Master PySpark: From Zero to Big Data Hero!!

Loading Data from CSV File into a DataFrame

Loading data into DataFrames is a fundamental step in any data processing workflow in PySpark. This document outlines how to load data from CSV files into a DataFrame, including using a custom schema and the implications of using the inferSchema option.

Step-by-Step Guide

1. Import Required Libraries

Before loading the data, ensure you import the necessary modules:

```
from pyspark.sql import SparkSession
from pyspark.sql.types import StructType, StructField, IntegerType,
StringType, DoubleType
```

2. Define the Schema

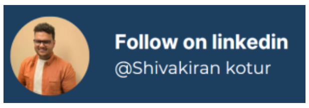
You can define a custom schema for your CSV file. This allows you to explicitly set the data types for each column.

```
# Define the schema for the CSV file
custom_schema = StructType([
    StructField("id", IntegerType(), True),
    StructField("name", StringType(), True),
    StructField("age", IntegerType(), True),
    StructField("salary", DoubleType(), True)
])
```

3. Read the CSV File

Load the CSV file into a DataFrame using the read.csv() method. Here, header=True treats the first row as headers, and inferSchema=True allows Spark to automatically assign data types to columns.

```
# Read the CSV file with the custom schema
df = spark.read.csv("your_file.csv", schema=custom_schema,
header=True)
```



Follow me on LinkedIn – Shivakiran kotur

4. Load Multiple CSV Files

To read multiple CSV files into a single DataFrame, you can pass a list of file paths. Ensure that the schema is consistent across all files.

```
# List of file paths
file_paths = ["file1.csv", "file2.csv", "file3.csv"]
# Read multiple CSV files into a single DataFrame
df = spark.read.csv(file_paths, header=True, inferSchema=True)
```

5. Load a CSV from FileStore

Here is an example of loading a CSV file from Databricks FileStore:

```
df = spark.read.csv("/FileStore/tables/Order.csv", header=True,
inferSchema=True, sep=',')
```

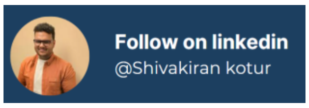
6. Display the DataFrame

Use the following commands to check the schema and display the DataFrame:

```
# Print the schema of the DataFrame
df.printSchema()

# Show the first 20 rows of the DataFrame
df.show()  # Displays only the first 20 rows

# Display the DataFrame in a tabular format
display(df)  # For Databricks notebooks
```



Follow me on LinkedIn – Shivakiran kotur

Interview Question: How Does inferSchema Work?

- **Behind the Scenes:** When you use inferSchema, Spark runs a job that scans the CSV file from top to bottom to identify the best-suited data type for each column based on the values it encounters.

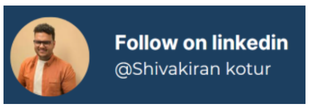
Does It Make Sense to Use inferSchema?

- **Pros**:
    - Useful when the schema of the file keeps changing, as it allows Spark to automatically detect the data types.
- **Cons**:
    - Performance Impact: Spark must scan the entire file, which can take extra time, especially for large files.
    - Loss of Control: You lose the ability to explicitly define the schema, which may lead to incorrect data types if the data is inconsistent.

Conclusion

Loading data from CSV files into a DataFrame is straightforward in PySpark. Understanding how to define a schema and the implications of using inferSchema is crucial for optimizing your data processing workflows.

This document provides a comprehensive overview of how to load CSV data into DataFrames in PySpark, along with considerations for using schema inference. Let me know if you need any more details or adjustments!



Follow me on LinkedIn – Shivakiran kotur