Bug Report: MindsDB `EVALUATE KNOWLEDGE_BASE` - Incorrect Behavior & Poor Error Handling

Reported By: Rajesh Adhikari / rajesh-adk-137 **Date:** June 29, 2025 **MindsDB Version: 8.0.17**

1. Overview

This report details significant issues with the `EVALUATE KNOWLEDGE_BASE` command in MindsDB. The command exhibits illogical behavior for both `doc_id` and `llm_relevancy` evaluation versions, where results do not correlate with the actual content or expected semantic matching. Furthermore, its error handling for LLM parameters (e.g., incorrect API keys) is deficient, leading to delayed, uninformative outputs rather than immediate, clear error messages.

2. Prerequisites & Setup

To reproduce this issue, please ensure you have set up the CharacterKB project following its main `README.md` and the `MINDSDB_SETUP.md` guides available in the repository: https://github.com/rajesh-adk-137/character_kb.git.

Specifically, ensure that:

- MindsDB is running via Docker Desktop.
- The `character_kb_10000` Knowledge Base is created and populated.

Prepare `testing_data.csv` for MindsDB Files

The `EVALUATE KNOWLEDGE_BASE` command relies on a `test_table`. For this report, we use `testing_data.csv`. This step is optional for the core application functionality but is crucial for reproducing this bug.

Steps to prepare `testing_data.csv` for MindsDB Files:

1. **Download `testing_data.csv`**: Locate and download the

`testing_data.csv` file from the `mindsdb/` folder in the repository.

2. **Upload to MindsDB Files**: In the MindsDB Studio GUI, navigate to the `Files` section (left sidebar) and upload `testing_data.csv`. Ensure you name the uploaded file exactly `testing_data`.

Once uploaded, you can inspect the data if needed:

SELECT * FROM files.testing_data LIMIT 5;

This is the testing_data.csv, the first 5 entries are valid.

A	B	C
doc_id	question	answer
1783	Iron Man, also ki	Iron Man, also k
3221	Joker is the prota	Joker is the prot
3237	Tsukushi Makinc	Tsukushi Makin
1	Valery Legasov i	Valery Legasov
15	Natsume is a cha	Natsume is a ch
5	-	
5	-	
5	-	
5	-	
5		
5		
5		

3. Understanding `EVALUATE KNOWLEDGE_BASE` (as per documentation)

Based on MindsDB documentation, the `EVALUATE KNOWLEDGE_BASE` command is intended to assess the relevancy and accuracy of documents returned by a Knowledge Base.

- **`test_table`**: A table containing test data, typically questions/content and expected document IDs or relevant answers.

- **`version`**: Defines the evaluation method:
  - `'doc_id'`: Checks if the document ID returned by the KB matches the expected `doc_id` in the `test_table`. Counts `total_found` (if found in top 100) and `retrieved_in_top_10`.
  - `'llm_relevancy'`: Uses an LLM to rank and evaluate responses for relevancy.
- **`llm`**: (Optional for `doc_id` version per docs, but seems required in practice) Defines the LLM for evaluation, especially for `'llm_relevancy'`.
- **Expected `doc_id` Output Fields**: `total`, `total_found`, `retrieved_in_top_10`, `cumulative_recall`, `avg_query_time`, `name`, `created_at`.
- **Expected `llm_relevancy` Output Fields**: `avg_relevancy`, `avg_relevance_score_by_k`, `avg_first_relevant_position`, `mean_mrr`, `hit_at_k`, `bin_precision_at_k`, `avg_entropy`, `avg_ndcg`, `avg_query_time`, `name`, `created_at`.

4. Steps to Reproduce & Actual Behavior

A. Reproducing with `version = 'doc_id'`

Execute the Evaluation: Execute the following SQL command in MindsDB Studio.
(Replace `YOUR_OPENAI_API_KEY` with your actual OpenAI API key)

```
EVALUATE KNOWLEDGE_BASE character_kb_10000

USING

  test_table = files.testing_data,

  version = 'doc_id',

  evaluate = true,

  llm = {

    'provider': 'openai',

    'api_key': 'YOUR_OPENAI_API_KEY', -- Required, despite documentation suggesting optional

    'model_name': 'gpt-4o'
```

};

1. **Observe Output:**

	total	total_found	retrieved_in_top_10	cumulative_recall
1	12	9	8	{"0":0.08333333333333333,"1":0.08333333333333333,"2":0.0

(It is clearly only counting those entries with doc_id less than 10 and 100)

2. **Actual Behavior (`doc_id` version):** The evaluation results appear to be entirely nonsensical and do not reflect actual semantic matching or Knowledge Base performance. Instead, the `total_found` and `retrieved_in_top_10` metrics seem to be calculated solely based on the `doc_id` value in the `testing_data.csv`, irrespective of the query/content field's semantic relevance or even its presence.
    - If a `doc_id` in `testing_data.csv` is `< 100`, it's counted in `total_found`.
    - If a `doc_id` in `testing_data.csv` is `< 10`, it's counted in `retrieved_in_top_10` (and consequently also in `total_found`).
    - The content/question field in `testing_data.csv` can be completely nonsensical or empty, and it still yields the same results based on the `doc_id` values. This indicates the semantic search component is entirely bypassed or ignored in this evaluation mode.
    - **Note:** The `llm` parameter is *required* for this version. Omitting it results in an error, contradicting MindsDB's documentation which suggests it's optional for `doc_id` evaluation.

B. Reproducing with `version = 'llm_relevancy'`

Execute the Evaluation: Execute the following SQL command in MindsDB Studio. *(Replace `YOUR_OPENAI_API_KEY` with your actual OpenAI API key)*

EVALUATE KNOWLEDGE_BASE character_kb_10000

USING

test_table = files.testing_data,

version = 'llm_relevancy',

evaluate = true,

llm = {

'provider': 'openai',

'api_key': 'YOUR_OPENAI_API_KEY', -- Replace with your OpenAI API Key

'model_name': 'gpt-4o'

};

1. **Observe Output:**

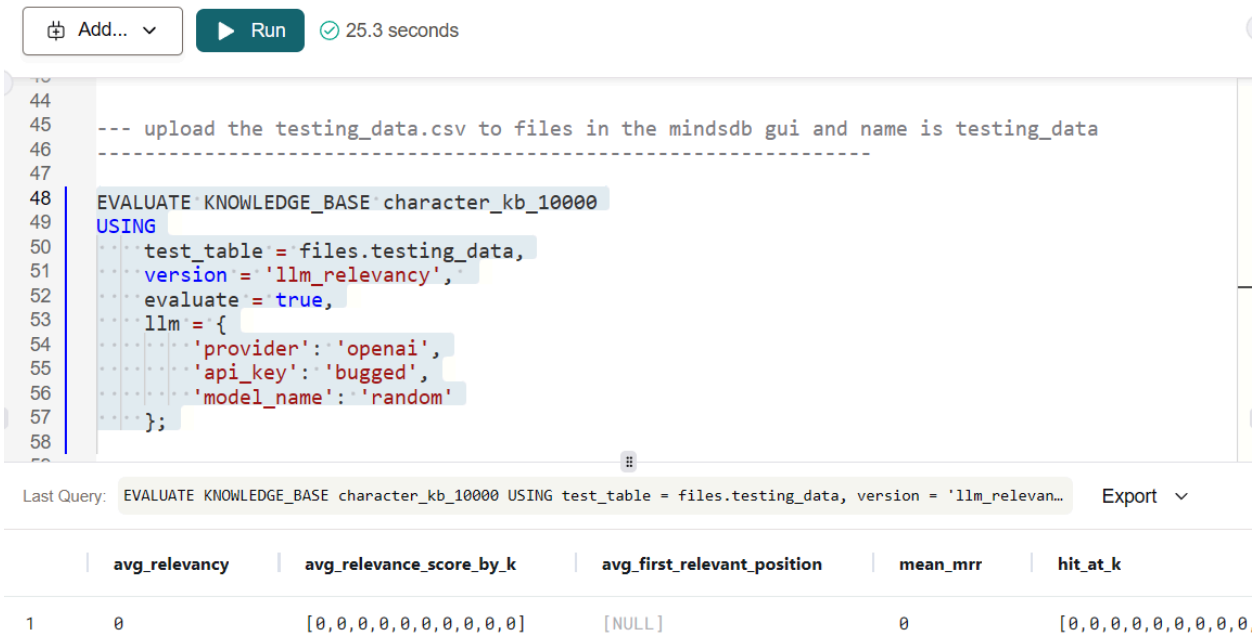
	avg_relevancy	avg_relevance_score_by_k	avg_first_relevant_position	mean_mrr	hit_at_k
1	0	[0,0,0,0,0,0,0,0,0,0]	[NULL]	0	[0,0,0,0,0,0,0,0,0,

(A clear wrong answer as there are valid entries)

2. **Actual Behavior (`llm_relevancy` version):** The output fields for `llm_relevancy` evaluation are nonsensical, typically returning zeros or `NULL` values for all metrics, such as: `0`, `[0,0,0,0,0,0,0,0,0,0]`, `[NULL]`, `0`, `[0,0,0,0,0,0,0,0,0,0]`, `[0,0,0,0,0,0,0,0,0,0]`, `0`. This indicates that the LLM-based evaluation is also not functioning correctly.

C. Reproducing LLM Parameter Error Handling Issue

1. **Execute with Incorrect LLM Parameters:** Execute the `llm_relevancy` evaluation query (as in Section 4.B) but intentionally provide a **wrong OpenAI API key** or a **non-existent `model_name`** (e.g., `'model_name': 'non-existent-model'`).
2. **Observe Behavior:**



(Clearly, the api keys and model_name are invalid, yet it still executed for 25 seconds and returned nonsensical result as before)

3. **Actual Behavior (LLM Error Handling):** Instead of an immediate error (e.g., "Invalid API Key" or "Model not found"), the command still runs for approximately 30 seconds before returning the same nonsensical output (all zeros/NULLs) as described in Section 4.B. This lack of immediate and clear error feedback makes debugging extremely difficult.

5. Expected Behavior (General)

- The `EVALUATE KNOWLEDGE_BASE` command should accurately and meaningfully assess the Knowledge Base's performance based on the provided `test_table` and `version`.
- For `version = 'doc_id'`: The evaluation should involve performing semantic queries against the KB using the content from `test_table` and then comparing the *returned* document IDs with the `doc_id` in the `test_table`.
- For `version = 'llm_relevancy'`: The evaluation should utilize the specified LLM to genuinely assess the relevance of the Knowledge Base's responses to the queries in `test_table`.
- **Robust Error Handling:** Invalid LLM parameters (e.g., incorrect API keys, non-existent models) should result in immediate, clear, and descriptive error

messages, rather than delayed, nonsensical outputs.
- **Documentation Compliance:** The `llm` parameter should truly be optional for `doc_id` evaluation if the documentation states so, or the documentation should be updated to reflect its mandatory nature.

6. Diagnostic Steps Performed & Observations

- Tested both `doc_id` and `llm_relevancy` evaluation versions.
- Confirmed `doc_id` evaluation results were independent of the semantic content in `testing_data.csv`.
- Confirmed `llm_relevancy` consistently returned non-informative zero/NULL values.
- Verified that providing incorrect OpenAI API keys or non-existent model names for `llm` parameters in `llm_relevancy` evaluation does not yield proper error messages but instead leads to a delayed, uninformative output.
- Attempted to use a Google Sheet as the `test_table` source instead of `files.testing_data` and observed the exact same problematic behavior, indicating the issue is not source-specific.
- The MindsDB documentation for `EVALUATE KNOWLEDGE_BASE` is relatively sparse and does not offer sufficient detail or examples to explore alternative approaches or troubleshoot these behaviors effectively.

7. Conclusion & Severity

The `EVALUATE KNOWLEDGE_BASE` feature in MindsDB appears to be fundamentally broken, exhibiting nonsensical evaluation logic and a critical lack of robust error handling. This renders the feature currently unusable for its intended purpose of assessing Knowledge Base performance. The behavior suggests a potentially "half-baked" or dummy implementation in the backend. This is a **High Severity** bug as it undermines a core evaluation capability.