1] Demonstration - Understanding MapReduce

1) Download attachment in **STAGING\_AREA**

2) Pull it in **LABS\_HOME/demos**

3) Open the lab manual

4) Wait for the instructions from an Instructor

Dataset:-

* data/constitution.txt

2] Writing down WordCount Program

1) Download the rar in **STAGING\_AREA** location

2) Extract it

3) Pull the extracted folder in LABS\_HOME/wordcount

4) Wait for the instructions from an Instructor

Code/Dataset:-

* data/wordcount.rar

3] Distributed GREP

Here attached the rar file. Perform following activities

1) Download the rar file in **STAGING\_AREA**, extrat it

2) Create a folder grep in **LABS\_AREA**. Import the java files from step 1 to this folder

3) Create a project **grep** in Eclipse

4) Import the java file from step 2 in the same

5) Resolve the compilation errors by adding hadoop client side libraries in class path of the project

6) Implement all **TODO**s in the code

7) Look at the comments at the end of the file and create a jar and execute it

8) During implementation of the code refer the API documentation.

9) What differences you are finding in this code and previous WordCount code?

**Time Line =** 40 Mins

Code/Dataset:-

* data/grep.rar

4] Inverted Index

Here attached the rar file. Perform following activities

1) Download the rar file in **STAGING\_AREA** location

2) Create a folder invertedindex in **LABS\_AREA**.

3) Create a project **invertedindex** in Eclipse

4) Import the java file in the same

5) Resolve the compilation errors by adding hadoop client side libraries in class path of the project

6) Implement all TODOs in the code

7) Look at the comments at the end of the file and create a jar and execute it

8) During implementation of the code refer the API documentation.

9) Have you seen such type of applications?

**Time Line =** 40 Mins

Code/Dataset:-

* data/InvertedIndex.rar

5] Implementing Counter

1) Download an attachment in **STAGING\_AREA** location

2) Extract it

3) Pull it to **LABS\_AREA**/counter

4) Study the code

5) Create **counties** folder in your home on HDFS

6) Put all **counties\_\*.csv** files from local linux file system to **counties** folder on HDFS

7) Run the application with command as **yarn jar average.jar <Main Class>**

8) See the output of the job on terminal window and confirm whether counters you created are working or not

Study the code

**Time Line** = 30 Mins

Code/Dataset:-

* data/Average.rar

6] Implement following business requirement for Average application above where

1) Business wants to know how many records are Good records and

how many are bad.

2) It should notify client about the number of good and bad records

3) Bad records should not be processed

7] Compression

1) Download the rar file in **STAGING\_AREA**

2) Extract it

3) Pull the extracted folder in **LABS\_AREA**

4) Create a **logfiles** folder on HDFS and put the log files from project folder into it

5) Create an eclipse project **compression**

6) Import the java files from project folder in **STAGING\_AREA** in to this project

7) Remove the compilation errors by adding hadoop client side libraries

8) Create a jar **compression.jar**

9) Have a first Run using **yarn jar compression.jar INFO** command

10) Wait for the instructions from the instructor

Code/Dataset:-

* data/Compression.rar

8] Streaming Job

1) Download RAR in **STAGING\_AREA**

2) Extract it

3) Pull the extracted folder in **LABS\_AREA**

4) Wait for the instructions from an Instructor

Code/Dataset:-

* data/MRStreaming.rar