Streamflix Database Schema Documentation

# Overview

The Streamflix database is designed to manage a movie streaming service, capturing information about movies, users, and their interactions via ratings. The schema supports key functionalities such as tracking viewership, user behavior, and movie performance metrics.

# Movies Table

Stores metadata for each movie in the catalog.

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Description |
| MovieID | INT | Unique identifier for each movie |
| Title | VARCHAR(500) | Movie title |
| Genres | VARCHAR(200) | Comma-separated genres (e.g., Action, Drama) |
| Language | VARCHAR(20) | Primary language of the movie |
| Country | VARCHAR(10) | Country of origin |
| TotalViews | INT | Number of views across all users |

Rationale: Designed to be scalable and support analysis of movie trends, genre preferences, and regional performance.

# Users Table

Captures demographic and behavioral data for each user.

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Description |
| UserID | INT | Unique identifier for each user |
| Age | INT | User's age |
| Gender | CHAR(1) | Gender (M, F, O) |
| Country | VARCHAR(10) | Country of residence |
| SubscriptionStatus | VARCHAR(20) | e.g., Free, Premium, Cancelled |
| TotalWatchTime | INT | Total time spent watching (in minutes) |
| Device | VARCHAR(50) | Device used to access content |

Rationale: Provides a profile for user segmentation and targeted recommendations.

# Ratings Table

Records individual movie ratings provided by users.

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Description |
| RatingID | INT | Unique identifier for each rating entry |
| UserID | INT | Foreign key referencing Users(UserID) |
| MovieID | INT | Foreign key referencing Movies(MovieID) |
| Rating | DECIMAL(2,1) | Rating value (e.g., 4.1, 3.5) |
| Timestamp | DATETIME | When the rating was submitted |

Rationale: Supports user feedback analysis, average rating calculation, and movie ranking features.

# Relationships & Constraints

- Ratings.UserID → Users.UserID  
- Ratings.MovieID → Movies.MovieID  
  
All primary and foreign keys are strongly typed and indexed for optimal join performance.

# Design Decisions & Rationale

1. Normalization: The schema is normalized to 3rd Normal Form (3NF) to reduce redundancy.  
2. Data Types: VARCHAR used for text flexibility; DECIMAL for precise ratings; DATETIME for timestamping.  
3. Scalability: Genres stored as strings initially for simplicity, but normalization is possible later.  
4. Performance: Proper indexing ensures fast JOIN operations.

# Future Enhancements

- Add WatchHistory table to track viewing per movie per user.  
- Normalize Genres into a separate table for better querying.  
- Track session data for behavioral analytics.