

HW_2 PySpark DataFrame API

Урок 2. PySpark DataFrame API

FINISHED

Took 0 sec. Last updated by anonymous at January 27 2021, 11:25:01 PM.

0. Подготовка

FINISHED

Нужно скопировать себе эту тетрадку и предоставить доступ к копии на чтение, запись и запуск тетрадки пользователю `admin` .
Параграфы с генерацией данных и созданием семплов запускать не нужно, они оставлены для ознакомления

Took 0 sec. Last updated by anonymous at January 27 2021, 11:44:33 PM.

Генерация events таблицы

FINISHED

```
import org.apache.spark.mllib.random.RandomRDDs._
import java.time.LocalDate
import java.time.format.DateTimeFormatter

val dates = (0 to 14).map(LocalDate.of(2020, 11, 1).plusDays(_).format(DateTimeFormatter.ofPattern("yyyy-MM-dd"))).toSeq

def generateCity(r: Double): String = if (r < 0.9) "BIG_CITY" else "SMALL_CITY_" + scala.math.round((r - 0.9) * 1000)

def generateCityUdf = udf(generateCity _)

// spark.sql("drop table hw2.events_full")

for(i <- dates) {
  uniformRDD(sc, 1000000L, 1)
    .toDF("uid")
    .withColumn("date", lit(i))
    .withColumn("city", generateCityUdf($"uid"))
    .selectExpr("date", " sha2(cast(uid as STRING), 256) event_id", "city")
    .withColumn("skew_key", when($"city" === "BIG_CITY", lit("big_event")).otherwise($"event_id"))
    .write.mode("append")
    .partitionBy("date")
    .saveAsTable("hw2.events_full")
}
```

Генерация events_sample HW_2 PySpark DataFrame API

READY

```
spark.table("hw2.events_full")  
.select("event_id")  
.sample(0.0005)  
.repartition(2)  
.write.mode("overwrite")  
.saveAsTable("hw2.sample")
```

READY

```
spark.table("hw2.sample")  
.limit(100)  
.coalesce(1)  
.write.mode("overwrite")  
.saveAsTable("hw2.sample_small")
```

READY

```
spark.table("hw2.events_full")  
.select("event_id")  
.sample(0.003)  
.repartition(1)  
.write.mode("overwrite")  
.saveAsTable("hw2.sample_big")
```

READY

```
spark.table("hw2.events_full")  
.select("event_id")  
.sample(0.015)  
.repartition(1)  
.write.mode("overwrite")  
.saveAsTable("hw2.sample_very_big")
```

FINISHED

Задание 1

Для упражнений стренирован большой набор синтетических данных в таблице `hw2.events_full`.
Из этого набора данных созданы маленькие (относительно исходного набора) таблицы разного размера `kotelnikov.sample_[small, big, very_big]`.

Ответить на вопросы:

- какова структура таблиц
- сколько в них записей
- сколько места занимают данные

Took 0 sec. Last updated by anonymous at January 29 2021, 12:12:30 AM.

Создаем датафреймы

FINISHED

```
%pyspark

events_full = spark.table("hw2.events_full")
sample = spark.table("hw2.sample")
sample_small = spark.table("hw2.sample_small")
sample_big = spark.table("hw2.sample_big")
sample_very_big = spark.table("hw2.sample_very_big")
```

Hive Session ID = 779aff96-5614-4914-8b96-dd7c1fc741ac

Took 2 min 2 sec. Last updated by anonymous at January 29 2021, 11:55:45 PM.

- структура таблиц

FINISHED

```
%pyspark

events_full.printSchema()

root
 |-- event_id: string (nullable = true)
```

```
-- city: string (nullable = true)
-- skew_key: string (nullable = true)
-- date: string (nullable = true)
```

HW_2 PySpark DataFrame API

Took 2 min 5 sec. Last updated by anonymous at January 29 2021, 10:40:56 PM. (outdated)

- сколько записей в таблицах

SPARK JOB FINISHED

```
%pyspark
for i in [
    "hw2.events_full",
    "hw2.sample_small",
    "hw2.sample_big",
    "hw2.sample_very_big"]:
    print(
        "Records in {table}: {counts}".format(
            table=i counts=spark.table(i).count())
```

Records in hw2.events_full: 110000000

Records in hw2.sample_small: 100

Records in hw2.sample_big: 330052

Records in hw2.sample_very_big: 1651244

Took 5 sec. Last updated by anonymous at January 29 2021, 10:45:20 PM. (outdated)

- сколько места занимают данные

FINISHED

```
%sh

hdfs dfs -count -h -v /apps/spark/warehouse/hw2.db/events_full
hdfs dfs -count -h /apps/spark/warehouse/hw2.db/sample
hdfs dfs -count -h /apps/spark/warehouse/hw2.db/sample_small
hdfs dfs -count -h /apps/spark/warehouse/hw2.db/sample_big
hdfs dfs -count -h /apps/spark/warehouse/hw2.db/sample_very_big
```

DIR_COUNT	FILE_COUNT	CONTENT_SIZE	PATHNAME
9	12	7.3 G	/apps/spark/warehouse/hw2.db/events_full
1	3	3.4 M	/apps/spark/warehouse/hw2.db/sample
1	2	7.0 K	/apps/spark/warehouse/hw2.db/sample_small
1	2	20.0 M	/apps/spark/warehouse/hw2.db/sample_big
1	2	100.3 M	/apps/spark/warehouse/hw2.db/sample_very_big

Took 13 sec. Last updated by anonymous at January 29 2021, 11:19:40 PM. (outdated)

Задание 2

FINISHED

HW_2 PySpark DataFrame API

Получить планы запросов для джойна большой таблицы `hw2.events_full` с каждой из таблиц `hw2.sample`, `hw2.sample_big`, `hw2.sample_very_big` по полю `event_id`. В каких случаях используется *BroadcastHashJoin*?

BroadcastHashJoin автоматически выполняется для джойна с таблицами, размером меньше параметра

`spark.sql.autoBroadcastJoinThreshold`.

Узнать его значение можно командой:

```
spark.conf.get("spark.sql.autoBroadcastJoinThreshold")
```

Took 0 sec. Last updated by anonymous at January 29 2021, 11:35:39 PM.

```
%pyspark
print("spark.sql.autoBroadcastJoinThreshold")
spark.conf.get("spark.sql.autoBroadcastJoinThreshold")
```

FINISHED

```
spark.sql.autoBroadcastJoinThreshold
u'26214400'
```

Took 0 sec. Last updated by anonymous at January 29 2021, 10:40:56 PM.

```
%pyspark

events_full.join(sample, "event id").explain()

== Physical Plan ==
*(2) Project [event_id#0, city#1, skew_key#2, date#3]
+- *(2) BroadcastHashJoin [event_id#0], [event_id#8], Inner, BuildRight
   :- *(2) Project [event_id#0, city#1, skew_key#2, date#3]
   :  +- *(2) Filter isnotnull(event_id#0)
   :    +- *(2) FileScan parquet hw2.events_full[event_id#0,city#1,skew_key#2,date#3] Batched: true, Format: Parquet, Location: CatalogFileIndex[hdfs://bigdataanalytics-head-0.novalocal:8020/apps/spark/warehouse/hw2.db/events..., PartitionCount: 8, PartitionFilters: [], PushedFilters: [IsNotNull(event_id)], ReadSchema: struct<event_id:string,city:string,skew_key:string>
```

FINISHED

HW_2 PySpark DataFrame API

```
+-- BroadcastExchange HashedRelationBroadcastMode(List(input[0, string, true]))
+-- *(1) Project [event_id#8]
+-- *(1) Filter isnotnull(event_id#8)
+-- *(1) FileScan parquet hw2.sample[event_id#8] Batched: true, Format: Parquet, Location: InMemoryFileIndex[hdfs://bigdataanalytics-head-0.novalocal:8020/apps/spark/warehouse/hw2.db/sample], PartitionFilters: [], PushedFilters: [IsNotNull(event_id)], ReadSchema: struct<event_id:string>
```

Took 1 sec. Last updated by anonymous at January 29 2021, 11:24:57 PM. (outdated)

```
%pyspark
```

FINISHED

```
events_full.join(sample_big, "event_id").explain()
```

```
== Physical Plan ==
```

```
*(2) Project [event_id#0, city#1, skew_key#2, date#3]
+- *(2) BroadcastHashJoin [event_id#0], [event_id#12], Inner, BuildRight
   :- *(2) Project [event_id#0, city#1, skew_key#2, date#3]
      : +- *(2) Filter isnotnull(event_id#0)
         : +- *(2) FileScan parquet hw2.events_full[event_id#0,city#1,skew_key#2,date#3] Batched: true, Format: Parquet, Location: CatalogFileIndex[hdfs://bigdataanalytics-head-0.novalocal:8020/apps/spark/warehouse/hw2.db/events..., PartitionCount: 8, PartitionFilters: [], PushedFilters: [IsNotNull(event_id)], ReadSchema: struct<event_id:string,city:string,skew_key:string>
      +- BroadcastExchange HashedRelationBroadcastMode(List(input[0, string, true]))
         +- *(1) Project [event_id#12]
            +- *(1) Filter isnotnull(event_id#12)
               +- *(1) FileScan parquet hw2.sample_big[event_id#12] Batched: true, Format: Parquet, Location: InMemoryFileIndex[hdfs://bigdataanalytics-head-0.novalocal:8020/apps/spark/warehouse/hw2.db/sample..., PartitionFilters: [], PushedFilters: [IsNotNull(event_id)], ReadSchema: struct<event_id:string>
```

Took 0 sec. Last updated by anonymous at January 29 2021, 11:25:04 PM. (outdated)

```
%pyspark
```

FINISHED

```
events_full.join(sample_very_big, "event_id").explain()
```

```
== Physical Plan ==
```

```
*(5) Project [event_id#0, city#1, skew_key#2, date#3]
+- *(5) SortMergeJoin [event_id#0], [event_id#14], Inner
   :- *(2) Sort [event_id#0 ASC NULLS FIRST], false, 0
      : +- Exchange hashpartitioning(event_id#0, 200)
         : +- *(1) Project [event_id#0, city#1, skew_key#2, date#3]
            : +- *(1) Filter isnotnull(event_id#0)
               : +- *(1) FileScan parquet hw2.events_full[event_id#0,city#1,skew_key#2,date#3] Batched: true, Format: Parquet, Location: CatalogFileIndex[hdfs://bigdataanalytics-head-0.novalocal:8020/apps/spark/warehouse/hw2.db/events..., PartitionCount: 8, PartitionFilters: [], PushedFilters: [IsNo
```

```
tNull(event_id)], ReadSchema: struct<event_id:string,city:string,skew_key:string>
+- *(4) Sort [event_id#14 ASC NULLS FIRST], false, 0
+- Exchange hashpartitioning(event_id#14, 200)
+- *(3) Project [event_id#14]
```

HW_2 PySpark DataFrame API

```
+- *(3) FileScan parquet hw2.sample_very_big[event_id#14] Batched: true, Format: Parquet, Location: InMemoryFileIndex[hdfs://bigdataan
alytics-head-0.novalocal:8020/apps/spark/warehouse/hw2.db/sampl..., PartitionFilters: [], PushedFilters: [IsNotNull(event_id)], ReadSchema: struct<ev
ent_id:string>
```

Took 0 sec. Last updated by anonymous at January 29 2021, 11:25:09 PM.

```
%md
##### Вывод:
*BroadcastHashJoin* делается для таблиц `sample` и `sample_big`
```

FINISHED

Вывод:

BroadcastHashJoin делается для таблиц `sample` и `sample_big`

Took 0 sec. Last updated by anonymous at January 29 2021, 11:35:30 PM.

Задание 3

FINISHED

Выполнить джойны с таблицами `hw2.sample`, `hw2.sample_big` в отдельных параграфах, чтобы узнать время выполнения запросов (например, вызвать `.count()` для результатов запросов). Время выполнения параграфа считается автоматически и указывается в нижней части по завершении

Зайти в `spark ui` (ссылку сгенерировать в следующем параграфе).

- Сколько **tasks** создано на каждую операцию?
- Почему именно столько?
- Каков DAG вычислений?

Took 0 sec. Last updated by anonymous at January 29 2021, 11:31:21 PM.

```
%pyspark
```

```
events_full \
  .join(sample, "event_id") \
  .count()
```

 SPARK JOB FINISHED

HW_2 PySpark DataFrame API

Took 48 sec. Last updated by anonymous at January 29 2021, 11:57:42 PM.

```
%pyspark
```

```
events_full \
  .join(sample_big, "event_id") \
  .count()
```

 SPARK JOB FINISHED

330052

Took 1 min 26 sec. Last updated by anonymous at January 29 2021, 11:58:24 PM.

Генерация ссылки на spark UI

FINISHED

185.241.193.174:8088/proxy/application_1611766633932_0058/jobs/

Took 1 sec. Last updated by anonymous at January 30 2021, 12:38:47 AM. (outdated)

```
%md
```

```
##### Вывод:
```

- Сколько tasks создано на каждую операцию?
 - 48 seconds join with `sample`. tasks = 61
 - 1 minute join with `sample_big` tasks = 61
- Почему именно столько?
- Каков DAG вычислений?

Вывод:

- Сколько tasks создано на каждую операцию?
 - 48 seconds join with sample . tasks = 61
 - 1 minute join with sample_big tasks = 61
- Почему именно столько?

FINISHED

- Каков DAG вычислений?

Took 0 sec. Last updated by anonymous at February 06 2021, 3:47:29 PM.

HW_2 PySpark DataFrame API

Насильный broadcast

FINISHED

Оптимизировать джойн с таблицами `hw2.sample_big`, `hw2.sample_very_big` с помощью `broadcast(df)`. Выполнить запрос, посмотреть в UI, как поменялся план запроса, DAG, количество задач. Второй запрос не выполнится.

Took 0 sec. Last updated by anonymous at January 27 2021, 11:31:44 PM.

```
%pyspark

from pyspark.sql.functions import broadcast

events_full.ioin(broadcast(sample_big). "event id").explain()

== Physical Plan ==
*(2) Project [event_id#0, city#1, skew_key#2, date#3]
+- *(2) BroadcastHashJoin [event_id#0], [event_id#12], Inner, BuildRight
   :- *(2) Project [event_id#0, city#1, skew_key#2, date#3]
   :  +- *(2) Filter isnotnull(event_id#0)
   :    +- *(2) FileScan parquet hw2.events_full[event_id#0,city#1,skew_key#2,date#3] Batched: true, Format: Parquet, Location: CatalogFileIndex[hdfs://bigdataanalytics-head-0.novalocal:8020/apps/spark/warehouse/hw2.db/events..., PartitionCount: 8, PartitionFilters: [], PushedFilters: [IsNotNull(event_id)], ReadSchema: struct<event_id:string,city:string,skew_key:string>
   +- BroadcastExchange HashedRelationBroadcastMode(List(input[0, string, true]))
      +- *(1) Project [event_id#12]
         +- *(1) Filter isnotnull(event_id#12)
            +- *(1) FileScan parquet hw2.sample_big[event_id#12] Batched: true, Format: Parquet, Location: InMemoryFileIndex[hdfs://bigdataanalytics-head-0.novalocal:8020/apps/spark/warehouse/hw2.db/sample..., PartitionFilters: [], PushedFilters: [IsNotNull(event_id)], ReadSchema: struct<event_id:string>
```

FINISHED

Took 0 sec. Last updated by anonymous at January 30 2021, 12:05:02 AM.

```
%pyspark

from pyspark.sql.functions import broadcast

events_full.ioin(broadcast(sample_big). "event id").count()
```

 SPARK JOB FINISHED

330052

Took 42 sec. Last updated by anonymous at January 30 2021, 12:03:35 AM.

HW_2 PySpark DataFrame API

Отключение auto broadcast

FINISHED


Отключить автоматический бродкаст командой:

```
spark.conf.set("spark.sql.autoBroadcastJoinThreshold", "-1")
```

Сделать джойн с семплом `hw2.sample`, сравнить время выполнения запроса.

Took 0 sec. Last updated by anonymous at January 30 2021, 12:05:50 AM.

%pyspark

 SPARK JOB (<http://bigdataanalytics-head-0.novalocal:4040/jobs/job?id=6>) FINISHED

```
spark.conf.set("spark.sql.autoBroadcastJoinThreshold", "-1")
events full.join(sample, "event id").count()
```

55273

Took 2 min 8 sec. Last updated by anonymous at January 30 2021, 12:08:55 AM.

%md

FINISHED

Вывод:

- при отключении автоматического **broadcast** время выполнения:

time: 48seconds -> 2 min

tasks: 61 -> 263

Вывод:

- при отключении автоматического *broadcast* время выполнения:
time: 48seconds -> 2 min
tasks: 61 -> 263

Took 0 sec. Last updated by anonymous at January 30 2021, 12:10:56 AM.

Вернуть настройку к исходной

FINISHED

```
spark.conf.set("spark.sql.autoBroadcastJoinThreshold", "26214400")
```

HW_2 PySpark DataFrame API

```
spark.sql("clear cache")
```

FINISHED

```
res3: org.apache.spark.sql.DataFrame = []
```

Took 1 sec. Last updated by anonymous at January 30 2021, 12:11:16 AM.


Задание 4

FINISHED

В процессе обработки данных может возникнуть перекос объёма партиций по количеству данных (**data skew**). В таком случае время выполнения запроса может существенно увеличиться, так как данные распределяются по исполнителям неравномерно. В следующем параграфе происходит инициализация датафрейма, этот параграф нужно выполнить, изменять код нельзя. В задании нужно работать с инициализированным датафреймом.

Датафрейм разделен на 30 партиций по ключу **city**, который имеет сильно неравномерное распределение.

Took 0 sec. Last updated by anonymous at January 30 2021, 12:11:24 AM.

нужно выполнить, изменять код нельзя SPARK JOB (<http://bigdataanalytics-head-0.novalocal:4041/jobs/job?id=0>) FINISHED

```
%pyspark
from pyspark.sql.functions import col

skew_df = spark.table("hw2.events_full")\
    .where("date = '2020-11-01'")\
    .repartition(30, col("city"))\
    .cache()

skew_df.count()
```

Hive Session ID = a1af16f2-a650-4671-9637-1cca84b43165
20000000

Took 1 min 47 sec. Last updated by anonymous at February 11 2021, 1:05:48 PM.

4.1 Наблюдение проблем

HW_2 PySpark DataFrame API

FINISHED

Посчитать количество `event_count` различных событий `event_id`, содержащихся в `skew_df` с группировкой по городам. Результат упорядочить по `event_count`.

В spark ui в разделе jobs выбрать последнюю, в ней зайти в stage, состоящую из 30 задач (из такого количества партиций состоит `skew_df`). На странице стейджа нажать кнопку *Event Timeline* и увидеть время выполнения задач по экзеkjюторам. Одному из них выпала партиция с существенно большим количеством данных. Остальные экзеkjюторы в это время бездействуют – это и является проблемой, которую предлагается решить далее.

Took 0 sec. Last updated by anonymous at January 30 2021, 12:12:24 AM.

%pyspark

 SPARK JOB (<http://bigdataanalytics-head-0.novalocal:4041/jobs/job?id=8>) FINISHED

```
skew_df\  
  .groupBy("city")\  
  .count()\  
  .orderBy(col("count"))\  
  .show()
```


4.2 repartition

FINISHED

Один из способов решения проблемы агрегации по неравномерно распределенному ключу является предварительное перемешивание данных. Его можно сделать с помощью метода `repartition(p_num)`, где `p_num` – количество партиций, на которые будет перемешан исходный датафрейм.

Took 0 sec. Last updated by anonymous at January 27 2021, 11:40:08 PM.

%pyspark

 SPARK JOB (<http://bigdataanalytics-head-0.novalocal:4041/jobs/job?id=10>) FINISHED

```
skew_df\  
  .repartition(5)
```

```
.groupBy("city")\
  .count()\
  .orderBy(col("count"))\
  .show(5)
```

HW_2 PySpark DataFrame API

```
+-----+-----+
|SMALL_CITY_100| 9917|
|  SMALL_CITY_0| 9964|
|  SMALL_CITY_54|19683|
|  SMALL_CITY_39|19695|
|  SMALL_CITY_45|19704|
+-----+-----+
only showing top 5 rows
```

Took 17 sec. Last updated by anonymous at February 11 2021, 1:15:10 PM.

4.3 Key Salting

FINISHED

Другой способ исправить неравномерность по ключу – создание синтетического ключа с равномерным распределением. В нашем случае неравномерность исходит от единственного значения `city='BIG_CITY'`, которое часто повторяется в данных и при группировке попадает к одному экзекьютору. В таком случае лучше провести группировку в два этапа по синтетическому ключу `CITY_SALT`, который принимает значение `BIG_CITY_rand` (`rand` – случайное целое число) для популярного значения `BIG_CITY` и `CITY` для остальных значений. На втором этапе восстанавливаем значения `CITY` и проводим повторную агрегацию, которая не занимает времени, потому что проводится по существенно меньшего размера данным.

Такая же техника применима и к джойнам по неравномерному ключу, см., например <https://itnext.io/handling-data-skew-in-apache-spark-9f56343e58e8> (<https://itnext.io/handling-data-skew-in-apache-spark-9f56343e58e8>)

Что нужно реализовать:

- добавить синтетический ключ
- группировка по синтетическому ключу
- восстановление исходного значения

- группировка по исходной колонке

Took 0 sec. Last updated by anonymous at January 27 2021, 11:41:18 PM. (outdated)

HW_2 PySpark DataFrame API

%pyspark  SPARK .IQR (http://bigdataanalytics-head-0 novalocal:4041/jobs/job?id=11) FINISHED

```
from pyspark.sql.functions import when, rand, round, expr, lit, sum, countDistinct
```

```
salt_size = 10
```

```
skew_df\
    .withColumn('salt_index', round(100 * rand()))\
    .withColumn('city_salt', when(col('city') == 'BIG_CITY', expr('CONCAT(city, salt_index)')).otherwise(col('city')))\
    .groupBy('city_salt').agg(countDistinct('event_id').alias('count'))\
    .withColumn('city', when(expr('city_salt not like "SMALL%"'), lit('BIG_CITY')).otherwise(col('city_salt')))\
    .groupBy('city')\
    .agg(sum('count').alias('count'))\
    .orderBy(col('count'), ascending=False)\
    , '\n'
```

```
+-----+-----+
|      city|   count|
+-----+-----+
|   BIG_CITY|17999446|
|SMALL_CITY_71|   20477|
|SMALL_CITY_61|   20278|
|SMALL_CITY_90|   20261|
|SMALL_CITY_76|   20261|
|SMALL_CITY_99|   20247|
|SMALL_CITY_87|   20223|
|SMALL_CITY_96|   20211|
|SMALL_CITY_25|   20205|
|SMALL_CITY_86|   20193|
|SMALL_CITY_36|   20173|
|SMALL_CITY_67|   20167|
|SMALL_CITY_28|   20165|
|SMALL_CITY_44|   20164|
|SMALL_CITY_22|   20163|
```

Took 3 min 0 sec. Last updated by anonymous at February 11 2021, 1:22:37 PM.

spark.stop

READY

