Question 1
Which of the statements are correct when choosing between lakehouse and Datawarehouse?

○ Traditional Data warehouses have special indexes which are optimized for Machine learning

○ Traditional Data warehouses can serve low query latency with high reliability for BI workloads

○ SQL support is only available for Traditional Datawarehouse’s, Lakehouses support Python and Scala

○ Traditional Data warehouses are the preferred choice if we need to support ACID, Lakehouse does not support ACID.

○ Lakehouse replaces the current dependency on data lakes and data warehouses uses an open standard storage format and supports low latency BI workloads.

Explanation
The lakehouse replaces the current dependency on data lakes and data warehouses for modern data companies that desire:
· Open, direct access to data stored in standard data formats.
· Indexing protocols optimized for machine learning and data science.
· Low query latency and high reliability for BI and advanced analytics.

Question 2

Where are Interactive notebook results stored in Databricks product architecture?

- ○ Data plane
- ○ Control plane
- ○ Data and Control plane
- ○ JDBC data source
- ○ Databricks web application

Explanation
The answer is Data and Control plane,
Interactive notebook results are stored in a combination of the control plane (partial results for presentation in the UI) and customer storage.
https://docs.microsoft.com/en-us/azure/databricks/getting-started/overview#--high-level-architecture

Question 3

Which of the following statements are true about a lakehouse?

○ Lakehouse only supports Machine learning workloads and Data warehouses support BI workloads

○ Lakehouse only supports end-to-end streaming workloads and Data warehouses support Batch workloads

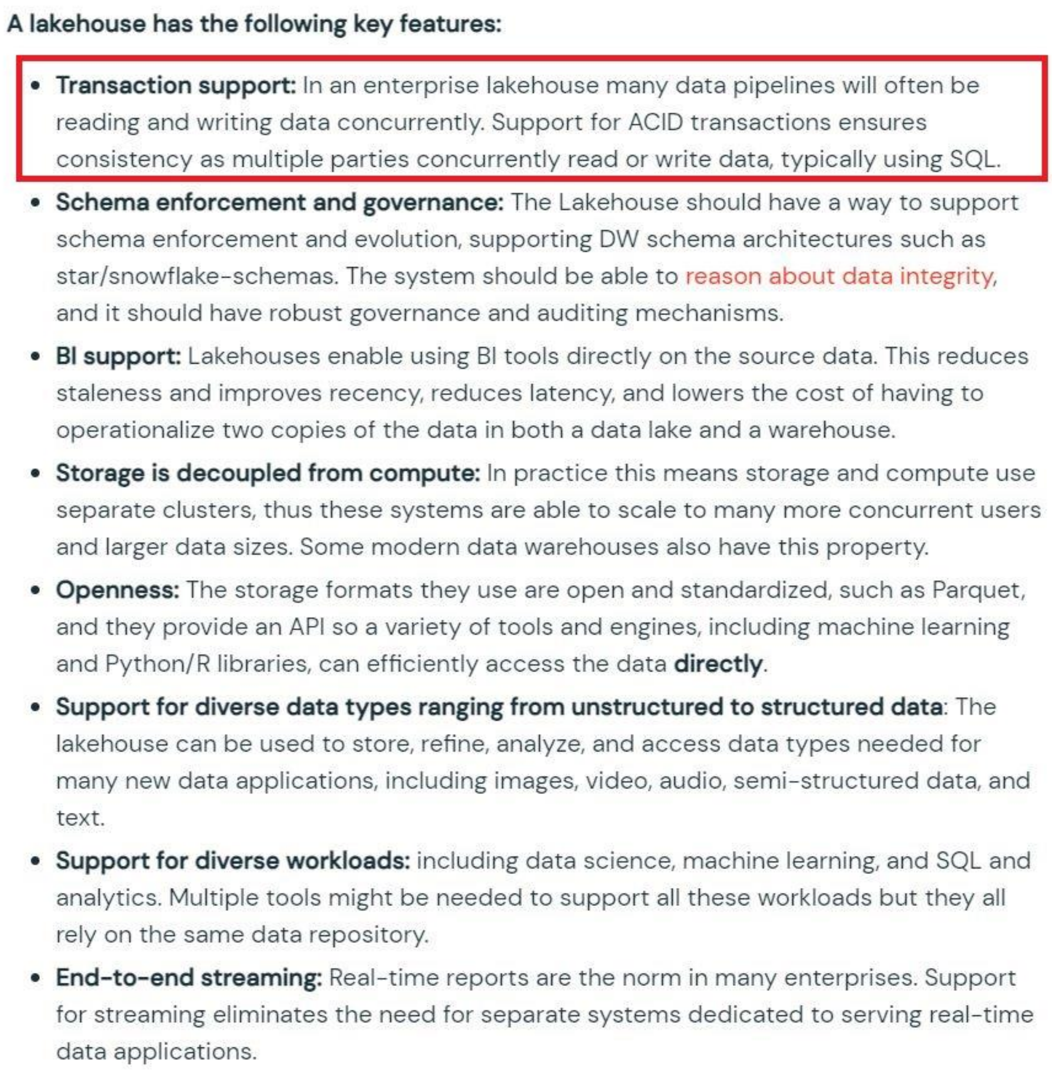
○ Lakehouse does not support ACID

○ Lakehouse do not support SQL

○ Lakehouse supports Transactions

Explanation

What Is a Lakehouse? - The Databricks Blog



**A lakehouse has the following key features:**

- **Transaction support:** In an enterprise lakehouse many data pipelines will often be reading and writing data concurrently. Support for ACID transactions ensures consistency as multiple parties concurrently read or write data, typically using SQL.
- **Schema enforcement and governance:** The Lakehouse should have a way to support schema enforcement and evolution, supporting DW schema architectures such as star/snowflake-schemas. The system should be able to reason about data integrity, and it should have robust governance and auditing mechanisms.
- **BI support:** Lakehouses enable using BI tools directly on the source data. This reduces staleness and improves recency, reduces latency, and lowers the cost of having to operationalize two copies of the data in both a data lake and a warehouse.
- **Storage is decoupled from compute:** In practice this means storage and compute use separate clusters, thus these systems are able to scale to many more concurrent users and larger data sizes. Some modern data warehouses also have this property.
- **Openness:** The storage formats they use are open and standardized, such as Parquet, and they provide an API so a variety of tools and engines, including machine learning and Python/R libraries, can efficiently access the data **directly**.
- **Support for diverse data types ranging from unstructured to structured data**: The lakehouse can be used to store, refine, analyze, and access data types needed for many new data applications, including images, video, audio, semi-structured data, and text.
- **Support for diverse workloads:** including data science, machine learning, and SQL and analytics. Multiple tools might be needed to support all these workloads but they all rely on the same data repository.
- **End-to-end streaming:** Real-time reports are the norm in many enterprises. Support for streaming eliminates the need for separate systems dedicated to serving real-time data applications.

Question 4

You had setup a new all-purpose cluster, but the cluster is unable to start which of the following steps do you need to take to resolve this issue?

- ○ Check the cluster driver logs
- ○ Check the cluster event logs
- ○ Workspace logs
- ○ Storage account
- ○ Data plane

Explanation

Cluster event logs are very useful, to identify issues pertaining to cluster availability.  Cluster may not start due to resource limitations or issues with cloud provider.

Question 5

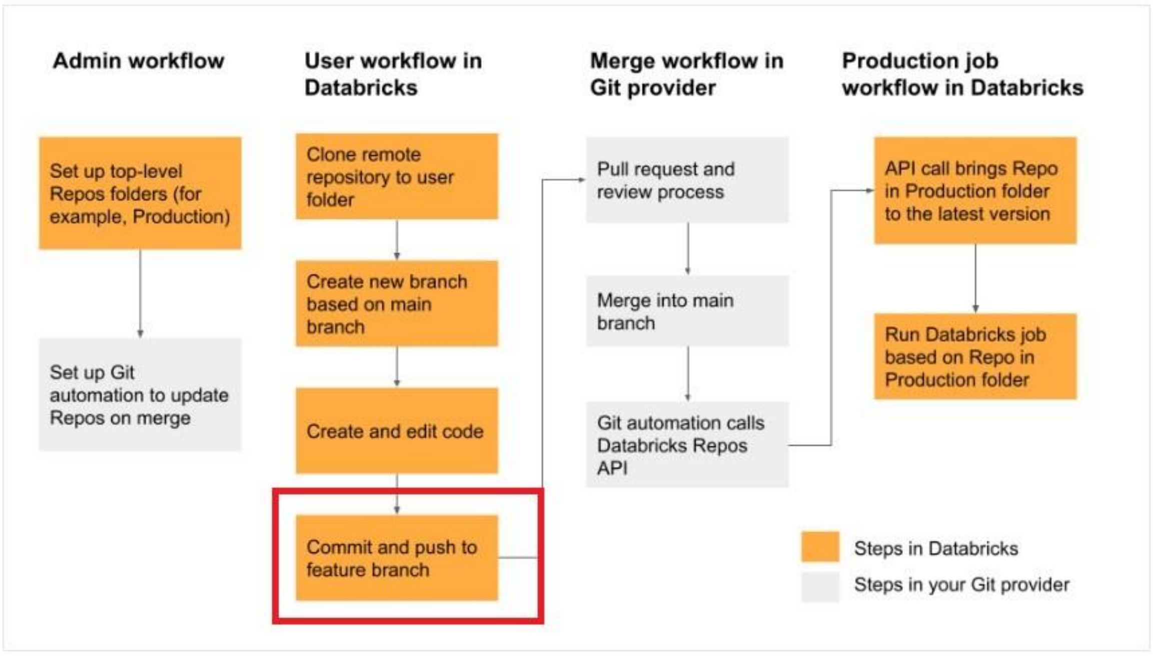
Which of the following developer operations in CI/CD flow can be implemented in Databricks Repos?

- ○ Delete branch
- ○ Trigger Databricks CICD pipeline
- ○ Commit and push code
- ○ Create a pull request
- ○ Approve the pull request

Explanation
The answer is Commit and push code.

See the below diagram to understand the role Databricks Repos and Git provider plays when building a CI/CD workflow.

All the steps highlighted in yellow can be done Databricks Repo, all the steps highlighted in Gray are done in a git provider like Github or Azure Devops


Admin workflow
User workflow in Databricks
Merge workflow in Git provider
Production job workflow in Databricks
Set up top-level Repos folders (for example, Production)
Set up Git automation to update Repos on merge
Clone remote repository to user folder
Create new branch based on main branch
Create and edit code
Commit and push to feature branch
Pull request and review process
Merge into main branch
Git automation calls Databricks Repos API
API call brings Repo in Production folder to the latest version
Run Databricks job based on Repo in Production folder
Steps in Databricks
Steps in your Git provider

Question 6

You noticed that a team member is using an all-purpose cluster to run a job every 30 mins, what would you recommend in reducing the compute cost.

- ○ Reduce the size of the cluster
- ○ Reduce the number of nodes and enable auto scale
- ○ Enable auto termination after 30 mins
- ○ Change the cluster all-purpose to job cluster
- ○ Change the cluster mode from all-purpose to single-mode

Explanation
Anytime you don't need to interact with a notebook, especially for a scheduled job use job cluster. Using all-purpose cluster can be twice as expensive as a job cluster.

Question 7

Which of the following commands can be used to run one notebook from another notebook?

- ◯ `notebook.utils.run("full notebook path")`
- ◯ `execute.utils.run("full notebook path")`
- ◯ `dbutils.notebook.run("full notebook path")`
- ◯ only job clusters can run notebook
- ◯ `spark.notebook.run("full notebook path")`

Explanation

The answer is `dbutils.notebook.run(" full notebook path ")`

Here is the full command with additional options.

`run(path: String, timeout_seconds: int, arguments: Map): String`

```
dbutils.notebook.run("ful-notebook-name", 60, {"argument": "data", "argument2":
"data2", ...})
```

Question 8

Which of the following SQL command can be used to insert or update or delete rows based on a condition to check if a row(s) exists?

- ○ `MERGE INTO table_name`
- ○ `COPY INTO table_name`
- ○ `UPDATE table_name`
- ○ `INSERT INTO OVERWRITE table_name`
- ○ `INSERT IF EXISTS table_name`

Explanation
here is the additional documentation for your review.
https://docs.databricks.com/spark/latest/spark-sql/language-manual/delta-merge-into.html

```
MERGE INTO target_table_name [target_alias]
   USING source_table_reference [source_alias]
   ON merge_condition
   [ WHEN MATCHED [ AND condition ] THEN matched_action ] [...]
   [ WHEN NOT MATCHED [ AND condition ]  THEN not_matched_action ] [...]

matched_action
 { DELETE |
   UPDATE SET * |
   UPDATE SET { column1 = value1 } [, ...] }

not_matched_action
 { INSERT * |
   INSERT (column1 [, ...] ) VALUES (value1 [, ...])
```

Question 9

When investigating a data issue, if you are looking compare the current version of the data with the previous version, what is the best way to query historical data?

- ○ `SELECT * FROM TIME_TRAVEL(table_name) where time_stamp = 'timestamp'`
- ○ `TIME_TRAVEL FROM table_name where time_stamp = 'timestamp'`
- ○ `SELECT * FROM table_name as of 'timestamp'`
- ○ `DISCRIBE HISTORY table_name as of 'timestmap'`
- ○ `SHOW HISTORY table_name as of 'timestmap'`

Explanation
The answer is `SELECT * FROM table_name as of 'timestamp'`

FYI, Time travel supports two ways one is using timestamp and the second way is using version number,

Timestamp:

```
SELECT count(*) FROM my_table TIMESTAMP AS OF "2019-01-01"
SELECT count(*) FROM my_table TIMESTAMP AS OF date_sub(current_date(), 1)
SELECT count(*) FROM my_table TIMESTAMP AS OF "2019-01-01 01:30:00.000"
```

Version Number:

```
SELECT count(*) FROM my_table VERSION AS OF 5238
SELECT count(*) FROM my_table@v5238
SELECT count(*) FROM delta.`/path/to/my/table@v5238`
```

[https://databricks.com/blog/2019/02/04/introducing-delta-time-travel-for-large-scale-data-lakes.html](https://databricks.com/blog/2019/02/04/introducing-delta-time-travel-for-large-scale-data-lakes.html)

Question 10

While querying the previous version of the table using time travel you realized you are no longer able to query the historical data?

- You currently do not have access to view historical data
- By default, historical data is cleaned every 180 days in DELTA
- A command `VACUUM table_name RETAIN 0` was ran
- Time travel is disabled
- Time travel must be enabled before you query previous data

Explanation
The answer is, VACUUM table_name RETAIN 0 was ran

Recursively vacuum directories associated with the Delta table and remove data files that are no longer in the latest state of the transaction log for the table and are older than a retention threshold. Default is 7 Days.
When VACUUM table_name RETAIN 0 is ran all of the historical versions of data are lost time travel can only provide the current state.

Question 11

You have accidentally deleted records from a table called transactions, what is the easiest way to restore the records deleted or the previous state of the table?

[✓ `RESTORE TABLE transactions FROM VERSION as of 'timestamp'`]

○ `RESTORE TABLE transactions TO VERSION as of 'timestamp'`

○
```
INSERT INTO OVERWRITE transactions
SELECT * FROM transactions as of ‘timestamp’
MINUS
SELECT * FROM transactions
```

○
```
INSERT INTO OVERWRITE transactions
SELECT * FROM transactions as of ‘timestamp’
INTERSECT
SELECT * FROM transactions
```

○ `COPY OVERWRITE transactions from VERSION as of 'timestamp'`

Explanation

[RESTORE (Databricks SQL) | Databricks on AWS]

```
RESTORE [TABLE] table_name [TO] time_travel_version
```

Time travel supports using timestamp or version number

```
time_travel_version
 { TIMESTAMP AS OF timestamp_expression |
   VERSION AS OF version }
```

`timestamp_expression` can be any one of:
`'2018-10-18T22:15:12.013Z'`, that is, a string that can be cast to a timestamp
`cast('2018-10-18 13:36:32 CEST' as timestamp)`
`'2018-10-18'`, that is, a date string
`current_timestamp() - interval 12 hours`
`date_sub(current_date(), 1)`
Any other expression that is or can be cast to a timestamp

Question 12

Create schema called bronze using location '/mnt/delta/bronze', check if schema exists before creating.

- ○ `CREATE SCHEMA IF NOT EXISTS bronze LOCATION '/mnt/delta/bronze'`
- ○ `CREATE SCHEMA bronze IF NOT EXISTS LOCATION '/mnt/delta/bronze'`
- ○ `if IS_SCHEMA('bronze'): CREATE SCHEMA bronze LOCATION '/mnt/delta/bronze'`
- ○ Schema creation is not available in metastore, it can only be done in Unity catalog UI
- ○ Cannot create schema without a database

Explanation
https://docs.databricks.com/sql/language-manual/sql-ref-syntax-ddl-create-schema.html

```
CREATE SCHEMA [ IF NOT EXISTS ] schema_name [ LOCATION schema_directory ]
```

Question 13

How do you check the location of an existing schema in Delta Lake?

○ Run SQL command `SHOW LOCATION schema_name`

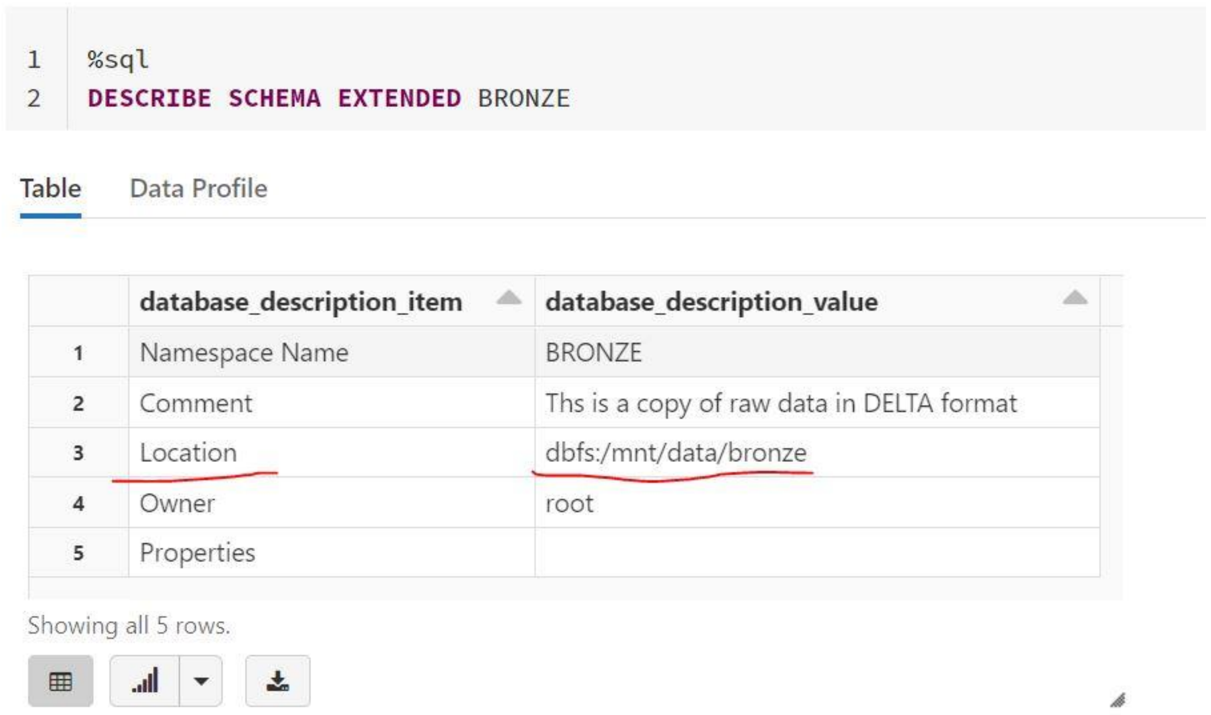
○ Check unity catalog UI

○ Use Data explorer

○ Run SQL command `DESCRIBE SCHEMA EXTENDED schema_name`

○ Schemas are internally in-store external hive meta stores like MySQL or SQL Server

Explanation
Here is an example of how it looks



```
%sql
DESCRIBE SCHEMA EXTENDED BRONZE
```

| | database_description_item | database_description_value |
|---|---|---|
| 1 | Namespace Name | BRONZE |
| 2 | Comment | Ths is a copy of raw data in DELTA format |
| 3 | Location | dbfs:/mnt/data/bronze |
| 4 | Owner | root |
| 5 | Properties | |

Showing all 5 rows.

Question 14

Which of the below SQL commands create a Global temporary view?

○
```
CREATE OR REPLACE TEMPORARY VIEW view_name
    AS SELECT * FROM table_name
```
○
```
 CREATE OR REPLACE LOCAL TEMPORARY VIEW view_name
    AS SELECT * FROM table_name
```
○
```
 CREATE OR REPLACE GLOBAL TEMPORARY VIEW view_name
    AS SELECT * FROM table_name
```
○
```
CREATE OR REPLACE VIEW view_name
    AS SELECT * FROM table_name
```
○
```
 CREATE OR REPLACE LOCAL VIEW view_name
    AS SELECT * FROM table_name
```

Explanation

```
CREATE OR REPLACE GLOBAL TEMPORARY VIEW view_name
   AS SELECT * FROM table_name
```

There are two types of temporary views that can be created Local and Global
A session-scoped temporary view is only available with a spark session, so another notebook in the same cluster can not access it. if a notebook is detached and reattached local temporary view is lost.
A global temporary view is available to all the notebooks in the cluster but if a cluster restarts a global temporary view is lost.

Question 15

When you drop a managed table using SQL syntax `DROP TABLE table_name` how does it impact metadata, history, and data stored in the table?

○ Drops table from meta store, drops metadata, history, and data in storage.

○ Drops table from meta store and data from storage but keeps metadata and history in storage

○ Drops table from meta store, meta data and history but keeps the data in storage

○ Drops table but keeps meta data, history and data in storage

○ Drops table and history but keeps meta data and data in storage

Explanation
For a managed table, a drop command will drop everything from metastore and storage.

Question 16

The team has decided to take advantage of table properties to identify a business owner for each table, which of the following table DDL syntax allows you to populate a table property identifying the business owner of a table

○
```
CREATE TABLE inventory (id INT, units FLOAT)
SET TBLPROPERTIES business_owner = 'supply chain'
```

○
```
CREATE TABLE inventory (id INT, units FLOAT)
TBLPROPERTIES (business_owner = 'supply chain')
```

○
```
CREATE TABLE inventory (id INT, units FLOAT)
SET (business_owner = ‘supply chain’)
```
○
```
CREATE TABLE inventory (id INT, units FLOAT)
SET PROPERTY (business_owner = ‘supply chain’)
```
○
```
CREATE TABLE inventory (id INT, units FLOAT)
SET TAG (business_owner = ‘supply chain’)
```

Explanation
```
CREATE TABLE inventory (id INT, units FLOAT) TBLPROPERTIES (business_owner
= ‘supply chain’)
```

Table properties and table options (Databricks SQL) | Databricks on AWS

Alter table command can used to update the TBLPROPERTIES

```
ALTER TABLE inventory SET TBLPROPERTIES(business_owner , 'operations')
```

Question 17

Data science team has requested they are missing a column in the table called average price, this can be calculated using units sold and sales amt, which of the following SQL statements allow you to reload the data with additional column

```
INSERT OVERWRITE sales
SELECT *, salesAmt/unitsSold as avgPrice FROM sales
```

```
CREATE OR REPALCE TABLE sales
AS SELECT *, salesAmt/unitsSold as avgPrice FROM sales
```

```
MERGE INTO sales USING (SELECT *, salesAmt/unitsSold as avgPrice FROM
sales)
```

```
OVERWRITE sales AS SELECT *, salesAmt/unitsSold as avgPrice FROM sales
```

```
COPY INTO SALES AS SELECT *, salesAmt/unitsSold as avgPrice FROM sales
```

Explanation

```
CREATE OR REPALCE TABLE sales
AS SELECT *, salesAmt/unitsSold as avgPrice FROM sales
```

The main difference between INSERT OVERWRITE and CREATE OR REPLACE TABLE(CRAS) is that CRAS can modify the schema of the table, i.e it can add new columns or change data types of existing columns. By default INSERT OVERWRITE only overwrites the data.
INSERT OVERWRITE can also be used to overwrite schema, only when `spark.databricks.delta.schema.autoMerge.enabled is set true` if this option is not enabled and if there is a schema mismatch command will fail.

Question 18

You are working on a process to load external CSV files into a Delta by leveraging the COPY INTO command, but after running the command for the second time no data was loaded into the table name, why is that?

```
COPY INTO table_name
FROM 'dbfs:/mnt/raw/*.csv'
FILEFORMAT = CSV
```

○ COPY INTO only works one time data load

○ Run REFRESH TABLE sales before running COPY INTO

○ COPY INTO did not detect new files after the last load

○ Use incremental = TRUE option to load new files

○ COPY INTO does not support incremental load, use AUTO LOADER

Explanation
The answer is COPY INTO did not detect new files after the last load,
COPY INTO keeps track of files that were successfully loaded into the table, the next time when the COPY INTO runs it skips them.

FYI, you can change this behavior by using COPY_OPTIONS 'force'= 'true', when this option is enabled all files in the path/pattern are loaded.

```
COPY INTO table_identifier
  FROM [ file_location | (SELECT identifier_list FROM file_location) ]
  FILEFORMAT = data_source
  [FILES = [file_name, ... | PATTERN = 'regex_pattern']
  [FORMAT_OPTIONS ('data_source_reader_option' = 'value', ...)]
  [COPY_OPTIONS 'force' = ('false'|'true')]
```

Question 19

What is the main difference between the below two commands?

```
INSERT OVERWRITE table_name
SELECT * FROM table
CREATE OR REPLACE TABLE table_name
AS SELECT * FROM table
```

○ `INSERT OVERWRITE` replaces data by default, `CREATE OR REPLACE` replaces data and Schema by default

○ `INSERT OVERWRITE` replaces data and schema by default, `CREATE OR REPLACE` replaces data by default

○ `INSERT OVERWRITE` maintains historical data versions by default, `CREATE OR REPLACE` clears the historical data versions by default

○ `INSERT OVERWRITE` clears historical data versions by default, `CREATE OR REPLACE` maintains the historical data versions by default

○ Both are same and results in identical outcomes

Explanation

The answer is, `INSERT OVERWRITE` replaces data, CRAS replaces data and Schema

The main difference between `INSERT OVERWRITE` and `CREATE OR REPLACE TABLE(CRAS)` is that CRAS can modify the schema of the table, i.e it can add new columns or change data types of existing columns. By default `INSERT OVERWRITE` only overwrites the data.
`INSERT OVERWRITE` can also be used to overwrite schema, only when `spark.databricks.delta.schema.autoMerge.enabled` is set `true` if this option is not enabled and if there is a schema mismatch command will fail.

Question 20

Which of the following functions can be used to convert JSON string to Struct data type?

○ TO_STRUCT (json value)

○ FROM_JSON (json value)

○ FROM_JSON (json value, schema of json)

○ CONVERT (json value, schema of json)

○ CAST (json value as STRUCT)

Explanation
Syntax
Copy
```
from_json(jsonStr, schema [, options])
```
Arguments
`jsonStr`: A STRING expression specifying a row of CSV data.
`schema`: A STRING literal or invocation of schema_of_json function (Databricks SQL).
`options`: An optional MAP<STRING,STRING> literal specifying directives.
Refer documentation for more details,
https://docs.microsoft.com/en-us/azure/databricks/sql/language-manual/functions/from_json

Question 21

You are working on a marketing team request to identify customers with same information between two tables CUSTOMERS_2021 and CUSTOMERS_2020 each table contains 25 columns with same schema, You are looking to identify rows match between two tables across all columns, which of the following can be used to perform in SQL

○
```
SELECT * FROM CUSTOMERS_2021
 UNION
SELECT * FROM CUSTOMERS_2020
```

○
```
SELECT * FROM CUSTOMERS_2021
 UNION ALL
SELECT * FROM CUSTOMERS_2020
```

○
```
SELECT * FROM CUSTOMERS_2021 C1
INNER JOIN CUSTOMERS_2020 C2
ON C1.CUSTOMER_ID = C2.CUSTOMER_ID
```

○
```
SELECT * FROM CUSTOMERS_2021
 INTERSECT
SELECT * FROM CUSTOMERS_2020
```

○
```
SELECT * FROM CUSTOMERS_2021
EXCEPT
SELECT * FROM CUSTOMERS_2020
```

Explanation
Answer is
```
SELECT * FROM CUSTOMERS_2021
 INTERSECT
SELECT * FROM CUSTOMERS_2020
```

INTERSECT [ALL | DISTINCT]

Returns the set of rows which are in both subqueries.
If `ALL` is specified a row that appears multiple times in the `subquery1` as well as in `subquery` will be returned multiple times.
If `DISTINCT` is specified the result does not contain duplicate rows. This is the default.

Question 22

You are looking to process the data based on two variables, one to check if the department is supply chain and second to check if process flag is set to True

○ `if department = “supply chain” & process:`

○ `if department == “supply chain” && process:`

○ `if department == “supply chain” & process == TRUE:`

○ `if department == “supply chain” & if process == TRUE:`

○ `if department == "supply chain" and process:`

Question 23

You were asked to create a notebook that can take department as a parameter and process the data accordingly, which is the following statements result in storing the notebook parameter into a python variable

○
```
SET department = dbutils.widget.get("department")
```
○
```
ASSIGN department == dbutils.widget.get("department")
```
○
```
department = dbutils.widget.get("department")
```

○
```
department = notebook.widget.get("department")
```
○
```
department = notebook.param.get("department")
```

Explanation
The answer is `department = dbutils.widget.get("department")`
Refer to additional documentation here
https://docs.databricks.com/notebooks/widgets.html

Question 24

Which of the following statements can successfully read the notebook widget and pass the python variable to a SQL statement in a Python notebook cell?

○
```
order_date  = dbutils.widgets.get("widget_order_date")

spark.sql(f"SELECT * FROM sales WHERE orderDate = 'f{order_date }'")
```
○
```
order_date  = dbutils.widgets.get("widget_order_date")

spark.sql(f"SELECT * FROM sales WHERE orderDate = 'order_date' ")
```
○
```
order_date  = dbutils.widgets.get("widget_order_date")

spark.sql(f"SELECT * FROM sales WHERE orderDate = '${order_date }' ")
```
○
```
order_date  = dbutils.widgets.get("widget_order_date")

spark.sql(f"SELECT * FROM sales WHERE orderDate = '{order_date}' ")
```
○
```
order_date  = dbutils.widgets.get("widget_order_date")

spark.sql("SELECT * FROM sales WHERE orderDate = order_date")
```

Question 25

For a spark stream process to read a delta table to event logs and create a summary table with customerId and the number of times the customerId is present in the event log and write a one-time micro batch, fill in the blanks to complete the query.

```
spark._________
  .format("delta")
  .table("events_log")
  .groupBy("customerId")
  .count()
  ._______
  .format("delta")
  .outputMode("complete")
  .option("checkpointLocation", "/tmp/delta/eventsByCustomer/_checkpoints/")
  .trigger(______)
  .table("target_table")
```

- writeStream, readStream, once
- readStream, writeStream, once
- writeStream, processingTime = once
- writeStream, readStream, once = True
- readStream, writeStream, once = True

Explanation
The answer is readStream, writeStream, once = True.

```
spark.readStream
  .format("delta")
  .table("events_log")
  .groupBy("customerId")
  .count()
  .writeStream
  .format("delta")
  .outputMode("complete")
  .option("checkpointLocation", "/tmp/delta/eventsByCustomer/_checkpoints/")
  .trigger(once = True)
  .table("target_table")
```

Question 26

You would like to build a streaming process to read from a Kafka queue and write to a Delta table every 15 minutes, what is the correct trigger option

○ `trigger("15 minutes")`

○ `trigger(process "15 minutes")`

○ `trigger(processingTime = 15)`

○ `trigger(processingTime = "15 Minutes")`

○ `trigger(15)`

Explanation
The answer is `trigger(processingTime = "15 Minutes")`

Triggers:

Unspecified
This is the default. This is equivalent to using processingTime="500ms"
*Fixed interval micro-batches .trigger(processingTime="2 minutes")*
The query will be executed in micro-batches and kicked off at the user-specified intervals
One-time micro-batch .trigger(once=True)
The query will execute a single micro-batch to process all the available data and then stop on its own
One-time micro-batch.trigger .trigger(availableNow=True) -- New feature a better version of (once=True)

Databricks supports `trigger(availableNow=True)` in Databricks Runtime 10.2 and above for Delta Lake and Auto Loader sources. This functionality combines the batch processing approach of trigger once with the ability to configure batch size, resulting in multiple parallelized batches that give greater control for right-sizing batches and the resultant files.

Question 27
Which of the following scenarios is the best fit for the AUTO LOADER solution?

○ Efficiently process new data incrementally from cloud object storage

○ Incrementally process new streaming from data into delta lake

○ Incrementally process new data from relational databases like MySQL

○ Efficiently copy data from data lake location to another data lake location

○ Efficiently move data incrementally from one delta table to another delta table

Explanation
Refer to more documentation here,
https://docs.microsoft.com/en-us/azure/databricks/ingestion/auto-loader

Question 28

You had AUTO LOADER to process millions of files a day and noticed slowness in load process, so you scaled up the Databricks cluster but realized the performance of the Auto loader is still not improving, what is the best way to resolve this.

○ AUTO LOADER is not suitable to process millions of files a day

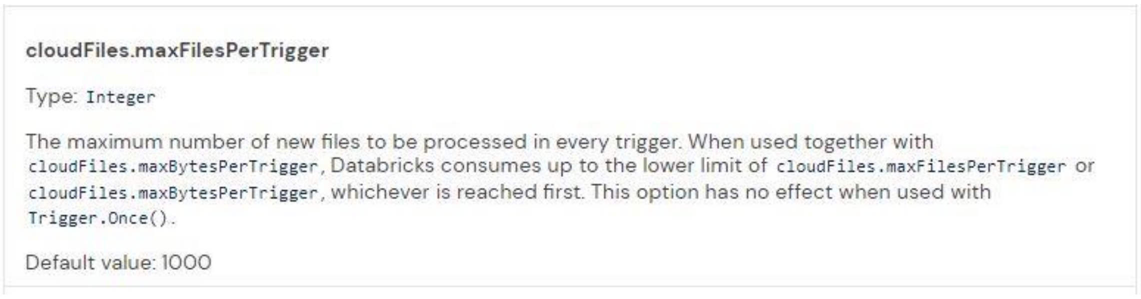
○ Setup a second AUTO LOADER process to process the data

○ Increase the maxFilesPerTrigger option to a sufficiently high number

○ Copy the data from cloud storage to local disk on the cluster for faster access

○ Merge files to one large file

Explanation
The default value of maxFilesPerTrigger is 1000 it can be increased to a much higher number but will require a much larger compute to process.



**cloudFiles.maxFilesPerTrigger**

Type: `Integer`

The maximum number of new files to be processed in every trigger. When used together with `cloudFiles.maxBytesPerTrigger`, Databricks consumes up to the lower limit of `cloudFiles.maxFilesPerTrigger` or `cloudFiles.maxBytesPerTrigger`, whichever is reached first. This option has no effect when used with `Trigger.Once()`.

Default value: 1000

https://docs.databricks.com/ingestion/auto-loader/options.html

Question 29
The current ELT pipeline is receiving data from the operations team once a day so you had setup an AUTO LOADER process to run once a day using trigger (Once = True) and scheduled a job to run once a day, operations team recently rolled out a new feature that allows them to send data every 1 min, what changes do you need to make to AUTO LOADER to process the data every 1 min.

○ Convert AUTO LOADER to structured streaming

○ Change AUTO LOADER trigger to .trigger(ProcessingTime = "1 minute")

○ Setup a job cluster run the notebook once a minute

○ Enable stream processing

○ Change AUTO LOADER trigger to ("1 minute")

Question 30

What is the purpose of bronze layer in a Multi hop architecture?

○ Copy of raw data, easy to query and ingest data for downstream processes.

○ Powers ML applications

○ Data quality checks, corrupt data quarantined

○ Contain aggregated data that is to be consumed into Silver

○ Reduces data storage by compressing the data

Explanation

Medallion Architecture – Databricks

Bronze Layer:
1. Raw copy of ingested data
2. Replaces traditional data lake
3. Provides efficient storage and querying of full, unprocessed history of data
4. No schema is applied at this layer

Question 31

What is the purpose of the silver layer in a Multi hop architecture?

○ Replaces a traditional data lake

○ Efficient storage and querying of full, unprocessed history of data

○ Eliminates duplicate data, quarantines bad data

○ Refined views with aggregated data

○ Optimized query performance for business-critical data

Explanation
Medallion Architecture – Databricks
Silver Layer:

1. Reduces data storage complexity, latency, and redundancy
2. Optimizes ETL throughput and analytic query performance
3. Preserves grain of original data (without aggregation)
4. Eliminates duplicate records
5. production schema enforced
6. Data quality checks, quarantine corrupt data

Question 32

What is the purpose of gold layer in Multi hop architecture?

○ Optimizes ETL throughput and analytic query performance

○ Eliminate duplicate records

○ Preserves grain of original data, without any aggregations

○ Data quality checks and schema enforcement

○ Optimized query performance for business-critical data

Explanation
Medallion Architecture – Databricks
Gold Layer:

1. Powers Ml applications, reporting, dashboards, ad hoc analytics
2. Refined views of data, typically with aggregations
3. Reduces strain on production systems
4. Optimizes query performance for business-critical data

Question 33

The Delta Live Tables Pipeline is configured to run in Development mode using the Triggered Pipeline Mode. what is the expected outcome after clicking Start to update the pipeline?

○ All datasets will be updated once and the pipeline will shut down. The compute resources will be terminated

○ All datasets will be updated at set intervals until the pipeline is shut down. The compute resources will be deployed for the update and terminated when the pipeline is stopped

○ All datasets will be updated at set intervals until the pipeline is shut down. The compute resources will persist after the pipeline is stopped to allow for additional development and testing

○ All datasets will be updated once and the pipeline will shut down. The compute resources will persist to allow for additional development and testing

○ All datasets will be updated continuously and the pipeline will not shut down. The compute resources will persist with the pipeline

Explanation
The answer is All datasets will be updated once and the pipeline will shut down. The compute resources will persist to allow for additional testing.

DLT pipeline supports two modes Development and Production, you can switch between the two based on the stage of your development and deployment lifecycle.

Development and production modes
When you run your pipeline in development mode, the Delta Live Tables system:
Reuses a cluster to avoid the overhead of restarts.
Disables pipeline retries so you can immediately detect and fix errors.
In production mode, the Delta Live Tables system:
Restarts the cluster for specific recoverable errors, including memory leaks and stale credentials.
Retries execution in the event of specific errors, for example, a failure to start a cluster.
Use the

buttons in the Pipelines UI to switch between development and production modes. By default, pipelines run in development mode.
Switching between development and production modes only controls cluster and pipeline execution behavior. Storage locations must be configured as part of pipeline settings and are not affected when switching between modes.

Please review additional DLT concepts using below link

https://docs.databricks.com/data-engineering/delta-live-tables/delta-live-tables-concepts.html#delta-live-tables-concepts

Question 34

The DELT LIVE TABLE Pipeline is configured to run in Production mode using the continuous Pipeline Mode. what is the expected outcome after clicking Start to update the pipeline?

○ All datasets will be updated once and the pipeline will shut down. The compute resources will be terminated

○ All datasets will be updated at set intervals until the pipeline is shut down. The compute resources will be deployed for the update and terminated when the pipeline is stopped

○ All datasets will be updated at set intervals until the pipeline is shut down. The compute resources will persist after the pipeline is stopped to allow for additional testing

○ All datasets will be updated once and the pipeline will shut down. The compute resources will persist to allow for additional testing

○ All datasets will be updated continuously and the pipeline will not shut down. The compute resources will persist with the pipeline

Explanation
The answer is, All datasets will be updated at set intervals until the pipeline is shut down. The compute resources will be deployed for the update and terminated when the pipeline is stopped.

DLT pipeline supports two modes Development and Production, you can switch between the two based on the stage of your development and deployment lifecycle.

Development and production modes
Development:
When you run your pipeline in development mode, the Delta Live Tables system:
Reuses a cluster to avoid the overhead of restarts.
Disables pipeline retries so you can immediately detect and fix errors.
Production:
In production mode, the Delta Live Tables system:
Restarts the cluster for specific recoverable errors, including memory leaks and stale credentials.
Retries execution in the event of specific errors, for example, a failure to start a cluster.
Use the

buttons in the Pipelines UI to switch between development and production modes. By default, pipelines run in development mode. Switching between development and production modes only controls cluster and pipeline execution behavior. Storage locations must be configured as part of pipeline settings and are not affected when switching between modes.

Please review additional DLT concepts using the below link

https://docs.databricks.com/data-engineering/delta-live-tables/delta-live-tables-concepts.html#delta-live-tables-concepts

Question 35

you are working to set up two notebooks to run on a schedule second notebook is dependent on first notebook, both notebooks need different types of compute to run in an optimal fashion? What is the best way to setup these notebooks as jobs?

○ Use DELTA LIVE PIPELINES instead of notebook tasks

○ A Job can only use single cluster, setup job for each notebook and use job dependency to link both jobs together

○ Each task can use different cluster, add these two notebooks as two tasks in a single job with linear dependency and modify the cluster as needed for each of the tasks

○ Use a single job to setup both notebooks as individual tasks, but use the cluster API to setup the second cluster before the start of second task

○ Use a very large cluster to run both the tasks in a single job

Explanation
Tasks in Jobs support different clusters for each task in the same job.

Question 36
You are tasked to setup a set notebook as a job for six departments and each department can run the task parallelly, the notebook takes an input parameter dept number to process the data by department how do you go about to setup this in job?

○
Use a single notebook as task in the job and use dbutils.notebook.run to run each notebook with parameter in a different cell

○
A task in the job cannot take an input parameter, create six notebooks with hardcoded dept number and setup six tasks with linear dependency in the job

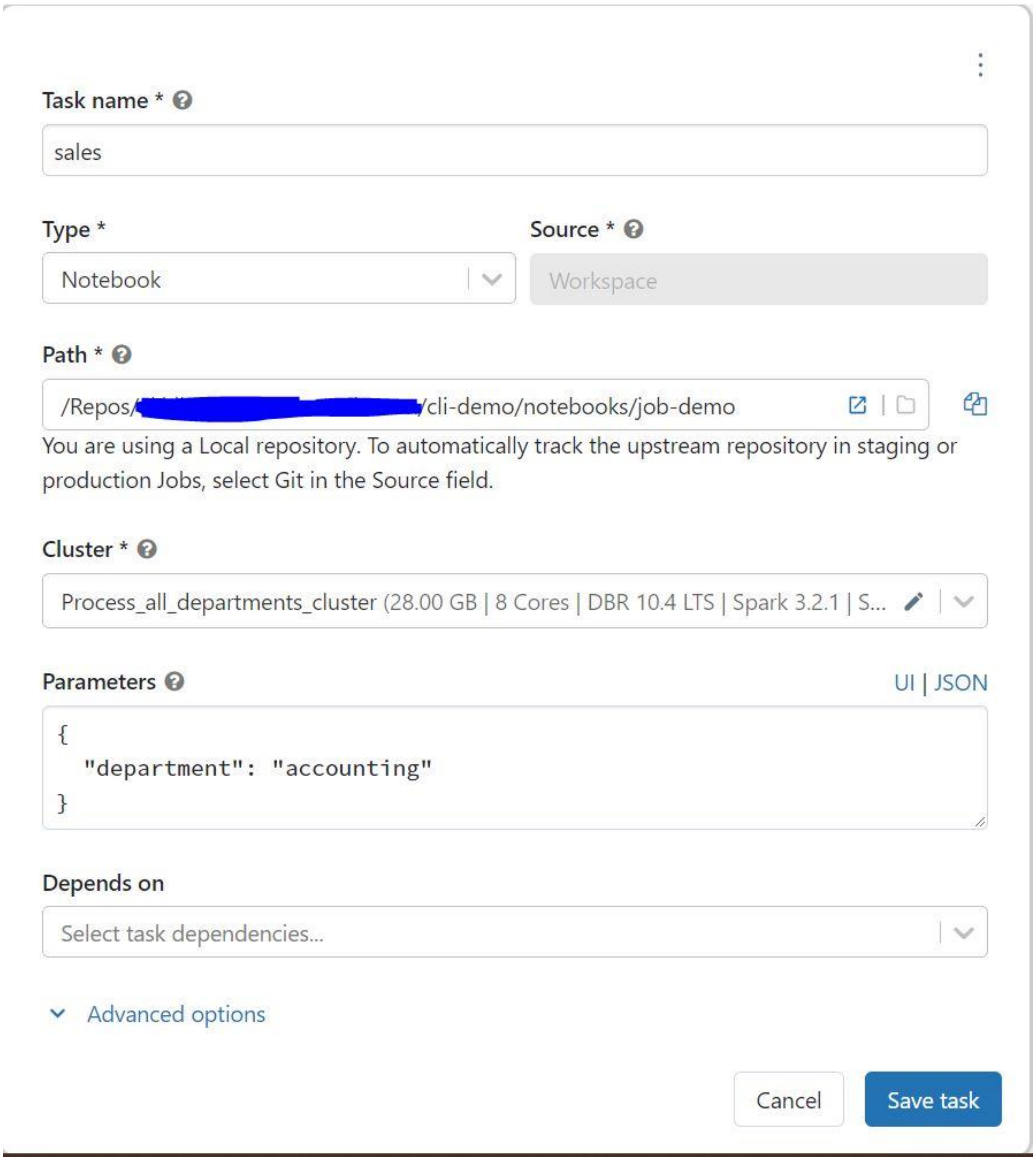
○
A task accepts key-value pair parameters, creates six tasks pass department number as parameter foreach task with no dependency in the job as they can all run in parallel.

○
A parameter can only be passed at the job level, create six jobs pass department number to each job with linear job dependency

○
A parameter can only be passed at the job level, create six jobs pass department number to each job with no job dependency

Explanation
Here is how you setup

Create a single job and six tasks with the same notebook and assign a different parameter for each task ,



Task name *

sales

Type *

Notebook

Source *

Workspace

Path *

/Repos/[redacted]/cli-demo/notebooks/job-demo

You are using a Local repository. To automatically track the upstream repository in staging or production Jobs, select Git in the Source field.

Cluster *

Process_all_departments_cluster (28.00 GB | 8 Cores | DBR 10.4 LTS | Spark 3.2.1 | S...

Parameters

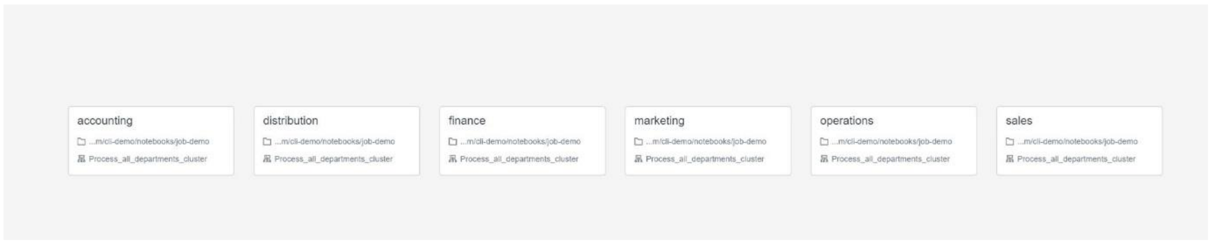
UI | JSON

```
{
  "department": "accounting"
}
```

Depends on

Select task dependencies...

All tasks are added in a single job and can run parallel either using single shared cluster or with individual clusters.


accounting
...m/cli-demo/notebooks/job-demo
Process_all_departments_cluster
distribution
...m/cli-demo/notebooks/job-demo
Process_all_departments_cluster
finance
...m/cli-demo/notebooks/job-demo
Process_all_departments_cluster
marketing
...m/cli-demo/notebooks/job-demo
Process_all_departments_cluster
operations
...m/cli-demo/notebooks/job-demo
Process_all_departments_cluster
sales
...m/cli-demo/notebooks/job-demo
Process_all_departments_cluster

Question 37

You are asked to setup two tasks, first task runs a notebook to download the from a remote database, second task is a DLT pipeline that can process this data, how do you plan to configure this in Jobs UI

○

Single job cannot have a notebook task and DLT Pipeline task, use two different jobs with linear dependency.

○

Jobs UI does not support DTL pipeline, setup the first task using jobs UI and setup the DLT to run in continuous mode.

○

Jobs UI does not support DTL pipeline, setup the first task using jobs UI and setup the DLT to run in trigger mode.

○

Single job can be used to setup both notebook and DLT pipeline, use two different tasks with linear dependency.

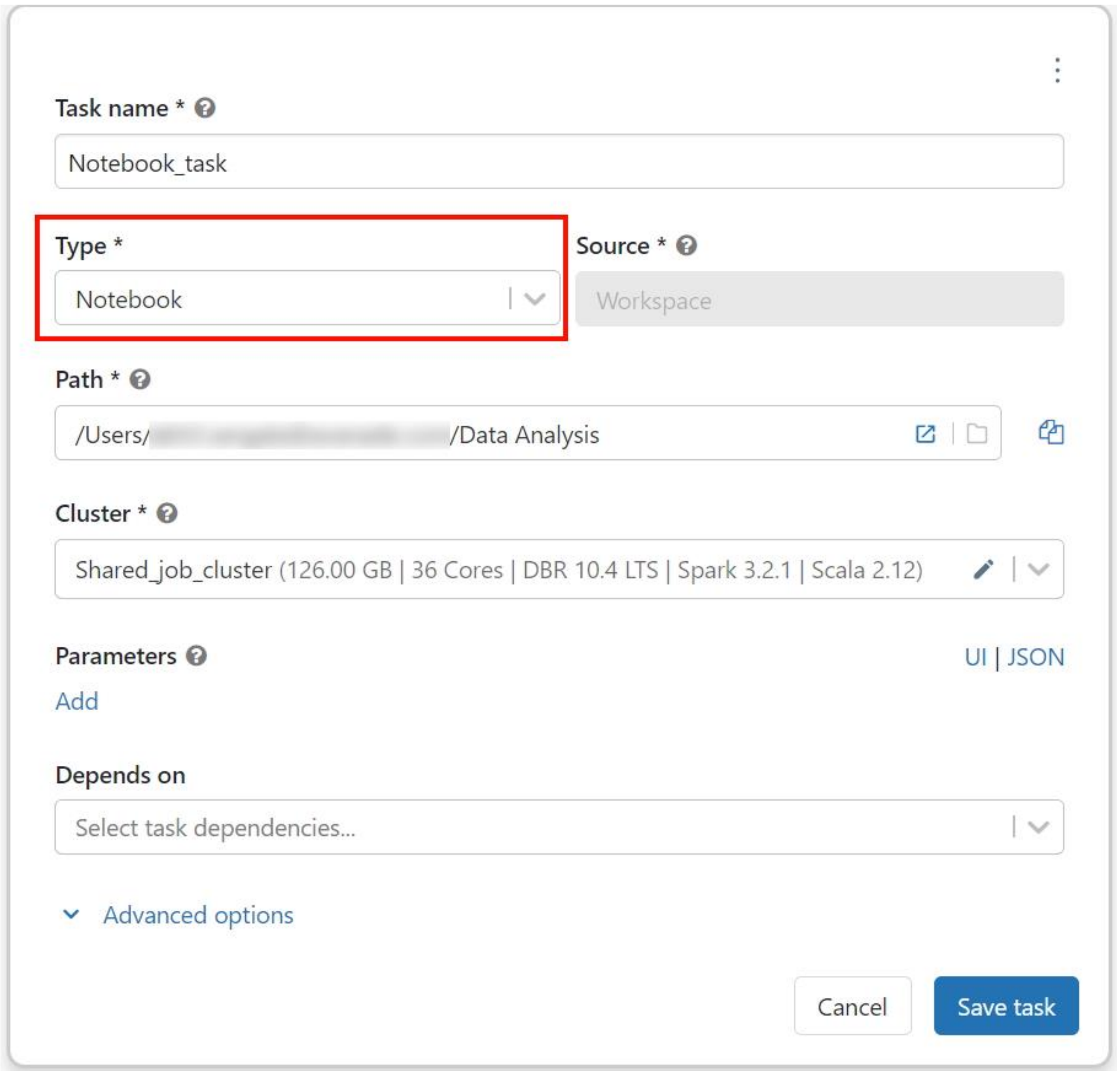
○

Add first step in the DLT pipeline and run the DLT pipeline as triggered mode in JOBS UI

Explanation
The answer is Single job can be used to set up both notebook and DLT pipeline, use two different tasks with linear dependency,

Here is the JOB UI
Create a notebook task
Create DLT task
add notebook task as dependency
Final view

Create the notebook task



Task name *
Notebook_task

Type *
Notebook

Source *
Workspace

Path *
/Users/[redacted]/Data Analysis

Cluster *
Shared_job_cluster (126.00 GB | 36 Cores | DBR 10.4 LTS | Spark 3.2.1 | Scala 2.12)

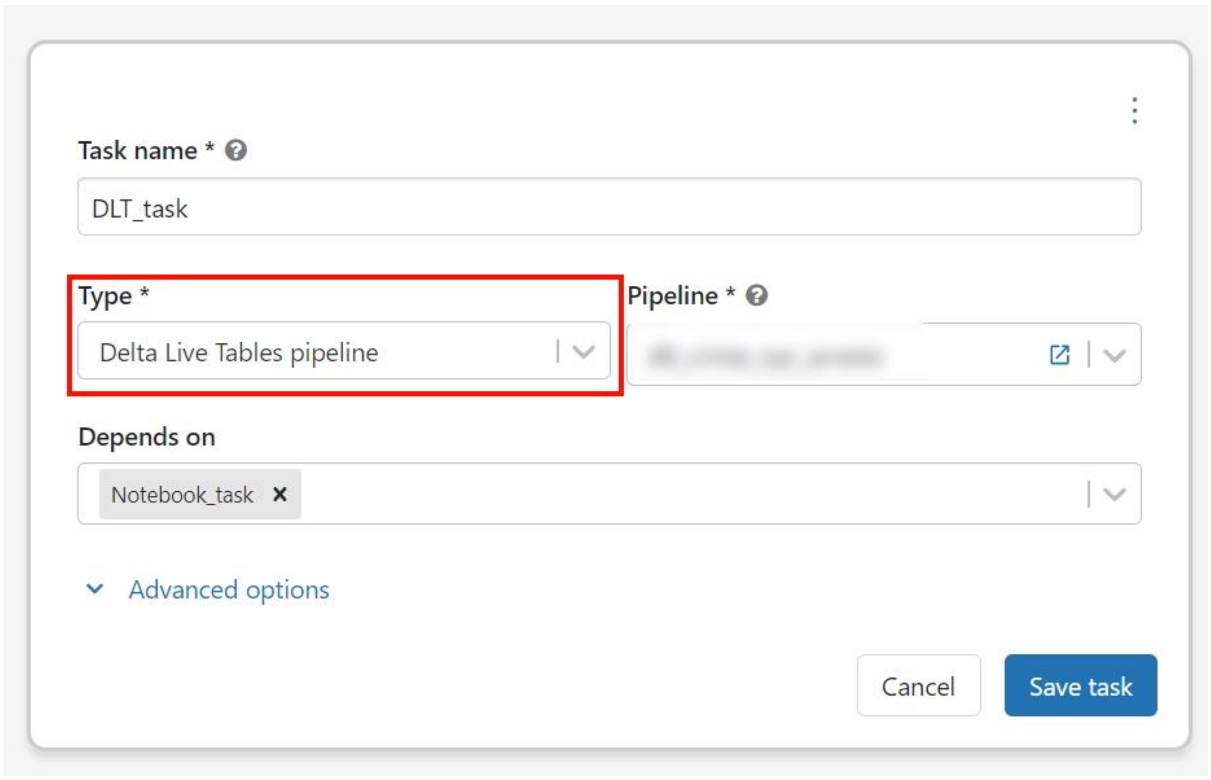
Parameters
UI | JSON
Add

Depends on
Select task dependencies...

Advanced options

Cancel
Save task

DLT task


Task name *

DLT_task

Type *

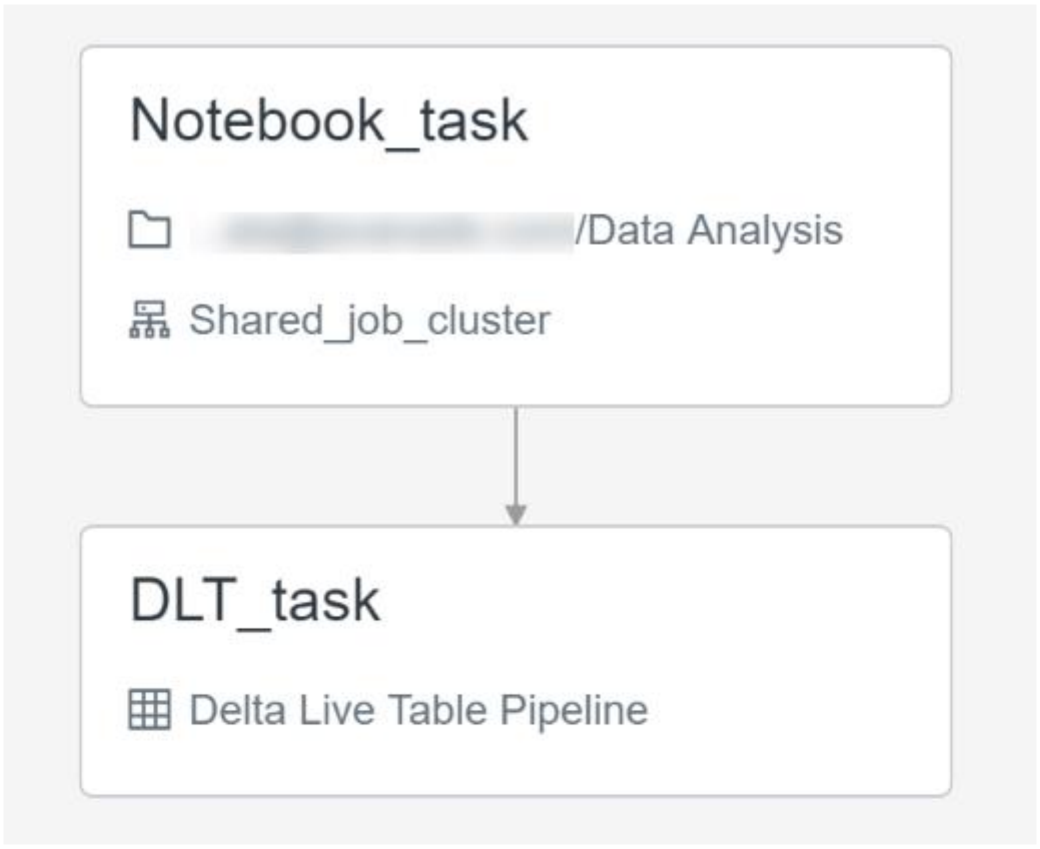
Delta Live Tables pipeline

Pipeline *

Depends on

Notebook_task

Final view


Notebook_task

/Data Analysis

Shared_job_cluster

DLT_task

Delta Live Table Pipeline

Question 38
You are asked to setup the alert to notify every time once the data is loaded into a reporting table, team also asked to include the number of records in the alert email notification.

○ Use notebook and python code to run every minute, using python variables to capture send the information in an email

○ Setup an alert but use the default template to notify the message in email’s subject

○ Setup an alert but use the custom template to notify the message in email’s subject

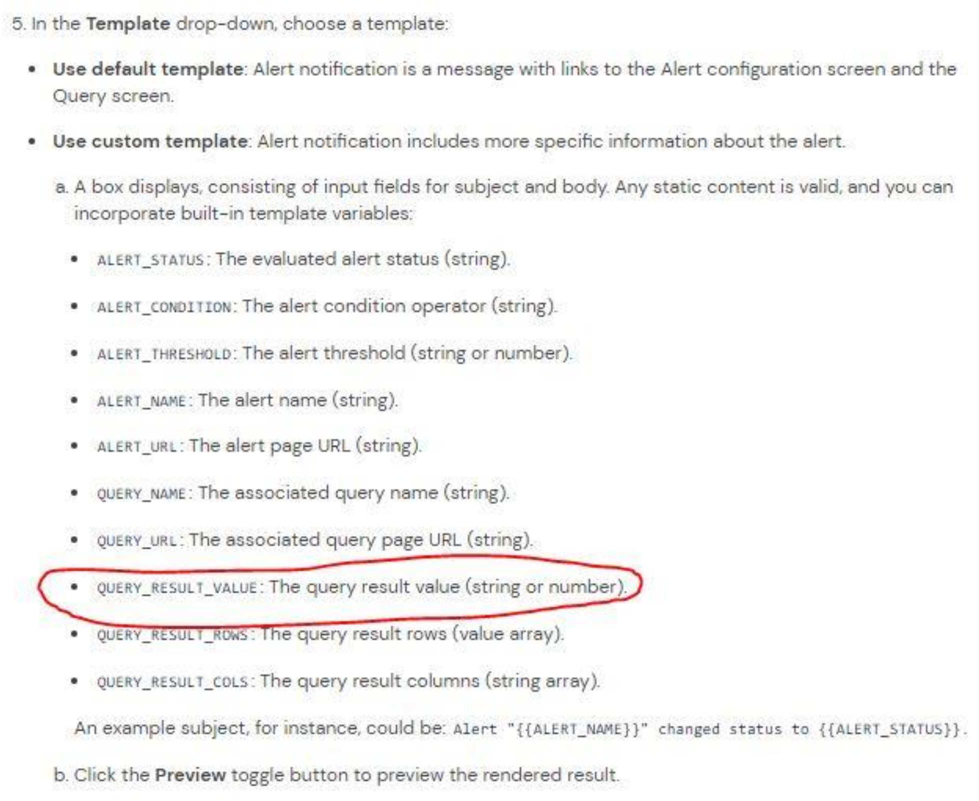
○ Use the webhook destination instead so alert message can be customized

○ Use custom email hook to customize the message

Explanation

Alerts support custom template supports using variables to customize the default message.  setup the query to count the number rows based on your criteria and use the variable QUERY_RESULT_VALUE

Alerts | Databricks on AWS



5. In the **Template** drop-down, choose a template:

- **Use default template**: Alert notification is a message with links to the Alert configuration screen and the Query screen.
- **Use custom template**: Alert notification includes more specific information about the alert.

  a. A box displays, consisting of input fields for subject and body. Any static content is valid, and you can incorporate built-in template variables:

  - `ALERT_STATUS`: The evaluated alert status (string).
  - `ALERT_CONDITION`: The alert condition operator (string).
  - `ALERT_THRESHOLD`: The alert threshold (string or number).
  - `ALERT_NAME`: The alert name (string).
  - `ALERT_URL`: The alert page URL (string).
  - `QUERY_NAME`: The associated query name (string).
  - `QUERY_URL`: The associated query page URL (string).
  - `QUERY_RESULT_VALUE`: The query result value (string or number).
  - `QUERY_RESULT_ROWS`: The query result rows (value array).
  - `QUERY_RESULT_COLS`: The query result columns (string array).

  An example subject, for instance, could be: `Alert "{{ALERT_NAME}}" changed status to {{ALERT_STATUS}}`.

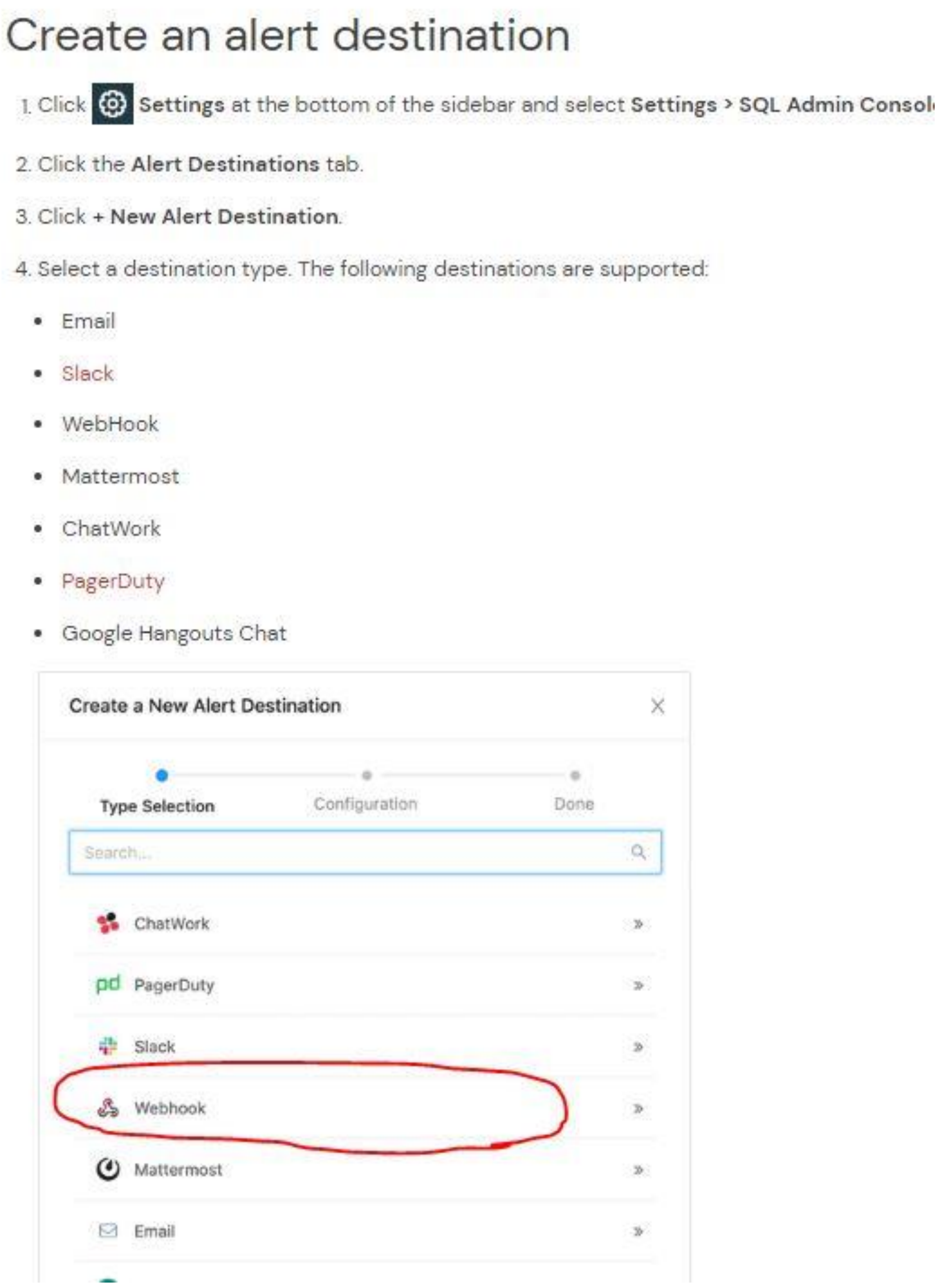
  b. Click the **Preview** toggle button to preview the rendered result.

Question 39
Operations team is using a centralized data quality monitoring system, a user can publish data quality metrics through a webhook, you were asked to develop a process to send messages using webhook if there is atleast one duplicate record, which of following approaches can be taken to integrate with current data quality monitoring system

- Use notebook and Jobs to use python to publish DQ metrics
- Setup an alert to send an email, use python to parse email, and publish a webhook message
- Setup an alert with custom template
- Setup an alert with custom Webhook destination
- Setup an alert with dynamic template

Explanation
Alerts supports multiple destinations, email is the default destinations.

Alert destinations | Databricks on AWS



## Create an alert destination

1. Click **Settings** at the bottom of the sidebar and select **Settings > SQL Admin Consol**
2. Click the **Alert Destinations** tab.
3. Click **+ New Alert Destination**.
4. Select a destination type. The following destinations are supported:

- Email
- Slack
- WebHook
- Mattermost
- ChatWork
- PagerDuty
- Google Hangouts Chat

Create a New Alert Destination

Type Selection
Configuration
Done

Search...

ChatWork »
PagerDuty »
Slack »
Webhook »
Mattermost »
Email »

Question 40
You are currently working with application team to setup a SQL Endpoint point, once the team started consuming the SQL Endpoint you noticed that during peak hours as the number of users increases you are seeing degradation in the query performance, which of the following steps can be taken to resolve the issue?

○ They can turn on the Serverless feature for the SQL endpoint.

○ They can increase the maximum bound of the SQL endpoint's scaling range.

○ They can increase the cluster size of the SQL endpoint.

○ They can turn on the Auto Stop feature for the SQL endpoint.

○ They can turn on the Serverless feature for the SQL endpoint and change the Spot Instance Policy from "Cost optimized" to "Reliability Optimized."

Explanation
The answer is, They can increase the maximum bound of the SQL endpoint's scaling range.

SQL endpoint scales horizontally(scale-out) and vertical (scale-up), you have to understand when to use what.
Scale-out -> to add more clusters for a SQL endpoint, change max number of clusters scaling range.
If you are trying to improve the throughput, being able to run as many queries as possible then having an additional cluster(s) will improve the performance.
Scale-up-> Increase the size of the SQL endpoint, change cluster size from x-small to small, to medium, X Large....
If you are trying to improve the performance of a single query having additional memory, additional nodes and cpu in the cluster will improve the performance.

Question 41

Data engineering team is using a SQL query to review data quality and monitor the ETL job everyday, which of the following approaches can be used to setup a schedule and automate this process?

○ They can schedule the query to run every 1 day from the Jobs UI

○ They can schedule the query to refresh every 1 day from the query's page in Databricks SQL.

○ They can schedule the query to run every 12 hours from the Jobs UI.

○ They can schedule the query to refresh every 1 day from the SQL endpoint's page in Databricks SQL.

○ They can schedule the query to refresh every 12 hours from the SQL endpoint's page in Databricks SQL

Explanation
Individual queries can be refreshed on a schedule basis,

To set the schedule:
Click the query info tab.

Click the link to the right of Refresh Schedule to open a picker with schedule intervals.

Set the schedule.
The picker scrolls and allows you to choose:
An interval: 1-30 minutes, 1-12 hours, 1 or 30 days, 1 or 2 weeks
A time. The time selector displays in the picker only when the interval is greater than 1 day and the day selection is greater than 1 week. When you schedule a specific time, Databricks SQL takes input in your computer's timezone and converts it to UTC. If you want a query to run at a certain time in UTC, you must adjust the picker by your local offset. For example, if you want a query to execute at `00:00` UTC each day, but your current timezone is PDT (UTC-7), you should select `17:00` in the picker:

Click OK.
Your query will run automatically.
If you experience a scheduled query not executing according to its schedule, you should manually trigger the query to make sure it doesn't fail. However, you should be aware of the following:
If you schedule an interval—for example, "every 15 minutes"—the interval is calculated from the *last successful execution*. If you manually execute a query, the scheduled query will not be executed until the interval has passed.
If you schedule a time, Databricks SQL waits for the results to be "outdated". For example, if you have a query set to refresh every Thursday and you manually execute it on Wednesday, by Thursday the results will still be considered "valid", so the query wouldn't be scheduled for a new execution. Thus, for example, when setting a weekly schedule, check the last query execution time and expect the scheduled query to be executed on the selected day after that execution is a week old. Make sure not to manually execute the query during this time.
If a query execution fails, Databricks SQL retries with a back-off algorithm. The more failures the further away the next retry will be (and it might be beyond the refresh interval).

Refer documentation for additional info,

https://docs.microsoft.com/en-us/azure/databricks/sql/user/queries/schedule-query

Question 42

In order to use Unity catalog features, which of the following steps needs to be taken on managed/external tables in the Databricks workspace?

○ Enable unity catalog feature in workspace settings

○ Migrate/upgrade objects in workspace managed/external tables/view to unity catalog

○ Upgrade to DBR version 15.0

○ Copy data from workspace to unity catalog

○ Upgrade workspace to Unity catalog

Explanation
Upgrade tables and views to Unity Catalog - Azure Databricks | Microsoft Docs

Managed table: Upgrade a managed to Unity Catalog
External table: Upgrade an external table to Unity Catalog

Question 43

What is the top-level object in unity catalog?

○ Catalog

○ Table

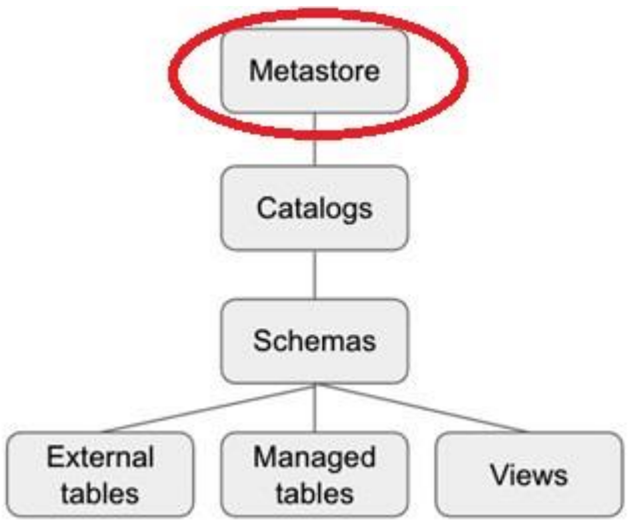
○ Workspace

○ Database

○ Metastore

Explanation

[Key concepts - Azure Databricks | Microsoft Docs]



Question 44

One of the team members Steve who has the ability to create views, created a new view called regional_sales_vw on the existing table called sales which is owned by John, and the second team member Kevin who works with regional sales managers wanted to query the data in regional_sales_vw, so Steve granted the permission to Kevin using command `GRANT VIEW, USAGE ON regional_sales_vw to kevin@company.com` but Kevin is still unable to access the view?

○ Kevin needs select access on the table sales

○ Kevin needs owner access on the view regional_sales_vw

○ Steve is not the owner of the sales table

○ Kevin is not the owner of the sales table

○ Table access control is not enabled on the table and view

Explanation
Ownership determines whether or not you can grant privileges on derived objects to other users, since Steve is not the owner of the underlying sales table, he can not grant access to the table or data in the table indirectly.

Only owner(user or group) can grant access to a object

https://docs.microsoft.com/en-us/azure/databricks/security/access-control/table-acls/object-privileges#a-user-has-select-privileges-on-a-view-of-table-t-but-when-that-user-tries-to-select-from-that-view-they-get-the-error-user-does-not-have-privilege-select-on-table

Data object privileges - Azure Databricks | Microsoft Doc

Question 45

Kevin is the owner of the schema sales, Steve wanted to create new table in sales schema called regional_sales so Kevin grants the create table permissions to Steve. Steve creates the new table called regional_sales in sales schema, who is the owner of the table regional_sales

○ Kevin is the owner of sales schema, all the tables in the schema will be owned by Kevin

○ Steve is the owner of the table

○ By default ownership is assigned DBO

○ By default ownership is assigned to DEFAULT_OWNER

○ Kevin and Smith both are owners of table

Explanation

A user who creates the object becomes its owner, does not matter who is the owner of the parent object.