



Databricks

Exam Questions Databricks-Certified-Professional-Data-Engineer

Databricks Certified Data Engineer Professional Exam

About ExamBible

Your Partner of IT Exam

Found in 1998

ExamBible is a company specialized on providing high quality IT exam practice study materials, especially Cisco CCNA, CCDA, CCNP, CCIE, Checkpoint CCSE, CompTIA A+, Network+ certification practice exams and so on. We guarantee that the candidates will not only pass any IT exam at the first attempt but also get profound understanding about the certificates they have got. There are so many alike companies in this industry, however, ExamBible has its unique advantages that other companies could not achieve.

Our Advances

* 99.9% Uptime

All examinations will be up to date.

* 24/7 Quality Support

We will provide service round the clock.

* 100% Pass Rate

Our guarantee that you will pass the exam.

* Unique Gurantee

If you do not pass the exam at the first time, we will not only arrange FULL REFUND for you, but also provide you another exam of your claim, ABSOLUTELY FREE!

NEW QUESTION 1

A Databricks job has been configured with 3 tasks, each of which is a Databricks notebook. Task A does not depend on other tasks. Tasks B and C run in parallel, with each having a serial dependency on Task A.

If task A fails during a scheduled run, which statement describes the results of this run?

- A. Because all tasks are managed as a dependency graph, no changes will be committed to the Lakehouse until all tasks have successfully been completed.
- B. Tasks B and C will attempt to run as configured; any changes made in task A will be rolled back due to task failure.
- C. Unless all tasks complete successfully, no changes will be committed to the Lakehouse; because task A failed, all commits will be rolled back automatically.
- D. Tasks B and C will be skipped; some logic expressed in task A may have been committed before task failure.
- E. Tasks B and C will be skipped; task A will not commit any changes because of stage failure.

Answer: D

Explanation:

When a Databricks job runs multiple tasks with dependencies, the tasks are executed in a dependency graph. If a task fails, the downstream tasks that depend on it are skipped and marked as Upstream failed. However, the failed task may have already committed some changes to the Lakehouse before the failure occurred, and those changes are not rolled back automatically. Therefore, the job run may result in a partial update of the Lakehouse. To avoid this, you can use the transactional writes feature of Delta Lake to ensure that the changes are only committed when the entire job run succeeds.

Alternatively, you can use the Run if condition to configure tasks to run even when some or all of their dependencies have failed, allowing your job to recover from failures and

continue running. References:

? transactional writes: <https://docs.databricks.com/delta/delta-intro.html#transactional-writes>

? Run if: <https://docs.databricks.com/en/workflows/jobs/conditional-tasks.html>

NEW QUESTION 2

An upstream source writes Parquet data as hourly batches to directories named with the current date. A nightly batch job runs the following code to ingest all data from the previous day as indicated by the date variable:

```
(spark.read
  .format("parquet")
  .load(f"/mnt/raw_orders/{date}")
  .dropDuplicates(["customer_id", "order_id"])
  .write
  .mode("append")
  .saveAsTable("orders")
)
```

Assume that the fields customer_id and order_id serve as a composite key to uniquely identify each order.

If the upstream system is known to occasionally produce duplicate entries for a single order hours apart, which statement is correct?

- A. Each write to the orders table will only contain unique records, and only those records without duplicates in the target table will be written.
- B. Each write to the orders table will only contain unique records, but newly written records may have duplicates already present in the target table.
- C. Each write to the orders table will only contain unique records; if existing records with the same key are present in the target table, these records will be overwritten.
- D. Each write to the orders table will only contain unique records; if existing records with the same key are present in the target table, the operation will fail.
- E. Each write to the orders table will run deduplication over the union of new and existing records, ensuring no duplicate records are present.

Answer: B

Explanation:

This is the correct answer because the code uses the dropDuplicates method to remove any duplicate records within each batch of data before writing to the orders table. However, this method does not check for duplicates across different batches or in the target table, so it is possible that newly written records may have duplicates already present in the target table. To avoid this, a better approach would be to use Delta Lake and perform an upsert operation using mergeInto. Verified References: [Databricks Certified Data Engineer Professional], under "Delta Lake" section; Databricks Documentation, under "DROP DUPLICATES" section.

NEW QUESTION 3

A data ingestion task requires a one-TB JSON dataset to be written out to Parquet with a target part-file size of 512 MB. Because Parquet is being used instead of Delta Lake, built-in file-sizing features such as Auto-Optimize & Auto-Compaction cannot be used.

Which strategy will yield the best performance without shuffling data?

- A. Set spark.sql.files.maxPartitionBytes to 512 MB, ingest the data, execute the narrow transformations, and then write to parquet.
- B. Set spark.sql.shuffle.partitions to 2,048 partitions (1TB*1024*1024/512), ingest the data, execute the narrow transformations, optimize the data by sorting it (which automatically repartitions the data), and then write to parquet.
- C. Set spark.sql.adaptive.advisoryPartitionSizeInBytes to 512 MB bytes, ingest the data, execute the narrow transformations, coalesce to 2,048 partitions (1TB*1024*1024/512), and then write to parquet.
- D. Ingest the data, execute the narrow transformations, repartition to 2,048 partitions (1TB* 1024*1024/512), and then write to parquet.
- E. Set spark.sql.shuffle.partitions to 512, ingest the data, execute the narrow transformations, and then write to parquet.

Answer: B

Explanation:

The key to efficiently converting a large JSON dataset to Parquet files of a specific size without shuffling data lies in controlling the size of the output files directly.

? Setting spark.sql.files.maxPartitionBytes to 512 MB configures Spark to process

data in chunks of 512 MB. This setting directly influences the size of the part-files in the output, aligning with the target file size.

? Narrow transformations (which do not involve shuffling data across partitions) can then be applied to this data.

? Writing the data out to Parquet will result in files that are approximately the size specified by `spark.sql.files.maxPartitionBytes`, in this case, 512 MB.

? The other options involve unnecessary shuffles or repartitions (B, C, D) or an incorrect setting for this specific requirement (E).

References:

? Apache Spark Documentation: Configuration - `spark.sql.files.maxPartitionBytes`

? Databricks Documentation on Data Sources: Databricks Data Sources Guide

NEW QUESTION 4

The data architect has mandated that all tables in the Lakehouse should be configured as external Delta Lake tables.

Which approach will ensure that this requirement is met?

- A. Whenever a database is being created, make sure that the location keyword is used
- B. When configuring an external data warehouse for all table storag
- C. leverage Databricks for all ELT.
- D. Whenever a table is being created, make sure that the location keyword is used.
- E. When tables are created, make sure that the external keyword is used in the create table statement.
- F. When the workspace is being configured, make sure that external cloud object storage has been mounted.

Answer: C

Explanation:

This is the correct answer because it ensures that this requirement is met. The requirement is that all tables in the Lakehouse should be configured as external Delta Lake tables. An external table 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 by using the location keyword to specify the path to an existing directory in a cloud storage system, such as DBFS or S3. By creating external tables, the data engineering team can avoid losing data if they drop or overwrite the table, as well as leverage existing data without moving or copying it.

Verified References: [Databricks Certified Data Engineer Professional], under “Delta Lake” section; Databricks Documentation, under “Create an external table” section.

NEW QUESTION 5

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. pyspark.ml.feature
- D. Scala Datasets
- E. C++

Answer: A

Explanation:

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

NEW QUESTION 6

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.

Answer: C

Explanation:

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>

NEW QUESTION 7

A Databricks job has been configured with 3 tasks, each of which is a Databricks notebook. Task A does not depend on other tasks. Tasks B and C run in parallel, with each having a serial dependency on task A.

If tasks A and B complete successfully but task C fails during a scheduled run, which statement describes the resulting state?

- A. All logic expressed in the notebook associated with tasks A and B will have been successfully completed; some operations in task C may have completed

successfully.

B. All logic expressed in the notebook associated with tasks A and B will have been successfully completed; any changes made in task C will be rolled back due to task failure.

C. All logic expressed in the notebook associated with task A will have been successfully completed; tasks B and C will not commit any changes because of stage failure.

D. Because all tasks are managed as a dependency graph, no changes will be committed to the Lakehouse until all tasks have successfully been completed.

E. Unless all tasks complete successfully, no changes will be committed to the Lakehouse; because task C failed, all commits will be rolled back automatically.

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 resulting state after running the second command is that an external table will be created in the storage container mounted to /mnt/finance_eda_bucket with the new name prod.sales_by_store. The command will not change any data or move any files in the storage container; it will only update the table reference in the metastore and create a new Delta transaction log for the renamed table. Verified References: [Databricks Certified Data Engineer Professional], under “Delta Lake” section; Databricks Documentation, under “ALTER TABLE RENAME TO” section; Databricks Documentation, under “Create an external table” section.

NEW QUESTION 8

Which statement describes the default execution mode for Databricks Auto Loader?

A. New files are identified by listing the input directory; new files are incrementally and idempotently loaded into the target Delta Lake table.

B. Cloud vendor-specific queue storage and notification services are configured to track newly arriving files; new files are incrementally and idempotently into the target Delta Lake table.

C. Webhook trigger Databricks job to run anytime new data arrives in a source directory; new data automatically merged into target tables using rules inferred from the data.

D. New files are identified by listing the input directory; the target table is materialized by directory querying all valid files in the source directory.

Answer: A

Explanation:

Databricks Auto Loader simplifies and automates the process of loading data into Delta Lake. The default execution mode of the Auto Loader identifies new files by listing the input directory. It incrementally and idempotently loads these new files into the target Delta Lake table. This approach ensures that files are not missed and are processed exactly once, avoiding data duplication. The other options describe different mechanisms or integrations that are not part of the default behavior of the Auto Loader.

References:

? Databricks Auto Loader Documentation: Auto Loader Guide

? Delta Lake and Auto Loader: Delta Lake Integration

NEW QUESTION 9

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

Answer: A

NEW 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

Answer: B

Explanation:

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

NEW QUESTION 10

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.

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.

NEW QUESTION 13

A Delta Lake table was created with the below query:

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.

Answer: C

Explanation:

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>

NEW QUESTION 18

The business reporting team requires that data for their dashboards be updated every hour. The total processing time for the pipeline that extracts transforms and load the data for their pipeline runs in 10 minutes.

Assuming normal operating conditions, which configuration will meet their service-level agreement requirements with the lowest cost?

- A. Schedule a job to execute the pipeline once an hour on a dedicated interactive cluster.
- B. Schedule a Structured Streaming job with a trigger interval of 60 minutes.
- C. Schedule a job to execute the pipeline once an hour on a new job cluster.
- D. Configure a job that executes every time new data lands in a given directory.

Answer: C

Explanation:

Scheduling a job to execute the data processing pipeline once an hour on a new job cluster is the most cost-effective solution given the scenario. Job clusters are ephemeral in nature; they are spun up just before the job execution and terminated upon completion, which means you only incur costs for the time the cluster is active. Since the total processing time is only 10 minutes, a new job cluster created for each hourly execution minimizes the running time and thus the cost, while also fulfilling the requirement for hourly data updates for the business reporting team's dashboards.

References:

? Databricks documentation on jobs and job clusters: <https://docs.databricks.com/jobs.html>

NEW QUESTION 20

A junior member of the data engineering team is exploring the language interoperability of Databricks notebooks. The intended outcome of the below code is to register a view of all sales that occurred in countries on the continent of Africa that appear in the geo_lookup table.

Before executing the code, running SHOW TABLES on the current database indicates the database contains only two tables: geo_lookup and sales.

```
Cmd 1
%python
countries_af = [x[0] for x in
spark.table("geo_lookup").filter("continent='AF'").select("country").collect()]
```

```
Cmd 2
%sql
CREATE VIEW sales_af AS
SELECT *
FROM sales
WHERE city IN countries_af
AND CONTINENT = "AF"
```

Which statement correctly describes the outcome of executing these command cells in order in an interactive notebook?

- A. Both commands will succeed
- B. Executing show tables will show that countries at and sales at have been registered as views.
- C. Cmd 1 will succeed
- D. Cmd 2 will search all accessible databases for a table or view named countries af: if this entity exists, Cmd 2 will succeed.
- E. Cmd 1 will succeed and Cmd 2 will fail, countries at will be a Python variable representing a PySpark DataFrame.
- F. Both commands will fail
- G. No new variables, tables, or views will be created.
- H. Cmd 1 will succeed and Cmd 2 will fail, countries at will be a Python variable containing a list of strings.

Answer: E

Explanation:

This is the correct answer because Cmd 1 is written in Python and uses a list comprehension to extract the country names from the geo_lookup table and store them in a Python variable named countries af. This variable will contain a list of strings, not a PySpark DataFrame or a SQL view. Cmd 2 is written in SQL and tries to create a view named sales af by selecting from the sales table where city is in countries af. However, this command will fail because countries af is not a valid SQL entity and cannot be used in a SQL query. To fix this, a better approach would be to use spark.sql() to execute a SQL query in Python and pass the countries af variable as a parameter. Verified References: [Databricks Certified Data Engineer Professional], under “Language Interoperability” section; Databricks Documentation, under “Mix languages” section.

NEW QUESTION 25

A Delta Lake table representing metadata about content posts from users has the following schema:

user_id LONG, post_text STRING, post_id STRING, longitude FLOAT, latitude FLOAT, post_time TIMESTAMP, date DATE

This table is partitioned by the date column. A query is run with the following filter: longitude < 20 & longitude > -20

Which statement describes how data will be filtered?

- A. Statistics in the Delta Log will be used to identify partitions that might include files in the filtered range.
- B. No file skipping will occur because the optimizer does not know the relationship between the partition column and the longitude.
- C. The Delta Engine will use row-level statistics in the transaction log to identify the files that meet the filter criteria.
- D. Statistics in the Delta Log will be used to identify data files that might include records in the filtered range.
- E. The Delta Engine will scan the parquet file footers to identify each row that meets the filter criteria.

Answer: D

Explanation:

This is the correct answer because it describes how data will be filtered when a query is run with the following filter: longitude < 20 & longitude > -20. The query is run on a Delta Lake table that has the following schema: user_id LONG, post_text STRING, post_id STRING, longitude FLOAT, latitude FLOAT, post_time TIMESTAMP, date DATE. This table is partitioned by the date column. When a query is run on a partitioned Delta Lake table, Delta Lake uses statistics in the Delta Log to identify data files that might include records in the filtered range. The statistics include information such as min and max values for each column in each data file. By using these statistics, Delta Lake can skip reading data files that do not match the filter condition, which can improve query performance and reduce I/O costs. Verified References: [Databricks Certified Data Engineer Professional], under “Delta Lake” section; Databricks Documentation, under “Data skipping” section.

NEW QUESTION 29

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.

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.

NEW QUESTION 33

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

- A. Use &Pip 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 &sh install in a notebook cell

Answer: C

Explanation:

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

NEW QUESTION 36

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

Answer: D

Explanation:

The configuration that automatically recovers from query failures and keeps costs low is to use a new job cluster, set retries to unlimited, and set maximum concurrent runs to 1. This configuration has the following advantages:

? A new job cluster is a cluster that is created and terminated for each job run. This means that the cluster resources are only used when the job is running, and no idle costs are incurred. This also ensures that the cluster is always in a clean state and has the latest configuration and libraries for the job1.

? Setting retries to unlimited means that the job will automatically restart the query in case of any failure, such as network issues, node failures, or transient errors. This improves the reliability and availability of the streaming job, and avoids data loss or inconsistency2.

? Setting maximum concurrent runs to 1 means that only one instance of the job can run at a time. This prevents multiple queries from competing for the same resources or writing to the same output location, which can cause performance degradation or data corruption3.

Therefore, this configuration is the best practice for scheduling Structured Streaming jobs for production, as it ensures that the job is resilient, efficient, and consistent.

References: Job clusters, Job retries, Maximum concurrent runs

NEW QUESTION 38

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

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.

Answer: B

Explanation:

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>

NEW QUESTION 40

The data engineering team maintains a table of aggregate statistics through batch nightly updates. This includes total sales for the previous day alongside totals and averages for a variety of time periods including the 7 previous days, year-to-date, and quarter-to-date. This table is named store_sales_summary and the schema is as follows:

The table daily_store_sales contains all the information needed to update store_sales_summary. The schema for this table is: store_id INT, sales_date DATE, total_sales FLOAT If daily_store_sales is implemented as a Type 1 table and the total_sales column might be adjusted after manual data auditing, which approach is the safest to generate accurate reports in the store_sales_summary table?

- A. Implement the appropriate aggregate logic as a batch read against the daily_store_salestable and overwrite the store_sales_summary table with each Update.
- B. Implement the appropriate aggregate logic as a batch read against the daily_store_sales table and append new rows nightly to the store_sales_summary table.
- C. Implement the appropriate aggregate logic as a batch read against the daily_store_sales table and use upsert logic to update results in the store_sales_summary table.
- D. Implement the appropriate aggregate logic as a Structured Streaming read against the daily_store_sales table and use upsert logic to update results in the store_sales_summary table.
- E. Use Structured Streaming to subscribe to the change data feed for daily_store_sales and apply changes to the aggregates in the store_sales_summary table with each update.

Answer: E

Explanation:

The daily_store_sales table contains all the information needed to update store_sales_summary. The schema of the table is:

store_id INT, sales_date DATE, total_sales FLOAT

The daily_store_sales table is implemented as a Type 1 table, which means that old values are overwritten by new values and no history is maintained. The total_sales column might be adjusted after manual data auditing, which means that the data in the table may change over time.

The safest approach to generate accurate reports in the store_sales_summary table is to use Structured Streaming to subscribe to the change data feed for daily_store_sales and apply changes to the aggregates in the store_sales_summary table with each update. Structured Streaming is a scalable and fault-tolerant

stream processing engine built on Spark SQL. Structured Streaming allows processing data streams as if they were tables or DataFrames, using familiar operations such as select, filter, groupBy, or join. Structured Streaming also supports output modes that specify how to write the results of a streaming query to a sink, such as append, update, or complete. Structured Streaming can handle both streaming and batch data sources in a unified manner. The change data feed is a feature of Delta Lake that provides structured streaming sources that can subscribe to changes made to a Delta Lake table. The change data feed captures both data changes and schema changes as ordered events that can be processed by downstream applications or services. The change data feed can be configured with different options, such as starting from a specific version or timestamp, filtering by operation type or partition values, or excluding no-op changes. By using Structured Streaming to subscribe to the change data feed for daily_store_sales, one can capture and process any changes made to the total_sales column due to manual data auditing. By applying these changes to the aggregates in the store_sales_summary table with each update, one can ensure that the reports are always consistent and accurate with the latest data. Verified References: [Databricks Certified Data Engineer Professional], under “Spark Core” section; Databricks Documentation, under “Structured Streaming” section; Databricks Documentation, under “Delta Change Data Feed” section.

NEW QUESTION 43

A data team's Structured Streaming job is configured to calculate running aggregates for item sales to update a downstream marketing dashboard. The marketing team has introduced a new field to track the number of times this promotion code is used for each item. A junior data engineer suggests updating the existing query as follows: Note that proposed changes are in bold.

Original query:

```
df.groupBy("item")
  .agg(count("item").alias("total_count"),
       mean("sale_price").alias("avg_price"))
.writeStream
  .outputMode("complete")
  .option("checkpointLocation", "/item_agg/__checkpoint")
  .start("/item_agg")
```

Proposed query:

```
df.groupBy("item")
  .agg(count("item").alias("total_count"),
       mean("sale_price").alias("avg_price"),
       count("promo_code = 'NEW_MEMBER')".alias("new_member_promo"))
.writeStream
  .outputMode("complete")
  .option('mergeSchema', 'true')
  .option("checkpointLocation", "/item_agg/__checkpoint")
  .start("/item_agg")
```

Which step must also be completed to put the proposed query into production?

- A. Increase the shuffle partitions to account for additional aggregates
- B. Specify a new checkpointLocation
- C. Run REFRESH TABLE delta, /item_agg'
- D. Remove .option('mergeSchema', 'true') from the streaming write

Answer: B

Explanation:

When introducing a new aggregation or a change in the logic of a Structured Streaming query, it is generally necessary to specify a new checkpoint location. This is because the checkpoint directory contains metadata about the offsets and the state of the aggregations of a streaming query. If the logic of the query changes, such as including a new aggregation field, the state information saved in the current checkpoint would not be compatible with the new logic, potentially leading to incorrect results or failures. Therefore, to accommodate the new field and ensure the streaming job has the correct starting point and state information for aggregations, a new checkpoint location should be specified. References:

? Databricks documentation on Structured Streaming:

<https://docs.databricks.com/spark/latest/structured-streaming/index.html>

? Databricks documentation on streaming checkpoints: <https://docs.databricks.com/spark/latest/structured-streaming/production.html#checkpointing>

NEW QUESTION 45

The Databricks workspace administrator has configured interactive clusters for each of the data engineering groups. To control costs, clusters are set to terminate after 30 minutes of inactivity. Each user should be able to execute workloads against their assigned clusters at any time of the day.

Assuming users have been added to a workspace but not granted any permissions, which of the following describes the minimal permissions a user would need to start and attach to an already configured cluster.

- A. "Can Manage" privileges on the required cluster
- B. Workspace Admin privileges, cluster creation allowe
- C. "Can Attach To" privileges on the required cluster
- D. Cluster creation allowe
- E. "Can Attach To" privileges on the required cluster
- F. "Can Restart" privileges on the required cluster
- G. Cluster creation allowe
- H. "Can Restart" privileges on the required cluster

Answer: D

Explanation:

<https://learn.microsoft.com/en-us/azure/databricks/security/auth-authz/access-control/cluster-acl>

<https://docs.databricks.com/en/security/auth-authz/access-control/cluster-acl.html>

NEW QUESTION 48

A data engineer wants to join a stream of advertisement impressions (when an ad was shown) with another stream of user clicks on advertisements to correlate

when impression led to monetizable clicks.

```
In the code below, Impressions is a streaming DataFrame with a watermark ("event_time", "10 minutes")
.groupBy(
  window("event_time", "5 minutes"),
  "id")
.count()
).      withWatermark("event_time", 2 hours)
impressions.join(clicks, expr("clickId = impressionId"), "inner")
```

Which solution would improve the performance?

- A)
Joining on event time constraint: `clickTime == impressionTime` using a `leftOuter` join
- B)
Joining on event time constraint: `clickTime >= impressionTime - interval 3 hours` and removing watermarks
- C)
Joining on event time constraint: `clickTime + 3 hours < impressionTime - 2 hours`
- D)
Joining on event time constraint: `clickTime >= impressionTime AND clickTime <= impressionTime + interval 1 hour`

- A. Option A
B. Option B
C. Option C
D. Option D

Answer: A

Explanation:

When joining a stream of advertisement impressions with a stream of user clicks, you want to minimize the state that you need to maintain for the join. Option A suggests using a left outer join with the condition that `clickTime == impressionTime`, which is suitable for correlating events that occur at the exact same time. However, in a real-world scenario, you would likely need some leeway to account for the delay between an impression and a possible click. It's important to design the join condition and the window of time considered to optimize performance while still capturing the relevant user interactions. In this case, having the watermark can help with state management and avoid state growing unbounded by discarding old state data that's unlikely to match with new data.

NEW QUESTION 49

A data engineer, User A, has promoted a new pipeline to production by using the REST API to programmatically create several jobs. A DevOps engineer, User B, has configured an external orchestration tool to trigger job runs through the REST API. Both users authorized the REST API calls using their personal access tokens.

Which statement describes the contents of the workspace audit logs concerning these events?

- A. Because the REST API was used for job creation and triggering runs, a Service Principal will be automatically used to identity these events.
- B. Because User B last configured the jobs, their identity will be associated with both the job creation events and the job run events.
- C. Because these events are managed separately, User A will have their identity associated with the job creation events and User B will have their identity associated with the job run events.
- D. Because the REST API was used for job creation and triggering runs, user identity will not be captured in the audit logs.
- E. Because User A created the jobs, their identity will be associated with both the job creation events and the job run events.

Answer: C

Explanation:

The events are that a data engineer, User A, has promoted a new pipeline to production by using the REST API to programmatically create several jobs, and a DevOps engineer, User B, has configured an external orchestration tool to trigger job runs through the REST API. Both users authorized the REST API calls using their personal access tokens. The workspace audit logs are logs that record user activities in a Databricks workspace, such as creating, updating, or deleting objects like clusters, jobs, notebooks, or tables. The workspace audit logs also capture the identity of the user who performed each activity, as well as the time and details of the activity. Because these events are managed separately, User A will have their identity associated with the job creation events and User B will have their identity associated with the job run events in the workspace audit logs. Verified References: [Databricks Certified Data Engineer Professional], under "Databricks Workspace" section; Databricks Documentation, under "Workspace audit logs" section.

NEW QUESTION 50

A team of data engineer are adding tables to a DLT pipeline that contain repetitive expectations for many of the same data quality checks.

One member of the team suggests reusing these data quality rules across all tables defined for this pipeline.

What approach would allow them to do this?

- A. Maintain data quality rules in a Delta table outside of this pipeline's target schema, providing the schema name as a pipeline parameter.
- B. Use global Python variables to make expectations visible across DLT notebooks included in the same pipeline.
- C. Add data quality constraints to tables in this pipeline using an external job with access to pipeline configuration files.
- D. Maintain data quality rules in a separate Databricks notebook that each DLT notebook of file.

Answer: A

Explanation:

Maintaining data quality rules in a centralized Delta table allows for the reuse of these rules across multiple DLT (Delta Live Tables) pipelines. By storing these rules outside the pipeline's target schema and referencing the schema name as a pipeline parameter, the team can apply the same set of data quality checks to different tables within the pipeline. This approach ensures consistency in data quality validations and reduces redundancy in code by not having to replicate the same rules in each DLT notebook or file. References:

? Databricks Documentation on Delta Live Tables: Delta Live Tables Guide

NEW QUESTION 53

Spill occurs as a result of executing various wide transformations. However, diagnosing spill requires one to proactively look for key indicators. Where in the Spark UI are two of the primary indicators that a partition is spilling to disk?

- A. Stage's detail screen and Executor's files
- B. Stage's detail screen and Query's detail screen
- C. Driver's and Executor's log files
- D. Executor's detail screen and Executor's log files

Answer: B

Explanation:

In Apache Spark's UI, indicators of data spilling to disk during the execution of wide transformations can be found in the Stage's detail screen and the Query's detail screen. These screens provide detailed metrics about each stage of a Spark job, including information about memory usage and spill data. If a task is spilling data to disk, it indicates that the data being processed exceeds the available memory, causing Spark to spill data to disk to free up memory. This is an important performance metric as excessive spill can significantly slow down the processing.

References:

? Apache Spark Monitoring and Instrumentation: Spark Monitoring Guide

? Spark UI Explained: Spark UI Documentation

NEW QUESTION 55

The data engineer is using Spark's MEMORY_ONLY storage level.

Which indicators should the data engineer look for in the spark UI's Storage tab to signal that a cached table is not performing optimally?

- A. Size on Disk is> 0
- B. The number of Cached Partitions> the number of Spark Partitions
- C. The RDD Block Name included the " annotation signaling failure to cache
- D. On Heap Memory Usage is within 75% of off Heap Memory usage

Answer: C

Explanation:

In the Spark UI's Storage tab, an indicator that a cached table is not performing optimally would be the presence of the _disk annotation in the RDD Block Name. This annotation indicates that some partitions of the cached data have been spilled to disk because there wasn't enough memory to hold them. This is suboptimal because accessing data from disk is much slower than from memory. The goal of caching is to keep data in memory for fast access, and a spill to disk means that this goal is not fully achieved.

NEW QUESTION 57

A DLT pipeline includes the following streaming tables:

Raw_lot ingest raw device measurement data from a heart rate tracking device. Bgm_stats incrementally computes user statistics based on BPM measurements from raw_lot.

How can the data engineer configure this pipeline to be able to retain manually deleted or updated records in the raw_lot table while recomputing the downstream table when a pipeline update is run?

- A. Set the skipChangeCommits flag to true on bpm_stats
- B. Set the SkipChangeCommits flag to true raw_lot
- C. Set the pipelines, reset, allowed property to false on bpm_stats
- D. Set the pipelines, reset, allowed property to false on raw_lot

Answer: D

Explanation:

In Databricks Lakehouse, to retain manually deleted or updated records in the raw_lot table while recomputing downstream tables when a pipeline update is run, the property pipelines.reset.allowed should be set to false. This property prevents the system from resetting the state of the table, which includes the removal of the history of changes, during a pipeline update. By keeping this property as false, any changes to the raw_lot table, including manual deletes or updates, are retained, and recomputation of downstream tables, such as bpm_stats, can occur with the full history of data changes intact. References:

? Databricks documentation on DLT pipelines: <https://docs.databricks.com/data-engineering/delta-live-tables/delta-live-tables-overview.html>

NEW QUESTION 60

The DevOps team has configured a production workload as a collection of notebooks scheduled to run daily using the Jobs UI. A new data engineering hire is onboarding to the team and has requested access to one of these notebooks to review the production logic.

What are the maximum notebook permissions that can be granted to the user without allowing accidental changes to production code or data?

- A. Can manage
- B. Can edit
- C. Can run
- D. Can Read

Answer: D

Explanation:

Granting a user 'Can Read' permissions on a notebook within Databricks allows them to view the notebook's content without the ability to execute or edit it. This level of permission ensures that the new team member can review the production logic for learning or auditing purposes without the risk of altering the notebook's code or affecting production data and workflows. This approach aligns with best practices for maintaining security and integrity in production environments, where strict access controls are essential to prevent unintended modifications. References: Databricks documentation on access control and permissions for notebooks within the workspace (<https://docs.databricks.com/security/access-control/workspace-acl.html>).

NEW QUESTION 64

The data architect has mandated that all tables in the Lakehouse should be configured as external (also known as "unmanaged") Delta Lake tables.

Which approach will ensure that this requirement is met?

- A. When a database is being created, make sure that the LOCATION keyword is used.
- B. When configuring an external data warehouse for all table storage, leverage Databricks for all ELT.
- C. When data is saved to a table, make sure that a full file path is specified alongside the Delta format.
- D. When tables are created, make sure that the EXTERNAL keyword is used in the CREATE TABLE statement.
- E. When the workspace is being configured, make sure that external cloud object storage has been mounted.

Answer: D

Explanation:

To create an external or unmanaged Delta Lake table, you need to use the EXTERNAL keyword in the CREATE TABLE statement. This indicates that the table is not managed by the catalog and the data files are not deleted when the table is dropped. You also need to provide a LOCATION clause to specify the path where the data files are stored. For example:

```
CREATE EXTERNAL TABLE events ( date DATE, eventId STRING, eventType STRING, data STRING) USING DELTA LOCATION '/mnt/delta/events';
```

This creates an external Delta Lake table named events that references the data files in the '/mnt/delta/events' path. If you drop this table, the data files will remain intact and you can recreate the table with the same statement.

References:

? <https://docs.databricks.com/delta/delta-batch.html#create-a-table>

? <https://docs.databricks.com/delta/delta-batch.html#drop-a-table>

NEW QUESTION 66

The following code has been migrated to a Databricks notebook from a legacy workload:

```
%sh
git clone https://github.com/foo/data_loader;
python ./data_loader/run.py;
mv ./output /dbfs/mnt/new_data
```

The code executes successfully and provides the logically correct results, however, it takes over 20 minutes to extract and load around 1 GB of data.

Which statement is a possible explanation for this behavior?

- A. %sh triggers a cluster restart to collect and install Gi
- B. Most of the latency is related to cluster startup time.
- C. Instead of cloning, the code should use %sh pip install so that the Python code can get executed in parallel across all nodes in a cluster.
- D. %sh does not distribute file moving operations; the final line of code should be updated to use %fs instead.
- E. Python will always execute slower than Scala on Databrick
- F. The run.py script should be refactored to Scala.
- G. %sh executes shell code on the driver nod
- H. The code does not take advantage of the worker nodes or Databricks optimized Spark.

Answer: E

Explanation:

<https://www.databricks.com/blog/2020/08/31/introducing-the-databricks-web-terminal.html>

The code is using %sh to execute shell code on the driver node. This means that the code is not taking advantage of the worker nodes or Databricks optimized Spark. This is why the code is taking longer to execute. A better approach would be to use Databricks libraries and APIs to read and write data from Git and DBFS, and to leverage the parallelism and performance of Spark. For example, you can use the Databricks Connect feature to run your Python code on a remote Databricks cluster, or you can use the Spark Git Connector to read data from Git repositories as Spark DataFrames.

NEW QUESTION 70

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.

Answer: A

Explanation:

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>

NEW QUESTION 73

A Delta Lake table representing metadata about content from user has the following schema:
Based on the above schema, which column is a good candidate for partitioning the Delta Table?

- A. Date
- B. Post_id
- C. User_id
- D. Post_time

Answer: A

Explanation:

Partitioning a Delta Lake table improves query performance by organizing data into partitions based on the values of a column. In the given schema, the date column is a good candidate for partitioning for several reasons:

? Time-Based Queries: If queries frequently filter or group by date, partitioning by the date column can significantly improve performance by limiting the amount of data scanned.

? Granularity: The date column likely has a granularity that leads to a reasonable number of partitions (not too many and not too few). This balance is important for optimizing both read and write performance.

? Data Skew: Other columns like post_id or user_id might lead to uneven partition sizes (data skew), which can negatively impact performance.

Partitioning by post_time could also be considered, but typically date is preferred due to its more manageable granularity.

References:

? Delta Lake Documentation on Table Partitioning: Optimizing Layout with Partitioning

NEW QUESTION 76

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%

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.

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.

NEW QUESTION 81

Which statement describes integration testing?

- A. Validates interactions between subsystems of your application
- B. Requires an automated testing framework
- C. Requires manual intervention
- D. Validates an application use case
- E. Validates behavior of individual elements of your application

Answer: D

Explanation:

This is the correct answer because it describes integration testing. Integration testing is a type of testing that validates interactions between subsystems of your application, such as modules, components, or services. Integration testing ensures that the subsystems work together as expected and produce the correct outputs or results. Integration testing can be done at different levels of granularity, such as component integration testing, system integration testing, or end-to-end testing. Integration testing can help detect errors or bugs that may not be found by unit testing, which only validates behavior of individual elements of your application. Verified References: [Databricks Certified Data Engineer Professional], under “Testing” section; Databricks Documentation, under “Integration testing” section.

NEW QUESTION 83

The data engineering team has configured a job to process customer requests to be forgotten (have their data deleted). All user data that needs to be deleted is stored in Delta Lake tables using default table settings.

The team has decided to process all deletions from the previous week as a batch job at 1am each Sunday. The total duration of this job is less than one hour.

Every Monday at 3am, a batch job executes a series of VACUUM commands on all Delta Lake tables throughout the organization.

The compliance officer has recently learned about Delta Lake's time travel functionality. They are concerned that this might allow continued access to deleted data. Assuming all delete logic is correctly implemented, which statement correctly addresses this concern?

- A. Because the vacuum command permanently deletes all files containing deleted records, deleted records may be accessible with time travel for around 24 hours.
- B. Because the default data retention threshold is 24 hours, data files containing deleted records will be retained until the vacuum job is run the following day.
- C. Because Delta Lake time travel provides full access to the entire history of a table, deleted records can always be recreated by users with full admin privileges.

- D. Because Delta Lake's delete statements have ACID guarantees, deleted records will be permanently purged from all storage systems as soon as a delete job completes.
- E. Because the default data retention threshold is 7 days, data files containing deleted records will be retained until the vacuum job is run 8 days later.

Answer: E

Explanation:

<https://learn.microsoft.com/en-us/azure/databricks/delta/vacuum>

NEW QUESTION 88

The view updates represents an incremental batch of all newly ingested data to be inserted or updated in the customers table.

The following logic is used to process these records.

Which statement describes this implementation?

- A. The customers table is implemented as a Type 3 table; old values are maintained as a new column alongside the current value.
- B. The customers table is implemented as a Type 2 table; old values are maintained but marked as no longer current and new values are inserted.
- C. The customers table is implemented as a Type 0 table; all writes are append only with no changes to existing values.
- D. The customers table is implemented as a Type 1 table; old values are overwritten by new values and no history is maintained.
- E. The customers table is implemented as a Type 2 table; old values are overwritten and new customers are appended.

Answer: A

Explanation:

The logic uses the MERGE INTO command to merge new records from the view updates into the table customers. The MERGE INTO command takes two arguments: a target table and a source table or view. The command also specifies a condition to match records between the target and the source, and a set of actions to perform when there is a match or not. In this case, the condition is to match records by customer_id, which is the primary key of the customers table. The actions are to update the existing record in the target with the new values from the source, and set the current_flag to false to indicate that the record is no longer current; and to insert a new record in the target with the new values from the source, and set the current_flag to true to indicate that the record is current. This means that old values are maintained but marked as no longer current and new values are inserted, which is the definition of a Type 2 table. Verified References: [Databricks Certified Data Engineer Professional], under “Delta Lake” section; Databricks Documentation, under “Merge Into (Delta Lake on Databricks)” section.

NEW QUESTION 91

Which distribution does Databricks support for installing custom Python code packages?

- A. sbt
- B. CRAN
- C. CRAM
- D. nom
- E. Wheels
- F. jars

Answer: D

NEW QUESTION 94

A Databricks SQL dashboard has been configured to monitor the total number of records present in a collection of Delta Lake tables using the following query pattern:

SELECT COUNT (*) FROM table -

Which of the following describes how results are generated each time the dashboard is updated?

- A. The total count of rows is calculated by scanning all data files
- B. The total count of rows will be returned from cached results unless REFRESH is run
- C. The total count of records is calculated from the Delta transaction logs
- D. The total count of records is calculated from the parquet file metadata
- E. The total count of records is calculated from the Hive metastore

Answer: C

Explanation:

<https://delta.io/blog/2023-04-19-faster-aggregations-metadata/#:~:text=You%20can%20get%20the%20number,a%20given%20Delta%20table%20version.>

NEW QUESTION 96

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.

Answer: E

Explanation:

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.

NEW QUESTION 101

The data governance team is reviewing user for deleting records for compliance with GDPR. The following logic has been implemented to propagate deleted requests from the user_lookup table to the user_aggregate table.

```
(spark.read
  .format("delta")
  .option("readChangeData", True)
  .option("startingTimestamp", '2021-08-22 00:00:00')
  .option("endingTimestamp", '2021-08-29 00:00:00')
  .table("user_lookup")
  .createOrReplaceTempView("changes"))

spark.sql("""
  DELETE FROM user_aggregates
  WHERE user_id IN (
    SELECT user_id
    FROM changes
    WHERE _change_type='delete'
  )
""")
```

Assuming that user_id is a unique identifying key and that all users have requested deletion have been removed from the user_lookup table, which statement describes whether successfully executing the above logic guarantees that the records to be deleted from the user_aggregates table are no longer accessible and why?

- A. No: files containing deleted records may still be accessible with time travel until a VACUUM command is used to remove invalidated data files.
- B. Yes: Delta Lake ACID guarantees provide assurance that the DELETE command succeeded fully and permanently purged these records.
- C. No: the change data feed only tracks inserts and updates not deleted records.
- D. No: the Delta Lake DELETE command only provides ACID guarantees when combined with the MERGE INTO command

Answer: A

Explanation:

The DELETE operation in Delta Lake is ACID compliant, which means that once the operation is successful, the records are logically removed from the table. However, the underlying files that contained these records may still exist and be accessible via time travel to older versions of the table. To ensure that these records are physically removed and compliance with GDPR is maintained, a VACUUM command should be used to clean up these data files after a certain retention period. The VACUUM command will remove the files from the storage layer, and after this, the records will no longer be accessible.

NEW QUESTION 104

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.

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.

"Auto compaction occurs after a write to a table has succeeded and runs synchronously on the cluster that has performed the write. Auto compaction only compacts files that haven't been compacted previously."

<https://learn.microsoft.com/en-us/azure/databricks/delta/tune-file-size>

NEW QUESTION 108

A new data engineer notices that a critical field was omitted from an application that writes its Kafka source to Delta Lake. This happened even though the critical field was in the Kafka source. That field was further missing from data written to dependent, long-term storage. The retention threshold on the Kafka service is seven days. The pipeline has been in production for three months.

Which describes how Delta Lake can help to avoid data loss of this nature in the future?

- A. The Delta log and Structured Streaming checkpoints record the full history of the Kafka producer.
- B. Delta Lake schema evolution can retroactively calculate the correct value for newly added fields, as long as the data was in the original source.
- C. Delta Lake automatically checks that all fields present in the source data are included in the ingestion layer.
- D. Data can never be permanently dropped or deleted from Delta Lake, so data loss is not possible under any circumstance.

E. Ingesting all raw data and metadata from Kafka to a bronze Delta table creates a permanent, replayable history of the data state.

Answer: E

Explanation:

This is the correct answer because it describes how Delta Lake can help to avoid data loss of this nature in the future. By ingesting all raw data and metadata from Kafka to a bronze Delta table, Delta Lake creates a permanent, replayable history of the data state that can be used for recovery or reprocessing in case of errors or omissions in downstream applications or pipelines. Delta Lake also supports schema evolution, which allows adding new columns to existing tables without affecting existing queries or pipelines. Therefore, if a critical field was omitted from an application that writes its Kafka source to Delta Lake, it can be easily added later and the data can be reprocessed from the bronze table without losing any information. Verified References: [Databricks Certified Data Engineer Professional], under “Delta Lake” section; Databricks Documentation, under “Delta Lake core features” section.

NEW QUESTION 110

Each configuration below is identical to the extent that each cluster has 400 GB total of RAM, 160 total cores and only one Executor per VM. Given a job with at least one wide transformation, which of the following cluster configurations will result in maximum performance?

- A. • Total VMs: 1 • 400 GB per Executor • 160 Cores / Executor
- B. • Total VMs: 8 • 50 GB per Executor • 20 Cores / Executor
- C. • Total VMs: 4 • 100 GB per Executor • 40 Cores/Executor
- D. • Total VMs: 2 • 200 GB per Executor • 80 Cores / Executor

Answer: B

Explanation:

This is the correct answer because it is the cluster configuration that will result in maximum performance for a job with at least one wide transformation. A wide transformation is a type of transformation that requires shuffling data across partitions, such as join, groupBy, or orderBy. Shuffling can be expensive and time-consuming, especially if there are too many or too few partitions. Therefore, it is important to choose a cluster configuration that can balance the trade-off between parallelism and network overhead. In this case, having 8 VMs with 50 GB per executor and 20 cores per executor will create 8 partitions, each with enough memory and CPU resources to handle the shuffling efficiently. Having fewer VMs with more memory and cores per executor will create fewer partitions, which will reduce parallelism and increase the size of each shuffle block. Having more VMs with less memory and cores per executor will create more partitions, which will increase parallelism but also increase the network overhead and the number of shuffle files. Verified References: [Databricks Certified Data Engineer Professional], under “Performance Tuning” section; Databricks Documentation, under “Cluster configurations” section.

NEW QUESTION 115

.....

Relate Links

100% Pass Your Databricks-Certified-Professional-Data-Engineer Exam with ExamBible Prep Materials

<https://www.exambible.com/Databricks-Certified-Professional-Data-Engineer-exam/>

Contact us

We are proud of our high-quality customer service, which serves you around the clock 24/7.

Viste - <https://www.exambible.com/>