Databricks-Certified-Professional-Data-Engineer Dumps

Databricks Certified Data Engineer Professional Exam

https://www.certleader.com/Databricks-Certified-Professional-Data-Engineer-dumps.html

NEW QUESTION 1
A Databricks job has been configured with 3 tasks, each of which is a Databricks notebook. Task A does not depend on other tasks. Tasks B and C run in parallel, with each having a serial dependency on Task A.
If task A fails during a scheduled run, which statement describes the results of this run?

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

Answer: D

Explanation:
When a Databricks job runs multiple tasks with dependencies, the tasks are executed in a dependency graph. If a task fails, the downstream tasks that depend on it are skipped and marked as Upstream failed. However, the failed task may have already committed some changes to the Lakehouse before the failure occurred, and those changes are not rolled back automatically. Therefore, the job run may result in a partial update of the Lakehouse. To avoid this, you can use the transactional writes feature of Delta Lake to ensure that the changes are only committed when the entire job run succeeds.
Alternatively, you can use the Run if condition to configure tasks to run even when some or all of their dependencies have failed, allowing your job to recover from failures and
continue running. References:
? transactional writes: https://docs.databricks.com/delta/delta- intro.html#transactional-writes
? Run if: https://docs.databricks.com/en/workflows/jobs/conditional-tasks.html

NEW QUESTION 2
Review the following error traceback:
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: E

Explanation:
The error being raised is an AnalysisException, which is a type of exception that occurs when Spark SQL cannot analyze or execute a query due to some logical or semantic error1. In this case, the error message indicates that the query cannot resolve the column name 'heartrateheartrateheartrate' given the input columns 'heartrate' and 'age'. This means that there is no column in the table named 'heartrateheartrateheartrate', and the query is invalid. A possible cause of this error is a typo or a copy-paste mistake in the query. To fix this error, the query should use a valid column name that exists in the table, such as 'heartrate'.
References: AnalysisException

NEW QUESTION 3
A junior data engineer has been asked to develop a streaming data pipeline with a grouped aggregation using DataFrame df. The pipeline needs to calculate the average humidity and average temperature for each non-overlapping five-minute interval. Events are recorded once per minute per device.
Streaming DataFrame df has the following schema:
"device_id INT, event_time TIMESTAMP, temp FLOAT, humidity FLOAT" Code block:
Choose the response that correctly fills in the blank within the code block to complete this task.

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

Answer: B

Explanation:
This is the correct answer because the window function is used to group streaming data by time intervals. The window function takes two arguments: a time column and a window duration. The window duration specifies how long each window is, and must be a multiple of 1 second. In this case, the window duration is "5 minutes", which means each window will cover a non-overlapping five-minute interval. The window function also returns a struct column with two fields: start and end, which represent the start and end time of each window. The alias function is used to rename the struct column as "time". Verified References: [Databricks Certified Data Engineer Professional], under "Structured Streaming" section; Databricks Documentation, under "WINDOW" section.
https://www.databricks.com/blog/2017/05/08/event-time-aggregation-watermarking-apache-sparks-structured-streaming.html

NEW QUESTION 4
A data ingestion task requires a one-TB JSON dataset to be written out to Parquet with a target part-file size of 512 MB. Because Parquet is being used instead of Delta Lake, built- in file-sizing features such as Auto-Optimize & Auto-Compaction cannot be used.
Which strategy will yield the best performance without shuffling data?

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

Answer: B

Explanation:
 The key to efficiently converting a large JSON dataset to Parquet files of a specific size without shuffling data lies in controlling the size of the output files directly.
? Setting spark.sql.files.maxPartitionBytes to 512 MB configures Spark to process
data in chunks of 512 MB. This setting directly influences the size of the part-files in the output, aligning with the target file size.
? Narrow transformations (which do not involve shuffling data across partitions) can
then be applied to this data.
? Writing the data out to Parquet will result in files that are approximately the size specified by spark.sql.files.maxPartitionBytes, in this case, 512 MB.
? The other options involve unnecessary shuffles or repartitions (B, C, D) or an incorrect setting for this specific requirement (E).
References:
? Apache Spark Documentation: Configuration - spark.sql.files.maxPartitionBytes
? Databricks Documentation on Data Sources: Databricks Data Sources Guide

NEW QUESTION 5
A junior data engineer seeks to leverage Delta Lake's Change Data Feed functionality to create a Type 1 table representing all of the values that have ever been valid for all rows in a bronze table created with the property delta.enableChangeDataFeed = true. They plan to execute the following code as a daily job:
Which statement describes the execution and results of running the above query multiple times?

A. Each time the job is executed, newly updated records will be merged into the target table, overwriting previous values with the same primary keys.
B. Each time the job is executed, the entire available history of inserted or updated records will be appended to the target table, resulting in many duplicate entries.
C. Each time the job is executed, the target table will be overwritten using the entire history of inserted or updated records, giving the desired result.
D. Each time the job is executed, the differences between the original and current versions are calculated; this may result in duplicate entries for some records.
E. Each time the job is executed, only those records that have been inserted or updated since the last execution will be appended to the target table giving the desired result.

Answer: B

Explanation:
 Reading table's changes, captured by CDF, using spark.read means that you are reading them as a static source. So, each time you run the query, all table's changes (starting from the specified startingVersion) will be read.

NEW QUESTION 6
A data engineer needs to capture pipeline settings from an existing in the workspace, and use them to create and version a JSON file to create a new pipeline.
Which command should the data engineer enter in a web terminal configured with the Databricks CLI?

A. Use the get command to capture the settings for the existing pipeline; remove the pipeline_id and rename the pipeline; use this in a create command
B. Stop the existing pipeline; use the returned settings in a reset command
C. Use the alone command to create a copy of an existing pipeline; use the get JSON command to get the pipeline definition; save this to git
D. Use list pipelines to get the specs for all pipelines; get the pipeline spec from the return results parse and use this to create a pipeline

Answer: A

Explanation:
 The Databricks CLI provides a way to automate interactions with Databricks services. When dealing with pipelines, you can use the databricks pipelines get --
pipeline-id command to capture the settings of an existing pipeline in JSON format. This JSON can then be modified by removing the pipeline_id to prevent
conflicts and renaming the pipeline to create a new pipeline. The modified JSON file can then be used with the databricks pipelines create command to create a
new pipeline with those settings. References:
? Databricks Documentation on CLI for Pipelines: Databricks CLI - Pipelines

NEW QUESTION 7
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?

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

Explanation:
 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.
References:
? Delta Lake Documentation on Time Travel: Delta Time Travel
? Delta Lake Versioning: Delta Lake Versioning Guide

NEW QUESTION 8
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 beaccessible 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

Explanation:
 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. References:
? Databricks best practices for securing data:
https://docs.databricks.com/security/index.html

NEW QUESTION 9
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

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

NEW QUESTION 10
A junior data engineer has configured a workload that posts the following JSON to the Databricks REST API endpoint 2.0/jobs/create.

```
{
  "name": "Ingest new data",
  "existing_cluster_id": "6015-954420-peace720",
  "notebook_task": {
    "notebook_path": "/Prod/ingest.py"
  }
}
```

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: E

Explanation:
 This is the correct answer because the JSON posted to the Databricks REST API endpoint 2.0/jobs/create defines a new job with a name, an existing cluster id, and a notebook task. However, it does not specify any schedule or trigger for the job execution. Therefore, three new jobs with the same name and configuration will be created in the workspace, but none of them will be executed until they are manually triggered or scheduled. Verified References: [Databricks Certified Data Engineer Professional], under "Monitoring & Logging" section; [Databricks Documentation], under "Jobs API - Create" section.

NEW QUESTION 10
A Databricks job has been configured with 3 tasks, each of which is a Databricks notebook. Task A does not depend on other tasks. Tasks B and C run in parallel, with each having a serial dependency on task A.
If tasks A and B complete successfully but task C fails during a scheduled run, which statement describes the resulting state?

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

task failure.
C. All logic expressed in the notebook associated with task A will have been successfully completed; tasks B and C will not commit any changes because of stage failure.
D. Because all tasks are managed as a dependency graph, no changes will be committed to the Lakehouse until ail tasks have successfully been completed.
E. Unless all tasks complete successfully, no changes will be committed to the Lakehouse; because task C failed, all commits will be rolled back automatically.

Answer: A

Explanation:
 The query uses the CREATE TABLE USING DELTA syntax to create a Delta Lake table from an existing Parquet file stored in DBFS. The query also uses the LOCATION keyword to specify the path to the Parquet file as /mnt/finance_eda_bucket/tx_sales.parquet. By using the LOCATION keyword, the query creates an external table, which is a table that is stored outside of the default warehouse directory and whose metadata is not managed by Databricks. An external table can be created from an existing directory in a cloud storage system, such as DBFS or S3, that contains data files in a supported format, such as Parquet or CSV. The resulting state after running the second command is that an external table will be created in the storage container mounted to /mnt/finance_eda_bucket with the new name prod.sales_by_store. The command will not change any data or move any files in the storage container; it will only update the table reference in the metastore and create a new Delta transaction log for the renamed table. Verified References: [Databricks Certified Data Engineer Professional], under "Delta Lake" section; Databricks Documentation, under "ALTER TABLE RENAME TO" section; Databricks Documentation, under "Create an external table" section.

NEW QUESTION 13
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

Explanation:
 https://spark.apache.org/docs/3.1.3/api/python/reference/api/pyspark.sql.functions.broadca st.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.
References:
? Databricks Documentation on Broadcast Joins: Databricks Broadcast Join Guide
? PySpark API Reference: pyspark.sql.functions.broadcast

NEW QUESTION 14
A junior data engineer is migrating a workload from a relational database system to the Databricks Lakehouse. The source system uses a star schema, leveraging foreign key constrains and multi-table inserts to validate records on write.
Which consideration will impact the decisions made by the engineer while migrating this workload?

A. All Delta Lake transactions are ACID compliance against a single table, and Databricks does not enforce foreign key constraints.
B. Databricks only allows foreign key constraints on hashed identifiers, which avoid collisions in highly-parallel writes.
C. Foreign keys must reference a primary key field; multi-table inserts must leverage Delta Lake's upsert functionality.
D. Committing to multiple tables simultaneously requires taking out multiple table locks and can lead to a state of deadlock.

Answer: A

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

NEW QUESTION 15
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.

Answer: D

Explanation:
 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 16
Which REST API call can be used to review the notebooks configured to run as tasks in a multi-task job?

A. /jobs/runs/list
B. /jobs/runs/get-output
C. /jobs/runs/get
D. /jobs/get
E. /jobs/list

Answer: D

Explanation:
This is the correct answer because it is the REST API call that can be used to review the notebooks configured to run as tasks in a multi-task job. The REST API is an interface that allows programmatically interacting with Databricks resources, such as clusters, jobs, notebooks, or tables. The REST API uses HTTP methods, such as GET, POST, PUT, or DELETE, to perform operations on these resources. The /jobs/get endpoint is a GET method that returns information about a job given its job ID. The information includes the job settings, such as the name, schedule, timeout, retries, email notifications, and tasks. The tasks are the units of work that a job executes. A task can be a notebook task, which runs a notebook with specified parameters; a jar task, which runs a JAR uploaded to DBFS with specified main class and arguments; or a python task, which runs a Python file uploaded to DBFS with specified parameters. A multi-task job is a job that has more than one task configured to run in a specific order or in parallel. By using the /jobs/get endpoint, one can review the notebooks configured to run as tasks in a multi-task job.
Verified References: [Databricks Certified Data Engineer Professional], under "Databricks Jobs" section; Databricks Documentation, under "Get" section; Databricks Documentation, under "JobSettings" section.

NEW QUESTION 20
A junior data engineer has manually configured a series of jobs using the Databricks Jobs UI. Upon reviewing their work, the engineer realizes that they are listed as the "Owner" for each job. They attempt to transfer "Owner" privileges to the "DevOps" group, but cannot successfully accomplish this task.
Which statement explains what is preventing this privilege transfer?

A. Databricks jobs must have exactly one owner; "Owner" privileges cannot be assigned to a group.
B. The creator of a Databricks job will always have "Owner" privileges; this configuration cannot be changed.
C. Other than the default "admins" group, only individual users can be granted privileges on jobs.
D. A user can only transfer job ownership to a group if they are also a member of that group.
E. Only workspace administrators can grant "Owner" privileges to a group.

Answer: E

Explanation:
The reason why the junior data engineer cannot transfer "Owner" privileges to the "DevOps" group is that Databricks jobs must have exactly one owner, and the owner must be an individual user, not a group. A job cannot have more than one owner, and a job cannot have a group as an owner. The owner of a job is the user who created the job, or the user who was assigned the ownership by another user. The owner of a job has the highest level of permission on the job, and can grant or revoke permissions to other users or groups. However, the owner cannot transfer the ownership to a group, only to another user. Therefore, the junior data engineer's attempt to transfer "Owner" privileges to the "DevOps" group is not possible. References:
? Jobs access control: https://docs.databricks.com/security/access-control/table-acls/index.html
? Job permissions: https://docs.databricks.com/security/access-control/table-acls/privileges.html#job-permissions

NEW QUESTION 21
A Delta Lake table was created with the below query:
Consider the following query:
DROP TABLE prod.sales_by_store -
If this statement is executed by a workspace admin, which result will occur?

A. Nothing will occur until a COMMIT command is executed.
B. The table will be removed from the catalog but the data will remain in storage.
C. The table will be removed from the catalog and the data will be deleted.
D. An error will occur because Delta Lake prevents the deletion of production data.
E. Data will be marked as deleted but still recoverable with Time Travel.

Answer: C

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

NEW QUESTION 23
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 jo to execute the pipeline once and 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 hour on a new job cluster.
D. Configure a job that executes every time new data lands in a given directory.

Answer: C

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

NEW QUESTION 26
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

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

NEW QUESTION 27
A Delta Lake table representing metadata about content posts from users has the following schema:
user_id LONG, post_text STRING, post_id STRING, longitude FLOAT, latitude FLOAT, post_time TIMESTAMP, date DATE
This table is partitioned by the date column. A query is run with the following filter: longitude < 20 & longitude > -20
Which statement describes how data will be filtered?

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

Answer: D

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

NEW QUESTION 32
What is a method of installing a Python package scoped at the notebook level to all nodes in the currently active cluster?

A. Use &Pip install in a notebook cell
B. Run source env/bin/activate in a notebook setup script
C. Install libraries from PyPi using the cluster UI
D. Use &sh install in a notebook cell

Answer: C

Explanation:
Installing a Python package scoped at the notebook level to all nodes in the currently active cluster in Databricks can be achieved by using the Libraries tab in the cluster UI. This interface allows you to install libraries across all nodes in the cluster. While the %pip command in a notebook cell would only affect the driver node, using the cluster UI ensures that the package is installed on all nodes.
References:
? Databricks Documentation on Libraries: Libraries

NEW QUESTION 34
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.write.mode("append").saveAsTable("churn_preds")
B) preds.write.format("delta").save("/preds/churn_preds")
C)

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

D)

```
(preds.write
    .format("delta")
    .mode("overwrite")
    .saveAsTable("churn_preds")
)
```

E)

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

A. Option A
B. Option B
C. Option C
D. Option D
E. Option E

Answer: A

NEW QUESTION 35
A data engineer, User A, has promoted a new pipeline to production by using the REST API to programmatically create several jobs. A DevOps engineer, User B, has configured an external orchestration tool to trigger job runs through the REST API. Both users authorized the REST API calls using their personal access tokens.
Which statement describes the contents of the workspace audit logs concerning these events?

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

Answer: C

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

NEW QUESTION 39
A team of data engineer are adding tables to a DLT pipeline that contain repetitive expectations for many of the same data quality checks.
One member of the team suggests reusing these data quality rules across all tables defined for this pipeline.
What approach would allow them to do this?

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

Answer: A

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

NEW QUESTION 40
Spill occurs as a result of executing various wide transformations. However, diagnosing spill requires one to proactively look for key indicators.
Where in the Spark UI are two of the primary indicators that a partition is spilling to disk?

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

Answer: B

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

NEW QUESTION 43
The data engineer is using Spark's MEMORY_ONLY storage level.
Which indicators should the data engineer look for in the spark UI's Storage tab to signal that a cached table is not performing optimally?

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

Answer: C

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

NEW QUESTION 47
The DevOps team has configured a production workload as a collection of notebooks scheduled to run daily using the Jobs UI. A new data engineering hire is onboarding to the team and has requested access to one of these notebooks to review the production logic.
What are the maximum notebook permissions that can be granted to the user without allowing accidental changes to production code or data?

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

Answer: D

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

NEW QUESTION 51
The data architect has mandated that all tables in the Lakehouse should be configured as external (also known as "unmanaged") Delta Lake tables.
Which approach will ensure that this requirement is met?

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

Answer: D

Explanation:
 To create an external or unmanaged Delta Lake table, you need to use the EXTERNAL keyword in the CREATE TABLE statement. This indicates that the table is not managed by the catalog and the data files are not deleted when the table is dropped. You also need to provide a LOCATION clause to specify the path where the data files are stored. For example:
CREATE EXTERNAL TABLE events (date DATE, eventId STRING, eventType STRING, data STRING) USING DELTA LOCATION '/mnt/delta/events';
This creates an external Delta Lake table named events that references the data files in the '/mnt/delta/events' path. If you drop this table, the data files will remain intact and you can recreate the table with the same statement.
References:
? https://docs.databricks.com/delta/delta-batch.html#create-a-table
? https://docs.databricks.com/delta/delta-batch.html#drop-a-table

NEW QUESTION 56
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

Answer: B

Explanation:
 When viewing a Databricks Python notebook in a text editor, the first line indicates the format and source type of the notebook. The correct option is % Databricks notebook source, which is a magic command that specifies the start of a Databricks notebook source file.

NEW QUESTION 58
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 abronze 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: B

Explanation:
 This is the correct answer because it meets the requirements of maintaining a full record of all values that have ever been valid in the source system and recreating the current table state with only the most recent value for each record. The code ingests all log information into a bronze table, which preserves the raw CDC data as it is. Then, it uses merge into to perform an upsert operation on a silver table, which means it will insert new records or update or delete existing records based on the change type and the pk_id columns. This way, the silver table will always reflect the current state of the source table, while the bronze table will keep the history of all changes. Verified References: [Databricks Certified Data Engineer Professional], under "Delta Lake" section; Databricks Documentation, under "Upsert into a table using merge" section.

NEW QUESTION 63
The following code has been migrated to a Databricks notebook from a legacy workload:

```
%sh

git clone https://github.com/foo/data_loader;

python ./data_loader/run.py;

mv ./output /dbfs/mnt/new_data
```

The code executes successfully and provides the logically correct results, however, it takes over 20 minutes to extract and load around 1 GB of data.
Which statement is a possible explanation for this behavior?

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

Answer: E

Explanation:
 https://www.databricks.com/blog/2020/08/31/introducing-the-databricks-web- terminal.html
The code is using %sh to execute shell code on the driver node. This means that the code is not taking advantage of the worker nodes or Databricks optimized

Spark. This is why the code is taking longer to execute. A better approach would be to use Databricks libraries and APIs to read and write data from Git and DBFS, and to leverage the parallelism and performance of Spark. For example, you can use the Databricks Connect feature to run your Python code on a remote Databricks cluster, or you can use the Spark Git Connector to read data from Git repositories as Spark DataFrames.

NEW QUESTION 68
A data pipeline uses Structured Streaming to ingest data from kafka to Delta Lake. Data is being stored in a bronze table, and includes the Kafka_generated timesamp, key, and value. Three months after the pipeline is deployed the data engineering team has noticed some latency issued during certain times of the day.
A senior data engineer updates the Delta Table's schema and ingestion logic to include the current timestamp (as recoded by Apache Spark) as well the Kafka topic and partition. The team plans to use the additional metadata fields to diagnose the transient processing delays:
Which limitation will the team face while diagnosing this problem?

A. New fields not be computed for historic records.
B. Updating the table schema will invalidate the Delta transaction log metadata.
C. Updating the table schema requires a default value provided for each file added.
D. Spark cannot capture the topic partition fields from the kafka source.

Answer: A

Explanation:
When adding new fields to a Delta table's schema, these fields will not be retrospectively applied to historical records that were ingested before the schema change. Consequently, while the team can use the new metadata fields to investigate transient processing delays moving forward, they will be unable to apply this diagnostic approach to past data that lacks these fields.
References:
? Databricks documentation on Delta Lake schema management: https://docs.databricks.com/delta/delta-batch.html#schema-management

NEW QUESTION 69
Which Python variable contains a list of directories to be searched when trying to locate required modules?

A. importlib.resource path
B. ,sys.path
C. os-path
D. pypi.path
E. pylib.source

Answer: B

NEW QUESTION 73
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.
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

Explanation:
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.
References:
? Databricks Documentation on Delta Live Tables and Expectations: Delta Live Tables Expectations

NEW QUESTION 75
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

Explanation:
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 References: [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 79

The data engineering team maintains the following code:

```
accountDF = spark.table("accounts")
orderDF = spark.table("orders")
itemDF = spark.table("items")

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

    itemDF.itemName))

finalDF = (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

Explanation:

 This is the correct answer because it describes what will occur when this code is executed. The 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. Verified References: [Databricks Certified Data Engineer Professional], under "Delta Lake" section; Databricks Documentation, under "Write to Delta tables" section.

NEW QUESTION 84

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

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

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

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

A. Because the vacuum command permanently deletes all files containing deleted records, deleted records may be accessible with time travel for around 24 hours.
B. Because the default data retention threshold is 24 hours, data files containing deleted records will be retained until the vacuum job is run the following day.
C. Because Delta Lake time travel provides full access to the entire history of a table, deleted records can always be recreated by users with full admin privileges.
D. Because Delta Lake's delete statements have ACID guarantees, deleted records will be permanently purged from all storage systems as soon as a delete job completes.
E. Because the default data retention threshold is 7 days, data files containing deleted records will be retained until the vacuum job is run 8 days later.

Answer: E

Explanation:

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

NEW QUESTION 89
The view updates represents an incremental batch of all newly ingested data to be inserted or updated in the customers table.
The following logic is used to process these records.
Which statement describes this implementation?

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

Answer: A

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

NEW QUESTION 91
The data engineering team has configured a Databricks SQL query and alert to monitor the values in a Delta Lake table. The recent_sensor_recordings table contains an identifying sensor_id alongside the timestamp and temperature for the most recent 5 minutes of recordings.
The below query is used to create the alert:

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

The query is set to refresh each minute and always completes in less than 10 seconds. The alert is set to trigger when mean (temperature) > 120. Notifications are triggered to be sent at most every 1 minute.
If this alert raises notifications for 3 consecutive minutes and then stops, which statement must be true?

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

Answer: E

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

NEW QUESTION 96
Which distribution does Databricks support for installing custom Python code packages?

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

Answer: D

NEW QUESTION 100
Which of the following is true of Delta Lake and the Lakehouse?

A. Because Parquet compresses data row by ro
B. strings will only be compressed when a character is repeated multiple times.
C. Delta Lake automatically collects statistics on the first 32 columns of each table which are leveraged in data skipping based on query filters.
D. Views in the Lakehouse maintain a valid cache of the most recent versions of source tables at all times.
E. Primary and foreign key constraints can be leveraged to ensure duplicate values are never entered into a dimension table.
F. Z-order can only be applied to numeric values stored in Delta Lake tables

Answer: B

Explanation:
https://docs.delta.io/2.0.0/table-properties.html
Delta Lake automatically collects statistics on the first 32 columns of each table, which are leveraged in data skipping based on query filters1. Data skipping is a performance optimization technique that aims to avoid reading irrelevant data from the storage layer1. By collecting statistics such as min/max values, null counts, and bloom filters, Delta Lake can efficiently prune unnecessary files or partitions from the query plan1. This can significantly improve the query performance and reduce the I/O cost.

The other options are false because:
? Parquet compresses data column by column, not row by row2. This allows for better compression ratios, especially for repeated or similar values within a column2.
? Views in the Lakehouse do not maintain a valid cache of the most recent versions of source tables at all times3. Views are logical constructs that are defined by a SQL query on one or more base tables3. Views are not materialized by default, which means they do not store any data, but only the query definition3. Therefore, views always reflect the latest state of the source tables when queried3. However, views can be cached manually using the CACHE TABLE or CREATE TABLE AS SELECT commands.
? Primary and foreign key constraints can not be leveraged to ensure duplicate values are never entered into a dimension table. Delta Lake does not support enforcing primary and foreign key constraints on tables. Constraints are logical rules that define the integrity and validity of the data in a table. Delta Lake relies on the application logic or the user to ensure the data quality and consistency.
? Z-order can be applied to any values stored in Delta Lake tables, not only numeric values. Z-order is a technique to optimize the layout of the data files by sorting them on one or more columns. Z-order can improve the query performance by clustering related values together and enabling more efficient data skipping. Z-order can be applied to any column that has a defined ordering, such as numeric, string, date, or boolean values.
References: Data Skipping, Parquet Format, Views, [Caching], [Constraints], [Z-Ordering]

NEW QUESTION 101
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 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

Explanation:
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. Verified References: [Databricks Certified Data Engineer Professional], under "Databricks Workspace" section; Databricks Documentation, under "Choose a region" section.

NEW QUESTION 104
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:
After the database was successfully created and permissions configured, a member of the finance team runs the following code:
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: A

Explanation:
https://docs.databricks.com/en/lakehouse/data-objects.html

NEW QUESTION 108
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

Explanation:
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.References: Databricks documentation on implementing SCDs using Delta Lake and the MERGE statement (https://docs.databricks.com/delta/delta-update.html#upsert-into-a-table-using-merge).

NEW QUESTION 113
......

Thank You for Trying Our Product

* 100% Pass or Money Back

    All our products come with a 90-day Money Back Guarantee.

* One year free update

    You can enjoy free update one year. 24x7 online support.

* Trusted by Millions

    We currently serve more than 30,000,000 customers.

* Shop Securely

    All transactions are protected by VeriSign!

100% Pass Your Databricks-Certified-Professional-Data-Engineer Exam with Our Prep Materials Via below:

https://www.certleader.com/Databricks-Certified-Professional-Data-Engineer-dumps.html