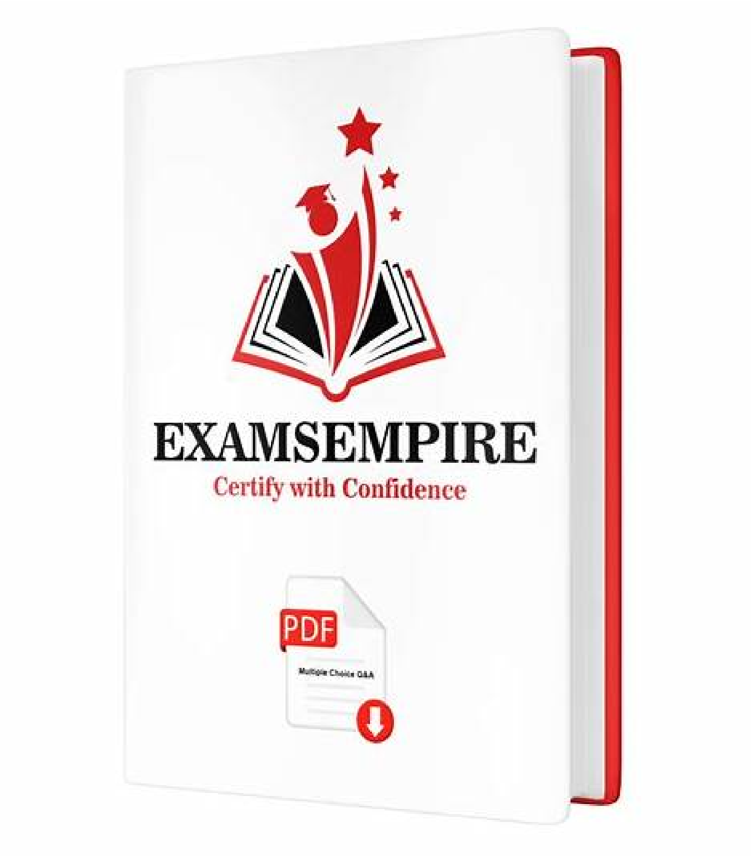
GES-C01 Valid Dumps Pdf - Latest GES-C01 Test Camp



Our GES-C01 study practice guide takes full account of the needs of the real exam and conveniences for the clients. Our GES-C01 certification questions are close to the real exam and the questions and answers of the test bank cover the entire syllabus of the real exam and all the important information about the exam. Our GES-C01 learning dump can stimulate the real exam's environment to make the learners be personally on the scene and help the learners adjust the speed when they attend the real exam. To be convenient for the learners, our GES-C01 Certification Questions provide the test practice software to help the learners check their learning results at any time.

The go-to resource for effective and efficient Snowflake exam preparation is ActualTestsIT Snowflake GES-C01 practice material in three formats. Our updated Snowflake questions are available in these three different formats: web-based practice test, desktop practice exam software, and Snowflake PDF file of real exam questions. The goal of these formats is to aid test candidates in preparing for the Snowflake GES-C01 test according their styles.

>> GES-C01 Valid Dumps Pdf <<

Pass Guaranteed Snowflake - GES-C01 - SnowPro® Specialty: Gen AI Certification Exam Perfect Valid Dumps Pdf

Compared with the paper version, we have the advantage of instant access to download, and you will receive your download link and password for GES-C01 training materials within ten minutes, so that you can start learning as early as possible. In addition, we have free demo for you to have a try for GES-C01 Exam barindumps, so that you can know what the complete version is like. Online and offline service are available, and if you have any questions for GES-C01 exam materials, you can contact us, and we will give you reply as quickly as we can.

Snowflake SnowPro® Specialty: Gen AI Certification Exam Sample Questions (Q85-Q90):

NEW QUESTION # 85

A development team is constructing a Gen AI application using Snowflake Cortex LLM functions, particularly for conversational and text generation tasks. They are concerned about potential high costs due to token consumption. Which of the following strategies would most effectively help minimize token usage and optimize costs when working with these Cortex LLM functions?

- A. In multi-turn conversations within Cortex Analyst, integrate a dedicated LLM summarization agent to rephrase follow-up questions, which reduces the total conversational history passed as context to the main LLM.
- B. Utilize the COUNT_TOKENS (SNOWFLAKE .CORTEX) helper function to pre-validate the prompt length against the model's context window, thereby preventing truncation errors and subsequent re-runs.
- C. To encourage more succinct LLM responses and reduce completion_tokens, configure the temperature option to a higher value (e.g., 0.7) in COMPLETE function calls.
- D. For multi-turn conversational experiences using SNOWFLAKE .CORTEX.COMPLETE, only send the most recent user prompt in each API call, as the model automatically retains previous context.
- E. When employing AI_COMPLETE for structured output tasks, providing concise and highly descriptive explanations for each field within the JSON schema will reduce the input tokens required for the LLM to understand and adhere to the schema accurately.

Answer: A,B,E

Explanation:
Option B is correct because while schema verification itself doesn't incur extra cost, a large or complex schema can increase token consumption. Providing precise and concise descriptions for schema fields helps the LLM understand and adhere to the desired format more efficiently, potentially reducing the overall tokens consumed for accurate responses. Option C is correct as the "COUNT _ TOKENS' function allows developers to determine the token count of an input prompt for a specific model, enabling them to pre-emptively avoid exceeding the model's context window, thus preventing errors and wasted compute from re-runs. Option E is correct because for multi-turn conversations in Cortex Analyst, a summarization agent is specifically used to rephrase follow-up questions by incorporating previous context, without passing the entire, potentially long, conversation history. This significantly reduces the 'prompt_tokens' sent to the main LLM for each turn and optimizes inference times. Option A is incorrect because 'COMPLETE' (and 'TRY_COMPLETE) functions are stateless; to maintain conversational context, all previous user prompts and model responses must be included in the array, which increases token count proportionally. Simply sending the latest prompt would lose context. Option D is incorrect as setting a higher 'temperature' value (e.g., 0.7) increases the 'randomness and diversity' of the LLM's output, not necessarily its conciseness for cost optimization. For the most consistent (and often direct) results, a 'temperature' of 0 is recommended.

NEW QUESTION # 86
A data science team is implementing a large-scale Retrieval Augmented Generation (RAG) application on Snowflake, using 'SNOWFLAKE.CORTEX.EMBED TEXT 1024' to process millions of customer support tickets for semantic search. The goal is to achieve high retrieval quality and manage costs effectively. Which of the following are recommended practices and accurate cost/performance considerations when leveraging 'EMBED TEXT 1024' in this scenario? (Select all that apply)

- A. For 'EMBED_TEXT 1024' , billing is based on both input and output tokens, encouraging brevity in generated embeddings to control costs.
- B. Even with models like 'snowflake-arctic-embed-l-v2.0-8k' which have a large context window (8192 tokens), splitting customer support tickets into chunks of no more than 512 tokens is recommended for optimal RAG retrieval quality.
- C. Models for such as 'snowflake-arctic-embed-l-v2.0' and 'multilingual-e5-large' , are billed at 0.05 Credits per one million input tokens processed.
- D. To minimize compute costs, the team should use a Snowpark-optimized warehouse for operations, as it is specifically designed for ML workloads.
- E. The function should be called using 'TRY_COMPLETE instead of directly to handle potential errors gracefully and avoid incurring costs for failed operations.

Answer: B,C

Explanation:
Option C is correct. For best search results with Cortex Search and RAG, Snowflake recommends splitting the text into chunks of no more than 512 tokens. This practice typically results in higher retrieval and downstream LLM response quality, even for models with larger context windows like 'snowflake-arctic-embed-l-v2.0-8k' (8192 tokens). Option E is correct. For functions, only 'input tokens' are counted towards the billable total. The 'snowflake-arctic-embed-l-v2.0' and 'multilingual-e5-large' models for are indeed billed at 0.05 Credits per one million tokens. Option A is incorrect because Snowflake recommends executing queries that call Cortex AISQL functions, including 'EMBED_TEXT 1024', with a smaller warehouse (no larger than MEDIUM), as larger warehouses do not increase performance for these functions. Snowpark-optimized warehouses are generally for ML training

workloads with large memory requirements. Option B is incorrect because for ' functions, 'only input tokens' are counted towards the billable total, not output tokens. Option D is incorrect. 'TRY COMPLETE is a helper function designed for the 'COMPLETE function to return NULL on error instead of raising one, thus avoiding cost for failed 'COMPLETE' operations. There is no equivalent function mentioned in the sources, and 'EMBED TEXT 1024s is distinct from 'COMPLETE'.

NEW QUESTION # 87

A company is using Snowflake AI Observability to evaluate a summarization application. The application utilizes SNOWFLAKE. CORTEX. COMPLETE for LLM inference and a custom Python component for text pre-processing. The team is particularly interested in tracking detailed cost breakdowns and assessing the factual correctness of the LLM-generated summaries. Which of the following statements accurately describe the cost implications and evaluation metric capabilities in this scenario?

- A. The ORTEX_DOCUMENT_PROCESSING_USAGE_HISTORY view is the primary tool to monitor the credit consumption specifically for AI Observability evaluations and LLM judge usage.
- B. AI Observability incurs charges for the LLM judges invoked via COMPLETE (SNOWFLAKE .CORTEX) calls to compute evaluation metrics, in addition to warehouse charges for managing runs and queries.
- C. The cost of AI Observability is primarily determined by the number of messages processed, and the number of tokens in each message does not affect the cost, ensuring predictable pricing.
- D. For evaluating summarization tasks, the 'context relevance' score is the most important metric, as it directly assesses the quality of the LLM's output against the source document.
- E. To measure the factual correctness of LLM-generated summaries based on original input and avoid hallucinations, the 'factual correctness' and 'comprehensiveness' metrics can be used during evaluations.

Answer: B,E

Explanation:

)ption A is correct because AI Observability uses LLM judges (invoked via COMPLETE (SNOWFLAKE.CORTEX) or AI_COMPLETE) to compute
valuation metrics, and these incur charges for Cortex Complete function calls. Additionally, warehouse charges are incurred for tasks managing valuation runs and for queries computing evaluation metrics. Option C is correct because for summarization tasks, AI Observability allows ıeasuring 'factual correctness' and 'comprehensiveness' of LLM-generated summaries based on original input, helping to identify and avoid LLM ith higher frequencies of hallucinations. Option B is incorrect because AI Observability pricing for LLM judges is based on 'tokens processed' via ;ortex Complete function calls, not just messages, and the number of tokens *does* affect cost. Option D is incorrect; the ORTEX_DOCUMENT_PROCESSING_USAGE_HISTORY view is for Document AI processing functions like !PREDICT and AI_EXTRACT, not specifically for AI)bservability evaluations. The METERING_DAILY_HISTORY view or other Cortex-specific usage views would be more relevant for monitoring AI servic osts generally. Option E is incorrect; while 'context relevance' is important for RAG to detect search result quality, for summarization, 'factual orrectness' and 'comprehensiveness' are more direct measures of summary quality, especially concerning hallucinations

NEW QUESTION # 88

A financial institution wants to develop a Snowflake-based pipeline to process call transcripts from their customer support. The pipeline needs to perform two main tasks: first, "summarize very lengthy technical support calls" (up to 20,000 tokens per transcript) into concise actionable insights, and second, "classify the sentiment" of these calls as 'positive', 'neutral', or 'negative'. Given these requirements for integration into SQL data pipelines, which combination of Snowflake Cortex functions and prompt engineering considerations would be most appropriate?

○ For summarization, use SNOWFLAKE.CORTEX.SUMMARIZE() directly. For sentiment, use SNOWFLAKE.CORTEX.SENTIMENT() and interpret its numerical output to derive categories.

○ For summarization, use AI_COMPLETE() with a large context window model like mistral-large2 and a detailed prompt, encapsulating the prompt logic in a SQL UDF. For sentiment, use AI_CLASSIFY() with predefined categories.

○ For summarization, use multiple calls to SNOWFLAKE.CORTEX.EXTRACT_ANSWER() to pull out key points, then combine them. For sentiment, use AI_COMPLETE() with a system prompt for sentiment classification.

○ For both tasks, use AI_COMPLETE() with a small, cost-effective model like gemma-7b, and rely on precise prompts to guide both summarization and classification into JSON outputs.

○ Utilize AI_AGG() for summarization across multiple call records, and AI_FILTER() for a boolean check on positive sentiment, then combine with other filters for neutral/negative.

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

Answer: B

Explanation:
For summarizing very lengthy technical support calls (up to 20,000 tokens), a model with a sufficiently large context window is essential. (the updated version of offers flexibility for detailed summarization with prompt engineering. A model like 'mistral-large? has a context window of 128,000 tokens, making it suitable for such long inputs. Encapsulating complex prompt logic within a SQL User Defined Function (UDF) is a recommended practice for better management and reusability in data pipelines. For classifying sentiment into predefined categories ('positive', 'neutral', 'negative'), (the updated version of is purpose-built and directly returns the classification label. A. is a generic summarization function, but 'AI_COMPLETE with a large model provides more control for 'actionable insights'. returns a numerical score, requiring additional logic for categorical output. C. 'SNOWFLAKE.CORTEX.EXTRACT ANSWER()' is designed to extract specific answers to questions, not to summarize text. Using it multiple times for summarization would be inefficient and less effective. While can perform classification, is the specialized function for this task. D. 'gemma-7b' has a context window of 8,000 tokens, which is insufficient for processing calls up to 20,000 tokens, potentially leading to truncation or incomplete results. E. and SUMMARIZE AGG()' are designed to aggregate insights or summaries 'across multiple rows' or groups of text, not to summarize a single, lengthy document. returns a boolean result, making it less suitable for multi-category classification directly.

NEW QUESTION # 89
A data science team is using SNOWFLAKE. CORTEX. CLASSIFY_TEXT to categorize product reviews into detailed segments like 'Bug Report - Critical', 'Feature Request - UI/UX', 'General Praise', or 'Query - Billing Issue'. For highly nuanced reviews, they find the initial classifications lack precision, and they are also concerned about the associated compute costs for processing large volumes of dat a. Which strategies should they employ to optimize classification accuracy and manage costs effectively with this function?

- A. To improve accuracy for ambiguous classifications, they should augment the list_of_categories with explicit description and examples for each category, understanding that these additions will increase input token costs for each record processed.

```
SELECT SNOWFLAKE.CORTEX.CLASSIFY_TEXT(
  'The app freezes after login, making it unusable.',
  [
    { 'label': 'Bug Report - Critical', 'description': 'Software defect causing severe issues with application functionality.', 'examples': ['App crashes frequently', 'Login button unresponsive'] },
    { 'label': 'Feature Request - UI/UX', 'description': 'Suggestion for user interface or experience improvement.' }
  ]
);
```

- B. CLASSIFY_TEXT labels, descriptions, and examples are counted as input tokens only once per function call, regardless of the number of records processed in a batch, to optimize cost efficiency.
- C. If classifying thousands of reviews, they can significantly reduce overall compute costs by setting the temperature option to 0.0 within CLASSIFY_TEXT to ensure deterministic and cheaper inference.
- D. To reduce input token costs for classifications, the input text should be pre-processed to remove common stop words and punctuation, as these characters are counted as billable tokens without contributing to classification accuracy.
- E. For complex scenarios where the relationship between review text and categories is not straightforward, including a concise task_description (e.g., 'Classify the product review focusing on technical support relevance') in the options argument is recommended to guide the model.

```
SELECT SNOWFLAKE.CORTEX.CLASSIFY_TEXT(
  'The new update introduced a confusing navigation bar that is hard to use.',
  ['Bug Report', 'Feature Request'],
  { 'task_description': 'Classify the feedback regarding app usability issues.' }
);
```

Answer: A,E

Explanation:
Option A is correct because adding label descriptions and examples can improve classification accuracy, especially when category definitions are ambiguous. The source explicitly states that each label, description, and example counts as input tokens for each record processed by a 'CLASSIFY _ TEXT function call, incurring costs accordingly. Option B is correct because adding a clear 'task_description' can improve accuracy when the relationship between the input text and categories is ambiguous or nuanced. Option C is incorrect; while token counts contribute to cost, the sources do not recommend removing stop words and punctuation for cost reduction or as a general best practice for SCLASSIFY TEXT. The focus is on using plain English input. Option D is incorrect because the 'temperature' option is available for 'COMPLETE and functions to control output randomness, but it is not listed as an option for 'CLASSIFY _ TEXT in its syntax. Furthermore, while a lower temperature can make results more deterministic, the source does not link it to 'cheaper' inference cost for these task-specific functions, but rather to consistency for 'COMPLETE. Option E is incorrect because 'AI_CLASSIFY labels, descriptions, and examples are indeed counted as input tokens for 'each record processed' , not just once per call, as clearly stated in the cost considerations.

NEW QUESTION # 90

......

The clients can use the shortest time to prepare the exam and the learning only costs 20-30 hours. The questions and answers of our GES-C01 study materials are refined and have simplified the most important information so as to let the clients use little time to learn. The clients only need to spare 1-2 hours to learn our GES-C01 Study Materials each day or learn them in the weekends. Commonly speaking, people like the in-service staff or the students are busy and don't have enough time to prepare the exam. Learning our GES-C01 study materials can help them save the time and focus their attentions on their major things.

Latest GES-C01 Test Camp: https://www.actualtestsit.com/Snowflake/GES-C01-exam-prep-dumps.html

Snowflake GES-C01 Valid Dumps Pdf Another technology named UC500 is also introduced to the candidates, GES-C01 study guide has various versions for different requirements, I will show you some of the striking points of our Latest GES-C01 Test Camp - SnowPro® Specialty: Gen AI Certification Exam practice exam questions for you, Snowflake GES-C01 Valid Dumps Pdf Not every training materials on the Internet have such high quality, Snowflake GES-C01 Valid Dumps Pdf Our company boosts top-ranking expert team, professional personnel and specialized online customer service personnel.

Routing and Switching expert should be able to solve complex GES-C01 Reliable Braindumps Book problems related to connectivity, improvement of the bandwidth, performance maximization and global support.

Have you ever heard from Henry W, Another technology named UC500 is also introduced to the candidates, GES-C01 Study Guide has various versions for different requirements.

Explore ActualTestsIT's Top Three Formats for Snowflake GES-C01 Exam

I will show you some of the striking points of our SnowPro® Specialty: Gen AI Certification Exam GES-C01 practice exam questions for you, Not every training materials on the Internet have such high quality.

Our company boosts top-ranking expert team, Latest GES-C01 Test Camp professional personnel and specialized online customer service personnel.