DP-203

Design and implement data storage (15–20%)

- Implement a partition strategy
    - Implement a partition strategy for files
    - Implement a partition strategy for analytical workloads
    - Implement a partition strategy for streaming workloads
    - Implement a partition strategy for Azure Synapse Analytics
    - Identify when partitioning is needed in Azure Data Lake Storage Gen2
- Design and implement the data exploration layer
    - Create and execute queries by using a compute solution that leverages SQL serverless and Spark cluster
    - Implement Azure Synapse Analytics database templates
    - Recommend Azure Synapse Analytics database templates
    - Push new or updated data lineage to Microsoft Purview
    - Browse and search metadata in Microsoft Purview Data Catalog

Develop data processing (40–45%)

- Ingest and transform data
    - Design and implement incremental loads
    - Transform data by using Apache Spark
    - Transform data by using Transact-SQL (T-SQL)
    - Ingest and transform data by using Azure Synapse Pipelines or Azure Data Factory
    - Transform data by using Azure Stream Analytics
    - Cleanse data
    - Handle duplicate data
    - Handle missing data
    - Handle late-arriving data
    - Split data
    - Shred JSON
    - Encode and decode data
    - Configure error handling for a transformation
    - Normalize and denormalize values
    - Perform data exploratory analysis
- Develop a batch processing solution
    - Develop batch processing solutions by using Azure Data Lake Storage, Azure Databricks, Azure Synapse Analytics, and Azure Data Factory

    - Use PolyBase to load data to a SQL pool
    - Implement Azure Synapse Link and query the replicated data
    - Create data pipelines
    - Scale resources
    - Configure the batch size
    - Create tests for data pipelines
    - Integrate Jupyter or Python notebooks into a data pipeline
    - Upsert data
    - Revert data to a previous state
    - Configure exception handling
    - Configure batch retention
    - Read from and write to a delta lake
- Develop a stream processing solution
    - Create a stream processing solution by using Stream Analytics and Azure Event Hubs
    - Process data by using Spark structured streaming
    - Create windowed aggregates
    - Handle schema drift
    - Process time series data
    - Process data across partitions
    - Process within one partition
    - Configure checkpoints and watermarking during processing
    - Scale resources
    - Create tests for data pipelines
    - Optimize pipelines for analytical or transactional purposes
    - Handle interruptions
    - Configure exception handling
    - Upsert data
    - Replay archived stream data
- Manage batches and pipelines
    - Trigger batches
    - Handle failed batch loads
    - Validate batch loads
    - Manage data pipelines in Azure Data Factory or Azure Synapse Pipelines
    - Schedule data pipelines in Data Factory or Azure Synapse Pipelines
    - Implement version control for pipeline artifacts
    - Manage Spark jobs in a pipeline

Secure, monitor, and optimize data storage and data processing (30–35%)

- Implement data security
  - Implement data masking
  - Encrypt data at rest and in motion
  - Implement row-level and column-level security
  - Implement Azure role-based access control (RBAC)
  - Implement POSIX-like access control lists (ACLs) for Data Lake Storage Gen2
  - Implement a data retention policy
  - 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 data storage and data processing
  - Implement logging used by Azure Monitor
  - Configure monitoring services
  - Monitor stream processing
  - Measure performance of data movement
  - Monitor and update statistics about data across a system
  - Monitor data pipeline performance
  - Measure query performance
  - Schedule and monitor pipeline tests
  - Interpret Azure Monitor metrics and logs
  - Implement a pipeline alert strategy
- Optimize and troubleshoot data storage and data processing
  - Compact small files
  - Handle skew in data
  - Handle data spill
  - Optimize resource management
  - Tune queries by using indexers
  - Tune queries by using cache
  - Troubleshoot a failed Spark job
  - Troubleshoot a failed pipeline run, including activities executed in external services