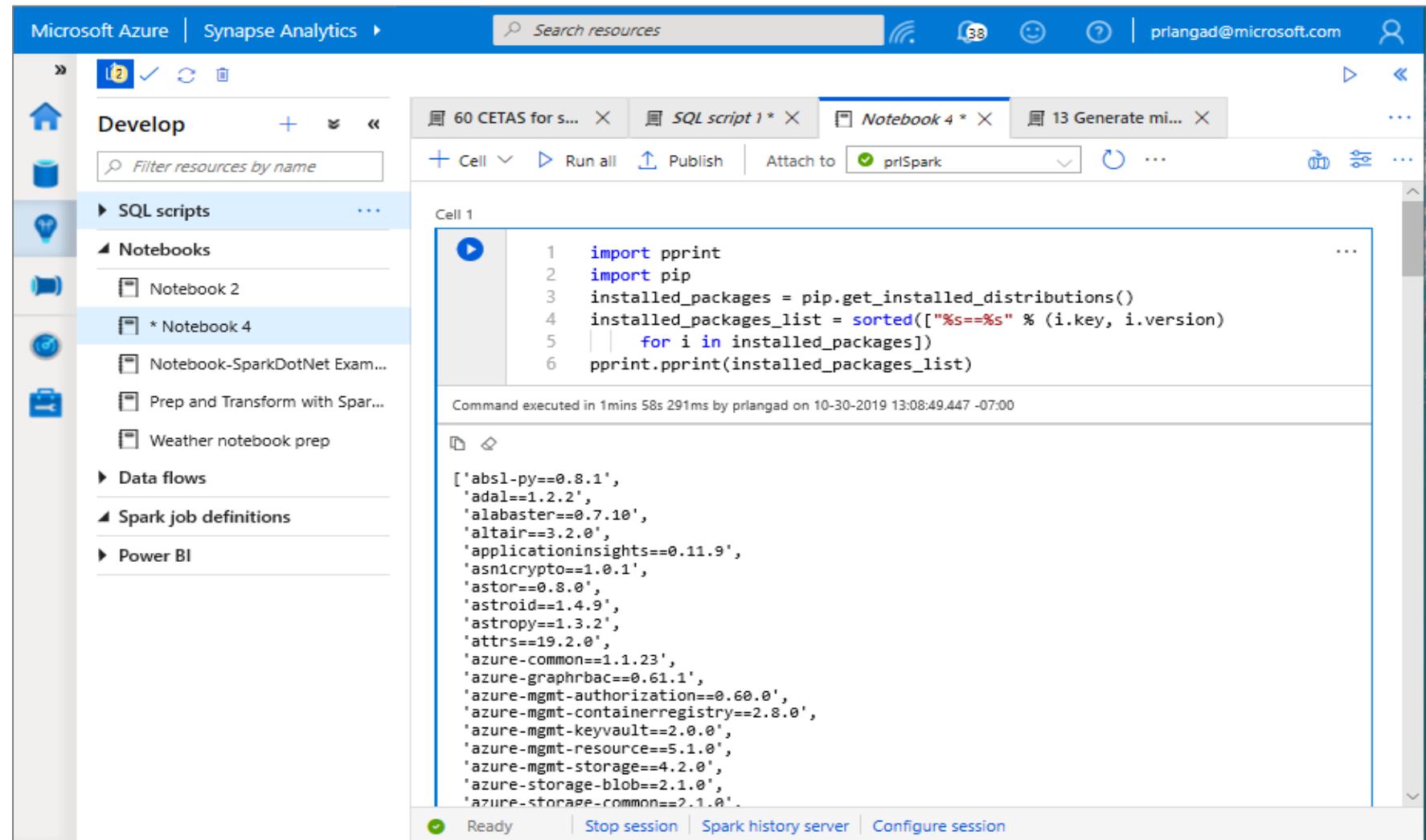
Packages
Upload environment configuration file ("PIP freeze" output).

File upload

Review + create < Previous Next: Tags >

Library Management - Python

Get list of installed libraries with version information



The screenshot shows the Microsoft Azure Synapse Analytics interface. On the left, the 'Develop' sidebar is open, showing a list of resources: SQL scripts, Notebooks, Data flows, Spark job definitions, and Power BI. A 'Notebook 4' item is selected. In the main workspace, there are four tabs at the top: '60 CETAS for s...', 'SQL script 1 * X', 'Notebook 4 * X' (which is active), and '13 Generate mi... X'. Below the tabs, a toolbar includes 'Cell', 'Run all', 'Publish', 'Attach to' (set to 'priSpark'), and other options. A code cell labeled 'Cell 1' contains the following Python script:

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

Below the code cell, a message indicates the command was executed: "Command executed in 1mins 58s 291ms by prlangad on 10-30-2019 13:08:49.447 -07:00". The output of the command is displayed in a large text area, listing numerous Python packages and their versions. At the bottom of the interface, there are buttons for 'Ready', 'Stop session', 'Spark history server', and 'Configure session'.

Spark ML Algorithms

Spark ML Algorithms

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

Microsoft Machine Learning for Apache Spark

v1.0-rc

Microsoft's Open Source
Contributions to Apache Spark



Distributed
Machine Learning



Fast Model
Deployment



Microservice
Orchestration



Multilingual Binding
Generation

www.aka.ms/spark

 [Azure/mmlspark](https://github.com/Azure/mmlspark)

Synapse Notebook: Connect to AML workspace

The screenshot shows a Microsoft Azure Synapse Analytics notebook interface. The left sidebar shows 'Develop' resources like SQL scripts, notebooks, Data flows, Spark job definitions, and Power BI. A red annotation 'Simple code to connect workspace' with a red arrow points to Cell 5.

Check the Azure ML Core SDK Version to Validate Your Installation

Cell 3

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

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

SDK Version: 1.0.69

Connect to Azure Workspace

Cell 5

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

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

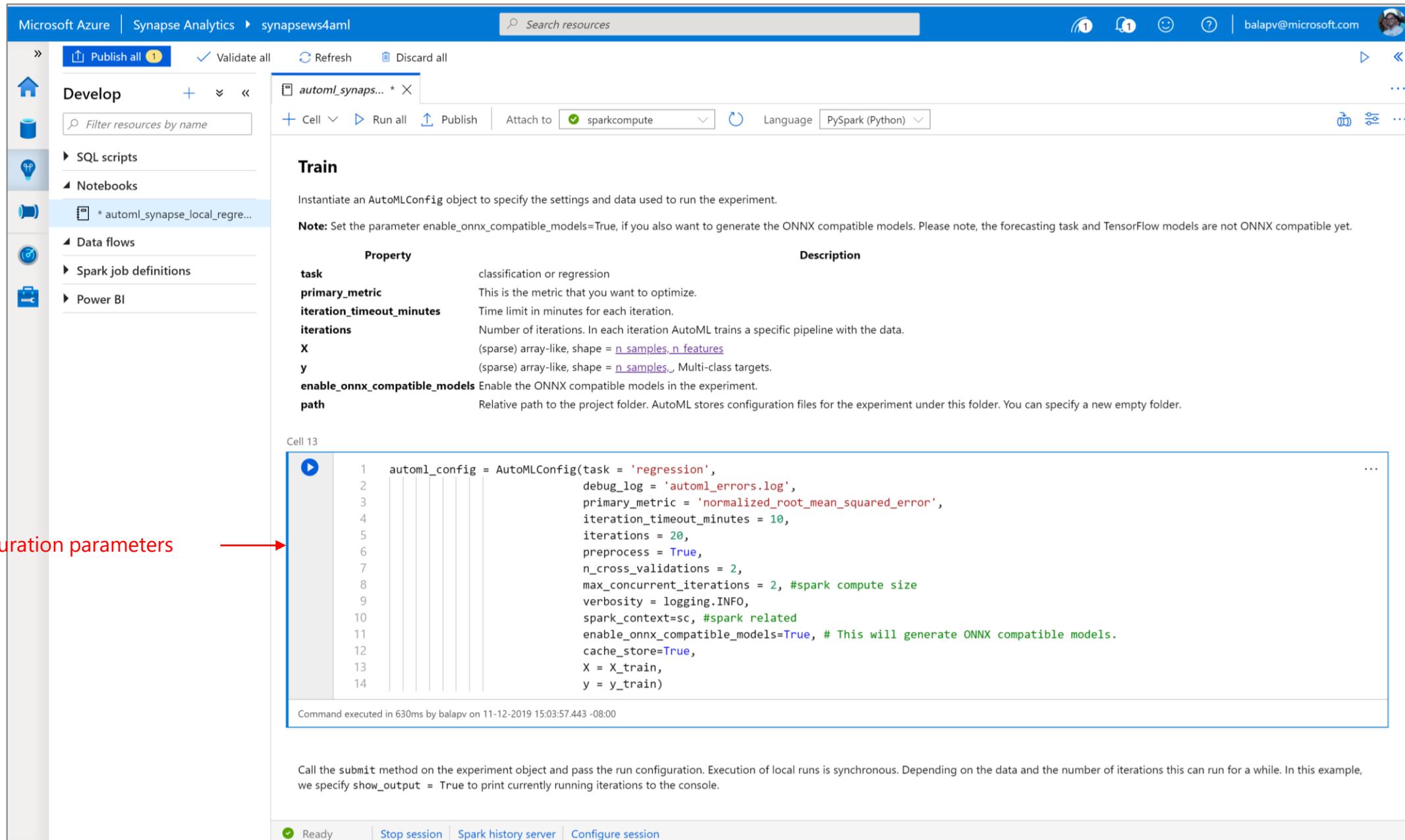
AML-WS-synapse westus2 balapv-synapse-rg

Cell 6

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

Running | Stop session | Spark history server | Configure session

Synapse Notebook: Configure AML job to run on Synapse



The screenshot shows the Microsoft Azure Synapse Analytics notebook interface. The left sidebar shows 'Develop' resources like SQL scripts, Notebooks, Data flows, Spark job definitions, and Power BI. A red arrow points from the text 'Configuration parameters' to the code cell below.

Train

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

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

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

Cell 13

```

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

```

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

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

Ready | Stop session | Spark history server | Configure session

Synapse Notebook: Run AML job

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

Publish all 1 | Validate all | Refresh | Discard all

Develop | Filter resources by name | automl_synaps... * X | Attach to sparkcompute | Language PySpark (Python)

Run AutoML job

Cell 15

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

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

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

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

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

Get Azure Portal URL for Monitoring Runs

Running | Stop session | Spark history server | Configure session

ML job execution result

**Industry-leading security
and compliance**

Enterprise-grade security



Defense-in-Depth

Industry-leading compliance



ISO 27001



SOC 1 Type 2



SOC 2 Type 2



PCI DSS Level 1



Cloud Controls Matrix



ISO 27018



Content Delivery and Security Association



Shared Assessments



FedRAMP JAB P-ATO



HIPAA / HITECH



FIPS 140-2



21 CFR Part 11



FERPA



DISA Level 2



CJIS



IRS 1075 / ITAR-ready



European Union Model Clauses



EU Safe Harbor



United Kingdom G-Cloud



China Multi Layer Protection Scheme



China GB 18030



China CCCPPF



Singapore MTCS Level 3



Australian Signals Directorate



New Zealand GCIO



Japan Financial Services



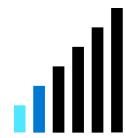
ENISA IAF

Threat Protection - Business requirements



How do we enumerate and track potential SQL vulnerabilities?

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



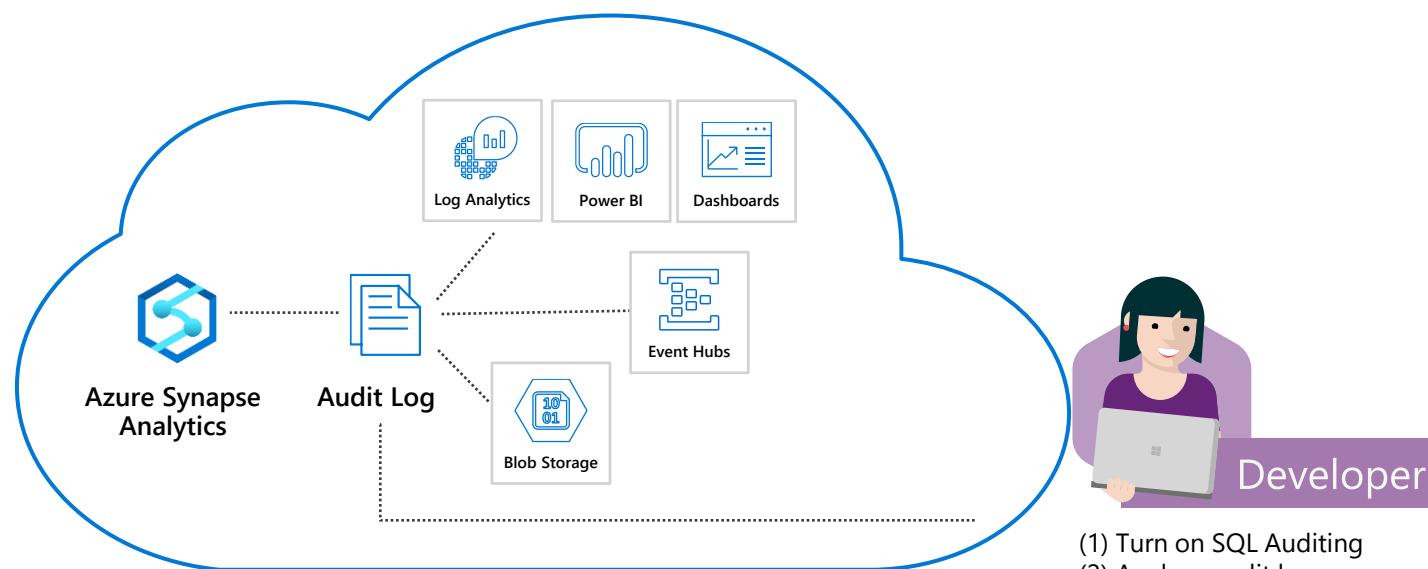
How do we discover and alert on suspicious database activity?

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



SQL auditing in Azure Log Analytics and Event Hubs

Gain insight into database audit log



The screenshot shows the Azure Log Analytics interface with the following details:

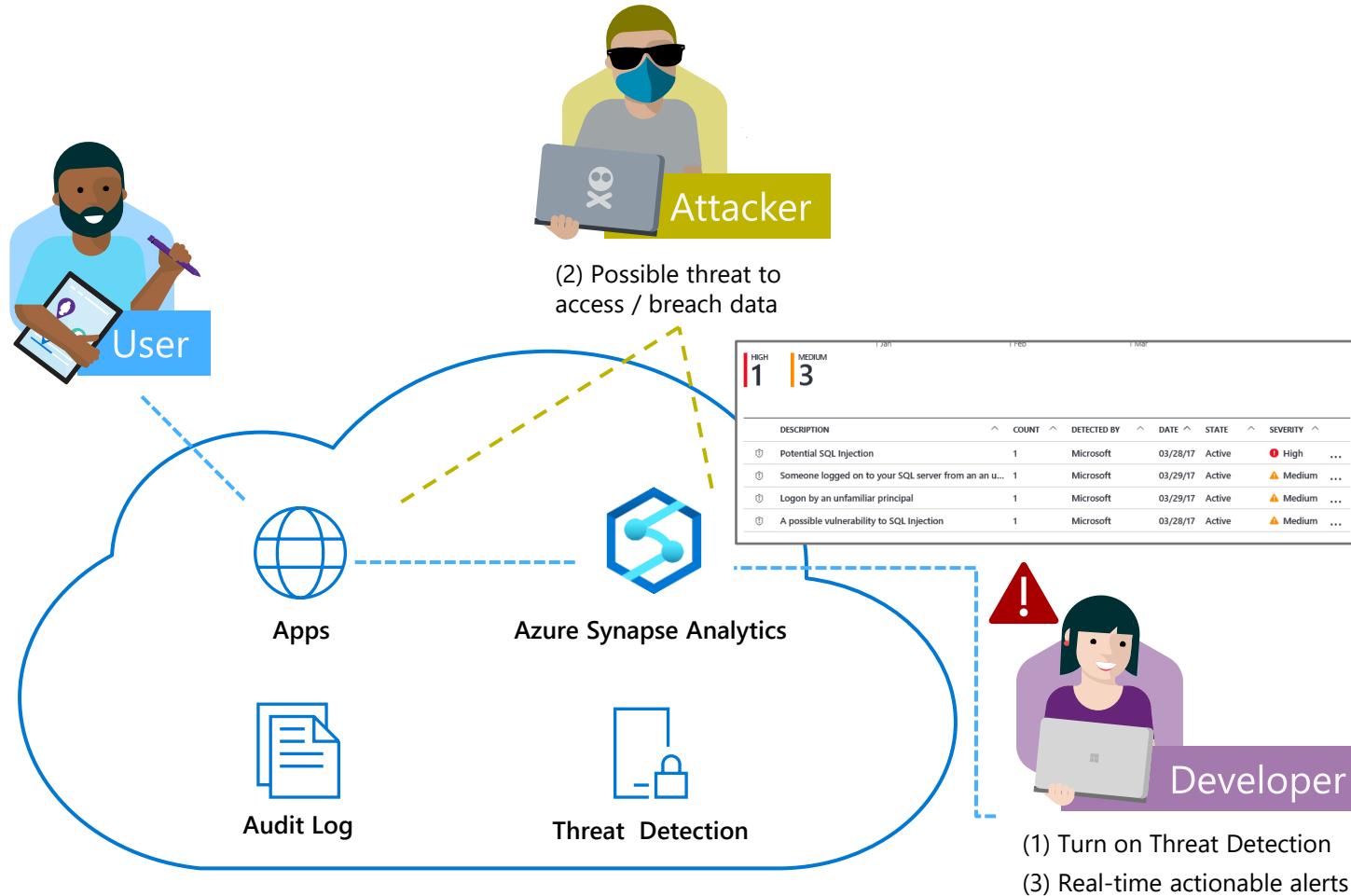
- Logs:** Logs workspace
- Search Bar:** search * | where Category == "SQLSecurityAuditEvents" | project TimeGenerated, server_principal_name_s, statement_s, affected_rows_d, SeverityLevel | sort by TimeGenerated asc
- Results:** 62 Results
- Table Headers:** TYPE (I), LOGICALSERVERNAME_S (I), CATEGORY (I)
- Table Data:** A list of audit events, including columns for TimeGenerated, server_principal_name_s, statement_s, affected_rows_d, and SeverityLevel. One row is highlighted in blue, showing the following details:

TimeGenerated	server_principal_name_s	statement_s	affected_rows_d	SeverityLevel
8/15/2018 12:00:22.521 AM	admin1	exec sp_executesql N'SELECT tbl.name AS [Name], SCHEMA_NAME(tbl...	0	0
8/15/2018 12:00:22.521 AM	admin1	exec sp_executesql N'SELECT ISNULL(HAS_PERMS_BY_NAME(QUOTED...	1	0
8/15/2018 12:00:22.521 AM	admin1	DECLARE @edition sysname; SET @edition = cast(SERVERPROPERTY(N...	4	0
8/15/2018 12:00:22.521 AM	admin1	exec sp_executesql N'SELECT CAST(ti.is_enabled AS bit) AS [isEnabled]...	0	0
8/15/2018 12:00:22.521 AM	admin1	IF OBJECT_ID ('[sys].[database_query_store_options]') IS NOT NULL BE...	1	0

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

SQL threat detection

Detect and investigate anomalous database activity



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

SQL Data Discovery & Classification

Discover, classify, protect and track access to sensitive data

The screenshot shows the Azure portal interface for SQL Data Discovery & Classification. The main dashboard displays various metrics and donut charts. Below the dashboard, a table lists columns by schema, table, column name, information type, and sensitivity label. A separate dialog box titled 'Settings - Information protection' shows a list of sensitivity labels with their descriptions and ordering options.

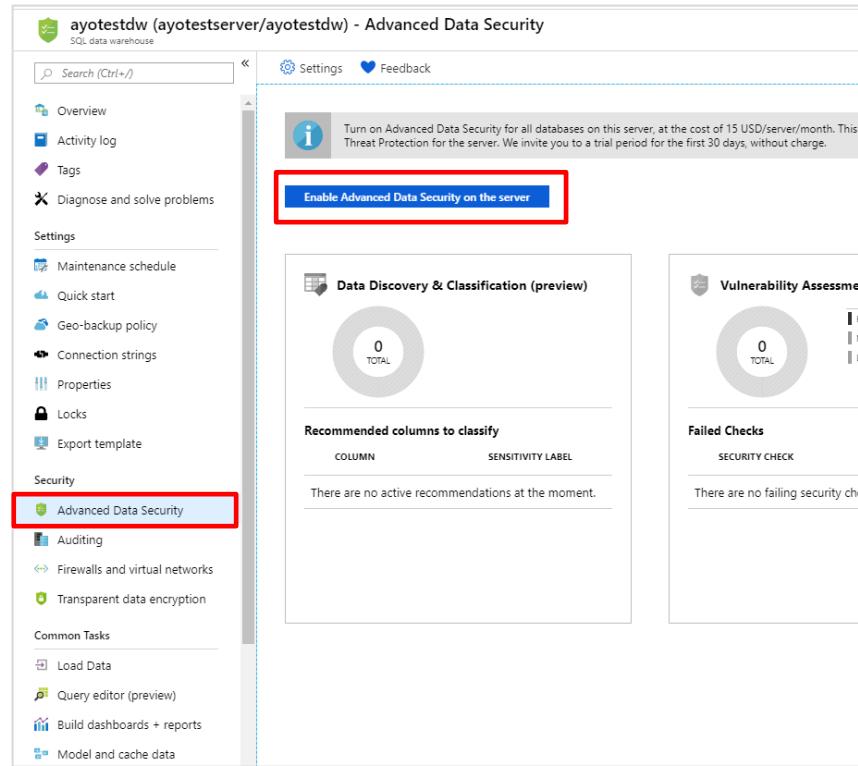
Schema	Table	Column	Information Type	Sensitivity Label
dbo	ErrorLog	UserName	Credentials	Confidential

DISPLAY NAME	DESCRIPTION
Public	Business data that is specifically prepared and approved for public consumption
General	Business data that is not intended for public consumption. However, this can be shared with...
Confidential	Sensitive business data that could cause damage to the business if shared with unauthorized...
Confidential - GDPR	Sensitive data containing personal information associated with an individual, that could b...
Highly confidential	Very sensitive business data that would cause damage to the business if it was shared with...
Highly confidential - GDPR	Sensitive data containing personal information associated with an individual, that can cau...

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

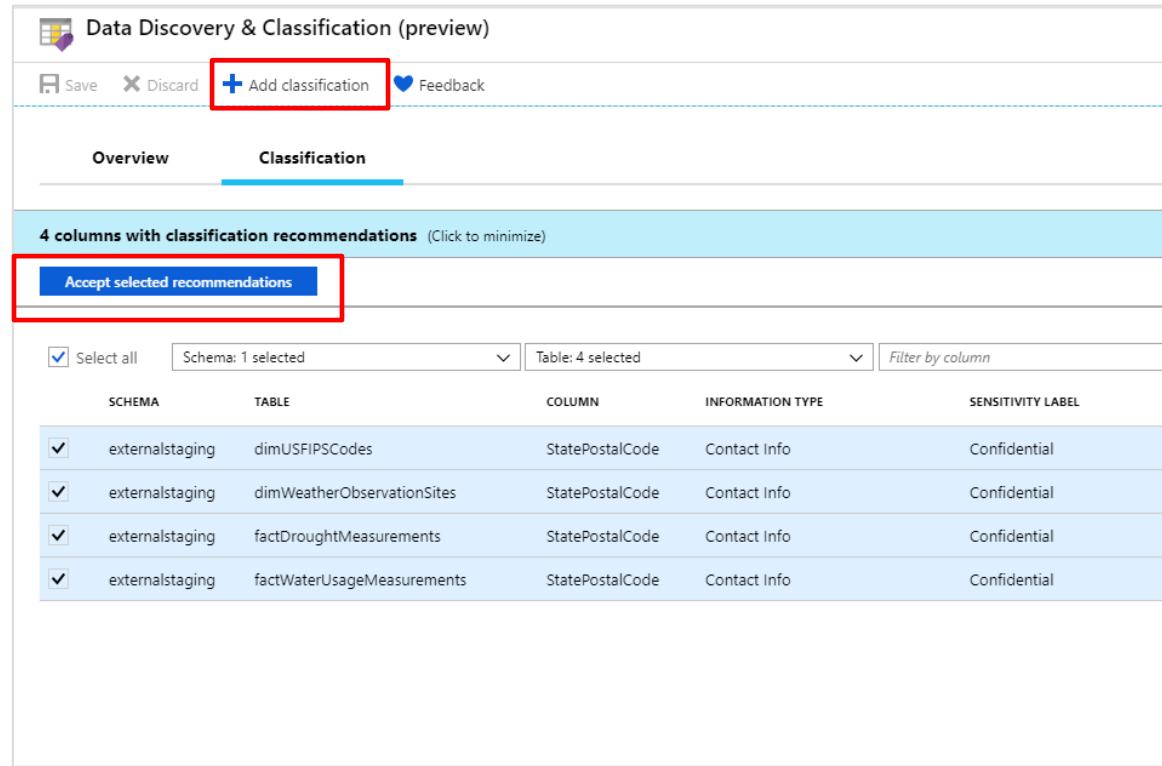
SQL Data Discovery & Classification - setup

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



The screenshot shows the 'ayotestdw (ayotestserver/ayotestdw) - Advanced Data Security' blade. On the left, there's a sidebar with 'Overview', 'Activity log', 'Tags', 'Diagnose and solve problems', 'Maintenance schedule', 'Quick start', 'Geo-backup policy', 'Connection strings', 'Properties', 'Locks', 'Export template', 'Security' (which is selected and highlighted with a red box), 'Auditing', 'Firewalls and virtual networks', and 'Transparent data encryption'. Under 'Common Tasks', there are 'Load Data', 'Query editor (preview)', 'Build dashboards + reports', and 'Model and cache data'. The main area has a heading 'Turn on Advanced Data Security for all databases on this server, at the cost of 15 USD/server/month. This includes Threat Protection for the server. We invite you to a trial period for the first 30 days, without charge.' Below it is a large blue button 'Enable Advanced Data Security on the server' with a red box around it.

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



The screenshot shows the 'Data Discovery & Classification (preview)' blade. At the top, there are 'Save', 'Discard', and a blue 'Add classification' button with a red box around it. Below that are tabs for 'Overview' and 'Classification' (which is selected and highlighted with a blue bar). A message says '4 columns with classification recommendations (Click to minimize)'. A blue button 'Accept selected recommendations' is highlighted with a red box. Below this, there are dropdown menus for 'Select all', 'Schema: 1 selected', 'Table: 4 selected', and 'Filter by column'. A table lists four columns: externalstaging.dimUSFIPSCode, externalstaging.dimWeatherObservationSites, externalstaging.factDroughtMeasurements, and externalstaging.factWaterUsageMeasurements. The table has columns for SCHEMA, TABLE, COLUMN, INFORMATION TYPE, and SENSITIVITY LABEL. All rows show 'externalstaging' as the schema, the respective table name as the table, 'StatePostalCode' as the column, 'Contact Info' as the information type, and 'Confidential' as the sensitivity label. There are checkboxes next to each row.

SCHEMA	TABLE	COLUMN	INFORMATION TYPE	SENSITIVITY LABEL
externalstaging	dimUSFIPSCode	StatePostalCode	Contact Info	Confidential
externalstaging	dimWeatherObservationSites	StatePostalCode	Contact Info	Confidential
externalstaging	factDroughtMeasurements	StatePostalCode	Contact Info	Confidential
externalstaging	factWaterUsageMeasurements	StatePostalCode	Contact Info	Confidential

SQL Data Discovery & Classification – audit sensitive data access

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

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

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

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

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

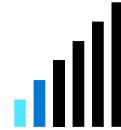
Field	Value
action_id	1176681924
additional_information	<login_information><error_code>18456</error_code><error_stat...
affected_rows	0
application_name	Net SqlClient Data Provider
audit_schema_version	1
class_type	16964
client_ip	10.0.0.4
connection_id	F1AD6457-9F40-409B-B43C-E638AAFA47902
data_sensitivity_information	
database_name	master
database_principal_id	-1
database_principal_name	
duration_milliseconds	0
event_time	2019-02-26 18:38:35.6743004
host_name	usgsvm089
is_column_permission	False
object_id	5
object_name	master

Network Security - Business requirements



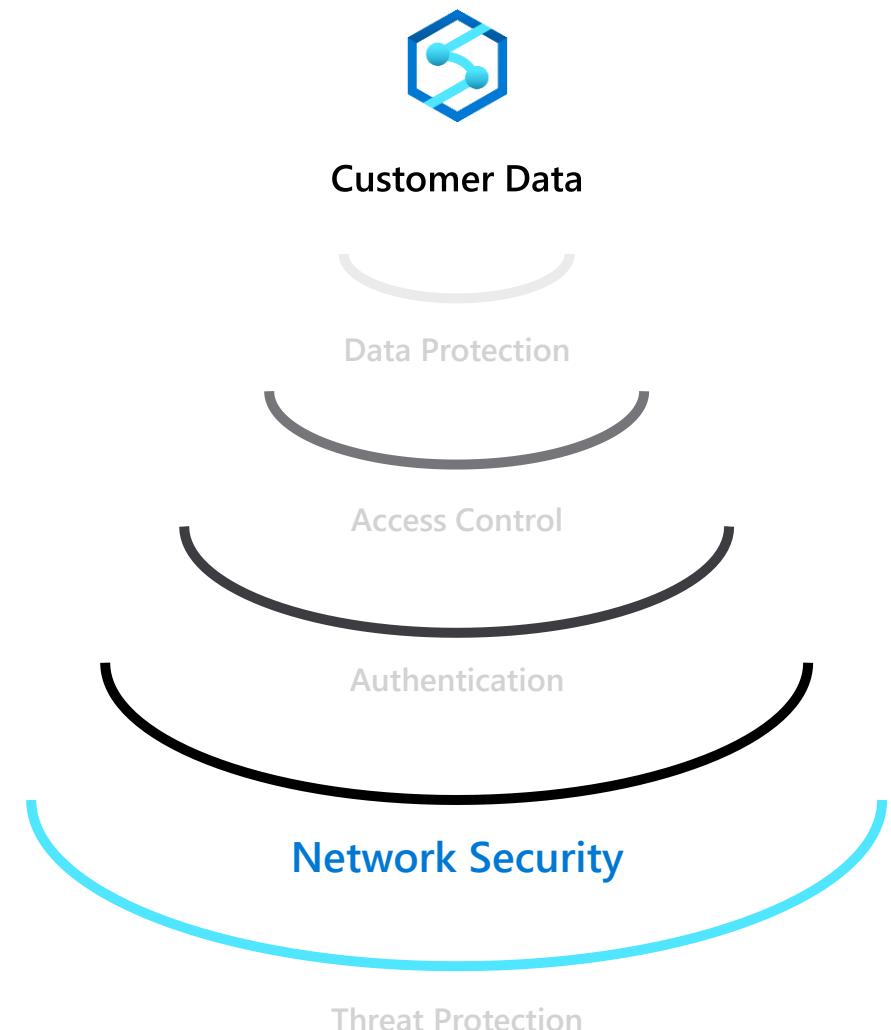
How do we implement network isolation?

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

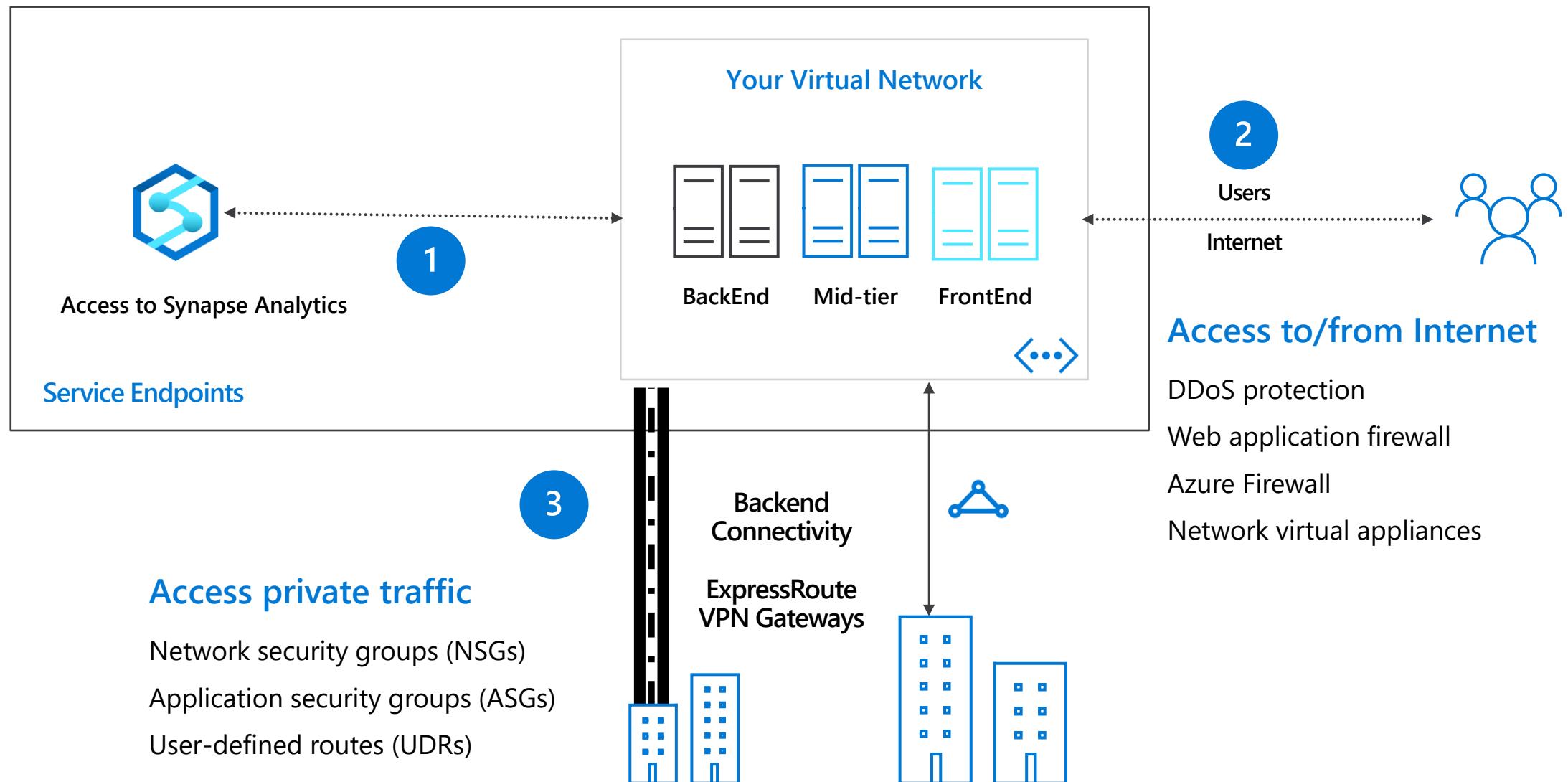


How do we achieve separation?

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



Azure networking: application-access patterns

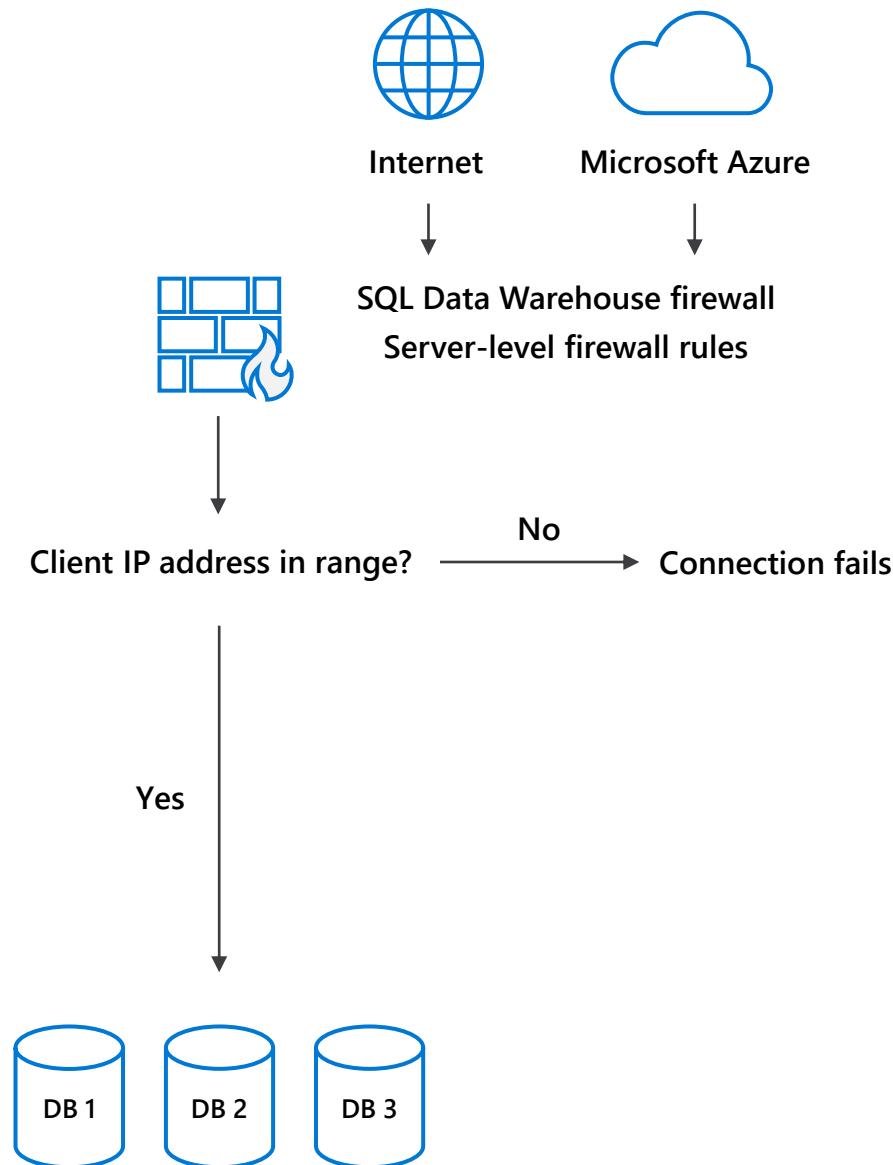


Securing with firewalls

Overview

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

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

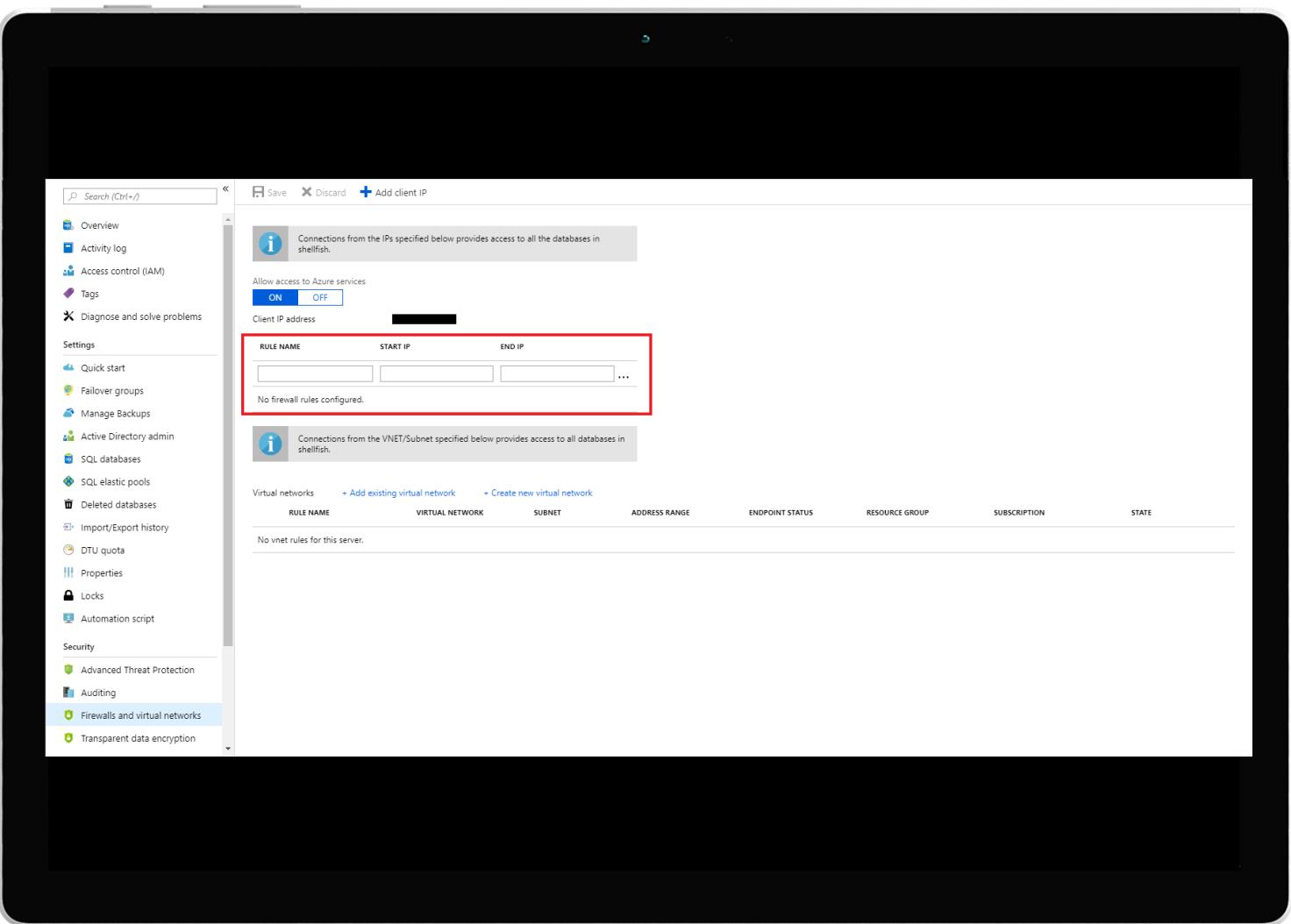


Firewall configuration on the portal

By default, Azure blocks all external connections to port 1433

Configure with the following steps:

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



Firewall configuration using REST API

Managing firewall rules through REST API must be authenticated.

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

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

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

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

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

PUT

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

DELETE

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

GET

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

Firewall configuration using PowerShell/T-SQL

Windows PowerShell Azure cmdlets

```
New-AzureRmSqlServerFirewallRule
```

```
Get-AzureRmSqlServerFirewallRule
```

```
Set-AzureRmSqlServerFirewallRule
```

Transact SQL

```
sp_set_firewall_rule
```

```
sp_delete_firewall_rule
```

```
# PS Allow external IP access to SQL DW
PS C:\> New-AzureRmSqlServerFirewallRule
          -ResourceGroupName "myResourceGroup" ` 
          -ServerName $servername ` 
          -FirewallRuleName "AllowSome" ` 
          -StartIpAddress "0.0.0.0" ` 
          -EndIpAddress "0.0.0.0"

-- T-SQL Allow external IP access to SQL DW
EXECUTE sp_set_firewall_rule
    @name = N'ContosoFirewallRule',
    @start_ip_address = '192.168.1.1',
    @end_ip_address = '192.168.1.10'
```

VNET configuration on Azure portal

Configure with the following steps:

Azure Synapse Analytics Resource:

Server name > Firewalls and virtual networks

REST API and PowerShell alternatives available

Note:

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

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

The screenshot shows the 'Firewall / Virtual Networks' settings for a SQL server named 'gm-sql-db-server-srv1'. At the top, there are 'Save' and 'Discard' buttons, and a '+ Add client IP' button. Below this is an information icon with the text: 'Connections from the IPs specified below provides access to all the databases in gm-sql-db-server-srv1.' A toggle switch labeled 'Allow access to Azure services' is set to 'OFF'. The 'Client IP address' is listed as 73.118.201.137. The main table lists two IP rules:

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

Below the table is another information icon with the text: 'Connections from the VNET/Subnet specified below provides access to all databases in gm-sql-db-server-srv1.' At the bottom, there are buttons for 'Virtual networks', '+ Add existing' (which is highlighted with a red box), and '+ Create new'. There is also a table with columns: RULE NAME, RESOURCE GROUP/VNET NAME, and SUBNET.

Authentication - Business requirements



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

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



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



Azure Active Directory authentication

Overview

Manage user identities in one location.

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

Azure Synapse Analytics

Benefits

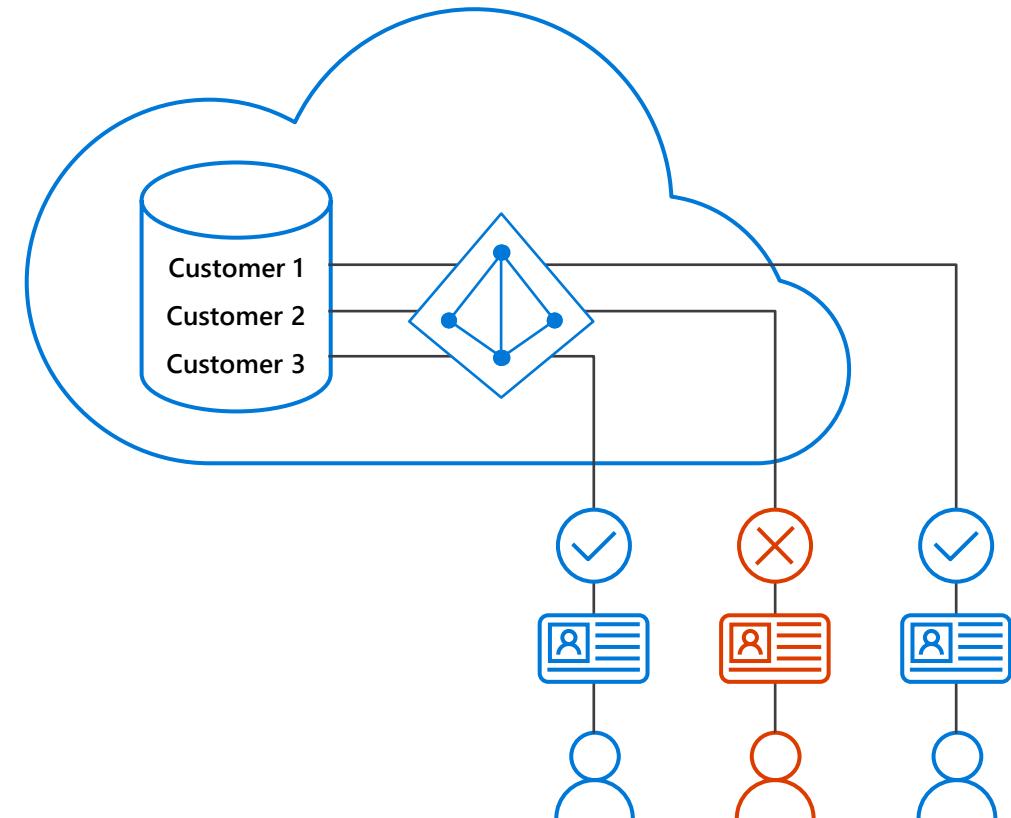
Alternative to SQL Server authentication

Limits proliferation of user identities across databases

Allows password rotation in a single place

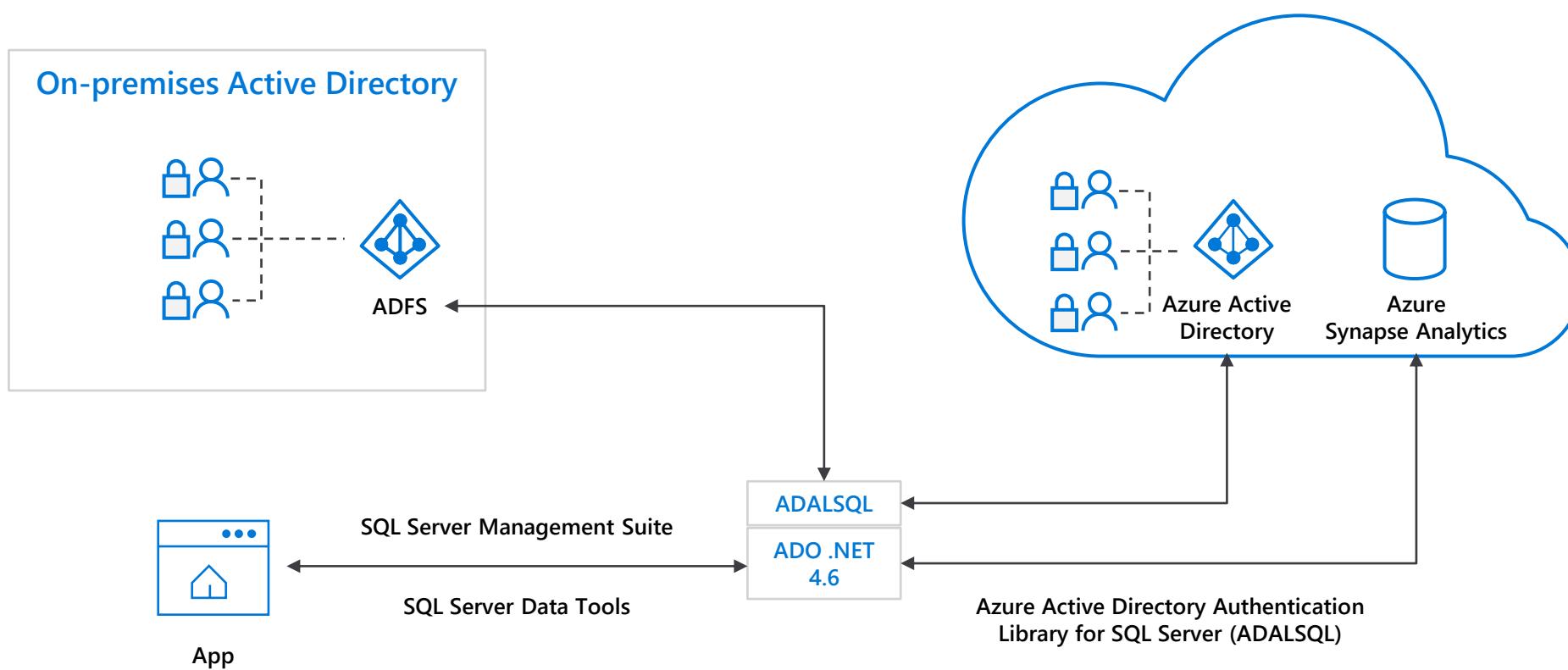
Enables management of database permissions by using external Azure Active Directory groups

Eliminates the need to store passwords



Azure Active Directory trust architecture

Azure Active Directory and Azure Synapse Analytics



SQL authentication

Overview

This authentication method uses a username and password.

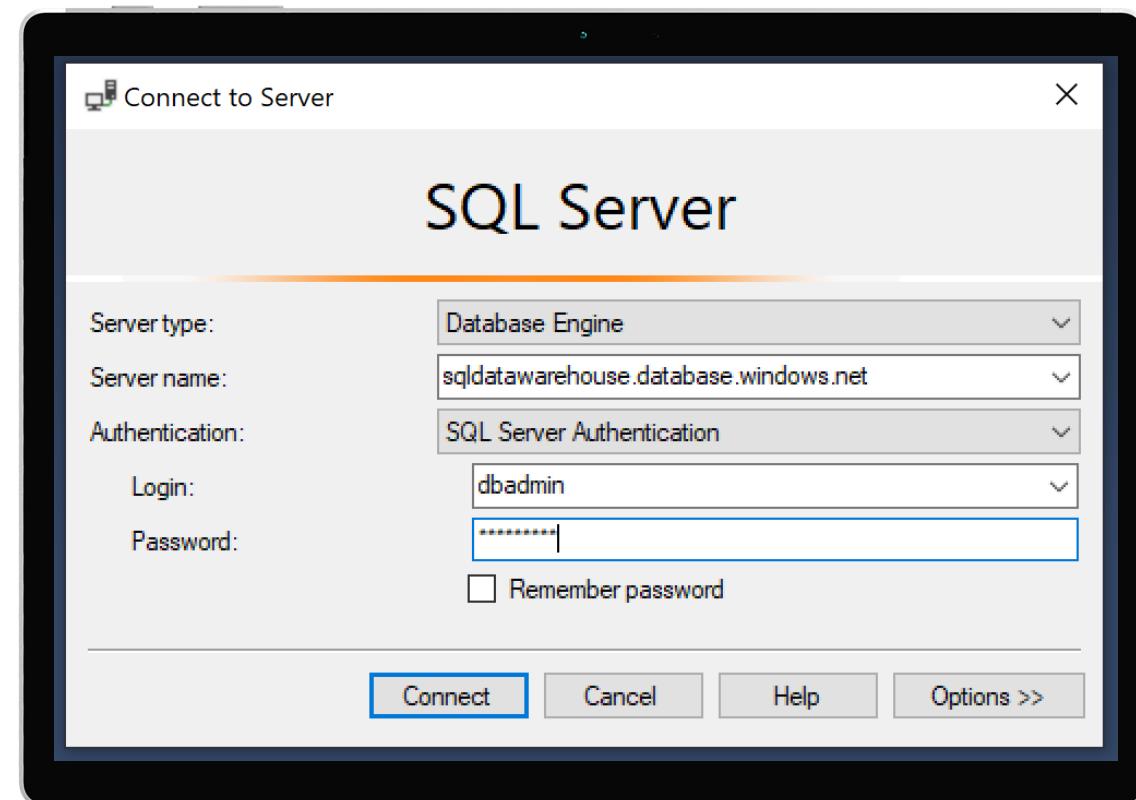
When you created the logical server for your data warehouse, you specified a "server admin" login with a username and password.

Using these credentials, you can authenticate to any database on that server as the database owner.

Furthermore, you can create user logins and roles with familiar SQL Syntax.

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

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



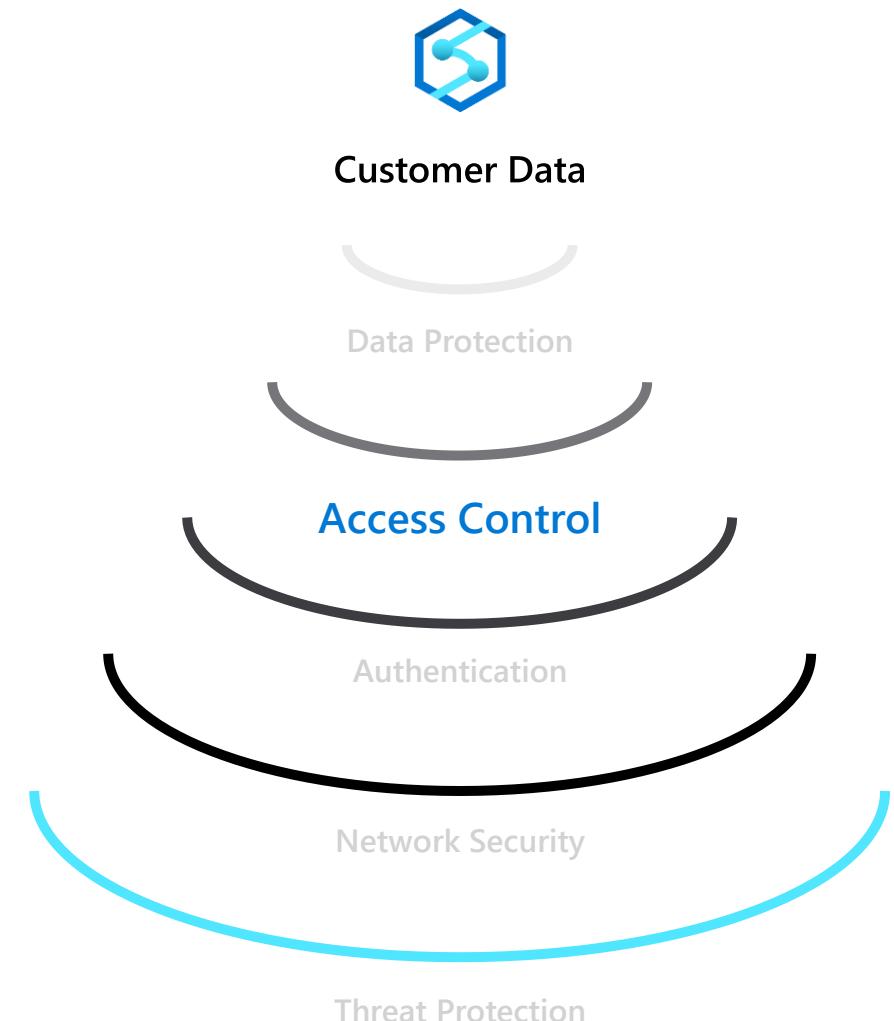
Access Control - Business requirements



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

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

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



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

Overview

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

Prevent unauthorized queries against certain tables.

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

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

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

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

Row-level security (RLS)

Overview

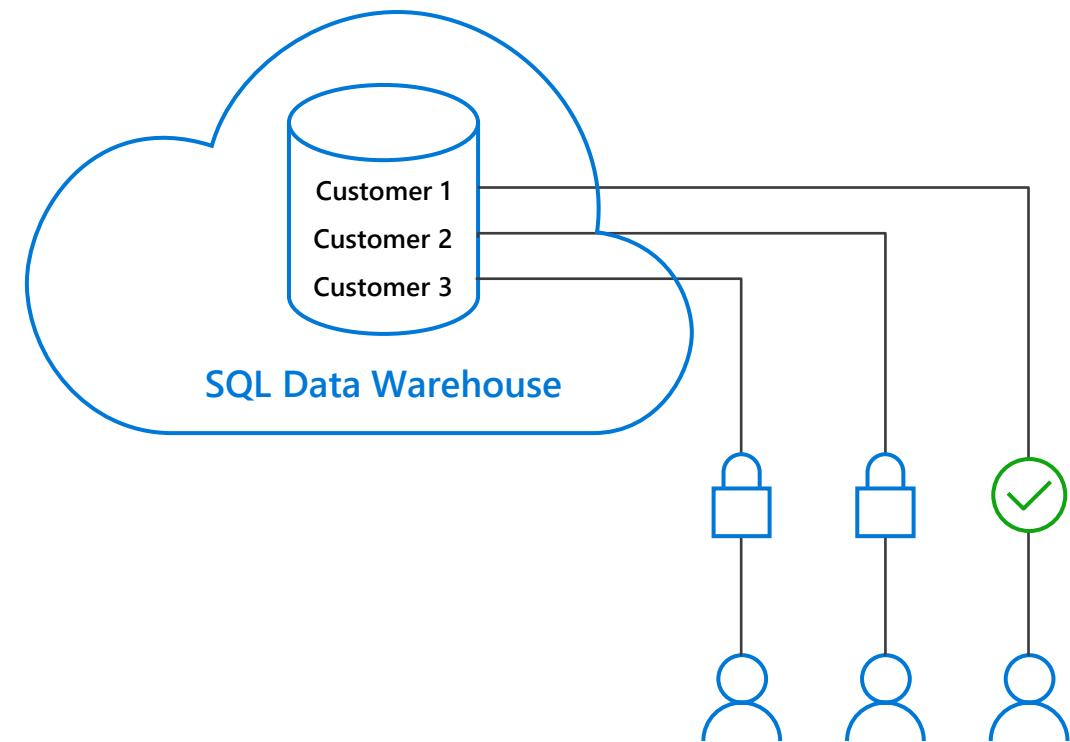
Fine grained access control of specific rows in a database table.

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

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

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

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



Row-level security

Creating policies

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

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

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

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

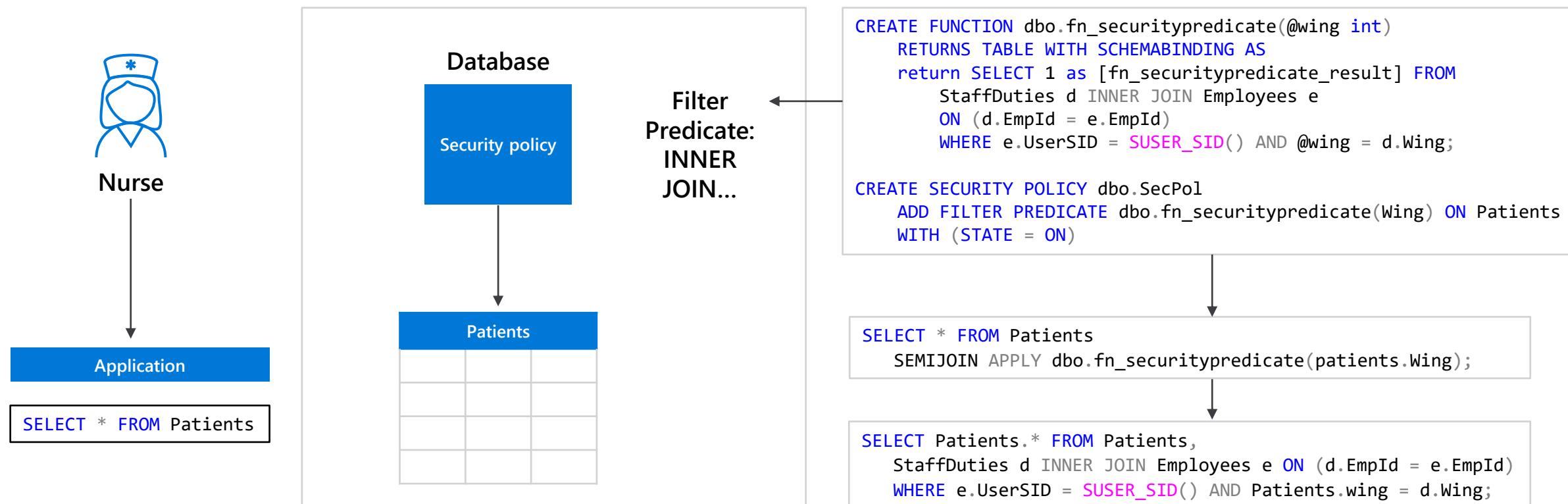
Row-level security

Three steps:

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



Policy manager



Column-level security

Overview

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

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

Administer via GRANT T-SQL statement.

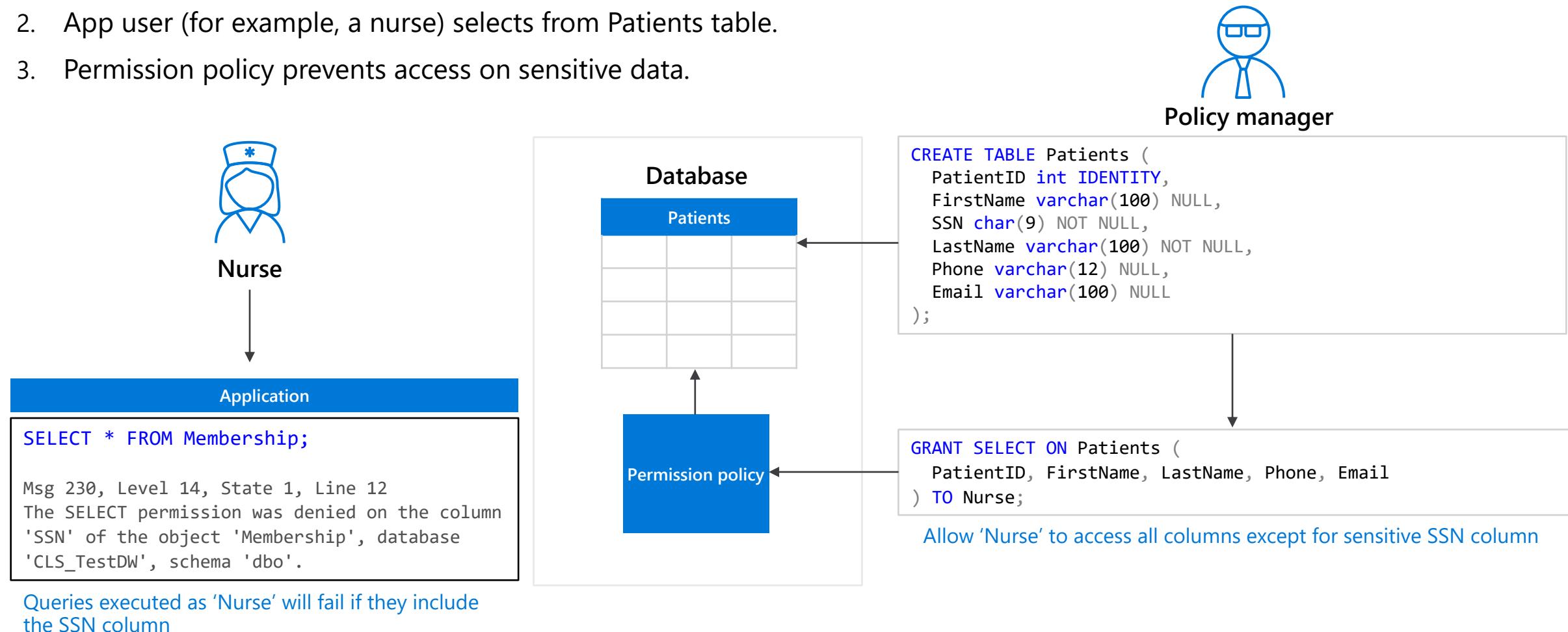
Both Azure Active Directory (AAD) and SQL authentication are supported.



Column-level security

Three steps:

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



Data Protection - Business requirements



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

What key management options do I have?



Dynamic Data Masking

Overview

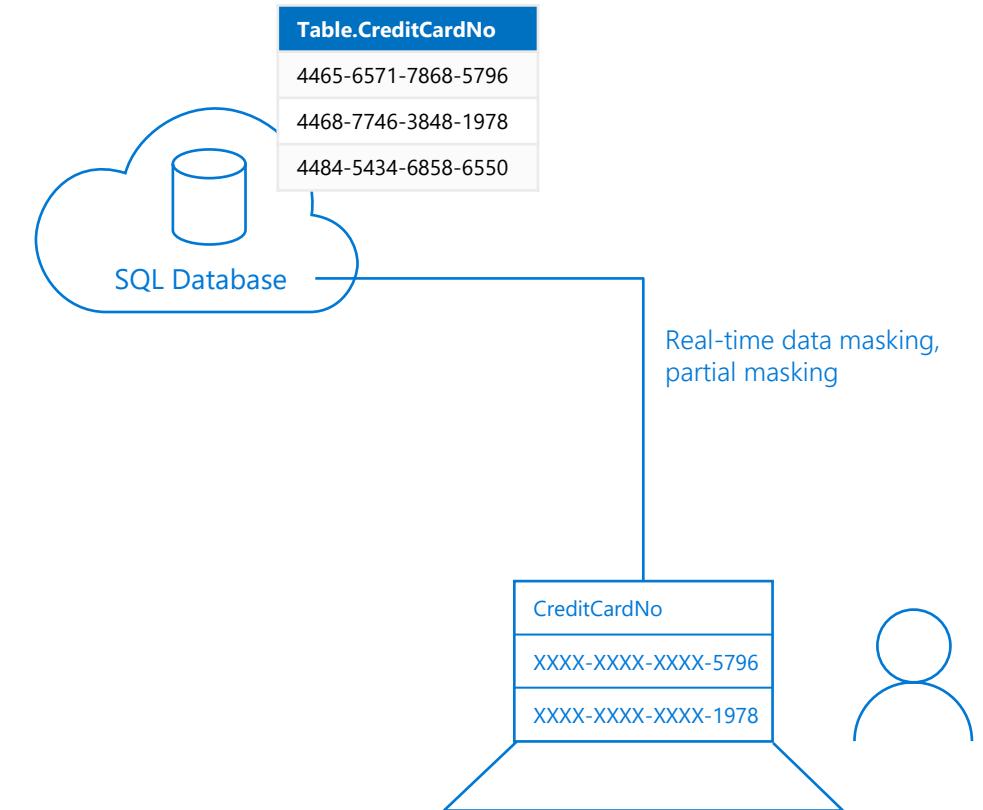
Prevent abuse of sensitive data by hiding it from users

Easy configuration in new Azure Portal

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

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

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



Dynamic Data Masking

Three steps

1. Security officer defines dynamic data masking policy in T-SQL over sensitive data in the Employee table. The security officer uses the built-in masking functions (default, email, random)
2. The app-user selects from the Employee table
3. The dynamic data masking policy obfuscates the sensitive data in the query results for non-privileged users



Security officer

```

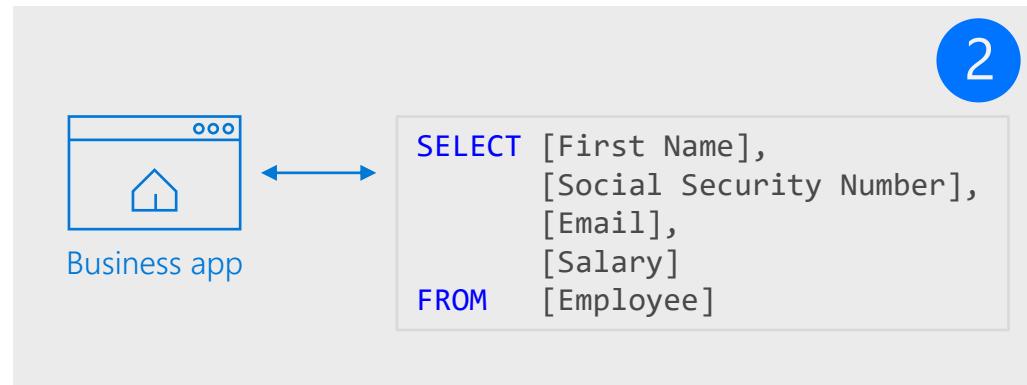
ALTER TABLE [Employee]
ALTER COLUMN [SocialSecurityNumber]
ADD MASKED WITH (FUNCTION = 'DEFAULT()')

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

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

GRANT UNMASK to admin1
    
```

1



2

Non-masked data (admin login)

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

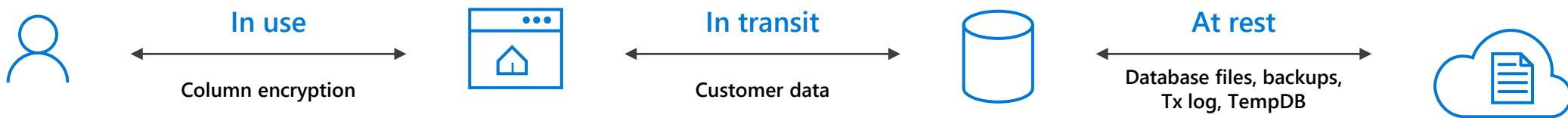
Masked data (admin1 login)

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

3

Types of data encryption

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



Transparent data encryption (TDE)

Overview

All customer data encrypted at rest

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

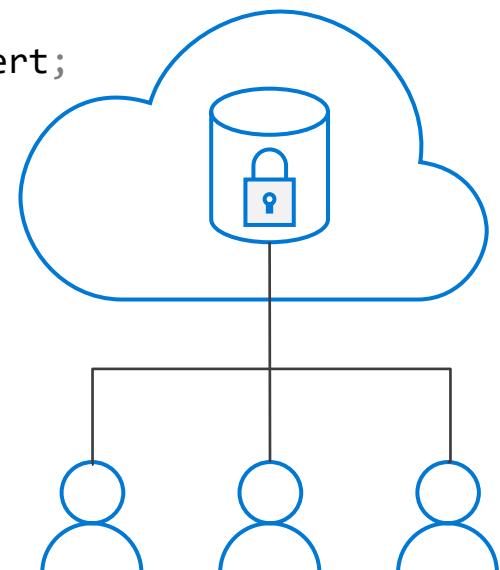
Service OR User managed keys.

Application changes kept to a minimum.

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

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

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



Transparent data encryption (TDE)

Key Vault

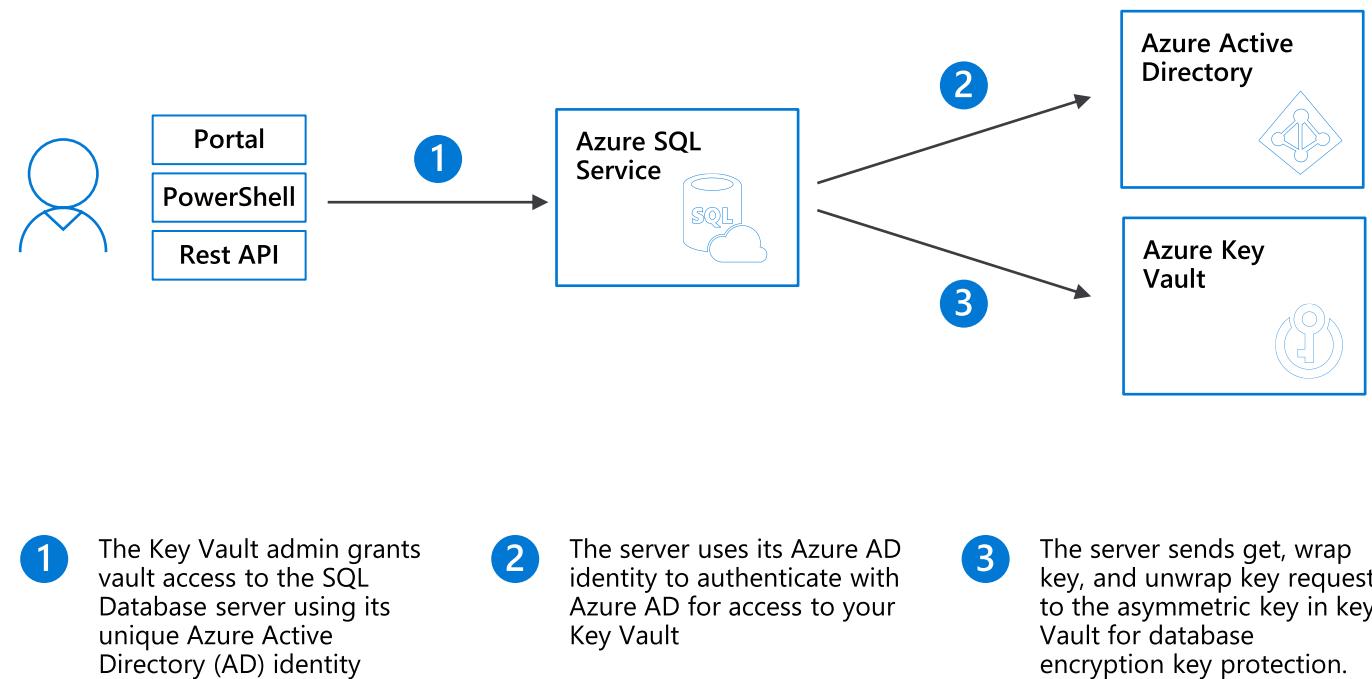
Benefits with User Managed Keys

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

Highly available and scalable cloud-based key store.

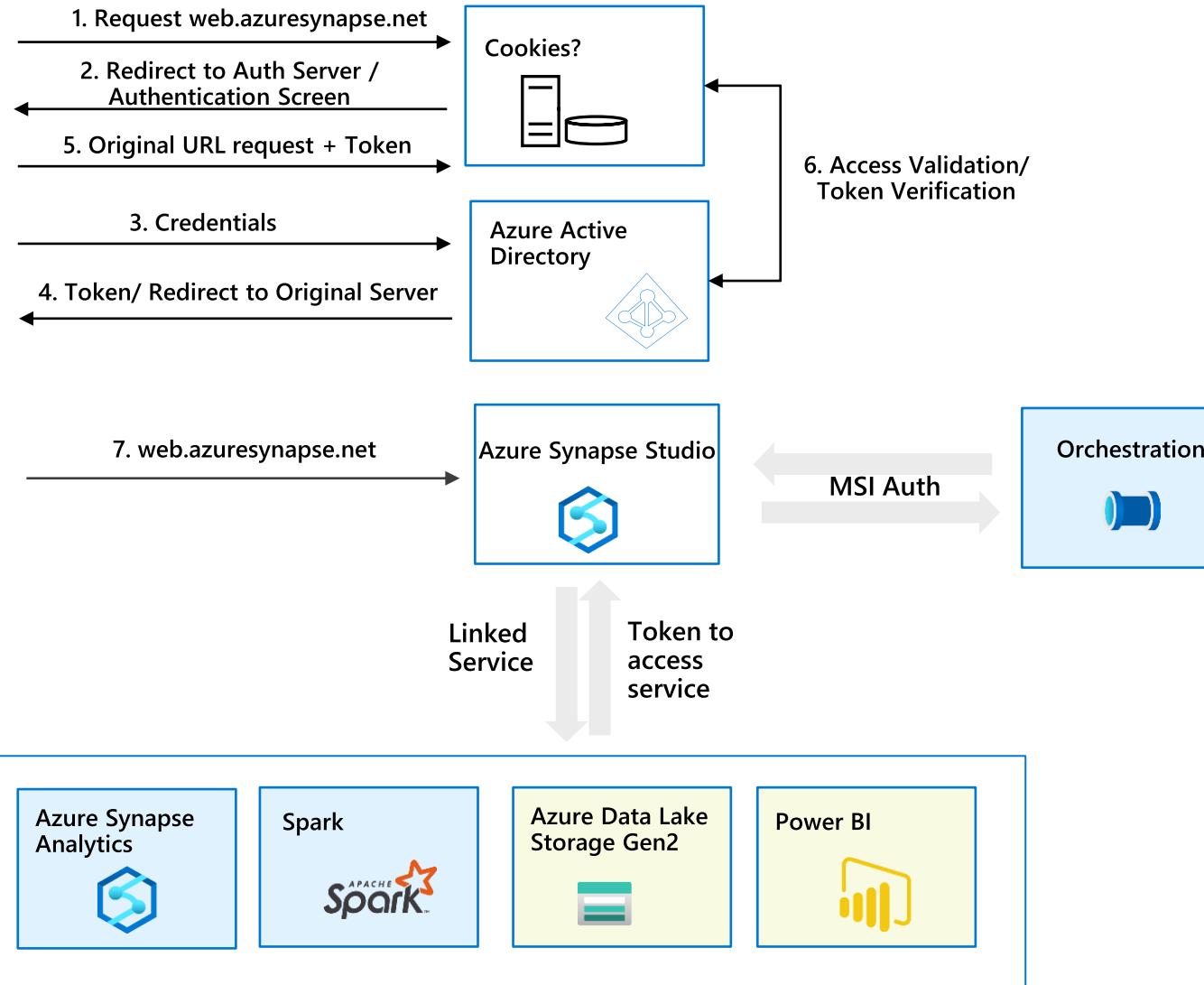
Central key management that allows separation of key management and data.

Configurable via Azure Portal, PowerShell, and REST API.



Single Sign-On

- Synapse Foundation Components
- Synapse Linked Services





End