DBMS Models and implementation
Instructor: Sharma Chakravarthy
Project 3: Map/Reduce Programming Exercise and Analysis

Made available on:	**11/03/2022**
Submit and Demo by:	**12/6/2022 (11:59 PM) No extension for this project (last day of classes)**
Submit to:	**Canvas (1 zipped folder containing all the files/sub-folders) https://elearn.uta.edu/**
Weight:	**15% of total**
Total Points:	**100**

One of the advantages of cloud computing is its ability to deal with very large data sets and still have a reasonable response time. Typically, the map/reduce paradigm is used for these types of problems in contrast to the RDBMS approach for storing, managing, and manipulating this data. An immediate analysis of a large data set does not require designing a schema and loading the data set into an RDBMS. Hadoop is a widely used open source map/reduce platform. Hadoop Map/Reduce is a software framework for writing applications which process vast amounts of data in parallel on large clusters. In this project, you will use the IMDB (International Movies) dataset and develop programs to get interesting insights into the dataset using Hadoop map/reduce paradigm. Please use the following links for a better understanding of Hadoop and Map/Reduce (https://hadoop.apache.org/docs/stable/hadoop-mapreduce-client/hadoop-mapreduce-client-core/MapReduceTutorial.html)

1.

i. SDSC Expanse M/R system

You will be using the SDSC Expanse system for your project. Separately, you will be given instructions for creating accounts, using the system for this project, etc. This is a facility supported by NSF for educational usage. There is a limit on the resources available. Keep that in mind while you do the project.

You can install Hadoop on your laptop/desktop for developing and testing the code before you run it on Expanse. This is for your convenience. For that you may have to install WSL 1 or 2 if you are using a windows machine. We may provide some instructions for that. Please look up and install Hadoop if you plan to do that.

ii. Local Hadoop system

If you want to install on your laptop/desktop, you are welcome to do so for developing code and testing. Submitted code MUST be from i. above and run on that machine. Local code execution will NOT be acceptable for this project.

2. IMDB Dataset

IMDB is a data set that contains information about movies (international) and TV episodes from their beginnings. The information includes movie titles, directors, actors, genre, and year produced/started. Some rating information is also included. The same is true for TV episodes

and includes number of seasons in terms of start and end years as well as episodes in each season. It is quite large and should not require additional information to understand the data set. The data that you will use is given below. Here are the 3 files you will be using for this project. You can download them by clicking on the link (zip files)

1. imdb00.title.basics.zip (tab separated file .tsv), 689.7MB, 9,099,385 lines
2. imdb00.title.actors.zip (comma separated file .csv), 682.8 MB, 20,114,478 lines
3. imdb00.title.crew.zip (tab separated file .tsv), 290.3 MB, 9.101,808 lines

i. imdb00.title.basics.tsv

This dataset contains the information about the various **IMDb titles** (movies, tv episodes, documentaries etc.), produced across the world. Each field in this dataset is separated by a tab. A random sample from this file has been shown below

tt0000091	short	The House of the Devil	1896	Horror,Short
tt0468569	movie	The Dark Knight	2008	Action,Crime,Thriller
tt0088610	tvSeries	Small Wonder	1985	Comedy,Family,Sci-Fi

The description of the various fields has been given below

Field Number	Field Name	Field Description
1	TITLE ID	The 9-digit unique IMDB title identifier attached to every entry, example: *tt0000091*
2	TITLE TYPE	Every IMDB title in the file is categorized as one of the following TITLETYPEs • *tvSpecial* • *tvMovie* • *tvShort* • *short* • *tvEpisode* • *videogame* • *movie* • *tvSeries* • *tvMiniSeries* • *video*
3	TITLE NAME	The name of the IMDB Title, example: *The Dark Knight*
4	YEAR	The year of release, example: *2008*
5	GENRE LIST	Multiple genres can be attached to a particular title, and they are separated by **comma,** example: *Action,Crime,Thriller*

ii. imdb00.title.actors.csv

This dataset contains the information about the **actors from each IMDB title**. Each field in this dataset is separated by a comma. A random sample from this file has been shown below

tt1410063,nm0000288,Christian Bale
tt1429751,nm0004266,Anne Hathaway
tt1872194,nm0000375,Robert Downey Jr.

The description of the various fields has been given below

Field Number	Field Name	Field Description
1	TITLE ID	The 9-digit unique IMDB title identifier attached to every entry, example: *tt0000091*
2	ACTOR ID	The 9-digit unique actor identifier, example: *nm0000288*
3	ACTOR NAME	The name of the actor, example: *Christian Bale*

iii. imdb00.title.crew.tsv

This dataset contains the information about the **directors for each IMDB title**. Each field in this dataset is separated by a tab. A random sample from this file has been shown below. **Note that there may be more than 1 director for a title.**

tt0000091 nm0617588 nm0617588
tt0000092 nm0617588 \N
tt0000093 nm0525910,nm0525908 \N
tt0000094 nm0617588 \N

The description of the various fields has been given below

Field Number	Field Name	Field Description
1	TITLE ID	The 9-digit unique IMDB title identifier attached to every entry, example: *tt0000091*
2	DIRECTORS	The 9-digit unique director identifier, example: *nm5387279* Comma separated if more than 1
3	WRITERS	The 9-digit unique writer identifier, example: *nm5387279* Comma separated if more than 1

3. Project 3 Problem Specification: You need to compute the following for the given data using the map/reduce paradigm. Try to compare and understand how you would do it using RDBMS if these files were stored as relations. This may help you understand when map/reduce is meaningful and when to use a RDBMS.

i. **[PROJECT 3 – 100 points]** There are many instances when a person who acts in a movie, tvEpisode, or tvMovie also directs it. For example, *Ben Affleck directed and starred in the 2012 movie Argo*. You need to compute actors (or actresses) who also as acted directors for the parameters given to you. You will compute it for 3 5-year periods for comparison. You will be given one of {movie, tvEpisode, tvMovie}. **Write a Map/Reduce program to list the names of people, who have directed and acted in the same IMDb title/category of any genre and output title, director, actor, genre, and year. All datasets are needed for this.**

You need to do this with multiple partitions (shards) of input (at least 4 for title_basics input) even if the total number of mappers used is less than the number of partitions (or shards). You need to analyze the response time of the alternatives.

Hint: This problem corresponds to a typical SQL query which has joins, group by and having clauses. The purpose of this bonus problem is to understand how some of the relational computations can be performed using the map/reduce paradigm. You can also write an SQL for this and run it on the Omega Oracle IMDb database that has been setup.

You will need use two configs: i) 3 mappers (one for each input) and 1 reducer and ii) 3 mappers (one for each input) and 2 reducers solve this problem.

Understand the difference between mappers and mapper tasks! They are not the same.

ii. **[PROJECT 3 BONUS PROBLEM – 5%]** For bonus, you need to increase the number of mappers for each input. You should also use more than one or two reducers. This exercise will allow you to understand the effect of distributed and parallel computation better. If you do this, you can compare the response time for analysis i) with this response time and do an analysis on that. You are free to choose the number of mappers and reducers and justify them. You can play with increasing only the mappers and then increasing the number of reducers to see the effects.

You need to try at least the following configurations:

- increase the number of mappers for each input (at least 2 alternatives) and 1 reducer
- Keep the number of mappers used above and for each increase the number of reducers by 1

You need to design and develop a map program (including a combiner if needed) and a reduce program to solve the above problems. If you need setup and cleanup phases, you need to develop code for them as well. The most important aspects of this design will be to identify the **<key, value>** pairs to be output by the mapper and computations in the reducer to produce the desired final output.

5% (i.e., 5 absolute points if you get 100) may result in improving by a grade (if you are on the border). Average will be calculated without using the bonus but grading will be done using the bonus.

4. Project Report: Please include (at least) the following sections in a **REPORT.{txt, pdf, doc}** file that you will turn in with your code:

i. **Overall Status**

Give a *brief* overview of how you implemented the major components. If you were unable to finish any portion of the project, please give details about what is completed

and your understanding of what is not. (This information is useful when determining partial credit.)

ii. **Approach:** for i) logic of mapper code, combiner code (if present), and reducer code. Why they work and produce what you are looking for. For ii) how and why you chose the number of mappers/reducers you did, justification behind them, and what you expected to see and what you actually saw.

iii. **Analysis:**
Time taken for i) and what you learnt from that. Compare the performance (response time) of the effect of increasing partitions. If you did bonus, how performance (total time taken) changed (improved/not-improved) when the number of mappers is increased and When the number of reducers increased. Analysis of different numbers of mappers and reducers.

iv. **File Descriptions**
List the files you have created and *briefly* explain their major functions and/or data structures.

v. **Division of Labor**
Describe how you divided the work, i.e. which group member did what. Please also include how much time each of you spent on this project. (This has no impact on your grade whatsoever; we will only use this as feedback in planning future projects -- so be honest!)

vi. **M/R configuration details for multiple inputs and other details**:
What libraries and packages you have used for dealing with multiple inputs.

5. What to submit:

- After you are satisfied that your code does exactly what the project requires, you may turn it in for grading. Please submit your project report with your project.
- You will turn in one zipped file containing **a) source code, b) outputs from the M/R code for each configuration, log files, and c) raw analysis results (spreadsheets etc.) as well as the d) report. This is for each configuration.**
- All of the above files should be placed in a single zipped folder named as - 'Fall_2022_**proj3_team_<*TEAM_NO*>**'. **Only one zipped folder should be uploaded using canvas.**
- You can submit your zip file at most 3 times. The latest one (based on timestamp) will be used for grading. So, be careful in what you turn in and when!
- **Only one person per group should turn in the zip file!**
- **Late Submissions not allowed**
- **Project 3 and the bonus part need to be submitted separately on Canvas! Use the name 'Fall_2022_bonus_proj3_team_<team_no>' for the bonus part submission.**

6. Coding style:

Be sure to observe the following standard Java naming conventions and style. These will be used across all projects for this course; hence it is necessary that you understand and follow them correctly. You can look this up on the web. Remember the following:

i. Class names begin with an upper-case letter, as do any subsequent words in the class name.
ii. Method names begin with a lower-case letter, and any subsequent words in the method name begin with an upper-case letter.
iii. Class, instance and local variables begin with a lower-case letter, and any subsequent words in the name of that variable begin with an upper-case letter.
iv. No hardwiring of constants. Constants should be declared using all upper case identifiers with _ as separators.
v. All user prompts (if any) must be clear and understandable
vi. Give meaningful names for classes, methods, and variables even if they seem to be long. The point is that the names should be easy to understand for a new person looking at your code
vii. Your program is properly indented to make it understandable. Proper matching of if … then … else and other control structures is important and should be easily understandable
viii. Do not put multiple statements in a single line

In addition, ensure that your code is properly documented in terms of comments and other forms of documentation for generating meaningful Javadoc.

7. Grading scheme:

The **PROBLEM 1** will be graded **out of 100** using the following scheme:

a.	Correctness of the Code	**30**
b.	Correctness of the Output for various partitions and configs	**20**
c.	Report	**20**
	i. Analyzing 3 mapper results	
	ii. Analysis of partitions and their response time	
d.	Answering questions during demo	**30**

The **BONUS PROBLEM** will be graded **out of 100** using the following scheme:

e.	Number of configurations used and their justification	**20**
f.	Correctness of the Code	10
g.	Correctness of the Output	**20**
h.	Report	**20**
	i. Analysis of configurations results - *15*	
	- **Analysis Results with graphs, plots etc., wherever necessary**	
	ii. Overall Completeness of report – *5*	
	- **Status, File Descriptions, Division of Labor, Logical errors**	
	iii. Answering questions during demo	**30**

8. Project 3 Demonstrations: Mandatory Team-wise demos will be scheduled after the last day of classes. A sign-up sheet for the demo Schedule will be provided. If you finish early, you are welcome to demonstrate early by sending the TA an email.

9. Additional information about registration and sample program can be downloaded from F22-supplementary-material.zip (1 zip file and 2 .docx files)

10. Please enter your ACCESS username into the google spread sheet before Nov 10th 2022 https://docs.google.com/spreadsheets/d/1zznI753pe_RaQivYKih_ZsAKiDtCMZXP0Dzbm3qoxRE/edit?usp=sharing