

# Spark SQL 核心知识 2

## 目录

1、Spark On YARN .....	1
1.1、配置 Spark 整合 YARN .....	1
1.2、Spark-Shell 测试.....	3
1.3、Spark-Submit 测试.....	3
2、Spark 整合 Hive .....	4
2.1、Spark 自带元数据库.....	4
2.2、Spark 整合 Hive 配置 .....	5
2.3、Spark SQL 脚本使用 .....	5
2.4、IDEA 编写 Spark 程序读写 Hive.....	7
2.4.1、spark-1.x 版本.....	7
2.4.2、spark-2.x 版本.....	8
3、SparkSQL 自定义聚合函数 .....	8
3.1、SparkSQL 定义普通函数 .....	8
3.2、定义 SparkSQL 的自定义聚集函数 .....	9
3.3、使用测试.....	12
4、SparkSQL 常用窗口分析函数 .....	12
5、综合练习.....	12

## 1、Spark On YARN

### 1.1、配置 Spark 整合 YARN

#### 1、设置 HADOOP\_CONF\_DIR

```
export HADOOP_CONF_DIR=/home/hadoop/apps/hadoop-2.7.6/etc/hadoop/
```

```
export JAVA_HOME=/usr/local/java/jdk1.8.0_73
export SPARK_MASTER_PORT=7077
export SPARK_DAEMON_JAVA_OPTS="-Dspark.deploy.recoveryMode=ZOOKEEPER -Dspark.deploy.zooUrls=zoo1:2181,zoo2:2181,zoo3:2181"
export SPARK_HISTORY_OPTS="-Dspark.history.ui.port=18080 -Dspark.history.retainedApplications=100"
export HADOOP_CONF_DIR=/home/hadoop/apps/hadoop-2.7.6/etc/hadoop/
```

2、拷贝 yarn-site.xml, hdfs-site.xml, core-site.xml 配置文件到\$SPARK\_HOME 下，重点是 yarn-site.xml，因为在搭建 spark HA 集群的时候，就已经把 core-site.xml 和 hdfs-site.xml 放置在这个目录下了。

```

hadoop02 x hadoop03 hadoop04 hadoop05
[hadoop@hadoop02 conf]$ ll
total 64
-rw-r--r-- 1 hadoop hadoop 1570 Jun 16 19:04 core-site.xml
-rw-rw-r-- 1 hadoop hadoop 996 Jun 16 19:01 docker.properties.template
-rw-rw-r-- 1 hadoop hadoop 1105 Jun 16 19:01 fairscheduler.xml.template
-rw-r--r-- 1 hadoop hadoop 4085 Jun 16 19:04 hdfs-site.xml
-rw-rw-r-- 1 hadoop hadoop 2909 Jul 26 18:50 hive-site.xml
-rw-rw-r-- 1 hadoop hadoop 2025 Jun 25 10:49 log4j.properties
-rw-rw-r-- 1 hadoop hadoop 2025 Jun 16 19:01 log4j.properties.template
-rw-rw-r-- 1 hadoop hadoop 7801 Jun 16 19:01 metrics.properties.template
-rw-rw-r-- 1 hadoop hadoop 893 Jun 16 19:01 slaves
-rw-rw-r-- 1 hadoop hadoop 865 Jun 16 19:01 slaves.template
-rw-rw-r-- 1 hadoop hadoop 1400 Jul 19 20:57 spark-defaults.conf
-rw-rw-r-- 1 hadoop hadoop 1292 Jun 16 19:01 spark-defaults.conf.template
-rwxrwxr-x 1 hadoop hadoop 4685 Jul 19 20:57 spark-env.sh
-rw-r--r-- 1 hadoop hadoop 2503 Jul 19 21:04 yarn-site.xml
[hadoop@hadoop02 conf]$

```

### 3、启动 Spark-Shell

```

spark-shell \
--master yarn \
--deploy-mode client \
--executor-memory 512m \
--total-executor-cores 1

```

```

hadoop03 hadoop02 hadoop03 hadoop04 hadoop05
[hadoop@hadoop04 conf]$ ~/apps/spark-2.3.1-bin-hadoop2.7/bin/spark-shell \
> --master yarn \
> --deploy-mode client \
> --executor-memory 512m \
> --total-executor-cores 1
18/07/30 08:37:13 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
18/07/30 08:37:19 WARN Client: Neither spark.yarn.jars nor spark.yarn.archive is set, falling back to uploading libraries under SPARK_HOME.
Spark context Web UI available at http://hadoop04:4040
Spark context available as 'sc' (master = yarn, app id = application_1532833611771_0015).
Spark session available as 'spark'.
Welcome to

  ____  __
 / ___/ /  _ \
/ /   / /  / \
/ /___/ /  /_/ \
\____/_/  \____/
version 2.3.1

Using Scala version 2.11.8 (Java HotSpot(TM) 64-Bit Server VM, Java 1.8.0_73)
Type in expressions to have them evaluated.
Type :help for more information.

scala>

```

Spark Master at spark1-1

Spark Master at spark1-2

History Server

Nameservice information...

Nameservice information...

All Applications

Job History

hadoop@4108021:~\$

hadoop@4108021:~\$

hadoop@4108021:~\$

hadoop@4108021:~\$

hadoop@4108021:~\$

hadoop

hadoop

hadoop

hadoop

hadoop

hadoop

Cluster

About Nodes

Node Labels

Applications

NEW

NEW SAVING

SUBMITTED

ACCEPTED

RUNNING

FINISHED

FAILED

KILLED

Scheduler

Tools

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	VCores Used	VCores Total	VCores Reserved	Active Nodes	Decommissioned Nodes	Lost Nodes
15	0	1	14	3	3 GB	24 GB	0 B	3	24	0	3	0	1

Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minimum Allocation	Maximum Allocation
Capacity Scheduler	[MEMORY]	<memory:1024, vCores:1>	<memory:8192, vCores:8>

Show 20 entries

ID	User	Name	Application Type	Queue	StartTime	FinishTime	State	FinalStatus	Progress	Tracked
application_1532833611771_0015	hadoop	Spark shell	SPARK	default	Mon Jul 30 08:37:22 +0800 2018	N/A	RUNNING	UNDEFINED		Application

如果报错是关于 yarn-application 资源相关的问题的, 请在 yarn-site.xml 配置文件中加入:

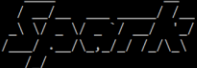
```

<property>
  <name>yarn.nodemanager.vmem-check-enabled</name>
  <value>false</value>
</property>

```

## 1.2、Spark-Shell 测试

```
[hadoop@hadoop02 ~]$ hadoop03 | hadoop04 | hadoop05 x
[hadoop@hadoop05 ~]$ spark-shell --master yarn --executor-memory 512m --total-executor-cores 1
18/07/30 08:04:19 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
18/07/30 08:04:26 WARN Client: Neither spark.yarn.jars nor spark.yarn.archive is set, falling back to uploading libraries under SPARK_HOME.
Spark context Web UI available at http://hadoop05:4040
Spark context available as 'sc' (master = yarn, app id = application_1532833611771_0012).
Spark session available as 'spark'.
Welcome to

 version 2.3.1

Using Scala version 2.11.8 (Java HotSpot(TM) 64-Bit Server VM, Java 1.8.0_73)
Type in expressions to have them evaluated.
Type :help for more information.

scala>

scala> val listWords = List("hello world", "word count", "hello hello lijie", "lalala lijie")
listWords: List[String] = List(hello world, word count, hello hello lijie, lalala lijie)

scala> val countWord1 = listWords.map(_.split(" ")).flatten.map((_,1)).groupBy(_._1).map(t => (t._1,t._2.size)).toList.sortBy(_._2).reverse
countWord1: List[(String, Int)] = List((hello,3), (lijie,2), (lalala,1), (word,1), (world,1), (count,1))

scala>
```

### 1.3、Spark-Submit 测试

```
~/apps/spark-2.3.1-bin-hadoop2.7/bin/spark-submit \
--class org.apache.spark.examples.SparkPi \
--master yarn \
--deploy-mode client \
--executor-memory 512m \
--total-executor-cores 1 \
```

```
~/apps/spark-2.3.1-bin-hadoop2.7/examples/jars/spark-examples_2.11-2.3.1.jar \  
100
```

```
[hadoop@hadoop02 conf]$ ~/apps/spark-2.3.1-bin-hadoop2.7/bin/spark-submit \  
> --class org.apache.spark.examples.SparkPi \  
> --master yarn \  
> --deploy-mode client \  
> --executor-memory 512m \  
> --total-executor-cores 1 \  
> ~/apps/spark-2.3.1-bin-hadoop2.7/examples/jars/spark-examples_2.11-2.3.1.jar \  
> 100  
18/07/30 08:30:47 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable  
18/07/30 08:30:50 WARN Client: Neither spark.yarn.jars nor spark.yarn.archive is set, falling back to uploading libraries under SPARK_HOME.  
Pi is roughly 3.1412851141285114  
[hadoop@hadoop02 conf]$
```

高可用方式下使用 spark-submit 提交一个任务到高可用的 YARN 集群，使用 cluster 模式：

```
~/apps/spark-2.3.1-bin-hadoop2.7/bin/spark-submit \  
--class org.apache.spark.examples.SparkPi \  
--master yarn \  
--deploy-mode cluster \  
--executor-memory 512m \  
--total-executor-cores 1 \  
~/apps/spark-2.3.1-bin-hadoop2.7/examples/jars/spark-examples_2.11-2.3.1.jar \  
100
```

```
[hadoop@hadoop02 conf]$ ~/apps/spark-2.3.1-bin-hadoop2.7/bin/spark-submit \  
> --class org.apache.spark.examples.SparkPi \  
> --master yarn \  
> --deploy-mode cluster \  
> --executor-memory 512m \  
> --total-executor-cores 1 \  
> ~/apps/spark-2.3.1-bin-hadoop2.7/examples/jars/spark-examples_2.11-2.3.1.jar \  
> 100  
18/07/30 08:32:51 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable  
18/07/30 08:32:52 WARN Client: Neither spark.yarn.jars nor spark.yarn.archive is set, falling back to uploading libraries under SPARK_HOME.  
[hadoop@hadoop02 conf]$
```

## 2、Spark 整合 Hive

### 2.1、Spark 自带元数据库

在\$SPARK\_HOME/conf 目录下如果没有 hive-site.xml 文件的情况下。 Spark 使用的元数据库是默认的 in-memory 模式，也就是使用自带的 derby 在当前会话中有效

```

spark-sql> create database if not exists myhive;
18/07/30 08:10:20 WARN ObjectStore: Failed to get database myhive, returning NoSuchObjectException
Time taken: 0.171 seconds
spark-sql> show databases;
default
myhive
Time taken: 0.044 seconds, Fetched 2 row(s)
spark-sql> use myhive;
Time taken: 0.032 seconds
spark-sql> drop table if exists student;
Time taken: 0.076 seconds
spark-sql> create table student(id int, name string, sex string, age int, department string) row format delimited fields terminated by ',';
18/07/30 08:10:35 WARN HiveMetaStore: Location: file:/home/hadoop/apps/spark-2.3.1-bin-hadoop2.7/conf/spark-warehouse/myhive.db/student spec
Time taken: 0.396 seconds
spark-sql> load data local inpath "/home/hadoop/student.txt" into table student;
18/07/30 08:10:40 ERROR KeyProviderCache: Could not find uri with key [dfs.encryption.key.provider.uri] to create a keyProvider !!
Time taken: 0.295 seconds
spark-sql> select * from student;
95002 刘晨 女 19 IS
95017 王凤娟 女 18 IS
95018 王一 女 19 IS
95013 冯伟 男 21 CS
95014 王小丽 女 19 CS
95019 邢小丽 女 19 IS
95020 赵钱 男 21 IS
95003 王敏 女 22 MA
95004 张立 男 19 IS
95012 孙花 女 20 CS
95010 孔小涛 男 19 CS
95005 刘刚 男 18 MA
95006 孙庆 男 23 CS
95007 易思玲 女 19 MA
95008 李娜 女 18 CS
95021 周二 男 17 MA
95022 郑明 男 20 MA
95001 李勇 男 20 CS
95011 包小柏 男 18 MA
95009 梦圆 女 18 MA
95015 王君 男 18 MA
Time taken: 0.842 seconds, Fetched 21 row(s)
spark-sql> select count(*) from student;

```

## 2.2、Spark 整合 Hive 配置

把 hive-site.xml 配置文件放到\$SPARK\_HOME/conf 目录中

然后两种方式配置:

启动 hive 的 metastore 服务

1、在 hive-site.xml 配置文件中加入以下一个配置:

```

<property>
  <name>hive.metastore.uris</name>
  <value>thrift://hadoop05:9083</value>
</property>

```

2、启动命令:

```
nohup hive --service metastore 1>/home/hadoop/logs/hive_thriftserver.log 2>&1 &
```

3、然后启动 spark-sql 直接启动:

```
[hadoop@hadoop02 conf]$ spark-sql
```

## 2.3、Spark SQL 脚本使用

在 hive 中的操作:

hadoop02 | hadoop02 (1) | hadoop03 | hadoop03 (1) | hadoop04 | **hadoop05** x | hadoop05 (1)

```
hive> show databases;
OK
default
mydb_test
myhive
Time taken: 2.512 seconds, Fetched: 3 row(s)
hive> use myhive;
OK
Time taken: 0.047 seconds
hive> show tables;
OK
student
student_s
Time taken: 0.067 seconds, Fetched: 2 row(s)
hive> select * from student;
OK
95002  刘晨      女      19      IS
95017  王凤娟    女      18      IS
95018  王一      女      19      IS
95013  冯伟      男      21      CS
95014  王小丽    女      19      CS
95019  邢小丽    女      19      IS
95020  赵钱      男      21      IS
95003  王敏      女      22      MA
95004  张立      男      19      IS
95012  孙花      女      20      CS
95010  孔小涛    男      19      CS
95005  刘刚      男      18      MA
95006  孙庆      男      23      CS
95007  易思玲    女      19      MA
95008  李娜      女      18      CS
95021  周二      男      17      MA
95022  郑明      男      20      MA
95001  李勇      男      20      CS
95011  包小柏    男      18      MA
95009  梦圆圆    女      18      MA
95015  王君      男      18      MA
Time taken: 1.068 seconds, Fetched: 21 row(s)
hive>
```

在 spark-sql 中的操作:

```
[hadoop@hadoop05 ~]$ spark-sql
18/07/30 08:41:11 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
18/07/30 08:41:11 WARN HiveConf: HiveConf of name hive.enable.spark.execution.engine does not exist
spark-sql> show databases;
default
mydb_test
myhive
Time taken: 1.38 seconds, Fetched 3 row(s)
spark-sql> use myhive;
Time taken: 0.036 seconds
spark-sql> show tables;
myhive student false
myhive student_s false
Time taken: 0.107 seconds, Fetched 2 row(s)
spark-sql> select * from student;
95002  刘晨      女      19      IS
95017  王凤娟    女      18      IS
95018  王一      女      19      IS
95013  冯伟      男      21      CS
95014  王小丽    女      19      CS
95019  邢小丽    女      19      IS
95020  赵钱      男      21      IS
95003  王敏      女      22      MA
95004  张立      男      19      IS
95012  孙花      女      20      CS
95010  孔小涛    男      19      CS
95005  刘刚      男      18      MA
95006  孙庆      男      23      CS
95007  易思玲    女      19      MA
95008  李娜      女      18      CS
95021  周二      男      17      MA
95022  郑明      男      20      MA
95001  李勇      男      20      CS
95011  包小柏    男      18      MA
95009  梦圆圆    女      18      MA
95015  王君      男      18      MA
Time taken: 1.004 seconds, Fetched 21 row(s)
spark-sql>
```

## 2.4、IDEA 编写 Spark 程序读写 Hive

### 2.4.1、spark-1.x 版本

```
/**
 * 作者: 马中华  https://blog.csdn.net/zhongqi2513
 * 时间: 2018/5/26 17:53
 *
 * 描述: 编写SparkSQL程序读取Hive中的数据
 *
 * 执行之前, 必须先把metastore服务启动起来, 启动Hive的metastore服务
 * nohup hive --service metastore 1>/home/hadoop/logs/hive_thriftserver.log 2>&1 &
 *
 * 启动hiveserver2服务
 */
object SparkSQL_Hive_1 {

  def main(args: Array[String]): Unit = {

    // -DHADOOP_USER_NAME=hadoop 或者不添加下面的代码, 可以给JVM设置一个这样的参数
    System.setProperty("HADOOP_USER_NAME", "hadoop")
    val conf = new SparkConf().setMaster("local").setAppName("SparkSQL_Hive_1")
    val sc = new SparkContext(conf)
    val hiveContext = new HiveContext(sc)

    hiveContext.sql(sqlText = "select * from myhive.student").show()

    sc.stop()
  }
}
```

## 2.4.2、spark-2.x 版本

```
/**
 * 作者: 马中华  https://blog.csdn.net/zhongqi2513
 * 时间: 2018/5/26 17:57
 *
 * 描述: 使用Spark2.x版本的API编写Spark代码整合Hive
 */
object SparkSQL_Hive_2 {

  def main(args: Array[String]): Unit = {

    System.setProperty("HADOOP_USER_NAME", "hadoop")

    val sparkSession: SparkSession = SparkSession.builder()
      .appName(name = "SparkSQL_Hive_2")
      .master(master = "local")
      .enableHiveSupport()
      .getOrCreate()

    sparkSession.sql(sqlText = "select * from myhive.student").show()

    sparkSession.stop()
  }
}
```

## 3、SparkSQL 自定义聚合函数

### 3.1、SparkSQL 定义普通函数

要点: spark.udf.register("function\_name", function)



```
/**
 * 第一步： 获取程序入口
 */
val sparkConf = new SparkConf()
sparkConf.setAppName("SparkSQL_UAF_Length").setMaster("local")
val sparkContext = new SparkContext(sparkConf)
val sqlContext = new SQLContext(sparkContext)

/**
 * 第二步： 获取到一个DataFrame， 然后注册为一张表
 * JDBC : 三个参数
 * url: String
 * table: String
 * properties: Properties
 */
val url = "jdbc:mysql://hadoop02:3306/bigdata"
val table = "student"
val properties = new Properties()
properties.put("user", "root")
properties.put("password", "root")
val studentDF: DataFrame = sqlContext.read.jdbc(url = url, table = table, properties = properties)

/**
 * 第三步： 把这个dataFrame注册为一张临时表
 */
studentDF.createTempView(viewName = "student")

/**
 * 第四步： 定义一个函数
 */
sqlContext.udf.register(name = "strLength", func = (x:String) => x.length)

/**
 * 第五步： 使用这个函数做一个操作， 求出某个字段的长度
 */
sqlContext.sql(sqlText = "select strLength(name) as name_len from student").show()

/**
 * 第六步： 程序完结， 关闭资源
 */
sparkContext.stop()
```

## 3.2、定义 SparkSQL 的自定义聚集函数

Class MyUDAF extends UserDefinedAggregationFunction  
spark.udf.register("function\_name", function)

```
package com.aura.mazh.spark.sql.udf

import org.apache.spark.sql.Row
import org.apache.spark.sql.expressions.{MutableAggregationBuffer,
UserDefinedAggregateFunction}
import org.apache.spark.sql.types._

/**
 * 定义 Student.txt 中， 求学生平均年龄的一个自定义函数
 *
 * 计算规则：
 * 1、 统计出所有学生的年龄之和 total
 * 2、 统计出所有学生的个数 count
 */
object SparkSQL_UDAF_AvgAge extends UserDefinedAggregateFunction{
```

```
/**
 * 定义输入的数据的类型
 */
override def inputSchema: StructType = StructType(
  StructField("age", DoubleType, true) :: Nil
)

/**
 * 定义辅助字段:
 *
 * 1、辅助字段1: 用来记录所有年龄之和 total
 * 2、辅助字段2: 用来总记录所有学生的个数 count
 */
override def bufferSchema: StructType = StructType(
  StructField("total", DoubleType, true)::
  StructField("count", IntegerType, true)::
  Nil
)

/**
 * 计算学生的平均年龄 计算公式: 学生年龄的总和 / 学生总数
 *
 * 所以要初始化要两个辅助字段:
 *   total : 0.0
 *   count : 0
 */
override def initialize(buffer: MutableAggregationBuffer): Unit = {
  buffer.update(0, 0.0)
  buffer.update(1, 0)
}

/**
 * 每次给一条记录, 然后进行累加。进行累加变量buffer的状态更新
 * 这是一个局部操作。
 */
override def update(buffer: MutableAggregationBuffer, input: Row): Unit = {
  val lastTotal = buffer.getDouble(0)
  val lastCount = buffer.getInt(1)
  val currentSalary = input.getDouble(0)
  buffer.update(0, lastTotal + currentSalary)
  buffer.update(1, lastCount+1)
}
```

```
/**
 * 当局部操作完成，最后需要一个全局合并的操作
 * 就相当于 reducer 阶段的最终合并
 */
override def merge(buffer1: MutableAggregationBuffer, buffer2: Row): Unit = {
    val total1 = buffer1.getDouble(0)
    val count1 = buffer1.getInt(1)

    val total2 = buffer2.getDouble(0)
    val count2 = buffer2.getInt(1)

    buffer1.update(0, total1 + total2)
    buffer1.update(1, count1 + count2)
}

/**
 * 计算平均年龄
 */
override def evaluate(buffer: Row): Any = {
    val total = buffer.getDouble(0)
    val count = buffer.getInt(1)
    total / count
}

/**
 * 返回结果数据类型
 */
override def dataType: DataType = DoubleType

/**
 * 输入和输出的字段类型是否匹配。也即是否一致
 */
override def deterministic: Boolean = true
}
```

### 3.3、使用测试

```
/**
 * 自定义UDAF函数
 * 作业： 求出每个部门的平均年龄
 */
object SparkSQL_UDAF_AvgAge_Test {

  def main(args: Array[String]): Unit = {

    System.setProperty("HADOOP_USER_NAME", "hadoop")

    /**
     * 第一步： 获取编程入口
     */
    val conf = new SparkConf()
    conf.setMaster("local").setAppName("UDAFTest")
    val sparkContext = new SparkContext(conf)
    val hiveContext = new HiveContext(sparkContext)

    /**
     * 第二步： 进行自定义函数的注册
     */
    hiveContext.udf.register( name = "avg_age", udaf = SparkSQL_UDAF_AvgAge)

    /**
     * 第三步： 读取hive表，调用自定义函数
     */
    hiveContext.sql( sqlText = "select avg_age(age) as avgage, department from myhive.student group by department").show()

    /**
     * 第四步： 回收资源
     */
    sparkContext.stop()
  }
}
```

## 4、SparkSQL 常用窗口分析函数

见文档“**案例 08--Hive 各种分析函数案例.rar**”

## 5、综合练习

SparkCore 影评案例的第八题

SparkCore 拉勾的第四题

NBA 案例

sparkcore 实现

sparkcore 实现

sparksql 实现

sparksql 实现