Report Project Databases: Game Industry

Kélian PONS, Clément MARTIN
1st year Master students, Master Data Science
Université de Lille

December 8, 2025

Contents

Introduction

The aim of the project for the Databases course is to create a database from a chosen data set. For this project we choose to create a database for the video game industry, with information on the games, the developers, the country and town of their headquarters.

1 Data of the project

We started by downloading data from different sources:

- Video game studios: From Kaggle, download here
- Games: From Huggingface, download here
- World cities: From Kaggle, download here
- World countries: From Github, download here

2 Modelling the database

2.1 Video game studios dataset

The video game studios dataset is a dataset from Kaggle, which contains information on the video game studios. The csv file contains the following atributes:

- **Developer**: name of the video game studio
- **City**: city where the video game studio is located
- **Administrative division**: administrative division of the city where the video game studio is located
- **Country**: country where the video game studio is located
- **Est.**: year when the video game studio was founded
- **Notable games, series or franchises**: list of notable games, series or franchises of the video game studio
- **Notes**: notes about the video game studio

2.2 Games dataset

The games dataset is a dataset from Huggingface, which contains information on the games. The csv file contains 40 atributes : AppID, Name, Release date, Estimated owners, Peak CCU, Required age, Price, Discount, DLCcount, About the game, Supported languages, Full audio languages, Reviews, Header image, Website, Support url, Support email, Windows, Mac, Linux, Metacritic score, Metacritic url, User score, Positive, Negative, Score rank, Achievements, Recommendations, Notes, Average playtime forever, Average playtime two weeks, Median playtime forever, Median playtime two weeks, Developers, Publishers, Categories, Genres, Tags, Screenshots, Movies,

We only kept the following atributes:

- **AppID**: The ID of the game.

- **Name**: The name of the game.
- **Release date**: The release date of the game.
- **Estimated owners**: The estimated range of owners of the game.
- **Required age**: The required age to play the game.
- **Price**: The price of the game.
- **DLCcount**: The number of DLCs in the game.
- **Supported languages**: The languages supported by the game.
- **Windows**: True if the game is supported on Windows, False otherwise.
- **Mac**: True if the game is supported on Mac, False otherwise.
- **Linux**: True if the game is supported on Linux, False otherwise.
- **Metacritic score**: The metacritic score of the game.
- **User score**: The user score of the game.
- **Positive**: The number of positive reviews of the game.
- **Negative**: The number of negative reviews of the game.
- **Achievements**: Number of achievements in the game.
- **Average playtime forever**: The average playtime of the game.
- **Developers**: The developers of the game.
- **Categories**: The categories of the game.
- **Genres**: The genres of the game.

2.3 World cities dataset

The world cities dataset is a dataset from Kaggle, which contains information on the world cities. The csv file contains the following atributes:

- **id**: id of the city.
- **City**: name of the city.
- **City ASCII**: name of the city in ASCII.
- **Latitude**: latitude of the city.
- **Longitude**: longitude of the city.
- **Country**: country of the city.
- **ISO2**: ISO2 code of the country.
- **ISO3**: ISO3 code of the country.
- **Admin name**: administrative name of the city.
- **Capital**: Information on the city as a capital.
- **Population**: population of the city.

2.4 World countries dataset

The world countries dataset is a dataset from Kaggle, which contains information on the world countries. The csv file contains the following atributes:

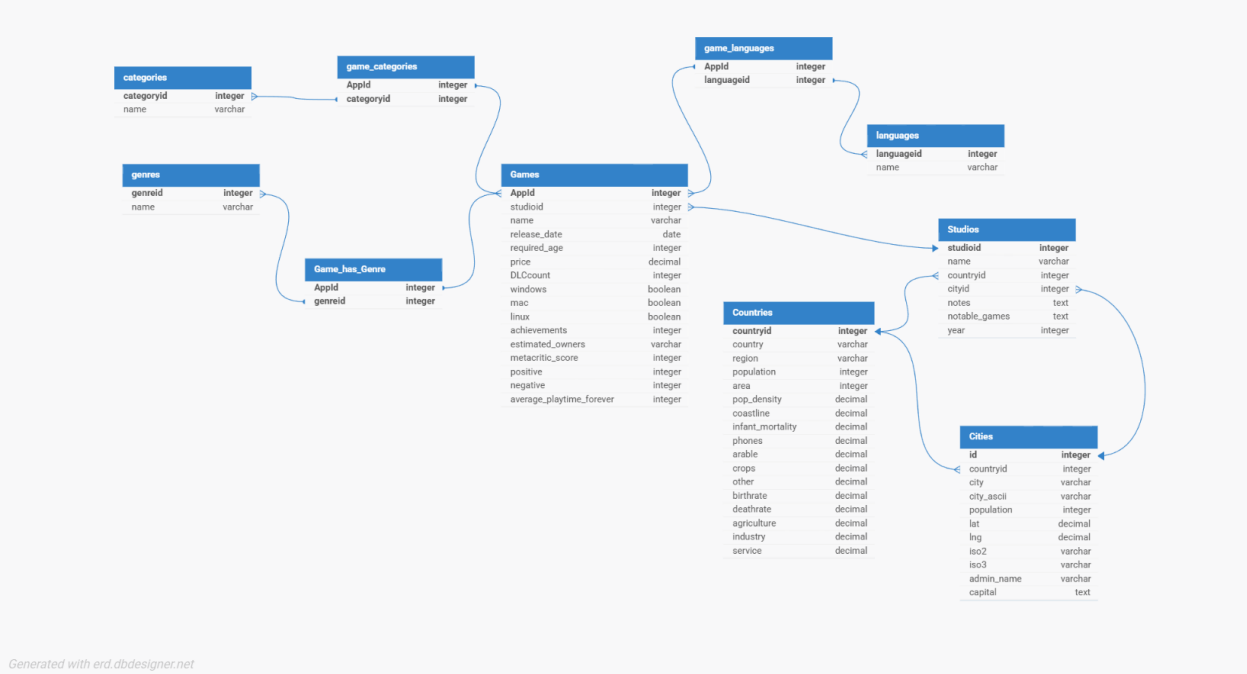
- **Country**: name of the country.
- **Region**: region of the country.
- **Population**: population of the country.
- **Area (sq. mi.)**: area of the country.
- **Pop. Density (per sq. mi.)**: population density of the country.
- **Coastline (coast/area ratio)**: coastline of the country.
- **Net migration**: net migration of the country.
- **Infant mortality (per 1000 births)**: infant mortality of the country.
- **GDP ($ per capita)**: GDP of the country.
- **Literacy (%)**: literacy of the country.
- **Phones (per 1000)**: phones of the country.
- **Arable (%)**: percentage of arable land of the country.
- **Crops (%)**: percentage of crops land of the country.
- **Other (%)**: percentage of other land of the country.
- **Climate**: climate of the country.
- **Birthrate**: birthrate of the country.
- **Deathrate**: deathrate of the country.
- **Agriculture**: some agriculture index of the country.
- **Industry**: some industry index of the country.
- **Service**: some service index of the country.

2.5 Modeling the database

After cleaning the data (see next section) we can now organize the tables. We choose to have the following tables with the primary keys in red and the foreign keys underlined:

- **countries**(countryid, country, region, population, area, pop_density, coastline, infant_mortality, phones, arable, crops, other, birthrate, deathrate, agriculture, industry, service)
- **cities**(id, city, city_ascii, lat, lng, countryid, iso2, iso3, admin_name, capital, population)
- **studios**(studioid, name, notable_games, notes, cityid, countryid, year)
- **games**(appid, name, studioid, release_date, required_age, price, dlccount, windows, mac, linux, achievements, estimated_owners, metacritic_score, positive, negative, average_playtime_forever)
- **categories**(name, categoryid)
- **genres**(name, genreid)
- **languages**(name, languageid)
- **game_genres**(appid, genreid)
- **game_categories**(appid, categoryid)
- **game_languages**(appid, languageid)

Database Figure



3 Population of tables

3.1 Modifications for Games

Since game can have multiple **genres**, the genres were split into additional tables (and csv files). We first create a table containing the **genreid** and **name** of the genres. Then we create another table containing the **appid** and the **genreid** , since the game can have multiple genres one AppID can appear in multiple rows. Finally we remove the original genres column from the games.

The same method was used for the categories and languages supported atributes. So we ended up with 6 new tables.

To handle the link with the studios table we modified the **Developers** column to match the **studioid**. For the Developers name that was not in the studios table we decided to drop the rows. Since we drop the missmatched data we can set the **studioid** column to not null, in orther words every game must have a studio that is in the studios table.

The games table contains 1670 games.
The genres table contains 28 genres.
The categories table contains 29 categories.
The languages table contains 121 languages.

3.2 Modifications for Studios

Frist manual cleaning (87 lines) was done on video-games-developers.csv to match the other datasets. Especially the **City** and **Country** columns, which contain non standard names. You can find all the manual modification in the modif_video_game_studio.json file.

Example of missmatched data between studios and cities datasets

```
Ultimate Play the Game,"Ashby-de-la-Zouch",England,United Kingdom,1982,JetpacLunar JetmanAtic AtacSabreman
```

Video game studios dataset

```
"Ashby de la Zouch","Ashby de la Zouch","52.7460","-1.4760","United Kingdom","GB","GBR","Leicestershire","",
```

World cities dataset

Then a DeveloperID base on the Developer column was created to be the primary key of the table. The City column was modified to **cityid** to match the cities table and be a foreign key. The Country column was also modified to **countryid** to match the countries table and it will be the foreign key to the countries table.

The final studios table contains 686 studios.

3.3 Modifications for Cities

For this dataset we had some missmatched and missing data for the countries. We decided to handle manually only the countries that were also present in the studios table (only the Czech Republic). We dropped the others and modified the country column to **countryid** to be the foreign key to the countries table.

The final cities table contains 47453 cities.

3.4 Modifications for Countries

For this dataset the only modification was to add a new column called **countryid** to be the primary key of the table. Some data formatting was also done to match the sql formats.

The final countries table contains 227 countries.

3.5 Table Creation

We started by creating the table for the countries since it does not have any foreign keys. We then add the cities table and the studios table with the constraints of the foreign keys. Then we added the games table, the genres table, the categories table and the languages table. We finished with the game_genres table, game_categories table.

We alter some of the table because we put too restrictive constriants on some atributes. We then add the data using the csv files. To see the table creation in detail you can see the **table_creation.py** file.

4 Querying

Free Games

```
SELECT name
FROM games
WHERE price = 0
LIMIT 10;
```

name
Apex Legends
The Lab
Back to the Future: The Game
Warhammer: Vermintide VR - Hero Trials
DRAGON QUEST XI: Echoes of an Elusive Age - Digital Edition of Light
STAR WARS: The Old Republic
BRINK
Prime World: Defenders 2
Resident Evil: Operation Raccoon City
Darkfall Unholy Wars

This allows us to identify titles accessible to players without any cost.

Games sorted by price

```
SELECT name, price
FROM games
ORDER BY price DESC
LIMIT 10;
```

name	price
Call of Duty: Ghosts - Digital Hardened Edition	99.990000
Microsoft Flight Simulator 2024	69.990000
F1 23	69.990000
Suicide Squad: Kill the Justice League	69.990000
FINAL FANTASY VII REMAKE INTERGRADE	69.990000
NBA 2K25	69.990000
F1 24	69.990000
Need for Speed Unbound	69.990000
Starfield	69.990000
Indiana Jones and the Great Circle	69.990000

We want to list all games sorted by price, as price is an important factor when choosing a video game.

Number of games by genre

```
SELECT ge.name, COUNT(*) AS "nb of game"
FROM genres ge
JOIN game_genres gg ON ge.genreID = gg.genreID
GROUP BY ge.genreID
ORDER BY "nb of game" DESC
LIMIT 10;
```

name	nb of game
Action	827
Adventure	572
RPG	373
Strategy	316
Indie	277
Simulation	273
Casual	190
Sports	109
Racing	97
Free to Play	73

It is interesting to determine which genres are the most developed by studios worldwide.

All game with a metacritic score after 2020

```
SELECT name, release_date, metacritic_score
FROM games
WHERE release_date >= '2020-01-01'
AND metacritic_score > 0
ORDER BY metacritic_score DESC
LIMIT 10;
```

name	release_date	metacritic_score
Mass Effect 2 (2010) Edition	2023-05-15	94
Half-Life: Alyx	2020-03-23	93
Mission: It's Complicated	2020-02-14	91
Crusader Kings III	2020-09-01	91
Microsoft Flight Simulator Game of the Year Edition	2020-04-17	91
The Making of Karateka	2023-08-29	90
Psychonauts 2	2021-08-24	89
Mass Effect 3 N7 Digital Deluxe Edition (2012)	2020-06-11	89
Apex Legends	2020-11-04	88
DOOM Eternal	2020-03-19	88

We want all games which have a metacritic score after 2020 since 2020 marks the beginning of the COVID-19 pandemic and the video game' s bargains exploded.

Number of studios by country

```
SELECT c.country, COUNT(*) AS nb_stud
FROM studios s JOIN countries c ON s.countryid = c.countryid
GROUP BY c.country
ORDER BY nb_stud DESC
LIMIT 10;
```

country	nb_stud
United States	232
Japan	150
United Kingdom	73
France	24
Canada	21
Germany	19
Sweden	17
Korea, South	14
Poland	13
Australia	11

This allows us to analyze global industry distribution and understand which countries contribute the most in the video game world.

Biggest Cities with at least 1 studio for each country

```
SELECT co.country, c.city, c.population
    FROM studios s JOIN cities c ON s.cityid = c.id
    JOIN (SELECT MAX(c.population) AS max_pop, c.countryid
          FROM studios s JOIN cities c ON s.cityid = c.id
          GROUP BY c.countryid) AS table_max_pop
    ON table_max_pop.countryid = s.countryid
    AND table_max_pop.max_pop = c.population
    JOIN countries co ON co.countryid = c.countryid
    GROUP BY co.country, c.city, c.population
    ORDER BY population DESC;
```

country	city	population
Japan	Tokyo	37785000
China	Guangzhou	26940000
Philippines	Manila	24922000
Korea, South	Seoul	23016000
Mexico	Mexico City	21804000
United States	New York	18832416
Russia	Moscow	17332000
Argentina	Buenos Aires	16710000
India	Bangalore	15386000
Turkey	Istanbul	14441000
United Kingdom	London	11262000
France	Paris	11060000
Malaysia	Kuala Lumpur	8911000
Vietnam	Hanoi	8587100
South Africa	Johannesburg	7860781
Chile	Santiago	7171000
Spain	Madrid	6211000
Australia	Melbourne	5031195
Germany	Berlin	4679500
Canada	Montréal	3675219
Greece	Athens	3059764
Ukraine	Kyiv	2952301
Italy	Rome	2748109
Taiwan	Taipei	2494813
Romania	Bucharest	2412530

It provides insight into where the game industry is concentrated inside each country.

It shows whether the largest urban centers are also the most active in game development.

Games with the Polish languages supported and the category Multi-player and that are not the Adventure game

```
SELECT g.name
FROM games g
JOIN game_languages gl ON g.appid = gl.appid
JOIN languages l ON l.languageid = gl.languageid
JOIN game_categories gc ON g.appid = gc.appid
JOIN categories c ON c.categoryid = gc.categoryid
WHERE l.name = 'Polish' AND c.name = 'Multi-player'
EXCEPT
SELECT g.name
FROM games g
JOIN game_genres gg ON g.appid = gg.appid
JOIN genres ge ON ge.genreid = gg.genreid
WHERE ge.name = 'Adventure'
ORDER BY name
LIMIT 10;
```

name
4th & Inches
8-Bit Armies
8-Bit Hordes
8-Bit Invaders!
9-Bit Armies: A Bit Too Far
ACL Pro Cornhole
Act of Aggression - Reboot Edition
Act of War: Direct Action
Act of War: High Treason
Actua Golf

It allows us to study how many games meet very specific combined criteria which is useful to compare between different genres and categories.

The most played game for each genres

```
SELECT ge.name AS "Game Genres", MAX(g.name) AS "The most played game",
       MAX(g.average_playtime_forever) AS "Time played"
FROM games g
JOIN game_genres as gg ON gg.appid = g.appid
JOIN genres as ge ON ge.genreid = gg.genreid
JOIN (SELECT gg.genreid AS genreid, MAX(g.average_playtime_forever) as max_time_played
      FROM games g
      JOIN game_genres as gg ON gg.appid = g.appid
      GROUP BY gg.genreid) as table_genre_maxtime
      ON table_genre_maxtime.genreid = gg.genreid
      AND table_genre_maxtime.max_time_played = g.average_playtime_forever
GROUP BY ge.name
ORDER BY "Time played" DESC;
```

Game Genres	The most played game	Time played
Strategy	Dota 2	37162
Action	Dota 2	37162
Free to Play	Dota 2	37162
Massively Multiplayer	FINAL FANTASY XIV Online	27478
RPG	FINAL FANTASY XIV Online	27478
Casual	MARVEL Puzzle Quest	18149
Adventure	Rust	16623
Indie	Rust	16623
Simulation	Arma 3	12276
Sports	NBA 2K20	9237
Racing	Micro Machines World Series	7004
Early Access	Mount & Blade II: Bannerlord	5482
Video Production	Source Filmmaker	1846
Animation & Modeling	Source Filmmaker	1846
Violent	Pathfinder Adventures	649
Sexual Content	Mary Skelter: Nightmares	400
Nudity	X-Blades	291
Web Publishing	Multiplicity	2
Audio Production	Multiplicity	2
Design & Illustration	Multiplicity	2
Education	Multiplicity	2
Photo Editing	Multiplicity	2
Software Training	Multiplicity	2
Utilities	Multiplicity	2
Accounting	Multiplicity	2
Game Development	Starfield: Creation Kit	0
Gore	Warhammer: Vermintide VR - Hero Trials	0
Free To Play	Project Fireball	0

We retrieve the most played game for each genre in order to help the customer to find the best option within their preferred genres.

Top 3 games in 2020 which has at least 3 supported languages

```
SELECT ga.name as "Game name", ga.metacritic_score as "Metacritic score",
       COUNT(gl.languageID) AS "language count"
FROM games ga
JOIN game_languages AS gl USING (appid)
WHERE ga.release_date >= '2020-01-01' AND ga.release_date < '2021-01-01'
GROUP BY ga.appid
HAVING COUNT(gl.languageID) >= 3
ORDER BY ga.metacritic_score DESC
LIMIT 3;
```

Game name	Metacritic score	language count
Half-Life: Alyx	93	12
Microsoft Flight Simulator Game of the Year Edition	91	12
Crusader Kings III	91	12

We retrieve the top 3 games of 2020 that support at least three languages, since global accessibility became especially important during the pandemic

Average games prices for each studio in Tokyo or New York or Montréal with more than 3 games

```
SELECT s.name as "studio name", AVG(ga.price) as "average price",
       COUNT(ga.name) as "nb of game",ci.city as city , ci.iso3 as country
FROM games ga
JOIN studios AS s USING(studioid)
JOIN cities AS ci ON ci.id = s.cityid
GROUP BY s.studioid, s.name, ci.iso3, ci.city
HAVING (ci.city = 'Tokyo' OR ci.city = 'New York' OR ci.city = 'Montréal')
       AND COUNT(ga.name) >= 3
ORDER BY "average price"
LIMIT 20;
```

studio name	average price	nb of game	city	country
Rockstar Games	14.578000	10	New York	USA
FromSoftware	19.996667	3	Tokyo	JPN
Idea Factory	22.767778	9	Tokyo	JPN
Compulsion Games	25.990000	3	Montréal	CAN
Square Enix	26.680877	57	Tokyo	JPN
Tango Gameworks	29.992000	5	Tokyo	JPN
Tamsoft	32.212222	9	Tokyo	JPN
Nihon Falcom	33.606667	30	Tokyo	JPN
Tokyo RPG Factory	46.656667	3	Tokyo	JPN

This query is interesting because it allows us to compare pricing strategies across different regions of the global video game industry since these cities represent important cultural, economic, and technological centers.

5 Database structure update

We can imagine extending our database so that it no longer includes only games from Steam, which focuses on computer games. Instead, we may want to incorporate games from others `platforms` , such as the *PlayStation 5* or *Nintendo Switch*.

To do this, we need to obtain a data set that lists all available platforms and create a dedicated table called, for example, platform, that stores information about each platform.

We then introduce an associative table, `game_platform`, which links `games` to `platforms`. This relationship is many-to-many, since multiple games can be available on several platforms, and each platform can support many games.

To design the platform table, we assign a serial integer as its primary key and include a non-null name attribute. We may also store additional information, such as the release date, price, or total sales, for each platform.

Finally, the `game_platform table` must contain two attributes that act as foreign keys referencing the `game` and `platform` tables. Together, these two attributes should form a composite primary key to ensure that each game–platform pair is unique.