



Python 数据科学 速查表

PySpark - SQL Basics

天善智能 商业智能与大数据社区 www.hellobi.com



PySpark 与 Spark SQL

Spark SQL 是 Apache Spark 处理结构化数据的模块。



初始化 SparkSession

SparkSession 用于创建数据框，将数据框注册为表，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()
```

创建数据框

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 |
+-----+-----+-----+-----+-----+
|[New York,10021,N...| 25 | John | Smith |[212 555-1234,ho...|
|[New York,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")
```

查阅数据信息

```
>>> df.dtypes
>>> df.show()
>>> df.head()
>>> df.first()
>>> df.take(2)
>>> df.schema
```

返回 df 的列名与数据类型
显示 df 的内容
返回前 n 行数据
返回第 1 行数据
返回前 n 行数据
返回 df 的 Schema

重复值

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

查询

```
>>> from pyspark.sql import functions as F
Select
>>> df.select("firstName").show()
>>> df.select("firstName", "lastName") \
    .show()
>>> df.select("firstName",
    "age",
    explode("phoneNumber") \
    .alias("contactInfo")) \
    .select("contactInfo.type",
    "firstName",
    "age") \
    .show()
>>> df.select(df["firstName"],df["age"]+ 1)
    .show()
>>> df.select(df["age"] > 24).show()
When
>>> df.select("firstName",
    F.when(df.age > 30, 1) \
    .otherwise(0)) \
    .show()
>>> df[df.firstName.isin("Jane", "Boris")]
    .collect()
Like
>>> df.select("firstName",
    df.lastName.like("Smith")) \
    .show()
Startswith - Endswith
>>> df.select("firstName",
    df.lastName \
    .startswith("Sm")) \
    .show()
>>> df.select(df.lastName.endswith("th")) \
    .show()
Substring
>>> df.select(df.firstName.substr(1, 3) \
    .alias("name")) \
    .collect()
Between
>>> df.select(df.age.between(22, 24)) \
    .show()
```

显示 firstName 列的所有条目

显示 firstName、age 的所有条目和类型

显示 firstName 和 age 列的所有记录，并对 age 记录添加 1
显示所有小于 24 岁的记录

显示 firstName，且大于 30 显示 1，小于 30 显示 0

显示符合指定条件的 firstName 列的记录

显示 lastName 列中包含 Smith 的 firstName 列的记录

显示 lastName 列中以 Sm 开头的 firstName 列的记录

显示以 th 结尾的 lastName

返回 firstName 的子字符串

显示介于 22 岁至 24 岁之间的 age 列的记录

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)
```

分组

```
>>> df.groupBy("age") \
    .count() \
    .show()
```

按 age 列分组，统计每组人数

筛选

```
>>> df.filter(df["age"]>24).show()
```

按 age 列筛选，保留年龄大于 24 岁的

排序

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

替换缺失值

```
>>> df.na.fill(50).show()
>>> df.na.drop().show()
>>> df.na \
    .replace(10, 20) \
    .show()
```

用一个值替换空值
去除 df 中为空值的行
用一个值替换另一个值

重分区

```
>>> df.repartition(10) \
    .rdd \
    .getNumPartitions()
>>> df.coalesce(1).rdd.getNumPartitions()
```

将 df 拆分为 10 个分区

将 df 合并为 1 个分区

运行 SQL 查询

将数据框注册为视图

```
>>> 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()
```

输出

数据结构

```
>>> rdd1 = df.rdd
>>> df.toJSON().first()
>>> df.toPandas()
```

将 df 转换为 RDD
将 df 转换为 RDD 字符串
将 df 的内容转为 Pandas 的数据框

保存至文件

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

终止 SparkSession

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

原作者

DataCamp
Learn Python for Data Science Interactively

