Project Report on

Data Analysis using Map-Reduce + Query Processing

(CSE 5331 – PROJECT 3)

Submitted by: Team 07

Sanket Rajendrakumar More (1002026812)

Mrunmai Nitin Magar (1002092125)

Shreya Manishkumar Patel (1002026812)

TABLE OF CONTENTS

OVERALL STATUS:

Task 1:

- The setup function initializes ‘genrePeriod’ with the genre combinations associated with specific time periods. This helps to identify which periods and genre combinations to look for in the given data.
- The process starts by parsing each line of input data, splitting it by semicolons to extract fields like title type, year, rating, and genres. It checks that the movie's year and rating are valid and that the rating meets a minimum of 7.5. If the title is a "movie" with a sufficient rating, the algorithm assesses each genre-period combination. It verifies if the movie's release year is within the period and if its genres match the required set for that combination. If both conditions are met, a key-value pair is emitted to the context, where the key is the period and genre combination (genres separated by semicolons) and the value is 1, denoting a single occurrence.
- The containsAllGenres function is a helper method designed to verify that a movie's genre list (movieGenres) encompasses all genres specified in the requiredGenres array. It utilizes Java streams to efficiently check that each genre listed in requiredGenres is present in movieGenres. This method ensures comprehensive genre matching, critical for filtering movies based on genre criteria.
- The reduce function takes a key representing a genre-period combination and a collection of integers (values), where each integer indicates an occurrence of that key as produced by a mapper. It aggregates these integers to calculate the total count of movies matching the key. Finally, it outputs the key alongside the summed total, presenting the final count for that genre-period combination.

Task2:

- The SQL query retrieves the top 5 movies from the IMDb database, meeting the criteria of having comedy and romance genres, released between 2001 and 2010, with at least 150,000 votes. It selects the movie title, average rating, and number of votes, and formats the output for clarity.
- We achieved this by first setting up the SQL environment with specific display settings using the SET commands. Then, we formatted the columns to ensure the output appears neatly.
- The SELECT statement retrieves the movie titles, ratings, and number of votes from the appropriate tables in the IMDb database. We filtered the results using the WHERE clause to include only movies with comedy and romance genres, released between 2001 and 2010, and with at least 150,000 votes. Finally, we sorted the results by rating in descending order and limited the output to the top 5 movies using the FETCH FIRST clause.
- With the help of 'EXPLAIN PLAN' statement, SQL statement is generated without actually executing the statement. It helps in understanding how Oracle will execute the query, including the sequence of operations and the estimated cost of each operation.
- We have used 'spool on' and spool off' to redirect the output of SQL commands to a file, that is 5331_Proj3Spring24_team_7.sql This allows you to save the results of SQL commands,

English queries, including query results and EXPLAIN PLAN output, to a file for later review or analysis.

ANALYSIS RESULT:

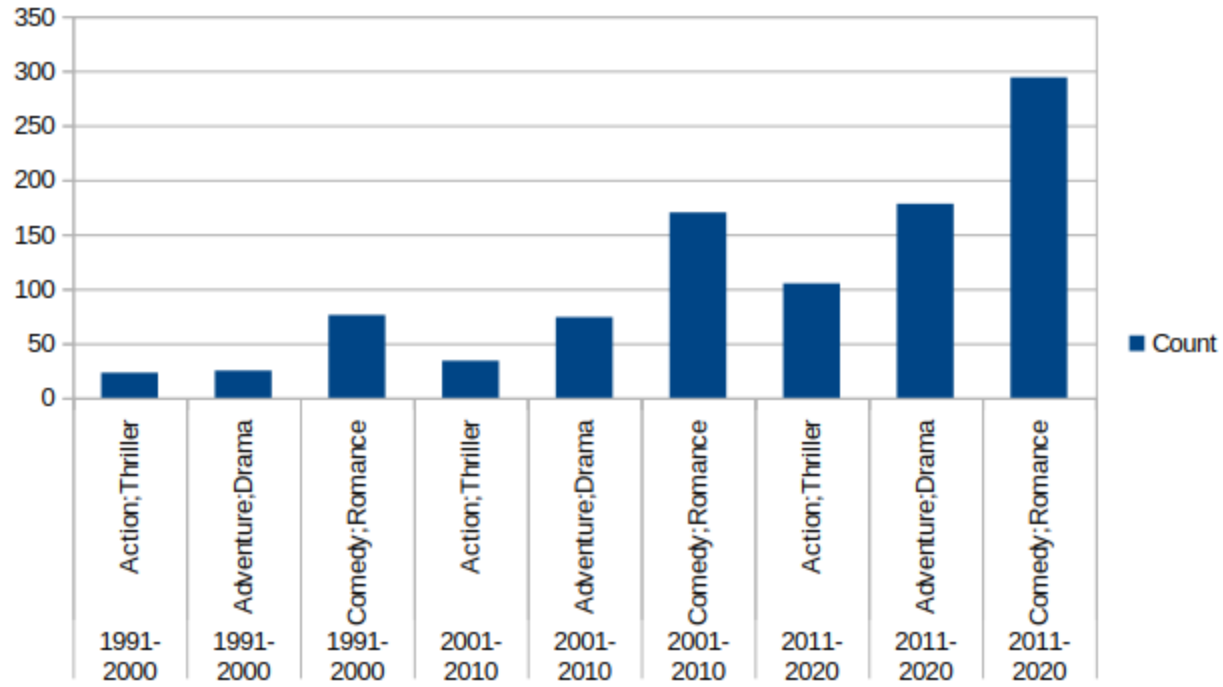
Task 1:

Output of the Hadoop Map-Reduce program:

```
shreya@shreya:~/Documents/hadoop-3.3.2$ bin/hadoop fs -cat /user/output2/part-r-00000
[1991-2000],Action;Thriller,23
[1991-2000],Adventure;Drama,25
[1991-2000],Comedy;Romance,76
[2001-2010],Action;Thriller,34
[2001-2010],Adventure;Drama,74
[2001-2010],Comedy;Romance,170
[2011-2020],Action;Thriller,105
[2011-2020],Adventure;Drama,178
[2011-2020],Comedy;Romance,294
```

Graph Analysis:

Period	Genre Combination	Count
1991-2000	Action;Thriller	23
1991-2000	Adventure;Drama	25
1991-2000	Comedy;Romance	76
2001-2010	Action;Thriller	34
2001-2010	Adventure;Drama	74
2001-2010	Comedy;Romance	170
2011-2020	Action;Thriller	105
2011-2020	Adventure;Drama	178
2011-2020	Comedy;Romance	294



This graph can show whether certain genres are becoming more or less popular over time. For example, if the line for "Comedy;Romance" shows an upward trend, it suggests increasing popularity.

Task 2:

SQL query output:

MOVIE_NAME	RATING	VOTES
Amelie	8.30	748048
500 Days of Summer	7.70	507857
Love Actually	7.60	476576
Sideways	7.50	190672
The Terminal	7.40	452046

EXPLAIN FOR query output:

```
PLAN_TABLE_OUTPUT
--------------------------------------------------------------------------------------------------------------
Plan hash value: 457476648

-----------------------------------------------------------------------------------------------------------
| Id  | Operation                          | Name              | Rows  | Bytes | Cost (%CPU)| Time     |
-----------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT                   |                   |     5 |  5270 |   135K  (1)| 00:00:06 |
|*  1 |  VIEW                              |                   |     5 |  5270 |   135K  (1)| 00:00:06 |
|*  2 |   WINDOW SORT PUSHED RANK          |                   |   276 |   280K|   135K  (1)| 00:00:06 |
|   3 |    VIEW                            |                   |   276 |   280K|   135K  (1)| 00:00:06 |
|   4 |     HASH UNIQUE                    |                   |   276 | 34776 |   135K  (1)| 00:00:06 |
|*  5 |      HASH JOIN SEMI                |                   |   276 | 34776 |   135K  (1)| 00:00:06 |
|   6 |       NESTED LOOPS                 |                   |   276 | 31740 |  3845   (1)| 00:00:01 |
|   7 |        NESTED LOOPS                |                   |  1380 | 31740 |  3845   (1)| 00:00:01 |
|*  8 |         TABLE ACCESS FULL          | TITLE_RATINGS     |  1380 | 23460 |  1084   (2)| 00:00:01 |
|*  9 |         INDEX UNIQUE SCAN          | SYS_C00547784     |     1 |       |     1   (0)| 00:00:01 |
|* 10 |        TABLE ACCESS BY INDEX ROWID| TITLE_BASICS      |     1 |    98 |     2   (0)| 00:00:01 |
|  11 |       TABLE ACCESS FULL            | TITLE_PRINCIPALS  |    51M|   538M|   131K  (1)| 00:00:06 |
-----------------------------------------------------------------------------------------------------------

Predicate Information (identified by operation id):
---------------------------------------------------

   1 - filter("from$_subquery$_009"."rowlimit_$$_rownumber"<=5)
   2 - filter(ROW_NUMBER() OVER ( ORDER BY INTERNAL_FUNCTION("from$_subquery$_008"."RATING")
              DESC )<=5)
   5 - access("TB"."TCONST"="TP"."TCONST")
   8 - filter("TR"."NUMVOTES">=150000)
   9 - access("TB"."TCONST"="TR"."TCONST")
  10 - filter("TB"."TITLETYPE"=U'movie' AND "TB"."STARTYEAR"<='2010' AND "TB"."GENRES" LIKE
              U'%Comedy%' AND "TB"."GENRES" LIKE U'%Romance%' AND "TB"."STARTYEAR">='2001' AND
              "TB"."GENRES" IS NOT NULL AND "TB"."GENRES" IS NOT NULL)
```

FILE DESCRIPTIONS:

Task 1:

Files created: Imdb.java

Description: The main functions are setup(), map(), containsAllGenres() and reduce(). These functions collectively give the number of movies for the given time period and genres combinations.

Task 2:

Files created: 5331_Proj3Spring24_Team_7.sql

Description: This file contains the English query, the corresponding SQL query, the EXPLAIN query and their outputs.

DIVISION OF LABOR

Sanket Rajendrakumar More: Task 1 – 3 weeks

Mrunmai Nitin Magar: Task 2 – 3 weeks

Shreya Manishkumar Patel: Task 1 – 3 weeks.