

SQL AI Assistant in DataHub (Preview)

Date published: 2024-02-29

Date modified: 2024-02-29

Legal Notice

© Cloudera Inc. 2024. All rights reserved.

The documentation is and contains Cloudera proprietary information protected by copyright and other intellectual property rights. No license under copyright or any other intellectual property right is granted herein.

Unless otherwise noted, scripts and sample code are licensed under the Apache License, Version 2.0.

Copyright information for Cloudera software may be found within the documentation accompanying each component in a particular release.

Cloudera software includes software from various open source or other third party projects, and may be released under the Apache Software License 2.0 ("ASLv2"), the Affero General Public License version 3 (AGPLv3), or other license terms.

Other software included may be released under the terms of alternative open source licenses. Please review the license and notice files accompanying the software for additional licensing information.

Please visit the Cloudera software product page for more information on Cloudera software. For more information on Cloudera support services, please visit either the Support or Sales page. Feel free to contact us directly to discuss your specific needs.

Cloudera reserves the right to change any products at any time, and without notice. Cloudera assumes no responsibility nor liability arising from the use of products, except as expressly agreed to in writing by Cloudera.

Cloudera, Cloudera Altus, HUE, Impala, Cloudera Impala, and other Cloudera marks are registered or unregistered trademarks in the United States and other countries. All other trademarks are the property of their respective owners.

Disclaimer: EXCEPT AS EXPRESSLY PROVIDED IN A WRITTEN AGREEMENT WITH CLOUDERA, CLOUDERA DOES NOT MAKE NOR GIVE ANY REPRESENTATION, WARRANTY, NOR COVENANT OF ANY KIND, WHETHER EXPRESS OR IMPLIED, IN CONNECTION WITH CLOUDERA TECHNOLOGY OR RELATED SUPPORT PROVIDED IN CONNECTION THEREWITH. CLOUDERA DOES NOT WARRANT THAT CLOUDERA PRODUCTS NOR SOFTWARE WILL OPERATE UNINTERRUPTED NOR THAT IT WILL BE FREE FROM DEFECTS NOR ERRORS, THAT IT WILL PROTECT YOUR DATA FROM LOSS, CORRUPTION NOR UNAVAILABILITY, NOR THAT IT WILL MEET ALL OF CUSTOMER'S BUSINESS REQUIREMENTS. WITHOUT LIMITING THE FOREGOING, AND TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, CLOUDERA EXPRESSLY DISCLAIMS ANY AND ALL IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, QUALITY, NON-INFRINGEMENT, TITLE, AND FITNESS FOR A PARTICULAR PURPOSE AND ANY REPRESENTATION, WARRANTY, OR COVENANT BASED ON COURSE OF DEALING OR USAGE IN TRADE.

Contents

Legal Notice	2
Contents	3
About the Hue SQL AI Assistant	4
Which AI models and services does Hue use?	4
What data is shared with the LLM models?	5
Prerequisites for enabling the SQL AI Assistant	5
Configuring the SQL AI Assistant on the Microsoft Azure OpenAI Service	5
Configuring the SQL AI Assistant on the OpenAI Service	7
Service and model-related configurations for setting up the SQL AI Assistant in DataHub	9
How to use the SQL AI Assistant?	10
Generating SQL from NQL	10
Editing the query in natural language	11
Getting an explanation of a SQL query in natural language	11
Optimizing a query	12
Fixing a SQL query	13
Limitations	14

About the Hue SQL AI Assistant

A SQL AI Assistant has been integrated into Hue with the capability to leverage the power of Large Language Models (LLMs) for various SQL tasks. It helps you to create, edit, optimize, fix, and succinctly summarize queries using natural language and makes SQL development faster, easier, and less error-prone. The Hue AI assistant is available with Cloudera Runtime 7.2.18 and higher.

Note: *The Hue SQL AI assistant is in technical preview and not recommended for use in production deployments. Cloudera recommends that you try this feature in test and development environments.*

Which AI models and services does Hue use?

The AI Assistant supports various LLMs and hosting services. The model can run in the infrastructure of a service provider, and the AI Assistant can be configured to use them remotely. Cloudera has tested with GPT running in Azure OpenAI, and the following service-model combinations are supported

Service Provider	Model	Model versions
OpenAI	OpenAI GPT	gpt-3.5-turbo gpt-3.5-turbo-16k
Microsoft Azure	OpenAI GPT	gpt-3.5-turbo gpt-3.5-turbo-16k
Amazon Bedrock	Anthropic Claude	anthropic.claude-v1 anthropic.claude-v2
Amazon Bedrock	Amazon Titan	amazon.titan-text-express-v1

Note: *Cloudera recommends using the Hue AI assistant with the Azure OpenAI service.*

For augmenting the results, the SQL AI Assistant uses a Retrieval Augmented Generation (RAG)-based architecture. It uses the sentence-transformer library for semantic search, and Hue can be configured with any of [these](#) pre-trained models for better multi-lingual support. By default, **all-MiniLM-L6-v2** models are used.

Embedding Model	Language Support
all-MiniLM-L6-v2	English only

distiluse-base-multilingual-cased-v1	Arabic, Chinese, Dutch, English, French, German, Italian, Korean, Polish, Portuguese, Russian, Spanish, and Turkish.
--------------------------------------	--

What data is shared with the LLM models?

The following details are shared with the Large Language Models (LLMs):

- Everything that a user inputs
- Dialect in use
- Table details such as table name, column names, column data types and related keys, partitions, and constraints that the logged-in user has access to.
- Three sample rows from the tables (as per the best practices specified in [Rajkumar et al. 2022](#))

Prerequisites for enabling the SQL AI Assistant

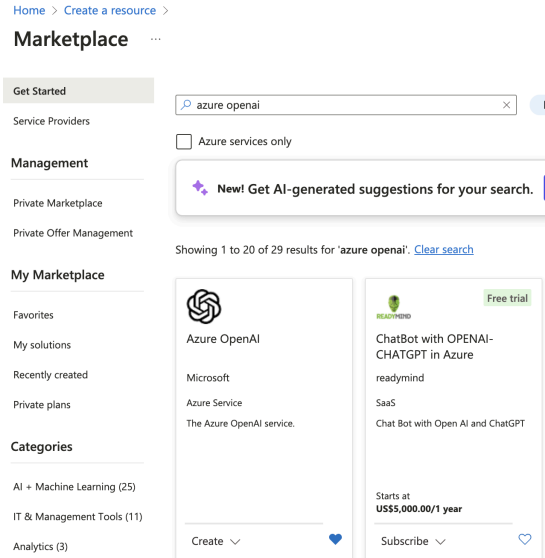
As an administrator, you must obtain clearance from your organization's infosec team to make sure it is safe to use the SQL AI Assistant because some of the table metadata and data, as mentioned in the previous section, is shared with the LLM.

Configuring the SQL AI Assistant on the Microsoft Azure OpenAI Service

Microsoft Azure allows dedicated deployments of OpenAI GPT models. Using Azure's OpenAI service is much more secure than the publicly hosted OpenAI APIs because the data can be processed in your Virtual Private Cloud (VPC). Due to security considerations, Cloudera recommends that you use GPT models in Hue SQL AI Assistant with Azure's OpenAI service.

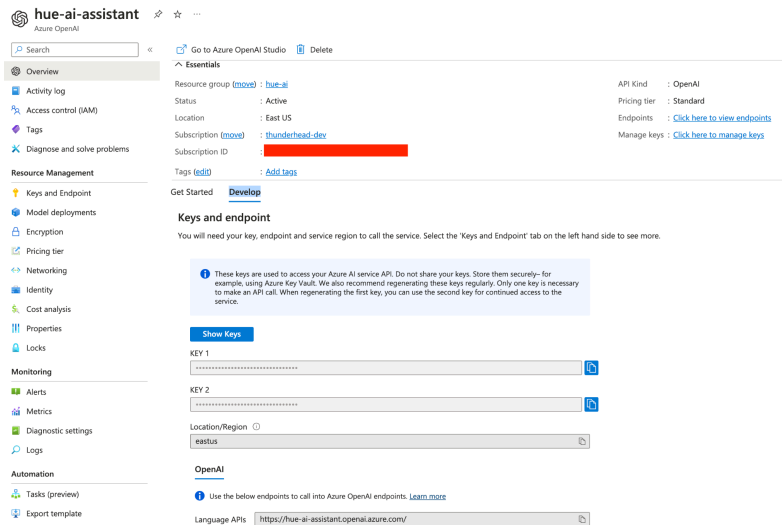
Steps

1. Obtain a Microsoft Azure subscription by working with your organization's IT team. Subscriptions vary based on your team and purpose.
2. Register to access the Azure OpenAI service.
Azure OpenAI requires registration and is currently only available to approved enterprise customers and partners. Customers who wish to use Azure OpenAI are required to submit a [registration form](#).
3. Create an Azure OpenAI resource in the Azure portal.



4. Obtain the resource URL and resource keys from the **Develop** tab under the resource details page.

You can use any one of the two keys as shown in the following image:



5. Go to Azure OpenAI Studio at <https://oai.azure.com/portal> and create your deployment under **Management > Deployments**. Select **gpt-35-turbo-16k** or higher.
6. Enable the Hue SQL AI Assistant as follows:
 - a. Log in to the CDP Management Console as an Administrator.
 - b. Go to **Environments** and select your environment.
 - c. Go to the **Data Lake** tab and click on the CM URL to open Cloudera Manager.
 - d. Go to **Clusters > Hue service > Configuration** and select add the following lines in the **Hue Service Advanced Configuration Snippet (Safety Valve)** for **hue_safety_valve.ini** field:
[desktop]

```
[[ai_interface]]
  service='azure'
  model_name=' [***DEPLOYMENT-NAME***] '
  base_url="https:// [***RESOURCE***].openai.azure.com/"
  token=" [***RESOURCE-KEY***] "
```

- e. Click **Save Changes**.

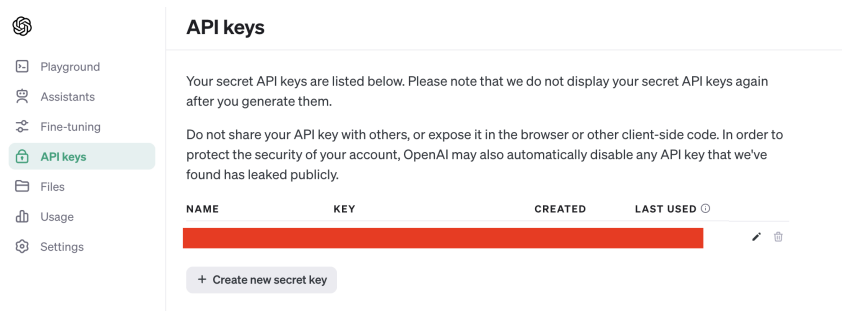
Configuring the SQL AI Assistant on the OpenAI Service

Before you begin

You must have created an account with the OpenAI platform.

Steps

1. Log in to the OpenAI portal.
2. Obtain the API key by navigating to the API keys from the left navigation pane. The key is required to integrate Hue SQL AI Assistant with the OpenAI service.



3. Enable the Hue SQL AI Assistant as follows:
 - a. Log in to the CDP Management Console as an Administrator.
 - b. Go to **Environments** and select your environment.
 - c. Go to the **Data Lake** tab and click on the CM URL to open Cloudera Manager.
 - d. Go to **Clusters > Hue service > Configuration** and select add the following lines in the **Hue Service Advanced Configuration Snippet (Safety Valve)** for **hue_safety_valve.ini** field:

```
[desktop]
  [[ai_interface]]
    service='openai'
    token=' [***API-KEY***] '
```

- e. Click **Save Changes**.

Configuring the SQL AI Assistant on the Amazon Bedrock Service

Amazon Bedrock is a fully managed service that makes foundation models from leading AI startups and Amazon available through an API.

Before you begin

You must have an AWS account with Bedrock access.

Steps

1. Log in to the Amazon Bedrock service.
2. Obtain your access key and secret as follows:
 - a. Go to the IAM console: <https://console.aws.amazon.com/iam>
 - b. Click on **Users** from the left menu and select the user you want to access.
 - c. Click on **Security credentials**.
 - d. Go to the **Access keys** section and note the access keys.
3. Establish Anthropic Claude access.

Claude from Anthropic is one of the best models available in Bedrock for SQL-related tasks. By default, Claude is not available on Bedrock. You need to place a special request for Claude. Once you have access, you can try Claude in the text playground under the Amazon Bedrock service. If you are in the us-east-1 region, this must take you to <https://us-east-1.console.aws.amazon.com/bedrock/home?region=us-east-1#/text-playground>.
4. Enable the Hue SQL AI Assistant as follows:
 - a. Log in to the CDP Management Console as an Administrator.
 - b. Go to **Environments** and select your environment.
 - c. Go to the **Data Lake** tab and click on the CM URL to open Cloudera Manager.
 - d. Go to **Clusters > Hue service > Configuration** and select add the following lines in the **Hue Service Advanced Configuration Snippet (Safety Valve) for hue_safety_valve.ini** field:

```
[aws]
    [[bedrock_account]]
        access_key_id=' [***ACCESS-KEY***] '
        secret_access_key=' [***SECRET-KEY***] '
        region='us-east-1'

[desktop]
    [[ai_interface]]
        service='bedrock'
        model='claude'
```

5. Click **Save Changes**.

Service and model-related configurations for setting up the SQL AI Assistant in DataHub

You can configure the AI services and models you want to use by going to **Cloudera Manager > Clusters > Hue service > Configurations > Hue Service Advanced Configuration Snippet (Safety Valve) for hue_safety_valve.ini** and adding the following lines:

```
[desktop]
  [[ai_interface]]
    [***CONFIG-KEY1***]=' [***VALUE***] '
    [***CONFIG-KEY2***]=' [***VALUE***] '
  [[semantic_search]]
    [***CONFIG-KEY1***]=' [***VALUE***] '
    [***CONFIG-KEY2***]=' [***VALUE***] '
```

Specify the service and model-related configurations under the `[[ai_interface]]` section as listed in the following table:

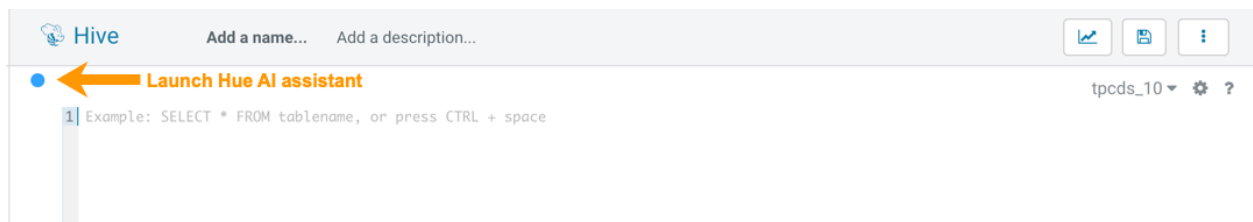
AI interface config key	Description
service	API service to be used for AI tasks. AI is disabled when a service is not configured. For example, <code>azure</code> , <code>openai</code> , <code>bedrock</code> , and <code>ai_assistant</code> .
trusted_service	Indicates whether the LLM is trusted or not. Turn on to disable the warning. The default value is <code>"True"</code> .
model	The AI model you want to use for AI tasks. For example, <code>gpt</code> , <code>llama</code> .
model_name	The fully qualified name of the model to be used. For example, <code>gpt-3.5-turbo-16k</code> .
base_url	Service API base URL.
add_table_data	When enabled, sample rows from the table are added to the prompt. The default value is <code>"True"</code> .
table_data_cache_size	Size of the LRU cache used for storing table sample data.
auto_fetch_table_meta_limit	Number of tables to load from a database, initially.
token	Service API secret token.
token_script	Provides a secure way to get the service API secret token.

Specify the semantic search-related configurations used for RAG under the `[[semantic_search]]` section, as listed in the following table:

Semantic search config key	Description
relevancy	The technology you want to use for semantic search. Acceptable values are <code>vector_search</code> or <code>vector_db</code> .
embedding_model	The model you want to use for data-embedding. This must be compatible with SentenceTransformer.
cache_size	Size of the LRU cache used for storing embedding.

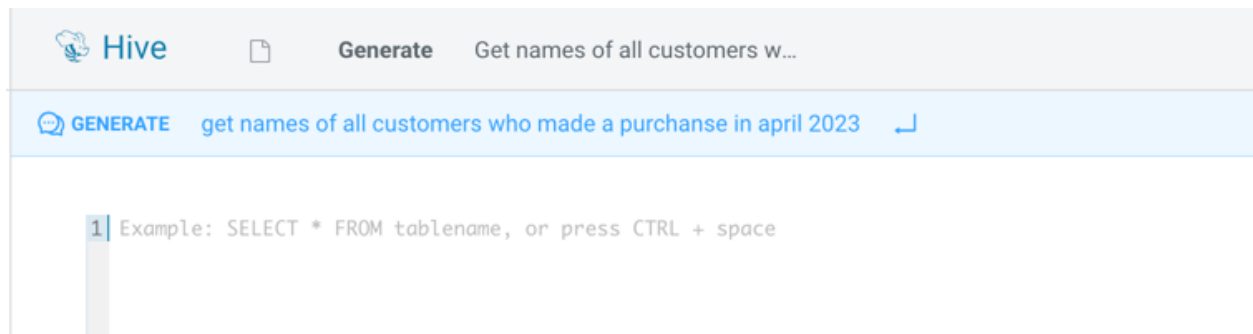
How to use the SQL AI Assistant?

To launch the Hue AI assistant, click the blue dot on the Hue web interface interface as shown in the following image:



Generating SQL from NQL

Click **GENERATE** and type the query in natural language as shown in the following image:



The SQL query is generated as shown in the following image. You can insert it into the editor by clicking **Insert** or you can copy the query into the editor.

Suggestion ✕

Autoformat SQL Include prompt as comment Replace comments

```
1 /* NQL: get names of all customers who made a purchase in april 2023 */
2 SELECT
3   c_first_name,
4   c_last_name
5 FROM
6   customer
7 WHERE
8   c_customer_sk IN (
9     SELECT
10    ss_customer_sk
11 FROM
12   store_sales
13 WHERE
14   ss_sold_date_sk >= 2451179
15   AND ss_sold_date_sk <= 2451208
16 )
```

ASSUMPTIONS

- The "purchase" in the NQL statement is a typo and should be "purchase".
- The date range for April 2023 is assumed to be from April 1, 2023 (2451179) to April 30, 2023 (2451208).

Insert Copy to clipboard Cancel

Editing the query in natural language

Click **Edit** as shown in the following image:

The screenshot shows the Hive interface with a search bar containing "Generate" and "Get names of all customers w...". Below the search bar is a menu with buttons for "GENERATE", "EDIT" (highlighted with an orange box), "EXPLAIN", "OPTIMIZE", and "FIX". The main area displays a SQL query snippet:

```
1 /* NQL: get names of all customers who made a purchase in april 2023 */
2 SELECT
3   c_first_name,
```

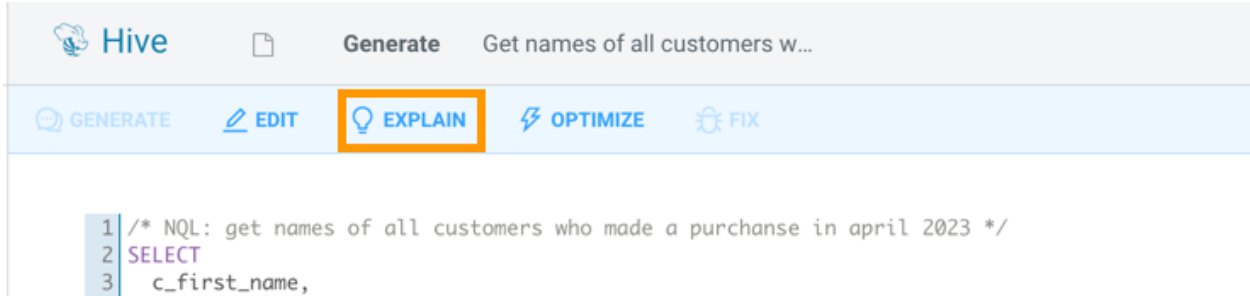
Make the required changes and press enter:

The screenshot shows the Hive interface with the search bar updated to "EDIT get names of all customers who made a purchase in June|2023". The main area displays the updated SQL query snippet:

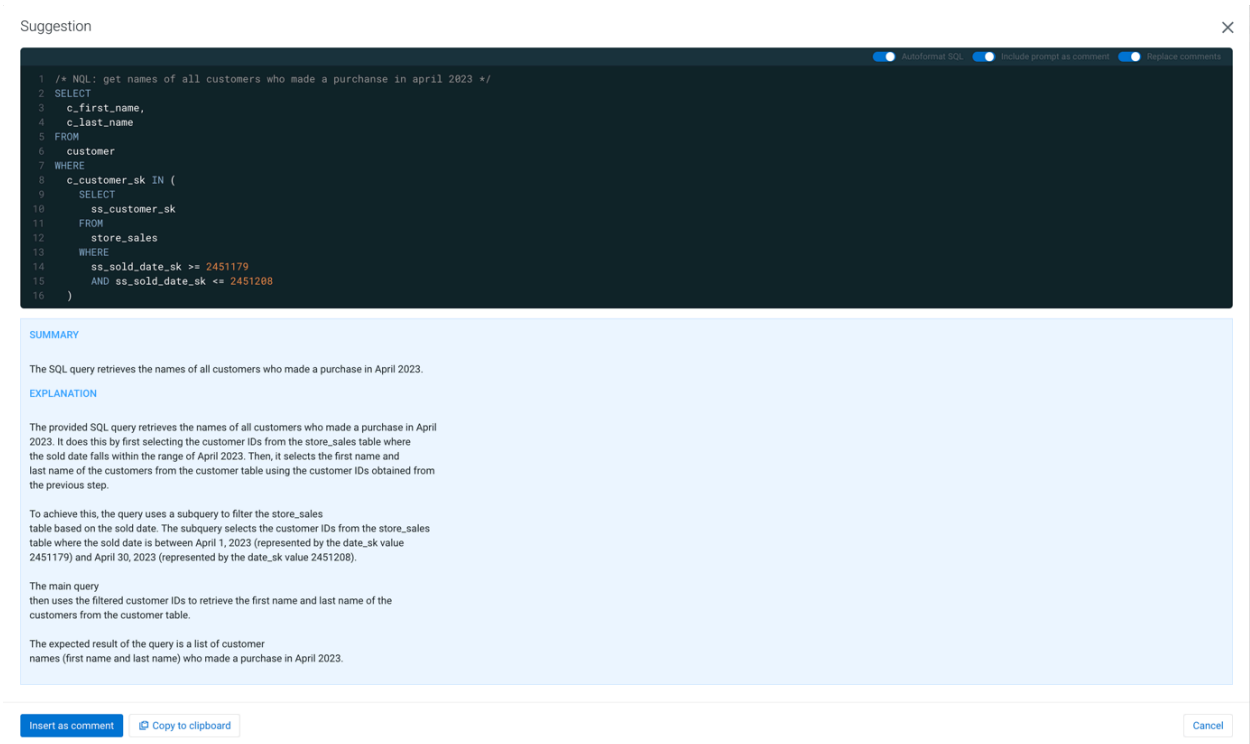
```
1 /* NQL: get names of all customers who made a purchase in april 2023 */
```

Getting an explanation of a SQL query in natural language

To understand complex queries in natural language, click **EXPLAIN** as shown in the following image:

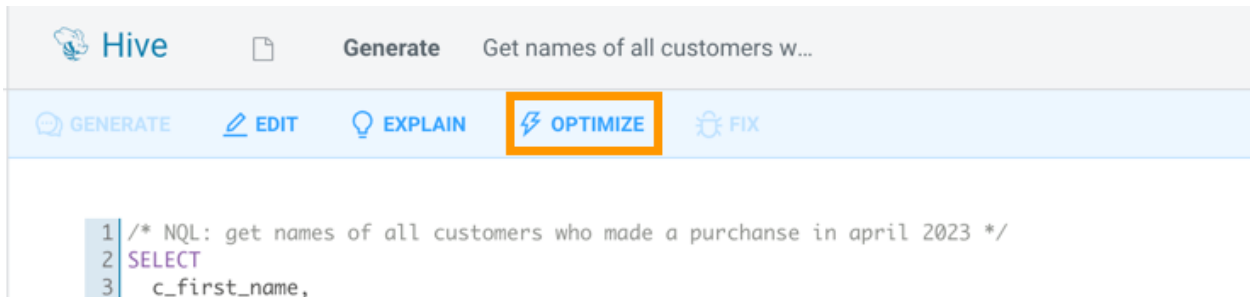


The LLM generates the explanation of the SQL query as shown in the following image:



Optimizing a query

Click **OPTIMIZE** to improve an existing query as shown in the following image:



The following example shows how Hue identifies issues, optimizes a query, and provides an explanation:

Suggestion X

```
1 SELECT
2   COUNT(*)
3 FROM
4 (
5   SELECT
6     *
7 FROM
8   (
9     SELECT
10      *
11 FROM
12   item
13 WHERE
14   i_current_price > 1
15 ) AS b
16 WHERE
17   b.i_category = 'Children'
18 ) AS c
19 WHERE
20   c.i_color = 'red';
```

```
1 SELECT
2   COUNT(*)
3 FROM
4 + item
5 WHERE
6 + i_current_price > 1
7 + AND i_category = 'Children'
8 + AND i_color = 'red';
```

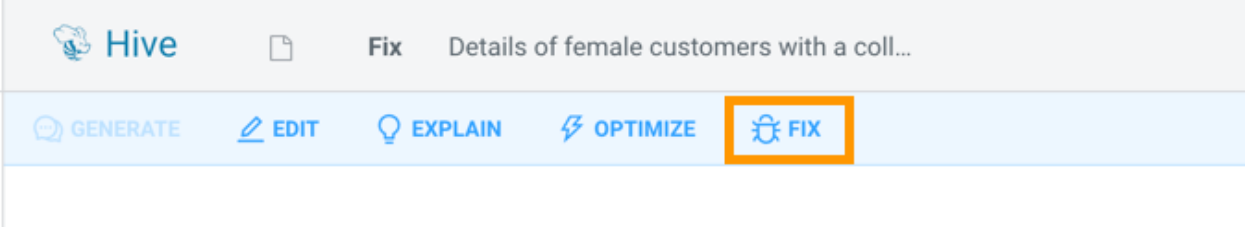
EXPLANATION

The original query can be optimized by removing unnecessary subqueries. Instead of using nested subqueries, we can combine the conditions in the WHERE clause of a single query. This eliminates the need for multiple levels of subqueries and improves the query performance.

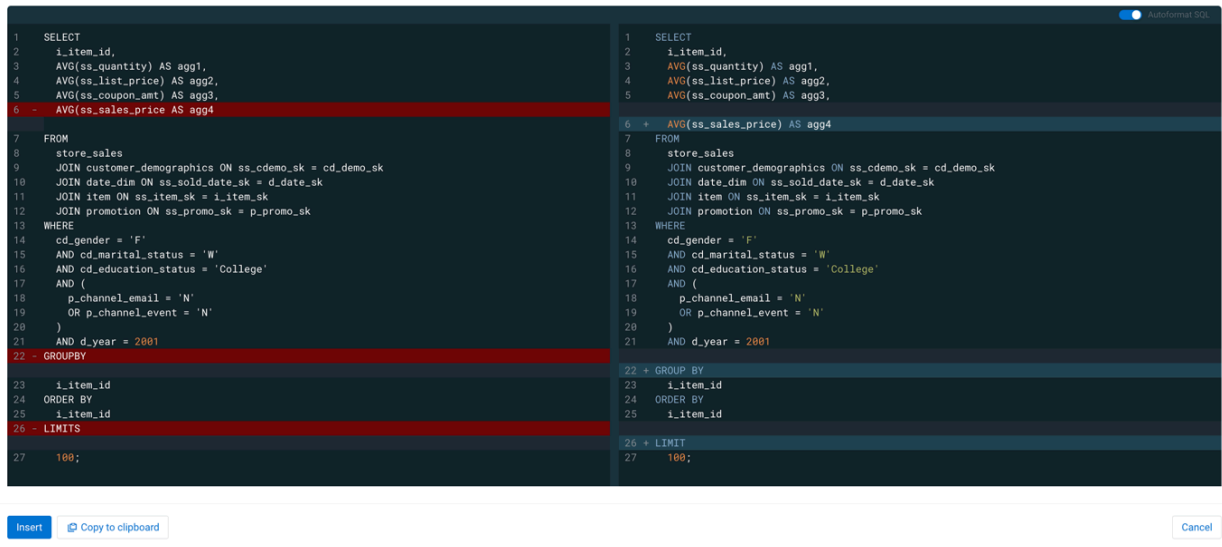
[Insert](#) [Copy to clipboard](#) [Cancel](#)

Fixing a SQL query

Click **FIX** to correct any syntax errors present in your SQL statement causing the query to fail as shown in the following image:



For example:



```
1 SELECT
2   i_item_id,
3   AVG(ss_quantity) AS agg1,
4   AVG(ss_list_price) AS agg2,
5   AVG(ss_coupon_amt) AS agg3,
6 -  AVG(ss_sales_price) AS agg4
7 FROM
8   store_sales
9   JOIN customer_demographics ON ss_demo_sk = cd_demo_sk
10  JOIN date_dim ON ss_sold_date_sk = d_date_sk
11  JOIN item ON ss_item_sk = i_item_sk
12  JOIN promotion ON ss_promo_sk = p_promo_sk
13 WHERE
14   cd_gender = 'F'
15   AND cd_marital_status = 'W'
16   AND cd_education_status = 'College'
17   AND (
18     p_channel_email = 'N'
19     OR p_channel_event = 'N'
20   )
21   AND d_year = 2001
22 - GROUPBY
23   i_item_id
24 ORDER BY
25   i_item_id
26 - LIMITS
27 100;
```

```
1 SELECT
2   i_item_id,
3   AVG(ss_quantity) AS agg1,
4   AVG(ss_list_price) AS agg2,
5   AVG(ss_coupon_amt) AS agg3,
6 +  AVG(ss_sales_price) AS agg4
7 FROM
8   store_sales
9   JOIN customer_demographics ON ss_demo_sk = cd_demo_sk
10  JOIN date_dim ON ss_sold_date_sk = d_date_sk
11  JOIN item ON ss_item_sk = i_item_sk
12  JOIN promotion ON ss_promo_sk = p_promo_sk
13 WHERE
14   cd_gender = 'F'
15   AND cd_marital_status = 'W'
16   AND cd_education_status = 'College'
17   AND (
18     p_channel_email = 'N'
19     OR p_channel_event = 'N'
20   )
21   AND d_year = 2001
22 + GROUP BY
23   i_item_id
24 ORDER BY
25   i_item_id
26 + LIMIT
27 100;
```

Insert Copy to clipboard Cancel

Limitations

- **Non-deterministic:** LLMs are non-deterministic, which means you cannot guarantee the same output for the same input every time, and it can lead to different responses to **similar** queries.
- **Ambiguity:** LLMs may struggle to handle ambiguous queries or contexts. SQL queries often rely on specific and unambiguous language, but LLMs can misinterpret or generate ambiguous SQL queries, leading to incorrect results.
- **Hallucination:** In the context of LLMs, hallucination refers to a phenomenon where these models generate text or responses that are incorrect, nonsensical, or fabricated. Occasionally you might see incorrect identifiers or literals in the response.