Spark SQL

点击一键运行代码→ 获取更多速查表→ 使用说明书 [视频 & 图文]→ bilibili Show Me AI

Spark 是基于内存计算的大数据并行计算框架。

它包含 MapReduce 计算模型，而且高效地支持更多计算模式，包括交互式查询和流处理。Spark 适用于各种各样原先需要多种不同的分布式平台的场景，包括批处理、迭代算法、交互式查询、流处理。Spark 生态系统已经发展成为一个包含多个子项目的集合，其中包含 SparkSQL、Spark Streaming、GraphX/GraphFrame、MLlib、SparkR 等子项目。

Spark SQL 是 Apache Spark 处理结构化数据的模块。

1. 初始化 SparkSession

初始化

SparkSession 用于创建数据帧，将数据帧注册为表，执行 SQL 查询，缓存表及读取 Parquet 文件。

```
> from pyspark.sql import SparkSession
  spark = SparkSession \
          .builder \
          .appName("Python Spark SQL basic example") \
          .config("spark.some.config.option", "some-value") \
          .getOrCreate()
```

3. 查阅数据信息

查阅 spark Dataframe 的信息

```
> df.dtypes # 返回 df 的列名与数据类型
> df.show() # 显示 df 的内容

> df.head(n) # 返回前 n 行数据
> df.first() # 返回第 1 行数据
> df.take(n) # 返回前 n 行数据

> df.schema # 返回 df 的 Schema

> df.describe().show() # 汇总统计数据
> df.columns # 返回 df 的列名
> df.count() # 返回 df 的行数

> df.distinct().count() # 返回 df 中不重复的行数

> df.printSchema() # 返回 df 的 Schema
> df.explain() # 返回逻辑与实体方案
```

Show Me AI

2. 创建数据帧

从 RDD 创建

```
> from pyspark.sql.types import *
```

推断 Schema

```
> sc = spark.sparkContext
> lines = sc.textFile("people.txt")
> parts = lines.map(lambda l: l.split(","))
> people = parts.map(lambda p: Row(name=p[0],age=int(p[1])))
> peopledf = spark.createDataFrame(people)
```

指定 Schema

```
> people = parts.map(lambda p: Row(name=p[0], age=int(p[1].strip())))
> schemaString = "name age"
> fields = [StructField(field_name, StringType(), True) for field_name in schemaString.split()]
> schema = StructType(fields)
> spark.createDataFrame(people, schema).show()
```

```
name  age
Mine  28
Filip 29
Jonathan 30
```

从 Spark 数据源创建

JSON

```
> df = spark.read.json("customer.json")
> df.show()
```

```
address    age    firstName    lastName    phoneNumber
[NewYork,10021,N...    25    John    Smith    [[212 555-1234,ho...]]
[NewYork,10021,N...    21    Jane    Doe      [[322 888-1234,ho...]]
```

```
> df2 = spark.read.load("people.json", format="json")
```

Parquet 文件

```
> df3 = spark.read.load("users.parquet")
```

文本文件

```
> df4 = spark.read.text("people.txt")
```

4. 重复值

dropDuplicates 函数

```
> df = df.dropDuplicates()
```


5. 查询

```
> from pyspark.sql import functions as F
```

Select

```
# 显示 firstName 列的所有条目
> df.select("firstName").show()
> df.select("firstName","lastName").show()

# 显示 firstName、age 的所有条目和类型
> df.select("firstName", "age", explode("phoneNumber").alias("contactInfo")) \
    .select("contactInfo.type", "firstName","age") \
    .show()

# 显示 firstName 和 age 列的所有 记录，并对 age 记录添加 1
> df.select(df["firstName"], df["age"]+ 1).show()

# 显示所有小于 24 岁的记录
> df.select(df['age'] > 24).show()
```

When

```
# 显示 firstName，且大于 30 岁显示 1，小于 30 岁显示 0
> df.select("firstName", F.when(df.age > 30, 1).otherwise(0)).show()

# 显示符合指定条件的 firstName 列 的记录
> df[df.firstName.isin("Jane","Boris")].collect()
```

Like

```
# 显示 lastName 列中包含 Smith 的 firstName 列的记录
> df.select("firstName", df.lastName.like("Smith")).show()
```

Startswith - Endswith

```
# 显示 lastName 列中以 Sm 开头的 firstName 列的记录
> df.select("firstName", df.lastName.startswith("Sm")).show()

# 显示以 th 结尾的 lastName
> df.select(df.lastName.endswith("th")).show()
```

Substring

```
# 返回 firstName 的子字符串
> df.select(df.firstName.substr(1, 3).alias("name")).collect()
```

Between

```
# 显示介于 22 岁至 24 岁之间的 age 列的记录
> df.select(df.age.between(22, 24)).show()
```

6. 添加、修改、删除列

添加列

```
> df = df.withColumn('city',df.address.city) \
        .withColumn('postalCode',df.address.postalCode) \
        .withColumn('state',df.address.state) \
        .withColumn('streetAddress',df.address.streetAddress) \
        .withColumn('telePhoneNumber', explode(df.phoneNumber.number)) \
        .withColumn('telePhoneType', explode(df.phoneNumber.type))
```

修改列

```
> df = df.withColumnRenamed('telePhoneNumber', 'phoneNumber')
```

删除列

```
> df = df.drop("address", "phoneNumber")
> df = df.drop(df.address).drop(df.phoneNumber)
```

7. 分组

groupBy 操作

```
> df.groupBy("age").count().show() # 按 age 列分组，统计每组人数
```

8. 筛选

filter 筛选

```
> df.filter(df["age"]>24).show() # 按 age 列筛选，保留年龄大于 24 岁的
```

9. 排序

sort 与 orderBy 操作

```
> peopledf.sort(peopledf.age.desc()).collect()
> df.sort("age", ascending=False).collect()
> df.orderBy(["age","city"],ascending=[0,1]).collect()
```

10. 替换缺失值

replace 操作

```
> df.na.fill(50).show() # 用一个值替换空值
> df.na.drop().show() # 去除 df 中为空值的行
> df.na.replace(10, 20).show() # 用一个值替换另一个值
```

11. 重分区

repartition 重分区

```
> df.repartition(10).rdd.getNumPartitions() # 将 df 拆分为 10 个分区
> df.coalesce(1).rdd.getNumPartitions() # 将 df 合并为 1 个分区
```

12. 运行 SQL 查询

将数据帧注册为视图

```
> peopledf.createGlobalTempView("people")
> df.createTempView("customer")
> df.createOrReplaceTempView("customer")
```

查询视图

```
> df5 = spark.sql("SELECT * FROM customer").show()
> peopledf2 = spark.sql("SELECT * FROM global_temp.people").show()
```

13. 输出

数据结构

```
> rdd1 = df.rdd          # 将 df 转换为 RDD
> df.toJSON().first()    # 将 df 转换为 RDD 字符串
> df.toPandas()          # 将 df 的内容转为 Pandas 的数据帧
```

保存至文件

```
> df.select("firstName", "city").write.save("nameAndCity.parquet")
> df.select("firstName", "age").write.save("namesAndAges.son",format="json")
```

14. 终止 SparkSession

终止 spark session

```
> spark.stop()
```

ShowMeAI

作者 | 韩信子 @ShowMeAI
设计 | 南 乔 @ShowMeAI
参考 | datacamp cheatsheet