DATABRICKS-CERTIFIED-PROFESSIONAL-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.passapply.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

QUESTION 1

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

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

Correct Answer: A

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 References:

[Databricks Certified Data Engineer Professional], under "Spark Configuration" section; Databricks Documentation, under "Available Properties-spark.sql.files.maxPartitionBytes" section.

QUESTION 2

An hourly batch job is configured to ingest data files from a cloud object storage container where each batch represent all records produced by the source system in a given hour. The batch job to process these records into the Lakehouse is sufficiently delayed to ensure no late-arriving data is missed. The user_id field represents a unique key for the data, which has the following schema:

user_id BIGINT, username STRING, user_utc STRING, user_region STRING, last_login BIGINT, auto_pay BOOLEAN, last_updated BIGINT

New records are all ingested into a table named account_history which maintains a full record of all data in the same schema as the source. The next table in the system is named account_current and is implemented as a Type 1 table representing the most recent value for each unique user_id.

Assuming there are millions of user accounts and tens of thousands of records processed hourly, which implementation can be used to efficiently update the described account_current table as part of each hourly batch job?

A. Use Auto Loader to subscribe to new files in the account history directory; configure a Structured Streaminq trigger once job to batch update newly detected files into the account current table.

B. Overwrite the account current table with each batch using the results of a query against the account history table grouping by user id and filtering for the max value of last updated.

C. Filter records in account history using the last updated field and the most recent hour processed, as well as the max last iogin by user id write a merge statement to update or insert the most recent value for each user id.

D. Use Delta Lake version history to get the difference between the latest version of account history and one version prior, then write these records to account current.

E. Filter records in account history using the last updated field and the most recent hour processed, making sure to deduplicate on username; write a merge statement to update or insert the most recent value for each username.

Correct Answer: C

This is the correct answer because it efficiently updates the account current table with only the most recent value for each user id. The code filters records in account history using the last updated field and the most recent hour processed, which means it will only process the latest batch of data. It also filters by the max last login by user id, which means it will only keep the most recent record for each user id within that batch. Then, it writes a merge statement to update or insert the most recent value for each user id into account current, which means it will perform an upsert operation based on the user id column. Verified References: [Databricks Certified Data Engineer Professional], under "Delta Lake" section; Databricks Documentation, under "Upsert into a table using merge" section.

QUESTION 3

A junior developer complains that the code in their notebook isn\\'t producing the correct results in the development environment. A shared screenshot reveals that while they\\'re using a notebook versioned with Databricks Repos, they\\'re using a personal branch that contains old logic. The desired branch named dev-2.3.9 is not available from the branch selection dropdown.

Which approach will allow this developer to review the current logic for this notebook?

A. Use Repos to make a pull request use the Databricks REST API to update the current branch to dev-2.3.9

B. Use Repos to pull changes from the remote Git repository and select the dev-2.3.9 branch.

C. Use Repos to checkout the dev-2.3.9 branch and auto-resolve conflicts with the current branch

D. Merge all changes back to the main branch in the remote Git repository and clone the repo again

E. Use Repos to merge the current branch and the dev-2.3.9 branch, then make a pull request to sync with the remote repository

Correct Answer: B

This is the correct answer because it will allow the developer to update their local repository with the latest changes from the remote repository and switch to the desired branch. Pulling changes will not affect the current branch or create any conflicts, as it will only fetch the changes and not merge them. Selecting the dev-2.3.9 branch from the dropdown will checkout that branch and display its contents in the notebook. Verified References: [Databricks Certified Data Engineer Professional], under "Databricks Tooling" section; Databricks Documentation, under "Pull changes from a remote repository" section.

QUESTION 4

A Delta Lake table in the Lakehouse named customer_parsams is used in churn prediction by the machine learning team. The table contains information about customers derived from a number of upstream sources. Currently, the data engineering team populates this table nightly by overwriting the table with the current valid values derived from upstream data sources.

Immediately after each update succeeds, the data engineer team would like to determine the difference between the

new version and the previous of the table.

Given the current implementation, which method can be used?

A. Parse the Delta Lake transaction log to identify all newly written data files.

B. Execute DESCRIBE HISTORY customer_churn_params to obtain the full operation metrics for the update, including a log of all records that have been added or modified.

C. Execute a query to calculate the difference between the new version and the previous version using Delta Lake\'s built-in versioning and time travel functionality.

D. Parse the Spark event logs to identify those rows that were updated, inserted, or deleted.

Correct Answer: C

Delta Lake provides built-in versioning and time travel capabilities, allowing users to query previous snapshots of a table. This feature is particularly useful for understanding changes between different versions of the table. In this scenario,

where the table is overwritten nightly, you can use Delta Lake\'s time travel feature to execute a query comparing the latest version of the table (the current state) with its previous version. This approach effectively identifies the differences

(such as new, updated, or deleted records) between the two versions. The other options do not provide a straightforward or efficient way to directly compare different versions of a Delta Lake table.

References:

Delta Lake Documentation on Time Travel: Delta Time Travel Delta Lake Versioning: Delta Lake Versioning Guide

---

QUESTION 5

The Databricks CLI is use to trigger a run of an existing job by passing the job_id parameter. The response that the job run request has been submitted successfully includes a filed run_id.

Which statement describes what the number alongside this field represents?

A. The job_id is returned in this field.

B. The job_id and number of times the job has been are concatenated and returned.

C. The number of times the job definition has been run in the workspace.

D. The globally unique ID of the newly triggered run.

Correct Answer: D

When triggering a job run using the Databricks CLI, the run_id field in the response represents a globally unique identifier for that particular run of the job. This run_id is distinct from the job_id. While the job_id identifies the job definition and is constant across all runs of that job, the run_id is unique to each execution and is used to track and query the status of that specific job run within the Databricks environment. This distinction allows users to manage and reference individual executions of a job directly.

DATABRICKS-CERTIFIED-
PROFESSIONAL-DATA-
ENGINEER PDF Dumps

DATABRICKS-CERTIFIED-
PROFESSIONAL-DATA-
ENGINEER Practice Test

DATABRICKS-CERTIFIED-
PROFESSIONAL-DATA-
ENGINEER Exam
Questions

DATABRICKS-CERTIFIED-PROFESSIONAL-DATA-ENGINEER PDF Dumps | DATABRICKS-CERTIFIED-
PROFESSIONAL-DATA-ENGINEER Practice Test |
DATABRICKS-CERTIFIED-PROFESSIONAL-DATA-ENGINEER Exam Questions

6 / 6