Group Member:(gac69, has175, dif24, yil210, lil131)

Step1: Convert .dat File to .csv File, Which Can be Recognized by Mongodb

See java program: DatToCSV.java

Strategy:

replace “,” with “#” , ” \” ” with” ^” , “::” with “,”

(Finally when we need it, we will change it back)

Step2: Open Mongodb Service

Command Line: mongo

Step3: Create Mongo Database and Collection

See java program: CreateMongoDB.java

Using java to link to localhost, and create database and collection.

Step4: Import .csv to target collection

Command Line:

mongoimport -d DA -c assignment2_movies --type csv --headerline (fileLocalUrl) (E.x.) (/Users/chengaoxiang/Desktop/18_Fall_Semester/Data_Analytics/Assignment/Assignment2/ml-10M100K/Movies.csv)

Step5: Running Query to Get Result

Using Java

Problem 1:

Program : query1.java Result : query1_result_MoviesRating

Strategy: Extracting data from mongodb_collection_movie and mongodb_collection_rating

Name[movieID] = movieName, aveRating[movieID] = Rating[movieID] / count[movieID]

For every movieID which count is not 0, output its aveRating.

Problem 2:

Program : query2.java Result : query2_result_SimilarUsers (I use three examples to test)

Strategy: Extracting data from mongodb_collection_ratings

{movie1, movie2,…} = { {user1,user2…} {user1, user5…}{…}…}

{user1, user2,…} = { {movie1,movie2…} {movie1, movie5…}{…}…}

For target user, return all movies he ratted, then for each of those movies, return users who have ratted it.

Tips: Since it has more than 1000w data, using list is not a good way to handle it, we choose to use hash function, used a linkedmultivaluemap to store information. And using hashset to deal with the duplicated result.

Problem 3:

Program : query3.java Result : query3_result_MovieGenresNumber

Strategy: Extracting data from mongodb_collection_movie

Extract Genres type, split with "|", using list store all genres, when contains, add new, when exists, arr[list.indexOf(genre)]++.

Problem 4:

Program : query4.java Result : query4,5,6_result

query4_result_TaggedMost

The movie gets most tags is: Pulp Fiction (1994), 308 times.

Problem 5:

Program : query4.java Result : query4,5,6_result

query5_result_RattedMost

The movie being ratted most time is : Pulp Fiction (1994), 34864 times

Problem 6:

Program : query6.java Result : query4,5,6_result

query6_result_YearMovieNumber

Year, MovieNumber

1915, 1

1916, 2

…,…

2007, 364

2008, 251

Sametime, lots of queries can be finished through this program, like movies ratted less than 10 times, and users ratting less than 10 times and so on.