Daniel S. Fridjonsson, Lars Andersen, Søren S. Als & Mathias W. Pedersen.
sw608f14 - Room: 2.1.10

Self study 1 - miniproject part 1

Actors, directors, and writers are decided to be stored only as person, since they can preform multiple roles throughout their career. A person has a name, birthday, and additional information. The table for persons will be:

PERSON ID	PERSON NAME	BIRTHDAY
123	Johnny Known	03-09-1934

A movie can contain movie name, movie id, release data, and rating. The rating is calculated by taking the average of the user ratings, see last table, for a specific movie.

MOVIE ID	MOVIE NAME	RELEASE DATE	RATING
123	The Hobbit	05-12-2014	9.2

To connect persons to movies another table is used. Where the role in this table is an enum that contains actor, director, and writer.

MOVIE ID	PERSON ID	ROLE
637	463	Actor
563	563	Director
637	674	Writer

Awards can be given to both persons and movies, therefore it is necessary to have two tables, since the id's can be the same for both persons and movies. The tables will be as follows:

MOVIE ID	AWARD	YEAR
542	Best Picture	2013

PERSON ID	MOVIE ID	AWARD	YEAR
524	542	Best Actor	2013

To store a user of the system it is necessary to store a username and password. The password should be encrypted, but security is not the focus of this exercise.

USER ID	USERNAME	PASSWORD
546	NiceName	123456

Furthermore, a user can rate movies, and a table for this is created. This table contains user id, movie id, and rating

MOVIE ID	USER ID	RATING
634	342	8

Daniel S. Fridjonsson, Lars Andersen, Søren S. Als & Mathias W. Pedersen.
sw608f14 - Room: 2.1.10

Self study 2 - miniproject part 2

ER Diagram

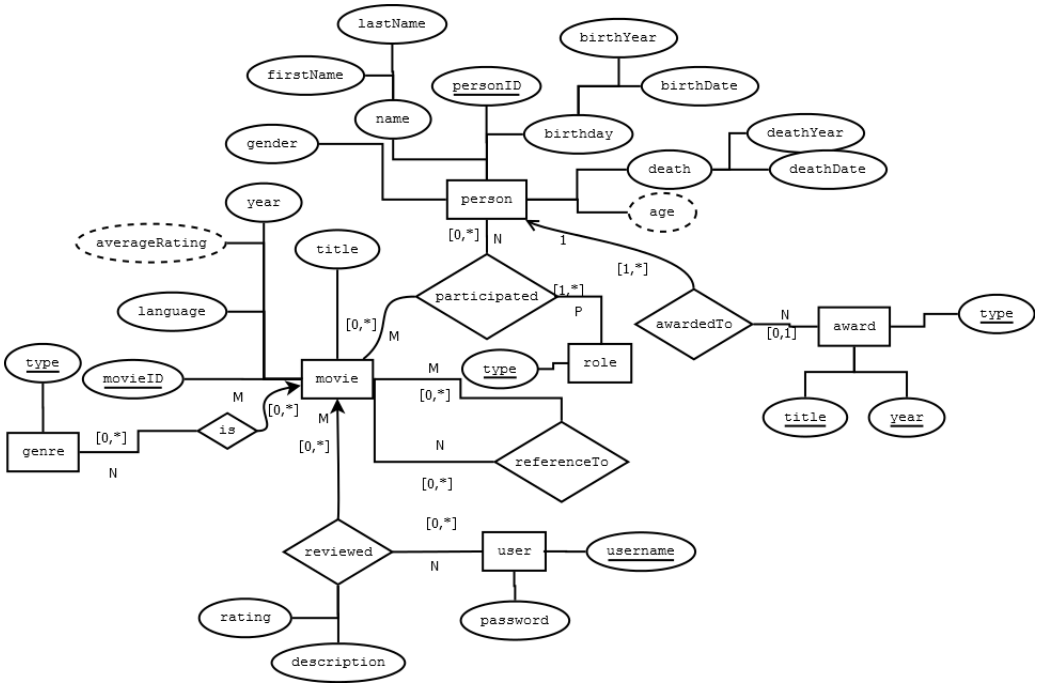


Figure 1: ER Diagram selfstudy 2

Relational Schema

genre

$\{[\underline{type : String}]\}$

movie

$\{[\underline{movieID : Int}, language : String, year : Int, title : String]\}$

user

$\{[\underline{username : String}, password : String]\}$

award

$\{[\underline{title : String, year : Int, type : String}, receiver \rightarrow person]\}$

person

$\{[\underline{personID : Int}, firstName : String, lastName : String, gender : String, birthYear : Int, birthDate : String, deathYear : Int, deathDate : String]\}$

participated

$\{[\underline{movieID \rightarrow movie, personID \rightarrow person, type \rightarrow role}]\}$

is $\{[\underline{type \rightarrow genre, movieID \rightarrow movie}]\}$

reviewed

$\{[\underline{movieID \rightarrow movie, username \rightarrow user}, rating : Int, description : String]\}$

referenceTo

$\{[\underline{from \rightarrow movie, to \rightarrow movie}]\}$

Reflections

The design is different from the initial design in that the initial design was at a mere table level. Furthermore, several additional attributes have been added to accommodate the expected information requirements. Another change is that there was no primary and foreign keys highlighted in the original answer. Average rating in the initial database was intended to be a calculated attribute, but was not listed as such, contrary to the new design. Additionally we have decided that an award only can be given to a person, as that made the design simpler. Finally we added the $referenceTo$ relation, as that was a requirement we missed in the first selfstudy.

Selfstudy 4

Revised ER Diagram

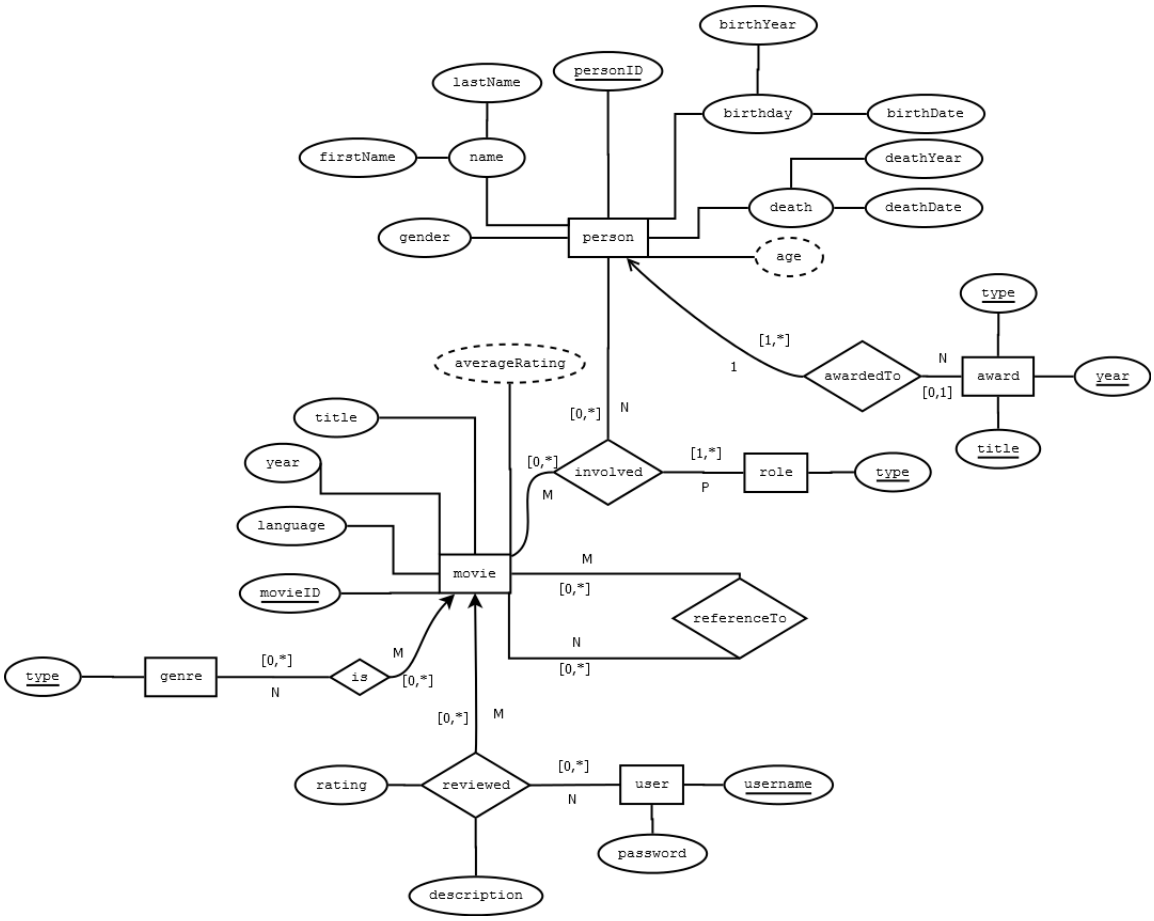


Figure 2: Revised ER Diagram selfstudy 4

Revised relational schema

ER Diagram

Relational Schema

genre

$\{[\underline{type} : String]\}$

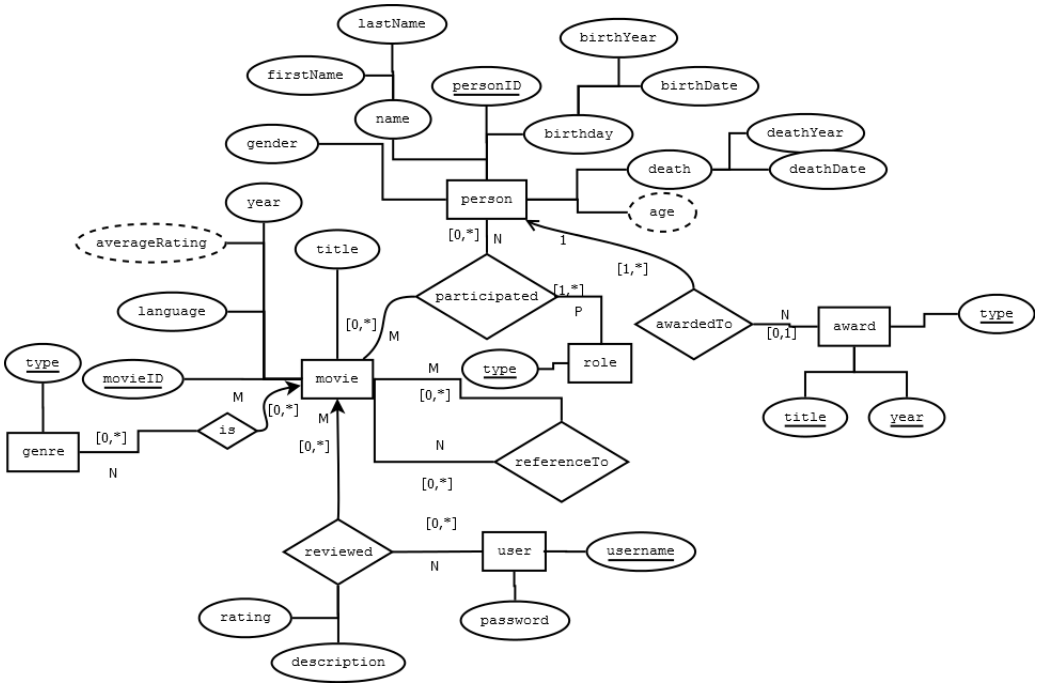
role $\{[\underline{type} : String]\}$

movie

$\{[\underline{movieID} : Int, language : String, year : Int, title : String]\}$

user

$\{[\underline{username} : String, password : String]\}$



award

$\{[\underline{title : String, year : Int, type : String}, receiver : Int \rightarrow person]\}$

person

$\{[\underline{personID : Int}, firstName : String, lastName : String, gender : String, birthYear : Int, birthDate : String, deathYear : Int, deathDate : String]\}$

involved

$\{[\underline{movieID : Int \rightarrow movie, personID : Int \rightarrow person, type : String \rightarrow role}]\}$

is $\{[\underline{type : String \rightarrow genre, movieID : Int \rightarrow movie}]\}$

reviewed

$\{[\underline{movieID : Int \rightarrow movie, username : String \rightarrow user}, rating : Int, description : String]\}$

referenceTo

$\{[\underline{from : Int \rightarrow movie, to : Int \rightarrow movie}]\}$

Functional Dependencies

$$
\begin{aligned}
genre.type &\rightarrow \emptyset \\
role.type &\rightarrow \emptyset \\
movieID &\rightarrow language, movie.year, movie.title \\
username &\rightarrow password \\
award.title, award.year, award.type &\rightarrow personID \\
personID &\rightarrow firstName, lastName, gender, birthYear, birthDate, deathYear, de \\
movieID, personID, role.type &\rightarrow \emptyset \\
genre.type, movieID &\rightarrow \emptyset \\
movieID, username &\rightarrow rating, description \\
movieID, movieID &\rightarrow \emptyset
\end{aligned}
$$

It is 3NF as rule 2 applies for all the functional dependencies, since the primary key in each relation is also a super key and due to it being 2NF, as it is fully functionally dependent. As it is rule 2 that applies for all the functional dependencies, it is also BCNF.

As the schema is already 3NF, there is no need to normalize the relations.

SQL Statements

```
CREATE TABLE genre
{
    type varchar(15) PRIMARY KEY
};

CREATE TABLE role
{
    type varchar(15) PRIMARY KEY
};


CREATE SEQUENCE serialmovie START 0;

CREATE TABLE movie
{
    movieID integer PRIMARY KEY DEFAULT nextval('serialmovie'),
    language varchar(20) DEFAULT 'English',
    year integer NOT NULL,
    title varchar(100) NOT NULL
};

CREATE TABLE user
{
    username varchar(20) PRIMARY KEY,
    password varchar(50) NOT NULL
};
```

```
CREATE TABLE award
{
      title varchar(30),
      year integer,
      type varchar(15),
      receiver integer NOT NULL,
      PRIMARY KEY(title,year,type),
      FOREIGN KEY(receiver) REFERENCES person(personID)
};

CREATE SEQUENCE serialperson START 0;

CREATE TABLE person
{
      personID integer PRIMARY KEY DEFAULT nextval('serialperson'),
      firstName varchar(20) NOT NULL,
      lastName varchar(50) NOT NULL,
      gender varchar(6),
      birthYear integer,
      birthDate varchar(7),
      deathYear integer,
      deathDate varchar(7)
};

CREATE TABLE involved
{
      movieID integer,
      personID integer,
      type varchar(15),
      PRIMARY KEY(movieID, personID, type),
      FOREIGN KEY(movieID) REFERENCES movie(movieID),
      FOREIGN KEY(personID) REFERENCES person(personID),
      FOREIGN KEY(type) REFERENCES role(type)
}

CREATE TABLE is
{
      type varchar(15),
      movieID integer,
      PRIMARY KEY(type,movieID),
      FOREIGN KEY(type) REFERENCES genre(type),
      FOREIGN KEY(movieID) REFERENCES movie(movieID)
};

CREATE TABLE reviewed
{
      movieID integer,
      username varchar(20),
      rating integer NOT NULL,
      description varchar(MAX),
```

```
    PRIMARY KEY(movieID, username),
    FOREIGN KEY(movieID) REFERENCES movie(movieID),
    FOREIGN KEY(username) REFERENCES user(username)
};

CREATE TABLE referenceTo
{
    from integer,
    to integer,
    PRIMARY KEY(from, to),
    FOREIGN KEY(from, to) REFERENCES movie(movieID)
};
```

Reflections

Differences from the original design is that we changed the layout of our ER Diagram and renamed the *participated* relation to *involved*. Considerations we have to take in the future is our encoding of dates, which could be changed to the *DATETIME* datatype. Also, we forgot the role relation in our first schema, and was thus added to the revised schema.