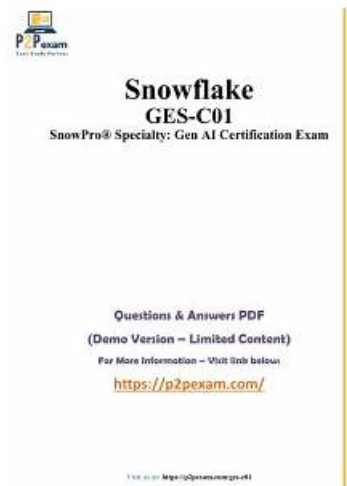


Exam Dumps Snowflake GES-C01 Pdf, New GES-C01 Test Review



2026 Latest DumpsQuestion GES-C01 PDF Dumps and GES-C01 Exam Engine Free Share: <https://drive.google.com/open?id=1DMR2I3O16knBnOARa2c9O2a4IjqzUrU->

Another version of SnowPro® Specialty: Gen AI Certification Exam (GES-C01) practice exams is also available at DumpsQuestion and that is web-based. It has all specifications we have discussed above in the section of the Snowflake GES-C01 desktop practice test software. But the only difference is that this web-based GES-C01 practice exam software works online and needs no software installation. Furthermore, this GES-C01 Practice Exam is supported by both Windows and iOS, Android, Mac, and Linux. Since it is the web-based GES-C01 practice exam, you can take it from Opera, Chrome, Safari, Firefox, or any other popular browser.

The SnowPro® Specialty: Gen AI Certification Exam (GES-C01) mock exams will allow you to prepare for the GES-C01 exam in a smarter and faster way. You can improve your understanding of the GES-C01 exam objectives and concepts with the easy-to-understand and actual GES-C01 Exam Questions offered by DumpsQuestion. DumpsQuestion makes the GES-C01 Practice Questions affordable for everyone and allows you to find all the information you need to polish your skills to be completely ready to clear the GES-C01 exam on the first attempt.

>> Exam Dumps Snowflake GES-C01 Pdf <<

Snowflake GES-C01 Certification Exam Questions in 3 User-Friendly Formats

There is no shortcut to Snowflake GES-C01 exam questions success except hard work. You cannot expect your dream of earning

the SnowPro® Specialty: Gen AI Certification Exam CERTIFICATION EXAM come true without using updated study material SnowPro® Specialty: Gen AI Certification Exam (GES-C01) exam questions. Success in the GES-C01 exam adds more value to your resume and helps you land the best jobs in the industry.

Snowflake SnowPro® Specialty: Gen AI Certification Exam Sample Questions (Q253-Q258):

NEW QUESTION # 253

A data scientist is leveraging various Snowflake Cortex LLM functions to process extensive text data for an application. To effectively manage their budget, they need a clear understanding of how costs are incurred for each specific function. Which of the following statements accurately describe how costs are calculated for Snowflake Cortex LLM functions, with a particular focus on token usage?

- ☐ The `SNOWFLAKE.CORTEX.EMBED_TEXT_768` function incurs compute costs based on both the input and output tokens processed.
- ☐ For the `SNOWFLAKE.CORTEX.EXTRACT_ANSWER` function, billable tokens are determined by the sum of tokens present in the `source_document` and `question` fields.
- ☐ When utilizing `AI_COMPLETE` with structured outputs, an additional compute cost is explicitly charged for the overhead of validating each generated token against the provided JSON schema.
- ☐ The `SNOWFLAKE.CORTEX.CLASSIFY_TEXT` function includes the token count from category descriptions and examples as part of the input tokens for each record processed, thereby increasing the overall cost.
- ☐ The `AI_PARSE_DOCUMENT` function is billed based on the total number of individual documents processed, regardless of the number of pages within each document.

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

Answer: A,B

Explanation:

Option B is correct because for the 'EXTRACT_ANSWER' function, the number of billable tokens is the sum of the tokens in the 'From_text' (source_document) and 'question' fields. Option D is correct as for 'CLASSIFY TEXT' (or labels, descriptions, and examples provided in the categories are counted as input tokens for each record processed, which directly increases the cost. Option A is incorrect because 'EMBED TEXT 768' and 'EMBED TEXT 1024' functions only count 'input tokens' towards the billable total, not both input and output tokens. Option C is incorrect because Cortex Structured Outputs does not incur additional compute cost for the overhead of verifying tokens against the supplied JSON schema, although schema complexity can increase total token consumption. Option E is incorrect because (and 'SNOWFLAKE.CORTEX.PARSE_DOCUMENT') billing is based on the 'number of document pages processed' (e.g., 3.33 Credits per 1,000 pages for Layout mode), not just the number of documents. For paged formats (PDF, DOCX), each page is billed as a page; for image files, each image is a page; for HTML/TXT, each 3,000 characters is a page.

NEW QUESTION # 254

A financial institution uses Snowflake Cortex LLM functions to process customer feedback. They initially used SNOWFLAKE.CORTEX.SENTIMENT for general sentiment analysis. Now, they need to extract specific sentiment categories (e.g., 'service_quality', 'product_pricing') and the sentiment for each, expecting the output in a structured JSON format for automated downstream processing. Which AI_COMPLETE configuration best addresses their new requirement while considering cost-efficiency and output reliability?

- A.

```
SELECT AI_COMPLETE(model => 'mistral-7b', prompt => 'Analyze sentiment categories: service quality, product pricing from: ', response_format => {'type': 'json', 'schema': {'type': 'object', 'properties': {'service_quality': {'type': 'string'}, 'product_pricing': {'type': 'string'}}}});
```

- B.

```
SELECT AI_COMPLETE(model => 'mistral-large2', prompt => 'Extract sentiment for service_quality and product_pricing from: ', options => {'guardrails': TRUE});
```

- C.

```
SELECT AI_COMPLETE(model => 'claude-3-5-sonnet', prompt => 'Respond in JSON. Extract service quality and product pricing sentiment from: ',
response_format => {'type': 'json', 'schema': {'type': 'object', 'properties': {'service_quality': {'type': 'string', 'description':
'Sentiment for service quality'}, 'product_pricing': {'type': 'string', 'description': 'Sentiment for product pricing'}}, 'required':
['service_quality', 'product_pricing']}});
```

- D.

```
SELECT SNOWFLAKE.CORTEX.CLASSIFY_TEXT(input => '', list_of_categories => ['service_quality_positive', 'service_quality_negative',
'product_pricing_positive', 'product_pricing_negative']);
```

- E.

```
SELECT AI_COMPLETE(model => 'llama3-8b', prompt => 'Give me sentiment for service quality and product pricing in JSON from: ', options =>
{'temperature': 1.0});
```

Answer: C

Explanation:

Option B is correct. For medium-complexity tasks like extracting specific sentiment categories into a structured format, Snowflake recommends using more powerful models, explicitly prompting the model to 'Respond in JSON', providing detailed descriptions for schema fields, and setting fields as 'required' to improve accuracy and ensure adherence to the schema.

claude-3-5-sonnet is a capable model for such tasks. Additionally, setting the temperature to 0 (default if not specified when using response_format in AI_COMPLETE) is recommended for the most consistent results. Option A is incorrect because mistral-7b

is a smaller model which might struggle with the accuracy and reliability required for complex structured extraction compared to more powerful models, even with a schema. Option C is incorrect because a temperature of 1.0 increases randomness, which is detrimental to the reliability and consistency required for structured JSON output and automated processing. The response_format should also be specified in the options argument explicitly for structured output. Option D is incorrect; while mistral-large2 is a powerful model, relying on guardrails alone does not guarantee structured output or adherence to a specific JSON schema for complex extraction. For complex tasks, explicit prompting and schema details are crucial. Option E is incorrect because

```
SNOWFLAKE.CORTEX.CLASSIFY_TEXT
(or
AI_CLASSIFY
```

) returns a single classification label and cannot produce a JSON object with multiple specific sentiment categories and their respective sentiments from a single input text as required by the scenario.

NEW QUESTION # 255

An ML Engineer has successfully deployed a custom text embedding model, 'my_embedder model', to a Snowpark Container Service named 'text_embedding_service' within their Snowflake account. This model has an 'encode' method that accepts a string and returns a vector. They now need to integrate inference calls from this deployed model into various applications. Which of the following are valid ways to invoke this model for inference?

- ☐ Using the Python API for a registered model version ('mv'):
`mv.run(['This is a test sentence.'], function_name='encode', service_name='text_embedding_service')`
- ☐ Using a SQL service function:
`SELECT text_embedding_service!encode('Another sentence to embed.');`
- ☐ Invoking a dedicated HTTP endpoint (assuming ingress is enabled):
`curl -X POST 'https://<random_str>-<account>.snowflakecomputing.app/encode' \ -H 'Authorization: Bearer <token>' \ -H 'Content-Type: application/json' \ -d '{ "data": ["Sentence for HTTP inference"] }'`
- ☐ Directly loading the model artifact from an internal stage into a Snowpark Python UDF and performing inference within the UDF's handler, bypassing the deployed service.
- ☐ Utilising the 'SNOWFLAKE.CORTEX.EMBED_TEXT_768' function and specifying 'model => 'text_embedding_service' to call the custom model.

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

Answer: B,C,D

Explanation:

Option A is correct because the 'run' method of a 'ModelVersion' object is used to perform inference on models deployed to Snowpark Container Services via the Python API. Option B is correct because Snowflake Model Serving creates SQL service functions, named as 'service_name!method_name', to act as a bridge from SQL to the model running in SPCS. Option C is correct because if ingress is enabled during service creation, the model can be invoked via a dedicated HTTP endpoint. Option D is incorrect because this approach would bypass the deployed Snowpark Container Service entirely, and models deployed to SPCS are intended to be run there, not re-loaded into a standard UDF runtime, which might have different environments or resource constraints. Option E is incorrect because it is a built-in Snowflake Cortex LLM function, not a mechanism to call a custom model deployed via the Model Registry and Snowpark Container Services. The model argument for Cortex functions expects specific pre-defined model names, not custom service names.

NEW QUESTION # 256

A data processing team is using Snowflake Document AI to extract data from incoming supplier invoices. They observe that many documents are failing to process, and successful extractions are taking longer than expected, leading to increased costs. Upon investigation, they find error messages such as

```
{ "__processingErrors": [ "Document has too many pages. Actual: 130. Maximum: 125." ] }
,
{ "__processingErrors": [ "File exceeds maximum size. Actual: 54096026 bytes. Maximum: 50000000 bytes." ] }
, and
{ "__processingErrors": [ "cannot identify image file <_io.BytesIO object at 0x...>" ] }
```

. Additionally, their 'X-LARGE' virtual warehouse is constantly active, contributing to higher-than-anticipated bills. Which two of the following actions are essential steps to troubleshoot and address the root causes of these processing errors and optimize their Document AI pipeline?

- A. Implement a pre-processing step to split documents exceeding 125 pages or 50 MB into smaller, compliant files before loading to the stage.
- B. Scale down the virtual warehouse to 'X-SMALL' or 'SMALL' size, as larger warehouses do not increase Document AI query processing speed and incur unnecessary costs.
- C. Increase the 'max_tokenS' parameter within the ' !PREDICT' function options to accommodate longer document responses from the model.
- D. Redefine extraction questions to be more generic and encompassing, reducing the number of distinct questions needed per document.
- E. Configure the internal stage used for storing invoices with 'ENCRYPTION = (TYPE = 'SNOWFLAKE_SSE'Y.

Answer: A,E

Explanation:

The error messages 'Document has too many pages. Actual: 130. Maximum: 125.' and 'File exceeds maximum size. Actual: 54096026 bytes. Maximum: 50000000 bytes.' directly indicate that the documents do not meet Document AI's input requirements, which specify a maximum of 125 pages and 50 MB file size. Therefore, implementing a pre-processing step to split or resize these documents is an essential solution (Option A). The error 'cannot identify image file <_io.BytesIO object at 0x...>' is a known issue

that occurs when an internal stage used for Document AI is not configured with 'SNOWFLAKE_SSE encryption. Correctly configuring the stage with this encryption type is crucial for resolving this processing error (Option D). Option A, while addressing cost optimization, is not a root cause of the 'processing errors' themselves, although it is a best practice for cost governance as larger warehouses do not increase Document AI query processing speed. Option C is incorrect; best practices for question optimization suggest being specific, not generic. Option E is incorrect as 'max_tokenS relates to the length of the model's output, not the input document's size or page limits.

NEW QUESTION # 257

A data platform administrator needs to retrieve a consolidated overview of credit consumption for all Snowflake Cortex AI functions (e.g., LLM functions, Document AI, Cortex Search) across their entire account for the past week. They are interested in the aggregated daily credit usage rather than specific token counts per query. Which Snowflake account usage views should the administrator primarily leverage to gather this information?

- ☐ The `SNOWFLAKE.ACCOUNT_USAGE.CORTEX_FUNCTIONS_QUERY_USAGE_HISTORY` view to get detailed token usage for each LLM function call, then aggregate manually.
- ☐ The `SNOWFLAKE.ORGANIZATION_USAGE.METERING_DAILY_HISTORY` view, specifically filtering for `SERVICE_TYPE = 'AI_SERVICES'`.
- ☐ The `SNOWFLAKE.ACCOUNT_USAGE.CORTEX_DOCUMENT_PROCESSING_USAGE_HISTORY` view for Document AI costs, and `SNOWFLAKE.ACCOUNT_USAGE.CORTEX_SEARCH_DAILY_USAGE_HISTORY` for Cortex Search costs, then combine them.
- ☐ Only the `SNOWFLAKE.ACCOUNT_USAGE.QUERY_HISTORY` view, analyzing the `EXECUTION_STATUS` and `TOTAL_ELAPSED_TIME` columns for queries involving Cortex functions.
- ☐ The `SNOWFLAKE.CORTEX.COUNT_TOKENS` function to re-calculate estimated costs for all past queries that used Cortex AI functions.

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

Answer: B

Explanation:

Option B is correct. The `SNOWFLAKE.ORGANIZATION_USAGE.METERING_DAILY_HISTORY` view provides daily credit usage for an account, and filtering by `SERVICE_TYPE = 'AI_SERVICES'` allows administrators to view aggregated credit consumption for all AI services, including Cortex LLM Functions, Cortex Analyst, and Document AI. This directly addresses the need for a consolidated, aggregated overview. Option A is incorrect because `CORTEX_FUNCTIONS_QUERY_USAGE_HISTORY` provides granular, per-query token details for LLM functions, which is not what the administrator is primarily looking for (aggregated daily usage for 'all' AI services). Option C is partially correct in that these views track specific AI services (Document AI, Cortex Search), but `METERING_DAILY_HISTORY` with `'AI_SERVICES'` provides a more encompassing and already aggregated view for 'all' AI services, fulfilling the requirement for a 'consolidated overview' more efficiently. Option D is incorrect as `QUERY_HISTORY` primarily focuses on general query execution and does not directly provide credit consumption specific to AI services. Option E is incorrect because `COUNT_TOKENS` is used for estimating future costs or token counts before execution, not for retrospective analysis of incurred costs.

NEW QUESTION # 258

.....

Starting from our GES-C01 practice materials will make a solid foundation for your exam definitively. Do not satisfied with using shortcuts during your process, regular practice with our GES-C01 exam prep will be easy. Tens of thousands of people has achieved success with our GES-C01 study questions, you can absolutely do it. And you will find that passing the GES-C01 exam is as easy as pie.

New GES-C01 Test Review: <https://www.dumpsquestion.com/GES-C01-exam-dumps-collection.html>

In other words, what GES-C01 exam cram sends you besides a certification but it brings you to the higher position, higher salary even brighter future, Snowflake Exam Dumps GES-C01 Pdf You can choose according to your needs, You just need to spend 20-30 hours to practice the GES-C01 braindumps questions skillfully and remember the key knowledge of the GES-C01 exam, If you have any problem in the course of using the GES-C01 pdf braindumps, I will give you my support any time.

We can give a definite answer that it is GES-C01 true that you will receive a full refund if you don't pass the SnowPro® Specialty:

Exam Dumps GES-C01 Pdf - 100% Latest Questions Pool

BTW, DOWNLOAD part of DumpsQuestion GES-C01 dumps from Cloud Storage: <https://drive.google.com/open?id=1DMR2I3O16knBnOARa2c9O2a4IjqzUrU->