Data Exploration in Spark

By the end of this activity, you will be able to perform the following in Spark:

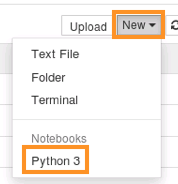
1. Read CSV files into Spark Dataframes.
2. Generate summary statistics.
3. Compute correlation coefficients between two columns.

In this activity, you will be programming in a Jupyter Python Notebook. If you have not already started the Jupyter Notebook server, see the instructions in the Reading *Instructions for Starting Jupyter*.

Step 1. **Create new Jupyter Python Notebook.** Open a web browser by clicking on the web browser icon at the top of the toolbar:

https://d3c33hcgiwev3.cloudfront.net/imageAssetProxy.v1/RCneZE7PEeaqTxIkdCEfsw_c491f272226b35805e44abef7a7a22a9_browser-icon.png?expiry=1598227200000&hmac=LtnkhyYkm1XmpzW-822mjRu0RFGYMPCwaLLXaC9joZQ

Create a new Python Notebook by clicking on *New,*and then click on *Python 3:*



Step 2. **Load data into Spark DataFrame.** First, we need to import the *SQLContext* class:

https://d3c33hcgiwev3.cloudfront.net/imageAssetProxy.v1/GfWWzIp5EeaCXQ5dBCgoUw_c6ca32ef91275765697b255eb4b2bf14_import.png?expiry=1598227200000&hmac=AYRo-DhBROtUbq4GcV9c8s4IF5JE6mbaMmMG4VVS324

Next, we create an SQLContext:

https://d3c33hcgiwev3.cloudfront.net/imageAssetProxy.v1/I-O0m4p5Eea63BLi-G7oTw_a1316d798446af158d77a212e35d8bbe_sqlcontext.png?expiry=1598227200000&hmac=33HZ5USoaSxTC3lVAtCHc5RA9GMrVmNipfxLB16EeOc

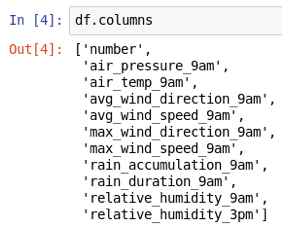
Then we read the weather data into a DataFrame:



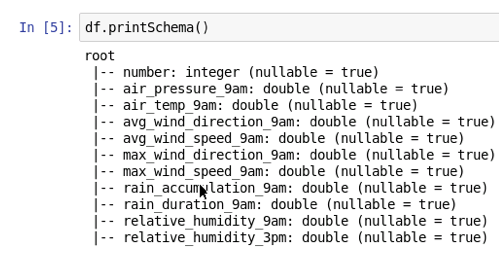
The first argument specifies the URL to the *daily\_weather.csv* file, the second argument specifies the *spark-csv* format, the third argument says the first line in *daily\_weather.csv* is the header, and the fourth argument says to infer the data types.

We use [*spark-csv*](https://github.com/databricks/spark-csv)to read CSV data directly into a Spark DataFrame. In Spark 1.x (the version on the Cloudera VM), *spark-csv* is an external package, but *spark-csv* is integrated with Spark 2.x.

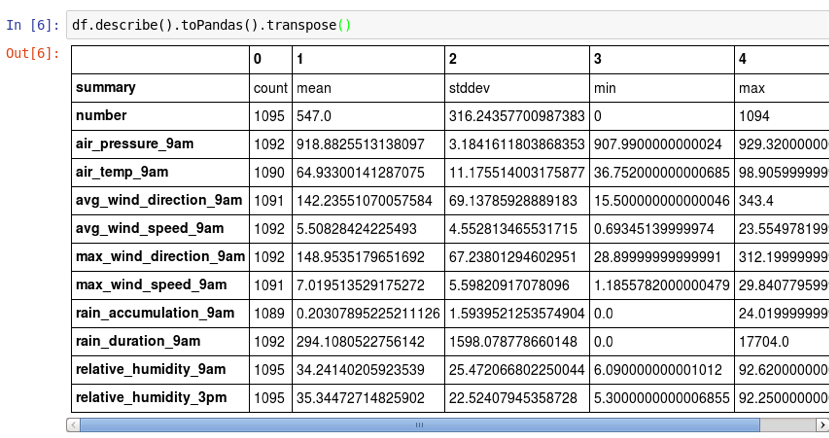
Step 4. **Look at data columns and types.** We can see the columns in the DataFrame by looking at the *columns* attribute:



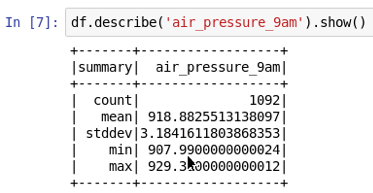
The data type for each column by calling *printSchema()*:



Step 5. **Print summary statistics**. We can print the summary statistics for all the columns using the *describe()*method:



We can also see the summary statistics for just one column:



Let's count the number of columns and rows in the DataFrame:



https://d3c33hcgiwev3.cloudfront.net/imageAssetProxy.v1/e8OVfop5EeaTPwpQDCu1cw_8bb4e6f495390d9f3796b3d6beeb0625_rows.png?expiry=1598227200000&hmac=JevW8K99Mp7CGn3PpS63zyaKjdSLX5NQHQlFim7Jfe8

The number of rows in the DataFrame is 1095, but the summary statistics for *air\_pressure\_9am* says there are only 1092 rows. These are different since 1095 - 1092 = 3 rows have missing values.

Step 6. **Drop rows with missing values.** Let's drop the rows with missing values in the *air\_pressure\_9am* column:

https://d3c33hcgiwev3.cloudfront.net/imageAssetProxy.v1/rCSIUop5EeaT1Q4Eo0QvOw_3e659a4f5ff525eb942f8a7cf03c9ab8_drop-missing.png?expiry=1598227200000&hmac=odvqoXmjTgp4x9TTza_VLzobMdaXLOH4q69V8Xyzehg

Now let's see the total number of rows:

https://d3c33hcgiwev3.cloudfront.net/imageAssetProxy.v1/tPjR2Ip5EeaDbw65org3Dw_442eea2c447b0358e4ee3a34ed3568fb_dropped-row-count.png?expiry=1598227200000&hmac=tdFfYce7tkkof1r29HxGth5wueE_t9UCtyrj3pgFCkY

The total number of rows and number of rows in the summary statistics are now the same.

Step 7. **Compute correlation between two columns.** We can compute the correlation between two columns in a DataFrame by using the *corr()* method. Let's compute the correlation between *rain\_accumulation\_9am* and *rain\_duration\_9am*: