Snowflake GES-C02 Dumps

Shared by Bonner on 25-11-2026

For More Free Questions and Preparation Resources

Check the Links on Last Page

Question 1

Question Type: MultipleChoice

A user opened the Snowflake Cortex LLM Playground in the AI & ML Studio within Snowsight, selected gemma-7b from the Standard models drop-down list, and typed "What is Gen AI?" into the dialog box.

Which SQL statement will achieve the same result?

Options:

A- SELECT SNOWFLAKE.CORTEX.CLASSIFY_TEXT('gemma-7b', 'What is Gen AI?');
B- SELECT SNOWFLAKE.CORTEX.COMPLETE('gemma-7b', 'What is Gen AI?');
C- SELECT SNOWFLAKE.CORTEX.EXTRACT_ANSWER('gemma-7b', 'What is Gen AI?');
D- SELECT SNOWFLAKE.CORTEX.SUMMARIZE('gemma-7b', 'What is Gen AI?');

Answer:

B

Explanation:

The Cortex LLM Playground in Snowsight provides an interactive graphical interface for experimenting with foundation models available within Snowflake Cortex. When a user selects a model from the Standard models drop-down (such as `gemma-7b`) and enters an input prompt, the underlying web interface executes a completion request.

The SQL programmatic equivalent of this playground interaction is `SNOWFLAKE.CORTEX.COMPLETE`. Its two-argument signature accepts:

1. The target model identifier string (`'gemma-7b'`).

2. The user-supplied input prompt (`'What is Gen AI?'`).

The function passes the prompt to the selected model and returns the generated text completion, matching the behavior of the Playground UI.

The other functions serve different tasks:

* `CLASSIFY_TEXT` assigns text to predefined taxonomy labels.

* `EXTRACT_ANSWER` searches an input document to answer a specific factual question.

* `SUMMARIZE` condenses long text passages into shorter summaries.

None of these alternatives provide open-ended prompt completion against a chosen model.
References & Study Guide Topics:

* Snowflake Gen AI Functions: `SNOWFLAKE.CORTEX.COMPLETE` Syntax and Invocation

* Snowsight Tooling: Cortex LLM Playground Mechanics and SQL Equivalents

Question 2

Question Type: MultipleChoice

A team has developed a machine learning model outside of Snowflake using a third-party framework. The model needs to be deployed within Snowflake to support:

* Centralized version control

* Metadata tracking

Which Snowflake feature should be used to meet these requirements?

Options:

A- ML Observability
B- Container Runtime for ML
C- Snowflake Feature Store
D- Snowflake Model Registry

Answer:

D

Explanation:

The Snowflake Model Registry provides centralized machine learning lifecycle governance and deployment inside the Snowflake Data Cloud. Designed to manage models regardless of where they were trained, it natively supports logging artifacts from third-party frameworks, including scikit-learn, PyTorch, TensorFlow, XGBoost, and Hugging Face.

The Model Registry satisfies both organizational requirements:

1. **Centralized Version Control:** Models are stored as schema-level entities that support multiple version deployments under a unified parent object. Teams can manage versions (e.g., v1, v2), assign default versions, and govern rollbacks through standard SQL or Python

commands.

2. **Metadata Tracking:** During logging, the Model Registry captures operational metadata, such as training dataset references, performance metrics, input/output schemas, target runtime dependencies, and custom tags. This metadata remains queryable and governed under Snowflake's RBAC framework.

ML Observability focuses on monitoring production models for data drift and latency; Container Runtime provides execution compute for training jobs; and the Snowflake Feature Store manages feature pipelines rather than versioning model artifacts.

References & Study Guide Topics:

* Snowflake Gen AI Governance: Snowflake Model Registry Governance and MLOps

* Enterprise Model Management: Artifact Versioning and Metadata Tracking

==========

Question 3

Question Type: MultipleChoice

Which Snowflake Cortex AI observability metrics are used to evaluate LLM-generated responses? (Choose two)

Options:

A- Accuracy
B- Coherence
C- Correctness
D- F1 Score
E- Precision

Answer:

B, C

Explanation:

Snowflake Cortex AI Observability provides specialized, automated evaluation metrics designed to systematically assess the quality, safety, and reliability of large language model completions in production and evaluation workflows.

Two core LLM evaluation metrics native to Snowflake Cortex are:

Correctness: Measures factual alignment and semantic fidelity between the LLM-generated output and an established ground-truth response or reference context, identifying factual discrepancies, omissions, or model hallucinations.

Coherence: Evaluates the structural and logical integrity of the generated output, verifying that the text is well-organized, fluent, logically consistent, and free of contradictions or abrupt narrative breaks.

Traditional statistical classification metrics---such as Accuracy, Precision, and F1 Score---are mathematical indicators used to evaluate supervised classification models against rigid true/false categorical labels; they are

not the qualitative, generative assessment metrics implemented in Cortex AI Observability for evaluating unstructured LLM response text.

References & Study Guide Topics:

Snowflake Gen AI Functions: Cortex AI Observability Evaluation Framework

LLM Quality Benchmarking: Qualitative Metrics (Correctness, Coherence, Hallucination Detection)

==========

Question 4

Question Type: MultipleChoice

Which of the Snowflake Cortex LLM functions are task-specific? (Choose two)

Options:

A- COMPLETE
B- COUNT_TOKENS
C- SUMMARIZE
D- TRANSLATE
E- TRY_COMPLETE

Answer:

C, D

Explanation:

Snowflake Cortex divides its natural language capabilities into general-purpose completion functions and specialized, task-specific functions:

* **Task-Specific Functions:** Built to automate defined NLP operations without requiring prompt engineering or model parameter selection. `SUMMARIZE` condenses English text into a concise overview, while `TRANSLATE` converts text from a designated source language into a target language. Both functions encapsulate purpose-built models tuned specifically for their respective tasks.

* **General-Purpose Completion Functions:** `COMPLETE` and `TRY_COMPLETE` are prompt-driven interfaces that accept arbitrary natural language instructions and route them to foundation models (such as Llama, Mistral, or Claude).

* **Utility Functions:** `COUNT_TOKENS` is an administrative utility that calculates the number of tokens in a string for a specified model; it is not an NLP task function.

Consequently, `SUMMARIZE` and `TRANSLATE` represent the task-specific LLM functions among the options.

References & Study Guide Topics:

* Snowflake Gen AI Functions: Specialized NLP Utilities vs. Foundation Completions

* Cortex Architecture: Automated Workflows with `SUMMARIZE` and `TRANSLATE`

==========

Question 5

Question Type: MultipleChoice

A Gen AI Specialist has deployed a base model as an LLM in a production environment for a proof of concept. The model is larger and more expensive than other base models.

How can the Specialist lower production costs without affecting performance for the limited tasks needed to support the proof of concept?

Options:

A- Use Snowflake Cortex Search to limit the number of context documents.
B- Use Snowflake Cortex Agents to ensure the base model is concise
C- Use Snowflake Cortex Fine-tuning on a different base model

D- Provide a semantic model to ensure the LLM is able to interpret the prompts

Answer:

C

Explanation:

In enterprise generative AI architectures, frontier foundation models (such as 70B+ parameter variants) offer strong general reasoning but incur higher credit consumption per token. When supporting a defined, specialized task for a proof of concept, relying on an oversized base model is often cost-inefficient.

Snowflake Cortex Fine-tuning (`SNOWFLAKE.CORTEX.FINETUNE`) allows developers to adapt a smaller, lower-cost foundation model (such as `llama3-8b` or `mistral-7b`) using domain-specific training data. By training on prompt-completion pairs representing the targeted workload, the smaller base model learns the required domain vocabulary, output formats, and task logic.

This specialized model can achieve task performance comparable to larger foundation models while running on a smaller parameter footprint, significantly reducing per-token credit consumption in production.

Cortex Search handles document retrieval rather than adapting model reasoning efficiency; Cortex Agents orchestrate multi-tool reasoning rather than optimizing inference billing; and semantic models support Cortex Analyst text-to-SQL generation rather than general completion cost optimization.

References & Study Guide Topics:

* Snowflake Gen AI Functions: Cortex Fine-Tuning Architecture and Economics

* LLM Optimization: Parameter Sizing, Model Distillation, and Domain Adaptation

==========

Question 6

Question Type: MultipleChoice

A company is organizing an event and asked participants about their food restrictions. The responses have been saved in a table named EVENT_REGISTRATIONS:

| ID | FOOD_RESTRICTIONS |

| :--- | :--- |

| 1 | allergic to nuts |

| 2 | no gluten |

| 3 | no milk or dairy products |

| 4 | no meat |

Which query will compile a list of foods that should not be served?

```sql

## Options:

A- SELECT SNOWFLAKE.CORTEX.COMPLETE(
'llama3.3-70b',
'Which foods must be avoided based on the customer provided food restrictions (respond with only a list of foods): '
|| food_restrictions
)
FROM EVENT_REGISTRATIONS;
B- SELECT SNOWFLAKE.CORTEX.SENTIMENT(
'The customer provided these food restrictions:'
|| food_restrictions
|| '.Which foods must be avoided'
)
FROM EVENT_REGISTRATIONS;
C- SELECT SNOWFLAKE.CORTEX.CLASSIFY_TEXT(
'The customer provided these food restrictions:'
|| food_restrictions
|| '.Which foods must be avoided', ['foods', 'diets']
)
FROM EVENT_REGISTRATIONS;
D- SELECT SNOWFLAKE.CORTEX.SUMMARIZE(
'Which foods must be avoided based on the customer provided food restrictions:'
|| food_restrictions
)
FROM EVENT_REGISTRATIONS;
```

Answer:

A

Explanation:

Inferring specific culinary items to exclude based on high-level dietary constraints (such as deriving wheat, barley, and rye from 'no gluten') requires domain reasoning and generative text synthesis.

`SNOWFLAKE.CORTEX.COMPLETE` accepts dynamic string concatenations that combine instructional prompts with tabular column values. In Option A, the query instructs an advanced foundation model (`llama3. 3-70b`) to evaluate the customer-provided restrictions and synthesize an explicit list of prohibited foods.

The alternative functions cannot perform this task:

* `SENTIMENT` returns a floating-point polarity score between -1 and 1.

* `CLASSIFY_TEXT` assigns the string to one of the provided static labels (`'foods'` or `'diets'`) rather than generating a list.

* `SUMMARIZE` condenses existing source text; it cannot infer new domain knowledge that is not already explicitly written in the source column.

References & Study Guide Topics:

* Snowflake Gen AI Functions: Generative Reasoning with `COMPLETE`

* Prompt Engineering: Dynamic SQL Concatenation for In-Database Inference

==========

Question 7

Question Type: MultipleChoice

Which arguments are required when you use the log_model method to log a model in the Snowflake Model Registry? (Choose two)

Options:

A- model
B- model_name
C- code_paths
D- version_name
E- python_version

Answer:

A, B

Explanation:

When managing machine learning models via the Snowflake Snowpark ML API, the `Registry.log_model()` method commits model artifacts, dependencies, and operational metadata as first-class objects into the Snowflake Model Registry.

The method signature strictly mandates two primary arguments:* **`model`:** The supported in-memory model object or model reference to be serialized (such as scikit-learn estimators, PyTorch models, Hugging Face pipelines, or custom Snowpark ML model classes).* **`model_name`:** A valid Snowflake database identifier string specifying the name under which the model will be cataloged within the active database and schema.

The remaining parameters are optional:* `version_name` is optional; if omitted, Snowflake automatically generates a unique version identifier (such as a timestamp-based string).* `code_paths` and `python_version` are optional configurations used to supply supplementary custom code dependencies and pin execution environment runtimes.References & Study Guide Topics:* Snowflake Gen AI Governance: Snowflake Model Registry API Specifications* ML Lifecycle Management: `Registry.log_model` Mandatory Parameters and Serialization==========

Question 8

Question Type: MultipleChoice

A Gen AI Specialist is writing a Python script that leverages Snowflake Cortex to answer users' questions.

The script contains this code:

Python

from snowflake.cortex import CompleteOptions, complete user_question = < 'Prompt that could generate an unsafe response' >

Regardless of the content of the prompt (user_question) that is passed to snowflake.cortex.complete, the Specialist must ensure that the function filters potentially unsafe or harmful response language.

Which code will accomplish this?

Python

Options:

A- response = complete('llama3.1-70b',
[dict(role='user', content=user_question)], options=CompleteOptions(temperature=1),
)
B- response = complete('llama3.1-70b',
[dict(role='user', content=user_question)], stream=False,
)
C- response = complete('llama3.1-70b',
[dict(role='user', content=user_question)],
options=CompleteOptions(guardrails=True),
)
D- response = complete('llama3.1-70b',
[dict(role='user', content=user_question)], options=CompleteOptions(response_format='safe'),
)

Answer:

C

Explanation:

To prevent large language models from returning toxic, violent, sexually explicit, or otherwise hazardous content, Snowflake integrates Cortex Guard directly into its LLM completion endpoints. Within the Snowpark Python API, this capability is exposed through the snowflake.cortex.CompleteOptions class.

Passing options=CompleteOptions(guardrails=True) instructs the Cortex inference engine to route the generated output through an automated safety evaluation model prior to returning it to the calling application. If an output violates safety boundaries, Cortex replaces the payload with a standardized safety rejection response, safeguarding downstream consumers regardless of the user's input prompt.

The other parameters shown serve entirely different functional roles: temperature governs sampling stochasticity and response creativity, stream dictates whether responses are returned incrementally via Server-Sent Events (SSE), and response_format accepts a structured JSON schema rather than safety flags.

References & Study Guide Topics:

Snowflake Gen AI Governance: Cortex Guard Configuration in Snowpark Python Enterprise AI Safety: Filtering Harmful LLM Completions via CompleteOptions

==========

Question 9

Question Type: MultipleChoice

Which APIs or functions can be used to query the Snowflake Cortex Search Service? (Choose three)

Options:

A- Snowflake Python APIs
B- Snowflake REST APIs
C- The CORTEX.COMPLETE function
D- The EMBED_TEXT_768 function
E- The SEARCH_PREVIEW function
F- The CORTEX.SUMMARIZE function

Answer:

A, B, E

Explanation:

Snowflake Cortex Search provides dedicated programmatic and SQL interfaces to query the hybrid search index and retrieve the top-k relevant context chunks for RAG architectures:

1. **Snowflake Python APIs:** Using the Snowflake Python SDK or Snowpark client (`snowflake.core. cortex.search_service`), developers can execute search queries directly from Python applications, notebooks, or Streamlit interfaces.

2. **Snowflake REST APIs:** External applications, microservices, and web clients can retrieve search chunks over standard HTTPS endpoints by making authenticated REST calls to the Cortex Search service endpoint.

3. **The SEARCH_PREVIEW function:** In SQL worksheets and stored procedures, developers can test, query, and preview retrieval results directly using the built-in system function `SNOWFLAKE.CORTEX. SEARCH_PREVIEW(...)`.

In contrast, `CORTEX.COMPLETE` executes text generation rather than searching an index;

`EMBED_TEXT_768` computes vector embeddings from input strings; and `CORTEX.SUMMARIZE` produces a condensed textual synopsis. None of these functions query a Cortex Search Service.

References & Study Guide Topics:

* Snowflake Gen AI Functions: Cortex Search Service Retrieval Protocols

* Application Development: Querying Indexes via Python, REST, and `SEARCH_PREVIEW`

==========

Question 10

Question Type: MultipleChoice

A Gen AI Specialist is developing a Gen AI application in Snowflake and must debug a particular record that produced an inaccurate response. The Specialist needs to see, step by step, how the input prompt was processed, what context was retrieved, which tools were called, and how the LLM responded.

Which Snowflake AI observability feature should be used?

Options:

A- Metrics
B- Logging
C- Tracing
D- Monitoring

Answer:

C

Explanation:

Snowflake AI Observability incorporates native telemetry tools---built on OpenTelemetry standards---to provide full runtime visibility across complex Generative AI and agentic pipelines.

When diagnosing compound Gen AI workflows (such as multi-step agent execution, tool-calling sequences, or Retrieval-Augmented Generation retrieval flows), Tracing is the required capability. Tracing records the end-to-end execution path of an individual request as a structured collection of spans within an Event Table. Each trace span captures granular, step-by-step metadata, including:

The raw input prompt and intermediate prompt transformations.

The specific documents, chunks, and similarity scores returned by retrieval engines (e.g., Cortex Search).

External functions or tools invoked by reasoning agents, including input arguments and returned

payloads.

The raw completion, token counts, and latency produced by the underlying LLM.

In contrast, Logging captures discrete, point-in-time diagnostic text messages emitted by application code; Metrics aggregate numeric operational counters over time (such as token rates or invocation throughput); and Monitoring provides dashboard-level health alerting rather than step-by-step contextual call graphs.

References & Study Guide Topics:

Snowflake Gen AI Governance: Cortex AI Observability & OpenTelemetry Telemetry Operational Debugging: Distributed Tracing and Span Analysis in Event Tables

==========

P2P exams

Question 11

Question Type: MultipleChoice

Which of the Snowflake Cortex LLM functions are task-specific? (Choose two)

Options:

A- COMPLETE
B- COUNT_TOKENS
C- SUMMARIZE
D- TRANSLATE
E- TRY_COMPLETE

P2P exams

Answer:

C, D

Explanation:

Snowflake Cortex divides its natural language capabilities into general-purpose completion functions and specialized, task-specific functions:

* **Task-Specific Functions:** Built to automate defined NLP operations without requiring prompt engineering or model parameter selection. `SUMMARIZE` condenses English text into a concise overview, while `TRANSLATE` converts text from a designated source language into a target language. Both functions encapsulate purpose-built models tuned specifically for their respective

tasks.

* **General-Purpose Completion Functions:** `COMPLETE` and `TRY_COMPLETE` are prompt-driven interfaces that accept arbitrary natural language instructions and route them to foundation models (such as Llama, Mistral, or Claude).

* **Utility Functions:** `COUNT_TOKENS` is an administrative utility that calculates the number of tokens in a string for a specified model; it is not an NLP task function.

Consequently, `SUMMARIZE` and `TRANSLATE` represent the task-specific LLM functions among the options.

References & Study Guide Topics:

* Snowflake Gen AI Functions: Specialized NLP Utilities vs. Foundation Completions

* Cortex Architecture: Automated Workflows with `SUMMARIZE` and `TRANSLATE`

==========

Question 12

Question Type: MultipleChoice

A Gen AI Specialist is writing a Python script that leverages Snowflake Cortex to answer users' questions.

The script contains this code:

Python

from snowflake.cortex import CompleteOptions, complete user_question = < 'Prompt that could generate an unsafe response' >

Regardless of the content of the prompt (user_question) that is passed to snowflake.cortex.complete, the Specialist must ensure that the function filters potentially unsafe or harmful response language.

Which code will accomplish this?

Python

Options:

A- response = complete('llama3.1-70b', [dict(role='user', content=user_question)], options=CompleteOptions(temperature=1),

)
B- response = complete('llama3.1-70b',
[dict(role='user', content=user_question)], stream=False,
)
C- response = complete('llama3.1-70b',
[dict(role='user', content=user_question)],
options=CompleteOptions(guardrails=True),
)
D- response = complete('llama3.1-70b',
[dict(role='user', content=user_question)], options=CompleteOptions(response_format='safe'),
)

Answer:

C

Explanation:

To prevent large language models from returning toxic, violent, sexually explicit, or otherwise hazardous content, Snowflake integrates Cortex Guard directly into its LLM completion endpoints. Within the Snowpark Python API, this capability is exposed through the snowflake.cortex.CompleteOptions class.

Passing options=CompleteOptions(guardrails=True) instructs the Cortex inference engine to route the generated output through an automated safety evaluation model prior to returning it to the calling application. If an output violates safety boundaries, Cortex replaces the payload with a standardized safety rejection response, safeguarding downstream consumers regardless of the user's input prompt.

The other parameters shown serve entirely different functional roles: temperature governs sampling stochasticity and response creativity, stream dictates whether responses are returned incrementally via Server-Sent Events (SSE), and response_format accepts a structured JSON schema rather than safety flags.

References & Study Guide Topics:

Snowflake Gen AI Governance: Cortex Guard Configuration in Snowpark Python Enterprise AI Safety: Filtering Harmful LLM Completions via CompleteOptions

==========

To Get Premium Files for GES-C02 Visit
https://www.p2pexams.com/products/ges-c02

For More Free Questions Visit
https://www.p2pexams.com/snowflake/pdf/ges-c02

