COSC 2637/2633 Big Data Processing
Assignment 2 – Handling Big Data with Apache Pig

Assessment Type	– Individual assignment. – Submit online via Canvas → Assignment 2. – Marks awarded for meeting requirements as closely as possible. – Clarifications/updates may be made via announcements or relevant discussion forums.
Due Date	23:59, 29 Sep
Marks	25

Overview
Write Apache Pig scripts which give you a chance to develop a basic understanding of principles when handling queries on large data stored on HDFS.

Learning Outcomes
The key course learning outcomes are:
- CLO 1: model and implement efficient big data solutions for various application areas using appropriately selected algorithms and data structures.
- CLO 2: analyze methods and algorithms, to compare and evaluate them with respect to time and space requirements and make appropriate design choices when solving real-world problems.
- CLO 3: motivate and explain trade-offs in big data processing technique design and analysis in written and oral form.
- CLO 4: explain the Big Data Fundamentals, including the evolution of Big Data, the characteristics of Big Data and the challenges introduced.
- CLO 6: apply the novel architectures and platforms introduced for Big data, i.e., Hadoop, MapReduce and Spark.

Assessment Details
This assignment adopts a sample Olympics database from an online source. It has games, event, competitor, and several other tables. The database is originally managed by a DBMS. If the database is too large, you should store the data on a cluster of computers and manage it using a Hadoop platform. Therefore, you need to develop Apache Pig scripts to process the data like SQL queries.

From the tables `noc_region, person_region, person, competitor_event, medal` of the database, a subset of tuples has been extracted and stored in five files `noc_region.csv, person_region.csv, person.csv, competitor_event.csv,` and `medal.csv`, respectively.

Samples in the five files:

person_region.csv

```
person_id,region_id
1,42
2,42
3,56
4,56
5,146
6,217
7,217
8,146
9,69
…
```

person.csv

```
id,full_name,gender,height,weight
1,"A Dijiang",M,180,80
2,"A Lamusi",M,170,60
3,"Gunnar Nielsen Aaby",M,0,0
4,"Edgar Lindenau Aabye",M,0,0
5,"Christine Jacoba Aaftink",F,185,82
6,"Per Knut Aaland",M,188,75
7,"John Aalberg",M,183,72
8,"Cornelia "Cor"" Aalten (-
Strannood)"",F,168,0
9,"Antti Sami Aalto",M,186,96
…
```

noc_region.csv

```
id,noc,region_name
1,AFG,Afghanistan
2,AHO,"Netherlands
Antilles"
3,ALB,Albania
4,ALG,Algeria
5,AND,Andorra
6,ANG,Angola
7,ANT,"Antigua and Barbuda"
8,ANZ,Australasia
9,ARG,Argentina
…
```

competitor_event.csv	medal_name.csv
event_id,competitor_id,medal_id 1,1,4 2,2,4 3,3,4 4,4,1 5,5,4 6,5,4 5,6,4 6,6,4 …	id,medal_name 1,Gold 2,Silver 3,Bronze 4,NA

Task 1. (8 marks) Developing an Apache Pig script that outputs the same as the following SQL query.

```
select noc_region.region_name as Region, count(*) as Gold
from noc_region,person_region, person, competitor_event, medal
where noc_region.id = person_region.region_id and
person.id = person_region.person_id and
competitor_event.competitor_id = person_region.person_id and
medal.id =  competitor_event.medal_id and
medal.medal_name = "Gold"
group by noc_region.id
order by Gold DESC, Region asc;
```

The output

Region	Gold
USA	8
Norway	5
Italy	3
Australia	2
Egypt	2
Estonia	2
Germany	2
Japan	2
Armenia	1
…	

Task 2. (17 marks)

1. (12 marks) Developing an Apache Pig script that outputs like below:

Region	Gold	Silver
USA	8	1
Norway	5	2
Italy	3	1
Australia	2	
Egypt	2	5
Estonia	2	
Germany	2	1
Japan	2	1
Armenia	1	
…		

In the output, the columns `Region` represents the region_name, `Gold` represents the gold medal counts, and `Silver` represents the silver medal counts from left to right. The output is sorted by `Gold` in descending order, and if there is a tie, then sorted by `Region` in ascending order.

2. (5 marks) This is an advanced research task. It is an extension of Task 2-1 so that only attempt it after you have completed Task 2-1. You are asked to figure out how to develop Apache Pig User Defined Function (UDF) using Python. It is not instructed in detail in the learning materials, but you can learn by studying https://pig.apache.org/docs/latest/udf.html#udfs. The UDF is designed to improve the output of Task 2-1. The purpose is to add 0 to the position empty, for example,

“Australia 2 ” is changed to “Australia 2 0”
“Estonia 2 ” is changed to “Estonia 2 0”
“Armenia 1 ” is changed to “Armenia 1 0”

The output is like below

Region	Gold	Silver
USA	8	1
Norway	5	2
Italy	3	1
Egypt	2	5
Germany	2	1
Japan	2	1
Australia	2	0
Estonia	2	0
Colombia	1	4
Canada	1	1
France	1	1
Lebanon	1	1
Pakistan	1	1
Switzerland	1	1
Armenia	1	0
…		

In the output, the columns `Region` represents the region_name, `Gold` represents the gold medal counts, and `Silver` represents the silver medal counts from left to right. The output is sorted by `Gold` in descending order, if there is a tie, then sorted by `Sliver` in descending order, and if there is a tie, then sorted by `Region` in ascending order.

Submission

Your assignment should follow the requirements below and be submitted via Canvas > Assignment 2.
Failure to follow the requirements incurs a 5-mark penalty for each.
You will submit two files: one zip file and one pdf file.

- If your student ID is s1234567, then please create a zip file named s1234567_BDP_A2.zip. This zip file contains
    - task1.pig
    - task2-1.pig
    - task2-2.pig
    - task2udf.py, and
    - README.txt. In the README, specify sufficient information on how to run your codes for each task on AWS EMR.
    - Please note: all files are in the same folder (i.e., no subfolders), and then zip the folder.
- Besides the zip file, you must submit a pdf file contains the source code of your pig files and py file. This PDF file is for Turnitin plagiarism check.

Assessment declaration: When you submit work electronically, you agree to the assessment declaration.

Format Requirements

Failure to follow the requirements incurs a 5-mark penalty for each.
On HDFS, the input files must be in /input/ and the output must be in /output/, as follows:

```
/input/noc_region.csv
/input/person_region.csv
/input/person.csv
/input/competitor_event.csv
/input/medal.csv
/output/task1
/output/task2-1
/output/task2-2
```

Functional Requirements

Failure to follow the requirements incurs up to a 5-mark penalty.
The code must include sufficient comments that can clearly explain the major logic flow of the program.

Academic integrity and plagiarism (standard warning)

Academic integrity is about the honest presentation of your academic work. It means acknowledging the work of others while developing your own insights, knowledge, and ideas. You should take extreme care that you have:

- Acknowledged words, data, diagrams, models, frameworks, and/or ideas of others you have quoted (i.e., directly copied), summarized, paraphrased, discussed, or mentioned in your assessment through the appropriate referencing methods.
- Provide a reference list of the publication details so your reader can locate the source if necessary. This includes material taken from Internet sites.

If you do not acknowledge the sources of your material, you may be accused of plagiarism because you have passed off the work and ideas of another person without appropriate referencing, as if they were your own.

RMIT University treats plagiarism as a very serious offense constituting misconduct. Plagiarism covers a variety of inappropriate behaviours, including:

- Failure to properly document a source.
- Copyright material from the internet or databases.
- Collusion between students.

For further information on our policies and procedures, please refer to
https://www.rmit.edu.au/students/student-essentials/rights-and-responsibilities/academic-integrity

Marking Guide

- Late submission results in a penalty of 10% marks for (up to) every 24 hours being late.
- If unexpected circumstances affect your ability to complete the assignment, you can apply for special consideration.
  - Requests for special consideration within 7*24 hours, please email the course coordinator directly with supporting evidence.
  - Request for special consideration of more than 7*24 hours must be via the University Special consideration: https://www.rmit.edu.au/students/student-essentials/assessment-and-exams/assessment/special-consideration.

Task 1	0 marks – script cannot run on Hadoop – no output – no field is correct – completely incorrect script	1-2 mark – output has one field(s) incorrect. – Output misses some fields or include more fields. – The tuples in the output are not ordered as required. – the script misses one or more necessary operators like group by, order by, join, and foreach. – Comments and readme are misleading and hard to follow.	3-4 marks – Output has the same fields as specified, and they are correct in general with obvious errors. – The script includes all necessary operators like group by, order by, join, and foreach but with obvious errors. – Comments and readme have various minor issues.	5-6 marks – Output has the same fields as specified, and they are correct in general but with minor errors. – The script includes all necessary operators like group by, order by, join, foreach but with minor errors. – Comments and readme have no obvious issues.	7-8 marks Correct output with clear, concise comments and readme.
Task 2-1	0 marks – script cannot run on Hadoop – no output – no field is correct – completely incorrect script	1-4 marks – output has one field(s) incorrect. – Output misses some fields or include more fields. – The tuples in the output are not ordered as required. – the script misses one or more necessary operators like group by, order by, join, and foreach. – Comments and readme are misleading and hard to follow.	5-7 marks – Output has the same fields as specified, and they are correct in general with obvious errors. – The script includes all necessary operators like group by, order by, join, and foreach but with obvious errors. – Comments and readme have various minor issues.	8-10 marks – Output has the same fields as specified, and they are correct in general but with minor errors. – The script includes all necessary operators like group by, order by, join, foreach but with minor errors. – Comments and readme have no obvious issues.	11-12 marks Correct output with clear, concise comments and readme.
Task 2-2	0 marks – script cannot run on Hadoop – no output – no field is correct – No UDF defined – completely incorrect script	1 mark – Output has one field(s) incorrect. – output misses one field(s) or include more fields. – The tuples in the output are not ordered as required. – The script misses one or more necessary operators like group by, order by, join, and foreach. – UDF defined but used but with errors. – Comments and readme are misleading and hard to follow.	2 marks – Output has the same fields as specified, and they are correct in general with obvious errors. – The script includes all necessary operators like group by, order by, join, and foreach but with obvious errors. – UDF defined and used but with minor errors. – Comments and readme have various minor issues.	3-4 marks – Output has the same fields as specified, and they are correct in general but with minor errors. – The script includes all necessary operators like group by, order by, join, foreach but with minor errors. – UDF defined and used correctly. – Comments and readme have no obvious issues.	5 marks Correct output with clear, concise comments and readme.
Functional requirement	Failure penalty on functional requirements detailed in the specification				
Format requirement	Failure penalty on format requirements detailed in the specification				