Overview of the Final Project

Gain exposure to EMR and Spark. You're gonna develop ETL pipelines with pyspark and postgres. We're extending the IMDB database with new data.

EMR is not covered by free tier. Remember to terminate the cluster after your spark job completes.

Overview of milestones

Using augmented data from Movielens, The-numbers and Cinemalytics for better analytics

Reading Quiz

Q1) A. The user writes java code containing the map and reduce functions. He/she then packages this as a JAR file and submits it as a Hadoop job.

Q2) B. The answer is RDD.

Q3) A. RDD is partitioned across across multiple nodes in a cluster. Spark has a lineage of all the transformations that are applied on an RDD. If there is a failure, it uses the lineage to recompute the results. The job will be slower(because of re-computations), but it won't result in total failure. RDDs are immutable. Transformation is applied to an RDD - original RDD is not changed by this. Spark is creating a new RDD each time there is a transformation. They are evaluated in a lazy manner by Spark. When you call map, you don't see the output. You have to call a function to execute the map transformations. Spark takes the list of transformations until the action and figures how to compute the transformations in an optimal manner.

Q4) B. When you call reduce, it triggers a computation behind the scenes. Spark evaluates all the transformation until the reduce function and output the results.

Q5) D.

Lecture

RDD key concepts:

Operations on RDD - transformations and actions(like reduce). Base RDD reads data from a data source - S3 or HDFS.

Map: uses a function as a parameter. Returns a new RDD as the output. sc: SparkContext, your starting point. Distributes data across the worker nodes - parallelize(). Numbers is your base RDD. You can either use a lambda function provided by python, or your own function. Collect() is an action - it fetches all elements from RDD, puts it in an array and returns it to the driver program(master node).

Filter: Similar to SQL where clause. Again, provide a boolean function as an input.

Reduce: Calculates a single aggregate over all elements of an RDD. It keeps applying the reduce function until there's only a single value remains in the RDD. There's no schema or structure to RDD, they're considered a sequence of characters. So having several duplicate values is fine.

ReduceByKey: Works like SQL group by. Now RDD - each element is a key,value pair. We use map to create a key,value pair. Key = name of the state, value = 1. ReducebyKey function, applies the aggregation function on a key-by-key basis.