PassQueen
a.k.a dumpsqueen.com

SnowPro Advanced: Architect Recertification Exam

Snowflake ARA-R01

Version Demo

Total Demo Questions: 18

Total Premium Questions: 182

Buy Premium PDF

https://passqueen.com

support@passqueen.com

Topic Break Down

Topic	No. of Questions
Topic 1, Account and Security	53
Topic 2, Data Engineering	79
Topic 3, Performance Optimization	30
Topic 4, Data Governance	20
Total	182

QUESTION NO: 1

A user, analyst_user has been granted the analyst_role, and is deploying a SnowSQL script to run as a background service to extract data from Snowflake.

What steps should be taken to allow the IP addresses to be accessed? (Select TWO).

A. ALTER ROLE ANALYST_ROLE SET NETWORK_POLICY= ' ANALYST_POLICY ' ;

B. ALTER USER ANALYST_USER SET NETWORK_POLICY = 'ANALYST_POLICY';

C. ALTER USER ANALYST_USER SET NETWORK_POLICY= ' 10.1.1.20 ' ;

D. USE ROLE SECURITYADMIN;CREATE OR REPLACE NETWORK POLICY ANALYST_POLICY ALLOWED_IP_LIST = (' 10.1.1.20 ');

E. USE ROLE USERADMIN;CREATE OR REPLACE NETWORK POLICY ANALYST_POLICYALLOWED_IP_LIST = (' 10.1.1.20 ');

ANSWER: B D

Explanation:

Creating `ANALYST_POLICY` with `ALLOWED_IP_LIST = ('10.1.1.20')` defines a Snowflake network policy that permits connections originating from that IP address. Network policies evaluate the client IP address at connection time, so this configuration is appropriate for a SnowSQL process running as a background service from a known, fixed egress address. The `SECURITYADMIN` role can create network policies through its default global `CREATE NETWORK POLICY` privilege (or an equivalent role can be granted that privilege).

Using `ALTER USER ANALYST_USER SET NETWORK_POLICY = 'ANALYST_POLICY'` attaches the defined policy directly to the service account. A user-level network policy applies specifically when that user authenticates, independently of the roles granted to the user. This is the appropriate scope when the requirement is to constrain the connections made by `analyst_user`, while allowing the account to maintain separate policies for other users or integrations. The role executing the alteration must have the required authority over the user. Snowflake documents network-policy creation, permitted IP lists, precedence, and assignment to users in its network policies guide and the ALTER USER reference.

QUESTION NO: 2

Which of the following ingestion methods can be used to load near real-time data by using the messaging services provided by a cloud provider?

A. Snowflake Connector for Kafka

B. Snowflake streams

C. Snowpipe

D. Spark

ANSWER: C

Explanation:

Snowpipe is the correct ingestion method because its auto-ingest capability supports continuous, near-real-time loading of files as they arrive in a cloud storage location. It integrates with cloud-provider event messaging to notify Snowflake when new data files are available. For example, Snowpipe auto-ingest can use Amazon S3 event notifications with Amazon SNS and SQS, Azure Event Grid, or Google Cloud Pub/Sub notifications. Once Snowflake receives a notification, Snowpipe retrieves the referenced files from the configured stage and loads them into the target table without requiring users to run a manual COPY INTO command for each batch. This event-driven pattern is designed for continuously arriving file-based data and minimizes the delay between a file landing in cloud storage and its availability in Snowflake. Snowpipe also manages the compute required for loading, which simplifies operating an ongoing ingestion pipeline. The applicable configuration and

cloud messaging integration differ by cloud platform, but the core behavior remains the same: cloud storage events trigger automated file ingestion. See Snowflake's Snowpipe Auto-ingest documentation and its Snowpipe overview.

QUESTION NO: 3

A user named USER_01 needs access to create a materialized view on a schema EDW. STG_SCHEMA. How can this access be provided?

A. GRANT CREATE MATERIALIZED VIEW ON SCHEMA EDW.STG_SCHEMA TO USER USER_01;

B. GRANT CREATE MATERIALIZED VIEW ON DATABASE EDW TO USER USERJD1;

C. GRANT ROLE NEW_ROLE TO USER USER_01;GRANT CREATE MATERIALIZED VIEW ON SCHEMA EDW.STG_SCHEMA TO ROLE NEW_ROLE;

D. GRANT ROLE NEW_ROLE TO USER_01;GRANT CREATE MATERIALIZED VIEW ON EDW.STG_SCHEMA TO NEW_ROLE;

ANSWER: C

Explanation:

`GRANT ROLE NEW_ROLE TO USER USER_01;` followed by `GRANT CREATE MATERIALIZED VIEW ON SCHEMA EDW.STG_SCHEMA TO ROLE NEW_ROLE;` is correct because Snowflake access control is role-based. Object privileges, including the schema-level `CREATE MATERIALIZED VIEW` privilege, are granted to roles; a role is then assigned to the user who needs to exercise that privilege. With NEW_ROLE active, USER_01 can create materialized views in EDW.STG_SCHEMA, subject to having the other access needed for the operation, such as USAGE on the containing database and schema and SELECT access to the source objects used in the materialized-view definition. In a production implementation, these supporting privileges can be assigned through the same role or through another role that is active in the user's session. This approach follows least-privilege design by limiting the create capability to the specific schema rather than extending it more broadly. Snowflake documents `CREATE MATERIALIZED VIEW` as a schema privilege and documents the supported role-grant pattern in its privilege-grant syntax. See GRANT <privileges> and Working with Materialized Views.

QUESTION NO: 4

Why might a Snowflake Architect use a star schema model rather than a 3NF model when designing a data architecture to run in Snowflake? (Select TWO).

A. Snowflake cannot handle the joins implied in a 3NF data model.

B. The Architect wants to remove data duplication from the data stored in Snowflake.

C. The Architect is designing a landing zone to receive raw data into Snowflake.

D. The BI tool needs a data model that allows users to summarize facts across different dimensions, or to drill down from the summaries.

E. The Architect wants to present a simple flattened single view of the data to a particular group of end users.

ANSWER: D

Explanation:

The BI tool needs a data model that allows users to summarize facts across different dimensions, or to drill down from the summaries. A star schema is a dimensional model organized around a central fact table containing measurable business events, such as sales amount or quantity, and surrounding dimension tables that provide descriptive attributes, such as date, product, customer, and location. This structure directly supports common BI workloads: aggregating measures by selected dimensions, filtering results, comparing categories, and navigating from a high-level total into more detailed dimensional values. It also gives analysts a business-oriented structure that is generally straightforward to understand when creating

reports, dashboards, and semantic-layer definitions. Snowflake's separation of compute and storage allows warehouses to scale for these analytical query workloads, while its SQL engine can efficiently execute the joins needed between facts and dimensions. A dimensional model therefore aligns the physical data presentation with how BI users explore and summarize data. Snowflake's documentation describes how virtual warehouses provide compute resources for query execution and can be independently sized for workload needs: Virtual warehouses. For general guidance on query execution and performance considerations in Snowflake, see Using persisted query results.

QUESTION NO: 5

A user, analyst_user has been granted the analyst_role, and is deploying a SnowSQL script to run as a background service to extract data from Snowflake.

What steps should be taken to allow the IP addresses to be accessed? (Select TWO).

A. ALTERROLEANALYST_ROLESETNETWORK_POLICY='ANALYST_POLICY';

B. ALTER USER ANALYST_USER SET NETWORK_POLICY = 'ANALYST_POLICY';

C. ALTERUSERANALYST_USERSETNETWORK_POLICY='10.1.1.20';

D. USE ROLE SECURITYADMIN;CREATE OR REPLACE NETWORK POLICY ANALYST_POLICY ALLOWED_IP_LIST = ('10.1.1.20');

E. USE ROLE USERADMIN;CREATE OR REPLACE NETWORK POLICY ANALYST_POLICYALLOWED_IP_LIST = ('10.1.1.20');

ANSWER: B D

Explanation:

The correct approach is to create a named network policy containing the approved client IP address, then assign that policy directly to `analyst_user`. A Snowflake network policy defines the IP addresses or CIDR ranges from which users may connect. Creating `ANALYST_POLICY` with `ALLOWED_IP_LIST = ('10.1.1.20')` establishes the approved origin for the SnowSQL background service. The role creating the policy must have the required network-policy privileges; `SECURITYADMIN` is appropriate because it manages security-related objects and privileges.

Assigning `ANALYST_POLICY` with `ALTER USER analyst_user SET NETWORK_POLICY = 'ANALYST_POLICY'` makes the restriction applicable to that specific service user. User-level network policies take precedence over an account-level policy, allowing an organization to apply targeted connection controls for an automated extraction identity without changing access rules for every account user. The policy must exist before it is assigned, so these commands together provide both the policy definition and its user-level enforcement. Snowflake evaluates the client source address during authentication, ensuring the service can connect only when its request originates from an allowed address.

See Snowflake's network policy documentation and the ALTER USER reference for policy assignment syntax and precedence behavior.

QUESTION NO: 6

Which organization-related tasks can be performed by the ORGADMIN role? (Choose three.)

A. Changing the name of the organization

B. Creating an account

C. Viewing a list of organization accounts

D. Changing the name of an account

E. Deleting an account

F. Enabling the replication of a database

ANSWER: B C F

Explanation:

Creating an account, viewing a list of organization accounts, and enabling the replication of a database are organization-level capabilities associated with the ORGADMIN role. ORGADMIN is the role that administers the Snowflake organization across its member accounts. It can create new accounts in the organization using the `CREATE ACCOUNT` command, subject to the organization's enabled account editions and cloud-region availability. This is a central organization-management operation rather than an account-local security task.

ORGADMIN can also obtain the organization's account inventory with `SHOW ORGANIZATION ACCOUNTS`. This provides account names and properties, supporting centralized visibility into the accounts governed by the organization.

Finally, database replication is enabled at the organization level by ORGADMIN. Snowflake uses the organization-wide `SYSTEM$GLOBAL_ACCOUNT_SET_PARAMETER` function to set replication-related parameters for an account, including enabling replication. Once enabled, authorized account roles can configure replication groups and replication operations between eligible accounts. This separation ensures that cross-account replication is first approved centrally, while implementation remains managed within the participating accounts. See Snowflake's documentation for CREATE ACCOUNT and SYSTEM$GLOBAL_ACCOUNT_SET_PARAMETER.

QUESTION NO: 7

Which Snowflake data modeling approach is designed for BI queries?

A. 3 NF

B. Star schema

C. Data Vault

D. Snowflake schema

ANSWER: B

Explanation:

Star schema is designed to support BI queries because it organizes analytical data around a central fact table linked directly to descriptive dimension tables. Facts hold measurable business events, such as sales amounts or quantities, while dimensions provide the attributes used to filter, group, and report on those measures, such as date, customer, product, or region. This intuitive structure maps naturally to the query patterns used by BI tools and business users: aggregating measures and slicing results by dimensional attributes.

In Snowflake, a star schema is a dimensional-modeling pattern that makes reporting-oriented data easy to understand and query. Keeping dimensions directly associated with the fact table reduces join complexity for common analytical requests and provides a clear semantic structure for dashboards and ad hoc reporting. Snowflake's architecture can efficiently process joins and aggregations at scale, while the star layout remains valuable for usability, consistent metrics, and straightforward BI-model design. Snowflake documentation describes dimensional models as schemas containing fact and dimension tables and identifies the star schema as a common form of dimensional modeling. See the Snowflake guidance for designing a schema and the Snowflake data modeling documentation.

QUESTION NO: 8

An Architect is troubleshooting a query with poor performance using the QUERY function. The Architect observes that the COMPILATION_TIME Is greater than the EXECUTION_TIME.

What is the reason for this?

A. The query is processing a very large dataset.

B. The query has overly complex logic.

C. The query Is queued for execution.

D. The query Is reading from remote storage

ANSWER: B

Explanation:

The query has overly complex logic. Snowflake compilation is the pre-execution phase in which Snowflake parses the SQL, resolves referenced objects, applies optimizations, and generates an execution plan. This work can take longer than execution when SQL has substantial logical complexity, such as many joins, nested subqueries, numerous common table expressions, complex expressions, extensive view expansion, or a large number of referenced objects. In this situation, the optimizer needs to evaluate more possible ways to transform and execute the statement before compute resources begin processing its data. A short execution time does not contradict a longer compilation time: after Snowflake selects an efficient plan, the actual data-processing work may complete quickly, particularly when the resulting data volume is modest or the selected operations are efficient. Query-history metrics report compilation time separately from execution time, so comparing them helps identify whether tuning should focus on simplifying SQL structure and object dependencies rather than on runtime compute capacity. Snowflake documents `COMPILATION_TIME` as the time spent compiling a query in its query-history data, and its query-profile guidance describes using these timing components to investigate query behavior. See QUERY_HISTORY account usage view and Query Profile.

QUESTION NO: 9

An Architect on a new project has been asked to design an architecture that meets Snowflake security, compliance, and governance requirements as follows:

1) Use Tri-Secret Secure in Snowflake

2) Share some information stored in a view with another Snowflake customer

3) Hide portions of sensitive information from some columns

4) Use zero-copy cloning to refresh the non-production environment from the production environment

To meet these requirements, which design elements must be implemented? (Choose three.)

A. Define row access policies.

B. Use the Business-Critical edition of Snowflake.

C. Create a secure view.

D. Use the Enterprise edition of Snowflake.

E. Use Dynamic Data Masking.

F. Create a materialized view.

ANSWER: B C E

Explanation:

Use the Business-Critical edition of Snowflake is required because Tri-Secret Secure is a Business Critical Edition feature. Tri-Secret Secure combines a Snowflake-managed key with a customer-managed key in the customer's cloud key-management service to protect Snowflake account data using a composite master key. This provides the customer-managed encryption-key component required by the stated security design.

Create a secure view is required when sharing data exposed through a view with another Snowflake customer. Snowflake requires views shared through a Secure Data Sharing share to be secure views. A secure view limits exposure of internal view details and supports safely exposing only the intended projected or filtered data to consumers.

Use Dynamic Data Masking is required to conceal all or part of sensitive values in selected columns at query time. A masking policy can return different representations of a column's value according to the querying role or other policy

conditions, allowing authorized users to see clear-text data while other users receive a partially or fully masked result. Zero-copy cloning is natively supported for eligible Snowflake objects and enables efficient non-production refreshes without an additional listed design element.

See Snowflake documentation for Tri-Secret Secure and Dynamic Data Masking.

QUESTION NO: 10

A company has several sites in different regions from which the company wants to ingest data.

Which of the following will enable this type of data ingestion?

A. The company must have a Snowflake account in each cloud region to be able to ingest data to that account.

B. The company must replicate data between Snowflake accounts.

C. The company should provision a reader account to each site and ingest the data through the reader accounts.

D. The company should use a storage integration for the external stage.

ANSWER: D

Explanation:

The company should use a storage integration for the external stage. A storage integration is a Snowflake object that stores a cloud-storage access configuration and lets Snowflake authenticate to an external cloud storage location without embedding long-lived cloud credentials in stages, commands, or application code. Each site can place its files in approved cloud storage locations, and an external stage can reference those locations for loading data into Snowflake with `COPY INTO`. This provides a controlled, scalable ingestion design for distributed sources while centralizing access governance in Snowflake.

For supported cloud storage providers, the integration specifies the permitted storage locations using the `STORAGE_ALLOWED_LOCATIONS` property. Snowflake administrators grant the required cloud IAM permissions to the Snowflake-generated identity, then authorize use of the integration and stage through Snowflake roles. This approach separates storage authentication from ingestion operations: users and pipelines can load files through the external stage without handling the underlying cloud credentials. Where regional network, cloud-provider, or compliance requirements apply, the storage locations and Snowflake deployment should be designed accordingly, but the external-stage storage integration is the Snowflake capability that enables this ingestion pattern. See Snowflake's storage integration documentation and external stage documentation.

QUESTION NO: 11

A data engineering team wants to validate a set of staged CSV files before loading any rows into a production table. The team needs COPY INTO to report file parsing and conversion errors without performing the load.

Which COPY INTO options can meet this requirement? (Select TWO).

A. VALIDATION_MODE = RETURN_ERRORS

B. VALIDATION_MODE = RETURN_ALL_ERRORS

C. ON_ERROR = CONTINUE

D. FORCE = TRUE

E. PURGE = TRUE

ANSWER: A B

Explanation:

VALIDATION_MODE = RETURN_ERRORS validates the files and returns errors encountered during validation without loading data into the target table. It is useful when the team needs an initial indication of whether invalid rows are present.

VALIDATION_MODE = RETURN_ALL_ERRORS also performs validation without loading data, but returns all detected errors. This is appropriate when the team needs a fuller error inventory before correcting files and running the production load.

ON_ERROR controls how a load handles errors while loading; it does not provide a validation-only execution. FORCE controls load-metadata behavior and can cause files to be loaded again, which is unsuitable when no production load should occur.

QUESTION NO: 12

An Architect has been asked to clone schema STAGING as it looked one week ago, Tuesday June 1st at 8:00 AM, to recover some objects.

The STAGING schema has 50 days of retention.

The Architect runs the following statement:

CREATE SCHEMA STAGING_CLONE CLONE STAGING at (timestamp => '2021-06-01 08:00:00');

The Architect receives the following error: Time travel data is not available for schema STAGING. The requested time is either beyond the allowed time travel period or before the object creation time.

The Architect then checks the schema history and sees the following:

CREATED_ON|NAME|DROPPED_ON

2021-06-02 23:00:00 | STAGING | NULL

2021-05-01 10:00:00 | STAGING | 2021-06-02 23:00:00

How can cloning the STAGING schema be achieved?

A. Undrop the STAGING schema and then rerun the CLONE statement.

B. Modify the statement: CREATE SCHEMA STAGING_CLONE CLONE STAGING at (timestamp => '2021-05-01 10:00:00');

C. Rename the STAGING schema and perform an UNDROP to retrieve the previous STAGING schema version, then run the CLONE statement.

D. Cloning cannot be accomplished because the STAGING schema version was not active during the proposed Time Travel time period.

ANSWER: C

Explanation:

Rename the STAGING schema and perform an UNDROP to retrieve the previous STAGING schema version, then run the CLONE statement. The currently active schema named STAGING was created at 2021-06-02 23:00:00, so it did not exist at the requested June 1 timestamp. Snowflake Time Travel queries and timestamp-based clones apply to a particular object incarnation; they cannot access a point before that currently named object was created.

The earlier STAGING incarnation was created on May 1 and dropped on June 2. Because the schema has a 50-day retention period, that dropped incarnation remains recoverable during its retention window. However, an `UNDROP SCHEMA STAGING` requires the STAGING name to be available. Renaming the current schema first frees that name, allowing the prior dropped schema to be restored with `UNDROP`. Once restored, STAGING refers to the incarnation that existed on June 1 at 8:00 AM, and `CREATE SCHEMA STAGING_CLONE CLONE STAGING AT (TIMESTAMP => '2021-06-01 08:00:00')` can create the required historical clone. Snowflake documents that UNDROP restores dropped schemas within their Time Travel retention period and that cloning supports Time Travel points for the source object. See UNDROP SCHEMA and CREATE <object> ... CLONE.

QUESTION NO: 13

A company's daily Snowflake workload consists of a huge number of concurrent queries triggered between 9pm and 11pm. At the individual level, these queries are smaller statements that get completed within a short time period.

What configuration can the company's Architect implement to enhance the performance of this workload? (Choose two.)

A. Enable a multi-clustered virtual warehouse in maximized mode during the workload duration.

B. Set the MAX_CONCURRENCY_LEVEL to a higher value than its default value of 8 at the virtual warehouse level.

C. Increase the size of the virtual warehouse to size X-Large.

D. Reduce the amount of data that is being processed through this workload.

E. Set the connection timeout to a higher value than its default.

ANSWER: A B

Explanation:

Enabling a multi-clustered virtual warehouse in maximized mode during the workload duration is appropriate for a predictable, time-bounded burst of high query concurrency. A multi-cluster warehouse adds compute capacity horizontally, allowing independent query workloads to be distributed across clusters rather than accumulating in a queue. With the maximized scaling policy, Snowflake starts the configured maximum number of clusters when the warehouse starts or resumes, making capacity available immediately for the known 9pm-to-11pm peak. This prioritizes low latency and throughput during the scheduled workload window; the warehouse can be resized or suspended outside that period to manage credit use. See Snowflake's multi-cluster warehouse documentation.

Setting the MAX_CONCURRENCY_LEVEL to a higher value than its default value of 8 at the virtual warehouse level can also improve throughput for a very large number of short-running statements. This parameter controls the maximum number of statements that may execute concurrently per cluster before additional statements queue. Since the individual queries are small and complete quickly, allowing more concurrent execution can better utilize available warehouse resources and reduce queueing pressure. The setting should be tuned while monitoring workload behavior and resource use, as documented for MAX_CONCURRENCY_LEVEL.

QUESTION NO: 14

A Snowpipe ingestion process has stopped making newly delivered files available in its target table. The Architect needs to determine whether the pipe is running and review the outcome of recent file load attempts.

Which Snowflake functions can be used for this investigation? (Choose two.)

A. SYSTEM$PIPE_STATUS

B. COPY_HISTORY

C. SYSTEM$STREAM_HAS_DATA

D. SYSTEM$CLUSTERING_INFORMATION

E. SYSTEM$ALLOWLIST

ANSWER: A B

Explanation:

SYSTEM$PIPE_STATUS returns operational status information for a pipe, including whether the pipe is running and information related to pending files and execution. This is appropriate for determining whether the pipe is functioning as expected.

The COPY_HISTORY table function returns recent load-history information, including file-level load status and error details. It can be used to review whether recently delivered files were loaded, skipped, or failed.

SYSTEM$STREAM_HAS_DATA evaluates whether a stream contains change data and does not report pipe state. SYSTEM$CLUSTERING_INFORMATION evaluates table clustering metadata, which is unrelated to Snowpipe execution and file-load status.

QUESTION NO: 15

What are characteristics of Dynamic Data Masking? (Select TWO).

A. A masking policy currently set on a table can be unset.

B. A single masking policy can be applied to columns in different tables.

C. A masking policy can be applied to the value column of an external table.

D. The role that creates the masking policy will always see unmasked data In query results

E. A masking policy can be applied to a column with the GEOGRAPHY data type.

ANSWER: A B

Explanation:

Dynamic Data Masking is implemented through masking policies that are assigned to table columns. An assigned policy is not permanent: the policy association can be removed from a column by using `ALTER TABLE` with the `UNSET MASKING POLICY` clause. This enables administrators to change or retire column-level masking behavior without recreating the table or its data. If the masking-policy object itself is to be dropped, Snowflake requires that its existing assignments first be removed.

A masking policy is reusable. After it is created, the same policy can be assigned to compatible columns in multiple tables, which supports consistent protection rules for the same category of sensitive data throughout a Snowflake environment. The policy signature determines compatibility: its input and return data types must be appropriate for each protected column. At query time, Snowflake evaluates the policy dynamically in the context of the querying role, so the same stored value can be returned differently for different users or roles. See Snowflake's Dynamic Data Masking documentation and the ALTER TABLE reference.

QUESTION NO: 16

When using the copy into

command with the CSV file format, how does the match_by_column_name parameter behave?

Options:
A.

It expects a header to be present in the CSV file, which is matched to a case-sensitive table column name.

B.

The parameter will be ignored.

C.

The command will return an error.

D.

The command will return a warning stating that the file has unmatched columns.

Show Answer
Answer:
B
Explanation:
Explanation:

Option B is the best design to meet the requirements because it uses Snowpipe to ingest the data continuously and efficiently as new records arrive in the object storage, leveraging event notifications. Snowpipe is a service that automates the loading of data from external sources into Snowflake tables1. It also uses streams and tasks to orchestrate transformations on the ingested data. Streams are objects that store the change history of a table, and tasks are objects that execute SQL statements on a schedule or when triggered by another task2. Option B also uses an external function to do model inference with Amazon Comprehend and write the final records to a Snowflake table. An external function is a user-defined function that calls an external API, such as Amazon Comprehend, to perform computations that are not natively supported by Snowflake3. Finally, option B uses the Snowflake Marketplace to make the de-identified final data set available publicly for advertising companies who use different cloud providers in different regions. The Snowflake Marketplace is a platform that enables data providers to list and share their data sets with data consumers, regardless of the cloud platform or region they use4.

Option A is not the best design because it uses copy into to ingest the data, which is not as efficient and continuous as Snowpipe. Copy into is a SQL command that loads data from files into a table in a single transaction. It also exports the data into Amazon S3 to do model inference with Amazon Comprehend, which adds an extra step and increases the operational complexity and maintenance of the infrastructure.

Option C is not the best design because it uses Amazon EMR and PySpark to ingest and transform the data, which also increases the operational complexity and maintenance of the infrastructure. Amazon EMR is a cloud service that provides a managed Hadoop framework to process and analyze large-scale data sets. PySpark is a Python API for Spark, a distributed computing framework that can run on Hadoop. Option C also develops a python program to do model inference by leveraging the Amazon Comprehend text analysis API, which increases the development effort.

Option D is not the best design because it is identical to option A, except for the ingestion method. It still exports the data into Amazon S3 to do model inference with Amazon Comprehend, which adds an extra step and increases the operational complexity and maintenance of the infrastructure.

References: 1: Snowpipe Overview 2: Using Streams and Tasks to Automate Data Pipelines 3: External Functions Overview 4: Snowflake Data Marketplace Overview : [Loading Data Using COPY INTO] : [What is Amazon EMR?] : [PySpark Overview]

The copy into
command is used to load data from staged files into an existing table in Snowflake. The command supports various file formats, such as CSV, JSON, AVRO, ORC, PARQUET, and XML1.
The match_by_column_name parameter is a copy option that enables loading semi-structured data into separate columns in the target table that match corresponding columns represented in the source data. The parameter can have one of the following values2:
The match_by_column_name parameter only applies to semi-structured data, such as JSON, AVRO, ORC, PARQUET, and XML. It does not apply to CSV data, which is considered structured data2.
When using the copy into
command with the CSV file format, the match_by_column_name parameter behaves as follows2:

References:

1: COPY INTO
| Snowflake Documentation
2: MATCH_BY_COLUMN_NAME | Snowflake Documentation

Questions 40

Which system functions does Snowflake provide to monitor clustering information within a table (Choose two.)

Options:
A.

SYSTEM$CLUSTERING_INFORMATION

B.

SYSTEM$CLUSTERING_USAGE

C.

SYSTEM$CLUSTERING_DEPTH

D.

SYSTEM$CLUSTERING_KEYS

E.

SYSTEM$CLUSTERING_PERCENT

Show Answer
Answer:
A, C
Explanation:
Explanation:

According to the Snowflake documentation, these two system functions are provided by Snowflake to monitor clustering information within a table. A system function is a type of function that allows executing actions or returning information about the system. A clustering key is a feature that allows organizing data across micro-partitions based on one or more columns in the table. Clustering can improve query performance by reducing the number of files to scan.

SYSTEM$CLUSTERING_INFORMATION is a system function that returns clustering information, including average clustering depth, for a table based on one or more columns in the table. The function takes a table name and an optional column name or expression as arguments, and returns a JSON string with the clustering information. The clustering information includes the cluster by keys, the total partition count, the total constant partition count, the average overlaps, and the average depth1.
SYSTEM$CLUSTERING_DEPTH is a system function that returns the clustering depth for a table based on one or more columns in the table. The function takes a table name and an optional column name or expression as arguments, and returns an integer value with the clustering depth. The clustering depth is the maximum number of overlapping micro-partitions for any micro-partition in the table. A lower clustering depth indicates a better clustering2.

References:

SYSTEM$CLUSTERING_INFORMATION | Snowflake Documentation
SYSTEM$CLUSTERING_DEPTH | Snowflake Documentation

Exam Code: ARA-R01
Certification Provider: Snowflake
Exam Name: SnowPro Advanced: Architect Recertification Exam
Last Update: Aug 8, 2026
Questions: 162

How to Pass ARA-R01 Exams

Snowflake ARA-R01 Exam Dumps Set 1
Snowflake ARA-R01 Exam Dumps Set 2
Snowflake ARA-R01 Exam Dumps Set 3
Snowflake ARA-R01 Exam Dumps Set 4
Snowflake ARA-R01 Exam Dumps Set 5
Snowflake ARA-R01 Premium Access

PDF + Testing Engine

~~$164.99~~
$49.5
Add to Cart

Testing Engine

~~$124.99~~
$37.5
Add to Cart

PDF (Q&A)

~~$104.99~~
$31.5
Add to Cart

A. It expects a header to be present in the CSV file, which is matched to a case-sensitive table column name.

B. The parameter will be ignored.

C. The command will return an error.

D. The command will return a warning stating that the file has unmatched columns.

E. SYSTEM$CLUSTERING_PERCENT

ANSWER: C

Explanation:

The command will return an error. `MATCH_BY_COLUMN_NAME` is a COPY option designed to map source fields to target-table columns by name for supported semi-structured and columnar formats. CSV is a positional, delimited text format, so Snowflake does not support this option when the source file format is CSV. Consequently, specifying `MATCH_BY_COLUMN_NAME` in a `COPY INTO` statement loading CSV data is invalid rather than being silently ignored or producing only a warning. For CSV loads, column mapping is ordinarily determined by the positional order of fields in each record relative to the target table columns. If a load requires a different mapping, the pipeline should explicitly transform or reorder the staged data through a SELECT-based load pattern or use an appropriate supported source format and loading design. Snowflake documents the allowed values and format-specific restrictions for this COPY option in its COPY INTO <table> reference. The documented CSV restriction is also covered in the CREATE FILE FORMAT reference, which describes CSV format behavior and available file-format controls.

QUESTION NO: 17

Which of the following are characteristics of Snowflake's parameter hierarchy?

A. Session parameters override virtual warehouse parameters.

B. Virtual warehouse parameters override user parameters.

C. Table parameters override virtual warehouse parameters.

D. Schema parameters override account parameters.

ANSWER: B

Explanation:

Virtual warehouse parameters override user parameters when the same supported parameter is configured at both levels. This enables an administrator to apply settings appropriate to the workload executed by a particular warehouse, rather than relying solely on an individual user's default configuration. For example, a warehouse can carry operational settings that establish consistent behavior for all work routed to that compute resource. Snowflake evaluates parameter values according to the parameter's supported scopes and its documented precedence rules; a value configured on the warehouse therefore takes precedence over a user-level value where warehouse-level configuration is supported. This workload-oriented control is useful in architecture designs because warehouses commonly separate reporting, transformation, data-science, or other workloads with distinct execution and governance requirements. Snowflake documents warehouse parameters as object-level settings and documents the parameter hierarchy and scope rules used to determine the effective value. See the Snowflake Parameters reference and ALTER WAREHOUSE documentation.

QUESTION NO: 18

A Snowflake Architect is designing a multi-tenant application strategy for an organization in the Snowflake Data Cloud and is considering using an Account Per Tenant strategy.

Which requirements will be addressed with this approach? (Choose two.)

A. There needs to be fewer objects per tenant.

B. Security and Role-Based Access Control (RBAC) policies must be simple to configure.

C. Compute costs must be optimized.

D. Tenant data shape may be unique per tenant.

E. Storage costs must be optimized.

ANSWER: B D

Explanation:

An Account Per Tenant model addresses the requirement that **Security and Role-Based Access Control (RBAC) policies must be simple to configure**. Each tenant is provisioned in a distinct Snowflake account, creating an account-level isolation boundary. Roles, users, authentication integrations, network policies, warehouses, databases, and grants can be administered within that tenant's own account rather than requiring a shared account to implement and maintain highly granular tenant-separation rules. This structure makes it easier to delegate administration and apply tenant-specific governance controls.

This model also addresses the requirement that **Tenant data shape may be unique per tenant**. Because a tenant has its own account and independently managed database objects, its tables, schemas, views, pipelines, retention settings, and application-specific object design can evolve without imposing a common schema or release cadence on other tenants. This is useful when customers require different data models, custom extensions, or independently managed environments. Snowflake identifies account-per-tenant as a multitenancy pattern that provides strong isolation and supports tenant-specific configuration. See Snowflake's multitenancy overview and its access control overview for the account and RBAC concepts underlying this design.