Roles:

1. Weilian Song
   - Django framework developer
   - Mid-term report content compiler
2. Greg McIntosh
   - Backend developer
   - Responsible for database schema and mysql dump file
3. Henry Uradu
   - Frontend developer
   - Unified Model Language diagram
4. Nathaniel Serpico
   - Frontend developer
   - ER diagram creator

Database Design:

When designing our database, we wanted to keep all the big components as separate tables. Therefore we have the following big items:

1. movie
2. actor
3. acted_in
4. director
5. producer
6. writer
7. editor
8. tags
9. user
10. rating
11. review

Below you will find all the details to each table, which we won't mention here. This structure allows for complex queries to be done fairly easily. For example, if we desire to know all users' username who rated at least an 8 on any action movies, we can do:

```
SELECT u.username
FROM user as u
WHERE u.id in
  SELECT r.uid
  FROM rating as r
  WHERE r.rating >= 8 and r.mid in
    SELECT m.id
    FROM movie as m
    WHERE m.genre = "action";
```

As you can see, the query might be triple nested, but it is clean and easily understood to any developers.

We have also made some assumptions about relationships between tables. All movies' director, producer, writer and editor ID's need to follow Key Constraints with respect to director, producer, writer and editor tables. All acted_in entries' actor and movie ID's need to follow Key Constraints with respect to actor and movie tables. All tags entries' mid need to follow Key Constraints with respect to the movie table. All rating and review table's entries' uid and mid need to follow Key Constraints with respect to user and movie tables.

movie, actor, director, producer, writer, editor and user's "id" needs to

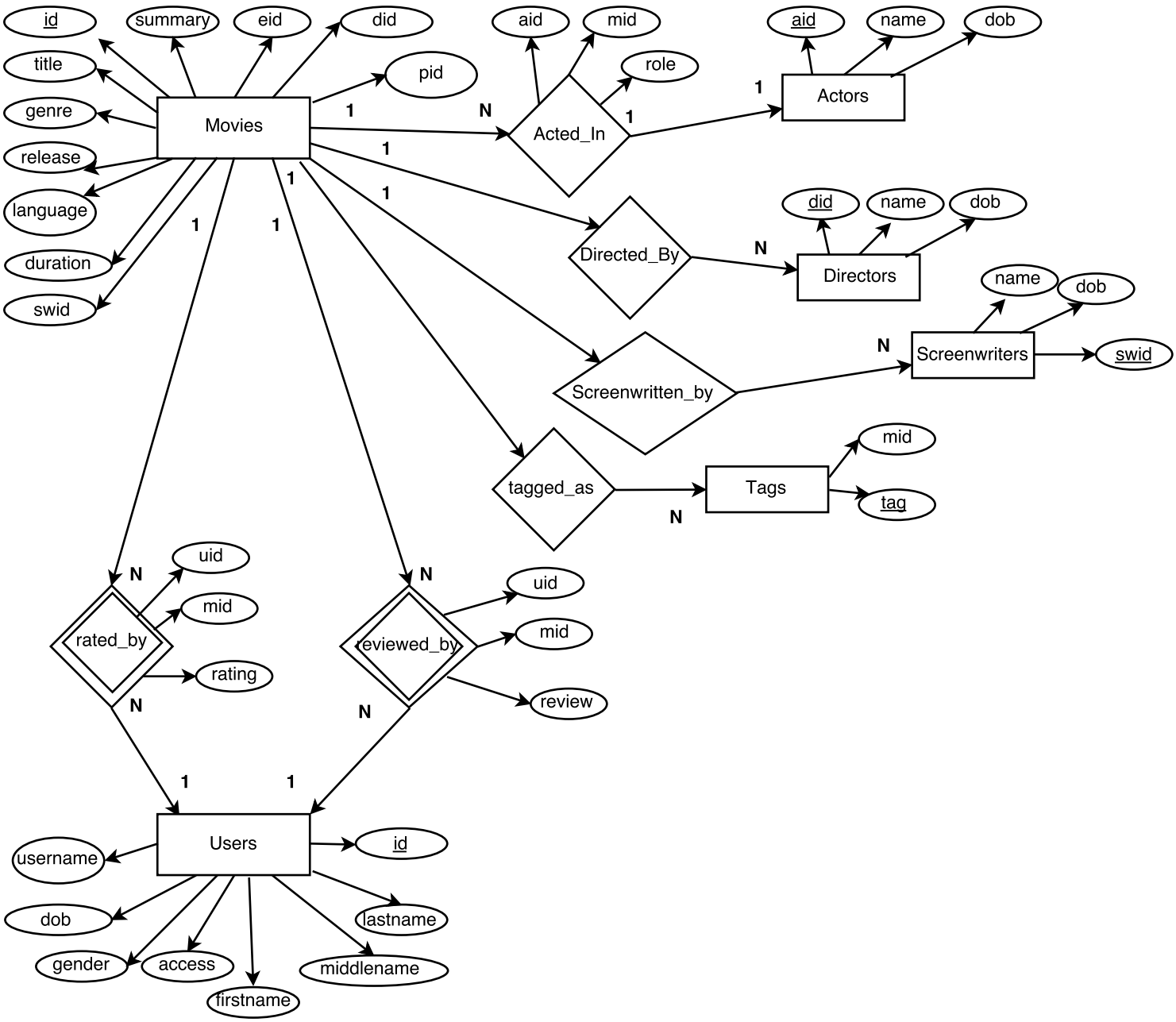
be non null and auto incrementing, as they are the primary key of each table.

All fields are reasonably limited in length, and for fields like DOB, MySQL's specific data types will enforce domain constraints.

Database Schema:

```
movie(id: int(11) auto_increment,
      title: varchar(20),
      genre: varchar(20),
      release: date,
      language: varchar(20),
      duration: int(11),
      summary: varchar(256),
      did: int(11),
      pid: int(11),
      swid: int(11),
      eid: int(11),
      PRIMARY_KEY(id),
      FOREIGN_KEY(did) REFERENCES director(id),
      FOREIGN_KEY(pid) REFERENCES producer(id),
      FOREIGN_KEY(swid) REFERENCES writer(id),
      FOREIGN_KEY(eid) REFERENCES editor(id))

actor(id:   int(11) auto_increment,
      name: varchar(20),
      dob:  date,
      PRIMARY_KEY(id))

acted_in(aid:  int(11),
         mid:  int(11),
         role: varchar(20),
         FOREIGN_KEY(aid) REFERENCES actor(id),
         FOREIGN_KEY(mid) REFERENCES movie(id))

director(id:   int(11) auto_increment,
         name: varchar(20),
         dob:  date,
         PRIMARY_KEY(id))

producer(id:   int(11) auto_increment,
         name: varchar(20),
         PRIMARY_KEY(id))

writer(id:   int(11) auto_increment,
       name: varchar(20),
       dob:  date,
       PRIMARY_KEY(id))

editor(id:   int(11) auto_increment,
       name: varchar(30),
       PRIMARY_KEY(id))

tags(mid: int(11),
     tag: varchar(20),
     FOREIGN_KEY(mid) REFERENCES movie(id))

user(id:        int(20) auto_increment,
```

```
     firstname: varchar(20),
     middle:    char(1),
     lastname:  varchar(20),
     username:  varchar(20),
     dob:       date,
     gender:    char(1),
     access:    varchar(20),
     PRIMARY_KEY(id))

rating(uid:    int(11),
       mid:    int(11),
       rating: int(11),
       FOREIGN_KEY(uid) REFERENCES user(id),
       FOREIGN_KEY(mid) REFERENCES movie(id))

review(uid:    int(11),
       mid:    int(11),
       review: varchar(256),
       FOREIGN_KEY(uid) REFERENCES user(id),
       FOREIGN_KEY(mid) REFERENCES movie(id))
```


id
summary
eid
did
title
pid
genre
Movies
release
language
duration
swid
aid
mid
role
Acted_In
aid
name
dob
Actors
1
N
1
1
1
1
1
1
Directed_By
did
name
dob
Directors
N
name
dob
Screenwritten_by
N
Screenwriters
swid
tagged_as
mid
Tags
tag
N
uid
N
mid
rated_by
rating
N
uid
N
mid
eviewed_by
review
N
1
1
Users
username
id
dob
lastname
gender
access
middlename
firstname