User Documentation

Edit New page Jump to bottom

linda.najar edited this page 4 days ago · 8 revisions

User Documentation for RTDIP-SDK Pipeline Configuration Assistant

Overview

Welcome to the RTDIP-SDK Pipeline Configuration Assistant. This tool leverages a Language Model (GPT 3.5-turbo) connected to a content hub that contains a variety of RTDIP-SDK pipeline configurations. Through the use of Retrieval Augmented Generation (RAG), the assistant can understand your input queries and provide relevant pipeline configuration suggestions.

Getting Started

Prerequisites

Before you begin, ensure that you have:

- Python version 3.11 or higher
- An OpenAI Access Token
- Docker Desktop

Accessing the Assistant

To begin using the RTDIP-SDK Pipeline Configuration Assistant, you'll first need to set it up on your system. Detailed instructions for building and deploying the assistant can be found in our Building and Deploying Documentation. This guide will walk you through the necessary steps to get the assistant up and running.

Once you have successfully deployed the assistant, you can access it through the provided URL.

Using the Assistant

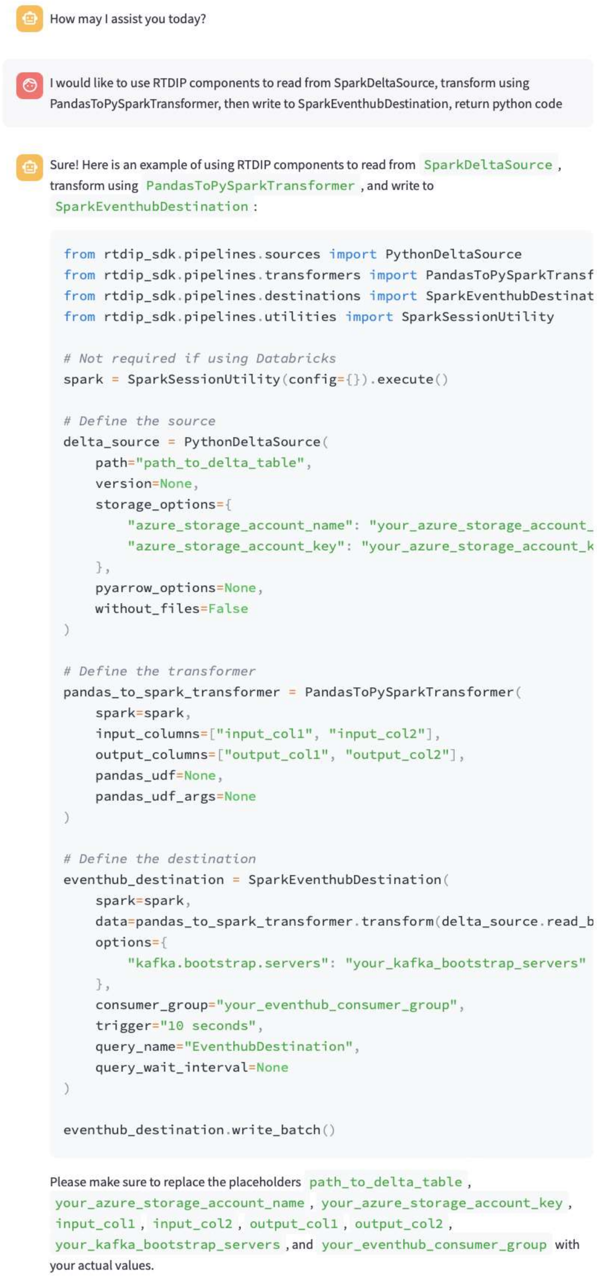
Submitting Queries

1. **Input Query**: In the provided input field, type your query related to RTDIP-SDK pipeline configurations. For example, "I would like to use RTDIP components to read from SparkDeltaSource, transform using PandasToPySparkTransformer, then write to SparkEventhubDestination , return Python code ."
2. **Submit Query**: Press the 'Enter' key to send your query to the assistant.

Receiving Answers

After submitting your query, the assistant will process it and display a suggested pipeline configuration or related information.

Example :



How may I assist you today?

I would like to use RTDIP components to read from SparkDeltaSource, transform using PandasToPySparkTransformer, then write to SparkEventhubDestination, return python code

Sure! Here is an example of using RTDIP components to read from `SparkDeltaSource`, transform using `PandasToPySparkTransformer`, and write to `SparkEventhubDestination`:

```python
from rtdip_sdk.pipelines.sources import PythonDeltaSource
from rtdip_sdk.pipelines.transformers import PandasToPySparkTransf
from rtdip_sdk.pipelines.destinations import SparkEventhubDestinat
from rtdip_sdk.pipelines.utilities import SparkSessionUtility

# Not required if using Databricks
spark = SparkSessionUtility(config={}).execute()

# Define the source
delta_source = PythonDeltaSource(
    path="path_to_delta_table",
    version=None,
    storage_options={
        "azure_storage_account_name": "your_azure_storage_account_
        "azure_storage_account_key": "your_azure_storage_account_k
    },
    pyarrow_options=None,
    without_files=False
)

# Define the transformer
pandas_to_spark_transformer = PandasToPySparkTransformer(
    spark=spark,
    input_columns=["input_col1", "input_col2"],
    output_columns=["output_col1", "output_col2"],
    pandas_udf=None,
    pandas_udf_args=None
)

# Define the destination
eventhub_destination = SparkEventhubDestination(
    spark=spark,
    data=pandas_to_spark_transformer.transform(delta_source.read_b
    options={
        "kafka.bootstrap.servers": "your_kafka_bootstrap_servers"
    },
    consumer_group="your_eventhub_consumer_group",
    trigger="10 seconds",
    query_name="EventhubDestination",
    query_wait_interval=None
)

eventhub_destination.write_batch()
```

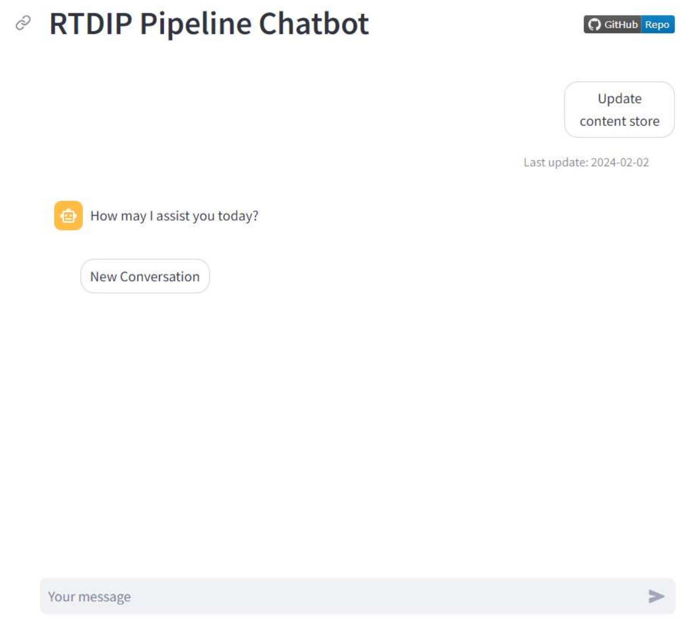
Please make sure to replace the placeholders `path_to_delta_table`, `your_azure_storage_account_name`, `your_azure_storage_account_key`, `input_col1`, `input_col2`, `output_col1`, `output_col2`, `your_kafka_bootstrap_servers`, and `your_eventhub_consumer_group` with your actual values.

Interacting with Content Store

- The assistant is connected to a content store that contains the examples of various RTDIP-SDK pipeline configurations.
- You can ask for specific examples or templates, and the assistant will retrieve relevant content from the hub.

Updating Content Store

The "Update Content Store" button is designed to synchronize your local content store with the latest updates from the 'core' of the RTDIP repository. When new sources, transformers, or destinations are committed to the repository, clicking this button will trigger the script to selectively check out these updates and apply them to your local content store. The script initializes a local repository, configures it for sparse checkout, pulls the specific content from the remote repository, and then copies it over to the designated local directory, ensuring that your local store is up-to-date with the remote repository's latest developments.



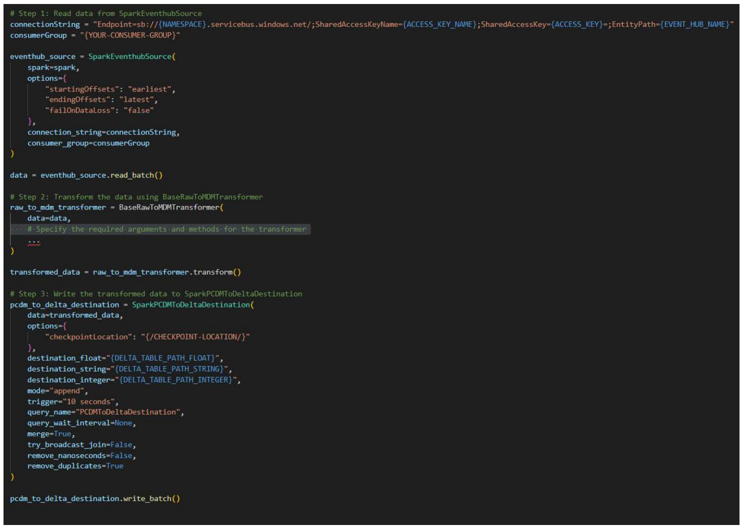
RTDIP Pipeline Chatbot

Last update: 2024-02-02

How may I assist you today?

Best Practices for Querying

- Be specific in your queries to receive the most accurate suggestions.
- Use relevant keywords related to the RTDIP-SDK pipeline configurations.
- If you specify a source, transformer, or destination that is not present in the content store, the output will still be correct. However, it will include the message 'Specify the required arguments and methods for the transformer' to prompt the user for additional input.



```
# Step 1: Read data from SparkEventhubSource
connectionString = "Endpoint=sb://{NAMESPACE}.servicebus.windows.net/;SharedAccessKeyName={ACCESS_KEY_NAME};SharedAccessKey={ACCESS_KEY}=;EntityPath={EVENT_HUB_NAME}"
consumerGroup = "{YOUR-CONSUMER-GROUP}"

eventhub_source = SparkEventhubSource(
    spark=spark,
    options={
        "startingOffsets": "earliest",
        "endingOffsets": "latest",
        "failOnDataLoss": "false"
    },
    connection_string=connectionString,
    consumer_group=consumerGroup
)

data = eventhub_source.read_batch()

# Step 2: Transform the data using BaseRawToMDMTransformer
raw_to_mdm_transformer = BaseRawToMDMTransformer(
    data=data,
    # Specify the required arguments and methods for the transformer
    ...
)

transformed_data = raw_to_mdm_transformer.transform()

# Step 3: Write the transformed data to SparkPCDMToDeltaDestination
pcdm_to_delta_destination = SparkPCDMToDeltaDestination(
    data=transformed_data,
    options={
        "checkpointLocation": "{/CHECKPOINT-LOCATION/}"
    },
    destination_float="{DELTA_TABLE_PATH_FLOAT}",
    destination_string="{DELTA_TABLE_PATH_STRING}",
    destination_integer="{DELTA_TABLE_PATH_INTEGER}",
    mode="append",
    trigger="10 seconds",
    query_name="PCDMToDeltaDestination",
    query_wait_interval=None,
    merge=True,
    try_broadcast_join=False,
    remove_nanoseconds=False,
    remove_duplicates=True
)

pcdm_to_delta_destination.write_batch()
```

Troubleshooting

- **No Response**: If you do not receive any response, verify your internet connection and try resubmitting your query.
- **Unclear Responses**: If the assistant's response is not clear, try rephrasing your query with more specific explanations.

FAQ

Q: Is there a limit to the number of queries I can submit?
A: No, you can submit as many queries as you like.

Q: What should I do if the assistant's suggestions don't fully meet my requirements? A: While the assistant aims to provide the most relevant suggestions based on your query, it might not always cover every specific requirement. In such cases, you can refine your query for more precision .

+ Add a custom footer

Pages 4

Find a page…

- Home
- Build and Deploy Steps
- Software Architecture Design
- User Documentation
  - User Documentation for RTDIP-SDK Pipeline Configuration Assistant
    - Overview
    - Getting Started
      - Prerequisites
    - Accessing the Assistant
    - Using the Assistant
      - Submitting Queries
      - Receiving Answers
      - Interacting with Content Store
      - Updating Content Store
    - Best Practices for Querying
    - Troubleshooting
    - FAQ

+ Add a custom sidebar

Clone this wiki locally

`https://github.com/amosproj/amos2023ws05-pipeline-config-chat-ai.wiki.git`