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.geekcert.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

A Structured Streaming job deployed to production has been experiencing delays during peak hours of the day. At present, during normal execution, each microbatch of data is processed in less than 3 seconds. During peak hours of the day, execution time for each microbatch becomes very inconsistent, sometimes exceeding 30 seconds. The streaming write is currently configured with a trigger interval of 10 seconds.

Holding all other variables constant and assuming records need to be processed in less than 10 seconds, which adjustment will meet the requirement?

A. Decrease the trigger interval to 5 seconds; triggering batches more frequently allows idle executors to begin processing the next batch while longer running tasks from previous batches finish.

B. Increase the trigger interval to 30 seconds; setting the trigger interval near the maximum execution time observed for each batch is always best practice to ensure no records are dropped.

C. The trigger interval cannot be modified without modifying the checkpoint directory; to maintain the current stream state, increase the number of shuffle partitions to maximize parallelism.

D. Use the trigger once option and configure a Databricks job to execute the query every 10 seconds; this ensures all backlogged records are processed with each batch.

E. Decrease the trigger interval to 5 seconds; triggering batches more frequently may prevent records from backing up and large batches from causing spill.

Correct Answer: E

The adjustment that will meet the requirement of processing records in less than 10 seconds is to decrease the trigger interval to 5 seconds. This is because triggering batches more frequently may prevent records from backing up and large

batches from causing spill. Spill is a phenomenon where the data in memory exceeds the available capacity and has to be written to disk, which can slow down the processing and increase the execution time1. By reducing the trigger interval,

the streaming query can process smaller batches of data more quickly and avoid spill. This can also improve the latency and throughput of the streaming job2.

The other options are not correct, because:

Option A is incorrect because triggering batches more frequently does not allow idle executors to begin processing the next batch while longer running tasks from previous batches finish. In fact, the opposite is true. Triggering batches more

frequently may cause concurrent batches to compete for the same resources and cause contention and backpressure2. This can degrade the performance and stability of the streaming job.

Option B is incorrect because increasing the trigger interval to 30 seconds is not a good practice to ensure no records are dropped. Increasing the trigger interval means that the streaming query will process larger batches of data less

frequently, which can increase the risk of spill, memory pressure, and timeouts12. This can also increase the latency and reduce the throughput of the streaming job. Option C is incorrect because the trigger interval can be modified without

modifying the checkpoint directory. The checkpoint directory stores the metadata and state of the streaming query, such as the offsets, schema, and configuration3. Changing the trigger interval does not affect the state of the streaming

query,

and does not require a new checkpoint directory. However, changing the number of shuffle partitions may affect the state of the streaming query, and may require a new checkpoint directory4.

Option D is incorrect because using the trigger once option and configuring a Databricks job to execute the query every 10 seconds does not ensure that all backlogged records are processed with each batch. The trigger once option means

that the streaming query will process all the available data in the source and then stop5. However, this does not guarantee that the query will finish processing within 10 seconds, especially if there are a lot of records in the source. Moreover,

configuring a Databricks job to execute the query every 10 seconds may cause overlapping or missed batches, depending on the execution time of the query.

References: Memory Management Overview, Structured Streaming Performance Tuning Guide, Checkpointing, Recovery Semantics after Changes in a Streaming Query, Triggers

QUESTION 2

A member of the data engineering team has submitted a short notebook that they wish to schedule as part of a larger data pipeline. Assume that the commands provided below produce the logically correct results when run as presented.

Cmd 1
```
rawDF = spark.table("raw_data")
```

Cmd 2
```
rawDF.printSchema()
```

Cmd 3
```
flattenedDF = rawDF.select("*", "values.*")
```

Cmd 4
```
finalDF = flattenedDF.drop("values")
```

Cmd 5
```
finalDF.explain()
```

Cmd 6
```
display(finalDF)
```

Cmd 7
```
finalDF.write.mode("append").saveAsTable("flat_data")
```

Which command should be removed from the notebook before scheduling it as a job?

A. Cmd 2

B. Cmd 3

C. Cmd 4

D. Cmd 5

E. Cmd 6

Correct Answer: E

Cmd 6 is the command that should be removed from the notebook before scheduling it as a job. This command is selecting all the columns from the finalDF dataframe and displaying them in the notebook. This is not necessary for the job, as

the finalDF dataframe is already written to a table in Cmd 7. Displaying the dataframe in the notebook will only consume

resources and time, and it will not affect the output of the job. Therefore, Cmd 6 is redundant and should be removed.

The other commands are essential for the job, as they perform the following tasks:

Cmd 1: Reads the raw_data table into a Spark dataframe called rawDF. Cmd 2: Prints the schema of the rawDF dataframe, which is useful for debugging and understanding the data structure.

Cmd 3: Selects all the columns from the rawDF dataframe, as well as the nested columns from the values struct column, and creates a new dataframe called flattenedDF.

Cmd 4: Drops the values column from the flattenedDF dataframe, as it is no longer needed after flattening, and creates a new dataframe called finalDF.

Cmd 5: Explains the physical plan of the finalDF dataframe, which is useful for optimizing and tuning the performance of the job.

Cmd 7: Writes the finalDF dataframe to a table called flat_data, using the append mode to add new data to the existing table.

QUESTION 3

What is a method of installing a Python package scoped at the notebook level to all nodes in the currently active cluster?

A. Use andPip install in a notebook cell

B. Run source env/bin/activate in a notebook setup script

C. Install libraries from PyPi using the cluster UI

D. Use andsh install in a notebook cell

Correct Answer: C

Installing a Python package scoped at the notebook level to all nodes in the currently active cluster in Databricks can be achieved by using the Libraries tab in the cluster UI. This interface allows you to install libraries across all nodes in the

cluster. While the %pip command in a notebook cell would only affect the driver node, using the cluster UI ensures that the package is installed on all nodes.

References:

Databricks Documentation on Libraries: Libraries

QUESTION 4

A junior data engineer is working to implement logic for a Lakehouse table named silver_device_recordings. The source data contains 100 unique fields in a highly nested JSON structure.

The silver_device_recordings table will be used downstream for highly selective joins on a number of fields, and will also be leveraged by the machine learning team to filter on a handful of relevant fields, in total, 15 fields have been identified

that will often be used for filter and join logic.

The data engineer is trying to determine the best approach for dealing with these nested fields before declaring the table schema.

Which of the following accurately presents information about Delta Lake and Databricks that may Impact their decision-making process?

A. Because Delta Lake uses Parquet for data storage, Dremel encoding information for nesting can be directly referenced by the Delta transaction log.

B. Tungsten encoding used by Databricks is optimized for storing string data: newly-added native support for querying JSON strings means that string types are always most efficient.

C. Schema inference and evolution on Databricks ensure that inferred types will always accurately match the data types used by downstream systems.

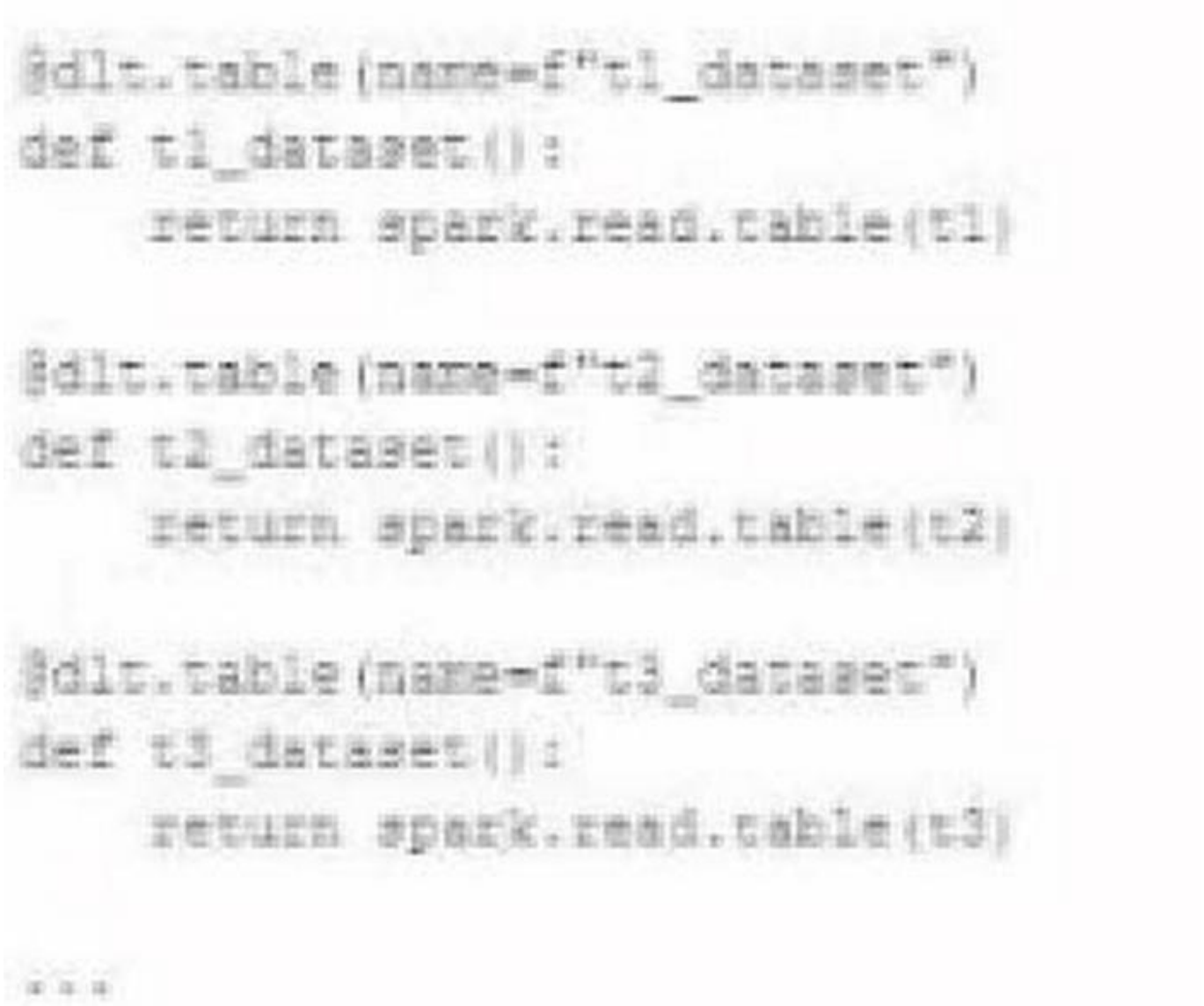
D. By default Delta Lake collects statistics on the first 32 columns in a table; these statistics are leveraged for data skipping when executing selective queries.

Correct Answer: D

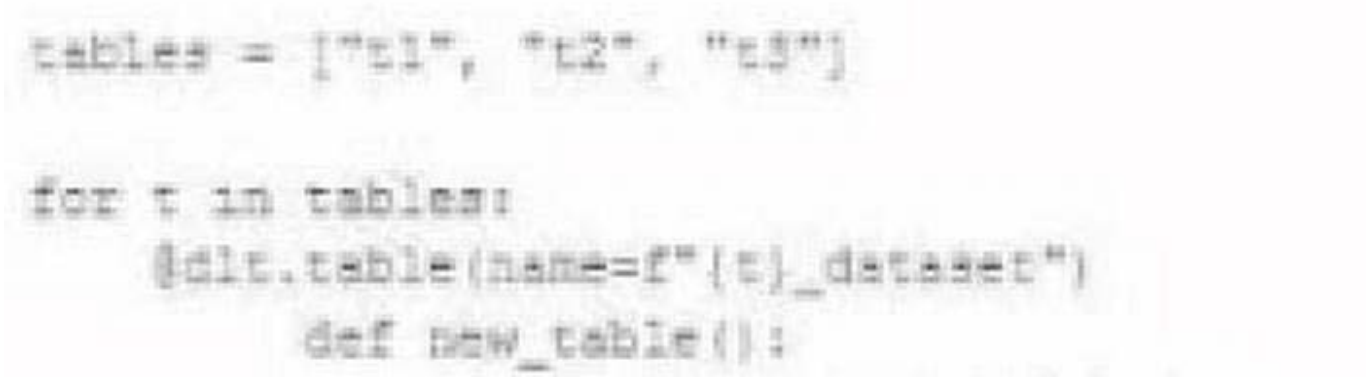
Delta Lake, built on top of Parquet, enhances query performance through data skipping, which is based on the statistics collected for each file in a table. For tables with a large number of columns, Delta Lake by default collects and stores statistics only for the first 32 columns. These statistics include min/max values and null counts, which are used to optimize query execution by skipping irrelevant data files. When dealing with highly nested JSON structures, understanding this behavior is crucial for schema design, especially when determining which fields should be flattened or prioritized in the table structure to leverage data skipping efficiently for performance optimization.References: Databricks documentation on Delta Lake optimization techniques, including data skipping and statistics collection (https://docs.databricks.com/delta/optimizations/index.html).

QUESTION 5

A data engineer wants to reflector the following DLT code, which includes multiple definition with very similar code:



In an attempt to programmatically create these tables using a parameterized table definition, the data engineer writes the following code.



The pipeline runs an update with this refactored code, but generates a different DAG showing incorrect configuration values for tables. How can the data engineer fix this?

A. Convert the list of configuration values to a dictionary of table settings, using table names as keys.

B. Convert the list of configuration values to a dictionary of table settings, using different input the for loop.

C. Load the configuration values for these tables from a separate file, located at a path provided by a pipeline parameter.

D. Wrap the loop inside another table definition, using generalized names and properties to replace with those from the inner table

Correct Answer: A

The issue with the refactored code is that it tries to use string interpolation to dynamically create table names within the dlc.table decorator, which will not correctly interpret the table names. Instead, by using a dictionary with table names as keys and their configurations as values, the data engineer can iterate over the dictionary items and use the keys (table names) to properly configure the table settings. This way, the decorator can correctly recognize each table name, and the corresponding configuration settings can be applied appropriately.