PassQueen
a.k.a dumpsqueen.com

SnowPro Core Recertification Exam

Snowflake COF-R02

Version Demo

Total Demo Questions: 37

Total Premium Questions: 373

Buy Premium PDF

https://passqueen.com

support@passqueen.com

Topic Break Down

Topic	No. of Questions
Topic 1, Snowflake Architecture	84
Topic 2, Account Access and Security	73
Topic 3, Performance Concepts	63
Topic 4, Data Loading and Unloading	78
Topic 5, Data Transformations	18
Topic 6, Data Protection and Data Sharing	57
Total	373

QUESTION NO: 1

Which of the following objects can be directly restored using the UNDROP command? (Choose two.)

A. Schema

B. View

C. Internal stage

D. Table

E. User

F. Role

ANSWER: A D

Explanation:

Schema and **Table** can be directly restored with Snowflake's UNDROP commands. Snowflake retains dropped schemas and tables for the applicable Time Travel retention period, enabling recovery when an object was removed accidentally or needs to be reinstated. The relevant commands are `UNDROP SCHEMA <name>` and `UNDROP TABLE <name>`. Restoring a schema returns the schema object and its eligible dropped contents according to Snowflake's documented restoration behavior; restoring a table returns the table with its metadata and data as retained through Time Travel. These operations are especially useful for operational recovery because they restore the dropped object under its original name, provided that name is not currently occupied by another object of the same type in the same namespace. Snowflake documents UNDROP as a set of object-specific commands, including the commands for schemas and tables. Administrators should perform recovery within the retention window and ensure they have the privileges required to create and own the restored object. See Snowflake's UNDROP SCHEMA documentation and UNDROP TABLE documentation for syntax, requirements, and restoration details.

QUESTION NO: 2

Which Snowflake function will interpret an input string as a JSON document, and produce a VARIANT value?

A. parse_json()

B. json_extract_path_text()

C. object_construct()

D. flatten

ANSWER: A

Explanation:

`parse_json()` is the Snowflake function that parses a string containing a JSON document and returns the result as a `VARIANT` value. This conversion makes the JSON structure available to Snowflake's semi-structured data features, so its object fields and array elements can subsequently be queried with path notation, cast to relational types where appropriate, or processed using semi-structured functions. For example, `PARSE_JSON('{"customer":"Ada","active":true}')` produces a VARIANT containing an object rather than retaining the input as plain VARCHAR text. The input must be valid JSON syntax; JSON `null` is represented as a VARIANT containing a JSON null, while a SQL NULL input produces SQL NULL. Snowflake documents `PARSE_JSON` specifically as interpreting an input string as a JSON document and returning a VARIANT value. See the Snowflake PARSE_JSON function reference and the introduction to semi-structured data for details on working with VARIANT data.

QUESTION NO: 3

Which of the following are handled by the cloud services layer of the Snowflake architecture? (Choose two.)

A. Query execution

B. Data loading

C. Time Travel data

D. Security

E. Authentication and access control

ANSWER: D E

Explanation:

The cloud services layer handles the centralized services that coordinate and govern activity across a Snowflake account. **Security** is handled at this layer because Snowflake centrally enforces security policies and manages the account-level services needed to protect data and control platform activity. This includes the mechanisms that ensure users and workloads operate within the permissions and policies defined for the account.

Authentication and access control are explicit cloud services responsibilities. Authentication verifies the identity of users or clients attempting to connect to Snowflake. Access control then evaluates the applicable roles, privileges, and object permissions to determine whether an authenticated principal can perform a requested action. These controls are applied consistently regardless of which virtual warehouse performs compute work or where the underlying data is stored. Snowflake documents authentication, access control, query optimization, metadata management, and infrastructure management as cloud services functions. This separation allows compute resources to focus on processing work while governance and coordination remain centrally managed. See the Snowflake key concepts and architecture documentation and the access control overview for details.

QUESTION NO: 4

Which columns are part of the result set of the Snowflake LATERAL FLATTEN command? (Choose two.)

A. CONTENT

B. PATH

C. BYTE_SIZE

D. INDEX

E. DATATYPE

ANSWER: B D

Explanation:

The correct result-set columns are **PATH** and **INDEX**. Snowflake's `FLATTEN` table function expands semi-structured data, such as a VARIANT containing an array or object, into rows. Along with the expanded data, it returns metadata columns that identify where each emitted element came from within the original structure.

PATH contains the path to the flattened element within the input semi-structured value. This is useful for identifying the element's location, particularly when flattening nested objects or arrays. **INDEX** contains the array index of an element when the flattened input is an array. Array indexes are zero-based; for object elements, this field is NULL because objects use keys rather than numeric positions.

The documented output schema for `FLATTEN` includes `SEQ`, `KEY`, `PATH`, `INDEX`, `VALUE`, and `THIS`. These columns remain available when the table function is invoked with a `LATERAL` join, enabling each source row to be correlated with its flattened elements. See Snowflake's FLATTEN function documentation and its guide to querying semi-structured data.

QUESTION NO: 5

True or False: Reader Accounts are able to extract data from shared data objects for use outside of Snowflake.

A. True

B. False

ANSWER: B

Explanation:

False is correct. A reader account is a restricted type of Snowflake account that a provider creates and manages so that a consumer can access and query data shared with them without having its own full Snowflake account. The consumer can use shared databases in queries and can perform analysis within Snowflake, but reader accounts are specifically designed to keep the provider's shared data governed within the Snowflake environment. Snowflake documentation states that reader accounts cannot extract data from shared databases for use outside Snowflake. This restriction is fundamental to the sharing model: the provider retains control of the source data and can manage access through the share, while the reader account receives secure, live access rather than an exportable copy. The limitation applies to data obtained through a share; therefore, querying shared objects does not grant permission to unload, download, or otherwise extract that shared data for external use. For additional detail, see Snowflake's reader account documentation and its introduction to Secure Data Sharing.

QUESTION NO: 6

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

A. Use the DROP command after the load completes.

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

C. Specify the PURGE = TRUE copy option in the COPY INTO command.

D. Use the REMOVE command after the load completes.

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

ANSWER: C D

Explanation:

Snowflake supports deleting staged files either as part of a successful data load or through an explicit stage-file removal operation. Specifying the `PURGE = TRUE` copy option in the `COPY INTO <table>` command directs Snowflake to remove source files from the stage after they have been successfully loaded. This is useful when the staged files are no longer needed and the load process should include cleanup. The purge behavior applies only after a successful load, helping avoid removal of files that were not loaded successfully. Snowflake notes that purging is performed on a best-effort basis, so operational processes should account for and, if needed, verify cleanup.

The `REMOVE` command provides explicit cleanup of files in an internal or external stage after loading is complete. It can remove all files at a stage location or target files using a path and optional pattern, making it appropriate for controlled or scheduled stage maintenance. The executing role must have the required stage privileges. See Snowflake's documentation for COPY INTO <table> and REMOVE.

QUESTION NO: 7

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

A. Columns can be aggregated

B. Columns can be joined with an existing table

C. Columns can be reordered

D. Columns can be omitted

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

ANSWER: C D

Explanation:

The correct tasks are **Columns can be reordered** and **Columns can be omitted**. When loading staged data with `COPY INTO <table>`, Snowflake supports specifying a target-column list. This list determines which destination columns receive loaded values and the order in which values are mapped to them. Consequently, a load can provide the destination columns in an order different from their physical definition in the target table, allowing columns to be reordered for the operation. A target-column list can also include only the columns intended to receive source data, omitting other target-table columns. Any omitted target columns receive their defined default value when applicable, or NULL when permitted. This makes COPY flexible for partial loads and for files whose field order does not match the table definition. Snowflake also supports a SELECT-based transformation in COPY for selecting and mapping fields before they are inserted, but the key capabilities tested here are the target-column mapping behaviors: selecting a subset of target columns and controlling the mapping order. See Snowflake's COPY INTO <table> reference and its documentation on transforming data during a load.

QUESTION NO: 8

Which command can be used to load data into an internal stage?

A. LOAD

B. copy

C. GET

D. PUT

ANSWER: D

Explanation:

The `PUT` command uploads data files from a local file system to a Snowflake internal stage. It supports named internal stages, user stages, and table stages, and it is typically used as the first step in a bulk-loading workflow. For example, `PUT file:///tmp/data.csv @my_internal_stage;` transfers the local CSV file into the specified internal stage. After the files are staged, the `COPY INTO <table>` command can load their records into a Snowflake table. The command can also apply upload controls such as automatic compression, parallelism, and overwrite behavior. Snowflake internal stages are managed storage locations within Snowflake, so no external cloud-storage credentials are required to upload files to them through `PUT`. The command is executed through supported Snowflake clients, such as SnowSQL, Snowflake CLI, drivers, or the web interface's supported mechanisms; it is not a SQL data-manipulation command for inserting table rows directly. For the syntax, supported stage targets, and upload behavior, see Snowflake's PUT command reference. The broader staged-file loading process is described in Loading data from a local file system.

QUESTION NO: 9

Which of the following describes how clustering keys work in Snowflake?

A. Clustering keys update the micro-partitions in place with a full sort, and impact the DML operations.

B. Clustering keys sort the designated columns over time, without blocking DML operations

C. Clustering keys create a distributed, parallel data structure of pointers to a table's rows and columns

D. Clustering keys establish a hashed key on each node of a virtual warehouse to optimize joins at run-time

ANSWER: B

Explanation:

Clustering keys sort the designated columns over time, without blocking DML operations. In Snowflake, a clustering key defines one or more columns or expressions that Snowflake uses to co-locate related rows within micro-partitions. Rather than modifying existing micro-partitions in place, Snowflake reclusters data by creating new, better-organized micro-partitions and removing the prior ones when they are no longer needed. This improves pruning for queries that filter on the clustering columns, potentially reducing the amount of data scanned.

When automatic clustering is enabled or Snowflake determines reclustering is needed, the work is performed asynchronously in the background. Consequently, normal table operations—including inserts, updates, deletes, and merges—can continue; they are not blocked while reclustering occurs. The degree of clustering can naturally change as DML adds or changes data, and Snowflake incrementally maintains clustering over time. Clustering keys are most useful for large tables with query patterns that repeatedly use selective predicates on the designated columns. See Snowflake's Clustering Keys documentation and its Automatic Clustering documentation.

QUESTION NO: 10

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 supported Snowflake data types for storing and working with semi-structured data. VARIANT is Snowflake's flexible, self-describing type: it can store a complete semi-structured value, including documents and nested values from formats such as JSON, Avro, ORC, and Parquet. Snowflake preserves the hierarchical structure on load, allowing SQL queries to navigate nested elements with path notation and to flatten values when rows are needed.

ARRAY is a native semi-structured type that represents an ordered collection of values. It is useful when a column or a value within a VARIANT document contains a list, such as product identifiers, event attributes, or measurements. Array elements can be accessed by index and expanded using Snowflake functions such as FLATTEN. Together with OBJECT, which represents key-value pairs, these types form Snowflake's native support for semi-structured data. The requested pair is therefore VARIANT and ARRAY. Snowflake documentation describes these native data types and their use in storing and querying hierarchical data: Semi-structured data types and Introduction to semi-structured data.

QUESTION NO: 11

Which query profile statistics help determine if efficient pruning is occurring? (Choose two.)

A. Bytes sent over network

B. Percentage scanned from cache

C. Partitions total

D. Bytes spilled to local storage

E. Partitions scanned

ANSWER: C E

Explanation:

Efficient micro-partition pruning is evaluated by comparing **Partitions scanned** with **Partitions total** in the query profile. Snowflake stores metadata for each micro-partition, including minimum and maximum values for columns. When a query predicate can use that metadata, Snowflake eliminates micro-partitions that cannot contain qualifying rows before reading their data. A query that scans only a small subset of the total available partitions is therefore benefiting from effective pruning.

The total-partition value establishes the scope of data that was eligible to be considered for the table scan, while the scanned-partition value shows how much of that scope Snowflake actually had to read. Reviewing both values together is essential: the count of scanned partitions alone does not indicate whether that count is efficient without the total number of partitions as context. These metrics are displayed in query-profile operator details and are useful when assessing predicate selectivity, clustering effectiveness, and opportunities to reduce scanned data. Snowflake documents micro-partition pruning and the metadata used for it in its Micro-partitions and data clustering guide; query-profile operator statistics are described in Using Query Profile.

QUESTION NO: 12

Which of the following statements are true of Resource Monitors? (Choose two.)

A. They can monitor credit usage by virtual warehouses

B. They can suspend assigned warehouses when a credit threshold is reached

C. They automatically delete data when a storage quota is reached

D. They prevent all Cloud Services credit consumption

ANSWER: A B

Explanation:

They can monitor credit usage by virtual warehouses and **they can suspend assigned warehouses when a credit threshold is reached**. A resource monitor tracks warehouse credit consumption against a defined quota during a specified frequency interval. Administrators can configure actions at one or more percentage thresholds, including sending notifications or suspending warehouses.

Resource monitors are used to help control virtual warehouse credit consumption. They do not measure storage consumption, and they do not prevent Cloud Services charges. See Snowflake's Resource monitors documentation and the CREATE RESOURCE MONITOR reference.

QUESTION NO: 13

How would a user execute a series of SQL statements using a task?

A. WHERE ..

B. call stored_proc1(); call stored_proc2();

C. Use a stored procedure executing multiple SQL statements and invoke the stored procedure from the task. CREATE TASK mytask .... AS call stored_proc_multiple_statements_inside();

D. Create a task for each SQL statement (e.g. resulting in task1, task2, etc.) and string the series of SQL statements by having a control task calling task1, task2, etc. sequentially.

ANSWER: C

Explanation:

Use a stored procedure executing multiple SQL statements and invoke the stored procedure from the task. CREATE TASK mytask .... AS call stored_proc_multiple_statements_inside(); is correct because a task can use its SQL action to call a stored procedure, while the procedure encapsulates the required sequence of SQL statements. This provides one scheduled or event-triggered task execution that invokes a single callable unit, with the procedure responsible for running its statements in the intended order. The task can be created with a `CALL` statement as its body, such as `CREATE TASK mytask ... AS CALL stored_proc_multiple_statements_inside();`. The stored procedure may be written using Snowflake Scripting, JavaScript, Java, Python, or Scala, depending on the implementation requirements, and can contain control flow, error handling, transactional design where applicable, and multiple SQL commands. This pattern also keeps orchestration simple: the task governs scheduling and execution, while procedure code contains the multi-step database logic. Snowflake documents that a task definition includes the SQL statement to execute and supports calling a stored procedure. See the CREATE TASK reference and Snowflake's stored procedure overview.

QUESTION NO: 14

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:

Snowflake supports MFA for interactive access through the Snowflake Web Interface (UI) and for supported client connectivity through JDBC, SnowSQL, ODBC, and the Python connector. MFA adds a second verification factor to the standard user sign-in process, helping protect accounts even if a password is compromised. For programmatic clients, Snowflake supports MFA authentication through the applicable connector configuration and can use MFA token caching where supported, reducing the need to complete a second-factor challenge for every new connection while retaining the configured security controls. SnowSQL also supports MFA-based authentication for command-line sessions. The web interface provides the browser-based sign-in flow where users complete their enrolled MFA challenge directly. Administrators must enable MFA for the user and the user must enroll an authenticator before the user can successfully perform MFA authentication. Snowflake documents the supported MFA clients and the relevant connection parameters in its MFA and connector documentation. See Snowflake Multi-factor Authentication documentation and Python Connector MFA documentation.

QUESTION NO: 15

Which of the following statement is true of Snowflake?

Select one.

A. It was built specifically for the cloud

B. it was built as an on-premises solution and then potted to the cloud

C. It was designed as a hybrid database to allow customers to store data either on premises or in the cloud

D. It was built for Hadoop architecture

E. It's based on an Oracle Architecture

ANSWER: A

Explanation:

It was built specifically for the cloud is correct. Snowflake is a cloud-native data platform, designed from its inception to run on public-cloud infrastructure rather than being an on-premises database adapted for cloud deployment. Its architecture separates storage, compute, and cloud services into independent layers. This design lets organizations scale virtual warehouses independently of stored data, run multiple workloads concurrently without requiring them to share compute resources, and use elastic cloud capacity as needed.

Snowflake is delivered as a fully managed service on Amazon Web Services, Microsoft Azure, and Google Cloud. Customers do not install or operate Snowflake servers, storage systems, or underlying database infrastructure; Snowflake manages the platform while cloud providers supply the foundational infrastructure. The cloud-built architecture also supports capabilities such as automatic infrastructure management, secure data sharing, and cross-region or cross-cloud replication features, depending on the account edition and configuration. Snowflake documentation describes the platform as built for the cloud and explains its distinct storage, compute, and cloud services layers. See the Snowflake key concepts documentation and the supported cloud platforms documentation.

QUESTION NO: 16

If 3 size Small virtual warehouse is made up of two servers, how many servers make up a Large warehouse?

A. 4

B. 8

C. 16

D. 32

ANSWER: B

Explanation:

8 is correct. Snowflake virtual warehouses use progressively larger compute configurations as their T-shirt size increases. In the standard sizing progression, a Small warehouse contains two servers, a Medium warehouse contains four servers, and a Large warehouse contains eight servers. Each step up doubles the underlying server count: moving from Small to Medium doubles two to four, and moving from Medium to Large doubles four to eight. Therefore, starting with the stated Small warehouse configuration of two servers, the corresponding Large warehouse is made up of eight servers.

This sizing model is important because it provides greater parallel processing capacity as warehouse size increases. A Large warehouse can execute more work concurrently than a Small warehouse, subject to the workload and query characteristics, while also consuming credits at the documented rate for that size. Snowflake documents the warehouse-size progression and its associated server counts in its virtual warehouse guidance. See Virtual warehouses and Warehouse considerations.

QUESTION NO: 17

When cloning a database containing stored procedures and regular views, that have fully qualified table references, which of the following will occur?

A. The cloned views and the stored procedures will reference the cloned tables in the cloned database.

B. An error will occur, as views with qualified references cannot be cloned.

C. An error will occur, as stored objects cannot be cloned.

D. The stored procedures and views will refer to tables in the source database.

ANSWER: D

Explanation:

The stored procedures and views will refer to tables in the source database. A fully qualified reference explicitly embeds the source database, schema, and object name in the stored view definition or procedure body. Cloning copies the relevant object definitions into the target database, but it does not rewrite those explicitly qualified identifiers to substitute the cloned database name. Consequently, after the clone is created, executing a cloned procedure or querying a cloned regular view that contains a reference such as `SOURCE_DB.PUBLIC.MY_TABLE` continues to resolve that reference to `SOURCE_DB.PUBLIC.MY_TABLE`, rather than to the corresponding table in the new clone.

This behavior is especially important when using clones for development, testing, or recovery workflows. The cloned objects exist independently, but their hard-coded fully qualified dependencies can still access the source database. To make such objects use cloned tables, their definitions must be updated to reference the clone explicitly, or designed with appropriate unqualified or dynamically resolved references where suitable. Snowflake documents object-cloning behavior and its considerations in the Object Cloning guide and the CREATE <object> ... CLONE reference.

QUESTION NO: 18

True or False: A Virtual Warehouse can be resized while suspended.

A. True

B. False

ANSWER: A

Explanation:

True is correct. Snowflake allows a virtual warehouse's size to be changed whether the warehouse is running or suspended. The change can be made with the `ALTER WAREHOUSE` command by setting the `WAREHOUSE_SIZE` property, or through the Snowsight interface. This capability is useful for planned workload changes: an administrator can suspend an idle warehouse to avoid consuming credits, adjust its size in advance, and then resume it when the workload is ready to run.

When a suspended warehouse is resized, no compute resources are active at the time of the configuration change. Snowflake applies the selected warehouse size when the warehouse next starts, provisioning the appropriate compute resources at resume time. This supports cost-conscious operational practices while still allowing teams to prepare capacity for expected query volume, concurrency, or more demanding processing requirements. Warehouse resizing is an administrative configuration operation and is independent of the warehouse's current execution state.

See Snowflake's documentation for ALTER WAREHOUSE and its guidance on virtual warehouses.

QUESTION NO: 19

What privileges are required to create a task?

A. The global privilege create task is required to create a new task.

B. Tasks are created at the Application level and can only be created by the Account Admin role.

C. Many Snowflake DDLs are metadata operations only, and create task DDL can be executed without virtual warehouse requirement or task specific grants.

D. The role must have access to the target schema and the create task privilege on the schema itself.

ANSWER: D

Explanation:

The role must have access to the target schema and the create task privilege on the schema itself. In Snowflake's access-control model, tasks are schema-level objects, so creation is authorized through privileges granted on the schema where the task will reside. The creating role needs the `CREATE TASK` privilege on that schema. It also needs `USAGE` on both the parent

database and the target schema, which provides the required access to resolve and use that namespace. For example, an administrator can grant `USAGE` on a database, `USAGE` on a schema, and `CREATE TASK` on that schema to a role before that role runs `CREATE TASK`. The task's owner is the role that creates it, and that ownership controls subsequent task administration. Additional privileges can be necessary depending on the task definition and execution model—for example, access to referenced objects, a warehouse, or account-level task execution privileges—but the core authorization to create the task object is schema access plus `CREATE TASK` on the schema. Snowflake documents task creation requirements and task privileges in its CREATE TASK reference and task privilege documentation.

QUESTION NO: 20

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 supports tables, named file formats, schemas, and databases. Cloning creates a new object that initially references the source object's existing micro-partitions rather than physically copying table data. This makes creating a clone fast and storage-efficient at the time it is created; additional storage is incurred only as either the source or clone is changed independently. Tables can be cloned at a current or specified historical point in time, subject to Time Travel data availability. Schemas and databases can also be cloned, allowing an entire logical grouping of objects to be copied quickly for development, testing, recovery, or isolated analysis. When a schema or database is cloned, eligible contained objects are cloned as part of that operation. Named file formats are independently cloneable schema objects, so their parsing and formatting configuration can be reproduced without manually recreating each setting. Snowflake documents supported clone operations and syntax under its zero-copy cloning guidance, including cloning databases, schemas, tables, and supported schema-level objects. See Snowflake zero-copy cloning and the CREATE FILE FORMAT reference.

QUESTION NO: 21

When a Pipe is recreated using the CREATE OR REPLACE PIPE command:

A. The Pipe load history is reset to empty

B. The REFRESH parameter is set to TRUE

C. Previously loaded files will be ignored

D. All of the above

ANSWER: A

Explanation:

The Pipe load history is reset to empty is correct. Snowpipe maintains load-history metadata for a pipe so it can identify files that have already been processed and avoid loading the same staged file repeatedly. Issuing `CREATE OR REPLACE PIPE` replaces the existing pipe definition, and Snowflake explicitly documents that this operation resets the pipe's load history. Consequently, the replacement pipe no longer has the prior pipe's file-processing record.

This behavior is significant when recreating a pipe that points to a stage containing files that remain eligible for loading. Since the per-pipe history has been cleared, files previously ingested through the original pipe can be considered for ingestion again if Snowpipe receives notifications or a refresh operation identifies them. Teams should therefore treat pipe replacement as a potentially replaying operation and ensure that downstream loading is designed to handle duplicate-file scenarios where needed. If the goal is only to change supported pipe properties without discarding the existing load-history context, evaluate whether an `ALTER PIPE` operation is suitable instead.

Snowflake documents this reset behavior in the CREATE PIPE reference. Related details on how Snowpipe tracks and prevents duplicate file loading are available in the data-loading considerations documentation.

QUESTION NO: 22

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

A. VARIANT

B. ARRAY

C. VARCHAR

D. XML

E. BLOB

ANSWER: A B

Explanation:

VARIANT and **ARRAY** are Snowflake-supported data types for storing and querying semi-structured data. VARIANT is the flexible, general-purpose type used to hold semi-structured values, including complete JSON, Avro, ORC, or Parquet documents and values with nested structures. Snowflake preserves the hierarchical structure in a VARIANT value, enabling SQL path notation to retrieve nested fields and elements directly during queries. For example, a query can use colon and dot notation to access an attribute embedded within a VARIANT column.

ARRAY is a native semi-structured type representing an ordered collection of values. Arrays can be stored directly or occur as nested values within a VARIANT document. Snowflake provides functions and query constructs, especially `FLATTEN`, to expand array elements into rows for analysis, while retaining the ability to navigate and process nested data efficiently. Together with OBJECT, these types are Snowflake's native semi-structured data types; in this question, VARIANT and ARRAY are the two applicable selections. See Snowflake's semi-structured data type documentation and its guide to querying semi-structured data for syntax and examples.

QUESTION NO: 23

What is the default character set used when loading CSV files into Snowflake?

A. UTF-8

B. UTF-16

C. ISO S859-1

D. ANSI_X3.A

ANSWER: A

Explanation:

The default character set for loading CSV data into Snowflake is UTF-8. In a Snowflake file format, the `ENCODING` option determines the character set used to interpret input files. For CSV files, this option defaults to UTF-8, which supports Unicode characters and is widely used for data interchange. Therefore, a CSV file encoded in UTF-8 can be loaded without

explicitly setting an encoding value in the file format or `COPY INTO` command. If an incoming file uses another supported character encoding, the file format's `ENCODING` setting should be explicitly configured so Snowflake decodes the bytes correctly during ingestion. Snowflake documents UTF-8 as the default value for the CSV `ENCODING` file-format option, including the UTF-8 byte-order-mark variant. Using the documented default helps ensure that accented characters, non-Latin scripts, and other Unicode content are interpreted consistently when staged files are loaded into tables. See Snowflake's CREATE FILE FORMAT reference and its data loading considerations for details on CSV encoding and supported character sets.

QUESTION NO: 24

What feature can be used to reorganize a very large table on one or more columns?

A. Micro-partitions

B. Clustering keys

C. Key partitions

D. Clustered partitions

ANSWER: B

Explanation:

Clustering keys are used to organize the data stored in a large Snowflake table according to one or more selected columns or expressions. When a clustering key is defined, Snowflake can recluster the table over time, maintaining improved clustering depth and reducing overlap among the micro-partitions for the key values. This organization helps pruning eliminate micro-partitions that cannot contain values relevant to a query predicate, which can substantially improve query performance for large tables that are frequently filtered, joined, or sorted using those columns.

Clustering is particularly appropriate for very large tables whose natural data-load order does not produce an effective layout, or whose data is significantly modified over time. A clustering key may include multiple columns, allowing Snowflake to organize the table based on a compound access pattern. Automatic Clustering can then perform the ongoing serverless maintenance needed to preserve the clustering quality without requiring manual table reorganization. Snowflake stores all table data in micro-partitions, but clustering keys provide the user-defined mechanism for influencing how those micro-partitions are organized. See Snowflake's Clustering Keys documentation and Automatic Clustering documentation.

QUESTION NO: 25

What is the purpose of the Query Acceleration Service?

Select one.

A. To reduce elapsed time for eligible queries without resizing the warehouse

B. To increase the Time Travel retention period for a table

C. To automatically recluster all tables in a database

D. To add users to a multi-cluster warehouse

ANSWER: A

Explanation:

To reduce elapsed time for eligible queries without resizing the warehouse is correct. The Query Acceleration Service can use shared compute resources to accelerate eligible portions of a query workload. It is intended for queries that can benefit from additional compute capacity, such as certain queries with large scans or selective filtering.

The service is enabled at the warehouse level and is controlled by a query acceleration scale factor. It does not replace warehouse sizing, multi-cluster warehouses, or query optimization practices, but it can improve performance for eligible workloads without requiring the warehouse itself to be resized. See Snowflake's Query Acceleration Service documentation.

QUESTION NO: 26

Which of the following statements are true of External Tables? (Choose two.)

A. The data files remain in external cloud storage

B. External tables can be queried using SQL

C. Rows can be updated using an UPDATE statement

D. Data files are automatically converted into Snowflake micro-partitions

ANSWER: A B

Explanation:

The data files remain in external cloud storage and **external tables can be queried using SQL**. An external table provides a Snowflake table-like metadata layer over files stored in an external stage. Snowflake stores metadata about the files, while the actual data remains in the external cloud storage location.

External tables support querying staged files with SQL, but they are read-only objects. Users cannot insert, update, or delete rows in an external table. The metadata must also be refreshed when appropriate so that Snowflake recognizes changes to the files in the external location. See Snowflake's Introduction to external tables and the CREATE EXTERNAL TABLE reference.

QUESTION NO: 27

Which Snowflake feature will allow small volumes of data to continuously load into Snowflake and will incrementally make the data available for analysis?

A. COPY INTO

B. CREATE PIPE

C. INSERT INTO

D. TABLE STREAM

ANSWER: B

Explanation:

CREATE PIPE is correct because it defines a Snowpipe object, which supports Snowflake's continuous data-loading service. A pipe contains a COPY statement that identifies the source location and target table, along with any required loading transformations or file-format settings. When new data files arrive in the configured stage and Snowpipe is triggered—for example, through cloud event notifications in auto-ingest mode—Snowflake loads the files as they become available rather than waiting for a large scheduled batch. This approach is designed for frequent, small-volume ingestion and makes each successfully loaded set of records available to queries incrementally. Snowpipe manages the loading infrastructure as a serverless service, so the workload does not require a user-managed virtual warehouse for the ingestion operation. Creating the pipe is therefore the essential SQL action for configuring this continuous loading pattern. Snowflake documents that a pipe encapsulates the COPY INTO command used by Snowpipe, and that Snowpipe continuously loads data from staged files into Snowflake tables. See the CREATE PIPE reference and the Snowpipe introduction.

QUESTION NO: 28

What is the recommended compressed file size range for continuous data loads using Snowpipe?

A. 8-16 MB

B. 16-24 MB

C. 10-99 MB

D. 100-250 MB

ANSWER: D

Explanation:

The recommended compressed file size range for continuous data loads using Snowpipe is **100-250 MB**. Snowpipe is designed to ingest data incrementally as files become available in a stage, and this range provides an effective balance between file-management overhead and parallel loading efficiency. Very small files can cause unnecessary per-file processing overhead, while appropriately sized files enable Snowflake to process incoming data efficiently without creating an excessive number of load operations.

Snowflake documents 100-250 MB as the general file-size recommendation for bulk loading and continuous loading scenarios, including Snowpipe. The stated size refers to the file after compression, which is important when preparing data extracts and configuring upstream delivery processes. Producers should therefore batch records into compressed files in this approximate range before placing them in the target stage and triggering Snowpipe ingestion. This practice helps support scalable, cost-effective continuous ingestion as data volume grows. See Snowflake's data loading considerations and Snowpipe introduction and best practices documentation.

QUESTION NO: 29

Which of the following statements describe features of Snowflake data caching? (Choose two.)

A. When a virtual warehouse is suspended, the data cache is saved on the remote storage layer.

B. When the data cache is full, the least-recently used data will be cleared to make room.

C. A user can only access their own queries from the query result cache.

D. A user must set USE_METADATA_CACHE to TRUE to use the metadata cache in queries.

E. The RESULT_SCAN table function can access and filter the contents of the query result cache.

ANSWER: B E

Explanation:

When the data cache is full, the least-recently used data will be cleared to make room. Snowflake's virtual warehouse cache, also called the local disk cache, stores data retrieved from remote storage on the warehouse's local disks. This cache is managed automatically to improve performance for subsequent queries that need the same data. As new data is cached and space is required, Snowflake evicts cached data according to a least-recently-used policy. The cache is tied to the warehouse and is available while that warehouse remains running, allowing repeated access patterns to benefit from reduced remote-storage reads.

The RESULT_SCAN table function can access and filter the contents of the query result cache. RESULT_SCAN exposes the result set produced by a previous command as rows that can be queried like a table. A user can identify the prior result using a query ID or a relative query index, then apply normal SQL operations such as selecting columns, filtering rows, joining, or aggregating the returned result. This makes it useful for further analysis of an already produced query result without rerunning the original statement. See Snowflake's documentation for persisted query results and the RESULT_SCAN function.

QUESTION NO: 30

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:

With `SCALING_POLICY = ECONOMY`, Snowflake delays adding capacity until it determines that the additional cluster is likely to be needed long enough to justify the cost of starting it. Specifically, Snowflake starts another cluster when the system has enough load for 6 minutes. This policy is intended to reduce credit consumption for workloads with short-lived demand spikes, while still allowing a multi-cluster warehouse to scale when sustained concurrency requires additional compute resources.

The six-minute threshold is an economic decision point used by Snowflake's warehouse scaling logic rather than a fixed query-runtime requirement. Once the workload indicates that a new cluster can be used for that duration, Snowflake provisions the cluster so queued and concurrent work can be handled with greater capacity. This behavior applies to auto-scaling multi-cluster warehouses, where Snowflake can start clusters between the configured minimum and maximum cluster counts. For the authoritative description of the Economy policy and its six-minute usage estimate, see Snowflake documentation on multi-cluster warehouse scaling policies. Additional configuration details are available in the ALTER WAREHOUSE reference.

QUESTION NO: 31

What is the default File Format used in the COPY command if one is not specified?

A. CSV

B. JSON

C. Parquet

D. XML

ANSWER: A

Explanation:

CSV is the default file format for a Snowflake COPY operation when no file format is specified, either inline through the `FILE_FORMAT` clause or by referencing a named file format object. This default applies to the standard delimited-text interpretation used when loading staged files with `COPY INTO <table>`. Snowflake documents the default type as `CSV`, with default CSV properties such as a comma field delimiter, newline record delimiter, and a double-quote enclosure character. Consequently, a basic COPY command can load conventionally formatted comma-separated files without first creating a file format object. For files whose structure or parsing requirements differ from those defaults, a user should explicitly specify an appropriate format type and options, or create and reference a reusable named file format. This makes file interpretation explicit and ensures that the load operation applies the intended parsing behavior. The Snowflake COPY command documentation describes the supported syntax and file-format handling at COPY INTO <table>. The documented default settings for the CSV file format are available in Snowflake's CREATE FILE FORMAT reference.

QUESTION NO: 32

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

A. Authentication

B. Resource management

C. Virtual warehouse caching

D. Query parsing and optimization

E. Query execution

F. Physical storage of micro-partitions

ANSWER: A B D

Explanation:

Snowflake's Cloud Services layer provides the centralized services that coordinate and govern activity across a Snowflake account. **Authentication** is a Cloud Services responsibility because this layer validates users and handles identity-related access to Snowflake. **Resource management** is also performed by Cloud Services: it manages the infrastructure-level coordination required to provision, manage, and monitor Snowflake resources, including virtual warehouses. **Query parsing and optimization** belongs in Cloud Services because submitted SQL is parsed and optimized there before the resulting execution plan is sent to compute resources. This separation enables Snowflake to independently manage security, metadata, optimization, and infrastructure coordination while virtual warehouses supply scalable compute for executing work. Snowflake documents the Cloud Services layer as the architectural layer responsible for activities including authentication, access control, query parsing and optimization, metadata management, and infrastructure management. See the Snowflake key concepts documentation and the virtual warehouses overview for details on this separation of services and compute.

QUESTION NO: 33

How long is the Fail-safe period for temporary and transient tables?

A. There is no Fail-safe period for these tables.

B. 1 day

C. 7 days

D. 31 days

E. 90 days

ANSWER: A

Explanation:

There is no Fail-safe period for these tables. Snowflake transient and temporary tables do not receive the additional Fail-safe data-protection stage that follows Time Travel for permanent objects. A transient table can use Time Travel, subject to its configured data-retention period and applicable Snowflake limits, but when that retention period ends, its historical data is purged and cannot enter Fail-safe recovery. Temporary tables are session-scoped and similarly have no Fail-safe protection; they are automatically removed when the creating session ends. This design makes transient and temporary tables appropriate for staging, intermediate transformations, and other data that does not require Snowflake's highest level of recoverability. Data that must remain eligible for Snowflake-managed disaster recovery after Time Travel expires should be stored in permanent tables, which include a seven-day Fail-safe period. See Snowflake's documentation on temporary and transient tables and its explanation of Fail-safe.

QUESTION NO: 34

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

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

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

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

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

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

ANSWER: B C

Explanation:

Micro-partitions are immutable objects that support the use of Time Travel. Snowflake automatically organizes table data into these immutable storage units, so when data is changed, Snowflake preserves the prior micro-partition versions for the applicable data-retention period. This underlying immutable-storage approach enables Time Travel operations, including querying, restoring, and cloning historical data states.

Micro-partitions can reduce the amount of I/O from object storage to virtual warehouses because Snowflake records metadata for each micro-partition, such as the range of values in its columns. When a query includes selective predicates, Snowflake can use this metadata to identify partitions that cannot contain matching rows and avoid scanning them. This process, called micro-partition pruning, reduces the volume of data read from storage and can improve query performance. These capabilities are automatic: users do not need to manually create or maintain traditional partitions. See Snowflake's Micro-partitions and data clustering documentation and its Time Travel and Fail-safe documentation.

QUESTION NO: 35

Which of the following statements are true of transient tables? (Choose two.)

A. They do not have a Fail-safe period

B. They can be cloned

C. They are automatically dropped when the current session ends

D. They support a Time Travel retention period of up to 90 days

ANSWER: A B

Explanation:

Transient tables do not have a Fail-safe period and **can be cloned**. Transient tables are intended for data that does not require the same level of data protection as permanent tables. They support Time Travel, although their retention period is limited to a maximum of one day, but they do not enter Fail-safe after their Time Travel retention period ends.

Like permanent tables, transient tables can be cloned using zero-copy cloning. The clone initially shares the source table's existing storage and incurs additional storage only when data changes independently in the source or clone. Snowflake documents transient-table behavior in its Temporary and transient tables documentation and cloning behavior in its Zero-copy cloning documentation.

QUESTION NO: 36

Which of the following activities consume virtual warehouse credits in the Snowflake environment? (Choose two.)

A. Caching query results

B. Running EXPLAIN and SHOW commands

C. Cloning a database

D. Running a custom query

E. Running COPY commands

ANSWER: D E

Explanation:

Running a custom query consumes virtual warehouse credits because virtual warehouses provide the compute resources that execute SQL workloads. Whenever a query requires processing, such as scanning table data, joining datasets, aggregating results, or performing transformations, Snowflake uses the assigned warehouse while it is running. Warehouse consumption is measured in credits according to the warehouse size and its running duration, subject to Snowflake's per-second billing rules and minimum billing requirements.

Running COPY commands also consumes virtual warehouse credits. COPY operations use warehouse compute to load data from a stage into a Snowflake table or to unload query and table data to a stage. During loading, compute resources parse input files, transform data as needed, and write micro-partitions; during unloading, they execute the source query and generate output files. The amount of warehouse compute required can vary with warehouse size, file volume, file format, transformation complexity, and parallelism. Snowflake documents that virtual warehouses are the compute resources used for query execution and for data loading and unloading operations. See Virtual warehouses and Overview of data loading.

QUESTION NO: 37

Which of the following are options when creating a Virtual Warehouse?

A. Auto-suspend

B. Auto-resume

C. Local SSD size

D. User count

ANSWER: A B

Explanation:

Auto-suspend and **Auto-resume** are configurable virtual warehouse properties in Snowflake. Auto-suspend specifies the amount of inactivity, in seconds, after which a running warehouse automatically suspends. This setting helps control credit consumption because a warehouse does not continue consuming compute credits while it is idle. Snowflake supports configuring this behavior when the warehouse is created and later altering it as operational requirements change.

Auto-resume determines whether a suspended warehouse automatically starts when a statement requiring compute is submitted. When enabled, Snowflake resumes the warehouse as needed, allowing users and workloads to run without requiring a manual resume operation. Together, these settings support a common cost-management pattern: suspend idle compute promptly while retaining seamless availability for new queries. Warehouse creation also supports other compute and behavior settings, such as warehouse size, minimum and maximum cluster counts for multi-cluster warehouses, scaling policy, query acceleration, and resource constraints. The authoritative syntax and property descriptions are documented in Snowflake's CREATE WAREHOUSE reference and its virtual warehouse overview.