Question 1: **Incorrect**
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 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 this for a Spark cluster. This is created for each cluster when it's created, but you can configure a cluster to use an existing one if necessary"*?

- **Databricks File System**

  (Incorrect)

- **Apache Spark clusters**

- **Hive metastore**

  (Correct)

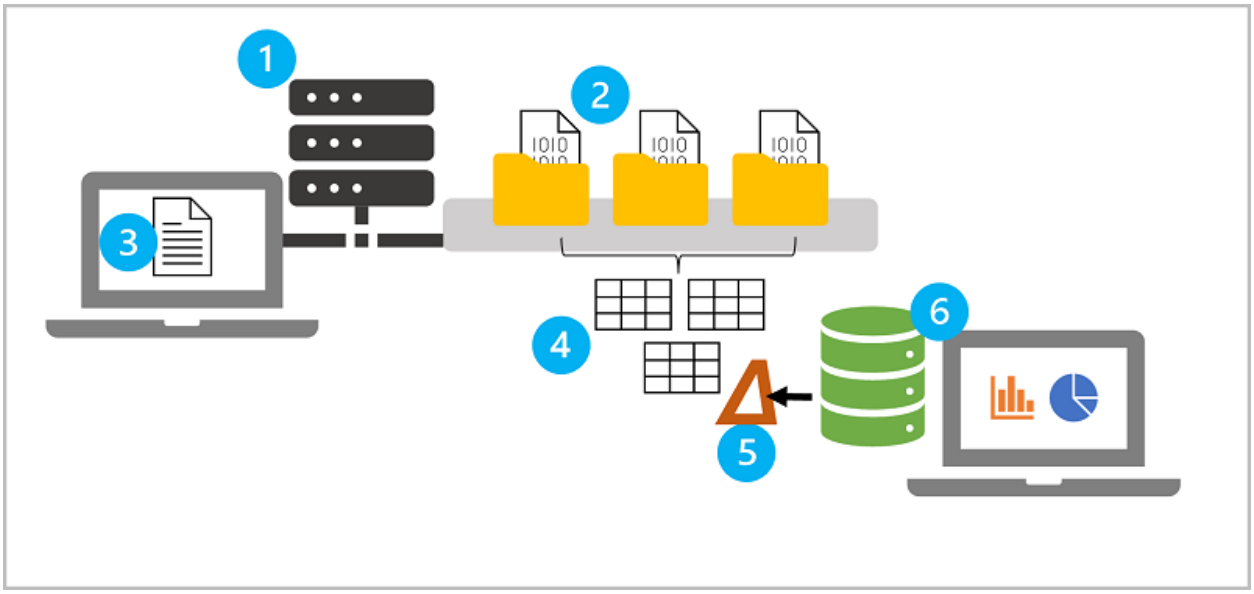
- **Delta Lake**

- **SQL Warehouses**

- **Notebook**

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 2: **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 the analytical processing, ingesting, synthesizing, and managing large sets of historical data.

-

ETL

- ○

  ADPS

- ○

  OLAP

  (Correct)

- ○

  OLTP

- ◉

  ELT

  (Incorrect)

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.

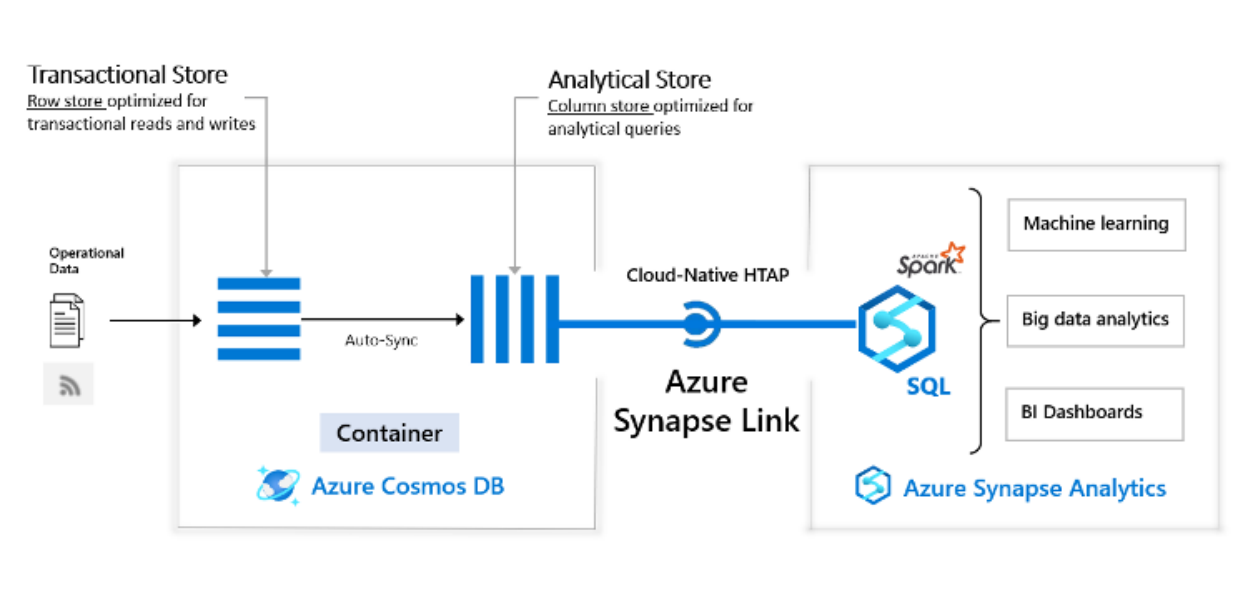
OLAP systems are optimized for the analytical processing, ingesting, synthesizing, and managing large sets of historical data. The data processed by OLAP systems largely originates from OLTP systems and needs to be loaded into the OLTP systems by means of batch processes commonly referred to as ETL (Extract, 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 3: **Incorrect**

Within Azure Synapse Link for Azure Cosmos DB, which Column-oriented store is optimized for queries?

- ○

  Transactional store

- ○

  Analytical store

  (Correct)

- ○

  Query store

- ◉

  Cosmos DB store

  (Incorrect)

Explanation
An analytical store is a data store optimized for analytical queries.

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

Question 4: **Incorrect**
In the context of analytics, data streams are event data generated by sensors or other sources that can be analyzed by another technology. Analyzing a data stream is typically done to measure the state change of a component or to capture information on an area of interest.

Which are approaches to processing data streams? (Select two)

- ☐

  Near real time

- ☐

  On-demand

(Correct)

- ☐

Live

(Correct)

- ☐

All the listed options

- ☑

Multiprocessing

(Incorrect)

Explanation
In the context of analytics, data streams are event data generated by sensors or other sources that can be analyzed by another technology. Analyzing a data stream is typically done to measure the state change of a component or to capture information on an area of interest. The intent being to:

• Continuously analyze data to detect issues and understand or respond to them.

• Understand component or system behaviour under various conditions to fuel further enhancements of said component or system.

•Trigger specific actions when certain thresholds are identified.

In today's world, data streams are ubiquitous. Companies can harness the latent knowledge in data streams to improve efficiencies and further innovation. Examples of use cases that analyze data streams include:

• Stock market trends.

• Monitoring data of water pipelines and electrical transmission and distribution systems by utility companies.

• Mechanical component health monitoring data in automotive and automobile industries.

• Monitoring data from industrial and manufacturing equipment.

• Sensor data in transportation, such as traffic management and highway toll lanes.

• Patient health monitoring data in the healthcare industry.

• Satellite data in the space industry.

• Fraud detection in the banking and finance industries.

• Sentiment analysis of social media posts.

Approaches to data stream processing

There are two approaches to processing data streams: on-demand and live.

Streaming data can be collected over time and persisted in storage as static data. The data can then be processed when convenient or during times when compute costs are lower. The downside to this approach is the cost of storing the data.

In contrast, live data streams have relatively low storage requirements. They also require more processing power to run computations in sliding windows over continuously incoming data to generate the insights.

https://www.simplilearn.com/what-is-data-processing-article

Question 5: **Incorrect**
Identify the missing word(s) in the following sentence within the context of Microsoft Azure.

In Data Lake Storage Gen1, data engineers query data by using the [?] language.

- **U-SQL**

  (Correct)

- **ADLS API**

- **ABS API**

- **T-SQL**
- **M-SQL**

  (Incorrect)

Explanation
Data Lake Storage Queries

In Data Lake Storage Gen1, data engineers query data by using the U-SQL language. U-SQL is a language that combines declarative SQL with imperative C# to let you process data at any scale. Through the scalable, distributed-query capability of U-SQL, you can efficiently analyze data across relational stores such as Azure SQL Database. With U-SQL, you can process unstructured data by applying schema on read and inserting custom logic and UDFs. Additionally, U-SQL includes extensibility that gives you fine-grained control over how to execute at scale.

In Gen 2, use the Azure Blob Storage API or the Azure Data Lake System (ADLS) API.

https://docs.microsoft.com/en-us/azure/data-lake-analytics/data-lake-analytics-u-sql-get-started

Question 6: **Incorrect**
Conditional access is a feature that enables you to define the conditions under which a user can connect to your Azure subscription and access services. Conditional access provides an additional layer of security that can be used in combination with authentication to strengthen the security access to your network.

Conditional Access policies at their simplest are [?].

- **If-then statements**

  (Correct)
- **If-else statements**

- ○

  While-if statements

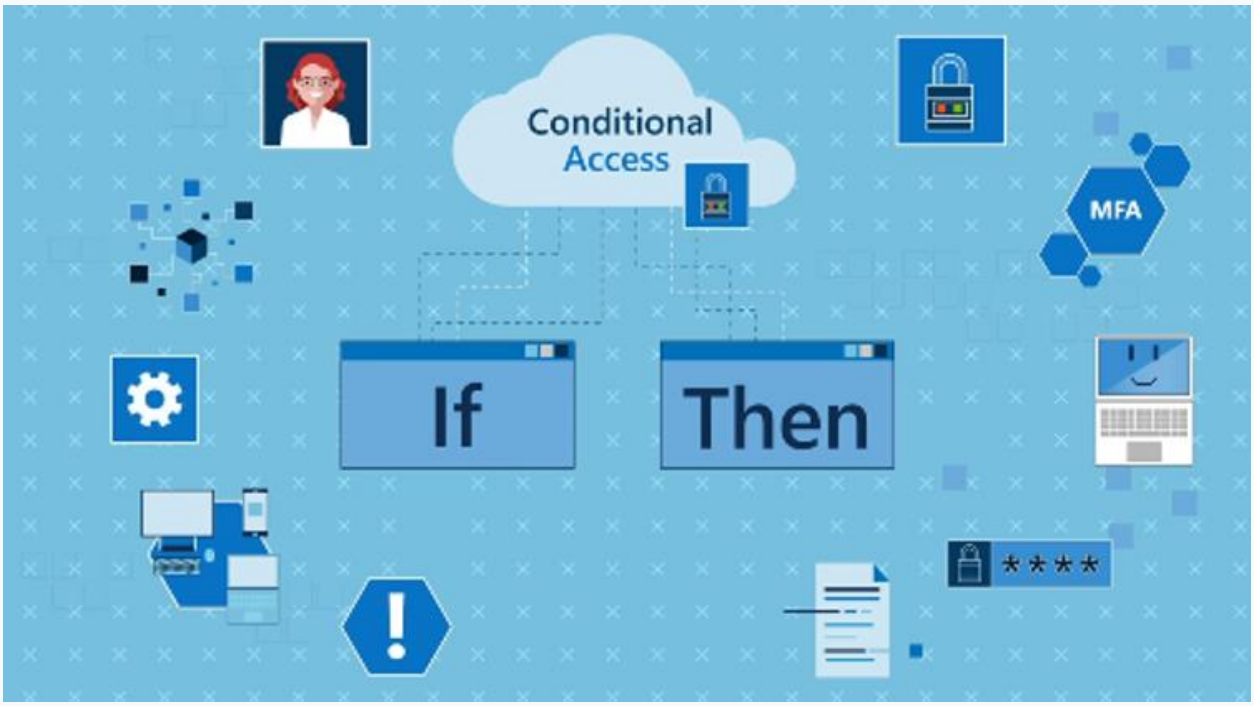
- ◉

  Where-having statements

  (Incorrect)

Explanation
Conditional access is a feature that enables you to define the conditions under which a user can connect to your Azure subscription and access services. Conditional access provides an additional layer of security that can be used in combination with authentication to strengthen the security access to your network.

Conditional Access policies at their simplest are if-then statements, if a user wants to access a resource, then they must complete an action. As an example, if a Data Engineer wishes to access services in Azure Synapse Analytics, they may be requested by the conditional access policy to perform an additional step of multi-factor authentication (MFA) to complete the authentication to get onto the service.



Conditional access policies use signals as a basis to determine if conditional access should first be applied. Common signals include:

• User or group membership names

• IP address information

• Device platforms or type

• Application access requests

• Real-time and calculated risk detection

• Microsoft Cloud App Security (MCAS)

Based on these signals, you can then choose to block access. The alternative is you can grant access, and at the same time request that the user perform an additional action including:

• Perform Multi-Factor authentication

• Use a specific device to connect

Given the amount of data that could potentially be stored, Azure Synapse Analytics dedicated SQL pools supports conditional access to provide protection for your data. It does require that Azure Synapse Analytics is configured to support Azure Active Directory, and that if you chose multi-factor authentication, that the tool you are using support it.

https://docs.microsoft.com/en-us/azure/azure-sql/database/conditional-access-configure

Question 7: **Correct**
How many access keys are provided for accessing your Azure storage account?

- **1**
- **1 per authorized user**
-

- **3**
- **4**
- **2**

  (Correct)

Explanation
Each storage account has two access keys. This lets you follow the best-practice guideline of periodically replacing the key used by your applications without incurring downtime.

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

Question 8: **Incorrect**
Usually when you see 'df' in some code it refers to a [?].

```
1. Python
2. new_rows = <additional code here>
3. demo_df = <additional code here>
```

- **Datafeature**
- **Dateformat**
- **Dataframe**

  (Correct)
- **Dataformat**

-

  Dataflow

  (Incorrect)

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 9: **Incorrect**
The SparkSQL library, which provides the dataframe structure also enables you to use SQL as a way of working with data. With this approach. You can query and transform data in dataframes by using SQL queries, and persist the results as tables.

What happens if you drop an external table that is based on existing files?

- **An error – you must delete the files first**
- **None of the provided options are correct**
- **The table is dropped from the metastore but the files remain unaffected**

  (Correct)
- **The table is dropped from the metastore and the files are deleted**

  (Incorrect)

Explanation

- *The tables are loosely coupled from the files allowing the table to be dropped while the files remain.*
- The files and table are loosely coupled, deleting the files doesn't remove the table from the metastore and dropping the table has no impact on the existing files.

Transform data with SQL

The SparkSQL library, which provides the dataframe structure also enables you to use SQL as a way of working with data. With this approach. You can query and transform data in dataframes by using SQL queries, and persist the results as tables.

Note: Tables are metadata abstractions over files. The data is not stored in a relational table, but the table provides a relational layer over files in the data lake.

Define tables and views

Table definitions in Spark are stored in the *metastore*, a metadata layer that encapsulates relational abstractions over files. *External* tables are relational tables in the metastore that reference files in a data lake location that you specify. You can access this data by querying the table or by reading the files directly from the data lake.

Note: External tables are "loosely bound" to the underlying files and deleting the table does not delete the files. This allows you to use Spark to do the heavy lifting of transformation then persist the data in the lake. After this is done you can drop the table and downstream processes can access these optimized structures. You can also define managed tables, for which the underlying data files are stored in an internally manages storage location associated with the metastore. Manages tables are "tightly-bound" to the files, and dropping a managed table deletes the associated files.

The following code example saves a dataframe (loaded from CSV files) as an external table name **sales_orders**. The files are stored in the **/sales_orders_table** folder in the data lake.

```
1. Python
2. order_details.write.saveAsTable('sales_orders', format='parquet',
   mode='overwrite', path='/sales_orders_table')
```

Use SQL to query and transform the data

After defining a table, you can use of SQL to query and transform its data. The following code creates two new derived columns named **Year** and **Month** and then creates a new table *transformed_orders* with the new derived columns added.

```
1. Python
```

```
2. # Create derived columns
3. sql_transform = spark.sql("SELECT *, YEAR(OrderDate) AS Year, MONTH(OrderDate)
   AS Month FROM sales_orders")
4.
5. # Save the results
6. sql_transform.write.partitionBy("Year","Month").saveAsTable('transformed_order
   s', format='parquet', mode='overwrite', path='/transformed_orders_table')
```

The data files for the new table are stored in a hierarchy of folders with the format of **Year=*NNNN* / Month=*N***, with each folder containing a parquet file for the corresponding orders by year and month.

Query the metastore

Because this new table was created in the metastore, you can use SQL to query it directly with the *%%sql* magic key in the first line to indicate that the SQL syntax will be used as shown in the following script:

```
1. SQL
2. %%sql
3.
4. SELECT * FROM transformed_orders
5. WHERE Year = 2021
6.     AND Month = 1
```

Drop tables

When working with external tables, you can use the `DROP` command to delete the table definitions from the metastore without affecting the files in the data lake. This approach enables you to clean up the metastore after using SQL to transform the data, while making the transformed data files available to downstream data analysis and ingestion processes.

```
1. SQL
2. %%sql
3.
4. DROP TABLE transformed_orders;
5. DROP TABLE sales_orders;
```

https://spark.apache.org/docs/latest/sql-programming-guide.html

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

Apache Spark Structured Streaming is a fast, scalable, and fault-tolerant stream processing API. You can use it to perform analytics on your streaming data in [?].

- **Historic batches**
- **Real-time**
- **Prediction mode**
- **Near real-time**

  (Correct)

Explanation
Apache Spark Structured Streaming is a fast, scalable, and fault-tolerant stream processing API. You can use it to perform analytics on your streaming data in near real time.

With Structured Streaming, you can use SQL queries to process streaming data in the same way that you would process static data. The API continuously increments and updates the final data.

Event Hubs and Spark Structured Streaming

Azure Event Hubs is a scalable real-time data ingestion service that processes millions of data in a matter of seconds. It can receive large amounts of data from multiple sources and stream the prepared data to Azure Data Lake or Azure Blob storage.

Azure Event Hubs can be integrated with Spark Structured Streaming to perform processing of messages in near real time. You can query and analyze the processed data as it comes by using a Structured Streaming query and Spark SQL.

Streaming concepts

Stream processing is where you continuously incorporate new data into Data Lake storage and compute results. The streaming data comes in faster than it can be consumed when using traditional batch-related processing techniques. A stream of data is treated as a table to which data is continuously appended. Examples of such data include bank card transactions, Internet of Things (IoT) device data, and video game play events.

A streaming system consists of:

• Input sources such as Kafka, Azure Event Hubs, IoT Hub, files on a distributed system, or TCP-IP sockets

• Stream processing using Structured Streaming, forEach sinks, memory sinks, etc.

https://spark.apache.org/docs/latest/structured-streaming-programming-guide.html

Question 11: **Incorrect**
Azure Data factory can accommodate organizations that are embarking on data integration projects from differing starting point. Typically, many data integration workflows must consider existing pipelines that have been created on previous projects, with different dependencies and using different technologies.

Which of the following are ingestion methods that can be used to extract data from a variety of sources?

- ☐

  Datasets

- ☐

  Linked Services

- ☐

  Copy Activity

  (Correct)

- ☐

  SSIS packages

  (Correct)

- ☑

  Compute resources

  (Correct)

- ☐

  Self-hosted

- ☐

  Activities

Explanation

Azure Data factory can accommodate organizations that are embarking on data integration projects from differing starting point. It is rare for a data migration project to be a green field project. Typically, many data integration workflows must consider existing pipelines that have been created on previous projects, with different dependencies and using different technologies. To that end, there are a variety of ingestion methods that can be used to extract data from a variety of sources.

Ingesting data using the Copy Activity

Use this method to build code free data ingestion pipelines that don't require any transformation during the extraction of the data. The Copy Activity has support for over 100 native connectors. This method can suit green field projects that have a simple method of extraction to an intermediary data store. An example of ingesting data using the Copy Activity can include extracting data from multiple source database systems and outputting the data to files in a data lake store. The benefit of this ingestion method is that they are simple to create, but they are not able to deal with sophisticated transformations or business logic.

Ingesting data using compute resources

Azure Data Factory can call on compute resources to process data by a data platform service that may be better suited to the job. A great example of this is that Azure Data Factory can create a pipeline to an analytical data platform such as Spark pools in an Azure Synapse Analytics instance to perform a complex calculation, which generates

new data. This data is then ingested back into the pipeline for further downstream processing. There a wide range of compute resource, and the associated activities that they can perform as shown in the following:

Compute environment: On-demand HDInsight cluster or your own HDInsight cluster

Activities: Hive, Pig, Spark, MapReduce, Hadoop Streaming

Compute environment: Azure Batch

Activities: Custom activities

Compute environment: Azure Machine Learning Studio Machine

Activities: Learning activities: Batch Execution and Update Resource

Compute environment: Azure Machine Learning

Activities: Azure Machine Learning Execute Pipeline

Compute environment: Azure Data Lake Analytics

Activities: Data Lake Analytics U-SQL

Compute environment: Azure SQL, Azure SQL Data Warehouse, SQL Server

Activities: Stored Procedure

Compute environment: Azure Databricks

Activities: Notebook, Jar, Python

Compute environment: Azure Function

Activities: Azure Function activity

Ingesting data using SSIS packages

Many organizations have decades of development investment in SQL Server Integration Services (SSIS) packages that contain both ingestion and transformation logic from on-premises and cloud data stores. Azure Data Factory provides the ability to lift and shift existing SSIS workload, by creating an Azure-SSIS Integration Runtime to natively execute SSIS packages, and will enable you to deploy and manage your existing SSIS packages with little to no change using familiar tools such as SQL Server Data Tools (SSDT) and SQL Server Management Studio (SSMS), just like using SSIS on premises.

https://docs.microsoft.com/en-us/azure/machine-learning/how-to-data-ingest-adf

Question 12: **Incorrect**
Which `ALTER DATABASE` statement parameter allows a dedicated SQL pool to scale?

- `OVER`
- `CHANGE`
- `MODIFY`

  (Correct)
- `SCALE`

  (Incorrect)

Explanation
`MODIFY` is used to scale a dedicated SQL pool.

In the following example, we use the ALTER DATABASE T-SQL statement to modify the service objective. Run the following query to change the service objective to DW300.

```
SQL
ALTER DATABASE mySampleDataWarehouse
MODIFY (SERVICE_OBJECTIVE = 'DW300c');
```

https://docs.microsoft.com/en-us/azure/synapse-analytics/sql-data-warehouse/quickstart-scale-compute-tsql

Question 13: **Incorrect**

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

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

Descriptive analytics answers the question [?].

- **What is likely to happen in the future based on previous trends and patterns?"**
- **What is happening in my business?**

  (Correct)
- **Why is it happening?**
- **When will the modification made meet my goals?**

  (Incorrect)

Explanation

Azure Synapse Analytics is an integrated analytics platform, which combines data warehousing, big data analytics, data integration, and visualization into a single environment. Azure Synapse Analytics empowers users of all abilities to gain access

and quick insights across all of their data, enabling a whole new level of performance and scale.

The range of analytical types that Azure Synapse Analytics can support include:

Descriptive analytics

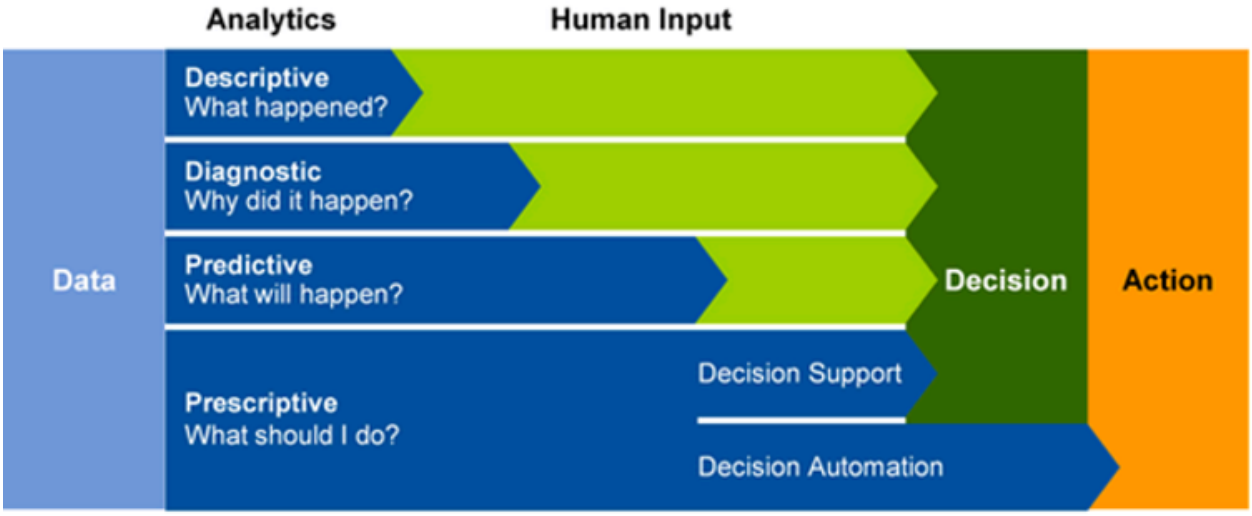
Descriptive analytics answers the question “What is happening in my business?” The data to answer this question is typically answered through the creation of a data warehouse. Azure Synapse Analytics leverages the dedicated SQL Pool capability that enables you to create a persisted data warehouse to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

Diagnostic analytics

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

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

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



Predictive analytics

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

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

Prescriptive analytics

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

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

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

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

Question 14: **Incorrect**
Identify the missing word(s) in the following sentence within the context of Microsoft Azure.

[?] 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.

- **Azure Databricks**

(Correct)

- **Apache Kafka**

- **Apache Spark**

- **Azure Event Hub**

  (Incorrect)

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 the team that started Apache Spark 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.

Optimized environment

To address the problems seen on other Big Data platforms, Azure Databricks was optimized from the ground up, with a focus on performance and cost-efficiency in the cloud. The Databricks Runtime adds several key capabilities to Apache Spark workloads that can increase performance and reduce costs by as much as 10-100x when running on Azure, including:

• High-speed connectors to Azure storage services, such as Azure Blob Store and Azure Data Lake

• Auto-scaling and auto-termination of Spark clusters to minimize costs

• Caching

• Indexing

• Advanced query optimization

By providing an optimized, easy to provision and configure environment, Azure Databricks gives developers a performant, cost-effective platform that enables them to spend more time building applications, and less time focused on managing clusters and infrastructure.

Who is Databricks?

Databricks was founded by the creators of Apache Spark, Delta Lake, and MLflow.

Over 2000 global companies use the Databricks platform across big data & machine learning lifecycle.

Databricks Vision: Accelerate innovation by unifying data science, data engineering and business.

Databricks Solution: Big Data Analytics Platform

What does Databricks offer that is not Open-Source Spark?

• Databricks Workspace - Interactive Data Science & Collaboration

• Databricks Workflows - Production Jobs & Workflow Automation

• Databricks Runtime

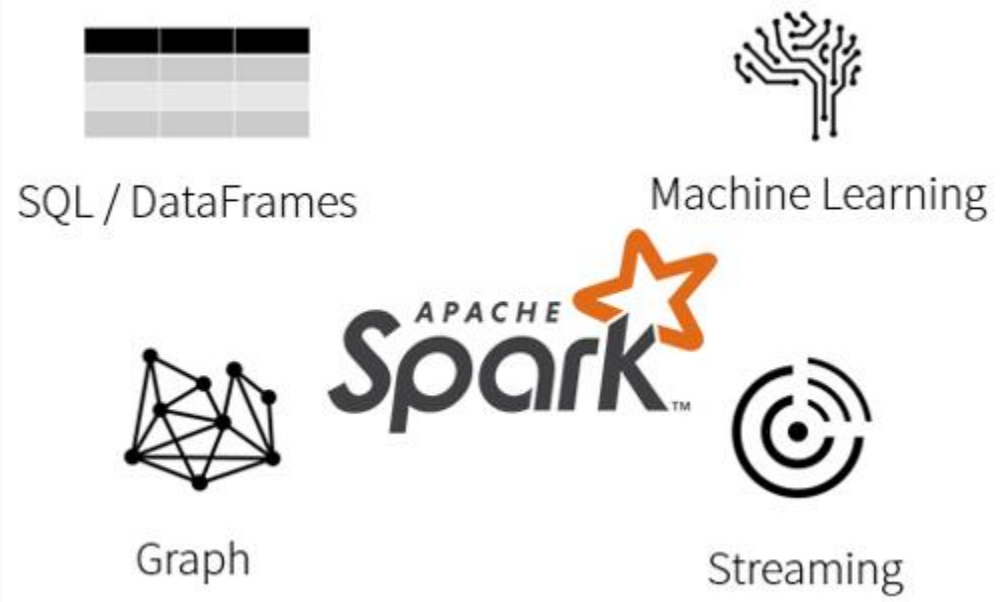
• Databricks I/O (DBIO) - Optimized Data Access Layer

• Databricks Serverless - Fully Managed Auto-Tuning Platform

• Databricks Enterprise Security (DBES) - End-To-End Security & Compliance

What is Apache Spark?

Spark is a unified processing engine that can analyze big data using SQL, machine learning, graph processing, or real-time stream analysis:



• At its core is the Spark Engine.

• The DataFrames API provides an abstraction above RDDs while simultaneously improving performance 5-20x over traditional RDDs with its Catalyst Optimizer.

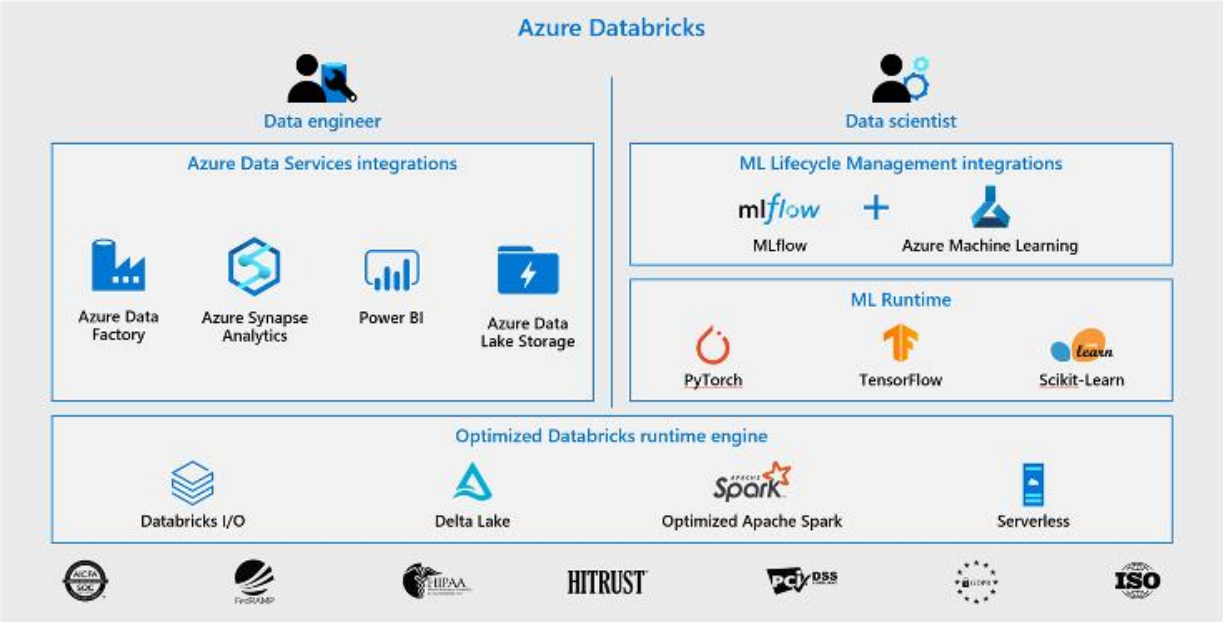
• Spark ML provides high quality and finely tuned machine learning algorithms for processing big data.

• The Graph processing API gives us an easily approachable API for modelling pairwise relationships between people, objects, or nodes in a network.

• The Streaming APIs give us End-to-End Fault Tolerance, with Exactly-Once semantics, and the possibility for sub-millisecond latency.

And it all works together seamlessly!

Azure Databricks



As a compute engine, Azure Databricks sits at the centre of your Azure-based software platform and provides native integration with Azure Active Directory (Azure AD) and other Azure services.

Scala, Python, Java, R & SQL

• Besides being able to run in many environments, Apache Spark makes the platform even more approachable by supporting multiple languages:

• Scala - Apache Spark's primary language

• Python - More commonly referred to as PySpark

• R - SparkR (R on Spark)

• Java

• SQL - Closer to ANSI SQL 2003 compliance

  • Now running all 99 TPC-DS queries

  • New standards-compliant parser (with good error messages!)

  • Subqueries (correlated & uncorrelated)

  • Approximate aggregate stats

• With the DataFrames API, the performance differences between languages are nearly nonexistence (especially for Scala, Java & Python).

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

Question 15: **Incorrect**

Scenario: O'Shaughnessy's is a fast food restaurant. The chain has stores nationwide and is rivalled by Big Belly Burgers. You have been hired by the company to advise on working with Microsoft Azure Synapse Analytics.

At the moment, you are leading a meeting where the topic at hand is designing an enterprise data warehouse.

The IT team at O'Shaughnessy's is working on a project to design and create an enterprise data warehouse in Azure Synapse Analytics which will contain a table named Customers. Customers will contain credit card information.

Because security is critical to O'Shaughnessy's, they have asked you to recommend a solution to provide salespeople with the ability to view all the entries in Customers and best reduce the risk of salespeople viewing or inferring the credit card information.

Which of the following techniques should you propose in your recommendation?

- **Row-level security**

  (Incorrect)

- **Cross-level security**

- **Always Encrypted**

- **Data masking**

  (Correct)

Explanation
You should propose the use of Data masking in your recommendation because SQL Database dynamic data masking limits sensitive data exposure by masking it to non-privileged users.

The Credit card masking method exposes the last four digits of the designated fields and adds a constant string as a prefix in the form of a credit card.

Example: XXXX-XXXX-XXXX-1234 -

Security Note: Bypassing masking using inference or brute-force techniques

Dynamic Data Masking is designed to simplify application development by limiting data exposure in a set of pre-defined queries used by the application. While Dynamic Data Masking can also be useful to prevent accidental exposure of sensitive data when accessing a production database directly, it is important to note that unprivileged users with ad-hoc query permissions can apply techniques to gain access to the actual data. If there is a need to grant such ad-hoc access, Auditing should be used to monitor all database activity and mitigate this scenario.

Dynamic data masking does not prevent inferring the data. It would take a while to infer Credit Card numbers, but it could be done.

https://docs.microsoft.com/en-us/sql/relational-databases/security/dynamic-data-masking?view=sql-server-ver15

Monitor and optimize data storage and data processing

Azure SQL Database, Azure SQL Managed Instance, and Azure Synapse Analytics support dynamic data masking. Dynamic data masking limits sensitive data exposure by masking it to non-privileged users.

Dynamic data masking helps prevent unauthorized access to sensitive data by enabling customers to designate how much of the sensitive data to reveal with minimal impact on the application layer. It’s a policy-based security feature that hides the sensitive data in the result set of a query over designated database fields, while the data in the database is not changed.

For example, a service representative at a call centre might identify a caller by confirming several characters of their email address, but the complete email address shouldn't be revealed to the service representative. A masking rule can be defined that masks all the email address in the result set of any query. As another example, an appropriate data mask can be defined to protect personal data, so that a developer can

query production environments for troubleshooting purposes without violating compliance regulations.

Dynamic data masking basics

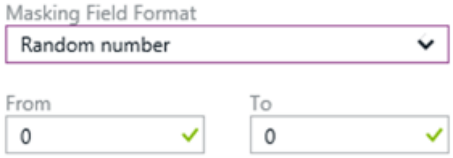
You set up a dynamic data masking policy in the Azure portal by selecting the **Dynamic Data Masking** blade under **Security** in your SQL Database configuration pane. This feature cannot be set using portal for SQL Managed Instance. For more information, see Dynamic Data Masking.

Dynamic data masking policy

SQL users excluded from masking - A set of SQL users or Azure AD identities that 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 - A set of rules that define the designated fields to be masked and the masking function that is used. The designated fields can be defined using a database schema name, table name, and column name.

Masking functions - A set of methods that control the exposure of data for different scenarios.

Masking function	Masking logic
Default	**Full masking according to the data types of the designated fields** • Use XXXX or fewer Xs if the size of the field is less than 4 characters for string data types (nchar, ntext, nvarchar). • Use a zero value for numeric data types (bigint, bit, decimal, int, money, numeric, smallint, smallmoney, tinyint, float, real). • Use 01-01-1900 for date/time data types (date, datetime2, datetime, datetimeoffset, smalldatetime, time). • For SQL variant, the default value of the current type is used. • For XML the document <masked/> is used. • Use an empty value for special data types (timestamp table, hierarchyid, GUID, binary, image, varbinary spatial types).
Credit card	**Masking method, which exposes the last four digits of the designated fields** and adds a constant string as a prefix in the form of a credit card. XXXX-XXXX-XXXX-1234
Email	**Masking method, which exposes the first letter and replaces the domain with XXX.com** using a constant string prefix in the form of an email address. aXX@XXXX.com
Random number	**Masking method, which generates a random number** according to the selected boundaries and actual data types. If the designated boundaries are equal, then the masking function is a constant number. 
Custom text	**Masking method, which exposes the first and last characters** and adds a custom padding string in the middle. If the original string is shorter than the exposed prefix and suffix, only the padding string is used. prefix[padding]suffix

Recommended fields to mask

The DDM recommendations engine, flags certain fields from your database as potentially sensitive fields, which may be good candidates for masking. In the Dynamic Data Masking blade in the portal, you will see the recommended columns for your database. All you need to do is click **Add Mask** for one or more columns and then **Save** to apply a mask for these fields.

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

Question 16: **Correct**
Which of the below have the following characteristics?

• Permit us to store data in a format that more closely meets the original structure.

• Does not use the tabular schema of columns and rows found in most traditional database systems.

• Uses a storage model that is enhanced for the specific requirements of the type of data being stored.

- ○

  Binary

- ○

  Structured

- ○

  Relational

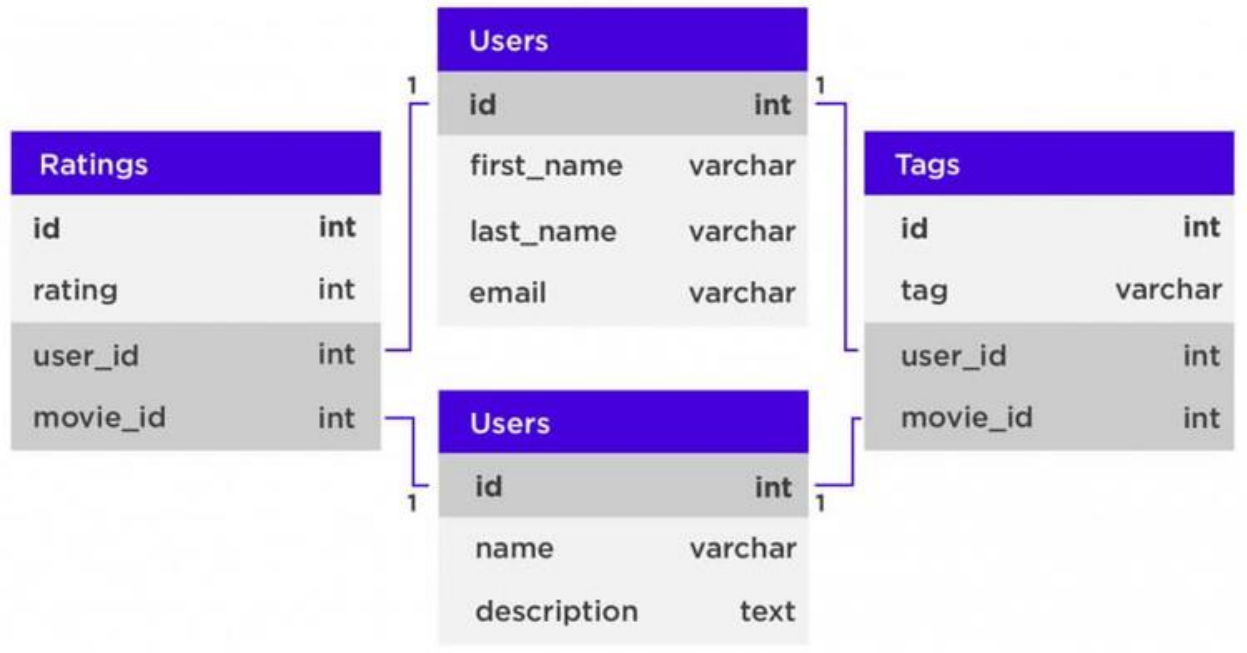
- ◉

  Non-Relational

  (Correct)

Explanation
Non-Relational Data

• Non-relational databases permit us to store data in a format that more closely meets the original structure.



• A ***non-relational database*** is a database that does not use the tabular schema of columns and rows found in most traditional database systems.

• It uses a storage model that is enhanced for the specific requirements of the type of data being stored.

• In a non-relational database the data may be stored as **JSON documents**, as simple **key/value pairs**, or as a **graph** consisting of edges and vertices.

• Examples of relational databases:

• Redis

• JanusGraph

• MongoDB

• RabbitMQ

https://f5a395285c.nxcli.net/microsoft-azure/dp-900/structured-data-vs-unstructured-data-vs-semi-structured-data/

Question 17: **Incorrect**
Azure provides many ways to store your data. An Azure storage account contains all of your Azure Storage data objects, including blobs, file shares, queues, tables, and disks. The storage account provides a unique namespace for your Azure Storage data that's accessible from anywhere in the world over HTTP or HTTPS. Data in your storage account is durable and highly available, secure, and massively scalable.

Azure Storage offers several types of storage accounts. Each type supports different features and has its own pricing model. Which of the following are valid types of storage accounts? (Select four)

- ☐

  Append blobs Storage

- ☐

  Premium File Share

  (Correct)

- ☐

  Premium Block

  (Correct)

- ☐

  General Purpose v1

- ☑

  Page blobs Storage

  (Incorrect)

- ☐

  General Purpose v2

  (Correct)

- ☐

  Premium Page Blob

  (Correct)

- ☐

  Block blobs Storage

- ☐

  Container Storage

- ☐

  Data Lake Storage

- ☐

  Classic Storage

-

  Data Pool Storage

Explanation

Storage Account: General Purpose V1 and Standard Blob are now legacy. Now we have General Purpose V2, Premium Block, Premium File Share and Premium Page Blob.

An Azure storage account contains all of your Azure Storage data objects, including blobs, file shares, queues, tables, and disks. The storage account provides a unique namespace for your Azure Storage data that's accessible from anywhere in the world over HTTP or HTTPS. Data in your storage account is durable and highly available, secure, and massively scalable.

Types of storage accounts

Azure Storage offers several types of storage accounts. Each type supports different features and has its own pricing model.

The following table describes the types of storage accounts recommended by Microsoft for most scenarios. All of these use the Azure Resource Manager deployment model.

Type of storage account	Supported storage services	Redundancy options	Usage
Standard general-purpose v2	Blob Storage (including Data Lake Storage[1]), Queue Storage, Table Storage, and Azure Files	Locally redundant storage (LRS) / geo-redundant storage (GRS) / read-access geo-redundant storage (RA-GRS) Zone-redundant storage (ZRS) / geo-zone-redundant storage (GZRS) / read-access geo-zone-redundant storage (RA-GZRS)[2]	Standard storage account type for blobs, file shares, queues, and tables. Recommended for most scenarios using Azure Storage. If you want support for network file system (NFS) in Azure Files, use the premium file shares account type.
Premium block blobs[3]	Blob Storage (including Data Lake Storage[1])	LRS ZRS[2]	Premium storage account type for block blobs and append blobs. Recommended for scenarios with high transactions rates, or scenarios that use smaller objects or require consistently low storage latency. Learn more about example workloads.
Premium file shares[3]	Azure Files	LRS ZRS[2]	Premium storage account type for file shares only. Recommended for enterprise or high-performance scale applications. Use this account type if you want a storage account that supports both Server Message Block (SMB) and NFS file shares.
Premium page blobs[3]	Page blobs only	LRS	Premium storage account type for page blobs only. Learn more about page blobs and sample use cases.

1 Data Lake Storage is a set of capabilities dedicated to big data analytics, built on Azure Blob storage. For more information, see Introduction to Data Lake Storage Gen2 and Create a storage account to use with Data Lake Storage Gen2.

2 ZRS, GZRS, and RA-GZRS are available only for standard general-purpose v2, premium block blobs, and premium file shares accounts in certain regions. For more information, see Azure Storage redundancy.

3 Premium performance storage accounts use solid-state drives (SSDs) for low latency and high throughput.

Legacy storage accounts are also supported. For more information, see Legacy storage account types.

You can't change a storage account to a different type after it's' created. To move your data to a storage account of a different type, you must create a new account and copy the data to the new account.

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

Question 18: **Correct**
Identify the missing word(s) in the following sentence within the context of Microsoft Azure.

Synapse dedicated SQL Pools supports JSON format data to be stored using [?]. The JSON format enables representation of complex or hierarchical data structures in tables. It allows to transform arrays of JSON objects into table format. The performance of JSON data can be optimized by using columnstore indexes and memory optimized tables.

- **Standard `NCHAR` table columns**
- **Standard `CHAR` table columns**
-

Standard `VARCHAR` table columns

- **Standard `NVARCHAR` table columns**

  (Correct)

Explanation
Synapse dedicated SQL Pools supports JSON format data to be stored using **standard `NVARCHAR` table columns.** The JSON format enables representation of complex or hierarchical data structures in tables. It allows to transform arrays of JSON objects into table format. The performance of JSON data can be optimized by using columnstore indexes and memory optimized tables.

Insert JSON data - JSON data can be inserted using the usual T-SQL INSERT statements.

Read JSON data - JSON data can be read using the following T-SQL functions and provides the ability to perform aggregation and filter on JSON values.

• `ISJSON` – verify if text is valid JSON

• `JSON_VALUE` – extract a scalar value from a JSON string

• `JSON_QUERY` – extract a JSON object or array from a JSON string

Modify JSON data - JSON data can be modified and queried using the following T-SQL functions providing ability to update JSON string using T-SQL and convert hierarchical data into flat tabular structure.

• `JSON_MODIFY` – modifies a value in a JSON string

• `OPENJSON` – convert JSON collection to a set of rows and columns

You can also query JSON files using SQL serverless. The query's objective is to read the following type of JSON files using `OPENROWSET`.

• Standard JSON files where multiple JSON documents are stored as a JSON array.

• Line-delimited JSON files, where JSON documents are separated with new-line character. Common extensions for these types of files are `jsonl`, `ldjson`, and `ndjson`.

https://docs.microsoft.com/en-us/azure/synapse-analytics/sql/query-json-files

Question 19: **Incorrect**
Scenario: One of your teammates has just executed `GetBlockBlobReference` with the name of a blob.

What will happen?

- **A `CloudBlockBlob` object is created locally. No network calls are made.**

  (Correct)

- **The contents of the named blob are downloaded.**

- **An exception is thrown if the blob does not exist in storage.**

- **A new block blob is created in storage.**

  (Incorrect)

Explanation
Getting a blob reference does not make any calls to Azure Storage, it simply creates an object locally that can work with a stored blob.

https://docs.microsoft.com/en-us/dotnet/api/microsoft.azure.storage.blob.cloudblobcontainer.getblockblobreference?view=azure-dotnet-legacy

Question 20: **Incorrect**
Which of the following are facets of Azure Databricks security? (Select four)

- **Network**

  (Correct)

-

- ☐ Encryption
- ☐ VNet Peering
- ☐ Compliance

  (Correct)
- ☐ IAM/Auth

  (Correct)
- ☑ Vault

  (Incorrect)
- ☐ Load Balancing
- ☐ Data Protection

  (Correct)

Explanation

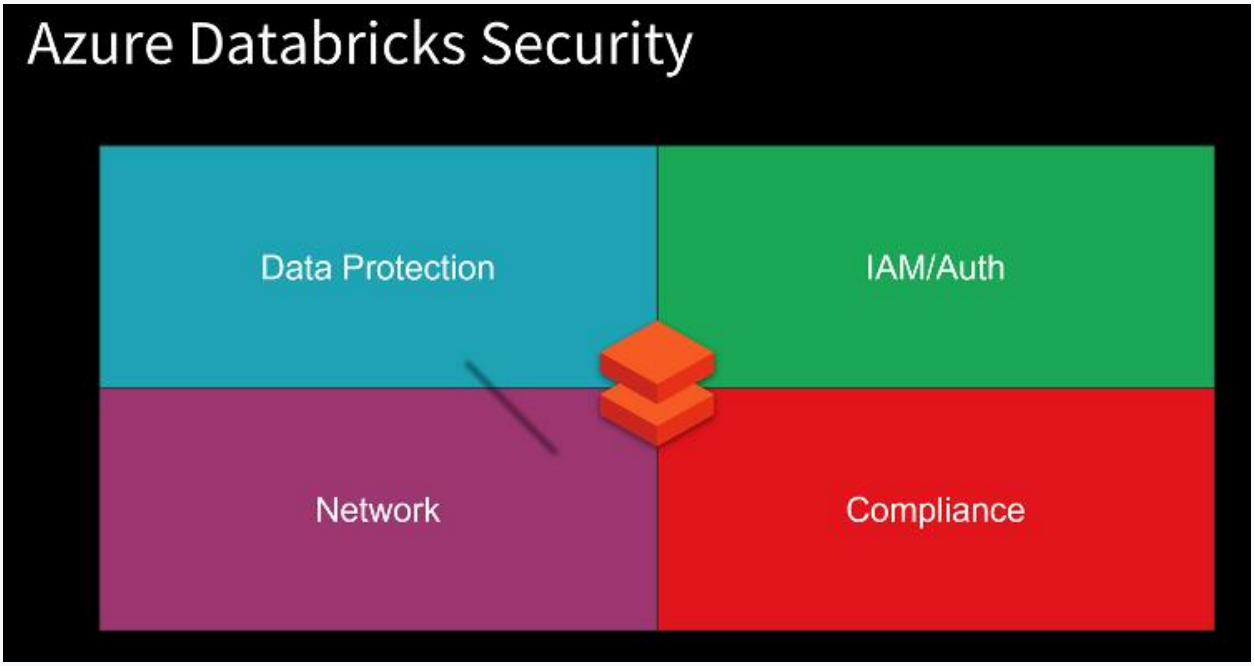
The following are the facets of Azure Databricks security:

• Data Protection

• IAM/Auth

• Network

• Compliance



Data Protection is comprised of the following:

• Encryption at-rest – Service Managed Keys, User Managed Keys

• Encryption in-transit (Transport Layer Security - TLS)

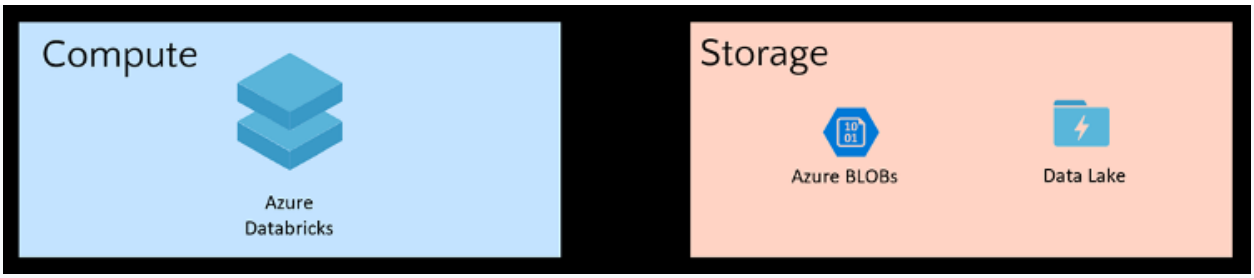
• File/Folder Level access control lists (ACLs) for Azure Active Directory (AAD) Users, Groups, Service Principals

• ACLs for Clusters, Folders, Notebooks, Tables, Jobs

• Secrets with Azure Key Vault

Encryption at-rest

Azure Databricks has separation of compute and storage.



Azure Databricks is a compute platform. It does not store data, except for notebooks. Clusters are transient in nature. They process the data then are terminated. All data is stored in the customer's subscription. Because the Azure storage services use server-side encryption, communication between these services and the Databricks clusters is seamless.

Storage Services such as Azure Storage Blobs and Azure Data Lake Storage (Gen1/2) provide:

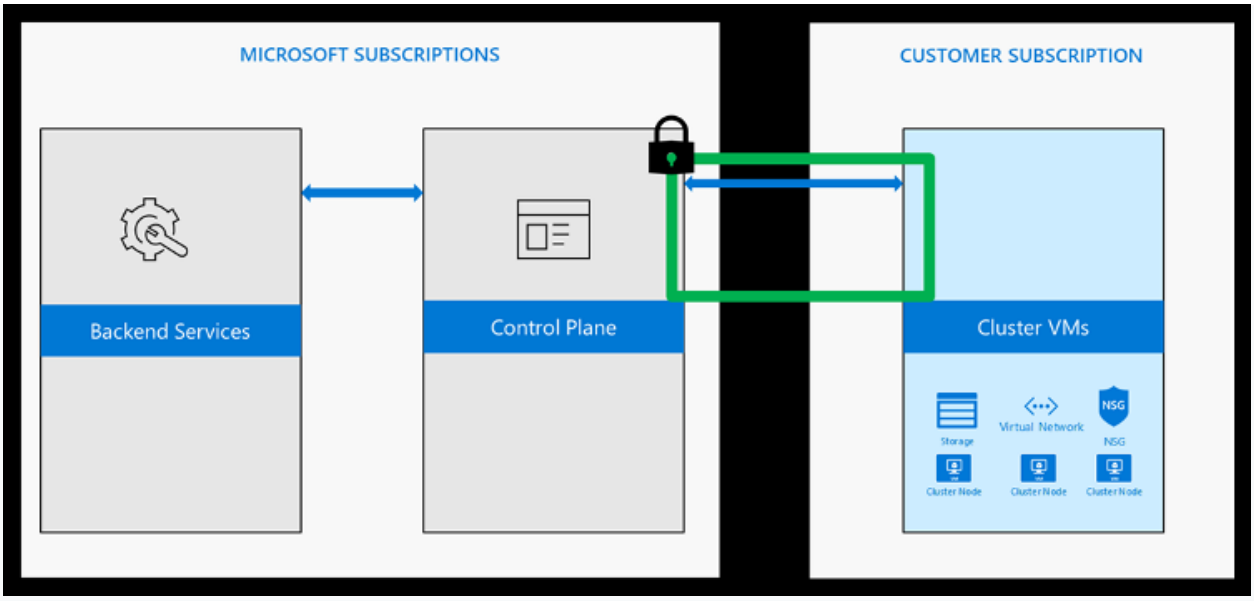
• Encryption of Data - Automatic server-side encryption in addition to encryption on storage attached to the VMs

• Customer Managed Keys - Bring your own keys with Key Vault integration

• File/Folder Level ACLs (Azure Data Lake Storage (Gen1/2))

Encryption in-transit

All the traffic from the Control Plane to the clusters in the customer subscription (Data Plane) is always encrypted with TLS.



When clusters access data from various Azure services, TLS is always used to ensure encryption in-transit.

When customers access notebooks via their web browsers, the connection is also secured with TLS.

Access control - ADLS Passthrough

When enabled, authentication automatically takes place in Azure Data Lake Storage (ADLS) from Azure Databricks clusters using the same Azure Active Directory (Azure AD) identity that one uses to log into Azure Databricks. Commands running on a configured cluster will be able to read and write data in ADLS without needing to configure service principal credentials. Any ACLs applied at the folder or file level in ADLS are enforced based on the user's identity.

ADLS Passthrough is configured when you create a cluster in the Azure Databricks workspace. ADLS Gen1 requires Databricks Runtime 5.1+. ADLS Gen2 requires 5.3+.

On a *standard cluster*, when you enable this setting you must set single user access to one of the Azure Active Directory (AAD) users in the Azure Databricks workspace. Only one user is allowed to run commands on this cluster when Credential Passthrough is enabled.

High-concurrency clusters can be shared by multiple users. When you enable ADLS Passthrough on this type of cluster, it does not require you to select a single user.

Access control - Folders

Access control is available only in the Premium SKU. By default, all users can create and modify workspace objects unless an administrator enables workspace access control. With workspace access control, individual permissions determine a user's abilities. This section describes the individual permissions and how to enable and configure workspace access control.

You can assign five permission levels to notebooks and folders: No Permissions, Read, Run, Edit, and Manage. The following tables lists the abilities for each permission.

Ability	No Permissions	Read	Run	Edit	Manage
View items		x	x	x	x
Create, clone, import, export items		x	x	x	x
Run commands on notebooks			x	x	x
Attach/detach notebooks			x	x	x
Delete items				x	x
Move/rename items				x	x
Change permissions					x

Access control - Notebooks

Ability	No Permissions	Read	Run	Edit	Manage
View cells		x	x	x	x
Comment		x	x	x	x
Run commands			x	x	x
Attach/detach notebooks			x	x	x
Edit cells				x	x
Change permissions					x

All notebooks in a folder inherit all permissions settings of that folder. For example, a user that has Run permission on a folder has Run permission on all notebooks in that folder.

To enable workspace access control:

• Go to the Admin Console.

• Select the Access Control tab.

• Click the Enable button next to Workspace Access Control.

• Click Confirm to confirm the change.

Access control - Clusters

All users can view libraries. To control who can attach libraries to clusters, manage access control on clusters.

By default, all users can create and modify clusters unless an administrator enables cluster access control. With cluster access control, permissions determine a user's abilities. There are four permission levels for a cluster: No Permissions, Can Attach To, Can Restart, and Can Manage:

Ability	No Permissions	Can Attach To	Can Restart	Can Manage
Attach notebook to cluster		x	x	x
View Spark UI		x	x	x
View cluster metrics		x	x	x
Terminate cluster			x	x
Start cluster			x	x
Restart cluster			x	x
Edit cluster				x
Attach library to cluster				x
Resize cluster				x
Modify permissions				x

Note: You have Can Manage permission for any cluster that you create.

Access control - Jobs

To control who can run jobs and see the results of job runs, manage access control on jobs.

There are five permission levels for jobs: No Permissions, Can View, Can Manage Run, Is Owner, and Can Manage. The Can Manage permission is reserved for administrators.

Ability	No Permissions	Can View	Can Manage Run	Is Owner	Can Manage (admin)
View job details and settings	x	x	x	x	x
View results, Spark UI, logs of a job run		x	x	x	x
Run now			x	x	x
Cancel run			x	x	x
Edit job settings				x	x
Modify permissions				x	x

Access control - Tables

Table access control (table ACLs) lets you programmatically grant and revoke access to your data from SQL, Python, and PySpark.

By default, all users have access to all data stored in a cluster's managed tables unless an administrator enables table access control for that cluster. Once table access control is enabled for a cluster, users can set permissions for data objects on that cluster.

Before you can grant or revoke privileges on data objects, an administrator must enable table access control for the cluster.

View-based access control model

The Azure Databricks view-based access control model defines the following privileges:

• `SELECT` – gives read access to an object.

• `CREATE` – gives ability to create an object (for example, a table in a database)

• `MODIFY` – gives ability to add/delete/modify data to/from an object.

• `READ_METADATA` – gives ability to view an object and its metadata.

• `CREATE_NAMED_FUNCTION` – gives ability to create a named UDF in an existing catalogue or database.

• `ALL PRIVILEGES` – gives all privileges (gets translated into all the above privileges)

The privileges above can apply to the following classes of objects:

• `CATALOG` - controls access to the entire data catalog.

• `DATABASE` - controls access to a database.

• `TABLE` - controls access to a managed or external table.

• `VIEW` - controls access to SQL views.

• `FUNCTION` - controls access to a named function.

• `ANONYMOUS FUNCTION` - controls access to anonymous or temporary functions.

• `ANY FILE` - controls access to the underlying filesystem.

Secrets

Using the Secrets APIs, Secrets can be securely stored including in an Azure Key Vault or Databricks backend. Authorized users can consume the secrets to access services.

Azure Databricks has two types of secret scopes: Key Vault-backed and Databricks-backed. These secret scopes allow you to store secrets, such as database connection strings, securely. If someone tries to output a secret to a notebook, it is replaced

by `[REDACTED]`. This helps prevent someone from viewing the secret or accidentally leaking it when displaying or sharing the notebook.

As a best practice, instead of directly entering your credentials into a notebook, use Azure Databricks secrets to store your credentials and reference them in notebooks and jobs.

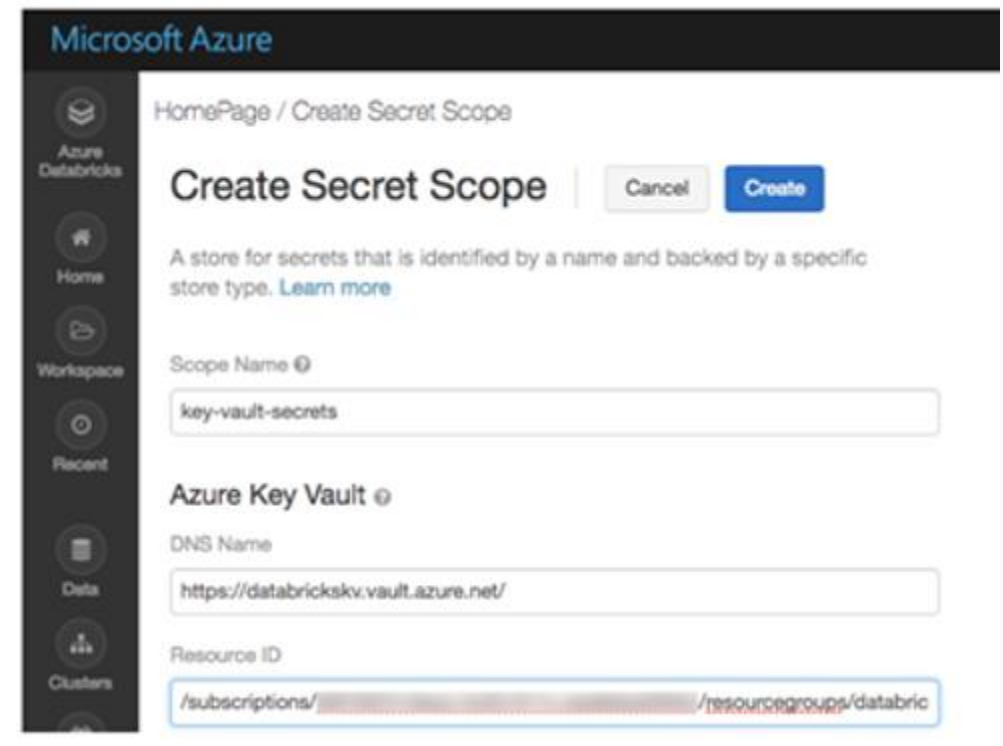
To set up secrets you:

• Create a secret scope. Secret scope names are case insensitive.

• Add secrets to the scope. Secret names are case insensitive.

• If you have the Azure Databricks Premium Plan, assign access control to the secret scope.

Screenshot of creating an Azure Key Vault-backed secret scope:



https://docs.microsoft.com/en-us/azure/databricks/scenarios/security-baseline

Question 21: **Correct**

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 plans to use Microsoft Azure Synapse Analytics and they are in need of expert guidance.

Wayne Enterprises has an Azure Active Directory (Azure AD) tenant which contains a security group named Group1. They also have an Azure Synapse Analytics dedicated SQL pool named bw1 that contains a schema named schema1.

Required:

• Grant Group1 read-only permissions to all the tables and views in schema1.

• The solution must use the principle of least privilege.

Bruce and the team have put together some options they are considering to fulfill the requirements:

a. Create a database role named Role1 and grant Role1 `SELECT` permissions to Schema1

b. Create user from external provider for Group1

c. Assign the Azure role-based access control (RBAC) Reader role for bw1 to Group1

d. Add user to the Role1

e. Create Role1 with `SELECT` on Schema1

f. Create a database user in bw1 that represents Group1 and uses the `FROM EXTERNAL PROVIDER` clause

g. Assign Role1 to the Group1 database use

Which sequence of actions should you recommend to the team to use?

- $\mathbf{b \rightarrow d \rightarrow c \rightarrow d}$

- **b → e → d**

  (Correct)

- **b → e → d → c**

- **a → d → g → c**

Explanation

The correct sequence of actions should you recommend to the team to use is: b → e → d

1. Create user from external provider for Group1

2. Create Role1 with select on schema1

3. Add user to the Role1

Authenticate to dedicated SQL pool (formerly SQL DW) in Azure Synapse Analytics

To connect to a dedicated SQL pool (formerly SQL DW), you must pass in security credentials for authentication purposes. Upon establishing a connection, certain connection settings are configured as part of establishing your query session.

SQL authentication

To connect to dedicated SQL pool (formerly SQL DW), you must provide the following information:

• Fully qualified servername

• Specify SQL authentication

• Username

• Password

• Default database (optional)

By default, your connection connects to the *master* database and not your user database. To connect to your user database, you can choose to do one of two things:

• Specify the default database when registering your server with the SQL Server Object Explorer in SSDT, SSMS, or in your application connection string. For example, include the InitialCatalog parameter for an ODBC connection.

• Highlight the user database before creating a session in SSDT.

Azure Active Directory authentication

Azure Active Directory authentication is a mechanism of connecting to SQL pool by using identities in Azure Active Directory (Azure AD). With Azure Active Directory authentication, you can centrally manage the identities of database users and other Microsoft services in one central location. Central ID management provides a single place to manage dedicated SQL pool (formerly SQL DW) users and simplifies permission management.

Benefits

Azure Active Directory benefits include:

• Provides an alternative to SQL Server authentication.

• Helps stop the proliferation of user identities across servers.

• Allows password rotation in a single place

• Manage database permissions using external (Azure AD) groups.

• Eliminates storing passwords by enabling integrated Windows authentication and other forms of authentication supported by Azure Active Directory.

• Uses contained database users to authenticate identities at the database level.

• Supports token-based authentication for applications connecting to SQL pool.

• Supports Multi-Factor authentication through Active Directory Universal Authentication for various tools including SQL Server Management Studio and SQL Server Data Tools.

Configuration steps

Follow these steps to configure Azure Active Directory authentication.

• Create and populate an Azure Active Directory

• Optional: Associate or change the active directory that is currently associated with your Azure Subscription

• Create an Azure Active Directory administrator for Azure Synapse

• Configure your client computers

• Create contained database users in your database mapped to Azure AD identities

• Connect to your SQL pool by using Azure AD identities

Find the details

• The steps to configure and use Azure Active Directory authentication are nearly identical for Azure SQL Database and Synapse SQL in Azure Synapse. Follow the detailed steps in the topic Connecting to SQL Database or SQL Pool By Using Azure Active Directory Authentication.

• Create custom database roles and add users to the roles. Then grant granular permissions to the roles. For more information, see Getting Started with Database Engine Permissions.

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

Question 22: **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 is reviewing the work done by his team to ensure optimal performance. Right now he is reviewing a data type for storing data that contains less than 128 characters.

To achieve improved query performance, which of the following would be the best data type for storing data that contains less than 128 characters?

- `VARCHAR(MAX)`
- `VARCHAR(MIN)`
-

`VARCHAR(128)`

(Correct)

-

`NVARCHAR(128)`

(Incorrect)

Explanation

Limiting the size of the data type and not using size variability will provide the best performance.

Dedicated SQL pools support the most commonly used data types. With that, it is essential to keep in mind that minimizing the size of data types shortens the row length, which leads to better query performance. Use the smallest data type that works for your data:

• Avoid defining character columns with a large default length. For example, if the longest value is 25 characters, define your column as `VARCHAR(25)`.

• Avoid using `NVARCHAR` when you only need `VARCHAR` to support non-unicode data.

• When possible, use `NVARCHAR(4000)` or `VARCHAR(8000)` instead of `NVARCHAR(MAX)` or `VARCHAR(MAX)`.

• Focus on using the minimal data type as possible. Use the minor data type that can reliably contain all possible values you have. For example, use `tinyint` instead of `smallint` or int for exact-numbers between 0-255. `tinyint` would be sufficient for a person's age because no one lives to be more than 255 years old.

• Use `DATETIME` instead of storing data values in different formats such as string or numeric values. When it comes to querying your data based on date, an additional conversion will put a massive overhead on your queries.

• If you only need to store a date and not the time make sure you use `SMALLDATETIME` instead of `DATETIME`. There is a 4 bytes difference between the two.

Note: If you are using PolyBase external tables to load your SQL pool tables, the table row's defined length cannot exceed 1 MB. When a row with variable-length data exceeds 1 MB, you can load the row with BCP but not with PolyBase.

Workarounds for unsupported data types

The following list shows the data types that dedicated SQL pool doesn't support and gives useful alternatives for unsupported data types.

Unsupported data type	Workaround
geometry	varbinary
geography	varbinary
hierarchyid	nvarchar
image	varbinary
text	varchar
ntext	nvarchar
sql_variant	Split column into several strongly typed columns.
table	Convert to temporary tables.
timestamp	Rework code to use datetime2 and the CURRENT_TIMESTAMP function. Only constants are supported as defaults, so current_timestamp can't be defined as a default constraint. If you need to migrate row version values from a timestamp typed column, use BINARY(8) or VARBINARY(8) for NOT NULL or NULL row version values.
xml]	varchar
user-defined type	Convert back to the native data type when possible.
default values	Default values support literals and constants only.

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

Question 23: **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

A Control Activity in Data Factory is defined in JSON format as follows:

```
JSON
{
"name": "Control Activity Name",
"description": "description",
"type": "<ActivityType>",
"typeProperties":
{
},
"dependsOn":
{
}
}
```

Which of the JSON properties are required? (Select all that apply)

- ☑ **type**

  (Correct)

- ☐ **name**

  (Correct)

- ☐ **description**

  (Correct)

- ☐ **typeProperties**

- ☐ **dependsOn**

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 control activities

A Control Activity in Data Factory is defined in JSON format as follows:

```
1. JSON
2. {
3. "name": "Control Activity Name",
4. "description": "description",
5. "type": "<ActivityType>",
6. "typeProperties":
7. {
8. },
9. "dependsOn":
10.{
11.}
12.}
```

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: typeProperties

Properties in the typeProperties section depend on each type of activity.

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 24: **Incorrect**
Scenario: Duncan + Dotter Design is an architectural business based in New York City which was founded by John Raymond. The company is embarking on a Microsoft journey by switching things over from various other systems they have used over the years.

The development team has created an Azure Databricks workspace in which they plan to use code in notebooks to process data.

A junior developer is not sure what to create in the workspace and comes to you for guidance.

What should you tell them to create in the workspace?

- **A Spark cluster**

  (Correct)

-

Azure Data Lake Storage

-

A SQL Warehouse

-

A Windows Server virtual machine

(Incorrect)

Explanation
A Spark cluster is required to process data using code in notebooks.

Azure Databricks is a fully managed, cloud-based data analytics platform, which empowers developers to accelerate AI and innovation by simplifying the process of building enterprise-grade data applications. Built as a joint effort by Microsoft and the team that started Apache Spark, Azure Databricks provides data science, engineering, and analytical 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.

Identify Azure Databricks workloads

Azure Databricks is a comprehensive platform that offers many data processing capabilities. While you can use the service to support any workload that requires scalable data processing, Azure Databricks is optimized for three specific types of data workload and associated user personas:

• Data Science and Engineering

• Machine Learning

• SQL*

**SQL workloads are only available in premium tier workspaces.*

The Azure Databricks user interface supports three corresponding *persona* views that you can switch between depending on the workload you're implementing.

Get started with Azure Databricks

Azure Databricks is a cloud-based distributed platform for data processing built on Apache Spark. Databricks was designed to unify data science, data engineering, and business data analytics on Spark by creating an easy-to-use environment that enables users to spend more time working effectively with data, and less time focused on managing clusters and infrastructure. As the platform has evolved, it has kept up to date with the latest advances in the Spark runtime and added usability features to support common data workloads in a single, centrally managed interface.

Azure Databricks is hosted on the Microsoft Azure cloud platform and integrated with Azure services such as Azure Active Directory, Azure Storage, Azure Synapse Analytics, and Azure Machine Learning. Organizations can apply their existing capabilities with the Databricks platform and build fully integrated data analytics solutions that work with cloud infrastructure used by other enterprise applications.

Creating an Azure Databricks workspace

To use Azure Databricks, you must create an Azure Databricks *workspace* in your Azure subscription. You can accomplish this by:

• Using the Azure portal user interface.

• Using an Azure Resource Manager (ARM) or Bicep template.

• Using the New-AzDatabricksWorkspace Azure PowerShell cmdlet

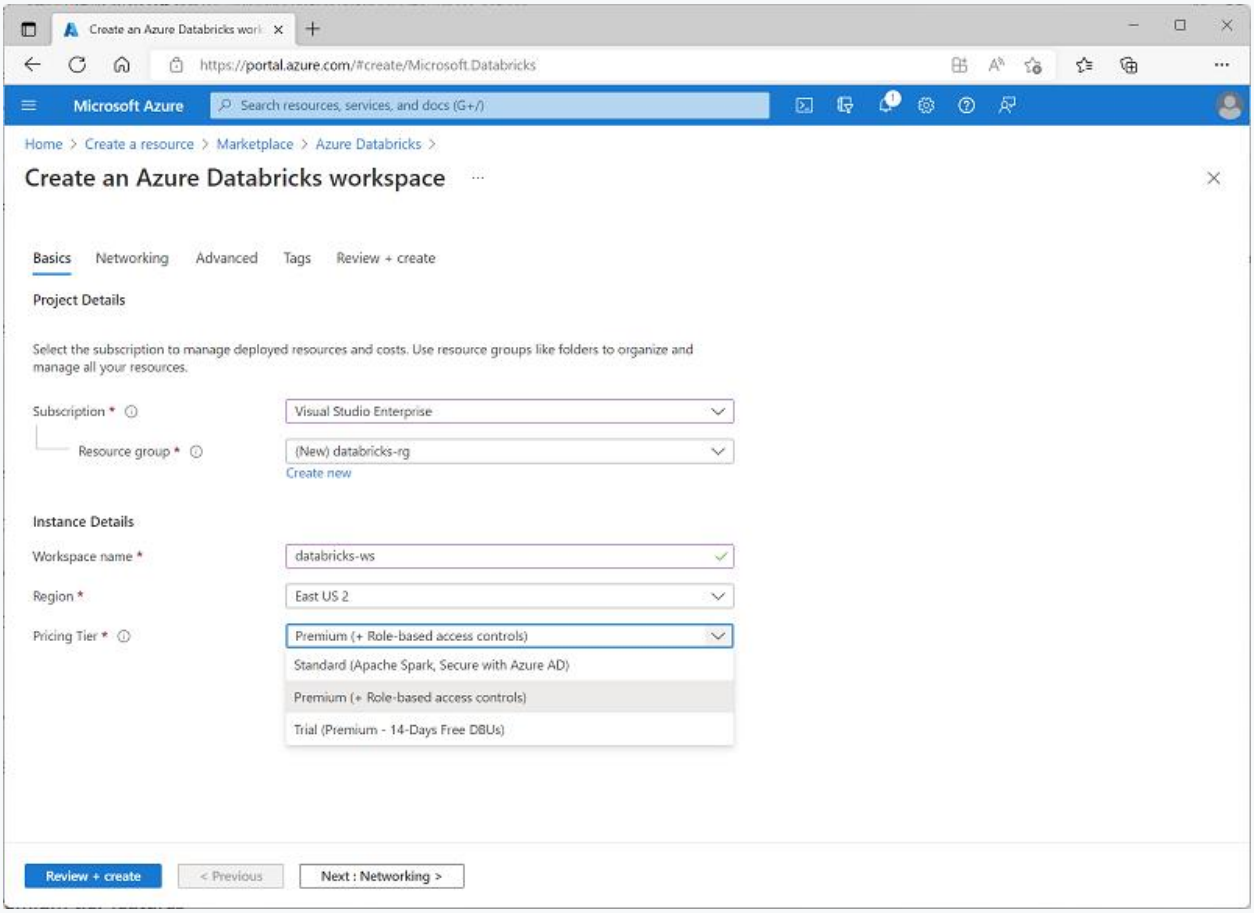
• Using the az databricks workspace create Azure command line interface (CLI) command.

When you create a workspace, you must specify one of the following pricing tiers:

• **Standard** - Core Apache Spark capabilities with Azure AD integration.

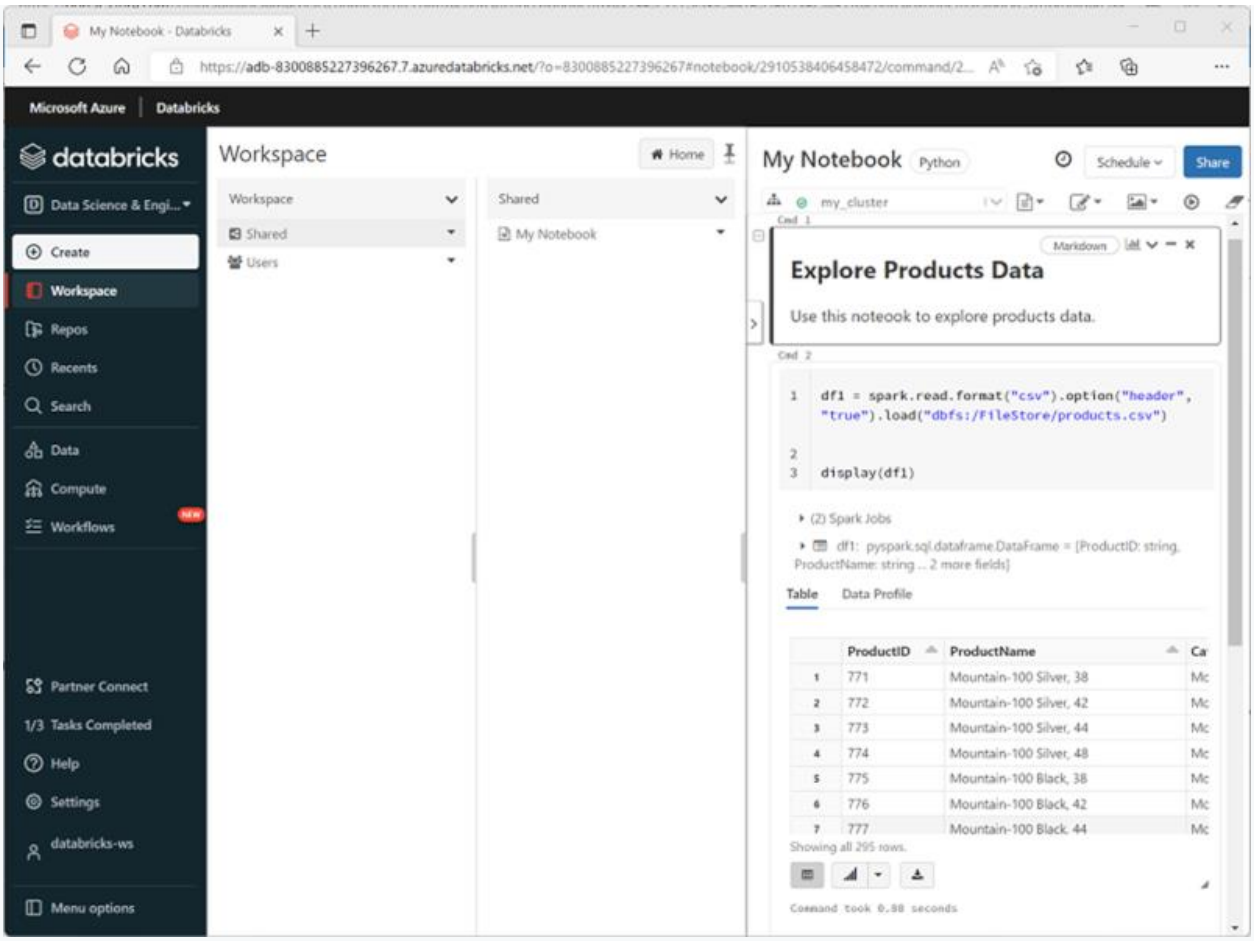
• **Premium** - Role-based access controls and other enterprise-level features.

• **Trial** - A 14-day free trial of a premium-level workspace



Using the Azure Databricks portal

After you've provisioned an Azure Databricks workspace, you can use the Azure Databricks portal to work with data and compute resources. The Azure Databricks portal is a web-based user interface through which you can create and manage workspace resources (such as Spark clusters) and work with data in files and tables.



https://learn.microsoft.com/en-us/azure/databricks/introduction/

Question 25: **Incorrect**
In which modes does Azure Databricks provide data encryption?

- **At-rest and in-transit**

  (Correct)

- **In-transit only**

- **At-rest only**

- **None of the listed options.**

  (Incorrect)

Explanation
Data stored in Azure Storage is encrypted using server-side encryption that is seamlessly accessed by Azure Databricks. All data transmitted between the Data Plane and the Control Plane is always encrypted in-flight via TLS.

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

Question 26: **Incorrect**
Which transformation is used to load data into a data store or compute resource?

- **Source**

- **Field**

- **Sink**

  (Correct)

- **Cache**

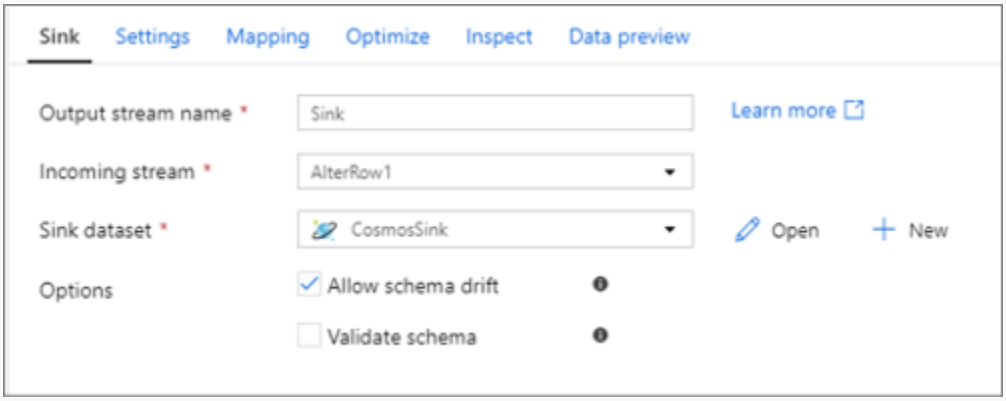
- **Window**

  (Incorrect)

Explanation
A Sink transformation allows you to choose a dataset definition for the destination output data. You can have as many sink transformations as your data flow requires.

Every data flow requires at least one sink transformation, but you can write to as many sinks as necessary to complete your transformation flow. To write to additional sinks, create new streams via new branches and conditional splits.

Each sink transformation is associated with exactly one Azure Data Factory dataset object or linked service. The sink transformation determines the shape and location of the data you want to write to.



A Sink transformation allows you to choose a dataset definition for the destination output data. You can have as many sink transformations as your data flow requires.

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

Question 27: **Incorrect**

Scenario: You are working as a consultant at Zerks Medical Insurance (ZMI) on Morag which was founded by the sisters Gamora and Nebula. Gamora wants to store details of customer attributes by which claims will be aggregated.

Which of the following table types should advise ZMI use?

- **Dimension table**

  (Correct)

-

Integrity table

-

Fact table

-

Staging table

(Incorrect)

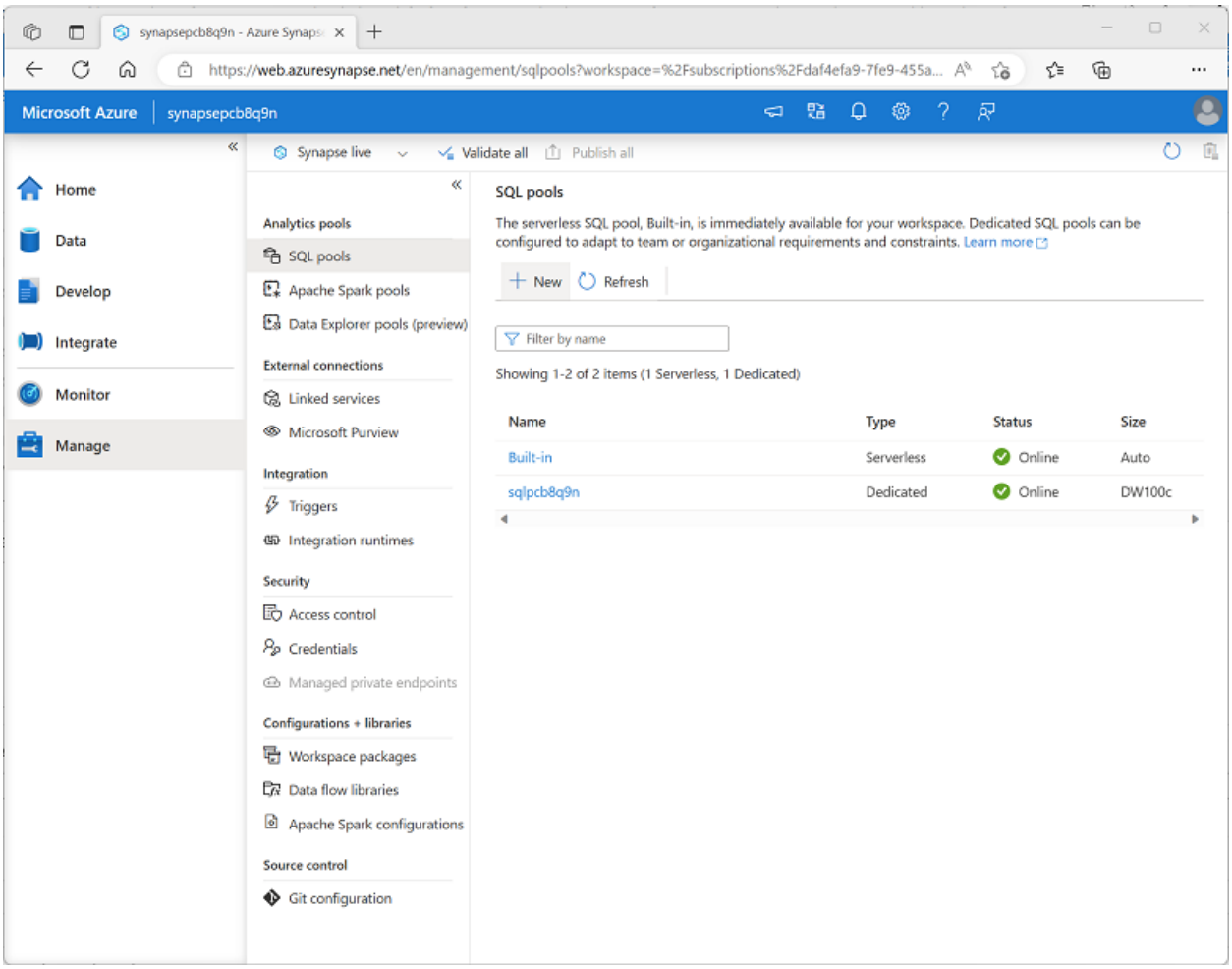
Explanation

- *Attributes of an entity by which numeric measures will be aggregated are stored in a dimension table.*
- Staging tables are used as temporary storage for data as it is loaded into the data warehouse.
- Fact tables are used to store numeric measures that can be aggregated.
- Integrity table is not a valid option.

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 28: **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.

What is the maximum size for a single publication (individual or batch) that is allowed by Azure Event Hub?

- **2 MB**
- **512 KB**
- **256 KB**

1 MB

(Correct)

Explanation

The maximum size for a single publication (individual or batch) that is allowed by Azure Event Hub is 1 MB.

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.

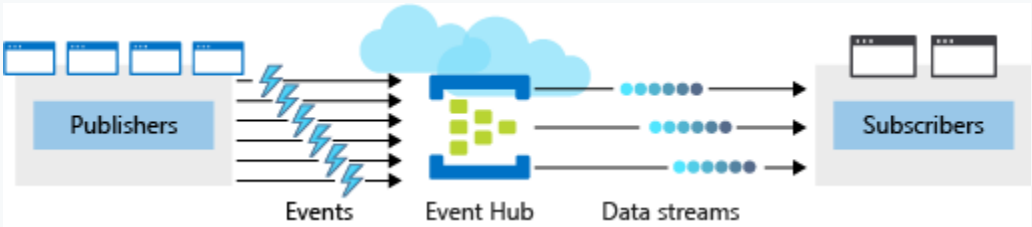
The load on your system can spike during weekends and holidays. The system must handle the increased throughput efficiently and accurately. Given the sensitive nature of the transactions, even the slightest error can have a considerable impact.

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.

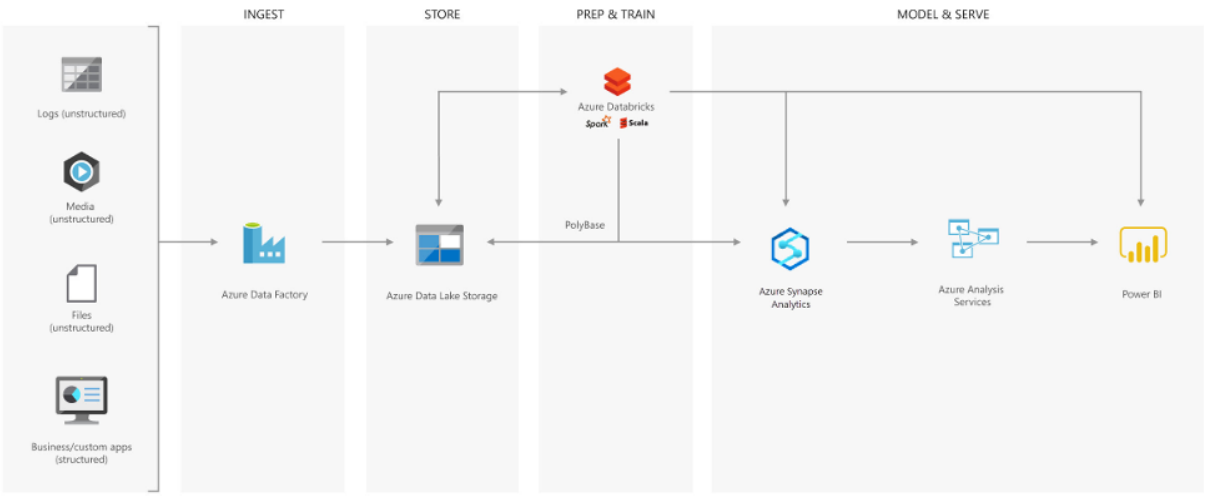
Summary

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.

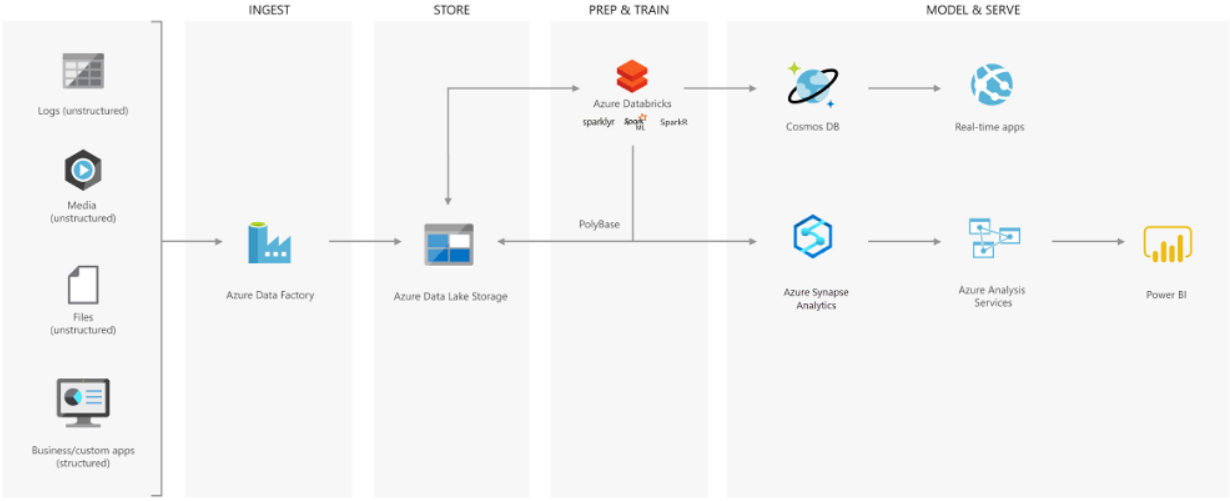
https://docs.microsoft.com/en-us/azure/event-hubs/event-hubs-quickstart-cli

Question 29: **Incorrect**
Scenario: 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. Review the following architecture designs.

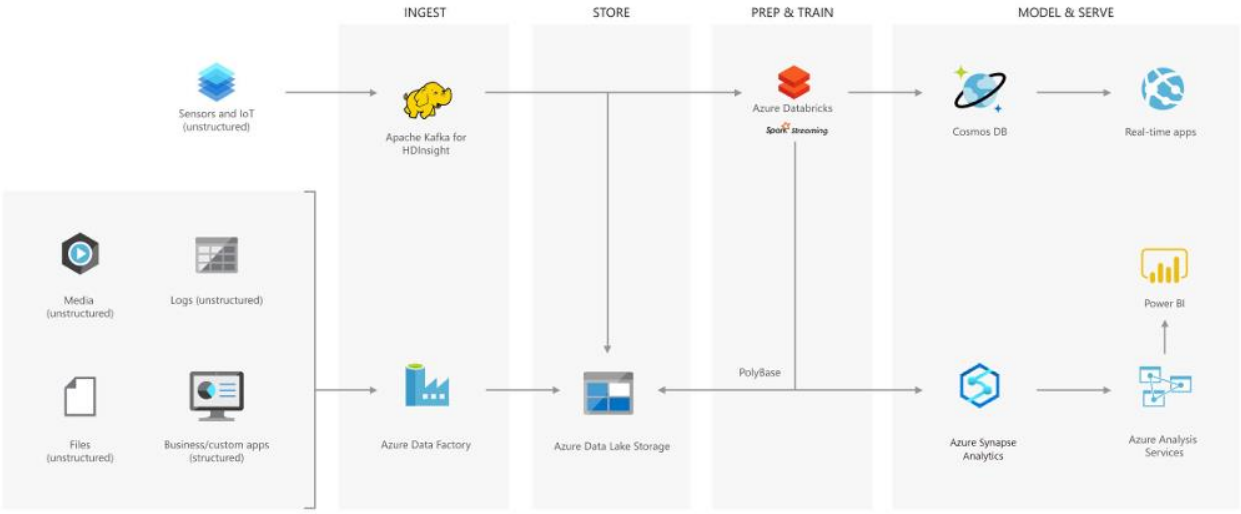
Design A:



Design B:



Design C:



Which architecture would be best suited for the need?

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

  (Incorrect)
- **Design A**
- **Design B**

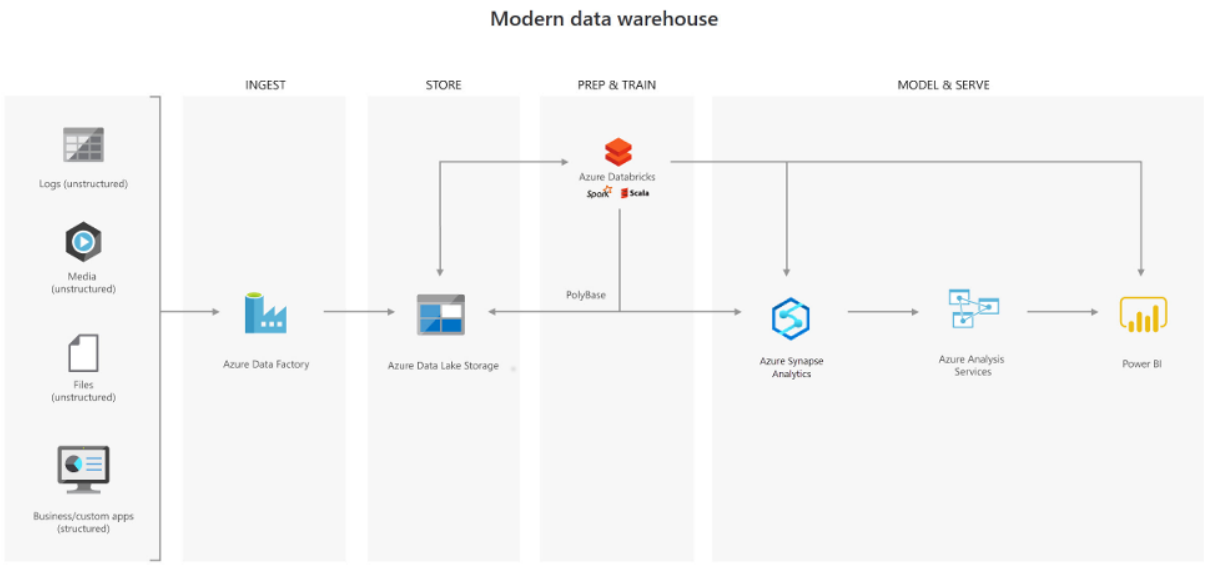
  (Correct)

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.

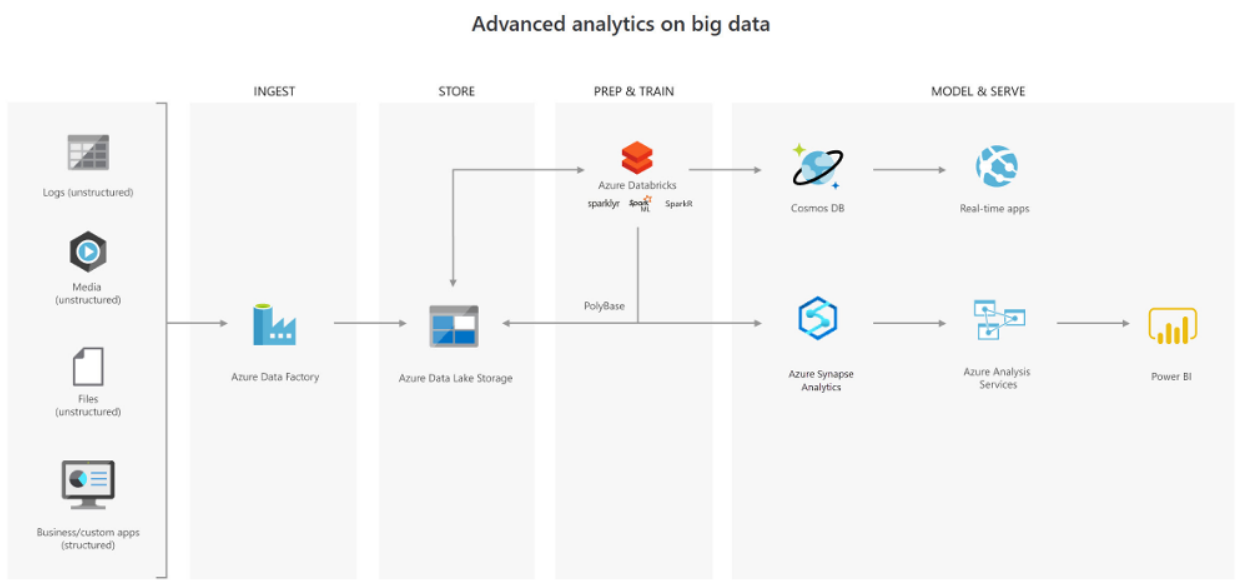
The company has turned to you for help with creating an architecture that can scale with the data needs that are required to create a predictive model and to handle the JSON data so that it's integrated into the BI solution. You suggest the following architecture:



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:



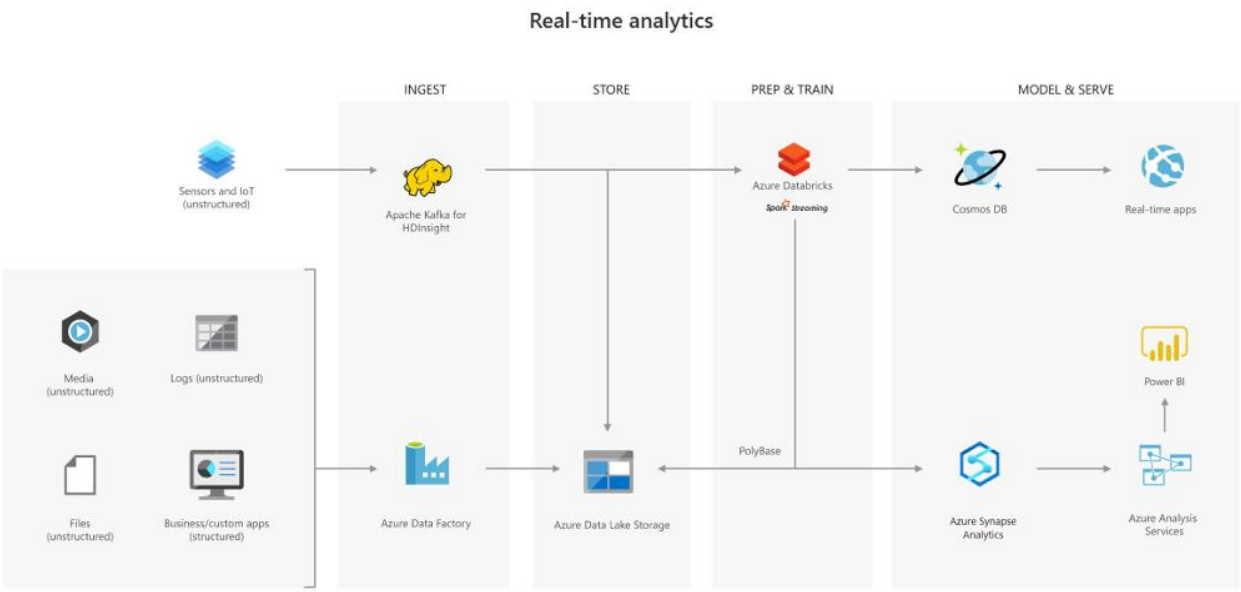
Real-time analytical solutions

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 30: **Incorrect**

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

As a Data Engineer, you can transfer and move data in several ways. The most common tool is [?], which provides robust resources and nearly 100 enterprise connectors. [?] also allows you to transform data by using a wide variety of languages.

- **Azure Data Factory**

  (Correct)

- **Azure Databricks**

- **Azure Stream Analytics**

- **Azure Data Catalogue**

- **Azure Data Lake Storage**

  (Incorrect)

Explanation
As a Data Engineer, you can transfer and move data in several ways. One way is to start an *Extract, Transform, and Load (ETL)* process.

Extraction sources can include databases, files, and streams. Each source has unique data formats that can be structured, semistructured, or unstructured. In Azure, data sources include Azure Cosmos DB, Azure Data Lake, files, and Azure Blob storage.

ETL tools

As a data engineer, you'll use several tools for ETL. The most common tool is Azure Data Factory, which provides robust resources and nearly 100 enterprise connectors. Data Factory also allows you to transform data by using a wide variety of languages.

You might find that you also need a repository to maintain information about your organization's data sources and dictionaries. Azure Data Catalog can store this information centrally.

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

Evolution from ETL

Azure has opened the way for technologies that can handle unstructured data at an unlimited scale. This change has shifted the paradigm for loading and transforming data from ETL to extract, load, and transform (ELT).

The benefit of ELT is that you can store data in its original format, be it JSON, XML, PDF, or images. In ELT, you define the data's structure during the transformation phase, so you can use the source data in multiple downstream systems.

In an ELT process, data is extracted and loaded in its native format. This change reduces the time required to load the data into a destination system. The change also limits resource contention on the data sources.

The steps for the ELT process are the same as for the ETL process. They just follow a different order.

Another process like ELT is called extract, load, transform, and load (ELTL). The difference with ELTL is that it has a final load into a destination system.

https://docs.microsoft.com/en-us/azure/architecture/data-guide/relational-data/etl

Question 31: **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 IT team has come up with a list of commands they are considering to execute which is shown below:

a. `CREATE EXTERNAL DATA SOURCE`

b. `CREATE EXTERNAL FILE FORMAT`

c. `CREATE EXTERNAL TABLE`

d. `CREATE EXTERNAL TABLE AS SELECT`

e. `CREATE DATABASE SCOPED CREDENTIAL`

The Ask:

Alfred places a great importance on this project and asks you to work closely with the team to ensure that the Twitter feed data can be analyzed in the dedicated SQL pool. The solution must meet the customer sentiment analytic requirements.

As the Azure expert, the team looks to you for direction with regards to the proper path forward.

Which Transact-SQL DDL commands should you recommend to be run in sequence?

- **e → d → e → a → c**

  (Incorrect)

-

d → a → b → e

- **a → b → d**

  (Correct)

- **c → d → a → e**

Explanation

The correct commands in sequence are a → b → d.

The requirement is to 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.

Command 1: `CREATE EXTERNAL DATA SOURCE`

External data sources are used to connect to storage accounts.

Command 2: `CREATE EXTERNAL FILE FORMAT`

`CREATE EXTERNAL FILE FORMAT` creates an external file format object that defines external data stored in Azure Blob Storage or Azure Data Lake Storage.

Creating an external file format is a prerequisite for creating an external table.

Command 3: `CREATE EXTERNAL TABLE AS SELECT`

When used in conjunction with the `CREATE TABLE AS SELECT` statement, selecting from an external table imports data into a table within the SQL pool. In addition to the `COPY` statement, external tables are useful for loading data.

The `CREATE EXTERNAL TABLE` command is a wrong choice because it creates an external table for Synapse SQL to access data stored in Azure Blob Storage or Azure Data Lake Storage.

External tables with Synapse SQL

An external table points to data located in Hadoop, Azure Storage blob, or Azure Data Lake Storage. External tables are used to read data from files or write data to files in

Azure Storage. With Synapse SQL, you can use external tables to read external data using dedicated SQL pool or serverless SQL pool.

Depending on the type of the external data source, you can use two types of external tables:

Hadoop external tables that you can use to read and export data in various data formats such as CSV, Parquet, and ORC. Hadoop external tables as available in dedicated Synapse SQL pools, but they are not available in serverless SQL pools.

Native external tables that you can use to read and export data in various data formats such as CSV as Parquet. Native external tables are available in serverless Synapse SQL pools, but they are not available in dedicated Synapse SQL pools.

The key differences between Hadoop and native external tables are presented in the following table:

External table type	Hadoop	Native
Dedicated SQL pool	Available	Not available
Serverless SQL pool	Not available	Available
Supported formats	Delimited/CSV, Parquet, ORC, Hive RC, and RC	Delimited/CSV and Parquet
Folder partition elimination	No	Only for the partitioned tables synchronized from Apache Spark pools in Synapse workspace
Custom format for location	No	Yes, using wildcards like `/year=*/month=*/day=*`
Recursive folder scan	Always	Only when specified `/**` in the location path
Storage authentication	Storage Access Key(SAK), AAD passthrough, Managed identity, Custom application Azure AD identity	Shared Access Signature(SAS), AAD passthrough, Managed identity

External tables in dedicated SQL pool and serverless SQL pool

You can use external tables to:

• Query Azure Blob Storage and Azure Data Lake Gen2 with Transact-SQL statements.

• Store query results to files in Azure Blob Storage or Azure Data Lake Storage using CETAS

• Import data from Azure Blob Storage and Azure Data Lake Storage and store it into dedicated SQL pool (only Hadoop tables in dedicated pool).

*Note: When used in conjunction with the CREATE TABLE AS SELECT statement, selecting from an external table imports data into a table within the **dedicated** SQL pool. In addition to the COPY statement, external tables are useful for loading data.*

You can create external tables in Synapse SQL pools via the following steps:

• `CREATE EXTERNAL DATA SOURCE`

• `CREATE EXTERNAL FILE FORMAT`

• `CREATE EXTERNAL TABLE`

Security

User must have `SELECT` permission on external table to read the data. External table access underlying Azure storage using the database scoped credential defined in data source using the following rules:

• Data source without credential enables external tables to access publicly available files on Azure storage.

• Data source can have credential that enables external tables to access only the files on Azure storage using SAS token or workspace Managed Identity

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

Question 32: **Correct**
Identify the missing word(s) in the following sentence within the context of Microsoft Azure.

In Azure Stream Analytics, a(n) [?] is a unit of execution.

- **Output**
- **Input**
- **Transformation query**
- **Job**

  (Correct)

Explanation
In Azure Stream Analytics, a *job* is a unit of execution. A Stream Analytics job pipeline consists of three parts:

• An **input** that provides the source of the data stream.

• A **transformation query** that acts on the input. For example, a transformation query could aggregate the data.

• An **output** that identifies the destination of the transformed data.

The Stream Analytics pipeline provides a transformed data flow from input to output, as the following diagram shows.

https://docs.microsoft.com/en-us/azure/stream-analytics/stream-analytics-monitoring
Question 33: **Incorrect**

If you want to store data *without performing analysis on the data*, how should you set the Hierarchical Namespace option within the storage account of an Azure Blob storage account?

- **Disabled / OFF**

  (Correct)

- **Enabled / ON**

- **Auto-scale / Dynamic**

- **Fixed**

  (Incorrect)

Explanation
In Azure Blob storage, you can store large amounts of unstructured ("object") data, in a single hierarchy, also known as a flat namespace. You can access this data by using `HTTP` or `HTTPs`. Azure Data Lake Storage Gen2 builds on blob storage and optimizes I/O of high-volume data by using hierarchical namespaces that you turned on in the previous exercise.

Hierarchical namespaces organize blob data into *directories* and stores metadata about each directory and the files within it. This structure allows operations, such as directory renames and deletes, to be performed in a single atomic operation. Flat namespaces, by contrast, require several operations proportionate to the number of objects in the structure. Hierarchical namespaces keep the data organized, which yields better storage and retrieval performance for an analytical use case and lowers the cost of analysis.

Azure Blob storage vs. Azure Data Lake Storage

If you want to store data *without performing analysis on the data*, set the **Hierarchical Namespace** option to **Disabled** to set up the storage account as an Azure Blob storage

account. You can also use blob storage to archive rarely used data or to store website assets such as images and media.

If you are performing analytics on the data, set up the storage account as an Azure Data Lake Storage Gen2 account by setting the **Hierarchical Namespace** option to **Enabled**. Because Azure Data Lake Storage Gen2 is integrated into the Azure Storage platform, applications can use either the Blob APIs or the Azure Data Lake Storage Gen2 file system APIs to access data.

https://blog.pragmaticworks.com/azure-data-lake-vs-azure-blob-storage-in-data-warehousing

Question 34: **Incorrect**
Where is the best place to monitor spark pools?

- **Azure Monitor from the Azure Portal linked to your Azure Synapse Workspace**
- **Monitor tab in Azure Advisor linked to your Azure Synapse Workspace**
- **Monitor tab in Azure Synapse Studio within your Azure Synapse Workspace**

  (Correct)
- **Any of the listed options are equally proficient to monitor spark pools**

  (Incorrect)

Explanation
If you want to monitor your spark pools the best place to go to is to navigate to the Monitor tab in Azure Synapse Studio within your Azure Synapse Workspace.

The Monitor hub enables you to view pipeline and trigger runs, view the status of the various integration runtimes that is running, view Apache Spark jobs, SQL requests, and data flow debug activities.

We will focus on the Apache Spark Pools jobs within the Monitor Tab in Azure Synapse Studio that you can access through your workspace. The reason why is, if you want to see the status of a job or activity, it's exactly where you want to go.

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.

If you have created a pipeline, and you ran that pipeline, you can see all the pipeline run activities here. It is also possible to view run details where you can see input and outputs for the activities in the pipeline as well as error messages that might have occurred.

If you automated a pipeline run by setting up automated triggers, you can find the runs here as well. If you would like to create new triggers, schedules, or tumbling windows and event-based triggers to execute a pipeline, it is where you need to go.

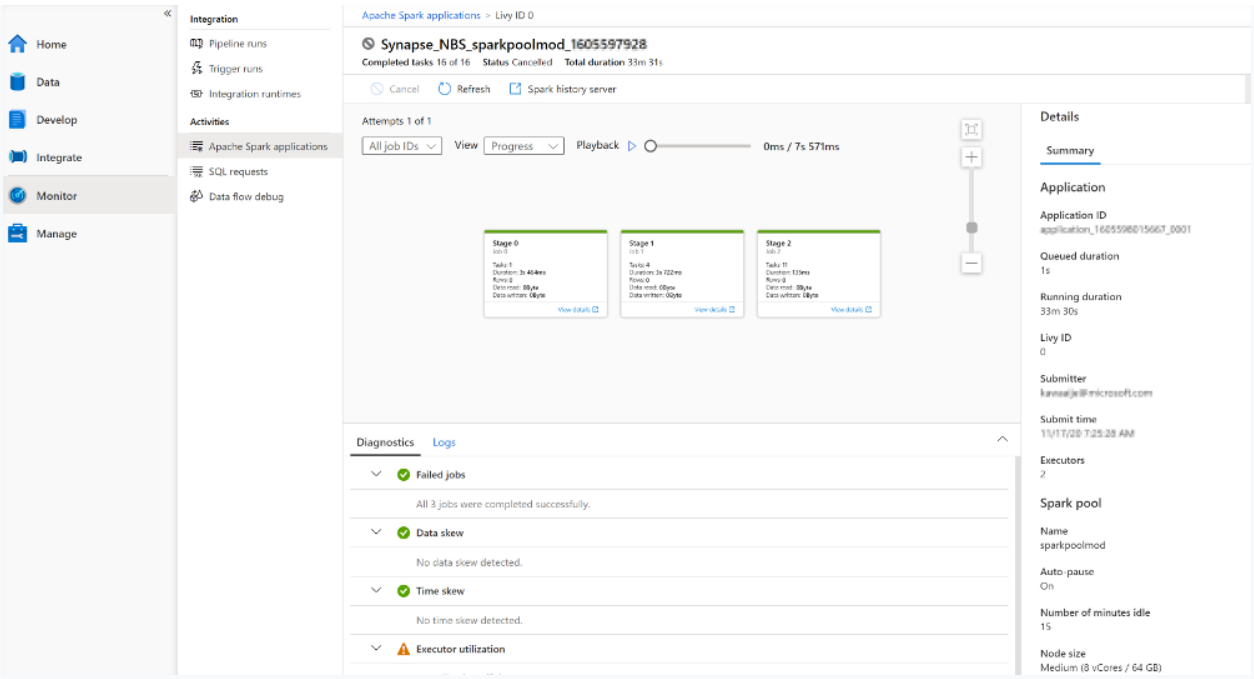
In relation to Apache Spark applications, you are able to see all the Spark applications that are running or have run in your workspace.

Let us deep dive into the monitor tab of the Synapse Studio environment within your Synapse Analytics Workspace.

Let's say you ran some Apache Spark activities, what do you do for monitoring?

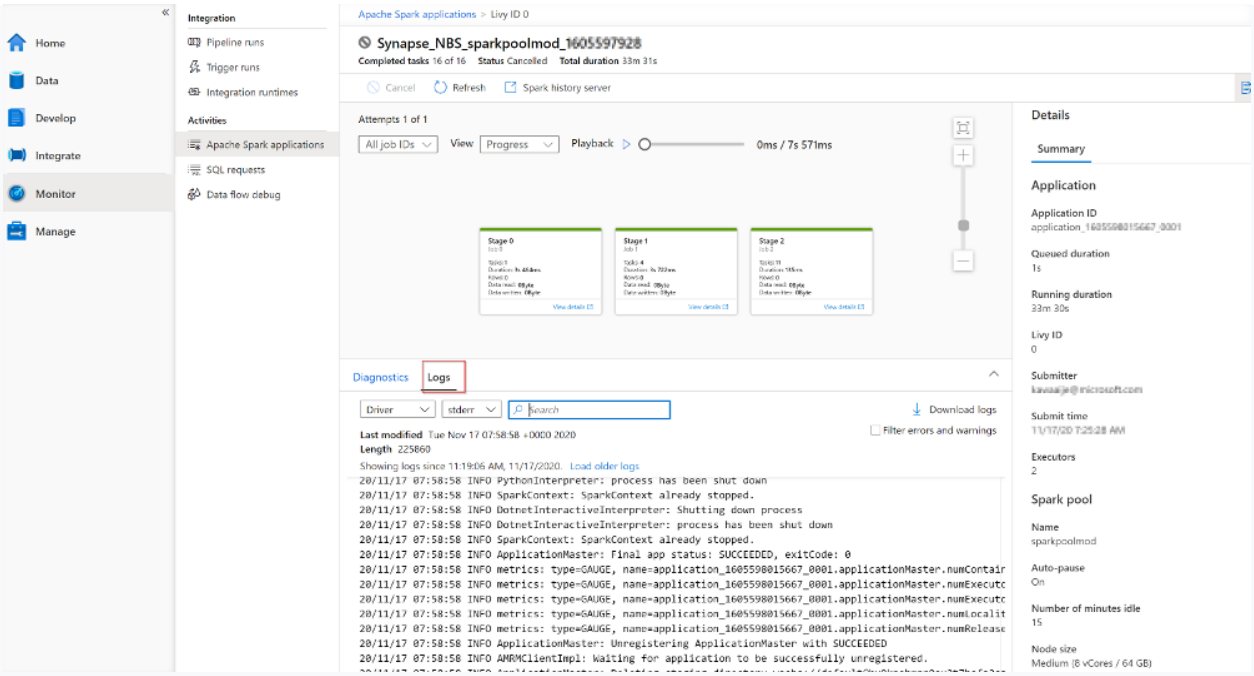
First, you should navigate to Monitor > Activities > Apache Spark applications. It's here where you can see all the Spark applications that are running or have run in your workspace. If you want to find out more about information about a Spark Application that is no longer running, you should click on the application name in the Monitor -> Apache Spark Application tab, name. Here you will find all the details of the spark application.

To give you a visual interpretation of how that looks like. see below:



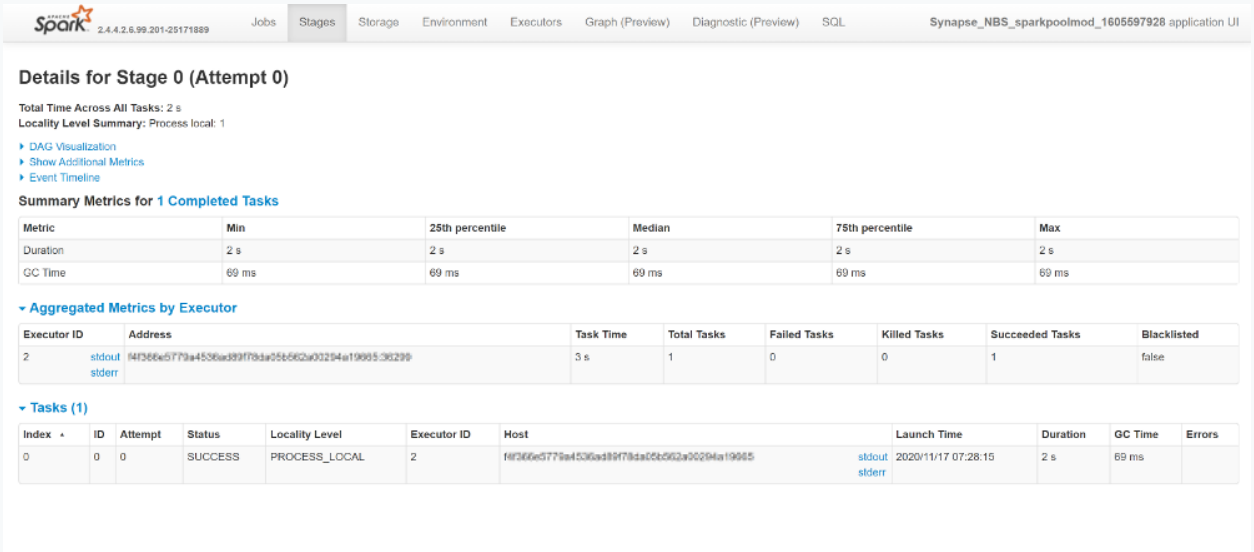
If you are familiar with Apache Spark, you can find the standard Apache Spark history server UI by clicking on Spark history server.

Not only can you check the diagnostics of the Spark application when you run, for example, a notebook attached to a spark pool, you can also check the logs if you navigate to the logs tab:



On the right-hand side, you'll find the details of the spark application as well as the running duration, number of executors, the spark pool details and many more.

In addition to that, if you want to view details for each stage, you can go to the View details tab of one of the stages that looks like the follows:



It will redirect you to the Apache Spark UI where you can find more details in relation to the stages of the Spark Pool.

https://docs.microsoft.com/en-us/azure/synapse-analytics/monitoring/how-to-monitor-spark-pools

Question 35: **Incorrect**
What is an example of a branching activity used in control flows in Azure Data Factory?

- **If-condition**

  (Correct)

- **Where-condition**

- **Lookup- condition**

- **Having-condition**

- **Until-condition**

  (Incorrect)

Explanation
If Condition activity in Azure Data Factory

The If Condition activity provides the same functionality that an if statement provides in programming languages. It executes a set of activities when the condition evaluates to true and another set of activities when the condition evaluates to false.

An example of a branching activity is the If-condition activity which is similar to an if-statement provided in programming languages.

https://docs.microsoft.com/en-us/azure/data-factory/control-flow-if-condition-activity

Question 36: **Incorrect**

Azure HDInsight provides technologies to help you ingest, process, and analyze big data. It supports batch processing, data warehousing, IoT, and data science.

Data processing within Hadoop uses which of the following to process big data? (Select three)

- ☐ **R**
- ☐ **JavaScript**
- ☐ **.NET**

  (Correct)
- ☑ **Python**

  (Correct)
- ☐ **C++**
- ☐ **Java**

  (Correct)
- ☐ **C#**

Explanation

Azure HDInsight provides technologies to help you ingest, process, and analyze big data. It supports batch processing, data warehousing, IoT, and data science.

Data processing

In Hadoop, use Java, Python and .NET to process big data. Mapper consumes and analyzes input data. It then emits tuples that Reducer can analyze. Reducer runs summary operations to create a smaller combined result set.

Spark processes streams by using Spark Streaming. For machine learning, use the 200 preloaded Anaconda libraries with Python. Use GraphX for graph computations.

Developers can remotely submit and monitor jobs from Spark. Storm supports common programming languages like Java, C#, and Python.

Queries

Hadoop supports Pig and HiveQL languages. In Spark, data engineers use Spark SQL.

Data security

Hadoop supports encryption, Secure Shell (SSH), shared access signatures, and Azure Active Directory security.

https://azure.microsoft.com/en-us/blog/manage-azure-hdinsight-clusters-using-net-python-or-java/

Question 37: **Incorrect**

Scenario: The Daily Bugle is a news organization led by J. Jonah Jameson which has a meager beginning and now has become a household name in news reporting. The company has grown well, despite some technology issues and now Jonah has hired you as an IT consultant to advise on several IT projects geared to improving the company's efficiencies.

One of the current projects is geared towards the design of an enterprise data warehouse in Azure Synapse Analytics.

Specifications:

• There will be millions of rows of data loaded to the data warehouse daily

• Staging tables must be optimized for data loading

Required:

• Design the staging tables

The team has come up with a list of options they are considering to use for the task at had but are not sure which to utilize. As you are the Azure expert, the team leader has asked for your opinion on which to use.

Which of the following should you recommend for the staging table design?

- ☑ **External table**

  (Incorrect)

- ☐ **Round-robin distributed table**

  (Correct)

- ☐ **Hash-distributed table**

- ☐ **Replicated table**

Explanation

You should recommend the Round-robin for the staging table. The load with CTAS is fast. Once the data is in the staging table, use `INSERT...SELECT` *to move the data to production tables.*

Round-robin tables

A round-robin table distributes table rows evenly across all distributions. The rows are distributed randomly. Loading data into a round-robin table is fast. But, queries can require more data movement than the other distribution methods.

Common distribution methods for tables

The table category often determines the optimal option for table distribution.

Fact Table category

Use hash-distribution with clustered columnstore index. Performance improves when two hash tables are joined on the same distribution column.

Dimension Table category

Use replicated for smaller tables. If tables are too large to store on each Compute node, use hash-distributed.

Staging Table category

Use round-robin for the staging table. The load with CTAS is fast. Once the data is in the staging table, use `INSERT...SELECT` to move the data to production tables.

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

Question 38: **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.

Which of the following services is an invalid input for an Azure Stream Analytics job? (Select all that apply)

- **Azure Cosmos DB**

  (Correct)

- **Azure Event Hubs**

- **Blob storage**

- **All of the listed options are valid inputs**

  (Incorrect)

Explanation
Azure Cosmos DB is a valid output for your Stream Analytics jobs but cannot be used as an input.

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.

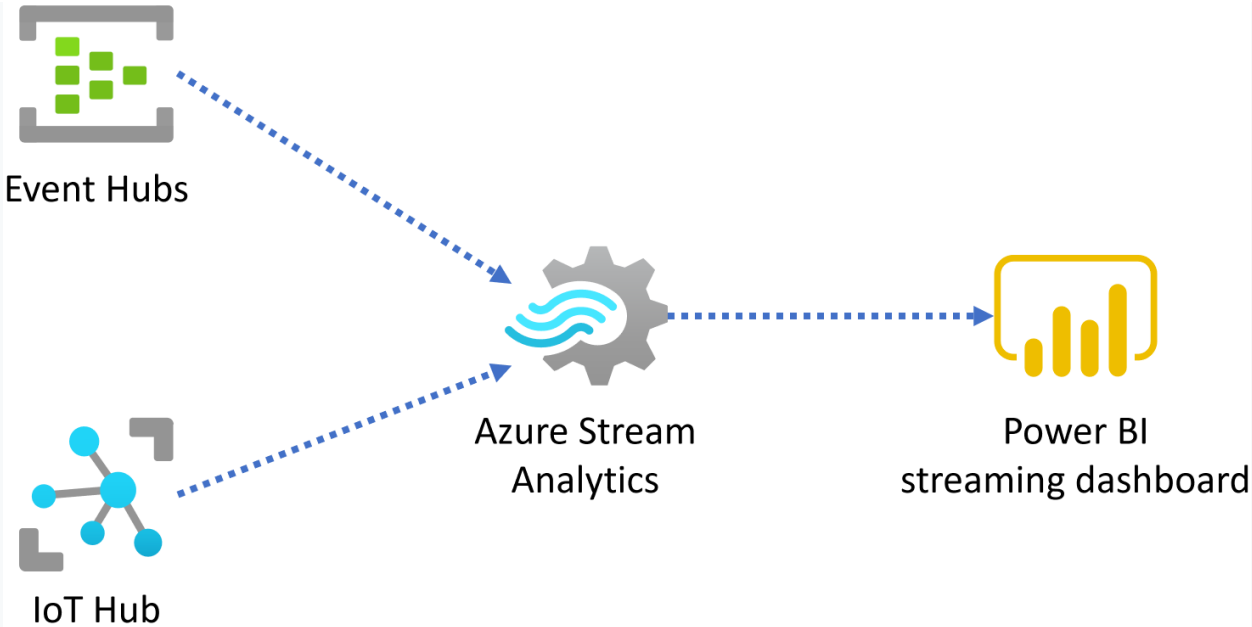
The fastest way to get streaming analytics running in Azure is to add an Azure Stream Analytics job to your application. Your Stream Analytics job would ingest the streaming data from one of the supported inputs and run real-time analytics queries against the streams. The built-in integration with Azure 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 39: **Incorrect**

Scenario: ACME Books Inc uses Azure Cosmos DB to store user profile data from their eCommerce site. The NoSQL document store provided by the Azure Cosmos DB SQL API provides the familiarity of managing their data using SQL syntax, while being able to read and write the files at a massive, global scale.

While ACME is happy with the capabilities and performance of Azure Cosmos DB, they are concerned about the cost of executing a large volume of analytical queries over multiple partitions (cross-partition queries) from their data warehouse. They want to efficiently access all the data without needing to increase the Azure Cosmos DB request units (RUs).

They have looked at options for extracting data from their containers to the data lake as it changes, through the Azure Cosmos DB change feed mechanism. The problem with this approach is the extra service and code dependencies and long-term maintenance of the solution. They could perform bulk exports from a Synapse Pipeline, but then they won't have the most up-to-date information at any given moment.

Required: Pick a solution will ensure all transactional data is automatically stored in a fully isolated column store without impacting the transactional workloads or incurring resource unit (RU) costs.

- **Enable Azure Private Link for SQL Database and enable the analytical store on their SQL Database containers.**

  (Incorrect)

- **Enable Azure Synapse Link for Cosmos DB and enable the analytical store on their Azure Cosmos DB containers.**

(Correct)

- **Enable Spark Pools for SQL Datawarehouse and enable the analytical store on their Azure Cosmos DB containers.**

- **None of the listed options.**

Explanation

The correct solution is to enable Azure Synapse Link for Cosmos DB and enable the analytical store on their Azure Cosmos DB containers. With this configuration, all transactional data is automatically stored in a fully isolated column store. This store enables large-scale analytics against the operational data in Azure Cosmos DB, without impacting the transactional workloads or incurring resource unit (RU) costs. Azure Synapse Link for Cosmos DB creates a tight integration between Azure Cosmos DB and Azure Synapse Analytics, which enables Tailwind Traders to run near real-time analytics over their operational data with no-ETL and full performance isolation from their transactional workloads.

By combining the distributed scale of Cosmos DB's transactional processing with the built-in analytical store and the computing power of Azure Synapse Analytics, Azure Synapse Link enables a Hybrid Transactional/Analytical Processing (HTAP) architecture for optimizing Tailwind Trader's business processes. This integration eliminates ETL processes, enabling business analysts, data engineers & data scientists to self-serve and run near real-time BI, analytics, and Machine Learning pipelines over operational data.

Azure Synapse Link for Cosmos DB is a cloud native HTAP (Hybrid Transactional and Analytical Processing) capability that allows us to run near-real time analytics over our data in Azure Cosmos DB.

This is possible through the Azure Cosmos DB Analytical Store, which provides us a way to perform near real-time analytics on our data without have to engineer our own ETL pipelines to do so.

https://medium.com/swlh/building-near-real-time-analytics-with-azure-synapse-link-for-azure-cosmos-db-eba35e759e1c

Hybrid Transactional and Analytical Processing enables businesses to perform analytics over a database system that is seen to provide transactional capabilities

without impacting the performance of the system. This enables organizations to use a database to fulfill both transactional and analytical needs to support near real-time analysis of operational data to make decisions about the information that is being analyzed.

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

Question 40: **Incorrect**
Identify the missing word(s) in the following sentence within the context of Microsoft Azure.

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

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

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

- ○

  Serverless architecture

- ◉

  No-SQL architecture

  (Incorrect)

- ○

  Lambda architecture

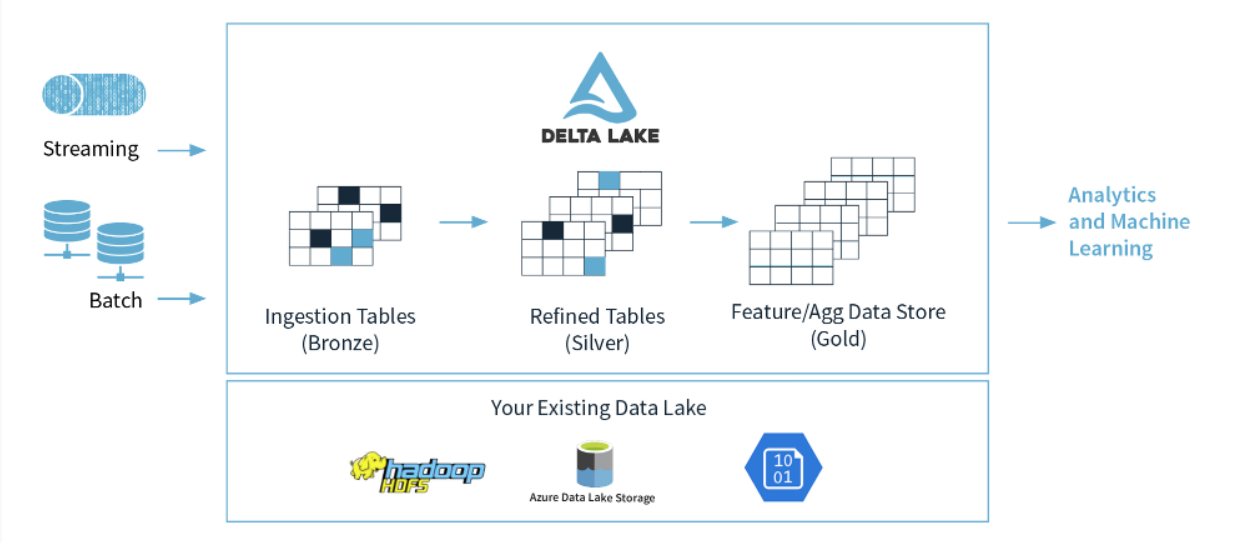
  (Correct)

- ○

Anaconda architecture

Explanation

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



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

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

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

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

Lambda architecture

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

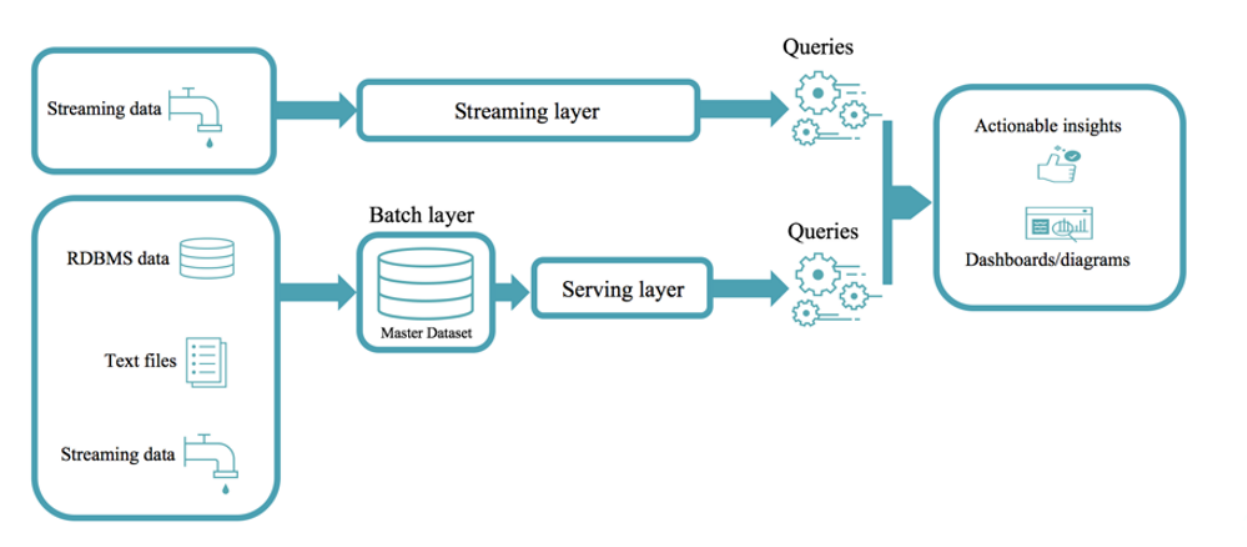
One drawback to this approach is that it introduces latency. If processing takes a few hours, a query may return results that are several hours old. Ideally, you would like to get some results in real time (perhaps with some loss of accuracy), and combine these results with the results from the batch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

The **lambda architecture** is a big data processing architecture that addresses this problem by combining both batch- and real-time processing methods. It features an append-only immutable data source that serves as system of record. Timestamped

events are appended to existing events (nothing is overwritten). Data is implicitly ordered by time of arrival.

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

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



Delta Lake architecture

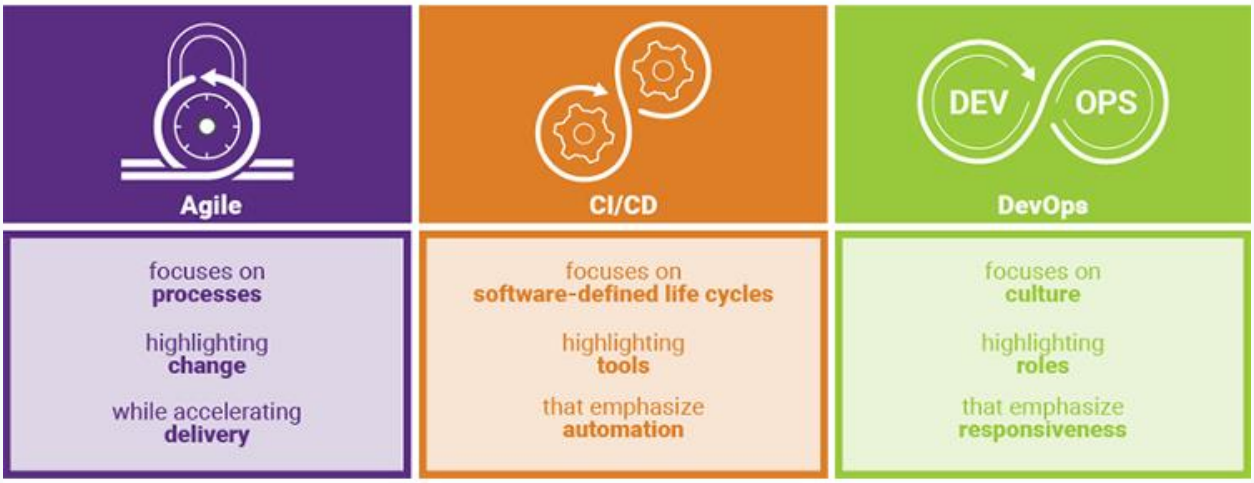
The Delta Lake Architecture is a vast improvement upon the traditional Lambda architecture. At each stage, we enrich our data through a unified pipeline that allows us to combine batch and streaming workflows through a shared filestore with ACID-compliant transactions.

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

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

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

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



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

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

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

Question 41: **Incorrect**
In which version of SQL Server was SSIS Projects introduced?

- **SQL Server 2014**
- **SQL Server 2012**

  (Correct)
-

SQL Server 2016

-

SQL Server 2008

(Incorrect)

Explanation
SSIS Projects was introduced in SQL Server 2012 and is the unit of deployment for SSIS solutions.

SQL Server 2012 was a major release for SSIS. It introduced the concept of the **project deployment model**, where entire projects with their packages are deployed to a server, instead of individual packages. The SSIS of SQL Server 2005 and 2008 is now referred to as the **(legacy) package deployment model**. SSIS 2012 made it easier to configure packages and it came with a centralized storage and management utility: the catalogue. We'll dive deeper into those topics later on in the tutorial.

SQL Server 2014 didn't have any changes for SSIS, but on the side new sources or transformations were added to the product. This was done by separate downloads trough CodePlex (an open-source code website) or through the SQL Server Feature Pack. Examples are the Azure feature pack (to connect to cloud sources and objects) and the balanced data distributor (to divide your data stream into multiple pipelines).

In SQL Server 2016 there were some updates to the SSIS product. Instead of deploying entire projects, you can new deploy packages individually again. There are additional sources – especially cloud and big data sources – and some important changes were made to the catalogue. You can find an overview of all new features here and here.

During all these years, SSIS has built itself a reputation for being a stable, robust and fast ETL tool with support for many sources. However, it's still mainly an on-premises solution, there is – at the time of writing – no real cloud alternative.

https://www.mssqltips.com/sqlservertutorial/9054/sql-server-integration-services-ssis-versions-and-tools/

Question 42: **Incorrect**
Apache Spark provides a powerful platform for performing data cleansing and transformation tasks on large volumes of data. You can then save the transformed data back to the data lake for downstream processing or ingestion into a data warehouse.

Azure Synapse Analytics provides Apache Spark pools that you can use to run Spark workloads to transform data as part of a data ingestion and preparation workload. You can use natively supported notebooks to write and run code on a Spark pool to prepare

data for analysis. You can then use other Azure Synapse Analytics capabilities such as SQL pools to work with the transformed data.

Which of the following should you do should you wish to use Spark to analyze data in a parquet file?

- **Load the parquet file into a dataframe**

  (Correct)

- **Import the data into a table in a serverless SQL pool**

- **Convert the data to CSV format**

- **None of the provided options are correct**

  (Incorrect)

Explanation

- *You can load data from files in many formats, including parquet, into a Spark dataframe.*
- You do not need to load the data into a SQL pool in order to analyze it with Spark.
- While you could load the data into Spark from a CSV file, you do not need to convert the data from parquet format in order to use it in Spark.

Modify and save dataframes

Apache Spark provides the *dataframe* object as the primary structure for working with data. You can use dataframes to query and transform data, and persist the results in a data lake. To load data into a dataframe, you use the **spark.read** function, specifying the file format, path, and optionally the schema of the data to be read. For example, the

following code loads data from all .csv files in the **orders** folder into a dataframe named **order_details** and then displays the first five records.

```
Python
order_details = spark.read.csv('/orders/*.csv', header=True, inferSchema=True)
display(order_details.limit(5))
```

Transform the data structure

After loading the source data into a dataframe, you can use the dataframe object's methods and Spark functions to transform it. Typical operations on a dataframe include:

- Filtering rows and columns
- Renaming columns
- Creating new columns, often derived from existing ones
- Replacing null or other values

In the following example, the code uses the `split` function to separate the values in the **CustomerName** column into two new columns named **FirstName** and **LastName**. Then it uses the `drop` method to delete the original **CustomerName** column.

```
Python
from pyspark.sql.functions import split, col

# Create the new FirstName and LastName fields
transformed_df = order_details.withColumn("FirstName",
split(col("CustomerName"), " ").getItem(0)).withColumn("LastName",
split(col("CustomerName"), " ").getItem(1))

# Remove the CustomerName field
transformed_df = transformed_df.drop("CustomerName")

display(transformed_df.limit(5))You can use the full power of the Spark SQL
library to transform the data by filtering rows, deriving, removing, renaming
columns, and any applying other required data modifications.
```

Save the transformed data

After your dataFrame is in the required structure, you can save the results to a supported format in your data lake.

The following code example saves the dataFrame into a *parquet* file in the data lake, replacing any existing file of the same name.

```
1. Python
2. transformed_df.write.mode("overwrite").parquet('/transformed_data/orders.parquet')
3. print ("Transformed data saved!")
```

Note: The Parquet format is typically preferred for data files that you will use for further analysis or ingestion into an analytical store. Parquet is a very efficient format that is supported by most large scale data analytics systems. In fact, sometimes your data transformation requirement may simply be to convert data from another format (such as CSV) to Parquet!

https://learn.microsoft.com/en-us/azure/databricks/getting-started/dataframes-python

Question 43: **Incorrect**
Which notebook format is used in Databricks?

- **DBC**

  (Correct)

- `.dbrk`

- `.spark`

- `.notebook`

  (Incorrect)

Explanation

The supported Databricks notebook format is the DBC file type.

Notebook external formats

Azure Databricks supports several notebook external formats:

• Source file: A file containing only source code statements with the extension `.scala`, `.py`, `.sq`l, or `.r`.

• HTML: An Azure Databricks notebook with the extension `.html`.

• DBC archive: A Databricks archive.

• IPython notebook: A Jupyter notebook with the extension `.ipynb`.

• RMarkdown: An R Markdown document with the extension `.Rmd`.

https://docs.microsoft.com/en-us/azure/databricks/notebooks/notebooks-manage

Question 44: **Incorrect**
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.

Which of the following functions should you used if you needed to compare approximate production volumes by product while optimizing query response time?

- `APPROX_COUNT_DISTINCT`

  (Correct)

- `NTILE`

- `COUNT`

-

  `EXACTMATCH` = `FALSE`

  (Incorrect)

Explanation

- *`APPROX_COUNT_DISTINCT` returns an approximate count within 2% of the actual count while optimizing for minimal response time.*
- The `COUNT` function will return accurate counts, but may take longer than other options.
- The `NTILE` function returns the percentile for a row in a partition based on a numeric ranking.
- `EXACTMATCH` = `FALSE` is not a valid option

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 45: **Incorrect**
There are two concepts within Apache Spark Pools in Azure Synapse Analytics, namely Spark Pools and Spark Instances.

Which of the following attributes belong to Spark Instances? (Select three)

- ☐

  Exists as Metadata

- ☐

  Permissions can be applied

- ☐

Creates a Spark Instance

- ☐

  Reusable

  (Correct)

- ☑

  Created when connected to Spark Pool, Session, or Job

  (Correct)

- ☐

  Multiple users can have access

  (Correct)

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

Spark Pools:

• Exists as Metadata

• Creates a Spark Instance

• No costs associated with creating Pool

• Permissions can be applied

• Best practices

Spark Instances:

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

• Multiple users can have access

• Reusable

A Spark pool is created in the Azure portal. It is 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; no resources are consumed, running, or charged for. A Spark pool has series of properties that control the characteristics of a Spark instance; these characteristics include but are not limited to name, size, scaling behaviour, time to live.

As there is no resource cost associated with creating Spark pools, any number of pools 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.

An example of Spark Pools:

• You create a Spark pool called SP1; it has a fixed cluster size of 20 nodes.

• You submit a notebook job, J1 that uses 10 nodes, a Spark instance, SI1 is created to process the job.

• You now submit another job, J2, that uses 10 nodes because there is still capacity in the pool and the instance, the J2, is processed by SI1.

• If J2 had asked for 11 nodes, there would not have been capacity in SP1 or SI1. In this case, if J2 comes from a notebook, then the job will be rejected; if J2 comes from a batch job, then it will be queued.

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, then if there is capacity in the pool, the existing Spark instance also has capacity then the existing instance will process the job; if not and there is capacity at the pool level, then a new Spark instance will be created.

An example of a Spark Instance:

• You create a Spark pool call SP2; it has an autoscale enabled 10 – 20 nodes

• You submit a notebook job, J1 that uses 10 nodes, a Spark instance, SI1, is created to process the job.

• You now submit another job, J2, that uses 10 nodes, because there is still capacity in the pool the instance auto grows to 20 nodes and processes J2.

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

Question 46: **Incorrect**
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.

Clicking Debug will provision the Spark clusters required to interact with the Mapping Data Flow transformations. If you select `AutoResolveIntegrationRuntime`, what will be the result? (Select all that apply)

- ☐ **All the listed options.**
- ☐ **Data engineers can develop data transformation logic with or without writing code.**
- ☐ **It typically takes 5-7 minutes for the cluster to spin up.**

  (Correct)
- ☑ **The number of rows that are returned within the data previewer are fixed by the AutoResolve Agent.**

  (Incorrect)
- ☐ **None of the listed options.**
- ☐ **A cluster with eight cores that will be available with a time to live value of 60 minutes.**

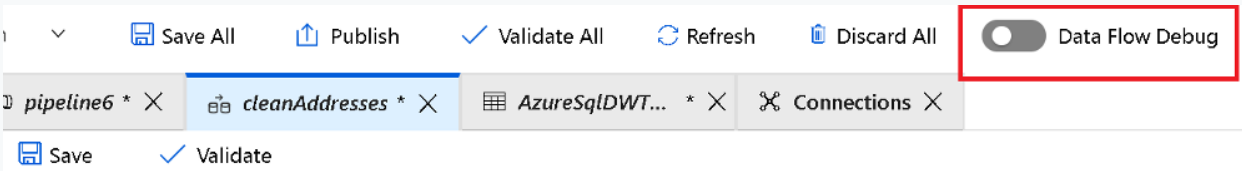
(Correct)

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.

During the building of Mapping Data Flows, you can interactively watch how the data transformations are executing so that you can debug them. To use this functionality, it is first necessary to turn on the "Data Flow Debug" feature.



Clicking Debug will provision the Spark clusters required to interact with the Mapping Data Flow transformations. On turning Debug on, you will be prompted to select the Integration Runtime that you require to use in the environment. **If you select AutoResolveIntegrationRuntime, a cluster with eight cores that will be available with a time to live value of 60 minutes.**

Note: It typically takes 5-7 minutes for the cluster to spin up. With this mode on and the Spark clusters running, you are able to build your data flow step by step and view the data as it runs through each transformation phase.

A Data Preview tab is available in Debug mode that will allow you to view the data at each stage of the pipeline. You can view the data after each transformation. The data previewer also provides the ability to actions on the data such as looking at descriptive statistics of the data, or the ability to modify the data.

movieId	title	genres	year
1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	1995
2	Jumanji (1995)	Adventure|Children|Fantasy	1995
3	Grumpier Old Men (1995)	Comedy|Romance	1995
4	Waiting to Exhale (1995)	Comedy|Drama|Romance	1995
5	Father of the Bride Part II (1995)	Comedy	1995
6	Heat (1995)	Action|Crime|Thriller	1995
7	Sabrina (1995)	Comedy|Romance	1995
8	Tom and Huck (1995)	Adventure|Children	1995
9	Sudden Death (1995)	Action	1995

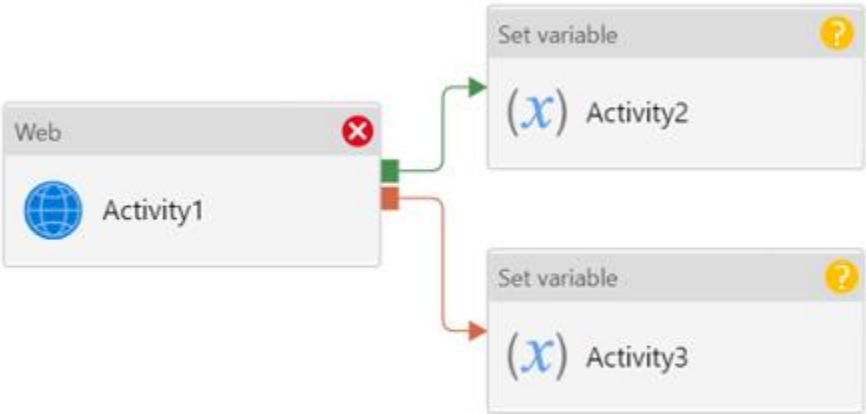
Finally, you can use the debug settings to control the number of rows that are returned within the data previewer.

Note: It is recommended to limit the number of rows that returns enough to enable you to confirm that the data is correct. The bigger the data set, the longer it takes to return the results back. You can also use the Debug settings to specify any parameter values that should be used during the execution of the pipeline.

https://docs.microsoft.com/en-us/azure/data-factory/concepts-data-flow-debug-mode

Question 47: **Incorrect**

Scenario: We are working on a project which has a pipeline with 3 activities, where Activity1 has a success path to Activity2 and a failure path to Activity3. *Activity1 fails, Activity2 is skipped, and Activity3 succeeds.*



What will the result be of the pipeline?

- **This pipeline reports success.**

  (Incorrect)

- **This pipeline reports skipped.**

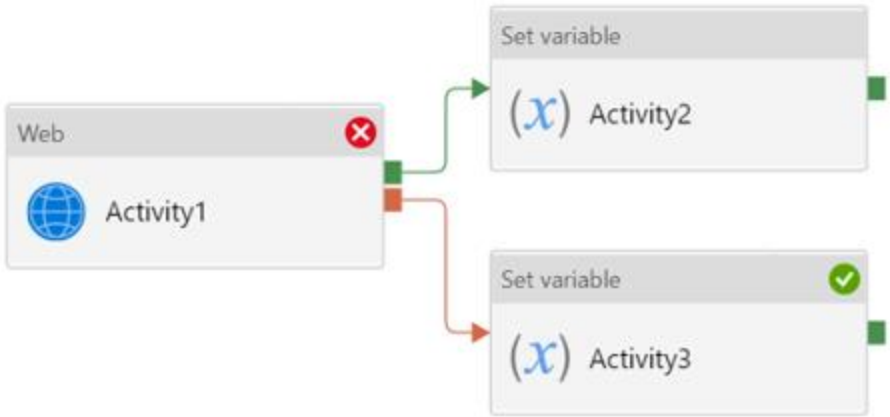
- **This pipeline reports failure.**

  (Correct)

- **This pipeline reports completed.**

Explanation

Let's say we have a pipeline with 3 activities, where Activity1 has a success path to Activity2 and a failure path to Activity3. If Activity1 fails and Activity3 succeeds, the pipeline will fail. The presence of the success path alongside the failure path changes the outcome reported by the pipeline.



Activity1 fails, Activity2 is skipped, and Activity3 succeeds. The pipeline reports failure.

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

[?] is an encryption mechanism to help you protect Azure Synapse Analytics. It will protect Azure Synapse Analytics against threats of malicious offline activity. The [?] way will do so by is encrypting data at rest. [?] performs real-time encryption as well as decryption of the database, associated backups, and transaction log files at rest without you having to make changes to the application.

- **Row-level security**

- **Database Encryption Key**

- Transparent Data Encryption

  (Correct)

- Column-level security

- Table-level security

  (Incorrect)

- Dynamic Data Masking

Explanation

Transparent Data Encryption

Transparent Data Encryption (TDE) is an encryption mechanism to help you protect Azure Synapse Analytics. It will protect Azure Synapse Analytics against threats of malicious offline activity. The way TDE will do so, is by encrypting data at rest. TDE performs real-time encryption as well as decryption of the database, associated backups, and transaction log files at rest without you having to make changes to the application. In order to use TDE for Azure Synapse Analytics, you will have to manually enable it.

What TDE does is performing I/O encryption and decryption of data at the page level in real time. When a page is read into memory, it is decrypted. It is encrypted before writing it to disk. TDE encrypts the entire data base storage, using a symmetric key called a Database Encryption Key (DEK). When you start up a database, the encrypted Database Encryption Key is decrypted when it then will be used for decryption and re-encryption of the database files in the SQL Server database engine. The DEK is protected by the Transparent Data Encryption Protector. This protector can be either a service-managed certificated, which is referred to as service-managed transparent data encryption, or an asymmetric key that is stored in Azure Key Vault (customer-managed transparent data encryption).

What is important to understand is that for Azure Synapse Analytics, this TDE protector is set on the server level. There it is inherited by all the databases that are attached or aligned to that server. The term server refers both to server and instance.

Service-managed transparent data encryption

As stated above, the DEK that is protected by the Transparent Encryption protector can be service-managed certificated which we call service-managed TDE. When you look in Azure, that default setting means that the DEK is protected by a built-in certificate unique for each server with encryption algorithm AES256. When a database is in a geo-replicated relationship then primary and the geo-secondary database are protected by the primary database's parent server key. If the databases are connected to the same server, they will also have the same built-in AES 256 certificate. As Microsoft we automatically rotate the certificates in compliance with the internal security policy. The root key is protected by a Microsoft internal secret store. Microsoft also seamlessly moves and manages the keys as needed for geo-replication and restores.

Transparent data encryption with bring your own key for customer-managed transparent data encryption

As stated above, the DEK that is protected by the Transparent Data Encryption Protector can also be customer managed by bringing an asymmetric key that is stored in Azure Key Vault (customer-managed transparent data encryption). This is also referred to as Bring Your Own Key (BYOK) support for TDE. When this is the scenario that is applicable to you, the TDE Protector that encrypts the DEK is a customer-managed asymmetric key. This is stored in your own and managed Azure Key Vault. Azure Key Vault is Azure's cloud-based external key management system. This managed key never leaves the key vault. The TDE Protector can be generated by the key vault. Another option is to transfer the TDE Protector to the key vault from, for example, an on-premise hardware security module (HSM) device. Azure Synapse Analytics needs to be granted permissions to the customer-owned key vault in order to decrypt and encrypt the DEK. If permissions of the server to the key vault are revoked, a database will be inaccessible, and all data is encrypted.

By using Azure Key Vault integration for TDE, you have control over the key management tasks such as key rotations, key backups, and key permissions. It also enables you for auditing and reporting on all the TDE protectors when using the Azure Key Vault functionality. The reason for using Key Vault is that it provides you with a central key management system where tightly monitored HSMs are leveraged. It also enables you to separate duties of management of keys and data in order to meet compliance with security policies.

Moving a transparent data encryption protected database

In some use cases you need to move a database that is protected with TDE. Within Azure, there is no need to decrypt the databases. The TDE settings on the source database or primary database, will be inherited on the target. Some of the operations within Azure that inherited the TDE are:

• Geo-restore

• Self-service point-in-time restore

• Restoration of a deleted database

• Active geo-replication

• Creation of a database copy

• Restore of backup file to Azure SQL Managed Instance

If you export a TDE-protected database, the exported content is not encrypted. This will be stored in an unencrypted BACPAC file. You need to make sure that you protect this BACPAC file and enable TDE as soon as the import of the bacpac file in the new database is finished.

Securing your credentials through linked services with TokenLibrary for Apache Spark

It is quite a common pattern to access data from external sources. Unless the external data source allows anonymous access, it is highly likely that you need to secure your connection with a credential, secret, or connection string.

Within Azure Synapse Analytics, the integration process is simplified by providing linked services. Doing so, the connection details can be stored in the linked service or an Azure Key Vault. If the Linked Service is created, Apache spark can reference the linked service to apply the connection information in your code. When you want to access files from the Azure Data Lake Storage Gen 2 within your Azure Synapse Analytics Workspace, it uses AAD passthrough for the authentication. Therefore, there is no need to use the TokenLibrary. However, to connect to other linked services, you are enabled to make a direct call to the TokenLibrary.

https://docs.microsoft.com/en-us/azure/synapse-analytics/sql-data-warehouse/sql-data-warehouse-encryption-tde-tsql

Question 49: **Incorrect**
In Azure Data Factory, you can raise alerts based upon metrics outputted by the monitoring service. Alerts provide information on a variety of scenarios such as … (Select all that apply)

- ☐

  Cancelled SSIS integration runtime start metrics

  (Correct)

- ☐

  Integration runtime CPU utilization

  (Correct)

- ☐

  Failed pipelines

  (Correct)

- ☐

  Integration runtime available memory

  (Correct)

- ☐

  Large factory sizes

  (Correct)

- ☐

  Failed activity run metrics

  (Correct)

- ☑

  Failed trigger runs metrics

  (Correct)

- ☐

  Successful trigger runs metrics

  (Correct)

- ☐

Cancelled activity runs

(Correct)

Explanation
In Azure Data Factory, you can raise alerts based upon metrics outputted by the monitoring service. Alerts allow you to get alerted for a variety of scenarios such as, but not limited to, failed pipelines, large factory sizes, and integration runtime CPU utilization.

DEFAULT DIRECTORY

Add criteria

Select one metric to set up the alert condition.

Metrics ↑↓
Cancelled activity runs metrics
Cancelled pipeline runs metrics
Cancelled SSIS integration runtime start metrics
Cancelled SSIS package execution metrics
Cancelled trigger runs metrics
Elapsed Time Pipeline Runs Metrics
Failed activity runs metrics
Failed pipeline runs metrics
Failed SSIS integration runtime start metrics
Failed SSIS package execution metrics
Failed trigger runs metrics
Integration runtime available memory
Integration runtime available node count
Integration runtime CPU utilization
Integration runtime queue duration
Integration runtime queue length
Maximum allowed entities count
Maximum allowed factory size (GB unit)
Stuck SSIS integration runtime stop metrics
Succeeded activity runs metrics
Succeeded pipeline runs metrics
Succeeded SSIS integration runtime start metrics
Succeeded SSIS integration runtime stop metrics
Succeeded SSIS package execution metrics
Succeeded trigger runs metrics
Total entities count
Total factory size (GB unit)

Alerts in the monitoring experience are based upon high-level metrics such as pipeline failures. For custom alerting on specific conditions that may occur within a pipeline or based upon data quality, it is recommended to configure these using a pipeline activity.

To get started, go to the **Monitor** tab and select **Alerts & metrics**.

https://azure.microsoft.com/en-us/blog/create-alerts-to-proactively-monitor-your-data-factory-pipelines/

Question 50: **Incorrect**
Scenario: You are working on a project and you have been tasked with starting up a data platform service to execute as Spark job. The objective on this job is to ingest and process data and then shut down the service after the job is complete.

Which of the following would be the best compute resource to use?

- **Azure-SSIS Runtime**

None of the listed options

- **On-demand HDInsight cluster**

  (Correct)

- **Azure Databricks**

- **HDInsight**

  (Incorrect)

Explanation
On-demand HDInsight cluster service is used to execute as Spark job to ingest and process data and then shut down the service after the job is complete.

Azure Data Factory can call on compute resources to process data by a data platform service that may be better suited to the job. A great example of this is that Azure Data Factory can create a pipeline to an analytical data platform such as Spark pools in an Azure Synapse Analytics instance to perform a complex calculation, which generates new data. This data is then ingested back into the pipeline for further downstream processing. There a wide range of computing resource, and the associated activities that they can perform as shown in the following table:

Compute environment	activities
On-demand HDInsight cluster or your own HDInsight cluster	Hive, Pig, Spark, MapReduce, Hadoop Streaming
Azure Batch	Custom activities
Azure Machine Learning Studio Machine	Learning activities: Batch Execution and Update Resource
Azure Machine Learning	Azure Machine Learning Execute Pipeline
Azure Data Lake Analytics	Data Lake Analytics U-SQL
Azure SQL, Azure SQL Data Warehouse, SQL Server	Stored Procedure
Azure Databricks	Notebook, Jar, Python
Azure Function	Azure Function activity

As shown in the table above, Spark is only mentioned in On-demand or your own HDInsight cluster.

https://www.red-gate.com/simple-talk/cloud/infrastructure-as-a-service/automating-azure-creating-demand-hdinsight-cluster/

Question 51: **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:

“The Union transformation that combines multiple data streams.”

- **Schema modifier transformations**
- **None of the listed options**

  (Incorrect)
- **Multiple inputs/outputs transformations**

  (Correct)
- **Row modifier transformations**

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 52: **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:

"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 Named**

  (Correct)

- **Interactive**

- **Global**

-

  Cluster Scoped

  (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 53: **Incorrect**
Scenario: You are working on a new project and you are in a meeting to discuss which Azure data platform technology is best for your company.

Requirement: A globally distributed, multimodel database that can perform queries in less than a second.

Which of the following should you choose?

- **Azure SQL Database**
- **Azure Data Factory**
- **Azure Cosmos DB**

  (Correct)
- **Azure SQL Data Warehouse**
-

Azure Databricks

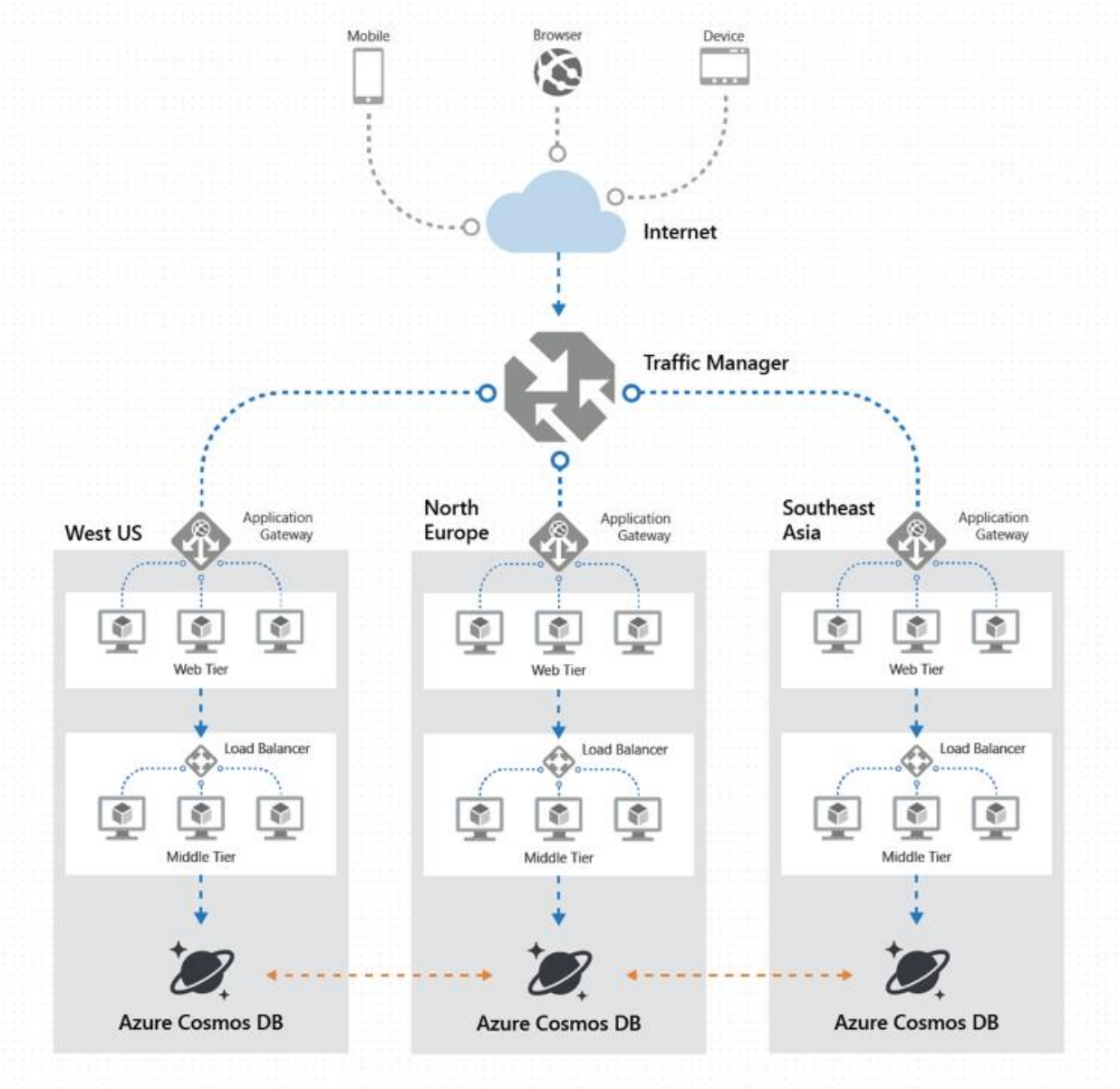
- **Azure SQL on VM**

  (Incorrect)

Explanation
Azure Cosmos DB is a globally distributed, multimodel database that can offer subsecond query performance. Azure Cosmos DB transparently replicates the data to all the regions associated with your Cosmos account. Azure Cosmos DB is a globally distributed database service that's designed to provide low latency, elastic scalability of throughput, well-defined semantics for data consistency, and high availability. In short, if your application needs fast response time anywhere in the world, if it's required to be always online, and needs unlimited and elastic scalability of throughput and storage, you should build your application on Azure Cosmos DB.

You can configure your databases to be globally distributed and available in any of the Azure regions. To lower the latency, place the data close to where your users are. Choosing the required regions depends on the global reach of your application and where your users are located. Cosmos DB transparently replicates the data to all the regions associated with your Cosmos account. It provides a single system image of your globally distributed Azure Cosmos database and containers that your application can read and write to locally.

With Azure Cosmos DB, you can add or remove the regions associated with your account at any time. Your application doesn't need to be paused or redeployed to add or remove a region.



https://docs.microsoft.com/en-us/azure/cosmos-db/distribute-data-globally

Question 54: **Incorrect**

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

A common use with [?] is to take the data that is shared and use it as a source into Azure Data Factory pipelines to use with your own internal data.

- **Azure Managed SQL Warehouse**

- **Azure SQL Database**

- **Azure Data Lake Storage**

- **Azure Data Share**

  (Correct)

- **Azure Databricks**

  (Incorrect)

Explanation
A common use with Azure Data Share is to take the data that is shared and use it as a source into Azure Data Factory pipelines to use with your own internal data.

Azure Data Factory will give you the opportunity to perform code-free ETL/ELT, which will result in a comprehensive overview of your data pipelines. As a data engineer, this gives you the confidence to work with more data.

In order to start creating a pipeline, we first need to set up linked services in Azure Data Factory. Linked services define the connection information for data factory to the external resources you want to connect with, for example an Azure SQL Database or Azure Data Lake Storage.

The connection to the data source and dataset that is linked to that linked service, represents the data structure. For example, an Azure Data Lake Storage linked service will specify the connection string to the Azure Data Lake Storage account.

The connection string can be passed through to Azure Data Factory by creating a linked service.

The purpose of linked services, is to represent and show data store as well as compute resources that need to be hosted for the execution of a pipeline or activity.

Using the code-free User Experience of Azure Data Factory from the Azure portal makes it easy for the non-coder to develop linked services.

Currently, Azure Data Factory supports over 85 of these connectors.

A pipeline in Azure Data Factory is a logical grouping of activities such as copy in order to perform a task. The activity defines the operation that you're performing on the data (therefore, a copy means copying the same data to another data store).

The dataset that you're using is pointing to the data that you're going to use from the linked service.

Therefore, if you have linked a SQL DB, which contains a database, which contains tables, you can select the table that you want to copy.

In doing so, the data from that table will be copied to an Azure Data Lake storage Account.

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

Question 55: **Incorrect**
What Transact-SQL function is used to perform a `HyperLogLog` function?

- `HYPER_LOG_LOG`
- `APPROX_COUNT_DISTINCT`

  (Correct)
- `COUNT_DISTINCT_APPROX`
- `COUNT`
- **None of the listed options.**

  (Incorrect)

Explanation
The `APPROX_COUNT_DISTINCT` function is used to perform a `HyperLogLog` function.

It is not uncommon for data engineers, data analysts, and data scientists alike to perform exploratory data analysis to gain an understanding of the data that they are working with. Exploratory data analysis can involve querying metadata about the data that is stored within the database, to running queries to provide a statistics information about the data such as average values for a column, through to distinct counts. Some of the activities can be time consuming, especially on large data sets.

For example, performing a distinct count of values in a Billion plus row table can be an expensive operation that takes time to resolve. As exploratory data analysis sometime doesn't require accurate information, there is a solution.

Azure Synapse Analytics supports Approximate execution using Hyperlog accuracy to reduce latency when executing queries with large datasets. Approximate execution is used to speed up the execution of queries with a compromise for a small reduction in accuracy. So if it takes too long to get basic information about the data in a large data set as you are exploring data of a big data set, then you can use the `HyperLogLog` accuracy and will return a result with a 2% accuracy of true cardinality on average. This is done by using the `APPROX_COUNT_DISTINCT` Transact-SQL function.

https://www.slideshare.net/jamserra/azure-synapse-analytics-overview

Question 56: **Incorrect**
Spark is a distributed computing environment. Therefore, work is parallelized across executors. At which two levels does this parallelization occur?

- ○

  The Executor and the Task

- ○

  The Executor and the Slot

  (Correct)

- ○

  The Driver and the Executor

- ◉

The Slot and the Task

(Incorrect)

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

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

Executor

The **executors** are responsible for actually executing the work that the **driver** assigns them. This means, each executor is responsible for only two things:

1.Executing code assigned to it by the driver

2.Reporting the state of the computation, on that executor, back to the driver node

Cores/Slots/Threads

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

• So for example:

• If we have **3** identical home desktops (*nodes*) hooked up together in a LAN (like through your home router), each with i7 processors (**8** cores), then that's a **3** node Cluster:

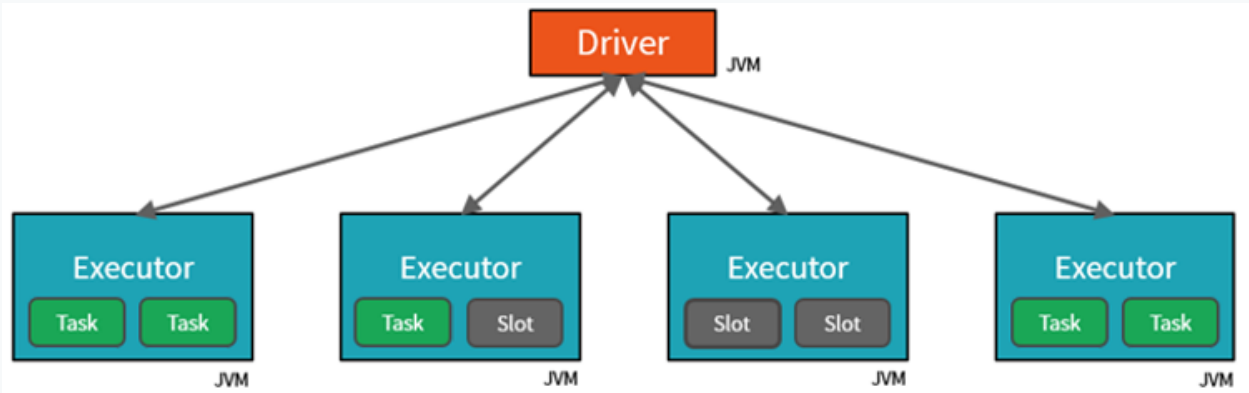
• **1** Driver node

• **2** Executor nodes

• The **8 cores per Executor node** means **8 Slots**, meaning the driver can assign each executor up to **8 Tasks**

• The idea is, an i7 CPU Core is manufactured by Intel such that it is capable of executing it's own Task independent of the other Cores, so **8 Cores = 8 Slots = 8 Tasks in parallel**

For example: the diagram below is showing 2 Core Executor nodes:



https://www.rakirahman.me/spark-certification-study-guide-part-1/

Question 57: **Incorrect**

Today, massive amounts of real-time data are generated by connected applications, Internet of Things (IoT) devices and sensors, and various other sources. The proliferation of streaming data sources has made the ability to consume and make informed decisions from these data in near-real-time an operational necessity for many organizations.

Some typical examples of streaming data workloads include:

- Online stores analyzing real-time clickstream data to provide product recommendations to consumers as they browse the website.
- Manufacturing facilities using telemetry data from IoT sensors to remotely monitor high-value assets.
- Credit card transactions from point-of-sale systems being scrutinized in real-time to detect and prevent potentially fraudulent activities.

Which of the following is the correct definition of stream processing?

- **Data is collected in a temporary store, and records are processed in batch sizes determined when the stream processing is setup to maximize efficiency; they are not processed as received**
-

Data is collected in a temporary store, and all records are processed together as a batch

(Incorrect)

- **Data that is incomplete or contains errors is redirected to separate storage for correction by a human operator**

- **Data is processed continually as new data records arrive**

  (Correct)

Explanation

- *Stream processing is used to continually process new data as it arrives.*
- Collecting and processing data periodically as a batch is referred to as batch processing.
- When incomplete or erroneous data is often logged, but isn't usually analyzed as a stream.

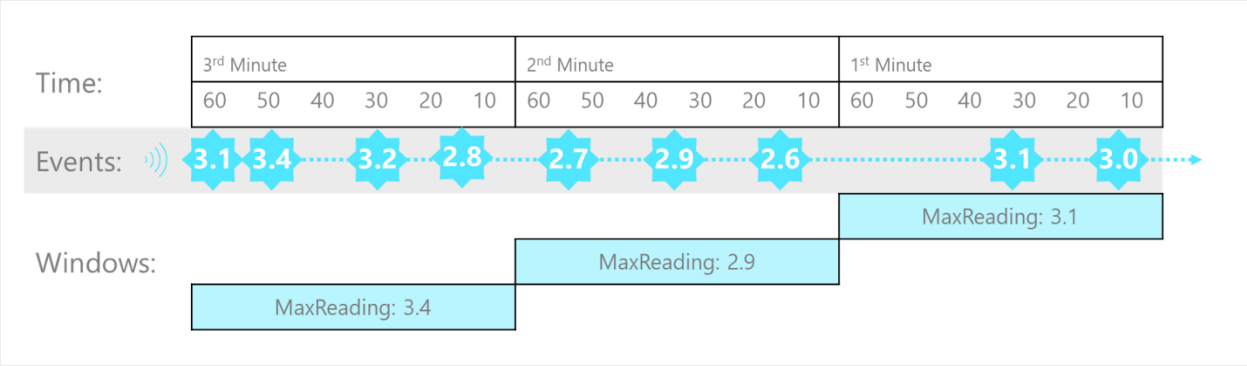
Understand window functions

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.

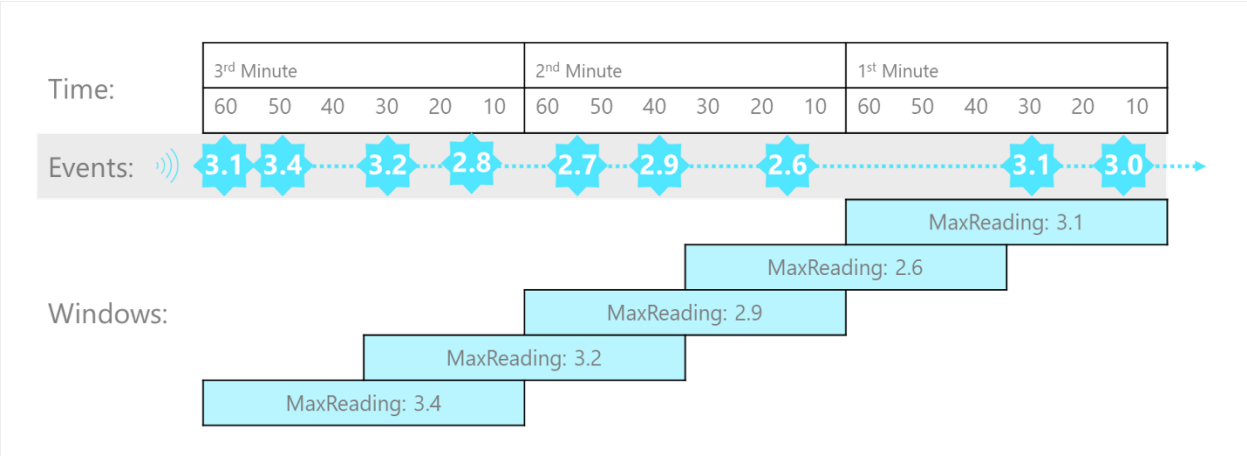


The Tumbling window example, represented by the following query, finds the maximum reading value in each one-minute window. Windowing functions are applied in Stream Analytics jobs using the GROUP BY clause of the query syntax. The `GROUP BY` clause in the following query contains the `TumblingWindow()` 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.



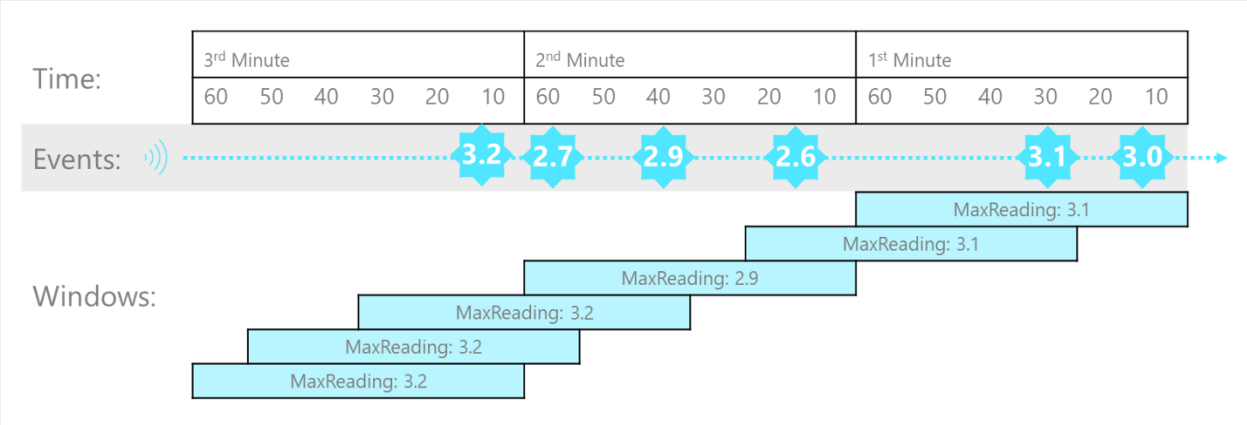
To create a hopping window, you must specify three parameters. The first parameter indicates the time unit, such as second, minute, or hour. The following parameter sets the window size, which designates how long each window lasts. The final required parameter is the hop size, which specifies how much 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.

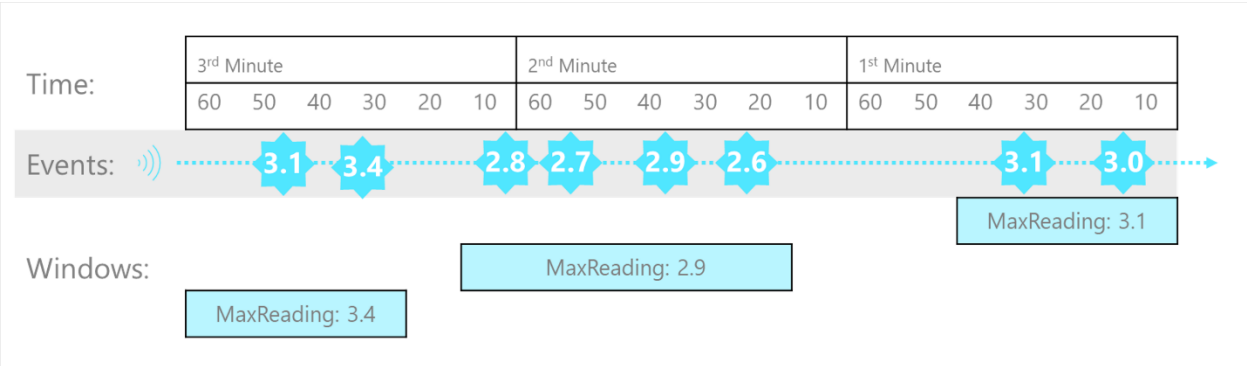


The following query uses the `SlidingWindow()` function to find the maximum reading value in each one-minute window in which an event occurred.

```
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

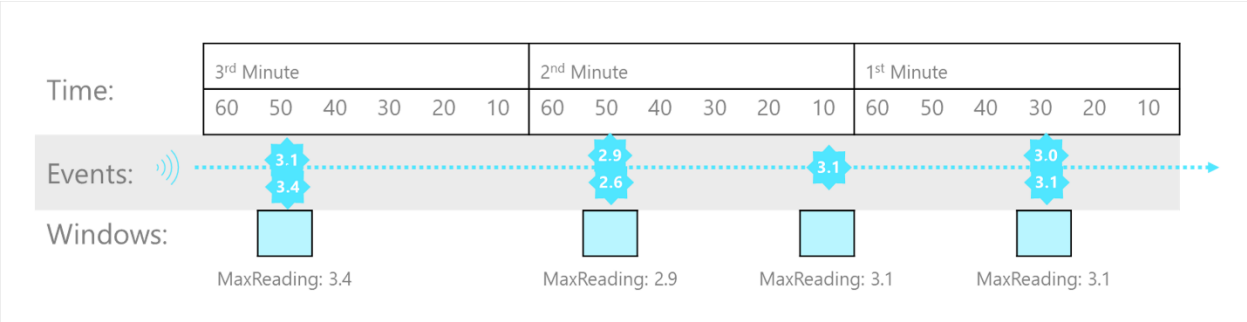
window will be extended to incorporate the new event. However, if no other events occur within the specified timeout period, the window will be closed at the timeout. If events keep happening within the specified timeout, the session window will extend until the maximum duration is reached.

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 58: **Correct**
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, 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.

Two of the core features of Delta Lake are performing `UPSERT`s and Time Travel operations.

What does the `UPSERT` command do?

- ○

  The command will `INSERT` a row and if the row already exists, append a new row in the table with an update notation.

- ◉

  The command will `INSERT` a row and if the row already exists, `UPDATE` the row.

(Correct)

- **The command will `INSERT` a column and if the column already exists, add a new column in the table with an update notation.**

- **The command will `INSERT` a column and if the column already exists, `UPDATE` the column.**

- **The command will `INSERT` a table and if the table already exists, `UPDATE` the table.**

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).

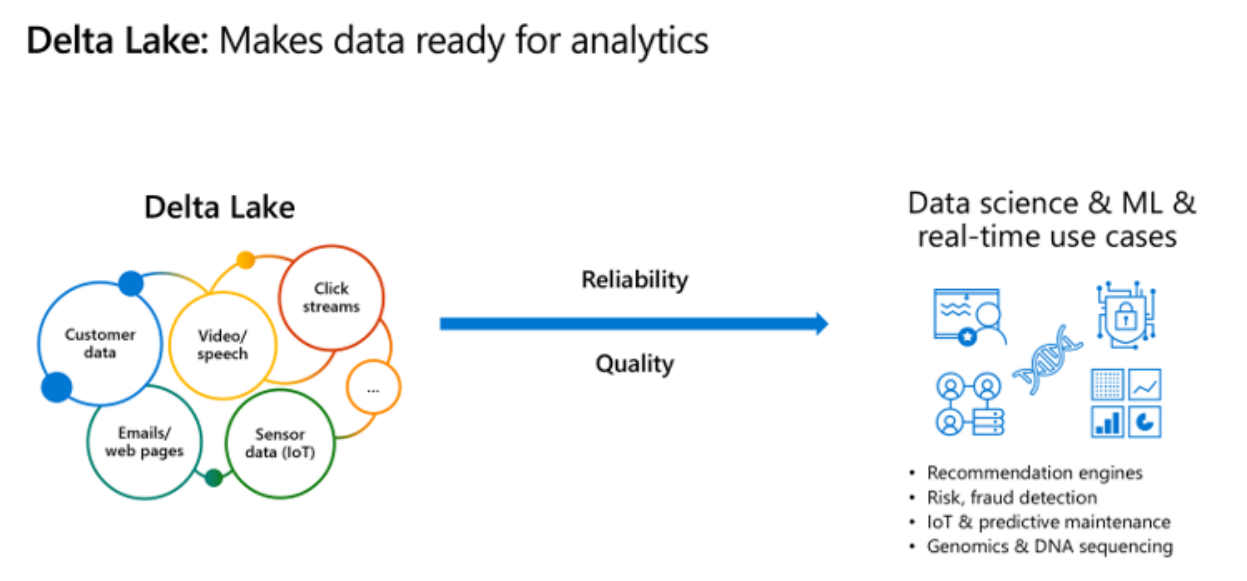
• **Partitioning also known as "poor man's indexing"**- breaks down if you picked the wrong fields or when data has many dimensions, high cardinality columns.

• **No caching** - cloud storage throughput is low (cloud object storage is 20-50MB/s/core vs 300MB/s/core for local SSDs).

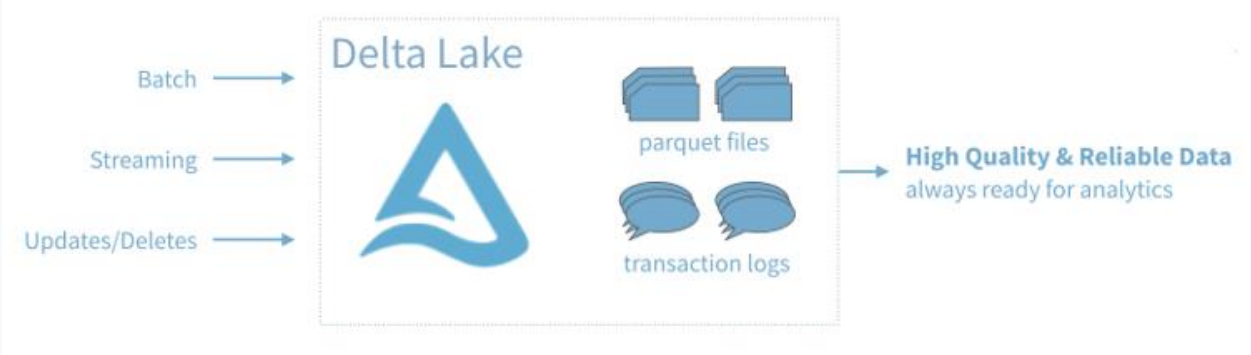
The solution: Delta Lake

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:

• `INSERT` 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:

```
SQL
MERGE INTO customers -- Delta table
USING updates
ON customers.customerId = source.customerId
WHEN MATCHED THEN
UPDATE SET address = updates.address
WHEN NOT MATCHED
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 59: **Incorrect**

Which method for renaming a `DataFrame` column is incorrect?

- `df.toDF("dateCaptured")`
- `df.select(col("timestamp").alias("dateCaptured"))`
- **All are correct.**
- `df.alias("timestamp", "dateCaptured")`

  (Correct)
- **All are incorrect.**

  (Incorrect)

Explanation

The `DataFrame` does not contain an alias method for a column. `df.alias("timestamp", "dateCaptured")` is an incorrect method for renaming a `DataFrame` column.

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

Question 60: **Correct**
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.

The team is new to Apache Spark and Teresa has asked you to speak to the team to help them understand some key concepts.

Which definition best describes Apache Spark?

- ○

  A highly scalable relational database management system.

- ○

  A 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.

- ○

  A virtual server with a Python runtime.

- ◉

  A distributed platform for parallel data processing using multiple languages.

  (Correct)

Explanation
Spark provides a highly scalable distributed platform on which you can run code written in many languages to process data.

Azure Databricks offers a highly scalable platform for data analytics and processing using Apache Spark.

Spark is a flexible platform that supports many different programming languages and APIs. Most data processing and analytics tasks can be accomplished using the **Dataframe** API, which is what we'll focus on in this module.

Apache Spark is a key technology used in big data analytics, and the Spark support in Azure Databricks enables you to combine big data processing in Spark with large-scale data analytics.

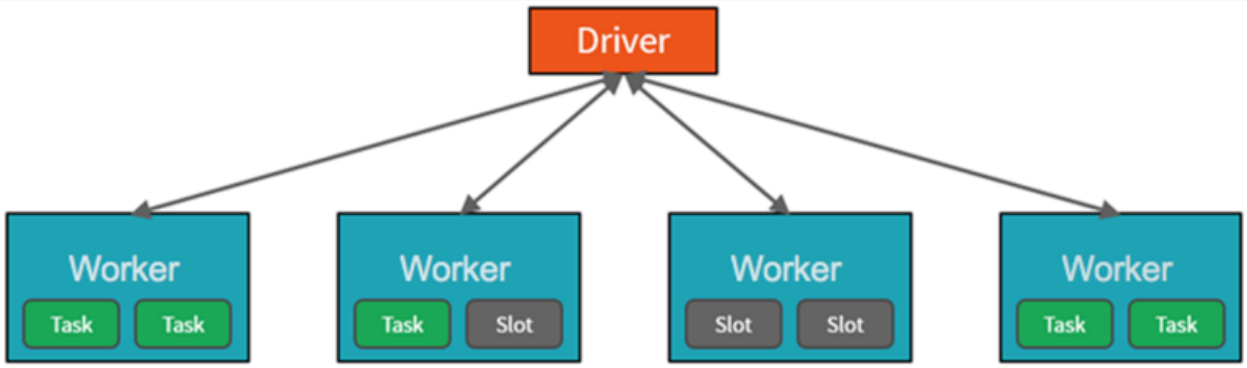
Get to know Spark

To gain a better understanding of how to process and analyze data with Apache Spark in Azure Databricks, it's important to understand the underlying architecture.

High-level overview

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

In Databricks, the notebook interface is typically the driver program. This driver program contains the main loop for the program and creates distributed datasets on the cluster, then applies operations to those datasets. Driver programs access Apache Spark through a *SparkSession* object regardless of deployment location.



Microsoft Azure manages the cluster, and auto-scales it as needed based on your usage and the setting used when configuring the cluster. Auto-termination can also be enabled, which allows Azure to terminate the cluster after a specified number of minutes of inactivity.

Spark jobs in detail

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 reorganized and shared over the network.

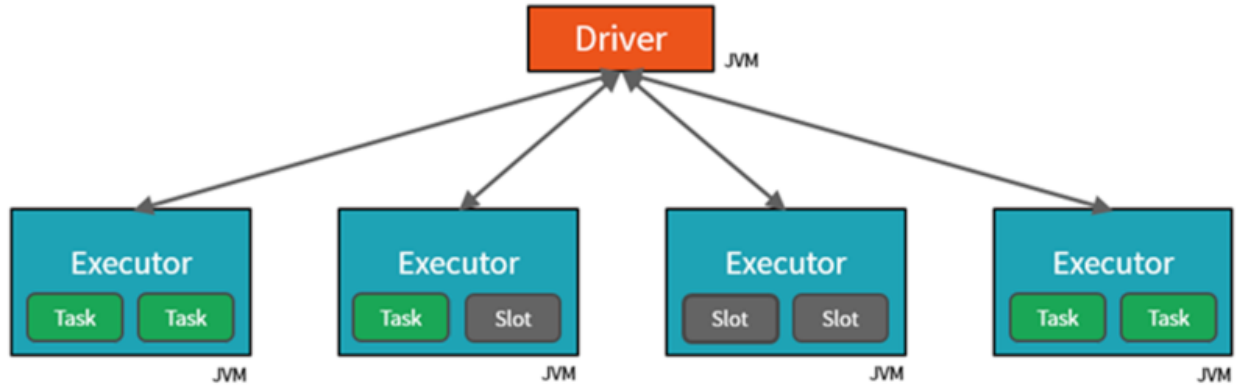
The secret to Spark's high performance is parallelism. Scaling *vertically* (by adding resources to a single computer) is limited to a finite amount of RAM, Threads and CPU speeds; but clusters scale *horizontally*, adding new nodes to the cluster as needed.

Spark parallelizes jobs at two levels:

• The first level of parallelization is the *executor* - a Java virtual machine (JVM) running on a worker 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 multiple slots to which parallelized tasks can be assigned.



The JVM is naturally multi-threaded, but a single JVM, such as the one coordinating the work on the driver, has a finite upper limit. By splitting the work into tasks, the driver can assign units of work to *slots in the executors on worker nodes for parallel execution. Additionally, the driver determines how to partition the data so that it can be distributed for parallel processing. So, the driver assigns a partition of data to each task so that each task knows which piece of data it is to process. Once started, each task will fetch the partition of data assigned to it.

Jobs and stages

Depending on the work being performed, multiple parallelized jobs may be required. Each job is broken down into *stages*. A useful analogy is to imagine that the job is to build a house:

• 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 roof.

Attempting to do any of these steps out of order just doesn't make sense and may in fact be impossible. Similarly, Spark breaks each job into stages to ensure everything is done in the right order.

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

Question 61: **Incorrect**
Scenario: You are configuring a new Azure Storage Account.

By default, what is the network rule set to?

- **None of the listed options.**
- **To allow all connection from a private IP address range.**
-

To allow all connections from all networks.

(Correct)

-

To deny all connections from all networks.

(Incorrect)

Explanation
The default network rule is to allow all connections from all networks.

To secure your storage account, you should first configure a rule to deny access to traffic from all networks (including internet traffic) on the public endpoint, by default. Then, you should configure rules that grant access to traffic from specific VNets. You can also configure rules to grant access to traffic from selected public internet IP address ranges, enabling connections from specific internet or on-premises clients. This configuration enables you to build a secure network boundary for your applications.

You can combine firewall rules that allow access from specific virtual networks and from public IP address ranges on the same storage account. Storage firewall rules can be applied to existing storage accounts, or when creating new storage accounts.

Storage firewall rules apply to the public endpoint of a storage account. You don't need any firewall access rules to allow traffic for private endpoints of a storage account. The process of approving the creation of a private endpoint grants implicit access to traffic from the subnet that hosts the private endpoint.

Network rules are enforced on all network protocols for Azure storage, including REST and SMB. To access data using tools such as the Azure portal, Storage Explorer, and AZCopy, explicit network rules must be configured.

Once network rules are applied, they're enforced for all requests. SAS tokens that grant access to a specific IP address serve to limit the access of the token holder, but don't grant new access beyond configured network rules.

Virtual machine disk traffic (including mount and unmount operations, and disk IO) is not affected by network rules. REST access to page blobs is protected by network rules.

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

Question 62: **Correct**

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. Notebook cells are individual blocks of code or text that can be ran independently or as a group.

True or False: It is possible to reference data or variables directly across different languages in a Synapse Studio notebook.

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

  (Correct)

Explanation
In order to understand the development of notebooks you need to understand that it consists of cells. Cells are individual blocks of code or text that can be ran independently or as a group.

In order to develop notebooks, there are a couple of things to take in mind

• Adding cells to notebooks This will give you the opportunity to add code in a different cell. There are multiple ways to add a new cell to your notebook.

• Setting a primary language Azure Synapse Studio notebook has support for four Apache Spark languages to be set as primary languages in a notebook.

These are pySpark (Python), Spark (Scala), SparkSQL, and .NET for Apache Spark (C#)

• Using multiple languages Within a notebook you are enabled to use multiple languages. The one thing to take in mind is that you need to use magic commands, at the beginning of a cell.

• Using temp tables to reference data across languages. **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.

• IDE-Style IntelliSense When you use Azure Synapse Studio notebooks, you'll see the integration with the Monaco editor. It enables you to bring IDE-style IntelliSense to the cell editor. This helps you in cases of Syntax highlight, error marker, and automatic code completions to help you to write code and identify issues quicker. You do have to take

in mind that The IntelliSense features are sometimes at different levels of maturity for different languages. So depending on the language you want to write your code in, in the Azure Synapse Studio Notebooks environment you could check the following table that explains what is supported using the types of language at your convenience.

Languages	Syntax Highlight	Syntax Error Marker	Syntax Code Completion	Variable Code Completion	System Function Code Completion	User Function Code Completion	Smart Indent	Code Folding
PySpark (Python)	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Spark (Scala)	Yes	Yes	Yes	Yes	-	-	-	Yes
SparkSQL	Yes	Yes	-	-	-	-	-	-
.NET for Spark (C#)	Yes	-	-	-	-	-	-	-

• Undo Cell operations. When you, for example, need to revoke a cell operation, you can do so within the Azure Synapse Studio notebook environment.

• Move a cell. If you want to align different cells of code and put them in a correct order, the notebook environment gives you the opportunity to do so.

• Delete a cell. If you have written a cell of code, but no longer need it or it needs to be deleted, the functionality can be used within the notebook environment.

• Collapse Cell in and output If you want to collapse a cell to check in or output, the functionality is available to you when using the notebook environment in Synapse Studio.

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

Question 63: **Incorrect**
What are the two prerequisites for connecting Azure Databricks with Azure Synapse Analytics that apply to the Azure Synapse Analytics instance?

- **Add the client IP address to the firewall's allowed IP addresses list and use the correctly formatted `ConnectionString`**

-

- **Create a database master key and configure the firewall to enable Azure services to connect**

  (Correct)

- **Use a correctly formatted `ConnectionString` and create a database master key**

  (Incorrect)

- **Generate a OTP to verify the account credentials, then set a master endpoint then configure the endpoint firewall to enable Azure services to connect.**

Explanation

Azure Databricks is an Apache Spark–based analytics platform that supports SQL analytics and can be integrated with Azure Synapse to run high-performance analytics. It allows faster interactive processing of batch and streaming data and has built-in functions for machine learning and big data processing.

The two prerequisites for connecting Azure Databricks with Azure Synapse Analytics that apply to the Azure Synapse Analytics instance are to create a database master key and configure the firewall to enable Azure services to connect.

https://docs.databricks.com/data/data-sources/azure/synapse-analytics.html

Question 64: **Incorrect**

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

[A] data is typically tabular data that is represented by rows and columns in a database. Databases that hold tables in this form are called [B] databases.

- **[A] Unstructured, [B] Binary**

- **[A] Relational, [B] Structured**

-

[A] Structured, [B] Relational

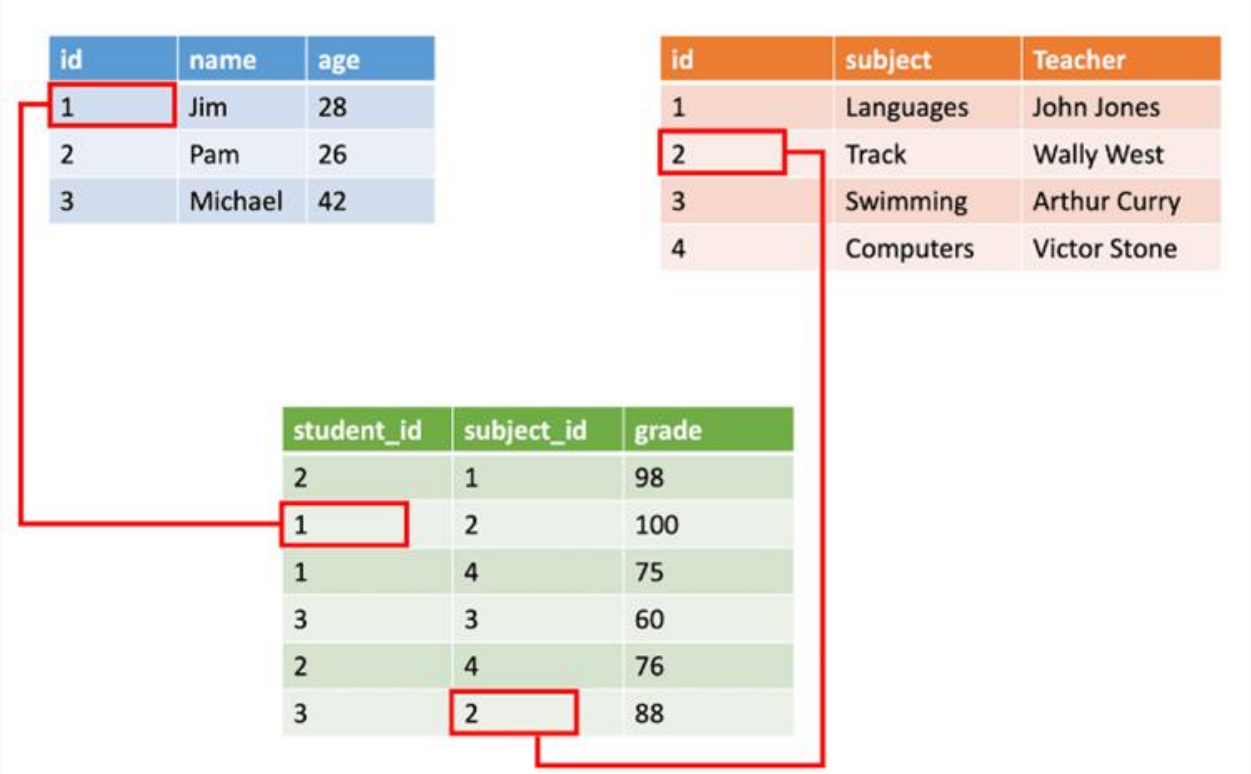
(Correct)

-

[A] JSON, [B] Semi-Structured

(Incorrect)

Explanation
Structured data is typically tabular data that is represented by rows and columns in a database. Databases that hold tables in this form are called *relational databases* (the mathematical term *relation* refers to an organized set of data held as a table).



https://f5a395285c.nxcli.net/microsoft-azure/dp-900/structured-data-vs-unstructured-data-vs-semi-structured-data

Question 65: **Correct**
How do you infer the data types and column names when you read a JSON file?

-

- `spark.read.inferSchema("true").json(jsonFile)`

- `spark.read.option("inferData", "true").json(jsonFile)`

- `spark.read.option.inferSchema("true").json(jsonFile)`

- `spark.read.option("inferSchema", "true").json(jsonFile)`

  (Correct)

Explanation
The `spark.read.option("inferSchema", "true").json(jsonFile)` approach is the correct way to infer the file's schema.

https://bartoszgajda.com/2020/06/26/exploiting-schema-inference-in-apache-spark/

Question 66: **Incorrect**
Which type of transactional database system would work best for product data?

- **OLAP**

- **ELT**

- **ADPS**

- **OLTP**

  (Correct)

-

  ETL

  (Incorrect)

Explanation
OLTP systems support a large set of users, have quick response times, handle large volumes of data, are highly available, and are great for small or relatively simple transactions.

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 67: **Incorrect**
What's the purpose of linked services in Azure Data Factory?

-

  To represent a data store or a compute resource that can host execution of an activity.

  (Correct)

-

  To represent a processing step in a pipeline.

- **To link data storage devices between on-prem and cloud environments.**

- **To link data stores or computer resources together for the movement of data between resources.**

  (Incorrect)

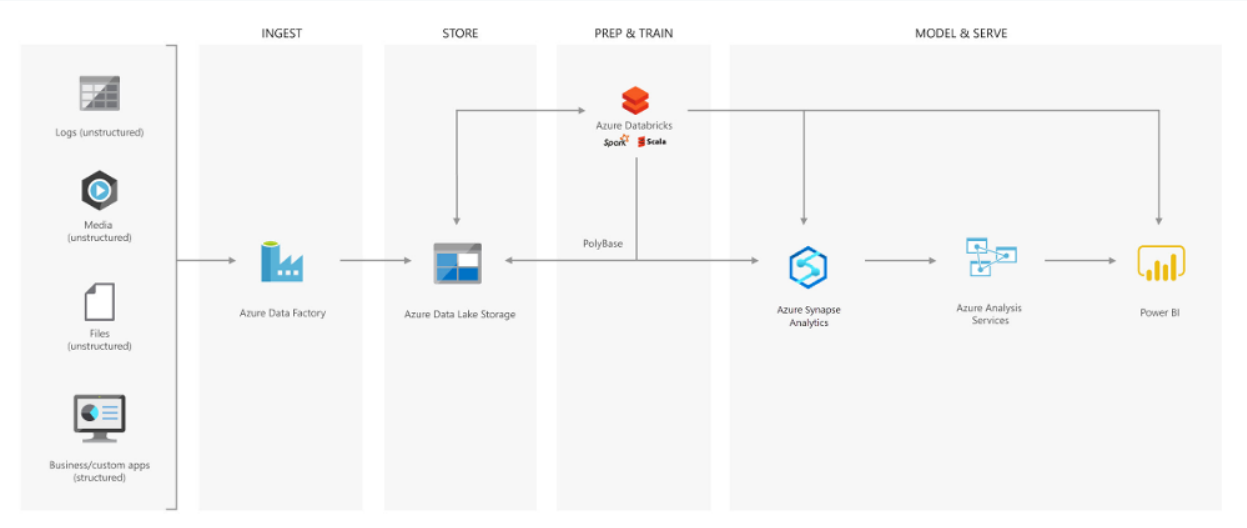
Explanation
Linked services define the connection information needed for Data Factory to connect to external resources.

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

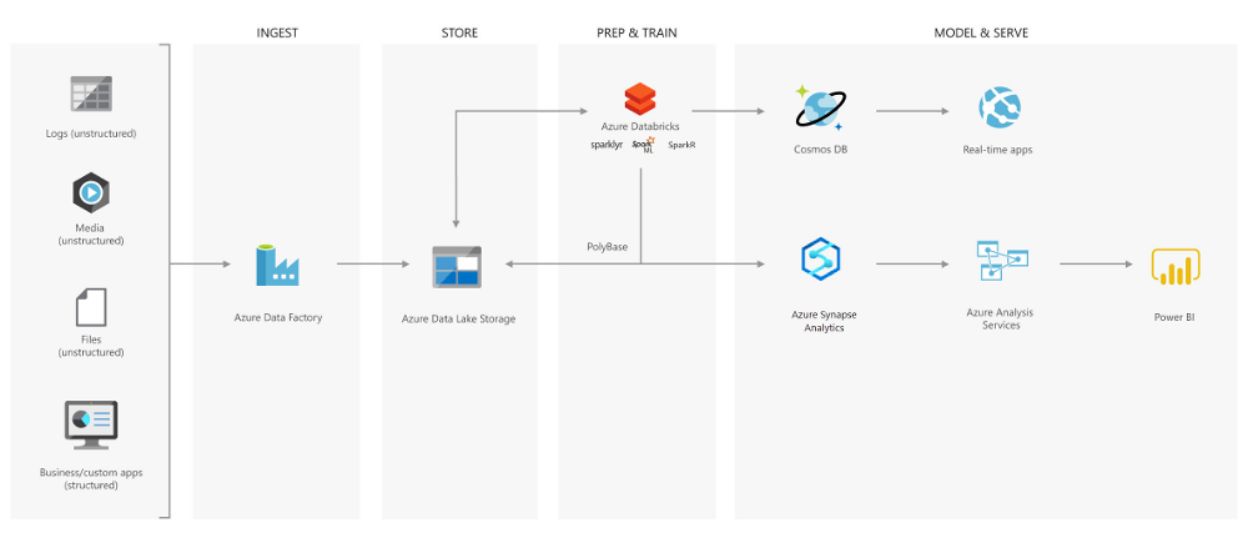
Question 68: **Incorrect**
Scenario: You are working at LexCorp which is a household appliance manufacturer that wishes to implement predictive maintenance analytics for its appliances while onsite in its customers residences.

Review the following architecture designs.

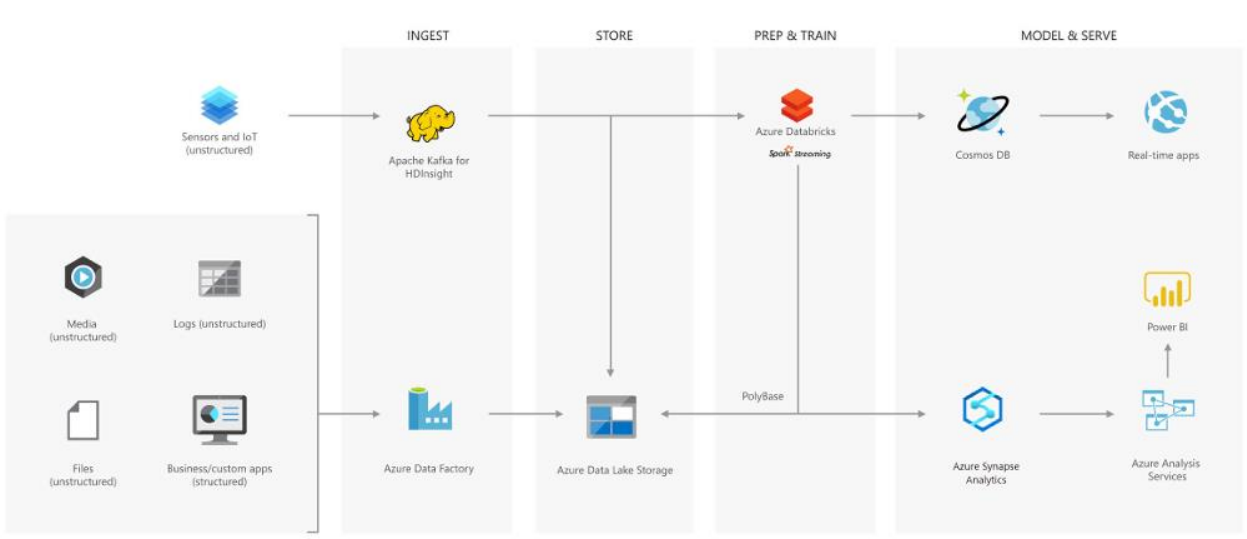
Design A:



Design B:



Design C:



Which architecture would be best suited for the need?

-

  Design B

-

Design C

(Correct)

-

Design A

(Incorrect)

-

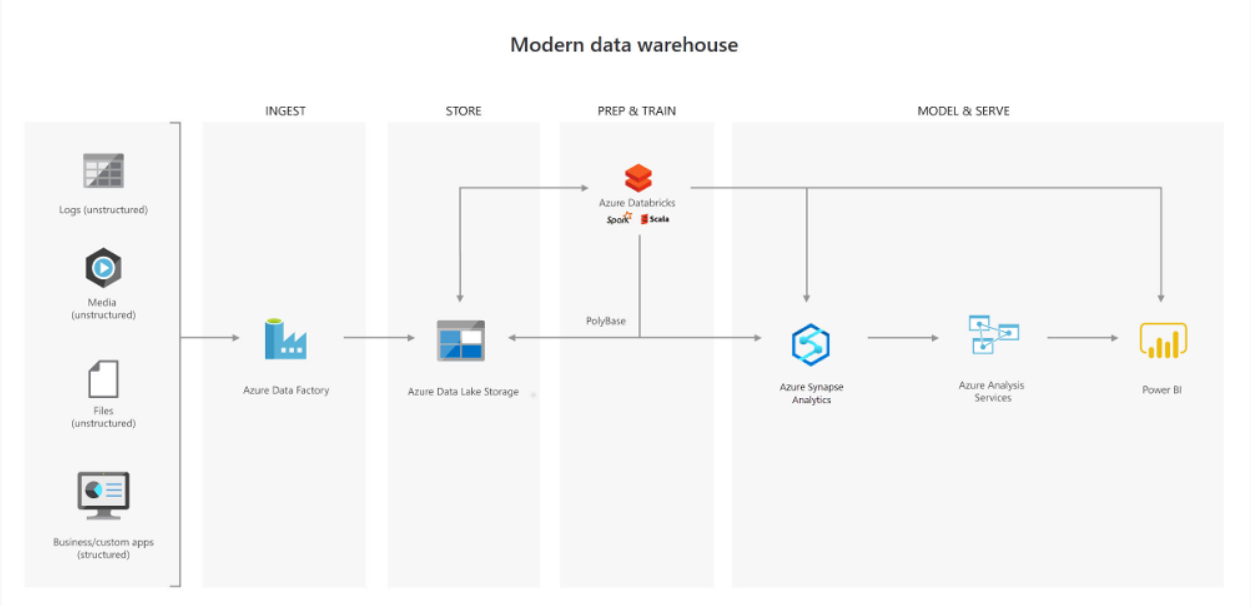
None of the listed options

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.

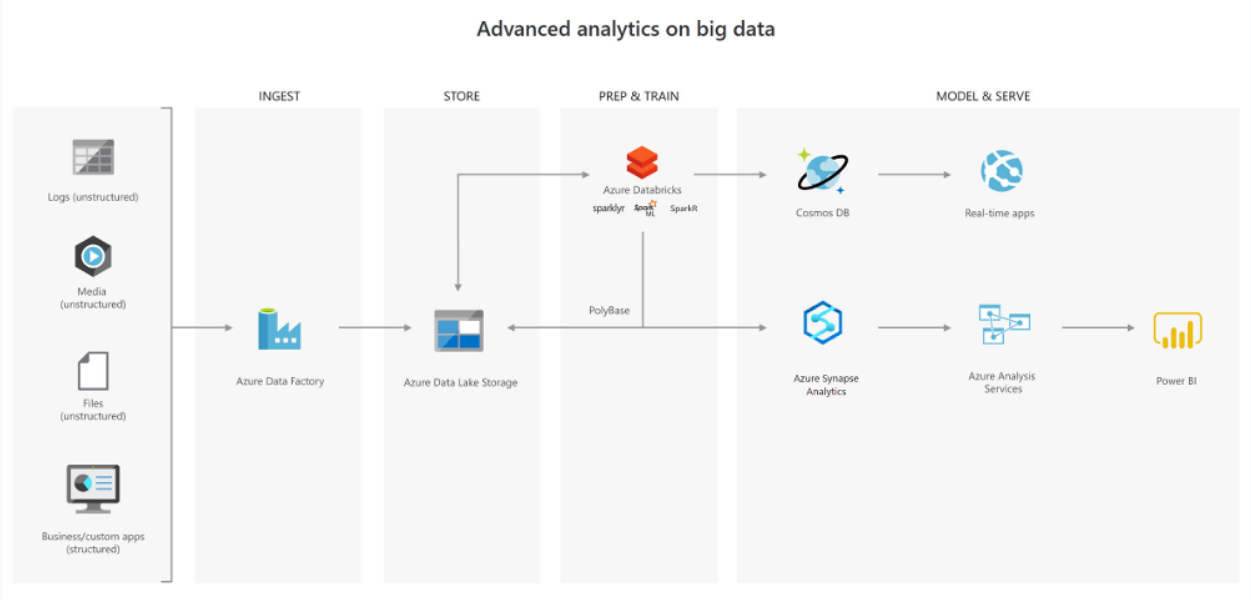
The company has turned to you for help with creating an architecture that can scale with the data needs that are required to create a predictive model and to handle the JSON data so that it's integrated into the BI solution. You suggest the following architecture:



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:

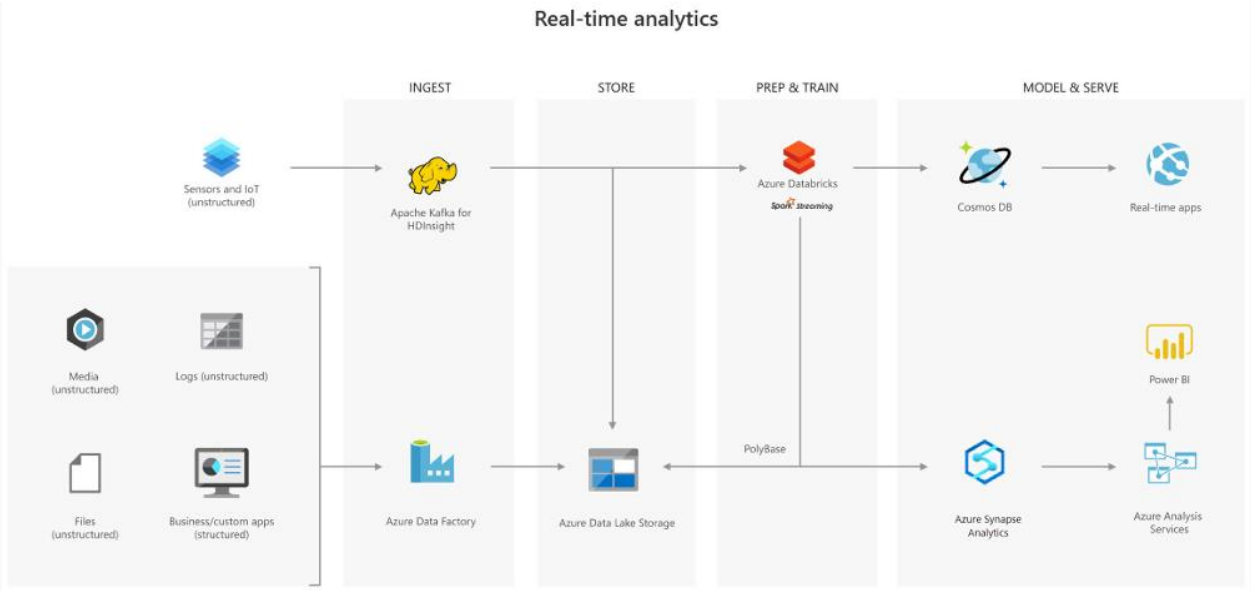


Real-time analytical solutions

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 69: **Incorrect**

Scenario: Mys-Tech is a British multinational chemical corporation with worldwide interests. It is headquartered in London, UK. Mys-Tech employs thousands of people has numerous facilities all over the world.

You are an Azure developer in Mys-Tech and have been tasked with enabling analytical queries over Azure Synapse Link from Azure Synapse Analytics.

Given: There is an existing container in a Cosmos DB core (SQL) database.

Which of the following must you do to meet the requirement?

- **Enable Azure Synapse Link in the container to create an analytical store**

  (Correct)

- **Delete and recreate the container**

- **None of the provided options are correct**

- **Add an item to the container**

  (Incorrect)

Explanation

- *Before a container can be used for analytical queries, you need to enable Synapse link for the container; which creates an analytical store.*
- You do not need to delete and recreate an existing container to enable Synapse Link integration.
- Adding an item to a container does not enable Synapse Link integration.

Query Cosmos DB with Synapse SQL

In addition to using a Spark pool, you can also query an Azure Cosmos DB analytical container by using a built-in serverless SQL pool in Azure Synapse Analytics. To do this, you can use the OPENROWSET SQL function to connect to the linked service for your Azure Cosmos DB database.

Using OPENROWSET with an authentication key

By default, access to an Azure Cosmos DB account is authenticated by an authentication key. You can use this key as part of a connection string in an OPENROWSET statement to connect through a linked service from a SQL pool, as shown in the following example:

```
SQL
SELECT *
FROM OPENROWSET(
    'CosmosDB',
    'Account=my-cosmos-db;Database=my-db;Key=abcd1234....==',
    [my-container]) AS products_data
```

Tip: You can find a primary and secondary key for your Cosmos DB account on its Keys page in the Azure portal.

The results of this query might look something like the following, including metadata and application-defined fields from the items in the Azure Cosmos DB container:

_rid	_ts	productID	productName	id	_etag
mjMaAL...==	1655414791	123	Widget	7248f072-11c3-42b1-a368-...	54004b09-0000-2300-...
mjMaAL...==	1655414829	124	Wotsit	dc33131c-65c7-421a-a0f7-...	5400ca09-0000-2300-...
mjMaAL...==	1655414835	125	Thingumy	ce22351d-78c7-428a-a1h5-...	5400ca09-0000-2300-...
...	...	...	...	...	...

The data is retrieved from the analytical store, and the query doesn't impact the operational store.

Using OPENROWSET with a credential

Instead of including the authentication key in each call to OPENROWSET, you can define a *credential* that encapsulates the authentication information for your Cosmos DB

account, and use the credential in subsequent queries. To create a credential, use the `CREATE CREDENTIAL` statement as shown in this example:

```
 SQL
CREATE CREDENTIAL my_credential
 WITH IDENTITY = 'SHARED ACCESS SIGNATURE',
 SECRET = 'abcd1234....==';
```

With the credential in place, you can use it in an `OPENROWSET` function like this:

```
SQL
SELECT *
FROM OPENROWSET(PROVIDER = 'CosmosDB',
                CONNECTION = 'Account=my-cosmos-db;Database=my-db',
                OBJECT = 'my-container',
                SERVER_CREDENTIAL = 'my_credential'
) AS products_data
```

Once again, the results include metadata and application-defined fields from the analytical store:

_rid	_ts	productID	productName	id	_etag
mjMaAL...==	1655414791	123	Widget	7248f072-11c3-42b1-a368-...	54004b09-0000-2300-...
mjMaAL...==	1655414829	124	Wotsit	dc33131c-65c7-421a-a0f7-...	5400ca09-0000-2300-...
mjMaAL...==	1655414835	125	Thingumy	ce22351d-78c7-428a-a1h5-...	5400ca09-0000-2300-...
...	...	...	...	...	...

Specifying a schema

The `OPENROWSET` syntax includes a `WITH` clause that you can use to define a schema for the resulting rowset. You can use this to specify individual fields and assign data types as shown in the following example:

```
SQL
 SELECT *
 FROM OPENROWSET(PROVIDER = 'CosmosDB',
                 CONNECTION = 'Account=my-cosmos-db;Database=my-db',
                 OBJECT = 'my-container',
                 SERVER_CREDENTIAL = 'my_credential'
 )
 WITH (
    productID INT,
    productName VARCHAR(20)
 ) AS products_data
```

In this case, assuming the fields in the analytical store include productID and productName, the resulting rowset will resemble the following table:

productID	productName
123	Widget
124	Wotsit
125	Thingumy
...	...

You can of course specify individual column names in the `SELECT` clause (for example, `SELECT productID, productName ...`), so this ability to specify individual columns may seem of limited use. However, consider cases where the source JSON documents stored in the operational store include multiple levels of fields, as show in the following example:

```
JSON
{
```

```
    "productID": 126,
    "productName": "Sprocket",
    "supplier": {
        "supplierName": "Contoso",
        "supplierPhone": "555-123-4567"
    }
    "id": "62588f072-11c3-42b1-a738-...",
    "_rid": "mjMaAL...==",
    ...
}
```

The `WITH` clause supports the inclusion of explicit JSON paths, enabling you to handle nested fields and to assign aliases to field names; as shown in this example:

```
SQL
SELECT *
FROM OPENROWSET(PROVIDER = 'CosmosDB',
                CONNECTION = 'Account=my-cosmos-db;Database=my-db',
                OBJECT = 'my-container',
                SERVER_CREDENTIAL = 'my_credential'
 )
WITH (
    ProductNo INT '$.productID',
    ProductName VARCHAR(20) '$.productName',
    Supplier VARCHAR(20) '$.supplier.supplierName',
    SupplierPhoneNo VARCHAR(15) '$.supplier.supplierPhone'
) AS products_data
```

The results of this query would include the following row for product 126:

ProductNo	ProductName	Supplier	SupplierPhoneNo
126	Sprocket	Contoso	555-123-4567

Creating a view in a database

If you need to query the same data frequently, or you need to use reporting and visualization tools that rely on `SELECT` statements that don't include the `OPENROWSET` function, you can use a view to abstract the data. To create a view, you

should create a new database in which to define it (user-defined views in the master database aren't supported), as shown in the following example:

```
SQL
CREATE DATABASE sales_db
   COLLATE Latin1_General_100_BIN2_UTF8;
 GO;

 USE sales_db;
 GO;

 CREATE VIEW products
 AS
 SELECT *
 FROM OPENROWSET(PROVIDER = 'CosmosDB',
                 CONNECTION = 'Account=my-cosmos-db;Database=my-db',
                 OBJECT = 'my-container',
                 SERVER_CREDENTIAL = 'my_credential'
 )
 WITH (
    ProductNo INT '$.productID',
    ProductName VARCHAR(20) '$.productName',
    Supplier VARCHAR(20) '$.supplier.supplierName',
    SupplierPhoneNo VARCHAR(15) '$.supplier.supplierPhone'
 ) AS products_data
 GO
```

Tip: When creating a database that will access data in Cosmos DB, it's best to use a UTF-8 based collation to ensure compatibility with strings in Cosmos DB.

After the view has been created, users and client applications can query it like any other SQL view or table:

```
SQL
SELECT * FROM products;
```

Considerations for Serverless SQL pools and Azure Cosmos DB

When planning to use a serverless SQL pool to query data in an Azure Cosmos DB analytical store, consider the following best practices:

- Provision your Azure Cosmos DB analytical storage and any client applications (for example Microsoft Power BI) in the same region as serverless SQL pool.
- Azure Cosmos DB containers can be replicated to multiple regions. If you have a multi-region container, you can specify a `region` parameter in the `OPENROWSET` connection string to ensure queries are sent to a specific regional replica of the container.
- When working with string columns, use the `OPENROWSET` function with the explicit WITH clause and specify an appropriate data length for the string data.

https://learn.microsoft.com/en-us/azure/synapse-analytics/sql/overview-features

Question 70: **Correct**
Scenario: You need to provision a data store that will store but not query data.

Which is the least expensive option which will meet the requirement?

- **Azure Table Storage**
- **Azure Database Server Storage**
- **Azure Queue Storage**
- **Azure Data Lake Storage**
- **Azure File Storage**
- **Azure Blob Storage**

  (Correct)

Azure Cosmos DB Storage

Explanation

When to use Blob Storage

If you need to provision a data store that will store but not query data, your cheapest option is to set up a storage account as a Blob store. Azure Blob storage is Microsoft's object storage solution for the cloud. Blob storage is optimized for storing massive amounts of unstructured data. Unstructured data is data that doesn't adhere to a particular data model or definition, such as text or binary data. Blob storage works well with images and unstructured data, and it's the cheapest way to store data in Azure.

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

Question 71: **Correct**

True or False: Unique credential creation is always required when utilizing the Azure Synapse Apache Spark Pool to Synapse SQL connector to enforce access control.

- True
- False

(Correct)

Explanation

The Authentication between the Azure Synapse Apache Spark Pool to Synapse SQL systems is made seamless due to the Azure Synapse Apache Spark Pool to Synapse SQL connector used in Azure Synapse Analytics. **The Token Service connects with Azure Active Directory to obtain security tokens for use when accessing the storage account or the data warehouse server.**

For this reason, there's no need to create credentials or specify them in the connector API as long as Azure AD-Auth is configured at the storage account and the data warehouse server. If not, SQL Auth can be specified. The only constraint that needs to be taken into account is that this connector is only working in scala.

There are some Prerequisites in order to authenticate namely:

• It needs to be a member of db_exporter role in the database or SQL pool from which you to transfer data to or from.

• It needs to be a member of the Storage Blob Data Contributor role on the default storage account.

If you want to create users, you need to connect to the SQL Pool database from which you want transfer data to/from.

Import statements are not needed since they are pre-loaded in case you use the notebook experience.

Once the authentication is in place, you are enabled to transfer data to or from a dedicated SQL pool attached within the workspace.

https://docs.microsoft.com/en-us/azure/synapse-analytics/spark/apache-spark-secure-credentials-with-tokenlibrary?pivots=programming-language-csharp

Question 72: **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 used together 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 Activity type?

- ○

  Activity type: Lookup activity

- ○

  Activity type: Event-based activity

- ○

  Activity type: Stored procedure activity

- ◉

  Activity type: Copy activity

  (Correct)

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.

Copy activity in Azure Data Factory

In Azure Data Factory, you can use the Copy activity to copy data among data stores located on-premises and in the cloud. After you copy the data, you can use other activities to further transform and analyze it. You can also use the Copy activity to publish transformation and analysis results for business intelligence (BI) and application consumption.

The Copy activity is executed on an integration runtime. You can use different types of integration runtimes for different data copy scenarios:

When you're copying data between two data stores that are publicly accessible through the internet from any IP, you can use the Azure integration runtime for the copy activity. This integration runtime is secure, reliable, scalable, and globally available.

When you're copying data to and from data stores that are located on-premises or in a network with access control (for example, an Azure virtual network), you need to set up a self-hosted integration runtime.

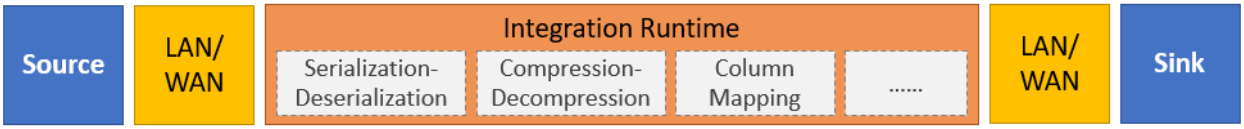
An integration runtime needs to be associated with each source and sink data store. For information about how the Copy activity determines which integration runtime to use, see Determining which IR to use.

To copy data from a source to a sink, the service that runs the Copy activity performs these steps:

Reads data from a source data store.

Performs serialization/deserialization, compression/decompression, column mapping, and so on. It performs these operations based on the configuration of the input dataset, output dataset, and Copy activity.

Writes data to the sink/destination data store.



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

Question 73: **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 which API to use for the database model and type based on the following information.

Specifications:

• The application uses a NoSQL database to store data

• The database uses the key-value and wide-column NoSQL database type.

Required: Developers need to access data in the database using an API.

Which of the following APIs should you recommend to Bruce and his team?

- ☐

  Table API

  (Correct)

- ☐

  Cassandra API

  (Correct)

- ☑

  SQL API

  (Incorrect)

- ☐

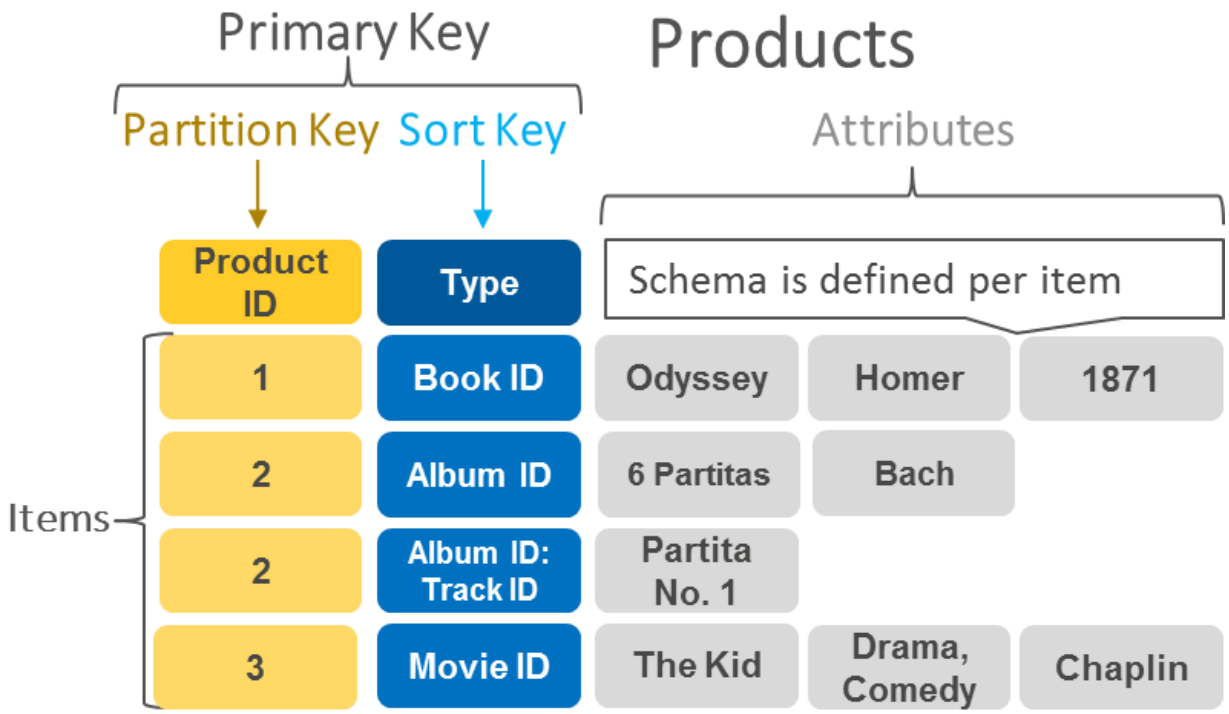
  Gremlin API

- ☐

  MongoDB API

Explanation

A **key-value database** is a type of nonrelational database that uses a simple key-value method to store data. A key-value database stores data as a collection of key-value pairs in which a key serves as a unique identifier. Both keys and values can be anything, ranging from simple objects to complex compound objects. Key-value databases are highly partitionable and allow horizontal scaling at scales that other types of databases cannot achieve.



https://aws.amazon.com/nosql/key-value/

Azure Cosmos DB provides the Table API for applications that are written for Azure Table storage and that need premium capabilities like:

• Turnkey global distribution.

• Dedicated throughput worldwide (when using provisioned throughput).

• Single-digit millisecond latencies at the 99th percentile.

• Guaranteed high availability.

• Automatic secondary indexing.

Applications written for Azure Table storage can migrate to Azure Cosmos DB by using the Table API with no code changes and take advantage of premium capabilities. The Table API has client SDKs available for .NET, Java, Python, and Node.js.

∴ The database uses the key-value → Table API

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

Wide Column Databases, or Column Family Databases, refers to a category of NoSQL databases that works well for storing enormous amounts of data that can be collected. Its architecture uses persistent, sparse matrix, multi-dimensional mapping (row-value, column-value, and timestamp) in a tabular format meant for massive scalability (over and above the petabyte scale). Column Family stores do not follow the relational model, and they aren't optimized for joins.

Good Wide Column Database use cases include:

• Sensor Logs [Internet of Things (IOT)]

• User preferences

• Geographic information

• Reporting systems

• Time Series Data

• Logging and other write heavy applications

Wide Column Databases are not the preferred choice for applications with ad-hoc query patterns, high level aggregations and changing database requirements. This type of data store does not keep good data lineage.

https://www.dataversity.net/wide-column-database/

Azure Cosmos DB Cassandra API can be used as the data store for apps written for Apache Cassandra. This means that by using existing Apache drivers compliant with CQLv4, your existing Cassandra application can now communicate with the Azure Cosmos DB Cassandra API. In many cases, you can switch from using Apache Cassandra to using Azure Cosmos DB's Cassandra API, by just changing a connection string.

The Cassandra API enables you to interact with data stored in Azure Cosmos DB using the Cassandra Query Language (CQL) , Cassandra-based tools (like cqlsh) and Cassandra client drivers that you're already familiar with.

Wide-column stores store data together as columns instead of rows and are optimized for queries over large datasets. The most popular are Cassandra and HBase.

∴ The database uses wide-column NoSQL database type → Cassandra API

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

Summary

Key-value databases → Table API

Columnar databases → Cassandra API

Graph databases → Gremlin API

Document databases → SQL API MongoDB API

Question 74: **Incorrect**
Scenario: While working on a project using Azure Data Factory, you are planning to load data into a data store or compute resource.

Which transformation in Mapping Data Flow is used to do this?

- ○

Window

- ○

Cache

- ○

Field mapping

- ○

Sink

(Correct)

- ◉

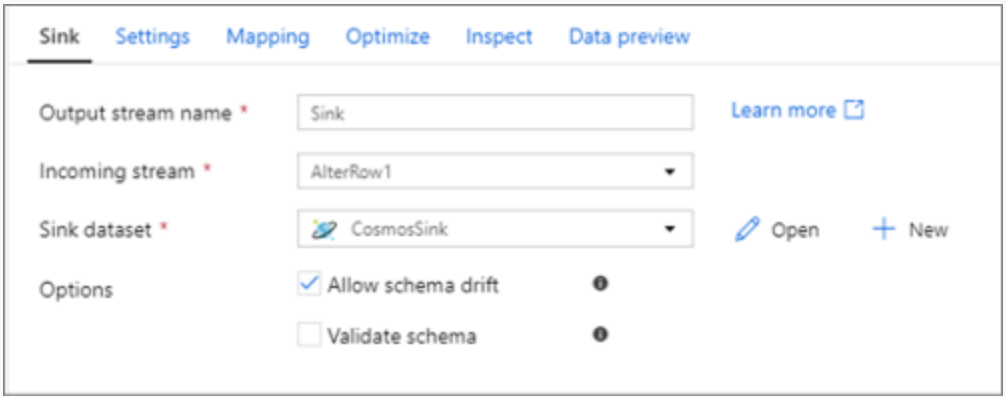
Source

(Incorrect)

Explanation

After you finish transforming your data, write it into a destination store by using the sink transformation. Every data flow requires at least one sink transformation, but you can write to as many sinks as necessary to complete your transformation flow. To write to additional sinks, create new streams via new branches and conditional splits.

Each sink transformation is associated with exactly one Azure Data Factory dataset object or linked service. The sink transformation determines the shape and location of the data you want to write to.



A Sink transformation allows you to choose a dataset definition for the destination output data. You can have as many sink transformations as your data flow requires.

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

Question 75: **Incorrect**

Azure Synapse Studio is the primary tool to use to interact with the many components that exist in the service. It organizes itself into hubs which allow you to perform a wide range of activities against your data.

Which of the following are the referenced hubs on Azure Synapse Studio? (Select six)

- ☐

  Manage

  (Correct)

- ☐

  Develop

  (Correct)

- ☐

  Data

  (Correct)

- ☐

  Monitor

  (Correct)

- ☑

  Ingest

  (Incorrect)

- ☐

  Import

- ☐

  Connect BI

- ☐

  Home

  (Correct)

- ☐

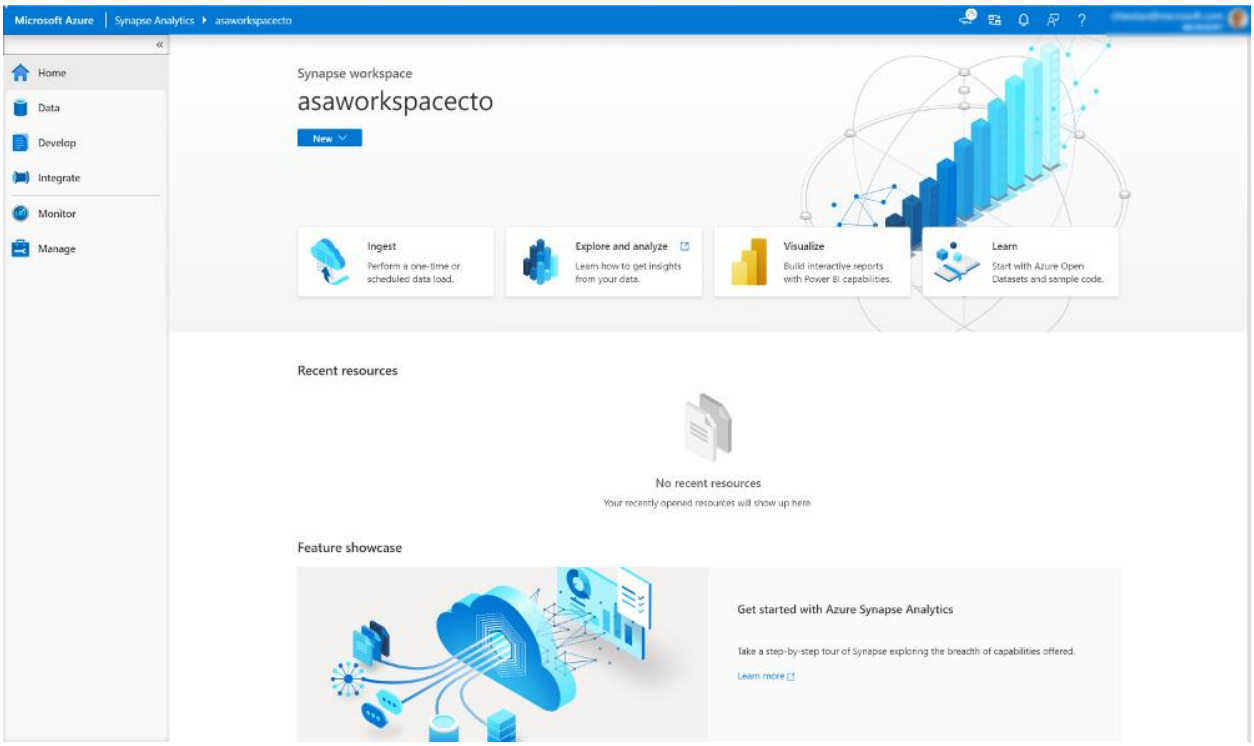
  Explore and analyze

- ☐

  Integrate

  (Correct)

Explanation
Azure Synapse Studio is the primary tool to use to interact with the many components that exist in the service. It organizes itself into hubs, as seen on the left-hand side of the Azure Synapse Studio UI, which allow you to perform a wide range of activities against your data.



The following hubs are available within Azure Synapse Studio.

Home

The home hub contains short cuts that enable you to ingest, explore, analyze, and visualize your data. These provide shortcuts to tools such as the Copy Data Tool for ingesting data, to connecting to a Power BI workspace for visualization. You will also find links to resources that such as the documentation and pricing page. It will also list any resources you recently accessed, or pinned as favourite.

Data

The data hub can be accessed by either clicking on the Explore link in the home hub, or by selecting data on the left of the application. In this hub, you can access your provisioned SQL pool databases and SQL serverless databases in your workspace, as well as external data sources, such as storage accounts and other linked services. You also can preview data tables and data files.

Develop

The Develop hub is where you manage SQL scripts, Synapse notebooks, data flows, and Power BI reports. It can also be accessed by clicking on the Analyze icon in the home page.

Integrate

Manage data integration pipelines within the Integrate hub. If you are familiar with Azure Data Factory, then you will feel at home in this hub. The pipeline creation experience is the same as in Azure Data Factory, which gives you another powerful integration built into Synapse Analytics, removing the need to use Azure Data Factory separately for data movement and transformation pipelines.

Monitor

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. If you want to see the status of a job or activity, this is where you want to go.

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.

Manage

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

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

Question 76: **Incorrect**
True or False: By default, Azure Storage accounts automatically encrypt data-at-rest and data-in-transit. This will protect data-in-transit regardless if an authorized connection uses HTTP or HTTPS.

- ○

  False

  (Correct)

- ◉

  True

  (Incorrect)

Explanation
Encryption in transit

Keep your data secure by enabling *transport-level security* between Azure and the client. Always use `HTTPS` to secure communication over the public internet. When you call the REST APIs to access objects in storage accounts, you can enforce the use of `HTTPS` by requiring secure transfer for the storage account. After you enable secure transfer, connections that use `HTTP` will be refused. This flag will also enforce secure transfer over SMB by requiring SMB 3.0 for all file share mounts.

https://docs.microsoft.com/en-us/azure/security/fundamentals/encryption-overview

Question 77: **Incorrect**
Which is the default distribution used for a table in Synapse Analytics?

- **Round-Robin distribution**

  (Correct)

- **Non-clustered distribution**

- **HASH distribution**

- **Replicated Table distribution**

- **Clustered distribution**

- **B-tree distribution**

  (Incorrect)

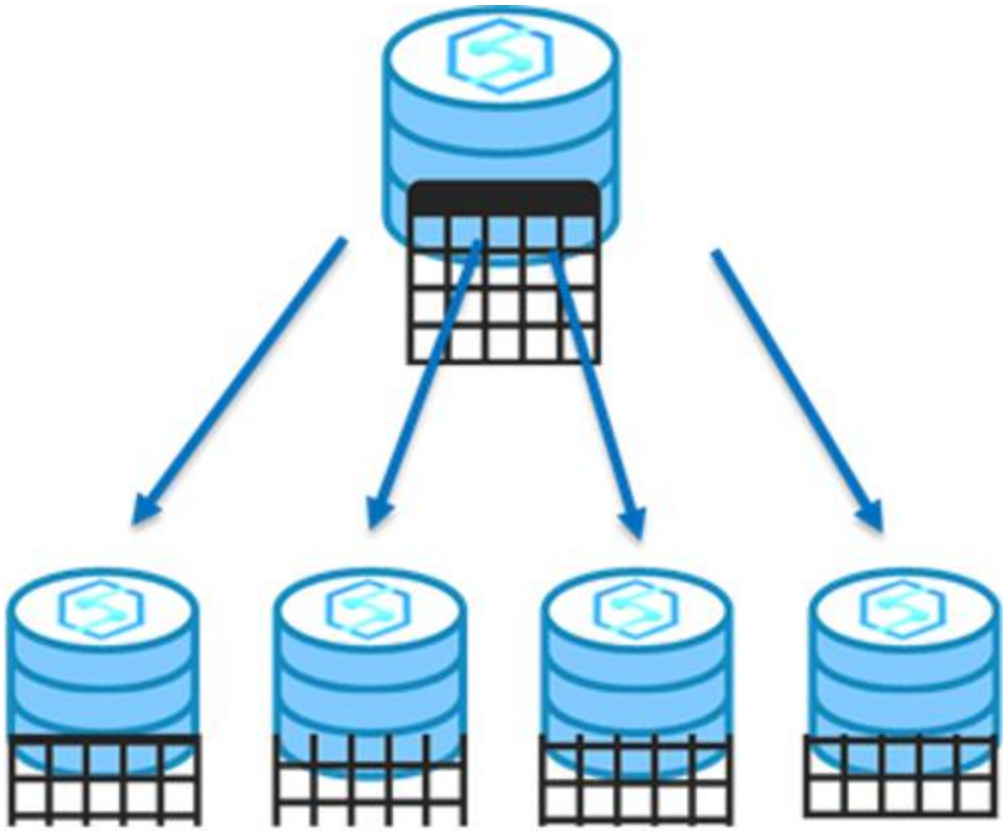
Explanation

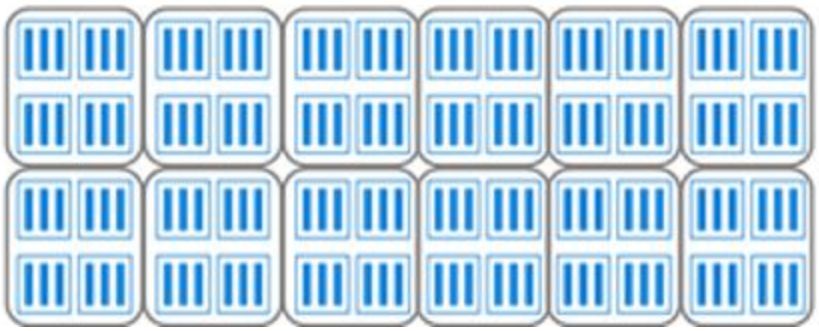
Round-Robin is the default distribution created for a table and delivers fast performance when used for loading data but may negatively impact larger queries.

There are three main table distributions available in Synapse Analytics SQL Pools.

Selecting the correct table distribution can have an impact on the data load and query performance as follows:

Round robin distribution





This is the default distribution created for a table and delivers fast performance when used for loading data.

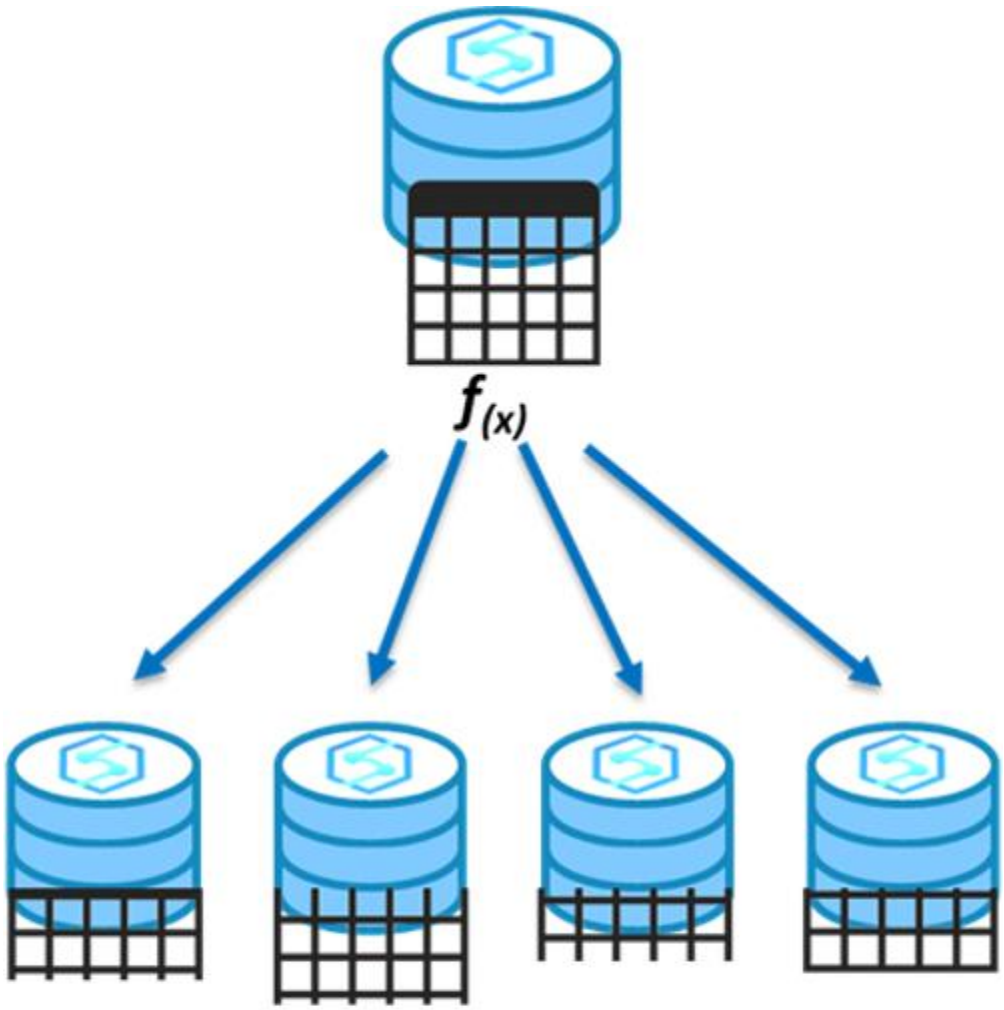
A round-robin distributed table distributes data evenly across the table but without any further optimization. A distribution is first chosen at random and then buffers of rows are assigned to distributions sequentially.

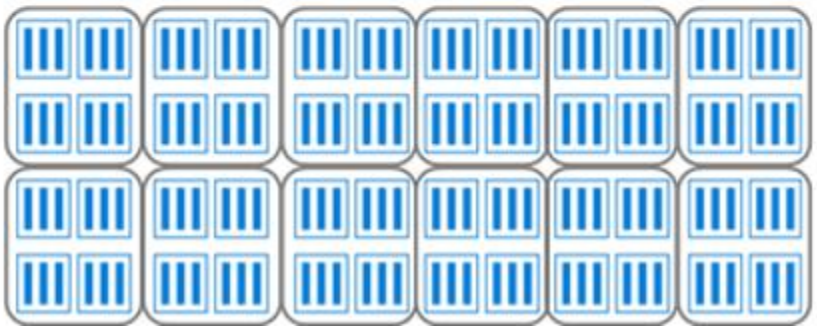
It is quick to load data into a round-robin table, but query performance can often be better with hash distributed tables for larger datasets.

Joins on round-robin tables may negatively affect query workloads, as data that is gathered for processing then has to be reshuffled to other compute nodes, which take additional time and processing.

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

Hash distribution





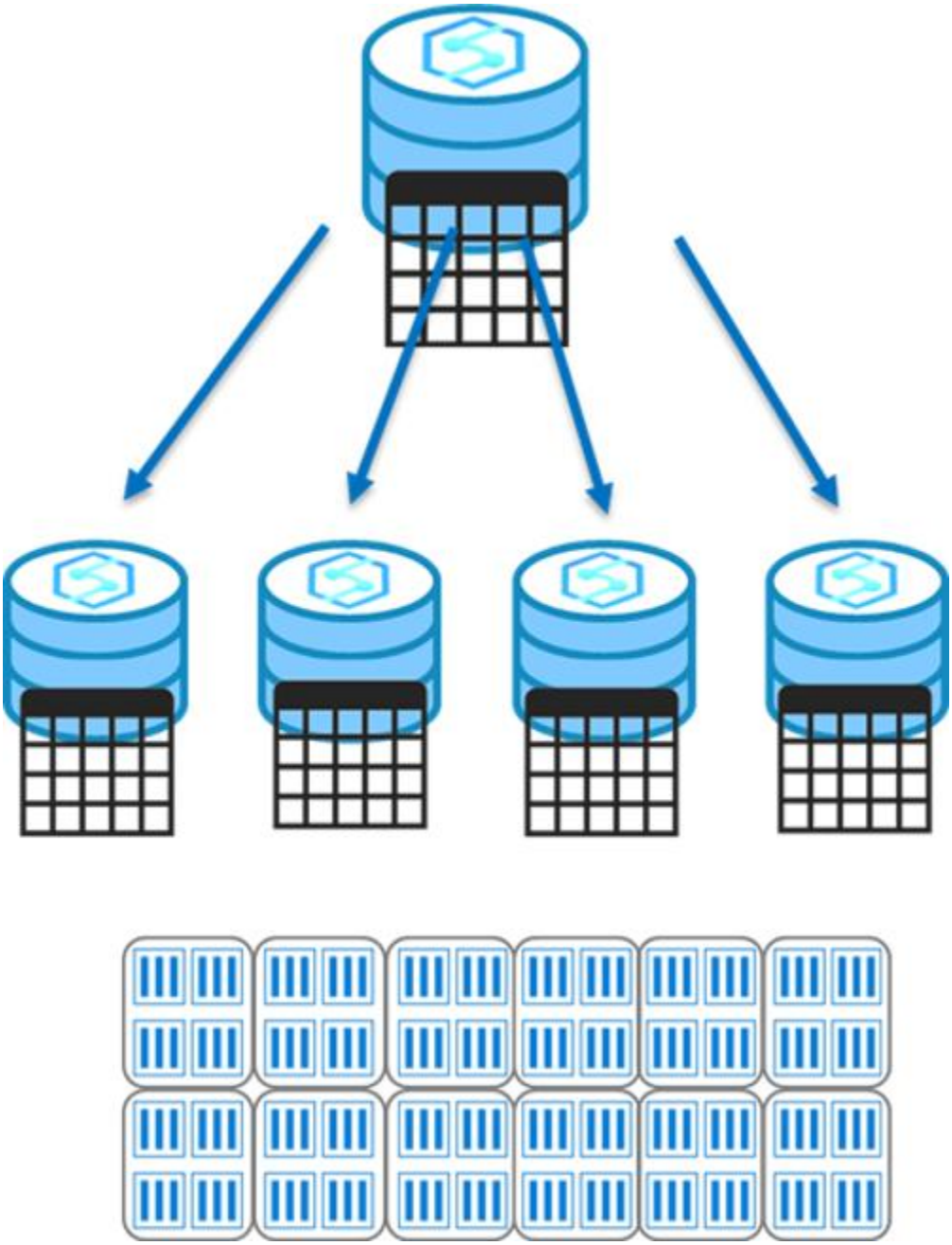
This distribution can deliver the highest query performance for joins and aggregations on large tables.

To shard data, a hash function is used to deterministically assign each row to a distribution. In the table definition, one of the columns is designated as the distribution column.

There are performance considerations for the selection of a distribution column, such as distinctness, data skew, and the types of queries that run on the system.

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

Replicated tables



A replicated table provides the fastest query performance for small tables.

A table that is replicated caches a full copy of the table on each compute node. Consequently, replicating a table removes the need to transfer data among compute nodes before a join or aggregation. As such extra storage is required and there is additional overhead that is incurred when writing data, which make large tables impractical.

Frequent data modifications will cause the cached copy to be invalidated, and require the table be recached.

Scaling the SQL Pool will also require the table be recached.

https://docs.microsoft.com/en-us/azure/synapse-analytics/sql-data-warehouse/design-guidance-for-replicated-tables

Question 78: **Incorrect**
The pipelines in Azure Data Factory typically perform the which of the following steps? (Select four)

- ☐

  Publish

  (Correct)

- ☐

  Connect and collect

  (Correct)

- ☐

  Treemap

- ☐

  Monitor

  (Correct)

- ☑

  Machine learning

  (Incorrect)

- ☐

  DataCopy

- ☐

  Migration

- ☐

  Transform and enrich

  (Correct)

Explanation
Data-driven workflows

The pipelines (data-driven workflows) in Azure Data Factory typically perform the following four steps:

Connect and collect

The first step in building an orchestration system is to define and connect all the required sources of data together, such as databases, file shares, and FTP web services. The next step is to ingest the data as needed to a centralized location for subsequent processing.

Transform and enrich

Compute services such as Databricks and Machine Learning can be used to prepare or produce transformed data on a maintainable and controlled schedule to feed production environments with cleansed and transformed data. In some instances, you may even augment the source data with additional data to aid analysis, or consolidate it through a normalization process to be used in a Machine Learning experiment as an example.

Publish

After the raw data has been refined into a business-ready consumable form from the transform and enrich phase, you can load the data into Azure Data Warehouse, Azure SQL Database, Azure Cosmos DB, or whichever analytics engine your business users can point to from their business intelligence tools

Monitor

Azure Data Factory has built-in support for pipeline monitoring via Azure Monitor, API, PowerShell, Azure Monitor logs, and health panels on the Azure portal, to monitor the scheduled activities and pipelines for success and failure rates.

https://mindmajix.com/azure-data-factory

Question 79: **Incorrect**
True or False: When using Azure Synapse Analytics, the preferred method for loading data is to use the service administrator account as this ensures the required permissions are available to write the content to the appropriate destinations. Lesser accounts run the risk of failure due to permission restrictions.

- **False**

  (Correct)

- **True**

  (Incorrect)

Explanation
A mistake that many people make when first exploring dedicated SQL Pools are to use the service administrator account as the one used for loading data. This account is limited to using the smallrc dynamic resource class that can use between 3% and 25% of the resources depending on the performance level of the provisioned SQL Pools.

Instead, it's better to create specific accounts assigned to different resource classes dependent on the anticipated task. This will optimize load performance and maintain concurrency as required by managing the available resource slots available within the dedicated SQL Pool.

https://docs.microsoft.com/en-us/azure/azure-resource-manager/management/move-resource-group-and-subscription

Question 80: **Incorrect**
Which of the following statements describes a wide transformations?

- **A wide transformation requires sharing data across workers. It does so by shuffling data.**

  (Correct)

- ○

  A wide transformation can be applied per partition/worker with no need to share or shuffle data to other workers.

- ○

  A wide transformation is where each input partition in the source data frame will contribute to sole output partition in the target data.

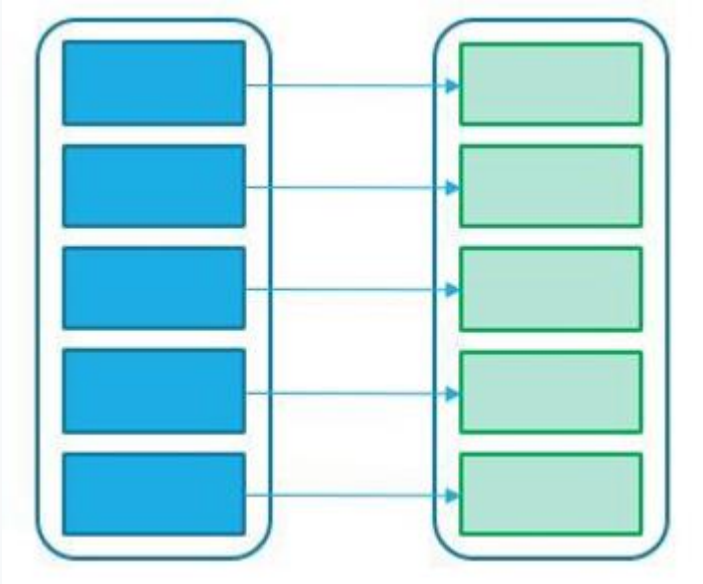
- ◉

  A wide transformation applies data transformation over a large number of columns.

  (Incorrect)

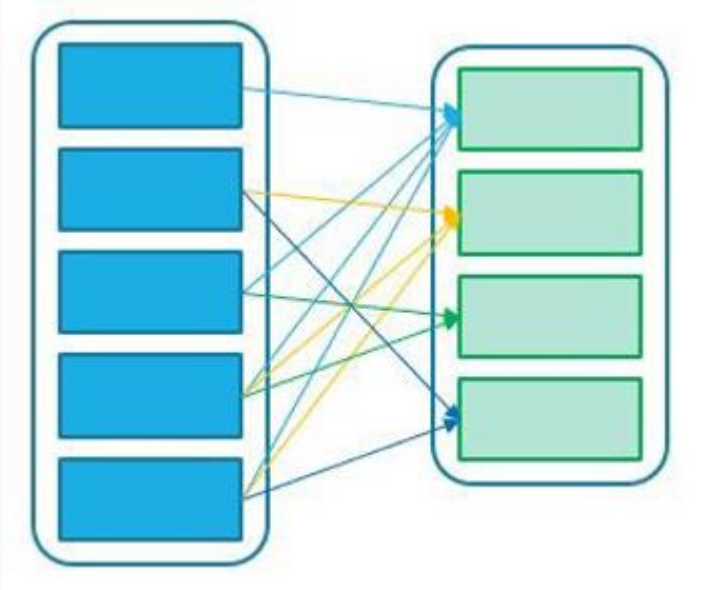
Explanation
Wide vs. Narrow Transformations

Transformations consisting of *narrow dependencies* (narrow transformations) are those where each input partition in the source data frame will contribute to only one output partition in the target data.



Spark will automatically perform an operation called pipe-lining on narrow dependencies. If multiple filters are specified on a source data frame, they'll all be optimized for example performed in-memory.

W*ide dependencies* (or wide transformations) will have input partitions contributing to many output partitions. This will result in a *shuffle* where Spark will exchange partitions between executors.



Lazy Evaluation

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

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.

https://medium.com/@lackshub/notes-for-databricks-crt020-exam-prep-9fbc97a2147e

Question 81: **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 this, you can create and schedule data-driven workflows 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.

- **Azure Synapse Link**
- **Azure Cosmos DB**
- **Azure Synapse SQL**

- **Apache Spark for Azure Synapse**

  (Incorrect)

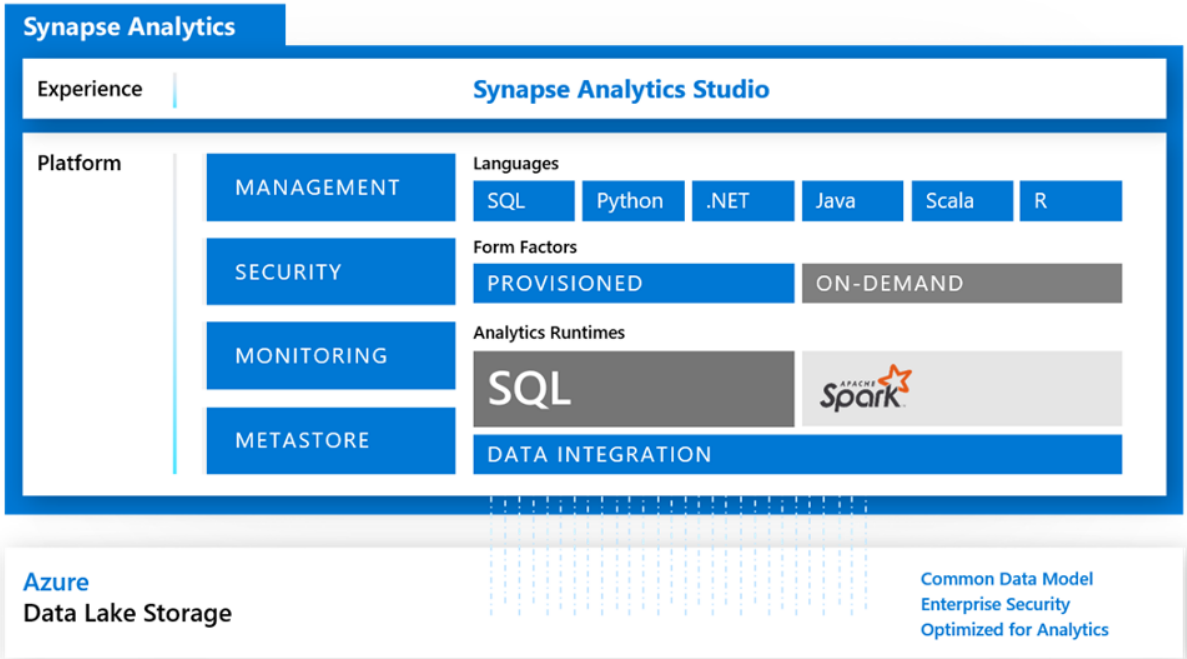
- **Azure Synapse Pipelines**

  (Correct)

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 82: **Incorrect**
Scenario: You are working as a consultant for Advanced Idea Mechanics (AIM) where the the IT team is working on an an Azure SQL database named AIM_Targets which contains a table named Targets_2021. This table has a field named Target_ID which is varchar(22).

Required: The team is to implement masking for the Target_ID field as per the following:

• Set the initial three prefix characters as "exposed".

• Set the final three suffix characters as "exposed".

• The remaining characters as "masked".

The team is planning to utilize data masking with a credit card function mask.

Will this solution meet the requirements?

- **No**

  (Correct)

- **Yes**

  (Incorrect)

Explanation

Using data masking with a credit card function mask will not be successful. To meet the requirements, AIM must use Custom Text data masking, which exposes the first and last characters as specified and adds a custom padding string in the middle.

Azure SQL Database, Azure SQL Managed Instance, and Azure Synapse Analytics support dynamic data masking. Dynamic data masking limits sensitive data exposure by masking it to non-privileged users.

Dynamic data masking helps prevent unauthorized access to sensitive data by enabling customers to designate how much of the sensitive data to reveal with minimal impact on the application layer. It's a policy-based security feature that hides the sensitive data in the result set of a query over designated database fields, while the data in the database is not changed.

For example, a service representative at a call centre might identify a caller by confirming several characters of their email address, but the complete email address shouldn't be revealed to the service representative. A masking rule can be defined that masks all the email address in the result set of any query. As another example, an appropriate data mask can be defined to protect personal data, so that a developer can query production environments for troubleshooting purposes without violating compliance regulations.

Dynamic data masking basics

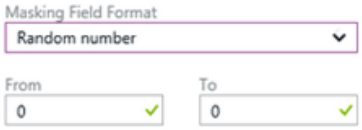
You set up a dynamic data masking policy in the Azure portal by selecting the **Dynamic Data Masking** blade under **Security** in your SQL Database configuration pane. This feature cannot be set using portal for SQL Managed Instance (use PowerShell or REST API). For more information, see Dynamic Data Masking.

Dynamic data masking policy

SQL users excluded from masking - A set of SQL users or Azure AD identities that 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 - A set of rules that define the designated fields to be masked and the masking function that is used. The designated fields can be defined using a database schema name, table name, and column name.

Masking functions - A set of methods that control the exposure of data for different scenarios.

Masking function	Masking logic
Default	**Full masking according to the data types of the designated fields** • Use XXXX or fewer Xs if the size of the field is less than 4 characters for string data types (nchar, ntext, nvarchar). • Use a zero value for numeric data types (bigint, bit, decimal, int, money, numeric, smallint, smallmoney, tinyint, float, real). • Use 01-01-1900 for date/time data types (date, datetime2, datetime, datetimeoffset, smalldatetime, time). • For SQL variant, the default value of the current type is used. • For XML the document <masked/> is used. • Use an empty value for special data types (timestamp table, hierarchyid, GUID, binary, image, varbinary spatial types).
Credit card	**Masking method, which exposes the last four digits of the designated fields** and adds a constant string as a prefix in the form of a credit card. XXXX-XXXX-XXXX-1234
Email	**Masking method, which exposes the first letter and replaces the domain with XXX.com** using a constant string prefix in the form of an email address. aXX@XXXX.com
Random number	**Masking method, which generates a random number** according to the selected boundaries and actual data types. If the designated boundaries are equal, then the masking function is a constant number. 
Custom text	**Masking method, which exposes the first and last characters** and adds a custom padding string in the middle. If the original string is shorter than the exposed prefix and suffix, only the padding string is used. prefix[padding]suffix

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

Question 83: **Incorrect**
Scenario: The Serpent Society is a business enterprise considered one of the best-organized, most successful sector coalitions in operation today. Seth Voelker is the founder of the Society and has come to you for assistance as a Microsoft Azure Expert.

Seth has asked you to use Azure Synapse Link for SQL to replicate tales from SQL Server 2022 to Azure Synapse Analytics.

In order to do this, what additional Azure resource must you create?

- **Azure Key Vault**
- **Azure Data Lake Storage Gen2**

  (Correct)
- **Azure Firewall Service**
- **Azure Application Insights**

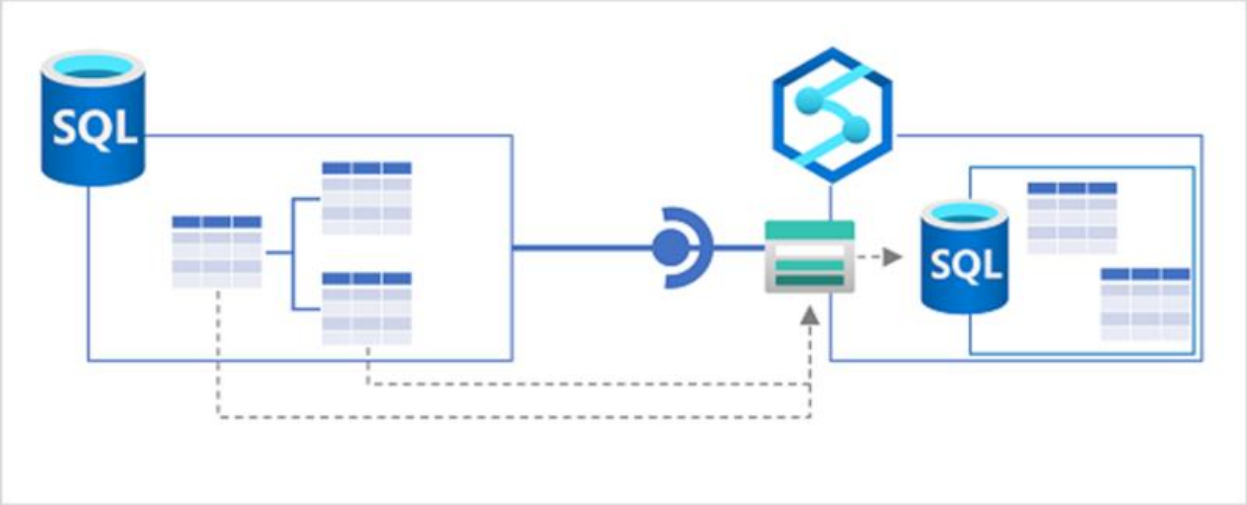
  (Incorrect)

Explanation
An Azure Data Lake Storage Gen2 account is required to be used as a landing zone when using Azure Synapse Link for 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:

```
1. SQL
2. 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:

```
1. SQL
2. 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 84: **Incorrect**
Managing default network access rules for Azure Storage accounts can be done with which of the following?

- ☐

  PowerShell

  (Correct)

- ☐

  Azure CLI

  (Correct)

- ☐

  ARM templates

- ☐

  Azure Portal

  (Correct)

- ☐

  Azure Designer

- ☑

  Azure CloudShell

  (Incorrect)

Explanation
Manage Default Network Access Rules

Manage default network access rules for storage accounts through the Azure portal, PowerShell, or the Azure CLI.

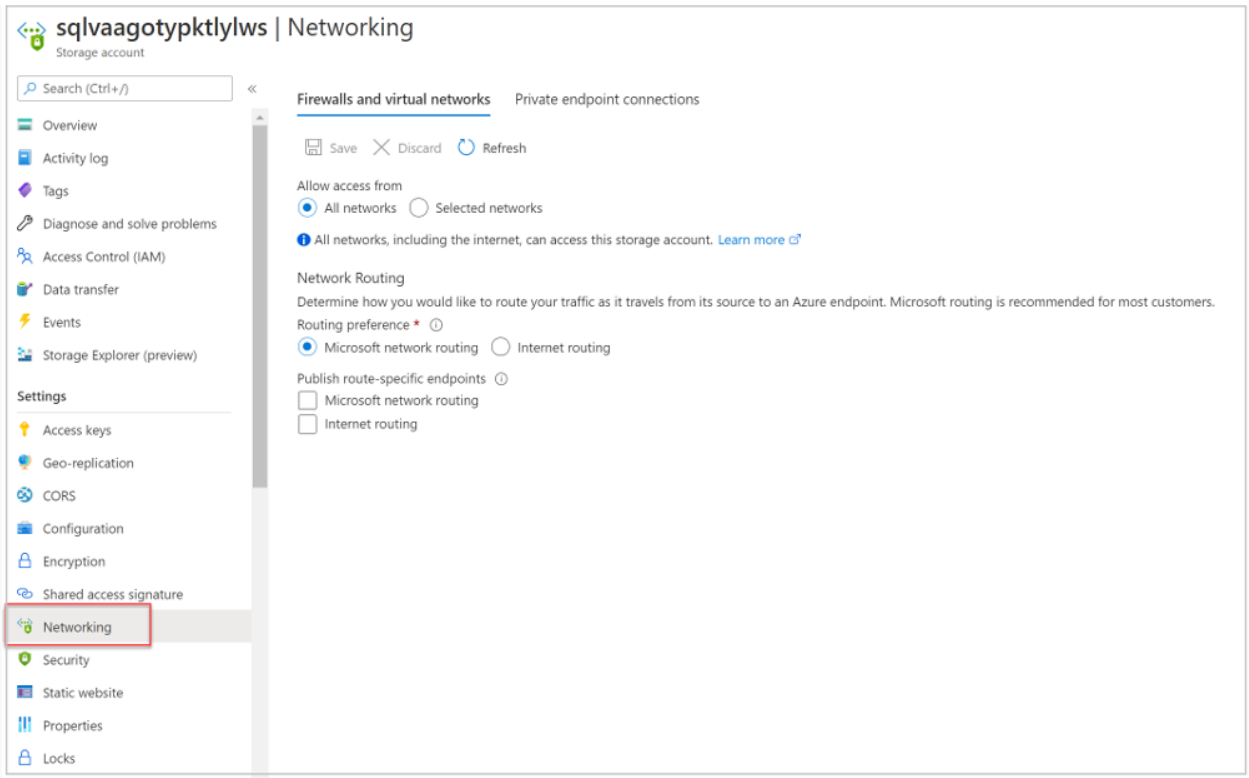
Follow these steps to change default network access in the Azure portal.

1. Go to the storage account you want to secure.

2. Select **Networking**.

3. To restrict traffic from selected networks, select **Selected networks**. To allow traffic from all networks, select **All networks**.

4. To apply your changes, select **Save**.



Note: The keyphrase here is "Managing Default", not modifying specific instances.

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

Question 85: **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. Azure Blob storage is an object storage solution optimized for storing massive amounts of unstructured data, such as text or binary data.

Azure Storage supports which kinds of blobs? (Select three)

- ☐

  Page

  (Correct)

- ☐

  GPv2

- ☐

  Queue

- ☐

  Block

  (Correct)

- ☑

  GPv1

  (Incorrect)

- ☐

  Append

  (Correct)

Explanation
Microsoft Azure Storage is a managed service that provides durable, secure, and scalable storage in the cloud.

Durable	Redundancy ensures that your data is safe in the event of transient hardware failures. You can also replicate data across datacenters or geographical regions for extra protection from local catastrophe or natural disaster. Data replicated in this way remains highly available in the event of an unexpected outage.
Secure	All data written to Azure Storage is encrypted by the service. Azure Storage provides you with fine-grained control over who has access to your data.
Scalable	Azure Storage is designed to be massively scalable to meet the data storage and performance needs of today's applications.
Managed	Microsoft Azure handles maintenance and any critical problems for you.

A single Azure subscription can host up to 200 storage accounts, each of which can hold 500 TB of data.

Azure data services

Azure storage includes four types of data:

• Azure Blobs: A massively scalable object store for text and binary data. Can include support for Azure Data Lake Storage Gen2.

• **Files**: Managed file shares for cloud or on-premises deployments.

• Azure Queues: A messaging store for reliable messaging between application components.

• Azure Tables: A NoSQL store for schema-less storage of structured data. Table Storage is not covered in this module.

• Azure Disks: Block-level storage volumes for Azure VMs.

All of these data types in Azure Storage are accessible from anywhere in the world over HTTP or HTTPS. Microsoft provides SDKs for Azure Storage in various languages, and a REST API. You can also visually explore your data right in the Azure portal.

Blob storage

Azure Blob storage is an object storage solution optimized for storing massive amounts of unstructured data, such as text or binary data. Blob storage is ideal for:

• Serving images or documents directly to a browser, including full static websites.

• Storing files for distributed access.

• Streaming video and audio.

• Storing data for backup and restoration, disaster recovery, and archiving.

• Storing data for analysis by an on-premises or Azure-hosted service.

Azure Storage supports three kinds of blobs:

Blob type	Description
Block blobs	Block blobs are used to hold text or binary files up to ~5 TB (50,000 blocks of 100 MB) in size. The primary use case for block blobs is the storage of files that are read from beginning to end, such as media files or image files for websites. They are named block blobs because files larger than 100 MB must be uploaded as small blocks. These blocks are then consolidated (or committed) into the final blob.
Page blobs	Page blobs are used to hold random-access files up to 8 TB in size. Page blobs are used primarily as the backing storage for the VHDs used to provide durable disks for Azure Virtual Machines (Azure VMs). They are named page blobs because they provide random read/write access to 512-byte pages.
Append blobs	Append blobs are made up of blocks like block blobs, but they are optimized for append operations. These blobs are frequently used for logging information from one or more sources into the same blob. For example, you might write all of your trace logging to the same append blob for an application running on multiple VMs. A single append blob can be up to 195 GB.

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

Question 86: **Incorrect**

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

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.

Azure Synapse Analytics uses the [?] approach for bulk data.

- **Automated Data Processing Equipment (ADPE)**
- **Extract, Load, and Transform (ELT)**

  (Correct)
- **Extract, Transform, and Load (ETL)**

- Atomicity, Consistency, Isolation, and Durability (ACID)

  (Incorrect)

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.

Ingesting and processing data

Azure Synapse Analytics uses the extract, load, and transform (ELT) approach for bulk data. SQL professionals are already familiar with bulk-copy tools such as bcp and the SQLBulkCopy API. Data engineers who work with Azure Synapse Analytics will soon learn how quickly PolyBase can load data.

PolyBase is a technology that removes complexity for data engineers. They take advantage of techniques for big-data ingestion and processing by offloading complex calculations to the cloud. Developers use PolyBase to apply stored procedures, labels, views, and SQL to their applications. You can also use Azure Data Factory to ingest and process data using PolyBase too.

Queries

As a data engineer, you can use the familiar Transact-SQL to query the contents of Azure Synapse Analytics. This method takes advantage of a wide range of features, including the WHERE, ORDER BY, and GROUP BY clauses. Load data fast by using PolyBase with additional Transact-SQL constructs such as `CREATE TABLE` and `SELECT`.

Data security

Azure Synapse Analytics supports both SQL Server authentication and Azure Active Directory. For high-security environments, set up multifactor authentication. From a data perspective, Azure Synapse Analytics supports security at the level of both columns and rows.

https://docs.microsoft.com/en-us/azure/architecture/reference-architectures/data/enterprise-bi-synapse

Question 87: **Incorrect**
Which T-SQL Statement loads data directly from Azure Storage?

- `LOAD DATA`
- `COPY`

  (Correct)
- `PULL`
- `INSERT FROM FILE`
- `DUPLICATE`
- `GET`

  (Incorrect)

Explanation
The T-SQL COPY Statement reads data from Azure Blob Storage or the Azure Data Lake and inserts it into a table within the SQL Pool.

The broad capabilities of the Copy Activity allow you to quickly and easily move data into SQL Pools from a variety of sources.

In Azure Data Factory, you can use the Copy activity to copy data among data stores located on-premises and in the cloud. After you copy the data, you can use other activities to further transform and analyze it. You can also use the Copy activity to publish transformation and analysis results for business intelligence (BI) and application consumption.

The Copy activity is executed on an integration runtime. You can use different types of integration runtimes for different data copy scenarios:

• When you're copying data between two data stores that are publicly accessible through the internet from any IP, you can use the Azure integration runtime for the copy activity. This integration runtime is secure, reliable, scalable, and globally available.

• When you're copying data to and from data stores that are located on-premises or in a network with access control (for example, an Azure virtual network), you need to set up a self-hosted integration runtime.

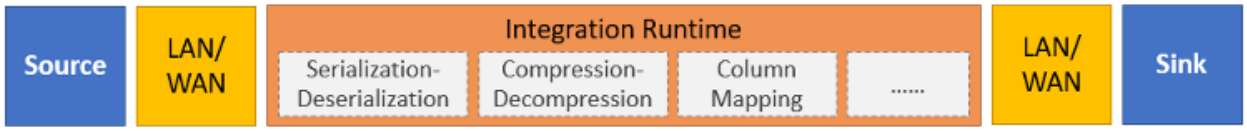
An integration runtime needs to be associated with each source and sink data store. For information about how the Copy activity determines which integration runtime to use, see Determining which IR to use.

To copy data from a source to a sink, the service that runs the Copy activity performs these steps:

1. Reads data from a source data store.

2. Performs serialization/deserialization, compression/decompression, column mapping, and so on. It performs these operations based on the configuration of the input dataset, output dataset, and Copy activity.

3. Writes data to the sink/destination data store.



The Copy Activity supports a large range of data sources and sinks on-premises and in the cloud. It facilitates the efficient, yet flexible parsing and transfer of data or files between systems in an optimized fashion as well as giving you capability of easily converting datasets into other formats.

In the following example, you can load data from a public storage account. Here the `COPY` statement's defaults match the format of the line item csv file.

```
1. SQL
2. COPY INTO dbo.[lineitem] FROM
   'https://unsecureaccount.blob.core.windows.net/customerdatasets/folder1/lineitem.csv'
```

The default values for csv files of the COPY command are:

- `DATEFORMAT = Session DATEFORMAT`
- `MAXERRORS = 0`
- `COMPRESSION` default is uncompressed
- `FIELDQUOTE = “”`
- `FIELDTERMINATOR = “,”`
- `ROWTERMINATOR = ‘\n’`
- `FIRSTROW = 1`
- `ENCODING = ‘UTF8’`
- `FILE_TYPE = ‘CSV’`
- `IDENTITY_INSERT = ‘OFF’`

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

Question 88: **Incorrect**
Azure Data Lake Storage Gen2 is a set of capabilities dedicated to big data analytics which stores data an HDFS-compatible file system hosted in Azure Storage.

Which of the following options must you enable to use Azure Data Lake Storage Gen2?

- C

Global replication

- ○

Data encryption

- ○

Hierarchical namespace

(Correct)

- ◉

Hierarchical namespace and Global replication

(Incorrect)

Explanation

- *To enable Azure Data Lake Storage Gen2 containers, you must turn on the Hierarchical namespace option.*
- Global replication does not enable Azure Data Lake Storage Gen2 containers.
- Data encryption does not enable Azure Data Lake Storage Gen2 containers.

In Azure Blob storage, you can store large amounts of unstructured ("object") data, in a single hierarchy, also known as a flat namespace. You can access this data by using HTTP or HTTPs. Azure Data Lake Storage Gen2 builds on blob storage and optimizes I/O of high-volume data by using hierarchical namespaces that you turned on in the previous exercise.

Hierarchical namespaces organize blob data into *directories* and stores metadata about each directory and the files within it. This structure allows operations, such as directory renames and deletes, to be performed in a single atomic operation. Flat namespaces, by contrast, require several operations proportionate to the number of objects in the structure. Hierarchical namespaces keep the data organized, which yields better storage and retrieval performance for an analytical use case and lowers the cost of analysis.

Azure Blob storage vs. Azure Data Lake Storage

If you want to store data *without performing analysis on the data*, set the **Hierarchical Namespace** option to **Disabled** to set up the storage account as an Azure Blob storage account. You can also use blob storage to archive rarely used data or to store website assets such as images and media.

If you are performing analytics on the data, set up the storage account as an Azure Data Lake Storage Gen2 account by setting the **Hierarchical Namespace** option to **Enabled**. Because Azure Data Lake Storage Gen2 is integrated into the Azure Storage platform, applications can use either the Blob APIs or the Azure Data Lake Storage Gen2 file system APIs to access data.

https://blog.pragmaticworks.com/azure-data-lake-vs-azure-blob-storage-in-data-warehousing

Question 89: **Correct**
Microsoft Power BI is used by organizations all around the world to create dynamic, interactive data visualizations that reveal insights on which important business decisions are based. Access to timely data can be the difference between failure and success, so the ability to capture and visualize data in real-time, or as near as possible, is critical in many scenarios.

Which type of Azure Stream Analytics output should you use to support real-time visualizations in Microsoft Power BI?

- **Power BI**

  (Correct)

- **SharePoint Online**

-

  Azure Event Hubs

-

  Azure Synapse Analytics

Explanation

- *A Power BI output creates a dataset with a table of streaming data in a Power BI workspace.*
- An Azure Synapse Analytics output writes data to a table in an Azure Synapse Analytics dedicated SQL pool.
- An Azure Event Hubs output writes data to an event hub for downstream processing.
- SharePoint Online empowers teamwork with dynamic and productive team sites for every project team, department, and division.

Use a Power BI output in Azure Stream Analytics

All Azure Stream Analytics jobs include at least one input and output. In most cases, inputs reference sources of streaming data (though you can also define inputs for static reference data to augment the streamed event data). Outputs determine where the results of the stream processing query will be sent. To support real-time data visualization, you can use a **Power BI** output.

Streaming data inputs

Inputs for streaming data consumed by Azure Stream Analytics can include:

- Azure Event Hubs
- Azure IoT Hubs
- Azure Blob or Data Lake Gen 2 Storage

Depending on the specific input type, the data for each streamed event includes the event's data fields and input-specific metadata fields. For example, data consumed from an Azure Event Hubs input includes an **EventEnqueuedUtcTime** field indicating the time when the event was received in the event hub.

https://learn.microsoft.com/en-us/azure/stream-analytics/stream-analytics-define-inputs

Power BI outputs

You can use a Power BI output to write the results of a Stream Analytics query to a table in a Power BI streaming dataset, from where it can be visualized in a dashboard. When adding a Power BI output to a Stream Analytics job, you need to specify the following properties:

- **Output alias**: A name for the output that can be used in a query.
- **Group workspace**: The Power BI workspace in which you want to create the resulting dataset.
- **Dataset name**: The name of the dataset to be generated by the output. You shouldn't pre-create this dataset as it will be created automatically (replacing any existing dataset with the same name).
- **Table name**: The name of the table to be created in the dataset.
- **Authorize connection**: You must authenticate the connection to your Power BI tenant so that the Stream Analytics job can write data to the workspace.

https://learn.microsoft.com/en-us/azure/stream-analytics/power-bi-output

Question 90: **Correct**
Scenario: Lobo Technologies is a subsidiary of Worthington Industries, owned by Maximus Lobo, headquartered in White Plains, New York.

James Howlett aka Logan, is a developer at Lobo and is going to create a database in SQL warehouse.

Which statement should Logan use to create it?

- ○

  `CREATE DBASE`

- ○

  `CREATE VIEW`

- ○

  `CREATE GROUP`

- ◉

`CREATE SCHEMA`

(Correct)

Explanation

The `CREATE SCHEMA` statement is used to create a database.

Create databases and tables

After creating and starting a SQL Warehouse, you can start to work with data in tables.

Databases

All SQL Warehouses contain a default database named **default**. You can use create tables in this database in order to analyze data. However, if you need to work with multiple tables in a relational schema, or you have multiple analytical workloads where you want to manage the data (and access to it) separately, you can create custom databases. To create a database, use the SQL editor to run a `CREATE DATABASE` or `CREATE SCHEMA` SQL statement. These statements are equivalent, but `CREATE SCHEMA` is preferred, as shown in this example:

```
1. SQL
2. CREATE SCHEMA salesdata;
```

https://learn.microsoft.com/en-us/azure/databricks/sql/language-manual/sql-ref-syntax-ddl-create-schema

Tables

You can use the user interface in the Azure Databricks portal to upload delimited data, or import data from a wide range of common data sources. The imported data is stored in files in Databricks File System (DBFS) storage, and a Delta table is defined for it in the Hive metastore.

If the data files already exist in storage, or you need to define an explicit schema for the table, you can use a `CREATE TABLE` SQL statement. For example, the following code creates a table named salesorders in the salesdata database, based on the /data/sales/ folder in DBFS storage.

```
SQL
CREATE TABLE salesdata.salesorders
(
    orderid INT,
    orderdate DATE,
    customerid INT,
    ordertotal DECIMAL
)
USING DELTA
LOCATION '/data/sales/';
```

https://learn.microsoft.com/en-us/azure/databricks/sql/language-manual/sql-ref-syntax-ddl-create-table

Question 91: **Correct**

`azure-cosmosdb-spark` is the official connector for Azure CosmosDB and Apache Spark. The connector allows you to easily read to and write from Azure Cosmos DB via Apache Spark DataFrames in `python` and `scala`. It also allows you to easily create a lambda architecture for batch-processing, stream-processing, and a serving layer while being globally replicated and minimizing the latency involved in working with big data.

Which of the following formats should you specify if you are writing PySpark code to load data from an Azure Cosmos DB analytical store into a dataframe?

- `cosmos.json`
- `cosmos.pys`
- `cosmos.sql`
- `cosmos.olap`

  (Correct)

Explanation

- `cosmos.olap` *is the appropriate format to read data from a Cosmos DB analytical store.*
- `cosmos.json` , `cosmos.sql` , and `cosmos.pys` are not valid formats.

`cosmos.olap` is the method that connects to the analytical store in Azure Cosmos DB.

The syntax to create a Spark table is as follows:

```
1. SQL
2. %%sql
3. -- To select a preferred list of regions in a multi-region Azure Cosmos DB
   account, add spark.cosmos.preferredRegions '<Region1>,<Region2>' in the config
   options
4.
5. create table call_center using cosmos.olap options (
6. spark.synapse.linkedService '<enter linked service name>',
7. spark.cosmos.container '<enter container name>'
8. )
```

https://docs.microsoft.com/en-us/azure/synapse-analytics/synapse-link/how-to-query-analytical-store-spark

Question 92: **Incorrect**
While Agile, CI/CD, and DevOps are different, they support one another

What does DevOps focus on?

- **Practices**
- **Culture**

  (Correct)

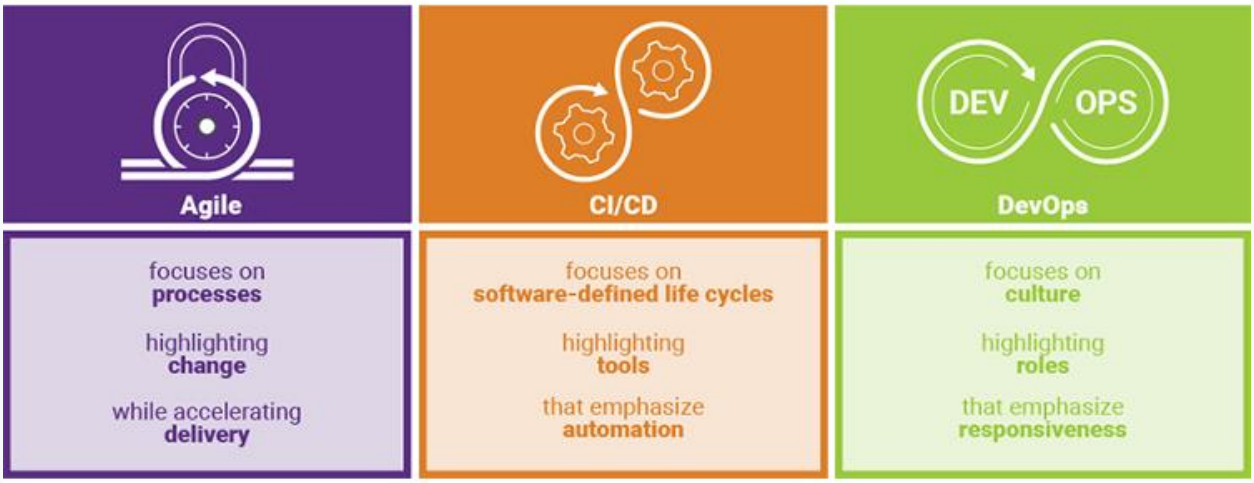
Development process

-

Strategy

(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 93: **Correct**
Azure infrastructure is composed of geographies, regions, and Availability Zones, which limit the blast radius of a failure and therefore limit potential impact to customer applications and data. Duplicating customer content for redundancy and meeting service-level agreements (SLAs) in Azure meets which cloud technical requirement?

- **None of the listed options.**
- **Maintainability**
- **All the listed options.**
-

Content distribution guarantees

-

Multilingual support

-

High availability

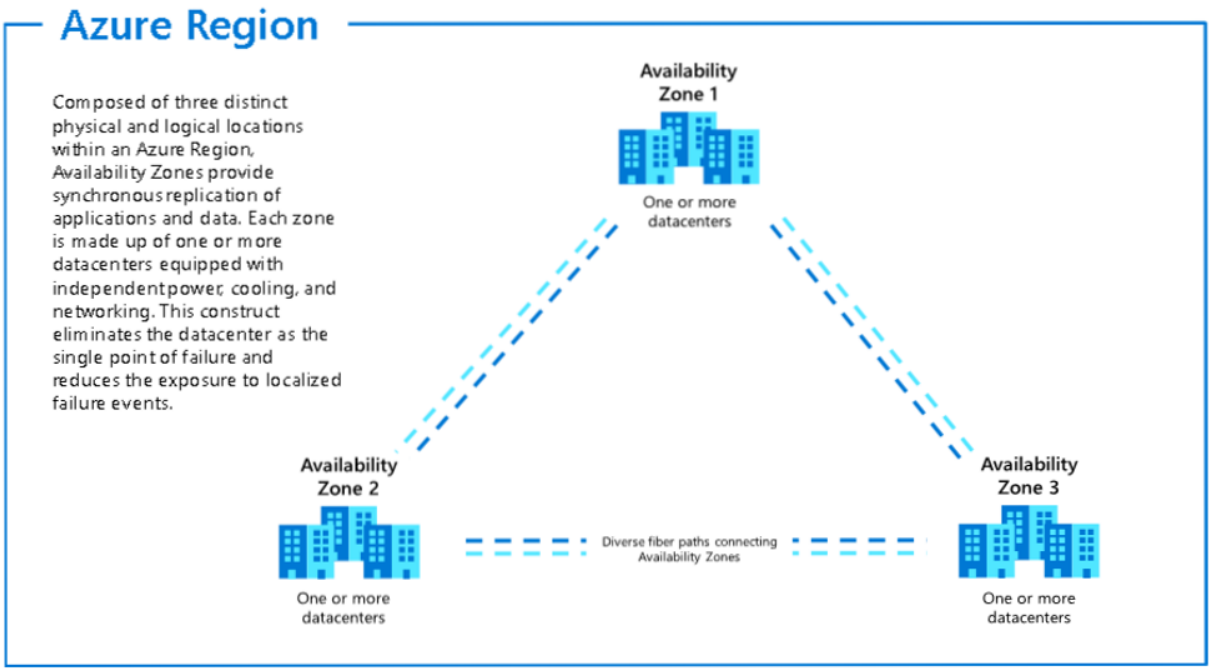
(Correct)

Explanation
High availability duplicates customer content for redundancy and meets SLAs in Azure.

Microsoft Azure global infrastructure is designed and constructed at every layer to deliver the highest levels of redundancy and resiliency to its customers. Azure infrastructure is composed of geographies, regions, and Availability Zones, which limit the blast radius of a failure and therefore limit potential impact to customer applications and data. The Azure Availability Zones construct was developed to provide a software and networking solution to protect against datacentre failures and to provide increased high availability (HA) to our customers.

Availability Zones are unique physical locations within an Azure region. Each zone is made up of one or more datacentres with independent power, cooling, and networking. The physical separation of Availability Zones within a region limits the impact to applications and data from zone failures, such as large-scale flooding, major storms and superstorms, and other events that could disrupt site access, safe passage, extended utilities uptime, and the availability of resources. Availability Zones and their associated datacentres are designed such that if one zone is compromised, the services, capacity, and availability are supported by the other Availability Zones in the region.

Availability Zones can be used to spread a solution across multiple zones within a region, allowing for an application to continue functioning when one zone fails. With Availability Zones, Azure offers industry best 99.99% Virtual Machine (VM) uptime service-level agreement (SLA). Zone-redundant services replicate your services and data across Availability Zones to protect from single points of failure.



https://docs.microsoft.com/en-us/azure/architecture/high-availability/building-solutions-for-high-availability

Question 94: **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 the following:

"A legacy account type which supports only block and append blobs."

- **Page storage accounts**
- **GPv2 storage accounts**
- **Queue storage accounts**

- **Block storage accounts**

- **GPv1 storage accounts**

- **Append storage accounts**

  (Incorrect)

- **Blob storage accounts**

  (Correct)

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 95: **Correct**

Scenario: You are working as a consultant at **Advanced Idea Mechanics** (**A.I.M.**) who is a privately funded think tank organized of a group of brilliant scientists whose sole dedication is to acquire and develop power through technological means. Their goal is to use this power to overthrow the governments of the world. They supply arms and technology to radicals and subversive organizations in order to foster a violent technological revolution of society while making a profit.

The company has 10,000 employees. Most employees are located in Europe. The company supports teams worldwide.

AIM has two main locations: a main office in London, England, and a manufacturing plant in Berlin, Germany.

During events, 100 engineers set up a remote portable office by using a VPN to connect the datacentre in the London office. The portable office is set up and torn down in approximately 20 different countries each year.

Chaos Central

During major events, AIM uses a primary application named Chaos Central. Each vehicle used in the activity has several sensors that send real-time telemetry data to the London datacentre. The data is used for real-time tracking of the vehicles. Chaos Central also sends batch updates to an application named Mechanical Workflow by using Microsoft SQL Server Integration Services (SSIS).

The telemetry data is sent to a MongoDB database. A custom application then moves the data to databases in SQL Server 2017. The telemetry data in MongoDB has more than 500 attributes. The application changes the attribute names when the data is moved to SQL Server 2017.

The database structure contains both OLAP and OLTP databases.

Mechanical Workflow

Mechanical Workflow is used to track changes and improvements made to the vehicles during their lifetime. Currently, Mechanical Workflow runs on SQL Server 2017 as an OLAP system. Mechanical Workflow has a named Table1 that is 1 TB. Large aggregations are performed on a single column of Table 1.

Requirements:

• Data collection for Chaos Central must be moved to Azure Cosmos DB and Azure SQL Database. The data must be written to the Azure datacentre closest to each race and must converge in the least amount of time.

• The query performance of Chaos Central must be stable, and the administrative time it takes to perform optimizations must be minimized.

• The datacentre for Mechanical Workflow must be moved to Azure SQL data Warehouse.

• Transparent data encryption (TDE) must be enabled on all data stores, whenever possible.

• An Azure Data Factory pipeline must be used to move data from Cosmos DB to SQL Database for Chaos Central. If the data load takes longer than 20 minutes, configuration changes must be made to Data Factory.

• The telemetry data must migrate toward a solution that is native to Azure.

• The telemetry data must be monitored for performance issues. You must adjust the Cosmos DB Request Units per second (RU/s) to maintain a performance SLA while minimizing the cost of the Ru/s.

Which of the following data stores should be configured to meet the TDE requirement?

- **SQL Data Warehouse**

  (Correct)

- **All of the listed items**

- **SQL Database**

- **Cosmos DB**

Explanation

Transparent data encryption (TDE) must be enabled on all data stores, whenever possible. The datacentre for Mechanical Workflow must be moved to Azure SQL data Warehouse. Cosmos DB does not support TDE.

Transparent data encryption (TDE) helps protect Azure SQL Database, Azure SQL Managed Instance, and Azure Synapse Analytics against the threat of malicious offline activity by encrypting data at rest. It performs real-time encryption and decryption of the database, associated backups, and transaction log files at rest without requiring changes to the application. By default, TDE is enabled for all newly deployed SQL Databases and must be manually enabled for older databases of Azure SQL Database, Azure SQL Managed Instance. TDE must be manually enabled for Azure Synapse Analytics.

TDE performs real-time I/O encryption and decryption of the data at the page level. Each page is decrypted when it's read into memory and then encrypted before being written to disk. TDE encrypts the storage of an entire database by using a symmetric key called the Database Encryption Key (DEK). On database startup, the encrypted DEK is decrypted and then used for decryption and re-encryption of the database files in the SQL Server database engine process. DEK is protected by the TDE protector. TDE protector is either a service-managed certificate (service-managed transparent data encryption) or an asymmetric key stored in Azure Key Vault (customer-managed transparent data encryption).

For Azure SQL Database and Azure Synapse, the TDE protector is set at the server level and is inherited by all databases associated with that server. For Azure SQL Managed Instance, the TDE protector is set at the instance level and it is inherited by all encrypted databases on that instance. The term *server* refers both to server and instance throughout this document, unless stated differently.

https://docs.microsoft.com/en-us/azure/azure-sql/database/transparent-data-encryption-tde-overview?tabs=azure-portal

Question 96: **Incorrect**
Within the context of Azure Databricks, sharing data from one worker to another can be a costly operation.

Spark has optimized this operation by using a format called [?] which prevents the need for expensive serialization and de-serialization of objects in order to get data from one JVM to another.

- **Stages**

- ○

  Pipelining

- ○

  Lineage

- ○

  Tungsten

  (Correct)

- ○

  Stage boundary

- ◉

  Shuffles

  (Incorrect)

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

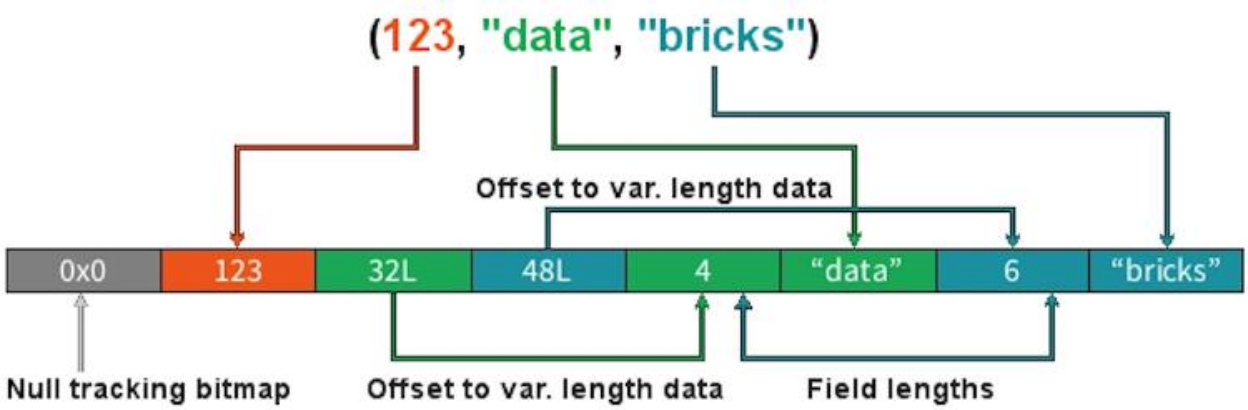
• The first field, "123", is stored in place as its primitive.

• 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.

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 97: **Incorrect**
Scenario: While working on a project, the need arises to develop T-SQL scripts and notebooks in Azure Synapse Analytics.

Which of the following may be used to accomplish this?

- **Azure Portal**
- **Azure Synapse Studio**

  (Correct)
- **Data Lake**
- **Databricks**
- **DevTest Labs**

  (Incorrect)

Explanation
Azure Synapse Studio is where you can develop T-SQL scripts and notebooks.

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

Question 98: **Incorrect**
Scenario: O'Shaughnessy's is a fast food restaurant. The chain has stores nationwide and is rivalled by Big Belly Burgers. You have been hired by the company to advise on the creation and implementation of a dimension table in Azure Data Warehouse.

Specifications:

• The dimension table will be less than 1 GB.

Required:

- Fastest available query time
- Minimize data movement

As the Azure expert, the O'Shaughnessy IT team looks to you for direction. Which of the following should you advise them to utilize?

- **Hash distributed**
- **Replicated**

  (Correct)
- **Heap**

  (Incorrect)
- **Round-robin**

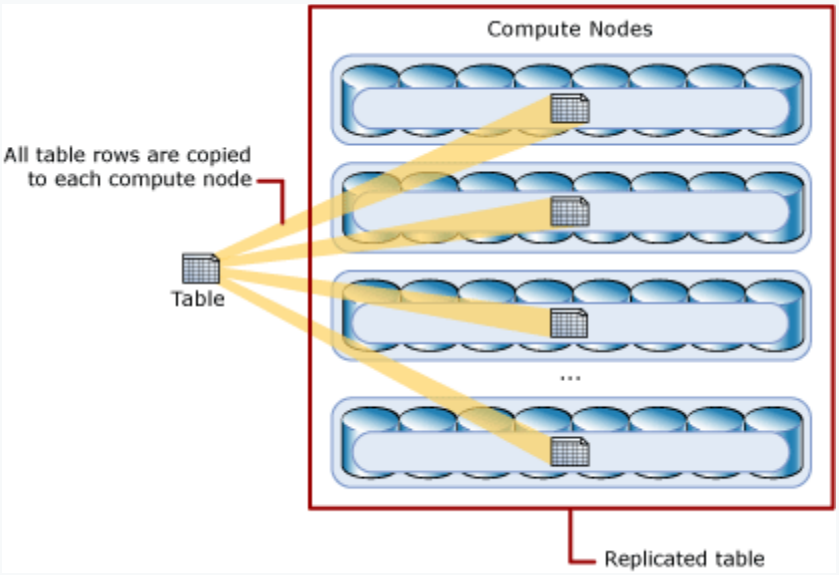
Explanation
Replicated tables work well for dimension tables in a star schema. Dimension tables are typically joined to fact tables which are distributed differently than the dimension table. Dimensions are usually of a size that makes it feasible to store and maintain multiple

copies. Dimensions store descriptive data that changes slowly, such as customer name and address, and product details. The slowly changing nature of the data leads to less maintenance of the replicated table.

What is a replicated table?

A replicated table has a full copy of the table accessible on each Compute node. Replicating a table removes the need to transfer data among Compute nodes before a join or aggregation. Since the table has multiple copies, replicated tables work best when the table size is less than 2 GB compressed. 2 GB is not a hard limit. If the data is static and does not change, you can replicate larger tables.

The following diagram shows a replicated table that is accessible on each Compute node. In SQL pool, the replicated table is fully copied to a distribution database on each compute node.



Replicated tables work well for dimension tables in a star schema. Dimension tables are typically joined to fact tables which are distributed differently than the dimension table. Dimensions are usually of a size that makes it feasible to store and maintain multiple copies. Dimensions store descriptive data that changes slowly, such as customer name and address, and product details. The slowly changing nature of the data leads to less maintenance of the replicated table.

Consider using a replicated table when:

• The table size on disk is less than 2 GB, regardless of the number of rows. To find the size of a table, you can use the `DBCPDW_SHOWSPACEUSED` command: `DBCC PDW_SHOWSPACEUSED('ReplTableCandidate')`.

• The table is used in joins that would otherwise require data movement. When joining tables that are not distributed on the same column, such as a hash-distributed table to a round-robin table, data movement is required to complete the query. If one of the tables is small, consider a replicated table. We recommend using replicated tables instead of round-robin tables in most cases. To view data movement operations in query plans, use `sys.dm_pdw_request_steps`. The `BroadcastMoveOperation` is the typical data movement operation that can be eliminated by using a replicated table.

Replicated tables may not yield the best query performance when:

• The table has frequent insert, update, and delete operations. The data manipulation language (DML) operations require a rebuild of the replicated table. Rebuilding frequently can cause slower performance.

• The SQL pool is scaled frequently. Scaling a SQL pool changes the number of Compute nodes, which incurs rebuilding the replicated table.

• The table has a large number of columns, but data operations typically access only a small number of columns. In this scenario, instead of replicating the entire table, it might be more effective to distribute the table, and then create an index on the frequently accessed columns. When a query requires data movement, SQL pool only moves data for the requested columns.

Use replicated tables with simple query predicates

Before you choose to distribute or replicate a table, think about the types of queries you plan to run against the table. Whenever possible,

• Use replicated tables for queries with simple query predicates, such as equality or inequality.

• Use distributed tables for queries with complex query predicates, such as LIKE or NOT LIKE.

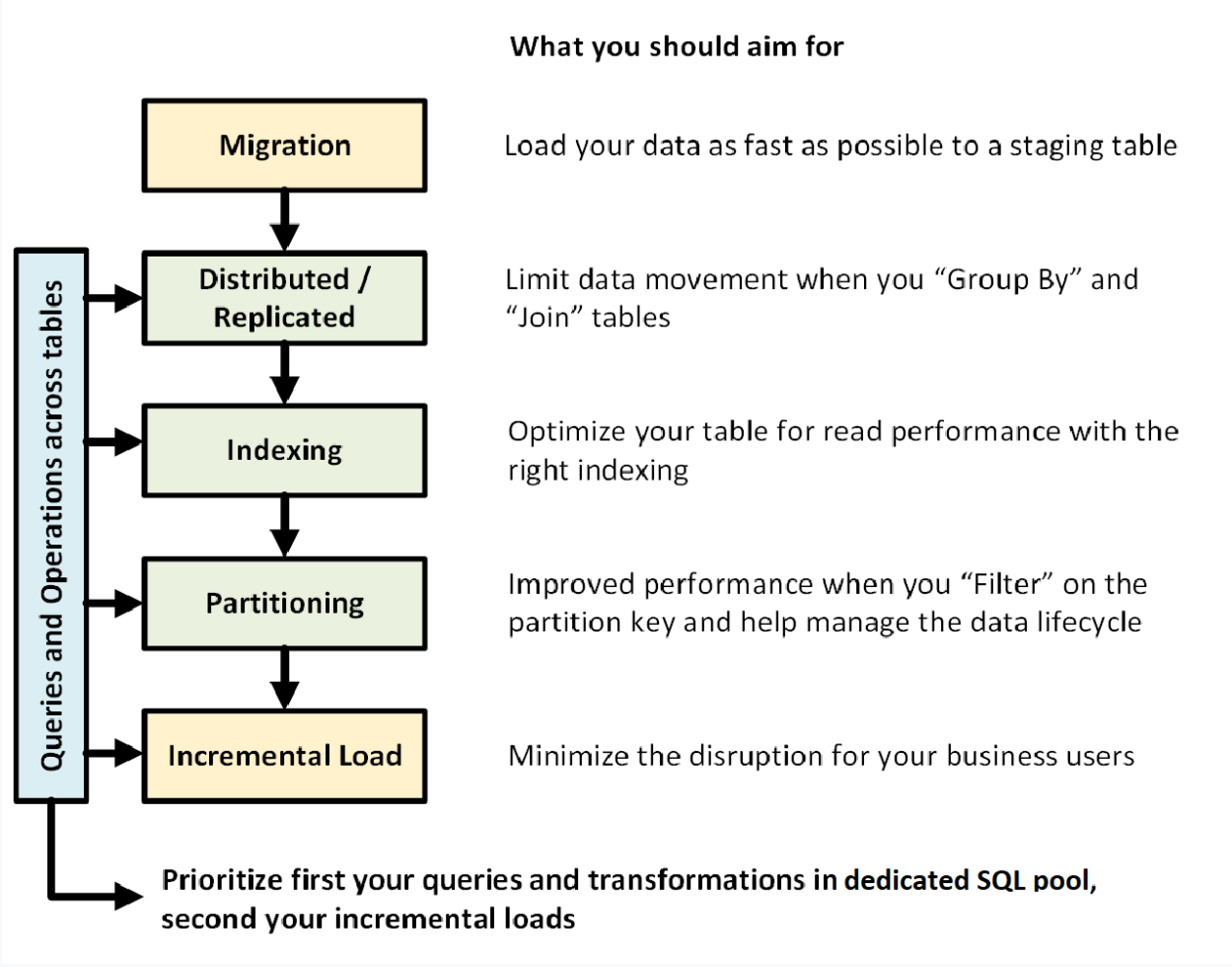
CPU-intensive queries perform best when the work is distributed across all of the Compute nodes. For example, queries that run computations on each row of a table perform better on distributed tables than replicated tables. Since a replicated table is stored in full on each Compute node, a CPU-intensive query against a replicated table runs against the entire table on every Compute node. The extra computation can slow query performance.

For example, this query has a complex predicate. It runs faster when the data is in a distributed table instead of a replicated table. In this example, the data can be round-robin distributed.

```
SQL
SELECT EnglishProductName
FROM DimProduct
WHERE EnglishDescription LIKE '%frame%comfortable%'
```

https://docs.microsoft.com/en-us/azure/synapse-analytics/sql-data-warehouse/design-guidance-for-replicated-tables

The following graphic shows the process of designing a data warehouse with dedicated SQL pool



Data migration

First, load your data into Azure Data Lake Storage or Azure Blob Storage. Next, use the COPY statement to load your data into staging tables. Use the following configuration:

Design	Recommendation
Distribution	Round Robin
Indexing	Heap
Partitioning	None
Resource Class	largerc or xlargerc

Distributed or replicated tables

Use the following strategies, depending on the table properties:

Type	Great fit for...	Watch out if...
Replicated	* Small dimension tables in a star schema with less than 2 GB of storage after compression (~5x compression)	* Many write transactions are on table (such as insert, upsert, delete, update) * You change Data Warehouse Units (DWU) provisioning frequently * You only use 2-3 columns but your table has many columns * You index a replicated table
Round Robin (default)	* Temporary/staging table * No obvious joining key or good candidate column	* Performance is slow due to data movement
Hash	* Fact tables * Large dimension tables	* The distribution key cannot be updated

Tips:

• Start with Round Robin, but aspire to a hash distribution strategy to take advantage of a massively parallel architecture.

• Make sure that common hash keys have the same data format.

• Don't distribute on varchar format.

• Dimension tables with a common hash key to a fact table with frequent join operations can be hash distributed.

• Use *sys.dm_pdw_nodes_db_partition_stats* to analyze any skewness in the data.

• Use *sys.dm_pdw_request_steps* to analyze data movements behind queries, monitor the time broadcast, and shuffle operations take. This is helpful to review your distribution strategy.

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

Question 99: **Incorrect**
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 which of the following indexing options available?

- ☐ **Clustered columnstore indexes**

  (Correct)

- ☐ **Clustered Rowstore Indexes**

  (Correct)

- ☐ **Hash indexes**

- ☐ **Key indexes**

- ☑ **Non-clustered index**

(Correct)

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

Non-clustered and clustered row-store indexes are handled the same way as in "classic" SQL Server engine, which implies they are built on B-Tree structure; it only applies to rowstore indexes, columnstore and in-memory tables are handled differently.

https://docs.microsoft.com/en-us/sql/relational-databases/indexes/clustered-and-nonclustered-indexes-described?view=sql-server-ver15#:~:text=Nonclustered%20indexes%20have%20a%20structure,is%20called%20a%20row%20locator

Question 100: **Incorrect**
When working with Azure Data Factory, before you create a dataset, you must create a Linked service to link your data store to the data factory.

You can programmatically define a linked service in the JSON format to be used via REST APIs or the SDK, using the following notation:

```
1. JSON
2. {
3. "name": "<Name of the linked service>",
4. "properties": {
5. "type": "<Type of the linked service>",
6. "typeProperties": {
7. "<data store or compute-specific type properties>"
8. },
9. "connectVia": {
10. "referenceName": "<name of Integration Runtime>",
11. "type": "IntegrationRuntimeReference"
12. }
13. }
14. }
```

Which of the JSON properties are required? (Select all that apply)

- ☑

  typeProperties

(Correct)

- ☐

name

(Correct)

- ☐

type

(Correct)

- ☐

connectVia

Explanation

Linked Service

When working with Azure Data Factory, before you create a dataset, you must create a **Linked service** to link your data store to the data factory. Linked services are much like connection strings, which define the connection information needed for Data Factory to connect to external resources. There are over 100 connectors that can be used to define a linked service.

A linked service in Data Factory can be defined using the Copy Data Activity in the ADF designer, or you can create them independently to point to a data store or a compute resources. The Copy Activity copies data between the source and destination, and when you run this activity you are asked to define a linked service as part of the copy activity definition

Alternatively you can programmatically define a linked service in the JSON format to be used via REST APIs or the SDK, using the following notation:

```
1. JSON
2. {
3. "name": "<Name of the linked service>",
4. "properties": {
5. "type": "<Type of the linked service>",
6. "typeProperties": {
7. "<data store or compute-specific type properties>"
8. },
9. "connectVia": {
10. "referenceName": "<name of Integration Runtime>",
11. "type": "IntegrationRuntimeReference"
```

```
}
}
}
```

The following describes properties in the above JSON:

Property: name

Name of the linked service.

Required: Yes

Property: type

Type of the linked service. For example: AzureStorage (data store) or AzureBatch (compute). See the description for typeProperties.

Required: Yes

Property: typeProperties

The type properties are different for each data store or compute. For the supported data store types and their type properties, see the dataset type table. Navigate to the data store connector article to learn about type properties specific to a data store.

Required: Yes

Property: connectVia

The Integration Runtime to be used to connect to the data store. You can use Azure Integration Runtime or Self-hosted Integration Runtime (if your data store is located in a private network). If not specified, it uses the default Azure Integration Runtime.

Required: No

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

Question 101: **Incorrect**
Continuous integration is the practice of testing each change made to your codebase automatically and as early as possible. Continuous delivery follows the testing that

happens during continuous integration and pushes changes to a staging or production system.

Below is a sample overview of the CI/CD lifecycle in an Azure data factory that's configured with Azure Repos Git.

The order of the activities has been shuffled.

a. A developer creates a feature branch to make a change. They debug their pipeline runs with their most recent changes.

b. A development data factory is created and configured with Azure Repos Git. All developers should have permission to author Data Factory resources like pipelines and datasets.

c. After a pull request is approved and changes are merged in the master branch, the changes get published to the development factory.

d. After the changes have been verified in the test factory, deploy to the production factory by using the next task of the pipelines release.

e. When the team is ready to deploy the changes to a test or UAT (User Acceptance Testing) factory, the team goes to their Azure Pipelines release and deploys the desired version of the development factory to UAT. This deployment takes place as part of an Azure Pipelines task and uses Resource Manager template parameters to apply the appropriate configuration.

f. After a developer is satisfied with their changes, they create a pull request from their feature branch to the master or collaboration branch to get their changes reviewed by peers.

Select the correct sequence of events in the CI/CD lifecycle.

- **a → b → c → f → d → e**

  (Incorrect)

- **b → a → f → d → e → c**

-

a → f → d → b → c → e

- **b → a → f → c → e → d**

  (Correct)

Explanation
Continuous integration is the practice of testing each change made to your codebase automatically and as early as possible. Continuous delivery follows the testing that happens during continuous integration and pushes changes to a staging or production system.

In Azure Data Factory, continuous integration and delivery (CI/CD) means moving Data Factory pipelines from one environment (development, test, production) to another. Azure Data Factory utilizes Azure Resource Manager templates to store the configuration of your various Azure Data Factory entities (pipelines, datasets, data flows, and so on). There are two suggested methods to promote a data factory to another environment:

• Automated deployment using Data Factory's integration with Azure Pipelines.

• Manually upload a Resource Manager template using Data Factory UX integration with Azure Resource Manager.

Continuous Integration/Continuous Delivery lifecycle

Below is a sample overview of the CI/CD lifecycle in an Azure data factory that's configured with Azure Repos Git.

1. A development data factory is created and configured with Azure Repos Git. All developers should have permission to author Data Factory resources like pipelines and datasets.

2. A developer creates a feature branch to make a change. They debug their pipeline runs with their most recent changes.

3. After a developer is satisfied with their changes, they create a pull request from their feature branch to the master or collaboration branch to get their changes reviewed by peers.

4. After a pull request is approved and changes are merged in the master branch, the changes get published to the development factory.

5. When the team is ready to deploy the changes to a test or UAT (User Acceptance Testing) factory, the team goes to their Azure Pipelines release and deploys the desired version of the development factory to UAT. This deployment takes place as part of an Azure Pipelines task and uses Resource Manager template parameters to apply the appropriate configuration.

6. After the changes have been verified in the test factory, deploy to the production factory by using the next task of the pipelines release.

https://docs.microsoft.com/en-us/azure/data-factory/continuous-integration-deployment

Question 102: **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.

Azure Synapse Analytics SME, the team looks to you for the best way forward. Which of the following should you recommend for the Sales table?

- **Distribution type: Round-robin, Distribution column: RegionKey**

  (Incorrect)

- **Distribution type: Round-robin, Distribution column: ProductKey**

- **Distribution type: Hash-distributed, Distribution column: RegionKey**

- **Distribution type: Hash-distributed, Distribution column: ProductKey**

  (Correct)

Explanation

You should recommend Hash-distributed for the Distribution type and ProductKey for the Distribution column.

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:

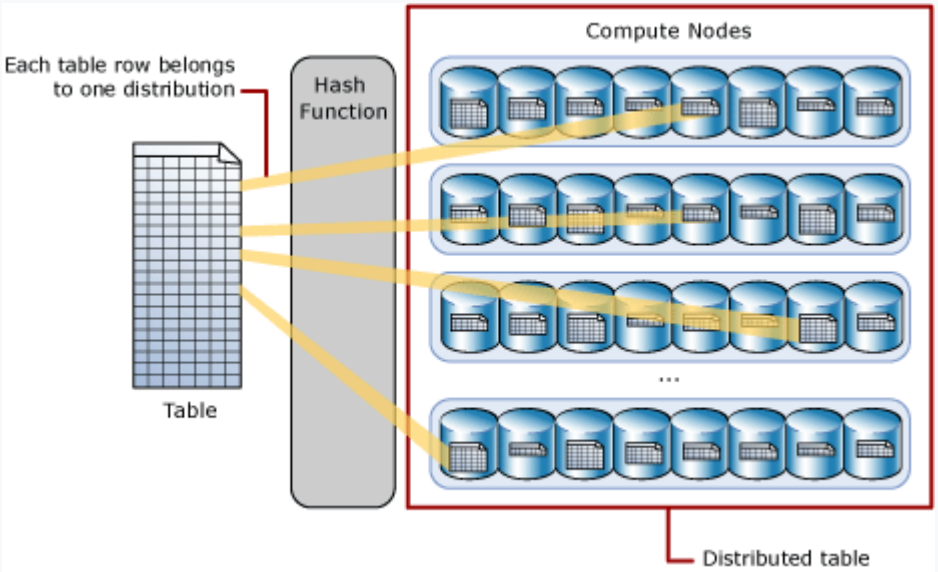
• How large is the table?

• 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 103: **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 must you create to run a pipeline at scheduled intervals?

- **A Trigger**

  (Correct)

- **A Control Flow**

- **An Orchestrator**

- **An Activity**

(Incorrect)

Explanation

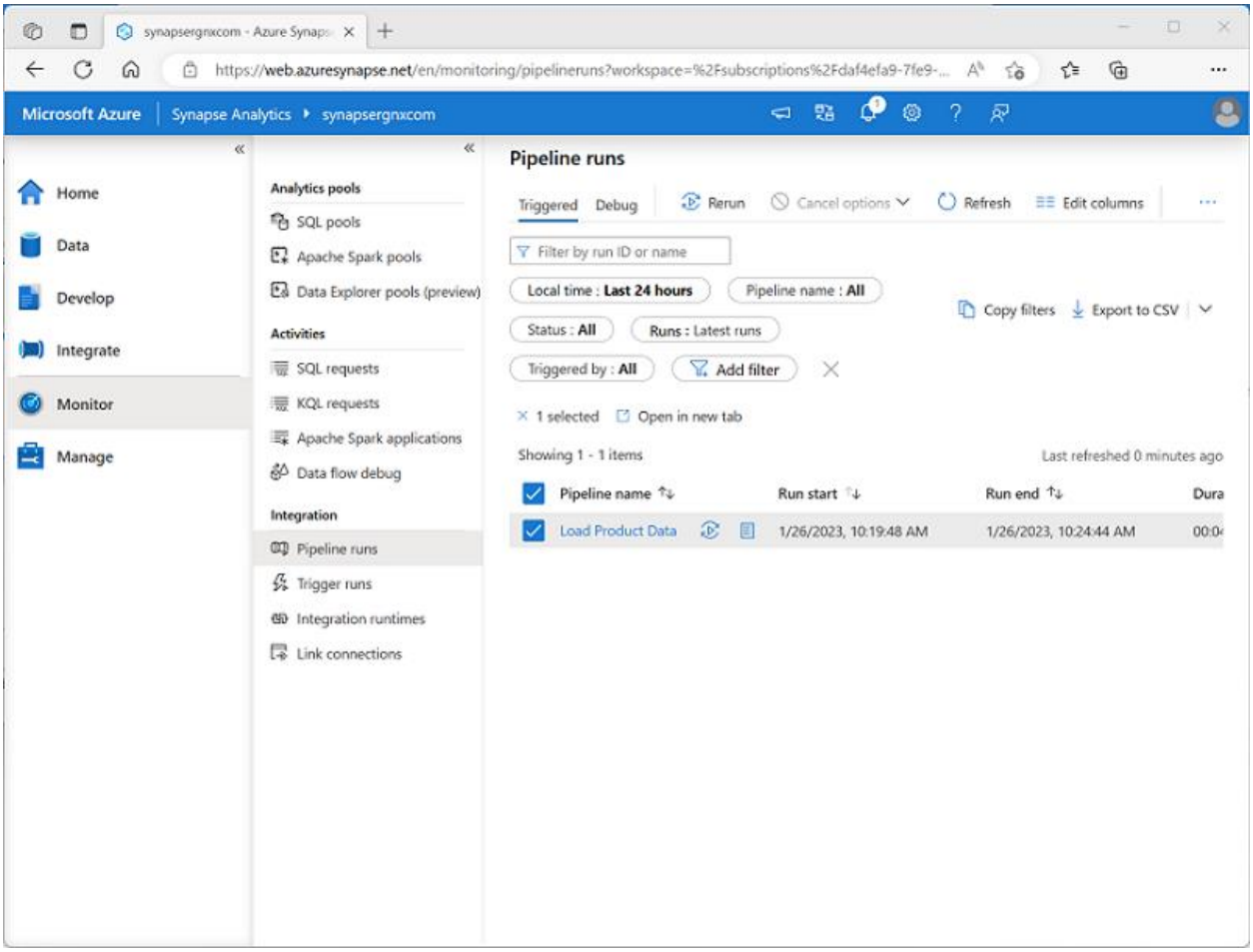
- *A trigger is used to initiate a pipeline run.*
- Control flow activities implement processing logic.
- Activities are the tasks within a pipeline.
- An orchestrator is not a valid option.

Run a pipeline

When you're ready, you can publish a pipeline and use a trigger to run it. Triggers can be defined to run the pipeline:

- Immediately
- At explicitly scheduled intervals
- In response to an event, such as new data files being added to a folder in a data lake.

You can monitor each individual run of a pipeline in the **Monitor** page in Azure Synapse Studio.



The ability to monitor past and ongoing pipeline runs is useful for troubleshooting purposes. Additionally, when combined with the ability to integrate Azure Synapse Analytics and Microsoft Purview, you can use pipeline run history to track data lineage data flows.

https://learn.microsoft.com/en-us/training/modules/integrate-microsoft-purview-azure-synapse-analytics

Question 104: **Incorrect**
In order to create a Spark pool in Azure Synapse Analytics, what needs to be created to do so?

- **Synapse Analytics Workspace**

(Correct)

- **Azure Databricks**

- **A Spark Instance**

- **HDI**

  (Incorrect)

Explanation
In order to create a Spark pool in Azure Synapse Analytics, you would have to create a Synapse Analytics Workspace.

https://docs.microsoft.com/en-us/azure/synapse-analytics/quickstart-create-apache-spark-pool-portal

Question 105: **Incorrect**
The pace of change in both the capabilities of technologies, and the elastic nature of cloud services has meant that new opportunities have been presented to evolve the data warehouse to handle modern workloads.

Which of the following are examples of these opportunities? (Select all that apply)

- **Static data velocities**

- **Advanced analytics for all users**

  (Correct)

- **Insights through analytical dashboards**

(Correct)

- ☐

  Increased flexibility for data volumes

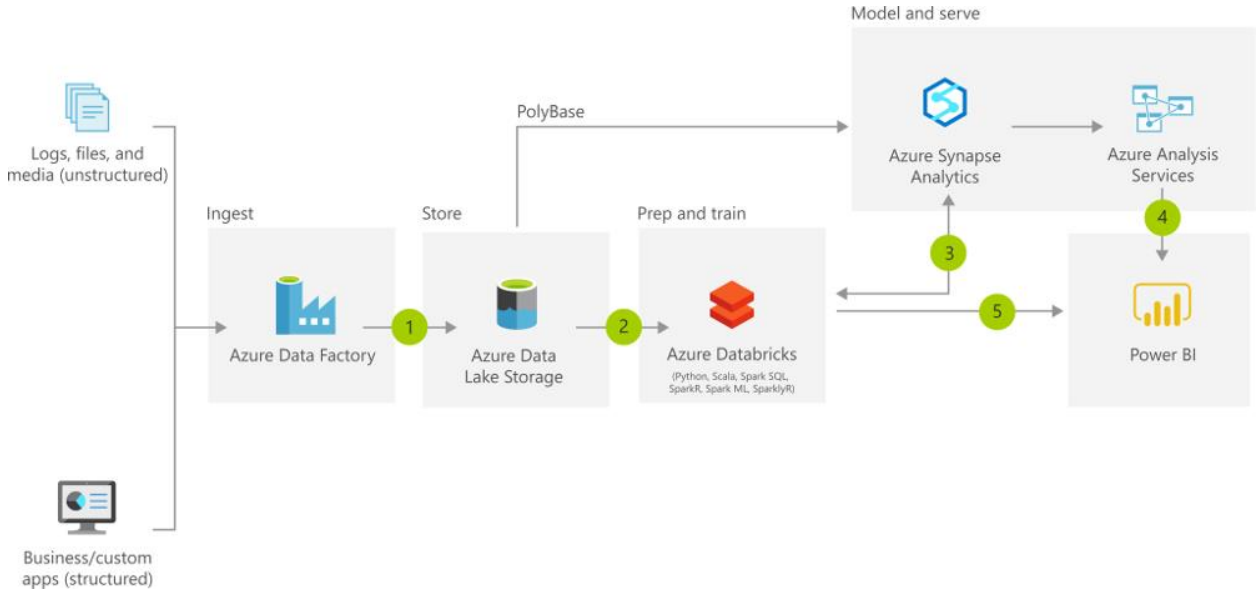
  (Correct)

- ☑

  New varieties of data

  (Correct)

Explanation
A modern data warehouse lets you bring together all your data at any scale easily, and means you can get insights through analytical dashboards, operational reports, or advanced analytics for all your users.



The pace of change in both the capabilities of technologies, and the elastic nature of cloud services has meant that new opportunities have been presented to evolve the data warehouse to handle modern workloads including:

Increased volumes of data

Microsoft Azure services have the capability to scale its capacity to meet the demands that an organization faces as its data grows. In traditional on-premises data, scaling on-premises servers is a non-trivial task that involves costs, procurement of additional hardware, as well as potential disruption to the business to meet the demand. With Azure, services such as Azure Synapse Analytics can be scaled at the click of a button, and can even be auto-scaled.

Staging data is also simplified using Azure Data Lake Store Gen2, which can store a wide variety of data in its raw format, making the process of ingesting data into a data warehouse much easier.

New varieties of data

Traditional data warehouse in the past have had difficulty in handling certain types of data. For example, extrapolating data from sources such as PDF files through to sound files where either too complex or cost prohibitive. The improvements in AI technologies such as Form Recognizer and Speech to Text Cognitive Services means that these types of data sources can now be passed through a cognitive service and outputted in a text-based format that can be stored in the Azure Data Lake Store Gen2, along with the source files themselves.

Data velocities

Traditional on-premises data warehouses in the main have dealt with the batch movement of data based on a schedule. Some organization may build real-time data warehouse if the business need is compelling and the organization can absorb the cost of the implementation. Azure has made it easier and much more cost effective to provision streaming services that can interact with a wide variety of services so that a modern data warehouse can deliver solutions in a batch or a real-time manner without the obstruction of cost.

https://docs.microsoft.com/en-us/azure/architecture/solution-ideas/articles/modern-data-warehouse

Question 106: **Incorrect**
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.

A storage account defines a policy that applies to all the storage services in the account.

Which are the settings that are controlled by a storage account. (Select all that apply)

- ☐

  Location

  (Correct)

- ☐

  Subscription

  (Correct)

- ☐

  Access tier

  (Correct)

- ☐

  Virtual networks

  (Correct)

- ☑

  Replication

  (Correct)

- ☐

  Secure transfer required

  (Correct)

- ☐

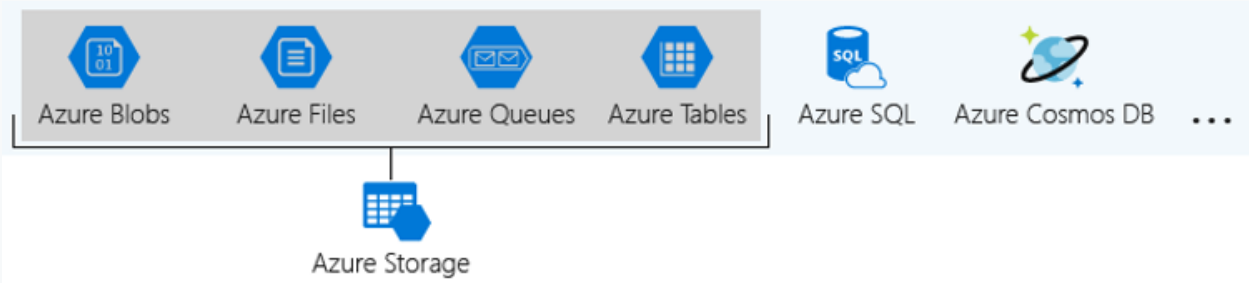
  Performance

  (Correct)

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.

Storage account settings

A storage account defines a policy that applies to all the storage services in the account. For example, you could specify that all the contained services will be stored in the West US datacentre, accessible only over https, and billed to the sales department's subscription.

The settings that are controlled by a storage account are:

• **Subscription**: The Azure subscription that will be billed for the services in the account.

• **Location**: The datacentre that will store the services in the account.

• **Performance**: Determines the data services you can have in your storage account and the type of hardware disks used to store the data. **Standard** allows you to have any data service (Blob, File, Queue, Table) and uses magnetic disk drives. **Premium** introduces additional services for storing data. For example, storing unstructured object data as block blobs or append blobs, and specialized file storage used to store and create premium file shares. These storage accounts use solid-state drives (SSD) for storage.

• **Replication**: Determines the strategy used to make copies of your data to protect against hardware failure or natural disaster. At a minimum, Azure will automatically maintain three copies of your data within the data centre associated with the storage account. This is called locally-redundant storage (LRS), and guards against hardware failure but does not protect you from an event that incapacitates the entire datacentre. You can upgrade to one of the other options such as geo-redundant storage (GRS) to get replication at different datacentres across the world.

• **Access tier**: Controls how quickly you will be able to access the blobs in this storage account. Hot gives quicker access than Cool, but at increased cost. This applies only to blobs, and serves as the default value for new blobs.

• **Secure transfer required**: A security feature that determines the supported protocols for access. Enabled requires HTTPs, while disabled allows HTTP.

• **Virtual networks**: A security feature that allows inbound access requests only from the virtual network(s) you specify.

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

Question 107: **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 working with materialized views.

Which of the following statements is a benefit of materialized views?

- **Increased resiliency benefits.**
- **Increased high availability.**
- **Material views isolate the queries for better action and result visibility.**
- **Reducing the execution time for complex queries with JOINs and aggregate functions.**

(Correct)

Explanation
Materialized views help to improve complex query performance. The more complex the query, the higher the potential for execution-time saving.

Materialized views in the Azure Synapse SQL pool provide a low maintenance method for complex analytical queries to quickly perform without any query change. SQL pool in Azure Synapse supports standard and materialized views. Both are virtual tables created with `SELECT` expressions and presented to queries as logical tables. Views encapsulate the complexity of common data computation and add an abstraction layer to computation changes, so there's no need to rewrite queries.

Comparison	View	Materialized view
View definition	Stored in SQL pool.	Stored in SQL pool.
View content	Generated each time when the view is used.	Pre-processed and stored in SQL pool during view creation. Updated as data is added to the underlying tables.
Data refresh	Always updated	Always updated
Speed to retrieve view data from complex queries	Slow	Fast
Extra storage	No	Yes
Syntax	CREATE VIEW	CREATE MATERIALIZED VIEW AS SELECT

A materialized view pre-computes, stores, and maintains its data in SQL pool just like a table. There's no recomputation needed each time when a materialized view is used. That's why queries that use all or a subset of the data in materialized views can get faster performance. Even better, queries can use a materialized view without directly referencing it, so there's no need to change the application code.

An adequately designed materialized view provides the following benefits:

• Reduce the execution time for complex queries with JOINs and aggregate functions. The more complex the query, the higher the potential for execution-time saving. The most benefit is gained when a query's computation cost is high, and the resulting data set is small.

• The optimizer in the SQL pool can automatically use deployed materialized views to improve query execution plans. This process is transparent to users providing faster query performance, and doesn't require queries to make direct reference to the materialized views.

•Require low maintenance on the views. All incremental data changes from the base tables are automatically added to the materialized views synchronously. This design allows querying materialized views to return the same data as directly querying the base tables.

• The data in a materialized view can be distributed differently from the base tables.

• Data in materialized views gets the same high availability and resiliency benefits as data in regular tables.

Comparing to other data warehouse providers, the materialized views implemented in Azure Synapse Analytics also provide the following additional benefits:

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

• Broad aggregate function support. See CREATE MATERIALIZED VIEW AS SELECT (Transact-SQL).

• The support for query-specific materialized view recommendation. See EXPLAIN (Transact-SQL).

Before you begin to create materialized views, it's essential to have a deep understanding of your workload in terms of query patterns, importance, frequency, and resulting data size. Users can run `EXPLAIN WITH_RECOMMENDATIONS <SQL_statement>` to get the materialized views recommended by the query optimizer. Since these recommendations are query-specific, a materialized view that benefits a single query may not be optimal for other queries in the same workload.

https://docs.microsoft.com/en-us/azure/synapse-analytics/sql/develop-materialized-view-performance-tuning

Question 108: **Incorrect**
Identify the missing word(s) in the following sentence within the context of Microsoft Azure.

A(n) [?] is one that appropriately manages the change of dimension members over time. It applies when the values of a business entity change over time, and not on a set schedule.

- ○

  Rapid changing dimension (RCD)

- ○

  Slowly changing dimension (SCD)

  (Correct)

- ○

  Azure Cosmos Historical Timeline (ACHT)

- ○

  Azure Datatrail

- ◉

  Azure Change Monitor (ACM)

  (Incorrect)

Explanation

*A **slowly changing dimension** (SCD) is one that appropriately manages the change of dimension members over time. It applies when the values of a business entity change over time, and not on a set schedule.* A good example of an SCD is a customer dimension, specifically its contact detail columns like email address and phone number. In contrast, some dimensions are considered to be rapidly changing when a dimension attribute changes often, like a stock's market price. The common design approach in these instances is to store rapidly changing attribute values in a fact table measure. However, for slowly changing dimensions the changing members should remain in the dimension table. The design used for handling changes depends on the business requirements for the dimension table.

Assume you are designing a customer dimension table as part of a star schema model for tracking sales transactions. This table would include attributes like company name and email address. If the company name or phone number changes, do you want to treat that as a new customer? Probably not. Instead, you need a way to update the existing record or to keep track of all historical changes to the values.

When handling data changes, the table design varies depending on if you choose to update values without history or to track each version of history. You need to identify if

your dimension will be a slowly changing dimension (SCD). In other words, will you allow updates to records? When the answer is yes you should add additional date fields to your table. These date fields will help you track when the data was updated in your analytics database.

A customer SCD will have a set of columns that can change (1) and a set of field to track when the record was added and modified (2).

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 109: **Incorrect**

Scenario: You are working as a consultant at Avengers Security. At the moment, you are consulting with Tony, the lead of the IT team and the topic of discussion is about a table in an enterprise data warehouse in Azure Synapse Analytics. See the subject table details below.

Table Characteristics:

• The file name is `SalesHistory`.

- The table contains sales data from the past 36 months.

- The file is partitioned by month.

- The file contains 1.5 billion rows.

- The file has clustered columnstore indexes.

Required:

At the beginning of each month, data older than 36 months must be promptly removed from the `SalesHistory` file.

The IT team has created a list of possible actions that should be performed to create a stored procedure, but there is debate about which are valid and the required sequence of the tabled actions. See the proposed action list below.

Proposed Actions:

a. Create an empty table named `SalesHistory_Current` that has a duplicate schema as the `SalesHistory` table.

b. Drop the `SalesHistory_Current` table.

c. Copy the data to a new table by using `CREATE TABLE AS SELECT`.

d. `TRUNCATE` the partition containing the stale data.

e. Switch the partition containing the stale data from `SalesHistory` to `SalesHistory_Current`.

f. Execute a `DELETE` statement where the value in the Date column is more than 36 months ago.

As the Azure SME, Tony and the team look to you to select the correct actions and put them in order in preparation for creation of the stored procedure. Which of the below contains the correct items in the correct sequence for the required stored procedure?

- **a → e → b**

  (Correct)

- **c → f → b**
- **c → b**
- **e → f → c**

  (Incorrect)
- **a → c → e → d**

Explanation

Step 1: *Create an empty table named* `SalesHistory_Current` *that has a duplicate schema as the* `SalesHistory` *table.*

Step 2: *Switch the partition containing the stale data from* `SalesHistory` *to* `SalesHistory_Current`*.*

SQL Data Warehouse supports partition splitting, merging, and switching. To switch partitions between two tables, you must ensure that the partitions align on their respective boundaries and that the table definitions match.

Loading data into partitions with partition switching is a convenient way stage new data in a table that is not visible to users the switch in the new data.

Step 3: *Drop the* `SalesHistory_Current` *table.*

Partitioning tables in dedicated SQL pool

Table partitions enable you to divide your data into smaller groups of data. In most cases, table partitions are created on a date column. Partitioning is supported on all dedicated SQL pool table types; including clustered columnstore, clustered index, and heap. Partitioning is also supported on all distribution types, including both hash or round robin distributed.

Partitioning can benefit data maintenance and query performance. Whether it benefits both or just one is dependent on how data is loaded and whether the same column can be used for both purposes, since partitioning can only be done on one column.

Benefits to loads

The primary benefit of partitioning in dedicated SQL pool is to improve the efficiency and performance of loading data by use of partition deletion, switching and merging. In most cases data is partitioned on a date column that is closely tied to the order in which the data is loaded into the SQL pool. One of the greatest benefits of using partitions to maintain data is the avoidance of transaction logging. While simply inserting, updating, or deleting data can be the most straightforward approach, with a little thought and effort, using partitioning during your load process can substantially improve performance.

Partition switching can be used to quickly remove or replace a section of a table. For example, a sales fact table might contain just data for the past 36 months. At the end of every month, the oldest month of sales data is deleted from the table. This data could be deleted by using a delete statement to delete the data for the oldest month.

However, deleting a large amount of data row-by-row with a delete statement can take too much time, as well as create the risk of large transactions that take a long time to rollback if something goes wrong. A more optimal approach is to drop the oldest partition of data. Where deleting the individual rows could take hours, deleting an entire partition could take seconds.

Benefits to queries

Partitioning can also be used to improve query performance. A query that applies a filter to partitioned data can limit the scan to only the qualifying partitions. This method of filtering can avoid a full table scan and only scan a smaller subset of data. With the introduction of clustered columnstore indexes, the predicate elimination performance benefits are less beneficial, but in some cases there can be a benefit to queries.

For example, if the sales fact table is partitioned into 36 months using the sales date field, then queries that filter on the sale date can skip searching in partitions that don't match the filter.

Sizing partitions

While partitioning can be used to improve performance some scenarios, creating a table with **too many** partitions can hurt performance under some circumstances. These concerns are especially true for clustered columnstore tables.

For partitioning to be helpful, it is important to understand when to use partitioning and the number of partitions to create. There is no hard fast rule as to how many partitions are too many, it depends on your data and how many partitions you loading simultaneously. A successful partitioning scheme usually has tens to hundreds of partitions, not thousands.

When creating partitions on **clustered columnstore** tables, it is important to consider how many rows belong to each partition. For optimal compression and performance of clustered columnstore tables, a minimum of 1 million rows per distribution and partition is needed. Before partitions are created, dedicated SQL pool already divides each table into 60 distributed databases.

Any partitioning added to a table is in addition to the distributions created behind the scenes. Using this example, if the sales fact table contained 36 monthly partitions, and given that a dedicated SQL pool has 60 distributions, then the sales fact table should contain 60 million rows per month, or 2.1 billion rows when all months are populated. If a table contains fewer than the recommended minimum number of rows per partition, consider using fewer partitions in order to increase the number of rows per partition.

How to split a partition that contains data

The most efficient method to split a partition that already contains data is to use a `CTAS` statement. If the partitioned table is a clustered columnstore, then the table partition must be empty before it can be split.

The following example creates a partitioned columnstore table. It inserts one row into each partition:

```
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])
,   PARTITION   (   [OrderDateKey] RANGE RIGHT FOR VALUES
                    (20000101
                    )
                )
)
;

INSERT INTO dbo.FactInternetSales
VALUES (1,19990101,1,1,1,1,1,1);
INSERT INTO dbo.FactInternetSales
VALUES (1,20000101,1,1,1,1,1,1);
```

The following query finds the row count by using the `sys.partitions` catalogue view:

```
SQL
```

```
SELECT  QUOTENAME(s.[name])+'.'+QUOTENAME(t.[name]) as Table_name
,       i.[name] as Index_name
,       p.partition_number as Partition_nmbr
,       p.[rows] as Row_count
,       p.[data_compression_desc] as Data_Compression_desc
FROM    sys.partitions p
JOIN    sys.tables     t    ON    p.[object_id]   = t.[object_id]
JOIN    sys.schemas    s    ON    t.[schema_id]   = s.[schema_id]
JOIN    sys.indexes    i    ON    p.[object_id]   = i.[object_Id]
                            AND   p.[index_Id]    = i.[index_Id]
WHERE t.[name] = 'FactInternetSales'
;
```

The following split command receives an error message:

```
SQL
ALTER TABLE FactInternetSales SPLIT RANGE (20010101);
```

Msg 35346, Level 15, State 1, Line 44 `SPLIT` clause of `ALTER PARTITION` statement failed because the partition is not empty. Only empty partitions can be split in when a columnstore index exists on the table. Consider disabling the columnstore index before issuing the `ALTER PARTITION` statement, then rebuilding the columnstore index after `ALTER PARTITION` is complete.

However, you can use `CTAS` to create a new table to hold the data.

```
SQL
CREATE TABLE dbo.FactInternetSales_20000101
    WITH    (   DISTRIBUTION = HASH(ProductKey)
            ,   CLUSTERED COLUMNSTORE INDEX
            ,   PARTITION   (   [OrderDateKey] RANGE RIGHT FOR VALUES
                                (20000101
                                )
                            )
            )
AS
SELECT *
FROM    FactInternetSales
WHERE   1=2
;
```

As the partition boundaries are aligned, a switch is permitted. This will leave the source table with an empty partition that you can subsequently split.

```
SQL
ALTER TABLE FactInternetSales SWITCH PARTITION 2 TO
FactInternetSales_20000101 PARTITION 2;

ALTER TABLE FactInternetSales SPLIT RANGE (20010101);
```

All that is left is to align the data to the new partition boundaries using `CTAS`, and then switch the data back into the main table.

```
SQL
CREATE TABLE [dbo].[FactInternetSales_20000101_20010101]
    WITH    (   DISTRIBUTION = HASH([ProductKey])
            ,   CLUSTERED COLUMNSTORE INDEX
            ,   PARTITION   (   [OrderDateKey] RANGE RIGHT FOR VALUES
                                (20000101,20010101
```

```
                                )
                            )
            )
AS
SELECT  *
FROM    [dbo].[FactInternetSales_20000101]
WHERE   [OrderDateKey] >= 20000101
AND     [OrderDateKey] <  20010101
;

ALTER TABLE dbo.FactInternetSales_20000101_20010101 SWITCH PARTITION 2 TO dbo.FactInternetSales PARTITION 2;
```

Once you have completed the movement of the data, it is a good idea to refresh the statistics on the target table. Updating statistics ensures the statistics accurately reflect the new distribution of the data in their respective partitions.

```
SQL
UPDATE STATISTICS [dbo].[FactInternetSales];
```

Load new data into partitions that contain data in one step

Loading data into partitions with partition switching is a convenient way to stage new data in a table that is not visible to users. It can be challenging on busy systems to deal with the locking contention associated with partition switching.

To clear out the existing data in a partition, an `ALTER TABLE` used to be required to switch out the data. Then another `ALTER TABLE` was required to switch in the new data.

In dedicated SQL pool, the `TRUNCATE_TARGET` option is supported in the `ALTER TABLE` command. With `TRUNCATE_TARGET` the `ALTER TABLE` command overwrites existing data in the partition with new data. Below is an example that uses `CTAS` to create a new table with the existing data, inserts new data, then switches all the data back into the target table, overwriting the existing data.

```
SQL
CREATE TABLE [dbo].[FactInternetSales_NewSales]
    WITH    (   DISTRIBUTION = HASH([ProductKey])
            ,   CLUSTERED COLUMNSTORE INDEX
            ,   PARTITION   (   [OrderDateKey] RANGE RIGHT FOR VALUES
                                (20000101,20010101
                                )
                            )
            )
AS
SELECT  *
FROM    [dbo].[FactInternetSales]
WHERE   [OrderDateKey] >= 20000101
AND     [OrderDateKey] <  20010101
;

INSERT INTO dbo.FactInternetSales_NewSales
VALUES (1,20000101,2,2,2,2,2,2);

```

```
ALTER TABLE dbo.FactInternetSales_NewSales SWITCH PARTITION 2 TO
dbo.FactInternetSales PARTITION 2 WITH (TRUNCATE_TARGET = ON);
```

Table partitioning source control

To avoid your table definition from **rusting** in your source control system, you may want to consider the following approach:

1. Create the table as a partitioned table but with no partition values

```
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])
,   PARTITION   (   [OrderDateKey] RANGE RIGHT FOR VALUES () )
)
;
```

2. `SPLIT` the table as part of the deployment process:

```
SQL
-- Create a table containing the partition boundaries

CREATE TABLE #partitions
WITH
(
    LOCATION = USER_DB
,   DISTRIBUTION = HASH(ptn_no)
)
AS
SELECT  ptn_no
,       ROW_NUMBER() OVER (ORDER BY (ptn_no)) as seq_no
FROM    (
    SELECT CAST(20000101 AS INT) ptn_no
    UNION ALL
    SELECT CAST(20010101 AS INT)
    UNION ALL
    SELECT CAST(20020101 AS INT)
    UNION ALL
    SELECT CAST(20030101 AS INT)
    UNION ALL
    SELECT CAST(20040101 AS INT)
) a
;

```

```
 -- Iterate over the partition boundaries and split the table

DECLARE @c INT = (SELECT COUNT(*) FROM #partitions)
,       @i INT = 1                                  --iterator for while loop
,       @q NVARCHAR(4000)                           --query
,       @p NVARCHAR(20)     = N''                   --partition_number
,       @s NVARCHAR(128)    = N'dbo'                --schema
,       @t NVARCHAR(128)    = N'FactInternetSales' --table
;

WHILE @i <= @c
BEGIN
    SET @p = (SELECT ptn_no FROM #partitions WHERE seq_no = @i);
    SET @q = (SELECT N'ALTER TABLE '+@s+N'.'+@t+N' SPLIT RANGE ('+@p+N');');

    -- PRINT @q;
    EXECUTE sp_executesql @q;
    SET @i+=1;
END

 -- Code clean-up

DROP TABLE #partitions;
```

With this approach, the code in source control remains static and the partitioning boundary values are allowed to be dynamic; evolving with the SQL pool over time.

https://docs.microsoft.com/en-us/azure/sql-data-warehouse/sql-data-warehouse-tables-partition

Question 110: **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 is concerned if the data his IT team is working with is unevenly allocated across all distributions.

Which of the following would be the best approach to investigate if the data at hand is unevenly allocated across all distributions?

- **Grouping the data based on partitions and counting rows with a T-SQL query.**

- **Using `DBCC PDW_SHOWSPACEUSED` to see the number of table rows that are stored in each of the 60 distributions.**

  (Correct)

- **Use Power BI to create a visualization to determine the distribution pattern.**

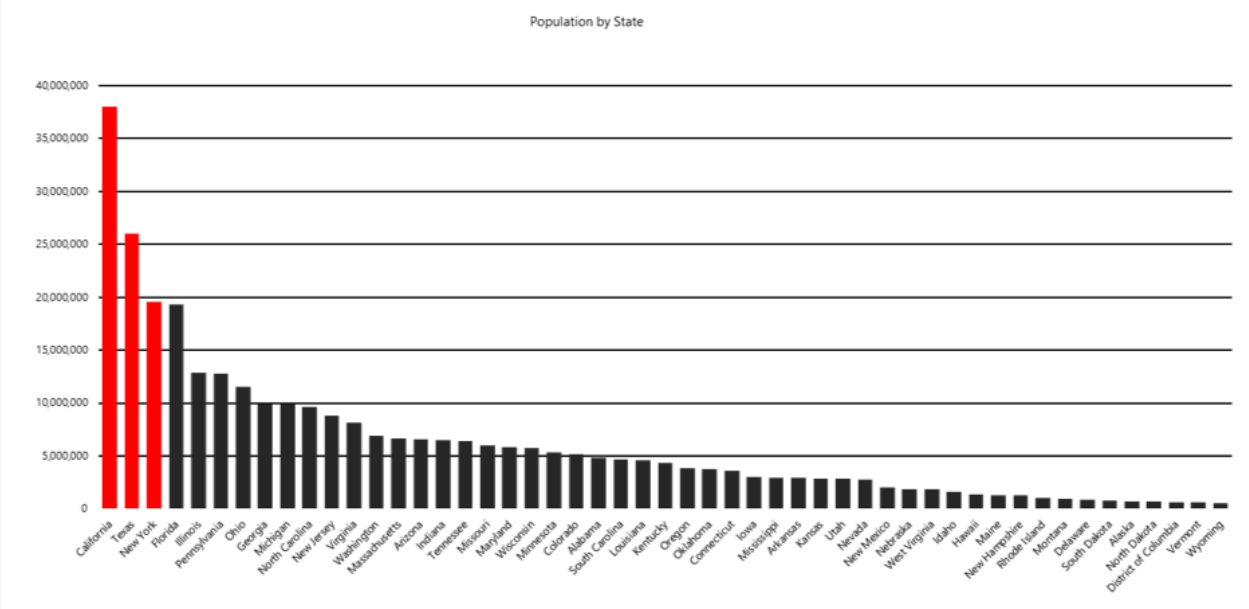
- **Monitor query speeds by testing the same query for each partition.**

  (Incorrect)

Explanation

`DBCC PDW_SHOWSPACEUSED` *returns the number of table rows that are stored in each of the 60 distributions.*

In simple terms, data skew is an over-represented value. Imagine that you have assigned 50 tax examiners to audit tax returns, one examiner for each US state. The Wyoming examiner, because the population there is small, has little to do. In California, however, the examiner is kept very busy because of the state's large population.



The data is unevenly distributed across all tax examiners, which means that some examiners must work more than others. In your job, you frequently experience situations like the tax-examiner example here. In more technical terms, one vertex gets much more data than its peers, a condition that makes the vertex work more than the others and that eventually slows down an entire job. What's worse, the job might fail

because vertices might have, for example, a 5-hour runtime limitation and a 6-GB memory limitation.

A quick way to check for data skew is to use DBCC PDW_SHOWSPACEUSED. The following SQL code returns the number of table rows that are stored in each of the 60 distributions. For balanced performance, the rows in your distributed table should be spread evenly across all the distributions.

```
1. SQL
2. -- Find data skew for a distributed table
3. DBCC PDW_SHOWSPACEUSED('dbo.FactInternetSales');
```

Another aspect of data storage in Azure Synapse dedicated SQL pools is to monitor the table data space usage and observe its relationship with different table distribution types. Additionally, it is helpful to know the number of rows and the storage space used for indexing. Below is a list of System Dynamic Management Views (DMVs) that you can use to dig for the information. During the next exercise you will create a view using these DMVs to get a better view of the data.

Table Name	Description
sys.schemas	All schemas in the database.
sys.tables	All tables in the database.
sys.indexes	All indexes in the database.
sys.columns	All columns in the database.
sys.pdw_table_mappings	Maps each table to local tables on physical nodes and distributions.
sys.pdw_nodes_tables	Contains information on each local table in each distribution.
sys.pdw_table_distribution_properties	Holds distribution information for tables (the type of distribution tables have).
sys.pdw_column_distribution_properties	Holds distribution information for columns. Filtered to include only columns used to distribute their parent tables (distribution_ordinal = 1).
sys.pdw_distributions	Holds information about the distributions from the SQL pool.
sys.dm_pdw_nodes	Holds information about the nodes from the SQL pool. Filtered to include only compute nodes (type = COMPUTE).
sys.dm_pdw_nodes_db_partition_stats	Returns page and row-count information for every partition in the current database.

Since a columnstore index scans a table by scanning column segments of individual rowgroups, maximizing the number of rows in each rowgroup enhances query performance. When rowgroups have a high number of rows, data compression improves which means there is less data to read from disk. The number of rows in a rowgroup determines the rowgroup's quality.

For best query performance, the goal is to maximize the number of rows per rowgroup in a columnstore index. A rowgroup can have a maximum of 1,048,576 rows. It's okay to not have the maximum number of rows per rowgroup. Columnstore indexes achieve good performance when rowgroups have at least 100,000 rows.

During a bulk load or columnstore index rebuild, sometimes there isn't enough memory available to compress all the rows designated for each rowgroup. When memory pressure is present, columnstore indexes trim the rowgroup sizes so compression into the columnstore can succeed.

The DMV sys.dm_pdw_nodes_db_column_store_row_group_physical_stats exposes useful information such as number of rows in rowgroups and the reason for trimming, if there was trimming.

There are two columns from dm_pdw_nodes_db_column_store_row_group_physical_stats worth looking into in detail. The state_desc column provides useful information on the state of a row group:

Name	Description
INVISIBLE	A rowgroup which is being compressed.
OPEN	A deltastore rowgroup that is accepting new rows. It is important to remember that an open rowgroup is still in rowstore format and has not been compressed to columnstore format.
CLOSED	A deltastore rowgroup that contains the maximum number of rows, and is waiting for the tuple mover process to compress it to the columnstore.
COMPRESSED	A row group that is compressed with columnstore compression and stored in the columnstore.
TOMBSTONE	A row group that was formerly in the deltastore and is no longer used.

Additionally, the trim_reason_desc column describes the reason that triggered the COMPRESSED rowgroup to have less than the maximum number of rows:

Name	Description
UNKNOWN_UPGRADED_FROM_PREVIOUS_VERSION	Occurred when upgrading from the previous version of SQL Server.
NO_TRIM	The row group was not trimmed. The row group was compressed with the maximum of 1,048,476 rows. The number of rows could be less if a subset of rows was deleted after delta rowgroup was closed.
BULKLOAD	The bulk-load batch size limited the number of rows. This is what you should be looking for when optimizing data loading, as it is an indicator of resource starvation during the loading process.
REORG	Forced compression as part of REORG command.
DICTIONARY_SIZE	Dictionary size grew too large to compress all of the rows together.
MEMORY_LIMITATION	Not enough available memory to compress all the rows together.
RESIDUAL_ROW_GROUP	Closed as part of last row group with rows < 1 million during index build operation.

https://docs.microsoft.com/en-us/azure/architecture/guide/technology-choices/data-store-overview