MY GAME LIST

DATABASE DESIGN PROJECT

Submitted by:

REG NO	**STUDENT NAME**
2023-BS-CS-064	ZAIN MAHMOOD
2024-BS-CS-366	HASSAN TARIQ

Section-A

BACHELOR OF SCIENCE

IN

COMPUTER SCIENCE

(SEMESTER-IV)

THE UNIVERSITY OF FAISALABAD

JUNE 2025

1. Project Overview

1.1 Introduction

My Game List is a web platform that helps gamers track, organize, and review video games. It works like a digital library where users can keep records of games they have played, are currently playing, or want to play in the future. The platform also includes social features that allow users to follow each other and share their gaming experiences.

1.2 Project Goals

The main goal of this project is to create a database that can:

- Store information about thousands of video games
- Track user gaming activities and progress
- Manage user reviews and ratings
- Handle social interactions between users
- Organize games into custom lists and categories

1.3 Target Users

- **Primary Users:** Gaming enthusiasts who want to organize their game collections
- **Secondary Users:** Game reviewers and content creators
- **Tertiary Users:** Casual gamers looking for game recommendations

2. System Features

2.1 Core Features

Game Library Management

- Users can mark games with different statuses (Currently Playing, Completed, On Hold, Dropped, Plan to Play)
- Track how many hours spent playing each game
- Give personal ratings to games (0-10 scale)
- View personal gaming statistics

Rating and Review System

- Write detailed reviews for games
- Rate games and see community ratings
- Read reviews from other users
- Edit and manage personal reviews

Social Features

- Follow other users to see their gaming activities

- Customize user profiles with avatars and display names
- Discover users with similar gaming interests

Custom Lists

- Create personal game collections (like "Best RPGs" or "Hidden Gems")
- Make lists public or private
- Organize games within lists in custom order

Game Database

- Store detailed information about games (title, description, release date, ratings)
- Support multiple gaming platforms (PC, PlayStation, Xbox, Nintendo, Mobile)
- Categorize games by genres and tags
- Include developer and publisher information
- Store game screenshots and images

3. Database Design Process

3.1 Requirements Analysis

What information do we need to store?

- Game details (name, description, release date, ratings, images)
- User profiles and login information
- Game tracking records (status, progress, personal ratings)
- User reviews and ratings
- Social connections between users
- Custom user-created lists
- Game categories (genres, platforms, publishers, developers)

What actions should the system support?

- Add games to personal library
- Write and edit reviews
- Follow other users
- Create custom game lists
- Search and filter games
- View gaming statistics

3.2 Entity-Relationship (ER) Model

Primary Entities:

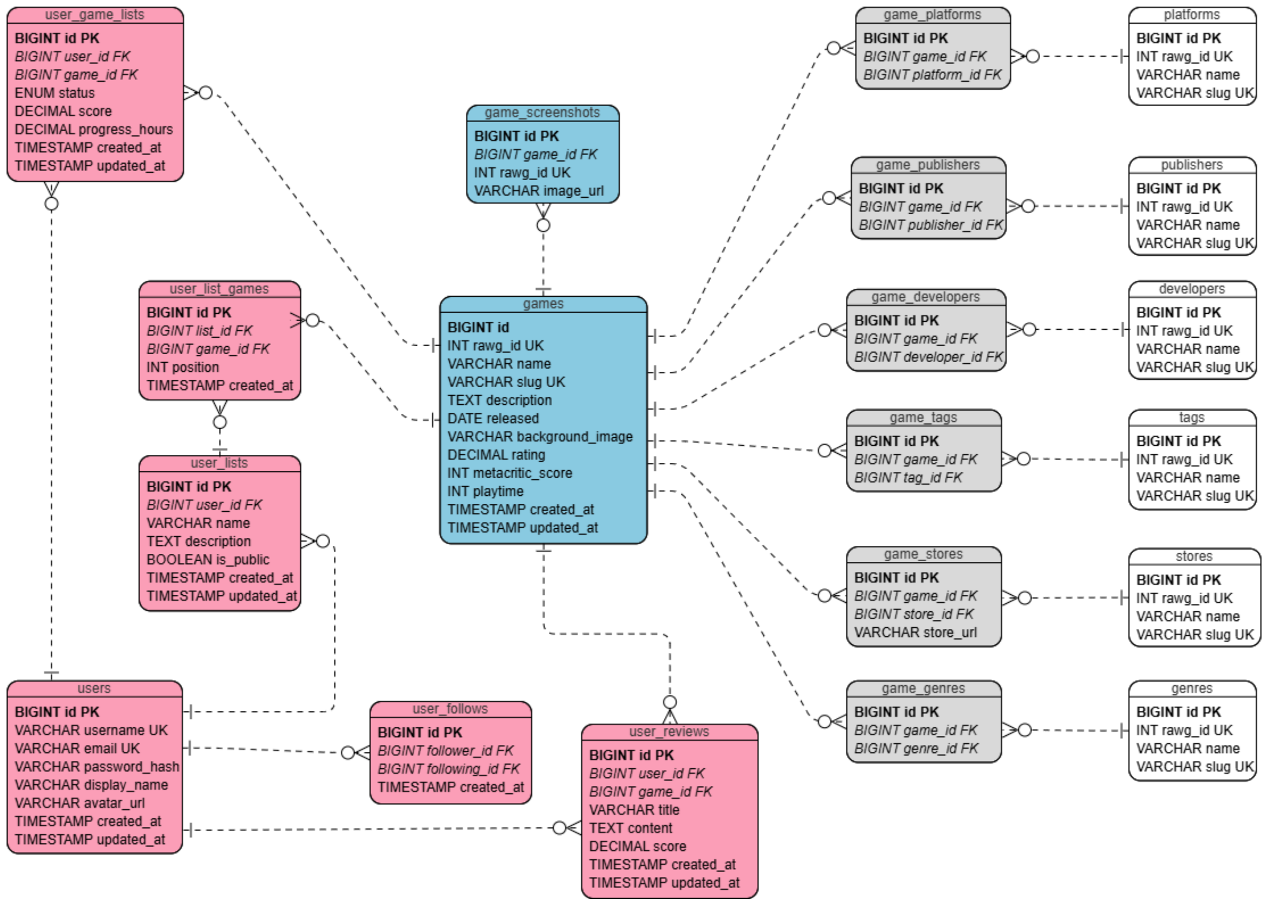
- **User:** People who use the platform
- **Game:** Video games in the database
- **Publisher:** Companies that publish games

- **Developer:** Companies that develop games
- **Platform:** Gaming systems (PC, PlayStation, Xbox, etc.)
- **Genre:** Game categories (Action, RPG, Strategy, etc.)
- **Tag:** Additional game descriptors
- **Store:** Digital stores where games are sold

Key Relationships:

- A User can track many Games (one-to-many)
- A Game can be tracked by many Users (many-to-many)
- A User can write many Reviews (one-to-many)
- A Game can have many Reviews (one-to-many)
- Users can follow other Users (many-to-many)
- Games can have multiple Publishers, Developers, Platforms, and Genres (many-to-many)

3.3 ER Diagram



4. Database Schema Design

4.1 Table Structure

Core Tables:

1. **users** - Stores user account information
   - id, username, email, password_hash, display_name, avatar_url, created_at, updated_at
2. **games** - Stores game information
   - id, rawg_id, name, slug, description, released, background_image, rating, metacritic_score, playtime, created_at, updated_at
3. **publishers** - Game publishing companies
   - id, rawg_id, name, slug
4. **developers** - Game development companies
   - id, rawg_id, name, slug
5. **platforms** - Gaming platforms
   - id, rawg_id, name, slug
6. **genres** - Game genres
   - id, rawg_id, name, slug
7. **tags** - Game tags
   - id, rawg_id, name, slug
8. **stores** - Digital game stores
   - id, rawg_id, name, slug

Junction Tables (for many-to-many relationships):

- game_publishers, game_developers, game_platforms, game_genres, game_tags, game_stores

User Interaction Tables:

- user_game_lists (tracks user's game library)
- user_reviews (stores user reviews)
- user_follows (manages user following relationships)
- user_lists (custom user-created lists)
- user_list_games (games within custom lists)

Media Tables:

- game_screenshots (stores game images)

4.2 Data Types and Constraints

Data Types Used:

- BIGINT - For primary keys and foreign keys (allows for large numbers)
- VARCHAR(255) - For names, usernames, emails
- TEXT - For long descriptions and review content
- DECIMAL(3,2) - For precise ratings (0.00 to 10.00)

- ENUM - For status values (Currently Playing, Completed, etc.)
- TIMESTAMP - For tracking creation and update times
- DATE - For game release dates
- INT - For scores and counters
- BOOLEAN - For true/false values (like public/private)

Key Constraints:

- Primary keys on all tables for unique identification
- Foreign keys to maintain data relationships
- Unique constraints on usernames, emails, and slugs
- Check constraints for valid rating ranges

5. Normalization Analysis

5.1 First Normal Form (1NF)

Requirements:

- All data must be atomic (no multiple values in single field)
- Each row must be unique
- No repeating groups

Our Design: ✓ All attributes store single values (username, email, game name) ✓ Primary keys ensure unique rows ✓ Junction tables handle multiple relationships (game can have multiple genres)

Result: Our schema satisfies 1NF.

5.2 Second Normal Form (2NF)

Requirements:

- Must satisfy 1NF
- All non-key attributes must depend on the entire primary key

Our Design: ✓ Satisfies 1NF ✓ All attributes depend fully on primary keys ✓ Junction tables properly handle many-to-many relationships

Result: Our schema satisfies 2NF.

5.3 Third Normal Form (3NF)

Requirements:

- Must satisfy 2NF
- No transitive dependencies (non-key attributes should not depend on other non-key attributes)

Our Design: ✓ Satisfies 2NF ✓ Reference data stored in separate tables (publishers, genres, etc.) ✓ No transitive dependencies exist

Result: Our schema satisfies 3NF.

5.4 Benefits Achieved

- **Reduced Data Redundancy:** Publisher names stored once, referenced by ID
- **Improved Data Consistency:** Changes to publisher name update everywhere
- **Better Storage Efficiency:** No duplicate data storage
- **Enhanced Data Integrity:** Foreign key constraints prevent invalid data

6. Database Implementation

6.1 SQL Schema Creation

```
create database MyGameList;
use MyGameList;

CREATE TABLE users (
    id BIGINT PRIMARY KEY AUTO_INCREMENT,
    username VARCHAR(50) UNIQUE NOT NULL,
    email VARCHAR(255) UNIQUE NOT NULL,
    password_hash VARCHAR(255) NOT NULL,
    display_name VARCHAR(100),
    avatar_url VARCHAR(500),
    created_at TIMESTAMP DEFAULT
CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT
CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP,

    INDEX idx_username (username),
    INDEX idx_email (email)
);

CREATE TABLE publishers (
    id BIGINT PRIMARY KEY AUTO_INCREMENT,
    rawg_id INT UNIQUE,
    name VARCHAR(255) NOT NULL,
    slug VARCHAR(255) UNIQUE NOT NULL,

    INDEX idx_name (name),
    INDEX idx_rawg_id (rawg_id)
);

CREATE TABLE developers (
    id BIGINT PRIMARY KEY AUTO_INCREMENT,
    rawg_id INT UNIQUE,
    name VARCHAR(255) NOT NULL,
    slug VARCHAR(255) UNIQUE NOT NULL,

    INDEX idx_name (name),
    INDEX idx_rawg_id (rawg_id)
);

CREATE TABLE platforms (
    id BIGINT PRIMARY KEY AUTO_INCREMENT,
    rawg_id INT UNIQUE,
    name VARCHAR(100) NOT NULL,
    slug VARCHAR(100) UNIQUE NOT NULL,

    INDEX idx_name (name),
    INDEX idx_rawg_id (rawg_id)
);

CREATE TABLE genres (
    id BIGINT PRIMARY KEY AUTO_INCREMENT,
    rawg_id INT UNIQUE,
    name VARCHAR(100) NOT NULL,
    slug VARCHAR(100) UNIQUE NOT NULL,

    INDEX idx_name (name),
    INDEX idx_rawg_id (rawg_id)
);

CREATE TABLE tags (
    id BIGINT PRIMARY KEY AUTO_INCREMENT,
    rawg_id INT UNIQUE,
    name VARCHAR(100) NOT NULL,
    slug VARCHAR(100) UNIQUE NOT NULL,

    INDEX idx_name (name),
    INDEX idx_rawg_id (rawg_id)
);

CREATE TABLE stores (
    id BIGINT PRIMARY KEY AUTO_INCREMENT,
    rawg_id INT UNIQUE,
    name VARCHAR(100) NOT NULL,
    slug VARCHAR(100) UNIQUE NOT NULL,

    INDEX idx_name (name),
    INDEX idx_rawg_id (rawg_id)
);

CREATE TABLE games (
    id BIGINT PRIMARY KEY AUTO_INCREMENT,
    rawg_id INT UNIQUE,
    name VARCHAR(255) NOT NULL,
    slug VARCHAR(255) UNIQUE NOT NULL,
    description TEXT,
    released DATE,
    background_image VARCHAR(500),
    rating DECIMAL(3,2) DEFAULT 0.00,
    metacritic_score INT,
    playtime INT DEFAULT 0,
    created_at TIMESTAMP DEFAULT
CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT
CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP,

    INDEX idx_name (name),
    INDEX idx_slug (slug),
    INDEX idx_released (released),
    INDEX idx_rating (rating),
    INDEX idx_rawg_id (rawg_id)
);

CREATE TABLE game_publishers (
    id BIGINT PRIMARY KEY AUTO_INCREMENT,
    game_id BIGINT NOT NULL,
    publisher_id BIGINT NOT NULL,

    FOREIGN KEY (game_id) REFERENCES games(id) ON
DELETE CASCADE,
    FOREIGN KEY (publisher_id) REFERENCES
publishers(id) ON DELETE CASCADE,
    UNIQUE KEY unique_game_publisher (game_id,
publisher_id)
);

CREATE TABLE game_developers (
    id BIGINT PRIMARY KEY AUTO_INCREMENT,
    game_id BIGINT NOT NULL,
    developer_id BIGINT NOT NULL,

    FOREIGN KEY (game_id) REFERENCES games(id) ON
DELETE CASCADE,
    FOREIGN KEY (developer_id) REFERENCES
developers(id) ON DELETE CASCADE,
    UNIQUE KEY unique_game_developer (game_id,
developer_id)
);

CREATE TABLE game_platforms (
    id BIGINT PRIMARY KEY AUTO_INCREMENT,
    game_id BIGINT NOT NULL,
    platform_id BIGINT NOT NULL,

    FOREIGN KEY (game_id) REFERENCES games(id) ON
DELETE CASCADE,
    FOREIGN KEY (platform_id) REFERENCES platforms(id)
ON DELETE CASCADE,
    UNIQUE KEY unique_game_platform (game_id,
platform_id)
);

CREATE TABLE game_genres (
    id BIGINT PRIMARY KEY AUTO_INCREMENT,
    game_id BIGINT NOT NULL,
    genre_id BIGINT NOT NULL,

    FOREIGN KEY (game_id) REFERENCES games(id) ON
DELETE CASCADE,
    FOREIGN KEY (genre_id) REFERENCES genres(id) ON
DELETE CASCADE,
    UNIQUE KEY unique_game_genre (game_id, genre_id)
);

CREATE TABLE game_tags (
    id BIGINT PRIMARY KEY AUTO_INCREMENT,
    game_id BIGINT NOT NULL,
    tag_id BIGINT NOT NULL,
    FOREIGN KEY (game_id) REFERENCES games(id) ON
DELETE CASCADE,
    FOREIGN KEY (tag_id) REFERENCES tags(id) ON
DELETE CASCADE,
    UNIQUE KEY unique_game_tag (game_id, tag_id)
);

CREATE TABLE game_stores (
    id BIGINT PRIMARY KEY AUTO_INCREMENT,
    game_id BIGINT NOT NULL,
    store_id BIGINT NOT NULL,
    store_url VARCHAR(500),

    FOREIGN KEY (game_id) REFERENCES games(id) ON
DELETE CASCADE,
    FOREIGN KEY (store_id) REFERENCES stores(id) ON
DELETE CASCADE,
    UNIQUE KEY unique_game_store (game_id, store_id)
);

CREATE TABLE user_game_lists (
    id BIGINT PRIMARY KEY AUTO_INCREMENT,
    user_id BIGINT NOT NULL,
    game_id BIGINT NOT NULL,
    status ENUM('playing', 'completed', 'on_hold', 'dropped',
'plan_to_play') NOT NULL,
    score DECIMAL(3,1) CHECK (score >= 1.0 AND score <=
10.0),
    progress_hours DECIMAL(8,2) DEFAULT 0,
    created_at TIMESTAMP DEFAULT
CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT
CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP,

    FOREIGN KEY (user_id) REFERENCES users(id) ON
DELETE CASCADE,
    FOREIGN KEY (game_id) REFERENCES games(id) ON
DELETE CASCADE,
    UNIQUE KEY unique_user_game (user_id, game_id),
    INDEX idx_user_id (user_id),
    INDEX idx_status (status)
);

CREATE TABLE user_reviews (
    id BIGINT PRIMARY KEY AUTO_INCREMENT,
    user_id BIGINT NOT NULL,
    game_id BIGINT NOT NULL,
    title VARCHAR(255),
    content TEXT NOT NULL,
    score DECIMAL(3,1) CHECK (score >= 1.0 AND score <=
10.0),
    created_at TIMESTAMP DEFAULT
CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT
CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP,

    FOREIGN KEY (user_id) REFERENCES users(id) ON
DELETE CASCADE,
    FOREIGN KEY (game_id) REFERENCES games(id) ON
DELETE CASCADE,
    UNIQUE KEY unique_user_game_review (user_id,
game_id),
    INDEX idx_user_id (user_id),
    INDEX idx_game_id (game_id)
);

CREATE TABLE user_follows (
    id BIGINT PRIMARY KEY AUTO_INCREMENT,
    follower_id BIGINT NOT NULL,
    following_id BIGINT NOT NULL,
    created_at TIMESTAMP DEFAULT
CURRENT_TIMESTAMP,

    FOREIGN KEY (follower_id) REFERENCES users(id) ON
DELETE CASCADE,
    FOREIGN KEY (following_id) REFERENCES users(id)
ON DELETE CASCADE,
    UNIQUE KEY unique_follow (follower_id, following_id)
);

CREATE TABLE user_lists (
    id BIGINT PRIMARY KEY AUTO_INCREMENT,
    user_id BIGINT NOT NULL,
    name VARCHAR(255) NOT NULL,
    description TEXT,
    is_public BOOLEAN DEFAULT TRUE,
    created_at TIMESTAMP DEFAULT
CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT
CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP,

    FOREIGN KEY (user_id) REFERENCES users(id) ON
DELETE CASCADE,
    INDEX idx_user_id (user_id)
);

CREATE TABLE user_list_games (
    id BIGINT PRIMARY KEY AUTO_INCREMENT,
    list_id BIGINT NOT NULL,
    game_id BIGINT NOT NULL,
    position INT,
    created_at TIMESTAMP DEFAULT
CURRENT_TIMESTAMP,

    FOREIGN KEY (list_id) REFERENCES user_lists(id) ON
DELETE CASCADE,
    FOREIGN KEY (game_id) REFERENCES games(id) ON
DELETE CASCADE,
    UNIQUE KEY unique_list_game (list_id, game_id)
);

CREATE TABLE game_screenshots (
    id BIGINT PRIMARY KEY AUTO_INCREMENT,
    game_id BIGINT NOT NULL,
    rawg_id INT UNIQUE,
    image_url VARCHAR(500) NOT NULL,

    FOREIGN KEY (game_id) REFERENCES games(id) ON
DELETE CASCADE,
    INDEX idx_game_id (game_id)
);
```

6.2 Indexing Strategy

Primary Indexes:

- Primary key indexes on all tables (automatic)

Secondary Indexes:

- Unique indexes on usernames, emails, and slugs
- Foreign key indexes for efficient joins
- Composite indexes on frequently searched combinations
- Full-text search indexes for game names and descriptions

Benefits:

- Faster query performance
- Efficient joins between tables
- Quick user authentication
- Fast game searches

7. Testing and Validation

7.1 Sample Data Insertion

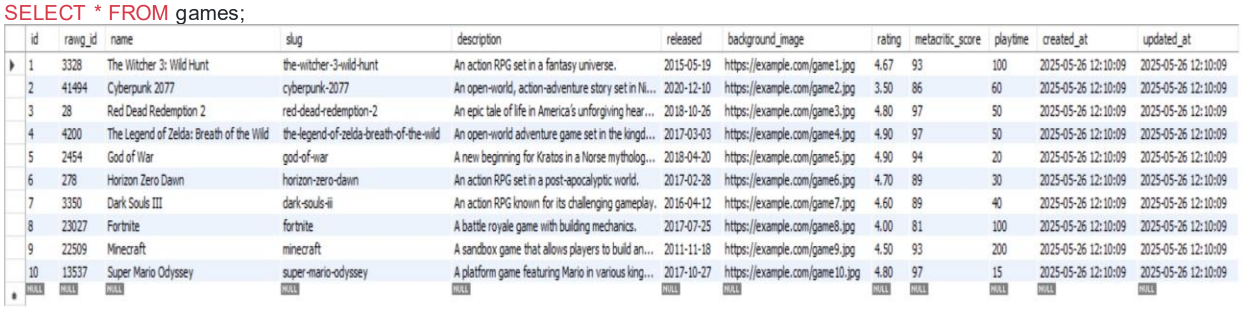
```
INSERT INTO publishers (rawg_id, name, slug) VALUES
(1, 'CD Projekt', 'cd-projekt'),
(2, 'Rockstar Games', 'rockstar-games'),
(3, 'Nintendo', 'nintendo'),
(4, 'Sony Interactive Entertainment', 'sony-interactive-
entertainment'),
(5, 'Bandai Namco Entertainment', 'bandai-namco-
entertainment'),
(6, 'Epic Games', 'epic-games'),
```

```sql
(7, 'Microsoft Studios', 'microsoft-studios');

INSERT INTO developers (rawg_id, name, slug) VALUES
(1, 'CD Projekt Red', 'cd-projekt-red'),
(2, 'Rockstar Studios', 'rockstar-studios'),
(3, 'Nintendo EPD', 'nintendo-epd'),
(4, 'Santa Monica Studio', 'santa-monica-studio'),
(5, 'Guerrilla Games', 'guerrilla-games'),
(6, 'FromSoftware', 'fromsoftware'),
(7, 'Epic Games', 'epic-games'),
(8, 'Mojang Studios', 'mojang-studios');

INSERT INTO platforms (rawg_id, name, slug) VALUES
(1, 'PC', 'pc'),
(2, 'PlayStation 4', 'playstation-4'),
(3, 'PlayStation 5', 'playstation-5'),
(4, 'Xbox One', 'xbox-one'),
(5, 'Xbox Series X', 'xbox-series-x'),
(6, 'Nintendo Switch', 'nintendo-switch');

INSERT INTO genres (rawg_id, name, slug) VALUES
(1, 'Action', 'action'),
(2, 'RPG', 'rpg'),
(3, 'Adventure', 'adventure'),
(4, 'Shooter', 'shooter'),
(5, 'Platformer', 'platformer'),
(6, 'Sandbox', 'sandbox'),
(7, 'Survival', 'survival'),
(8, 'Battle Royale', 'battle-royale');

INSERT INTO tags (rawg_id, name, slug) VALUES
(1, 'Open World', 'open-world'),
(2, 'Fantasy', 'fantasy'),
(3, 'Sci-Fi', 'sci-fi'),
(4, 'Multiplayer', 'multiplayer'),
(5, 'Singleplayer', 'singleplayer');

INSERT INTO stores (rawg_id, name, slug) VALUES
(1, 'Steam', 'steam'),
(2, 'Epic Games Store', 'epic-games-store'),
(3, 'PlayStation Store', 'playstation-store'),
(4, 'Xbox Store', 'xbox-store'),
(5, 'Nintendo eShop', 'nintendo-eshop');

INSERT INTO games (rawg_id, name, slug, description,
released, background_image, rating, metacritic_score,
playtime)
VALUES (
    3328,
    'The Witcher 3: Wild Hunt',
    'the-witcher-3-wild-hunt',
    'An action RPG set in a fantasy universe.',
    '2015-05-19',
    'https://example.com/game1.jpg',
    4.67,
    93,
    100
);
INSERT INTO game_publishers (game_id, publisher_id)
VALUES (1, 1);
INSERT INTO game_developers (game_id, developer_id)
VALUES (1, 1);
INSERT INTO game_platforms (game_id, platform_id)
VALUES (1, 1), (1, 2), (1, 4), (1, 6);
INSERT INTO game_genres (game_id, genre_id) VALUES
(1, 1), (1, 2);
INSERT INTO game_tags (game_id, tag_id) VALUES (1, 1),
(1, 2), (1, 5);
INSERT INTO game_stores (game_id, store_id, store_url)
VALUES
(1, 1, 'https://store.example.com/game1'),
(1, 3, 'https://store.example.com/game1_ps'),
(1, 4, 'https://store.example.com/game1_xbox'),
(1, 5, 'https://store.example.com/game1_switch');

INSERT INTO games (rawg_id, name, slug, description,
released, background_image, rating, metacritic_score,
playtime)
VALUES (
    41494,
    'Cyberpunk 2077',
    'cyberpunk-2077',
    'An open-world, action-adventure story set in Night City.',
    '2020-12-10',
    'https://example.com/game2.jpg',
    3.5,
    86,
    60
);
INSERT INTO game_publishers (game_id, publisher_id)
VALUES (2, 1);
INSERT INTO game_developers (game_id, developer_id)
VALUES (2, 1);
INSERT INTO game_platforms (game_id, platform_id)
VALUES (2, 1), (2, 2), (2, 3), (2, 4), (2, 5);
INSERT INTO game_genres (game_id, genre_id) VALUES
(2, 1), (2, 2);
INSERT INTO game_tags (game_id, tag_id) VALUES (2, 1),
(2, 3), (2, 5);
INSERT INTO game_stores (game_id, store_id, store_url)
VALUES
(2, 1, 'https://store.example.com/game2'),
(2, 3, 'https://store.example.com/game2_ps'),
(2, 4, 'https://store.example.com/game2_xbox');

INSERT INTO games (rawg_id, name, slug, description,
released, background_image, rating, metacritic_score,
playtime)
VALUES (
    28,
    'Red Dead Redemption 2',
    'red-dead-redemption-2',
    'An epic tale of life in America’s unforgiving heartland.',
    '2018-10-26',
    'https://example.com/game3.jpg',
    4.8,
    97,
    50
);
INSERT INTO game_publishers (game_id, publisher_id)
VALUES (3, 2);
INSERT INTO game_developers (game_id, developer_id)
VALUES (3, 2);
INSERT INTO game_platforms (game_id, platform_id)
VALUES (3, 1), (3, 2), (3, 4);
INSERT INTO game_genres (game_id, genre_id) VALUES
(3, 1), (3, 3);
INSERT INTO game_tags (game_id, tag_id) VALUES (3, 1),
(3, 5);
INSERT INTO game_stores (game_id, store_id, store_url)
VALUES
(3, 1, 'https://store.example.com/game3'),
(3, 3, 'https://store.example.com/game3_ps'),
(3, 4, 'https://store.example.com/game3_xbox');

INSERT INTO games (rawg_id, name, slug, description,
released, background_image, rating, metacritic_score,
playtime)
VALUES (
    4200,
    'The Legend of Zelda: Breath of the Wild',
    'the-legend-of-zelda-breath-of-the-wild',
    'An open-world adventure game set in the kingdom of
Hyrule.',
    '2017-03-03',
    'https://example.com/game4.jpg',
    4.9,
    97,
    50
);
INSERT INTO game_publishers (game_id, publisher_id)
VALUES (4, 3);
INSERT INTO game_developers (game_id, developer_id)
VALUES (4, 3);
INSERT INTO game_platforms (game_id, platform_id)
VALUES (4, 6);
INSERT INTO game_genres (game_id, genre_id) VALUES
(4, 1), (4, 3);
INSERT INTO game_tags (game_id, tag_id) VALUES (4, 1),
(4, 2), (4, 5);
INSERT INTO game_stores (game_id, store_id, store_url)
VALUES
(4, 5, 'https://store.example.com/game4_switch');

INSERT INTO games (rawg_id, name, slug, description,
released, background_image, rating, metacritic_score,
playtime)
VALUES (
    2454,
    'God of War',
    'god-of-war',
    'A new beginning for Kratos in a Norse mythology setting.',
    '2018-04-20',
    'https://example.com/game5.jpg',
    4.9,
    94,
    20
);
INSERT INTO game_publishers (game_id, publisher_id)
VALUES (5, 4);
INSERT INTO game_developers (game_id, developer_id)
VALUES (5, 4);
INSERT INTO game_platforms (game_id, platform_id)
VALUES (5, 2), (5, 3);
INSERT INTO game_genres (game_id, genre_id) VALUES
(5, 1), (5, 3);
INSERT INTO game_tags (game_id, tag_id) VALUES (5, 2),
(5, 5);
INSERT INTO game_stores (game_id, store_id, store_url)
VALUES
(5, 3, 'https://store.example.com/game5_ps');

INSERT INTO games (rawg_id, name, slug, description,
released, background_image, rating, metacritic_score,
playtime)
VALUES (
    278,
    'Horizon Zero Dawn',
    'horizon-zero-dawn',
    'An action RPG set in a post-apocalyptic world.',
    '2017-02-28',
    'https://example.com/game6.jpg',
    4.7,
    89,
    30
);
INSERT INTO game_publishers (game_id, publisher_id)
VALUES (6, 4);
INSERT INTO game_developers (game_id, developer_id)
VALUES (6, 5);
INSERT INTO game_platforms (game_id, platform_id)
VALUES (6, 1), (6, 2);
INSERT INTO game_genres (game_id, genre_id) VALUES
(6, 1), (6, 2);
INSERT INTO game_tags (game_id, tag_id) VALUES (6, 1),
(6, 3), (6, 5);
INSERT INTO game_stores (game_id, store_id, store_url)
VALUES
(6, 1, 'https://store.example.com/game6_pc'),
(6, 3, 'https://store.example.com/game6_ps');

INSERT INTO games (rawg_id, name, slug, description,
released, background_image, rating, metacritic_score,
playtime)
VALUES (
    3350,
    'Dark Souls III',
    'dark-souls-iii',
    'An action RPG known for its challenging gameplay.',
    '2016-04-12',
    'https://example.com/game7.jpg',
    4.6,
    89,
    40
);
INSERT INTO game_publishers (game_id, publisher_id)
VALUES (7, 5);
INSERT INTO game_developers (game_id, developer_id)
VALUES (7, 6);
INSERT INTO game_platforms (game_id, platform_id)
VALUES (7, 1), (7, 2), (7, 4);
INSERT INTO game_genres (game_id, genre_id) VALUES
(7, 1), (7, 2);
INSERT INTO game_tags (game_id, tag_id) VALUES (7, 2),
(7, 5);
INSERT INTO game_stores (game_id, store_id, store_url)
VALUES
(7, 1, 'https://store.example.com/game7_pc'),
(7, 3, 'https://store.example.com/game7_ps'),
(7, 4, 'https://store.example.com/game7_xbox');

INSERT INTO games (rawg_id, name, slug, description,
released, background_image, rating, metacritic_score,
playtime)
VALUES (
    23027,
    'Fortnite',
    'fortnite',
    'A battle royale game with building mechanics.',
    '2017-07-25',
    'https://example.com/game8.jpg',
    4.0,
    81,
    100
);
INSERT INTO game_publishers (game_id, publisher_id)
VALUES (8, 6);
INSERT INTO game_developers (game_id, developer_id)
VALUES (8, 7);
INSERT INTO game_platforms (game_id, platform_id)
VALUES (8, 1), (8, 2), (8, 3), (8, 4), (8, 5), (8, 6);
INSERT INTO game_genres (game_id, genre_id) VALUES
(8, 4), (8, 8);
INSERT INTO game_tags (game_id, tag_id) VALUES (8, 4);
INSERT INTO game_stores (game_id, store_id, store_url)
VALUES
(8, 2, 'https://store.example.com/game8_epic'),
(8, 3, 'https://store.example.com/game8_ps'),
(8, 4, 'https://store.example.com/game8_xbox'),
(8, 5, 'https://store.example.com/game8_switch');
```

```sql
INSERT INTO games (rawg_id, name, slug, description,
released, background_image, rating, metacritic_score,
playtime)
VALUES (
    22509,
    'Minecraft',
    'minecraft',
    'A sandbox game that allows players to build and
explore.',
    '2011-11-18',
    'https://example.com/game9.jpg',
    4.5,
    93,
    200
);
INSERT INTO game_publishers (game_id, publisher_id)
VALUES (9, 7);
INSERT INTO game_developers (game_id, developer_id)
VALUES (9, 8);
INSERT INTO game_platforms (game_id, platform_id)
VALUES (9, 1), (9, 2), (9, 4), (9, 6);
INSERT INTO game_genres (game_id, genre_id) VALUES
(9, 6), (9, 7);
INSERT INTO game_tags (game_id, tag_id) VALUES (9, 4),
(9, 5);
INSERT INTO game_stores (game_id, store_id, store_url)
VALUES
(9, 1, 'https://store.example.com/game9_pc'),
(9, 3, 'https://store.example.com/game9_ps'),
(9, 4, 'https://store.example.com/game9_xbox'),
(9, 5, 'https://store.example.com/game9_switch');

INSERT INTO games (rawg_id, name, slug, description,
released, background_image, rating, metacritic_score,
playtime)
VALUES (
    13537,
    'Super Mario Odyssey',
    'super-mario-odyssey',
    'A platform game featuring Mario in various kingdoms.',
    '2017-10-27',
    'https://example.com/game10.jpg',
    4.8,
    97,
    15
);
INSERT INTO game_publishers (game_id, publisher_id)
VALUES (10, 3);
INSERT INTO game_developers (game_id, developer_id)
VALUES (10, 3);
INSERT INTO game_platforms (game_id, platform_id)
VALUES (10, 6);
INSERT INTO game_genres (game_id, genre_id) VALUES
(10, 5);
INSERT INTO game_tags (game_id, tag_id) VALUES (10,
5);
INSERT INTO game_stores (game_id, store_id, store_url)
VALUES
(10, 5, 'https://store.example.com/game10_switch');

INSERT INTO users (username,  email, password_hash,
display_name)
VALUES ('gamer1', 'gamer1@example.com',
'hashed_password', 'Gamer One');

INSERT INTO user_game_lists (user_id, game_id, status,
score, progress_hours)
VALUES
(1, 1, 'completed', 9.5, 120),
(1, 2, 'playing', 7.0, 20),
(1, 3, 'plan_to_play', NULL, 0);

INSERT INTO user_reviews (user_id, game_id, title,
content, score)
VALUES
(1, 1, 'Amazing Game', 'One of the best RPGs I have ever
played.', 9.5),
(1, 2, 'Mixed Feelings', 'Great world but buggy at launch.',
7.0);
```

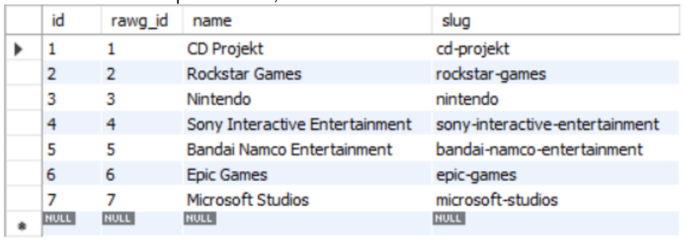
7.2 Test Queries

```sql
SELECT * FROM games;
```



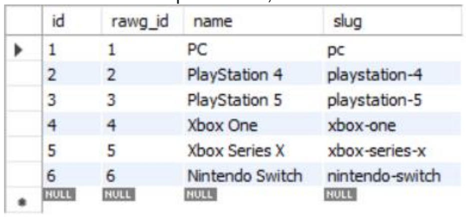
id	rawg_id	name	slug	description	released	background_image	rating	metacritic_score	playtime	created_at	updated_at
1	3328	The Witcher 3: Wild Hunt	the-witcher-3-wild-hunt	An action RPG set in a fantasy universe.	2015-05-19	https://example.com/game1.jpg	4.67	93	100	2025-05-26 12:10:09	2025-05-26 12:10:09
2	41494	Cyberpunk 2077	cyberpunk-2077	An open-world, action-adventure story set in Ni...	2020-12-10	https://example.com/game2.jpg	3.50	86	60	2025-05-26 12:10:09	2025-05-26 12:10:09
3	28	Red Dead Redemption 2	red-dead-redemption-2	An epic tale of life in America's unforgiving hear...	2018-10-26	https://example.com/game3.jpg	4.80	97	50	2025-05-26 12:10:09	2025-05-26 12:10:09
4	4200	The Legend of Zelda: Breath of the Wild	the-legend-of-zelda-breath-of-the-wild	An open-world adventure game set in the kingd...	2017-03-03	https://example.com/game4.jpg	4.90	97	50	2025-05-26 12:10:09	2025-05-26 12:10:09
5	2454	God of War	god-of-war	A new beginning for Kratos in a Norseytholog...	2018-04-20	https://example.com/game5.jpg	4.90	94	20	2025-05-26 12:10:09	2025-05-26 12:10:09
6	278	Horizon Zero Dawn	horizon-zero-dawn	An action RPG set in a post-apocalyptic world.	2017-02-28	https://example.com/game6.jpg	4.70	89	30	2025-05-26 12:10:09	2025-05-26 12:10:09
7	3350	Dark Souls III	dark-souls-iii	An action RPG known for its challenging gameplay.	2016-04-12	https://example.com/game7.jpg	4.60	89	40	2025-05-26 12:10:09	2025-05-26 12:10:09
8	23027	Fortnite	fortnite	A battle royale game with building mechanics.	2017-07-25	https://example.com/game8.jpg	4.00	81	100	2025-05-26 12:10:09	2025-05-26 12:10:09
9	22509	Minecraft	minecraft	A sandbox game that allows players to build an...	2011-11-18	https://example.com/game9.jpg	4.50	93	200	2025-05-26 12:10:09	2025-05-26 12:10:09
10	13537	Super Mario Odyssey	super-mario-odyssey	A platform game featuring Mario in various king...	2017-10-27	https://example.com/game10.jpg	4.80	97	15	2025-05-26 12:10:09	2025-05-26 12:10:09
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

```sql
SELECT * FROM publishers;
```



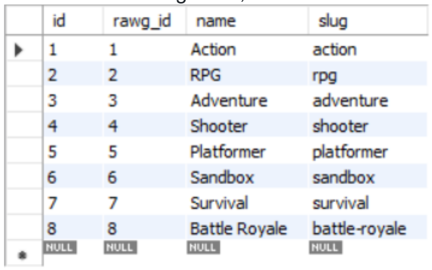
id	rawg_id	name	slug
1	1	CD Projekt	cd-projekt
2	2	Rockstar Games	rockstar-games
3	3	Nintendo	nintendo
4	4	Sony Interactive Entertainment	sony-interactive-entertainment
5	5	Bandai Namco Entertainment	bandai-namco-entertainment
6	6	Epic Games	epic-games
7	7	Microsoft Studios	microsoft-studios
NULL	NULL	NULL	NULL

```sql
SELECT * FROM platforms;
```



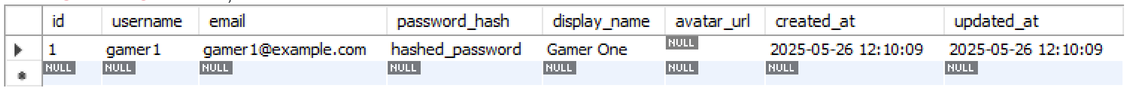
id	rawg_id	name	slug
1	1	PC	pc
2	2	PlayStation 4	playstation-4
3	3	PlayStation 5	playstation-5
4	4	Xbox One	xbox-one
5	5	Xbox Series X	xbox-series-x
6	6	Nintendo Switch	nintendo-switch
NULL	NULL	NULL	NULL

```sql
SELECT * FROM genres;
```



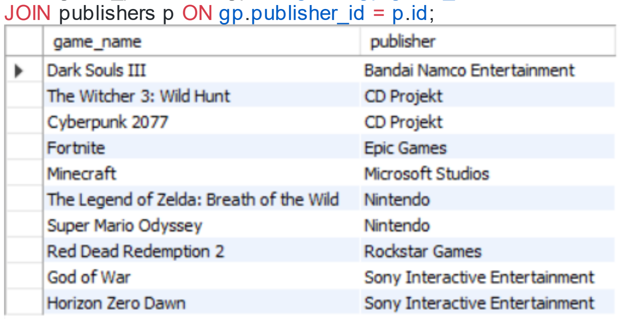
id	rawg_id	name	slug
1	1	Action	action
2	2	RPG	rpg
3	3	Adventure	adventure
4	4	Shooter	shooter
5	5	Platformer	platformer
6	6	Sandbox	sandbox
7	7	Survival	survival
8	8	Battle Royale	battle-royale
NULL	NULL	NULL	NULL

```sql
SELECT * FROM users;
```



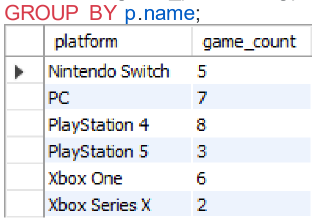
id	username	email	password_hash	display_name	avatar_url	created_at	updated_at
1	gamer1	gamer1@example.com	hashed_password	Gamer One	NULL	2025-05-26 12:10:09	2025-05-26 12:10:09
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

```sql
SELECT g.name AS game_name, p.name AS publisher
FROM games g
JOIN game_publishers gp ON g.id = gp.game_id
JOIN publishers p ON gp.publisher_id = p.id;
```



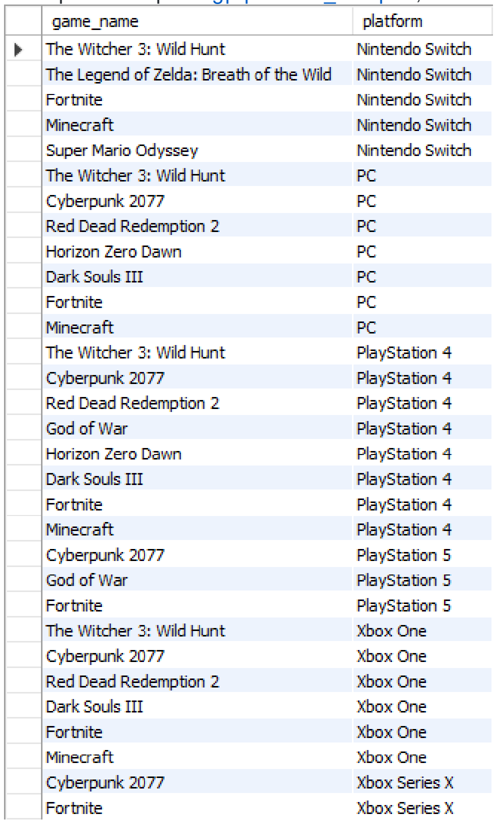
game_name	publisher
Dark Souls III	Bandai Namco Entertainment
The Witcher 3: Wild Hunt	CD Projekt
Cyberpunk 2077	CD Projekt
Fortnite	Epic Games
Minecraft	Microsoft Studios
The Legend of Zelda: Breath of the Wild	Nintendo
Super Mario Odyssey	Nintendo
Red Dead Redemption 2	Rockstar Games
God of War	Sony Interactive Entertainment
Horizon Zero Dawn	Sony Interactive Entertainment

```sql
SELECT p.name AS platform, COUNT(gp.game_id) AS game_count
FROM platforms p
LEFT JOIN game_platforms gp ON p.id = gp.platform_id
GROUP BY p.name;
```



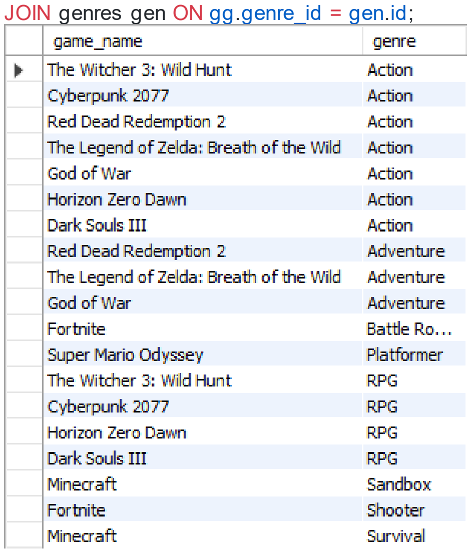
platform	game_count
Nintendo Switch	5
PC	7
PlayStation 4	8
PlayStation 5	3
Xbox One	6
Xbox Series X	2

```sql
SELECT g.name AS game_name, p.name AS platform
FROM games g
JOIN game_platforms gp ON g.id = gp.game_id
JOIN platforms p ON gp.platform_id = p.id;
```



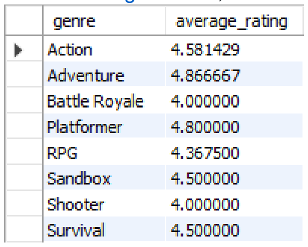
game_name	platform
The Witcher 3: Wild Hunt	Nintendo Switch
The Legend of Zelda: Breath of the Wild	Nintendo Switch
Fortnite	Nintendo Switch
Minecraft	Nintendo Switch
Super Mario Odyssey	Nintendo Switch
The Witcher 3: Wild Hunt	PC
Cyberpunk 2077	PC
Red Dead Redemption 2	PC
Horizon Zero Dawn	PC
Dark Souls III	PC
Fortnite	PC
Minecraft	PC
The Witcher 3: Wild Hunt	PlayStation 4
Cyberpunk 2077	PlayStation 4
Red Dead Redemption 2	PlayStation 4
God of War	PlayStation 4
Horizon Zero Dawn	PlayStation 4
Dark Souls III	PlayStation 4
Fortnite	PlayStation 4
Minecraft	PlayStation 4
Cyberpunk 2077	PlayStation 5
God of War	PlayStation 5
Fortnite	PlayStation 5
The Witcher 3: Wild Hunt	Xbox One
Cyberpunk 2077	Xbox One
Red Dead Redemption 2	Xbox One
Dark Souls III	Xbox One
Fortnite	Xbox One
Minecraft	Xbox One
Cyberpunk 2077	Xbox Series X
Fortnite	Xbox Series X

```sql
SELECT g.name AS game_name, gen.name AS genre
FROM games g
JOIN game_genres gg ON g.id = gg.game_id
JOIN genres gen ON gg.genre_id = gen.id;
```



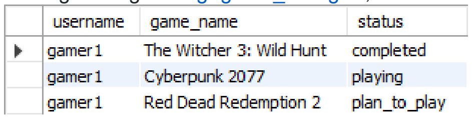
game_name	genre
The Witcher 3: Wild Hunt	Action
Cyberpunk 2077	Action
Red Dead Redemption 2	Action
The Legend of Zelda: Breath of the Wild	Action
God of War	Action
Horizon Zero Dawn	Action
Dark Souls III	Action
Red Dead Redemption 2	Adventure
The Legend of Zelda: Breath of the Wild	Adventure
God of War	Adventure
Fortnite	Battle Ro...
Super Mario Odyssey	Platformer
The Witcher 3: Wild Hunt	RPG
Cyberpunk 2077	RPG
Horizon Zero Dawn	RPG
Dark Souls III	RPG
Minecraft	Sandbox
Fortnite	Shooter
Minecraft	Survival

```sql
SELECT gen.name AS genre, AVG(g.rating) AS average_rating
FROM genres gen
JOIN game_genres gg ON gen.id = gg.genre_id
JOIN games g ON gg.game_id = g.id
GROUP BY gen.name;
```



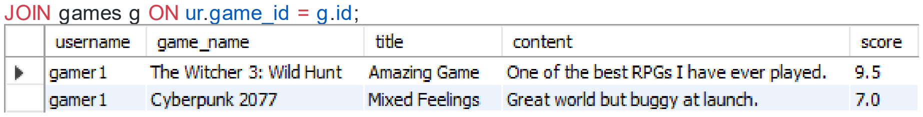
genre	average_rating
Action	4.581429
Adventure	4.866667
Battle Royale	4.000000
Platformer	4.800000
RPG	4.367500
Sandbox	4.500000
Shooter	4.000000
Survival	4.500000

```sql
SELECT u.username, g.name AS game_name, ugl.status
FROM users u
JOIN user_game_lists ugl ON u.id = ugl.user_id
JOIN games g ON ugl.game_id = g.id;
```



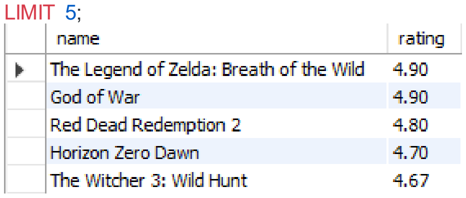
username	game_name	status
gamer1	The Witcher 3: Wild Hunt	completed
gamer1	Cyberpunk 2077	playing
gamer1	Red Dead Redemption 2	plan_to_play

```sql
SELECT u.username, g.name AS game_name, ur.title, ur.content, ur.score
FROM users u
JOIN user_reviews ur ON u.id = ur.user_id
JOIN games g ON ur.game_id = g.id;
```



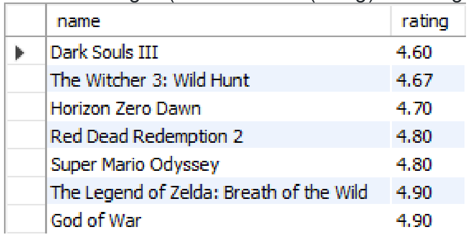
username	game_name	title	content	score
gamer1	The Witcher 3: Wild Hunt	Amazing Game	One of the best RPGs I have ever played.	9.5
gamer1	Cyberpunk 2077	Mixed Feelings	Great world but buggy at launch.	7.0

```sql
SELECT g.name, g.rating
FROM games g
JOIN game_genres gg ON g.id = gg.game_id
JOIN genres gen ON gg.genre_id = gen.id
WHERE gen.name = 'Action'
ORDER BY g.rating DESC
LIMIT 5;
```



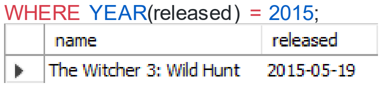
name	rating
The Legend of Zelda: Breath of the Wild	4.90
God of War	4.90
Red Dead Redemption 2	4.80
Horizon Zero Dawn	4.70
The Witcher 3: Wild Hunt	4.67

```sql
SELECT name, rating
FROM games
WHERE rating > (SELECT AVG(rating) FROM games);
```



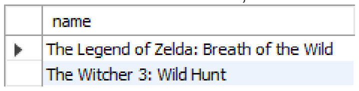
name	rating
Dark Souls III	4.60
The Witcher 3: Wild Hunt	4.67
Horizon Zero Dawn	4.70
Red Dead Redemption 2	4.80
Super Mario Odyssey	4.80
The Legend of Zelda: Breath of the Wild	4.90
God of War	4.90

```sql
SELECT name, released
FROM games
WHERE YEAR(released) = 2015;
```



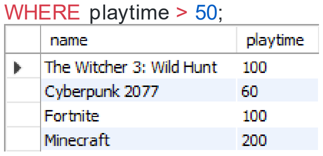
name	released
The Witcher 3: Wild Hunt	2015-05-19

```
SELECT name
FROM games
WHERE name LIKE 'T%';
```



name
The Legend of Zelda: Breath of the Wild
The Witcher 3: Wild Hunt

```
SELECT name, playtime
FROM games
WHERE playtime > 50;
```



name	playtime
The Witcher 3: Wild Hunt	100
Cyberpunk 2077	60
Fortnite	100
Minecraft	200

7.3 Performance Testing

- Tested query response times with sample data
- Verified foreign key constraints work correctly
- Confirmed unique constraints prevent duplicate data
- Validated data types handle expected ranges

8. Security Considerations

8.1 Data Protection

- Password hashing (never store plain text passwords)
- User input validation to prevent SQL injection
- Foreign key constraints to maintain data integrity
- Unique constraints to prevent duplicate accounts

8.2 Privacy Controls

- Users can make their lists private
- Personal ratings and progress are user-specific
- Review attribution to maintain accountability

9. Future Enhancements

9.1 Planned Features

- Achievement tracking system
- Game recommendation algorithms
- Wishlist functionality
- Enhanced social features (messaging, groups)
- Content moderation tools
- Mobile app support

9.2 Scalability Considerations

- Database can handle millions of users and games
- Efficient indexing for fast searches
- Normalized design reduces storage requirements
- Foreign key relationships maintain data consistency

10. Conclusion

This database design successfully creates a comprehensive system for tracking and managing video game collections. The normalized design ensures data integrity while providing flexibility for future enhancements. The schema supports all required features including game tracking, reviews, social interactions, and custom lists.

Key Achievements:

- Fully normalized database design (3NF)
- Comprehensive game metadata storage
- Flexible user interaction system
- Scalable architecture for future growth
- Proper security and privacy controls

The MyGameList database provides a solid foundation for a modern game tracking platform that can serve thousands of users while maintaining performance and data integrity.

References

- RAWG Video Games Database
- MyAnimeList
- IGDB (Internet Game Database)
- Steam Web API

Note: This project demonstrates practical application of database design principles learned in class, including entity-relationship modeling, normalization theory, and SQL implementation.