Assignment 4 - More Fun with Spark

No environment setup this week! Use the Spark shell (or notebook if you have set it up) to do this weeks assignment in either Scala or Python. Unless specified otherwise, please use the DataFrame API for this week's work; do not just use SQL string queries.

If you are unable to come up with code to answer a question please describe how you think you would solve the problem in a "sparkified" way based on what we have learned so far.

1) From home_data.csv, how many houses sold were built prior to 1979? (I promise that is the last time I'm going to ask this question)
2) From home_data.csv, what is the most expensive zipcode in the data set, defined as highest average sales price?
3) How many unique zipcodes have sales data in the home_data.csv data set?
4) Demonstrate how to drop the "sqft_living15" and "sqft_lot15" columns from your dataset.
5) Access the zipcode table stored in Hive (ok to use a SQL string query here) and demonstrate joining that data with a dataframe created from home_data.csv.

Bonus Exercise

Remember, bonus exercise is meant for those either with prior knowledge on the topic or the interest (and time) to do some research beyond what we covered in class and is optional.

Exercise 1:

You can register a user-defined function (UDF) in your code that can then be used like any built-in Spark functions.

In the airport_codes.csv file used in class this week, the data contains a field "iso_region" that has a syntax like this for rows in the US: "US-StateCode". So for example Seattle is "US-WA". Write a UDF to return the state from this column.

Exercise 2:

It is possible to programmatically apply a schema to a DataFrame, either if you don't have a schema in advance, or maybe you want to provide a different projection of the data to create multiple views of a base data set, or most commonly when going from a RDD to a DataFrame. Create a simple RDD, and then programmatically apply a schema to it to create a DataFrame.