ApacheSparkThroughEmail2

November 27, 2018

1 Apache Spark Through Email - where, Column methods, functions

```
In [1]: val records = spark.read.parquet("/datasets/enron/enron-small.parquet")
        records.cache
        records.count
```

```
Waiting for a Spark session to start...


records = [uuid: string, from: string ... 8 more fields]
```

```
Out[1]: 191926
```

1.1 Finding old records - where clause

```
In [2]: import java.time._

        val oldCutoff = ZonedDateTime.of(1980, 1, 1, 0, 0, 0, 0, ZoneId.of("UTC"))
        val oldCutoffInMillis = oldCutoff.toInstant.toEpochMilli
```

```
oldCutoff = 1980-01-01T00:00Z[UTC]
oldCutoffInMillis = 315532800000
```

```
Out[2]: 315532800000
```

```
In [3]: // https://spark.apache.org/docs/latest/api/scala/index.html#org.apache.spark.sql.func
        // https://spark.apache.org/docs/latest/api/scala/index.html#org.apache.spark.sql.Colu
        import org.apache.spark.sql.functions.from_unixtime

        // Column "dateUtcEpoch" - also $"dateUtcEpoch"
        val oldRecords = records.where(records("dateUtcEpoch") < oldCutoffInMillis).
            select("uuid","dateUtcEpoch").
            withColumn("dateString", from_unixtime($"dateUtcEpoch" / 1000)).
            orderBy("dateUtcEpoch").
```

```
            select("uuid","dateString")

        oldRecords.printSchema
```

```
root
 |-- uuid: string (nullable = true)
 |-- dateString: string (nullable = true)
```

```
oldRecords = [uuid: string, dateString: string]
```

```
Out[3]: [uuid: string, dateString: string]
```

```
In [4]: println(oldRecords.count)

        oldRecords.show(numRows = 10, truncate=false)
```

```
82
+------------------------------------+-------------------+
|uuid                                |dateString         |
+------------------------------------+-------------------+
|cc8c5d69-1687-4b32-9574-ae377bfad731|0001-08-13 08:06:06|
|886a9df5-13b3-4538-b70e-a30b1b1c1493|0001-08-20 08:11:21|
|cb3a8063-6e65-497d-aa72-6f2d26f96705|0001-08-22 08:05:53|
|cfc06550-9038-42ec-9bfd-ab6a74809350|0001-08-28 17:20:02|
|cfb7394d-69b5-414e-84bb-145909446884|0001-08-29 08:07:37|
|24b84513-2b3c-463f-b4e1-9da26db23434|0001-08-30 17:20:19|
|fdda2ad9-7a6e-4814-872d-3ac06b2a0bc2|0001-09-04 08:07:45|
|67e5207f-3fab-4f00-9dc2-7d3d1bc2b216|0001-09-04 08:08:57|
|4d041254-577b-4c90-afa6-dd16a41110b7|0001-09-04 18:30:08|
|6cceee88-d17c-4906-9aa0-c62b494d5d27|0001-09-05 08:05:39|
+------------------------------------+-------------------+
only showing top 10 rows
```

1.2 Grouping records by year - withColumn, Column / and cast

```
In [5]: import org.apache.spark.sql.functions._
        import org.apache.spark.sql.types.TimestampType
        val yearsWithCounts = records.withColumn("dateTs", ($"dateUtcEpoch" / 1000).cast(Timest
           withColumn("year", year($"dateTs")).
           groupBy("year").count.
           orderBy("year").
           collect
```

```
yearsWithCounts = Array([1,71], [2,11], [1980,276], [1997,436], [1998,3], [1999,2998], [2000,8
```

```
Out[5]: Array([1,71], [2,11], [1980,276], [1997,436], [1998,3], [1999,2998], [2000,80375], [20
```

```
In [6]: %AddDeps org.vegas-viz vegas_2.11 0.3.11 --transitive
```

```
Marking org.vegas-viz:vegas_2.11:0.3.11 for download
Obtained 42 files
```

```
In [7]: import vegas._
        import vegas.render.WindowRenderer._

        val yearsWithCountMaps = yearsWithCounts.map {r =>
            val year = r.getInt(0)
            val count = r.getLong(1)
            Map("year" -> year, "count" -> count)
        }
        val plot = Vegas("Year Distribution").
          withData(
            yearsWithCountMaps
          ).
          encodeX("year", Nom).
          encodeY("count", Quant).
          mark(Bar)

        plot.show
```

```
yearsWithCountMaps = Array(Map(year -> 1, count -> 71), Map(year -> 2, count -> 11), Map(year -
plot = ExtendedUnitSpecBuilder(ExtendedUnitSpec(None,None,Bar,Some(Encoding(None,None,Some(Posi
```

```
Out[7]: ExtendedUnitSpecBuilder(ExtendedUnitSpec(None,None,Bar,Some(Encoding(None,None,Some(Pos
```

```
In [8]: spark.close
```