Normalized Database Design for a Media Streaming Platform

Student Name: Mahla Entezari
Course: Databases - Fall 2024
Instructor: Dr. Mehrdad Ahmadzadeh Raji

May 3, 2025

1. Project Overview

This project involves the design, normalization, and implementation of a relational database system for a media streaming service. The goal is to create a robust and efficient schema using the Entity-Relationship (ER) model, ensure normalization up to 2NF, and implement the system in PostgreSQL using SQL commands.

2. EER Diagrams

The conceptual model was developed using Enhanced Entity-Relationship (EER) diagrams to represent entities such as Users, Media, Subscription, Ratings, and Comments, along with their relationships.

2.1 EER Diagram Part 1

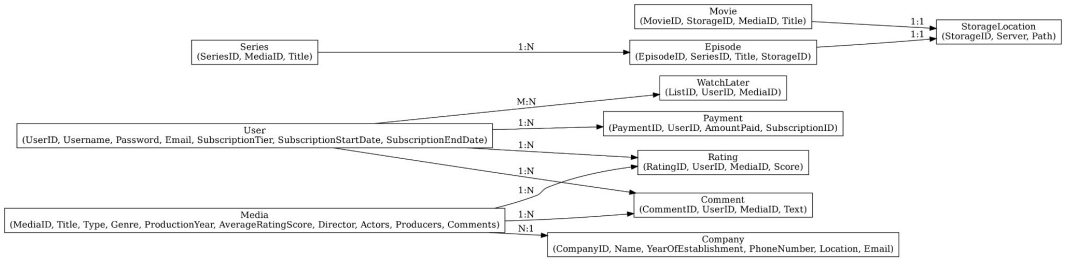

Figure 1: EER Diagram - User, Subscription, Media, and Comments

2.2 EER Diagram Part 2

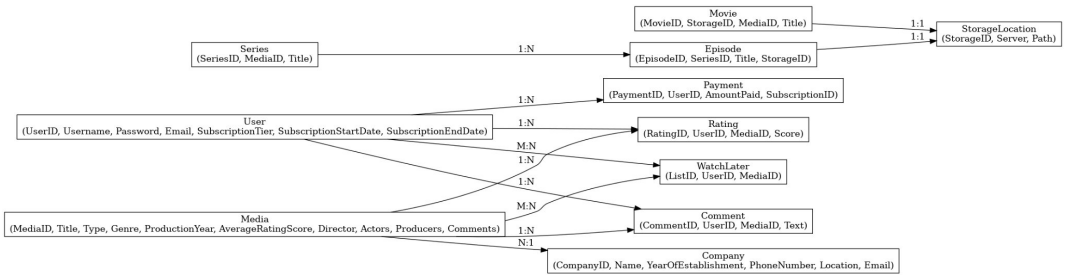

Figure 2: EER Diagram - Episodes, Movies, Series, and Storage

3. First Normal Form (1NF)

To satisfy 1NF:

- All attributes must be atomic.
- No multivalued or composite attributes are allowed.

In our initial schema, fields like ‘Actors‘, ‘Producers‘, and ‘Comments‘ were stored as text arrays. To convert to 1NF:

- Created separate tables: `MediaActor(MediaID, Actor)`, `MediaProducer(MediaID, Producer)`, `Comment(CommentID, UserID, MediaID, Text)`.

This ensured atomicity and removed repeating groups.

4. Second Normal Form (2NF)

2NF requires:

- Schema already in 1NF.
- No partial dependency of non-key attributes on part of a composite primary key.

In the 1NF schema, ‘SubscriptionTier‘, ‘StartDate‘, and ‘EndDate‘ depended only on ‘UserID‘ in a table where the composite key could include other fields like ‘PaymentID‘.

To fix this:

- Created a new table `Subscription(SubscriptionID, UserID, Tier, StartDate, EndDate)`.
- Updated related tables to remove redundant fields.

5. SQL Schema Overview

The normalized schema is implemented in PostgreSQL using SQL. Below is a sample:

```
CREATE TABLE Users (
    UserID SERIAL PRIMARY KEY,
    Username VARCHAR(255),
    Email VARCHAR(255),
    Password VARCHAR(255)
);

CREATE TABLE Subscription (
    SubscriptionID SERIAL PRIMARY KEY,
    UserID INT REFERENCES Users(UserID),
    Tier VARCHAR(50),
    StartDate DATE,
    EndDate DATE
);

CREATE TABLE Media (
    MediaID SERIAL PRIMARY KEY,
    Title VARCHAR(255),
    Genre VARCHAR(100),
    Type VARCHAR(50),
    ProductionYear INT,
    Director VARCHAR(255)
);
```

6. Sample Advanced SQL Queries

6.1 Top 5 Most Watched Media

```
SELECT m.Title, COUNT(*) AS Views
FROM WatchHistory w
JOIN Media m ON w.MediaID = m.MediaID
GROUP BY m.Title
ORDER BY Views DESC
LIMIT 5;
```

6.2 User Activity in Last 30 Days

```
SELECT u.Username, m.Title, w.Timestamp
FROM WatchHistory w
JOIN Users u ON w.UserID = u.UserID
JOIN Media m ON w.MediaID = m.MediaID
WHERE w.Timestamp >= NOW() - INTERVAL '30 days';
```

6.3 Average Rating per Genre

```
SELECT Genre, AVG(Score) AS AvgRating
FROM Media m
JOIN Rating r ON m.MediaID = r.MediaID
GROUP BY Genre;
```

7. Conclusion

This project demonstrates the transformation of a conceptual EER model into a relational schema, normalized to 2NF, and implemented in PostgreSQL. The structured design ensures data consistency, removes redundancy, and supports powerful SQL queries for media streaming insights.