Microsoft

Azure Synapse Analytics

James Serra

Data & AI Architect

Microsoft, NYC MTC

JamesSerra3@gmail.com

Blog: JamesSerra.com

About Me

- Microsoft, Big Data Evangelist
- In IT for 30 years, worked on many BI and DW projects
- Worked as desktop/web/database developer, DBA, BI and DW architect and developer, MDM architect, PDW/APS developer
- Been perm employee, contractor, consultant, business owner
- Presenter at PASS Business Analytics Conference, PASS Summit, Enterprise Data World conference
- Certifications: MCSE: Data Platform, Business Intelligence; MS: Architecting Microsoft Azure Solutions, Design and Implement Big Data Analytics Solutions, Design and Implement Cloud Data Platform Solutions
- Blog at JamesSerra.com
- Former SQL Server MVP
- Author of book “Reporting with Microsoft SQL Server 2012”

MVP Microsoft Most Valuable Professional

Agenda

- Introduction
- Studio
- Data Integration
- SQL Analytics
- Data Storage and Performance Optimizations
- SQL On-Demand
- Spark
- Security
- Connected Services

Microsoft

Azure Synapse Analytics is a limitless analytics service, that brings together enterprise data warehousing and Big Data analytics. It gives you the freedom to query data on your terms, using either serverless on-demand or provisioned resources, at scale. Azure Synapse brings these two worlds together with a unified experience to ingest, prepare, manage, and serve data for immediate business intelligence and machine learning needs.

Azure Synapse – SQL Analytics

focus areas

Best in class price per performance

Up to 94% less expensive than competitors

Industry-leading security

Defense-in-depth security and 99.9% financially backed availability SLA

Workload aware query execution

Manage heterogenous workloads through workload priorities and isolation

Data flexibility

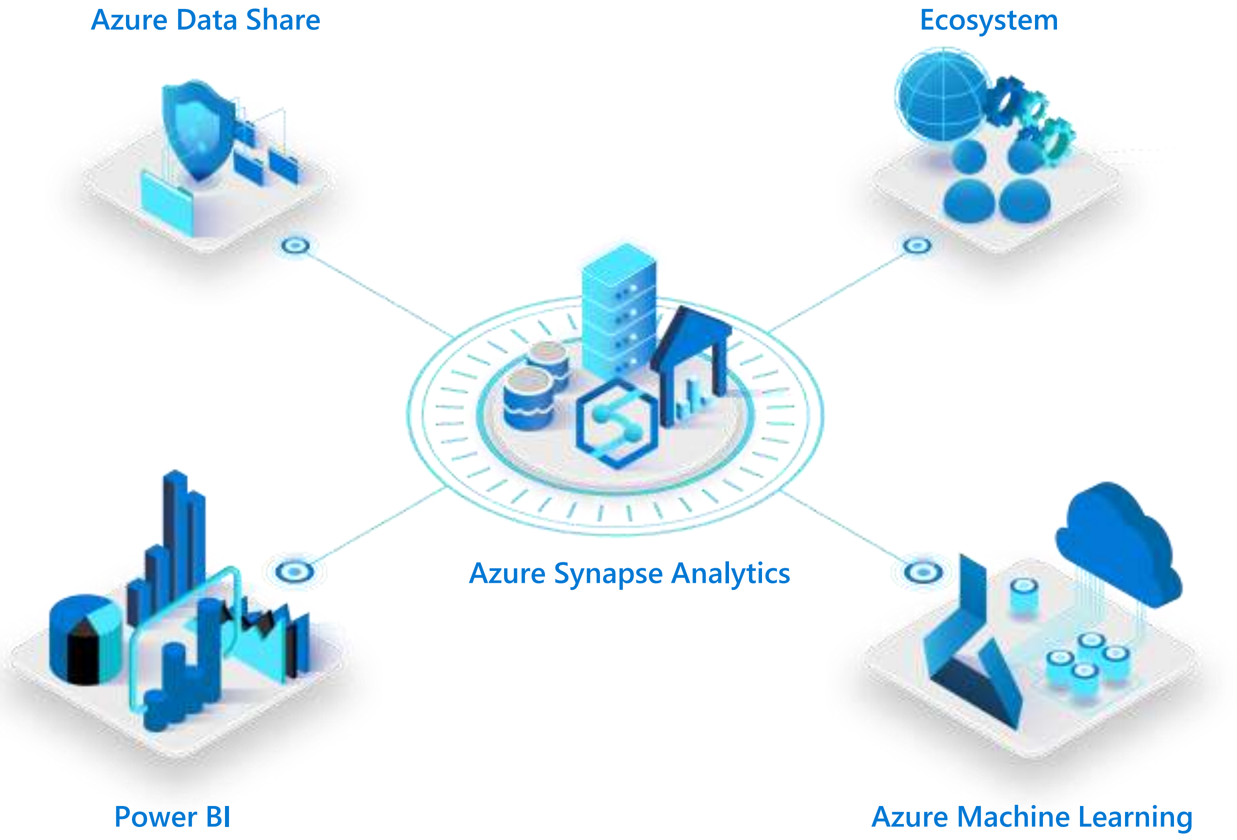
Ingest variety of data sources to derive the maximum benefit.

Query all data.

Developer productivity

Use preferred tooling for SQL data warehouse development

Leveraging ISV partners with Azure Synapse Analytics



Azure Data Share

Ecosystem

Azure Synapse Analytics

Power BI

Azure Machine Learning

Informatica talend ATTUNITY A Division of Qlik

panoply Azure Databricks + many more

Full backward compatibility with Azure SQL Data Warehouse for data integration and orchestration

Additional analytics capabilities in Azure Synapse unlocks new ISV scenarios

Azure Synapse + ISV can bring data continuity with Azure Machine Learning and Power BI

Reduce migration effort by reusing existing partner platforms

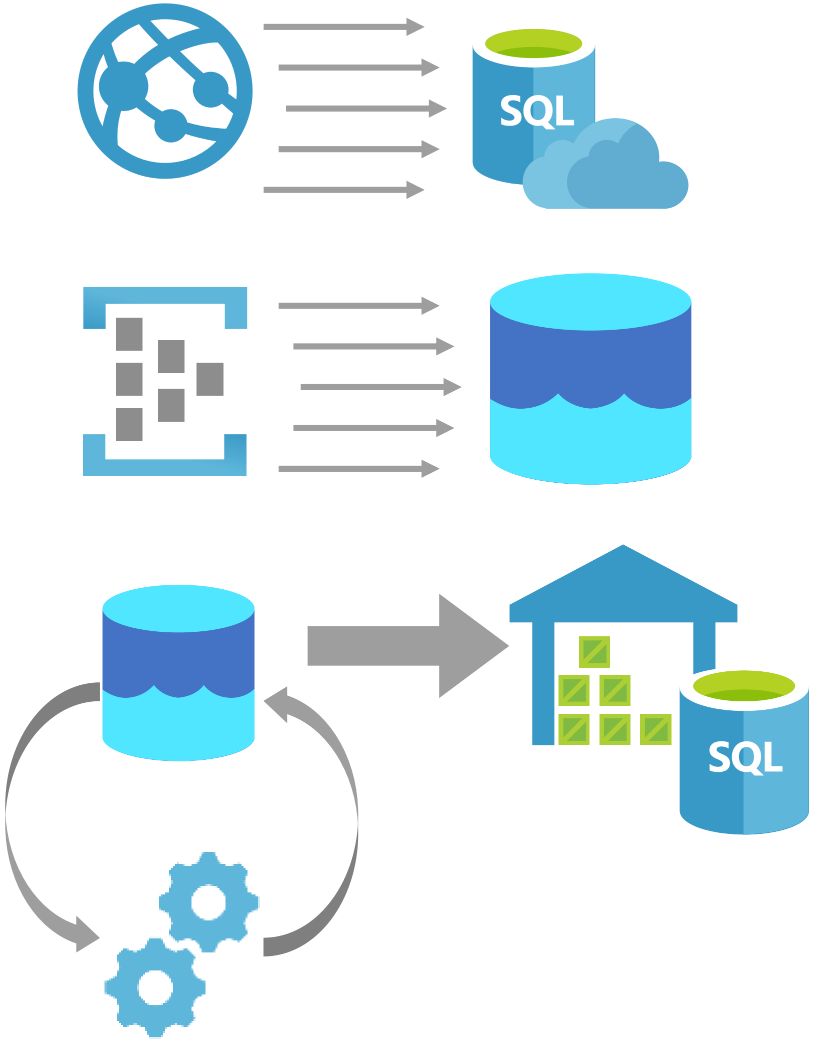
What workloads are NOT suitable?

Operational workloads (OLTP)

- High frequency reads and writes.
- Large numbers of singleton selects.
- High volumes of single row inserts.

Data Preparations

- Row by row processing needs.
- Incompatible formats (XML).



What Workloads are Suitable?

Analytics

Store large volumes of data.

Consolidate disparate data into a single location.

Shape, model, transform and aggregate data.

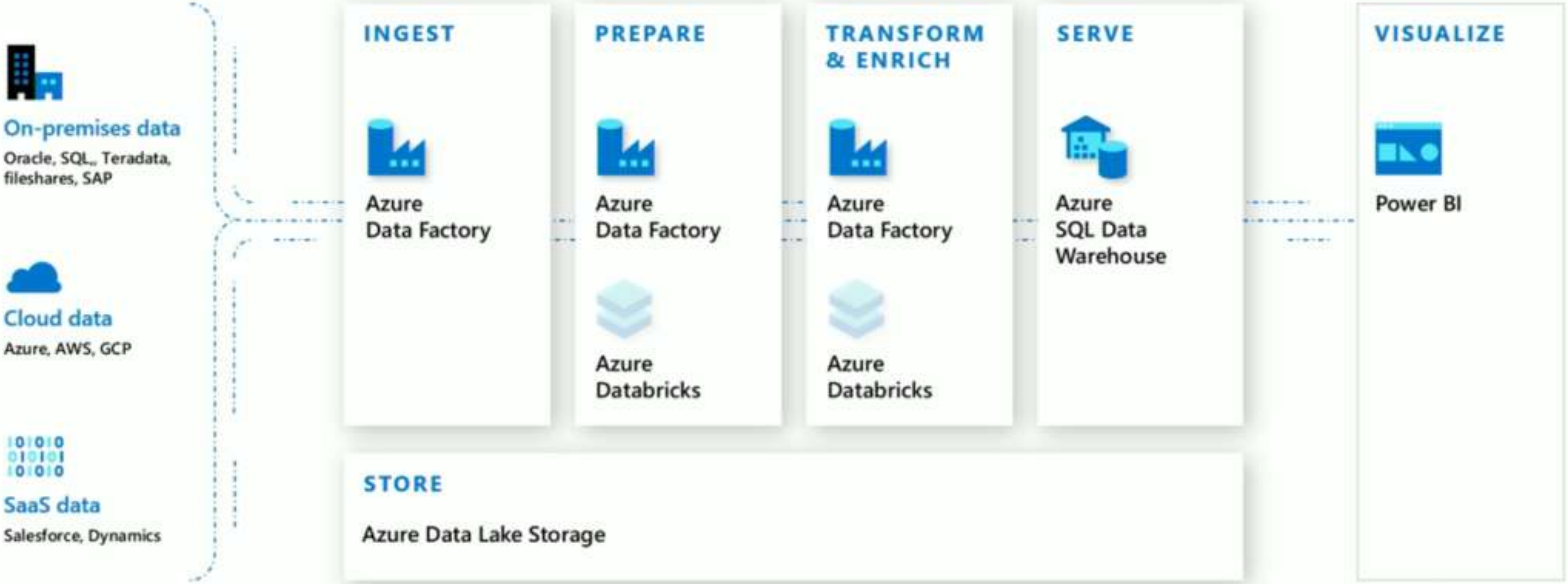
Batch/Micro-batch loads.

Perform query analysis across large datasets.

Ad-hoc reporting across large data volumes.

All using simple SQL constructs.

Modern Data Warehouse



On-premises data

Oracle, SQL,, Teradata, fileshares, SAP

Cloud data

Azure, AWS, GCP

SaaS data

Salesforce, Dynamics

INGEST

Azure Data Factory

PREPARE

Azure Data Factory

Azure Databricks

TRANSFORM & ENRICH

Azure Data Factory

Azure Databricks

SERVE

Azure SQL Data Warehouse

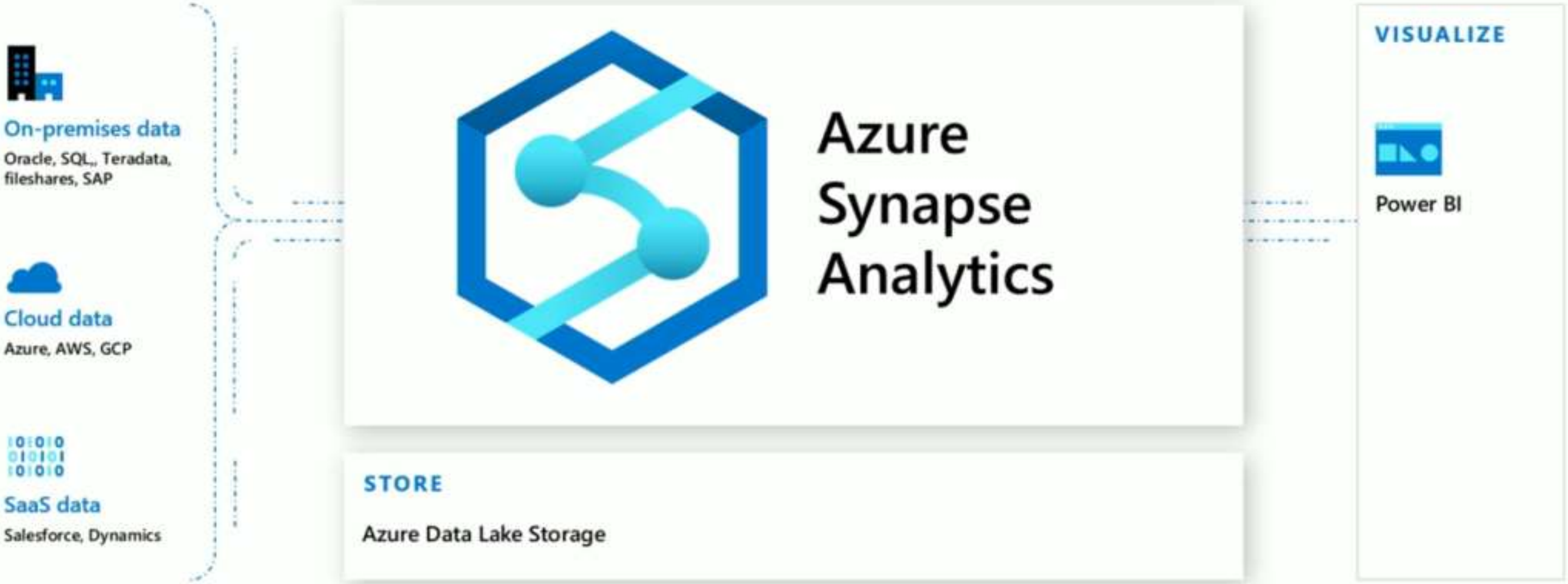
VISUALIZE

Power BI

STORE

Azure Data Lake Storage

Azure Synapse Analytics



On-premises data

Oracle, SQL,, Teradata, fileshares, SAP

Cloud data

Azure, AWS, GCP

SaaS data

Salesforce, Dynamics

Azure Synapse Analytics

STORE

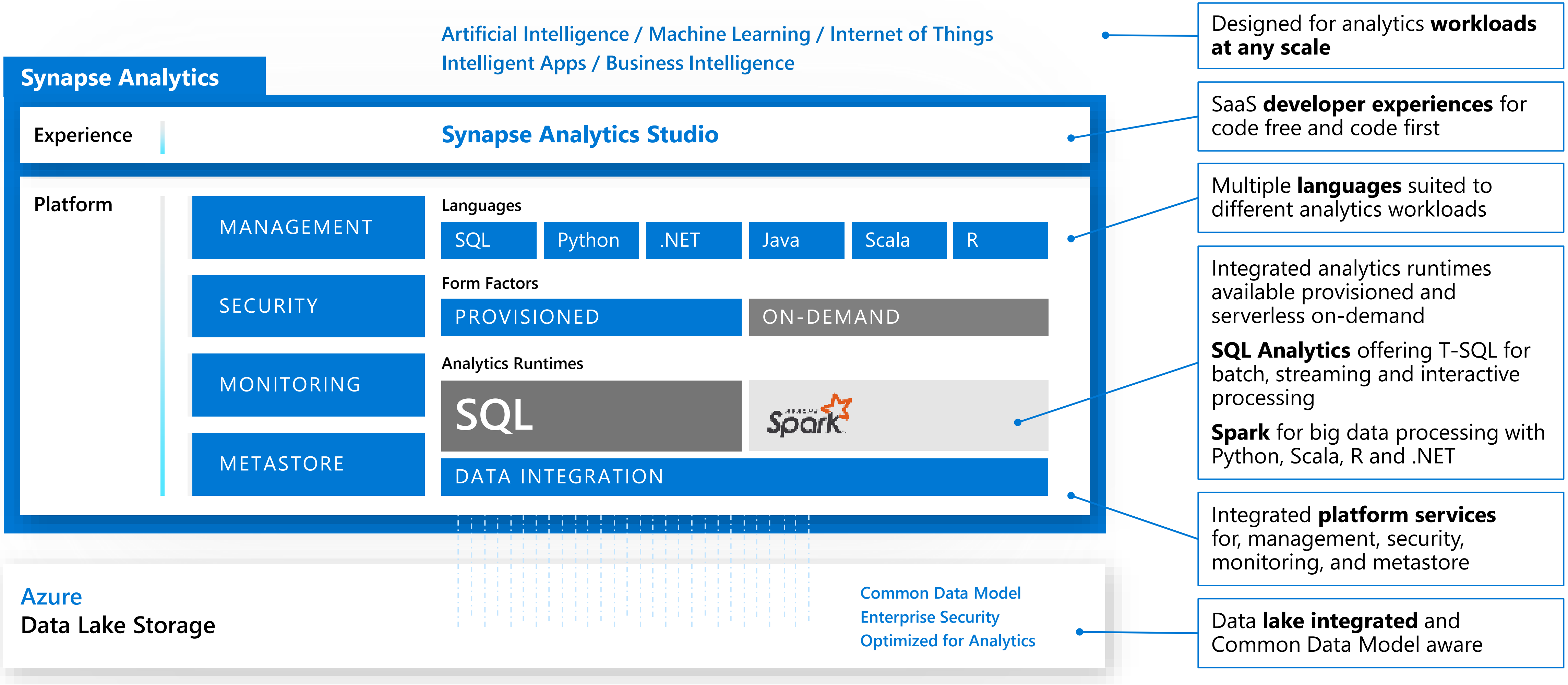
Azure Data Lake Storage

VISUALIZE

Power BI

Azure Synapse Analytics

Integrated data platform for BI, AI and continuous intelligence



Artificial Intelligence / Machine Learning / Internet of Things

Intelligent Apps / Business Intelligence

Synapse Analytics

Experience

Synapse Analytics Studio

Platform

MANAGEMENT

SECURITY

MONITORING

METASTORE

Languages

SQL

Python

.NET

Java

Scala

R

Form Factors

PROVISIONED

ON-DEMAND

Analytics Runtimes

SQL

Spark

DATA INTEGRATION

Azure

Data Lake Storage

Common Data Model

Enterprise Security

Optimized for Analytics

Designed for analytics **workloads at any scale**

SaaS **developer experiences** for code free and code first

Multiple **languages** suited to different analytics workloads

Integrated analytics runtimes available provisioned and serverless on-demand

SQL Analytics offering T-SQL for batch, streaming and interactive processing

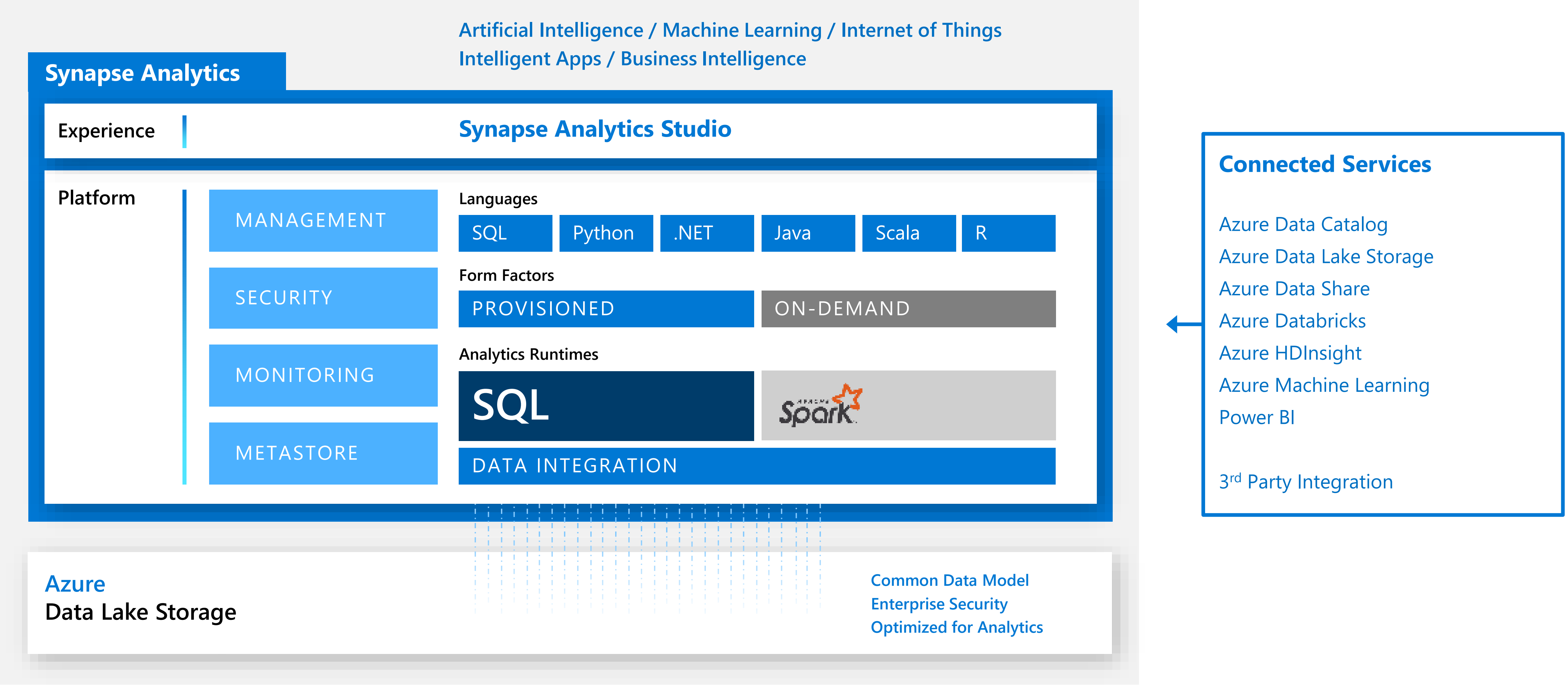
Spark for big data processing with Python, Scala, R and .NET

Integrated **platform services** for, management, security, monitoring, and metastore

Data **lake integrated** and Common Data Model aware

Azure Synapse Analytics

Integrated data platform for BI, AI and continuous intelligence



Artificial Intelligence / Machine Learning / Internet of Things
Intelligent Apps / Business Intelligence

Synapse Analytics

Experience — Synapse Analytics Studio

Platform

MANAGEMENT
SECURITY
MONITORING
METASTORE

Languages

SQL | Python | .NET | Java | Scala | R

Form Factors

PROVISIONED | ON-DEMAND

Analytics Runtimes

SQL | Spark

DATA INTEGRATION

Azure
Data Lake Storage

Common Data Model
Enterprise Security
Optimized for Analytics

Connected Services

Azure Data Catalog
Azure Data Lake Storage
Azure Data Share
Azure Databricks
Azure HDInsight
Azure Machine Learning
Power BI

3rd Party Integration

Provisioning Synapse workspace

Providing Synapse is easy

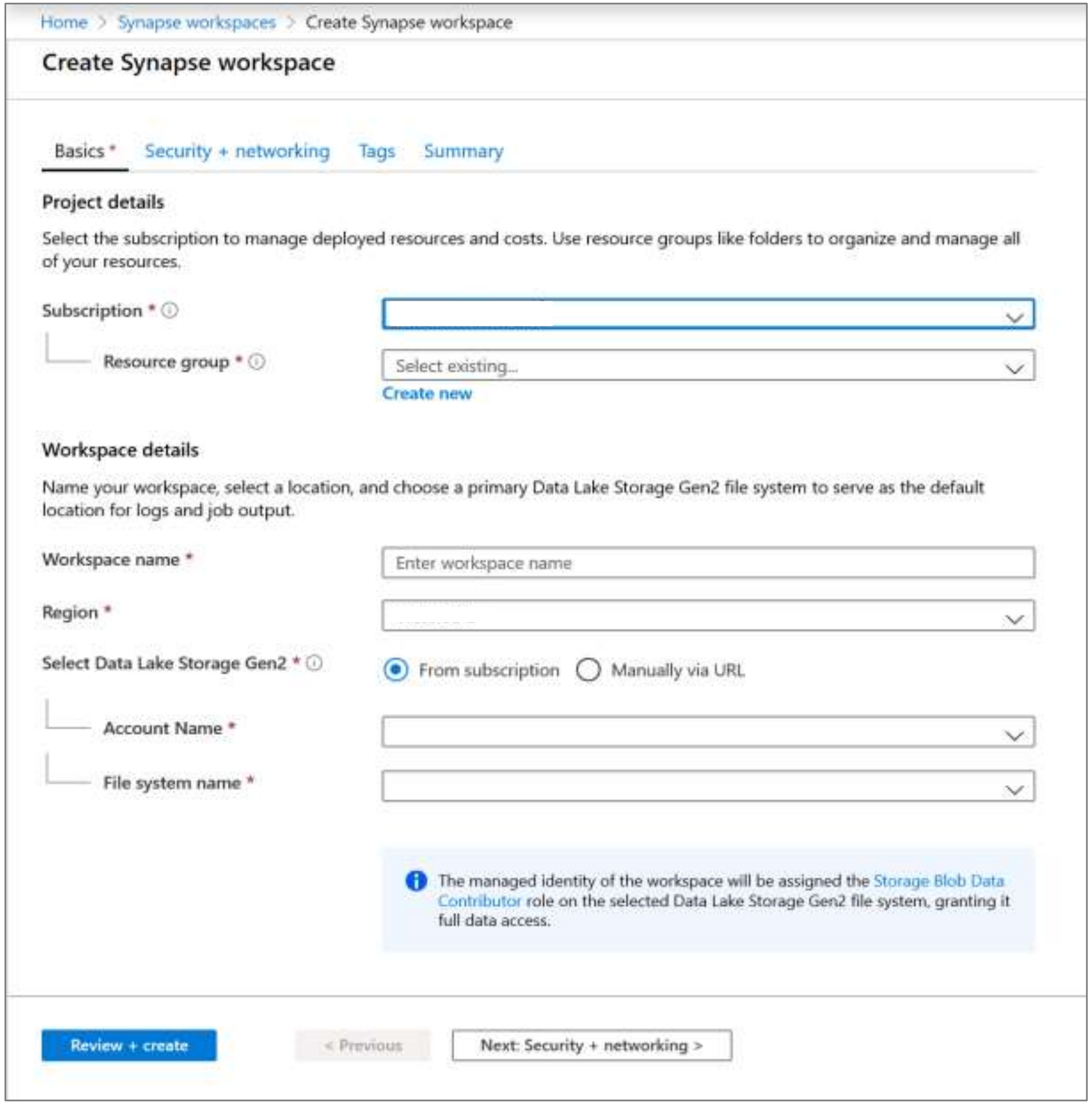
Subscription

Resource Group

Workspace Name

Region

Data Lake Storage Account



Home > Synapse workspaces > Create Synapse workspace

Create Synapse workspace

Basics * Security + networking Tags Summary

Project details

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all of your resources.

Subscription * ⓘ

Resource group * ⓘ Select existing...

Create new

Workspace details

Name your workspace, select a location, and choose a primary Data Lake Storage Gen2 file system to serve as the default location for logs and job output.

Workspace name * Enter workspace name

Region *

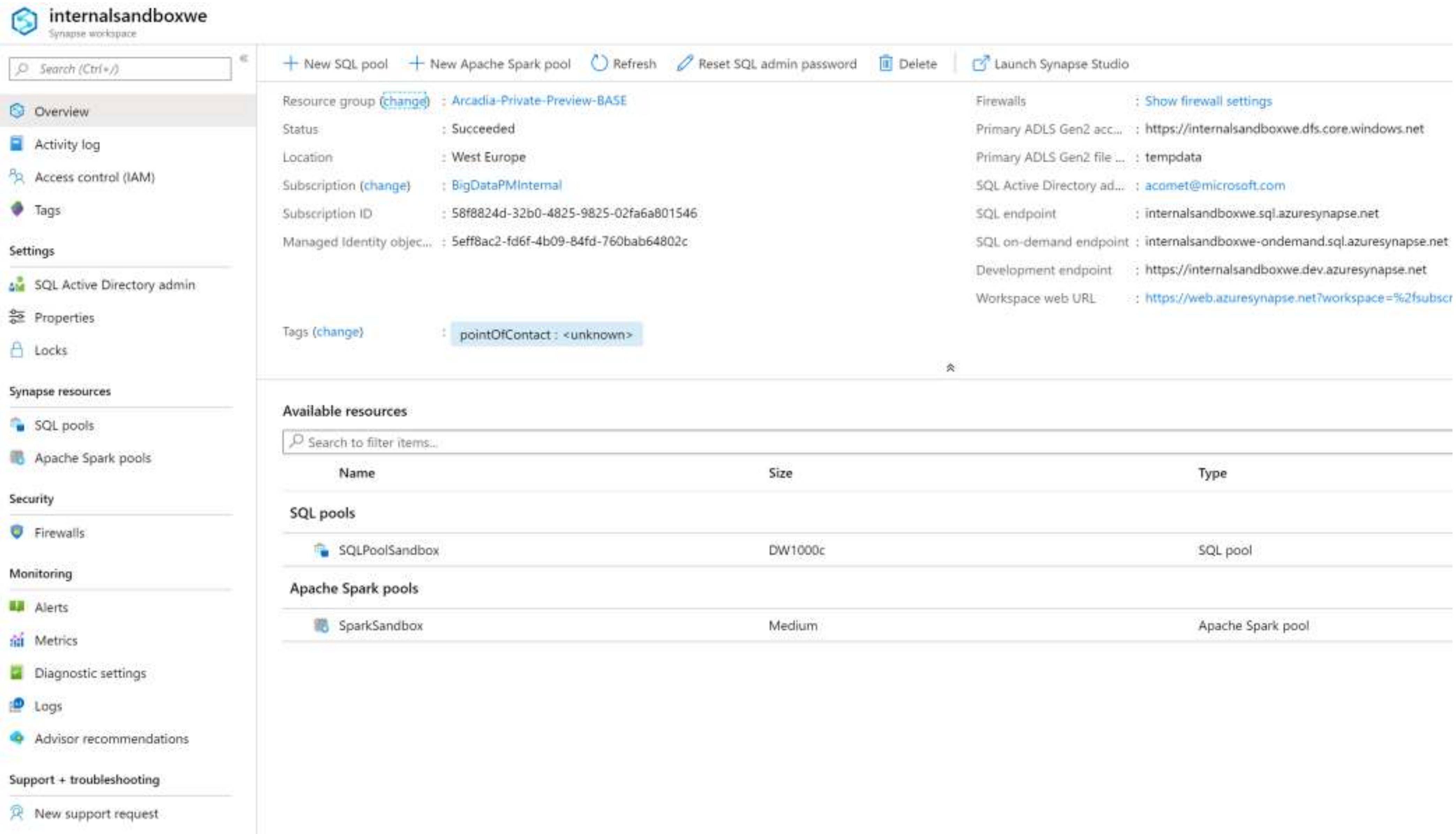
Select Data Lake Storage Gen2 * ⓘ ● From subscription ○ Manually via URL

Account Name *

File system name *

ⓘ The managed identity of the workspace will be assigned the Storage Blob Data Contributor role on the selected Data Lake Storage Gen2 file system, granting it full data access.

Synapse workspace



Resource group (change) : Arcadia-Private-Preview-BASE
Status : Succeeded
Location : West Europe
Subscription (change) : BigDataPMInternal
Subscription ID : 58f8824d-32b0-4825-9825-02fa6a801546
Managed Identity objec... : 5eff8ac2-fd6f-4b09-84fd-760bab64802c

Tags (change) : pointOfContact : <unknown>

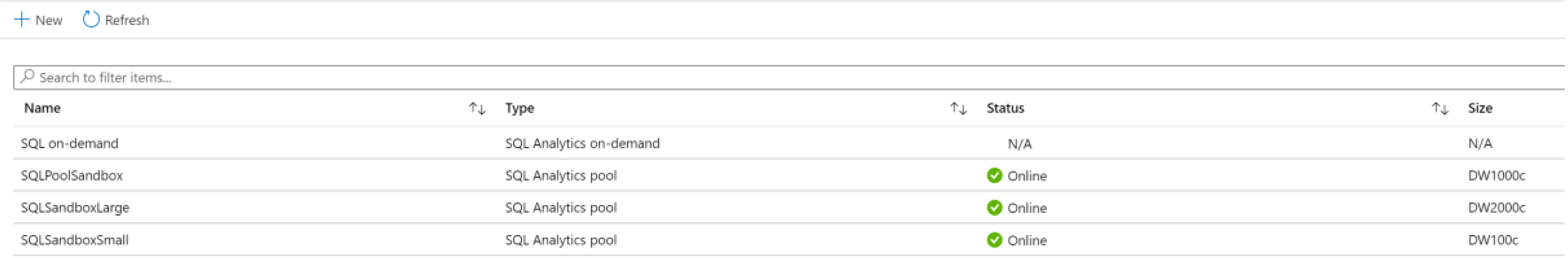
Firewalls : Show firewall settings
Primary ADLS Gen2 acc... : https://internalsandboxwe.dfs.core.windows.net
Primary ADLS Gen2 file ... : tempdata
SQL Active Directory ad... : acomet@microsoft.com
SQL endpoint : internalsandboxwe.sql.azuresynapse.net
SQL on-demand endpoint : internalsandboxwe-ondemand.sql.azuresynapse.net
Development endpoint : https://internalsandboxwe.dev.azuresynapse.net
Workspace web URL : https://web.azuresynapse.net?workspace=%2fsubscr

Available resources

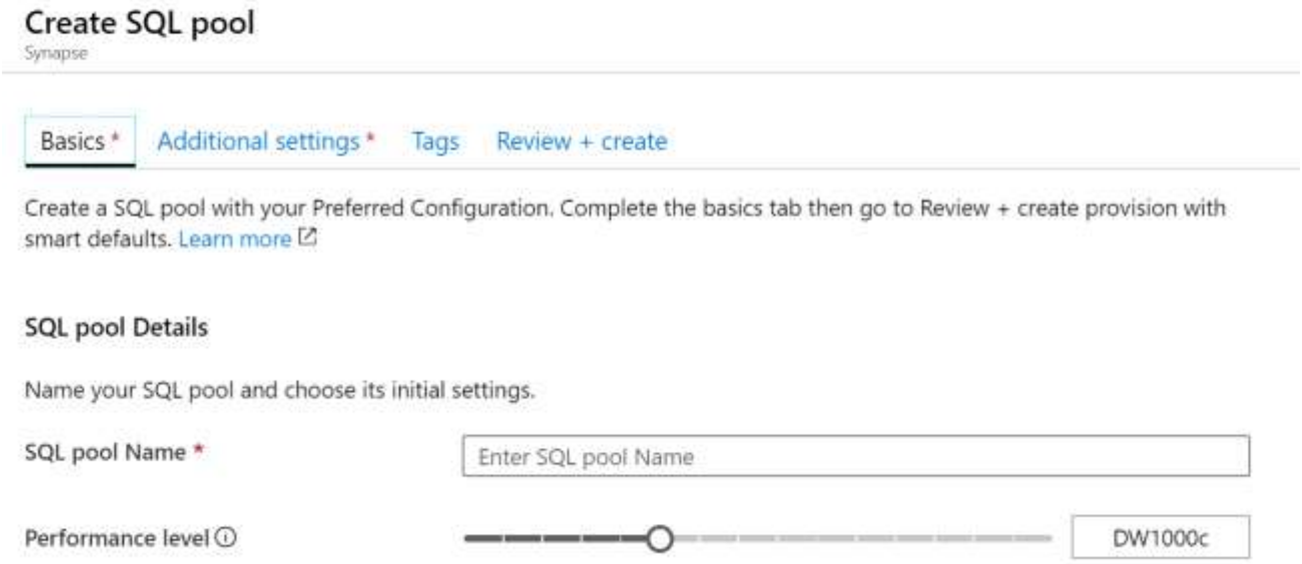
Search to filter items...

Name	Size	Type
SQL pools		
SQLPoolSandbox	DW1000c	SQL pool
Apache Spark pools		
SparkSandbox	Medium	Apache Spark pool

SQL pools



Name	Type	Status	Size
SQL on-demand	SQL Analytics on-demand	N/A	N/A
SQLPoolSandbox	SQL Analytics pool	✓ Online	DW1000c
SQLSandboxLarge	SQL Analytics pool	✓ Online	DW2000c
SQLSandboxSmall	SQL Analytics pool	✓ Online	DW100c



Create SQL pool

Synapse

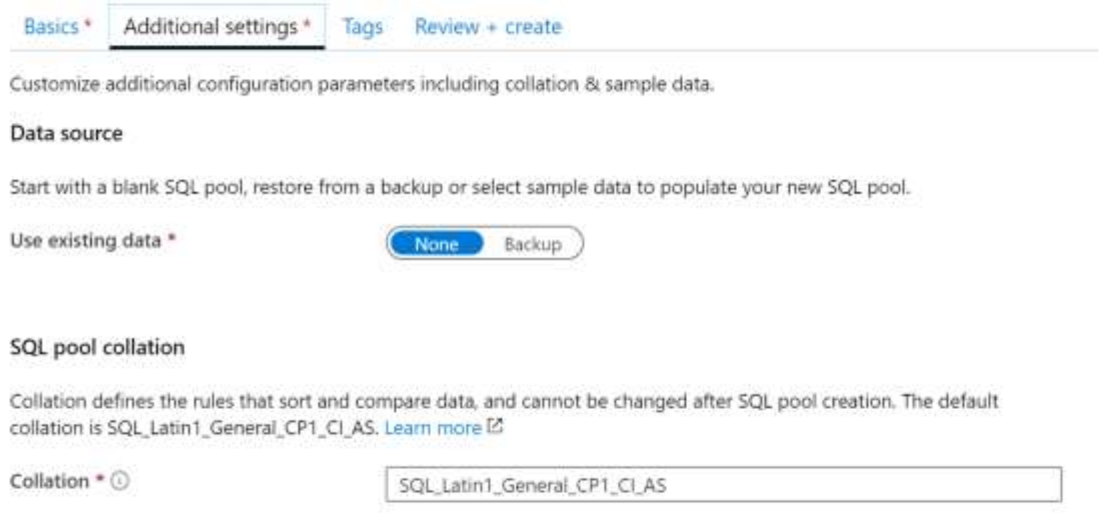
Create a SQL pool with your Preferred Configuration. Complete the basics tab then go to Review + create provision with smart defaults. Learn more

SQL pool Details

Name your SQL pool and choose its initial settings.

SQL pool Name * Enter SQL pool Name

Performance level ⓘ DW1000c



Customize additional configuration parameters including collation & sample data.

Data source

Start with a blank SQL pool, restore from a backup or select sample data to populate your new SQL pool.

Use existing data * [✓ None] Backup

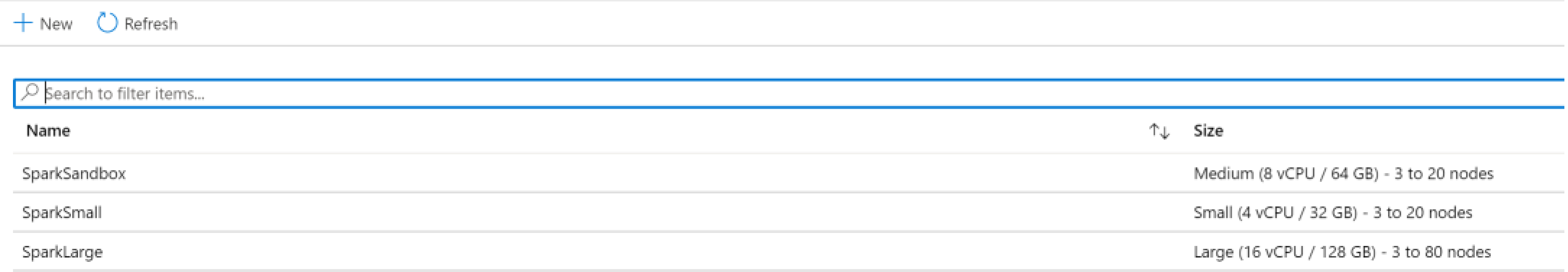
SQL pool collation

Collation defines the rules that sort and compare data, and cannot be changed after SQL pool creation. The default collation is SQL_Latin1_General_CP1_CI_AS. Learn more

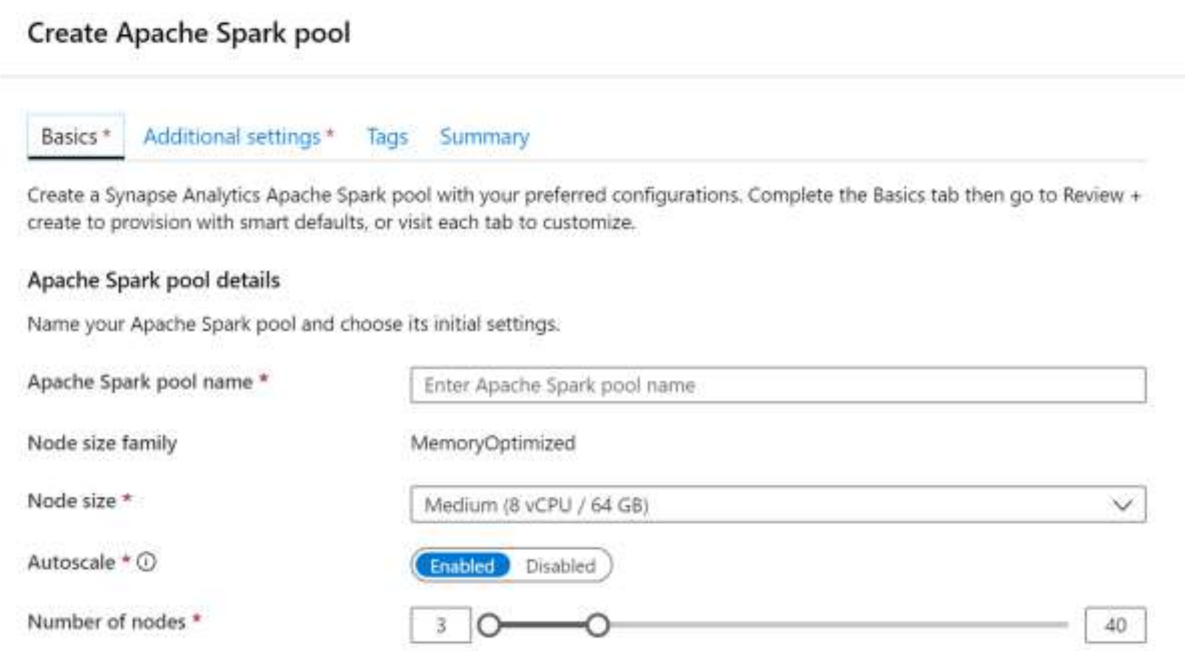
Collation * ⓘ SQL_Latin1_General_CP1_CI_AS

SQL Analytics pool = SQL Data Warehouse

Apache Spark pools



Name	Size
SparkSandbox	Medium (8 vCPU / 64 GB) - 3 to 20 nodes
SparkSmall	Small (4 vCPU / 32 GB) - 3 to 20 nodes
SparkLarge	Large (16 vCPU / 128 GB) - 3 to 80 nodes



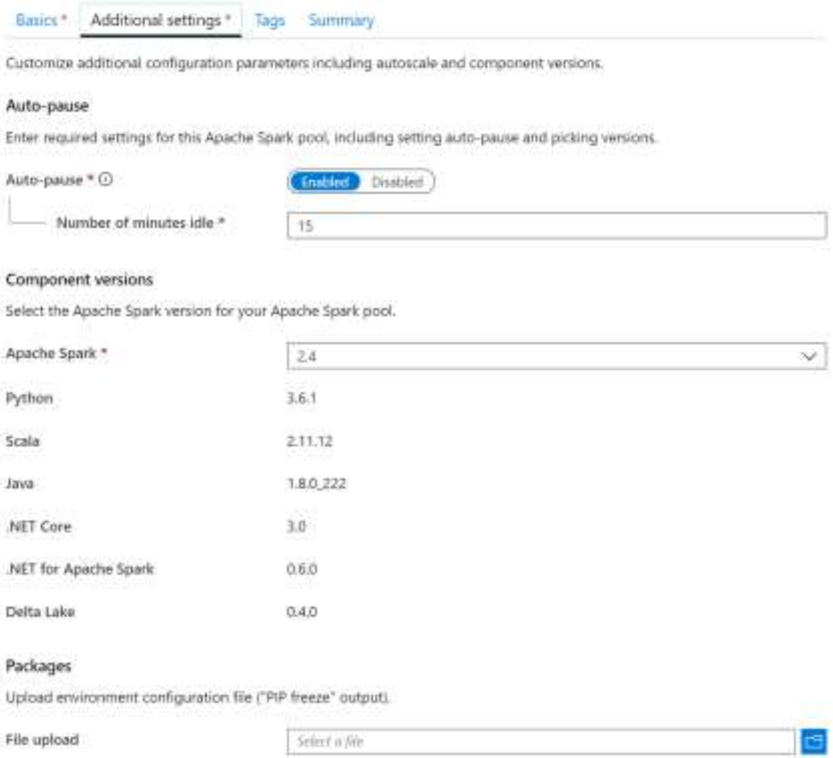
Create Apache Spark pool

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



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	Select a file

Note: There are no on-demand pools for Spark

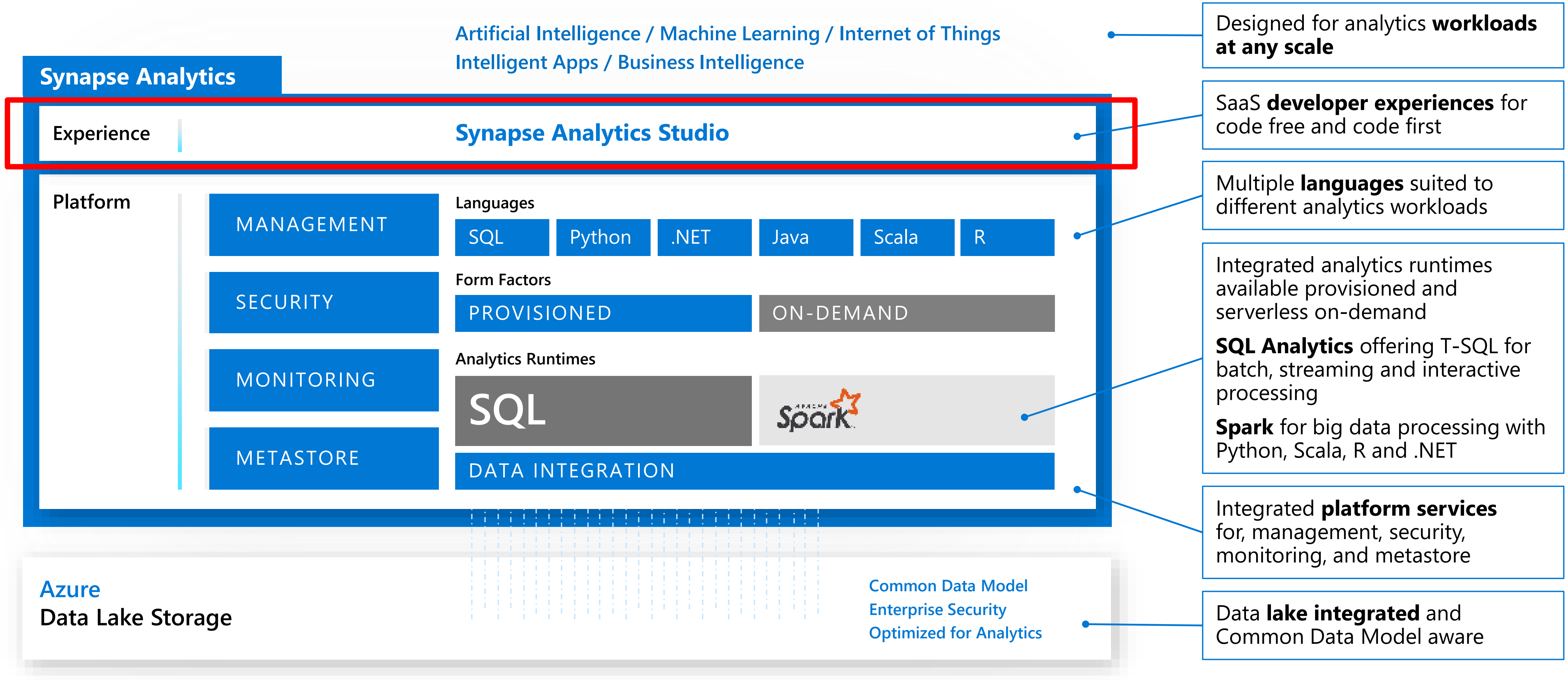
Microsoft

Azure Synapse Analytics

Studio

Azure Synapse Analytics

Integrated data platform for BI, AI and continuous intelligence



Artificial Intelligence / Machine Learning / Internet of Things
Intelligent Apps / Business Intelligence

Synapse Analytics

Experience | Synapse Analytics Studio

Platform

MANAGEMENT
SECURITY
MONITORING
METASTORE

Languages: SQL | Python | .NET | Java | Scala | R

Form Factors: PROVISIONED | ON-DEMAND

Analytics Runtimes: SQL | Spark

DATA INTEGRATION

Azure
Data Lake Storage

Common Data Model
Enterprise Security
Optimized for Analytics

Designed for analytics **workloads at any scale**

SaaS **developer experiences** for code free and code first

Multiple **languages** suited to different analytics workloads

Integrated analytics runtimes available provisioned and serverless on-demand

SQL Analytics offering T-SQL for batch, streaming and interactive processing

Spark for big data processing with Python, Scala, R and .NET

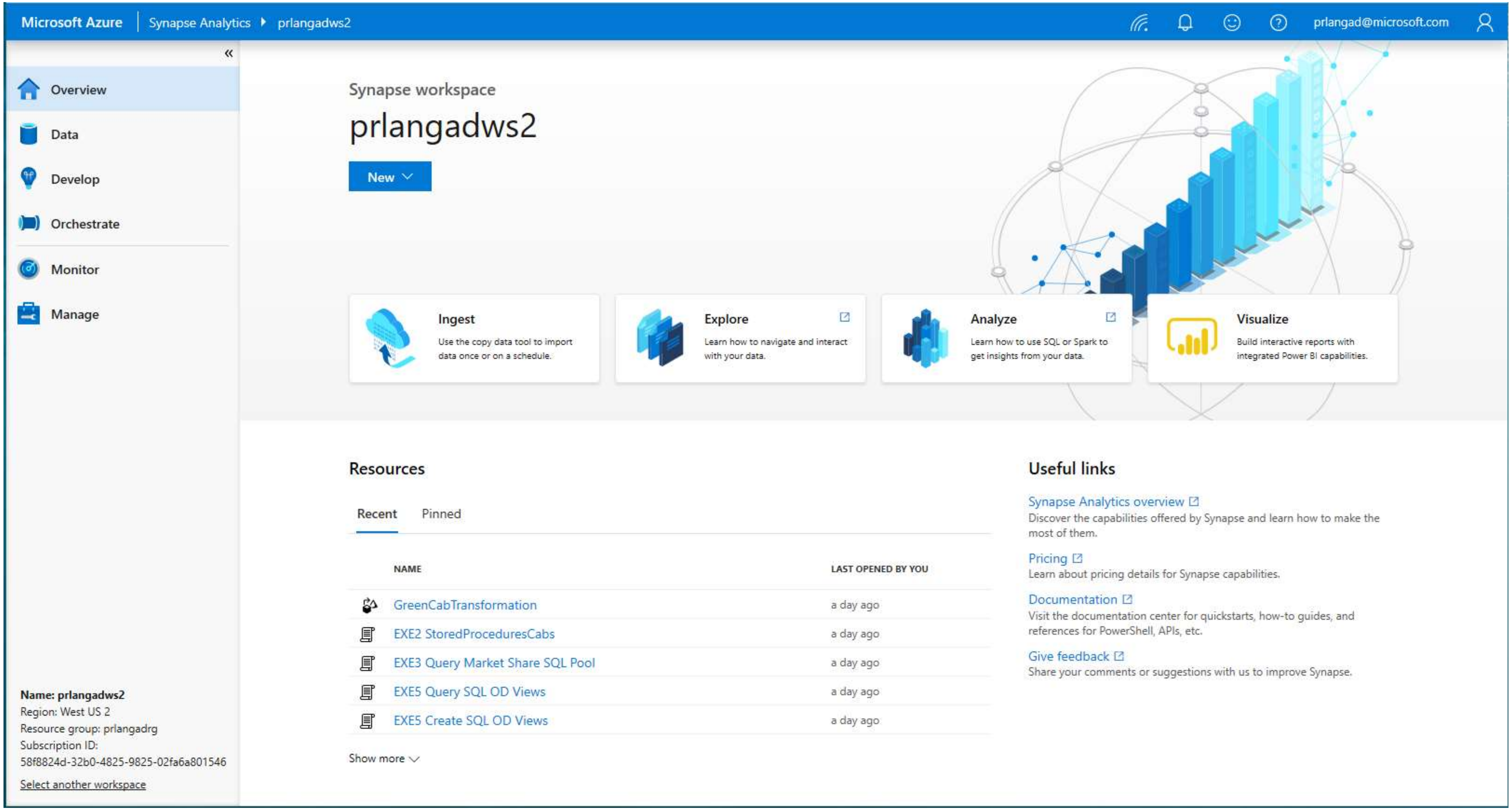
Integrated **platform services** for, management, security, monitoring, and metastore

Data **lake integrated** and Common Data Model aware

Studio

https://web.azuresynapse.net

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



Synapse workspace

prlangadws2

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
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

Useful links

Synapse Analytics overview
Discover the capabilities offered by Synapse and learn how to make the most of them.

Pricing
Learn about pricing details for Synapse capabilities.

Documentation
Visit the documentation center for quickstarts, how-to guides, and references for PowerShell, APIs, etc.

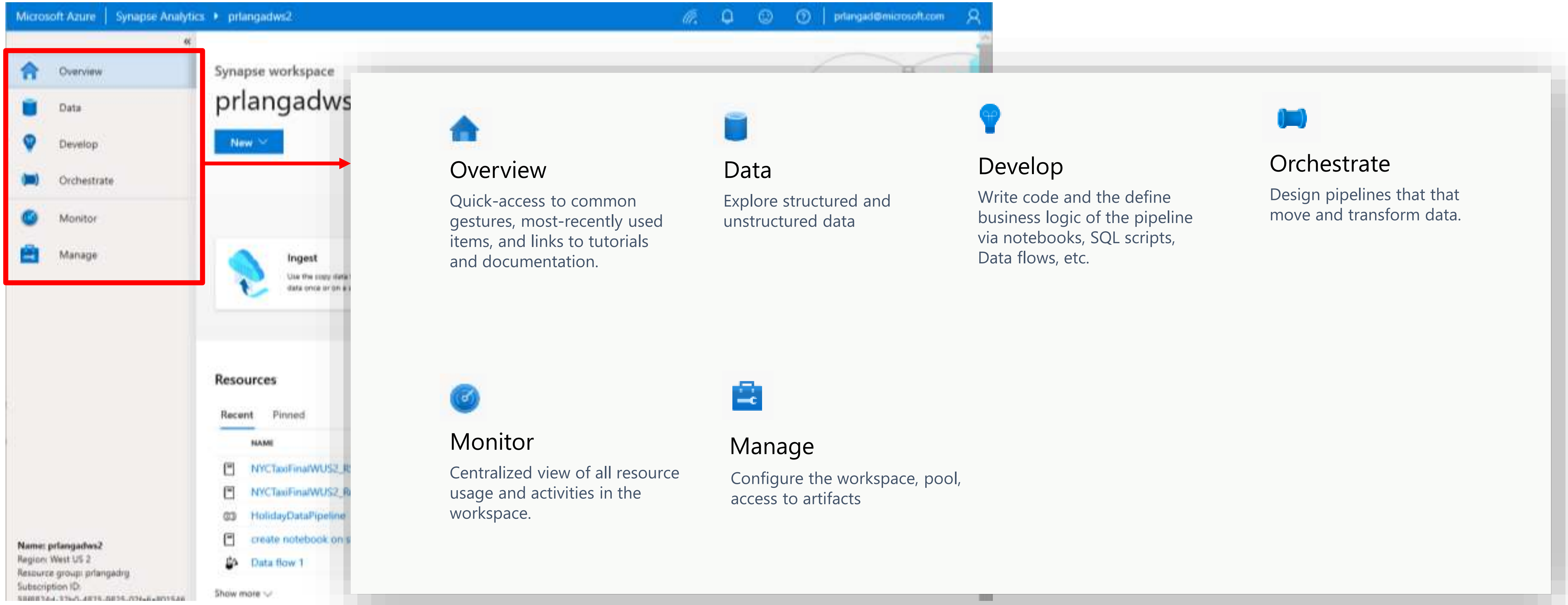
Give feedback
Share your comments or suggestions with us to improve Synapse.

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

Synapse Studio

Synapse Studio divided into **Activity hubs**.

These organize the tasks needed for building analytics solution.



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 the define business logic of the pipeline via notebooks, SQL scripts, Data flows, etc.

Orchestrate

Design pipelines that that move and transform data.

Monitor

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

Manage

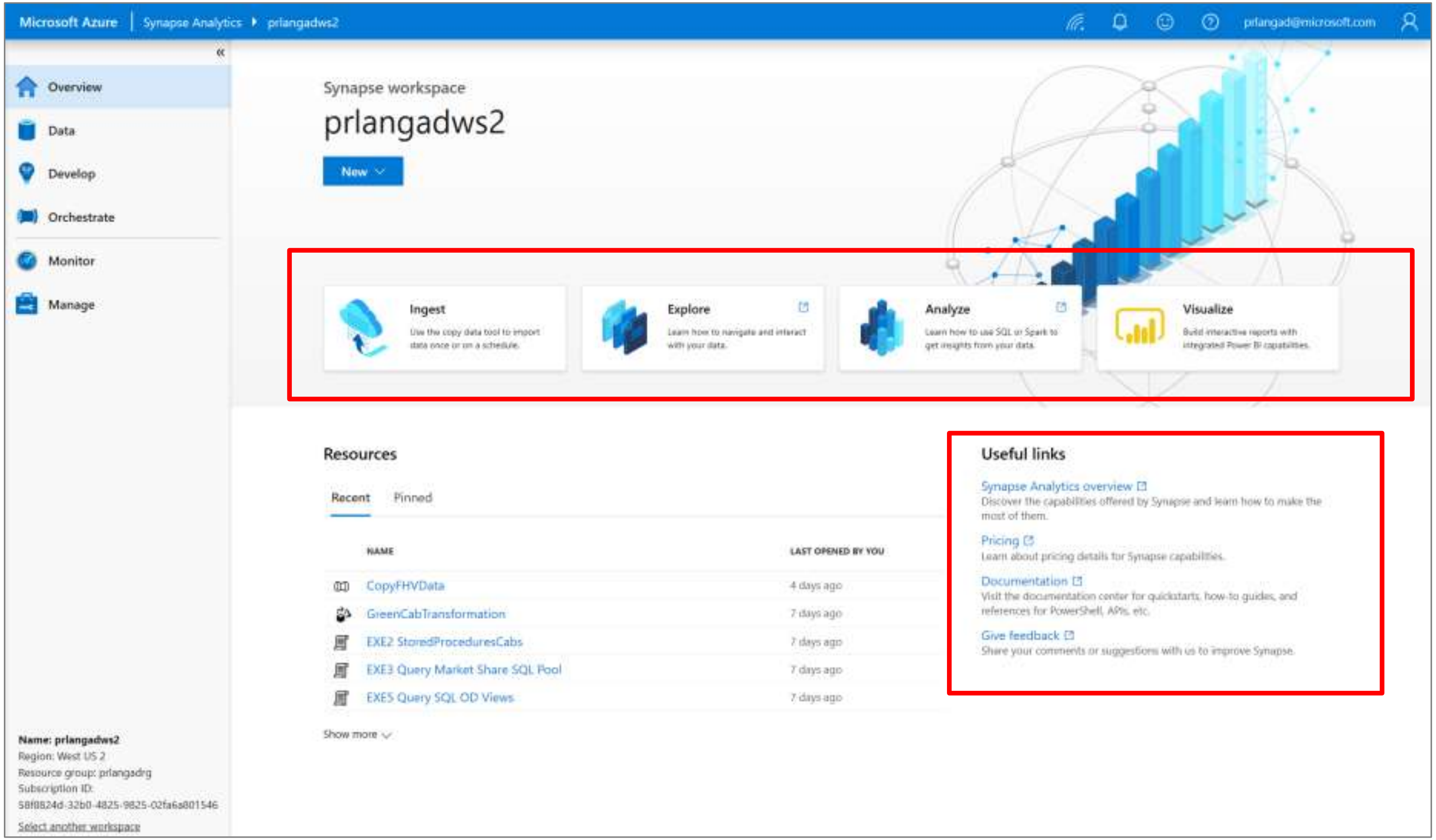
Configure the workspace, pool, access to artifacts

Synapse Studio

Overview hub

Overview Hub

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



prlangadws2

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

Explore
Learn how to navigate and interact with your data.

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

Visualize
Build interactive reports with integrated Power BI capabilities.

Resources

Recent | Pinned

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

Show more

Useful links

Synapse Analytics overview
Discover the capabilities offered by Synapse and learn how to make the most of them.

Pricing
Learn about pricing details for Synapse capabilities.

Documentation
Visit the documentation center for quickstarts, how-to guides, and references for PowerShell, APIs, etc.

Give feedback
Share your comments or suggestions with us to improve Synapse.

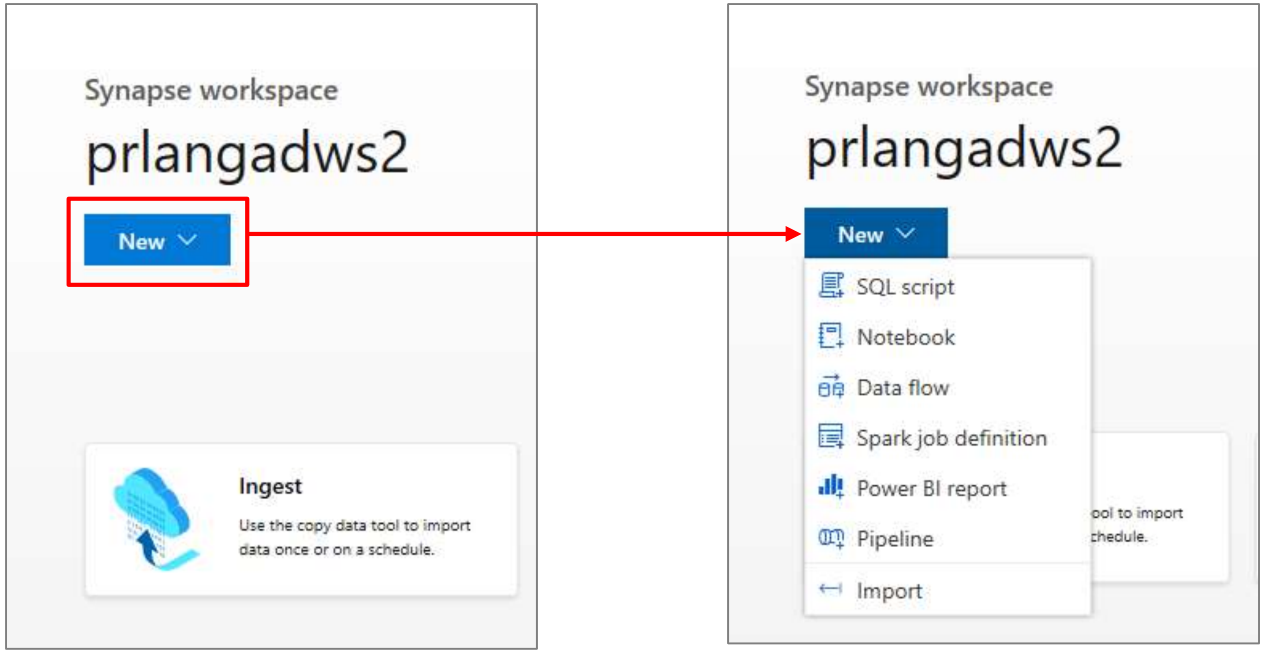
Name: prlangadws2
Region: West US 2
Resource group: prlangadrg
Subscription ID:
58f8824d-32b0-4825-9825-02fa6a801546
Select another workspace

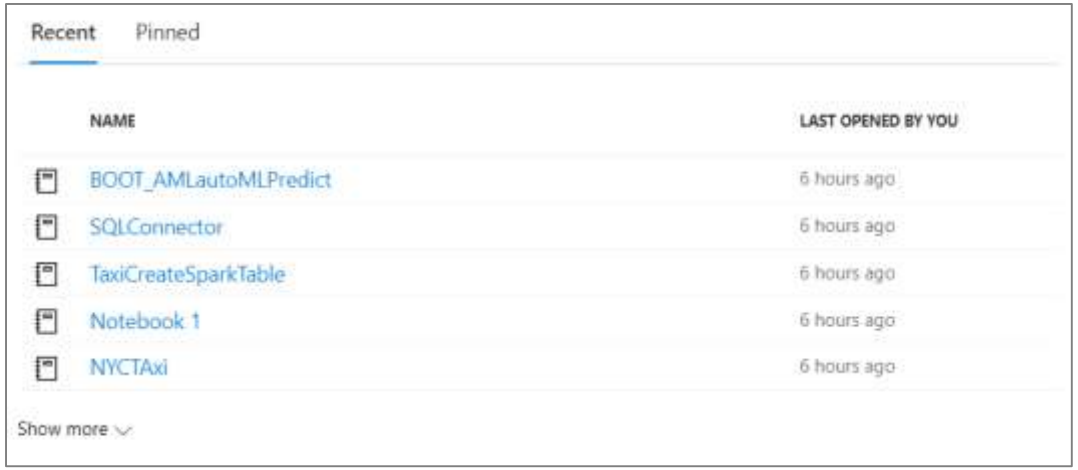
Overview Hub

Overview

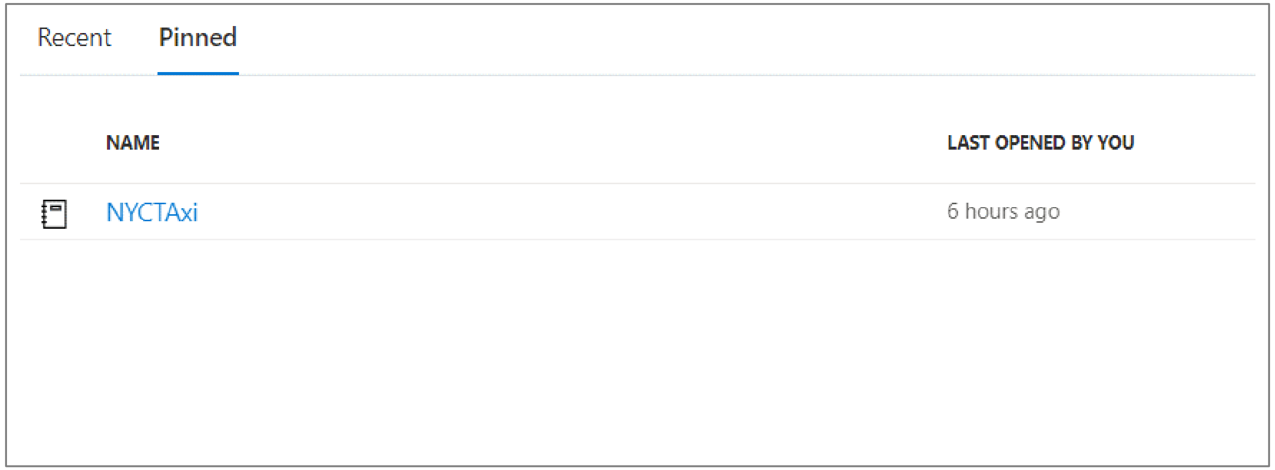
New dropdown – offers quickly start work item

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





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



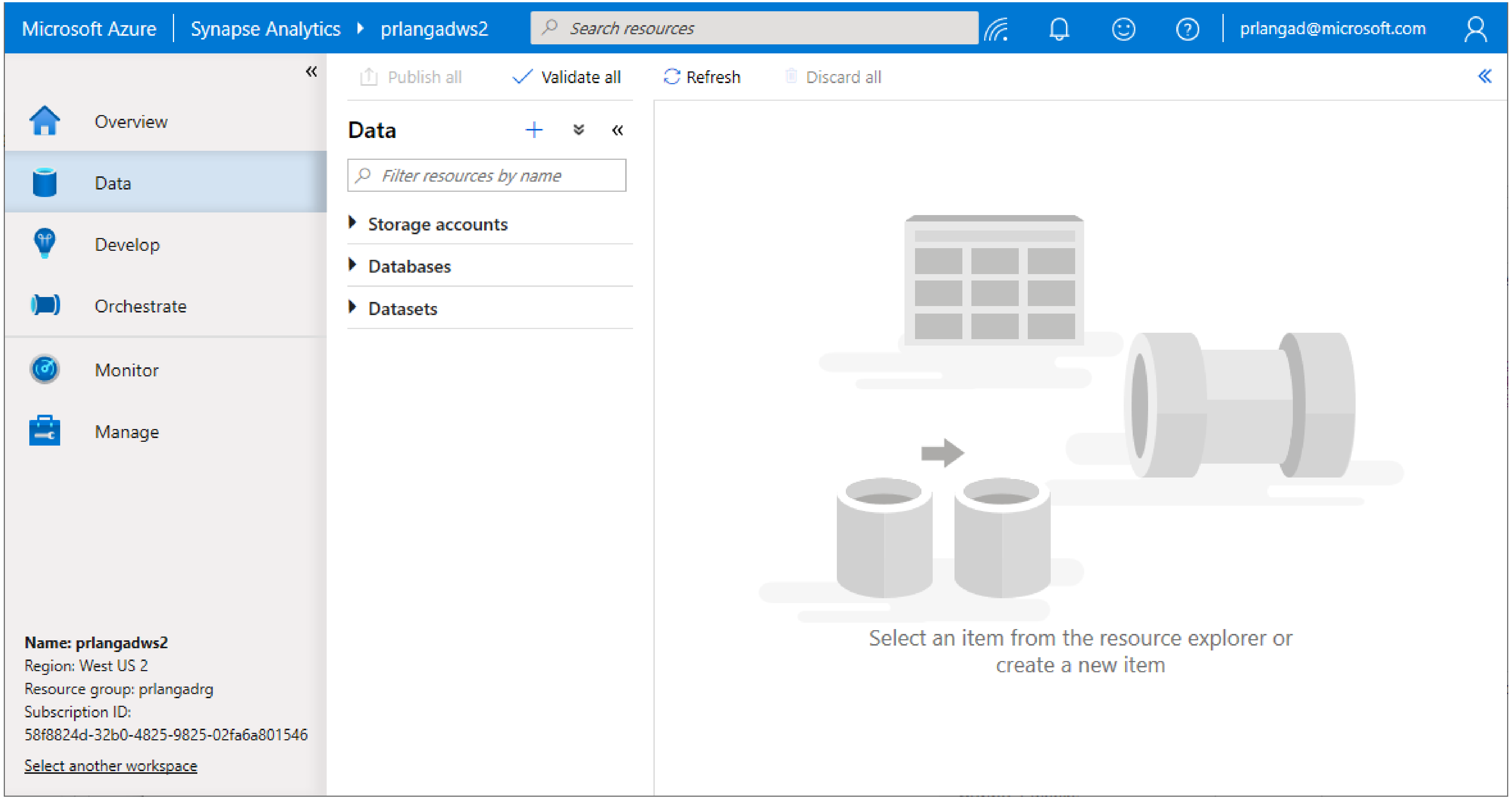
NAME	LAST OPENED BY YOU
NYCTAxi	6 hours ago

Synapse Studio

Data hub

Data Hub

Explore data inside the workspace and in linked storage accounts

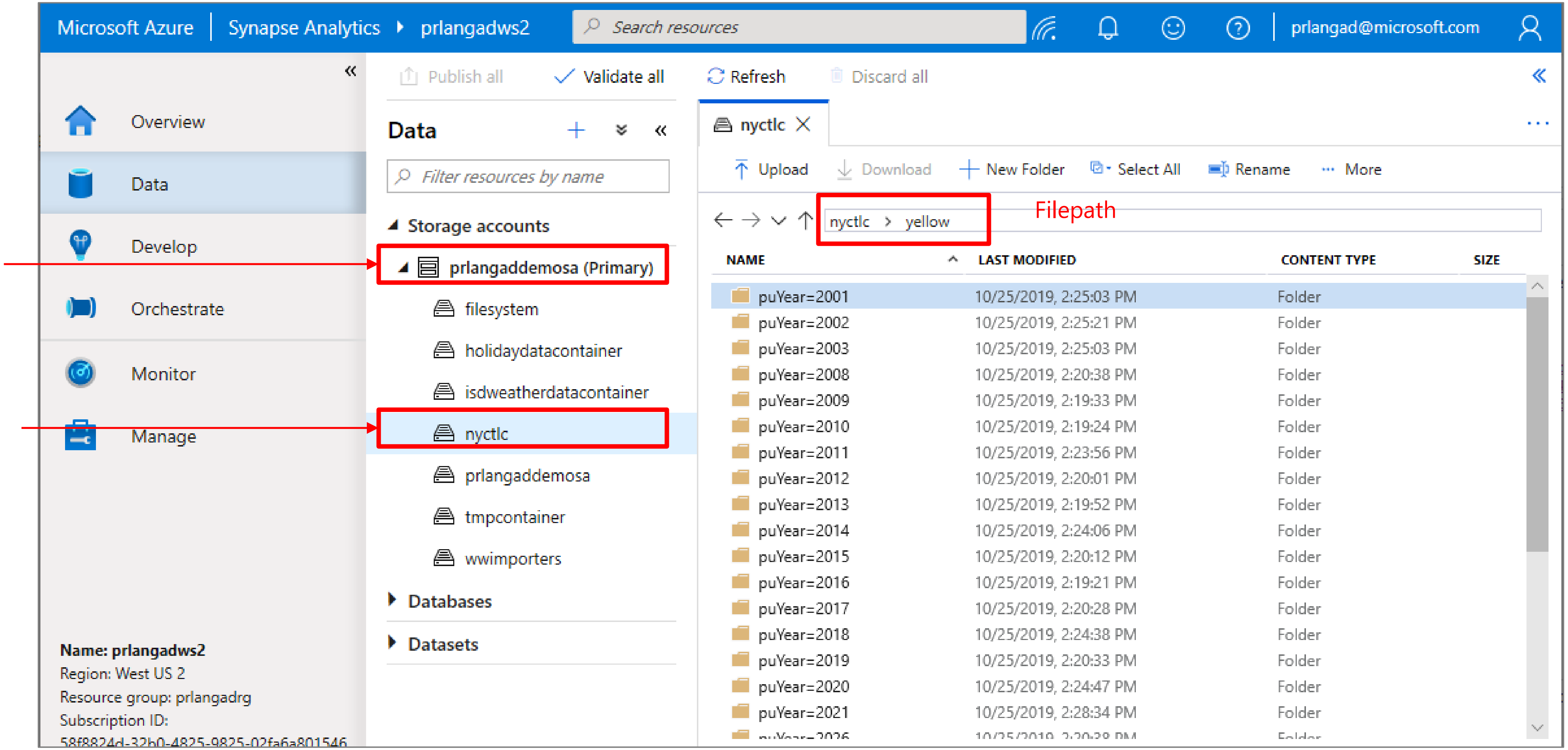


Name: prlangadws2
Region: West US 2
Resource group: prlangadrg
Subscription ID:
58f8824d-32b0-4825-9825-02fa6a801546
Select another workspace

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

Data Hub – Storage accounts

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



ADLS Gen2 Account

Container (filesystem)

- Storage accounts
  - prlangaddemosa (Primary)
    - filesystem
    - holidaydatacontainer
    - isdweatherdatacontainer
    - nyctlc
    - prlangaddemosa
    - tmpcontainer
    - wwimporters
- Databases
- Datasets

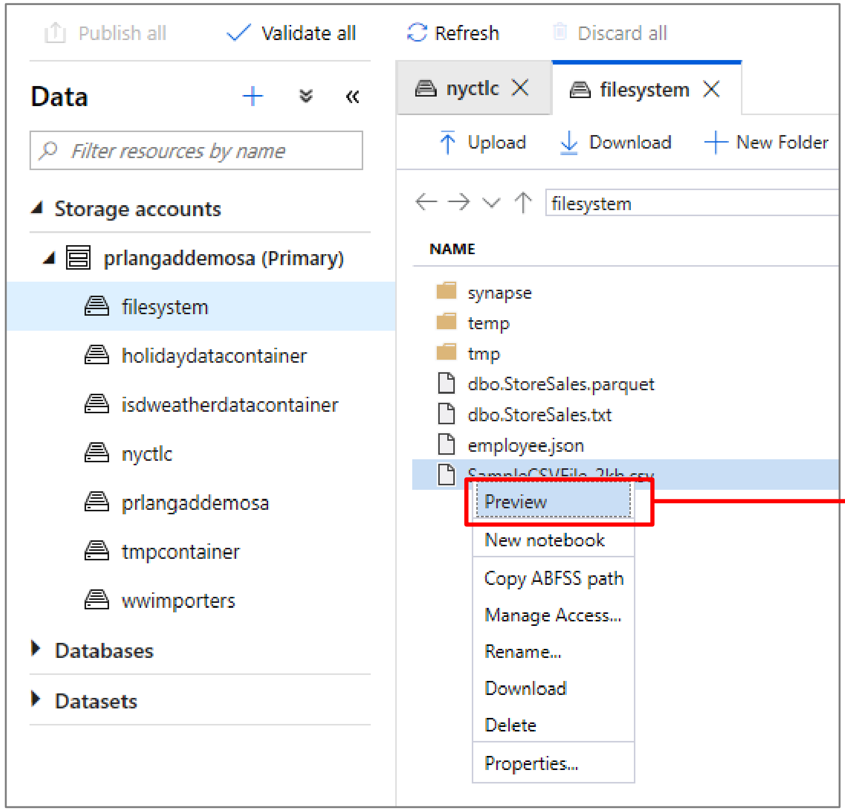
Name: prlangadws2
Region: West US 2
Resource group: prlangadrg
Subscription ID:
58f8824d-32b0-4825-9825-02fa6a801546

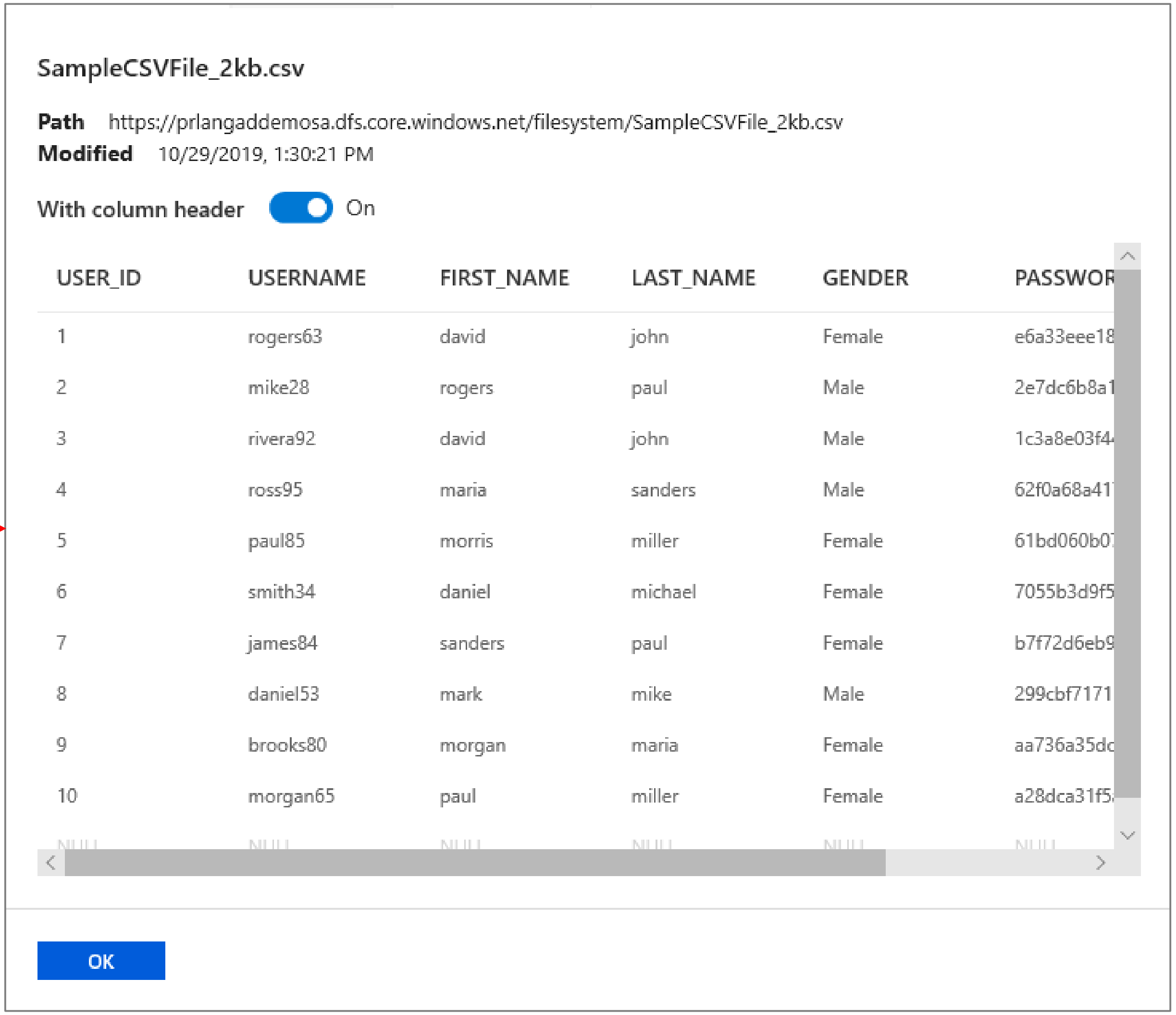
nyctlc > yellow

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, 2:20:38 PM	Folder	

Data Hub – Storage accounts

Preview a sample of your data





SampleCSVFile_2kb.csv

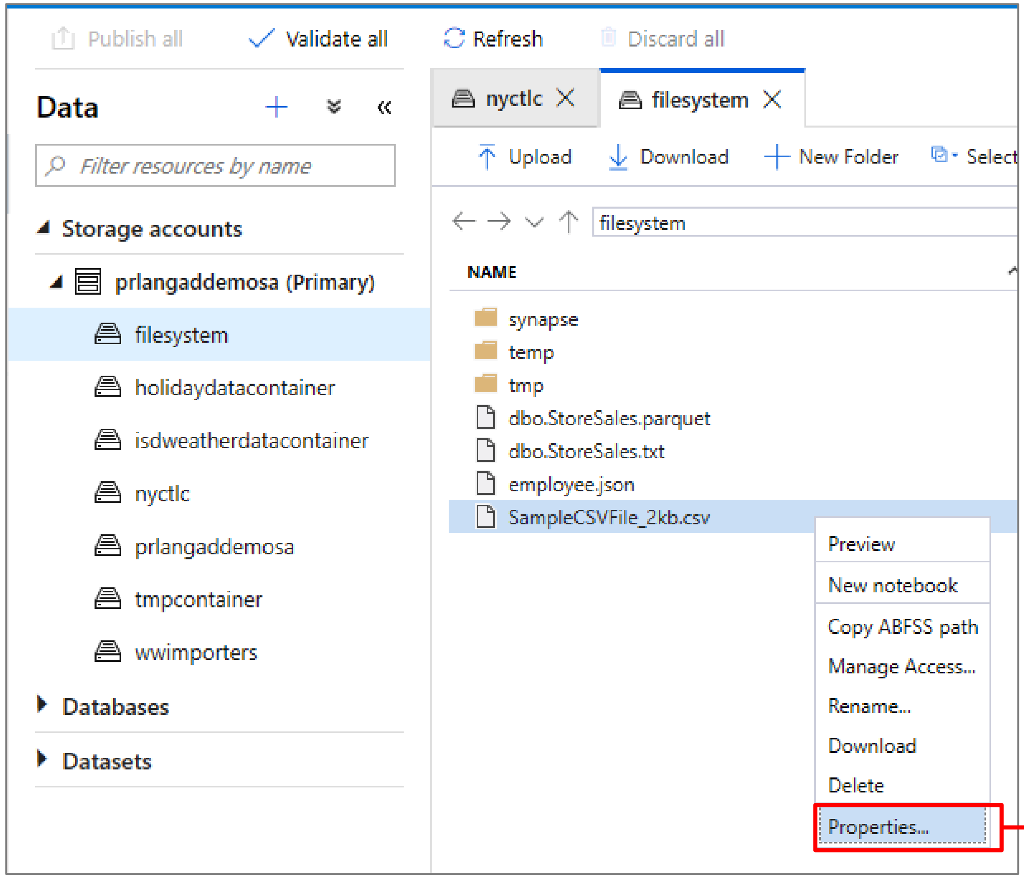
Path https://prlangaddemosa.dfs.core.windows.net/filesystem/SampleCSVFile_2kb.csv
Modified 10/29/2019, 1:30:21 PM

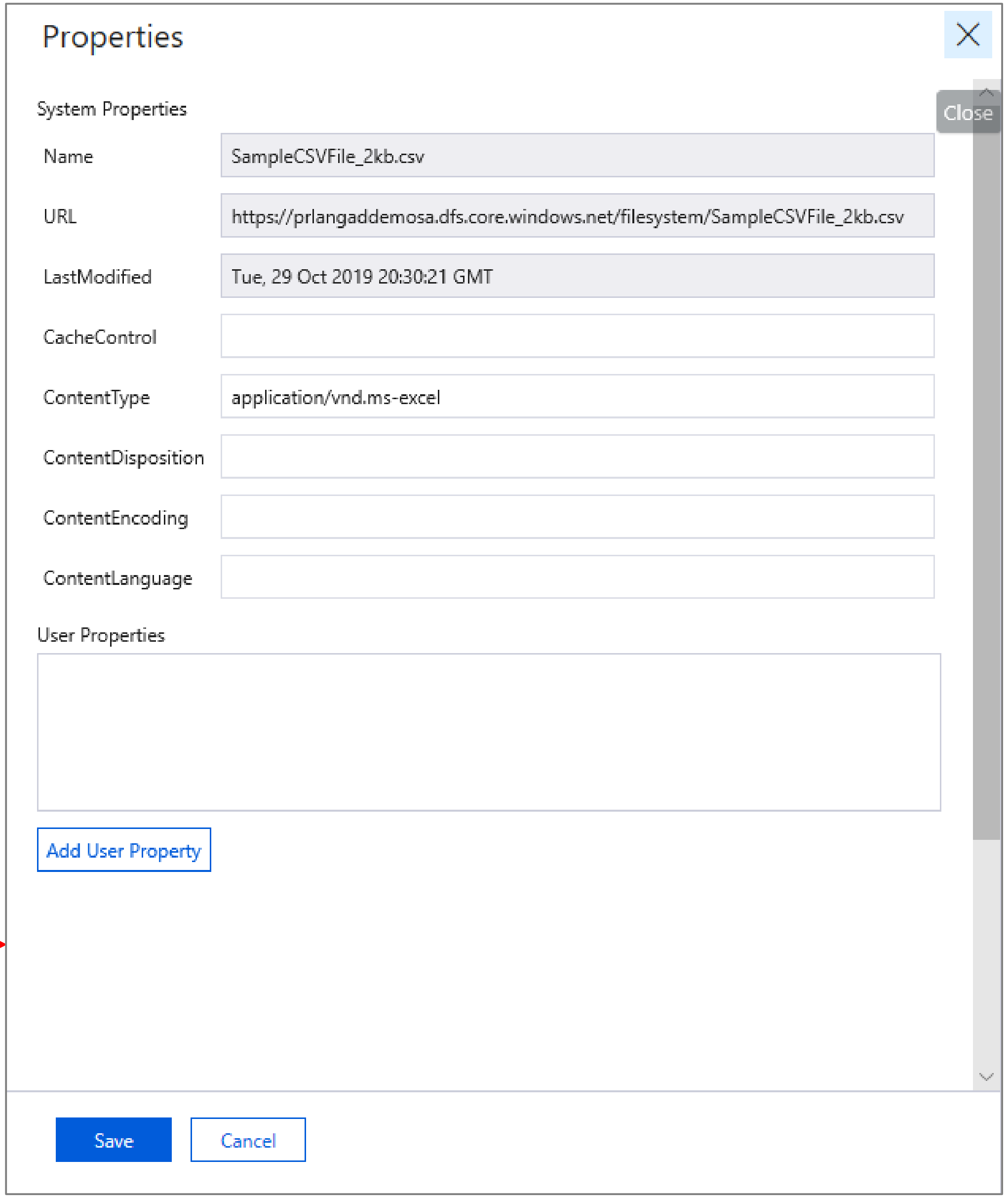
With column header ● On

USER_ID	USERNAME	FIRST_NAME	LAST_NAME	GENDER	PASSWOR
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
NULL	NULL	NULL	NULL	NULL	NULL

Data Hub – Storage accounts

See basic file properties

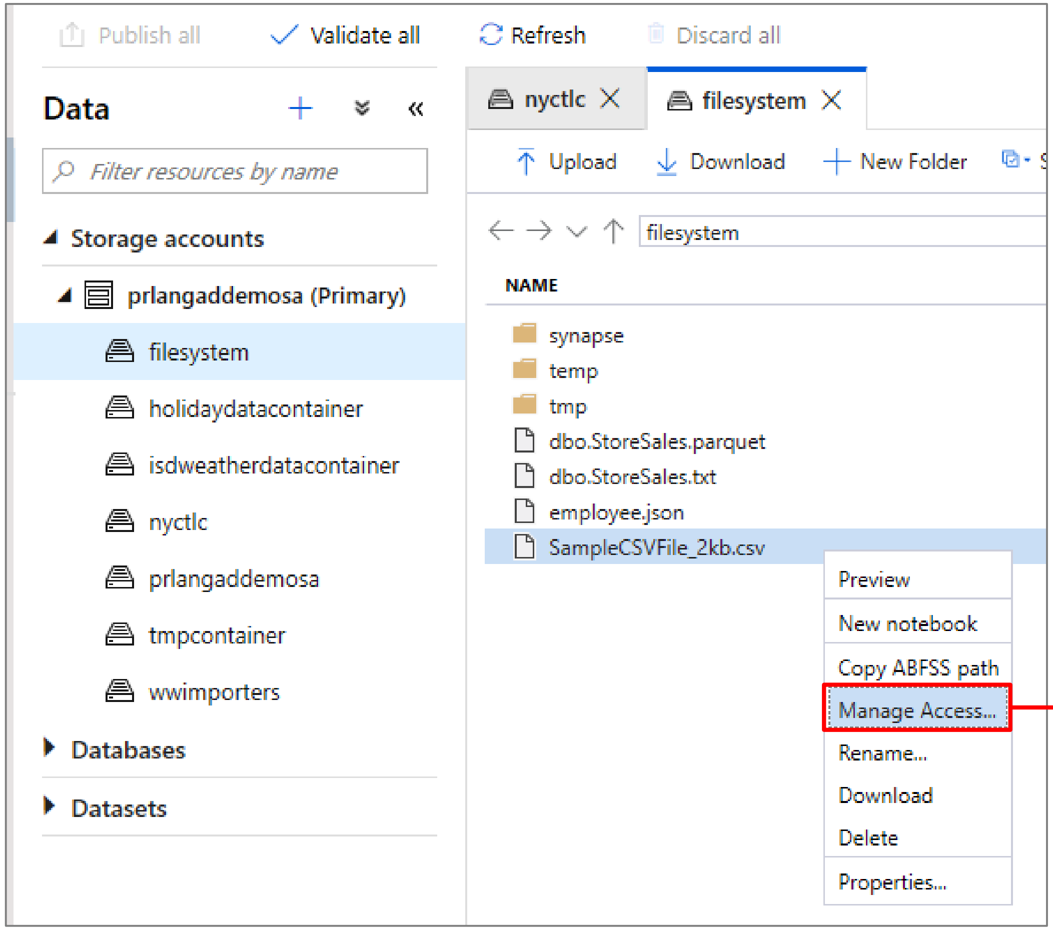


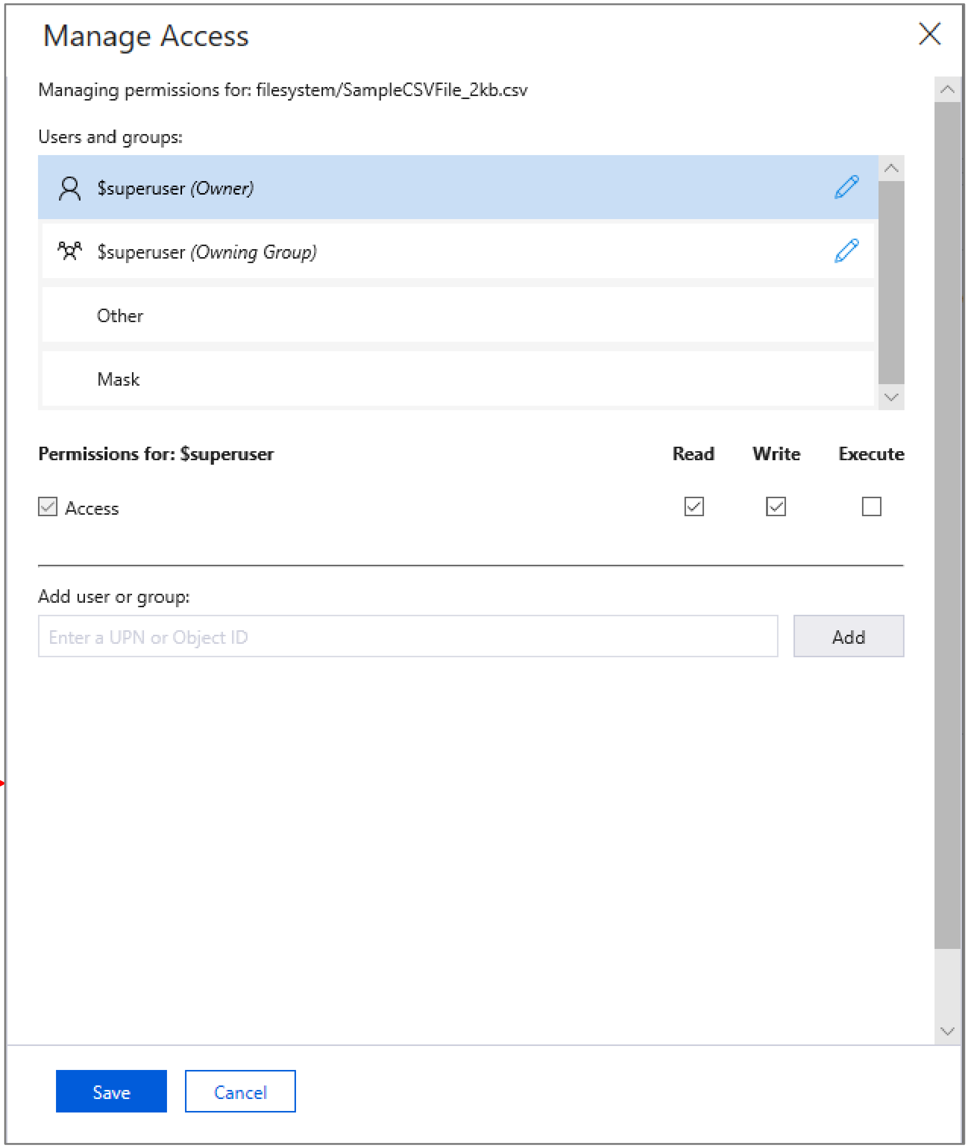


Name	SampleCSVFile_2kb.csv
URL	https://prlangaddemosa.dfs.core.windows.net/filesystem/SampleCSVFile_2kb.csv
LastModified	Tue, 29 Oct 2019 20:30:21 GMT
CacheControl	
ContentType	application/vnd.ms-excel
ContentDisposition	
ContentEncoding	
ContentLanguage	

Data Hub – Storage accounts

Manage Access - Configure standard POSIX ACLs on files and folders





Managing permissions for: filesystem/SampleCSVFile_2kb.csv

Users and groups:

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

Permissions for: $superuser	Read	Write	Execute
☑ Access	☑	☑	□

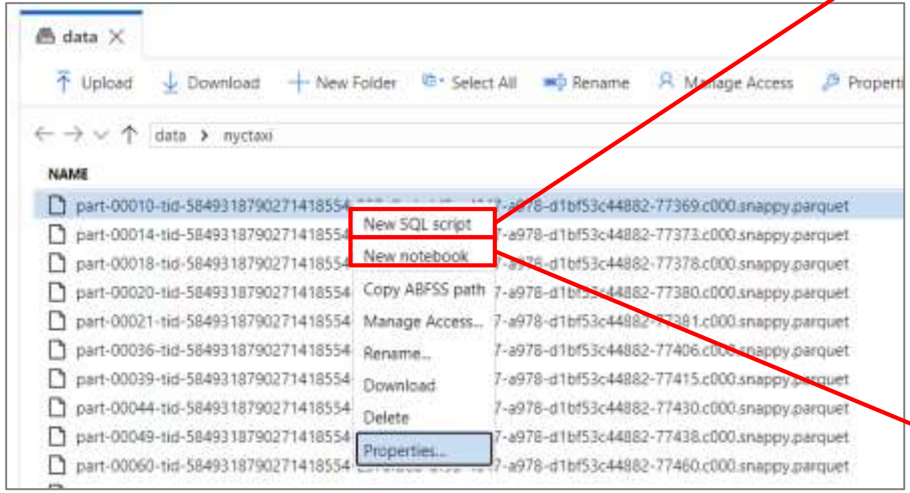
Add user or group:

Enter a UPN or Object ID

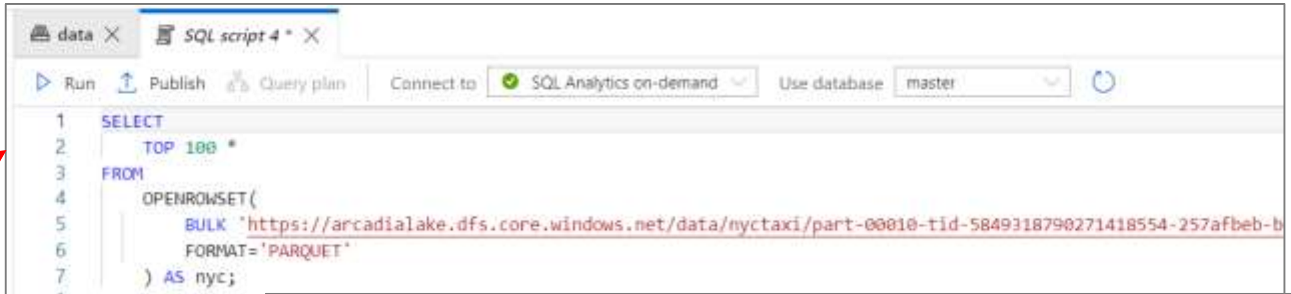
Data Hub – Storage accounts

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

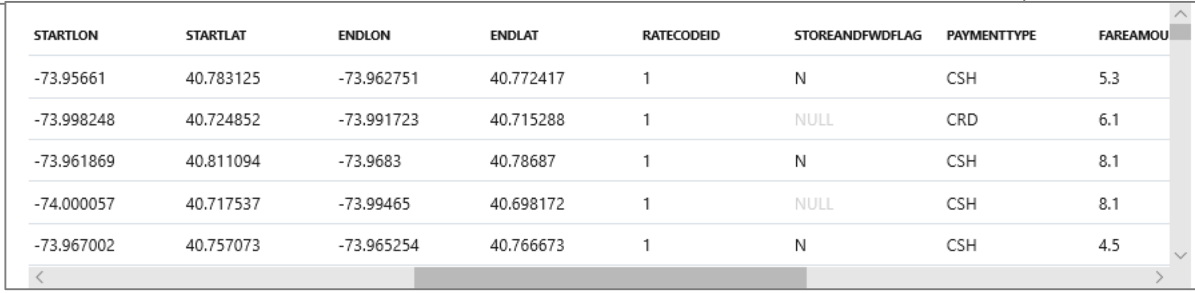
T-SQL or PySpark auto-generated.



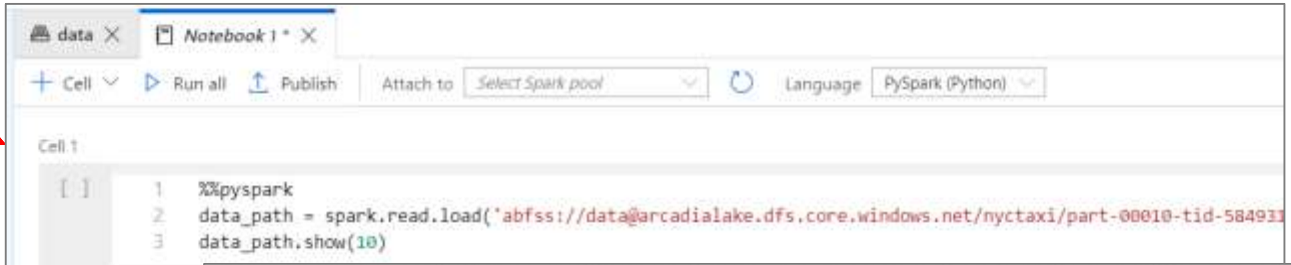
NAME
part-00010-tid-5849318790271418554 7-a978-d1bf53c44882-77369.c000.snappy.parquet
part-00014-tid-5849318790271418554 7-a978-d1bf53c44882-77373.c000.snappy.parquet
part-00018-tid-5849318790271418554 7-a978-d1bf53c44882-77378.c000.snappy.parquet
part-00020-tid-5849318790271418554 7-a978-d1bf53c44882-77380.c000.snappy.parquet
part-00021-tid-5849318790271418554 7-a978-d1bf53c44882-77381.c000.snappy.parquet
part-00036-tid-5849318790271418554 7-a978-d1bf53c44882-77406.c000.snappy.parquet
part-00039-tid-5849318790271418554 7-a978-d1bf53c44882-77415.c000.snappy.parquet
part-00044-tid-5849318790271418554 7-a978-d1bf53c44882-77430.c000.snappy.parquet
part-00049-tid-5849318790271418554 7-a978-d1bf53c44882-77438.c000.snappy.parquet
part-00060-tid-5849318790271418554 7-a978-d1bf53c44882-77460.c000.snappy.parquet



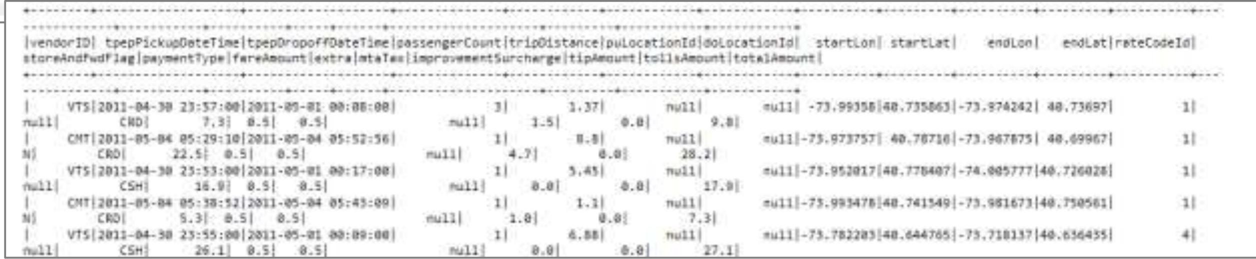
```sql
SELECT
    TOP 100 *
FROM
    OPENROWSET(
        BULK 'https://arcadialake.dfs.core.windows.net/data/nyctaxi/part-00010-tid-5849318790271418554-257afbeb-b
        FORMAT='PARQUET'
    ) AS nyc;
```



STARTLON	STARTLAT	ENDLON	ENDLAT	RATECODEID	STOREANDFWDFLAG	PAYMENTTYPE	FAREAMOU
-73.95661	40.783125	-73.962751	40.772417	1	N	CSH	5.3
-73.998248	40.724852	-73.991723	40.715288	1	NULL	CRD	6.1
-73.961869	40.811094	-73.9683	40.78687	1	N	CSH	8.1
-74.000057	40.717537	-73.99465	40.698172	1	NULL	CSH	8.1
-73.967002	40.757073	-73.965254	40.766673	1	N	CSH	4.5



```python
%%pyspark
data_path = spark.read.load('abfss://data@arcadialake.dfs.core.windows.net/nyctaxi/part-00010-tid-58493
data_path.show(10)
```

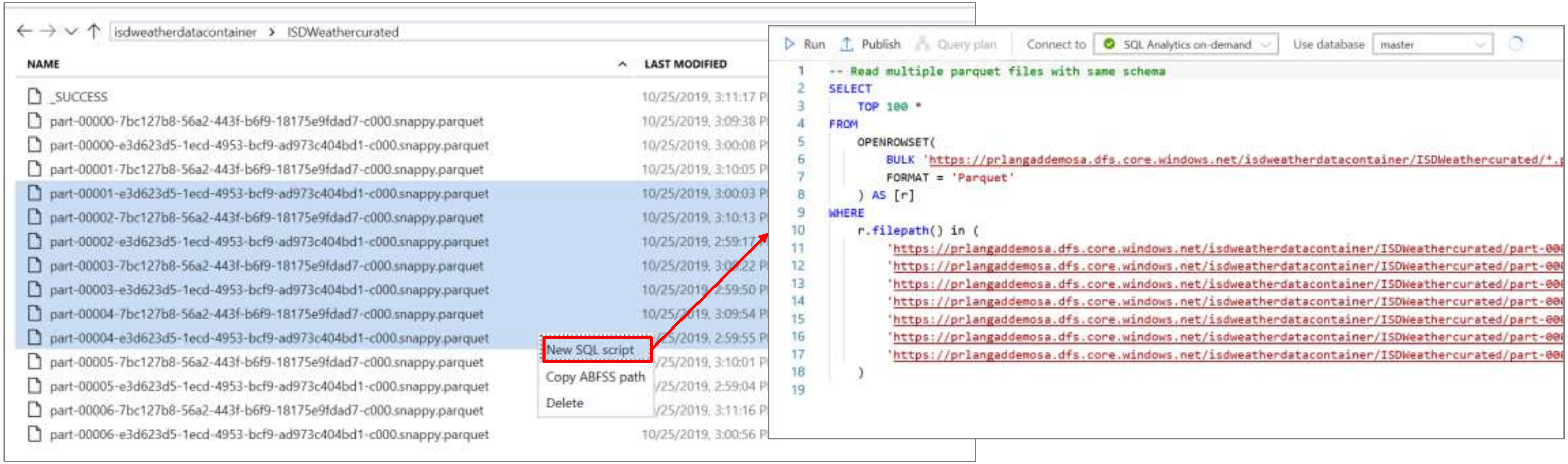


```
+--------+-------------------+-------------------+--------------+------------+------------+------------+----------+---------+----------+---------+----------+
---------------+-----------+----------+-----+------+--------------------+---------+-----------+-----------+
|vendorID| tpepPickupDateTime|tpepDropoffDateTime|passengerCount|tripDistance|puLocationId|doLocationId|  startLon| startLat|    endLon|   endLat|rateCodeId|
storeAndFwdFlag|paymentType|fareAmount|extra|mtaTax|improvementSurcharge|tipAmount|tollsAmount|totalAmount|
+--------+-------------------+-------------------+--------------+------------+------------+------------+----------+---------+----------+---------+----------+
---------------+-----------+----------+-----+------+--------------------+---------+-----------+-----------+
|     VTS|2011-04-30 23:57:00|2011-05-01 00:08:00|             3|        1.37|        null|        null| -73.99358|40.735863|-73.974242| 40.73697|         1|
null|        CRD|       7.3|  0.5|   0.5|                null|      1.5|        0.0|        9.8|
|     CMT|2011-05-04 05:29:10|2011-05-04 05:52:56|             1|         8.8|        null|        null|-73.973757| 40.78716|-73.967875| 40.69967|         1|
N|        CRD|      22.5|  0.5|   0.5|                null|      4.7|        0.0|       28.2|
|     VTS|2011-04-30 23:53:00|2011-05-01 00:17:00|             1|        5.45|        null|        null|-73.952017|40.778407|-74.005777|40.726028|         1|
null|        CSH|      16.9|  0.5|   0.5|                null|      0.0|        0.0|       17.9|
|     CMT|2011-05-04 05:38:52|2011-05-04 05:43:09|             1|         1.1|        null|        null|-73.993478|40.741549|-73.981673|40.750561|         1|
N|        CRD|       5.3|  0.5|   0.5|                null|      1.0|        0.0|        7.3|
|     VTS|2011-04-30 23:55:00|2011-05-01 00:09:00|             1|        6.88|        null|        null|-73.782203|40.644765|-73.718137|40.636435|         4|
null|        CSH|      26.1|  0.5|   0.5|                null|      0.0|        0.0|       27.1|
```

Data Hub – Storage accounts

SQL Script from Multiple files

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

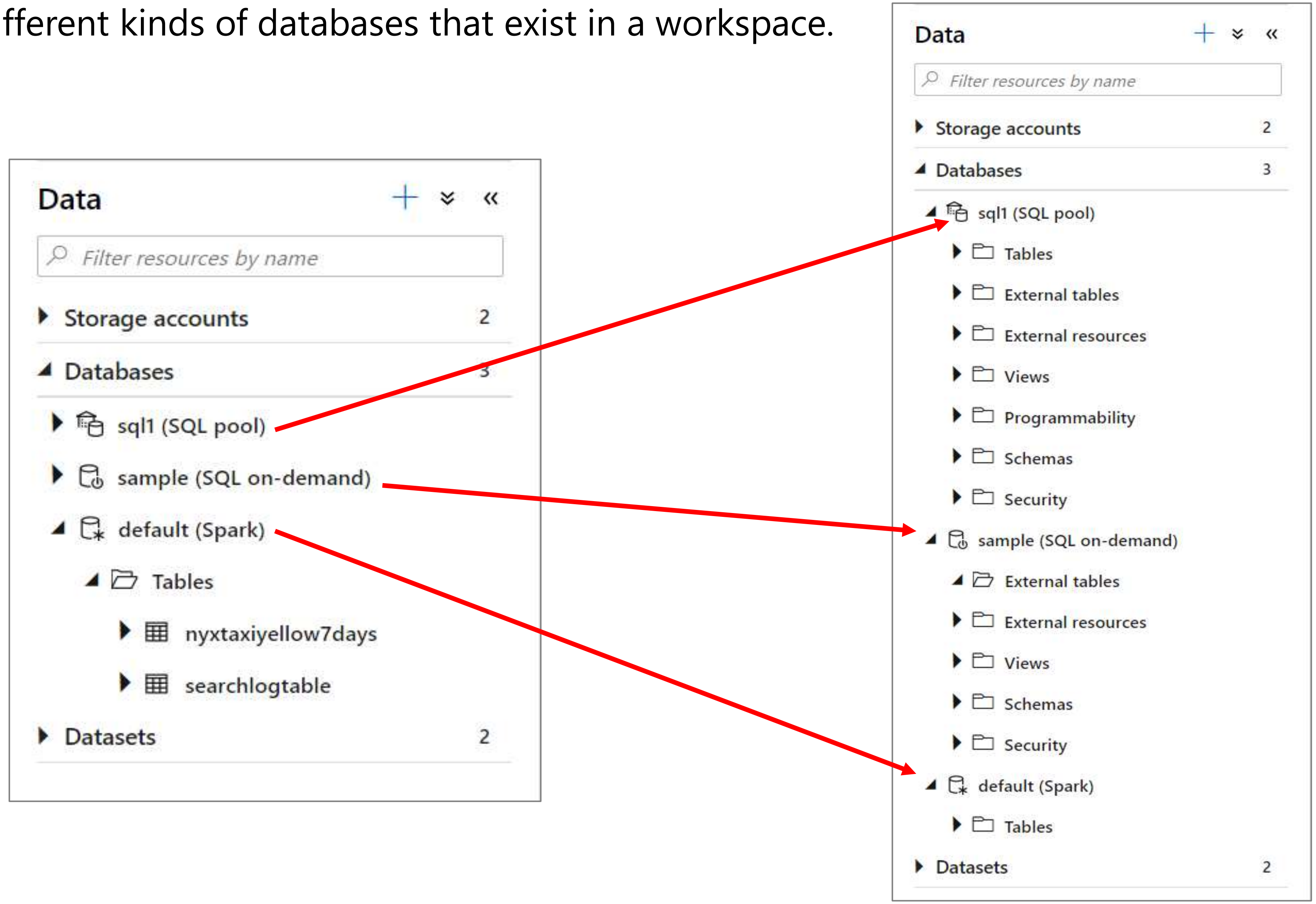


NAME	LAST MODIFIED
_SUCCESS	10/25/2019, 3:11:17 P
part-00000-7bc127b8-56a2-443f-b6f9-18175e9fdad7-c000.snappy.parquet	10/25/2019, 3:09:38 P
part-00000-e3d623d5-1ecd-4953-bcf9-ad973c404bd1-c000.snappy.parquet	10/25/2019, 3:00:08 P
part-00001-7bc127b8-56a2-443f-b6f9-18175e9fdad7-c000.snappy.parquet	10/25/2019,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

```sql
-- Read multiple parquet files with same schema
SELECT
    TOP 100 *
FROM
    OPENROWSET(
        BULK 'https://prlangaddemosa.dfs.core.windows.net/isdweatherdatacontainer/ISDWeathercurated/*.p
        FORMAT = 'Parquet'
    ) AS [r]
WHERE
    r.filepath() in (
        'https://prlangaddemosa.dfs.core.windows.net/isdweatherdatacontainer/ISDWeathercurated/part-000
        'https://prlangaddemosa.dfs.core.windows.net/isdweatherdatacontainer/ISDWeathercurated/part-000
        'https://prlangaddemosa.dfs.core.windows.net/isdweatherdatacontainer/ISDWeathercurated/part-000
        'https://prlangaddemosa.dfs.core.windows.net/isdweatherdatacontainer/ISDWeathercurated/part-000
        'https://prlangaddemosa.dfs.core.windows.net/isdweatherdatacontainer/ISDWeathercurated/part-000
        'https://prlangaddemosa.dfs.core.windows.net/isdweatherdatacontainer/ISDWeathercurated/part-000
        'https://prlangaddemosa.dfs.core.windows.net/isdweatherdatacontainer/ISDWeathercurated/part-000
    )
```

Data Hub – Databases

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



SQL pool

SQL on-demand

Spark

Data

Filter resources by name

- Storage accounts 2
- Databases 3
  - sql1 (SQL pool)
  - sample (SQL on-demand)
  - default (Spark)
    - Tables
      - nyxtaxiyellow7days
      - searchlogtable
- Datasets 2

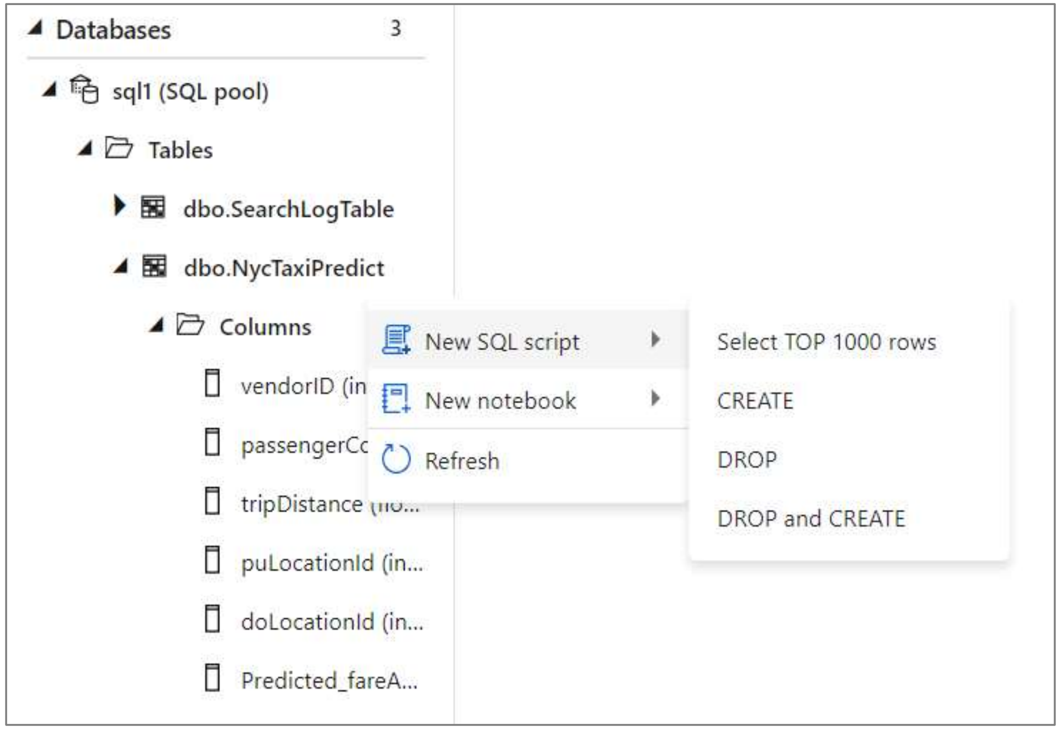
Data

Filter resources by name

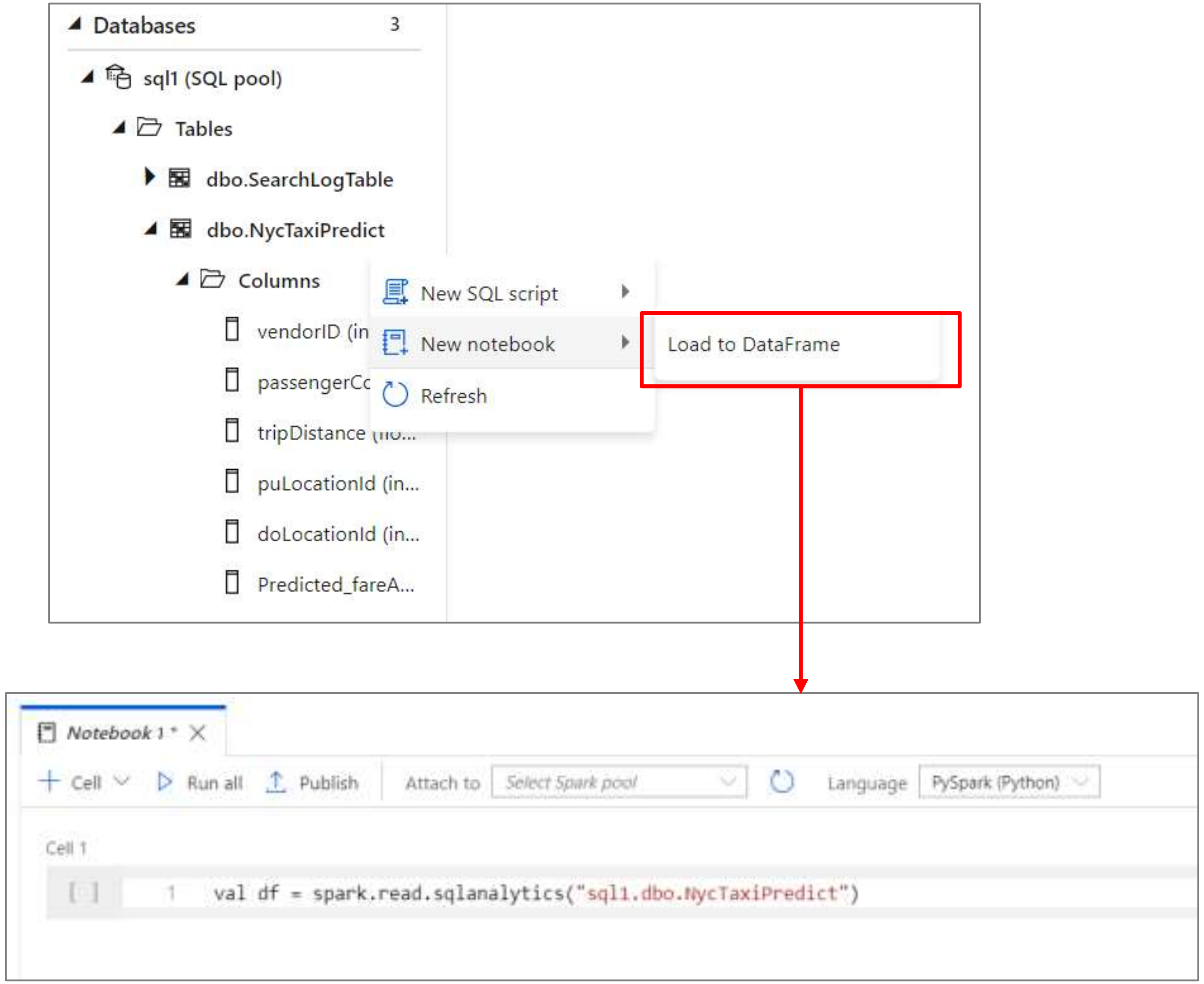
- Storage accounts 2
- Databases 3
  - sql1 (SQL pool)
    - Tables
    - External tables
    - External resources
    - Views
    - Programmability
    - Schemas
    - Security
  - sample (SQL on-demand)
    - External tables
    - External resources
    - Views
    - Schemas
    - Security
  - default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

Data Hub – Databases

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



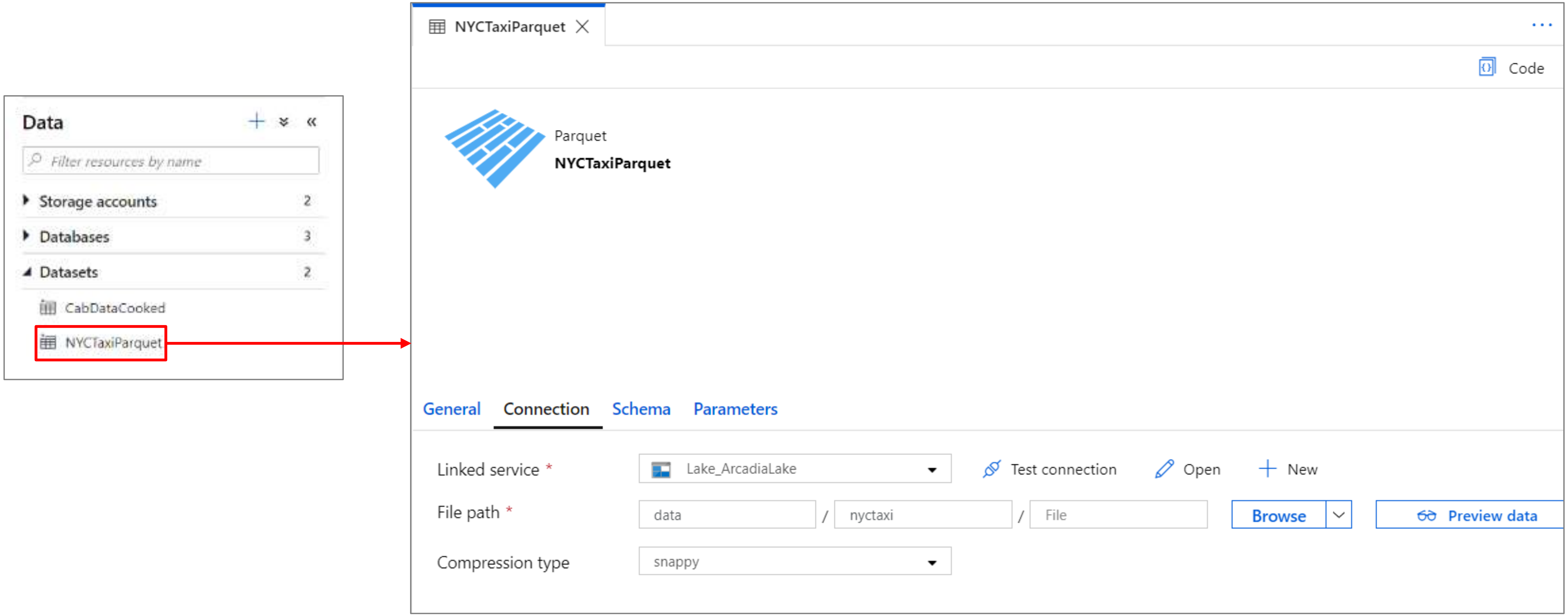
Starting from a table, auto-generate a single line of PySpark code that makes it easy to load a SQL table into a Spark dataframe



```
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.



Data

Filter resources by name

- Storage accounts 2
- Databases 3
- Datasets 2
  - CabDataCooked
  - NYCTaxiParquet

NYCTaxiParquet

Code

Parquet

NYCTaxiParquet

General | Connection | Schema | Parameters

Linked service *: Lake_ArcadiaLake

File path *: data / nyctaxi / File

Compression type: snappy

Synapse Studio
Develop hub

Develop Hub

Overview

It provides development experience to query, analyze, model data

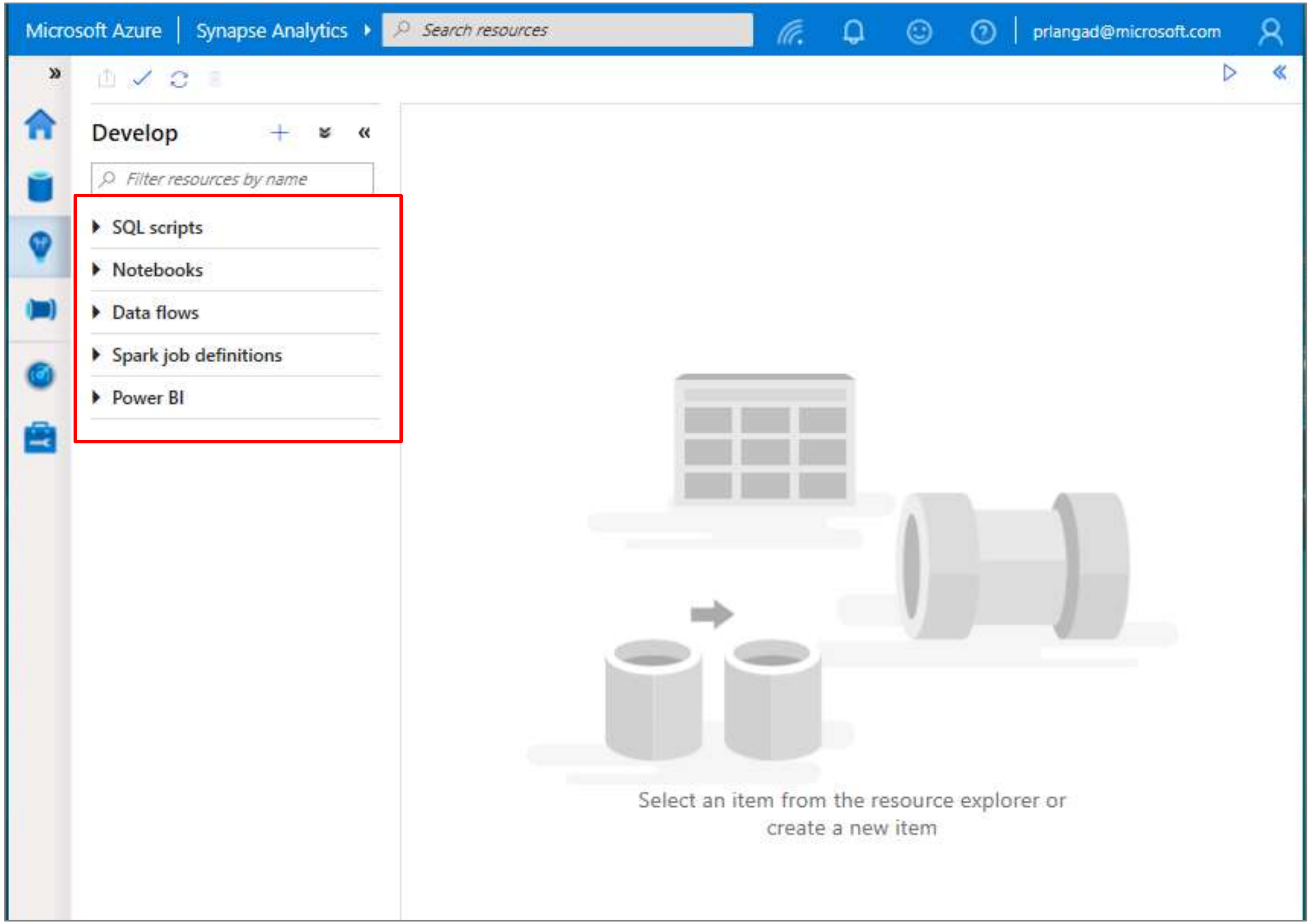
Benefits

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



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

Develop Hub - SQL scripts

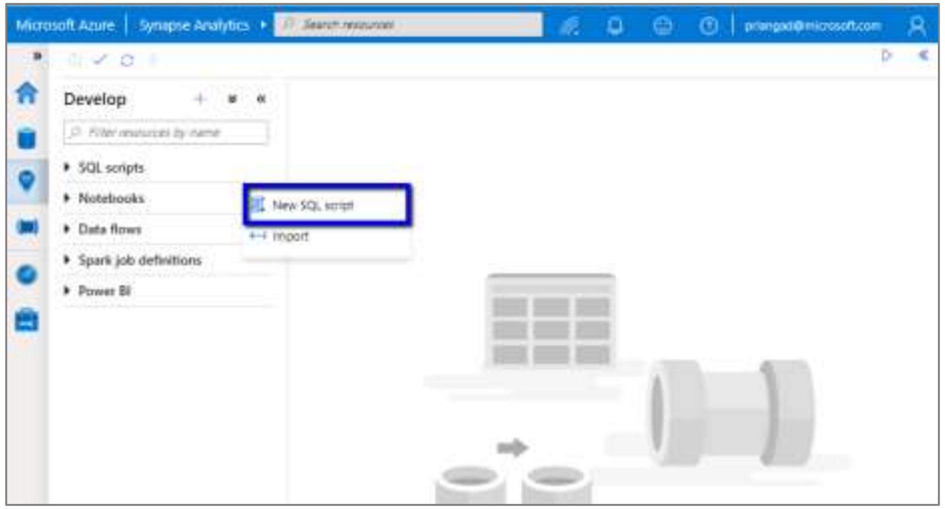
SQL Script

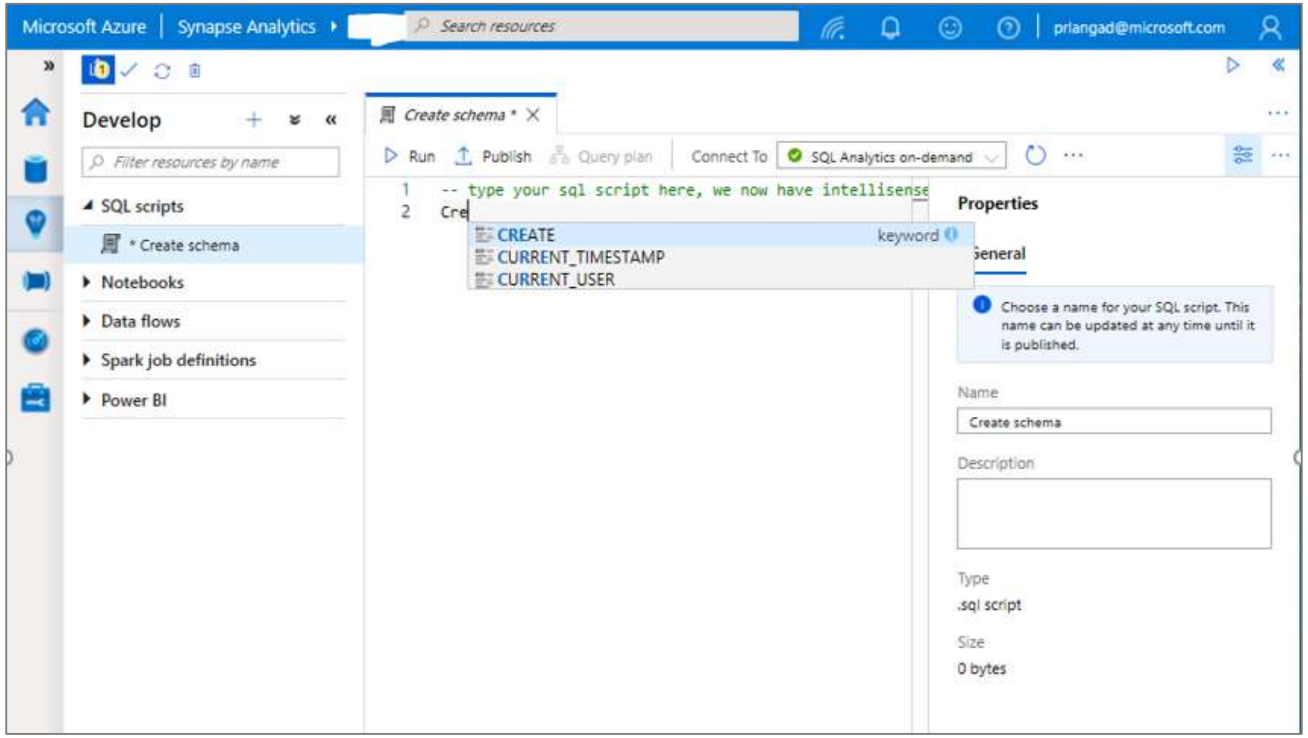
Authoring SQL Scripts

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





```
-- type your sql script here, we now have intellisense
Cre
```

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

Name: Create schema

Description:

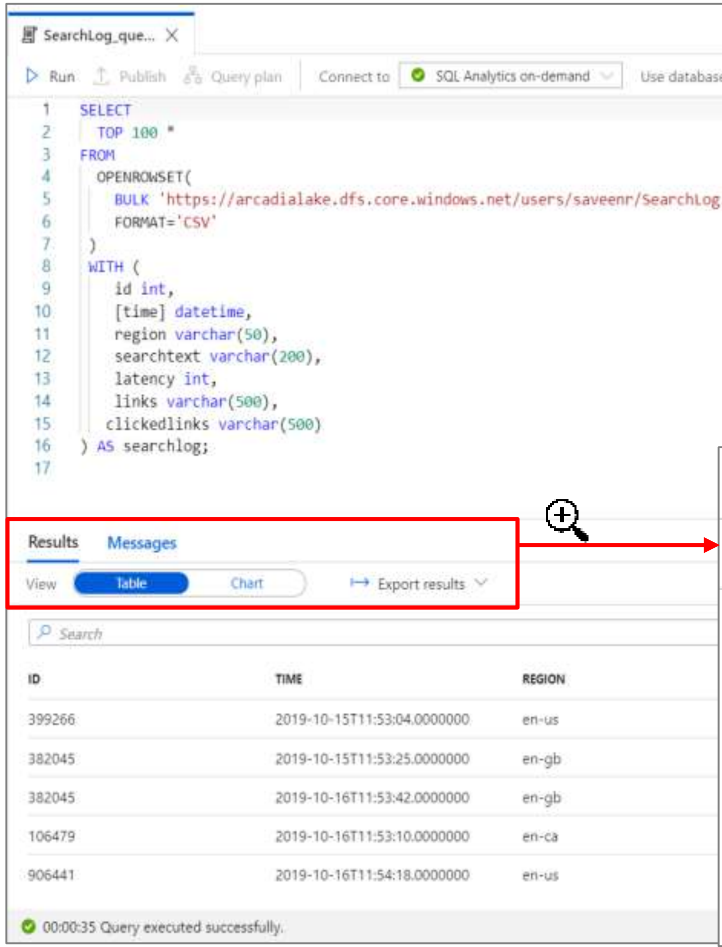
Type: .sql script

Size: 0 bytes

Develop Hub - SQL scripts

SQL Script

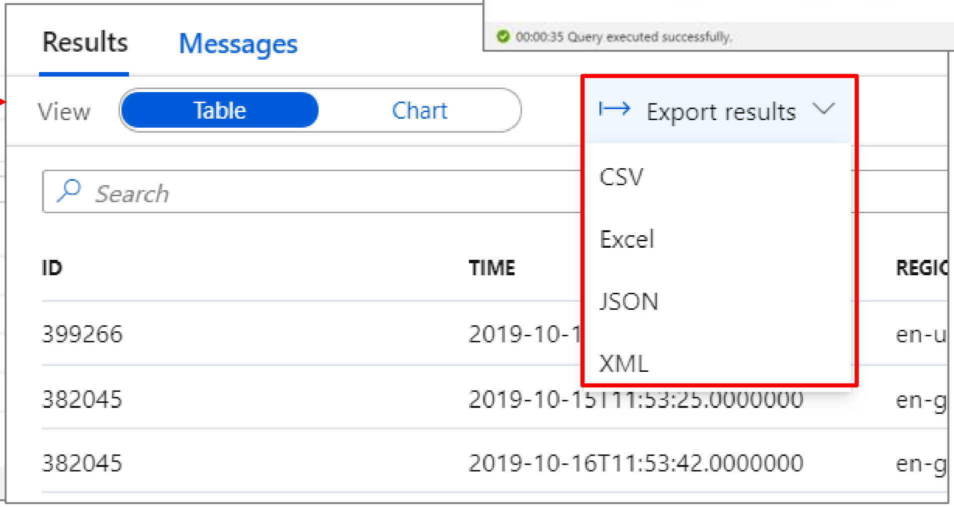
View results in Table or Chart form and export results in several popular formats



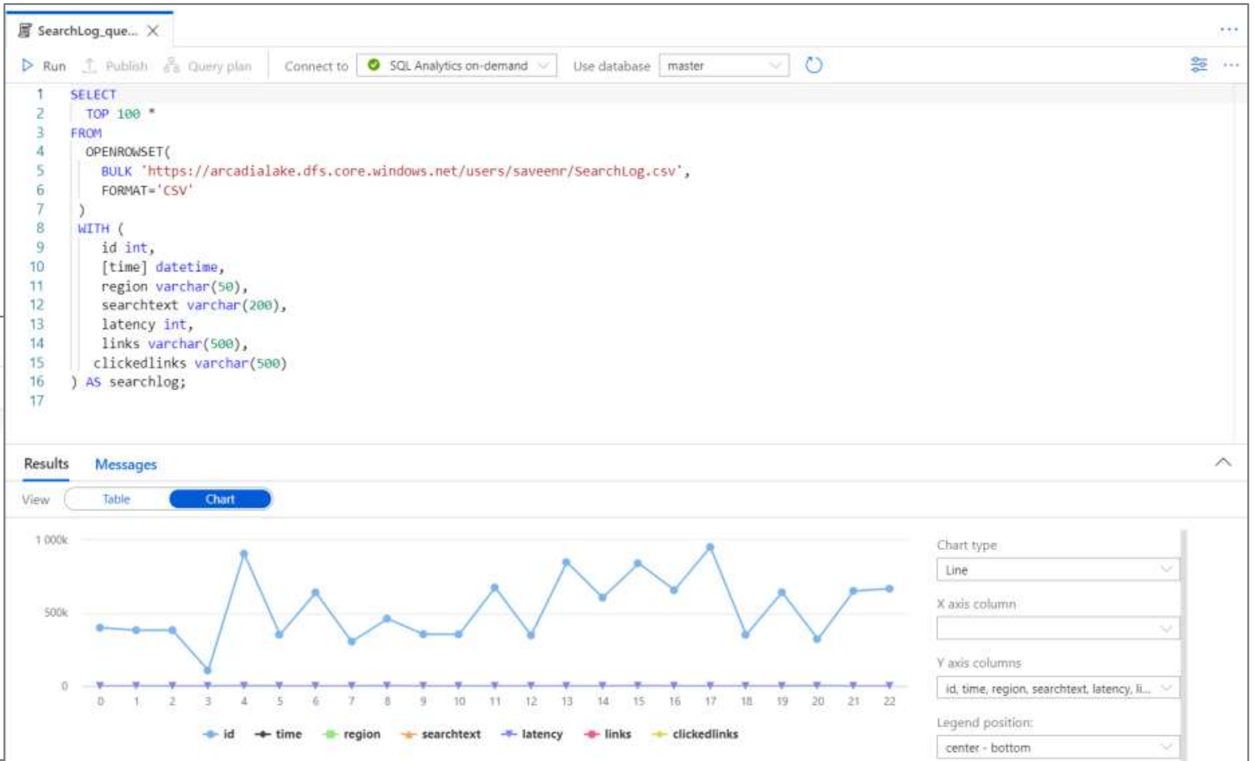
```sql
SELECT
  TOP 100 *
FROM
  OPENROWSET(
    BULK 'https://arcadialake.dfs.core.windows.net/users/saveenr/SearchLog.csv',
    FORMAT='CSV'
  )
  WITH (
    id int,
    [time] datetime,
    region varchar(50),
    searchtext varchar(200),
    latency int,
    links varchar(500),
   clickedlinks varchar(500)
) AS searchlog;
```

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

00:00:35 Query executed successfully.



ID	TIME	REGIO
399266	2019-10-1	en-u
382045	2019-10-15T11:53:25.0000000	en-g
382045	2019-10-16T11:53:42.0000000	en-g



```sql
SELECT
  TOP 100 *
FROM
  OPENROWSET(
    BULK 'https://arcadialake.dfs.core.windows.net/users/saveenr/SearchLog.csv',
    FORMAT='CSV'
  )
  WITH (
    id int,
    [time] datetime,
    region varchar(50),
    searchtext varchar(200),
    latency int,
    links varchar(500),
   clickedlinks varchar(500)
) AS searchlog;
```

00:00:35 Query executed successfully.

Develop Hub - Notebooks

Notebooks

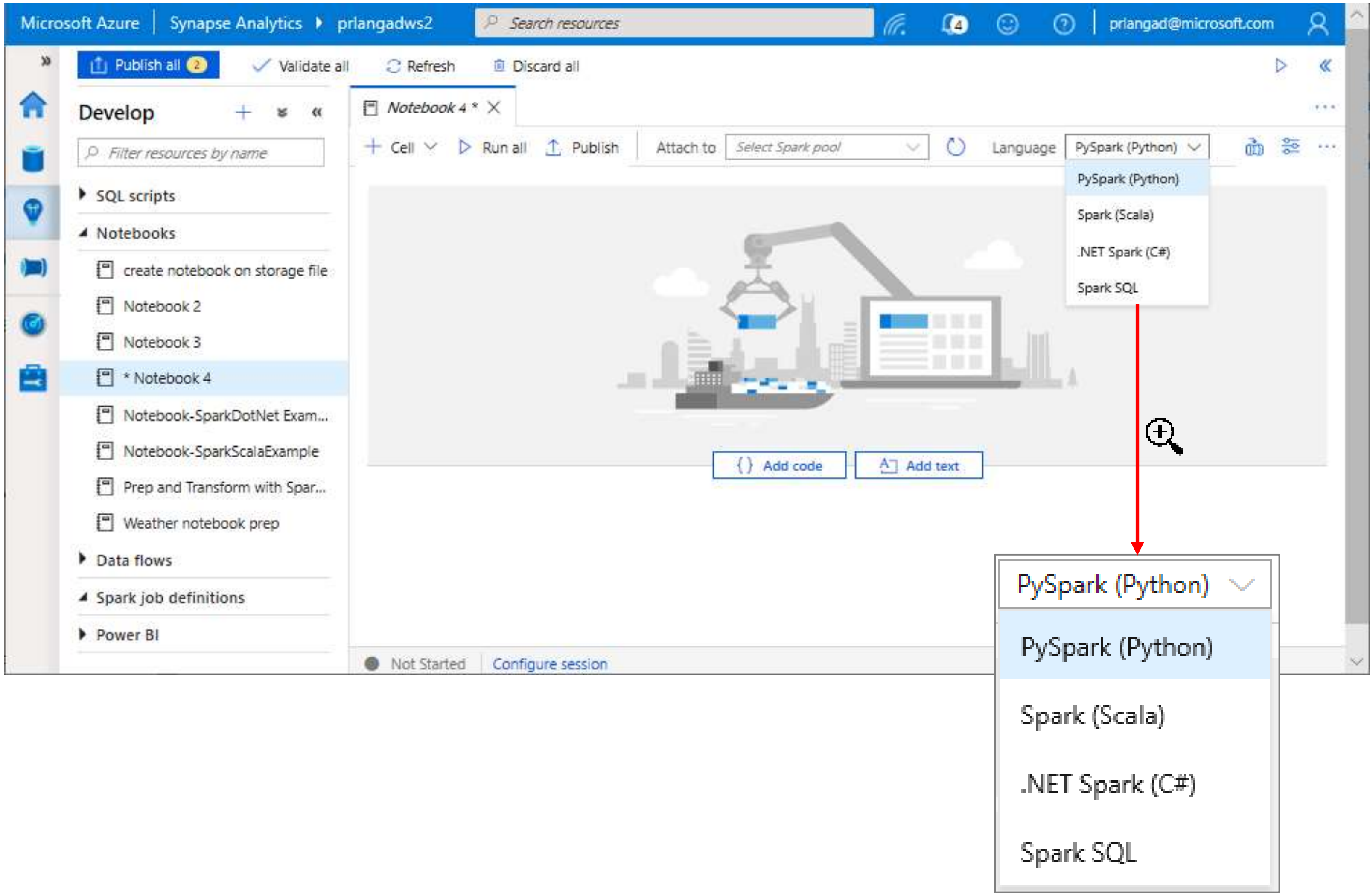
Allows to write multiple languages in one notebook

%%<Name of language>

Offers use of temporary tables across languages

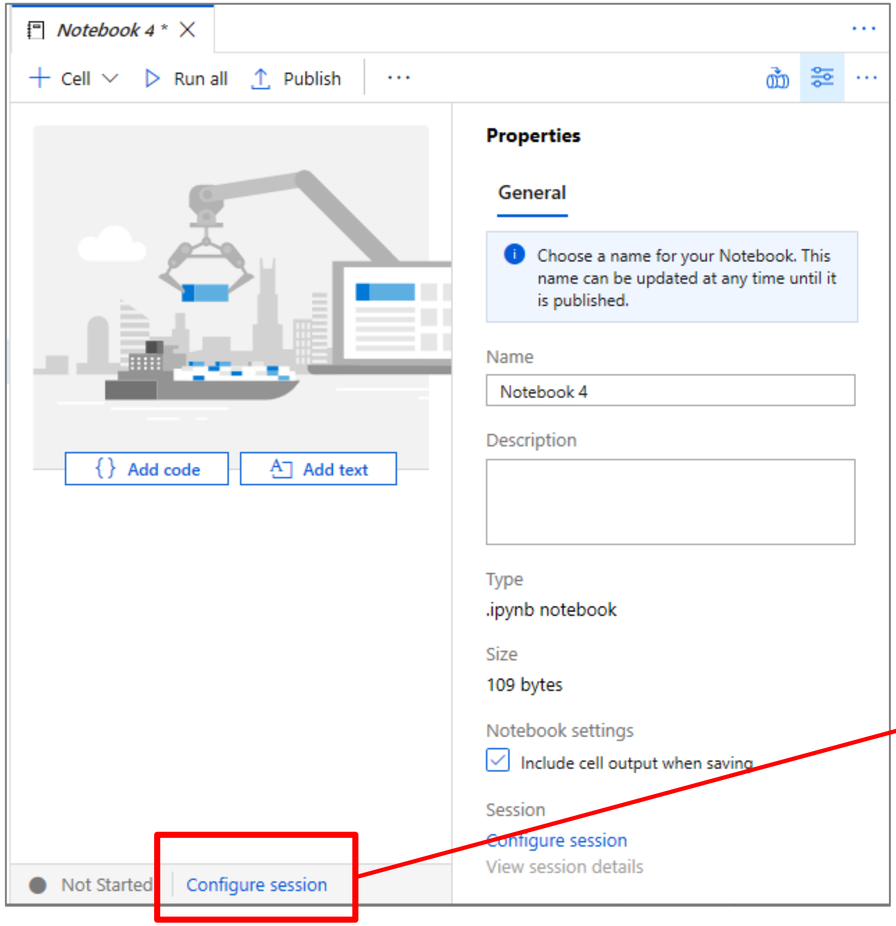
Language support for Syntax highlight, syntax error, syntax code completion, smart indent, code folding

Export results



Develop Hub - Notebooks

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



Properties

General

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

Name: Notebook 4

Description:

Type: .ipynb notebook

Size: 109 bytes

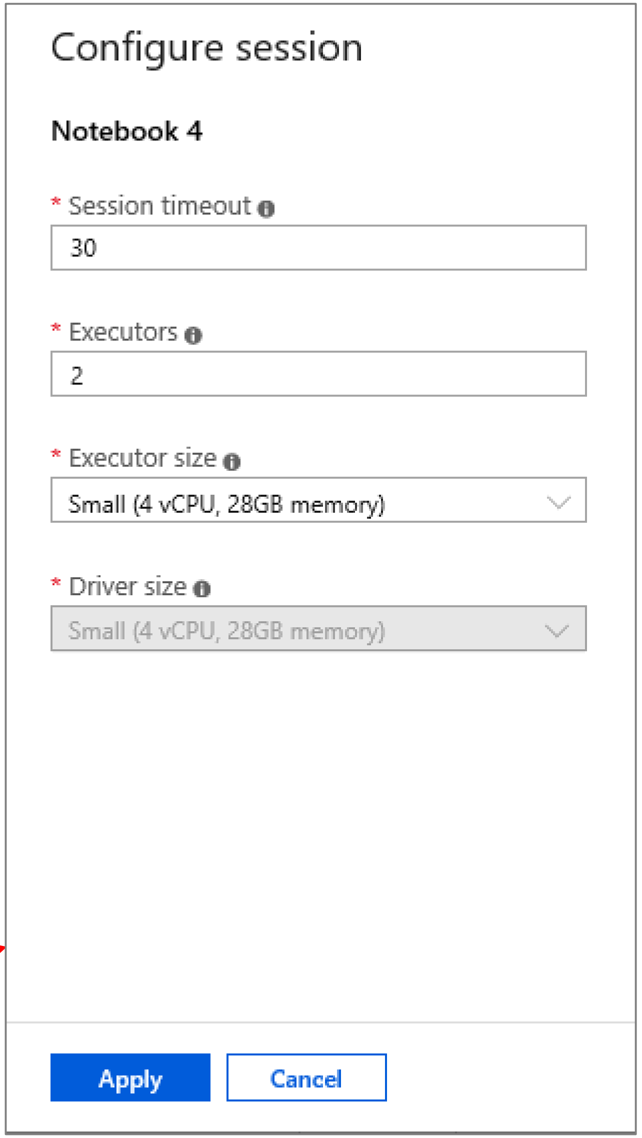
Notebook settings

☑ Include cell output when saving

Session

Configure session

View session details



Configure session

Notebook 4

* Session timeout: 30

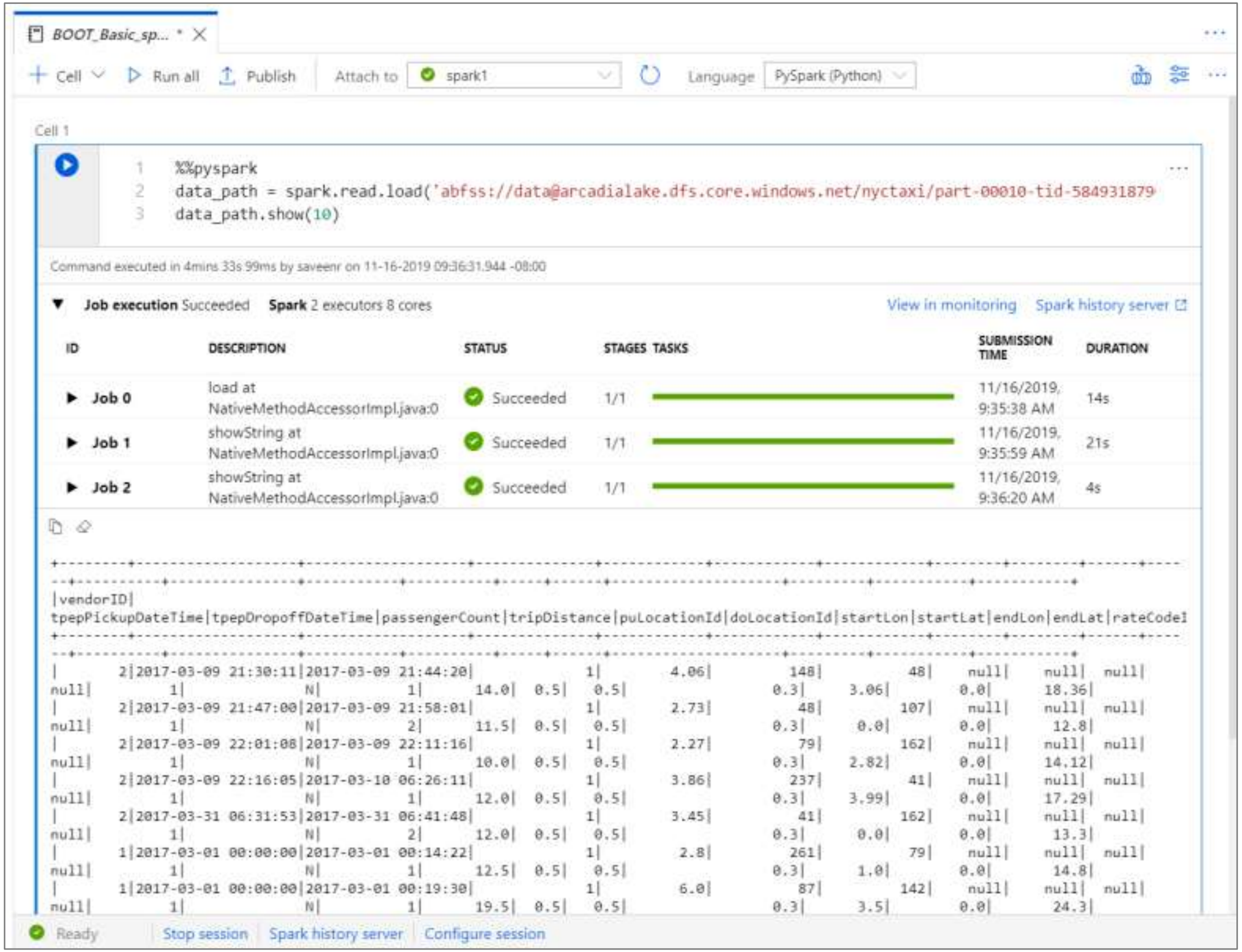
* Executors: 2

* Executor size: Small (4 vCPU, 28GB memory)

* Driver size: Small (4 vCPU, 28GB memory)

Develop Hub - Notebooks

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



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

Command executed in 4mins 33s 99ms by saveenr on 11-16-2019 09:36:31.944 -08:00

Job execution Succeeded **Spark** 2 executors 8 cores

View in monitoring   Spark history server

ID	DESCRIPTION	STATUS	STAGES	TASKS	SUBMISSION TIME	DURATION
Job 0	load at NativeMethodAccessorImpl.java:0	Succeeded	1/1		11/16/2019, 9:35:38 AM	14s
Job 1	showString at NativeMethodAccessorImpl.java:0	Succeeded	1/1		11/16/2019, 9:35:59 AM	21s
Job 2	showString at NativeMethodAccessorImpl.java:0	Succeeded	1/1		11/16/2019, 9:36:20 AM	4s

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

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…)

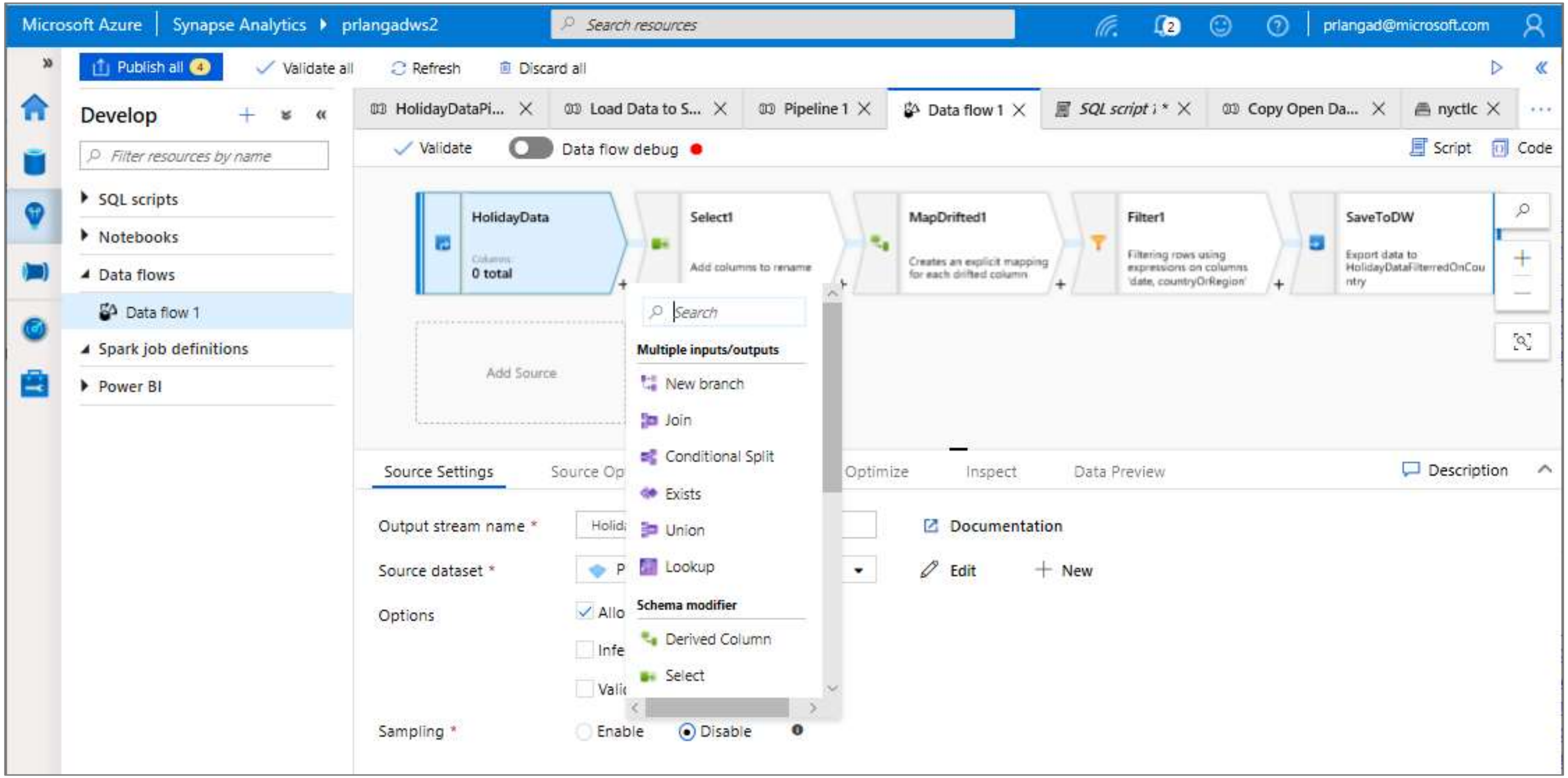
Data lineage – Capturing sink column lineage & impact analysis(invaluable if this is for enterprise deployment)

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

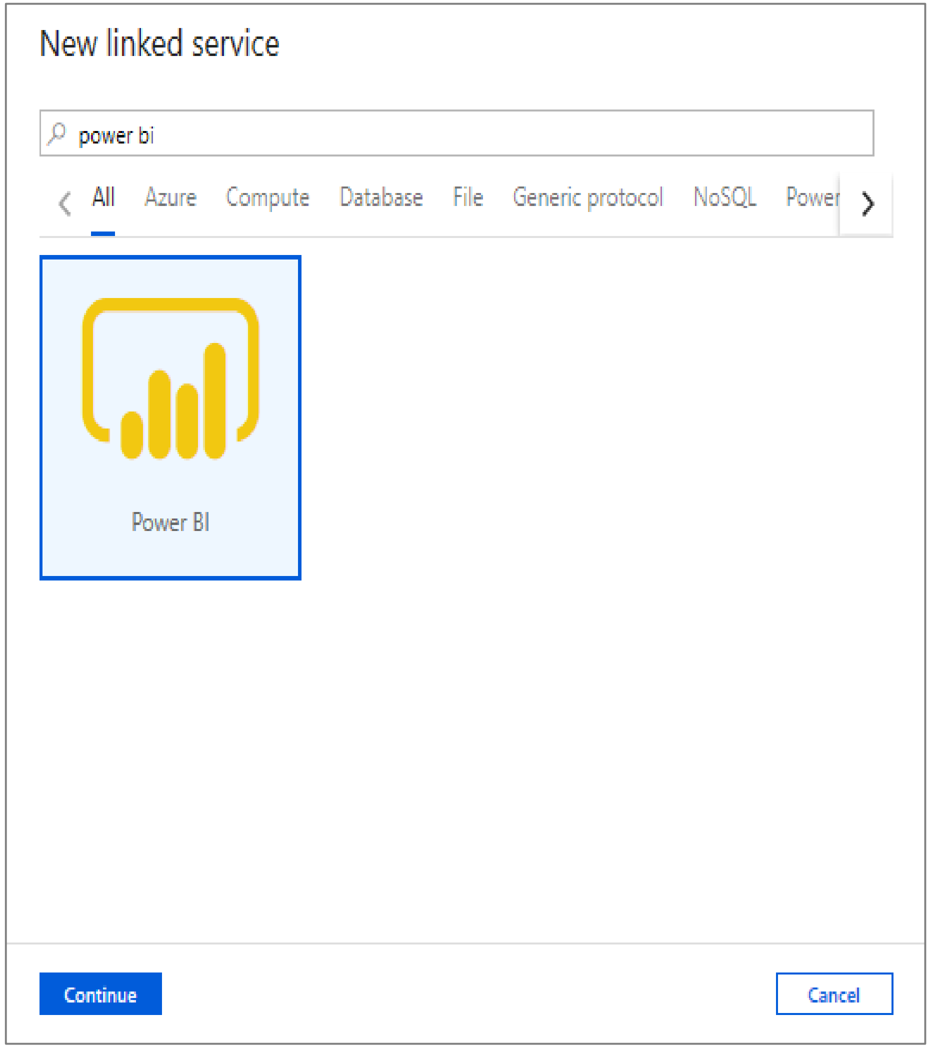
Overview

Create Power BI reports in the workspace

Provides access to published reports in the workspace

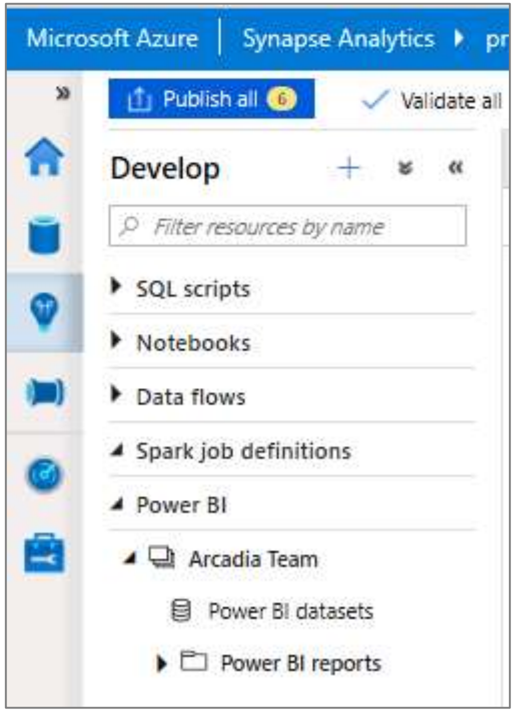
Update reports real time from Synapse workspace to get it reflected on Power BI service

Visually explore and analyze data



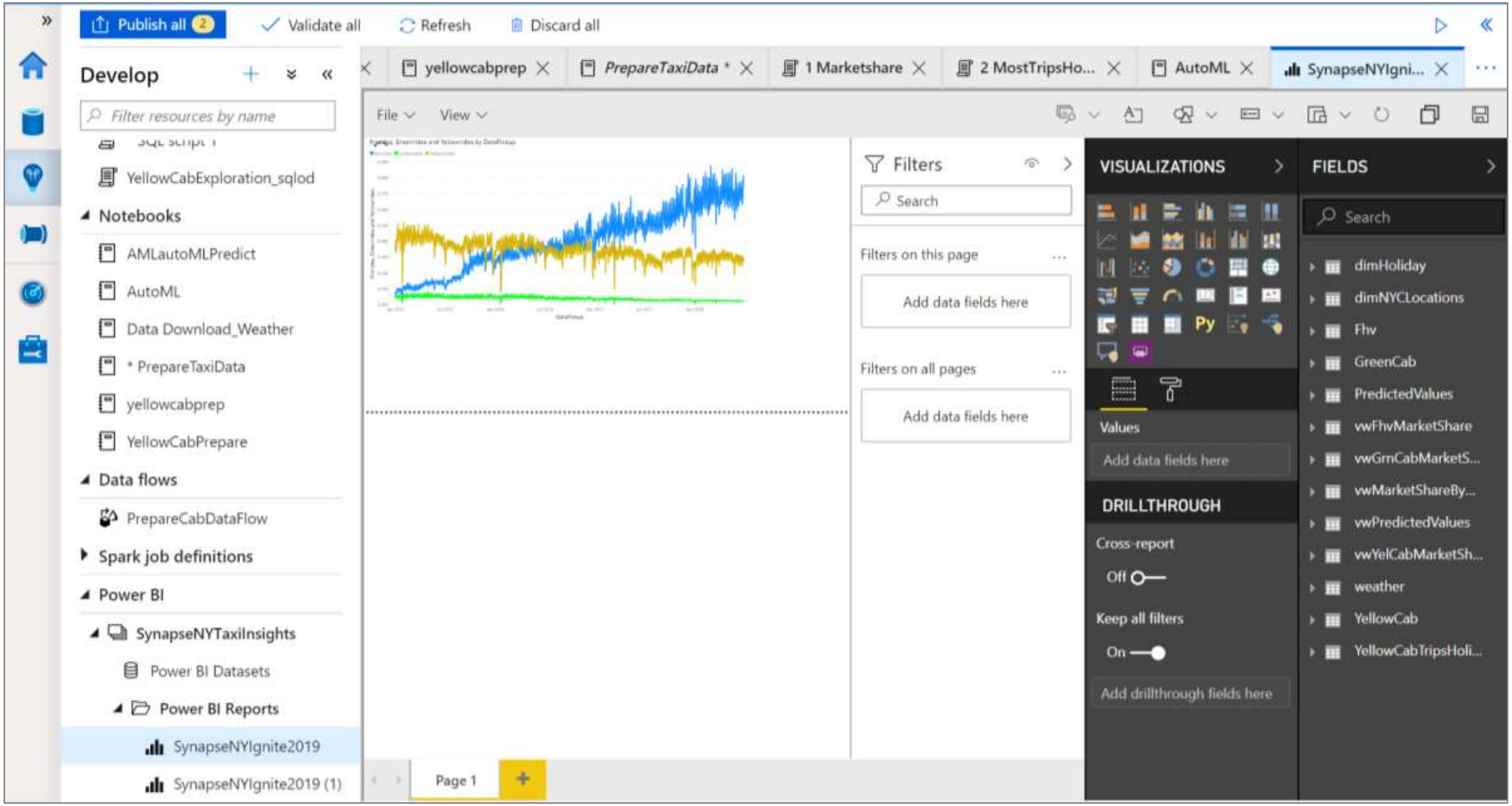
New linked service

power bi



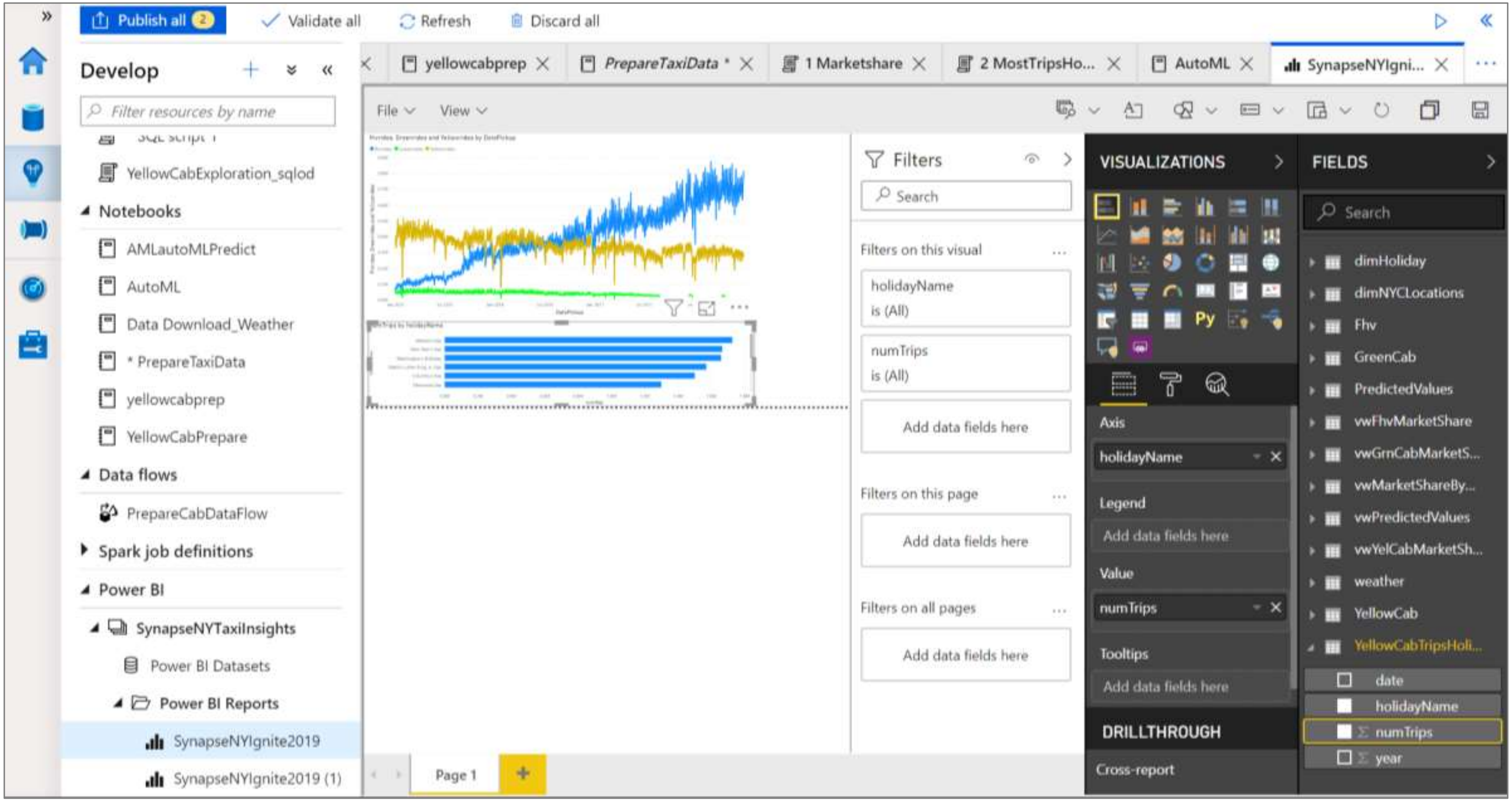
Develop Hub – Power BI

View published reports in Power BI workspace



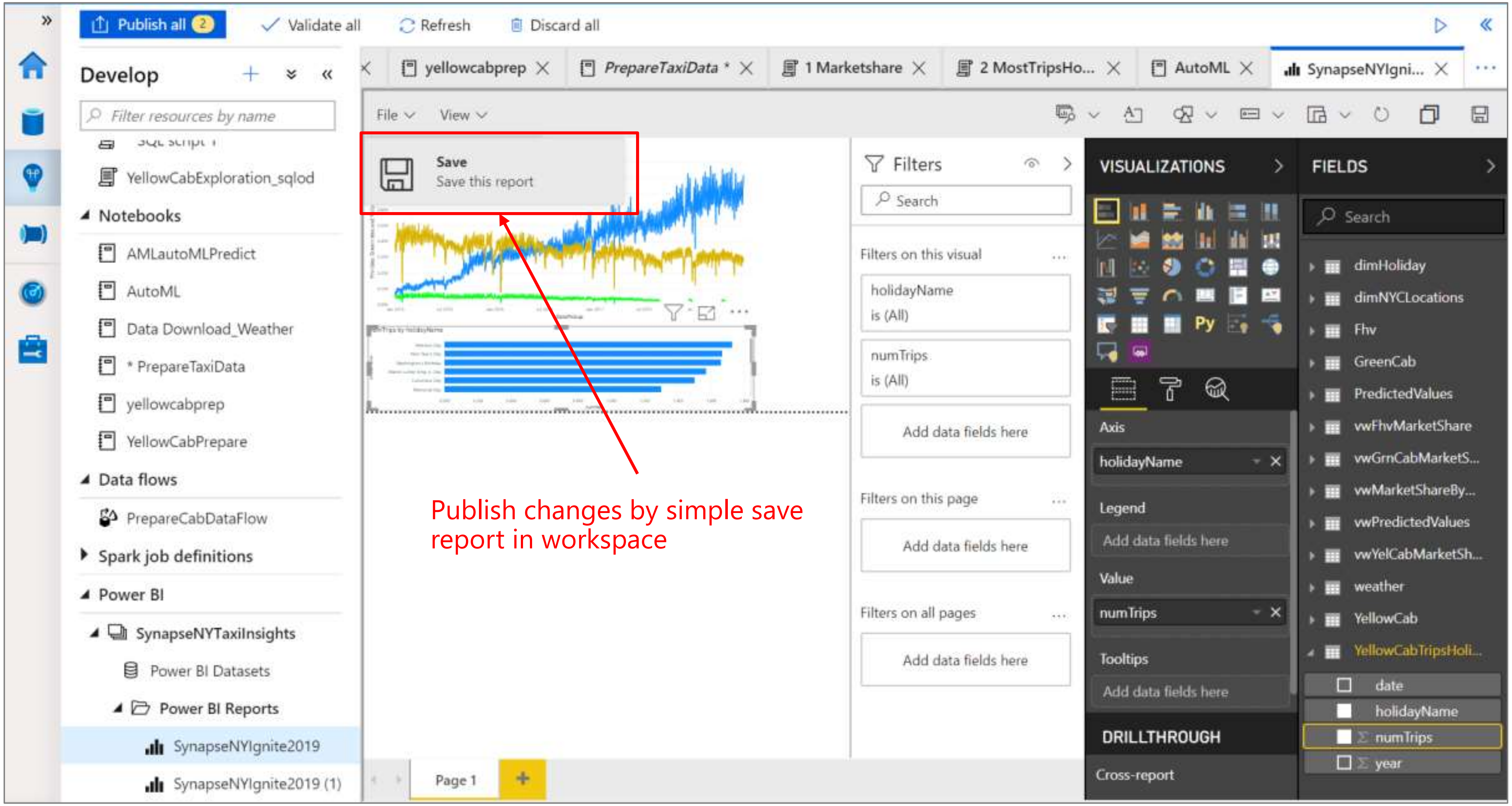
Develop Hub – Power BI

Edit reports in Synapse workspace

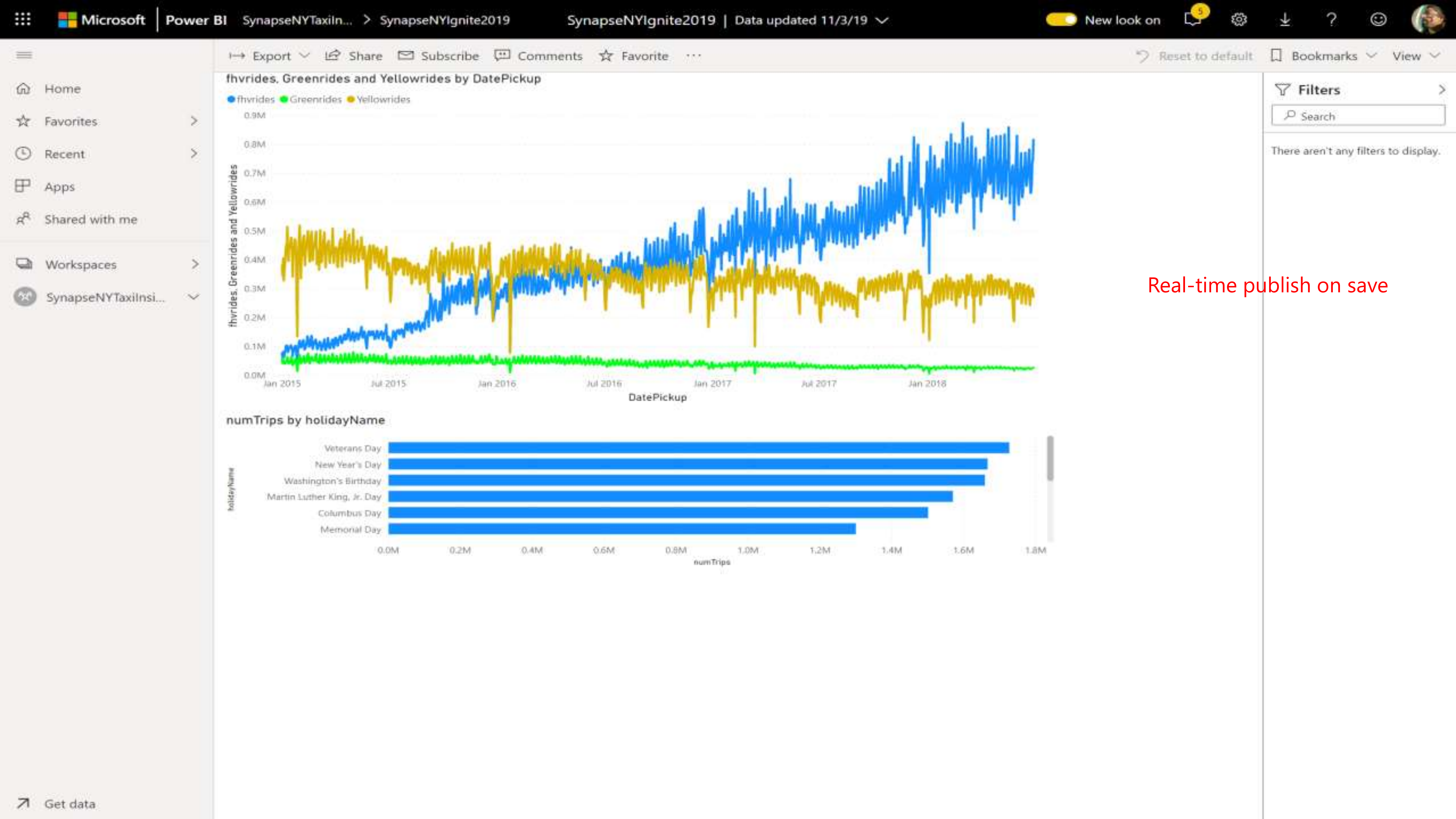


Develop Hub – Power BI

Publish edited reports in Synapse workspace to Power BI workspace



Publish changes by simple save report in workspace



There aren't any filters to display.

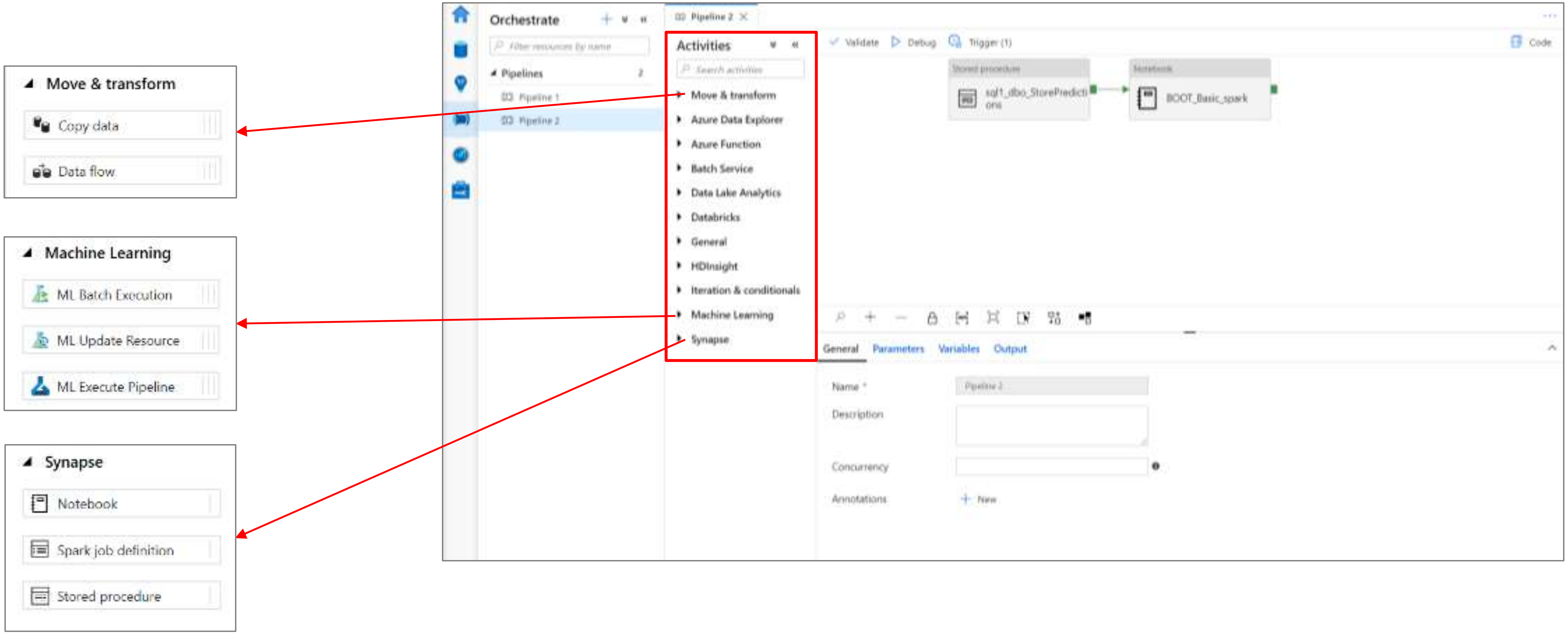
Real-time publish on save

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.

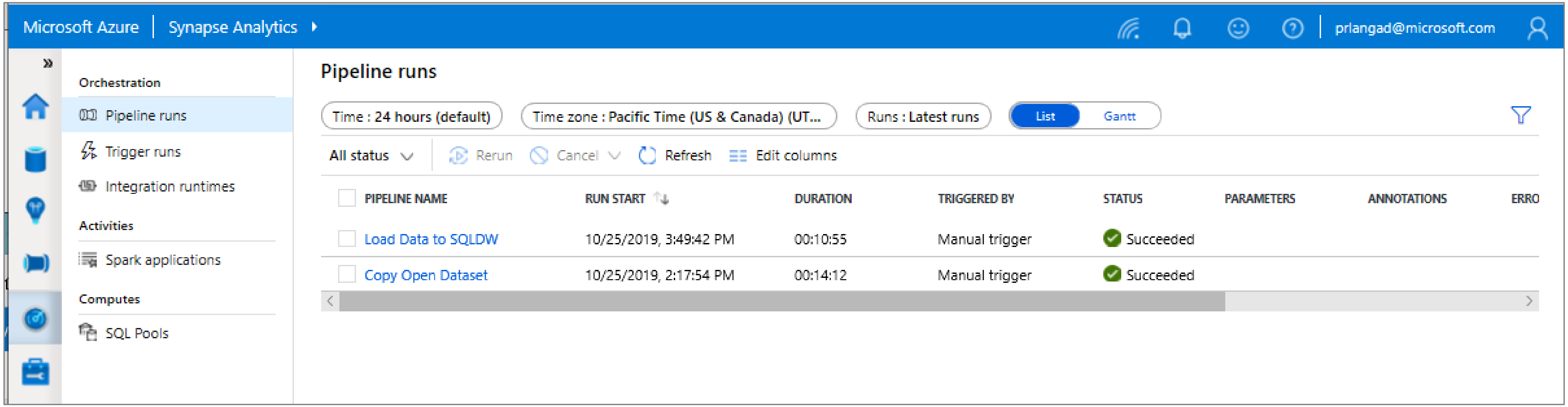


Synapse Studio
Monitor hub

Monitor Hub

Overview

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



PIPELINE NAME	RUN START	DURATION	TRIGGERED BY	STATUS	PARAMETERS	ANNOTATIONS	ERRO
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

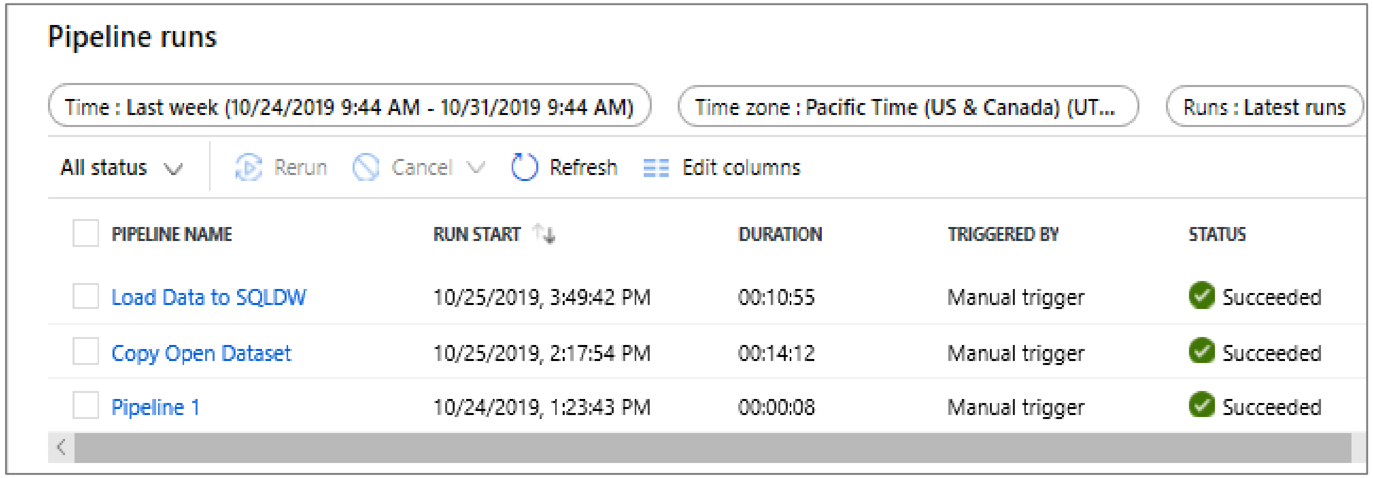
Monitor orchestration in the Synapse workspace for the progress and status of pipeline

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

□	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

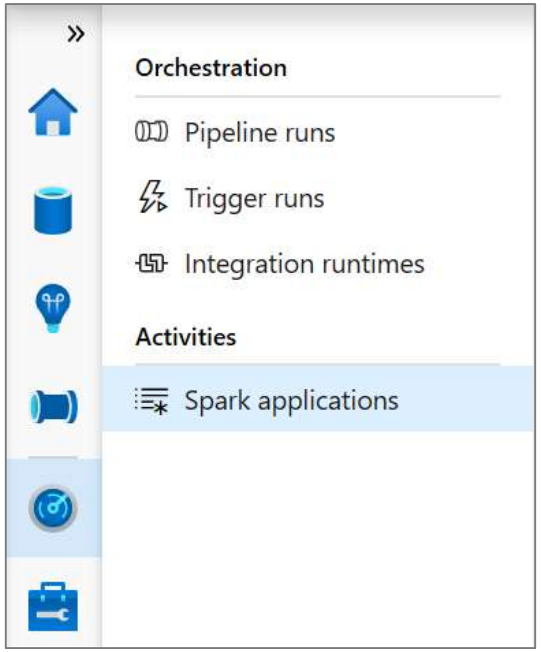
Overview

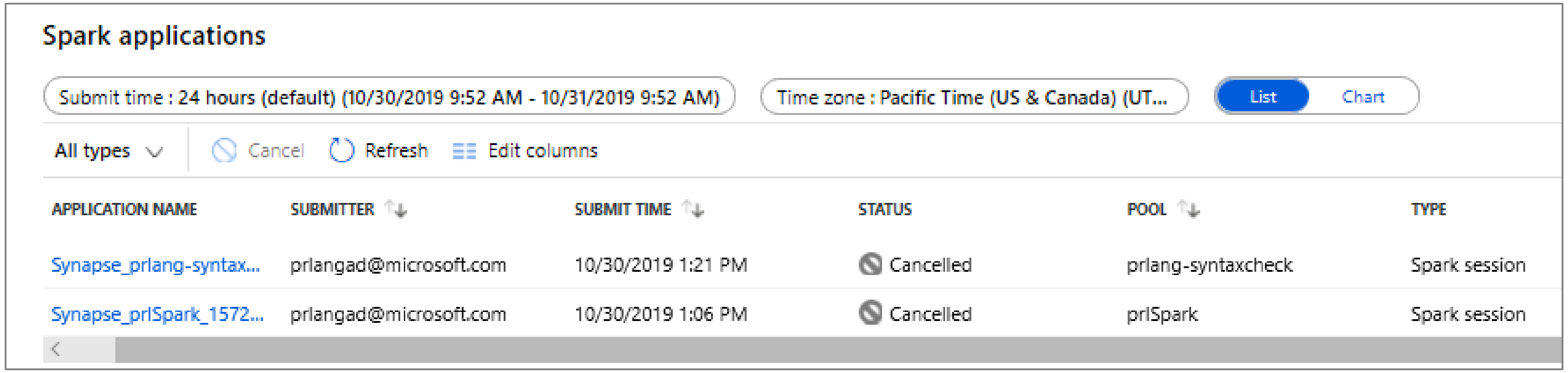
Monitor Spark pools, Spark applications for the progress and status of activities

Benefits

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

Track the usage of resources





Submit time : 24 hours (default) (10/30/2019 9:52 AM - 10/31/2019 9:52 AM)

Time zone : Pacific Time (US & Canada) (UT...

All types

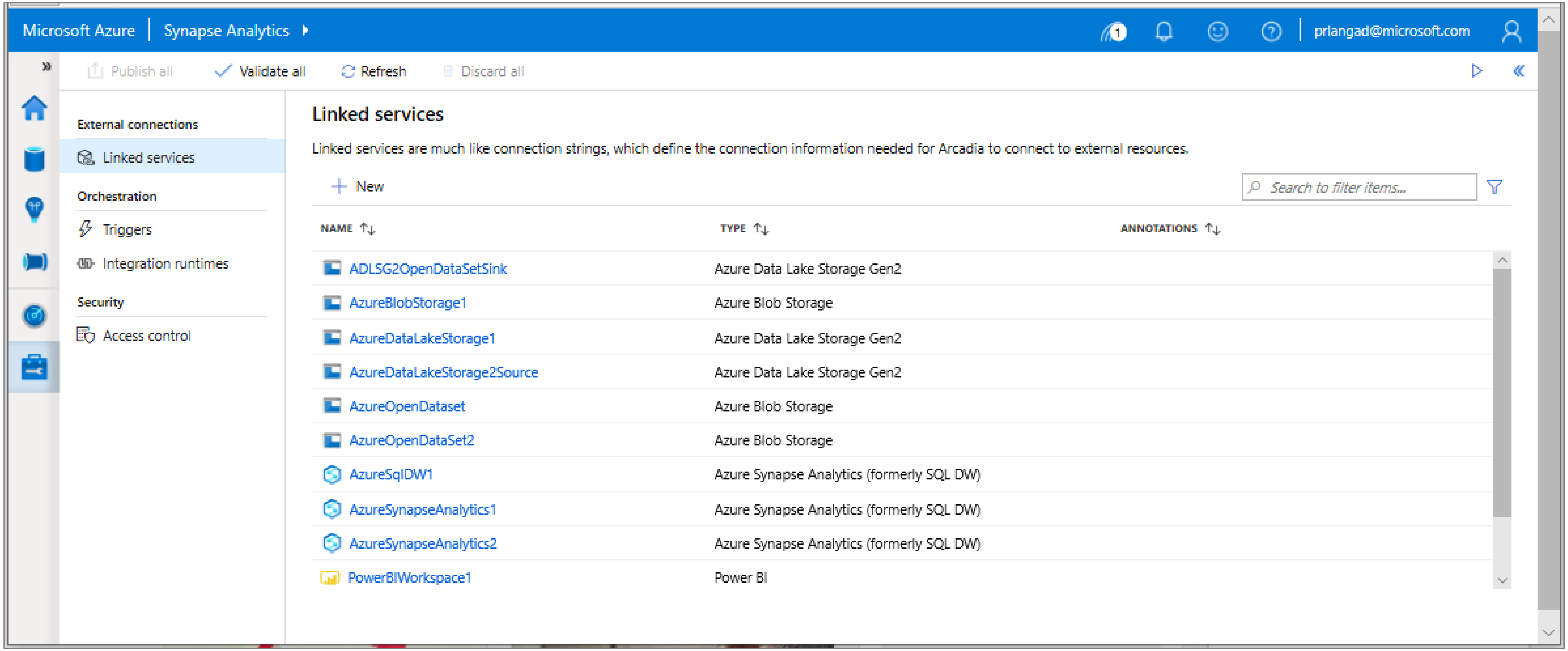
APPLICATION NAME	SUBMITTER	SUBMIT TIME	STATUS	POOL	TYPE
Synapse_prlang-syntax...	prlangad@microsoft.com	10/30/2019 1:21 PM	Cancelled	prlang-syntaxcheck	Spark session
Synapse_prlSpark_1572...	prlangad@microsoft.com	10/30/2019 1:06 PM	Cancelled	prlSpark	Spark session

Synapse Studio
Manage hub

Manage Hub

Overview

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



Linked services

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

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

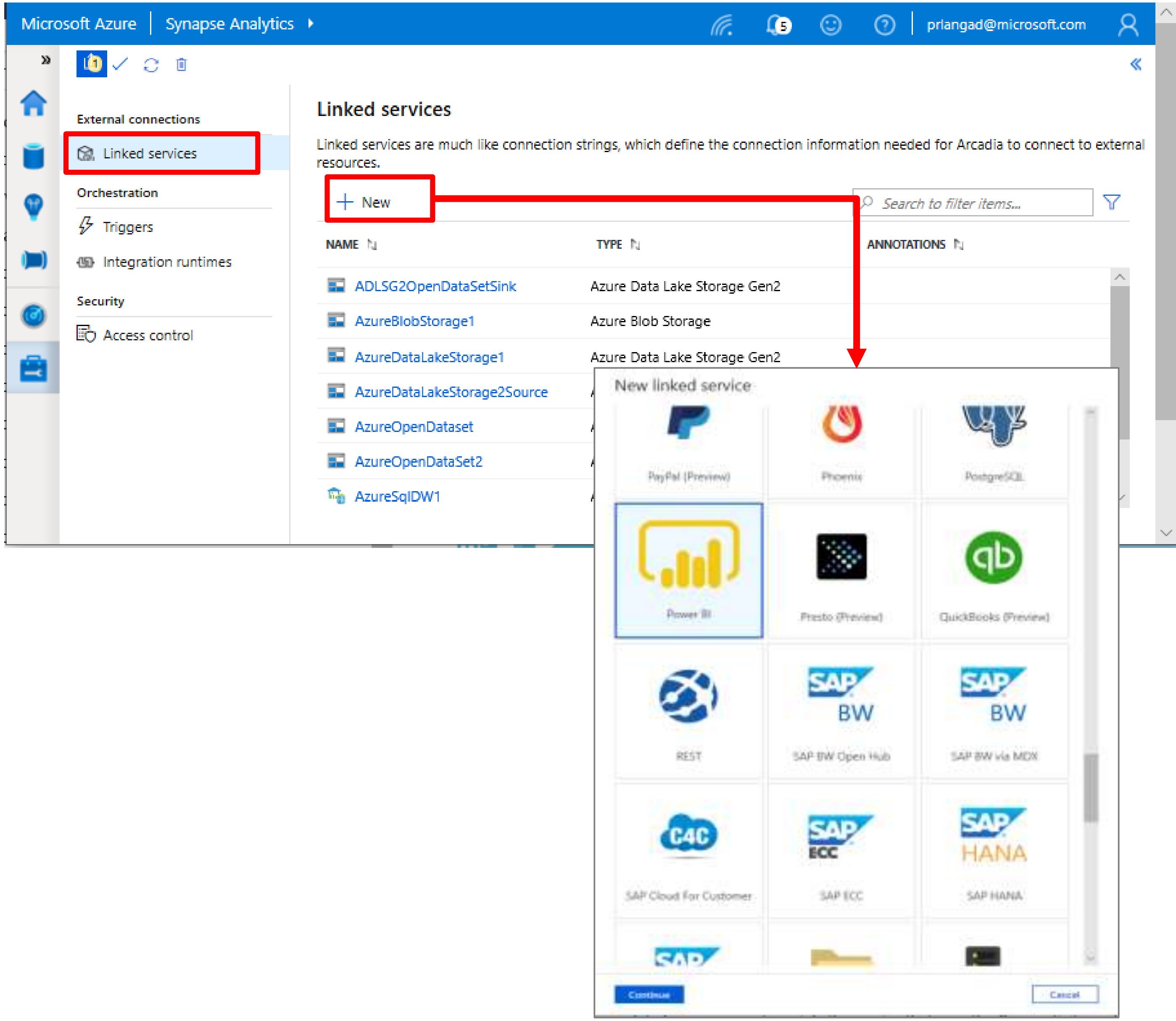
It defines the connection information needed to connect to external resources.

Benefits

Offers pre-build 90+ connectors

Easy cross platform data migration

Represents data store or compute resources



Linked services

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

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

Manage – Access Control

Overview

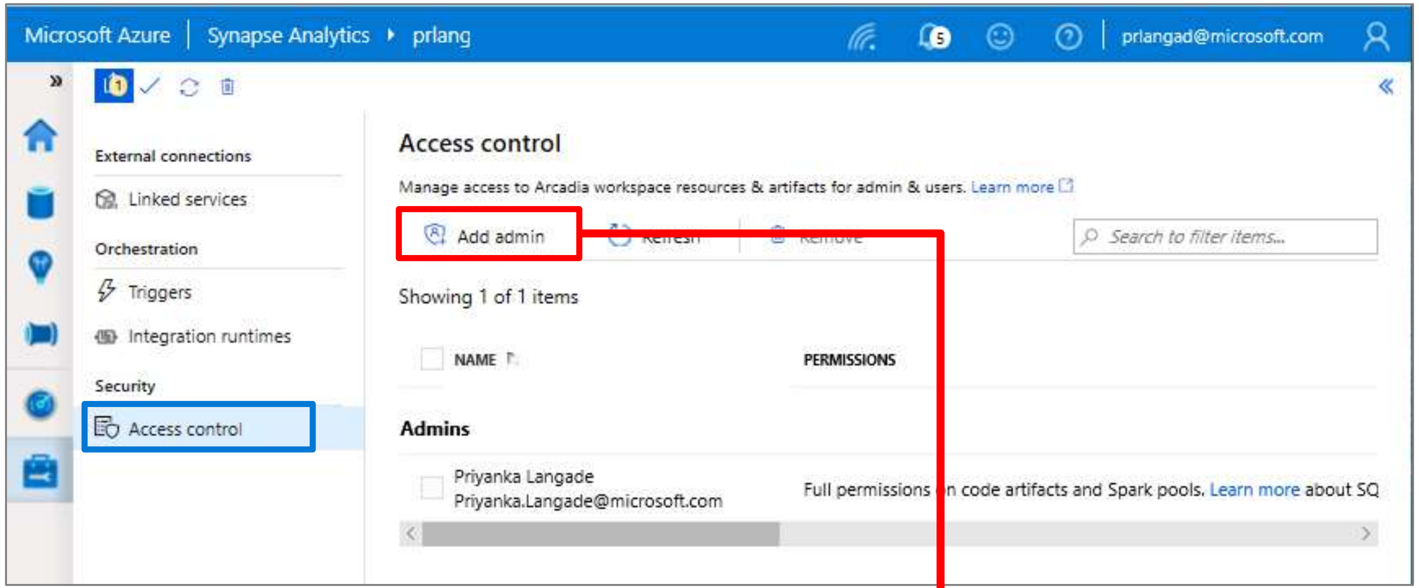
It provides access control management to workspace resources and artifacts for admin and users

Benefits

Share workspace with the team

Increases productivity

Manage permissions on code artifacts and Spark pools

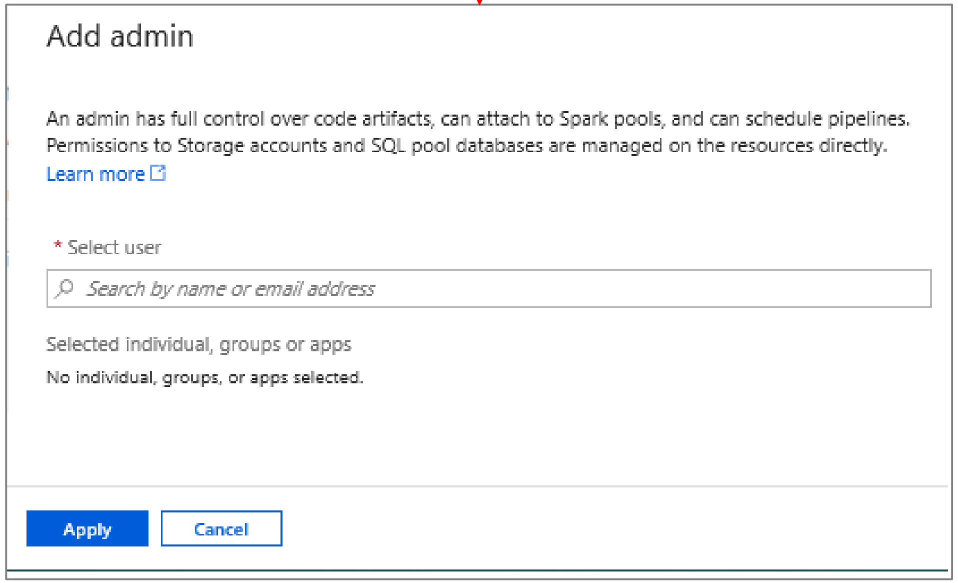


Access control

Manage access to Arcadia workspace resources & artifacts for admin & users. Learn more

Showing 1 of 1 items

NAME	PERMISSIONS
Admins	
Priyanka Langade Priyanka.Langade@microsoft.com	Full permissions on code artifacts and Spark pools. Learn more about SQ



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.

Manage – Triggers

Overview

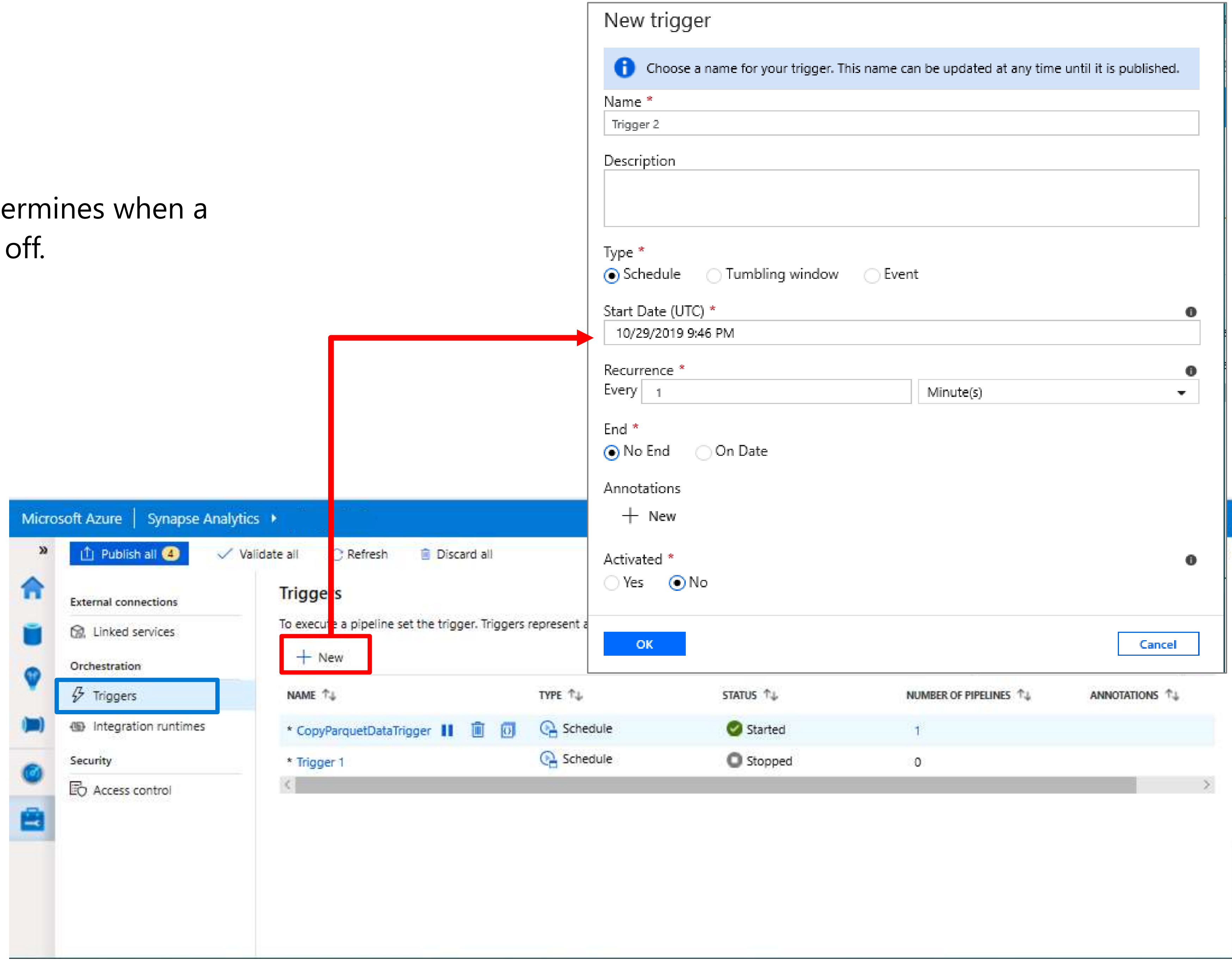
It defines a unit of processing that determines when a pipeline execution needs to be kicked off.

Benefits

Create and manage

- Schedule trigger
- Tumbling window trigger
- Event trigger

Control pipeline execution



Trigge s

To execu e a pipeline set the trigger. Triggers represent a

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

New trigger

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

Name * Trigger 2

Description

Type * ● Schedule ○ Tumbling window ○ Event

Start Date (UTC) * 10/29/2019 9:46 PM

Recurrence * Every 1 Minute(s)

End * ● No End ○ On Date

Annotations + New

Activated * ○ Yes ● No

OK Cancel

Manage – Integration runtimes

Overview

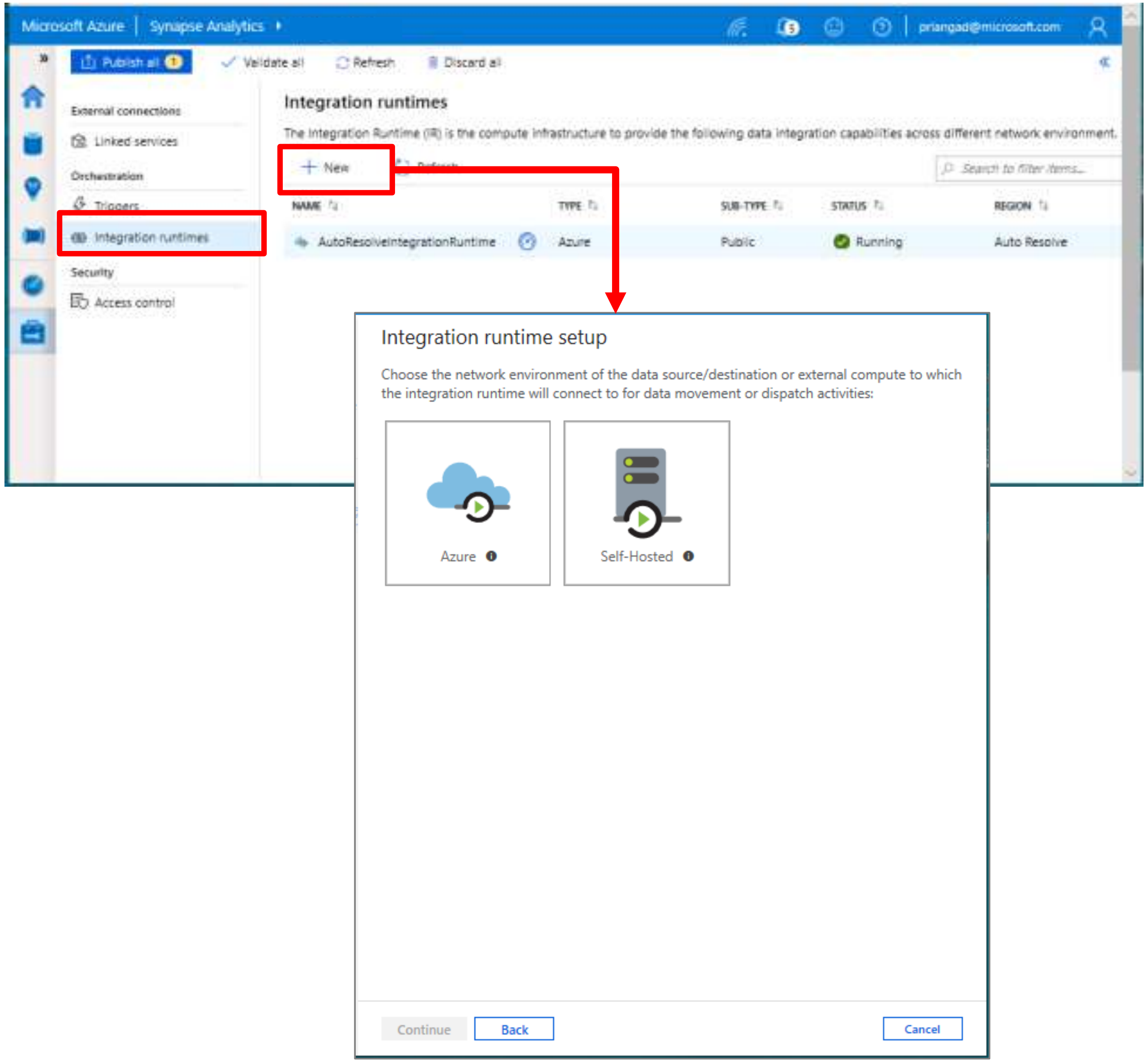
Integration runtimes are the compute infrastructure used by Pipelines to provide the data integration capabilities across different network environments. An integration runtime provides the bridge between the activity and linked services.

Benefits

Offers Azure Integration Runtime or Self-Hosted Integration Runtime

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

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



Integration runtimes

The Integration Runtime (IR) is the compute infrastructure to provide the following data integration capabilities across different network environment.

NAME	TYPE	SUB-TYPE	STATUS	REGION
AutoResolveIntegrationRuntime	Azure	Public	Running	Auto Resolve

Integration runtime setup

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

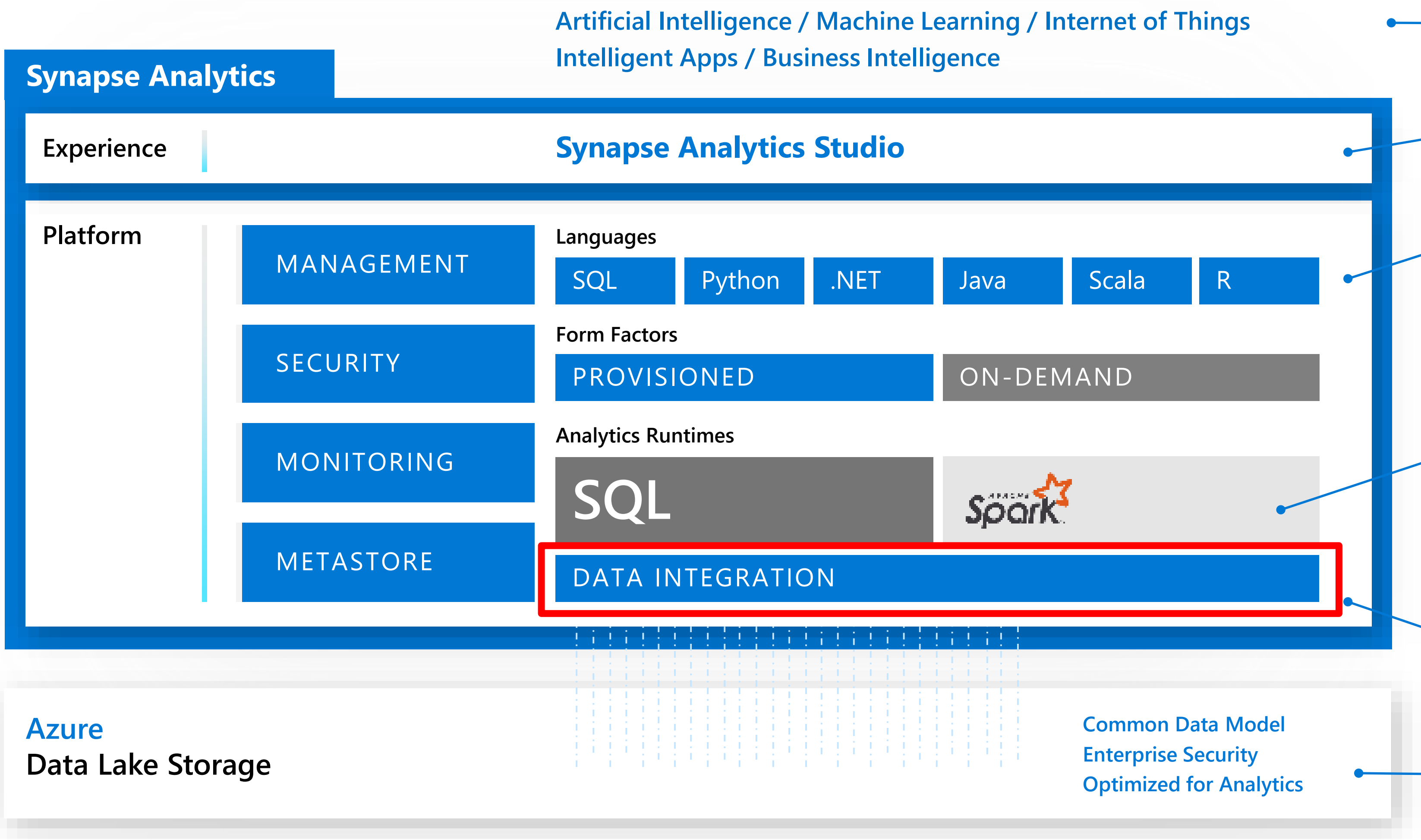
Microsoft

Azure Synapse Analytics

Data Integration

Azure Synapse Analytics

Integrated data platform for BI, AI and continuous intelligence



Designed for analytics **workloads at any scale**

SaaS **developer experiences** for code free and code first

Multiple **languages** suited to different analytics workloads

Integrated analytics runtimes available provisioned and serverless on-demand

SQL Analytics offering T-SQL for batch, streaming and interactive processing

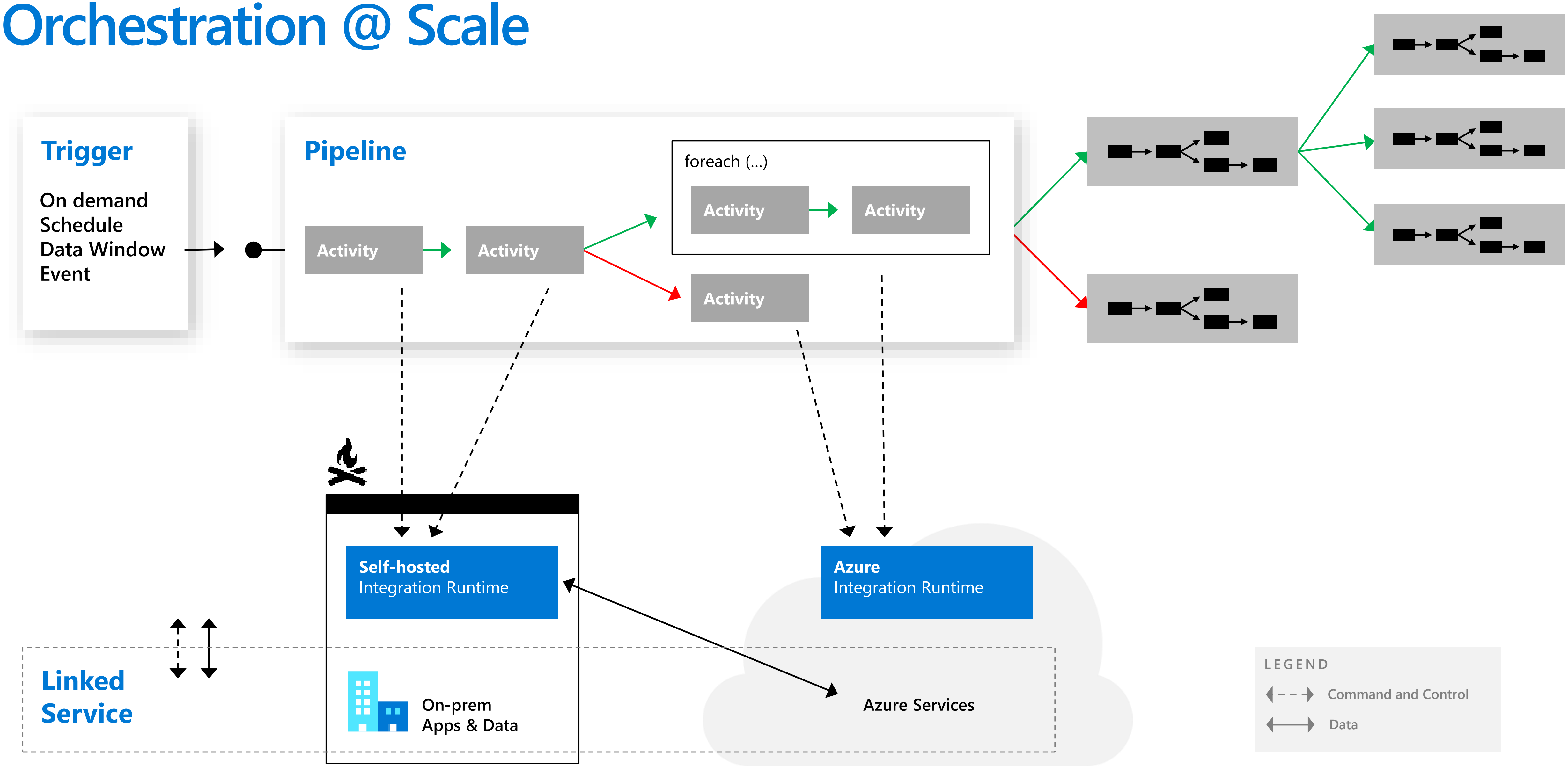
Spark for big data processing with Python, Scala, R and .NET

Integrated **platform services** for, management, security, monitoring, and metastore

Data **lake integrated** and Common Data Model aware

Data Integration = Separate version Azure Data Factory (ADF). Will have 1-click migration

Orchestration @ Scale



Data Movement

Scalable

per job elasticity

Up to 4 GB/s

Simple

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

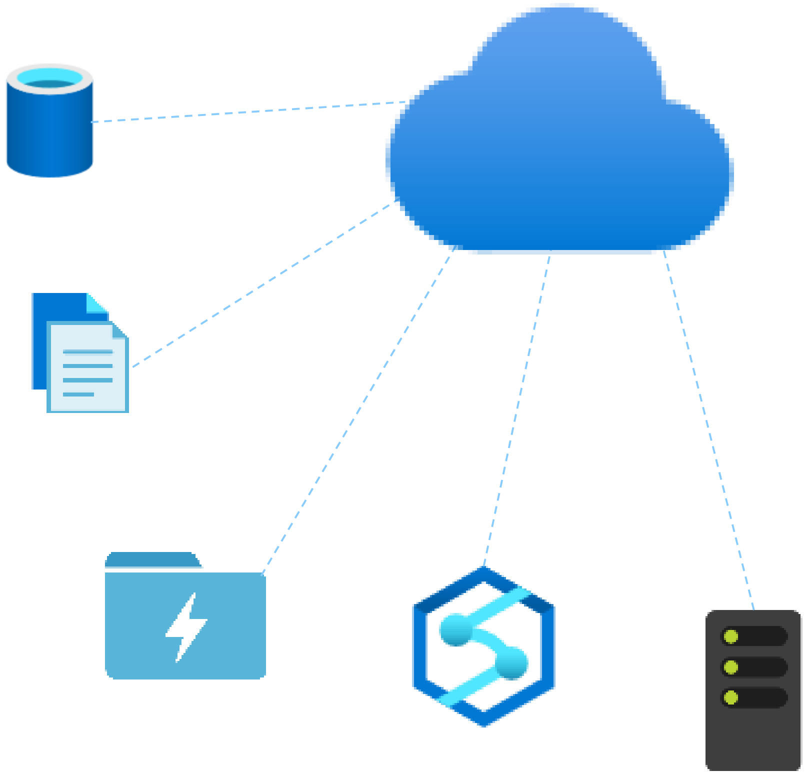
Serverless, no infrastructure to manage

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

Azure (15)	Database & DW (26)		File Storage (6)	File Formats(6)	NoSQL (3)	Services and App (28)		Generic (4)
Blob storage	Amazon Redshift	Oracle	Amazon S3	AVRO	Cassandra	Amazon MWS	Oracle Service Cloud	Generic HTTP
Cosmos DB - SQL API	DB2	Phoenix	File system	Binary	Couchbase	CDS for Apps	PayPal	Generic OData
Cosmos DB - MongoDB API	Drill	PostgreSQL	FTP	Delimited Text	MongoDB	Concur	QuickBooks	Generic ODBC
Data Explorer	Google BigQuery	Presto	Google Cloud Storage	JSON		Dynamics 365	Salesforce	Generic REST
Data Lake Storage Gen1	Greenplum	SAP BW Open Hub	HDFS	ORC		Dynamics AX	SF Service Cloud	
Data Lake Storage Gen2	HBase	SAP BW via MDX	SFTP	Parquet		Dynamics CRM	SF Marketing Cloud	
Database for MariaDB	Hive	SAP HANA				Google AdWords	SAP C4C	
Database for MySQL	Apache Impala	SAP table				HubSpot	SAP ECC	
Database for PostgreSQL	Informix	Spark				Jira	ServiceNow	
File Storage	MariaDB	SQL Server				Magento	Shopify	
SQL Database	Microsoft Access	Sybase				Marketo	Square	
SQL Database MI	MySQL	Teradata				Office 365	Web table	
SQL Data Warehouse	Netezza	Vertica				Oracle Eloqua	Xero	
Search index						Oracle Responsys	Zoho	
Table storage								

Pipelines

Overview

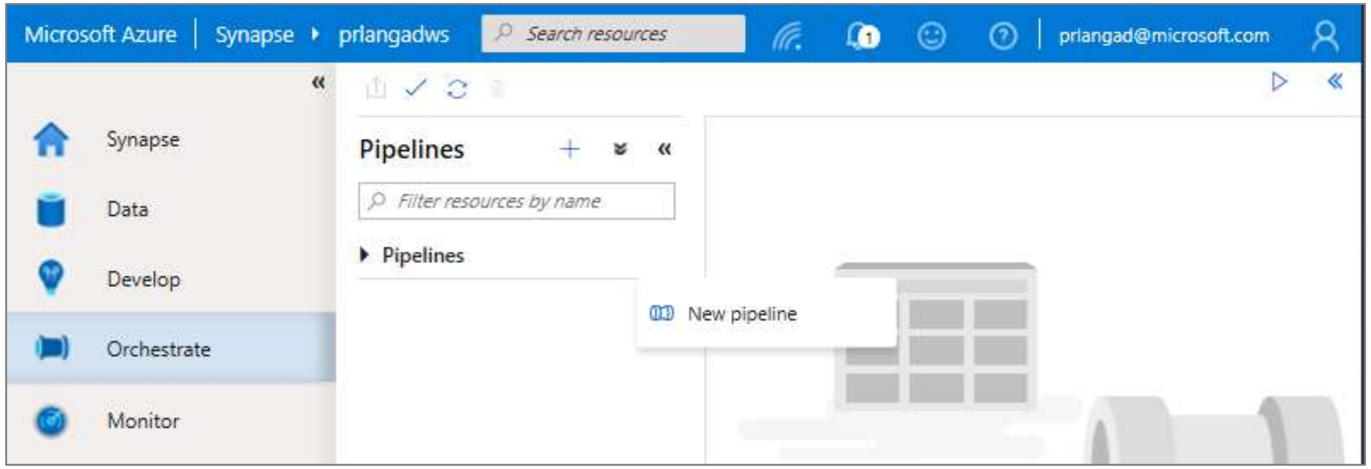
It provides ability to load data from storage account to desired linked service. Load data by manual execution of pipeline or by orchestration

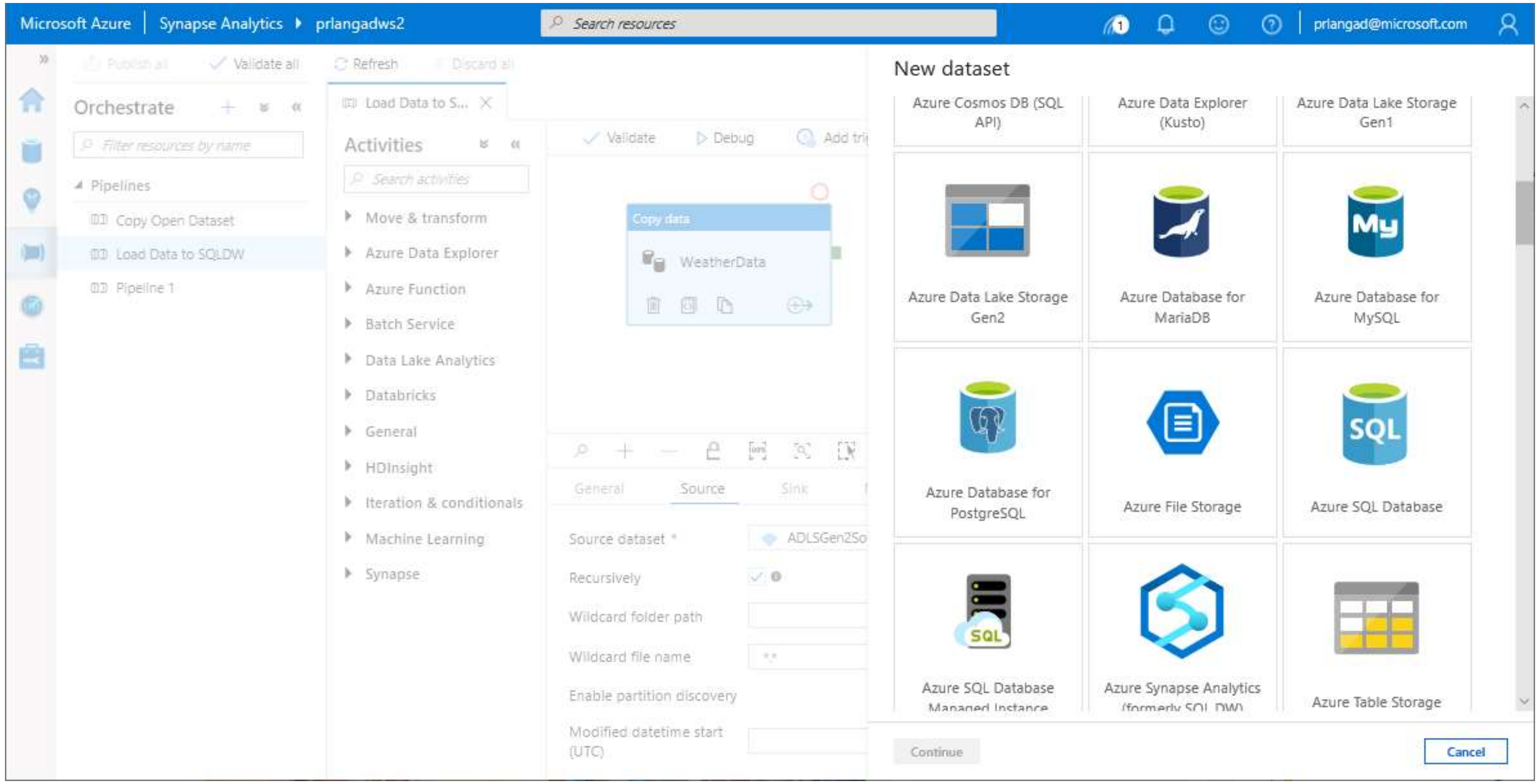
Benefits

Supports common loading patterns

Fully parallel loading into data lake or SQL tables

Graphical development experience

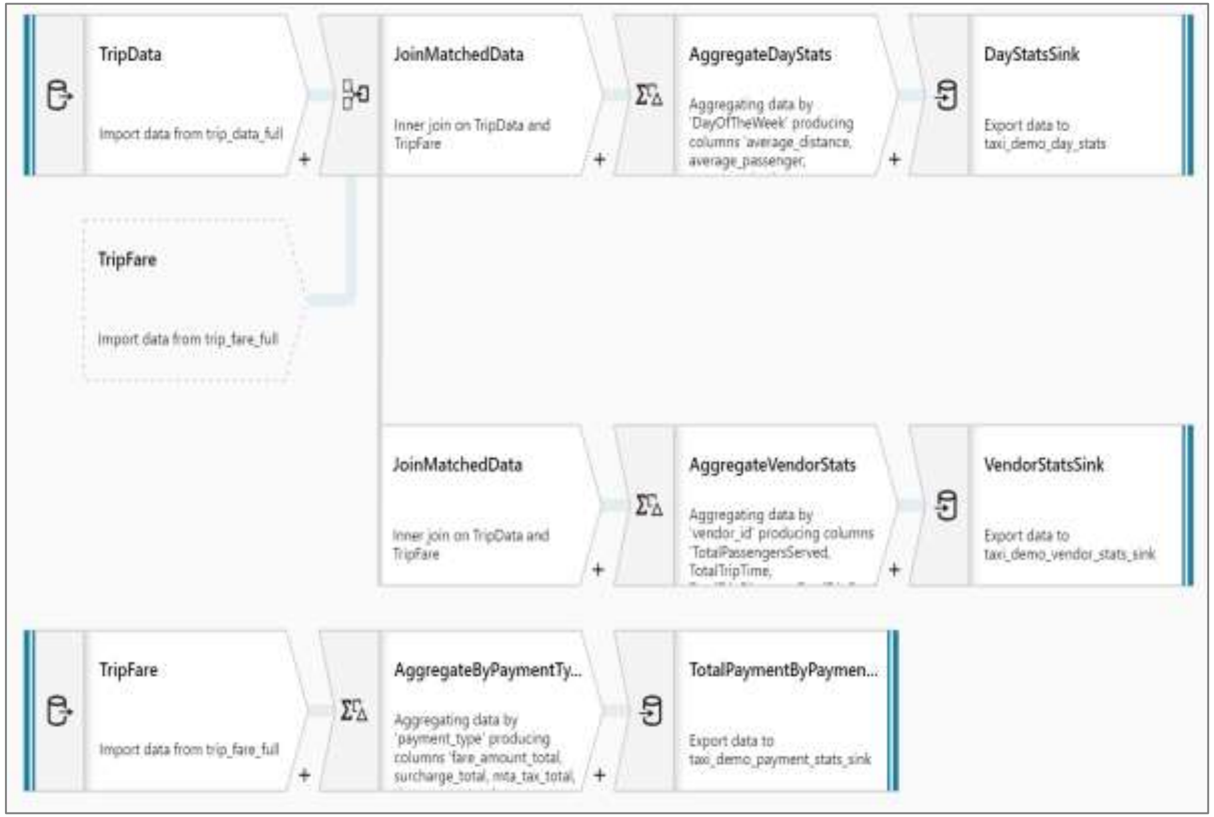




Prep & Transform Data

Mapping Dataflow

Code free data transformation @scale



TripData

Import data from trip_data_full

JoinMatchedData

Inner join on TripData and TripFare

AggregateDayStats

Aggregating data by 'DayOfTheWeek' producing columns 'average_distance, average_passenger,

DayStatsSink

Export data to taxi_demo_day_stats

TripFare

Import data from trip_fare_full

JoinMatchedData

Inner join on TripData and TripFare

AggregateVendorStats

Aggregating data by 'vendor_id' producing columns 'TotalPassengersServed, TotalTripTime,

VendorStatsSink

Export data to taxi_demo_vendor_stats_sink

TripFare

Import data from trip_fare_full

AggregateByPaymentTy...

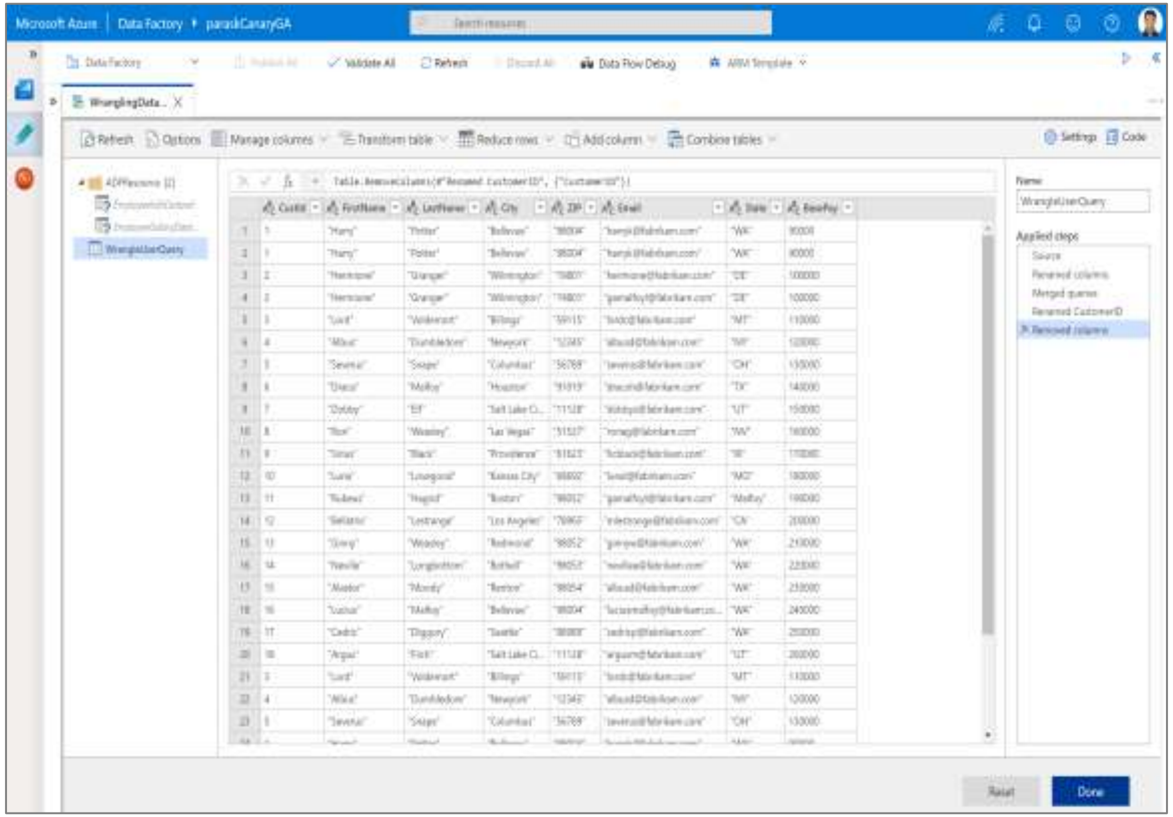
Aggregating data by 'payment_type' producing columns 'fare_amount_total, surcharge_total, mta_tax_total,

TotalPaymentByPaymen...

Export data to taxi_demo_payment_stats_sink

Wrangling Dataflow

Code free data preparation @scale



Triggers

Overview

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

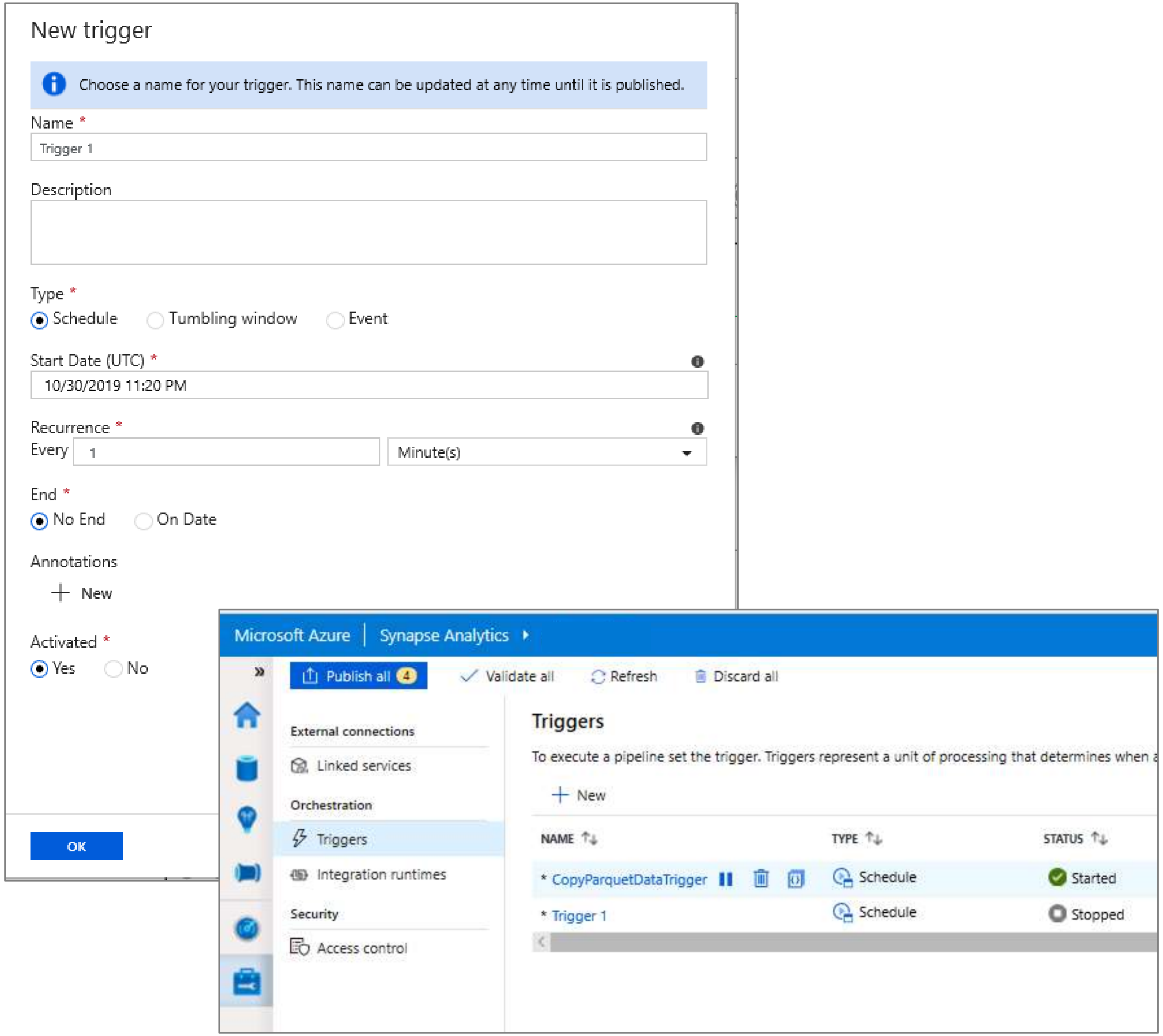
Data Integration offers 3 trigger types as –

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.



New trigger

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

Name *
Trigger 1

Description

Type *
● Schedule ○ Tumbling window ○ Event

Start Date (UTC) *
10/30/2019 11:20 PM

Recurrence *
Every 1 Minute(s)

End *
● No End ○ On Date

Annotations
+ New

Activated *
● Yes ○ No

OK

Triggers

To execute a pipeline set the trigger. Triggers represent a unit of processing that determines when a

+ New

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

Manage – Linked Services

Overview

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

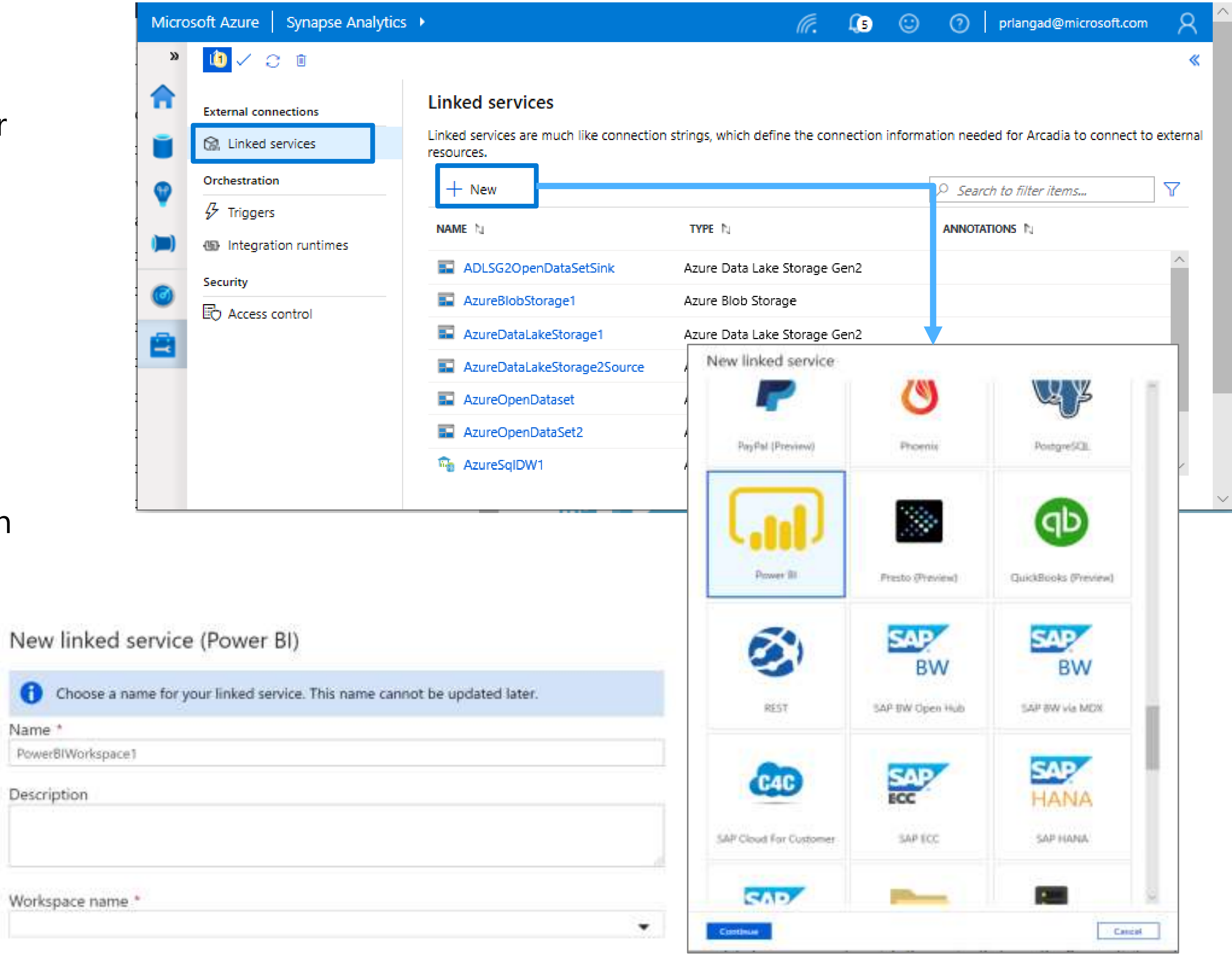
Benefits

Offers pre-build 85+ connectors

Easy cross platform data migration

Represents data store or compute resources

NOTE: Linked Services are all for Data Integration except for Power BI (eventually ADC, Databricks)



Linked services

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

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

New linked service

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

New linked service (Power BI)

Choose a name for your linked service. This name cannot be updated later.

Name *
PowerBIWorkspace1

Description

Workspace name *

Manage – Integration runtimes

Overview

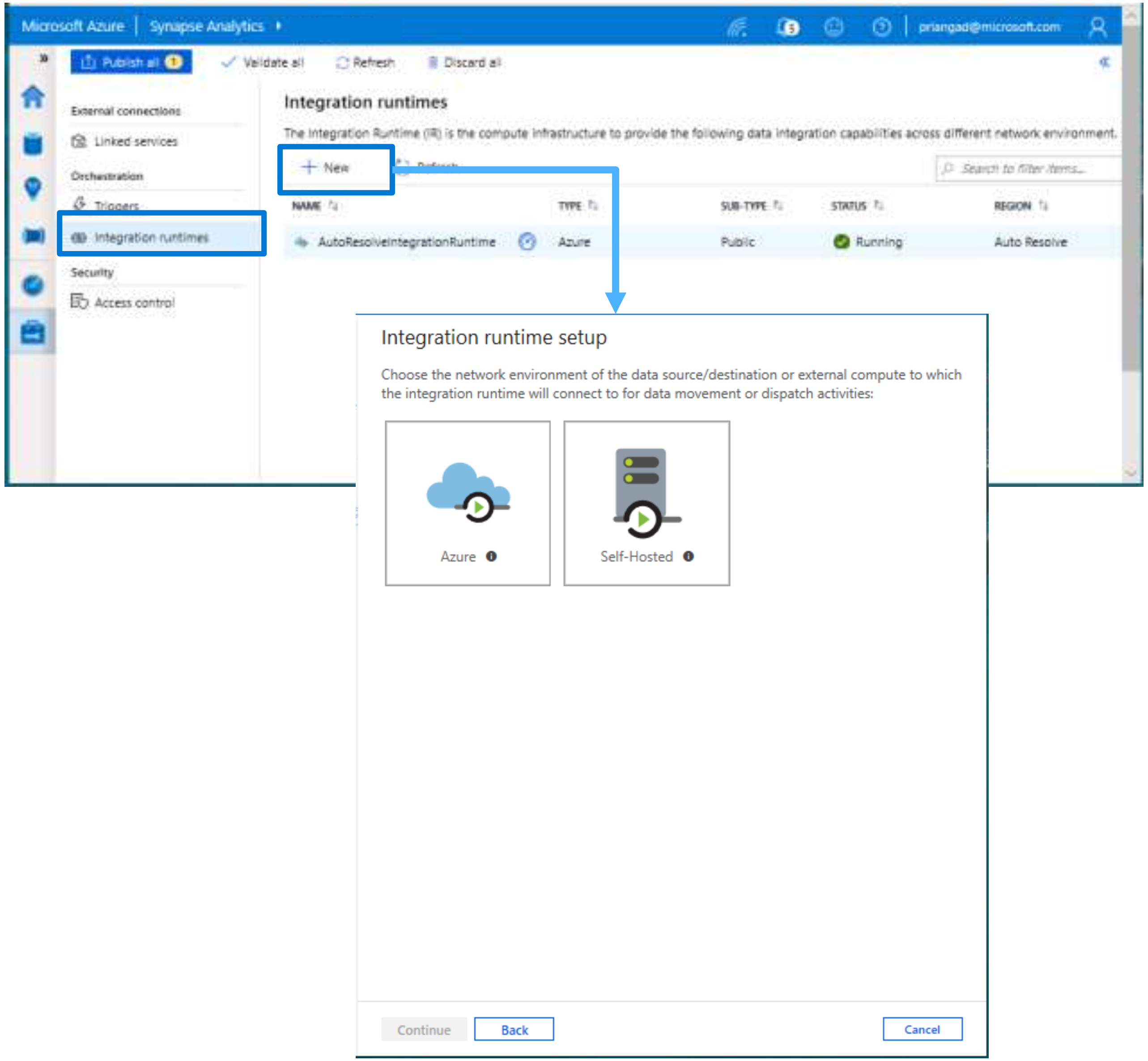
It is the compute infrastructure used by Pipelines to provide the data integration capabilities across different network environments. An integration runtime provides the bridge between the activity and linked Services.

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



Integration runtimes

The Integration Runtime (IR) is the compute infrastructure to provide the following data integration capabilities across different network environment.

NAME	TYPE	SUB-TYPE	STATUS	REGION
AutoResolveIntegrationRuntime	Azure	Public	Running	Auto Resolve

Integration runtime setup

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

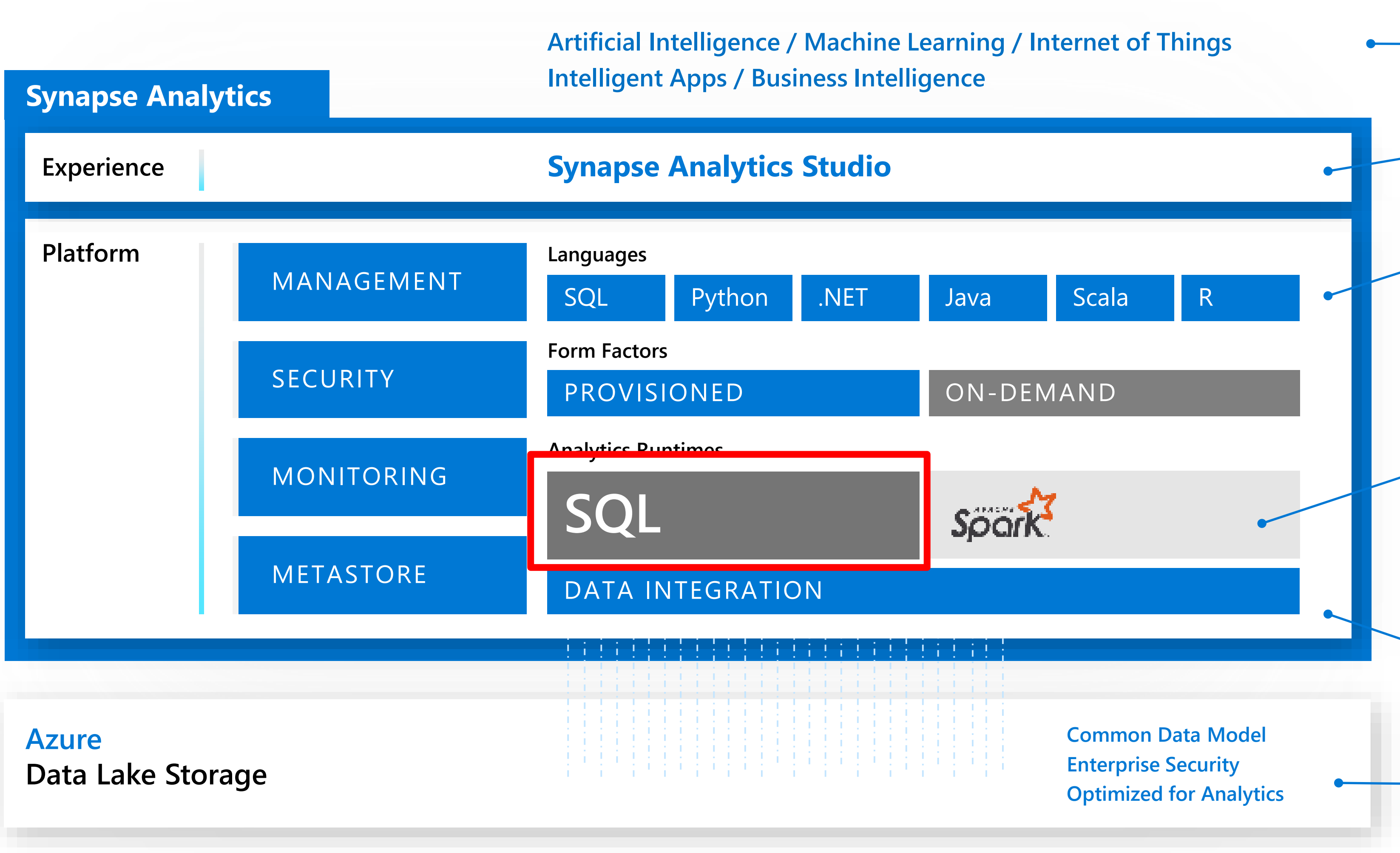
Microsoft

Azure Synapse Analytics

SQL Analytics

Azure Synapse Analytics

Integrated data platform for BI, AI and continuous intelligence



Designed for analytics **workloads at any scale**

SaaS **developer experiences** for code free and code first

Multiple **languages** suited to different analytics workloads

Integrated analytics runtimes available provisioned and serverless on-demand

SQL Analytics offering T-SQL for batch, streaming and interactive processing

Spark for big data processing with Python, Scala, R and .NET

Integrated **platform services** for, management, security, monitoring, and metastore

Data **lake integrated** and Common Data Model aware

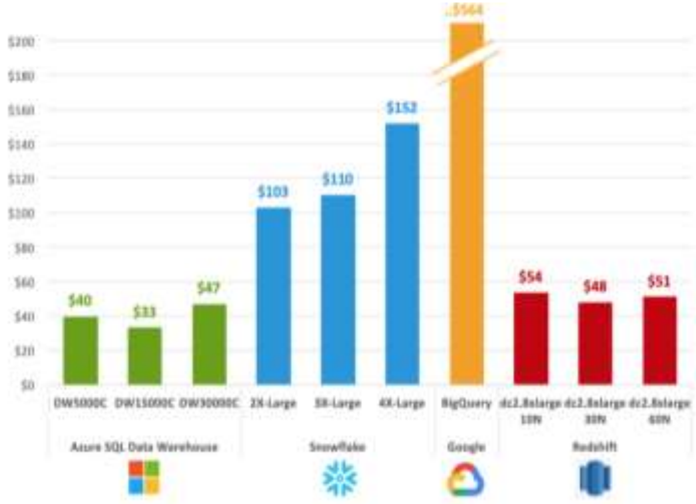
Platform: Performance

Overview

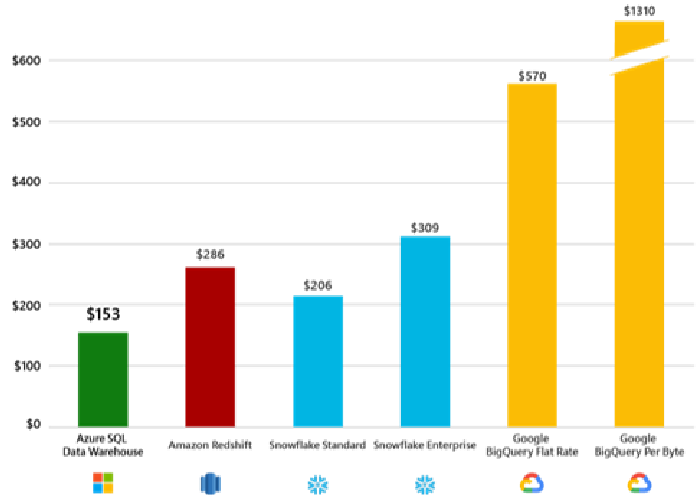
SQL Data Warehouse's industry leading price-performance comes from leveraging the Azure ecosystem and core SQL Server engine improvements to produce massive gains in performance.

These benefits require no customer configuration and are provided out-of-the-box for every data warehouse

- **Gen2 adaptive caching** – using non-volatile memory solid-state drives (NVMe) to increase the I/O bandwidth available to queries.
- **Azure FPGA-accelerated networking enhancements** – to move data at rates of up to 1GB/sec per node to improve queries
- **Instant data movement** – leverages multi-core parallelism in underlying SQL Servers to move data efficiently between compute nodes.
- **Query Optimization** – ongoing investments in distributed query optimization

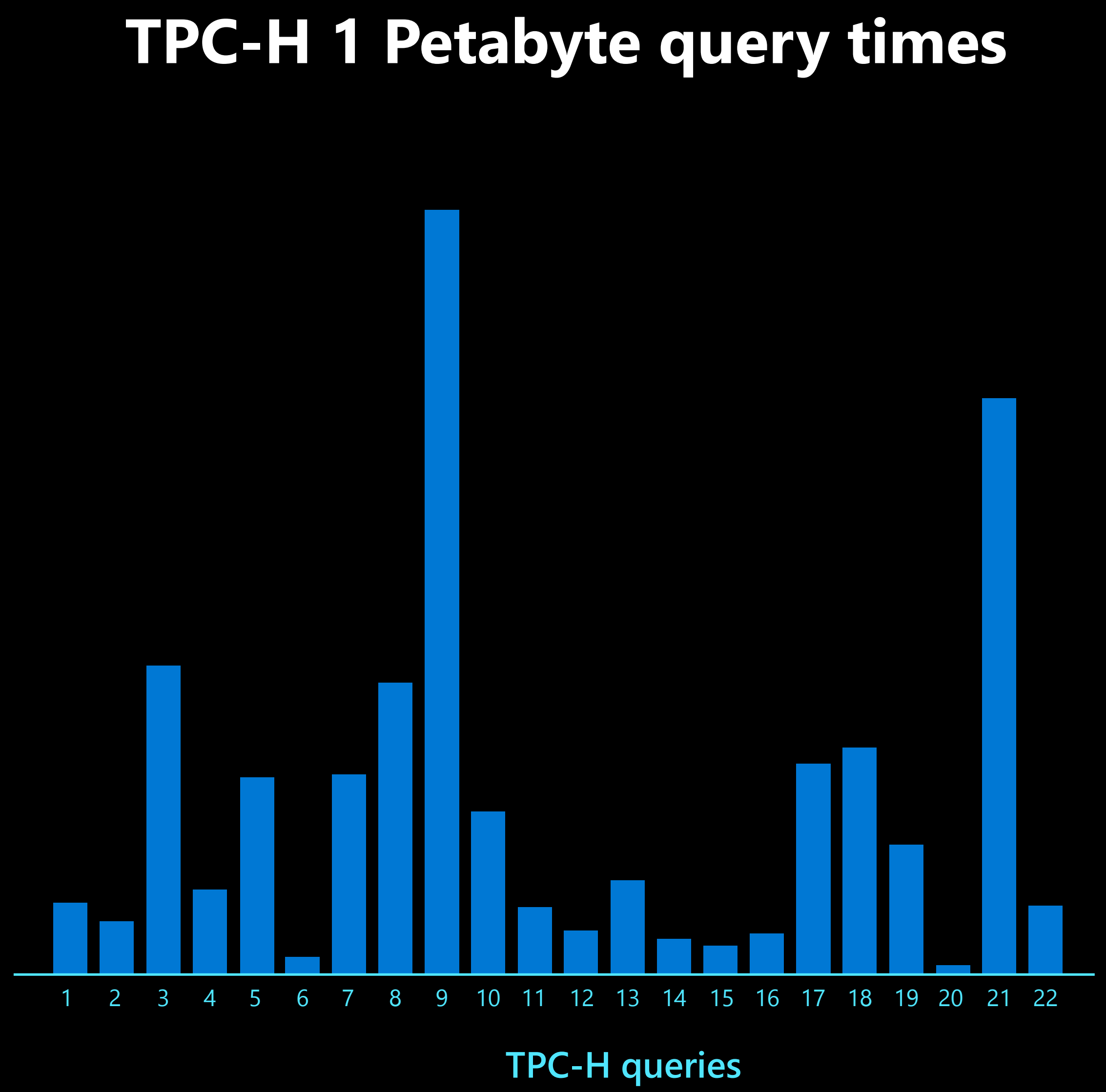


TPC-H 30TB Cloud DW Benchmark



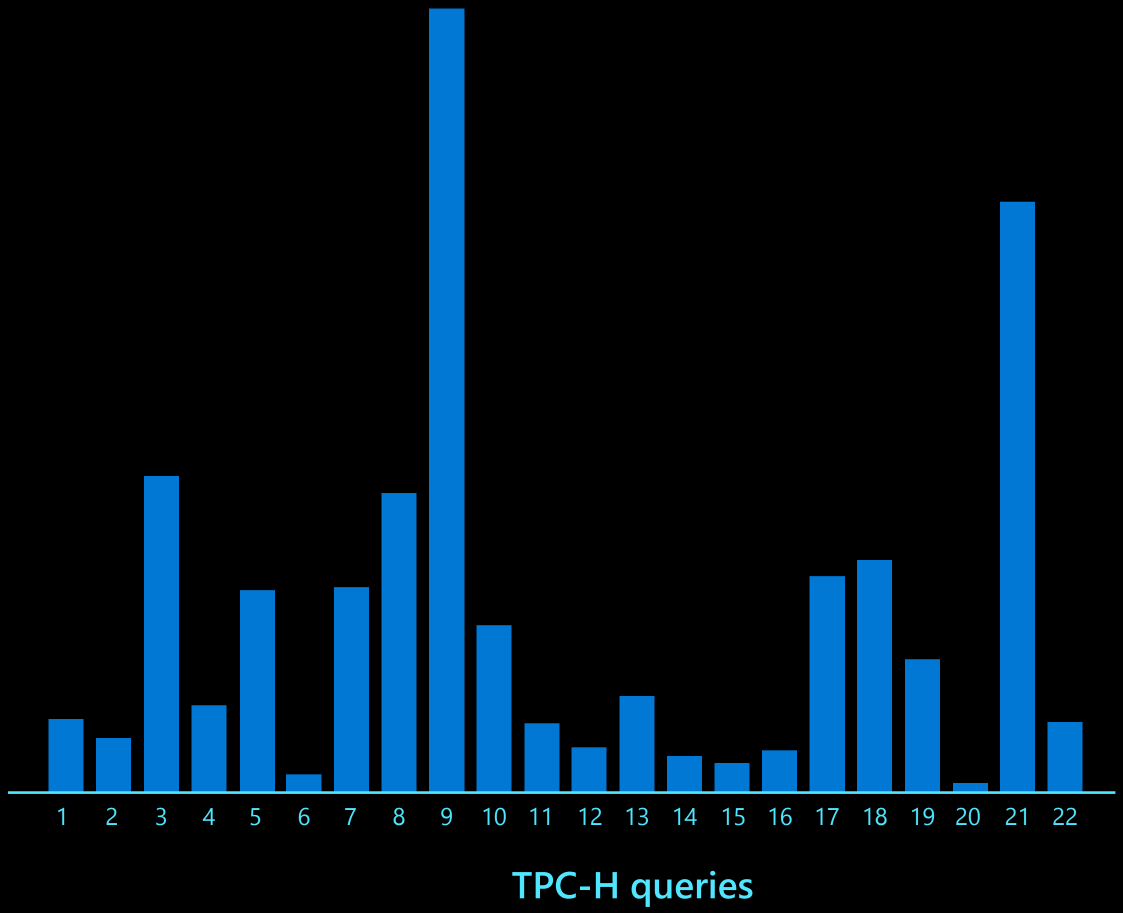
TPC-DS 30TB Cloud DW Benchmark

The first and only analytics system to have run all TPC-H queries at petabyte-scale



Azure Synapse is the first and only analytics system to have run all TPC-H queries at 1 petabyte-scale

TPC-H 1 Petabyte Query Execution



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

```sql
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)

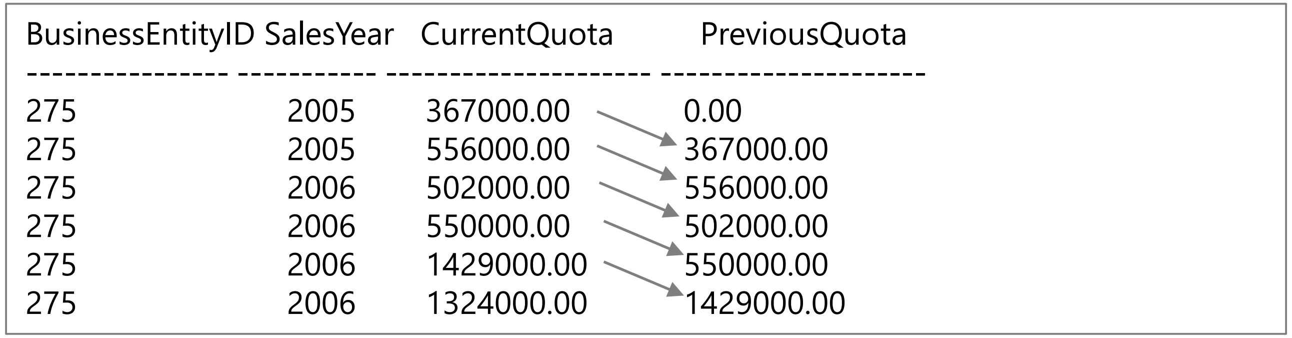
Analytical functions

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

```sql
-- 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
```

```sql
--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

```sql
-- 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

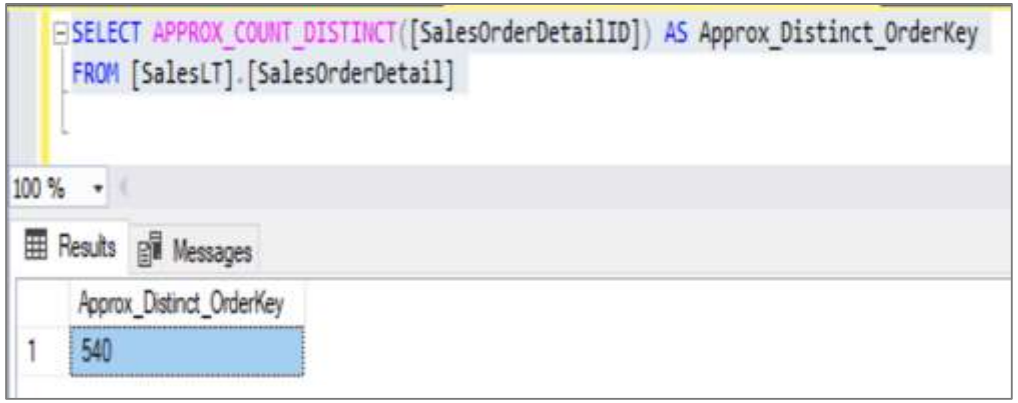
Returns the approximate number of unique non-null values in a group.

Use Case: Approximating web usage trend behavior

```sql
-- 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;
```

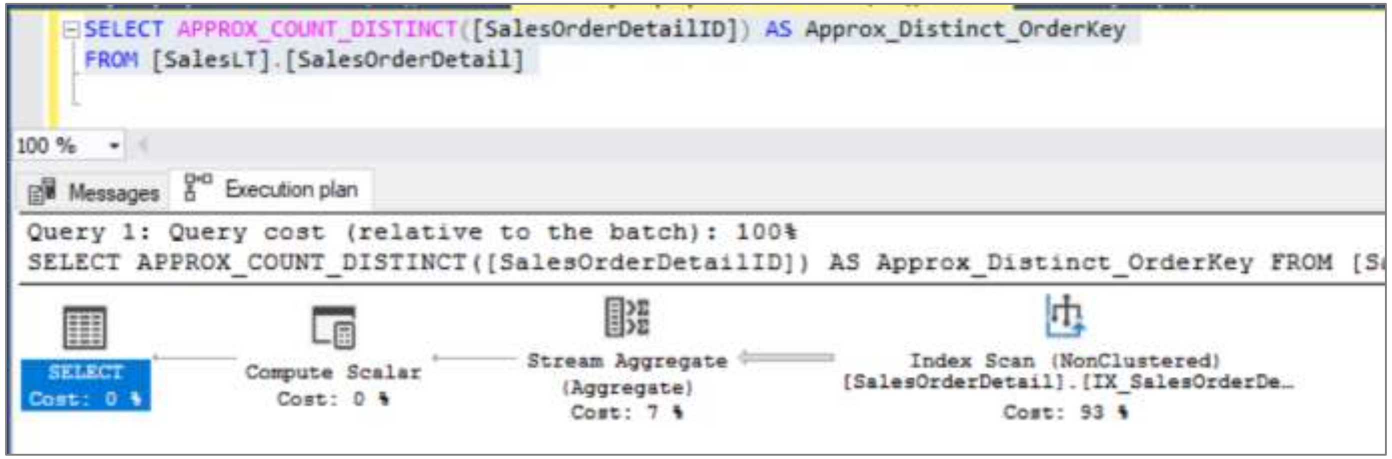
Approximate execution

APPROX_COUNT_DISTINCT



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

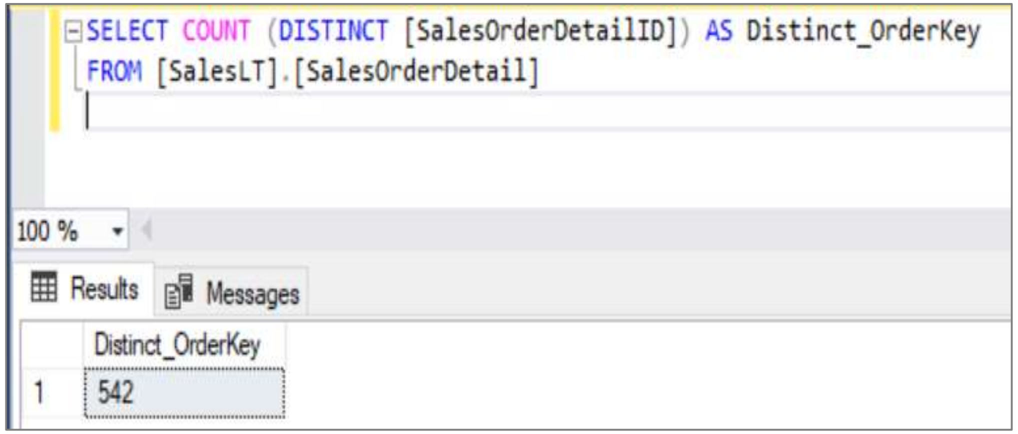
	Approx_Distinct_OrderKey
1	540



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

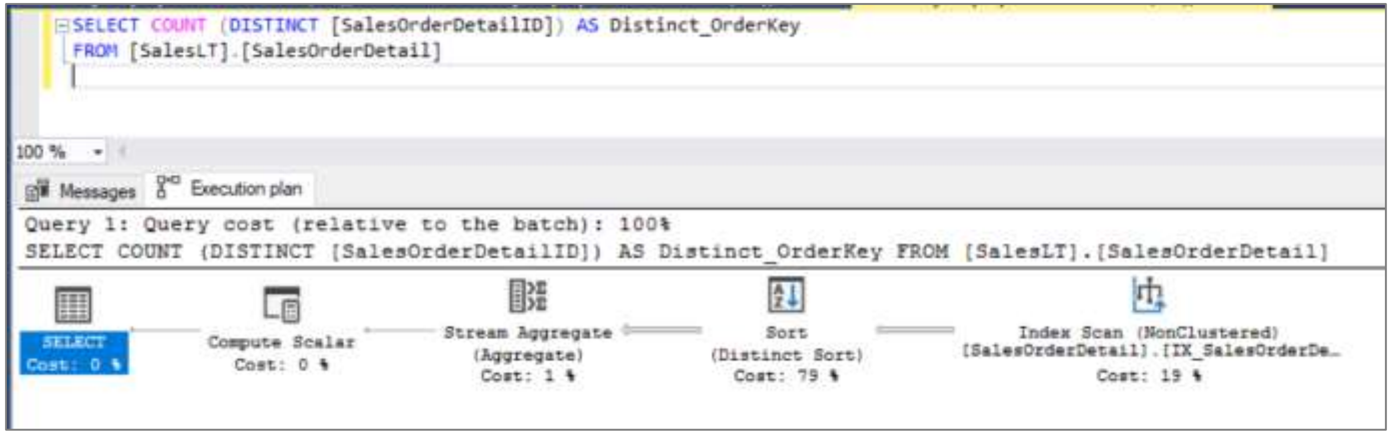
```
Query 1: Query cost (relative to the batch): 100%
SELECT APPROX_COUNT_DISTINCT([SalesOrderDetailID]) AS Approx_Distinct_OrderKey FROM [S
```

COUNT DISTINCT



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

	Distinct_OrderKey
1	542



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

```
Query 1: Query cost (relative to the batch): 100%
SELECT COUNT (DISTINCT [SalesOrderDetailID]) AS Distinct_OrderKey FROM [SalesLT].[SalesOrderDetail]
```

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

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, () );
```

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

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.

Isolation level

- READ COMMITTED
- REPEATABLE READ
- SERIALIZABLE
- READ UNCOMMITTED

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.

```
ALTER DATABASE MyDatabase
SET ALLOW_SNAPSHOT_ISOLATION ON

ALTER DATABASE MyDatabase SET
READ_COMMITTED_SNAPSHOT ON
```

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 runtime language (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;
```

Microsoft

Azure Synapse Analytics

Data Storage and Performance Optimizations

Database Tables

Optimized Storage

Reduce Migration Risk

Less Data Scanned

Smaller Cache Required

Smaller Clusters

Faster Queries

Columnar Storage

Columnar Ordering

Table Partitioning

Hash Distribution

Nonclustered Indexes

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);
```

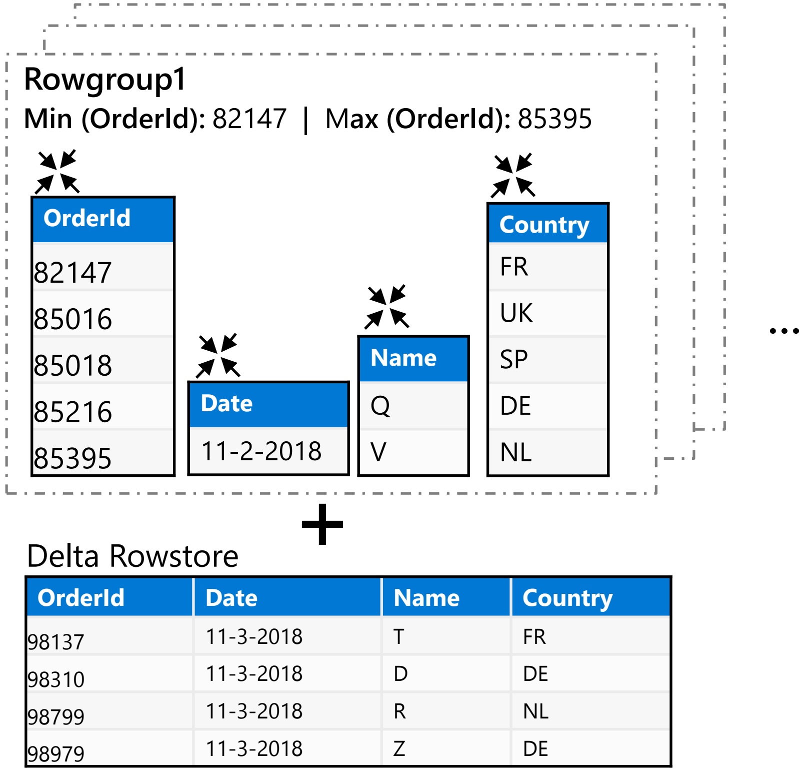
SQL Analytics Columnstore Tables

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)



Rowgroup1

Min (OrderId): 82147 | **Max (OrderId):** 85395

OrderId
82147
85016
85018
85216
85395

Date
11-2-2018

Name
Q
V

Country
FR
UK
SP
DE
NL

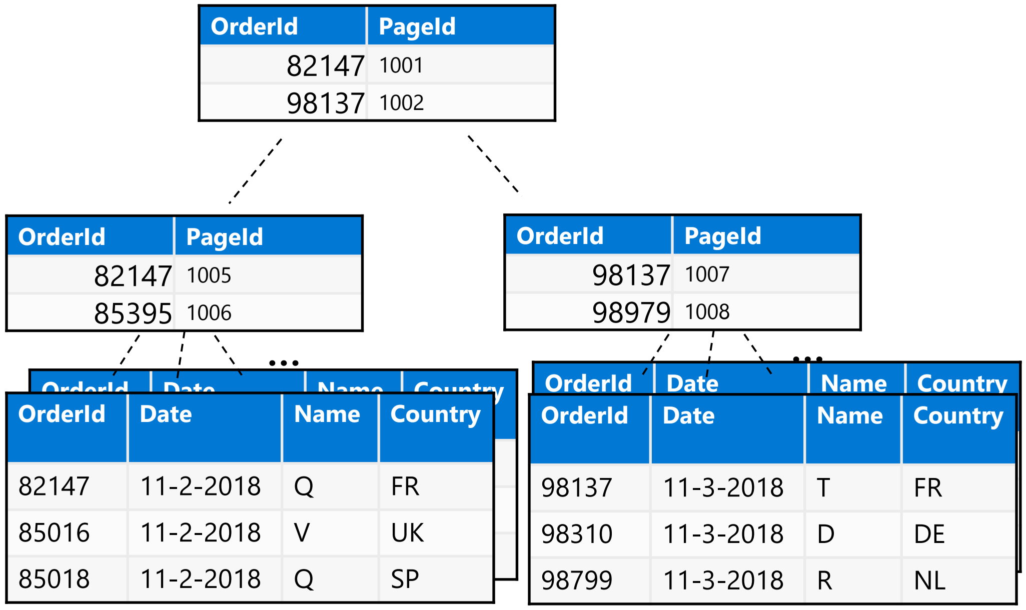
Delta Rowstore

OrderId	Date	Name	Country
98137	11-3-2018	T	FR
98310	11-3-2018	D	DE
98799	11-3-2018	R	NL
98979	11-3-2018	Z	DE

- 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)



OrderId	PageId
82147	1001
98137	1002

OrderId	PageId
82147	1005
85395	1006

OrderId	PageId
98137	1007
98979	1008

OrderId	Date	Name	Country
82147	11-2-2018	Q	FR
85016	11-2-2018	V	UK
85018	11-2-2018	Q	SP

OrderId	Date	Name	Country
98137	11-3-2018	T	FR
98310	11-3-2018	D	DE
98799	11-3-2018	R	NL

- 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

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'
    )
  )
);
```

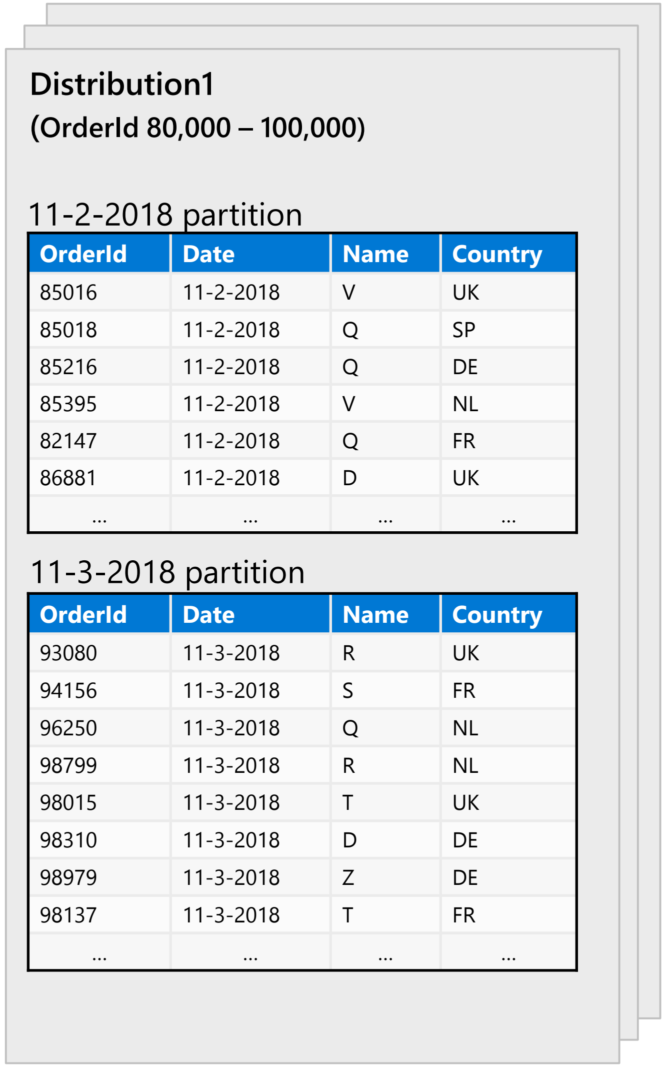
Tables – Distributions & Partitions

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	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. Operations that benefit: COUNT(DISTINCT(<hashed_key>)) OVER PARTITION BY <hashed_key> most JOIN <table_name> ON <hashed_key> GROUP BY <hashed_key>
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.

Database Views

Views

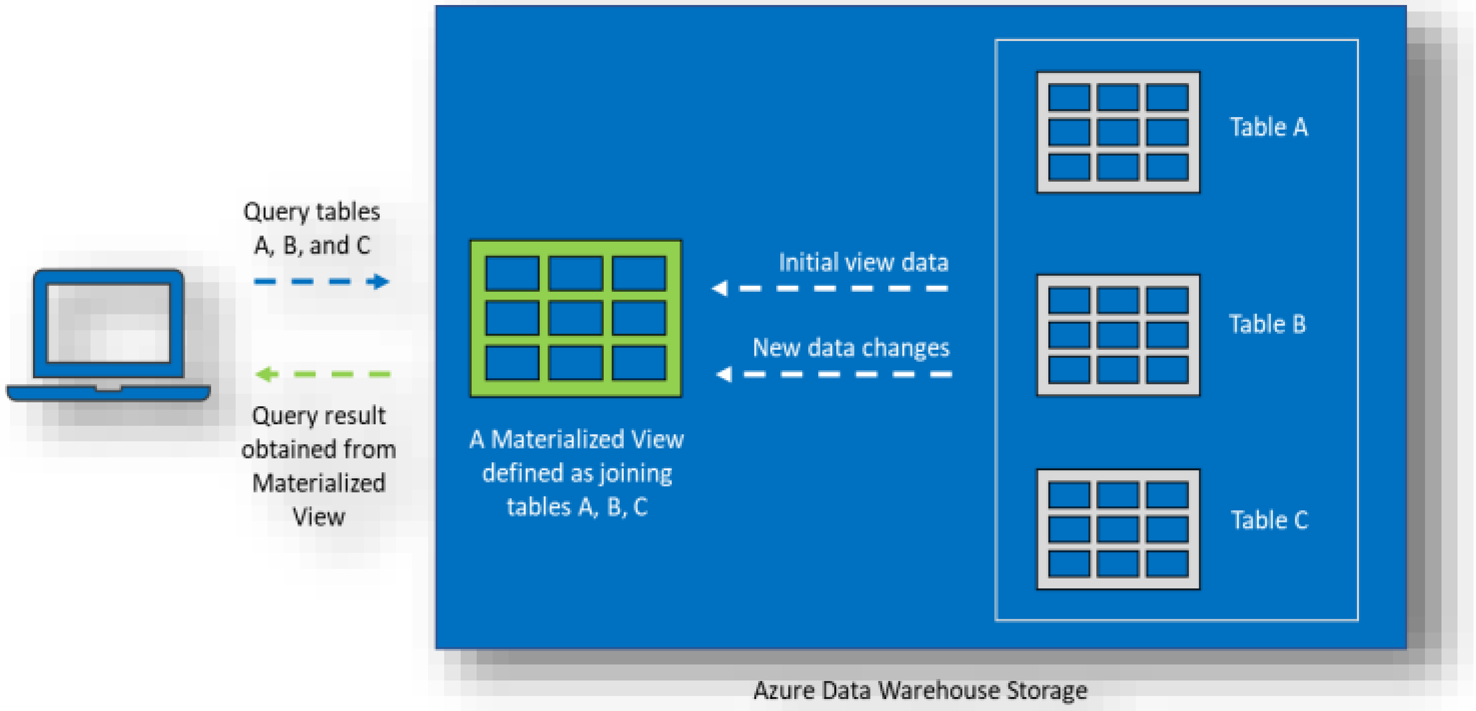
Materialized Views

Scope: **Generally Available**

Best in class price performance

Interactive dashboarding with Materialized Views

- Automatic data refresh and maintenance
- Automatic query rewrites to improve performance
- Built-in advisor


Query tables A, B, and C
Query result obtained from Materialized View
A Materialized View defined as joining tables A, B, C
Initial view data
New data changes
Table A
Table B
Table C
Azure Data Warehouse Storage

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

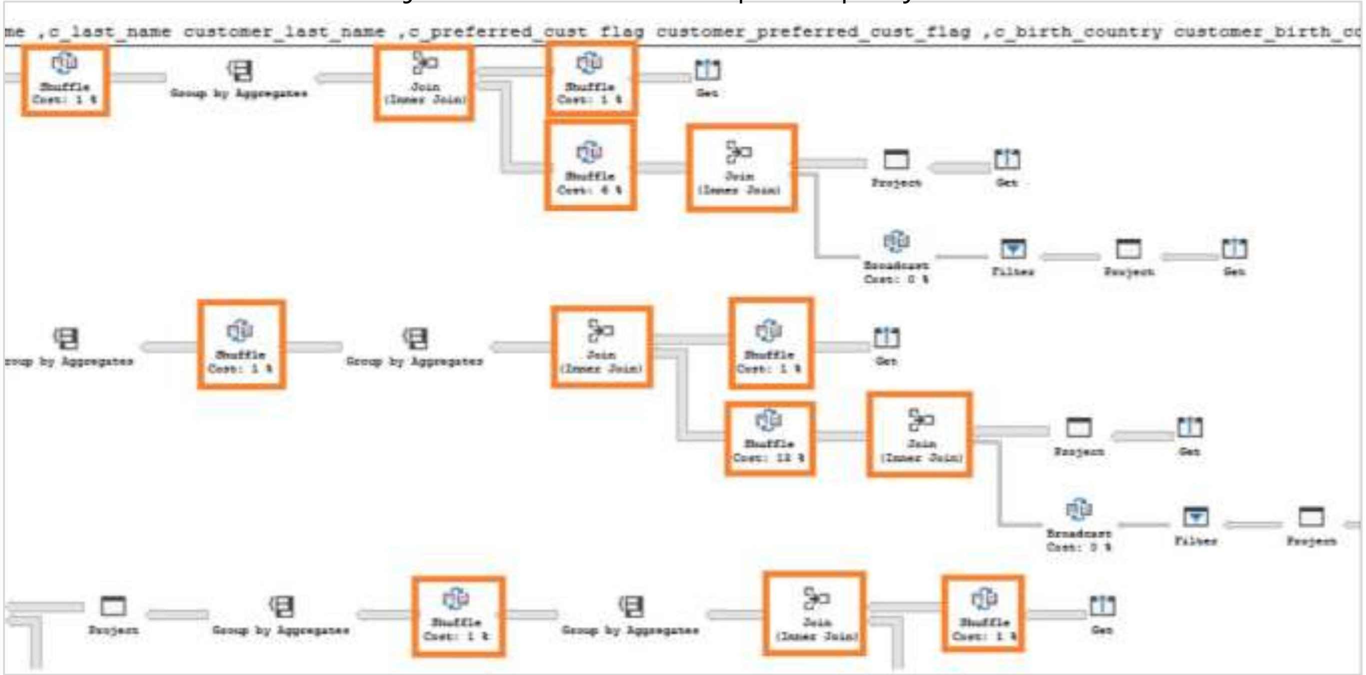
In this example, a query to get the year total sales per customer is shown to have a lot of data shuffles and joins that contribute to slow performance:

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



```
me ,c_last_name customer_last_name ,c_preferred_cust_flag customer_preferred_cust_flag ,c_birth_country customer_birth_cc
```

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

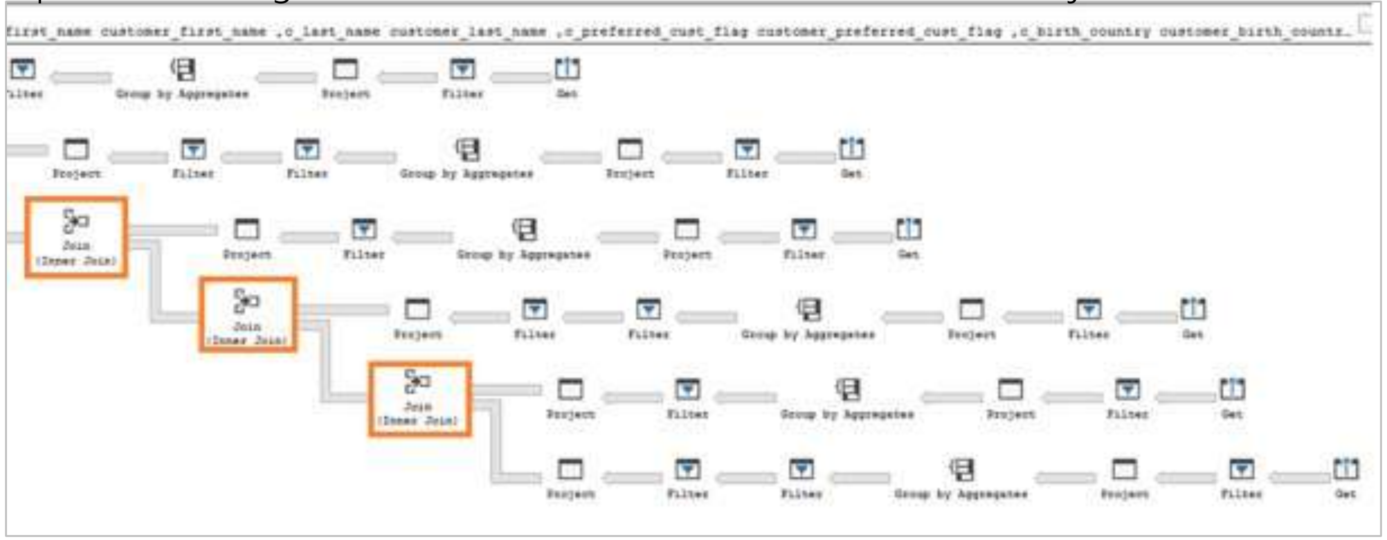
The SQL Data Warehouse query optimizer automatically leverages the indexed view to speed up the same query. Notice that the query does not need to reference the view directly

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

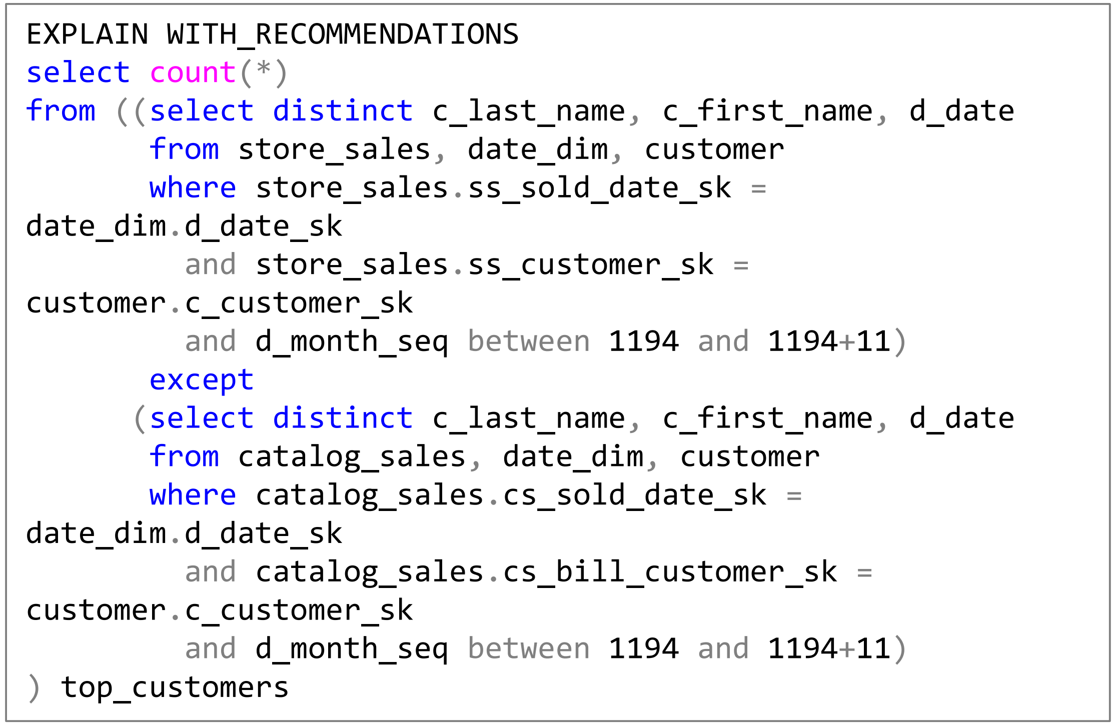


```
first_name customer_first_name ,c_last_name customer_last_name ,c_preferred_cust_flag customer_preferred_cust_flag ,c_birth_country customer_birth_countr…
```

Materialized views- Recommendations

EXPLAIN - provides query plan for SQL Data Warehouse 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.



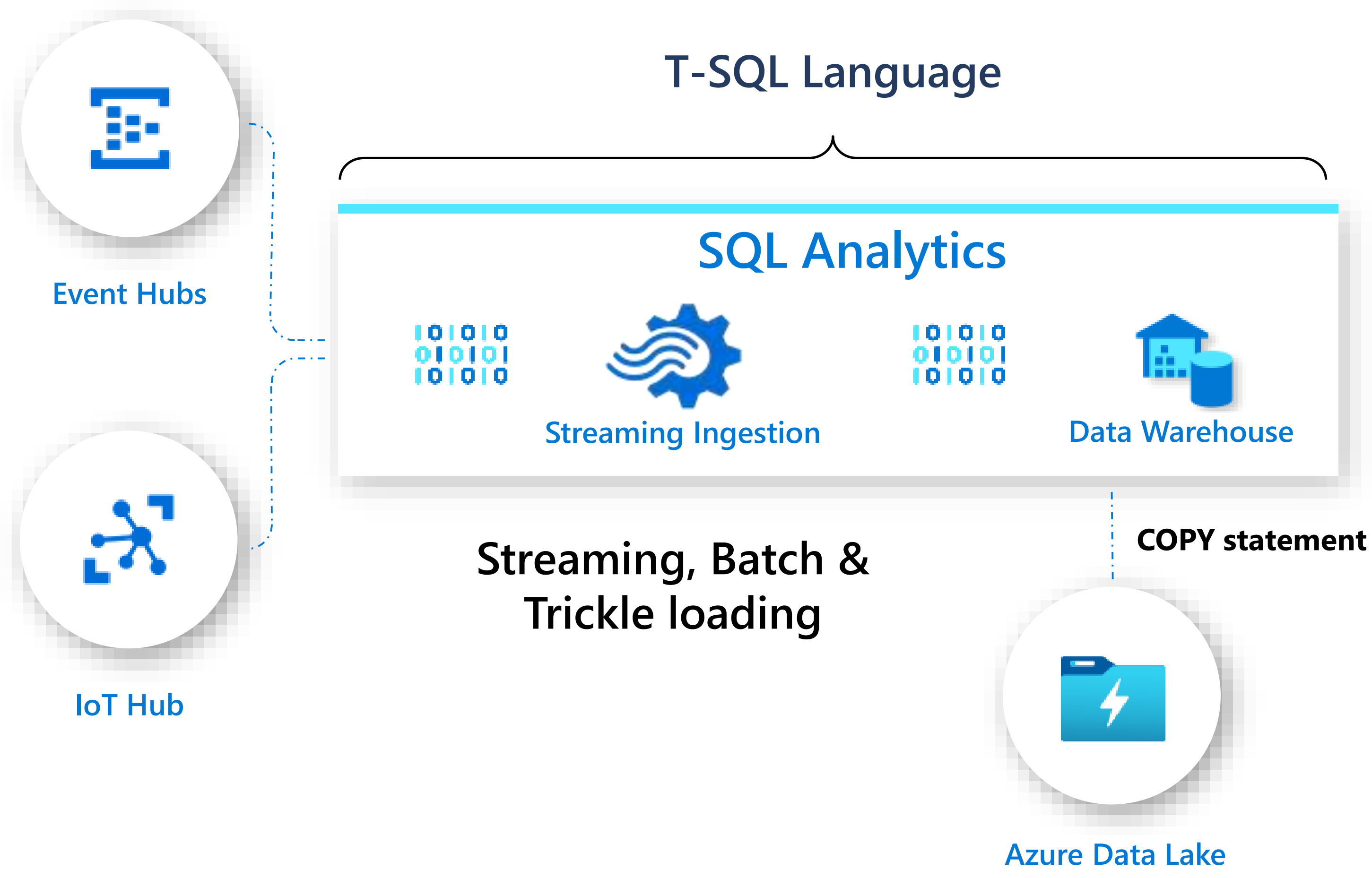
```
EXPLAIN WITH_RECOMMENDATIONS
select count(*)
from ((select distinct c_last_name, c_first_name, d_date
       from store_sales, date_dim, customer
       where store_sales.ss_sold_date_sk =
date_dim.d_date_sk
         and store_sales.ss_customer_sk =
customer.c_customer_sk
         and d_month_seq 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
```

Scope: **Public Preview**

Heterogenous Data Preparation & Ingestion

COPY statement

- Simplified permissions (no CONTROL required)
- No need for external tables
- Standard CSV support (i.e. custom row terminators, escape delimiters, SQL dates)
- User-driven file selection (wild card support)



```
--Copy files in parallel directly into data warehouse table
COPY INTO [dbo].[weatherTable]
FROM
'abfss://<storageaccount>.blob.core.windows.net/<filepath>'
WITH (
FILE_FORMAT = 'DELIMITEDTEXT’,
SECRET = CredentialObject);
```

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.

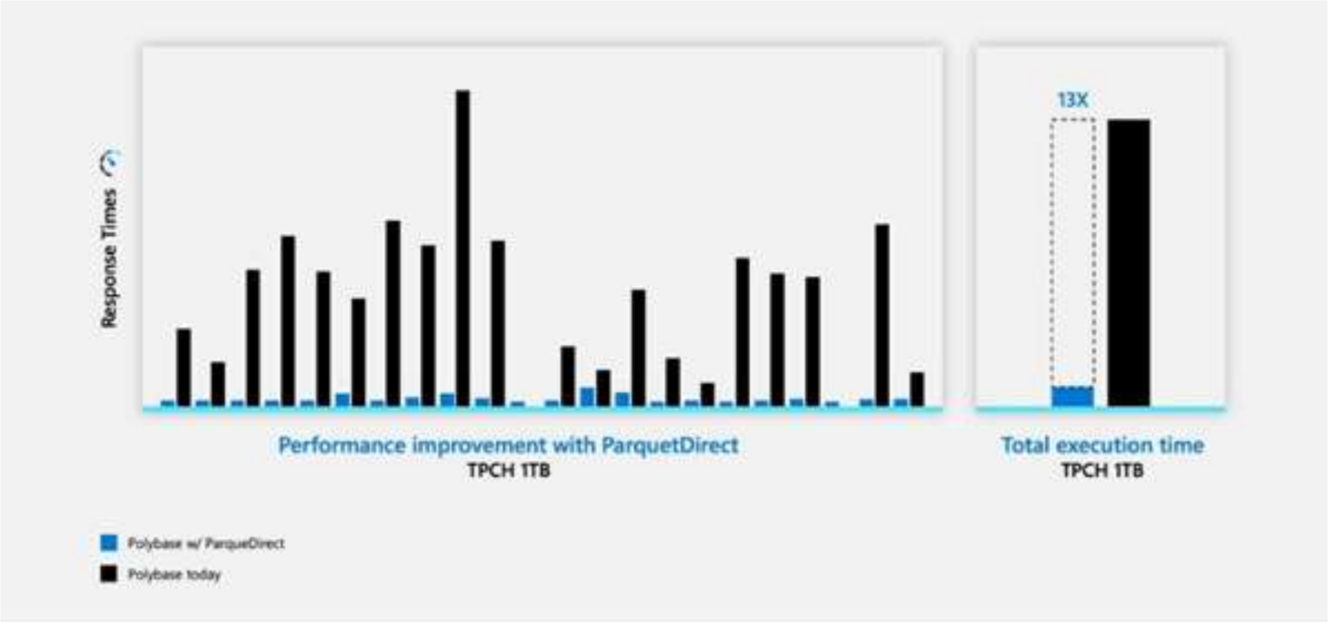
Supports CSV, PARQUET, ORC file formats

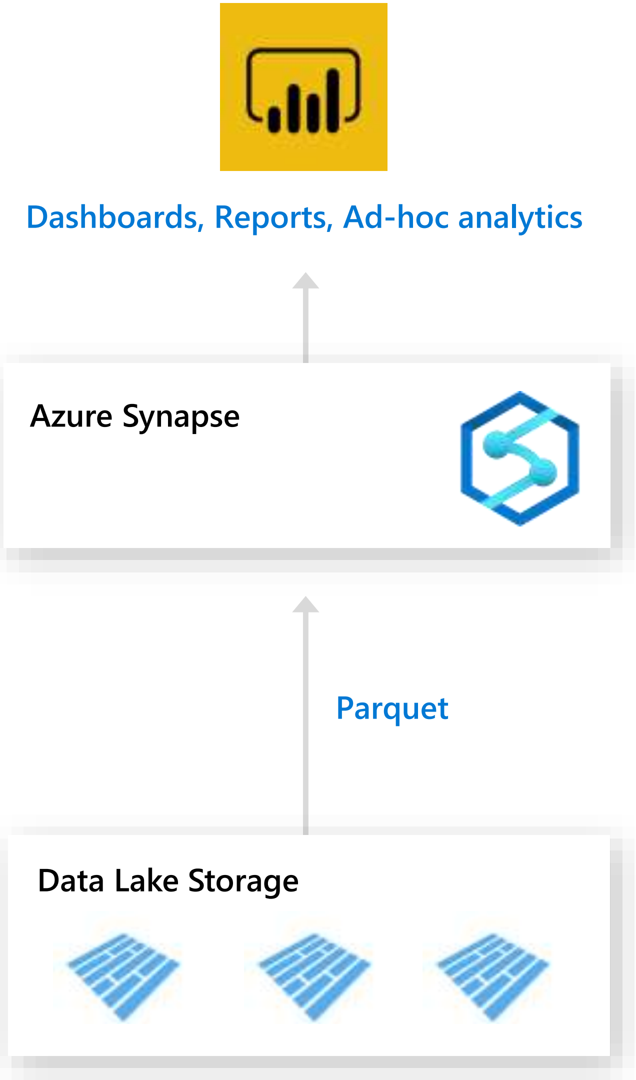
```
COPY INTO test_1
FROM
'https://XXX.blob.core.windows.net/customerdatasets/tes
t_1.txt'
WITH (
    FILE_TYPE = 'CSV',
    CREDENTIAL=(IDENTITY= 'Shared Access Signature',
SECRET='<Your_SAS_Token>'),
    FIELDQUOTE = '"',
    FIELDTERMINATOR=';',
    ROWTERMINATOR='0X0A',
    ENCODING = 'UTF8',
    DATEFORMAT = 'ymd',
    MAXERRORS = 10,
    ERRORFILE = '/errorsfolder/'--path starting from
the storage container,
    IDENTITY_INSERT
    )
```

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

Data Flexibility – Parquet Direct

Overview





Dashboards, Reports, Ad-hoc analytics

Azure Synapse

Parquet

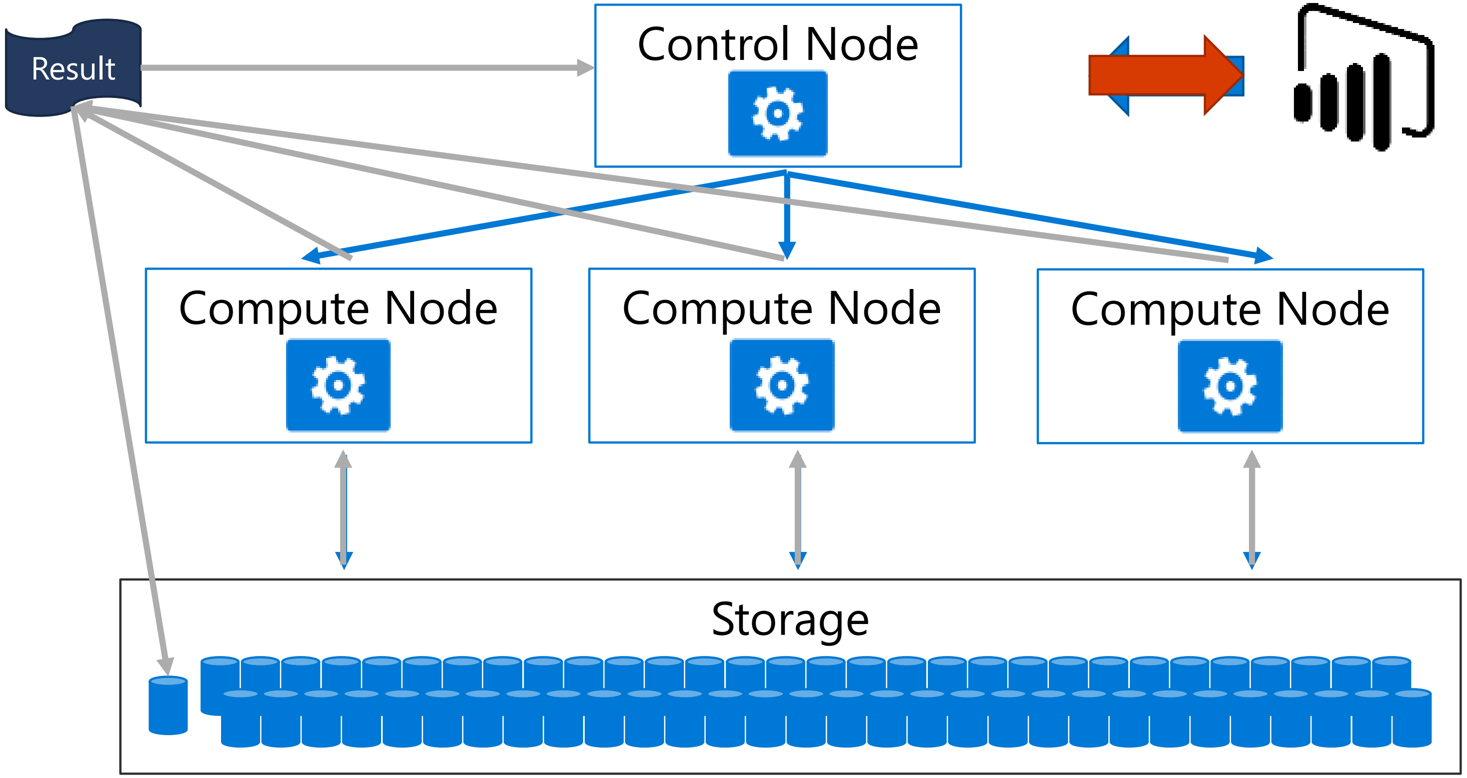
Data Lake Storage

Scope: **Generally Available**

Best in class price performance

Interactive dashboarding with Resultset Caching

- Millisecond responses with resultset caching
- Cache survives pause/resume/scale operations
- Fully managed cache (1TB in size)



Enable caching: **Alter Database** <DBNAME> **Set Result_Set_Caching ON**
Purge cache: **DBCC DropResultSetCache**

Result-set caching

Overview

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

The result-set cache persists even if a data warehouse 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

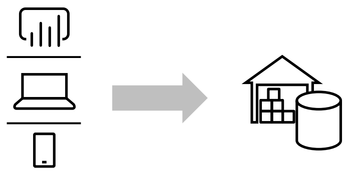
Enhances performance when same result is requested repetitively

Reduced load on server for repeated queries

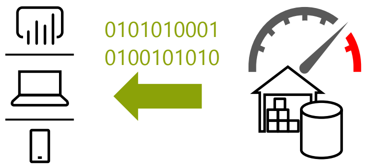
Offers monitoring of query execution with a result cache hit or miss

```sql
-- 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 data warehouse
SET RESULT_SET_CACHING {ON | OFF}

-- Check result-set caching setting for a database
-- Run on target data warehouse
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
```

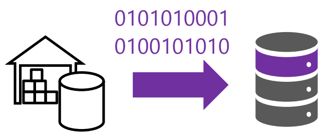
Result-set caching flow



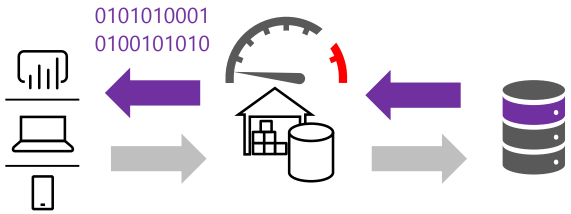
1 Client sends query to DW



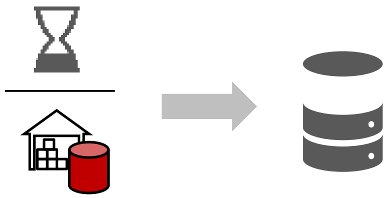
2 Query is processed using DW compute nodes which pull data from remote storage, process query and output back to client app



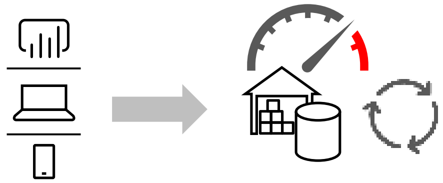
Query results are cached in remote storage so subsequent requests can be served immediately



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

Resource classes

Overview

Pre-determined resource limits defined for a user or role.

Benefits

Govern the system memory assigned to each query.

Effectively used to control the number of concurrent queries that can run on a data warehouse.

Exemptions to concurrency limit:

CREATE|ALTER|DROP (TABLE|USER|PROCEDURE|VIEW|LOGIN)

CREATE|UPDATE|DROP (STATISTICS|INDEX)

SELECT from system views and DMVs

EXPLAIN

Result-Set Cache

TRUNCATE TABLE

ALTER AUTHORIZATION

CREATE|UPDATE|DROP STATISTICS

```
/* View resource classes in the data warehouse */
SELECT name
FROM   sys.database_principals
WHERE  name LIKE '%rc%' AND type_desc = 'DATABASE_ROLE';

/* Change user’s resource class to 'largerc' */
EXEC sp_addrolemember 'largerc', 'loaduser’;

/* Decrease the loading user's resource class */
EXEC sp_droprolemember 'largerc', 'loaduser';
```

Resource class types

Static Resource Classes

Allocate the same amount of memory independent of the current service-level objective (SLO).

Well-suited for fixed data sizes and loading jobs.

Dynamic Resource Classes

Allocate a variable amount of memory depending on the current SLO.

Well-suited for growing or variable datasets.

All users default to the *smallrc* dynamic resource class.

Static resource classes:

```
staticrc10 | staticrc20 | staticrc30 |
staticrc40 | staticrc50 | staticrc60 |
staticrc70 | staticrc80
```

Dynamic resource classes:

```
smallrc | mediumrc | largerc | xlargerc
```

Resource Class	Percentage Memory	Max. Concurrent Queries
smallrc	3%	32
mediumrc	10%	10
largerc	22%	4
xlargerc	70%	1

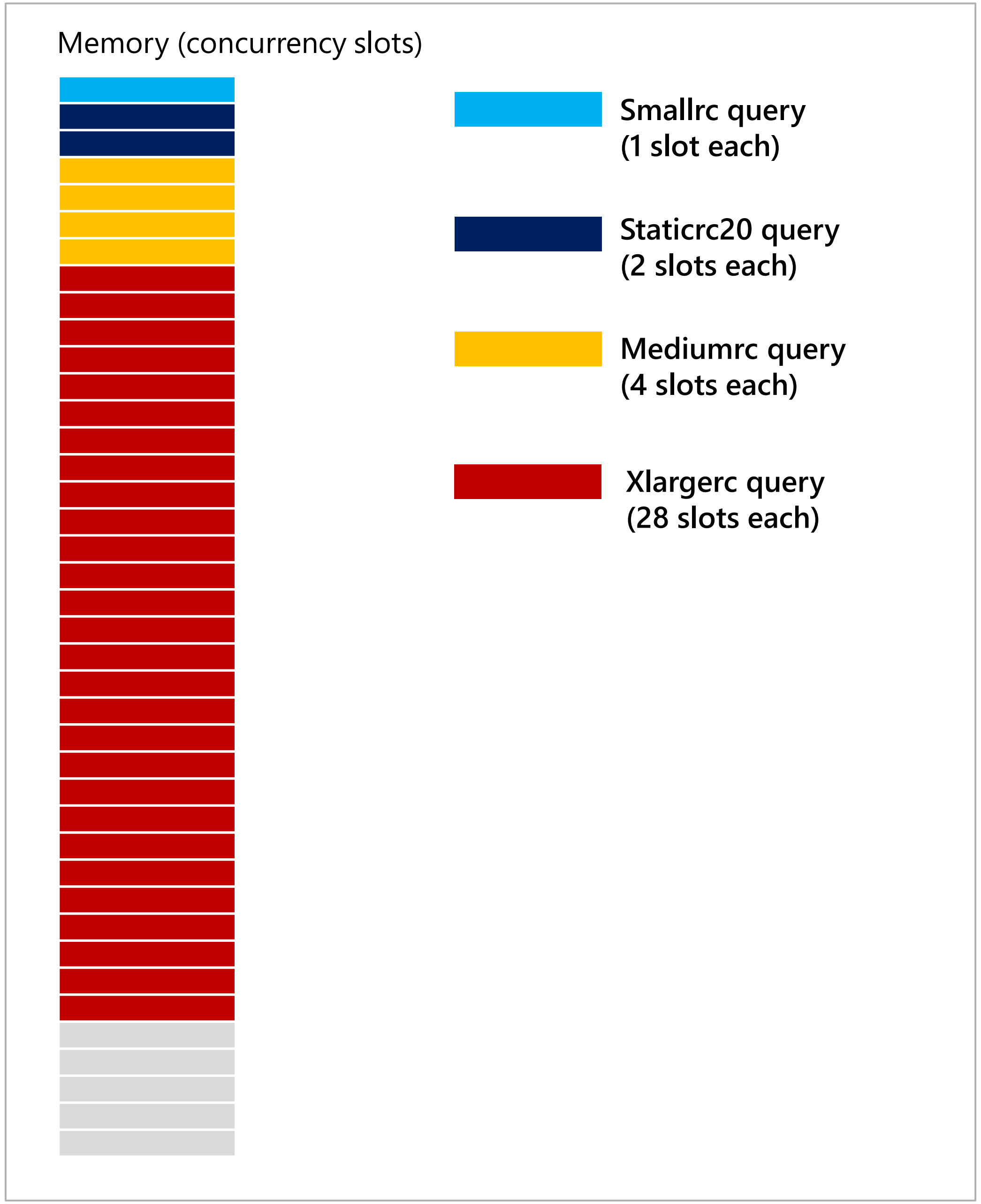
Concurrency slots

Overview

Queries running on a DW compete for access to system resources (CPU, IO, and memory).

To guarantee access to resources, running queries are assigned a chunk of system memory (**a concurrency slot**) for processing the query. The amount given is determined by the resource class of the user executing the query. Higher DW SLOs provide more memory and concurrency slots

@DW1000c: **40 concurrency slots**



Memory (concurrency slots)

Smallrc query (1 slot each)

Staticrc20 query (2 slots each)

Mediumrc query (4 slots each)

Xlargerc query (28 slots each)

Concurrent query limits

Overview

The limit on how many queries can run at the same time is governed by two properties:

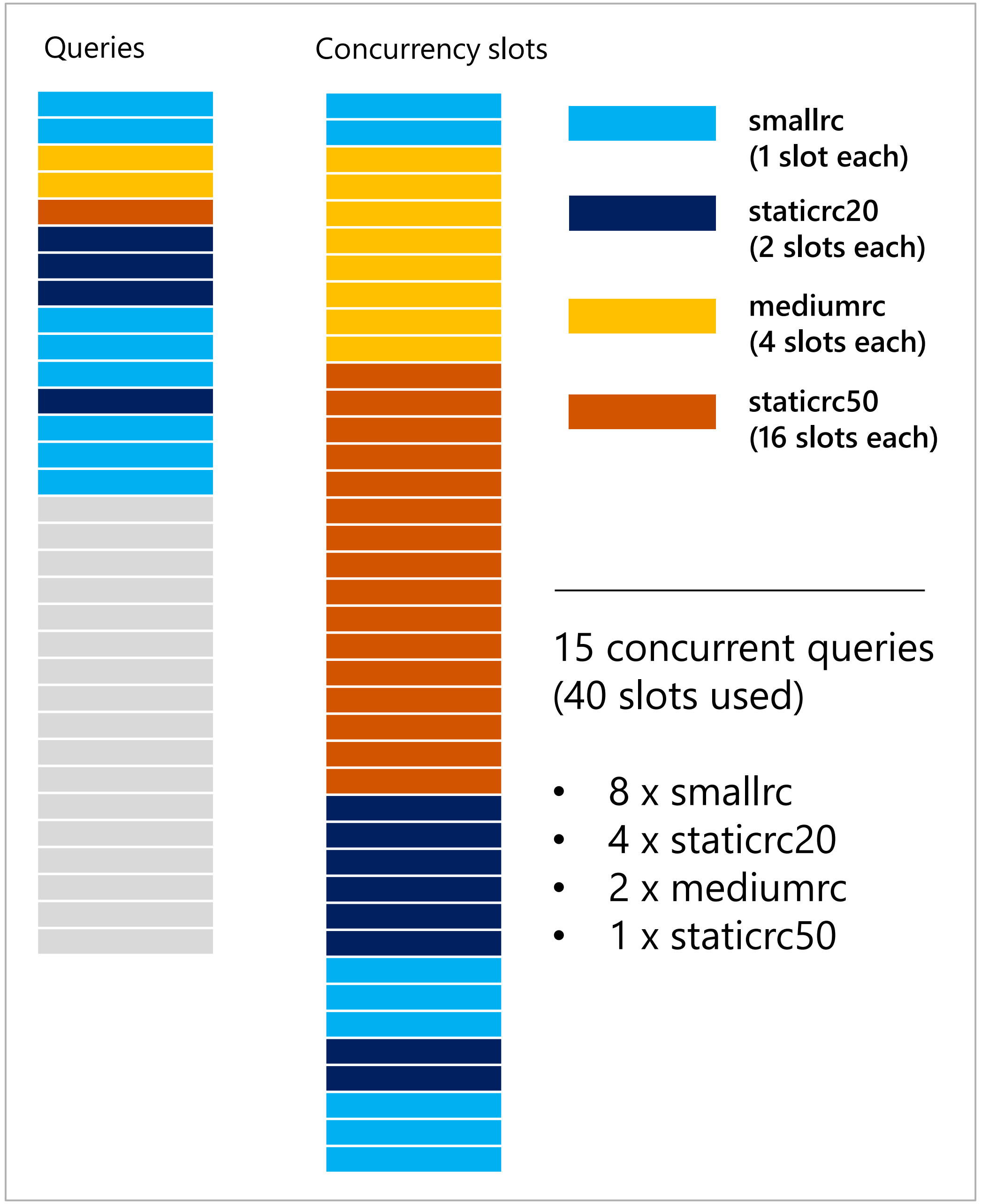
- The max. concurrent query count for the DW SLO
- The total available memory (concurrency slots) for the DW SLO

Increase the concurrent query limit by:

- Scaling up to a higher DW SLO (up to 128 concurrent queries)
- Using lower resource classes that use less memory per query

[Concurrency limits based on resource classes]()

@DW1000c: **32 max concurrent queries, 40 slots**



15 concurrent queries
(40 slots used)

- 8 x smallrc
- 4 x staticrc20
- 2 x mediumrc
- 1 x staticrc50

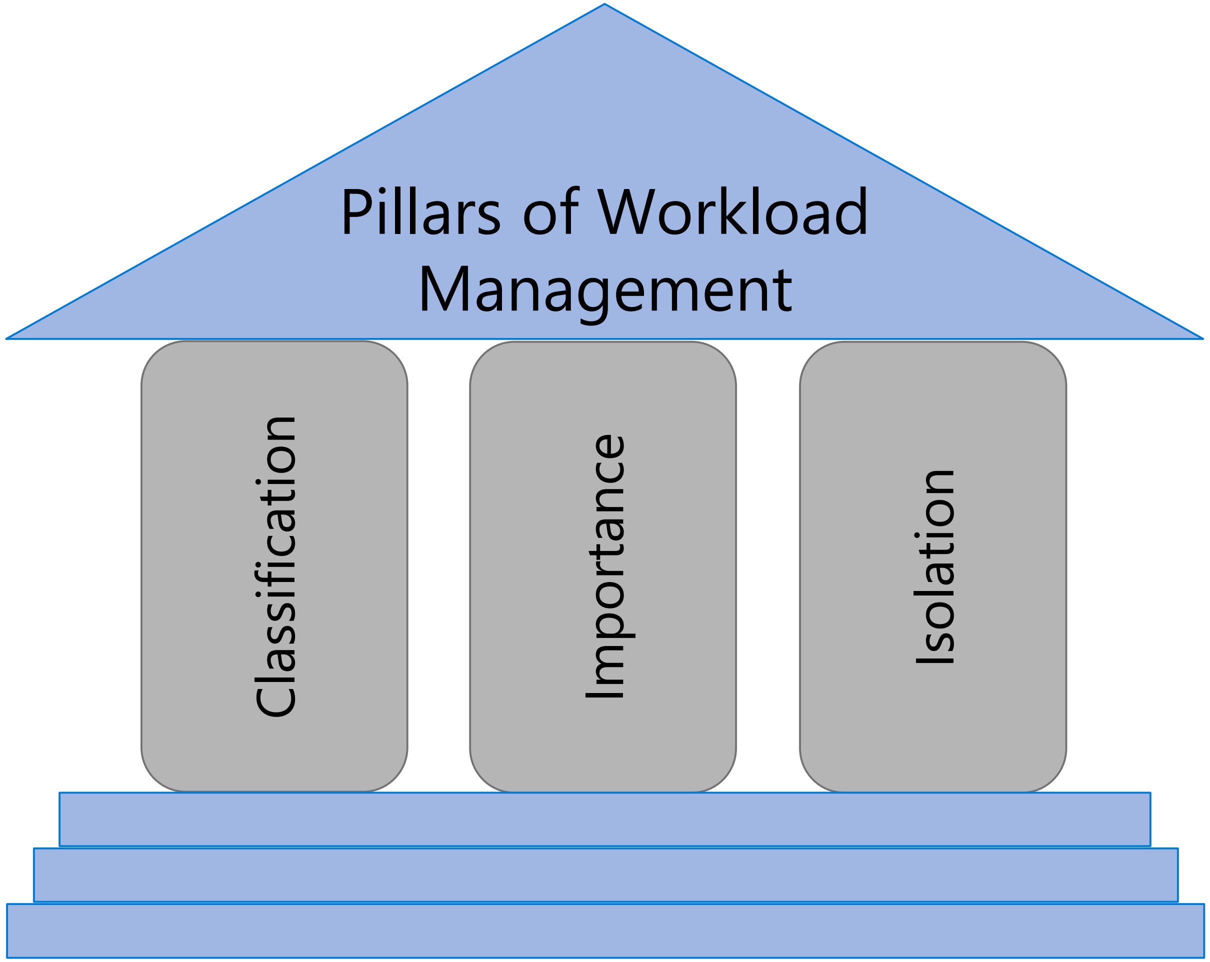
Workload Management

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 (smallrc resource class).

Benefits

Map queries to both Resource Management and Workload Isolation concepts.

Manage groups of users with only a few classifiers.

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 = '<Resource Class>' ]
  [IMPORTANCE = { LOW                      |
                   BELOW_NORMAL      |
                   NORMAL            |
                   ABOVE_NORMAL      |
                   HIGH
               }
  ]
  [MEMBERNAME = ‘security_account’]
)
 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

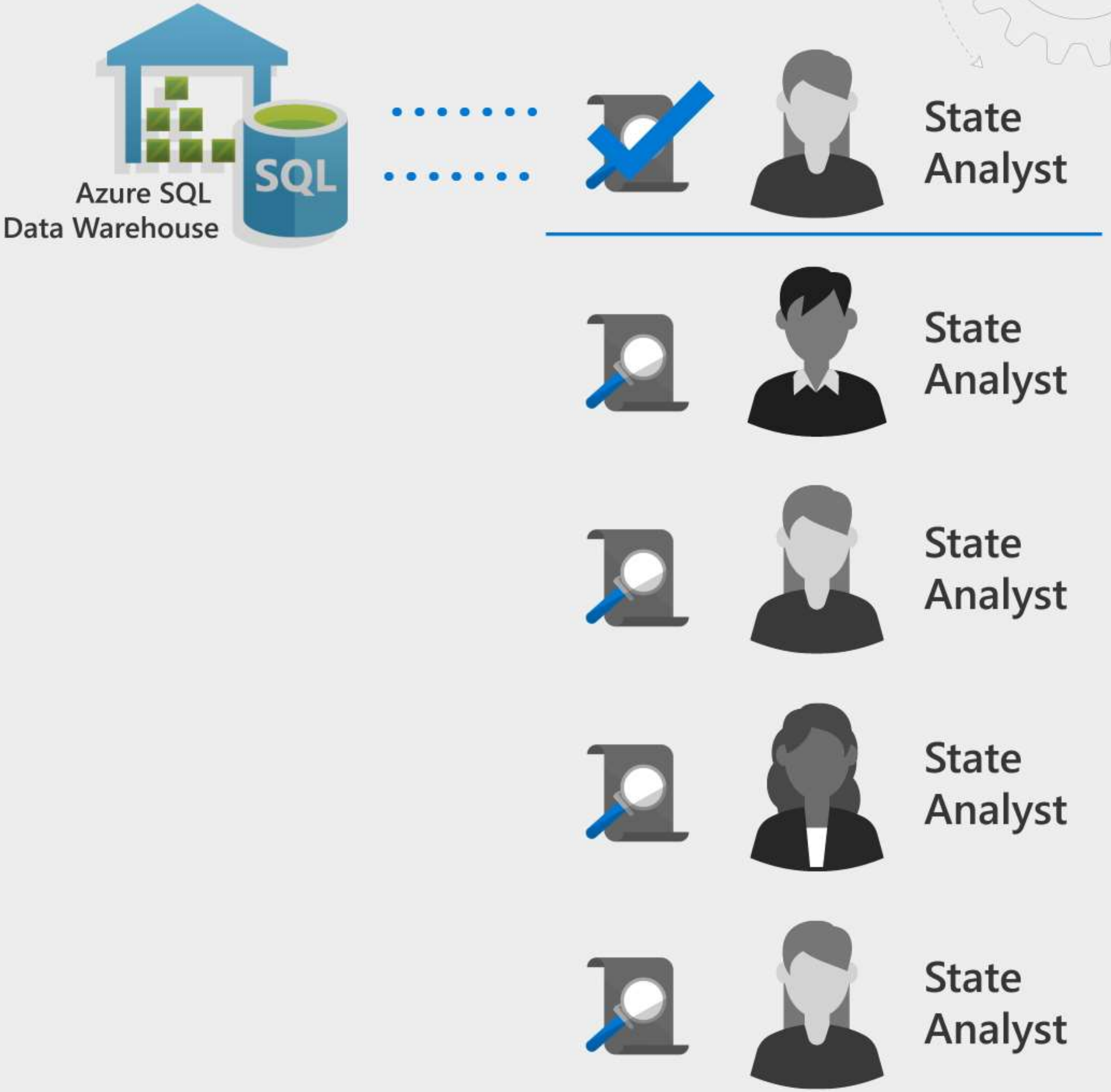
State analysts have normal importance.

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 = ‘smallrc’]
   [IMPORTANCE  = HIGH]
   [MEMBERNAME  = ‘National_Analyst_Login’]
```



Scope: **Public Preview**

```
CREATE WORKLOAD GROUP Sales
WITH
(
   [ MIN_PERCENTAGE_RESOURCE = 60 ]
   [ CAP_PERCENTAGE_RESOURCE = 100 ]
   [ MAX_CONCURRENCY = 6 ]  )
```

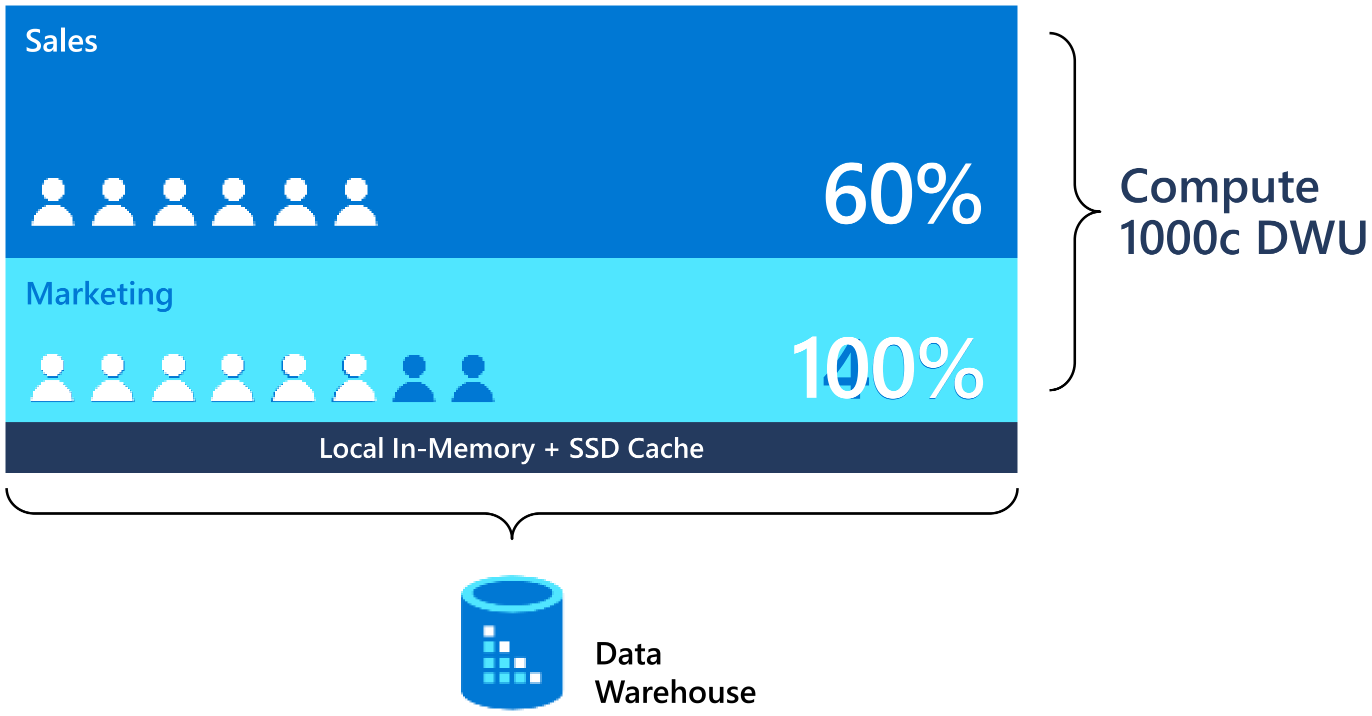
Workload aware query execution

Workload Isolation

- Multiple workloads share deployed resources
- Reservation or shared resource configuration
- Online changes to workload policies

Intra Cluster Workload Isolation

(Scale In)



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

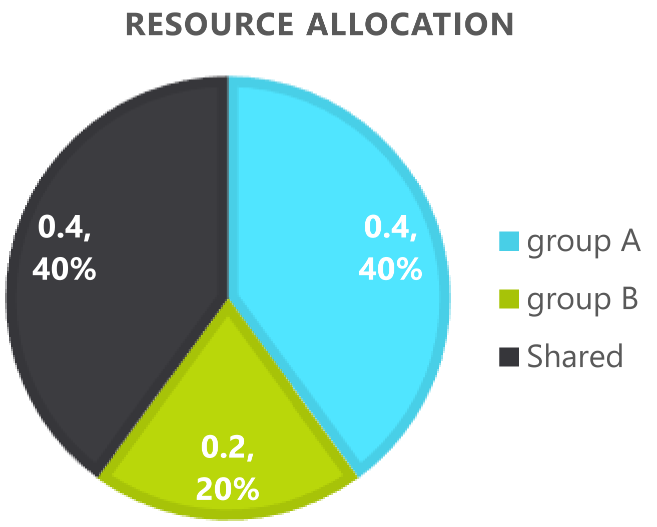
Set Query timeout value. Get DBAs out of the business of killing runaway queries

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 ]
 )[ ; ]
```



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

```sql
--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

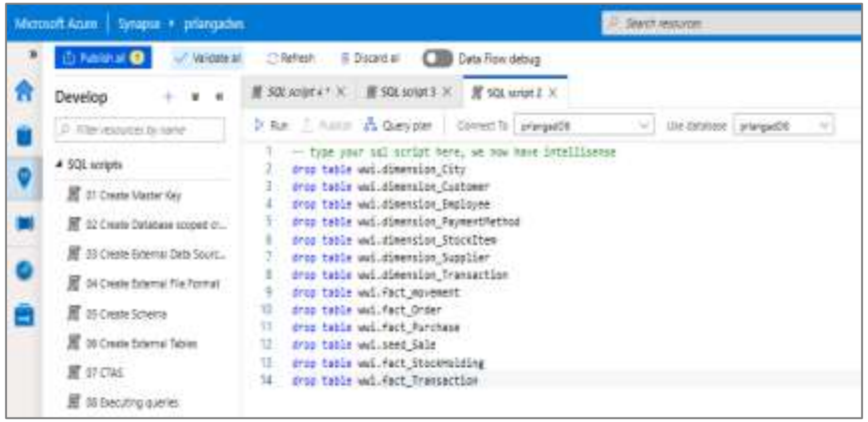
```sql
-- List all open sessions
SELECT * FROM sys.dm_pdw_exec_sessions where status <> 'Closed'
and session_id <> session_id();
```

List all active queries

```sql
-- 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;
```

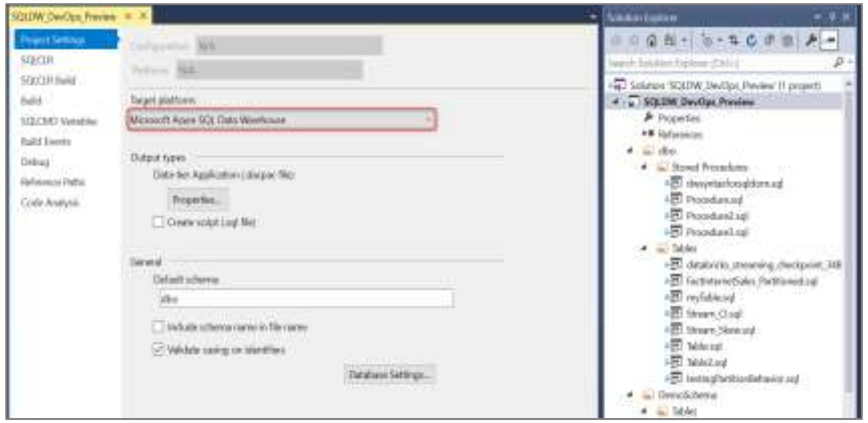
Developer Tools

Azure Synapse Analytics



```
-- type your sql script here, we now have intellisense
drop table wwi.dimension_City
drop table wwi.dimension_Customer
drop table wwi.dimension_Employee
drop table wwi.dimension_PaymentMethod
drop table wwi.dimension_StockItem
drop table wwi.dimension_Supplier
drop table wwi.dimension_Transaction
drop table wwi.fact_movement
drop table wwi.fact_Order
drop table wwi.fact_Purchase
drop table wwi.seed_Sale
drop table wwi.fact_StockHolding
drop table wwi.fact_Transaction
```

Visual Studio - SSDT database projects



Target platform: Microsoft Azure SQL Data Warehouse

Output types: Data-tier Application (.dacpac file)

□ Create script (.sql file)

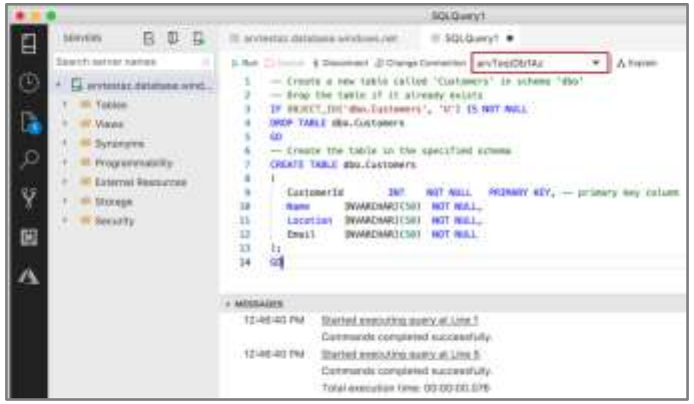
General

Default schema: dbo

□ Include schema name in file name

☑ Validate casing on identifiers

Azure Data Studio (queries, extensions etc.)

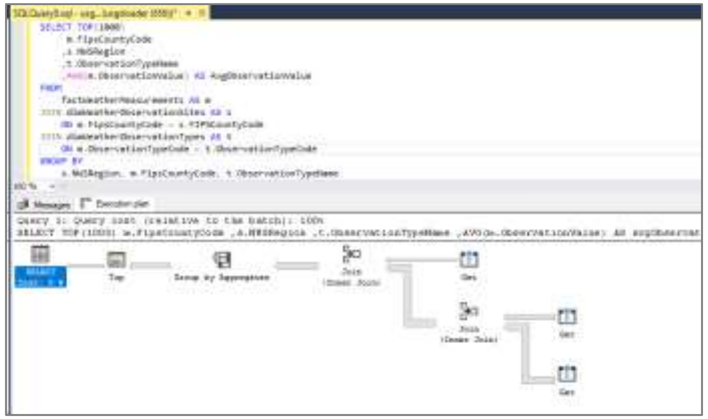


```
-- Create a new table called 'Customers' in schema 'dbo'
-- Drop the table if it already exists
IF OBJECT_ID('dbo.Customers', 'U') IS NOT NULL
DROP TABLE dbo.Customers
GO
-- Create the table in the specified schema
CREATE TABLE dbo.Customers
(
   CustomerId        INT     NOT NULL   PRIMARY KEY, -- primary key column
   Name       [NVARCHAR](50)  NOT NULL,
   Location   [NVARCHAR](50)  NOT NULL,
   Email      [NVARCHAR](50)  NOT NULL
);
GO
```

MESSAGES

```
12:48:40 PM   Started executing query at Line 1
              Commands completed successfully.
12:48:40 PM   Started executing query at Line 5
              Commands completed successfully.
              Total execution time: 00:00:00.076
```

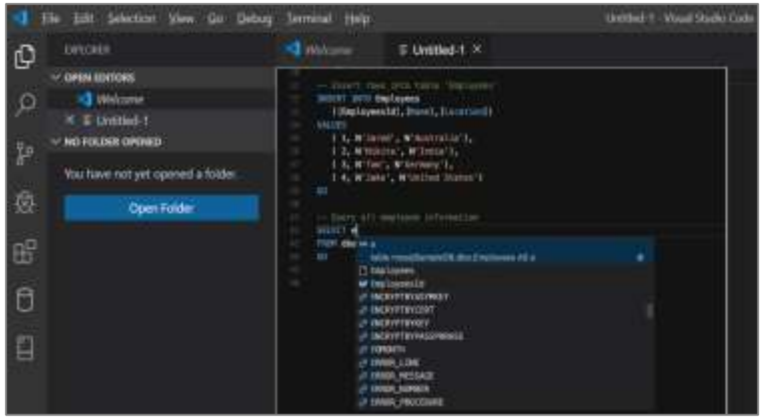
SQL Server Management Studio (queries, execution plans etc.)



```
SELECT TOP(1000)
    m.FipsCountyCode
   ,s.NWSRegion
   ,t.ObservationTypeName
   ,AVG(m.ObservationValue) AS AvgObservationValue
FROM
   factWeatherMeasurements AS m
JOIN dimWeatherObservationSites AS s
   ON m.FipsCountyCode = s.FIPSCountyCode
JOIN dimWeatherObservationTypes AS t
   ON m.ObservationTypeCode = t.ObservationTypeCode
GROUP BY
   s.NWSRegion, m.FipsCountyCode, t.ObservationTypeName
```

```
Query 1: Query cost (relative to the batch): 100%
SELECT TOP(1000) m.FipsCountyCode ,s.NWSRegion ,t.ObservationTypeName ,AVG(m.ObservationValue) AS avgObservat
```

Visual Studio Code

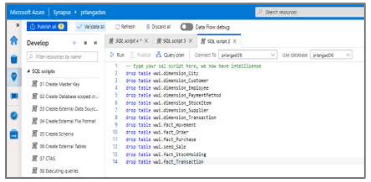


You have not yet opened a folder.

```
-- Insert rows into table 'Employees'
INSERT INTO Employees
   ([EmployeesId],[Name],[Location])
VALUES
   ( 1, N'Jared', N'Australia'),
   ( 2, N'Nikita', N'India'),
   ( 3, N'Tom', N'Germany'),
   ( 4, N'Jake', N'United States')
GO

-- Query all employee information
SELECT e
FROM dbo.Employees e
GO
```

Developer Tools

Azure Synapse Analytics

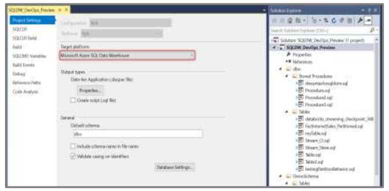


Azure Cloud Service

Offers end-to-end lifecycle for analytics

Connects to multiple services

Visual Studio - SSDT database projects



Runs on Windows

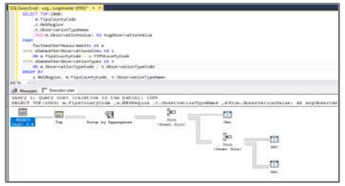
Create, maintain database code, compile, code refactoring

Azure Data Studio

Runs on Windows, Linux, macOS

Light weight editor, (queries and extensions)

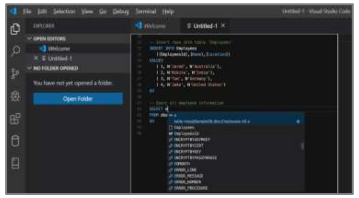
SQL Server Management Studio



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)

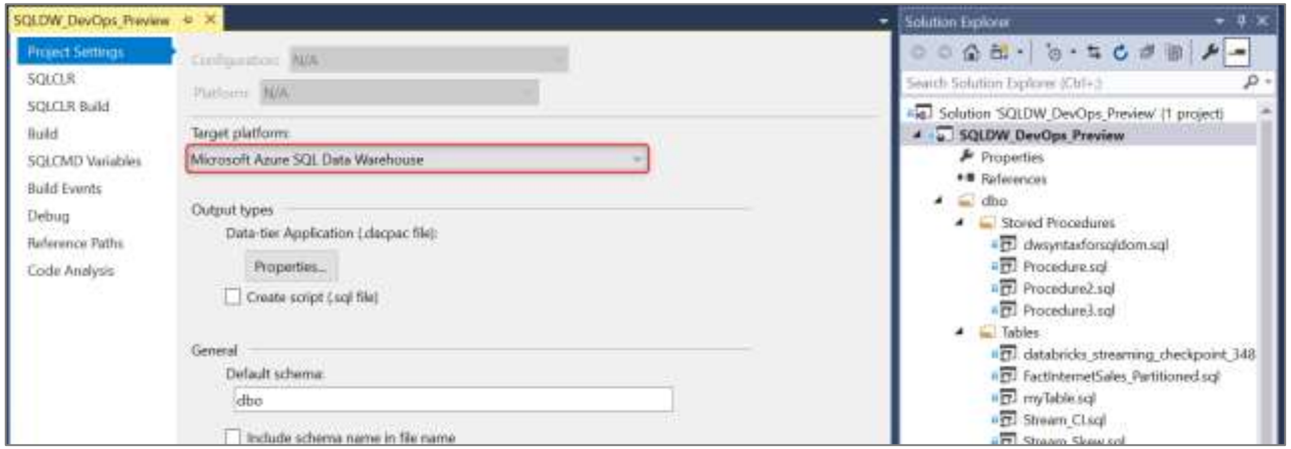
Overview

Database project support in SQL Server Data Tools (SSDT) allows teams of developers to collaborate over a version-controlled data warehouse, 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.)



SQLDW_DevOps_Preview

Configuration: N/A

Platform: N/A

Target platform: Microsoft Azure SQL Data Warehouse

Output types

Data-tier Application (.dacpac file):

☐ Create script (.sql file)

General

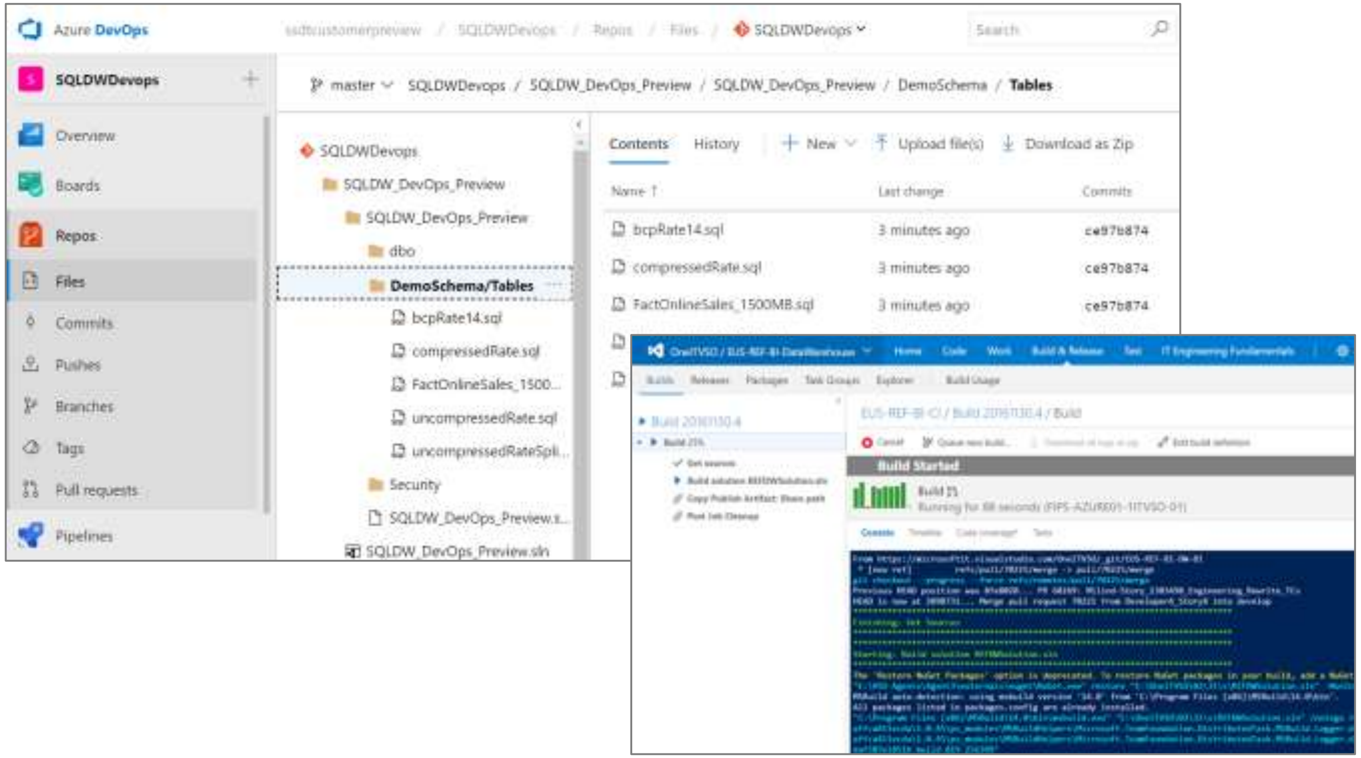
Default schema: dbo

☐ Include schema name in file name

Solution Explorer

Search Solution Explorer (Ctrl+;)

```
Solution 'SQLDW_DevOps_Preview' (1 project)
  SQLDW_DevOps_Preview
    Properties
    References
    dbo
      Stored Procedures
        dwsyntaxforsqldom.sql
        Procedure.sql
        Procedure2.sql
        Procedure3.sql
      Tables
        databricks_streaming_checkpoint_348
        FactInternetSales_Partitioned.sql
        myTable.sql
        Stream_CI.sql
        Stream_Skew.sql
```



Azure DevOps   ssdtcustomerpreview / SQLDWDevops / Repos / Files / SQLDWDevops

master ∨ SQLDWDevops / SQLDW_DevOps_Preview / SQLDW_DevOps_Preview / DemoSchema / **Tables**

```
SQLDWDevops
  SQLDW_DevOps_Preview
    SQLDW_DevOps_Preview
      dbo
      DemoSchema/Tables
        bcpRate14.sql
        compressedRate.sql
        FactOnlineSales_1500...
        uncompressedRate.sql
        uncompressedRateSpli...
      Security
      SQLDW_DevOps_Preview.s...
  SQLDW_DevOps_Preview.sln
```

Name ↑	Last change	Commits
bcpRate14.sql	3 minutes ago	ce97b874
compressedRate.sql	3 minutes ago	ce97b874
FactOnlineSales_1500MB.sql	3 minutes ago	ce97b874

Build Started

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

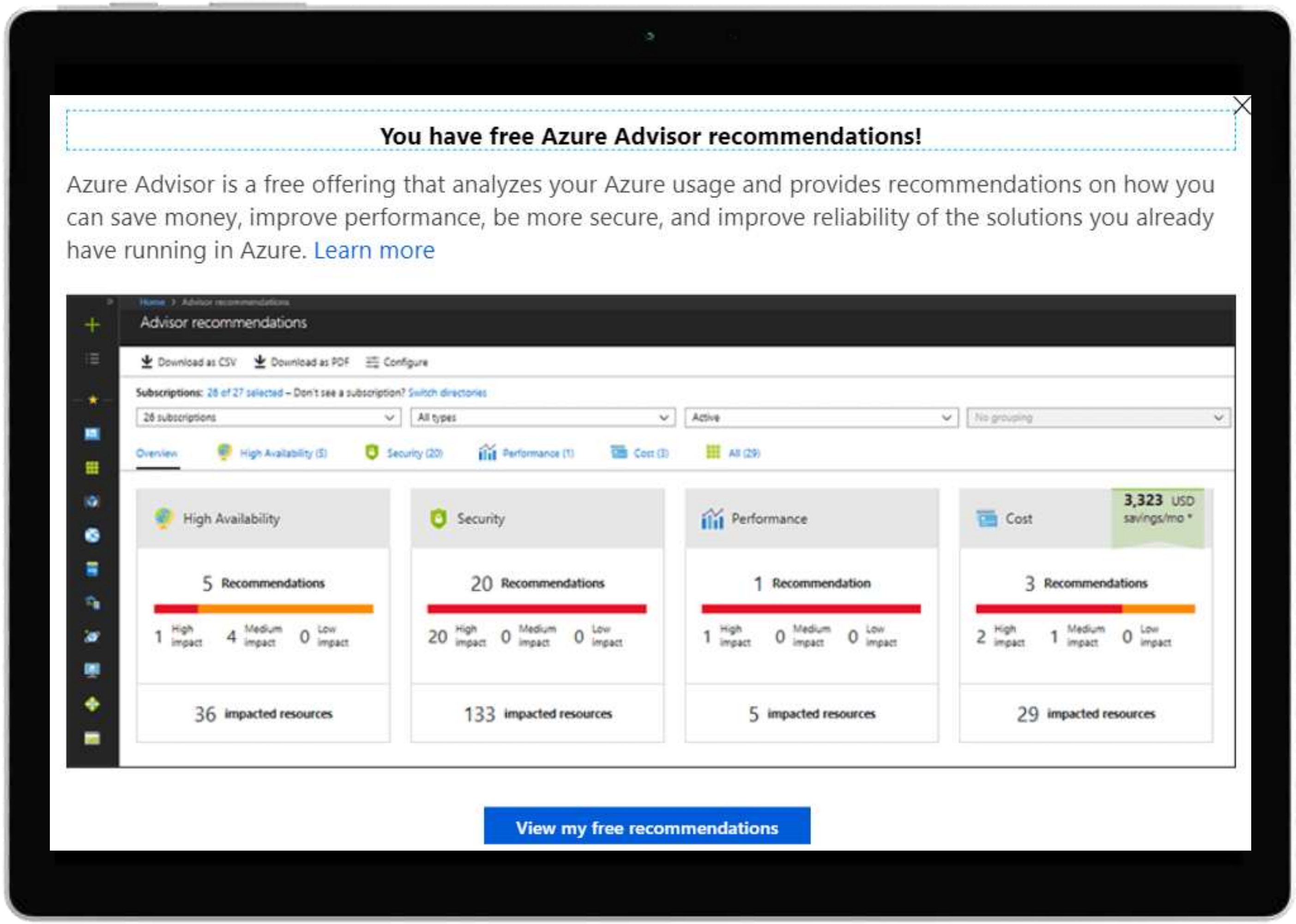
Provision additional capacity

Tempdb Contention

Scale or update user resource class

Suboptimal Plan Selection

Create or update table statistics



You have free Azure Advisor recommendations!

Azure Advisor is a free offering that analyzes your Azure usage and provides recommendations on how you can save money, improve performance, be more secure, and improve reliability of the solutions you already have running in Azure. Learn more

Advisor recommendations

Subscriptions: 28 of 27 selected – Don't see a subscription? Switch directories

High Availability: 5 Recommendations; 1 High impact, 4 Medium impact, 0 Low impact; 36 impacted resources

Security: 20 Recommendations; 20 High impact, 0 Medium impact, 0 Low impact; 133 impacted resources

Performance: 1 Recommendation; 1 High impact, 0 Medium impact, 0 Low impact; 5 impacted resources

Cost: 3,323 USD savings/mo *; 3 Recommendations; 2 High impact, 1 Medium impact, 0 Low impact; 29 impacted resources

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.

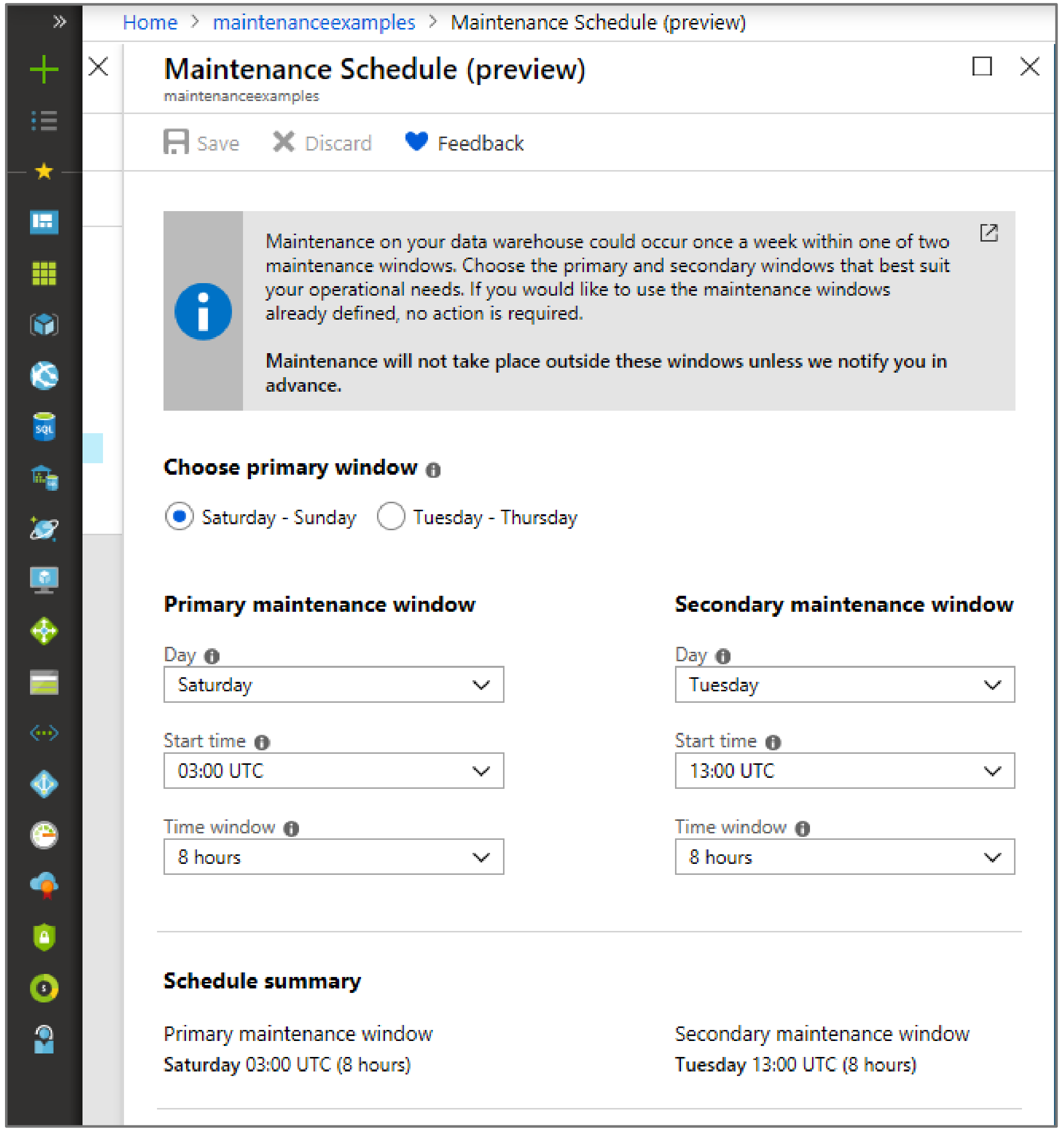
24-hour advance notification for maintenance events.

Benefits

Ensure upgrades happen on your schedule.

Predictable planning for long-running jobs.

Stay informed of start and end of maintenance.



Maintenance Schedule (preview)

maintenanceexamples

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

Day: Saturday

Start time: 03:00 UTC

Time window: 8 hours

Secondary maintenance window

Day: Tuesday

Start time: 13:00 UTC

Time window: 8 hours

Schedule summary

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

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

Automatic statistics management

Overview

Statistics are automatically created and maintained for 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

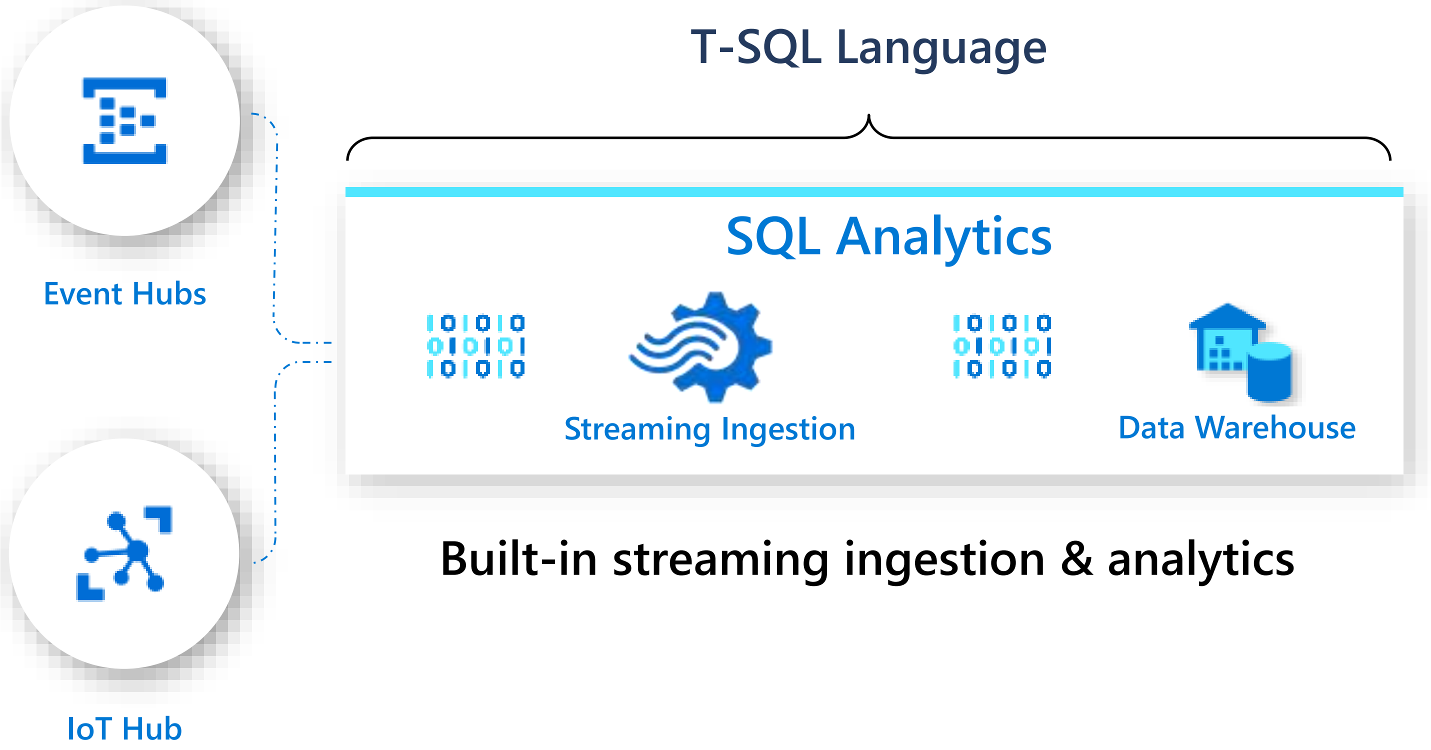
```
-- Turn on/off auto-create statistics settings
ALTER DATABASE {database_name}
SET AUTO_CREATE_STATISTICS { ON | OFF }

-- 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
```

Scope: **Private Preview (whitelisting needed)**

Heterogenous Data Preparation & Ingestion

Native SQL Streaming

- High throughput ingestion (up to 200MB/sec)
- Delivery latencies in seconds
- Ingestion throughput scales with compute scale
- Analytics capabilities (SQL-based queries for joins, aggregations, filters)
- *Removes the need to use Spark for streaming*



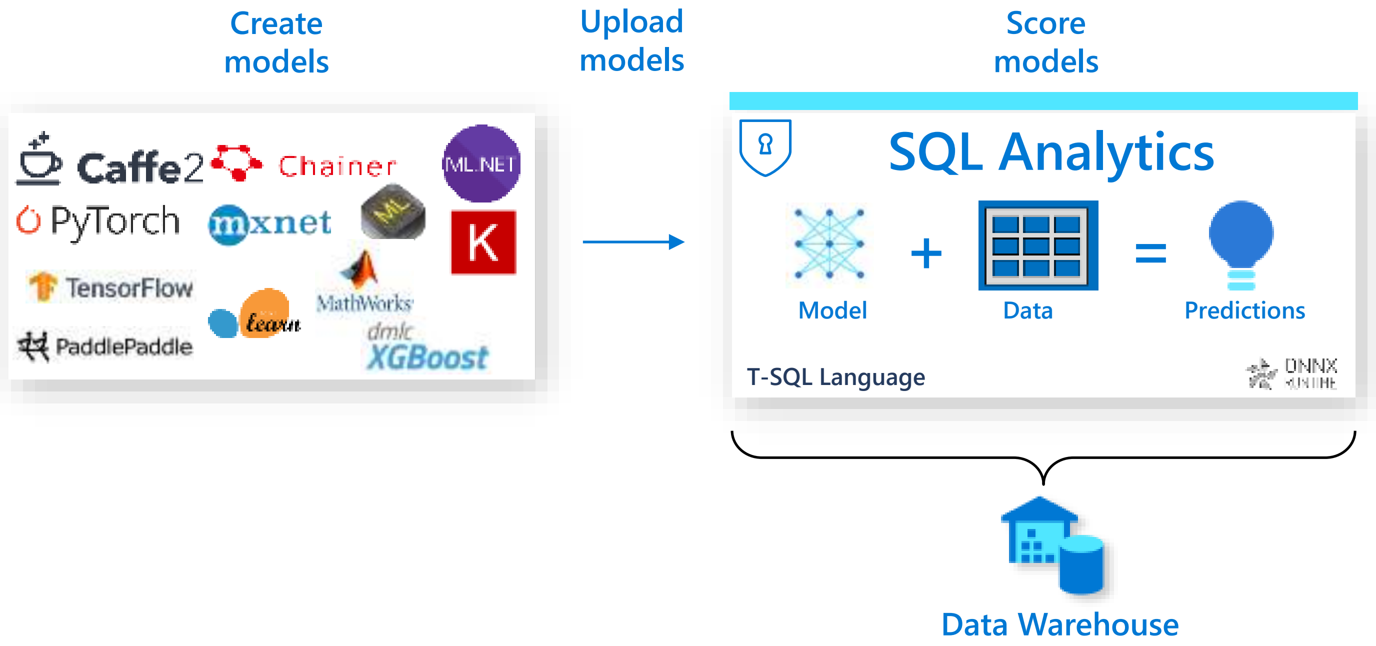
Built-in streaming ingestion & analytics

Scope: **Private Preview (whitelisting needed)**

Machine Learning enabled DW

Native PREDICT-ion

- T-SQL based experience (interactive./batch scoring)
- Interoperability with other models built elsewhere
- Execute scoring where the data lives



```
--T-SQL syntax for scoring data in SQL DW
SELECT d.*, p.Score
FROM PREDICT(MODEL = @onnx_model, DATA =
dbo.mytable AS d)
WITH (Score float) AS p;
```

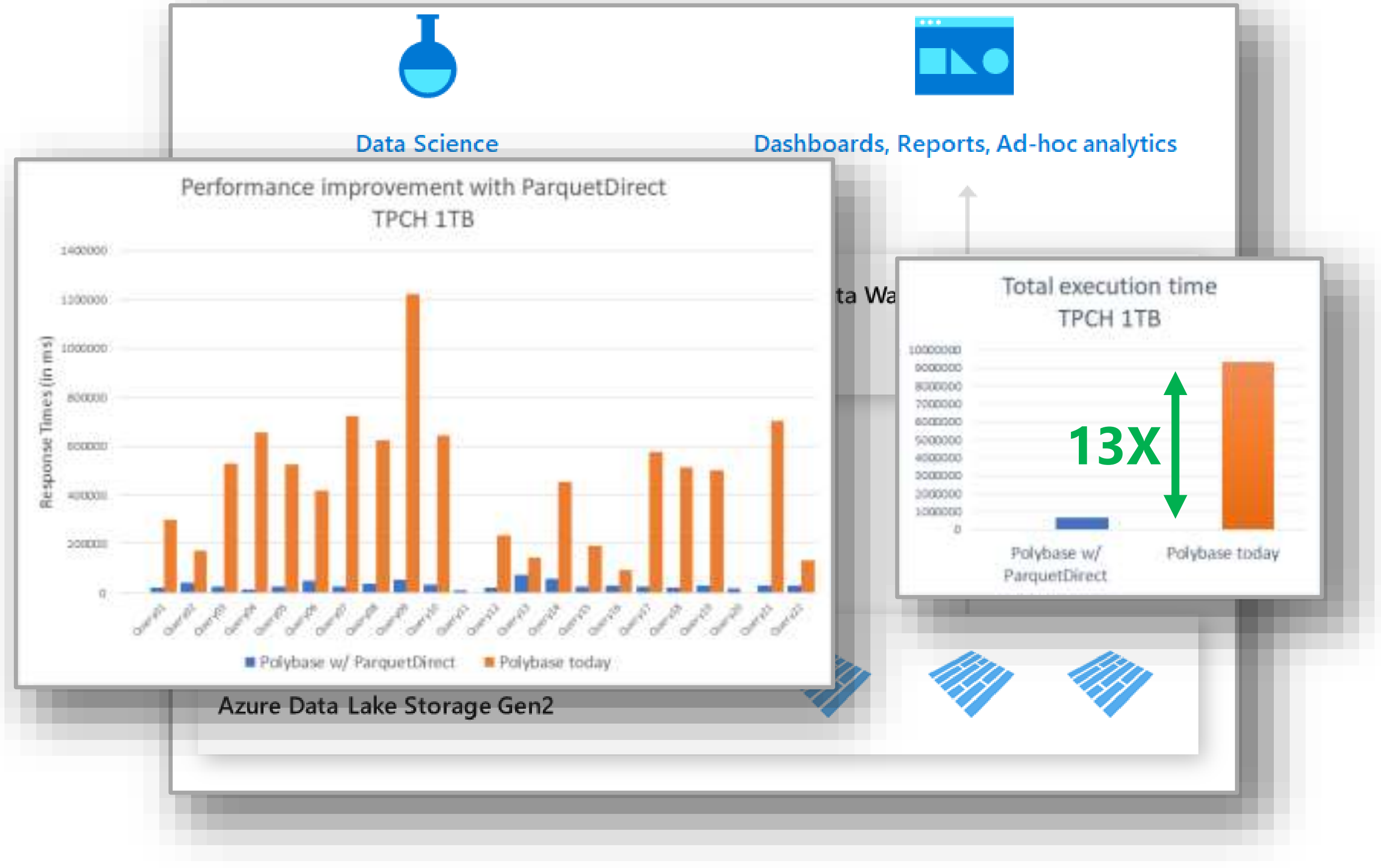
Scope: **Private Preview (whitelisting needed)**

Data Lake Integration

ParquetDirect for interactive data lake exploration

- >10X performance improvement
- Full columnar optimizations (optimizer, batch)
- Built-in transparent caching (SSD, in-memory, resultset)

SQL Analytics



Scope: **Generally Available**

Azure Data Share

Enterprise data sharing

- Share from DW to DW/DB/other systems
- Choose data format to receive data in (CSV, Parquet)
- One to many data sharing
- Share a single or multiple datasets

Any Azure Data Sources

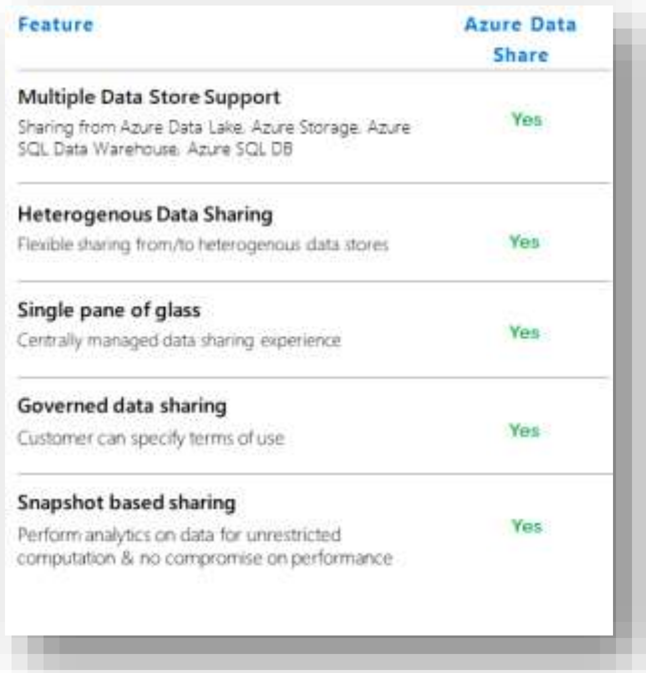
Share data from any Azure regions and data stores

Single Pane of Glass

Manage and monitor data sharing with multiple organizations

Rich Analytics Tools

Use Azure analytics tools to prepare data and derive insights



Feature	Azure Data Share
Multiple Data Store Support Sharing from Azure Data Lake, Azure Storage, Azure SQL Data Warehouse, Azure SQL DB	Yes
Heterogenous Data Sharing Flexible sharing from/to heterogenous data stores	Yes
Single pane of glass Centrally managed data sharing experience	Yes
Governed data sharing Customer can specify terms of use	Yes
Snapshot based sharing Perform analytics on data for unrestricted computation & no compromise on performance	Yes

Governance

Control data access governed by enterprise policies

Monetization

Charge for data or cost of data curation and access

SQL Analytics

new features available

GA features:

- **Performance:** Resultset caching
- **Performance:** Materialized Views
- **Performance:** Ordered columnstore
- **Heterogeneous data:** JSON support
- **Trustworthy compution:** Dynamic Data Masking
- **Continuous integration & deployment:** SSDT support
- **Language:** Read committed snapshot isolation

Public preview features:

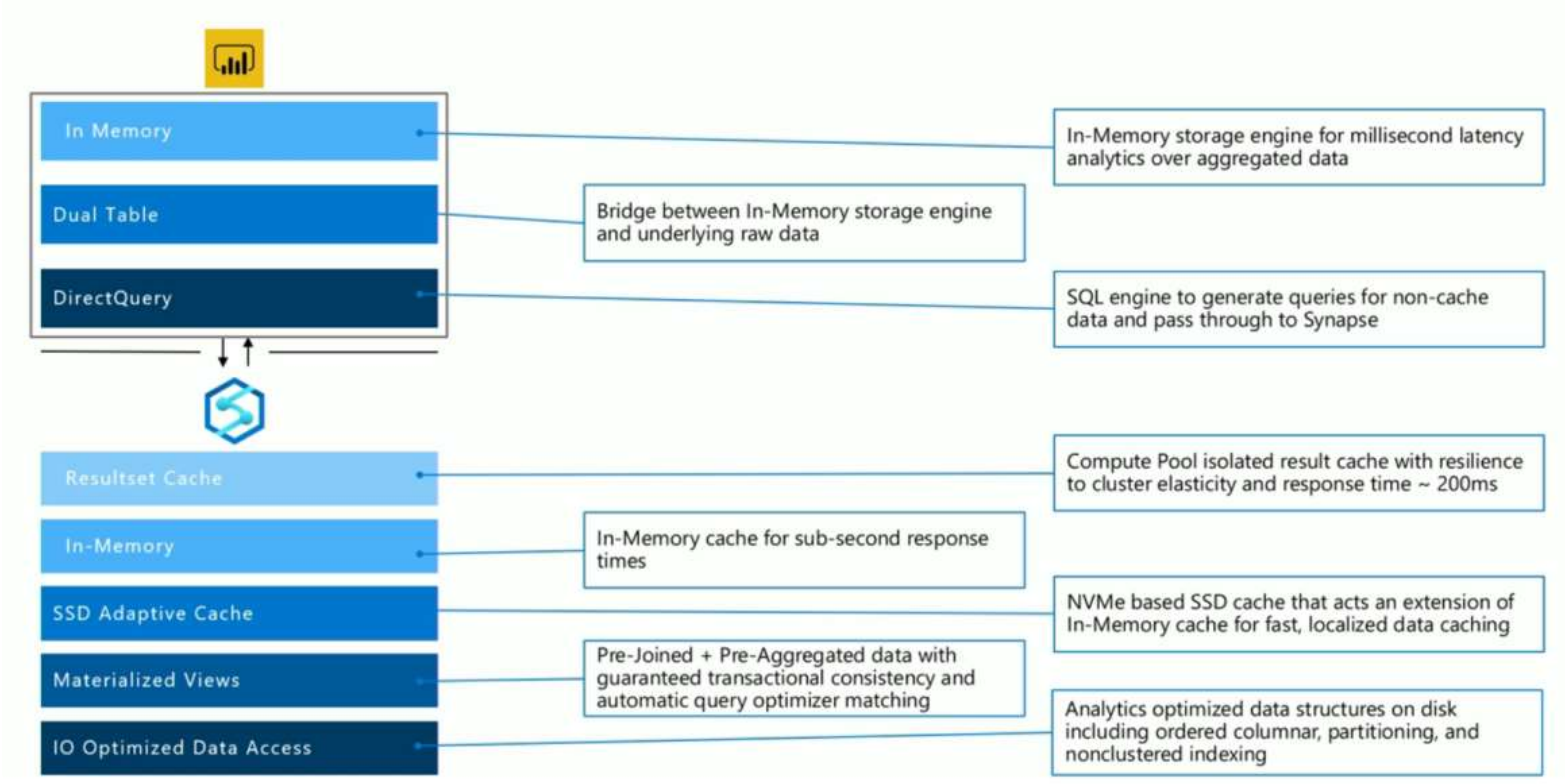
- **Workload management:** Workload Isolation
- **Data ingestion:** Simple ingestion with COPY
- **Data Sharing:** Share DW data with Azure Data Share
- **Trustworthy computation:** Private LINK support

Private preview features:

- **Data ingestion:** Streaming ingestion & analytics in DW
- **Built-in ML:** Native Prediction/Scoring
- **Data lake enabled:** Fast query over Parquet files
- **Language:** Updateable distribution column
- **Language:** FROM clause with joins
- **Language:** Multi-column distribution support
- **Security:** Column-level Encryption

Note: private preview features require whitelisting

Power BI Aggregations and Synapse query performance



In Memory

In-Memory storage engine for millisecond latency analytics over aggregated data

Dual Table

Bridge between In-Memory storage engine and underlying raw data

DirectQuery

SQL engine to generate queries for non-cache data and pass through to Synapse

Resultset Cache

Compute Pool isolated result cache with resilience to cluster elasticity and response time ~ 200ms

In-Memory

In-Memory cache for sub-second response times

SSD Adaptive Cache

NVMe based SSD cache that acts an extension of In-Memory cache for fast, localized data caching

Materialized Views

Pre-Joined + Pre-Aggregated data with guaranteed transactional consistency and automatic query optimizer matching

IO Optimized Data Access

Analytics optimized data structures on disk including ordered columnar, partitioning, and nonclustered indexing

Microsoft

Azure Synapse Analytics

SQL On-Demand

Query Options

1. Provisioned SQL over relational database – Traditional SQL DW [existing]
2. Provisioned SQL over ADLS Gen2 – via external tables or openrowset [existing via PolyBase]
3. On-demand SQL over relational database - dependency on the flexible data model (data cells) over columnstore data (preview) [new]
4. On-demand SQL over ADLS Gen2 – via external tables or openrowset [new]
5. Provisioned Spark over relational database – Not possible
6. Provisioned Spark over ADLS Gen2 [new]
7. On-demand Spark over relational database - On-demand Spark is not supported
8. On-demand Spark over ADLS Gen2 – On-demand Spark is not supported

Notes:

- Separation of state (data, metadata and transactional logs) and compute
- Queries against data loaded into SQL Analytics tables are faster 2-3X compared to queries over external tables
- Improved performance compared to PolyBase. PolyBase is not used, but functional aspects are supported
- SQL on-demand will push down queries from the front-end to back-end nodes
- Warm-up for first on-demand query takes about 20-25 seconds
- If you create a Spark Table, that table will be created as an external table in SQL Pool or On-Demand without having to keep a Spark cluster up and running

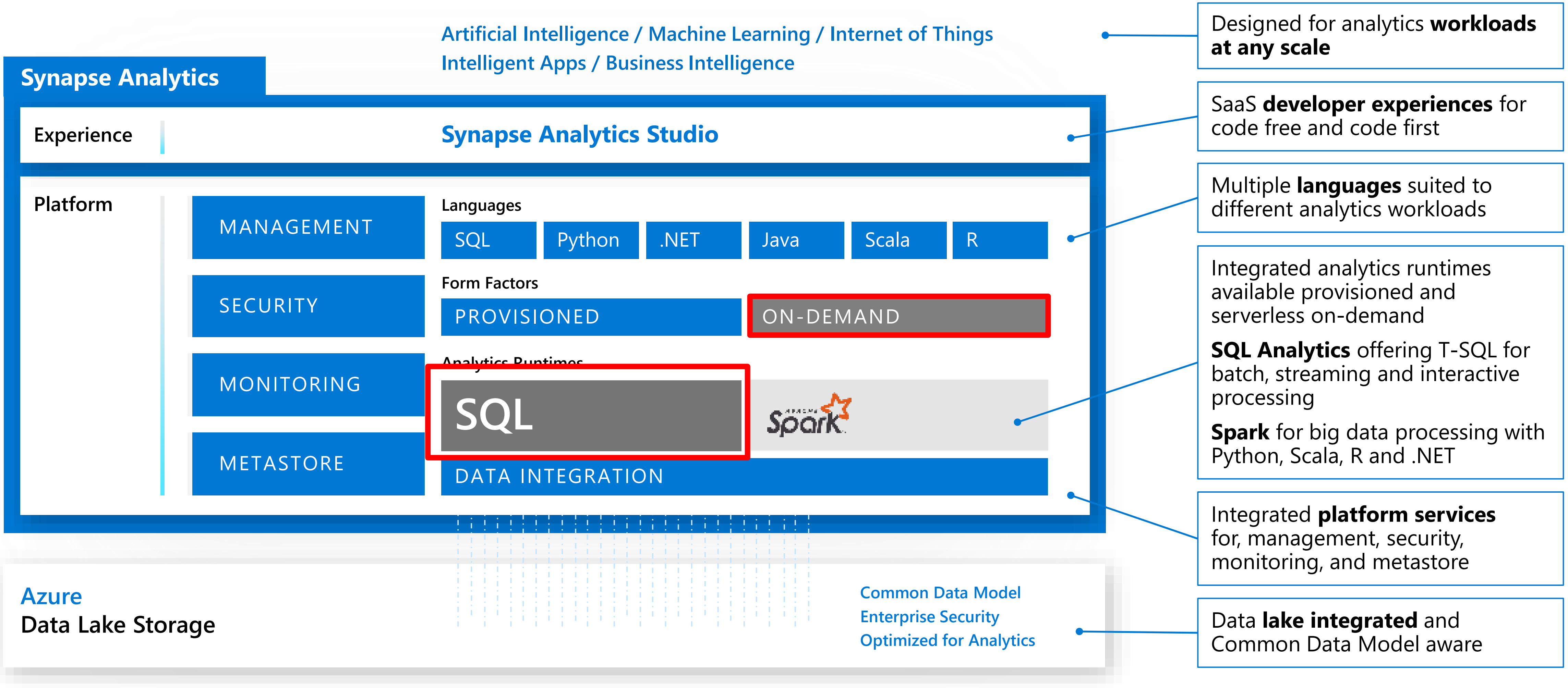
Distributed Query Processor (DQP)

- **Auto-scale compute nodes** - Instruct the underlying fabric the need for more compute power to adjust to peaks during the workload. If compute power is granted, the Polaris DQP will re-distribute tasks leveraging the new compute container. Note that in-flight tasks in the previous topology continue running, while new queries get the new compute power with the new re-balancing
- **Compute node fault tolerance** - Recover from faulty nodes while a query is running. If a node fails the DQP re-schedules the tasks in the faulted node through the remainder of the healthy topology
- **Compute node hot spot: rebalance queries or scale out nodes** - Can detect hot spots in the existing topology. That is, overloaded compute nodes due to data skew. In the advent of a compute node running hot because of skewed tasks, the DQP can decide to re-schedule some of the tasks assigned to that compute node amongst others where the load is less
- **Multi-cluster** - Multiple compute pools accessing the same data
- **Cross-database queries** – A query can specify multiple databases

These features work for both on-demand and provisioned over ADLS Gen2 and relational databases

Azure Synapse Analytics

Integrated data platform for BI, AI and continuous intelligence



Artificial Intelligence / Machine Learning / Internet of Things
Intelligent Apps / Business Intelligence

Synapse Analytics

Experience — Synapse Analytics Studio

Platform

MANAGEMENT
SECURITY
MONITORING
METASTORE

Languages: SQL, Python, .NET, Java, Scala, R

Form Factors: PROVISIONED, ON-DEMAND

Analytics Runtimes: SQL, Spark

DATA INTEGRATION

Azure
Data Lake Storage

Common Data Model
Enterprise Security
Optimized for Analytics

Designed for analytics **workloads at any scale**

SaaS **developer experiences** for code free and code first

Multiple **languages** suited to different analytics workloads

Integrated analytics runtimes available provisioned and serverless on-demand

SQL Analytics offering T-SQL for batch, streaming and interactive processing

Spark for big data processing with Python, Scala, R and .NET

Integrated **platform services** for, management, security, monitoring, and metastore

Data **lake integrated** and Common Data Model aware

Synapse SQL on-demand scenarios

Discovery and exploration	What’s in this file? How many rows are there? What’s the max value? **SQL On-demand reduces data lake exploration to the right-click!**
Data transformation	How to convert CSVs to Parquet quickly? How to transform the raw data? **Use the full power of T-SQL to transform the data in the data lake**

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

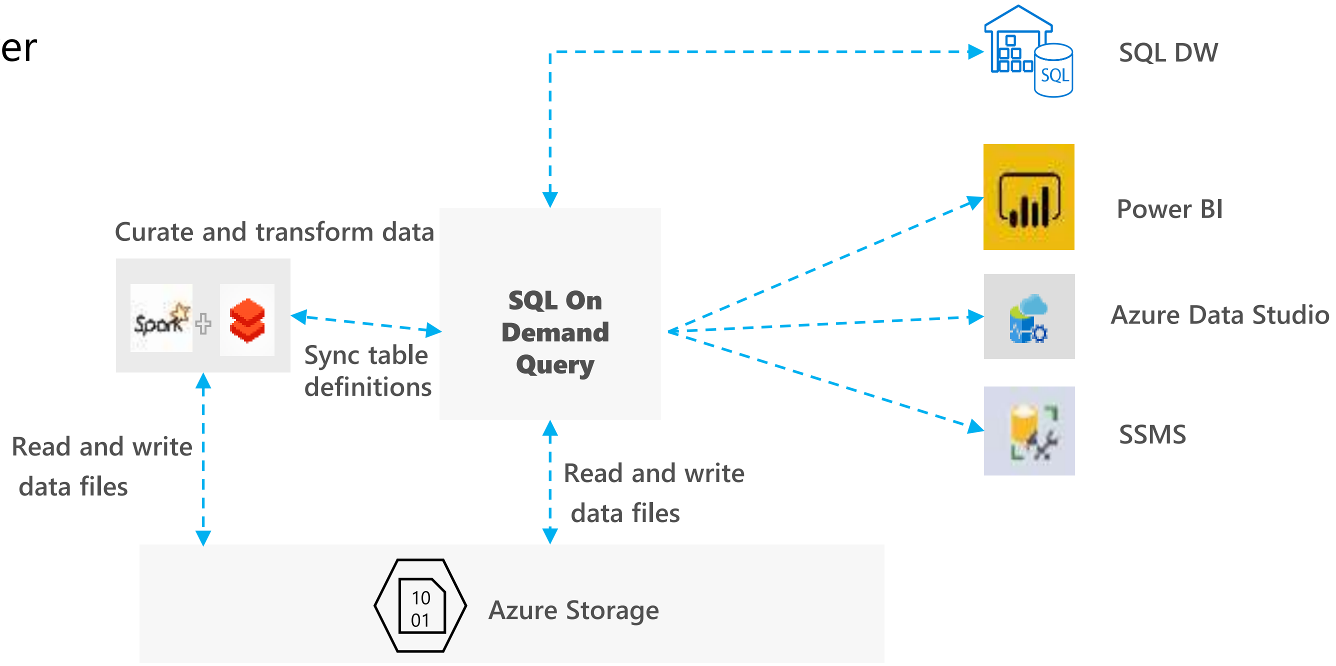
Offers security

Data integration with Databricks, HDInsight

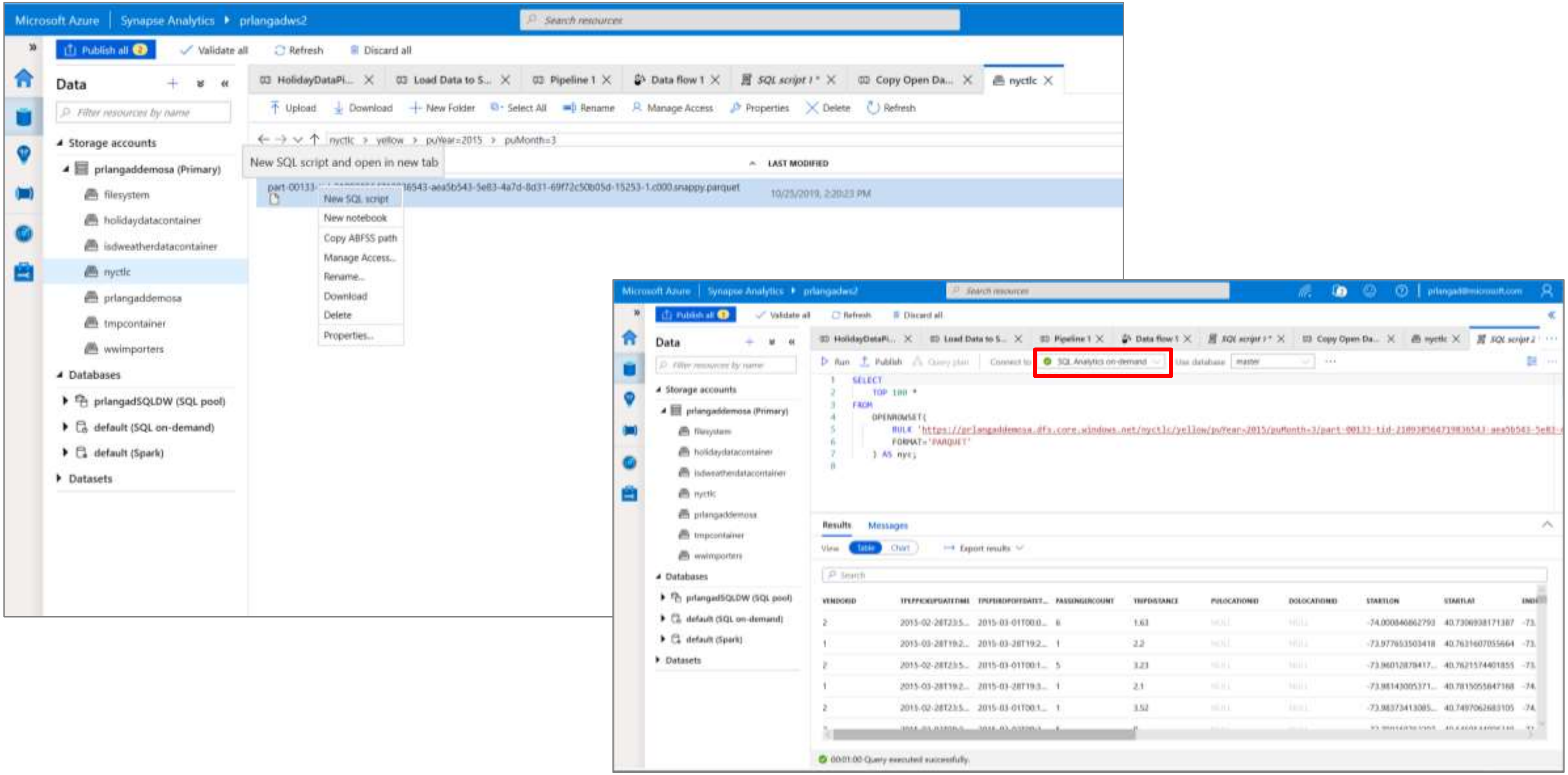
T-SQL syntax to query data

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

Support for BI ecosystem



SQL On Demand – Querying on storage



```
SELECT
    TOP 100 *
FROM
    OPENROWSET(
        BULK 'https://prlangaddemosa.dfs.core.windows.net/nyctlc/yellow/puYear=2015/puMonth=3/part-00133-tid-210938564719836543-aea5b543-5e83-
        FORMAT='PARQUET'
    ) AS nyc;
```

VENDORID	TPEPPICKUPDATETIME	TPEPDROPOFFDATET...	PASSENGERCOUNT	TRIPDISTANCE	PULOCATIONID	DOLOCATIONID	STARTLON	STARTLAT	END
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.

00:01:00 Query executed successfully.

SQL On Demand – Querying CSV File

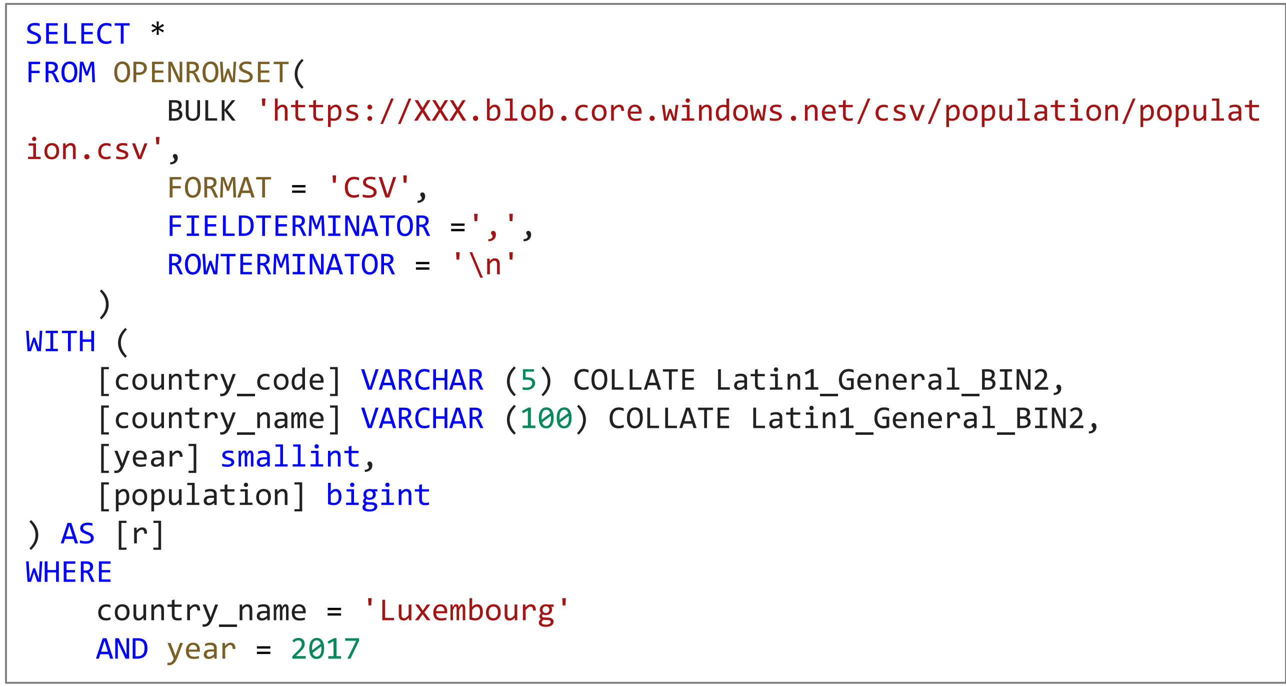
Overview

Uses OPENROWSET function to access data

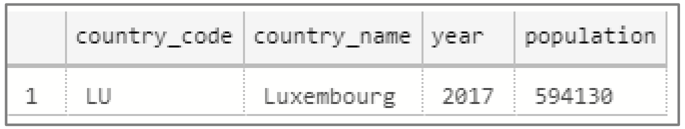
Benefits

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



```sql
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
```

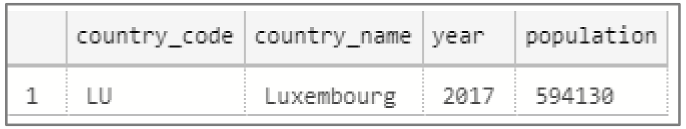


	country_code	country_name	year	population
1	LU	Luxembourg	2017	594130

SQL On Demand – Querying CSV File

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

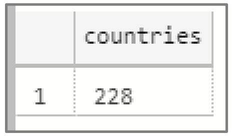
```
SELECT *
FROM OPENROWSET(
        BULK 'https://XXX.blob.core.windows.net/csv/population-
unix-hdr/population.csv',
        FORMAT = 'CSV',
        FIELDTERMINATOR =',',
        ROWTERMINATOR = '0x0a',
        FIRSTROW = 2
    )
    WITH (
        [country_code] VARCHAR (5) COLLATE Latin1_General_BIN2,
        [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/popul
ation/population.csv',
        FORMAT = 'CSV',
        FIELDTERMINATOR =',',
        ROWTERMINATOR = '\n'
    )
WITH (
        [country_name] VARCHAR (100) COLLATE Latin1_Gener
al_BIN2 2
) AS [r]
```



	countries
1	228

SQL On Demand – Querying folders

Overview

Uses OPENROWSET function to access data from multiple files or folders

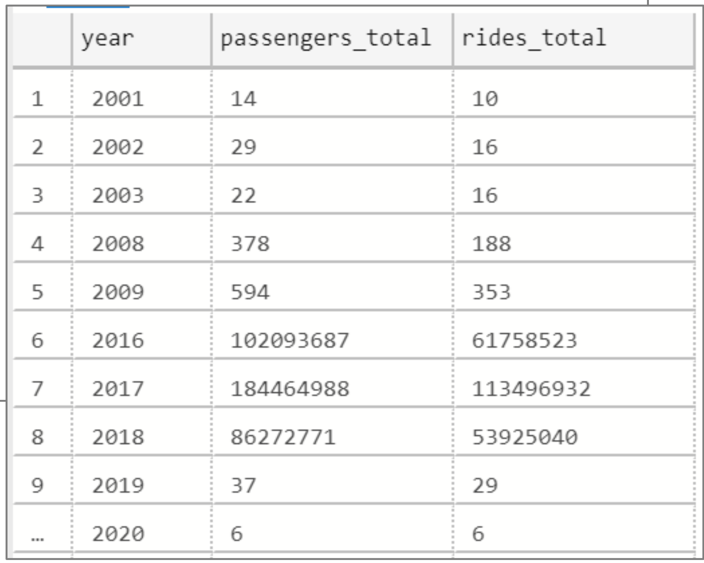
Benefits

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/*.*’,
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)
```

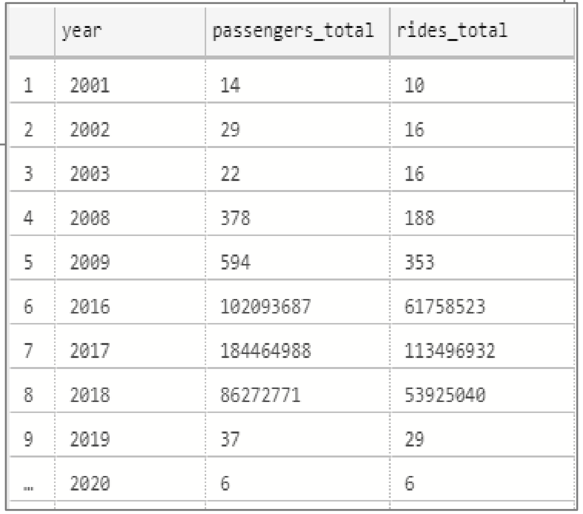


	year	passengers_total	rides_total
1	2001	14	10
2	2002	29	16
3	2003	22	16
4	2008	378	188
5	2009	594	353
6	2016	102093687	61758523
7	2017	184464988	113496932
8	2018	86272771	53925040
9	2019	37	29
…	2020	6	6

SQL On Demand – Querying folders

Read all files from multiple folders

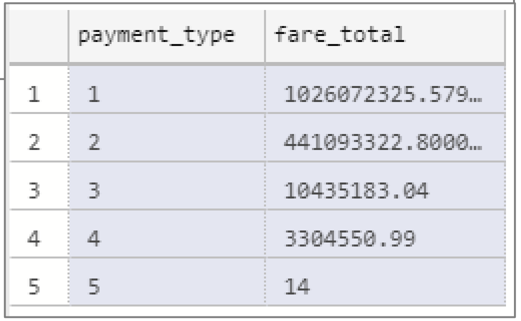
```
SELECT YEAR(pickup_datetime) as [year],
    SUM(passenger_count) AS passengers_total,
    COUNT(*) AS [rides_total]
FROM OPENROWSET(
    BULK 'https://XXX.blob.core.windows.net/csv/t*i/',
        FORMAT = 'CSV',
        FIRSTROW = 2    )
    WITH (
        vendor_id VARCHAR(100) COLLATE Latin1_General_BIN2,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



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

Read subset of files in folder

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



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

SQL On Demand – Querying specific files

Overview

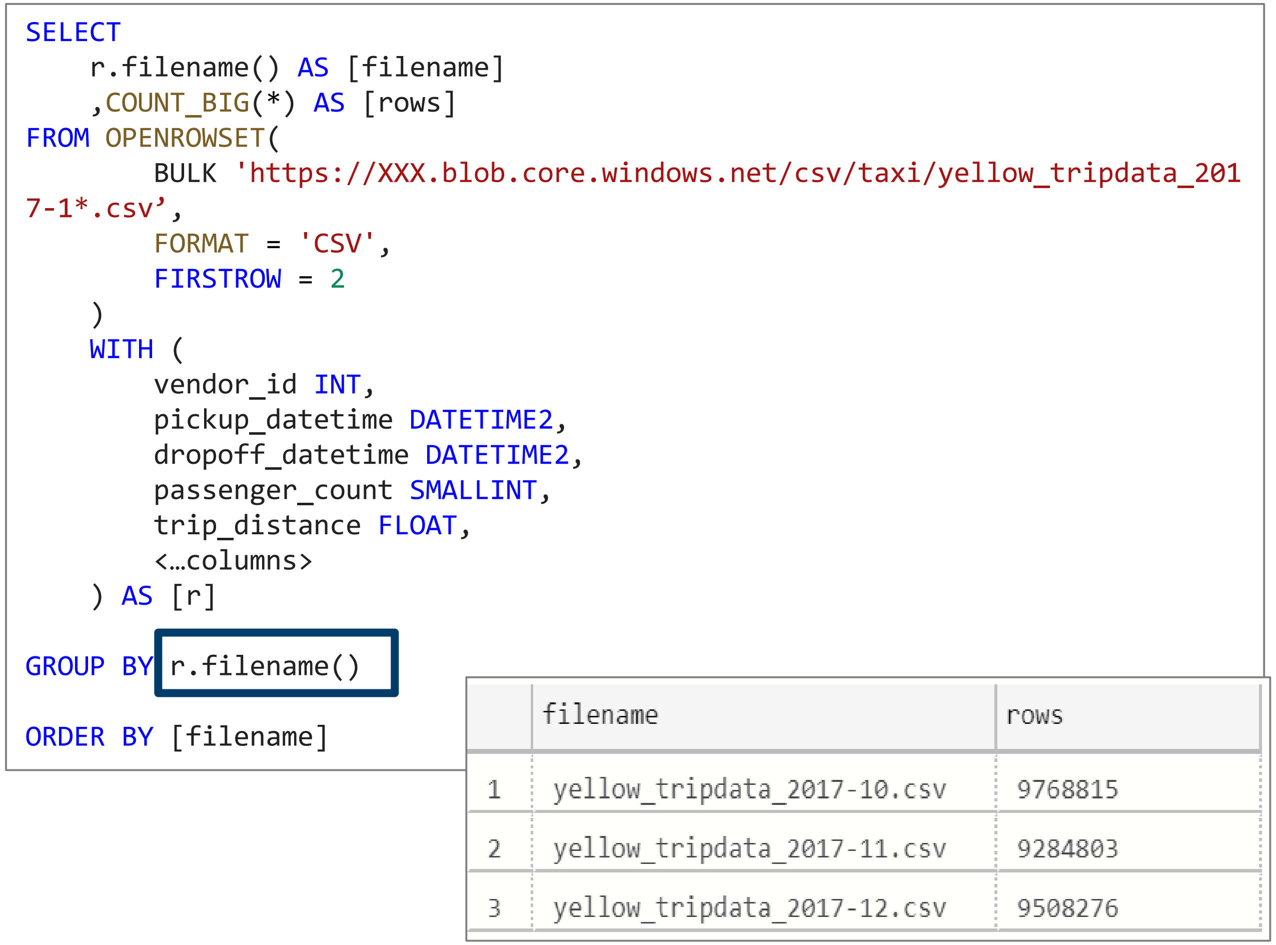
filename – Provides file name that originates row result

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



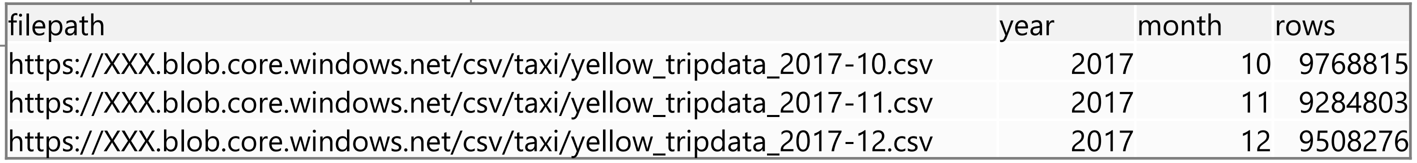
```
SELECT
    r.filename() AS [filename]
    ,COUNT_BIG(*) AS [rows]
FROM OPENROWSET(
        BULK 'https://XXX.blob.core.windows.net/csv/taxi/yellow_tripdata_201
7-1*.csv’,
        FORMAT = 'CSV',
        FIRSTROW = 2
    )
    WITH (
        vendor_id INT,
        pickup_datetime DATETIME2,
        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

```sql
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

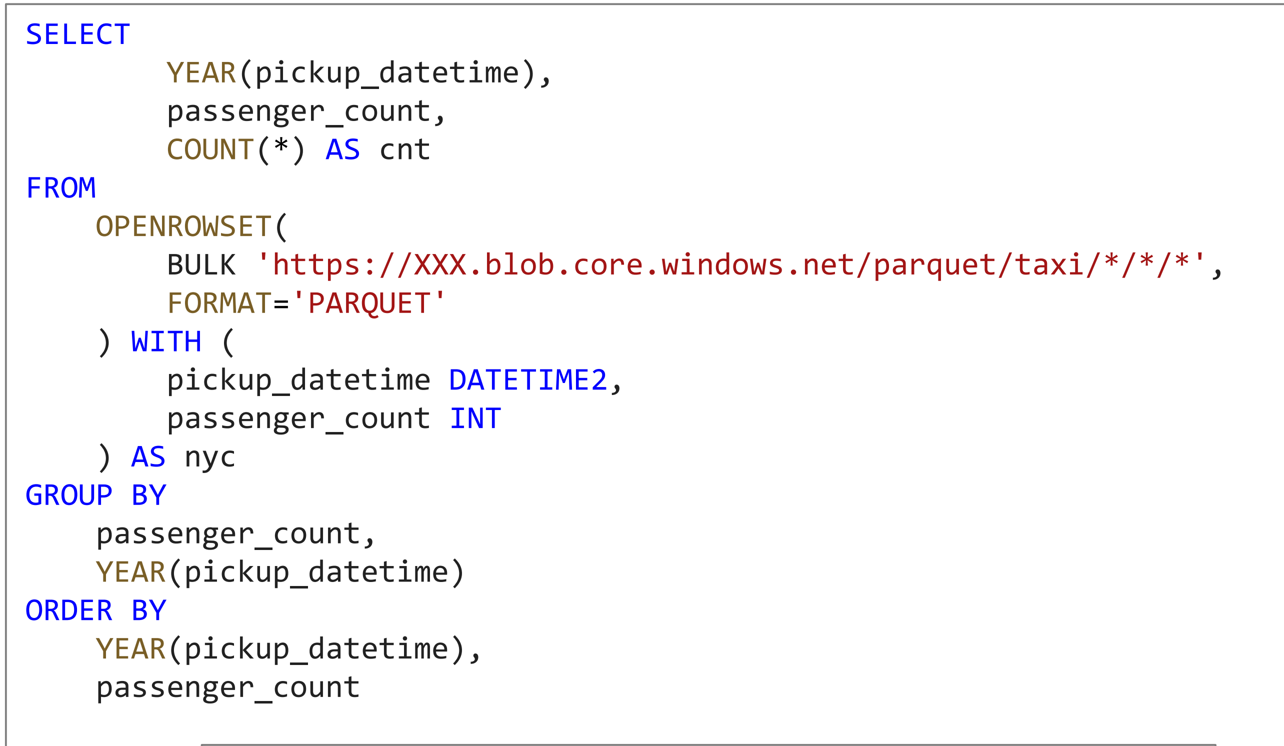
Uses OPENROWSET function to access data

Benefits

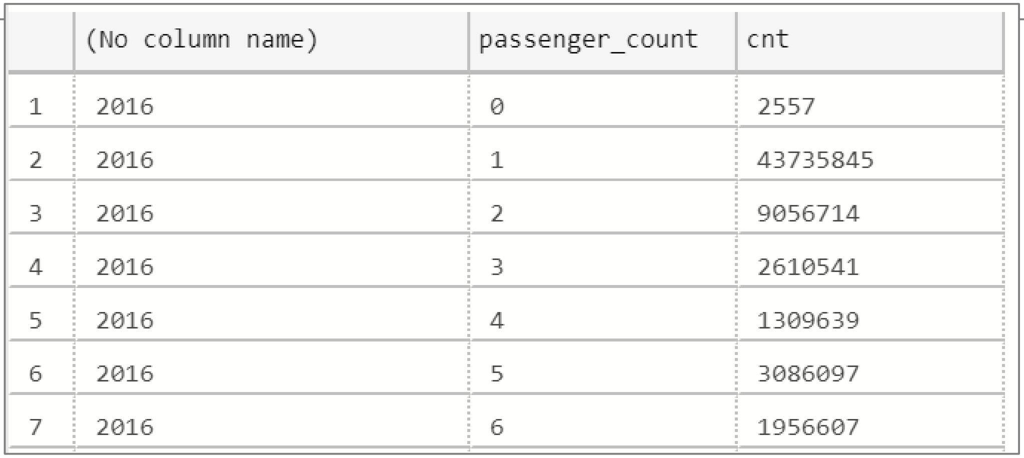
Ability to specify column names of interest

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

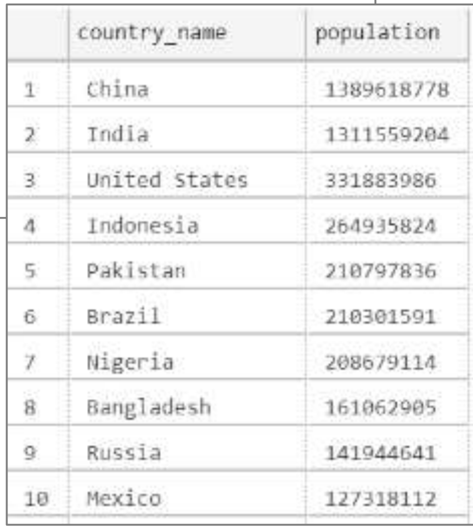
Create views using SQL On Demand queries

Benefits

Works same as standard views

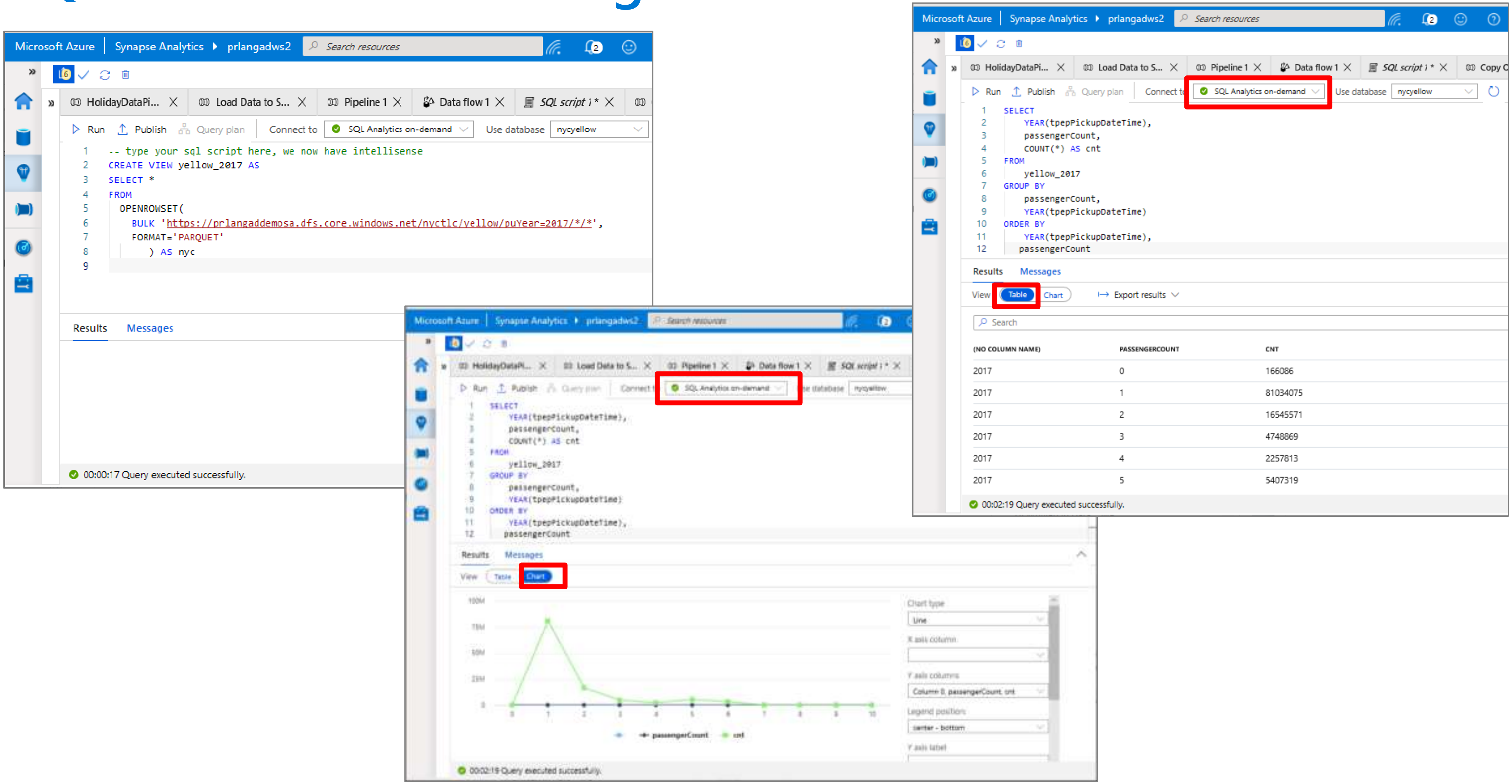
```
USE [mydbname]
GO

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



Connect to: SQL Analytics on-demand

Use database: nycyellow

```sql
-- 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
```

00:00:17 Query executed successfully.

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

View: Table | Chart

Chart type: Line

X axis column:

Y axis columns: Column 0, passengerCount, cnt

Legend position: center - bottom

Y axis label

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

00:02:19 Query executed successfully.

SQL On Demand – Querying JSON files

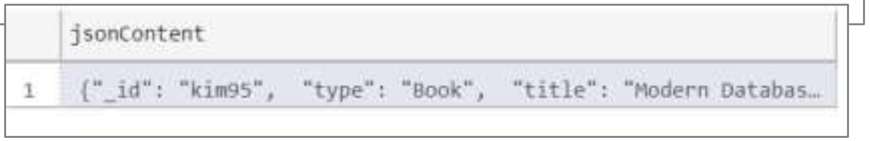
Overview

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/book
1.json’,
        FORMAT='CSV',
        FIELDTERMINATOR ='0x0b',
        FIELDQUOTE = '0x0b',
        ROWTERMINATOR = '0x0b'
    )
    WITH (
        jsonContent varchar(8000)
    ) AS [r]
```

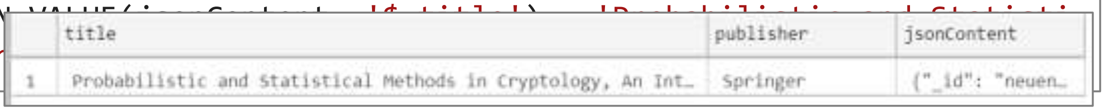


	jsonContent
1	{"_id": "kim95", "type": "Book", "title": "Modern Databas...

SQL On Demand – Querying JSON files

Example of JSON_VALUE function

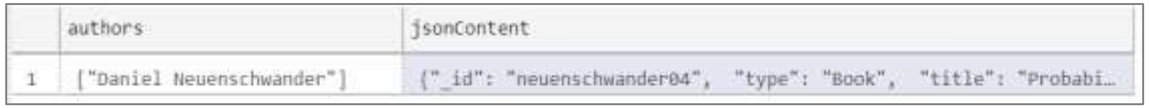
```
SELECT
    JSON_VALUE(jsonContent, '$.title') AS title,
    JSON_VALUE(jsonContent, '$.publisher') as publisher,
    jsonContent
FROM
    OPENROWSET(
        BULK 'https://XXX.blob.core.windows.net/json/books/*.json',
        FORMAT='CSV',
        FIELDTERMINATOR ='0x0b',
        FIELDQUOTE = '0x0b',
        ROWTERMINATOR = '0x0b'
    )
    WITH (
        jsonContent varchar(8000)
    ) AS [r]
WHERE
    JSON
cal Meth
```



	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 Statist
ical 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

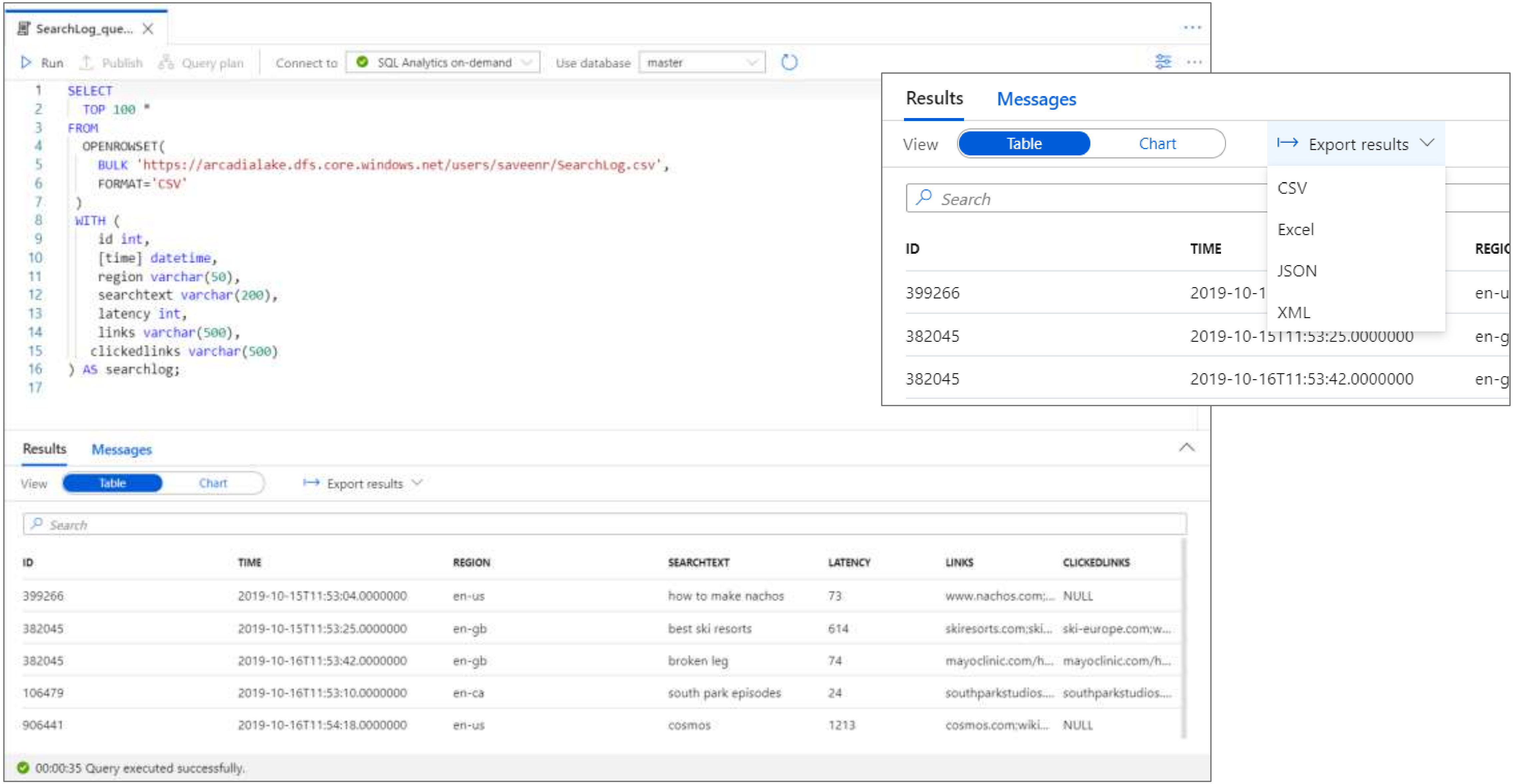
Creates an external table and then exports results of the Select statement. These operations will import data into the database for the duration of the query

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

SQL scripts > View and export results

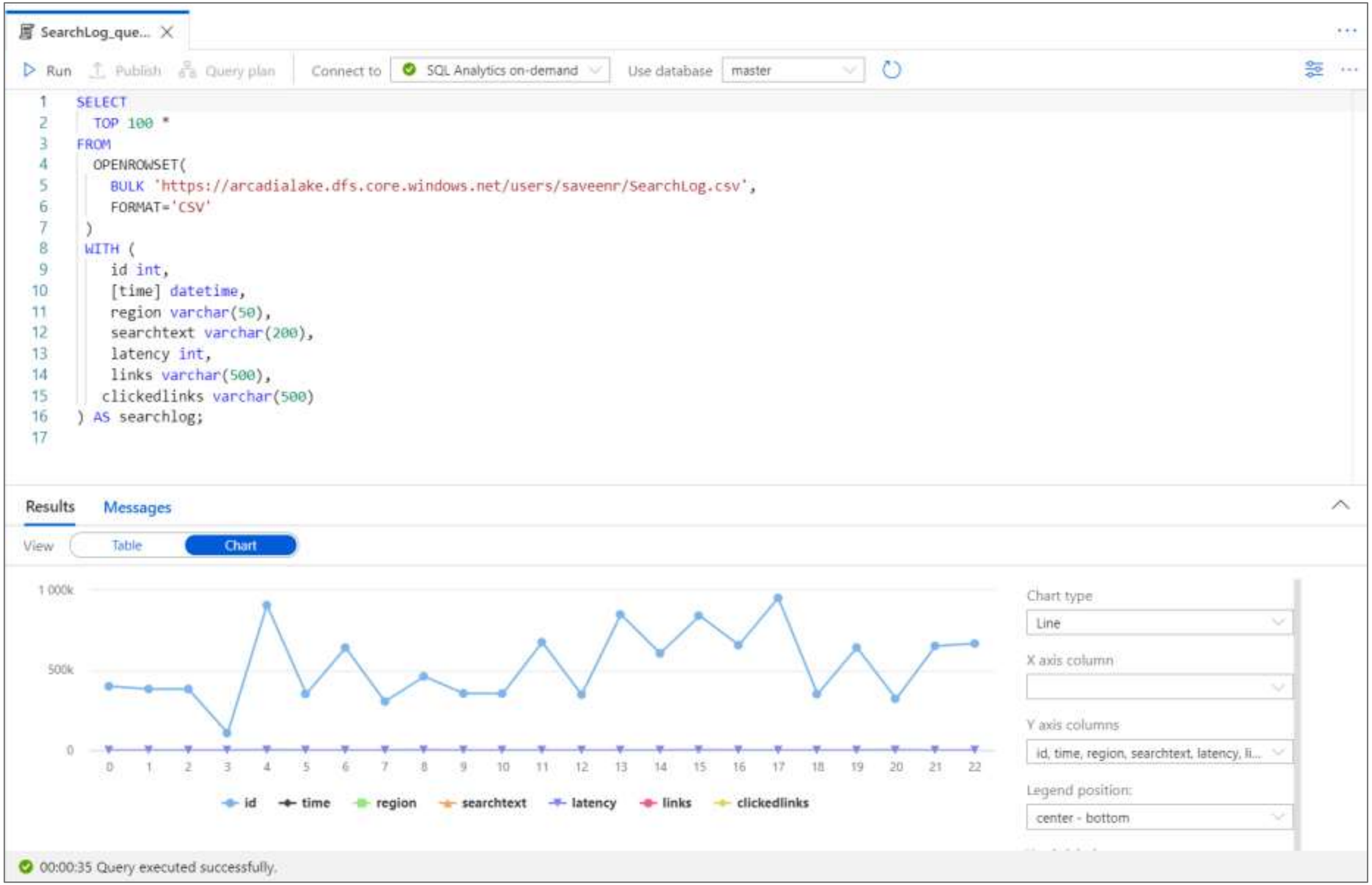


```sql
SELECT
  TOP 100 *
FROM
  OPENROWSET(
    BULK 'https://arcadialake.dfs.core.windows.net/users/saveenr/SearchLog.csv',
    FORMAT='CSV'
  )
 WITH (
    id int,
    [time] datetime,
    region varchar(50),
    searchtext varchar(200),
    latency int,
    links varchar(500),
   clickedlinks varchar(500)
) AS searchlog;
```

ID	TIME	REGION	SEARCHTEXT	LATENCY	LINKS	CLICKEDLINKS
399266	2019-10-15T11:53:04.0000000	en-us	how to make nachos	73	www.nachos.com;...	NULL
382045	2019-10-15T11:53:25.0000000	en-gb	best ski resorts	614	skiresorts.com;ski...	ski-europe.com;w...
382045	2019-10-16T11:53:42.0000000	en-gb	broken leg	74	mayoclinic.com/h...	mayoclinic.com/h...
106479	2019-10-16T11:53:10.0000000	en-ca	south park episodes	24	southparkstudios...	southparkstudios...
906441	2019-10-16T11:54:18.0000000	en-us	cosmos	1213	cosmos.com;wiki...	NULL

00:00:35 Query executed successfully.

SQL scripts > View results (chart)



Connect to: SQL Analytics on-demand

Use database: master

```sql
SELECT
  TOP 100 *
FROM
  OPENROWSET(
    BULK 'https://arcadialake.dfs.core.windows.net/users/saveenr/SearchLog.csv',
    FORMAT='CSV'
  )
 WITH (
    id int,
    [time] datetime,
    region varchar(50),
    searchtext varchar(200),
    latency int,
    links varchar(500),
   clickedlinks varchar(500)
) AS searchlog;
```

Chart type: Line

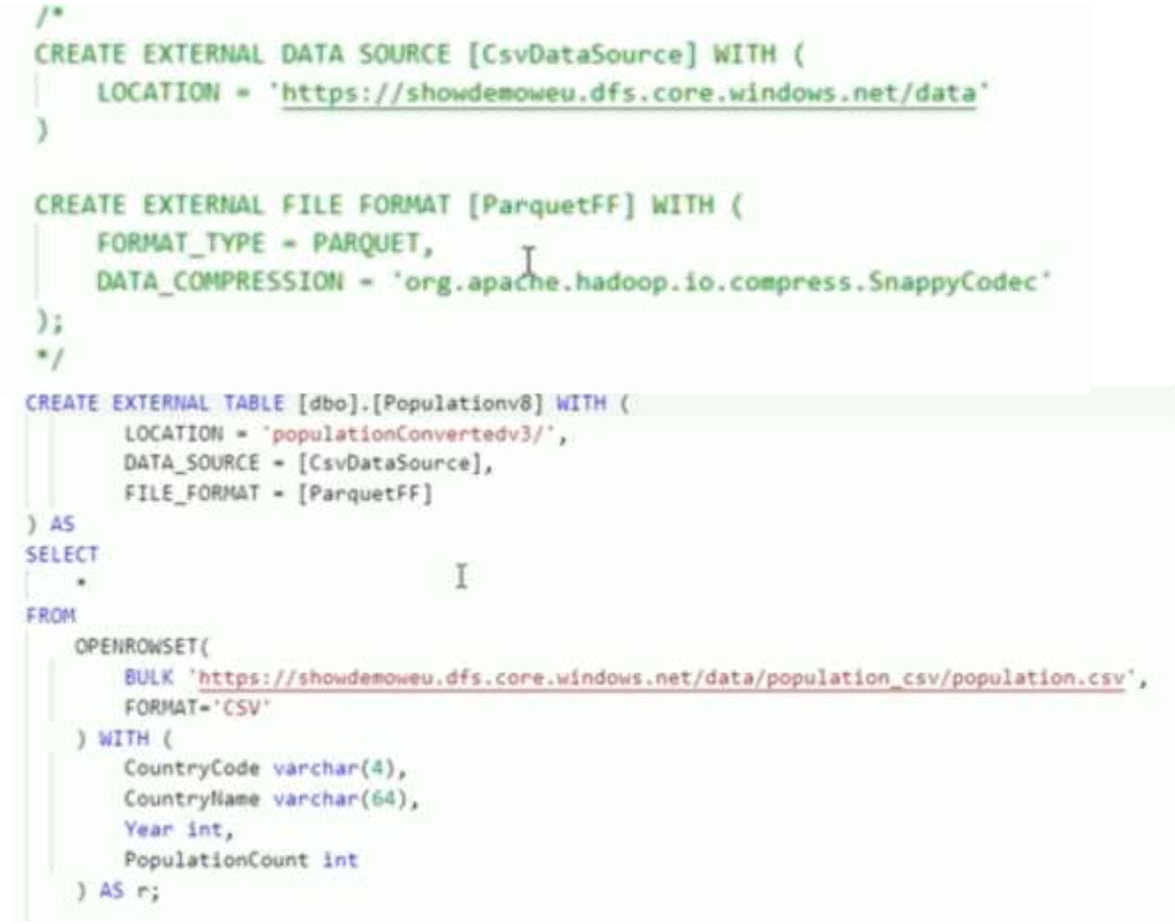
X axis column:

Y axis columns: id, time, region, searchtext, latency, li...

Legend position: center - bottom

00:00:35 Query executed successfully.

Convert from CSV to Parquet on-demand



```
/*
CREATE EXTERNAL DATA SOURCE [CsvDataSource] WITH (
    LOCATION = 'https://showdemoweu.dfs.core.windows.net/data'
)

CREATE EXTERNAL FILE FORMAT [ParquetFF] WITH (
    FORMAT_TYPE = PARQUET,
    DATA_COMPRESSION = 'org.apache.hadoop.io.compress.SnappyCodec'
);
*/
CREATE EXTERNAL TABLE [dbo].[Populationv8] WITH (
        LOCATION = 'populationConvertedv3/',
        DATA_SOURCE = [CsvDataSource],
        FILE_FORMAT = [ParquetFF]
) AS
SELECT
    *
FROM
    OPENROWSET(
        BULK 'https://showdemoweu.dfs.core.windows.net/data/population_csv/population.csv',
        FORMAT='CSV'
    ) WITH (
        CountryCode varchar(4),
        CountryName varchar(64),
        Year int,
        PopulationCount int
    ) AS r;
```

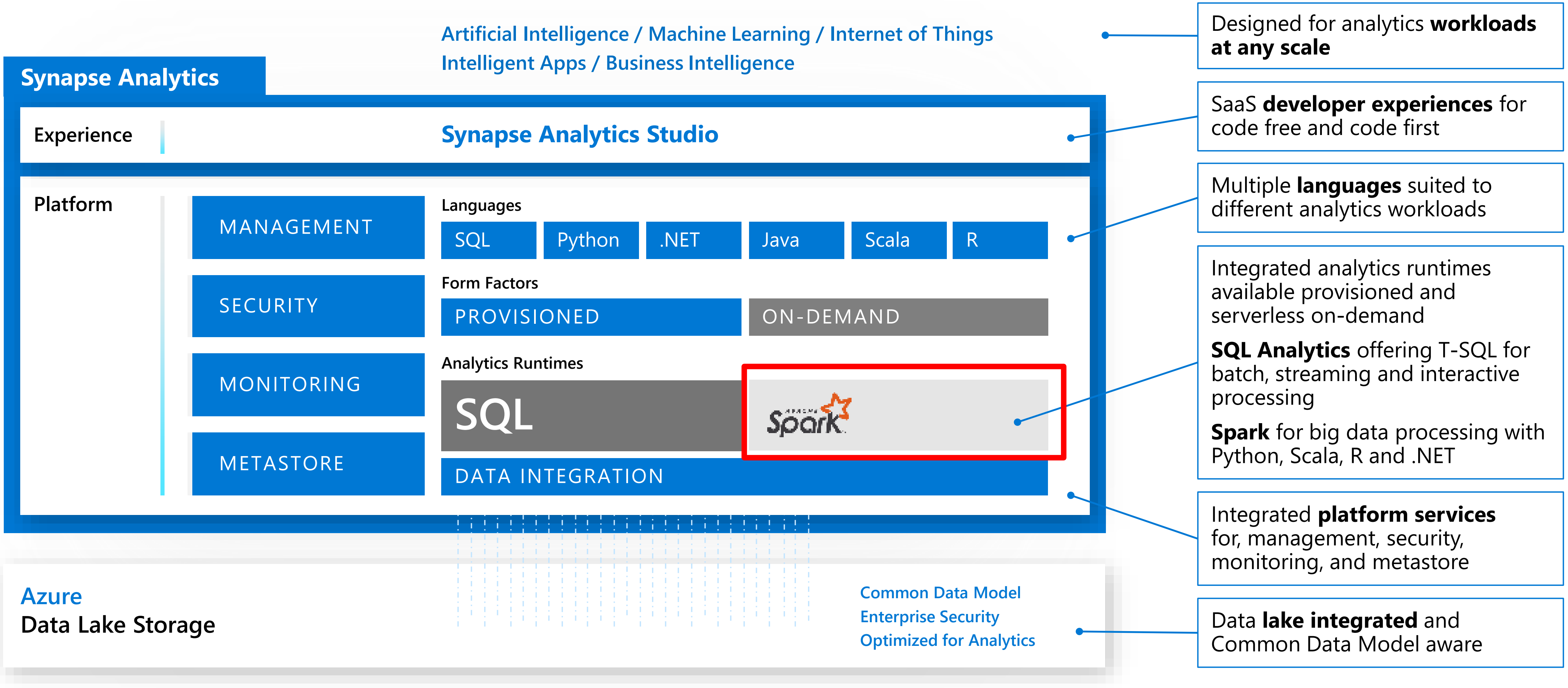
Microsoft

Azure Synapse Analytics

Spark

Azure Synapse Analytics

Integrated data platform for BI, AI and continuous intelligence



Artificial Intelligence / Machine Learning / Internet of Things
Intelligent Apps / Business Intelligence

Synapse Analytics

Experience

Synapse Analytics Studio

Platform

MANAGEMENT

SECURITY

MONITORING

METASTORE

Languages

SQL | Python | .NET | Java | Scala | R

Form Factors

PROVISIONED | ON-DEMAND

Analytics Runtimes

SQL | Spark

DATA INTEGRATION

Azure
Data Lake Storage

Common Data Model
Enterprise Security
Optimized for Analytics

Designed for analytics **workloads at any scale**

SaaS **developer experiences** for code free and code first

Multiple **languages** suited to different analytics workloads

Integrated analytics runtimes available provisioned and serverless on-demand

SQL Analytics offering T-SQL for batch, streaming and interactive processing

Spark for big data processing with Python, Scala, R and .NET

Integrated **platform services** for, management, security, monitoring, and metastore

Data **lake integrated** and Common Data Model aware

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 nteract based notebooks
  - Fast load of SQL Analytics pools

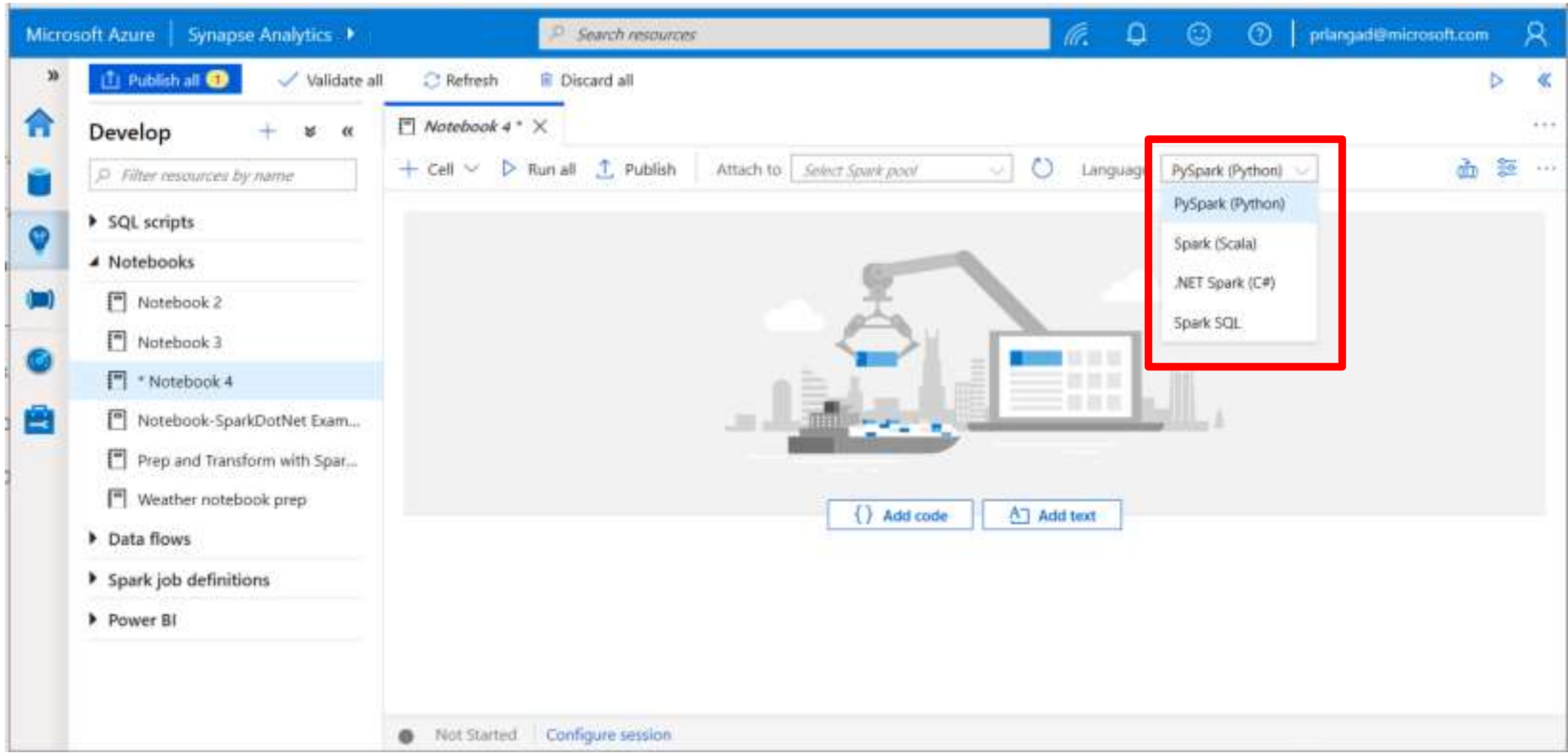
- **Core scenarios**
  - Data Prep/Data Engineering/ETL
  - Machine Learning via Spark ML and Azure ML integration
  - Extensible through library management
- **Efficient resource utilization**
  - Fast Start
  - Auto scale (up and down)
  - Auto pause
  - Min cluster size of 3 nodes
- **Multi Language Support**
  - .Net (C#), PySpark, Scala, Spark SQL, Java

Languages

Overview

Supports multiple languages to develop notebook

- PySpark (Python)
- Spark (Scala)
- .NET Spark (C#)
- Spark SQL
- Java
- R (early 2020)



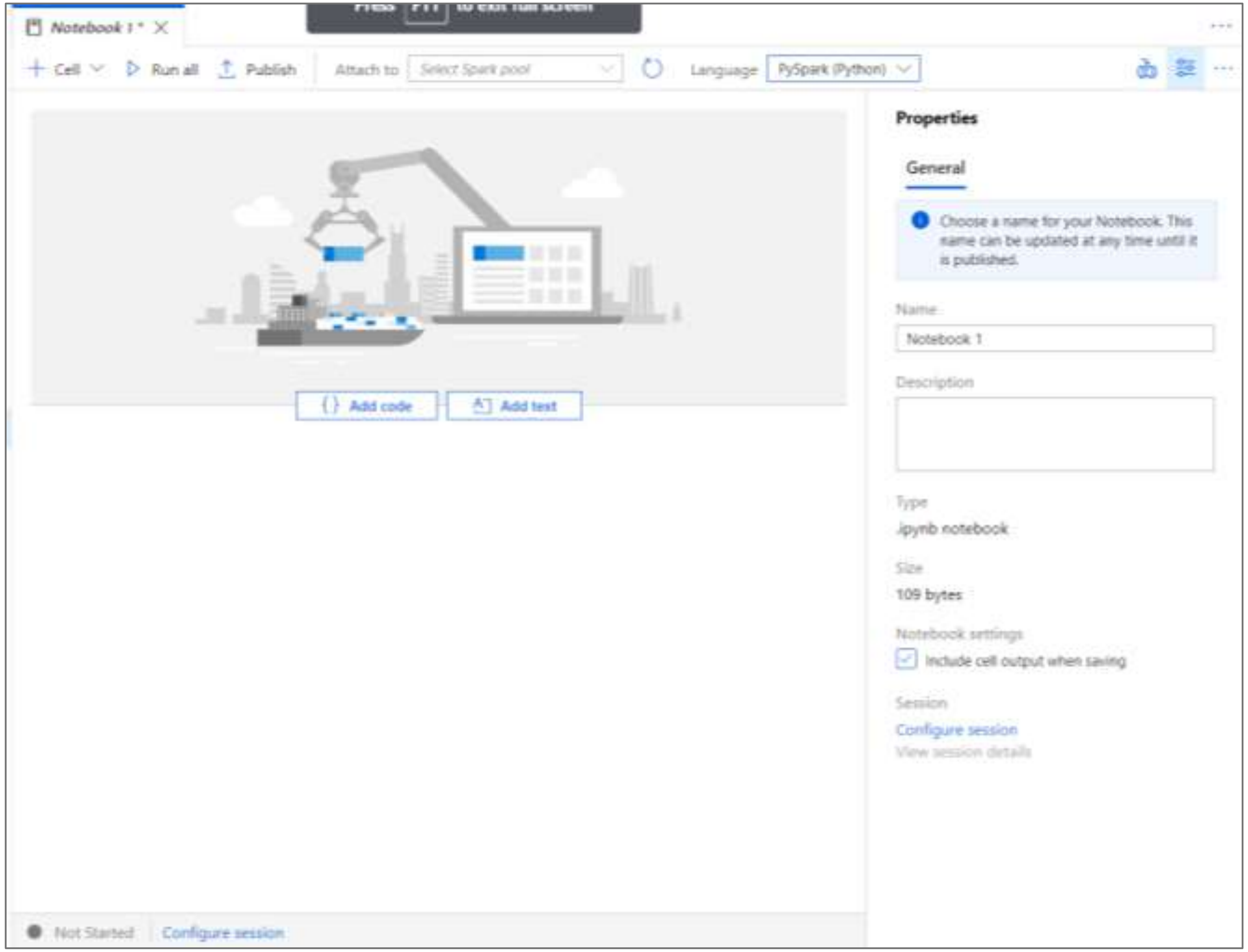
Benefits

Allows to write multiple languages in one notebook

%%<Name of language>

Offers use of temporary tables across languages

Notebooks > Configure Session



Properties

General

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

Name: Notebook 1

Description:

Type: .ipynb notebook

Size: 109 bytes

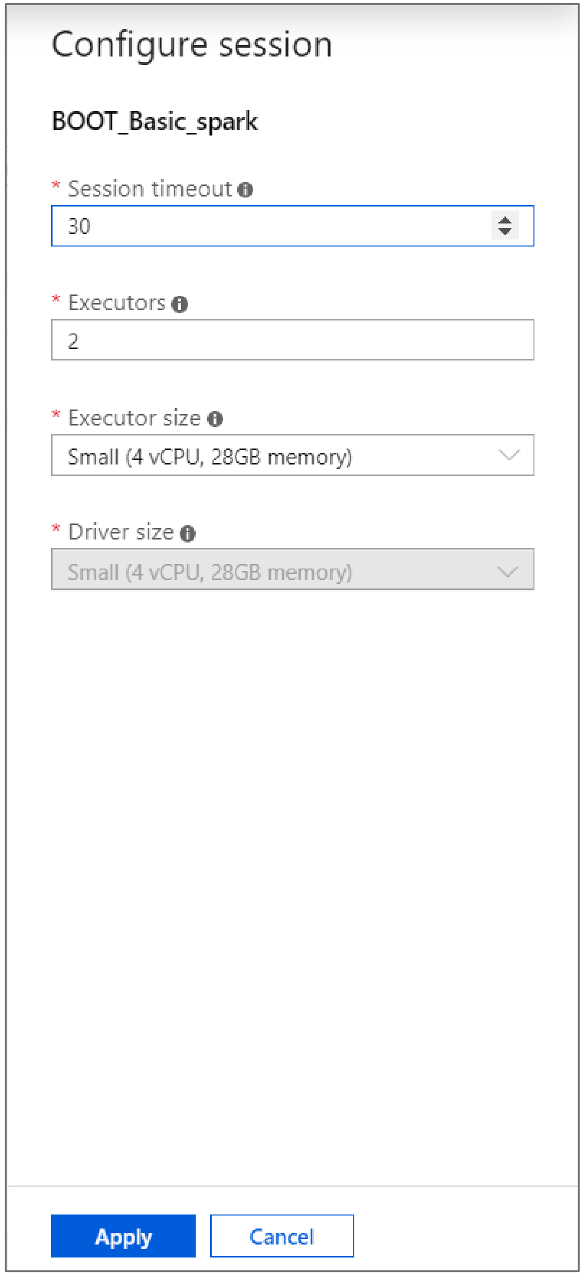
Notebook settings

☑ Include cell output when saving

Session

Configure session

View session details



Configure session

BOOT_Basic_spark

* Session timeout: 30

* Executors: 2

* Executor size: Small (4 vCPU, 28GB memory)

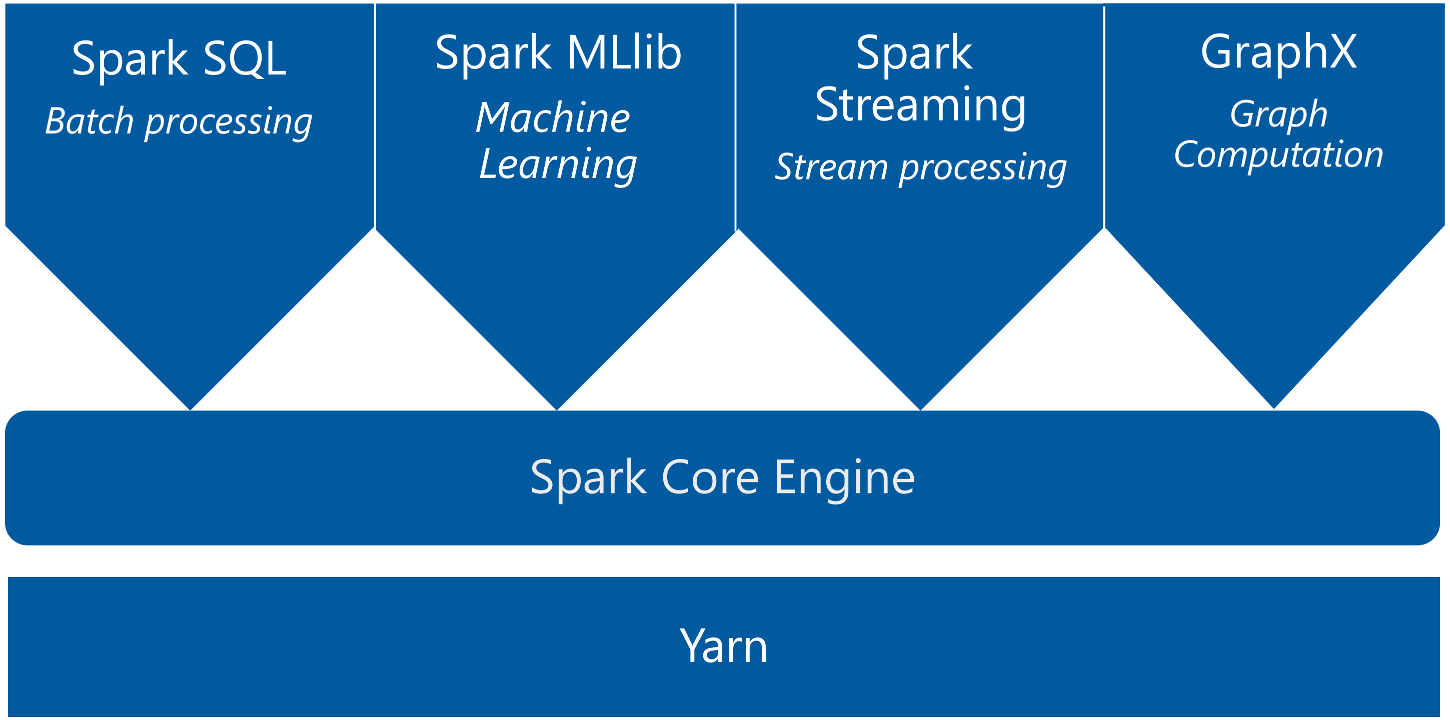
* Driver size: Small (4 vCPU, 28GB memory)

Apache Spark

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

Spark Unifies:

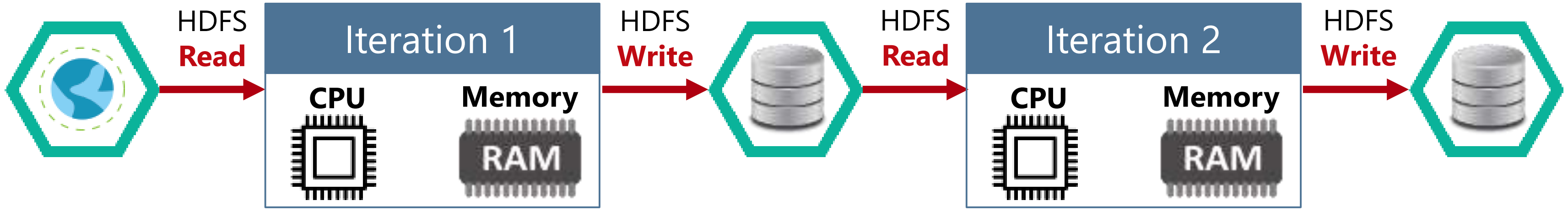
- Batch Processing
- Interactive SQL
- Real-time processing
- Machine Learning
- Deep Learning
- Graph Processing



http://spark.apache.org

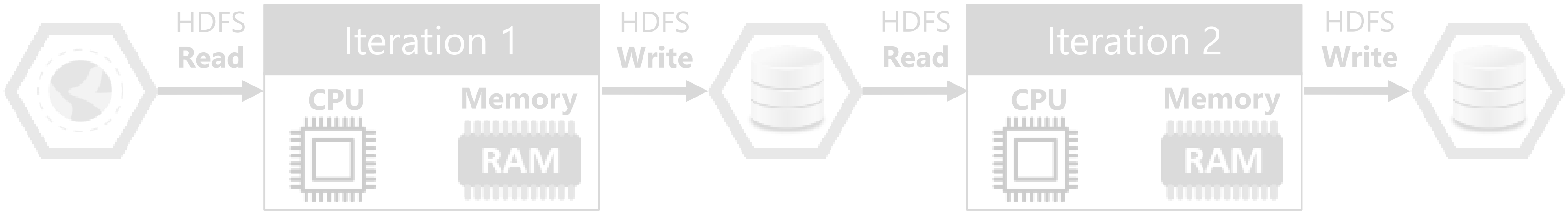
Motivation for Apache Spark

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

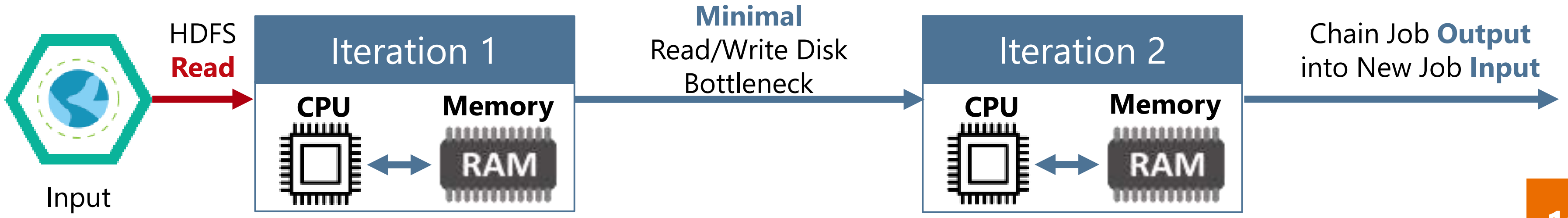


Motivation for Apache Spark

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

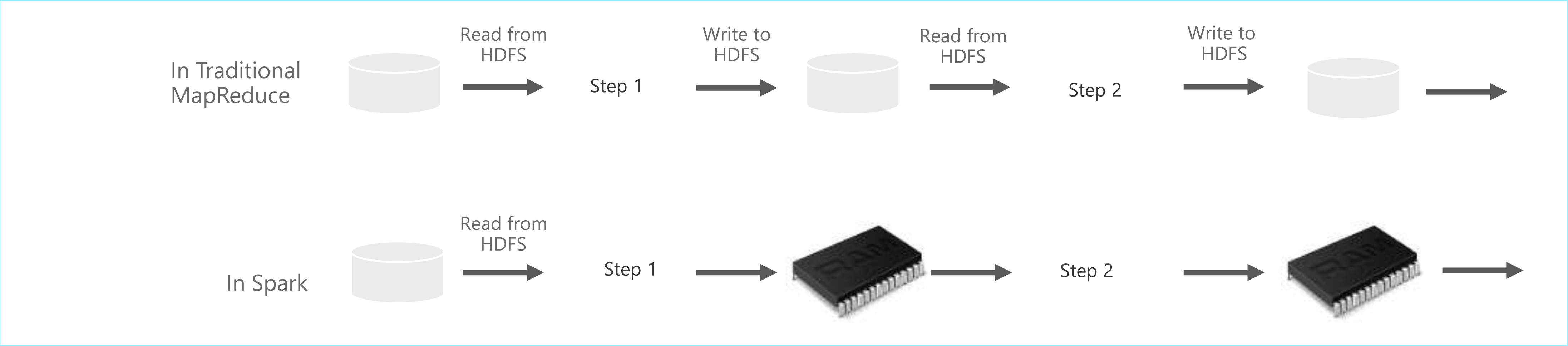


10–100x faster than network & disk

What makes Spark fast

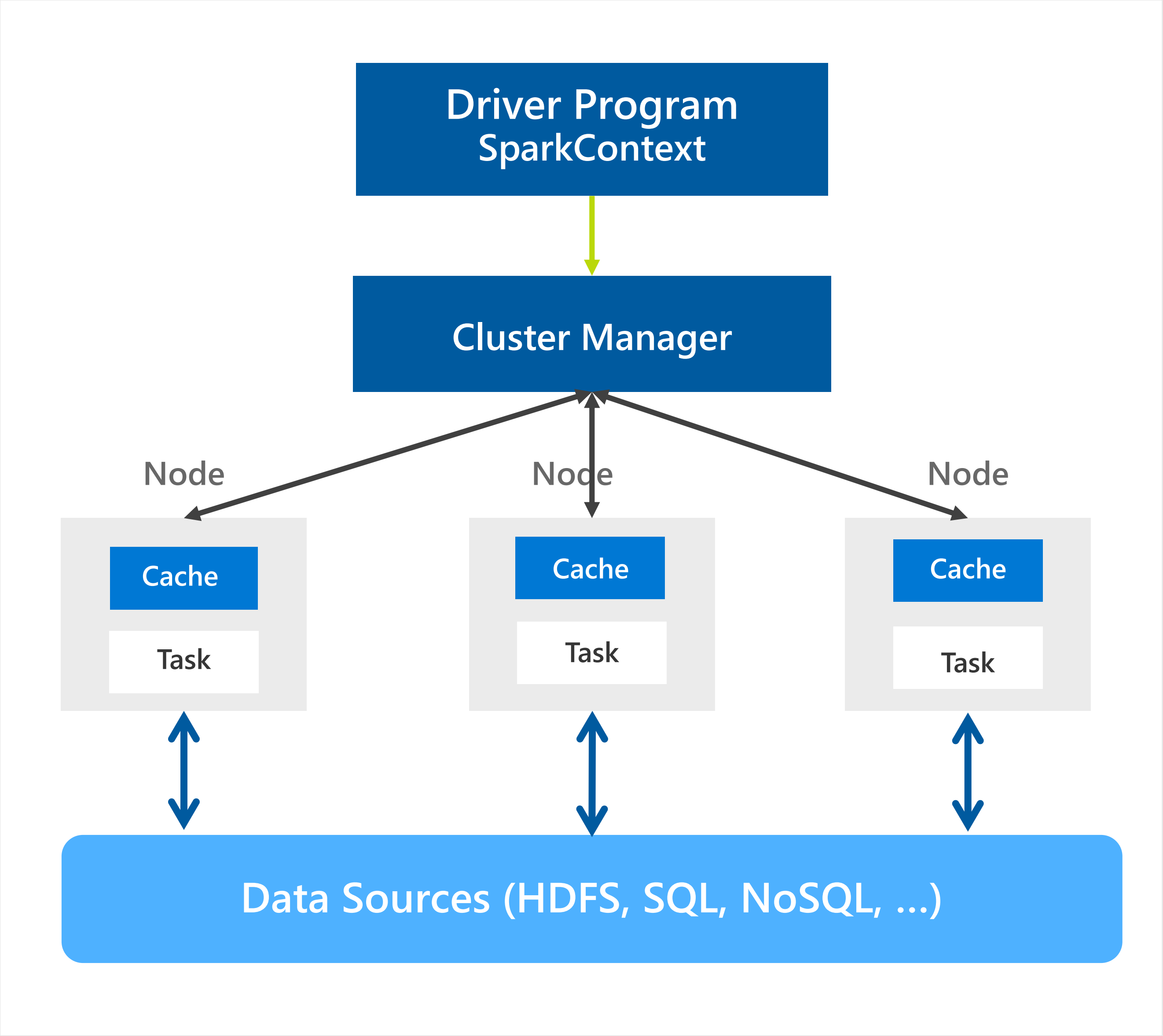
- **In-memory cluster computing**: Spark provides primitives for *in-memory* cluster computing. A Spark job can *load and cache* data into memory and query it repeatedly (iteratively) much quicker than disk-based systems.
- **Scala Integration**: Spark integrates into the Scala programming language, 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*.

Data Sharing between Steps of a Job



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

Mllib/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

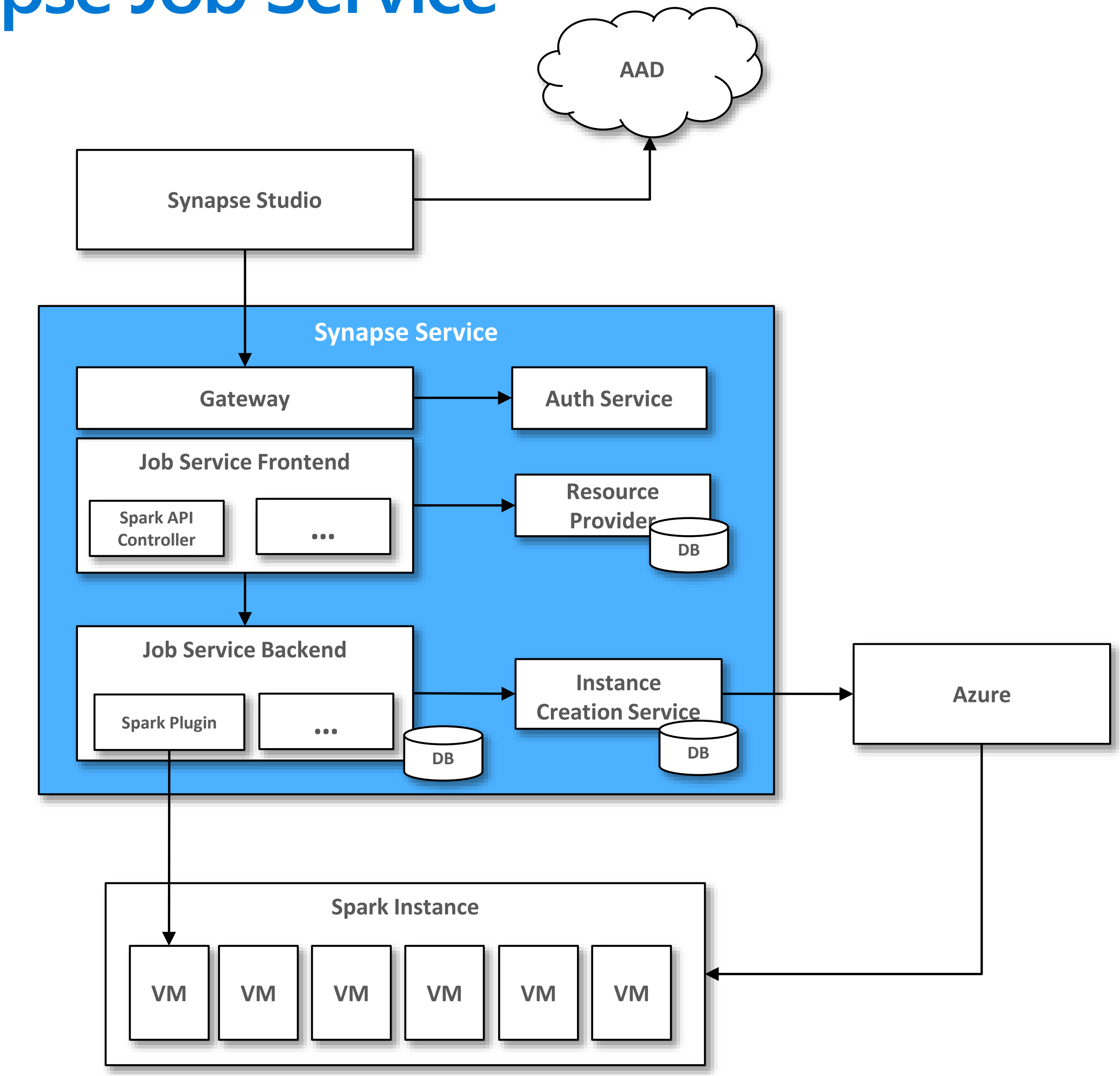
- Represent and analyze systems represented by graph nodes
- Trace interconnections between graph nodes
- Applicable to use cases in transportation, telecommunications, road networks, modeling personal relationships, social media, etc.

Microsoft

Azure Synapse Apache Spark

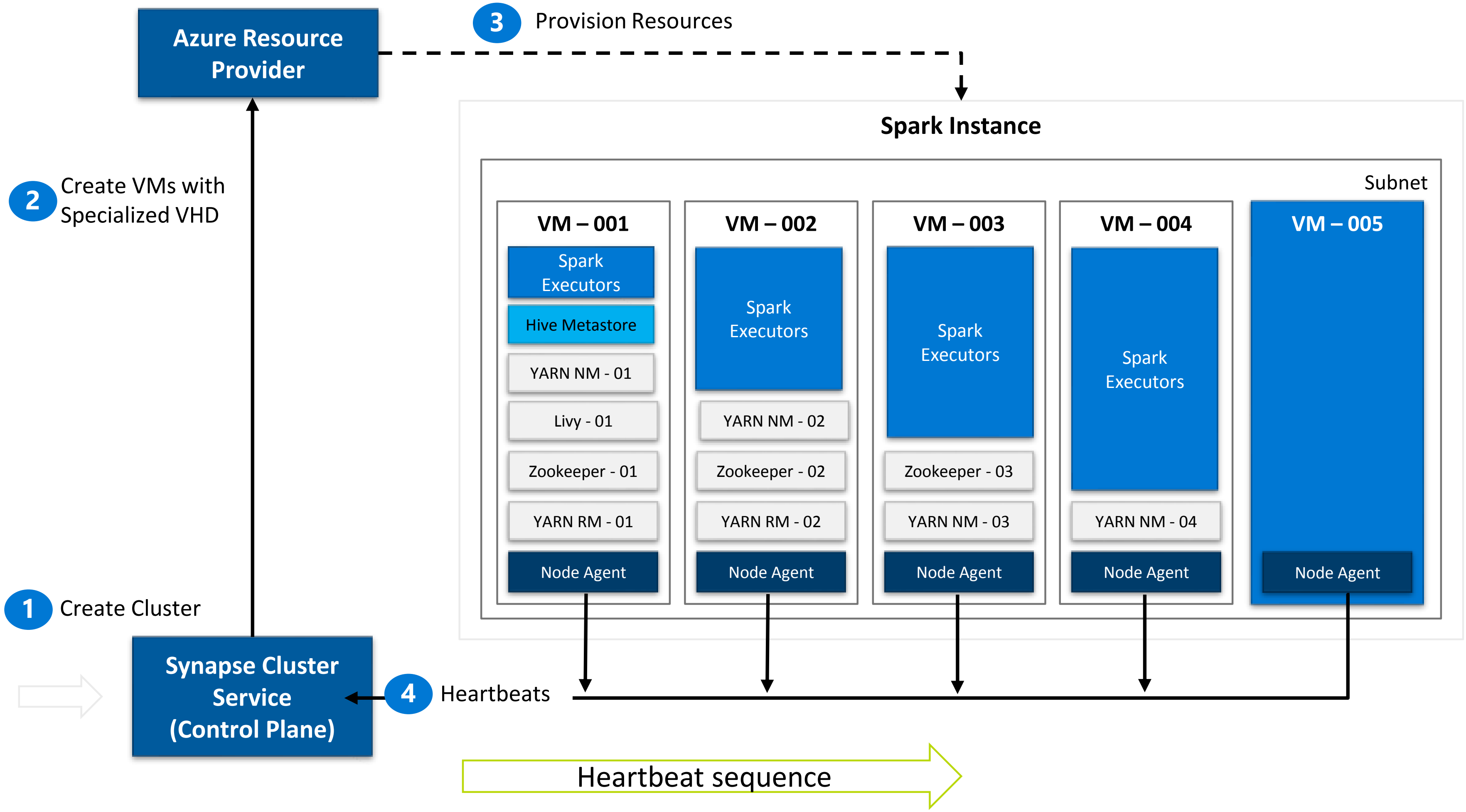
Architecture Overview

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

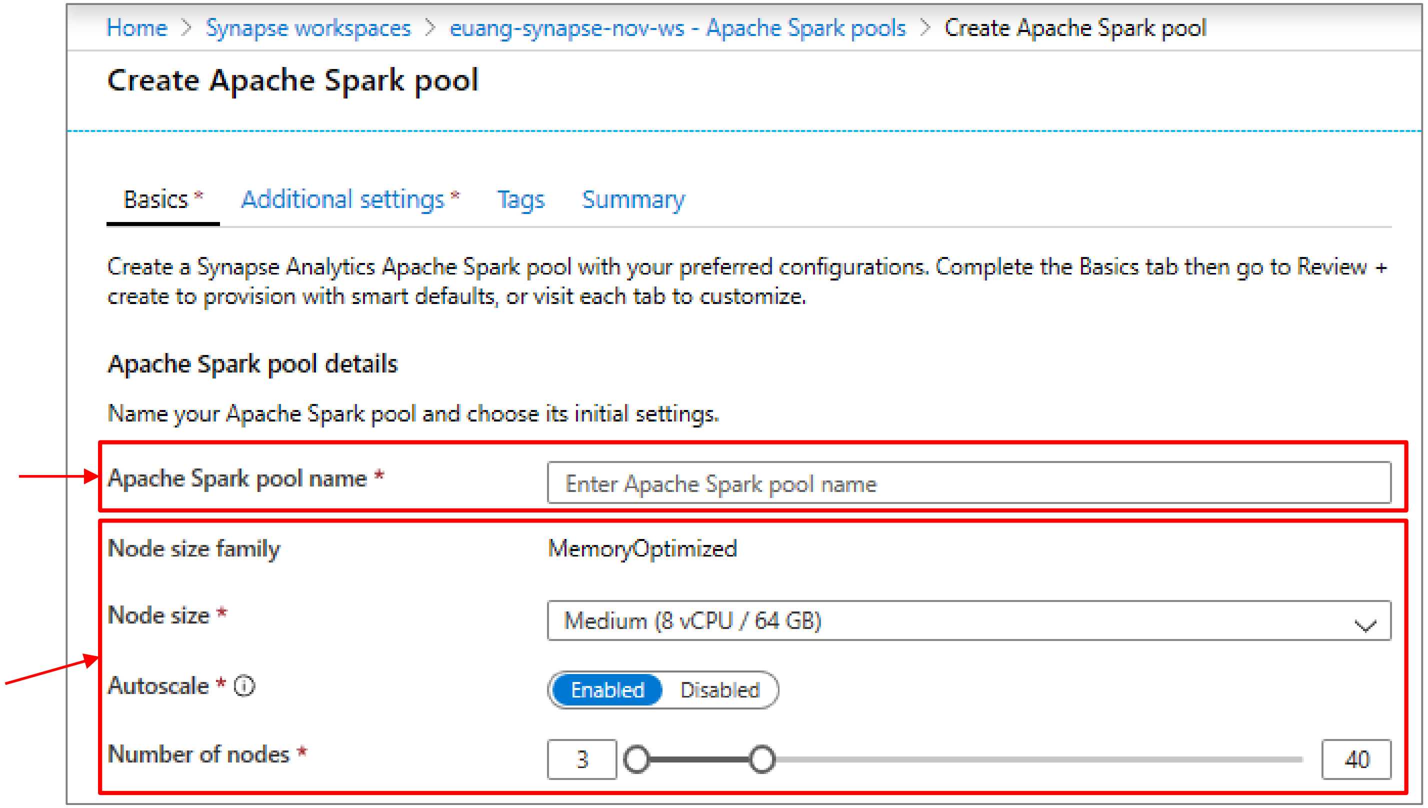


1. Synapse Job Service sends request to Cluster Service for creating BBC clusters per the description in the associated Spark pool.
2. Cluster Service sends request to Azure using Azure SDK to create VMs (required plus additional) with specialized VHD.
3. The specialized VHD contains bits for all the 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 end-point 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

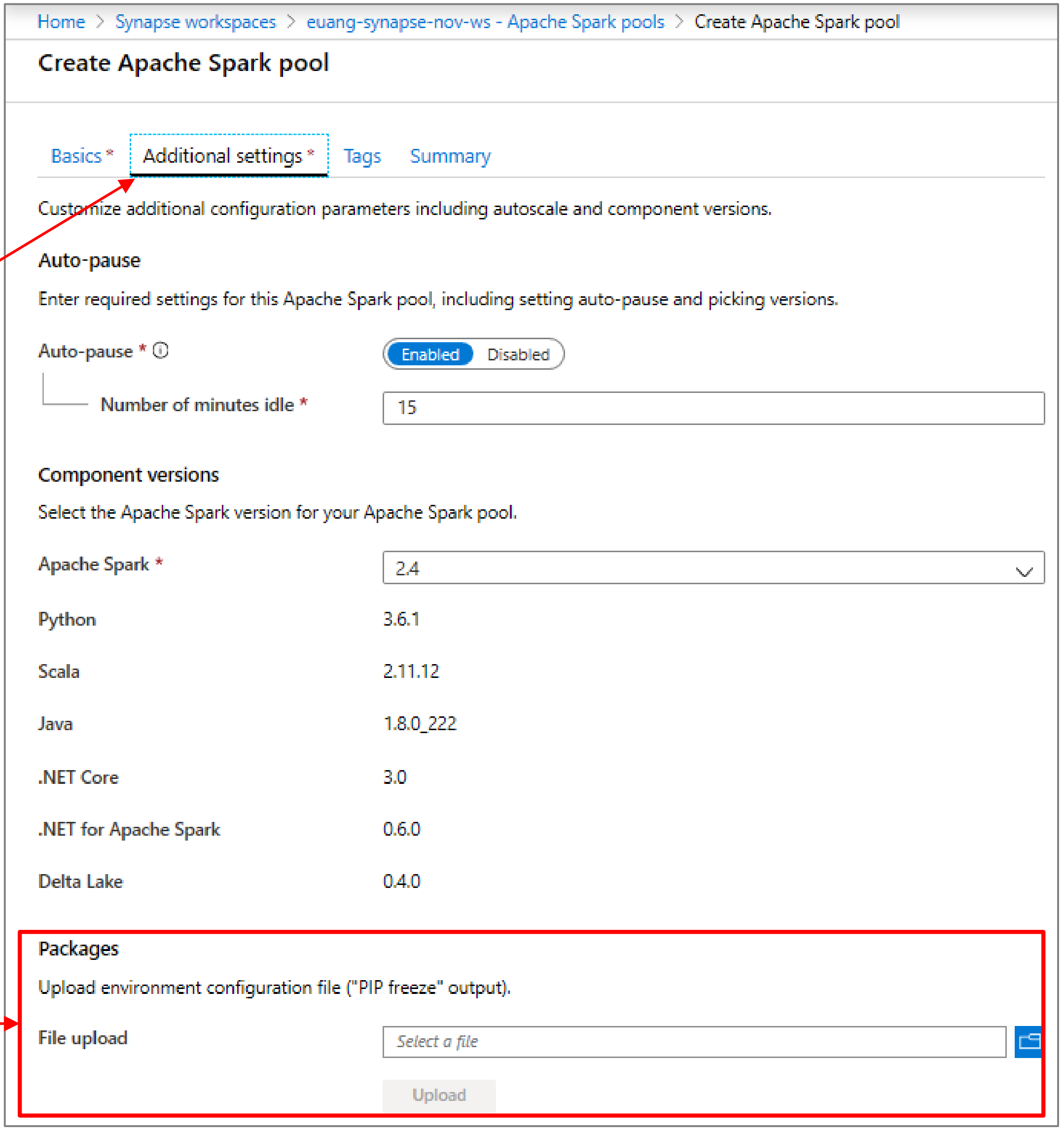
Default Settings

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

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 (selected) / 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).

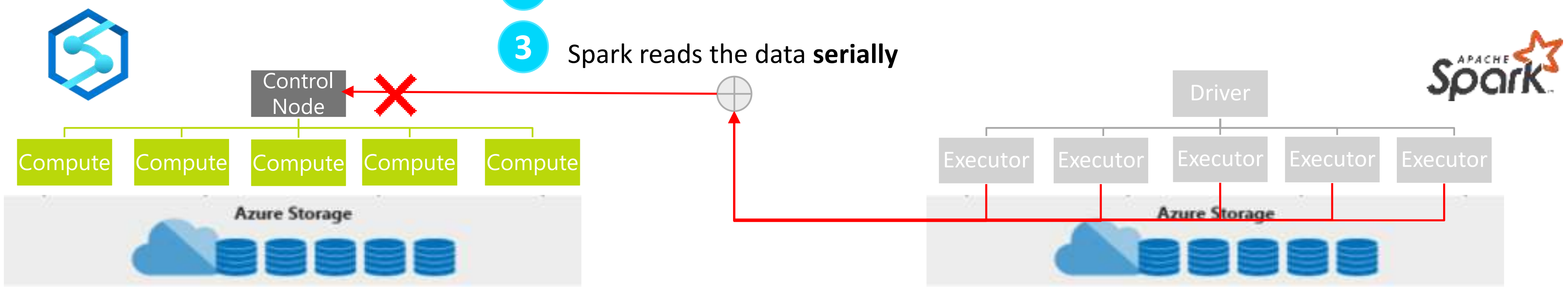
File upload: Select a file

Existing Approach: JDBC

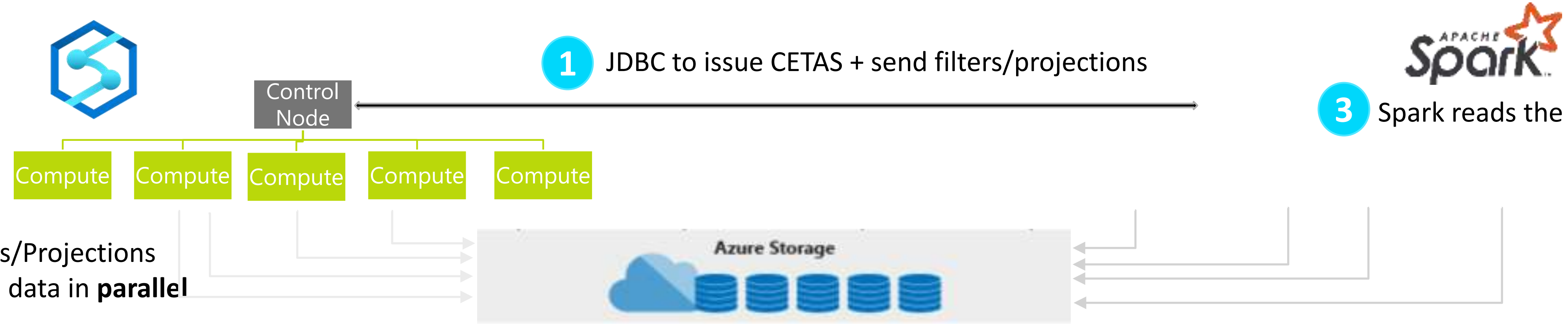
1 JDBC to open connection

2 Apply any Filters/Projections

3 Spark reads the data **serially**



New Approach: JDBC and Polybase



1 JDBC to issue CETAS + send filters/projections

2 Apply any Filters/Projections
DW exports the data in **parallel**

3 Spark reads the data in **parallel**

User Provisioned Workspace-Default Data Lake

Code-Behind Experience

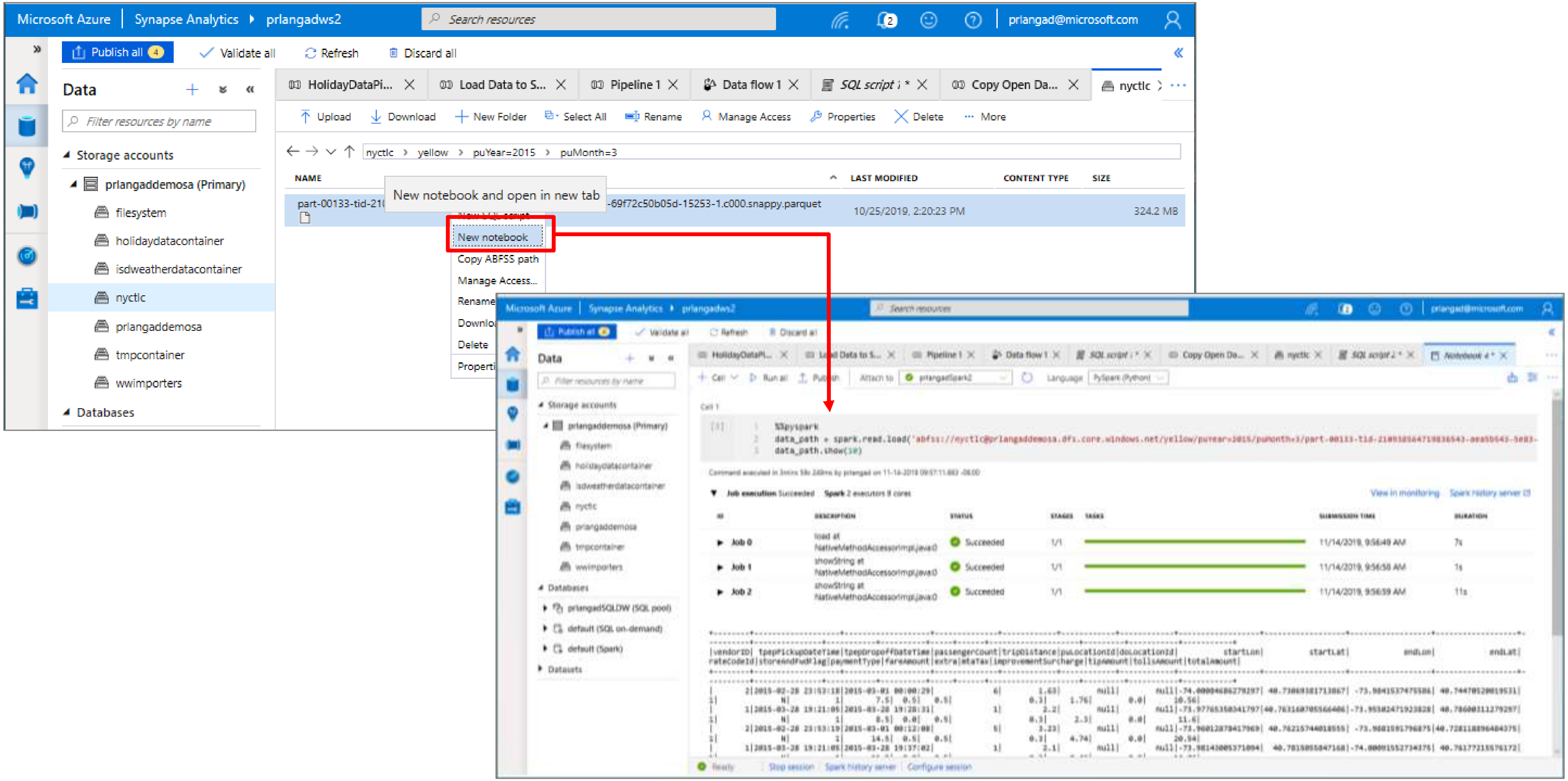
Existing Approach

```
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

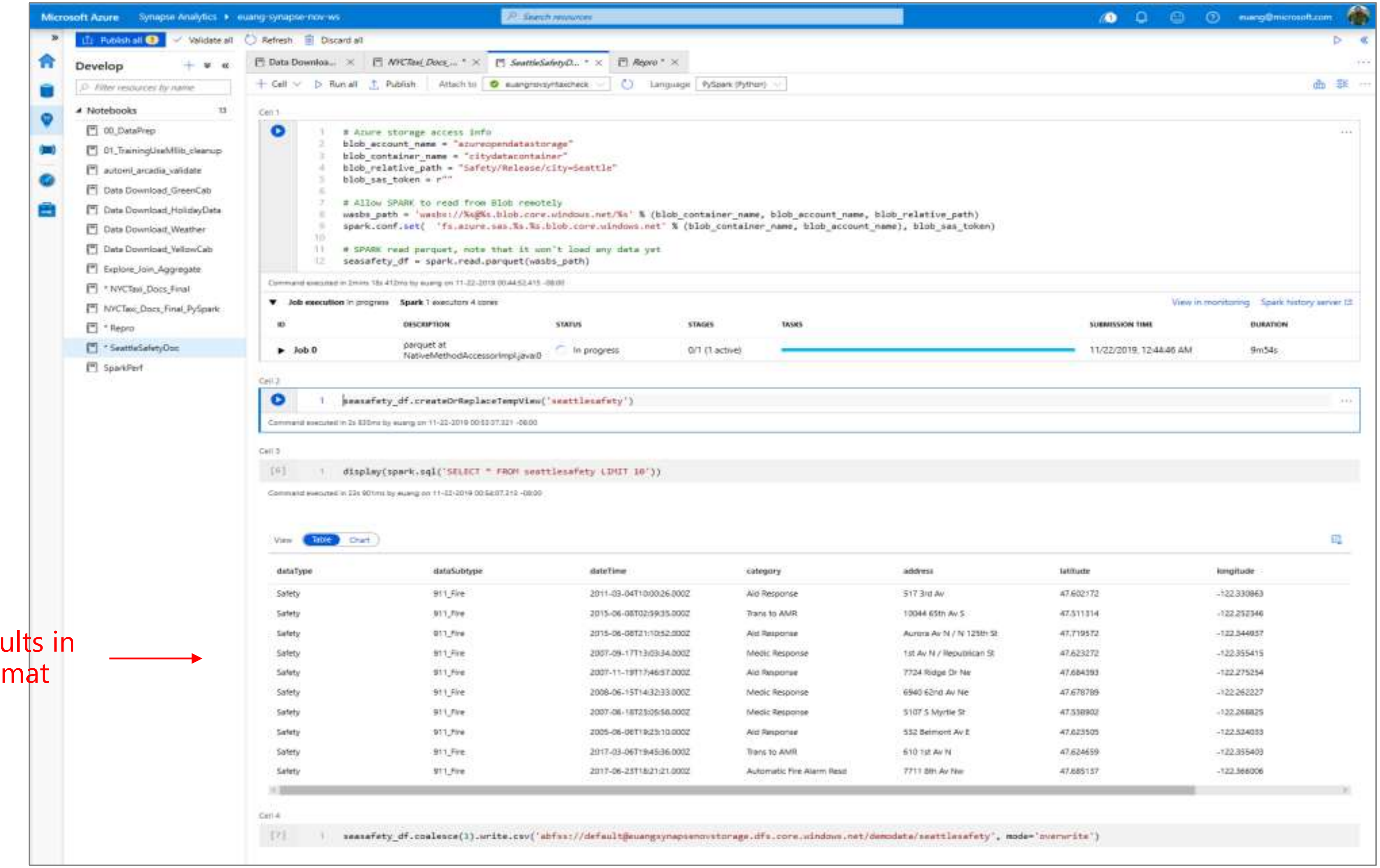


```
%%pyspark
data_path = spark.read.load('abfss://nyctlc@prlangaddemosa.dfs.core.windows.net/yellow/puYear=2015/puMonth=3/part-00133-tid-...
data_path.show(10)
```

Job execution Succeeded   Spark 2 executors 8 cores

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

```
|vendorID| tpepPickupDateTime| tpepDropoffDateTime|passengerCount|tripDistance|puLocationId|doLocationId| startLon| startLat| endLon| endLat|
rateCodeId|storeAndFwdFlag|paymentType|fareAmount|extra|mtaTax|improvementSurcharge|tipAmount|tollsAmount|totalAmount|
```



Cell 1

```
# Azure storage access info
blob_account_name = "azureopendatastorage"
blob_container_name = "citydatacontainer"
blob_relative_path = "Safety/Release/city=Seattle"
blob_sas_token = r""

# Allow SPARK to read from Blob remotely
wasbs_path = 'wasbs://%s@%s.blob.core.windows.net/%s' % (blob_container_name, blob_account_name, blob_relative_path)
spark.conf.set(  'fs.azure.sas.%s.%s.blob.core.windows.net' % (blob_container_name, blob_account_name), blob_sas_token)

# SPARK read parquet, note that it won't load any data yet
seasafety_df = spark.read.parquet(wasbs_path)
```

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

Job execution In progress **Spark** 1 executors 4 cores

ID	DESCRIPTION	STATUS	STAGES	TASKS	SUBMISSION TIME	DURATION
Job 0	parquet at NativeMethodAccessorImpl.java:0	In progress	0/1 (1 active)		11/22/2019, 12:44:46 AM	9m54s

Cell 2

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

Command executed in 2s 830ms by euang on 11-22-2019 00:53:37.321 -08:00

Cell 3

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

Command executed in 22s 901ms by euang on 11-22-2019 00:54:07.212 -08:00

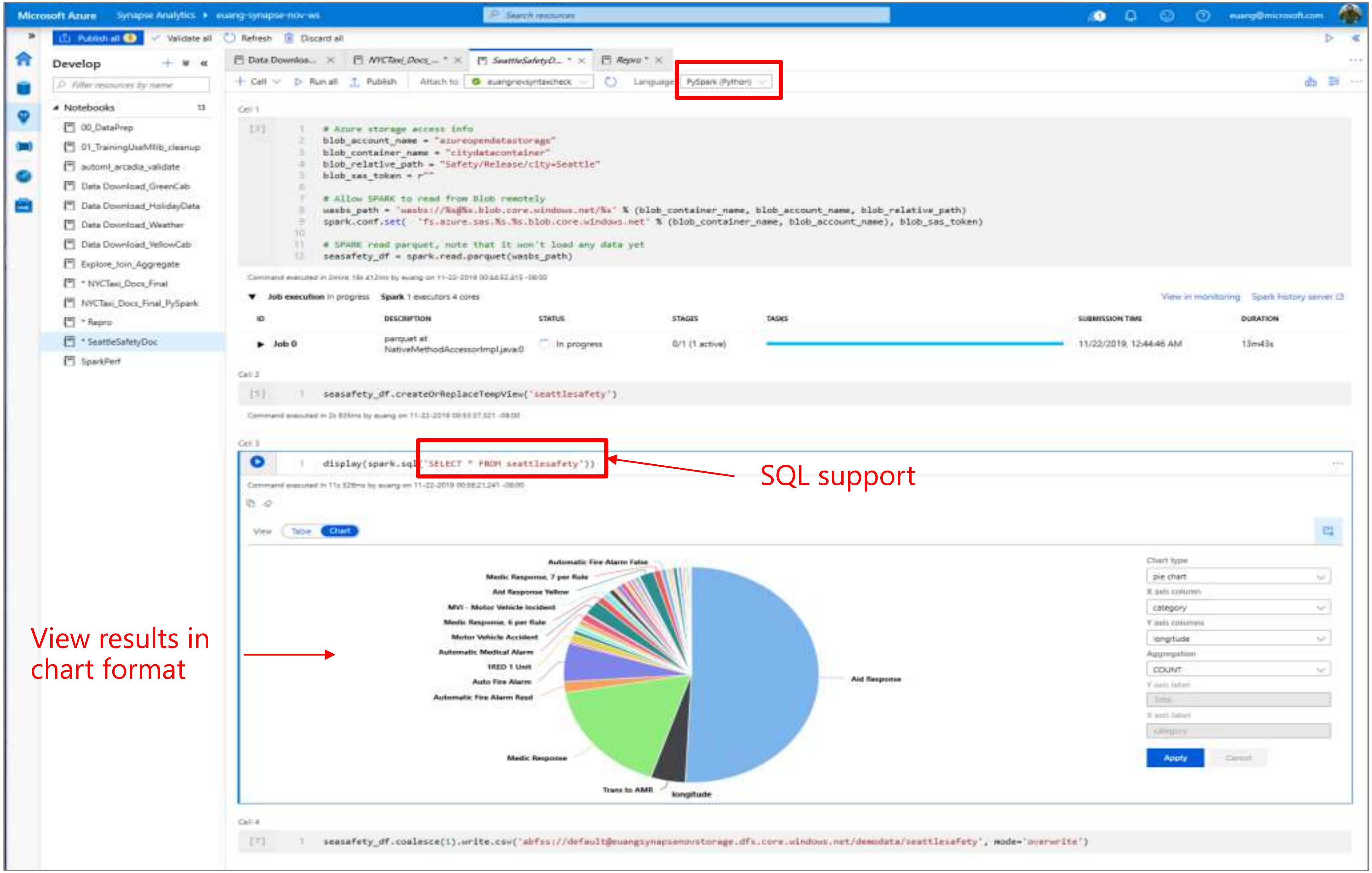
dataType	dataSubtype	dateTime	category	address	latitude	longitude
Safety	911_Fire	2011-03-04T10:00:26.000Z	Aid Response	517 3rd Av	47.602172	-122.330863
Safety	911_Fire	2015-06-08T02:59:35.000Z	Trans to AMR	10044 65th Av S	47.511314	-122.252346
Safety	911_Fire	2015-06-08T21:10:52.000Z	Aid Response	Aurora Av N / N 125th St	47.719572	-122.344837
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	2007-06-18T23:06:58.000Z	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]  seasafety_df.coalesce(1).write.csv('abfss://default@euangsynapsenovstorage.dfs.core.windows.net/demodata/seattlesafety', mode='overwrite')
```

View results in table format →

Azure Synapse Analytics > Spark > User Experience and Languages



Cell 1

```
# Azure storage access info
blob_account_name = "azureopendatastorage"
blob_container_name = "citydatacontainer"
blob_relative_path = "Safety/Release/city=Seattle"
blob_sas_token = r""

# Allow SPARK to read from Blob remotely
wasbs_path = 'wasbs://%s@%s.blob.core.windows.net/%s' % (blob_container_name, blob_account_name, blob_relative_path)
spark.conf.set(  'fs.azure.sas.%s.%s.blob.core.windows.net' % (blob_container_name, blob_account_name), blob_sas_token)

# SPARK read parquet, note that it won't load any data yet
seasafety_df = spark.read.parquet(wasbs_path)
```

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

Job execution In progress   Spark 1 executors 4 cores

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

Cell 2

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

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

Cell 3

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

SQL support

Command executed in 11s 326ms by euang on 11-22-2019 00:58:21.241 -08:00

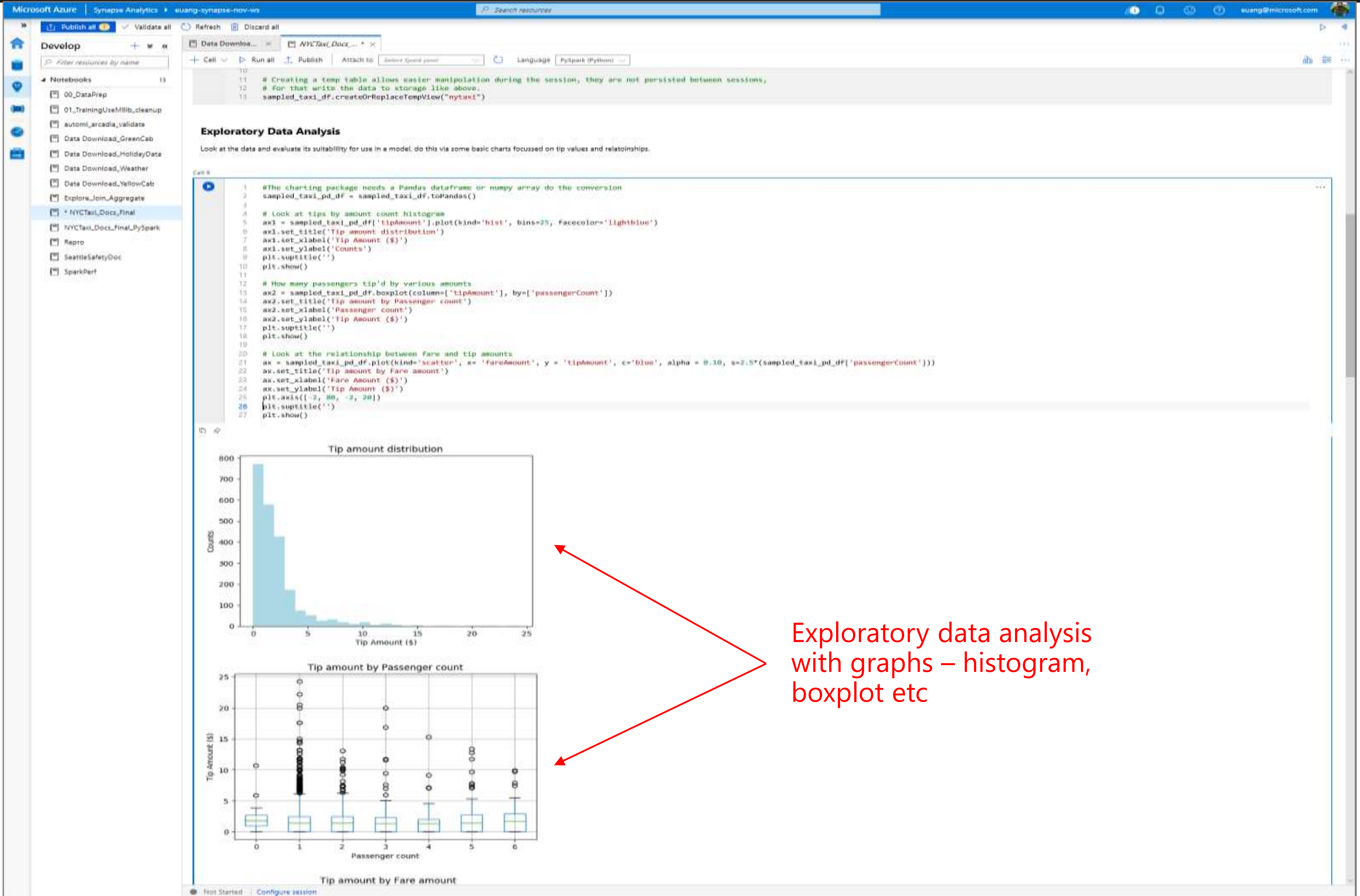
Chart type: pie chart
X axis column: category
Y axis columns: longitude
Aggregation: COUNT
Y axis label: Total
X axis label: category

View results in chart format

Cell 4

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

Azure Synapse Analytics > Spark > User Experience and Languages



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

Exploratory Data Analysis

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

```
#The charting package needs a Pandas dataframe or numpy array do the conversion
sampled_taxi_pd_df = sampled_taxi_df.toPandas()

# Look at tips by amount count histogram
ax1 = sampled_taxi_pd_df['tipAmount'].plot(kind='hist', bins=25, facecolor='lightblue')
ax1.set_title('Tip amount distribution')
ax1.set_xlabel('Tip Amount ($)')
ax1.set_ylabel('Counts')
plt.suptitle('')
plt.show()

# How many passengers tip'd by various amounts
ax2 = sampled_taxi_pd_df.boxplot(column=['tipAmount'], by=['passengerCount'])
ax2.set_title('Tip amount by Passenger count')
ax2.set_xlabel('Passenger count')
ax2.set_ylabel('Tip Amount ($)')
plt.suptitle('')
plt.show()

# Look at the relationship between fare and tip amounts
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']))
ax.set_title('Tip amount by Fare amount')
ax.set_xlabel('Fare Amount ($)')
ax.set_ylabel('Tip Amount ($)')
plt.axis([-2, 80, -2, 20])
plt.suptitle('')
plt.show()
```

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

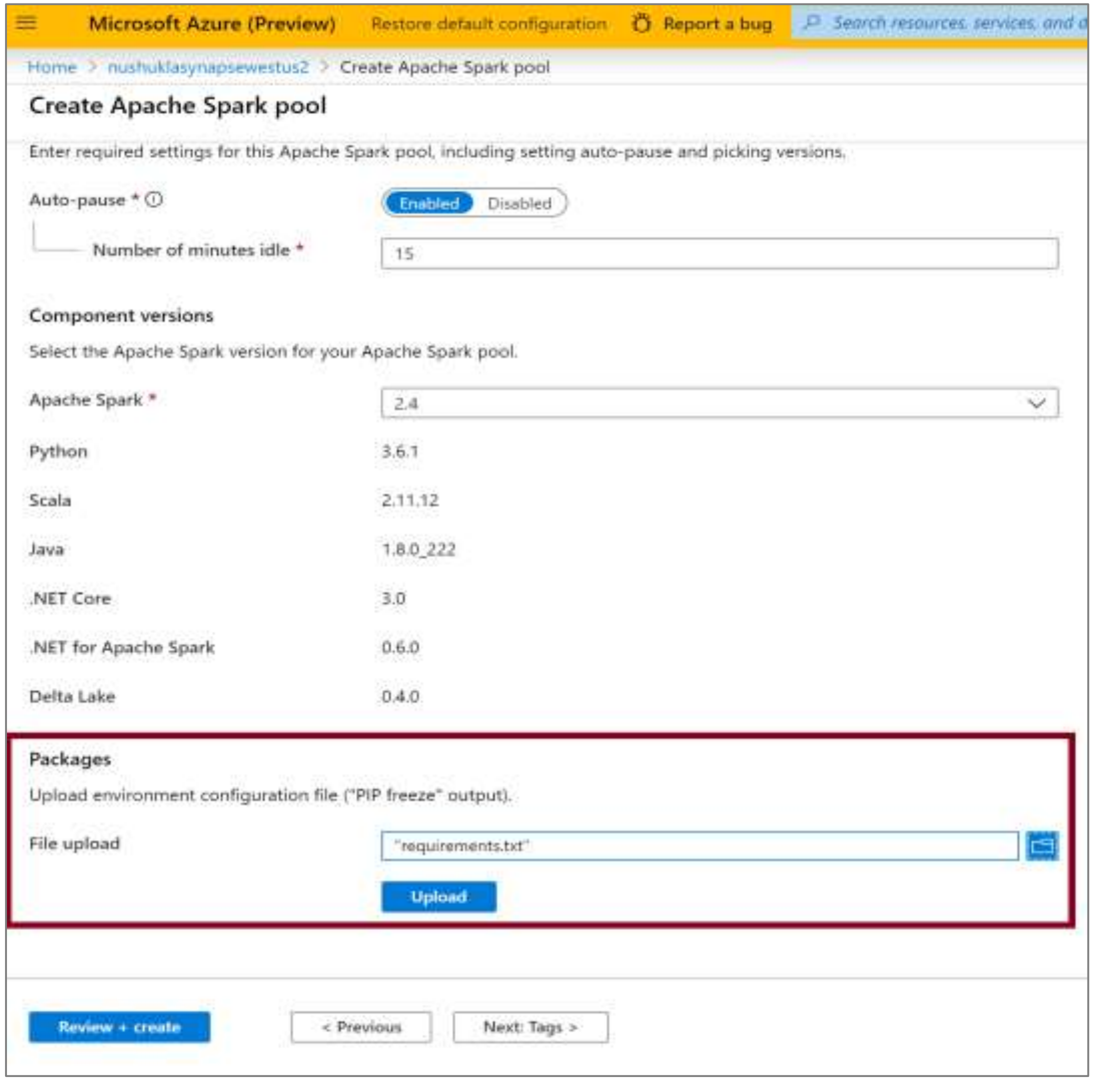
Constraints

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 d
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 * 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

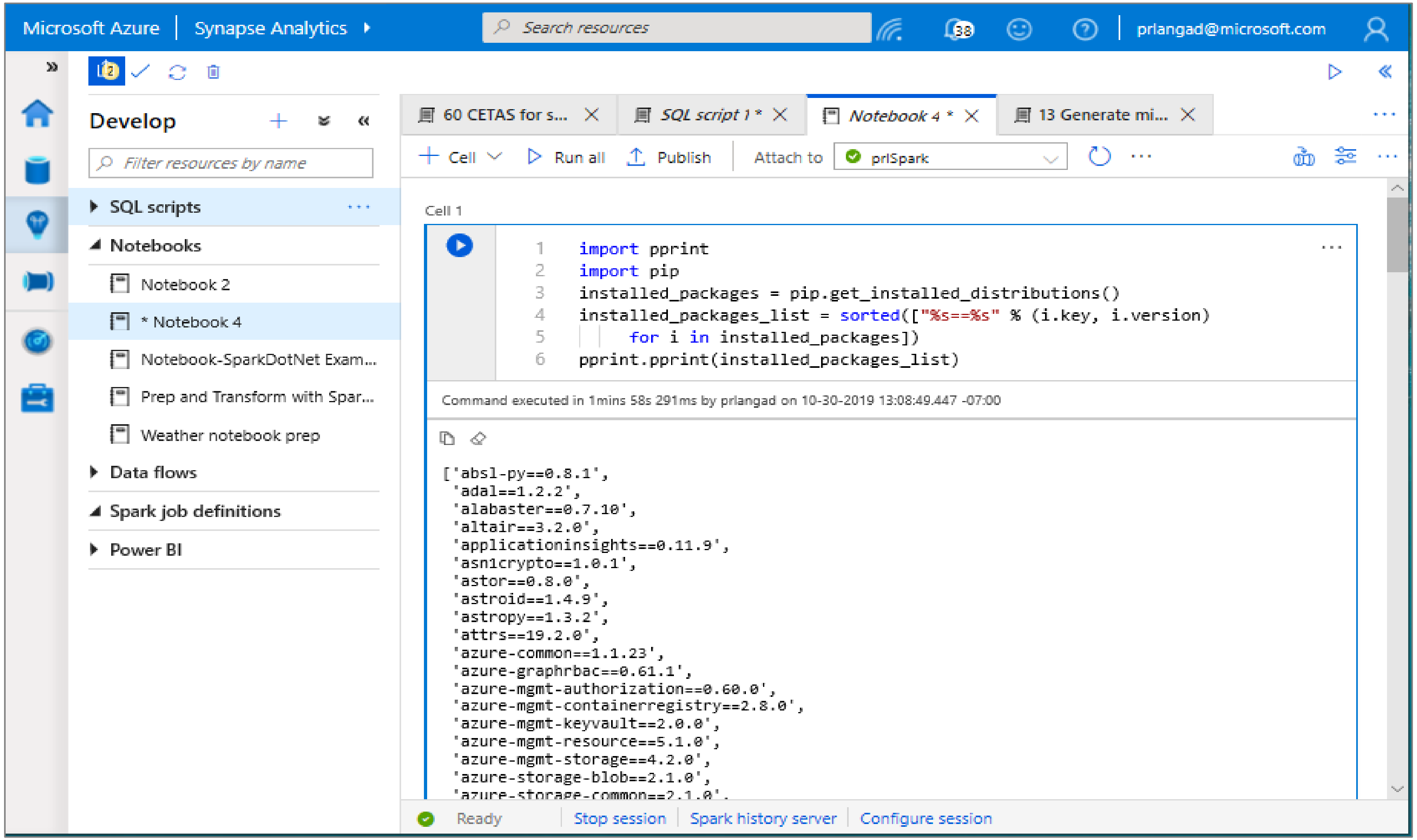
Packages

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

File upload "requirements.txt"

Library Management - Python

Get list of installed libraries with version information



```python
import pprint
import pip
installed_packages = pip.get_installed_distributions()
installed_packages_list = sorted(["%s==%s" % (i.key, i.version)
    for i in installed_packages])
pprint.pprint(installed_packages_list)
```

Command executed in 1mins 58s 291ms by prlangad on 10-30-2019 13:08:49.447 -07:00

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

Spark ML Algorithms

Spark ML Algorithms

Classification and Regression	• Linear Models (SVMs, logistic regression, linear regression) • Naïve Bayes • Decision Trees • Ensembles of trees (Random Forest, Gradient-Boosted Trees) • Isotonic regression
Clustering	• k-means and streaming k-means • Gaussian mixture • Power iteration clustering (PIC) • Latent Dirichlet allocation (LDA)
Collaborative Filtering	• Alternating least squares (ALS)
Dimensionality Reduction	• SVD • PCA
Frequent Pattern Mining	• FP-growth • Association rules
Basic Statistics	• 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

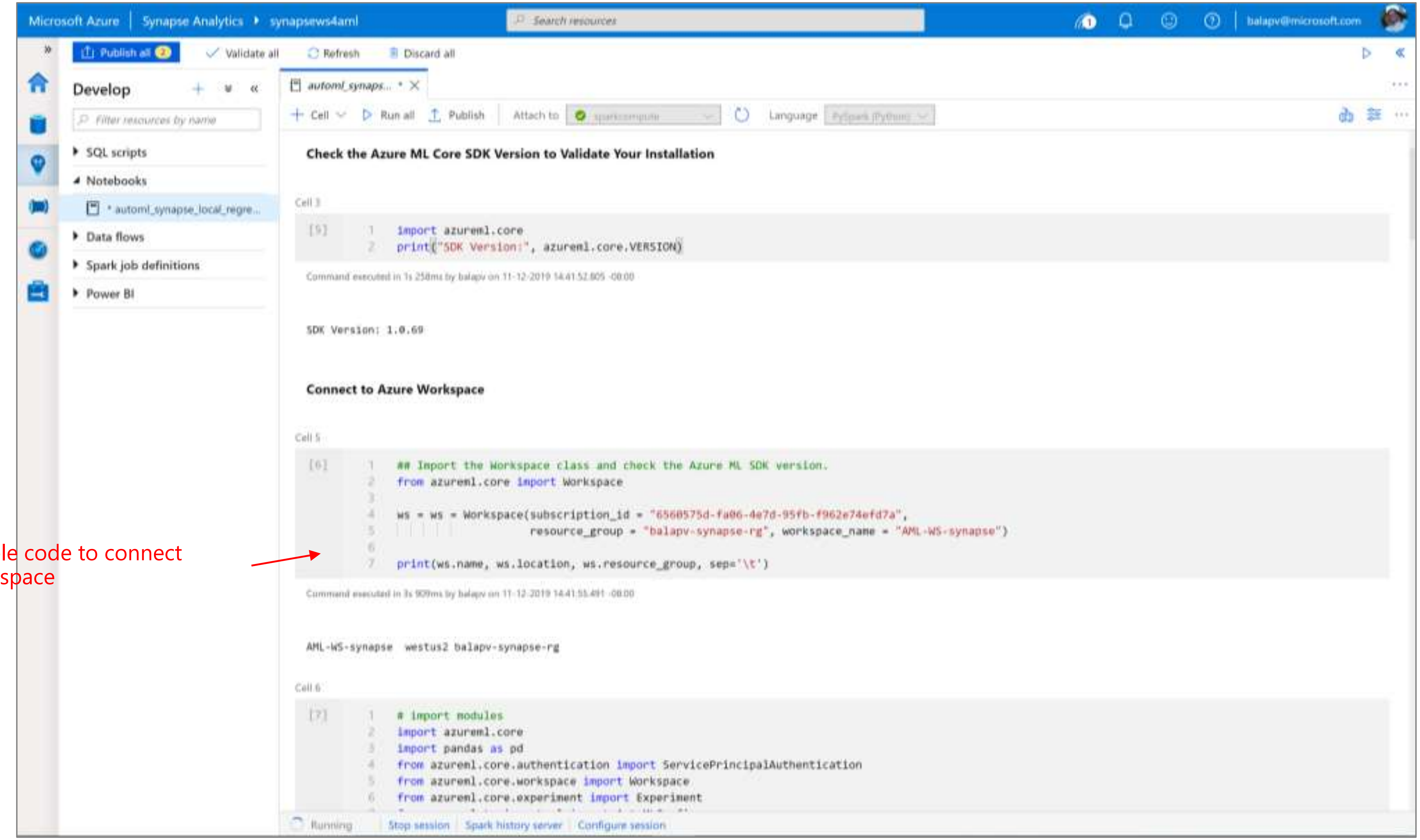
Microservice Orchestration

Multilingual Binding Generation

www.aka.ms/spark

Azure/mmlspark

Synapse Notebook: Connect to AML workspace



Check the Azure ML Core SDK Version to Validate Your Installation

Cell 3

```
[9]  import azureml.core
     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]  ## Import the Workspace class and check the Azure ML SDK version.
     from azureml.core import Workspace

     ws = ws = Workspace(subscription_id = "6560575d-fa06-4e7d-95fb-f962e74efd7a",
                         resource_group = "balapv-synapse-rg", workspace_name = "AML-WS-synapse")

     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

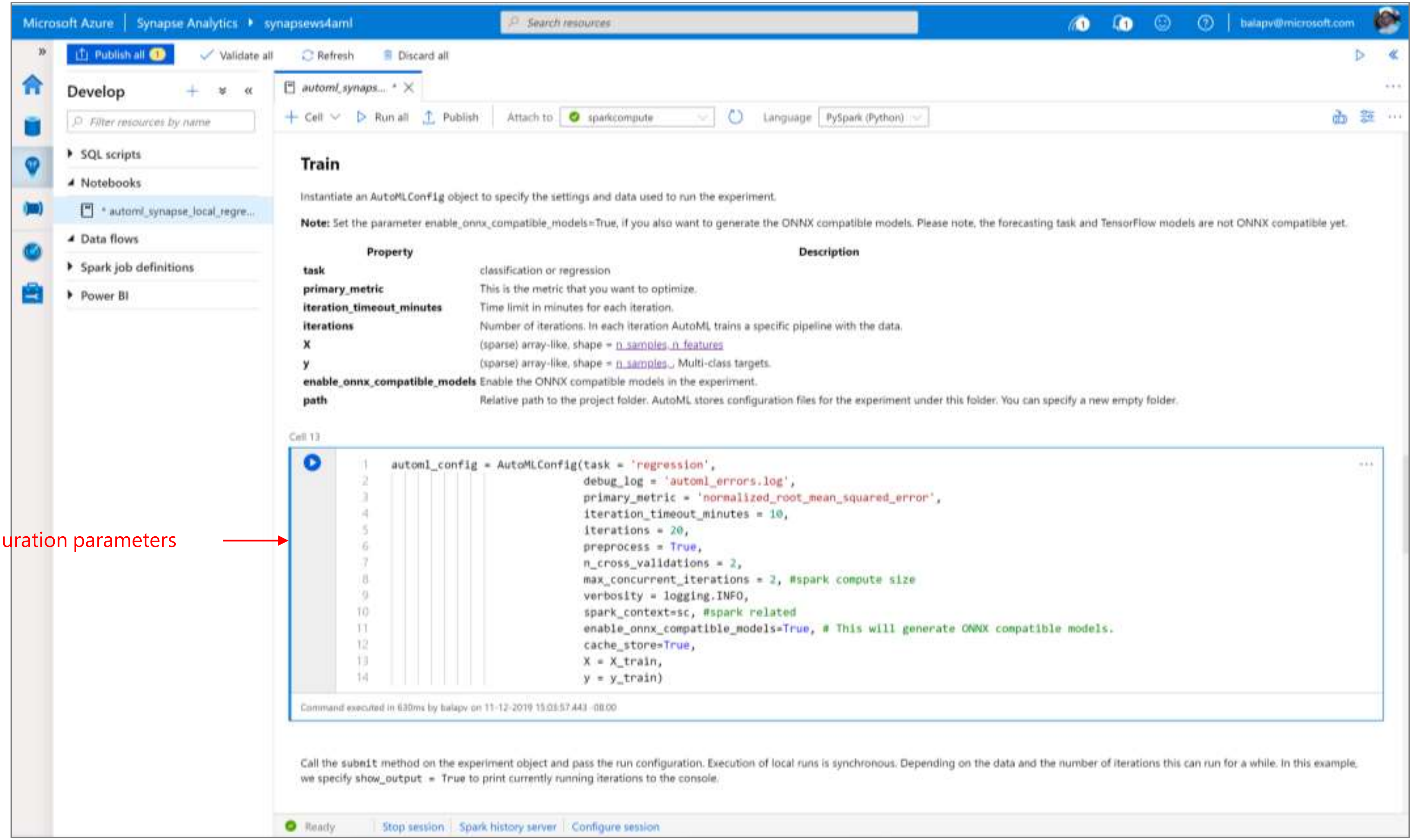
```
AML-WS-synapse	westus2	balapv-synapse-rg
```

Cell 6

```
[7]  # import modules
     import azureml.core
     import pandas as pd
     from azureml.core.authentication import ServicePrincipalAuthentication
     from azureml.core.workspace import Workspace
     from azureml.core.experiment import Experiment
```

Simple code to connect workspace

Synapse Notebook: Configure AML job to run on Synapse



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 = n_samples, n_features
y	(sparse) array-like, shape = n_samples, , 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.

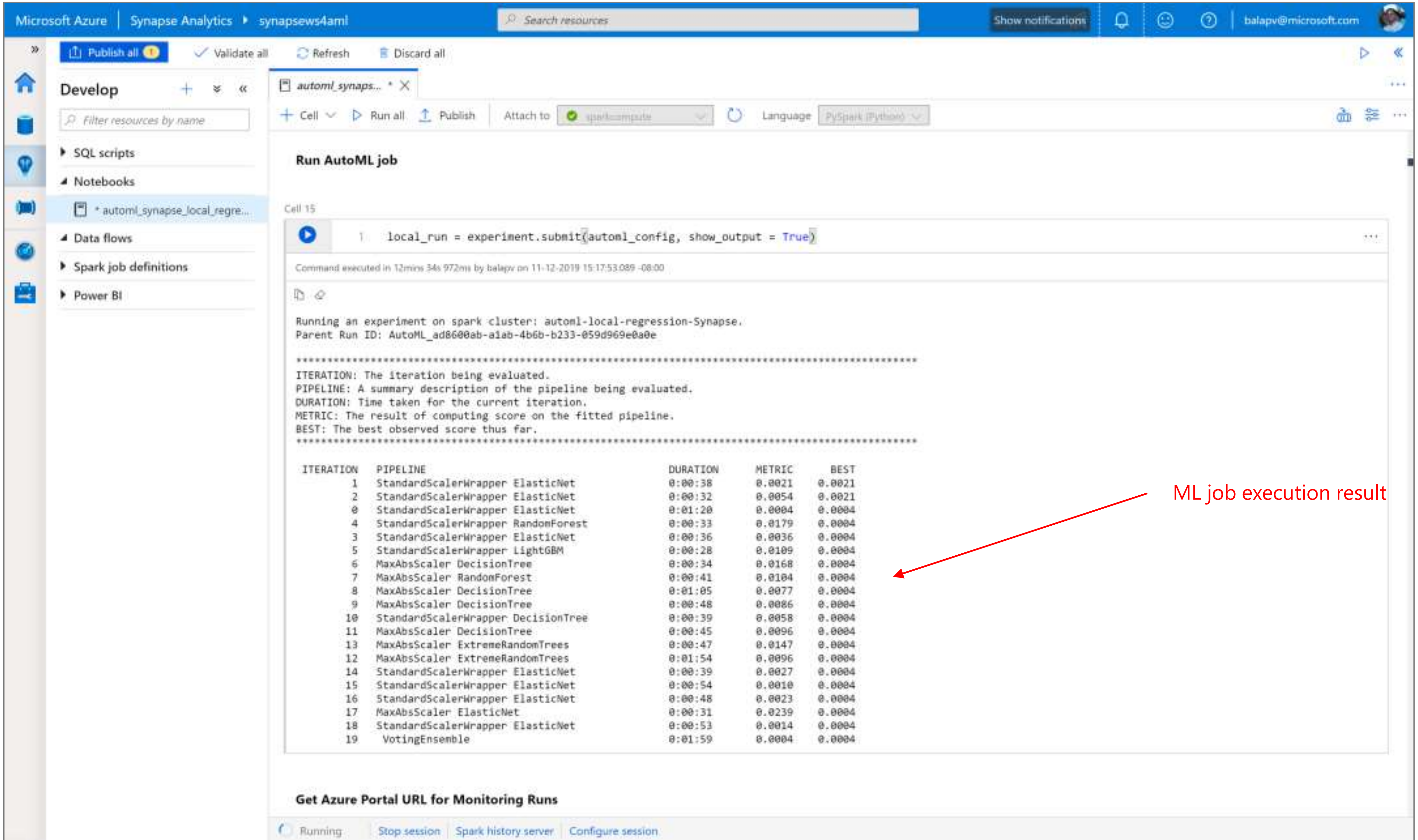
Configuration parameters

```
automl_config = AutoMLConfig(task = 'regression',
                             debug_log = 'automl_errors.log',
                             primary_metric = 'normalized_root_mean_squared_error',
                             iteration_timeout_minutes = 10,
                             iterations = 20,
                             preprocess = True,
                             n_cross_validations = 2,
                             max_concurrent_iterations = 2, #spark compute size
                             verbosity = logging.INFO,
                             spark_context=sc, #spark related
                             enable_onnx_compatible_models=True, # This will generate ONNX compatible models.
                             cache_store=True,
                             X = X_train,
                             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.

Synapse Notebook: Run AML job



Run AutoML job

Cell 15

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

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

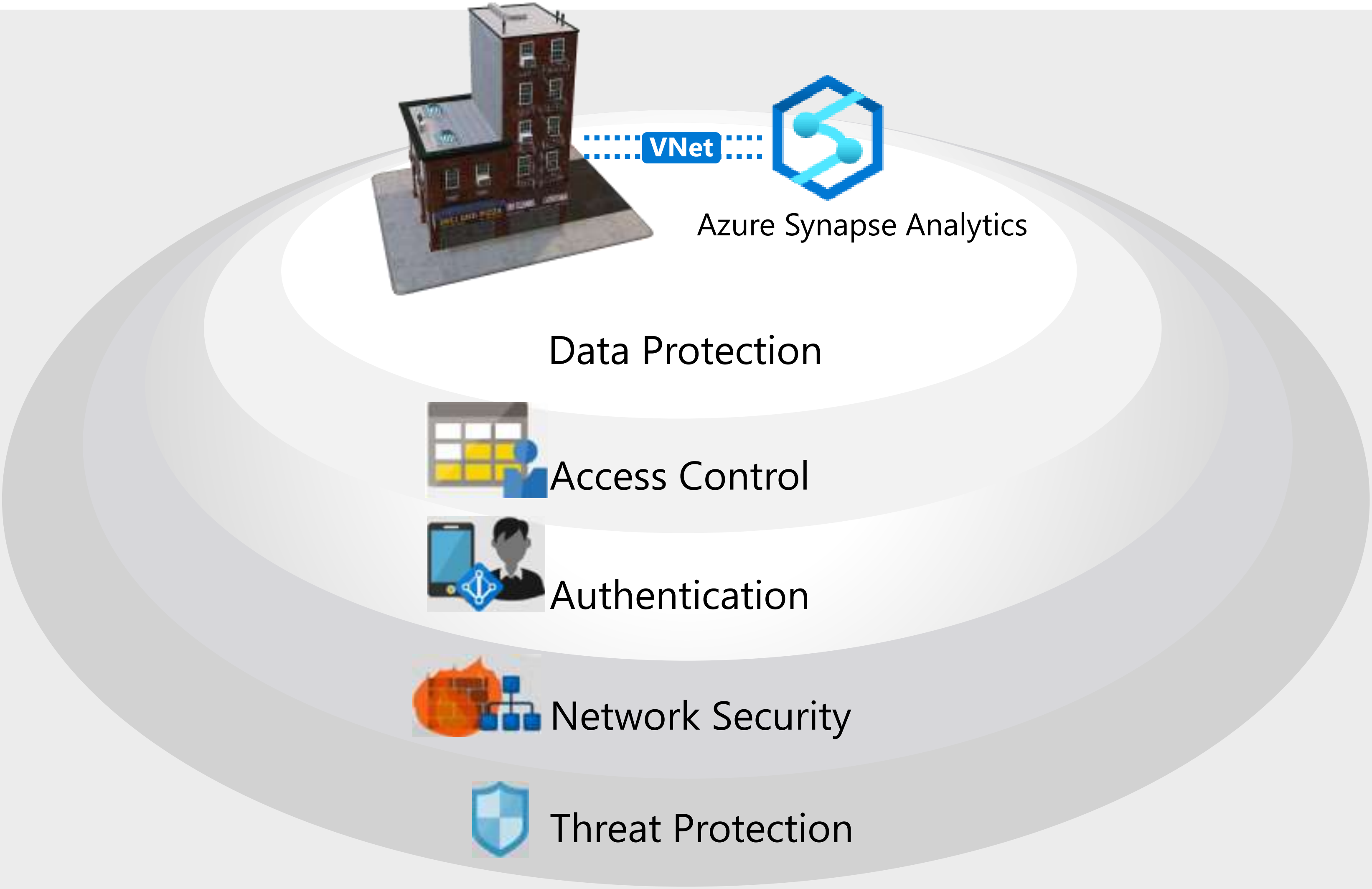
```
Running an experiment on spark cluster: automl-local-regression-Synapse.
Parent Run ID: AutoML_ad8600ab-a1ab-4b6b-b233-059d969e0a0e

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

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

ML job execution result

Get Azure Portal URL for Monitoring Runs

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

Section 508 VPAT

European Union Model Clauses

EU Safe Harbor

United Kingdom G-Cloud

China Multi Layer Protection Scheme

China GB 18030

China CCCPPF

Singapore MTCS Level 3

Australian Signals Directorate

New Zealand GCIO

Japan Financial Services

ENISA IAF

Comprehensive Security

Category	Feature	
Data Protection	Data in Transit	✓
	Data Encryption at Rest	✓
	Data Discovery and Classification	✓
Access Control	Object Level Security (Tables/Views)	✓
	Row Level Security	✓
	Column Level Security	✓
	Dynamic Data Masking	✓
Authentication	SQL Login	✓
	Azure Active Directory	✓
	Multi-Factor Authentication	✓
Network Security	Virtual Networks	✓
	Firewall	✓
	Azure ExpressRoute	✓
Threat Protection	Thread Detection	✓
	Auditing	✓
	Vulnerability Assessment	✓

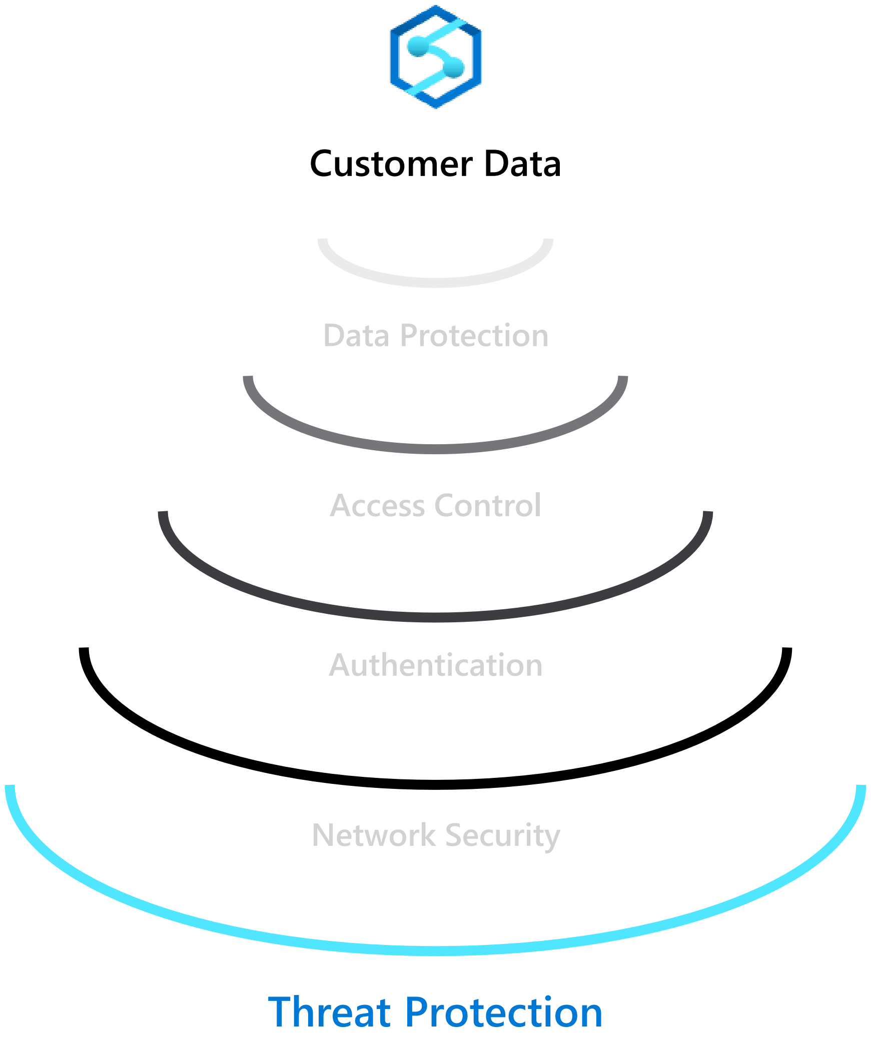
Threat Protection - Business requirements

How do we enumerate and track potential SQL vulnerabilities?

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

How do we discover and alert on suspicious database activity?

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



Customer Data

Data Protection

Access Control

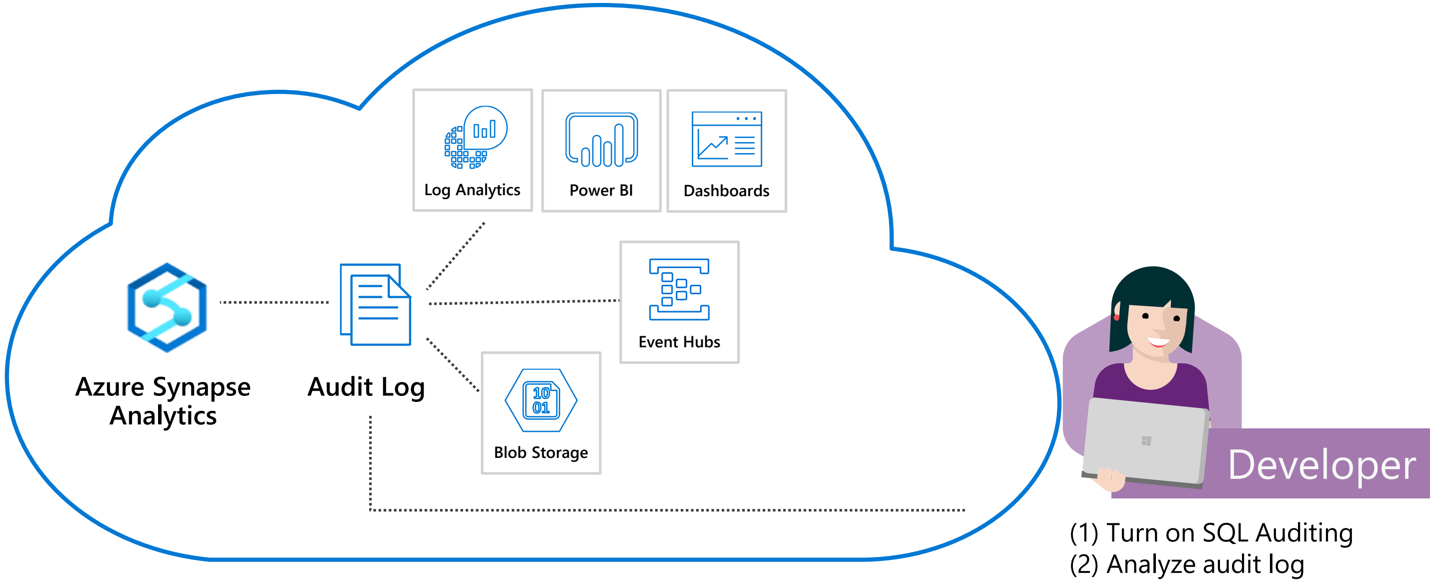
Authentication

Network Security

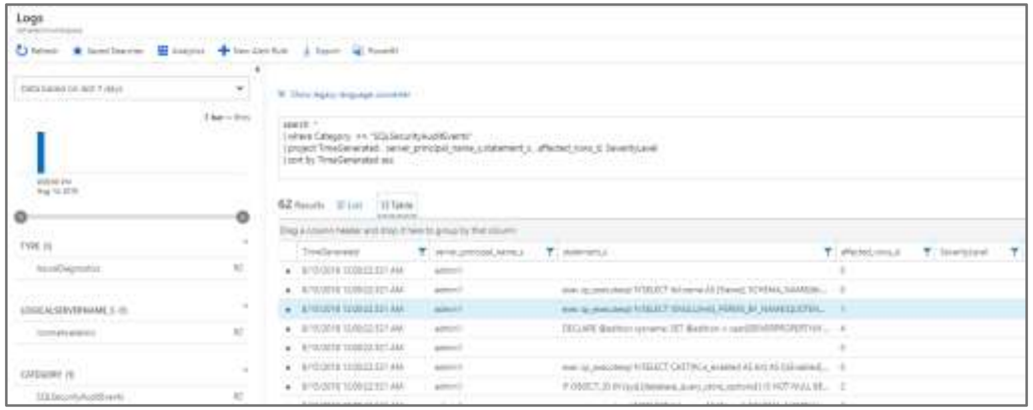
Threat Protection

SQL auditing in Azure Log Analytics and Event Hubs

Gain insight into database audit log



(1) Turn on SQL Auditing
(2) Analyze audit log



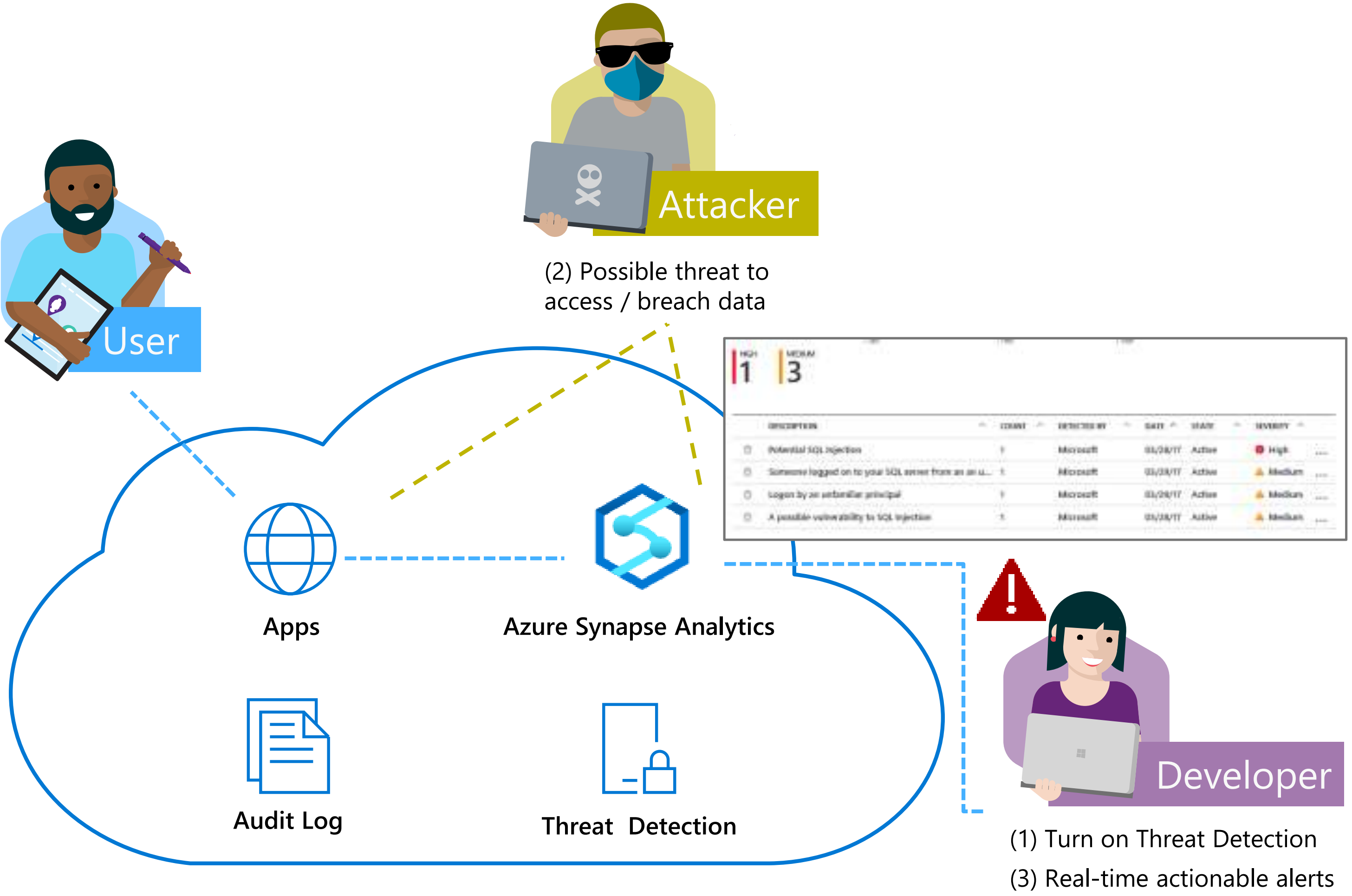
```
search *
| where Category == "SQLSecurityAuditEvents"
| project TimeGenerated, server_principal_name_s, statement_s, affected_rows_d, SeverityLevel
| sort by TimeGenerated asc
```

TimeGenerated	server_principal_name_s	statement_s	affected_rows_d	SeverityLevel
[illegible]	[illegible]		[illegible]	
[illegible]	[illegible]	[illegible]	[illegible]	
[illegible]	[illegible]	[illegible]	[illegible]	
[illegible]	[illegible]	DECLARE [illegible]	[illegible]	
[illegible]	[illegible]		[illegible]	
[illegible]	[illegible]	[illegible]	[illegible]	
[illegible]	[illegible]	[illegible]	[illegible]	

- ✓ 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



(2) Possible threat to access / breach data

	DESCRIPTION	COUNT	DETECTED BY	DATE	STATE	SEVERITY	
□	Potential SQL injection	1	Microsoft	03/28/17	Active	● High	...
□	Someone logged on to your SQL server from an u...	1	Microsoft	03/28/17	Active	▲ Medium	...
□	Logon by an unfamiliar principal	1	Microsoft	03/28/17	Active	▲ Medium	...
□	A possible vulnerability to SQL injection	1	Microsoft	03/28/17	Active	▲ Medium	...

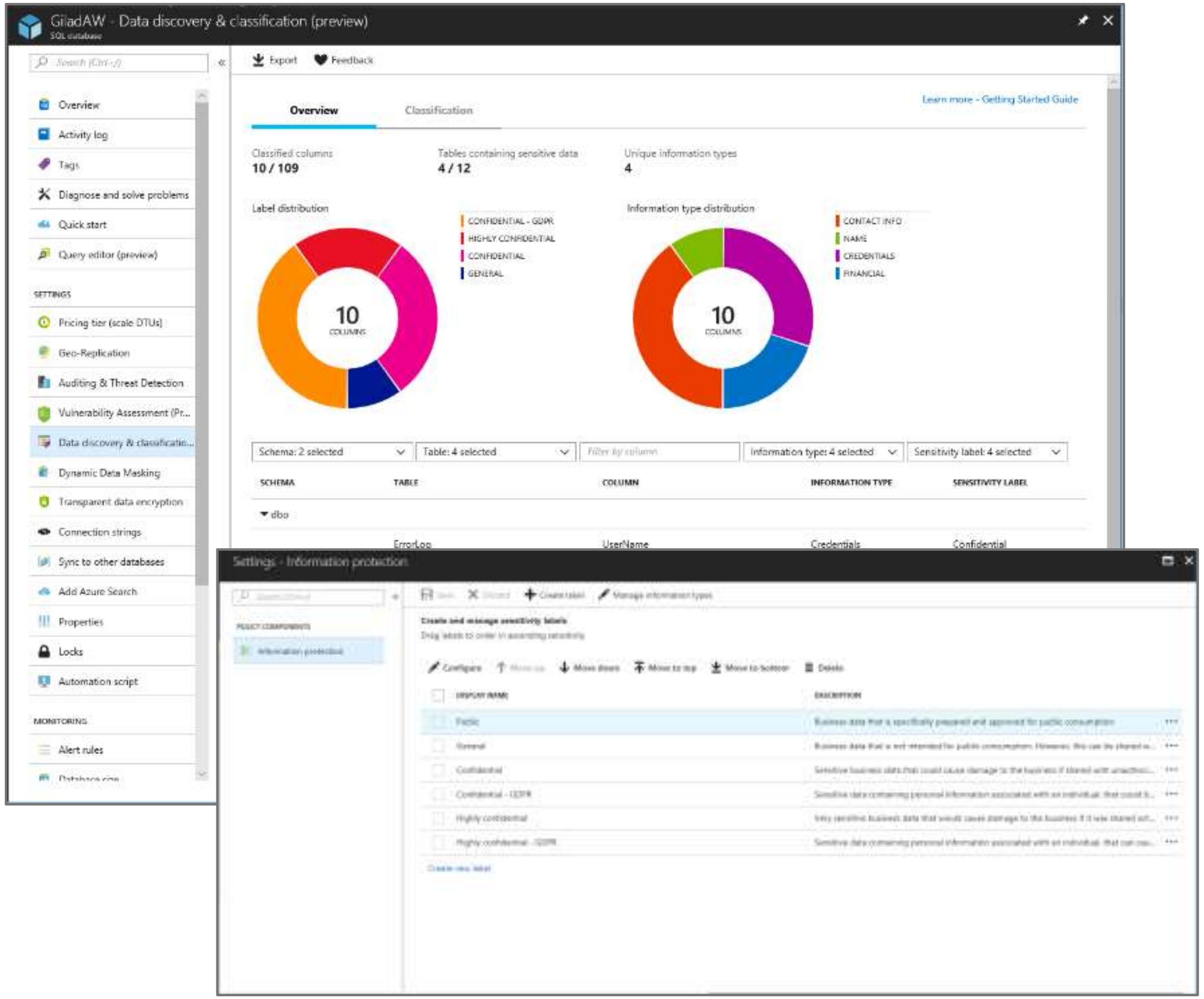
(1) Turn on Threat Detection

(3) Real-time actionable alerts

- ✓ 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



SCHEMA	TABLE	COLUMN	INFORMATION TYPE	SENSITIVITY LABEL
dbo				
	ErrorLog	UserName	Credentials	Confidential

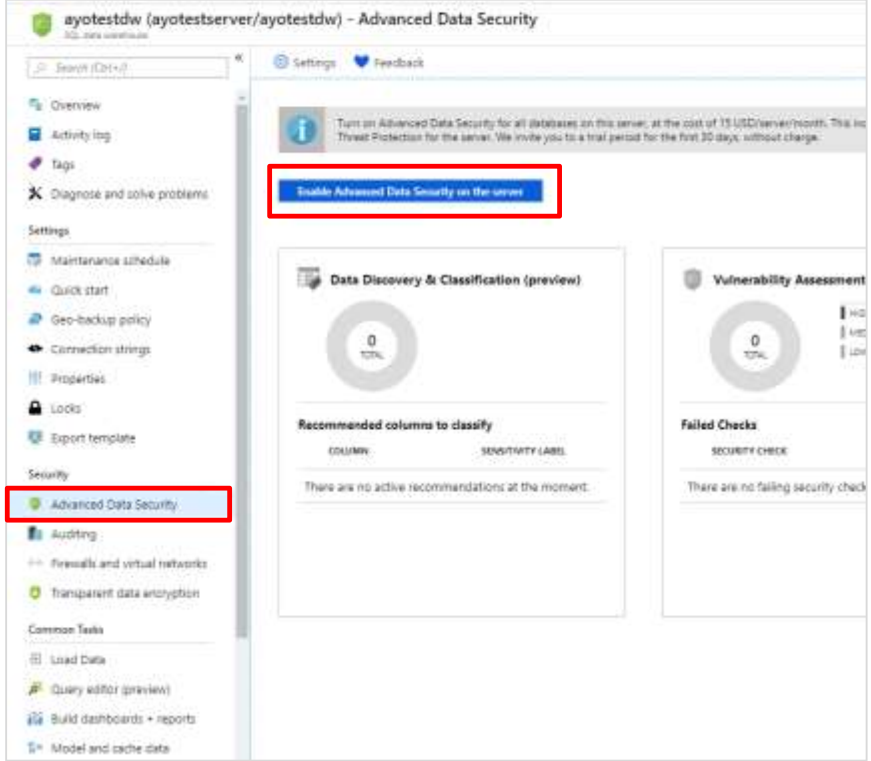
Create and manage sensitivity labels
Drag labels to order in ascending sensitivity

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 w...
Confidential	Sensitive business data that could cause damage to the business if shared with unauthori...
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 wit...
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



ayotestdw (ayotestserver/ayotestdw) - Advanced Data Security

Turn on Advanced Data Security for all databases on this server, at the cost of 15 USD/server/month. This inc
Threat Protection for the server. We invite you to a trial period for the first 30 days, without charge.

Enable Advanced Data Security on the server

Data Discovery & Classification (preview)

Recommended columns to classify

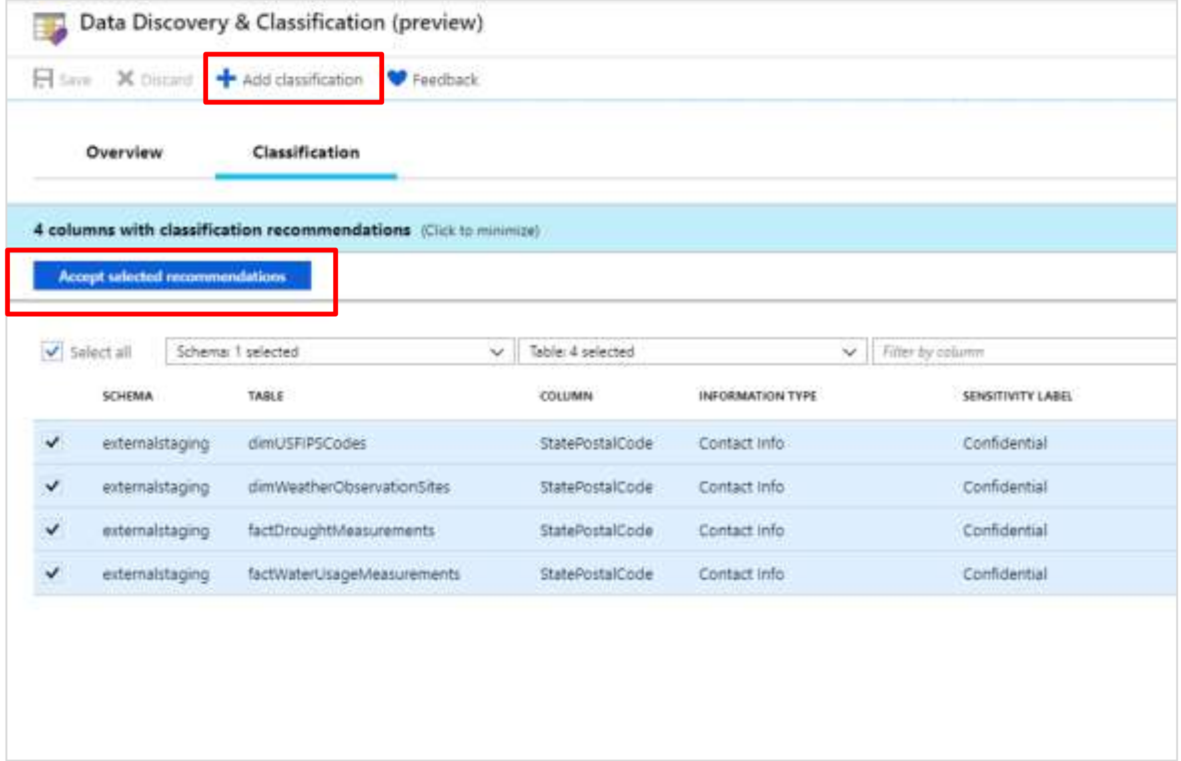
COLUMN	SENSITIVITY LABEL
There are no active recommendations at the moment.	

Vulnerability Assessment

Failed Checks

SECURITY CHECK
There are no failing security check

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



Data Discovery & Classification (preview)

4 columns with classification recommendations (Click to minimize)

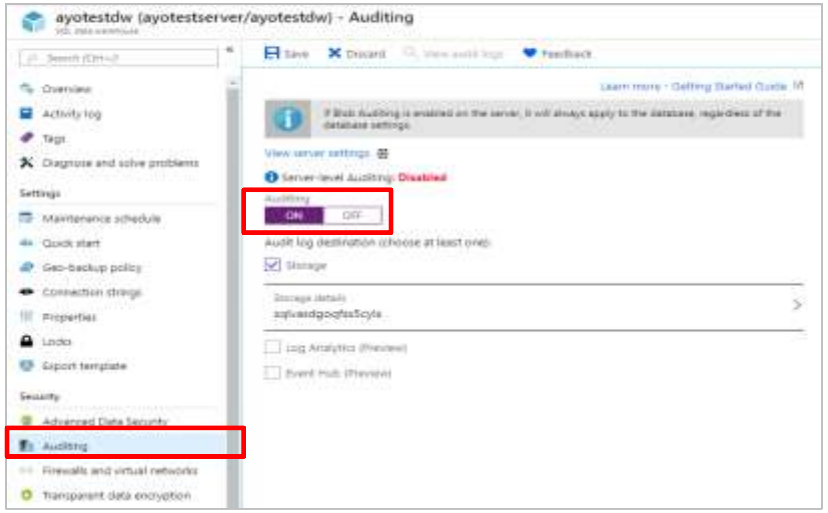
Accept selected recommendations

☑ Select all | Schema: 1 selected | Table: 4 selected | Filter by column

	SCHEMA	TABLE	COLUMN	INFORMATION TYPE	SENSITIVITY LABEL
✓	externalstaging	dimUSFIPSCodes	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.



ayotestdw (ayotestserver/ayotestdw) - Auditing

If Blob Auditing is enabled on the server, it will always apply to the database, regardless of the database settings.

Server-level Auditing: Disabled

Auditing: ON | OFF

Audit log destination (choose at least one):

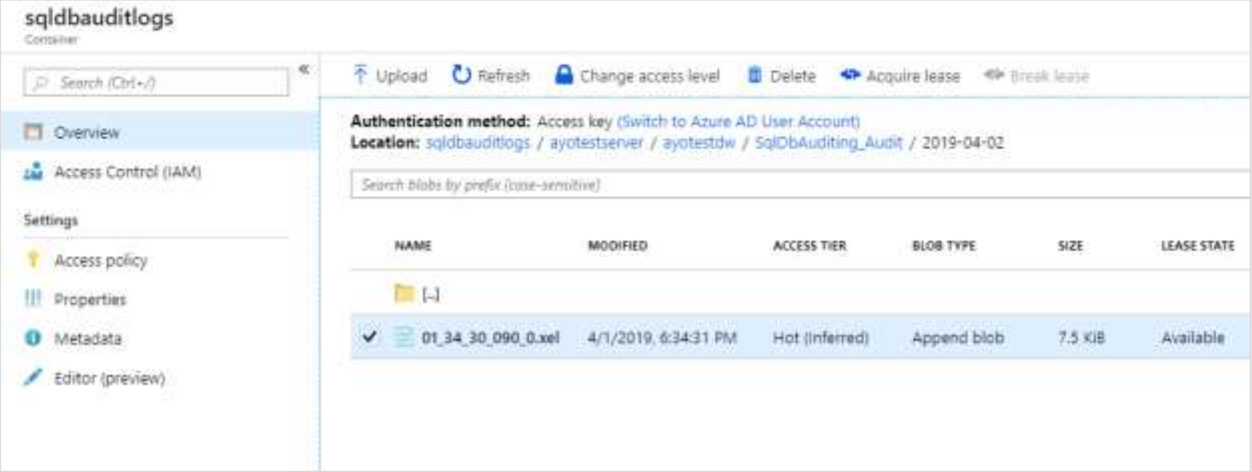
☑ Storage

Storage details: sqlvasdgoqfss5cyle

□ Log Analytics (Preview)

□ Event Hub (Preview)

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

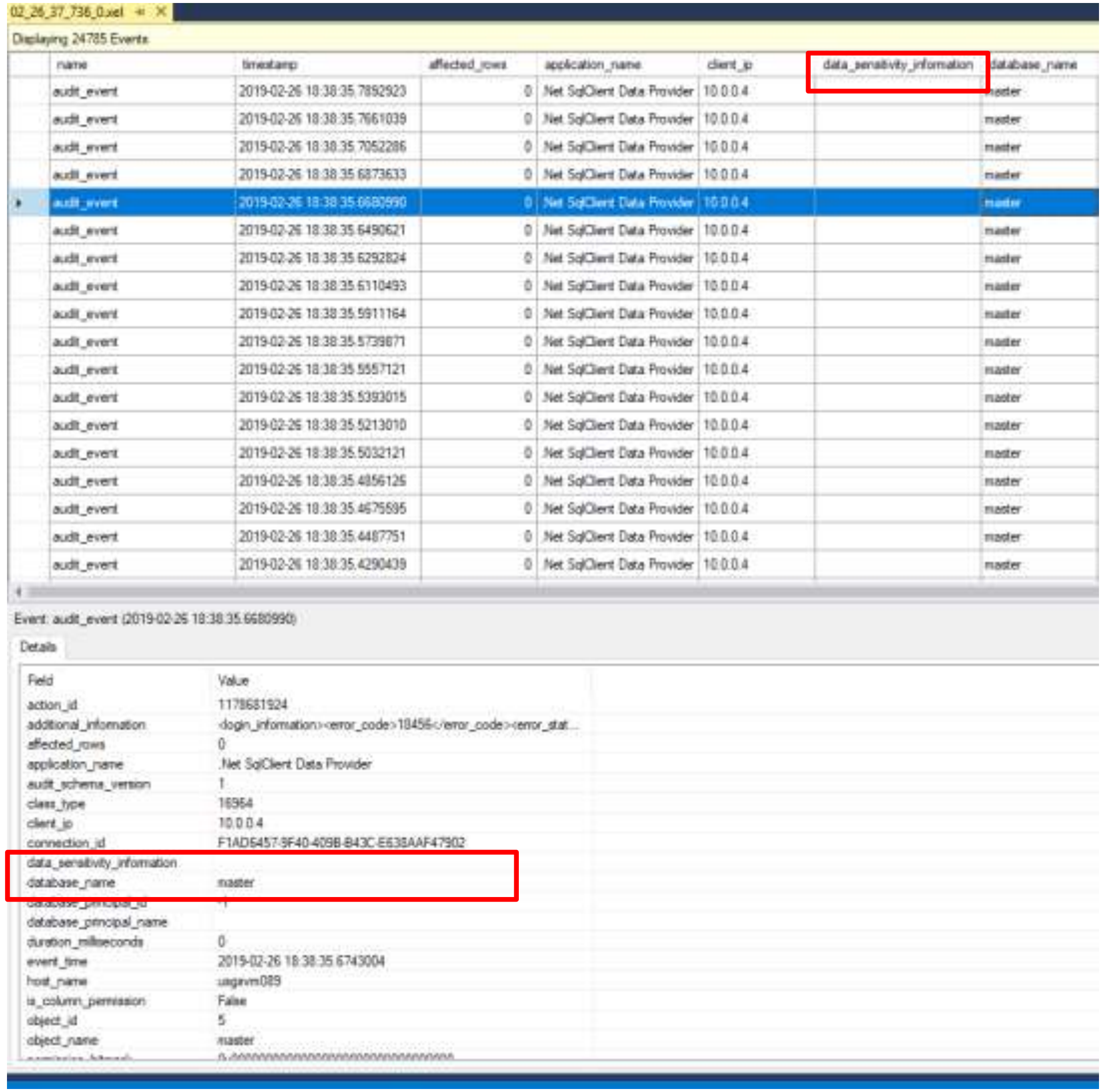


sqldbauditlogs

Authentication method: Access key (Switch to Azure AD User Account)
Location: sqldbauditlogs / ayotestserver / ayotestdw / SqlDbAuditing_Audit / 2019-04-02

NAME	MODIFIED	ACCESS TIER	BLOB TYPE	SIZE	LEASE STATE
[..]					
✓ 01_34_30_090_0.xel	4/1/2019, 6:34:31 PM	Hot (Inferred)	Append blob	7.5 KiB	Available

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



02_26_37_736_0.xel

Displaying 24785 Events

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 18:38:35.4856126	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)

Details

Field	Value
action_id	1178681924
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-E638AAF47902
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	usgavm089
is_column_permission	False
object_id	5
object_name	master
permission_bitmask	0x00000000000000000000000000000000

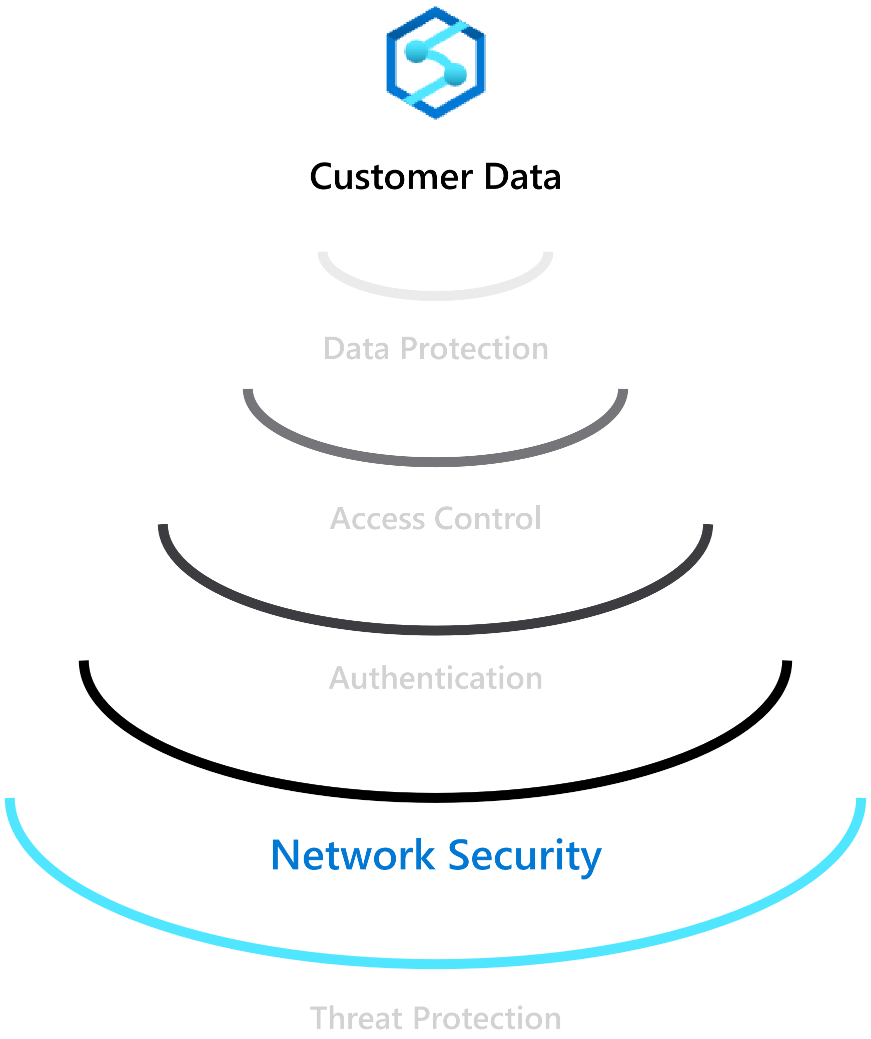
Network Security - Business requirements

How do we implement network isolation?

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

How do we achieve separation?

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



Customer Data

Data Protection

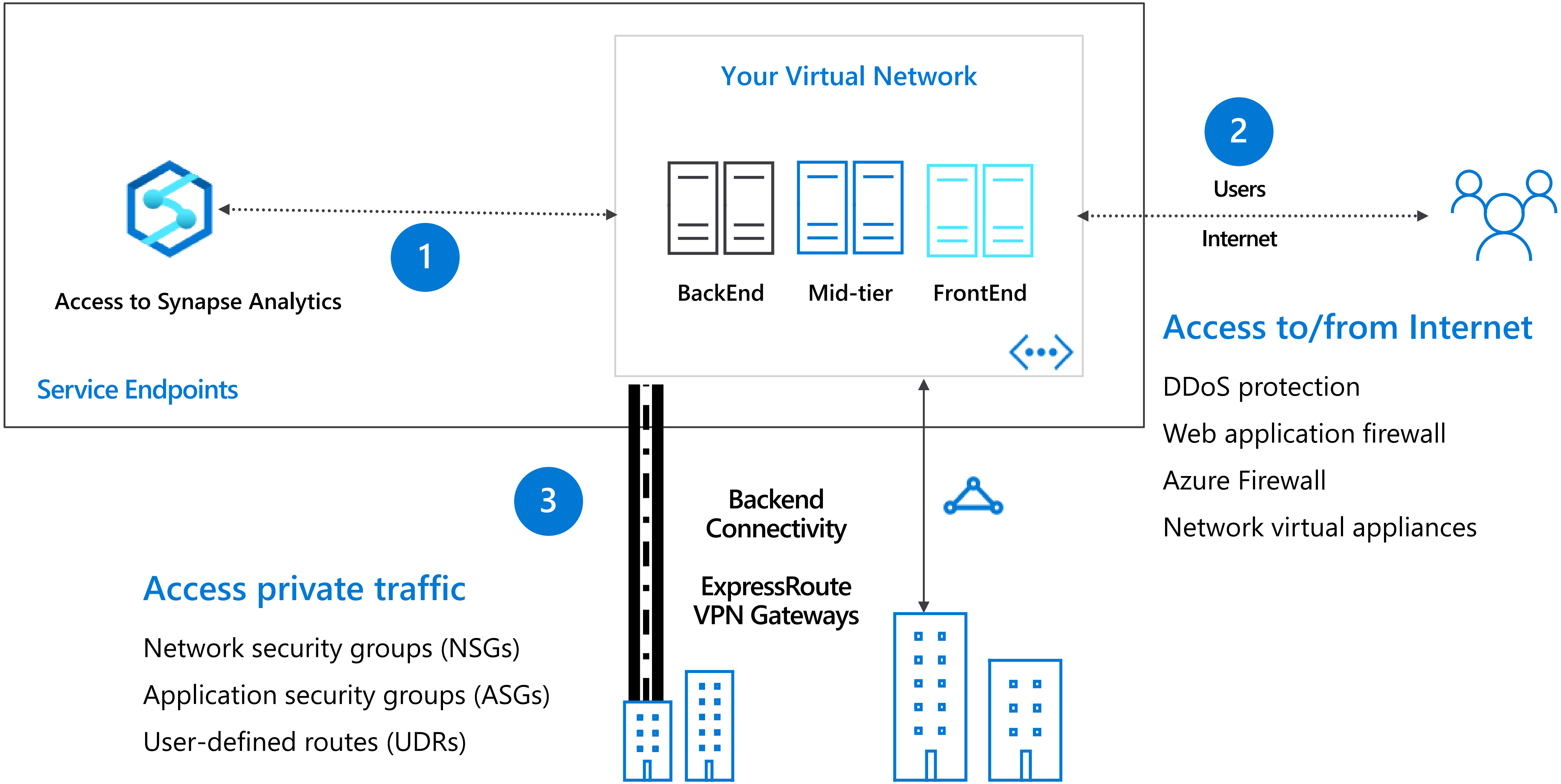
Access Control

Authentication

Network Security

Threat Protection

Azure networking: application-access patterns



Access to/from Internet

DDoS protection

Web application firewall

Azure Firewall

Network virtual appliances

Access private traffic

Network security groups (NSGs)

Application security groups (ASGs)

User-defined routes (UDRs)

Securing with firewalls

Overview

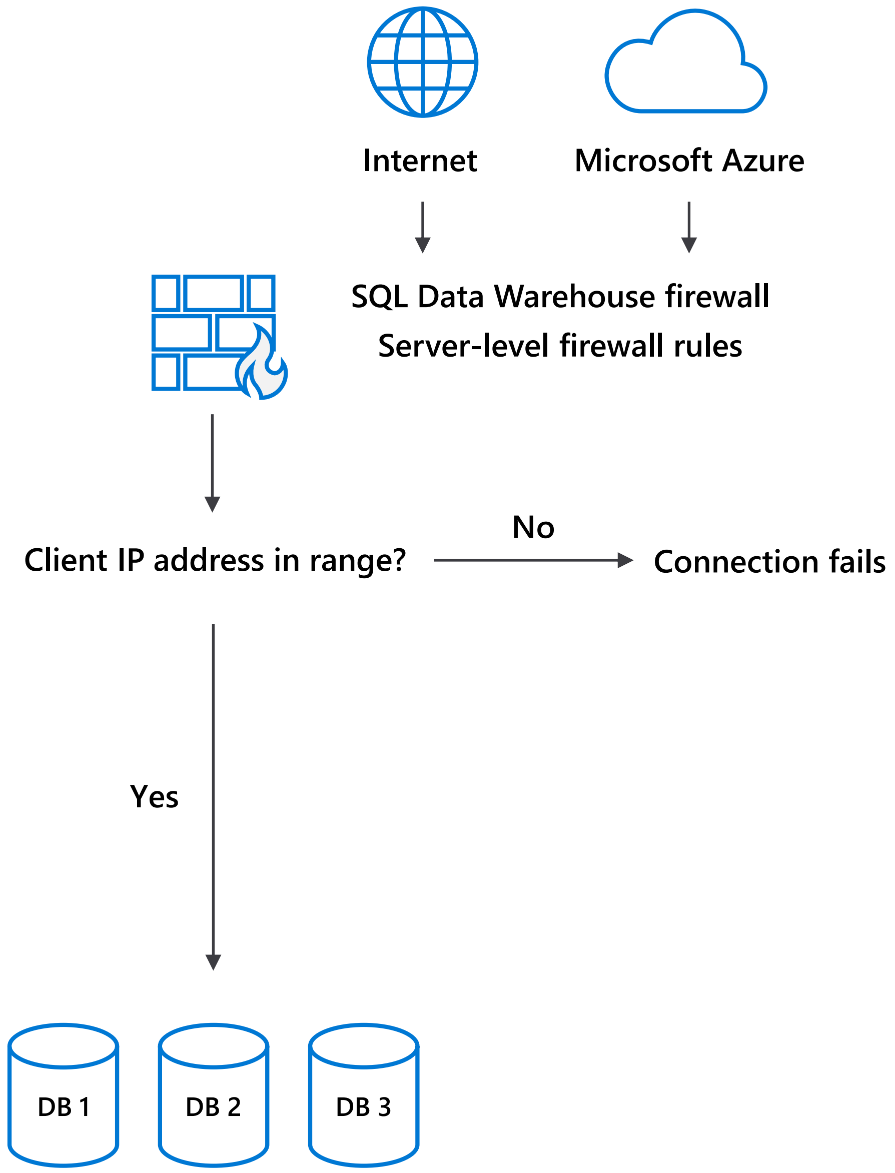
By default, all access to your Azure Synapse Analytics is blocked by the firewall.

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

Rules

Allow specific or range of whitelisted IP addresses.

Allow Azure applications to connect.

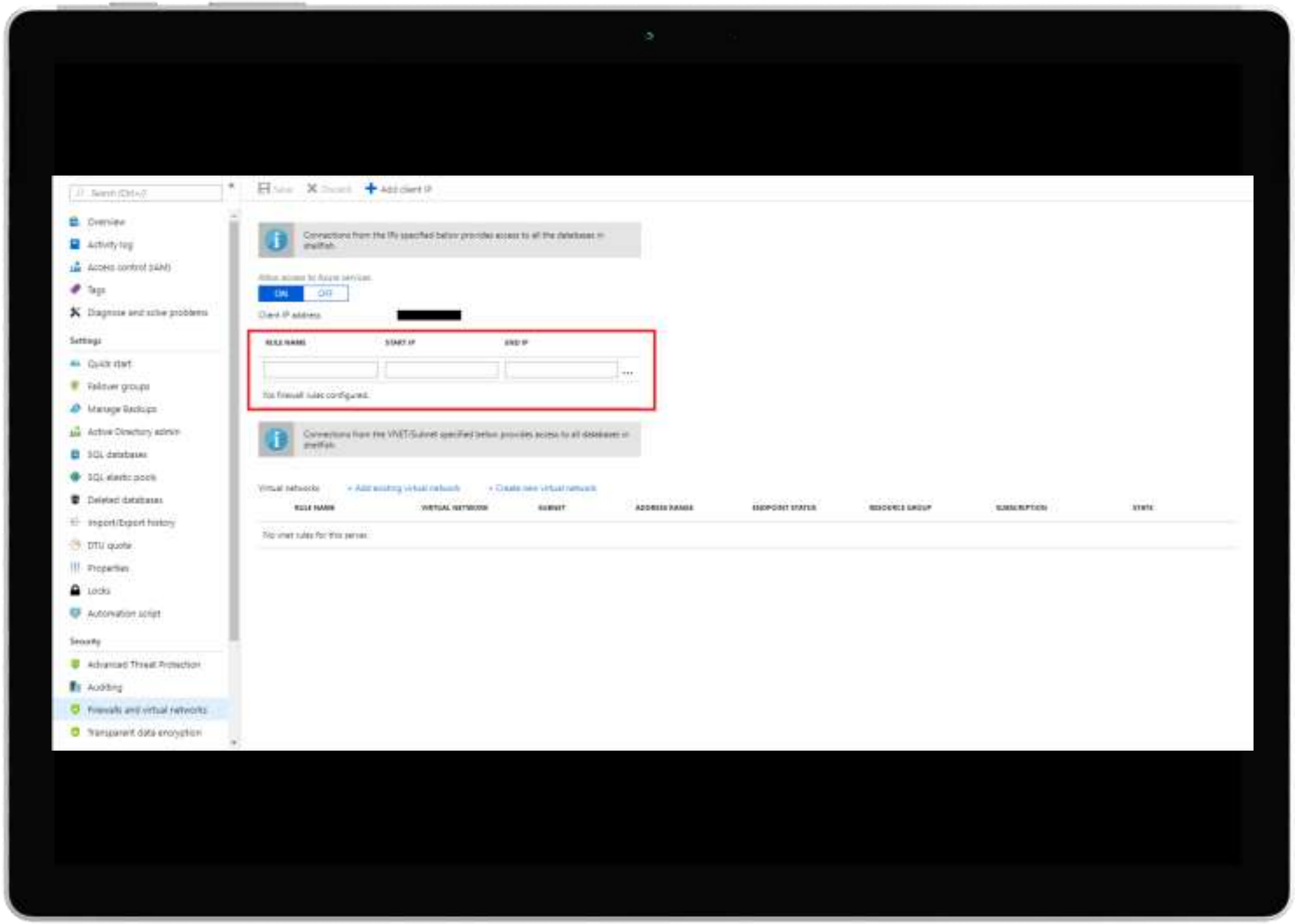

Internet
Microsoft Azure
SQL Data Warehouse firewall
Server-level firewall rules
Client IP address in range?
No
Connection fails
Yes
DB 1
DB 2
DB 3

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



Connections from the IPs specified below provides access to all the databases in shellfish.

Allow access to Azure services: [✓ ON] OFF

Client IP address [redacted]

RULE NAME	START IP	END IP

No firewall rules configured.

Connections from the VNET/Subnet specified below provides access to all databases in shellfish.

Virtual networks

RULE NAME	VIRTUAL NETWORK	SUBNET	ADDRESS RANGE	ENDPOINT STATUS	RESOURCE GROUP	SUBSCRIPTION	STATE

No vnet rules for this server.

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/{subscriptionI
d}/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/{subscriptionI
d}/resourceGroups/{resourceGroupName}/providers/Microsoft
.Sql/servers/{serverName}/firewallRules/{firewallRuleName
}?api-version=2014-04-01

GET
https://management.azure.com/subscriptions/{subscriptionI
d}/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:

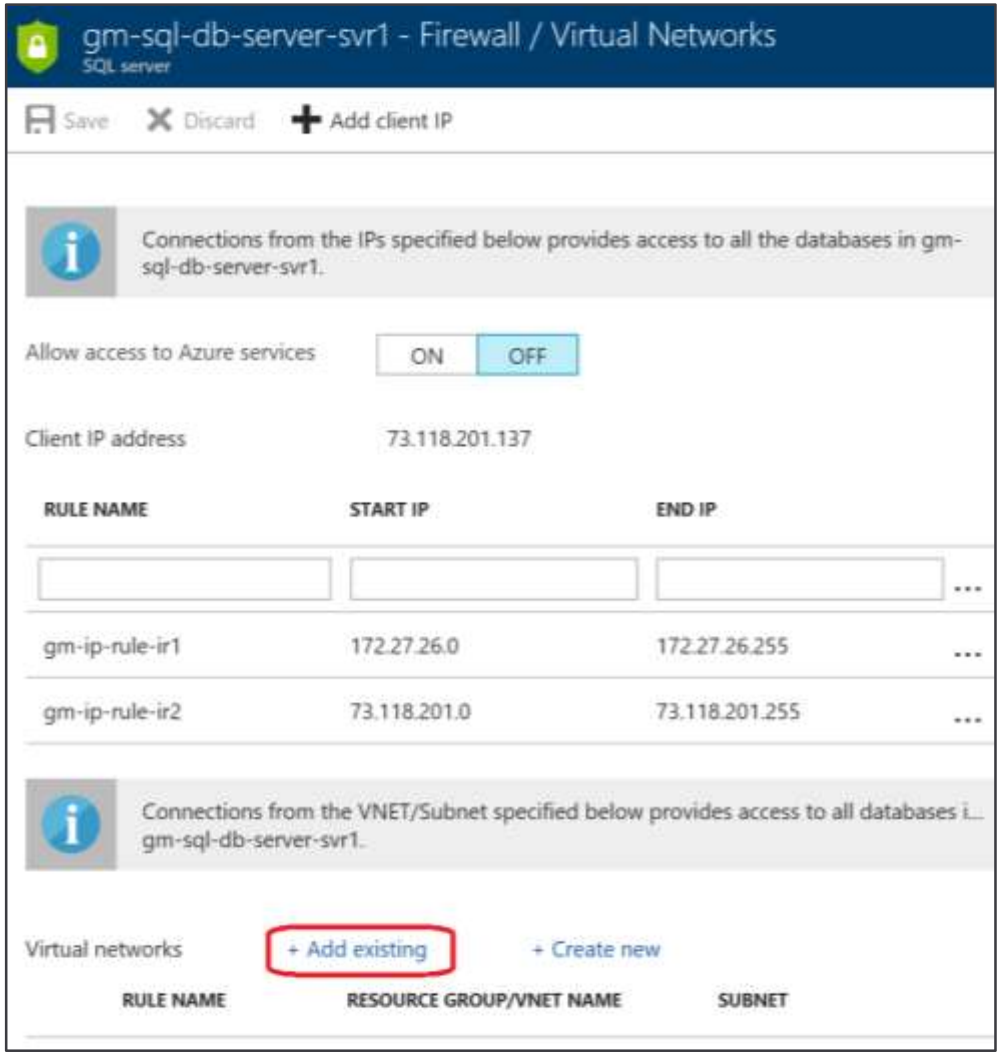
Azure Synapse Analytics Resource:
Server name > Firewalls and virtual networks

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.



gm-sql-db-server-svr1 - Firewall / Virtual Networks
SQL server

Connections from the IPs specified below provides access to all the databases in gm-sql-db-server-svr1.

Allow access to Azure services ON [✓ OFF]

Client IP address 73.118.201.137

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

Connections from the VNET/Subnet specified below provides access to all databases i... gm-sql-db-server-svr1.

Virtual networks + Add existing + Create new

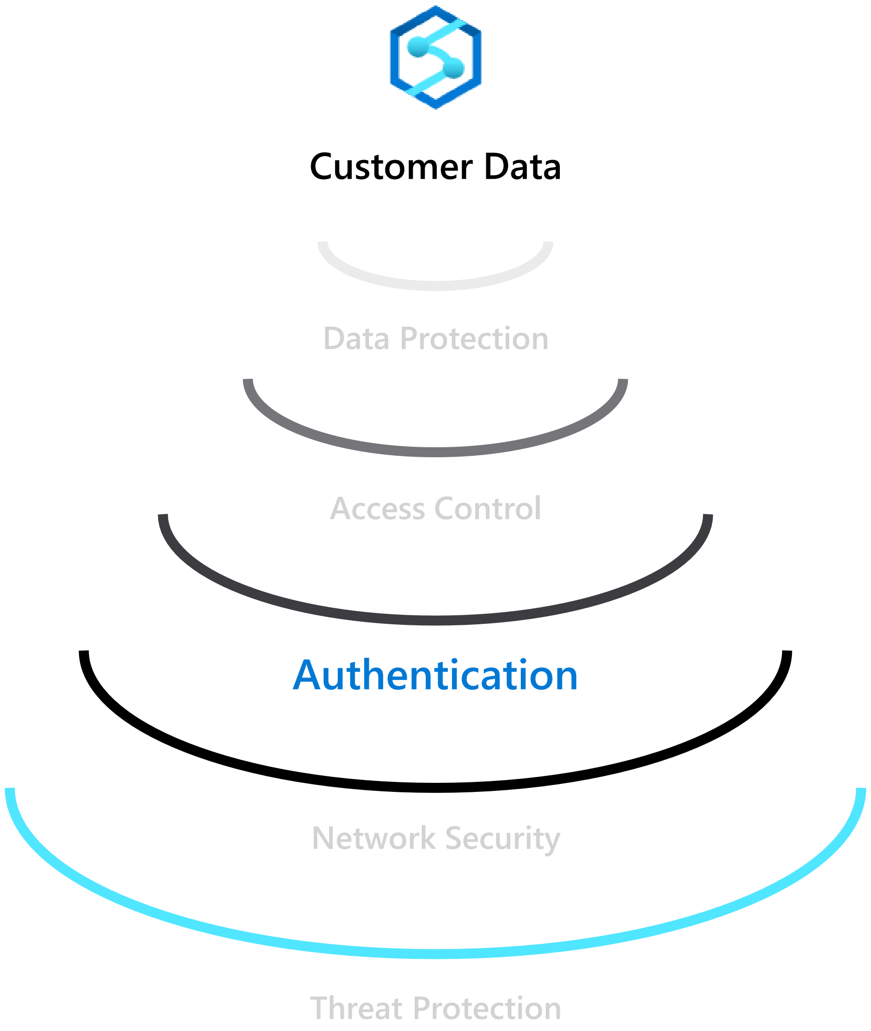
RULE NAME	RESOURCE GROUP/VNET NAME	SUBNET

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.

Benefits

Alternative to SQL Server authentication

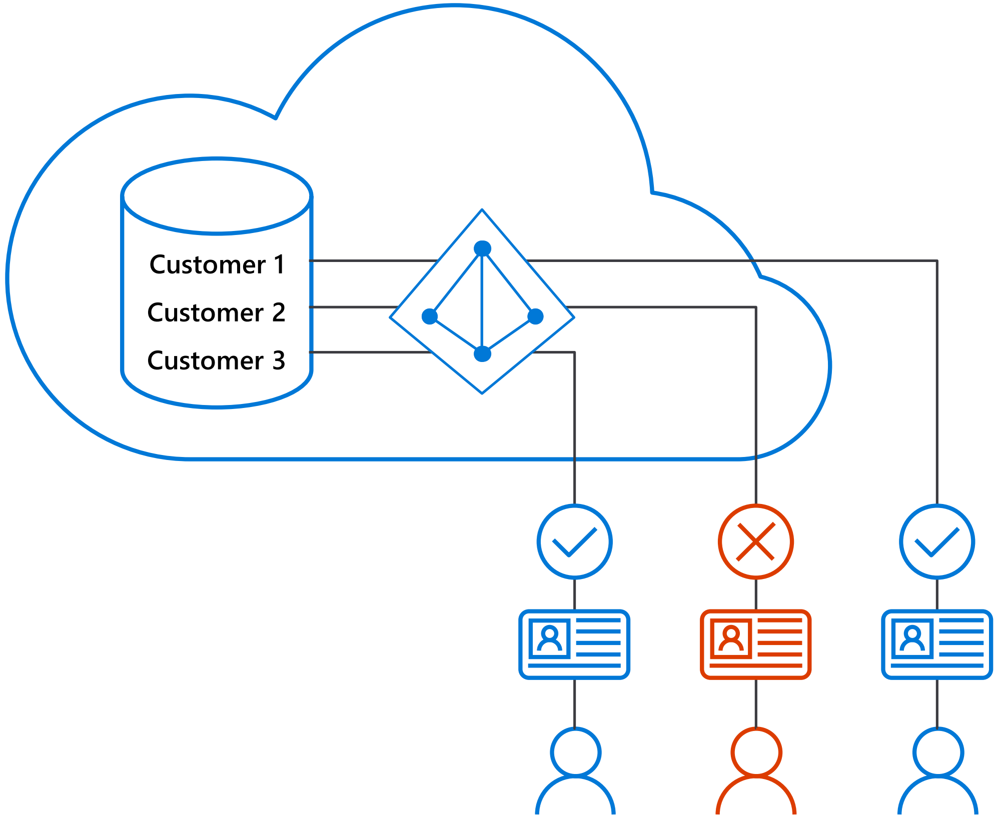
Limits proliferation of user identities across databases

Allows password rotation in a single place

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

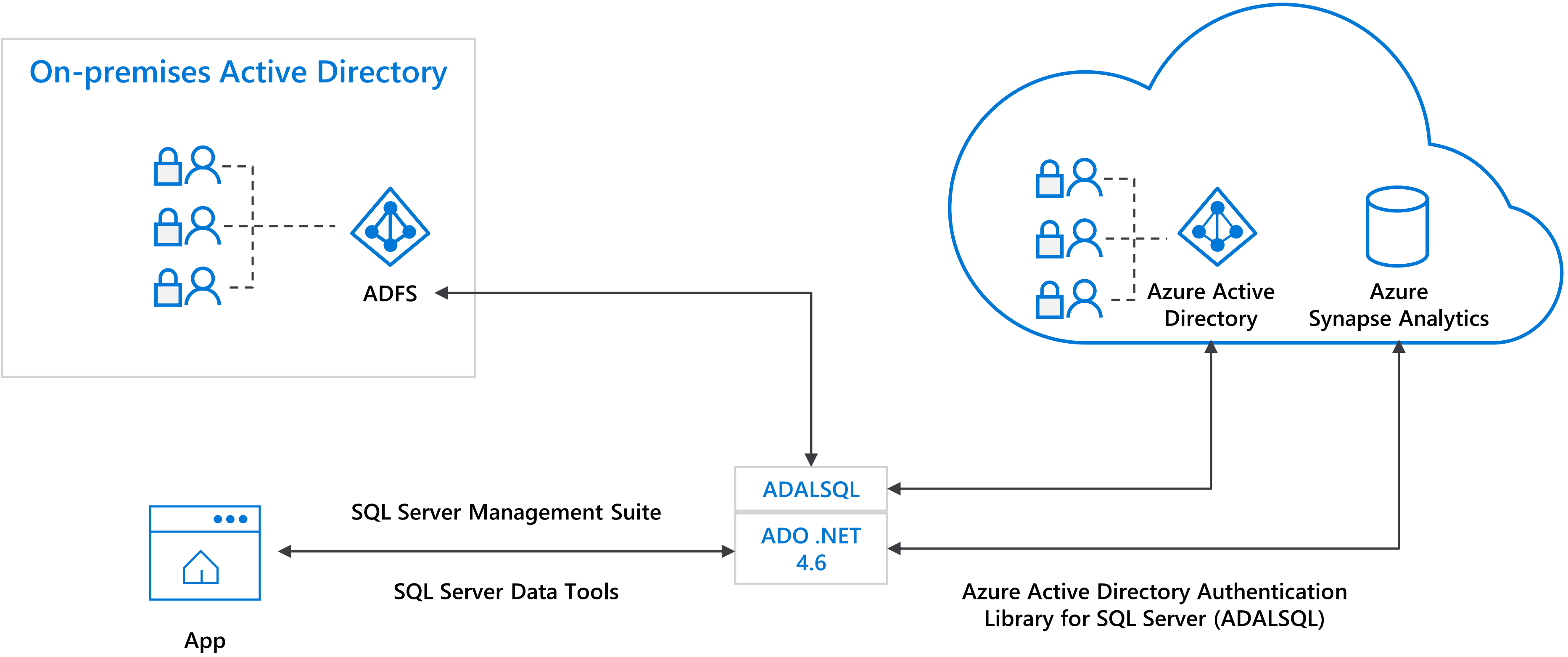
Eliminates the need to store passwords

Azure Synapse Analytics



Azure Active Directory trust architecture

Azure Active Directory and Azure Synapse Analytics


On-premises Active Directory
ADFS
Azure Active Directory
Azure Synapse Analytics
ADALSQL
ADO .NET 4.6
SQL Server Management Suite
SQL Server Data Tools
App
Azure Active Directory Authentication Library for SQL Server (ADALSQL)

SQL authentication

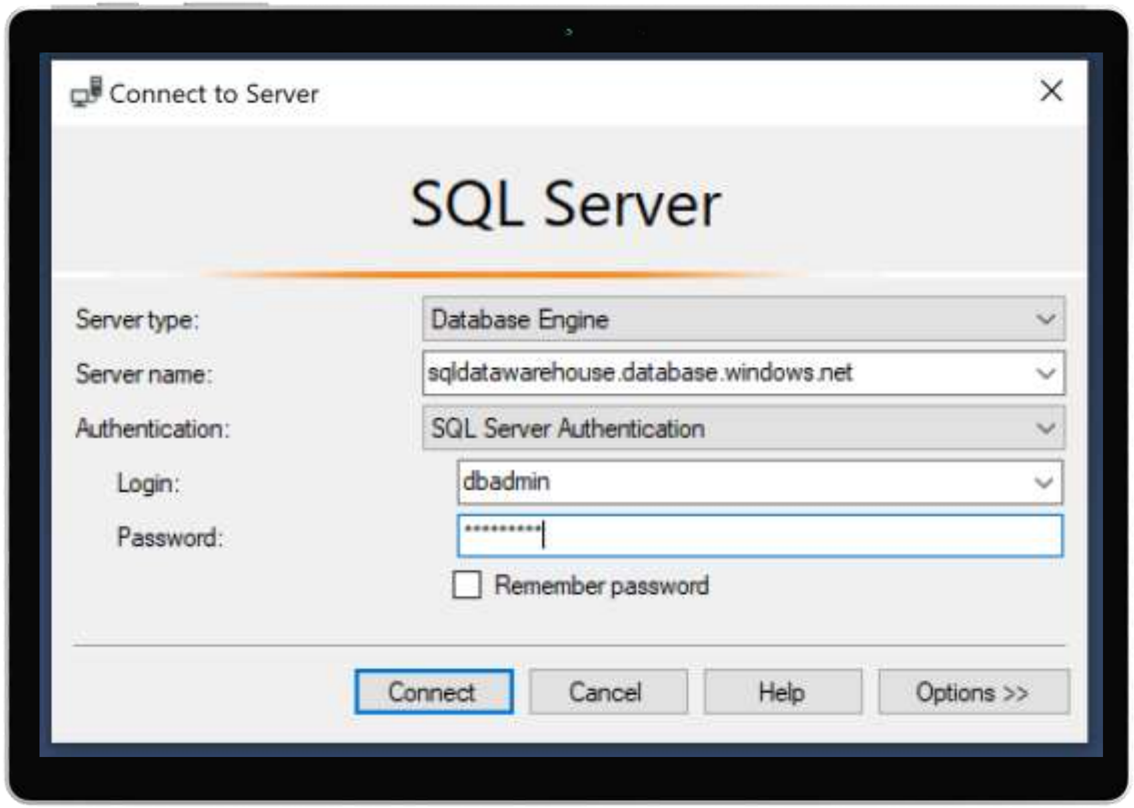
Overview

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 Server

SQL Server

Server type: Database Engine

Server name: sqldatawarehouse.database.windows.net

Authentication: SQL Server Authentication

Login: dbadmin

Password: *********

□ Remember password

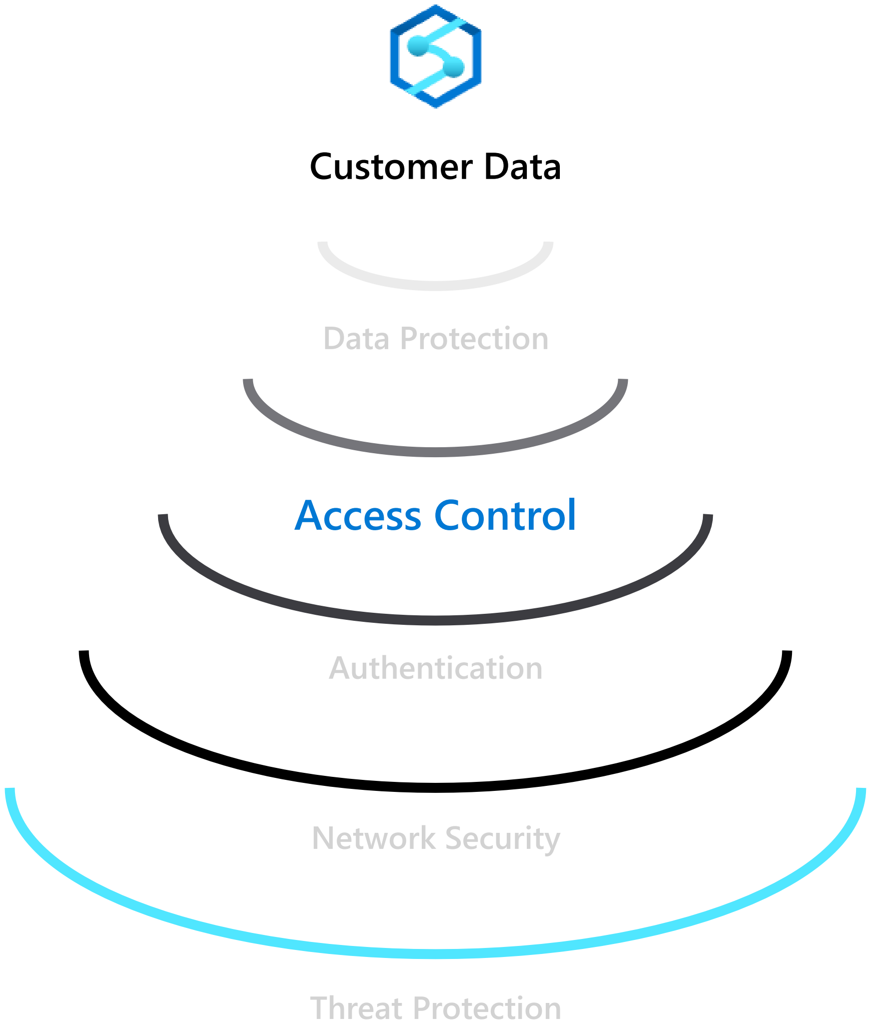
```
-- Connect to master database and create a login
CREATE LOGIN ApplicationLogin WITH PASSWORD = 'Str0ng_password';
CREATE USER ApplicationUser FOR LOGIN ApplicationLogin;

-- 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

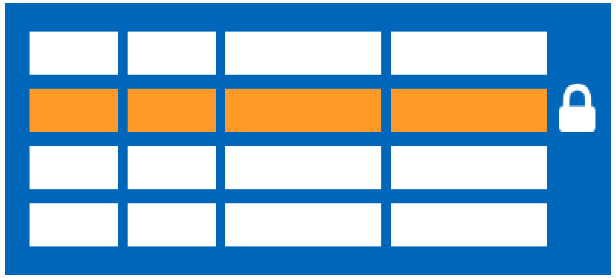
Fine grained access control of specific rows in a database table.

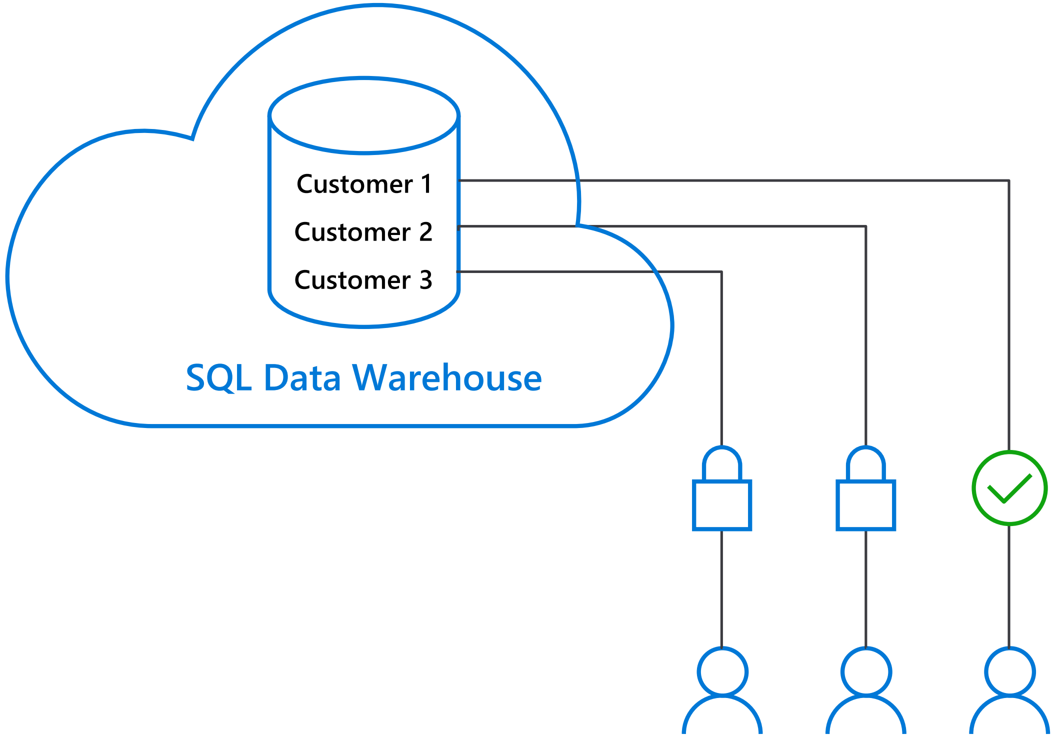
Help prevent unauthorized access when multiple users share the same tables.

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).

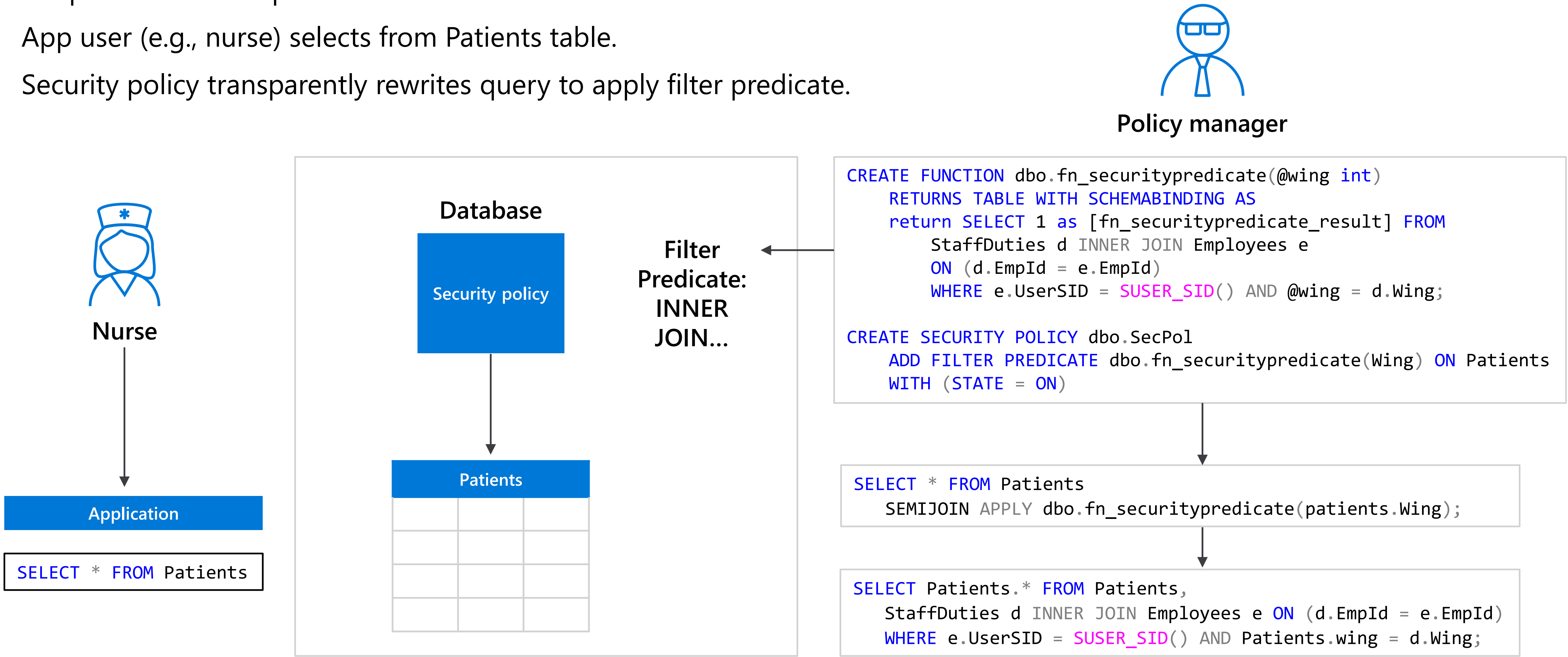
The following examples demonstrate the use of the CREATE SECURITY POLICY syntax

```
-- 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.



```
SELECT * FROM Patients
```

```
CREATE FUNCTION dbo.fn_securitypredicate(@wing int)
    RETURNS TABLE WITH SCHEMABINDING AS
    return SELECT 1 as [fn_securitypredicate_result] FROM
        StaffDuties d INNER JOIN Employees e
        ON (d.EmpId = e.EmpId)
        WHERE e.UserSID = SUSER_SID() AND @wing = d.Wing;

CREATE SECURITY POLICY dbo.SecPol
    ADD FILTER PREDICATE dbo.fn_securitypredicate(Wing) ON Patients
    WITH (STATE = ON)
```

```
SELECT * FROM Patients
   SEMIJOIN APPLY dbo.fn_securitypredicate(patients.Wing);
```

```
SELECT Patients.* FROM Patients,
   StaffDuties d INNER JOIN Employees e ON (d.EmpId = e.EmpId)
   WHERE e.UserSID = SUSER_SID() AND Patients.wing = d.Wing;
```

Column-level security

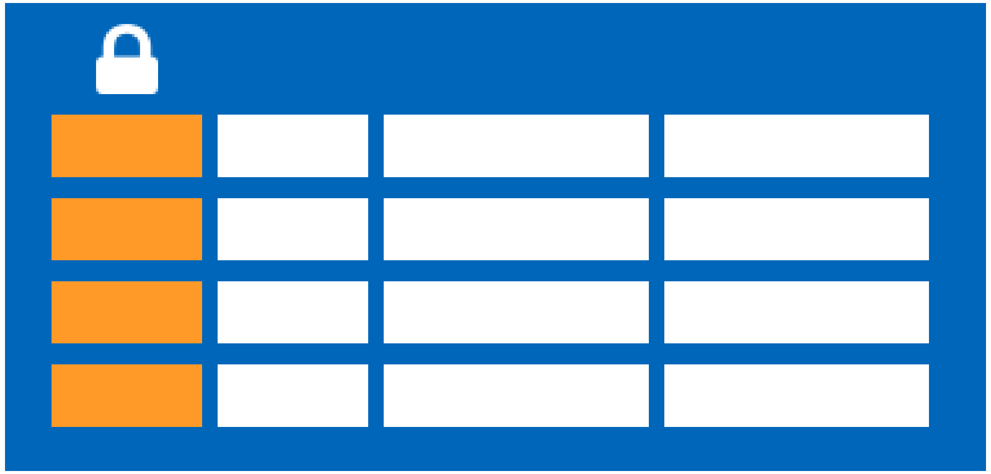
Overview

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.

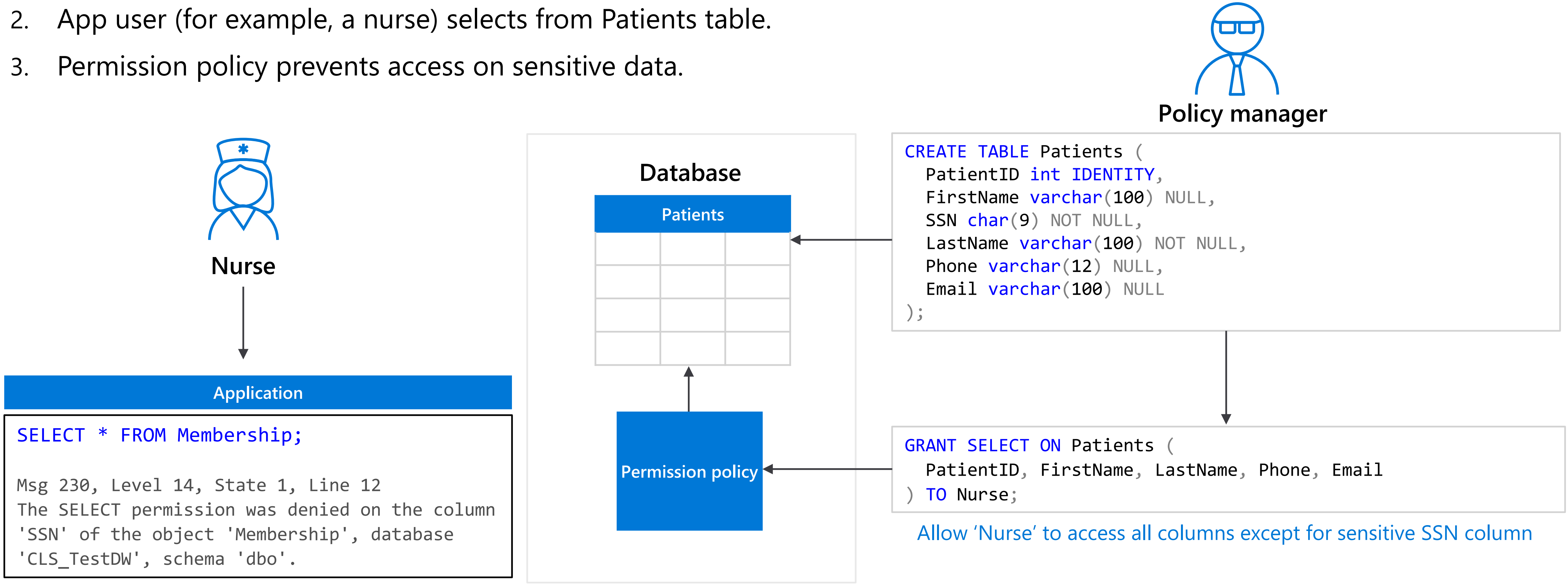
Both Azure Active Directory (AAD) and SQL authentication are supported.



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.



```sql
CREATE TABLE Patients (
  PatientID int IDENTITY,
  FirstName varchar(100) NULL,
  SSN char(9) NOT NULL,
  LastName varchar(100) NOT NULL,
  Phone varchar(12) NULL,
  Email varchar(100) NULL
);
```

```sql
GRANT SELECT ON Patients (
  PatientID, FirstName, LastName, Phone, Email
) TO Nurse;
```

Allow ‘Nurse’ to access all columns except for sensitive SSN column

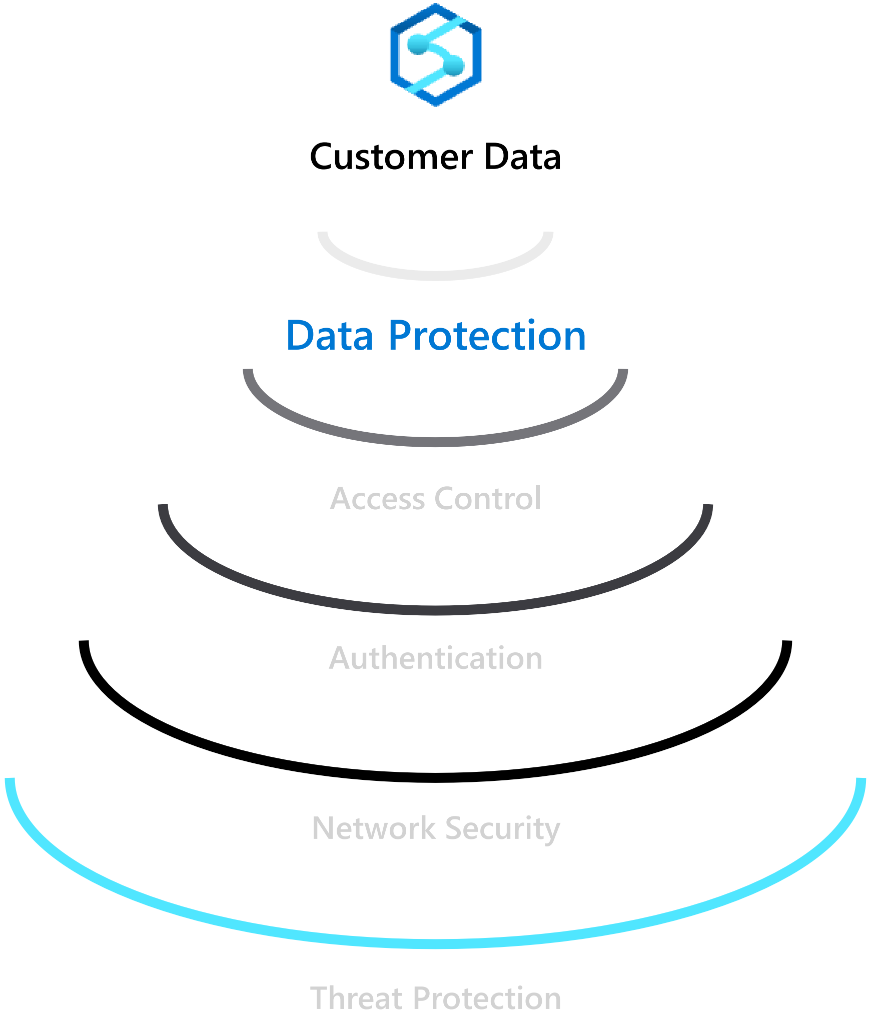
```
SELECT * FROM Membership;

Msg 230, Level 14, State 1, Line 12
The SELECT permission was denied on the column
'SSN' of the object 'Membership', database
'CLS_TestDW', schema 'dbo'.
```

Queries executed as ‘Nurse’ will fail if they include the SSN column

Data Protection - Business requirements

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

What key management options do I have?



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.)

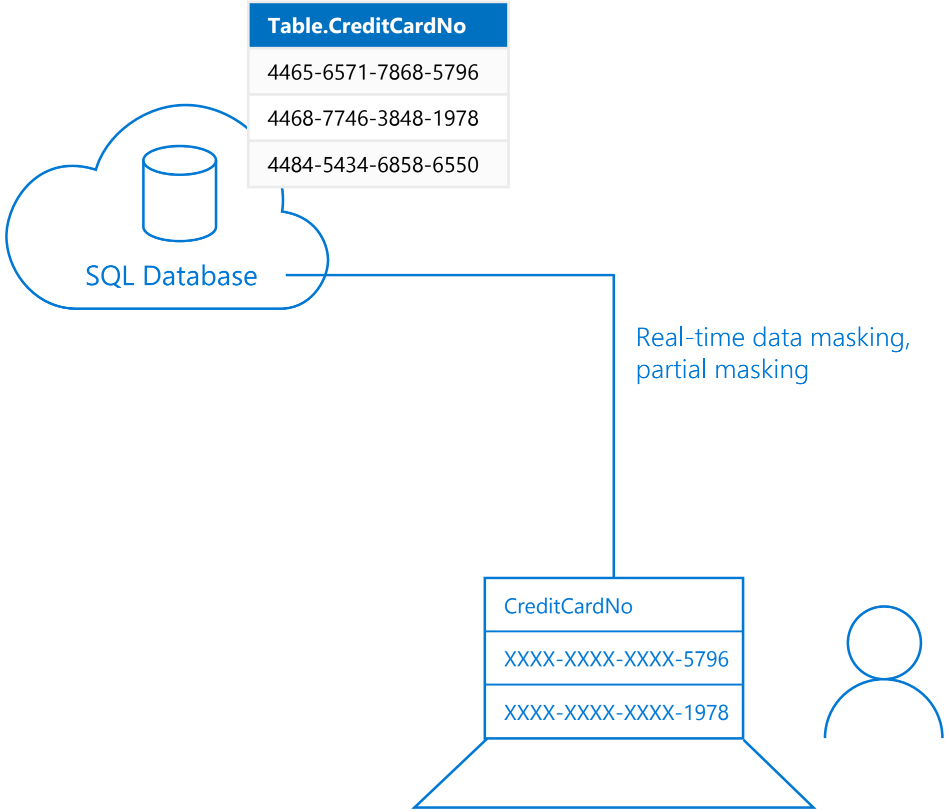


Table.CreditCardNo
4465-6571-7868-5796
4468-7746-3848-1978
4484-5434-6858-6550

CreditCardNo
XXXX-XXXX-XXXX-5796
XXXX-XXXX-XXXX-1978

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

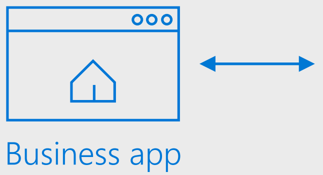
3. The dynamic data masking policy obfuscates the sensitive data in the query results for non-privileged users

Security officer

1

```
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
```

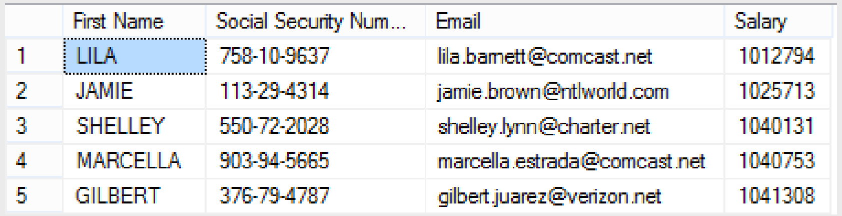
2



```
SELECT [First Name],
       [Social Security Number],
       [Email],
       [Salary]
FROM   [Employee]
```

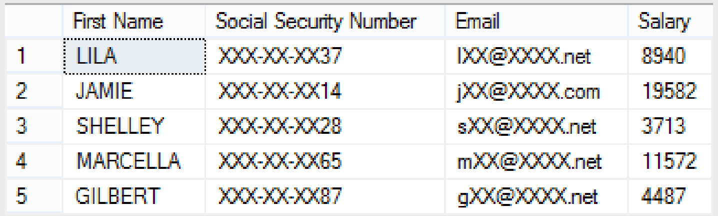
3

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

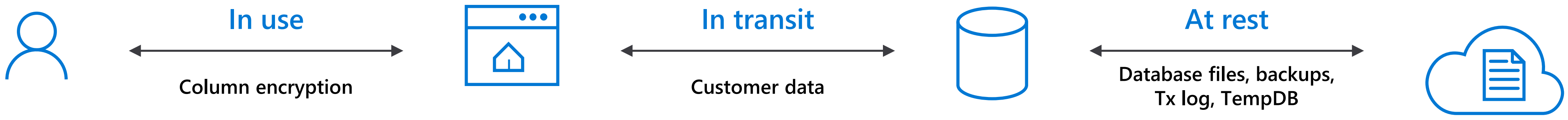
Masked data (admin1 login)



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

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.

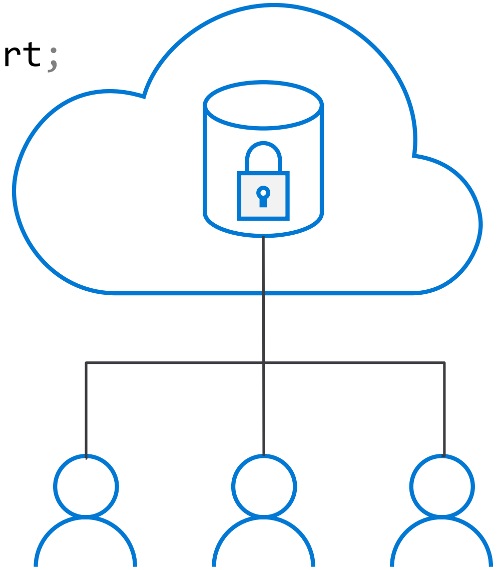
Service OR User managed keys.

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

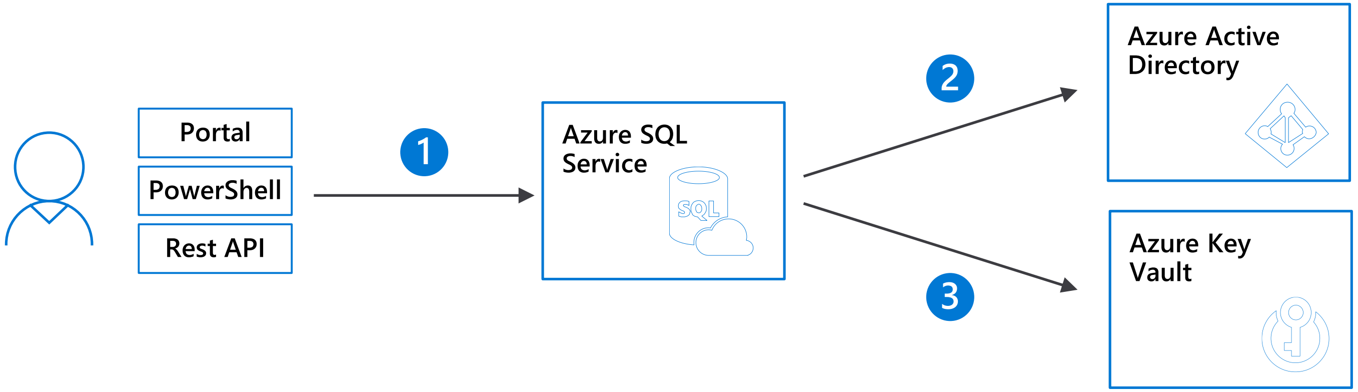
Benefits with User Managed Keys

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.



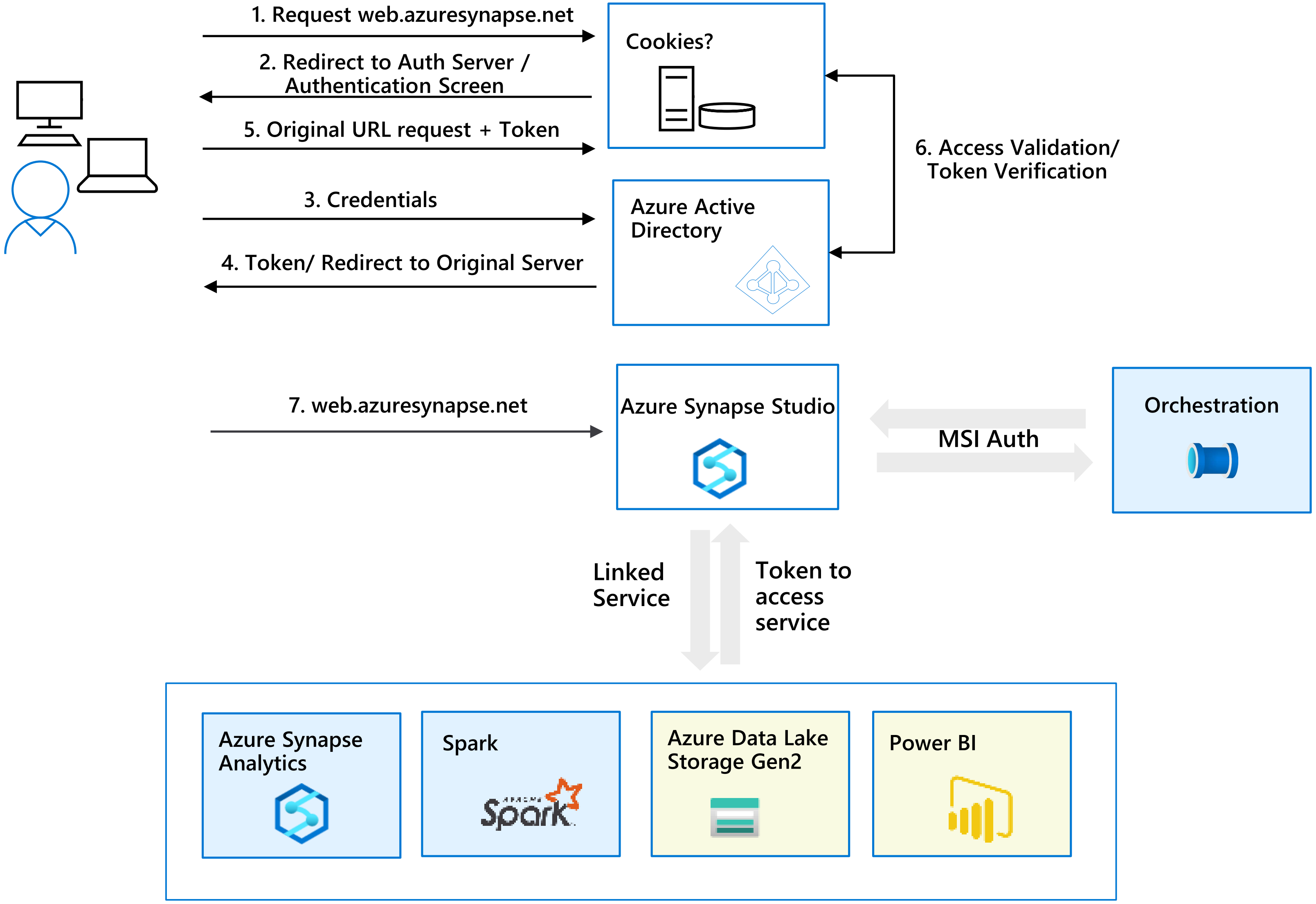
1 The Key Vault admin grants vault access to the SQL Database server using its unique Azure Active Directory (AD) identity

2 The server uses its Azure AD identity to authenticate with Azure AD for access to your Key Vault

3 The server sends get, wrap key, and unwrap key request to the asymmetric key in key Vault for database encryption key protection.

Single Sign-On

🟦 Synapse Foundation Components
🟨 Synapse Linked Services


1. Request web.azuresynapse.net
2. Redirect to Auth Server / Authentication Screen
5. Original URL request + Token
3. Credentials
4. Token/ Redirect to Original Server
Cookies?
Azure Active Directory
6. Access Validation/ Token Verification
7. web.azuresynapse.net
Azure Synapse Studio
MSI Auth
Orchestration
Linked Service
Token to access service
Azure Synapse Analytics
Spark
Azure Data Lake Storage Gen2
Power BI

Implicit authentication - User provides login credentials once to access Azure Synapse Workspace

AAD authentication - Azure Synapse Studio will request token to access each linked services as user. A separate token is acquired for each of the below services:

1. ADLS Gen2
2. Azure Synapse Analytics
3. Power BI
4. Spark – Spark Livy API
5. management.azure.com – resource provisioning
6. Develop artifacts – dev.workspace.net
7. Graph endpoints

MSI authentication - Orchestration uses MSI auth for automation

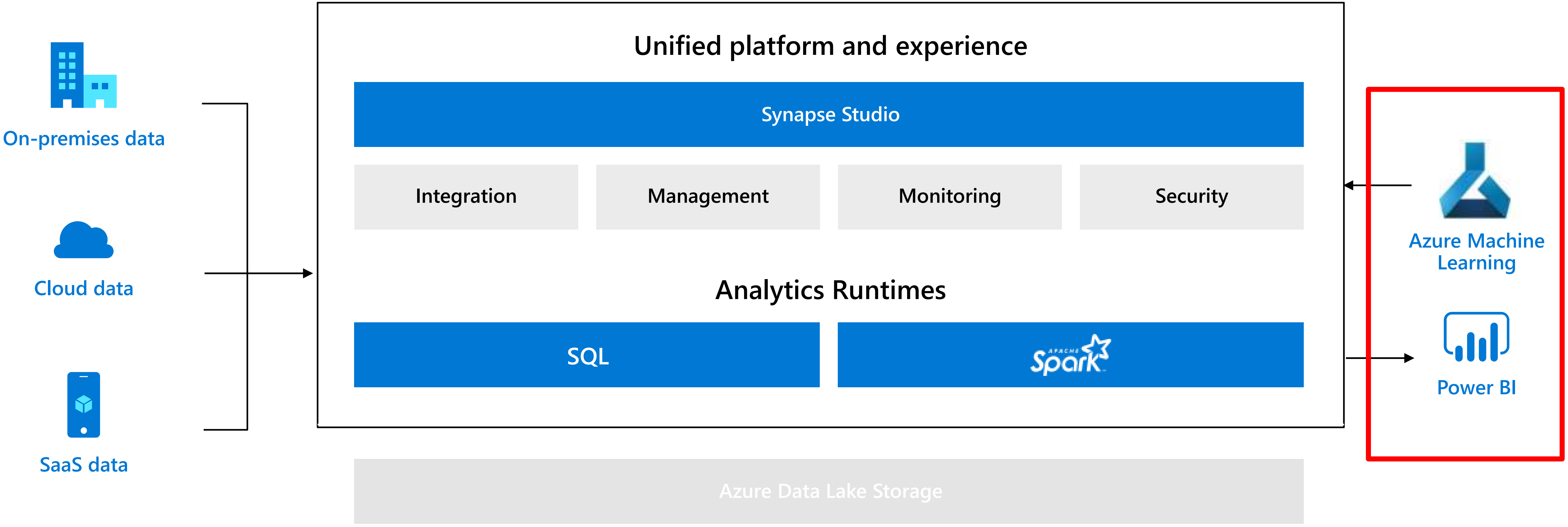
Microsoft

Azure Synapse Analytics

Connected Services

Azure Synapse Analytics

Limitless analytics service with unmatched time to insight



On-premises data

Cloud data

SaaS data

Unified platform and experience

Synapse Studio

Integration

Management

Monitoring

Security

Analytics Runtimes

SQL

Spark

Azure Data Lake Storage

Azure Machine Learning

Power BI

Azure Machine Learning

Overview

Data Scientists can use Azure ML notebooks to do (distributed) data preparation on Synapse Spark compute.

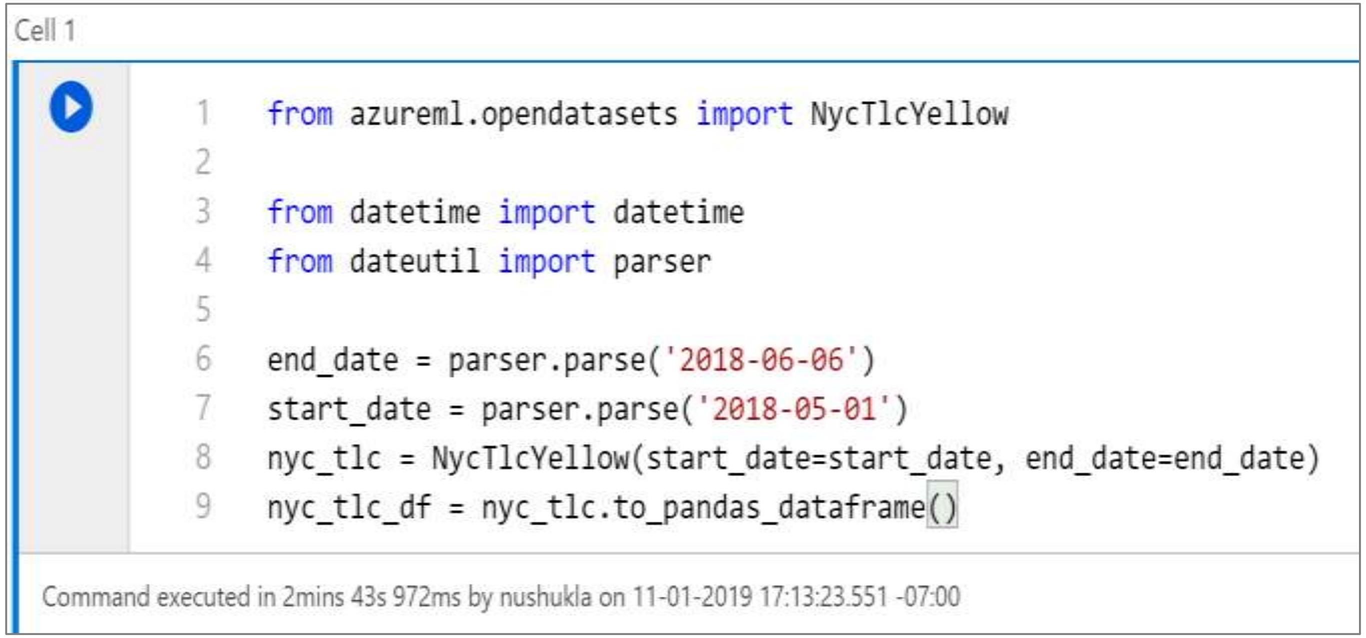
Benefits

Connect to your existing Azure ML workspace and project

Use the AutoML Classifier for classification or regression problem

Train the model

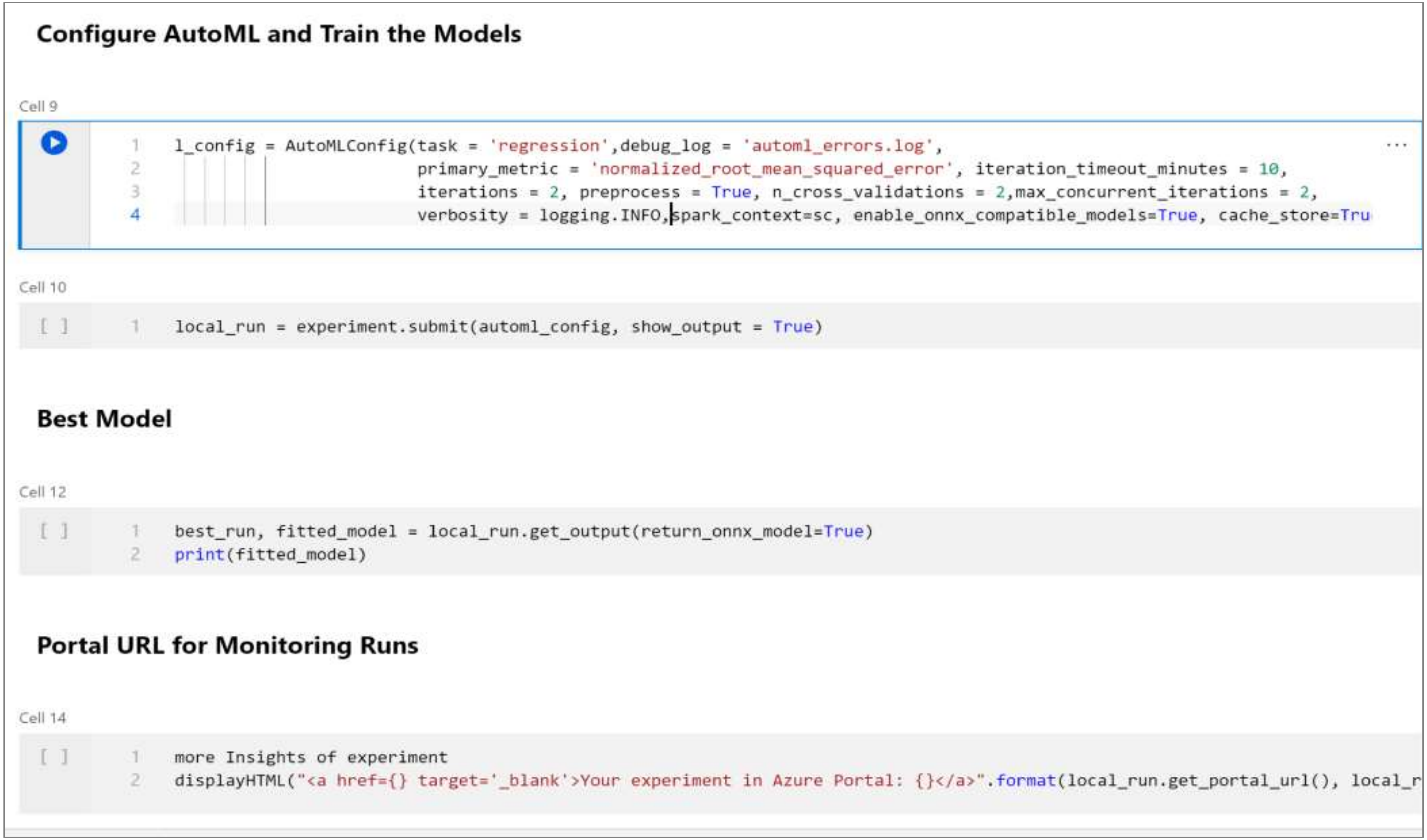
Access open datasets



```
from azureml.opendatasets import NycTlcYellow

from datetime import datetime
from dateutil import parser

end_date = parser.parse('2018-06-06')
start_date = parser.parse('2018-05-01')
nyc_tlc = NycTlcYellow(start_date=start_date, end_date=end_date)
nyc_tlc_df = nyc_tlc.to_pandas_dataframe()
```

```
Command executed in 2mins 43s 972ms by nushukla on 11-01-2019 17:13:23.551 -07:00
```

Azure Machine Learning (continued)



Configure AutoML and Train the Models

Cell 9

```
l_config = AutoMLConfig(task = 'regression',debug_log = 'automl_errors.log',
                        primary_metric = 'normalized_root_mean_squared_error', iteration_timeout_minutes = 10,
                        iterations = 2, preprocess = True, n_cross_validations = 2,max_concurrent_iterations = 2,
                        verbosity = logging.INFO,spark_context=sc, enable_onnx_compatible_models=True, cache_store=Tru
```

Cell 10

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

Best Model

Cell 12

```
best_run, fitted_model = local_run.get_output(return_onnx_model=True)
print(fitted_model)
```

Portal URL for Monitoring Runs

Cell 14

```
more Insights of experiment
displayHTML("<a href={} target='_blank'>Your experiment in Azure Portal: {}</a>".format(local_run.get_portal_url(), local_r
```

Power BI

Overview

Power BI is a business analytics service that delivers insights to enable fast, informed decisions

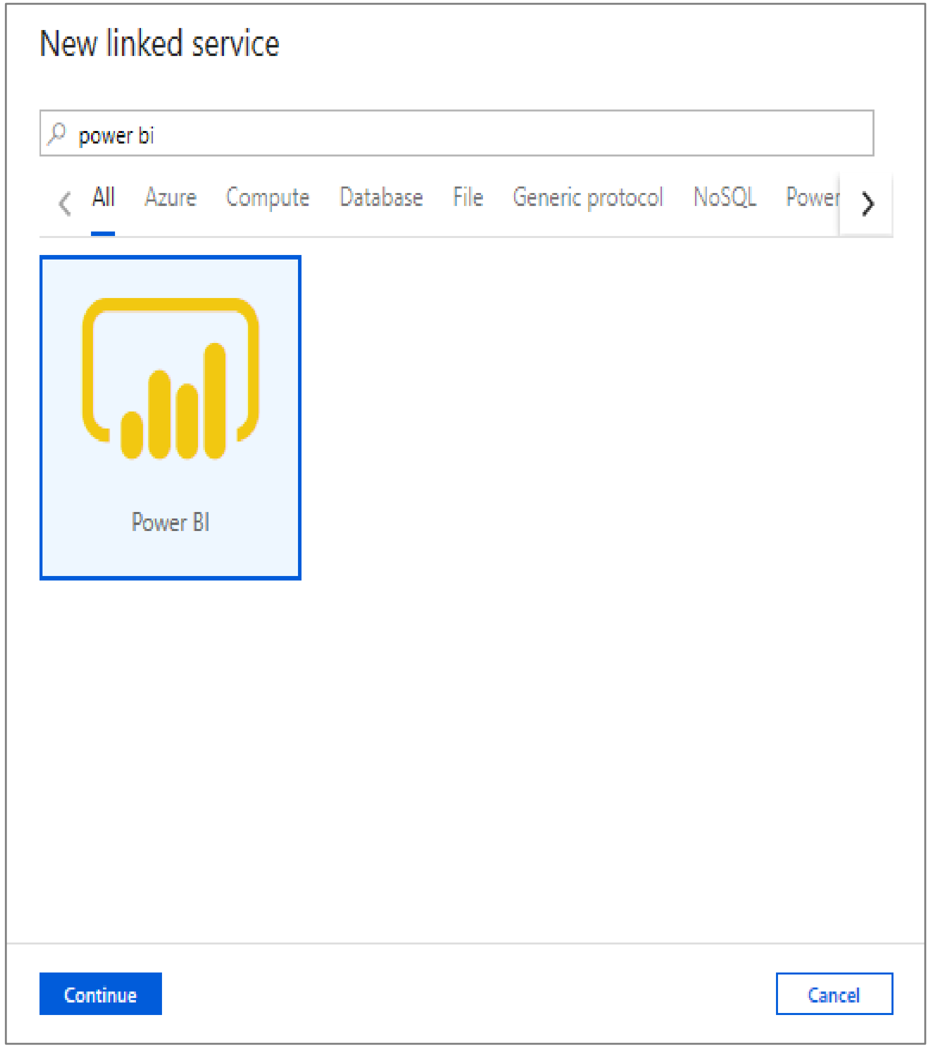
Benefits

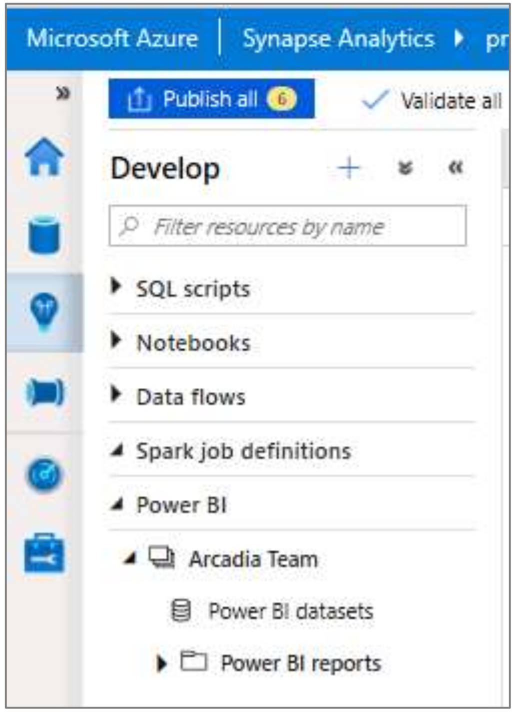
Create Power BI reports in the workspace

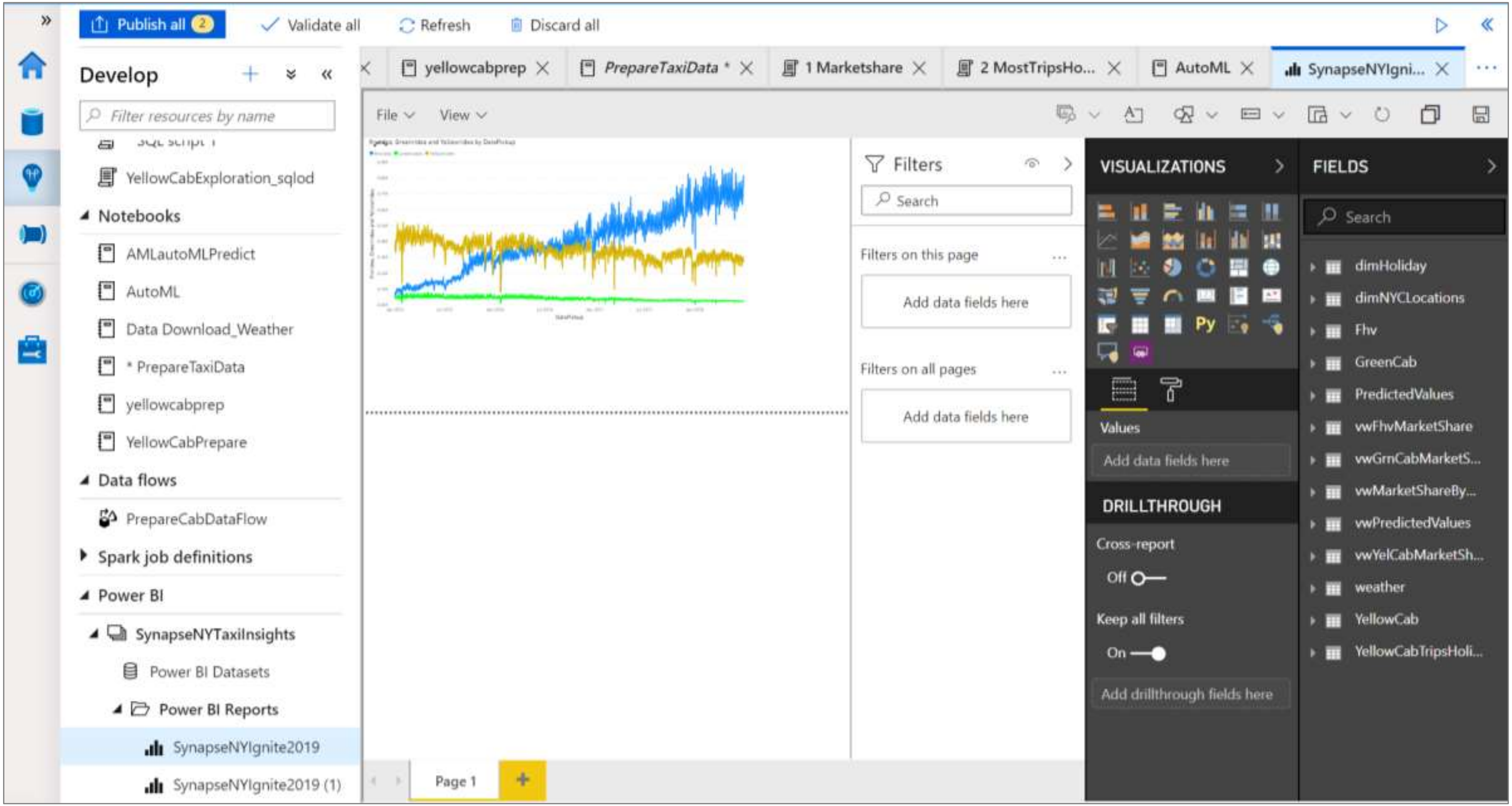
Have access to published reports in workspace

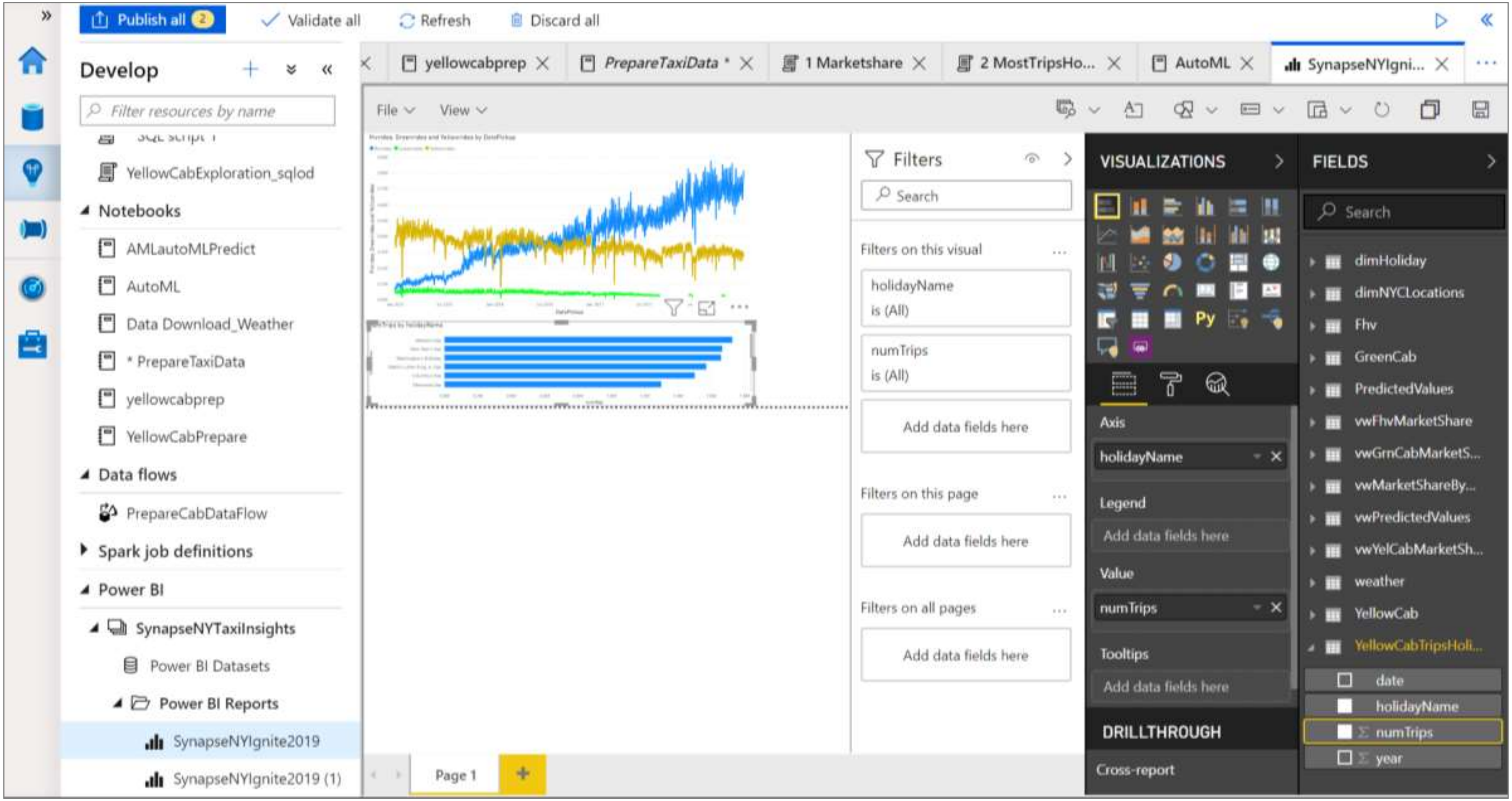
Update reports real time from Synapse workspace to get it reflected on Power BI service

Visually explore and analyze data






Publish all 2
Validate all
Refresh
Discard all
Develop
Filter resources by name
SQL script 1
YellowCabExploration_sqlod
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)
yellowcabprep
PrepareTaxiData *
1 Marketshare
2 MostTripsHo...
AutoML
SynapseNYIgni...
File
View
Filters
Search
Filters on this page
Add data fields here
Filters on all pages
Add data fields here
VISUALIZATIONS
Py
Values
Add data fields here
DRILLTHROUGH
Cross-report
Off
Keep all filters
On
Add drillthrough fields here
FIELDS
Search
dimHoliday
dimNYCLocations
Fhv
GreenCab
PredictedValues
vwFhvMarketShare
vwGrnCabMarketS...
vwMarketShareBy...
vwPredictedValues
vwYelCabMarketSh...
weather
YellowCab
YellowCabTripsHoli...
Page 1
+

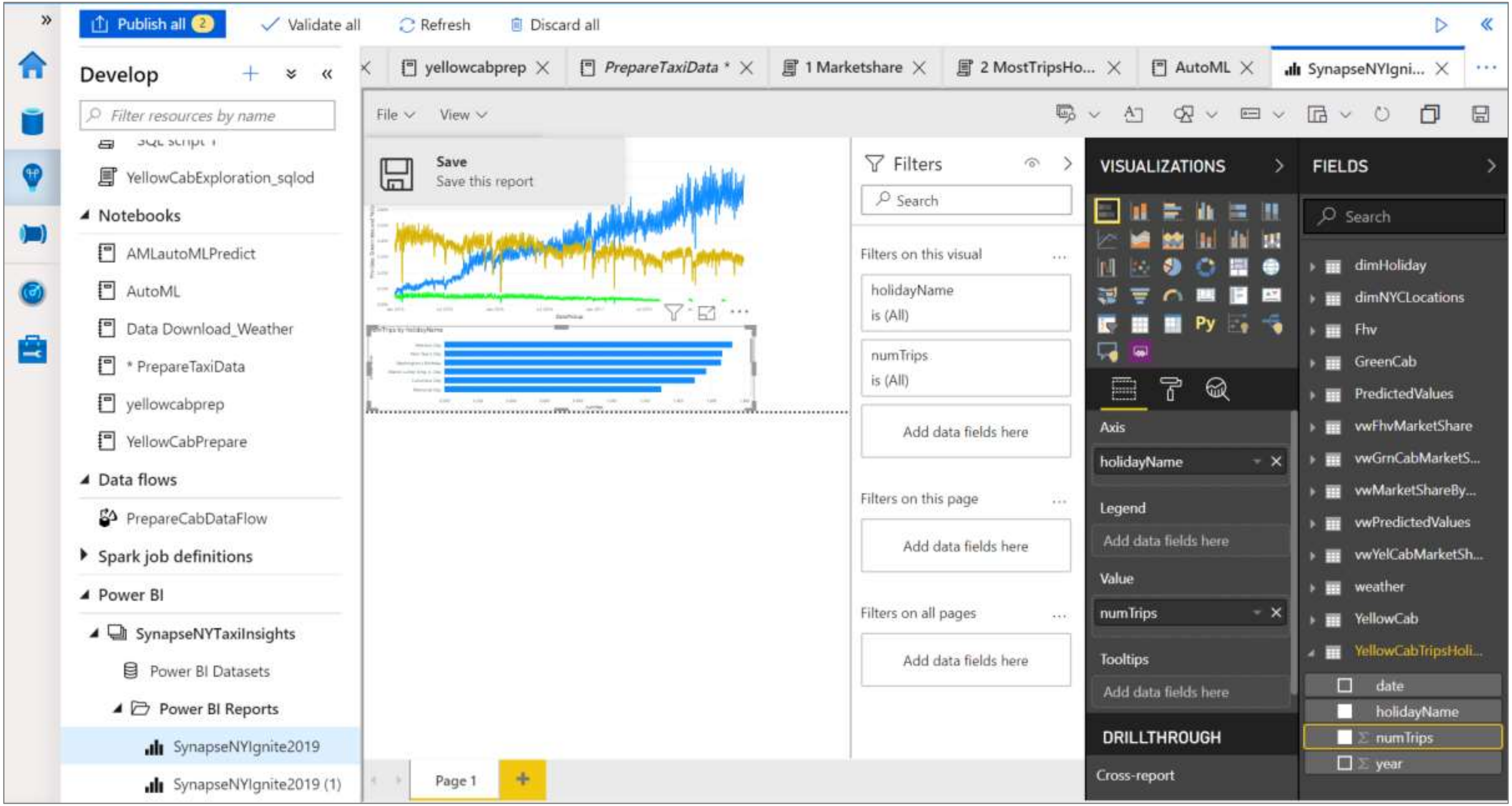


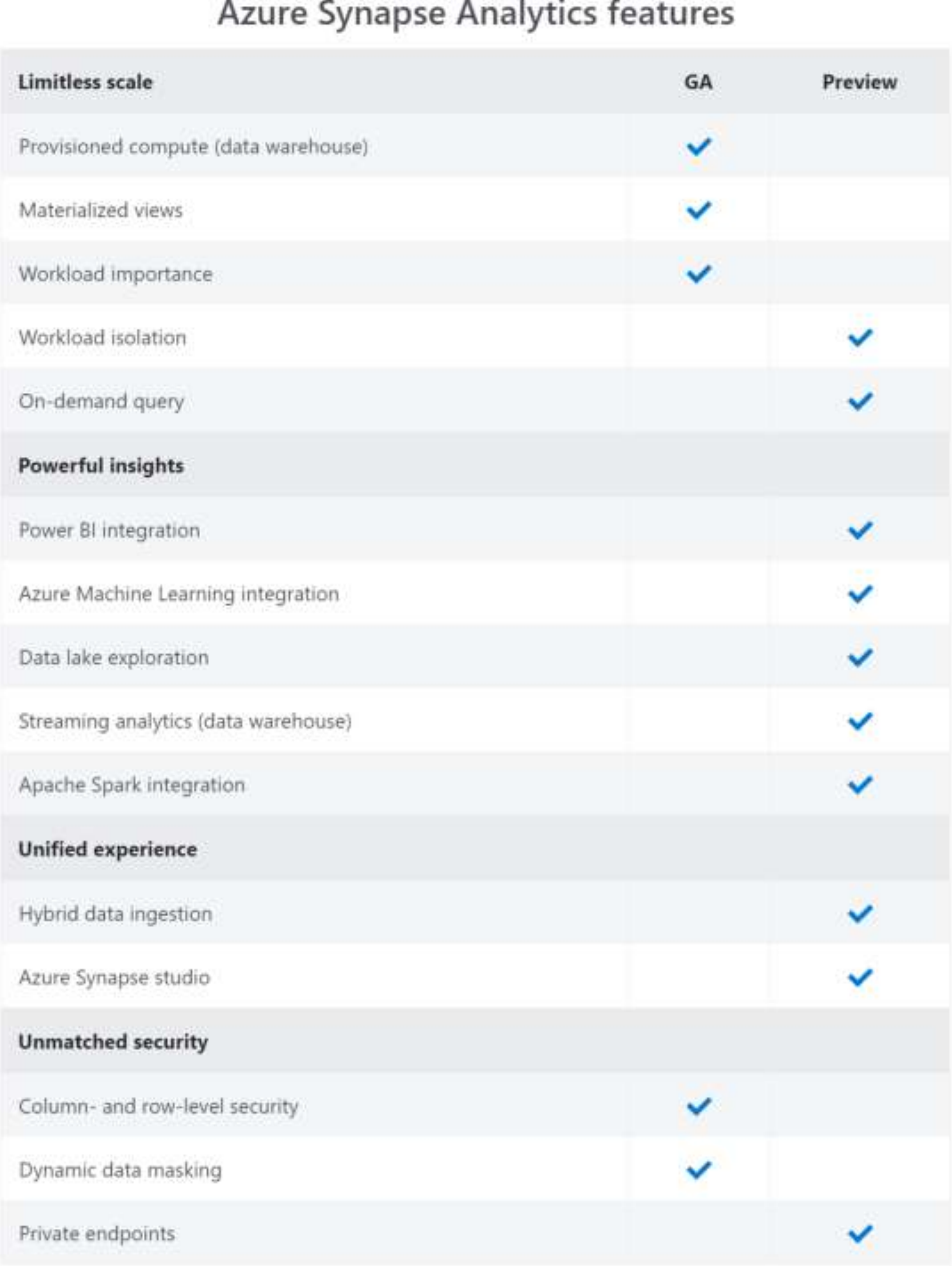
holidayName
is (All)

numTrips
is (All)

Axis
holidayName

Value
numTrips


Publish all 2
Validate all
Refresh
Discard all
Develop
Filter resources by name
SQL script 1
YellowCabExploration_sqlod
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)
yellowcabprep
PrepareTaxiData *
1 Marketshare
2 MostTripsHo...
AutoML
SynapseNYIgni...
File
View
Save
Save this report
Page 1
Filters
Search
Filters on this visual
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
VISUALIZATIONS
Py
Axis
holidayName
Legend
Add data fields here
Value
numTrips
Tooltips
Add data fields here
DRILLTHROUGH
Cross-report
FIELDS
Search
dimHoliday
dimNYCLocations
Fhv
GreenCab
PredictedValues
vwFhvMarketShare
vwGrnCabMarketS...
vwMarketShareBy...
vwPredictedValues
vwYelCabMarketSh...
weather
YellowCab
YellowCabTripsHoli...
date
holidayName
numTrips
year



Azure Synapse Analytics features

Limitless scale	**GA**	**Preview**
Provisioned compute (data warehouse)	✓	
Materialized views	✓	
Workload importance	✓	
Workload isolation		✓
On-demand query		✓
Powerful insights		
Power BI integration		✓
Azure Machine Learning integration		✓
Data lake exploration		✓
Streaming analytics (data warehouse)		✓
Apache Spark integration		✓
Unified experience		
Hybrid data ingestion		✓
Azure Synapse studio		✓
Unmatched security		
Column- and row-level security	✓	
Dynamic data masking	✓	
Private endpoints		✓

Migration Path

SQL DW – All of the data warehousing features that were generally available in Azure SQL Data Warehouse (intelligent workload management, dynamic data masking, materialized views, etc.) continue to be generally available today. Businesses can continue running their existing data warehouse workloads in production today with Azure Synapse and will automatically benefit from the new capabilities which are in preview (unified experience with Azure Synapse studio, query-as-a-service, built-in data integration, integrated Apache Spark, etc.) once they become generally available in 2020 and can use them in production if they choose to do so. Customers will not have to migrate any workloads

Azure Data Factory - Continue using Azure Data Factory. When the new functional of data integration within Azure Synapse becomes generally available, we will provide the capability to import your Azure Data Factory pipelines into Azure Synapse. Your existing Azure Data Factory accounts and pipelines will work with Azure Synapse if you choose not to import them into the Azure Synapse workspace. Note that Azure-SSIS Integration Runtime (IR) will not be supported in Synapse

Power BI – Customers link to a Power BI workspace within Azure Synapse Studio so no migration needed

ADLS Gen2 – Customers link to ADLS Gen2 within Azure Synapse Studio so no migration needed

Azure Databricks – TBD

Azure HDInsight - The Spark runtime within the Azure Synapse service is different from HDInsight

Q & A

James Serra, Big Data Evangelist
Email me at: JamesSerra3@gmail.com
Follow me at: @JamesSerra
Link to me at: www.linkedin.com/in/JamesSerra
Visit my blog at: JamesSerra.com (where this slide deck is posted under the “Presentations” tab)