**Hortonworks Spark Exam Questions:**

1. Read csv file –Airline data
   1. Convert to dataframe
      1. case class (declare only needed columns with correct data type)
      2. toInt, toDouble
   2. Register as table
      1. rdd.toDF().registerTempTable(“”)
   3. Filter dataframe
      1. Add order by in SQL
      2. Where delay >=10 min
   4. Store dataframe as csv file
      1. dataFram.rdd.map(x=>x.mkString(",")).saveAsTextFile(“/user/Horton/task1”)
2. Read JSON file –customer balance by state
   1. Filter and save as tab file
   2. Average by state for customers with balance greater than 2000
      1. dataFram.rdd.map(x=>x.mkString("\t")).saveAsTextFile(“/user/Horton/task2”)
3. Read hive table –blizzard data
   1. Filter, avg, convert string to int (cast funcation), substring date column.
   2. Order by snow\_depth asc
4. Read hive table
   1. Filter, order by
   2. Data type conversion
5. Accumulator
   1. Read log file
   2. Filter a value and count accumulator
   3. Print accumulator
   4. Check your count matches with actual data
      1. Hdfs dfs –cat /user/Horton/data/file.txt|grep ERROR |wc -l
6. Read two text files and join
   1. Weather.txt, station.txt
      1. Data is tab delimited
      2. Temp has “\*\*\*” values. Filter it
      3. Filter all Temp >= 32
      4. Use caseclass. Use toInt data type

**val** data1= weaterdata.map(line =>line.split(**","**)).filter(row=>row(3)!=**"\*\*\*\*"**).filter(r=>r(3).toInt>=32)

* 1. Select few columns from each data set
  2. Save data to hdfs

1. Spark submit as yarn client
   1. spark-shell --master yarn --deploy-mode client
   2. Read all features of submit
2. ./bin/spark-submit –-jars SparkYarn.jar --class SparkYarn --master yarn-client SparkYarn.jar
3. **import** sqlContext.implicits.\_
   1. This important to remember
4. Spark shell
   1. sc is context
   2. sqlContext – for sql
   3. convert dataframe to rdd to before save
5. **How to run spark file from spark-shell**
   1. **You need to save code in a file. Run your script file as shown below**

spark-shell -i task1