



VCE & PDF

PassApply.com

<https://www.passapply.com/databricks-certified-professional-data-engineer>.  
2024 Latest passapply DATABRICKS-CERTIFIED-PROFESSIONAL-DATA-ENGINEER PDF and VCE dumps Download

---

# DATABRICKS-CERTIFIED- PR OFESIONAL-DATA-ENGINEER<sup>Q&As</sup>

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



VCE & PDF

PassApply.com

[https://www.passapply.com/databricks-certified-professional-data-engineer.](https://www.passapply.com/databricks-certified-professional-data-engineer)  
2024 Latest passapply DATABRICKS-CERTIFIED-PROFESSIONAL-DATA-ENGINEER PDF and VCE dumps Download

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





### QUESTION 1

A data architect has designed a system in which two Structured Streaming jobs will concurrently write to a single bronze Delta table. Each job is subscribing to a different topic from an Apache Kafka source, but they will write data with the same schema. To keep the directory structure simple, a data engineer has decided to nest a checkpoint directory to be shared by both streams.

The proposed directory structure is displayed below:

```
./bronze
├── __checkpoint
├── __delta_log
├── year_week=2020_01
├── year_week=2020_02
└── ...
```

Which statement describes whether this checkpoint directory structure is valid for the given scenario and why?

- A. No; Delta Lake manages streaming checkpoints in the transaction log.
- B. Yes; both of the streams can share a single checkpoint directory.
- C. No; only one stream can write to a Delta Lake table.
- D. Yes; Delta Lake supports infinite concurrent writers.
- E. No; each of the streams needs to have its own checkpoint directory.

Correct Answer: E

Explanation: This is the correct answer because checkpointing is a critical feature of Structured Streaming that provides fault tolerance and recovery in case of failures. Checkpointing stores the current state and progress of a streaming query in a reliable storage system, such as DBFS or S3. Each streaming query must have its own checkpoint directory that is unique and exclusive to that query. If two streaming queries share the same checkpoint directory, they will interfere with each other and cause unexpected errors or data loss. Verified References: [Databricks Certified Data Engineer Professional], under "Structured Streaming" section; Databricks Documentation, under "Checkpointing" section.

### QUESTION 2

Incorporating unit tests into a PySpark application requires upfront attention to the design of your jobs, or a potentially significant refactoring of existing code.

Which statement describes a main benefit that offset this additional effort?

- A. Improves the quality of your data



- B. Validates a complete use case of your application
- C. Troubleshooting is easier since all steps are isolated and tested individually
- D. Yields faster deployment and execution times
- E. Ensures that all steps interact correctly to achieve the desired end result

Correct Answer: C

### QUESTION 3

A nightly job ingests data into a Delta Lake table using the following code:

```
from pyspark.sql.functions import current_timestamp, input_file_name, col
from pyspark.sql.column import Column

def ingest_daily_batch(time_col: Column, year:int, month:int, day:int):
    (spark.read
     .format("parquet")
     .load(f"/mnt/daily_batch/{year}/{month}/{day}")
     .select("*",
            time_col.alias("ingest_time"),
            input_file_name().alias("source_file")
            )
     .write
     .mode("append")
     .saveAsTable("bronze")
    )
```

The next step in the pipeline requires a function that returns an object that can be used to manipulate new records that have not yet been processed to the next table in the pipeline. Which code snippet completes this function definition?

- A. def new\_records():
- B. return spark.readStream.table("bronze")
- C. return spark.readStream.load("bronze")
- D. return spark.read.option("readChangeFeed", "true").table ("bronze")

Correct Answer: D

Explanation: This is the correct answer because it completes the function definition that returns an object that can be used to manipulate new records that have not yet been processed to the next table in the pipeline. The object returned by this function is a DataFrame that contains all change events from a Delta Lake table that has enabled change data feed. The readChangeFeed option is set to true to indicate that the DataFrame should read changes from the table, and the table argument specifies the name of the table to read changes from. The DataFrame will have a schema that includes four columns: operation, partition, value, and timestamp. The operation column indicates the type of change



event, such as insert, update, or delete. The partition column indicates the partition where the change event occurred. The value column contains the actual data of the change event as a struct type. The timestamp column indicates the time when the change event was committed. Verified References: [Databricks Certified Data Engineer Professional], under "Delta Lake" section; Databricks Documentation, under "Read changes in batch queries" section.

#### QUESTION 4

A junior data engineer has been asked to develop a streaming data pipeline with a grouped aggregation using `DataFrameDf`. The pipeline needs to calculate the average humidity and average temperature for each non-overlapping five-minute interval. Events are recorded once per minute per device.

Streaming `DataFrameDf` has the following schema:

```
"device_id INT, event_time TIMESTAMP, temp FLOAT, humidity FLOAT"
```

Code block:

```
df.withWatermark("event_time", "10 minutes")
  .groupBy(
    _____
    "device_id"
  )
  .agg(
    avg("temp").alias("avg_temp"),
    avg("humidity").alias("avg_humidity")
  )
  .writeStream
  .format("delta")
  .saveAsTable("sensor_avg")
```

Choose the response that correctly fills in the blank within the code block to complete this task.

- A. `to_interval("event_time", "5 minutes").alias("time")`
- B. `window("event_time", "5 minutes").alias("time")`
- C. `"event_time"`
- D. `window("event_time", "10 minutes").alias("time")`
- E. `lag("event_time", "10 minutes").alias("time")`

Correct Answer: B



Explanation: This is the correct answer because the window function is used to group streaming data by time intervals. The window function takes two arguments: a time column and a window duration. The window duration specifies how long each window is, and must be a multiple of 1second. In this case, the window duration is "5 minutes", which means each window will cover a non-overlapping five-minute interval. The window function also returns a struct column with two fields: start and end, which represent the start and end time of each window. The alias function is used to rename the struct column as "time". Verified References: [Databricks Certified Data Engineer Professional], under "Structured Streaming" section; Databricks Documentation, under "WINDOW" section.

## QUESTION 5

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 Streaming 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 login 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

Explanation: 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 6

To reduce storage and compute costs, the data engineering team has been tasked with curating a series of aggregate



tables leveraged by business intelligence dashboards, customer-facing applications, production machine learning models, and ad hoc analytical queries.

The data engineering team has been made aware of new requirements from a customer-facing application, which is the only downstream workload they manage entirely. As a result, an aggregate table used by numerous teams across the organization will need to have a number of fields renamed, and additional fields will also be added.

Which of the solutions addresses the situation while minimally interrupting other teams in the organization without increasing the number of tables that need to be managed?

- A. Send all users notice that the schema for the table will be changing; include in the communication the logic necessary to revert the new table schema to match historic queries.
- B. Configure a new table with all the requisite fields and new names and use this as the source for the customer-facing application; create a view that maintains the original data schema and table name by aliasing select fields from the new table.
- C. Create a new table with the required schema and new fields and use Delta Lake's deep clone functionality to sync up changes committed to one table to the corresponding table.
- D. Replace the current table definition with a logical view defined with the query logic currently writing the aggregate table; create a new table to power the customer-facing application.
- E. Add a table comment warning all users that the table schema and field names will be changing on a given date; overwrite the table in place to the specifications of the customer-facing application.

Correct Answer: B

Explanation: This is the correct answer because it addresses the situation while minimally interrupting other teams in the organization without increasing the number of tables that need to be managed. The situation is that an aggregate table used by numerous teams across the organization will need to have a number of fields renamed, and additional fields will also be added, due to new requirements from a customer-facing application. By configuring a new table with all the requisite fields and new names and using this as the source for the customer-facing application, the data engineering team can meet the new requirements without affecting other teams that rely on the existing table schema and name. By creating a view that maintains the original data schema and table name by aliasing select fields from the new table, the data engineering team can also avoid duplicating data or creating additional tables that need to be managed. Verified References: [Databricks Certified Data Engineer Professional], under "Lakehouse" section; Databricks Documentation, under "CREATE VIEW" section.

## QUESTION 7

When scheduling Structured Streaming jobs for production, which configuration automatically recovers from query failures and keeps costs low?

- A. Cluster: New Job Cluster; Retries: Unlimited; Maximum Concurrent Runs: Unlimited
- B. Cluster: New Job Cluster; Retries: None; Maximum Concurrent Runs: 1
- C. Cluster: Existing All-Purpose Cluster; Retries: Unlimited; Maximum Concurrent Runs: 1
- D. Cluster: Existing All-Purpose Cluster; Retries: Unlimited; Maximum Concurrent Runs: 1
- E. Cluster: Existing All-Purpose Cluster; Retries: None; Maximum Concurrent Runs: 1

Correct Answer: B



Explanation: This is the best configuration for scheduling Structured Streaming jobs for production, as it automatically recovers from query failures and keeps costs low. A new job cluster is created for each run of the job and terminated when the job completes, which saves costs and avoids resource contention. Retries are not needed for Structured Streaming jobs, as they can automatically recover from failures using checkpointing and write-ahead logs. Maximum concurrent runs should be set to 1 to avoid duplicate output or data loss. Verified References: Databricks Certified Data Engineer Professional, under "Monitoring and Logging" section; Databricks Documentation, under "Schedule streaming jobs" section.

## QUESTION 8

An external object storage container has been mounted to the location/mnt/finance\_eda\_bucket. The following logic was executed to create a database for the finance team:

```
CREATE DATABASE finance_eda_db
LOCATION '/mnt/finance_eda_bucket';
GRANT USAGE ON DATABASE finance_eda_db TO finance;
GRANT CREATE ON DATABASE finance_eda_db TO finance;
```

After the database was successfully created and permissions configured, a member of the finance team runs the following code:

```
CREATE TABLE finance_eda_db.tx_sales AS
SELECT *
FROM sales
WHERE state = "TX";
```

If all users on the finance team are members of thefinancegroup, which statement describes how thetx\_salestable will be created?

- A. A logical table will persist the query plan to the Hive Metastore in the Databricks control plane.
- B. An external table will be created in the storage container mounted to /mnt/finance\_eda\_bucket.
- C. A logical table will persist the physical plan to the Hive Metastore in the Databricks control plane.
- D. An managed table will be created in the storage container mounted to /mnt/finance\_eda\_bucket.
- E. A managed table will be created in the DBFS root storage container.

Correct Answer: B

Explanation: The code uses the CREATE TABLE USING DELTA command to create a Delta Lake table from an existing Parquet file stored in an external object storage container mounted to /mnt/finance\_eda\_bucket. The code also uses the LOCATION keyword to specify the path to the Parquet file as /mnt/finance\_eda\_bucket/tx\_sales.parquet. By using the LOCATION keyword, the code creates an external table, which is a table that is stored outside of the default warehouse directory and whose metadata is not managed by Databricks. An external table can be created from an existing directory in a cloud storage system, such as DBFS or S3, that contains data files in a supported format, such as





Parquet or CSV. Verified References: [Databricks Certified Data Engineer Professional], under "Delta Lake" section; Databricks Documentation, under "Create an external table" section.

### QUESTION 9

A Delta Lake table was created with the below query:

```
CREATE TABLE prod.sales_by_stor  
USING DELTA  
LOCATION "/mnt/prod/sales_by_store"
```

Realizing that the original query had a typographical error, the below code was executed:

```
ALTER TABLE prod.sales_by_stor RENAME TO prod.sales_by_store
```

Which result will occur after running the second command?

- A. The table reference in the metastore is updated and no data is changed.
- B. The table name change is recorded in the Delta transaction log.
- C. All related files and metadata are dropped and recreated in a single ACID transaction.
- D. The table reference in the metastore is updated and all data files are moved.
- E. A new Delta transaction log is created for the renamed table.

Correct Answer: A

Explanation: The query uses the CREATE TABLE USING DELTA syntax to create a Delta Lake table from an existing Parquet file stored in DBFS. The query also uses the LOCATION keyword to specify the path to the Parquet file as /mnt/finance\_eda\_bucket/tx\_sales.parquet. By using the LOCATION keyword, the query creates an external table, which is a table that is stored outside of the default warehouse directory and whose metadata is not managed by Databricks. An external table can be created from an existing directory in a cloud storage system, such as DBFS or S3, that contains data files in a supported format, such as Parquet or CSV. The result that will occur after running the second command is that the table reference in the metastore is updated and no data is changed. The metastore is a service that stores metadata about tables, such as their schema, location, properties, and partitions. The metastore allows users to access tables using SQL commands or Spark APIs without knowing their physical location or format. When renaming an external table using the ALTER TABLE RENAME TO command, only the table reference in the metastore is updated with the new name; no data files or directories are moved or changed in the storage system. The table will still point to the same location and use the same format as before. However, if renaming a managed table, which is a table whose metadata and data are both managed by Databricks, both the table reference in the metastore and the data files in the default warehouse directory are moved and renamed accordingly. Verified References: [Databricks Certified Data Engineer Professional], under "Delta Lake" section; Databricks Documentation, under "ALTER TABLE RENAME TO" section; Databricks Documentation, under "Metastore" section; Databricks Documentation, under "Managed and external tables" section.

### QUESTION 10



The data engineering team has configured a Databricks SQL query and alert to monitor the values in a Delta Lake table. The recent\_sensor\_recordings table contains an identifying sensor\_id alongside the timestamp and temperature for the most recent 5 minutes of recordings.

The below query is used to create the alert:

```
SELECT MEAN(temperature), MAX(temperature), MIN(temperature)
FROM recent_sensor_recordings
GROUP BY sensor_id
```

The query is set to refresh each minute and always completes in less than 10 seconds. The alert is set to trigger when  $\text{mean(temperature)} > 120$ . Notifications are triggered to be sent at most every 1 minute.

If this alert raises notifications for 3 consecutive minutes and then stops, which statement must be true?

- A. The total average temperature across all sensors exceeded 120 on three consecutive executions of the query
- B. The recent\_sensor\_recordings table was unresponsive for three consecutive runs of the query
- C. The source query failed to update properly for three consecutive minutes and then restarted
- D. The maximum temperature recording for at least one sensor exceeded 120 on three consecutive executions of the query
- E. The average temperature recordings for at least one sensor exceeded 120 on three consecutive executions of the query

Correct Answer: E

Explanation: This is the correct answer because the query is using a GROUP BY clause on the sensor\_id column, which means it will calculate the mean temperature for each sensor separately. The alert will trigger when the mean temperature for any sensor is greater than 120, which means at least one sensor had an average temperature above 120 for three consecutive minutes. The alert will stop when the mean temperature for all sensors drops below 120.  
Verified References: [Databricks Certified Data Engineer Professional], under "SQL Analytics" section; Databricks Documentation, under "Alerts" section.

[Latest DATABRICKS-CERTIFIED-PROFESSIONAL-DATA-ENGINEER Dumps](#)

[DATABRICKS-CERTIFIED-PROFESSIONAL-DATA-ENGINEER Exam Questions](#)

[DATABRICKS-CERTIFIED-PROFESSIONAL-DATA-ENGINEER Braindumps](#)