

Exam DP-203: Data Engineering on Microsoft Azure – Skills Measured

Audience Profile

Candidates for this exam should have subject matter expertise integrating, transforming, and consolidating data from various structured and unstructured data systems into a structure that is suitable for building analytics solutions.

Azure Data Engineers help stakeholders understand the data through exploration, and they build and maintain secure and compliant data processing pipelines by using different tools and techniques. These professionals use various Azure data services and languages to store and produce cleansed and enhanced datasets for analysis.

Azure Data Engineers also help ensure that data pipelines and data stores are high-performing, efficient, organized, and reliable, given a set of business requirements and constraints. They deal with unanticipated issues swiftly, and they minimize data loss. They also design, implement, monitor, and optimize data platforms to meet the data pipelines needs.

A candidate for this exam must have strong knowledge of data processing languages such as SQL, Python, or Scala, and they need to understand parallel processing and data architecture patterns.

Skills Measured

NOTE: The bullets that follow each of the skills measured are intended to illustrate how we assess that skill. This list is not definitive or exhaustive.

NOTE: Most questions cover features that are General Availability (GA). The exam may contain questions on Preview features, if those features are commonly used.

Design and Implement Data Storage (40-45%)

Design a data storage structure

- design an Azure Data Lake solution
- recommend file types for storage
- recommend file types for analytical queries
- design for efficient querying
- design for data pruning
- design a folder structure that represents the levels of data transformation
- design a distribution strategy

- design a data archiving solution

Design a partition strategy

- design a partition strategy for files
- design a partition strategy for analytical workloads
- design a partition strategy for efficiency/performance
- design a partition strategy for Azure Synapse Analytics
- identify when partitioning is needed in Azure Data Lake Storage Gen2

Design the serving layer

- design star schemas
- design slowly changing dimensions
- design a dimensional hierarchy
- design a solution for temporal data
- design for incremental loading
- design analytical stores
- design metastores in Azure Synapse Analytics and Azure Databricks

Implement physical data storage structures

- implement compression
- implement partitioning
- implement sharding
- implement different table geometries with Azure Synapse Analytics pools
- implement data redundancy
- implement distributions
- implement data archiving

Implement logical data structures

- build a temporal data solution
- build a slowly changing dimension
- build a logical folder structure
- build external tables
- implement file and folder structures for efficient querying and data pruning

Implement the serving layer

- deliver data in a relational star schema
- deliver data in Parquet files
- maintain metadata
- implement a dimensional hierarchy

Design and Develop Data Processing (25-30%)

Ingest and transform data

- transform data by using Apache Spark
- transform data by using Transact-SQL
- transform data by using Data Factory
- transform data by using Azure Synapse Pipelines
- transform data by using Stream Analytics
- cleanse data
- split data
- shred JSON
- encode and decode data
- configure error handling for the transformation
- normalize and denormalize values
- transform data by using Scala
- perform data exploratory analysis

Design and develop a batch processing solution

- develop batch processing solutions by using Data Factory, Data Lake, Spark, Azure Synapse Pipelines, PolyBase, and Azure Databricks
- create data pipelines
- design and implement incremental data loads
- design and develop slowly changing dimensions
- handle security and compliance requirements
- scale resources
- configure the batch size
- design and create tests for data pipelines
- integrate Jupyter/IPython notebooks into a data pipeline
- handle duplicate data
- handle missing data
- handle late-arriving data
- upsert data
- regress to a previous state
- design and configure exception handling
- configure batch retention
- design a batch processing solution
- debug Spark jobs by using the Spark UI

Design and develop a stream processing solution

- develop a stream processing solution by using Stream Analytics, Azure Databricks, and Azure Event Hubs

- process data by using Spark structured streaming
- monitor for performance and functional regressions
- design and create windowed aggregates
- handle schema drift
- process time series data
- process across partitions
- process within one partition
- configure checkpoints/watermarking during processing
- scale resources
- design and create tests for data pipelines
- optimize pipelines for analytical or transactional purposes
- handle interruptions
- design and configure exception handling
- upsert data
- replay archived stream data
- design a stream processing solution

Manage batches and pipelines

- trigger batches
- handle failed batch loads
- validate batch loads
- manage data pipelines in Data Factory/Synapse Pipelines
- schedule data pipelines in Data Factory/Synapse Pipelines
- implement version control for pipeline artifacts
- manage Spark jobs in a pipeline

Design and Implement Data Security (10-15%)

Design security for data policies and standards

- design data encryption for data at rest and in transit
- design a data auditing strategy
- design a data masking strategy
- design for data privacy
- design a data retention policy
- design to purge data based on business requirements
- design Azure role-based access control (Azure RBAC) and POSIX-like Access Control List (ACL) for Data Lake Storage Gen2
- design row-level and column-level security

Implement data security

- implement data masking

- encrypt data at rest and in motion
- implement row-level and column-level security
- implement Azure RBAC
- implement POSIX-like ACLs for Data Lake Storage Gen2
- implement a data retention policy
- implement a data auditing strategy
- manage identities, keys, and secrets across different data platform technologies
- implement secure endpoints (private and public)
- implement resource tokens in Azure Databricks
- load a DataFrame with sensitive information
- write encrypted data to tables or Parquet files
- manage sensitive information

Monitor and Optimize Data Storage and Data Processing (10-15%)

Monitor data storage and data processing

- implement logging used by Azure Monitor
- configure monitoring services
- measure performance of data movement
- monitor and update statistics about data across a system
- monitor data pipeline performance
- measure query performance
- monitor cluster performance
- understand custom logging options
- schedule and monitor pipeline tests
- interpret Azure Monitor metrics and logs
- interpret a Spark directed acyclic graph (DAG)

Optimize and troubleshoot data storage and data processing

- compact small files
- rewrite user-defined functions (UDFs)
- handle skew in data
- handle data spill
- tune shuffle partitions
- find shuffling in a pipeline
- optimize resource management
- tune queries by using indexers
- tune queries by using cache
- optimize pipelines for analytical or transactional purposes
- optimize pipeline for descriptive versus analytical workloads
- troubleshoot a failed spark job
- troubleshoot a failed pipeline run