Homework 6
Kaijun he

```
package homework6;


import java.io.BufferedReader;

import java.io.File;

import java.io.FileNotFoundException;

import java.io.FileReader;

import java.io.IOException;

import java.sql.*;

import java.text.DateFormat;

import java.text.SimpleDateFormat;

import java.util.Scanner;

import java.util.TimeZone;


public class query {


        public static void main( String[] args ) throws IOException, SQLException {



                DateFormat format = new SimpleDateFormat("dd/MM/yyyy HH:mm:ss");

                format.setTimeZone(TimeZone.getTimeZone("Etc/UTC"));

            Connection conn = null;

            Statement stmt = null;

            /*****************************************************************

            *  determine if the JDBC driver exists and load it...

            *****************************************************************/

            System.out.print( "\nLoading JDBC driver...\n\n" );

            try {

              Class.forName("oracle.jdbc.OracleDriver");

              }
```

```
            catch(ClassNotFoundException e) {
              System.out.println(e);
              System.exit(1);
              }

            /*******************************************************************
            *  establish a connection to the database...
            *******************************************************************/
            try {
              System.out.print( "Connecting to ACADPRD0 database...\n\n" );
              conn =
DriverManager.getConnection("jdbc:oracle:thin:@acadoradbprd01.dpu.depaul.edu:1521:ACADPRD0",
"KHE11", "cdm1886997");
              System.out.println( "Connected to database ACADPRD0..." );

              /*******************************************************************
              *  create an object by which we will pass SQL stmts to the database...
              *******************************************************************/
             stmt = conn.createStatement();
              }
            catch (SQLException se) {
              System.out.println(se);
              System.exit(1);
              }

            System.out.println("since if i run drop table and create table statement to database, it
reloads millions rows data for too long time, so i decide to comment my drop table and create table
codes");
                  /*    try {
                  String dropRatings = "DROP TABLE RATINGS;";
```

```
        stmt.executeUpdate(dropRatings);
        }
      catch (SQLException se) {
            System.out.println("rating table is not exist");
      }
    try {
                              String dropMovieGenres = "DROP TABLE MOVIEGENRES;";
        stmt.executeUpdate(dropMovieGenres);
        }
      catch (SQLException se) {
            System.out.println("moveigebres table is not exist");
      }
    try {
                              String dropMovies = "DROP TABLE MOVIES;";
        stmt.executeUpdate(dropMovies);


        }
      catch (SQLException se) {
            System.out.println("movies table is not exist");
      }
                    try {
                                    String dropUsers = "DROP TABLE USERS;";
                  stmt.executeUpdate(dropUsers);
                  }
                catch (SQLException se) {
                     System.out.println("users table is not exist");
                }

                      try {
```

Homework 6
Kaijun he

```
            String usersql = "CREATE TABLE USERS(\r\n" +
                                " USERID INT,\r\n" +
                                " GENDER VARCHAR2(5),\r\n" +
                                " AGECODE VARCHAR2(50),\r\n" +
                                " OCCIPATION VARCHAR2(50),\r\n" +
                                " ZIPCODE VARCHAR2(50),\r\n" +
                                " PRIMARY KEY(USERID))";
    stmt.executeQuery(usersql);
    System.out.println("users table created");
}
    catch (SQLException ex) {
        System.out.println("users table has been created");
    }
try {
                    String moviesql = "CREATE TABLE MOVIES (\r\n" +
                                        " MOVIEID INT,\r\n" +
                                        " TITLE VARCHAR2(200),\r\n" +
                                        " YEAR INT,\r\n" +
                                        " PRIMARY KEY(MOVIEID)\r\n";
                stmt.executeQuery(moviesql);
                System.out.println("movie table created");
            }
                catch (SQLException ex) {
                    System.out.println("movie table has been created");
                }
try {
                        String movieGenresql = "CREATE TABLE MOVIEGENRES (\r\n" +
                                        " MOVIEID INT,\r\n" +
                                        " GENRES VARCHAR2(100),\r\n" +
```

```
                                                    " PRIMARY KEY (MOVIEID, GENRES),\r\n" +
                                                    " FOREIGN KEY (MOVIEID) REFERENCES
MOVIES(MOVIEID))\r\n";
                                stmt.executeQuery(movieGenresql);
                                System.out.println("movieGenresql table created");
                            }
                                catch (SQLException ex) {
                                    System.out.println("movieGenresql table has been created");
                                }



                        try {
                                String Ratingsql = "CREATE TABLE RATINGS (\r\n" +
                                                " USERID INT,\r\n" +
                                                " MOVIEID INT,\r\n" +
                                                " RATING INT,\r\n" +
                                                " TIMESTAMPS TIMESTAMP,\r\n" +
                                                " PRIMARY KEY (USERID, MOVIEID),\r\n" +
                                                " FOREIGN KEY (MOVIEID) REFERENCES
MOVIES(MOVIEID))\r\n";
                            stmt.executeQuery(Ratingsql);
                            System.out.println("Rating table created");
                        }
                            catch (SQLException ex) {
                                System.out.println("Rating table has been created");
                            }
*/
```

```
        try {
          System.out.println("read users file and parse data");
                BufferedReader br = new BufferedReader(new FileReader(new
File("users.dat")));
                String st;
                String[] tableOne = new String[5];
                PreparedStatement updateUsers = conn.prepareStatement("Insert INTO
USERS(USERID, GENDER, AGECODE,OCCIPATION,ZIPCODE ) VALUES(?,?,?,?,?)");
                conn.setAutoCommit(false);
                while((st = br.readLine()) != null) {
                        tableOne = st.trim().split("::");
                        updateUsers.setInt(1, Integer.parseInt(tableOne[0]));
                        updateUsers.setString(2, tableOne[1]);
                        updateUsers.setString(3, transferAgeCode(tableOne[2]));
                        updateUsers.setString(4, transferOccupation(tableOne[3]));
                        updateUsers.setString(5, tableOne[4]);
                        updateUsers.executeUpdate();
                }
                conn.commit();


         }
                catch(SQLException m) {
                }
        try {
                System.out.println("read movie file and parse data into movies table and
moviegenres table");
```

```
                    BufferedReader br = new BufferedReader(new FileReader(new
File("movies.dat")));
                    String st;
                    String[] tableTwo = new String[3];
                    PreparedStatement updateMovies = conn.prepareStatement("Insert
INTO MOVIES(MOVIEID, TITLE, YEAR ) VALUES(?,?,?)");
                    PreparedStatement updateMovieGenres =
conn.prepareStatement("Insert INTO MOVIEGENRES(MOVIEID, GENRES) VALUES(?,?)");
                    conn.setAutoCommit(false);
                    while((st = br.readLine()) != null) {
                        tableTwo = st.trim().split("::");
                        updateMovies.setInt(1, Integer.parseInt(tableTwo[0]));
                        updateMovies.setString(2, tableTwo[1].substring(0,
tableTwo[1].length()-6));
                        updateMovies.setInt(3,
Integer.parseInt(tableTwo[1].substring(tableTwo[1].length()-5, tableTwo[1].length()-1)));
                        updateMovies.executeUpdate();
                        String[] genres = tableTwo[2].trim().split("\\|");
                        for(int i = 0; i < genres.length; i++) {

                        updateMovieGenres.setInt(1, Integer.parseInt(tableTwo[0]));

                        updateMovieGenres.setString(2, genres[i]);
                        updateMovieGenres.executeUpdate();
                        }

                    }
                    conn.commit();
                }
                    catch(SQLException m) {
```

```
                    }



            try {
                System.out.println("read Ratings file and parse data");
                    BufferedReader brTwo = new BufferedReader(new FileReader(new
File("ratings.dat")));
                    String st;
                    String[] tableThree = new String[4];
                    PreparedStatement updateRating = conn.prepareStatement("Insert
INTO RATINGS(USERID, MOVIEID, RATING,TIMESTAMP) VALUES(?,?,?,?)");
                    conn.setAutoCommit(false);
                    while((st = brTwo.readLine()) != null) {
                        tableThree = st.trim().split("::");
                        updateRating.setInt(1, Integer.parseInt(tableThree[0]));
                        updateRating.setInt(2, Integer.parseInt(tableThree[1]));
                        updateRating.setInt(3, Integer.parseInt(tableThree[2]));
                         long unix_seconds = Integer.parseInt(tableThree[3]);
                          Date date = new Date(unix_seconds*1000L);
                          SimpleDateFormat jdf = new SimpleDateFormat("yyyy-MM-dd
HH:mm:ss z");
                          jdf.setTimeZone(TimeZone.getTimeZone("GMT-6"));
                          String java_date = jdf.format(date);
                        updateRating.setString(4, java_date);
                        updateRating.executeUpdate();


                    }
                    conn.commit();
```

```
            }
            catch(SQLException e) {


            }

                System.out.println("study queries of number of female and male who have high rating
movies in 1993 and 2000");
            ResultSet rset = stmt.executeQuery("SELECT COUNT(RATINGS.RATING) AS
NUMBER_FEMALE_HIGH_RATING_IN_1993\r\n" +
                        "FROM MOVIES\r\n" +
                        "INNER JOIN RATINGS ON RATINGS.MOVIEID = MOVIES.MOVIEID\r\n" +
                        "INNER JOIN USERS ON USERS.USERID = RATINGS.USERID\r\n" +
                        "WHERE USERS.GENDER = 'F' AND MOVIES.YEAR = '1993' AND RATINGS.RATING
= '5'");
                          System.out.println("NUMBER_FEMALE_HIGH_RATING_IN_1993");
                          while( rset.next() ) {
                          System.out.println(
                                    rset.getString("NUMBER_FEMALE_HIGH_RATING_IN_1993") );
                          }
            ResultSet rsetTwo = stmt.executeQuery("SELECT COUNT(RATINGS.RATING) AS
NUMBER_MALE_HIGH_RATING_IN_1993\r\n" +
                                    "FROM MOVIES\r\n" +
                                    "INNER JOIN RATINGS ON RATINGS.MOVIEID =
MOVIES.MOVIEID\r\n" +
                                    "INNER JOIN USERS ON USERS.USERID = RATINGS.USERID\r\n" +
                                    "WHERE USERS.GENDER = 'M' AND MOVIES.YEAR = '1993' AND
RATINGS.RATING = '5'");
                          System.out.println("NUMBER_MALE_HIGH_RATING_IN_1993 ");
                          while( rsetTwo.next() )

System.out.println(rsetTwo.getString("NUMBER_MALE_HIGH_RATING_IN_1993") );
```

Homework 6
Kaijun he

```
                    ResultSet rsetThree = stmt.executeQuery("SELECT COUNT(RATINGS.RATING)
AS NUMBER_FEMALE_HIGH_RATING_IN_2000\r\n" +
                                        "FROM MOVIES\r\n" +
                                        "INNER JOIN RATINGS ON RATINGS.MOVIEID =
MOVIES.MOVIEID\r\n" +
                                        "INNER JOIN USERS ON USERS.USERID = RATINGS.USERID\r\n" +
                                        "WHERE USERS.GENDER = 'F' AND MOVIES.YEAR = '2000' AND
RATINGS.RATING = '5'");

System.out.println("NUMBER_FEMALE_HIGH_RATING_IN_2000");
                                          while( rsetThree.next() ) {
                                          System.out.println(

        rsetThree.getString("NUMBER_FEMALE_HIGH_RATING_IN_2000") );
                                          }
                                ResultSet rsetFour = stmt.executeQuery("SELECT COUNT(RATINGS.RATING)
AS NUMBER_MALE_HIGH_RATING_IN_2000\r\n" +
                                                    "FROM MOVIES\r\n" +
                                                    "INNER JOIN RATINGS ON RATINGS.MOVIEID =
MOVIES.MOVIEID\r\n" +
                                                    "INNER JOIN USERS ON USERS.USERID =
RATINGS.USERID\r\n" +
                                                    "WHERE USERS.GENDER = 'M' AND
MOVIES.YEAR = '2000' AND RATINGS.RATING = '5'");
                                          System.out.println("NUMBER_MALE_HIGH_RATING_IN_2000
");
                                          while( rsetFour.next() )

System.out.println(rsetFour.getString("NUMBER_MALE_HIGH_RATING_IN_2000") );
                                rset.close();
                                rsetTwo.close();
                                rsetThree.close();
                                rsetFour.close();
```

```
            stmt.close();
                        conn.close();
        }

 // this transferagecode function to modify data from given users dat file
        public static String transferAgeCode(String ageCode) {
                int age_code = Integer.parseInt(ageCode);
                if(age_code == 1) {
                        ageCode = "Under 18";
                }
                if(age_code == 18) {
                        ageCode = "18-24";
                }
                if(age_code == 25) {
                        ageCode = "25-34";
                }
                if(age_code == 35) {
                        ageCode = "35-44";
                }
                if(age_code == 45) {
                        ageCode = "45-49";
                }
                if(age_code == 50) {
                        ageCode = "50 -55";
                }
                if(age_code == 56) {
                        ageCode = "56+";
                }
```

```
            return ageCode;


    }
    // this function is used to change occupation by given occupation number
    public static String transferOccupation(String occup) {
            int op = Integer.parseInt(occup);
            if(op == 0) {
                    occup = "other";
            }
            if(op == 1) {
                    occup = "academic/educator";
            }
            if(op == 2) {
                    occup = "artist";
            }
            if(op == 3) {
                    occup = "clerical/admin";
            }
            if(op == 4) {
                    occup = "college/grad student";
            }
            if(op == 5) {
                    occup = "customer service";
            }
            if(op == 6) {
                    occup = "doctor/health care";
            }
            if(op == 7) {
                    occup = "executive/managerial";
```

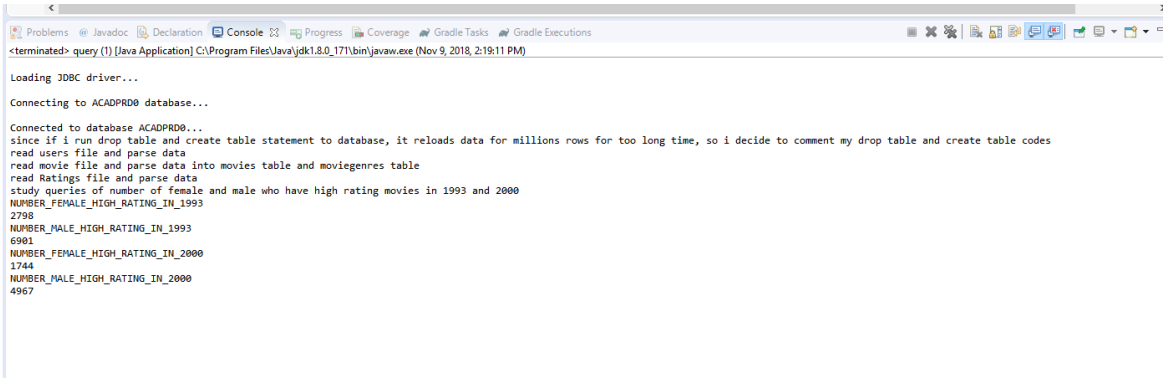
```
}
if(op == 8) {
        occup = "farmer";
}
if(op == 9) {
        occup = "homemaker";
}
if(op == 10) {
        occup = "K-12 student";
}
if(op == 11) {
        occup = "lawyer";
}
if(op == 12) {
        occup = "programmer";
}
if(op == 13) {
        occup = "retired";
}
if(op == 14) {
        occup = "sales/marketing";
}
if(op == 15) {
        occup = "scientist";
}
if(op == 16) {
        occup = "self-employed";
}
if(op == 17) {
```

Homework 6
Kaijun he

```
                occup = "technician/engineer";
            }
            if(op == 18) {
                occup = "tradesman/craftsman";
            }
            if(op == 19) {
                occup = "unemployed";
            }
            if(op == 20) {
                occup = "writer";
            }

            return occup;
        }

}
```



Homework 6
Kaijun he

Analysis of queries:

1. For users queries, I used preparedStatement to call insert statement, and I create two additional functions to transfer agecode and occupationscode from reading dat file
2. For movies table, I also used preparedStatement to call insert statement, but this time I will use two insert statement to insert into movies table and moviegenres table, and I use string split() function, substring and regex to separated given data into two created table
3. For rating table, I also use preparedStatement to call insert statement, difference between rating table and users table, this time I will transfer the long timestamp into data by using epoch time in Chicago time zone.
4. For interesting queries, I create studies queries of number of female and male with high ratings in 1993 and 2000. By using resultSet and statement call executeQuery(sql) function. And we find out the interesting results that the ratio of female and male with high ratings is almost same as 1:3 in 1993 and 2000.