



Vendor: Snowflake

Exam Code: SOL-C01

Exam Name: Snowflake Certified SnowPro Associate -
Platform Certification

Version: DEMO

QUESTION 1

You have a Python script running in a Snowflake Notebook that retrieves data from a Snowflake table, performs some complex calculations, and then visualizes the results using Matplotlib. The script is running slowly, even after optimizing the SQL query. Which of the following steps would MOST likely improve the performance of the Python script within the Snowflake Notebook environment?

- A. Increase the size of the virtual warehouse associated with the Snowflake session.
- B. Vectorize the calculations using NumPy instead of looping through the data row by row.
- C. Use the '%osql' magic command to execute the calculations directly in Snowflake SQL.
- D. Store the intermediate results in a Snowflake temporary table and retrieve them later.
- E. Use a smaller data sample by adding 'LIMIT 1 00' in the SQL query to speed up the process.

Answer: B

Explanation:

Option B, vectorizing calculations using NumPy, is the most likely to improve performance. Python loops are generally slower than vectorized operations performed by NumPy. Vectorization allows NumPy to perform calculations on entire arrays at once, significantly speeding up the process. Increasing the warehouse size (A) primarily improves SQL query performance, not Python code execution. Using (C) would offload the calculation to SQL, which might be faster if the calculation can be expressed efficiently in SQL, but the question states that the SQL query has already been optimized. Storing intermediate results in a temporary table (D) might be helpful in some cases, but doesn't directly address the slow Python calculations. Option E reduces the data to improve performance, but it doesn't solve the underlying issue of slow python calculation.

QUESTION 2

You are tasked with creating a Snowflake Notebook that reads data from an external stage (AWS S3), transforms it using Python, and then loads the transformed data into a Snowflake table. The external stage requires authentication using AWS IAM roles. Which of the following steps are necessary to configure the connection and access the data from the external stage within your Snowflake Notebook using Python? (Choose all that apply.)

- A. Create a Snowflake storage integration object that defines the AWS IAM role ARN and the S3 bucket URL.
- B. Grant the necessary privileges (e.g., USAGE, READ) on the storage integration object to the Snowflake role used by the Notebook session.
- C. Configure the AWS IAM role to trust the Snowflake account and grant it access to the S3 bucket.
- D. Use the AWS SDK for Python (bot03) within the Snowflake Notebook to assume the IAM role and access the S3 bucket directly.
- E. Create a Snowflake external table that points to the S3 bucket and uses the storage integration.

Answer: ABCE

Explanation:

Options A, B, C, and E are essential. To access data in an external stage secured with IAM roles, you need a Snowflake storage integration object (A) that defines the connection to S3. You must grant privileges on this integration to the appropriate Snowflake role (B). The IAM role must be configured to trust Snowflake (C). Finally, you typically create an external table (E) that uses the storage integration to access the data in S3. Using bot03 directly (D) is generally not necessary and bypasses Snowflake's security and governance mechanisms for accessing external stages.

QUESTION 3

You are using a Snowflake Notebook to perform data analysis on a large dataset. As part of your

analysis, you need to create a custom

Python function that calculates a complex metric based on multiple columns in a Snowflake table.

You want to apply this function to each row of the table and store the results in a new column.

Which of the following approaches is the MOST efficient and scalable way to achieve this using Snowflake and Python?

- A. Load the entire Snowflake table into a Pandas DataFrame, apply the Python function to each row using 'DataFrame.apply()', and then write the modified DataFrame back to Snowflake.
- B. Use the '%%osql' magic command to execute a series of SQL UPDATE statements that call the Python function using a IJDF.
- C. Create a Snowflake Python User-Defined Function (UDF) that encapsulates the calculation logic and then use it in a 'SELECT statement to create a new column with the calculated values. Store the result in a new table using 'CREATE TABLE AS SELECT
- D. Iterate over the rows of the Snowflake table using the Snowflake Connector for Python, call the Python function for each row, and then use "INSERT statements to insert the calculated values into a new table.
- E. Create a stored procedure in Snowflake that runs the logic in a separate environment.

Answer: C

Explanation:

Option C, creating a Snowflake Python IJDF and using it in a 'SELECT statement within a 'CREATE TABLE AS SELECT statement, is the most efficient and scalable approach. Snowflake IJDFs allow you to execute Python code directly within the Snowflake engine, leveraging Snowflake's distributed processing capabilities. This avoids the overhead of transferring large amounts of data between Snowflake and the Python environment in the Notebook. Loading the entire table into a Pandas DataFrame (A) is not scalable for large datasets and can lead to memory issues. Using '%%osql' with 'UPDATE statements (B) would be very slow due to the row-by-row updates. Iterating over rows using the Snowflake Connector (D) is also inefficient and not scalable. Option E is incorrect because it doesn't directly use Python code from the Notebook.

QUESTION 4

You're writing a Snowflake Notebook that needs to dynamically create tables based on a list of table names stored in a Python list called 'table_names'. You want to iterate through this list and execute a CREATE TABLE statement for each table name. However, the names might include spaces and special characters. Which approach or approaches can you use to achieve this goal safely and effectively?

- ☐ Iterate through the list and execute `session.sql(f'CREATE TABLE "{table_name}" (id INT)').collect()` for each table name.
- ☐ Iterate through the list and execute `session.sql('CREATE TABLE ' + table_name.replace(' ', '_') + ' (id INT)').collect()` for each table name, replacing spaces with underscores.
- ☐ Iterate through the list and use Python's `shlex.quote()` function to escape each table name before using it in `session.sql(f'CREATE TABLE {shlex.quote(table_name)} (id INT)').collect()`
- ☐ Iterate through the list and create a DataFrame using `session.createDataFrame([], schema=[StructField('id', IntegerType())]).write.saveAsTable(table_name)`.
- ☐ Iterate through the list and execute `session.sql('CREATE TABLE ' + table_name + ' (id INT)').collect()` for each table name.

- A. Option A
- B. Option B
- C. Option C
- D. Option D
- E. Option E

Answer: AD

Explanation:

Options A and D are correct. Option A uses double quotes to safely enclose the table name within the CREATE TABLE statement. This is the standard way to handle identifiers with spaces or special characters in SQL. Option D provides a Snowpark DataFrame-centric approach that automatically handles identifier quoting when creating the table using `saveAsTable`. Option B modifies the table names by replacing spaces. Option C's reliance on shell escaping (`'shlex.quote'`) is not relevant or necessary when executing SQL within a Snowpark session. Option E is unsafe as it does not handle the case where table names have spaces or other special characters and will result in SQL errors.

QUESTION 5

A data engineer needs to create a new database named `SALES DATA` in Snowflake. They want to ensure that only users with the `'ACCOUNTADMIN'` role can initially modify the database. After creation, they will grant specific privileges to other roles. Which of the following SQL statements is the most appropriate way to create the database with these requirements?

- A. `CREATE DATABASE SALES DATA;`
- B. `CREATE OR REPLACE DATABASE SALES DATA;`
- C. `CREATE DATABASE SALES DATA WITH DATA RETENTION TIME IN DAYS-I;`
- D. `CREATE DATABASE SALES DATA WITH OWNER = 'ACCOUNTADMIN';`
- E. `CREATE DATABASE SALES DATA COMMENT = 'Sales Data Database';`

Answer: A

Explanation:

Option A creates the database. Initially, `'ACCOUNTADMIN'` role has ownership and thus can modify it. Options B, C, and E are valid syntax but do not explicitly restrict initial modification access to only `'ACCOUNTADMIN'` post-creation. Option D is invalid syntax as `OWNER` cannot be specified during database creation. Permissions are managed separately using `GRANT` statements after creation.

QUESTION 6

A Snowflake administrator needs to determine the default collation setting for a newly created database. Which of the following methods can be used to identify the default collation of the database named `'MY DATABASE'`?

- A. `DESCRIBE DATABASE MY DATABASE;`
- B. `SHOW DATABASES LIKE 'MY DATABASE';`
- C. `SELECT` followed by `SELECT collation FROM INFORMATION_SCHEMA.DATABASES WHERE database_name =`
- D. `SELECT FROM WHERE option_name = 'COLLATION';`
- E. `SELECT DATABASE COLLATION FROM INFORMATION_SCHEMA.DATABASES WHERE DATABASE NAME = 'MY DATABASE';`

Answer: A

Explanation:

`'DESCRIBE DATABASE MY_DATABASE;'` will show all properties of the database, including the default collation. `'SHOW DATABASES LIKE 'MY DATABASE';'` will also show information about the database but does not guarantee displaying the default collation. Options C, D, and E use the `INFORMATION_SCHEMA`, which is useful, but option A gives direct and complete metadata. `'DATABASE_OPTIONS'` table does not exist in information schema. The most efficient method and recommended way is to use `'DESCRIBE DATABASE`.

QUESTION 7

A Snowflake user is attempting to query a table named 'CUSTOMERS' within the 'SALES DB.PUBLIC' schema. However, they are receiving an error indicating insufficient privileges. Assuming the user has been granted the 'USAGE' privilege on the database 'SALES', which additional privilege(s) is/are required for the user to successfully query the 'CUSTOMERS' table?

- A. SELECT privilege on the 'CUSTOMERS' table.
- B. USAGE privilege on the 'PUBLIC' schema.
- C. READ privilege on the 'CUSTOMERS' table.
- D. USAGE privilege on the 'SALES DB' database and SELECT privilege on the 'CUSTOMERS' table.
- E. USAGE privilege on the 'PUBLIC' schema and SELECT privilege on the 'CUSTOMERS' table.

Answer: E

Explanation:

To query a table, a user needs 'SELECT' privilege on the table and 'USAGE' privilege on the schema containing the table. Having 'USAGE' on the database is not enough; 'USAGE' is also needed on the schema. 'READ' is not a valid privilege on tables.

QUESTION 8

A Snowflake environment has two databases, 'DEV DB' and 'PROD DB'. A table 'EMPLOYEES' exists in both databases with identical schemas. A developer needs to create a view in 'DEV DB' that references the 'EMPLOYEES' table in 'PROD DB' to perform cross-database joins. Which of the following SQL statements will successfully create this view in 'DEV DB'?

- A. CREATE VIEW DEV DB.PUBLIC.EMPLOYEE VIEW AS SELECT FROM PROD DB.PUBLIC.EMPLOYEES;
- B. CREATE VIEW DEV DB.PUBLIC.EMPLOYEE VIEW AS SELECT FROM PROD DB.PUBLIC.EMPLOYEES WITH READ ONLY;
- C. CREATE OR REPLACE VIEW DEV DB.PUBLIC.EMPLOYEE VIEW AS SELECT FROM PROD DB.PUBLIC.EMPLOYEES;
- D. CREATE OR REPLACE VIEW DEV DB.PUBLIC.EMPLOYEE VIEW AS SELECT FROM PROD DB.PUBLIC.EMPLOYEES WITH SECURE;
- E. CREATE SECURE VIEW DEV DB.PUBLIC.EMPLOYEE VIEW AS SELECT FROM PROD DB.PUBLIC.EMPLOYEES;

Answer: CE

Explanation:

To create a view that references objects in other databases, you should fully qualify the object names in the 'SELECT' statement. Both options create the view referencing the PROD DB. Option C uses 'CREATE OR REPLACE VIEW' and option E uses 'CREATE SECURE VIEW' which are correct and commonly used options for this scenario. 'WITH READ ONLY' is not a supported clause for creating views in Snowflake. 'CREATE SECURE VIEW' creates a secure view which is recommended to protect the underlying objects' definition.

QUESTION 9

A data analyst wants to clone the 'SALES DB' database to create a development environment called 'DEV SALES DB'. The analyst needs to ensure that all schemas within 'SALES DB', including custom schemas with data, are replicated in the clone, and that the clone operates as a completely independent database. Which of the following sequence of commands is the most efficient and reliable way to achieve this?

- A. CREATE DATABASE CREATE SCHEMA CREATE SCHEMA Repeat for all schemas in

- SALES_DB; Manually copy data into each table.
- B. CREATE DATABASE DEV SALES DB CLONE SALES DB;
 - C. BACKUP DATABASE SALES DB TO S3; RESTORE DATABASE DEV SALES DB FROM S3;
 - D. CREATE OR REPLACE DATABASE DEV SALES DB AS COPY OF SALES DB;
 - E. CREATE DATABASE DEV SALES DB CLONE SALES DB COPY GRANTS;

Answer: B

Explanation:

'CREATE DATABASE CLONE SALES_DB;' is the correct and most efficient way to clone a database in Snowflake. It creates a point-in-time snapshot clone of the entire database, including all schemas, tables, and other objects. Option A is manual and highly inefficient. Option C involves backups and restores, which are not the intended approach for cloning within Snowflake. Option D's syntax is incorrect as 'AS COPY OF' is not a valid clause, and 'COPY GRANTS' with clone may not be relevant here.

QUESTION 10

A data engineer is tasked with creating multiple schemas within a Snowflake database called 'SALES_DB'. These schemas need to be isolated from each other to prevent accidental data access. Which of the following approaches is the MOST secure and efficient way to achieve this?

- A. Create all schemas under the 'PUBLIC' schema of 'SALES_DB' and grant specific privileges on each table.
- B. Create a separate database for each schema, leading to multiple databases instead of schemas within 'SALES_DB'.
- C. Create separate schemas within 'SALES_DB' and grant specific privileges on each schema to appropriate roles.
- D. Create one large schema and prefix all table names with the schema name to simulate schema separation.
- E. Creating external tables and configuring different storage integrations for each table.

Answer: C

Explanation:

Creating separate schemas within the database and assigning granular privileges provides the most secure and manageable separation. Option A is less secure. Option B is inefficient and overcomplicates the setup. Option D lacks proper schema isolation and management. Option E applies to external tables, which are not the focus of internal schema segregation.

QUESTION 11

You are tasked with deploying a new data application to Snowflake. This application requires several schemas for staging, transformation, and reporting. What is the recommended approach to create these schemas using Infrastructure as Code (IaC) and ensuring consistency across multiple Snowflake environments (DEV, TEST, PROD)?

- A. Manually create the schemas in each environment using the Snowflake web UI, documenting the steps for repeatability.
- B. Use a Snowflake client (e.g., SnowSQL) and a script to execute 'CREATE SCHEMA' statements, parameterizing the environment-specific database and schema names.
- C. Utilize a third-party IaC tool (e.g., Terraform) to define the schemas as resources, managing their creation and updates across environments.
- D. Copy the DDL (Data Definition Language) scripts generated by the Snowflake UI after manually creating schemas in one environment and executing them in others.
- E. Utilize Snowflake's data replication feature to replicate schemas between different environments

Answer: C

Explanation:

Using a dedicated IAC tool like Terraform is the best practice for managing Snowflake resources, including schemas. It ensures consistency, repeatability, and version control. Options A and D are manual and error-prone. Option B is better than A and D but lacks the full features of IAC tools (state management, dependency resolution, etc.). Option E is for data replication and not schema creation.

QUESTION 12

Which of the following statements are true regarding schemas in Snowflake? (Select TWO)

- A. A schema can only contain tables and views.
- B. Schemas are logical groupings of database objects within a database.
- C. Schemas must be explicitly created using the 'CREATE SCHEMA' command; Snowflake does not create any schemas by default.
- D. A database can contain multiple schemas, allowing for logical separation of data and objects.
- E. Schemas do not provide any security benefits; role-based access control must be configured on individual objects.

Answer: BD

Explanation:

Schemas are logical groupings of database objects within a database, allowing for logical separation. A database can indeed contain multiple schemas. Option A is incorrect as schemas can contain stages, file formats, sequences, user-defined functions, and other objects besides tables and views. Option C is false since the 'PUBLIC' schema is created by default. Option E is false since schemas are also used in access control, although individual object privileges provide more granular control.

QUESTION 13

A Snowflake administrator wants to ensure that all newly created schemas in a specific database automatically inherit certain grants (e.g., 'SELECT' privilege on all tables in the schema to a reporting role). Which of the following approaches is the MOST suitable to accomplish this?

- A. Create a stored procedure that is automatically executed after each 'CREATE SCHEMA' statement using a Snowflake Task.
- B. Use the 'DEFAULT_PRIVILEGES' parameter at the database level to automatically grant the desired privileges on future schemas and their objects.
- C. Implement a custom event listener that triggers upon schema creation and executes the necessary GRANT statements.
- D. Manually execute GRANT FUTURE GRANTS' statements after each schema creation to grant the desired privileges.
- E. Schedule a periodic Snowflake Task to search for recently created schemas and grant needed permissions.

Answer: B

Explanation:

'DEFAULT PRIVILEGES' is the intended and most efficient mechanism for automatically granting privileges on future objects within schemas. Options A, C, D, and E are more complex and less efficient. Option D requires manual execution each time. Option A requires more overhead. Option C is not available in Snowflake. Option E has a delay before new permissions are granted and can be a performance overhead.

QUESTION 14

You have a Snowflake environment with a database named 'ANALYTICS DB'. You need to load data into a table named 'CUSTOMER DATA' that resides in a schema called 'RAW DATA' within 'ANALYTICS DB'. The data is staged in an internal stage called '@RAW DATA STAGE'. You encounter an error during the 'COPY INTO' command. Identify TWO potential reasons that could be causing this error, assuming the stage exists, the file format is correctly specified, and the data file is present.

- A. The 'RAW DATA' schema does not exist within the 'ANALYTICS DB' database.
- B. The role executing the 'COPY INTO' command lacks the 'USAGE' privilege on the 'RAW DATA' schema.
- C. The table has a defined primary key, but the data in the staged file contains duplicate key values.
- D. The user executing the 'COPY INTO' command does not have the 'OWNERSHIP' privilege on the 'RAW DATA STAGE'.
- E. The warehouse associated with the current session has not been resumed, so resources aren't available.

Answer: AB

Explanation:

The 'USAGE' privilege on a schema is required to access objects within that schema. If the role lacks this privilege, it will not be able to access the 'CUSTOMER DATA' table for the 'COPY INTO' command. If the schema does not exist, then the 'COPY INTO' command can not find it and throws an error. Having no ownership on the stage is not important in this operation; COPY INTO commands need the READ privilege on the stage, not ownership. Option C assumes there is a primary key, and while this could cause failure, it isn't always the reason. If a warehouse is suspended, Snowflake should automatically resume it when a query is run.

QUESTION 15

A data engineer needs to create a table named 'EMPLOYEES' in the 'PUBLIC' schema of the database 'COMPANY DATA'. The table should store employee IDs as integers, names as strings, and hire dates as dates. Which of the following SQL statements correctly creates this table?

- A. CREATE TABLE COMPANY_DATA.PUBLIC.EMPLOYEES (EmployeeID INT, Name STRING, HireDate DATE);
- B. CREATE OR REPLACE TABLE EMPLOYEES (EmployeeID NUMBER, Name VARCHAR, HireDate DATE);
- C. CREATE TABLE EMPLOYEES (EmployeeID INTEGER, Name VARCHAR(255), HireDate DATETIME);
- D. CREATE OR REPLACE TABLE COMPANY_DATA.PUBLIC.EMPLOYEES (EmployeeID NUMBER, Name VARCHAR(255), HireDate DATE);
- E. CREATE OR REPLACE TABLE COMPANY_DATA.PUBLIC.EMPLOYEES (EmployeeID INT, Name VARCHAR(255), HireDate DATE);

Answer: E

Explanation:

The correct answer is E. It uses 'CREATE OR REPLACE' to avoid errors if the table exists, specifies the full table name with database and schema ('COMPANY_DATA.PUBLIC.EMPLOYEES'), and uses appropriate data types: 'INT' for integers, 'VARCHAR(255)' for strings, and 'DATE' for dates. Option A would cause errors if table already exists, Option B and C does not specify the database and schema. Option D uses NUMBER datatype.

QUESTION 16

You are tasked with optimizing a frequently queried table 'SALES DATA' that experiences many updates. Which of the following Snowflake table features can significantly improve query performance in this scenario?

- A. Using a Transient table instead of a Permanent table.
- B. Creating a View on top of the table.
- C. Clustering the table on the most frequently filtered columns.
- D. Using a Temporary table instead of a Permanent table.
- E. Partitioning the table on the date column.

Answer: C

Explanation:

Clustering is the most effective way to improve query performance for frequently queried tables, especially those with updates. Clustering organizes the physical storage of data based on the clustering keys, allowing Snowflake to efficiently prune data during queries. Views do not affect the storage of the actual data, and transient/temporary tables are generally used for other purpose and will not improve the performance of the queries. Snowflake does not support Partitioning of the table by Users.

QUESTION 17

You have a table named 'CUSTOMER ORDERS' with columns 'ORDER ID', 'CUSTOMER ID', and 'ORDER DATE'. You want to create a clone of this table for testing purposes, ensuring the cloned table contains only the schema definition and no data. Which of the following commands achieves this?

- A. CREATE TABLE TEST ORDERS LIKE CUSTOMER ORDERS;
- B. CREATE TABLE TEST ORDERS AS SELECT FROM CUSTOMER ORDERS WHERE 1=0;
- C. CREATE TABLE TEST ORDERS CLONE CUSTOMER ORDERS;
- D. CREATE TABLE TEST ORDERS LIKE CUSTOMER ORDERS DATA ONLY;
- E. CREATE TABLE TEST ORDERS CLONE CUSTOMER ORDERS WHERE 1=0;

Answer: B

Explanation:

Option B, 'CREATE TABLE TEST_ORDERS AS SELECT FROM CUSTOMER_ORDERS WHERE 1=0;', creates a new table TEST_ORDERS' with the same schema as but without any data, as the 'WHERE condition ensures no rows are selected. Option A works but Snowflake recommends cloning for metadata copying. Option C copies both schema and data. There is no such thing like 'DATA ONLY' option D. Option E is syntactically incorrect.

Thank You for Trying Our Product

Lead2pass Certification Exam Features:

- ★ More than **99,900** Satisfied Customers Worldwide.
- ★ Average **99.9%** Success Rate.
- ★ **Free Update** to match latest and real exam scenarios.
- ★ **Instant Download** Access! No Setup required.
- ★ Questions & Answers are downloadable in **PDF** format and **VCE** test engine format.
- ★ Multi-Platform capabilities - **Windows, Laptop, Mac, Android, iPhone, iPod, iPad**.
- ★ **100%** Guaranteed Success or **100%** Money Back Guarantee.
- ★ **Fast**, helpful support **24x7**.



View list of all certification exams: <http://www.lead2pass.com/all-products.html>



Microsoft



ORACLE



JUNIPER
NETWORKS



EMC²
where information lives

10% Discount Coupon Code: ASTR14