Reading: User-Defined Schema (UDS) for DSL and SQL

Estimated time needed: 10 minutes

How to Define and Enforce a User-Defined Schema in PySpark?

In this reading, you will learn how to define and enforce a user-defined schema in PySpark.

Spark provides a structured data processing framework that can define and enforce schemas for various data sources, including CSV files. Let's look at the steps to define and use a user-defined schema for a CSV file in PySpark:

Step 1:

Import the required libraries.

```
1. 1
```

```
1. from pyspark.sql.types import StructType, IntegerType, FloatType, StringType, StructField
```

Copied!

Step 2:

Define the schema.

Understanding the data before defining a schema is an important step.

Let's take a look at the step-by-step approach to understanding the data and defining an appropriate schema for a given input file:

1. **Explore the data:** Understand the different data types present in each column.
2. **Column data types:** Determine the appropriate data types for each column based on your observed values.
3. **Define the schema:** Use the 'StructType' class in Spark and create a 'StructField' for each column, mentioning the column name, data type, and other properties.

Example:

```
1. 1
2. 2
3. 3
4. 4
5. 5
6. 6
7. 7
```

```
schema = StructType([
    StructField("Emp_Id", StringType(), False),
    StructField("Emp_Name", StringType(), False),
    StructField("Department", StringType(), False),
    StructField("Salary", IntegerType(), False),
    StructField("Phone", IntegerType(), True),
])
```

Copied!

'False' indicates null values are **NOT** allowed for the column.

The schema defined above can be utilized for the below CSV file data:

Filename: employee.csv

```
1
2
3
4
```

```
emp_id,emp_name,dept,salary,phone
A101,jhon,computer science,1000,+1 (701) 846 958
A102,Peter,Electronics,2000,
A103,Micheal,IT,2500,
```

Copied!

Step 3: Read the input file with user-defined schema.

```
1
2
3
4
5
6
7
8
9
```

```
#create a dataframe on top a csv file
df = (spark.read
  .format("csv")
  .schema(schema)
  .option("header", "true")
  .load("employee.csv")
)
# display the dataframe content
df.show()
```

Copied!

Step 4: Use the `printSchema()` method in Spark to display the schema of a DataFrame and ensure that the schema is applied correctly to the data.

```
1
```

```
df.printSchema()
```

Copied!

Through the preceding four steps, you've acquired the ability to establish a schema for a CSV file. Additionally, you've employed this user-defined schema (UDF) to read the CSV file, exhibit its contents, and showcase the schema itself.

Author(s)

- Raghul Ramesh

Changelog

Date	Version	Changed by	Change Description
2023-09-07	0.2	Rahul Rawat	QA pass with edits
2023-09-06	0.1	Gagandeep Singh	Initial version created