06: Learn how to access Hive from Spark via SparkSQL & Dataframes by example

Posted on November 8, 2017

These Hadoop tutorials assume that you have installed Cloudera QuickStart, which has the Hadoop eco system like HDFS, Spark, Hive, HBase, YARN, etc.

This example extends Learn Hive to write to and read from AVRO & Parquet files by examples to access Hive metastore via Spark SQL. You run the Spark job in **local mode**.

Spark invokes SQL on Hive table

The code below uses Spark SQL and SPark Hive libraries to run SQL against HIve tables. As we saw earlier, Hive metadata are stored in a relational database. You can access the Hive metstaore via the "thrift" protocol.

```
1
2  import java.io.IOException;
3
4  import org.apache.spark.SparkConf;
5  import org.apache.spark.api.java.JavaRDD;
6  import org.apache.spark.api.java.JavaSparkContext;
7  import org.apache.spark.sql.DataFrame;
8  import org.apache.spark.sql.Row;
9  import org.apache.spark.sql.SQLContext;
10 import org.apache.spark.sql.hive.HiveContext;
11
12 public class SparkSimple {
13
14     public static void main(String[] args) throws IOException {
15
16         System.setProperty("hive.metastore.uris", "thrift://ip-addre
17
18         SparkConf sparkConf = new SparkConf().setAppName("Spark simp
19
20         JavaSparkContext javaSparkContext = new JavaSparkContext(spa
21         SQLContext sqlContext = new HiveContext(javaSparkContext.sc(
22
```

300+ Java Interview FAQs

300+ Java FAQs 🔥
16+ Java Key Areas Q&As
150+ Java Architect FAQs
80+ Java Code Quality Q&As
150+ Java Coding Q&As

300+ Big Data Interview FAQs

300+ Big Data FAQs 🔥

FAQs Data - 01: SQL
FAQs Data - 02: Data Modelling
FAQs Data - 02: Data Warehouse
FAQs Data - 03: Big Data
FAQs Data - 04: Hadoop (HDFS)
FAQs Data - 05: MapReduce
FAQs Data - 06: Hive
FAQs Data - 07: Impala
FAQs Data - 08: Spark
FAQs Data - 09: Spark SQL
FAQs Data - 10: Apache Kafka
FAQs Data - 11: Data Governance
FAQs Data - 11: NoSQL
FAQs Data - 12: Data security
FAQs Data - 13: Analytics & Science
FAQs Data - 14: AWS
FAQs Data - 15: Sqoop & Nifi
FAQs Data - 16: Yarn, Zookeeper
FAQs Data - 40: Scala 140+ FAQs

```
        DataFrame df = sqlContext.sql("Select * from  z_test.parquet

        df.show();

        JavaRDD<Row> javaRDD = df.javaRDD();

        javaRDD.foreach(x -> System.out.println(x.getAs("name").toSt

        javaSparkContext.close();
    }

}
```

Output:

```
+------+
|  name|
+------+
|  John|
|Samuel|
| Peter|
+------+
```

```
John
17/11/07 15:53:13 INFO executor.Executor: Finished task 0.0 in stage
17/11/07 15:53:13 INFO compress.CodecPool: Got brand-new decompresso
Peter
17/11/07 15:53:13 INFO executor.Executor: Finished task 2.0 in stage
17/11/07 15:53:13 INFO scheduler.TaskSetManager: Finished task 0.0 i
17/11/07 15:53:13 INFO scheduler.TaskSetManager: Finished task 2.0 i
Samuel
//......
```

How to get the thrift hostname & port?

1) In the edge node under "**/etc/hive/conf**" you will find "hive-site.xml", which will have

```
 <property>
    <name>hive.metastore.uris</name>
    <value>thrift://ip-address:9083</value>
  </property>
```

2) If you have installed Cloudera, go to the Cloudera Manger, and click on Hive. You will see a drop down named "**Actions**". Click on "Download Client Configuration". This is a zip file with client configuration files such as **hive-site.xml**, core-site.xml, hdfs-site.xml, hive-env.sh, etc.

pom.xml for the Spark libraries

```
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http:/
    xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://mav
    <modelVersion>4.0.0</modelVersion>
    <groupId>com.mytutorial</groupId>
    <artifactId>simple-spark</artifactId>
    <packaging>jar</packaging>
```

FAQs Data - 41: 100+ Python FAQs
Tutorials - Big Data
UT - Starting Big Data
TUT - Starting Spark & Scala
TUT - Starting with Python
TUT - Kafka
TUT - Pig
TUT - Apache Storm
TUT - Spark Scala on Zeppelin
TUT - Cloudera
TUT - Cloudera on Docker
TUT - File Formats
TUT - Spark on Docker
TUT - Flume
TUT - Hadoop (HDFS)
TUT - HBase (NoSQL)
TUT - Hive (SQL)
TUT - Hadoop & Spark
TUT - MapReduce
TUT - Spark and Scala
TUT - Spark & Java
TUT - PySpark on Databricks
TUT - Zookeeper

800+ Java Interview Q&As
300+ Core Java Q&As
300+ Enterprise Java Q&As
150+ Java Frameworks Q&As
120+ Companion Tech Q&As
Tutorials - Enterprise Java

```
    <version>1.0-SNAPSHOT</version>
    <name>simple-spark</name>
    <url>http://maven.apache.org</url>

    <properties>
        <maven.compiler.source>1.8</maven.compiler.source>
        <maven.compiler.target>1.8</maven.compiler.target>
        <hadoop.version>2.7.2</hadoop.version>
        <avro.version>1.7.6-cdh5.8.3</avro.version>
        <spark.version>1.6.0-cdh5.8.3</spark.version>
        <scala.version>2.10.4</scala.version>
        <parquet.version>1.5.0-cdh5.8.3</parquet.version>
        <scala.binary.version>2.10</scala.binary.version>
        <hbase.version>1.2.0-cdh5.8.3</hbase.version>
    </properties>

    <repositories>
        <repository>
            <id>central</id>
            <name>Maven Central</name>
            <url>http://repo1.maven.org/maven2/</url>
        </repository>
        <repository>
            <id>cloudera</id>
            <url>http://repository.cloudera.com/artifactory/cloudera
        </repository>
    </repositories>

    <dependencies>
        <dependency>
            <groupId>org.scala-lang</groupId>
            <artifactId>scala-library</artifactId>
            <version>${scala.version}</version>
        </dependency>

        <!-- Spark libraries -->
        <dependency>
            <groupId>org.apache.spark</groupId>
            <artifactId>spark-core_${scala.binary.version}</artifact
            <version>${spark.version}</version>
            <scope>provided</scope>
        </dependency>
        <dependency>
            <groupId>org.apache.spark</groupId>
            <artifactId>spark-sql_${scala.binary.version}</artifactI
            <version>${spark.version}</version>
            <scope>provided</scope>
        </dependency>
        <dependency>
            <groupId>org.apache.spark</groupId>
            <artifactId>spark-hive_${scala.binary.version}</artifact
            <version>${spark.version}</version>
            <scope>provided</scope>
        </dependency>
    </dependencies>

</project>


```

With SparkSQL and Dataframes you can perform tables joins, filtering, etc.
You can also write dataframes to to different file paths as AVRO or Parquet.

‹ 05: Learn Hive to write to and read from AVRO & Parquet files by examples

07: Learn Spark Dataframes to do ETL in Java with examples ›

Arulkumaran

Mechanical Engineer to self-taught Java engineer within 2 years & a **freelancer** within 3 years. Freelancing since 2003. Preparation empowered me to **attend 190+ job interviews** & choose from **150+ job offers** with sought-after contract rates. Author of the book "**Java/J2EE job interview companion**", which sold **35K+ copies** & superseded by this site with 2,050+ registered users. Amazon.com profile | **Reviews** | LinkedIn | LinkedIn Group | YouTube

Contact us: java-interview@hotmail.com

Disclaimer