

Feature Building with Spark ML

A Hortonworks University

Hadoop Training Course

Title: LAB GUIDE: Data Science for the Hortonworks Data Platform

Revision 2

Copyright **©** 2015 Hortonworks Inc 2015 All rights reserved.

All Rights Reserved. Hadoop and the Hadoop elephant logo are trademarks of the Apache Software Foundation.

The contents of this course and all its related materials, including lab exercises and files, are Copyright **©** 2015 Hortonworks Inc.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means electronic, mechanical, photocopying, recording or otherwise without the prior written permission of Hortonworks Inc.

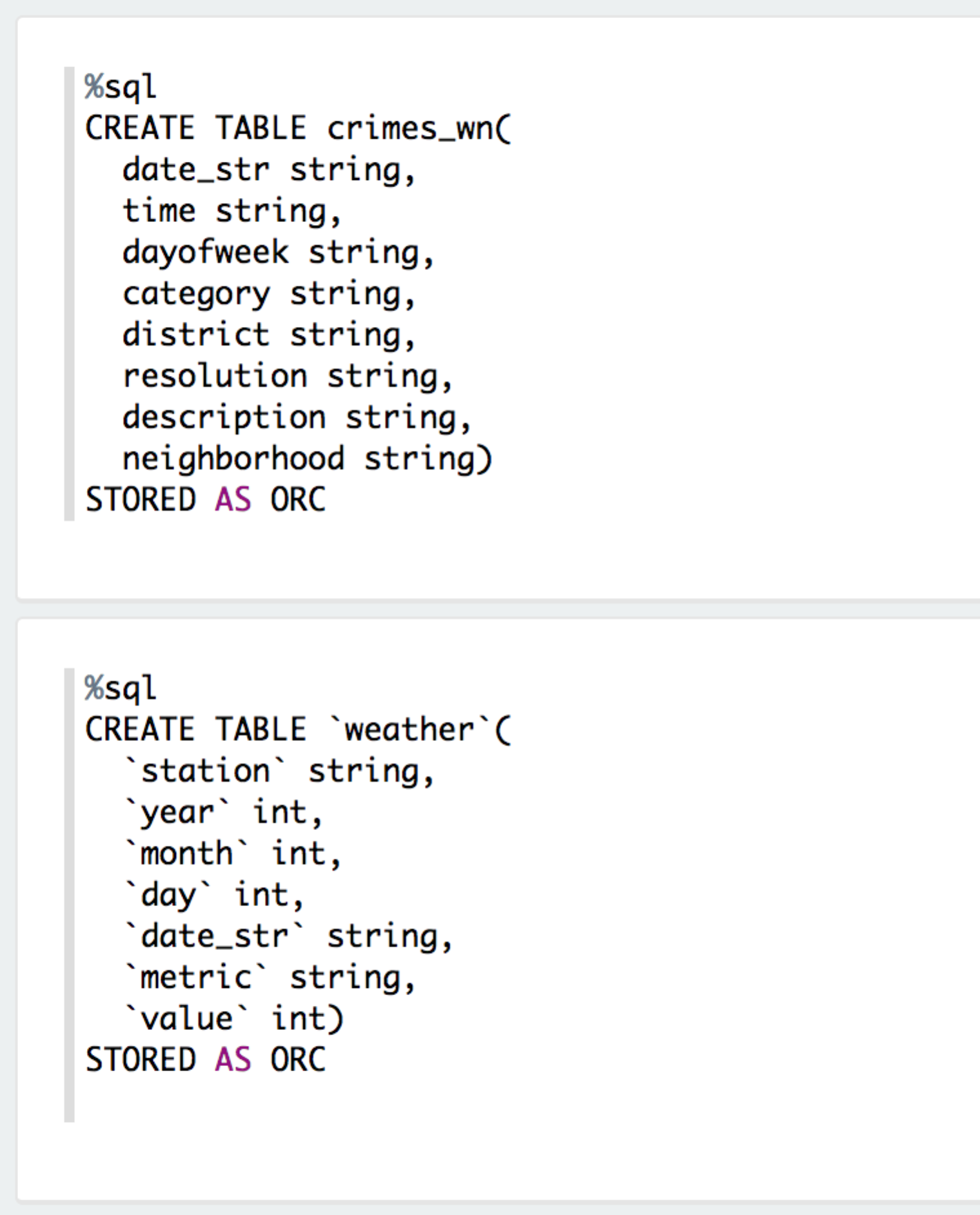
Lab: Feature Building with Spark ML

|  |  |
| --- | --- |
| **Objective:** | Become familiar with using Spark MLlib to run data science algorithms on a Hadoop cluster. |
| **Successful Outcome:** | You will have created a spam classifier with MLlib. |
| **Before You Begin:** | Your HDP cluster should be up and running in the classroom VM. |

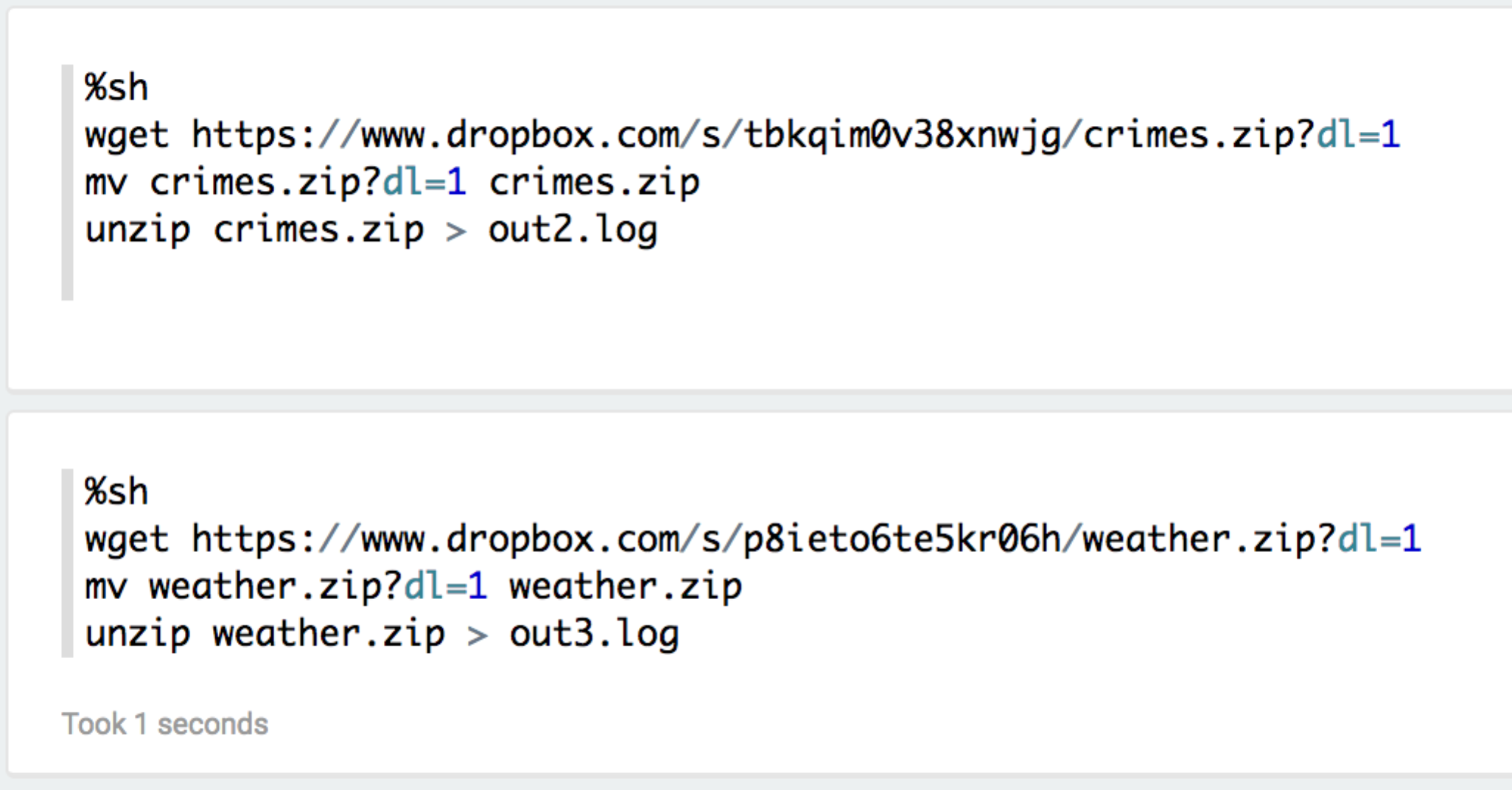
* 1. Before we can begin this lab, we need to retrieve the data and put it in the HDFS. Run the following commands below the screenshot to retrieve the data, unzip the file, and put the data into the hdfs. We will also be building hive tables over this data.

%sql  
CREATE TABLE crimes\_wn(  
 date\_str string,  
 time string,  
 dayofweek string,  
 category string,  
 district string,  
 resolution string,  
 description string,  
 neighborhood string)  
STORED AS ORC

%sql  
CREATE TABLE `weather`(  
`station` string,  
`year` int,  
`month` int,  
`day` int,  
`date\_str` string,  
`metric` string,  
`value` int)  
STORED AS ORC



%sh  
wget <https://www.dropbox.com/s/tbkqim0v38xnwjg/crimes.zip?dl=1>  
mv crimes.zip?dl=1 crimes.zip  
unzip crimes.zip > out2.log  
wget <https://www.dropbox.com/s/p8ieto6te5kr06h/weather.zip?dl=1>  
mv weather.zip?dl=1 weather.zip  
unzip weather.zip > out3.log



%sql  
load data local inpath '/home/zeppelin/crimes\_wn/\*' overwrite into table crimes\_wn  
  
%sql  
load data local inpath '/home/zeppelin/weather/\*' overwrite into table weather



* 1. Assuming we did everything correctly, go ahead and verify all the data is loaded into our hive tables. The following queries should return results

%sql  
select \* from crimes\_wn limit 10

%sql  
select \* from weather limit 10

1. Now that the data is loaded, lets start manipulating it and building our features.
   1. Firse we want to grab all the data between 2011 and 2014. The date string contains the year, so we need to extract that.

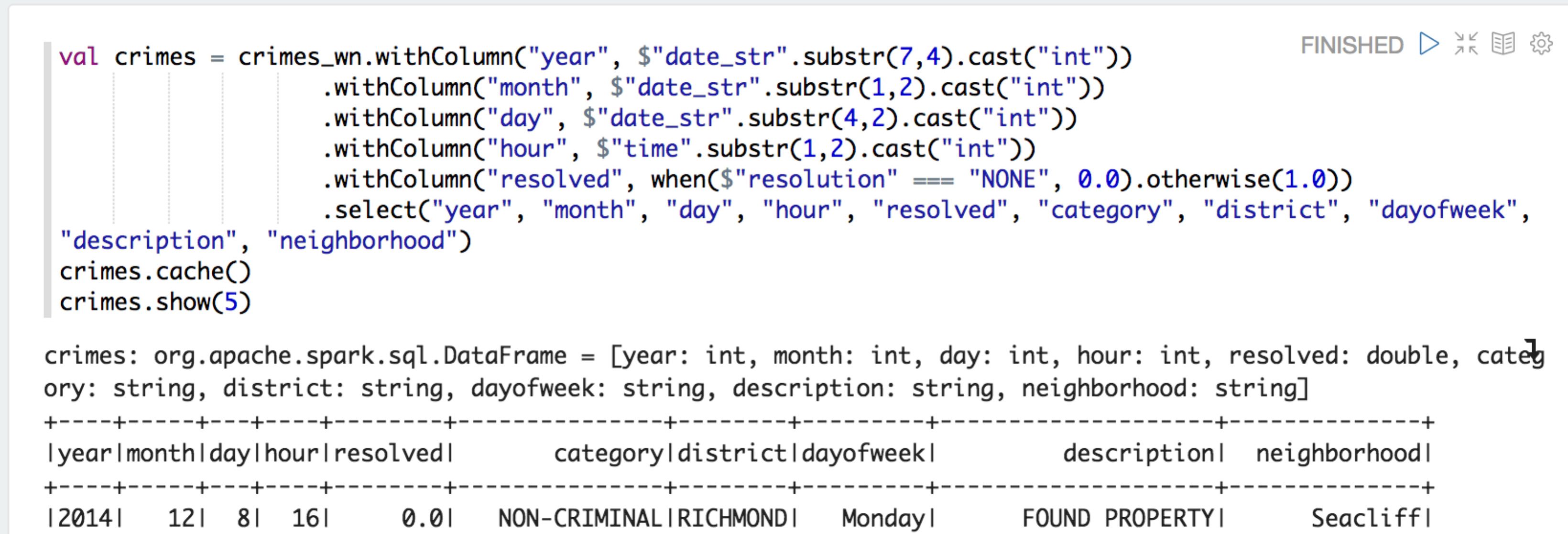
val crimes\_wn\_all = sqlContext.table("crimes\_wn")

val crimes\_wn = crimes\_wn\_all.filter($"date\_str".substr(7,4) >= "2011").filter($"date\_str".substr(7,4) <= "2014")

* 1. Next we want to exract the year, month and day from the date, hour from the time, and we want to create a “resovled” or not label. In addition, we want to use category, district, dave of week, description and neighborhood from the original dataset.

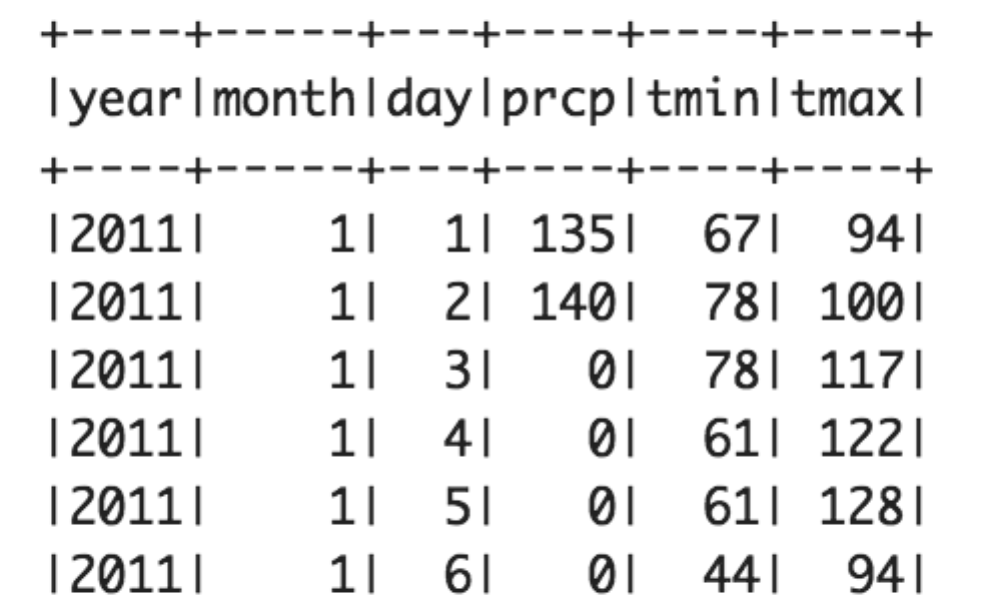
val crimes = crimes\_wn.withColumn("year",  
$"date\_str".substr(7,4).cast("int")) .withColumn("month", $"date\_str".substr(1,2).cast("int"))  
.withColumn("day", $"date\_str".substr(4,2).cast("int"))  
.withColumn("hour", $"time".substr(1,2).cast("int"))  
.withColumn("resolved", when($"resolution" === "NONE", 0.0).otherwise(1.0))  
.select("year", "month", "day", "hour", "resolved", "category", "district", "dayofweek", "description", "neighborhood")

crimes.cache()



* 1. We have a hunch that the weather is going to play a roll in the crime dataset. Lets go ahead get the relavent data for the bay area between the year 2011 and 2014. We’re going to look at precipitation, max and min temperatures.

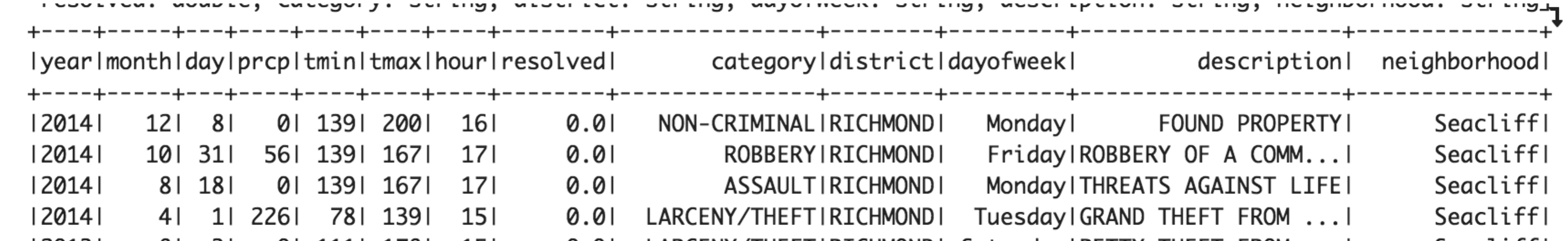
val weather = sqlContext.sql("select \* from weather WHERE year>=2011 and year<=2014 and station == 'USW00023272'").cache()  
val prcp = weather.filter($"metric"==="PRCP").withColumnRenamed("value", "prcp").alias("prcp")  
val tmin weather.filter($"metric"==="TMIN").withColumnRenamed("value", "tmin").alias("tmin")  
val tmax = weather.filter($"metric"==="TMAX").withColumnRenamed("value", "tmax").alias("tmax")  
val wdata = prcp.join(tmin, "date\_str").join(tmax, "date\_str").select(prcp("year"), prcp("month"), prcp("day"), prcp("prcp"), tmin("tmin"), tmax("tmax")).cache()  
wdata.show()



* 1. Now that we have our weather dataset, in addition to our crime dataset, lets combine them, and do a little bit of clean up.

val joined = wdata.join(crimes, wdata("month") === crimes("month") && wdata("day")===crimes("day") && wdata("year")===crimes("year"), "inner").drop(crimes("day")).drop(crimes("month")).drop(crimes("year"))

joined.show()



* 1. Save the dataframe to the hdfs in an ORC file format.

