DUMPSBOSS.

SnowPro Core Certification 2026 Exam

Snowflake COF-C03

Version Demo

Total Demo Questions: 89

Total Premium Questions: 893

Buy Premium PDF

https://dumpsboss.co

support@dumpsboss.co

support@dumpsboss.co
dumpsboss.co

Topic Break Down

Topic	No. of Questions
Topic 1, Snowflake Architecture	146
Topic 2, Account Access and Security	198
Topic 3, Performance Concepts	157
Topic 4, Data Loading and Unloading	174
Topic 5, Data Transformations	87
Topic 6, Data Protection and Data Sharing	131
Total	893

QUESTION NO: 1

Which COPY INTO < location > command options impact the number of files generated? Select TWO.

A. SINGLE

B. RECORD_DELIMITER

C. MAX_FILE_SIZE

D. INCLUDE_QUERY_ID

E. DETAILED_OUTPUT

ANSWER: A C

Explanation:

`SINGLE` and `MAX_FILE_SIZE` are the COPY INTO <location> options that affect the number of files created when Snowflake unloads query results to a stage or external location. Setting `SINGLE = TRUE` instructs Snowflake to unload the result set into one output file rather than its normal parallel set of output files. This is useful when a downstream consumer requires a single object, although using a single file can reduce unloading parallelism for large result sets.

`MAX_FILE_SIZE` specifies the target maximum size for each unloaded data file. Snowflake splits unloaded data into files based on this setting and execution characteristics; consequently, a lower maximum file size generally produces more files, while a higher setting generally permits fewer, larger files. The setting is expressed in bytes and has a documented default of 16 MB for unloaded files. Together, these parameters directly govern whether output is consolidated and how output is partitioned by file size. Snowflake documents both parameters in the optional COPY INTO <location> parameters: COPY INTO <location>. Additional practical guidance on controlling unloaded file sizing is available in Data unloading considerations.

QUESTION NO: 2

Which database objects can be shared using Secure Data Sharing? (Select TWO).

A. Users

B. Roles

C. Stages

D. Tables

E. Dynamic tables

ANSWER: D E

Explanation:

Tables can be added to a Snowflake share, allowing consumer accounts to access the provider's current table data without copying or moving the data. The provider retains ownership and control of the shared table, including its source data and access through the share. This is a core Secure Data Sharing use case: a provider grants a consumer account access to selected database objects, and the consumer creates a database from the share to query the shared data. Snowflake's data-sharing documentation identifies tables as supported objects for shares.

Dynamic tables can also be shared. A dynamic table is a Snowflake-managed table-like object whose contents are automatically refreshed according to its declared query and target lag. When it is included in a share, consumers can query the resulting dynamically maintained data. The sharing capability supports distributing curated, refreshed data products without requiring a separate data-export process. Snowflake documents this explicitly in the dynamic-table limitations and sharing guidance; consumers have read-only access to the shared object and applicable restrictions remain in effect for downstream use. See Introduction to Secure Data Sharing and Dynamic tables limitations.

QUESTION NO: 3

Which query types will have significant performance improvement when run using the search optimization service? (Select TWO)

A. Range searches

B. Equality searches

C. Substring searches

D. Queries with IN predicates

E. Queries with aggregation

ANSWER: B D

Explanation:

Equality searches and queries with IN predicates can benefit significantly from the search optimization service when the predicates are selective and reference columns that have been added to the search optimization configuration. Snowflake builds and maintains specialized search access paths for configured data, allowing it to identify the micro-partitions containing matching values without scanning a large portion of the table. This is particularly valuable for highly selective point-lookups, such as filtering for a customer identifier, order identifier, or another value with relatively few matching rows.

An `IN` predicate is supported as an equality-style lookup because it searches for rows matching one of a supplied set of discrete values. For example, a filter such as `customer_id IN (101, 205, 389)` can use search optimization to locate the relevant values efficiently. The benefit depends on selectivity: querying for a small set of values is generally a stronger use case than retrieving a substantial fraction of the table. Snowflake documents equality predicates, including `IN`, among the supported query patterns for search optimization. See the Search Optimization Service overview and supported query types documentation.

QUESTION NO: 4

Which of the following are benefits of micro-partitioning? (Select TWO)

A. Micro-partitions cannot overlap in their range of values

B. Micro-partitions are immutable objects that support the use of Time Travel.

C. Micro-partitions can reduce the amount of I/O from object storage to virtual warehouses

D. Rows are automatically stored in sorted order within micro-partitions

E. Micro-partitions can be defined on a schema-by-schema basis

ANSWER: B C

Explanation:

Micro-partitions are immutable objects that support the use of Time Travel because Snowflake does not update data in place within an existing micro-partition. When data is changed, Snowflake writes new micro-partitions and retains the prior versions for the applicable data-retention period. This architecture provides the historical versions needed for Time Travel queries, such as querying a table at an earlier point in time or recovering data that was changed or deleted within the retention window.

Micro-partitions can reduce the amount of I/O from object storage to virtual warehouses through micro-partition pruning. Snowflake automatically records metadata for each micro-partition, including information such as the range of values for columns. During query optimization, Snowflake uses this metadata to identify partitions that cannot contain rows satisfying a query predicate. The warehouse can then avoid reading those irrelevant partitions, reducing the volume of data scanned from storage and often improving query performance. These benefits are automatic characteristics of Snowflake's storage

architecture and require no user-defined partition management. See Snowflake's documentation on micro-partitions and data clustering and Time Travel.

QUESTION NO: 5

Which account__usage views are used to evaluate the details of dynamic data masking? (Select TWO)

A. ROLES

B. POLICY_REFERENCES

C. QUERY_HISTORY

D. RESOURCE_MONITORS

E. ACCESS_HISTORY

ANSWER: B E

Explanation:

`POLICY_REFERENCES` is used to examine where a masking policy is associated within the account. It records policy-to-object relationships, including the database, schema, table, view, or column context in which a policy is referenced. This lets administrators inventory masking-policy coverage and validate that sensitive data columns have the intended policy assignments. The view is especially useful for reviewing direct assignments as well as relationships involving tags and policy applications.

`ACCESS_HISTORY` provides the audit-oriented perspective needed to evaluate use of protected data. Its records identify objects and columns accessed by queries and include policy information associated with access events. Administrators can use this history to investigate who accessed data protected by a masking policy, determine the relevant queries and objects, and support governance or compliance reviews of dynamic masking activity. Together, `POLICY_REFERENCES` establishes where masking policies are applied, while `ACCESS_HISTORY` helps assess actual access to the governed data. Snowflake documents policy-reference metadata in the POLICY_REFERENCES Account Usage view and query-level access auditing in the ACCESS_HISTORY Account Usage view.

QUESTION NO: 6

A user wants to add additional privileges to the system-defined roles for their virtual warehouse. How does Snowflake recommend they accomplish this?

A. Grant the additional privileges to a custom role.

B. Grant the additional privileges to the ACCOUNTADMIN role.

C. Grant the additional privileges to the SYSADMIN role.

D. Grant the additional privileges to the ORGADMIN role.

ANSWER: A

Explanation:

Grant the additional privileges to a custom role. Snowflake's access-control model is role-based, and its recommended practice is to create custom roles that represent job functions or access requirements, then grant the required object privileges to those roles. A custom role can receive warehouse privileges such as USAGE, OPERATE, MONITOR, MODIFY, or APPLYBUDGET, as appropriate for the user's responsibilities. The custom role can then be granted to users directly or incorporated into the organization's role hierarchy by granting it to another role. This approach separates operational access from Snowflake's built-in administrative roles and enables least-privilege access: users receive only the warehouse capabilities needed for their work. It also makes permissions easier to audit, maintain, and adjust as responsibilities change. Snowflake provides system-defined roles for specific administrative purposes, but organizations should use custom roles to

model their own functional access patterns rather than modifying their security design around highly privileged built-in roles. For details, see Snowflake's Access Control Overview and Access Control Configuration.

QUESTION NO: 7

Which function, when combined with an INSERT statement, will insert semi-structured data into a VARIANT column?

A. TO_JSON

B. TO_VARCHAR

C. PARSE_JSON

D. OBJECT_INSERT

ANSWER: C

Explanation:

`PARSE_JSON` is correct because it takes a string containing valid JSON and parses it into a Snowflake `VARIANT` value. `VARIANT` is Snowflake's native data type for storing semi-structured data, including JSON objects and arrays. Therefore, when JSON is supplied as text in an `INSERT` statement, `PARSE_JSON` performs the required conversion before the value is stored.

For example, after defining `CREATE TABLE json_table (src VARIANT);`, an insert such as `INSERT INTO json_table SELECT PARSE_JSON('{"customer_id":101,"region":"WEST"}');` stores the JSON document as structured VARIANT data. Snowflake can then query elements using semi-structured-data path notation, such as `src:customer_id`, while retaining the complete document in the column.

This behavior is particularly useful for SQL workloads that receive JSON payloads from applications, APIs, staged files, or data pipelines. The input must be valid JSON text, and the function returns a VARIANT representation suitable for direct insertion. Snowflake documents `PARSE_JSON` as interpreting a string as JSON and returning a `VARIANT` value. See the PARSE_JSON function reference and Snowflake's guide to semi-structured data.

QUESTION NO: 8

Which commands are supported by the query acceleration service? (Select TWO).

A. INSERT

B. MERGE

C. SELECT

D. UPDATE

E. DELETE

ANSWER: A C

Explanation:

The supported choices are **INSERT** and **SELECT**. Snowflake Query Acceleration Service (QAS) can provide additional compute resources to accelerate eligible portions of a SELECT workload, particularly when large volumes of data must be scanned and filtered but relatively few rows are returned. QAS also supports INSERT operations when the insert statement uses a query as its source, such as `INSERT INTO target_table SELECT ...`. In that pattern, the service can accelerate the eligible SELECT processing that supplies the rows being inserted. QAS eligibility is determined per query and depends on the query plan; enabling the service does not guarantee that every eligible statement will be accelerated. The virtual warehouse must have QAS enabled through its `QUERY_ACCELERATION_MAX_SCALE_FACTOR` setting, and Snowflake applies the service only when it estimates a performance benefit. Snowflake documents supported query forms

and the requirements for using the service in its Query Acceleration Service documentation. For warehouse-level configuration and monitoring details, see QUERY_ACCELERATION_MAX_SCALE_FACTOR.

QUESTION NO: 9

What is the purpose of configuring the **scale factor** parameter when using the Query Acceleration Service?

A. It increases the default selectivity of any filter parameters when the query is accelerated.

B. It is used when there are not enough partitions in a table to accelerate the query.

C. It controls costs by setting an upper bound on the amount of compute resources a warehouse can use for query acceleration.

D. It is a concurrency control tool used to avoid unnecessary scaling up when the query is accelerated and the warehouse is running in Maximized mode.

ANSWER: C

Explanation:

It controls costs by setting an upper bound on the amount of compute resources a warehouse can use for query acceleration. The Query Acceleration Service can allocate additional, shared compute resources to eligible portions of a query, allowing work to be processed in parallel with the warehouse and potentially reducing elapsed query time. The warehouse-level `QUERY_ACCELERATION_MAX_SCALE_FACTOR` parameter establishes the maximum scale factor that Snowflake may use for this accelerated compute. In practical terms, it lets an administrator balance performance gains against the credit consumption that can result from acceleration. A larger value permits more resources to be applied when Snowflake identifies an eligible query and determines that acceleration can help; a smaller value constrains that additional resource usage. The configured value is an upper limit rather than a guarantee that the corresponding amount of compute will always be allocated. Snowflake still determines eligibility and the useful amount of acceleration for each query. This parameter is therefore a cost-governance and performance-tuning control for Query Acceleration Service at the warehouse level. See Snowflake's Query Acceleration Service documentation and the QUERY_ACCELERATION_MAX_SCALE_FACTOR parameter reference.

QUESTION NO: 10

Which of the following practices are recommended when creating a user in Snowflake? (Choose two.)

A. Configure the user to be initially disabled.

B. Force an immediate password change.

C. Set a default role for the user.

D. Set the number of minutes to unlock to 15 minutes.

E. Set the user ' s access to expire within a specified timeframe.

ANSWER: B C

Explanation:

Force an immediate password change. and **Set a default role for the user.** are recommended user-creation practices in Snowflake. When a password is being configured for a user, setting `MUST_CHANGE_PASSWORD` to `TRUE` ensures that the user replaces the administrator-provided initial password at their next login. This avoids relying on a password that was generated, communicated, or temporarily known by an administrator, and establishes a password chosen by the user under the account's configured password policy.

Assigning a default role establishes the role that is active when the user starts a Snowflake session, unless the user or connection configuration deliberately selects another granted role. This supports predictable access behavior and helps

ensure users begin with the intended least-privilege role rather than an unintended role. The default role must be granted to the user before it can be designated as that user's default. Snowflake documents both properties as supported `CREATE USER` or `ALTER USER` settings and describes default roles as part of user access management. See the CREATE USER reference and Snowflake's user management guidance.

QUESTION NO: 11

How does Snowflake Fail-safe protect data in a permanent table?

A. Fail-safe makes data available up to 1 day, recoverable by user operations.

B. Fail-safe makes data available for 7 days, recoverable by user operations.

C. Fail-safe makes data available for 7 days, recoverable only by Snowflake Support.

D. Fail-safe makes data available up to 1 day, recoverable only by Snowflake Support.

ANSWER: C

Explanation:

Fail-safe makes data available for 7 days, recoverable only by Snowflake Support. For permanent tables, Fail-safe begins after the applicable Time Travel retention period ends. It provides an additional, non-configurable seven-day window during which Snowflake may be able to recover historical data following a catastrophic event or other exceptional circumstance. Fail-safe is intended as a final data-protection mechanism, rather than a feature for routine querying, restoration, or self-service recovery. Consequently, users cannot run SQL commands to access or restore data that has entered Fail-safe. A customer must contact Snowflake Support to request recovery, and recovery is not guaranteed because it depends on the circumstances and technical feasibility. Snowflake documentation also notes that data storage charges can continue to apply while historical data is retained through Time Travel and Fail-safe. This distinction is important operationally: use Time Travel for user-managed recovery within its retention period, and treat Fail-safe as an emergency safeguard administered by Snowflake. See the Snowflake Fail-safe documentation and Snowflake Time Travel documentation.

QUESTION NO: 12

Which objects together comprise a namespace in Snowflake? (Select TWO).

A. Account

B. Database

C. Schema

D. Table

E. Virtual warehouse

ANSWER: B C

Explanation:

In Snowflake, the namespace for database objects is formed by the combination of a database and a schema. A database is a top-level logical container that holds schemas, while a schema is a logical container within a database that holds objects such as tables, views, stages, file formats, sequences, and functions. Together, these two names provide the namespace portion of an object's fully qualified name. For example, in `MY_DATABASE.MY_SCHEMA.MY_TABLE`, `MY_DATABASE` and `MY_SCHEMA` identify the namespace in which `MY_TABLE` resides. This organization enables Snowflake to distinguish objects with the same object name when they are located in different schemas or databases. Snowflake documentation describes the database and schema hierarchy and recommends fully qualifying object names when needed to avoid ambiguity, particularly in shared or multi-team environments. See the Snowflake database, schema, and table concepts and the documentation for object name resolution.

QUESTION NO: 13

How does a Snowflake user extract the URL of a directory table on an external stage for further transformation?

A. Use the SHOW STAGES command.

B. Use the DESCRIBE STAGE command.

C. Use the GET_ABSOLUTE_PATH function.

D. Use the GET_STAGE_LOCATION function.

ANSWER: D

Explanation:

Use the `GET_STAGE_LOCATION` function. This Snowflake SQL function returns the location associated with a named stage. For an external stage, that result is the external cloud-storage URL configured for the stage, such as an Amazon S3, Google Cloud Storage, or Azure Storage location. The function accepts a stage reference, for example `SELECT GET_STAGE_LOCATION('@my_external_stage');`, allowing the returned location to be incorporated into subsequent SQL processing or transformation logic.

This is distinct from querying directory-table metadata itself: a directory table exposes file-level metadata, whereas `GET_STAGE_LOCATION` retrieves the configured location of the stage that hosts those files. Snowflake documents that the function returns the stage URL for external stages and an absolute path for internal stages, making it the direct programmatic method for extracting the requested external-stage URL. The stage must be referenced with the required `@` prefix, and the executing role must have the appropriate privileges to access the stage. See Snowflake's GET_STAGE_LOCATION function reference and its directory tables documentation for the relationship between stages and directory metadata.

QUESTION NO: 14

Which Snowflake tasks are managed in the cloud services layer? Select TWO.

A. Performance tuning

B. Query acceleration

C. Metadata management

D. Query processing

E. Query parsing

ANSWER: C E

Explanation:

Snowflake's cloud services layer provides the coordination and control services that sit above storage and virtual warehouse compute. It manages the system-level functions needed to receive a request, validate it, understand the referenced database objects, and prepare the request for execution. Therefore, **Metadata management** is a cloud services responsibility. Snowflake maintains metadata describing database objects and the organization of stored data; this information supports efficient planning and management without requiring users to administer infrastructure directly.

Query parsing is also managed in the cloud services layer. Before a virtual warehouse performs execution work, Snowflake parses and optimizes SQL statements through cloud services. This step interprets the submitted SQL and creates an efficient execution plan, after which compute resources in a virtual warehouse carry out the query processing. This separation is a core part of Snowflake's architecture: cloud services coordinates activities and supplies services such as authentication, access control, metadata management, and query parsing and optimization, while the compute layer performs query execution.

See Snowflake's Snowflake architecture overview and its Cloud Services Layer documentation for the listed responsibilities.

QUESTION NO: 15

How can unstructured data be processed in a Snowflake stage? (Select TWO).

A. Use a SQL procedure.

B. Use external functions.

C. Use a C# procedure.

D. Implement a Golang (Go) User-Defined Function (UDF).

E. Implement a Python User-Defined Function (UDF).

ANSWER: E

Explanation:

Implement a Python User-Defined Function (UDF) is a supported Snowflake-native method for processing files held in a stage. A Python UDF can accept a staged file through the `FILE` SQL type and use the SnowflakeFile API to open and read its contents. This permits code running within Snowflake's Python execution environment to parse or transform unstructured file formats without first copying the file out of the stage. For example, the function can open a file passed as a scoped URL, read binary or text content, and return derived values to SQL. Snowflake documents this pattern specifically for processing unstructured data, including the use of `SnowflakeFile.open` and scoped URLs to provide controlled access to staged files. The execution model also keeps the processing governed by Snowflake roles, stage privileges, and function permissions, making it appropriate for operational pipelines where staged documents, images, or other non-tabular files need programmatic handling. See Snowflake's Python UDF SnowflakeFile example and its guidance for processing unstructured data.

QUESTION NO: 16

What can a Snowflake user do with the information included in the details section of a Query Profile?

A. Determine the total duration of the query.

B. Determine the role of the user who ran the query.

C. Determine the source system that the queried table is from.

D. Determine if the query was on structured or semi-structured data.

ANSWER: A

Explanation:

Determine the total duration of the query. Snowflake's Query Profile provides visual and statistical information about query execution, including the overall elapsed execution time. This duration is displayed as part of the profile's query overview and is fundamental for assessing how long the statement took to run. A user can use this value as a starting point for performance investigation, then inspect the profile's execution graph and operator-level details to understand where time was spent, such as processing, data scanning, partition pruning, local or remote disk activity, or data exchange between operations. Query Profile information is intended to help users diagnose query behavior and identify opportunities to improve query performance, warehouse sizing, clustering, or SQL design. The total duration is therefore a directly available and meaningful metric that can be determined from the Query Profile information. Snowflake documents Query Profile as a tool for examining execution details and monitoring query performance, including the elapsed time of a query. See the Query Profile documentation and Snowflake's Monitoring query activity documentation.

QUESTION NO: 17

While running a query on a virtual warehouse in auto-scale mode, additional clusters are stated immediately if which setting is configured?

A. MAX_CLUSTER_COUNT is increased and new_max_clusters is greater than running_clusters

B. MAX_CLUSTER_COUNT is decreased and new_max_clusters is less than running_clusters

C. MIN_CLUSTER_COUNT is increased and new_min_clusters is greater than running_clusters

D. MIN_CLUSTER_COUNT is decreased and new_min_clusters is less than running_clusters

ANSWER: C

Explanation:

`MIN_CLUSTER_COUNT is increased and new_min_clusters is greater than running_clusters` is correct. For a multi-cluster virtual warehouse in auto-scale mode, `MIN_CLUSTER_COUNT` establishes the minimum number of clusters Snowflake must keep available while the warehouse is running. When this value is raised above the number of clusters that are currently running, Snowflake immediately starts enough additional clusters to meet the newly configured minimum. This is a direct configuration-driven action: Snowflake does not wait for queued queries, sustained load, or its normal scaling evaluation before bringing the warehouse into compliance with the higher minimum.

This behavior is useful when an administrator anticipates increased concurrency and wants capacity online before queries begin queuing. For example, if one cluster is running and the minimum is changed to three, Snowflake starts two more clusters immediately. The warehouse must be a multi-cluster warehouse, meaning its cluster limits permit more than one cluster. Snowflake documents this immediate-start behavior in its guidance for changing multi-cluster warehouse properties and describes how minimum and maximum cluster counts control the available cluster range. See ALTER WAREHOUSE and Multi-cluster warehouses.

QUESTION NO: 18

What features does Snowflake Time Travel enable?

A. Querying data-related objects that were created within the past 365 days

B. Restoring data-related objects that have been deleted within the past 90 days

C. Conducting point-in-time analysis for BI reporting

D. Analyzing data usage/manipulation over all periods of time

ANSWER: B C

Explanation:

Snowflake Time Travel enables recovery and historical analysis of data within the applicable data-retention period. “Restoring data-related objects that have been deleted within the past 90 days” reflects its support for recovering eligible dropped databases, schemas, and tables using `UNDROP`, provided the object is still within its configured Time Travel retention window. The available retention period depends on the object type and Snowflake edition: permanent objects can have a retention period of up to 90 days in eligible editions, while other object types have more limited retention capabilities. Snowflake retains the required historical data and metadata during that window so the object can be recovered without recreating it manually.

“Conducting point-in-time analysis for BI reporting” is also enabled by Time Travel. A query can use `AT` or `BEFORE` clauses to access a table’s prior state at a specified timestamp, offset, or statement ID. This supports reproducible reports, auditing, investigation of unexpected changes, and comparison of current results with historical versions of the same data. Both recovery and historical querying are bounded by the configured retention period rather than being available indefinitely. See Snowflake’s Time Travel documentation and its guidance on AT | BEFORE clauses.

QUESTION NO: 19

A user is loading JSON documents composed of a huge array containing multiple records into Snowflake. The user enables the strip__outer_array file format option

What does the STRIP_OUTER_ARRAY file format do?

A. It removes the last element of the outer array.

B. It removes the outer array structure and loads the records into separate table rows,

C. It removes the trailing spaces in the last element of the outer array and loads the records into separate table columns

D. It removes the NULL elements from the JSON object eliminating invalid data and enables the ability to load the records

ANSWER: B

Explanation:

`STRIP_OUTER_ARRAY = TRUE` removes the enclosing brackets of a top-level JSON array during loading, so Snowflake can treat each array element as an individual record. Consequently, an input document such as `[{"id":1},{"id":2}]` is processed as two separate records rather than as one record containing an array. When loading semi-structured JSON into a table with a `VARIANT` column, each element is loaded into its own row, preserving the element's JSON structure in that row. This setting is useful for JSON files exported as a single outer array of objects, because it avoids requiring the source data to be rewritten into one JSON document per line before the load. Snowflake documents this behavior as stripping the outer array and loading the records inside the array as separate rows. The option is a JSON file-format setting and can be supplied when creating or altering a file format, or in an applicable load operation. See Snowflake's JSON file-format options and data loading considerations documentation.

QUESTION NO: 20

How can a row access policy be applied to a table or a view? (Choose two.)

A. Within the policy DDL

B. Within the create table or create view DDL

C. By future APPLY for all objects in a schema

D. Within a control table

E. Using the command ALTER < object > ADD ROW ACCESS POLICY < policy > ;

ANSWER: B E

Explanation:

A row access policy can be attached when the table or view is created by including a `WITH ROW ACCESS POLICY` clause in the `CREATE TABLE` or `CREATE VIEW` statement. The clause specifies the policy and the table or view column(s) that are passed as arguments to the policy. This lets access filtering be established as part of provisioning the object, so the object is protected immediately upon creation.

A policy that already exists can also be attached to an existing table or view using `ALTER <object> ADD ROW ACCESS POLICY <policy>`, along with the required column mapping. Snowflake evaluates the policy automatically whenever a user queries the protected object, returning only rows permitted for that user's active role and policy logic. The policy definition itself defines the filtering expression and signature, but it does not independently attach the policy to a table or view; attachment occurs through the object's create or alter DDL. Snowflake documents both creation-time attachment and post-creation attachment in its row access policy and SQL command references. See Row access policies and ALTER TABLE.

QUESTION NO: 21

What are best practice recommendations for using the ACCOUNTADMIN system-defined role in Snowflake? (Choose two.)

A. Ensure all ACCOUNTADMIN roles use Multi-factor Authentication (MFA).

B. All users granted ACCOUNTADMIN role must be owned by the ACCOUNTADMIN role.

C. The ACCOUNTADMIN role must be granted to only one user.

D. Assign the ACCOUNTADMIN role to at least two users, but as few as possible.

E. All users granted ACCOUNTADMIN role must also be granted SECURITYADMIN role.

ANSWER: A D

Explanation:

Snowflake recommends requiring multi-factor authentication (MFA) for every user who is assigned the ACCOUNTADMIN role. ACCOUNTADMIN is the highest-privileged role in an account: it can manage account-level objects and settings, administer users and roles, and access or delegate powerful administrative capabilities. Requiring MFA provides an important additional authentication control for these highly sensitive identities, reducing the impact of a compromised password.

Snowflake also recommends assigning ACCOUNTADMIN to at least two users, while keeping membership as limited as practical. Having two or more designated administrators prevents an operational lockout or recovery delay if a sole administrator is unavailable or cannot access their account. Restricting the role to the minimum number of trusted users follows the principle of least privilege and limits exposure to account-wide administrative authority. These practices should be supported by using lower-privilege system roles for routine administration whenever possible, reserving ACCOUNTADMIN for tasks that require account-level control.

See Snowflake's guidance on access control considerations and its documentation for multi-factor authentication.

QUESTION NO: 22

Which stage type can be altered and dropped?

A. Database stage

B. External stage

C. Table stage

D. User stage

ANSWER: B

Explanation:

External stage is correct. In Snowflake, an external stage is a named stage object that stores connection and location metadata for files held in external cloud storage, such as Amazon S3, Google Cloud Storage, or Azure Blob Storage. Because it is a named schema object, its configuration can be maintained with the `ALTER STAGE` command, including supported changes to properties such as file-format or directory-table settings. When the stage is no longer required, it can be removed with `DROP STAGE`, subject to the privileges needed to manage the stage. Dropping the Snowflake stage removes its metadata definition from Snowflake; it does not delete the data files at the external storage location. This makes external stages appropriate when a reusable, centrally managed reference to cloud-storage data is needed. Snowflake documents external stages as a form of named stage and provides dedicated syntax for both altering and dropping named stage objects. See the ALTER STAGE reference and the DROP STAGE reference.

QUESTION NO: 23

Which statements describe Snowflake materialized views? (Choose two.)

A. They are automatically maintained when underlying base-table data changes.

B. They can be used by the optimizer to improve eligible query performance.

C. They must be manually refreshed after every base-table change.

D. They initially share all table storage through zero-copy cloning.

E. They do not incur storage or maintenance costs.

ANSWER: A B

Explanation:

Materialized views store precomputed query results and are automatically maintained by Snowflake when data in the underlying base tables changes. Snowflake can also use a materialized view to improve performance through query rewrite when the optimizer determines that the materialized view can satisfy all or part of a query.

Materialized views are not zero-copy clones and require storage. Their maintenance can also consume credits, so they should be evaluated against the expected performance benefit for repeated query patterns.

QUESTION NO: 24

At what levels can a resource monitor be configured? (Select TWO).

A. Account

B. Database

C. Organization

D. Schema

E. Virtual warehouse

ANSWER: A E

Explanation:

Resource monitors can be configured at the **Account** level and for a **Virtual warehouse**. An account-level resource monitor tracks the credit consumption of all supported warehouses in the account that are assigned to that monitor. This allows an administrator to establish an overall credit-consumption budget and receive notifications or apply configured actions as usage reaches specified thresholds. A resource monitor can also be assigned to an individual virtual warehouse, allowing its credit usage to be monitored and controlled separately. This is useful when distinct teams, workloads, or cost centers use different warehouses and require their own consumption limits. Resource monitor thresholds are defined as percentages of the monitor's credit quota and can trigger notifications, warehouse suspension, or warehouse suspension followed by prevention of automatic resumption. Snowflake notes that resource monitors apply to virtual warehouse credit usage; they do not monitor all possible account charges. The monitor's scope is therefore determined by whether it is assigned at the account level or directly to one or more virtual warehouses. See Snowflake's resource monitor documentation and the CREATE RESOURCE MONITOR reference.

QUESTION NO: 25

Which command should be used to unload all the rows from a table into one or more files in a named stage?

A. COPY INTO

B. GET

C. INSERT INTO

D. PUT

ANSWER: A

Explanation:

`COPY INTO` is the Snowflake SQL command used to unload data from a table or query result into one or more data files in a stage, including a named internal stage or an external cloud-storage stage. For this use case, the destination is specified first, followed by a query or table source, for example: `COPY INTO @my_stage/export/ FROM my_table;`. Snowflake executes the unload operation in parallel when appropriate and writes the output as one or more files. The command also supports unload-specific controls such as the target file format, compression, header inclusion, single-file output, maximum file size, and overwrite behavior. This makes it the standard mechanism for exporting all rows from a table to staged files for sharing, downstream processing, or archival. The caller must have the necessary privileges to read the table and write to the destination stage. Snowflake documents this operation as the `COPY INTO <location>` form of the COPY command; the location is the stage destination and the source is a table or SQL query. See the Snowflake COPY INTO <location> reference and the guide for unloading data from Snowflake.

QUESTION NO: 26

What impacts the credit consumption of maintaining a materialized view? (Choose two.)

A. Whether or not it is also a secure view

B. How often the underlying base table is queried

C. How often the base table changes

D. Whether the materialized view has a cluster key defined

E. How often the materialized view is queried

ANSWER: C D

Explanation:

“How often the base table changes” affects materialized-view maintenance because Snowflake must keep the materialized-view data synchronized as data in its source table changes. Inserts, updates, deletes, and other source-table modifications can require refresh work, and a greater volume or frequency of change generally requires more maintenance processing. This maintenance is performed automatically by Snowflake and consumes credits as serverless compute.

“Whether the materialized view has a cluster key defined” also affects credit consumption. A materialized view may be defined with a clustering key, and maintaining the desired clustering of its stored results can require automatic reclustering work in addition to keeping the view current with base-table changes. Consequently, the physical organization specified by a clustering key is a relevant component of the ongoing cost profile. Snowflake recommends considering both refresh and automatic-clustering costs when evaluating materialized views, particularly when the underlying data changes frequently. See Snowflake’s materialized views documentation and its automatic clustering documentation for details on maintenance behavior and associated serverless credit usage.

QUESTION NO: 27

A Snowflake user wants to optimize performance for a query that queries only a small number of rows in a table. The rows require significant processing. The data in the table does not change frequently.

What should the user do?

A. Add a clustering key to the table.

B. Add the search optimization service to the table.

C. Create a materialized view based on the query.

D. Enable the query acceleration service for the virtual warehouse.

ANSWER: C

Explanation:

Create a materialized view based on the query. is correct because a materialized view persistently stores the results of a supported query and is automatically maintained by Snowflake as base-table data changes. This avoids repeatedly performing the significant processing required by the original query. It is particularly valuable when the same or similar expensive query pattern is run repeatedly and the underlying data changes relatively infrequently, because the incremental maintenance cost is generally lower than repeatedly computing the full result at query time.

Snowflake's optimizer can transparently rewrite eligible queries to use a materialized view, even when the query does not explicitly name that view. As a result, users can retain their query pattern while benefiting from precomputed data and reduced execution work. Materialized views are designed for query acceleration where precomputation of expensive operations, such as filters, projections, and aggregations supported by materialized-view rules, provides a meaningful runtime advantage. The low rate of data change in this scenario also helps keep ongoing materialized-view maintenance practical relative to the processing saved during reads.

For materialized-view design, supported query limitations, automatic maintenance behavior, and cost considerations, see Snowflake's Materialized Views documentation and its materialized view usage notes.

QUESTION NO: 28

Which data types can be used in a Snowflake table that holds semi-structured data? (Select TWO).

A. ARRAY

B. BINARY

C. TEXT

D. VARIANT

E. VARCHAR

ANSWER: A D

Explanation:

ARRAY and VARIANT are Snowflake semi-structured data types and are appropriate for columns in a table that stores semi-structured content. VARIANT is Snowflake's flexible universal type for representing hierarchical or otherwise non-relational values. A VARIANT value can hold native Snowflake values as well as data ingested from supported semi-structured formats, including JSON, Avro, ORC, Parquet, and XML. It preserves the document-like structure so that nested attributes can be queried with Snowflake's semi-structured data path notation and functions.

ARRAY represents an ordered collection of values. It is useful when a semi-structured record contains list-like data, such as a collection of products, events, tags, or measurements. Snowflake supports both structured and semi-structured arrays; the semi-structured form can contain values of differing types, including nested objects and arrays. ARRAY values can be stored directly in table columns and later expanded or analyzed with functions such as `FLATTEN`. These types allow the table to retain meaningful nested structures while still making their contents queryable in SQL. See Snowflake's semi-structured data types documentation and its guide to querying semi-structured data.

QUESTION NO: 29

What AUTHENTICATION_METHODS value should be selected for the Single Sign-On, or SSO, login option to appear when creating a Snowflake authentication policy?

A. PASSWORD

B. KEYPAIR

C. SAML

D. OAUTH

ANSWER: C

Explanation:

SAML is the appropriate `AUTHENTICATION_METHODS` value because Snowflake's browser-based federated single sign-on uses Security Assertion Markup Language (SAML) 2.0. A Snowflake authentication policy controls which methods a user may use to authenticate. Including `SAML` in that policy permits users to sign in through the SAML identity provider that has been configured for the Snowflake account, which is what enables the SSO sign-in experience.

For example, an administrator can define a policy with `AUTHENTICATION_METHODS = ('SAML')`, then apply that policy to the relevant users or account scope. The identity provider authenticates the user and sends a SAML assertion to Snowflake; Snowflake validates the assertion and creates the authenticated session. This method is commonly used with enterprise identity providers such as Microsoft Entra ID, Okta, Ping Identity, or similar services.

Snowflake documents `SAML` as a supported authentication-method value in authentication policies and describes SAML SSO as the federated authentication integration used for user sign-in. See the CREATE AUTHENTICATION POLICY reference and Snowflake's federated authentication and SSO documentation.

QUESTION NO: 30

Which privilege is required for a role to be able to resume a suspended warehouse if auto-resume is not enabled?

A. USAGE

B. OPERATE

C. MONITOR

D. MODIFY

ANSWER: B

Explanation:

OPERATE is the required warehouse privilege to manually resume a suspended virtual warehouse. Snowflake separates warehouse permissions according to the action being performed. The OPERATE privilege authorizes operational lifecycle actions, including starting (resuming), suspending, and aborting queries running on a warehouse. Therefore, when a warehouse is suspended and auto-resume is disabled, a user acting through a role with OPERATE on that warehouse can explicitly resume it using a command such as `ALTER WAREHOUSE <name> RESUME` or through Snowsight. In practice, the role also needs appropriate access to the warehouse for the intended workload, commonly USAGE, but USAGE alone does not authorize the manual resume operation. Snowflake documents OPERATE as the privilege that enables warehouse start and stop actions, making it the specific privilege asked for here. See the Snowflake warehouse privilege table in the virtual warehouse privileges documentation and the authorization requirements for ALTER WAREHOUSE.

QUESTION NO: 31

What is the !define < variable > = < value > command used for in SnowSQL?

A. Assigns a name to a variable

B. Assigns a value to a variable

C. Sets the context for a variable

D. Establishes the metrics of a variable

ANSWER: B

Explanation:

`!define <variable> = <value>` assigns a value to a SnowSQL substitution variable. SnowSQL processes this client-side command before sending SQL statements to Snowflake, storing the supplied value under the specified variable name. The variable can then be referenced later in the same interactive session or script using SnowSQL variable-substitution syntax, enabling scripts to reuse values such as object names, dates, or environment-specific settings without repeating literal text.

For example, `!define table_name = my_table` defines `table_name` with the value `my_table`. A later statement can use `&table_name`, and SnowSQL substitutes the stored value when it processes the script. This feature is specifically part of SnowSQL's variable support and is useful for parameterizing SQL scripts executed through the SnowSQL client. Snowflake documents `!define` as the command used to define a variable and assign its value; the value remains available for substitution until it is changed or the SnowSQL session ends. See Snowflake's Using Variables in SnowSQL documentation and the Using SnowSQL guide.

QUESTION NO: 32

Which SQL commands, when committed, will consume a stream and advance the stream offset? (Choose two.)

A. UPDATE TABLE FROM STREAM

B. SELECT FROM STREAM

C. INSERT INTO TABLE SELECT FROM STREAM

D. ALTER TABLE AS SELECT FROM STREAM

E. BEGIN COMMIT

ANSWER: A C

Explanation:

`UPDATE TABLE FROM STREAM` and `INSERT INTO TABLE SELECT FROM STREAM` are stream-consuming operations when they are executed as committed data-manipulation transactions. A Snowflake stream records a position, or offset, in the source object's change history rather than storing an independent copy of changed rows. When a committed DML statement uses the stream as its source, Snowflake advances that stream's offset to the transaction's current timestamp. This marks the changes visible before that timestamp as consumed for subsequent queries of that stream.

An update can use stream rows to apply changes to an existing target table, while an insert can use stream rows to load changed records into another table. Both patterns are common building blocks for incremental pipelines, including change-data-capture processing and merge-style loading designs. The offset movement is transactionally consistent: if the DML transaction does not commit, the stream offset does not advance. This behavior permits reliable processing because a failed load can be retried without losing the stream records that were intended for processing. Snowflake documents that a stream advances when it is used in a DML transaction that commits, and further describes stream offsets and consumption semantics in its stream documentation. See Introduction to streams and Data manipulation language reference.

QUESTION NO: 33

When should a multi-cluster warehouse be used in auto-scaling mode?

A. When it is unknown how much compute power is needed

B. If the select statement contains a large number of temporary tables or Common Table Expressions (CTEs)

C. If the runtime of the executed query is very slow

D. When a large number of concurrent queries are run on the same warehouse

ANSWER: D

Explanation:

A multi-cluster warehouse in auto-scaling mode should be used when a large number of concurrent queries are run on the same warehouse. This feature is designed to manage workload concurrency: Snowflake can automatically start additional clusters as query demand rises and shut them down again as demand falls, within the configured minimum and maximum cluster limits. By adding clusters, Snowflake provides more compute resources to process independent queued queries in parallel, which helps maintain predictable response times during peaks in user or workload activity. Auto-scaling is especially appropriate for workloads with variable concurrency, such as dashboards, reporting applications, and many simultaneous users submitting queries. The cluster count is managed automatically according to demand, so administrators do not need to predict the exact number of clusters required at every point in time. Multi-cluster warehouses address queuing caused by concurrent query volume; warehouse sizing and query optimization remain separate considerations for improving the performance of an individual resource-intensive query. See Snowflake's Multi-cluster warehouses documentation and its guidance on warehouse concurrency and scaling.

QUESTION NO: 34

By default, which tasks can a user with the ORGADMIN role perform? Select TWO.

A. Create a share.

B. Enable replication.

C. Create an account.

D. Create a resource monitor.

E. Create a reader account.

ANSWER: B C

Explanation:

`ORGADMIN` is Snowflake's organization-level administrator role. It is intended for tasks that span the Snowflake organization rather than administration confined to one individual account. Consequently, **Enable replication.** is a supported default responsibility: an organization administrator can enable replication for accounts in the organization, allowing the organization to use cross-region and cross-cloud replication and failover capabilities where applicable. This organization-level control is needed because replication can involve account relationships and infrastructure in separate Snowflake regions.

Create an account. is also a default ORGADMIN capability. Organizations use this role to provision and manage Snowflake accounts, including accounts in different regions or cloud platforms where supported. The user must have the ORGADMIN organization role active to perform organization administration operations such as account creation. Account creation establishes a new, separately administered Snowflake environment under the same organization, which is distinct from creating objects inside an existing account.

Snowflake documents ORGADMIN as the role for administering organization-level resources and identifies account creation and enabling replication among its capabilities. See Snowflake's Organization roles documentation and the Create an account documentation.

QUESTION NO: 35

What is the minimum Snowflake edition required for row level security?

A. Standard

B. Enterprise

C. Business Critical

D. Virtual Private Snowflake

ANSWER: B

Explanation:

The minimum required edition is **Enterprise**. Snowflake implements row-level security through row access policies, which evaluate a row against the current user's role or other contextual values at query time and return only rows the policy permits that user to see. This allows organizations to enforce centralized, consistent data-entitlement rules directly on tables and views, rather than requiring separate copies of data or relying solely on object-level privileges. For example, a policy can allow regional managers to access records for their assigned region while administrators retain broader visibility. Snowflake lists row access policies among the Enterprise Edition capabilities. Higher editions also include this feature because they build on Enterprise functionality, but the question asks for the minimum edition rather than an edition that merely supports it. The policy is created with SQL and then attached to a table or view; Snowflake applies it automatically whenever protected data is queried. For implementation details and edition availability, see Snowflake's row access policy documentation and the Snowflake editions documentation.

QUESTION NO: 36

What are characteristic of Snowsight worksheet? (Select TWO.)

A. Worksheets can be grouped under folder, and a folder of folders.

B. Each worksheet is a unique Snowflake session.

C. Users are limited to running only one on a worksheet.

D. The Snowflake session ends when a user switches worksheets.

E. Users can import worksheets and share them with other users.

ANSWER: A B E

Explanation:

Snowsight worksheets support organization through folders, including nested folders, so users can group related SQL, Python, and analysis work in a structured workspace. This is useful for separating work by project, team, environment, or use case while retaining a manageable worksheet inventory.

Each Snowsight worksheet also has its own Snowflake session. Consequently, worksheet-specific session context, such as the active role, warehouse, database, schema, and session variables, can be independently established for that worksheet. This isolation lets a user work in separate contexts without one worksheet's session settings becoming the active context of another worksheet.

Snowsight additionally supports importing worksheet content from SQL files and sharing worksheets with other Snowflake users. Sharing enables a recipient to access a copy of the worksheet, facilitating collaboration while allowing each user to run the work under their own permissions and compute context. Although the prompt says to select two, these three statements are documented Snowsight worksheet capabilities and characteristics.

See Snowflake's Snowsight worksheets documentation and its guidance for organizing worksheets.

QUESTION NO: 37

What file formats does Snowflake support for loading semi-structured data? (Choose three.)

A. TSV

B. JSON

C. PDF

D. Avro

E. Parquet

F. JPEG

ANSWER: B D E

Explanation:

Snowflake natively supports loading JSON, Avro, and Parquet as semi-structured data formats. When these files are loaded into a table, Snowflake can store the document or record content in a `VARIANT` column. VARIANT preserves the hierarchical nature of the source data and enables SQL queries to navigate nested objects and arrays using path notation and the `FLATTEN` function when needed. JSON is commonly used for object-and-array-based records, Avro is a schema-oriented row-based serialization format, and Parquet is a columnar format that Snowflake can read efficiently. File-format options can be configured with `CREATE FILE FORMAT` or supplied in a `COPY INTO` command, depending on the loading workflow. Snowflake documentation specifically identifies JSON, Avro, ORC, Parquet, and XML among the supported semi-structured file formats; therefore, the three listed supported formats in this question are JSON, Avro, and Parquet. See Snowflake's Introduction to semi-structured data and its CREATE FILE FORMAT reference for supported type values and loading configuration details.

QUESTION NO: 38

What happens when an external or an internal stage is dropped? (Select TWO).

A. When dropping an external stage, the files are not removed and only the stage is dropped

B. When dropping an external stage, both the stage and the files within the stage are removed

C. When dropping an internal stage, the files are deleted with the stage and the files are recoverable

D. When dropping an internal stage, the files are deleted with the stage and the files are not recoverable

E. When dropping an internal stage, only selected files are deleted with the stage and are not recoverable

ANSWER: A D

Explanation:

When dropping an external stage, the files are not removed and only the stage is dropped. An external stage is a Snowflake metadata object that stores the connection details and location information for data held in a cloud storage provider, such as Amazon S3, Google Cloud Storage, or Azure Blob Storage. Dropping that object removes Snowflake's stage definition, but it does not issue deletion operations against the objects held in the external storage location. The files therefore remain under the lifecycle, retention, and access controls configured in the cloud provider.

When dropping an internal stage, the files are deleted with the stage and the files are not recoverable. Internal stages use Snowflake-managed storage for staged files. Consequently, removal of the internal stage also removes the staged files associated with it; they cannot be retrieved by recreating a stage with the same name. This behavior makes it important to load, copy, or otherwise preserve any files needed later before dropping an internal named stage. Snowflake documents the stage removal behavior in the DROP STAGE reference; background on the storage models is available in the internal stage documentation.

QUESTION NO: 39

Which features could be used to improve the performance of queries that return a small subset of rows from a large table? (Select TWO).

A. Search optimization service

B. Automatic clustering

C. Row access policies

D. Multi-cluster virtual warehouses

E. Secure views

ANSWER: A B

Explanation:

Search optimization service is designed to accelerate highly selective queries, including point lookups and queries with selective predicates that return a small number of rows from large tables. Snowflake builds and maintains specialized search access paths so that query processing can identify the relevant micro-partitions more efficiently, reducing the amount of data that must be scanned. This is especially useful when ordinary micro-partition pruning alone does not sufficiently limit scanning for highly selective filters.

Automatic clustering can also improve this workload pattern when query predicates commonly filter on one or more clustering dimensions. It automatically maintains the table's clustering over time, improving the correlation between related values and micro-partitions. Better clustering increases micro-partition pruning, allowing Snowflake to eliminate nonmatching partitions before scanning them. As a result, selective queries can read less data and complete more quickly. The appropriate choice of clustering keys should reflect frequently used, selective filter columns and consider the ongoing compute cost of maintaining clustering. Snowflake documents search optimization for selective lookups and automatic clustering for maintaining clustering depth and pruning efficiency. See Search Optimization Service and Automatic Clustering.

QUESTION NO: 40

By default, which **system roles** can create and manage users? (Select TWO)

A. USERADMIN

B. SYSADMIN

C. SECURITYADMIN

D. PUBLIC

E. ORGADMIN

ANSWER: A C

Explanation:

USERADMIN is the dedicated system role for identity administration in a Snowflake account. It has the privileges needed to create and manage users and roles, including altering user properties and assigning roles to users. Snowflake recommends delegating user and role administration through this role rather than performing routine identity-management work with the highest-level administrative role.

SECURITYADMIN can also create and manage users because it inherits the privileges of USERADMIN in Snowflake's system-role hierarchy. In addition to those inherited user and role administration capabilities, SECURITYADMIN is responsible for managing access control: it can grant or revoke privileges and roles, including through its MANAGE GRANTS capability. This makes it suitable for security administrators who must coordinate both identities and authorization assignments.

These capabilities come from Snowflake's predefined system roles and their inheritance relationships. A role higher in the hierarchy can use the privileges inherited from roles granted to it. See Snowflake's documentation on system-defined roles and the role hierarchy and privilege inheritance.

QUESTION NO: 41

How do Snowflake data providers share data that resides in different databases?

A. External tables

B. Secure views

C. Materialized views

D. User-Defined Functions (UDFs)

ANSWER: B

Explanation:

Secure views are the appropriate mechanism because a Snowflake share is associated with one database, while a secure view can encapsulate a query that accesses objects in more than one database within the provider account. The provider creates the secure view in the database that will be added to the share, defines the view using fully qualified references to the required source tables or views, and grants the share access to the secure view. Consumers then query the shared secure view without receiving direct access to the underlying objects or needing visibility into the source databases. Secure views are specifically designed for controlled data sharing because their definition and certain underlying details are not exposed to consumers, helping the provider maintain the intended security boundary while presenting a unified result set. Snowflake documents secure views as a way to share data from multiple databases and recommends fully qualifying referenced object names when creating a secure view for sharing. See Snowflake's sharing data from multiple databases documentation and its secure views documentation.

QUESTION NO: 42

What Multi-Factor Authentication, or MFA, service is provided by Snowflake?

A. Microsoft Entra ID

B. Google Authenticator

C. Duo Security

D. Microsoft Authenticator

ANSWER: C

Explanation:

Duo Security is the MFA service integrated and provided by Snowflake for users authenticating with Snowflake-managed credentials. After a user supplies their Snowflake username and password, MFA requires an additional verification through Duo, such as approving a push notification in Duo Mobile, entering a passcode, or using another enrolled Duo-supported factor. This additional proof of identity helps protect accounts if a password is compromised, because possession of the password alone is not sufficient to complete the sign-in process.

Administrators can enroll users in Snowflake MFA and can require MFA for users through authentication policies. Snowflake also supports federation with external identity providers, where that provider can apply its own authentication and MFA controls; however, Snowflake's documented native MFA integration is Duo Security. In practical terms, users who are configured for Snowflake MFA complete the secondary verification with their Duo enrollment, while Snowflake retains the account and session controls associated with the login. This distinction is important when selecting the MFA service named in the question. Snowflake's documentation describes the enrollment and use of Duo as part of its MFA workflow. See Multi-factor authentication — Snowflake Documentation and Authentication and access control — Snowflake Documentation.

QUESTION NO: 43

What are value types that a VARIANT column can store? (Select TWO)

A. STRUCT

B. OBJECT

C. BINARY

D. ARRAY

E. CLOB

ANSWER: B D

Explanation:

`OBJECT` and `ARRAY` are value types natively represented and stored in Snowflake `VARIANT` columns. An `OBJECT` is a collection of key-value pairs, such as a JSON object. It is appropriate for representing named attributes and can contain nested objects, arrays, and scalar values. An `ARRAY` is an ordered collection of values. Its elements may be scalars or nested semi-structured values, including additional arrays and objects. These structures allow a single `VARIANT` value to preserve hierarchical data without requiring a fixed relational schema for every nested attribute.

Snowflake's semi-structured data model uses `VARIANT` to hold data such as JSON-derived structures, and it provides path syntax, `FLATTEN`, and casting functions to query and transform nested object fields and array elements. For example, a JSON document loaded into a `VARIANT` column can retain both its object properties and its embedded arrays for later analysis. See Snowflake's documentation on semi-structured data types and querying semi-structured data.

QUESTION NO: 44

Which Snowflake data governance feature supports the tracking of sensitive data for compliance, discovery, protection, and resource usage?

A. Row access policies

B. Data classification

C. Object dependencies

D. Object tagging

ANSWER: D

Explanation:

Object tagging is the Snowflake governance feature designed to attach descriptive metadata to Snowflake objects, including databases, schemas, tables, views, columns, warehouses, and other supported objects. Tags can identify data sensitivity, ownership, cost center, retention category, or compliance classification. For example, an organization can apply a tag such as `PII`, `CONFIDENTIAL`, or `GDPR` to columns containing regulated information, making that metadata available for governance processes and auditing.

Tags support discovery because administrators and governance teams can query tag references to locate objects carrying particular classifications. They support compliance and protection workflows by providing consistent metadata that can be used alongside Snowflake policy features, including masking policies and row access policies. Tags also support resource-usage analysis and cost allocation when applied to relevant account objects such as warehouses. Snowflake provides account usage and information schema views for examining tag assignments, helping teams maintain a centralized inventory of governed assets and monitor the use of sensitive data. See Snowflake's object tagging documentation and its guidance on data governance.

QUESTION NO: 45

What does a Notify & Suspend action for a resource monitor do?

A. Send an alert notification to all account users who have notifications enabled.

B. Send an alert notification to all virtual warehouse users when thresholds over 100% have been met.

C. Send a notification to all account administrators who have notifications enabled, and suspend all assigned warehouses after all statements being executed by the warehouses have completed.

D. Send a notification to all account administrators who have notifications enabled, and suspend all assigned warehouses immediately, canceling any statements being executed by the warehouses.

ANSWER: C

Explanation:

Send a notification to all account administrators who have notifications enabled, and suspend all assigned warehouses after all statements being executed by the warehouses have completed. This accurately describes the combined `NOTIFY & SUSPEND` action available for a Snowflake resource monitor. When the monitor reaches its configured credit-usage threshold, Snowflake sends a resource-monitor notification to account administrators who have enabled notifications. It also suspends every warehouse assigned to that monitor, but does so only after currently running statements on those warehouses finish. This lets in-flight work complete in an orderly manner while preventing the warehouses from consuming additional credits through subsequent execution. Resource-monitor actions are evaluated at the defined threshold percentage of the monitor's credit quota, enabling administrators to establish automated spending controls without manually monitoring warehouse consumption. Snowflake documents `SUSPEND` as the action that suspends assigned warehouses after executing statements complete, and distinguishes it from the immediate-suspension action intended for stopping work at once. For configuration details and action definitions, see the Snowflake resource monitors documentation and the CREATE RESOURCE MONITOR reference.

QUESTION NO: 46

What data-type constructs are used to create a hierarchy when organizing semi-structured data? (Select TWO).

A. A VARIANT can hold a value of any other data type.

B. An ARRAY or OBJECT holds a value of type INTEGER.

C. An ARRAY or OBJECT holds a value of type VARIANT.

D. An ARRAY or OBJECT holds a value of type VARCHAR.

E. An ARRAY or OBJECT holds a value of type VECTOR.

ANSWER: A C

Explanation:

`VARIANT` is Snowflake's flexible semi-structured data type and can store values of many Snowflake data types, including nested `ARRAY` and `OBJECT` values. This enables a single column value to represent a complete JSON-like document whose structure can vary by row. When nested content is loaded into a `VARIANT` value, Snowflake retains the hierarchical relationships between parent values and child values, allowing paths to be queried with dot notation and bracket notation.

`ARRAY` and `OBJECT` provide the actual collection constructs used within that hierarchy. An `ARRAY` is an ordered collection, while an `OBJECT` is a collection of key-value pairs. Their members are represented as `VARIANT` values, which permits nesting arrays inside objects, objects inside arrays, and further combinations at multiple levels. Together, the ability of `VARIANT` to contain other types and the ability of `ARRAY`/`OBJECT` to contain `VARIANT` values support arbitrary semi-structured hierarchies. See Snowflake's documentation on semi-structured data types and querying semi-structured data.

QUESTION NO: 47

In the Data Exchange, who can get or request data from the listings? (Select TWO).

A. Users with ACCOUNTADMIN role

B. Users with sysadmin role

C. Users with ORGADMIN role

D. Users with import share privilege

E. Users with manage grants privilege

ANSWER: A D

Explanation:

Users with ACCOUNTADMIN role can get or request data from Data Exchange listings because ACCOUNTADMIN is the top-level system role in a Snowflake account and has the authority needed to perform account-level data-sharing operations. Snowflake also permits this capability to be delegated without granting broad administrative control: a role that has been granted the IMPORT SHARE global privilege can get or request data from listings. This privilege is specifically intended for consumer-side access to shared data and listings, enabling an appropriately authorized role to add data made available through the exchange to the consumer account. After a listing is obtained or a request is approved, the consumer can create a database from the resulting share and then grant access to that database through the account's normal role-based access-control model. Therefore, ACCOUNTADMIN provides the built-in administrative path, while IMPORT SHARE provides the least-privilege path for users who need to acquire shared listing data. Snowflake documents IMPORT SHARE as the global privilege that allows a role to import a share, and its Data Exchange documentation describes consumers getting data products through listings. See Snowflake global privileges and Snowflake data sharing documentation.

QUESTION NO: 48

How often are the Account and Table master keys automatically rotated by Snowflake?

A. 30 Days

B. 60 Days

C. 90 Days

D. 365 Days.

ANSWER: A

Explanation:

30 Days is correct. Snowflake's hierarchical encryption key model uses account master keys to protect table master keys, which in turn protect the keys used to encrypt stored data. As part of Snowflake-managed encryption key lifecycle management, Snowflake automatically rotates both account master keys and table master keys every 30 days. Rotation creates new active key material while preserving the ability to access existing encrypted data through the relevant key hierarchy. This recurring rotation is performed by the Snowflake service and does not require customers to schedule, initiate, or manage the process for Snowflake-managed keys. The 30-day schedule is an important aspect of Snowflake's default encryption architecture and supports ongoing cryptographic hygiene by limiting the duration for which a particular master key remains active. Snowflake documents this behavior in its encryption-key management guidance, including the relationship between account and table master keys and the automatic rotation interval. For additional detail, see the Snowflake documentation on managing encryption keys and the overview of encryption in Snowflake.

QUESTION NO: 49

Which TABLE function helps to convert semi-structured data to a relational representation?

A. CHECK_JSON

B. TO_JSON

C. FLATTEN

D. PARSE_JSON

ANSWER: C

Explanation:

FLATTEN is correct because it is Snowflake's table function for expanding semi-structured values, including ARRAY, OBJECT, and VARIANT data, into a row-based result set that can be queried relationally. It is commonly used with a lateral join, such as `LATERAL FLATTEN(input => column_name)`, to produce one output row for each element of an array or each key-value entry in an object. The function returns useful metadata columns including `KEY`, `INDEX`, `PATH`, `VALUE`, and `THIS`. These columns let SQL queries access nested values, preserve array positions where needed, and continue flattening multiple levels of a hierarchical document. For example, a JSON array stored in a VARIANT column can be flattened so each array element is available as an individual row and can be selected, filtered, joined, or aggregated using standard SQL. This makes FLATTEN the appropriate function for transforming semi-structured data into a relational representation. See Snowflake's FLATTEN function reference and its guidance on querying semi-structured data.

QUESTION NO: 50

What are characteristics of Snowflake network policies? (Select TWO).

A. They can be set for any Snowflake Edition.

B. They can be applied to roles.

C. They restrict or enable access to specific IP addresses.

D. They are activated using ALTER DATABASE SQL commands.

E. They can only be managed using the ORGADMIN role.

ANSWER: A C

Explanation:

Snowflake network policies provide an IP-based access-control layer for Snowflake accounts. They define allowed and blocked IPv4 address ranges using IP addresses or CIDR notation, enabling an organization to limit which client network locations may initiate connections to Snowflake. When a policy is active, Snowflake evaluates the originating client IP address during authentication and permits or denies access according to the policy's configured rules. This makes network policies useful for enforcing access from approved corporate networks, VPN egress addresses, or other trusted locations.

They can be set for any Snowflake Edition because network policies are a generally available Snowflake security capability rather than an edition-specific feature. Administrators can create and manage policies at the account level and may associate a policy with an individual user when more targeted controls are needed. The policy mechanism is designed to protect access across supported Snowflake clients and interfaces by applying the IP restrictions before an authenticated session is established. Snowflake documents the policy syntax, allowed and blocked IP-list behavior, account-level activation, and user-level assignment in its network-policy documentation. See Network policies and the CREATE NETWORK POLICY reference.

QUESTION NO: 51

Which Snowflake tool would be BEST to troubleshoot network connectivity?

A. SnowCLI

B. SnowUI

C. SnowSQL

D. SnowCD

ANSWER: D

Explanation:

SnowCD is the Snowflake Connectivity Diagnostic Tool and is specifically designed to investigate client-to-Snowflake network connectivity problems. It runs diagnostic checks from the environment where a user or application is attempting to

connect, helping identify whether required network paths and endpoints can be reached. Its tests cover items such as DNS resolution, TCP connectivity, proxy configuration, and connectivity to Snowflake services that are needed for login, query execution, and data transfer. The tool produces diagnostic output that can be reviewed by administrators or supplied to Snowflake Support when investigating a connection issue. This makes it the most appropriate first-line utility when the central question is whether a network, firewall, proxy, DNS, or endpoint-access configuration is preventing Snowflake connectivity. Snowflake documents SnowCD as a downloadable diagnostic utility and provides instructions for executing it and interpreting the generated report. See the SnowCD documentation and Snowflake's client connectivity troubleshooting guidance.

QUESTION NO: 52

How can staged files be removed during data loading once the files have loaded successfully?

A. Use the DROP command

B. Use the purge copy option.

C. Use the FORCE = TRUE parameter

D. Use the LOAD UNCERTAIN FILES copy option.

ANSWER: B

Explanation:

Use the purge copy option. In a Snowflake `COPY INTO <table>` statement, setting `PURGE = TRUE` instructs Snowflake to remove staged data files after they have been successfully loaded into the target table. For example, a load can specify `COPY INTO my_table FROM @my_stage FILE_FORMAT = (TYPE = CSV) PURGE = TRUE;`. This is useful when the files are no longer needed for reload, audit, or recovery and the organization wants to avoid retaining unnecessary objects in the stage.

The purge action applies only after a successful load operation. Snowflake documents that this behavior is best effort: if an underlying storage issue prevents file deletion, the load can still succeed while some files remain in the stage. Therefore, workflows requiring strict retention, independent archival, or guaranteed cleanup should incorporate an appropriate verification and lifecycle process. The `PURGE` parameter is specifically designed to couple successful ingestion with staged-file cleanup, making it the appropriate mechanism during data loading. See the Snowflake COPY INTO <table> reference and its data staging considerations documentation.

QUESTION NO: 53

A user frequently runs a reporting query that returns the total record count and the minimum and maximum timestamp from a 100 TB fact table.

The table is loaded once per week and remains static between loads. The reporting query runs hundreds of times per day and causes significant warehouse-credit consumption because it repeatedly scans the fact table.

Which object will improve performance, minimize repeated query-compute costs, and provide the fastest response time?

A. Standard view

B. Dynamic table

C. Temporary table

D. Materialized view

ANSWER: D

Explanation:

Materialized view is correct because it persistently stores the precomputed result of a qualifying query, allowing this repeated aggregation workload to avoid scanning the 100 TB fact table on every execution. A materialized view can maintain aggregate results such as the row count and minimum and maximum timestamp, so subsequent requests can be satisfied from a very small, already-computed data set rather than performing a full-table aggregation. This substantially reduces the warehouse work associated with hundreds of identical or compatible reporting queries each day and delivers a much faster response time.

Snowflake automatically and transparently maintains a materialized view as data in its base table changes. Since this table is loaded only weekly and is static between loads, the maintenance frequency is far lower than the reporting-query frequency. This makes the pattern especially suitable: maintenance credits incurred after the weekly load can be offset by avoiding repeated large scans throughout the week. Snowflake's optimizer may also rewrite eligible queries against the base table to use the materialized view automatically, without requiring users to explicitly reference the materialized-view object. See Snowflake's materialized views documentation and materialized-view query optimization guidance.

QUESTION NO: 54

Which of the following describes how multiple Snowflake accounts in a single organization relate to various cloud providers?

A. Each Snowflake account can be hosted in a different cloud vendor and region.

B. Each Snowflake account must be hosted in a different cloud vendor and region

C. All Snowflake accounts must be hosted in the same cloud vendor and region

D. Each Snowflake account can be hosted in a different cloud vendor, but must be in the same region.

ANSWER: A

Explanation:

Each Snowflake account can be hosted in a different cloud vendor and region. A Snowflake organization is a logical grouping of accounts, and its accounts are not required to share one deployment region or cloud platform. When an account is created, it is associated with a specific cloud platform and Snowflake region, such as AWS in a particular geographic region, Microsoft Azure in a particular region, or Google Cloud in a particular region. The same organization can therefore operate accounts deployed across supported AWS, Azure, and Google Cloud regions. This supports business requirements such as serving users near their locations, complying with data-residency requirements, separating environments, and supporting regional operational needs. Snowflake identifies an account's deployment location through its region and cloud-platform information, which is reflected in account locators and URLs. Although organization-level capabilities can centrally manage and coordinate accounts, they do not impose a requirement that all accounts use the same cloud provider or geographic region. See Snowflake's Snowflake regions and cloud platforms documentation and its Organizations documentation for details.

QUESTION NO: 55

Which Snowflake edition enables data sharing only through Snowflake Support?

A. Virtual Private Snowflake

B. Business Critical

C. Enterprise

D. Standard

ANSWER: A

Explanation:

Virtual Private Snowflake is correct. Virtual Private Snowflake (VPS) is Snowflake's isolated deployment option, designed for organizations with stringent security, regulatory, or network-isolation requirements. A VPS account is isolated from

Snowflake's broader multi-tenant environment, which affects its default data-sharing connectivity. In particular, sharing from a VPS environment to accounts outside that VPS is not enabled by default. Snowflake Support must enable the required capability before an account can share data externally. This controlled process helps preserve the isolation model that VPS customers require while allowing approved sharing scenarios when business requirements call for them. Once the appropriate configuration has been enabled, providers can use Snowflake's secure sharing capabilities without copying the underlying data, subject to the applicable account, region, cloud-platform, and policy requirements. Snowflake documents the VPS isolation and sharing considerations in its Virtual Private Snowflake guidance and describes the general secure-data-sharing model in its data-sharing documentation. See Virtual Private Snowflake and Introduction to Secure Data Sharing.

QUESTION NO: 56

What step can reduce data spilling in Snowflake?

A. Using a larger virtual warehouse

B. Increasing the virtual warehouse maximum timeout limit

C. Increasing the amount of remote storage for the virtual warehouse

D. Using a common table expression (CTE) instead of a temporary table

ANSWER: A

Explanation:

Using a larger virtual warehouse can reduce data spilling because a larger warehouse provides more compute resources, including more memory, to execute queries. Snowflake spills intermediate query-processing data when the data required for operations such as large joins, aggregations, or sorts exceeds the memory available to the warehouse. The spilled data may be written first to local disk and, if needed, to remote storage, which can increase query execution time. Resizing the warehouse upward gives the query more memory capacity and processing power, reducing the likelihood and volume of spill for memory-intensive workloads. Snowflake query profiles expose these conditions through the bytes spilled to local storage and bytes spilled to remote storage metrics, allowing teams to identify queries that would benefit from warehouse right-sizing. Warehouse sizing should be evaluated alongside query design and data-volume considerations, but increasing the warehouse size is a direct operational step for reducing spill in an affected workload. See Snowflake's Query Profile documentation for spill metrics and its Virtual Warehouses documentation for information on warehouse resources and resizing.

QUESTION NO: 57

What SQL command would be used to view all roles that were granted to user.1?

A. show grants to user USER1;

B. show grants of user USER1;

C. describe user USER1;

D. show grants on user USER1;

ANSWER: A

Explanation:

The command `show grants to user USER1;` is the appropriate Snowflake statement for viewing grants assigned directly to the specified user. In Snowflake, the `SHOW GRANTS` command supports different targets through its `TO` clause. Using `TO USER` returns the roles that have been granted to that user, along with any other privileges granted directly to the user. The result set includes grant details such as the granted role or privilege, the object type, the grantee name, and the role that issued the grant. This is useful for reviewing a user's directly assigned access and validating role-based access control configuration. The executing role must have sufficient visibility into grant metadata; Snowflake notes that results

shown by `SHOW GRANTS` depend on the privileges available to the active role. For the formal syntax and supported `SHOW GRANTS` forms, see the Snowflake SHOW GRANTS command reference. For background on granting roles to users in Snowflake's access-control model, see Access control overview.

QUESTION NO: 58

Which steps will reduce memory spillage when running a very large query? (Select TWO)

A. Enable auto-scale mode on the virtual warehouse.

B. Increase the network bandwidth.

C. Process the data in smaller batches.

D. Add a DISTINCT clause to the query.

E. Increase the size of the virtual warehouse.

ANSWER: C E

Explanation:

Processing the data in smaller batches reduces the amount of intermediate data—such as rows being sorted, joined, aggregated, or held for window-function processing—that must be retained at one time. This lowers peak memory demand for each execution step and can prevent the query from spilling intermediate results beyond local memory. Where the workload and business logic permit it, breaking a large operation into staged, smaller operations is therefore a practical way to reduce resource pressure.

Increasing the size of the virtual warehouse is also an effective response because larger Snowflake warehouses provide more compute resources, including memory available to query-processing operations. Snowflake records bytes spilled to local and remote storage in query profiles; significant spill volume is an indicator that the query's intermediate processing requirements exceeded the memory available on the executing warehouse. Resizing to a larger warehouse can provide sufficient memory for these operations and reduce or eliminate spilling. Snowflake specifically recommends considering a larger warehouse when a query spills a significant amount of data to remote storage. See the Snowflake Query Profile documentation and Virtual Warehouses overview.

QUESTION NO: 59

What are benefits of using Snowpark with Snowflake? (Select TWO).

A. Snowpark uses a Spark engine to generate optimized SQL query plans.

B. Snowpark automatically sets up Spark within Snowflake virtual warehouses.

C. Snowpark does not require that a separate cluster be running outside of Snowflake.

D. Snowpark allows users to run existing Spark code on virtual warehouses without the need to reconfigure the code.

E. Snowpark executes as much work as possible in the source databases for all operations including User-Defined Functions (UDFs).

F. Snowpark enables developers to write data-processing code in supported languages, such as Python, Java, and Scala, and execute it in Snowflake.

ANSWER: C F

Explanation:

Snowpark does not require that a separate cluster be running outside of Snowflake because Snowpark workloads are executed using Snowflake-managed compute, such as virtual warehouses for stored procedures and user-defined functions.

This keeps data processing close to the data governed in Snowflake and removes the operational overhead of provisioning, securing, scaling, and maintaining an external processing cluster for these workloads.

Snowpark enables developers to write data-processing code in supported, familiar languages, including Python, Java, and Scala, and execute that code within Snowflake's governed environment. Its DataFrame API lets developers express transformations programmatically; Snowflake then evaluates the operations and executes the resulting work in Snowflake. This supports building data pipelines, machine-learning workflows, and application logic without moving data to a separate compute platform. Snowflake documents Snowpark as a developer framework for processing data in Snowflake and documents that Snowpark code runs with Snowflake compute resources, depending on the feature being used. See Snowpark documentation and Snowpark stored procedures documentation.

QUESTION NO: 60

What are the responsibilities of Snowflake ' s Cloud Service layer? (Choose three.)

A. Authentication

B. Resource management

C. Virtual warehouse caching

D. Query parsing and optimization

E. Query execution

F. Physical storage of micro-partitions

ANSWER: A B D

Explanation:

Snowflake's Cloud Services layer is the intelligence and coordination tier of Snowflake's architecture. **Authentication** is a Cloud Services responsibility because this layer validates users and supports secure access to Snowflake before users perform work. **Resource management** is also handled through Cloud Services: it coordinates and manages the resources needed for Snowflake operations, including the virtual warehouses that provide compute capacity. This centralized management lets Snowflake separate compute administration from the execution resources themselves. **Query parsing and optimization** belongs to Cloud Services because submitted SQL is parsed, compiled, and optimized into an execution plan before a virtual warehouse processes the work. The optimization process uses Snowflake's metadata and statistics to select an efficient approach for retrieving and processing data. Together, these responsibilities show that Cloud Services manages the control-plane functions that authenticate requests, coordinate resources, and prepare queries for execution, while the independently scalable compute and storage layers perform their respective data-processing and storage functions. Snowflake documents these functions in its architecture overview and describes query processing as a Cloud Services activity before compute execution. See Snowflake key concepts and architecture and Virtual warehouses overview.

QUESTION NO: 61

Which Snowflake object can be accessed in he FROM clause of a query, returning a set of rows having one or more columns?

A. A User-Defined Table Function (UDTF)

B. A Scalar User Function (UDF)

C. A stored procedure

D. A task

ANSWER: A

Explanation:

A User-Defined Table Function (UDTF) is correct because it produces tabular output: a set of zero or more rows containing one or more columns. Snowflake invokes a UDTF in a query's `FROM` clause by using the `TABLE(...)` construct, allowing its returned rows to be selected, filtered, joined, grouped, and otherwise processed in the same way as other relational query sources. For example, a UDTF that accepts an input value and returns multiple related records can be referenced as `FROM TABLE(function_name(...))`. UDTFs are useful when reusable logic must return a relation rather than one individual value. Snowflake supports SQL, JavaScript, Java, Python, and Scala implementations for table functions, subject to the applicable runtime and handler support. The declared return schema defines the columns available to the calling query, ensuring that the UDTF integrates cleanly into standard SQL operations. See Snowflake's User-Defined Table Functions documentation and its table function SQL reference for syntax and usage details.

QUESTION NO: 62

How does Snowflake support the use of structured and semi-structured data? (Select TWO)

A. Structured data is supported through the use of traditional relational database tables, organized into rows and columns.

B. Structured and semi-structured data are stored in separate physical storage locations; users must choose which type of data to use when creating a table.

C. Semi-structured data is supported through the use of automatically indexed database tables.

D. All semi-structured data is automatically converted into structured data when it is loaded into Snowflake.

E. Users can load and query semi-structured data using SQL using the SQL semi-structured data handling functions.

ANSWER: A E

Explanation:

Structured data is supported through the use of traditional relational database tables, organized into rows and columns. Snowflake tables use a familiar relational model: each column has a defined data type, and rows represent records. This allows organizations to load and query conventional tabular data with SQL while benefiting from Snowflake's scalable storage and compute architecture.

Users can load and query semi-structured data using SQL using the SQL semi-structured data handling functions. Snowflake natively supports common semi-structured formats, including JSON, Avro, ORC, Parquet, and XML. Data can be stored in the `VARIANT`, `OBJECT`, and `ARRAY` data types, preserving hierarchical structures while making individual elements accessible through SQL. Functions such as `PARSE_JSON` and `FLATTEN`, along with path notation, enable users to extract, transform, and analyze nested attributes without first fully modeling them as relational columns. This lets structured and semi-structured information coexist in the same Snowflake environment and be queried with standard SQL workflows. See Snowflake's semi-structured data overview and semi-structured data types documentation.

QUESTION NO: 63

When unloading to a stage, which of the following is a recommended practice or approach?

A. Set SINGLE: = true for larger files

B. Use OBJECT_CONSTRUCT (*) when using Parquet

C. Avoid the use of the CAST function

D. Define an individual file format

ANSWER: B

Explanation:

`Use OBJECT_CONSTRUCT ( * ) when using Parquet` is the recommended approach. `OBJECT_CONSTRUCT(*)` creates an object from the columns in each selected row, producing a single `VARIANT` value that Snowflake can serialize as

a nested Parquet record during the unload. This is particularly useful when exporting relational rows to Parquet because it retains the row's column names as object fields and provides a clear, self-describing structure for downstream consumers. The expression can be placed in the query supplied to `COPY INTO <location>`, for example by selecting `OBJECT_CONSTRUCT(*)` from the source table and specifying a Parquet file format. Snowflake documents use of `OBJECT_CONSTRUCT` for representing row data as an object, and its unloading guidance covers using query transformations with `COPY INTO` to prepare exported data. See the OBJECT_CONSTRUCT reference and Snowflake's data unloading overview.

QUESTION NO: 64

What are advantages clones have over tables created with CREATE TABLE AS SELECT statement? (Choose two.)

A. The clone always stays in sync with the original table.

B. The clone has better query performance.

C. The clone is created almost instantly.

D. The clone will have time travel history from the original table.

E. The clone saves space by not duplicating storage.

ANSWER: C E

Explanation:

"The clone is created almost instantly." and "The clone saves space by not duplicating storage." are advantages of Snowflake zero-copy cloning compared with creating a table through CREATE TABLE AS SELECT (CTAS). A clone is created as a metadata operation: initially, it references the same immutable micro-partitions as its source object rather than copying all of the source data. Consequently, cloning generally completes very quickly, even for a large table, and does not immediately incur storage for a full second copy of the data. Storage begins to diverge only when changes are made to either the source or clone, because Snowflake retains the micro-partitions needed by each independent version. In contrast, CTAS executes a query and materializes its result into a newly created table, requiring compute resources and storage for the resulting data. The clone and its source are independent after creation, so subsequent changes do not synchronize between them. Snowflake documents the metadata-based behavior and storage efficiency of zero-copy cloning in its Cloning considerations documentation. Details of CTAS syntax and behavior are available in the CREATE TABLE reference.

QUESTION NO: 65

What is the default file size when unloading data from Snowflake using the COPY command?

A. 5 MB

B. 8 GB

C. 16 MB

D. 32 MB

ANSWER: C

Explanation:

16 MB is correct. When data is unloaded from a Snowflake table or query to an external stage with `COPY INTO <location>`, Snowflake uses the `MAX_FILE_SIZE` copy option to target the maximum size of each output file. Its default value is 16,777,216 bytes, which is 16 MB. This setting is a target rather than an absolute guarantee: Snowflake can produce files that vary somewhat in size based on data distribution, compression, file format, partitioning, and the parallel processing used for the unload operation. The value can be adjusted in the `COPY INTO` statement when a workload needs a different output-file size, subject to the documented supported range. For example, choosing a larger target size can reduce the number of files generated, while a smaller size can be useful for downstream systems that impose file-size limits.

Snowflake's syntax reference explicitly documents the default `MAX_FILE_SIZE` value as 16,777,216 bytes. See the COPY INTO <location> reference and Snowflake's data unloading considerations for details on output files and unload behavior.

QUESTION NO: 66

How can the outer array structure of a semi-structured file be removed?

A. Use the parameter strip_outer_array = true in a COPY INTO < table > command.

B. Set the file format to eliminate any outer array structure before initiating the COPY INTO < table > command.

C. Filter the outer array structure using a PUT command with the include_outer_array = false parameter.

D. Use the FLATTEN command with the outer_array = false parameter.

ANSWER: A

Explanation:

Use the parameter strip_outer_array = true in a COPY INTO < table > command. In Snowflake, `STRIP_OUTER_ARRAY` is a JSON file-format option that instructs the loader to remove one outermost array from a JSON document during loading. This is useful when a file is structured as a single top-level array whose elements are the individual records to load. With this setting enabled, Snowflake treats each element of that outer array as a separate record rather than loading the entire array as one value. The setting can be included in an inline `FILE_FORMAT` clause in the `COPY INTO` statement or defined in a named JSON file format referenced by that command. For example, an inline file format can specify `TYPE = JSON STRIP_OUTER_ARRAY = TRUE` while loading staged files into a table. Snowflake documents that this option applies to JSON data and removes the outer array element before loading, allowing the enclosed elements to be processed individually. See Snowflake's JSON file format options and the COPY INTO <table> reference for syntax and loading behavior.

QUESTION NO: 67

Which COPY INTO < table > options can be used to load data from a staged file when the metadata in the file has expired? (Select TWO)

A. LOAD_UNCERTAIN_FILES

B. LOAD_EXPIRED_FILES

C. FORCE

D. ENFORCE_LENGTH

E. PURGE

ANSWER: A C

Explanation:

`LOAD_UNCERTAIN_FILES` and `FORCE` can be used when Snowflake's load metadata is no longer available for a staged file. Snowflake retains file-load metadata to prevent an already loaded file from being loaded again, but this metadata expires after 64 days. When metadata has expired, Snowflake cannot determine with certainty whether a file was previously loaded. Setting `LOAD_UNCERTAIN_FILES = TRUE` explicitly permits loading files whose load status is uncertain because their metadata has expired. This setting is intended for files that have been staged for more than 64 days and should be used only when the operator has determined that reloading the files is appropriate. Setting `FORCE = TRUE` instructs `COPY INTO <table>` to load files regardless of Snowflake's load-history determination. It can therefore be used to reload a specific file or set of files when the normal duplicate-load protections would otherwise prevent or make the load uncertain. Both settings require care because they can result in duplicate rows if a file was in fact loaded previously. See Snowflake's COPY INTO <table> reference and its guidance on data loading considerations.

QUESTION NO: 68

Which of the following indicates that it may be appropriate to use a clustering key for a table? (Select TWO).

A. The table contains a column that has very low cardinality

B. DML statements that are being issued against the table are blocked

C. The table has a small number of micro-partitions

D. Queries on the table are running slower than expected

E. The clustering depth for the table is large

ANSWER: D E

Explanation:

“Queries on the table are running slower than expected” can indicate that the table is not benefiting from effective micro-partition pruning. A clustering key may improve performance when a large table is frequently queried with selective predicates or range filters on columns whose values are not naturally co-located. By organizing related values into fewer overlapping micro-partitions, clustering can reduce the amount of data scanned and improve query efficiency. Before defining a key, Snowflake recommends evaluating the query patterns and ensuring that the expected pruning benefit justifies the ongoing maintenance cost.

“The clustering depth for the table is large” is a direct indicator that a clustering key may be useful, particularly when the table already has a defined key or when candidate key columns reflect common selective query filters. Clustering depth measures the average overlap of micro-partitions; higher depth means values are distributed across more overlapping partitions. Reducing that overlap can enable more effective pruning. Snowflake provides the `SYSTEM$CLUSTERING_INFORMATION` function to assess clustering quality and help identify whether reclustering or a clustering key could provide value. See Snowflake’s clustering key guidance and SYSTEM$CLUSTERING_INFORMATION documentation.

QUESTION NO: 69

Which statement accurately describes a characteristic of a materialized view?

A. A materialized view can query only a single table.

B. Data accessed through materialized views can be stale.

C. Materialized view refreshes need to be maintained by the user.

D. Querying a materialized view is slower than executing a query against the base table of the view.

ANSWER: A

Explanation:

A materialized view can query only a single table. In Snowflake, a materialized view is a precomputed result set defined over one base table, subject to the SQL restrictions for materialized-view definitions. This single-table limitation means that a materialized view definition cannot include joins between multiple tables. The purpose of the object is to persist computed results for an eligible query pattern so Snowflake can use those results to reduce the work required at query time. Snowflake manages materialized-view maintenance automatically as data changes in the base table; users do not schedule or manually operate refresh jobs for the view. Materialized views are particularly useful where repeated queries perform expensive filtering, projection, or aggregation against a large base table and the cost of maintaining the view is justified by query-performance benefits. The single-base-table rule is an explicit Snowflake limitation and is therefore the accurate defining characteristic in this question. See Snowflake’s Materialized Views documentation and its documented limitations on creating materialized views.

QUESTION NO: 70

What does Snowflake attempt to do if any of the compute resources for a virtual warehouse fail to provision during start-up?

A. Repair the failed resources.

B. Restart failed resources.

C. Queue the failed resources

D. Provision the failed resources

ANSWER: B

Explanation:

Restart failed resources. is correct. A virtual warehouse consists of one or more Snowflake-managed compute resources that are provisioned when the warehouse starts or resumes. Snowflake manages this infrastructure layer automatically, including responding to transient provisioning or resource-level failures without requiring an administrator to intervene. If a compute resource does not successfully provision during warehouse startup, Snowflake attempts to restart that failed resource so the warehouse can obtain the compute capacity required for its configured size and process workload. This behavior is part of Snowflake's fully managed warehouse model: customers configure warehouse properties such as size, scaling, and suspension behavior, while Snowflake operates the underlying compute infrastructure. The restart attempt helps make warehouse startup resilient to temporary infrastructure issues and supports the expected availability of the virtual warehouse. See Snowflake's Virtual warehouses overview and Introduction to virtual warehouses for details on Snowflake-managed warehouse compute and lifecycle behavior.

QUESTION NO: 71

A running virtual warehouse is suspended.

What is the MINIMUM amount of time that the warehouse will incur charges for when it is restarted?

A. 1 second

B. 60 seconds

C. 5 minutes

D. 60 minutes

ANSWER: B

Explanation:

60 seconds is correct. Snowflake bills virtual warehouse compute usage per second while a warehouse is running, but applies a 60-second minimum each time a warehouse is started or resumed. Therefore, when the suspended warehouse is restarted, it incurs at least one minute of warehouse credit consumption, even if it runs for less than 60 seconds before being suspended again. After that initial one-minute minimum has been met for the restart event, billing continues on a per-second basis for the time that the warehouse remains running. This behavior is important when configuring auto-suspend: very short idle periods can cause repeated resume events, each of which is subject to the one-minute minimum. Selecting an appropriate auto-suspend interval helps balance responsiveness for users and workloads with efficient credit consumption. Snowflake documents that warehouse usage is charged by the second, with a minimum of 60 seconds whenever a warehouse starts or resumes. See the Snowflake virtual warehouse credit usage documentation and Understanding compute cost.

QUESTION NO: 72

What does an integration between Snowflake and Microsoft Private Link or AWS PrivateLink support?

A. The isolation of data within a Snowflake account.

B. The use of Secure Data Sharing among Snowflake accounts.

C. A Virtual Private Network (VPN) between a user and Snowflake.

D. A secure, direct connection to Snowflake that does not use the internet.

ANSWER: D

Explanation:

A secure, direct connection to Snowflake that does not use the internet is correct. Snowflake supports private connectivity through cloud-provider services such as AWS PrivateLink and Azure Private Link. This lets clients access Snowflake through private endpoints using the cloud provider's private network rather than routing traffic over the public internet. The integration is designed to provide private network paths between an organization's virtual network environment and Snowflake service endpoints, which can help organizations meet network-isolation and security requirements. In Snowflake, private connectivity can cover account URLs and, depending on the cloud platform and configured features, other supported endpoints such as Snowflake's internal stages or Snowsight. Private connectivity does not alter Snowflake's authentication, authorization, or encryption model; instead, it changes the network route used to reach the service. Configuration requires the relevant Snowflake account edition and cloud-provider networking setup, including approval and DNS configuration so that Snowflake hostnames resolve to the private endpoint. Snowflake documents these services as private connectivity options specifically intended to keep supported traffic off the public internet while preserving access to the Snowflake account. See Snowflake's private connectivity documentation and AWS's AWS PrivateLink overview.

QUESTION NO: 73

Which methods can be used to delete staged files from a Snowflake stage? (Choose two.)

A. Use the DROP < file > command after the load completes.

B. Specify the TEMPORARY option when creating the file format.

C. Specify the PURGE copy option in the COPY INTO < table > command.

D. Use the REMOVE command after the load completes.

E. Use the DELETE LOAD HISTORY command after the load completes.

ANSWER: C D

Explanation:

Specifying the `PURGE` copy option in the `COPY INTO <table>` command can remove source files from a stage after Snowflake successfully loads them. This is useful when staged data is intended to be transient and should not remain available after ingestion. The option is specified as `PURGE = TRUE`; Snowflake removes files only after a successful load, so it supports an automated load-and-cleanup workflow. For external stages, the Snowflake storage integration must have permission to delete objects in the cloud storage location.

Using the `REMOVE` command after the load completes is the direct administrative method for deleting one or more files from an internal or external stage. It supports a stage path and an optional pattern, allowing targeted cleanup rather than removing all staged data. Appropriate authorization and, for external stages, cloud-storage delete permissions are required. Snowflake documents `PURGE` as a copy option for deleting loaded files and documents `REMOVE` specifically for removing files from stages. See the COPY INTO <table> reference and the REMOVE command reference.

QUESTION NO: 74

What factors impact storage costs in Snowflake? (Select TWO).

A. The account type

B. The storage file format

C. The cloud region used by the account

D. The type of data being stored

E. The cloud platform being used

ANSWER: A C

Explanation:

Snowflake storage is billed using a per-terabyte, per-month rate calculated from the average daily amount of compressed data stored. The account type affects the commercial pricing arrangement applied to that storage. For example, an on-demand account uses published consumption prices, while a capacity account uses pricing established under its capacity contract. Therefore, the applicable account type can affect the storage rate used for billing.

The cloud region used by the account also affects storage cost because Snowflake pricing is region-specific. Storage rates are published or contractually established for the cloud region where the Snowflake account is deployed. As a result, otherwise equivalent storage consumption can have a different monetary cost when accounts are located in different regions. These factors concern the rate charged for stored data; Snowflake then applies that rate to the measured average storage usage over the billing period. Snowflake documents its storage billing model and regional pricing information in its cost and pricing documentation. See Understanding overall cost and Snowflake pricing.

QUESTION NO: 75

Which common query problems are identified by the Query Profile? (Select TWO.)

A. Syntax error

B. Inefficient pruning

C. Ambiguous column names

D. Queries too large to fit in memory

E. Object does not exist or not authorized

ANSWER: B D

Explanation:

Inefficient pruning and **Queries too large to fit in memory** are common performance issues that Snowflake Query Profile can reveal. Query Profile presents the query execution plan and operator-level statistics, including the amount of data scanned, partitions scanned, rows processed, elapsed time, and bytes spilled to local or remote storage. These details help an engineer determine whether micro-partition pruning is limiting scans effectively. When a filter does not eliminate enough micro-partitions, scan operators can show unexpectedly large volumes of data being read, which indicates an opportunity to improve filtering, clustering strategy, or predicate design.

Query Profile also exposes memory pressure during execution. A query that cannot retain its intermediate data in warehouse memory may spill data to local or remote disk storage. Spill metrics shown for relevant operators make this visible and help identify queries that may benefit from a larger warehouse, reduced intermediate result sizes, or a revised query design. Snowflake documents Query Profile as a tool for analyzing query execution details and diagnosing bottlenecks, while its performance guidance specifically describes pruning and spilling as important optimization signals. See the Snowflake Query Profile documentation and common query problems identified by Query Profile.

QUESTION NO: 76

Which privilege is required to grant a database role to another role?

A. USAGE on the database

B. OWNERSHIP on the database role

C. CREATE ROLE on the account

D. MANAGE GRANTS on the account

ANSWER: B

Explanation:

OWNERSHIP on the database role is required because a database role is a securable object whose owner controls its role grants. Snowflake's `GRANT DATABASE ROLE` command requires the executing role to hold the OWNERSHIP privilege on the database role being granted. This authority allows the owner to manage the database role's membership, including granting that database role to an account role or to another database role where the role hierarchy supports the relationship.

Database roles are scoped to a single database and are intended to encapsulate privileges on objects in that database. Their ownership therefore provides the administrative control needed to decide which other roles inherit those privileges. For example, a role that owns `MY_DB.ANALYST_DB_ROLE` can execute a statement such as `GRANT DATABASE ROLE MY_DB.ANALYST_DB_ROLE TO ROLE ANALYST;`, subject to the applicable database-role grant rules. This requirement ensures that delegation of the permissions collected in a database role remains under the control of its designated owner. Snowflake documents the command-specific access-control requirement in its GRANT DATABASE ROLE reference; see also the overview of Snowflake access control.

QUESTION NO: 77

What kind of value does a User-Defined Function (UDF) return? (Select TWO).

A. Dictionary

B. List

C. Object

D. Scalar

E. Tabular

ANSWER: D E

Explanation:

Snowflake supports two fundamental UDF return forms: scalar values and tabular results. A scalar UDF returns one value for each invocation, and its declared return type can be a standard Snowflake data type, including semi-structured types such as VARIANT, ARRAY, or OBJECT where appropriate. Scalar UDFs are commonly used in expressions in SELECT lists, predicates, and other SQL contexts that expect a single value. A tabular UDF, also called a user-defined table function (UDTF), returns a relation: zero or more rows with one or more columns defined by its RETURNS TABLE clause. It is invoked in the FROM clause through the TABLE function syntax, enabling its output to be queried and joined like a table. Thus, the applicable return categories are **Scalar** and **Tabular**. Snowflake's UDF overview distinguishes scalar functions, which return a single value, from tabular functions, which return tabular results. See the Snowflake UDF overview and the documentation for SQL tabular UDFs.

QUESTION NO: 78

What happens when a Snowflake user changes the data retention period at the schema level?

A. All child objects will retain data for the new retention period.

B. All child objects that do not have an explicit retention period will automatically inherit the new retention period.

C. All child objects with an explicit retention period will be overridden with the new retention period.

D. All explicit child object retention periods will remain unchanged.

ANSWER: B

Explanation:

All child objects that do not have an explicit retention period will automatically inherit the new retention period. Snowflake manages `DATA_RETENTION_TIME_IN_DAYS` through a parameter hierarchy: account settings can be inherited by databases, database settings by schemas, and schema settings by eligible objects within that schema, such as tables. A setting applied directly at a more specific object level takes precedence, while an object without its own configured value uses the inherited value from its parent scope.

Therefore, after the schema's retention setting is changed, the effective Time Travel retention period is updated for child objects that rely on the schema-level value. This allows an administrator to establish a consistent retention policy for objects in a schema without needing to configure each object individually. The retention period controls how long historical data is available for Time Travel operations, subject to the edition-specific limits and Snowflake's documented behavior for different object types. See Snowflake's Data Retention Period documentation and the DATA_RETENTION_TIME_IN_DAYS parameter reference.

QUESTION NO: 79

What command will return a list of files that have been unloaded to a user stage?

A. DESC @~

B. GET @~

C. SHOW @~

D. LIST @~

ANSWER: D

Explanation:

`LIST @~` returns the files stored in the current user's stage. In Snowflake, `@~` is the stage reference for the current user stage, which is a personal, automatically provisioned internal stage associated with that user. Files created by unloading query results with `COPY INTO @~`, or otherwise uploaded to that user stage, can be enumerated by issuing the `LIST` command against this reference. The output includes file metadata such as the file name, size, MD5 checksum, and last-modified timestamp, allowing the user to confirm that the unload produced the expected staged files before downloading or processing them further. A path can also be appended to the stage reference when files were written to a particular directory-like prefix, such as `LIST @~/exports/`. Snowflake documents `LIST` specifically as the SQL command for listing files in a stage, including user stages. See the Snowflake LIST command reference and the documentation on user stages.

QUESTION NO: 80

When reviewing a query profile, what is a symptom that a query is too large to fit into the memory?

A. A single join node uses more than 50% of the query time

B. Partitions scanned is equal to partitions total

C. An AggregateOperacor node is present

D. The query is spilling to remote storage

ANSWER: D

Explanation:

The query is spilling to remote storage is the relevant query-profile symptom. Snowflake query processing uses the memory available to the virtual warehouse for intermediate query data, including data needed for operations such as joins, aggregations, and sorts. When the working data set exceeds available memory, Snowflake first spills data to local disk and can then spill to remote storage. Remote spilling indicates more substantial memory pressure and typically has a significant performance cost because intermediate data must be written to and read from remote storage rather than remaining in memory. In Query Profile, this behavior is exposed through the operator statistics for bytes spilled to local storage and bytes spilled to remote storage. A nonzero remote-spill value is therefore a clear signal that the query's intermediate processing requirements exceeded the warehouse memory capacity available for that execution. This information helps identify candidates for query tuning, such as reducing intermediate result sizes, improving join selectivity, or selecting a larger warehouse when appropriate. Snowflake documents that remote disk spilling can adversely affect performance and recommends reviewing these metrics in Query Profile when diagnosing inefficient queries. See Using Query Profile and Optimizing warehouses for performance.

QUESTION NO: 81

What tasks can be completed using the copy command? (Select TWO)

A. Columns can be aggregated

B. Columns can be joined with an existing table

C. Columns can be reordered

D. Columns can be omitted

E. Data can be loaded without the need to spin up a virtual warehouse

ANSWER: C D

Explanation:

Columns can be reordered and **Columns can be omitted** are supported during bulk loading with Snowflake's `COPY INTO` command. A load statement can specify a target-column list, which determines the destination columns receiving the incoming values. This lets a load map source fields to table columns in an order that differs from the physical order of columns in the destination table. The source expression list can likewise select only the fields required for the load, so not every source-file column needs to be loaded. This is particularly useful when staged files contain extra attributes, when only a subset of a target table should receive values, or when a file layout does not directly match the destination schema. The column list and source query provide explicit control over the mapping, while normal table behavior applies to any target columns not included in the load, such as using defaults where defined or accepting null values where permitted. Snowflake documents column selection and transformation within `COPY INTO <table>`, including specifying target columns and selecting file columns through a query. See the COPY INTO <table> reference and Snowflake's data loading and transformation guidance.

QUESTION NO: 82

What target locations can be used when unloading data using the COPY INTO < location > command? (Select TWO)

A. A Snowflake schema

B. A Snowflake stage

C. A local file system

D. A Snowflake stream

E. Cloud storage in Google (GCP), AWS, or Microsoft Azure

ANSWER: B E

Explanation:

`COPY INTO <location>` unloads the results of a query or table data from Snowflake into files at a stage or a supported cloud-storage location. A Snowflake stage is a valid target because stages provide a Snowflake-recognized file location; this includes internal stages managed by Snowflake and external stages that point to cloud object storage. When a stage is used as the destination, it is specified with the stage syntax, such as `@my_stage`, and Snowflake writes the unloaded files there according to the selected file format and copy options.

Cloud storage in Google (GCP), AWS, or Microsoft Azure is also a valid target. Snowflake supports unloading directly to Amazon S3, Google Cloud Storage, and Microsoft Azure storage by supplying the appropriate storage URL, such as an `s3://`, `gcs://`, or `azure://` location, provided the required access integration or credentials and permissions are configured. These destinations are designed for scalable file export and downstream use outside Snowflake. Snowflake documents both stage paths and cloud-storage URLs as supported unload destinations for this command. See COPY INTO <location> and Overview of data unloading.

QUESTION NO: 83

Which command can be used to load data files into a Snowflake stage?

A. JOIN

B. COPY INTO

C. PUT

D. GET

ANSWER: C

Explanation:

The `PUT` command uploads data files from a local file system to a Snowflake internal stage. It can target a user stage, a table stage, or a named internal stage, making the files available for subsequent loading into a table with `COPY INTO <table>`. For example, `PUT file:///tmp/data.csv @my_internal_stage;` transfers the local file to the specified stage. Snowflake clients that support `PUT`, including SnowSQL, can perform this upload. The command also supports options such as automatic compression, parallel upload operations, and overwrite behavior. A stage is Snowflake's location for temporarily or persistently storing files used in data loading and unloading workflows. The essential distinction is that `PUT` moves local files into an internal Snowflake stage; after the file is staged, a loading command can ingest its contents into a database table. See the Snowflake PUT command reference and the loading data from a local file system guide.

QUESTION NO: 84

Which data types does Snowflake support when querying semi-structured data? (Select TWO)

A. VARIANT

B. ARRAY

C. VARCHAR

D. XML

E. BLOB

ANSWER: A B

Explanation:

Snowflake natively supports semi-structured data through types including VARIANT, ARRAY, and OBJECT. Of the listed choices, **VARIANT** and **ARRAY** are the applicable semi-structured data types. VARIANT is Snowflake's flexible universal type for storing values with differing or nested structures, such as JSON documents and data loaded from formats including

Avro, ORC, Parquet, and XML. A VARIANT value can contain nested arrays and objects, and Snowflake provides path notation, bracket notation, and FLATTEN to access and transform nested content during queries.

ARRAY represents an ordered collection of values and is also a native semi-structured type. It is useful when the source data includes lists, repeated elements, or nested collections. ARRAY values can be stored directly or embedded within a VARIANT value, then queried through array indexing or expanded into rows with the FLATTEN table function. These capabilities allow Snowflake to preserve hierarchical source structures while still enabling SQL-based analysis. Snowflake documents VARIANT, ARRAY, and OBJECT as its semi-structured data types; therefore, the two listed types that meet the question's requirement are VARIANT and ARRAY. See Snowflake's semi-structured data type documentation and its guide to querying semi-structured data.

QUESTION NO: 85

What are characteristics of transient tables in Snowflake? (Select TWO).

A. Transient tables have a Fail-safe period of 7 days.

B. Transient tables can be cloned to permanent tables.

C. Transient tables persist until they are explicitly dropped.

D. Transient tables can be altered to make them permanent tables.

E. Transient tables have Time Travel retention periods of 0 or 1 day.

ANSWER: B C E

Explanation:

Transient tables are intended for data that must remain available beyond an individual session but does not require the same level of data-recovery protection as permanent tables. They persist until they are explicitly dropped, so they are suitable for staging, intermediate processing, and other short-lived operational datasets that may be used across sessions. Snowflake also supports cloning a transient table into a permanent table. This enables a workload to retain a durable permanent copy when needed, while the source remains transient. In addition, transient tables support a Time Travel retention period of either 0 or 1 day. A retention value of 0 disables Time Travel for the object, whereas 1 day provides a limited recovery window. This limited retention is a defining storage and recovery characteristic of transient objects; unlike permanent objects, transient objects do not have a Fail-safe period. Snowflake documents the supported data-retention behavior for transient tables and explains that clones can be created with a different table type, including creating a permanent clone from a transient source. See the Snowflake documentation on temporary and transient tables and Snowflake documentation on zero-copy cloning.

QUESTION NO: 86 - (SIMULATION)

Which object can be used to query the data loading history for the last 365 days for a Snowflake account?

The SNOWFLAKE.ACCOUNT_USAGE.COPY_HISTORY view

The SNOWFLAKE.ACCOUNT_USAGE.DATA_TRANSFER_HISTORY view

The database INFORMATION_SCHEMA.COPY_HISTORY table function

The database INFORMATION_SCHEMA.DATA_TRANSFER_HISTORY table function

ANSWER: See the explanation for the answer

Explanation:

Correct answer: `SNOWFLAKE.ACCOUNT_USAGE.COPY_HISTORY` view.

This Account Usage view provides data-loading history for COPY INTO operations across the Snowflake account and retains history for up to **365 days**.

The `INFORMATION_SCHEMA.COPY_HISTORY` table function has a much shorter history window (typically 14 days), while the DATA_TRANSFER_HISTORY objects are for data transfer activity rather than loading data into tables.

QUESTION NO: 87

Which Snowflake feature enables loading data from cloud storage as soon as files are available in a stage?

A. copy_into < location > command

B. Data replication

C. Snowpipe

D. Direct share

ANSWER: C

Explanation:

Snowpipe is correct because it provides Snowflake's continuous, automated data-ingestion service for files delivered to a stage. When configured with cloud event notifications, Snowpipe detects newly available files in an external stage and loads them into a target table shortly after they arrive, without requiring a user to manually run a load command each time. This event-driven approach is designed for near-real-time ingestion from supported cloud storage platforms and allows organizations to keep tables current as files are produced by upstream systems. A pipe defines the loading instructions, including the target table and the COPY statement Snowpipe executes on the arriving files. Snowflake manages the compute resources used by the serverless ingestion service, so users do not need to provision or manage a virtual warehouse for normal Snowpipe loading. Snowpipe also tracks loaded files to help avoid loading the same staged file repeatedly. For setup, the storage location must be configured as a stage and integrated with the applicable cloud notification mechanism so that file-arrival events can trigger the pipe. See Snowflake's Snowpipe introduction and automating Snowpipe with cloud event notifications documentation.

QUESTION NO: 88

Which of the following features, associated with Continuous Data Protection (CDP), require additional Snowflake-provided data storage? (Choose two.)

A. Tri-Secret Secure

B. Time Travel

C. Fail-safe

D. Data encryption

E. External stages

ANSWER: B C

Explanation:

Time Travel and **Fail-safe** require additional Snowflake-provided storage because both preserve historical versions of changed or deleted data beyond the currently active version of a table. With Time Travel, Snowflake retains prior data states for the configured retention period, enabling users to query, restore, or clone data as it existed at an earlier point in time. The storage occupied by retained historical data contributes to Time Travel storage usage.

After the Time Travel retention period ends, Fail-safe provides a further seven-day recovery window intended for Snowflake-assisted disaster recovery. Snowflake retains the applicable historical data during this period, and that retained data also consumes Fail-safe storage. These capabilities are part of Snowflake's continuous data protection model: they preserve recoverable data versions without requiring customers to maintain their own backup copies. Snowflake documents that both

Time Travel and Fail-safe storage are billed according to the amount of retained data. See Understanding and using Time Travel and Understanding Continuous Data Protection.

QUESTION NO: 89

What are ways to create and manage data shares in Snowflake? (Select TWO)

A. Through the Snowflake web interface (UI)

B. Through the DATA_SHARE=TRUE parameter

C. Through SQL commands

D. Through the enable__share=true parameter

E. Using the CREATE SHARE AS SELECT * TABLE command

ANSWER: A C

Explanation:

Through the Snowflake web interface (UI) and through SQL commands are valid ways to create and manage Snowflake shares. In the Snowflake web interface, users can use Snowsight to work with shares through a graphical workflow. This includes viewing shares and, for users with the necessary privileges, creating a share, adding database objects, specifying consumer accounts, and monitoring shared data. The interface is useful when administrators want an interactive way to configure and review sharing relationships.

SQL provides the programmatic and automatable method for the same core sharing lifecycle. A role with the required privileges can issue statements such as `CREATE SHARE`, `GRANT USAGE`, `GRANT SELECT`, `ALTER SHARE ... ADD ACCOUNTS`, and `DROP SHARE`. SQL is particularly appropriate for repeatable deployment processes, administrative scripts, and infrastructure-as-code workflows. A share does not contain copied data; rather, the provider grants consumers access to selected objects in the provider account. Snowflake documents both the SQL commands used to create and administer shares and the Snowsight workflow for managing shared data. See CREATE SHARE and Introduction to Secure Data Sharing.