We'll begin by starting an EMR cluster as we did last week. However, this time, instead of choosing "Core Hadoop" for our "Applications", we will pick "Spark". In order to take advantage of the Spark REPL, we will be using [Apache Zeppelin](https://zeppelin.apache.org/).

Now that we are using Spark, it is more important that our data be able to fit in memory. Make sure that that is the case by running aws s3 ls --summarize --human-readable --recursive on your S3 bucket to see how much data you've accumulated so far. The default EMR cluster consists of just 2 m3.xlarge core instances (not counting the master node) with a combined memory of just 30 GiB. You might wish to upgrade to m3.2xlarge or one of the m4 instances and/or create a larger cluster.

Once your cluster is launched and waiting, put the URL of the master node into your favorite web browser followed by :8890 to specify port 8890, the default port for Apache Zeppelin on EMR. (For example, [http://ec2-XXX-XXX-XXX-XXX.compute-1.amazonaws.com:8890/](http://ec2-xxx-xxx-xxx-xxx.compute-1.amazonaws.com:8890/).) Hint: Make sure that the security group for your master node allows connections on that port.

Click "Create new note". This will create a Zeppelin notebook, which is much like a Jupyter notebook. However, be aware that the default language is Scala. If you want to use PySpark, you will have to use the %pyspark magic in each cell.

It's worth noting that your SparkContext (*i.e.* sc) is automatically given to you when you create a Zepplein notebook. (In other words do not try to [instantiate a new SparkContext](http://stackoverflow.com/questions/23280629/multiple-sparkcontexts-error-in-tutorial).)

For our first Spark lab, we will reproduce what we did using Hadoop, only now using Spark. That is to say, we will be using the map/reduce paradigm on Spark.

Hint: remember that Spark uses HDFS and Amazon EMR is able to map S3 onto HDFS as if it were part of the file system. So you do not need to use boto or boto3 for this lab at all. You just need to specify the filesystem as s3a://. See [S3 Support in Apache Hadoop](https://wiki.apache.org/hadoop/AmazonS3) and [EMR File Systems.](https://docs.aws.amazon.com/emr/latest/ManagementGuide/emr-plan-file-systems.html) for more details.

**Submission:**Upload your Zepplin notebook.