1 Initial notes

It is recommended to do Hadoop stream jobs using Python rather than **R**, especially since the Amazon Web Service (AWS) machines have an old version of **R**. It is also recommended to slightly modify Python code from the previous lecture for Lab 2, rather than trying to write it yourself in **R**. Thirdly, the Amazon machines have Hive and Pig preconfigured; unsure about Mahout.

2 The Amazon Web Service (AWS)

To upload data, do the following. Prerequisite: need a public/private key for AWS (instructions provided in Lab 2).

1. Go to *https* : *//console.aws.amazon.com/console/* and log in.
2. Click on Elastic Map Reduce, then Create New Job Flow. This will show several options for setting up your "cluster" for Hadoop. What options to use are provided in Lab 2. You will be asked for a "Amazon EC2 Key Pair" in one of the menus; this is your private key obtained as a prerequisite.
3. When complete (i.e. the cluster is starting), it will take a little while for everything to set up. Refresh the page until you see a value in the "Master DNS Name" field (abbreviated as MDN).
4. Now, use the MDN the ssh onto AWS. **ssh -i** [private key file extension .pem] **hadoop@**[MDN], where the values in [ ] are to be replaced accordingly.
5. Once successful, make a directory. **hadoop fs -mkdir** [mydirectory]
6. Transfer your data files from your data bucket on https://aws.amazon.com/s3 to the new directory. Note: you won't need to create a data bucket, Lab 2 provides the data bucket already. **hadoop distcp s3://**[data bucket name] [mydirectory]
7. !!! **Terminate your job when finished through https://console.aws.amazon.com/console/** !!! It costs money to keep it open, even if you have no jobs running.

Notes: When your hadoop code is first run, it won't work, unless you're some mad pro. To check on your job's status, open a new terminal and ssh into hadoop@[...] again. Run **lynx http://localhost:9100/** to show a JobTracker link; this link can be opened in a browser to show your job's status. An overview of these steps can also be seen in Lecture 10 slides, page 10.

3 Hive

Tutorials: *https* : *//cwiki.apache.org/confluence/display/Hive/Tutorial* Hive can be run on hadoop@[...] by using *hive* on the command line.
When storing data for Hive (which are stored in structures like dataframes). The columns can have the following types of data:

- **TINYINT, SMALLINT, INT, BIGINT** are 1, 2, 4, 8 byte ints respectively
- **BOOLEAN** are TRUE/FALSE
- **FLOAT, DOUBLE** are single and double precision values respectively
- **STRING** are characters
- **STRUCTS** have form like STRUCT{a INT; b INT}, where a and b are variables for the struct
- **ARRAYS** are indexed by ARRAY[0], ..., ARRAY[n-1]

3.1 Tables

Hive code is similar to SQL. To create a table with the data loaded from the S3 website:
CREATE TABLE [name]**(**
[variable$_1$] [variable type], ..., [variable$_k$] [variable type]
);
ROW FORMAT DELIMITED FIELDS TERMINATED BY ’’
STORED AS SEQUENCEFILE
LOCATION ’S3://[data bucket name]**’**
To see what variables are in a table, use **Describe** [table name].

3.2 Select a column

The following selects the column for a variable, with an optional condition (e.g. var$_1$¿100) only selects the rows if the condition is satisfied.
SELECT [variable] **from** [table] **WHERE** [conditions]

3.3 Inserting into a table

You can’t specifically add to a table; you will need to create another table with whatever added, or load the data from a site like before.
INSERT OVERWRITE TABLE [table name]
SELECT [var$_1$], ..., [var$_p$]
FROM [table name source]
WHERE [optional conditions]

3.4 Additional commands

Describe [table name] - shows the variables and types of the columns for the table.
SHOW TABLES - shows all defined tables.