ALLSTAR
Final Report

CS542 Spring 16 Team 2

Jingjun Zhang
Fan Yang
Bian Du

03-29-2016

ALLSTAR

Table of Contents

Introduction

How would you do to find next movie to watch at home? Do you have trouble deciding what films to see? Will you be upset when you are familiar with this movie but cannot tell the name? Do you want to know if the movie you watch won the awards?

According to result of our survey, we discovered some habits when people choosing new movie. First, a large number of people prefer Hollywood blockbusters that won significant awards like OSCAR, while some tend to choose new movies by chasing movie stars or directors who are more familiar to them; this habit also explains why box-office incomes of some movies are so high no matter how lousy they are, especially in China. Second, most people would like to choose the movies that gain high scores given by some famous websites like IMDB, and they will select films whose scores are higher than 8.0/10.0. Moreover, people need the information about relation of movies and stars to explore in movie world. For instance, we sometimes would like to know who the leading actor was after we watched a fantastic movie, which is another obvious requirement.

You will not worry about film selection when you have ALLSTAR! ALLSTAR is a desktop application proposed by Team 2 from CS542 class. This application aims at providing information of movies, stars and awards to help people finding movies they could be interested in.

Software information

Features

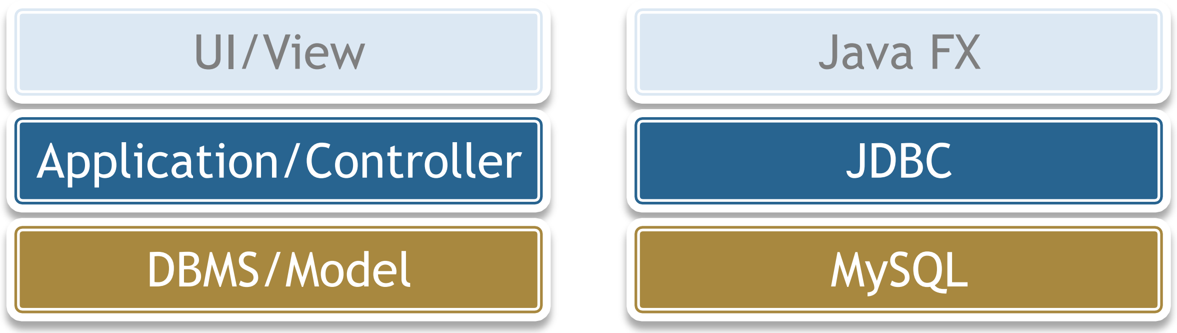
Our application has implemented the following features:

- List all stars that match information input by user, including first name, last name, date of birth, gender, nationality, age, the movie he or she participated, award he or she received, and the year when the award is received;
- When all the stars are found, all relevant information for each star will be displayed in the form;
- List all movies that match information input by user, including title, year of production, genre, language, studio of production, rating, stars in the movie, award and the year that the award is received;
- When all the movies are found, all relevant information for each movie will be displayed in the form;
- List top award winner in a list and a bar chart, and show them in a bar chart;
- List stars who participated in most movies, and show them in a bar chart;
- List directors who directed most movies, and show them in a bar chart;
- List most popular movie according to rating, and show them in a bar chart;
- List movies which received most awards, and show them in a bar chart;
- Add, delete and edit basic information of stars;
- Add, delete and edit basic information of movies;
- Add, delete and edit connections between stars and movies;
- Add, delete and edit connections between awards and stars, and adding ceremony if it does not exist;
- Add, delete and edit connections between awards and movies, and adding ceremony if it does not exist;

Software Architecture

This application consists of three major components:

- User interface at frontend: this component makes it possible to interact with user, providing information user request. We provide a GUI based on java FX. This part acts as view component in MVC architecture.
- Application at backend: User input is stored and processed in this part, and JDBC will be used to connect DBMS, transfer processed user input and retrieve information in database. This part acts as controller component in MVC architecture.
- DBMS at local or remote host: There is a DBMS and a corresponding database, which can store and present information about movies, stars and awards.



Software Developing Environment

- Database Server: MySQL Ver 14.14 Distrib 5.7.10
- MySQL Java Connecter: Ver 5.1.38
- Java Environment: Java 8 Update 73
- IDE: Netbeans 8.1
- UI Tool: Scene Builder 2.0

Developing Plan

The project kicked off on Feb 10th, and the following table presents our brief division of development cycle.

Stage	Deadline
Requirement Analysis	Feb 8 to Feb 10
Conceptual Design and Logical Design	Feb 10 to Feb 14
Coding	Feb 14 to Mar 20
Testing	Mar 20 to Mar 24

Currently we have finished the first version of ALLSTAR application. We had decomposed the whole projects into following tasks and assign owners for each task:

- Implement star searching feature, Jingjun Zhang;
- Implement movie searching feature, Jingjun Zhang;

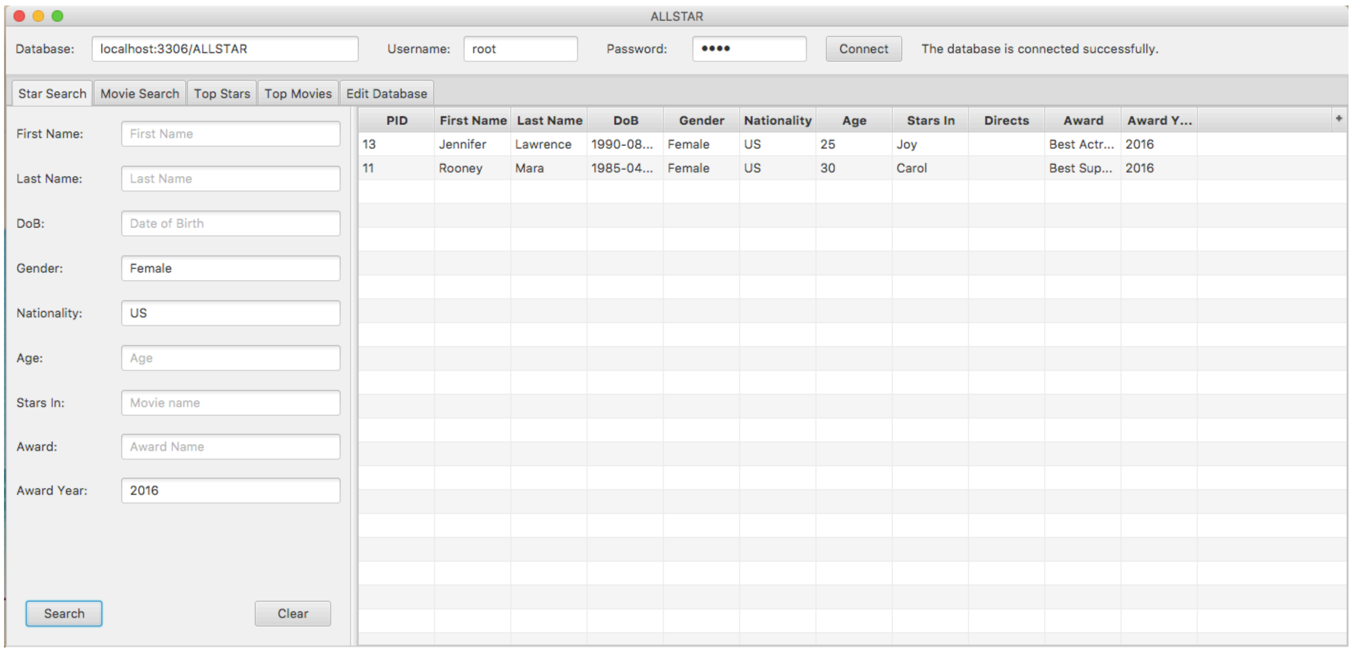
- Implement top star searching feature, Bian Du;
- Implement top movie searching feature, Fan Yang;
- Record insertion/deletion/modification feature, Jingjun Zhang;
- UI design and integration, Jingjun Zhang;
- Data collection from Internet, Bian Du;
- UI optimization, Fan Yang;
- System testing, Fan Yang, Bian Du;

UI Design

The whole UI design including four tabs and a login bar. User will be reminded to log into the database before input any information. After successfully login, users will be able to switch between five tabs to find the information they are interested in.

Star search

This first tab allows user to search for a set of stars by inputting several keywords, and ALLSTAR will return all relevant information about these stars in a table. For example, if you choose an US actress with award year 2016, the results will be shown as follows. All the contents satisfy the three conditions the user prefers.



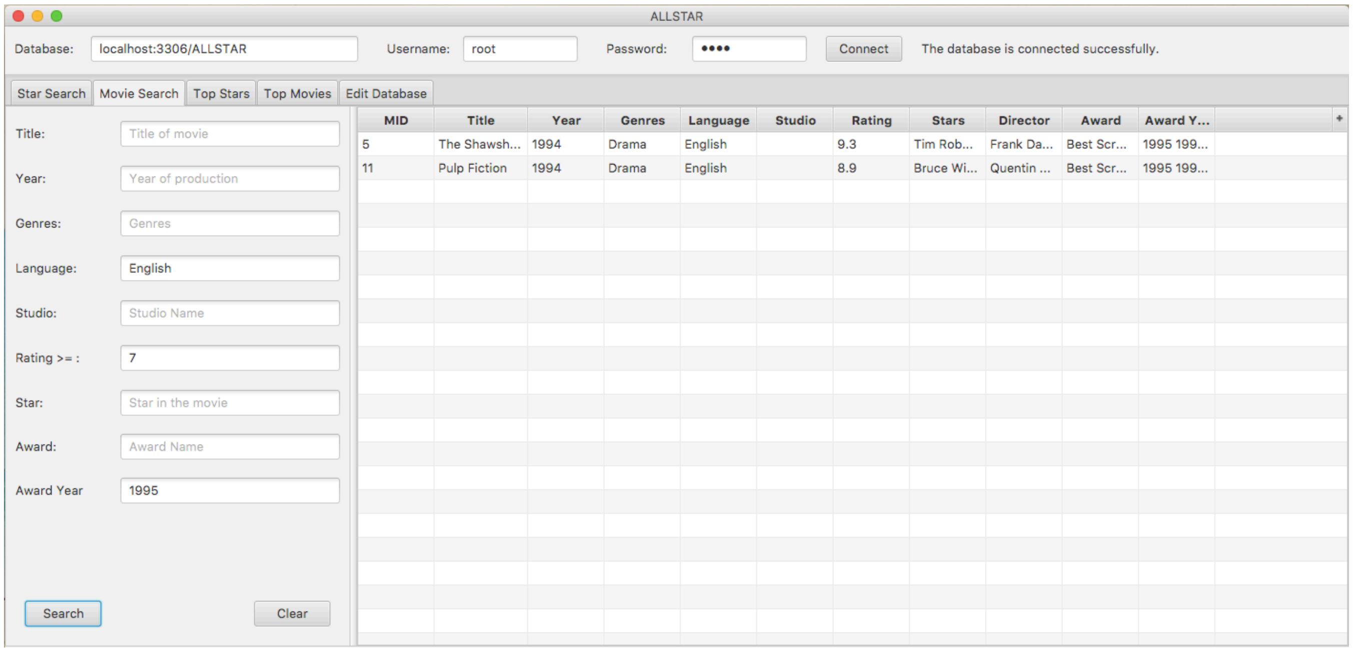
Database: localhost:3306/ALLSTAR Username: root Password: •••• The database is connected successfully.

First Name:	First Name
Last Name:	Last Name
DoB:	Date of Birth
Gender:	Female
Nationality:	US
Age:	Age
Stars In:	Movie name
Award:	Award Name
Award Year:	2016

PID	First Name	Last Name	DoB	Gender	Nationality	Age	Stars In	Directs	Award	Award Y...
13	Jennifer	Lawrence	1990-08...	Female	US	25	Joy		Best Actr...	2016
11	Rooney	Mara	1985-04...	Female	US	30	Carol		Best Sup...	2016

Movie search

The second tab allows user to search for movies after inputting several pieces of information. Similarly, a table consisting of relevant information of movies will be displayed. For example, If we want to watch an English movie with rating larger than 7 and award year 1995, the movies are “The Shawshank Redemption” and “Pulp Function”.



Database:	localhost:3306/ALLSTAR
Username:	root
Password:	••••

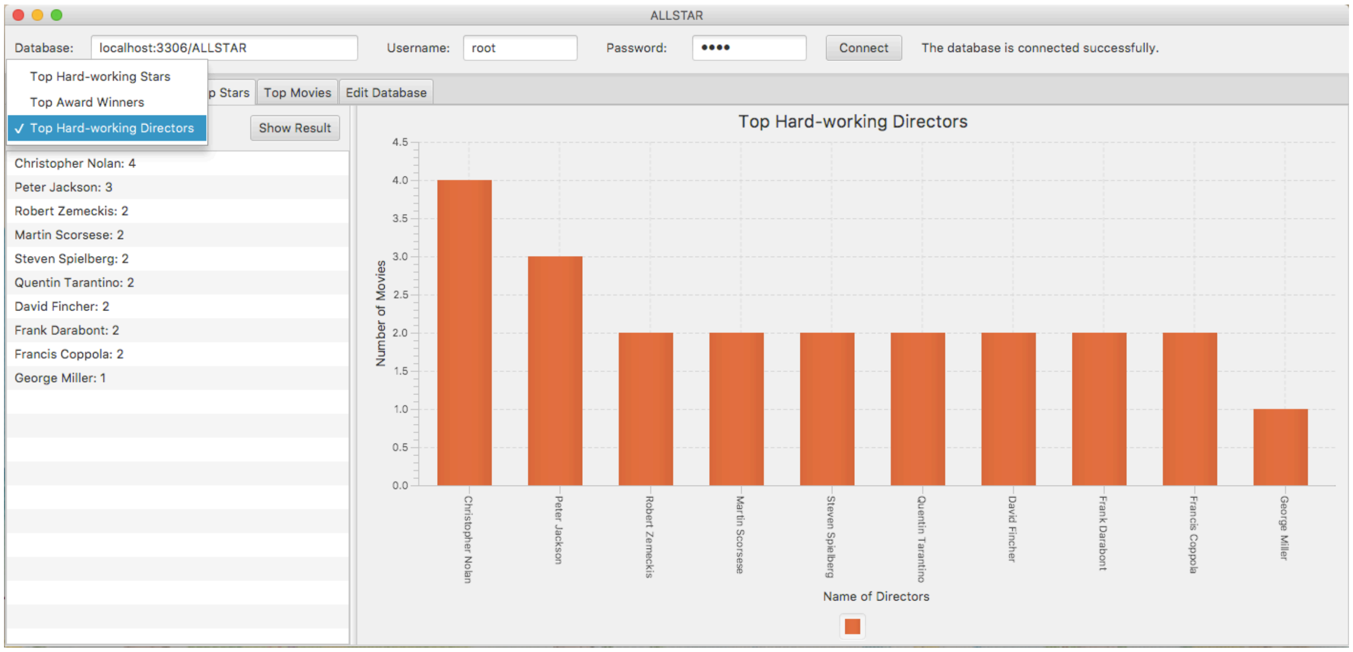
The database is connected successfully.

Title:	Title of movie
Year:	Year of production
Genres:	Genres
Language:	English
Studio:	Studio Name
Rating >= :	7
Star:	Star in the movie
Award:	Award Name
Award Year	1995

MID	Title	Year	Genres	Language	Studio	Rating	Stars	Director	Award	Award Y...
5	The Shawsh...	1994	Drama	English		9.3	Tim Rob...	Frank Da...	Best Scr...	1995 199...
11	Pulp Fiction	1994	Drama	English		8.9	Bruce Wi...	Quentin ...	Best Scr...	1995 199...

The top star and top movie

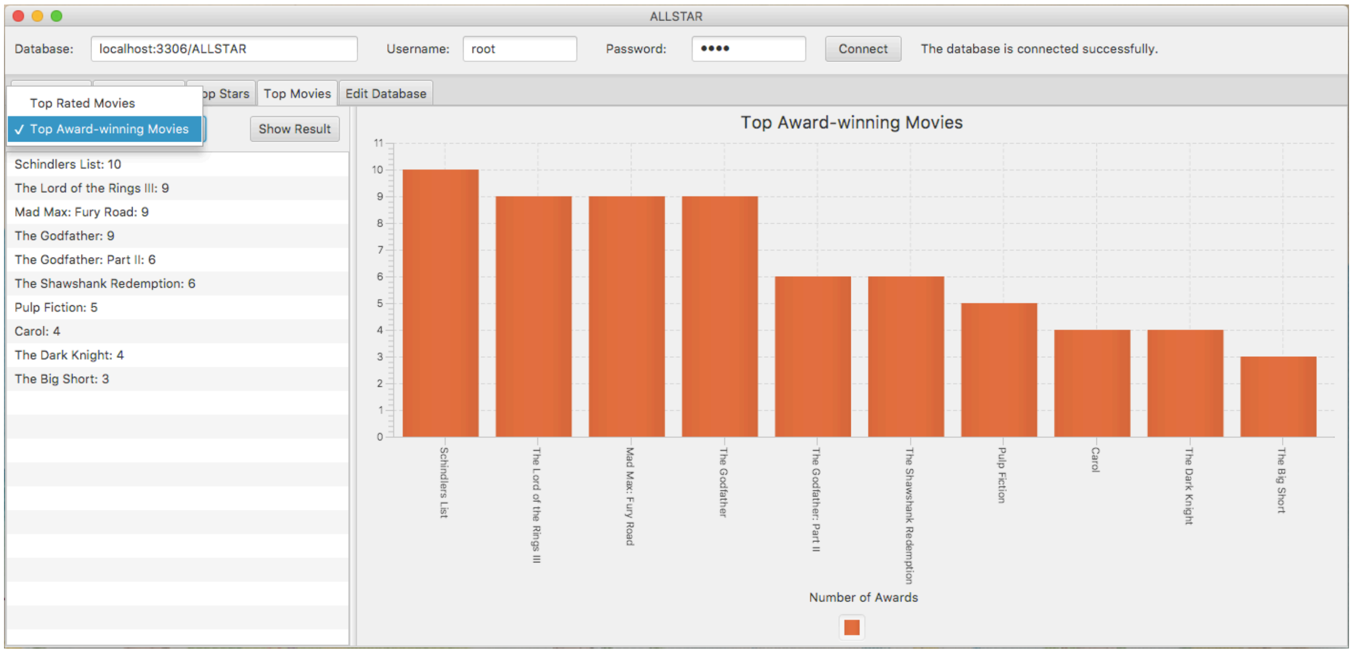
The top star and top movie tab allows user find top stars who participates in most movies or receives most awards, movies with highest rating and so on. For example, if we want to find the most hard-working director and top award-winning movies, just choose the corresponding bar and the result will be displayed in the content pane. It will be presented as histogram.



Database:	localhost:3306/ALLSTAR
Username:	root
Password:	••••

The database is connected successfully.

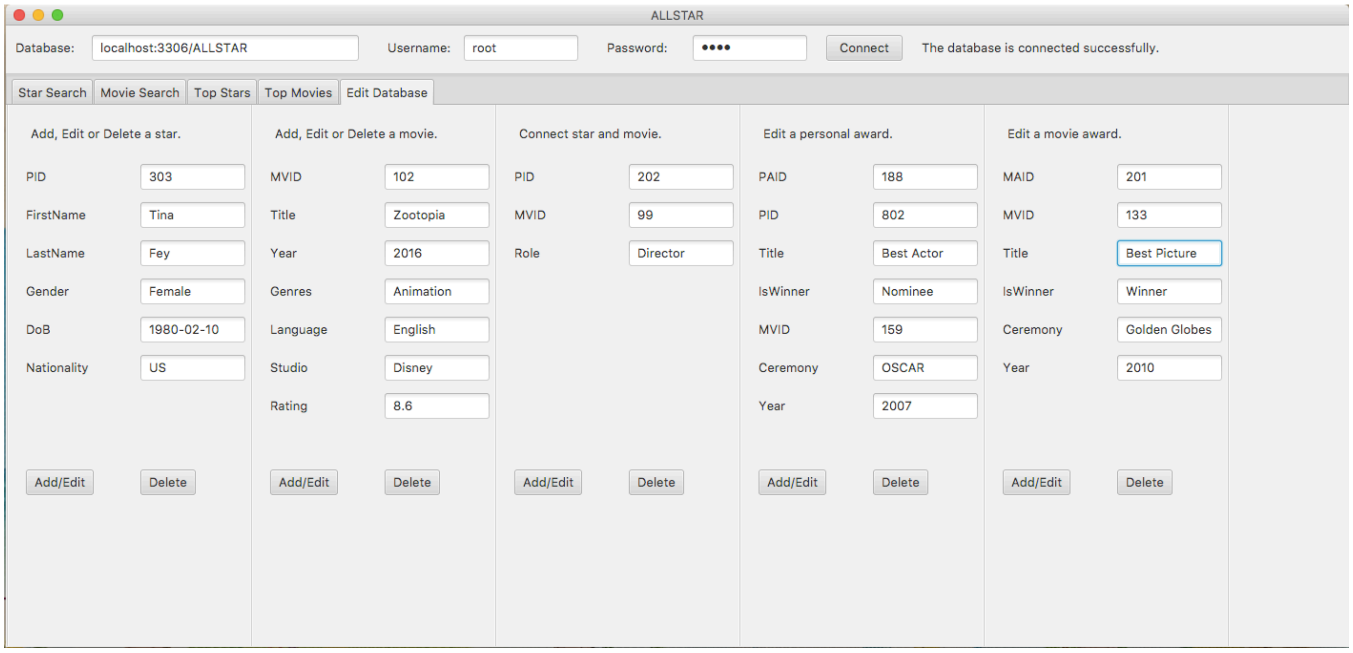
Christopher Nolan: 4
Peter Jackson: 3
Robert Zemeckis: 2
Martin Scorsese: 2
Steven Spielberg: 2
Quentin Tarantino: 2
David Fincher: 2
Frank Darabont: 2
Francis Coppola: 2
George Miller: 1



Schindlers List: 10
The Lord of the Rings III: 9
Mad Max: Fury Road: 9
The Godfather: 9
The Godfather: Part II: 6
The Shawshank Redemption: 6
Pulp Fiction: 5
Carol: 4
The Dark Knight: 4
The Big Short: 3

Modify functionality

The last tab allows user to modify records in database directly. However we hide the tables about ceremony, but once user add a new award, ALLSTAR will check if corresponding ceremony exists and determine if it should be created automatically.



Add, Edit or Delete a star.

PID	303
FirstName	Tina
LastName	Fey
Gender	Female
DoB	1980-02-10
Nationality	US

Add, Edit or Delete a movie.

MVID	102
Title	Zootopia
Year	2016
Genres	Animation
Language	English
Studio	Disney
Rating	8.6

Connect star and movie.

PID	202
MVID	99
Role	Director

Edit a personal award.

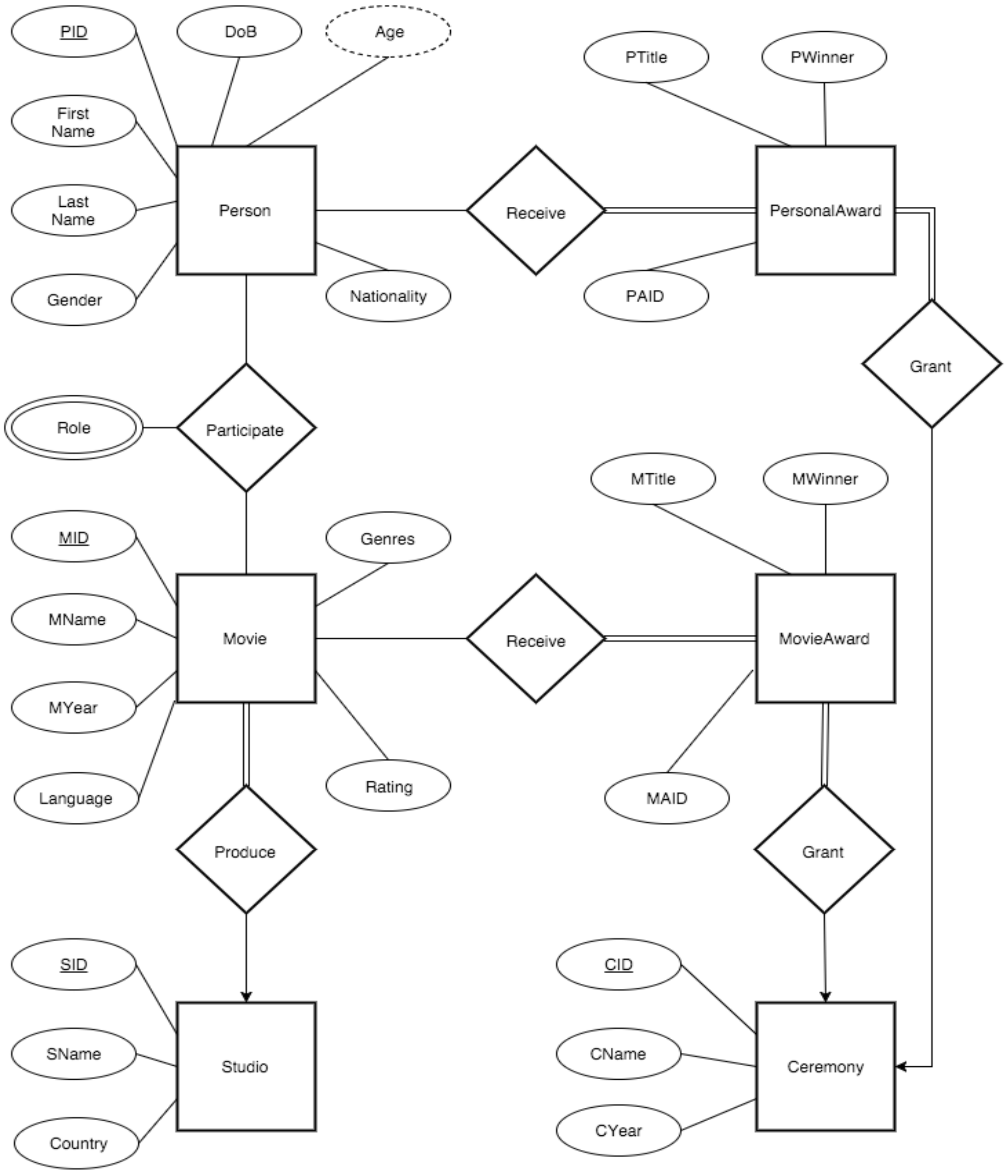
PAID	188
PID	802
Title	Best Actor
IsWinner	Nominee
MVID	159
Ceremony	OSCAR
Year	2007

Edit a movie award.

MAID	201
MVID	133
Title	Best Picture
IsWinner	Winner
Ceremony	Golden Globes
Year	2010

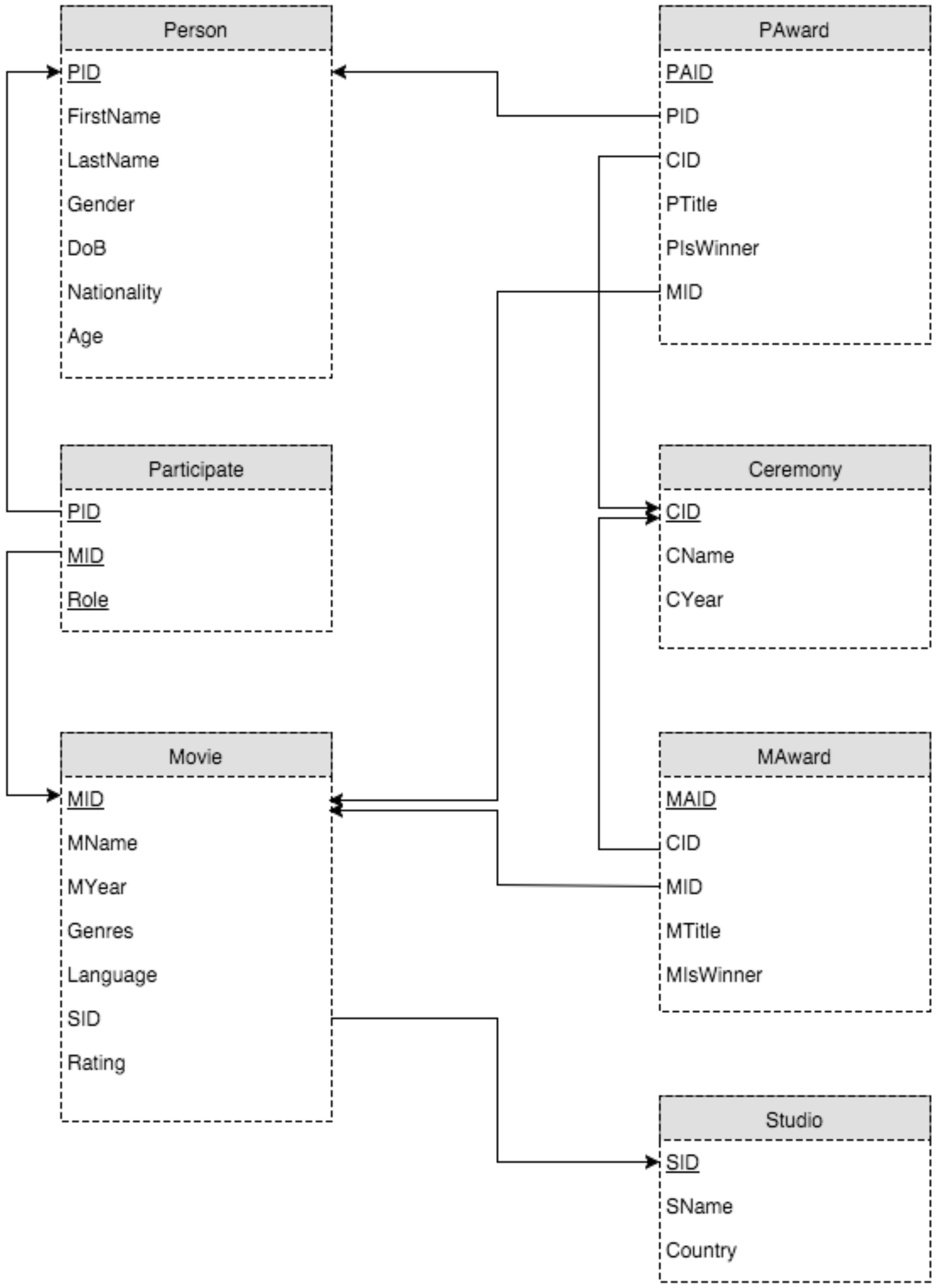
Diagram, Table conversion and Query design

ER Diagram



ER Diagram

Relational Diagram



Relational Diagram

Table Conversion

The conversion from ER diagram to relational diagram and model is based on following ideas:

- Movie, star, studio and award are independent entities; we should create tables for them separately.
- The relationship between star and movie is many to many; it cannot be merged into any entity table so we have to create a new table for it.
- Award itself is a little bit complicated since it can be granted to a star or a movie, and we also deem nomination as some kind of honor along with award. So we decide first create two tables for personal award and movie award separately. Second, we add a column "IsWinner" to distinguish winner from nominees, while we use an ID to distinguish between nominees.
- To combine the common information of personal award and movie award, we create a table named ceremony, so the redundancy is eliminated.
- The relationship between person and personal award now is one to many and award is total participation into it, so we can combine this relationship into personal award table.
- The relationship between movie and personal award can also be combined into movie award table.
- To indicate in which movie a star's performance leads to the award, we added an additional column movie ID into personal award table.

Searching Query Design

Taking star searching as example, a typical searching should consists following steps:

- Taking searching keywords provided by user
- Parse them one by one, and meanwhile adding conditions into where clause of initial SQL query string.
- Execute initial SQL query from five relevant tables and store star information returned from database, including person ID.
- For each person ID, execute two separate queries from participate table and personal award table, because there could be a long list of movies and awards for a single star and it could be complicated to retrieve all these lists in the previous query. Then we store the results, movie list and award list, into two strings and send them to UI along with personal information.
- UI displays result in a table view.

Possible Improvement

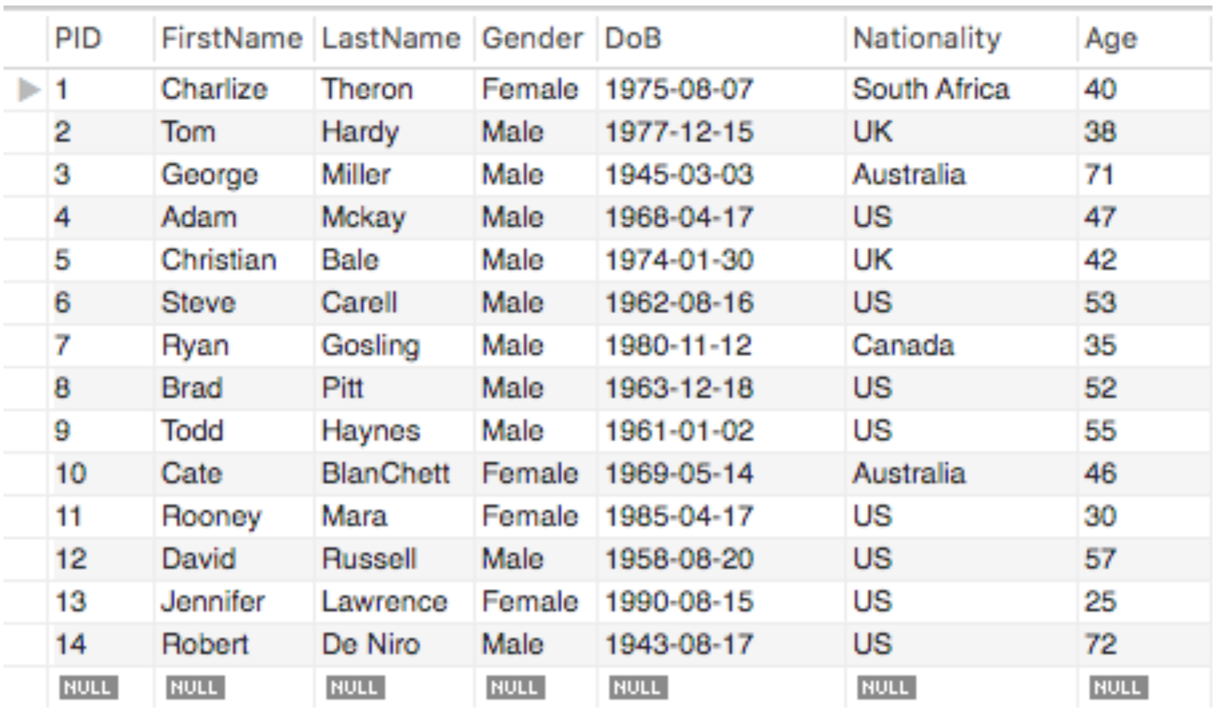
If we have extra time to improve our application, here are some of our ideas:

- User Data Validation. SQL injection attack is a common attack against DBMS, and currently we did not check user input strictly, so it could be vulnerable to SQL injection. We can check user input for malicious strings while escaping some characters like “ ‘ ”.
- Table Expansion. Recently we found that create an additional table for genres or countries could be a good idea, which requires expansion of table numbers; and we can also add more columns into tables like movie or star to provide more detailed information to user.
- Value Integrity. Currently we did not filter invalid input from user, a possible improvement is to check user input and if it is invalid, we pop up a message box to notify users.
- Content Expansion. We have only limited numbers of records for each table, they are enough for demonstration but we can import more records once we verified basic features with current database.
- Introduce Trigger. Column ‘age’ is derivative from ‘DoB’, we could use internal triggers to modify age automatically if we have extra time to do that.
- UI improvement. We want our application more user-friendly by ‘guessing’ what they want based on their input strings. For now, we can only search based on exactly what users give.
- Check our database design against normal forms to find if there is anything we can do better.

Sample

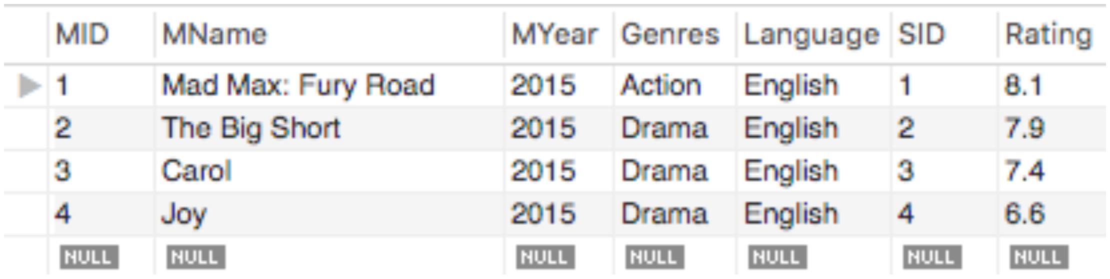
Sample Tables

To enable development of application, we created seven tables and inserted basic data to verify correctness of Java code and SQL queries.



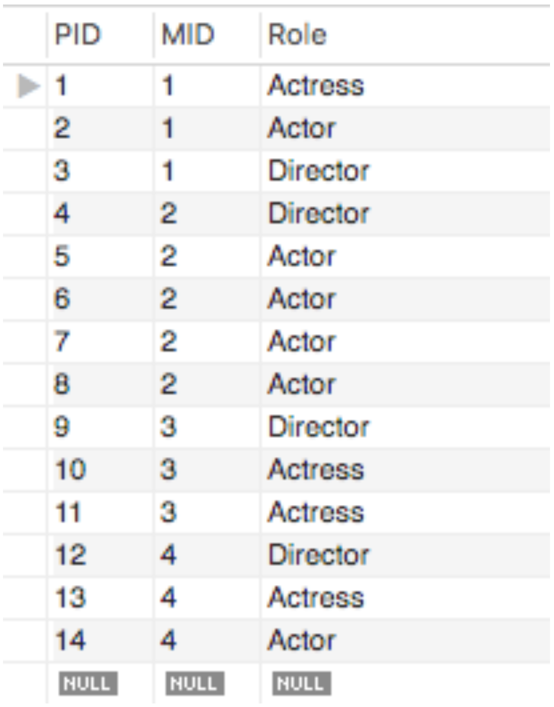
PID	FirstName	LastName	Gender	DoB	Nationality	Age
1	Charlize	Theron	Female	1975-08-07	South Africa	40
2	Tom	Hardy	Male	1977-12-15	UK	38
3	George	Miller	Male	1945-03-03	Australia	71
4	Adam	Mckay	Male	1968-04-17	US	47
5	Christian	Bale	Male	1974-01-30	UK	42
6	Steve	Carell	Male	1962-08-16	US	53
7	Ryan	Gosling	Male	1980-11-12	Canada	35
8	Brad	Pitt	Male	1963-12-18	US	52
9	Todd	Haynes	Male	1961-01-02	US	55
10	Cate	BlanChett	Female	1969-05-14	Australia	46
11	Rooney	Mara	Female	1985-04-17	US	30
12	David	Russell	Male	1958-08-20	US	57
13	Jennifer	Lawrence	Female	1990-08-15	US	25
14	Robert	De Niro	Male	1943-08-17	US	72
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Person Table



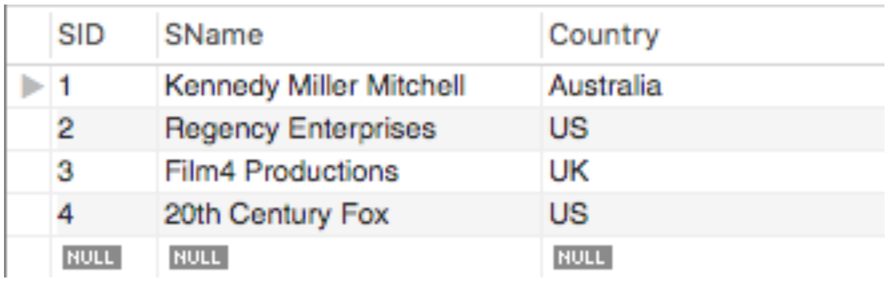
MID	MName	MYear	Genres	Language	SID	Rating
1	Mad Max: Fury Road	2015	Action	English	1	8.1
2	The Big Short	2015	Drama	English	2	7.9
3	Carol	2015	Drama	English	3	7.4
4	Joy	2015	Drama	English	4	6.6
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Movie Table



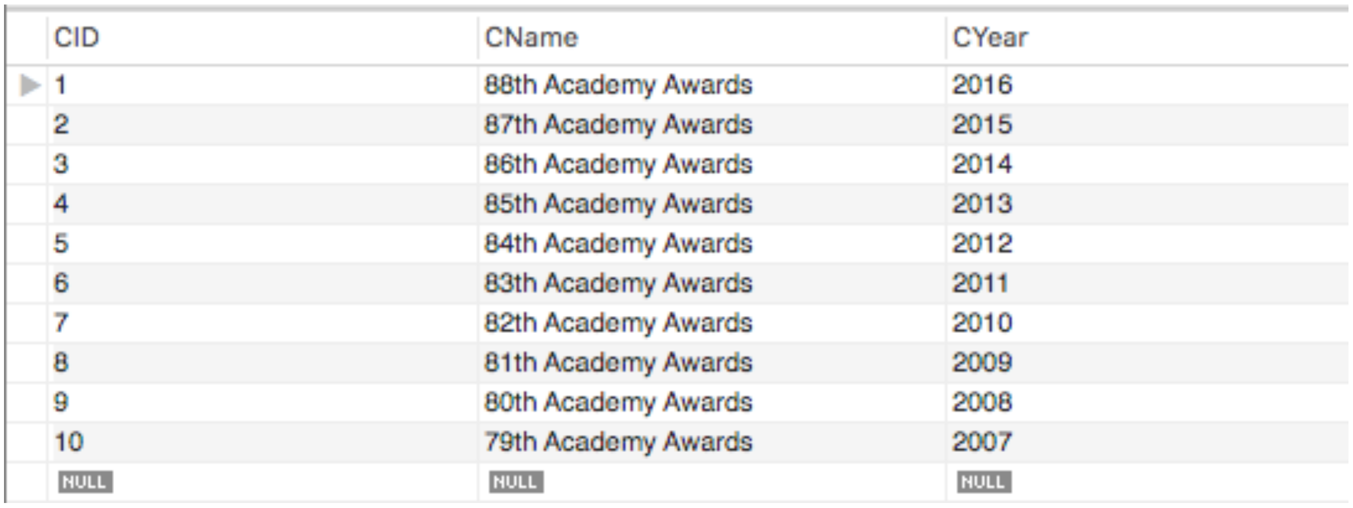
PID	MID	Role
1	1	Actress
2	1	Actor
3	1	Director
4	2	Director
5	2	Actor
6	2	Actor
7	2	Actor
8	2	Actor
9	3	Director
10	3	Actress
11	3	Actress
12	4	Director
13	4	Actress
14	4	Actor
NULL	NULL	NULL

Participate Table



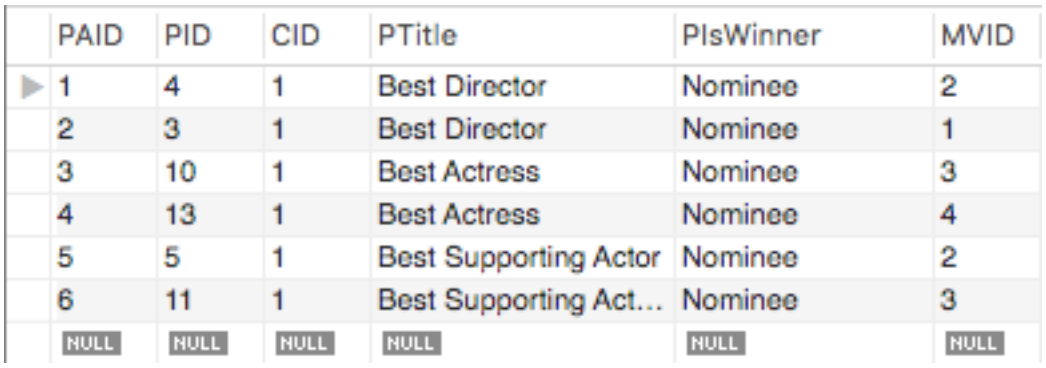
SID	SName	Country
1	Kennedy Miller Mitchell	Australia
2	Regency Enterprises	US
3	Film4 Productions	UK
4	20th Century Fox	US
NULL	NULL	NULL

Studio Table



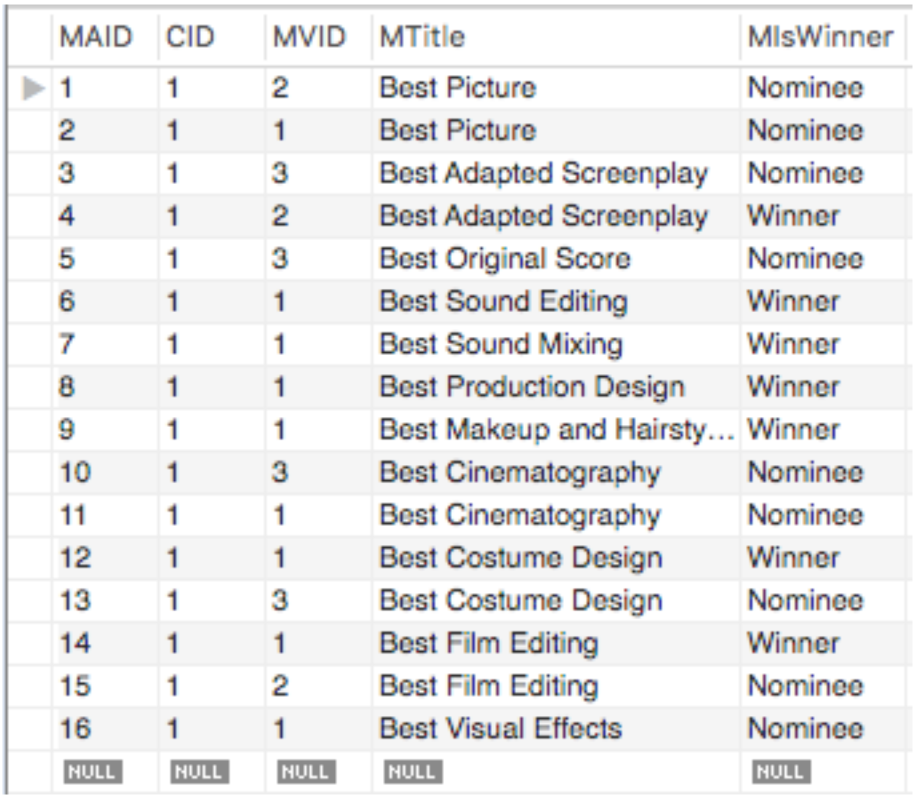
CID	CName	CYear
1	88th Academy Awards	2016
2	87th Academy Awards	2015
3	86th Academy Awards	2014
4	85th Academy Awards	2013
5	84th Academy Awards	2012
6	83th Academy Awards	2011
7	82th Academy Awards	2010
8	81th Academy Awards	2009
9	80th Academy Awards	2008
10	79th Academy Awards	2007
NULL	NULL	NULL

Ceremony Table



PAID	PID	CID	PTitle	PIsWinner	MVID
1	4	1	Best Director	Nominee	2
2	3	1	Best Director	Nominee	1
3	10	1	Best Actress	Nominee	3
4	13	1	Best Actress	Nominee	4
5	5	1	Best Supporting Actor	Nominee	2
6	11	1	Best Supporting Act…	Nominee	3
NULL	NULL	NULL	NULL	NULL	NULL

Personal Award Table



MAID	CID	MVID	MTitle	MIsWinner
1	1	2	Best Picture	Nominee
2	1	1	Best Picture	Nominee
3	1	3	Best Adapted Screenplay	Nominee
4	1	2	Best Adapted Screenplay	Winner
5	1	3	Best Original Score	Nominee
6	1	1	Best Sound Editing	Winner
7	1	1	Best Sound Mixing	Winner
8	1	1	Best Production Design	Winner
9	1	1	Best Makeup and Hairsty…	Winner
10	1	3	Best Cinematography	Nominee
11	1	1	Best Cinematography	Nominee
12	1	1	Best Costume Design	Winner
13	1	3	Best Costume Design	Nominee
14	1	1	Best Film Editing	Winner
15	1	2	Best Film Editing	Nominee
16	1	1	Best Visual Effects	Nominee
NULL	NULL	NULL	NULL	NULL

Movie Award Table

Sample Queries

SQL query for star list looks like the example below, or much simpler, depending on input parameters:

```
SELECT DISTINCT P.PID, P.FirstName, P.LastName, P.DoB, P.Gender, P.Nationality, P.Age
FROM Person P, Participate PT, Movie M, PAward PA, Ceremony C
```

```
WHERE P.PID=PT.PID AND PT.MVID=M.MVID AND M.MName='gg' AND P.PID=PA.PID AND PA.CID=C.CID AND
PA.PTitle='hh' AND C.CYear=2001 AND P.FirstName='aa' AND P.LastName='bb' AND P.DoB='cc' AND P.Gender='dd'
AND P.Nationality='ee' AND P.Age=34;
```

After we got star list, two separated queries as shown below will be executed for each star:

```
SELECT M.MName, PT.Role
FROM Person P, Participate PT, Movie M
WHERE P.PID=PT.PID AND PT.MVID=M.MVID AND P.PID=xx;

SELECT PA.PTitle, PA.PIsWinner, C.CName, C.CYear
FROM Person P, PAward PA, Ceremony C
WHERE P.PID=PA.PID AND PA.CID=C.CID AND P.PID=xx;
```

Relevant Java Code Samples

Generate SQL query for star list

```
private String generateStarQuery (String fName,
                                  String lName,
                                  String dob,
                                  String gender,
                                  String nation,
                                  String age,
                                  String starsIn,
                                  String awardS,
                                  String awardYS) {
        String finalQuery;
        String queryCommand = "SELECT DISTINCT P.PID, P.FirstName, P.LastName, P.DoB, P.Gender,
P.Nationality, P.Age FROM Person P";
        String queryCondition = " WHERE";
        if (starsIn.trim().length()!=0){
            queryCommand = queryCommand.concat(", Participate PT, Movie M");
            queryCondition = queryCondition.concat(" P.PID=PT.PID AND PT.MVID=M.MVID AND M.MName='" +
starsIn + "' AND");
        }
        if ((awardYS.trim().length()!=0)&&(awardS.trim().length()!=0)) {
            queryCommand = queryCommand.concat(", PAward PA, Ceremony C");
            queryCondition = queryCondition.concat(" P.PID=PA.PID AND PA.CID=C.CID AND PA.PTitle='" + awardS
+ "' AND C.CYear=" + awardYS + " AND");
        } else if (awardYS.trim().length()!=0){
            queryCommand = queryCommand.concat(", PAward PA, Ceremony C");
            queryCondition = queryCondition.concat(" P.PID=PA.PID AND PA.CID=C.CID AND C.CYear=" + awardYS +
" AND");
        } else if (awardS.trim().length()!=0) {
            queryCommand = queryCommand.concat(", PAward PA");
            queryCondition = queryCondition.concat(" P.PID=PA.PID AND PA.PTitle='" + awardS + "' AND");
        }
        if (fName.trim().length()!=0){
            queryCondition = queryCondition.concat(" P.FirstName='"+ fName +"' AND");
        }
        if (lName.trim().length()!=0){
            queryCondition = queryCondition.concat(" P.LastName='"+ lName +"' AND");
        }
        if (dob.trim().length()!=0){
            queryCondition = queryCondition.concat(" P.DoB='"+ dob +"' AND");
        }
        if (gender.trim().length()!=0){
            queryCondition = queryCondition.concat(" P.Gender='"+ gender +"' AND");
        }
        if (nation.trim().length()!=0){
            queryCondition = queryCondition.concat(" P.Nationality='"+ nation +"' AND");
        }
        if (age.trim().length()!=0){
            queryCondition = queryCondition.concat(" P.Age="+ age +" AND");
        }

        finalQuery = queryCommand.concat(queryCondition);
        finalQuery = finalQuery.substring(0, finalQuery.lastIndexOf(" AND")).concat(";");

        System.out.println(finalQuery);

        return finalQuery;
    }
```

Store star list and send additional queries for movie and award information

```
public ArrayList<Star> getStarResult (String fName,
                                      String lName,
                                      String dob,
                                      String gender,
                                      String nation,
                                      String age,
                                      String starsIn,
                                      String awardS,
                                      String awardYS
                                      ) throws SQLException {
        try {
            String starSearchQuery = generateStarQuery(fName, lName, dob, gender, nation, age, starsIn,
awardS, awardYS);
            Statement starSearch = connDB.createStatement();
            ResultSet starResult = starSearch.executeQuery(starSearchQuery);

            starList.clear();

            while (starResult.next()) {
                starList.add(new Star(starResult.getString("PID"),
                                starResult.getString("FirstName"),
                                starResult.getString("LastName"),
                                starResult.getString("DoB"),
                                starResult.getString("Gender"),
                                starResult.getString("Nationality"),
                                starResult.getString("Age"),
                                "", "", "", ""));

          }

          PreparedStatement starMovieState = connDB.prepareStatement("SELECT M.MName, PT.Role FROM
Person P, Participate PT, Movie M WHERE P.PID=PT.PID AND PT.MVID=M.MVID AND P.PID=?;");
          PreparedStatement starAwardState = connDB.prepareStatement("SELECT PA.PTitle, PA.PIsWinner,
C.CName, C.CYear FROM Person P, PAward PA, Ceremony C WHERE P.PID=PA.PID AND PA.CID=C.CID AND
P.PID=?;");

          for (Star s: starList) {
             starMovieState.setInt(1, Integer.parseInt(s.getPid()));
             ResultSet starMovies = starMovieState.executeQuery();
             while (starMovies.next()) {
                if (starMovies.getString("Role").contains("Director")) {
                   s.setDirects(s.getDirects().concat(starMovies.getString("MName") + " "));
                } else if (starMovies.getString("Role").contains("Act")) {
                   s.setStarsIn(s.getStarsIn().concat(starMovies.getString("MName") + " "));
                }
             }

             starAwardState.setInt(1, Integer.parseInt(s.getPid()));
             ResultSet starAwards = starAwardState.executeQuery();
             while (starAwards.next()){
                s.setAwardS(s.getAwardS().concat(starAwards.getString("PTitle")         +         "         "         +
starAwards.getString("PIsWinner") + " in " +starAwards.getString("CName") + " " + starAwards.getString("CYear")
+ " "));
                s.setAwardYS(s.getAwardYS().concat(starAwards.getString("CYear") + " "));
             }

          }

          for (Star s : starList) {
             System.out.println(s.getPid() + " " + s.getFName() + " " + s.getLName() + " " + s.getStarsIn() + " " +
s.getDirects() + " " + s.getAwardS() + " " + s.getAwardYS());
          }

       }
       catch (SQLException e) {
          Logger.getLogger(UIController.class.getName()).log(Level.SEVERE, null, e);
       }

       return starList;
    }
```