Homework 7: Databases

In this homework, you will be populating a database table with information about Pokemon and then writing code to fetch data from that table. You will need the starter code, called HW7.py, and the pokemon file, called pokemon.txt.

In case you aren't familiar: Pokemon is a video game series from Nintendo with a corresponding card game, television show, and several movies. Pokemon (short for pocket monsters) are creatures with special abilities that can be caught and taught to battle by Pokemon Trainers. Each pokemon has a set of unique features, called stats, that affect their fighting ability, including hit points (HP), speed, attack power, and defense power. Many Pokemon can evolve into new forms from gaining battle experience. Every pokemon has a type (such as water, fire, or grass – and some have more than one) that defines what type of abilities they can use.

We have provided the code for the following:

1. To read the cache data (***read_data()*** function)
2. To create the database and set up the connection and cursor (***open_database()*** function)
3. To set up one of the tables, called Types, in the database (***make_types_table()*** function)

We have also provided test cases that will pass if the functions are written correctly. **You should not edit these test cases.** NOTE: It is okay for the extra credit test case to fail if you do not attempt the extra credit (***test_pokemon_of_type()***).

When done with the assignment, your database will have two tables. You should start by running the starter code and looking at the structure of the Types table in the DB Browser.

Tasks

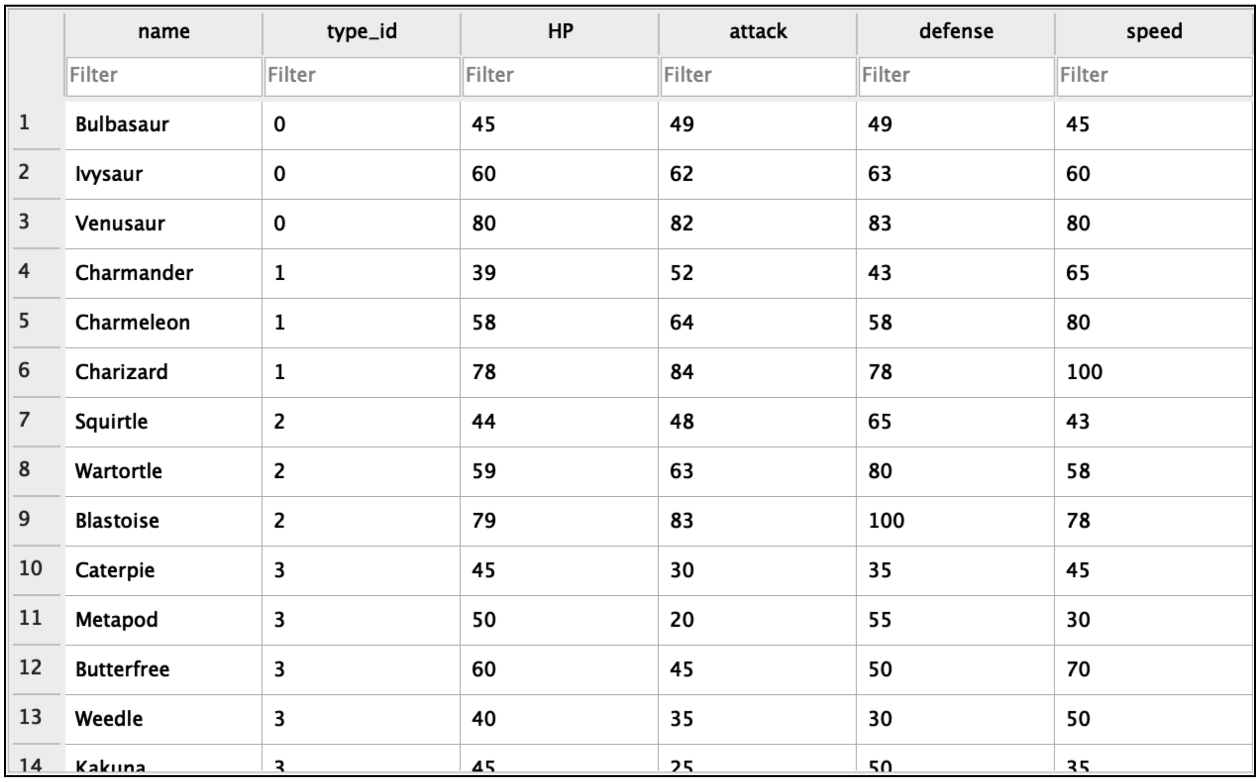
1. ***make_pokemon_table()***

   This function takes 3 arguments: JSON data, the database cursor, and the database connection object. It iterates through the JSON data to get a list of pokemon and loads all of the pokemon into a database table called 'Pokemon' with the following columns:

   a. name (datatype: text; Primary key)
   b. type_id (datatype: integer)
   c. HP (datatype: integer)
   d. attack (datatype: integer)
   e. defense (datatype: integer)
   f. speed (datatype: integer)

To find the type_id for each pokemon, you will have to look up the first type of each pokemon in the types table we create for you -- see make_types_table above for details.

This is what your Pokemon table should look like when viewed in DB Browser:



	name	type_id	HP	attack	defense	speed
	Filter	Filter	Filter	Filter	Filter	Filter
1	Bulbasaur	0	45	49	49	45
2	Ivysaur	0	60	62	63	60
3	Venusaur	0	80	82	83	80
4	Charmander	1	39	52	43	65
5	Charmeleon	1	58	64	58	80
6	Charizard	1	78	84	78	100
7	Squirtle	2	44	48	65	43
8	Wartortle	2	59	63	80	58
9	Blastoise	2	79	83	100	78
10	Caterpie	3	45	30	35	45
11	Metapod	3	50	20	55	30
12	Butterfree	3	60	45	50	70
13	Weedle	3	40	35	30	50
14	Kakuna	3	45	25	50	35

2. ***hp_search()***

   This function takes 3 arguments as input: an HP integer, the database cursor, and the database connection object. It selects all the pokemon of a particular HP and returns a list of tuples each containing the pokemon's name, type_id, and HP.

   Expected output for pokemon with HP of "50":

   [('Cloyster', 2, 50), ('Cubone', 7, 50), ('Hitmonchan', 12, 50), ('Hitmonlee', 12, 50), ('Magneton', 6, 50), ('Metapod', 3, 50), ('Sandshrew', 7, 50), ('Trubbish', 5, 50)]

3. ***hp_speed_attack_search()***
   This function takes 5 arguments as input: an HP integer, a speed integer, an attack integer, the database cursor, and the database connection object. It selects all the pokemon at the HP passed to the function that have a speed **greater than** the speed rating passed to the function and at an attack **greater than** the attack rating passed to the function. This function returns a list of tuples each containing the pokemon's name, speed, attack, and defense.

Expected Output for pokemon with HP = 60, speed > 20, and attack > 85:

```
[('Arbok', 80, 95, 69), ('Raichu', 110, 90, 55), ('Parasect', 30, 95, 80),
('Dodrio', 110, 110, 70), ('Zoroark', 105, 105, 60)]
```

4. ***type_speed_attack_search()***

   This function takes 5 arguments as input: a type string, a defense integer, a speed integer, the database cursor, and the database connection object. It selects all pokemon of the passed type that have speed **greater than** the passed speed and **greater than** the passed defense. It returns a list of tuples, each containing the pokemon name, type, speed, and defense.

   HINT: You'll have to use JOIN for this task.

   Expected Output for speed > 50, defense > 60, and type “Grass”:

   ```
   [('Exeggutor', 'Grass', 55, 85), ('Ivysaur', 'Grass', 60, 63),
   ('Tangela', 'Grass', 60, 115), ('Venusaur', 'Grass', 80, 83)]
   ```

5. **Extra Credit**

   You’ll make 2 new functions, ***make_bilingual_table()*** and ***bilingual_type_search()***, and then write at least 2 meaningful test cases for each of them.

   The first function takes 3 arguments: JSON data, the database cursor, and the database connection object. It iterates through the JSON data to get a list of pokemon and loads all of the pokemon into a database table called ‘Bilingual_Pokemon' with the following columns:

   a. english_name (datatype: text; Primary key)
   b. french_name
   c. type_id (datatype: integer)
   g. hp

   To find the type_id for each pokemon, you will have to look up the first type of each pokemon in the types table we create for you -- see make_types_table above for details.

   The second function takes in a type string, the database cursor, and the database connection object. It selects all the pokemon of the given type that have the same name in French and English. It returns a list of tuples, each with a pokemon’s name, type, and hp.

Grading Rubric

1. ***make_pokemon_table()*** - 25 points
    a. 5 points for making a table
    b. 5 points for adding all attributes correctly
    c. 15 points for filling table with pokemon

2. ***hp_search()*** - 10 points

3. ***hp_speed_attack_search()*** - 10 points

4. ***type_speed_defense_search()*** - 15 points

5. **Extra credit** - 6 bonus points

Submission

Make at least 4 git commits and turn in your GitHub repo URL on Canvas by the due date to receive credit.