Databricks.Databricks-Certified-Data-Engineer-Professional.v2026-08-06.q122

Exam Code:	Databricks-Certified-Data-Engineer-Professional
Exam Name:	Databricks Certified Data Engineer Professional Exam
Certification Provider:	Databricks
Free Question Number:	122
Version:	v2026-08-06
# of views:	120
# of Questions views:	1220
https://www.freeqas.com/qa/Databricks/Databricks-Certified-Data-Engineer-Professional/Databricks.Databricks-Certified-Data-Engineer-Professional.v2026-08-06.q122.html	

NEW QUESTION: 1
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.

Which statement correctly describes the outcome of executing these command cells in order in an interactive notebook?
A. Both commands will succeed. Executing show tables will show that countries at and sales at have been registered as views.
B. Cmd 1 will succeed. Cmd 2 will search all accessible databases for a table or view named countries af: if this entity exists, Cmd 2 will succeed.
C. Cmd 1 will succeed and Cmd 2 will fail, countries at will be a Python variable representing a PySpark DataFrame.
D. Both commands will fail. No new variables, tables, or views will be created.
E. Cmd 1 will succeed and Cmd 2 will fail, countries at will be a Python variable containing a list of strings.
Answer: (SHOW ANSWER)
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.

NEW QUESTION: 2
A data engineer created a daily batch ingestion pipeline using a cluster with the latest DBR version to store banking transaction data, and persisted it in a MANAGED DELTA table called prod.gold.all_banking_transactions_daily. The data engineer is constantly receiving complaints from business users who query this table ad hoc through a SQL Serverless Warehouse about poor query performance. Upon analysis, the data engineer identified that these users frequently use high- cardinality columns as filters. The engineer now seeks to implement a data layout optimization technique that is incremental, easy to maintain, and can evolve over time.
Which command should the data engineer implement?
A. Alter the table to use Hive-Style Partitions + Z-ORDER and implement a periodic OPTIMIZE command.
B. Alter the table to use Liquid Clustering and implement a periodic OPTIMIZE command.
C. Alter the table to use Hive-Style Partitions and implement a periodic OPTIMIZE command.
D. Alter the table to use Z-ORDER and implement a periodic OPTIMIZE command.
Answer: B (LEAVE A REPLY)
Databricks recommends Liquid Clustering for optimizing data layout in large Delta tables where query filters involve high-cardinality columns. Liquid Clustering automatically manages file organization and supports incremental maintenance without the need to rewrite data when clustering keys evolve. This is a key advantage over static partitioning or Z-ordering, which require costly file rewrites whenever optimization keys change. By combining Liquid Clustering with a periodic OPTIMIZE command, Databricks automatically compacts small files and maintains efficient data skipping performance. As stated in the Delta Lake optimization guide, Liquid Clustering is designed for scalability, minimal maintenance, and adaptability for analytical workloads with evolving query patterns--making B the correct answer.

NEW QUESTION: 3
A data engineer is using Lakeflow Declarative Pipeline to propagate row deletions from a source bronze table (user_bronze) to a target silver table (user_silver). The engineer wants deletions in user_bronze to automatically delete corresponding rows in user_silver during pipeline execution.
Which configuration ensures deletions in the bronze table are propagated to the silver table?
A. Use apply_changes without CDF and filter rows where _soft_deleted is true.
B. Enable Change Data Feed (CDF) on user_bronze, read its CDF stream, and use apply_changes() with apply_as_deletes=True for user_silver.
C. Enable CDF on user_silver, read its transaction log, and use MERGE to sync deletions.
D. Configure VACUUM on user_bronze to delete files, then rebuild user_silver from scratch.
Answer: B (LEAVE A REPLY)
According to Databricks documentation, Change Data Feed (CDF) allows pipelines to read incremental data changes, including inserts, updates, and deletes, from a Delta table. When deletions occur in the source table, reading the CDF stream ensures downstream consumers receive the deletion records. The Lakeflow Declarative Pipelines API provides the apply_changes() function (or auto-CDC pipelines) with the apply_as_deletes parameter to correctly apply those deletions to the target table. This enables automatic synchronization between bronze and silver layers. Options A and D either require manual handling or complete rebuilds, and

C incorrectly applies CDF to the target rather than the source. Therefore, enabling CDF on the bronze table and using apply_as_deletes=True is the correct, Databricks-supported configuration.

NEW QUESTION: 4
The data engineering team maintains the following code:

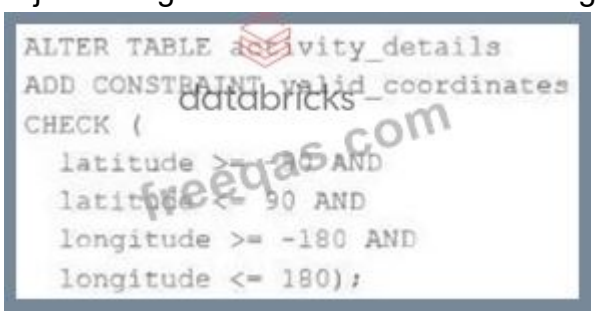
```
import pyspark.sql.functions as F

(spark.table("silver_customer_sales")
  .groupBy("customer_id")
  .agg(
    F.min("sale_date").alias("first_transaction_date"),
    F.max("sale_date").alias("last_transaction_date"),
    F.mean("sale_total").alias("average_sales"),
    F.countDistinct("order_id").alias("total_orders"),
    F.sum("sale_total").alias("lifetime_value")
  ).write
  .mode("overwrite")
  .table("gold_customer_lifetime_sales_summary")
```


Assuming that this code produces logically correct results and the data in the source table has been de-duplicated and validated, which statement describes what will occur when this code is executed?
A. The silver_customer_sales table will be overwritten by aggregated values calculated from all records in the gold_customer_lifetime_sales_summary table as a batch job.
B. A batch job will update the gold_customer_lifetime_sales_summary table, replacing only those rows that have different values than the current version of the table, using customer_id as the primary key.
C. The gold_customer_lifetime_sales_summary table will be overwritten by aggregated values calculated from all records in the silver_customer_sales table as a batch job.
D. An incremental job will leverage running information in the state store to update aggregate values in the gold_customer_lifetime_sales_summary table.
E. An incremental job will detect if new rows have been written to the silver_customer_sales table; if new rows are detected, all aggregates will be recalculated and used to overwrite the gold_customer_lifetime_sales_summary table.
Answer: (SHOW ANSWER)
This code is using the pyspark.sql.functions library to group the silver_customer_sales table by customer_id and then aggregate the data using the minimum sale date, maximum sale total, and sum of distinct order ids. The resulting aggregated data is then written to the gold_customer_lifetime_sales_summary table, overwriting any existing data in that table. This is a batch job that does not use any incremental or streaming logic, and does not perform any merge or update operations. Therefore, the code will overwrite the gold table with the aggregated values from the silver table every time it is executed.

NEW QUESTION: 5

The downstream consumers of a Delta Lake table have been complaining about data quality issues impacting performance in their applications. Specifically, they have complained that invalid latitude and longitude values in the activity_details table have been breaking their ability to use other geolocation processes.
A junior engineer has written the following code to add CHECK constraints to the Delta Lake table:



A senior engineer has confirmed the above logic is correct and the valid ranges for latitude and longitude are provided, but the code fails when executed.
Which statement explains the cause of this failure?
A. Because another team uses this table to support a frequently running application, two-phase locking is preventing the operation from committing.
B. The activity details table already exists; CHECK constraints can only be added during initial table creation.
C. The activity details table already contains records that violate the constraints; all existing data must pass CHECK constraints in order to add them to an existing table.
D. The activity details table already contains records; CHECK constraints can only be added prior to inserting values into a table.
E. The current table schema does not contain the field valid coordinates; schema evolution will need to be enabled before altering the table to add a constraint.
Answer: C (LEAVE A REPLY)
The failure is that the code to add CHECK constraints to the Delta Lake table fails when executed. The code uses ALTER TABLE ADD CONSTRAINT commands to add two CHECK constraints to a table named activity_details. The first constraint checks if the latitude value is between -90 and 90, and the second constraint checks if the longitude value is between -180 and 180. The cause of this failure is that the activity_details table already contains records that violate these constraints, meaning that they have invalid latitude or longitude values outside of these ranges. When adding CHECK constraints to an existing table, Delta Lake verifies that all existing data satisfies the constraints before adding them to the table. If any record violates the constraints, Delta Lake throws an exception and aborts the operation.

NEW QUESTION: 6
To identify the top users consuming compute resources, a data engineering team needs to monitor usage within their Databricks workspace for better resource utilization and cost control.
The team decided to use Databricks system tables, available under the System catalog in Unity Catalog, to gain detailed visibility into workspace activity. Which SQL query should the team run from the System catalog to achieve this?
A. SELECT sku_name,
identity_metadata.created_by AS user_email,
COUNT(usage_quantity) AS total_dbus
FROM system.billing.usage
GROUP BY user_email, sku_name
ORDER BY total_dbus DESC

LIMIT 10
B. SELECT identity_metadata.run_as AS user_email,
SUM(usage_quantity) AS total_dbus
FROM system.billing.usage
GROUP BY user_email
ORDER BY total_dbus DESC
LIMIT 10
C. SELECT sku_name,
identity_metadata.created_by AS user_email,
SUM(usage_quantity * usage_unit) AS total_dbus
FROM system.billing.usage
GROUP BY user_email, sku_name
ORDER BY total_dbus DESC
LIMIT 10
D. SELECT sku_name,
usage_metadata.run_name AS user_email,
SUM(usage_quantity) AS total_dbus
FROM system.billing.usage
GROUP BY user_email, sku_name
ORDER BY total_dbus DESC
LIMIT 10
Answer: (SHOW ANSWER)
The system.billing.usage table in the Unity Catalog System schema provides detailed usage metrics for each workload in the workspace. The field identity_metadata.run_as identifies the user or service principal under which the job or query executed. Summing usage_quantity provides total DBU (Databricks Unit) consumption per user. According to Databricks documentation, this table is the authoritative source for monitoring workspace cost drivers, showing compute SKU, user, and DBU consumption over time. Grouping by identity_metadata.run_as and summing usage_quantity produces the correct aggregation to determine top users. Other queries use non- existent or incorrect fields (created_by, run_name, or multiplied usage quantities), which do not reflect actual billing metrics.

NEW 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
Get Latest & Actual Certified-Data-Engineer-Professional Exam's Question and Answers from
E. Overall cluster CPU utilization is around 25%
Answer: E (LEAVE A REPLY)

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.

NEW QUESTION: 8
A distributed team of data analysts share computing resources on an interactive cluster with autoscaling configured. In order to better manage costs and query throughput, the workspace administrator is hoping to evaluate whether cluster upscaling is caused by many concurrent users or resource-intensive queries.
In which location can one review the timeline for cluster resizing events?
A. Workspace audit logs
B. Driver's log file
C. Ganglia
D. Cluster Event Log
E. Executor's log file
Answer: D (LEAVE A REPLY)
The Cluster Event Log in Databricks will show the timeline for cluster resizing events, including details about when and why a cluster was resized (scaled up or down). This log would help the workspace administrator determine the causes of cluster scaling, whether due to many concurrent users submitting jobs or a few users running resource-intensive queries.

NEW QUESTION: 9
Which statement describes the correct use of pyspark.sql.functions.broadcast?
A. It marks a column as having low enough cardinality to properly map distinct values to available partitions, allowing a broadcast join.
B. It marks a column as small enough to store in memory on all executors, allowing a broadcast join.
C. It caches a copy of the indicated table on attached storage volumes for all active clusters within a Databricks workspace.
D. It marks a DataFrame as small enough to store in memory on all executors, allowing a broadcast join.
E. It caches a copy of the indicated table on all nodes in the cluster for use in all future queries during the cluster lifetime.
Answer: D (LEAVE A REPLY)
https://spark.apache.org/docs/3.1.3/api/python/reference/api/pyspark.sql.functions.broadcast.html The broadcast function in PySpark is used in the context of joins. When you mark a DataFrame with broadcast, Spark tries to send this DataFrame to all worker nodes so that it can be joined with another DataFrame without shuffling the larger DataFrame across the nodes. This is particularly beneficial when the DataFrame is small enough to fit into the memory of each node. It helps to optimize the join process by reducing the amount of data that needs to be shuffled across the cluster, which can be a very expensive operation in terms of computation and time.
The pyspark.sql.functions.broadcast function in PySpark is used to hint to Spark that a DataFrame is small enough to be broadcast to all worker nodes in the cluster. When this hint is applied, Spark can perform a broadcast join, where the smaller DataFrame is sent to

each executor only once and joined with the larger DataFrame on each executor. This can significantly reduce the amount of data shuffled across the network and can improve the performance of the join operation. In a broadcast join, the entire smaller DataFrame is sent to each executor, not just a specific column or a cached version on attached storage. This function is particularly useful when one of the DataFrames in a join operation is much smaller than the other, and can fit comfortably in the memory of each executor node.

NEW QUESTION: 10
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 an extremely long-running job for which completion must be guaranteed, which cluster configuration will be able to guarantee completion of the job in light of one or more VM failures?
A. - Total VMs: 2
- 200 GB per Executor
- 80 Cores / Executor
B. - Total VMs: 4
- 100 GB per Executor
- 40 Cores / Executor
C. - Total VMs: 8
- 50 GB per Executor
- 20 Cores / Executor
D. - Total VMs: 16
- 25 GB per Executor
- 10 Cores / Executor
E. - Total VMs: 1
- 400 GB per Executor
- 160 Cores/Executor
Answer: D (LEAVE A REPLY)

NEW QUESTION: 11
Which statement regarding stream-static joins and static Delta tables is correct?
A. Each microbatch of a stream-static join will use the most recent version of the static Delta table as of each microbatch.
B. Each microbatch of a stream-static join will use the most recent version of the static Delta table as of the job's initialization.
C. The checkpoint directory will be used to track state information for the unique keys present in the join.
D. Stream-static joins cannot use static Delta tables because of consistency issues.
E. The checkpoint directory will be used to track updates to the static Delta table.
Answer: A (LEAVE A REPLY)
This is the correct answer because stream-static joins are supported by Structured Streaming when one of the tables is a static Delta table. A static Delta table is a Delta table that is not updated by any concurrent writes, such as appends or merges, during the execution of a streaming query. In this case, each microbatch of a stream-static join will use the most recent version of the static Delta table as of each microbatch, which means it will reflect any changes made to the static Delta table before the start of each microbatch.

NEW QUESTION: 12
A data governance team at a large enterprise is improving data discoverability across its organization. The team has hundreds of tables in their Databricks Lakehouse with thousands of columns that lack proper documentation. Many of these tables were created by different teams over several years, with missing context about column meanings and business logic. The data governance team needs to quickly generate comprehensive column descriptions for all existing tables to meet compliance requirements and improve data literacy across the organization. They want to leverage modern capabilities to automatically generate meaningful descriptions rather than manually documenting each column, which would take months to complete. Which approach should the team use in Databricks to automatically generate column comments and descriptions for existing tables?
A. Navigate to the table in Databricks Catalog Explorer, select the table schema view, and use the AI Generate option which leverages artificial intelligence to automatically create meaningful column descriptions based on column names, data types, sample values, and data patterns.
B. Use Delta Lake's DESCRIBE HISTORY command to analyze table evolution and infer column purposes from historical changes.
C. Use the DESCRIBE TABLE command to extract existing schema information and manually write descriptions based on column names and data types.
D. Write custom PySpark code using df.describe() and df.schema to programmatically generate basic statistical descriptions for each column.
Answer: (SHOW ANSWER)
The Catalog Explorer provides an AI-powered "AI Generate" capability that automatically creates intelligent column descriptions by analyzing column names, data types, sample values, and observed data patterns. This approach enables rapid, scalable documentation of existing tables, significantly improving data discoverability and compliance without manual effort.

NEW QUESTION: 13
Assuming that the Databricks CLI has been installed and configured correctly, which Databricks CLI command can be used to upload a custom Python Wheel to object storage mounted with the DBFS for use with a production job?
A. configure
B. fs
C. jobs
D. libraries
E. workspace
Answer: (SHOW ANSWER)
https://docs.databricks.com/en/archive/dev-tools/cli/dbfs-cli.html

NEW QUESTION: 14
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++
Answer: A (LEAVE A REPLY)

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.

NEW QUESTION: 15
Two data engineers are working on the same Databricks notebook in separate branches. Both have edited the same section of code. When one tries to merge the other's branch into their own using the Databricks Git folders UI, a merge conflict occurs on that notebook file. The UI highlights the conflict and presents options for resolution. How should the data engineers resolve this merge conflict using Databricks Git folders?
A. Abort the merge, discard all local changes, and try the merge operation again without reviewing the conflicting code.
B. Delete the conflicted notebook file via the Databricks workspace UI, commit the deletion, and recreate the notebook from scratch in a new commit to bypass the conflict entirely.
C. Use the Git CLI in the cluster's web terminal to force-push the conflicted merge (git push -force), overriding the remote branch with the local version and discarding changes.
D. Use the Git folders UI to manually edit the notebook file, selecting the desired lines from both versions and removing the conflict markers, then mark the conflict as resolved.
Answer: (SHOW ANSWER)
In the Databricks Git folders integration, when merge conflicts arise in notebooks, the UI provides a visual diff editor that highlights conflicting code segments. Users can manually choose which changes to keep from each branch, edit directly in the notebook UI, and remove conflict markers.
After resolving, the engineer must mark the conflict as resolved, save, and commit the final version.
This process ensures that both contributors' valid code segments are merged correctly and version history is maintained.
Forcing a push (C) or deleting notebooks (B) introduces data loss or versioning issues. Aborting without review (A) violates collaborative best practices. Therefore, D is the only correct and Databricks-approved way to resolve notebook merge conflicts.

NEW QUESTION: 16
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: (SHOW ANSWER)
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.

NEW QUESTION: 17
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: A (LEAVE A REPLY)
https://docs.databricks.com/en/clusters/cluster-config-best-practices.html

NEW QUESTION: 18
A data engineer is performing a join operation to combine values from a static userlookup table with a streaming DataFrame streamingDF.
Which code block attempts to perform an invalid stream-static join?
A. userLookup.join(streamingDF, ["userid"], how="inner")
B. streamingDF.join(userLookup, ["user_id"], how="outer")
C. streamingDF.join(userLookup, ["user_id"], how="left")
D. streamingDF.join(userLookup, ["userid"], how="inner")
E. userLookup.join(streamingDF, ["user_id"], how="right")
Answer: B (LEAVE A REPLY)
https://spark.apache.org/docs/latest/structured-streaming-programming-guide.html#support- matrix-for-joins-in-streaming-queries

NEW QUESTION: 19
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 allowed. "Can Attach To" privileges on the required cluster

C. Cluster creation allowed. "Can Attach To" privileges on the required cluster

D. "Can Restart" privileges on the required cluster

E. Cluster creation allowed. "Can Restart" privileges on the required cluster

Answer: (SHOW ANSWER)

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: 20

A table is registered with the following code:

```
CREATE TABLE recent_orders AS (
  SELECT a.user_id, a.email, b.order_id, b.order_date
  FROM
    (SELECT user_id, email
    FROM users) a
    INNER JOIN
    (SELECT user_id, order_id, order_date
    FROM orders
    WHERE order_date >= (current_date() - 7)) b
    ON a.user_id = b.user_id
)
```

Both users and orders are Delta Lake tables. Which statement describes the results of querying recent_orders?

A. All logic will execute at query time and return the result of joining the valid versions of the source tables at the time the query finishes.

B. All logic will execute when the table is defined and store the result of joining tables to the DBFS; this stored data will be returned when the table is queried.

C. Results will be computed and cached when the table is defined; these cached results will incrementally update as new records are inserted into source tables.

D. All logic will execute at query time and return the result of joining the valid versions of the source tables at the time the query began.

E. The versions of each source table will be stored in the table transaction log; query results will be saved to DBFS with each query.

Answer: B (LEAVE A REPLY)

Table is created and data of join will be stored on DBFS and it will be returned on query time.

NEW QUESTION: 21

A data engineering team is migrating off its legacy Hadoop platform. As part of the process, they are evaluating storage formats for performance comparison. The legacy platform uses ORC and RCFile formats. After converting a subset of data to Delta Lake, they noticed significantly better query performance. Upon investigation, they discovered that queries reading from Delta tables leveraged a

Shuffle Hash Join, whereas queries on legacy formats used Sort Merge Joins. The queries reading Delta Lake data also scanned less data. Which reason could be attributed to the difference in query performance?

A. Delta Lake enables data skipping and file pruning using a vectorized Parquet reader.

B. The queries against the Delta Lake tables were able to leverage the dynamic file pruning optimization.

C. Shuffle Hash Joins are always more efficient than Sort Merge Joins.

D. The queries against the ORC tables leveraged the dynamic data skipping optimization but not the dynamic file pruning optimization.

Answer: A (LEAVE A REPLY)

Delta Lake outperforms legacy Hadoop formats because it leverages Parquet-based storage, data skipping, and file pruning. According to Databricks documentation, Delta Lake automatically stores detailed statistics (min/max values and file-level metadata) in the transaction log. During query planning, the engine uses these statistics to skip entire files that do not match query filters, a process called data skipping and file pruning. Additionally, Delta uses a vectorized Parquet reader, which reduces I/O and CPU overhead. Together, these optimizations allow Delta to scan significantly less data and produce more efficient physical query plans (e.g., Shuffle Hash Join instead of Sort Merge Join). The performance gain is due to efficient data skipping, not the inherent superiority of join type.

NEW QUESTION: 22

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.
MERGE INTO customers
USING (
SELECT updates.customer_id as merge_ey, updates .*
FROM updates
UNION ALL
SELECT NULL as merge_key, updates .*
FROM updates JOIN customers
ON updates.customer_id = customers.customer_id
WHERE customers.current = true AND updates.address <> customers.address) staged_updates ON customers.customer_id = mergekey WHEN MATCHED AND customers. current = true AND customers.address <> staged_updates.address THEN UPDATE SET current = false, end_date = staged_updates.effective_date WHEN NOT MATCHED THEN INSERT (customer_id, address, current, effective_date, end_date) VALUES (staged_updates.customer_id, staged_updates.address, true, staged_updates.effective_date, null) Which statement describes this implementation?

A. The customers table is implemented as a Type 2 table; old values are overwritten and new customers are appended.

B. The customers table is implemented as a Type 1 table; old values are overwritten by new values and no history is maintained.

C. 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.

D. The customers table is implemented as a Type 0 table; all writes are append only with no changes to existing values.

Answer: C (LEAVE A REPLY)

The provided MERGE statement is a classic implementation of a Type 2 SCD in a data warehousing context. In this approach, historical data is preserved by keeping old records (marking them as not current) and adding new records for changes. Specifically, when a match is found and there's a change in the address, the existing record in the customers table is updated to mark it as no longer current (current = false), and an end date is assigned (end_date = staged_updates.effective_date). A new record for the

customer is then inserted with the updated information, marked as current. This method ensures that the full history of changes to customer information is maintained in the table, allowing for time-based analysis of customer data.

NEW QUESTION: 23
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: A (LEAVE A REPLY)
Integration testing is a type of software testing where components of the software are gradually integrated and then tested as a unified group.

NEW QUESTION: 24
A data architect is designing a Databricks solution to efficiently process data for different business requirements. In which scenario should a data engineer use a materialized view compared to a streaming table?
A. Implementing a CDC (Change Data Capture) pipeline that needs to detect and respond to database changes within seconds.
B. Ingesting data from Apache Kafka topics with sub-second processing requirements for immediate alerting.
C. Precomputing complex aggregations and joins from multiple large tables to accelerate BI dashboard performance.
D. Processing high-volume, continuous clickstream data from a website to monitor user behavior in real-time.
Answer: C (LEAVE A REPLY)
Materialized views in Databricks are optimized for precomputing and caching results of complex SQL queries, joins, and aggregations. They store query outputs physically and automatically refresh on a schedule or incremental change basis, drastically improving BI dashboard performance and reducing compute costs.
Conversely, streaming tables are designed for real-time data ingestion and processing, enabling event-driven analytics and low-latency use cases.
Databricks documentation explicitly recommends materialized views for analytical workloads with periodic updates and streaming tables for continuously updating sources. Therefore, the correct choice is C, where complex aggregations from large tables benefit most from materialized precomputation for fast reporting.

NEW QUESTION: 25
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.
Answer: B (LEAVE A REPLY)
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.

NEW QUESTION: 26
A workspace admin has created a new catalog called finance_data and wants to delegate permission management to a finance team lead without giving them full admin rights. Which privilege should be granted to the finance team lead?
A. ALL PRIVILEGES on the finance_data catalog.
B. Make the finance team lead a metastore admin.
C. GRANT OPTION privilege on the finance_data catalog.
D. MANAGE privilege on the finance_data catalog.
Answer: D (LEAVE A REPLY)
The MANAGE privilege in Unity Catalog provides the ability to grant and revoke privileges on the specified object (in this case, a catalog) without giving full administrative access or ownership.
This is the Databricks-recommended approach for delegating governance responsibilities while preserving the principle of least privilege.
By contrast, the ALL PRIVILEGES option grants excessive access (including read and write permissions), and metastore admin status provides global control over all catalogs--far exceeding the requirement. The MANAGE privilege enables the finance team lead to control access to objects within finance_data responsibly while limiting overall administrative exposure.

NEW QUESTION: 27
Two of the most common data locations on Databricks are the DBFS root storage and external object storage mounted with dbutils.fs.mount().
Which of the following statements is correct?
A. DBFS is a file system protocol that allows users to interact with files stored in object storage using syntax and guarantees similar to Unix file systems.
B. By default, both the DBFS root and mounted data sources are only accessible to workspace administrators.
C. The DBFS root is the most secure location to store data, because mounted storage volumes must have full public read and write permissions.

D. Neither the DBFS root nor mounted storage can be accessed when using %sh in a Databricks notebook.
E. The DBFS root stores files in ephemeral block volumes attached to the driver, while mounted directories will always persist saved data to external storage between sessions.
Answer: A (LEAVE A REPLY)
DBFS is a file system protocol that allows users to interact with files stored in object storage using syntax and guarantees similar to Unix file systems. DBFS is not a physical file system, but a layer over the object storage that provides a unified view of data across different data sources. By default, the DBFS root is accessible to all users in the workspace, and the access to mounted data sources depends on the permissions of the storage account or container. Mounted storage volumes do not need to have full public read and write permissions, but they do require a valid connection string or access key to be provided when mounting. Both the DBFS root and mounted storage can be accessed when using %sh in a Databricks notebook, as long as the cluster has FUSE enabled. The DBFS root does not store files in ephemeral block volumes attached to the driver, but in the object storage associated with the workspace. Mounted directories will persist saved data to external storage between sessions, unless they are unmounted or deleted.

NEW QUESTION: 28
A data engineering team is collaborating on a Databricks project where each team member needs to develop and test code independently before merging changes into the main branch.
They want to avoid accidental overwrites or branch switching issues while ensuring that all work is version- controlled and can be integrated into their CI/CD pipeline.
How should the data engineer achieve collaboration?
A. Each team member creates their own Databricks Git folder, mapped to the same remote Git repository, and works in their own development branch within their personal folder.
B. All team members work in the same Databricks Git folder and perform Git operations (pull, push, commit, branch switching) directly in that shared folder.
C. Team members edit notebooks directly in the workspace's shared folder and periodically copy changes into a Git folder for version control.
D. Team members use the Databricks CLI to clone the Git repository and perform Git operations from a cluster's web terminal.
Answer: A (LEAVE A REPLY)
Using separate Databricks Git folders per user mapped to the same remote repository allows each team member to work independently on their own branch without interfering with others.
This prevents accidental overwrites or branch conflicts while ensuring all changes are version- controlled and easily integrated into CI/CD workflows.

NEW QUESTION: 29
An upstream system is emitting change data capture (CDC) logs that are being written to a cloud object storage directory. Each record in the log indicates the change type (insert, update, or delete) and the values for each field after the change. The source table has a primary key identified by the field pk_id.
For auditing purposes, the data governance team wishes to maintain a full record of all values that have ever been valid in the source system. For analytical purposes, only the most recent value for each record needs to be recorded. The Databricks job to ingest these records occurs once per hour, but each individual record may have changed multiple times over the course of an hour.
Which solution meets these requirements?

A. Create a separate history table for each pk_id resolve the current state of the table by running a union all filtering the history tables for the most recent state.
B. Use merge into to insert, update, or delete the most recent entry for each pk_id into a bronze table, then propagate all changes throughout the system.
C. Iterate through an ordered set of changes to the table, applying each in turn; rely on Delta Lake's versioning ability to create an audit log.
D. Use Delta Lake's change data feed to automatically process CDC data from an external system, propagating all changes to all dependent tables in the Lakehouse.
E. Ingest all log information into a bronze table; use merge into to insert, update, or delete the most recent entry for each pk_id into a silver table to recreate the current table state.
Answer: E (LEAVE A REPLY)
CDF captures changes only from a Delta table and is only forward-looking once enabled. The CDC logs are writing to object storage. So you would need to ingestion those and merge into downstream tables.

NEW QUESTION: 30
A nightly batch job is configured to ingest all data files from a cloud object storage container where records are stored in a nested directory structure YYYY/MM/DD. The data for each date represents all records that were processed by the source system on that date, noting that some records may be delayed as they await moderator approval. Each entry represents a user review of a product and has the following schema:
user_id STRING, review_id BIGINT, product_id BIGINT, review_timestamp TIMESTAMP, review_text STRING The ingestion job is configured to append all data for the previous date to a target table reviews_raw with an identical schema to the source system. The next step in the pipeline is a batch write to propagate all new records inserted into reviews_raw to a table where data is fully deduplicated, validated, and enriched.
Which solution minimizes the compute costs to propagate this batch of data?
Get Latest & Actual Certified-Data-Engineer-Professional Exam's Question and Answers from
A. Perform a batch read on the reviews_raw table and perform an insert-only merge using the natural composite key user_id, review_id, product_id, review_timestamp.
B. Configure a Structured Streaming read against the reviews_raw table using the trigger once execution mode to process new records as a batch job.
C. Use Delta Lake version history to get the difference between the latest version of reviews_raw and one version prior, then write these records to the next table.
D. Filter all records in the reviews_raw table based on the review_timestamp; batch append those records produced in the last 48 hours.
E. Reprocess all records in reviews_raw and overwrite the next table in the pipeline.
Answer: B (LEAVE A REPLY)
https://www.databricks.com/blog/2017/05/22/running-streaming-jobs-day-10x-cost-savings.html

NEW QUESTION: 31
The data engineering team maintains the following code:
Get Latest & Actual Certified-Data-Engineer-Professional Exam's Question and Answers from

```
rderDF = spark.table("orders")
temDF = spark.table("items")

rderWithItemDF = (orderDF.join(
   itemDF,
   orderDF.itemID == itemDF.itemID)
 .select(
   orderDF.accountID,
   orderDF.itemID,

   itemDF.itemName))

inalDF = (accountDF.join(
   orderWithItemDF,
   accountDF.accountID == orderWithItemDF.accountID
 .select(
   orderWithItemDF["*"],

   accountDF.city))

finalDF.write
 .mode("overwrite")
 .table("enriched_itemized_orders_by_account"))
```


Assuming that this code produces logically correct results and the data in the source tables has been de-duplicated and validated, which statement describes what will occur when this code is executed?

A. A batch job will update the enriched_itemized_orders_by_account table, replacing only those rows that have different values than the current version of the table, using accountID as the primary key.

B. The enriched_itemized_orders_by_account table will be overwritten using the current valid version of data in each of the three tables referenced in the join logic.

C. An incremental job will leverage information in the state store to identify unjoined rows in the source tables and write these rows to the enriched_iteinized_orders_by_account table.

D. An incremental job will detect if new rows have been written to any of the source tables; if new rows are detected, all results will be recalculated and used to overwrite the enriched_itemized_orders_by_account table.

E. No computation will occur until enriched_itemized_orders_by_account is queried; upon query materialization, results will be calculated using the current valid version of data in each of the three tables referenced in the join logic.

Answer: B (LEAVE A REPLY)

This is the correct answer because it describes what will occur when this code is executed. The Get Latest & Actual Certified-Data-Engineer-Professional Exam's Question and Answers from code uses three Delta Lake tables as input sources: accounts, orders, and order_items. These tables are joined together using SQL queries to create a view called new_enriched_itemized_orders_by_account, which contains information about each order item and its associated account details. Then, the code uses write.format("delta").mode("overwrite") to overwrite a target table called enriched_itemized_orders_by_account using the data from the view. This means that every time this code is executed, it will replace all existing data in the target table with new data based on the current valid version of data in each of the three input tables.

NEW QUESTION: 32
A small company based in the United States has recently contracted a consulting firm in India to implement several new data engineering pipelines to power artificial intelligence applications. All the company's data is stored in regional cloud storage in the United States.
The workspace administrator at the company is uncertain about where the Databricks workspace used by the contractors should be deployed.
Assuming that all data governance considerations are accounted for, which statement accurately informs this decision?
A. Databricks runs HDFS on cloud volume storage; as such, cloud virtual machines must be deployed in the region where the data is stored.
B. Databricks workspaces do not rely on any regional infrastructure; as such, the decision should be Get Latest & Actual Certified-Data-Engineer-Professional Exam's Question and Answers from made based upon what is most convenient for the workspace administrator.
C. Cross-region reads and writes can incur significant costs and latency; whenever possible, compute should be deployed in the same region the data is stored.
D. Databricks leverages user workstations as the driver during interactive development; as such, users should always use a workspace deployed in a region they are physically near.
E. Databricks notebooks send all executable code from the user's browser to virtual machines over the open internet; whenever possible, choosing a workspace region near the end users is the most secure.
Answer: C (LEAVE A REPLY)
This is the correct answer because it accurately informs this decision. The decision is about where the Databricks workspace used by the contractors should be deployed. The contractors are based in India, while all the company's data is stored in regional cloud storage in the United States. When choosing a region for deploying a Databricks workspace, one of the important factors to consider is the proximity to the data sources and sinks. Cross-region reads and writes can incur significant costs and latency due to network bandwidth and data transfer fees.
Therefore, whenever possible, compute should be deployed in the same region the data is stored to optimize performance and reduce costs.

NEW QUESTION: 33
Which configuration parameter directly affects the size of a spark-partition upon ingestion of data into Spark?
A. spark.sql.files.maxPartitionBytes
B. spark.sql.autoBroadcastJoinThreshold
C. spark.sql.files.openCostInBytes

D. spark.sql.adaptive.coalescePartitions.minPartitionNum
E. spark.sql.adaptive.advisoryPartitionSizeInBytes
Answer: A (LEAVE A REPLY)
Get Latest & Actual Certified-Data-Engineer-Professional Exam's Question and Answers from This is the correct answer because spark.sql.files.maxPartitionBytes is a configuration parameter that directly affects the size of a spark-partition upon ingestion of data into Spark. This parameter configures the maximum number of bytes to pack into a single partition when reading files from file- based sources such as Parquet, JSON and ORC. The default value is 128 MB, which means each partition will be roughly 128 MB in size, unless there are too many small files or only one large file.

NEW QUESTION: 34
An upstream system is emitting change data capture (CDC) logs that are being written to a cloud object storage directory. Each record in the log indicates the change type (insert, update, or delete) and the values for each field after the change. The source table has a primary key identified by the field pk_id.
For auditing purposes, the data governance team wishes to maintain a full record of all values that have ever been valid in the source system. For analytical purposes, only the most recent value for each record needs to be recorded. The Databricks job to ingest these records occurs once per hour, but each individual record may have changed multiple times over the course of an hour.
Which solution meets these requirements?
A. Create a separate history table for each pk_id resolve the current state of the table by running a Get Latest & Actual Certified-Data-Engineer-Professional Exam's Question and Answers from union all filtering the history tables for the most recent state.
B. Use merge into to insert, update, or delete the most recent entry for each pk_id into a bronze table, then propagate all changes throughout the system.
C. Iterate through an ordered set of changes to the table, applying each in turn; rely on Delta Lake's versioning ability to create an audit log.
D. Use Delta Lake's change data feed to automatically process CDC data from an external system, propagating all changes to all dependent tables in the Lakehouse.
E. Ingest all log information into a bronze table; use merge into to insert, update, or delete the most recent entry for each pk_id into a silver table to recreate the current table state.
Answer: E (LEAVE A REPLY)
CDF captures changes only from a Delta table and is only forward-looking once enabled. The CDC logs are writing to object storage. So you would need to ingestion those and merge into downstream tables.

NEW QUESTION: 35
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 (LEAVE A REPLY)

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.

NEW QUESTION: 36

What statement is true regarding the retention of job run history?

A. It is retained until you export or delete job run logs

B. It is retained for 30 days, during which time you can deliver job run logs to DBFS or S3

C. It is retained for 60 days, during which you can export notebook run results to HTML

D. It is retained for 60 days, after which logs are archived

E. It is retained for 90 days or until the run-id is re-used through custom run configuration

Answer: C (LEAVE A REPLY)

https://docs.databricks.com/en/workflows/jobs/monitor-job-runs.html

NEW QUESTION: 37

The data science team has created and logged a production model using MLflow. The following code correctly imports and applies the production model to output the predictions as a new DataFrame named preds with the schema "customer_id LONG, predictions DOUBLE, date DATE".

```
from pyspark.sql.functions import current_date

model = mlflow.pyfunc.spark_udf(spark, model_uri="models:/churn/prod")
df = spark.table("customers")
columns = ["account_age", "time_since_last_seen", "app_rating"]
preds = (df.select(
  "customer_id",
  model(*columns).alias("predictions"),
  current_date().alias("date")
  )
```

The data science team would like predictions saved to a Delta Lake table with the ability to compare all predictions across time. Churn predictions will be made at most once per day.

Which code block accomplishes this task while minimizing potential compute costs?

A.

```
(preds.writeStream
  .outputMode("append")
  .option("checkpointPath", "/_checkpoints/churn_preds")
  .table("churn_preds")
)
```

B. preds.write.mode("append").saveAsTable("churn_preds")

```
(preds.writeStream
  .outputMode("overwrite")
  .option("checkpointPath", "/_checkpoints/churn_preds")
  .start("/preds/churn_preds")
)
```

C.

D. preds.write.format("delta").save("/preds/churn_preds")

E.

Answer: B (LEAVE A REPLY)

NEW QUESTION: 38

A departing platform owner currently holds ownership of multiple catalogs and controls storage credentials and external locations. A data engineer has been asked to ensure continuity: transfer catalog ownership to the platform team group, delegate ongoing privilege management, and retain the ability to receive and share data via Delta Sharing. Which role must be in place to perform these actions across the metastore?

A. Metastore Admin, because metastore admins can transfer ownership and manage privileges across all metastore objects, including shares and recipients.

B. Account Admin, because account admins can only create metastores but cannot change ownership of catalogs.

C. Workspace Admin, because workspace admins can transfer ownership of any Unity Catalog object.

D. Catalog Owner, because catalog owners can transfer any object in any catalog in the metastore.

Answer: A (LEAVE A REPLY)

Metastore Admins have the highest administrative privileges within a Unity Catalog metastore.

They can transfer ownership of any Unity Catalog object, including catalogs, schemas, tables, storage credentials, and external locations. Metastore Admins are also required to manage Delta Sharing configurations such as creating or transferring shares and recipients.

Account Admins, by contrast, only create metastores and cannot change ownership or manage Delta Sharing objects. Workspace Admins have privileges limited to workspace-level management, not cross-metastore access.

NEW QUESTION: 39

A data engineer is optimizing a MERGE operation on an 800GB UC-managed table that experiences frequent updates and deletions. Which two actions should the engineer prioritize to improve MERGE performance? (Choose two.)

A. Apply liquid clustering using the merge join keys.

B. Enable deletion vectors on the table if not already enabled.

C. Partition the table by date.

D. Use ZORDER on high-cardinality columns.

E. Overwrite the table instead of Merge.

Answer: A,B (LEAVE A REPLY)

Liquid clustering on the merge join keys improves data locality and reduces the amount of data scanned during MERGE operations, which is especially effective for large, frequently updated tables. Enabling deletion vectors avoids rewriting entire Parquet files for updates and deletes, significantly reducing I/O and improving MERGE performance on Unity Catalog-managed tables.

NEW QUESTION: 40
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 an extremely long-running job for which completion must be guaranteed, which cluster configuration will be able to guarantee completion of the job in light of one or more VM failures?
A. - Total VMs: 8
- 50 GB per Executor
- 20 Cores / Executor
B. - Total VMs: 16
- 25 GB per Executor

- 10 Cores / Executor
C. - Total VMs: 1
- 400 GB per Executor
- 160 Cores/Executor
D. - Total VMs: 4
- 100 GB per Executor
- 40 Cores / Executor
E. - Total VMs: 2
- 200 GB per Executor
- 80 Cores / Executor
Answer: B (LEAVE A REPLY)

NEW QUESTION: 41
A junior data engineer is working to implement logic for a Lakehouse table named silver_device_recordings. The source data contains 100 unique fields in a highly nested JSON structure.
The silver_device_recordings table will be used downstream for highly selective joins on a number of fields, and will also be leveraged by the machine learning team to filter on a handful of relevant fields, in total, 15 fields have been identified that will often be used for filter and join logic.
The data engineer is trying to determine the best approach for dealing with these nested fields before declaring the table schema.
Which of the following accurately presents information about Delta Lake and Databricks that may Impact their decision-making process?
A. Because Delta Lake uses Parquet for data storage, Dremel encoding information for nesting can be directly referenced by the Delta transaction log.
B. Tungsten encoding used by Databricks is optimized for storing string data: newly-added native support for querying JSON strings means that string types are always most efficient.

C. Schema inference and evolution on Databricks ensure that inferred types will always accurately match the data types used by downstream systems.
D. By default Delta Lake collects statistics on the first 32 columns in a table; these statistics are leveraged for data skipping when executing selective queries.
Answer: D (LEAVE A REPLY)
Delta Lake, built on top of Parquet, enhances query performance through data skipping, which is based on the statistics collected for each file in a table. For tables with a large number of columns, Delta Lake by default collects and stores statistics only for the first 32 columns. These statistics include min/max values and null counts, which are used to optimize query execution by skipping irrelevant data files. When dealing with highly nested JSON structures, understanding this behavior is crucial for schema design, especially when determining which fields should be flattened or prioritized in the table structure to leverage data skipping efficiently for performance optimization.

NEW QUESTION: 42
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 (LEAVE A REPLY)
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.

NEW QUESTION: 43
A data engineering team uses Databricks Lakehouse Monitoring to track the percent_null metric for a critical column in their Delta table.
The profile metrics table (prod_catalog.prod_schema.customer_data_profile_metrics) stores hourly percent_null values.
The team wants to:
Trigger an alert when the daily average of percent_null exceeds 5% for
three consecutive days.
Ensure that notifications are not spammed during sustained issues.
A. SELECT percent_null
FROM prod_catalog.prod_schema.customer_data_profile_metrics
WHERE window.end >= CURRENT_TIMESTAMP - INTERVAL '1' DAY

Alert Condition: percent_null > 5
Notification Frequency: At most every 24 hours
B. WITH daily_avg AS (
SELECT DATE_TRUNC('DAY', window.end) AS day,
AVG(percent_null) AS avg_null
FROM prod_catalog.prod_schema.customer_data_profile_metrics
GROUP BY DATE_TRUNC('DAY', window.end)
)
SELECT day, avg_null
FROM daily_avg
ORDER BY day DESC
LIMIT 3
Alert Condition: ALL avg_null > 5 for the latest 3 rows
Notification Frequency: Just once
C. SELECT AVG(percent_null) AS daily_avg
FROM prod_catalog.prod_schema.customer_data_profile_metrics
WHERE window.end >= CURRENT_TIMESTAMP - INTERVAL '3' DAY
Alert Condition: daily_avg > 5
Notification Frequency: Each time alert is evaluated
D. SELECT SUM(CASE WHEN percent_null > 5 THEN 1 ELSE 0 END) AS violation_days FROM prod_catalog.prod_schema.customer_data_profile_metrics WHERE window.end >= CURRENT_TIMESTAMP - INTERVAL '3' DAY
Alert Condition: violation_days >= 3 Notification Frequency: Just once
Answer: B (LEAVE A REPLY)
The key requirement is to detect when the daily average of percent_null is greater than 5% for three consecutive days.
Option A only checks the last 24 hours, not consecutive days. It would trigger too frequently and cause spam.
Option C calculates an average across all records in the last 3 days, but this could be skewed by one high or low day -- it does not ensure consecutive daily violations.
Option D simply counts days where the threshold was exceeded, but it does not guarantee that those days were consecutive. This could incorrectly trigger on non-adjacent violations.
Option B is correct:
It aggregates hourly values into daily averages.
It checks that the last 3 consecutive days all had averages above 5%.
It avoids redundant alerts by using Notification Frequency: Just once.
This matches Databricks Lakehouse Monitoring best practices, where SQL alerts should be designed to aggregate metrics to the correct granularity (daily here) and ensure consecutive threshold violations before triggering.

NEW QUESTION: 44
What statement is true regarding the retention of job run history?
A. It is retained for 30 days, during which time you can deliver job run logs to DBFS or S3
B. It is retained for 60 days, during which you can export notebook run results to HTML
C. It is retained until you export or delete job run logs

D. It is retained for 60 days, after which logs are archived
E. It is retained for 90 days or until the run-id is re-used through custom run configuration
Answer: B (LEAVE A REPLY)

NEW QUESTION: 45
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
E. End-to-end
Answer: B (LEAVE A REPLY)
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.

NEW QUESTION: 46
A table named user_ltv is being used to create a view that will be used by data analysts 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 email_ltv AS
SELECT
CASE WHEN
  is_member('marketing') THEN email
  ELSE 'REDACTED'
END AS email,
ltv
FROM user_ltv
```


An analyst who is not a member of the marketing group executes the following query:
SELECT * FROM email_ltv
Which statement describes the results returned by this query?
A. Three columns will be returned, but one column will be named "redacted" and contain only null values.
B. Only the email and itv columns will be returned; the email column will contain all null values.
C. The email and ltv columns will be returned with the values in user itv.
D. The email, age. and ltv columns will be returned with the values in user ltv.

E. Only the email and ltv columns will be returned; the email column will contain the string "REDACTED" in each row.

Answer: E (LEAVE A REPLY)

The code creates a view called email_ltv that selects the email and ltv columns from a table called user_ltv, which has the following schema: email STRING, age INT, ltv INT. The code also uses the CASE WHEN expression to replace the email values with the string "REDACTED" if the user is not a member of the marketing group. The user who executes the query is not a member of the marketing group, so they will only see the email and ltv columns, and the email column will contain the string "REDACTED" in each row.

NEW QUESTION: 47

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.

Answer: B (LEAVE A REPLY)

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 Get Latest & Actual Certified-Data-Engineer-Professional Exam's Question and Answers from constraint, and the target table will not be updated.

NEW QUESTION: 48
A platform team is creating a standardized template for Databricks Asset Bundles to support CI/CD. The template must specify defaults for artifacts, workspace root paths, and a run identity, while allowing a "dev" target to be the default and override specific paths. How should the team use databricks.yml to satisfy these requirements?
A. Use deployment, builds, context, identity, and environments; set dev as default environment and override paths under builds.
B. Use roots, modules, profiles, actor, and targets; where profiles contain workspace and artifacts defaults and actor sets run identity.
C. Use project, packages, environment, identity, and stages; set dev as default stage and override workspace under environment.
D. Use bundle, artifacts, workspace, run_as, and targets at the top level; set one target with default:true and override workspace paths or artifacts under that target.
Answer: D (LEAVE A REPLY)
In Databricks Asset Bundles, the databricks.yml file defines all top-level configuration keys, including bundle, artifacts, workspace, run_as, and targets. The targets section defines specific deployment contexts (for example, dev, test, prod). Setting default: true for a target marks it as the default environment. Overrides for workspace paths and artifact configurations can be defined inside each target while keeping defaults at the top level.

NEW QUESTION: 49
Which statement regarding spark configuration on the Databricks platform is true?
A. Spark configuration properties set for an interactive cluster with the Clusters UI will impact all notebooks attached to that cluster.
B. When the same spar configuration property is set for an interactive to the same interactive cluster.
Get Latest & Actual Certified-Data-Engineer-Professional Exam's Question and Answers from
C. Spark configuration set within an notebook will affect all SparkSession attached to the same interactive cluster
D. The Databricks REST API can be used to modify the Spark configuration properties for an interactive cluster without interrupting jobs.
E. Spark configuration properties can only be set for an interactive cluster by creating a global init script.
Answer: A (LEAVE A REPLY)
When Spark configuration properties are set for an interactive cluster using the Clusters UI in Databricks, those configurations are applied at the cluster level. This means that all notebooks attached to that cluster will inherit and be affected by these configurations. This approach ensures consistency across all executions within that cluster, as the Spark configuration properties dictate aspects such as memory allocation, number of executors, and other vital execution parameters. This centralized configuration management helps maintain standardized execution environments across different notebooks, aiding in debugging and performance optimization.

NEW QUESTION: 50
A data engineer is designing a system to process batch patient encounter data stored in an S3 bucket, creating a Delta table (patient_encounters) with columns encounter_id, patient_id, encounter_date, diagnosis_code, and treatment_cost. The table is queried frequently by patient_id and encounter_date, requiring fast performance. Fine-grained access controls must be enforced. The engineer wants to minimize maintenance and boost performance. How should the data engineer create the patient_encounters table?
A. Create an external table in Unity Catalog, specifying an S3 location for the data files. Enable predictive optimization through table properties, and configure Unity Catalog permissions for access controls.

B. Create a managed table in Unity Catalog. Configure Unity Catalog permissions for access controls, and rely on predictive optimization to enhance query performance and simplify maintenance.
C. Create a managed table in Unity Catalog. Configure Unity Catalog permissions for access controls, schedule jobs to run OPTIMIZE and VACUUM commands daily to achieve best performance.
D. Create a managed table in Hive Metastore. Configure Hive Metastore permissions for access controls, and rely on predictive optimization to enhance query performance and simplify maintenance.
Answer: B (LEAVE A REPLY)
Databricks documentation specifies that Unity Catalog managed tables are the preferred choice for secure, low-maintenance Delta Lake architectures. Managed tables provide full lifecycle management, including metadata, file storage, and access control integration with Unity Catalog.
Fine-grained permissions can be enforced at the column and row level through built-in Unity Catalog governance.
Additionally, Predictive Optimization (Auto Optimize + Auto Compaction) automatically manages file sizes, metadata pruning, and layout optimization, eliminating the need for manual maintenance such as scheduling OPTIMIZE or VACUUM.
External tables (A) require manual path management, and Hive Metastore tables (D) do not support Unity Catalog access policies.
Therefore, creating a managed Unity Catalog table with predictive optimization provides both the security and performance benefits needed, making B the correct solution.

NEW QUESTION: 51
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 (LEAVE A REPLY)
Delta Lake maintains a transaction log that records details about every change made to a table.
When you execute a count operation on a Delta table, Delta Lake can use the information in the transaction log to calculate the total number of records without having to scan all the data files.
This is because the transaction log includes information about the number of records in each file, allowing for an efficient aggregation of these counts to get the total number of records in the table.

NEW QUESTION: 52
The Databricks CLI is use to trigger a run of an existing job by passing the job_id parameter. The response that the job run request has been submitted successfully includes a filed run_id.
Which statement describes what the number alongside this field represents?
A. The job_id is returned in this field.
B. The job_id and number of times the job has been are concatenated and returned.
C. The number of times the job definition has been run in the workspace.

D. The globally unique ID of the newly triggered run.
E. The total number of jobs that have been run in the workspace.
Answer: D (LEAVE A REPLY)
When triggering a job run using the Databricks CLI, the run_id field in the response represents a globally unique identifier for that particular run of the job. This run_id is distinct from the job_id.
While the job_id identifies the job definition and is constant across all runs of that job, the run_id is unique to each execution and is used to track and query the status of that specific job run within the Databricks environment. This distinction allows users to manage and reference individual executions of a job directly.

NEW QUESTION: 53
A Delta Lake table in the Lakehouse named customer_parsams is used in churn prediction by the machine learning team. The table contains information about customers derived from a number of upstream sources. Currently, the data engineering team populates this table nightly by overwriting the table with the current valid values derived from upstream data sources.
Immediately after each update succeeds, the data engineer team would like to determine the difference between the new version and the previous of the table. Given the current implementation, which method can be used?
Get Latest & Actual Certified-Data-Engineer-Professional Exam's Question and Answers from
A. Parse the Delta Lake transaction log to identify all newly written data files.
B. Execute DESCRIBE HISTORY customer_churn_params to obtain the full operation metrics for the update, including a log of all records that have been added or modified.
C. Execute a query to calculate the difference between the new version and the previous version using Delta Lake's built-in versioning and time travel functionality.
D. Parse the Spark event logs to identify those rows that were updated, inserted, or deleted.
Answer: C (LEAVE A REPLY)
Delta Lake provides built-in versioning and time travel capabilities, allowing users to query previous snapshots of a table. This feature is particularly useful for understanding changes between different versions of the table. In this scenario, where the table is overwritten nightly, you can use Delta Lake's time travel feature to execute a query comparing the latest version of the table (the current state) with its previous version. This approach effectively identifies the differences (such as new, updated, or deleted records) between the two versions. The other options do not provide a straightforward or efficient way to directly compare different versions of a Delta Lake table.

NEW QUESTION: 54
The data engineering team is migrating an enterprise system with thousands of tables and views into the Lakehouse. They plan to implement the target architecture using a series of bronze, silver, and gold tables. Bronze tables will almost exclusively be used by production data engineering workloads, while silver tables will be used to support both data engineering and machine learning workloads. Gold tables will largely serve business intelligence and reporting purposes. While personal identifying information (PII) exists in all tiers of data, pseudonymization and anonymization rules are in place for all data at the silver and gold levels.
The organization is interested in reducing security concerns while maximizing the ability to collaborate across diverse teams.
Which statement exemplifies best practices for implementing this system?
A. Isolating tables in separate databases based on data quality tiers allows for easy permissions management through database ACLs and allows physical separation of default storage locations for managed tables.

B. Because databases on Databricks are merely a logical construct, choices around database organization do not impact security or discoverability in the Lakehouse.
C. Storinq all production tables in a single database provides a unified view of all data assets available throughout the Lakehouse, simplifying discoverability by granting all users view privileges on this database.
D. Working in the default Databricks database provides the greatest security when working with managed tables, as these will be created in the DBFS root.
E. Because all tables must live in the same storage containers used for the database they're created in, organizations should be prepared to create between dozens and thousands of databases depending on their data isolation requirements.
Answer: A (LEAVE A REPLY)
This is the correct answer because it exemplifies best practices for implementing this system. By isolating tables in separate databases based on data quality tiers, such as bronze, silver, and gold, the data engineering team can achieve several benefits. First, they can easily manage permissions for different users and groups through database ACLs, which allow granting or revoking access to databases, tables, or views. Second, they can physically separate the default storage locations for managed tables in each database, which can improve performance and reduce costs. Third, they can provide a clear and consistent naming convention for the tables in each database, which can improve discoverability and usability.

NEW QUESTION: 55
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: A (LEAVE A REPLY)
Get Latest & Actual Certified-Data-Engineer-Professional Exam's Question and Answers from
https://docs.databricks.com/en/clusters/cluster-config-best-practices.html

NEW QUESTION: 56
A data engineer manages a Unity Catalog table customer_data in schema finance that includes sensitive fields like ssn and credit_score. Intern Group should only see masked values, while Analyst Group should only access rows for their assigned region. The data engineer needs to restrict access based on user role and region without duplicating data. How should the data engineer enforce this security policy?

A. Use Unity Catalog's row filters based on the region and column masks based on user roles.
B. Create views using current_user() and is_account_group_member() functions, and apply masking logic inside the SQL SELECT clause for each sensitive column.
C. Create dynamic views for each user role and manage access with ACLs.
D. Use Unity Catalog's row filters based on the user roles and column masks based on the region.
Answer: A (LEAVE A REPLY)
Unity Catalog row filters can restrict which rows are visible based on attributes such as the user's assigned region, while column masks can dynamically obfuscate sensitive fields like ssn and credit_score based on user roles. This enforces fine-grained, role-and region-based access control directly at the table level without duplicating data or relying on custom views.

NEW QUESTION: 57
A data engineer us ingesting JSON files from cloud object storage using Databricks Auto Loader.
The source folder may occasionally receive large files of data, which risks overwhelming the stream. To ensure predictable micro-batch sizes, the team wants to throttle ingestion based on the volume of data scanned at 1 GB, regardless of the number of files. Which Auto Loader configuration should the data engineer used to achieve this?
A. Configure cloudFiles.maxBytesPerTrigger with 1 GB to place a limit.
B. Configure cloudFiles.maxSizePerTrigger with 1 GB to place a limit.
C. Configure cloudFiles.maxFilesPerTrigger and estimate the average file size to approximate a size-based throttle of 1 GB.
D. Configure cloudFiles.maxPartitionBytes with 1GB to limit data in each partition.
Answer: (SHOW ANSWER)
cloudFiles.maxBytesPerTrigger limits the total volume of data scanned in each micro-batch based on size rather than file count. Setting it to 1 GB ensures predictable ingestion throughput even when large files arrive, preventing any single trigger from overwhelming the streaming job.

NEW QUESTION: 58
Two of the most common data locations on Databricks are the DBFS root storage and external object storage mounted with dbutils.fs.mount().
Which of the following statements is correct?
A. DBFS is a file system protocol that allows users to interact with files stored in object storage using syntax and guarantees similar to Unix file systems.
B. By default, both the DBFS root and mounted data sources are only accessible to workspace administrators.
C. The DBFS root is the most secure location to store data, because mounted storage volumes must have full public read and write permissions.
D. Neither the DBFS root nor mounted storage can be accessed when using %sh in a Databricks notebook.
E. The DBFS root stores files in ephemeral block volumes attached to the driver, while mounted directories will always persist saved data to external storage between sessions.
Answer: A (LEAVE A REPLY)
DBFS is a file system protocol that allows users to interact with files stored in object storage using syntax and guarantees similar to Unix file systems. DBFS is not a physical file system, but a layer over the object storage that provides a unified view of data across different data sources. By Get Latest & Actual Certified-Data-Engineer-Professional Exam's Question and Answers from default, the DBFS root is accessible to all users in the workspace, and the access to mounted data sources depends on the permissions of the

storage account or container. Mounted storage volumes do not need to have full public read and write permissions, but they do require a valid connection string or access key to be provided when mounting. Both the DBFS root and mounted storage can be accessed when using %sh in a Databricks notebook, as long as the cluster has FUSE enabled. The DBFS root does not store files in ephemeral block volumes attached to the driver, but in the object storage associated with the workspace. Mounted directories will persist saved data to external storage between sessions, unless they are unmounted or deleted.

NEW QUESTION: 59
A view is registered with the following code:

Both users and orders are Delta Lake tables.
Which statement describes the results of querying recent_orders?
A. Results will be computed and cached when the view is defined; these cached results will incrementally update as new records are inserted into source tables.
B. All logic will execute at query time and return the result of joining the valid versions of the source tables at the time the query began.
C. All logic will execute when the view is defined and store the result of joining tables to the DBFS; this stored data will be returned when the view is queried.
D. All logic will execute at query time and return the result of joining the valid versions of the source tables at the time the query finishes.
Answer: (SHOW ANSWER)

NEW QUESTION: 60
A job runs four independent tasks (X, Y, Z, W) in parallel to process regional sales data. The Data Engineering team recently updated its cluster policy to ban cost-prohibitive instance types. Task Y now fails due to the newly enforced cluster policy restricting the use of a specific instance type.
A data engineer needs to resolve the failure quickly without disrupting the other tasks. How should the data engineer resolve the failure of tasks?
A. Delete the failed run, disable the cluster policy, and re-execute all tasks.
B. Manually create a new cluster for Task Y, update the job configuration, and trigger a full re-run.
C. Use "Repair run", override the cluster configuration for Task Y to use a permitted instance type, and let Databricks re-run only Task Y.
D. Edit the global cluster policy to allow the restricted instance type, then re-run the entire job.
Answer: (SHOW ANSWER)
Repair run allows re-running only the failed task without affecting successfully completed tasks.

Overriding the cluster configuration for the specific task resolves the policy violation quickly while keeping the rest of the job intact and minimizing disruption.

NEW QUESTION: 61
A data engineering team is implementing an append-only data pipeline using Delta Lake, and wants to ensure that data is never modified or deleted once written. Which Delta Lake feature should the data engineer enable to prevent modifications to existing data?
A. Delta APPEND_ONLY
B. Delta VACUUM
C. Delta OPTIMIZE
D. Delta Time Travel
Answer: A (LEAVE A REPLY)
Enabling the append-only table property enforces that data can only be inserted into the Delta table. Updates and deletes are blocked, ensuring that once data is written it is never modified or removed, which is essential for strict append-only pipeline guarantees.

NEW QUESTION: 62
Review the following error traceback:

```
AnalysisException                         Traceback (most recent call last)
<command-3293767849433948> in <module>
----> 1 display(df.select(3*"heartrate"))

/databricks/spark/python/pyspark/sql/dataframe.py in select(self, *cols)
   1690         [Row(name='Alice', age=12), Row(name='Bob', age=15)]
   1691         """
-> 1692         jdf = self._jdf.select(self._jcols(*cols))
   1693         return DataFrame(jdf, self.sql_ctx)
   1694

/databricks/spark/python/lib/py4j-0.10.9-src.zip/py4j/java_gateway.py in __call__(self, *args
   1302
   1303         answer = self.gateway_client.send_command(command)
-> 1304         return_value = get_return_value(
   1305             answer, self.gateway_client, self.target_id, self.name)
   1306

/databricks/spark/python/pyspark/sql/utils.py in deco(*a, **kw)
    121                 # Hide where the exception came from that shows a non-Pythonic
    122                 # JVM exception message.
--> 123                 raise converted from None
    124             else:
    125                 raise

AnalysisException: cannot resolve '`heartrateheartrateheartrate`' given input columns:
[spark_catalog.database.table.device_id, spark_catalog.database.table.heartrate,
spark_catalog.database.table.mrn, spark_catalog.database.table.time];
'Project ['heartrateheartrateheartrate]
+- SubqueryAlias spark_catalog.database.table
```

Which statement describes the error being raised?

A. The code executed was PvSoark but was executed in a Scala notebook.

B. There is no column in the table named heartrateheartrateheartrate

C. There is a type error because a column object cannot be multiplied.

D. There is a type error because a DataFrame object cannot be multiplied.

E. There is a syntax error because the heartrate column is not correctly identified as a column.

Answer: B (LEAVE A REPLY)

https://sparkbyexamples.com/spark/spark-cannot-resolve-given-input-columns/

NEW QUESTION: 63

The data governance team has instituted a requirement that the "user" table containing Personal Identifiable Information (PII) must have the appropriate masking on the SSN column. This means that anyone outside of the HRAdminGroup should see masked social security numbers as ***-**-
****.

The team created a masking function:

```
CREATE FUNCTION ssn_mask(ssn STRING)
  RETURN CASE WHEN is_member(HRAdminGroup) THEN ssn ELSE '***-**-****' END;
```

What does the data governance team need to do next to achieve this goal?

A. CREATE TABLE users
(name STRING);
ALTER TABLE users CREATE COLUMN ssn CREATE MASK ssn_mask;

B. CREATE TABLE users
(name STRING, int STRING);
ALTER TABLE users ALTER COLUMN ssn CREATE MASK if is_member('HRAdminGroup');

C. CREATE TABLE users
(name STRING, ssn INT MASKED ssn_mask);

D. CREATE TABLE users
(name STRING, ssn STRING);
ALTER TABLE users ALTER COLUMN ssn SET MASK ssn_mask;

Answer: (SHOW ANSWER)

In Databricks, after creating a masking function, you apply it to a column using ALTER TABLE <table> ALTER COLUMN <column> SET MASK <mask_function>. The table must already include the column (here, ssn as STRING). This ensures that only users in the HRAdminGroup see the unmasked SSN, while all others see the masked value.

NEW QUESTION: 64

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

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

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


The next step in the pipeline requires a function that returns an object that can be used to manipulate new records that have not yet been processed to the next table in the pipeline.

Which code snippet completes this function definition?

A. def new_records():

B. return spark.readStream.table("bronze")
C. return spark.readStream.load("bronze")
D. return spark.read.option("readChangeFeed", "true").table ("bronze")

```
return (spark.read
    .table("bronze")
    .filter(col("source_file") == f"/mnt/daily_batch/{year}/{month}/{day}")
)
```

E.
Answer: (SHOW ANSWER)
https://docs.databricks.com/en/delta/delta-change-data-feed.html
Get Latest & Actual Certified-Data-Engineer-Professional Exam's Question and Answers from

NEW QUESTION: 65
An hourly batch job is configured to ingest data files from a cloud object storage container where each batch represent all records produced by the source system in a given hour. The batch job to process these records into the Lakehouse is sufficiently delayed to ensure no late-arriving data is missed. The user_id field represents a unique key for the data, which has the following schema: user_id BIGINT, username STRING, user_utc STRING, user_region STRING, last_login BIGINT, auto_pay BOOLEAN, last_updated BIGINT New records are all ingested into a table named account_history which maintains a full record of all data in the same schema as the source. The next table in the system is named account_current and is implemented as a Type 1 table representing the most recent value for each unique user_id.
Assuming there are millions of user accounts and tens of thousands of records processed hourly, which implementation can be used to efficiently update the described account_current table as part of each hourly batch job?
A. Use Auto Loader to subscribe to new files in the account history directory; configure a Structured Streaminq trigger once job to batch update newly detected files into the account current table.
B. Overwrite the account current table with each batch using the results of a query against the account history table grouping by user id and filtering for the max value of last updated.
C. Filter records in account history using the last updated field and the most recent hour processed, as well as the max last iogin by user id write a merge statement to update or insert the most recent value for each user id.
D. Use Delta Lake version history to get the difference between the latest version of account history and one version prior, then write these records to account current.
E. Filter records in account history using the last updated field and the most recent hour processed, making sure to deduplicate on username; write a merge statement to update or insert the most recent value for each username.
Answer: C (LEAVE A REPLY)
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.

NEW QUESTION: 66
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 Run

C. No permissions

D. Can Read

E. Can Edit

Answer: D (LEAVE A REPLY)

NEW QUESTION: 67

A junior data engineer has configured a workload that posts the following JSON to the Databricks REST API endpoint 2.0/jobs/create.

Assuming that all configurations and referenced resources are available, which statement describes the result of executing this workload three times?

A. Three new jobs named "Ingest new data" will be defined in the workspace, and they will each run once daily.

B. The logic defined in the referenced notebook will be executed three times on new clusters with the configurations of the provided cluster ID.

C. Three new jobs named "Ingest new data" will be defined in the workspace, but no jobs will be executed.

D. One new job named "Ingest new data" will be defined in the workspace, but it will not be executed.

E. The logic defined in the referenced notebook will be executed three times on the referenced existing all purpose cluster.

Answer: C (LEAVE A REPLY)

Databricks jobs create will create a new job with the same name each time it is run.

In order to overwrite the extsting job you need to run databricks jobs reset

NEW QUESTION: 68

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 (LEAVE A REPLY)

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.

NEW QUESTION: 69
A data engineer is building a streaming data pipeline to ingest JSON files from cloud storage into a Delta Lake table. The pipeline must process files incrementally, handle schema evolution automatically, ensure exactly-once processing, and minimize manual infrastructure management.
How should the data engineer fulfill these requirements?
A. Use Lakeflow Spark Declarative Pipelines with a static DataFrame read, merge schema with spark.conf.set ("spark.databricks.delta.schema.autoMerge.enabled", "true")
B. Use Auto Loader in batch mode with a daily job to overwrite the Delta table.
C. Use Lakeflow Spart Declarative Pipelines with Auto Loader and enabling schema inference with "cloudFiles.schemaEvolutionMode"= "addNewColumns"
D. Use traditional Spark Structured Streaming with Auto Loader, manually configuring checkpoints location and enabling schema inference with "mergeSchema"= "true"
Answer: (SHOW ANSWER)
Lakeflow Spark Declarative Pipelines combined with Auto Loader provide fully managed incremental file ingestion with exactly-once guarantees and minimal operational overhead.
Enabling schema inference and evolution allows new columns in incoming JSON files to be incorporated automatically, satisfying the requirements for streaming ingestion, schema evolution, and reduced manual infrastructure management.

NEW QUESTION: 70
A security analytics pipeline must enrich billions of raw connection logs with geolocation data.
The join hinges on finding which IPv4 range each event's address falls into.
Table 1: network_events (5 billion rows)
event_id ip_int
42 3232235777
Table 2: ip_ranges (2 million rows)
start_ip_int end_ip_int country
3232235520 3232236031 US
The query is currently very slow:
SELECT n.event_id, n.ip_int, r.country
FROM network_events n
JOIN ip_ranges r
ON n.ip_int BETWEEN r.start_ip_int AND r.end_ip_int;
Which change will most dramatically accelerate the query while preserving its logic?
A. Increase spark.sql.shuffle.partitions from 200 to 10000.
B. Add a range-join hint /*+ RANGE_JOIN(r, 65536) */.
C. Force a sort-merge join with /*+ MERGE(r) */.

D. Add a broadcast hint: /*+ BROADCAST(r) */ for ip_ranges.

Answer: (SHOW ANSWER)

The query joins billions of rows (network_events) with millions of rows (ip_ranges) using a range predicate (BETWEEN). Unlike equality joins (=), range joins are not efficiently handled by broadcast or sort-merge joins because:

Broadcast Join (D): Effective for small tables but only for equality joins. Since this query uses a range condition, broadcast will not reduce the complexity of scanning billions of records across non-equality conditions.

Sort-Merge Join (C): Works for ordered joins but is inefficient on range conditions. Sorting billions of records adds excessive overhead and will not resolve the bottleneck.

Increasing Shuffle Partitions (A): Only spreads out shuffle work but does not address the fundamental inefficiency of range-based lookups at scale.

Range Joins in Spark (RANGE_JOIN hint):

Databricks provides range join optimizations specifically for conditions such as BETWEEN. By applying a RANGE_JOIN hint, Spark can build optimized data structures (such as interval indexes or partition pruning strategies) that map billions of input rows to ranges much faster. This avoids brute- force scans and unnecessary shuffle costs.

Thus, Option B is the correct solution because:

It leverages range-join optimization, which is purpose-built for queries joining massive event logs to smaller lookup tables with IP ranges.

This ensures Spark can evaluate billions of rows against millions of ranges with optimized matching logic, drastically improving query performance while preserving correctness.

NEW QUESTION: 71

What is the first line of a Databricks Python notebook when viewed in a text editor?

A. %python

B. # Databricks notebook source

C. -- Databricks notebook source

D. // Databricks notebook source

E. # MAGIC %python

Answer: B (LEAVE A REPLY)

https://docs.databricks.com/en/notebooks/notebook-export-import.html#import-a-file-and-convert- it-to-a-notebook

NEW QUESTION: 72

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 (LEAVE A REPLY)
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 Get Latest & Actual Certified-Data-Engineer-Professional Exam's Question and Answers from 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.

NEW QUESTION: 73
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 Get Latest & Actual Certified-Data-Engineer-Professional Exam's Question and Answers from events and the job run events.
Answer: C (LEAVE A REPLY)
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.

NEW QUESTION: 74
Which statement describes Delta Lake optimized writes?
A. A shuffle occurs prior to writing to try to group data together resulting in fewer files instead of each executor writing multiple files based on directory partitions.
B. Optimized writes logical partitions instead of directory partitions partition boundaries are only Get Latest & Actual Certified-Data-Engineer-Professional Exam's Question and Answers from represented in metadata fewer small files are written.
C. An asynchronous job runs after the write completes to detect if files could be further compacted; yes, an OPTIMIZE job is executed toward a default of 1 GB.
D. Before a job cluster terminates, OPTIMIZE is executed on all tables modified during the most recent job.
Answer: (SHOW ANSWER)
Delta Lake optimized writes involve a shuffle operation before writing out data to the Delta table.
The shuffle operation groups data by partition keys, which can lead to a reduction in the number of output files and potentially larger files, instead of multiple smaller files. This approach can significantly reduce the total number of files in the table, improve read performance by reducing the metadata overhead, and optimize the table storage layout, especially for workloads with many small files.

NEW QUESTION: 75
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.
Get Latest & Actual Certified-Data-Engineer-Professional Exam's Question and Answers from

```
MERGE INTO customers
USING (
  SELECT updates.customer_id as merge_ey, updates.*
  FROM updates

  UNION ALL

  SELECT NULL as merge_key, updates.*
  FROM updates JOIN customers
  ON updates.customer_id = customers.customer_id
  WHERE customers.current = true AND updates.address <> customers.address
) staged_updates
ON customers.customer_id = mergeKey
WHEN MATCHED AND customers.current = true AND customers.address <> staged_updates.address THEN
  UPDATE SET current = false, end_date = staged_updates.effective_date
WHEN NOT MATCHED THEN
  INSERT(customer_id, address, current, effective_date, end_date)
  VALUES(staged_updates.customer_id, staged_updates.address, true, staged_updates.effective_date,
null)
```

Which statement describes this implementation?
A. The customers table is implemented as a Type 2 table; old values are overwritten and new customers are appended.
B. The customers table is implemented as a Type 3 table; old values are maintained as a new column alongside the current value.
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 maintained but marked as no longer current and new values are inserted.
Answer: E (LEAVE A REPLY)

NEW QUESTION: 76
A data engineer, while designing a Pandas UDF to process financial time-series data with complex calculations that require maintaining state across rows within each stock symbol group, must ensure the function is efficient and scalable. Which approach will solve the problem with minimum overhead while preserving data integrity?
A. Use a SCALAR_ITER Pandas UDF with iterator-based processing, implementing state management through persistent storage (Delta tables) that gets updated after each batch to maintain continuity across iterator chunks.
B. Use a SCALAR Pandas UDF that processes the entire dataset at once, implementing custom partitioning logic within the UDF to group by stock symbol and maintain state using global variables shared across all executor processes.
C. Use applyInPandas() on a Spark DataFrame that receives all rows for each stock symbol as a Pandas DataFrame, allowing processing within each group while maintaining state variables local to each group's processing function.
D. Use a grouped_agg Pandas UDF that processes each stock symbol group independently, maintaining state through intermediate aggregation results that get passed between successive UDF calls via broadcast variables.
Answer: C (LEAVE A REPLY)
The Databricks documentation recommends applyInPandas() for complex per-group operations where maintaining internal state within each group is necessary. When using applyInPandas(), Spark provides all records for each grouping key as a Pandas DataFrame to the function, allowing efficient vectorized operations with local state management. This approach ensures high performance and scalability while maintaining logical isolation between groups. In contrast, SCALAR and SCALAR_ITER UDFs operate on individual rows or batches and cannot maintain inter-row state effectively. grouped_agg UDFs are limited to computing aggregates and do not support complex multi-row transformations. Therefore, applyInPandas() is the correct and Databricks-recommended solution for stateful per-group time-series computations.

NEW QUESTION: 77
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 Get Latest & Actual Certified-Data-Engineer-Professional Exam's Question and Answers from 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 (LEAVE A REPLY)
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.

NEW QUESTION: 78
A data engineer wants to ingest a large collection of image files (JPEG and PNG) from cloud object storage into a Unity Catalog-managed table for analysis and visualization. Which two configurations and practices are recommended to incrementally ingest these images into the table? (Choose two.)
A. Move files to a volume and read with SQL editor.
B. Use Auto Loader and set cloudFiles.format to "BINARYFILE".
C. Use Auto Loader and set cloudFiles.format to "TEXT".
D. Use Auto Loader and set cloudFiles.format to "IMAGE".
E. Use the pathGlobFilter option to select only image files (e.g., "*.jpg,*.png").
Answer: (SHOW ANSWER)
Databricks Auto Loader supports ingestion of binary file formats using the cloudFiles.format option. For ingesting JPEG or PNG image files, the correct setting is "BINARYFILE", which loads the raw binary content and file metadata into a DataFrame. Additionally, when processing files from object storage, it is best practice to apply pathGlobFilter to limit ingestion to specific file types and reduce unnecessary scanning of non-image files. Options like "IMAGE" or "TEXT" are invalid, and using volumes with SQL editors does not provide incremental ingestion. Therefore, combining Auto Loader with cloudFiles.format="BINARYFILE" and pathGlobFilter ensures scalable, incremental ingestion of image data into Unity Catalog tables.

NEW QUESTION: 79
A data engineering team is setting up deployment automation. To deploy workspace assets remotely using the Databricks CLI command, they must configure it with proper authentication.
Which authentication approach will provide the highest level of security?
A. Use a service principal with OAuth token federation.
B. Use a service principal ID and its OAuth client secret.
C. Use a service principal and its Personal Access Token.
D. Use a shared user account and its OAuth client secret.
Answer: (SHOW ANSWER)
The most secure and enterprise-recommended authentication method for Databricks automation is OAuth token federation with service principals.

This configuration allows service principals (non-human identities) to authenticate using temporary OAuth access tokens from a trusted identity provider (such as Azure AD or AWS IAM federation). These tokens are short-lived and scoped, significantly reducing credential exposure risks.
By contrast, static client secrets (B) or PATs (C) are long-lived and require periodic manual rotation, increasing security vulnerability. Shared user accounts (D) violate least-privilege and auditability principles. Therefore, A provides the strongest, most compliant authentication model for automated CLI and CI/CD workflows.

NEW QUESTION: 80
A table named user_ltv is being used to create a view that will be used by data analysts 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 analyst who is not a member of the auditing group executes the following query:
SELECT * FROM user_ltv_no_minors
Which statement describes the results 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.
E. All values for the age column will be returned as null values, all other columns will be returned with the values in user_ltv.
Answer: A (LEAVE A REPLY)
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.

NEW QUESTION: 81
A table in the Lakehouse named customer_churn_params is used in churn prediction by the machine learning team. The table contains information about customers derived from a number of upstream sources. Currently, the data engineering team populates this table nightly by overwriting the table with the current valid values derived from upstream data sources.
The churn prediction model used by the ML team is fairly stable in production. The team is only interested in making predictions on records that have changed in the past 24 hours.
Which approach would simplify the identification of these changed records?
A. Apply the churn model to all rows in the customer_churn_params table, but implement logic to perform an upsert into the predictions table that ignores rows where predictions have not changed.
B. Convert the batch job to a Structured Streaming job using the complete output mode; configure a Structured Streaming job to read from the customer_churn_params table and incrementally predict against the churn model.
C. Calculate the difference between the previous model predictions and the current customer_churn_params on a key identifying unique customers before making new predictions; only make predictions on those customers not in the previous predictions.
D. Modify the overwrite logic to include a field populated by calling spark.sql.functions.current_timestamp() as data are being written; use this field to identify records written on a particular date.
E. Replace the current overwrite logic with a merge statement to modify only those records that have changed; write logic to make predictions on the changed records identified by the change data feed.
Answer: (SHOW ANSWER)
The approach that would simplify the identification of the changed records is to replace the current overwrite logic with a merge statement to modify only those records that have changed, and write logic to make predictions on the changed records identified by the change data feed.
This approach leverages the Delta Lake features of merge and change data feed, which are designed to handle upserts and track row-level changes in a Delta table. By using merge, the data engineering team can avoid overwriting the entire table every night, and only update or insert the records that have changed in the source data. By using change data feed, the ML team can easily access the change events that have occurred in the customer_churn_params table, and filter them by operation type (update or insert) and timestamp. This way, they can only make predictions on the records that have changed in the past 24 hours, and avoid re-processing the unchanged records.

NEW QUESTION: 82
A developer has successfully configured their credentials for Databricks Repos and cloned a remote Git repository. They do not have privileges to make changes to the main branch, which is the only branch currently visible in their workspace. Which approach allows this user to share their code updates without the risk of overwriting the work of their teammates?
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
E. Use Repos to pull changes from the remote Git repository; commit and push changes to a branch that appeared as changes were pulled.
Answer: C (LEAVE A REPLY)

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.

NEW QUESTION: 83
A data engineer needs to create an application that will collect information about the latest job run including the repair history. How should the data engineer format the request?
A. Call/api/2.1/jobs/runs/list with the run_id and include_history parameters
B. Call/api/2.1/jobs/runs/get with the run_id and include_history parameters
C. Call/api/2.1/jobs/runs/get with the job_id and include_history parameters
D. Call/api/2.1/jobs/runs/list with the job_id and include_history parameters
Answer: D (LEAVE A REPLY)
To retrieve information about the latest job runs along with their repair history, you use the jobs/runs/list endpoint with the job_id and include_history=true. This endpoint returns a list of runs for a specific job, including details about retries and repair attempts, which is not available via runs/get that retrieves a single run by run_id.

NEW QUESTION: 84
A data engineer wants to refactor the following DLT code, which includes multiple table definitions with very similar code.

```
@dlt.table(name=f"t1_dataset")
def t1_dataset():
    return spark.read.table(t1)

@dlt.table(name=f"t2_dataset")
def t2_dataset():
    return spark.read.table(t2)

@dlt.table(name=f"t3_dataset")
def t3_dataset():
    return spark.read.table(t3)

...
```


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

```
tables = ["t1", "t2", "t3"]

for t in tables:
    @dlt.table(name=f"{t}_dataset")
        def new_table():
            return spark.read.table(t)
```

The pipeline runs an update with this refactored code, but generates a different DAG showing incorrect configuration values for these tables.

How can the data engineer fix this?

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

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

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

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

Answer: A (LEAVE A REPLY)

The issue with the refactored code is that it tries to use string interpolation to dynamically create table names within the dlc.table decorator, which will not correctly interpret the table names.

Instead, by using a dictionary with table names as keys and their configurations as values, the data engineer can iterate over the dictionary items and use the keys (table names) to properly configure the table settings. This way, the decorator can correctly recognize each table name, and the corresponding configuration settings can be applied appropriately.

NEW QUESTION: 85

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/daily_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 tail.

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: (SHOW ANSWER)

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.

NEW QUESTION: 86

The security team is exploring whether or not the Databricks secrets module can be leveraged for connecting to an external database. After testing the code with all Python variables being defined with strings, they upload the password to the secrets module and configure the correct permissions for the currently active user. They then modify their code to the following (leaving all other variables unchanged).

```
password = dbutils.secrets.get(scope="db_creds", key="jdbc_password")

print(password)

df = (spark
  .read
  .format("jdbc")
  .option("url", connection)
  .option("dbtable", tablename)
  .option("user", username)
  .option("password", password)
  )
```


Which statement describes what will happen when the above code is executed?

A. The connection to the external table will fail; the string "redacted" will be printed.

B. An interactive input box will appear in the notebook; if the right password is provided, the connection will succeed and the encoded password will be saved to DBFS.

C. An interactive input box will appear in the notebook; if the right password is provided, the connection will succeed and the password will be printed in plain text.

D. The connection to the external table will succeed; the string value of password will be printed in plain text.

E. The connection to the external table will succeed; the string "redacted" will be printed.

Answer: E (LEAVE A REPLY)

This is the correct answer because the code is using the dbutils.secrets.get method to retrieve the password from the secrets module and store it in a variable. The secrets module allows users to securely store and access sensitive information such as passwords, tokens, or API keys. The connection to the external table will succeed because the password variable will contain the actual password value. However, when printing the password variable, the string "redacted" will be displayed instead of the plain text password, as a security measure to prevent exposing sensitive information in notebooks.

NEW QUESTION: 87

A data engineer is analyzing a large, partitioned retail dataset in Databricks, where each row represents a sale made by a salesperson. The dataset contains millions of records with the following schema:
sales_df: [salesperson_id: string, region: string, sale_amount: double, sale_date: date] The data engineer needs to generate a DataFrame that ranks salespeople within each region based on their total cumulative sales, with the highest seller ranked as 1. If multiple salespeople have the same total sales, they should share the same rank.
The data engineer wants to implement this logic using a PySpark window function and the dense_rank () function.
Which code snippet will perform this ranking?

A.
```
from pyspark.sql import functions as F
from pyspark.sql.window import Window

window_spec = Window.partitionBy("region").orderBy
(F.desc("sale_amount"))

ranked_df= sales_df.withColumn("rank",
F.dense_rank().over(window_spec))
```

B.
```
from pyspark.sql import functions as F
from pyspark.sql.window import Window

agg_df= sales_df.groupBY("salesperson_id", "region") \
   .agg(F.sum("sale_amount").alias("total_sales"))

window_spec = Window.partitionBy("region").orderBy
(F.desc("total_sales"))

ranked_df= agg_df.withColumn("rank", F.rank().over(window_spec))
```

C.
```
from pyspark.sql import functions as F
from pyspark.sql.window import Window

agg_df= sales_df.groupBY("salesperson_id", "region") \
   .agg(F.sum("sale_amount").alias("total_sales"))

window_spec = Window.orderBy (F.desc("total_sales"))

ranked_df= agg_df.withColumn("rank",
F.dense_rank().over(window_spec))
```

D.

```
from pyspark.sql import functions as F
from pyspark.sql.window import Window

agg_df= sales_df.groupBY("salesperson_id", "region") \
   .agg(F.sum("sale_amount").alias("total_sales"))

window_spec =
Window.parititionBy("region").orderBy(F.desc("total_sales"))

ranked_df=agg_df.withcolumn("rank",
F.dense_rank().over(window_spec))
```


Answer: (SHOW ANSWER)

This approach first aggregates sales by salesperson and region to compute total cumulative sales. It then applies a window function partitioned by region and ordered by total sales in descending order, using dense_rank to assign ranks so that salespeople with equal totals share the same rank and the highest total receives rank 1.

NEW QUESTION: 88

A data engineer and a platform engineer are working together to automate their system tasks. A script needs to be executed outside of Databricks only if a particular daily Databricks job finishes successfully for the day. Databricks CLI command was used to check the last execution of the job. What are the required command options for that task?

A. databricks jobs list-runs --job-id JOB_ID --start-time-to TODAY_MIDNIGHT_EPOCH_MS -- completed-only

B. databricks jobs list-runs --job-id JOB_ID --start-time-from TODAY_MIDNIGHT_EPOCH_MS -- active-only

C. databricks jobs list-runs --job-id JOB_ID --start-time-to TODAY_MIDNIGHT_EPOCH_MS --active- only

D. databricks jobs list-runs --job-id JOB_ID --start-time-from TODAY_MIDNIGHT_EPOCH_MS -- completed-only

Answer: D (LEAVE A REPLY)

Using --start-time-from with today's midnight epoch ensures only runs that started during the current day are considered, and --completed-only filters out in-progress runs. This allows the script to reliably check whether the daily job has finished successfully before triggering downstream external actions.

NEW QUESTION: 89

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

Answer: (SHOW ANSWER)
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.

NEW QUESTION: 90
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 impotently 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.
E. Cloud vendor-specific queue storage and notification services are configured to track newly arriving files; the target table is materialized by directly querying all valid files in the source directory.
Answer: A (LEAVE A REPLY)
Get Latest & Actual Certified-Data-Engineer-Professional Exam's Question and Answers from 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.

NEW QUESTION: 91
A data engineer wants to create a cluster using the Databricks CLI for a big ETL pipeline. The cluster should have five workers, one driver of type i3.xlarge, and should use the '14.3.x- scala2.12' runtime. Which command should the data engineer use?
A. databricks clusters create 14.3.x-scala2.12 --num-workers 5 --node-type-id i3.xlarge --cluster- name DataEngineer_cluster
B. databricks clusters add 14.3.x-scala2.12 --num-workers 5 --node-type-id i3.xlarge --cluster-name Data Engineer_cluster
C. databricks compute add 14.3.x-scala2.12 --num-workers 5 --node-type-id i3.xlarge --cluster-name Data Engineer_cluster
D. databricks compute create 14.3.x-scala2.12 --num-workers 5 --node-type-id i3.xlarge --cluster- name Data Engineer_cluster
Answer: A (LEAVE A REPLY)
The correct Databricks CLI command to create a new cluster is databricks clusters create. You specify the runtime with --spark-version (here '14.3.x-scala2.12'), the number of workers with -- num-workers, the node type with --node-type-id, and the cluster name with --cluster-name. This command properly initializes the cluster with the desired configuration.

NEW QUESTION: 92
What is the first of a Databricks Python notebook when viewed in a text editor?
A. %python
B. # Databricks notebook source
C. -- Databricks notebook source
D. // Databricks notebook source
E. # MAGIC %python
Answer: (SHOW ANSWER)
https://docs.databricks.com/en/notebooks/notebook-export-import.html#import-a-file-and-convert- it-to-a-notebook

NEW QUESTION: 93
A data engineer is creating a data ingestion pipeline to understand where customers are taking their rented bicycles during use. The engineer noticed that, over time, data being transmitted from the bicycle sensors fail to include key details like latitude and longitude. Downstream analysts need both the clean records and the quarantined records available for separate processing.
The data engineer already has this code:
import dlt
from pyspark.sql.functions import expr
rules = {
"valid_lat": "(lat IS NOT NULL)",
"valid_long": "(long IS NOT NULL)"
}
quarantine_rules = "NOT({})".format(" AND ".join(rules.values()))
@dlt.view
def raw_trips_data():
return spark.readStream.table("ride_and_go.telemetry.trips")
How should the data engineer meet the requirements to capture good and bad data?
A. @dlt.table(name="trips_data_quarantine")
def trips_data_quarantine():
return (
spark.readStream.table("raw_trips_data")
.filter(expr(quarantine_rules))
)
B. @dlt.view

@dlt.expect_or_drop("lat_long_present", "(lat IS NOT NULL AND long IS NOT NULL)") def trips_data_quarantine():
return spark.readStream.table("ride_and_go.telemetry.trips")
C. @dlt.table
@dlt.expect_all_or_drop(rules)
def trips_data_quarantine():
return spark.readStream.table("raw_trips_data")
D. @dlt.table(partition_cols=["is_quarantined",])
@dlt.expect_all(rules)
def trips_data_quarantine():
return (
spark.readStream.table("raw_trips_data")
.withColumn("is_quarantined", expr(quarantine_rules))
)
Answer: (SHOW ANSWER)
The requirement is that both valid (good) and invalid (bad) records must be captured and available separately for downstream processing. Invalid records should not simply be dropped; they must be quarantined in a dedicated table.
In Databricks Lakeflow Declarative Pipelines (DLT), this is achieved by creating separate output tables:
One table for valid records (Silver table) that pass the expectations.
Another quarantine table that explicitly captures records failing the expectations.
Option A correctly implements this by:
Declaring a DLT table trips_data_quarantine.
Using .filter(expr(quarantine_rules)) to isolate invalid records (records where latitude or longitude is NULL).
This ensures analysts can query both good records (from the main Silver pipeline table) and bad records (from the quarantine table).

NEW QUESTION: 94
A Data Engineer is building a simple data pipeline using Lakeflow Declarative Pipelines (LDP) in Databricks to ingest customer data. The raw customer data is stored in a cloud storage location in JSON format. The task is to create Lakeflow Declarative Pipelines that read the raw JSON data and write it into a Delta table for further processing. Which code snippet will correctly ingest the raw JSON data and create a Delta table using LDP?
A. import dlt
@dlt.table
def raw_customers():
return spark.read.format("csv").load("s3://my-bucket/raw-customers/")
B. import dlt
@dlt.table
def raw_customers():
return spark.read.json("s3://my-bucket/raw-customers/")
C. import dlt
@dlt.table
def raw_customers():
return spark.read.format("parquet").load("s3://my-bucket/raw-customers/")

D. import dlt
@dlt.view
def raw_customers():
return spark.format.json("s3://my-bucket/raw-customers/")
Answer: B (LEAVE A REPLY)
The correct method to define a table using Lakeflow Declarative Pipelines (LDP) is with the @dlt.table decorator, which persists the output as a managed Delta table. When ingesting raw JSON data, spark.read.json() or spark.read.format("json").load() is the standard approach. This reads JSON- formatted files from the source and stores them in Delta format automatically managed by Databricks.

NEW QUESTION: 95
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 BACUM command is used to remove invalidated data files.
B. Yes; Delta Lake ACID guarantees provide assurance that the DELETE command successed 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
E. Yes; the change data feed uses foreign keys to ensure delete consistency throughout the Lakehouse.
Answer: A (LEAVE A REPLY)
Get Latest & Actual Certified-Data-Engineer-Professional Exam's Question and Answers from 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: 96
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. Ingestine all raw data and metadata from Kafka to a bronze Delta table creates a permanent, replayable history of the data state.Get Latest & Actual Certified-Data-Engineer-Professional Exam's Question and Answers from
Answer: E (LEAVE A REPLY)
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.

NEW QUESTION: 97
A data engineer needs to productionize a new Spark application written by teammate. This application has numerous external dependencies, including libraries, and requires custom environment variables and Spark configuration parameters to be set. Which two methods will help the data engineer accomplish the task? (Choose two.)
A. Install libraries on DBFS
B. Add libraries to compute policies
C. Use secrets in init scripts to store configuration data
D. Use compute policies to set system properties, environment variables, and Spark configuration parameters.
E. Create init scripts on DBFS.
Answer: D,E (LEAVE A REPLY)
Compute policies allow centrally defining and enforcing Spark configuration parameters, system properties, and environment variables required by the application, ensuring consistent production settings. Init scripts enable installing external dependencies and performing custom environment setup at cluster startup, making them essential for productionizing Spark applications with complex dependency and configuration requirements.

NEW QUESTION: 98

The data governance team is reviewing code used for deleting records for compliance with GDPR. They note the following logic is used to delete records from the Delta Lake table named users.

```
DELETE FROM users
WHERE user_id IN
   (SELECT user_id FROM delete_requests)
```

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

A. Yes; Delta Lake ACID guarantees provide assurance that the delete command succeeded fully and permanently purged these records.

B. No; the Delta cache may return records from previous versions of the table until the cluster is restarted.

C. Yes; the Delta cache immediately updates to reflect the latest data files recorded to disk.

D. No; the Delta Lake delete command only provides ACID guarantees when combined with the merge into command.

E. No; files containing deleted records may still be accessible with time travel until a vacuum command is used to remove invalidated data files.

Answer: (SHOW ANSWER)

The code uses the DELETE FROM command to delete records from the users table that match a condition based on a join with another table called delete_requests, which contains all users that have requested deletion. The DELETE FROM command deletes records from a Delta Lake table by creating a new version of the table that does not contain the deleted records. However, this does not guarantee that the records to be deleted are no longer accessible, because Delta Lake supports time travel, which allows querying previous versions of the table using a timestamp or version number. Therefore, files containing deleted records may still be accessible with time travel until a vacuum command is used to remove invalidated data files from physical storage.

NEW QUESTION: 99

A data engineer wants to refactor the following DLT code, which includes multiple table definitions with very similar code.

```
@dlt.table(name=f"t1_dataset")
def t1_dataset():
    return spark.read.table(t1)

@dlt.table(name=f"t2_dataset")
def t2_dataset():
    return spark.read.table(t2)

@dlt.table(name=f"t3_dataset")
def t3_dataset():
    return spark.read.table(t3)

...
```

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

```
tables = ["t1", "t2", "t3"]

for t in tables:
    @dlt.table(name=f"{t}_dataset")
        def new_table():
            return spark.read.table(t)
```


The pipeline runs an update with this refactored code, but generates a different DAG showing incorrect configuration values for these tables.

How can the data engineer fix this?

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

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

C. Move the table definition into a separate function, and make calls to this function using different input parameters inside the for loop.

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

Answer: C (LEAVE A REPLY)

In the provided refactored code, the for loop dynamically attempts to define multiple tables, but the use of a loop within the DLT (@dlt.table) decorator does not work properly because it results in a single function reference being overwritten for each iteration. This leads to an incorrect DAG because all the table definitions end up pointing to the last iteration of the loop.

NEW QUESTION: 100

A data engineer is performing a join operating to combine values from a static userlookup table with a streaming DataFrame streamingDF.

Which code block attempts to perform an invalid stream-static join?

A. userLookup.join(streamingDF, ["userid"], how="inner")

B. streamingDF.join(userLookup, ["user_id"], how="outer")

C. streamingDF.join(userLookup, ["user_id"], how="left")

D. streamingDF.join(userLookup, ["userid"], how="inner")

E. userLookup.join(streamingDF, ["user_id"], how="right")

Answer: (SHOW ANSWER)

https://spark.apache.org/docs/latest/structured-streaming-programming-guide.html#support- matrix-for-joins-in-streaming-queries

NEW QUESTION: 101

Which statement describes a key benefit of an end-to-end test?

A. It closely simulates real world usage of your application.

B. It pinpoint errors in the building blocks of your application.

C. It provides testing coverage for all code paths and branches.

D. It makes it easier to automate your test suite

Answer: (SHOW ANSWER)

End-to-end testing is a methodology used to test whether the flow of an application, from start to finish, behaves as expected. The key benefit of an end-to-end test is that it closely simulates real- world, user behavior, ensuring that the system as a whole operates correctly.
Get Latest & Actual Certified-Data-Engineer-Professional Exam's Question and Answers from

NEW QUESTION: 102
A member of the data engineering team has submitted a short notebook that they wish to schedule as part of a larger data pipeline. Assume that the commands provided below produce the logically correct results when run as presented.

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

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

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

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

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

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

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


Which command should be removed from the notebook before scheduling it as a job?
A. Cmd 2
B. Cmd 3
C. Cmd 4
D. Cmd 5
E. Cmd 6
Answer: E (LEAVE A REPLY)

When scheduling a Databricks notebook as a job, it's generally recommended to remove or modify commands that involve displaying output, such as using the display() function. Displaying data using display() is an interactive feature designed for exploration and visualization within the notebook interface and may not work well in a production job context.
The finalDF.explain() command, which provides the execution plan of the DataFrame transformations and actions, is often useful for debugging and optimizing queries. While it doesn't display interactive visualizations like display(), it can still be informative for understanding how Spark is executing the operations on your DataFrame.

NEW QUESTION: 103
While reviewing a query's execution in the Databricks Query Profiler, a data engineer observes that the Top Operators panel shows a Sort operator with high Time Spent and Memory Peak metrics. The Spark UI also reports frequent data spilling. How should the data engineer address this issue?
A. Switch to a broadcast join to reduce memory usage.
B. Repartition the DataFrame to a single partition before sorting.
C. Convert the sort operation to a filter operation.
D. Increase the number of shuffle partitions to better distribute data.
Answer: D (LEAVE A REPLY)
Increasing the number of shuffle partitions distributes the data across more tasks, reducing per- task memory pressure during the sort operation. This helps mitigate spilling by lowering memory peak usage per task and improves overall sort performance in large-scale distributed queries.

NEW QUESTION: 104
A user wants to use DLT expectations to validate that a derived table report contains all records from the source, included in the table validation_copy.
The user attempts and fails to accomplish this by adding an expectation to the report table definition.

```
CREATE LIVE TABLE report (
  CONSTRAINT no_missing_records EXPECT (key IN validation_copy)
)
AS SELECT <...>
```

Which approach would allow using DLT expectations to validate all expected records are present in this table?
A. Define a SQL UDF that performs a left outer join on two tables, and check if this returns null values for report key values in a DLT expectation for the report table.
B. Define a function that performs a left outer join on validation_copy and report and report, and check against the result in a DLT expectation for the report table
C. Define a temporary table that perform a left outer join on validation_copy and report, and define an expectation that no report key values are null
D. Define a view that performs a left outer join on validation_copy and report, and reference this view in DLT expectations for the report table
Answer: D (LEAVE A REPLY)
To validate that all records from the source are included in the derived table, creating a view that performs a left outer join between the validation_copy table and the report table is effective. The view can highlight any discrepancies, such as null values in the report

table's key columns, indicating missing records. This view can then be referenced in DLT (Delta Live Tables) expectations for the report table to ensure data integrity. This approach allows for a comprehensive comparison between the source and the derived table.

NEW QUESTION: 105
The data engineer team is configuring environment for development testing, and production before beginning migration on a new data pipeline. The team requires extensive testing on both the code and data resulting from code execution, and the team want to develop and test against similar production data as possible.
A junior data engineer suggests that production data can be mounted to the development testing environments, allowing pre production code to execute against production data. Because all users have Admin privileges in the development environment, the junior data engineer has offered to configure permissions and mount this data for the team.
Which statement captures best practices for this situation?
A. Because access to production data will always be verified using passthrough credentials it is safe to mount data to any Databricks development environment.
B. All developer, testing and production code and data should exist in a single unified workspace; creating separate environments for testing and development further reduces risks.
C. In environments where interactive code will be executed, production data should only be accessible with read permissions; creating isolated databases for each environment further reduces risks.
D. Because delta Lake versions all data and supports time travel, it is not possible for user error or malicious actors to permanently delete production data, as such it is generally safe to mount production data anywhere.
Answer: C (LEAVE A REPLY)
The best practice in such scenarios is to ensure that production data is handled securely and with proper access controls. By granting only read access to production data in development and testing environments, it mitigates the risk of unintended data modification. Additionally, maintaining isolated databases for different environments helps to avoid accidental impacts on production data and systems.

NEW QUESTION: 106
The data engineer team has been tasked with configured connections to an external database that does not have a supported native connector with Databricks. The external database already has data security configured by group membership. These groups map directly to user group already created in Databricks that represent various teams within the company. A new login credential has been created for each group in the external database. The Databricks Utilities Secrets module will be used to make these credentials available to Databricks users. Assuming that all the credentials are configured correctly on the external database and group membership is properly configured on Databricks, which statement describes how teams can be granted the minimum necessary access to using these credentials?
A. "Read" permissions should be set on a secret key mapped to those credentials that will be used by a given team.
B. No additional configuration is necessary as long as all users are configured as administrators in the workspace where secrets have been added.
C. "Read" permissions should be set on a secret scope containing only those credentials that will be used by a given team.
D. "Manage" permission should be set on a secret scope containing only those credentials that will be used by a given team.
Answer: (SHOW ANSWER)
In Databricks, using the Secrets module allows for secure management of sensitive information such as database credentials. Granting 'Read' permissions on a secret key that maps to database credentials for a specific team ensures that only members of that

team can access these credentials. This approach aligns with the principle of least privilege, granting users the minimum level of access required to perform their jobs, thus enhancing security.

NEW QUESTION: 107
A company stores account transactions in a Delta Lake table. The company needs to apply frequent account-level correlations (e.g., UPDATE statements) but wants to avoid rewriting entire Parquet files for each change to reduce file churn and improve write performance. Which Delta Lake feature should they enable?
A. Enable automatic file compaction on writes
B. Enable change data feed on the Delta table
C. Partition the Delta table by account_id
D. Enable deletion vectors on the Delta table
Answer: D (LEAVE A REPLY)
Deletion vectors allow Delta Lake to track row-level deletes and updates without rewriting entire Parquet files. By recording changes separately from the base files, this feature significantly reduces file churn and improves write performance for workloads with frequent row-level modifications such as account-level updates.

NEW QUESTION: 108
The business reporting tem 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 (LEAVE A REPLY)
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.

NEW QUESTION: 109

Given the following PySpark code snippet in a Databricks notebook:
filtered_df = spark.read.format("delta").load("/mnt/data/large_table")
\
.filter("event_date > '2024-01-01'")
filtered_df.count()
The data engineer notices from the Query Profiler that the scan operator for filtered_df is reading almost all files, despite the filter being applied.
What is the probable reason for poor data skipping?
A. The Delta table lacks optimization that enables dynamic file pruning.
B. The filter is executed only after the full data scan, preventing data skipping.
C. The event_date column is outside the table's partitioning and Z-ordering scheme.
D. The filter condition involves a data type excluded from data skipping support.
Answer: C (LEAVE A REPLY)
Delta Lake's data skipping relies on partitioning and clustering (such as Z-ordering) on the filtered columns. If event_date is neither a partition column nor included in the table's clustering strategy, Spark must scan most files because file-level statistics cannot be effectively used to prune irrelevant data.

NEW QUESTION: 110
An external object storage container has been mounted to the location /mnt/finance_eda_bucket.
The following logic was executed to create a database for the finance team:

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

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

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

If all users on the finance team are members of the finance group, which statement describes how the tx_sales table will be created?
A. A logical table will persist the query plan to the Hive Metastore in the Databricks control plane.
B. An external table will be created in the storage container mounted to /mnt/finance eda bucket.
C. A logical table will persist the physical plan to the Hive Metastore in the Databricks control plane.
D. An managed table will be created in the storage container mounted to /mnt/finance_eda_bucket.
E. A managed table will be created in the DBFS root storage container.
Answer: D (LEAVE A REPLY)
https://docs.databricks.com/en/data-governance/unity-catalog/create-schemas.html#language- SQL

NEW QUESTION: 111
A company wants to implement Lakehouse Federation across multiple data sources but is concerned about data consistency and ensuring that all teams access the same authoritative version of their data. Which statement is applicable for Lakehouse Federations to maintain data consistency?

A. Federation provides read-only access that reflects the current state of source systems.
B. Federation implements change data capture (CDC) from all sources.
C. A separate data synchronization service must be deployed.
D. Federation creates local copies that must be manually refreshed.
Answer: A (LEAVE A REPLY)
Lakehouse Federation allows Databricks to query and manage external data sources through a single governance layer, without moving or copying data. The documentation specifies that
"Federated queries provide read-only access to data, reflecting the current state of the underlying source system." This ensures consistency across teams since all users access the same source of truth directly from the external system through Unity Catalog. Federation does not perform CDC replication or local caching; it queries live data on demand. Hence, option A accurately represents how Lakehouse Federation maintains consistency across federated sources.

NEW QUESTION: 112
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 monitizable clicks.

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
Answer: (SHOW ANSWER)
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: 113

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.
Get Latest & Actual Certified-Data-Engineer-Professional Exam's Question and Answers from

Which statement correctly describes the outcome of executing these command cells in order in an interactive notebook?
A. Both commands will succeed. Executing show tables will show that countries at and sales at have been registered as views.
B. Cmd 1 will succeed. Cmd 2 will search all accessible databases for a table or view named countries af: if this entity exists, Cmd 2 will succeed.
C. Cmd 1 will succeed and Cmd 2 will fail, countries at will be a Python variable representing a PySpark DataFrame.
D. Both commands will fail. No new variables, tables, or views will be created.
E. Cmd 1 will succeed and Cmd 2 will fail, countries at will be a Python variable containing a list of strings.
Answer: (SHOW ANSWER)
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.

NEW QUESTION: 114
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. Incremental state information should be maintained for 10 minutes for late-arriving data.
Streaming DataFrame df has the following schema:
"device_id INT, event_time TIMESTAMP, temp FLOAT, humidity FLOAT"
Code block:

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

A. withWatermark("event_time", "10 minutes")

B. awaitArrival("event_time", "10 minutes")

C. await("event_time + `10 minutes'")

D. slidingWindow("event_time", "10 minutes")

E. delayWrite("event_time", "10 minutes")

Answer: A (LEAVE A REPLY)

This is because the question asks for incremental state information to be maintained for 10 minutes for late-arriving data. The withWatermark method is used to define the watermark for late data. The watermark is a timestamp column and a threshold that tells the system how long to wait for late data. In this case, the watermark is set to 10 minutes. The other options are incorrect because they are not valid methods or syntax for watermarking in Structured Streaming.

NEW QUESTION: 115

A data engineer needs to design an efficient pipeline that automatically processes new CSV files as they arrive in S3 storage. Which Databricks feature should the data engineer use to meet these requirements?

A. Streaming from cloud storage using standard Spark readStream with format ("csv") and format ("json")

B. COPY INTO SQL command with parameters to track processed files

C. Traditional batch processing with scheduled Databricks Jobs

D. Auto Loader with schema inference and evolution enabled

Answer: D (LEAVE A REPLY)

Auto Loader is designed to efficiently and incrementally process new files as they arrive in cloud object storage. It provides scalable file discovery, supports schema inference and evolution, and minimizes overhead compared to traditional batch or manual streaming approaches.

NEW QUESTION: 116

A user new to Databricks is trying to troubleshoot long execution times for some pipeline logic they are working on. Presently, the user is executing code cell-by-cell, using display() calls to confirm code is producing the logically correct results as new transformations are added to an operation. To get a measure of average time to execute, the user is running each cell multiple times interactively.

Which of the following adjustments will get a more accurate measure of how code is likely to perform in production?

A. The Jobs Ul should be leveraged to occasionally run the notebook as a job and track execution time during incremental code development because Photon can only be enabled on clusters launched for scheduled jobs.
B. Scala is the only language that can be accurately tested using interactive notebooks; because the best performance is achieved by using Scala code compiled to JARs. all PySpark and Spark SQL logic should be refactored.
C. Production code development should only be done using an IDE; executing code against a local build of open source Spark and Delta Lake will provide the most accurate benchmarks for how code will perform in production.
D. Calling display () forces a job to trigger, while many transformations will only add to the logical query plan; because of caching, repeated execution of the same logic does not provide meaningful results.
E. The only way to meaningfully troubleshoot code execution times in development notebooks Is to use production-sized data and production-sized clusters with Run All execution.
Answer: (SHOW ANSWER)

NEW QUESTION: 117
A data team is working to optimize an existing large, fast-growing table 'orders' with high cardinality columns, which experiences significant data skew and requires frequent concurrent writes. The team notice that the columns 'user_id', 'event_timestamp' and 'product_id' are heavily used in analytical queries and filters, although those keys may be subject to change in the future due to different business requirements. Which partitioning strategy should the team choose to optimize the table for immediate data skipping, incremental management over time, and flexibility?
A. Partition the table with: ALTER TABLE orders PARTITION BY user_id, product_id, event_timestamp
B. Use z-order after partitiing the table: OPTIMIZE orders ZORDER BY (user_id, product_id) WHERE event_timestamp = current date () - 1 DAY
C. Cluster the table with: ALTER TABLE orders CLUSTER BY user_id, product_id, event_timestamp
D. Z-order the table with OPTIMIZE orders ZORDER BY (user_id, product_id, event_timestamp)
Answer: D (LEAVE A REPLY)
Z-ordering optimizes data skipping for selective queries on high-cardinality columns without physically repartitioning the table, making it flexible if query patterns change. Using OPTIMIZE ...
ZORDER BY (user_id, product_id, event_timestamp) improves query performance for filters and joins while allowing incremental writes, avoiding the data skew and maintenance overhead that explicit partitioning or clustering could introduce.

NEW QUESTION: 118
The data architect has decided that once data has been ingested from external sources into the Databricks Lakehouse, table access controls will be leveraged to manage permissions for all production tables and views.
The following logic was executed to grant privileges for interactive queries on a production database to the core engineering group.
GRANT USAGE ON DATABASE prod TO eng;
GRANT SELECT ON DATABASE prod TO eng;
Assuming these are the only privileges that have been granted to the eng group and that these users are not workspace administrators, which statement describes their privileges?
A. Group members have full permissions on the prod database and can also assign permissions to other users or groups.
B. Group members are able to list all tables in the prod database but are not able to see the results of any queries on those tables.
C. Group members are able to query and modify all tables and views in the prod database, but cannot create new tables or views.
D. Group members are able to query all tables and views in the prod database, but cannot create or edit anything in the database.

E. Group members are able to create, query, and modify all tables and views in the prod database, but cannot define custom functions.
Answer: (SHOW ANSWER)
The GRANT USAGE ON DATABASE prod TO eng command grants the eng group the permission to use the prod database, which means they can list and access the tables and views in the database. The GRANT SELECT ON DATABASE prod TO eng command grants the eng group the permission to select data from the tables and views in the prod database, which means they can query the data using SQL or DataFrame API. However, these commands do not grant the eng group any other permissions, such as creating, modifying, or deleting tables and views, or defining custom functions. Therefore, the eng group members are able to query all tables and views in the prod database, but cannot create or edit anything in the database.

NEW QUESTION: 119
A Delta table of weather records is partitioned by date and has the below schema:
date DATE, device_id INT, temp FLOAT, latitude FLOAT, longitude FLOAT
To find all the records from within the Arctic Circle, you execute a query with the below filter:
latitude > 66.3
Which statement describes how the Delta engine identifies which files to load?
A. All records are cached to an operational database and then the filter is applied
B. The Parquet file footers are scanned for min and max statistics for the latitude column
C. All records are cached to attached storage and then the filter is applied
D. The Delta log is scanned for min and max statistics for the latitude column
E. The Hive metastore is scanned for min and max statistics for the latitude column
Answer: D (LEAVE A REPLY)
This is the correct answer because Delta Lake uses a transaction log to store metadata about each table, including min and max statistics for each column in each data file. The Delta engine can use this information to quickly identify which files to load based on a filter condition, without scanning the entire table or the file footers. This is called data skipping and it can improve query performance significantly. Verified Reference: [Databricks Certified Data Engineer Professional], under "Delta Lake" section; [Databricks Documentation], under "Optimizations - Data Skipping" section.
In the Transaction log, Delta Lake captures statistics for each data file of the table. These statistics indicate per file:
- Total number of records
- Minimum value in each column of the first 32 columns of the table
- Maximum value in each column of the first 32 columns of the table
- Null value counts for in each column of the first 32 columns of the table When a query with a selective filter is executed against the table, the query optimizer uses these statistics to generate the query result. it leverages them to identify data files that may contain records matching the conditional filter.
For the SELECT query in the question, The transaction log is scanned for min and max statistics for the price column.

NEW QUESTION: 120
The data governance team has instituted a requirement that all tables containing Personal Identifiable Information (PH) must be clearly annotated. This includes adding column comments, table comments, and setting the custom table property "contains_pii" = true.
The following SQL DDL statement is executed to create a new table:

```
CREATE TABLE dev.pii_test
(id INT, name STRING COMMENT "PII")
COMMENT "Contains PII"
TBLPROPERTIES ('contains_pii' = True)
```

Which command allows manual confirmation that these three requirements have been met?

A. DESCRIBE EXTENDED dev.pii test

B. DESCRIBE DETAIL dev.pii test

C. SHOW TBLPROPERTIES dev.pii test

D. DESCRIBE HISTORY dev.pii test

E. SHOW TABLES dev

Answer: A (LEAVE A REPLY)

This is the correct answer because it allows manual confirmation that these three requirements have been met. The requirements are that all tables containing Personal Identifiable Information (PII) must be clearly annotated, which includes adding column comments, table comments, and setting the custom table property "contains_pii" = true. The DESCRIBE EXTENDED command is used to display detailed information about a table, such as its schema, location, properties, and comments. By using this command on the dev.pii_test table, one can verify that the table has been created with the correct column comments, table comment, and custom table property as specified in the SQL DDL statement.

NEW QUESTION: 121

A Delta Lake table representing metadata about content from user has the following schema:
user_id LONG, post_text STRING, post_id STRING, longitude FLOAT, latitude FLOAT, post_time TIMESTAMP, date DATE 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

E. latitude

Answer: A (LEAVE A REPLY)

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.

NEW QUESTION: 122
A data engineer needs to install the PyYAML Python package within an air-gapped Databricks environment. The workspace has no direct internet access to PyPI. The engineer has downloaded the .whl file locally and wants it available automatically on all new clusters. Which approach should the data engineer use?
A. Upload the PyYAML .whl file to the user home directory and create a cluster-scoped init script to install it.
B. Upload the PyYAML .whl file to a Unity Catalog Volume, ensure it's allow-listed, and create a cluster-scoped init script that installs it from that path.
C. Set up a private PyPI repository and install via pip index URL.
D. Add the .whl file to Databricks Git Repos and assume automatic installation.
Answer: (SHOW ANSWER)
For secure, air-gapped Databricks deployments, the recommended practice is to host dependency files such as .whl packages in Unity Catalog Volumes -- a managed storage layer governed by Unity Catalog.
Once stored in a volume, these files can be safely referenced from cluster-scoped init scripts, which automatically execute installation commands (e.g., pip install
/Volumes/catalog/schema/path/PyYAML.whl) during cluster startup.
This ensures consistent environment setup across clusters and compliance with data governance rules.
User directories (A) lack enterprise security controls; private repositories (C) are not viable in air- gapped setups; and Git repos (D) do not trigger package installation. Therefore, B is the correct and officially approved method.