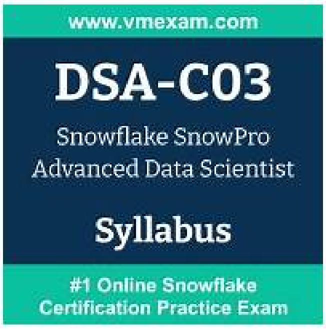
DSA-C03 Test Topics Pdf, DSA-C03 Learning Materials



P.S. Free & New DSA-C03 dumps are available on Google Drive shared by DumpsMaterials: https://drive.google.com/open?id=1EYP_HkBJ35TCdVgwXZoGTZEi8TB-P2ka

Many students often feel that their own gains are not directly proportional to efforts in their process of learning. This is because they have not found the correct method of learning so that they often have low learning efficiency. If you have a similar situation, we suggest you try DSA-C03 practice materials. DSA-C03 test guide is compiled by experts of several industries tailored to DSA-C03 Exam to help students improve their learning efficiency and pass the exam in the shortest time. DSA-C03 test guide involve hundreds of professional qualification examinations. No matter which industry you are in, DSA-C03 practice materials can meet you.

The DSA-C03 exam prep is produced by our expert, is very useful to help customers pass their exams and get the certificates in a short time. We are going to show our DSA-C03 guide braindumps to you. We can sure that our product will help you get the certificate easily. If you are wailing to believe us and try to learn our DSA-C03 Exam Torrent, you will get an unexpected result.

>> DSA-C03 Test Topics Pdf <<

DSA-C03 Learning Materials | DSA-C03 Testking Learning Materials

At the moment when you decided to choose our Snowflake DSA-C03 real dumps, we feel the responsibility to be with you during your journey to prepare for the DSA-C03 exam. So we clearly understand our duty to offer help in this area. If you have any question, you can just contact our online service, they will give you the most professional advice on our Snowflake DSA-C03 Exam Guide.

Snowflake SnowPro Advanced: Data Scientist Certification Exam Sample Questions (Q62-Q67):

NEW QUESTION # 62
You've built a machine learning model in scikit-learn and want to deploy it to Snowflake for real-time inference. You have the following options for deploying the model. Select all that apply and are considered a best practice for cost and time optimization:

- A. Migrate your entire Snowflake data warehouse to a different platform which better supports real-time ML inference.
- B. Implement a custom microservice that reads data from Snowflake, performs inference using the scikit-learn model, and

writes the predictions back to Snowflake.

- C. Use Snowflake's Snowpark Python API to directly load the model from a stage and execute inference using Snowpark DataFrames, which will implicitly handle the distributed processing of the data.
- D. Package the scikit-learn model using 'joblib' or 'pickle' , store it in a Snowflake stage, and create a Snowflake UDF (User-Defined Function) in Python to load the model from the stage and perform inference.
- E. Create a Snowflake external function that calls a cloud-based (AWS SageMaker, Azure Machine Learning, GCP Vertex A1) endpoint for inference, passing the input data to the endpoint and receiving the prediction back.

Answer: C,D

Explanation:
Options A and B are the recommended approaches. Option A leverages Snowflake UDFs for inference, which minimizes data transfer and leverages Snowflake's compute. Option B, using Snowpark, provides a more seamless integration with Snowflake's distributed processing capabilities. Option C introduces external dependencies and latency. Option D requires managing and maintaining a separate microservice and data transfer and Option E is not viable.

NEW QUESTION # 63
You are developing a Snowflake Native App that leverages Snowflake Cortex for text summarization. The app needs to process user-provided text input in real-time and return a summarized version. You want to expose this functionality as a secure and scalable REST API endpoint within the Snowflake environment. Which of the following strategies are MOST suitable for achieving this, considering best practices for security and performance?

- A. Create a Snowflake External Function using Python that directly calls the 'SNOWFLAKE.CORTEX.SUMMARIZE' function and expose this function via a REST API gateway outside of Snowflake.
- B. Write a Snowflake Stored Procedure using Javascript to invoke the 'SNOWFLAKE.CORTEX.SUMMARIZE function, deploy the procedure to a Snowflake stage, and then trigger it via an AWS Lambda function integrated with Snowflake.
- C. Utilize a Snowflake Stored Procedure written in SQL that invokes the 'SNOWFLAKE.CORTEX.SUMMARIZE' function, and then create a Snowflake API Integration to expose the stored procedure as a REST endpoint.
- D. Develop a Snowflake Native App containing a Python UDF that calls 'SNOWFLAKCORTEX.SUMMARIZE function, and expose it as a REST API endpoint using Snowflake's API Integration feature within the app package.
- E. Develop a Snowflake Native App that includes a Java UDF that calls 'SNOWFLAKE.CORTEX.SUMMARIZE and expose a REST API using Snowflake's built-in REST API capabilities within the Native App framework.

Answer: C,D

Explanation:
Options B and E are the most suitable. B: Using a stored procedure and API integration is a secure and standard way to expose Snowflake functionality as a REST API. The API Integration handles authentication and authorization within the Snowflake environment. E: Snowflake Native App containing a Python UDF is correct as using Snowflake's API integration is appropriate way to expose the endpoint as REST API with secure connectivity. Option A: Directly calling Cortex using external function and exposing it outside of Snowflake is not as secure as it requires managing authentication and authorization outside of Snowflake. Option C: Java UDF can be used but using snowflake API is not recommended. Option D: Deploying stored procedures to a stage and triggering them with Lambda is more complex and less secure compared to using API Integrations within Snowflake.

NEW QUESTION # 64
A data scientist is exploring customer purchase data in Snowflake to identify high-value customer segments. They have a table named 'CUSTOMER TRANSACTIONS with columns 'CUSTOMER ID', 'TRANSACTION_DATE', and 'PURCHASE_AMOUNT'. They want to calculate the interquartile range (IQR) of 'PURCHASE AMOUNT for each customer. Which SQL query using Snowsight is the most efficient and accurate way to calculate and display the IQR for each 'CUSTOMER ID?

```
SELECT CUSTOMER_ID, APPROX_PERCENTILE(PURCHASE_AMOUNT, 0.75) - APPROX_PERCENTILE(PURCHASE_AMOUNT, 0.25) AS IQR FROM CUSTOMER_TRANSACTIONS GROUP BY
CUSTOMER_ID;

SELECT CUSTOMER_ID, PERCENTILE_CONT(0.75) WITHIN GROUP (ORDER BY PURCHASE_AMOUNT) - PERCENTILE_CONT(0.25) WITHIN GROUP (ORDER BY PURCHASE_AMOUNT)
AS IQR FROM CUSTOMER_TRANSACTIONS GROUP BY CUSTOMER_ID;

SELECT CUSTOMER_ID, MAX(PURCHASE_AMOUNT) - MIN(PURCHASE_AMOUNT) AS IQR FROM CUSTOMER_TRANSACTIONS GROUP BY CUSTOMER_ID;

SELECT CUSTOMER_ID, STDDEV(PURCHASE_AMOUNT) AS IQR FROM CUSTOMER_TRANSACTIONS GROUP BY CUSTOMER_ID;

SELECT CUSTOMER_ID, QUANTILE(PURCHASE_AMOUNT, 4)[3] - QUANTILE(PURCHASE_AMOUNT, 4)[1] AS IQR FROM CUSTOMER_TRANSACTIONS GROUP BY CUSTOMER_ID;
```


- A. Option E

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

Answer: A

Explanation:
Option E, using 'QUANTILE, is the most accurate way to calculate the IQR. 4)' returns an array representing the quartiles (0%, 25%, 50%, 75%, 100%). Subtracting the 25th percentile (index 1) from the 75th percentile (index 3) gives the IQR. Other options either approximate the percentiles (APPROX_PERCENTILE), calculate the range (MAX-MIN), or calculate standard deviation, none of which directly give the IQR. Option B while syntactically valid is less performant and returns the IQR on entire table not grouped by customer.

NEW QUESTION # 65
You are analyzing website clickstream data stored in Snowflake to identify user behavior patterns. The data includes user ID, timestamp, URL visited, and session ID. Which of the following unsupervised learning techniques, combined with appropriate data transformations in Snowflake SQL, would be most effective in discovering common navigation paths followed by users? (Choose two)

- A. K-Means clustering on features extracted from the URL data, such as the frequency of visiting specific domains or the number of pages visited per session. This requires feature engineering using SQL.
- B. Association rule mining (e.g., Apriori) applied directly to the raw URL data to find frequent itemsets of URLs visited together within the same session. No SQL transformations are required.
- C. Principal Component Analysis (PCA) to reduce the dimensionality of the URL data, followed by hierarchical clustering. This will group similar URLs together.
- D. Sequence clustering using time-series analysis techniques (e.g., Hidden Markov Models), after transforming the data into a sequence of URLs for each session using Snowflake's LISTAGG function ordered by timestamp.
- E. DBSCAN clustering on the raw URL data, treating each URL as a separate dimension. This will identify URLs that are frequently visited by many users.

Answer: A,D

Explanation:
Sequence clustering is appropriate for identifying navigation paths because it considers the order of URLs visited within a session. Using Snowflake's LISTAGG function allows for creating the required sequential data. K-Means clustering can also be effective if relevant features are engineered from the URL data (e.g., frequency of visiting specific domains). Association rule mining is less suitable for identifying navigation paths as it focuses on co-occurrence rather than sequence. PCA followed by hierarchical clustering and DBSCAN are not well-suited for identifying sequential navigation paths from clickstream data. Option 'A' is incorrect because association rule mining directly on raw URL data is unlikely to be effective without prior sequence extraction. Option 'D' and 'E' are not suitable for this type of analysis.

NEW QUESTION # 66
You are a data scientist working for an e-commerce company. You have a table named 'sales_data' with columns 'product_id' , customer_id' , 'transaction_date' , and 'sale_amount'. You need to identify the top 5 products by total sale amount for each month. Which of the following Snowflake SQL queries is the MOST efficient and correct way to achieve this, while also handling potential ties in sale amounts?

- A.

```
SELECT product_id, SUM(sale_amount) AS total_sales,
       ROW_NUMBER() OVER (PARTITION BY DATE_TRUNC('MONTH', transaction_date) ORDER BY SUM(sale_amount) DESC)
FROM sales_data
GROUP BY product_id, DATE_TRUNC('MONTH', transaction_date)
HAVING ROW_NUMBER() <= 5;
```

- B.

```
SELECT product_id, total_sales
FROM (
    SELECT product_id, SUM(sale_amount) AS total_sales,
           DENSE_RANK() OVER (PARTITION BY DATE_TRUNC('MONTH', transaction_date) ORDER BY SUM(sale_amount) DESC) AS rank_num
    FROM sales_data
    GROUP BY product_id, DATE_TRUNC('MONTH', transaction_date)
) subquery
WHERE rank_num <= 5;
```

- C.

```
SELECT product_id, SUM(sale_amount) AS total_sales,
       NTILE(5) OVER (PARTITION BY DATE_TRUNC('MONTH', transaction_date) ORDER BY SUM(sale_amount) DESC)
FROM sales_data
GROUP BY product_id, DATE_TRUNC('MONTH', transaction_date)
HAVING NTILE(5) = 1;
```

- D.

```
SELECT product_id, total_sales
FROM (
    SELECT product_id, SUM(sale_amount) AS total_sales,
           RANK() OVER (PARTITION BY DATE_TRUNC('MONTH', transaction_date) ORDER BY SUM(sale_amount) DESC) AS rank_num
    FROM sales_data
    GROUP BY product_id, DATE_TRUNC('MONTH', transaction_date)
) subquery
WHERE rank_num <= 5;
```

- E.

```
SELECT product_id, SUM(sale_amount) AS total_sales,
       RANK() OVER (PARTITION BY DATE_TRUNC('MONTH', transaction_date) ORDER BY SUM(sale_amount) DESC)
FROM sales_data
GROUP BY product_id, DATE_TRUNC('MONTH', transaction_date)
HAVING RANK() <= 5;
```

Answer: B,D

Explanation:
Options C and E are correct. Both use a subquery to calculate the rank of each product within each month's sales, then filter for the top 5 products. The main difference is that option C uses DENSE_RANK(), which assigns consecutive ranks even if there are ties in sales amount (resulting in more than 5 products being selected if there are ties for the 5th position), while option E uses RANK(), which assigns the same rank to tied values but can skip ranks. Option A is incorrect because it attempts to filter using HAVING on a ranking calculated within the same query level, which is not allowed in many SQL implementations (and can be logically incorrect). Options B and D are incorrect as they employ ROW_NUMBER() and NTILE(5) respectively. ROW_NUMBER will not handle ties correctly, while NTILE just divides the data into 5 groups without explicitly identifying the 'top' 5. Option A uses a rank function inside the HAVING clause which is often syntactically invalid.

NEW QUESTION # 67

......

improvement.

Excellent Offers By DumpsMaterials – Free Snowflake DSA-C03 Dumps Updates and Free Demo

We will give you full refund if you fail to pass the exam after buying DSA-C03 Exam Torrent from us, If you focus on the study materials from our company, you will find that the pass rate of our products is higher than other study materials in the market, yes, we have a 99% pass rate, which means if you take our the DSA-C03 study materials into consideration, it is very possible for you to pass your exam and get the related certification.

You will frequently find these DSA-C03 PDF files downloadable and can then archive or print them for extra reading or studying on-the-go, Here are the respective features and detailed disparities of our DSA-C03 practice materials.

One of the most favorable demo of our DSA-C03 exam questions on the web is also written in PDF version, in the form of Q&A, can be downloaded for free.

BTW, DOWNLOAD part of DumpsMaterials DSA-C03 dumps from Cloud Storage: https://drive.google.com/open?id=1EYP_HkBJ35TCdVgwXZoGTZEi8TB-P2ka