Connecting to MongoDB in Databricks Using PySpark

Overview

This document describes the steps to establish a connection to a MongoDB database from Databricks using PySpark. It covers the setup of the Spark session, connection details, and examples of data transformations.

Prerequisites

- An active MongoDB Atlas account (or a local MongoDB instance).
- A Databricks account to access the Databricks workspace.

Steps to Set Up Databricks

1. Create a Databricks Account

1. Go to the [Databricks website](https://databricks.com/).
2. Click on **Get Started for Free**.
3. Fill in the required information to create your account.
4. Confirm your email address and log in to your Databricks account.

2. Create a Workspace

1. After logging in, navigate to the **Workspace** tab.
2. Click on the **Create Workspace** button if prompted, or select your existing workspace.
3. Follow the instructions to set up your workspace.

3. Create a Cluster

1. Go to the **Clusters** section in the left sidebar of your workspace.
2. Click on **Create Cluster**.
3. Fill in the cluster name, select the appropriate Databricks runtime version, and configure the settings as needed.

4. Click on **Create Cluster** to provision your cluster. Wait for it to start.

4. Install Necessary Libraries

Once your cluster is running, install the necessary libraries:

1. Navigate to the **Libraries** tab of your cluster.
2. Click on **Install New**.
3. Select **PyPI** for pymongo and pyspark:
   - For pymongo, enter pymongo and click **Install**.
   - For pyspark, enter pyspark and click **Install**.
4. For the MongoDB Spark Connector, use the **Maven** option:
   - Enter org.mongodb.spark:mongo-spark-connector_2.12:3.0.1 and click **Install**.

Steps to Connect to MongoDB

1. Configure MongoDB Network Access

Before connecting to MongoDB from Databricks, ensure that your MongoDB Atlas cluster allows connections from your IP address:

1. Log in to your MongoDB Atlas account.
2. Navigate to the **Network Access** section in the left sidebar.
3. Click on **Add IP Address** or edit the existing access list entry.
4. Select **Allow Access from Anywhere** (0.0.0.0/0) or specify your current IP address to grant access.
5. Save the changes.

2. Set Up Spark Session

Begin by importing necessary libraries and setting up the Spark session. The following code initializes the Spark session with the MongoDB Spark Connector.

```python
from pyspark.sql import SparkSession
from pyspark.sql.functions import length

# MongoDB connection details
host = "******"  # Masked host
username = "******"  # Masked username
password = "******"  # Masked password
database = "news_database"  # Your database name
collection = "news_collection"  # Your collection name

# Create a Spark session
spark = SparkSession.builder \
    .appName("MongoDB Integration") \
    .config("spark.mongodb.input.uri", f"mongodb+srv://{username}:{password}@{host}/{database}.{collection}") \
    .config("spark.mongodb.output.uri", f"mongodb+srv://{username}:{password}@{host}/{database}.{collection}") \
    .getOrCreate()
```

3. Load Data from MongoDB

To load data from a MongoDB collection into a Spark DataFrame, use the following code:

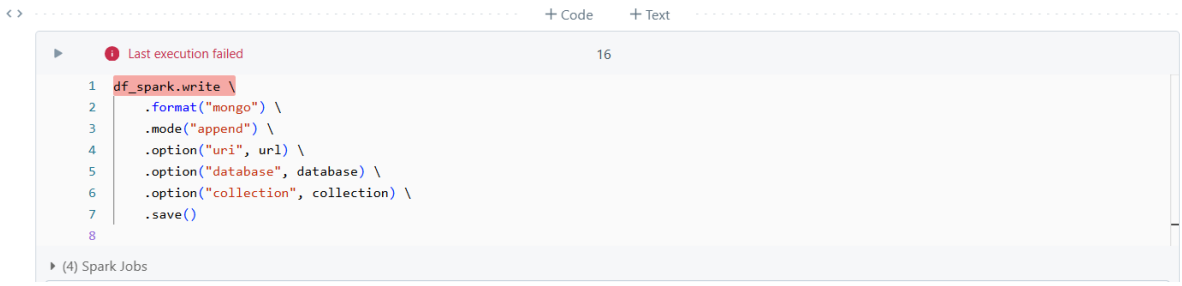
```python
# Load data from MongoDB
df_spark = spark.read.format("com.mongodb.spark.sql.DefaultSource") \
    .option("uri", f"mongodb+srv://{username}:{password}@{host}/{database}.{collection}") \
    .load()

# Show the DataFrame
df_spark.show()
```

4. Data Transformations

Once the data is loaded into the DataFrame, you can perform various transformations.

5. Write Data Back to MongoDB

To write the transformed DataFrame back to MongoDB, use the following code:



```python
df_spark.write \
    .format("mongo") \
    .mode("append") \
    .option("uri", url) \
    .option("database", database) \
    .option("collection", collection) \
    .save()
```

6. Conclusion

This document provides a step-by-step guide to connecting to MongoDB in Databricks using PySpark. You can perform various data transformations and write data back to MongoDB as needed. Be sure to handle your connection credentials securely.