Final Team Project —— Group 31 : 110550118 林佑家

Final project document format:

①. An introduction of your application, including why you want to develop the application and the main functions of your application.

「菇菇栽培」這款遊戲雖然說不算主流，但卻是一個眾所皆知的遊戲，然而，搜尋網站或是 app 卻很少有關於菇菇栽培的圖鑑資訊，只有一些遊戲玩家的攻略新德里會出現相關的資訊，但也欠缺圖鑑的基本元素(搜尋模式、篩選等等)，所以才決定實作一個關於菇菇栽培的圖鑑。

而菇菇栽培有許多不同的版本，而這個 project 是以「菇菇栽培研究室 作伙來同樂」這個版本的菇菇栽培遊戲為基底實作的一個遊戲圖鑑。

2. Database design - describe the schema of all your tables in the database, including keys and index, if applicable (why you need the keys or why you think adding an index is or is not helpful).

①. The schema of all my tables in the database:

這個 project 的 database 只有 1 個名為 mushroom_dictionary 的 table：

mushroom_dictionary(theme_id, theme_name, id, name, rare_state, rarity, NP)

- theme_id：栽培主題對應的編號。
- theme_name：每一種菇菇所屬的栽培主題之名稱。
- id：每一種菇菇在其栽培主題中的編號。
- name：每一種菇菇的名稱。
- rare_state：

- 相同名稱的菇菇可能有 2 種以上的狀態：普通、稀有。(不是全部的菇菇都有稀有狀態)
- 普通狀態的菇菇為-1
- 稀有狀態的菇菇為 1、2、……，取決於稀有狀態種類的數量，以此類推。

- rarity：每一種菇菇會有相對應的稀有度(1~5 星)。
- NP：每一種菇菇的賣價。
- primary key(name, rare_state)

②. 因為 database 的資料量沒有很多，而且每一種菇菇都有相對應的編號(栽培主題編號(theme_id)+菇菇在其栽培主題中的編號(id))，所以使用 index 沒有太大的效用。

3. Database design - describe the normal form of all your tables. If the tables are not in BCNF, please include the reason (performance trade-off, etc.).

mushroom_dictionary :

- 1NF(沒有 redundancy data 但有 partial dependency)
- 因為 schema 的 attribute 沒有很多，但如果要 normalization 會生成很多 table，要 query data 就必須 join 多個 table，而且 non-normalized table 比較清楚明確每一筆資料對應的 value。況且，這個 database 的資料並沒有很多，即使 non-normalization 產生 extra space 和 execution，對整個 performance 也沒有太大的影響，所以沒有對 mushroom_dictionary 進行 normalization。

  (偏好：faster lookup > improve performance)

4. From the data sources to the database - describe the data source and the original format.

①. The data source to the database： 巴哈姆特(網址如下)

“https://forum.gamer.com.tw/C.php?bsn=76116&snA=27”的 2 樓：圖鑑區

②. The original format:

資料原本的格式是巴哈姆特 2 樓的 195 張圖照片，每一張照片包含每一筆資料的內容。

5. From the data sources to the database - describe the methods of importing the original data to your database and strategies for updating the data, if you have one.

①. The method of importing the original data to my database:

先用 AWS 創建一個 RDS 的 database，再用 pgadmin4 在這個 database 裡創建名為 mushroom_dictionary 之 table 的 schema，最後右鍵點選創建的 table，點選“Import/Export Data…”，再把所有要 import 的.csv 檔 import 進去。

至於 csv 檔的來源是將原本 195 張照片裡的內容，手動輸入資料到 excel 檔，最後再轉成 csv 檔的格式。在這邊我對每一個栽培主題個創建一個包含該栽培主題之所有菇菇的 csv 檔，並且每一個栽培主題對應的 csv 的 schema 都是相同的，所以總共產生 8 個 csv 檔。列表如下：

- 菇菇栽培圖鑑_1_菇菇研究所.csv
- 菇菇栽培圖鑑_2_獨享燒肉.csv
- 菇菇栽培圖鑑_3_水族箱.csv
- 菇菇栽培圖鑑_4_魔女工坊.csv
- 菇菇栽培圖鑑_5_娃娃屋.csv
- 菇菇栽培圖鑑_6_星象儀.csv
- 菇菇栽培圖鑑_7_菇菇農場.csv
- 菇菇栽培圖鑑_8_賞花.csv

②. The strategy for updating the data:

點選創建的 table，點選“Query Tool”，再用 SQL 相關語法執行對 data 的 update。

6. Application with database - explain why your application needs a database.

這個 project 的圖鑑的搜尋功能，會需要從資料庫中去選擇符合條件的菇菇來執行搜尋的動作。

7. Application with database - includes the queries that are performed by your application, how your application performed these queries (connections between application and database), and what the cooperating functions for your application.

在這個圖鑑中，總共有 5 種搜尋的模式，分別對應到 main.py 的中的 5 個 function，這 5 個 function 分別應用到不同的 queries。

①. 輸入名稱搜尋：search_muchrooms_by_name()

- select * from mushroom_dictionary where name like ‘□’ order by theme_id, id, rare_state asc：main.py 第 124 行
- □是輸入名稱欄的文字

②. 輸入栽培主題搜尋：search_mushrooms_by_theme()

- select * from mushroom_dictionary where theme_name like ‘□’ order by theme_id, id, rare_state asc：main.py 第 177 行
- □是輸入栽培主題欄的文字

③. 選擇稀有度搜尋：search_mushrooms_by_rarity()

- select * from mushroom_dictionary where rarity = ‘□’ order by theme_id, id, rare_state asc：main.py 第 247 行
- □是稀有度選擇欄所選擇之稀有度的數字

④. 選擇賣價範圍搜尋：search_mushrooms_by_price()

- select * from mushroom_dictionary where NP >= ‘□’ and NP <= ‘□’ order by NP, theme_id, id, rare_state asc：main.py 第 302 行
  - 第 1 個□是輸入 NP 欄的下限數字
  - 第 2 個□是輸入 NP 欄的上限數字
- select * from mushroom_dictionary where NP = -1 order by NP, theme_id, id, rare_state asc：main.py 第 304 行
  - 處理未輸入的情況⇒無搜尋結果

⑤. 列出所有菇菇：search_all_mushrooms ()

- select * from mushroom_dictionary order by theme_id, id, rare_state asc：main.py 第 74 行

至於如何執行這些 SQL 是在 main.py 的這 5 個 function 創建遊標 cur 後，再將這些 SQL 放入 cur.execute(_____)中(底線填入想要執行的 SQL)，就可以執行 query。

8. All the other details of your application that you want us to know.

①. 執行 main.py，就會出現菇菇栽培圖鑑 UI。

②. 輸入名稱後，按下名稱對應的搜尋，便可列出名稱符合關鍵字的菇菇。

③. 輸入栽培主題名稱後，按下栽培主題對應的搜尋，便可列出栽培主題符合關鍵字的菇菇。

④. 在稀有度選單選擇稀有度，按下對應的搜尋，便可列出符合條件的菇菇。

⑤. 輸入賣價範圍後，按下對應的搜尋，便可列出符合條件的菇菇。

⑥. 按下「所有」，會列出所有菇菇的資訊。

⑦. 按下「搜尋」或「所有」，符合條件的菇菇會列表出現在中間的白色長方形區域。

⑧. 名稱為「菇菇」和「枯哭菇菇」，因為這 2 種菇菇對應所有的栽培主

題，所以在 mushroom_dictionary table 中 theme_name 的欄位會顯示「all」，只要在栽培主題中關鍵字搜尋有搜尋到栽培主題，便會列出「菇菇」和「枯哭菇菇」這兩筆資料。