RMIT UNIVERSITY

Database Systems
COSC 2406/2407
Assignment 1

Assessment Type	Individual assignment. Submit online via Canvas→Assignments→Assignment 1. Marks awarded for meeting requirements as closely as possible. Clarifications/updates may be made via announcements/relevant discussion forums.
Due Date	Week 6, Tuesday 5 April 2022, 11:59pm
Marks	100 points (20% of the overall assessment)

1. Overview

You will use the AWS Linux instance assigned to you and the data from a public source to complete the following tasks:

1. implement in Java a heap file to store the data;
2. implement in Java a query on the data;
3. store and query the data in an Apache Derby relational database that you create, and
4. store and query the data in a MongoDB database that you create.

In the second assignment, you will extend your solution developed in this assignment and conduct further timing experiments on your AWS Linux instance with indexes.

2. Learning Outcomes

This assessment relates to the following learning outcomes of the course:

- CLO 1: Explain and critique data structures and algorithms used to efficiently store and retrieve information in database systems,
- CLO 2: Evaluate, critically analyse and compare alternative designs for implementation of database systems, including data models, file structures, index schemes, and query evaluation, and
- CLO 4: Design, implement and report on significant software components of a database system (such as file structures and index schemes) according to analysis of requirements and specified constraints.

3. Submission

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

https://www.rmit.edu.au/students/student-essentials/assessment-and-exams/assessment/assessment-declaration

Submit on Canvas: Assignments > Assignment 1. You MUST submit:

- a zip file of your code for tasks 1 and 2 (all Java sources files including your git log); and
- your report (a single PDF file) that explains your approach and answers for each task (1, 2 and 3), including description of any scripts for data pre-processing, queries you used, and output.

Walk-through of your Java code in Week 4 You must undertake a walk-through during a scheduled lab class in week 4 explaining your Java code for Task 1 and answering questions about it. Failing to do this result in a penalty of 10 points.

Late submission:

- After the due time, you will have 7*24 hours to submit your assignment as a late submission. Late submissions follow the same procedure but will be penalised by 10 points for each (up to) 24 hours being late. For assignments that are more than 7*24 hours late, zero points will be awarded.
- Tasks (Section 4) in this assignment require you run database systems and timing experiments in a cloud(AWS) Linux instance. All experiments and database systems must be shut down before you log out of your Linux instance. Otherwise your instance may run out of memory. Other precautions are provided via announcements on Canvas. *Ignoring such precautions will NOT be grounds for extension to due time.*

4. Academic integrity and plagiarism (standard warning)

Academic integrity is about 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), summarised, paraphrased, discussed or mentioned in your assessment through the appropriate referencing methods,
- Provided 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 offence 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

5. Marking Guidelines

Task 1 Heap file implementation: 40/100

Task 2 Range query implementation: 30/100

Task 3 Derby database and queries: 15/100

Task 4 MongoDB database and queries: 15/100

Assessment details

Data: The data that you are going to use in this assignment is available from:
https://canvas.rmit.edu.au/courses/90603/assignments/679757
and you can download the file on another machine and use scp to copy to your AWS Linux instance (you may need to temporarily store files in the temp directory on titan if you are doing this outside of RMIT).
The first four lines are not data but headers for you to create and name fields in the databases for Tasks 3 and 4.

Run database systems and timing experiments
Timing experiments are only reliable when running one program at a time. You should only run one database system, Derby or MongoDB, at a time and shut down one database system before starting the other. All database systems must be shut down before you run your own program and before you log out of your Linux instance. Otherwise, your instance may run out of memory. Other precautions are provided via announcements on Canvas. *Ignoring such precautions will NOT be grounds for extension to due time.*

Walkthrough of your Java code in Week 4 You must undertake a walk-through during a scheduled lab class in week 4 explaining your Java code for Task 1 and answer questions about it. Failing to do this will result in a penalty of 10 points.

Code and git log: You must use git to track your assignment code. You need to set up your git repository so that each commit identifies you with your full name as per course enrolment and your student email address. It sets an expectation of professionalism. You must submit a text file of your git log. Do not include the git repository in your code submission.

Report: Create a file called report.pdf (various software including Word processors can export as PDF). Use this file to report on the following four tasks. Each task should be reported under a separate heading with the task name and description, for example for the first task use the heading: Task 1: Derby. *Limit your report to three pages.*

- Description of scripts for data preprocessing for loading MongoDB and Derby databases, rather than the scripts themselves, should be included in the report**.** Scripts are submitted with your code in the zip file to enable markers to test your scripts.
- Description of query output, rather than a long list of output, should be included in the report.

Task 1: Implement a Heap File in Java (40 points)

You are ONLY going to use the following 11 fields of the source dataset (field names are in first line of the file, with the exception of personName, the second column, which is called rdf-schema#label):

personName STRING	instrument label STRING
birthDate DATE	nationality label STRING
birthPlace label STRING	thumbnail STRING
deathDate DATE	wikiPageID INTEGER
field label STRING	description STRING
genre label STRING	

Write a program in Java to store and organise the data in a heap file using Java on your AWS Linux instance.

The source records are variable-length. Your heap file may hold fixed-length records (you will need to choose appropriate maximum lengths for each field). However, you may choose to implement variable lengths for some fields, especially if you run out of disk storage space or RAM.

All attributes with `Int` type must be stored in 4 bytes of binary, e.g. if the value of ID is equal to 70, it must be stored as 70 (in decimal) or 46 (in hexadecimal; in Java: 0x46). It must not be stored as the string "70", occupying two bytes. Your heap file is therefore a binary file. For simplicity, the heap file does not need a header (containing things like the number of records in the file or a free space list), though you might need to keep a count of records in each page. The file should be packed, i.e. there is no gap between records, but there may be gaps at the end of each page.
The executable name of your program to build a heap file must be `dbload` and should be executed using the command:
`java dbload -p pagesize datafile`
The output file will be `heap.pagesize` where your converted binary data is written as a heap. Where 'pagesize' means the actual page size in bytes, eg: 1024, 2048, 4096, etc.

Your program should write out one "page" of the file at a time. For example, with a pagesize of 4096, you would write out a page of 4096 bytes possibly containing multiple records of data to disk at a time. You are not required to implement spanning of records across multiple pages. Your `dbload` program must also output the following to `stdout`, the number of records loaded, number of pages used and the number of milliseconds to create the heap file. You are also suggested the use of utilities like xxd for examining the output heap file to see if their code is producing the expected format:

```
xxd heap.pagesize | less
```

In your report, explain how your code implements a heap file. Give detailed instructions to test your heap file and show that it includes all records from the data source.

Task 2. Implement a range query on your heap file (30 points).

Write a program to perform range query search operations on the field "birthDate" heap file (without an index) produced by your dbload program in Task 1. Your program must be named birthDate with two date values as input. For example for the query "List all artists born in 1970". The command below execute this query, where the integer 19700101 is for the date "1st January 1970":

```
java birthDate 19700101 19701230
```

Your program should read in the file, one "page" at a time. For example, if the pagesize parameter is 4096, your program should read in the first records from disk. These can then be scanned, in-memory, for a match. If a match is found, print the matching record to stdout, there may be multiple answers. If no match is found, read in the next pagesize records of the file. The process should continue until there are no more records in the file to process. In addition, the program must always output the total time taken to do all the search operations in milliseconds to stdout.

In your report, explain how your code implements the range query, provide the query, a description of returned results and timings Give instructions to test your program.

Task 3: Derby database and queries (15 points)

You are required to load the data into Derby. In your report:

- Explain how have you chosen to structure the Derby relational database and give reasons.
- Provide details of the time to load the data into Derby. You need to analyse the data and consider appropriate ways to structure the data and then using any scripting, programming or other tools to format the data accordingly.
- In your report you are expected to provide reasoning for the way you have structured the data in each database.
- *Postgraduate students only:* What alternative way or ways could you have organised the data when storing in Derby, and what advantages or disadvantages would these alternative designs have?

Undertake the following queries (include timings when you run the queries). Run each query twice and compare the times (first after rebooting the machine and then again after the query has already been run once).

- How many artists are in the movement known as Post-Impressionism?
- Which punk rock musicians were born in Australia?

In your report provide the queries, description of answers returned by Derby and timings. Explain differences in timings.

Task 4: MongoDB database and queries (15 points)

You are required to load the data into MongoDB. In your report:

- Explain how you have chosen to structure the data inserted in MongoDB.
- Provide details of the time taken to load the data (mongoimport is one utility that will provide such information). A naive import into a flat structure in Mongodb will not accrue you a great mark. You need to analyse the data and consider appropriate ways to structure the data and then use any scripting, programming or other tools to format the data accordingly.
- *Postgraduate students only:* What alternative way or ways could you have organised the data when storing in MongoDB, and what advantages or disadvantages would these alternative designs have?

Undertake the following queries (include timings when you run the queries). Run each query twice and compare the times (first after rebooting the machine and then again after the query has already been run once).

- How many artists are in the movement known as Post-Impressionism?
- Which punk rock musicians were born in Australia?

In your report provide the queries, details of the answers returned by MongoDB and query timings. Explain differences in timings.