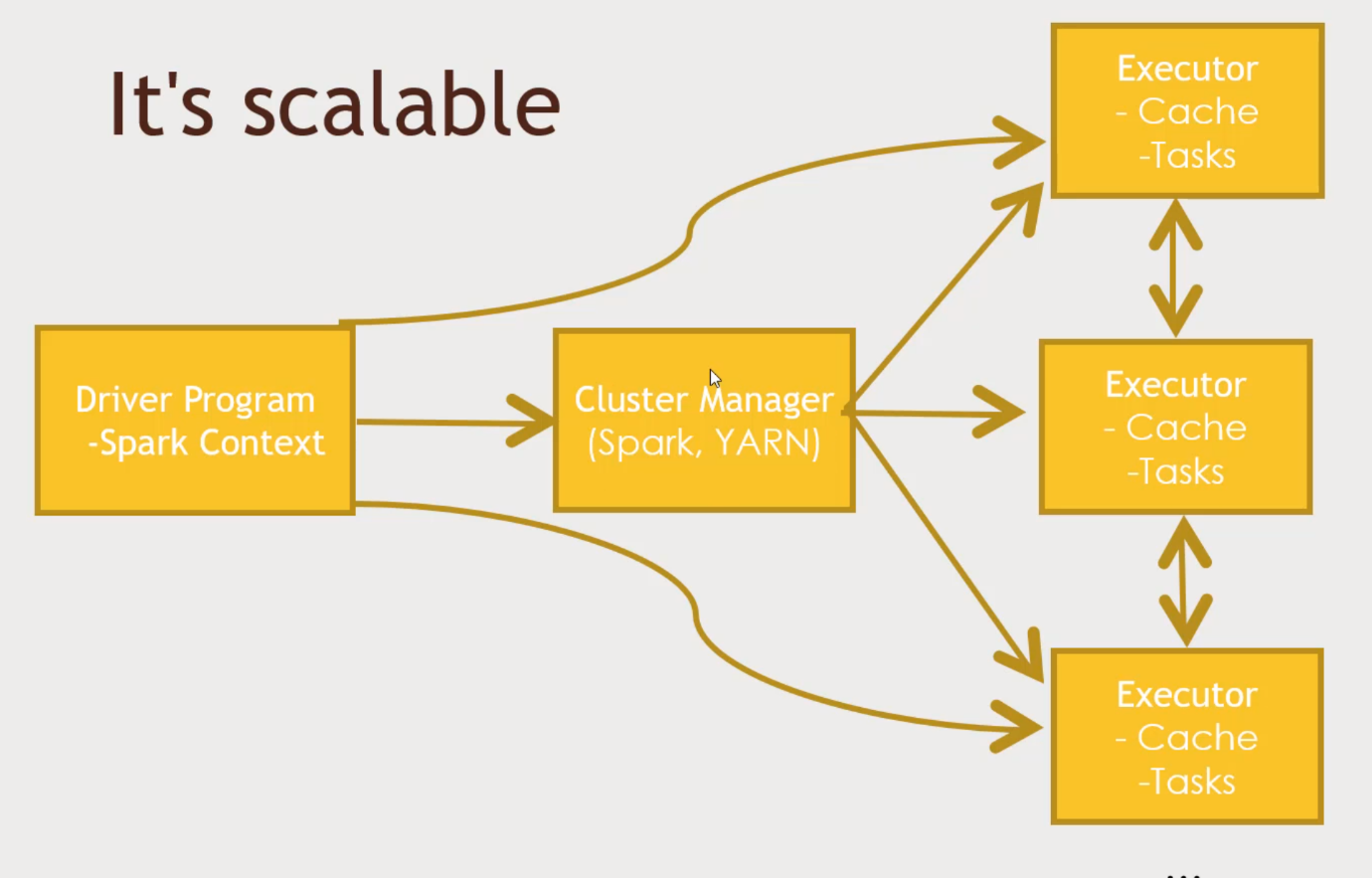
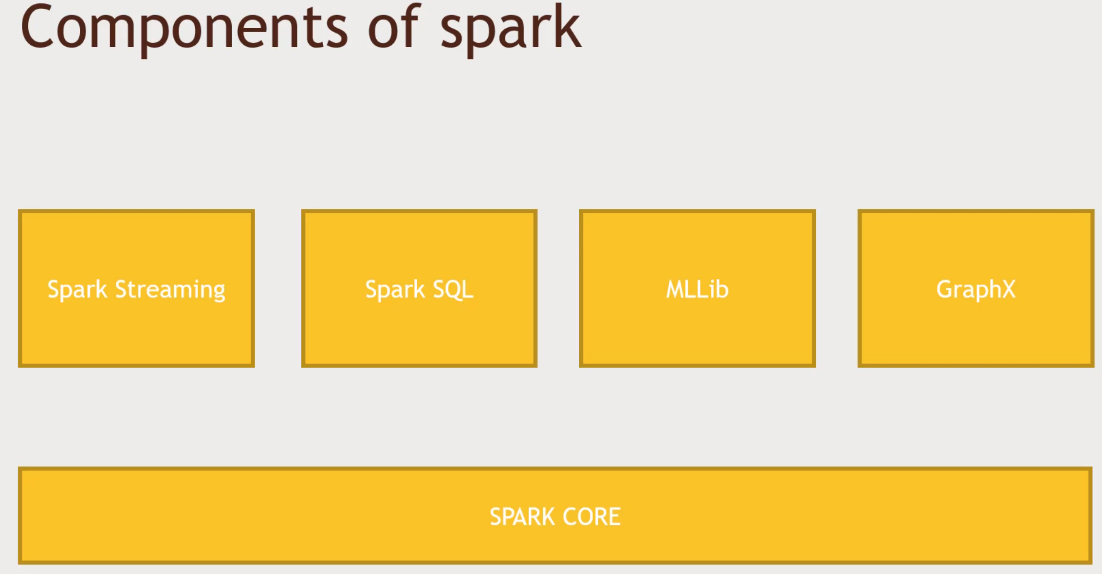
Spark –

* A fast and general engine for large-scale data processing
* A lot of flexibility, Real script programming languages, Python, Java, Scala
* Rich Ecosystem on top of Spark: Machine Learning, Data Mining, Streaming, Graph Analysis
* Spark is a Memory based solution
* Directed Acyclic Graphs

Run programs up to 100x faster than Hadoop MapReduce in memory, or 10x faster on Disk

DAG Engine (directed acyclic graph) optimizes workflows.

Tons of ppl are using Spark.

* Code in Python, Java or Scala
* Built around one main concept: the Resilient Distributed Dataset (RDD) – various functions inside RDD. To produce new RDD. Takes RDD and transform it

Spark Streaming – Real time Web.

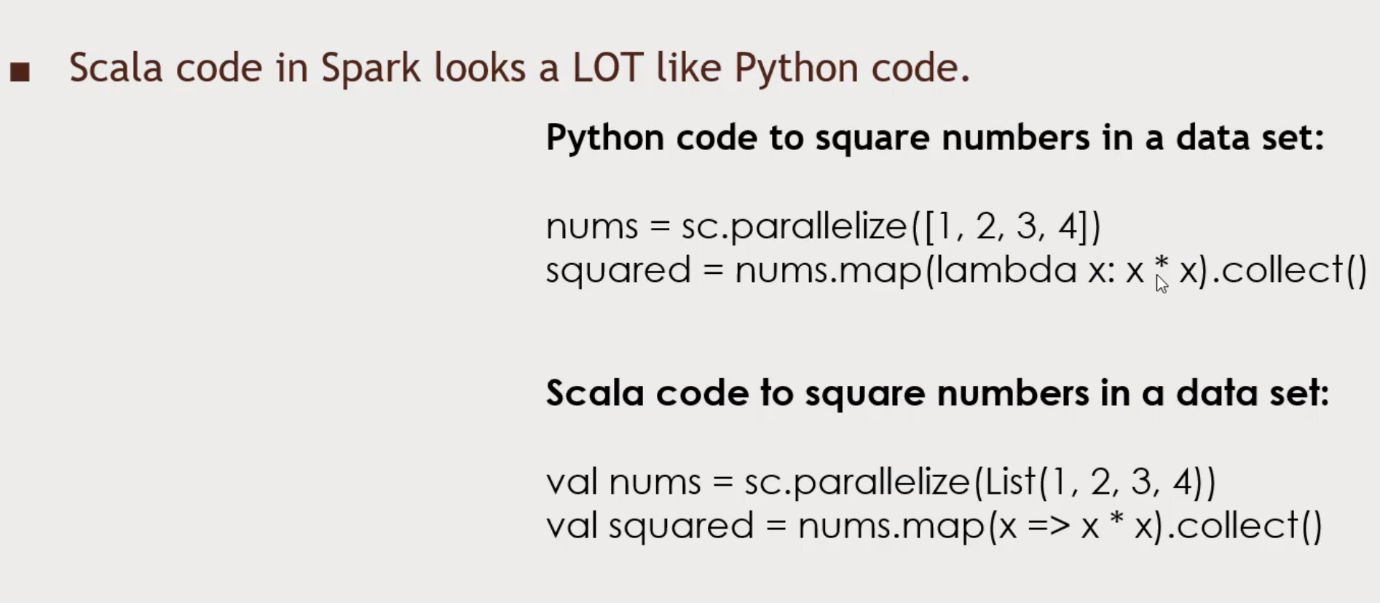
Spark SQL – SQL interface to Spark

MLLib – Machine Learning and Data Mining tools. Clustering, Regression analysis to Mappers and Reducers.

GraphX – Graph theory. Who is connected to who and whats the shortest way.

Python –

But. Spark itself is written in Scala

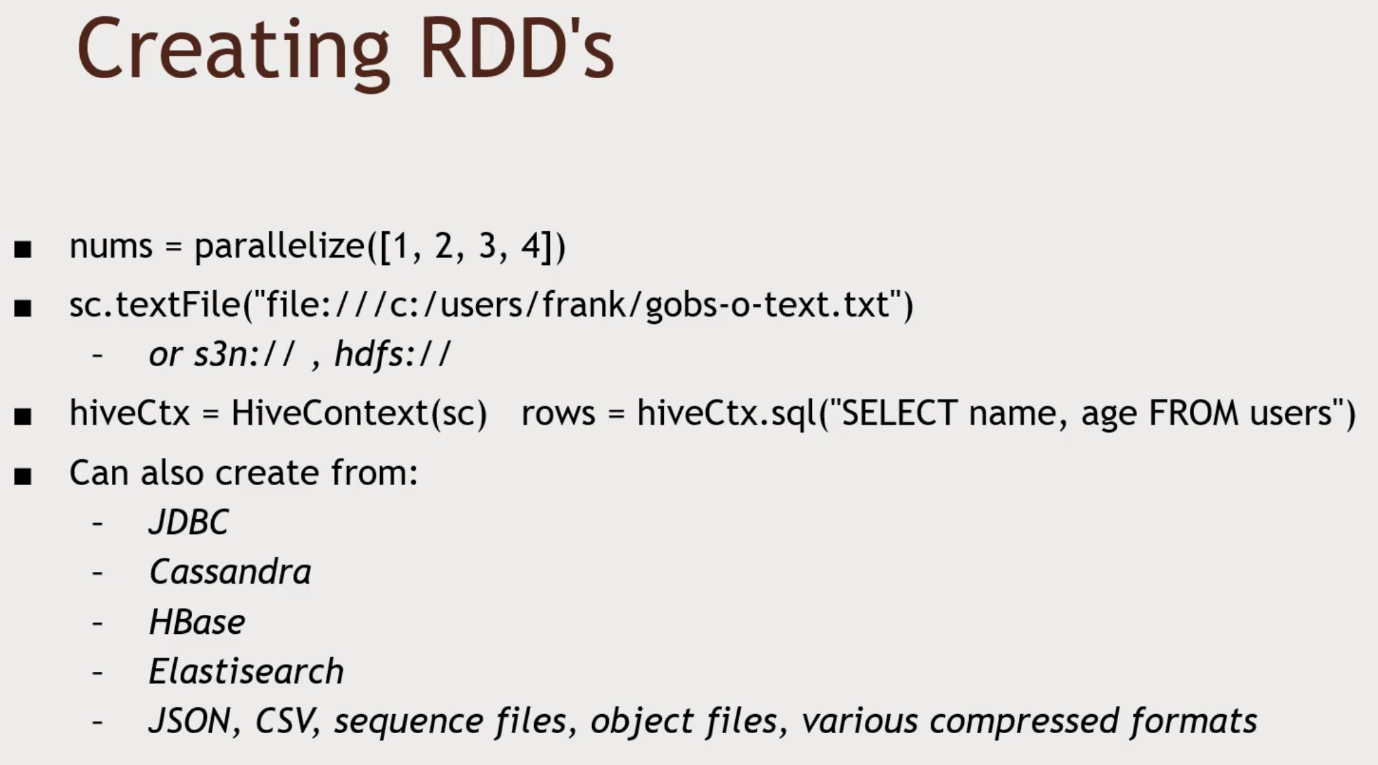
Scala’s functional programming model is a good fit for distributed processing. Gives you fast performance. (Scala combines to Java bytecode), Less code & boilerplate stuff than Java

**Resilient Distributed Dataset (RDD)**

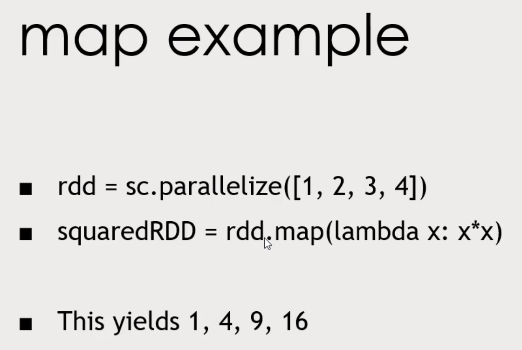
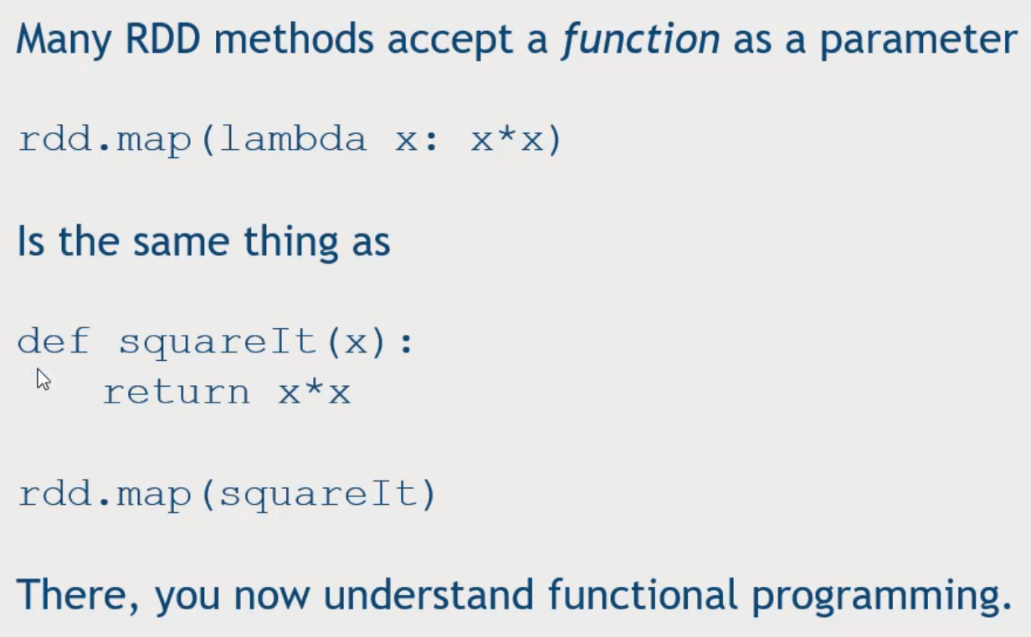
* Resilient
* Distributed
* Dataset

Storing information in an object that can do the right thing.

The spark Context

* Created by your driver program
* Is responsible for making RDD’s resilient and distributed!
* Creates RDD’s
* The Spark Shell creates a ‘sc’ object for you

Transforming RDDs

* Map – apply function to every input row of your RDD for transformation, 1 to 1 relationship between input and output
* Flatmap – Split out/discard rows. Can have any relationship between input and output
* Filter – take stuff out of RDD
* Distinct – Unique values
* Sample,
* Union, intersection, subtract, cartesian

RDD actions (reduce)

* Collect – suck them down into memory. Print it out
* Count – how many rows in the RDD
* countByValue – how many times each value occurs
* take – take top 10 results
* top
* reduce – define a function that combine all fn with each unique key.

Nothing actually happens in your driver program until an action is called.

Go to Ambari, Advanced spark-log4j.properties and change rootCategory = Error and restart the services.

Spark-submit to run the python script.

Datasets and Spark 2.0

* Working with structured data

Extends RDD to a ‘DataFrame’ object

DataFrames

* Contain Row objects
* Can run SQL queries
* Has a schema (leading to more efficient storage)
* Read and write to JSON, Hive, Parquet
* Communicates with JDBC/ODBC, Tableau

Using SparkSQL in Python

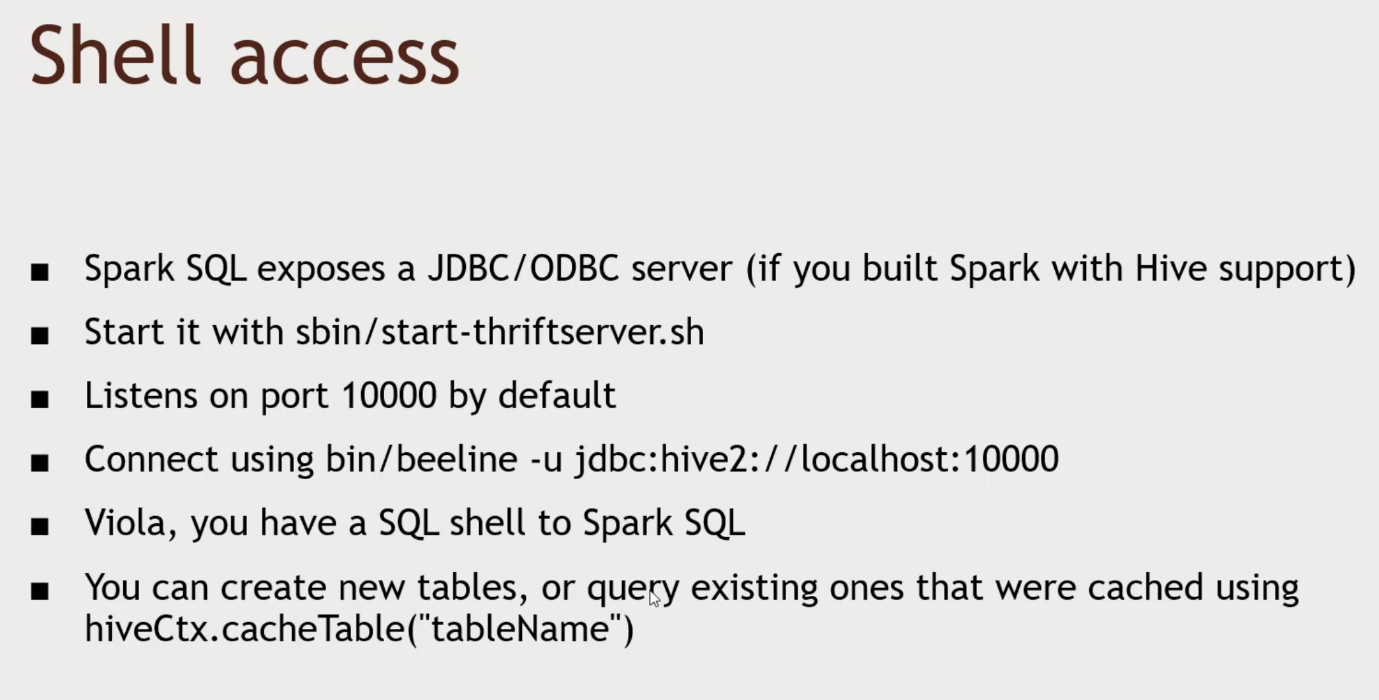
* From pyspark.sql import SQLContext, Row
* hiveContext = HiveContext(sc)
* inputData = spark.read.json(dataFile)
* inputData.createOrReplaceTempView(‘myStructuredStuff’)
* myResultDataFrame = hiveContext.sql(‘’’’SELECT foo FROM bar ORDER BY foobar”””)

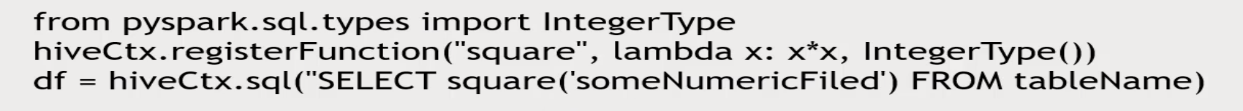
myResultDataFrame.show()

myResultDataFrame.select(‘someFieldName’)

myResultDataFrame.filter(myResultDataFrame(‘’someFieldname’’ >20)

myResultDataFrame.groupBy(myResultDataFrame(‘SomeFieldName’)).mean()

myResultDataFrame.rdd().map(mapperFunction)

User-defined Functions(UDF’s)