**GCD Assignment Week 4**

Hadoop

**Apache Hadoop** is an open source [software framework](https://en.wikipedia.org/wiki/Software_framework) written in [Java](https://en.wikipedia.org/wiki/Java_%28programming_language%29) for [distributed storage](https://en.wikipedia.org/wiki/Clustered_file_system) and [distributed processing](https://en.wikipedia.org/wiki/Distributed_processing) of very large data sets on [computer clusters](https://en.wikipedia.org/wiki/Computer_cluster) built from [commodity hardware](https://en.wikipedia.org/wiki/Commodity_hardware). All the modules in Hadoop are designed with a fundamental assumption that hardware failures (of individual machines, or racks of machines) are common and thus should be automatically handled in software by the framework.

The core of Apache Hadoop consists of a storage part ([Hadoop Distributed File System](https://en.wikipedia.org/wiki/Apache_Hadoop#HDFS) (HDFS)) and a processing part ([MapReduce](https://en.wikipedia.org/wiki/MapReduce))**.**Hadoop splits files into large blocks and distributes them amongst the nodes in the cluster. To process the data, Hadoop MapReduce transfers [packaged code](https://en.wikipedia.org/wiki/JAR_%28file_format%29) for nodes to process in parallel, based on the data each node needs to process. This approach takes advantage of data locality-nodes manipulating the data that they have on hand—to allow the data to be processed faster and more efficiently than it would be in a more conventional supercomputer architecture that relies on a parallel file system where computation and data are connected via high-speed networking.

The base Apache Hadoop framework is composed of the following modules:

* *Hadoop Common* – contains libraries and utilities needed by other Hadoop modules;
* *Hadoop Distributed File System (HDFS)* – a distributed file-system that stores data on commodity machines, providing very high aggregate bandwidth across the cluster;
* *Hadoop YARN* – a resource-management platform responsible for managing computing resources in clusters and using them for scheduling of users' applications;
* *Hadoop MapReduce* – a programming model for large scale data processing.

HDFS

The Hadoop Distributed File System (HDFS) is a distributed file system designed to run on commodity hardware. It has many similarities with existing distributed file systems. However, the differences from other distributed file systems are significant. HDFS is highly fault-tolerant and is designed to be deployed on low-cost hardware. HDFS provides high throughput access to application data and is suitable for applications that have large data sets. HDFS relaxes a few POSIX requirements to enable streaming access to file system data.

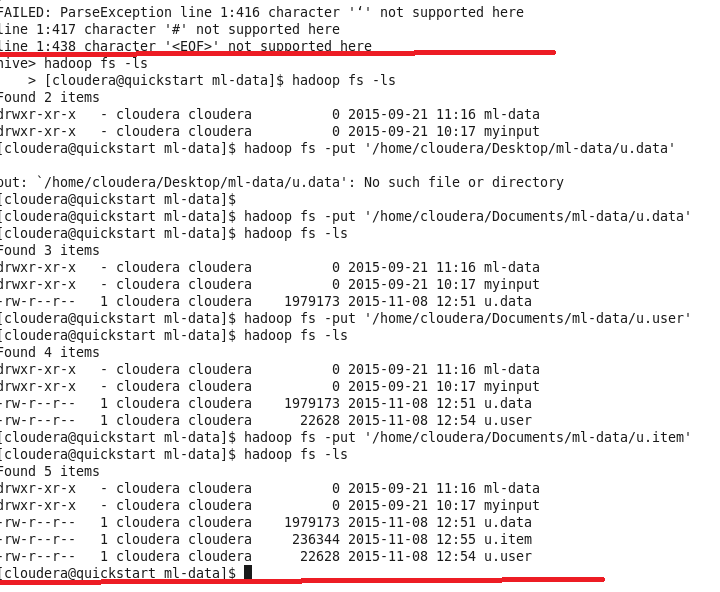
**Large Data Sets (HDFS)**

Applications that run on HDFS have large data sets. A typical file in HDFS is gigabytes to terabytes in size. Thus, HDFS is tuned to support large files. It should provide high aggregate data bandwidth and scale to hundreds of nodes in a single cluster. It should support tens of millions of files in a single instance.

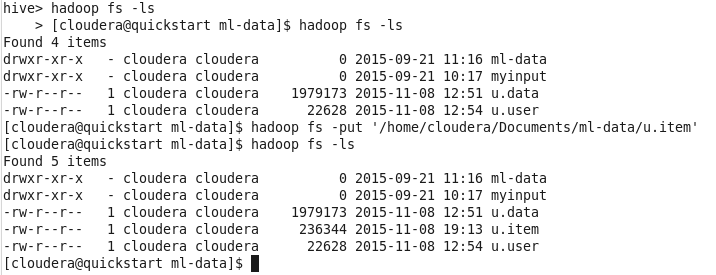
**Data Integrity (HDFS)**

It is possible that a block of data fetched from a DataNode arrives corrupted. This corruption can occur because of faults in a storage device, network faults, or buggy software. The HDFS client software implements checksum checking on the contents of HDFS files. When a client creates an HDFS file, it computes a checksum of each block of the file and stores these checksums in a separate hidden file in the same HDFS namespace. When a client retrieves file contents it verifies that the data it received from each DataNode matches the checksum stored in the associated checksum file. If not, then the client can opt to retrieve that block from another DataNode that has a replica of that block.

**Activity 1 : Store files in HDFS**

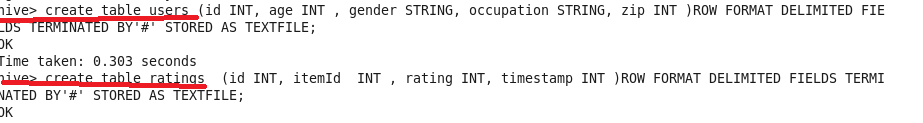


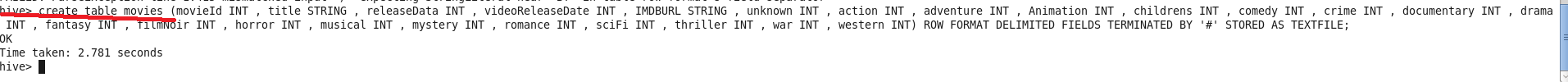
* Second Try



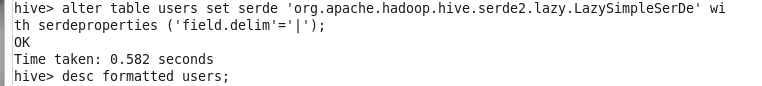
**Activity 2 : Import HDFS files to Hive**

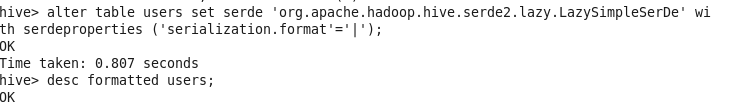
* **CREATING TABLES:**



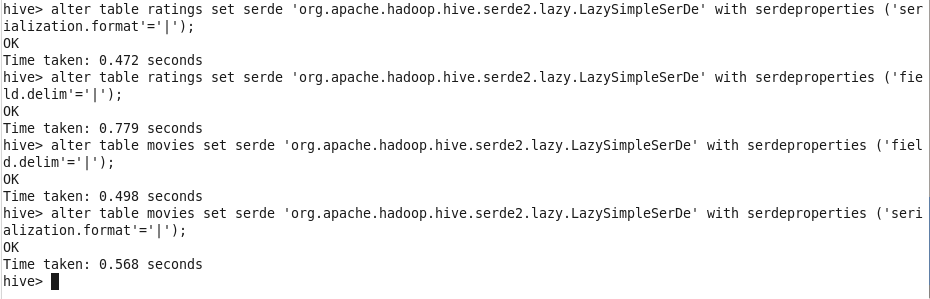


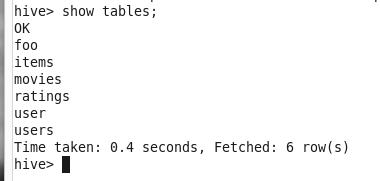
* CHANGING THE DELIMITER OF THE TABLES



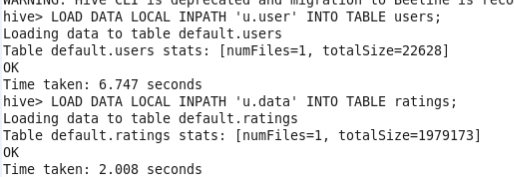


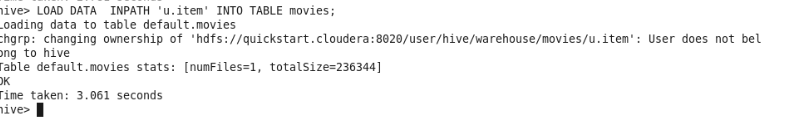


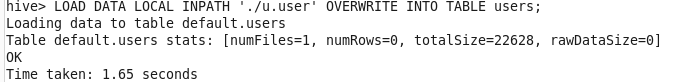




* **IMPORTING DATA INTO TABLES :**





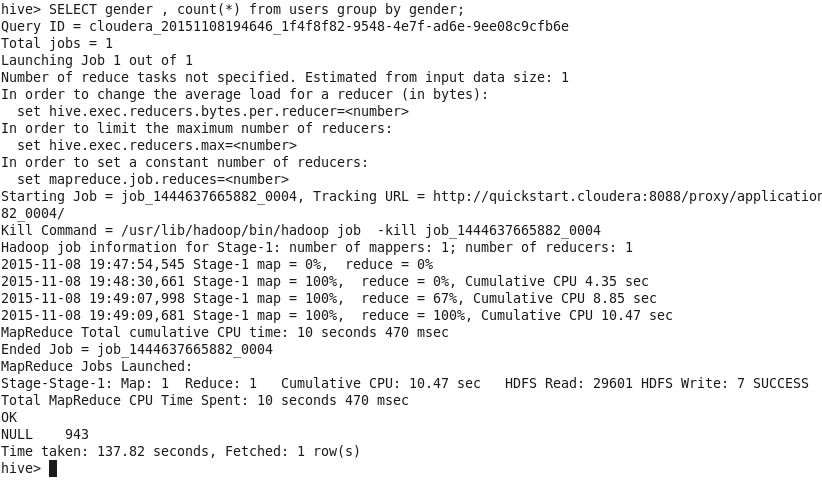


**Activity 3 : Hive Queries**

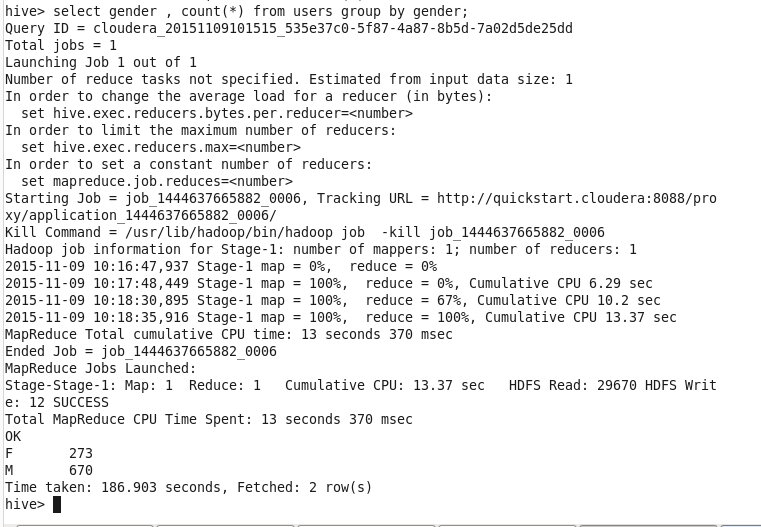
* **Give the number of male and female users.**

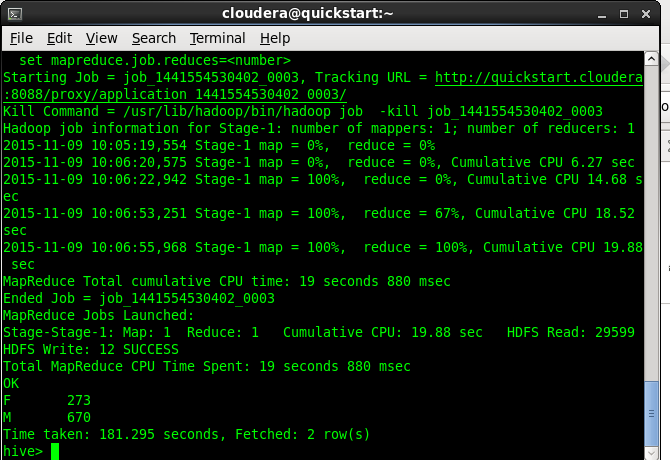


But we had problem the first time

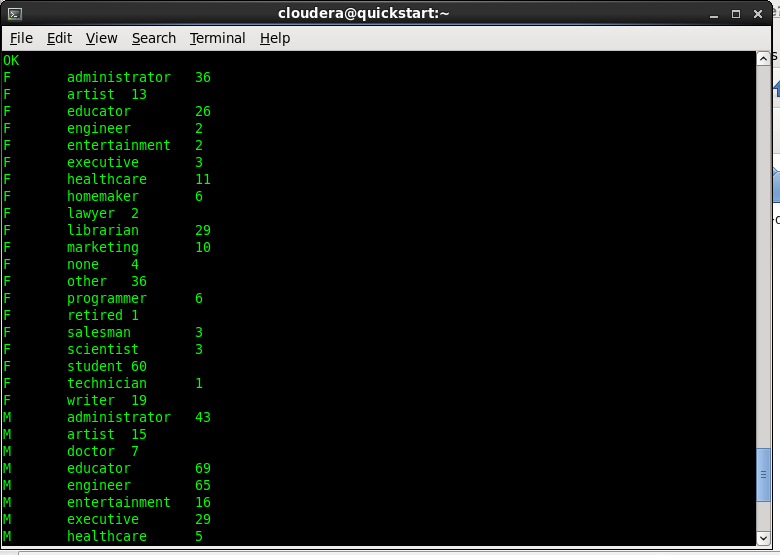


After that we fixed it

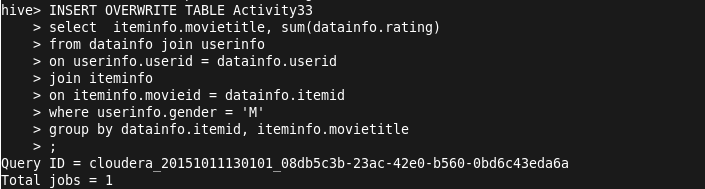


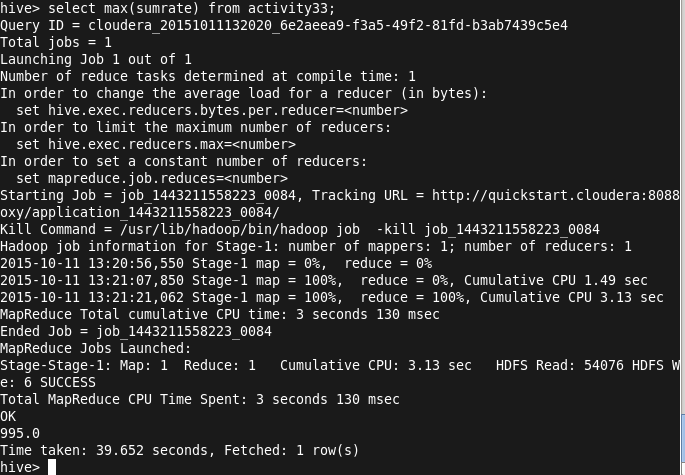


**Give the number of men and women per occupation:**



**Activity 3.3**





**Activity 4**

* **In question 3: how do you calculate “highest rating”? Do you use MAX, SUM, AVG? Or an other function?**

First I joined the three tables to get a list of movie titles with their total number of rates and inserted it into a table called Activity33, from then I took the new table and got the maximum of the rates.

**Activity 5:**

How does IMDB rate the movies?

1. MDb registered users can cast a vote (from 1 to 10) on every released title in the database. Users can vote as many times as they want but every vote will overwrite the previous one so it is one vote per title per user.
2. They took all the individual votes cast by IMDb registered users and use them to calculate a single rating. They don't use the arithmetic mean of the votes .The rating displayed on a film's page is a **weighted average**.
3. The following formula is used to calculate the Top Rated 250 titles. This formula provides a true 'Bayesian estimate', which takes into account the number of votes each title has received, minimum votes required to be on the list, and the mean vote for all titles:  
     
   weighted rating (WR) = (v ÷ (v+m)) × R + (m ÷ (v+m)) × C   
     
   Where:   
   R = average for the movie (mean) = (Rating)  
   v = number of votes for the movie = (votes)  
   m = minimum votes required to be listed in the Top 250 (currently 25,000)  
   C = the mean vote across the whole report

**Conclusion :**

They take all the individual votes cast by IMDB registered users and use them to calculate a single rating. The rating displayed on a film’s page is a weighted average.