PassQueen
a.k.a dumpsqueen.com

SnowPro Core Certification Exam

Snowflake SnowPro-Core

Version Demo

Total Demo Questions: 77

Total Premium Questions: 775

Buy Premium PDF

Topic Break Down

Topic	No. of Questions
Topic 1, Snowflake AI Data Cloud Features and Architecture	126
Topic 2, Account Access and Security	170
Topic 3, Performance Concepts	167
Topic 4, Data Loading and Unloading	143
Topic 5, Data Transformations	53
Topic 6, Data Protection and Data Sharing	116
Total	775

QUESTION NO: 1

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: A E

Explanation:

Configuring the user to be initially disabled is recommended because it supports a controlled provisioning workflow. An administrator can create the identity, assign the appropriate roles, configure authentication and network policies, and validate the account setup before permitting the user to sign in. Snowflake exposes the `DISABLED` user property specifically to prevent login while retaining the user object and its configuration. This is particularly useful when user provisioning is automated or when access must be approved before activation.

Setting the user's access to expire within a specified timeframe is also recommended because Snowflake supports an `EXPIRES_AT` property for time-bounded access. An expiration date helps enforce least privilege for temporary staff, contractors, migrations, training, or other limited-duration requirements. It reduces reliance on manual follow-up to remove access and provides a clear lifecycle boundary for the account. Administrators can later extend or remove the expiration if the user's approved business need continues. These settings are available in Snowflake user-management DDL and are valuable controls for secure identity lifecycle management. See Snowflake's CREATE USER reference and user management documentation.

QUESTION NO: 2

Which database objects can be shared with the Snowflake secure data sharing feature? (Choose two.)

A. Files

B. External tables

C. Secure User-Defined Functions (UDFs)

D. Sequences

E. Streams

ANSWER: B C

Explanation:

Snowflake Secure Data Sharing supports sharing both external tables and secure user-defined functions (UDFs). An external table is a supported shareable object, allowing a provider to expose data that is stored externally while making its metadata and query access available to consumers through the share. Consumers can query the shared external table without receiving copies of the underlying data, consistent with Snowflake's no-copy sharing model.

Secure UDFs are also supported in shares. The `SECURE` designation is important because it protects the function definition from being exposed to consumers while still allowing them to invoke the function when it is included in a shared database. This enables providers to share governed calculation or transformation logic along with data, without revealing proprietary implementation details. The shared function must be a secure UDF and must be included in the share using the appropriate grant. Snowflake documents the supported database objects for shares and the requirements for securely sharing UDFs in its data-sharing documentation.

References: Snowflake Secure Data Sharing overview; CREATE SHARE reference.

QUESTION NO: 3

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 query at a time 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: B C

Explanation:

Each worksheet is a unique Snowflake session because Snowsight associates a separate session with each worksheet. Session-scoped context, such as the active role, selected warehouse, current database and schema, session variables, and temporary objects, belongs to that worksheet's session. This isolation allows a user to work in separate worksheets without relying on session state established in another worksheet. Snowflake documents that a worksheet session persists while the worksheet remains open and is distinct from sessions used by other worksheets.

Users are limited to running only one query at a time on a worksheet because a worksheet has one active execution context for query submission and result presentation. A worksheet can contain multiple SQL statements and can run them as a script, but simultaneous independent executions require using separate worksheets. Since each worksheet has its own session, separate worksheets are also the appropriate way to maintain separate query activity and session state while working concurrently. See Snowflake's Snowsight worksheets documentation and its guidance on running queries in worksheets.

QUESTION NO: 4

Which ACCOUNT_USAGE schema database role provides visibility into policy-related information?

A. USAGE_VIEWER

B. GOVERNANCE_VIEWER

C. OBJECT_VIEWER

D. SECURITY_VIEWER

ANSWER: B

Explanation:

GOVERNANCE_VIEWER is the ACCOUNT_USAGE database role intended to provide access to governance-focused metadata, including information associated with Snowflake policies. Snowflake supplies this role as part of its predefined shared database roles for the SNOWFLAKE database, allowing an administrator to grant narrowly scoped visibility to governance data without granting broad administrative privileges. This supports delegation of governance reporting and review responsibilities, such as inspecting policy assignments and policy-related account metadata, while preserving separation of duties. The role is granted to an account role, which then enables that role to query the applicable ACCOUNT_USAGE views exposed through the SNOWFLAKE database. Snowflake documents GOVERNANCE_VIEWER as the predefined database role for access to governance-related views and notes that the ACCOUNT_USAGE schema is the source for historical account-level metadata. For current details about the views and the privileges made available through this role, consult Snowflake's documentation on predefined database roles and the ACCOUNT_USAGE schema.

QUESTION NO: 5

The number of queries that a Virtual Warehouse can concurrently process is determined by (Choose two.):

A. The complexity of each query

B. The CONCURRENT_QUERY_LIMIT parameter set on the Snowflake account

C. The size of the data required for each query

D. The tool that is executing the query

ANSWER: A C

Explanation:

A virtual warehouse's effective query concurrency depends on the resource requirements of the workload it receives. **The complexity of each query** is a determining factor because queries with more joins, aggregations, sorts, large scans, or computationally intensive operations require more warehouse CPU and memory resources. As those resources are consumed by active work, fewer similarly demanding queries can execute efficiently at the same time.

The size of the data required for each query also affects concurrency. Queries that scan, join, aggregate, or move larger volumes of data generally require more processing, memory, and temporary storage resources. Therefore, a warehouse can sustain a different number of simultaneous queries depending on how much data each query must process. Snowflake automatically queues queries when the warehouse cannot immediately provide sufficient resources, and warehouse sizing or multi-cluster warehouses can be used to address workload demand and concurrency needs.

See Snowflake's documentation on virtual warehouse capabilities and workloads and multi-cluster warehouses for concurrency.

QUESTION NO: 6

Which of the following conditions must be met in order to return results from the results cache? (Select TWO).

A. The user has the appropriate privileges on the objects associated with the query

B. Micro-partitions have been reclustered since the query was last run

C. The new query is run using the same virtual warehouse as the previous query

D. The query includes a User Defined Function (UDF)

E. The query has been run within 24 hours of the previously-run query

ANSWER: A E

Explanation:

The user has the appropriate privileges on the objects associated with the query is required because Snowflake validates access when a persisted query result is reused. A cached result does not bypass Snowflake's authorization model: the user's active role must still have the necessary privileges on every object referenced by the query. This ensures that results previously generated by another session or user are not exposed to a role that is not authorized to access the underlying data.

The query has been run within 24 hours of the previously-run query is also required because persisted query results are initially retained for 24 hours after execution. When an identical eligible query is submitted during that period, Snowflake can return the stored result rather than re-executing the query. Each reuse resets the 24-hour retention period, up to Snowflake's maximum persisted-result retention limit of 31 days. Result-cache reuse also depends on query and data eligibility requirements, but the two listed conditions directly reflect the authorization and initial cache-lifetime requirements. See Snowflake's documentation on using persisted query results and persisted result retrieval optimization.

QUESTION NO: 7

A sales table FCT_SALES has 100 million records.

The following Query was executed

SELECT COUNT (1) FROM FCT__SALES;

How did Snowflake fulfill this query?

A. Query against the result set cache

B. Query against a virtual warehouse cache

C. Query against the most-recently created micro-partition

D. Query against metadata

ANSWER: D

Explanation:

Query against metadata is correct. For an unfiltered aggregate that counts every row in a table, Snowflake can satisfy the query using table metadata rather than scanning the table's micro-partitions. Snowflake maintains metadata about its storage, including row-count information, and the optimizer can use this information for supported queries such as an unqualified `COUNT(*)`. In this context, `COUNT(1)` counts one non-NULL constant for each row and is therefore equivalent to counting all rows, allowing the same metadata-based optimization. Consequently, the table having 100 million rows does not by itself require a full data scan or a virtual warehouse cache lookup. This optimization is particularly valuable for large tables because it can return the count efficiently while avoiding unnecessary compute work. Snowflake documents that metadata-based optimizations can be used for queries including `SELECT COUNT(*) FROM <table>`, subject to applicable query and table conditions. See Snowflake's [Using persisted query results] documentation for the distinction between result reuse and query execution, and its [Micro-partitions and data clustering] documentation for how Snowflake maintains table storage metadata.

QUESTION NO: 8

What is the most granular object that the Time Travel retention period can be defined on?

A. Account

B. Database

C. Schema

D. Table

ANSWER: D

Explanation:

Table is the most granular object on which a Time Travel data retention period can be configured. Snowflake uses the `DATA_RETENTION_TIME_IN_DAYS` parameter to specify how long historical data is retained and available for Time Travel operations, such as querying data at an earlier point in time or restoring an object. This parameter can be set at the account, database, schema, and individual table levels. Because a table is the lowest-level object among these supported scopes, it provides the most precise control over the retention period.

Parameter inheritance enables broader defaults to be established at the account, database, or schema level, while a table-level setting can tailor retention to the business, recovery, compliance, and storage requirements of that particular table. For example, a critical transactional table can have a different retention setting from other tables in the same schema. Snowflake documents that the parameter is configurable for accounts, databases, schemas, and tables, with more specific object-level values taking precedence where applicable. See the [Snowflake Time Travel documentation] and the [DATA_RETENTION_TIME_IN_DAYS parameter reference].

QUESTION NO: 9

Which of the following Snowflake features provide continuous data protection automatically? (Select TWO).

A. Internal stages

B. Incremental backups

C. Time Travel

D. Zero-copy clones

E. Fail-safe

ANSWER: C E

Explanation:

Time Travel and **Fail-safe** are Snowflake's automatic, continuous data-protection capabilities. Time Travel preserves historical versions of data after changes, including updates, deletes, and accidental object drops, for the configured data-retention period. Users can query prior data states and, when applicable, restore dropped objects during that retention window. This provides self-service recovery for operational mistakes without requiring customers to run or manage traditional backups.

After the Time Travel retention period ends, Snowflake automatically moves eligible permanent-data recovery protection into Fail-safe. Fail-safe provides an additional recovery window intended for disaster-recovery situations. Recovery during this period is performed by Snowflake rather than directly by users, which reflects its role as a final layer of platform-managed protection. Together, these capabilities protect data continuously across successive recovery windows: Time Travel supports accessible historical recovery first, and Fail-safe supplies an additional managed safeguard afterward. Snowflake documents these features as part of its built-in data protection model, rather than requiring separately scheduled incremental backup operations. See Snowflake's Time Travel documentation and Continuous Data Protection documentation.

QUESTION NO: 10

What is the recommended Snowflake data type to store semi-structured data like JSON?

A. VARCHAR

B. RAW

C. LOB

D. VARIANT

ANSWER: D

Explanation:

VARIANT is the recommended Snowflake data type for storing semi-structured data such as JSON. VARIANT is designed to hold hierarchical and nested structures, including JSON objects, arrays, and scalar values, while preserving their native structure rather than requiring the data to be flattened into relational columns before storage. Snowflake also automatically detects the data type of individual values within a VARIANT value where possible, enabling efficient use of JSON fields in SQL expressions.

After loading JSON into a VARIANT column, users can query nested elements with Snowflake's semi-structured data traversal syntax, such as colon and bracket notation, and can cast extracted values to appropriate SQL types when needed. This supports flexible ingestion as schemas evolve while retaining the ability to analyze nested data directly. VARIANT is part of Snowflake's semi-structured data support alongside ARRAY and OBJECT, but VARIANT is the general-purpose type specifically suited to storing an entire JSON document or values of differing structures in one column. See Snowflake's semi-structured data type documentation and guidance for querying semi-structured data.

QUESTION NO: 11

Which of the following describes external functions in Snowflake?

A. They are a type of User-defined Function (UDF).

B. They contain their own SQL code.

C. They call code that is stored inside of Snowflake.

D. They can return multiple rows for each row received

ANSWER: A

Explanation:

“They are a type of User-defined Function (UDF).” is correct because an external function is a Snowflake user-defined function whose implementation runs outside Snowflake. Rather than containing the implementation directly in the function definition, the function sends requests through a configured API integration to a remote service, typically reached through a cloud API gateway. That remote service executes the required logic and returns a result to Snowflake, allowing SQL statements to invoke externally hosted processing as though it were a function.

This design is useful when a query needs to integrate with services or code that cannot, or should not, run within Snowflake itself. Snowflake manages invocation of the function and the request/response exchange, while the external service owns the implementation. External functions are therefore part of Snowflake’s broader UDF capability, but are specifically distinguished by their ability to call an external endpoint. Snowflake documents that external functions are UDFs that call code executed outside Snowflake and describes the required API integration and proxy service architecture. See Introduction to external functions and CREATE EXTERNAL FUNCTION.

QUESTION NO: 12

What parameter controls if the Virtual warehouse starts immediately after the CREATE WAREHOUSE statement?

Select one.

A. INITTIALLY_SUSPENDED = TRUE/FALSE

B. START_AFTCR_CREATE = TRUE/FALSE

C. START_TTIME = 60 // (seconds from now)

D. START.TIME = CURRENT.DATE()

ANSWER: A

Explanation:

The correct parameter is `INITIALLY_SUSPENDED = TRUE/FALSE`. In Snowflake, the `INITIALLY_SUSPENDED` property of `CREATE WAREHOUSE` determines the warehouse’s state immediately after it is created. Setting it to `FALSE` causes the virtual warehouse to start (resume) immediately, making compute resources available for queries as soon as creation completes. Setting it to `TRUE` creates the warehouse in a suspended state, so it does not consume compute credits until it is explicitly resumed or automatically resumed by a query when `AUTO_RESUME` is enabled. The default value is `FALSE`, meaning a newly created warehouse starts immediately unless the statement specifies otherwise. This setting is useful when infrastructure is provisioned in advance but compute should remain inactive until a workload is ready to run. Snowflake documents `INITIALLY_SUSPENDED` as a warehouse property in the `CREATE WAREHOUSE` syntax and describes its effect on the initial warehouse state. See the CREATE WAREHOUSE reference and the virtual warehouses overview.

QUESTION NO: 13

How does Snowflake handle the data retention period for a table if a stream has not been consumed?

A. The data retention period is reduced to a minimum of 14 days.

B. The data retention period is permanently extended for the table.

C. The data retention period is temporarily extended to the stream ' s offset.

D. The data retention period is not affected by the stream consumption.

ANSWER: C

Explanation:

The data retention period is temporarily extended to the stream ' s offset. A Snowflake stream records the change data that occurred in its source object starting at the stream's current offset. If the normal Time Travel retention boundary would pass that offset before the stream is consumed, Snowflake automatically extends retention for the source table as needed to preserve the historical data required to calculate the stream's unconsumed change records. This safeguard prevents the stream from becoming stale solely because its offset would otherwise fall outside the available historical data.

The extension is tied to the stream's offset and is temporary rather than a permanent change to the table's configured `DATA_RETENTION_TIME_IN_DAYS` setting. Once the stream advances through consumption, Snowflake no longer needs to retain historical data solely for the prior offset. This behavior enables a stream consumer to read pending inserts, deletes, and updates accurately, even when the stream has not yet been consumed during the table's ordinary retention interval. Snowflake documents this as automatic retention extension for source objects that have unconsumed streams. See Introduction to streams and Understanding and using Time Travel.

QUESTION NO: 14

A user needs to create a materialized view in the schema MYDB.MYSCHEMA.

Which statements will provide this access?

A. GRANT ROLE MYROLE TO USER USER1;
CREATE MATERIALIZED VIEW ON SCHEMA MYDB.MYSCHEMA TO ROLE MYROLE;

B. GRANT ROLE MYROLE TO USER USER1;
CREATE MATERIALIZED VIEW ON SCHEMA MYDB.MYSCHEMA TO USER USER1;

C. GRANT ROLE MYROLE TO USER USER1;
CREATE MATERIALIZED VIEW ON SCHEMA MYDB.MYSCHEMA TO USER1;

D. GRANT ROLE MYROLE TO USER USER1;
CREATE MATERIALIZED VIEW ON SCHEMA MYDB.MYSCHEMA TO MYROLE;

E. GRANT ROLE MYROLE TO USER USER1;
GRANT USAGE ON DATABASE MYDB TO ROLE MYROLE;
GRANT USAGE ON SCHEMA MYDB.MYSCHEMA TO ROLE MYROLE;
GRANT CREATE MATERIALIZED VIEW ON SCHEMA MYDB.MYSCHEMA TO ROLE MYROLE;

ANSWER: E

Explanation:

The statements that grant `MYROLE` to `USER1`, grant `USAGE` on both `MYDB` and `MYDB.MYSCHEMA`, and grant `CREATE MATERIALIZED VIEW` on the schema to `MYROLE` provide the required access. Snowflake access control is role-based: privileges are granted to roles, and users obtain those privileges when using a role granted to them. Therefore, the materialized-view creation privilege must be granted with the valid `GRANT ... TO ROLE` syntax.

Creating an object in a schema also requires the role to be able to access the containing database and schema. `USAGE` on the database allows the role to resolve and access the database, while `USAGE` on the schema allows access to that schema. The schema-level `CREATE MATERIALIZED VIEW` privilege then authorizes creation of a materialized view there. Once `USER1` activates `MYROLE`, these combined grants allow the requested operation. Snowflake documents the privileges required for materialized-view creation and the role-based grant model in its schema privileges reference and GRANT syntax reference.

QUESTION NO: 15

Which user object property requires contacting Snowflake Support in order to set a value for it?

A. DISABLED

B. MINS TO BYPASS MFA

C. MINS TO BYPASS NETWORK POLICY

D. MINS TO UNLOCK

ANSWER: B

Explanation:

`MINS TO BYPASS MFA` is the correct property. In Snowflake SQL syntax, this user-object property is written as `MINS_TO_BYPASS_MFA` and specifies a temporary period, measured in minutes, during which a user may sign in without being prompted for multi-factor authentication. This setting is intended for exceptional access-recovery scenarios, such as when a user cannot complete their normal MFA challenge. Because bypassing MFA reduces an important authentication safeguard, Snowflake restricts configuration of this value: an account administrator must contact Snowflake Support to have the capability enabled before assigning a value for a user. Once configured, the bypass is temporary rather than a permanent removal of MFA requirements, helping organizations recover access while retaining their broader authentication controls. Snowflake documents this requirement directly in the user-management SQL reference under the `MINS_TO_BYPASS_MFA` property. See the CREATE USER reference and Snowflake's multi-factor authentication documentation for the property's purpose and MFA administration context.

QUESTION NO: 16

Which Snowflake function returns the name of the role currently active in the session?

A. CURRENT_USER()

B. CURRENT_ROLE()

C. CURRENT_DATABASE()

D. CURRENT_SCHEMA()

ANSWER: B

Explanation:

CURRENT_ROLE() is correct. The `CURRENT_ROLE` function returns the name of the role that is currently active for the session. Snowflake evaluates privileges for SQL statements using the active role, making this function useful for validating session context, troubleshooting access issues, and implementing role-aware logic in views, policies, procedures, and queries.

`CURRENT_USER()` returns the current user, while `CURRENT_DATABASE()` and `CURRENT_SCHEMA()` return the current database and schema context. These functions do not identify the active role. See the CURRENT_ROLE documentation.

QUESTION NO: 17

What are the three layers that make up Snowflake's architecture? (Choose three.)

A. Compute

B. Tri-Secret Secure

C. Storage

D. Cloud Services

ANSWER: A C D

Explanation:

Snowflake uses a multi-cluster shared data architecture that separates its core functions into the Storage, Compute, and Cloud Services layers. The Storage layer persists data in cloud storage and automatically manages how data is organized, compressed, encrypted, and optimized for Snowflake use. Users do not need to manage files, storage layouts, or indexes directly.

The Compute layer consists of virtual warehouses. A virtual warehouse provides the compute resources used to execute queries, load data, and perform other data-processing operations. Because compute is independent from storage, warehouses can scale independently and multiple warehouses can access the same stored data concurrently without competing for a single compute cluster.

The Cloud Services layer coordinates Snowflake activities. It includes services such as authentication, access control, metadata management, query parsing and optimization, transaction management, and infrastructure management. This separation is fundamental to Snowflake's elastic scaling and workload isolation capabilities. See Snowflake's Key Concepts and Architecture documentation and its Virtual Warehouses documentation for further details.

QUESTION NO: 18

A tabular User-Defined Function (UDF) is defined by specifying a return clause that contains which keyword?

A. ROW_NUMBER

B. TABLE

C. TABULAR

D. VALUES

ANSWER: B

Explanation:

The correct keyword is **TABLE**. In Snowflake, a tabular UDF—also called a user-defined table function (UDTF)—returns a relation rather than one scalar value. Its definition declares the result schema in the return clause using the `RETURNS TABLE (...)` syntax. The column names and data types inside the parentheses describe the rows that the function produces. For example, a SQL UDTF can be created with `RETURNS TABLE (col1 VARCHAR, col2 NUMBER)` and queried as a table source with `SELECT * FROM TABLE(function_name(...))`. This TABLE declaration is what establishes that the function returns a tabular result set suitable for use in a query's FROM clause. Snowflake documents the syntax and behavior for SQL table functions at SQL user-defined table functions. The general CREATE FUNCTION syntax likewise specifies `RETURNS TABLE` for functions that return tabular data.

QUESTION NO: 19

When should a stored procedure be created with caller ' s rights?

A. When the caller needs to be prevented from viewing the source code of the stored procedure

B. When the caller needs to run a statement that could not execute outside of the stored procedure

C. When the stored procedure needs to run with the privileges of the role that called the stored procedure

D. When the stored procedure needs to operate on objects that the caller does not have privileges on

ANSWER: C

Explanation:

A stored procedure should be created with caller's rights when it must execute using the privileges held by the role of the user who invokes it. With caller's rights, Snowflake evaluates object access and supported session operations in the caller's security context. This is appropriate for procedures intended to act on data or objects that each caller is already authorized to access, such as a utility procedure that queries a caller-selected table within that caller's permitted scope. It ensures the procedure does not become a privilege-escalation path: invoking the procedure does not grant access beyond what the caller's active role could normally use. The procedure owner still owns the procedure object, but the caller's role determines the privileges available while the procedure runs. Snowflake also supports caller's rights procedures receiving the caller's current session context, subject to the documented restrictions, which makes this model suitable when behavior must reflect the invoking user's environment and permissions. Define this explicitly with `EXECUTE AS CALLER` when creating the procedure. For details on the caller's-rights execution model and its security implications, see Snowflake's stored procedure caller's and owner's rights documentation and the CREATE PROCEDURE reference.

QUESTION NO: 20

Which of the following connectors are available in the Downloads section of the Snowflake Web Interface (UI)? (Choose two.)

A. SnowSQL

B. ODBC

C. R

D. HIVE

ANSWER: A B

Explanation:

SnowSQL and **ODBC** are available from the Downloads area of the Snowflake web interface. SnowSQL is Snowflake's command-line client, intended for interactive querying, running SQL scripts, and administrative or data-loading tasks from a terminal. Snowflake provides platform-specific installation packages for the client, so making it available through the interface's downloads location supports users who need a supported local command-line tool.

The Snowflake ODBC driver is also distributed as a downloadable client component. It enables ODBC-compliant applications, reporting tools, and ETL products to establish connections to Snowflake using the standard ODBC interface. The driver is provided in operating-system-specific packages and is installed locally before an application can use a Snowflake connection. Snowflake's documentation describes obtaining and installing SnowSQL at SnowSQL installation and configuration and provides ODBC driver download and installation guidance at Downloading and installing the ODBC driver.

QUESTION NO: 21

Which data types are supported by Snowflake when using semi-structured data? (Choose two.)

A. VARIANT

B. VARRAY

C. STRUCT

D. ARRAY

E. QUEUE

ANSWER: A D

Explanation:

VARIANT and **ARRAY** are Snowflake-supported data types for storing and working with semi-structured data. VARIANT is a flexible, tagged universal type that can contain values from semi-structured formats such as JSON, Avro, ORC, Parquet, or XML. It preserves the structure and native types of the loaded data, enabling SQL path expressions to access nested elements without requiring a fully predefined relational schema.

ARRAY stores an ordered collection of values. In semi-structured workloads, arrays commonly represent JSON lists and can contain heterogeneous values, including nested arrays, objects, and VARIANT-compatible scalar values. Snowflake provides functions such as `FLATTEN` to turn array elements into rows for querying and transformation. Snowflake identifies VARIANT, ARRAY, and OBJECT as its semi-structured data types; therefore, the two applicable choices supplied are VARIANT and ARRAY. See Snowflake's semi-structured data type reference and its documentation on FLATTEN for details on querying nested arrays and other semi-structured values.

QUESTION NO: 22

What MINIMUM privilege is required on the external stage for any role in the GET REST API to access unstructured data files using a file URL?

A. READ

B. OWNERSHIP

C. USAGK

D. WRTTF

ANSWER: A

Explanation:

READ is the minimum required privilege on an external stage because it authorizes a role to access files stored in that stage. For unstructured data, Snowflake uses stage-based access controls to govern whether a role can retrieve file content. When a file URL is used with the GET REST API, the role must have permission to read the referenced staged file; granting READ on the external stage supplies that authorization without requiring ownership of the stage. This follows Snowflake's privilege model for stages, in which READ provides access to files in an external stage. In practice, the role also needs the applicable access path to the stage, such as USAGE privileges on the containing database and schema, but those are separate object privileges. Since the question asks specifically for the minimum privilege *on the external stage*, READ is the required privilege. Snowflake documents the semantics of stage privileges, including READ, in its access-control privilege reference: Snowflake stage privileges. Further details on accessing unstructured data stored in stages are available in the Snowflake unstructured data documentation.

QUESTION NO: 23

Two users share a virtual warehouse named wh dev 01. When one of the users loads data, the other one experiences performance issues while querying data.

How does Snowflake recommend resolving this issue?

A. Scale up the existing warehouse.

B. Create separate warehouses for each user.

C. Create separate warehouses for each workload.

D. Stop loading and querying data at the same time.

ANSWER: C

Explanation:

Snowflake recommends **creating separate warehouses for each workload**. A virtual warehouse supplies the compute resources used to execute SQL statements, including data-loading operations and queries. When loading and querying share the same warehouse, they compete for the warehouse's finite compute capacity. This contention can increase queuing and reduce the resources available for interactive query execution, causing the querying user to experience slower performance. Separating the load workload from the query workload provides compute isolation: each warehouse has independent resources, can be sized appropriately for its own demand, and can use its own auto-suspend and auto-resume settings. This approach also enables independent cost governance and scaling decisions, without allowing a resource-intensive ingestion process to affect analytical users. Snowflake's workload-management guidance specifically recommends using separate warehouses to isolate workloads with different performance and concurrency requirements. The warehouse assigned to loading can be optimized for ingestion throughput, while the warehouse used for querying can be optimized for responsive analysis. See Snowflake's virtual warehouse overview and its warehouse considerations and workload guidance.

QUESTION NO: 24

Which commands are restricted in owner's rights stored procedures? (Select TWO).

A. SHOW

B. MERGE

C. INSERT

D. DELETE

E. DESCRIBE

ANSWER: A E

Explanation:

In an owner's rights stored procedure, Snowflake executes SQL using the privileges of the procedure owner rather than those of the caller. Snowflake places restrictions on commands that expose object-definition and metadata details, including **SHOW** and **DESCRIBE**. These commands can enumerate account, database, schema, or object metadata and can disclose details beyond what the invoking role would ordinarily be allowed to inspect. Restricting them helps preserve the security boundary between the procedure owner's elevated privileges and the caller's permissions. This is especially important because owner's rights procedures are commonly used to provide controlled access to privileged operations without directly granting all underlying object privileges to every caller. Snowflake documents the limitations and restricted commands for owner's rights procedures in its stored-procedure rights guidance. For the authoritative list and applicable behavior, see Understanding caller's rights and owner's rights stored procedures and Snowflake SHOW command reference.

QUESTION NO: 25

Which of the following connectors allow Multi-Factor Authentication (MFA) authorization when connecting?

(Choose all that apply.)

A. JDBC

B. SnowSQL

C. Snowflake Web Interface (UI)

D. ODBC

E. Python

ANSWER: A B C D E

Explanation:

JDBC, SnowSQL, the Snowflake web interface, ODBC, and the Python connector can all be used with Snowflake MFA-enabled authentication. The Snowsight web interface presents the normal interactive sign-in flow, including the MFA challenge required by the user's configured authentication policy. For client tools and connectors, Snowflake supports browser-based authentication through the `externalbrowser` authenticator. This opens a browser so that the user can complete their organization's interactive authentication process, including MFA, before the client connection is established. JDBC and ODBC expose this mechanism through their connection parameters; SnowSQL supports it through its connection configuration or command-line parameters; and the Python Connector supports it through the `authenticator` connection parameter. Snowflake also documents MFA token caching for supported client applications and drivers, which can reduce repeated MFA prompts after a successful interactive authentication while retaining the configured security controls. In practice, the exact sign-in experience depends on whether Snowflake-native MFA or federated single sign-on is configured, but each listed interface supports an MFA-capable connection flow. See Snowflake's Multi-factor authentication documentation and its Python Connector connection documentation.

QUESTION NO: 26

A Snowflake user has been granted the create data EXCHANGE listing privilege with their role.

Which tasks can this user now perform on the Data Exchange? (Select TWO).

A. Rename listings.

B. Delete provider profiles.

C. Modify listings properties.

D. Modify incoming listing access requests.

E. Submit listings for approval/publishing.

ANSWER: C E

Explanation:

The `CREATE DATA EXCHANGE LISTING` privilege authorizes a role to work with provider listings in a Snowflake Data Exchange. In particular, the role can modify listing properties, such as the listing's descriptive metadata and other configuration details that define how the shared data product is presented to consumers. This supports the iterative process of preparing a listing before it is made available.

The role can also submit listings for approval/publishing. Publication is the step that makes a prepared listing available through the applicable exchange workflow. Depending on the exchange and its governance configuration, a listing may be submitted for approval or published according to the provider's permitted publication process. These capabilities align with Snowflake's provider listing workflow: create and configure a listing, then publish it or submit it for review. Snowflake's current listing documentation describes creating, editing, and publishing listings, while its access-control documentation defines the privileges used to authorize object-management activities. See Create and publish listings and Access control privileges.

QUESTION NO: 27

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 supported mechanism for sharing data that resides in multiple databases. A Snowflake share is associated with a single database, so a provider can create a secure view in the shared database that presents a consolidated result while referencing eligible objects located in other databases in the same provider account. The provider then grants the share access to that secure view. Secure views are specifically designed for controlled data sharing because Snowflake does not expose the view definition or underlying query logic to consumers. This allows the provider to publish only the intended columns, rows, joins, filters, and transformations while keeping the underlying database design and source-object details protected. When a secure view references objects outside the database containing the shared view, the provider must also configure the required reference permissions so that Snowflake can resolve those underlying objects when consumers query the view. This pattern gives consumers a single shareable object while allowing the provider to combine governed data from distinct databases. Snowflake documents secure views as a means to share data from multiple databases and describes the required privileges for sharing views with cross-database references. See Snowflake: Share data from multiple databases and Snowflake: Working with secure views.

QUESTION NO: 28

Which of the following statements are true of resource monitors? (Choose three.)

A. A resource monitor can notify users when a credit-consumption threshold is reached

B. A resource monitor can suspend assigned warehouses after a threshold is reached

C. A resource monitor can be assigned to more than one warehouse

D. A resource monitor tracks storage consumption for tables and stages

E. A resource monitor prevents all cloud services credit consumption

ANSWER: A B C

Explanation:

Resource monitors help control virtual warehouse credit consumption. A resource monitor can track the credit usage of one or more assigned warehouses against a specified credit quota. At defined consumption thresholds, the monitor can notify administrators or other specified recipients. A monitor can also suspend assigned warehouses, either after currently executing statements finish or immediately, depending on the configured action.

Resource monitors track warehouse compute credits; they do not monitor storage consumption. Storage usage is monitored through Snowflake billing and usage views rather than through warehouse resource monitors. See the resource monitors documentation and the CREATE RESOURCE MONITOR reference.

QUESTION NO: 29

True or False: Pipes can be suspended and resumed.

A. True

B. False

ANSWER: A

Explanation:

True is correct. A Snowpipe pipe can be paused (suspended) and later resumed by using the `ALTER PIPE` command and setting the `PIPE_EXECUTION_PAUSED` property. Setting this property to `TRUE` pauses execution for the pipe, preventing Snowpipe from loading newly discovered files while the pipe remains defined. Setting it back to `FALSE` resumes pipe execution. This is useful for planned maintenance, controlling ingestion during data-quality investigations, or temporarily stopping automated loading without dropping and recreating the pipe definition or its associated notification integration. For example, an authorized user can run `ALTER PIPE my_pipe SET PIPE_EXECUTION_PAUSED = TRUE` to suspend it and later run `ALTER PIPE my_pipe SET PIPE_EXECUTION_PAUSED = FALSE` to resume it. The ability to manage this state is documented as a supported pipe property in Snowflake's ALTER PIPE syntax. Resuming the pipe restores Snowpipe's normal processing behavior for eligible staged files, subject to Snowpipe's file-loading and notification behavior. See the Snowflake ALTER PIPE reference and the Snowpipe overview.

QUESTION NO: 30

What is the minimum Snowflake edition that provides multi-cluster warehouses and up to 90 days of Time Travel?

A. Standard

B. Premier

C. Enterprise

D. Business Critical Edition

ANSWER: C

Explanation:

Enterprise is the minimum Snowflake edition that includes both requested capabilities. Enterprise Edition enables multi-cluster warehouses, which can automatically add or remove compute clusters according to workload demand. This supports greater concurrency while preserving the warehouse configuration and workload behavior expected by users. Snowflake documents multi-cluster warehouse capability as an Enterprise Edition feature (and it remains available in higher editions). Enterprise Edition also provides a configurable Time Travel retention period of up to 90 days for eligible permanent objects. Time Travel allows users to access, query, clone, and restore historical data within the configured retention period, helping with recovery from unintended data changes and supporting historical analysis. The actual retention value is configured through the `DATA_RETENTION_TIME_IN_DAYS` parameter and can be set anywhere within the limits of the account's edition and object type. Therefore, because the question asks for the lowest edition that combines multi-cluster warehouse support with up to 90 days of Time Travel, Enterprise is the correct selection. See Snowflake's Multi-cluster warehouses documentation and its Time Travel documentation.

QUESTION NO: 31

A user needs to create a materialized view in the schema MYDB.MYSCHEMA.

Which statements will provide this access?

A. GRANT ROLE MYROLE TO USER USER1;CREATE MATERIALIZED VIEW ON SCHEMA MYDB.MYSCHEMA TO ROLE MYROLE;

B. GRANT ROLE MYROLE TO USER USER1;CREATE MATERIALIZED VIEW ON SCHEMA MYDB.MYSCHEMA TO USER USER1;

C. GRANT ROLE MYROLE TO USER USER1;CREATE MATERIALIZED VIEW ON SCHEMA MYDB.MYSCHEMA TO USER1;

D. GRANT ROLE MYROLE TO USER USER1;CREATE MATERIALIZED VIEW ON SCHEMA MYDB.MYSCHEMA TO MYROLE;

E. GRANT ROLE MYROLE TO USER USER1;GRANT CREATE MATERIALIZED VIEW ON SCHEMA MYDB.MYSCHEMA TO ROLE MYROLE;

ANSWER: E

Explanation:

`GRANT ROLE MYROLE TO USER USER1;` followed by `GRANT CREATE MATERIALIZED VIEW ON SCHEMA MYDB.MYSCHEMA TO ROLE MYROLE;` is correct because Snowflake access privileges are granted to roles, and users exercise those privileges through roles granted to them. The first statement makes `MYROLE` available to `USER1`. The second statement grants the schema-level `CREATE MATERIALIZED VIEW` privilege, which is the specific privilege that authorizes a role to create materialized views in `MYDB.MYSCHEMA`. The `GRANT` keyword is required when assigning a privilege; `CREATE MATERIALIZED VIEW` is the privilege name, not a standalone grant command.

In a complete implementation, the active role also needs the relevant `USAGE` privileges on the database and schema, as well as `SELECT` privileges on the base table or view used by the materialized view definition. Those are separate access requirements from the schema-level create privilege addressed here. Snowflake documents the valid privilege-grant syntax and schema privileges in its GRANT <privileges> documentation; materialized-view privilege requirements are described in the Materialized views documentation.

QUESTION NO: 32

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. VARCHAK

ANSWER: A D

Explanation:

ARRAY and **VARIANT** are Snowflake semi-structured data types and can be defined as columns in a table that stores semi-structured data. VARIANT is a tagged universal type that can hold values with differing native structures and types, including data represented in JSON, Avro, ORC, Parquet, and XML formats. This enables Snowflake to load and query hierarchical data while retaining its structure rather than requiring all fields to be decomposed into conventional relational columns before storage.

ARRAY stores an ordered collection of values. Its elements can include values of varying types, including nested arrays, objects, and VARIANT values, making it appropriate for semi-structured records containing lists. Snowflake also provides OBJECT as another semi-structured type for key-value-pair structures, but the requested two selections from the provided choices are ARRAY and VARIANT. Snowflake documentation categorizes ARRAY, OBJECT, and VARIANT as its dedicated semi-structured data types and describes using them to store and operate on semi-structured values directly. See Snowflake semi-structured data types and Introduction to semi-structured data.

QUESTION NO: 33

Which data types optimally store semi-structured data? (Select TWO).

A. ARRAY

B. CHARACTER

C. STRING

D. VARCHAR

E. VARIANT

ANSWER: A E

Explanation:

ARRAY and **VARIANT** are native Snowflake semi-structured data types. VARIANT is the general-purpose type for storing hierarchical or heterogeneous data in a single column. It can hold values ingested from formats including JSON, Avro, ORC, Parquet, and XML, while preserving the structure so that nested elements can be queried with Snowflake's semi-structured-data path syntax and functions. This makes it particularly appropriate where records can vary in shape or contain nested objects and lists.

ARRAY is designed to store an ordered collection of values. It is useful when the semi-structured value being modeled is inherently a list, such as a JSON array of products, tags, events, or measurements. ARRAY values can contain elements of differing types, including nested semi-structured values, and Snowflake provides functions for constructing, accessing, transforming, and flattening array contents. Together, these types support efficient native storage and SQL querying of flexible, nested data rather than requiring the source document to be treated only as unstructured text. Snowflake documents VARIANT, OBJECT, and ARRAY as its semi-structured data types; for the two requested selections, ARRAY and VARIANT directly represent list-based and general semi-structured payloads. See Snowflake's semi-structured data type reference and introduction to querying semi-structured data.

QUESTION NO: 34

A table needs to be loaded. The input data is in JSON format and is a concatenation of multiple JSON documents. The file size is 3 GB. A warehouse size small is being used. The following COPY INTO

command was executed:

COPY INTO SAMPLE FROM @~/SAMPLE.JSON (TYPE=JSON)

The load failed with this error:

Max LOB size (16777216) exceeded, actual size of parsed column is 17894470.

How can this issue be resolved?

A. Compress the file and load the compressed file.

B. Split the file into multiple files in the recommended size range (100 MB - 250 MB).

C. Use a larger-sized warehouse.

D. Set STRIP_OUTER_ARRAY=TRUE in the COPY INTO command.

ANSWER: B

Explanation:

Split the file into multiple files in the recommended size range (100 MB - 250 MB) is correct because the error identifies that a parsed JSON value exceeded the 16 MB large-object limit in effect for this load. The relevant constraint is the size of the JSON data parsed into an individual column value or record, not the total 3 GB size of the staged file or the compute capacity of the warehouse. The concatenated JSON input should be divided at valid JSON document boundaries so that Snowflake can parse and load manageable records without producing an oversized parsed value. As a loading best practice, Snowflake recommends staging data in multiple reasonably sized files—typically about 100 MB to 250 MB compressed—to support efficient parallel loading. While file sizing alone must be implemented without splitting an individual JSON document, dividing this concatenated input into valid, smaller JSON files resolves the oversized parsed-column condition when the documents are distributed appropriately. Snowflake documents file-sizing guidance for bulk loading and describes JSON loading behavior in its loading documentation: Preparing your data files and Loading semi-structured data.

QUESTION NO: 35

What is a responsibility of Snowflake's virtual warehouses?

A. Infrastructure management

B. Metadata management

C. Query execution

D. Query parsing and optimization

E. Permanent storage of micro-partitions

ANSWER: C

Explanation:

Snowflake virtual warehouses are the platform's compute layer, and their core responsibility is query execution. A virtual warehouse is a cluster of compute resources that Snowflake uses to execute SQL statements, including loading data, unloading data, and performing queries. Warehouses can be independently started, stopped, resized, and configured as multi-cluster warehouses to supply the compute capacity needed for a workload. This separation of compute from storage is a central element of Snowflake's architecture: persistent data is held in the storage layer, while a warehouse provides the CPU and memory resources used when work must be performed against that data. Because warehouses are independent, separate workloads can use separate warehouses, helping avoid resource contention and allowing compute usage to scale according to operational needs. Snowflake manages the underlying infrastructure on the customer's behalf, while its cloud services layer handles functions such as authentication, metadata management, and query parsing and optimization. For the official definition and operational details, see Snowflake's Virtual warehouses overview and its key concepts documentation.

QUESTION NO: 36

How are serverless features billed?

A. Per second multiplied by an automatic sizing for the job

B. Per minute multiplied by an automatic sizing for the job, with a minimum of one minute

C. Per second multiplied by the size, as determined by the SERVERLESS_FEATURES_SIZE account parameter

D. Serverless features are not billed, unless the total cost for the month exceeds 10% of the warehouse credits, on the account

ANSWER: A

Explanation:

Serverless features are billed **per second**one-minute minimumSnowflake serverless compute documentation and Understanding compute cost.

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 files in these formats are loaded, Snowflake can store each record in a `VARIANT` column, preserving the hierarchical or nested structure so that elements can subsequently be queried with semi-structured data functions and path notation. JSON is a text-based format commonly used for nested objects and arrays. Avro is a row-oriented binary serialization format whose schema and records can be ingested by Snowflake. Parquet is a columnar binary format that Snowflake supports for loading semi-structured records, including nested data. These formats are specified through a file format object or inline file-format settings used with `COPY INTO`. Snowflake's supported semi-structured formats also include ORC and XML, but those formats are not among the available choices. Official documentation lists JSON, Avro, ORC, Parquet, and XML as supported semi-structured data formats and describes loading them into `VARIANT` columns. See Snowflake's Introduction to semi-structured data and Loading semi-structured data documentation for supported formats and loading guidance.

QUESTION NO: 38

Which is true of Snowflake network policies? A Snowflake network policy: (Choose two.)

A. Is available to all Snowflake Editions

B. Is only available to customers with Business Critical Edition

C. Restricts or enables access to specific IP addresses

D. Is activated using an “ALTER DATABASE” command

ANSWER: A C

Explanation:

Snowflake network policies are available across Snowflake editions and provide a way to control the network locations from which users can connect. A policy contains lists of IPv4 address ranges in CIDR notation: an allowed IP list, a blocked IP list, or both. This lets an organization limit access to known corporate egress addresses, VPN ranges, or other approved network endpoints. When both lists are present, Snowflake evaluates the blocked list first, allowing administrators to explicitly deny an address or range even where it otherwise overlaps with an allowed range.

Network policies can be assigned at the account level to govern access for the account, or associated with individual users to apply user-specific access controls. They are therefore an important defense-in-depth control for reducing the set of source networks permitted to authenticate to Snowflake. Snowflake documents that network policies can restrict access to specific IP addresses and that the feature is not limited to a particular higher Snowflake edition. For implementation details, including IP-list behavior and account or user assignment, see the Snowflake network policies documentation and the CREATE NETWORK POLICY reference.

QUESTION NO: 39

What type of query benefits the MOST from search optimization?

A. A query that uses only disjunction (i.e., OR) predicates

B. A query that includes analytical expressions

C. A query that uses equality predicates or predicates that use IN

D. A query that filters on semi-structured data types

ANSWER: C

Explanation:

A query that uses equality predicates or predicates that use IN benefits the most from Snowflake's search optimization service. Search optimization is designed to accelerate highly selective lookup operations by maintaining optimized search access paths that help Snowflake identify the small set of micro-partitions likely to contain matching values. For example, filters such as `WHERE customer_id = 12345` or `WHERE order_id IN (101, 205, 309)` can avoid scanning a large proportion of a table when the predicate matches relatively few rows. This is particularly valuable for very large tables where a conventional scan would otherwise read many micro-partitions to locate a limited number of records. The service can improve performance for equality and `IN` predicates on supported data types and expressions, including joins where the build-side input contains a small number of distinct values. The greatest benefit occurs when the predicate is selective: as the number of values or matching rows increases, more partitions may need to be accessed and the relative advantage declines. Snowflake documents equality and `IN` predicates as core query patterns that can benefit from search optimization. See [Search Optimization Service] and [Queries That Benefit from Search Optimization].

QUESTION NO: 40

When should a user consider disabling auto-suspend for a virtual warehouse? (Select TWO).

A. When users will be using compute at different times throughout a 24/7 period

B. When managing a steady workload

C. When the compute must be available with no delay or lag time

D. When the user does not want to have to manually turn on the warehouse each time it is needed

E. When the warehouse is shared across different teams

ANSWER: B C

Explanation:

"When managing a steady workload" is correct because a warehouse serving continuous or highly regular activity may suspend and resume frequently if auto-suspend is configured too aggressively. In that circumstance, keeping the warehouse running can avoid repeated provisioning cycles and provide more consistent compute availability. Snowflake recommends selecting an auto-suspend setting based on workload characteristics, balancing the cost of idle compute against the operational need for ready capacity.

"When the compute must be available with no delay or lag time" is also correct. Although auto-resume starts a suspended warehouse automatically when a query arrives, resuming requires time for Snowflake to provision compute resources. Workloads with strict responsiveness requirements, such as latency-sensitive applications or processes that must begin immediately, can justify disabling auto-suspend so that the warehouse remains available. This choice should be made deliberately because a running warehouse continues to consume credits even when it is idle. Snowflake documentation describes how auto-suspend and auto-resume control warehouse availability and credit usage: https://docs.snowflake.com/en/user-guide/warehouses-tasks. Additional warehouse management guidance is available at https://docs.snowflake.com/en/user-guide/warehouses-overview.

QUESTION NO: 41

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 managed under one organization name; it is not limited to one cloud platform or one Snowflake region. This lets an organization support deployment requirements such as operating workloads near users, meeting regional dataresidency requirements, or using different supported cloud providers. For example, the same organization may have one account deployed on AWS in one region and another deployed on Microsoft Azure or Google Cloud in a different region. Each account remains an independent Snowflake deployment with its own cloud platform and region, while organization-level capabilities provide a common organizational context across those accounts. Snowflake documents that an organization can include accounts in different regions and on different cloud platforms, subject to the organization's enabled regions and account-creation permissions. See the Snowflake documentation on organizations and account identifiers, regions, and cloud platforms for details.

QUESTION NO: 42

Which Snowflake feature allows a user to track sensitive data for compliance, discovery, protection, and resource usage?

A. Tags

B. Comments

C. Internal tokenization

D. Row access policies

ANSWER: A

Explanation:

Tags are correct because Snowflake tags are schema-level objects used to assign descriptive metadata to supported objects, including databases, schemas, tables, columns, warehouses, users, and roles. They enable an organization to classify and consistently track sensitive data, such as personally identifiable information, financial data, or regulated records. This classification supports governance and compliance workflows by making data easier to discover and report on across the account. Tags can also be used as the basis for tag-based masking policies, allowing protection rules to be applied consistently to columns that carry a particular sensitivity classification. In addition, Snowflake supports object tagging for cost attribution and resource-usage analysis, such as associating warehouses or other objects with a cost center, business unit, or project. Account usage views and functions can then be used to inspect tag assignments and analyze tagged-object usage. This combination of metadata classification, governance-policy integration, discovery, and cost/resource attribution is the intended purpose of Snowflake tags. See the Snowflake documentation on object tagging and cost attribution.

QUESTION NO: 43

What is the recommended way to change the existing file format type in my format from CSV to JSON?

A. ALTER FILE FORMAT my_format SET TYPE=JSON;

B. ALTER FILE FORMAT my format SWAP TYPE WITH JSON;

C. CREATE OR REPLACE FILE FORMAT my format TYPE-JSON;

D. REPLACE FILE FORMAT my format TYPE-JSON;

ANSWER: A

Explanation:

`ALTER FILE FORMAT my_format SET TYPE=JSON;` is the appropriate command because Snowflake supports modifying an existing named file format with `ALTER FILE FORMAT`. The `SET` clause updates file-format properties, including the `TYPE` property. Setting `TYPE = JSON` changes the format definition from its prior CSV configuration to JSON, allowing subsequent commands that reference `my_format`, such as staging and loading operations, to apply JSON parsing rules. The file format object retains its name and any properties that are not explicitly changed by the command, which makes this a direct and manageable way to revise the existing definition. The identifier `my_format` is also a valid unquoted Snowflake identifier because it uses letters and an underscore. Snowflake documents the `ALTER FILE FORMAT` syntax and lists supported values for the `TYPE` parameter, including both `CSV` and `JSON`. Before making the change in a shared environment, review dependent load processes because they will begin using the JSON settings once the altered file format is referenced. See the Snowflake ALTER FILE FORMAT reference and the file format type documentation.

QUESTION NO: 44

Which Snowflake object does not consume any storage costs?

A. Secure view

B. Materialized view

C. Temporary table

D. Transient table

ANSWER: A

Explanation:

Secure view is correct because it is a logical view: Snowflake stores the view definition, including its SQL query and security properties, rather than storing a separate copy of the query result or underlying table data. When a user queries a secure view, Snowflake executes the view definition against the underlying data, subject to the view's security behavior. Consequently, the data storage charges remain associated with the underlying tables; the secure view itself does not create stored table data that incurs storage costs. Secure views are specifically intended to expose selected data while preventing users from seeing details that might reveal the underlying tables, columns, or view definition. This makes them appropriate for securely sharing a controlled representation of data without duplicating that data. Snowflake documentation describes views as saved queries that can be queried like tables, and identifies secure views as views with enhanced privacy protections. See Introduction to views and Working with secure views.

QUESTION NO: 45

The effects of query pruning can be observed by evaluating which statistics? (Select TWO).

A. Partitions scanned

B. Partitions total

C. Bytes scanned

D. Bytes read from result

E. Bytes written

ANSWER: A B

Explanation:

Query pruning is evaluated most directly by comparing **Partitions scanned** with **Partitions total**. Snowflake stores table data in micro-partitions and maintains metadata about the values within each micro-partition. When a query predicate allows Snowflake to eliminate micro-partitions whose metadata cannot satisfy the predicate, those partitions are pruned and do not need to be scanned. The total-partitions statistic establishes the number of micro-partitions available for the relevant table access, while the scanned-partitions statistic shows how many were actually read for the query. A relatively small scanned value compared with the total value is evidence that pruning is effectively reducing the data access required by the operation. Conversely, values that are close together indicate that the query had limited opportunity to eliminate micro-partitions. These statistics can be viewed in the Query Profile and are particularly useful when assessing predicate selectivity, clustering quality, and whether a table may benefit from reclustering or a different access pattern. Snowflake documents partition pruning as using micro-partition metadata to avoid scanning unnecessary data, and identifies partition-scanning information in query performance analysis. See Micro-partitions and data clustering and Using Query Profile.

QUESTION NO: 46

Which command sets the Virtual Warehouse for a session?

A. COPY WAREHOUSE FROM <>;

B. SET WAREHOUSE = <>;

C. USE WAREHOUSE <>;

D. USE VIRTUAL_WAREHOUSE <>;

ANSWER: C

Explanation:

USE WAREHOUSE <<warehouse name="">>; is the Snowflake SQL command used to set the active virtual warehouse for the current session. A virtual warehouse supplies the compute resources that execute queries and perform operations such as loading data. Once a warehouse is selected with the USE WAREHOUSE command, subsequent statements in that session run using that warehouse, provided the active role has the required USAGE privilege on it. The chosen warehouse remains the current warehouse until the session ends or another warehouse is selected. Snowflake also supports setting a default warehouse for a user, but the session-level SQL command for explicitly choosing or changing the active warehouse is USE WAREHOUSE. This command can be issued with an unquoted warehouse identifier or, when required by the identifier's naming rules, a quoted identifier. For example, USE WAREHOUSE analytics_wh; makes analytics_wh the current compute warehouse for the session. Snowflake documents this syntax in its USE WAREHOUSE command reference and describes warehouses as the compute resources used to execute queries in the Virtual warehouses overview.

QUESTION NO: 47

What types of data listings are available in the Snowflake Data Marketplace? (Choose two.)

A. Reader

B. Consumer

C. Vendor

D. Standard

E. Personalized

ANSWER: D E

Explanation:

Snowflake Data Marketplace listings can be published as **Standard** listings or **Personalized** listings. A Standard listing is intended for broad discovery and access through the Marketplace, enabling eligible Snowflake customers to discover the offered data product and request or obtain access according to the provider's sharing configuration. This approach supports distributing a data product to a broad audience without having to create a separate listing for every potential consumer.

A Personalized listing is designed for a particular consumer or a defined set of consumers. It lets a provider tailor distribution to the intended recipients, which is useful when access, content, commercial terms, or the relationship with the consumer requires an explicitly targeted offering. Both listing types support Snowflake's secure data-sharing model: consumers access shared data in Snowflake without the provider needing to copy or move the underlying data into each consumer environment.

These terms are associated with the Data Marketplace listing model used in SnowPro Core exam material. Snowflake's current Collaboration documentation further describes how providers create and manage listings and distribute them to intended audiences. See Snowflake Marketplace documentation and Creating listings documentation.

QUESTION NO: 48

What computer language can be selected when creating User-Defined Functions (UDFs) using the Snowpark API?

A. Swift

B. JavaScript

C. Python

D. SQL

ANSWER: C

Explanation:

Python is correct because Snowpark provides a Python API that developers can use to define and register Python user-defined functions for execution in Snowflake. A Python UDF is implemented with a Python handler function, and Snowflake executes that handler within its managed runtime environment. This lets developers apply familiar Python logic to data processing without moving data out of Snowflake. When creating the function, the implementation specifies Python as the language and identifies the handler that Snowflake should invoke. Snowpark also supplies Python DataFrame capabilities, allowing Python-based data transformations and custom logic to be developed in a style familiar to data engineers and data scientists, while computation remains governed and scalable in the Snowflake platform. Snowflake documents the process for creating Python UDFs, including defining the function, selecting the Python runtime, and configuring packages when dependencies are needed. The Snowpark API overview also identifies Python as a supported Snowpark language. See the Snowflake Python UDF introduction and the Snowpark documentation.

QUESTION NO: 49

When cloning a database, what is cloned with the database? (Choose two.)

A. Privileges on the database

B. Existing child objects within the database

C. Future child objects within the database

D. Privileges on the schemas within the database

E. Only schemas and tables within the database

ANSWER: B D

Explanation:

A database clone is a zero-copy, point-in-time copy of the source database. Therefore, **Existing child objects within the database** are included as they existed at the selected cloning point. This includes the database's schemas and the supported objects contained within them, such as tables, views, stages, sequences, streams, tasks, and file formats. The clone is independent for future DDL purposes, even though cloned data can initially share underlying storage through Snowflake's zero-copy cloning architecture.

Privileges on the schemas within the database are also included because schemas are child objects that are cloned as part of the database hierarchy, along with the grants associated with those cloned schema objects. This preserves the relevant schema-level access-control metadata for the cloned environment. The clone represents the source state at clone creation (or the requested Time Travel point), so it does not automatically acquire objects created later in the source database. Snowflake documents that cloning a database includes its child objects and explains the access-control behavior of cloned objects. See Snowflake Object Cloning and Snowflake Access Control Privileges.

QUESTION NO: 50

Which data formats are supported by Snowflake when unloading semi-structured data? (Select TWO).

A. Binary file in Avro

B. Binary file in Parquet

C. Comma-separated JSON

D. Newline Delimited JSON

E. Plain text file containing XML elements

ANSWER: B D

Explanation:

Snowflake supports unloading query results to staged files in both Parquet and JSON formats. The JSON unload format writes records as separate JSON documents, one per line, which is commonly called newline-delimited JSON (NDJSON). This makes it suitable for exporting semi-structured values such as VARIANT data while retaining JSON representations that downstream systems can process record by record.

Parquet is also a supported Snowflake unload format. It is a binary, columnar format designed for analytical data exchange and is widely supported across data platforms. When unloading to Parquet, Snowflake produces Parquet files that can efficiently carry query-result data for downstream analytics or storage workflows. Snowflake documents JSON and PARQUET as supported values for the `TYPE` setting in a file format used by `COPY INTO <location>`. For JSON output, Snowflake specifically notes that each JSON object is separated by a newline character, matching the NDJSON convention.

See Snowflake's COPY INTO <location> documentation and its CREATE FILE FORMAT reference for supported unload file-format types and JSON output behavior.

QUESTION NO: 51

Which of the following are true of multi-cluster Warehouses? Select all that apply below.

A. A multi-cluster Warehouse can add clusters automatically based on query activity

B. A multi-cluster Warehouse can automatically turn itself off after a period of inactivity

C. A multi-cluster Warehouse can scale down when query activity slows

D. A multi-cluster Warehouse can automatically turn itself on when a query is executed against it

ANSWER: A B C D

Explanation:

Multi-cluster warehouses provide additional compute clusters to handle concurrent query workloads while retaining the same warehouse definition and size for each cluster. When configured with a minimum and maximum cluster count greater than one, Snowflake can automatically start additional clusters as demand and queueing require, then remove unneeded clusters as demand decreases. This automatic scaling helps maintain query throughput during periods of high concurrency. Multi-cluster warehouses also use the standard warehouse lifecycle settings: auto-suspend stops the warehouse after its configured idle interval, and auto-resume starts it when a statement that requires the warehouse is submitted. These capabilities mean that a multi-cluster warehouse can add clusters automatically based on query activity, scale down when activity slows, suspend after inactivity, and resume automatically when a query executes against it. Administrators configure these behaviors through the warehouse properties, including minimum/maximum cluster count, scaling policy, AUTO_SUSPEND, and AUTO_RESUME. Snowflake documents multi-cluster scaling and its relationship to workload concurrency at Multi-cluster warehouses. The suspend and resume lifecycle behavior is documented at Starting and stopping warehouses.

QUESTION NO: 52

What are the benefits of the replication feature in Snowflake? (Select TWO).

A. Disaster recovery

B. Time Travel

C. Fail-safe

D. Database failover and fallback

E. Data security

ANSWER: A D

Explanation:

Disaster recovery is a core benefit of Snowflake replication. Replication creates and regularly refreshes a secondary copy of supported objects from a source account to a target account, including across regions and, where supported, cloud platforms. This provides an alternate environment that an organization can use to recover business operations when a regional or account-level disruption affects the primary environment. Replication is therefore a key component of a business-continuity and disaster-recovery design.

Database failover and fallback is also enabled by replication. Snowflake replication groups and failover groups support replication of databases and other account objects needed by an application. A secondary failover group can be promoted to become the primary group during an outage, allowing client workloads to resume against the secondary account. After the original primary environment is available again, Snowflake supports replication and failback processes to restore the intended primary/secondary topology. This operational capability minimizes recovery time and provides a controlled path for both failover and return to the original environment. See Snowflake's Introduction to replication and failover and Failover and failback documentation.

QUESTION NO: 53

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 full data-protection lifecycle of permanent tables. They persist as database objects until they are explicitly dropped; unlike temporary tables, their existence is not limited to the session that created them. This makes transient tables useful for intermediate pipeline data, staging data, or other short-lived-but-shared workloads.

A transient table can also be used as the source for a clone created as a permanent table. Snowflake supports zero-copy cloning across table types through the applicable `CREATE TABLE ... CLONE` syntax. The resulting permanent table has permanent-table behavior for its subsequent lifecycle, while Snowflake documents important storage and data-protection considerations for cloned micro-partitions.

In addition, the Time Travel retention period for a transient table is limited to 0 or 1 day. This limited retention is a defining characteristic of transient objects and helps reduce storage costs compared with permanent objects. Snowflake's documentation describes transient tables, their lack of Fail-safe, and their Time Travel retention limits in the Temporary and transient tables guide. The supported clone syntax and behavior are documented in CREATE TABLE.

QUESTION NO: 54

Which of the following SQL statements will list the version of the drivers currently being used?

A. Execute SELECT CURRENT_ODBC_CLIENT(); from the Web UI

B. Execute SELECT CURRENT_JDBC_VERSION() ; from SnowSQL

C. Execute SELECT CURRENT_CLIENT(); from an application

D. Execute SELECT CURRENT_VERSION (); from the Python Connector

ANSWER: C

Explanation:

`Execute SELECT CURRENT_CLIENT(); from an application` is correct because Snowflake's `CURRENT_CLIENT()` context function identifies the client interface for the active session, including its name and version. This is the appropriate SQL-level method for determining the driver or connector currently being used to establish the session. For example, when a session originates through a JDBC, ODBC, Python, or other supported Snowflake client, the returned value identifies that client and its associated version information. Because the function reports information about the active connection, it is useful for diagnostics, auditing connection behavior, and confirming the client software in use without relying on assumptions about the calling application. The function is executed as a normal SQL query in the same session whose client details are being inspected. Snowflake documents `CURRENT_CLIENT()` as a context function that returns information about the current client. See the official CURRENT_CLIENT documentation and Snowflake's context functions reference.

QUESTION NO: 55

The following JSON is stored in a VARIANT column called src of the CAR_SALES table:

```
{
  "customer": [
    {
      "address": San Francisco, CA",
      "name": "Jane Doe"
    }
  ],
  "date":  "2022-01-28",
  "dealership": "Town Auto Sales",
  "salesperson": {
     "id": "55"
  }
}
```

A user needs to extract the dealership information from the JSON.

How can this be accomplished?

A. select src:dealership from car_sales;

B. select src.dealership from car_sales;

C. select src:Dealership from car_sales;

D. select dealership from car_sales;

ANSWER: A

Explanation:

`select src:dealership from car_sales;` is correct because Snowflake queries elements within semi-structured data stored in a `VARIANT` column using colon notation. Here, `src` is the column containing the JSON document, and `:dealership` traverses from that document's root to the `dealership` key. The expression returns the corresponding JSON value as a VARIANT value for each row in `CAR_SALES`. Snowflake supports this concise syntax for extracting values from objects and arrays in VARIANT data, including when the expression is used directly in a SELECT list. JSON element names are case-sensitive, so the path must match the key name represented in the stored JSON. If a relational string result is needed for later operations, the extracted value can additionally be cast, for example with `src:dealership::STRING`; however, casting is not required simply to extract the dealership value. See Snowflake's documentation on querying semi-structured data and semi-structured data types.

QUESTION NO: 56

Which Snowflake objects can be shared with other Snowflake accounts? (Choose three.)

A. Schemas

B. Roles

C. Secure Views

D. Stored Procedures

E. Tables

F. Secure User-Defined Functions (UDFs)

ANSWER: A C E F

Explanation:

Snowflake Secure Data Sharing lets a provider expose selected objects in a database to consumer accounts through a share, without copying or moving the underlying data. A share can grant `USAGE` on a schema, which makes that schema available as the organizational container for shared objects. The provider can grant `SELECT` on tables, allowing consumers to query the provider-managed data directly. Secure views are specifically designed for sharing because their definition is not exposed to consumers; consumers query the view while the provider can use it to present a controlled subset, transformation, or join of data. Secure user-defined functions can also be added to a share when defined as secure, enabling consumers to invoke governed logic without exposing the function definition. These grants are made to the share, and a consumer accesses them after creating a database from that share. Because the listed answers contain four shareable object types, the instruction to choose three is inaccurate; all four applicable objects should be selected. See Snowflake's Introduction to Secure Data Sharing and GRANT <privileges> ... TO SHARE documentation.

QUESTION NO: 57

Which of the following features are available with the Snowflake Enterprise edition? (Choose two.)

A. Database replication and failover

B. Automated index management

C. Customer managed keys (Tri-secret secure)

D. Extended time travel

E. Native support for geospatial data

ANSWER: A D

Explanation:

Snowflake Enterprise edition includes advanced availability and data-protection capabilities beyond the standard edition. **Database replication and failover** supports business-continuity and disaster-recovery designs by enabling database objects and data to be replicated to another Snowflake account, with failover capabilities available subject to the applicable account edition and replication configuration. This allows organizations to establish a secondary environment that can be used when a primary environment is unavailable. Snowflake documents replication and failover groups as its framework for replicating supported objects and coordinating failover of replicated environments.

Extended time travel is also an Enterprise edition capability. It increases the maximum Time Travel data-retention period from the standard one-day retention period to as much as 90 days for permanent databases, schemas, and tables. Longer retention enables recovery of historical data, dropped objects, and prior data versions over a substantially broader recovery window. These Enterprise features help organizations meet stronger resilience, operational-recovery, and data-protection requirements. See Snowflake's edition overview and its documentation on replication and failover.

QUESTION NO: 58

Which of the following objects can be cloned? (Choose four.)

A. Tables

B. Named File Formats

C. Schemas

D. Shares

E. Databases

F. Users

ANSWER: A B C E

Explanation:

Snowflake zero-copy cloning creates a logical copy of supported objects without initially duplicating their underlying micro-partition data. **Tables** can be cloned at a specified point in time or from their current state, allowing teams to create isolated copies for development, testing, recovery, or analysis while initially sharing the source table's storage. **Schemas** and **Databases** can also be cloned. Their clones reproduce the contained supported objects and their metadata, which makes it practical to provision a consistent environment from an existing database or schema.

Named File Formats are supported cloneable objects in the context of cloning a schema or database. A file format is a named schema object that defines parsing settings for staged data, and its definition is included when its containing schema or database is cloned. The clone is independent for subsequent metadata changes, while Snowflake's zero-copy architecture avoids unnecessary immediate data duplication for cloned data-bearing objects. Snowflake documents the supported object types and the behavior of cloned database and schema contents in its Object Cloning guide. The SQL forms and point-in-time cloning behavior are documented in the CREATE ... CLONE reference.

QUESTION NO: 59

True or False: It is possible to unload structured data to semi-structured formats such as JSON and Parquet.

A. True

B. False

ANSWER: A

Explanation:

True is correct. Snowflake's `COPY INTO <location>` command can unload the results of a query or table data to files in a stage, including the semi-structured output formats JSON and Parquet. For JSON unloading, Snowflake writes each unloaded row as a JSON object. For Parquet unloading, Snowflake serializes the result set into the columnar Parquet file format, which is commonly used by analytics engines and data-lake platforms. This capability lets teams extract relational, structured Snowflake data for downstream systems that consume semi-structured or columnar files, without first needing to manually transform the data outside Snowflake. The file format is specified through the `FILE_FORMAT` clause, either by referencing a named file format object or supplying format options inline. Snowflake documentation lists JSON and PARQUET among the supported output file formats for unloading data and describes the format-specific controls available during the unload operation. See Snowflake's overview of unloading data and the COPY INTO <location> command reference for supported output formats and syntax.

QUESTION NO: 60

How can an administrator check for updates (for example, SCIM API requests) sent to Snowflake by the identity provider?

A. ACCESS_HISTORY

B. LOAD_HISTORY

C. QUERY_HISTORY

D. REST EVENT HISTORY

ANSWER: D

Explanation:

REST EVENT HISTORY is correct because Snowflake provides the REST_EVENT_HISTORY table function specifically to expose recent REST API activity sent to a Snowflake account. SCIM provisioning uses REST-based requests from an identity provider to create, update, deactivate, and manage users and roles. An administrator can query this function to review those incoming requests, their timestamps, endpoints, request and response details, status codes, and any associated errors. This makes it the appropriate operational auditing source when diagnosing provisioning issues or confirming that an identity provider sent an expected update. The function is queried from the INFORMATION_SCHEMA and accepts a time range, allowing administrators to focus on a recent troubleshooting window. For example, an administrator can request events from the previous hour and inspect the returned records for SCIM-related activity and outcomes. Snowflake documents REST_EVENT_HISTORY as the source for REST API request history and specifically describes its use in monitoring SCIM activity. Access to this information should be granted and used according to the account's administrative security model, since the records can contain useful audit details about account-management API operations. See Snowflake's REST_EVENT_HISTORY function reference and its SCIM provisioning documentation.

QUESTION NO: 61

Which of the following are considerations when using a directory table when working with unstructured data? (Choose two.)

A. A directory table is a separate database object.

B. Directory tables store data file metadata.

C. A directory table will be automatically added to a stage.

D. Directory tables do not have their own grantable privileges.

E. Directory table data can not be refreshed manually.

ANSWER: B D

Explanation:

Directory tables store data file metadata. A directory table is metadata automatically maintained by Snowflake for files in a stage when the stage's directory-table feature is enabled. This metadata supports querying and locating unstructured files without loading their contents into relational table rows. Snowflake exposes useful attributes such as each file's relative path, size, entity tag, last-modified timestamp, and a scoped file URL. These attributes enable applications and SQL queries to discover files and then access them through supported unstructured-data capabilities.

Directory tables do not have their own grantable privileges. They are associated with a stage rather than existing as independently securable database objects. Consequently, access is governed through the privileges assigned for the underlying stage and its containing objects, including the permissions required to list or read staged files. This security model keeps file metadata and file access aligned with the stage where the files reside. See Snowflake's Directory tables documentation and access-control privilege documentation for the directory-table model and applicable stage permissions.

QUESTION NO: 62

The Query History in the Snowflake Web Interface (UI) is kept for approximately:

A. 60 minutes

B. 24 hours

C. 14 days

D. 30 days

E. 1 year

ANSWER: C

Explanation:

14 days is correct because the Query History page in the Snowflake web interface provides access to query activity from the previous 14 days. This UI history is intended for interactive operational investigation: users can review submitted statements, their execution status, timing, query text, associated warehouse, and other execution details during that recent-history window. The available results are also governed by Snowflake's access-control model, so a user sees query information appropriate to the privileges and roles available to them. For longer-term auditing, monitoring, reporting, or programmatic analysis, Snowflake provides history through its metadata and usage views rather than relying on the UI's recent-query display. In particular, the ACCOUNT_USAGE schema includes the QUERY_HISTORY view, which retains historical query records for substantially longer and is commonly used to build administrative reporting and cost or performance analyses. Therefore, when the question specifically asks about Query History in the web UI, the relevant approximate retention period is 14 days. See Snowflake's Query History and Query Profile documentation and the ACCOUNT_USAGE QUERY_HISTORY view reference for the UI history scope and longer-term account usage history.

QUESTION NO: 63

Which of the following accurately represents how a table fits into Snowflake's logical container hierarchy?

A. Account -> Schema -> Database -> Table

B. Account -> Database -> Schema -> Table

C. Database -> Schema -> Table -> Account

D. Database -> Table -> Schema -> Account

ANSWER: B

Explanation:

Account -> Database -> Schema -> Table is the correct logical container hierarchy in Snowflake. An account is the top-level Snowflake environment that contains databases. Each database is a logical container for schemas, which provide namespaces used to organize database objects. A table is a schema-level object, so it is created within a schema in a database. This relationship is also reflected in a fully qualified table name, expressed as `database_name.schema_name.table_name`. For example, `SALES.PUBLIC.ORDERS` identifies the `ORDERS` table in the `PUBLIC` schema of the `SALES` database, all within the current Snowflake account. Snowflake's object model uses this hierarchy for object organization, naming, access control, and SQL context. Users can set a current database and schema for a session, but the underlying ownership hierarchy remains account, database, schema, then table. See Snowflake's documentation on key concepts and objects and the object name resolution and fully qualified names.

QUESTION NO: 64

Which statements are correct concerning the leveraging of third-party data from the Snowflake Data Marketplace? (Choose two.)

A. Data is live, ready-to-query, and can be personalized.

B. Data needs to be loaded into a cloud provider as a consumer account.

C. Data is not available for copying or moving to an individual Snowflake account.

D. Data is available without copying or moving.

E. Data transformations are required when combining Data Marketplace datasets with existing data in Snowflake.

ANSWER: A D

Explanation:

Data Marketplace listings are delivered to consumers through Snowflake's secure data-sharing architecture. After a consumer obtains a listing, the provider's shared data is available as a database in the consumer's Snowflake account and can be queried immediately with standard SQL. This provides access to current, live data rather than requiring the consumer to first download files, stage them, and load them into storage. Snowflake describes Marketplace data as ready to query and supports using it to enrich and personalize analysis, applications, and business decisions.

Data is available without copying or moving because secure sharing exposes the provider-managed data to the consumer without physically replicating the underlying storage into the consumer account. The consumer receives metadata and query access, while the provider remains responsible for the shared dataset. This architecture enables consumers to join Marketplace data with their own Snowflake data and analyze both through SQL, avoiding traditional data-transfer and ingestion workflows. Provider updates to a shared listing can therefore be made available to authorized consumers without a separate data-delivery process. See Snowflake's Snowflake Marketplace documentation and its introduction to secure data sharing.

QUESTION NO: 65

Which use case does the search optimization service support?

A. Disjuncts (OR) in join predicates

B. LIKE/ILIKE/RLIKE join predicates

C. Join predicates on VARIANT columns

D. Conjunctions (AND) of multiple equality predicates

ANSWER: D

Explanation:

Conjunctions (AND) of multiple equality predicates are supported by Snowflake's search optimization service for join workloads. Search optimization can accelerate eligible joins by using a search access path to locate matching rows in the optimized table efficiently, rather than requiring broad scans of its micro-partitions. For this benefit, the join conditions use equality comparisons, and multiple equality conditions connected with AND can be part of the join condition. For example, a join matching both a customer identifier and a region identifier can be eligible when search optimization is enabled appropriately on the columns of the large, optimized table. The practical performance benefit depends on the query plan and, in particular, on the number of distinct values that must be looked up from the other side of the join; lower lookup cardinality generally provides the strongest benefit. Search optimization is enabled with a search optimization policy and is intended for selective access patterns where its maintenance cost is justified by query-performance gains. Snowflake documents join-query acceleration and the supported equality-predicate patterns in its Search Optimization Service for Join Queries guide. Broader configuration and operational details are available in the Search Optimization Service documentation.

QUESTION NO: 66

Which services does the Snowflake Cloud Services layer manage? (Choose two.)

A. Compute resources

B. Query execution

C. Authentication

D. Data storage

E. Metadata

ANSWER: C E

Explanation:

Authentication and **Metadata** are managed by Snowflake's Cloud Services layer. This layer is the collection of services that coordinates activities across Snowflake and provides the control-plane capabilities needed to operate the platform. Authentication is part of these services because Snowflake must validate users and other principals before allowing them to access account objects, submit work, or use secured resources. Cloud Services supports Snowflake's identity and access workflow alongside related functions such as access control and session management.

Metadata management also belongs to Cloud Services. Snowflake maintains metadata describing database objects and their definitions, including objects such as databases, schemas, tables, views, roles, and policies. It also maintains operational metadata needed for governance and query coordination. This centrally managed metadata enables users and services to locate objects, apply privileges, resolve object references, and coordinate operations consistently without requiring customers to administer infrastructure for the metadata layer. Snowflake documents Cloud Services as the layer responsible for services including authentication, access control, metadata management, query parsing and optimization, and infrastructure management. See the Snowflake key concepts documentation and the Cloud Services layer overview.

QUESTION NO: 67

How does Snowflake reorganize data when it is loaded? (Select TWO).

A. Binary format

B. Columnar format

C. Compressed format

D. Raw format

E. Zipped format

ANSWER: B C

Explanation:

Snowflake stores loaded table data in a **columnar format** and in a **compressed format**. Columnar organization means values for a given column are physically grouped together in Snowflake's immutable micro-partitions. This layout is especially effective for analytical workloads because queries commonly reference a subset of a table's columns; Snowflake can read the relevant column data and use micro-partition metadata to eliminate partitions that cannot contain qualifying rows. The result is reduced data scanning and efficient query processing.

Compression is applied automatically as part of Snowflake's storage management. Snowflake uses optimized compression for the values stored in its micro-partitions, reducing the physical storage footprint without requiring users to choose, configure, or maintain compression settings. The compressed columnar representation is integral to Snowflake's managed architecture: users load data into tables, while Snowflake transparently organizes that data into micro-partitions and maintains the storage characteristics needed for scalable analytics. These two properties therefore describe how Snowflake reorganizes persistent data after loading it. See Snowflake's key concepts documentation and its micro-partitions documentation for details.

QUESTION NO: 68

What causes objects in a data share to become unavailable to a consumer account?

A. The DATA_RETENTI0N_TIME_IN_DAYS parameter in the consumer account is set to 0.

B. The consumer account runs the GRANT IMPORTED PRIVILEGES command on the data share every 24 hours.

C. The objects in the data share are being deleted and the grant pattern is not re-applied systematically.

D. The consumer account acquires the data share through a private data exchange.

ANSWER: C

Explanation:

The objects in the data share are being deleted and the grant pattern is not re-applied systematically. A share exposes provider-owned database objects to consumer accounts only when the provider grants the required object privileges to the share. The consumer's imported database is therefore dependent on both the continued existence of the provider object and the grant relationship between that object and the share. If a provider drops a shared object, it is no longer available through the consumer's imported database. Further, recreating an object after it was dropped creates a new object that must be granted to the share again; access is not restored merely because the replacement has the same name. Providers that use repeatable deployment or grant processes should ensure those processes consistently reapply the necessary grants whenever shared objects are recreated or replaced. This approach keeps the share definition synchronized with the provider's current objects and preserves consumer availability. Snowflake documents the provider workflow for granting database, schema, and object privileges to a share in its Data sharing providers guide. The required syntax and supported privileges for granting objects to a share are described in the GRANT <privileges> ... TO SHARE reference.

QUESTION NO: 69

A deterministic query is run at 8am, takes 5 minutes, and the results are cached. Which of the following statements are true? (Choose two.)

A. The exact query will ALWAYS return the precomputed result set for the RESULT_CACHE_ACTIVE = time period

B. The same exact query will return the precomputed results if the underlying data hasn't changed and the results were last accessed within the previous 24-hour period

C. The same exact query will return the precomputed results even if the underlying data has changed as long as the results were last accessed within the previous 24 hour period

D. The "24 hour" timer on the precomputed results gets renewed every time the exact query is executed

ANSWER: B D

Explanation:

Snowflake can reuse persisted query results for an identical query when the query is deterministic and the data contributing to the result has not changed. Therefore, “The same exact query will return the precomputed results if the underlying data hasn’t changed and the results were last accessed within the previous 24-hour period” is correct. Persisted results are available for reuse for 24 hours after the query completes, provided all reuse requirements remain satisfied. This can avoid both query execution and warehouse compute for the repeated query.

“The ‘24 hour’ timer on the precomputed results gets renewed every time the exact query is executed” is also correct in the context of a repeat query that reuses the persisted result. Each successful reuse refreshes the result’s access window for another 24 hours. Snowflake limits this rolling renewal period: persisted query results are retained for no more than 31 days from their original creation, even when repeatedly reused. Result reuse also requires the identical query text, appropriate permissions, and compatible session and configuration conditions. See Snowflake’s Using Persisted Query Results documentation and its retrieval optimization requirements.

QUESTION NO: 70

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:

Materialized-view maintenance is driven by changes to the underlying data. Therefore, *How often the base table changes* directly affects credit consumption: when Snowflake detects inserts, updates, deletes, or other changes in a base table, it performs background maintenance to keep the materialized-view results current. A higher volume or frequency of base-table changes generally requires more maintenance work and can consume more credits.

Whether the materialized view has a cluster key defined also affects consumption because a materialized view can be clustered, and maintaining its clustering can require automatic reclustering work. Snowflake charges credits for both materialized-view maintenance and automatic clustering. Consequently, a clustered materialized view may incur credits not only to refresh the view after base-table changes, but also to preserve the clustering organization as its data changes. These are serverless Snowflake services, so the related consumption is recorded separately from credits used by a virtual warehouse to execute user queries. Snowflake documents materialized-view maintenance and its credit considerations at Materialized views, and describes the credit usage associated with clustering at Automatic clustering.

QUESTION NO: 71

What transformations are supported in a CREATE PIPE ... AS COPY ... FROM (....) statement? (Select TWO.)

A. Data can be filtered by an optional where clause

B. Incoming data can be joined with other tables

C. Columns can be reordered

D. Columns can be omitted

E. Row level access can be defined

ANSWER: C D

Explanation:

“Columns can be reordered” and “Columns can be omitted” are supported because a Snowpipe definition can use a select statement as the source of the COPY operation. In this pattern, the select list determines both which staged-file fields are loaded and the order in which those selected values map to the target table columns. For example, the select list can reference file columns in a different sequence from their source positions, allowing the incoming fields to be mapped to a target schema with a different column order. The select list can also include only the fields required by the destination table, leaving unselected source fields out of the load. This makes Snowpipe suitable for lightweight, row-by-row load-time transformations, including selecting fields and applying supported expressions such as casts. Snowflake documents this capability as using a select statement in the COPY INTO command to perform transformations during a Snowpipe load, while retaining the operational simplicity of continuous ingestion. See the Snowpipe documentation and Snowflake’s guidance on transforming data during a load.

QUESTION NO: 72

A Snowflake user wants to temporarily bypass a network policy by configuring the user object property MINS_TO_BYPASS_NETWORK_POLICY.

What should they do?

A. Use the SECURITYADMIN role.

B. Use the SYSADMIN role.

C. Use the USERADMIN role.

D. Contact Snowflake Support.

ANSWER: C

Explanation:

Use the USERADMIN role. USERADMIN is Snowflake’s dedicated system role for user and role administration, making it the appropriate role to administer user-object settings such as `MINS_TO_BYPASS_NETWORK_POLICY`. This property is set with the `ALTER USER` command and specifies a limited number of minutes during which the user can authenticate without the applicable network policy being enforced. It is intended for a controlled, temporary access-recovery scenario rather than a permanent removal of network restrictions. In practice, the acting role must also have the required authority over the specific user object; Snowflake’s access-control model requires ownership of an object to alter it, and USERADMIN owns users it creates. Organizations can delegate or transfer ownership as needed so that the administrative role responsible for the user can perform this change. The time-limited setting supports secure operational continuity while retaining the account’s network-policy configuration after the bypass interval ends. See Snowflake’s ALTER USER reference for the parameter and command syntax, and its system-defined roles documentation for USERADMIN’s user-management purpose.

QUESTION NO: 73

What is the lowest Snowflake edition that offers Time Travel up to 90 days?

A. Standard Edition

B. Premier Edition

C. Enterprise Edition

D. Business Critical Edition

ANSWER: C

Explanation:

Enterprise Edition is the lowest Snowflake edition that supports extending the Time Travel data retention period to as many as 90 days. Time Travel preserves historical table, schema, and database data, enabling users to query, clone, and restore data as it existed at a prior point within the configured retention period. This longer retention capability is particularly valuable for recovery from accidental changes and for audit or operational requirements that need access to historical data over an extended timeframe.

For permanent objects, the `DATA_RETENTION_TIME_IN_DAYS` parameter can be set from 0 through 90 when the account is Enterprise Edition or higher. The effective retention period can be configured at the account, database, schema, or individual object level, subject to inherited settings and applicable privileges. Therefore, although higher editions also provide this capability, Enterprise Edition is the first edition tier that offers a maximum Time Travel retention period of 90 days. See Snowflake's Understanding and using Time Travel documentation and the DATA_RETENTION_TIME_IN_DAYS parameter reference.

QUESTION NO: 74

Which formats does Snowflake store unstructured data in? (Choose two.)

A. GeoJSON

B. Array

C. XML

D. Object

E. BLOB

ANSWER: B D

Explanation:

The terminology in this question is imprecise: `ARRAY` and `OBJECT` are Snowflake's native data types for representing nested, semi-structured document content. An `ARRAY` stores an ordered list of values, including values of different data types and nested structures. An `OBJECT` stores a set of key-value pairs, which is the natural representation for JSON-style records and nested documents. Both types can be nested within each other and can also be held in a `VARIANT` value, allowing Snowflake to preserve and query hierarchical data without first flattening it into relational columns. These types are therefore the applicable native formats among the choices for storing data with flexible or nested structure in Snowflake. Snowflake documents `ARRAY`, `OBJECT`, and `VARIANT` as its semi-structured data types and describes how they support hierarchical data representations. See the Snowflake semi-structured data type reference and the introduction to semi-structured data.

QUESTION NO: 75

Which activities are managed by Slowflake's Cloud Services layer? (Select TWO).

A. Authorisation

B. Access delegation

C. Data pruning

D. Data compression

E. Query parsing and optimization

ANSWER: A E

Explanation:

Snowflake's Cloud Services layer manages the services that coordinate and govern activity across the platform, rather than supplying the virtual machines that execute queries or the storage layer that persists data. **Authorisation** is managed here because Cloud Services authenticates users and evaluates Snowflake's access-control model, including roles and privileges, to determine whether a requested operation is permitted. This centralized control enables consistent governance for database objects and account-level actions.

Query parsing and optimization is also a Cloud Services responsibility. When a SQL statement is submitted, Snowflake parses and compiles it, then applies its optimizer to produce an efficient execution plan. The resulting plan is dispatched to the appropriate virtual warehouse, where the compute resources carry out the actual query execution. Separating these coordination and optimization functions from compute is a core aspect of Snowflake's multi-cluster shared-data architecture. Snowflake documents Cloud Services as handling authentication, access control, query parsing, optimization, and related metadata-management services. See the Snowflake key concepts and architecture documentation and the access control overview for details.

QUESTION NO: 76

In an auto-scaling multi-cluster virtual warehouse with the setting SCALING_POLICY = ECONOMY enabled, when is another cluster started?

A. When the system has enough load for 2 minutes

B. When the system has enough load for 6 minutes

C. When the system has enough load for 8 minutes

D. When the system has enough load for 10 minutes

ANSWER: B

Explanation:

When the system has enough load for 6 minutes is correct. With an auto-scaling multi-cluster warehouse, Snowflake evaluates queued workload and can add clusters between the configured minimum and maximum cluster counts. The ECONOMY scaling policy is designed to reduce credit consumption by applying a more conservative threshold before it starts an additional cluster. Specifically, Snowflake starts another cluster when it determines there is enough load to keep that cluster busy for at least six minutes. This avoids starting clusters for short-lived spikes that may clear quickly, while still allowing the warehouse to scale out when sustained concurrent demand justifies the additional compute capacity.

This behavior applies to multi-cluster warehouses operating in auto-scale mode; the warehouse's MIN_CLUSTER_COUNT and MAX_CLUSTER_COUNT define the allowed scaling range. Once demand subsides, Snowflake also evaluates idle clusters and can reduce the number of active clusters, subject to the scaling policy and configured limits. For the authoritative definition of ECONOMY and details on multi-cluster warehouse behavior, see the Snowflake documentation for multi-cluster warehouses and the SCALING_POLICY parameter reference.

QUESTION NO: 77

Which statements reflect valid commands when using secondary roles? (Select TWO).

A. Use SECONDARY ROLES RESUME

B. USE SECONDARY ROLES SUSPEND

C. USE SECONDARY RLES ALL

D. USE SECONDARY ROLES ADD

E. Use SECONDARY ROLES NONE

ANSWER: C E

Explanation:

Secondary roles let a user make the privileges granted through additional assigned roles available during the current session, in addition to the active primary role. The valid `USE SECONDARY ROLES ALL` form enables all roles granted to the current user as secondary roles. This is useful when a session needs the combined privileges of the user's granted roles without requiring the user to switch their primary role repeatedly. The command is evaluated for the current session, and the enabled secondary-role privileges participate in authorization checks for supported operations.

The valid `USE SECONDARY ROLES NONE` form disables secondary roles for the current session. With this setting, Snowflake does not use privileges inherited from secondary roles; authorization is based on the primary role and its inherited role hierarchy. This can be useful where a session should operate with a narrower, explicitly selected privilege set. Snowflake documents `ALL` and `NONE` as the supported values for the `USE SECONDARY ROLES` command. See the USE SECONDARY ROLES command reference and Snowflake's secondary roles access-control documentation.