**Tables and Relationships:**

1. **users Table:**
   * **Columns**: id, username, password, email, role
   * **Primary Key**: id
   * **Relationships**:
     + One-to-Many with ratings (One user can have many ratings).
2. **genres Table:**
   * **Columns**: id, name
   * **Primary Key**: id
   * **Relationships**:
     + Many-to-Many with movies through movie\_genres (A genre can belong to many movies).
3. **movies Table:**
   * **Columns**: id, title, description, image\_url, director\_name
   * **Primary Key**: id
   * **Relationships**:
     + Many-to-Many with genres through movie\_genres (A movie can have many genres).
     + One-to-Many with ratings (A movie can have many ratings).
4. **movie\_genres Table (Junction Table):**
   * **Columns**: movie\_id, genre\_id
   * **Primary Key**: Composite key of movie\_id and genre\_id
   * **Relationships**:
     + Many-to-One with movies (Each entry is associated with one movie).
     + Many-to-One with genres (Each entry is associated with one genre).
5. **ratings Table:**
   * **Columns**: id, movie\_id, user\_id, score, novelty, created\_at
   * **Primary Key**: id
   * **Relationships**:
     + Many-to-One with users (Each rating is associated with one user).
     + Many-to-One with movies (Each rating is associated with one movie).

**ERD Representation:**

* **Entities**:
  + Users
  + Genres
  + Movies
  + Ratings
  + Movie\_Genres (Junction Table)
* **Relationships**:
  + Users (1) → Ratings (N)
  + Movies (1) → Ratings (N)
  + Movies (M) → Genres (M) via Movie\_Genres (Junction Table)

**ERD Diagram Description:**

1. **Users**:
   * id (PK)
   * username
   * password
2. **Genres**:
   * id (PK)
   * name
3. **Movies**:
   * id (PK)
   * title
   * description
   * image\_url
   * director\_name
4. **Movie\_Genres** (Junction Table):
   * movie\_id (FK from Movies)
   * genre\_id (FK from Genres)
5. **Ratings**:
   * id (PK)
   * movie\_id (FK from Movies)
   * user\_id (FK from Users)
   * score
   * novelty
   * created\_at