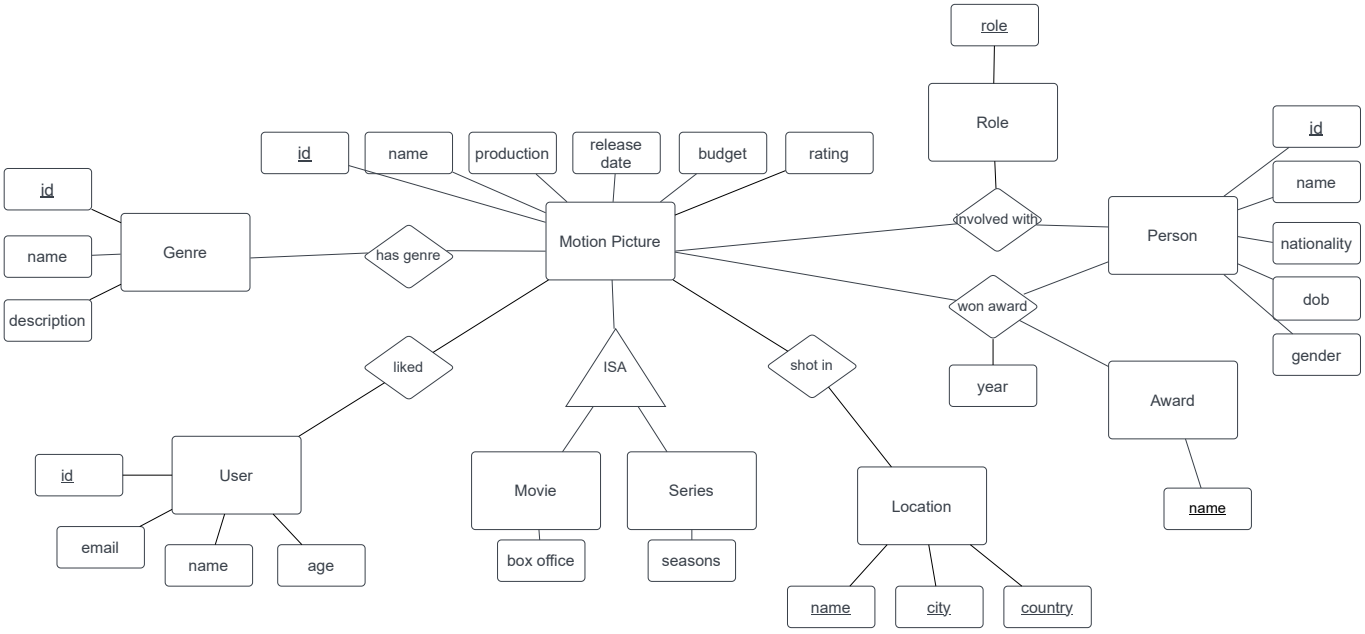
Project 1.1 Part 1

A) ER Diagram



ER Diagram Details

This project's ER diagram was made through draw.io

Primary keys

- Motion Picture: id
- Genre: id
- User: id
- Person: id
- Role: role
- Award: name
- Location: (name, city, country)

Candidate Keys

- Motion Picture: potentially some combination of (name, production, release date)
- Genre: name
- User: potentially (email)

Key Constraints

- Movie, Series ISA Motion Picture: covering constraint (yes), overlap constraint (disallowed)

Assumptions

- The system stores the rating of a movie as a single float (0-10), not as individual ratings.

B) MySQL Schema

```
-- Entity: MotionPicture
CREATE TABLE IF NOT EXISTS motion_picture (
    id INTEGER AUTO_INCREMENT PRIMARY KEYNOT NULL,
    name VARCHAR(255) NOT NULL,
    -- genres Genre[]
    production VARCHAR(255) NOT NULL,
    release_date DATETIME NOT NULL,
    budget FLOAT NOT NULL,
    -- awards Award[]
    -- shooting_locations ShootingLocation[]
    rating FLOAT NOT NULL,

    ADD CONSTRAINT RatingBetween0And10
    CHECK ( rating >= 0 AND rating <= 10 )
);

-- Entity: Movie
CREATE TABLE IF NOT EXISTS movie (
    id INTEGER PRIMARY KEYNOT NULL,
    box_office_collection FLOAT,

    FOREIGN KEY(id) REFERENCES motion_picture(id) ON DELETE CASCADE
);

-- Entity: TvSeries
CREATE TABLE IF NOT EXISTS tv_series (
    id INTEGER PRIMARY KEYNOT NULL,
    no_seasons INTEGER,
    FOREIGN KEY(id) REFERENCES motion_picture(id) ON DELETE CASCADE
);

-- Entity: Genre
CREATE TABLE IF NOT EXISTS genre (
    id INTEGER AUTO_INCREMENT PRIMARY KEYNOT NULL,
    name VARCHAR(32) NOT NULL,
    description VARCHAR(255) NOT NULL,
    -- motion_pictures MotionPicture[]

    UNIQUE (name)
);
```

```sql
-- Relation: MotionPicture + Genre
CREATE TABLE IF NOT EXISTS motion_picture_genre_association (
    motion_picture_id INTEGER NOT NULL,
    genre_id INTEGER NOT NULL,

    PRIMARY KEY(motion_picture_id, genre_id),
    FOREIGN KEY(motion_picture_id) REFERENCES motion_picture(id) ON DELETE CASCADE,
    FOREIGN KEY(genre_id) REFERENCES genre(id) ON DELETE CASCADE
)

-- Entity: Person
CREATE TABLE IF NOT EXISTS person (
    id INTEGER AUTO_INCREMENT PRIMARY KEYNOT NULL,
    name VARCHAR(255) NOT NULL,
    nationality VARCHAR(255) NOT NULL,
    dob DATE NOT NULL,
    gender VARCHAR(32) NOT NULL,
    -- roles Role[]
    -- awards Award[]
);

-- Entity: Role
CREATE TABLE IF NOT EXISTS role (
    person_id INTEGER NOT NULL,
    motion_picture_id INTEGER NOT NULL,
    role VARCHAR(255) NOT NULL,

    PRIMARY KEY(person_id, motion_picture_id, role),
    FOREIGN KEY(person_id) REFERENCES person(id) ON DELETE CASCADE,
    FOREIGN KEY(motion_picture_id) REFERENCES motion_picture(id) ON DELETE CASCADE
);

-- Entity: Award
CREATE TABLE IF NOT EXISTS award (
    person_id INTEGER NOT NULL,
    motion_picture_id INTEGER NOT NULL,
    name VARCHAR(255) NOT NULL,
    year_received INTEGER NOT NULL,

    PRIMARY KEY(person_id, motion_picture_id, name),
    FOREIGN KEY(person_id) REFERENCES person(id) ON DELETE CASCADE,
    FOREIGN KEY(motion_picture_id) REFERENCES motion_picture(id) ON DELETE CASCADE
);

-- Entity: ShootingLocation
CREATE TABLE IF NOT EXISTS shooting_location (
    motion_picture_id INTEGER NOT NULL,
    name VARCHAR(255) NOT NULL,
```

```
    city VARCHAR(255) NOT NULL,
    country VARCHAR(255) NOT NULL,

    FOREIGN KEY(motion_picture_id) REFERENCES motion_picture(id) ON DELETE CASCADE,
    PRIMARY KEY(motion_picture_id, name, city, country)
);

-- Entity: User
CREATE TABLE IF NOT EXISTS user (
    id INTEGER AUTO_INCREMENT PRIMARY KEYNOT NULL
    email VARCHAR(255) NOT NULL
    name VARCHAR(255) NOT NULL,
    age INTEGER NOT NULL, -- positive

    UNIQUE(email),
    CHECK (age >= 0)
);

CREATE TABLE IF NOT EXISTS motion_picture_like (
    user_id INTEGER NOT NULL,
    motion_picture_id INTEGER NOT NULL,

    PRIMARY KEY(user_id, motion_picture_id),
    FOREIGN KEY(user_id) REFERENCES user(id) ON DELETE CASCADE,
    FOREIGN KEY(motion_picture_id) REFERENCES motion_picture(id) ON DELETE CASCADE
);
```