Databricks-Certified-Professional-Data-Engineer Practice Questions

Shared by Lopez on 18-08-2026

For More Free Questions and Preparation Resources

Check the Links on Last Page

Question 1

Question Type: MultipleChoice

A data engineer needs to install the PyYAML Python package for YAML file processing within their Databricks environment. However, the Databricks workspace is air-gapped and does not have direct internet access to download packages from PyPI. The engineer has already downloaded the required PyYAML wheel (.whl) file onto their laptop. The data engineer wants to install the PyYAML package from the local wheel file so that it is automatically available whenever any new cluster is provisioned in their Databricks workspace. Which approach should the data engineer use?

Options:

A- Upload the PyYAML.whl file to a Unity Catalog volume. Add the path to the Unity Catalog allowlist if required. Then create a cluster-scoped init script that executes pip install /path/to/PyYAML.whl.
B- Set up a private PyPI repository, register the wheel there, and create a cluster-scoped init script that executes /databricks/python/bin/pip install --index-url=https://{repo-url} PyYAML on the cluster.
C- Upload the PyYAML.whl file under the data engineer's user home directory in the Workspace, and create a cluster-scoped init script that executes %pip install /path/to/PyYAML.whl on the shared cluster.
D- Add the PyYAML.whl file directly to Databricks Git Repos and assume that any cluster linked to the Repo will automatically have PyYAML installed from that file.

Answer:

A

Explanation:

Databricks documents that libraries and init scripts can be sourced from Unity Catalog volumes, and for standard access mode, relevant paths can require Unity Catalog allowlist configuration. Databricks also documents that init scripts run on every cluster startup, which is the mechanism that makes a package automatically available whenever a new cluster is provisioned. (Databricks Documentation)

Option A fits the air-gapped requirement because it does not depend on internet access and uses a startup-time installation mechanism. Option B still depends on reachable repository infrastructure. Option C is incorrect because %pip is notebook magic, not the correct form inside an init script. Option D is unsupported because storing a wheel in Git Repos does not automatically install it on cluster creation. (Databricks Documentation)

======

Question 2

Question Type: MultipleChoice

To identify the top users consuming compute resources, a data engineering team needs to monitor usage within their Databricks workspace for better resource utilization and cost control. The team decided to use Databricks system tables, available under the System catalog in Unity Catalog, to gain detailed visibility into workspace activity.

Which SQL query should the team run from the System catalog to achieve this?

Options:

A- SELECT sku_name,identity_metadata.created_by AS user_email,COUNT(usage_quantity) AS total_dbusFROM system.billing.usageGROUP BY user_email, sku_nameORDER BY total_dbus DESCLIMIT 10
B- SELECT identity_metadata.run_as AS user_email,SUM(usage_quantity) AS total_dbusFROM system.billing.usageGROUP BY user_emailORDER BY total_dbus DESCLIMIT 10
C- SELECT sku_name,identity_metadata.created_by AS user_email,SUM(usage_quantity * usage_unit) AS total_dbusFROM system.billing.usageGROUP BY user_email, sku_nameORDER BY total_dbus DESCLIMIT 10
D- SELECT sku_name,usage_metadata.run_name AS user_email,SUM(usage_quantity) AS total_dbusFROM system.billing.usageGROUP BY user_email, sku_nameORDER BY total_dbus DESCLIMIT 10

Answer:

B

Explanation:

The system.billing.usage table in the Unity Catalog System schema provides detailed usage metrics for each workload in the workspace. The field identity_metadata.run_as identifies the user or service principal under which the job or query executed. Summing usage_quantity provides total DBU (Databricks Unit) consumption per user. According to Databricks documentation, this table is the authoritative source for monitoring workspace cost drivers, showing compute SKU, user, and DBU consumption over time. Grouping by identity_metadata.run_as and summing usage_quantity produces the correct aggregation to determine top users. Other queries use non-existent or incorrect fields (created_by, run_name, or multiplied usage quantities), which do not reflect actual billing metrics.

Question 3

Question Type: MultipleChoice

A data engineer is designing a data pipeline in Databricks that needs to process records from a Kafka stream where late-arriving data is common. Which approach should the data engineer use?

Options:

A- Use a watermark to specify the allowed lateness to accommodate records that arrive after their expected window, ensuring correct aggregation and state management.
B- Use an Auto CDC pipeline with batch tables to simplify late data handling.
C- Use batch processing and overwrite the entire output table each time to ensure late data is incorporated correctly.
D- Implement a custom solution using Databricks Jobs to periodically reprocess all historical data.

Answer:

A

Explanation:

Databricks and Apache Spark document watermarks as the standard mechanism for handling late-arriving data in Structured Streaming. A watermark defines how long the engine should continue waiting for out-of-order event-time data and helps manage state for aggregations, joins, and deduplication. (Databricks Documentation)

This directly matches Kafka streaming scenarios where lateness is common. The other options are not the standard streaming solution for event-time late data: Auto CDC is for change data capture use cases, and repeated full historical reprocessing is inefficient and unnecessary when watermarking is the built-in feature designed for this problem. (Databricks Documentation)

======

Question 4

Question Type: MultipleChoice

A data engineering team has a time-consuming data ingestion job with three data sources. Each notebook takes about one hour to load new data. One day, the job fails because a notebook

update introduced a new required configuration parameter. The team must quickly fix the issue and load the latest data from the failing source.

Which action should the team take?

Options:

A- Repair the run with the new parameter, and update the task by adding the missing task parameter.
B- Update the task by adding the missing task parameter, and manually run the job.
C- Repair the run with the new parameter.
D- Share the analysis with the failing notebook owner so that they can fix it quickly.

Answer:

A

Explanation:

The repair run capability in Databricks Jobs allows re-execution of failed tasks without re-running successful ones. When a parameterized job fails due to missing or incorrect task configuration, engineers can perform a repair run to fix inputs or parameters and resume from the failed state.

This approach saves time, reduces cost, and ensures workflow continuity by avoiding unnecessary recomputation. Additionally, updating the task definition with the missing parameter prevents future runs from failing.

Running the job manually (B) loses run context; (C) alone does not prevent recurrence; (D) delays resolution. Thus, A follows the correct operational and recovery practice.

Question 5

Question Type: MultipleChoice

A data engineer is implementing Unity Catalog governance for a multi-team environment. Data scientists need interactive clusters for basic data exploration tasks, while automated ETL jobs require dedicated processing.

How should the data engineer configure cluster isolation policies to enforce least privilege and ensure Unity Catalog compliance?

Options:

A- Use only DEDICATED access mode for both interactive workloads and automated jobs to maximize security isolation.
B- Allow all users to create any cluster type and rely on manual configuration to enable Unity Catalog access modes.
C- Configure all clusters with NO ISOLATION_SHARED access mode since Unity Catalog works with any cluster configuration.
D- Create compute policies with STANDARD access mode for interactive workloads and DEDICATED access mode for automated jobs.

Answer:

D

Explanation:

Unity Catalog enforces governance and data isolation through cluster access modes and compute policies. According to Databricks documentation, ''Interactive clusters that multiple users share should use Standard access mode, while automated jobs and production pipelines should use Dedicated access mode for stricter isolation.'' Standard access mode allows multiple users to share the same compute resources but still respects Unity Catalog permissions. Dedicated access mode isolates the job run's execution environment, ensuring that data access is limited to the job's identity. Configuring these modes within compute policies enforces least privilege and ensures all compute complies with Unity Catalog security standards. Options A and C are incorrect because using only Dedicated clusters reduces resource efficiency, while ''No isolation'' clusters are not Unity Catalog compliant.

Question 6

Question Type: MultipleChoice

Which REST API call can be used to review the notebooks configured to run as tasks in a multi-task job?

Options:

A- /jobs/runs/list
B- /jobs/runs/get-output
C- /jobs/runs/get
D- /jobs/get
E- /jobs/list

Answer:

D

Explanation:

This is the correct answer because it is the REST API call that can be used to review the notebooks configured to run as tasks in a multi-task job. The REST API is an interface that allows programmatically interacting with Databricks resources, such as clusters, jobs, notebooks, or tables. The REST API uses HTTP methods, such as GET, POST, PUT, or DELETE, to perform operations on these resources. The /jobs/get endpoint is a GET method that returns information about a job given its job ID. The information includes the job settings, such as the name, schedule, timeout, retries, email notifications, and tasks. The tasks are the units of work that a job executes. A task can be a notebook task, which runs a notebook with specified parameters; a jar task, which runs a JAR uploaded to DBFS with specified main class and arguments; or a python task, which runs a Python file uploaded to DBFS with specified parameters. A multi-task job is a job that has more than one task configured to run in a specific order or in parallel. By using the /jobs/get endpoint, one can review the notebooks configured to run as tasks in a multi-task job. Verified Reference: [Databricks Certified Data Engineer Professional], under ''Databricks Jobs'' section; Databricks Documentation, under ''Get'' section; Databricks Documentation, under ''JobSettings'' section.

Question 7

Question Type: MultipleChoice

Which configuration parameter directly affects the size of a spark-partition upon ingestion of data into Spark?

Options:

A- spark.sql.files.maxPartitionBytes
B- spark.sql.autoBroadcastJoinThreshold
C- spark.sql.files.openCostInBytes
D- spark.sql.adaptive.coalescePartitions.minPartitionNum
E- spark.sql.adaptive.advisoryPartitionSizeInBytes

Answer:

A

Explanation:

This is the correct answer because spark.sql.files.maxPartitionBytes is a configuration parameter that directly affects the size of a spark-partition upon ingestion of data into Spark. This parameter configures the maximum number of bytes to pack into a single partition when reading files from file-based sources such as Parquet, JSON and ORC. The default value is 128 MB, which means each partition will be roughly 128 MB in size, unless there are too many small files or only one large file. Verified Reference: [Databricks Certified Data Engineer Professional], under ''Spark Configuration'' section;Databricks Documentation, under ''Available Properties - spark.sql.files.maxPartitionBytes'' section.