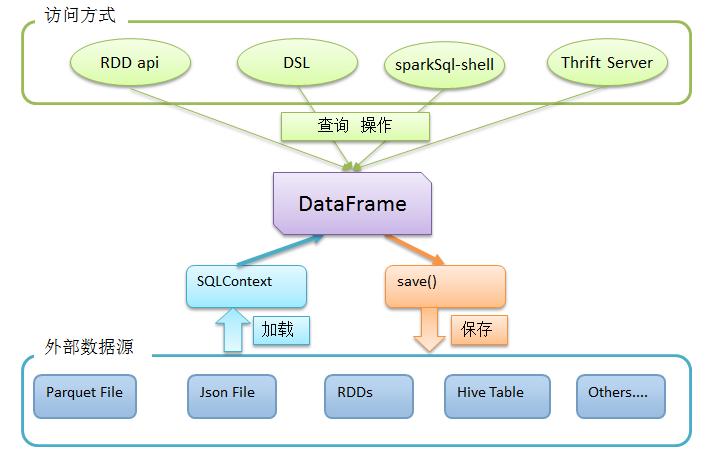
Spark SQL

Spark SQL是一个用来处理结构化数据的Spark组件，其提供了一个叫做DataFrames的可编程抽象数据模型，并且可视为一个分布式的SQL查询引擎。Spark的基础数据模式是DataFrames，由命名列（类似关系表的字段定义）组织起来的分布式数据集合。

Spark SQL的使用如下图所示：



1）SqlContext从外部数据源加载数据为DataFrame，Spark支持从各种各样的数据源中创建DataFrame，内置支持的数据源包括ParquetFile、JsonFile、外部数据库、Hive表及RDD等。

2）利用DataFrame上丰富的API进行查询，转换

3）将结果进行展示或者存储为外部数据形式

# 1.Spark SQL CLI

Spark SQL CLI使得SparkSQL通过Hive MetaStore就可以直接对Hive进行查询。

1. 将hive-site.xml,core-site.xml和hdfs-site.xml配置文件放到conf目录中
2. 执行SQL CLI命令

*./bin/spark-sql*

使用如下：

*spark-sql> describe function 'map';*

*Function: map*

*Class: org.apache.spark.sql.catalyst.expressions.CreateMap*

*Usage: map(key0, value0, key1, value1, ...) - Creates a map with the given key/value pairs.*

*Time taken: 1.778 seconds, Fetched 3 row(s)*

*17/06/27 23:28:57 INFO CliDriver: Time taken: 1.778 seconds, Fetched 3 row(s)*

*spark-sql>*

*spark-sql> show tables; //查看hive中的表信息*

*default dim\_custom false*

*default dim\_day false*

*default dim\_product false*

*default dim\_salesperson false*

*default fact\_order false*

*spark-sql> select \* from dim\_custom;*

*scheduler.DAGScheduler: ResultStage 4 (processCmd at CliDriver.java:376) finished in 19.093 s*

*INFO scheduler.DAGScheduler: Job 4 finished: processCmd at CliDriver.java:376, took 19.265239 s*

*ct001 custom\_john ct001 beijing zgx-beijing beijing beijing internet*

*ct002 custom\_herry ct002 henan shlinjie shangdang henan interne*

# 2.Spark SQL API

DataFrame是以命名列方式组织的分布式数据集，等同于关系型数据库的表，可以从多个数据源构建数据集，如hive中结构表，外部数据库和已有的RDDs。在API中，程序开发人员使用DataSet<Row>来表示DataFrame。Spark API的入口是SparkSession，如下所示:

*import org.apache.spark.sql.SparkSession*

*val spark = SparkSession*

*.builder()*

*.appName("Spark SQL basic example")*

*.config("spark.some.config.option", "some-value")*

*.getOrCreate()*

*// For implicit conversions like converting RDDs to DataFrames*

*import spark.implicits.\_*

创建DataFrames，使用SparkSession，应用程序可以使用existing RDD来创建DataFrames，例如使用Json文件，如下所示：

*val df = spark.read.json("examples/src/main/resources/people.json")*

*// Displays the content of the DataFrame to stdout*

*df.show()*

*// +----+-------+*

*// | age| name|*

*// +----+-------+*

*// |null|Michael|*

*// | 30| Andy|*

*// | 19| Justin|*

*// +----+-------+*

DataFrames提供DSL语言来操作结构化数据，下例是基本的结构化数据处理：

*// This import is needed to use the $-notation*

*import spark.implicits.\_*

*// Print the schema in a tree format*

*df.printSchema()*

*// root*

*// |-- age: long (nullable = true)*

*// |-- name: string (nullable = true)*

*// Select only the "name" column*

*df.select("name").show()*

*// +-------+*

*// | name|*

*// +-------+*

*// |Michael|*

*// | Andy|*

*// | Justin|*

*// +-------+*

*// Select everybody, but increment the age by 1*

*df.select($"name", $"age" + 1).show()*

*// +-------+---------+*

*// | name|(age + 1)|*

*// +-------+---------+*

*// |Michael| null|*

*// | Andy| 31|*

*// | Justin| 20|*

*// +-------+---------+*

*// Select people older than 21*

*df.filter($"age" > 21).show()*

*// +---+----+*

*// |age|name|*

*// +---+----+*

*// | 30|Andy|*

*// +---+----+*

*// Count people by age*

*df.groupBy("age").count().show()*

*// +----+-----+*

*// | age|count|*

*// +----+-----+*

*// | 19| 1|*

*// |null| 1|*

*// | 30| 1|*

*// +----+-----+*

可以在SparkSession执行SQL查询，结果以DataFrame的形式返回，使用如下：

*// Register the DataFrame as a SQL temporary view*

*df.createOrReplaceTempView("people")*

*val sqlDF = spark.sql("SELECT \* FROM people")*

*sqlDF.show()*

*// +----+-------+*

*// | age| name|*

*// +----+-------+*

*// |null|Michael|*

*// | 30| Andy|*

*// | 19| Justin|*

*// +----+-------+*

# 3.Spark Thrift Server

Spark Thrift Server使用HiveServer2，启动JDBC/ODBC Server如下命令：

*./sbin/start-thriftserver.sh*

可以通过系统参数来启动：

*./sbin/start-thriftserver.sh \*

*--hiveconf hive.server2.thrift.port=<listening-port> \*

*--hiveconf hive.server2.thrift.bind.host=<listening-host> \*

*--master <master-uri>*

*...*

使用如下：

*sbin/start-thriftserver.sh*

*--hiveconf hive.server2.thrift.port=10000*

*--hiveconf hive.server2.thrift.bind.host=fys1.cmss.com*

*--master spark://fys1.cmss.com:7077*

# 4.Spark Submit Java Programe

Java程序的提交，命令如下：

*bin/spark-submit*

*--class org.apache.spark.examples.sql.JavaSparkSQLExample*

*examples/jars/spark-examples\_2.11-2.1.1.jar*

输出如下：

*+----+-------+*

*| age| name|*

*+----+-------+*

*|null|Michael|*

*| 30| Andy|*

*| 19| Justin|*

*+----+-------+*

*root*

*|-- age: long (nullable = true)*

*|-- name: string (nullable = true)*

*+-------+*

*| name|*

*+-------+*

*|Michael|*

*| Andy|*

*| Justin|*

*+-------+*

*+-------+---------+*

*| name|(age + 1)|*

*+-------+---------+*

*|Michael| null|*

*| Andy| 31|*

*| Justin| 20|*

*+-------+---------+*

*+---+----+*

*|age|name|*

*+---+----+*

*| 30|Andy|*

*+---+----+*

*+----+-----+*

*| age|count|*

*+----+-----+*

*| 19| 1|*

*|null| 1|*

*| 30| 1|*

*+----+-----+*

*+----+-------+*

*| age| name|*

*+----+-------+*

*|null|Michael|*

*| 30| Andy|*

*………………….*

https://jaceklaskowski.gitbooks.io/mastering-apache-spark/content/spark-sql-spark-sql.html