KaozhengPro

IT認證考試題庫

專業平臺

考證寶提供最新考古題與模擬試題
協助您高效通過認證考試

www.kaozhengpro.com

Exam : **DP-750**

Title : Implementing Data Engineering Solutions Using Azure Databricks

Version : DEMO

1. Set up and configure an Azure Databricks environment

Testlet 1
This is a case study. Case studies are not timed separately from other exam sections. You can use as much exam time as you would like to complete each case study. However, there might be additional case studies or other exam sections. Manage your time to ensure that you can complete all the exam sections in the time provided. Pay attention to the Exam Progress at the top of the screen so you have sufficient time to complete any exam sections that follow this case study.

To answer the case study questions, you will need to reference information that is provided in the case. Case studies and associated questions might contain exhibits or other resources that provide more information about the scenario described in the case. Information provided in an individual question does not apply to the other questions in the case study.

A Review Screen will appear at the end of this case study. From the Review Screen, you can review and change your answers before you move to the next exam section. After you leave this case study, you will NOT be able to return to it.

To start the case study
To display the first question in this case study, select the “Next” button. To the left of the question, a menu provides links to information such as business requirements, the existing environment, and problem statements. Please read through all this information before answering any questions. When you are ready to answer a question, select the “Question” button to return to the question.

Overview
Company Information
Contoso, Inc. is a renewable energy provider that operates solar and wind farms across North America.

Existing Environment
Azure Environment
Contoso has a single Azure Databricks workspace named Workspace1 in the West US Azure region.
Workspace1 is enabled for Unity Catalog.
Workspace1 contains all-purpose clusters for both development and production workloads.
The company's Azure environment contains:
- In the West US, Central US, and East US Azure regions, Azure event hubs that stream telemetry data and an Azure Data Lake Storage Gen2 account in each region for each hub
- A single Azure SQL database in the West US region that hosts enterprise resource planning (ERP) data
- An Azure Database for PostgreSQL server in the West US region that stores operational maintenance data

Data Environment
Contoso ingests the following operational and business data:
- Telemetry data: More than 40,000 IoT sensors across 28 sites emit JSON telemetry events every few seconds. Each site sends the events to the nearest event hub, which writes the data into the

corresponding Data Lake Storage Gen2 account. These files frequently experience schema drift.
- Maintenance logs: Maintenance systems generate historical repair logs, daily incremental updates, technician notes, and unstructured attachments that are stored in the Data Lake Storage Gen2 accounts.
- Operational maintenance data: Structured operational maintenance data is stored on the Azure Database for PostgreSQL server.
- External weather data: Hourly weather forecasts are retrieved from a REST API and written to the Data Lake Storage Gen2 accounts.
- ERP data: Daily CSV extracts of 50 to 100 GB contain equipment metadata, work orders, and purchase order information.

Problem Statements
The company's existing analytics environment has several issues:

Ingestion
- Telemetry pipelines fall behind during peak loads.
- Telemetry ingestion fails when schema drift occurs.
- Streaming pipelines reprocess events after a pipeline restarts.

Compute
- Production and development workloads run on the same all-purpose clusters.
- Production and development workloads do NOT support autoscaling or workload isolation.

Governance
- The ERP data is duplicated across systems and development teams.
- Naming conventions are inconsistent across development teams, regions, and products.
- Ownership of the IoT sensors changes over time, and analysts must track the full history of the ownership.
- Occasionally, equipment manufacturers must correct data-entry mistakes in equipment names. Historical values are NOT required.

Pipeline operations
- Pipelines lack resiliency, alerting, and centralized scheduling.

Requirements
Planned Changes
Contoso plans to implement the following changes:
- Implement scalable data pipeline orchestration.
- Create a managed analytics catalog in Unity Catalog.
- Implement a consistent approach to creating curated datasets.
- Establish a centralized governance model across ingestion, cleansed, and curated layers.
- Grant data engineers access to the ERP tables by using minimal development effort.
- Adopt a compute strategy that isolates production workloads and supports autoscaling.
- Adopt a slowly changing dimension (SCD) approach to address current data modeling issues.

Technical Requirements
Contoso identifies the following environment and compute requirements:
- Ensure that production ingestion workloads run on compute clusters that can scale automatically during telemetry spikes.
- Provide fast and consistent performance for business intelligence (BI) workloads.
- Prevent development activity from affecting production pipelines.
- Production ingestion workloads must run as scheduled, non-interactive pipelines rather than on shared interactive development clusters.

Contoso identifies the following data ingestion and processing requirements:
- Auto-scale ingestion pipelines to handle bursty workloads.
- Handle schema drift for the maintenance and telemetry data.
- Ingest file-based telemetry data by using minimal operational effort.
- Store all the ingested data in a format that supports incremental processing.
- Support the continuous ingestion of telemetry data from the event hubs by using exactly-once semantics.
- Support the ingestion of the structured maintenance data from the Azure Database for PostgreSQL server.
- Build a new telemetry pipeline that ingests raw events from the event hubs, cleanses the data, and publishes curated tables to Unity Catalog.
- Ensure that the Apache Spark Structured Streaming pipelines reading from the event hubs write the data into a managed Delta table named telemetry.raw_events. The pipelines must support schema drift and resume processing after failures without reprocessing the data.

Contoso identifies the following data modeling and optimization requirements:
- Build curated tables that standardize business logic.
- Overwrite equipment metadata attributes, such as name, manufacturer, model, and commissioning date, when the attributes change. Historical values are NOT required.

Contoso identifies the following pipeline deployment and operation requirements:
- Orchestrate multi-step ingestion and transformation workflows.
- Define a clear execution order and dependencies.
- Automatically retry failed steps and notify operators.
- Schedule ingestion and transformation workloads consistently.

Governance Requirements
Contoso identifies the following governance requirements:
- Centralize the metadata catalog.
- Provide isolated development areas that follow standard naming conventions.
- Establish a consistent structure for organizing raw, cleansed, and curated data.
- Provide a read-only mechanism to reference the ERP data through a foreign catalog.

Business Requirements
Contoso identifies the following business requirements:

- Improve ingestion reliability and reduce operational effort.
- Standardize data definitions across development teams.

You need to configure compute for the ingestion of telemetry data. The solution must meet the data ingestion and processing requirements.
What should you do?
A. Move the ingestion pipelines to shared compute.
B. Enable Photon acceleration for a job compute cluster.
C. Increase an all-purpose cluster to a larger fixed node type.
D. Disable autoscaling for a job compute cluster.
Answer: B
Explanation:
Enabling Photon acceleration on a job compute cluster is the best option. It significantly speeds up data engineering pipelines, handles JSON file ingestion seamlessly, and optimizes cost-efficiency for bursty production workloads, which are key for processing rapidly arriving IoT sensor events.
Enable Photon acceleration for a job compute cluster:
Databricks Auto Loader is designed for low-latency, high-volume file ingestion. Running Auto Loader on a job compute cluster provides the lowest operational cost because job clusters are billed at a much lower rate than all-purpose clusters.
Enabling Photon acceleration heavily optimizes the ingestion, parsing, and processing of raw JSON data strings, allowing the cluster to handle bursty workloads faster and auto-scale more efficiently.
Scenario, Data Environment, Contoso ingests the following operational and business data: Telemetry data: More than 40,000 IoT sensors across 28 sites emit JSON telemetry events every few seconds. Each site sends the events to the nearest event hub, which writes the data into the corresponding Data Lake Storage Gen2 account. These files frequently experience schema drift.
Contoso identifies the following data ingestion and processing requirements:
-> Auto-scale ingestion pipelines to handle bursty workloads.
Handle schema drift for the maintenance and telemetry data.
-> Ingest file-based telemetry data by using minimal operational effort.
Reference: https://medium.com/@krthiak/10-days-of-data-engineering-interview-qna-day-5-db1c0b58bf86

2. Set up and configure an Azure Databricks environment
Question Set 2

You have an Azure Databricks workspace.
You are creating a Lakeflow Spark Declarative Pipelines (SDP) pipeline that scales automatically.
You need to configure compute for the pipeline. The solution must minimize operational costs and effort.
What should you use?
A. the existing SQL warehouse
B. an all-purpose cluster that uses autoscaling
C. a job cluster that uses autoscaling
D. a single-node, all-purpose cluster
Answer: C

Explanation:
The best option for a Lakeflow Spark Declarative Pipelines (SDP) pipeline that scales automatically while keeping costs and administrative effort low is a job cluster that uses autoscaling.
Lowest Costs: Job clusters (also called automated compute) are billed at a significantly lower Data Processing Unit (DBU) rate compared to all-purpose clusters. By enabling autoscaling, Databricks dynamically allocates or removes worker nodes based on real-time pipeline demand, ensuring you never pay for unutilized resources.
Low Administrative Effort: While Databricks generally recommends Serverless compute as the absolute ideal for zero-admin pipelines, when selecting from classic compute options, a job cluster automatically handles its own lifecycle. It deploys when the pipeline starts executing and terminates automatically when processing is finished.
Incorrect:
[Not A]
Databricks SQL warehouses are designed to run standalone materialized views and streaming tables via standard SQL. They are not the native compute vehicle for running a fully automated, dedicated Lakeflow Spark Declarative Pipelines (SDP) deployment framework.
[Not B]
All-purpose compute is meant for interactive development, debugging, and ad-hoc analysis. It is billed at a much higher DBU rate, which violates the requirement to keep costs low.
[Not D]
Aside from the higher billing rate of all-purpose compute, a single-node configuration does not scale horizontally. This directly conflicts with your requirement to build a pipeline that scales automatically.
Reference: https://docs.databricks.com/gcp/en/ldp/auto-scaling

3.DRAG DROP
You have an Azure Databricks workspace that contains an all-purpose compute cluster named Cluster1. Cluser1 is used for interactive development.
You need to configure Cluster1 to meet the following requirements:
- Automatically add and remove worker nodes based on workload demand.
- Automatically shut down when the cluster has been idle for a specific period.
What should you configure for each requirement? To answer, drag the appropriate options to the correct requirements. Each option may be used once, more than once, or not at all. You may need to drag the split bar between panes or scroll to view content. NOTE: Each correct selection is worth one point.
Select and Place:

Options

- Auto termination
- Autoscaling
- Max workers
- Min workers
- Photon acceleration
- Spot instances
- The driver node type

Answer Area

Automatically add and remove worker nodes: Option

Automatically shut down: Option

Answer:

Options

- Auto termination
- Autoscaling
- Max workers
- Min workers
- Photon acceleration
- Spot instances
- The driver node type

Answer Area

Automatically add and remove worker nodes: Autoscaling

Automatically shut down: Auto termination

Explanation:

Box 1: Autoscaling

To dynamically scale your worker nodes based on workload demand, you must enable and configure Autoscaling on the cluster.

Cluster Type: Select All-Purpose during creation.

*-> Autoscaling Toggle: Check the Enable autoscaling checkbox.

Minimum Workers: Define the lowest number of nodes to keep active.

Maximum Workers: Define the highest number of nodes allowed to launch.

Box 2: Auto Termination

To configure an Azure Databricks all-purpose compute cluster to shut down automatically when idle, you must enable and configure the Auto Termination feature.

Default Value: Azure Databricks typically sets a default auto-termination period of 120 minutes.

Minimum Value: You can set the idle timeout to a minimum of 10 minutes.

Activity Triggers: The idle timer resets whenever commands are run, or when Spark jobs, notebook executions, or API calls interact with the cluster.

Cost Management: Setting a lower threshold (e.g., 20–30 minutes) prevents unnecessary costs from clusters left running overnight or during breaks.

Reference: https://maqsoftware.com/insights/azure-databricks-best-practices.html

4.You have an Azure Databricks workspace that is attached to a Unity Catalog metastore named metastore1, metastore1 contains a catalog named catalog1.
You need to create a new schema named schema2 that meets the following requirements:
- Is contained in catalog1
- Uses abfss://container@storageaccount.dfs.core.windows.net/data as the managed location
Which SQL statement should you execute?
A. CREATE SCHEMA catalog1.schema2
LOCATION 'abfss://container@storageaccount.dfs.core.windows.net/data';
B. CREATE SCHEMA catalog1.schema2
MANAGED LOCATION 'abfss://container@storageaccount.dfs.core.windows.net/ data';
C. CREATE CATALOG schema2
MANAGED LOCATION 'abfss://container@storageaccount.dfs.core.windows.net/ data';
D. CREATE SCHEMA catalog1.schema2
WITH DBPROPERTIES (LOCATION-'abfss:// container@storageaccount.dfs.core.windows.net/data');
Answer: B
Explanation:
To create a new schema with a specific managed location in Azure Databricks Unity Catalog, use the CREATE SCHEMA command.
CREATE SCHEMA catalog_name.schema_name
MANAGED LOCATION 'abfss://container@storageaccount.dfs.core.windows.net/data';
Catalog Name: Replace catalog_name with your existing catalog.
Schema Name: Replace schema_name with your desired new schema name.Quotes: The storage path must be enclosed in single quotes.
Privileges: You must have CREATE SCHEMA privileges on the parent catalog and ownership (or adequate permissions) on the external location.
Reference: https://dev.to/encorepartners/creating-your-first-catalog-schema-and-tables-in-databricks-20p3

5.You have an Azure Databricks workspace named Workspace1.
You create a compute cluster named Cluster1 that will be used to ingest data.
You need to install the required libraries on Cluster1. The solution must use Unity Catalog for access control.
What should you do?
A. Install the libraries by using pip3.
B. Create a custom dependency management script and run the script from a Databricks notebook.
C. Upload the libraries to Workspace1 and install the libraries on Cluster1.
D. Install the libraries on Cluster1 and manually restart the cluster.
Answer: C
Explanation:
The best action is uploading the libraries to the workspace and installing the libraries on the cluster (or ideally uploading them to Unity Catalog volumes).
Unity Catalog Compatibility: When using Unity Catalog for access control, compute clusters are typically configured with Standard (Shared) access mode. In this mode, traditional cluster init scripts [Not B.] face strict execution restrictions or are completely blocked to maintain secure user isolation.

Governance: Uploading your packages as Workspace Files or to Unity Catalog volumes allows administrators to manage access permissions directly and add them to an allowlist if needed.
Cluster-Wide Availability: Installing the libraries via the cluster's Libraries tab ensures that the required ingestion packages are automatically pre-installed and available across all nodes and notebooks running on that cluster.
Incorrect:
[Not A]
Running pip3 install manually on a cluster terminal or inside a notebook only applies to the specific notebook session (notebook-scoped). It does not natively persist across cluster restarts or handle cross-node execution effectively for data ingestion pipelines.
[Not B]
Running a custom script or a legacy init script to modify system-level paths introduces security risks and is generally incompatible with Unity Catalog's strict execution isolation policies for shared compute.
Reference: https://docs.databricks.com/aws/en/libraries/