Azure Data Factory is composed of four core components. These components work together to provide the platform on which you can compose data-driven workflows with steps to move and transform data.

Which component is best described by:

*"It is a specific action performed on the data in a workflow like the transformation or ingestion of the data. Each workflow can have one or more of these in it."*

- ○
  Activity
      **(Correct)**

- ○
  Pipeline

- ○
  Dataset

- ○
  Linked service

**Explanation**
An Azure subscription might have one or more Azure Data Factory instances. Azure Data Factory is composed of four core components. These components work together to provide the platform on which you can compose data-driven workflows with steps to move and transform data.



• **Pipeline:** It is created to perform a specific task by composing the different activities in the task in a single workflow. Activities in the pipeline can be data ingestion (Copy data to Azure) -> data processing (Perform Hive Query). Using pipeline as a single task user can schedule the task and manage all the activities in a single process also it is used to

run the multiple operation parallel. Multiple activities can be logically grouped together with an object referred to as a **Pipeline**, and these can be *scheduled* to execute, or a *trigger* can be defined that determines when a pipeline execution needs to be kicked off. There are different types of triggers for different types of events.

• **Activity:** It is a specific action performed on the data in a pipeline like the transformation or ingestion of the data. Each pipeline can have one or more activities in it. If the data is copied from one source to destination using Copy Monitor then it is a data movement activity. If data transformation is performed on the data using a hive query or spark job then it is a data transformation activity.

• **Datasets:** It is basically collected data users required which are used as input for the ETL process. Datasets have different formats; they can be in JSON, CSV, ORC, or text format.

• **Linked services:** It has information on the different data sources and the data factory uses this information to connect to data originating sources. It is mainly used to locate the data stores in the machines and also represent the compute services for the activity to be executed like running spark jobs on spark clusters or running hive queries using the hive services from the cloud.

https://www.educba.com/azure-data-factory/

What is a lambda architecture and what does it try to solve?

- ○

  An architecture that splits incoming data into two paths - a batch path and a streaming path. This architecture helps address the need to provide real-time processing in addition to slower batch computations.
  **(Correct)**

- ○

  An architecture that defines a data processing pipeline whereby microservices act as compute resources for efficient large-scale data processing.

- ○

  An architecture that employs the latest Scala runtimes in one or more Databricks clusters to provide the most efficient data processing platform available today.

- ○

  None of the listed options.

**Explanation**

The lambda architecture is a big data processing architecture that combines both batch- and real-time processing methods.

An example of a Delta Lake Architecture might be as shown in the diagram below.
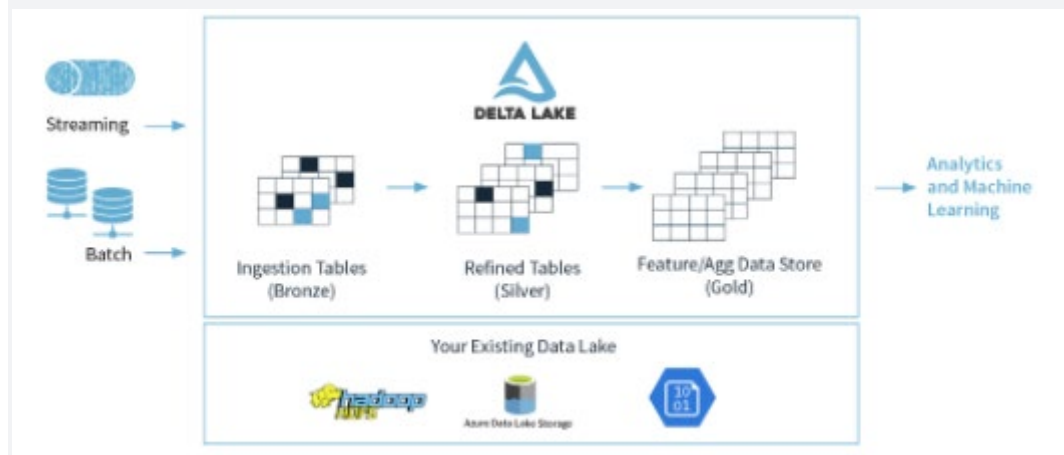


• Many **devices** generate data across different ingestion paths.

• Streaming data can be ingested from **IOT Hub** or **Event Hub**.

• Batch data can be ingested by **Azure Data Factory** or **Azure Databricks**.

• Extracted, Transformed data is loaded into a **Delta Lake**.

**Lambda architecture**

When working with large data sets, it can take a long time to run the sort of queries that clients need. These queries can't be performed in real time, and often require algorithms such as MapReduce that operate in parallel across the entire data set. The results are then stored separately from the raw data and used for querying.

One drawback to this approach is that it introduces latency. If processing takes a few hours, a query may return results that are several hours old. Ideally, you would like to get some results in real time (perhaps with some loss of accuracy), and combine these results with the results from the batch analytics.

The **lambda architecture** is a big data processing architecture that addresses this problem by combining both batch- and real-time processing methods. It features an append-only immutable data source that serves as system of record. Timestamped events are appended to existing events (nothing is overwritten). Data is implicitly ordered by time of arrival.

Notice how there are really two pipelines here, one batch and one streaming, hence the name *lambda* architecture.

It is difficult to combine processing of batch and real-time data as is evidenced by the diagram below:
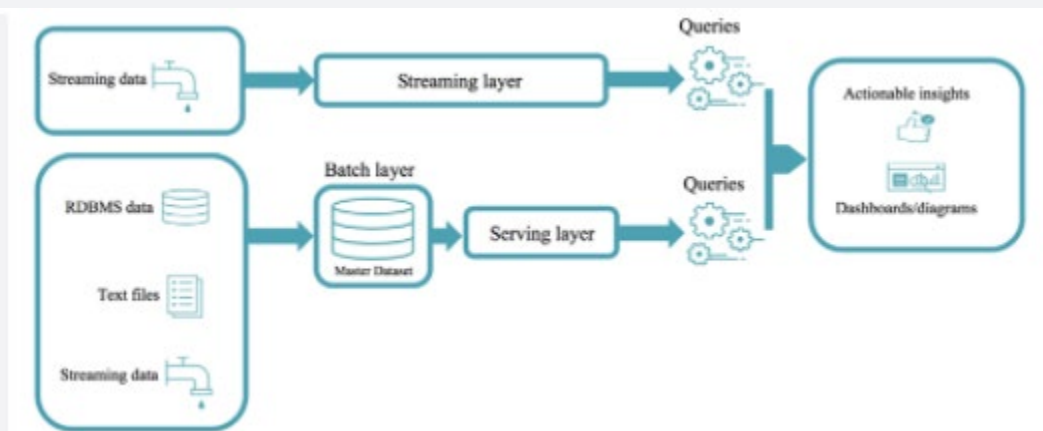


**Delta Lake architecture**

The Delta Lake Architecture is a vast improvement upon the traditional Lambda architecture. At each stage, we enrich our data through a unified pipeline that allows us to combine batch and streaming workflows through a shared filestore with ACID-compliant transactions.

**Bronze** tables contain raw data ingested from various sources (JSON files, RDBMS data, IoT data, etc.).

**Silver** tables will provide a more refined view of our data. We can join fields from various bronze tables to enrich streaming records, or update account statuses based on recent activity.

**Gold** tables provide business level aggregates often used for reporting and dashboarding. This would include aggregations such as daily active website users, weekly sales per store, or gross revenue per quarter by department.

The end outputs are actionable insights, dashboards, and reports of business metrics.
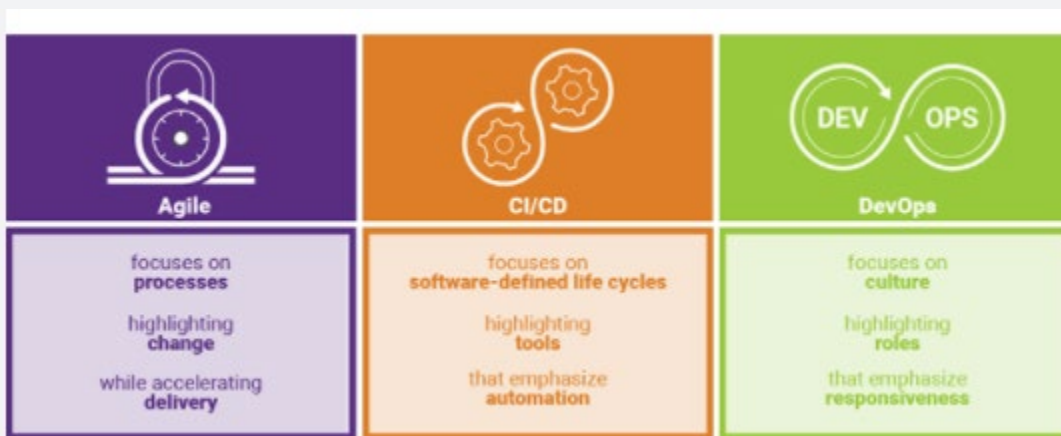


By considering our business logic at all steps of the extract-transform-load (ETL) pipeline, we can ensure that storage and compute costs are optimized by reducing unnecessary duplication of data and limiting ad hoc querying against full historic data.

Each stage can be configured as a batch or streaming job, and ACID transactions ensure that we succeed or fail completely.

https://www.jamesserra.com/archive/2019/10/databricks-delta-lake/

Which data processing framework will a data engineer use to ingest data onto cloud data platforms in Azure?

- ○

  Atomicity, Consistency, Isolation, and Durability (ACID)

- ○

  Online transaction processing (OLTP)

- ○

  Automated Data Processing Equipment (ADPE)

- ○

  Extract, transform, and load (ETL)

- ○

  Extract, load, and transform (ELT)
  **(Correct)**

**Explanation**

ELT is a typical process for ingesting data from an on-premises database into the cloud.

Traditional SMP dedicated SQL pools use an Extract, Transform, and Load (ETL) process for loading data. Synapse SQL, within Azure Synapse Analytics, uses distributed query processing architecture that takes advantage of the scalability and flexibility of compute and storage resources.

Using an Extract, Load, and Transform (ELT) process leverages built-in distributed query processing capabilities and eliminates the resources needed for data transformation prior to loading.

While dedicated SQL pools support many loading methods, including popular SQL Server options such as bcp and the SqlBulkCopy API, the fastest and most scalable way to load data is through PolyBase external tables and the COPY statement.

With PolyBase and the COPY statement, you can access external data stored in Azure Blob storage or Azure Data Lake Store via the T-SQL language. For the most flexibility when loading, we recommend using the COPY statement.

**What is ELT?**

Extract, Load, and Transform (ELT) is a process by which data is extracted from a source system, loaded into a dedicated SQL pool, and then transformed.
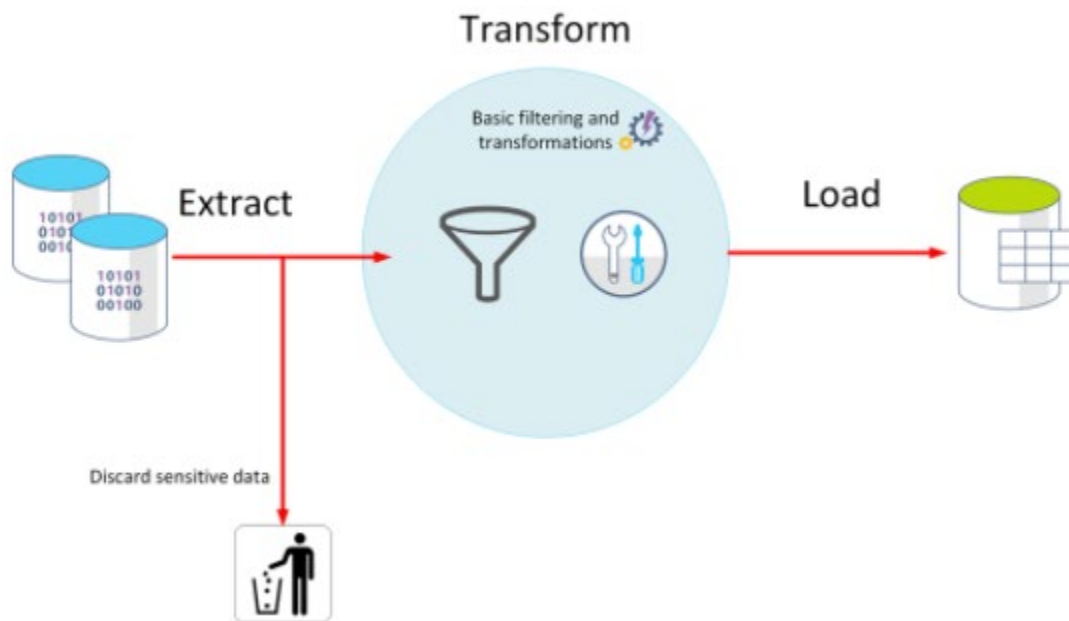
The basic steps for implementing ELT are:

1. Extract the source data into text files.

2. Land the data into Azure Blob storage or Azure Data Lake Store.

3. Prepare the data for loading.

4. Load the data into staging tables with PolyBase or the COPY command.

5. Transform the data.

6. Insert the data into production tables.

https://docs.microsoft.com/en-us/azure/synapse-analytics/sql-data-warehouse/design-elt-data-loading

Identify the missing word(s) in the following sentence within the context of Microsoft Azure.

Azure Synapse Analytics is an integrated analytics platform, which combines data warehousing, big data analytics, data integration, and visualization into a single environment. Azure Synapse Analytics empowers users of all abilities to gain access and quick insights across all of their data, enabling a whole new level of performance and scale.

Azure Synapse Analytics enables you to answer the question ... [?] (Select all that apply)

- ☐
  *"What is likely to happen in the future based on previous trends and patterns?"*
  **(Correct)**

- ☐
  *"What is happening in my business?"*
  **(Correct)**

- ☐
  *"Why is it happening?"*
  **(Correct)**

- ☐
  *"When will the modification made meet my goals?"*

**Explanation**
**Azure Synapse Analytics enables you to answer the questions:**

• *"Why is it happening?"*

• *"What is happening in my business?"*

• *"What is likely to happen in the future based on previous trends and patterns?"*

**"When will the modification made meet my goals?" is future telling, not a capability of Azure Synapse Analytics.**

Azure Synapse Analytics is an integrated analytics platform, which combines data warehousing, big data analytics, data integration, and visualization into a single environment. Azure Synapse Analytics empowers users of all abilities to gain access and quick insights across all of their data, enabling a whole new level of performance and scale.

Gartner defines a range of analytical types that Azure Synapse Analytics can support including:

**Descriptive analytics**

Descriptive analytics answers the question "What is happening in my business?" The data to answer this question is typically answered through the creation of a data warehouse. Azure Synapse Analytics leverages the dedicated SQL Pool capability that enables you to create a persisted data warehouse to perform this type of analysis. You can also make use of SQL Serverless to prepare data from files to create a data warehouse interactively to answer the question too.

**Diagnostic analytics**

Diagnostic analytics deals with answering the question "Why is it happening?" this may involve exploring information that already exists in a data warehouse, but typically involves a wider search of your data estate to find more data to support this type of analysis.

You can use the same SQL serverless capability within Azure Synapse Analytics that enables you to interactively explore data within a data lake. This can quickly enable a user to search for additional data that may help them to understand "Why is it happening?"

https://www.valamis.com/hub/descriptive-analytics
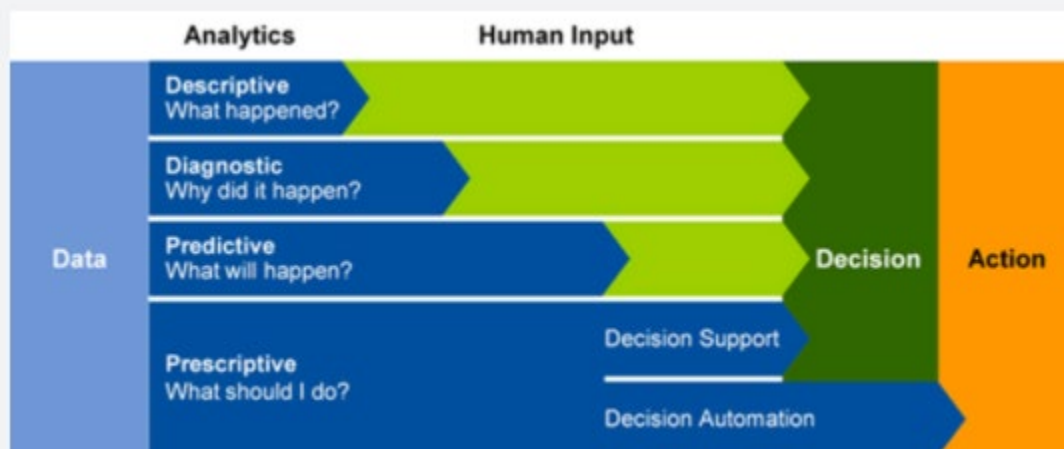
**Predictive analytics**

Azure Synapse Analytics also enables you to answer the question **"What is likely to happen in the future based on previous trends and patterns?"** by using its integrated Apache Spark engine. This can also be used in conjunction with other services such as Azure Machine Learning Services, or Azure Databricks.

https://www.ibm.com/analytics/predictive-analytics

**Prescriptive analytics**

This type of analytics looks at executing actions based on real-time or near real-time analysis of data, using predictive analytics. Azure Synapse Analytics provides this capability through both Apache Spark, Azure Synapse Link, and by integrating streaming technologies such as Azure Stream Analytics.

https://www.talend.com/resources/what-is-prescriptive-analytics/

Azure Synapse Analytics gives the users of the service the freedom to query data on their own terms, using either serverless or dedicated resources at scale. Azure Synapse Analytics brings these two worlds together with a unified data integration experience to ingest, prepare, manage, and serve data using Azure Synapse Pipelines. In addition, you can visualize the data in the form of dashboards and reports for immediate analysis using Power BI which is integrated into the service too.

https://docs.microsoft.com/en-us/azure/synapse-analytics/overview-what-is

**Scenario:** Stark Industries was founded by Howard Stark during the early twentieth century, a great pioneer in different types of technology and constantly helping the United States Armed Forces with different and innovative weapons. Some years after the death of Howard Stark, his son Tony took over the company.

Currently in the process of making many IT improvements, you have been hired as a consultant to advise on several Microsoft Azure projects.

Stark has an Azure Data Lake Storage Gen2 account that contains a JSON file for customers. The file contains two attributes named FirstName and LastName.

The current project requires the team to copy the data from the JSON file to an Azure Synapse Analytics table by using Azure Databricks. A new column must be created that concatenates the FirstName and LastName values.

The team has created the following components:

• A destination table in Azure Synapse
• An Azure Blob storage container
• A service principal

The team has also created a list of actions they are considering to take in their Databricks notebook:

a. Mount the Data Lake Storage onto DBFS.

b. Write the results to a table in Azure Synapse.

c. Perform transformations on the file.

d. Specify a temporary folder to stage the data.

e. Write the results to Data Lake Storage.

f. Read the file into a data frame.

g. Drop the data frame.

h. Perform transformations on the data frame.

As the Azure expert, Tony looks to you to guide the team on which actions to take, and the order to take them.

Which actions should you recommend the team perform in sequence in the Databricks notebook?

- ○
  d → a → c → f → g

- ○
  h → c → d → g → b

- ○
  c → a → f → h

- ○
  f → c → d → b → g
  **(Correct)**

**Explanation**
*The correct actions in sequence are: f → c → d → b → g*

**Step 1:** Read the file into a data frame.
You can load the json files as a data frame in Azure Databricks.
**Step 2:** Perform transformations on the data frame.
**Step 3:** Specify a temporary folder to stage the data
Specify a temporary folder to use while moving data between Azure Databricks and Azure Synapse.
**Step 4:** Write the results to a table in Azure Synapse.
You upload the transformed data frame into Azure Synapse. You use the Azure Synapse connector for Azure Databricks to directly upload a dataframe as a table in a Azure Synapse.
**Step 5:** Drop the data frame.
Clean up resources. You can terminate the cluster. From the Azure Databricks workspace, select Clusters on the left. For the cluster to terminate, under Actions, point to the ellipsis (...) and select the Terminate icon.
https://docs.microsoft.com/en-us/azure/azure-databricks/databricks-extract-load-sql-data-warehouse

**Scenario:** You are working at OZcorp which is a multi-million dollar company run by Mayor Norman Osborn. Profits from the company are used to fund Norman's operatives, such as a police task force. You have been hired by OZcorp as a Microsoft Azure Expert.

At the moment, the team is designing the folder structure for an Azure Data Lake Storage Gen2 container where OZcorp users will query data by using a variety of services including Azure Databricks and Azure Synapse Analytics serverless SQL pools. The data will be secured by subject area. Most queries will include data from the current year or current month.

As the Azure expert, which folder structure should you recommend to support fast queries and simplified folder security?

- ○
  `/{SubjectArea}/{DataSource}/{YYYY}/{MM}/{DD}/{FileData}_{YYYY}_{MM}_{DD}.csv`
  **(Correct)**

- ○
  `/{YYYY}/{MM}/{DD}/{SubjectArea}/{DataSource}/{FileData}_{YYYY}_{MM}_{DD}.csv`

- ○
  `/{DD}/{MM}/{YYYY}/{SubjectArea}/{DataSource}/{FileData}_{YYYY}_{MM}_{DD}.csv`

- ○
  `/{SubjectArea}/{DataSource}/{DD}/{MM}/{YYYY}/{FileData}_{YYYY}_{MM}_{DD}.csv`

**Explanation**
*You should recommend the structure* `/{SubjectArea}/{DataSource}/{YYYY}/{MM}/{DD}/{FileData}_{YYYY}_{MM}_{DD}.csv` .

There's an important reason to put the date at the end of the directory structure.

If you want to lock down certain regions or subject matters to users/groups, then you can easily do so with the POSIX permissions. Otherwise, if there was a need to restrict a certain security group to viewing just the UK data or certain planes, with the date structure in front a separate permission would be required for numerous directories under every hour directory. Additionally, having the date structure in front would exponentially increase    the    number    of    directories    as    time    went    on.
*Note: In IoT workloads, there can be a great deal of data being landed in the data store that spans across numerous products, devices, organizations, and customers. It is important to pre-plan the directory layout for organization, security, and efficient processing of the data for down-stream consumers.*

A general template to consider might be the following layout:
`{Region}/{SubjectMatter(s)}/{yyyy}/{mm}/{dd}/{hh}/`

**Batch jobs structure**

From a high-level, a commonly used approach in batch processing is to land data in an "in" directory. Then, once the data is processed, put the new data into an "out" directory for downstream processes to consume. This directory structure is seen sometimes for jobs that require processing on individual files and might not require massively parallel processing over large datasets. Like the IoT structure recommended above, a good directory structure has the parent-level directories for things such as region and subject matters (for example, organization, product/producer). This structure helps with securing the data across your organization and better management of the data in your workloads. Furthermore, consider date and time in the structure to allow better organization, filtered searches, security, and automation in the processing. The level of granularity for the date structure is determined by the interval on which the data is uploaded or processed, such as hourly, daily, or even monthly.

Sometimes file processing is unsuccessful due to data corruption or unexpected formats. In such cases, directory structure might benefit from a `/bad` folder to move the files to for further inspection. The batch job might also handle the reporting or notification of these *bad* files for manual intervention. Consider the following template structure:

`{Region}/{SubjectMatter(s)}/In/{yyyy}/{mm}/{dd}/{hh}/`

`{Region}/{SubjectMatter(s)}/Out/{yyyy}/{mm}/{dd}/{hh}/`

`{Region}/{SubjectMatter(s)}/Bad/{yyyy}/{mm}/{dd}/{hh}/`

For example, a marketing firm receives daily data extracts of customer updates from their clients in North America. It might look like the following snippet before and after being processed:

`NA/Extracts/ACMEPaperCo/In/2017/08/14/updates_08142017.csv`

`NA/Extracts/ACMEPaperCo/Out/2017/08/14/processed_updates_08142017.csv`

In the common case of batch data being processed directly into databases such as Hive or traditional SQL databases, there isn't a need for an **/in** or **/out** folder since the output already goes into a separate folder for the Hive table or external database. For example, daily extracts from customers would land into their respective folders, and orchestration by something like Azure Data Factory, Apache Oozie, or Apache Airflow would trigger a daily Hive or Spark job to process and write the data into a Hive table.

https://docs.microsoft.com/en-us/azure/storage/blobs/data-lake-storage-best-practices#batch-jobs-structure

Azure Synapse Analytics allows you to create, control, and manage resource availability when workloads are competing. This allows you to manage the relative importance of each workload when waiting for available resources.

To facilitate faster load times, you can create a workload classifier for the load user with … [?]

- the "importance" set to above_normal or High.
  **(Correct)**

- the "priority" set to 1.

- the "urgency" set to 10.

- the "rank" set to 10.

**Explanation**
Azure Synapse Analytics allows you to create, control, and manage resource availability when workloads are competing. This allows you to manage the relative importance of each workload when waiting for available resources.

**To facilitate faster load times, you can create a workload classifier for the load user with the "importance" set to above_normal or High.** Workload importance ensures that the load takes precedence over other waiting tasks of a lower importance rating. Use this in conjunction with your own workload group definitions for workload isolation to manage minimum and maximum resource allocations during peak and quiet periods.

Dedicated SQL pool workload management in Azure Synapse consists of three high-level concepts:

• Workload Classification

• Workload Importance

• Workload Isolation

These capabilities give you more control over how your workload utilizes system resources.

**Workload classification**

Workload management classification allows workload policies to be applied to requests through assigning resource classes and importance.

While there are many ways to classify data warehousing workloads, the simplest and most common classification is load and query. You load data with insert, update, and delete statements. You query the data using selects. A data warehousing solution will often have a workload policy for load activity, such as assigning a higher resource class with more resources. A different workload policy could apply to queries, such as lower importance compared to load activities.

You can also subclassify your load and query workloads. Subclassification gives you more control of your workloads. For example, query workloads can consist of cube refreshes, dashboard queries or ad-hoc queries. You can classify each of these query workloads with different resource classes or importance settings. Load can also benefit from subclassification. Large transformations can be assigned to larger resource classes. Higher importance can be used to ensure key sales data is loader before weather data or a social data feed.

Not all statements are classified as they do not require resources or need importance to influence execution. DBCC commands, `BEGIN`, `COMMIT`, and `ROLLBACK` `TRANSACTION` statements are not classified.

**Workload importance**

Workload importance influences the order in which a request gets access to resources. On a busy system, a request with higher importance has first access to resources. Importance can also ensure ordered access to locks. There are five levels of importance: low, below_normal, normal, above_normal, and high. Requests that don't set importance are assigned the default level of normal. Requests that have the same importance level have the same scheduling behaviour that exists today.

**Workload isolation**

Workload isolation reserves resources for a workload group. Resources reserved in a workload group are held exclusively for that workload group to ensure execution. Workload groups also allow you to define the amount of resources that are assigned per request, much like resource classes do. Workload groups give you the ability to reserve or cap the amount of resources a set of requests can consume. Finally, workload groups are a mechanism to apply rules, such as query timeout, to requests.

https://docs.microsoft.com/en-us/azure/synapse-analytics/sql-data-warehouse/sql-data-warehouse-workload-management

What is the Databricks Delta command to display metadata?

- ○
  `SHOW SCHEMA tablename`

- ○
  `MSCK DETAIL tablename`

- ○
  `METADATA SHOW tablename`

- ○
  `DESCRIBE DETAIL tableName`
  **(Correct)**

**Explanation**

You display metadata by using `DESCRIBE DETAIL tableName`.

https://docs.microsoft.com/en-us/azure/databricks/spark/2.x/spark-sql/language-manual/describe-table

Monitoring is a key part of any mission-critical workload. It helps to proactively detect and prevent issues that might otherwise cause application or service downtime.

You can monitor Azure Stream Analytics jobs by using which of the following? (Select all that apply)

- ☐
  Alerts on issues in applications or services
    **(Correct)**

- ☐
  Real-time dashboards that show service and application health trends
    **(Correct)**

- ☐
  An activity log for each running job
    **(Correct)**

- ☐
  Diagnostic logs
    **(Correct)**

- ☐
  Predictive dashboards that show expected service and application health status

**Explanation**
Monitoring is a key part of any mission-critical workload. It helps to proactively detect and prevent issues that might otherwise cause application or service downtime.
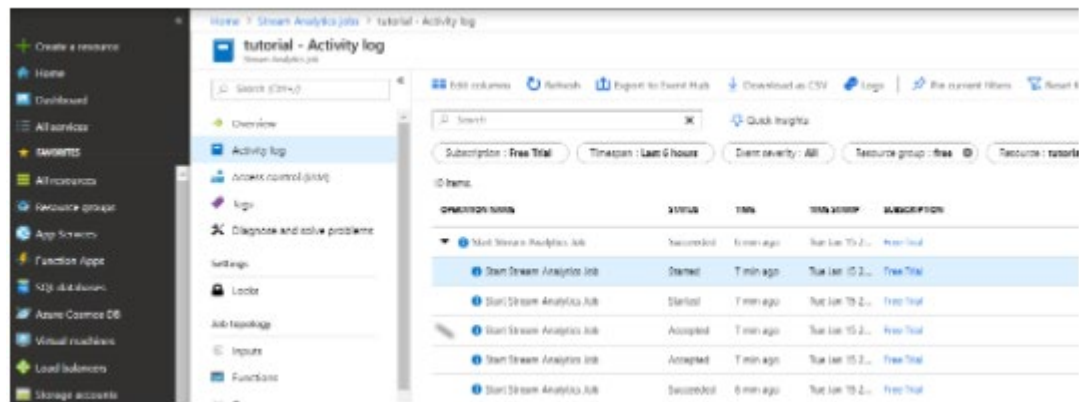
You can monitor Azure Stream Analytics jobs by using several tools:

• An activity log for each running job

• Real-time dashboards that show service and application health trends

• Alerts on issues in applications or services

• Diagnostic logs

**Activity logs**

The Stream Analytics activity log provides details about each job you run. This low-level troubleshooting tool can help you identify issues with data sources, outputs, or transformation queries.

Each job you create has an activity log. In the log, expand each job, and then select an event to see details in the JSON file.



## Dashboards

Dashboards show key health metrics for your Stream Analytics jobs. To view a live dashboard, go to the Azure portal, select your Stream Analytics job, and under **Monitoring**, select **Metrics**.
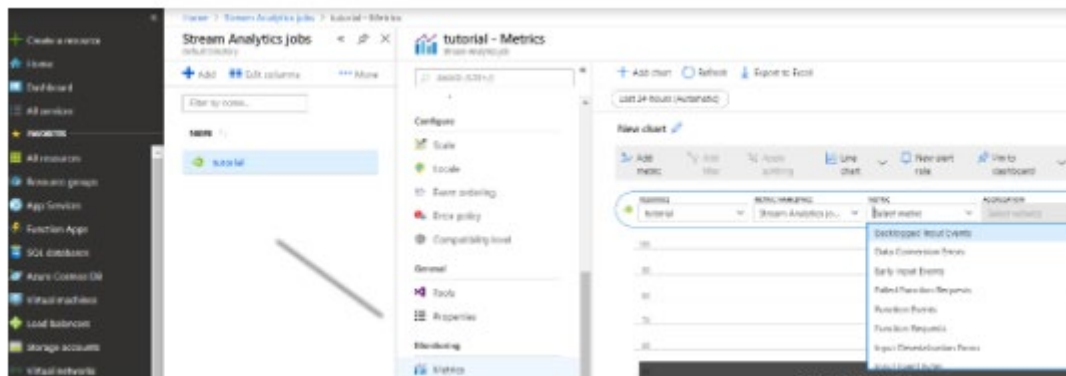


## Alerts

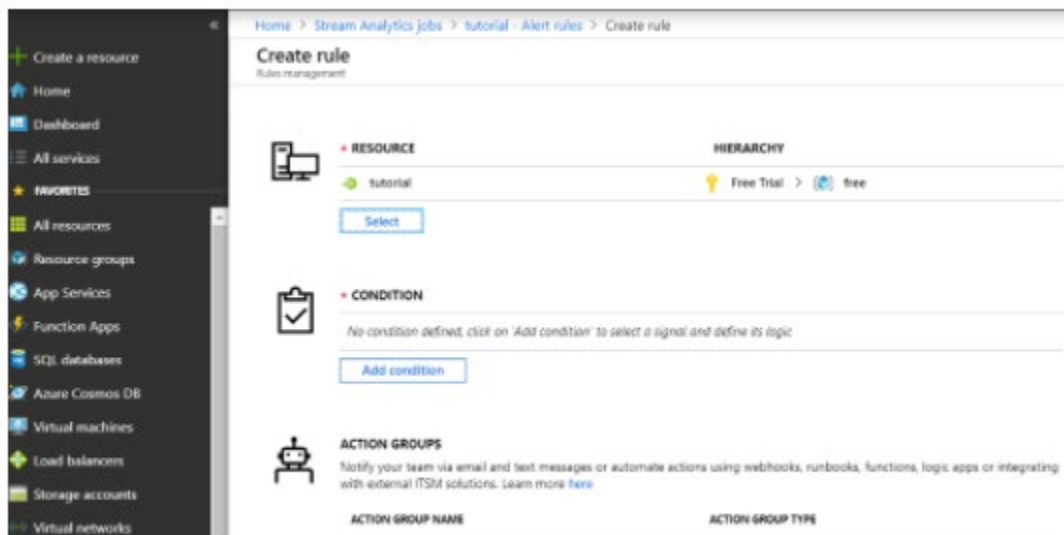To proactively detect issues, you can set up Stream Analytics to fire alerts based on various metrics and thresholds. To set up alerts in the Azure portal, in your Stream Analytics job, under **Monitoring**, select **Alert rules** > **New alert rule**.
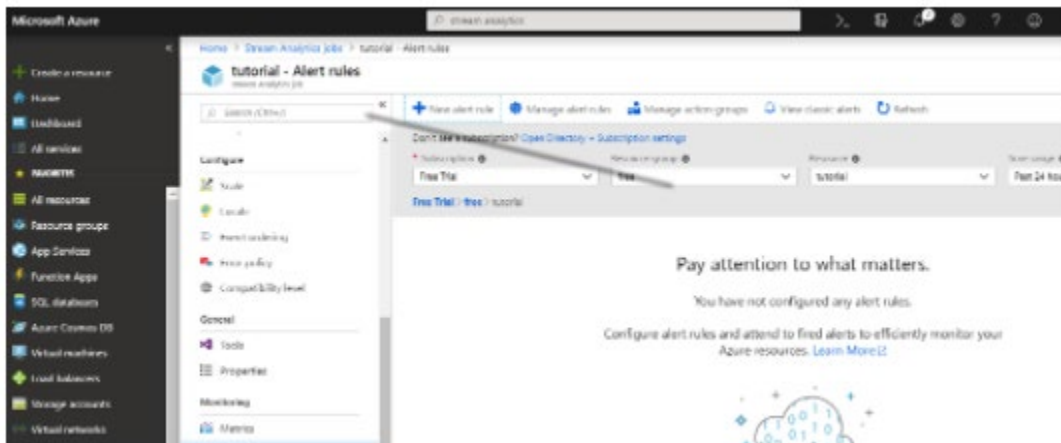
As you're setting up your rules, you can choose to send alerts by email, SMS, or voicemail. You can also use alerts to trigger workflows.



**Diagnostic logs**

Diagnostic logging is a key part of operational infrastructure. Use diagnostic logs to help find root-cause issues in production deployments. You can conveniently deliver diagnostic logs to various sinks or destinations for root-cause analysis.

Stream Analytics diagnostics is turned off by default. In the Azure portal, you can turn it on when you need it. In the Stream Analytics job, under **Monitoring**, select **Diagnostic logs**.

You can persist diagnostics settings in an Azure Storage account or send them to Azure Event Hubs or Azure Log Analytics. Generate diagnostics logs for job execution or job authoring.



Modern data warehouse

https://docs.microsoft.com/en-us/azure/stream-analytics/stream-analytics-monitor-jobs

Question 10: Skipped

**Scenario**: To access data in your company storage account, your client makes requests over HTTP or HTTPS. Every request to a secure resource must be authorized.

Which service ensures that the client has the permissions required to access the data.

- ○ Azure AD

- ○ RBAC
  **(Correct)**

- ○ Private Link

- ○ Vault

**Explanation**
**Role-based access control**

You can choose from several access options. Arguably, the most flexible option is role-based access.

Azure Storage supports Azure Active Directory and role-based access control (RBAC) for both resource management and data operations. To security principals, you can assign RBAC roles that are scoped to the storage account. Use Active Directory to authorize resource management operations, such as configuration. Active Directory is supported for data operations on Blob and Queue storage.

To a security principal or a managed identity for Azure resources, you can assign RBAC roles that are scoped to a subscription, a resource group, a storage account, or an individual container or queue.

https://docs.microsoft.com/en-us/azure/role-based-access-control/overview

Knowing now the different concepts of spark it is imperative to understand how it fits in with the different Data services on Azure.

Which of the following is best described by:

*"An open-source memory optimized system for managing big data workloads, which is used when you want a spark engine for big data processing or data science where you don't mind that there is no SLA provided. Usually it is of interest to Open Source Professionals and the reason for this product is to overcome the limitations known as SMP systems for big data workloads."*

- ○ Spark Pools in Azure Synapse Analytics

- ○ Apache Spark
  **(Correct)**

- ○ Azure Databricks

- ○ HDI

**Explanation**
There are two concepts within Apache Spark Pools in Azure Synapse Analytics, namely Spark pools and Spark Instances. In short, they do the following:

**Spark Pools:**

• Exists as Metadata

• Creates a Spark Instance

• No costs associated with creating Pool

• Permissions can be applied

• Best practices

Spark Instances:

• Created when connected to Spark Pool, Session, or Job

• Multiple users can have access

• Reusable

Knowing now the different concepts of spark it is imperative to understand how it fits in with the different Data services on Azure. Below is a table where "the when to use what" is outlined:

| | Apache Spark | HDInsight | Azure Databricks | Synapse Spark |
|---|---|---|---|---|
| What | Is an Open Source memory optimized system for managing big data workloads | Microsoft implementation of Open Source Spark managed within the realms of Azure | AA managed Spark as a Service solution | Embedded Spark capability within Azure Synapse Analytics |
| When | When you want to benefits of spark for big data processing and/or data science work without the Service Level Agreements of a provider | When you want to benefits of OSS spark with the Service Level Agreement of a provide | Provides end to end data engineering and data science solution and management platform | Enables organizations without existing Spark implementations to fire up a Spark cluster to meet data engineering needs without the overheads of the other Spark platforms listed |

| | Apache Spark | HDInsight | Azure Databricks | Synapse Spark |
|---|---|---|---|---|
| Who | Open Source Professionals | Open Source Professionals wanting SLA's and Microsoft Data Platform experts | Data Engineers and Data Scientists working on big data projects every day | Data Engineers, Data Scientists, Data Platform experts and Data Analysts |
| Why | To overcome the limitations of SMP systems imposed on big data workloads | To take advantage of the OSS Big Data Analytics platform with SLA's in place to ensure business continuity | It provides the ability to create and manage an end to end big data/data science project using one platform | It provides the ability to scale efficiently with spark clusters within a one stop shop DataWarehousing platform of Synapse. |

***Spark Pools in Azure Synapse Analytics:*** Spark in Azure Synapse Analytics is a capability of Spark embedded in Azure Synapse Analytics in which organizations that don't have existing spark implementations yet, get the functionality to spin up a spark cluster to meet data engineering needs without the overhead of the other Spark Platforms listed. Data Engineers, Data scientist, Data Platform Experts, and Data Analyst can come together within Synapse Analytics where the Spark cluster is spun up quickly to meet the needs. It provides scale in an efficient way for Spark Clusters and integrates with the one stop shop Data warehousing platform of Synapse.

***Apache Spark:*** Apache Spark is an open-source memory optimized system for managing big data workloads, which is used when you want a spark engine for big data processing or data science where you don't mind that there is no SLA provided. Usually it is of interest of Open Source Professionals and the reason for Apache spark is to overcome the limitations of what was known as SMP systems for big data workloads.

***HDI:*** HDI is an implementation by Microsoft of Open Source Spark, managed on the Azure Platform. You can use HDI for a spark environment when you are aware of the benefits of Apache Spark in its OSS form, but you want a SLA. Usually this of interest of Open Source Professionals needing an SLA as well as Data Platform experts experienced with Microsoft.

***Azure Databricks:*** Azure Databricks is a managed Spark as a Service propriety Solution that provides an end to end data engineering/data science platform as a solution. Azure Databricks is of interest for Data Engineers and Data Scientists, working on big data projects daily because it provides the whole platform in which you have the ability to create and manage the big data/data science pipelines/projects all on one platform.

https://docs.microsoft.com/en-us/azure/synapse-analytics/spark/apache-spark-overview

**Scenario:** You are determining the type of Azure service needed to fit the following specifications and requirements:

**Data classification:** Semi-structured because of the need to extend or modify the schema for new products

**Operations:**

• Customers require a high number of read operations, with the ability to query many fields within the database.

• The business requires a high number of write operations to track its constantly changing inventory.

**Latency & throughput:** High throughput and low latency.

**Transactional support:** Because all of the data is both historical and yet changing, transactional support is required.

Which would be the best Azure service to select?

- ○ Azure Queue Storage

- ○ Azure Route Table

- ○ Azure Blob Storage

- ○ Azure SQL Database

- ○ Azure Cosmos DB
  **(Correct)**

**Explanation**
**Recommended service: Azure Cosmos DB**

Azure Cosmos DB supports semi-structured data, or No-SQL data, by design. So, supporting new fields, such as the "Bluetooth-enabled" field or any new fields you need in the future, is a given with Azure Cosmos DB.

Azure Cosmos DB supports SQL for queries and every property is indexed by default. You can create queries so that your customers can filter on any property in the catalogue.

Azure Cosmos DB is also ACID-compliant, so you can be assured that your transactions are completed according to those strict requirements.

As an added plus, Azure Cosmos DB also enables you to replicate your data anywhere in the world with the click of a button. So, if your e-commerce site has users concentrated in the US, France, and England, you can replicate your data to those data centres to reduce latency, as you've physically moved the data closer to your users.

Even with data replicated around the world, you can choose from one of five consistency levels. By choosing the right consistency level, you can determine the tradeoffs to make between consistency, availability, latency, and throughput. You can scale up to handle higher customer demand during peak shopping times, or scale down during slower times to conserve cost.
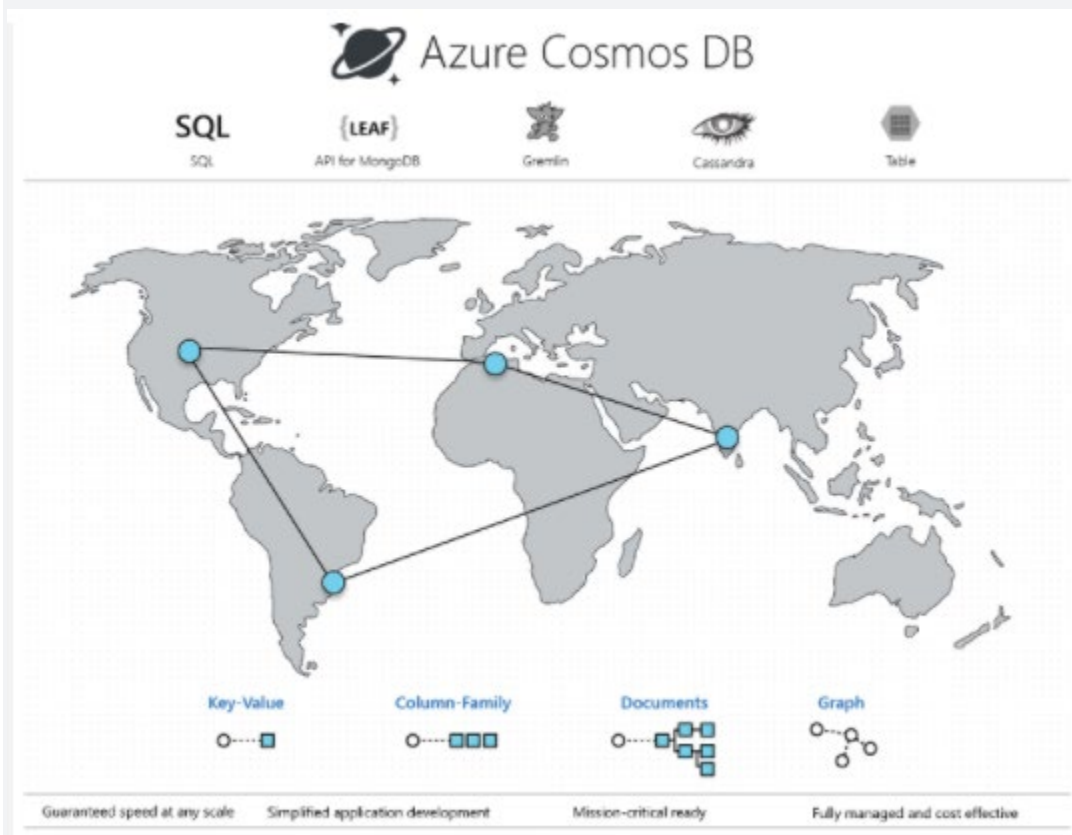
**Why not other Azure services?**

Azure SQL Database would be an excellent choice for this data set if you could identify the subset of properties that are common for most of the products and the variable properties that might not exist in some products. Azure SQL Database enables you to combine structured data in the columns, and semi-structured data stored as JSON columns that can be easily extended. Azure SQL Database can provide many of the same benefits of Azure Cosmos DB, but it provides little benefit if the structure of your data is changing in different entities, and you cannot pre-define a set of common properties that are repeated in most of the entities. Unlike Azure CosmosDB that index every property in the documents, in Azure SQL Database you need to explicitly define what properties from semi-structured documents should be indexed. Azure Cosmos DB is better choice for highly unstructured and variable data where you cannot predict what are the properties that should be indexed.

Other Azure services, such as Azure Table storage, Azure HBase as a part of HDInsight, and Azure Cache for Redis, can also store No-SQL data. In this scenario, users will want to query on multiple fields, so Azure Cosmos DB is a better fit. Azure Cosmos DB indexes every field by default, whereas the other services are limited in the data they index, and querying on non-indexed fields results in reduced performance.

When queries are submitted, a dedicated SQL pool query optimizer tries to determine which access paths to the data will result in the least amount of effort to retrieve the data required to resolve the query. It is a cost-based optimizer, and compares the cost of various query plans, and then chooses the plan with the lowest cost.

Statistics in a serverless SQL pool have the same objective of using a cost-based optimizer to choose an execution plan that will execute the fastest. How it creates the statistics is different.

**True or False:** In a serverless SQL pool, if statistics are missing, the query optimizer creates statistics on entire tables in the query predicate or join condition to improve cardinality estimates for the query plan.

- ○
  True

- ○
  False
  **(Correct)**

**Explanation**
When queries are submitted, a dedicated SQL pool query optimizer tries to determine which access paths to the data will result in the least amount of effort to retrieve the data required to resolve the query. It is a cost-based optimizer, and compares the cost of various query plans, and then chooses the plan with the lowest cost.

**Statistics in dedicated SQL pools**

To aid this process, statistics are required that describe the amount of data that is present within ranges of values, and range of rows that may be returned to fulfill a query filter or join. Therefore, after loading data into a dedicated SQL pool, collecting statistics on your data is one of the most important things you can do for query optimization.

When you create a database in a dedicated SQL pool in Azure Synapse Analytics, the automatic creation of statistics is turned on by default. This means that statistics are created when you run the following type of Transact-SQL statements:

- `SELECT`

- `INSERT-SELECT`

- `CTAS`

- `UPDATE`

- `DELETE`

- `EXPLAIN` when containing a join or the presence of a predicate is detected

When executing the above Transact-SQL statements, that the statistics creation is performed on the fly, and as a result, there can be a slight degradation in query performance.

To avoid this, statistics are also created on any index that you create that helps aid the query optimize process. As this is an action that is performed in advance of querying the table on which the index is based, it means that the statistics are created in advance. However, you must consider that as new data is loaded into the table, the statistics may become out of date.

As such, it is important to update the statistics after you load data or update large ranges of data, so that queries can benefit from the updated statistics information.

You can check if your data warehouse has `AUTO_CREATE_STATISTICS` configured by running the following command:

```SQL
SELECT name, is_auto_create_stats_on
FROM sys.databases
```

If your data warehouse doesn't have `AUTO_CREATE_STATISTICS` enabled, it is recommended that you enable this property by running the following command:

```SQL
ALTER DATABASE <yourdatawarehousename>
SET AUTO_CREATE_STATISTICS ON
```

**Statistics in serverless SQL pools**

Statistics in a serverless SQL pool has the same objective of using a cost-based optimizer to choose an execution plan that will execute the fastest. How it creates its statistics is different. Serverless SQL pool analyses incoming user queries for missing statistics. **If statistics are missing, the query optimizer creates statistics on individual columns in the query predicate or join condition to improve cardinality estimates for the query plan.** The `SELECT` statement will trigger automatic creation of statistics. You can also manually create statistics, this is important when working with CSV files, as automatic statistics creation is not enabled for them. In the following example, a system stored

procedure is used to specify the creation of statistics for a specific Transact-SQL statement

```SQL
sys.sp_create_openrowset_statistics [ @stmt = ] N'statement_text'
```

To create statistics for a specific column within a csv file, you can run the following code:

```SQL
/* make sure you have the credentials to access the storage account created

IF EXISTS (SELECT * FROM sys.credentials WHERE name = 'https://azureopendatastorage.blob.core.windows.net/censusdatacontainer')

DROP CREDENTIAL [https://azureopendatastorage.blob.core.windows.net/censusdatacontainer]

GO


CREATE CREDENTIAL [https://azureopendatastorage.blob.core.windows.net/censusdatacontainer]

WITH IDENTITY='SHARED ACCESS SIGNATURE',

SECRET = ''

GO

*/


/*

The following code will create statistics on a column named year, from a file named population.csv

*/


EXEC sys.sp_create_openrowset_statistics N'SELECT year

FROM OPENROWSET(

BULK ''https://sqlondemandstorage.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]

'
```

You should also update the statistics when the data in the files change. In fact, Serverless SQL pool automatically recreates statistics if data is changed significantly. Every time statistics are automatically created, the current state of the dataset is also saved: file paths, sizes, last modification dates. To update statistics for the year column in the dataset, which is based on the population.csv file, you need to drop and then create them, here is the drop statement:

```SQL
EXEC sys.sp_drop_openrowset_statistics N'SELECT year

FROM OPENROWSET(

BULK ''https://sqlondemandstorage.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]

'
```

To update statistics for a statement, you need to drop and create statistics. The following stored procedure is used to drop statistics against a specific Transact-SQL text:

```SQL
sys.sp_drop_openrowset_statistics [ @stmt = ] N'statement_text'
```

https://docs.microsoft.com/en-us/azure/synapse-analytics/sql/develop-tables-statistics

**Scenario:** You are working on a project where you create a DataFrame which is designed to read data from Azure Blob Storage. Next, you plan to create as additional DataFrame by filtering the initial DataFrame.

Which feature of Spark causes these transformation to be analyzed?

- ○

  Lazy Execution
  **(Correct)**

- ○

  Java Garbage Collection

- ○

  Tungsten Record Format

- ○

  Cluster configuration

**Explanation**
**Lazy Evaluation**

The concept of *lazy evaluation* means that Spark will wait until required to execute the graph of computation instructions.

Transformations applied to `DataFrame` are lazy, meaning they will not trigger any jobs. If you pass the `DataFrame` to a display function, a job will be triggered because display is an action.

https://www.linkedin.com/pulse/catalyst-tungsten-apache-sparks-speeding-engine-deepak-rajak?articleId=6674601890514378752

**Scenario:** Iceberg Lounge is Gotham's coolest night club and Penguin's pad for operations. It's a high-end nightclub in Gotham which acts as a legit forefront of his illegal activities.

You have been hired as a contractor for Iceberg Lounge and you are consulting on various IT functions. Oswald Cobblepot runs the show.

Today, the team is working on an enterprise-wide Azure Data Lake Storage Gen2 account. The data lake is accessible only through an Azure virtual network named VNET1.

Oswald wants the team to build a SQL pool in Azure Synapse that will use data from the data lake. All the members of the Iceberg Lounge sales team are in an Azure Active Directory group named Sales. POSIX controls are used to assign the Sales group access to the files in the data lake. The team plans to load data to the SQL pool every hour.

**Required:**
• Ensure that the SQL pool can load the sales data from the data lake.

The team has assembled some actions being considered to meet the requirement which are shown below. As the Azure expert, Oswald looks to you for advice of the correct actions to take.

Which of the following actions should you recommend they perform? (Select three)

- ☐
  Add your Azure Active Directory (Azure AD) account to the Sales group.

- ☐
  Use the snared access signature (SAS) as the credentials for the data load process.

- ☐
  Use the managed identity as the credentials for the data load process.
    **(Correct)**

- ☐
  Add the managed identity to the Sales group.
    **(Correct)**

- ☐
  Create a managed identity.
    **(Correct)**

- ☐
  Create a shared access signature (SAS).

**Explanation**
*The actions you should recommend are:*

*• Create a managed identity.*

*• Add the managed identity to the Sales group.*

*• Use the managed identity as the credentials for the data load process.*

The managed identity grants permissions to the dedicated SQL pools in the workspace.

*Note: Managed identity for Azure resources is a feature of Azure Active Directory. The feature provides Azure services with an automatically managed identity in.*

**Managed identities**

Managed identity for Azure resources is a feature of Azure Active Directory. The feature provides Azure services with an automatically managed identity in Azure AD. You can use the Managed Identity capability to authenticate to any service that support Azure AD authentication.

Managed identities for Azure resources are the new name for the service formerly known as Managed Service Identity (MSI).
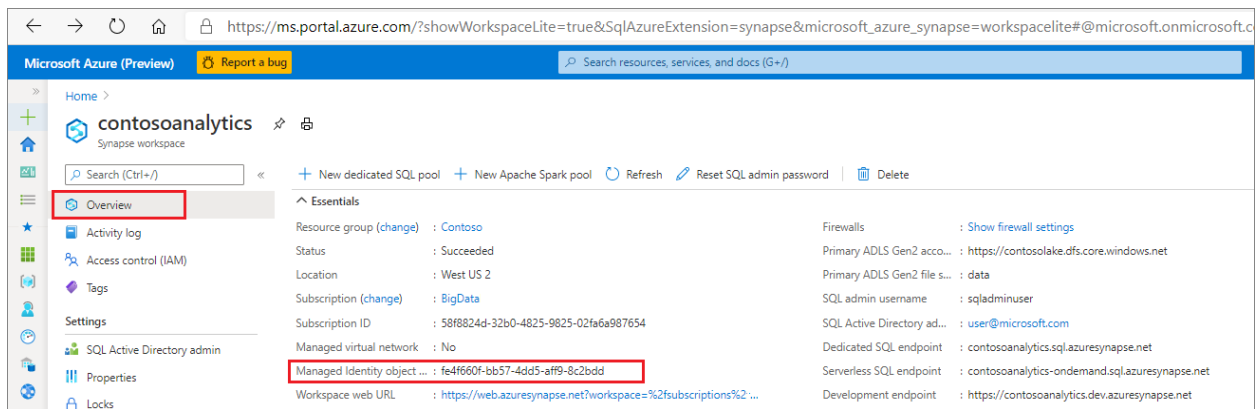
**Azure Synapse workspace managed identity**

A system-assigned managed identity is created for your Azure Synapse workspace when you create the workspace.

Azure Synapse uses the managed identity to integrate pipelines. The managed identity lifecycle is directly tied to the Azure Synapse workspace. If you delete the Azure Synapse workspace, then the managed identity is also cleaned up.

The workspace managed identity needs permissions to perform operations in the pipelines. You can use the object ID or your Azure Synapse workspace name to find the managed identity when granting permissions.

Retrieve managed identity in Azure portal

You can retrieve the managed identity in Azure portal. Open your Azure Synapse workspace in Azure portal and select **Overview** from the left navigation. The managed identity's object ID is displayed to in the main screen.

The managed identity information will also show up when you create a linked service that supports managed identity authentication from Azure Synapse Studio.

Launch **Azure Synapse Studio** and select the **Manage** tab from the left navigation. Then select **Linked services** and choose the **+ New** option to create a new linked service.
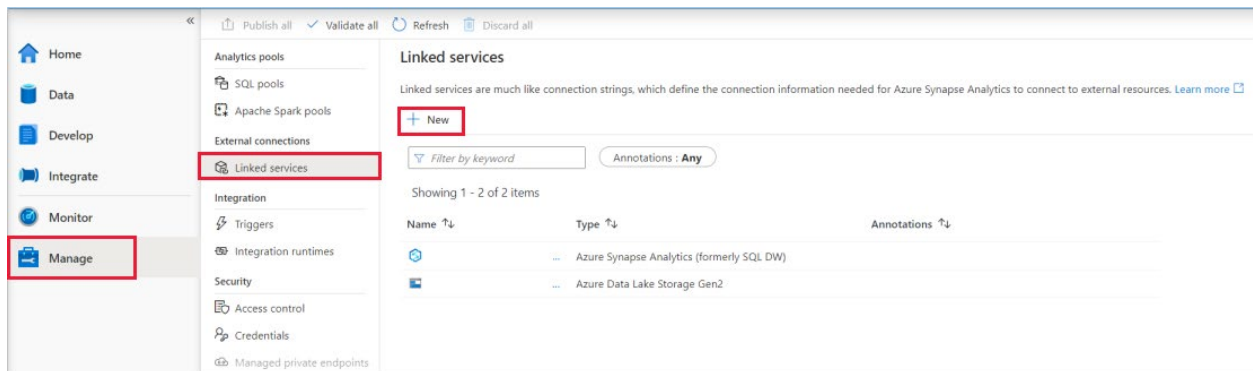


In the **New linked service** window, type *Azure Data Lake Storage Gen2*. Select the **Azure Data Lake Storage Gen2** resource type from the list below and choose **Continue**.
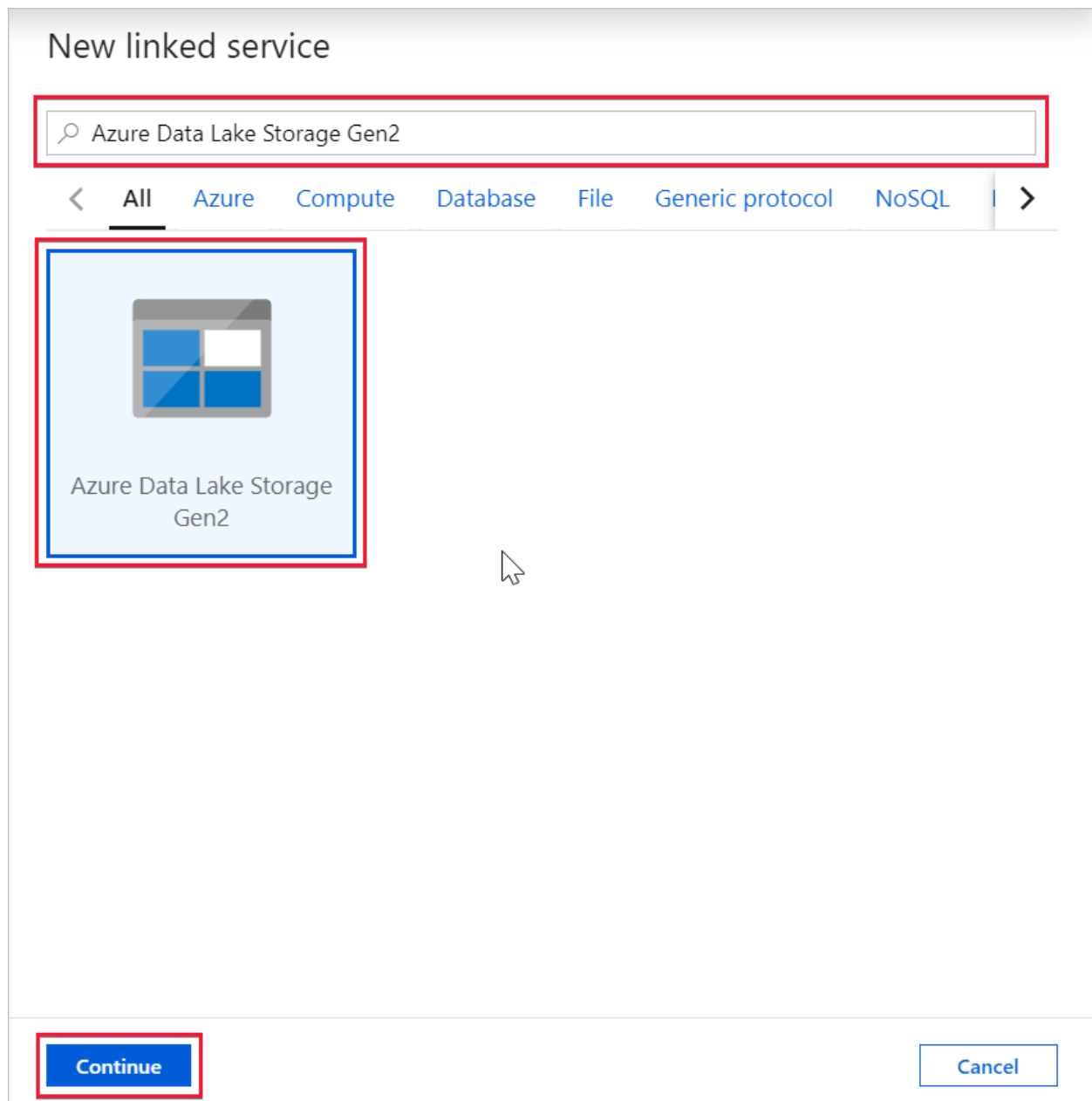
In the next window, choose **Managed Identity** for **Authentication method**. You'll see the managed identity's **Name** and **Object ID**.

New linked service (Azure Data Lake Storage Gen2)

Connect via integration runtime *

AutoResolveIntegrationRuntime

Authentication method

Managed Identity

Account selection method

⦿ From Azure subscription    ◯ Enter manually

Azure subscription

Select all

Storage account name *

Managed identity name: **ignitedemowus2**
Managed identity object ID: **1feefd93-aa22-4626-9d0c-473529613b7d**
Grant workspace service managed identity access to your Azure Data Lake Storage Gen2. Details

https://docs.microsoft.com/en-us/azure/synapse-analytics/security/synapse-workspace-managed-identity

**Scenario:** Stark Industries was founded by Howard Stark during the early twentieth century, a great pioneer in different types of technology and constantly helping the United States Armed Forces with different and innovative weapons. Some years after the death of Howard Stark, his son Tony took over the company.

Currently in the process of making many IT improvements at Stark Industries, you have been hired as a consultant to advise on several Microsoft Azure projects.

At the moment, the team is designing an Azure Stream Analytics job to process incoming events from sensors in retail environments.

**Required:**

• Process the events to produce a running average of shopper counts during the previous 15 minutes, calculated at five-minute intervals.

As the Azure expert, Tony expects you to advise the team on the best course of action.

Which type of window should you advise the team to use?

- ○ Snapshot

- ○ Sliding

- ○ Tumbling

- ○ Hopping
  **(Correct)**

**Explanation**
*You should advise the team to use a Hopping window.*

**Hopping Window (Azure Stream Analytics)**

Unlike tumbling windows, hopping windows model scheduled overlapping windows. A hopping window specification consist of three parameters: the *timeunit*, the *windowsize* (how long each window lasts) and the *hopsize* (by how much each window moves forward relative to the previous one). Additionally, *offsetsize* may be used as an optional fourth parameter. Note that a tumbling window is simply a hopping window whose 'hop' is equal to its 'size'.

The following illustration shows a stream with a series of events. Each box represents a hopping window and the events that are counted as part of that window, assuming that the 'hop' is 5, and the 'size' is 10.

Every 5 seconds give me the count of tweets over the last 10 seconds

A 10-second Hopping Window with a 5-second "Hop"
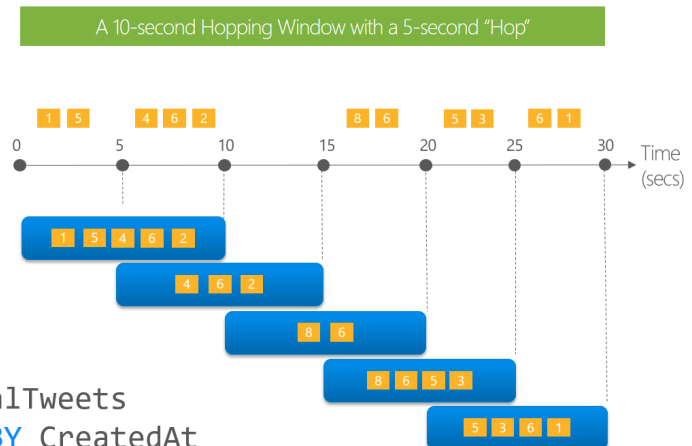
```
SELECT Topic, COUNT(*) AS TotalTweets
FROM TwitterStream TIMESTAMP BY CreatedAt
GROUP BY Topic, HoppingWindow(second, 10 , 5)
```

## Syntax

```
SQL

{HOPPINGWINDOW | HOPPING} ( timeunit  , windowsize , hopsize, [offsetsize] )

{HOPPINGWINDOW | HOPPING} ( Duration( timeunit  , windowsize ) , Hop (timeunit  ,
windowsize ), [Offset(timeunit  , offsetsize)])
```

## Arguments

## timeunit

Is the unit of time for the *windowsize* or the *hopsize*. The following table lists all valid *timeunit* arguments.

| Timeunit | Abbreviations |
|---|---|
| day | dd, d |
| hour | hh |
| minute | mi, n |
| second | ss, s |
| millisecond | ms |
| microsecond | mcs |

**windowsize**

A big integer which describes the size of the window. The windowsize is static and cannot be changed dynamically at runtime.

The maximum size of the window in all cases is 7 days.

**hopsize**

A big integer which describes the size of the Hop.

**offsetsize**

By default, hopping windows are inclusive in the end of the window and exclusive in the beginning – for example 12:05 PM – 1:05 PM window will include events that happened exactly at 1:05 PM, but will not include events that happened at 12:05:PM (these event will be part of 12:00 PM – 01:00 PM window). The Offset parameter can be used to change behaviour and include the events in the beginning of the window and exclude the ones that happened in the end.

**Time consideration**

Every window operation outputs event at the end of the window (in the case of hopping windows, this happens at every hop size). The windows of Azure Stream Analytics are opened at the window start time and closed at the window end time. For example, if you have a 5 minute window from 12:00 AM to 12:05 AM all events with timestamp greater than 12:00 AM and up to timestamp 12:05 AM inclusive will be included within this window. The output of the window will be a single event based on the aggregate function used with a timestamp equal to the window end time. The timestamp of the output event

of the window can be projected in the SELECT statement using the System.Timestamp() property using an alias.

https://docs.microsoft.com/en-us/stream-analytics-query/hopping-window-azure-stream-analytics


## Tumbling Window (Azure Stream Analytics)

Tumbling windows are a series of fixed-sized, non-overlapping and contiguous time intervals. The following diagram illustrates a stream with a series of events and how they are mapped into 10-second tumbling windows.
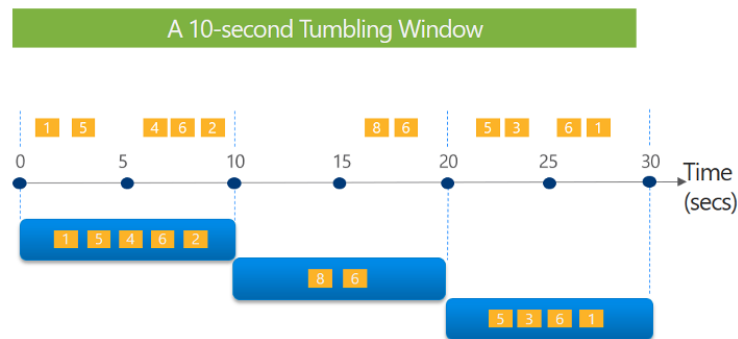


```
SELECT TimeZone, COUNT(*) AS Count
FROM TwitterStream TIMESTAMP BY CreatedAt
GROUP BY TimeZone, TumblingWindow(second,10)
```

**Time considerations**

Every window operation outputs event at the end of the window. The windows of Azure Stream Analytics are opened at the window start time and closed at the window end time. For example, if you have a 5 minute window from 12:00 AM to 12:05 AM all events with timestamp greater than 12:00 AM and up to timestamp 12:05 AM inclusive will be included within this window. The output of the window will be a single event based on the aggregate function used with a timestamp equal to the window end time. The timestamp of the output event of the window can be projected in the SELECT statement using the System.Timestamp() property using an alias.

https://docs.microsoft.com/en-us/stream-analytics-query/tumbling-window-azure-stream-analytics

Identify the missing word(s) in the following sentence within the context of Microsoft Azure.

Azure Cosmos DB supports 99.999 percent uptime. You can invoke a regional failover by using programming or the Azure portal. An Azure Cosmos DB database will automatically failover if there's a regional disaster. By using multimaster replication in Azure Cosmos DB, you can often achieve a response time of less than 1 second from anywhere in the world. Azure Cosmos DB is guaranteed to achieve a response time of less than [?] for reads and writes.

- 10 ms
  **(Correct)**

- 100 ms

- 1000 ms

- 200 ms

- 500 ms

- 1 ms

**Explanation**
**Azure Cosmos DB Key features**

Azure Cosmos DB supports 99.999 percent uptime. You can invoke a regional failover by using programing or the Azure portal. An Azure Cosmos DB database will automatically failover if there's a regional disaster. By using multimaster replication in Azure Cosmos DB, you can often achieve a response time of less than one second from anywhere in the world. **Azure Cosmos DB is guaranteed to achieve a response time of less than 10 ms for reads and writes.**

To maintain the consistency of the data in Azure Cosmos DB, your engineering team should introduce a new set of consistency levels that address the unique challenges of planet-scale solutions. Consistency levels include strong, bounded staleness, session, consistent prefix, and eventual.

https://azure.microsoft.com/en-us/support/legal/sla/cosmos-db/v1_3/

Identify the missing word(s) in the following sentence within the context of Microsoft Azure.

From a high level, the Azure Databricks service launches and manages Apache Spark clusters within your Azure subscription. Apache Spark clusters are groups of computers that are treated as a single computer and handle the execution of commands issued from notebooks.

In Databricks, the notebook interface … [?]

- is the driver program.
  **(Correct)**

- specifies the types and sizes of the virtual machines.

- provides the fastest virtualized network infrastructure in the cloud.

- pulls data from a specified data source.

**Explanation**
To gain a better understanding of how to develop with Azure Databricks, it is important to understand the underlying architecture. We will look at two aspects of the Databricks architecture: the Azure Databricks service and Apache Spark clusters.

**High-level overview**

From a high level, the Azure Databricks service launches and manages Apache Spark clusters within your Azure subscription. Apache Spark clusters are groups of computers that are treated as a single computer and handle the execution of commands issued from notebooks. Using a master-worker type architecture, clusters allow processing of data to be parallelized across many computers to improve scale and performance. They consist of a Spark Driver (master) and worker nodes. The driver node sends work to the worker nodes and instructs them to pull data from a specified data source.

**In Databricks, the notebook interface is the driver program.** This driver program contains the main loop for the program and creates distributed datasets on the cluster, then applies operations (transformations & actions) to those datasets. Driver programs access Apache Spark through a SparkSession object regardless of deployment location.
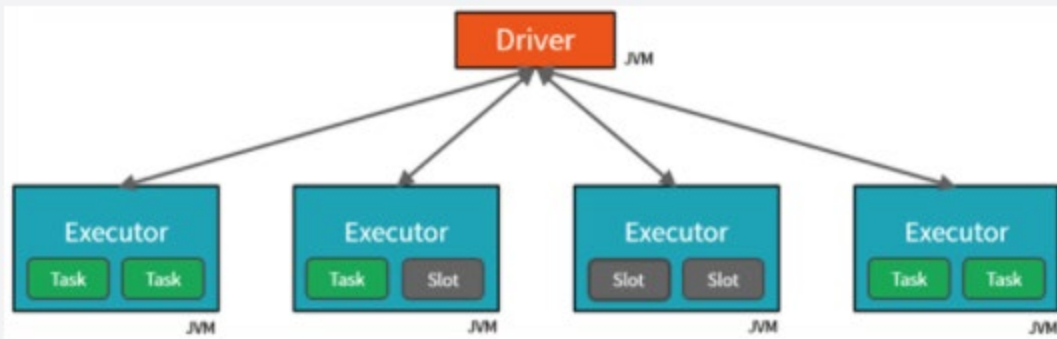
Microsoft Azure manages the cluster, and auto-scales it as needed based on your usage and the setting used when configuring the cluster. Auto-termination can also be enabled, which allows Azure to terminate the cluster after a specified number of minutes of inactivity.

**Under the covers**

Now let's take a deeper look under the covers. When you create an Azure Databricks service, a "Databricks appliance" is deployed as an Azure resource in your subscription. At the time of cluster creation, you specify the types and sizes of the virtual machines (VMs) to use for both the Driver and Worker nodes, but Azure Databricks manages all other aspects of the cluster.

You also have the option of using a Serverless Pool. A Serverless Pool is self-managed pool of cloud resources that is auto-configured for interactive Spark workloads. You provide the minimum and maximum number of workers and the worker type, and Azure Databricks provisions the compute and local storage based on your usage.

The "Databricks appliance" is deployed into Azure as a managed resource group within your subscription. This resource group contains the Driver and Worker VMs, along with other required resources, including a virtual network, a security group, and a storage account. All metadata for your cluster, such as scheduled jobs, is stored in an Azure Database with geo-replication for fault tolerance.

Internally, Azure Kubernetes Service (AKS) is used to run the Azure Databricks control-plane and data-planes via containers running on the latest generation of Azure hardware (Dv3 VMs), with NvMe SSDs capable of blazing 100us latency on IO. These make Databricks I/O performance even better. In addition, accelerated networking provides the fastest virtualized network infrastructure in the cloud. Azure Databricks utilizes these features to further improve Spark performance. Once the services within this managed resource group are ready, you will be able to manage the Databricks cluster through the Azure Databricks UI and through features such as auto-scaling and auto-termination.

https://databricks.com/blog/2017/11/15/a-technical-overview-of-azure-databricks.html
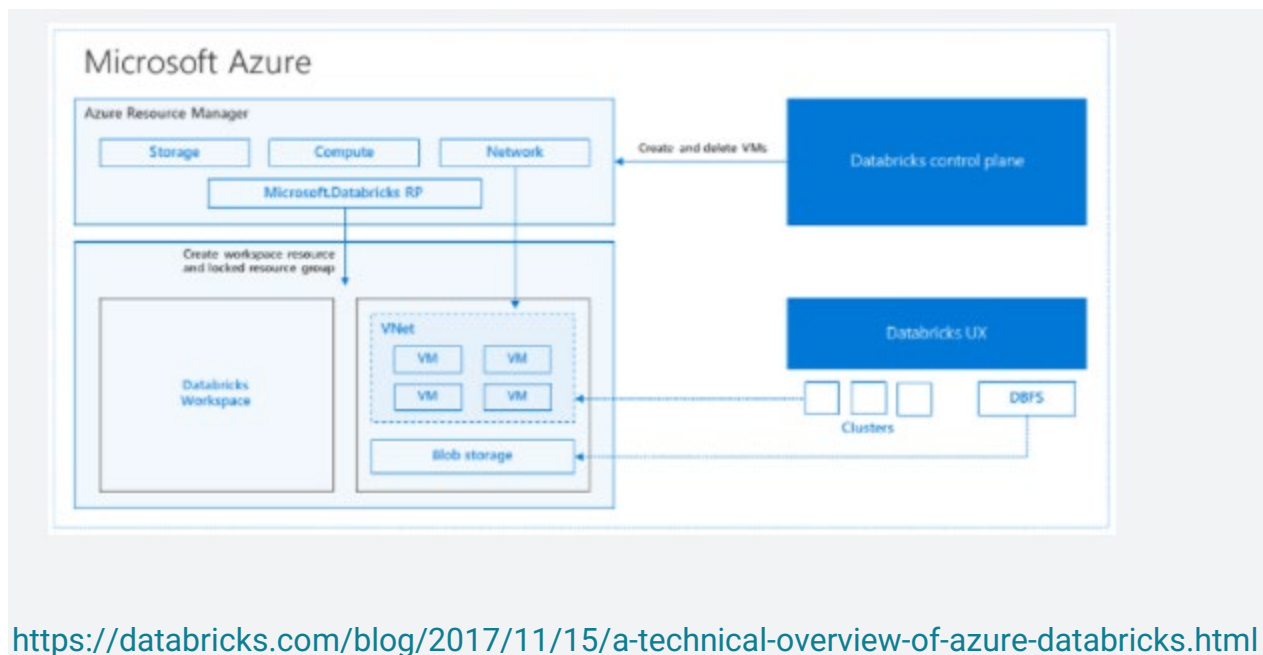
**Scenario:** The company you work at is a financial services firm, and can only have account managers allowed to access a customer's social insurance number, phone numbers or other personal identifiable information. It is imperative to distinguish the role of an account manager versus the manager of the account managers.

Which type of security would typically be best used in for this scenario?

- Dynamic Data Masking

- Row-level security

- Column-level security
    **(Correct)**

- Table-level security

**Explanation**
Authentication is the process of validating credentials as you access resources in a digital infrastructure. This ensures that you can validate that an individual, or a service that wants to access a service in your environment can prove who they are. Azure Synapse Analytics provides several different methods for authentication.

**Column level security in Azure Synapse Analytics**

**Generally speaking, column level security is simplifying a design and coding for the security in your application. It allows you to restrict column access in order to protect sensitive data.** For example, if you want to ensure that a specific user 'Leo' can only access certain columns of a table because he's in a specific department. The logic for 'Leo' only to access the columns specified for the department he works in, is a logic that is located in the database tier, rather on the application level data tier. If he needs to access data from any tier, the database should apply the access restriction every time he tries to access data from another tier. The reason for doing so, is to make sure that your security is reliable and robust since we're reducing the surface area of the overall security system. Column level security will also eliminate the necessity for the introduction of view, where you would filter out columns, to impose access restrictions on 'Leo'

The way to implement column level security, is by using the `GRANT` T-SQL statement. Using this statement, SQL and Azure Active Directory (AAD) support the authentication.

The syntax to use for implementing column level security looks as follows:

```SQL
GRANT <permission> [ ,...n ] ON

[ OBJECT :: ][ schema_name ]. object_name [ ( column [ ,...n ] ) ] // specifying
the column access

TO <database_principal> [ ,...n ]

[ WITH GRANT OPTION ]

[ AS <database_principal> ]

<permission> ::=

SELECT

| UPDATE

<database_principal> ::=

Database_user // specifying the database user

| Database_role // specifying the database role

| Database_user_mapped_to_Windows_User

| Database_user_mapped_to_Windows_Group
```

So when would you use column-level security? Let's say that you are a financial services firm, and can only have account manager allowed to have access to a customer's social security number, phone numbers or other personal identifiable information. It is imperative to distinguish the role of an account manager versus the manager of the account managers.

Another use case might be related to the Healthcare Industry. Let's say you have a specific health care provider. This healthcare provider only wants doctors and nurses to

be able to access medical records. The billing department should not have access to view this data. Column-level security would typically be the option to use.

**Row level security in Azure Synapse Analytics**

Row-level security (RLS) can help you to create a group membership or execution context in order to control not just columns in a database table, but actually, the rows. RLS, just like column-level security, can simply help and enable your design and coding of your application security. However, compared to column-level security where it's focused on the columns (parameters), RLS helps you implement restrictions on data row access. Let's say that your employee can only access rows of data that are important of the department, you should implement RLS. If you want to restrict for example, customer's data access that is only relevant to the company, you can implement RLS. The restriction on access of the rows, is a logic that is located in the database tier, rather on the application level data tier. If 'Leo' needs to access data from any tier, the database should apply the access restriction every time he tries to access data from another tier. The reason for doing so, is to make sure that your security is reliable and robust since we're reducing the surface area of the overall security system.

The way to implement RLS is by using the `CREATE SECURITY POLICY[!INCLUDEtsql]` statement. The predicates are created as inline table-valued functions. It is imperative to understand that within Azure Synapse, it only supports filter predicates. If you need to use a block predicate, you won't be able to find support at this moment within in Azure synapse.



**Description of row level security in relation to filter predicates**

RLS within Azure Synapse supports one type of security predicates, which are Filter predicates, not block predicates. What filter predicates do, are silently filtering the rows that are available for read operations such as `SELECT`, `UPDATE`, `DELETE`.

The access to row-level data in a table, is restricted as an inline table-valued function, which is a security predicate. This table-valued function will then be invoked and enforced by the security policy that you need. An application, is not aware of rows that are filtered from the result set for filter predicates. So what will happen is that if all rows are filtered, a null set is returned.

When you are using filter predicates, it will be applied when data is read from the base table. The filter predicate affects all get operations such as `SELECT`, `DELETE`, `UPDATE`. You are unable to select or delete rows that have been filtered. It is not possible for you to update a row that has been filtered. What you can do, is update rows in a way that they will be filtered afterwards.

**Permissions**

If you want to create, alter or drop the security policies, you would have to use the `ALTER ANY SECURITY POLICY` permission. The reason for that is when you are creating or dropping a security policy it requires `ALTER` permissions on the schema.

In addition to that, there are other permissions required for each predicate that you would add:

• `SELECT` and `REFERENCES` permissions on the inline table-valued function being used as a predicate.

• `REFERENCES` permission on the table that you target to be bound to the policy.

• `REFERENCES` permission on every column from the target table used as arguments.

Once you've set up the security policies, they will apply to all the users (including dbo users in the database) Even though DBO users can alter or drop security policies, their changes to the security policies can be audited. If you have special circumstances where highly privileged users, like a sysadmin or db_owner, need to see all rows to troubleshoot or validate data, you would still have to write the security policy in order to allow that.

If you have created a security policy where `SCHEMABINDING = OFF`, in order to query the target table, the user must have the SELECT or EXECUTE permission on the predicate function. They also need permissions to any additional tables, views, or functions used within the predicate function. If a security policy is created with `SCHEMABINDING = ON` (the default), then these permission checks are bypassed when users query the target table.

**Best practices**

There are some best practices to take in mind when you want to implement RLS. We recommended creating a separate schema for the RLS objects. RLS objects in this context would be the predicate functions, and security policies. Why is that a best practice? It helps to separate the permissions that are required on these special objects from the target tables. In addition to that, separation for different policies and predicate functions may be needed in multi-tenant-databases. However, it is not a standard for every case.

Another best practice to bear in mind is that the `ALTER ANY SECURITY POLICY` permission should only be intended for highly privileged users (such as a security policy manager). The security policy manager should not require `SELECT` permission on the tables they protect.

In order to avoid potential runtime errors, you should take in mind type conversions in predicate functions that you write. Also, you should try to avoid recursion in predicate functions. The reason for this is to avoid performance degradation. Even though the query optimizer will try to detect the direct recursions, there is no guarantee to find the indirect recursions. With an indirect recursion we mean where a second function call the predicate function.

It would also be recommended to avoid the use of excessive table joins in predicate functions. This would maximize performance.

Generally speaking when it comes to the logic of predicates, you should try to avoid logic that depends on session-specific SET options. Even though this is highly unlikely to be used in practical applications, predicate functions whose logic depends on certain session-specific `SET` options can leak information if users are able to execute arbitrary queries. For example, a predicate function that implicitly converts a string to **datetime** could filter different rows based on the `SET  DATEFORMAT` option for the current session.

https://docs.microsoft.com/en-us/azure/synapse-analytics/sql-data-warehouse/column-level-security

Azure Cosmos DB analytical store is a fully isolated column store for enabling large-scale analytics against operational data in your Azure Cosmos DB, without any impact to your transactional workloads.

**True or False:** You can only enable analytical store at the time of creating a new container.

- ○ False

- ○ True
  **(Correct)**

**Explanation**
Azure Cosmos DB analytical store is a fully isolated column store for enabling large-scale analytics against operational data in your Azure Cosmos DB, without any impact to your transactional workloads.

**Row-oriented transactional store**

Operational data in an Azure Cosmos DB container is internally stored in an indexed row-based "transactional store". The row store format and its associated b-tree index are designed to allow fast transactional reads and writes with single-digit millisecond response times, and high-performance operational queries. As your dataset grows large, complex analytical queries can become expensive as they use up more of the provisioned throughput resources. The increased consumption of provisioned throughput in turn impacts the performance of transactional workloads.

**Column-oriented analytical store**

Azure Cosmos DB analytical store addresses the complexity and latency challenges that occur with the traditional ETL pipelines. Azure Cosmos DB analytical store can automatically sync data from the transactional store into a separate column store. Column store format is suitable for large-scale analytical queries to be performed in an optimized manner, resulting in improving the latency of such queries.
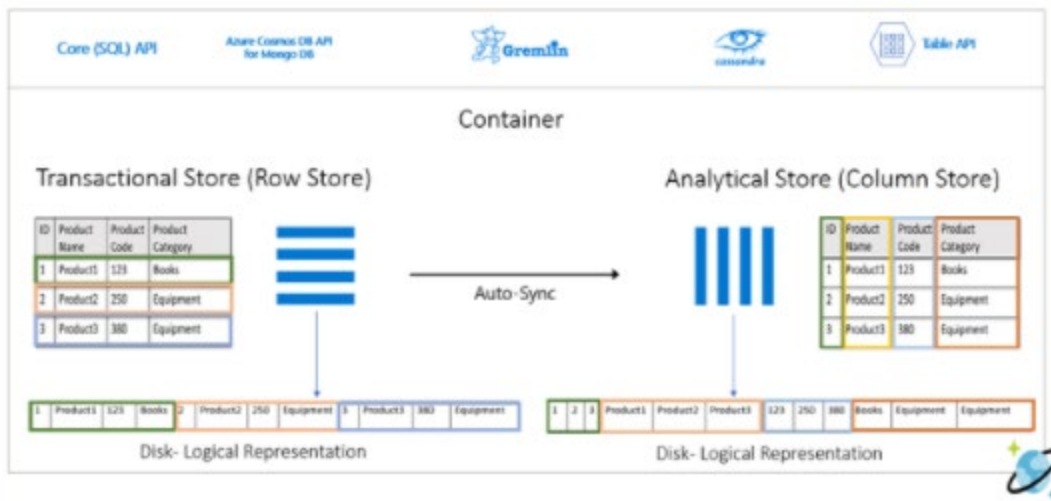
## Features of the analytical store

When you create an Azure Cosmos DB container you have the option of enabling analytical store, a new column-store structure is created within the container duplicating the data of the transactional store. This column store structure data is persisted separately from the row-oriented transactional store with the inserts, updates, and deletes performed on the transactional store being transparently copied by means of a fully managed internal autosync process to the analytical store in near real time.

*Note:*

*• You can only enable analytical store at the time of creating a new container.*

*• Azure Synapse Link is supported for the Azure Cosmos DB SQL (Core) API and for the Azure Cosmos DB API for MongoDB.*

Data is typically automatically synchronized between the transactional store and the analytical store within 2 minutes by means of the autosync process. However, in some circumstances -most notably in situations shared throughput database with many containers, the autosync latency could take up to 5 minutes.

Due to the fact that the transactional store and analytical store are persisted and queried separately the workloads associated with these stores are isolated from each other, that is to say queries against the analytical store (or the autosync process itself) does not impact the performance of nor use up resources (throughput or request units) provisioned for the transactional store, and operations performed against the transactional store does not impact autosync latency.

*Note:*

*• The transactions (read & write) and storage costs for the analytical store are charged separately from the transactional store storage and throughput.*

The autosync process also takes care of schema updates to the schematized analytical store automatically for you as unique new properties are added over time to items within your container. This allows you to take advantage of the performance advantages provided by schematization without any effort on your part. We will get into more of the details of how analytical store schema is managed and exposed to the Synapse Analytics query capabilities in the next unit.

You can configure the default Time to Live (TTL) property for records stored within the transactional store and analytical store independently of each other. The TTL value of a record defines when it will be automatically deleted from the store. By configuring the default TTL value of both stores, you can manage the lifecycle of data and define how long it will be retained for in each store. You can override the default TTL value (at the item level) for the transactional store however the default TTL value will always apply to data in the analytical and cannot be overwritten at the item level.

Azure Cosmos DB support global distributed accounts replicating your containers transparently to the Azure regions choose. When enabled on a container the analytical store will automatically be configured in all chosen global distribution regions, you cannot selectively choose which regions to deploy an analytical store. It is also recommended that you choose and configure your global distribution regions on the account prior to enabling analytical store on a container.

https://docs.microsoft.com/en-us/azure/cosmos-db/analytical-store-introduction

Which of the following tools are used to create and deploy SQL Server Integration Packages on an Azure-SSIS integration runtime, or for on-premises SQL Server?

- ☐
  SQL Server Management Studio
  **(Correct)**

- ☐
  Data Migration Assistant

- ☐
  Data Migration Service

- ☐
  SQL Server Data Tools
  **(Correct)**

- ☐
  `dtexec`

- ☐
  SQL Server Upgrade Advisor

- ☐
  SQL Server Management Studio

- ☐
  Data Migration Assessment

**Explanation**
**SQL Server Data Tools** is typically used to create and deploy SQL Server Integration Services (SSIS) packages.

https://docs.microsoft.com/en-us/sql/integration-services/lift-shift/ssis-azure-lift-shift-ssis-packages-overview?view=sql-server-ver15

When you use Package Deployment Model, you can choose whether you want to provision your Azure-SSIS IR with package stores. They provide a package management layer on top of file system, Azure Files, or MSDB hosted by Azure SQL Managed Instance. Azure-SSIS IR package store allows you to import/export/delete/run packages and monitor/stop running packages via **SQL Server Management Studio (SSMS)** similar to the legacy SSIS package store.

https://docs.microsoft.com/en-us/azure/data-factory/azure-ssis-integration-runtime-package-store

**Question 22:** <mark>Skipped</mark>

When performing the batch movement of data to populate a data warehouse, it is typical for the data engineer to understand the schedule on which the data loads take place. In these circumstances, you may be able to predict the periods of downtime in the data loading and querying process and take advantage of the pause operations to minimize your costs.

In the Azure Portal you can use the Pause command within the dedicated SQL pool and within Azure Synapse Studio, in the [?] hub which allows you to enable it, and set the number of minutes idle.

- ○ Develop

- ○ Integrate

- ○ Ingest

- ○ Data

- ○ Manage
  **(Correct)**

- ○ Monitor

- ○ Explore and analyze

**Explanation**

When performing the batch movement of data to populate a data warehouse, it is typical for the data engineer to understand the schedule on which the data loads take place. In these circumstances, you may be able to predict the periods of downtime in the data loading and querying process and take advantage of the pause operations to minimize your costs.

In the Azure Portal you can use the Pause command within the dedicated SQL pool and within Azure Synapse Studio, in the Manage hub which allows you to enable it, and set the number of minutes idle.

## Auto-pause settings

⬚ SparkPool01

Configure the auto-pause settings for the
Apache Spark pool.

Auto-pause * ⓘ

**Enabled**    Disabled

Number of minutes idle *

15

https://docs.microsoft.com/en-us/azure/synapse-analytics/sql-data-warehouse/pause-and-resume-compute-powershell

The Manage hub enables you to perform some of the same actions as in the Azure Portal, such as managing SQL and Spark pools. However, there is a lot more you can do in this hub that you cannot do anywhere else, such as managing Linked Services and integration runtimes, and creating pipeline triggers.

https://docs.microsoft.com/en-us/azure/synapse-analytics/quickstart-power-bi

**Question 23:** <mark>Skipped</mark>

How do you create a DataFrame object? (Select all that apply)

- ☐ Use the `createDataFrame()` function
  **(Correct)**

- ☐ Introduce a variable name and equate it to something like `myDataFrameDF =`
  **(Correct)**

- ☐ Use the `DF.create()` syntax

- ☐ Execute `createOrReplaceObject()`

**Explanation**

The approach *"Introduce a variable name and equate it to something like* `myDataFrameDF =` *"* is a correct way to create `DataFrame` objects.

https://docs.microsoft.com/en-us/azure/databricks/getting-started/spark/dataframes

You can also *use the* `createDataFrame()` *function* to create `DataFrame` objects.

https://docs.microsoft.com/en-us/dotnet/api/microsoft.spark.sql.sparksession.createdataframe?view=spark-dotnet

It is a good practice to store documentation about a data source.

Which Azure service is the best choice to do this?

- ○

  Azure Data Catalogue
  **(Correct)**

- ○

  Azure Data Factory

- ○

  Azure Stream Analytics

- ○

  Azure Databricks

- ○

  Azure Data Lake Storage

**Explanation**
Azure Data Catalogue is a central place where an organization's users can contribute their knowledge. Together, they build a community of data sources that the organization owns.

**Azure Data Catalogue**

Analysts, data scientists, developers, and others use Data Catalogue to discover, understand, and consume data sources. Data Catalogue features a crowdsourcing model of metadata and annotations. In this central location, an organization's users contribute their knowledge to build a community of data sources that are owned by the organization.

Data Catalogue is a fully managed cloud service. Users discover and explore data sources, and they help the organization document information about their data sources.

https://docs.microsoft.com/en-us/azure/data-catalog/overview

You can natively perform data transformations with Azure Synapse pipelines code free using the Mapping Data Flow task. Mapping Data Flows provide a fully visual experience with no coding required. Your data flows will run on your own execution cluster for scaled-out data processing.

Data flow activities can be operationalized via which of the following? (Select four)

- ☐ Flow capabilities
  **(Correct)**

- ☐ Integrate hub

- ☐ Monitoring capabilities
  **(Correct)**

- ☐ Manage hub

- ☐ Control capabilities
  **(Correct)**

- ☐ Data Factory scheduling
  **(Correct)**

- ☐ Data hub

- ☐ Monitor hub

**Explanation**
You can natively perform data transformations with Azure Synapse pipelines code free using the Mapping Data Flow task. Mapping Data Flows provide a fully visual experience with no coding required. Your data flows will run on your own execution cluster for scaled-out data processing. **Data flow activities can be operationalized via existing Data Factory scheduling, control, flow, and monitoring capabilities.**

When building data flows, you can enable debug mode, which turns on a small interactive Spark cluster. Turn on debug mode by toggling the slider at the top of the authoring module. Debug clusters take a few minutes to warm up, but can be used to interactively preview the output of your transformation logic.

With the Mapping Data Flow added, and the Spark cluster running, this will enable you to perform the transformation, and run and preview the data. No coding is required as Azure Data Factory handles all the code translation, path optimization, and execution of your data flow jobs.

*Note:*

*• If your dataset is pointing at a folder with other files and you only want to use one file, you may need to create another dataset or utilize parameterization to make sure only a specific file is read*

*• If you have not imported your schema in your ADLS, but have already ingested your data, go to the dataset's 'Schema' tab and click 'Import schema' so that your data flow knows the schema projection.*

Mapping Data Flow follows an extract, load, transform (ELT) approach and works with staging datasets that are all in Azure. Currently the following datasets can be used in a source transformation:

• Azure Blob Storage (JSON, Avro, Text, Parquet)

• Azure Data Lake Storage Gen1 (JSON, Avro, Text, Parquet)

• Azure Data Lake Storage Gen2 (JSON, Avro, Text, Parquet)

• Azure Synapse Analytics

• Azure SQL Database

• Azure CosmosDB

Azure Data Factory has access to over 80 native connectors. To include data from those other sources in your data flow, use the Copy Activity to load that data into one of the supported staging areas. Once your debug cluster is warmed up, verify your data is loaded correctly via the Data Preview tab. Once you click the refresh button, Mapping Data Flow will show a snapshot of what your data looks like when it is at each transformation.

https://techcommunity.microsoft.com/t5/azure-synapse-analytics/ingest-and-transform-data-with-azure-synapse-analytics-with-ease/ba-p/1975563

In Azure Synapse Studio, the Data hub is where you access which of the following? (Select three)

- ☐ Provisioned SQL pool databases
  **(Correct)**

- ☐ Pipeline canvas

- ☐ Master Pipeline

- ☐ SQL scripts

- ☐ Data flows

- ☐ SQL serverless databases
  **(Correct)**

- ☐ Notebooks

- ☐ Power BI

- ☐ External data sources
  **(Correct)**

- ☐ Activities

**Explanation**

In Azure Synapse Studio, the Data hub is where you access your provisioned **SQL pool databases and SQL serverless databases in your workspace, as well as external data sources, such as storage accounts and other linked services.**

Every Synapse workspace has a primary ADLS Gen2 account associated with it. This serves as the data lake, which is a great place to store flat files, such as files copied over from on-premises data stores, exported data or data copied directly from external services and applications, telemetry data, etc. Everything is in one place.

The file explorer capabilities allow you to quickly find files and perform actions on them, like preview file contents, generate new SQL scripts or notebooks to access the file, create a new data flow or dataset, and manage the file.
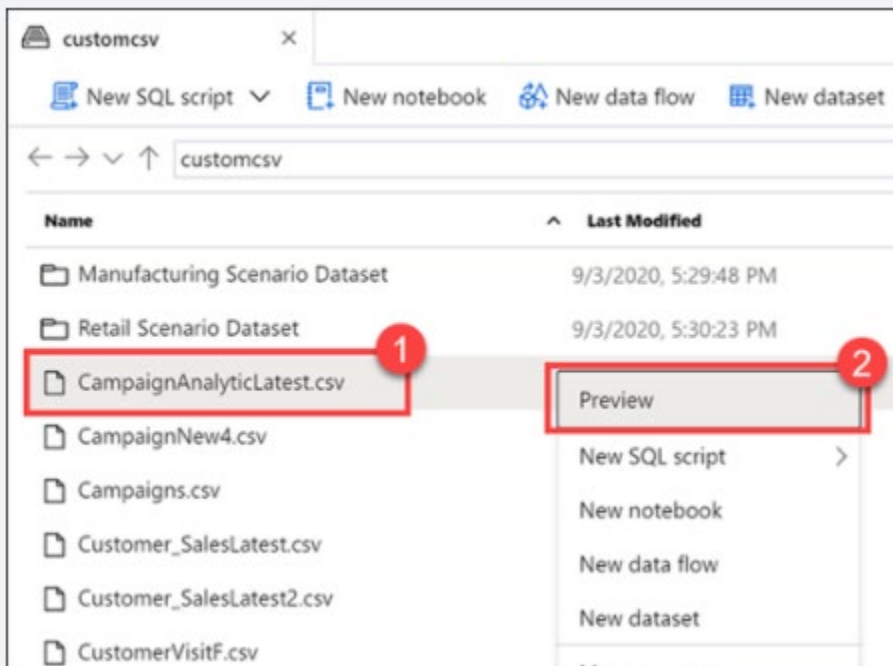
Integration Runtime (IR) is the compute infrastructure used by Azure Data Factory. It provides data integration capabilities across different network environments.

Which of the following are valid data integration capabilities? (Select four)

- ☐ Activity dispatch
  **(Correct)**

- ☐ Data transformation activities

- ☐ Control Flow

- ☐ Data Flow
  **(Correct)**

- ☐ Test Lab execution

- ☐ Analytic dispatch

- ☐ SSIS package execution
  **(Correct)**

- ☐ Data storage

- ☐ Data movement
  **(Correct)**

**Explanation**

In Data Factory, an activity defines the action to be performed. A linked service defines a target data store or a compute service. An integration runtime provides the infrastructure for the activity and linked services.

Integration Runtime is referenced by the linked service or activity, and provides the compute environment where the activity either runs on or gets dispatched from. This way, the activity can be performed in the region closest possible to the target data store or compute service in the most performant way while meeting security and compliance needs.

In short, the Integration Runtime (IR) is the compute infrastructure used by Azure Data Factory. It provides the following data integration capabilities across different network environments, including:

• **Data Flow**: Execute a Data Flow in managed Azure compute environment.

• **Data movement**: Copy data across data stores in public network and data stores in private network (on-premises or virtual private network). It provides support for built-in connectors, format conversion, column mapping, and performant and scalable data transfer.

• **Activity dispatch**: Dispatch and monitor transformation activities running on a variety of compute services such as Azure Databricks, Azure HDInsight, Azure Machine Learning, Azure SQL Database, SQL Server, and more.

• **SSIS package execution**: Natively execute SQL Server Integration Services (SSIS) packages in a managed Azure compute environment.

Whenever an Azure Data Factory instance is created, a default Integration Runtime environment is created that supports operations on cloud data stores and compute services in public network. This can be viewed when the integration runtime is set to Auto-Resolve.

https://docs.microsoft.com/en-us/azure/data-factory/concepts-integration-runtime

**Scenario**: You are working in the sales department of a company and part of your role is to manage the storage of customer profile and sales data. A common request is to generate a list of *"the top 100 customers including name, account number and sales figures for a given time period"* or *"who are the customers within a given geographic region?"*

Is Azure Blob storage a good choice for this data?

- ○ Yes

- ○ No
  **(Correct)**

**Explanation**

Blobs are not appropriate for structured data that needs to be queried frequently. They have higher latency than memory and local disk and don't have the indexing features that make databases efficient at running queries.

https://docs.microsoft.com/en-us/azure/storage/blobs/storage-blobs-introduction

Within Azure Data Factory, it is possible to parameterize a linked service in which you can pass through dynamic values while at run time.

Which of the following is a benefit of to parameterizing a linked service in Azure Data Factory?

- You don't have to create a single linked service for each database that uses a set of SQL Servers. A single parameterized linked service can be used for multiple SQL Servers regardless if they are all of the same type of SQL Server or not (Azure, MySQL, MariaDB, PostgreSQL, Oracle, Amazon...). They must all be relational database types.

- You don't have to create a single linked service for each database that is on the same SQL Server.
  **(Correct)**

- None of the listed options.

- You don't have to create a single linked service for each database that uses a set of SQL Servers. A single parameterized linked service can be used for multiple SQL Servers providing they are all of the same type of SQL Server (Azure, MySQL, MariaDB, PostgreSQL, Oracle, Amazon...).

**Explanation**
**Parameterize linked services in Azure Data Factory**

Within Azure Data Factory, it is possible to parameterize a linked service in which you can pass through dynamic values while at run time. A use-case for this situation could be connecting to several different databases that are on the same SQL server, in which you might think about parameterizing the database name in the linked service definition. **The benefit of doing so, is that you don't have to create a single linked service for each database that is on the same SQL Server.** It is also possible to parameterize other properties of the linked service like a Username.

If you decide to parameterize linked services in Azure Data Factory, you have the possibility to do so in the Azure Data Factory User Interface, the Azure portal or a programming interface to your liking.

If you choose to author the linked service through the User Interface, Data Factory can provide you with built-in parameterization for some of the connectors:

• Amazon Redshift

• Azure Cosmos DB (SQL API)

• Azure Database for MySQL

• Azure SQL Database

• Azure Synapse Analytics (formerly SQL DW)

• MySQL

• Oracle

• SQL Server

• Generic HTTP

• Generic REST

If you navigate to the creation/edit blade of the linked service, you will find the options for the parameterizing.
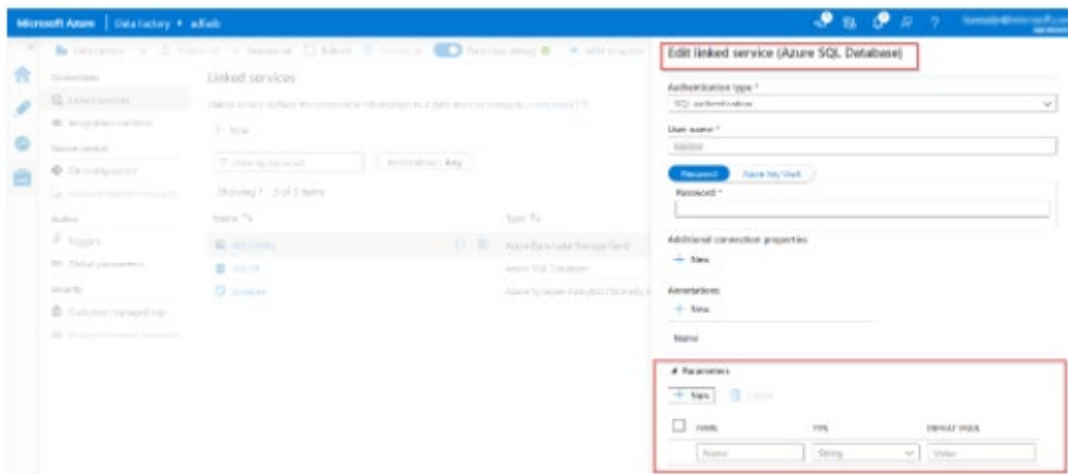


If you cannot use the built-in parameterization since you're using a different type of connector, you are able to edit the JSON through the UI:

In linked service creation/edit blade → expand "Advanced" at the bottom → check "Specify dynamic contents in JSON format" checkbox → specify the linked service JSON payload.
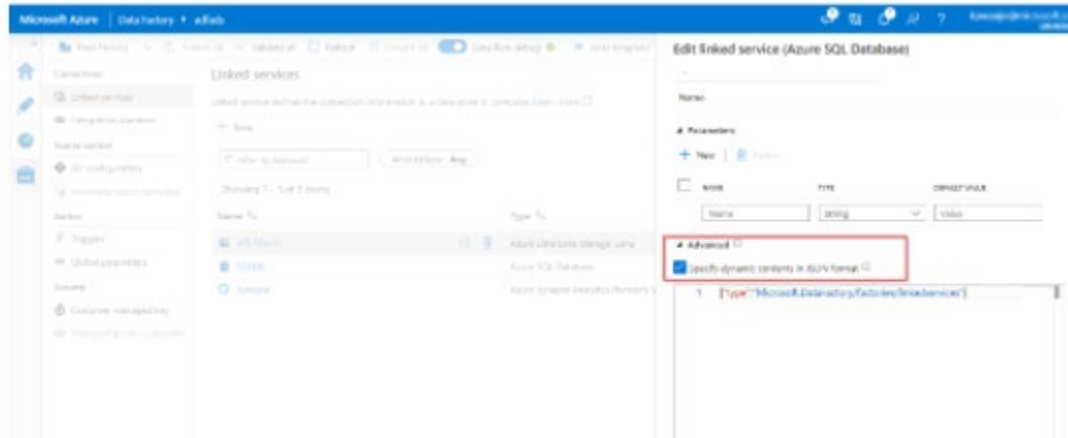


Or, after you create a linked service without parameterization, in Management hub → Linked services → find the specific linked service → click "Code" (button "{}") to edit the JSON.

https://docs.microsoft.com/en-us/azure/data-factory/parameterize-linked-services

**Scenario:** Your teammate is using import statements for transferring data between a dedicated SQL and Spark pool. Another teammate believes this is not necessary.

When is it unnecessary to use import statements for transferring data between a dedicated SQL and Spark pool?

- ○

  Use the PySpark connector.

- ○

  None of the listed options.

- ○

  Use token-based authentication.

- ○

  It is always necessary to use import statements for transferring data between a dedicated SQL and Spark pool.

- ○

  Import statements are not needed since they are pre-loaded with the Azure Synapse Studio integrated notebook experience.
  **(Correct)**

**Explanation**

**Import statements are not needed since they are pre-loaded with the Azure Synapse Studio integrated notebook experience.**

A Synapse Studio notebook is a web interface for you to create files that contain live code, visualizations, and narrative text. Notebooks are a good place to validate ideas and use quick experiments to get insights from your data. Notebooks are also widely used in data preparation, data visualization, machine learning, and other Big Data scenarios.

With an Azure Synapse Studio notebook, you can:

• Get started with zero setup effort.

• Keep data secure with built-in enterprise security features.

• Analyze data across raw formats (CSV, txt, JSON, etc.), processed file formats (parquet, Delta Lake, ORC, etc.), and SQL tabular data files against Spark and SQL.

• Be productive with enhanced authoring capabilities and built-in data visualization

Synapse team brought the new notebooks component into Synapse Studio to provide consistent notebook experience for Microsoft customers and maximize discoverability, productivity, sharing, and collaboration.

https://docs.microsoft.com/en-us/azure/synapse-analytics/spark/apache-spark-development-using-notebooks?tabs=classical

**Scenario:** You are working on a project and are preparing to ingest data from a SQL Server database hosted on an on-premises Windows Server.

Which integration runtime is required for Azure Data Factory to ingest data from the on-premises server?

- ○

  None of the listed options.

- ○

  Self-Hosted Integration Runtime
  **(Correct)**

- ○

  Azure Integration Runtime

- ○

  On-demand HDInsight cluster

- ○

  Azure-SSIS Integration Runtime

**Explanation**

A self-hosted integration runtime can run copy activities between a cloud data store and a data store in a private network. It also can dispatch transform activities against compute resources in an on-premises network or an Azure virtual network.

In Data Factory, an activity defines the action to be performed. A linked service defines a target data store or a compute service. An integration runtime provides the bridge between the activity and linked services.

**Self-hosted integration runtime**

A self-hosted integration runtime is capable of:

• Running copy activity between a cloud data stores and a data store in private network.

• Dispatching the following transform activities against compute resources in on-premises or Azure Virtual Network:

  • HDInsight Hive activity (BYOC-Bring Your Own Cluster)

  • HDInsight Pig activity (BYOC)

  • HDInsight MapReduce activity (BYOC)

• HDInsight Spark activity (BYOC)

• HDInsight Streaming activity (BYOC)

• Machine Learning Batch Execution activity

• Machine Learning Update Resource activities

• Stored Procedure activity

• Data Lake Analytics U-SQL activity

• Custom activity (runs on Azure Batch)

• Lookup activity

• Get Metadata activity.

The self-hosted integration runtime is logically registered to the Azure Data Factory and the compute resource used to support its functionality as provided by you. Therefore there is no explicit location property for self-hosted IR. When used to perform data movement, the self-hosted IR extracts data from the source and writes into the destination.

https://docs.microsoft.com/en-us/azure/data-factory/create-self-hosted-integration-runtime

While Agile, CI/CD, and DevOps are different, they support one another.

Which is best described by:

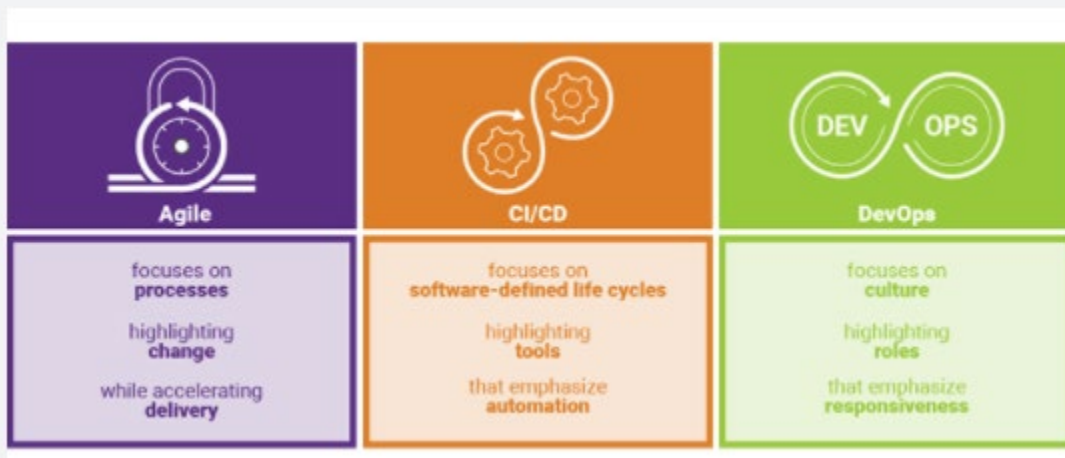*"Focuses on culture highlighting roles that emphasize."*

- ○

  SDLC

- ○

  DevOps
     **(Correct)**

- ○

  CI/CD

- ○

  Agile

**Explanation**
While Agile, CI/CD, and DevOps are different, they support one another. Agile focuses on the development process, CI/CD on practices, and DevOps on culture.



• **Agile** focuses on processes highlighting change while accelerating delivery.

• **CI/CD** focuses on software-defined life cycles highlighting tools that emphasize automation.

• **DevOps** focuses on culture highlighting roles that emphasize responsiveness.

https://www.synopsys.com/blogs/software-security/agile-cicd-devops-difference/

Azure DevOps is a collection of services that provide an end-to-end solution for the five core practices of DevOps: planning and tracking, development, build and test, delivery, and monitoring and operations.

It is possible to put an Azure Databricks Notebook under Version Control in an Azure Devops repo. Using Azure DevOps, you can then build Deployment pipelines to manage your release process.

**CI/CD with Azure DevOps**

Here are some of the features that make it well-suited to CI/CD with Azure Databricks.

• Integrated Git repositories

• Integration with other Azure services

• Automatic virtual machine management for testing builds

• Secure deployment

• Friendly GUI that generates (and accepts) various scripted files

**But what is CI/CD?**

**Continuous Integration**

Throughout the development cycle, developers commit code changes locally as they work on new features, bug fixes, etc. If the developers practice continuous integration, they merge their changes back to the main branch as often as possible. Each merge into the master branch triggers a build and automated tests that validate the code changes to ensure successful integration with other incoming changes. This process avoids integration headaches that frequently happen when people wait until the release day before they merge all their changes into the release branch.

**Continuous Delivery**

Continuous delivery builds on top of continuous integration to ensure you can successfully release new changes in a fast and consistent way. This is because, in addition to the automated builds and testing provided by continuous integration, the release process is automated to the point where you can deploy your application with the click of a button.

**Continuous Deployment**

Continuous deployment takes continuous delivery a step further by automatically deploying your application without human intervention. This means that merged changes pass through all stages of your production pipeline and, unless any of the tests fail, automatically release to production in a fully automated manner.

**Who benefits?**

*Everyone.* Once properly configured, automated testing and deployment can free up your engineering team and enable your data team to push their changes into production. For example:

• Data engineers can easily deploy changes to generate new tables for BI analysts.

• Data scientists can update models being used in production.

• Data analysts can modify scripts being used to generate dashboards.

In short, changes made to a Databricks notebook can be pushed to production with a simple mouse click (and then any amount of oversight that your DevOps team feels is appropriate).

https://docs.microsoft.com/en-us/azure/devops/user-guide/alm-devops-features?view=azure-devops

Identify the missing word(s) in the following sentence within the context of Microsoft Azure.

Azure Synapse SQL enables you to implement data warehouse solutions, or to perform data virtualization.

[?] enables ad hoc data preparation scenarios, where organizations are wanting to unlock insights from their own data stores without going through the formal processes of setting up a data warehouse.

- ○
  Data virtualization
  **(Correct)**

- ○
  Synapse pipelines

- ○
  The ETL process

- ○
  Synapse SQL

**Explanation**
**Azure Synapse SQL enables you to implement data warehouse solutions, or to perform data virtualization.**

A data warehouse is a core component of Business Intelligence (BI) solutions that provides a central repository of data stored in relational tables. It facilitates solutions around descriptive analytics. The data is retrieved, cleansed, and transformed from a range of source data system, and is then served in a structured relational format commonly referred to as a star schema.
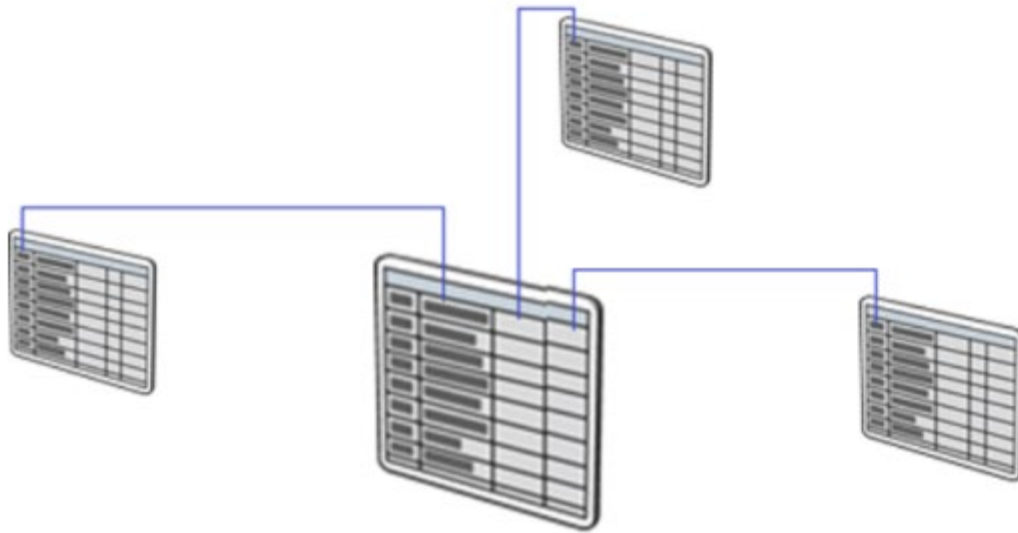
Data in a data warehouse is stored in permanent tables that are populated using an extract, transform, and load (ETL) process by services such as Azure Synapse pipelines, or Azure Data Factory. As a result, you need to understand the data that is stored in the sources systems, how it should arrive within the data warehouse, which in turn dictates how you should cleanse or transform the data.

Data virtualization allows you to interact with data without the need to understand how the data is formatted, structured, or what is its data type. It enables you to explore the data without understanding the technical specifications of the source data, which can be very helpful when performing diagnostic analytics where the need to access data in a timely manner to answer a question is more important.

**Data virtualization also enables ad hoc data preparation scenarios, where organizations are wanting to unlock insights from their own data stores without going through the formal processes of setting up a data warehouse.** You can extract data from a source system in a raw format and loading it into a data lake. From here, transformations may be applied to present the data as required. As the most complex part of the extract, load, and transform (ELT) process is at the end, it means that the access to the data is much quicker.

To meet the delivery of these types of solutions, Azure Synapse SQL offers both a dedicated and serverless model of the service to meet the different demands of both solutions.

The dedicated model is referred to as dedicated SQL Pools. It refers to the data warehousing features that are generally available in Azure Synapse Analytics. Dedicated SQL pools represent a collection of analytic resources that are being provisioned when using Synapse SQL. When you need predictable performance and cost, creating dedicated SQL pools to reserve processing power for data permanently stored in SQL tables in a data warehouse house is the best approach to take.

The serverless model is ideal for unplanned or ad hoc workloads that the diagnostic analytics approach would generate. Therefore, if you are performing data exploration, are preparing data for data virtualization, then SQL serverless would be the better model to use.

https://azure.microsoft.com/en-us/services/synapse-analytics/

**Question 34:**

When you want to switch to SparkSQL in a notebook, what is the first command to type?

- ○

  `%%spark`

- ○

  `%%pyspark`

- ○

  `%%csharp`

- ○

  `%%sql`

  **(Correct)**

- ○

  `%%sparksql`

**Explanation**

You can use multiple languages in one notebook by specifying the correct language magic command at the beginning of a cell. The following table lists the magic commands to switch cell languages.

| Magic command | Language | Description |
|---|---|---|
| %%pyspark | Python | Execute a **Python** query against Spark Context. |
| %%spark | Scala | Execute a **Scala** query against Spark Context. |
| %%sql | SparkSQL | Execute a **SparkSQL** query against Spark Context. |
| %%csharp | .NET for Spark C# | Execute a **.NET for Spark C#** query against Spark Context. |

When you want to switch to SparkSQL in a notebook, type the %%sql command.

https://docs.microsoft.com/en-us/azure/synapse-analytics/spark/apache-spark-development-using-notebooks?tabs=classical

Azure Storage provides a REST API to work with the containers and data stored in each account. Client libraries can save a significant amount of work for app developers because the API is tested and it often provides nicer wrappers around the data models sent and received by the REST API. Microsoft has Azure client libraries that support a number of languages and frameworks.

Which are Azure Storage supported languages and frameworks? (Select all that apply)

- ☐
  Go
  **(Correct)**

- ☐
  Java
  **(Correct)**

- ☐
  C#
  **(Correct)**

- ☐
  Node.js
  **(Correct)**

- ☐
  .NET
  **(Correct)**

- ☐
  Python
  **(Correct)**

**Explanation**
Azure Storage provides a REST API to work with the containers and data stored in each account. There are independent APIs available to work with each type of data you can store. We have four specific data types:

• **Blobs** for unstructured data such as binary and text files.

• **Queues** for persistent messaging.

• **Tables** for structured storage of key/values.

• **Files** for traditional SMB file shares.

**Use a client library**

Client libraries can save a significant amount of work for app developers because the API is tested and it often provides nicer wrappers around the data models sent and received by the REST API.

**Microsoft has Azure client libraries that support a number of languages and frameworks, including:**

• **.NET**

• **Java**

• **Python**

• **Node.js**

• **Go**

• **C#**



For example, to retrieve the same list of blobs in C#, we could use the following code snippet:

```csharp
C#

string containerName = "...";

BlobContainerClient container = new BlobContainerClient(connectionString, containerName);


var blobs = container.GetBlobs();

foreach (var blob in blobs)

{

Console.WriteLine($"{blob.Name} --> Created On: {blob.Properties.CreatedOn:YYYY-MM-dd HH:mm:ss} Size: {blob.Properties.ContentLength}");

}
```

https://docs.microsoft.com/en-us/azure/storage/common/storage-introduction

Identify the missing word(s) in the following sentence within the context of Microsoft Azure.

Azure Databricks (ADB) is a Big Data analytics service. Being a Cloud Optimized managed PaaS offering, it is designed to hide the underlying distributed systems and networking complexity as much as possible from the end user.

Multiple clusters can exist within a workspace, and there's a one-to-many mapping between a Subscription to Workspaces, and further, from one Workspace to multiple Clusters.

Azure Databricks is a multitenant service and to provide fair resource sharing to all regional customers, it imposes limits on API calls. These limits are expressed at the Workspace level and are due to internal ADB components.

Key workspace limits are:

• The maximum number of jobs that a workspace can create in an hour is [A]

• At any time, you cannot have more than [B] jobs simultaneously running in a workspace

• There can be a maximum of [C] notebooks or execution contexts attached to a cluster

• There can be a maximum of [D] Azure Databricks API calls/hour

- ○
  [A] 500, [B] 100, [C] 250, [D] 1000

- ○
  [A] 1000, [B] 150, [C] 150, [D] 1500
  **(Correct)**

- ○
  [A] 250, [B] 50, [C] 200, [D] 500

- ○
  [A] 750, [B] 250, [C] 300, [D] 1250

**Explanation**
Azure Databricks (ADB) is a Big Data analytics service. Being a Cloud Optimized managed PaaS offering, it is designed to hide the underlying distributed systems and networking complexity as much as possible from the end user. It is backed by a team of support staff who monitor its health, debug tickets filed via Azure, etc. This allows ADB users to focus on developing value generating apps rather than stressing over infrastructure management.

In this scenario, you are a data engineer who has been tasked with re-evaluating your organization's Azure Databricks environment due to a high volume of growth, which highlighted some weaknesses in your current configuration. A part of this strategy is evaluating the need for separating development, staging, and production Azure Databricks environments to contend with capacity limits. Automation is key when you need to deploy multiple instances of an environment.

You can deploy Azure Databricks using Azure portal or using Azure Resource Manager templates. One successful ADB deployment produces exactly one Workspace, a space where users can log in and author analytics apps. It comprises the file browser, notebooks, tables, clusters, DBFS storage, etc. More importantly, Workspace is a fundamental isolation unit in Databricks. All workspaces are isolated from each other.

Each workspace is identified by a globally unique 53-bit number, called **Workspace ID or Organization ID**. The URL that a customer sees after logging in always uniquely identifies the workspace they are using:

```
https://regionName.azuredatabricks.net/?o=workspaceId
```

Example: `https://eastus2.azuredatabricks.net/?o=12345`

Azure Databricks uses Azure Active Directory (AAD) as the exclusive Identity Provider and there's a seamless out of the box integration between them. This makes ADB tightly integrated with Azure just like its other core services. Any AAD member assigned to the Owner or Contributor role can deploy Databricks and is automatically added to the ADB members list upon first login. If a user is not a member of the Active Directory tenant, they can't log in to the workspace.

Azure Databricks comes with its own user management interface. You can create users and groups in a workspace, assign them certain privileges, etc. While users in AAD are equivalent to Databricks users, by default AAD roles have no relationship with groups created inside ADB, unless you use SCIM for provisioning users and groups. With SCIM, you can import both groups and users from AAD into Azure Databricks, and the synchronization is automatic after the initial import. ADB also has a special group called **Admins**, not to be confused with AAD's role Admin.

The first user to login and initialize the workspace is the workspace **owner**, and they are automatically assigned to the Databricks admin group. This person can invite other users to the workspace, add them as admins, create groups, etc. The ADB logged in user's identity is provided by AAD, and shows up under the user menu in Workspace:

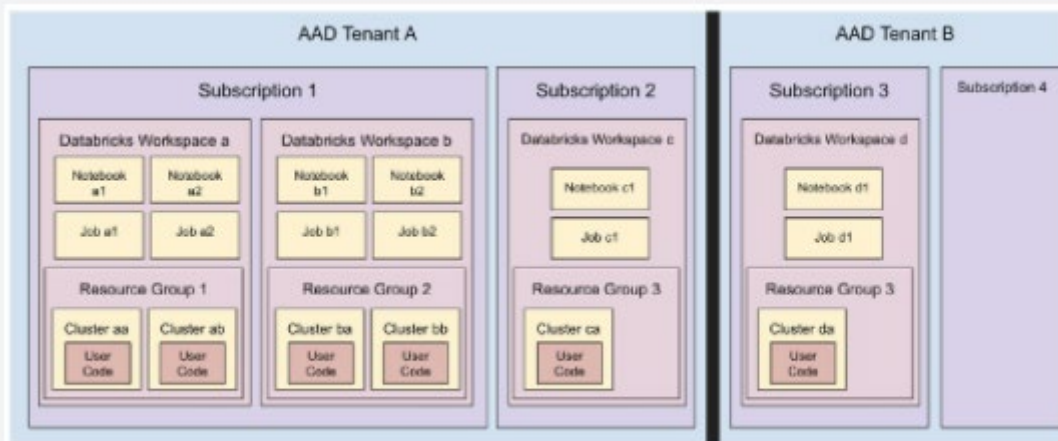Signed in as
Tony@StarkIndustires.com

User Settings

Admin Console

Manage Account

Log Out

Multiple clusters can exist within a workspace, and there's a one-to-many mapping between a Subscription to Workspaces, and further, from one Workspace to multiple Clusters.



With this basic understanding, let's discuss how to plan a typical ADB deployment. We first grapple with the issue of how to divide workspaces and assign them to users and teams.

**Map workspaces to business divisions**

How many workspaces do you need to deploy? The answer to this question depends a lot on your organization's structure. We recommend that you assign workspaces based on a related group of people working together collaboratively. This also helps in streamlining your access control matrix within your workspace (folders, notebooks etc.) and also across all your resources that the workspace interacts with (storage, related

data stores like Azure SQL DB, Azure SQL DW etc.). This type of division scheme is also known as the Business Unit Subscription design pattern and it aligns well with the Databricks chargeback model.
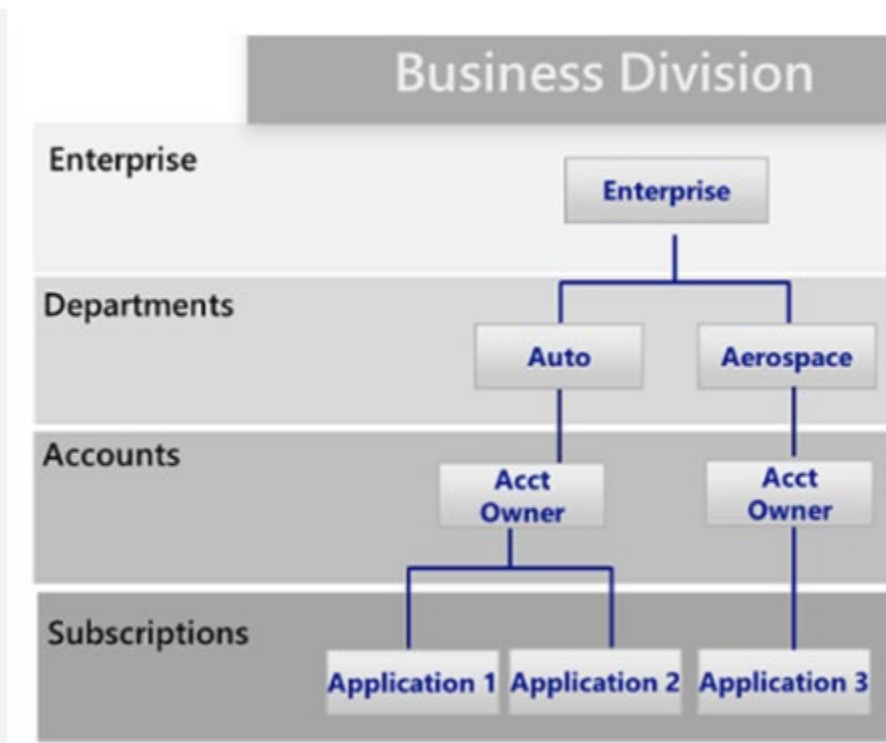


**Deploy workspaces in multiple subscriptions to honour Azure capacity limits**

Customers commonly partition workspaces based on teams or departments and arrive at that division naturally. But it is also important to partition keeping Azure Subscription and ADB Workspace limits in mind.

**Databricks workspace limits**

Azure Databricks is a multitenant service and to provide fair resource sharing to all regional customers, it imposes limits on API calls. These limits are expressed at the Workspace level and are due to internal ADB components. For instance, you can only run up to 150 concurrent jobs in a workspace. Beyond that, ADB will deny your job submissions. There are also other limits such as max hourly job submissions, max notebooks, etc.

Key workspace limits are:

• The maximum number of jobs that a workspace can create in an hour is **1000**

• At any time, you cannot have more than **150 jobs** simultaneously running in a workspace

• There can be a maximum of **150 notebooks or execution contexts** attached to a cluster

• There can be a maximum of **1500** Azure Databricks API calls/hour

**Azure subscription limits**

Next, there are Azure limits to consider since ADB deployments are built on top of the Azure infrastructure.

Key Azure limits are:

• Storage accounts per region per subscription: **250**

• Maximum egress for general-purpose v2 and Blob storage accounts (all regions): **50 Gbps**

• Virtual Machines (VMs) per subscription per region: **25,000**

• Resource groups per subscription: **980**

These limits are at this point in time and might change going forward. Some of them can also be increased if needed. For more help in understanding the impact of these limits or options of increasing them, please contact Microsoft or Databricks technical architects.

***Due to scalability reasons, MS highly recommend separating the production and dev/stage environments into separate subscriptions.***

**High availability / Disaster recovery (HA/DR)**

Within each subscription, consider the following best practices for HA/DR:

• Deploy Azure Databricks in two paired Azure regions, ideally mapped to different control plane regions.

    • For example, East US2 and West US2 will map to different control planes

    • Whereas West and North Europe will map to same control plane

• Use Azure Traffic Manager to load balance and distribute API requests between two deployments, when the platform is primarily being used in a backend non-interactive mode.

**Additional considerations**

• Create different workspaces by different department / business team / data tier, and per environment (development, staging, and production) - across relevant Azure subscriptions

• Define workspace level tags which propagate to initially provisioned resources in managed resource group (Tags could also propagate from parent resource group)

• Use Azure Resource Manager templates templates (search "databricks") to have a more managed way of deploying the workspaces - whether via CLI, PowerShell, or some SDK

• Create relevant groups of users - using Group REST API or by using Azure Active Directory Group Sync with SCIM

https://docs.microsoft.com/en-us/azure/databricks/scenarios/what-is-azure-databricks

**Scenario**: You are working on a project and there are two video files stored as blobs.

**Video 1**: business-critical; requires a replication policy that creates multiple copies across geographically diverse datacentres

**Video 2**: non-business-critical; local replication policy is sufficient

Which of the following options would satisfy both data diversity and cost sensitivity consideration?

- ○

  None of the listed options.

- ○

  Create a single storage account that makes use of Local-redundant storage (LRS) and host both videos from here.

- ○

  Create a single storage account that makes use of Geo-redundant storage (GRS) and host both videos from here.

- ○

  Create two storage accounts. The first account makes use of Geo-redundant storage (GRS) and hosts the business-critical video content. The second account makes use of Local-redundant storage (LRS) and hosts the non-critical video content.
  **(Correct)**

**Explanation**
In general, increased diversity means an increased number of storage accounts. A storage account by itself has no financial cost. However, the settings you choose for the account do influence the cost of services in the account. Use multiple storage accounts to reduce costs.

A storage account represents a collection of settings like location, replication strategy, and subscription owner. You need one storage account for every group of settings that you want to apply to your data. The following illustration shows two storage accounts that differ in one setting; that one difference is enough to require separate storage accounts.

| Storage account | Storage account |
|---|---|
| Subscription: Production | Subscription: Production |
| Location: **West US** | Location: **North Europe** |
| Performance: Standard | Performance: Standard |
| Replication: GRS | Replication: GRS |
| Access tier: Hot | Access tier: Hot |
| Secure transfer: Enabled | Secure transfer: Enabled |
| Virtual networks: Disabled | Virtual networks: Disabled |

The number of storage accounts you need is typically determined by your data diversity, cost sensitivity, and tolerance for management overhead.

**Data diversity**

Organizations often generate data that differs in where it is consumed, how sensitive it is, which group pays the bills, etc. Diversity along any of these vectors can lead to multiple storage accounts. Let's consider two examples:

1. Do you have data that is specific to a country or region? If so, you might want to locate it in a data centre in that country for performance or compliance reasons. You will need one storage account for each location.

2. Do you have some data that is proprietary and some for public consumption? If so, you could enable virtual networks for the proprietary data and not for the public data. This will also require separate storage accounts.

In general, increased diversity means an increased number of storage accounts.

https://docs.microsoft.com/en-us/azure/storage/common/storage-account-overview

Azure role-based access control (RBAC) is the authorization system you use to manage access to Azure resources. To grant access, you may assign roles to which of the following top level classifications? (Select four)

- ☐ Workflows

- ☐ Devices

- ☐ Assets

- ☐ Managed identities
  **(Correct)**

- ☐ Orchestrations

- ☐ Groups
  **(Correct)**

- ☐ Users
  **(Correct)**

- ☐ Service principals
  **(Correct)**

- ☐ Attributes

**Explanation**

Azure role-based access control (RBAC) is the authorization system you use to manage access to Azure resources. To grant access, you assign roles to users, groups, service principals, or managed identities at a particular scope.

**Access control (IAM)** is the blade that you use to assign roles to grant access to Azure resources. It's also known as identity and access management and appears in several locations in the Azure portal.

The following shows an example of the Access control (IAM) blade for an Azure Databricks service:
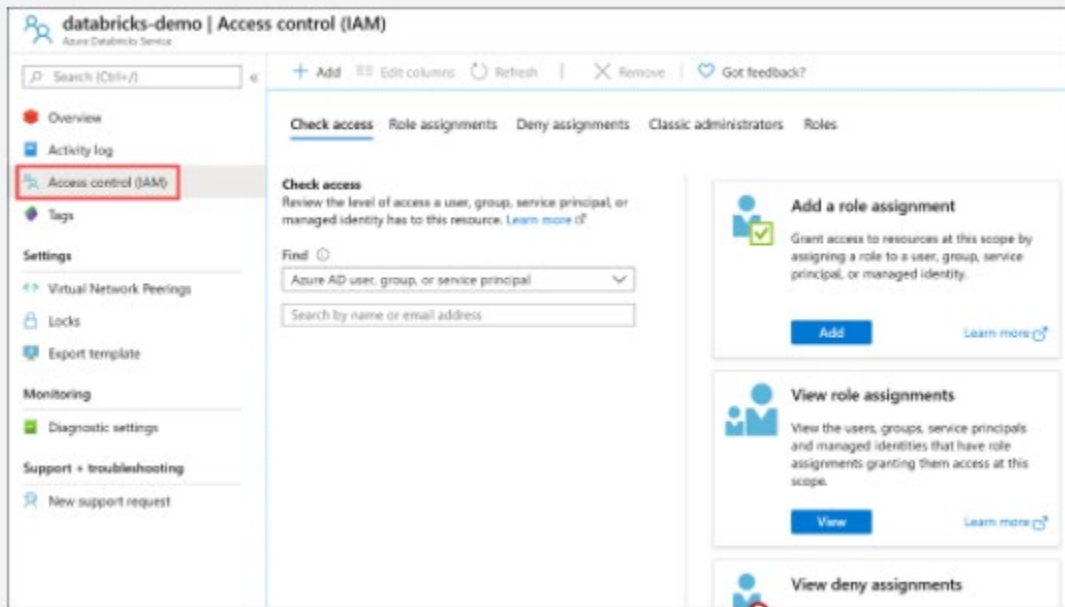
RBAC and IAM are both enabled by Azure Active Directory (Azure AD), an enterprise identity service that provides single sign-on and multi-factor authentication, which helps users securely sign in and access resources in:

• External resources, such as Microsoft 365, the Azure portal, and thousands of other SaaS applications.

• Internal resources, such as apps on your corporate network and intranet, along with any cloud apps developed by your own organization.

**Azure Databricks provides first-party Azure AD integration**
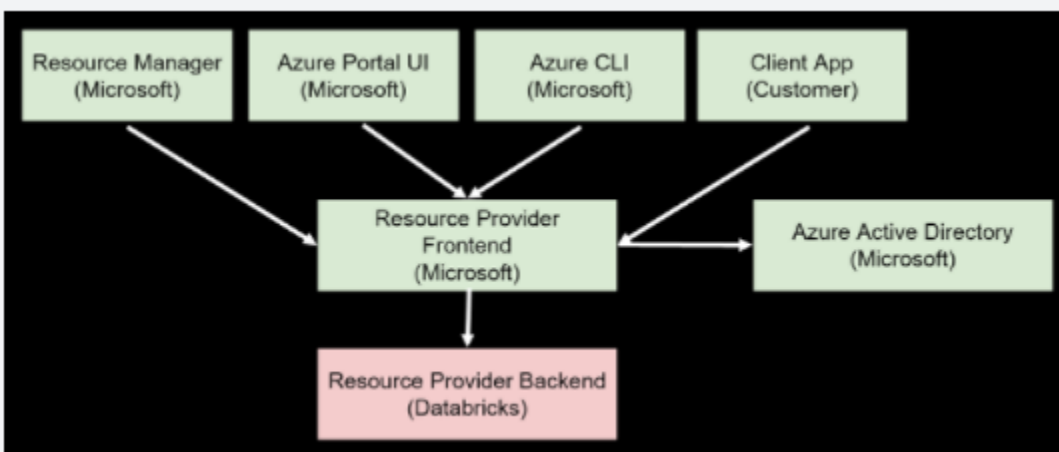
Users access Azure Databricks workspace with an Azure AD account. The Resource Provider Frontend checks a user's authorization against an Azure Active Directory tenant. The user's Azure AD account has to be added to the Azure Databricks workspace before they can access it.

**SCIM integration**

Azure Databricks supports SCIM, or System for Cross-domain Identity Management, an open standard that allows you to automate user provisioning. SCIM lets you use Azure Active Directory to create users in Azure Databricks and give them the proper level of access, as well as remove access for users (deprovision them) when they leave the organization or no longer need access to Azure Databricks.

Customers can sync Azure Active Directory groups with Databricks groups using SCIM functionality. Using groups makes it easy to assign permissions to users in Databricks by applying them to groups instead of individuals if there are a lot of users.

Example attribute mappings:



**Attribute Mappings**
Attribute mappings define how attributes are synchronized between Azure Active Directory and customappsso

| AZURE ACTIVE DIRECTORY ATTRIBUTE | CUSTOMAPPSS... | MATCHING ... | |
|---|---|---|---|
| userPrincipalName | userName | 1 | Delete |
| extensionAttribute1 | id | | Delete |
| mail | emails[type e... | | **Delete** |
| Join(" ", [givenName], [surname]) | displayName | | **Delete** |
| Switch([IsSoftDeleted], , "False", "True", "True", | active | | **Delete** |
| displayName | displayName | 1 | Delete |
| extensionAttribute1 | id | 2 | Delete |
| members | members | | **Delete** |

Add New Mapping

**Conditional access**

Azure Databricks supports Azure Active Directory conditional access, which allows administrators to control where and when users are permitted to sign in to Azure Databricks. For example, conditional access policies can restrict sign-in to your corporate network or can require multi-factor authentication.
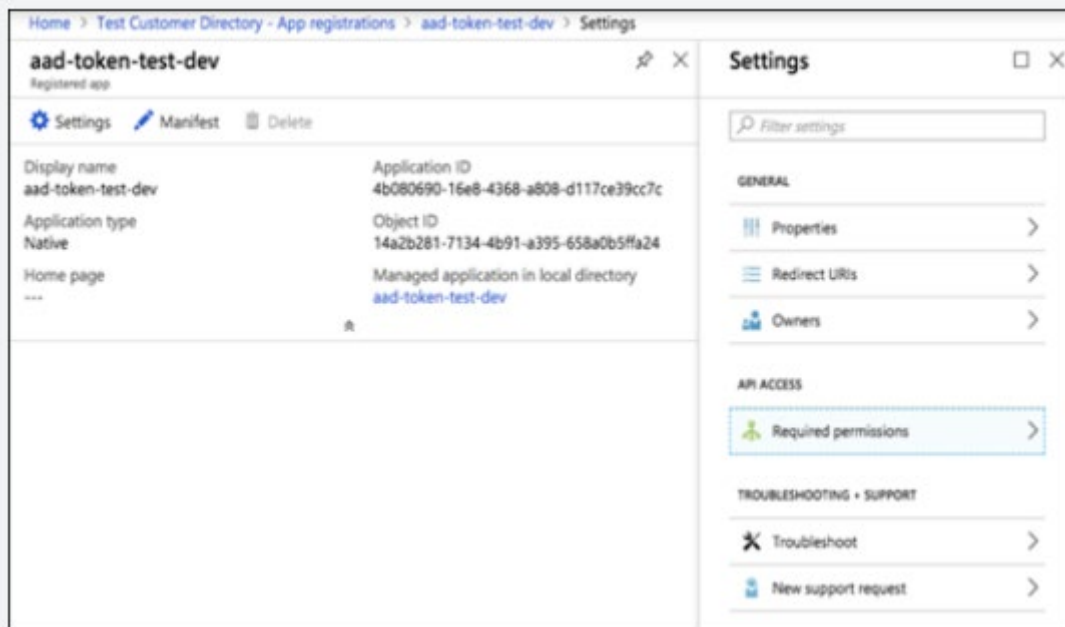
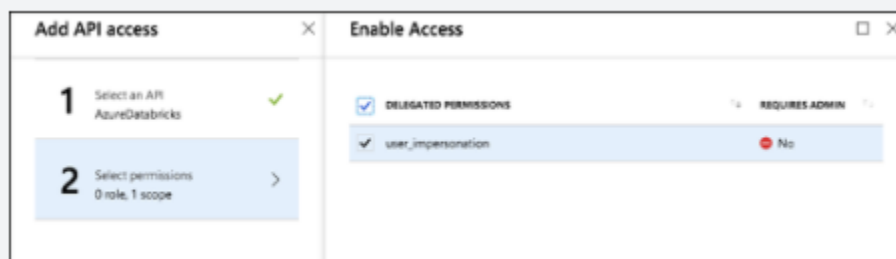This feature is available in the Azure Databricks premium tier only.
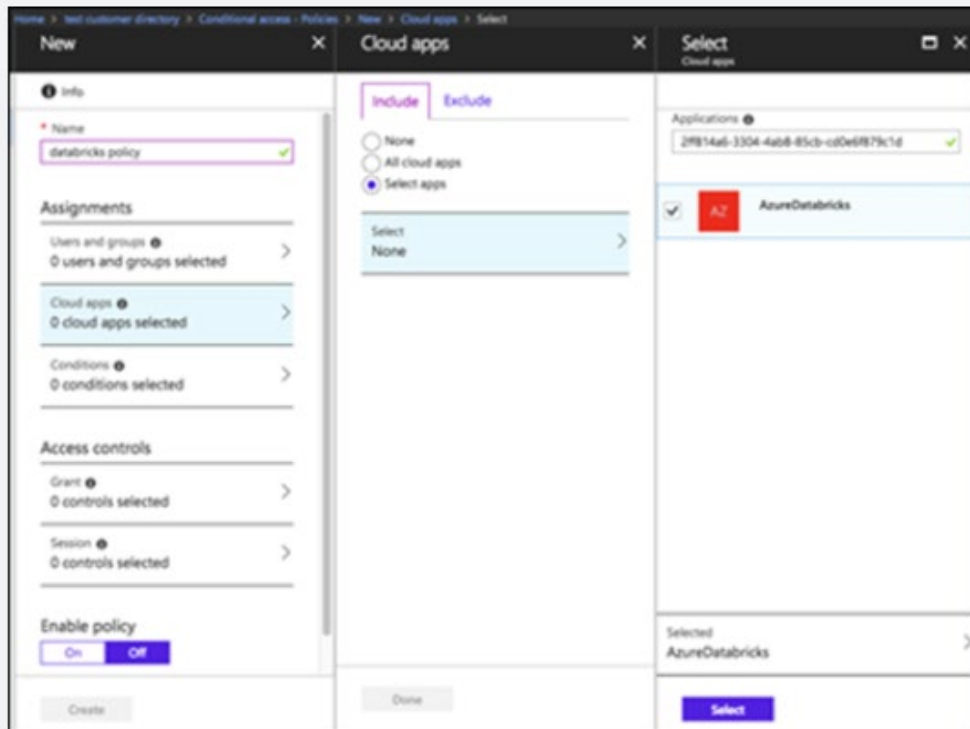


**Azure Active Directory token support**

You can use Azure AD tokens to automate provisioning of Azure Databricks workspaces and access the Databricks REST API. Typically, a user needs to use a personal access token (JWT token) that a they create in the Azure Databricks workspace in order to access the Databricks REST API. The problem with using personal access tokens for the REST API is that your ability to create new tokens exists only after a workspace is created. Since the Azure Databricks workspace creation can only occur within the Azure portal, it is difficult to perform end-to-end automation without using Azure Active Directory tokens. This means that, if you want to automate creating an Azure Databricks workspace, then within that workspace, use the REST API to automate creating users, clusters, jobs, etc., you would first need to create the workspace in the portal, then sign in to the workspace to create the first personal access token.

However, if you want to automate provisioning an Azure Databricks workspace and use the Databricks API in a completely non-interactive way through scripting, you can accomplish this by first creating an enterprise Azure AD application and adding a user to the application and an Azure Databricks workspace. In addition, you will need to enable user impersonation user_impersonation delegated API access in the Azure AD application, as shown in the screenshots below:



API access:



https://docs.microsoft.com/en-us/azure/cloud-adoption-framework/ready/enterprise-scale/identity-and-access-management

Within creating a notebook, you need to specify the pool that needs to be attached to the notebook that is, a SQL or Spark pool.

In order to bring data to a notebook, you have several options.

It is possible to load data from which of the following?

- ☐
  Azure File Storage
  **(Correct)**

- ☐
  Azure Blob Storage
  **(Correct)**

- ☐
  SQL Pool
  **(Correct)**

- ☐
  Azure Cosmos DB
  **(Correct)**

- ☐
  Azure Data Lake Store Gen 2
  **(Correct)**

- ☐
  Azure Data Factory

**Explanation**
In order to bring data to a notebook, you have several options. Currently it is possible to load data from **Azure Blob Storage, Azure Data Lake Store Gen 2, SQL pool as well as other services**. Some examples are:

• Read a CSV from Azure Data Lake Store Gen2 as a Spark DataFrame

• Read a CSV from Azure Blob Storage as a Spark DataFrame

• Read data from the primary storage account

The first possibility is to read a CSV from Azure Data Lake Store Gen2 as a Spark DataFrame. The way you could set it up is as follows:

```Python
from pyspark.sql import SparkSession
```

```
from pyspark.sql.types import *

account_name = "Your account name"

container_name = "Your container name"

relative_path = "Your path"

adls_path = 'abfss://%s@%s.dfs.core.windows.net/%s' % (container_name, account_na
me, relative_path)


spark.conf.set("fs.azure.account.auth.type.%s.dfs.core.windows.net" %account_name
, "SharedKey")

spark.conf.set("fs.azure.account.key.%s.dfs.core.windows.net" %account_name ,"You
r ADLS Gen2 Primary Key")


df1 = spark.read.option('header', 'true') \

.option('delimiter', ',') \

.csv(adls_path + '/Testfile.csv')
```

The variables that you create are:

- `account_nam`e This is your storage account name

- `container_name` This is the name of your storage container

- `relative_path` The relative path of the file

- `adls_path` Will be created by passing through the above parameters.

The second possibility is to read a CSV from Azure Blob Storage as a Spark DataFrame. The way you could set it up is as follows:

```Python
from pyspark.sql import SparkSession

from pyspark.sql.types import *


blob_account_name = "Your blob account name"

blob_container_name = "Your blob container name"

blob_relative_path = "Your blob relative path"

blob_sas_token = "Your blob sas token"
```

```
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)


df = spark.read.option("header", "true") \
.option("delimiter","|") \
.schema(schema) \
.csv(wasbs_path)
```
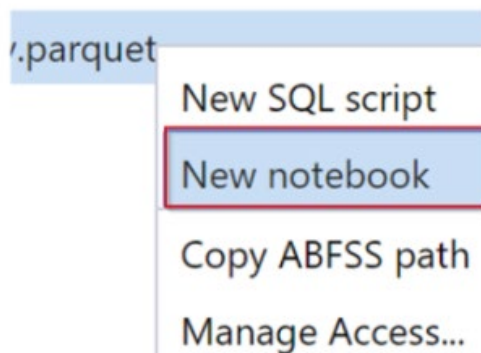
The parameters that it takes into account are:

- `blob_account_name` This is the name of your blob account.

- `blob_container_name` This is the name of the blob container the file is in.

- `blob_relative_path` This is the relative path pointing to the csv you want to read.

- `blob_sas_token` Your blob sas token.

The third possibility is to read data from the primary storage account through using the Data tab in the synapse studio environment. If you right-click on a file and select **New notebook** to see a new notebook with data extractor autogenerated.



https://docs.microsoft.com/en-us/azure/synapse-analytics/spark/apache-spark-development-using-notebooks?tabs=classical

**Scenario:** You are working on a project and have begun creating an Azure Data Lake Storage Gen2 account. Configuration of this account must allow for processing analytical data workloads for best performance.

Which option should you configure when creating the storage account?

- ○

  On the Basic tab, set the Performance option to Standard.

- ○

  On the Basic Tab, set the Performance option to ON.

- ○

  On the Networking tab, set the Hierarchical Namespace to ON.

- ○

  On the Advanced tab, set the Hierarchical Namespace to Enabled.
  **(Correct)**

**Explanation**
If you want to enable the best performance for analytical workloads in Data Lake Storage Gen2, then **on the Advanced tab of the Storage Account creation set the Hierarchical Namespace to Enabled.**

https://docs.microsoft.com/en-us/azure/storage/common/storage-account-create?tabs=azure-portal

**Scenario:** Queen Consolidated was overtaken by Raymond Carson Palmer and re-branded as Palmer Technologies. Now that Ray is overseeing the operations at Palmer, Ray has decided to implement better applications.

You are working as a consultant with Palmer, and in a meeting with Ray and his IT team discussing Azure Data Factory. The team plans to use Azure Data Factory to prepare data to be queried by Azure Synapse Analytics serverless SQL pools.

Files will be initially ingested into an Azure Data Lake Storage Gen2 account as 10 small JSON files. Each file will contain the same data attributes and data from a subsidiary of Palmer. The team needs to move the files to a different folder and transform the data.

**Required:**
• Provide the fastest possible query times.
• Automatically infer the schema from the underlying files.

As the Azure expert, the team looks to you for advice on how they should configure the Data Factory copy activity with respect to the sink file type.

Which of the following should you advise them to use?

- ○ TXT

- ○ Parquet
  **(Correct)**

- ○ JSON

- ○ CSV

**Explanation**

*You should advise Palmer to use Parquet because it supports the schema property; Azure Data Factory parquet format is supported for Azure Data Lake Storage Gen2.*

Parquet format is supported for the following connectors: Amazon S3, Amazon S3 Compatible Storage, Azure Blob, Azure Data Lake Storage Gen1, Azure Data Lake Storage Gen2, Azure File Storage, File System, FTP, Google Cloud Storage, HDFS, HTTP, Oracle Cloud Storage and SFTP.

**Parquet as source**

The following properties are supported in the copy activity *source* section.

| Property | Description | Required |
|---|---|---|
| type | The type property of the copy activity source must be set to **ParquetSource**. | Yes |
| storeSettings | A group of properties on how to read data from a data store. Each file-based connector has its own supported read settings under storeSettings. **See details in connector article -> Copy activity properties section.** | |

**Parquet as sink**

The following properties are supported in the copy activity *sink* section.

| Property | Description | Required |
|---|---|---|
| type | The type property of the copy activity sink must be set to **ParquetSink**. | Yes |
| formatSettings | A group of properties. Refer to **Parquet write settings** table below. | No |
| storeSettings | A group of properties on how to write data to a data store. Each file-based connector has its own supported write settings under storeSettings. **See details in connector article -> Copy activity properties section.** | |

https://docs.microsoft.com/en-us/azure/data-factory/format-parquet

Once you have checked the monitor tab within the Azure Synapse Studio environment, and feel that you could improve the performance of the run, you have several things to consider:

• Choose the data abstraction

• Use the optimal data format

• Use the cache option

• Check the memory efficiency

• Use Bucketing

• Optimize Joins and Shuffles if appropriate

• Optimize Job Execution

If you did decide to use bucketed tables, which of the following are recommended practices? (Select all that apply)

- ☐
  The order of the different type of joins does matter when it comes to resource consumption.
  **(Correct)**

- ☐
  It's advised to start with the broadest selective joins.

- ☐
  Move joins that increase the number of rows after aggregations.
  **(Correct)**

- ☐
  Avoid the use of SortMerge join when possible.
  **(Correct)**

**Explanation**
Once you have checked the monitor tab within the Azure Synapse Studio environment, and feel that you could improve the performance of the run, you have several things to take in mind:

• Choose the data abstraction

• Use the optimal data format

• Use the cache option

• Check the memory efficiency

• Use Bucketing

• Optimize Joins and Shuffles if appropriate

• Optimize Job Execution

In order to optimize the Apache Spark Jobs in Azure Synapse Analytics, you need to take into account the cluster configuration for the workload you're running on that cluster. You might run into challenges where memory pressure (if not configured well, like not choosing the right size of executors), long running operations and tasks that might result in Cartesian operations. If you want to speed up the jobs, you'd have to configure the appropriate caching for that task, as well as checking joins and shuffles in relation to data skew. Therefore, it is so imperative that you monitor and review Spark Job executions that are long running or resource-consuming.

Some recommendations in order for you to optimize the Spark Job might include the following:

**Choosing the data abstraction**

Some of the earlier Spark versions use RDDs to abstract the data. Spark 1.3 and 1.6 introduced the use of DataFrames and Datasets. The following relative merits might help you to optimize in relation to your data abstraction:

**DataFrames**

Using DataFrames would be a great place to start. DataFrames provide query optimization through Catalyst. It also includes a whole-stage code generation with direct memory access. One thing to take in mind is that when you want to have the best-developer-friendly experience it might be better to use DataSets, since there are no compile-time checks or domain object programming.

On that note, let's look into DataSets: *DataSets are good in complex ETL pipelines optimization where the performance impact is acceptable. Just be cautious when using DataSets in aggregations, since it might impact the performance. However, it will provide query optimization through Catalyst and is developer-friendly by providing object programming and compile-time checks. DataSets do add serialization/deserialization overhead and has a high GC overhead.

Looking at RDDs we would advise as follows: It is not necessary to use RDDs unless you want or need to build a new custom RDD. However, there is no query optimization through

Catalyst as well as no whole-stage code generation and would still have a high GC overhead. The only way to use RDDs is that it needs SPark 1.x legacy APIs.

When looking at your data format, spark provides many. Formats that you can use are `csv`, `json`, `xml`, `parquet` etc. It can also be extended by other formats with external data sources. A tip that might be useful is using parquet with snappy compression (which also happen to be the default in Spar 2.x.) Why Parquet? It stores data in a columnar format, is compressed and highly optimized in Spark, as well as, splittable in order to decompress.

When it comes to the caching, there is a native built in Spark caching mechanism. It can be used through different methods like: `.persist(), .cache()`, and `CACHE TABLE`. When using small datasets, it might be effective. In ETL pipelines where caching of intermediate results is necessary this might come in handy too. Just take in mind that is you need to do partitioning, the spark native caching mechanism might have some downsides. The reason for that is that a cached table won't keep the partitioning data.

It is also imperative to understand how to use the memory efficiently. What you have to understand is that Spark operates by placing data in memory. Therefore, managing memory resources is an aspect of optimizing Spark jobs executions. The way to manage it, might be to check smaller data partitions and checking data size, types and distributions when you formulate a partitioning strategy. Another way to optimize is to consider Kryo data serialization: Kryo data serialization, versus the default Java serialization. Always bear in mind though, to keep monitoring and tuning the Spark configuration settings.

Another thing to look at might be bucketing.

**Use bucketing**

Bucketing is almost the same as data partitioning. The way it differs is that a bucket holds a set of column values instead of one. It might work well when you partition on large (millions or more) values like product identifiers. A bucket is determined by hashing the bucket key of a row. The way bucketed tables are optimized is because it's because the metadata about how it was bucketed and sorted are stored.

Some advanced bucketing features are:

• Query optimization based on bucketing meta-information.

• Optimized aggregations.

• Optimized joins.

However, bucketing doesn't exclude partitioning. They go hand in had. You can use partitioning and bucketing at the same time.

**Optimize joins and shuffles**

Sometimes, when you have a slower performance on join or shuffle jobs, it can be caused by data skew. What is data skew? It's asymmetry in your job data. An example might be that a job only takes 20 sec regularly, however running the same job where data is joined and shuffled can take up hours. In order to fix that data skew, you can salt the entire key, or use an isolated salt for only some subset of keys. Another option to look into might be the introduction of a bucket column and pre-aggregate in buckets first. However, there's more to causing slow joins, since it might be the join type. Spark uses the `SortMerge` join type. This type of join is best suited for large data sets, but is otherwise computationally expensive because it must first sort the left and right sides of data before merging them. Therefore, a `Broadcast` join might be better suited for smaller data sets, or where one side of the join is much smaller than the other side.

You can change the join type in your configuration by setting `spark.sql.autoBroadcastJoinThreshold,` or you can set a join hint using the DataFrame APIs `(dataframe.join(broadcast(df2)))`.

```scala
Scala
// Option 1
spark.conf.set("spark.sql.autoBroadcastJoinThreshold", 1*1024*1024*1024)


// Option 2
val df1 = spark.table("FactTableA")
val df2 = spark.table("dimMP")
df1.join(broadcast(df2), Seq("PK")).
createOrReplaceTempView("V_JOIN")


sql("SELECT col1, col2 FROM V_JOIN")
```

If you did decide to use bucketed tables, you will have a third join type, the Merge join. A correctly pre-partitioned and pre-sorted dataset will skip the expensive sort phase from a `SortMerge` join. Another thing to take in mind is that the order of the different type of joins does matter, especially in complex queries. Therefore, it's advised to start with the most selective joins. In addition, try to move joins that increase the number of rows after aggregations when possible.

Looking at the sizing of executors in order to increase performance in your spark job, you could consider the Java garbage Collection Overhead (GC) overhead.

• Factors to reduce executor size:

• Reduce heap size below 32 GB to keep GC overhead < 10%.

• Reduce the number of cores to keep GC overhead < 10%.

• Factors to increase executor size:

• Reduce communication overhead between executors.

• Reduce the number of open connections between executors (N2) on larger clusters (>100 executors).

• Increase heap size to accommodate for memory-intensive tasks.

• Optional: Reduce per-executor memory overhead.

• Optional: Increase utilization and concurrency by oversubscribing CPU.

As a general rule of thumb when selecting the executor size:

• Start with 30 GB per executor and distribute available machine cores.

• Increase the number of executor cores for larger clusters (> 100 executors).

• Modify size based both on trial runs and on the preceding factors such as GC overhead.

When running concurrent queries, consider as follows:

• Start with 30 GB per executor and all machine cores.

• Create multiple parallel Spark applications by oversubscribing CPU (around 30% latency improvement).

• Distribute queries across parallel applications.

• Modify size based both on trial runs and on the preceding factors such as GC overhead.

As stated before, it's important to keep monitoring the performance, especially outliers, using the timeline view, SQL graph, job statistics etc. It might be a case where one of the executors is slower than the other, which most frequently happens on large clusters (30+

nodes). What you then might consider is to divide the work into more tasks such that the scheduler can compensate for the slower tasks.

If there is an optimization necessary in relation to the optimization of a job execution, make sure you keep in mind the caching (an example might be using the data twice, but cache it). IF you broadcast variables on all the executors you set up, due to the variables only being serialized once, you'll have faster lookups. In another case you might use the thread pool that runs on the driver, which could result in faster operations for many tasks.

https://docs.microsoft.com/en-us/azure/synapse-analytics/spark/apache-spark-performance

A shuffle occurs when we need to move data from one node to another in order to complete a stage. Depending on the type of transformation, you are doing you may cause a shuffle to occur. This happens when all the executors require seeing all of the data in order to accurately perform the action. If the Job requires a wide transformation, you can expect the job to execute slower because all of the partitions need to be shuffled around in order to complete the job.

There are two control knobs of a shuffle that can used to optimize.

Which are these options? (Select two)

- ☐
  Cap on compute resource allocation

- ☐
  Region allocation

- ☐
  Number of partitions being shuffled
  **(Correct)**

- ☐
  Number of partitions that you can compute in parallel
  **(Correct)**

- ☐
  Minimum data set size

- ☐
  Number containers that can be accessed by a specific partition

- ☐
  Maximum data set size

**Explanation**
**Tune shuffle for optimal performance**

A shuffle occurs when we need to move data from one node to another in order to complete a stage. Depending on the type of transformation, you are doing you may cause a shuffle to occur. This happens when all the executors require seeing all of the data in order to accurately perform the action. If the Job requires a wide transformation, you can expect the job to execute slower because all of the partitions need to be shuffled around in order to complete the job. For example: Group by, Distinct.
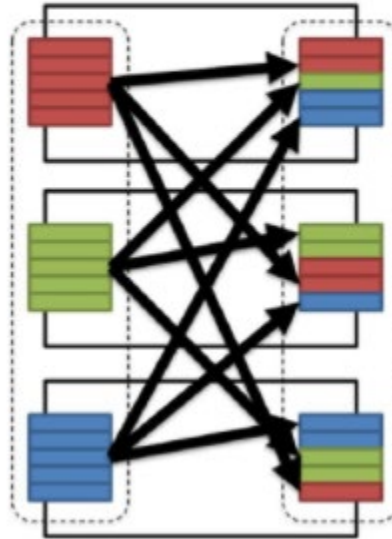
## Narrow transformation
- Input and output stays in same partition
- No data movement is needed
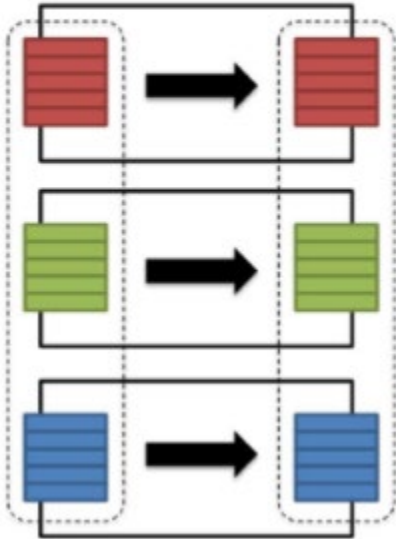
## Wide transformation
- Input from other partitions are required
- Data shuffling is needed before processing

There are two control knobs of a shuffle that can used to optimize:

• The number of partitions being shuffled:

```Python
spark.conf.set("spark.sql.shuffle.partitions", 10)
```

• The number of partitions that you can compute in parallel.

• This is equal to the number of cores in a cluster.

These two determine the partition size, which we recommend should be in the Megabytes to 1-Gigabyte range. If your shuffle partitions are too small, you may be unnecessarily adding more tasks to the stage. But if they are too big, you may get bottlenecked by the network.

**Partition your data**

This is a broad Big Data best practice not limited to Azure Databricks, and we mention it here because it can notably impact the performance of Databricks jobs. Storing data in

partitions allows you to take advantage of partition pruning and data skipping, two important features that can avoid unnecessary data reads. Most of the time partitions will be on a date field but you should choose your partitioning field based on the predicates most often used by your queries. For example, if you're always going to be filtering based on "Region," then consider partitioning your data by region.

• Evenly distributed data across all partitions (date is the most common)

• 10 s of GB per partition (~10 to ~50 GB)

• Small data sets should not be partitioned

• Beware of over partitioning

https://bigdatatn.blogspot.com/2017/05/spark-performance-optimization-shuffle.html

On initial deployment Azure Synapse Analytics, there are a few resources that deploy along with it.

Which of the following are deployed along with Azure Synapse Analytics? (Select all that apply)

- ☐ Azure Synapse Workspace
  **(Correct)**

- ☐ Azure Data Lake Storage Gen2
  **(Correct)**

- ☐ Azure Queue Storage

- ☐ Azure Kubernetes Service

- ☐ Azure Machine Learning

**Explanation**
Azure Synapse Analytics can work by acting as the one stop shop to meet all of your analytical needs in an integrated environment if you do not have an analytical environment in place already.

**Integrate with a variety of Azure Data Platform technologies**

For organizations that have existing analytical solutions, Azure Synapse Analytics can integrate with a wide variety of technologies to complement them. For example, if you are already using Azure Data Factory to build data integration pipelines, these can be used to load data into Azure Synapse Analytics. You can also integrate existing data preparation or data science projects that you may hold in Azure Databricks. Theirs is also integration with many of Azure security components to ensure that you meet security and compliance requirements within your organization. **On initial deployment Azure Synapse Analytics, there are a few resources that deploy along with it, including the Azure Synapse Workspace and an Azure Data Lake Storage Gen2 (ADLS Gen2) account that acts as the primary storage for the workspace.**
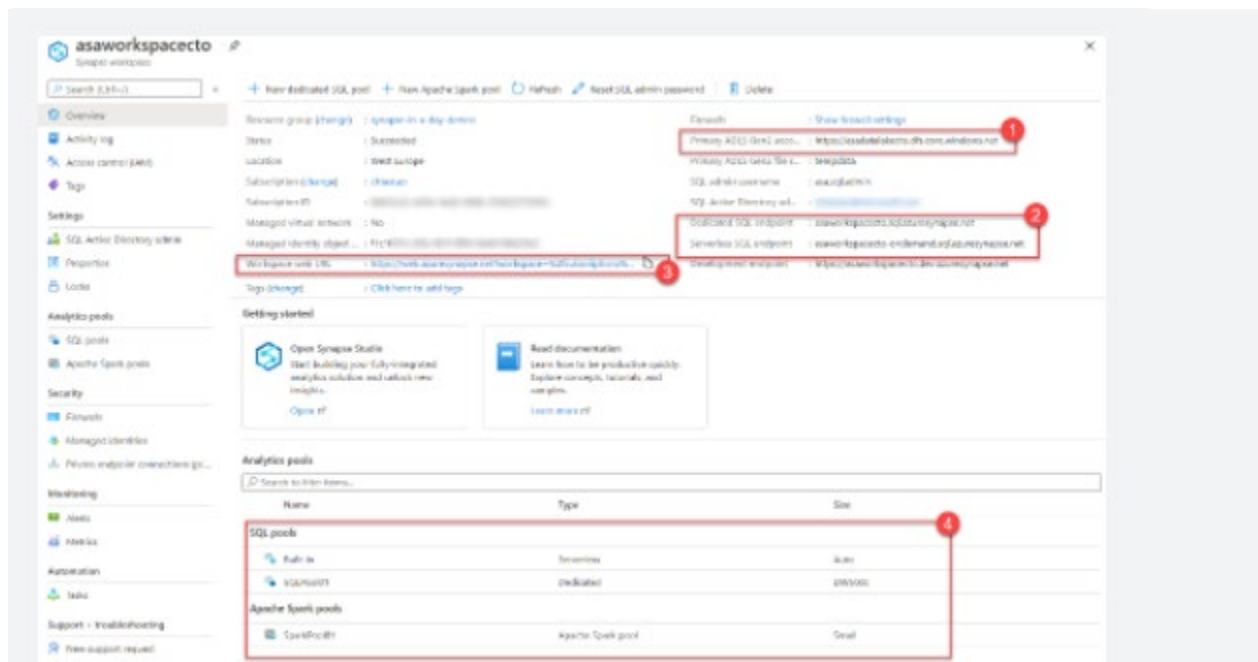
Within the Azure portal, there are links to configure your workspace, manage access through Access control (IAM), firewalls, managed identities, and private endpoint connections, as well as view metrics.

It also contains important information about your Synapse Analytics environment, such as:

• The **Primary ADLS Gen2 account URL (1)**, which identifies the primary data lake storage account.

• The **SQL endpoint** and **SQL on-demand endpoint (2)**, which are used to integrate with external tools, such as SQL Server Management Studio (SSMS), Azure Data Studio, and Power BI.

• The **Workspace web URL (3)**, a direct link to Synapse Studio for the workspace.

• Available resources, such as **SQL pools** and **Apache Spark pools (4)**.

https://docs.microsoft.com/en-us/azure/synapse-analytics/overview-what-is

Wrangling Data Flow is a data flow object that can be added to the canvas designer as an activity in an Azure Data Factory pipeline to perform code free data preparation.

There are two ways to create a wrangling data flow in Azure Data Factory.

• Click the plus icon and select Data Flow in the factory resources pane.

• In the activities pane of the pipeline canvas, open the Move and Transform accordion and drag the Data flow activity onto the canvas.

In both methods, in the side pane that opens, select Create new data flow and choose Wrangling data flow. Click OK.

Once you have selected a source, then clicked on create, what is the result?

- ○
  This opens the Online Mashup Editor.
  **(Correct)**

- ○
  This both opens the Data Flow Wrangler UI and creates a new instance of Data Factory which can be manipulated in either a CLI or GUI environment.

- ○
  None of the listed options.

- ○
  This creates a new instance of Data Factory which can be manipulated in either a CLI or GUI environment.

- ○
  This both opens the Online Mashup Editor and creates a new instance of Data Factory which can be manipulated in either a CLI or GUI environment.

- ○
  This opens the Data Flow Wrangler UI.

**Explanation**
Wrangling Data Flow is a data flow object that can be added to the canvas designer as an activity in an Azure Data Factory pipeline to perform code free data preparation. It enables individuals who are not conversant with the traditional data preparation technologies such as Spark or SQL Server, and languages such as Python and T-SQL to prepare data at cloud scale iteratively.

**There are two ways to create a wrangling data flow in Azure Data Factory.**

• Click the plus icon and select Data Flow in the factory resources pane.

• In the activities pane of the pipeline canvas, open the Move and Transform accordion and drag the Data flow activity onto the canvas.

In both methods, in the side pane that opens, select Create new data flow and choose Wrangling data flow. Click OK.

**Once you have selected a source, then click on create.**

**This opens the Online Mashup Editor.**
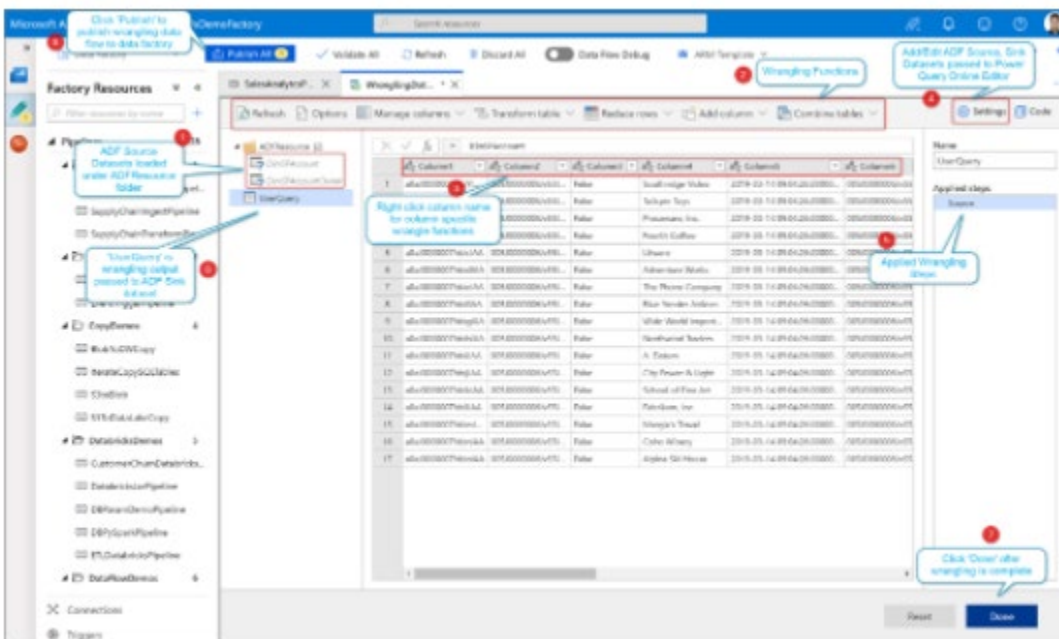


**It consists of the following components:**

1. Dataset list

This will provide the datasets that have been defined as the source for the Data Wrangling.

2. Wrangling Function toolbar

The toolbar contains a variety of data wrangling functions that the user can access to manipulate the data including:

- Managing columns

- Transforming tables

- Reducing rows

- Adding columns

- Combining tables

Each item is context-sensitive and contains sub functions specific to it.

3. Column headings

As well as having the ability to rename columns, right-clicking the column will bring up context-sensitive items for managing columns.

4. Settings

This enables you to add or edit data sources and data sinks, and modify setting for the wrangling data task.

5. Steps window

This window shows the steps that have been applied to the wrangling output. In the example in the graphic, the step named "Source" has been applied the wrangling output named "UserQuery".

6. Wrangling output list

Lists the data wrangling output that has been defined.

7. Done

Completes the Data Wrangling task work.

8. Publish button

Enables you to publish the work that has been created. A wrangling data flow task appears in the canvass designer just like a Copy Activity task, or a Mapping Data Flow task and can be managed and monitored in the same way.

https://docs.microsoft.com/en-us/azure/data-factory/wrangling-overview