Spark : python + sql

* Java + sql = spark-java
* python + sql = pyspark
* Scala + sql = spark- Scala
* R +sql = spark-R
* What is spark?

Spark is a open source frame work. It is distributed computing system designed to process large volumns of data in parallel across a cluster of computers.

* It support various programming languages:
* Java
* Scala
* Python
* R

It offers wide range of libraries and tools for data processing ,machine learning , graph processing and more.

* Key features in spark:
* In memory processing
* Distributed computing.
* Resilient distributed datasets(RDD):

It process the data in distributed manner.

It follows type strict.

It is low level API.

* Data farm API:

It is organized into named columns.

It allows sql queries.

It has optimization techniques.

It is high level API.

* Machine learning.
* Streaming data processing.
* Graph data processing

Overall spark is powerful and flexible frame work that provides a wide range of features for processing and analyzing large volumes of data.

* Spark latest version 3.4.O
* Hadoop (2.X) supports spark by yarn source manager.

Map reduce – data processing

HDFS – data storage

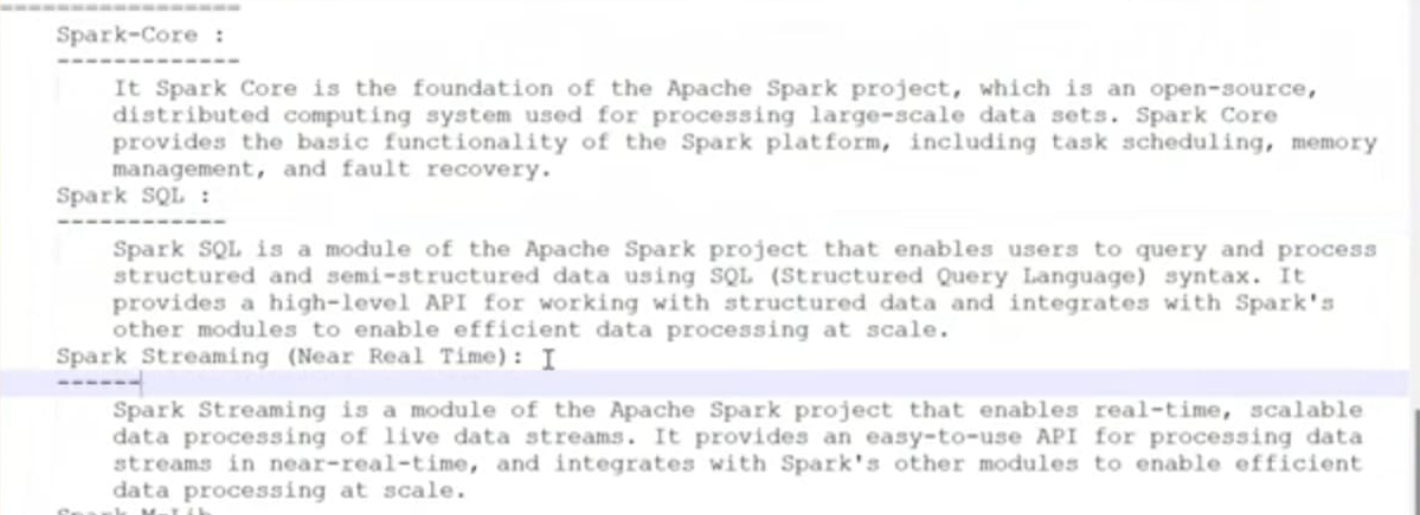
YARN – Cluster resource manager.

MAP REDUSE OTHERS

YARN

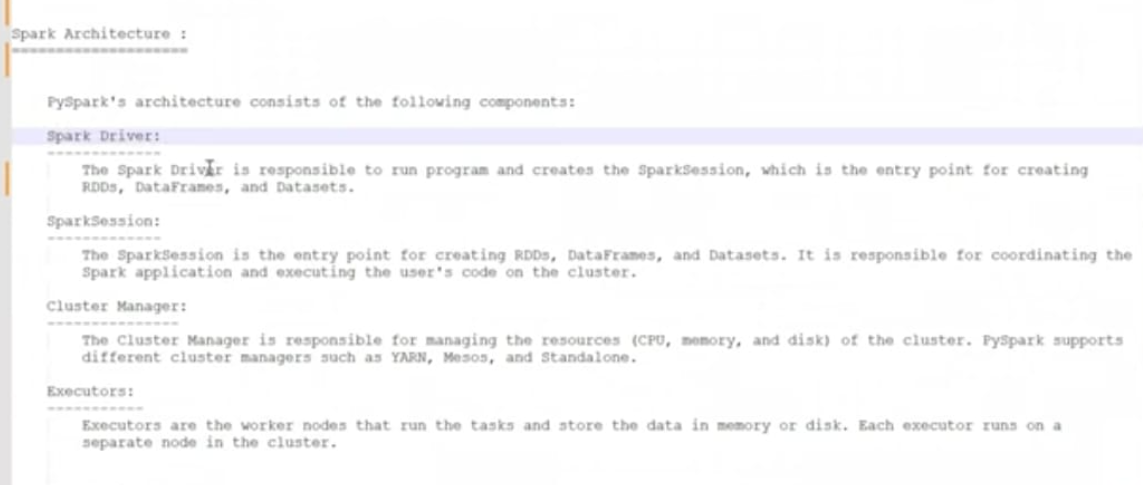
HDFS

* Sark components:
* Spark – core
* Spark – sql
* Spark – streaming
* Spark – M-lib
* Spark – graph



* Spark supports source system
* Hadoop environment
* Local environment
* Cloud environment
* RDBMS
* Different files
* Data ware house:
  + DynamoDB
  + Hive
  + Redshift
  + Big query





**Import Py Spark**:

python

from pyspark.sql import SparkSession

# Create a Spark session

spark = SparkSession.builder.appName("myApp").getOrCreate()

# Example: Creating a DataFrame

data = [("Alice", 1), ("Bob", 2), ("Cathy", 3)]

df = spark.createDataFrame(data, ["Name", "Value"])

# Show the DataFrame

df.show()

**Example: Using PySpark**

Here’s a more detailed example that includes creating a DataFrame and performing some operations:

python

from pyspark.sql import SparkSession

# Initialize Spark session

spark = SparkSession.builder.appName("exampleApp").getOrCreate()

# Create a DataFrame

data = [("John", "Doe", 30), ("Jane", "Doe", 25), ("Sam", "Smith", 35)]

columns = ["First Name", "Last Name", "Age"]

df = spark.createDataFrame(data, columns)

# Show the DataFrame

df.show()

# Perform some operations

df.filter(df["Age"] > 30).show()

df.select("First Name", "Age").show()

# Stop the Spark session

spark.stop()

HOW TO INSTALL PYSPARK

Installing PySpark is a straightforward process. Here’s a step-by-step guide to get you up and running:

### Step 1: Install Java

Apache Spark requires Java to run. Make sure you have Java installed on your machine. You can download it from the official Oracle website. Once installed, set up the JAVA\_HOME environment variable to point to your Java installation.

### Step 2: Install PySpark

You can install PySpark using pip, the Python package installer. Open your command prompt or terminal and run the following command:

sh

pip install pyspark

### Step 3: Verify the Installation

After installing PySpark, you can verify the installation by running the following Python script. Create a new Python file (e.g., test\_pyspark.py) and add the following code:

python

from pyspark.sql import SparkSession

# Create a Spark session

spark = SparkSession.builder.appName("testApp").getOrCreate()

# Check the Spark version

print("Spark version:", spark.version)

# Stop the Spark session

spark.stop()

Run the script using your preferred Python interpreter:

sh

python test\_pyspark.py

If the script runs successfully and prints the Spark version, your PySpark installation is verified.

### Optional: Set Up Environment Variables

If you encounter issues with the PySpark installation, you may need to set up environment variables for Spark. This involves adding the Spark bin directory to your system's PATH variable and setting the SPARK\_HOME variable. Here’s how to do it:

#### On Windows:

1. Right-click on This PC or My Computer and select Properties.
2. Click on Advanced system settings.
3. Click on Environment Variables.
4. Under System variables, click New and add SPARK\_HOME with the path to your Spark installation.
5. Find the Path variable, click Edit, and add %SPARK\_HOME%\bin.

#### On macOS/Linux:

Add the following lines to your .bashrc or .zshrc file:

sh

export SPARK\_HOME=/path/to/spark

export PATH=$SPARK\_HOME/bin:$PATH

### Starting a PySpark Shell

You can start an interactive PySpark shell by running:

sh

pyspark

Multiple ways to create RDD

1.list to RDD:

* EG:
* my\_ list =(“hello world” ,”hello world”)

Print(type(my\_list))

* List convert into RDD:

My\_RDD= spark. sparkContext . parallelize (my\_list)

Print(My\_RDD. Collect())

Print(type(My\_RDD))

* What is parallelize?

List convert to RDD is parallelize

2.Text to RDD:

My\_RDD= spark. sparkContext . text file (“file path//filename.txt”)

Print(My\_RDD. Collect())

Print(type(My\_RDD))

3.RDD to RDD:

My\_RDD= spark. sparkContext . parallelize ([1,2,3,4,5,6,7])

New \_ RDD =rdd.filter(lambda x: x>=3)

Print(MY\_RDD .collect())

Print(New\_ RDD. collect())

4.Datafrme to RDD:

DF= spark.read.csv(“path:// file name”, header=True)

My\_RDD= DF.RDD

Print(type(My\_RDD))

5.Empty RDD:

My\_RDD= spark. SparkContext.emptyRDD()

Print(MY\_RDD .collect())

Print(type(My\_RDD))