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.certbus.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

The data science team has requested assistance in accelerating queries on free form text from user reviews. The data is currently stored in Parquet with the below schema:

item_id INT, user_id INT, review_id INT, rating FLOAT, review STRING

The review column contains the full text of the review left by the user. Specifically, the data science team is looking to identify if any of 30 key words exist in this field. A junior data engineer suggests converting this data to Delta Lake will improve query performance.

Which response to the junior data engineer s suggestion is correct?

A. Delta Lake statistics are not optimized for free text fields with high cardinality.

B. Text data cannot be stored with Delta Lake.

C. ZORDER ON review will need to be run to see performance gains.

D. The Delta log creates a term matrix for free text fields to support selective filtering.

E. Delta Lake statistics are only collected on the first 4 columns in a table.

Correct Answer: A

Converting the data to Delta Lake may not improve query performance on free text fields with high cardinality, such as the review column. This is because Delta Lake collects statistics on the minimum and maximum values of each column, which are not very useful for filtering or skipping data on free text fields. Moreover, Delta Lake collects statistics on the first 32 columns by default, which may not include the review column if the table has more columns. Therefore, the junior data engineer\\'s suggestion is not correct. A better approach would be to use a full-text search engine, such as Elasticsearch, to index and query the review column. Alternatively, you can use natural language processing techniques, such as tokenization, stemming, and lemmatization, to preprocess the review column and create a new column with normalized terms that can be used for filtering or skipping data. References: Optimizations: https://docs.delta.io/latest/optimizations-oss.html Full-text search with Elasticsearch: https://docs.databricks.com/data/data-sources/elasticsearch.html Natural language processing: https://docs.databricks.com/applications/nlp/index.html

QUESTION 2

In order to facilitate near real-time workloads, a data engineer is creating a helper function to leverage the schema detection and evolution functionality of Databricks Auto Loader. The desired function will automatically detect the schema of the source directly, incrementally process JSON files as they arrive in a source directory, and automatically evolve the schema of the table when new fields are detected.

The function is displayed below with a blank:

```
def auto_load_json(source_path: str,
                   checkpoint_path: str,
                   target_table_path: str):
    (spark.readStream
        .format("cloudFiles")
        .option("cloudFiles.format", "json")
        .option("cloudFiles.schemaLocation", checkpoint_path)
        .load(source_path)
        _______
    )
```

Which response correctly fills in the blank to meet the specified requirements?

A.
```
.writeStream
.option("mergeSchema", True)
.start(target_table_path)
```

B.
```
.writeStream
.option("checkpointLocation", checkpoint_path)
.option("mergeSchema", True)
.trigger(once=True)
.start(target_table_path)
```

C.
```
.write
.option("checkpointLocation", checkpoint_path)
.option("mergeSchema", True)
.outputMode("append")
.save(target_table_path)
```

D.
```
.write
.option("mergeSchema", True)
.mode("append")
.save(target_table_path)
```

E.
```
.writeStream
.option("checkpointLocation", checkpoint_path)
.option("mergeSchema", True)
.start(target_table_path)
```

A. Option A

B. Option B

C. Option C

D. Option D

E. Option E

Correct Answer: B

Option B correctly fills in the blank to meet the specified requirements. Option B uses the "cloudFiles.schemaLocation" option, which is required for the schema detection and evolution functionality of Databricks Auto Loader. Additionally,

option B uses the "mergeSchema" option, which is required for the schema evolution functionality of Databricks Auto Loader. Finally, option B uses the "writeStream" method, which is required for the incremental processing of JSON files as

they arrive in a source directory. The other options are incorrect because they either omit the required options, use the wrong method, or use the wrong format. References:

Configure schema inference and evolution in Auto Loader:

https://docs.databricks.com/en/ingestion/auto-loader/schema.html Write streaming data: https://docs.databricks.com/spark/latest/structured-streaming/writing-streaming-data.html

QUESTION 3

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 to power several production monitoring dashboards and a production model. At present, 45 of the 100 fields are being used in at least one of these applications.

The data engineer is trying to determine the best approach for dealing with schema declaration given the highly-nested structure of the data and the numerous fields.

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

A. The 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.

B. Because Delta Lake uses Parquet for data storage, data types can be easily evolved by just modifying file footer information in place.

C. Human labor in writing code is the largest cost associated with data engineering workloads; as such, automating table declaration logic should be a priority in all migration workloads.

D. Because Databricks will infer schema using types that allow all observed data to be processed, setting types manually provides greater assurance of data quality enforcement.

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

Correct Answer: D

This is the correct answer because it accurately presents information about Delta Lake and Databricks that may impact the decision-making process of a junior data engineer who is trying to determine the best approach for dealing with schema declaration given the highly-nested structure of the data and the numerous fields. Delta Lake and Databricks support schema inference and evolution, which means that they can automatically infer the schema of a table from the source data and allow adding new columns or changing column types without affecting existing queries or pipelines. However, schema inference and evolution may not always be desirable or reliable, especially when dealing with complex or nested data structures or when enforcing data quality and consistency across different systems. Therefore,

setting types manually can provide greater assurance of data quality enforcement and avoid potential errors or conflicts due to incompatible or unexpected data types. Verified References: [Databricks Certified Data Engineer Professional], under "Delta Lake" section; Databricks Documentation, under "Schema inference and partition of streaming DataFrames/ Datasets" section.

QUESTION 4

Which of the following technologies can be used to identify key areas of text when parsing Spark Driver log4j output?

A. Regex

B. Julia

C. pyspsark.ml.feature

D. Scala Datasets

E. C++

Correct Answer: A

Regex, or regular expressions, are a powerful way of matching patterns in text. They can be used to identify key areas of text when parsing Spark Driver log4j output, such as the log level, the timestamp, the thread name, the class name, the method name, and the message. Regex can be applied in various languages and frameworks, such as Scala, Python, Java, Spark SQL, and Databricks notebooks. References: https://docs.databricks.com/notebooks/notebooks-use.html#use-regular-expressions https://docs.databricks.com/spark/latest/spark-sql/udf-scala.html#using-regular-expressions-in-udfs https://docs.databricks.com/spark/latest/sparkr/functions/regexp_extract.html https://docs.databricks.com/spark/latest/sparkr/functions/regexp_replace.html

QUESTION 5

Although the Databricks Utilities Secrets module provides tools to store sensitive credentials and avoid accidentally displaying them in plain text users should still be careful with which credentials are stored here and which users have access to using these secrets.

Which statement describes a limitation of Databricks Secrets?

A. Because the SHA256 hash is used to obfuscate stored secrets, reversing this hash will display the value in plain text.

B. Account administrators can see all secrets in plain text by logging on to the Databricks Accounts console.

C. Secrets are stored in an administrators-only table within the Hive Metastore; database administrators have permission to query this table by default.

D. Iterating through a stored secret and printing each character will display secret contents in plain text.

E. The Databricks REST API can be used to list secrets in plain text if the personal access token has proper credentials.

Correct Answer: E

This is the correct answer because it describes a limitation of Databricks Secrets. Databricks Secrets is a module that

provides tools to store sensitive credentials and avoid accidentally displaying them in plain text. Databricks Secrets allows creating secret scopes, which are collections of secrets that can be accessed by users or groups. Databricks Secrets also allows creating and managing secrets using the Databricks CLI or the Databricks REST API. However, a limitation of Databricks Secrets is that the Databricks REST API can be used to list secrets in plain text if the personal access token has proper credentials. Therefore, users should still be careful with which credentials are stored in Databricks Secrets and which users have access to using these secrets. Verified References: [Databricks Certified Data Engineer Professional], under "Databricks Workspace" section; Databricks Documentation, under "List secrets" section.

QUESTION 6

A developer has successfully configured credential for Databricks Repos and cloned a remote Git repository. Hey don not have privileges to make changes to the main branch, which is the only branch currently visible in their workspace.

Use Response to pull changes from the remote Git repository commit and push changes to a branch that appeared as a changes were pulled.

A. Use Repos to merge all differences and make a pull request back to the remote repository.

B. Use repos to merge all difference and make a pull request back to the remote repository.

C. Use Repos to create a new branch commit all changes and push changes to the remote Git repertory.

D. Use repos to create a fork of the remote repository commit all changes and make a pull request on the source repository

Correct Answer: C

In Databricks Repos, when a user does not have privileges to make changes directly to the main branch of a cloned remote Git repository, the recommended approach is to create a new branch within the Databricks workspace. The developer can then make changes in this new branch, commit those changes, and push the new branch to the remote Git repository. This workflow allows for isolated development without affecting the main branch, enabling the developer to propose changes via a pull request from the new branch to the main branch in the remote repository. This method adheres to common Git collaboration workflows, fostering code review and collaboration while ensuring the integrity of the main branch. References: Databricks documentation on using Repos with Git: https://docs.databricks.com/repos.html

QUESTION 7

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

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.

In a Spark cluster, the driver node is responsible for managing the execution of the Spark application, including scheduling tasks, managing the execution plan, and interacting with the cluster manager. If the overall cluster CPU utilization is low (e.g., around 25%), it may indicate that the driver node is not utilizing the available resources effectively and might be a bottleneck.

QUESTION 8

A CHECK constraint has been successfully added to the Delta table named activity_details using the following logic:

```
ALTER TABLE activity_details
ADD CONSTRAINT valid_coordinates
CHECK (
  latitude >= -90 AND
  latitude <= 90 AND
  longitude >= -180 AND
  longitude <= 180);
```

A batch job is attempting to insert new records to the table, including a record where latitude = 45.50 and longitude = 212.67. Which statement describes the outcome of this batch insert?

A. The write will fail when the violating record is reached; any records previously processed will be recorded to the target table.

B. The write will fail completely because of the constraint violation and no records will be inserted into the target table.

C. The write will insert all records except those that violate the table constraints; the violating records will be recorded to a quarantine table.

D. The write will include all records in the target table; any violations will be indicated in the boolean column named valid_coordinates.

E. The write will insert all records except those that violate the table constraints; the violating records will be reported in a warning log.

Correct Answer: B

The CHECK constraint is used to ensure that the data inserted into the table meets the specified conditions. In this case, the CHECK constraint is used to ensure that the latitude and longitude values are within the specified range. If the data does not meet the specified conditions, the write operation will fail completely and no records will be inserted into the target table. This is because Delta Lake supports ACID transactions, which means that either all the data is written or none of it is written. Therefore, the batch insert will fail when it encounters a record that violates the constraint, and the target table will not be updated. References: Constraints: https://docs.delta.io/latest/delta-constraints.html ACID Transactions: https://docs.delta.io/latest/delta-intro.html#acid-transactions

QUESTION 9

A junior data engineer is migrating a workload from a relational database system to the Databricks Lakehouse. The source system uses a star schema, leveraging foreign key constrains and multi-table inserts to validate records on write.

Which consideration will impact the decisions made by the engineer while migrating this workload?

A. All Delta Lake transactions are ACID compliance against a single table, and Databricks does not enforce foreign key constraints.

B. Databricks only allows foreign key constraints on hashed identifiers, which avoid collisions in highly-parallel writes.

C. Foreign keys must reference a primary key field; multi-table inserts must leverage Delta Lake\\'s upsert functionality.

D. Committing to multiple tables simultaneously requires taking out multiple table locks and can lead to a state of deadlock.

Correct Answer: A

In Databricks and Delta Lake, transactions are indeed ACID-compliant, but this compliance is limited to single table transactions. Delta Lake does not inherently enforce foreign key constraints, which are a staple in relational database systems for maintaining referential integrity between tables. This means that when migrating workloads from a relational database system to Databricks Lakehouse, engineers need to reconsider how to maintain data integrity and relationships that were previously enforced by foreign key constraints. Unlike traditional relational databases where foreign key constraints help in maintaining the consistency across tables, in Databricks Lakehouse, the data engineer has to manage data consistency and integrity at the application level or through careful design of ETL processes.References: Databricks Documentation on Delta Lake: Delta Lake Guide Databricks Documentation on ACID Transactions in Delta Lake: ACID Transactions in Delta Lake

QUESTION 10

A data engineer is testing a collection of mathematical functions, one of which calculates the area under a curve as described by another function.

Which kind of the test does the above line exemplify?

A. Integration

B. Unit

C. Manual

D. functional

Correct Answer: B

A unit test is designed to verify the correctness of a small, isolated piece of code, typically a single function. Testing a mathematical function that calculates the area under a curve is an example of a unit test because it is testing a specific,

individual function to ensure it operates as expected.

References:

Software Testing Fundamentals: Unit Testing

QUESTION 11

A Spark job is taking longer than expected. Using the Spark UI, a data engineer notes that the Min, Median, and Max Durations for tasks in a particular stage show the minimum and median time to complete a task as roughly the same, but the max duration for a task to be roughly 100 times as long as the minimum.

Which situation is causing increased duration of the overall job?

A. Task queueing resulting from improper thread pool assignment.

B. Spill resulting from attached volume storage being too small.

C. Network latency due to some cluster nodes being in different regions from the source data

D. Skew caused by more data being assigned to a subset of spark-partitions.

E. Credential validation errors while pulling data from an external system.

Correct Answer: D

This is the correct answer because skew is a common situation that causes increased duration of the overall job. Skew occurs when some partitions have more data than others, resulting in uneven distribution of work among tasks and executors. Skew can be caused by various factors, such as skewed data distribution, improper partitioning strategy, or join operations with skewed keys. Skew can lead to performance issues such as long-running tasks, wasted resources, or even task failures due to memory or disk spills. Verified References: [Databricks Certified Data Engineer Professional], under "Performance Tuning" section; Databricks Documentation, under "Skew" section.

QUESTION 12

A Delta Lake table was created with the below query:

```
CREATE TABLE prod.sales_by_store
AS (
   SELECT *
   FROM prod.sales a
   INNER JOIN prod.store b
   ON a.store_id = b.store_id
)
```

Consider the following query:

DROP TABLE prod.sales_by_store-

If this statement is executed by a workspace admin, which result will occur?

A. Nothing will occur until a COMMIT command is executed.

B. The table will be removed from the catalog but the data will remain in storage.

C. The table will be removed from the catalog and the data will be deleted.

D. An error will occur because Delta Lake prevents the deletion of production data.

E. Data will be marked as deleted but still recoverable with Time Travel.

Correct Answer: C

When a table is dropped in Delta Lake, the table is removed from the catalog and the data is deleted. This is because Delta Lake is a transactional storage layer that provides ACID guarantees. When a table is dropped, the transaction log is updated to reflect the deletion of the table and the data is deleted from the underlying storage. References: https://docs.databricks.com/delta/quick-start.html#drop-a-table https://docs.databricks.com/delta/delta-batch.html#drop-table

QUESTION 13

A junior data engineer has been asked to develop a streaming data pipeline with a grouped aggregation using DataFrame df. 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 DataFrame df 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

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 1 second. 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. https://www.databricks.com/blog/2017/05/08/event-time-aggregation-watermarking-apache-sparks-structured-streaming.html

QUESTION 14

A data engineer is configuring a pipeline that will potentially see late-arriving, duplicate records.

In addition to de-duplicating records within the batch, which of the following approaches allows the data engineer to deduplicate data against previously processed records as it is inserted into a Delta table?

A. Set the configuration delta.deduplicate = true.

B. VACUUM the Delta table after each batch completes.

C. Perform an insert-only merge with a matching condition on a unique key.

D. Perform a full outer join on a unique key and overwrite existing data.

E. Rely on Delta Lake schema enforcement to prevent duplicate records.

Correct Answer: C

To deduplicate data against previously processed records as it is inserted into a Delta table, you can use the merge operation with an insert-only clause. This allows you to insert new records that do not match any existing records based on a unique key, while ignoring duplicate records that match existing records. For example, you can use the following syntax: MERGE INTO target_table USING source_table ON target_table.unique_key = source_table.unique_key WHEN NOT MATCHED THEN INSERT * This will insert only the records from the source table that have a unique key that is not present in the target table, and skip the records that have a matching key. This way, you can avoid inserting duplicate records into the Delta table. References: https://docs.databricks.com/delta/delta-update.html#upsert-into-a-table-using-merge https://docs.databricks.com/delta/delta-update.html#insert-only-merge

---

QUESTION 15

A table named user_ltv is being used to create a view that will be used by data analysis on various teams. Users in the workspace are configured into groups, which are used for setting up data access using ACLs. The user_ltv table has the following schema:

email STRING, age INT, ltv INT

The following view definition is executed:

```
CREATE VIEW user_ltv_no_minors AS
SELECT email, age, ltv
FROM user_ltv
WHERE
  CASE
    WHEN is_member("auditing") THEN TRUE
    ELSE age >= 18
  END
```

An analyze who is not a member of the auditing group executing the following query:

```
SELECT * FROM user_ltv_no_minors
```

Which result will be returned by this query?

A. All columns will be displayed normally for those records that have an age greater than 18; records not meeting this condition will be omitted.

B. All columns will be displayed normally for those records that have an age greater than 17; records not meeting this condition will be omitted.

C. All age values less than 18 will be returned as null values all other columns will be returned with the values in user_ltv.

D. All records from all columns will be displayed with the values in user_ltv.

Correct Answer: A

Given the CASE statement in the view definition, the result set for a user not in the auditing group would be constrained by the ELSE condition, which filters out records based on age. Therefore, the view will return all columns normally for records with an age greater than 18, as users who are not in the auditing group will not satisfy the

is_member(\\'auditing\\') condition. Records not meeting the age > 18 condition will not be displayed.