HeatWave GenAI Getting Started Guide

Version `9.0.0 - Innovation`

Public

Table of contents

HeatWave GenAI Overview

HeatWave GenAI lets you communicate with HeatWave using natural language queries. Both the entered queries and responses generated by the system are in natural language.

Note: This document assumes that you're familiar with the HeatWave database systems.

HeatWave GenAI includes the following functionalities:

- **Generative AI**
  HeatWave GenAI uses large language models (LLMs) to enable natural language communication. You can use the generative AI capabilities of the LLMs to create fresh content and summarize existing content. However, as these LLMs are trained on public data, the responses to your queries are generated based on information available in the public data sources. This is best suited for cases you want to write fresh content or prepare text summaries such as generating insights for private and public documents as well as summarizing logs.
- **Vector Search using RAG**
  HeatWave GenAI provides an inbuilt vector store that you can use to store enterprise-specific proprietary content. It uses vector embeddings and retrieval-augmented generation (RAG) to enable context-based search using documents available in the object storage, which lets you perform similarity searches across the available documents, and helps GenAI to produce more relevant, personalised, and accurate results.
  This is best suited for cases when you need most accurate and relevant responses from information available in public and private enterprise data, such as analysis reports and feedback summaries.
- **HeatWave Chat**
  This is an inbuilt chatbot, extends text-generation and vector search to let you ask multiple follow-up questions about a topic in a single session. It can even draw its knowledge from documents ingested by the vector store. This is best suited for cases where you want to make continuous conversation about a subject such as for creating AI-powered customer support chatbots.

Benefits

Some key benefits of using HeatWave GenAI are described below:

- The Natural language processing (NLP) capabilities of the LLMs let non-technical users have human-like conversations with the system in natural language.
- The in-database integration of LLMs and embedding generation eliminates the need for using external solutions, and ensures the security of the proprietary content.
- The in-database integration of LLMs, vector store, and embedding generation simplifies complexity of applications that use these features.
- The cost of running natural language queries is significantly low as HeatWave GenAI is available at no additional cost for HeatWave users.
- HeatWave GenAI integrates with other in-database capabilities such as machine learning, analytics, and Lakehouse.

See Also

- HeatWave GenAI: Technical Overview
- HeatWave GenAI Chapter in HeatWave User Guide

Getting Started

- Review the Requirements.
- To view step-by-step guidance on how to run HeatWave Chat with a vector store, see Quickstart: Setting Up a GenAI-Powered Help Chat.

Requirements

- To use HeatWave GenAI, you require a HeatWave database system connection
    - The database system must be version `9.0.0 - Innovation` or higher.
    - The ECPU Shape of database system must be `MySQL.32`.
    - A HeatWave Cluster must be added to your database system. The shape of the cluster must be `HeatWave.512GB`.
    - HeatWave Lakehouse must be enabled on the database system.
- To run vector search, you need an OCI Object Storage bucket for storing files that you want the vector store to ingest. Vector store can ingest files in the following formats: PDF, PPT, TXT, HTML, and DOC.

Generating Text-Based Content

For generating text-based content and summarizing text, use the following methods:

- The `ML_MODEL_LOAD` method loads a large language model (LLM) into the HeatWave cluster.
- The `ML_GENERATE` method uses the LLM to generate the text output.

Before You Begin

- Connect to your HeatWave database system.

Generating New Content

To generate text-based content using HeatWave GenAI, perform the following steps:

1. To load the LLM in HeatWave memory, use the `ML_MODEL_LOAD` method:

   ```
   call sys.ML_MODEL_LOAD('<LLMModel>', NULL);
   ```

   Replace <***LLMModel***> with the name of the LLM model that you want to use. The available models are: `mistral-7b-instruct-v1` and `llama2-7b-v1`.

   For example:

   ```
   call sys.ML_MODEL_LOAD('mistral-7b-instruct-v1', NULL);
   ```

2. To define your natural language query, set the `@query` session variable:

   ```
   set @query="<QueryInNaturalLanguage>";
   ```

   Replace <***QueryInNaturalLanguage***> with a natural language query of your choice.

   For example:

   ```
   set @query="Write an article on Artificial intelligence in 200 words.";
   ```

3. To generate text-based content, pass the query to the LLM using the `ML_GENERATE` method with the second parameter set to `generation`:

   ```
   select sys.ML_GENERATE(@query, JSON_OBJECT("task", "generation",
   "model_id", "mistral-7b-instruct-v1"));
   ```

   Text-based content that is generated by the LLM in response to your query is printed as output. It looks similar to the text output shown below:

   ```
   | {"text": " Artificial Intelligence, commonly referred to as AI, is a rapidly growing
   field that focuses on creating intelligent machines capable of performing tasks that
   typically require human intelligence. These tasks include things like understanding natural
   language, recognizing images, and making decisions.\n\nAI technology has come a long way in
   recent years, thanks to advances in machine learning and deep learning algorithms. These
   algorithms allow machines to learn from data and improve their performance over time. This
   has led to the development of more advanced AI systems, such as virtual assistants like
   Siri and Alexa, which can help users with tasks like setting reminders and answering
   questions.\n\nAI is also being used in a variety of other industries, including healthcare,
   finance, and transportation. In healthcare, AI is being used to help doctors diagnose
   diseases and develop treatment plans. In finance, AI is being used to detect fraud and make
   investment decisions. In transportation, AI is being used to develop self-driving cars and
   improve traffic flow.\n\nDespite the many benefits of AI, there are also concerns about its
   potential impact on society. Some worry that AI could lead to job displacement and a loss
   of privacy. Others worry that AI could be used for malicious purposes, such as cyber
   attacks or surveillance.\n"} |
   ```

Summarizing Text

To summarize text, perform the following steps:

1. To load the LLM in HeatWave memory, use the `ML_MODEL_LOAD` the method:

   ```
   call sys.ML_MODEL_LOAD('<LLMModel>', NULL);
   ```

   Replace <***LLMModel***> with the name of the LLM model that you want to use. The available models are: `mistral-7b-instruct-v1` and `llama2-7b-v1`.

   For example:

   ```
   call sys.ML_MODEL_LOAD('mistral-7b-instruct-v1', NULL);
   ```

2. To define the text that you want to summarize, set the `@text` session variable:

   ```
   set @text="<TextToSummarize>";
   ```

   Replace <***TextToSummarize***> with the text that you want to summarize.

For example:

```
set @text="Artificial Intelligence (AI) is a rapidly growing field that has the
potential to revolutionize how we live and work. AI refers to the development of
computer systems that can perform tasks that typically require human intelligence,
such as visual perception, speech recognition, decision-making, and language
translation.\n\nOne of the most significant developments in AI in recent years has
been the rise of machine learning, a subset of AI that allows computers to learn
from data without being explicitly programmed. Machine learning algorithms can
analyze vast amounts of data and identify patterns, making them increasingly
accurate at predicting outcomes and making decisions.\n\nAI is already being used
in a variety of industries, including healthcare, finance, and transportation. In
healthcare, AI is being used to develop personalized treatment plans for patients
based on their medical history and genetic makeup. In finance, AI is being used to
detect fraud and make investment recommendations. In transportation, AI is being
used to develop self-driving cars and improve traffic flow.\n\nDespite the many
benefits of AI, there are also concerns about its potential impact on society. Some
worry that AI could lead to job displacement, as machines become more capable of
performing tasks traditionally done by humans. Others worry that AI could be used
for malicious ";
```

3. To generate the text summary, pass the original text to the LLM using the `ML_GENERATE` method, with the second parameter set to `summarization`:

   ```
   select sys.ML_GENERATE(@text, JSON_OBJECT("task", "summarization",
   "model_id", "mistral-7b-instruct-v1"));
   ```

   A text summary generated by the LLM in response to your query is printed as output. It looks similar to the text output shown below:

   ```
   | {"text": " Artificial Intelligence (AI) is a rapidly growing field with the potential to
   revolutionize how we live and work. It refers to computer systems that can perform tasks
   requiring human intelligence, such as visual perception, speech recognition, decision-
   making, and language translation. Machine learning, a subset of AI, allows computers to
   learn from data without being explicitly programmed, making them increasingly accurate at
   predicting outcomes and making decisions. AI is already being used in healthcare, finance,
   and transportation industries for personalized treatment plans, fraud detection, and self-
   driving cars. However, there are concerns about its potential impact on society, including
   job displacement and malicious use."} |
   ```

Performing a Vector Search

Using the inbuilt vector store and retrieval-augmented generation (RAG), you can load and query unstructured documents stored in HeatWave Lakehouse using natural language within the HeatWave ecosystem.

The following sections describe how to setup and perform a vector search.

- Before You Begin
- Setting Up a Vector Store
- Updating the Vector Store
- Running RAG

Before You Begin

- Connect to your HeatWave database system. Ensure that you pass the `--sqlc` flag while connecting to the database:

  ```
  mysqlsh -u<Admin> -p<Password> -h<PrivateIP> --sqlc
  ```

- Upload the files that you want the vector store to ingest to the Object Storage bucket. For more information, see Create an Object Storage Bucket and Upload the File to an Object Storage Bucket.

Setting Up a Vector Store

The `vector_store_load` method creates and loads vector embeddings into the vector store table.

You can load the source files into the vector store using the following methods:

- Using the Uniform Resource Identifier
- Using a Pre-Authenticated Request

Using the Uniform Resource Identifier

This section describes how to load source documents from the Object Storage bucket into the vector table using the uniform resource identifier (URI) of the object.

However, to use this method, you need to enable the database system to access an OCI Object Storage bucket. For more information, see Resource Principals.

To set up a new vector store using an object URI, perform the following steps:

1. To create the vector store table, use a new or existing database:

   ```
   use <DBName>;
   ```

   Replace <***DBName***> with the database name.

2. If you're loading a vector store table on a database system for the first time, call the following method to create a schema used for task management:

   ```
   select mysql_task_management_ensure_schema();
   ```

3. To ingest the file from the object storage, create vector embeddings, and load the vector embeddings into HeatWave, use the `vector_store_load` method:

   ```
   call
   sys.vector_store_load('oci://<BucketName>@<Namespace>/<Path>/<Filename>',
   '{"table_name": "<EmbeddingsTableName>"}');
   ```

   Replace the following:

   - <***BucketName***>: the OCI Object Storage bucket name.
   - <***Namespace***> : the name of the Object Storage bucket namespace.
   - <***Path***>: path to the folder that contains the source file.
   - <***Filename***>: the filename with the file extension.
   - <***EmbeddingsTableName***>: the name you want for the vector embeddings table.

   For example:

   ```
   call
   sys.vector_store_load('oci://demo_bucket@demo_namespace/demo_folder/demo_fi
   le.pdf', '{"table_name": "demo_embeddings"}');
   ```

   This creates a task that runs in background and loads the vector embeddings into the specified table. The output of the `vector_store_load` method contains the following:

   - An ID of the task which was created.
   - A task query that you can use to track the progress of task.

4. After the task is completed, verify that embeddings are loaded in the vector embeddings table:

   ```
   select count(*) from <EmbeddingsTableName>;
   ```

   For example:

   ```
   select count(*) from demo_embeddings;
   ```

   If you see a numerical value in the output, your embeddings are successfully loaded in the table.

Using a Pre-Authenticated Request

This section describes how to ingest source documents from the object storage using pre-authenticated requests (PAR). Use this method if OCI Object Storage bucket access isn't enabled on your database system.

To learn how to create PAR for your object storage, see Creating a PAR Request in Object Storage.

If you're created a PAR for a folder or the object storage, then select **Enable Object Listing** in the **Create Pre-Authenticated Request** dialog to enable object listing.

To set up a new vector store, perform the following steps:

1. To create the vector store table, use a new or existing database:

   ```
   use <DBName>;
   ```

   Replace ***<DBName>*** with the database name.

2. If you're loading a vector store table on a database system for the first time, call the following method to create a schema used for task management:

   ```
   select mysql_task_management_ensure_schema();
   ```

3. To ingest the file from the object storage, create vector embeddings, and load the vector embeddings into HeatWave, use the `vector_store_load` method:

   ```
   call sys.vector_store_load('<PAR>', '{"table_name":
   "<EmbeddingsTableName>"}');
   ```

   Replace the following:

   - ***<PAR>***: PAR of the bucket, folder, or file that you want to use to set up the vector store.
   - ***<EmbeddingsTableName>***: the name you want for the vector embeddings table.

   For example:

   ```
   call sys.vector_store_load('https://demo.objectstorage.us-ashburn-
   1.oci.customer-oci.com/p/demo-url/n/demo/b/demo-bucket/o/', '{"table_name":
   "demo_embeddings"}');
   ```

   This creates a task that runs in background and loads the vector embeddings into the specified table. The output of the `vector_store_load` method contains the following:

   - An ID of the task which was created.
   - A task query that you can use to track the progress of task.

4. After the task is completed, verify that embeddings are loaded in the vector embeddings table:

   ```
   select count(*) from <EmbeddingsTableName>;
   ```

   For example:

   ```
   select count(*) from demo_embeddings;
   ```

   If you see a numerical value in the output, your embeddings are successfully loaded in the table.

Updating the Vector Store

To keep up with the changes and updates in the documents in your object storage, you must update the vector embeddings loaded in the vector store table on a regular basis. This ensures that the responses generated by HeatWave GenAI are not only accurate, but also up-to-date. And it deletes the embeddings that are no longer useful.

To update the vector embeddings, perform the following steps:

1. Delete the vector store table:

   ```
   drop table <EmbeddingsTableName>;
   ```

   Replace ***<EmbeddingsTableName>*** with the vector embeddings table name .

2. To create new embeddings for the updated documents, repeat the steps to Set Up the Vector Store.

Running RAG

HeatWave retrieves content from the vector store and provide that as context to the LLM. This process called as retrieval-augmented generation or RAG. This helps the LLM to produce more relevant and accurate results for your queries. The `ML_MODEL_LOAD` method loads the LLM, and the `ML_RAG` method runs RAG to generate accurate responses for your queries.

To run queries and generate accurate results using RAG, perform the following steps:

1. To load the LLM in HeatWave memory, use the `ML_MODEL_LOAD` method:

   ```
   call sys.ML_MODEL_LOAD('<LLMModel>', NULL);
   ```

   Replace ***<LLMModel>*** with the name of the LLM model that you want to use. The available models are: `mistral-7b-instruct-v1` and `llama2-7b-v1`.

   For example:

   ```
   call sys.ML_MODEL_LOAD('mistral-7b-instruct-v1', NULL);
   ```

2. To specify the table for retrieving the vector embeddings, set the `@options` session variable:

   ```
   set @options = JSON_OBJECT("vector_store",
   JSON_ARRAY("<DBName>.<EmbeddingsTableName>"));
   ```

   For example:

   ```
   set @options = JSON_OBJECT("vector_store",
   JSON_ARRAY("demo_db.demo_embeddings"));
   ```

3. To define your natural language query, set the session `@query` variable:

   ```
   set @query="<AddYourQuery>";
   ```

   Replace *<**AddYourQuery**>* with your natural language query.

   For example:
   ```
   set @query="What is AutoML?";
   ```

4. To retrieve the augmented prompt, use the `ML_RAG` method:

   ```
   call sys.ML_RAG(@query,@output,@options);
   ```

5. Print the output:

   ```
   select JSON_PRETTY(@output);
   ```

   Text-based content that is generated by the LLM in response to your query is printed as output. The output generated by RAG is comprised of two parts:

   - The `text` section contains the text-based content generated by the LLM as a response for your query.
   - The `citations` section shows the segments and documents it referred to as context.

   The output looks similar to the following:

   ```
     "text": " AutoML is a machine learning technique that automates the process of selecting,
   training, and evaluating machine learning models. It involves using algorithms and
   techniques to automatically identify the best model for a given dataset and optimize its
   hyperparameters without requiring manual intervention from data analysts or ML
   practitioners. AutoML can be used in various stages of the machine learning pipeline,
   including data preprocessing, feature engineering, model selection, hyperparameter tuning,
   and model evaluation.",
     "citations": [
       {
         "segment": "Oracle AutoML also produces high quality models very efficiently, which
   is achieved through a scalable design and intelligent choices that reduce trials at each
   stage in the pipeline.\n Scalable design: The Oracle AutoML pipeline is able to exploit
   both HeatWave internode and intranode parallelism, which improves scalability and reduces
   runtime.",
         "distance": 0.4262576103210449,
         "document_name":
   "https://objectstorage.<Region>.oraclecloud.com/n/<Namespace>/b/<BucketName>/o/<Path>/<File
   name>"
       },
       {
         "segment": "The HeatWave AutoML ML_TRAIN routine leverages Oracle AutoML technology
   to automate the process of training a machine learning model. Oracle AutoML replaces the
   laborious and time consuming tasks of the data analyst whose workflow is as follows:\n1.
   Selecting a model from a large number of viable candidate models.\n2.\n99",
         "distance": 0.4311879277229309,
         "document_name": " https://objectstorage.
   <Region>.oraclecloud.com/n/<Namespace>/b/<BucketName>/o/<Path>/<Filename>"
       },
       {
         "segment": "3.1 HeatWave AutoML Features HeatWave AutoML makes it easy to use machine
   learning, whether you are a novice user or an experienced ML practitioner. You provide the
   data, and HeatWave AutoML analyzes the characteristics of the data and creates an optimized
   machine learning model that you can use to generate predictions and explanations.",
         "distance": 0.4441382884979248,
         "document_name": "https://objectstorage.
   <Region>.oraclecloud.com/n/<Namespace>/b/<BucketName>/o/<Path>/<Filename>"
       }
     ],
     "vector_store": [
       "demo_db.demo_embeddings"
     ]
   }
   ```

   To continue running more queries in the same session, repeat steps **3** to **5**.

Running HeatWave Chat

You can use HeatWave Chat to simulate human-like conversations where you can get responses for multiple queries in the same session. HeatWave Chat is a conversational agent that utilizes large language models (LLMs) to understand inputs and responds in natural manner. It extends the text generation by using a chat history that lets you ask follow-up questions, and uses the vector search functionality to draw its knowledge from the inbuilt vector store. The responses generated by HeatWave Chat are quick and secure as all the communication and processing happens within the HeatWave service.

Before You Begin

- If you want to ask specific questions about the information available in your proprietary documents available in the vector store, complete the steps to Set Up a Vector Store.

Running the Chat

When you run HeatWave Chat, it automatically loads the LLM model. If you don't have a vector store set up, then HeatWave Chat uses information available in public data sources to generate a response for your query. However, if you have a vector store set up, then HeatWave Chat by default performs a global context search across all the loaded vector store tables to generate a response for your query.

To run HeatWave Chat, perform the following steps:

1. To delete previous chat output and state if any, reset the `@chat_options` session variable:

   ```
   set @chat_options=NULL;
   ```

2. Then, add your query to HeatWave Chat by using the `heatwave_chat` method:

   ```
   call sys.heatwave_chat("<YourQuery>");
   ```

   For example:

   ```
   call sys.heatwave_chat("What is HeatWave AutoML?");
   ```

   The output looks similar to the following:

   ```
   |  HeatWave AutoML is a feature of MySQL HeatWave that makes it easy to use machine
   learning, whether you are a novice user or an experienced ML practitioner. It analyzes the
   characteristics of the data and creates an optimized machine learning model that can be
   used to generate predictions and explanations. The data and models never leave MySQL
   HeatWave, saving time and effort while keeping the data and models secure. HeatWave AutoML
   is optimized for HeatWave shapes and scaling, and all processing is performed on the
   HeatWave Cluster. |
   ```

   Repeat this step to ask follow-up questions using the `heatwave_chat` method:

   ```
   call sys.heatwave_chat("What learning algorithms does it use?");
   ```

   The output looks similar to the following:

   ```
   |  HeatWave AutoML uses a variety of machine learning algorithms. It leverages Oracle
   AutoML technology which includes a range of algorithms such as decision trees, random
   forests, neural networks, and support vector machines (SVMs). The specific algorithm used
   by HeatWave AutoML depends on the characteristics of the data being analyzed and the goals
   of the model being created. |
   ```

Viewing Chat Session Details

To view the chat session details, perform the following step:

- Inspect the `@chat_options` session variable:

  ```
  select JSON_PRETTY(@chat_options);
  ```

  The output includes the following details about a chat session:

  - **Vector store tables:** in the database which were referenced by HeatWave Chat.
  - **Text segments:** that were retrieved from the vector store and used as context to prepare responses for your queries.
  - **Chat history:** which includes both your queries and responses generated by HeatWave Chat.
  - **LLM model details:** which was used by the method to generate responses.

The output looks similar to the following:

```
| {
  "tables": [
    {
      "table_name": "`quickstart_embeddings`",
      "schema_name": "`quickstart_db`"
    }
  ],
  "response": " HeatWave AutoML uses a variety of machine learning algorithms. It leverages
Oracle AutoML technology which includes a range of algorithms such as decision trees,
random forests, neural networks, and support vector machines (SVMs). The specific algorithm
used by HeatWave AutoML depends on the characteristics of the data being analyzed and the
goals of the model being created.",
  "documents": [
    {
      "id":
"https://objectstorage.<Region>.oraclecloud.com/n/<Namespace>/b/<BucketName>/o/<Path>/heatw
ave-en.a4.pdf",
      "title": "heatwave-en.a4.pdf",
      "segment": "3.1 HeatWave AutoML Features HeatWave AutoML makes it easy to use machine
learning, whether you are a novice user or an experienced ML practitioner. You provide the
data, and HeatWave AutoML analyzes the characteristics of the data and creates an optimized
machine learning model that you can use to generate predictions and explanations.",
      "distance": 0.18456566333770752
    },
    {
      "id":
"https://objectstorage.<Region>.oraclecloud.com/n/<Namespace>/b/<BucketName>/o/<Path>/heatw
ave-en.a4.pdf",
      "title": "heatwave-en.a4.pdf",
      "segment": "The HeatWave AutoML ML_TRAIN routine leverages Oracle AutoML technology
to automate the process of training a machine learning model. Oracle AutoML replaces the
laborious and time consuming tasks of the data analyst whose workflow is as follows:\n1.
Selecting a model from a large number of viable candidate models.\n2. For each model,
tuning hyperparameters.\n3. Selecting only predictive features to speed up the pipeline and
reduce over-fitting.\n99",
      "distance": 0.22687965631484985
    },
    {
      "id":
"https://objectstorage.<Region>.oraclecloud.com/n/<Namespace>/b/<BucketName>/o/<Path>/heatw
ave-en.a4.pdf",
      "title": "heatwave-en.a4.pdf",
      "segment": "3.1.1 HeatWave AutoML Supervised Learning\nHeatWave AutoML supports
supervised machine learning. That is, it creates a machine learning model by analyzing a
labeled dataset to learn patterns that enable it to predict labels based on the features of
the dataset. For example, this guide uses the Census Income Data Set in its examples, where
features such as age, education, occupation, country, and so on, are used to predict the
income of an individual (the label).",
      "distance": 0.2275727391242981
    }
  ],
  "chat_history": [
    {
      "user_message": "What is HeatWave AutoML?",
      "chat_query_id": "99471681-387f-11ef-96d7-020017331ed6",
      "chat_bot_message": " HeatWave AutoML is a feature of MySQL HeatWave that makes it
easy to use machine learning, whether you are a novice user or an experienced ML
practitioner. It analyzes the characteristics of the data and creates an optimized machine
learning model that can be used to generate predictions and explanations. The data and
models never leave MySQL HeatWave, saving time and effort while keeping the data and models
secure. HeatWave AutoML is optimized for HeatWave shapes and scaling, and all processing is
performed on the HeatWave Cluster."
    },
    {
      "user_message": "What learning algorithms does it use?",
      "chat_query_id": "c59140f5-387f-11ef-96d7-020017331ed6",
      "chat_bot_message": " HeatWave AutoML uses a variety of machine learning algorithms.
It leverages Oracle AutoML technology which includes a range of algorithms such as decision
trees, random forests, neural networks, and support vector machines (SVMs). The specific
algorithm used by HeatWave AutoML depends on the characteristics of the data being analyzed
and the goals of the model being created."
    }
  ],
  "model_options": {
    "model_id": "mistral-7b-instruct-v1"
  },
  "request_completed": true
} |
```

Troubleshooting Issues and Errors

This section describes some common issues and errors you might see while using HeatWave GenAI and provides their workaround.

- **Issue**: Command fails to run due to unidentified characters .

  Workaround: Copy and paste the command in a text editor to clear the text formatting, and fix the text formatting and ensure that correct characters, such as quotes and double quotes are used, and run the command again.

- **Issue**: When you try to verify whether the vector embeddings were correctly loaded, if you see a message which indicates that the vector embeddings or table didn't load in HeatWave, then it could be due one of the following reasons:

  - The task that loads the vector embeddings into the vector store table might still be running.

    Workaround: Check the task status by using the query that was printed by the `vector_store_load` method:

    ```
    select * from mysql_task_management.task_status where id = <TaskID>;
    ```

    Or, to see the log messages, check the task logs table:

    ```
    select * from mysql_task_management.task_log where task_id =
    <TaskID>;
    ```

    Replace <***TaskID***> with the ID for the task which was printed by the `vector_store_load` method.

  - The folder you're trying to load might contain unsupported format files or the file that you're trying to load might be of an unsupported format.

    Workaround: The supported file formats are: PDF, TXT, PPT, HTML, and DOC.

    If you find unsupported format files, then try one of the following:

    - Delete the files with unsupported formats from the folder, and run the `vector_store_load` command again to load the vector embeddings into the table again.
    - Move the files with supported formats to another folder, create a new PAR and run the `vector_store_load` command with the new PAR to load the vector embeddings into the table again.

- **Issue**: the `vector_store_load` command fails unexpectedly

  Workaround: Ensure that you use the `--sqlc` flag when you connect to your database system:

  ```
  mysqlsh -u<Admin> -p<Password> -h<PrivateIP> --sqlc
  ```

  Replace the following:

  - <***Admin***>: the admin name.
  - <***Password***>: the database system password.
  - <***PrivateIP***>: The private IP address of the database system.

If you still aren't able to ingest files using `vector_store_load`, then try using the `heatwave_load` method.

Ingesting Files Using **`heatwave_load`** Method

To ingest files using the heatwave_load method, do the following:

1. In your HeatWave Database System, create and use a new database:

   ```
   create database <DBName>;
   use <DBName>;
   ```

   Replace <***DBName***> with the name you want for the new database.

2. To ingest the file from the object store and create vector embeddings in a new table, set the `@dl_tables` session variable:

   ```
   set @dl_tables = '[
     {
       "db_name": "<DBName>",
       "tables": [
       {
           "table_name": "<EmbeddingsTableName>",
           "engine_attribute": {
              "dialect": {"format": "<FileFormat>"},
               "file": [
       {"par": "<PAR>"}
   ]
           }
         }
       ]
   }]';
   ```

   Replace the following:

   - <***DBName***>: the database name.
   - <***EmbeddingsTableName***>: the name you want for the table where the vector embeddings are stored.
   - <***FileFormat***>: the format of the file you uploaded to the object store bucket. The supported file formats are `pdf`, `ppt`, `txt`, and `doc`.
   - <***PAR***>: the pre-authenticated request (PAR) detail of the bucket, folder, or file that you want to use to set up the vector store.

   To learn how to create PAR for your object storage, see [Creating a PAR request in Object Storage]().

   Note: If you're created a PAR for a folder or the object store, then select **Enable Object Listing** to enable object listing in the **Create Pre-Authenticated Request** dialog while creating the PAR.

For example:

```
set @dl_tables = '[
  {
    "db_name": "demo_db",
    "tables": [
    {
        "table_name": "demo_embeddings",
        "engine_attribute": {
           "dialect": {"format": "pdf"},
            "file": [
    {"par": "https://demo.objectstorage.us-ashburn-1.oci.customer-
oci.com/p/demo-url/n/demo/b/demo_bucket/o/heatwave-en.a4.pdf"}
]
        }
      }
    ]
}]';
```

3. To prepare for loading the vector embeddings into the HeatWave system, set the `@options` session variable:

   ```
   set @options = JSON_OBJECT('mode', 'normal');
   ```

4. To load the vector embeddings into HeatWave, use the `heatwave_load` method:

   ```
   call sys.heatwave_load(CAST(@dl_tables AS JSON), @options);
   ```

   This creates and stores the vector embeddings in the specified table.

5. Verify that embeddings are loaded in the vector embeddings table:

   ```
   select count(*) from <EmbeddingsTableName>;
   ```

   For example:

   ```
   select count(*) from demo_embeddings;
   ```

   If you see a numerical value in the output, your embeddings are successfully loaded in the table.

Contacting Support

If you run into issues that you're not able to resolve, contact the Support team.

Quickstart: Setting Up a GenAI-Powered Help Chat

This quickstart shows how to use the vector store functionality and use HeatWave Chat to create a GenAI-powered Help chat that refers to the HeatWave user guide to respond to HeatWave related queries.

Note: This quickstart assumes that you're familiar with the HeatWave database systems.

This quickstart contains the following sections:

Before You Begin

- Review the Requirements.

Setting Up the Object Storage Bucket

1. Download the HeatWave user guide PDF (A4) - 1.7Mb.
2. Create an Object Storage Bucket with the name `quickstart_bucket`.
3. Upload the PDF file to the Object Storage Bucket using the prefix `quickstart/` to create a new folder by the name `quickstart`.

Setting Up the Environment

- Connect to your HeatWave database system.

  Ensure that you pass the `--sqlc` flag while connecting to the database:

  ```
  mysqlsh -u<Admin> -p<Password> -h<PrivateIP> --sqlc
  ```

  Replace the following:

  - `<Admin>`: the admin name.
  - `<Password>`: the database system password.
  - `<PrivateIP>`: The private IP address of the database system.

- If not already done, add a HeatWave Cluster to your database system.
- If not already done, enable HeatWave Lakehouse on the database system.
- Enable the database system to access an OCI Object Storage bucket. For more information, see Resource Principals.

Setting Up the Vector Store

1. Create and use a new database:

   ```
   create database quickstart_db;
   use quickstart_db;
   ```

2. Call the following method to create a schema used for task management:

   ```
   select mysql_task_management_ensure_schema();
   ```

3. Create the vector table and load the source document:

```
call
sys.vector_store_load('oci://quickstart_bucket@<Namespace>/quickstart/heatwa
ve-en.a4.pdf', '{"table_name": "quickstart_embeddings"}');
```

Replace <***Namespace***> with the name of the name of the Object Storage bucket namespace that you're using.

This creates a task in the background which loads the vector embeddings into the specified table `quickstart_embeddings`.

4. To track the progress of the task, run the task query displayed on the screen:

```
select id, name, message, progress, status,
scheduled_time,estimated_completion_time, estimated_remaining_time,
progress_bar FROM mysql_task_management.task_status WHERE id=<TaskID>\G
```

Replace <***TaskID***> with the displayed task ID.

The output looks similar to the following:

```
                       id: 1
                     name: Vector Store Loader
                  message: Task starting.
                 progress: 0
                   status: RUNNING
           scheduled_time: 2024-07-02 14:42:38
estimated_completion_time: NULL
 estimated_remaining_time: NULL
             progress_bar: __________
```

5. After the task status has changed to `Completed`, verify that embeddings are loaded in the vector embeddings table:

```
select count(*) from quickstart_embeddings;
```

If you a numerical value in the output, your embeddings are successfully loaded in the table.

Starting a Chat Session

1. Clear the previous chat history and states:

```
set @chat_options=NULL;
```

2. Ask your question using HeatWave Chat:

```
call sys.heatwave_chat("What is HeatWave AutoML?");
```

The `heatwave_chat` method automatically loads the LLM and runs a contextual search on the available vector stores by default. The output is similar to the following:

```
|  HeatWave AutoML is a feature of MySQL HeatWave that makes it easy to use machine
learning, whether you are a novice user or an experienced ML practitioner. It analyzes the
characteristics of the data and creates an optimized machine learning model that can be
used to generate predictions and explanations. The data and models never leave MySQL
HeatWave, saving time and effort while keeping the data and models secure. HeatWave AutoML
is optimized for HeatWave shapes and scaling, and all processing is performed on the
HeatWave Cluster. |
```

3. Ask a follow-up question:

```
call sys.heatwave_chat("How to set it up?");
```

The output is similar to the following:

```
|  To set up HeatWave AutoML in MySQL HeatWave, you need to follow these steps:

1. Ensure that you have an operational MySQL DB System and are able to connect to it using
a MySQL client. If not, complete the steps described in Getting Started with MySQL
HeatWave.
2. Ensure that your MySQL DB System has an operational HeatWave Cluster. If not, complete
the steps described in Adding a HeatWave Cluster.
3. Obtain the MySQL user privileges described in Section 3.2, Before You Begin.
4. Prepare and load training and test data. See Section 3.4, Preparing Data.
5. Train a machine learning model. See Section 3.5, Training a Model.
6. Make predictions using the trained model. See Section 3.6, Making Predictions.
7. Generate explanations for the predictions made by the model. See Section 3.7, Generating
Explanations.
8. Monitor and manage the performance of the model. See Section 3.8, Monitoring and
Managing Performance. |
```

You can continue asking follow-up questions in the same chat session.

Cleaning Up

To avoid being billed for the resources that you created for this quickstart, perform the following steps:

1. Delete the database that you created:

   ```
   drop database quickstart_db;
   ```

2. Delete `quickstart_bucket`. For more information, see Deleting the Object Storage Bucket.

Connect with us

Call +**1.800.ORACLE1** or visit **oracle.com**. Outside North America, find your local office at: **oracle.com/contact**.

blogs.oracle.com facebook.com/oracle twitter.com/oracle