-- Create Genres table

CREATE TABLE Genres (

genreID INT PRIMARY KEY,

genreName VARCHAR(50) NOT NULL

);

-- Create Platforms table

CREATE TABLE Platforms (

platformID INT PRIMARY KEY,

platformName VARCHAR(50) NOT NULL

);

-- Create Publishers table

CREATE TABLE Publishers (

publisherID INT PRIMARY KEY,

publisherName VARCHAR(100) NOT NULL

);

-- Create Developers table

CREATE TABLE Developers (

developerID INT PRIMARY KEY,

developerName VARCHAR(100) NOT NULL

);

-- Create Games table with foreign key references

CREATE TABLE Games (

gameID INT PRIMARY KEY,

title VARCHAR(100) NOT NULL,

releaseYear INT,

genreID INT,

platformID INT,

publisherID INT,

developerID INT,

FOREIGN KEY (genreID) REFERENCES Genres(genreID),

FOREIGN KEY (platformID) REFERENCES Platforms(platformID),

FOREIGN KEY (publisherID) REFERENCES Publishers(publisherID),

FOREIGN KEY (developerID) REFERENCES Developers(developerID)

);

-- Insert data into Genres table

INSERT INTO Genres (genreID, genreName) VALUES

(1, 'Action'),

(2, 'Adventure'),

(3, 'Role-Playing'),

(4, 'Sports'),

(5, 'Simulation');

-- Insert data into Platforms table

INSERT INTO Platforms (platformID, platformName) VALUES

(1, 'PC'),

(2, 'PlayStation'),

(3, 'Xbox'),

(4, 'Nintendo Switch'),

(5, 'Mobile');

-- Insert data into Publishers table

INSERT INTO Publishers (publisherID, publisherName) VALUES

(1, 'Electronic Arts'),

(2, 'Ubisoft'),

(3, 'Activision'),

(4, 'Nintendo'),

(5, 'Square Enix');

-- Insert data into Developers table

INSERT INTO Developers (developerID, developerName) VALUES

(1, 'Rockstar Games'),

(2, 'Naughty Dog'),

(3, 'CD Projekt'),

(4, 'Valve'),

(5, 'Epic Games');

-- Insert data into Games table

INSERT INTO Games (gameID, title, releaseYear, genreID, platformID, publisherID, developerID) VALUES

(101, 'Game1', 2020, 1, 1, 1, 1),

(102, 'Game2', 2018, 2, 2, 2, 2),

(103, 'Game3', 2019, 3, 3, 3, 3),

(104, 'Game4', 2021, 4, 4, 4, 4),

(105, 'Game5', 2022, 5, 5, 5, 5);

Example Queries:

Retrieve all games and their genres:

SELECT Games.title, Genres.genreName

FROM Games

JOIN Genres ON Games.genreID = Genres.genreID;

List all platforms and the number of games available for each:

SELECT Platforms.platformName, COUNT(\*) AS NumberOfGames

FROM Games

JOIN Platforms ON Games.platformID = Platforms.platformID

GROUP BY Platforms.platformName;

Find games released in 2020 or later:

SELECT title, releaseYear

FROM Games

WHERE releaseYear >= 2020;

Show the names of all publishers and the number of games they have published:

SELECT Publishers.publisherName, COUNT(\*) AS NumberOfGamesPublished

FROM Games

JOIN Publishers ON Games.publisherID = Publishers.publisherID

GROUP BY Publishers.publisherName;

Retrieve games developed by a specific developer (e.g., Naughty Dog):

SELECT title, releaseYear

FROM Games

JOIN Developers ON Games.developerID = Developers.developerID

WHERE Developers.developerName = 'Naughty Dog';

**List all genres and the average release year of games in each genre:**

SELECT Genres.genreName, AVG(Games.releaseYear) AS AverageReleaseYear

FROM Games

JOIN Genres ON Games.genreID = Genres.genreID

GROUP BY Genres.genreName;

**Find the game with the highest release year:**

SELECT title, releaseYear

FROM Games

ORDER BY releaseYear DESC

LIMIT 1;

Show the names of all developers who have developed more than one game:

SELECT Developers.developerName, COUNT(\*) AS NumberOfGamesDeveloped

FROM Games

JOIN Developers ON Games.developerID = Developers.developerID

GROUP BY Developers.developerName

HAVING NumberOfGamesDeveloped > 1;

**Retrieve games of a specific genre (e.g., Role-Playing) released before 2020:**

SELECT title, releaseYear

FROM Games

JOIN Genres ON Games.genreID = Genres.genreID

WHERE Genres.genreName = 'Role-Playing' AND releaseYear < 2020;

**List all games along with their genres and platforms:**

SELECT Games.title, Genres.genreName, Platforms.platformName

FROM Games

JOIN Genres ON Games.genreID = Genres.genreID

JOIN Platforms ON Games.platformID = Platforms.platformID;

**Show the total number of games in the database:**

SELECT COUNT(\*) AS TotalGames

FROM Games;

**Retrieve games published by a specific publisher (e.g., Ubisoft) on a specific platform (e.g., PC):**

SELECT title, releaseYear

FROM Games

JOIN Publishers ON Games.publisherID = Publishers.publisherID

JOIN Platforms ON Games.platformID = Platforms.platformID

WHERE Publishers.publisherName = 'Ubisoft' AND Platforms.platformName = 'Playstation';

**Find the oldest game in the database:**

SELECT title, releaseYear

FROM Games

ORDER BY releaseYear

LIMIT 1;

**List all genres and the number of games in each genre:**

SELECT Genres.genreName, COUNT(\*) AS NumberOfGames

FROM Games

JOIN Genres ON Games.genreID = Genres.genreID

GROUP BY Genres.genreName;

**Retrieve games released between 2015 and 2020 (inclusive):**

SELECT title, releaseYear

FROM Games

WHERE releaseYear BETWEEN 2015 AND 2020;

Show the average release year of all games in the database:

SELECT AVG(releaseYear) AS AverageReleaseYear

FROM Games;

Retrieve games with a specific title (e.g., 'Game3'):

SELECT \*

FROM Games

WHERE title = 'Game3';

List all developers and the number of games they have developed, ordered by the number of games in descending order:

SELECT Developers.developerName, COUNT(\*) AS NumberOfGamesDeveloped

FROM Games

JOIN Developers ON Games.developerID = Developers.developerID

GROUP BY Developers.developerName

ORDER BY NumberOfGamesDeveloped DESC;

Show the names of all platforms where at least one game has been released:

SELECT DISTINCT Platforms.platformName

FROM Games

JOIN Platforms ON Games.platformID = Platforms.platformID;

Retrieve games that have not been assigned a genre:

SELECT title, releaseYear

FROM Games

WHERE genreID IS NULL;

These queries cover a variety of scenarios and can serve as a starting point for exploring and querying the video game database. Adjust them based on your specific needs.

Find the average release year of games for each combination of genre and platform:

SELECT Genres.genreName, Platforms.platformName, AVG(Games.releaseYear) AS AverageReleaseYear

FROM Games

JOIN Genres ON Games.genreID = Genres.genreID

JOIN Platforms ON Games.platformID = Platforms.platformID

GROUP BY Genres.genreName, Platforms.platformName;

Retrieve games with the highest release year in each genre:

SELECT Genres.genreName, Games.title, Games.releaseYear

FROM Games

JOIN Genres ON Games.genreID = Genres.genreID

WHERE (Genres.genreID, releaseYear) IN (

SELECT genreID, MAX(releaseYear) AS MaxReleaseYear

FROM Games

GROUP BY genreID

);

List the top 5 developers with the highest average release year of their games:

SELECT Developers.developerName, AVG(Games.releaseYear) AS AverageReleaseYear

FROM Games

JOIN Developers ON Games.developerID = Developers.developerID

GROUP BY Developers.developerName

ORDER BY AverageReleaseYear DESC

LIMIT 5;

Retrieve games that share the same genre and platform with at least one other game:

SELECT g1.title AS Game1, g2.title AS Game2, Genres.genreName, Platforms.platformName

FROM Games g1

JOIN Games g2 ON g1.genreID = g2.genreID AND g1.platformID = g2.platformID AND g1.gameID <> g2.gameID

JOIN Genres ON g1.genreID = Genres.genreID

JOIN Platforms ON g1.platformID = Platforms.platformID;

Retrieve games with titles containing the same word as the developer's name:

SELECT Games.title, Developers.developerName

FROM Games

JOIN Developers ON Games.developerID = Developers.developerID

WHERE Games.title LIKE CONCAT('%', Developers.developerName, '%');

Retrieve games that have been released on all available platforms:

SELECT title, GROUP\_CONCAT(DISTINCT Platforms.platformName ORDER BY Platforms.platformName) AS AvailablePlatforms

FROM Games

JOIN Platforms ON Games.platformID = Platforms.platformID

GROUP BY title

HAVING COUNT(DISTINCT Platforms.platformID) = (SELECT COUNT(\*) FROM Platforms);

Show the number of games released each year, along with the cumulative number of games up to that year:

SELECT releaseYear, COUNT(\*) AS NumberOfGames,

SUM(COUNT(\*)) OVER (ORDER BY releaseYear) AS CumulativeGames

FROM Games

GROUP BY releaseYear

ORDER BY releaseYear;