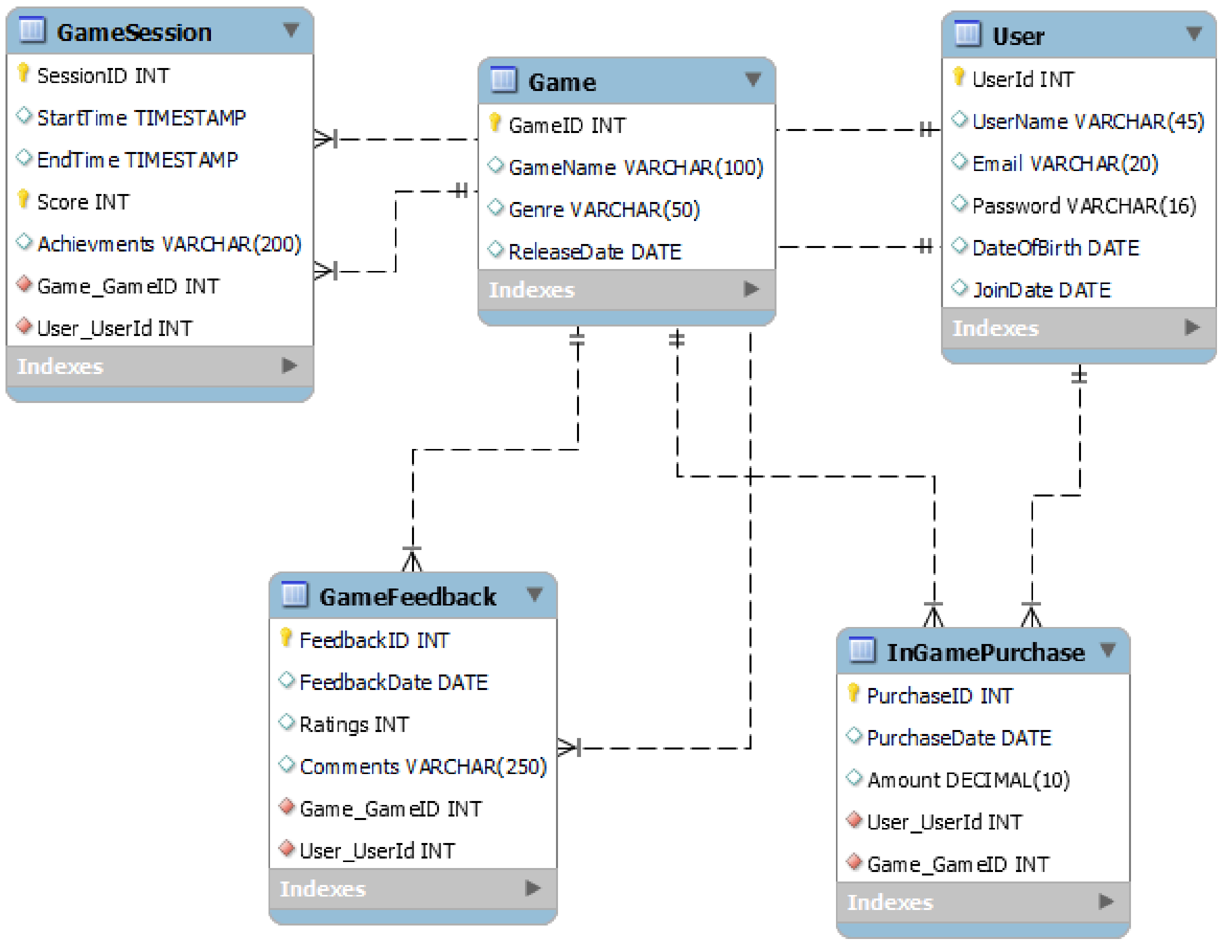
Virtual Reality Gaming Analytics

Introduction:

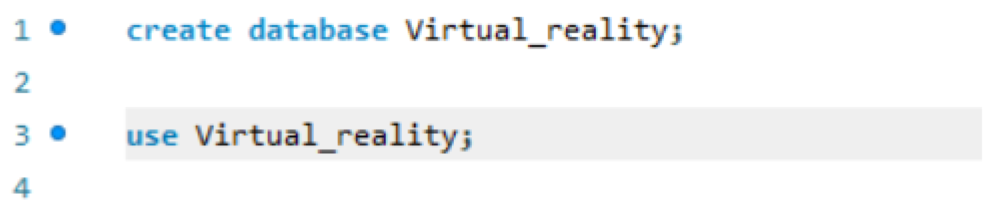
I have done a case study on Virtual Reality Gaming. Since Virtual reality gaming users have increased, I have created a dataset where we can see the amount spent on games, time spent on game, how many users have been playing etc.

Simply, we can find how people are more active in virtual reality than in reality.

Entity relationship diagram:



Database:


```sql
create database Virtual_reality;

use Virtual_reality;
```

- To create a database, I have used “Create database” keyword
- With the help of the “use” keyword I have used the database.

User Table

```sql
CREATE TABLE User (
   UserId INT PRIMARY KEY NOT NULL,
   Username VARCHAR(50),
   Email VARCHAR(50),
   Password VARCHAR(50),
   DateOfBirth DATE,
   JoinDate DATE
);

INSERT INTO User (UserId, Username, Email, Password, DateOfBirth,
JoinDate) VALUES
(1, 'john_doe', 'john.doe@example.com', 'password123', '1990-01-01', '2020-01-01'),
(2, 'jane_smith', 'jane.smith@example.com', 'password456', '1991-02-15',
'2020-02-15'),
(3, 'michael_brown', 'michael.brown@example.com', 'password789',
'1992-03-20', '2020-03-20'),
(4, 'emily_davis', 'emily.davis@example.com', 'password101', '1993-04-25',
'2020-04-25'),
(5, 'daniel_miller', 'daniel.miller@example.com', 'password102', '1994-05-30',
'2020-05-30'),
(6, 'sarah_wilson', 'sarah.wilson@example.com', 'password103', '1995-06-10',
'2020-06-10'),
(7, 'david_moore', 'david.moore@example.com', 'password104', '1996-07-15',
'2020-07-15'),
(8, 'laura_taylor', 'laura.taylor@example.com', 'password105', '1997-08-20',
'2020-08-20'),
(9, 'james_anderson', 'james.anderson@example.com', 'password106',
'1998-09-25', '2020-09-25'),
(10, 'olivia_jackson', 'olivia.jackson@example.com', 'password107', '1999-10-30',
'2020-10-30'),
(11, 'benjamin_thomas', 'benjamin.thomas@example.com', 'password108',
'1988-11-15', '2020-11-15'),
(12, 'isabella_lee', 'isabella.lee@example.com', 'password109', '1989-12-20',
'2020-12-20'),
(13, 'lucas_white', 'lucas.white@example.com', 'password110', '1990-01-10',
'2021-01-10'),
(14, 'mia_harris', 'mia.harris@example.com', 'password111', '1991-02-15',
'2021-02-15'),
(15, 'alexander_martin', 'alexander.martin@example.com', 'password112',
'1992-03-20', '2021-03-20'),
(16, 'ava_thompson', 'ava.thompson@example.com', 'password113', '1993-04-25',
'2021-04-25'),
(17, 'ethan_garcia', 'ethan.garcia@example.com', 'password114', '1994-05-30',
'2021-05-30'),
(18, 'sophia_martinez', 'sophia.martinez@example.com', 'password115',
'1995-06-10', '2021-06-10'),
(19, 'jackson_rodriguez', 'jackson.rodriguez@example.com', 'password116',
'1996-07-15', '2021-07-15'),
(20, 'amelia_clark', 'amelia.clark@example.com', 'password117', '1997-08-20',
'2021-08-20'),
(21, 'logan_hernandez', 'logan.hernandez@example.com', 'password118',
'1998-09-25', '2021-09-25'),
(22, 'ella_lewis', 'ella.lewis@example.com', 'password119', '1999-10-30',
'2021-10-30'),
(23, 'mason_hall', 'mason.hall@example.com', 'password120', '1988-11-15',
'2021-11-15'),
(24, 'sophie_allen', 'sophie.allen@example.com', 'password121', '1989-12-20',
'2021-12-20'),
(25, 'liam_walker', 'liam.walker@example.com', 'password122', '1990-01-10',
'2022-01-10'),
(26, 'charlotte_scott', 'charlotte.scott@example.com', 'password123', '1991-02-15',
'2022-02-15'),
(27, 'noah_adams', 'noah.adams@example.com', 'password124', '1992-03-20',
'2022-03-20'),
(28, 'grace_mitchell', 'grace.mitchell@example.com', 'password125', '1993-04-25',
'2022-04-25'),
(29, 'oliver_perez', 'oliver.perez@example.com', 'password126', '1994-05-30',
'2022-05-30'),
(30, 'chloe_turner', 'chloe.turner@example.com', 'password127', '1995-06-10',
'2022-06-10');
```

Game Table

```
CREATE TABLE Game (
   GameID INT PRIMARY KEY NOT NULL,
   GameName VARCHAR(100),
   Genre VARCHAR(50),
   ReleaseDate DATE,
   DeveloperID INT
);

INSERT INTO Game (GameID, GameName, Genre, ReleaseDate, DeveloperID)
VALUES
(1, 'VR Adventure Quest', 'Adventure', '2020-01-15', 1),
(2, 'Space Explorer', 'Sci-Fi', '2020-02-20', 2),
(3, 'Zombie Apocalypse', 'Horror', '2020-03-25', 3),
(4, 'Fantasy Kingdom', 'Fantasy', '2020-04-30', 4),
(5, 'Superhero Battle', 'Action', '2020-05-10', 5),
(6, 'Mystery Mansion', 'Mystery', '2020-06-15', 6),
(7, 'Racing Thrills', 'Racing', '2020-07-20', 7),
(8, 'Underwater World', 'Adventure', '2020-08-25', 8),
(9, 'Alien Invasion', 'Sci-Fi', '2020-09-30', 9),
(10, 'Haunted House', 'Horror', '2020-10-10', 10),
(11, 'Medieval Warrior', 'Fantasy', '2020-11-15', 1),
(12, 'Ninja Legends', 'Action', '2020-12-20', 2),
(13, 'Pirate Treasure Hunt', 'Adventure', '2021-01-25', 3),
(14, 'Dinosaur Safari', 'Adventure', '2021-02-28', 4),
(15, 'Futuristic City', 'Sci-Fi', '2021-03-15', 5),
(16, 'Ghost Town', 'Horror', '2021-04-20', 6),
(17, 'Dragon Slayer', 'Fantasy', '2021-05-25', 7),
(18, 'War Zone', 'Action', '2021-06-30', 8),
(19, 'Jungle Expedition', 'Adventure', '2021-07-10', 9),
(20, 'Galactic War', 'Sci-Fi', '2021-08-15', 10),
(21, 'Witch Hunt', 'Horror', '2021-09-20', 1),
(22, 'Elf Kingdom', 'Fantasy', '2021-10-25', 2),
(23, 'Martial Arts Master', 'Action', '2021-11-30', 3),
(24, 'Treasure Island', 'Adventure', '2021-12-10', 4),
(25, 'Robot Rebellion', 'Sci-Fi', '2022-01-15', 5),
(26, 'Creepy Castle', 'Horror', '2022-02-20', 6),
(27, 'Mage Wars', 'Fantasy', '2022-03-25', 7),
(28, 'Battle Royale', 'Action', '2022-04-30', 8),
(29, 'Safari Adventure', 'Adventure', '2022-05-10', 9),
(30, 'Space Odyssey', 'Sci-Fi', '2022-06-15', 10);
```

Game Session Table

```
CREATE TABLE GameSession (
   SessionID INT PRIMARY KEY,
   StartTime DATE,
   EndTime DATE,
   Score INT,
   Achievements VARCHAR(100),
   UserId INT,
   GameID INT,
   FOREIGN KEY (UserId) REFERENCES User(UserId),
   FOREIGN KEY (GameID) REFERENCES Game(GameID)
);

INSERT INTO GameSession (SessionID, StartTime, EndTime, Score,
Achievements, UserId, GameID) VALUES
(101, '2023-01-01 14:00:00', '2023-01-01 15:00:00', 500, 'First Blood', 1, 1),
(102, '2023-01-02 16:00:00', '2023-01-02 17:00:00', 600, 'Sharp Shooter', 2, 2),
(103, '2023-01-03 18:00:00', '2023-01-03 19:00:00', 700, 'Zombie Slayer', 3, 3),
(104, '2023-01-04 20:00:00', '2023-01-04 21:00:00', 800, 'Dragon Tamer', 4, 4),
(105, '2023-01-05 14:00:00', '2023-01-05 15:00:00', 900, 'Heroic Savior', 5, 5),
(106, '2023-01-06 16:00:00', '2023-01-06 17:00:00', 1000, 'Ghost Hunter', 6, 6),
(107, '2023-01-07 18:00:00', '2023-01-07 19:00:00', 1100, 'Race Champion', 7, 7),
(108, '2023-01-08 20:00:00', '2023-01-08 21:00:00', 1200, 'Underwater Explorer', 8,
8),
(109, '2023-01-09 14:00:00', '2023-01-09 15:00:00', 1300, 'Alien Defeater', 9, 9),
(110, '2023-01-10 16:00:00', '2023-01-10 17:00:00', 1400, 'Haunted House Master',
10, 10),
(111, '2023-01-11 18:00:00', '2023-01-11 19:00:00', 1500, 'Medieval Conqueror', 1, 11),
(112, '2023-01-12 20:00:00', '2023-01-12 21:00:00', 1600, 'Ninja Warrior', 2, 12),
(113, '2023-01-13 14:00:00', '2023-01-13 15:00:00', 1700, 'Pirate King', 3, 13),
(114, '2023-01-14 16:00:00', '2023-01-14 17:00:00', 1800, 'Dinosaur Tracker', 4, 14),
(115, '2023-01-15 18:00:00', '2023-01-15 19:00:00', 1900, 'City Protector', 5, 15),
(116, '2023-01-16 20:00:00', '2023-01-16 21:00:00', 2000, 'Ghostbuster', 6, 16),
(117, '2023-01-17 14:00:00', '2023-01-17 15:00:00', 2100, 'Dragon Slayer', 7, 17),
(118, '2023-01-18 16:00:00', '2023-01-18 17:00:00', 2200, 'War Hero', 8, 18),
(119, '2023-01-19 18:00:00', '2023-01-19 19:00:00', 2300, 'Jungle Explorer', 9, 19),
(120, '2023-01-20 20:00:00', '2023-01-20 21:00:00', 2400, 'Galactic Commander',
10, 20),
(121, '2023-01-21 14:00:00', '2023-01-21 15:00:00', 2500, 'Witch Hunter', 1, 21),
(122, '2023-01-22 16:00:00', '2023-01-22 17:00:00', 2600, 'Elf Protector', 2, 22),
(123, '2023-01-23 18:00:00', '2023-01-23 19:00:00', 2700, 'Martial Arts Master', 3,
23),
(124, '2023-01-24 20:00:00', '2023-01-24 21:00:00', 2800, 'Treasure Hunter', 4, 24),
(125, '2023-01-25 14:00:00', '2023-01-25 15:00:00', 2900, 'Robot Destroyer', 5, 25),
(126, '2023-01-26 16:00:00', '2023-01-26 17:00:00', 3000, 'Castle Conqueror', 6, 26),
(127, '2023-01-27 18:00:00', '2023-01-27 19:00:00', 3100, 'Mage Master', 7, 27),
(128, '2023-01-28 20:00:00', '2023-01-28 21:00:00', 3200, 'Battle Royale
Champion', 8, 28),
(129, '2023-01-29 14:00:00', '2023-01-29 15:00:00', 3300, 'Safari Explorer', 9, 29),
(130, '2023-01-30 16:00:00', '2023-01-30 17:00:00', 3400, 'Space Adventurer', 10,
30);
```

In Game Purchase Table

```
CREATE TABLE InGamePurchase (
   PurchaseID INT PRIMARY KEY,
   PurchaseDate DATE,
   Amount DECIMAL(10,2),
   UserId INT,
    GameID INT,
    FOREIGN KEY (UserId) REFERENCES User(UserId),
    FOREIGN KEY (GameID) REFERENCES Game(GameID)
);

INSERT INTO InGamePurchase (PurchaseID, PurchaseDate, Amount, UserId,
GameID) VALUES
(1, '2024-01-01', 19.99, 1, 1),
(2, '2024-01-02', 29.99, 2, 2),
(3, '2024-01-03', 39.99, 3, 3),
(4, '2024-01-04', 49.99, 4, 4),
(5, '2024-01-05', 59.99, 5, 5),
(6, '2024-01-06', 69.99, 6, 6),
(7, '2024-01-07', 79.99, 7, 7),
(8, '2024-01-08', 89.99, 8, 8),
(9, '2024-01-09', 99.99, 9, 9),
(10, '2024-01-10', 109.99, 10, 10),
(11, '2024-01-11', 119.99, 1, 2),
(12, '2024-01-12', 129.99, 2, 3),
(13, '2024-01-13', 139.99, 3, 4),
(14, '2024-01-14', 149.99, 4, 5),
(15, '2024-01-15', 159.99, 5, 6),
(16, '2024-01-16', 169.99, 6, 7),
(17, '2024-01-17', 179.99, 7, 8),
(18, '2024-01-18', 189.99, 8, 9),
(19, '2024-01-19', 199.99, 9, 10),
(20, '2024-01-20', 209.99, 10, 1),
(21, '2024-01-21', 219.99, 1, 3),
(22, '2024-01-22', 229.99, 2, 4),
(23, '2024-01-23', 239.99, 3, 5),
(24, '2024-01-24', 249.99, 4, 6),
(25, '2024-01-25', 259.99, 5, 7),
(26, '2024-01-26', 269.99, 6, 8),
(27, '2024-01-27', 279.99, 7, 9),
(28, '2024-01-28', 289.99, 8, 10),
(29, '2024-01-29', 299.99, 9, 1),
(30, '2024-01-30', 309.99, 10, 2);
```

Game Feedback Table

```
CREATE TABLE GameFeedback (
   FeedbackID INT PRIMARY KEY NOT NULL,
   FeedbackDate DATE,
   Rating INT,
   Comments VARCHAR(255),
   UserId INT,
   GameID INT,
   FOREIGN KEY (UserId) REFERENCES User(UserId),
   FOREIGN KEY (GameID) REFERENCES Game(GameID)
);

INSERT INTO GameFeedback (FeedbackID, FeedbackDate, Rating,
Comments, UserId, GameID) VALUES
(1, '2024-01-01', 5, 'Amazing game! Loved the graphics and gameplay.', 1, 1),
(2, '2024-01-02', 4, 'Great game, but could use more levels.', 2, 2),
(3, '2024-01-03', 3, 'It’s okay, but not my favorite.', 3, 3),
(4, '2024-01-04', 2, 'Had some bugs that need fixing.', 4, 4),
(5, '2024-01-05', 1, 'Very disappointing. Not worth the price.', 5, 5),
(6, '2024-01-06', 5, 'One of the best games I have played this year!', 6, 6),
(7, '2024-01-07', 4, 'Enjoyable but repetitive after a while.', 7, 7),
(8, '2024-01-08', 3, 'The game is good, but the controls are frustrating.', 8, 8),
(9, '2024-01-09', 2, 'Not as good as I expected.', 9, 9),
(10, '2024-01-10', 1, 'Didn’t like it at all.', 10, 10),
(11, '2024-01-11', 5, 'Fantastic experience! Highly recommend.', 1, 2),
(12, '2024-01-12', 4, 'Very fun, though it needs more customization options.', 2,
3),
(13, '2024-01-13', 3, 'Decent game, but there are better options.', 3, 4),
(14, '2024-01-14', 2, 'Not impressed with the storyline.', 4, 5),
(15, '2024-01-15', 1, 'Worst game ever. Waste of money.', 5, 6),
(16, '2024-01-16', 5, 'Incredible graphics and sound. Loved every moment.', 6, 7),
(17, '2024-01-17', 4, 'Good game but needs more frequent updates.', 7, 8),
(18, '2024-01-18', 3, 'Average game, not too special.', 8, 9),
(19, '2024-01-19', 2, 'Too many bugs, didn’t enjoy playing it.', 9, 10),
(20, '2024-01-20', 1, 'Terrible experience. Would not recommend.', 10, 1),
(21, '2024-01-21', 5, 'Perfect game, can’t wait for the sequel!', 1, 3),
(22, '2024-01-22', 4, 'Great graphics, but the story was lacking.', 2, 4),
```

(23, '2024-01-23', 3, 'It's fine, but there are better games out there.', 3, 5),
(24, '2024-01-24', 2, 'Disappointing. Not as good as the previous version.', 4, 6),
(25, '2024-01-25', 1, 'The worst game I've played this year.', 5, 7),
(26, '2024-01-26', 5, 'Excellent gameplay and story. Highly recommended!', 6, 8),
(27, '2024-01-27', 4, 'Fun to play but can get boring after a while.', 7, 9),
(28, '2024-01-28', 3, 'The game is okay, but needs improvement.', 8, 10),
(29, '2024-01-29', 2, 'Many issues with performance. Not satisfied.', 9, 1),
(30, '2024-01-30', 1, 'Very disappointing experience. Would not buy again.', 10, 2);

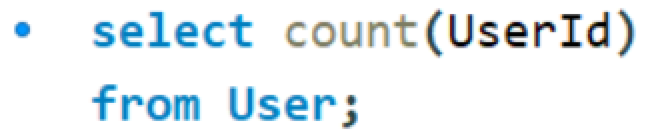
Explanation:

- I have created a set of tables using the **"create table"** command.
- Also I have entered the columns along with their data types.
- The ID's from each table should be unique and not null so I have mentioned the ID as **"Primary key".**
- The **"foreign key"** refers to the column name from the other table.
- I have inserted a 30 rows of values in each tables using **"insert into values();"**

Let's get into Virtual Reality.

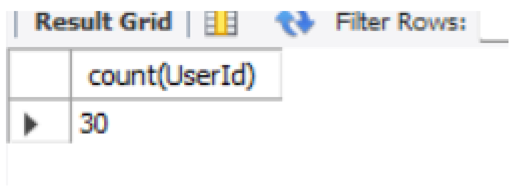
1 . How many users are there in the database?

syntax:



```
select count(UserId)
from User;
```

Output:



count(UserId)
30

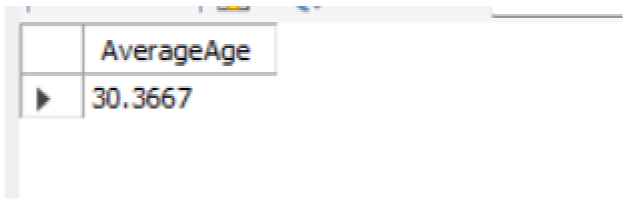
- I have selected the UserId from the table User and with the help of **count()** function, I have calculated the total users.

2 . What is the average age of users?

Syntax:

```
SELECT
AVG(TIMESTAMPDIFF(YEAR, DateOfBirth, CURDATE())) AS AverageAge
FROM User;
```

Output:



```
AverageAge
30.3667
```

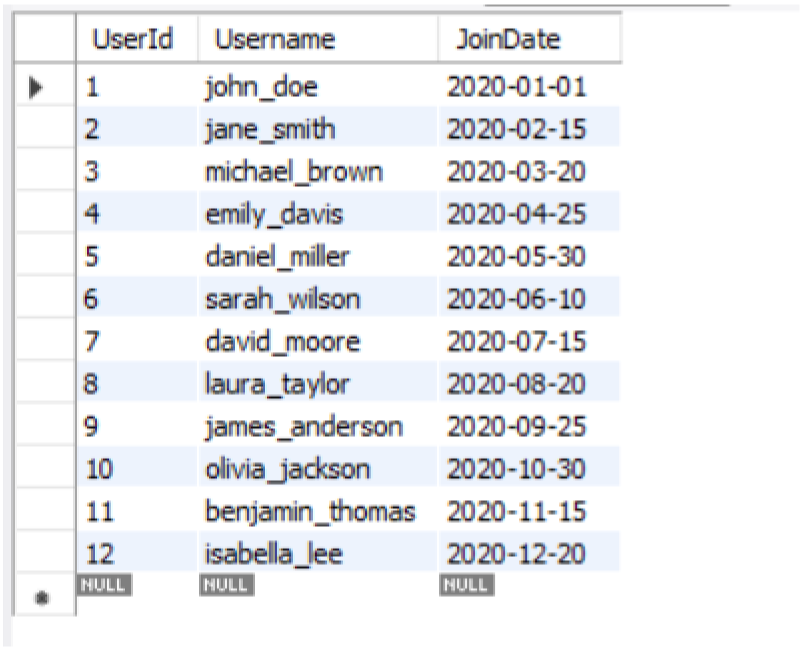
- The **“TIMESTAMPDIFF(YEAR, DateOfBirth, CURDATE())”** function helps to find the difference in years between the D.O.B
- The CURDATE() function helps to find the current date which gives the current age of the user.
- AVG() function helps to calculate the average of the age.

3 . Which users joined the platform in a specific year?

Syntax:

```
select UserId, Username, JoinDate
from User
where year(JoinDate) = 2020;
```

Output:



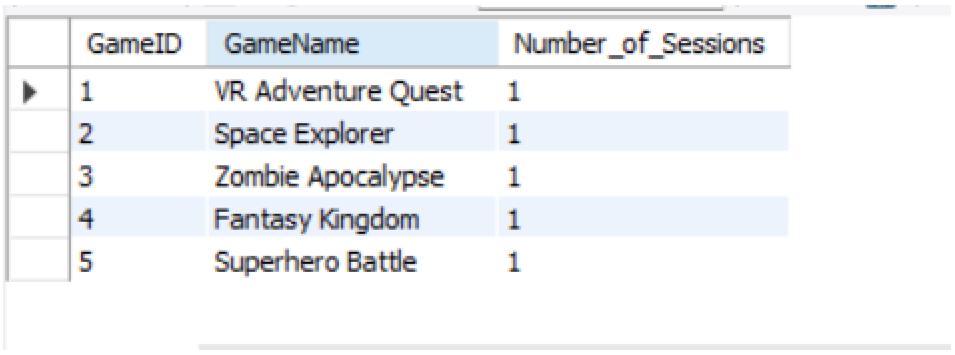
	UserId	Username	JoinDate
▶	1	john_doe	2020-01-01
	2	jane_smith	2020-02-15
	3	michael_brown	2020-03-20
	4	emily_davis	2020-04-25
	5	daniel_miller	2020-05-30
	6	sarah_wilson	2020-06-10
	7	david_moore	2020-07-15
	8	laura_taylor	2020-08-20
	9	james_anderson	2020-09-25
	10	olivia_jackson	2020-10-30
	11	benjamin_thomas	2020-11-15
	12	isabella_lee	2020-12-20
*	NULL	NULL	NULL

- I have selected the UserId, Username and JoinDate using **select** from the table user.
- **Where year(JoinDate) = 2020** condition helps to retrieve the data of the specific year.

4 . What are the top 5 most played games based on the number of game sessions?
Syntax:

```
select Game.GameID, Game.GameName, count(GameSession.SessionID) as Number_of_Sessions
from Game
join GameSession on Game.GameId =  GameSession.GameID
group by Game.GameID, Game.GameName
order by Number_of_Sessions desc
limit 5;
```

Output:



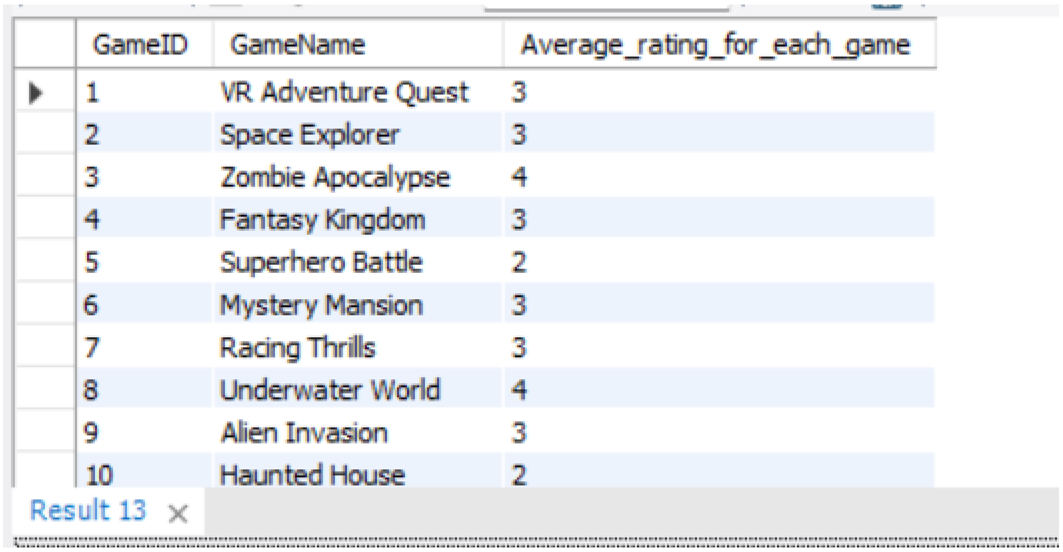
	GameID	GameName	Number_of_Sessions
▶	1	VR Adventure Quest	1
	2	Space Explorer	1
	3	Zombie Apocalypse	1
	4	Fantasy Kingdom	1
	5	Superhero Battle	1

- Here the GameID was the same in Game table and game sessions table.
- So i have joined both the tables using **join** function
- **“Group by ''** used to group the values as GameID and GameName wise.
- **“Order by”** used to order the values in ascending or descending order.
- In default it will order by ascending.
- To get the values in descending we should mention **“desc”** and **“asc”** for ascending.
- **“Limit”** function is used to limit the values.

5 . What is the average rating for each game?
Syntax:

```
select Game.GameID, Game.GameName, round(AVG(GameFeedback.Rating)) as Average_rating_for_each_game
from Game
join GameFeedback on Game.GameID = GameFeedback.GameID
group by Game.GameId, Game.GameName;
```

Output:



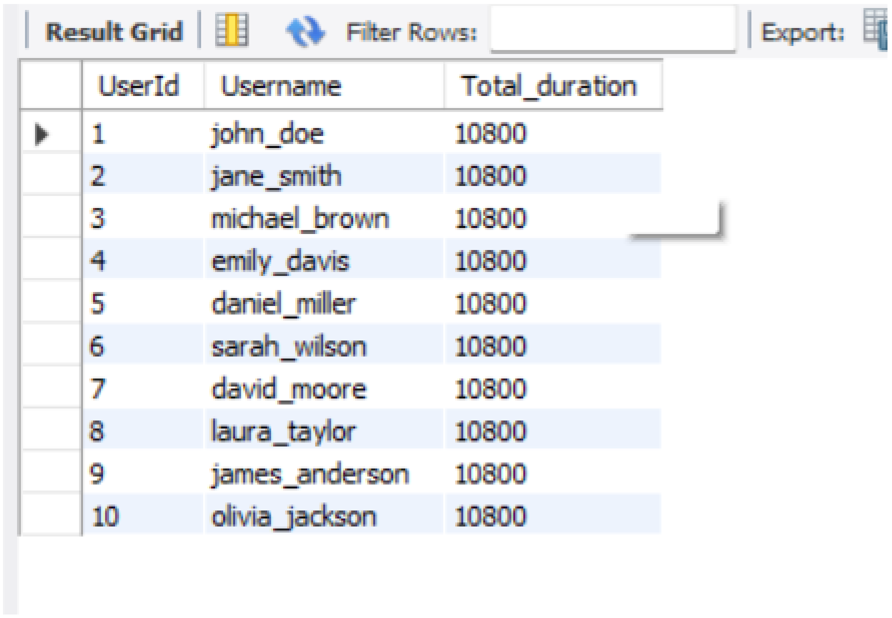
GameID	GameName	Average_rating_for_each_game
1	VR Adventure Quest	3
2	Space Explorer	3
3	Zombie Apocalypse	4
4	Fantasy Kingdom	3
5	Superhero Battle	2
6	Mystery Mansion	3
7	Racing Thrills	3
8	Underwater World	4
9	Alien Invasion	3
10	Haunted House	2

- Joined the 2 tables using **joins** function
- **“Round”** function is used to round the values.
- **“Avg”** to calculate the average.
- Here I have used an alias which is mentioned as **“as”** and the table name I want.
- **“Group by”** used to group the values on GameId and GameName wise.

6 . What is the total duration of game sessions for each user?
Syntax:

```
select User.UserId, User.Username, SUM(TIMESTAMPDIFF(SECOND, GameSession.StartTime, GameSession.EndTime)) as  Total_duration
from User
join GameSession on User.UserId = GameSession.UserId
group by User.UserId, User.Username;
```

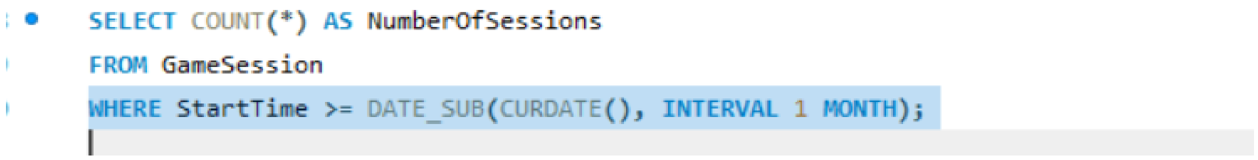
Output:



UserId	Username	Total_duration
1	john_doe	10800
2	jane_smith	10800
3	michael_brown	10800
4	emily_davis	10800
5	daniel_miller	10800
6	sarah_wilson	10800
7	david_moore	10800
8	laura_taylor	10800
9	james_anderson	10800
10	olivia_jackson	10800

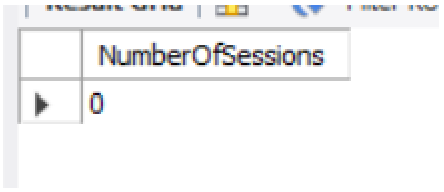
- **Sum()** function is used to calculate the Start time and the end time.
- **TIMESTAMPDIFF()** function is used to calculate the difference between two datetime or date expressions.

7 . How many game sessions were held in the last month?
Syntax:



```
SELECT COUNT(*) AS NumberOfSessions
FROM GameSession
WHERE StartTime >= DATE_SUB(CURDATE(), INTERVAL 1 MONTH);
```

Output:



NumberOfSessions
0

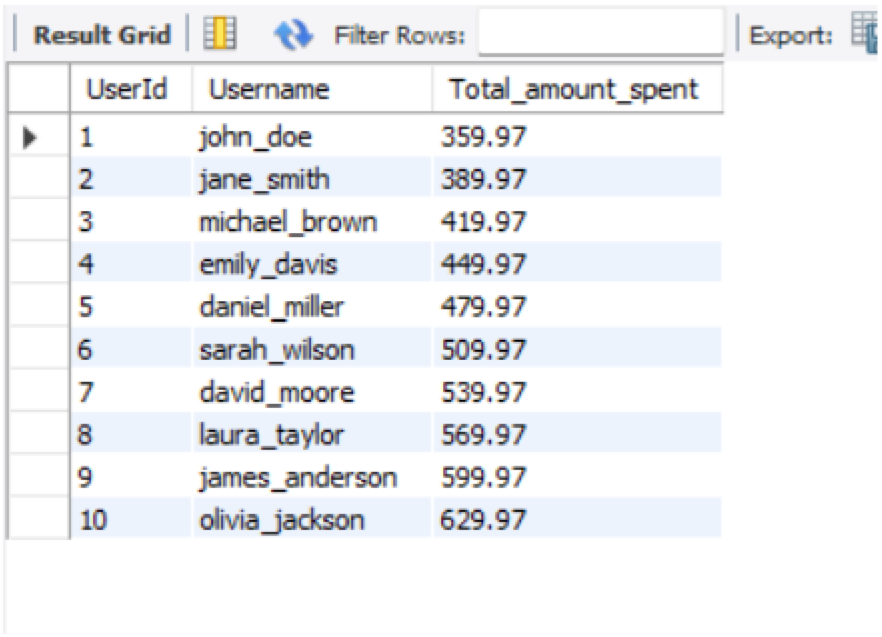
- **CURDATE()** function returns the current date in the format YYYY-MM-DD.

- **DATE_SUB(CURDATE(), INTERVAL 1 MONTH)** function subtracts one month from the current date.
- INTERVAL 1 MONTH specifies the amount of time to subtract.
- If today is 2024-07-18, then **DATE_SUB(CURDATE(), INTERVAL 1 MONTH)** would result in 2024-06-18.

8 . What is the total amount spent by each user on in-game purchases?
Syntax:

```
select User.UserId, User.Username, sum(InGamePurchase.Amount) as Total_amount_spent
from User
join InGamePurchase on User.UserId = InGamePurchase.UserId
group by User.UserId, User.Username;
```

Output:

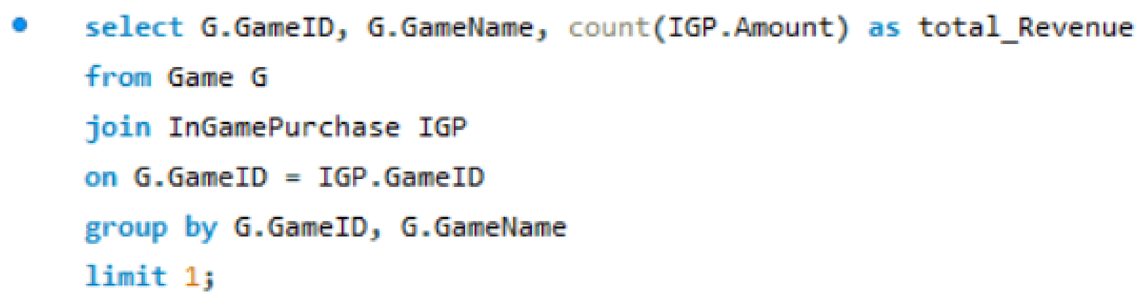


UserId	Username	Total_amount_spent
1	john_doe	359.97
2	jane_smith	389.97
3	michael_brown	419.97
4	emily_davis	449.97
5	daniel_miller	479.97
6	sarah_wilson	509.97
7	david_moore	539.97
8	laura_taylor	569.97
9	james_anderson	599.97
10	olivia_jackson	629.97

- The **SUM()** function will calculate the total sum of the amount column form the In game purchase table.
- Used **JOIN** functions joined the user and In game purchase table.

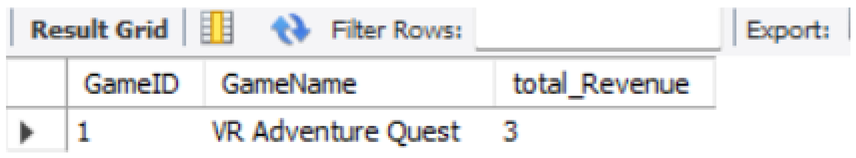
9 . Which virtual asset has generated the highest revenue?

Syntax:



```
select G.GameID, G.GameName, count(IGP.Amount) as total_Revenue
from Game G
join InGamePurchase IGP
on G.GameID = IGP.GameID
group by G.GameID, G.GameName
limit 1;
```

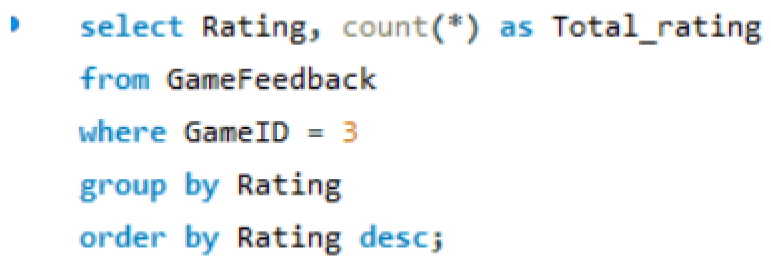
Output:



	GameID	GameName	total_Revenue
▶	1	VR Adventure Quest	3

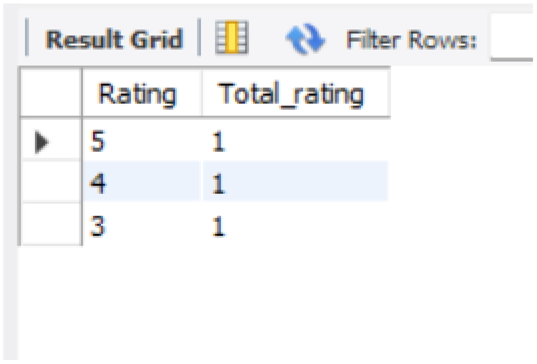
- The **COUNT()** function returns the number of rows that matches a specified criterion.
- Used **JOIN** functions joined the Game and In game purchase table.
- The limit function will limit the value based on the given condition.
- Here the limit is 1. So it gave only one output.

10 . What is the distribution of ratings given by users for a specific game?
Syntax:



```
select Rating, count(*) as Total_rating
from GameFeedback
where GameID = 3
group by Rating
order by Rating desc;
```

Output:



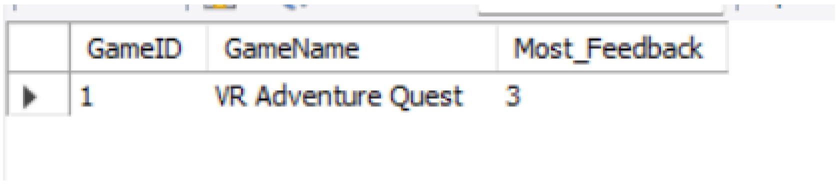
Rating	Total_rating
5	1
4	1
3	1

- The **COUNT()** function returns the number of rows that matches a specified criterion.
- By using where condition retrieved only the value of id 3.
- Ordered in descending **order by** using order by.

11. Which game has received the most feedback?
Syntax:

```
select G.GameID, G.GameName, count(GF.Rating) as Most_Feedback
from Game G
join GameFeedback GF on G.GameID = GF.GameID
group by G.GameID, G.GameName
order by Most_Feedback desc
limit 1;
```

Output:



GameID	GameName	Most_Feedback
1	VR Adventure Quest	3

- The **COUNT()** function returns the number of rows that matches a specified criterion.
- Joined 2 tables using JOINS.
- Grouped the game id and game name by using group by.
- Ordered in descending order by using order by.
- The limit function will limit the value based on the given condition.Here the limit is 1. So it gave only one output.

12 . Delete a DeveloperID column from the table Game
Syntax:

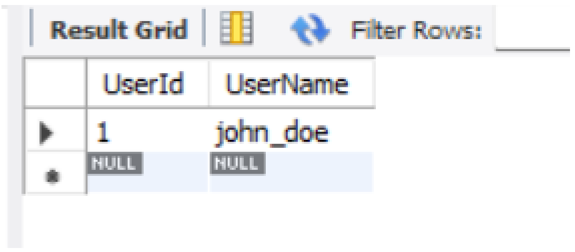
```
• alter table Game
  drop column DeveloperID;
```

- **Alter** is used to alter the table.
- **DROP** is used to delete the column and its values.

13 . select User who achieved a specific title
Syntax:

```
select UserId,UserName
from User
where UserId in (select UserId
from GameSession
where Achievements = "First Blood"
group by UserId);
```

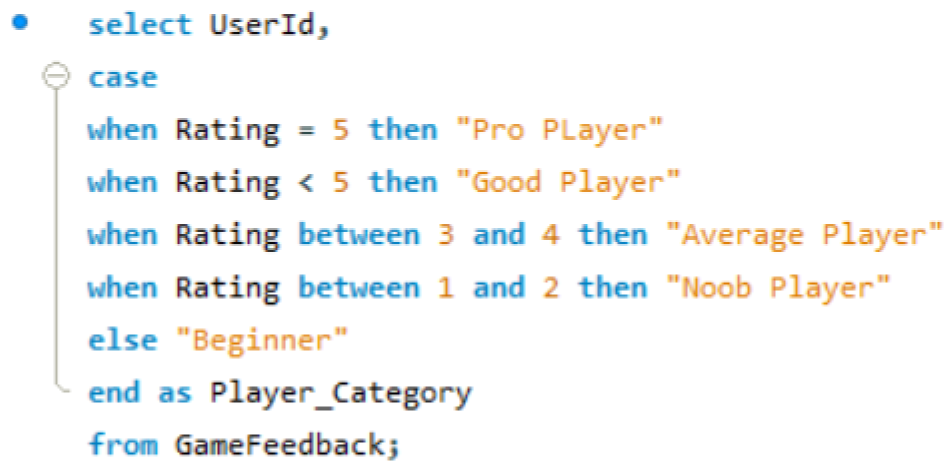
Output:



UserId	UserName
1	john_doe
NULL	NULL

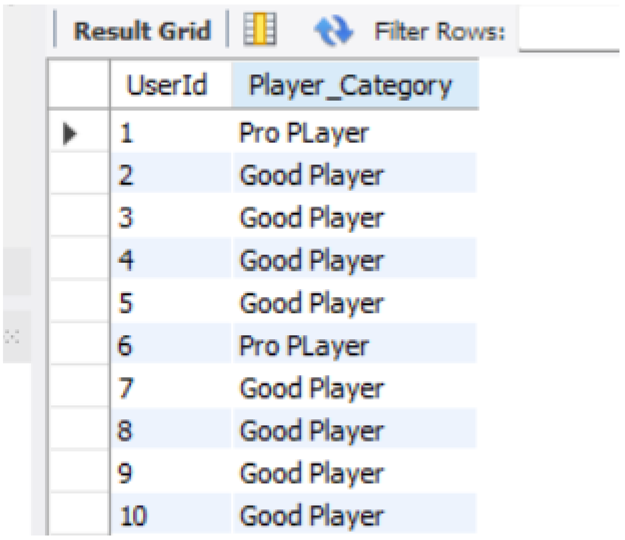
- Used **"SUB QUERY"** for this query.
- With the help of the sub query the UserID and User name has been retrieved.

14 . Specify the player category using case statement
Syntax:



```
select UserId,
case
when Rating = 5 then "Pro PLayer"
when Rating < 5 then "Good Player"
when Rating between 3 and 4 then "Average Player"
when Rating between 1 and 2 then "Noob Player"
else "Beginner"
end as Player_Category
from GameFeedback;
```

Output:



UserId	Player_Category
1	Pro PLayer
2	Good Player
3	Good Player
4	Good Player
5	Good Player
6	Pro PLayer
7	Good Player
8	Good Player
9	Good Player
10	Good Player

- Used case statement to specify the user category.
- The case statement started at the “Case” query and ended by declaring “end”.