How did we do the Battles database

Description

The battles database is a historically accurate database containing most of the bellicose encounters from the year 1600 until today. However, there is far more than this in this database: we can extract information about the countries that participated in it, the weather during the battles, the type of terrain, the duration of the battles…

I must add that this database is a potential source for research about all kinds of factors that affect the duration of the battles or the bellicosity of a country over time. That is why we chose it!

Disclaimers

Some of the data about battles and commanders was lost due to some of the keys not being unique (although we managed to save some of them).

Also, everything related to weapons and uniforms is completely made up, it is randomly generated data.

Resources and creation process

- https://www.kaggle.com/residentmario/database-of-battles
    - That is the link we got the data from. However, there was some keys that lacked a description, such as "logsa" or "momta" (the names make no sense). So, we had to find a solution: either we delete those values, or we find a description somewhere.

- https://github.com/jrnold/CDB90
    - Luckily, in the description of the database in Kaggle, we found a link to a GitHub repository containing a Linux program to generate the database for us from a JSON file containing all the information. However, the program did not work (it was outdated) and we did not manage to make it work, therefore we had to create our own solution.

- The JSON file
    - Finally! The solution for all our problems! (not really)
    - We created a python script to analyze and extract all the information about the database. Then, it generates the basic layout of the database in PostgreSQL language:
        - The tables with each column and its datatype
        - Descriptions for each column and each table
        - A ton of errors

- Then, we spent about three weeks trying to fix the errors generated from the JSON, creating the new made-up tables (and their data) and adding all the entity relationships.
- Sadly, the JSON did not contain **any** information about which were the primary keys for each table or the entity relationships, so it took a long time to figure that out.

- Made-up data
  - The list of weapons and their descriptions was extracted from Wikipedia.
  - The list of uniform colors was completely made up.
  - Excel has a beautiful function called *RANDBETWEEN*, which helped us generate the random data (we also created other function to select random data from a set of predefined values). However, we faced a serious problem with the primary keys (unique values repeating and foreign keys not existing). We had to create a complex Excel function to delete the repeated values and the ones that did not exist as foreign keys. As an example, from 1300 values generated in actor_uniforms, only 23 were valid…

Requirements

Writing the text

In order to write the text, we have taken as an example an exercise from class about generated/consumed electricity in a country. When specifying the attributes, we have looked the JSON and at the same time, checked the values in the data provided in the CSVs. Also, we have tried to make clear the relationships between each of the tables. Since some data was made-up there can be inconsistencies in some parts of the text. We have tried to make the text more entertaining by using modern references such as MET; however, we know that given the amount of data and attributes it stores, the text ended up being a bit dense.

Writing the queries

The data in the CSVs, the JSON and some classroom exercises were taken as an inspiration to write the queries. We tried making them as diverse as possible and make them involve more than one table.

Commentaries

If we had to re-do everything from scratch, we would change several things:

- Naming
  - As previously mentioned, some of the attributes have very confusing names. If we could start all over again, we would rename most of the things to specify what they are (for example "*wxa*" → "*degInfluenceTerrain*").
  - We would have named the made-up tables consistently with the naming of the dataset, that is, with a "enum_" before the name to specify that it is a list of data.

- Descriptions
    - In PostgreSQL there exists "*comment on table foo is 'something'*", which allows us to assign descriptions to table outside the code too, facilitating queries and understanding the data.
    - Using that knowledge, we would have generated all the descriptions from the JSON f
    - file (we tried but there were issues with some of the special characters).

Work distribution

- Silvia Rodríguez Bares
  Database text, queries (and checked the queries for mistakes) and helping with the problems of the creation of the database.

- Marcos Gutiérrez Alonso
  I made the database architecture, the data extraction (descriptions on the attributes), the problem solving and checked the queries for mistakes.

Conclusion

This is an interesting database, a cool project and something we can expand on. I honestly think that is a shame that we had to introduce "fake" data, but overall, I think the result is pretty good.

Also, I think there was a mistake defining the primary keys for battle_actors because we had to eliminate quite a few from the original database for it to work (we tried different combinations of primary keys).