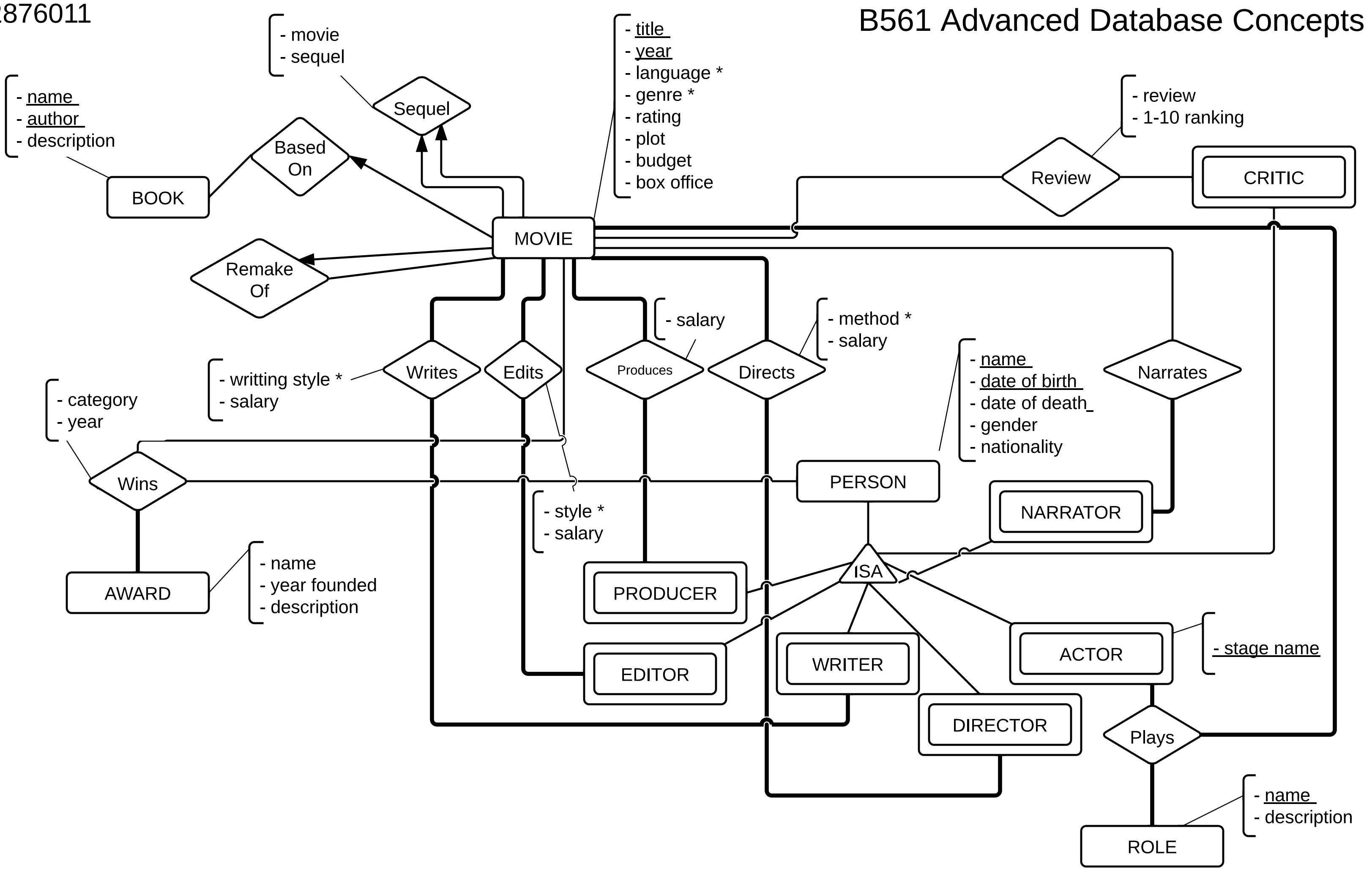
Enrique Areyan
UID: 0002876011

Movies ER Diagram

B561 Advanced Database Concepts



Scenarios considered and Non-trivial decisions:

- Instead of having only one relationship between person and movie, the relationships produces, writes, narrates,... are separated because we want to ensure that a movie has at least one or more directors, producers, writers and editors, but not necessarily a narrator, for instance. Also, an actor is handled different so that its relationship must be separated. On each relationship we can store information pertaining only to a person in a given movie, for instance, we can store the writing style a particular writer used in a movie. This person may use different writting styles in different movies.
- Awards' categories may change from one year to another. A recipient of an award will probably change each time and has a movie as a context for the award. I model this as a ternary relationship between award, person and movie. An example instance of this relation would be (Nicolas Cage, Oscar, Best Actor, 2002, Adaptation)
- In the *remake of r*elation, one movie (the new movie) may be a remake of only another movie, but the old movie may have many remakes. Also, a movie may only have one sequel and be the sequel to only one movie.
- The design is flexible in that different types of personnel can be added easily.
- Reviews of critics are separated from both the critic and the movie so that many reviews on different movies are possible.

Enrique Areyan
UID: 0002876011

Homework 1 Answers
B561 Advanced Database Concepts

2. Relational Model

Movies(id_movie: integer, title: string, year: integer, rating: string, plot: string, budget: real, boxoffi: real)
Languages(name: string)
Movie_Languages(id_movie REF Movies, name REF Languages)
Genres(name: string, description:string)
Movie_Genres(id_movie REF Movise, name REF Genres)
Sequels(id_movie REF Movies, id_sequel REF Movies NOT NULL)
RemakeOf(id_movie REF Movies, id_remake REF Movies NOT NULL)
Books(id_book:integer, name: string, author: string)
BasedOn(id_movie REF Movies, id_book REF Books NOT NULL)
People(id_person: integer, name: string, dob: date, dod: date, gender: string, nationality: string)
Producers(id_producer REF People)
Narrators(id_narrator REF People)
Writers(id_writer REF People)
Directors(id_director REF People)
Editors(id_editor REF People)
Critics(id_critic REF People)
Actors(id_actor REF People)
Roles(name: string, description: string)
Plays(id_movie REF Movies, id_actor REF Actors, name REF Roles)
Produces(id_producer REF Producers, id_movie REF Movies, salary: real)
Edits(id_editor REF Editors, id_movie REF Movies, salary: real, name REF Edit_Style)
Edit_Style(name: string, description: string)
Writes(id_writer REF Writers, id_movie REF Movies, salary: real, name REF Writting_Style)
Writting_Style(name: string, description: string)
Directs(id_director REF Directors, id_movie REF Movies, salary: real, method REF Directing_Style)
Directing_Method(method: string, description: string)
Narrates(id_narrator REF Narrators, id_movie REF Movies)
Review(review: string, ranking: integer, id_critic REF Critics, id_movie REF Movies)
Awards(name: string, description: string, founded: date)
Wins(name REF Awards, id_person REF people, id_movie REF Movies, category: string, year: date)

Comments:

- A movie may participate in many relations and thus, an artificial primary key (id_movie) was created for Movies. In this way we do not need to repeat the information of a movie elsewhere. This situation is similar to that of Books and People.
- The relationships involving a person and a movie such as Edits, Writes, ..., all have as primary key the combination of the person and the movie.
- The multi-valued attributes genre and language on the movies relation needed to be separated. In this way we normalize the relation and allow movies to have multiple languages and genres.
- In the RemakeOf and Sequels relations both attributes refer to Movies but ith different meaning. For instance, in the RemakeOf relation a movie with id_movie (primary key) can be the remake of only another movie, i.e. id_remake, which must have a value. A similar situation occurs in Sequels.
- The Wins relation is a ternary relationship between awards, people, and movies. The idea is that any person may win an award in a given category and year. Thus, all the attributes form the primary key.
- The plays relation is a ternary relation including an actor playing a role in a movie. The combination of actor, role, and movie define the primary key. Thus, an actor may play more than one role in the same movie and in different movies.

Enrique Areyan
UID: 0002876011

Homework 1 Answers
B561 Advanced Database Concepts

3. Functional Dependencies

Movies:
- id_movie -> title, year, rating, plot, budget, boxoffice

Genres:
- name -> description

Sequels
- id_movie -> id_sequel

RemakeOf
- id_movie -> id_remake

Books
- id_book -> name, author

BasedOn
- id_movie -> id_book

People
- id_person -> name, dob, dod, gender, nationality

Roles
- name -> description

Produces
- id_producer, id_movie -> salary

Edits
- id_editor, id_movie -> salary, name

Edit_Style
- name -> description

Writes
- id_writer,id_movie -> salary, name

Writting_Style
- name -> description

Directs
- id_director, id_movie -> salary, method

Directing_Method
- method -> description

Review
- id_critic, id_movie -> review, ranking

Awards
- name -> description, founded

4.

In my design it is possible for one actor to perform more than one role in the same movie. This is because an actor is associated with a role and a movie through the *Plays* relation. In this context, a role is the fictional character played by the actor in the movie.
For example, consider what one tuple on the *Plays* relation might look like: (Nicolas Cage, Adaptation, Charlie Kaufman), and (Nicolas Cage, Adaptation, Donald Kaufman). In this example and actor performs two different roles or characters in the same movie. (Actual names are not stored but forgein keys to the appropiate entities)
Moreover, one person may be a director, producer, editor,..., and also play different roles as an actor in the same movie.