Product Questions: 162
Version: 8.1

Question: 1

What built-in Snowflake features make use of the change tracking metadata for a table? (Choose two.)
A. The MERGE command
B. The UPSERT command
C. The CHANGES clause
D. A STREAM object
E. The CHANGE_DATA_CAPTURE command

Answer: A, D

Explanation:
In Snowflake, the change tracking metadata for a table is utilized by the MERGE command and the STREAM object. The MERGE command uses change tracking to determine how to apply updates and inserts efficiently based on differences between source and target tables. STREAM objects, on the other hand, specifically capture and store change data, enabling incremental processing based on changes made to a table since the last stream offset was committed.
Reference: Snowflake Documentation on MERGE and STREAM Objects.

Question: 2

When using the Snowflake Connector for Kafka, what data formats are supported for the messages? (Choose two.)
A. CSV
B. XML
C. Avro
D. JSON
E. Parquet

Answer: C, D

Explanation:
The data formats that are supported for the messages when using the Snowflake Connector for Kafka are Avro and JSON. These are the two formats that the connector can parse and convert into Snowflake table rows. The connector supports both schemaless and schematized JSON, as well as Avro with or without a schema registry1. The other options are incorrect because they are not supported data formats for the messages. CSV, XML, and Parquet are not formats that the connector can parse and convert into Snowflake table rows. If the messages are in these formats, the connector will load them as VARIANT data type and store them as raw strings in the table2. Reference: Snowflake Connector for Kafka | Snowflake Documentation, Loading Protobuf Data using the Snowflake Connector for Kafka | Snowflake Documentation

Question: 3

At which object type level can the APPLY MASKING POLICY, APPLY ROW ACCESS POLICY and APPLY SESSION POLICY privileges be granted?
A. Global
B. Database
C. Schema
D. Table

Answer: A

Explanation:
The object type level at which the APPLY MASKING POLICY, APPLY ROW ACCESS POLICY and APPLY SESSION POLICY privileges can be granted is global. These are account-level privileges that control who can apply or unset these policies on objects such as columns, tables, views, accounts, or users. These privileges are granted to the ACCOUNTADMIN role by default, and can be granted to other roles as needed. The other options are incorrect because they are not the object type level at which these privileges can be granted. Database, schema, and table are lower-level object types that do not support these privileges. Reference: Access Control Privileges | Snowflake Documentation, Using Dynamic Data Masking | Snowflake Documentation, Using Row Access Policies | Snowflake Documentation, Using Session Policies | Snowflake Documentation

Question: 4

An Architect uses COPY INTO with the ON_ERROR=SKIP_FILE option to bulk load CSV files into a table called TABLEA, using its table stage. One file named file5.csv fails to load. The Architect fixes the file and re-loads it to the stage with the exact same file name it had previously.
Which commands should the Architect use to load only file5.csv file from the stage? (Choose two.)
A. COPY INTO tablea FROM @%tablea RETURN_FAILED_ONLY = TRUE;
B. COPY INTO tablea FROM @%tablea;
C. COPY INTO tablea FROM @%tablea FILES = ('file5.csv');
D. COPY INTO tablea FROM @%tablea FORCE = TRUE;
E. COPY INTO tablea FROM @%tablea NEW_FILES_ONLY = TRUE;
F. COPY INTO tablea FROM @%tablea MERGE = TRUE;

Answer: BC

Explanation:
Option A (RETURN_FAILED_ONLY) will only load files that previously failed to load. Since file5.csv already exists in the stage with the same name, it will not be considered a new file and will not be loaded.
Option D (FORCE) will overwrite any existing data in the table. This is not desired as we only want to load the data from file5.csv.
Option E (NEW_FILES_ONLY) will only load files that have been added to the stage since the last COPY command. This will not work because file5.csv was already in the stage before it was fixed.
Option F (MERGE) is used to merge data from a stage into an existing table, creating new rows for any data not already present. This is not needed in this case as we simply want to load the data from file5.csv.
Therefore, the architect can use either COPY INTO tablea FROM @%tablea or COPY INTO tablea FROM @%tablea FILES = ('file5.csv') to load only file5.csv from the stage. Both options will load the data from the specified file without overwriting any existing data or requiring additional configuration

Question: 5

A large manufacturing company runs a dozen individual Snowflake accounts across its business divisions. The company wants to increase the level of data sharing to support supply chain optimizations and increase its purchasing leverage with multiple vendors.
The company's Snowflake Architects need to design a solution that would allow the business divisions to decide what to share, while minimizing the level of effort spent on configuration and management. Most of the company divisions use Snowflake accounts in the same cloud deployments with a few exceptions for European-based divisions.
According to Snowflake recommended best practice, how should these requirements be met?
A. Migrate the European accounts in the global region and manage shares in a connected graph architecture. Deploy a Data Exchange.
B. Deploy a Private Data Exchange in combination with data shares for the European accounts.
C. Deploy to the Snowflake Marketplace making sure that invoker_share() is used in all secure views.
D. Deploy a Private Data Exchange and use replication to allow European data shares in the Exchange.

Answer: D

Explanation:
According to Snowflake recommended best practice, the requirements of the large manufacturing company should be met by deploying a Private Data Exchange in combination with data shares for the European accounts. A Private Data Exchange is a feature of the Snowflake Data Cloud platform that enables secure and governed sharing of data between organizations. It allows Snowflake customers to create their own data hub and invite other parts of their organization or external partners to access and contribute data sets. A Private Data Exchange provides centralized management, granular access control, and data usage metrics for the data shared in the exchange1.
A data share is a secure and direct way of sharing data between Snowflake accounts without having to copy or move the data. A data share allows the data provider to grant privileges on selected objects in their account to one or more data consumers in other accounts2. By using a Private Data Exchange in combination with data shares, the company can achieve the following benefits:
The business divisions can decide what data to share and publish it to the Private Data Exchange, where it can be discovered and accessed by other members of the exchange. This reduces the effort and complexity of managing multiple data sharing relationships and configurations.
The company can leverage the existing Snowflake accounts in the same cloud deployments to create the Private Data Exchange and invite the members to join. This minimizes the migration and setup costs and leverages the existing Snowflake features and security.
The company can use data shares to share data with the European accounts that are in different regions or cloud platforms. This allows the company to comply with the regional and regulatory requirements for data sovereignty and privacy, while still enabling data collaboration across the organization.
The company can use the Snowflake Data Cloud platform to perform data analysis and transformation on the shared data, as well as integrate with other data sources and applications. This enables the company to optimize its supply chain and increase its purchasing leverage with multiple vendors.

Question: 6

A user has the appropriate privilege to see unmasked data in a column.
If the user loads this column data into another column that does not have a masking policy, what will

occur?
A. Unmasked data will be loaded in the new column.
B. Masked data will be loaded into the new column.
C. Unmasked data will be loaded into the new column but only users with the appropriate privileges will be able to see the unmasked data.
D. Unmasked data will be loaded into the new column and no users will be able to see the unmasked data.

Answer: A

Explanation:
According to the SnowPro Advanced: Architect documents and learning resources, column masking policies are applied at query time based on the privileges of the user who runs the query. Therefore, if a user has the privilege to see unmasked data in a column, they will see the original data when they query that column. If they load this column data into another column that does not have a masking policy, the unmasked data will be loaded in the new column, and any user who can query the new column will see the unmasked data as well. The masking policy does not affect the underlying data in the column, only the query results.
Reference:
Snowflake Documentation: Column Masking
Snowflake Learning: Column Masking

Question: 7

How can an Architect enable optimal clustering to enhance performance for different access paths on a given table?
A. Create multiple clustering keys for a table.
B. Create multiple materialized views with different cluster keys.
C. Create super projections that will automatically create clustering.
D. Create a clustering key that contains all columns used in the access paths.

Answer: B

Explanation:
According to the SnowPro Advanced: Architect documents and learning resources, the best way to enable optimal clustering to enhance performance for different access paths on a given table is to create multiple materialized views with different cluster keys. A materialized view is a pre-computed result set that is derived from a query on one or more base tables. A materialized view can be clustered by specifying a clustering key, which is a subset of columns or expressions that determines how the data in the materialized view is co-located in micro-partitions. By creating multiple materialized views with different cluster keys, an Architect can optimize the performance of queries that use different access paths on the same base table. For example, if a base table has columns A, B, C, and D, and there are queries that filter on A and B, or on C and D, or on A and C, the Architect can create three materialized views, each with a different cluster key: (A, B), (C, D), and (A, C). This way, each query can leverage the optimal clustering of the corresponding materialized view and achieve faster scan efficiency and better compression.
Reference:
Snowflake Documentation: Materialized Views
Snowflake Learning: Materialized Views
https://www.snowflake.com/blog/using-materialized-views-to-solve-multi-clustering-performance-problems/

Question: 8

Company A would like to share data in Snowflake with Company B. Company B is not on the same cloud platform as Company A.
What is required to allow data sharing between these two companies?
A. Create a pipeline to write shared data to a cloud storage location in the target cloud provider.
B. Ensure that all views are persisted, as views cannot be shared across cloud platforms.
C. Setup data replication to the region and cloud platform where the consumer resides.
D. Company A and Company B must agree to use a single cloud platform: Data sharing is only possible if the companies share the same cloud provider.

Answer: C

Explanation:
According to the SnowPro Advanced: Architect documents and learning resources, the requirement to allow data sharing between two companies that are not on the same cloud platform is to set up data replication to the region and cloud platform where the consumer resides. Data replication is a feature of Snowflake that enables copying databases across accounts in different regions and cloud platforms. Data replication allows data providers to securely share data with data consumers across different regions and cloud platforms by creating a replica database in the consumer's account. The replica database is read-only and automatically synchronized with the primary database in the provider's account. Data replication is useful for scenarios where data sharing is not possible or desirable due to latency, compliance, or security reasons1. The other options are incorrect because they are not required or feasible to allow data sharing between two companies that are not on the same cloud platform. Option A is incorrect because creating a pipeline to write shared data to a cloud storage location in the target cloud provider is not a secure or efficient way of sharing data. It would require additional steps to load the data from the cloud storage to the consumer's account, and it would not leverage the benefits of Snowflake's data sharing features. Option B is incorrect because ensuring that all views are persisted is not relevant for data sharing across cloud platforms. Views can be shared across cloud platforms as long as they reference objects in the same database. Persisting views is an option to improve the performance of querying views, but it is not required for data sharing2. Option D is incorrect because Company A and Company B do not need to agree to use a single cloud platform. Data sharing is possible across different cloud platforms using data replication or other methods, such as listings or auto-fulfillment3. Reference: Replicating Databases Across Multiple Accounts | Snowflake Documentation, Persisting Views | Snowflake Documentation, Sharing Data Across Regions and Cloud Platforms | Snowflake Documentation

Question: 9

What are some of the characteristics of result set caches? (Choose three.)
A. Time Travel queries can be executed against the result set cache.
B. Snowflake persists the data results for 24 hours.
C. Each time persisted results for a query are used, a 24-hour retention period is reset.
D. The data stored in the result cache will contribute to storage costs.
E. The retention period can be reset for a maximum of 31 days.
F. The result set cache is not shared between warehouses.

Answer: B, C, F

Explanation:

In Snowflake, the characteristics of result set caches include persistence of data results for 24 hours (B), each use of persisted results resets the 24-hour retention period (C), and result set caches are

not shared between different warehouses (F). The result set cache is specifically designed to avoid repeated execution of the same query within this timeframe, reducing computational overhead and speeding up query responses. These caches do not contribute to storage costs, and their retention period cannot be extended beyond the default duration nor up to 31 days, as might be misconstrued. Reference: Snowflake Documentation on Result Set Caching.

Question: 10

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:
According to the SnowPro Advanced: Architect documents and learning resources, the organization-related tasks that can be performed by the ORGADMIN role are:
Creating an account in the organization. A user with the ORGADMIN role can use the CREATE ACCOUNT command to create a new account that belongs to the same organization as the current account1.
Viewing a list of organization accounts. A user with the ORGADMIN role can use the SHOW ORGANIZATION ACCOUNTS command to view the names and properties of all accounts in the organization2. Alternatively, the user can use the Admin » Accounts page in the web interface to view the organization name and account names3.
Enabling the replication of a database. A user with the ORGADMIN role can use the SYSTEM$GLOBAL_ACCOUNT_SET_PARAMETER function to enable database replication for an account in the organization. This allows the user to replicate databases across accounts in different regions and cloud platforms for data availability and durability4.
The other options are incorrect because they are not organization-related tasks that can be performed by the ORGADMIN role. Option A is incorrect because changing the name of the organization is not a task that can be performed by the ORGADMIN role. To change the name of an organization, the user must contact Snowflake Support3. Option D is incorrect because changing the name of an account is not a task that can be performed by the ORGADMIN role. To change the name of an account, the user must contact Snowflake Support5. Option E is incorrect because deleting an account is not a task that can be performed by the ORGADMIN role. To delete an account, the user must contact Snowflake Support. Reference: CREATE ACCOUNT | Snowflake Documentation, SHOW ORGANIZATION ACCOUNTS | Snowflake Documentation, Getting Started with Organizations | Snowflake Documentation, SYSTEM$GLOBAL_ACCOUNT_SET_PARAMETER | Snowflake Documentation, ALTER ACCOUNT | Snowflake Documentation, [DROP ACCOUNT | Snowflake Documentation]

Question: 11

A Data Engineer is designing a near real-time ingestion pipeline for a retail company to ingest event logs into Snowflake to derive insights. A Snowflake Architect is asked to define security best practices to configure access control privileges for the data load for auto-ingest to Snowpipe.
What are the MINIMUM object privileges required for the Snowpipe user to execute Snowpipe?

A. OWNERSHIP on the named pipe, USAGE on the named stage, target database, and schema, and INSERT and SELECT on the target table
B. OWNERSHIP on the named pipe, USAGE and READ on the named stage, USAGE on the target database and schema, and INSERT end SELECT on the target table
C. CREATE on the named pipe, USAGE and READ on the named stage, USAGE on the target database and schema, and INSERT end SELECT on the target table
D. USAGE on the named pipe, named stage, target database, and schema, and INSERT and SELECT on the target table

Answer: B

Explanation:
According to the SnowPro Advanced: Architect documents and learning resources, the minimum object privileges required for the Snowpipe user to execute Snowpipe are:
OWNERSHIP on the named pipe. This privilege allows the Snowpipe user to create, modify, and drop the pipe object that defines the COPY statement for loading data from the stage to the table1.
USAGE and READ on the named stage. These privileges allow the Snowpipe user to access and read the data files from the stage that are loaded by Snowpipe2.
USAGE on the target database and schema. These privileges allow the Snowpipe user to access the database and schema that contain the target table3.
INSERT and SELECT on the target table. These privileges allow the Snowpipe user to insert data into the table and select data from the table4.
The other options are incorrect because they do not specify the minimum object privileges required for the Snowpipe user to execute Snowpipe. Option A is incorrect because it does not include the READ privilege on the named stage, which is required for the Snowpipe user to read the data files from the stage. Option C is incorrect because it does not include the OWNERSHIP privilege on the named pipe, which is required for the Snowpipe user to create, modify, and drop the pipe object. Option D is incorrect because it does not include the OWNERSHIP privilege on the named pipe or the READ privilege on the named stage, which are both required for the Snowpipe user to execute Snowpipe. Reference: CREATE PIPE | Snowflake Documentation, CREATE STAGE | Snowflake Documentation, CREATE DATABASE | Snowflake Documentation, CREATE TABLE | Snowflake Documentation

Question: 12

The IT Security team has identified that there is an ongoing credential stuffing attack on many of their organization's system.
What is the BEST way to find recent and ongoing login attempts to Snowflake?
A. Call the LOGIN_HISTORY Information Schema table function.
B. Query the LOGIN_HISTORY view in the ACCOUNT_USAGE schema in the SNOWFLAKE database.
C. View the History tab in the Snowflake UI and set up a filter for SQL text that contains the text "LOGIN".
D. View the Users section in the Account tab in the Snowflake UI and review the last login column.

Answer: B

Explanation:
This view can be used to query login attempts by Snowflake users within the last 365 days (1 year). It provides information such as the event timestamp, the user name, the client IP, the authentication method, the success or failure status, and the error code or message if the login attempt was unsuccessful. By querying this view, the IT Security team can identify any suspicious or malicious login attempts to Snowflake and take appropriate actions to prevent credential stuffing attacks1. The

other options are not the best ways to find recent and ongoing login attempts to Snowflake. Option A is incorrect because the LOGIN_HISTORY Information Schema table function only returns login events within the last 7 days, which may not be sufficient to detect credential stuffing attacks that span a longer period of time2. Option C is incorrect because the History tab in the Snowflake UI only shows the queries executed by the current user or role, not the login events of other users or roles3. Option D is incorrect because the Users section in the Account tab in the Snowflake UI only shows the last login time for each user, not the details of the login attempts or the failures.

Question: 13

An Architect has a VPN_ACCESS_LOGS table in the SECURITY_LOGS schema containing timestamps of the connection and disconnection, username of the user, and summary statistics.
What should the Architect do to enable the Snowflake search optimization service on this table?
A. Assume role with OWNERSHIP on future tables and ADD SEARCH OPTIMIZATION on the SECURITY_LOGS schema.
B. Assume role with ALL PRIVILEGES including ADD SEARCH OPTIMIZATION in the SECURITY LOGS schema.
C. Assume role with OWNERSHIP on VPN_ACCESS_LOGS and ADD SEARCH OPTIMIZATION in the SECURITY_LOGS schema.
D. Assume role with ALL PRIVILEGES on VPN_ACCESS_LOGS and ADD SEARCH OPTIMIZATION in the SECURITY_LOGS schema.

Answer: C

Explanation:
According to the SnowPro Advanced: Architect Exam Study Guide, to enable the search optimization service on a table, the user must have the ADD SEARCH OPTIMIZATION privilege on the table and the schema. The privilege can be granted explicitly or inherited from a higher-level object, such as a database or a role. The OWNERSHIP privilege on a table implies the ADD SEARCH OPTIMIZATION privilege, so the user who owns the table can enable the search optimization service on it. Therefore, the correct answer is to assume a role with OWNERSHIP on VPN_ACCESS_LOGS and ADD SEARCH OPTIMIZATION in the SECURITY_LOGS schema. This will allow the user to enable the search optimization service on the VPN_ACCESS_LOGS table and any future tables created in the SECURITY_LOGS schema. The other options are incorrect because they either grant excessive privileges or do not grant the required privileges on the table or the schema. Reference:
SnowPro Advanced: Architect Exam Study Guide, page 11, section 2.3.1
Snowflake Documentation: Enabling the Search Optimization Service

Question: 14

A table contains five columns and it has millions of records. The cardinality distribution of the columns is shown below:

Column	Number of Distinct Values
C1	10,790
C2	108
C3	302,605
C4	1.117,736
C5	2.205,400

Column C4 and C5 are mostly used by SELECT queries in the GROUP BY and ORDER BY clauses.

Whereas columns C1, C2 and C3 are heavily used in filter and join conditions of SELECT queries.
The Architect must design a clustering key for this table to improve the query performance.
Based on Snowflake recommendations, how should the clustering key columns be ordered while defining the multi-column clustering key?
A. C5, C4, C2
B. C3, C4, C5
C. C1, C3, C2
D. C2, C1, C3

Answer: D

Explanation:
According to the Snowflake documentation, the following are some considerations for choosing clustering for a table1:
Clustering is optimal when either:
You require the fastest possible response times, regardless of cost.
Your improved query performance offsets the credits required to cluster and maintain the table.
Clustering is most effective when the clustering key is used in the following types of query predicates:
Filter predicates (e.g. WHERE clauses)
Join predicates (e.g. ON clauses)
Grouping predicates (e.g. GROUP BY clauses)
Sorting predicates (e.g. ORDER BY clauses)
Clustering is less effective when the clustering key is not used in any of the above query predicates, or when the clustering key is used in a predicate that requires a function or expression to be applied to the key (e.g. DATE_TRUNC, TO_CHAR, etc.).
For most tables, Snowflake recommends a maximum of 3 or 4 columns (or expressions) per key.
Adding more than 3-4 columns tends to increase costs more than benefits.
Based on these considerations, the best option for the clustering key columns is C. C1, C3, C2, because:
These columns are heavily used in filter and join conditions of SELECT queries, which are the most effective types of predicates for clustering.
These columns have high cardinality, which means they have many distinct values and can help reduce the clustering skew and improve the compression ratio.
These columns are likely to be correlated with each other, which means they can help co-locate similar rows in the same micro-partitions and improve the scan efficiency.
These columns do not require any functions or expressions to be applied to them, which means they can be directly used in the predicates without affecting the clustering.
Reference: 1: Considerations for Choosing Clustering for a Table | Snowflake Documentation

Question: 15

Which security, governance, and data protection features require, at a MINIMUM, the Business Critical edition of Snowflake? (Choose two.)
A. Extended Time Travel (up to 90 days)
B. Customer-managed encryption keys through Tri-Secret Secure
C. Periodic rekeying of encrypted data
D. AWS, Azure, or Google Cloud private connectivity to Snowflake
E. Federated authentication and SSO

Answer: B, D