School of Computing & Information Technology
University of Wollongong

Session: **4, September 2021**
Lecturer: **Janusz R. Getta**

CSCI312 Big Data Management
Singapore 2021-4
Assignment 1

Published on 25 September 2021

Scope

The objectives of Assignment 1 include implementation of HDFS applications, implementation of simple MapReduce applications, and describing an implementation of complex MapReduce application.

This assignment is due on **Saturday, 18 October 2021, 8:00pm** (sharp) Singaporean Time (SGT).

This assignment is worth **10%** of the total evaluation in the subject.

The assignment consists of 4 tasks and specification of each task starts from a new page.

Only electronic submission through Moodle at:
`https://moodle.uowplatform.edu.au/login/index.php`
will be accepted. A submission procedure is explained at the end of Assignment 1 specification.

A policy regarding late submissions is included in the subject outline.

Only one submission of Assignment 1 is allowed and only one submission per student is accepted.

A submission marked by Moodle as "late" is always treated as a late submission no matter how many seconds it is late.

A submission that contains an incorrect file attached is treated as a correct submission with all consequences coming from the evaluation of the file attached.

All files left on Moodle in a state "`Draft(not submitted)`" will not be evaluated.

A submission of compressed files (zipped, gzipped, rared, tared, 7-zipped, lhzed, … etc) is not allowed. The compressed files will not be evaluated.

An implementation that does not compile well due to one or more syntactical and/or run time errors scores no marks.

The first assignment is an **individual assignment** and it is expected that all its tasks will be solved **individually without any cooperation** with the other students. However, it is allowed to declare in the submission comments that a particular component or task of this assignment has been implemented in cooperation with another student. In such a case evaluation of a task or component may be shared with another student. In all other cases plagiarism will result in a **FAIL** grade being recorded for entire assignment. If you have any doubts, questions, etc. please consult your lecturer or tutor during laboratory/tutorial classes or over e-mail.

Task 1 (2 marks)
Implementation of HDFS application

Implement a HDFS application that merges two files located in HDFS into one file also located in HDFS.

The application must have the following parameters.
(1) A path to, and a name of the first input file in HDFS.
(2) A path to, and a name of the second input file in HDFS.
(3) A path to, and a new name of an output file to be created in HDFS. The file supposed to contain the contents of the first input file followed by the contents of the second input file.

Perform the following steps.

Implement the application and save its source code in a file `solution1.java`.

Upload to two files to HDFS. The contents, the name, and the locations of the files in HDSF are up to you.

When ready, compile, create `jar` file, and process your application. Display the results created by the application.

Use Hadoop to provide an evidence, that two files uploaded into HDFS has been successful merged in one file in HDFS.

Deliverables
A file `solution1.java` with a source code of the application that merges two HDFS files. A file `solution1.pdf` that contains the contents of Terminal window with a report from compilation, creation of jar file, uploading to HDFS two small files for testing, processing of the application, and an evidence that two files uploaded into HDFS has been successful merges in one file in HDFS.

Task 2 (3 marks)
Implementation of MapReduce application

Assume, that a speed camera records the speed of passing cars and saves the measurements in a text file. The speed of each car is measured in kilometres per hour. A single row in the file contains a car registration number, a date when the speed has been measured and the speed of a car with the recorded registration number. The values are always separated with a single blank.

For example, a sample file with the speed measurements contains the following lines:

```
PKR856 12-DEC-2018 120
UPS234 17-JAN-2019 190
PKR856 12-FEB-2018 80
PKR856 01-JAN-2019 60
UPS234 21-OCT-2020 200
UPS234 22-OCT-2020 160
```

Assume, that a speed limit in a location of the speed camera is `60` kilometres per hour.

Your task is to implement a MapReduce application, that finds an average speed of all cars, that exceeded a speed limit in the location of the speed camera.

An input file with the speed measurements must include the lines listed above and it must contain at least 20 measurements. All additional measurements are up to you.

Save your solution in a file `solution2.java`.

When ready, compile, create `jar` file, and process your application. Display the results created by the application. Next, list your input file with the speed measurements. When finished, Copy and Paste the messages from a Terminal screen into a file `solution2.pdf`.

Deliverables
A file `solution2.java` with a source code of the application that implement the functionality of `SELECT` statement given above. A file `solution2.pdf` with a report from compilation, creating `jar` file, processing, displaying the results of processing `solution2.java`, and listing of your input file with the speed measurements.

Task 3 (2 marks)
Implementation of MapReduce application

An application MinMax described in an Exercise 2 has the functionality the same as the following SQL statement.

```
SELECT key, MIN(value), MAX(value)
FROM Sequence-of-key-value-pairs
GROUP BY key;
```

Extend Java code of the application such that it implements the functionality the same as the following SQL statement.

```
SELECT key, key, MIN(value), MAX(value), SUM(value), AVG(value)
FROM Sequence-of-key-value-pairs
GROUP BY key;
```

Save your solution in a file `solution3.java`.

When ready, compile, create `jar` file, and process your application. To test your application, you can use a file sales.txt included in a folder with a specification of Exercise 2. Display the results created by the application. When finished, Copy and Paste the messages from a Terminal screen into a file `solution3.pdf`.

Deliverables
A file `solution3.java` with a source code of the application that implement the functionality of `SELECT` statement given above. A file `solution3.pdf` with a report from compilation, creating `jar` file, processing, and displaying the results of processing `solution3.java`.

<u>Task 4 (3 marks)</u>
Describing MapReduce implementation of join operation

Assume, that a file `measurement.txt` contains the speed measurements of the passing cars (see Task 2). A single row in the file contains a car registration number, a date when the speed has been measured and the speed of a car with the recorded registration number. A sample contents of the is listed below.

```
PKR856 12-DEC-2018 120
UPS234 17-JAN-2019 190
PKR856 12-FEB-2018 80
PKR856 01-JAN-2019 60
UPS234 21-OCT-2020 200
UPS234 22-OCT-2020 160
...    ...
```

Assume, that a file `car.txt` contains the technical descriptions of the cars. A single row in the file contains a car registration number, maximum speed and fuel consumption

```
PKR856 180 10
UPS234 200 15
...    ...
```

Assume that both files have been loaded to HDFS.

Write the comprehensive explanations how would you implement in Java a MapReduce application that finds the cars, that reached its maximum speed at the speed checkpoint.

Save you explanations in a file `solution4.pdf`. This task does not require you to write any code in Java. However, the comprehensive explanations related to all stages of data processing are expected. You are allowed to support your explanations with the fragments of pseudocode. Try to be as specific as it is possible.

Deliverables
A file `solution4.pdf` with the comprehensive explanations how would you implement in Java a MapReduce application that finds the cars that reached its maximum speed at the speed checkpoint.

Submission of Assignment 1

Note, that you have only one submission. So, make it absolutely sure that you submit the correct files with the correct contents. No other submission is possible !

Submit the files **`solution1.java`**, **`solution1.pdf`**, **`solution2.java`**, **`solution2.pdf`**, **`solution2.java`**, **`solution3.java`**, **`solution3.pdf`**, and **`solution4.pdf`** through Moodle in the following way:

(1) Access Moodle at **`http://moodle.uowplatform.edu.au/`**
(2) To login use a **`Login`** link located in the right upper corner the Web page or in the middle of the bottom of the Web page
(3) When logged select a site **`ISIT312 (SP421) Big Data Management`**
(4) Scroll down to a section **`SUBMISSIONS`**
(5) Click at **`In this place you can submit the outcomes of your work on the tasks included in Assignment 1`** link.
(6) Click at a button **`Add Submission`**
(7) Move a file **`solution1.java`** into an area **`You can drag and drop files here to add them`**. You can also use a link **`Add`**...
(8) Repeat step (7) for the remaining files **`solution1.pdf`**, **`solution2.java`**, **`solution2.pdf`**, **`solution2.java`**, **`solution3.java`**, **`solution3.pdf`**, and **`solution4.pdf`**
(9) Click at a button **`Save changes`**
(10) Click at a button **`Submit assignment`**
(11) Click at the checkbox with a text attached: **`By checking this box, I confirm that this submission is my own work,`** ... in order to confirm authorship of your submission.
(12) Click at a button **`Continue`**

End of specification