

DATABRICKS-CERTIFIED- PR OFSSIONAL-DATA-ENGINEER^{Q&As}

Databricks Certified Professional Data Engineer Exam

**Pass Databricks DATABRICKS-CERTIFIED-
PROFESSIONAL-DATA-ENGINEER Exam with 100%
Guarantee**

Free Download Real Questions & Answers **PDF** and **VCE** file from:

<https://www.leads4pass.com/databricks-certified-professional-data-engineer.html>

100% Passing Guarantee
100% Money Back Assurance

Following Questions and Answers are all new published by Databricks
Official Exam Center

- ⚙️ **Instant Download** After Purchase
- ⚙️ **100% Money Back** Guarantee
- ⚙️ **365 Days** Free Update
- ⚙️ **800,000+** Satisfied Customers



QUESTION 1

Which statement regarding stream-static joins and static Delta tables is correct?

- A. Each microbatch of a stream-static join will use the most recent version of the static Delta table as of each microbatch.
- B. Each microbatch of a stream-static join will use the most recent version of the static Delta table as of the job's initialization.
- C. The checkpoint directory will be used to track state information for the unique keys present in the join.
- D. Stream-static joins cannot use static Delta tables because of consistency issues.
- E. The checkpoint directory will be used to track updates to the static Delta table.

Correct Answer: A

Explanation: This is the correct answer because stream-static joins are supported by Structured Streaming when one of the tables is a static Delta table. A static Delta table is a Delta table that is not updated by any concurrent writes, such as appends or merges, during the execution of a streaming query. In this case, each microbatch of a stream-static join will use the most recent version of the static Delta table as of each microbatch, which means it will reflect any changes made to the static Delta table before the start of each microbatch. Verified References: [Databricks Certified Data Engineer Professional], under "Structured Streaming" section; Databricks Documentation, under "Stream and static joins" section.

QUESTION 2

Which statement describes Delta Lake Auto Compaction?

- A. An asynchronous job runs after the write completes to detect if files could be further compacted; if yes, an optimize job is executed toward a default of 1 GB.
- B. Before a Jobs cluster terminates, optimize is executed on all tables modified during the most recent job.
- C. Optimized writes use logical partitions instead of directory partitions; because partition boundaries are only represented in metadata, fewer small files are written.
- D. Data is queued in a messaging bus instead of committing data directly to memory; all data is committed from the messaging bus in one batch once the job is complete.
- E. An asynchronous job runs after the write completes to detect if files could be further compacted; if yes, an optimize job is executed toward a default of 128 MB.

Correct Answer: E

Explanation: This is the correct answer because it describes the behavior of Delta Lake Auto Compaction, which is a feature that automatically optimizes the layout of Delta Lake tables by coalescing small files into larger ones. Auto Compaction runs as an asynchronous job after a write to a table has succeeded and checks if files within a partition can be further compacted. If yes, it runs an optimize job with a default target file size of 128 MB. Auto Compaction only compacts files that have not been compacted previously. Verified References: [Databricks Certified Data Engineer Professional], under "Delta Lake" section; Databricks Documentation, under "Auto Compaction for Delta Lake on Databricks" section.

QUESTION 3

A junior data engineer has configured a workload that posts the following JSON to the Databricks REST API endpoint `2.0/jobs/create`.

```
{
  "name": "Ingest new data",
  "existing_cluster_id": "6015-954420-peace720",
  "notebook_task": {
    "notebook_path": "/Prod/ingest.py"
  }
}
```

Assuming that all configurations and referenced resources are available, which statement describes the result of executing this workload three times?

- A. Three new jobs named "Ingest new data" will be defined in the workspace, and they will each run once daily.
- B. The logic defined in the referenced notebook will be executed three times on new clusters with the configurations of the provided cluster ID.
- C. Three new jobs named "Ingest new data" will be defined in the workspace, but no jobs will be executed.
- D. One new job named "Ingest new data" will be defined in the workspace, but it will not be executed.
- E. The logic defined in the referenced notebook will be executed three times on the referenced existing all purpose cluster.

Correct Answer: C

Explanation: This is the correct answer because the JSON posted to the Databricks REST API endpoint `2.0/jobs/create` defines a new job with a name, an existing cluster id, and a notebook task. However, it does not specify any schedule or trigger for the job execution. Therefore, three new jobs with the same name and configuration will be created in the workspace, but none of them will be executed until they are manually triggered or scheduled. Verified References: [Databricks Certified Data Engineer Professional], under "Monitoring and Logging" section; [Databricks Documentation], under "Jobs API - Create" section.

QUESTION 4

A production cluster has 3 executor nodes and uses the same virtual machine type for the driver and executor.

When evaluating the Ganglia Metrics for this cluster, which indicator would signal a bottleneck caused by code executing on the driver?

- A. The five Minute Load Average remains consistent/flat
- B. Bytes Received never exceeds 80 million bytes per second
- C. Total Disk Space remains constant

- D. Network I/O never spikes
- E. Overall cluster CPU utilization is around 25%

Correct Answer: E

Explanation: This is the correct answer because it indicates a bottleneck caused by code executing on the driver. A bottleneck is a situation where the performance or capacity of a system is limited by a single component or resource. A bottleneck can cause slow execution, high latency, or low throughput. A production cluster has 3 executor nodes and uses the same virtual machine type for the driver and executor. When evaluating the Ganglia Metrics for this cluster, one can look for indicators that show how the cluster resources are being utilized, such as CPU, memory, disk, or network. If the overall cluster CPU utilization is around 25%, it means that only one out of the four nodes (driver + 3 executors) is using its full CPU capacity, while the other three nodes are idle or underutilized. This suggests that the code executing on the driver is taking too long or consuming too much CPU resources, preventing the executors from receiving tasks or data to process. This can happen when the code has driver-side operations that are not parallelized or distributed, such as collecting large amounts of data to the driver, performing complex calculations on the driver, or using non-Spark libraries on the driver. Verified References: [Databricks Certified Data Engineer Professional], under "Spark Core" section; Databricks Documentation, under "View cluster status and event logs - Ganglia metrics" section; Databricks Documentation, under "Avoid collecting large RDDs" section.

QUESTION 5

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

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

Correct 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 References: [Databricks Certified Data Engineer Professional], under "Databricks Jobs" section; Databricks Documentation, under "Get" section; Databricks Documentation, under "JobSettings" section.

[Latest DATABRICKS-CERTIFIED-PROFESSIONAL-DATA-ENGINEER Dumps](#) | [DATABRICKS-CERTIFIED-PROFESSIONAL-DATA-ENGINEER PDF Dumps](#) | [DATABRICKS-CERTIFIED-PROFESSIONAL-DATA-ENGINEER Practice Test](#)