Question 1: **Correct**

Scenario: We are working on a project which has a pipeline with two activities where Activity2 has a failure dependency on Activity1.

What will the result be of the pipeline?

- **This pipeline reports success.**

  (Correct)

- **This pipeline reports failure.**

- **This pipeline reports skipped.**

- **This pipeline reports completed.**

Explanation

If we have a pipeline with two activities where Activity2 has a failure dependency on Activity1, the pipeline will not fail just because Activity1 failed. If Activity1 fails and Activity2 succeeds, the pipeline will succeed. This scenario is treated as a try-catch block by Data Factory.

Azure Data Factory

In order to work with data factory pipelines, it is imperative to understand what a pipeline in Azure Data Factory is.

A pipeline in Azure Data Factory represents a logical grouping of activities where the activities together perform a certain task.

An example of a combination of activities in one pipeline can be, ingesting and cleaning log data in combination with a mapping data flow that analyzes the log data that has been cleaned.

A pipeline enables you to manage the separate individual activities as a set, which would otherwise be managed individually. It enables you to deploy and schedule the activities efficiently, through the use of a single pipeline, versus managing each activity independently.

Activities in a pipeline are referred to as actions that you perform on your data. An activity can take zero or more input datasets and produce one or more output datasets.

An example of an action can be the use of a copy activity, where you copy data from an Azure SQL Database to an Azure DataLake Storage Gen2. To build on this example, you can use a data flow activity or an Azure Databricks Notebook activity for processing and transforming the data that was copied to your Azure Data Lake Storage Gen2 account, in order to have the data ready for business intelligence reporting solutions like in Azure Synapse Analytics.

Since there are many activities that are possible in a pipeline in Azure Data Factory, we have grouped the activities in three categories:

• *Data movement activities*: the Copy Activity in Data Factory copies data from a source data store to a sink data store.

• *Data transformation activities*: Azure Data Factory supports transformation activities such as Data Flow, Azure Function, Spark, and others that can be added to pipelines either individually or chained with another activity.

• *Control activities*: Examples of control flow activities are 'get metadata', 'For Each', and 'Execute Pipeline'.

Activities can depend on each other. What we mean, is that the activity dependency defines how subsequent activities depend on previous activities. The dependency itself can be based on a condition of whether to continue in the execution of previous defined activities in order to complete a task. An activity that depends on one or more previous activities, can have different dependency conditions.

The four dependency conditions are:

• Succeeded

• Failed

• Skipped

• Completed

For example, if a pipeline has an Activity A, followed by an Activity B and Activity B has as a dependency condition on Activity A 'Succeeded', then Activity B will only run if Activity A has the status of succeeded.

If you have multiple activities in a pipeline and subsequent activities are not dependent on previous activities, the activities may run in parallel.

https://datasavvy.me/2021/02/18/azure-data-factory-activity-failures-and-pipeline-outcomes/

Question 2: **Incorrect**
Azure Data Factory is a cloud-based ETL and data integration service that allows you to create data-driven workflows for orchestrating data movement and transforming data at scale. Using Azure Data Factory, you can create and schedule data-driven workflows (called pipelines) that can ingest data from disparate data stores.

True or False: Each data factory has a single dedicated pipeline. When additional pipelines are needed for workloads, additional data factory deployments can be used to create an unlimited number of pipelines.

- False

  (Correct)

-

  True

  (Incorrect)

Explanation
Azure Data Factory is a cloud-based ETL and data integration service that allows you to create data-driven workflows for orchestrating data movement and transforming data at scale. Using Azure Data Factory, you can create and schedule data-driven workflows (called pipelines) that can ingest data from disparate data stores.

A data factory can have one or more pipelines. A pipeline is a logical grouping of activities that together perform a task. For example, a pipeline could contain a set of activities that ingest and clean log data, and then kick off a mapping data flow to analyze the log data. The pipeline allows you to manage the activities as a set instead of each one individually. You deploy and schedule the pipeline instead of the activities independently.

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

Question 3: **Correct**
Scenario: While working on a project using Azure Data Factory, you are routing data rows to different streams based on matching conditions. Which transformation in Mapping Data Flow is used to do this?

-

  Inspect

-

  Select

-

  Lookup

-

  Optimize

-

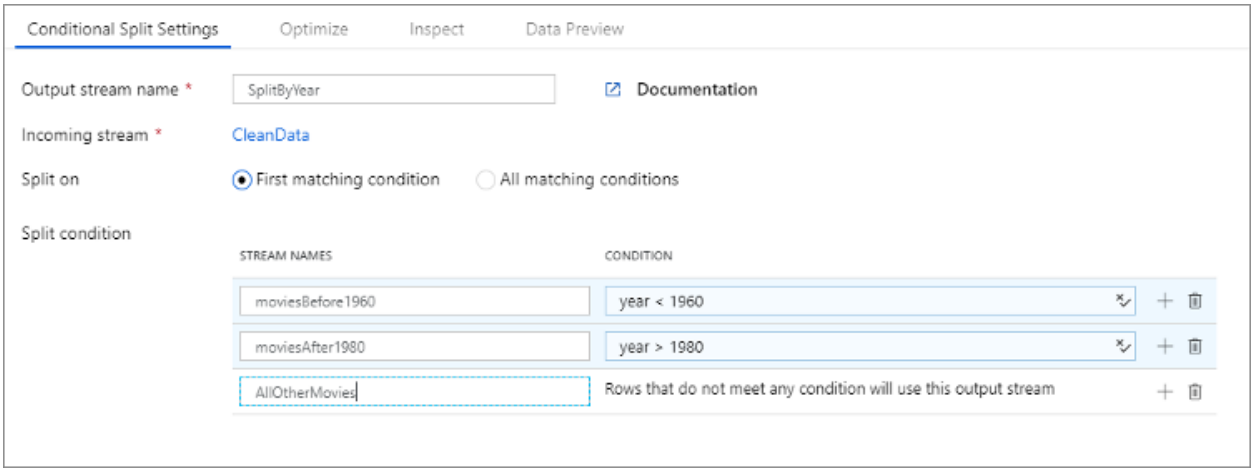
Conditional Split

(Correct)

Explanation
Conditional Split transformation routes data rows to different streams based on matching conditions. The conditional split transformation is similar to a CASE decision structure in a programming language.

The **Split on** setting determines whether the row of data flows to the first matching stream or every stream it matches to.

Use the data flow expression builder to enter an expression for the split condition. To add a new condition, click on the plus icon in an existing row. A default stream can be added as well for rows that don't match any condition.



https://docs.microsoft.com/en-us/azure/data-factory/data-flow-conditional-split

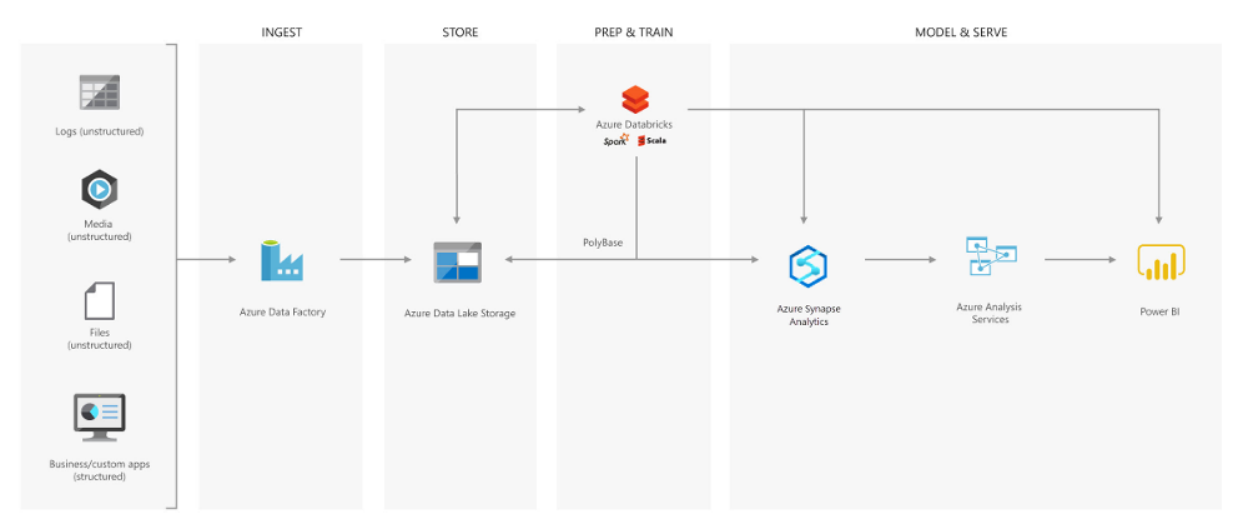
Question 4: **Correct**

Scenario: You are a Data Engineering consultant for a Avengers Security. In the past, they've created an on-premises business intelligence solution that used a Microsoft SQL Server Database Engine, SQL Server Integration Services, SQL Server Analysis Services, and SQL Server Reporting Services to provide historical reports. They tried using the Analysis Services Data Mining component to create a predictive analytics solution to predict the buying behaviour of customers. While this approach worked well with low volumes of data, it couldn't scale after more than a gigabyte of data was collected. Furthermore, they were never able to deal with the JSON data that a third-party application generated when a customer used the feedback module of the point of sale (POS) application.

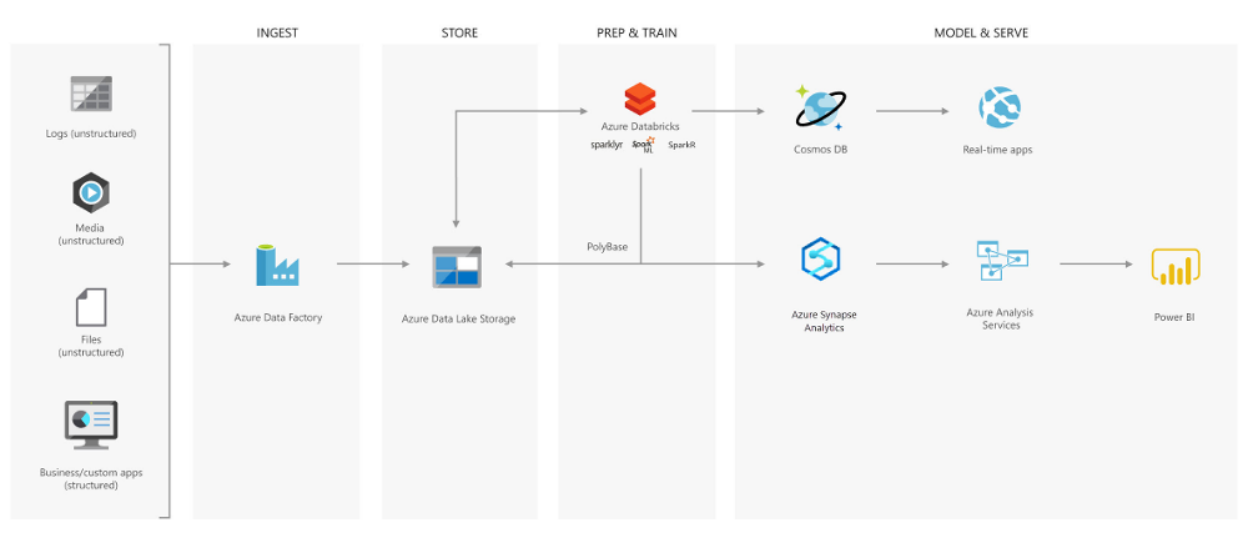
The company has turned to you for help with creating an architecture that can scale with the data needs that are required to create a predictive model and to handle the JSON data so that it's integrated into the BI solution.

Review the following architecture designs.

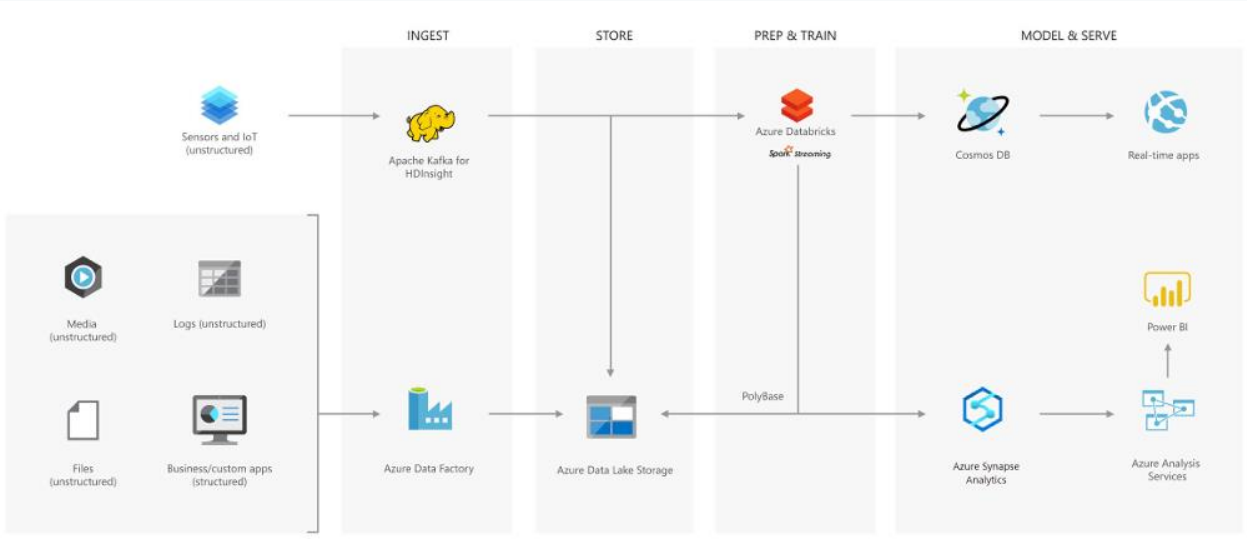
Design A:



Design B:



Design C:



Which architecture would be best suited for the need?

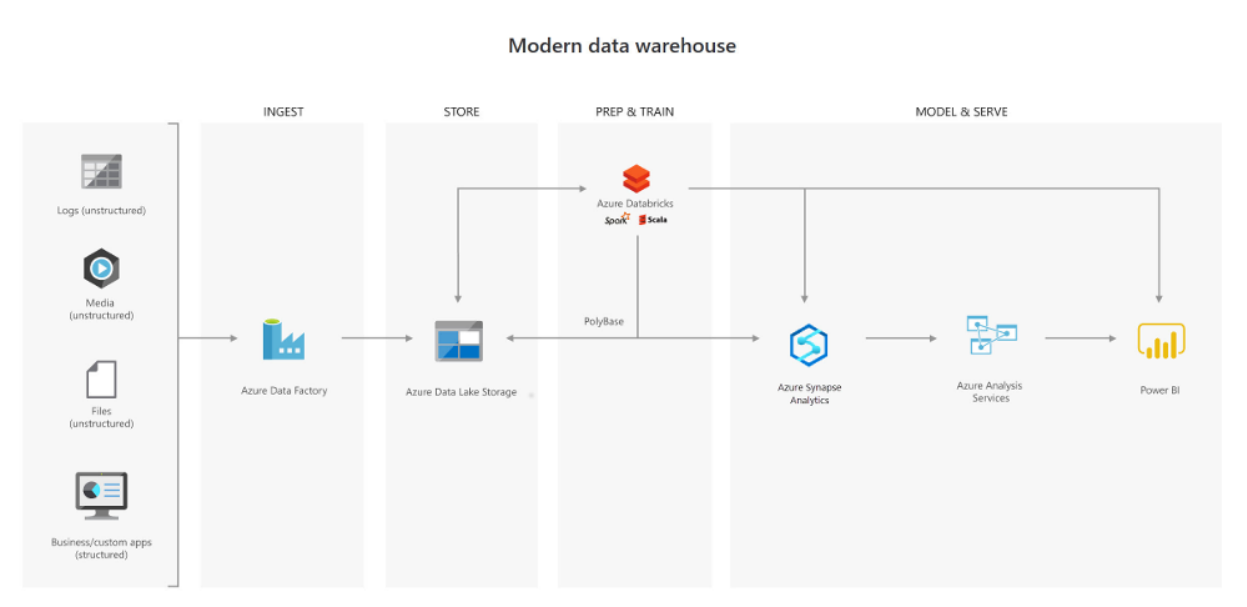
- **Design C**
- **None of the listed options**
- **Design B**
- **Design A**

  (Correct)

Explanation
Creating a modern data warehouse

Imagine you're a Data Engineering consultant for a Avengers Security. In the past, they've created an on-premises business intelligence solution that used a Microsoft SQL Server Database Engine, SQL Server Integration Services, SQL Server Analysis Services, and SQL Server Reporting Services to provide historical reports. They tried using the Analysis Services Data Mining component to create a predictive analytics solution to predict the buying behaviour of customers. While this approach worked well with low volumes of data, it couldn't scale after more than a gigabyte of data was collected. Furthermore, they were never able to deal with the JSON data that a third-party application generated when a customer used the feedback module of the point of sale (POS) application.

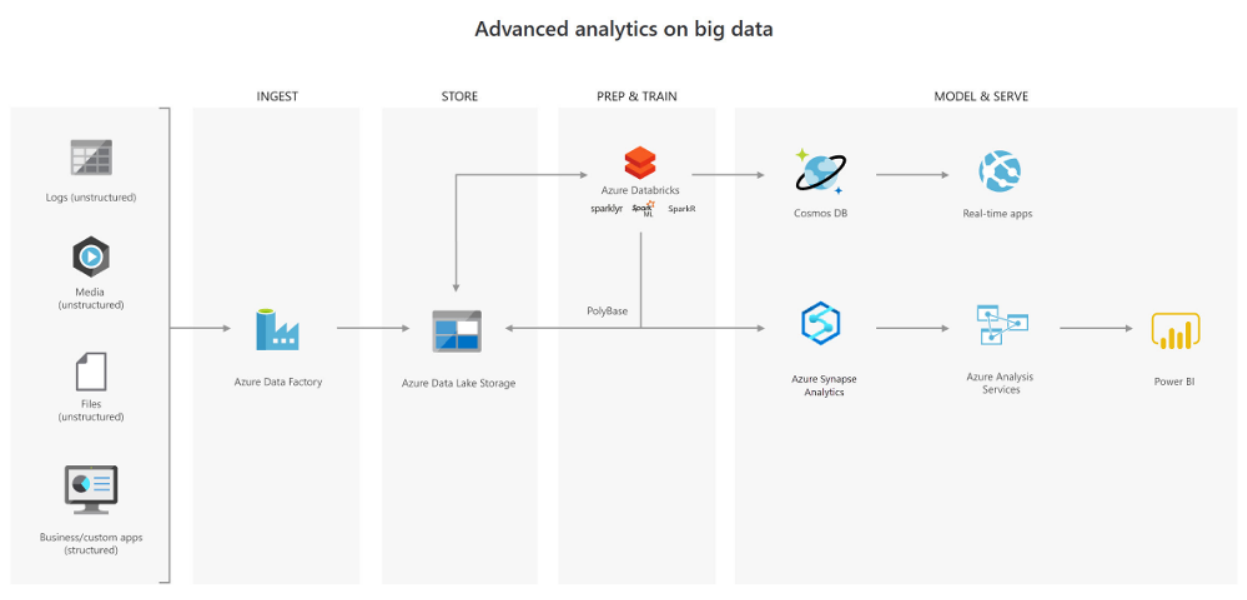
The company has turned to you for help with creating an architecture that can scale with the data needs that are required to create a predictive model and to handle the JSON data so that it's integrated into the BI solution. You suggest the following architecture:



The architecture uses Azure Data Lake Storage at the centre of the solution for a modern data warehouse. Integration Services is replaced by Azure Data Factory to ingest data into the Data Lake from a business application. This is the source for the predictive model that is built into Azure Databricks. PolyBase is used to transfer the historical data into a big data relational format that is held in Azure Synapse Analytics, which also stores the results of the trained model from Databricks. Azure Analysis Services provides the caching capability for SQL Data Warehouse to service many users and to present the data through Power BI reports.

Advanced analytics for big data

In this second use case, Azure Data Lake Storage plays an important role in providing a large-scale data store. Your skills are needed by Hydra Corporation, which is a global seller of bicycles and cycling components through a chain of resellers and on the internet. As their customers browse the product catalogue on their websites and add items to their baskets, a recommendation engine that is built into Azure Databricks recommends other products. They need to make sure that the results of their recommendation engine can scale globally. The recommendations are based on the web log files that are stored on the web servers and transferred to the Azure Databricks model hourly. The response time for the recommendation should be less than 1 ms. You propose the following architecture:



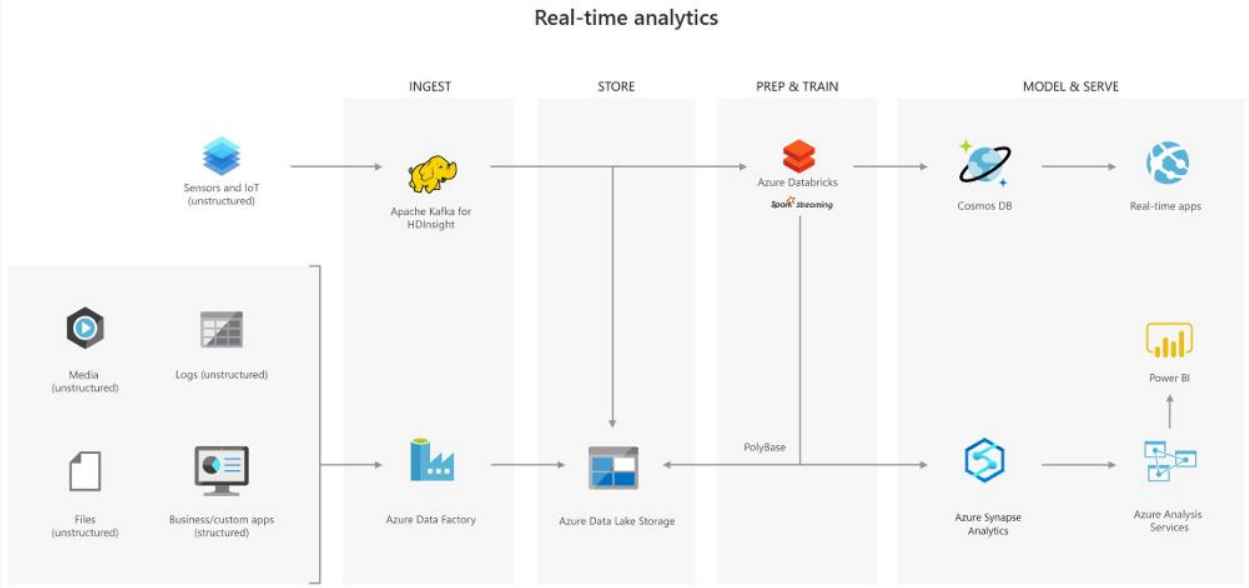
Real-time analytical solutions

To perform real-time analytical solutions, the ingestion phase of the architecture is changed for processing big data solutions. In this architecture, note the introduction of Apache Kafka for Azure HDInsight to ingest streaming data from an Internet of Things (IoT) device, although this could be replaced with Azure IoT Hub and Azure Stream Analytics. The key point is that the data is persisted in Data Lake Storage Gen2 to service other parts of the solution.

In this use case, you are a Data Engineer for HAMMER Industries, an organization that is working with a transport company to monitor the fleet of Heavy Goods Vehicles (HGV)

that drive around Europe. Each HGV is equipped with sensor hardware that will continuously report metric data on the temperature, the speed, and the oil and brake solution levels of an HGV. When the engine is turned off, the sensor also outputs a file with summary information about a trip, including the mileage and elevation of a trip. A trip is a period in which the HGV engine is turned on and off.

Both the real-time data and batch data is processed in a machine learning model to predict a maintenance schedule for each of the HGVs. This data is made available to the downstream application that third-party garage companies can use if an HGV breaks down anywhere in Europe. In addition, historical reports about the HGV should be visually presented to users. As a result, the following architecture is proposed:



In this architecture, there are two ingestion streams. Azure Data Factory ingests the summary files that are generated when the HGV engine is turned off. Apache Kafka provides the real-time ingestion engine for the telemetry data. Both data streams are stored in Azure Data Lake Store for use in the future, but they are also passed on to other technologies to meet business needs. Both streaming and batch data are provided to the predictive model in Azure Databricks, and the results are published to Azure Cosmos DB to be used by the third-party garages. PolyBase transfers data from the Data Lake Store into SQL Data Warehouse where Azure Analysis Services creates the HGV reports by using Power BI.

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

Question 5: **Incorrect**
Synapse Analytics removes the barrier of setting up multiple different services for Spark or SQL. The interoperability between Spark and SQL helps you achieve as follows:

• A shared Hive-compatible metadata system enables you to define tables on files in the data lake such that it can be consumed by either Spark or Hive.

• Both SQL and Spark can directly explore, and analyze Parquet, CSV, TSV, and JSON files stored in the data lake.

• The enablement of fast scalable load and unload for data transferring between SQL and Spark databases.

The Azure Synapse Apache Spark to Synapse SQL connector is designed to efficiently transfer data between serverless Apache Spark pools and SQL pools in Azure Synapse.

Which of the following are valid use cases for Apache Spark and SQL integration within Synapse analytics? (Select all that apply)

- ☐

  Dealing with different type of analytics

  (Correct)

- ☐

  Scalability

  (Correct)

- ☐

  Big data computational powers

  (Correct)

- ☑

  Flexibility in the use of Spark and SQL languages and frameworks

  (Correct)

- ☐

VNet and On-prem sync

Explanation

Synapse Analytics removes the barrier of setting up multiple different services for Spark or SQL. Therefore, it removes the traditional thinking about these technologies. It enables you to use both technologies within one platform, which allowed you to switch between Spark or SQL based on the needs and expertise you have in-house.

A spark orientated data engineer can now easily communicate with a SQL based data engineer and communicate together on the same platform.

The interoperability between Spark and SQL helps you achieve as follows:

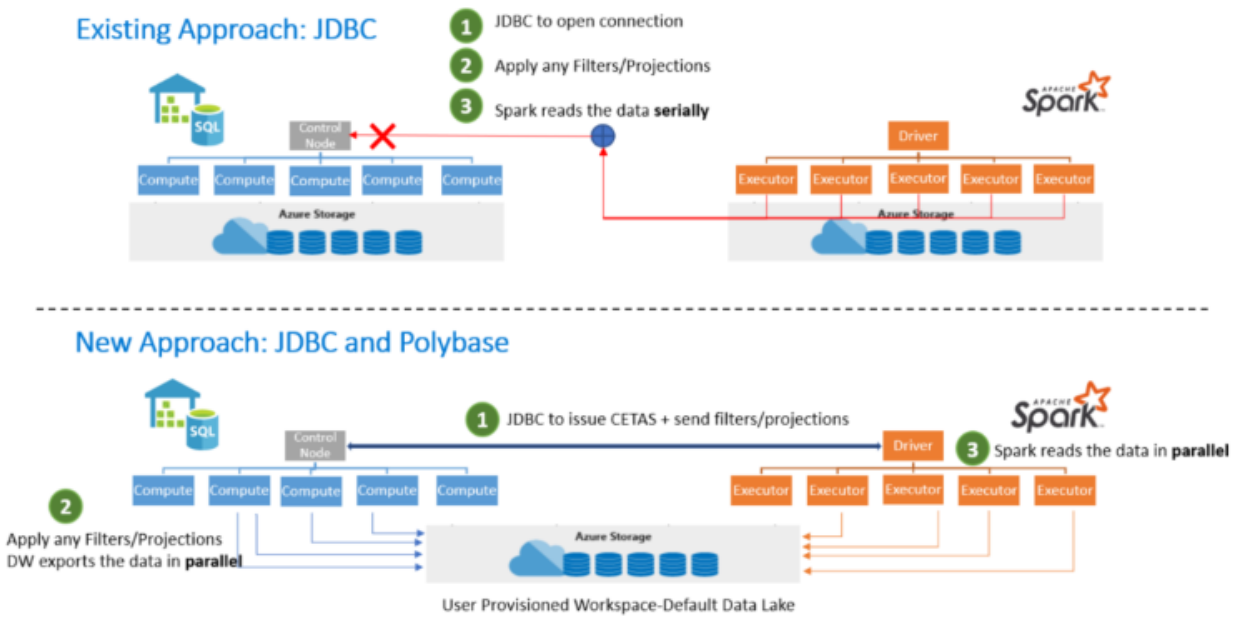
• A shared Hive-compatible metadata system enables you to define tables on files in the data lake such that it can be consumed by either Spark or Hive.

• Both SQL and Spark can directly explore, and analyze Parquet, CSV, TSV, and JSON files stored in the data lake.

• The enablement of fast scalable load and unload for data transferring between SQL and Spark databases.

The question might raise as how would that SQL and Spark integration then work.

That's when the Azure Synapse Apache Spark to Synapse SQL connector comes in place. It is designed to efficiently transfer data between serverless Apache Spark pools (preview) and SQL pools in Azure Synapse. However, at the moment, the Azure Synapse Apache Spark to Synapse SQL connector works on dedicated SQL pools only, it doesn't work with serverless SQL pools.



In the commonly used existing approach, you often see the use of the JDBC. The JDBC would open the connection. Then, filters and projections would be applied and spark would read the data serially. Given two distributed systems such as Spark and SQL pools, JDBC could become a bottleneck with serial data transfer.

Therefore the New Approach we would take is JDBC and PolyBase. First, the JDBC issues CETAS and send filters and projections. Then filters and projections would be applied and the DataWarehouse exports the data in parallel. Spark reads the data in parallel all based on the user provisioned workspace default data lake storage.

The Azure Synapse Apache Spark Pool to Synapse SQL connector would then be a data source implementation for apache spark where the ADLS Gen 2 is used as well as PolyBase in the dedicated SQL Pools to transfer data between the Spark instance and SQL pool efficiently.

The use cases for Apache Spark and SQL integration within Synapse analytics are as following:

- **Dealing with different type of analytics**

- **Scalability**

- **Big data computational powers**

- **Flexibility in the use of Spark and SQL languages and frameworks**

Since Apache Spark is integrated in Synapse Analytics, there is more to that than giving use for the big data analytics framework Apache Spark enables. When you deploy a synapse cluster, ADLS Gen2 capacity that can store Spark SQL Tables is provisioned with it.

If you use Spark SQL Tables, you might know that these tables can be queried from a SQL-server-based T-SQL language without you having to use commands like CREATE EXTERNAL TABLE. Within synapse analytics, these queries integrate natively with data files that are stored in an Apache Parquet format.

The other thing to take in mind is that beyond the capabilities mentioned above, the Azure Synapse Studio experience gives you an integrated notebook experience. Within this notebook experience, you can attach a SQL or Spark pool, and develop and execute, for example, transformation pipelines using Python, Scala, and native Spark SQL.

So, let's say you would like to write to a SQL pool after you've performed engineering tasks in spark. You can reference the SQL Pool data as a source for joining with Spark Dataframes that can contain data from other files. When you decide to use the Azure Synapse Apache Spark to Synapse SQL connector, you're now able to efficiently transfer data between the Spark and SQL Pools.

The Azure Synapse Apache Spark pool to Synapse SQL connector is a data source implementation for Apache Spark. It uses the Azure Data Lake Storage Gen2 and PolyBase in SQL pools to efficiently transfer data between the Spark cluster and the Synapse SQL instance.

https://docs.microsoft.com/en-us/azure/synapse-analytics/spark/synapse-spark-sql-pool-import-export

Question 6: **Correct**

Scenario: Big Belly Foods, Inc. (BB) owns and operates 300 convenience stores across LatAm. The company sells a variety of packaged foods and drinks, as well as a variety of prepared foods, such as sandwiches and pizzas. The company has a loyalty club whereby members can get daily discounts on specific items by providing their membership number at checkout.

BB employs business analysts who prefer to analyze data by using Microsoft Power BI, and data scientists who prefer analyzing data in Azure Databricks notebooks. You have been hired as an Azure Expert SME and you are to consult the IT team on various Azure related projects.

Business Requirements:

BB wants to create a new analytics environment in Azure to meet the following requirements:

• See inventory levels across the stores. Data must be updated as close to real time as possible.

• Execute ad hoc analytical queries on historical data to identify whether the loyalty club discounts increase sales of the discounted products.

• Every four hours, notify store employees about how many prepared food items to produce based on historical demand from the sales data.

Technical Requirements:

BB identifies the following technical requirements:

• Minimize the number of different Azure services needed to achieve the business goals.

• Use platform as a service (PaaS) offerings whenever possible and avoid having to provision virtual machines that must be managed by BB.

• Ensure that the analytical data store is accessible only to the company's on-premises network and Azure services.

• Use Azure Active Directory (Azure AD) authentication whenever possible.

• Use the principle of least privilege when designing security.

• Stage Inventory data in Azure Data Lake Storage Gen2 before loading the data into the analytical data store. BB wants to remove transient data from Data

• Lake Storage once the data is no longer in use. Files that have a modified date that is older than 14 days must be removed.

• Limit the business analysts' access to customer contact information, such as phone numbers, because this type of data is not analytically relevant.

• Ensure that you can quickly restore a copy of the analytical data store within one hour in the event of corruption or accidental deletion.

Planned Environment:

BB plans to implement the following environment:

• The application development team will create an Azure event hub to receive real-time sales data, including store number, date, time, product ID, customer loyalty number,

price, and discount amount, from the point of sale (POS) system and output the data to data storage in Azure.

• Customer data, including name, contact information, and loyalty number, comes from Salesforce, a SaaS application, and can be imported into Azure once every eight hours. Row modified dates are not trusted in the source table.

• Product data, including product ID, name, and category, comes from Salesforce and can be imported into Azure once every eight hours. Row modified dates are not trusted in the source table.

• Daily inventory data comes from a Microsoft SQL server located on a private network.

• BB currently has 5 TB of historical sales data and 100 GB of customer data. The company expects approximately 100 GB of new data per month for the next year.

• BB will build a custom application named FoodPrep to provide store employees with the calculation results of how many prepared food items to produce every four hours.

• BB does not plan to implement Azure ExpressRoute or a VPN between the on-premises network and Azure.

The Ask:

The team looks to you for direction on what should be done to improve high availability of the real-time data processing solution. Which of the following should you propose as the best solution?

- ○

  Deploy a High Concurrency Databricks cluster.

- ○

  Deploy an Azure Stream Analytics job and use an Azure Automation runbook to check the status of the job and to start the job if it stops.

- ○

  Set Data Lake Storage to use geo-redundant storage (GRS).

- ◉

  Deploy identical Azure Stream Analytics jobs to paired regions in Azure.

(Correct)

Explanation

The best solution to move forward is to deploy identical Azure Stream Analytics jobs to paired regions in Azure. The application development team should create an Azure event hub to receive real-time sales data, including store number, date, time, product ID, customer loyalty number, price, and discount amount, from the point of sale (POS) system and output the data to data storage in Azure.

Guarantee Stream Analytics job reliability during service updates

Part of being a fully managed service is the capability to introduce new service functionality and improvements at a rapid pace. As a result, Stream Analytics can have a service update deploy on a weekly (or more frequent) basis. No matter how much testing is done there is still a risk that an existing, running job may break due to the introduction of a bug. If you are running mission critical jobs, these risks need to be avoided. You can reduce this risk by following Azure's **paired region** model.

How do Azure paired regions address this concern?

Stream Analytics guarantees jobs in paired regions are updated in separate batches. As a result there is a sufficient time gap between the updates to identify potential issues and remediate them.

With the exception of Central India (whose paired region, South India, does not have Stream Analytics presence), the deployment of an update to Stream Analytics would not occur at the same time in a set of paired regions. Deployments in multiple regions **in the same group** may occur **at the same time**.

The article on **availability and paired regions** has the most up-to-date information on which regions are paired.

It is recommended to deploy identical jobs to both paired regions. You should then monitor these jobs to get notified when something unexpected happens. If one of these jobs ends up in a Failed state after a Stream Analytics service update, you can contact customer support to help identify the root cause. You should also fail over any downstream consumers to the healthy job output.

https://docs.microsoft.com/en-us/azure/stream-analytics/stream-analytics-job-reliability

Question 7: **Incorrect**

Which Workload Management capability manages minimum and maximum resource allocations during peak periods?

- **Workload Containment**
- **Workload Isolation**

  (Correct)
- **Workload Importance**
- **Workload Classification**

  (Incorrect)

Explanation
Workload Isolation assigns maximum and minimum usage values for varying resources under load. These adjustments can be done live without having to take the SQL Pool offline.

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

• Workload Classification

• Workload Importance

• Workload Isolation

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

Question 8: **Incorrect**

Scenario: Pennyworth's Haberdashery is a clothing retailer based in London. The company has 2,000 retail stores across the EU and an emerging online presence. The network contains an Active Directory forest named pennyworths.com. The forest it integrated with an Azure Active Directory (Azure AD) tenant named pennyworths.com. Pennyworth's has an Azure subscription associated to the pennyworths.com Azure AD tenant.

Pennyworth's has three years of customer, transactional, operational, sourcing, and supplier data comprised of 10 billion records stored across multiple on-premises. Microsoft SQL Server servers. The SQL Server instances contain data from various operational systems. The data is loaded into the instances by using SQL Server Integration Services (SSIS) packages.

You have been hired as a consultant by Alfred Pennyworth to advise on very important projects within the company.

During your assessment of the IT environment, you estimate that combining all product sales transactions into a company-wide sales transactions dataset will result in a single table that contains 5 billion rows, with one row per transaction.

Most queries targeting the sales transactions data will be used to identify which products were sold in retail stores and which products were sold online during different time periods. Sales transaction data that is older than three years will be removed monthly.

The IT team plans to create a retail store table that will contain the address of each retail store. The table will be approximately 2 MB. Queries for retail store sales will include the retail store addresses.

They also plan to create a promotional table that will contain a promotion ID. The promotion ID will be associated to a specific product. The product will be identified by a product ID. The table will be approximately 5 GB.

The e-commerce department at Pennyworth's develops an Azure logic app that captures trending Twitter feeds referencing the company's products and pushes the products to Azure Event Hubs.

Planned Changes and Requirements

Pennyworth's plans to implement the following changes:

• Load the sales transaction dataset to Azure Synapse Analytics.

• Integrate on-premises data stores with Azure Synapse Analytics by using SSIS packages.

• Use Azure Synapse Analytics to analyze Twitter feeds to assess customer sentiments about products.

Sales Transaction Dataset Requirements

Pennyworth's identifies the following requirements for the sales transaction dataset:

• Partition data that contains sales transaction records. Partitions must be designed to provide efficient loads by month. Boundary values must belong to the partition on the right.

• Ensure that queries joining and filtering sales transaction records based on product ID complete as quickly as possible.

• Implement a surrogate key to account for changes to the retail store addresses.

• Ensure that data storage costs and performance are predictable.

• Minimize how long it takes to remove old records.

Customer Sentiment Analytics Requirements

Pennyworth's identifies the following requirements for customer sentiment analytics:

• Allow Pennyworth's users to use PolyBase in an Azure Synapse Analytics dedicated SQL pool to query the content of the data records that host the Twitter feeds.

• Data must be protected by using row-level security (RLS). The users must be authenticated by using their own Azure AD credentials.

• Maximize the throughput of ingesting Twitter feeds from Event Hubs to Azure Storage without purchasing additional throughput or capacity units.

• Store Twitter feeds in Azure Storage by using Event Hubs Capture. The feeds will be converted into Parquet files.

• Ensure that the data store supports Azure AD-based access control down to the object level.

• Minimize administrative effort to maintain the Twitter feed data records.

• Purge Twitter feed data records that are older than two years.

Data Integration Requirements

Pennyworth's identifies the following requirements for data integration:

• Use an Azure service that leverages the existing SSIS packages to ingest on-premises data into datasets stored in a dedicated SQL pool of Azure Synapse Analytics and transform the data.

• Identify a process to ensure that changes to the ingestion and transformation activities can be version-controlled and developed independently by multiple data engineers.

The Ask:

Alfred places a great importance on this project and asks you to work closely with the team to implement the surrogate key for the retail store table. The solution must meet the sales transaction dataset requirements.

Which of the following should you advise the team to create?

- ○

  A table that has an `IDENTITY` property.

  (Correct)

- ○

  A system-versioned temporal table.

- ○

  A table that has a `FOREIGN KEY` constraint.

- ◉

  A user-defined `SEQUENCE` object.

  (Incorrect)

Explanation

The best way to implement a surrogate key to account for changes to the retail store addresses is to create a table that has an `IDENTITY` property.

A surrogate key on a table is a column with a unique identifier for each row. The key is not generated from the table data. Data modelers like to create surrogate keys on their tables when they design data warehouse models. You can use the `IDENTITY` property to achieve this goal simply and effectively without affecting load performance.

Using IDENTITY to create surrogate keys using dedicated SQL pool in Azure Synapse Analytics

What is a surrogate key?

A surrogate key on a table is a column with a unique identifier for each row. The key is not generated from the table data. Data modelers like to create surrogate keys on their tables when they design data warehouse models. You can use the IDENTITY property to achieve this goal simply and effectively without affecting load performance.

Note: In Azure Synapse Analytics, the IDENTITY value increases on its own in each distribution and does not overlap with IDENTITY values in other distributions. The IDENTITY value in Synapse is not guaranteed to be unique if the user explicitly inserts a duplicate value with "SET IDENTITY_INSERT ON" or reseeds IDENTITY. For details, see [CREATE TABLE (Transact-SQL) IDENTITY (Property)](#).

Creating a table with an IDENTITY column

The IDENTITY property is designed to scale out across all the distributions in the dedicated SQL pool without affecting load performance. Therefore, the implementation of IDENTITY is oriented toward achieving these goals.

You can define a table as having the IDENTITY property when you first create the table by using syntax that is similar to the following statement:

```
SQL
CREATE TABLE dbo.T1
(    C1 INT IDENTITY(1,1) NOT NULL
,    C2 INT NULL
)
WITH
(   DISTRIBUTION = HASH(C2)
,   CLUSTERED COLUMNSTORE INDEX
)
;
```

In the preceding example, two rows landed in distribution 1. The first row has the surrogate value of 1 in column `C1`, and the second row has the surrogate value of 61. Both of these values were generated by the IDENTITY property. However, the allocation of the values is not contiguous. This behavior is by design.

Skewed data

The range of values for the data type are spread evenly across the distributions. If a distributed table suffers from skewed data, then the range of values available to the datatype can be exhausted prematurely. For example, if all the data ends up in a single distribution, then effectively the table has access to only one-sixtieth of the values of the data type. For this reason, the IDENTITY property is limited to `INT` and `BIGINT` data types only.

SELECT..INTO

When an existing IDENTITY column is selected into a new table, the new column inherits the IDENTITY property, unless one of the following conditions is true:

• The SELECT statement contains a join.

• Multiple SELECT statements are joined by using UNION.

• The IDENTITY column is listed more than one time in the SELECT list.

• The IDENTITY column is part of an expression.

If any one of these conditions is true, the column is created NOT NULL instead of inheriting the IDENTITY property.

CREATE TABLE AS SELECT

`CREATE TABLE AS SELECT` (CTAS) follows the same SQL Server behavior that's documented for SELECT..INTO. However, you can't specify an IDENTITY property in the column definition of the `CREATE TABLE` part of the statement. You also can't use the IDENTITY function in the `SELECT` part of the CTAS. To populate a table, you need to use `CREATE TABLE` to define the table followed by `INSERT..SELECT` to populate it.

Explicitly inserting values into an IDENTITY column

Dedicated SQL pool supports `SET IDENTITY_INSERT <your table> ON|OFF` syntax. You can use this syntax to explicitly insert values into the IDENTITY column.

Many data modelers like to use predefined negative values for certain rows in their dimensions. An example is the -1 or "unknown member" row.

The next script shows how to explicitly add this row by using SET IDENTITY_INSERT:

```
1. SQL
2. SET IDENTITY_INSERT dbo.T1 ON;
```

```
3.
4. INSERT INTO dbo.T1
5. (   C1
6. ,   C2
7. )
8. VALUES (-1,'UNKNOWN')
9. ;
10.
11.SET IDENTITY_INSERT dbo.T1 OFF;
12.
13.SELECT     *
14.FROM    dbo.T1
15.;
```

Loading data

The presence of the IDENTITY property has some implications to your data-loading code. This section highlights some basic patterns for loading data into tables by using IDENTITY.

To load data into a table and generate a surrogate key by using IDENTITY, create the table and then use INSERT..SELECT or INSERT..VALUES to perform the load.

The following example highlights the basic pattern:

```
1. SQL
2. --CREATE TABLE with IDENTITY
3. CREATE TABLE dbo.T1
4. (    C1 INT IDENTITY(1,1)
5. ,    C2 VARCHAR(30)
6. )
7. WITH
8. (   DISTRIBUTION = HASH(C2)
9. ,   CLUSTERED COLUMNSTORE INDEX
10.)
11.;
12.
13.--Use INSERT..SELECT to populate the table from an external table
14.INSERT INTO dbo.T1
15.(C2)
16.SELECT     C2
17.FROM    ext.T1
18.;
19.
20.SELECT *
21.FROM   dbo.T1
22.;
23.
24.DBCC PDW_SHOWSPACEUSED('dbo.T1');
```

Note: It's not possible to use `CREATE TABLE AS SELECT` currently when loading data into a table with an IDENTITY column.

System views

You can use the sys.identity_columns catalog view to identify a column that has the IDENTITY property.

To help you better understand the database schema, this example shows how to integrate sys.identity_column` with other system catalog views:

```
SQL
SELECT  sm.name
,       tb.name
,       co.name
,       CASE WHEN ic.column_id IS NOT NULL
             THEN 1
        ELSE 0
        END AS is_identity
FROM        sys.schemas AS sm
JOIN        sys.tables  AS tb           ON  sm.schema_id = tb.schema_id
JOIN        sys.columns AS co           ON  tb.object_id = co.object_id
LEFT JOIN   sys.identity_columns AS ic  ON  co.object_id = ic.object_id
                                        AND co.column_id = ic.column_id
WHERE   sm.name = 'dbo'
AND     tb.name = 'T1'
;
```

Limitations

The IDENTITY property can't be used:

• When the column data type is not INT or BIGINT

• When the column is also the distribution key

• When the table is an external table

The following related functions are not supported in dedicated SQL pool:

• IDENTITY()

• @@IDENTITY

• SCOPE_IDENTITY

• IDENT_CURRENT

• IDENT_INCR

• IDENT_SEED

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

Question 9: **Correct**
How do you list files in DBFS within a notebook?

- `%dfs ls /my-file-path`
- `ls /my-file-path`
- `%fs dir /my-file-path`
- `%fs ls /my-file-path`

  (Correct)

Explanation
DBFS and local driver node paths

You can work with files on DBFS or on the local driver node of the cluster. You can access the file system using magic commands such as `%fs` or `%sh`.

You add the file system magic to the cell before executing the ls command.

https://docs.microsoft.com/en-us/azure/databricks/data/databricks-file-system

Question 10: **Correct**
Identify the missing word(s) in the following sentence within the context of Microsoft Azure.

You can use *account-level* SAS to allow access to anything that a service-level SAS can allow, plus additional resources and abilities. You'd use this type of SAS, for example, … [?] (Select all that apply)

- **to allow an app to retrieve a list of files in a file system.**

to allow an app to download a file.

- **None of the listed options.**

- **to allow the ability to create file systems.**

  (Correct)

Explanation
Types of shared access signatures

You can use a *service-level* SAS to allow access to specific resources in a storage account. You'd use this type of SAS, for example, to allow an app to retrieve a list of files in a file system, or to download a file.

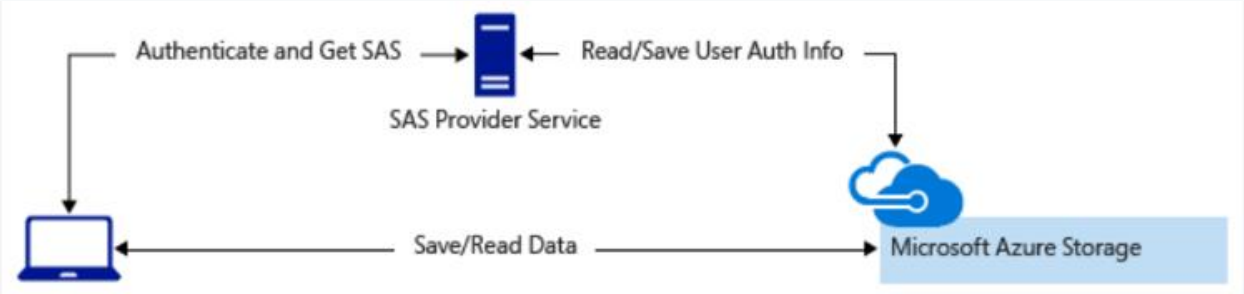
Use an *account-level* SAS to allow access to anything that a service-level SAS can allow, plus additional resources and abilities. For example, you can use an account-level SAS to allow the ability to create file systems.

You'd typically use a SAS for a service where users read and write their data to your storage account. Accounts that store user data have two typical designs:

• Clients upload and download data through a front-end proxy service, which performs authentication. This front-end proxy service has the advantage of allowing validation of business rules. But, if the service must handle large amounts of data or high-volume transactions, you might find it complicated or expensive to scale this service to match demand.

• A lightweight service authenticates the client, as needed. Next, it generates a SAS. After receiving the SAS, the client can access storage account resources directly. The

SAS defines the client's permissions and access interval. It reduces the need to route all data through the front-end proxy service.



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

Question 11: **Correct**

Scenario: Dr. Karl Malus works for the Power Broker Corporation (PBC) founded by Curtiss Jackson, using technology to service various countries and their military efforts. You have been contracted by the company to assist Dr. Malus with their Microsoft Azure Synapse projects.

PBC has an Azure subscription that contains a logical Microsoft SQL server named Server1. Server1 hosts an Azure Synapse Analytics SQL dedicated pool named Pool1.

Dr. Malus is looking for a recommendation for a Transparent Data Encryption (TDE) solution for Server1. The solution must meet the following requirements:

Required:

• Track the usage of encryption keys.

• Maintain the access of client apps to Pool1 in the event of an Azure datacentre outage that affects the availability of the encryption keys.

Which of the following should you include in the recommendation for the "Track the usage of encryption key" requirement?

- **TDE with customer-managed keys**

  (Correct)

- **TDE with platform-managed keys**

- **Always Encrypted**
- **None of the options listed will meet the requirement**
- **Any of the options listed will meet the requirement**

Explanation

You should include in "TDE with customer-managed keys" in the recommendation for the first requirement listed.

Customer-managed keys are stored in the Azure Key Vault. You can monitor how and when your key vaults are accessed, and by whom. You can do this by enabling logging for Azure Key Vault, which saves information in an Azure storage account that you provide.

After you create one or more key vaults, you'll likely want to monitor how and when your key vaults are accessed, and by whom. You can do this by enabling logging for Azure Key Vault, which saves information in an Azure storage account that you provide. For step by step guidance on setting this up, see How to enable Key Vault logging.

What is logged:

All authenticated REST API requests, including failed requests as a result of access permissions, system errors, or bad requests.

Operations on the key vault itself, including creation, deletion, setting key vault access policies, and updating key vault attributes such as tags.

Operations on keys and secrets in the key vault, including: Creating, modifying, or deleting these keys or secrets. Signing, verifying, encrypting, decrypting, wrapping and unwrapping keys, getting secrets, and listing keys and secrets (and their versions).

Unauthenticated requests that result in a 401 response. Examples are requests that don't have a bearer token, that are malformed or expired, or that have an invalid token.

Azure Event Grid notification events for the following conditions: expired, near expiration, and changed vault access policy (the new version event isn't logged). Events are logged even if there's an event subscription created on the key vault.

You can access your logging information 10 minutes (at most) after the key vault operation. In most cases, it will be quicker than this. It's up to you to manage your logs in your storage account:

Use standard Azure access control methods in your storage account to secure your logs by restricting who can access them.

Delete logs that you no longer want to keep in your storage account.

https://docs.microsoft.com/en-us/azure/key-vault/general/logging?tabs=Vault

Question 12: **Incorrect**
Identify the missing word(s) in the following sentence within the context of Microsoft Azure.

Spark is a Distributed computing environment. The unit of distribution is a Spark Cluster. Every Cluster has a Driver and one or more executors. Work submitted to the Cluster … [?]

- ○

  split into as many independent Jobs as needed.

  (Correct)

- ○

  divided into a maximum of 10 independent Jobs.

- ○

  specifies the types and sizes of the virtual machines.

- ◉

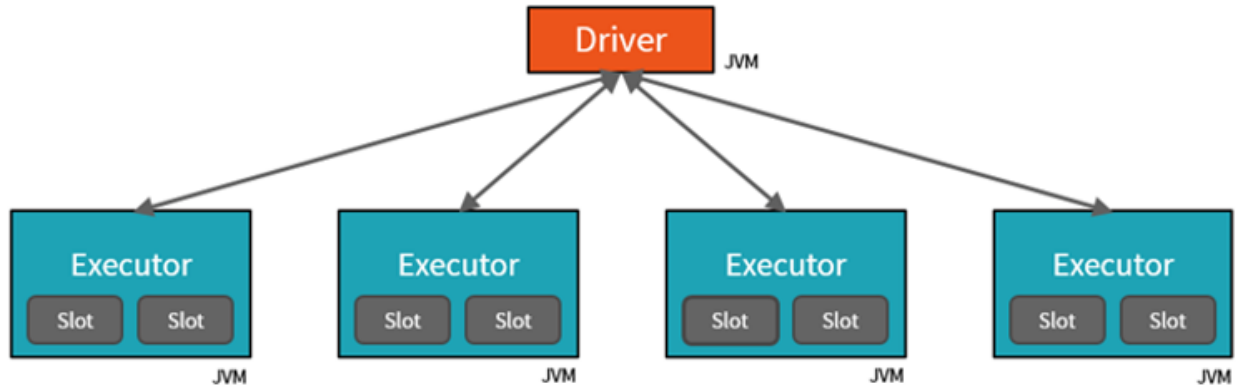
  must decide how to partition the data so that it can be distributed for parallel processing.

  (Incorrect)

Explanation
Spark is a Distributed computing environment. The unit of distribution is a Spark Cluster. Every Cluster has a Driver and one or more executors. Work submitted to the Cluster is split into as many independent Jobs as needed. This is how work is

distributed across the Cluster's nodes. Jobs are further subdivided into tasks. The input to a job is partitioned into one or more partitions. These partitions are the unit of work for each slot. In between tasks, partitions may need to be re-organized and shared over the network.

The cluster: Drivers, executors, slots & tasks



• The **Driver** is the JVM in which our application runs.

• The secret to Spark's awesome performance is parallelism.

  • Scaling vertically is limited to a finite amount of RAM, Threads and CPU speeds.

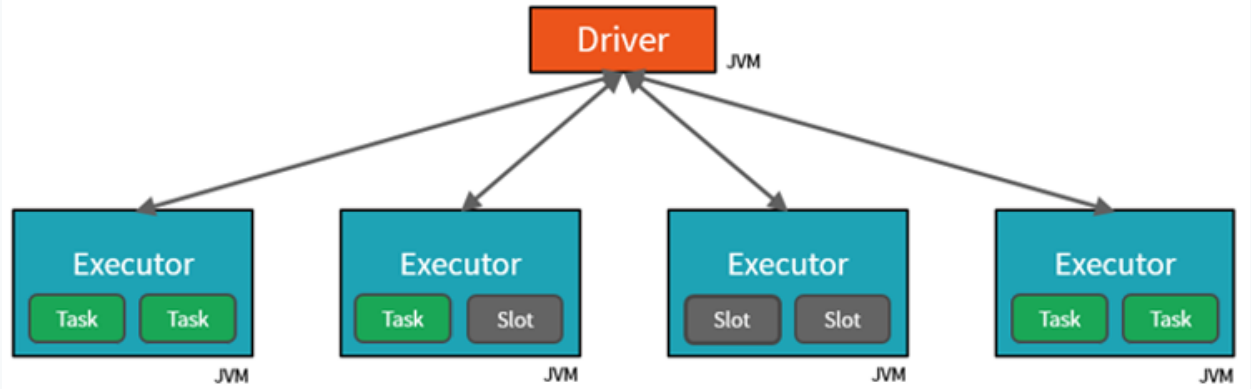
  • Scaling horizontally means we can simply add new "nodes" to the cluster almost endlessly.

• We parallelize at two levels:

  • The first level of parallelization is the **Executor** - a Java virtual machine running on a node, typically, one instance per node.

  • The second level of parallelization is the **Slot** - the number of which is determined by the number of cores and CPUs of each node.

• Each **Executor** has a number of **Slots** to which parallelized **Tasks** can be assigned to it by the **Driver**.



• The JVM is naturally multithreaded, but a single JVM, such as our **Driver**, has a finite upper limit.

• By creating **Tasks**, the **Driver** can assign units of work to **Slots** for parallel execution.

• Additionally, the **Driver** must also decide how to partition the data so that it can be distributed for parallel processing (not shown here).

• Consequently, the **Driver** is assigning a **Partition** of data to each task - in this way each **Task** knows which piece of data it is to process.

• Once started, each **Task** will fetch from the original data source the **Partition** of data assigned to it.

Jobs & stages

• Each parallelized action is referred to as a **Job**.

• The results of each **Job** (parallelized/distributed action) is returned to the **Driver**.

• Depending on the work required, multiple **Jobs** will be required.

• Each **Job** is broken down into **Stages**.

  • This would be analogous to building a house (the job)

  • The first stage would be to lay the foundation.

  • The second stage would be to erect the walls.

  • The third stage would be to add the room.

• Attempting to do any of these steps out of order just won't make sense, if not just impossible.

Cluster management

• At a much lower level, Spark Core employs a **Cluster Manager** that is responsible for provisioning nodes in our cluster.

• Databricks provides a robust, high-performing **Cluster Manager** as part of its overall offerings.

• In each of these scenarios, the **Driver** is [presumably] running on one node, with each **Executors** running on N different nodes.

• From a developer's and learner's perspective my primary focus is on...

• The number of **Partitions** my data is divided into.

• The number of **Slots** I have for parallel execution.

• How many **Jobs** am I triggering?

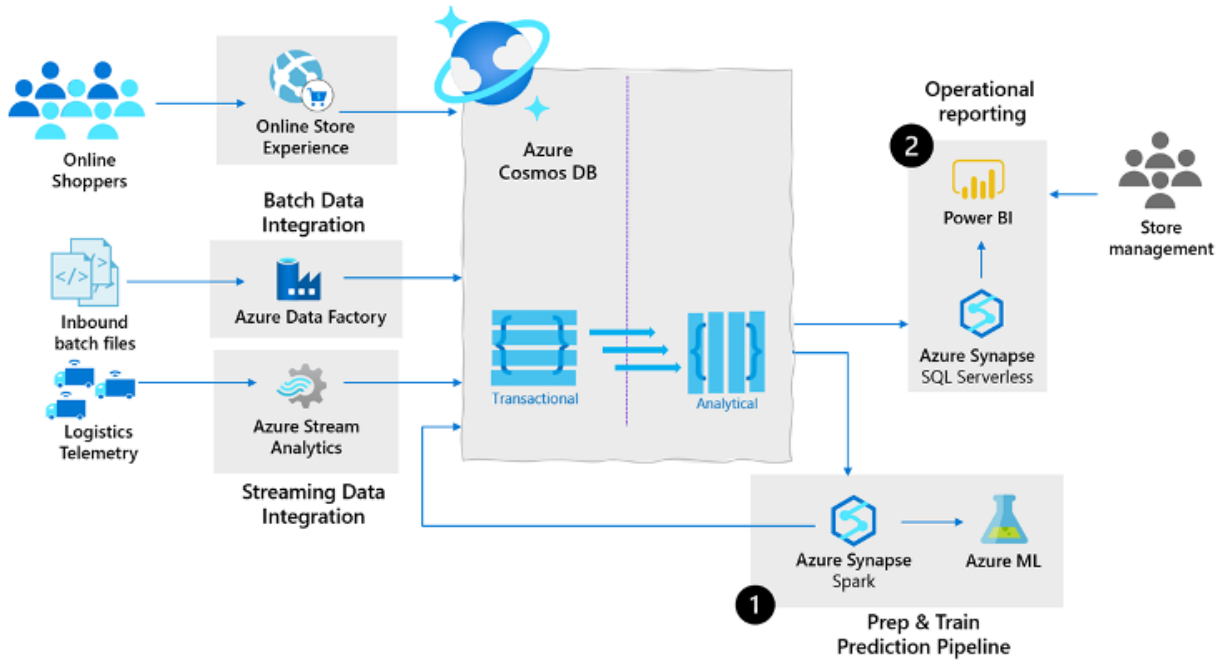
• And lastly the **Stages** those jobs are divided into.

https://databricks.com/blog/2017/11/15/a-technical-overview-of-azure-databricks.html

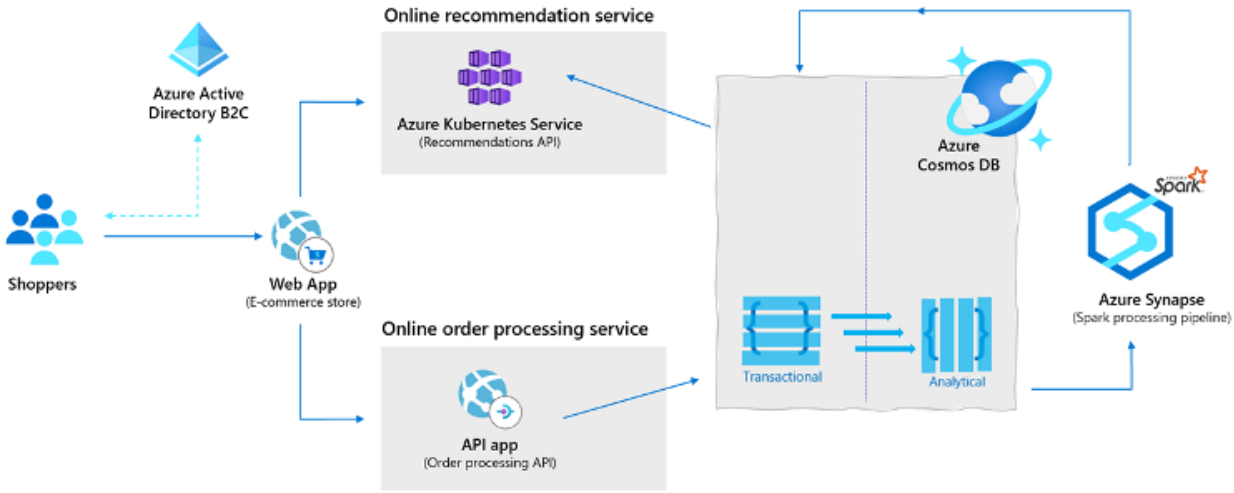
Question 13: **Incorrect**
Scenario: You are working at Jungle.com which is a web-based retailer which needs to perform real-time basket analysis to make product recommendations to customers who are about to purchase products. This increased revenues for the organization as the provided targeted suggestions at the point of sale.

Review the following architecture designs.

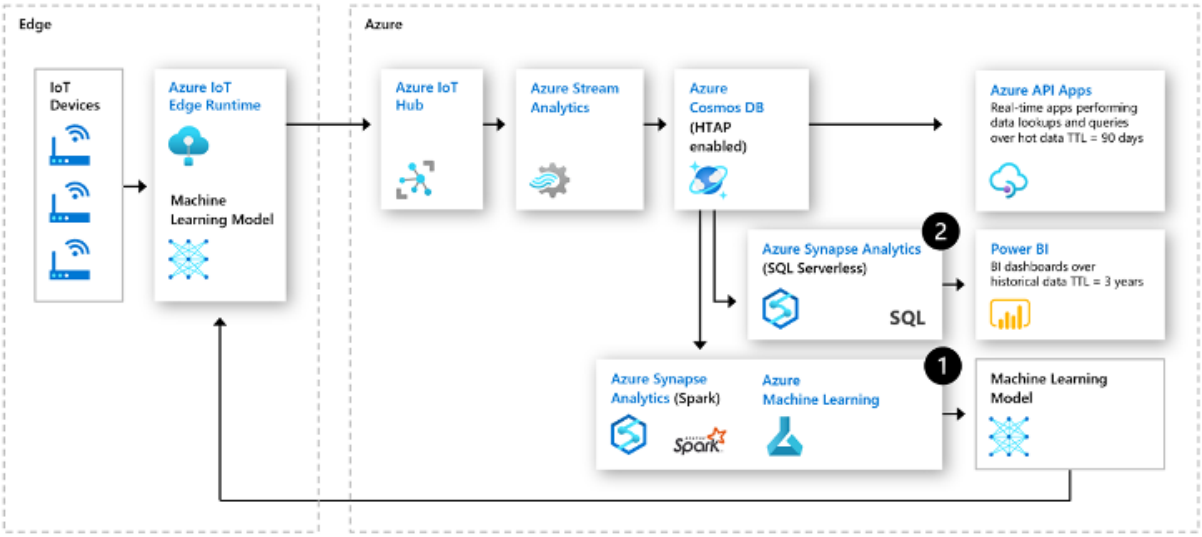
Design A:



Design B:



Design C:



Which design would be best suited for the need?

- **Design C**

  (Incorrect)

- **None of the listed options**

- **Design A**

- **Design B**

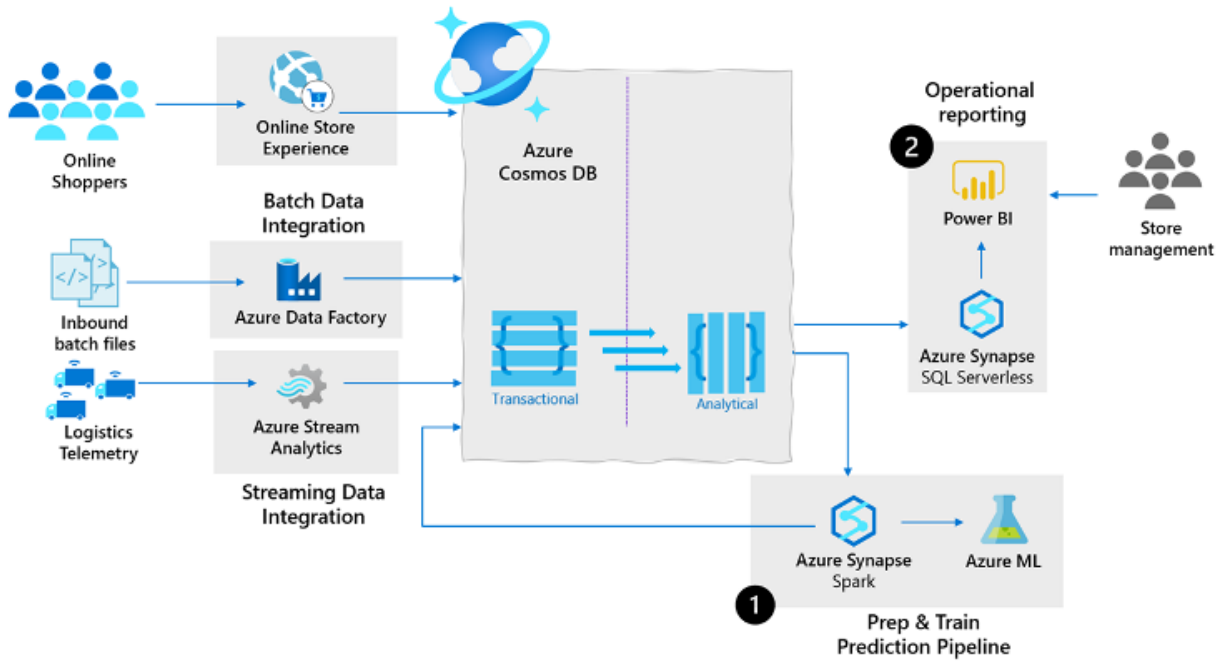
  (Correct)

Explanation
Supply chain analytics, forecasting and reporting.

With supply chains generating increasing volumes of operational data every minute for orders, shipments and sales transactions, manufactures and retailers need an

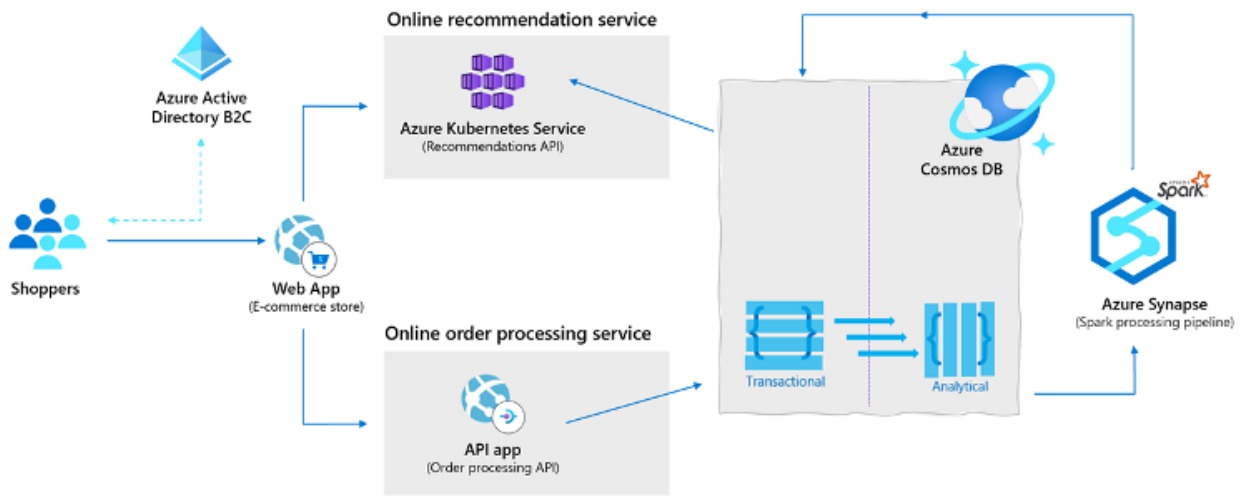
operational database that can scale to handle the data volumes as well as an analytical platform to get to a level of real-time contextual intelligence to stay ahead of the curve.

Azure Synapse Link for Cosmos DB allows these organizations to store data from their sales systems, ingest real-time telemetry data from in vehicle systems and integrate date from their ERP systems into a common operational store in Azure Cosmos DB and then leverage the data from Synapse analytics to enable both predictive analytics scenarios such as stock out monitoring and supply chain bottleneck management (1) in addition to enabling operational reporting directly on their operation data using standard reporting tools such as Power BI (2).



Retail real-time personalization.

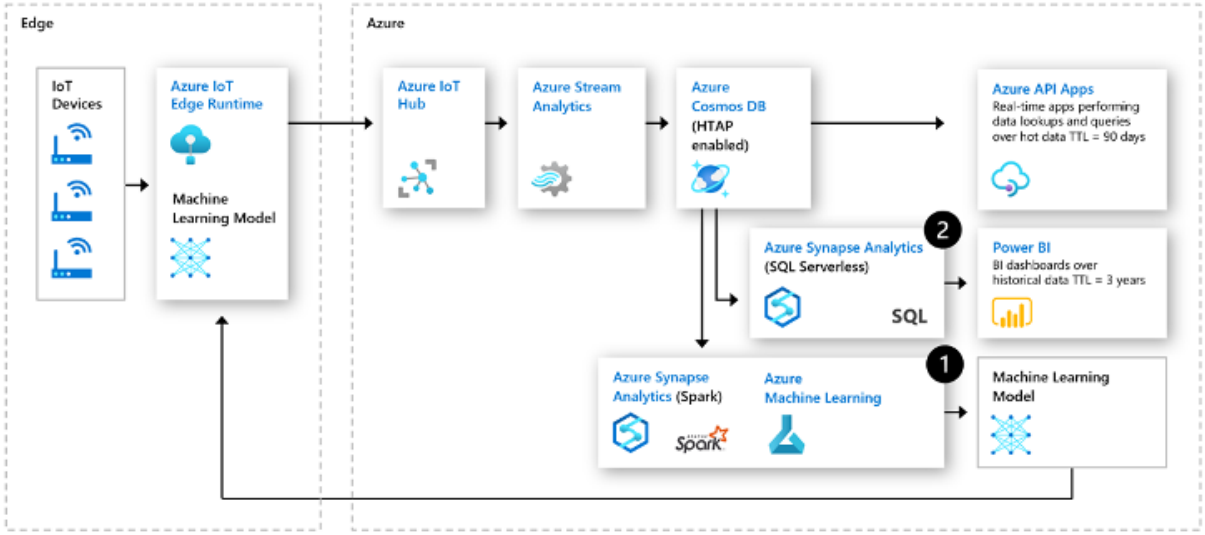
In retail, many web-based retailers will perform real-time basket analysis to make product recommendations to customers who are about to purchase products. This increased revenues for these organizations as the provided targeted suggestions at the point of sales.



Predictive maintenance using anomaly detection with IOT

Industrial IOT innovations have drastically reduced downtimes of machinery and increased overall efficiency across all fields of industry. One of such innovations is predictive maintenance analytics for machinery at the edge of the cloud.

The following architecture leverages the cloud native HTAP capabilities of Azure Synapse Link for Azure Cosmos DB in IoT predictive maintenance:



https://docs.microsoft.com/en-us/azure/cosmos-db/synapse-link-use-cases

Question 14: **Incorrect**

Scenario: Café Rivoli is a restaurant in Manhattan that has a very exclusive clientele. Steve Rogers and Bruce Banner are the creators of this iconic restaurant and continue to strive to improve the business. Part of the improvements has been to implement Microsoft Azure into their business and have hired you to lead several projects in the company.

Steve plans to have the IT team use a slowly changing dimension (SCD) to update the dimension members without keeping track of history?

Which SCD type would be the best fit?

- ○

  Type 3 SCD

- ○

  Type 6 SCD

- ○

  Type 2 SCD

- ◉

  Type 4 SCD

  (Incorrect)

- ○

  Type 5 SCD

- ○

  Type 1 SCD

  (Correct)

Explanation
When a value changes, Type 1 SCD will update the existing record without keeping history.

Star schema design theory refers to common SCD types. The most common are Type 1 and Type 2. In practice a dimension table may support a combination of history tracking methods, including Type 3 and Type 6.

Type 1 SCD

A **Type 1 SCD** always reflects the latest values, and when changes in source data are detected, the dimension table data is overwritten. This design approach is common for columns that store supplementary values, like the email address or phone number of a customer. When a customer email address or phone number changes, the dimension table updates the customer row with the new values. It's as if the customer always had this contact information. The key field, such as CustomerID, would stay the same so the records in the fact table automatically link to the updated customer record.

CustomerID	FirstName	LastName	EmailAddress	CompanyName	InsertedDate	ModifiedDate
2	Keith	Harris	keith0@aw.com	Progressive Sports	2021-03-20	2021-03-20
3	Donna	Carreras	donna0@aw.com	A Bike Store	2021-03-20	2021-03-20

CustomerID	FirstName	LastName	EmailAddress	CompanyName	InsertedDate	ModifiedDate
2	Keith	Harris	keith0@aw.com	Progressive Sports	2021-03-20	2021-03-20
3	Donna	Carreras	donna0@aw.com	Bikes, Bikes, Bikes	2021-03-20	2021-03-22

Type 2 SCD

A Type 2 SCD supports versioning of dimension members. Often the source system doesn't store versions, so the data warehouse load process detects and manages changes in a dimension table. In this case, the dimension table must use a surrogate key to provide a unique reference to a version of the dimension member. It also includes columns that define the date range validity of the version (for example, StartDate and EndDate) and possibly a flag column (for example, IsCurrent) to easily filter by current dimension members.

For example, Adventure Works assigns salespeople to a sales region. When a salesperson relocates region, a new version of the salesperson must be created to ensure that historical facts remain associated with the former region. To support accurate historic analysis of sales by salesperson, the dimension table must store versions of salespeople and their associated region(s). The table should also include start and end date values to define the time validity. Current versions may define an

empty end date (or 12/31/9999), which indicates that the row is the current version. The table must also define a surrogate key because the business key (in this instance, employee ID) won't be unique.

SalesRepID	RepSourceId	FirstName	LastName	Region	StartDate	EndDate	IsCurrent
1	312	Jun	Cao	Southwest	2021-03-20	9999-12-31	True
2	331	Susan	Eaton	Southcentral	2021-03-20	9999-12-31	True

SalesRepID	RepSourceId	FirstName	LastName	Region	StartDate	EndDate	IsCurrent
1	312	Jun	Cao	Southwest	2021-03-20	9999-12-31	True
2	331	Susan	Eaton	Southcentral	2021-03-20	2021-03-21	False
3	331	Susan	Eaton	Southeast	2021-03-22	9999-12-31	True

It's important to understand that when the source data doesn't store versions, you must use an intermediate system (like a data warehouse) to detect and store changes. The table load process must preserve existing data and detect changes. When a change is detected, the table load process must expire the current version. It records these changes by updating the EndDate value and inserting a new version with the StartDate value commencing from the previous EndDate value. Also, related facts must use a time-based lookup to retrieve the dimension key value relevant to the fact date.

Type 3 SCD

A Type 3 SCD supports storing two versions of a dimension member as separate columns. The table includes a column for the current value of a member plus either the original or previous value of the member. So Type 3 uses additional columns to track one key instance of history, rather than storing additional rows to track each change like in a Type 2 SCD.

This type of tracking may be used for one or two columns in a dimension table. It is not common to use it for many members of the same table. It is often used in combination with Type 1 or Type 2 members.

CustomerID	FirstName	LastName	CurrentEmail	OriginalEmail	CompanyName	InsertedDate	ModifiedDate
2	Keith	Harris	keith0@aw.com	keith0@aw.com	Progressive Sports	2021-03-20	2021-03-20
3	Donna	Carreras	donna0@aw.com	donna0@aw.com	A Bike Store	2021-03-20	2021-03-20

CustomerID	FirstName	LastName	CurrentEmail	OriginalEmail	CompanyName	InsertedDate	ModifiedDate
2	Keith	Harris	keith0@aw.com	keith0@aw.com	Progressive Sports	2021-03-20	2021-03-20
3	Donna	Carreras	dc3@aw.com	donna0@aw.com	A Bike Store	2021-03-20	2021-03-22

Type 6 SCD

A Type 6 SCD combines Type 1, 2, and 3. When a change happens to a Type 2 member you create a new row with appropriate StartDate and EndDate. In Type 6 design you also store the current value in all versions of that entity so you can easily report on the current value or the historical value.

Using the sales region example, you split the Region column into CurrentRegion and HistoricalRegion. The CurrentRegion always shows the latest value and the HistoricalRegion shows the region that was valid between the StartDate and EndDate. So for the same salesperson, every record would have the latest region populated in CurrentRegion while HistoricalRegion works exactly like the region field in the Type 2 SCD example.

SalesRepID	RepSourceId	FirstName	LastName	CurrentRegion	HistoricalRegion	StartDate	EndDate	IsCurrent
1	312	Jun	Cao	Southwest	Southwest	2021-03-20	9999-12-31	True
2	331	Susan	Eaton	Southcentral	Southcentral	2021-03-20	9999-12-31	True

SalesRepID	RepSourceId	FirstName	LastName	CurrentRegion	HistoricalRegion	StartDate	EndDate	IsCurrent
1	312	Jun	Cao	Southwest	Southwest	2021-03-20	9999-12-31	True
2	331	Susan	Eaton	Southeast	Southcentral	2021-03-20	2021-03-21	False
3	331	Susan	Eaton	Southeast	Southeast	2021-03-22	9999-12-31	True

https://learn.microsoft.com/en-us/sql/integration-services/data-flow/transformations/slowly-changing-dimension-columns-slowly-changing-dimension-wizard?view=sql-server-ver16

Question 15: **Correct**

Scenario: You are working at a bank setting up a database which will be used by all employee-levels of the bank. At the moment, you are setting up permissions for service representatives in a call centre.

Often, due to compliance, the caller has to identify themselves by giving them the last four digits of their credit card number that they may have an issue with. These data items cannot be fully exposed to the service representative in that call centre.

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

- **Row-level security**
- **Column-level security**
- **Table-level security**
- **Dynamic Data Masking**

  (Correct)

Explanation
If you would define a masking rule, that masks all but the last four digits for example of that credit card number, you would get a query that only gives as a result the last four digits of the credit card number.

Dynamic Data Masking

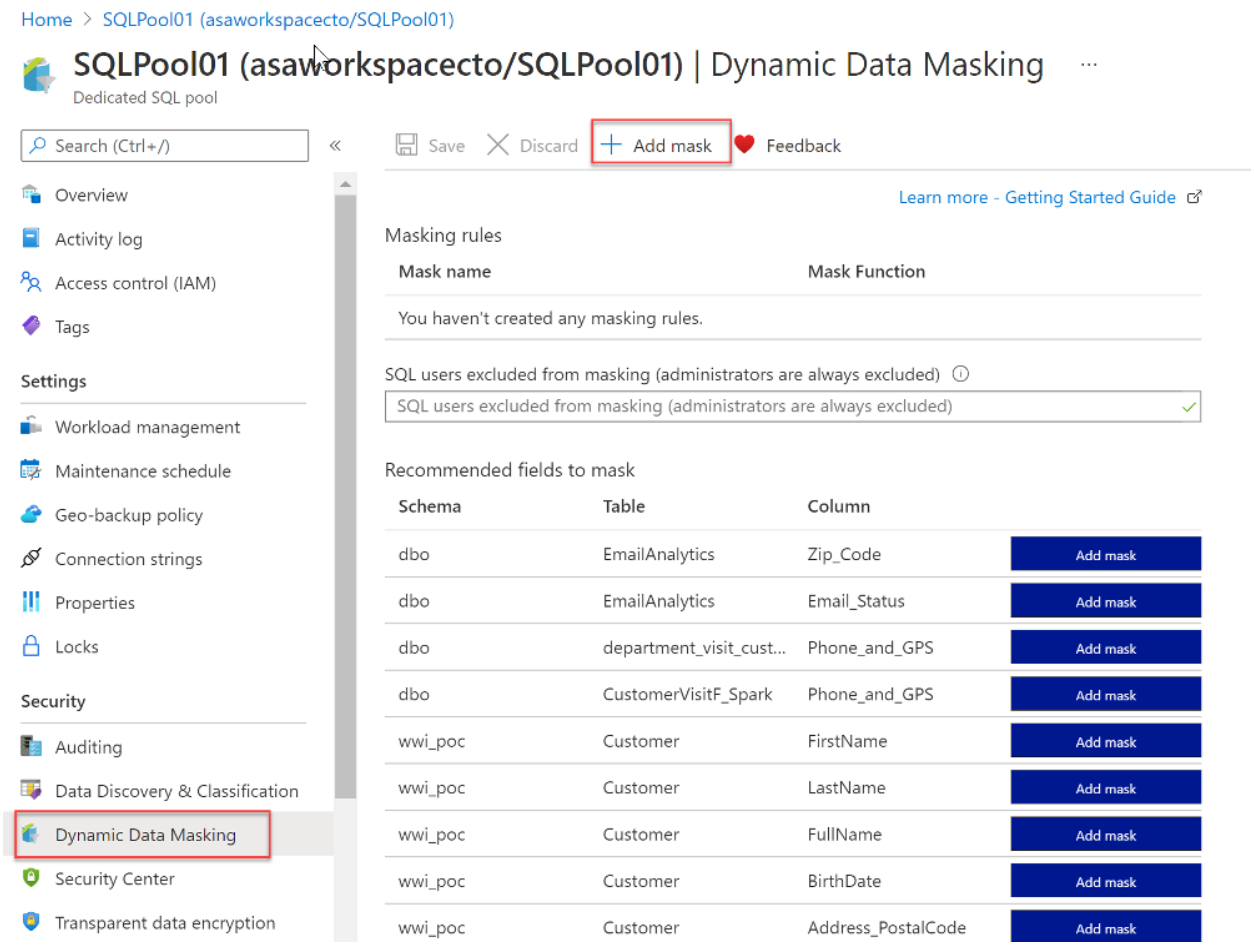
Azure SQL Database, Azure SQL Managed Instance, and Azure Synapse Analytics support Dynamic Data Masking. It's all in the name, Dynamic Data Masking is masking and ensures limited data exposure to non-privileged users, such that they can't see it. It also helps you in preventing unauthorized access to sensitive data. The way Dynamic Data Masking does it, is helping customers to designate how much of the sensitive data to reveal such that it has minimal impact on the application layer. Dynamic Data Masking is a policy-based security feature. It will hide the sensitive data in a result set of a query that runs over designated database fields. However, the data in the database will not be changed.

Let's give you an example how it works. Let's say you work at a bank as a service representative in a call centre. Sometime, due to compliance, the caller has to identify themselves by giving them several digits of their credit card number that they might

have an issue with. However, these data items, should not be fully exposed to the service representative in that call centre, answering the call. If you would define a masking rule, that masks all but the last four digits for example of that credit card number, you would get a query that only gives as a result the last four digits of the credit card number.
If the caller, for example, also had to provide the representative with personal information, that should not be seen by the developer that can query the production environments in order to troubleshoot, you should appropriately mask data in order to protect the given personal data such that compliance is not violated.

For Azure Synapse Analytics, the way to set up a Dynamic Data Masking policy is using PowerShell or the REST API. Bear in mind that it won't be possible for Azure Synapse Analytics to set the Dynamic Data Masking policy in the Azure portal through selecting the Dynamic Data Masking page under Security in the SQL DB configuration pane. You need to set it up using PowerShell or REST API as mentioned before. However, the configuration of the Dynamic Data Masking policy can be done by the Azure SQL Database admin, server admin, or SQL Security Manager roles.

In Azure Synapse Analytics, you can find Dynamic Data Masking here.



Looking into Dynamic Data Masking Policies:

• **SQL users are excluded from masking**

A couple of SQL users or Azure AD identities can get unmasked data in the SQL query results. Users with administrator privileges are always excluded from masking, and see the original data without any mask.

• **Masking rules** - Masking rules are a set of rules that define the designated fields to be masked including the masking function that is used. The designated fields can be defined using a database schema name, table name, and column name.

• **Masking functions** - Masking functions are a set of methods that control the exposure of data for different scenarios.

Dynamic Data Masking for your database in Azure Synapse Analytics using PowerShell cmdlets

• Data masking policies

• `Get-AzSqlDatabaseDataMaskingPolicy`

The `Get-AzSqlDatabaseDataMaskingPolicy` gets the data masking policy for a database.

The syntax for the `Get-AzSqlDatabaseDataMaskingPolicy` in PowerShell is as follows:

```
1. PowerShell
2. Get-AzSqlDatabaseDataMaskingPolicy [-ServerName] <String> [-DatabaseName]
   <String>
3. [-ResourceGroupName] <String> [-DefaultProfile <IAzureContextContainer>] [-
   WhatIf] [-Confirm]
4. [<CommonParameters>]
```

What the `Get-AzSqlDatabaseDataMaskingPolicy` cmdlet does, is getting the data masking policy of an Azure SQL database.

To use this cmdlet in PowerShell, you'd have to specify the following parameters to identify the database:

• *ResourceGroupName*: name of the resource group you deployed the database in

• *ServerName*: sql server name

• *DatabaseName* : name of the database

This cmdlet is also supported by the SQL Server Stretch Database service on Azure.

• `Set-AzSqlDatabaseDataMaskingPolicy`

The `Set-AzSqlDatabaseDataMaskingPolicy` sets data masking for a database.

The syntax for the `Set-AzSqlDatabaseDataMaskingPolicy` in PowerShell is as follows:

```
1. PowerShell
2. Set-AzSqlDatabaseDataMaskingPolicy [-PassThru] [-PrivilegedUsers <String>] [-
   DataMaskingState <String>]
3. [-ServerName] <String> [-DatabaseName] <String> [-ResourceGroupName] <String>
4. [-DefaultProfile <IAzureContextContainer>] [-WhatIf] [-Confirm]
   [<CommonParameters>]
```

What the `Set-AzSqlDatabaseDataMaskingPolicy` cmdlet does is setting the data masking policy for an Azure SQL database.

To use this cmdlet in PowerShell, you'd have to specify the following parameters to identify the database:

•*ResourceGroupName*: name of the resource group that you deployed the database in

•*ServerName* : sql server name

•*DatabaseName* : name of the database

In addition, you will need to set the *DataMaskingState* parameter to specify whether data masking operations are enabled or disabled.

If the cmdlet succeeds and the *PassThru* parameter is used, it will return an object describing the current data masking policy in addition to the database identifiers.

Database identifiers can include, **ResourceGroupName**, **ServerName**, and **DatabaseName**.

This cmdlet is also supported by the SQL Server Stretch Database service on Azure.

• Data masking rules

• `Get-AzSqlDatabaseDataMaskingRule`

The `Get-AzSqlDatabaseDataMaskingRule` Gets the data masking rules from a database.

The syntax for the `Get-AzSqlDatabaseDataMaskingRule` in PowerShell is as follows:

```
1. PowerShell
2. Get-AzSqlDatabaseDataMaskingRule [-SchemaName <String>] [-TableName <String>]
   [-ColumnName <String>]
3. [-ServerName] <String> [-DatabaseName] <String> [-ResourceGroupName] <String>
4. [-DefaultProfile <IAzureContextContainer>] [-WhatIf] [-Confirm]
   [<CommonParameters>]
```

What the `Get-AzSqlDatabaseDataMaskingRule` cmdlet does it getting either a specific data masking rule or all of the data masking rules for an Azure SQL database.

To use the cmdlet in PowerShell, you'd have to specify the following parameters to identify the database:

To use this cmdlet in PowerShell, you'd have to specify the following parameters to identify the database:

• *ResourceGroupName*: name of the resource group that you deployed the database in

• *ServerName* : sql server name

• *DatabaseName* : name of the database

You'd also have to specify the *RuleId* parameter to specify which rule this cmdlet returns.

If you do not provide *RuleId*, all the data masking rules for that Azure SQL database are returned.

This cmdlet is also supported by the SQL Server Stretch Database service on Azure.

• `New-AzSqlDatabaseDataMaskingRule`

The `New-AzSqlDatabaseDataMaskingRule` creates a data masking rule for a database.

The syntax for the `New-AzSqlDatabaseDataMaskingRule` in PowerShell is as follows:

```
1. PowerShell
2. New-AzSqlDatabaseDataMaskingRule -MaskingFunction <String> [-PrefixSize
   <UInt32>] [-ReplacementString <String>]
3. [-SuffixSize <UInt32>] [-NumberFrom <Double>] [-NumberTo <Double>] [-PassThru]
   -SchemaName <String>
4. -TableName <String> -ColumnName <String> [-ServerName] <String> [-
   DatabaseName] <String>
5. [-ResourceGroupName] <String> [-DefaultProfile <IAzureContextContainer>] [-
   WhatIf] [-Confirm]
6. [<CommonParameters>]
```

What the `New-AzSqlDatabaseDataMaskingRule` cmdlet does is creating a data masking rule for an Azure SQL database.

To use this cmdlet in PowerShell, you'd have to specify the following parameters to identify the rule:

• *ResourceGroupName*: name of the resource group that you deployed the database in

• *ServerName* : sql server name

• *DatabaseName* : name of the database

Providing the *TableName* and *ColumnName* is necessary in order to specify the target of the rule.

The *MaskingFunction* parameter is necessary to define how the data is masked.

If *MaskingFunction* has a value of Number or Text, you can specify the *NumberFrom* and *NumberTo* parameters, for number masking, or the *PrefixSize*, *ReplacementString*, and *SuffixSize* for text masking.

If the command succeeds and the *PassThru* parameter is used, the cmdlet returns an object describing the data masking rule properties in addition to the rule identifiers.

Rule identifiers can be, for example, *ResourceGroupName*, *ServerName*, *DatabaseName*, and *RuleID*.

This cmdlet is also supported by the SQL Server Stretch Database service on Azure.

• `Remove-AzSqlDatabaseDataMaskingRule`

The `Remove-AzSqlDatabaseDataMaskingRule` removes a data masking rule from a database.

The syntax for the `Remove-AzSqlDatabaseDataMaskingRule` in PowerShell is as follows:

```
1. PowerShell
2. Remove-AzSqlDatabaseDataMaskingRule [-PassThru] [-Force] -SchemaName <String>
   -TableName <String>
3. -ColumnName <String> [-ServerName] <String> [-DatabaseName] <String> [-
   ResourceGroupName] <String>
4. [-DefaultProfile <IAzureContextContainer>] [-WhatIf] [-Confirm]
   [<CommonParameters>]
```

What the `Remove-AzSqlDatabaseDataMaskingRule` cmdlet does, is it removes a specific data masking rule from an Azure SQL database.

To use this cmdlet in PowerShell, you'd have to specify the following parameters to identify the rule that needs to be removed:

• *ResourceGroupName*: name of the resource group that you deployed the database in

• *ServerName* : sql server name

• *DatabaseName* : name of the database

• *RuleId* : identifier of the rule

This cmdlet is also supported by the SQL Server Stretch Database service on Azure.

• `Set-AzSqlDatabaseDataMaskingRule`

The `Set-AzSqlDatabaseDataMaskingRule` Sets the properties of a data masking rule for a database.

The syntax for the `Set-AzSqlDatabaseDataMaskingRule` in PowerShell is as follows:

```
1. PowerShell
2. Set-AzSqlDatabaseDataMaskingRule [-MaskingFunction <String>] [-PrefixSize
   <UInt32>]
3. [-ReplacementString <String>] [-SuffixSize <UInt32>] [-NumberFrom <Double>] [-
   NumberTo <Double>] [-PassThru]
```

```
4. -SchemaName <String> -TableName <String> -ColumnName <String> [-ServerName]
   <String> [-DatabaseName] <String>
5. [-ResourceGroupName] <String> [-DefaultProfile <IAzureContextContainer>] [-
   WhatIf] [-Confirm]
6. [<CommonParameters>]
```

What the `Set-AzSqlDatabaseDataMaskingRule` cmdlet does is setting a data masking rule for an Azure SQL database.

To use this cmdlet in PowerShell, you'd have to specify the following parameters to identify the rule:

• *ResourceGroupName*: name of the resource group that you deployed the database in

• *ServerName* : sql server name

• *DatabaseName* : name of the database

• *RuleId* : identifier of the rule

You can provide any of the parameters of *SchemaName*, *TableName*, and *ColumnName* to retarget the rule.

Specify the *MaskingFunction* parameter to modify how the data is masked.

If you specify a value of Number or Text for *MaskingFunction*, you can specify the *NumberFrom* and *NumberTo* parameters for number masking or the *PrefixSize*, *ReplacementString*, and *SuffixSize* parameters for text masking.

If the command succeeds, and if you specify the *PassThru* parameter, the cmdlet returns an object that describes the data masking rule properties and the rule identifiers.

Rule identifiers can be, **ResourceGroupName**, **ServerName**, **DatabaseName**, and **RuleId**.

This cmdlet is also supported by the SQL Server Stretch Database service on Azure.

Set up Dynamic Data Masking for your database in Azure Synapse Analytics using the REST API

For setting up Dynamic Data Masking in Azure Synapse Analytics, the other possibility is make use of the REST API.

It will enable to programmatically manage data masking policy and rules.

The REST API will support the following operations:

• Data masking policies

• Create Or Update

The Create Or Update masking policy using the REST API will create or update a database data masking policy.

In HTTP the following request can be made:

```
1. HTTP
2. PUT
   https://management.azure.com/subscriptions/{subscriptionId}/resourceGroups/{re
   sourceGroupName}/providers/Microsoft.Sql/servers/{serverName}/databases/{datab
   aseName}/dataMaskingPolicies/Default?api-version=2014-04-01
```

The following parameters need to be passed through:

• *SubscriptionID*: the ID of the subscription

• *ResourceGroupName*: name of the resource group that you deployed the database in

• *ServerName* : sql server name

• *DatabaseName* : name of the database

• *dataMaskingPolicyName*: the name of the data masking policy

• *api version*: version of the api that is used.

• Get

The Get policy, Gets a database data masking policy.

In HTTP the following request can be made:

```
1. HTTP
2. GET
   https://management.azure.com/subscriptions/{subscriptionId}/resourceGroups/{re
   sourceGroupName}/providers/Microsoft.Sql/servers/{serverName}/databases/{datab
   aseName}/dataMaskingPolicies/Default?api-version=2014-04-01
```

The following parameters need to be passed through:

• *SubscriptionID*: the ID of the subscription

• *ResourceGroupName*: name of the resource group that you deployed the database in

• *ServerName* : sql server name

• *DatabaseName* : name of the database

• *dataMaskingPolicyName*: the name of the data masking policy

• *api version*: version of the api that is used.

• Data masking rules

• Create Or Update

The Create or Update masking rule creates or updates a database data masking rule.

In HTTP the following request can be made:

```
1. HTTP
2. PUT
   https://management.azure.com/subscriptions/{subscriptionId}/resourceGroups/{re
   sourceGroupName}/providers/Microsoft.Sql/servers/{serverName}/databases/{datab
   aseName}/dataMaskingPolicies/Default/rules/{dataMaskingRuleName}?api-
   version=2014-04-01
```

The following parameters need to be passed through:

• *SubscriptionID*: the ID of the subscription

• *ResourceGroupName*: name of the resource group that you deployed the database in

• *ServerName* : sql server name

• *DatabaseName* : name of the database

• *dataMaskingPolicyName*: the name of the data masking policy

• *dataMaskingRuleName*: the name of the rule for data masking

• *api version*: version of the api that is used.

• List By Database

The List By Database request gets a list of database data masking rules.

In HTTP the following request can be made:

```
1. HTTP
2. GET
   https://management.azure.com/subscriptions/{subscriptionId}/resourceGroups/{re
   sourceGroupName}/providers/Microsoft.Sql/servers/{serverName}/databases/{datab
   aseName}/dataMaskingPolicies/Default/rules?api-version=2014-04-01
```

The following parameters need to be passed through:

• *SubscriptionID*: the ID of the subscription

• *ResourceGroupName*: name of the resource group that you deployed the database in

• *ServerName* : sql server name

• *DatabaseName* : name of the database

• *dataMaskingPolicyName*: the name of the data masking policy

• *api version*: version of the api that is used.

https://docs.microsoft.com/en-us/azure/azure-sql/database/dynamic-data-masking-overview

Question 16: **Incorrect**
With the wide range of data stores available in Azure, there's the need to manage and orchestrate the movement data between them. In fact, you'll usually want to automate *extract, transform, and load* (ETL) workloads as a regular process in a wider enterprise analytical solution. *Pipelines* are a mechanism for defining and orchestrating data movement activities.

What kind of object should you add to a data flow to define a target to which data is loaded?

- **Transformation**
- **Source**
- **Faucet**
- **Sink**

(Correct)

-

Drain

(Incorrect)

-

Supply line

Explanation

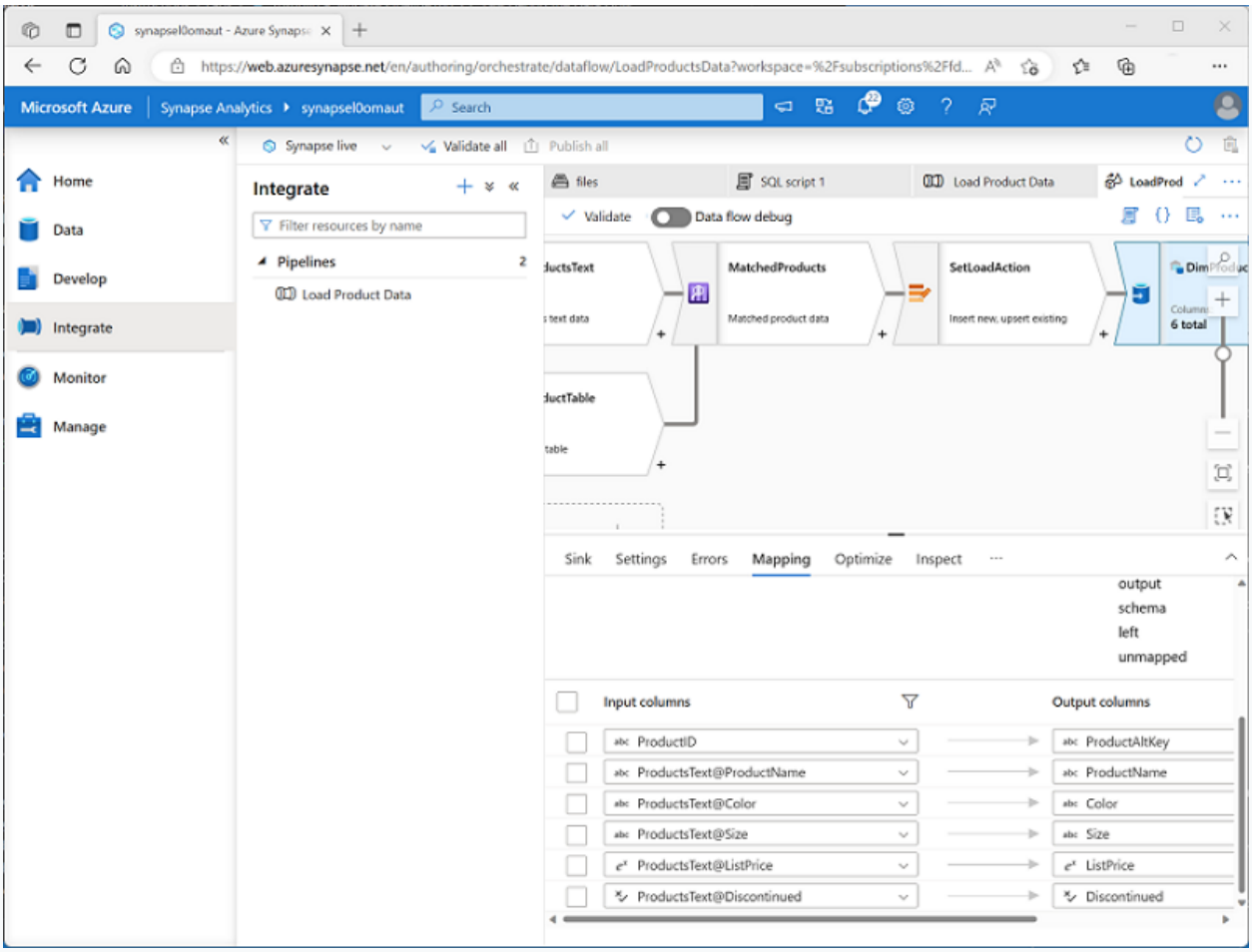
- **A sink represents a target in a data flow.**
- A source provides the input data for a data flow.
- You can use various transformations to modify data in a data flow.
- Faucet, Drain, and Supply line are not valid options.

Define data flows

A **Data Flow** is a commonly used activity type to define data flow and transformation. Data flows consist of:

- **Sources** - The input data to be transferred.
- **Transformations** – Various operations that you can apply to data as it streams through the data flow.
- **Sinks** – Targets into which the data will be loaded.

When you add a **Data Flow** activity to a pipeline, you can open it in a separate graphical design interface in which to create and configure the required data flow elements.



An important part of creating a data flow is to define mappings for the columns as the data flows through the various stages, ensuring column names and data types are defined appropriately. While developing a data flow, you can enable the **Data flow debug** option to pass a subset of data through the flow, which can be useful to test that your columns are mapped correctly.

https://learn.microsoft.com/en-us/azure/data-factory/control-flow-execute-data-flow-activity

Question 17: **Correct**
What happens if the command option ("checkpointLocation", pointer-to-checkpoint directory) is not specified in Structured Streaming?

- ⟳

When the streaming job stops, all state around the streaming job dumped to a default location, and upon restart, the job must start from aggregated data rather than tuned specific data.

- **It will not be possible to create more than one streaming query that uses the same streaming source since they will conflict.**

- **The streaming job will function as expected since the checkpointLocation option does not exist.**

- **When the streaming job stops, all state data around the streaming job is lost, and upon restart, the job must start from scratch.**

  (Correct)

Explanation
Setting the checkpointLocation is required for many sinks used in Structured Streaming. For those sinks where this setting is optional, keep in mind that when you do not set this value, you risk losing your place in the stream.

https://www.waitingforcode.com/apache-spark-structured-streaming/checkpoint-storage-structured-streaming/read

Question 18: **Incorrect**
Scenario: Big Belly Foods, Inc. (BB) owns and operates 300 convenience stores across LatAm. The company sells a variety of packaged foods and drinks, as well as a variety of prepared foods, such as sandwiches and pizzas. The company has a loyalty club whereby members can get daily discounts on specific items by providing their membership number at checkout.

BB employs business analysts who prefer to analyze data by using Microsoft Power BI, and data scientists who prefer analyzing data in Azure Databricks notebooks. You have been hired as an Azure Expert SME and you are to consult the IT team on various Azure related projects.

Business Requirements:

BB wants to create a new analytics environment in Azure to meet the following requirements:

• See inventory levels across the stores. Data must be updated as close to real time as possible.

• Execute ad hoc analytical queries on historical data to identify whether the loyalty club discounts increase sales of the discounted products.

• Every four hours, notify store employees about how many prepared food items to produce based on historical demand from the sales data.

Technical Requirements:

BB identifies the following technical requirements:

• Minimize the number of different Azure services needed to achieve the business goals.

• Use platform as a service (PaaS) offerings whenever possible and avoid having to provision virtual machines that must be managed by BB.

• Ensure that the analytical data store is accessible only to the company's on-premises network and Azure services.

• Use Azure Active Directory (Azure AD) authentication whenever possible.

• Use the principle of least privilege when designing security.

• Stage Inventory data in Azure Data Lake Storage Gen2 before loading the data into the analytical data store. BB wants to remove transient data from Data

• Lake Storage once the data is no longer in use. Files that have a modified date that is older than 14 days must be removed.

• Limit the business analysts' access to customer contact information, such as phone numbers, because this type of data is not analytically relevant.

• Ensure that you can quickly restore a copy of the analytical data store within one hour in the event of corruption or accidental deletion.

Planned Environment:

BB plans to implement the following environment:

• The application development team will create an Azure event hub to receive real-time sales data, including store number, date, time, product ID, customer loyalty number, price, and discount amount, from the point of sale (POS) system and output the data to data storage in Azure.

• Customer data, including name, contact information, and loyalty number, comes from Salesforce, a SaaS application, and can be imported into Azure once every eight hours. Row modified dates are not trusted in the source table.

• Product data, including product ID, name, and category, comes from Salesforce and can be imported into Azure once every eight hours. Row modified dates are not trusted in the source table.

• Daily inventory data comes from a Microsoft SQL server located on a private network.

• BB currently has 5 TB of historical sales data and 100 GB of customer data. The company expects approximately 100 GB of new data per month for the next year.

• BB will build a custom application named FoodPrep to provide store employees with the calculation results of how many prepared food items to produce every four hours.

• BB does not plan to implement Azure ExpressRoute or a VPN between the on-premises network and Azure.

The Ask:

The team looks to you for direction on what should be used to import the daily inventory data from the SQL server to Azure Data Lake Storage. Which Azure Data Factory components should you recommend for the trigger type?

- **Tumbling window trigger**
- **Event-based trigger**
- **Schedule trigger**

  (Correct)

-

  Scaling window trigger

  (Incorrect)

Explanation
The following are the recommends you should present:

• A self-hosted IR is capable of running copy activity between a cloud data stores and a data store in private network.

• Schedule trigger set for an 8 hour interval.

• A copy activity type

Rational:

• Customer data, including name, contact information, and loyalty number, comes from Salesforce and can be imported into Azure once every eight hours. Row modified dates are not trusted in the source table.

• Product data, including product ID, name, and category, comes from Salesforce and can be imported into Azure once every eight hours. Row modified dates are not trusted in the source table.

Create a trigger that runs a pipeline on a schedule

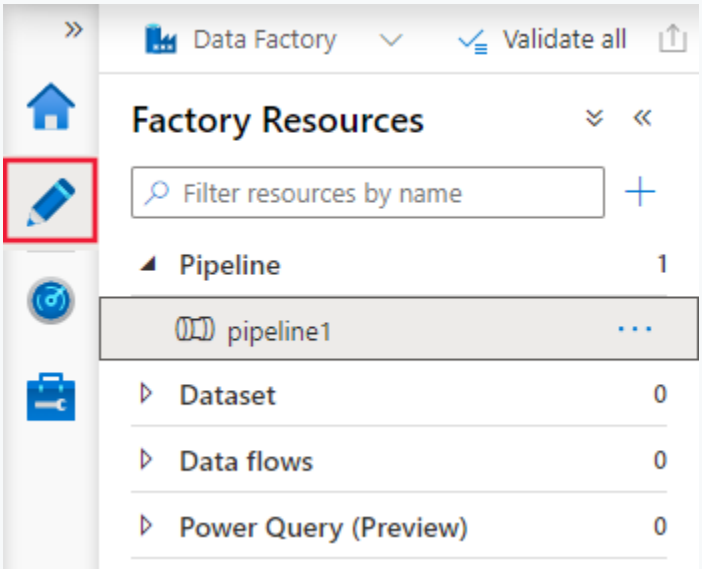
When creating a schedule trigger, you specify a schedule (start date, recurrence, end date etc.) for the trigger, and associate with a pipeline. Pipelines and triggers have a many-to-many relationship. Multiple triggers can kick off a single pipeline. A single trigger can kick off multiple pipelines.

Note: For a complete walkthrough of creating a pipeline and a schedule trigger, which associates the trigger with the pipeline, and runs and monitors the pipeline, see Quickstart: create a data factory using Data Factory UI.

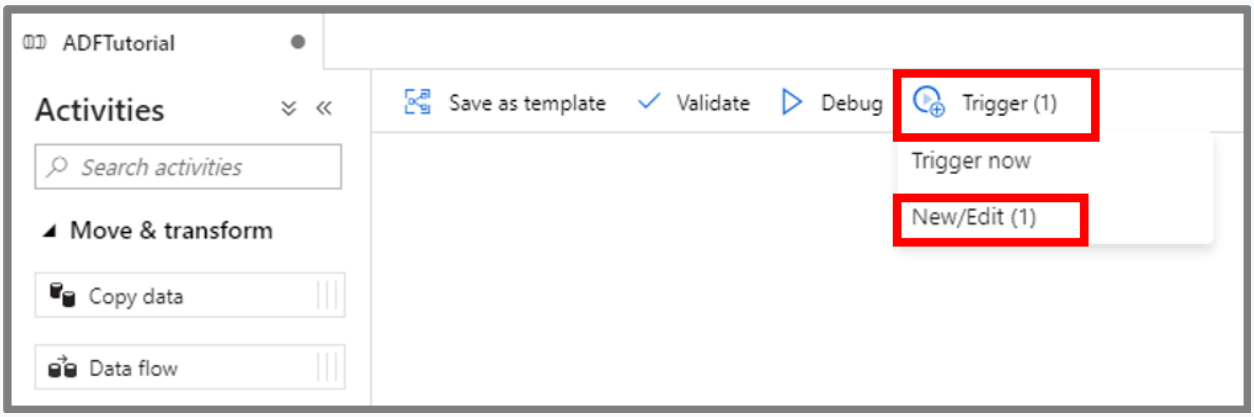
Data Factory UI

You can create a **schedule trigger** to schedule a pipeline to run periodically (hourly, daily, etc.).

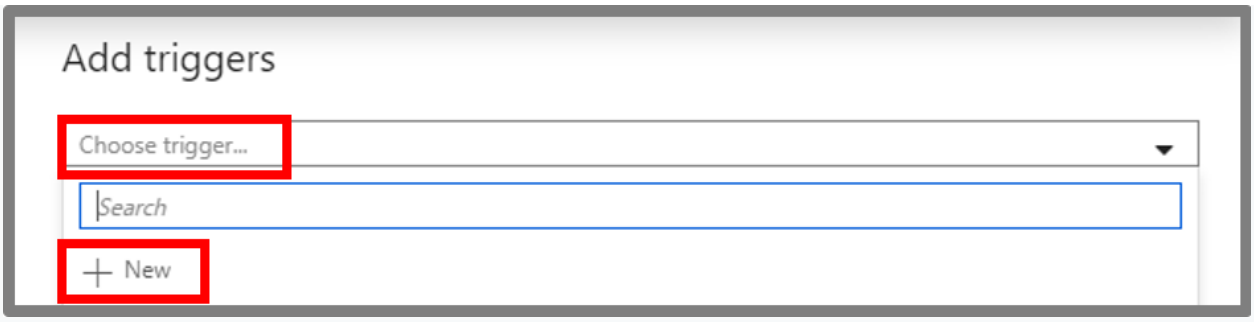
1. Switch to the **Edit** tab, shown with a pencil symbol.



2. Select **Trigger** on the menu, then select **New/Edit**.



3. On the **Add Triggers** page, select **Choose trigger...**, then select **+New**.



4. On the **New Trigger** page, do the following steps:

• Confirm that **Schedule** is selected for **Type**.

• Specify the start datetime of the trigger for **Start Date**. It's set to the current datetime in Coordinated Universal Time (UTC) by default.

• Specify the time zone that the trigger will be created in. The time zone setting will apply to **Start Date**, **End Date**, and **Schedule Execution Times** in Advanced recurrence options. Changing Time Zone setting will not automatically change your start date. Make sure the Start Date is correct in the specified time zone. Please note that Scheduled Execution time of Trigger will be considered post the Start Date (Ensure Start Date is atleast 1minute lesser than the Execution time else it will trigger pipeline in next recurrence).

Note: For time zones that observe daylight saving, trigger time will auto-adjust for the twice a year change. To opt out of the daylight saving change, please select a time zone that does not observe daylight saving, for instance UTC

• Specify **Recurrence** for the trigger. Select one of the values from the drop-down list (Every minute, Hourly, Daily, Weekly, and Monthly). Enter the multiplier in the text box. For example, if you want the trigger to run once for every 15 minutes, you select **Every Minute**, and enter **15** in the text box.

• To specify an end date time, select **Specify an End Date**, and specify *Ends On*, then select **OK**. There is a cost associated with each pipeline run. If you are testing, you may want to ensure that the pipeline is triggered only a couple of times. However, ensure that there is enough time for the pipeline to run between the publish time and the end time. The trigger comes into effect only after you publish the solution to Data Factory, not when you save the trigger in the UI.

New trigger

Name *

trigger4

Description

Type *

Schedule | Tumbling window | Event

Start date *

10/29/2020 3:30 PM

Time zone *

Pacific Time (US & Canada) (UTC-8)

This time zone observes daylight savings. Trigger will auto-adjust for one hour difference.

Recurrence *

Every 15 Minute(s)

Specify an end date

End On *

10/31/2020 6:30 PM

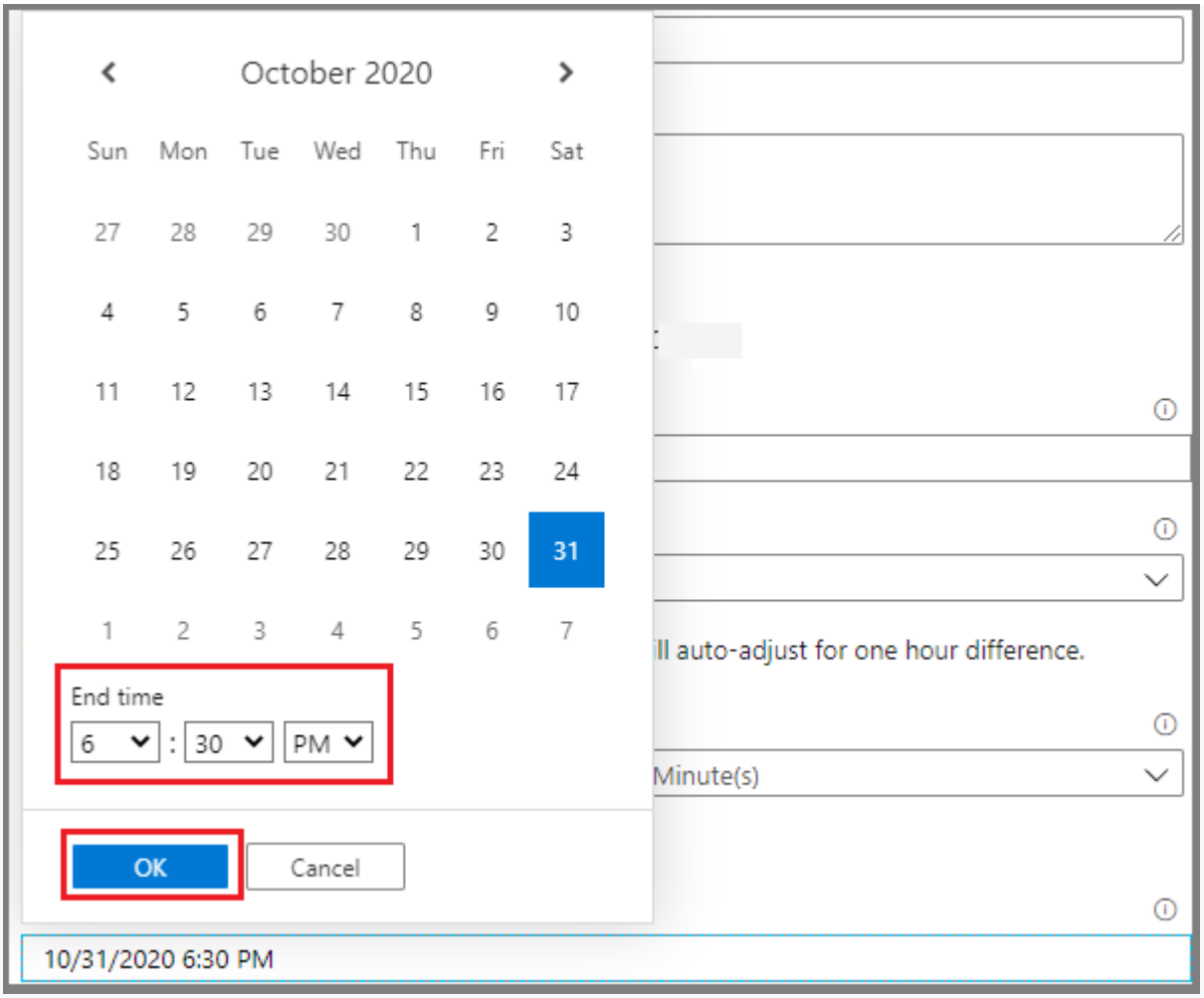
Annotations

+ New

Name

Activated *

Yes | No



5. In the **New Trigger** window, select **Yes** in the **Activated** option, then select **OK**. You can use this checkbox to deactivate the trigger later.

New trigger

Name *

trigger4

Description

Type *

Schedule Tumbling window Event

Start date *

10/29/2020 3:30 PM

Time zone *

Pacific Time (US & Canada) (UTC-8)

This time zone observes daylight savings. Trigger will auto-adjust for one hour difference.

Recurrence *

Every 15 Minute(s)

Specify an end date

End On *

10/31/2020 6:30 PM

Annotations

+ New

Name

Activated *

Yes No

OK Cancel

6. In the **New Trigger** window, review the warning message, then select **OK**.

New trigger

Trigger Run Parameters

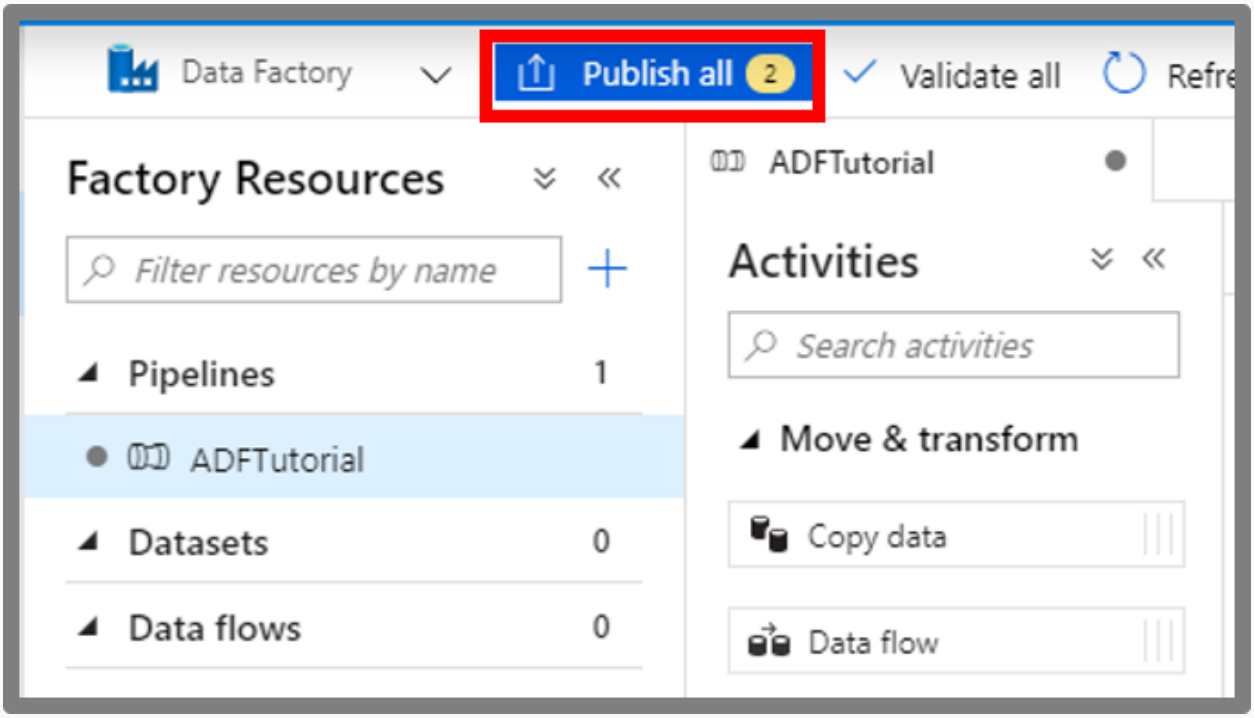
NAME	TYPE	VALUE
This pipeline has no parameters		

Make sure to "Publish" for trigger to be activated after clicking "OK"

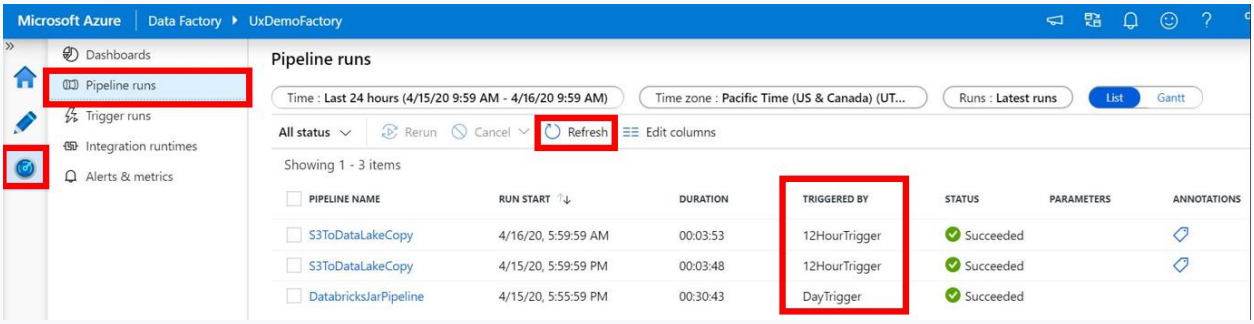
OK

Cancel

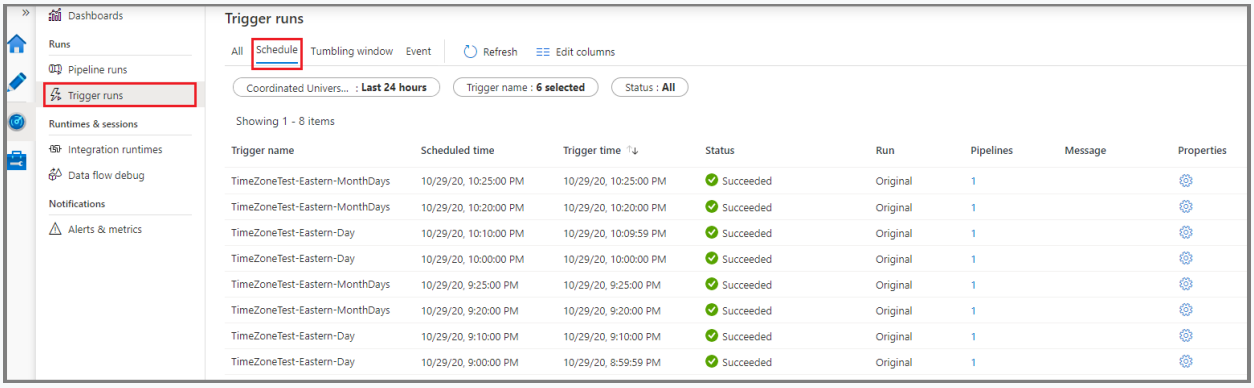
7. Select **Publish all** to publish the changes to Data Factory. Until you publish the changes to Data Factory, the trigger doesn't start triggering the pipeline runs.



8. Switch to the **Pipeline runs** tab on the left, then select **Refresh** to refresh the list. You will see the pipeline runs triggered by the scheduled trigger. Notice the values in the **Triggered By** column. If you use the **Trigger Now** option, you will see the manual trigger run in the list.



9. Switch to the **Trigger Runs \ Schedule** view.



https://docs.microsoft.com/en-us/azure/data-factory/how-to-create-schedule-trigger

Question 19: **Correct**
Identify the missing word(s) in the following sentence within the context of Microsoft Azure.

Azure Databricks is a fully-managed, cloud-based Big Data and Machine Learning platform, which empowers developers to accelerate AI and innovation by simplifying the process of building enterprise-grade production data applications. Built as a joint effort by Databricks and Microsoft, Azure Databricks provides data science and engineering teams with a single platform for Big Data processing and Machine Learning.

A Microsoft-managed Azure Databricks workspace virtual network (VNet) exists within the customer subscription. Information exchanged between this VNet and the Microsoft-managed Azure Databricks Control Plane VNet is sent over a secure TLS connection through ports (22 and 5557) that are enabled by Network Security Groups (NSGs) and protected with port IP filtering.

The Blob Storage account provides default file storage within the workspace (databricks file system (DBFS)). This resource and all other Microsoft-managed resources are completely locked from changes made by the customer.

True or False: You can write to the default DBFS file storage as needed, but you cannot change the Blob Storage account settings.

- False

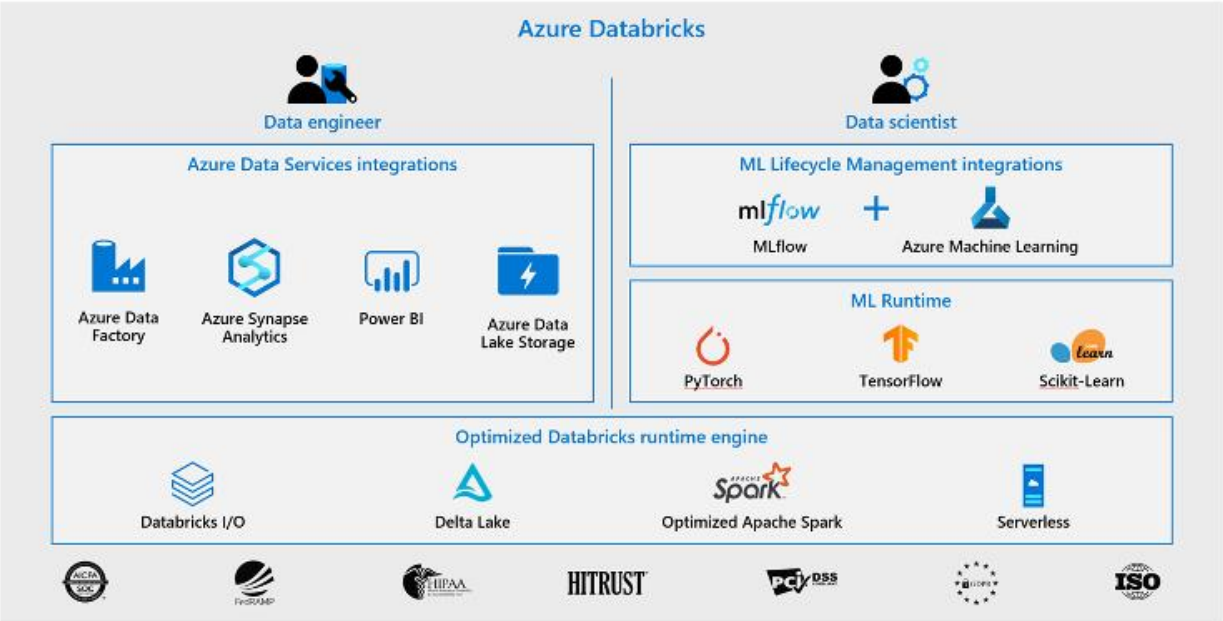
- True

  (Correct)

Explanation
Azure Databricks is a fully-managed, cloud-based Big Data and Machine Learning platform, which empowers developers to accelerate AI and innovation by simplifying the process of building enterprise-grade production data applications. Built as a joint effort by Databricks and Microsoft, Azure Databricks provides data science and engineering teams with a single platform for Big Data processing and Machine Learning.

By combining the power of Databricks, an end-to-end, managed Apache Spark platform optimized for the cloud, with the enterprise scale and security of Microsoft's Azure platform, Azure Databricks makes it simple to run large-scale Spark workloads.

Conceptual view of Azure Databricks



To provide the best platform for data engineers, data scientists, and business users, Azure Databricks is natively integrated with Microsoft Azure, providing a "first party" Microsoft service. The Azure Databricks collaborative workspace enables these teams to work together through features such as user management, git source code repository integration, and user workspace folders.

Microsoft is working to integrate Azure Databricks closely with all features of the Azure platform. Below is a list of some of the integrations completed so far:

• **VM types**: Many existing VMs can be used for clusters, including F-series for machine learning scenarios, M-series for massive memory scenarios, and D-series for general purpose.

• **Security and Privacy**: Ownership and control of data is with the customer, and Microsoft aims for Azure Databricks to adhere to all the compliance certifications that the rest of Azure provides.

• **Flexibility in network topology**: Azure Databricks supports deployments into virtual networks (VNETs), which can control which sources and sinks can be accessed and how they are accessed.

• **Orchestration**: ETL/ELT workflows (including analytics workloads in Azure Databricks) can be operationalized using Azure Data Factory pipelines.

• **Power BI**: Power BI can be connected directly to Databricks clusters using JDBC in order to query data interactively at massive scale using familiar tools.

• **Azure Active Directory**: Azure Databricks workspaces deploy into customer subscriptions, so naturally AAD can be used to control access to sources, results, and jobs.

• **Data stores**: Azure Storage and Data Lake Store services are exposed to Databricks users via Databricks File System (DBFS) to provide caching and optimized analysis over existing data. Azure Databricks easily and efficiently uploads results into Azure Synapse Analytics, Azure SQL Database, and Azure Cosmos DB for further analysis and real-time serving, making it simple to build end-to-end data architectures on Azure.

• **Real-time analytics**: Integration with IoT Hub, Azure Event Hubs, and Azure HDInsight Kafka clusters enables developers to build scalable streaming solutions for real-time analytics.

For developers, this design provides three things. First, it enables easy connection to any storage resources in their account, such as an existing Blob storage or Data Lake Store. Second, they are able to take advantage of deep integrations with other Azure services to quickly build data applications. Third, Databricks is managed centrally from the Azure control centre, requiring no additional setup, which allows developers to focus on core business value, not infrastructure management.

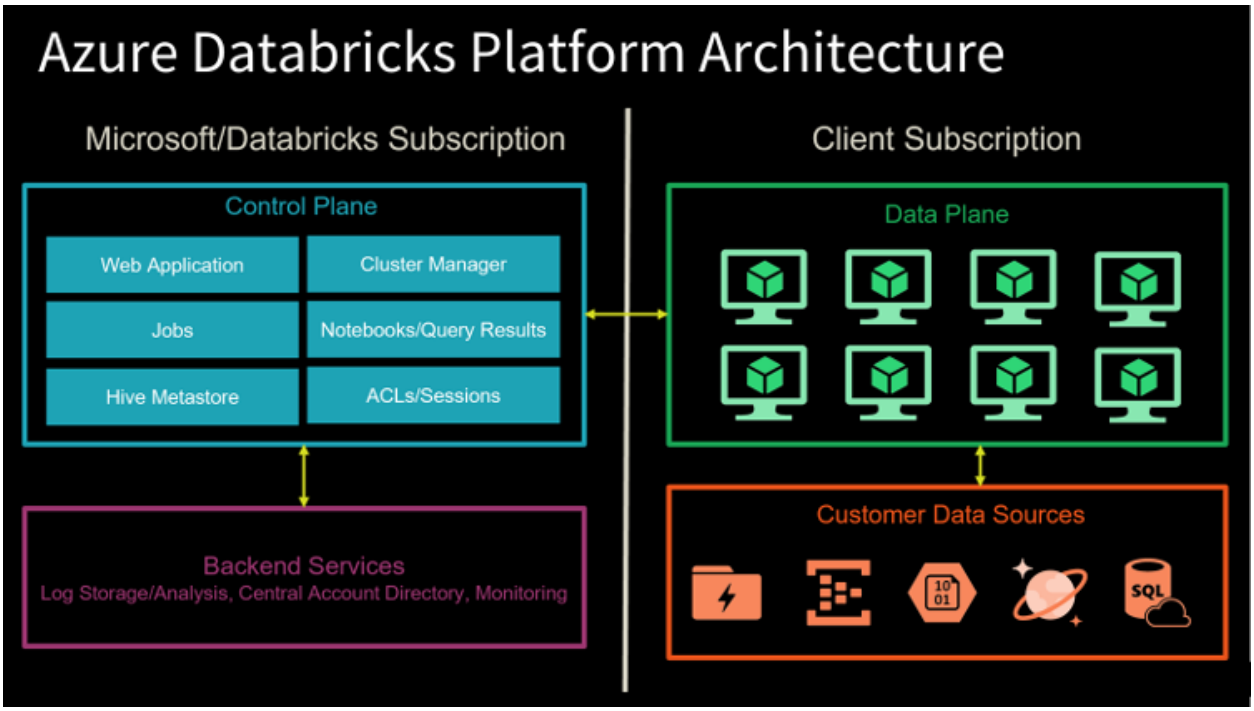
Azure Databricks platform architecture

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.

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.

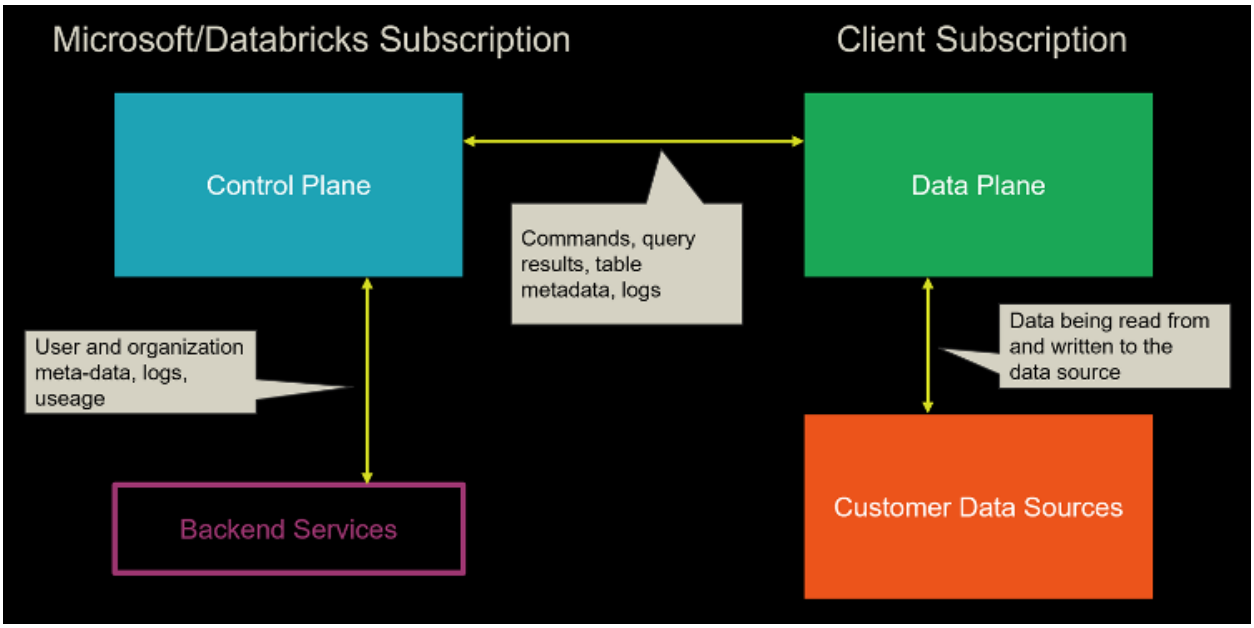
NAME	TYPE	LOCATION
03a67d3205c04e2aa8604531d8946956	Virtual machine	East US 2
03a67d3205c04e2aa8604531d8946956_OsDisk_1_d0553bd1c27948fa90f088b6c3d09251	Disk	East US 2
03a67d3205c04e2aa8604531d8946956-containerRootVolume	Disk	East US 2
03a67d3205c04e2aa8604531d8946956-privateNIC	Network interface	East US 2
03a67d3205c04e2aa8604531d8946956-publicIP	Public IP address	East US 2
03a67d3205c04e2aa8604531d8946956-publicNIC	Network interface	East US 2
2300d7f9bf814f6ea728c4e54032bc2a-containerRootVolume	Disk	East US 2
430185d0fed946e2a9b703bc3bf96f95	Virtual machine	East US 2
430185d0fed946e2a9b703bc3bf96f95_OsDisk_1_c5831ef3af08415795e1b879402bfd29	Disk	East US 2
430185d0fed946e2a9b703bc3bf96f95-containerRootVolume	Disk	East US 2
430185d0fed946e2a9b703bc3bf96f95-privateNIC	Network interface	East US 2
430185d0fed946e2a9b703bc3bf96f95-publicIP	Public IP address	East US 2
430185d0fed946e2a9b703bc3bf96f95-publicNIC	Network interface	East US 2
dbstoragezkbo4ipeo56z2	Storage account	East US 2
workers-sg	Network security group	East US 2
workers-vnet	Virtual network	East US 2

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 this to further improve Spark performance.

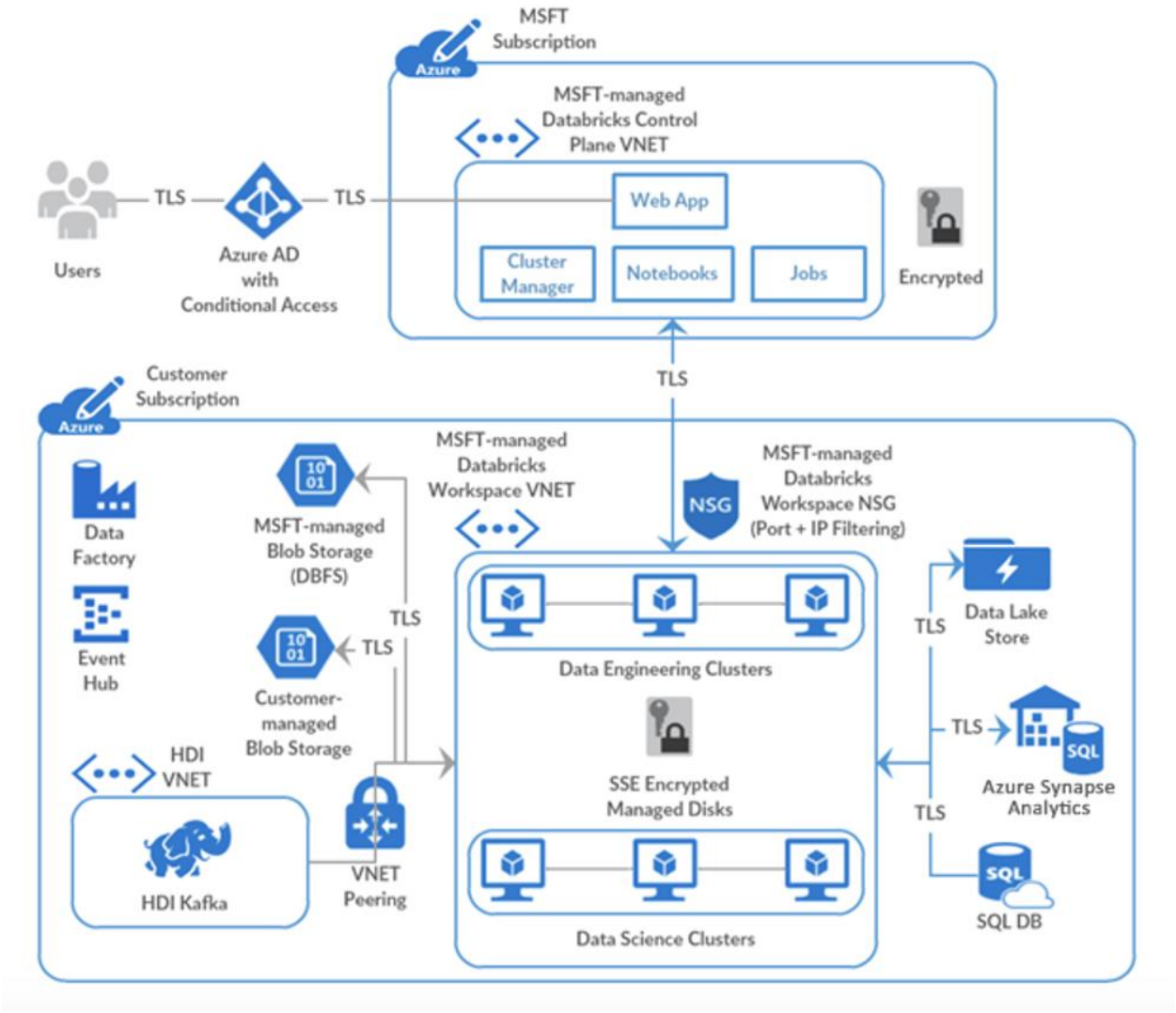


The diagram above shows a Control Plane on the left, which hosts Databricks jobs, notebooks with query results, the cluster manager, web application, Hive metastore, and security access control lists (ACLs) and user sessions. These components are managed by Microsoft in collaboration with Databricks and do not reside within your Azure subscription.

On the right-hand side is the Data Plane, which contains all the Databricks runtime clusters hosted within the workspace. All data processing and storage exists within the client subscription. This means no data processing ever takes place within the Microsoft/Databricks-managed subscription.



Moving one level deeper, the diagram above shows what is being exchanged between the Azure Databricks platform components. Since the web app and cluster manager is part of the Control Plane, any commands executed in a notebook are sent from the cluster manager to the customer's clusters in the Data Plane. This is because the data processing only occurs within the customer's own subscription, as stated earlier. Any table metadata and logs are exchanged between these two high-level components. Customer data sources within the client subscription exchange data with the Data Plane through read and write activities.



The diagram above shows a standard deployment that contains the boundaries between the Control Plane and the Data Plane with the Azure components deployed to each. At the top of the diagram is the Control Plane that exists within the Microsoft subscription. The customer subscription is at the bottom of the diagram, which contains the Data Plane and data sources.

A Microsoft-managed Azure Databricks workspace virtual network (VNet) exists within the customer subscription. Information exchanged between this VNet and the Microsoft-managed Azure Databricks Control Plane VNet is sent over a secure TLS connection through ports (22 and 5557) that are enabled by Network Security Groups (NSGs) and protected with port IP filtering.

The Blob Storage account provides default file storage within the workspace (databricks file system (DBFS)). This resource and all other Microsoft-managed

resources are completely locked from changes made by the customer. All other resources within the customer subscription are customer-managed and can be added or modified per your Azure subscription permissions. Connectivity between these resources and the Databricks clusters that reside within the Data Plane is secured via TLS.

To clarify, you can write to the default DBFS file storage as needed, but you cannot change the Blob Storage account settings since the account is managed by the Microsoft-managed Control Plane. As a best practice, only use the default storage for temporary files and mount additional storage accounts (Blob Storage or Azure Data Lake Storage Gen2) that you create in your Azure subscription, for long-term file storage. This is because the default file storage is tied to the lifecycle of your Azure Databricks account. If you delete the Azure Databricks account, the default storage gets deleted with it.

If you need advanced network connectivity, such as custom VNet peering and VNet injection, you could deploy Azure Databricks Data Plane resources within your own VNet.

https://docs.databricks.com/getting-started/overview.html

Question 20: **Incorrect**
Identify the missing word(s) in the following sentence within the context of Microsoft Azure.

When a table is created, by default the data structure has no indexes and is called a(n) [?].

- **NoMap object**
- **N-tree**
- **Heap**

  (Correct)

Open table

(Incorrect)

Explanation
When a table is created, by default the data structure has no indexes and is called a heap. A well-designed indexing strategy can reduce disk I/O operations and consume less system resources therefore improving query performance, especially when using filtering, scans, and joins in a query.

Dedicated SQL Pools have the following indexing options available:

Clustered columnstore index

Dedicated SQL Pools create a clustered columnstore index when no index options are specified on a table. Clustered columnstore indexes offer both the highest level of data compression as well as the best overall query performance. Clustered columnstore indexes will generally outperform clustered rowstore indexes or heap tables and are usually the best choice for large tables.

Additional compression on the data can be gained also with the index option COLUMNSTORE_ARCHIVE. These reduced sizes allow less memory to be used when accessing and using the data as well as reducing the IOPs required to retrieve data from storage.

Columnstore works on segments of 1,024,000 rows that get compressed and optimized by column. This segmentation further helps to filter out and reduce the data accessed through leveraging metadata stored which summarizes the range and values within each segment during query optimization.

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

Clustered index

Clustered Rowstore Indexes define how the table itself is stored, ordered by the columns used for the Index. There can be only one clustered index on a table.

Clustered indexes are best for queries and joins that require ranges of data to be scanned, preferably in the same order that the index is defined.

Non-clustered index

A non-clustered index can be defined on a table or view with a clustered index or on a heap. Each index row in the non-clustered index contains the non-clustered key value and a row locator. This is a data structure separate/additional to the table or heap. You can create multiple non-clustered indexes on a table.

Non clustered indexes are best used when used for the columns in a join, group by statement or where clauses that return an exact match or few rows.

https://docs.microsoft.com/en-us/sql/t-sql/statements/create-table-azure-sql-data-warehouse?view=aps-pdw-2016-au7

Question 21: **Correct**
True or False: Azure Blob Storage is the least expensive method to store data and one of it best features is that it allows for querying the data directly within the Blob environment.

- **True**
- **False**

  (Correct)

Explanation
Azure Blob Storage Queries

If you create a storage account as a Blob store, you can't query the data directly. To query it, either move the data to a store that supports queries or set up the Azure Storage account for a data lake storage account. Azure Blob storage has no API to query data within the blob - it's just dumb storage. You're essentially limited to reading, deserializing and grabbing your value(s).

https://stackoverflow.com/questions/38721458/query-blobs-in-blob-storage

Question 22: **Correct**
True or False: Mapping data flows are visually displayed data transformations in Azure Data Factory. Data flows allow data engineers to develop data transformation logic with or without writing code.

-

True

False

(Correct)

Explanation

Transforming data with the Mapping Data Flow

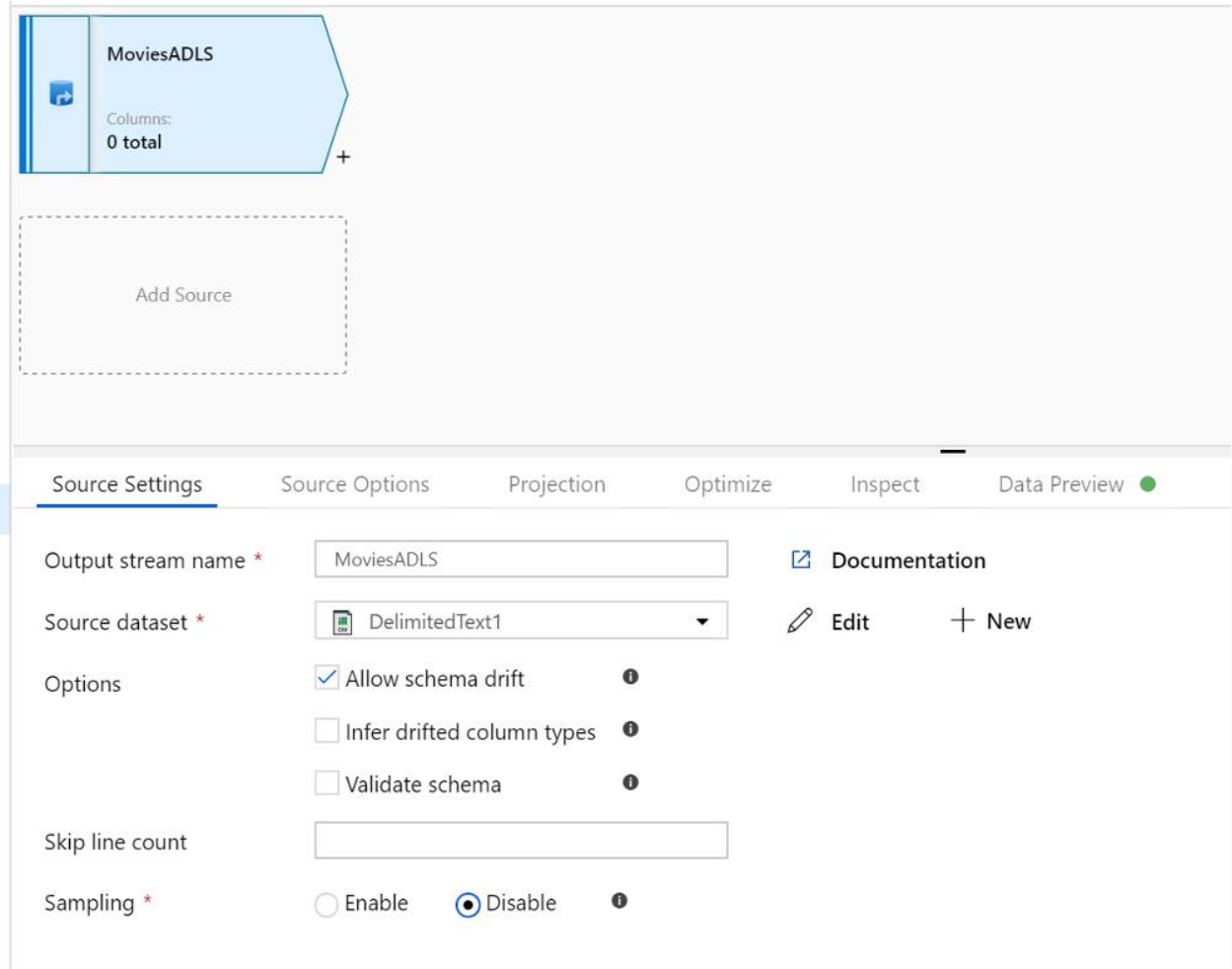
You can natively perform data transformations with Azure Data Factory code free using the Mapping Data Flow task. **Mapping Data Flows provide a fully visual experience with no coding required.** Your data flows will run on your own execution cluster for scaled-out data processing. Data flow activities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

Adding source data to the Mapping Data Flow

Open the Mapping Data Flow canvas. Click on the Add Source button in the Data Flow canvas. In the source dataset dropdown, select your data source, n this case the ADLS Gen2 dataset is used in this example



There are a couple of points to 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)

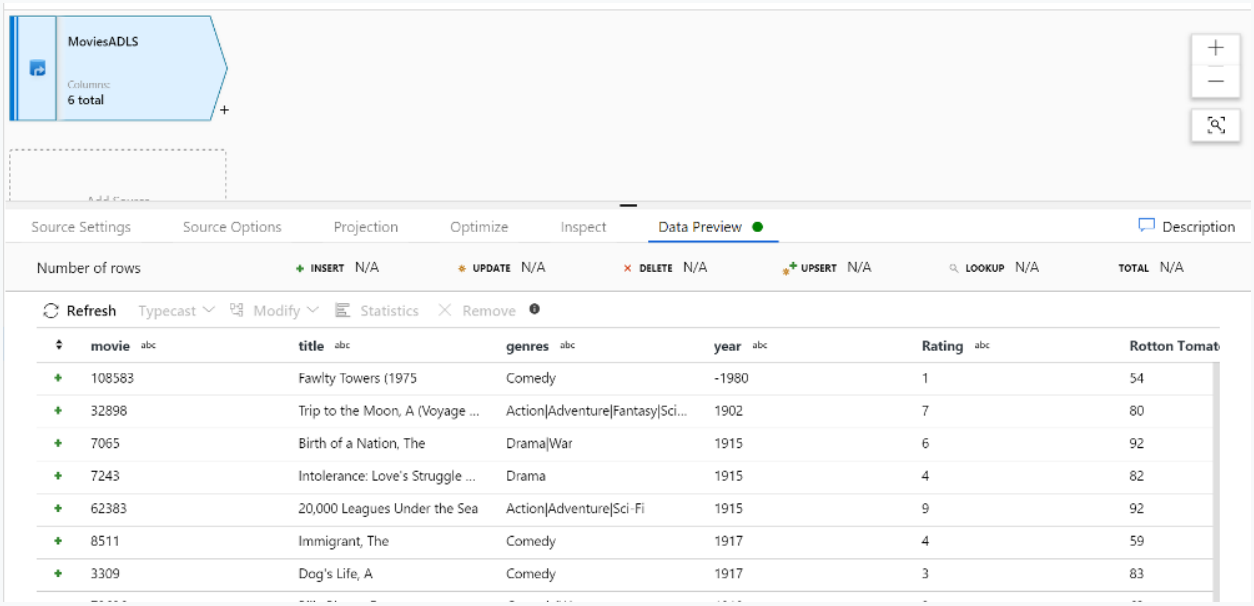
• Azure Synapse Analytics

• 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://docs.microsoft.com/en-us/azure/data-factory/concepts-data-flow-overview

Question 23: **Incorrect**

Dynamic Management Views provide a programmatic experience for monitoring the Azure Synapse Analytics SQL pool activity by using the Transact-SQL language. What type of information or assistance do the views provide? (Select all that apply)

- ☐

  Connection information and activity

  (Correct)

- ☐

  Data movement service activity

  (Correct)

- ☐

  Troubleshoot workload performance bottlenecks

  (Correct)

- ☐

  Resource blocking and locking activity

  (Correct)

- ☐

  Encryption deficiencies

- ☑

  SQL execution requests and queries

  (Correct)

- ☐

  Identify workload performance bottlenecks

  (Correct)

Explanation
Dynamic Management Views provide a programmatic experience for monitoring the Azure Synapse Analytics SQL pool activity by using the Transact-SQL language. The views that are provided, not only enable you to troubleshoot and identify performance bottlenecks with the workloads working on your system, but they are also used by other services such as Azure Advisor to provide recommendations about Azure Synapse Analytics.

There are over 90 Dynamic Management Views that can queried against dedicated SQL pools to retrieve information about the following areas of the service:

• Connection information and activity

• SQL execution requests and queries

• Index and statistics information

• Resource blocking and locking activity

• Data movement service activity

• Errors

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

Question 24: **Correct**
Scenario: Honest Eddie's Car Dealership is an establishment in South Carolina USA, which is dedicated to the purchase and sale of cars and light trucks. Currently the IT team is trying to determine whether they should use a star schema or a snowflake schema.

What is the difference between a star schema and a snowflake schema?

- **A star schema uses surrogate keys while a snowflake schema uses business keys.**

- **All dimensions in a star schema are normalized while all dimensions in a snowflake schema join directly to the fact table (denormalized).**

- **A star schema has one fact table while a snowflake schema has multiple fact tables.**

- **All dimensions in a star schema join directly to the fact table (denormalized) while some dimension tables in a snowflake schema are normalized.**

  (Correct)

Explanation

A star schema is highly denormalized so that the fact table joins directly to dimension; a snowflake schema normalizes some dimensions into multiple tables such as `DimProduct`, `DimProductSubcategory`, and `DimProductCategory`.

Star schema is a mature modelling approach widely adopted by relational data warehouses. It requires modellers to classify their model tables as either dimension or fact.

Dimension tables

Dimension tables describe business entities—the things you model. Entities can include products, people, places, and concepts including time itself. The most consistent table you'll find in a star schema is a date dimension table. A dimension table contains a key column (or columns) that acts as a unique identifier, and descriptive columns.

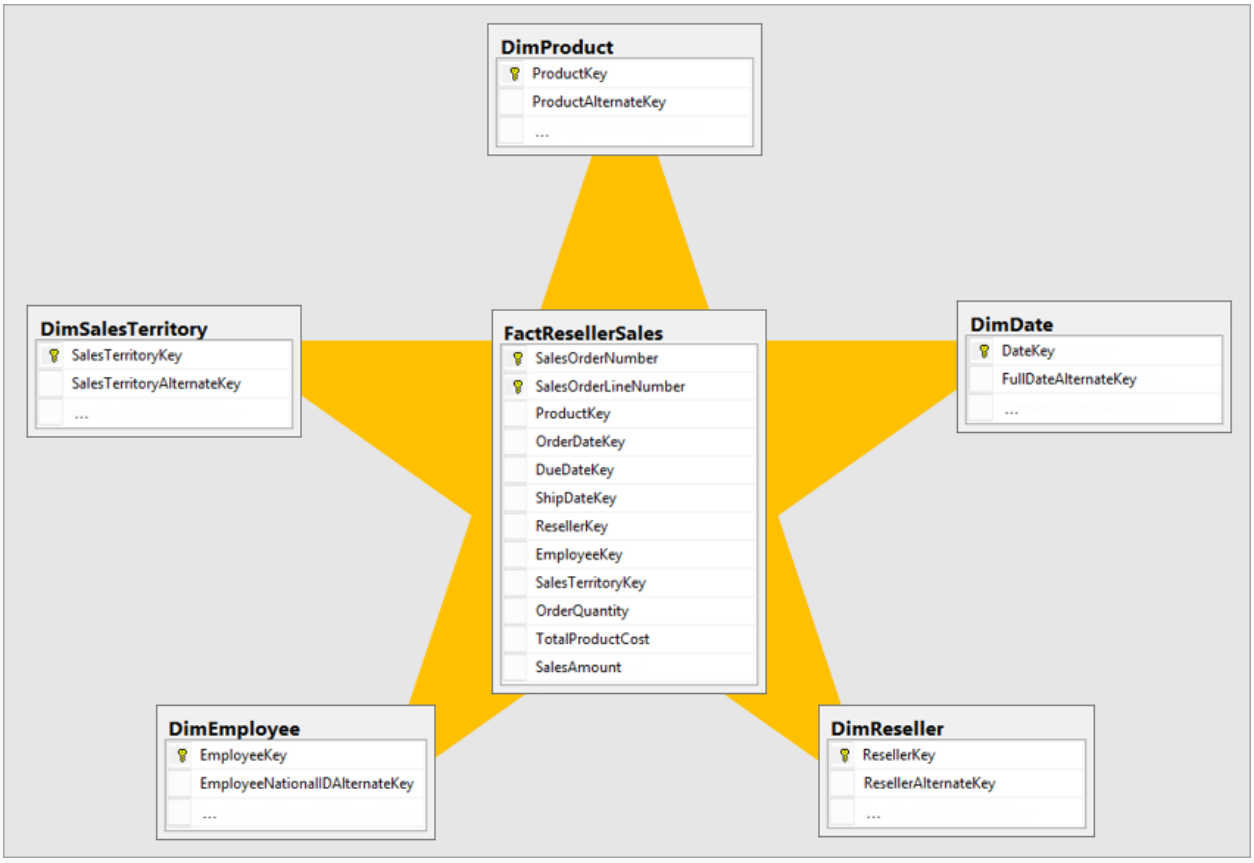
Dimension tables contain attribute data that might change but usually changes infrequently. For example, a customer's name and address are stored in a dimension table and updated only when the customer's profile changes. To minimize the size of a large fact table, the customer's name and address don't need to be in every row of a fact table. Instead, the fact table and the dimension table can share a customer ID. A query can join the two tables to associate a customer's profile and transactions.

Fact tables

Fact tables store observations or events, and can be sales orders, stock balances, exchange rates, temperatures, etc. A fact table contains dimension key columns that relate to dimension tables, and numeric measure columns. The dimension key columns determine the dimensionality of a fact table, while the dimension key values determine the granularity of a fact table. For example, consider a fact table designed to store sale targets that has two dimension key columns Date and ProductKey. It's easy to understand that the table has two dimensions. The granularity, however, can't be determined without considering the dimension key values. In this example, consider that the values stored in the Date column are the first day of each month. In this case, the granularity is at month-product level.

Generally, dimension tables contain a relatively small number of rows. Fact tables, on the other hand, can contain a very large number of rows and continue to grow over time.

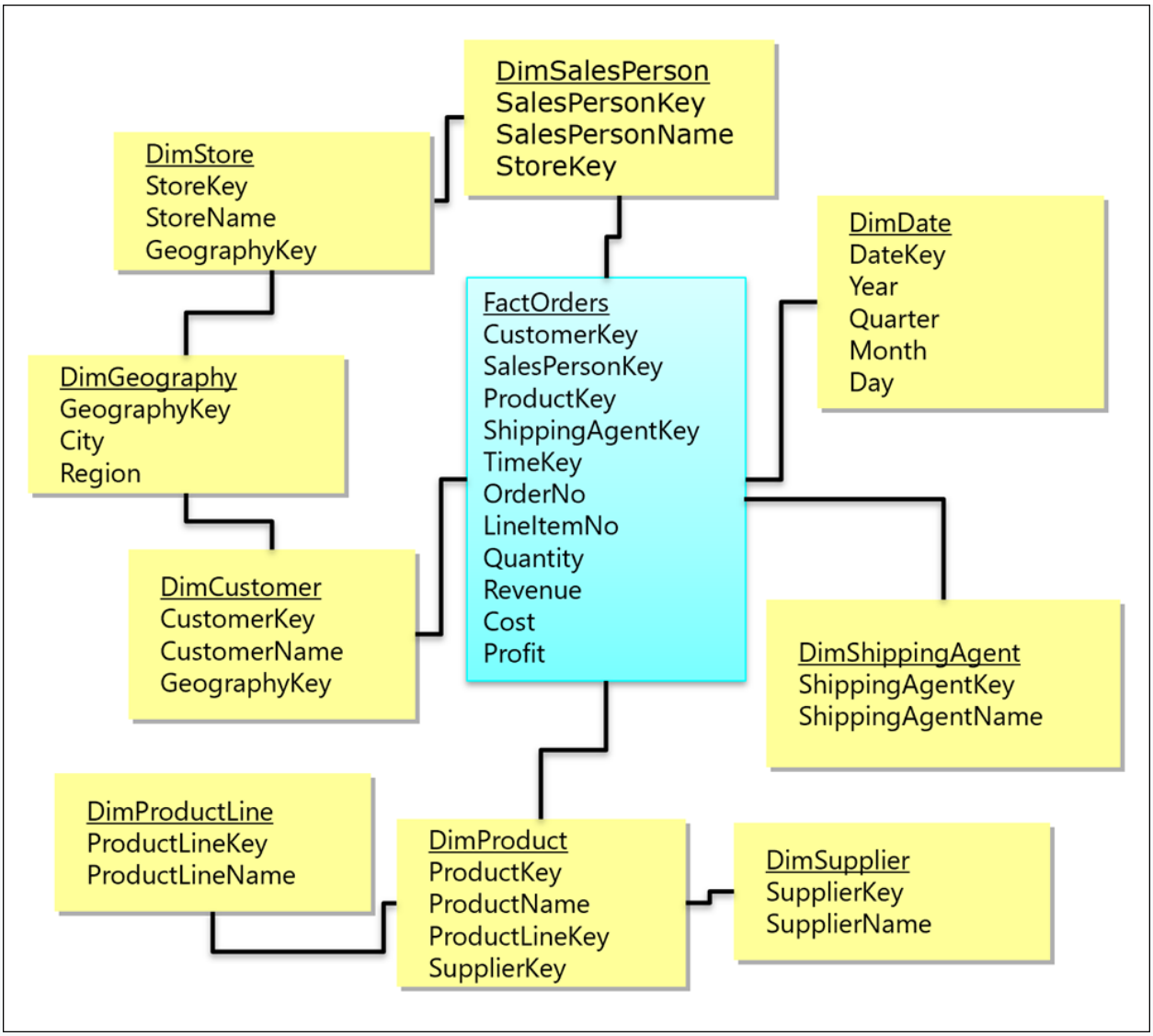
Below is an example star schema, where the fact table is in the middle, surrounded by dimension tables:



https://docs.microsoft.com/en-us/power-bi/guidance/star-schema

A **snowflake schema** is a set of normalized tables for a single business entity. For example, Adventure Works classifies products by category and subcategory. Categories are assigned to subcategories, and products are in turn assigned to subcategories. In the Adventure Works relational data warehouse, the product dimension is normalized and stored in three related tables: `DimProductCategory`, `DimProductSubcategory`, and `DimProduct`.

The snowflake schema is a variation of the star schema. You add normalized dimension tables to a star schema to create a snowflake pattern. In the following diagram, you see the yellow dimension tables surrounding the blue fact table. Notice that many of the dimension tables relate to one another in order to normalize the business entities:



https://www.guru99.com/star-snowflake-data-warehousing.html

Question 25: **Correct**

True or False: In simple terms, you could view DataFrames as you might see in excel, which we could also refer to as a table of data

- ○ **False**
- ◉ **True**

(Correct)

Explanation
What are dataframes?

Basically you could view DataFrames as you might see in excel. It's like a box with squares in it, that organizes data, which we could also refer to as a table of data.

What does a table of data mean?

It is a single set of two-dimensional data that can have multiple rows and columns in the data. Each row, is a sample of data. Each column is a variable or parameter that is able to describe the row that contains the sample of data.

A DataFrame creates a data structure and it's one of the core data structures in Spark. In Spark, it is seen as a distributed collection of data that is organized into columns that have names.

What you see in Data Engineering is that you start with reading or loading data that can be unstructured, semi-structured, or structured, which is stored in a DataFrame and start transforming that data in order to get insights. You can use different functionalities in order to do so, like using Spark SQL, PySpark, and others.

Usually when you see 'df' in some code it refers to a dataframe.

You can either create your own dataframe as this example shows:

```
1. Python
2. new_rows = [('CA',22, 45000),("WA",35,65000) ,("WA",50,85000)]
3. demo_df = spark.createDataFrame(new_rows, ['state', 'age', 'salary'])
4. demo_df.show()
```

Or load a file that contains data into a dataframe like in the below example where the open taxi dataset is used:

```
1. Python
2. from azureml.opendatasets import NycTlcYellow
3.
4. data = NycTlcYellow()
5. data_df = data.to_spark_dataframe()
6. display(data_df.limit(10))
```

Once you're at the stage where you'd like to manipulate the data that is stored in a DataFrame, you can use User-Defined Functions (UDFs) that are column-based and help you transform and manipulate the data stored in a DataFrame.

https://www.tutorialspoint.com/spark_sql/spark_sql_dataframes.htm

Question 26: **Incorrect**

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

Many business application architectures separate transactional and analytical processing into separate systems with data stored and processed on separate infrastructures. [?] systems are optimized for dealing with discrete system or user requests immediately and responding as quickly as possible.

- **OLTP**

  (Correct)

- **ELT**

  (Incorrect)

- **OLAP**

- **ADPS**

- **ETL**

Explanation

Many business application architectures separate transactional and analytical processing into separate systems with data stored and processed on separate infrastructures. These infrastructures are commonly referred to as OLTP (online transaction processing) systems working with operational data, and OLAP (online analytical processing) systems working with historical data, with each system is optimized for their specific task.

OLTP systems are optimized for dealing with discrete system or user requests immediately and responding as quickly as possible.

OLAP systems are optimized for the analytical processing, ingesting, synthesizing, and managing large sets of historical data. The data processed by OLAP systems largely

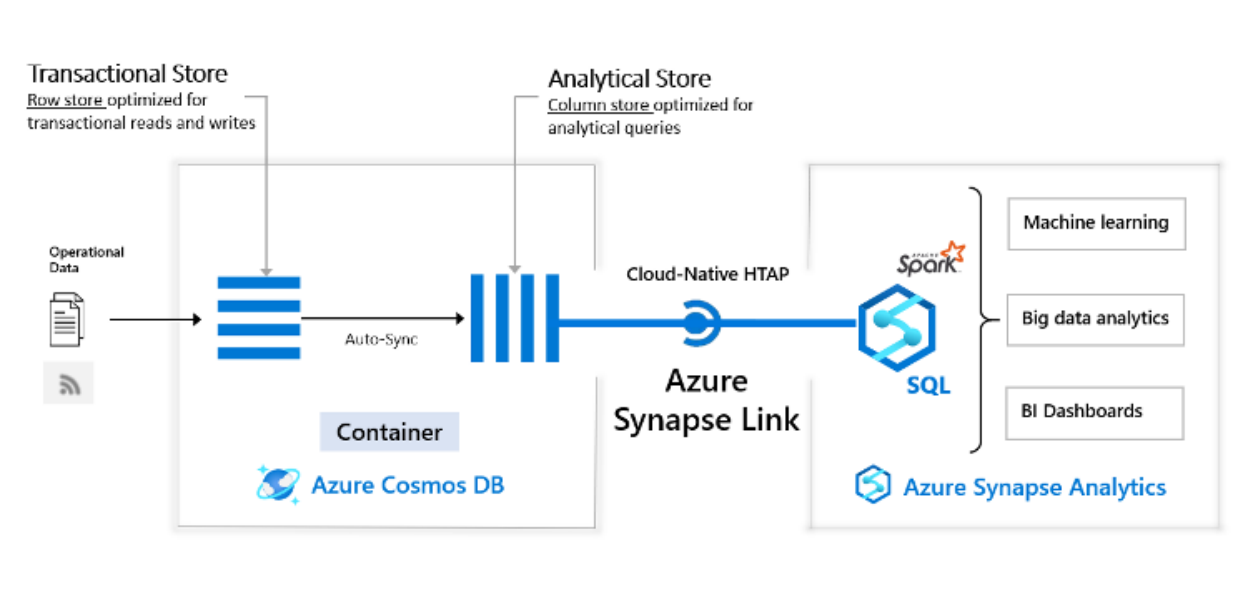
originates from OLTP systems and needs to be loaded into the OLTP systems by means of batch processes commonly referred to as ETL (Extract, Transform, and Load) jobs.

Due to their complexity and the need to physically copy large amounts of data, this creates a delay in data being available to provide insights by way of the OLAP systems.

As more and more businesses move to digital processes, they increasingly recognize the value of being able to respond to opportunities by making faster and well-informed decisions. HTAP (Hybrid Transactional/Analytical processing) enables business to run advanced analytics in near-real-time on data stored and processed by OLTP systems.

Azure Synapse Link for Azure Cosmos DB

Azure Synapse Link for Azure Cosmos DB is a cloud-native HTAP capability that enables you to run near-real-time analytics over operational data stored in Azure Cosmos DB. Azure Synapse Link creates a tight seamless integration between Azure Cosmos DB and Azure Synapse Analytics.



Azure Cosmos DB provides both a transactional store optimized for transactional workloads and an analytical store optimized for analytical workloads and a fully managed autosync process to keep the data within these stores in sync.

Azure Synapse Analytics provides both a SQL Serverless query engine for querying the analytical store using familiar T-SQL and an Apache Spark query engine for leveraging

the analytical store using your choice of Scala, Java, Python or SQL and provides a user-friendly notebook experience.

Together Azure Cosmos DB and Synapse Analytics enable organizations to generate and consume insights from their operational data in near-real time, using the query and analytics tools of their choice. All of this is achieved without the need for complex ETL pipelines and without affecting the performance of their OLTP systems using Azure Cosmos DB.

https://docs.microsoft.com/en-us/azure/cosmos-db/synapse-link

Question 27: **Incorrect**
Azure provides many ways to store your data. A Storage account defines a policy that applies to all the storage services in the account. One of the settings within the Storage account is the Deployment Model which is the system Azure uses to organize the recourses.

Which of the following are valid deployment methods? (Select two)

- ☐

  CloudShell

- ☐

  Resource Manager

  (Correct)

- ☐

  Boards

- ☑

  PowerShell

  (Incorrect)

- ☐

  CLI

- ☐

Classic

(Correct)

Explanation
Azure Storage Deployment Models

A *deployment model* is the system Azure uses to organize your resources. The model defines the API that you use to create, configure, and manage those resources. Azure provides two deployment models:

• **Resource Manager**: the current model that uses the Azure Resource Manager API

• **Classic**: a legacy offering that uses the Azure Service Management API

Most Azure resources only work with Resource Manager, and makes it easy to decide which model to choose. However, storage accounts, virtual machines, and virtual networks support both, so you must choose one or the other when you create your storage account.

The key feature difference between the two models is their support for grouping. The Resource Manager model adds the concept of a *resource group*, which is not available in the classic model. A resource group lets you deploy and manage a collection of resources as a single unit.

Microsoft recommends that you use **Resource Manager** for all new resources.

https://docs.microsoft.com/en-us/azure/azure-resource-manager/management/deployment-models

Question 28: **Correct**
You can integrate your Azure Synapse Analytics workspace with a new Power BI workspace so that you can get you data from within Azure Synapse Analytics visualized in a Power BI report or dashboard.

Which icon should you click on the home page of Azure Synapse Studio to begin the integration?

- **Connect BI**

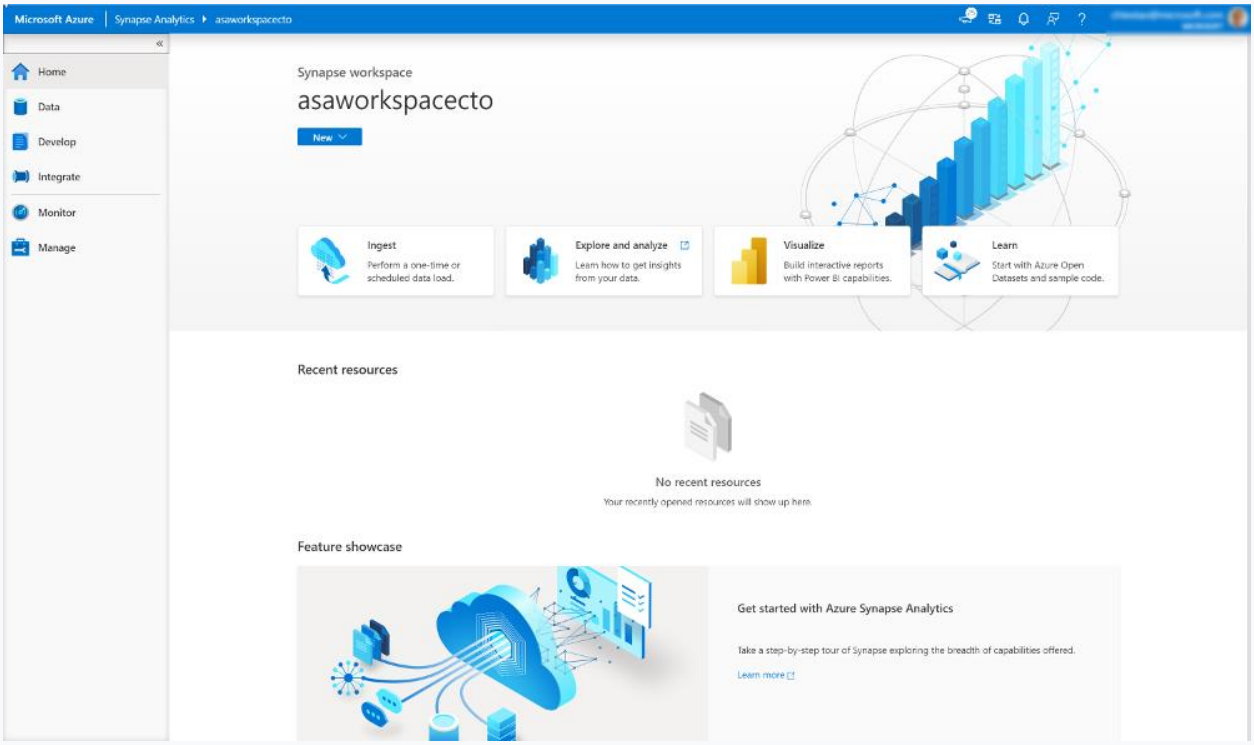
- **Ingest**

- **Explore and analyze**
- **Import**
- **None of the listed options**

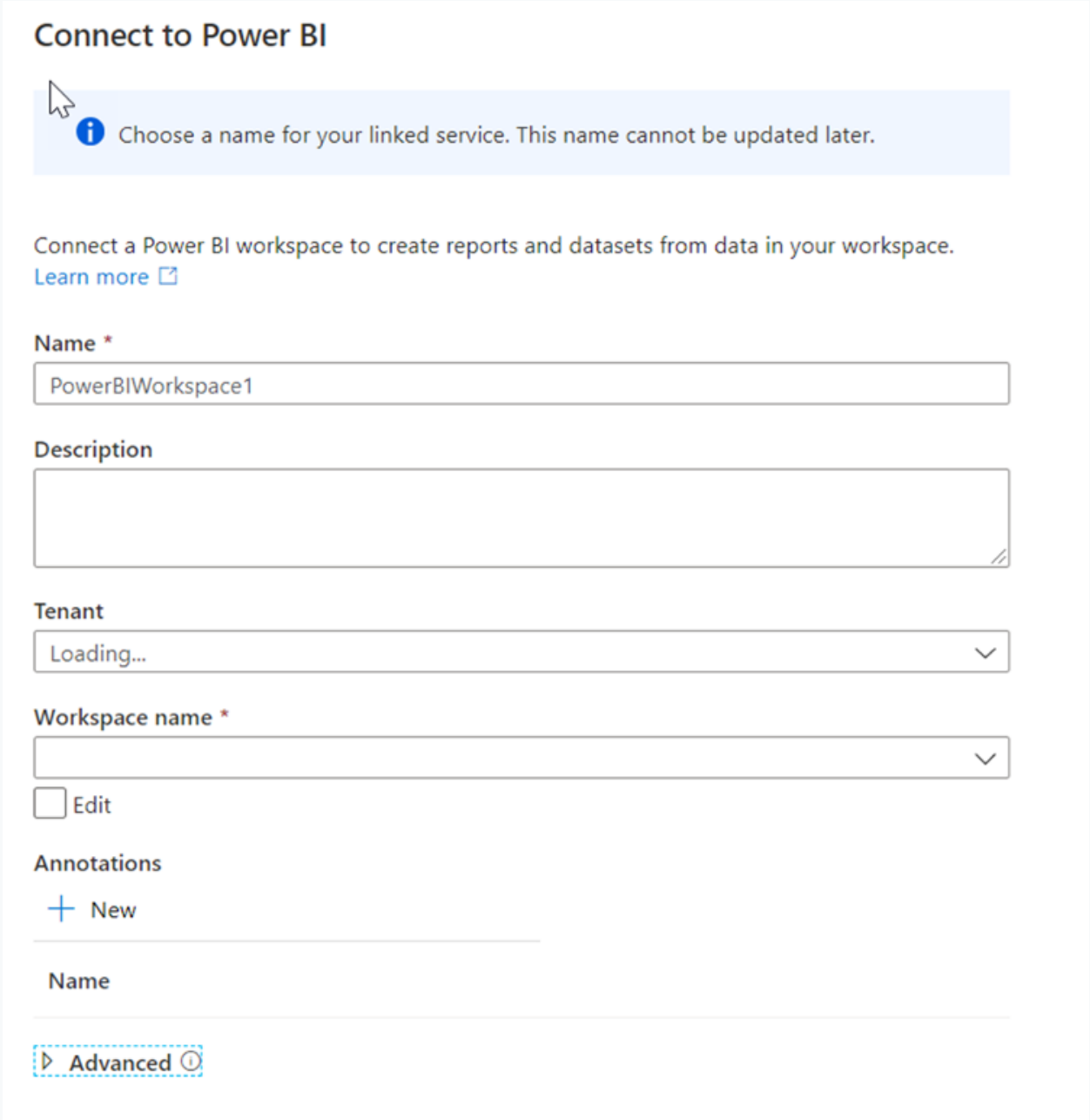
  (Correct)

Explanation
You can integrate your Azure Synapse Analytics workspace with a new Power BI workspace so that you can get you data from within Azure Synapse Analytics visualized in a Power BI report or dashboard.

You can perform this step by clicking on the visualize icon on the home page of Azure Synapse Studio.



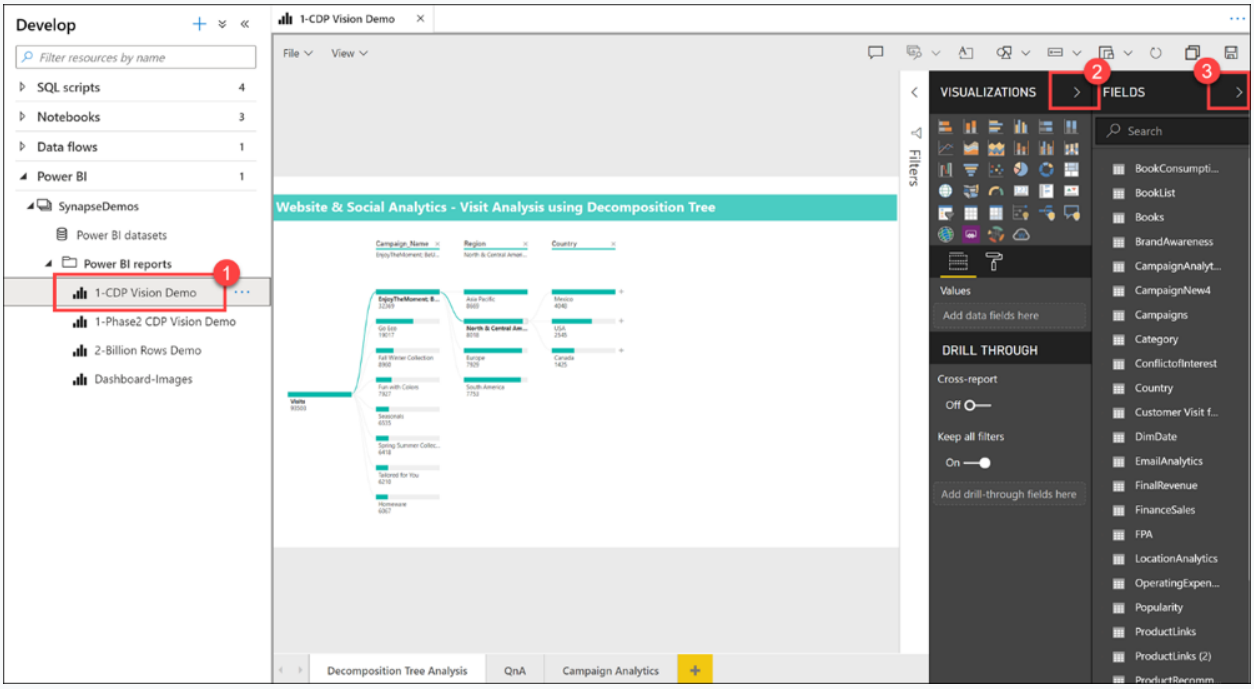
Which will bring up the Connect to Power BI screen.



From here you can define a name and description for the Power BI Workspace. Then you would select the Tenant and Workspace name. Once you have connected to your

workspace, you will be able to access the existing reports in the Power BI workspace in the Develop hub in Azure Synapse Studio.

Expand Power BI, expand SynapseDemos, expand Power BI reports, then select **1-CDP Vision Demo (1)**. Select the arrows to collapse the **Visualizations pane **(2)** and the **Fields** pane **(3)** to increase the report size.



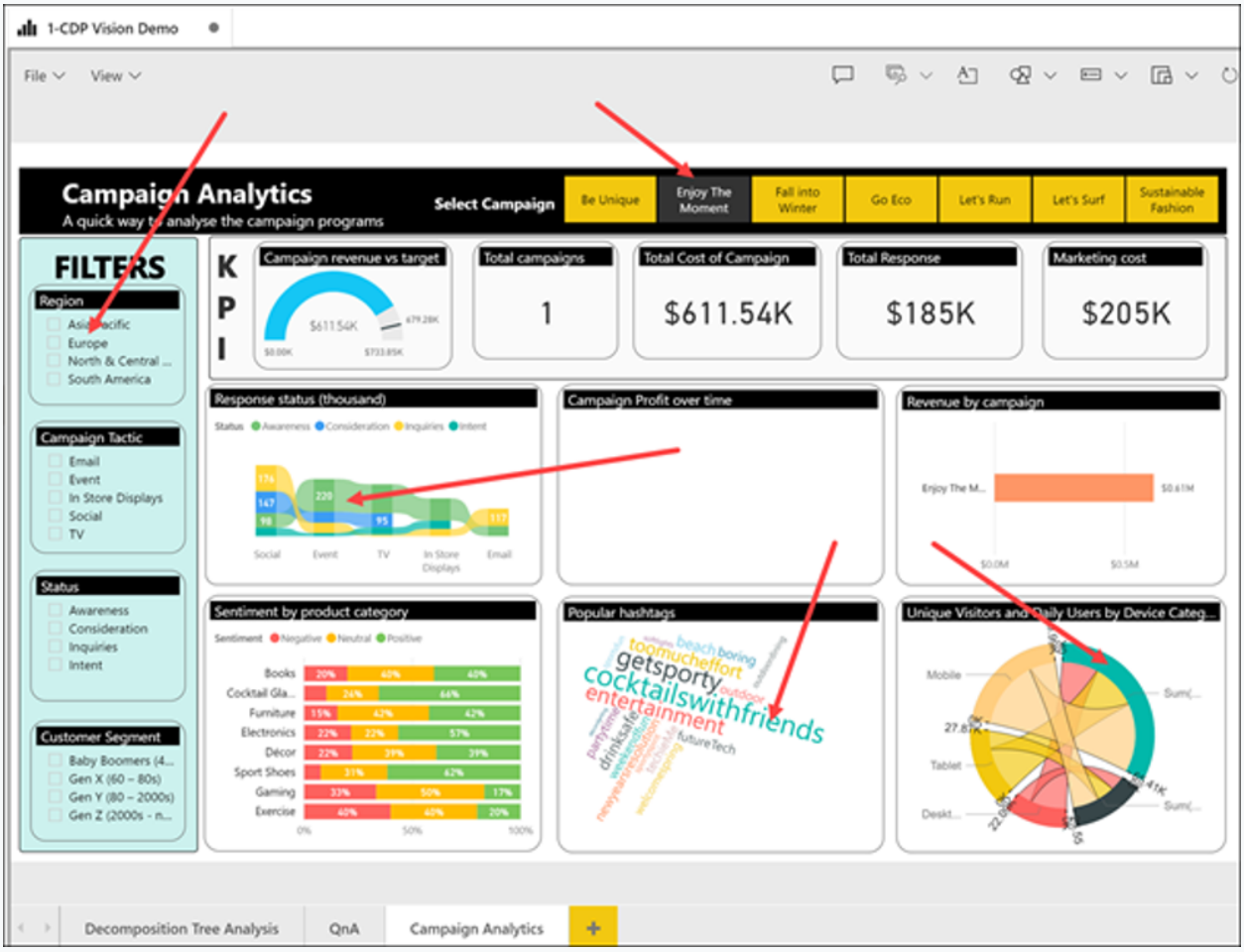
As you can see, we can create, edit, and view Power BI reports from within Synapse Studio! As a business analyst, data engineer, or developer, you no longer need to open another browser window, sign in to Power BI, and toggle back and forth between environments.

Select a **Campaign Name** and **Region** within the **Decomposition Tree Analysis** tab to explore the data. If you hover over an item, you will see a tool tip.

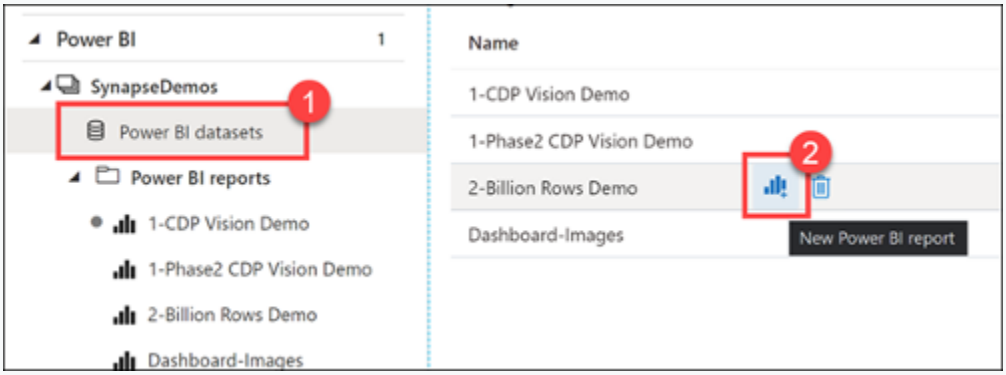
Select the **Campaign Analytics** tab at the bottom of the report.

The Campaign Analytics report combines data from the various data sources to create a compelling visualization of valuable data within an interactive interface.

You can select various filters, campaigns, and chart values to filter the results. Select an item to for the second time to deselect it.

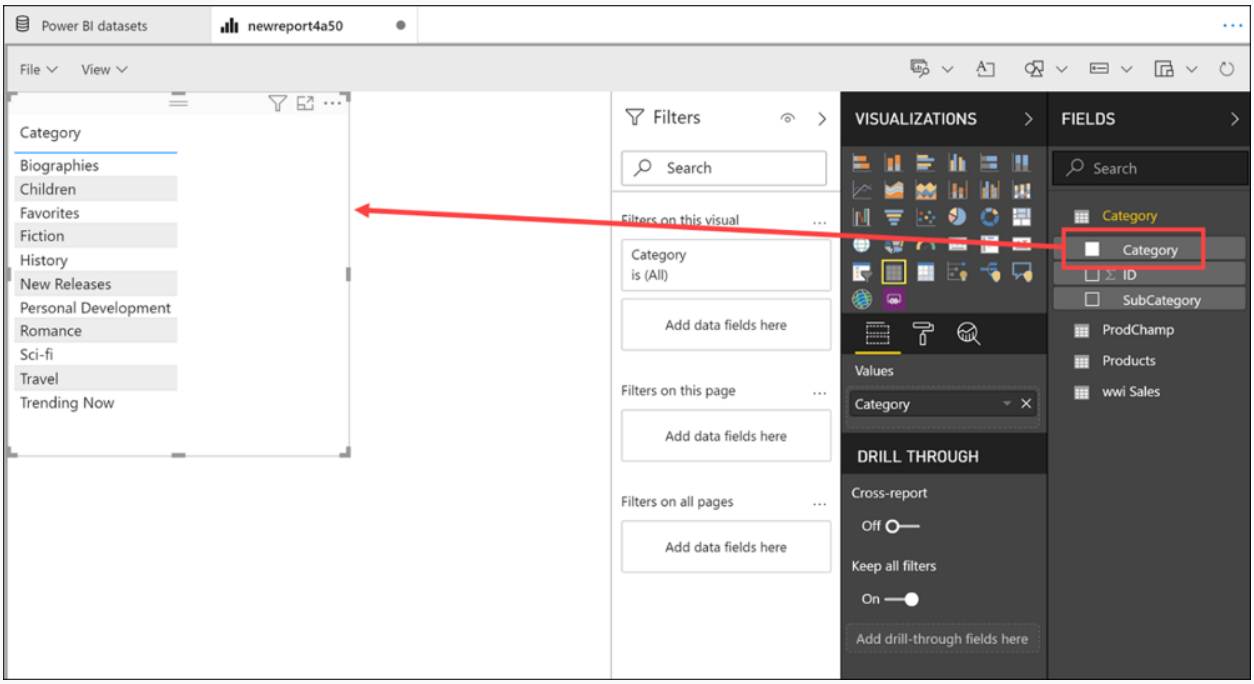


Select **Power BI datasets (1)** in the left-hand menu, hover over the **2-Billion Rows Demo** dataset and select the **New Power BI report** icon **(2)**.

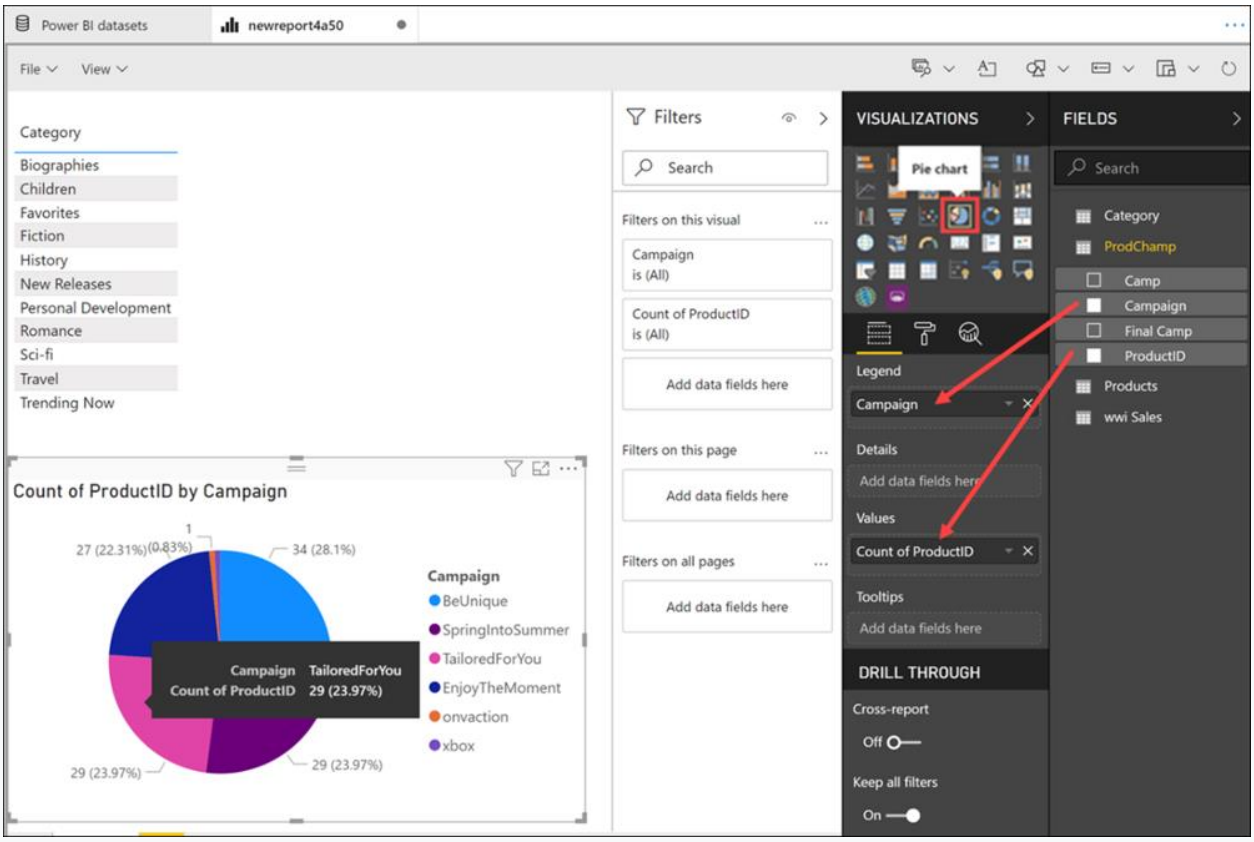


Here is how we can create a brand new Power BI report from a dataset that is part of the linked Power BI workspace, from within Synapse Studio.

Expand the Category table, then **drag-and-drop** the **Category** field on to the report canvas. This creates a new Table visualization that shows the categories.



Select a blank area on the report canvas to deselect the table, then select the **Pie chart** visualization.



Expand the ProdChamp table. Drag **Campaign** onto the **Legend** field, then drag **ProductID** onto the **Values** field. Resize the pie chart and hover over the pie slices to see the tool tips.

We have very quickly created a new Power BI report, using data stored within our Synapse Analytics workspace, without ever leaving the studio.

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

Question 29: **Incorrect**
Which statement about the Azure Databricks Data Plane is true?

- **The Data Plane is where you manage Key Vault itself and it is the interface used to create and delete vaults.**
- **The Data Plane is hosted within a Microsoft-managed subscription.**

- **The Data Plane is hosted within the client subscription and is where all data is processed and stored.**

  (Correct)

- **The Data Plane contains the Cluster Manager and coordinates data processing jobs.**

  (Incorrect)

Explanation
All data is processed by clusters hosted within the client Azure subscription and data is stored within Azure Blob storage and any connected Azure services within this portion of the platform architecture.

https://docs.microsoft.com/en-us/azure/key-vault/general/security-overview

Question 30: **Incorrect**
What size does `OPTIMIZE` compact small files to?

- **Around 1 GB**

  (Correct)

- **Around 100 MB**

- **Around 500 MB**

- **Around 2 GB**

  (Incorrect)

Explanation
The `OPTIMIZE` command compacts small files to around 1GB. The Spark optimization team determined this value to be a good compromise between speed and performance.

https://docs.databricks.com/spark/latest/spark-sql/language-manual/delta-optimize.html

Question 31: **Incorrect**
While Agile, CI/CD, and DevOps are different, they support one another

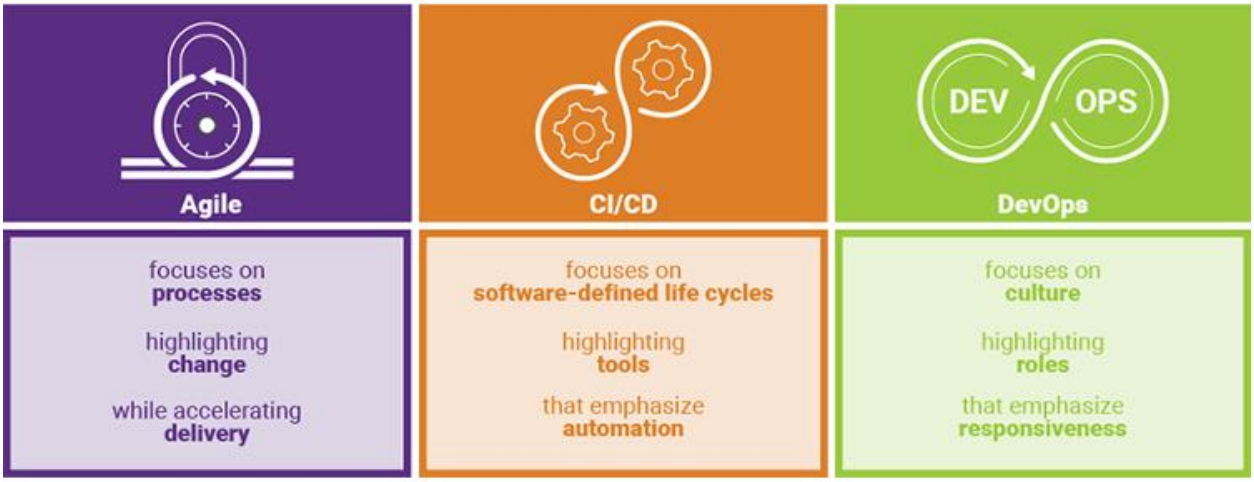
What does CI/CD focus on?

- **Development process**
- **Practices**

  (Correct)
- **Strategy**
- **Culture**

  (Incorrect)

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

Question 32: **Incorrect**
Which Azure Synapse Analytics component enables you to perform Hybrid Transactional and Analytical Processing?

- **Azure Synapse Spark pools**
- **Azure Synapse Studio**
- **Azure Synapse Link**

  (Correct)
- **Azure Synapse Pipeline**
- **Azure Stream Analytics**
- **None of the listed options**

  (Incorrect)
- **Azure Data Explorer**

-

Azure Data Warehouse

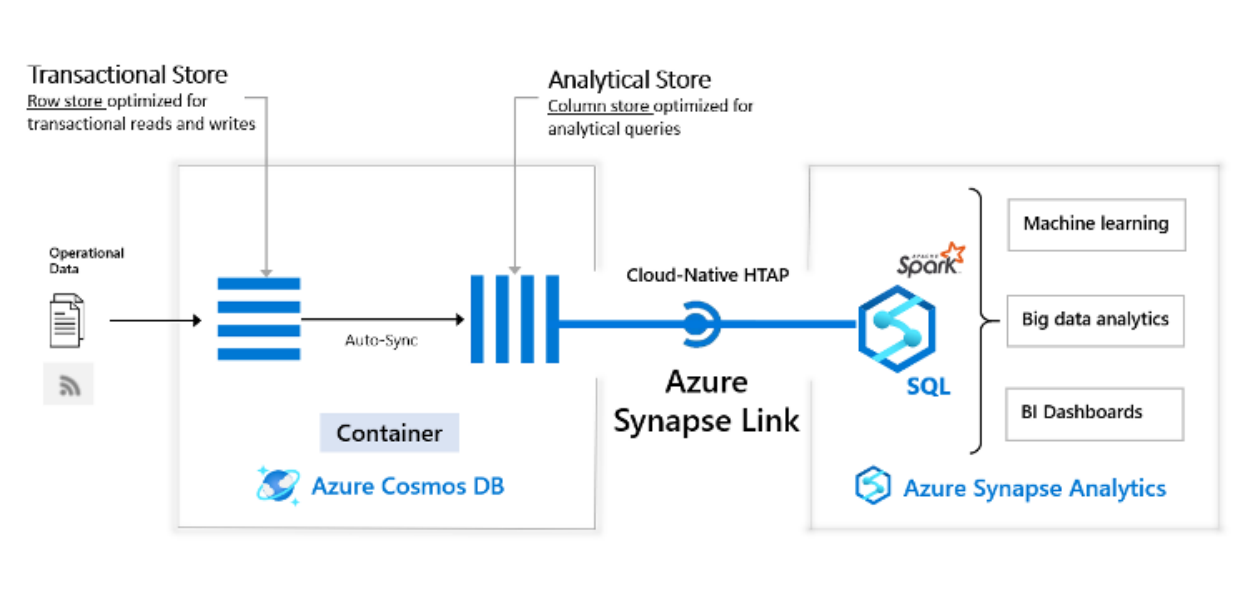
Explanation

Azure Synapse Link is the component that enables Hybrid Transactional and Analytical Processing.

Azure Synapse Link for Azure Cosmos DB is a cloud-native hybrid transactional and analytical processing (HTAP) capability that enables you to run near real-time analytics over operational data in Azure Cosmos DB. Azure Synapse Link creates a tight seamless integration between Azure Cosmos DB and Azure Synapse Analytics.

Using Azure Cosmos DB analytical store, a fully isolated column store, Azure Synapse Link enables no Extract-Transform-Load (ETL) analytics in Azure Synapse Analytics against your operational data at scale. Business analysts, data engineers and data scientists can now use Synapse Spark or Synapse SQL interchangeably to run near real-time business intelligence, analytics, and machine learning pipelines. You can achieve this without impacting the performance of your transactional workloads on Azure Cosmos DB.

The following image shows the Azure Synapse Link integration with Azure Cosmos DB and Azure Synapse Analytics:



https://docs.microsoft.com/en-us/azure/cosmos-db/synapse-link

Question 33: **Incorrect**

Which of the following are supported connectors for built-in parameterization? (Select all that apply)

- ☐

  Azure Data Lake Storage Gen2

- ☐

  Amazon S3

  (Correct)

- ☐

  Azure Key Vault

  (Correct)

- ☐

  Azure Data Lake Storage Gen1

- ☑

  Azure Synapse Analytics

  (Correct)

Explanation
Azure Synapse Analytics is a supported connector for built-in parameterization for Linked Services in Azure Data Factory.

Supported linked service types

You can parameterize any type of linked service. When authoring linked service on UI, Data Factory provides built-in parameterization experience for the following types of linked services. In linked service creation/edit blade, you can find options to new parameters and add dynamic content.

• Amazon Redshift

• Amazon S3

• Azure Cosmos DB (SQL API)

• Azure Database for MySQL

• Azure Databricks

• Azure Key Vault

• Azure SQL Database

• Azure SQL Managed Instance

• Azure Synapse Analytics

• MySQL

• Oracle

• SQL Server

• Generic HTTP

• Generic REST

For other linked service types that are not in above list, you can parameterize the linked service by editing the JSON on 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.

Refer to the JSON sample to add `parameters` section to define parameters and reference the parameter using `@{linkedService().paraName}`.

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

Question 34: **Incorrect**
Which is an element of a Spark Pool in Azure Synapse Analytics?

•

- **Databricks**

- **Spark Instance**

  (Correct)

- **HDI**

- **Spark Console**

  (Incorrect)

Explanation
The definition of a Spark pool is that, when instantiated, it is used to create a Spark instance that processes data.

Apache Spark is a parallel processing framework that supports in-memory processing to boost the performance of big-data analytic applications. Apache Spark in Azure Synapse Analytics is one of Microsoft's implementations of Apache Spark in the cloud.

Azure Synapse makes it easy to create and configure Spark capabilities in Azure. Azure Synapse provides a different implementation of these Spark capabilities that are documented here.

Spark pools

A serverless Apache Spark pool is created in the Azure portal. It's the definition of a Spark pool that, when instantiated, is used to create a Spark instance that processes data. When a Spark pool is created, it exists only as metadata, and no resources are consumed, running, or charged for. A Spark pool has a series of properties that control the characteristics of a Spark instance. These characteristics include but aren't limited to name, size, scaling behaviour, time to live.

As there's no dollar or resource cost associated with creating Spark pools, any number can be created with any number of different configurations. Permissions can also be applied to Spark pools allowing users only to have access to some and not others.

A best practice is to create smaller Spark pools that may be used for development and debugging and then larger ones for running production workloads.

Spark instances

Spark instances are created when you connect to a Spark pool, create a session, and run a job. As multiple users may have access to a single Spark pool, a new Spark instance is created for each user that connects.

When you submit a second job, if there is capacity in the pool, the existing Spark instance also has capacity. Then, the existing instance will process the job. Otherwise, if capacity is available at the pool level, then a new Spark instance will be created.

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

Question 35: **Incorrect**
Identify the missing word(s) in the following sentence within the context of Microsoft Azure.

Within Azure Synapse SQL, [?] stores a copy of the result set on the control node so that queries do not need to pull data from the storage subsystem or compute nodes.

- **Result-set caching**

  (Correct)

- **Site caching**

- **VM caching**

- **Browser caching**

- **Server caching**

(Incorrect)

Explanation
Enable result-set caching when you expect results from queries to return the same values.

This option stores a copy of the result set on the control node so that queries do not need to pull data from the storage subsystem or compute nodes. The capacity for the result set cache is 1 TB and the data within the result-set cache is expired and purged after 48 hours of not being accessed.

Azure Synapse SQL automatically caches query results in the user database for repetitive use. Result-set caching allows subsequent query executions to get results directly from the persisted cache so recomputation is not needed. Result set caching improves query performance and reduces compute resource usage.

To enable result set caching, run this command when connecting to the MASTER database.

```
1. SQL
2. ALTER DATABASE [database_name]
3. SET RESULT_SET_CACHING ON;
```

https://docs.microsoft.com/en-us/azure/synapse-analytics/sql-data-warehouse/performance-tuning-result-set-caching

Question 36: **Incorrect**
What should be done when a connector in data factory is not supported in mapping data flow in order to transform data from one of these sources? (Select all that apply)

- ☐

  Use a group by activity in Dataflow.

- ☐

  Use a generic ODBC connector.

  (Correct)

- ☐

  Ingest the data into a supported source using the copy activity.

  (Correct)

- ☐

Use a generic REST connector.

(Correct)

- ☑

Use an aggregate transformation in Dataflow.

(Incorrect)

Explanation
If a connector in Data factory is not supported, create a copy activity of the source data into a supported data source in mapping dataflow and continue the transformations from there.

Integrate with more data stores

Azure Data Factory can reach a very broad set of data stores. If you need to move data to/from a data store that is not in the Azure Data Factory built-in connector list, here are some extensible options:

• For database and data warehouse, usually you can find a corresponding ODBC driver, with which you can use generic ODBC connector.

• For SaaS applications:

  • If it provides RESTful APIs, you can use generic REST connector.

  • If it has OData feed, you can use generic OData connector.

  • If it provides SOAP APIs, you can use generic HTTP connector.

  • If it has ODBC driver, you can use generic ODBC connector.

• For others, check if you can load data to or expose data as any ADF supported data stores, e.g. Azure Blob/File/FTP/SFTP/etc, then let ADF pick up from there. You can invoke custom data loading mechanism via Azure Function, Custom activity, Databricks/HDInsight, Web activity, etc.

Question 37: **Incorrect**
Scenario: You are working as a consultant at Avengers Security and the IT team has developed a data ingestion process to import data to a Microsoft Azure SQL Data Warehouse. They are using an Azure Data Lake Gen 2 storage account to store the data to be ingested. The data to be ingested resides in parquet files.

Required: Load the data from the Azure Data Lake Gen 2 storage account into the Azure SQL Data Warehouse.

The Avengers IT team has proposed the following solution:

1. Create an external data source pointing to the Azure storage account

2. Create an external file format and external table using the external data source

3. Load the data using the `INSERT ... SELECT` statement

Will the solution proposed by the Avengers IT team meet the requirement?

- **No**

  (Correct)

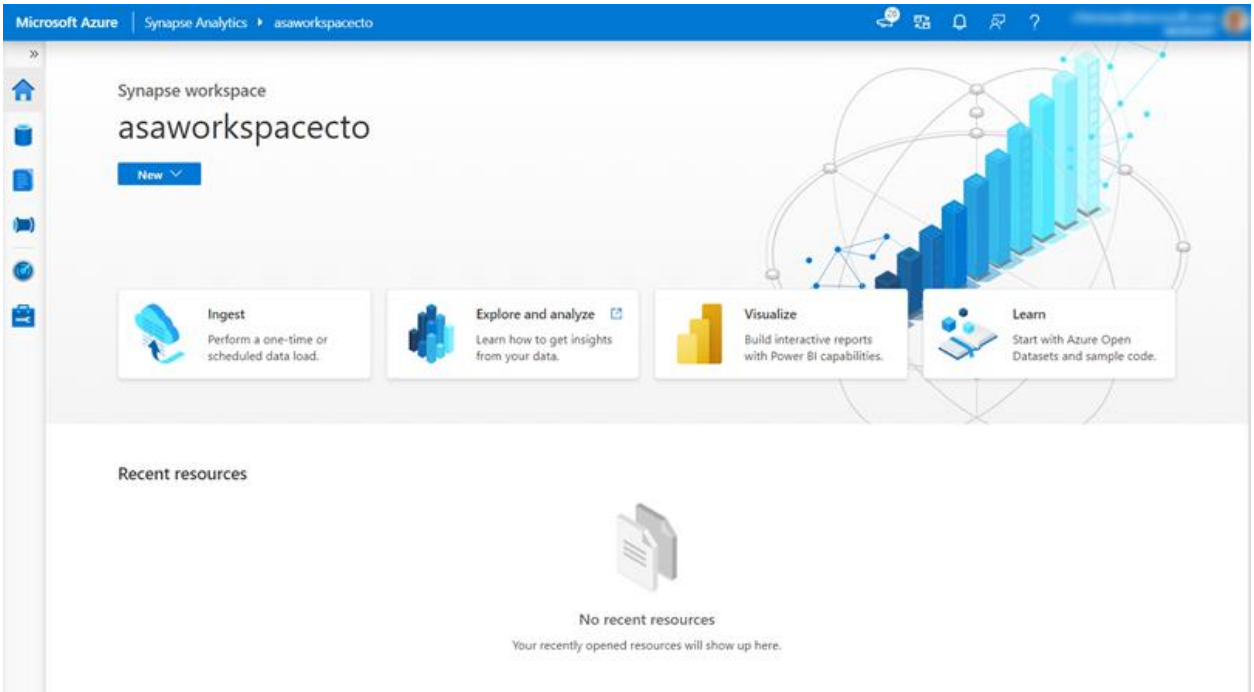
- **Yes**

  (Incorrect)

Explanation
The proposed solution will not meet the requirement. They need to create an external file format and external table using the external data source. To load the data, use the `CREATE TABLE ... AS SELECT` statement.

Use polybase by defining external tables

Using Transact-SQL, you can use PolyBase to access files that are located directly on Azure Storage as if they were structured tables within your SQL Pool. You define an **external data source** pointing to the location of the file or the folder the files reside in, the external file format, which can be GZip compressed delimited text, ORC, Parquet or JSON, and then the external table with the column attributes that map to the structure from the external files.



Create an import database

The first step in using PolyBase is to create a database-scoped credential that secures the credentials to the blob storage. Create a master key first, and then use this key to encrypt the database-scoped credential named **AzureStorageCredential**.

1. Paste the following code into the query window. Replace the `SECRET` value with the access key you retrieved in the previous exercise.

```
SQL
CREATE MASTER KEY;

CREATE DATABASE SCOPED CREDENTIAL AzureStorageCredential
WITH
    IDENTITY = 'demodwStorage',
    SECRET = 'THE-VALUE-OF-THE-ACCESS-KEY' -- put key1's value here
;
```

2. Select **Run** to run the query. It should report `Query succeeded: Affected rows: 0.`

Create an external data source connection

Use the database-scoped credential to create an external data source named **AzureStorage**. Note the location URL point to the container named **data-files** that you created in the blob storage. The type **Hadoop** is used for both Hadoop-based and Azure Blob storage-based access.

1. Paste the following code into the query window. Replace the `LOCATION` value with your correct value from the previous exercise.

```
SQL
CREATE EXTERNAL DATA SOURCE AzureStorage
WITH (
    TYPE = HADOOP,
    LOCATION = 'wasbs://data-files@demodwstorage.blob.core.windows.net',
    CREDENTIAL = AzureStorageCredential
);
```

2. Select **Run** to run the query. It reports `Query succeeded: Affected rows: 0.`.

Define the import file format

Define the external file format named **TextFile**. This name indicates to PolyBase that the format of the text file is **DelimitedText** and the field terminator is a comma.

1. Paste the following code into the query window.

```
SQL
CREATE EXTERNAL FILE FORMAT TextFile
WITH (
    FORMAT_TYPE = DelimitedText,
    FORMAT_OPTIONS (FIELD_TERMINATOR = ',')
);
```

2. Select **Run** to run the query. It reports `Query succeeded: Affected rows: 0.`.

Create a temporary table

Create an external table named `dbo.temp` with the column definition for your table. At the bottom of the query, use a `WITH` clause to call the data source definition named **AzureStorage**, as previously defined, and the file format named **TextFile**, as previously defined. The location denotes that the files for the load are in the root folder of the data source.

Note: External tables are in-memory tables that don't persist onto the physical disk. External tables can be queried like any other table.

The table definition must match the fields defined in the input file. There are 12 defined columns, with data types that match the input file data.

1. Add the following code into the Visual Studio window underneath the previous code.

```
SQL
-- Create a temp table to hold the imported data
CREATE EXTERNAL TABLE dbo.Temp (
    [Date] datetime2(3) NULL,
    [DateKey] decimal(38, 0) NULL,
```

```
    [MonthKey] decimal(38, 0) NULL,
    [Month] nvarchar(100) NULL,
    [Quarter] nvarchar(100) NULL,
    [Year] decimal(38, 0) NULL,
    [Year-Quarter] nvarchar(100) NULL,
    [Year-Month] nvarchar(100) NULL,
    [Year-MonthKey] nvarchar(100) NULL,
    [WeekDayKey] decimal(38, 0) NULL,
    [WeekDay] nvarchar(100) NULL,
    [Day Of Month] decimal(38, 0) NULL
)
WITH (
    LOCATION='../',
    DATA_SOURCE=AzureStorage,
    FILE_FORMAT=TextFile
);
```

2. Select **Run** to run the query. It takes a few seconds to complete and reports `Query succeeded: Affected rows: 0.`.

Create a destination table

Create a physical table in the Azure Synapse Analytics database. In the following example, you create a table named `dbo.StageDate`. The table has a clustered column store index defined on all the columns. It uses a table geometry of `round_robin` by design because `round_robin` is the best table geometry to use for loading data.

1. Paste the following code into the query window.

```
SQL
-- Load the data from Azure Blob storage to Azure Synapse Analytics
CREATE TABLE [dbo].[StageDate]
WITH (
    CLUSTERED COLUMNSTORE INDEX,
    DISTRIBUTION = ROUND_ROBIN
)
AS
SELECT * FROM [dbo].[Temp];
```

2. Select **Run** to run the query. It takes a few seconds to complete and reports `Query succeeded: Affected rows: 0.`.

Add statistics onto columns to improve query performance

As an optional step, create statistics on columns that feature in queries to improve the query performance against the table.

1. Paste the following code into the query window.

```
SQL
-- Create statistics on the new data
CREATE STATISTICS [DateKey] on [StageDate] ([DateKey]);
```

```
4. CREATE STATISTICS [Quarter] on [StageDate] ([Quarter]);
5. CREATE STATISTICS [Month] on [StageDate] ([Month]);
```

2. Select **Run** to run the query. It reports `Query succeeded: Affected rows: 0.`.

You've loaded your first staging table in Azure Synapse Analytics. From here, you can write further Transact-SQL queries to perform transformations into dimension and fact tables. Try it out by querying the `StageDate` table in the query explorer or in another query tool. Refresh the view on the left to see the new table or tables that you created. Reuse the previous steps in a persistent SQL script to load additional data, as necessary.

https://docs.microsoft.com/en-us/azure/sql-data-warehouse/sql-data-warehouse-load-from-azure-data-lake-store

Question 38: **Incorrect**
Scenario: You are working on a project with a 3rd party vendor to build a website for a customer. The image assets that will be used on the website are stored in an Azure Storage account that is held in your subscription. You want to give read access to this data for a limited period of time.

What security option would be the best option to use?

- **Storage Account**
- **Shared Access Signatures**

  (Correct)
- **CORS Support**
- **Private Link**

  (Incorrect)

Explanation

A shared access signature is a string that contains a security token that can be attached to a URI. Use a shared access signature to delegate access to storage objects and specify constraints, such as the permissions and the time range of access.

Shared Access Signatures (SAS)

Access keys are the easiest approach to authenticating access to a storage account. However they provide full access to anything in the storage account, similar to a root password on a computer.

Storage accounts offer a separate authentication mechanism called *shared access signatures* that support expiration and limited permissions for scenarios where you need to grant limited access. You should use this approach when you are allowing other users to read and write data to your storage account. There are links to our documentation on this advanced topic at the end of the module.

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

Question 39: **Incorrect**
By default, the Azure Data Factory user interface experience (UX) authors directly against the data factory service.

To provide a better authoring experience, Azure Data Factory allows you to configure a Git repository with either Azure Repos or GitHub. Git is a version control system that allows for easier change tracking and collaboration.

True or False: Configuring a git repository allows you to save changes, letting you only publish when you have tested your changes to your satisfaction.

- **True**

  (Correct)

- **False**

  (Incorrect)

Explanation
By default, the Azure Data Factory user interface experience (UX) authors directly against the data factory service. This experience has the following limitations:

• The Data Factory service doesn't include a repository for storing the JSON entities for your changes. The only way to save changes is via the **Publish All** button and all changes are published directly to the data factory service.

• The Data Factory service isn't optimized for collaboration and version control.

• The Azure Resource Manager template required to deploy Data Factory itself is not included.

To provide a better authoring experience, Azure Data Factory allows you to configure a Git repository with either Azure Repos or GitHub. Git is a version control system that allows for easier change tracking and collaboration.

Advantages of Git integration

Below is a list of some of the advantages git integration provides to the authoring experience:

• **Source control:** As your data factory workloads become crucial, you would want to integrate your factory with Git to leverage several source control benefits like the following:

- Ability to track/audit changes.
- Ability to revert changes that introduced bugs.

• **Partial saves:** When authoring against the data factory service, you can't save changes as a draft and all publishes must pass data factory validation. Whether your pipelines are not finished or you simply don't want to lose changes if your computer crashes, git integration allows for incremental changes of data factory resources regardless of what state they are in. **Configuring a git repository allows you to save changes, letting you only publish when you have tested your changes to your satisfaction.**

• **Collaboration and control:** If you have multiple team members contributing to the same factory, you may want to let your teammates collaborate with each other via a code review process. You can also set up your factory such that not every contributor has equal permissions. Some team members may only be allowed to make changes via Git and only certain people in the team are allowed to publish the changes to the factory.

• **Better CI/CD:** If you are deploying to multiple environments with a continuous delivery process, git integration makes certain actions easier. Some of these actions include:

- Configure your release pipeline to trigger automatically as soon as there are any changes made to your 'dev' factory.

• Customize the properties in your factory that are available as parameters in the Resource Manager template. It can be useful to keep only the required set of properties as parameters, and have everything else hard coded.

• **Better Performance:** An average factory with git integration loads 10 times faster than one authoring against the data factory service. This performance improvement is because resources are downloaded via Git.

Connect to a Git repository

There are different ways to connect a Git repository to your data factory for both Azure Repos and GitHub. After you connect to a Git repository, you can view and manage your configuration in the management hub under **Git configuration** in the **Source control** section.

Configuration method 1: Home page

In the Azure Data Factory home page, select **Set up Code Repository**.

Configuration method 2: Authoring canvas

In the Azure Data Factory UX authoring canvas, select the **Data Factory** drop-down menu, and then select **Set up Code Repository**.

Configuration method 3: Management hub

Go to the management hub in the Azure Data Factory UX. Select **Git configuration** in the **Source control** section. If you have no repository connected, click **Set up code repository**.

https://docs.microsoft.com/en-us/azure/data-factory/source-control

Question 40: **Incorrect**
Azure Data Lake Storage combines a file system with a storage platform to help you quickly identify insights into your data. Data Lake Storage Gen2 builds on Azure Blob storage capabilities to optimize it specifically for analytics workloads.

You can set permissions at a directory level or file level for the data stored within the data lake. This security is configurable through technologies such as Hive and Spark, or utilities such as Azure Storage Explorer. All data that is stored is encrypted at rest by using either Microsoft or customer-managed keys.

Data Lake Storage Gen2 supports which of the following to enhance security?

- ☐

- ACLs

  (Correct)

- LRS

- HDFS

- POSIX

  (Correct)

- AWS

- GRS

Explanation

A data lake is a repository of data that is stored in its natural format, usually as blobs or files. Azure Data Lake Storage is a comprehensive, scalable, and cost-effective data lake solution for big data analytics built into Azure.

Azure Data Lake Storage combines a file system with a storage platform to help you quickly identify insights into your data. Data Lake Storage Gen2 builds on Azure Blob storage capabilities to optimize it specifically for analytics workloads. This integration enables analytics performance, the tiering and data lifecycle management capabilities of Blob storage, and the high-availability, security, and durability capabilities of Azure Storage.

The variety and volume of data that is generated and analyzed today is increasing. Companies have multiple sources of data, from websites to Point of Sale (POS) systems, and more recently from social media sites to Internet of Things (IoT) devices. Each source provides an essential aspect of data that needs to be collected, analyzed, and potentially acted upon.

Benefits

Data Lake Storage Gen2 is designed to deal with this variety and volume of data at exabyte scale while securely handling hundreds of gigabytes of throughput. With this, you can use Data Lake Storage Gen2 as the basis for both real-time and batch solutions. Here is a list of additional benefits that Data Lake Storage Gen2 brings:

Hadoop compatible access

A benefit of Data Lake Storage Gen2 is that you can treat the data as if it's stored in a Hadoop Distributed File System. With this feature, you can store the data in one place and access it through compute technologies including Azure Databricks, Azure HDInsight, and Azure Synapse Analytics without moving the data between environments.

Security

Data Lake Storage Gen2 supports access control lists (ACLs) and Portable Operating System Interface (POSIX) permissions. You can set permissions at a directory level or file level for the data stored within the data lake. This security is configurable through technologies such as Hive and Spark, or utilities such as Azure Storage Explorer. All data that is stored is encrypted at rest by using either Microsoft or customer-managed keys.

Performance

Azure Data Lake Storage organizes the stored data into a hierarchy of directories and subdirectories, much like a file system, for easier navigation. As a result, data processing requires less computational resources, reducing both the time and cost.

Data redundancy

Data Lake Storage Gen2 takes advantage of the Azure Blob replication models that provide data redundancy in a single data centre with locally redundant storage (LRS), or to a secondary region by using the Geo-redundant storage (GRS) option. This feature ensures that your data is always available and protected if catastrophe strikes.

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

Question 41: **Incorrect**
What does the `APPROX_COUNT_DISTINCT` Transact-SQL function do?

- **None of the listed options.**

- **Approximate execution using Hyperlog accuracy**

- **Calculates the approximate number of distinct records in a relational database.**

  (Correct)

- **Approximate count on distinct executions within a specified time period on a specific endpoint.**

- **Calculates the approximate number of distinct records in a non-relational database.**

  (Incorrect)

Explanation
APPROX_COUNT_DISTINCT (Transact-SQL)

This function returns the approximate number of unique non-null values in a group.

Syntax

```
1. syntaxsql
2. APPROX_COUNT_DISTINCT ( expression )
```

Arguments

expression
An expression of any type, except **image**, **sql_variant**, **ntext**, or **text**.

Return types

bigint

Remarks

`APPROX_COUNT_DISTINCT( expression )` evaluates an expression for each row in a group, and returns the approximate number of unique non-null values in a group. This

function is designed to provide aggregations across large data sets where responsiveness is more critical than absolute precision.

`APPROX_COUNT_DISTINCT` is designed for use in big data scenarios and is optimized for the following conditions:

Access of data sets that are millions of rows or higher *and*

Aggregation of a column or columns that have many distinct values

The function implementation guarantees up to a 2% error rate within a 97% probability.

`APPROX_COUNT_DISTINCT` requires less memory than an exhaustive COUNT DISTINCT operation. Given the smaller memory footprint, `APPROX_COUNT_DISTINCT` is less likely to spill memory to disk compared to a precise COUNT DISTINCT operation. To learn more about the algorithm used to achieve this, see HyperLogLog.

https://docs.microsoft.com/en-us/sql/t-sql/functions/approx-count-distinct-transact-sql?view=sql-server-ver15

Question 42: **Incorrect**
Which hub is where you can grant access to Synapse workspace and resources?

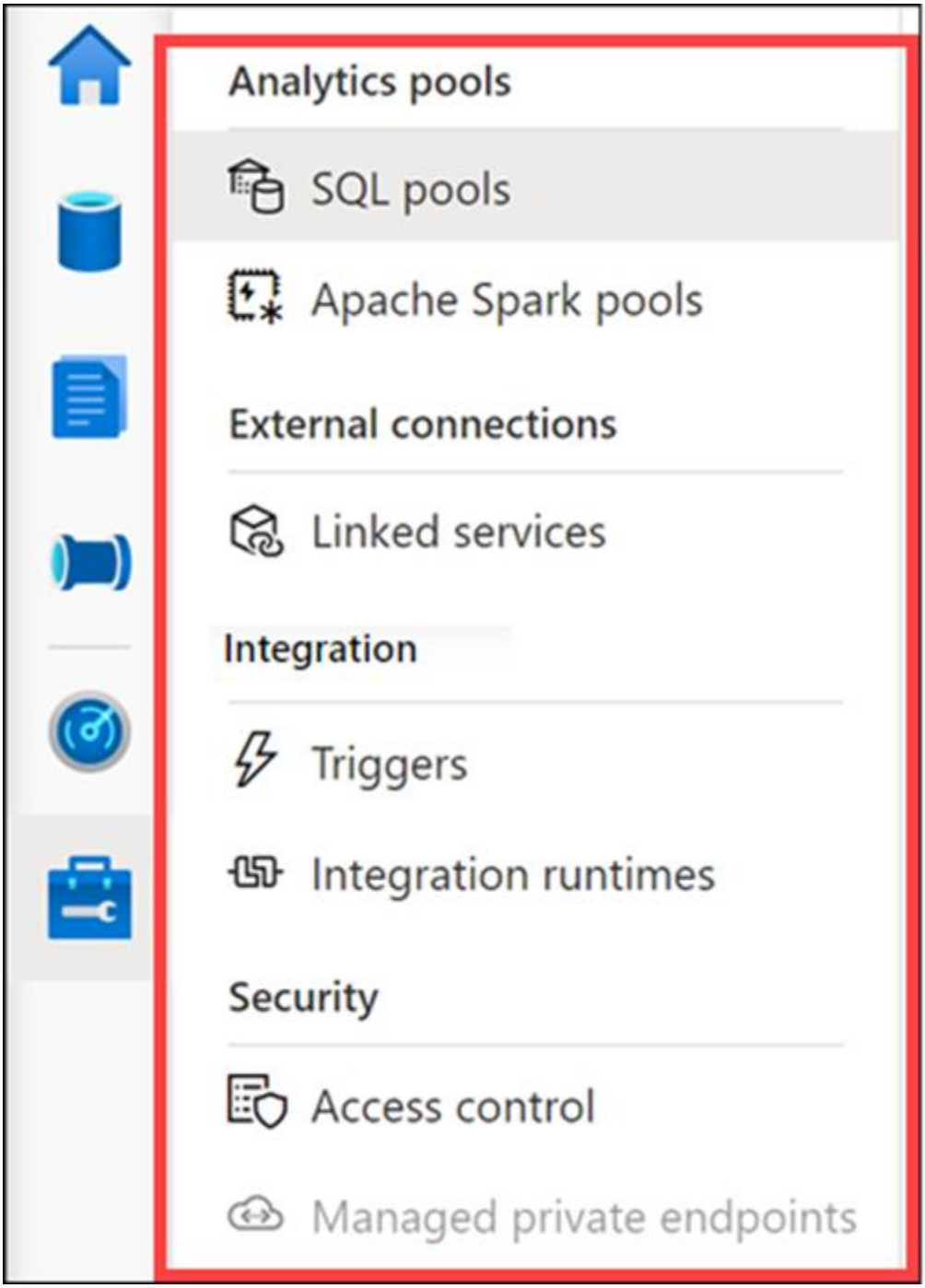
- **Monitor hub**
- **Data hub**
- **Manage hub**

  (Correct)
- **None of the listed options**
- **Create hub**

(Incorrect)

Explanation

In Azure Synapse Studio, the Manage hub enables you to perform some of the same actions available 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.



• **SQL pools**. Lists the provisioned SQL pools and on-demand SQL serverless pools for the workspace. You can add new pools or hover over a SQL pool to **pause** or **scale** it. You should pause a SQL pool when it's not being used to save costs.

• **Apache Spark pools**. Lets you manage your Spark pools by configuring the auto-pause and auto-scale settings. You can provision a new Apache Spark pool from this blade.

• **Linked services**. Enables you to manage connections to external resources. Here you can see linked services for our data lake storage account, Azure Key Vault, Power BI, and Synapse Analytics. **Task**: Select **+ New** to show how many types of linked services you can add.

• **Triggers**. Provides you a central location to create or remove pipeline triggers. Alternatively, you can add triggers from the pipeline.

• **Integration runtimes**. Lists the IR for the workspace, which serves as the compute infrastructure for data integration capabilities, like those provided by pipelines. **Task**: Hover over the integration runtimes to show the monitoring, code, and delete (if applicable) links. Click on a **code link** to show how you can modify the parameters in JSON format, including the TTL (time to live) setting for the IR.

• **Access control**. This is where you go to add and remove users to one of three security groups: workspace admin, SQL admin, and Apache Spark for Azure Synapse Analytics admin.

• **Managed private endpoints**. This is where you manage private endpoints, which use a private IP address from within a virtual network to connect to an Azure service or your own private link service. Connections using private endpoints listed here provide access to Synapse workspace endpoints (SQL, SqlOndemand and Dev).

https://techcommunity.microsoft.com/t5/azure-synapse-analytics/explore-the-manage-hub-in-synapse-studio-to-provision-and-secure/ba-p/1987788

Question 43: **Incorrect**
Identify the missing word(s) in the following sentence within the context of Microsoft Azure.

Microsoft Azure Storage is a managed service that provides durable, secure, and scalable storage in the cloud. A single Azure subscription can host up to [A] TB storage accounts, each of which can hold [B] PiB of data.

- **[A] 500, [B] 2.5**
- **[A] 250, [B] 5**

(Correct)

- **[A] 500, [B] 10**

- **[A] 200, [B] 5**

(Incorrect)

Explanation
Scale targets for standard storage accounts

The following table describes default limits for Azure general-purpose v1, v2, Blob storage, and block blob storage accounts. The *ingress* limit refers to all data that is sent to a storage account. The *egress* limit refers to all data that is received from a storage account.

Resource	Limit
Number of storage accounts per region per subscription, including standard, and premium storage accounts.	250
Maximum storage account capacity	5 PiB [1]
Maximum number of blob containers, blobs, file shares, tables, queues, entities, or messages per storage account	No limit
Maximum request rate[1] per storage account	20,000 requests per second
Maximum ingress[1] per storage account (US, Europe regions)	10 Gbps
Maximum ingress[1] per storage account (regions other than US and Europe)	5 Gbps if RA-GRS/GRS is enabled, 10 Gbps for LRS/ZRS[2]
Maximum egress for general-purpose v2 and Blob storage accounts (all regions)	50 Gbps
Maximum egress for general-purpose v1 storage accounts (US regions)	20 Gbps if RA-GRS/GRS is enabled, 30 Gbps for LRS/ZRS[2]
Maximum egress for general-purpose v1 storage accounts (non-US regions)	10 Gbps if RA-GRS/GRS is enabled, 15 Gbps for LRS/ZRS[2]
Maximum number of virtual network rules per storage account	200
Maximum number of IP address rules per storage account	200

https://docs.microsoft.com/en-us/azure/storage/common/scalability-targets-standard-account

Question 44: **Incorrect**

Init Scripts provide a way to configure cluster's nodes. It is recommended to favour Cluster Scoped Init Scripts over Global and Named scripts.

Which of the following is best described by:

"By placing the init script in `/databricks/init_folder`, you force the script's execution every time any cluster is created or restarted by users of the workspace."

- ○

Cluster Named

- Global

  (Correct)

- Cluster Scoped

- Interactive

  (Incorrect)

Explanation
Favour cluster scoped init scripts over global and named scripts

Init Scripts provide a way to configure cluster's nodes and to perform custom installs. Init scripts can be used in the following modes:

• **Global:** by placing the Init script in `/databricks/init` folder, you force the script's execution every time any cluster is created or restarted by users of the workspace.

• **Cluster Named (deprecated):** you can limit the init script to run only on for a specific cluster's creation and restarts by placing it in `/databricks/init/<cluster_name>` folder.

• **Cluster Scoped**: in this mode, the Init script is not tied to any cluster by its name and its automatic execution is not a virtue of its dbfs location. Rather, you specify the script in cluster's configuration by either writing it directly in the cluster configuration UI or storing it on Databricks File System (DBFS) and specifying the path in Cluster Create API. Any location under `DBFS /databricks` folder except `/databricks/init` can be used for this purpose, such as: `/databricks/<my-directory>/set-env-var.sh`

You should treat Init scripts with *extreme* caution because they can easily lead to intractable cluster launch failures. If you really need them, please use the **Cluster Scoped execution mode** as much as possible because:

• ADB executes the script's body in each cluster node. Thus, a successful cluster launch and subsequent operation are predicated on all nodal Init scripts executing in a timely manner without any errors and reporting a zero exit code. This process is highly error prone, especially for scripts downloading artifacts from an external service over unreliable and/or misconfigured networks.

• Because Global and Cluster Named Init scripts execute automatically due to their placement in a special DBFS location, it is easy to overlook that they could be causing a cluster to not launch. By specifying the Init script in the Configuration, there's a higher chance that you'll consider them while debugging launch failures.

Use cluster log delivery feature to manage logs

By default, Cluster logs are sent to default DBFS but you should consider sending the logs to a blob store location under your control using the Cluster Log Delivery feature. The Cluster Logs contain logs emitted by user code, as well as Spark framework's Driver and Executor logs. Sending them to a blob store controlled by yourself is recommended over default DBFS location because:

• ADB's automatic 30-day default DBFS log purging policy might be too short for certain compliance scenarios. A blob store location in your subscription will be free from such policies.

• You can ship logs to other tools only if they are present in your storage account and a resource group governed by you. The root DBFS, although present in your subscription, is launched inside a Microsoft Azure managed resource group and is protected by a read lock. Because of this lock, the logs are only accessible by privileged Azure Databricks framework code. However, constructing a pipeline to ship the logs to downstream log analytics tools requires logs to be in a lock-free location first.

https://github.com/Azure/AzureDatabricksBestPractices/blob/master/toc.md

Question 45: **Incorrect**

Scenario: Honest Eddie's Car Dealership is an establishment in South Carolina USA, which is dedicated to the purchase and sale of cars and light trucks. Eddie has hired you as an expert consultant for Azure projects and you are holding a workgroup session with the team discussing inputs for Azure Stream Analytics jobs. The team has collectively created a list of key benefits of using Azure Stream Analytics to process streaming data. Eddie thinks that some of the responses are not correct and has asked you to confirm which are not correct.

Which of the following statements are not primary benefits of using Azure Stream Analytics to process streaming data?

- **The ability to write and test transformation queries in the Azure portal**
-

All of the listed options are primary benefits

- **Being able to rapidly deploy queries into production by creating and starting an Azure Stream Analytics job**

  (Incorrect)

- **Integration with Azure Blob storage**

  (Correct)

Explanation
Integration with Azure Blob storage is not one of the primary benefits of using Azure Stream Analytics to process streaming data. The integration with Blob storage can be used to process static data.

Azure Stream Analytics, the recommended service for stream analytics on Azure, easily integrates with your applications and connected devices and sensors to transform streaming data. The process of consuming data streams, analyzing them, and deriving actionable insights is called stream processing. You can transform streaming data using the SQL-like Stream Analytics Query Language to perform temporal and other aggregations against a data stream to gain insights. Streaming data inputs include Azure Event Hubs and IoT Hub. And static data held in Blob storage can also be processed using Stream Analytics jobs.

Stream processing refers to the continuous ingestion, transformation, and analysis of data streams generated by applications, IoT devices and sensors, and other sources to derive actionable insights in near-real-time. Data stream analysis frequently involves using temporal operations, such as windowed aggregates, temporal joins, and temporal analytic functions to measure changes or differences over time. The intent being to:

• Continuously monitor data using time-boxes windows to understand better how specific areas of interest change or fluctuate over time

• Identify and react to anomalies or irregularities within data in real-time

• Perpetually analyze new data to identify and respond to issues in real-time

• Trigger specific actions when certain thresholds are identified

The exponential propagation of connected applications, devices, and sensors has fuelled the necessity for organizations to analyze streaming data as it arrives and use the latent knowledge contained within the data to make business decisions in near-real-time. Some example use cases of streaming data analysis include:

• Anomaly detection to identify potentially fraudulent transactions in finance industries

• Making product recommendations to online customers in real-time

• Monitoring pipelines and distribution systems by oil companies

• Generating predictive maintenance schedules for industrial and manufacturing equipment

• Sentiment analysis of social media posts

Approaches to data stream processing

The primary approach to stream processing is to analyze new data continuously, transforming incoming data as it arrives to facilitate near-real-time insights. Computations and aggregations can be executed against the data using temporal analysis and sent to a Power BI dashboard for real-time visualization and analysis. This approach typically involves persisting the streaming data into a data store, such as Azure Data Lake Storage (ADLS) Gen2, for further examination or more advanced analytics workloads.

An alternative approach for processing streaming data is to persist incoming data in a data store, such as Azure Data Lake Storage (ADLS) Gen2. You can then process the *static* data in batches at a later time. This approach is frequently used to take advantage of lower compute costs when processing large sets of existing data.

Azure Stream Analytics is the recommended service for stream analytics on Azure. Stream Analytics provides you with the ability to ingest, process, and analyze streaming data from Azure Event Hubs (including Azure Event Hubs from Apache Kafka) and Azure IoT Hub. You can also configure static data ingestion from Azure Blob Storage. This integration allows you to quickly create hot-path analytics pipelines to generate powerful insights to drive real-time actions. Azure Stream Analytics is meant for a wide range of scenarios that include, but aren't limited to:

• Dashboards for data visualization

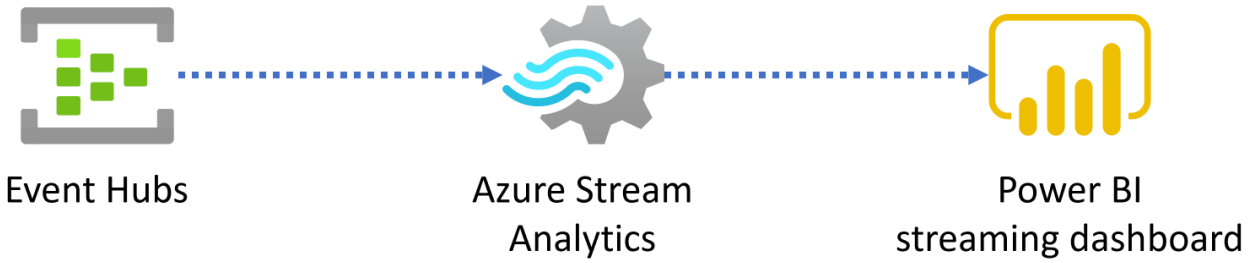
•Real-time alerts from temporal and spatial patterns or anomalies

• Extract, Transform, Load (ETL)

• Event Sourcing pattern

• IoT Edge

Contoso collects vehicle telemetry and wants to use Event Hubs to ingest and store the data in its raw form. They want to perform several aggregations on the telemetry data in near-real-time. In the end, they would like to visualize the aggregated data on a dashboard that automatically updates with new data as it arrives. They want the dashboard to contain various visualizations of detected anomalies, like engines overheating, abnormal oil pressure, and aggressive driving. They are interested in utilizing a mapping component to show irregularities related to locations, as well as various charts and graphs depicting this information. The CIO asked you to assist them in setting up a near-real-time analytics pipeline built on Event Hubs, Azure Stream Analytics, and Power BI.



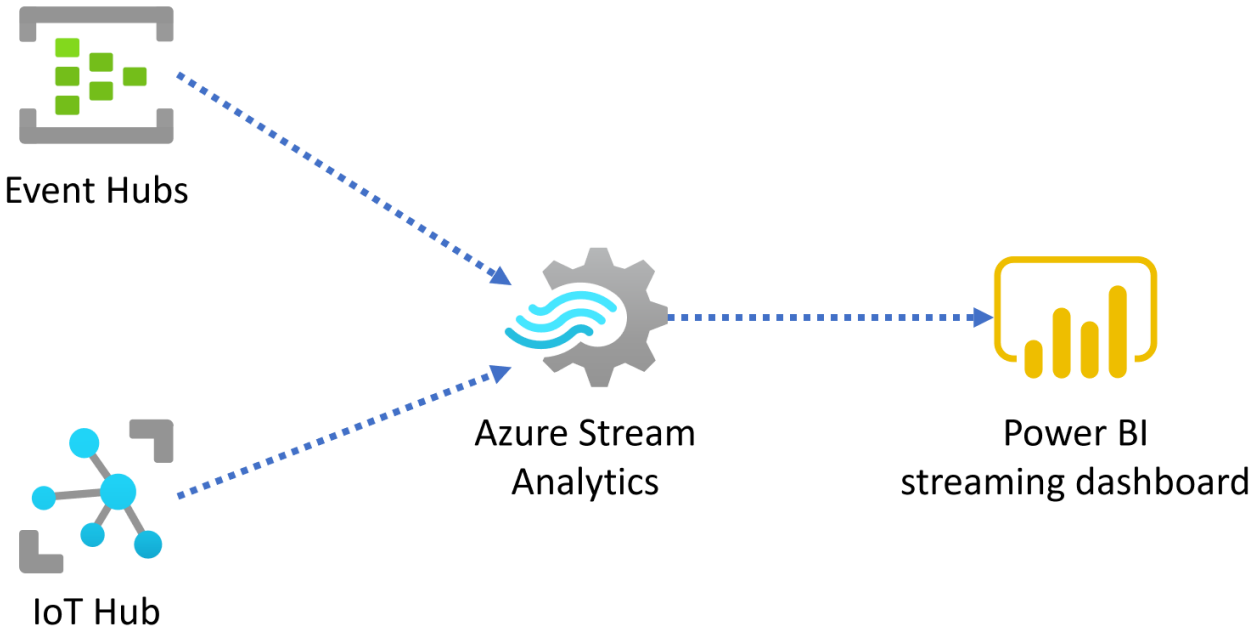
The fastest way to get streaming analytics running in Azure is to add an Azure Stream Analytics job to your application. Your Stream Analytics job would ingest the streaming data from one of the supported inputs and run real-time analytics queries against the streams. The built-in integration with Azure Event Hubs and IoT Hub provides a rapid mechanism to create these streaming analytics pipelines. Stream Analytics also supports various inputs and outputs. It also provides the capability to use Azure Machine Learning functions to make it a robust tool for analyzing data streams. The primary benefits of processing streaming data with Azure Stream Analytics include:

• The ability to **preview and visualize incoming data** directly in the Azure portal.

• Using the Azure portal to **write and test your transformation queries** using the SQL-like Stream Analytics Query Language (SAQL). You can use the built-in functions of SAQL to find interesting patterns from the incoming stream of data.

• Rapid **deployment of your queries into production** by creating and starting an Azure Stream Analytics job.

With Azure Stream Analytics, you can quickly stand up real-time dashboards and alerts. An example of a simple solution, as depicted in the diagram below, includes ingesting streaming data from Event Hubs or IoT Hub. The streaming data can then be transformed using Stream Analytics windowing queries. The aggregated data are then sent to a Power BI dashboard with a streaming data set.



The rich out-of-the-box functionality of Azure Stream Analytics allows you to immediately take advantage of the following features without any additional setup:

• Built-in temporal operators, such as windowed aggregates, temporal joins, and temporal analytic functions

• Native Azure input and output adapters

• Support for slow-changing reference data (also known as lookup tables), including joining with geospatial reference data for geofencing

• Integrated solutions, such as Anomaly Detection

• Multiple time windows in the same query

• Ability to compose multiple temporal operators in arbitrary sequences

Operational aspects

Azure Stream Analytics is a PaaS job service, so it is fully managed and highly reliable. You do not have to spend time managing clusters or worrying about downtime. Job-level billing ensures low startup costs (three Streaming Units, by default). And, jobs are scalable up to 192 Streaming Units to provide the performance necessary to run even the most demanding jobs effectively. It's much more cost-effective to run a few Stream Analytics jobs than run and maintain a Spark cluster.

Streaming Units (SUs) represents the computing resources designated to execute Stream Analytics jobs. Increasing the number of SUs means more CPU and memory resources are allocated to the job. Azure Stream Analytics jobs perform all processing in memory to achieve the low latency required for efficient stream processing. Handling compute capacity in this manner allows you to focus on writing queries and leaves hardware management to Microsoft.

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

Question 46: **Incorrect**
Identify the missing word(s) in the following sentence within the context of Microsoft Azure.

A(n) [?] schema must be defined before query time.

- ○

  Structured data type

  (Correct)

- ○

  Hybrid data type

- ○

  Azure Cosmos DB data type

- ◉

  Unstructured data type

  (Incorrect)

Explanation
Structured data

In relational database systems like Microsoft SQL Server, Azure SQL Database, and Azure SQL Data Warehouse, data structure is defined at design time. Data structure is designed in the form of tables. This means it's designed before any information is loaded into the system. The data structure includes the relational model, table structure, column width, and data types.

Relational systems react slowly to changes in data requirements because the structural database needs to change every time a data requirement changes. When new columns are added, you might need to bulk-update all existing records to populate the new column throughout the table.

Relational systems typically use a querying language such as Transact-SQL (T-SQL).

https://k21academy.com/microsoft-azure/dp-900/relational-and-non-relational-datastores/

Question 47: **Incorrect**
Identify the missing word(s) in the following sentence within the context of Microsoft Azure.

[?] is a fully managed cloud service. Analysts, data scientists, developers, and others use [?] to discover, understand, and consume data sources. It 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.

- **Azure Databricks**
- **Azure Cosmos DB**
- **Azure Data Catalog**

  (Correct)
-

- **Azure Data Factory**

- **Azure Data Lake Storage**

  (Incorrect)

- **Azure SQL Datawarehouse**

- **Azure Storage Explorer**

Explanation

Azure Data Catalog

Analysts, data scientists, developers, and others use Data Catalog to discover, understand, and consume data sources. Data Catalog 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 Catalog 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

Question 48: **Correct**
Which Transact-SQL function verifies if a piece of text is valid JSON?

- `JSON_VALUE`

- **None of the listed options**

- `JSON_QUERY`

- `JSON_VALID`

- `ISJSON`

  (Correct)

Explanation
`ISJSON` is a Transact-SQL function that verifies if a piece of text is valid JSON.

https://docs.microsoft.com/en-us/sql/t-sql/functions/isjson-transact-sql?view=sql-server-ver15

Question 49: **Incorrect**
Scenario: You are new on the job and are looking through the Azure knowledgebase to determine which Azure product is the right choice for an ingestion point for data streaming in an event processing solution that uses static data as a source.

You narrowed the choices down to the below list.

Which is the best choice?

- **Azure IoT Hub**

- **Azure Event Hubs**

- **Azure Sphere**

- **Azure Blob Storage**

  (Correct)

- **Azure IoT Central**

  (Incorrect)

Explanation
Azure Blob storage provides an ingestion point for data streaming in an event processing solution that uses static data as a source.

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

Question 50: **Incorrect**
Scenario: You are working in an Azure Databricks workspace and you want to filter based on the end of a column value using the Column Class. Specifically, you are looking at a column named verb and filtered by words ending with *"ing"*.

Which command filters based on the end of a column value as required?

- `df.filter("verb like '_ing'")`

- `df.filter().col("verb").like("%ing")`

- `df.filter(col("verb").endswith("ing"))`

  (Correct)

- `df.filter("verb like '%ing'")`

  (Incorrect)

Explanation
The Column Class supports both the `endswith()` method and the like() method (example - `col("verb").like("%ing")`).

https://docs.databricks.com/spark/latest/dataframes-datasets/introduction-to-dataframes-python.html

Question 51: **Incorrect**

Large data projects can be complex. The projects often involve hundreds of decisions. Multiple people are typically involved, and each person helps take the project from design to production.

Roles such as business stakeholders, business analysts, and business intelligence developers are well known and valuable.

Which of the available roles is best described by:

"Works with artificial intelligence services such as Cognitive Services, Cognitive Search, and Bot Framework. Cognitive Services includes Computer Vision, Text Analytics, Bing Search, and Language Understanding (LUIS).

Rather than creating models, they apply the pre-built capabilities of Cognitive Services APIs. Part of their job is to embed these capabilities within a new or existing application or bot. They rely on the expertise of data engineers to store information that's generated from artificial intelligence."

- **BI Engineer**
- **AI Engineer**

  (Correct)
- **Project Manager**

  (Incorrect)
- **Data Engineer**
-

RPA Developers

- **Data Scientist**

- **Solution Architects**

- **System Administrators**

Explanation

AI Engineer

AI engineers work with AI services such as Cognitive Services, Cognitive Search, and Bot Framework. Cognitive Services includes Computer Vision, Text Analytics, Bing Search, and Language Understanding (LUIS).

Rather than creating models, AI engineers apply the pre-built capabilities of Cognitive Services APIs. AI engineers embed these capabilities within a new or existing application or bot. AI engineers rely on the expertise of data engineers to store information that's generated from AI.

AI engineers add the intelligent capabilities of vision, voice, language, and knowledge to applications. To do this, they use the Cognitive Services offerings that are available out of the box.

When a Cognitive Services application reaches its capacity, AI engineers call on data scientists. Data scientists develop machine learning models and customize components for an AI engineer's application.

For example, an AI engineer might be working on a Computer Vision application that processes images. This AI engineer would ask a data engineer to provision an Azure Cosmos DB instance to store the metadata and tags that the Computer Vision application generates.

https://www.whizlabs.com/blog/azure-data-engineer-roles/

Question 52: **Incorrect**
Identify the missing word(s) in the following sentence within the context of Microsoft Azure.

Azure provides many ways to store your data. A storage account is a(n) [?] that groups a set of Azure Storage services together.

- **VM**
- **Container**

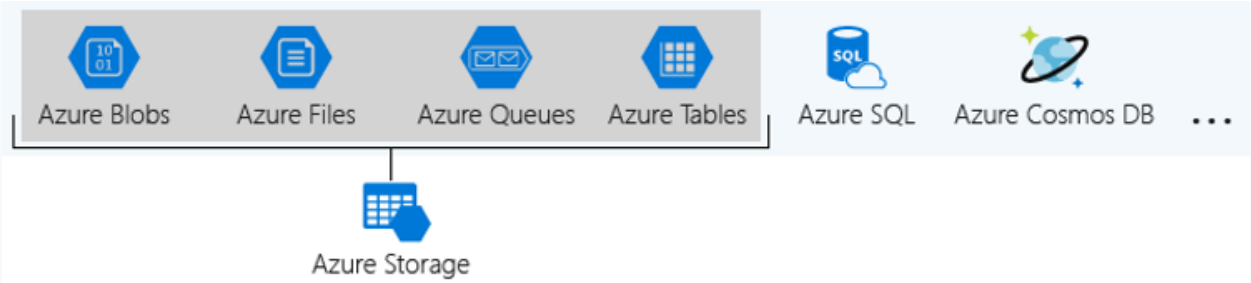
  (Correct)
- **Unstructured dataset**
- **Blob**
- **Structured dataset**

  (Incorrect)

Explanation
What is Azure Storage?

Azure provides many ways to store your data. There are multiple database options like Azure SQL Database, Azure Cosmos DB, and Azure Table Storage. Azure offers multiple ways to store and send messages, such as Azure Queues and Event Hubs. You can even store loose files using services like Azure Files and Azure Blobs.

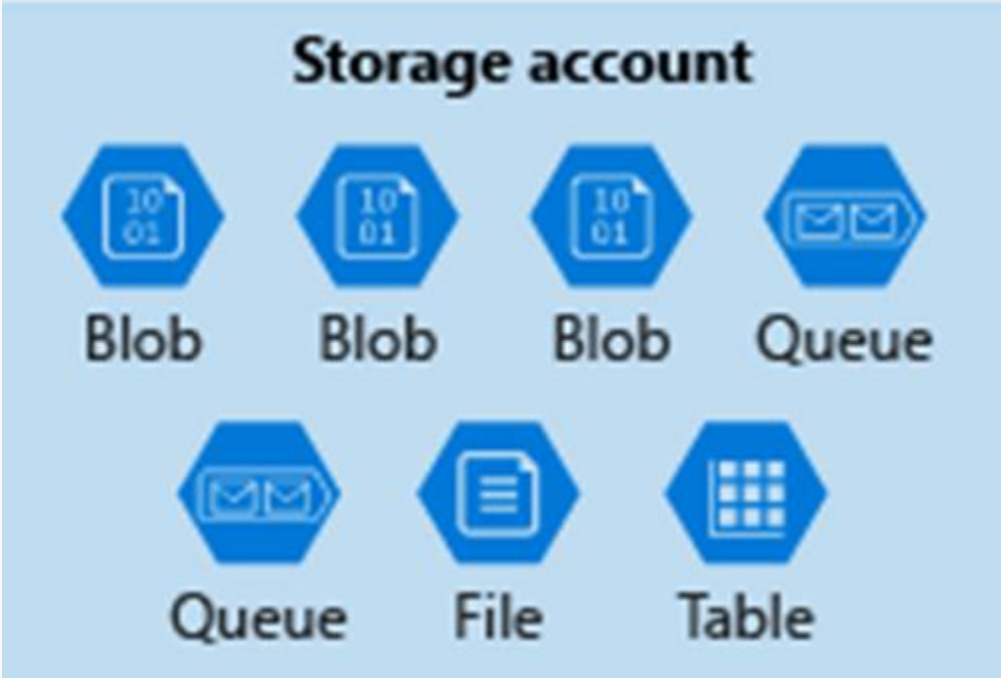
Azure selected four of these data services and placed them together under the name *Azure Storage*. The four services are Azure Blobs, Azure Files, Azure Queues, and Azure Tables. The following illustration shows the elements of Azure Storage.



These four were given special treatment because they are all primitive, cloud-based storage services and are often used together in the same application.

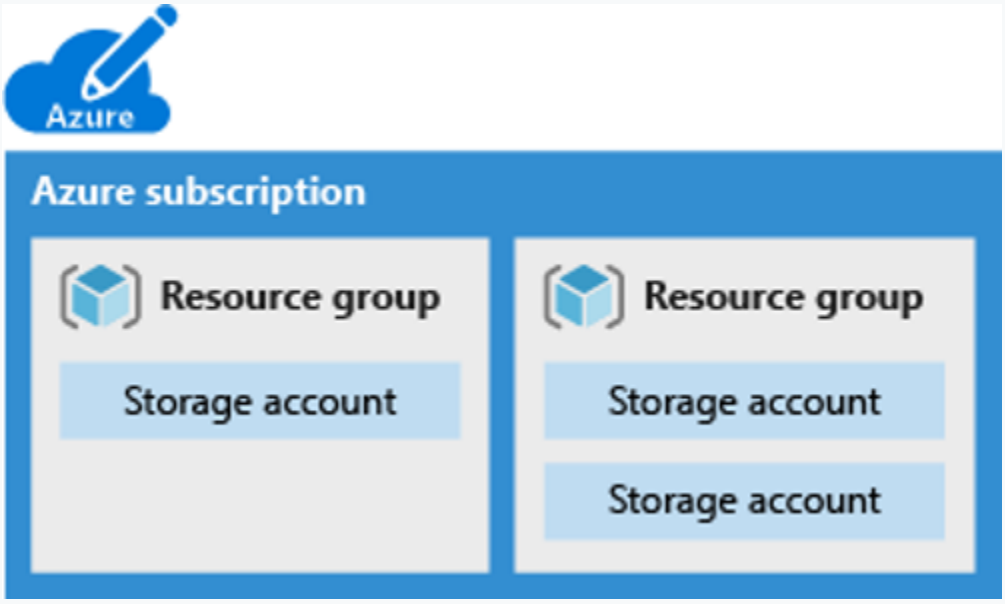
What is a storage account?

A *storage account* is a container that groups a set of Azure Storage services together. Only data services from Azure Storage can be included in a storage account (Azure Blobs, Azure Files, Azure Queues, and Azure Tables). The following illustration shows a storage account containing several data services.

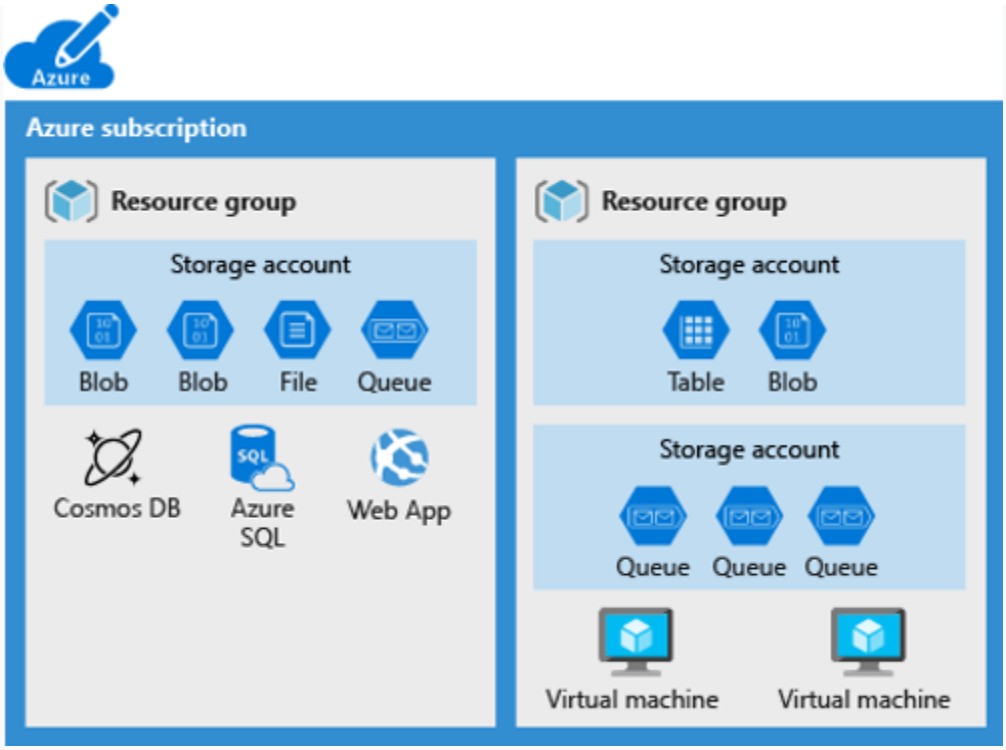


Combining data services into a storage account lets you manage them as a group. The settings you specify when you create the account, or any that you change after creation, are applied to everything in the account. Deleting the storage account deletes all of the data stored inside it.

A storage account is an Azure resource and is included in a resource group. The following illustration shows an Azure subscription containing multiple resource groups, where each group contains one or more storage accounts.



Other Azure data services like Azure SQL and Azure Cosmos DB are managed as independent Azure resources and cannot be included in a storage account. The following illustration shows a typical arrangement: Blobs, Files, Queues, and Tables are inside storage accounts, while other services are not.



https://www.c-sharpcorner.com/article/what-is-microsoft-azure-storage/

Question 53: **Incorrect**

Scenario: Dr. Karl Malus works for the Power Broker Corporation (PBC) founded by Curtiss Jackson, using technology to service various countries and their military efforts. You have been contracted by the company to assist Dr. Malus with their Microsoft Azure implementation.

As the Azure expert, Dr. Malus asks you to describe the Delta Lake concept to him and the team.

Which of the following descriptions should you provide?

- **An Azure Cosmos Service for exporting data from a relational database into CSV files.**

- **A relational storage layer for Spark that supports tables based on Parquet files.**

  (Correct)

-

A Spark API for exporting data from a relational database into CSV files.

- **A synchronization solution that replicates data between SQL pools and Spark pools.**

  (Incorrect)

Explanation
Delta Lake provides a relational storage layer in which you can create tables based on Parquet files in a data lake.

Delta Lake is an open-source storage layer that adds relational database semantics to Spark-based data lake processing. Delta Lake is supported in Azure Synapse Analytics Spark pools for PySpark, Scala, and .NET code.

The benefits of using Delta Lake in a Synapse Analytics Spark pool include:

• **Relational tables that support querying and data modification**. With Delta Lake, you can store data in tables that support *CRUD* (create, read, update, and delete) operations. In other words, you can *select*, *insert*, *update*, and *delete* rows of data in the same way you would in a relational database system.

• **Support for *ACID* transactions**. Relational databases are designed to support transactional data modifications that provide *atomicity* (transactions complete as a single unit of work), *consistency* (transactions leave the database in a consistent state), *isolation* (in-process transactions can't interfere with one another), and *durability* (when a transaction completes, the changes it made are persisted). Delta Lake brings this same transactional support to Spark by implementing a transaction log and enforcing serializable isolation for concurrent operations.

• **Data versioning and *time travel***. Because all transactions are logged in the transaction log, you can track multiple versions of each table row and even use the *time travel* feature to retrieve a previous version of a row in a query.

• **Support for batch and streaming data**. While most relational databases include tables that store static data, Spark includes native support for streaming data through the Spark Structured Streaming API. Delta Lake tables can be used as both *sinks* (destinations) and *sources* for streaming data.

• **Standard formats and interoperability**. The underlying data for Delta Lake tables is stored in Parquet format, which is commonly used in data lake ingestion pipelines. Additionally, you can use the serverless SQL pool in Azure Synapse Analytics to query Delta Lake tables in SQL.

https://learn.microsoft.com/en-us/azure/synapse-analytics/spark/apache-spark-what-is-delta-lake

Question 54: **Incorrect**
Scenario: You have been assigned to a new project and your first task is to initialize the Blob Storage client library within an application.

Which of the following can be used to do this?

- ○

  The Azure Storage account datacentre and location identifiers.

- ○

  A globally-unique identifier (GUID) that represents the application.

- ○

  The Azure Storage account connection string.

  (Correct)

- ◉

  An Azure username and password.

  (Incorrect)

Explanation
A storage account connection string contains all the information needed to connect to Blob storage, most importantly the account name and the account key.

https://docs.microsoft.com/en-us/azure/storage/common/storage-configure-connection-string

Question 55: **Incorrect**
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.

An Azure integration runtime is capable of which of the following? (Select all that apply)

- ☐

  Triggering batch movement of ETL data on a dynamic schedule for most analytics solutions.

- ☐

  Dispatching transform activities in public network utilizing platforms such as Databricks Notebook/ Jar/ Python activity, HDInsight Hive activity and more.

  (Correct)

- ☐

  Running Copy Activity between cloud data stores

  (Correct)

- ☑

  All the listed options.

  (Incorrect)

- ☐

  None of the listed options.

- ☐

  Running Data Flows in Azure

  (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 bridge between the activity and linked services.

Azure integration runtime

An Azure integration runtime is capable of:

• Running Data Flows in **Azure**

• Running Copy Activity **between cloud data stores**

• Dispatching the following transform activities in **public network**: Databricks Notebook/ Jar/ Python activity, HDInsight Hive activity, HDInsight Pig activity, HDInsight MapReduce activity, HDInsight Spark activity, HDInsight Streaming activity, Machine Learning Batch Execution activity, Machine Learning Update Resource activities, Stored Procedure activity, Data Lake Analytics U-SQL activity, .NET custom activity, Web activity, Lookup activity, and Get Metadata activity.

You can set a certain location of an Azure IR, in which case the data movement or activity dispatch will happen in that specific region. If you choose to use the auto-resolve Azure IR which is the default, ADF will make a best effort to automatically detect your sink and source data store to choose the best location either in the same region if available or the closest one in the same geography for the Copy Activity. For anything else, it will use the IR in the Data Factory region. Azure Integration Runtime also has support for virtual networks.

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

Question 56: **Correct**
Microsoft Purview is a unified data-governance service that helps you manage and govern your on-premises, multicloud, and software-as-a-service (SaaS) data. You can easily create a holistic, up-to-date map of your data landscape with automated data discovery, sensitive data classification, and end-to-end data lineage. You can also empower data consumers to find valuable, trustworthy data.

Where would you register your data sources for use in Microsoft Purview?

- ○ **In the Managed Endpoints tab of the Microsoft Purview account page**
- ○ **In the Managed Resources tab of the Microsoft Purview account page**
- ◉ **In the Microsoft Purview governance portal**

  (Correct)

-

In the Overview tab of the Microsoft Purview account page

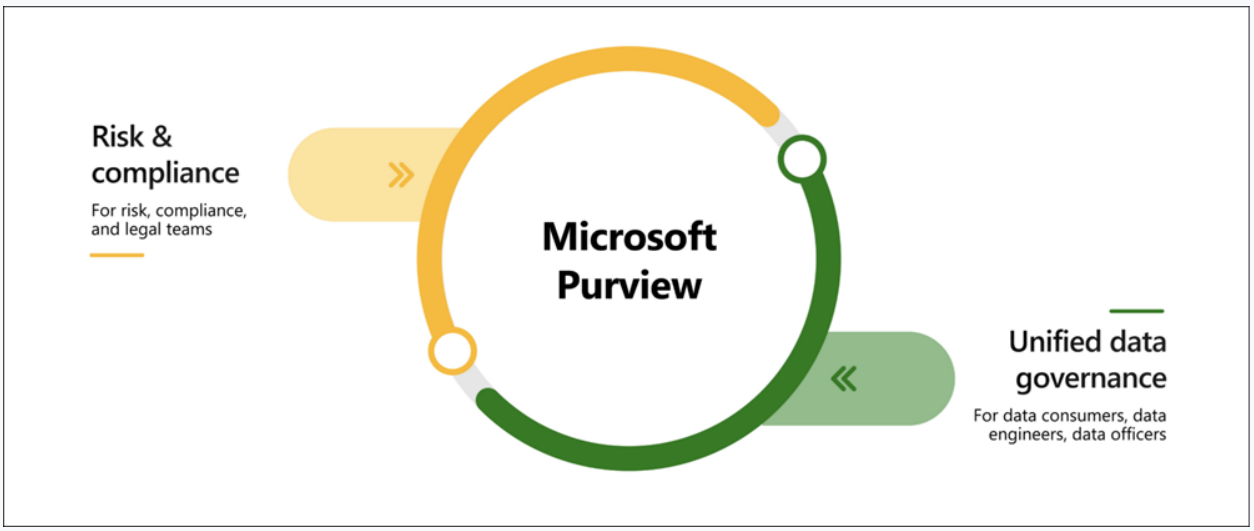
Explanation

- *The Microsoft Purview governance portal is where you register your data sources.*
- The overview page for the Microsoft Purview account provides information about that account only.
- The Managed resources tab is used to configure and manage the resource group, storage account, and event hub resources.
- The Managed Endpoints tab is an invalid option

Microsoft Purview

Loading data in the Data Map

The Purview Data Map is a unified map of your data assets and their relationships that enables more effective governance for your data estate. It's a knowledge graph that's the underpinning for the Purview Data Catalogue and all the features that it has to offer. It's scalable and robust to meet your enterprise-compliance requirements.

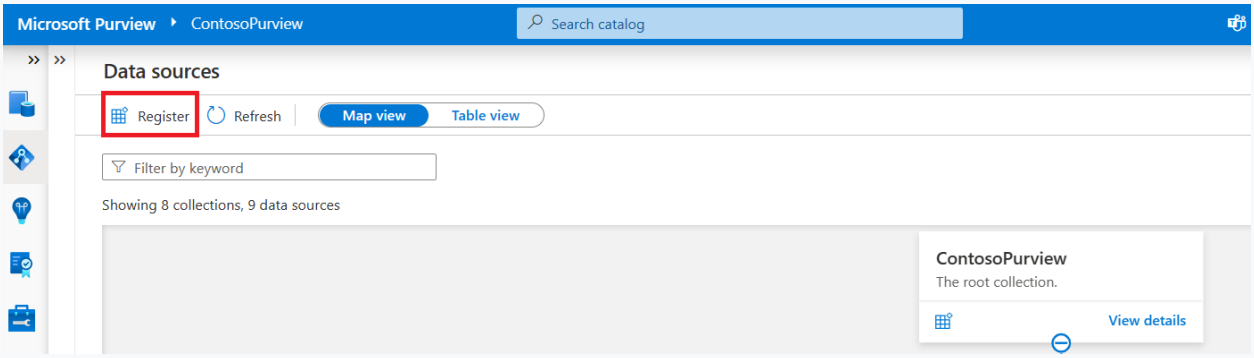


Sourcing data

Sourcing your data starts with a process where you register data sources. Microsoft Purview supports an array of data sources that span on-premises, multicloud, and

software-as-a-service (SaaS) options. You'll register the various data sources so Microsoft Purview is aware of them. The data remains in its location and isn't migrated to any other platform.

You'll register a data source from within the Azure portal. Once you have a Microsoft Purview service configured in Azure, you'll use the Microsoft Purview governance portal to register your data sources.



Each type of data source you choose will require specific information to complete the registration. For example, if your data sources reside in your Azure subscription, you'll choose the necessary subscription and storage account name. The following image is an example of choosing an Azure Blob Storage source:

Each data source will have specific requirements for authenticating and configuration to permit scanning of the assets in that data source. If you have data stored in an Amazon S3 standard bucket, you'll need to provide a configuration for the connection. For this service, you'll use Purview to provide a Microsoft account with secure access to AWS, where the Purview scanner will run. The Purview scanner uses this access to your Amazon S3 buckets to read your data, then reports the scanning results (including only the metadata and classification) back to Azure. You can use the Purview classification and labelling reports to analyze and review your data scan results.

https://learn.microsoft.com/en-us/azure/purview/register-scan-amazon-s3

In Microsoft Purview, there are a few options to use for authentication when the service needs to scan data sources. Some of these options are:

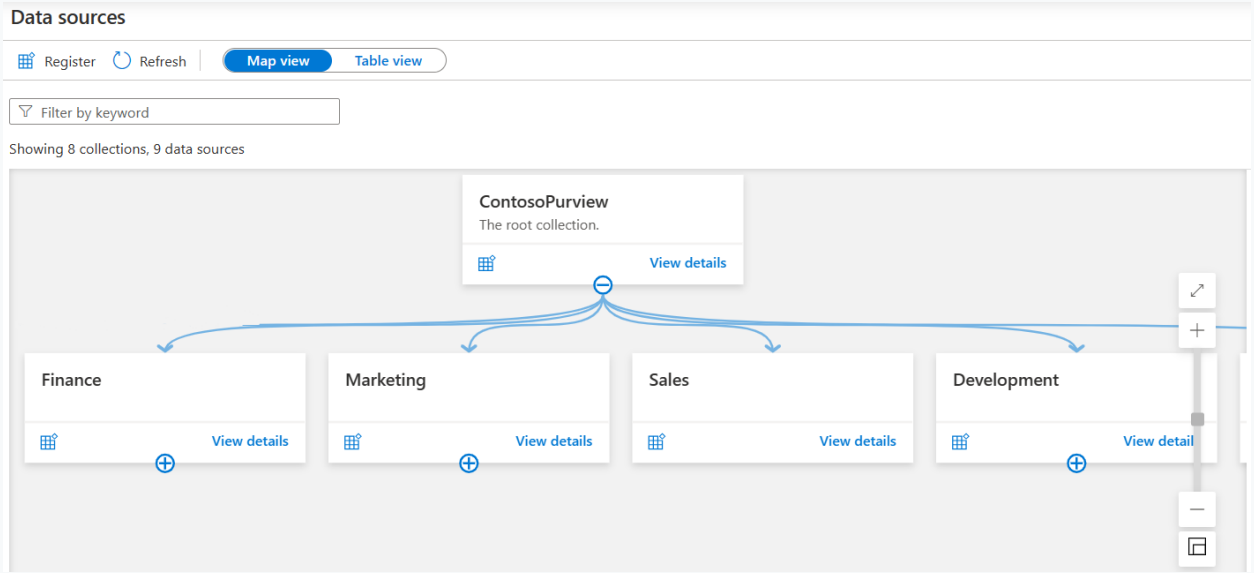
- Microsoft Purview Managed Identity
- Account Key (using Key Vault)
- SQL Authentication (using Key Vault)
- Service Principal (using Key Vault)

Mapping data

A data map is the foundational platform for Microsoft Purview. Data Map = Data assets + Lineage + Classifications + Business Context. Customers create a knowledge graph of data coming in from a range of sources. Purview makes it easy to register and automatically scan and classify data at scale. Within the data map, you can identify the type of data source, along with other details around security, scanning, and so on. The data map allows you to use collections as well.

Collections are a way of grouping data assets into logical categories (collections) to simplify management and discovery of assets within the catalogue.

Selecting the Map view displays the data sources in a graphical view, along with the collections you've created for them.

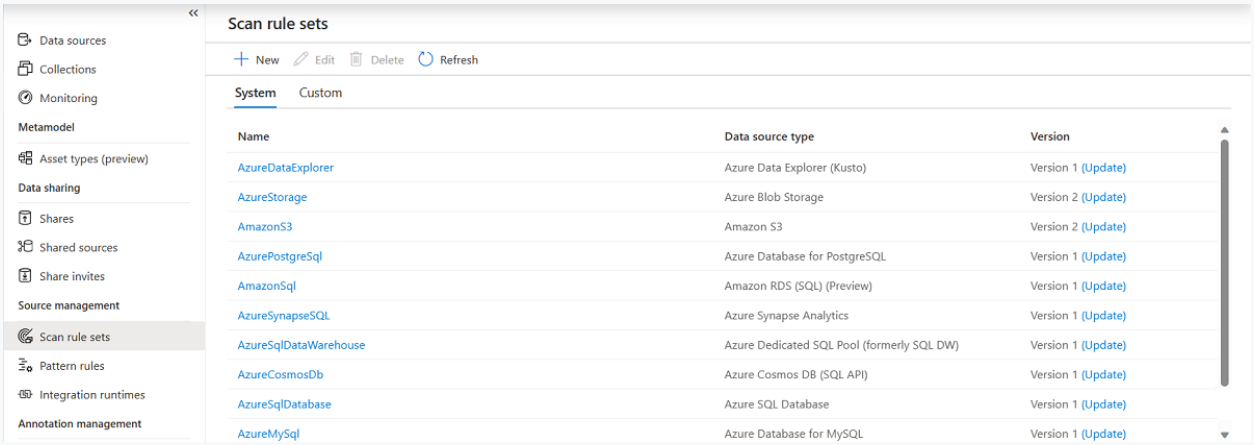


Scanning data

Once you've registered your data sources, you'll need to run a scan to be able to access the metadata and browse the asset information. You can configure scan rules for the data you want to scan. In a Microsoft Purview catalogue, you can create scan rule sets to enable you to quickly scan data sources in your organization.

A scan rule set is a container for grouping a set of scan rules together so that you can easily associate them with a scan. A scan rule set lets you select file types for schema extraction and classification, and also lets you define new custom file types. For example, you might create a default scan rule set for each of your data source types, then use these scan rule sets by default for all scans within your company. You might also want users with the right permissions to create other scan rule sets with different configurations based on business need.

Scanning the data sources will require the entry of credentials for these sources. You can store the credentials in an Azure Key Vault for security and ease of access by your scan rules. The Microsoft Purview governance portal comes with existing system scan rule sets that you can select when creating a new scan rule, or you can specify a custom scan rule set.



You can also select the file types to include in the scan. The file types will be used in the schema extraction and classification of the data. Built-in file-type support includes:

- CSV
- JSON

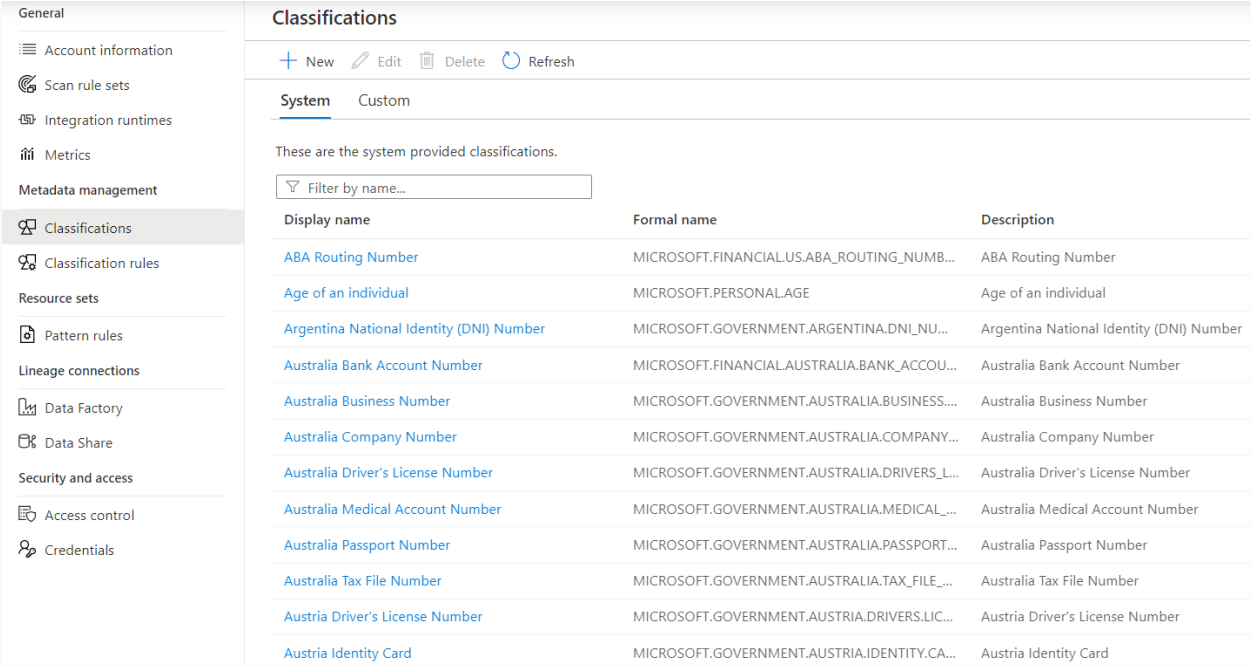
- PSV
- SSV
- TSV
- TXT
- XML
- PARQUET
- AVRO
- ORC
- Document file types (.doc, pdf, xls, etc.)

You can select which system and custom classification rules you want to run. The system-classification rules are the same as the sensitive information types in Microsoft 365, which will let you extend your sensitivity-labelling policies in the Microsoft Purview compliance portal to Microsoft Purview supported stores.

Classification

Metadata is used to help describe the data that is being scanned and made available in the catalogue. During the configuration of a scan set, you can specify classification rules to apply during the scan that will also serve as metadata. The classification rules fall under five major categories:

- **Government**: Covers attributes such as government identity cards, driver license numbers, passport numbers, and so on.
- **Financial**: Covers attributes such as bank account numbers or credit card numbers.
- **Personal**: Personal information such as a person's age, date of birth, email address, phone number, and so on.
- **Security**: Attributes like passwords that may be stored.
- **Miscellaneous**: Attributes not covered in the other categories.



After you register a data source, you can then enrich its metadata. Either the user who registered the data source or another user in the enterprise adds the metadata. Any user can annotate a data source by providing descriptions, tags, or other metadata for requesting data-source access. This descriptive metadata supplements the structural metadata, such as column names and data types, that's registered from the data source.

Discovering and understanding data sources and their use is the primary purpose of registering the sources. Enterprise users might need data for business intelligence, application development, data science, or any other task where the right data is required. They'll use the data catalogue discovery experience to quickly find data that matches their needs, understand the data to evaluate its fitness for the purpose, and consume the data by opening the data source in their tool of choice.

At the same time, users can contribute to the catalogue by tagging, documenting, and annotating data sources that have already been registered. They can also register new data sources, which are then discovered, understood, and consumed by the community of catalogue users.

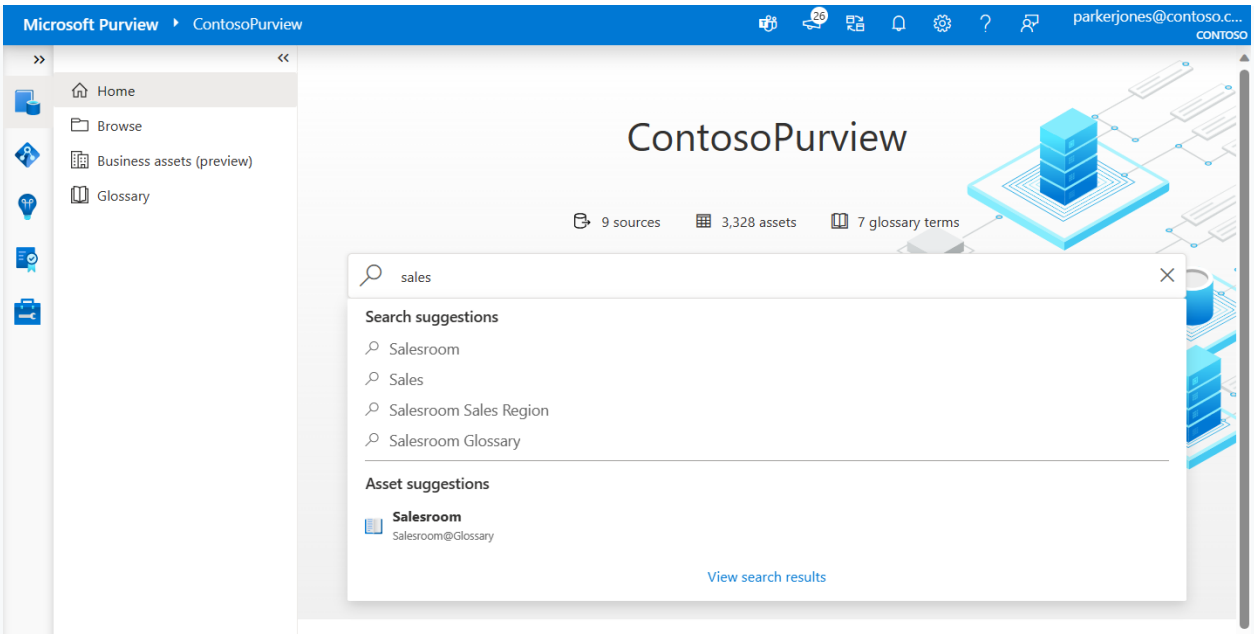
Browse and search

Microsoft Purview allows you to search information from Data Map using Purview Catalogue. You can perform text-based search and incorporate business context into the search as well.

You can use business context to search information from the Purview catalogue. You can define business glossaries (and bulk import existing ones, too) and apply business context onto assets in the Data Map. They can then use this business context to browse and search information from the Data Map.

Enable effortless discovery with:

- Semantic search and browse
- Business glossary and workflows
- Data lineage with sources, owners, transformations, and lifecycle

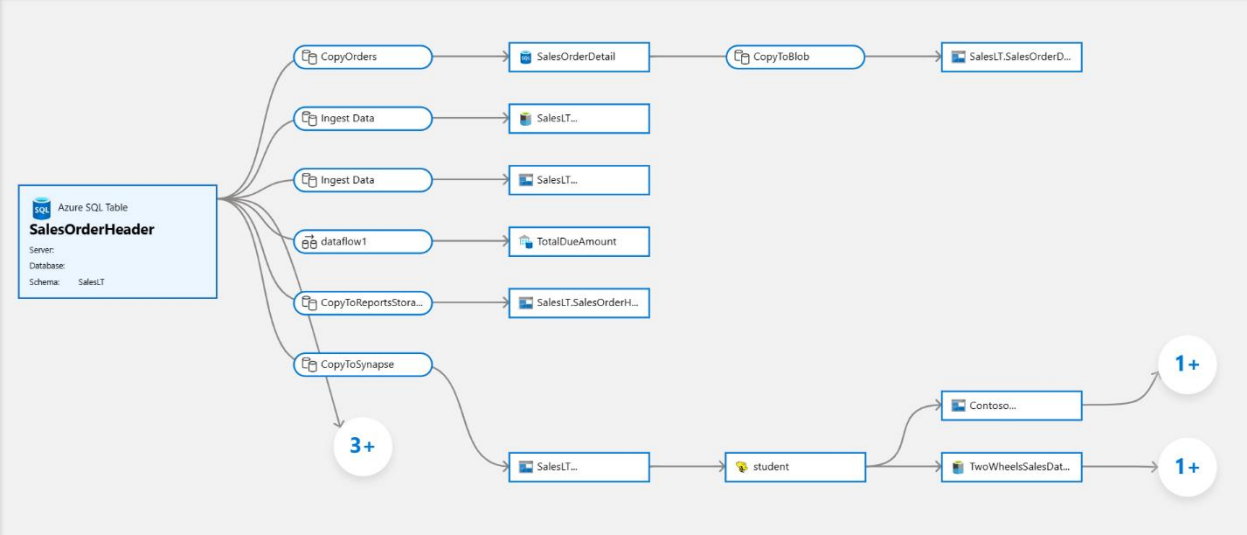


Data lineage

The concept of data lineage focuses on the lifecycle of data. The lifecycle concerns itself with the various stages data may go through. Data is sourced, moved, and stored throughout the lifecycle. Data may also undergo transformations in the extract load transform/extract transform load (ELT/ETL) operations.

Data lineage can offer insights into the data lifecycle by looking at the data pipeline. You can use the lineage to identify the root cause of issues, perform data quality analysis, and verify compliance.

Microsoft Purview represents this data lineage in a visual form, showing data movement from source to destination.



https://learn.microsoft.com/en-us/purview/purview

Question 57: **Incorrect**

Scenario: Pennyworth's Haberdashery has been around for over 50 years now and they have several stores in the Greater London area. They have just purchased a smaller clothier line based in Madrid and are integrating their systems with Pennyworth's Microsoft Power Platform service.

Alfred Pennyworth has hired you as a Microsoft Power Platform Expert for guidance on the merger of the systems.

Currently the IT team is working with Azure Synapse Link for SQL and wants to replicate data to Azure Synapse Analytics.

From which of the following data sources can they use Azure Synapse Link for SQL to replicate data to Azure Synapse Analytics?

- **SQL Server 2022**

  (Correct)

- **Azure Cosmos DB**

  (Incorrect)

- **Azure SQL Managed Instance**

- **Azure Data Factory**

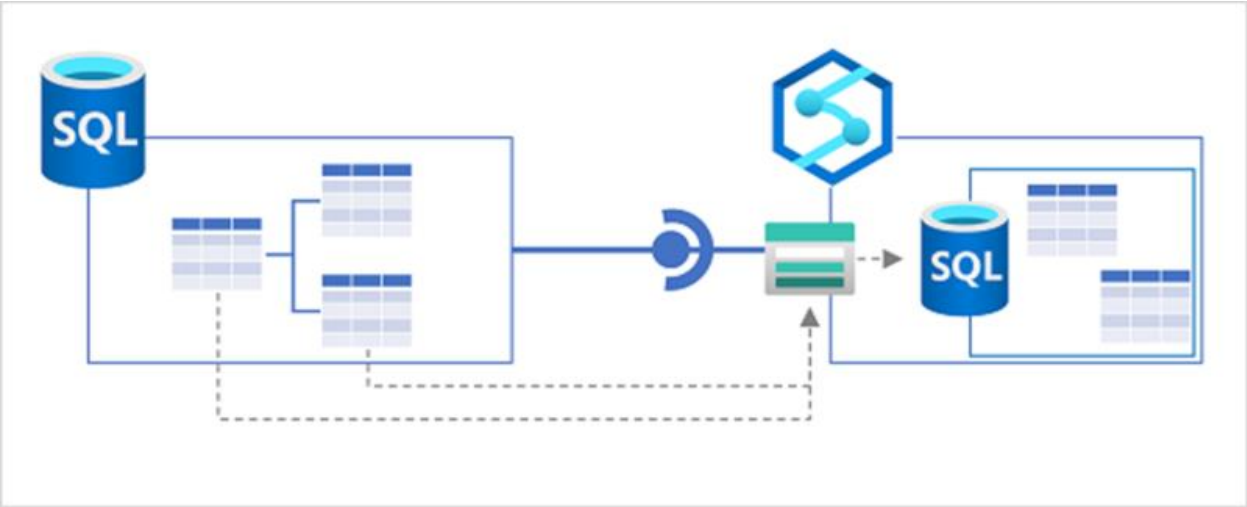
Explanation

You can use Azure Synapse Link for SQL to replicate data from SQL Server 2022.

Configure Azure Synapse Link for SQL Server 2022

Microsoft SQL Server is one of the world's most commonly used relational database systems. SQL Server 2022 is the latest release and includes many enhancements and new features; including the ability to be used as a source for Azure Synapse Link.

Azure Synapse Link for SQL Server uses a *link connection* to map one or more tables in an Azure SQL Database instance to tables in a dedicated SQL pool in Azure Synapse Analytics. When the link connection is started, the tables are initialized by copying a .parquet file for each source table to a *landing zone* in Azure Data Lake Storage Gen2; from where the data is imported into tables in the dedicated SQL pool. Subsequently, the change feed process copies all changes as .csv files to the landing zone where they're applied to the target tables.



Synchronization between SQL Server (which can be on-premises or in a private network) and Azure Synapse Analytics is achieved through a self-hosted integration runtime. An integration runtime is a software agent that handles secure connectivity when using Azure Data Factory or Azure Synapse Analytics to transfer data across networks. It must be installed on a Microsoft Windows computer with direct access to your SQL Server instance.

https://learn.microsoft.com/en-us/azure/data-factory/create-self-hosted-integration-runtime?tabs=synapse-analytics

Implementing Azure Synapse Link for SQL Server 2022

To use Azure Synapse Link for SQL Server 2022, you need to create storage for the landing zone in Azure and configure your SQL Server instance before creating a link connection in Azure Synapse Analytics.

Create landing zone storage

You need to create an Azure Data Lake Storage Gen2 account in your Azure subscription to use as a landing zone. You can't use the default storage account for your Azure Synapse Analytics workspace.

https://learn.microsoft.com/en-us/azure/storage/blobs/create-data-lake-storage-account?azure-portal-true

Create a master key in the SQL Server database

To support Azure Synapse Link, your SQL Server database must contain a master key. You can use a `CREATE MASTER KEY` SQL statement like the following example to create one:

```
SQL
CREATE MASTER KEY ENCRYPTION BY PASSWORD = 'my$ecretPa$$w0rd’;
```

Create a dedicated SQL pool in Azure Synapse Analytics

In your Azure Synapse Analytics workspace, you need to create a dedicated SQL pool where the target tables will be created. You also need to create master key in this database by using the following SQL statement:

```
SQL
CREATE MASTER KEY
```

Create a linked service for the SQL Server source database

Next, in Azure Synapse Analytics, create a linked service for your SQL Server database. When you do this, you need to specify the self-hosted integration runtime to be used for connectivity between SQL Server and Azure Synapse Analytics. If you haven't already configured a self-hosted integration runtime, you can create one now, and then download and install the agent onto a Windows machine in the network where your SQL Server instance is located.

Create a linked service for your Data Lake Storage Gen2 account

In addition to the linked service for SQL Server, you need a linked service for the Data Lake Storage Gen2 account that will be used as a landing zone. To support this, you need to add the managed identity of your Azure Synapse Analytics Workspace to

the **Storage Blob Data Contributor** role for your storage account and configure the linked service to use the managed identity for authentication.

Create a link connection for Azure Synapse Link

Finally, you're ready to create a link connection for Azure Synapse Link data synchronization. As you do so, you'll specify the service link for the SQL Server source database, the individual tables to be replicated, the number of CPU cores to be used for the synchronization process, and the Azure Data Lake Storage Gen2 linked service and folder location for the landing zone.

After the link connection is created, you can start it to initialize synchronization. After a short time, the tables will be available to query in the dedicated SQL pool and will be kept in sync with modifications in the source database by the change feed process.

https://learn.microsoft.com/en-us/azure/synapse-analytics/synapse-link/sql-server-2022-synapse-link

https://learn.microsoft.com/en-us/azure/synapse-analytics/synapse-link/synapse-link-for-sql-known-issues

https://learn.microsoft.com/en-us/azure/synapse-analytics/synapse-link/connect-synapse-link-sql-server-2022

Question 58: **Incorrect**
Scenario: The Eat-More Corporation is a U.S.-based fast-food restaurant chain headed by Teresa Payton. The main headquarters and processing plants of the Eat-More Corporation are located in Sedona, Arizona. Eat-More has restaurants located all over the United States and has plans to expand internationally. The expansion plans are underway and have presented several IT challenges which Teresa has contracted you to advise her IT staff on.

In the current project the team needs to process a stream of sensor data, aggregating values over one-minute windows and storing the results in a data lake.

Which service should you advise Teresa and her team to use?

- **Azure Stream Analytics**

(Correct)

- **Azure SQL Database**

  (Incorrect)

- **Azure REST API**

- **Azure Cosmos DB**

Explanation

- *Azure Stream Analytics is a stream processing engine.*
- Azure SQL Database is a relational data store - not a stream processing engine.
- Azure Representational State Transfer (REST) APIs are service endpoints that support sets of HTTP operations (methods), which provide create, retrieve, update, or delete access to the service's resources - also not a stream processing engine.
- Azure Cosmos DB is a data storage solution for multiple types of data, but doesn't provide stream processing functionality.

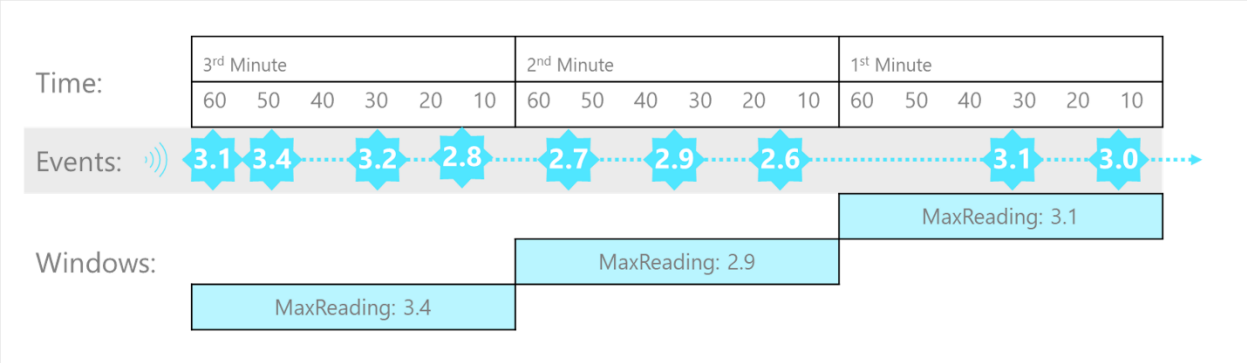
Understand window functions

A common goal of stream processing is to aggregate events into temporal intervals, or *windows*. For example, to count the number of social media posts per minute or to calculate the average rainfall per hour.

Azure Stream Analytics includes native support for five kinds of temporal windowing functions. These functions enable you to define temporal intervals into which data is aggregated in a query. The supported windowing functions are Tumbling, Hopping, Sliding, Session, and Snapshot.

Tumbling

Tumbling window functions segment a data stream into a contiguous series of fixed-size, non-overlapping time segments and operate against them. Events can't belong to more than one tumbling window.

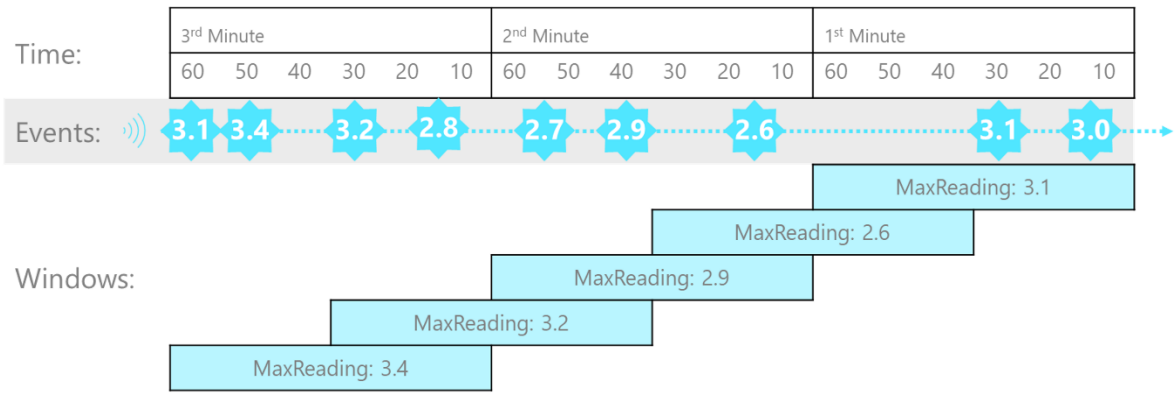


The Tumbling window example, represented by the following query, finds the maximum reading value in each one-minute window. Windowing functions are applied in Stream Analytics jobs using the GROUP BY clause of the query syntax. The `GROUP BY` clause in the following query contains the `TumblingWindow()` function, which specifies a one-minute window size.

```
SQL
SELECT DateAdd(minute,-1,System.TimeStamp) AS WindowStart,
       System.TimeStamp() AS WindowEnd,
       MAX(Reading) AS MaxReading
INTO
    [output]
FROM
    [input] TIMESTAMP BY EventProcessedUtcTime
GROUP BY TumblingWindow(minute, 1)
```

Hopping

Hopping window functions model scheduled overlapping windows, jumping forward in time by a fixed period. It's easiest to think of them as Tumbling windows that can overlap and be emitted more frequently than the window size. In fact, tumbling windows are simply a hopping window whose `hop` is equal to its `size`. When you use Hopping windows, events can belong to more than one window result set.



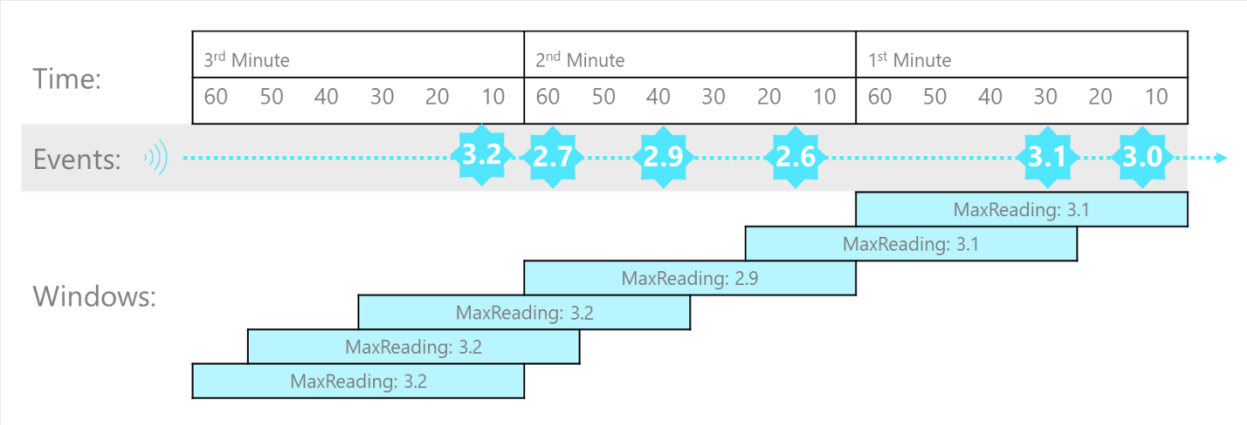
To create a hopping window, you must specify three parameters. The first parameter indicates the time unit, such as second, minute, or hour. The following parameter sets the window size, which designates how long each window lasts. The final required parameter is the hop size, which specifies how much each window moves forward relative to the previous one. An optional fourth parameter denoting the offset size may also be used.

The following query demonstrates using a `HoppingWindow()` where the `timeunit` is set to `second`. The `windowsize` is 60 seconds, and the `hopsize` is 30 seconds. This query outputs an event every 30 seconds containing the maximum reading value that occurred over the last 60 seconds.

```
SQL
SELECT DateAdd(second,-60,System.TimeStamp) AS WindowStart,
       System.TimeStamp() AS WindowEnd,
       MAX(Reading) AS MaxReading
INTO
    [output]
FROM
    [input] TIMESTAMP BY EventProcessedUtcTime
GROUP BY HoppingWindow(second, 60, 30)
```

Sliding

Sliding windows generate events for points in time when the content of the window actually changes. This function model limits the number of windows that need to be considered. Azure Stream Analytics outputs events for only those points in time when an event entered or exited the window. As such, every window contains a minimum of one event. Events in Sliding windows can belong to more than one sliding window, similar to Hopping windows.

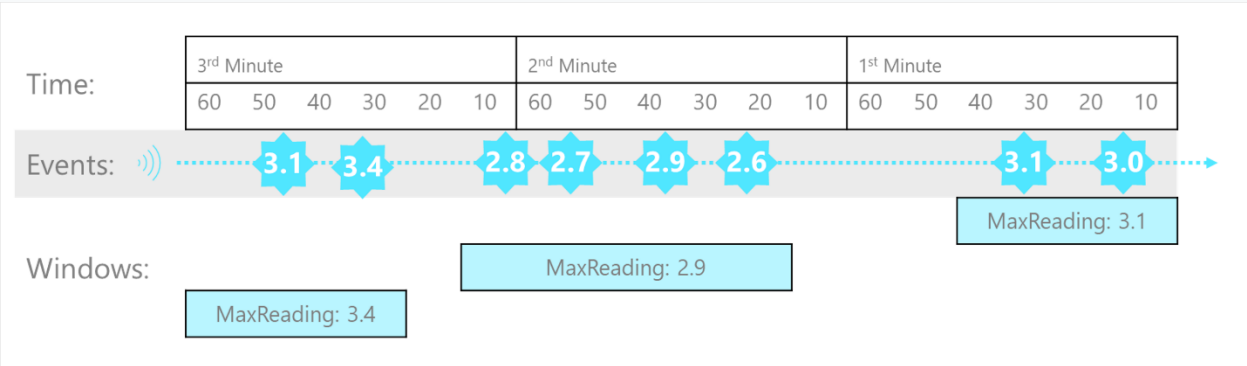


The following query uses the `SlidingWindow()` function to find the maximum reading value in each one-minute window in which an event occurred.

```
SQL
SELECT DateAdd(minute,-1,System.TimeStamp) AS WindowStart,
       System.TimeStamp() AS WindowEnd,
       MAX(Reading) AS MaxReading
INTO
    [output]
FROM
    [input] TIMESTAMP BY EventProcessedUtcTime
GROUP BY SlidingWindow(minute, 1)
```

Session

Session window functions cluster together events that arrive at similar times, filtering out periods of time where there's no data. It has three primary parameters: timeout, maximum duration, and partitioning key (optional).



The occurrence of the first event starts a session window. Suppose another event occurs within the specified timeout from the last ingested event. In that case, the

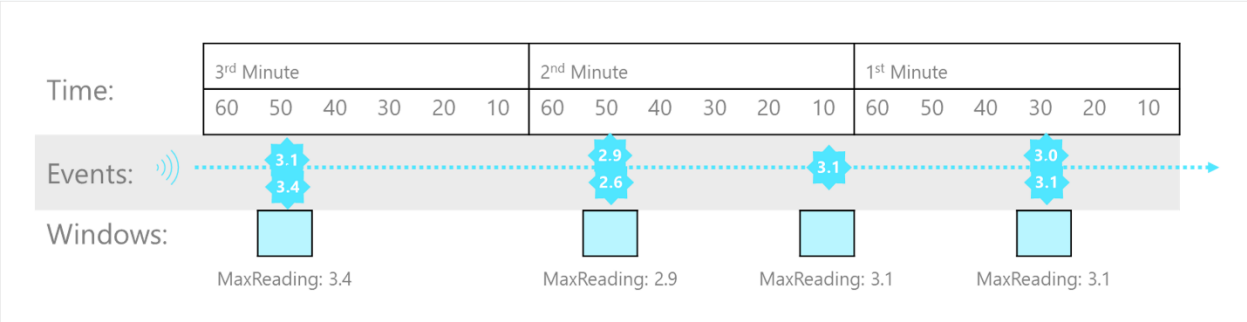
window will be extended to incorporate the new event. However, if no other events occur within the specified timeout period, the window will be closed at the timeout. If events keep happening within the specified timeout, the session window will extend until the maximum duration is reached.

The following query measures user session length by creating a `SessionWindow` over clickstream data with a `timeoutsize` of 20 seconds and a `maximumdurationsize` of 60 seconds.

```
SQL
SELECT DateAdd(second,-60,System.TimeStamp) AS WindowStart,
       System.TimeStamp() AS WindowEnd,
       MAX(Reading) AS MaxReading
INTO
    [output]
FROM
    [input] TIMESTAMP BY EventProcessedUtcTime
GROUP BY SessionWindow(second, 20, 60)
```

Snapshot

Snapshot windows groups events by identical timestamp values. Unlike other windowing types, a specific window function isn't required. You can employ a snapshot window by specifying the `System.Timestamp()` function to your query's `GROUP BY` clause.



For example, the following query finds the maximum reading value for events that occur at precisely the same time.

```
SQL
SELECT System.TimeStamp() AS WindowTime,
       MAX(Reading) AS MaxReading
INTO
```

```
    [output]
FROM
    [input] TIMESTAMP BY EventProcessedUtcTime
GROUP BY System.Timestamp()
```

`System.Timestamp()` is considered in the `GROUP BY` clause as a snapshot window definition because it groups events into a window based on the equality of timestamps.

https://learn.microsoft.com/en-us/azure/stream-analytics/stream-analytics-introduction

Question 59: **Correct**

Which transformation in the Mapping Data Flow is used to routes data rows to different streams based on matching conditions?

- **Lookup**
- **Derived column**
- **Alter row**
- **Multiple inputs/outputs**
- **Select**
- **Conditional Split**

  (Correct)

Explanation
A Conditional Split transformation routes data rows to different streams based on matching conditions. The conditional split transformation is similar to a CASE decision structure in a programming language.

https://docs.microsoft.com/en-us/azure/data-factory/data-flow-conditional-split

Transforming data using Mapping Data Flow

Mapping Data Flows provide an environment for building a wide range of data transformations visually without the need to use code. The resulting data flows that are created are subsequently executed on scaled-out Apache Spark clusters that are automatically provisioned when you execute the Mapping Data Flow. Mapping Data Flows also provides the capability to monitor the execution of the transformations so that you can view how the transformations are progressing, or to understand any errors that may occur

Mapping Data Flows provides a number of different transformations types that enable you to modify data. They are broken down into the following categories:

• Schema modifier transformations

• Row modifier transformations

• Multiple inputs/outputs transformations

Below is a list of transformations that is available in the Mapping Data Flows:

Name & Category: Aggregate - Schema modifier

Description: Define different types of aggregations such as SUM, MIN, MAX, and COUNT grouped by existing or computed columns.

Name & Category: Alter row - Row modifier

Description: Set insert, delete, update, and upsert policies on rows. You can add one-to-many conditions as expressions. These conditions should be specified in order of priority, as each row will be marked with the policy corresponding to the first-matching expression. Each of those conditions can result in a row (or rows) being inserted, updated, deleted, or upserted. Alter Row can produce both DDL & DML actions against your database.

Name & Category: Conditional split - Multiple inputs/outputs

Description: Route rows of data to different streams based on matching conditions.

Name & Category: Derived column - Schema modifier

Description: Generate new columns or modify existing fields using the data flow expression language.

Name & Category: Exists - Multiple inputs/outputs

Description: Check whether your data exists in another source or stream.

Name & Category: Filter - Row modifier

Description: Filter a row based upon a condition.

Name & Category: Flatten - Schema modifier

Description: Take array values inside hierarchical structures such as JSON and unroll them into individual rows.

Name & Category: Join - Multiple inputs/outputs

Description: Combine data from two sources or streams.

Name & Category: Lookup - Multiple inputs/outputs

Description: Enables you to reference data from another source.

Name & Category: New branch - Multiple inputs/outputs

Description: Apply multiple sets of operations and transformations against the same data stream.

Name & Category: Pivot - Schema modifier

Description: An aggregation where one or more grouping columns has distinct row values transformed into individual columns.

Name & Category: Select - Schema modifier

Description: Alias columns and stream names, and drop or reorder columns.

Name & Category: Sink – N/A

Description: A final destination for your data.

Name & Category: Sort - Row modifier

Description: Sort incoming rows on the current data stream.

Name & Category: Source – N/A

Description: A data source for the data flow.

Name & Category: Surrogate key - Schema modifier

Description: Add an incrementing non-business arbitrary key value.

Name & Category: Union - Multiple inputs/outputs

Description: Combine multiple data streams vertically.

Name & Category: Unpivot - Schema modifier

Description: Pivot columns into row values.

Name & Category: Window - Schema modifier

Description: Define window-based aggregations of columns in your data streams.

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

Question 60: **Incorrect**
How many drivers does a Cluster have?

- **Configurable between one and ten**
- **Only one**

  (Correct)
- **Two, running in parallel**
- **Configurable between one and eight**

  (Incorrect)

Explanation
A Cluster has one and only one driver.

Cluster node type

A cluster consists of one driver node and worker nodes.

You can pick separate cloud provider instance types for the driver and worker nodes, although by default the driver node uses the same instance type as the worker node. Different families of instance types fit different use cases, such as memory-intensive or compute-intensive workloads.

https://docs.microsoft.com/en-us/azure/databricks/clusters/configure

Question 61: **Incorrect**
What is the name of the application architecture that enables near real-time querying to provide insights?

- **ELT**
- **ADPS**
- **OLTP**
- **HTAP**

  (Correct)
- **ETL**

  (Incorrect)
- **OLAP**

Explanation
HTAP stands for Hybrid Transactional and Analytical Processing that enable you to gain insights from operational systems without impacting the performance of the operational system.

https://www.zdnet.com/article/what-is-hybrid-transactionanalytical-processing-htap/

Question 62: **Incorrect**
Identify the missing word(s) in the following sentence within the context of Microsoft Azure.

Transactional databases are often called [?] systems. These systems commonly support lots of users, have quick response times, and handle large volumes of data.

- **Automated Data Processing Structured (ADPS)**
- **Atomicity, Consistency, Isolation, and Durability (ACID)**
- **OLAP (Online Analytical Processing)**
- **Extract, load, and transform (ELT)**

  (Incorrect)
- **OLTP (Online Transaction Processing)**

  (Correct)
- **Extract, transform, and load (ETL)**

Explanation
A transaction is a logical group of database operations that execute together.

Here's the question to ask yourself regarding whether you need to use transactions in your application: Will a change to one piece of data in your dataset impact another? If the answer is yes, then you'll need support for transactions in your database service.

Transactions are often defined by a set of four requirements, referred to as ACID guarantees. ACID stands for **A**tomicity, **C**onsistency, **I**solation, and **D**urability:

• **Atomicity** means a transaction must execute exactly once and must be atomic; either all of the work is done, or none of it is. Operations within a transaction usually share a common intent and are interdependent.

• **Consistency** ensures that the data is consistent both before and after the transaction.

• **Isolation** ensures that one transaction is not impacted by another transaction.

• **Durability** means that the changes made due to the transaction are permanently saved in the system. Committed data is saved by the system so that even in the event of a failure and system restart, the data is available in its correct state.

When a database offers ACID guarantees, these principles are applied to any transactions in a consistent manner.

OLTP vs OLAP

Transactional databases are often called OLTP (Online Transaction Processing) systems. OLTP systems commonly support lots of users, have quick response times, and handle large volumes of data. They are also highly available (meaning they have very minimal downtime), and typically handle small or relatively simple transactions.

On the contrary, OLAP (Online Analytical Processing) systems commonly support fewer users, have longer response times, can be less available, and typically handle large and complex transactions.

The terms OLTP and OLAP aren't used as frequently as they used to be, but understanding them makes it easier to categorize the needs of your application.

Now that you're familiar with transactions, OLTP, and OLAP, let's walk through each of the data sets in the online retail scenario, and determine the need for transactions.

https://www.guru99.com/oltp-vs-olap.html

Question 63: **Correct**
Before we can query our data using Azure Synapse Analytics using Azure Synapse Link, we must first create the container that is going to hold our data at the same time enabling it to have an analytical store.

True or False: Enabling analytical store is only available at the time of creating a container however it can be deactivated or reactivated at anytime thereafter.

- **True**

- **False**

  (Correct)

Explanation
Before we can query our data using Azure Synapse Analytics using Azure Synapse Link, we must first create the container that is going to hold our data at the same time enabling it to have an analytical store.

Enabling analytical store is only available at the time of creating a container and cannot be completely disabled without deleting the container. Setting the default analytical store TTL value to 0 or null effectively disables the analytical store by no longer synchronize new items to it from the transactional store and deleting items already synchronized from the analytical store.

https://docs.microsoft.com/en-us/azure/cosmos-db/configure-synapse-link

Question 64: **Incorrect**
Scenario: You are working in a department which requires preparation of data for ad hoc data exploration and analysis based on market fluctuations. The Department Head has tasked you with determining the most effective resource model in Azure Synapse Analytics to employ.

Which of the following should you choose?

- **Dedicated**

- **Serverless**

  (Correct)

- **Databricks**

  (Incorrect)

-

Pipelines

-

IoT Central

Explanation

Serverless SQL pool is a pay per query service that doesn't require you to pick the right size. The system automatically adjusts based on your requirements, freeing you up from managing your infrastructure and picking the right size for your solution.

The serverless resource model is the ideal resource model in this scenario as it makes use of the resources when required.

https://docs.microsoft.com/en-us/azure/synapse-analytics/sql/resource-consumption-models

Question 65: **Incorrect**

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

Azure Monitor provides base-level infrastructure metrics and logs for most Azure services. Azure diagnostic logs are emitted by a resource and provide rich, frequent data about the operation of that resource. Azure Data Factory (ADF) can write diagnostic logs in Azure Monitor.

Data Factory stores pipeline-run data for [?] days.

-

10

-

30

-

21

(Incorrect)

-

45

(Correct)

-

15

Explanation

Monitor using Azure Monitor

Azure Monitor provides base-level infrastructure metrics and logs for most Azure services. Azure diagnostic logs are emitted by a resource and provide rich, frequent data about the operation of that resource. Azure Data Factory (ADF) can write diagnostic logs in Azure Monitor.

Data Factory stores pipeline-run data for only 45 days. Use Azure Monitor if you want to keep that data for a longer time. With Monitor, you can route diagnostic logs for analysis to multiple different targets.

• **Storage Account**: Save your diagnostic logs to a storage account for auditing or manual inspection. You can use the diagnostic settings to specify the retention time in days.

• **Event Hub**: Stream the logs to Azure Event Hubs. The logs become input to a partner service/custom analytics solution like Power BI.

• **Log Analytics**: Analyze the logs with Log Analytics. The Data Factory integration with Azure Monitor is useful in the following scenarios:

• You want to write complex queries on a rich set of metrics that are published by Data Factory to Monitor. You can create custom alerts on these queries via Monitor.

• You want to monitor across data factories. You can route data from multiple data factories to a single Monitor workspace.

You can also use a storage account or event-hub namespace that isn't in the subscription of the resource that emits logs. The user who configures the setting must have appropriate Azure role-based access control (Azure RBAC) access to both subscriptions.

https://docs.microsoft.com/en-us/azure/data-factory/monitor-using-azure-monitor

Question 66: **Incorrect**

Scenario: Queen Consolidated was overtaken by Raymond Carson Palmer and rebranded as Palmer Technologies. Now that Ray is overseeing the operations at Palmer, Ray has decided to move away from on-prem datacentres to Azure. Ray and the IT team are developing a new data engineering solutions for a company.

The current project is dealing with social media and has the following requirements.

Required:

• Real-time Twitter feed analysis of posts which contain specific keywords and must be stored as well as processed on MS Azure then displayed using MS Power BI.

Ray and the IT team have put together a list of actions they think need to be performed to meet the needs of the project, but they are not sure on the order to execute. Below is a list of the actions they are considering.

Proposed Actions:

a. Create an HDInisght cluster with the Spark cluster type.

b. Create a Jupyter Notebook.

c. Run a job that uses the Spark Streaming API to ingest data from Twitter.

d. Create a Runbook.

e. Create an HDInsight cluster with the Spark cluster type.

f. Create a HVAC table.

g. Load the HVAC table into Power BI Desktop

As you are the Azure SME, Ray and the team look to you for direction on selecting the required items and putting them in the proper order. Which of the below contains the correct items in the correct sequence to meet the requirements?

- **e → a → c → g → d**

  (Incorrect)

- **f → b → d → a → g → c**

- **b → a → e → f → c → g**

-

  a → b → f → c → g

  (Correct)

Explanation

Step 1: Create an HDInisght cluster with the Spark cluster type.

Step 2: Create a Jyputer Notebook.

Step 3: Create HVAC table.

The Jupyter Notebook that you created in the previous step includes code to create an HVAC table.

Step 4: Run a job that uses the Spark Streaming API to ingest data from Twitter.

Step 5: Load the HVAC table into Power BI Desktop.

You use Power BI to create visualizations, reports, and dashboards from the Spark cluster data.

https://www.youtube.com/watch?v=_RJ0VjZ2-og

https://docs.microsoft.com/en-us/azure/hdinsight/spark/apache-spark-use-with-data-lake-store

Question 67: **Correct**
Identify the missing word(s) in the following sentence within the context of Microsoft Azure.

[?] is typically used to automate the process of extracting, transforming, and loading the data through a batch process against structured and unstructured data sources.

- **Azure Conductor**
- **Azure Functions**
- **Azure Designer**
- **Azure Orchestrator**
- **Azure PowerShell**
- **Azure Data Factory**

  (Correct)
- **Azure Stored Procedure**

Explanation
Modern Data Warehouse workloads:

A Modern Data Warehouse is a centralized data store that provides descriptive analytics and decision support services across the whole enterprise using structured, unstructured, or streaming data sources. Data flows into the warehouse from multiple transactional systems, relational databases, and other data sources on a periodic basis. The stored data is used for historical and trend analysis reporting. The data warehouse acts as a central repository for many subject areas and contains the "single source of truth."

Azure Data factory is typically used to automate the process of extracting, transforming, and loading the data through a batch process against structured and unstructured data sources.

Advanced Analytical Workloads

You can perform advanced analytics in the form of predictive or preemptive analytics using a range of Azure data platform services. Azure Data Factory provides the integration from source systems into a Data Lake store, and can initiate compute resources such as Azure Databricks, or HDInsight to use the data to perform the advanced analytical work

https://cloudblogs.microsoft.com/industry-blog/en-gb/technetuk/2020/08/25/data-orchestration-with-azure-data-factory/

Question 68: **Correct**
Azure Databricks is an amalgamation of multiple technologies that enable you to work with data at scale.

Which of the following is best described by *"This builds on the relational table schema abstraction over files in the data lake to add support for SQL semantics commonly found in relational database systems. Capabilities provided by this include transaction logging, data type constraints, and the ability to incorporate streaming data into a relational table"*?

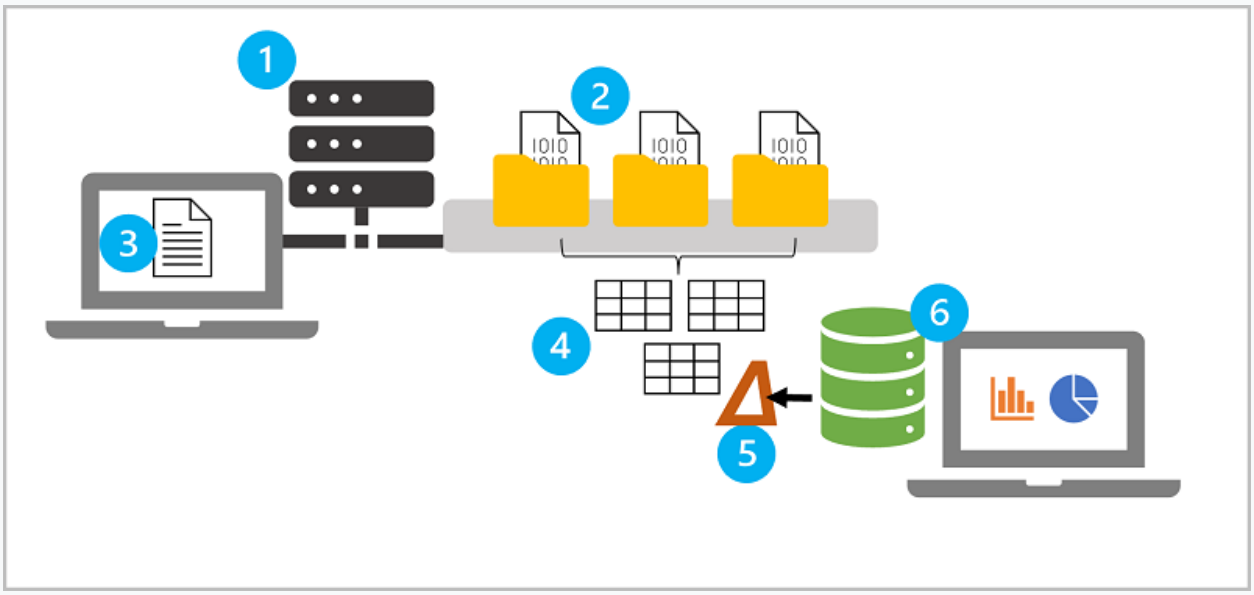
- **Hive metastore**
- **SQL Warehouses**
- **Apache Spark clusters**

- Databricks File System
- Notebook
- Delta Lake

  (Correct)

Explanation
Understand key concepts

Azure Databricks is an amalgamation of multiple technologies that enable you to work with data at scale. Before using Azure Databricks, there are some key concepts that you should understand.



1. **Apache Spark clusters** - Spark is a distributed data processing solution that makes use of *clusters* to scale processing across multiple compute *nodes*. Each Spark cluster has a *driver* node to coordinate processing jobs, and one or more *worker* nodes on

which the processing occurs. This distributed model enables each node to operate on a subset of the job in parallel; reducing the overall time for the job to complete.

https://learn.microsoft.com/en-us/azure/databricks/clusters/

2. Databricks File System (DBFS) - While each cluster node has its own local file system (on which operating system and other node-specific files are stored), the nodes in a cluster have access to a shared, distributed file system in which they can access and operate on data files. The *Databricks File System* (DBFS) enables you to mount cloud storage and use it to work with and persist file-based data.

https://learn.microsoft.com/en-us/azure/databricks/data/databricks-file-system

3. Notebooks - One of the most common ways for data analysts, data scientists, data engineers, and developers to work with Spark is to write code in *notebooks*. Notebooks provide an interactive environment in which you can combine text and graphics in *Markdown* format with cells containing code that you run interactively in the notebook session.

https://learn.microsoft.com/en-us/azure/databricks/notebooks/

4. Hive metastore - *Hive* is an open-source technology used to define a relational abstraction layer of tables over file-based data. The tables can then be queried using SQL syntax. The table definitions and details of the file system locations on which they're based is stored in the metastore for a Spark cluster. A *Hive metastore* is created for each cluster when it's created, but you can configure a cluster to use an existing external metastore if necessary.

https://learn.microsoft.com/en-us/azure/databricks/data/metastores/

5. Delta Lake - *Delta Lake* builds on the relational table schema abstraction over files in the data lake to add support for SQL semantics commonly found in relational database systems. Capabilities provided by Delta Lake include transaction logging, data type constraints, and the ability to incorporate streaming data into a relational table.

https://learn.microsoft.com/en-us/azure/databricks/delta/

6. SQL Warehouses - *SQL Warehouses* are relational compute resources with endpoints that enable client applications to connect to an Azure Databricks workspace and use SQL to work with data in tables. The results of SQL queries can be used to create data visualizations and dashboards to support business analytics and decision making. SQL Warehouses are only available in *premium* tier Azure Databricks workspaces.

https://learn.microsoft.com/en-us/azure/databricks/sql/admin/sql-endpoints

Question 69: **Incorrect**

Scenario: The Tivan Group is a prestigious and notoriously powerful group in the galactic community. Responsible for the founding of Knowhere, the Group is considerably wealthy and are often the source of exponential contributions to the galaxy. Taneleer Tivan, the infamous Collector, controls the group and is implementing Microsoft Azure into the company. The lead developer is Peter Quill.

Currently, Peter is working on a project and has an Azure Cosmos DB for NoSQL account and an Azure Synapse Analytics workspace.

Required: Enable HTAP integration with Azure Synapse Analytics

Which of the following must Peter do to start off?

- **Create a dedicated SQL pool in Azure Synapse Analytics**
- **Select the region to deploy the Azure Synapse Analytics**
- **Enable Azure Synapse Link in Azure Cosmos DB**

  (Correct)
- **Configure global replication in Azure Cosmos DB**

(Incorrect)

Explanation

- *The first step in setting up HTAP integration is to enable Azure Synapse Link in Azure Cosmos DB.*
- Global replication helps scale Cosmos DB, but it's not required for HTAP integration with Azure Synapse Analytics.
- A dedicated SQL pool is used for data warehousing in Azure Synapse Analytics. It's not required for HTAP integration with Azure Cosmos DB.

Run a job to ingest data

When you've created and saved your query, you can run the Azure Stream Analytics job to process events in the input(s) and write the results to output(s). Once started, the query will run perpetually until stopped; constantly ingesting new event data into your Azure Synapse Analytics workspace (into a table in relational data warehouse or files in a data lake, depending on the output type).

Working with ingested data

You can work with the ingested streaming data like any other data in Azure Synapse Analytics, combining it with data ingested using batch processing techniques or synchronized from operational data sources by using Azure Synapse Link.

Querying data in a relational data warehouse

If you used an Azure Synapse Analytics output to ingest the results of your stream processing job into a table in a dedicated SQL pool, you can query the table using a SQL query, just like any other table. The results of the query will always include the latest data to be ingested at the time the query is run. Your data warehouse can include tables for streaming data as well as tables for batch ingested data, enabling you to join real-time and batch data for historical analytics.

For example, the following SQL code could be used to query a table named **factSensorReadings** that contains the results of stream processing, and combine it with a **dimDate** table containing detailed data about the dates on which readings were captured.

```
SQL
SELECT d.Weekday, s.SensorID, AVG(s.SensorReading) AS AverageReading
FROM factSensorReadings AS s
JOIN dimDate AS d
    ON CAST(s.ReadingTime AS DATE) = d.DateKey
GROUP BY d.Weekday, s.SensorID
```

https://learn.microsoft.com/en-us/training/modules/design-multidimensional-schema-to-optimize-analytical-workloads/

Querying data in a data lake

As streaming data is ingested into files in a data lake, you can query those files by using a serverless SQL pool in Azure Synapse Analytics. For example, the following query reads all fields from all Parquet files under the **sensors** folder in the **data** file system container.

```
SQL
SELECT *
FROM OPENROWSET(
    BULK 'https://mydatalake.blob.core.windows.net/data/sensors/*',
    FORMAT = 'parquet') AS rows
```

https://learn.microsoft.com/en-us/training/modules/query-data-lake-using-azure-synapse-serverless-sql-pools

You can also query the data lake by using code running in an Apache Spark pool, as shown in this example:

```
Python
%%pyspark
df = spark.read.load('abfss://data@datalake.dfs.core.windows.net/sensors/*',
format='parquet'
)
display(df)
```

https://learn.microsoft.com/en-us/training/modules/understand-big-data-engineering-with-apache-spark-azure-synapse-analytics/

Question 70: **Correct**
True or False: Access keys are the easiest approach to authenticating access to a storage account which provide full access to anything in the storage account, similar to a root password on a computer.

- False
- True

  (Correct)

Explanation
Shared Access Signatures (SAS)

Access keys are the easiest approach to authenticating access to a storage account. However they provide full access to anything in the storage account, similar to a root password on a computer.

Storage accounts offer a separate authentication mechanism called *shared access signatures* that support expiration and limited permissions for scenarios where you need to grant limited access. You should use this approach when you are allowing other users to read and write data to your storage account. There are links to our documentation on this advanced topic at the end of the module.

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

Question 71: **Incorrect**
Scenario: Data loads at your company have increased the processing time for on-premises data warehousing descriptive analytic solutions. You have been tasked with looking into a cloud-based alternative to reduce processing time and release business intelligence reports faster. Your boss wants you to first consider scaling up on-premises servers but you discover this approach would reach its physical limits shortly.

The new solution must be on a petabyte scale that doesn't involve complex installations and configurations.

Which of the following would best suit the need?

- **Azure DataNow**
- **Azure On-prem Solution**

  (Incorrect)
- **Azure Stream Analytics**
- **Azure Cosmos DB**
- **Azure Table Storage**
- **Azure Synapse Analytics**

  (Correct)

Explanation
Azure Synapse Analytics is a cloud-based data platform that brings together enterprise data warehousing and Big Data analytics. It can process massive amounts of data and answer complex business questions with limitless scale.

When to use Azure Synapse Analytics

The SQL Pools capability of Azure Synapse Analytics can meet the scenario needs.

The volume and variety of data that is being generated are providing opportunities to perform different types of analysis on the data. This can include techniques such as exploratory data analysis to identify initial patterns or meaning in the data. It can also include conducting predictive analytics for forecasting, or segmenting data. The Big Data Analytics capability of Azure Synapse Analytics will accommodate this.

Key features

SQL Pools uses massively parallel processing (MPP) to quickly run queries across petabytes of data. Because the storage is separated from the compute nodes, you can scale the compute nodes independently to meet any demand at any time.

In Azure Synapse Analytics, the Data Movement Service (DMS) coordinates and transports data between compute nodes as necessary. But you can use a replicated table to reduce data movement and improve performance. Azure Synapse Analytics supports three types of distributed tables: hash, round-robin and replicated. Use these tables to tune performance.

Importantly, Azure Synapse Analytics can also pause and resume the compute layer. This means you pay only for the computation you use. This capability is useful in data warehousing.

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

Question 72: **Incorrect**
Azure Synapse Link for SQL is a *hybrid transactional / analytical processing* (HTAP) capability in Azure Synapse Analytics that you can use to synchronize transactional data in Azure SQL Database or Microsoft SQL Server with a dedicated SQL pool in Azure Synapse Analytics. This synchronization enables you to perform near real-time analytical workloads on operational data with minimal impact on the transactional store used by business applications.

What is required if you plan to use a Spark pool in Azure Synapse Analytics to query an existing analytical store in Azure Cosmos DB?

- ○

  Create a linked service for the Azure Cosmos DB database where the analytical store enabled container is defined.

  (Correct)

- ○

  Disable automatic pausing for the Spark pool in Azure Synapse Analytics

- ◉

  Install the Azure Cosmos DB SDK for Python package in the Spark pool

  (Incorrect)

-

  None of the provided options are correct

Explanation

- *A linked service that connects to the Azure Cosmos DB account containing the analytical store enabled container is required.*
- You do not need to disable automatic pausing for a Spark pool to be used with Synapse Link for Azure Cosmos DB.
- The Azure Cosmos DB SDK for Python is not required for Synapse Link integration.

What is Azure Synapse Link for SQL?

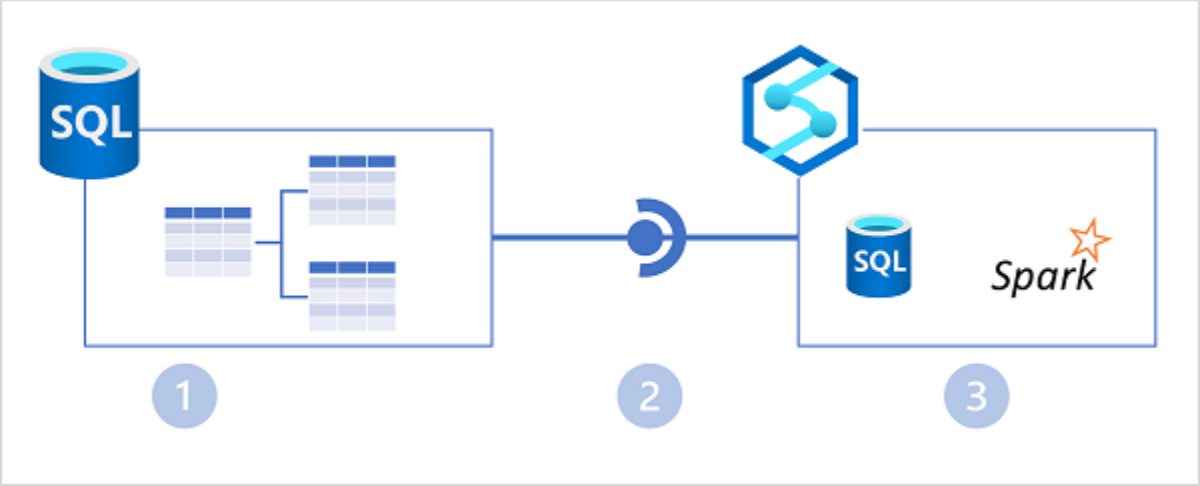
Many organizations use a relational database in Azure SQL Database or Microsoft SQL Server to support business applications. These databases are optimized for transactional workloads that store and manipulate operational data. Performing analytical queries on the data in these databases to support reporting and data analysis incurs resource contention that can be detrimental to application performance.

A traditional approach to resolving this problem is to implement an *extract, transform, and load* (ETL) solution that loads data from the operational data store into an analytical store as a batch operation at regular intervals. While this solution supports the analytical workloads required for reporting and data analysis, it suffers from the following limitations:

- The ETL process can be complex to implement and operate.
- The analytical store is only updated at periodic intervals, so reporting doesn't reflect the most up-to-date operational data.

Azure Synapse Link for SQL

Azure Synapse Link for SQL addresses the limitations of a traditional ETL process by automatically replicating changes made to tables in the operational database to corresponding tables in an analytical database. After the initial synchronization process, the changes are replicated in near real-time without the need for a complex ETL batch process.



In the diagram above, the following key features of the Azure Synapse Link for SQL architecture are illustrated:

1. An Azure SQL Database or SQL Server 2022 instance contains a relational database in which transactional data is stored in tables.
2. Azure Synapse Link for SQL replicates the table data to a dedicated SQL pool in an Azure Synapse workspace.
3. The replicated data in the dedicated SQL pool can be queried in the dedicated SQL pool, or connected to as an external source from a Spark pool without impacting the source database.

Source and target databases

Azure Synapse Link for SQL supports the following source databases (used as operational data stores):

- Azure SQL Database
- Microsoft SQL Server 2022

Note: Azure Synapse link for SQL is not supported for Azure SQL Managed Instance.

The target database (used as an analytical data store) must be a dedicated SQL pool in an Azure Synapse Analytics workspace.

The implementation details for Azure Synapse Link vary between the two types of data source, but the high-level principle is the same - changes made to tables in the source database are synchronized to the target database.

Change feed

Azure Synapse Link for SQL uses the *change feed* feature in Azure SQL Database and Microsoft SQL Server 2022 to capture changes to the source tables. All data modifications are recorded in the transaction log for the source database. The change feed feature monitors the log and applies the same data modifications in the target database. In the case of Azure SQL Database, the modifications are made directly to the target database. When using Azure Synapse Link for SQL Server, the changes are recorded in files and saved to a *landing zone* in Azure Data Lake Gen2 storage before being applied to the target database.

Note: Change feed is similar to the change data capture (CDC) feature in SQL Server. The key difference is that CDC is used to reproduce data modifications in a table in the same database as the modified table. Change feed caches the data modification in memory and forwards it to Azure Synapse Analytics.

After implementing Azure Synapse Link for SQL, you can use system views and stored procedures in your Azure SQL Database or SQL Server database to monitor and manage change feed activity.

1. https://learn.microsoft.com/en-us/sql/sql-server/synapse-link/synapse-link-sql-server-change-feed
2. https://learn.microsoft.com/en-us/sql/sql-server/synapse-link/synapse-link-sql-server-change-feed-manage

Question 73: **Incorrect**

With Azure Synapse Analytics pipelines, you can orchestrate data transfer and transformation activities and build data integration solutions across multiple systems. When you're working with analytical data in a data lake, Apache Spark provides a scalable, distributed processing platform that you can use to process huge volumes of data efficiently.

The Synapse Notebook activity enables you to run data processing code in Spark notebooks as a task in a pipeline; making it possible to automate big data processing and integrate it into *extract, transform, and load* (ETL) workloads.

What kind of pool is required to run a Synapse Notebook in a pipeline?

- **A Dedicated SQL pool**
- **A Data Explorer pool**
- **An Apache Spark pool**

  (Correct)
- **A Cosmos DB**

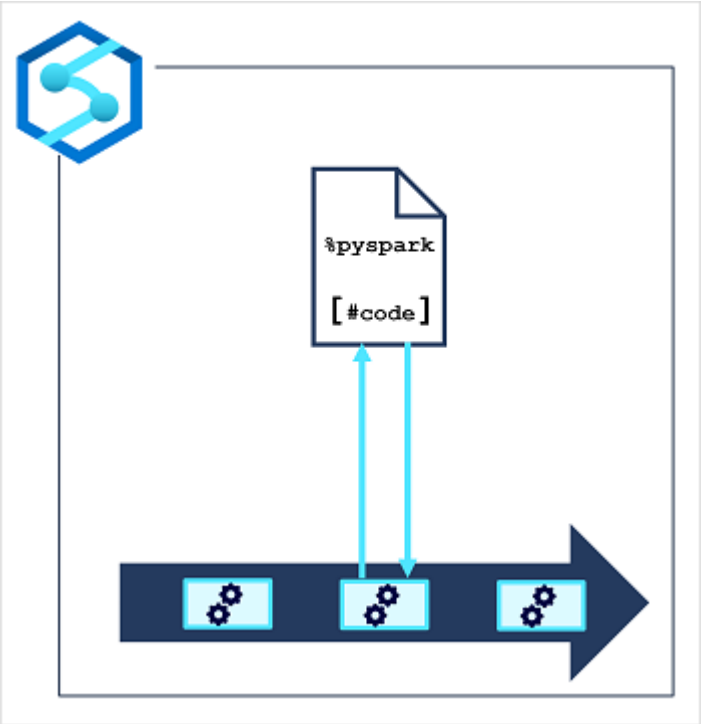
  (Incorrect)

Explanation

- *Notebooks are run in an Apache Spark pool.*
- A dedicated SQL pool is used to host a relational data warehouse.
- A Data Explorer pool is used to run Kusto queries.
- A Cosmos DB is a fully managed NoSQL and relational database for modern app development.

Understand Synapse Notebooks and Pipelines

Azure Synapse Pipelines enable you to create, run, and manage data integration and data flow activities. While many of these activities are built-into the Azure Synapse Pipeline platform and run natively in the integration runtime for your pipeline, you can also use external processing resources to perform specific tasks. One such external resource is an Apache Spark pool in your Azure Synapse Analytics workspace on which you can run code in a notebook.



It's common in big data analytics solutions for data engineers to use Spark notebooks for initial data exploration and interactive experimentation when designing data transformation processes. When the transformation logic has been completed, you can perform some final code optimization and refactoring for maintainability, and then include the notebook in a pipeline. The pipeline can then be run on a schedule or in response to an event (such as new data files being loaded into the data lake).

The notebook is run on a Spark pool, which you can configure with the appropriate compute resources and Spark runtime for your specific workload. The pipeline itself is run in an integration runtime that orchestrates the activities in the pipeline, coordinating the external services needed to run them.

Best Practices

There are several best practices that can help make working with Spark notebooks more efficient and effective. Some of these include:

- Keep your code organized: Use clear and descriptive variable and function names, and organize your code into small, reusable chunks.
- Cache intermediate results: Spark allows you to cache intermediate results, which can significantly speed up the performance of your notebook.

- Avoid unnecessary computations: Be mindful of the computations you are performing and try to avoid unnecessary steps. For example, if you only need a subset of your data, filter it out before running any further computations.
- Avoid using collect() unless necessary: When working with large datasets, it is often better to perform operations on the entire dataset rather than bringing the data into the driver node using the collect() method.
- Use Spark UI for monitoring and debugging: Spark's web-based user interface (UI) provides detailed information about the performance of your Spark jobs, including task execution times, input and output data sizes, and more.
- Keep your dependencies version-consistent and updated: when working with Spark, it is important to keep dependencies version-consistent across your cluster and to use the latest version of Spark and other dependencies if possible.

https://learn.microsoft.com/en-us/azure/synapse-analytics/get-started-pipelines

Question 74: **Incorrect**
Scenario: You are working on a project and your team is moving data from an Azure Data Lake Gen2 store to Azure Synapse Analytics. The team is planning to do a data copy activity and you are discussing with integration runtime to use.

Which Azure Data Factory integration runtime should be used in a data copy activity?

- **Datasets**
- **Azure-SSIS**
- **Azure**

  (Correct)
- **Linked Services**
- **Activities**

-

  Self-hosted

  (Incorrect)

Explanation
When moving data between Azure data platform technologies, the Azure Integration runtime is used when copying data between two Azure data platform.

Integration runtime types

Data Factory offers three types of Integration Runtime, and you should choose the type that best serve the data integration capabilities and network environment needs you are looking for. These three types are:

• Azure

• Self-hosted

• Azure-SSIS

You can explicitly define the Integration Runtime setting in the **connectVia** property, if this is not defined, then the default Integration Runtime is used with the property set to Auto-Resolve.

The following describes the capabilities and network support for each of the integration runtime types:

IR type: Azure

Public network: Data Flow Data movement Activity dispatch

Private network: --

IR type: Self-hosted

Public network: Data movement Activity dispatch

Private network: Data movement Activity dispatch

IR type: Azure-SSIS

Public network: SSIS package execution

Private network: SSIS package execution

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

Question 75: **Incorrect**
Azure Data Factory provides a variety of methods for ingesting data, and also provides a range of methods to perform transformations.

These methods are:

• Mapping Data Flows

• Compute Resources

• SSIS Packages

Mapping Data Flows provides a number of different transformations types that enable you to modify data. They are broken down into the following categories:

• Schema modifier transformations

• Row modifier transformations

• Multiple inputs/outputs transformations

Which transformations type is best described by:

"A Sort transformation that orders the data."

- **Multiple inputs/outputs transformations**
- **None of the listed options.**

  (Incorrect)
-

Row modifier transformations

(Correct)

-

Transformations of schema modifiers

Explanation

Just as Azure Data Factory provides a variety of methods for ingesting data, it also provides a range of methods to perform transformations. You can pick a method that matches the skillsets of your team or takes advantage of existing technologies that you already have in your data estate. There is also the opportunity to perform transformations without writing code at all using the Mapping Data Flow.

Transforming data using Mapping Data Flow

Mapping Data Flows provide an environment for building a wide range of data transformations visually without the need to use code. The resulting data flows that are created are subsequently executed on scaled-out Apache Spark clusters that are automatically provisioned when you execute the Mapping Data Flow. Mapping Data Flows also provides the capability to monitor the execution of the transformations so that you can view how the transformations are progressing, or to understand any errors that may occur

Mapping Data Flows provides a number of different transformations types that enable you to modify data. They are broken down into the following categories:

Category Name: Schema modifier transformations

Description: These types of transformations will make a modification to a sink destination by creating new columns based on the action of the transformation. An example of this is the Derived Column transformation that will create a new column based on the operations performed on existing column.

Category Name: Row modifier transformations

Description: These types of transformations impact how the rows are presented in the destination. An example of this is a Sort transformation that orders the data.

Category Name: Multiple inputs/outputs transformations

Description: These types of transformations will generate new data pipelines or merge pipelines into one. An example of this is the Union transformation that combines multiple data streams.

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

Question 76: **Incorrect**
There are a range of network security steps that you should consider to secure Azure Synapse Analytics. One of the first aspects that you will consider is securing access to the service itself. This can be achieved by creating the following network objects including:

• Firewall rules

• Virtual networks

• Private endpoints

Which of the following are benefits of using a managed workspace virtual network? (Select all that apply)

- ☐

  Managed workspace Virtual Network along with Managed private endpoints protects against data exfiltration.

  (Correct)

- ☐

  It ensures that your workspace is a consolidated network with your other workspaces.

- ☑

  With a Managed workspace Virtual Network, you can offload the burden of managing the Virtual Network to Azure Synapse.

  (Correct)

- ☐

You don't have to configure inbound NSG rules on your own Virtual Networks to allow Azure Synapse management traffic to enter your Virtual Network.

(Correct)

- ☐

You don't need to create a subnet for your Spark clusters based on peak load.

(Correct)

Explanation

There are a range of network security steps that you should consider to secure Azure Synapse Analytics. One of the first aspects that you will consider is securing access to the service itself. This can be achieved by creating the following network objects including:

• Firewall rules

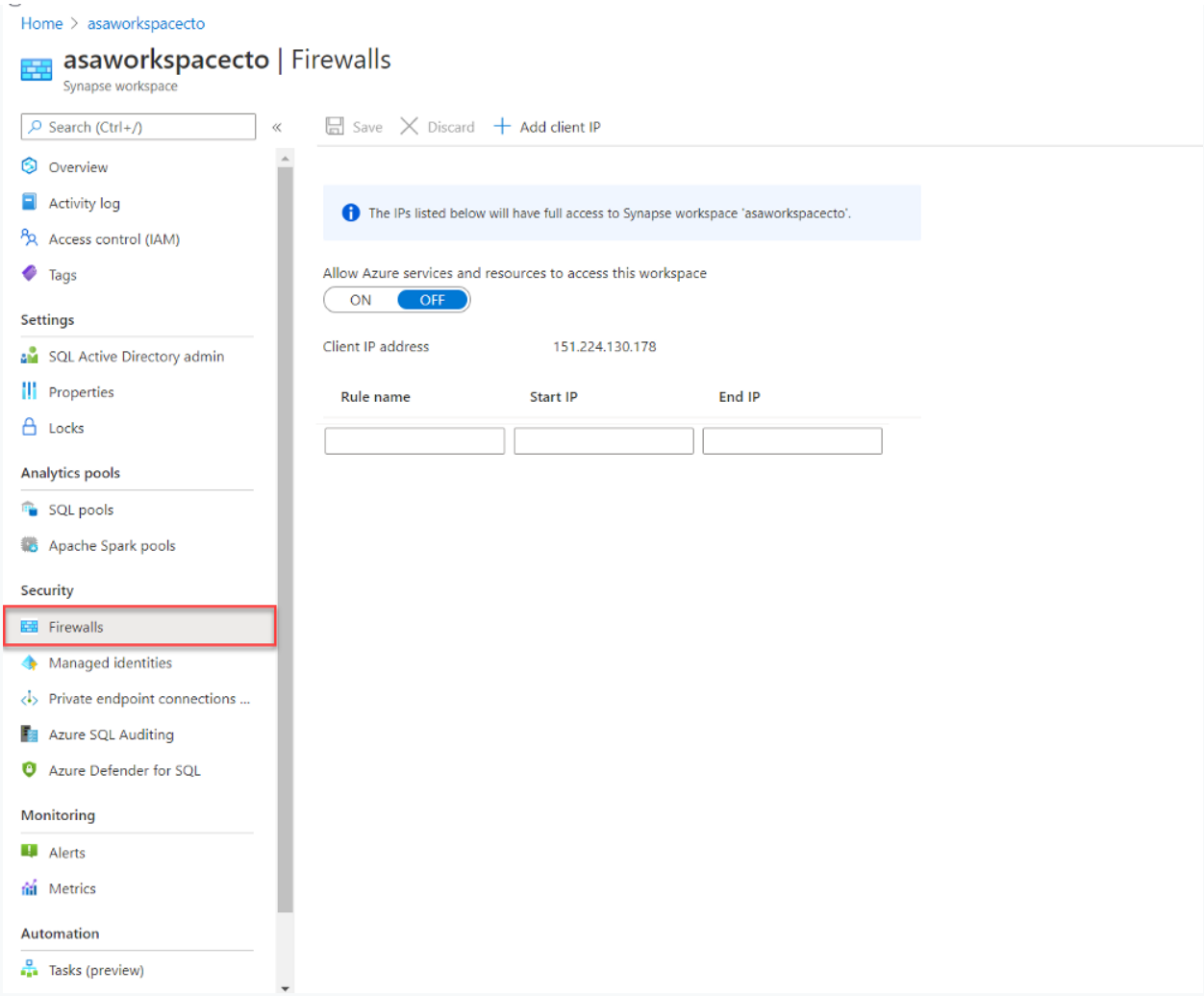
• Virtual networks

• Private endpoints

Firewall rules

Firewall rules enable you to define the type of traffic that is allowed or denied access to an Azure Synapse workspace using the originating IP address of the client that is trying to access the Azure Synapse Workspace. IP firewall rules configured at the workspace level apply to all public endpoints of the workspace including dedicated SQL pools, serverless SQL pool, and the development endpoint.

You can choose to allow connections from all IP addresses as you are creating the Azure Synapse Workspaces, although this is not recommended as it does not allow for control access to the workspace. Instead, within the Azure portal, you can configure specific IP address ranges and associate them with a rule name so that you have greater control.



Make sure that the firewall on your network and local computer allows outgoing communication on TCP ports 80, 443 and 1443 for Synapse Studio.

Also, you need to allow outgoing communication on UDP port 53 for Synapse Studio. To connect using tools such as SSMS and Power BI, you must allow outgoing communication on TCP port 1433.

https://docs.microsoft.com/en-us/azure/synapse-analytics/security/synapse-workspace-ip-firewall

Virtual networks

Azure Virtual Network (VNet) enables private networks in Azure. VNet enables many types of Azure resources, such as Azure Synapse Analytics, to securely communicate with other virtual networks, the internet, and on-premises networks. When you create your Azure Synapse workspace, you can choose to associate it to a Microsoft Azure Virtual Network. The Virtual Network associated with your workspace is managed by Azure Synapse. This Virtual Network is called a Managed workspace Virtual Network.

Using a managed workspace virtual network provides the following benefits:

• With a Managed workspace Virtual Network, you can offload the burden of managing the Virtual Network to Azure Synapse.

• You don't have to configure inbound NSG rules on your own Virtual Networks to allow Azure Synapse management traffic to enter your Virtual Network. Misconfiguration of these NSG rules causes service disruption for customers.

• You don't need to create a subnet for your Spark clusters based on peak load.

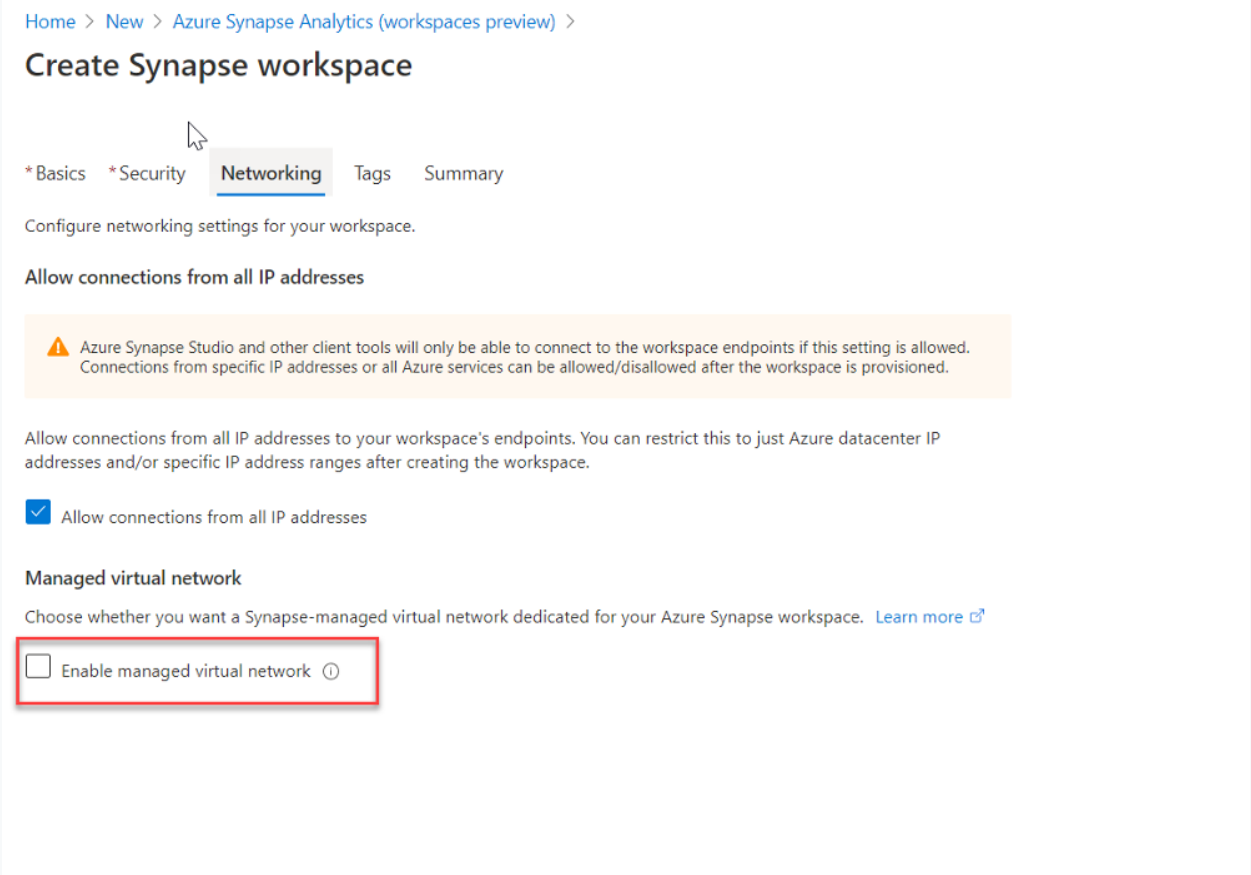
• Managed workspace Virtual Network along with Managed private endpoints protects against data exfiltration. You can only create Managed private endpoints in a workspace that has a Managed workspace Virtual Network associated with it.

• It ensures that your workspace is network isolated from other workspaces.

If your workspace has a Managed workspace Virtual Network, Data integration and Spark resources are deployed in it. A Managed workspace Virtual Network also provides user-level isolation for Spark activities because each Spark cluster is in its own subnet.

Dedicated SQL pool and serverless SQL pool are multi-tenant capabilities and therefore reside outside of the Managed workspace Virtual Network. Intra-workspace communication to dedicated SQL pool and serverless SQL pool use Azure private links. These private links are automatically created for you when you create a workspace with a Managed workspace Virtual Network associated to it.

You can only choose to enable managed virtual networks as you are creating the Azure Synapse Workspaces.



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

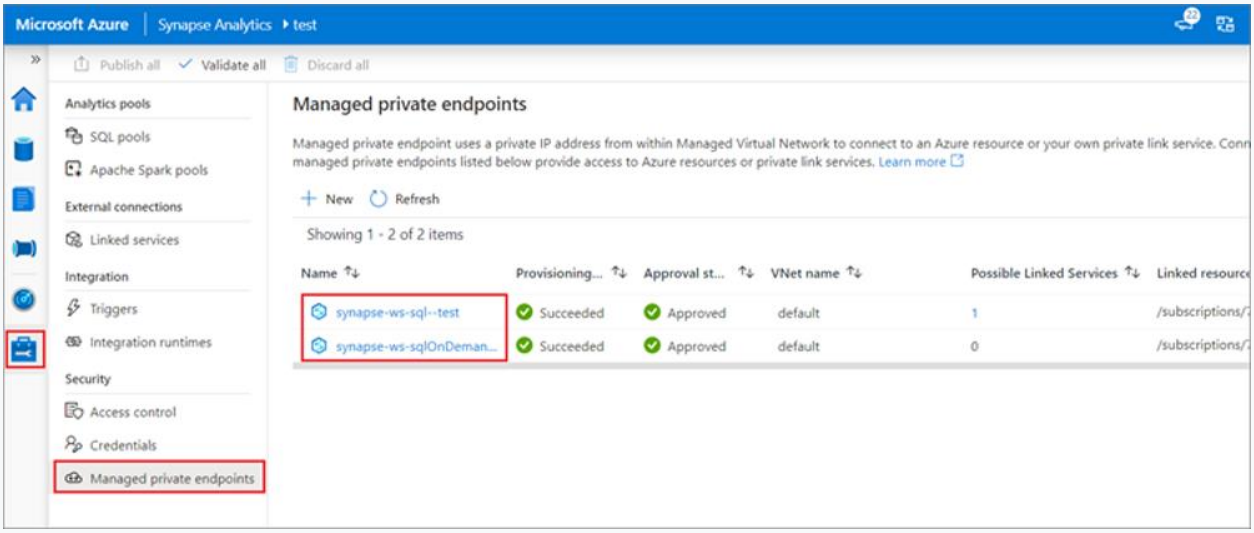
Private endpoints

Azure Synapse Analytics enables you to connect upto its various components through endpoints. You can set up managed private endpoints to access these components in a secure manner known as private links. This can only be achieved in an Azure Synapse workspace with a Managed workspace Virtual Network. Private link enables you to access Azure services (such as Azure Storage and Azure Cosmos DB) and Azure hosted customer/partner services from your Azure Virtual Network securely.

When you use a private link, traffic between your Virtual Network and workspace traverses entirely over the Microsoft backbone network. Private Link protects against data exfiltration risks. You establish a private link to a resource by creating a private endpoint.

Private endpoint uses a private IP address from your Virtual Network to effectively bring the service into your Virtual Network. Private endpoints are mapped to a specific

resource in Azure and not the entire service. Customers can limit connectivity to a specific resource approved by their organization. You can manage the private endpoints in the Azure Synapse Studio manage hub.



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

Question 77: **Incorrect**
What is a dataframe?

- **An Array**
- **A creation of a data structure**

  (Correct)
- **A CSV file**
- **A parquet file**

(Incorrect)

Explanation
A DataFrame creates a data structure and it's one of the core data structures in Spark.

What are dataframes?

Basically you could view DataFrames as you might see in excel. It's like a box with squares in it, that organizes data, which we could also refer to as a table of data.

What does a table of data mean?

It is a single set of two-dimensional data that can have multiple rows and columns in the data. Each row, is a sample of data. Each column is a variable or parameter that is able to describe the row that contains the sample of data.

A DataFrame creates a data structure and it's one of the core data structures in Spark. In Spark, it is seen as a distributed collection of data that is organized into columns that have names.

What you see in Data Engineering is that you start with reading or loading data that can be unstructured, semi-structured, or structured, which is stored in a DataFrame and start transforming that data in order to get insights. You can use different functionalities in order to do so, like using Spark SQL, PySpark, and others.

Usually when you see 'df' in some code it refers to a dataframe.

You can either create your own dataframe as this example shows:

```
1. Python
2. new_rows = [('CA',22, 45000),("WA",35,65000) ,("WA",50,85000)]
3. demo_df = spark.createDataFrame(new_rows, ['state', 'age', 'salary'])
4. demo_df.show()
```

Or load a file that contains data into a dataframe like in the below example where the open taxi dataset is used:

```
1. Python
2. from azureml.opendatasets import NycTlcYellow
3.
4. data = NycTlcYellow()
5. data_df = data.to_spark_dataframe()
6. display(data_df.limit(10))
```

Once you're at the stage where you'd like to manipulate the data that is stored in a DataFrame, you can use User-Defined Functions (UDFs) that are column-based and help you transform and manipulate the data stored in a DataFrame.

https://www.tutorialspoint.com/spark_sql/spark_sql_dataframes.htm

Question 78: **Incorrect**
What function provides a `rowset` view over a JSON document?

- `OPENROWSET`
- `OPENJSON`

  (Correct)
- `VIEWRSET`

  (Incorrect)
- `WITH`

Explanation
`OPENJSON` (Transact-SQL) is a table-valued function that parses JSON text and returns objects and properties from the JSON input as rows and columns. In other words, `OPENJSON` provides a rowset view over a JSON document. You can explicitly specify the columns in the rowset and the JSON property paths used to populate the columns. Since `OPENJSON` returns a set of rows, you can use `OPENJSON` in the FROM clause of a Transact-SQL statement just as you can use any other table, view, or table-valued function.

Use `OPENJSON` to import JSON data into SQL Server, or to convert JSON data to relational format for an app or service that can't consume JSON directly.

The `OPENJSON` function provides a `rowset` view over a JSON document.

https://docs.microsoft.com/en-us/sql/t-sql/functions/openjson-transact-sql?view=sql-server-ver15

Question 79: **Incorrect**
Scenario: Queen Consolidated was overtaken by Raymond Carson Palmer and rebranded as Palmer Technologies. Now that Ray is overseeing the operations at Palmer, Ray has decided to implement on-premises Microsoft SQL Server pipelines by using a custom solution.

Currently, you are in a meeting with the IT team and discussing a project to pull data from SQL Server and migrate it to Azure Blob storage.

Required:

• The process must orchestrate and manage the data lifecycle.

• The process must configure Azure Data Factory to connect to the on-premises SQL Server database.

Ray and the IT team have put together a list of actions they think need to be performed to meet the needs of the project, but they are not sure on the order to execute. Below is a list of the actions they are considering.

Proposed Actions:

a. Create an Azure Data Factory resource.

b. Install and configure Azure Data Factory Integration Runtime

c. Create a virtual private network (VPN) connection from on-prem to MS Azure.

d. Create a database master key on SQL Server.

e. Backup the database and send it to Azure Blob storage.

f. Configure the on-prem SQL Server instance with an integration runtime.

As you are the Azure SME, Ray and the team look to you for direction on selecting the required items and putting them in the proper order. Which of the below contains the correct items in the correct sequence to meet the requirements?

- **c → d → a → b → f**

  (Incorrect)

- **c → a → b → f**

-

d → c → e → b

- **a → b → f**

  (Correct)

Explanation
Step 1: Create an Azure Data Factory

The instructions for creating a new Azure Data Factory and a resource group in the Azure portal are provided Create an Azure Data Factory. Name the new ADF instance adfdsp and name the resource group created adfdsprg.

Step 2: Install and configure Azure Data Factory Integration Runtime

The Integration Runtime is a customer-managed data integration infrastructure used by Azure Data Factory to provide data integration capabilities across different network environments. This runtime was formerly called "Data Management Gateway".

To set up, follow the instructions for creating a pipeline

Step 3: Configure the on-prem SQL Server instance with an integration runtime.

Create linked services to connect to the data resources. A linked service defines the information needed for Azure Data Factory to connect to a data resource. We have three resources in this scenario for which linked services are needed:

1. On-premises SQL Server

2. Azure Blob Storage

3. Azure SQL Database

https://docs.microsoft.com/pt-pt/azure/machine-learning/team-data-science-process/move-sql-azure-adf

It's not necessary to *Create a virtual private network (VPN) connection from on-premises to Microsoft Azure* - all communication from IR to ADF is over HTTPS, ∴ **VPN is not a Required item.**

Encryption in transit

All data transfers are via secure channel HTTPS and TLS over TCP to prevent man-in-the-middle attacks during communication with Azure services.

You can also use IPSec VPN or Azure ExpressRoute to further secure the communication channel between your on-premises network and Azure.

Azure Virtual Network is a logical representation of your network in the cloud. You can connect an on-premises network to your virtual network by setting up IPSec VPN (site-to-site) or ExpressRoute (private peering).

The following table summarizes the network and self-hosted integration runtime configuration recommendations based on different combinations of source and destination locations for hybrid data movement.

Source	Destination	Network configuration	Integration runtime setup
On-premises	Virtual machines and cloud services deployed in virtual networks	IPSec VPN (point-to-site or site-to-site)	The self-hosted integration runtime should be installed on an Azure virtual machine in the virtual network.
On-premises	Virtual machines and cloud services deployed in virtual networks	ExpressRoute (private peering)	The self-hosted integration runtime should be installed on an Azure virtual machine in the virtual network.
On-premises	Azure-based services that have a public endpoint	ExpressRoute (Microsoft peering)	The self-hosted integration runtime can be installed on-premises or on an Azure virtual machine.

https://docs.microsoft.com/en-us/azure/data-factory/data-movement-security-considerations

Move data from a SQL Server database to the SQL Database with Azure Data Factory

AZure Data Factory is a fully managed cloud-based data integration service that orchestrates and automates the movement and transformation of data. The key concept in the ADF model is the pipeline. A pipeline is a logical grouping of Activities,

each of which defines the actions to be performed on the data contained in the Data Sets. The linked services are used to define the information necessary for the Data Factory to connect to the data resources.

With the ADF, existing data processing services can be composed of highly available data pipelines and managed in the cloud. These data pipelines can be programmed to ingest, prepare, transform, analyze and publish data, and the ADF manages and orchestrates the complex data and processing dependencies. Solutions can be quickly built and deployed in the cloud, connecting an increasing number of data sources on-premises and in the cloud.

Consider using the ADF:

• when data needs to be continuously migrated in a hybrid scenario that accesses both on-premises and cloud resources

• when data needs transformation or has business logic added to it when it is migrated.

The ADF allows scheduling and monitoring of jobs using simple JSON scripts that manage the movement of data on a periodic basis. ADF also has other capabilities, such as supporting complex operations. For more information about the ADF, see the documentation at Azure Data Factory (ADF).

The set

We created an ADF pipeline that comprises two data migration activities. Together, they move data daily between a SQL Server database and the Azure SQL Database. The two activities are:

• Copy data from a SQL Server database to an Azure Blob storage account.

• Copy the data from the Azure Blob storage account to the Azure SQL Database.

Upload the data to your instance of SQL Server

Create an Azure Data Factory

Instructions for creating a new Azure Data Factory and resource group on the Azure portal are provided Create an Azure Data Factory. Name the new adf instance adfdsp and name the resource group created adfdsprg.

Install and configure Azure data factory integration time

Integration Runtime is a customer-managed data integration infrastructure used by Azure Data Factory to provide data integration capabilities in different network environments. This uptime was previously called "Data Management Gateway".

https://docs.microsoft.com/en-us/azure/machine-learning/team-data-science-process/move-sql-azure-adf

Question 80: **Correct**
Apache Spark offers data engineers a powerful platform for transforming and processing data. The ability to include Spark notebooks in a pipeline enables you to automate Spark processing and integrate it into a data integration workflow.

A given notebook cell contains variable declarations.

How can these variable declarations be used as parameters?

- **Add a `%%Decl` magic at the beginning of the cell**
- **Add a `%%Spark` magic at the beginning of the cell**
- **Use the `var` keyword for each variable declaration**
- **Toggle the Parameters cell setting for the cell**

  (Correct)

Explanation

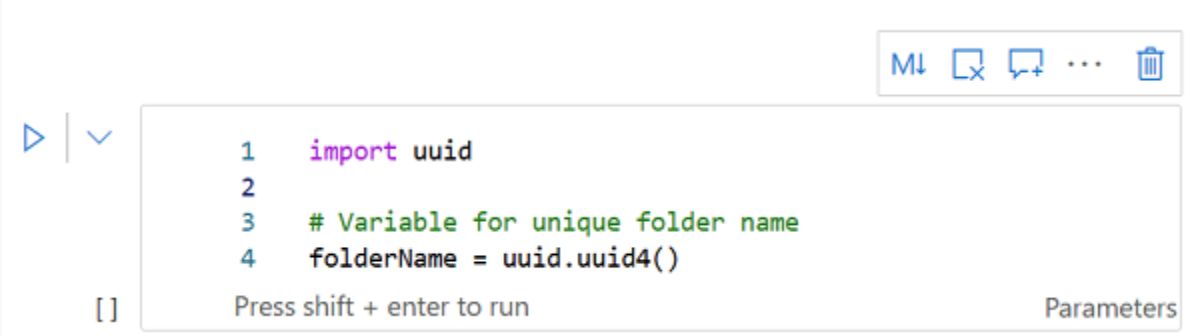
- *Toggle the Parameters cell option to use variables as parameters.*
- A `%%Spark` magic would configure the cell to be run as Scala code.
- The `var` keyword is used to declare variables in Scala.
- `%%Decl` is not a valid option.

Use parameters in a notebook

Parameters enable you to dynamically pass values for variables in the notebook each time it's run. This approach provides flexibility, enabling you to adjust the logic encapsulated in the notebook for each run.

Create a *parameters* cell in the notebook

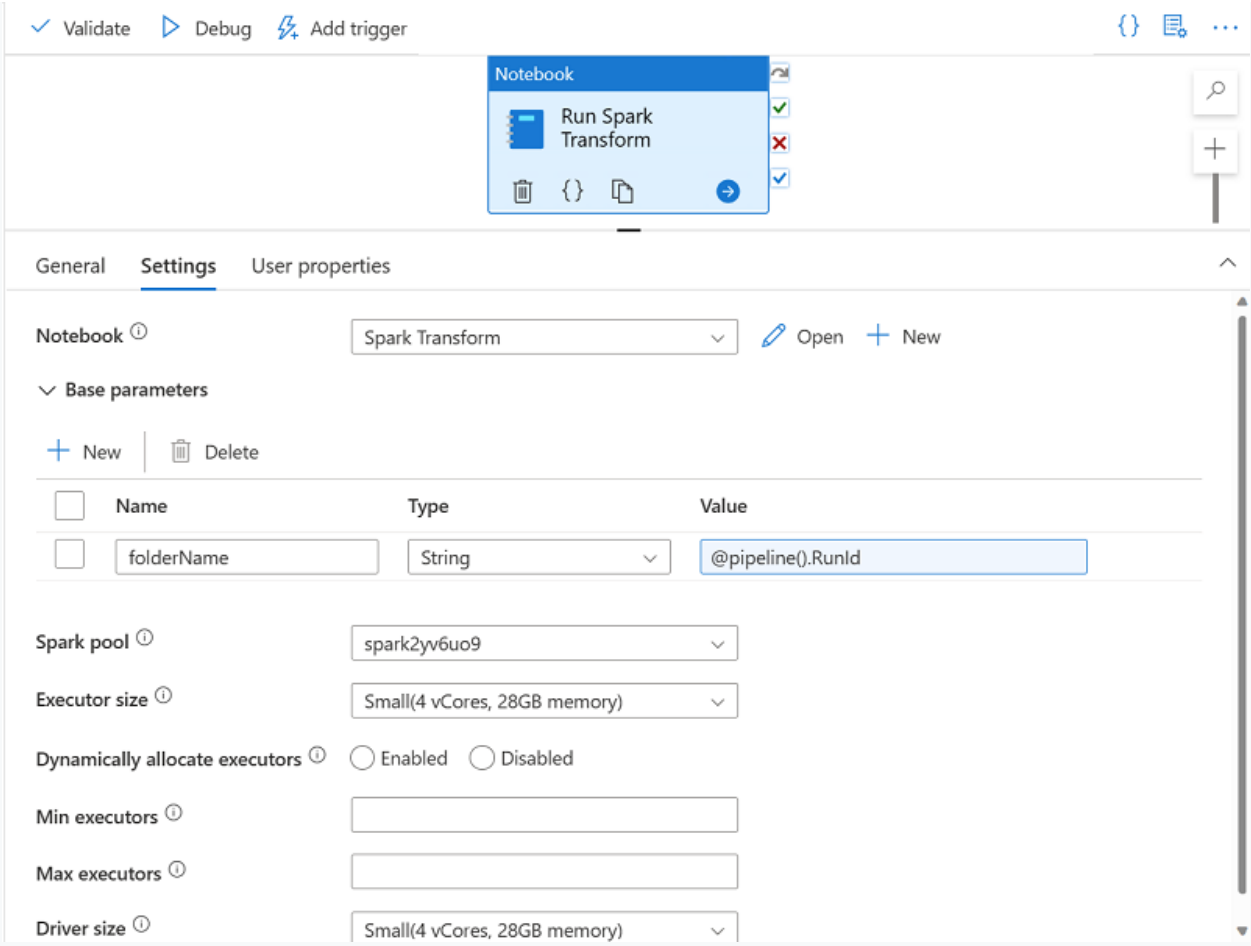
To define the parameters for a notebook, you declare and initialize variables in a cell, which you then configure as a **Parameters** cell by using the toggle option in the notebook editor interface.



Initializing a variable ensures that it has a default value, which will be used if the parameter isn't set in the notebook activity.

Set *base parameters* for the notebook activity

After defining a parameters cell in the notebook, you can set values to be used when the notebook is run by a notebook activity in a pipeline. To set parameter values, expand and edit the **Base parameters** section of the settings for the activity.



You can assign explicit parameter values, or use an expression to assign a dynamic value. For example, the expression `@pipeline().RunId` returns the unique identifier for the current run of the pipeline.

https://learn.microsoft.com/en-us/sql/azure-data-studio/notebooks/run-with-parameters?view=sql-server-ver16

Question 81: **Incorrect**
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. When it comes to the languages, a notebook has to be set with a primary language.

True or False: It is possible to use multiple languages in one notebook.

- True

- True

  (Correct)

- False

  (Incorrect)

Explanation
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. When it comes to the languages, a notebook has to be set with a primary language.

The primary languages available within the notebook environment are:

• PySpark (Python)

• Spark (Scala)

• .NET Spark (C#)

• Spark SQL

However, it is possible to use multiple languages in one notebook by specifying the language using a 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.

It is not possible to reference data or variables directly across different languages in a Synapse Studio notebook. In Spark, it is possible to reference a temporary table across languages.

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

Question 82: **Incorrect**
All data written to Azure Storage is automatically encrypted by Storage Service Encryption (SSE) with a 256-bit Advanced Encryption Standard (AES) cipher, and is FIPS 140-2 compliant.

True or False: For virtual machines (VMs), Azure lets you encrypt virtual hard disks (VHDs) by using Azure Disk Encryption. If someone gets access to the VHD image and downloads it, they can't access the data on the VHD unless they have an Azure Storage account as well. If a bad actor restores the image within their own Azure environment, they will have access to the data on the image.

- False

  (Correct)

- True

  (Incorrect)

Explanation
Encryption at rest

All data written to Azure Storage is automatically encrypted by Storage Service Encryption (SSE) with a 256-bit Advanced Encryption Standard (AES) cipher, and is FIPS 140-2 compliant. SSE automatically encrypts data when writing it to Azure Storage. When you read data from Azure Storage, Azure Storage decrypts the data before returning it. This process incurs no additional charges and doesn't degrade performance. It can't be disabled.

For virtual machines (VMs), Azure lets you encrypt virtual hard disks (VHDs) by using Azure Disk Encryption. This encryption uses BitLocker for Windows images, and it uses dm-crypt for Linux.

Azure Key Vault stores the keys automatically to help you control and manage the disk-encryption keys and secrets. So even if someone gets access to the VHD image and downloads it, they can't access the data on the VHD.

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

Question 83: **Incorrect**
Because the Databricks API is declarative, a large number of optimizations are available to us. Among the most powerful components of Spark are Spark SQL. At its core lies the Catalyst optimizer.

When you execute code, Spark SQL uses Catalyst's general tree transformation framework in four phases, as shown below:

1. analyzing a logical plan to resolve references

2. logical plan optimization

3. physical planning

4. code generation to compile parts of the query to Java bytecode

In the physical planning phase, Catalyst may generate multiple plans and compare them based on [?].

- **Region**
- **Cost**

  (Correct)
- **Rules**

  (Incorrect)
- **Permissions**

Explanation
Because the Databricks API is declarative, a large number of optimizations are available to us.

Some of the examples include:

• Optimizing data type for storage

• Rewriting queries for performance

• Predicate push downs

Among the most powerful components of Spark are Spark SQL. At its core lies the Catalyst optimizer. This extensible query optimizer supports both rule-based and cost-based optimization.

When you execute code, Spark SQL uses Catalyst's general tree transformation framework in four phases, as shown below:

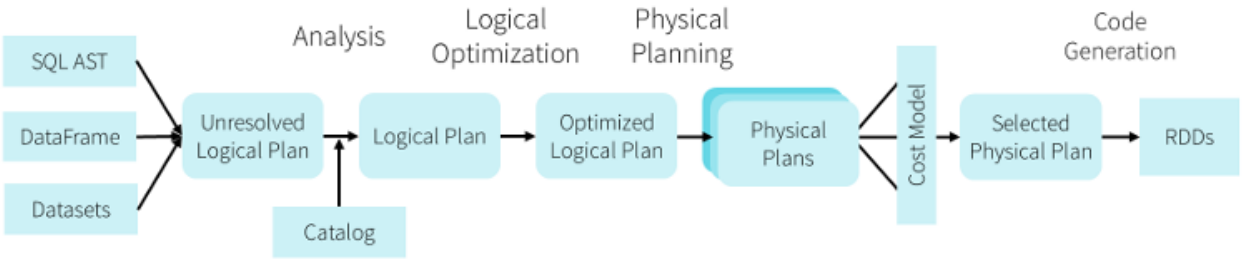
1. analyzing a logical plan to resolve references

2. logical plan optimization

3. physical planning

4. code generation to compile parts of the query to Java bytecode

In the physical planning phase, Catalyst may generate multiple plans and compare them based on cost. All other phases are purely rule-based.



Catalyst is based on functional programming constructs in Scala and designed with these key two purposes:

• Easily add new optimization techniques and features to Spark SQL

• Enable external developers to extend the optimizer (e.g. adding data source specific rules, support for new data types, etc.)

https://data-flair.training/blogs/spark-sql-optimization/

Question 84: **Incorrect**
Which Azure data platform is commonly used to process data in an ELT framework?

- **Azure Data Lake Storage**
- **Azure Stream Analytics**
- **Azure Data Catalog**
- **Azure Data Factory**

  (Correct)
- **Azure Databricks**

  (Incorrect)

Explanation
Azure Data Factory

Data Factory is a cloud-integration service. It orchestrates the movement of data between various data stores.

As a data engineer, you can create data-driven workflows in the cloud to orchestrate and automate data movement and data transformation. Use Data Factory to create and schedule data-driven workflows (called pipelines) that can ingest data from data stores.

Data Factory processes and transforms data by using compute services such as Azure HDInsight, Hadoop, Spark, and Azure Machine Learning. Publish output data to data

stores such as Azure SQL Data Warehouse so that business intelligence applications can consume the data. Ultimately, you use Data Factory to organize raw data into meaningful data stores and data lakes so your organization can make better business decisions.

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

Question 85: **Incorrect**

Microsoft Azure Storage is a managed service that provides durable, secure, and scalable storage in the cloud. You can create an Azure storage account using the Azure Portal, Azure PowerShell, or Azure CLI. Azure Storage provides three distinct account options with different pricing and features supported.

Which of the Azure Storage account options is best described by:

"Support all of the latest features for blobs, files, queues, and tables. Pricing has been designed to deliver the lowest per gigabyte prices."

- **GPv1**
- **Append**
- **Block**
- **Page**

  (Incorrect)
- **Blob storage accounts**
- **GPv2**

(Correct)

- **Queue**

Explanation

Create a storage account

You can create an Azure storage account using the Azure portal, Azure PowerShell, or Azure CLI. Azure Storage provides three distinct account options with different pricing and features supported.

General-purpose v1 (GPv1)

General-purpose v1 (GPv1) accounts provide access to all Azure Storage services but may not have the latest features or the lowest per gigabyte pricing. For example, cool storage and archive storage are not supported in GPv1. Pricing is lower for GPv1 transactions, so workloads with high churn or high read rates may benefit from this account type.

General-purpose v2 (GPv2)

General-purpose v2 (GPv2) accounts are storage accounts that support all of the latest features for blobs, files, queues, and tables. Pricing for GPv2 accounts has been designed to deliver the lowest per gigabyte prices.

Blob storage accounts

A legacy account type, blob storage accounts support all the same block blob features as GPv2, but they are limited to supporting only block and append blobs. Pricing is broadly similar to pricing for general-purpose v2 accounts.

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

Question 86: **Incorrect**
Which of the below have the following characteristics?

• Provide undoubtedly the most well-understood model for holding data.

• The simplest structure of columns and tables makes them very easy to use initially, but the inflexible structure can cause some problems.

• We can communicate with relational databases using SQL.

- ○

  Key-Value

- ○

  Relational

  (Correct)

- ○

  Non-Relational

- ◉

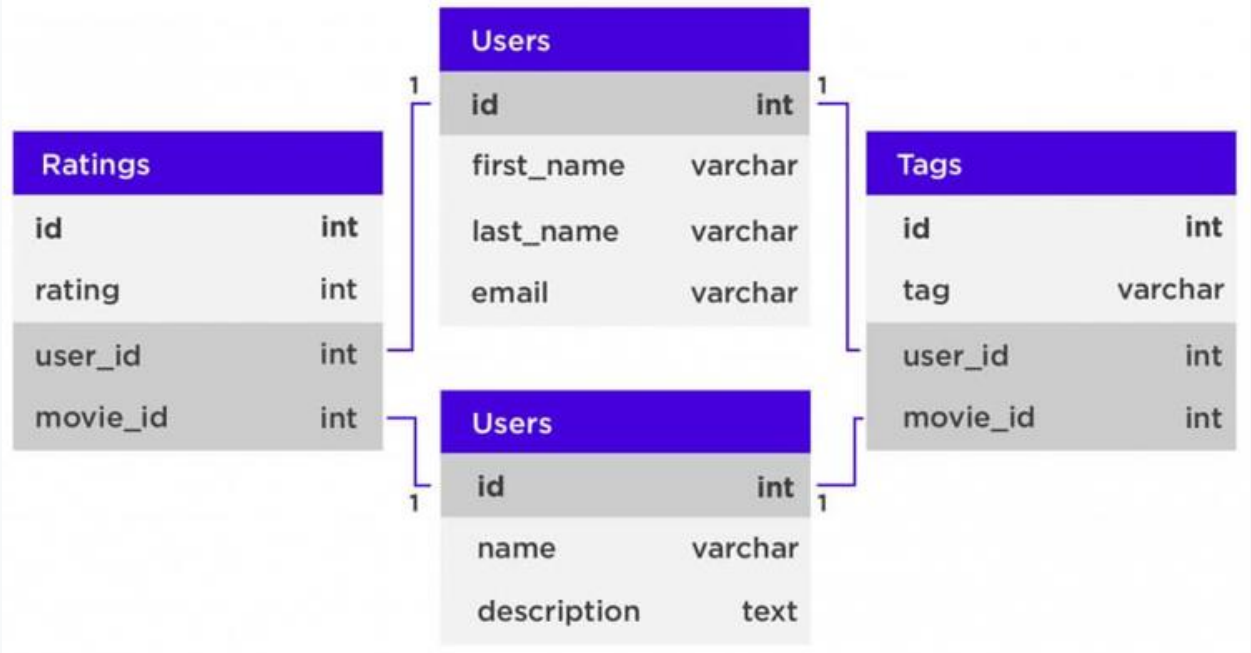
  JSON

  (Incorrect)

Explanation
Relational Data

• Relational databases provide undoubtedly the most well-understood model for holding data.

• The simplest structure of columns and tables makes them very easy to use initially, but the inflexible structure can cause some problems.



• We can communicate with relational databases using **Structured Query Language (SQL).**

• SQL allows the joining of tables using a few lines of code, with a structure most beginner employees can learn very fast.

• Examples of relational databases:

- MySQL
- PostgreSQL
- Db2
- SQL Server

https://f5a395285c.nxcli.net/microsoft-azure/dp-900/structured-data-vs-unstructured-data-vs-semi-structured-data/

Question 87: **Correct**

Which feature in alerts can be used to determine how an alert is fired?

- **Add severity**
- **Add specifications**
- **Add rule**
- **Add criteria**

  (Correct)

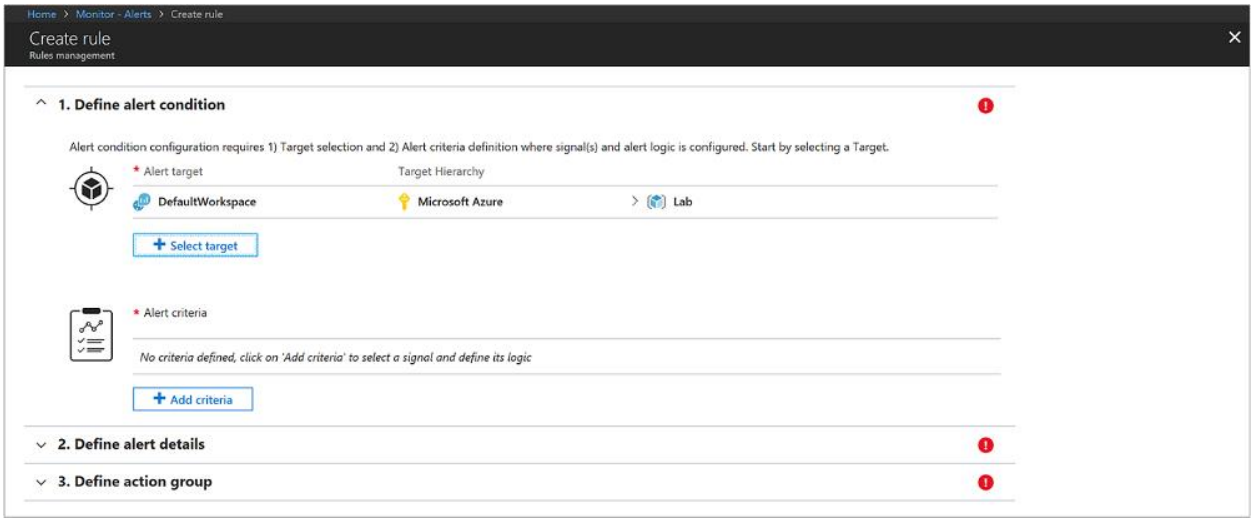
Explanation

Azure Data Factory Alerts provide an automated response that can be beneficial to monitor and audit Azure Data Factory activity. These alerts are very proactive and more efficient than manual monitoring operations. Alerts can be fired on both success and failure of a pipeline based on the rule configuration.

Alert Rule

Azure Data Factory Alerts use an alert rule which states the criteria upon which the alerts should trigger. We can enable or disable the alert rules.

• The **add criteria** feature enables you to determine how an alert is fired.



https://docs.microsoft.com/en-us/azure/azure-monitor/alerts/tutorial-response

Question 88: **Incorrect**

With the Azure-SSIS integration runtime installed and SQL Server Data Tools (SSDT) you have the capability to deploy and manage SSIS packages that you create in the cloud. For some packages, you may be able to rebuild them by redeploying them in the Azure-SSIS runtime. However, there may be some SSIS packages that already exist within your environment that may not be compatible.

You can use the [?] to perform an assessment of the SSIS packages that exist and identify any compatibility issues with them.

-

  Azure Advisor

-

Azure Lab Services

-

Azure ARM templates

-

Azure SQL Server Management Studio

(Incorrect)

-

Azure SQL Server Upgrade Advisor

-

Azure Data Migration Assistant

(Correct)

Explanation
With the Azure-SSIS integration runtime installed and SQL Server Data Tools (SSDT) you have the capability to deploy and manage SSIS packages that you create in the cloud. For some packages, you may be able to rebuild them by redeploying them in the Azure-SSIS runtime. However, there may be some SSIS packages that already exist within your environment that may not be compatible? How should you deal with them?

Perform assessments of your SSIS packages.

When you migrate your database workloads from SQL Server on premises to Azure SQL database services, you may have to migrate SSIS packages as well. The first step required is to perform an assessment of you current SSIS packages to make sure that they are compatible in Azure. Fortunately, **you can use the Data Migration Assistant (DMA) to perform an assessment of the SSIS packages that exist and identify any compatibility issues with them.** The Data Migration Assistant has two main categories of information:

• Migration blockers: Issues that prevent your existing SSIS packages to run on Azure-SSIS Integration Runtime environments.

• Information issues: SSIS features within your packages that are only partially supported, or are deprecated. Regardless of which category of information you receive,

the Data Migration Assistant will perform the assessment on a batch of SSIS packages and provide guidance and potential mitigation steps that you can use to address the blockers and issues that are raised.

Perform a migration of your packages

Before migrating, you must know which Azure SQL database service you are migrating to. This can include migrating to Azure SQL Managed Instance (MI), or Azure SQL Database. Furthermore, when migrating SSIS packages. you have to consider the location of the SSIS packages that you are migrating, as this can impact how you migrate the packages, and which tool you will need to use. There are four types of storage including:

• SSIS Catalog (also known as SSISDB)

• File System

• MSDB database in SQL Server

• SSIS Package store

Based on this information, you can use the following table as a basis for understanding the tools you can use to perform migration assessments, and to perform the migration itself.

Source: SQL Server + SQL Agent		Destination: Azure SQL MI + MI Agent		Destination: Azure SQL DB + ADF	
Storage Types	**Package Assessment**	**Package Migration**	**Job Migration**	**Package Migration**	**Job Migration**
SSISDB	• Data Migration Assistant tool • SQL Server Data Tools	• Migrate the SSISDB to SSISDB using the Database Migration Service (DMS)	• Migrate SQL Server Agent Jobs to Managed Instance (MI) Agent Using PowerShell, T-SQL, or C# script. • Recreate in the Managed Instance (MI) Agent via SQL Server Management Studio (SSMS)	• Deploy to the SSISDB via SQL Server Data Tools (SSDT) or SQL Server Management Studio (SSMS)	• Migrate SQL Server Agent Jobs to Azure Data Factory (ADF) using PowerShell, T-SQL, or C# scripts • Recreate in Azure Data Factory (ADF) using SQL Server Management Studio (SSMS) or the Azure Data Factory (ADF) portal
File Systems	• Data Migration Assistant tool • SQL Server Data Tools	• Deploy to file shares, or Azure Files using dtinstall, or dtutil, or by a manual copy • Keep in file systems and access via Vnet, or Self-Hosted Integration Runtime (IR)	• Migrate SQL Server Agent Jobs to Managed Instance (MI) Agent Using PowerShell, T-SQL, or C# script. • Recreate in the Managed Instance (MI) Agent via SQL Server Management Studio (SSMS)	• Deploy to file shares, or Azure Files using dtinstall, dtutil, or by a manual copy • Keep in file systems and access via Vnet, or Self-Hosted Integration Runtime (IR)	• Migrate SQL Server Agent Jobs to Azure Data Factory (ADF) using SQL Server Management Studio (SSMS) • Recreate in Azure Data Factory (ADF) using SQL Server Management Studio (SSMS) or the Azure Data Factory (ADF) portal
MSDB	• Data Migration Assistant tool • SQL Server Data Tools	• Export to file systems, file shares, or Azure Files via SQL Server Management Studio (SSMS) or dtutil • Import and export to the Package store, or MSDB via SQL Server Management Studio (SSMS) or dtutil	• Migrate SQL Server Agent Jobs to Azure Data Factory (ADF) using PowerShell, T-SQL, or C# scripts • Recreate in Azure Data Factory (ADF) using SQL Server Management Studio (SSMS) or the Azure Data Factory (ADF) portal	▪ Export to file systems, file shares, or Azure Files using SQL Server Management Studio (SSMS) or dtutil	• Migrate SQL Server Agent Jobs to Azure Data Factory (ADF) using PowerShell, T-SQL, or C# scripts • Recreate in Azure Data Factory (ADF) using SQL Server Management Studio (SSMS) or the Azure Data Factory (ADF) portal
SSIS Package Store	• Data Migration Assistant tool • SQL Server Data Tools	• Export to file systems, file shares, or Azure Files via SQL Server Management Studio (SSMS) or dtutil • Import and export to the Package store, or MSDB via SQL Server Management Studio (SSMS) or dtutil	• Migrate SQL Server Agent Jobs to Azure Data Factory (ADF) using PowerShell, T-SQL, or C# scripts • Recreate in Azure Data Factory (ADF) using SQL Server Management Studio (SSMS) or the Azure Data Factory (ADF) portal	▪ Export to file systems, file shares, or Azure Files using SQL Server Management Studio (SSMS) or dtutil	• Migrate SQL Server Agent Jobs to Azure Data Factory (ADF) using PowerShell, T-SQL, or C# scripts • Recreate in Azure Data Factory (ADF) using SQL Server Management Studio (SSMS) or the Azure Data Factory (ADF) portal

Microsoft Data Migration Assistant

The Data Migration Assistant helps you upgrade to a modern data platform by detecting compatibility issues that can impact database functionality in your new version of SQL Server or Azure SQL Database. DMA recommends performance and reliability improvements for your target environment and allows you to move your schema, data, and objects from your source server to your target server.

This tool can be helpful to you in identifying any issues that can affect a migration to an Azure SQL data platform. The DMA can run assessment projects that will identify any blocking issues or unsupported features that are currently in use with your on-premises SQL Server. It can also help you understand the new features in the target SQL Server platform that the database can benefit from after a migration. The DMA can also perform migration projects that can migrate an on-premises SQL Server instance to a modern SQL Server instance hosted on-premises or on an Azure virtual machine (VM) that is accessible from your on-premises network.

The Data Migration Assistant replaces all previous versions of SQL Server Upgrade Advisor and should be used for upgrades for most SQL Server versions.

https://www.sqlshack.com/move-local-ssis-packages-to-azure-data-factory/

Question 89: **Incorrect**
Scenario: Honest Eddie's Car Dealership is an establishment in South Carolina USA, which is dedicated to the purchase and sale of cars and light trucks.

Eddie's Car Dealership is using Azure where you are working as a contract developer. You use the `RANK` function in a query to rank customers in order of the number of purchases they have made. Five customers have made the same number of purchases and are all ranked equally as 1.

What rank will the customer with the next highest number of purchases be assigned?

- **2**
- **3**

- 5
- 1

  (Incorrect)
- 6

  (Correct)
- 0
- 4

Explanation

- There are five customers with a higher number of purchases, and `RANK` takes these into account.
- A value of 2 would be returned by `DENSE_RANK`

Query a data warehouse

When the dimension and fact tables in a data warehouse have been loaded with data, you can use SQL to query the tables and analyze the data they contain. The Transact-SQL syntax used to query tables in a Synapse dedicated SQL pool is similar to SQL used in SQL Server or Azure SQL Database.

Aggregating measures by dimension attributes

Most data analytics with a data warehouse involves aggregating numeric measures in fact tables by attributes in dimension tables. Because of the way a star or snowflake schema is implemented, queries to perform this kind of aggregation rely

on `JOIN` clauses to connect fact tables to dimension tables, and a combination of aggregate functions and `GROUP BY` clauses to define the aggregation hierarchies.

For example, the following SQL queries the **FactSales** and **DimDate** tables in a hypothetical data warehouse to aggregate sales amounts by year and quarter:

```
SQL
SELECT  dates.CalendarYear,
        dates.CalendarQuarter,
        SUM(sales.SalesAmount) AS TotalSales
FROM dbo.FactSales AS sales
JOIN dbo.DimDate AS dates ON sales.OrderDateKey = dates.DateKey
GROUP BY dates.CalendarYear, dates.CalendarQuarter
ORDER BY dates.CalendarYear, dates.CalendarQuarter;
```

The results from this query would look similar to the following table:

CalendarYear	CalendarMonth	TotalSales
2020	1	25980.16
2020	2	27453.87
2020	3	28527.15
2020	4	31083.45
2021	1	34562.96
2021	2	36162.27
...	...	...

You can join as many dimension tables as needed to calculate the aggregations you need. For example, the following code extends the previous example to break down the quarterly sales totals by city based on the customer's address details in the **DimCustomer** table:

```
SQL
SELECT  dates.CalendarYear,
        dates.CalendarQuarter,
        custs.City,
        SUM(sales.SalesAmount) AS TotalSales
FROM dbo.FactSales AS sales
JOIN dbo.DimDate AS dates ON sales.OrderDateKey = dates.DateKey
JOIN dbo.DimCustomer AS custs ON sales.CustomerKey = custs.CustomerKey
GROUP BY dates.CalendarYear, dates.CalendarQuarter, custs.City
ORDER BY dates.CalendarYear, dates.CalendarQuarter, custs.City;
```

This time, the results include a quarterly sales total for each city:

CalendarYear	CalendarMonth	City	TotalSales
2020	1	Amsterdam	5982.53
2020	1	Berlin	2826.98
2020	1	Chicago	5372.72
...	...	...	..
2020	2	Amsterdam	7163.93
2020	2	Berlin	8191.12
2020	2	Chicago	2428.72
...	...	...	..
2020	3	Amsterdam	7261.92
2020	3	Berlin	4202.65
2020	3	Chicago	2287.87
...	...	...	..
2020	4	Amsterdam	8262.73
2020	4	Berlin	5373.61
2020	4	Chicago	7726.23
...	...	...	..
2021	1	Amsterdam	7261.28
2021	1	Berlin	3648.28
2021	1	Chicago	1027.27
...	...	...	..

Joins in a snowflake schema

When using a snowflake schema, dimensions may be partially normalized; requiring multiple joins to relate fact tables to snowflake dimensions. For example, suppose your data warehouse includes a **DimProduct** dimension table from which the product categories have been normalized into a separate **DimCategory** table. A query to aggregate items sold by product category might look similar to the following example:

```
SQL
SELECT  cat.ProductCategory,
        SUM(sales.OrderQuantity) AS ItemsSold
FROM dbo.FactSales AS sales
JOIN dbo.DimProduct AS prod ON sales.ProductKey = prod.ProductKey
JOIN dbo.DimCategory AS cat ON prod.CategoryKey = cat.CategoryKey
GROUP BY cat.ProductCategory
ORDER BY cat.ProductCategory;
```

The results from this query include the number of items sold for each product category:

ProductCategory	ItemsSold
Accessories	28271
Bits and pieces	5368
...	...

*Note: JOIN clauses for **FactSales** and **DimProduct** and for **DimProduct** and **DimCategory** are both required, even though no fields from **DimProduct** are returned by the query.*

Using ranking functions

Another common kind of analytical query is to partition the results based on a dimension attribute and *rank* the results within each partition. For example, you might want to rank stores each year by their sales revenue. To accomplish this goal, you can use Transact-SQL *ranking* functions such as `ROW_NUMBER`, `RANK`, `DENSE_RANK`, and `NTILE`.

These functions enable you to partition the data over categories, each returning a specific value that indicates the relative position of each row within the partition:

- **ROW_NUMBER** returns the ordinal position of the row within the partition. For example, the first row is numbered 1, the second 2, and so on.
- **RANK** returns the ranked position of each row in the ordered results. For example, in a partition of stores ordered by sales volume, the store with the highest sales volume is ranked 1. If multiple stores have the same sales volumes, they'll be ranked the same, and the rank assigned to subsequent stores reflects the number of stores that have higher sales volumes - including ties.
- **DENSE_RANK** ranks rows in a partition the same way as **RANK**, but when multiple rows have the same rank, subsequent rows are ranking positions ignore ties.
- **NTILE** returns the specified percentile in which the row falls. For example, in a partition of stores ordered by sales volume, `NTILE(4)` returns the quartile in which a store's sales volume places it.

For example, consider the following query:

```
SQL
SELECT  ProductCategory,
        ProductName,
        ListPrice,
        ROW_NUMBER() OVER
            (PARTITION BY ProductCategory ORDER BY ListPrice DESC) AS RowNumber,
        RANK() OVER
            (PARTITION BY ProductCategory ORDER BY ListPrice DESC) AS Rank,
        DENSE_RANK() OVER
            (PARTITION BY ProductCategory ORDER BY ListPrice DESC) AS DenseRank,
        NTILE(4) OVER
            (PARTITION BY ProductCategory ORDER BY ListPrice DESC) AS Quartile
FROM dbo.DimProduct
ORDER BY ProductCategory;
```

The query partitions products into groupings based on their categories, and within each category partition, the relative position of each product is determined based on its list price. The results from this query might look similar to the following table:

ProductCategory	ProductName	ListPrice	RowNumber	Rank	DenseRank	Quartile
Accessories	Widget	8.99	1	1	1	1
Accessories	Knicknak	8.49	2	2	2	1
Accessories	Sprocket	5.99	3	3	3	2
Accessories	Doodah	5.99	4	3	3	2
Accessories	Spangle	2.99	5	5	4	3
Accessories	Badabing	0.25	6	6	5	4
Bits and pieces	Flimflam	7.49	1	1	1	1
Bits and pieces	Snickity wotsit	6.99	2	2	2	1
Bits and pieces	Flange	4.25	3	3	3	2
...	...	...	...	...	...	...

Note: The sample results demonstrate the difference between `RANK` and `DENSE_RANK`. Note that in the Accessories category, the Sprocket and Doodah products have the same list price; and are both ranked as the 3rd highest priced product. The next highest priced product has a RANK of 5 (there are four products more expensive than it) and a DENSE_RANK of 4 (there are three higher prices).

https://learn.microsoft.com/en-us/sql/t-sql/functions/ranking-functions-transact-sql?view=azure-sqldw-latest

Retrieving an approximate count

While the purpose of a data warehouse is primarily to support analytical data models and reports for the enterprise; data analysts and data scientists often need to perform some initial data exploration, just to determine the basic scale and distribution of the data.

For example, the following query uses the `COUNT` function to retrieve the number of sales for each year in a hypothetical data warehouse:

```
SQL
SELECT dates.CalendarYear AS CalendarYear,
    COUNT(DISTINCT sales.OrderNumber) AS Orders
FROM FactSales AS sales
JOIN DimDate AS dates ON sales.OrderDateKey = dates.DateKey
GROUP BY dates.CalendarYear
ORDER BY CalendarYear;
```

The results of this query might look similar to the following table:

CalendarYear	Orders
2019	239870
2020	284741
2021	309272
...	...

The volume of data in a data warehouse can mean that even simple queries to count the number of records that meet specified criteria can take a considerable time to run. In many cases, a precise count isn't required - an approximate estimate will suffice. In such cases, you can use the `APPROX_COUNT_DISTINCT` function as shown in the following example:

```
SQL
SELECT dates.CalendarYear AS CalendarYear,
    APPROX_COUNT_DISTINCT(sales.OrderNumber) AS ApproxOrders
FROM FactSales AS sales
JOIN DimDate AS dates ON sales.OrderDateKey = dates.DateKey
GROUP BY dates.CalendarYear
ORDER BY CalendarYear;
```

The `APPROX_COUNT_DISTINCT` function uses a *HyperLogLog* algorithm to retrieve an approximate count. The result is guaranteed to have a maximum error rate of 2% with

97% probability, so the results of this query with the same hypothetical data as before might look similar to the following table:

CalendarYear	ApproxOrders
2019	235552
2020	290436
2021	304633
...	...

The counts are less accurate, but still sufficient for an approximate comparison of yearly sales. With a large volume of data, the query using the `APPROX_COUNT_DISTINCT` function completes more quickly, and the reduced accuracy may be an acceptable trade-off during basic data exploration.

https://learn.microsoft.com/en-us/sql/t-sql/functions/approx-count-distinct-transact-sql?view=azure-sqldw-latest

Question 90: **Incorrect**
A data warehouse that is built on a Massively Parallel Processing (MPP) system is built for processing and analyzing large datasets. As such they perform well with larger batch type loads and updates that can be distributed across the compute nodes and storage.

Which of the following is the best approach if singleton or smaller transaction batch loads must be added to an MPP data warehouse?

- **All the approaches are equally valid.**
- **None of the listed options.**

Develop a process that writes the outputs of an INSERT statement to a to the target file automatically, avoiding the need to do the INSERT manually.

- ○

Develop two processes: one that writes the outputs of an INSERT statement to a file, and then another process to periodically load this file.

(Correct)

- ◉

Manually create an append file with a trigger that once the contents of the manually created file reach a predetermined size, an automation process will be triggered to append the data to the data warehouse.

(Incorrect)

Explanation
A data warehouse that is built on a Massively Parallel Processing (MPP) system is built for processing and analyzing large datasets. As such they perform well with larger batch type loads and updates that can be distributed across the compute nodes and storage.

Singleton or smaller transaction batch loads should be grouped into larger batches to optimize the Synapse SQL Pools processing capabilities. To be clear, A one-off load to a small table with an INSERT statement may be the best approach, if it is a one-off.

However, if you need to load thousands or millions of rows throughout the day, then singleton `INSERT`s aren't optimal against an MPP system. One way to solve this issue is to develop one process that writes the outputs of an `INSERT` statement to a file, and then another process to periodically load this file to take advantage of the parallelism that Azure Synapse Analytics.

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

Question 91: **Incorrect**
Azure HDInsight is a low-cost cloud solution which provides technologies to help you ingest, process, and analyze big data.

Which of the following are supported in the HDInsight solution? (Select all that apply)

- ☐

Interactive Query

(Correct)

- ☐

Spark

(Correct)

- ☐

Storm

(Correct)

- ☐

Kafka

(Correct)

- ☐

Repos

- ☑

Sphere

(Incorrect)

- ☐

Hbase

(Correct)

- ☐

Sentinel

- ☐

PowerShell

- ☐

Hadoop

(Correct)

Explanation
Azure HDInsight provides technologies to help you ingest, process, and analyze big data. It supports batch processing, data warehousing, IoT, and data science.

Key features

HDInsight is a low-cost cloud solution. HDInsight supports the latest open-source projects from the Apache Hadoop and Spark ecosystems. It includes Apache Hadoop, Spark, Kafka, HBase, Storm, and Interactive Query.

• **Hadoop** includes Apache Hive, HBase, Spark, and Kafka. Hadoop stores data in a file system (HDFS). Spark stores data in memory. This difference in storage makes Spark about 100 times faster.

• **HBase** is a NoSQL database built on Hadoop. It's commonly used for search engines. HBase offers automatic failover.

• **Storm** is a distributed real-time streamlining analytics solution.

• **Kafka** is an open-source platform that's used to compose data pipelines. It offers message queue functionality, which allows users to publish or subscribe to real-time data streams.

Ingesting data

As a data engineer, use Hive to run ETL operations on the data you're ingesting. Or orchestrate Hive queries in Azure Data Factory.

https://azure.microsoft.com/en-us/services/hdinsight/#features

Question 92: **Incorrect**
To create a relational data warehouse in Azure Synapse Analytics, you must create a dedicated SQL Pool. The simplest way to do this in an existing Azure Synapse Analytics workspace.

Which of the following shows the correct order should you load tables in the data warehouse?

- ○

  1. Fact tables

  2. Dimension tables

  3. Staging tables

- ○

  1. Dimension tables

  2. Staging tables

  3. Fact tables

- ◉

  1. Staging tables

  2. Fact tables

  3. Dimension tables

  (Incorrect)

- ○

  1. Staging tables

2. Dimension tables

3. Fact tables

(Correct)

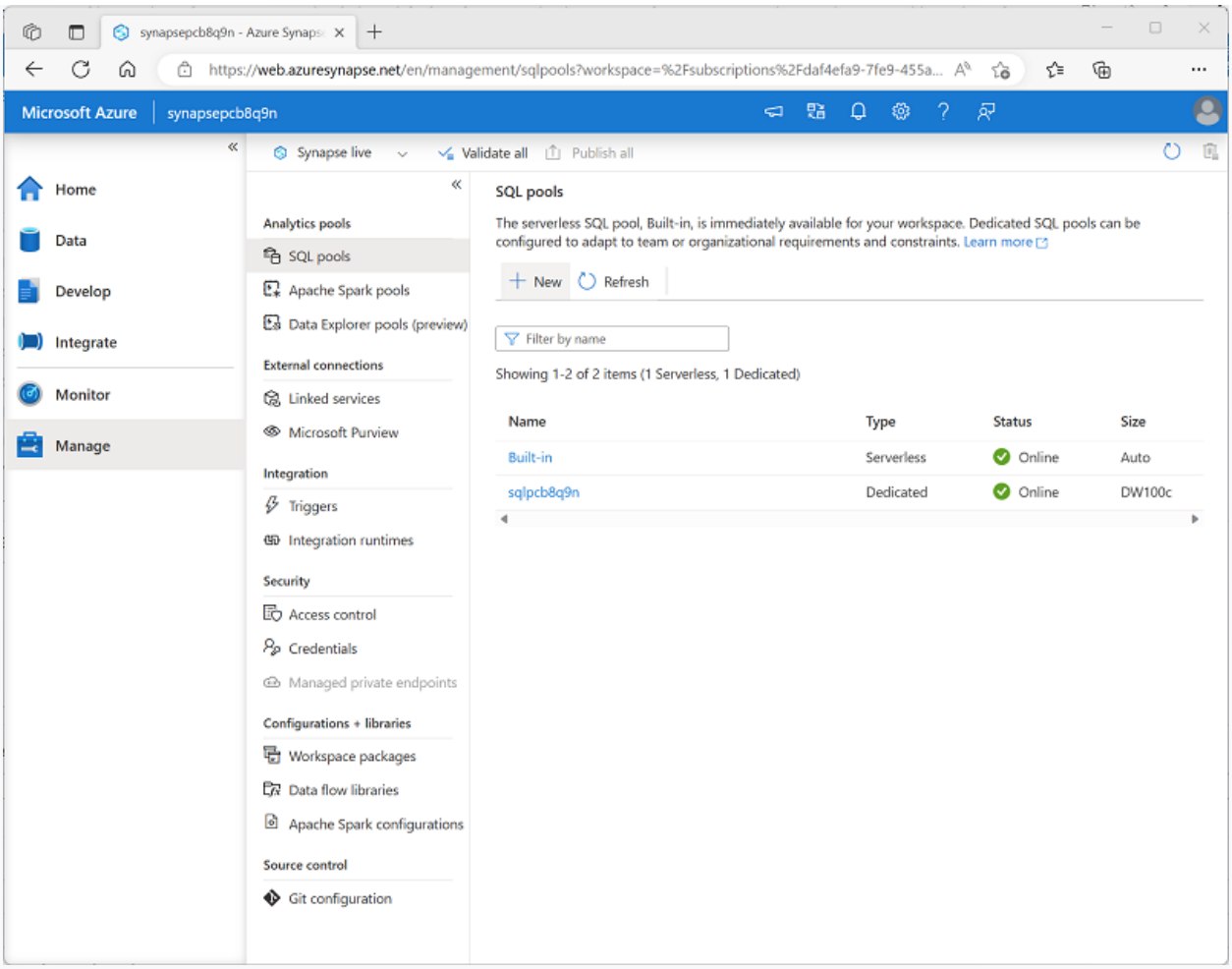
Explanation

- *The correct order of operations is stage, populate dimensions to transform keys and SCDs, then load facts with numeric values.*
- Fact tables should be loaded last with the appropriate surrogate keys from dimensions.
- The Dimension tables, then Staging tables, then Fact tables approach could cause corruption if there were changes to a dimension.

Create data warehouse tables

Creating a dedicated SQL pool

To create a relational data warehouse in Azure Synapse Analytics, you must create a dedicated SQL Pool. The simplest way to do this in an existing Azure Synapse Analytics workspace is to use the **Manage** page in Azure Synapse Studio, as shown here:



When provisioning a dedicated SQL pool, you can specify the following configuration settings:

- A unique name for the dedicated SQL pool.
- A performance level for the SQL pool, which can range from *DW100c* to *DW30000c* and which determines the cost per hour for the pool when it's running.
- Whether to start with an empty pool or restore an existing database from a backup.
- The *collation* of the SQL pool, which determines sort order and string comparison rules for the database. (*You can't change the collation after creation*).

After creating a dedicated SQL pool, you can control its running state in the **Manage** page of Synapse Studio; pausing it when not required to prevent unnecessary costs.

When the pool is running, you can explore it on the **Data** page, and create SQL scripts to run in it.

Considerations for creating tables

To create tables in the dedicated SQL pool, you use the `CREATE TABLE` (or sometimes the `CREATE EXTERNAL TABLE`) Transact-SQL statement. The specific options used in the statement depend on the type of table you're creating, which can include:

- Staging tables
- Dimension tables
- Fact tables

Note: The data warehouse is composed of fact and dimension tables as discussed previously. Staging tables are often used as part of the data warehousing loading process to ingest data from source systems.

When designing a star schema model for small or medium sized datasets you can use your preferred database, such as Azure SQL. For larger data sets you may benefit from implementing your data warehouse in Azure Synapse Analytics instead of SQL Server. It's important to understand some key differences when creating tables in Synapse Analytics.

Data integrity constraints

Dedicated SQL pools in Synapse Analytics don't support *foreign key* and *unique* constraints as found in other relational database systems like SQL Server. This means that jobs used to load data must maintain uniqueness and referential integrity for keys, without relying on the table definitions in the database to do so.

https://learn.microsoft.com/en-us/azure/synapse-analytics/sql-data-warehouse/sql-data-warehouse-table-constraints

Indexes

While Synapse Analytics dedicated SQL pools support *clustered* indexes as found in SQL Server, the default index type is *clustered columnstore*. This index type offers a significant performance advantage when querying large quantities of data in a typical data warehouse schema and should be used where possible. However, some tables may include data types that can't be included in a clustered columnstore index (for example, VARBINARY(MAX)), in which case a clustered index can be used instead.

https://learn.microsoft.com/en-us/azure/synapse-analytics/sql-data-warehouse/sql-data-warehouse-tables-index

Distribution

Azure Synapse Analytics dedicated SQL pools use a massively parallel processing (MPP) architecture, as opposed to the symmetric multiprocessing (SMP) architecture used in most OLTP database systems. In an MPP system, the data in a table is distributed for processing across a pool of nodes. Synapse Analytics supports the following kinds of distribution:

- **Hash**: A deterministic hash value is calculated for the specified column and used to assign the row to a compute node.
- **Round-robin**: Rows are distributed evenly across all compute nodes.
- **Replicated**: A copy of the table is stored on each compute node.

The table type often determines which option to choose for distributing the table.

Table type	Recommended distribution option
Dimension	Use replicated distribution for smaller tables to avoid data shuffling when joining to distributed fact tables. If tables are too large to store on each compute node, use hash distribution.
Fact	Use hash distribution with clustered columnstore index to distribute fact tables across compute nodes.
Staging	Use round-robin distribution for staging tables to evenly distribute data across compute nodes.

https://learn.microsoft.com/en-us/azure/synapse-analytics/sql-data-warehouse/sql-data-warehouse-tables-distribute

Creating Dimension Tables

When you create a dimension table, ensure that the table definition includes surrogate and alternate keys as well as columns for the attributes of the dimension that you want to use to group aggregations. It's often easiest to use an `IDENTITY` column to auto-generate an incrementing surrogate key (otherwise you need to generate unique keys every time you load data). The following example shows a `CREATE TABLE` statement for a hypothetical **DimCustomer** dimension table.

```
SQL
CREATE TABLE dbo.DimCustomer
(
    CustomerKey INT IDENTITY NOT NULL,
    CustomerAlternateKey NVARCHAR(15) NULL,
    CustomerName NVARCHAR(80) NOT NULL,
    EmailAddress NVARCHAR(50) NULL,
    Phone NVARCHAR(25) NULL,
    StreetAddress NVARCHAR(100),
    City NVARCHAR(20),
    PostalCode NVARCHAR(10),
    CountryRegion NVARCHAR(20)
)
WITH
(
    DISTRIBUTION = REPLICATE,
    CLUSTERED COLUMNSTORE INDEX
);
```

*Note: If desired, you can create a specific schema as a namespace for your tables. In this example, the default **dbo** schema is used.*

*If you intend to use a snowflake schema in which dimension tables are related to one another, you should include the key for the parent dimension in the definition of the child dimension table. For example, the following SQL code could be used to move the geographical address details from the **DimCustomer** table to a separate **DimGeography** dimension table:*

```
SQL
CREATE TABLE dbo.DimGeography
(
    GeographyKey INT IDENTITY NOT NULL,
    GeographyAlternateKey NVARCHAR(10) NULL,
    StreetAddress NVARCHAR(100),
    City NVARCHAR(20),
```

```
    PostalCode NVARCHAR(10),
    CountryRegion NVARCHAR(20)
)
WITH
(
    DISTRIBUTION = REPLICATE,
    CLUSTERED COLUMNSTORE INDEX
);

CREATE TABLE dbo.DimCustomer
(
    CustomerKey INT IDENTITY NOT NULL,
    CustomerAlternateKey NVARCHAR(15) NULL,
    GeographyKey INT NULL,
    CustomerName NVARCHAR(80) NOT NULL,
    EmailAddress NVARCHAR(50) NULL,
    Phone NVARCHAR(25) NULL
)
WITH
(
    DISTRIBUTION = REPLICATE,
    CLUSTERED COLUMNSTORE INDEX
);
```

Time Dimension tables

Most data warehouses include a *time* dimension table that enables you to aggregate data by multiple hierarchical levels of time interval. For example, the following example creates a **DimDate** table with attributes that relate to specific dates.

```
SQL
CREATE TABLE dbo.DimDate
(
    DateKey INT NOT NULL,
    DateAltKey DATETIME NOT NULL,
    DayOfMonth INT NOT NULL,
    DayOfWeek INT NOT NULL,
    DayName NVARCHAR(15) NOT NULL,
    MonthOfYear INT NOT NULL,
    MonthName NVARCHAR(15) NOT NULL,
    CalendarQuarter INT  NOT NULL,
    CalendarYear INT NOT NULL,
    FiscalQuarter INT NOT NULL,
    FiscalYear INT NOT NULL
)
WITH
(
    DISTRIBUTION = REPLICATE,
    CLUSTERED COLUMNSTORE INDEX
);
```

Tip: A common pattern when creating a dimension table for dates is to use the numeric date in DDMMYYYY or YYYYMMDD format as an integer surrogate key, and the date as a `DATE` or `DATETIME` datatype as the alternate key.

Creating Fact tables

Fact tables include the keys for each dimension to which they're related, and the attributes and numeric measures for specific events or observations that you want to analyze.

The following code example creates a hypothetical fact table named **FactSales** that is related to multiple dimensions through key columns (date, customer, product, and store)

```
SQL
CREATE TABLE dbo.FactSales
(
    OrderDateKey INT NOT NULL,
    CustomerKey INT NOT NULL,
    ProductKey INT NOT NULL,
    StoreKey INT NOT NULL,
    OrderNumber NVARCHAR(10) NOT NULL,
    OrderLineItem INT NOT NULL,
    OrderQuantity SMALLINT NOT NULL,
    UnitPrice DECIMAL NOT NULL,
    Discount DECIMAL NOT NULL,
    Tax DECIMAL NOT NULL,
    SalesAmount DECIMAL NOT NULL
)
WITH
(
    DISTRIBUTION = HASH(OrderNumber),
    CLUSTERED COLUMNSTORE INDEX
);
```

Creating Staging tables

Staging tables are used as temporary storage for data as it's being loaded into the data warehouse. A typical pattern is to structure the table to make it as efficient as possible to ingest the data from its external source (often files in a data lake) into the relational

database, and then use SQL statements to load the data from the staging tables into the dimension and fact tables.

The following code example creates a staging table for product data that will ultimately be loaded into a dimension table:

```
SQL
CREATE TABLE dbo.StageProduct
(
    ProductID NVARCHAR(10) NOT NULL,
    ProductName NVARCHAR(200) NOT NULL,
    ProductCategory NVARCHAR(200) NOT NULL,
    Color NVARCHAR(10),
    Size NVARCHAR(10),
    ListPrice DECIMAL NOT NULL,
    Discontinued BIT NOT NULL
)
WITH
(
    DISTRIBUTION = ROUND_ROBIN,
    CLUSTERED COLUMNSTORE INDEX
);
```

Using external tables

In some cases, if the data to be loaded is in files with an appropriate structure, it can be more effective to create external tables that reference the file location. This way, the data can be read directly from the source files instead of being loaded into the relational store. The following example, shows how to create an external table that references files in the data lake associated with the Synapse workspace:

```
SQL
-- External data source links to data lake location
CREATE EXTERNAL DATA SOURCE StagedFiles
WITH (
    LOCATION = 'https://mydatalake.blob.core.windows.net/data/stagedfiles/'
);
GO

-- External format specifies file format
CREATE EXTERNAL FILE FORMAT ParquetFormat
WITH (
    FORMAT_TYPE = PARQUET,
    DATA_COMPRESSION = 'org.apache.hadoop.io.compress.SnappyCodec'
);
```

```
GO

-- External table references files in external data source
CREATE EXTERNAL TABLE dbo.ExternalStageProduct
(
    ProductID NVARCHAR(10) NOT NULL,
    ProductName NVARCHAR(200) NOT NULL,
    ProductCategory NVARCHAR(200) NOT NULL,
    Color NVARCHAR(10),
    Size NVARCHAR(10),
    ListPrice DECIMAL NOT NULL,
    Discontinued BIT NOT NULL
)
WITH
(
    DATA_SOURCE = StagedFiles,
    LOCATION = 'products/*.parquet',
    FILE_FORMAT = ParquetFormat
);
GO
```

https://learn.microsoft.com/en-us/azure/synapse-analytics/sql/develop-tables-external-tables

Question 93: **Incorrect**

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

Azure Synapse Analytics can work by acting as the one stop shop to meet all of your analytical needs in an integrated environment.

[?] leverages the capabilities of Azure Data Factory and is the cloud-based ETL and data integration service that allows you to create data-driven workflows for orchestrating data movement and transforming data at scale. Using [?], you can create and schedule data-driven workflows that can ingest data from disparate data stores.

- **Azure Synapse Link**
- **Azure Cosmos DB**
- **Azure Synapse Pipelines**

(Correct)

- **Apache Spark for Azure Synapse**

  (Incorrect)

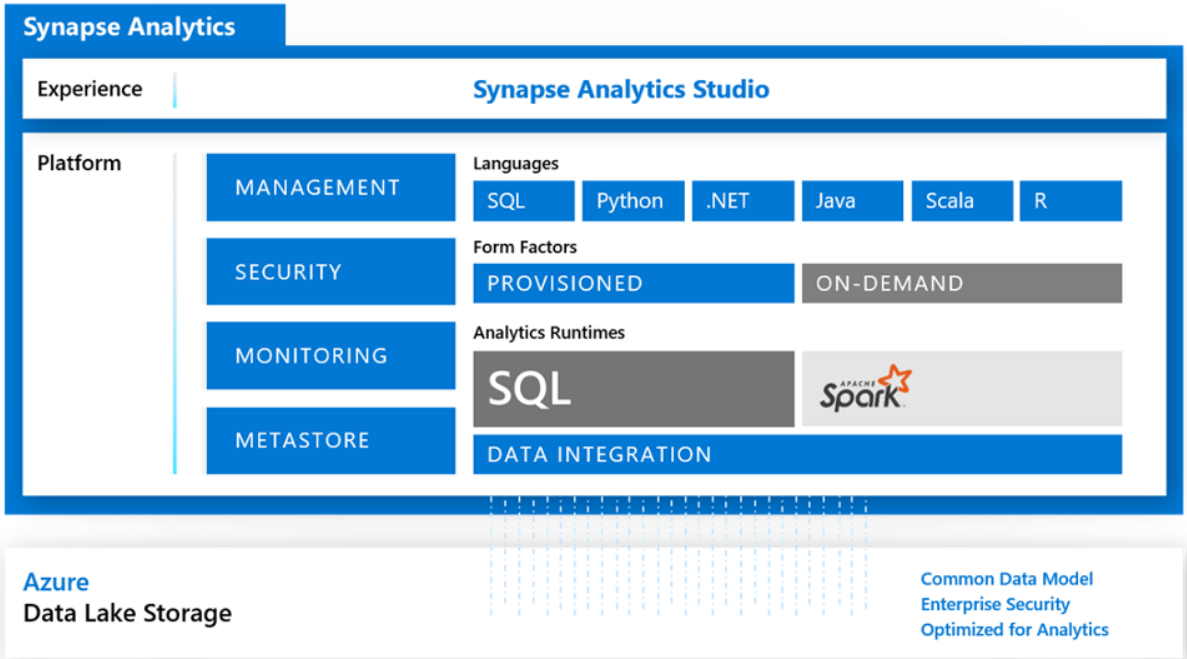
- **Azure Synapse SQL**

Explanation

Azure Synapse Analytics can work by acting as the one stop shop to meet all of your analytical needs in an integrated environment. It does this by providing the following capabilities:

Analytics capabilities offered through Azure Synapse SQL through either dedicated SQL pools or SQL Serverless pools

Azure Synapse SQL is a distributed query system that enables you to implement data warehousing and data virtualization scenarios using standard T-SQL experiences familiar to data engineers. Synapse SQL offers both serverless and dedicated resource models to work with both descriptive and diagnostic analytical scenarios. For predictable performance and cost, create dedicated SQL pools to reserve processing power for data stored in SQL tables. For unplanned or ad-hoc workloads, use the always-available, serverless SQL endpoint.



Apache Spark pool with full support for Scala, Python, SparkSQL, and C#

You can develop big data engineering and machine learning solutions using Apache Spark for Azure Synapse. You can take advantage of the big data computation engine to deal with complex compute transformations that would take too long in a data warehouse. For machine learning workloads, you can use SparkML algorithms and AzureML integration for Apache Spark 2.4 with built-in support for Linux Foundation Delta Lake. There is a simple model for provisioning and scaling the Spark clusters to meet your compute needs, regardless of the operations that you are performing on the data.

Data integration to integrate your data with Azure Synapse Pipelines

Azure Synapse Pipelines leverages the capabilities of Azure Data Factory and is the cloud-based ETL and data integration service that allows you to create data-driven workflows for orchestrating data movement and transforming data at scale. Using Azure Synapse Pipelines, you can create and schedule data-driven workflows (called pipelines) that can ingest data from disparate data stores. You can build complex ETL processes that transform data visually with data flows or by using compute services such as Azure HDInsight Hadoop, or Azure Databricks.

Perform operational analytics with near real-time hybrid transactional and analytical processing with Azure Synapse Link

Azure Synapse Analytics enables you to reach out to operational data using Azure Synapse Link, and is achieved without impacting the performance of the transactional data store. For this to happen, you have to enable the feature within both Azure Synapse Analytics, and within the data store to which Azure Synapse Analytics will connect, such as Azure Cosmos DB. In the case of Azure Cosmos DB, this will create an analytical data store. As data changes in the transactional system, the changed data is fed to the analytical store in a Column store format from which Azure Synapse Link can query with no disruption to the source system.

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

Question 94: **Incorrect**

Scenario: You have been contracted by Wayne Enterprises, a company owned by Bruce Wayne with market value of over twenty seven million dollars. Bruce founded Wayne Enterprises shortly after he created the Wayne Foundation and he became the president and chairman of the company.

Bruce has come to you because his IT team needs advice on the use of Azure SQL Database to support a mission-critical application.

Required:

• The application must be highly available

• No performance loss during maintenance cycles

Which of the following applications should you recommend to Bruce and his team to adopt?

- ☐

  Premium service tier

  (Correct)

- ☐

  Virtual machine Scale Sets

- ☑

  Zone-redundant configuration

  (Correct)

- ☐

  SQL Data Sync

- ☐

  Always On availability groups

  (Correct)

- ☐

  Ultra service tier

Explanation

Premium service tier: Premium/business critical service tier model that is based on a cluster of database engine processes. This architectural model relies on a fact that there is always a quorum of available database engine nodes and has minimal performance impact on your workload even during maintenance activities.

Always On availability groups: In the premium model, Azure SQL database integrates compute and storage on the single node. High availability in this architectural model is achieved by replication of compute (SQL Server Database Engine process) and storage (locally attached SSD) deployed in 4-node cluster, using technology similar to SQL

Zone redundant configuration: By default, the quorum-set replicas for the local storage configurations are created in the same datacentre. With the introduction of Azure Availability Zones, you have the ability to place the different replicas in the quorum-sets to different availability zones in the same region. To eliminate a single point of failure, the control ring is also duplicated across multiple zones as three gateway rings (GW).

The goal of the high availability architecture in Azure SQL Database and SQL Managed Instance is to guarantee that your database is up and running minimum of 99.99% of time (For more information regarding specific SLA for different tiers, Please refer SLA for Azure SQL Database and SQL Managed Instance), without worrying about the impact of maintenance operations and outages. Azure automatically handles critical servicing tasks, such as patching, backups, Windows and Azure SQL upgrades, as well as unplanned events such as underlying hardware, software, or network failures. When the underlying database in Azure SQL Database is patched or fails over, the downtime is not noticeable if you employ retry logic in your app. SQL Database and SQL Managed Instance can quickly recover even in the most critical circumstances ensuring that your data is always available.

The high availability solution is designed to ensure that committed data is never lost due to failures, that maintenance operations do not affect your workload, and that the

database will not be a single point of failure in your software architecture. There are no maintenance windows or downtimes that should require you to stop the workload while the database is upgraded or maintained.

There are two high availability architectural models:

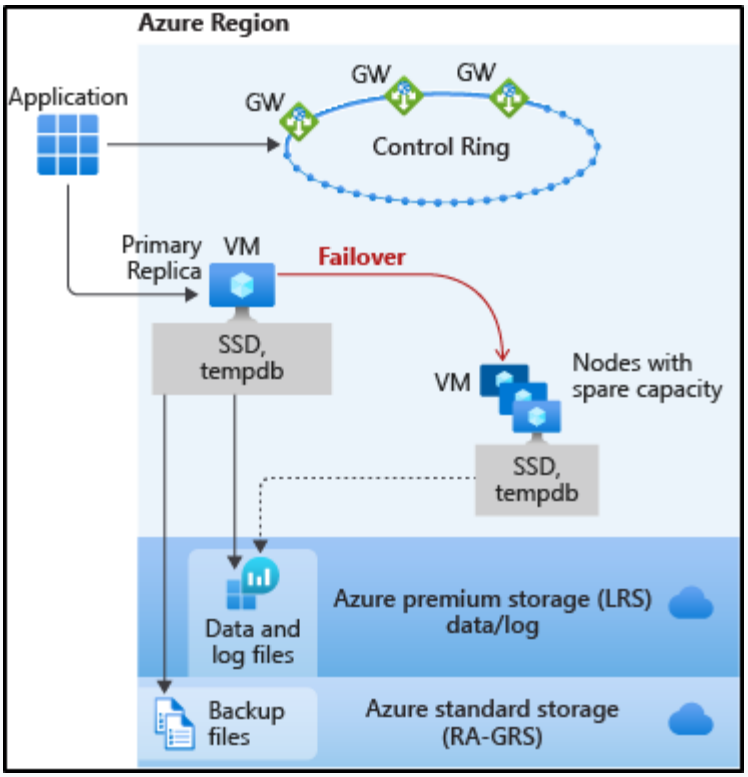
Standard availability model that is based on a separation of compute and storage. It relies on high availability and reliability of the remote storage tier. This architecture targets budget-oriented business applications that can tolerate some performance degradation during maintenance activities.

Premium availability model that is based on a cluster of database engine processes. It relies on the fact that there is always a quorum of available database engine nodes. This architecture targets mission critical applications with high IO performance, high transaction rate and guarantees minimal performance impact to your workload during maintenance activities.

SQL Database and SQL Managed Instance both run on the latest stable version of the SQL Server database engine and Windows operating system, and most users would not notice that upgrades are performed continuously.

Basic, Standard, and General Purpose service tier locally redundant availability

The Basic, Standard, and General Purpose service tiers leverage the standard availability architecture for both serverless and provisioned compute. The following figure shows four different nodes with the separated compute and storage layers.



The standard availability model includes two layers:

A stateless compute layer that runs the `sqlservr.exe` process and contains only transient and cached data, such as TempDB, model databases on the attached SSD, and plan cache, buffer pool, and columnstore pool in memory. This stateless node is operated by Azure Service Fabric that initializes `sqlservr.exe`, controls health of the node, and performs failover to another node if necessary.

A stateful data layer with the database files (.mdf/.ldf) that are stored in Azure Blob storage. Azure blob storage has built-in data availability and redundancy feature. It guarantees that every record in the log file or page in the data file will be preserved even if `sqlservr.exe` process crashes.

Whenever the database engine or the operating system is upgraded, or a failure is detected, Azure Service Fabric will move the stateless `sqlservr.exe` process to another stateless compute node with sufficient free capacity. Data in Azure Blob storage is not affected by the move, and the data/log files are attached to the newly initialized `sqlservr.exe` process. This process guarantees 99.99% availability, but a heavy workload may experience some performance degradation during the transition since the new `sqlservr.exe` process starts with cold cache.

General Purpose service tier zone redundant availability

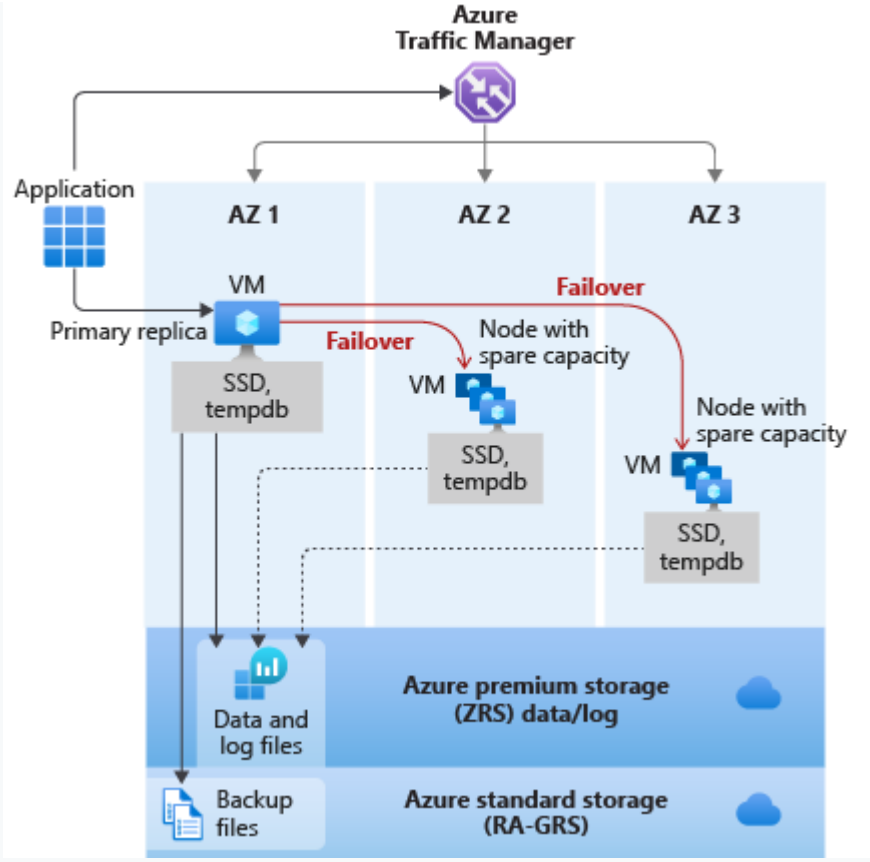
Zone redundant configuration for the general purpose service tier is offered for both serverless and provisioned compute. This configuration utilizes Azure Availability Zones to replicate databases across multiple physical locations within an Azure region. By selecting zone redundancy, you can make your new and existing serverlesss and provisioned general purpose single databases and elastic pools resilient to a much larger set of failures, including catastrophic datacenter outages, without any changes of the application logic.

Zone redundant configuration for the general purpose tier has two layers:

A stateful data layer with the database files (.mdf/.ldf) that are stored in ZRS(zone-redundant storage). Using ZRS the data and log files are synchronously copied across three physically-isolated Azure availability zones.

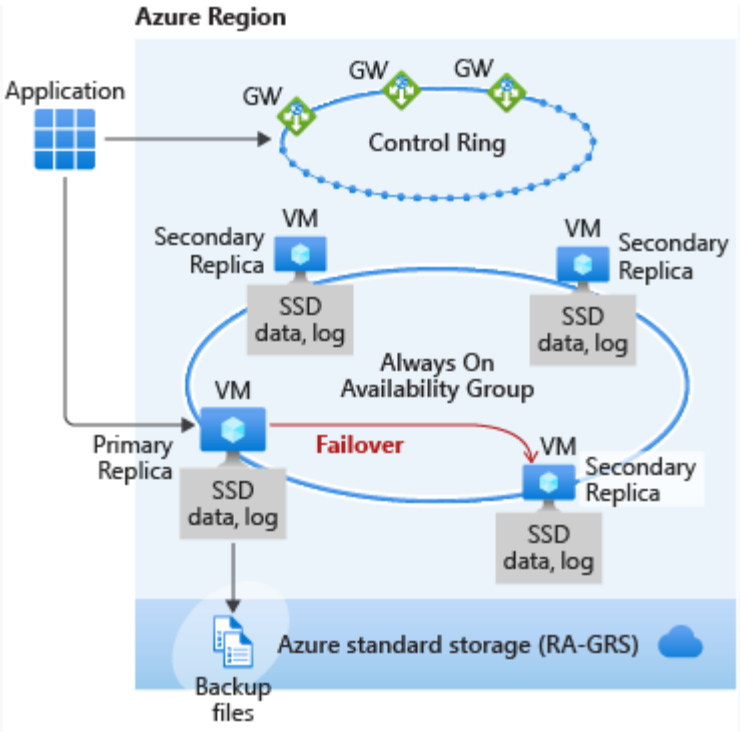
A stateless compute layer that runs the sqlservr.exe process and contains only transient and cached data, such as TempDB, model databases on the attached SSD, and plan cache, buffer pool, and columnstore pool in memory. This stateless node is operated by Azure Service Fabric that initializes sqlservr.exe, controls health of the node, and performs failover to another node if necessary. For zone redundant serverless and provisioned general purpose databases, nodes with spare capacity are readily available in other Availability Zones for failover.

The zone redundant version of the high availability architecture for the general purpose service tier is illustrated by the following diagram:



Premium and Business Critical service tier locally redundant availability

Premium and Business Critical service tiers leverage the Premium availability model, which integrates compute resources (`sqlservr.exe` process) and storage (locally attached SSD) on a single node. High availability is achieved by replicating both compute and storage to additional nodes creating a three to four-node cluster.



The underlying database files (.mdf/.ldf) are placed on the attached SSD storage to provide very low latency IO to your workload. High availability is implemented using a technology similar to SQL Server Always On availability groups. The cluster includes a single primary replica that is accessible for read-write customer workloads, and up to three secondary replicas (compute and storage) containing copies of data. The primary node constantly pushes changes to the secondary nodes in order and ensures that the data is synchronized to at least one secondary replica before committing each transaction. This process guarantees that if the primary node crashes for any reason, there is always a fully synchronized node to fail over to. The failover is initiated by the Azure Service Fabric. Once the secondary replica becomes the new primary node, another secondary replica is created to ensure the cluster has enough nodes (quorum set). Once failover is complete, Azure SQL connections are automatically redirected to the new primary node.

As an extra benefit, the premium availability model includes the ability to redirect read-only Azure SQL connections to one of the secondary replicas. This feature is called Read Scale-Out. It provides 100% additional compute capacity at no extra charge to off-load read-only operations, such as analytical workloads, from the primary replica.

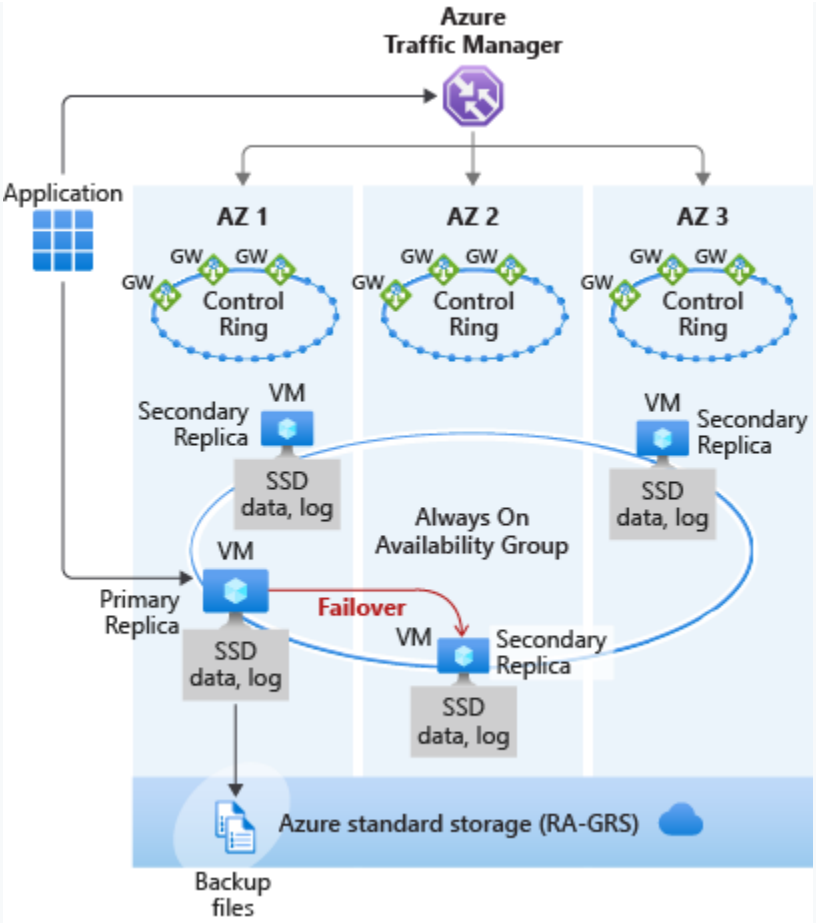
Premium and Business Critical service tier zone redundant availability

By default, the cluster of nodes for the premium availability model is created in the same datacenter. With the introduction of Azure Availability Zones, SQL Database can

place different replicas of the Business Critical database to different availability zones in the same region. To eliminate a single point of failure, the control ring is also duplicated across multiple zones as three gateway rings (GW). The routing to a specific gateway ring is controlled by Azure Traffic Manager (ATM). Because the zone redundant configuration in the Premium or Business Critical service tiers does not create additional database redundancy, you can enable it at no extra cost. By selecting a zone redundant configuration, you can make your Premium or Business Critical databases resilient to a much larger set of failures, including catastrophic datacenter outages, without any changes to the application logic. You can also convert any existing Premium or Business Critical databases or pools to the zone redundant configuration.

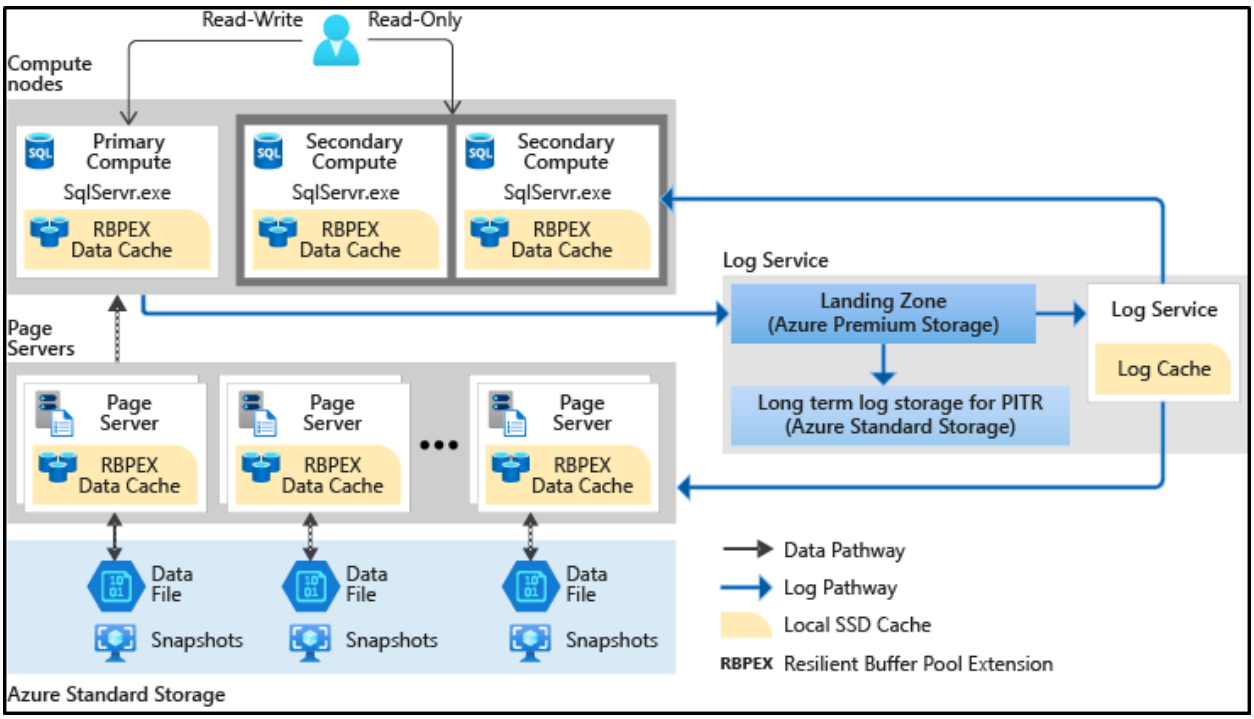
Because the zone redundant databases have replicas in different datacenters with some distance between them, the increased network latency may increase the commit time and thus impact the performance of some OLTP workloads. You can always return to the single-zone configuration by disabling the zone redundancy setting. This process is an online operation similar to the regular service tier upgrade. At the end of the process, the database or pool is migrated from a zone redundant ring to a single zone ring or vice versa.

The zone redundant version of the high availability architecture is illustrated by the following diagram:



Hyperscale service tier availability

The Hyperscale service tier architecture is described in [Distributed functions architecture] and is only currently available for SQL Database, not SQL Managed Instance.



The availability model in Hyperscale includes four layers:

A stateless compute layer that runs the `sqlservr.exe` processes and contains only transient and cached data, such as non-covering RBPEX cache, TempDB, model database, etc. on the attached SSD, and plan cache, buffer pool, and columnstore pool in memory. This stateless layer includes the primary compute replica and optionally a number of secondary compute replicas that can serve as failover targets.

A stateless storage layer formed by page servers. This layer is the distributed storage engine for the `sqlservr.exe` processes running on the compute replicas. Each page server contains only transient and cached data, such as covering RBPEX cache on the attached SSD, and data pages cached in memory. Each page server has a paired page server in an active-active configuration to provide load balancing, redundancy, and high availability.

A stateful transaction log storage layer formed by the compute node running the Log service process, the transaction log landing zone, and transaction log long term storage. Landing zone and long term storage use Azure Storage, which provides availability and redundancy for transaction log, ensuring data durability for committed transactions.

A stateful data storage layer with the database files (.mdf/.ndf) that are stored in Azure Storage and are updated by page servers. This layer uses data availability

and redundancy features of Azure Storage. It guarantees that every page in a data file will be preserved even if processes in other layers of Hyperscale architecture crash, or if compute nodes fail.

Compute nodes in all Hyperscale layers run on Azure Service Fabric, which controls health of each node and performs failovers to available healthy nodes as necessary.

For more information on high availability in Hyperscale, see Database High Availability in Hyperscale.

Accelerated Database Recovery (ADR)

Accelerated Database Recovery (ADR) is a new database engine feature that greatly improves database availability, especially in the presence of long running transactions. ADR is currently available for Azure SQL Database, Azure SQL Managed Instance, and Azure Synapse Analytics.

Testing application fault resiliency

High availability is a fundamental part of the SQL Database and SQL Managed Instance platform that works transparently for your database application. However, we recognize that you may want to test how the automatic failover operations initiated during planned or unplanned events would impact an application before you deploy it to production. You can manually trigger a failover by calling a special API to restart a database, an elastic pool, or a managed instance. In the case of a zone redundant serverless or provisioned General Purpose database or elastic pool, the API call would result in redirecting client connections to the new primary in an Availability Zone different from the Availability Zone of the old primary. So in addition to testing how failover impacts existing database sessions, you can also verify if it changes the end-to-end performance due to changes in network latency. Because the restart operation is intrusive and a large number of them could stress the platform, only one failover call is allowed every 15 minutes for each database, elastic pool, or managed instance.

Azure SQL Database and Azure SQL Managed Instance feature a built-in high availability solution, that is deeply integrated with the Azure platform. It is dependent on Service Fabric for failure detection and recovery, on Azure Blob storage for data protection, and on Availability Zones for higher fault tolerance (as mentioned earlier in document not applicable to Azure SQL Managed Instance yet). In addition, SQL Database and SQL Managed Instance leverage the Always On availability group technology from the SQL Server instance for replication and failover. The combination of these technologies enables applications to fully realize the benefits of a mixed storage model and support the most demanding SLAs.

https://docs.microsoft.com/en-us/azure/sql-database/sql-database-high-availability

Question 95: **Incorrect**
In Azure Synapse Studio, use the Monitor hub is where you access which of the following? (Select six)

- ☐

  Data flows

- ☐

  Provisioned SQL pool databases

- ☐

  SQL requests

  (Correct)

- ☐

  External data sources

- ☐

  Data flow debug

  (Correct)

- ☐

  SQL serverless databases

- ☑

  Trigger runs

  (Correct)

- ☐

  Pipeline runs

  (Correct)

- ☐

  Power BI

- ☐

  Notebooks

- ☐

  Integration runtimes

  (Correct)

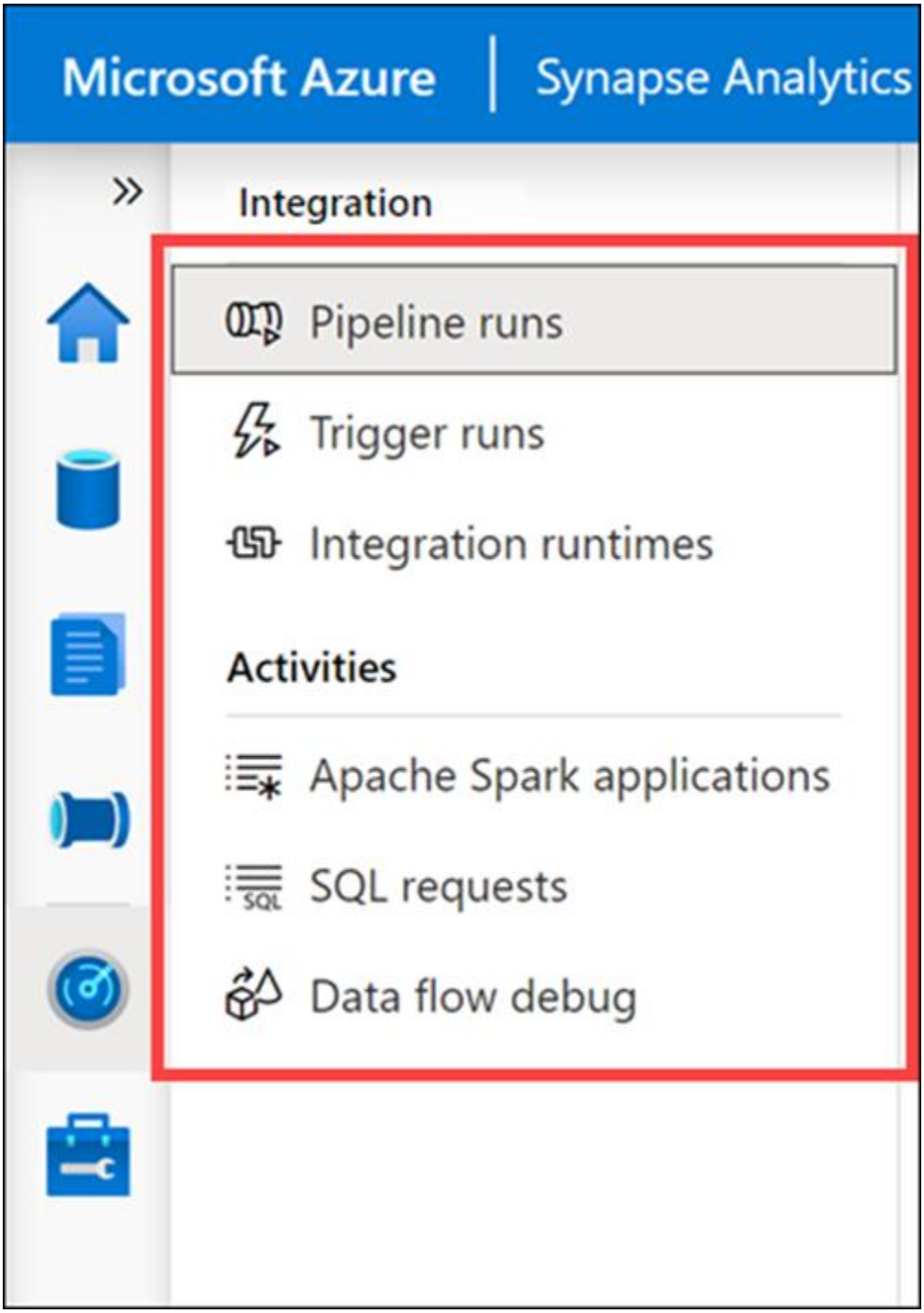
- ☐

  Apache Spark jobs

  (Correct)

Explanation
In Azure Synapse Studio, use the Monitor hub to view pipeline and trigger runs, view the status of the various integration runtimes that are running, view Apache Spark jobs, SQL requests, and data flow debug activities.

The Monitor hub is your first stop for debugging issues and gaining insight on resource usage. You can see a history of all the activities taking place in the workspace and which ones are active now.



https://techcommunity.microsoft.com/t5/azure-synapse-analytics/explore-the-monitor-hub-in-synapse-studio-to-keep-track-of-all/ba-p/1987405

Question 96: **Incorrect**

Activities within Azure Data Factory define the actions that will be performed on the data and there are three categories including:

• Data movement activities

• Data transformation activities

• Control activities

When using JSON notation, the activities section can have one or more activity defined within it.

They have the following top-level structure:

```
JSON
{
"name": "Execution Activity Name",
"description": "description",
"type": "<ActivityType>",
"typeProperties":
{
},
"linkedServiceName": "MyLinkedService",
"policy":
{
},
"dependsOn":
{
}
}
```

Which of the JSON properties are required for HDInsight? (Select all that apply)

- ☑ **description**

  (Correct)

- ☐ **linkedServiceName**

  (Correct)

- ☐ **policy**

- ☐ **type**

(Correct)

- ☐

dependsOn

- ☐

name

(Correct)

- ☐

typeProperties

Explanation

Activities within Azure Data Factory define the actions that will be performed on the data and there are three categories including:

• Data movement activities

• Data transformation activities

• Control activities

Activities and pipelines

Defining activities

When using JSON notation, the activities section can have one or more activities defined within it. There are two main types of activities: Execution and Control Activities. Execution (also known as Compute) activities include data movement and data transformation activities. They have the following top-level structure:

```
JSON
{
"name": "Execution Activity Name",
"description": "description",
"type": "<ActivityType>",
"typeProperties":
{
},
"linkedServiceName": "MyLinkedService",
"policy":
{
},
"dependsOn":
```

```
14. {
15. }
16. }
```

The following describes properties in the above JSON:

Property: name

Name of the activity.

Required: Yes

Property: description

Text describing what the activity or is used for.

Required: Yes

Property: type

Defines the type of the activity.

Required: Yes

Property: linkedServiceName

Name of the linked service used by the activity.

Required: Yes for HDInsight, Machine Learning Batch Scoring Activity and Stored Procedure Activity

Property: typeProperties

Properties in the typeProperties section depend on each type of activity.

Required: No

Property: policy

Policies that affect the run-time behaviour of the activity. This property includes timeout and retry behaviour.

Required: No

Property: dependsOn

This property is used to define activity dependencies, and how subsequent activities depend on previous activities.

Required: No

https://docs.microsoft.com/en-us/azure/data-factory/concepts-pipelines-activities

Question 97: **Correct**
What can cause a slower performance on join or shuffle jobs?

- **Enablement of autoscaling**
- **Use the cache option**
- **Bucketing**
- **Data skew**

  (Correct)

Explanation
The data skew is one of the most common reasons why your Apache Spark job is underperforming. Data skew can cause a slower performance on join or shuffle jobs due to asymmetry in your job data.

Spark is a distributed system, and as such, it divides the data into multiple pieces, called partitions, moves them into the different cluster nodes, and processes them in parallel. If one of these partitions happens to be much larger than others, the node processing it may experience the resource issues and slow down entire execution. This kind of data imbalance is called a data skew.

The size of the partitions depends on the factors, like partitioning configuration of the source files, the number of CPU cores and the nature of your query. The most common scenarios, involving the data skew problems, include the aggregation and join queries, where the grouping or joining field has unequally distributed keys (i.e. few keys have much more rows, than the remaining keys). In this scenario, Spark will send the rows with the same key to the same partition and cause data skew issues.

A traditional Apache Spark UI has some dashboards to determine data skew issues. In addition to that, Azure Synapse Analytics introduced nice data skew diagnosis tools.

https://www.mssqltips.com/sqlservertip/6747/azure-synapse-analytics-analyze-data-skew-issues/

Question 98: **Incorrect**
As great as data lakes are at inexpensively storing our raw data, they also bring with them performance challenges:

• **Too many small or very big files** - more time opening & closing files rather than reading contents (worse with streaming).

• **Partitioning also known as "poor man's indexing"**- breaks down if you picked the wrong fields or when data has many dimensions, high cardinality columns.

• **No caching** - cloud storage throughput is low (cloud object storage is 20-50MB/s/core vs 300MB/s/core for local SSDs).

As a solution to the challenges with Data Lakes noted above, [?] is a file format that can help you build a data lake comprised of one or many tables in [?] format. [?] integrates tightly with Apache Spark, and uses an open format that is based on Parquet. Because it is an open-source format, [?] is also supported by other data platforms, including Azure Synapse Analytics.

- **Delta Lake**

  (Correct)

- Data Organizer
- Augmenter

  (Incorrect)
- Data Sea

Explanation

Delta Lake is a transactional storage layer designed specifically to work with Apache Spark and Databricks File System (DBFS). At the core of Delta Lake is an optimized Spark table. It stores your data as Apache Parquet files in DBFS and maintains a transaction log that efficiently tracks changes to the table.

Data lakes

A data lake is a storage repository that inexpensively stores a vast amount of raw data, both current and historical, in native formats such as XML, JSON, CSV, and Parquet. It may contain operational relational databases with live transactional data.

Enterprises have been spending millions of dollars getting data into data lakes with Apache Spark. The aspiration is to do data science and ML on all that data using Apache Spark.

But the data is not ready for data science & ML. The majority of these projects are failing due to unreliable data!

The challenge with data lakes

Why are these projects struggling with reliability and performance?

To extract meaningful information from a data lake, you must solve problems such as:

• Schema enforcement when new tables are introduced.

• Table repairs when any new data is inserted into the data lake.

• Frequent refreshes of metadata.

• Bottlenecks of small file sizes for distributed computations.

• Difficulty sorting data by an index if data is spread across many files and partitioned.

There are also data reliability challenges with data lakes:

• Failed production jobs leave data in corrupt state requiring tedious recovery.

• Lack of schema enforcement creates inconsistent and low quality data.

• Lack of consistency makes it almost impossible to mix appends and reads, batch and streaming.

As great as data lakes are at inexpensively storing our raw data, they also bring with them performance challenges:

• **Too many small or very big files** - more time opening & closing files rather than reading contents (worse with streaming).

• **Partitioning also known as "poor man's indexing"**- breaks down if you picked the wrong fields or when data has many dimensions, high cardinality columns.

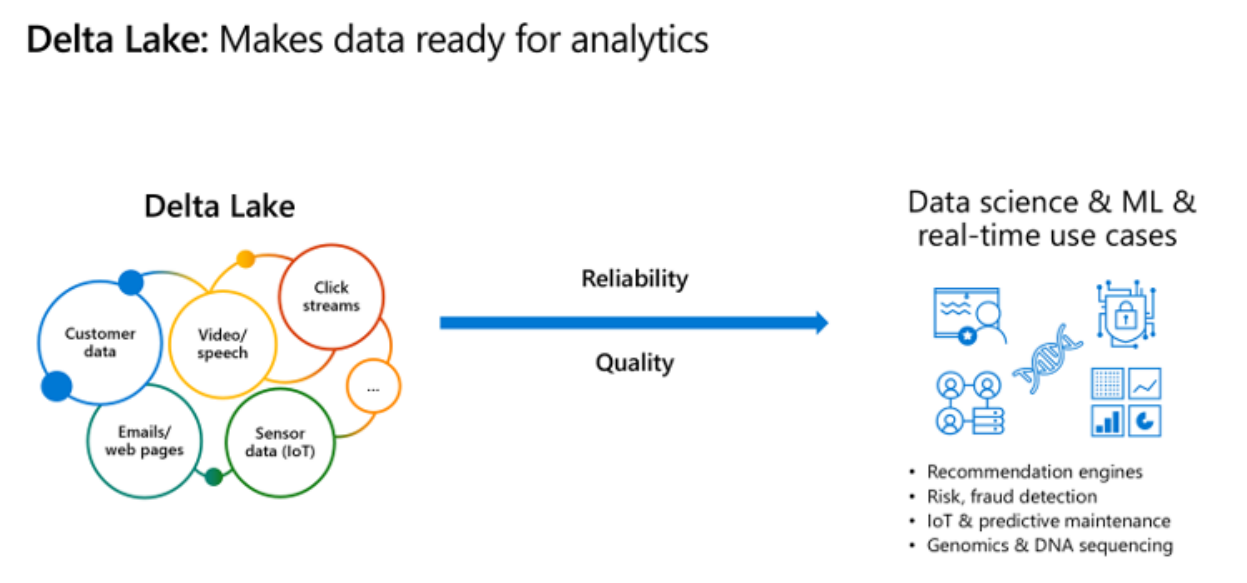
• **No caching** - cloud storage throughput is low (cloud object storage is 20-50MB/s/core vs 300MB/s/core for local SSDs).

The solution: Delta Lake

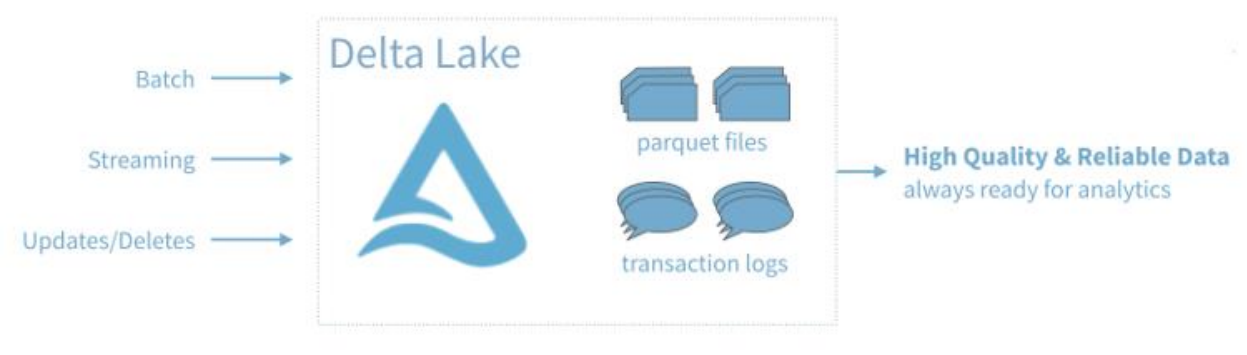
Delta Lake is a file format that can help you build a data lake comprised of one or many tables in Delta Lake format. Delta Lake integrates tightly with Apache Spark, and uses

an open format that is based on Parquet. Because it is an open-source format, Delta Lake is also supported by other data platforms, including Azure Synapse Analytics.

Delta Lake makes data ready for analytics.



Delta Lake is an open-source storage layer that brings ACID transactions to Apache Spark™ and big data workloads.



You can read and write data that's stored in Delta Lake by using Apache Spark SQL batch and streaming APIs. These are the same familiar APIs that you use to work with Hive tables or DBFS directories. Delta Lake provides the following functionality:

ACID Transactions: Data lakes typically have multiple data pipelines reading and writing data concurrently, and data engineers have to go through a tedious process to ensure data integrity, due to the lack of transactions. Delta Lake brings ACID transactions to your data lakes. It provides serializability, the strongest level of isolation level.

Scalable Metadata Handling: In big data, even the metadata itself can be "big data". Delta Lake treats metadata just like data, leveraging Spark's distributed processing power to handle all its metadata. As a result, Delta Lake can handle petabyte-scale tables with billions of partitions and files at ease.

Time Travel (data versioning): Delta Lake provides snapshots of data enabling developers to access and revert to earlier versions of data for audits, rollbacks or to reproduce experiments.

Open Format: All data in Delta Lake is stored in Apache Parquet format enabling Delta Lake to leverage the efficient compression and encoding schemes that are native to Parquet.

Unified Batch and Streaming Source and Sink: A table in Delta Lake is both a batch table, as well as a streaming source and sink. Streaming data ingest, batch historic backfill, and interactive queries all just work out of the box.

Schema Enforcement: Delta Lake provides the ability to specify your schema and enforce it. This helps ensure that the data types are correct and required columns are present, preventing bad data from causing data corruption.

Schema Evolution: Big data is continuously changing. Delta Lake enables you to make changes to a table schema that can be applied automatically, without the need for cumbersome DDL.

100% Compatible with Apache Spark API: Developers can use Delta Lake with their existing data pipelines with minimal change as it is fully compatible with Spark, the commonly used big data processing engine.

Get started with Delta using Spark APIs

Delta Lake is included with Azure Databricks. You can start using it today. To quickly get started with Delta Lake, do the following:

Instead of parquet...

```
1. Python
2. CREATE TABLE ...
3. USING parquet
4. ...
5.
6. dataframe
7. .write
8. .format("parquet")
9. .save("/data")
10.... simply say delta
11.Python
12.CREATE TABLE ...
13.USING delta
14....
15.
16.dataframe
17.write
18.format("delta")
19.save("/data")
```

Using Delta with your existing Parquet tables

Step 1: Convert Parquet to Delta tables:

```
1. Python
2. CONVERT TO DELTA parquet.`path/to/table` [NO STATISTICS]
3. [PARTITIONED BY (col_name1 col_type1, col_name2 col_type2, ...)]
4. Step 2: Optimize layout for fast queries:
5. Python
6. OPTIMIZE events
7. WHERE date >= current_timestamp() - INTERVAL 1 day
8. ZORDER BY (eventType)
```

Basic syntax

Two of the core features of Delta Lake are performing upserts (insert/updates) and Time Travel operations.

To `UPSERT` means to "UPdate" and "inSERT". In other words, `UPSERT` is literally TWO operations. It is not supported in traditional data lakes, as running an UPDATE could invalidate data that is accessed by the subsequent INSERT operation.

Using Delta Lake, however, we can do `UPSERTS`. Delta Lake combines these operations to guarantee atomicity to

• `UPSERT` a row

• if the row already exists, `UPDATE` the row.

Upsert syntax

Upserting, or merging, in Delta Lake provides fine-grained updates of your data. The following syntax shows how to perform an Upsert:

```
1. SQL
2. MERGE INTO customers -- Delta table
3. USING updates
4. ON customers.customerId = source.customerId
5. WHEN MATCHED THEN
6. UPDATE SET address = updates.address
7. WHEN NOT MATCHED
8. THEN INSERT (customerId, address) VALUES (updates.customerId, updates.address)
```

Time Travel syntax

Because Delta Lake is version controlled, you have the option to query past versions of the data. Using a single file storage system, you now have access to several versions your historical data, ensuring that your data analysts will be able to replicate their reports (and compare aggregate changes over time) and your data scientists will be able to replicate their experiments.

Other time travel use cases are:

• Re-creating analyses, reports, or outputs (for example, the output of a machine learning model). This could be useful for debugging or auditing, especially in regulated industries.

• Writing complex temporal queries.

• Fixing mistakes in your data.

• Providing snapshot isolation for a set of queries for fast changing tables.

Example of using time travel to reproduce experiments and reports:

```
1. SQL
2. SELECT count(*) FROM events
3. TIMESTAMP AS OF timestamp
4.
5. SELECT count(*) FROM events
6. VERSION AS OF version
1. Python
2. spark.read.format("delta").option("timestampAsOf",
   timestamp_string).load("/events/")
3. If you need to rollback accidental or bad writes:
4. SQL
5. INSERT INTO my_table
6. SELECT * FROM my_table TIMESTAMP AS OF
7. date_sub( current_date(), 1)
```

https://docs.microsoft.com/en-us/azure/synapse-analytics/spark/apache-spark-what-is-delta-lake

Question 99: **Incorrect**
You need to be familiar with the considerations and techniques that apply to loading a data warehouse. Many enterprise analytical solutions include a relational data warehouse. Data engineers are responsible for implementing ingestion solutions that load data into the data warehouse tables, usually on a regular schedule.

One of the most common patterns for loading a data warehouse is to transfer data from source systems to files in a data lake, ingest the file data into staging tables, and then use SQL statements to load the data from the staging tables into the dimension and fact tables. Usually data loading is performed as a periodic batch process in which inserts and updates to the data warehouse are coordinated to occur at a regular interval.

Which command should you use to load a staging table with data from files in the data lake?

- `COPY`

  (Correct)

- `LOAD`

- `INSERT`

  (Incorrect)

- `STAGE`

Explanation

- *With existing tables, the `COPY` command is the most efficient way to populate the warehouse from the data lake.*
- Although `INSERT` is an option, it doesn't perform as well as the `COPY` command.
- Load and stage are not valid operations in this context

Load Staging tables

One of the most common patterns for loading a data warehouse is to transfer data from source systems to files in a data lake, ingest the file data into staging tables, and then use SQL statements to load the data from the staging tables into the dimension and fact tables. Usually data loading is performed as a periodic batch process in which inserts and updates to the data warehouse are coordinated to occur at a regular interval (for example, daily, weekly, or monthly).

Creating Staging tables

Many organized warehouses have standard structures for staging the database and may even use a specific schema for staging the data. The following code example creates a staging table for product data that will ultimately be loaded into a dimension table:

*Note: This example creates a staging table in the default **dbo** schema. You can also create separate schemas for staging tables with a meaningful name, such as **stage** so architects and users understand the purpose of the schema.*

```
SQL
CREATE TABLE dbo.StageProduct
(
    ProductID NVARCHAR(10) NOT NULL,
    ProductName NVARCHAR(200) NOT NULL,
    ProductCategory NVARCHAR(200) NOT NULL,
    Color NVARCHAR(10),
    Size NVARCHAR(10),
    ListPrice DECIMAL NOT NULL,
    Discontinued BIT NOT NULL
)
WITH
(
    DISTRIBUTION = ROUND_ROBIN,
    CLUSTERED COLUMNSTORE INDEX
);
```

Using the COPY command

You can use the `COPY` statement to load data from the data lake, as shown in the following example:

Note: This is generally the recommended approach to load staging tables due to its high performance throughput.

```
SQL
COPY INTO dbo.StageProduct
    (ProductID, ProductName, ...)
FROM 'https://mydatalake.../data/products*.parquet'
WITH
(
    FILE_TYPE = 'PARQUET',
    MAXERRORS = 0,
    IDENTITY_INSERT = 'OFF'
);
```

https://learn.microsoft.com/en-us/sql/t-sql/statements/copy-into-transact-sql

Using External tables

In some cases, if the data to be loaded is stored in files with an appropriate structure, it can be more effective to create external tables that reference the file location. This way, the data can be read directly from the source files instead of being loaded into the relational store. The following example, shows how to create an external table that references files in the data lake associated with the Azure Synapse Analytics workspace:

```
SQL
CREATE EXTERNAL TABLE dbo.ExternalStageProduct
 (
     ProductID NVARCHAR(10) NOT NULL,
     ProductName NVARCHAR(10) NOT NULL,
 ...
 )
WITH
 (
    DATE_SOURCE = StagedFiles,
    LOCATION = 'folder_name/*.parquet',
    FILE_FORMAT = ParquetFormat
 );
GO
```

https://learn.microsoft.com/en-us/azure/synapse-analytics/sql/develop-tables-external-tables

Question 100: **Correct**
Scenario: Honest Eddie's Car Dealership is an establishment in South Carolina USA, which is dedicated to the purchase and sale of cars and light trucks. Currently the IT team is planning to use applications which publish messages to Azure Event Hub very frequently.

By default, how many partitions will a new Event Hub have?

- **3**
- **2**
- **1**
- **4**

  (Correct)

Explanation
Event Hubs default to 4 partitions. Partitions are the buckets within an Event Hub. Each publication will go into only one partition. Each consumer group may read from one or more than one partition.

Big data apps must be able to process increased throughput by scaling out to meet increased transaction volumes.

Suppose you work in the credit card department of a bank. You're part of a team that manages the system responsible for fraud testing to determine whether to approve or decline each transaction. Your system receives a stream of transactions and needs to process them in real time.

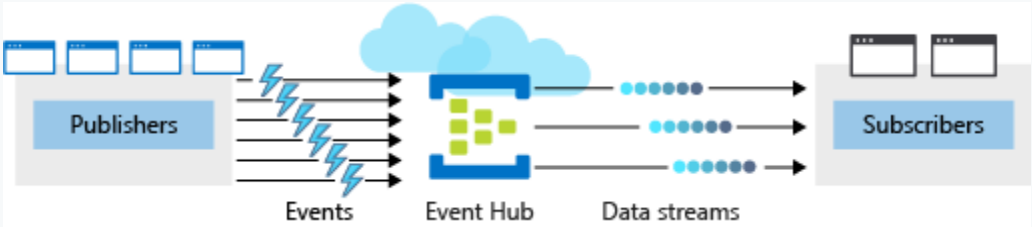
The load on your system can spike during weekends and holidays. The system must handle the increased throughput efficiently and accurately. Given the sensitive nature of the transactions, even the slightest error can have a considerable impact.

Azure Event Hubs can receive and process a large number of transactions. It can also be configured to scale dynamically, when required, to handle increased throughput.

What is an Azure Event Hub?

Azure Event Hubs is a cloud-based, event-processing service that can receive and process millions of events per second. Event Hubs acts as a front door for an event pipeline, to receive incoming data and stores this data until processing resources are available.

An entity that sends data to the Event Hubs is called a *publisher*, and an entity that reads data from the Event Hubs is called a *consumer* or a *subscriber*. Azure Event Hubs sits between these two entities to divide the production (from the publisher) and consumption (to a subscriber) of an event stream. This decoupling helps to manage scenarios where the rate of event production is much higher than the consumption. The following illustration shows the role of an Event Hub.



Events

An **event** is a small packet of information (a *datagram*) that contains a notification. Events can be published individually, or in batches, but a single publication (individual or batch) can't exceed 1 MB.

Publishers and subscribers

Event publishers are any app or device that can send out events using either HTTPS or Advanced Message Queuing Protocol (AMQP) 1.0.

For publishers that send data frequently, AMQP has better performance. However, it has a higher initial session overhead, because a persistent bidirectional socket and transport-level security (TLS) or SSL/TLS has to be set up first.

For more intermittent publishing, HTTPS is the better option. Though HTTPS requires additional overhead for each request, there isn't the session initialization overhead.

Note: Existing Kafka-based clients, using Apache Kafka 1.0 and newer client versions, can also act as Event Hubs publishers.

Event subscribers are apps that use one of two supported programmatic methods to receive and process events from an Event Hub.

• **EventHubReceiver** - A simple method that provides limited management options.

• **EventProcessorHost** - An efficient method that we'll use later in this module.

Consumer groups

An Event Hub **consumer group** represents a specific view of an Event Hub data stream. By using separate consumer groups, multiple subscriber apps can process an event stream independently, and without affecting other apps. However, the use of many consumer groups isn't a requirement, and for many apps, the single default consumer group is sufficient.

Pricing

There are three pricing tiers for Azure Event Hubs: Basic, Standard, and Dedicated. The tiers differ in terms of supported connections, the number of available Consumer groups, and throughput. When using Azure CLI to create an Event Hubs namespace, if you don't specify a pricing tier, the default of **Standard** (20 Consumer groups, 1000 Brokered connections) is assigned.

Create and configure new Azure Event Hubs

There are two main steps when creating and configuring new Azure Event Hubs. The first step is to define the Event Hubs **namespace**. The second step is to create an Event Hub in that namespace.

Define an Event Hubs namespace

An Event Hubs namespace is a containing entity for managing one or more Event Hubs. Creating an Event Hubs namespace typically involves the following configuration:

Define namespace-level settings

Certain settings such as namespace capacity (configured using **throughput units**), pricing tier, and performance metrics are defined at the namespace level. These settings apply to all the Event Hubs within that namespace. If you don't define these settings, a default value is used: *1* for capacity and *Standard* for pricing tier.

Keep the following aspects in mind:

• You must balance your configuration against your Azure budget expectations.

• You might consider configuring different Event Hubs for different throughput requirements. For example, if you have a sales data app, and you're planning for two Event Hubs, it would make sense to use a separate namespace for each hub.

• You'll configure one namespace for high throughput collection of real-time sales data telemetry and one namespace for infrequent event log collection. This way, you only need to configure (and pay for) high throughput capacity on the telemetry hub.

1. Select a unique name for the namespace. The namespace is accessible through this URL: *namespace.servicebus.windows.net*

2. Define the following optional properties:

• Enable Kafka. This option enables Kafka apps to publish events to the Event Hub.

• Make this namespace zone redundant. Zone-redundancy replicates data across separate data centers with their independent power, networking, and cooling infrastructures.

• Enable Auto-Inflate and Auto-Inflate Maximum Throughput Units. Auto-Inflate provides an automatic scale-up option by increasing the number of throughput units up to a maximum value. This option is useful to avoid throttling in situations when incoming or outgoing data rates exceed the currently set number of throughput units.

Azure CLI commands to create an Event Hubs namespace

To create a new Event Hubs namespace, use the `az eventhubs namespace` commands. Here's a brief description of the subcommands you'll use in the exercise.

Command	Description
create	Create the Event Hubs namespace.
authorization-rule	All Event Hubs within the same Event Hubs namespace share common connection credentials. You'll need these credentials when you configure apps to send and receive messages using the Event Hub. This command returns the connection string for your Event Hubs namespace.

Configure a new Event Hub

After you create the Event Hubs namespace, you can create an Event Hub. When creating a new Event Hub, there are several mandatory parameters.

The following parameters are required to create an Event Hub:

• **Event Hub name** - Event Hub name that is unique within your subscription and:

• Is between 1 and 50 characters long.

• Contains only letters, numbers, periods, hyphens, and underscores.

• Starts and ends with a letter or number.

• **Partition Count** - The number of partitions required in an Event Hub (between 2 and 32). The partition count should be directly related to the expected number of concurrent consumers and can't be changed after the hub has been created. The partition separates the message stream so that consumer or receiver apps only need to read a specific subset of the data stream. If not defined, this value defaults to *4*.

• **Message Retention** - The number of days (between 1 and 7) that messages will remain available if the data stream needs to be replayed for any reason. If not defined, this value defaults to *7*.

You can also optionally configure an Event Hub to stream data to an Azure Blob storage or Azure Data Lake Store account.

Azure CLI commands to create an Event Hub

To create a new Event Hub with the Azure CLI, you'll run the `az eventhubs eventhub` command set. Here's a brief description of the subcommands we'll be using.

Command	Description
create	Creates the Event Hub in a specified namespace.
show	Displays the details of your Event Hub.

Summary

To deploy Azure Event Hubs, you must configure an Event Hubs namespace, and then configure the Event Hub itself. In the next unit, you'll go through the detailed configuration steps to create a new namespace and Event Hub.

https://docs.microsoft.com/en-us/azure/event-hubs/event-hubs-quickstart-cli

Question 101: **Incorrect**

Scenario: The UCWF (Unlimited Class Wrestling Federation) was founded by a promoter named Edward Garner, who strives to use technology to improve his business and has come to you for assistance with the company's Microsoft Azure service.

Edward wants to create an Azure Synapse Analytics workspace to implement Azure Synapse Link for Azure SQL Database.

Which of the following should UCWF use?

- **A link connection for your Azure SQL Database**
- **A serverless SQL pool**
- **A linked service for your Azure SQL Database**

(Correct)

- **An Azure SQL Managed Instance**

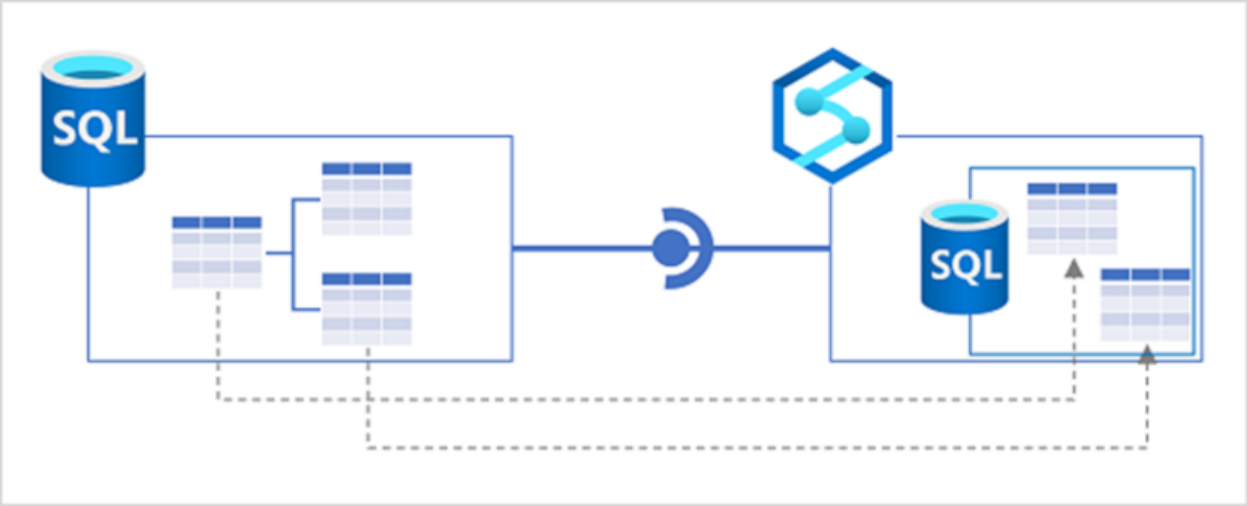
  (Incorrect)

Explanation
They should linked service for your Azure SQL Database.

Configure Azure Synapse Link for Azure SQL Database

Azure SQL Database is a platform-as-a-service (PaaS) relational database service based on the SQL Server database engine. It's commonly used in cloud-native applications as a scalable, secure, and easy to manage relational database store for operational data.

Azure Synapse Link for Azure SQL Database uses a *link connection* to map one or more tables in an Azure SQL Database instance to tables in a dedicated SQL pool in Azure Synapse Analytics. When the link connection is started, the tables are initialized by copying a snapshot of the source tables to the target tables. Subsequently, the change feed process applies all modifications made in the source tables to the target tables.



Implementing Azure Synapse Link for Azure SQL Database

To use Azure Synapse Link for Azure SQL Database, you need to configure some settings in your Azure SQL Database server, before creating a link connection in Azure Synapse Analytics.

Configure Azure SQL Database

Before you can use Azure SQL Database as a source for a linked connection in Azure Synapse Analytics, you must ensure the following settings are configured in the Azure SQL Database server that hosts the database you want to synchronize:

• **System assigned managed identity** - enable this option so that your Azure SQL Database server uses a system assigned managed identity.

• **Firewall rules** - ensure that Azure services can access your Azure SQL Database server.

In addition to these server-level settings, if you plan to configure the link connection from Azure Synapse Analytics to use a managed identity when connecting to Azure SQL Database, you must create a user for the workspace identity in the database and add it to the **db_owner** role, as shown in the following code example:

```
1. SQL
2. CREATE USER my_synapse_workspace FROM EXTERNAL PROVIDER;
3. ALTER ROLE [db_owner] ADD MEMBER my_synapse_workspace;
```

If you intend to use SQL authentication, you can omit this step.

Prepare the target SQL pool

Azure Synapse Link for Azure SQL Database synchronizes the source data to tables in a dedicated SQL pool in Azure Synapse Analytics. You therefore need to create and start a dedicated SQL pool in your Azure Synapse Analytics workspace before you can create the link connection.

The database associated with the dedicated SQL pool must include the appropriate schema for the target table. If source tables are defined in a schema other than the default **dbo** schema, you must create a schema of the same name in the dedicated SQL pool database:

```
1. SQL
2. CREATE SCHEMA myschema;
```

Create a link connection

To create a linked connection, add a **linked connection** on the **Integrate** page in Azure Synapse Studio. You'll need to:

1. Select or create a *linked service* for your Azure SQL Database. You can create this separately ahead of time, or as part of the process of creating a linked connection for Azure Synapse Link. You can use a managed identity or SQL authentication to connect the linked service to Azure SQL Database.

2. Select the tables in the source database that you want to include in the linked connection.

3. Select the target dedicated SQL pool in which the target tables should be created.

4. Specify the number of CPU cores you want to use to process synchronization. Four driver cores will be used in addition to the number of cores you specify.

After creating the linked connection, you can configure the mappings between the source and target tables. In particular, you can specify the table structure (index) type and distribution configuration for the target tables.

Some data types in your source tables may not be supported by specific dedicated SQL pool index types. For example, you cannot use a clustered columnstore index for tables that include `VARBINARY(MAX)` columns. You can map such tables to a heap (an unindexed table) in the dedicated SQL pool.

When the linked connection is configured appropriately, you can start it to initialize synchronization. The source tables are initially copied to the target database as snapshots, and then subsequent data modifications are replicated.

https://learn.microsoft.com/en-us/azure/synapse-analytics/synapse-link/sql-database-synapse-link

https://learn.microsoft.com/en-us/azure/synapse-analytics/synapse-link/synapse-link-for-sql-known-issues

https://learn.microsoft.com/en-us/azure/synapse-analytics/synapse-link/connect-synapse-link-sql-database

Question 102: **Incorrect**
Within the context of Azure Databricks, sharing data from one worker to another can be a costly operation.

Sharing data from one worker to another can be a costly operation.

Spark has optimized this operation by using a format called Tungsten which prevents the need for expensive serialization and de-serialization of objects in order to get data from one JVM to another.

The data that is "shuffled" is in a format known as `UnsafeRow`, or more commonly, the Tungsten Binary Format.

When we shuffle data, it creates what is known as a stage boundary which represents a process bottleneck which Spark will break this one job into two stages.

In **Stage #1**, Spark will create a pipeline of transformations in which the data is read into RAM.

For **Stage #2**, Spark will again create a pipeline of transformations in which the shuffle data is read into RAM.

From the developer's perspective, we start with a read and conclude (in this case) with a write:

Step Transformation

1 Read

2 Select

3 Filter

4 GroupBy

5 Select

6 Filter

7 Write

However, Spark starts with the action (`write(..)` in this case).

What is the main benefit of working backward through your action's lineage?

- **It allows Spark to determine if it is necessary to execute every transformation.**

  (Correct)

- **It allows Azure to distribute the load to the required number of processors to optimize the load.**

  (Incorrect)

- **It serializes the work to make the work sequential, thereby lowering CPU and RAM cost.**

- **It allows Spark to work on various activities simultaneously using multiple nodes.**

Explanation

As opposed to narrow transformations, wide transformations cause data to shuffle between executors. This is because a wide transformation requires sharing data across workers. **Pipelining** helps us optimize our operations based on the differences between the two types of transformations.

Pipelining

• Pipelining is the idea of executing as many operations as possible on a single partition of data.

• Once a single partition of data is read into RAM, Spark will combine as many narrow operations as it can into a single **Task**

• Wide operations force a shuffle, conclude a stage, and end a pipeline.

Shuffles

A shuffle operation is triggered when data needs to move between executors.

To carry out the shuffle operation Spark needs to:

• Convert the data to the UnsafeRow, commonly referred to as **Tungsten Binary Format**.

• Write that data to disk on the local node - at this point the slot is free for the next task.

• Send that data across the wire to another executor

  • Technically the Driver decides which executor gets which piece of data.

  • Then the executor pulls the data it needs from the other executor's shuffle files.

• Copy the data back into RAM on the new executor

• The concept, if not the action, is just like the initial read "every“ `DataFrame` starts with.

• The main difference being it's the 2nd+ stage.

As we will see in a moment, this amounts to a free cache from what is effectively temp files.

Some actions induce in a shuffle. Good examples would include the operations `count()` and `reduce(..)`.

UnsafeRow (also known as Tungsten Binary Format)

Sharing data from one worker to another can be a costly operation.

Spark has optimized this operation by using a format called **Tungsten**.

Tungsten prevents the need for expensive serialization and de-serialization of objects in order to get data from one JVM to another.

The data that is "shuffled" is in a format known as `UnsafeRow`, or more commonly, the Tungsten Binary Format.

`UnsafeRow` is the in-memory storage format for Spark SQL, DataFrames & Datasets.

Advantages include:

• Compactness:

• Column values are encoded using custom encoders, not as JVM objects (as with RDDs).

• The benefit of using Spark 2.x's custom encoders is that you get almost the same compactness as Java serialization, but significantly faster encoding/decoding speeds.

• Also, for custom data types, it is possible to write custom encoders from scratch.

• Efficiency: Spark can operate *directly out of Tungsten*, without first deserializing Tungsten data into JVM objects.

How UnsafeRow works

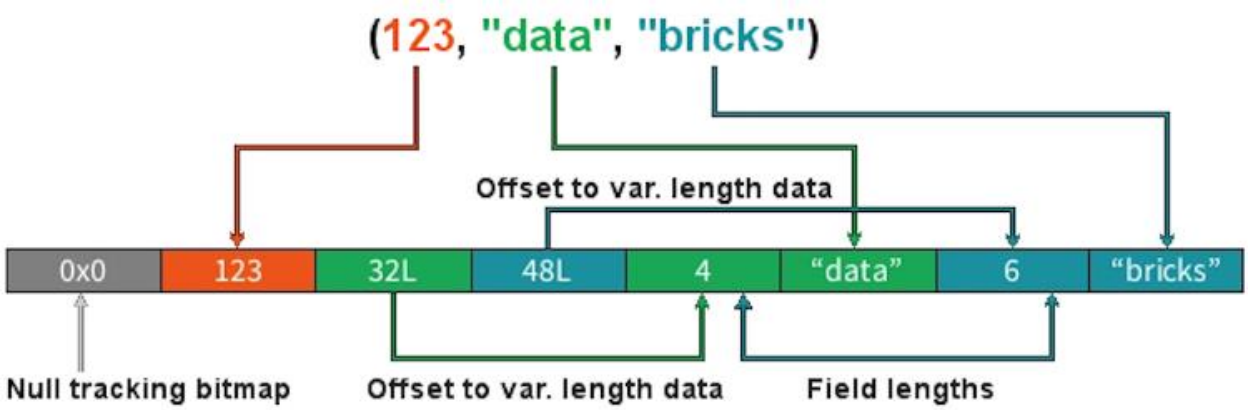
• The first field, "123", is stored in place as its primitive.

• The next 2 fields, "data" and "bricks", are strings and are of variable length.

• An offset for these two strings is stored in place (32L and 48L respectively shown in the picture below).

• The data stored in these two offset's are of format "length + data".

• At offset 32L, we store 4 + "data" and likewise at offset 48L we store 6 + "bricks".



Stages

• When we shuffle data, it creates what is known as a stage boundary.

• Stage boundaries represent a process bottleneck.

Take for example the following transformations:

Step Transformation

1 Read

2 Select

3 Filter

4 GroupBy

5 Select

6 Filter

7 Write

Spark will break this one job into two stages (steps 1-4b and steps 4c-7):

Stage #1

Step Transformation

1 Read

2 Select

3 Filter

4a GroupBy 1/2

4b shuffle write

Stage #1

Step Transformation

4c shuffle read

4d GroupBy 2/2

5 Select

6 Filter

7 Write

In **Stage #1**, Spark will create a pipeline of transformations in which the data is read into RAM (Step #1), and then perform steps #2, #3, #4a & #4b

All partitions must complete **Stage #1** before continuing to **Stage #2**

• It's not possible to group all records across all partitions until every task is completed.

• This is the point at which all the tasks must synchronize.

• This creates our bottleneck.

• Besides the bottleneck, this is also a significant performance hit: disk IO, network IO and more disk IO.

Once the data is shuffled, we can resume execution...

For **Stage #2**, Spark will again create a pipeline of transformations in which the shuffle data is read into RAM (Step #4c) and then perform transformations #4d, #5, #6 and finally the write action, step #7.

Lineage

From the developer's perspective, we start with a read and conclude (in this case) with a write:

Step Transformation

1 Read

2 Select

3 Filter

4 GroupBy

5 Select

6 Filter

7 Write

However, Spark starts with the action (write(..) in this case).

Next, it asks the question, what do I need to do first?

It then proceeds to determine which transformation precedes this step until it identifies the first transformation.

Step Transformation

7 Write Depends on #6

6 Filter Depends on #5

5 Select Depends on #4

4 GroupBy Depends on #3

3 Filter Depends on #2

2 Select Depends on #1

1 Read First

Why Work Backwards?

Question: So what is the benefit of working backward through your action's lineage?

Answer: It allows Spark to determine if it is necessary to execute every transformation.

Take another look at our example:

• Say we've executed this once already

• On the first execution, step #4 resulted in a shuffle

• Those shuffle files are on the various executors (src & dst)

• Because the transformations are immutable, no aspect of our lineage can change.

• That means the results of our last shuffle (if still available) can be reused.

Why Work Backwards?

Step Transformation

7 Write Depends on #6

6 Filter Depends on #5

5 Select Depends on #4

4 GroupBy <<< shuffle

3 Filter don't care

2 Select don't care

1 Read don't care

In this case, what we end up executing is only the operations from **Stage #2**.

This saves us the initial network read and all the transformations in **Stage #1**

Step Transformation

1 Read skipped

2 Select skipped

3 Filter skipped

4a GroupBy 1/2 skipped

4b shuffle write skipped

4c shuffle read -

4d GroupBy 2/2 -

5 Select -

6 Filter -

7 Write

And Caching...

The reuse of shuffle files (also known as our temp files) is just one example of Spark optimizing queries anywhere it can.

We cannot assume this will be available to us.

Shuffle files are by definition temporary files and will eventually be removed.

However, we cache data to explicitly accomplish the same thing that happens inadvertently with shuffle files.

In this case, the lineage plays the same role. Take for example:

Step Transformation

7 Write Depends on #6

6 Filter Depends on #5

5 Select <<< cache

4 GroupBy <<< shuffle files

3 Filter ?

2 Select ?

1 Read ?

In this case we cached the result of the select(..).

We never even get to the part of the lineage that involves the shuffle, let alone Stage #1.

Instead, we pick up with the cache and resume execution from there:

Step Transformation

1 Read skipped

2 Select skipped

3 Filter skipped

4a GroupBy 1/2 skipped

4b shuffle write skipped

4c shuffle read skipped

4d GroupBy 2/2 skipped

5a cache read -

5b Select -

6 Filter -

7 Write

https://databricks.com/blog/2015/04/28/project-tungsten-bringing-spark-closer-to-bare-metal.html

Question 103: **Incorrect**

Scenario: Marchand Pharmaceutical is a medical research corporation, owned and administrated by Michael Marchand. Company resources were used in the studying and implementing the partial data belonging to Dr. Jonathan Drew.

Michael is preparing a report to the board and has asked you to share data visualizations with him, including charts and tables of data. These visualizations will be shared with others in the organization as well.

What should you create?

- **A table**
- **A query**
- **A dashboard**

  (Correct)
- **A view**

  (Incorrect)

Explanation

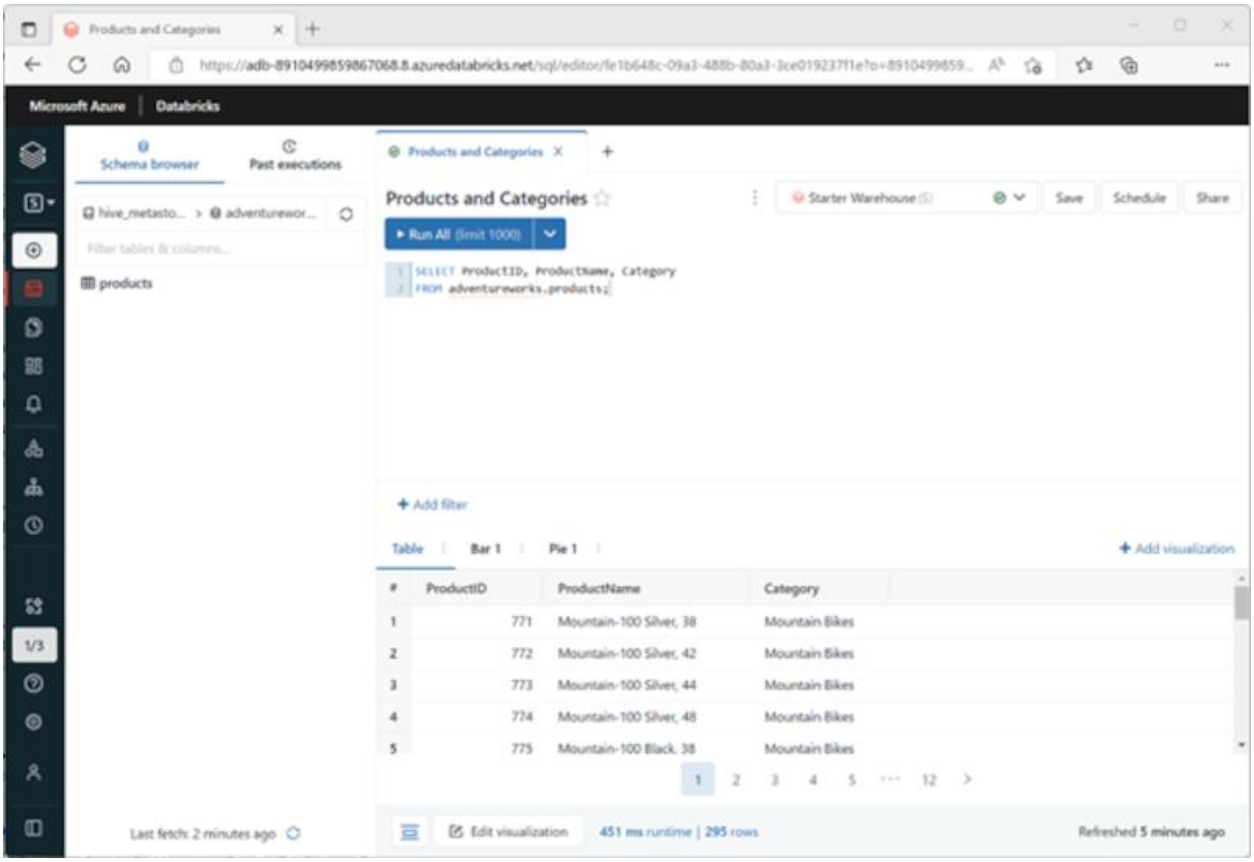
A dashboard can be used to share data visualizations with other users.

Create queries and dashboards

Azure Databricks SQL is primarily designed for data analytics and visualization workloads. To support these workloads, users can create *queries* to retrieve and summarize data from tables, and *dashboards* to share visualizations of the data.

Queries

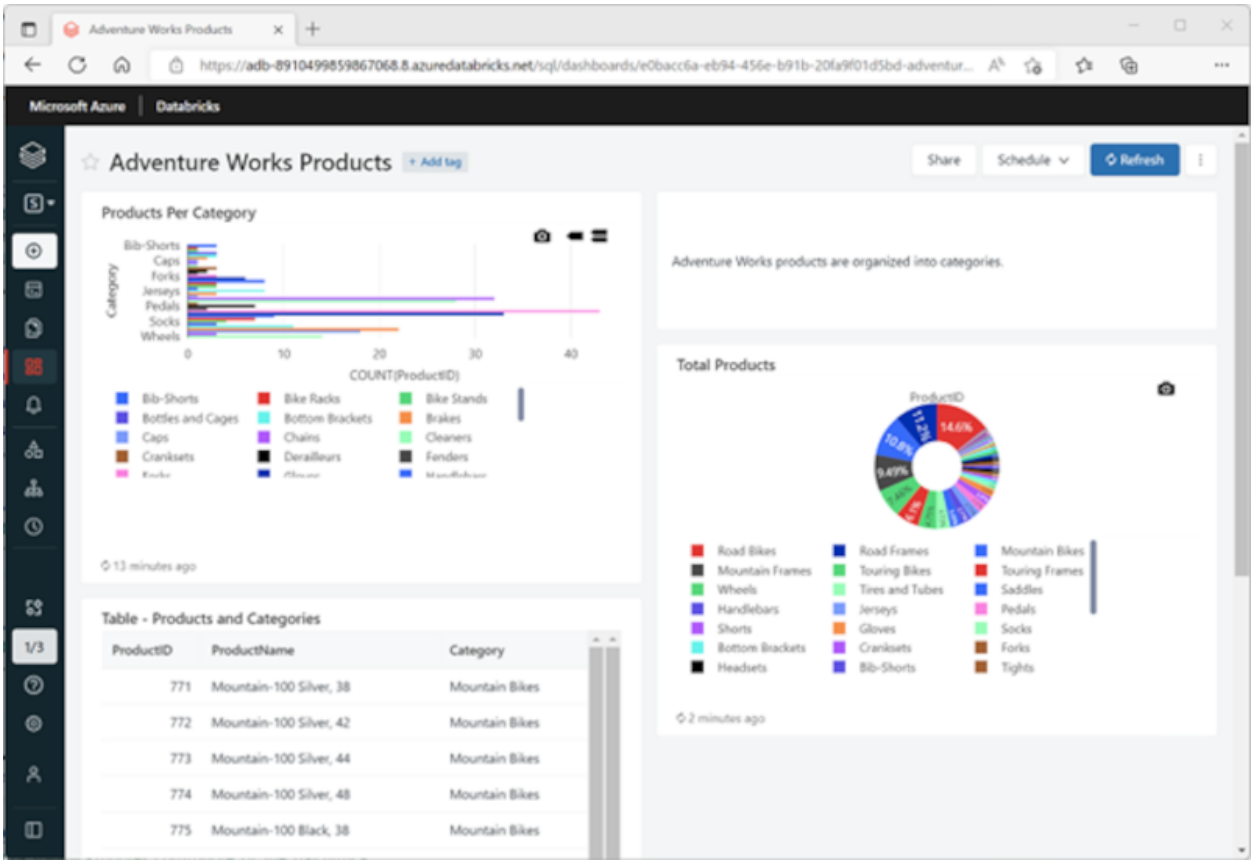
You can use the SQL Editor in the Azure Databricks portal to create a query based on any valid SQL `SELECT` statement, and then save the query with a meaningful name to be retrieved and run later.



After saving the query, you can schedule it to be run automatically at regular intervals to refresh the data, or you can open it and run it interactively.

Dashboards

Dashboards enable you to display the results of queries, either as tables of data or as graphical visualizations.



You can create multiple visualizations in a dashboard and share it with users in your organization. As with individual queries, you can schedule the dashboard to refresh is data periodically and notify subscribers by email that new data is available.

https://learn.microsoft.com/en-us/azure/databricks/sql/user/dashboards/

Question 104: **Incorrect**

See the following code:

```
PowerShell
$SubscriptionId = "add your subscription here"

Add-AzureRmAccount
Set-AzureRmContext -SubscriptionId $SubscriptionId

Register-AzureRmResourceProvider -ProviderNamespace Microsoft.DataFactory

$resourceGroupName = "cto_ignite"
$rglocation = "West US 2"

New-AzureRmDataFactoryV2 -ResourceGroupName $resourceGroupName -Name "ctoigniteADF" -Location $rglocation
```

What is this code template used to setup?

- **Azure Linked Service**
- **Azure Storage Account**

  (Incorrect)
- **Azure Network Security Groups**
- **Azure Private Endpoint**
- **Azure Synapse Spark**
- **Azure SQL Datawarehouse**
- **Azure Data Factory**

(Correct)

Explanation
It is easy to set up Azure Data Factory from within the Azure portal, you only require the following information:

• **Name**: The name of the Azure Data Factory instance

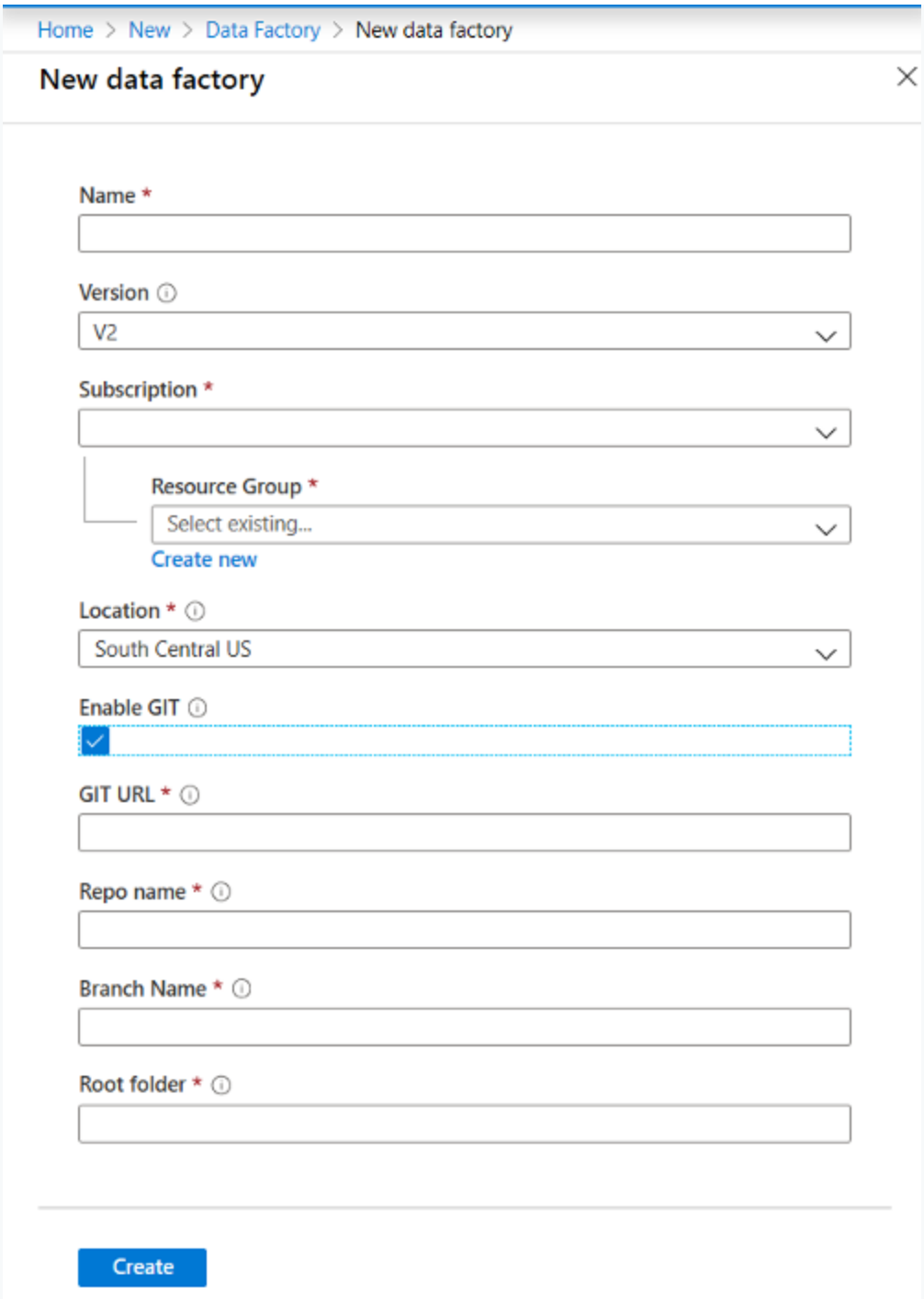
• **Subscription**: The subscription in which the ADF instance is created

• **Resource group**: The resource group where the ADF instance will reside

• **Version**: select V2 for the latest features

• **Location**: The datacentre location in which the instance is stored

Enable Git provides the capability to integrate the code that you create with a Git repository enabling you to source control the code that you would create. Define the GIT url, repository name, branch name, and the root folder.



Alternatively, there are a number of different ways that you can provision the service programmatically. In this example you can see PowerShell at work to set up the environment.

```
1. PowerShell
2. ######################################################################
```

```
## PART I: Creating an Azure Data Factory ##
#####################################################################


# Sign in to Azure and set the WINDOWS AZURE subscription to work with
$SubscriptionId = "add your subscription in the quotes"

Add-AzureRmAccount
Set-AzureRmContext -SubscriptionId $SubscriptionId

# register the Microsoft Azure Data Factory resource provider
Register-AzureRmResourceProvider -ProviderNamespace Microsoft.DataFactory

# DEFINE RESOURCE GROUP NAME AND LOCATION PARAMETERS
$resourceGroupName = "cto_ignite"
$rglocation = "West US 2"

# CREATE AZURE DATA FACTORY
New-AzureRmDataFactoryV2 -ResourceGroupName $resourceGroupName -Name "ctoigniteADF" -Location $rglocation
```

https://docs.microsoft.com/en-us/azure/data-factory/quickstart-create-data-factory-portal

Question 105: **Correct**

Spark pools in Azure Synapse Analytics is one of Microsoft's implementation of Apache Spark.

Which of the following are true about Spark pools in Azure Synapse Analytics? (Select all that apply)

- ☐ **The SparkContext is able to connect to the cluster manager, which allocates resources across applications. The cluster manager is Adobe Hadoop WOOL.**
- ☐ **Once connected, Sparkle gets the executors on nodes in the pool. Those processes run computations and store data on your local machine.**
- ☐ **The SparkContext connects to the Sparkle pool in Synapse Analytics. It is responsible for converting an application to an Excel file.**
- ☑ **Spark applications act as independent sets of processes on a pool. It is coordinated by the SParkContext object in a main (driver) program**

(Correct)

Explanation
Apache Spark in Azure Synapse Analytics

Spark pools in Azure Synapse Analytics is one of Microsoft's implementation of Apache Spark, version Spark 2.4 for the Azure cloud.

Azure Synapse Analytics enables you to have a one-stop shop for your Analytics environment. With the addition of Spark Pools in Azure Synapse Analytics, it is now also possible to benefit from the features of Apache Spark in the same environment where you can set up your data warehousing solution. The spark pools within Azure Synapse Analytics are compatible with different Azure Storage solutions such as ADLS Gen2 and Blob Storage. It is imperative to know that currently providing Spark pools in an Azure Synapse Analytics workspace preview environment, is provided without a service level agreement and therefore not (yet) recommended for production workloads. In addition, some of the official Apache Spark documentation relies on using the spark console. At this moment, the spark console is not available on Azure Synapse Spark, so therefore it is highly recommended to use the notebook or IntelliJ experiences instead.

Spark Pools in Azure Synapse Analytics, a fully managed and integrated Spark service

Benefits of Spark Pools in Azure Synapse Analytics are listed below:

• Speed and Efficiency: Quick start-up time for nodes, automatic shut-down when instances are not used within 5 min after last job, unless there is a live notebook connection.

• Ease of creation: Creating a spark pool can be done through the Azure portal, PowerShell, or .NET SDK for Azure Synapse Analytics.

• Ease of use: Within the Azure Synapse Analytics workspace, you can connect directly to the Spark pool and interact with the integrated notebook experience, or use custom notebooks derived from Nteract. Notebook integration helps you in developing interactive data processing and visualization pipelines.

• REST APIs: In order to monitor and submit jobs remotely, you can use Apache Livy as Rest API Spark job server.

• Integration with third-party IDEs: Azure Synapse Analytics provides an IDE for IntelliJ to create and submit applications to the spark pool

• Pre-loaded Anaconda libraries: Over 200 Anaconda libraries pre-installed on the spark pool.

• Scalability: Possibility for autoscale, such that pools can be up/down scaled as required by adding or removing nodes.

Spark pools in Azure Synapse include the following components that are available on the pools by default.

• Spark Core. Includes Spark Core, Spark SQL, GraphX, and MLlib.

• Anaconda

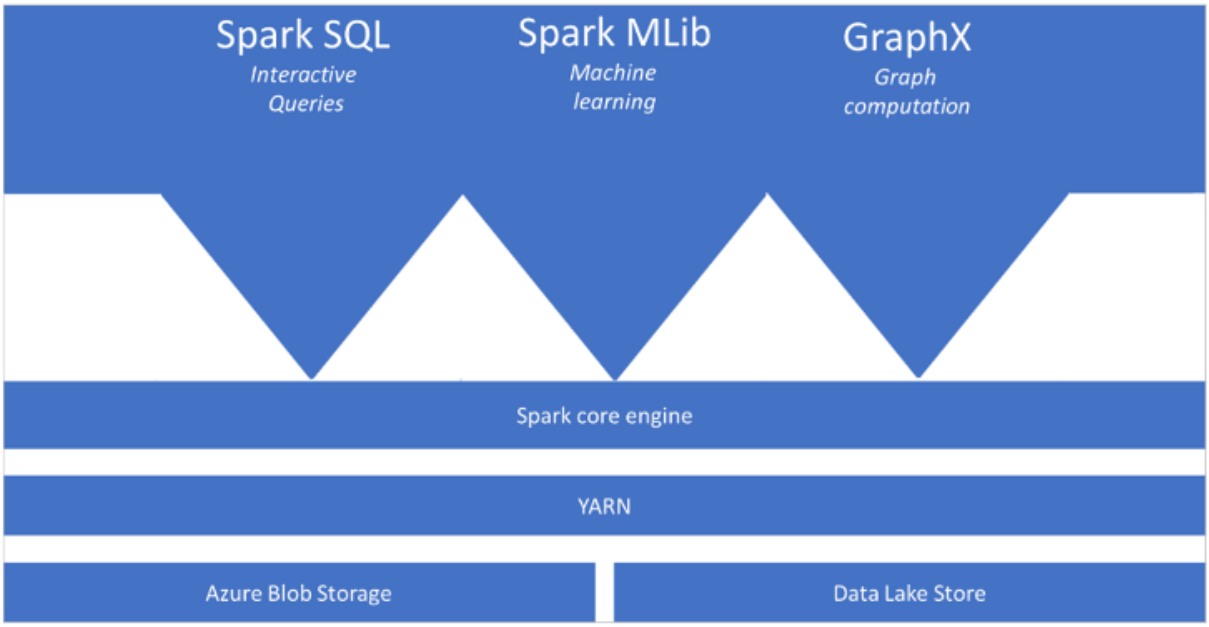
• Apache Livy

• Nteract notebook

The supported languages and runtime versions for Apache spark and dependent components in Azure Synapse analytics can be found here:

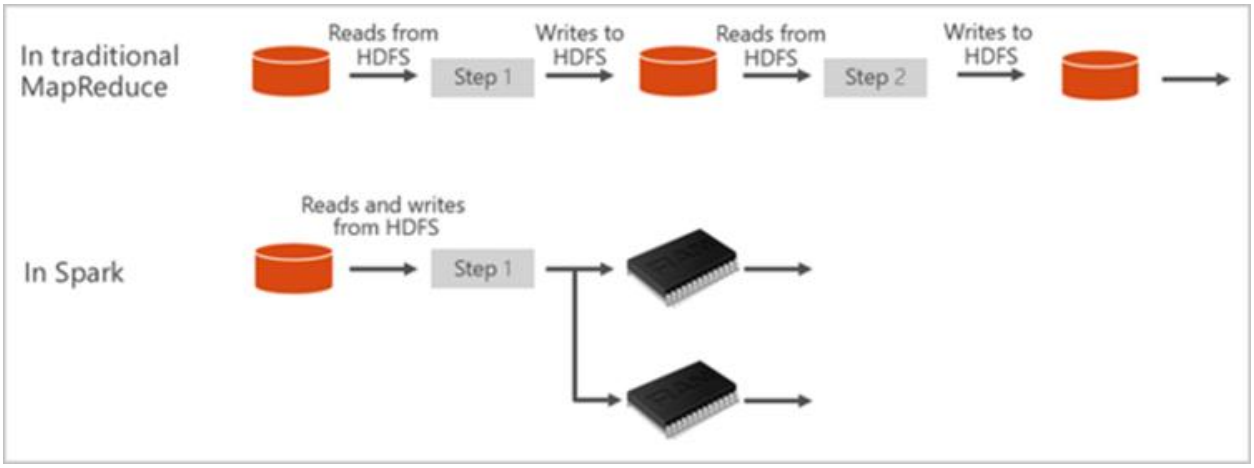
• Apache Spark components in Azure Synapse Analytics

Spark pool architecture

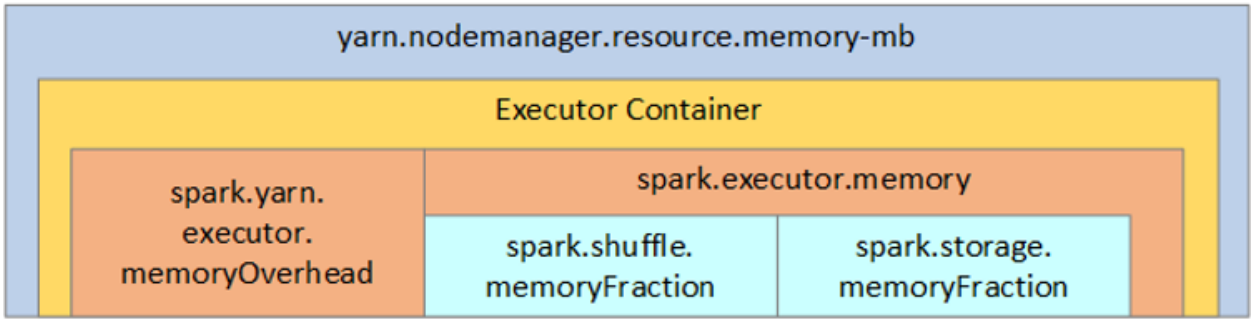


It is imperative to understand the components of Spark by understanding how Spark runs on Synapse Analytics. **The different spark applications act as independent sets of**

processes on a pool. It is coordinated by the SParkContext object in a main (driver) program.



The SparkContext is able to connect to the cluster manager, which allocates resources across applications. The cluster manager is [Apache Hadoop YARN](#).



Once connected, Spark gets the executors on nodes in the pool. Those processes run computations and store data for your application. What follows is that your application code (defined by JAR or Python files passed to SparkContext) will be sent to the executors. Finally, SparkContext is able to send tasks to the executors to run.

The SparkContext runs the user's so your main function. What is then will do is execute the various parallel operations on the nodes. Then, the SparkContext will collect all the results of the operations that were sent to the nodes. The nodes are able to read and write data from and to the file system. Like mentioned in the introduction, the nodes caches the transformed data in-memory as Resilient Distributed Datasets (RDDs).

The SparkContext connects to the Spark pool in Synapse Analytics. It is responsible for converting an application to a directed acyclic graph (DAG). The graph consists of individual tasks that get executed within an executor process on the nodes. Each application gets its own executor processes, which stay up for the duration of the whole application and run tasks in multiple threads.

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

Question 106: **Incorrect**
What steps are required to authorize Azure DevOps to connect to and deploy notebooks to a staging or production Azure Databricks workspace?

- **None of the listed options.**

- **Create an Azure Active Directory application, copy the application ID, then use that as the Databricks bearer token in the Databricks Notebooks Deployment step of the Release pipeline.**

  (Correct)

- **Create a new Access Token within the user settings in the production Azure Databricks workspace, then use the token as the Databricks bearer token in the Databricks Notebooks Deployment step of the Release pipeline.**

- **In the production or staging Azure Databricks workspace, enable Git integration to Azure DevOps, then link to the Azure DevOps source code repo.**

  (Incorrect)

Explanation
To authorize Azure DevOps to connect to and deploy notebooks to a staging or production Azure Databricks workspace, create an Azure Active Directory application, copy the application ID, then use that as the Databricks bearer token in the Databricks Notebooks Deployment step of the Release pipeline.

The Access Token allows you to grant access to resources within an Azure Databricks workspace without passing in user credentials.

https://social.technet.microsoft.com/wiki/contents/articles/53094.azure-devops-integrate-with-an-azure-subscription-or-management-group.aspx

Question 107: **Incorrect**
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.

At the moment, you have been hired by OZcorp as a Microsoft Azure Synapse Analytics SME.

Given:

OZcorp has an on-premises data warehouse that includes the following fact tables. Both tables have the following columns: DateKey, ProductKey, RegionKey.

• **Table - Sales:** The table is 600 GB in size. DateKey is used extensively in the `WHERE` clause queries. ProductKey is used extensively in join operations. RegionKey is used for grouping. Seventy-five percent of the records relate to one of forty regions.

• **Table - Invoice:** The table is 6 GB in size. DateKey and ProductKey are used extensively in the `WHERE` clause queries. RegionKey is used for grouping.

• There are 120 unique product keys and 65 unique region keys.

• Queries that use the data warehouse take a long time to complete.

Required:

The team plans to migrate the solution to use Azure Synapse Analytics and they need to ensure that the Azure-based solution optimizes query performance and minimizes processing skew.

Proposed Solution:

The team has chosen to use the following:

• **Table - Sales:** Distribution type: Hash-distributed, Distribution column: ProductKey

• **Table - Invoice:** Distribution type: Round-robin, Distribution column: RegionKey

Azure Synapse Analytics SME, the team looks to you for reassurance that they made the right choices. Did they?

- **No**

  (Correct)

- **Yes**

  (Incorrect)

Explanation
No, the team did not choose the correct option; both hashes are > 2GB. The Invoice table RegionKey cannot be used with Round-robin distribution as Round-robin does not take a distribution key. Hash-distributed for the Distribution type and ProductKey for the Distribution column is correct for the Sales table.

This is because ProductKey is used extensively in joins and Hash-distributed tables improve query performance on large fact tables.

What is a distributed table?

A distributed table appears as a single table, but the rows are actually stored across 60 distributions. The rows are distributed with a hash or round-robin algorithm.

Hash-distributed tables improve query performance on large fact tables, and are the focus of this article. **Round-robin tables** are useful for improving loading speed. These design choices have a significant impact on improving query and loading performance.

Another table storage option is to replicate a small table across all the Compute nodes. For more information, see Design guidance for replicated tables. To quickly choose among the three options, see Distributed tables in the tables overview.

As part of table design, understand as much as possible about your data and how the data is queried. For example, consider these questions:

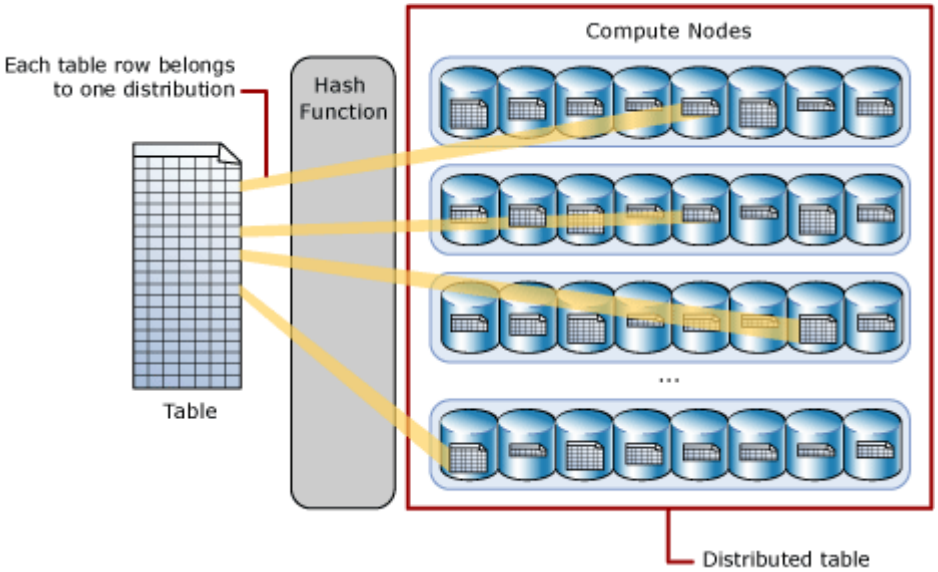
• How large is the table?

• How often is the table refreshed?

• Do I have fact and dimension tables in a dedicated SQL pool?

Hash distributed

A hash-distributed table distributes table rows across the Compute nodes by using a deterministic hash function to assign each row to one distribution.



Since identical values always hash to the same distribution, SQL Analytics has built-in knowledge of the row locations. In dedicated SQL pool this knowledge is used to minimize data movement during queries, which improves query performance.

Hash-distributed tables work well for large fact tables in a star schema. They can have very large numbers of rows and still achieve high performance. There are, of course, some design considerations that help you to get the performance the distributed system is designed to provide. Choosing a good distribution column is one such consideration that is described in this article.

Consider using a hash-distributed table when:

• The table size on disk is more than 2 GB.

• The table has frequent insert, update, and delete operations.

Round-robin distributed

A round-robin distributed table distributes table rows evenly across all distributions. The assignment of rows to distributions is random. Unlike hash-distributed tables, rows with equal values are not guaranteed to be assigned to the same distribution.

As a result, the system sometimes needs to invoke a data movement operation to better organize your data before it can resolve a query. This extra step can slow down

your queries. For example, joining a round-robin table usually requires reshuffling the rows, which is a performance hit.

Consider using the round-robin distribution for your table in the following scenarios:

• When getting started as a simple starting point since it is the default

• If there is no obvious joining key

• If there is no good candidate column for hash distributing the table

• If the table does not share a common join key with other tables

• If the join is less significant than other joins in the query

• When the table is a temporary staging table

Choosing a distribution column

A hash-distributed table has a distribution column that is the hash key. For example, the following code creates a hash-distributed table with ProductKey as the distribution column.

```
SQL
CREATE TABLE [dbo].[FactInternetSales]
(   [ProductKey]            int          NOT NULL
,   [OrderDateKey]          int          NOT NULL
,   [CustomerKey]           int          NOT NULL
,   [PromotionKey]          int          NOT NULL
,   [SalesOrderNumber]      nvarchar(20) NOT NULL
,   [OrderQuantity]         smallint     NOT NULL
,   [UnitPrice]             money        NOT NULL
,   [SalesAmount]           money        NOT NULL
)
WITH
(   CLUSTERED COLUMNSTORE INDEX
,  DISTRIBUTION = HASH([ProductKey])
)
;
```

Data stored in the distribution column can be updated. Updates to data in the distribution column could result in data shuffle operation.

Choosing a distribution column is an important design decision since the values in this column determine how the rows are distributed. The best choice depends on several factors, and usually involves tradeoffs. Once a distribution column is chosen, you cannot change it.

If you didn't choose the best column the first time, you can use CREATE TABLE AS SELECT (CTAS) to re-create the table with a different distribution column.

Choose a distribution column with data that distributes evenly

For best performance, all of the distributions should have approximately the same number of rows. When one or more distributions have a disproportionate number of rows, some distributions finish their portion of a parallel query before others. Since the query can't complete until all distributions have finished processing, each query is only as fast as the slowest distribution.

Data skew means the data is not distributed evenly across the distributions

Processing skew means that some distributions take longer than others when running parallel queries. This can happen when the data is skewed.

To balance the parallel processing, select a distribution column that:

Has many unique values. The column can have some duplicate values. However, all rows with the same value are assigned to the same distribution. Since there are 60 distributions, the column should have at least 60 unique values. Usually the number of unique values is much greater.

Does not have NULLs, or has only a few NULLs. For an extreme example, if all values in the column are NULL, all the rows are assigned to the same distribution. As a result, query processing is skewed to one distribution, and does not benefit from parallel processing.

Is not a date column. All data for the same date lands in the same distribution. If several users are all filtering on the same date, then only 1 of the 60 distributions do all the processing work.

Choose a distribution column that minimizes data movement

To get the correct query result queries might move data from one Compute node to another. Data movement commonly happens when queries have joins and aggregations on distributed tables. Choosing a distribution column that helps minimize data movement is one of the most important strategies for optimizing performance of your dedicated SQL pool.

To minimize data movement, select a distribution column that:

Is used in `JOIN`, `GROUP BY`, `DISTINCT`, `OVER`, and `HAVING` clauses. When two large fact tables have frequent joins, query performance improves when you distribute both tables

on one of the join columns. When a table is not used in joins, consider distributing the table on a column that is frequently in the `GROUP BY` clause.

Is *not* used in `WHERE` clauses. This could narrow the query to not run on all the distributions.

Is *not* a date column. `WHERE` clauses often filter by date. When this happens, all the processing could run on only a few distributions.

https://docs.microsoft.com/en-us/azure/sql-data-warehouse/sql-data-warehouse-tables-distribute

Question 108: **Correct**
Scenario: You are working as a consultant at Avengers Security and at the moment, you are working with the data engineering team which manages Azure HDInsight clusters at the company. The group spends an enormous amount of time creating and destroying clusters each day due to the fact that the majority of the data pipeline process runs in minutes.

Required: Utilize an Azure solution which will deploy multiple HDInsight clusters with minimal effort.

Which of the following should recommend to the IT team to implement?

- **Azure Databricks**
- **Azure PowerAutomate**
- **Azure Traffic Manager**
- **Azure Resource Manager templates**

  (Correct)

Explanation

A Resource Manager template makes it easy to create the following resources for your application in a single, coordinated operation:

• HDInsight clusters and their dependent resources (such as the default storage account).

• Other resources (such as Azure SQL Database to use Apache Sqoop).

In the template, you define the resources that are needed for the application. You also specify deployment parameters to input values for different environments.

The template consists of JSON and expressions that you use to construct values for your deployment.

https://docs.microsoft.com/en-us/azure/hdinsight/hdinsight-hadoop-create-linux-clusters-arm-templates

Question 109: **Incorrect**
Identify the missing word(s) in the following sentence within the context of Microsoft Azure.

Azure Synapse Link for SQL is a *hybrid transactional / analytical processing* (HTAP) capability in Azure Synapse Analytics that you can use to synchronize transactional data in [?] or [?] with a dedicated SQL pool in Azure Synapse Analytics.

This synchronization enables you to perform near real-time analytical workloads on operational data with minimal impact on the transactional store used by business applications. (Select two)

- ☐

  Azure Database for MariaDB

- ☐

  Azure Database for MySQL

- ☐

  Microsoft SQL Server

  (Correct)

- ☐

Azure Database for PostgreSQL

- ☑

  Azure SQL Database

  (Correct)

Explanation
*Azure Synapse Link for SQL is a hybrid transactional / analytical processing (HTAP) capability in Azure Synapse Analytics that you can use to synchronize transactional data in **Azure SQL Database** or **Microsoft SQL Server** with a dedicated SQL pool in Azure Synapse Analytics.*

This synchronization enables you to perform near real-time analytical workloads on operational data with minimal impact on the transactional store used by business applications.

What is Azure Synapse Link for SQL?

Many organizations use a relational database in Azure SQL Database or Microsoft SQL Server to support business applications. These databases are optimized for transactional workloads that store and manipulate operational data. Performing analytical queries on the data in these databases to support reporting and data analysis incurs resource contention that can be detrimental to application performance.

A traditional approach to resolving this problem is to implement an *extract, transform, and load* (ETL) solution that loads data from the operational data store into an analytical store as a batch operation at regular intervals. While this solution supports the analytical workloads required for reporting and data analysis, it suffers from the following limitations:

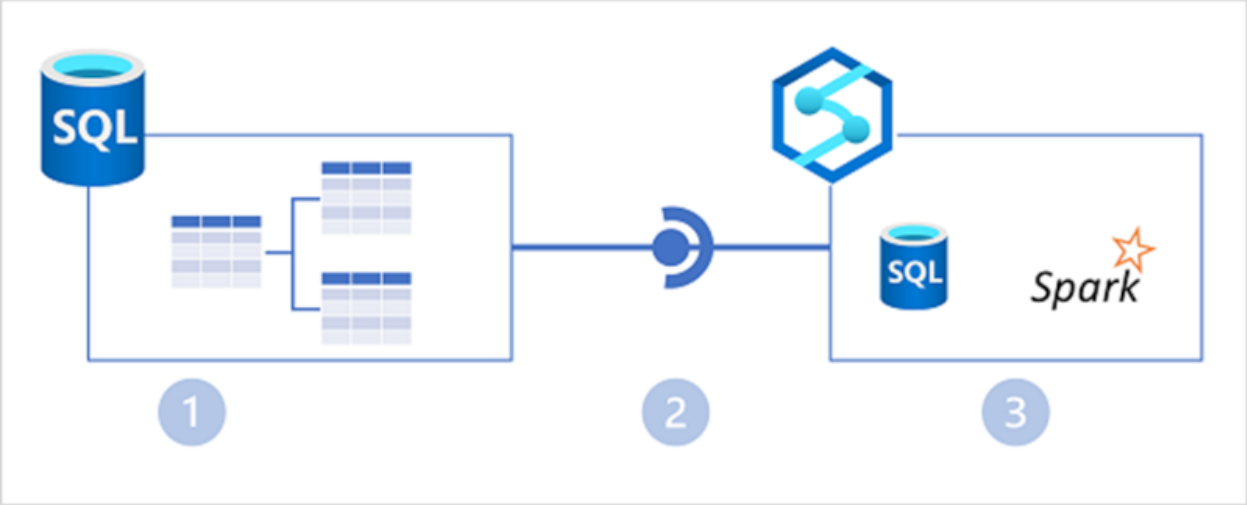
• The ETL process can be complex to implement and operate.

• The analytical store is only updated at periodic intervals, so reporting doesn't reflect the most up-to-date operational data.

Azure Synapse Link for SQL

Azure Synapse Link for SQL addresses the limitations of a traditional ETL process by automatically replicating changes made to tables in the operational database to corresponding tables in an analytical database. After the initial synchronization process,

the changes are replicated in near real-time without the need for a complex ETL batch process.



In the diagram above, the following key features of the Azure Synapse Link for SQL architecture are illustrated:

1. An Azure SQL Database or SQL Server 2022 instance contains a relational database in which transactional data is stored in tables.

2. Azure Synapse Link for SQL replicates the table data to a dedicated SQL pool in an Azure Synapse workspace.

3. The replicated data in the dedicated SQL pool can be queried in the dedicated SQL pool or connected to as an external source from a Spark pool without impacting the source database.

Source and target databases

Azure Synapse Link for SQL supports the following source databases (used as operational data stores):

• Azure SQL Database

• Microsoft SQL Server 2022

Azure Synapse link for SQL is not supported for Azure SQL Managed Instance.

The target database (used as an analytical data store) must be a dedicated SQL pool in an Azure Synapse Analytics workspace.

The implementation details for Azure Synapse Link vary between the two types of data source, but the high-level principle is the same - changes made to tables in the source database are synchronized to the target database.

Change feed

Azure Synapse Link for SQL uses the *change feed* feature in Azure SQL Database and Microsoft SQL Server 2022 to capture changes to the source tables. All data modifications are recorded in the transaction log for the source database. The change feed feature monitors the log and applies the same data modifications in the target database. In the case of Azure SQL Database, the modifications are made directly to the target database. When using Azure Synapse Link for SQL Server, the changes are recorded in files and saved to a *landing zone* in Azure Data Lake Gen2 storage before being applied to the target database.

Change feed is similar to the change data capture (CDC) feature in SQL Server. The key difference is that CDC is used to reproduce data modifications in a table in the same database as the modified table. Change feed caches the data modification in memory and forwards it to Azure Synapse Analytics.

After implementing Azure Synapse Link for SQL, you can use system views and stored procedures in your Azure SQL Database or SQL Server database to monitor and manage change feed activity.

https://learn.microsoft.com/en-us/sql/sql-server/synapse-link/synapse-link-sql-server-change-feed

https://learn.microsoft.com/en-us/sql/sql-server/synapse-link/synapse-link-sql-server-change-feed-manage

Question 110: **Correct**

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

Security administrators can control data access by using [?] within Data Lake Storage. Built-in security groups include `ReadOnlyUsers`, `WriteAccessUsers`, and `FullAccessUsers`.

- **AD Desired State Configuration (ADDSC)**
- **AD OAuth**
- **Active Directory Application Groups**
- **Active Directory Security Groups**

  (Correct)

Explanation
Data Lake Storage Data Security

Because Data Lake Storage supports Azure Active Directory ACLs, security administrators can control data access by using the familiar Active Directory Security Groups. Role-based access control (RBAC) is available both in Gen1 and Gen2. Built-in security groups include `ReadOnlyUsers`, `WriteAccessUsers`, and `FullAccessUsers`.

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