World Heritage Sites: the 2024 database

Databases Project Report
Conclusion Date: 8-12-2024

Developed by the group: G1E
up202303768 Gonçalo Almas
up202306707 Maria Mota
up202307836 Pedro Pereira
up202304752 Simão Gomes

U.PORTO
FACULDADE DE CIÊNCIAS
UNIVERSIDADE DO PORTO

Universe in question and description of the Database

Theme: World Heritage Sites

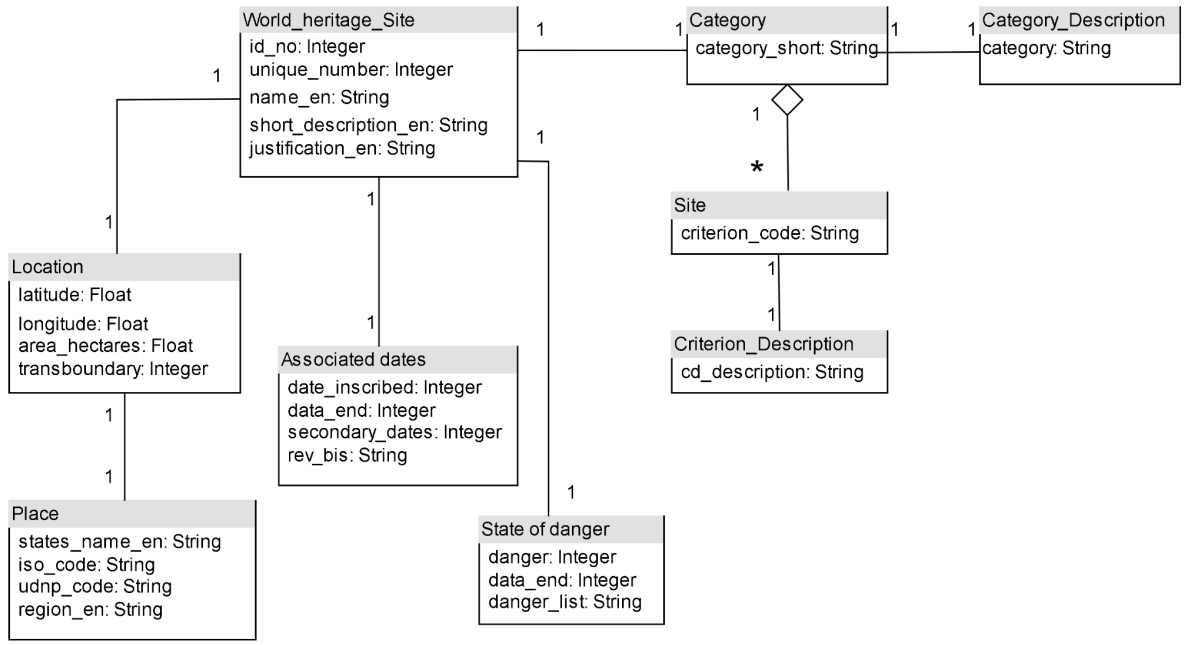
This database documents UNESCO's World Heritage Sites, recognized by their cultural, historic, scientific or natural relevance. This list is crucial in the promotion of conservation of cultural and/or natural appreciation on a global scale. By shining a spotlight on these sites of extreme importance, UNESCO intends to raise international awareness of the need to protect these places, assuring their preservation for the generations to come.

The scope of the Data Set ranges from the creation of the World Heritage Sites list, in 1978, until the most recent data available (2024). This time range allows us to accompany its evolution including when sites were added, reviewed and movements on the endangered list across the years.

We have a global range, including sites in every continent and region on the planet. There's a regional division that takes into consideration both the geographical and political divisions and the cultural ones. Some of the sites present in this database are transboundary and extend across multiple countries.

Each of the database's entries reflect a unique site and their most important characteristics in an uniform and easy way to read. It also includes a description of the site as well as the reason(s) as to why it is included in the World Heritage List.

UML Diagram



Notes:

- ➔ World_Heritage_Site represents the central table of the database which contains general information of the sites (name, description, justification) and is related directly or indirectly with every other table, reflecting its importance.
- ➔ Initially, we had a column named “criteria_txt” which is a repetition of the information on the column C1 through N10, thus we opted to exclude its inclusion.
- ➔ We adapted the columns C1 through N10 in a table where, to each site_number corresponds its criterion to simplify the interaction with the database and to include information not previously present (the description of each criterion).

Example:

For site_number = 208 we had:

C1	C2	C3	C4	C5	C6	N7	N8	N9	N10
1	1	1	1	0	1	0	0	0	0

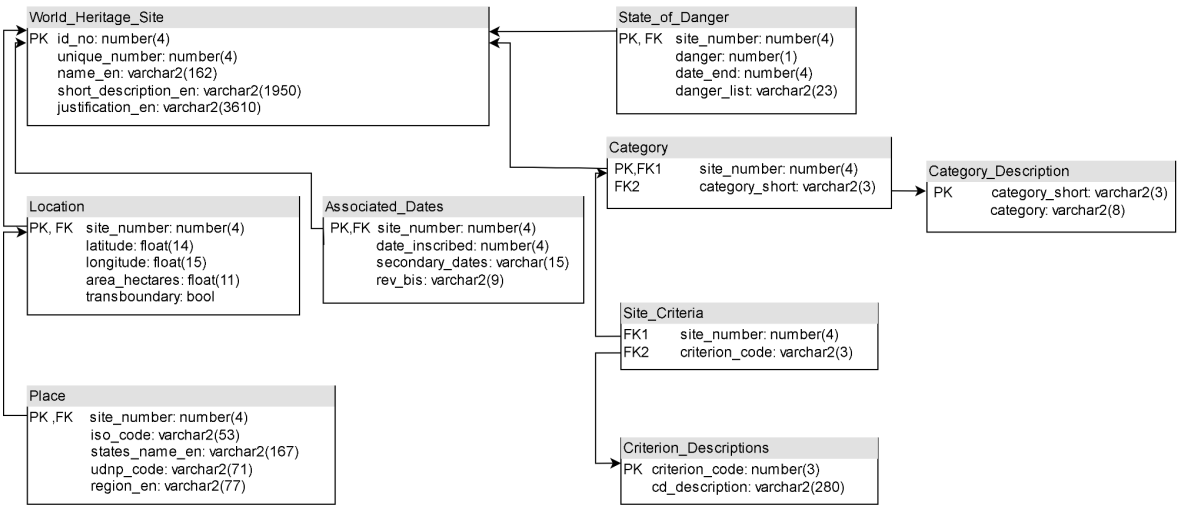
We turned it into:

site_numeber	criterion_code
208	C1
208	C2
208	C3
208	C4
208	C6

Like what happened with the criterion, we could have done something similar with the transboundary countries and separated the different countries.

Besides adding a big amount of entries to our database and since we have 49 transboundary sites having one that spans across 18 different countries, we would need a way to ensure that the iso_code, udnp_code and the states_name_en would match, and seeing that there isn’t a rule that applies to all we would have to do that, for example, with dictionaries.

RM Diagram



Notes:

- ➔ The table Site_Criteria only having FK is a consequence of its role as a connection table between two different identities: Category and Criterion Descriptions.
- ➔ Defining variables as Primary Keys and Foreign Keys improves the efficiency of the consultations, since the PKs are automatically indexed, allowing for easier access and clarity.

Populating of the tables

Table Name	Number of lines
World_Heritage_Site	1223
Location	1223
Place	1223
Associated_Dates	1223
State_of_Danger	1223
Category	1223
Category_Description	3
Site_Criteria	2876
Criterion_Description	10

Based on the .xlsx file provided to us we used a Python script to create the .db file and to transfer the data from one file to another. In that script we used the pandas library and the sqlite3 library.

The pandas library is a very popular data analysis and manipulation python library and the sqlite3 library lets us use SQL queries in Python, the combination of those libraries allowed us to create our “World_Heritage_Sites.db” file and populate it.

We started by dividing our different tables and its data through different sheets of our .xlsx file, then we created the tables. For example:

```
cursor.execute("""
CREATE TABLE Location (
        site_number INTEGER PRIMARY KEY,
        latitude FLOAT,
        longitude FLOAT,
        area_hectares FLOAT,
        transboundary BOOLEAN,
        FOREIGN KEY (site_number) REFERENCES
        World_Heritage_site(id_no) );
         """)
```

This created a file with empty tables. Then, we proceeded to populate the tables we created.

```
df = pd.read_excel(path, sheet_name="Location")
df.to_sql("Location", conn, if_exists="append", index=False)
```

The process of simplifying the C1-N10 columns

```
df_criteria = pd.read_excel(path, sheet_name="Criteria")

# Colocar os dados horizontais verticalmente
df_long = df_criteria.melt(
    id_vars=['site_number'], #Colunas a manter iguais
    value_vars=['C1', 'C2', 'C3', 'C4', 'C5', 'C6', 'N7', 'N8', 'N9', 'N10'], #Colunas a 'derreter'
    var_name='criterion_code', #Nome da nova coluna que tem os valores 'melted'
    value_name='has_criterion') #Nome da coluna que tem os valores que estavam na coluna 'melted'

#Deixar só os critérios aplicáveis
df_filtered = df_long[df_long['has_criterion'] == 1].drop(columns=['has_criterion'])

df_filtered.to_sql(
    'Site_Criteria', #Tabela que vai guardar os novos dados (id_no criterio)
    conn,
    index=False)
```

SQL Queries

1. **Top 5 countries with the biggest amount of non-transboundary sites. Show the country name and the amount of sites, and order them (DESC) by amount.**

```
SELECT states_name_en AS states_name, COUNT(*) AS count
FROM Place
NATURAL JOIN Location
GROUP BY states_name_en
HAVING transboundary=0
ORDER BY count DESC
LIMIT 5;
```

2. **The names, latitude and longitude of the sites below the equator (latitude<0) and to the left of Greenwich meridian (longitude<0)? Order them from North to South.**

```
SELECT id_no, name_en, l.latitude, l.longitude
FROM location l
JOIN world_heritage_site whs ON l.site_number = whs.id_no
WHERE l.latitude < 0 AND longitude < 0
ORDER BY l.latitude DESC;
```

3. **Which sites does Unesco consider representative of “a masterpiece of human creative genius”? Order them by name.**

```
SELECT whs.id_no, whs.name_en
FROM site_criteria st
JOIN world_heritage_site whs ON whs.id_no = st.site_number
WHERE criterion_code LIKE (
   SELECT cd.criterion_code
   FROM criterion_descriptions cd
   WHERE cd.cd_description LIKE "%To represent a masterpiece of human creative
genius%")
ORDER BY whs.name_en;
```

4. **Which sites, ID and name, became a part of the list in the year we were born(2005)? Order them by name.**

```
SELECT whs.id_no, whs.name_en AS name
FROM associated_dates ad
JOIN world_heritage_site whs ON ad.site_number = whs.id_no
WHERE ad.date_inscribed = 2005
ORDER BY name;
```

5. **Has any site stopped being endangered this year (2024)? Show their ID's, names and descriptions, and order by name.**

```
SELECT id_no, name_en AS name, short_description_en AS description
FROM state_of_danger sd
JOIN world_heritage_site whs ON sd.site_number = whs.id_no
WHERE sd.date_end = 2024
ORDER BY name;
```

Reason: https://whc.unesco.org/en/news/2703

6. **Which site(s),show the ID and Name, combine the biggest amount of criteria to be considered World Heritage and what is that amount as 'Number of criteria'? Order them by name.**

```
WITH numb_criteria AS (SELECT site_number, count(site_number) AS cont_criteria
FROM site_criteria sc
GROUP BY sc.site_number
ORDER BY sc.site_number)

SELECT whs.id_no, whs.name_en AS name, cont_criteria
FROM numb_criteria nc
JOIN world_heritage_site whs ON nc.site_number = whs.id_no
WHERE cont_criteria = (
   SELECT max(cont_criteria)
   FROM numb_criteria nc
   )
ORDER BY name;
```

7. **How many sites are there from each category? Show the category name and the amount of sites.**

```
SELECT cd.category as category, count(c.site_number) as count
FROM category c
JOIN category_description cd on c.category_short = cd.Category_short
GROUP BY c.category_short;
```

8. **What are the names, region and countries (transboundary sites) of the sites in France? Order them by name.**

```
SELECT id_no as site_number, name_en as site_name, region_en as info_1,
states_name_en as info_2
FROM World_Heritage_Site
JOIN Place ON World_Heritage_Site.id_no = Place.site_number
WHERE states_name_en LIKE '%France%'
ORDER BY site_name;
```

9. **The sites associated with countries that start with the letter B and the name of their country/countries. Show their ID, name and country or countries, and order them by ID.**

```
SELECT whs.id_no as site_number, whs.name_en as site_name, p.states_name_en
FROM World_Heritage_Site whs
JOIN Place p ON whs.id_no = p.site_number
WHERE p.states_name_en LIKE 'B%' OR p.states_name_en LIKE '%,B%'
ORDER BY whs.id_no;
```

10. **What is the average area of the sites of each category? Show the category name and the average area.**

```
SELECT category_short, AVG(area_hectares) AS avg_area
FROM Location
JOIN Category ON Location.site_number = Category.site_number
GROUP BY category_short;
```

11. **What is the amount of sites in each hemisphere (North and South)?**

```
SELECT CASE WHEN latitude >= 0 THEN 'North' ELSE 'South' END AS hemisphere,
COUNT(*) AS site_count
FROM Location
GROUP BY hemisphere;
```

12. **List the 10 biggest sites in area and their regions. Show their ID, name, area, and region, and order by area (DESC).**

```
SELECT whs.id_no,whs.name_en, l.area_hectares,p.region_en
FROM World_Heritage_Site whs
JOIN Location l ON whs.id_no = l.site_number
JOIN Place p ON whs.id_no = p.site_number
ORDER BY l.area_hectares DESC
LIMIT 10;
```

13. **List the sites in the torrid zone (between the tropics of cancer and capricorn) that are currently in danger. Show their ID, name, latitude, longitude, and order them by ID.**

```
SELECT id_no, name_en, latitude, longitude
FROM Location as l
JOIN World_Heritage_Site as whs ON l.site_number = whs.id_no
JOIN State_Of_Danger as d ON whs.id_no=d.site_number
WHERE danger=1 AND latitude BETWEEN -23.5 AND 23.5
ORDER BY site_number;
```

14. **Find the sites that belong to the region with the biggest amount of sites. Show their ID, name and the region, and order by ID.**

```
SELECT id_no, whs.name_en, p.region_en
FROM World_Heritage_Site whs
JOIN Place p ON whs.id_no = p.site_number
WHERE P.region_en = (
        SELECT region_en
        FROM Place
        GROUP BY region_en
        ORDER BY COUNT(*) DESC
        LIMIT 1)
ORDER BY whs.id_no;
```

15. **What were the sites added during the first 10 years of the World Heritage sites list? Show their ID_no, name, the country or countries where it is located, and the date of inscription and order them by date.**

```
WITH first_years AS (
        SELECT DISTINCT date_inscribed
        FROM Associated_Dates
        ORDER BY date_inscribed
        LIMIT 10)

SELECT whs.id_no, whs.name_en AS name, p.states_name_en, as.date_inscribed
FROM associated_dates ad
JOIN world_heritage_site whs ON ad.site_number = whs.id_no
JOIN place p ON whs.id_no=p.site_number
WHERE ad.date_inscribed IN first_years
ORDER BY ad.date_inscribed;
```

Python Application: endpoints

/	A small section about the database and some interesting stats about the sites as well as the place to search sites by different parameters.
/sites/	Lists all the World Heritage Sites.
/sites/<int:id>/	Retrieves all the relevant information about a site specified by its respective ID.
/sites/<int:id>/criteria/	Exhibits all the criteria of the respective site identified by its ‘id_no’.
/sites/search/<country>	Allows us to search World Heritage Sites by country and list them by id_no and name.
/sites/search/<int:year>	Searches the World Heritage Sites by year of inscription on the list.
/transboundary/	Lists the transboundary sites.
/transboundary/search/<country>	Allows us to search transboundary World Heritage Sites by country and list them by id_no and name.
/country/<country>	Lists all the sites in a certain country contained on the list.
/sites/category/<category>	Lists all the sites of a determined category (<category> can take the values: Cultural, Mixed or Natural).
/sites/danger	Lists all the currently endangered sites.
/sites/danger/not-in-danger	Lists all the currently non-endangered sites.
/queries/<int:n_pergunta>	Exhibits the answer to a query based on its number (n_pergunta), you can also check a suggestion of one of the SQL queries that leads you to the intended result.
/authors	Info of the project’s authors.

Notes: We used dynamic parameters, for example <country>, which can be directly associated with the name of the country the user searches.

Issues:

- ➔ On our index, we had two different “search by country” and, initially, we used the same form name (name=”country_search”), leading the app to always search the name of the first country instead of our intended search in the second search box. To solve this issue we attributed different names to each search box’s form name: country_search_1 and country_search_2.
- ➔ The endpoint sites/search/<int:year> was initially sites/<int:year> leading to confusion with sites/<int:id> since they have the same variable type (when clicking year on a site page, it assumed it as Id_no)
- ➔ The names of the sites, the description and the justification all had HTML tags. To solve this issue we simply added “| safe” to the attribute access.

Conclusion

The development of this assignment allowed us to solidify and deepen technical knowledge previously studied through the exploration of the database, drawing of the diagrams that allow us to understand the relationship between the different aspects of the database, the thoughtful and logical reshaping of data and the selection of interesting and challenging SQL queries.

A notable new element to us was the development of an interface in Python using the Flask framework. Combining it with SQL we created a more refined way to interact with the database easily and efficiently.

This enriching and challenging assignment required a detail-attentive and structured approach contributing to our ‘soft skills’ and ‘hard skills’, including programming and problem-solving.

Ultimately, the obtained result reflects the commitment and learning across the entire project, providing a practical and relevant experience. The completed work left a feeling of satisfaction, as much for the achieved objectives as for the opportunity to apply the gained knowledge in a real-life context.

References

Class Documentation: Material provided through the semester, used as a basis for the development of the project and the definition of the structures and features of the app as well as the Movie App given.
https://github.com/jresende/MoviesApp.

Official Flask Documentation: Consulted to implement routes, template rendering and assure the utilization of good practices in the development of the app.
https://flask.palletsprojects.com

UNESCO information: Used to better understand the structure of the data and validate the presentation of information relative to the World Heritage Sites.
https://whc.unesco.org

These sources served as a fundamental support during this project's development, guiding our technical decisions and assuring the quality and functionality of the work done.

Git-Hub Repository: contains all the files including all code, the database, source material and additional info.
https://github.com/simao-g/WorldHeritageApp