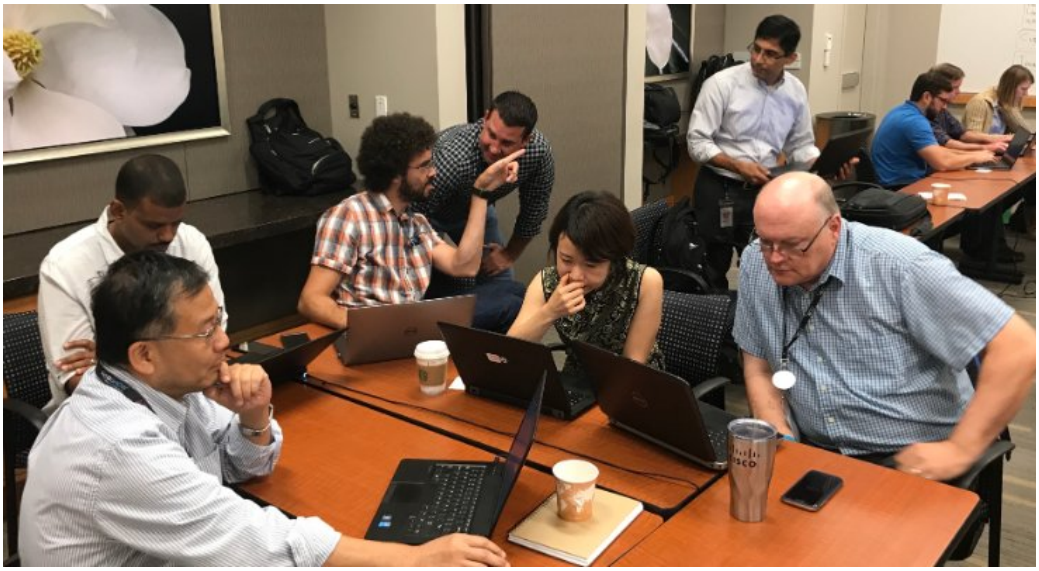

On-site Spark Training in Georgia

Simple Apache Spark PID masking with DataFrame, SQLContext, regexp_replace, Hive, and Oracle.

Published on August 21, 2017 Edit article | View stats

Laurent Weichberger
Changing the world one Big Data client at a time ....
31 articles

196 19 0

Recently I taught our standard Apache Spark training at an on-site client. Rapidly they realized the power of Spark and asked me if we could please substitute a regular (canned) lab-exercise with one of their own use cases which urgently needed to be solved.

I said -- "Sure, which one?"

Without getting into the details of their use case, let me just say they wanted to use Spark to read in production data, mask PID ("data elements, which identify a unique individual"), and write that masked data back out to a new Apache Hive table, and also back out to an Oracle table, all from a Spark DataFrame. I said, let's do it as a project on the last day. We all came to an agreement about the use case on Wednesday before the end of class, this left us one day (Thursday) to solve it.

Messaging

We started our work at 9:00am. I had them decompose the use case into user stories (ala Agile-Scrum), and we created 30 minute Sprints. Everyone was working on a story, either alone or in a team. It was amazingly productive. We had a one hour lunch break and then solved this and tested it by 2:00pm.

1. We moved a demo version of the production code into safe location for the project, and loaded it into a DataFrame for starters:

```
val dataframeA = sqlContext.sql("select * from data_worker.demo")
```

2.a. We used the withColumn() method to add a column to the DataFrame to hold the masked version of the data, and named it "text_masked." The value of the new column is the old column "text" with the regexp_replace function applied to it.

```
val dataframeB = dataframeA
  .withColumn("text_masked",
     regexp_replace(dataframeA("text"), "\\d{2,}", " * ")) //See note 1 b
       .drop("text")
```

2.b. The regexp_replace takes three parameters, the column you wish to transform, the pattern to apply, and the new value for that which is found using the pattern. Here we are only concerned with numbers, so we say in the text column, whenever we find two or more numbers consecutively, replace that collection of numbers with an asterisk surrounded by white space on each side. Note we drop() the old column "text" so that we don't continue to propagate the PID data around the systems.

So, a SSN value of 789-01-2345 would become * - * - * and a telephone number (212) 345-6789 would become (*) * - *. This was good enough for our PoC lab exercise solution.

3.a. To write out to Hive with this data, we used a simple DataFrame method registerTempTable followed by the SQLContext "CREATE TABLE" invocation:

```
dataframeB.registerTempTable("demoTemp")
sqlContext.sql("CREATE TABLE demoHive AS SELECT * FROM demoTemp")
```

3.b. To write out to Oracle we had a more complex invocation from DataFrame:

```
import java.util.Properties
import java.io._

val prop = new Properties()

prop.setProperty("user", “user”)
prop.setProperty("password", “password”)
prop.setProperty("driver", "oracle.jdbc.driver.OracleDriver")

dataframeB.write.mode(org.apache.spark.sql.SaveMode.Append)
  .jdbc("jdbc:oracle:thin:@(DESCRIPTION=
     (ADDRESS=(PROTOCOL=TCP)(HOST=<your-host>)(PORT=<your-port))
     (CONNECT_DATA=(SERVER=DEDICATED)))",
     "call.<oracle-table-name>", prop)
```

Notes: 1. For regex_replace See:

Enjoy!

For more information: Laurent Weichberger, Big Data Bear, Hortonworks :: lweichberger@hortonworks.com

Laurent Weichberger
Changing the world one Big Data client at a time ....

More from Laurent Weichberger See all 31 articles

Summarization Optimization with HDP Apache Spark & Avro.
Laurent Weichberger on LinkedIn

Store Level Forecasting with Apache Spark Machine Learning
Laurent Weichberger on LinkedIn

Use Case Discovery workshop :: GeoLocation Engine with Apache P…
Laurent Weichberger on LinkedIn