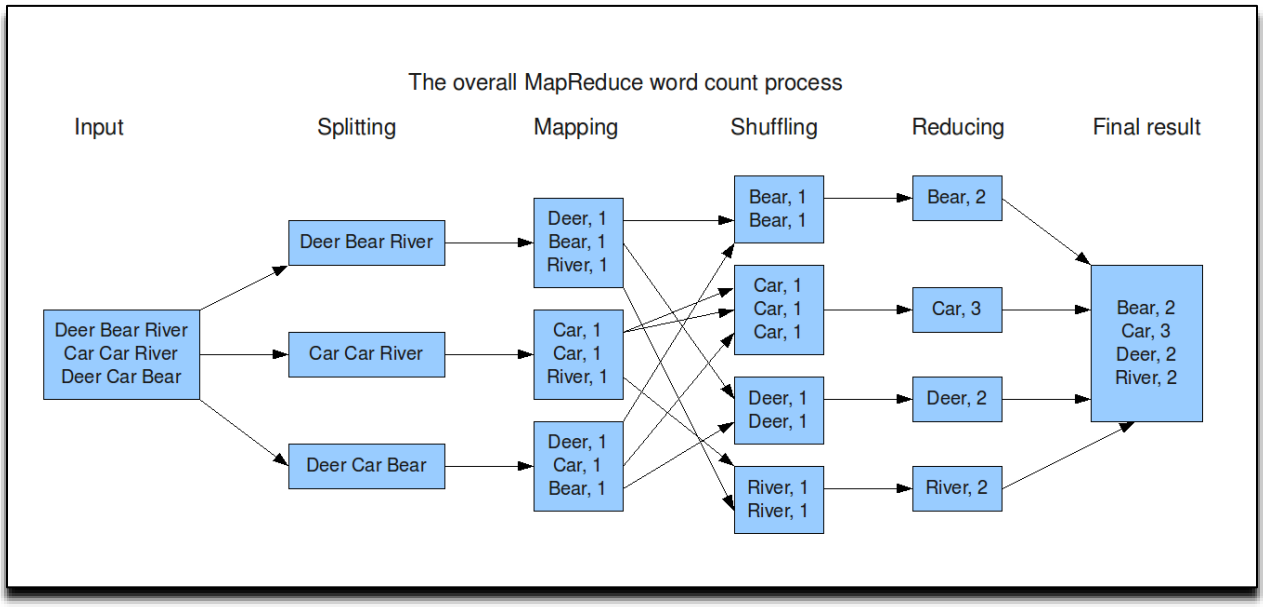
COSC 3365 – Distributed Databases Using Hadoop
Project 4 – MapReduce WordCount Example

In the MapReduce framework, a job reads the text from a set of text files distributed across a cluster's nodes via the distributed file system. A "mapper" process converts the input to a set of <key, value> pairs. A "shuffle/sorter" process sorts these pairs by their *key* values. A "reducer" process then combines these sorted pairs in a way that solves a problem, and typically produces a different set of <key, value> pairs, which is the output of the job.

The basic steps of a job are thus:
(input)
-> splitting -> <k1, v1>
-> mapping -> <k2, v2>
-> shuffling/sorting -> <k2, v2>
-> reducing -> <k3, v3>
-> (output)



This *WordCount* problem is often used as the "Hello World" example for MapReduce computations. In this assignment, you will create a Java program that counts the number of occurrences of each word in a text file.

Use the attached Java program, **WordCount.java**, and Eclipse or your favorite IDE to complete this assignment. **(Note: Help is provided for only the Eclipse IDE)**

Download the attached file, **WordCount.java** and rename it to **Project4.java.** Change the file name on line 7 to **Project4.java.** Change the class name, **WordCount,** on line 33 to **Project4.** Change **WordCount.class** on line 54 to **Project4.class.**

Rewrite the **MapForWordCount** and **ReduceForWordCount** classes as separate public non-static classes and do not nest them in the Project4 class. Import the appropriate packages used in these classes and remove the ones not used in the Project4 class.

Complete the comment section for the all the classes.

Insert the appropriate code where necessary.

Add the Hadoop jar files to correct the errors in the code.
See the attached file **Eclipse-Help.pdf** for help

Compile the program into a .jar file for Hadoop.
See the attached file **Eclipse-Help.pdf** for help

Create a directory in HDFS and move the input text file, **Project4.txt**, there.
Take a screenshot of this process showing the HDFS command and save the image as JPG or PNG in a file named Project4-1.xxx (where "xxx" should indicate the file format).

Execute the Hadoop jar file.
Take a screenshot of this process showing the Hadoop command and save the image as JPG or PNG in a file named Project4-2.xxx (where "xxx" should indicate the file format).

Display the output file in HDFS and verify that it is correct.
Take a screenshot of this process showing the HDFS command and save the image as JPG or PNG in a file named Project4-3.xxx (where "xxx" should indicate the file format).

Take a screenshot of your output and save the image as JPG or PNG file named Project4-4.xxx (where "xxx" should indicate the file format).

Create a folder named, **YourFullName_Project4**. Copy your image files, the *.java files, and the jar file to the folder. **Zip the folder as *.zip file and upload it to Blackboard**.

Before you upload your project to Blackboard:

- Ensure that your code conforms to the style expectations set out in class and briefly discussed below.

- Make sure your variable names and methods are descriptive and follow standard capitalization conventions.

- Put comments wherever necessary. Comments at the top of each module should include your name, file name, and a description of the module. Comments at the beginning of methods describe what the method does, what the parameters are,

and what the return value is. Use comments elsewhere to help your reader follow the flow of your code.

- *Program readability and elegance are as important as correctness.* After you have written your method, read and re-read it to eliminate any redundant lines of code, and to make sure variables and methods names are intuitive and relevant.

Read the assignment very carefully to ensure that you have followed all instructions and satisfied all requirements. **You will not get full credit for this project if it is not written as instructed even if it works as expected.**