Video games are big business: the global gaming market is projected to be worth more than $300 billion by 2027 according to Mordor Intelligence. With so much money at stake, the major game publishers are hugely incentivized to create the next big hit. But are games getting better, or has the golden age of video games already passed?

In this project, you'll analyze video game critic and user scores as well as sales data for the top 400 video games released since 1977. You'll search for a golden age of video games by identifying release years that users and critics liked best, and you'll explore the business side of gaming by looking at game sales data.

Your search will involve joining datasets and comparing results with set theory. You'll also filter, group, and order data. Make sure you brush up on these skills before trying this project! The database contains two tables. Each table has been limited to 400 rows for this project, but you can find the complete dataset with over 13,000 games on Kaggle.

`game_sales` table

Column	Definition	Data Type
name	Name of the video game	`varchar`
platform	Gaming platform	`varchar`
publisher	Game publisher	`varchar`
developer	Game developer	`varchar`
games_sold	Number of copies sold (millions)	`float`
year	Release year	`int`

`reviews` table

Column	Definition	Data Type
name	Name of the video game	`varchar`
critic_score	Critic score according to Metacritic	`float`
user_score	User score according to Metacritic	`float`

`users_avg_year_rating` table

Column	Definition	Data Type
year	Release year of the games reviewed	`int`
num_games	Number of games released that year	`int`
avg_user_score	Average score of all the games ratings for the year	`float`

`critics_avg_year_rating` table

Column	Definition	Data Type
year	Release year of the games reviewed	`int`
num_games	Number of games released that year	`int`
avg_critic_score	Average score of all the games ratings for the year	`float`

Projects Data ⊞ game_sales

in... ⋯ ↑↓	name ⋯ ↑↓	plat... ⋯ ↑↓	publisher ⋯
0	7 Days to Die for PC	PC	The Fun Pimps
1	ARK: Survival Evolved for PC	PC	Studio Wildcard
2	Age of Empires II: HD Edition for PC	PC	Microsoft Studios
3	Animal Crossing: City Folk for Wii	Wii	Nintendo
4	Animal Crossing: New Horizons for NS	NS	Nintendo
5	Animal Crossing: New Leaf for 3DS	3DS	Nintendo
6	Animal Crossing: Wild World for DS	DS	Nintendo
7	Arma 2: Operation Arrowhead for PC	PC	Meridian4
8	Arma III for PC	PC	Bohemia Interactive
9	Assassin's Creed II for PS3	PS3	Ubisoft
10	Assassin's Creed II for X360	X360	Ubisoft
11	Assassin's Creed III for PS3	PS3	Ubisoft
12	Assassin's Creed III for X360	X360	Ubisoft
13	Assassin's Creed Origins for PS4	PS4	Ubisoft
14	Assassin's Creed for PS3	PS3	Ubisoft
15	Assassin's Creed for X360	X360	Ubisoft

Rows: 150   ⤢ Expand

Projects Data   DataFrame as `best_selling_games`

```
-- best_selling_games
SELECT *
FROM game_sales
ORDER BY games_sold DESC
LIMIT 10;
```

index ⋯ ↑↓	name ⋯ ↑↓	platform ⋯ ↑↓	publisher ⋯
0	Wii Sports for Wii	Wii	Nintendo
1	Super Mario Bros. for NES	NES	Nintendo
2	Counter-Strike: Global Offensive for PC	PC	Valve
3	Mario Kart Wii for Wii	Wii	Nintendo
4	PLAYERUNKNOWN'S BATTLEGROUNDS for PC	PC	PUBG Corporation
5	Minecraft for PC	PC	Mojang
6	Wii Sports Resort for Wii	Wii	Nintendo
7	Pokemon Red / Green / Blue Version for GB	GB	Nintendo
8	New Super Mario Bros. for DS	DS	Nintendo
9	New Super Mario Bros. Wii for Wii	Wii	Nintendo

Rows: 10   ⤢ Expand

Hidden no-code
Hidden output

Projects Data   DataFrame as df2

```sql
SELECT *
FROM game_sales
INNER JOIN reviews ON game_sales.name = reviews.name;
```

	name		publisher	developer
0	ARK: Survival Evolved for PC	PC	Studio Wildcard	Studio Wildcard
1	Age of Empires II: HD Edition for PC	PC	Microsoft Studios	Hidden Path Entertainment
2	Animal Crossing: City Folk for Wii	Wii	Nintendo	Nintendo EAD
3	Animal Crossing: New Horizons for NS	NS	Nintendo	Nintendo
4	Animal Crossing: New Leaf for 3DS	3DS	Nintendo	Nintendo EAD
5	Animal Crossing: Wild World for DS	DS	Nintendo	Nintendo EAD
6	Assassin's Creed II for PS3	PS3	Ubisoft	Ubisoft Montreal
7	Assassin's Creed II for X360	X360	Ubisoft	Ubisoft Montreal
8	Assassin's Creed III for PS3	PS3	Ubisoft	Ubisoft Montreal
9	Assassin's Creed III for X360	X360	Ubisoft	Ubisoft Montreal
10	Assassin's Creed Origins for PS4	PS4	Ubisoft	Ubisoft Montreal
11	Assassin's Creed for PS3	PS3	Ubisoft	Ubisoft Montreal
12	Assassin's Creed for X360	X360	Ubisoft	Ubisoft Montreal
13	Assassin's Creed: Revelations for PS3	PS3	Ubisoft	Ubisoft Montreal
14	Assassin's Creed: Revelations for X360	X360	Ubisoft	Ubisoft Montreal
15	Asteroids for 2600	2600	Atari	Atari

Rows: 369   ↗ Expand

Projects Data   DataFrame as d

```sql
-- user_score_top_ten_years
SELECT game_sales.year, COUNT(game_sales.name) AS num_games, ROUND(AVG(reviews.user_score), 2) AS avg_user_score
FROM game_sales
INNER JOIN reviews ON game_sales.name = reviews.name
GROUP BY game_sales.year
HAVING COUNT(game_sales.name) >= 4
ORDER BY avg_user_score DESC
LIMIT 10;
```

		n.	avg_use…
0	1997	8	9.5
1	1998	10	9.4
2	2010	23	9.24
3	2009	20	9.18
4	2008	20	9.03
5	1996	5	9
6	2005	13	8.95
7	2006	16	8.95
8	2000	8	8.8
9	2002	9	8.8

Rows: 10   ↗ Expand

Projects Data   DataFrame as c

```sql
-- critics_top_ten_years
SELECT game_sales.year, COUNT(game_sales.name) AS num_games, ROUND(AVG(reviews.critic_score), 2) AS avg_critic_score
FROM game_sales
INNER JOIN reviews ON game_sales.name = reviews.name
GROUP BY game_sales.year
HAVING COUNT(game_sales.name) >= 4
ORDER BY avg_critic_score DESC
LIMIT 10;
```

···	↑↓	···	↑↓	n. ···	↑↓	avg_critic_s… ···	↑↓
0		1998		10		9.32	
1		2004		11		9.03	
2		2002		9		8.99	
3		1999		11		8.93	
4		2001		13		8.82	
5		2011		26		8.76	
6		2016		13		8.67	
7		2013		18		8.66	
8		2008		20		8.63	
9		2017		13		8.62	

Rows: 10   ↗ Expand

1 hidden cell

Projects Data   DataFrame as g

```sql
-- golden_years
SELECT users_avg_year_rating.year,
       users_avg_year_rating.num_games,
       users_avg_year_rating.avg_user_score,
       critics_avg_year_rating.avg_critic_score,
       (critics_avg_year_rating.avg_critic_score - users_avg_year_rating.avg_user_score) AS diff
FROM users_avg_year_rating
INNER JOIN critics_avg_year_rating ON users_avg_year_rating.year = critics_avg_year_rating.year
WHERE users_avg_year_rating.avg_user_score > 9 OR critics_avg_year_rating.avg_critic_score > 9
GROUP BY users_avg_year_rating.year,
         users_avg_year_rating.num_games,
         users_avg_year_rating.avg_user_score,
         critics_avg_year_rating.avg_critic_score
ORDER BY users_avg_year_rating.year ASC
LIMIT 10;
```

···	↑↓	···	↑↓	n. ···	↑↓	avg_use… ···	↑↓	avg_critic_s… ···	↑↓	···	↑↓
0		1997		8		9.5		7.93		-1.57	
1		1998		10		9.4		9.32		-0.08	
2		2004		11		8.55		9.03		0.48	
3		2008		20		9.03		8.63		-0.4	
4		2009		20		9.18		8.55		-0.63	
5		2010		23		9.24		8.41		-0.83	

Rows: 6   ↗ Expand