Fall 2021 Quarter
Graduate Project Final Write-Up

Submitted By:
Name : 1. Chinmay Tawde
2. Varun Jaisundar Raju

Pokémon Database

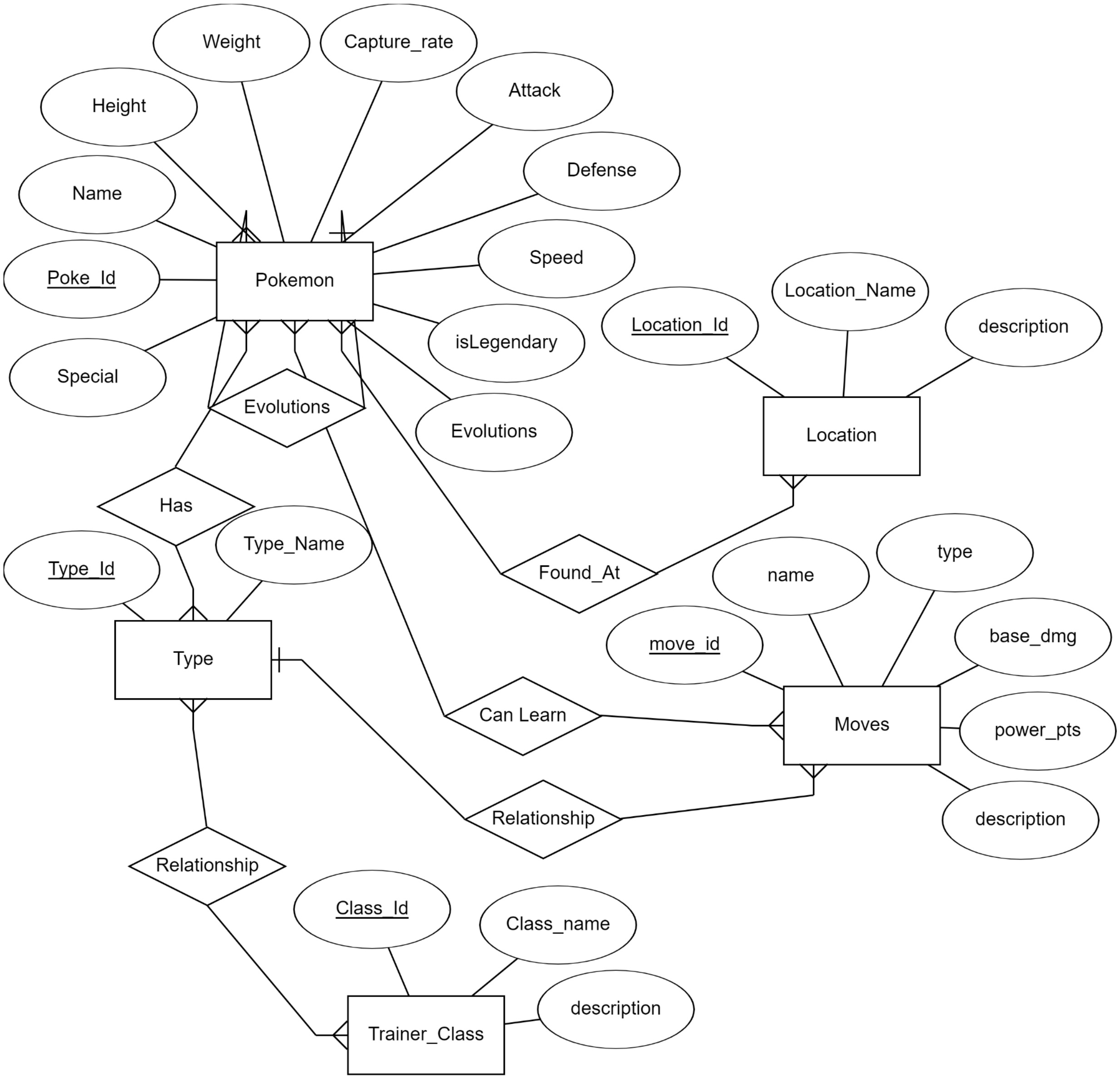
Outline

The database idea is inspired by the 90's popular cartoon TV show Pokemon (short for "Pocket Monsters") created by Nintendo. In this imaginative world, there are two main entities, Number one are the pokemons which are quirky monsters found throughout the world. They belong to a particular type (water, grass, fire, etc) and have their own unique skill set and statistics. The other entity are the trainers who travel to different places and try to capture these pokemon and put them in battle against other trainer's pokemon. This helps them to gain monetary benefits as well as gym badges that help them to become the very best pokemon trainer in the world.

The database would be a replica of an item shown in the TV show called "Pokédex" which stores all the information about the pokemon, their types, moves and locations at which they are seen. As the TV show has become severely popular and has many different seasons, many new pokemons have been created.

But for this project, we will be focusing on the original 151 pokemons which are found in the Kanto region and the database would revolve around the Generation I games (Red/Blue/Yellow) based on these 151 pokemons. (Some table examples include Pokemon, Trainer, moves, types, pokemon_location, pokemon_evolution_chain, etc)

ER Diagram


Weight
Capture_rate
Height
Attack
Defense
Name
Speed
Poke_Id
Pokemon
Location_Name
Location_Id
description
isLegendary
Special
Evolutions
Evolutions
Location
Has
Type_Name
Found_At
type
Type_Id
name
base_dmg
move_id
Type
Can Learn
Moves
power_pts
description
Relationship
Relationship
Class_Id
Class_name
description
Trainer_Class

CREATE Table Statements

There are total 10 tables in the Pokemon Database, those are :

- **Pokemon -** The main table that contains details of all the Pokemons
- **Type -** The domain table that stores details regarding different types that exist in the Pokemon world.
- **Pokemon_Types -** This table provides information about the type that each Pokemon belongs to. Ex. Bulbasaur is a grass type as well as poison type Pokemon
- **Pokemon_Evolutions -** This table stores information about Pokemon's prevolution
- **Trainer_Class -** This table stores information about different Trainer AI's present in the game.
- **Trainer_PkmnTypes -** This table stores information about different Pokemons that each Trainer has.
- **Location -** The table stores information about different locations present in the Pokemon world.
- **Pokemon_Location -** This table contains information about the whereabouts of each Pokemon throughout the world.
- **Moves -** This table contains information about different Pokemon moves.
- **Pokemon_moves** - This table contains information about moves that each Pokemon posessess.

```
CREATE Table Pokemon (
      Poke_id                 INT,
      Name                    TEXT NOT NULL,
      Height                  DECIMAL,
      Weight                  DECIMAL,
      Capture_rate            INT,
      HP                      INT,
      Attack                  INT,
      Defense                 INT,
      Special                 INT,
      Speed                   INT,
      Evolutions              INT,
      isLegendary             INT,
    PRIMARY KEY(Poke_id)
);


CREATE Table Type (
      Type_id           SERIAL PRIMARY KEY,
      Type_name         VARCHAR(20)
);

CREATE Table Pokemon_Types (
      pokemon_id              INT,
      type_id                 INT,
      CONSTRAINT fk_pokemon FOREIGN KEY(pokemon_id) REFERENCES
Pokemon(poke_id),
      CONSTRAINT fk_type FOREIGN KEY(type_id) REFERENCES Type(Type_id)
);

CREATE Table Location (
      Location_id             SERIAL PRIMARY KEY,
      location_name           VARCHAR(100),
      description             TEXT
);

CREATE Table Pokemon_location (
      pokemon_id              INT,
      location_id             INT
      CONSTRAINT fk_pokemon FOREIGN KEY(pokemon_id) REFERENCES
Pokemon(poke_id),
      CONSTRAINT fk_location FOREIGN KEY(location_id) REFERENCES
Location(Location_id),
);


CREATE Table Moves (
      move_id                       SERIAL PRIMARY KEY,
      name                          VARCHAR(50),
      type                          INT,
      base_dmg                      INT,
      power_pts                     INT,
      description                   TEXT,
      CONSTRAINT fk_move_type
    FOREIGN KEY(type)
      REFERENCES Type(type_id)
);

CREATE Table Pokemon_moves (
      pokemon_id                    INT,
      move_id                       INT,
      CONSTRAINT fk_pokemon FOREIGN KEY(pokemon_id) REFERENCES
Pokemon(poke_id),
      CONSTRAINT fk_move FOREIGN KEY(move_id) REFERENCES Moves(move_id)
);

CREATE Table Pokemon_Evolutions (
      pokemon_id                  INT,
      pre_evolution_id            INT,
      isFinalEvolution            BOOL,
      CONSTRAINT fk_poke_pre_evol FOREIGN KEY(pre_evolution_id) REFERENCES
Pokemon(Poke_id),
      CONSTRAINT fk_poke_id FOREIGN KEY(pokemon_id) REFERENCES
Pokemon(Poke_id)
);
```

All the queries related to table created can be found in the github repository by the file named create_table.sql

How were the tables Populated?

The tables were populated using the following sources:
We referred various websites such as
https://pokemondb.net/pokedex/game/red-blue-yellow
https://www.serebii.net/pokemon/gen1pokemon.shtml

Also relied on datasets (csv's) from Kaggle
https://www.kaggle.com/dizzypanda/gen-1-pokemon
https://www.kaggle.com/mariotormo/complete-pokemon-dataset-updated-090420

Using these sources csv files were created which are present in the repository and imported using the command :

COPY <Table_Name> FROM <FILE_LOCATION_PATH> DELIMITER ‘,’ CSV HEADER;

Also some were directly imported using the pgAdmin GUI interface.
Some of the tables were directly populated using regular insert queries such as:

```
INSERT INTO Type (type_name) VALUES ('Bug');
```

Whereas, other complex tables containing foreign key relations were populated by creating a stored procedure and using that procedure to populate them.
Example,

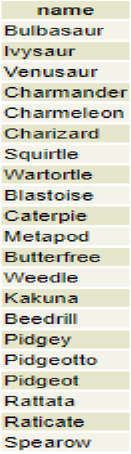
```
CREATE OR REPLACE PROCEDURE populate_pokemon_location(poke_Name text,
loc_Name text)
LANGUAGE SQL
AS $$
INSERT INTO pokemon_location (pokemon_id, location_id) VALUES ((SELECT
poke_id FROM pokemon WHERE name =
poke_Name), (SELECT location_id FROM location WHERE location_name =
loc_Name));
$$;


CALL populate_pokemon_location('Caterpie','Route 25');
CALL populate_pokemon_location('Caterpie','Viridian Forest');
CALL populate_pokemon_location('Metapod','Route 24');
```

All the insert related queries can be found in the github repository by the file named insert_queries.sql

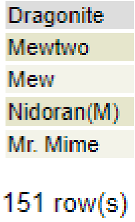
20 English Questions and their queries.

1. List all the pokemons available in the database

```
SELECT name FROM pokemon;
```



name
Bulbasaur
Ivysaur
Venusaur
Charmander
Charmeleon
Charizard
Squirtle
Wartortle
Blastoise
Caterpie
Metapod
Butterfree
Weedle
Kakuna
Beedrill
Pidgey
Pidgeotto
Pidgeot
Rattata
Raticate
Spearow



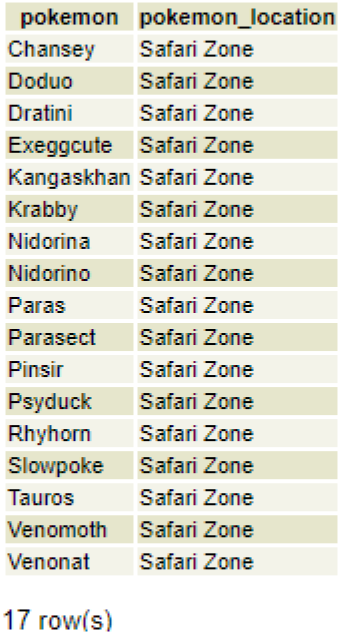
Dragonite
Mewtwo
Mew
Nidoran(M)
Mr. Mime

151 row(s)

2. List all the pokemon’s those are found in the particular location

Ex location=’Safari Zone’

```
select name as Pokemon,location_name as Pokemon_Location
from pokemon,location,pokemon_location where
pokemon.poke_id=pokemon_location.pokemon_id and
location.location_id=pokemon_location.location_id and
location.location_name='Safari Zone'  group by
pokemon.name,location.location_name;
```

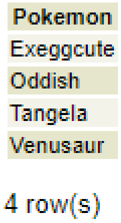


pokemon	pokemon_location
Chansey	Safari Zone
Doduo	Safari Zone
Dratini	Safari Zone
Exeggcute	Safari Zone
Kangaskhan	Safari Zone
Krabby	Safari Zone
Nidorina	Safari Zone
Nidorino	Safari Zone
Paras	Safari Zone
Parasect	Safari Zone
Pinsir	Safari Zone
Psyduck	Safari Zone
Rhyhorn	Safari Zone
Slowpoke	Safari Zone
Tauros	Safari Zone
Venomoth	Safari Zone
Venonat	Safari Zone

17 row(s)

3 List all the pokemon that know a particular move Ex move='Absorb'

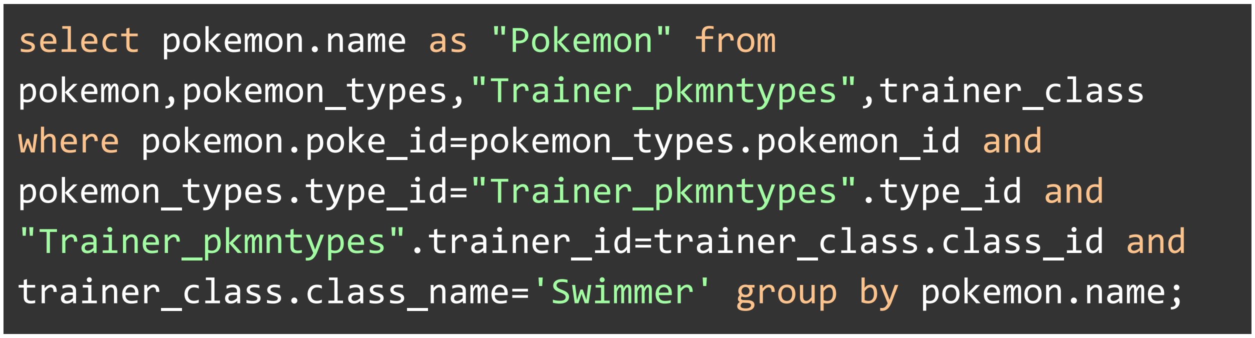
```
select pokemon.name as "Pokemon" from
pokemon,moves,pokemon_moves where
pokemon.poke_id=pokemon_moves.pokemon_id and
moves.move_id=pokemon_moves.move_id and moves.name='Absorb'
group by pokemon.name;
```



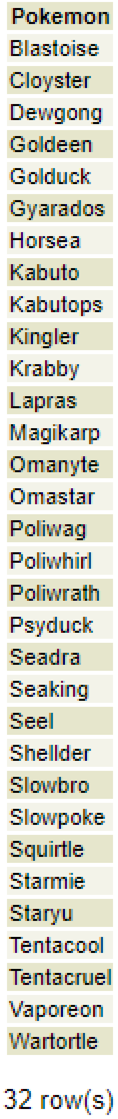
Pokemon
Exeggcute
Oddish
Tangela
Venusaur

4 row(s)

4. List all the pokemons those are captured by a specific trainer
Ex trainer_class=’Swimmer’



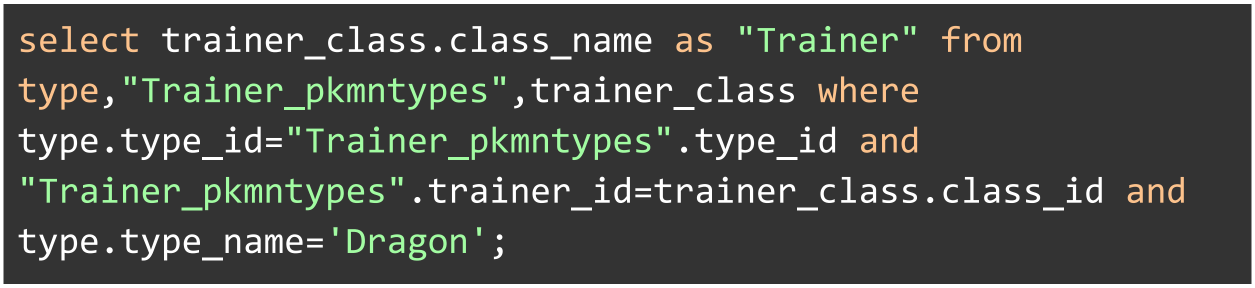
```
select pokemon.name as "Pokemon" from
pokemon,pokemon_types,"Trainer_pkmntypes",trainer_class
where pokemon.poke_id=pokemon_types.pokemon_id and
pokemon_types.type_id="Trainer_pkmntypes".type_id and
"Trainer_pkmntypes".trainer_id=trainer_class.class_id and
trainer_class.class_name='Swimmer' group by pokemon.name;
```



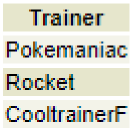
Pokemon
Blastoise
Cloyster
Dewgong
Goldeen
Golduck
Gyarados
Horsea
Kabuto
Kabutops
Kingler
Krabby
Lapras
Magikarp
Omanyte
Omastar
Poliwag
Poliwhirl
Poliwrath
Psyduck
Seadra
Seaking
Seel
Shellder
Slowbro
Slowpoke
Squirtle
Starmie
Staryu
Tentacool
Tentacruel
Vaporeon
Wartortle

32 row(s)

5. List all the trainers who belong to the particular type
Example type='Dragon'



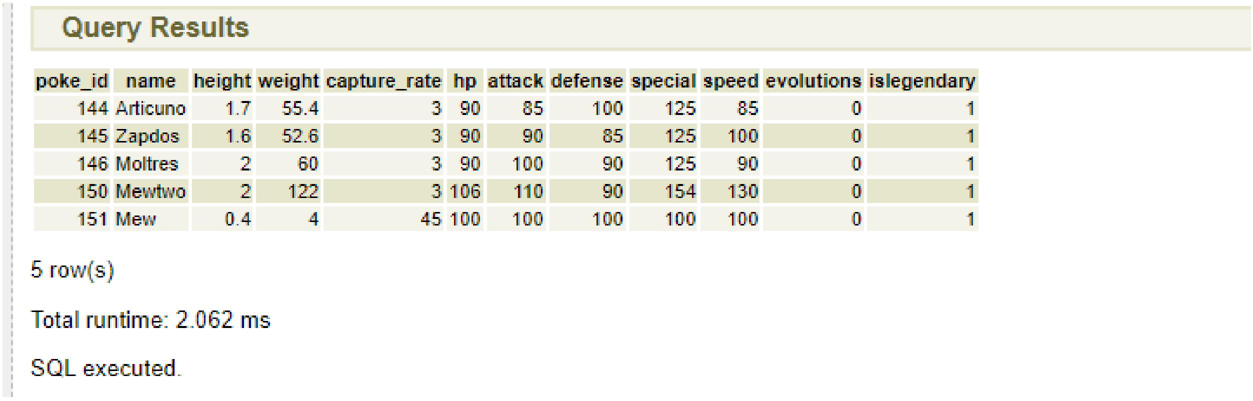
```
select trainer_class.class_name as "Trainer" from
type,"Trainer_pkmntypes",trainer_class where
type.type_id="Trainer_pkmntypes".type_id and
"Trainer_pkmntypes".trainer_id=trainer_class.class_id and
type.type_name='Dragon';
```



Trainer
Pokemaniac
Rocket
CooltrainerF

6. List all the legendary pokemon present in the pokemon world.
(Changed from List all the trainers those have the pokemon's that belong to a specific type(ex fire,grass etc) because it was similar to previous question)

SELECT * FROM pokemon WHERE isLegendary = 1;



Query Results

poke_id	name	height	weight	capture_rate	hp	attack	defense	special	speed	evolutions	islegendary
144	Articuno	1.7	55.4	3	90	85	100	125	85	0	1
145	Zapdos	1.6	52.6	3	90	90	85	125	100	0	1
146	Moltres	2	60	3	90	100	90	125	90	0	1
150	Mewtwo	2	122	3	106	110	90	154	130	0	1
151	Mew	0.4	4	45	100	100	100	100	100	0	1

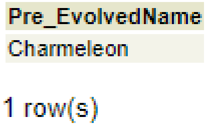
5 row(s)

Total runtime: 2.062 ms

SQL executed.

7. Find the predecessor name of a given pokemon
Example : Charizard

```
select pokemon.name as "Pre_EvolvedName" from pokemon where
pokemon.poke_id in (select pre_evolution_id from
pokemon_evolutions where pokemon_id in (select
pokemon.poke_id from pokemon where
pokemon.name='Charizard'));
```

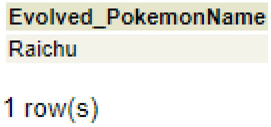


Pre_EvolvedName
Charmeleon

1 row(s)

8. Find the evolution name of a given pokemon
Ex: Pikachu

```
select pokemon.name as "Evolved_PokemonName" from pokemon
where pokemon.poke_id in (select pokemon_id from
pokemon_evolutions where pre_evolution_id in (select
pokemon.poke_id from pokemon where
pokemon.name='Pikachu'));
```

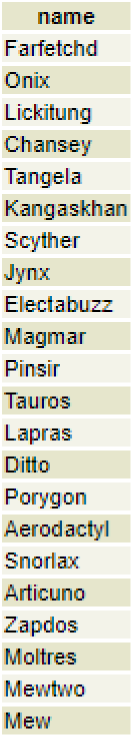


Evolved_PokemonName
Raichu

1 row(s)

9. Find the pokemon that does not have any evolution.

```
select pokemon.name from pokemon,pokemon_evolutions where
pokemon.poke_id=pokemon_evolutions.pokemon_id and
pokemon_evolutions.pre_evolution_id is NULL and
pokemon_evolutions.isfinalevolution=true;
```

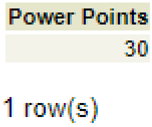


name
Farfetchd
Onix
Lickitung
Chansey
Tangela
Kangaskhan
Scyther
Jynx
Electabuzz
Magmar
Pinsir
Tauros
Lapras
Ditto
Porygon
Aerodactyl
Snorlax
Articuno
Zapdos
Moltres
Mewtwo
Mew

10. Find the power_points of the given move

Ex Move name=’Cut’

```
select power_pts as "Power Points" from moves where
name='Cut';
```

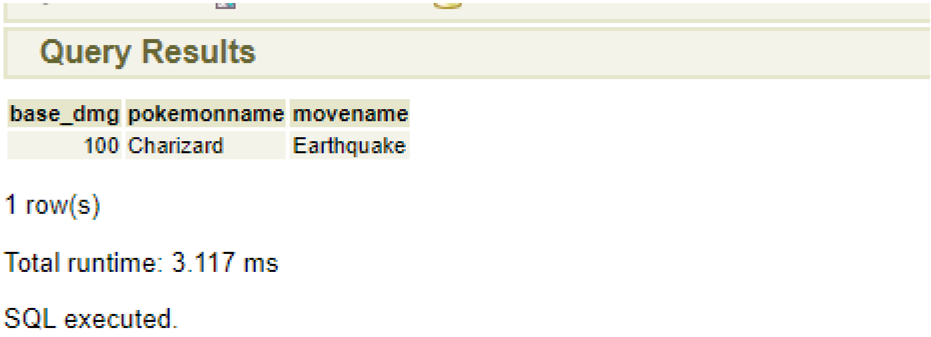


Power Points
30

1 row(s)

11. Find the base_damage of a given pokemon that has a specific move
Example pokemon=’Charizard’ move=’Earthquake’

```
SELECT base_dmg, pokemon.name AS pokemonName, moves.name AS
moveName FROM moves JOIN pokemon_moves ON (moves.move_id =
pokemon_moves.move_id AND moves.name = 'Earthquake') JOIN
pokemon ON (pokemon.poke_id = pokemon_moves.pokemon_id AND
pokemon.name = 'Charizard');
```



Query Results

base_dmg	pokemonname	movename
100	Charizard	Earthquake

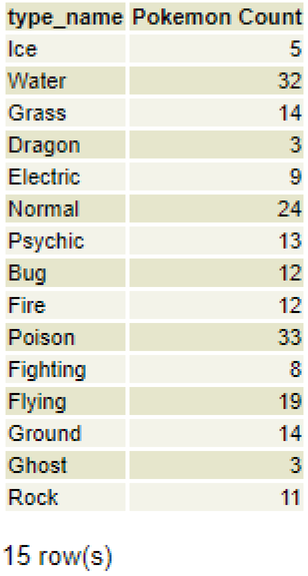
1 row(s)

Total runtime: 3.117 ms

SQL executed.

12. Find the total number of pokemon that belong to each type.

```
select type.type_name,count(pokemon.poke_id) as "Pokemon
Count" from type,pokemon,pokemon_types where
pokemon.poke_id=pokemon_types.pokemon_id and
type.type_id=pokemon_types.type_id group by type.type_name;
```

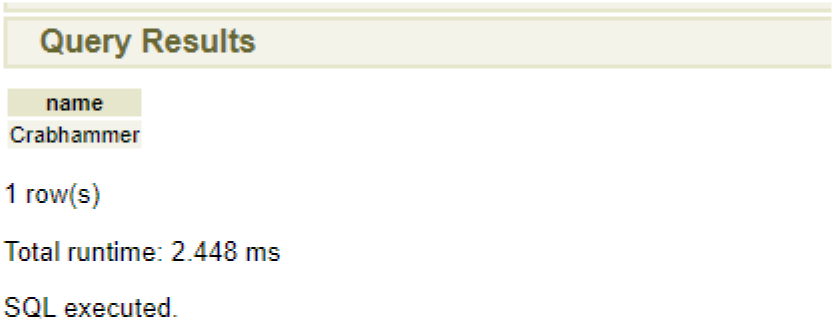


type_name	Pokemon Count
Ice	5
Water	32
Grass	14
Dragon	3
Electric	9
Normal	24
Psychic	13
Bug	12
Fire	12
Poison	33
Fighting	8
Flying	19
Ground	14
Ghost	3
Rock	11

15 row(s)

13. Find the move that most pokemon know.

```
SELECT name FROM moves JOIN (SELECT move_id, COUNT(move_id)
AS moveCount FROM pokemon_moves GROUP BY move_id ORDER by
moveCount DESC LIMIT 1) sk ON (sk.move_id = moves.move_id);
```



Query Results

name
Crabhammer

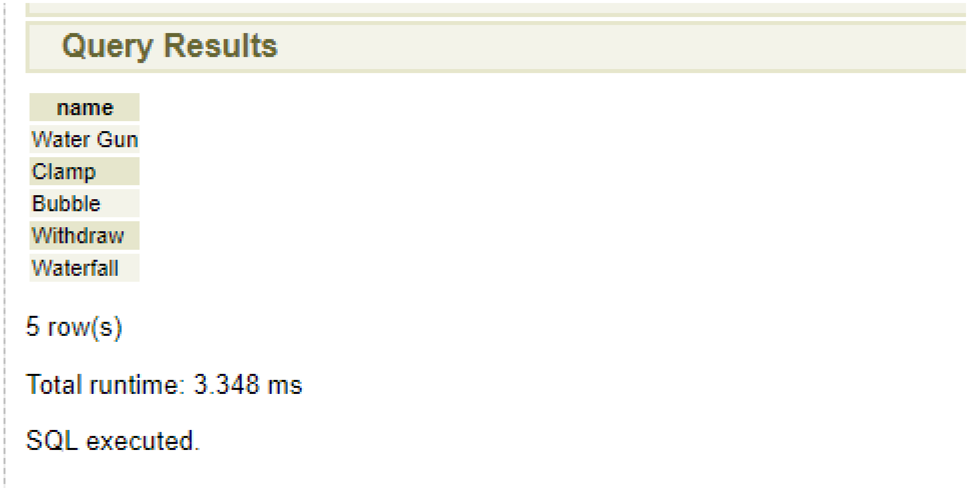
1 row(s)

Total runtime: 2.448 ms

SQL executed.

14. Find all moves of the pokemon that belong to a given type (Ex. Blastoise is a pokemon that belongs to Water Type)

```
SELECT moves.name FROM type JOIN pokemon_types ON
(type.type_name = 'Water' AND type.type_id =
pokemon_types.type_id) JOIN pokemon ON (pokemon.poke_id =
pokemon_types.pokemon_id) JOIN pokemon_moves ON
(pokemon.poke_id = pokemon_moves.pokemon_id) JOIN moves ON
(pokemon_moves.move_id = moves.move_id);
```



Query Results

name
Water Gun
Clamp
Bubble
Withdraw
Waterfall

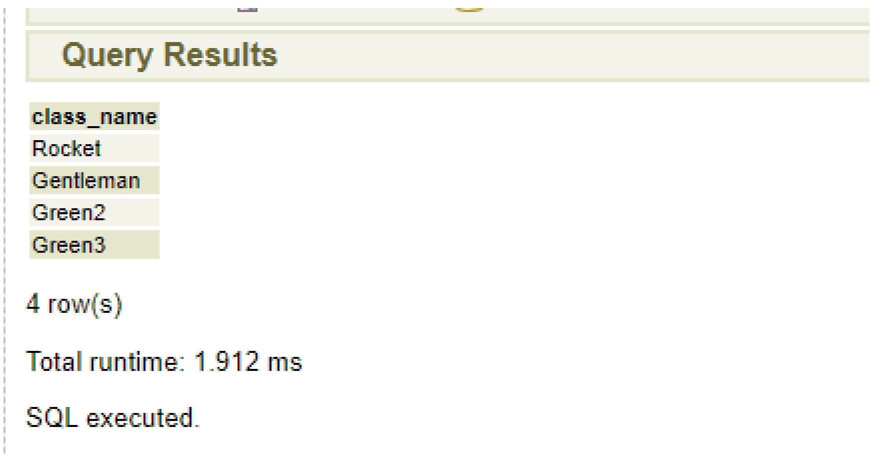
5 row(s)

Total runtime: 3.348 ms

SQL executed.

15. List all the trainers who have captured more than 1 type of pokemon

```
SELECT class_name FROM trainer_class WHERE class_id IN
(SELECT trainer_id FROM trainer_pkmntypes  GROUP BY
trainer_id HAVING COUNT(*) > 1)
```



Query Results

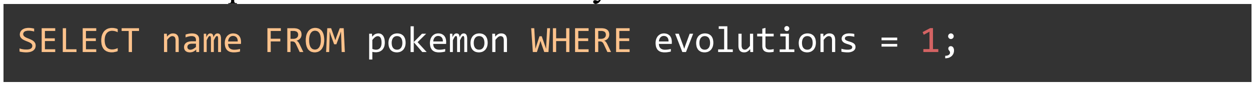
class_name
Rocket
Gentleman
Green2
Green3

4 row(s)

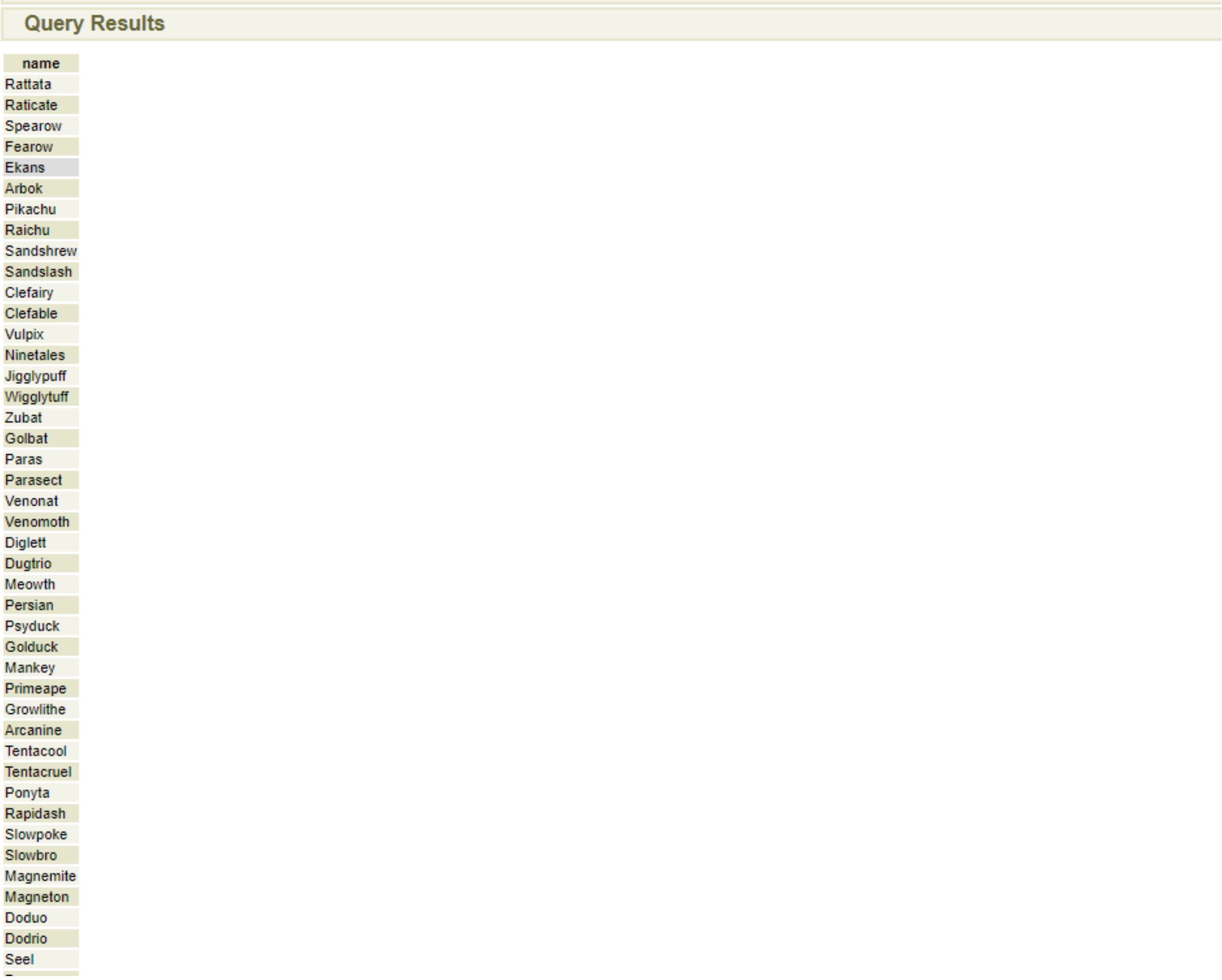
Total runtime: 1.912 ms

SQL executed.

16. List all the pokemon those have only one evolved form



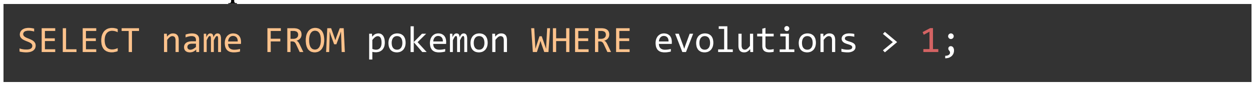
```
SELECT name FROM pokemon WHERE evolutions = 1;
```



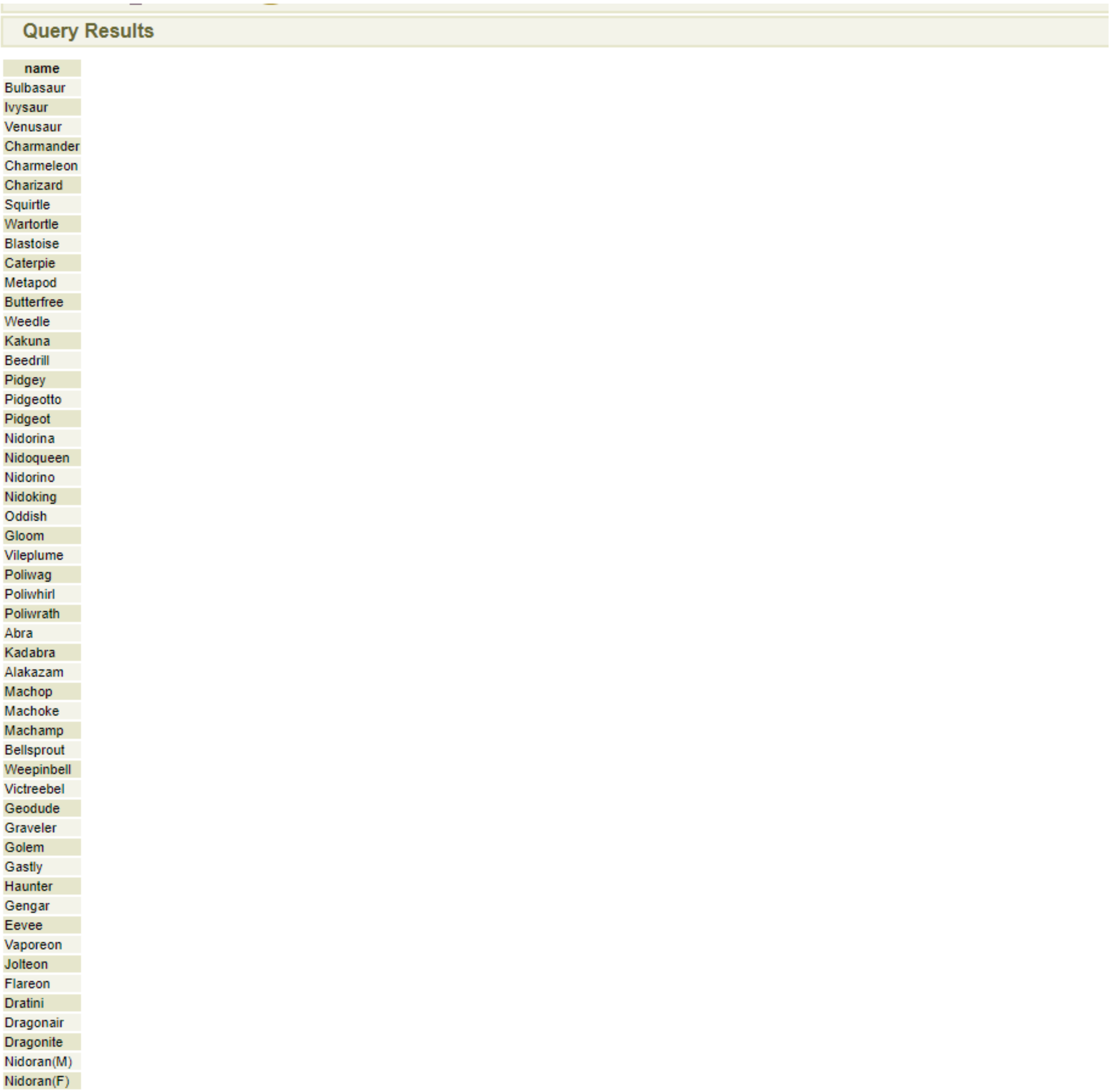
Query Results

name
Rattata
Raticate
Spearow
Fearow
Ekans
Arbok
Pikachu
Raichu
Sandshrew
Sandslash
Clefairy
Clefable
Vulpix
Ninetales
Jigglypuff
Wigglytuff
Zubat
Golbat
Paras
Parasect
Venonat
Venomoth
Diglett
Dugtrio
Meowth
Persian
Psyduck
Golduck
Mankey
Primeape
Growlithe
Arcanine
Tentacool
Tentacruel
Ponyta
Rapidash
Slowpoke
Slowbro
Magnemite
Magneton
Doduo
Dodrio
Seel

17. List all the pokemon that have more than one evolved form



```
SELECT name FROM pokemon WHERE evolutions > 1;
```

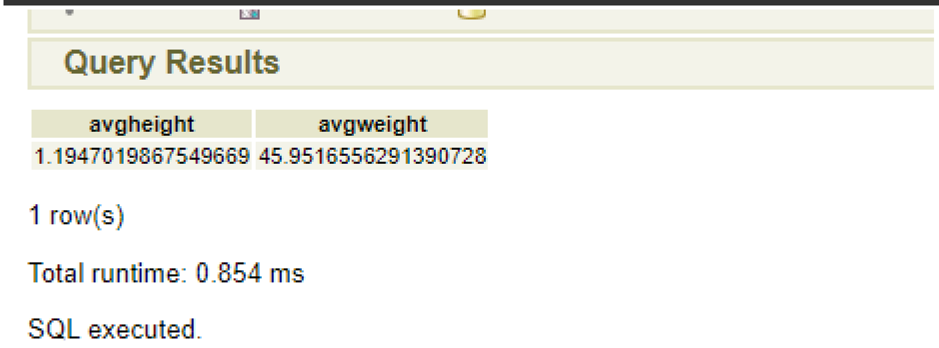


Query Results

name
Bulbasaur
Ivysaur
Venusaur
Charmander
Charmeleon
Charizard
Squirtle
Wartortle
Blastoise
Caterpie
Metapod
Butterfree
Weedle
Kakuna
Beedrill
Pidgey
Pidgeotto
Pidgeot
Nidorina
Nidoqueen
Nidorino
Nidoking
Oddish
Gloom
Vileplume
Poliwag
Poliwhirl
Poliwrath
Abra
Kadabra
Alakazam
Machop
Machoke
Machamp
Bellsprout
Weepinbell
Victreebel
Geodude
Graveler
Golem
Gastly
Haunter
Gengar
Eevee
Vaporeon
Jolteon
Flareon
Dratini
Dragonair
Dragonite
Nidoran(M)
Nidoran(F)

18. Find the average weight and height of pokemon
(Changed from List all the pokemons which have highest power_points because the power_points is associated with moves rather than a particular pokemon)

```
SELECT AVG(height), AVG(weight) FROM pokemon;
```



Query Results

avgheight	avgweight
1.1947019867549669	45.9516556291390728

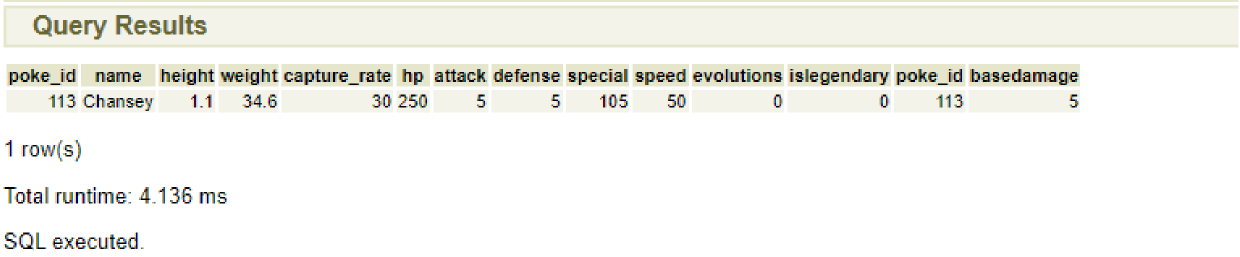
1 row(s)

Total runtime: 0.854 ms

SQL executed.

19. Find the pokemon which have the least base_damage

```
SELECT * FROM pokemon JOIN (SELECT poke_id, SUM(attack) AS
BaseDamage FROM pokemon GROUP by poke_id ORDER BY
BaseDamage ASC LIMIT 1) pokestats ON (pokestats.poke_id =
pokemon.poke_id);
```



Query Results

poke_id	name	height	weight	capture_rate	hp	attack	defense	special	speed	evolutions	islegendary	poke_id	basedamage
113	Chansey	1.1	34.6	30	250	5	5	105	50	0	0	113	5

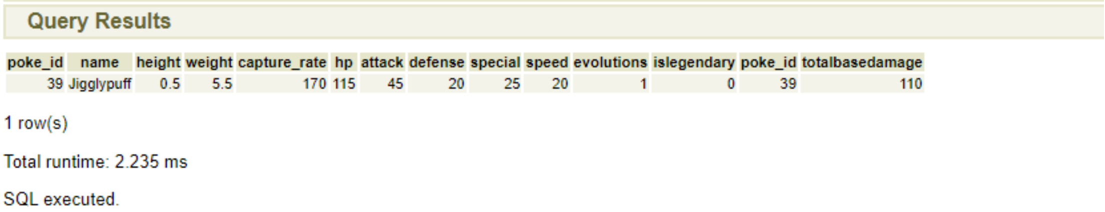
1 row(s)

Total runtime: 4.136 ms

SQL executed.

20. Find the pokemon that has the least total base damage.

```
SELECT * FROM pokemon JOIN (SELECT poke_id, SUM(attack +
defense + special +speed) AS totalBaseDamage FROM pokemon
GROUP by poke_id ORDER BY totalBaseDamage ASC LIMIT 1)
pokestats ON (pokestats.poke_id = pokemon.poke_id);
```



Query Results

poke_id	name	height	weight	capture_rate	hp	attack	defense	special	speed	evolutions	islegendary	poke_id	totalbasedamage
39	Jigglypuff	0.5	5.5	170	115	45	20	25	20	1	0	39	110

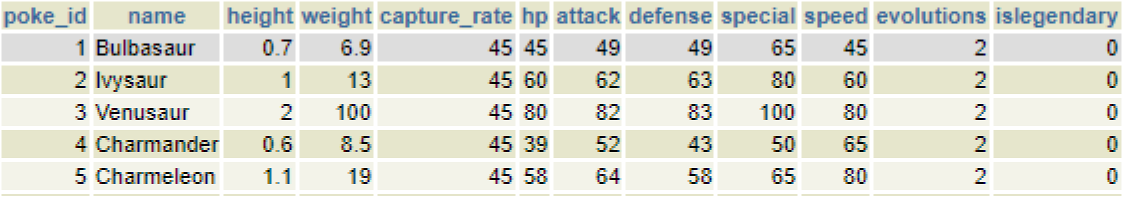
1 row(s)

Total runtime: 2.235 ms

SQL executed.

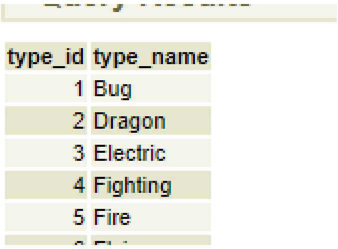
Listing of 5 rows from each of your tables

- **Pokemon**



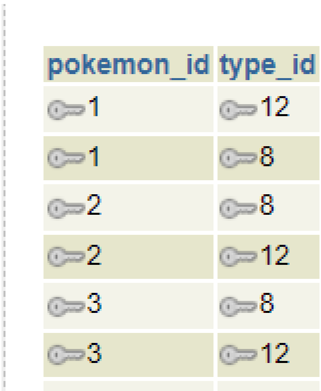
poke_id	name	height	weight	capture_rate	hp	attack	defense	special	speed	evolutions	islegendary
1	Bulbasaur	0.7	6.9	45	45	49	49	65	45	2	0
2	Ivysaur	1	13	45	60	62	63	80	60	2	0
3	Venusaur	2	100	45	80	82	83	100	80	2	0
4	Charmander	0.6	8.5	45	39	52	43	50	65	2	0
5	Charmeleon	1.1	19	45	58	64	58	65	80	2	0

- **Type**



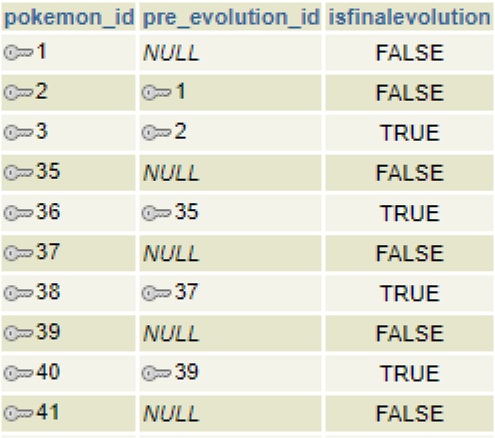
type_id	type_name
1	Bug
2	Dragon
3	Electric
4	Fighting
5	Fire

- **Pokemon_Types**



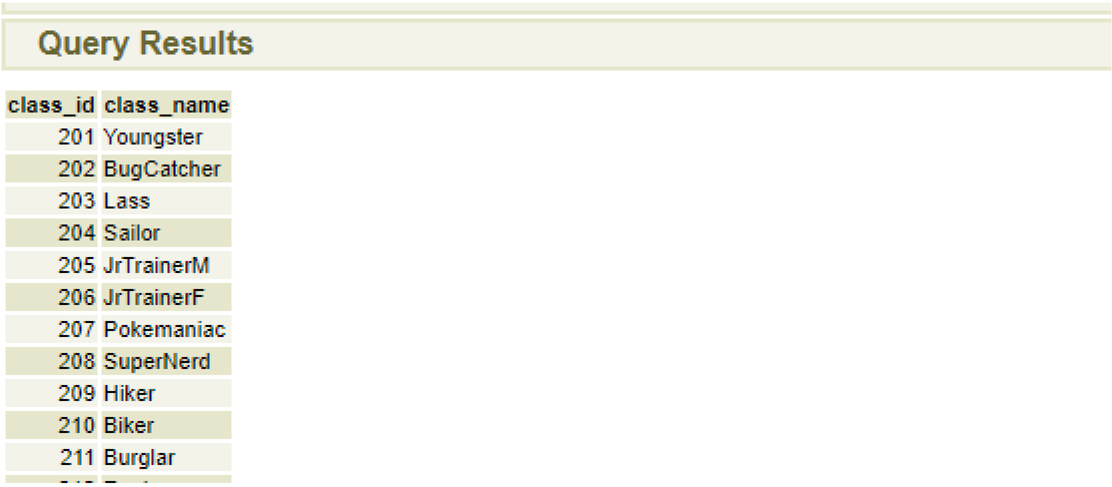
pokemon_id	type_id
1	12
1	8
2	8
2	12
3	8
3	12

- **Pokemon_Evolutions**



pokemon_id	pre_evolution_id	isfinalevolution
1	*NULL*	FALSE
2	1	FALSE
3	2	TRUE
35	*NULL*	FALSE
36	35	TRUE
37	*NULL*	FALSE
38	37	TRUE
39	*NULL*	FALSE
40	39	TRUE
41	*NULL*	FALSE

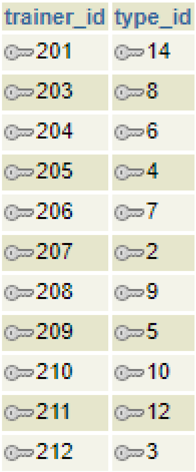
- **Trainer_Class**



Query Results

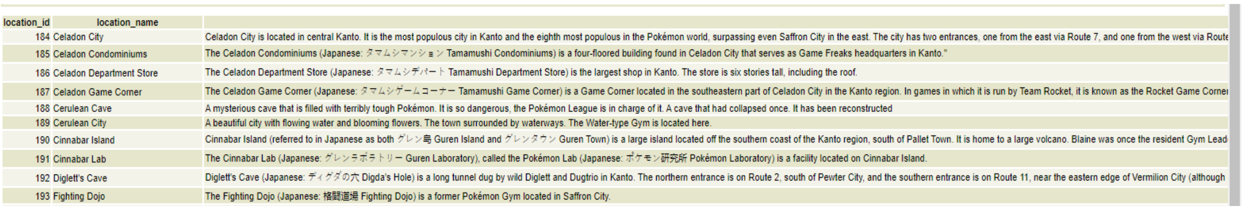
class_id	class_name
201	Youngster
202	BugCatcher
203	Lass
204	Sailor
205	JrTrainerM
206	JrTrainerF
207	Pokemaniac
208	SuperNerd
209	Hiker
210	Biker
211	Burglar

- **Trainer_PkmnTypes**



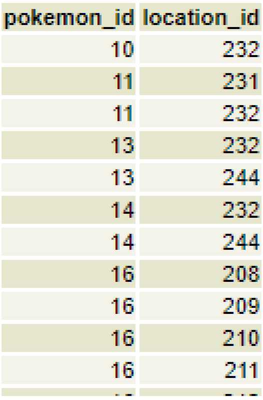
trainer_id	type_id
201	14
203	8
204	6
205	4
206	7
207	2
208	9
209	5
210	10
211	12
212	3

- **Location**



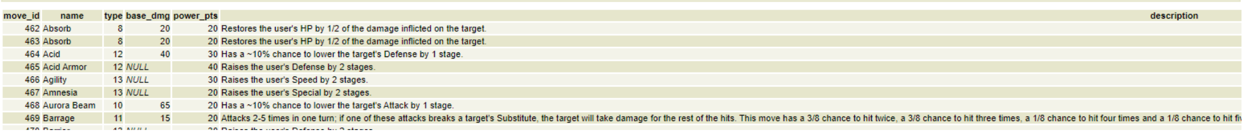
location_id	location_name	
184	Celadon City	Celadon City is located in central Kanto. It is the most populous city in Kanto and the eighth most populous in the Pokémon world, surpassing even Saffron City in the east. The city has two entrances, one from the east via Route 7, and one from the west via Route
185	Celadon Condominiums	The Celadon Condominiums (Japanese: タマムシマンション Tamamushi Condominiums) is a four-floored building found in Celadon City that serves as Game Freaks headquarters in Kanto."
186	Celadon Department Store	The Celadon Department Store (Japanese: タマムシデパート Tamamushi Department Store) is the largest shop in Kanto. The store is six stories tall, including the roof.
187	Celadon Game Corner	The Celadon Game Corner (Japanese: タマムシゲームコーナー Tamamushi Game Corner) is a Game Corner located in the southeastern part of Celadon City in the Kanto region. In games in which it is run by Team Rocket, it is known as the Rocket Game Corner
188	Cerulean Cave	A mysterious cave that is filled with terribly tough Pokémon. It is so dangerous, the Pokémon League is in charge of it. A cave that had collapsed once. It has been reconstructed
189	Cerulean City	A beautiful city with flowing water and blooming flowers. The town surrounded by waterways. The Water-type Gym is located here.
190	Cinnabar Island	Cinnabar Island (referred to in Japanese as both グレン島 Guren Island and グレンタウン Guren Town) is a large island located off the southern coast of the Kanto region, south of Pallet Town. It is home to a large volcano. Blaine was once the resident Gym Leade
191	Cinnabar Lab	The Cinnabar Lab (Japanese: グレンラボラトリー Guren Laboratory), called the Pokémon Lab (Japanese: ポケモン研究所 Pokémon Laboratory) is a facility located on Cinnabar Island.
192	Diglett's Cave	Diglett's Cave (Japanese: ディグダの穴 Digda's Hole) is a long tunnel dug by wild Diglett and Dugtrio in Kanto. The northern entrance is on Route 2, south of Pewter City, and the southern entrance is on Route 11, near the eastern edge of Vermilion City (although
193	Fighting Dojo	The Fighting Dojo (Japanese: 格闘道場 Fighting Dojo) is a former Pokémon Gym located in Saffron City.

- **Pokemon_Location**



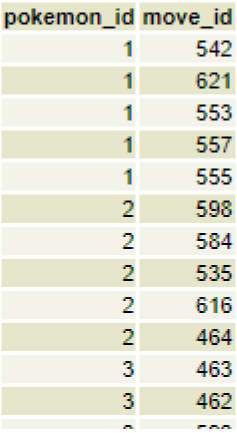
pokemon_id	location_id
10	232
11	231
11	232
13	232
13	244
14	232
14	244
16	208
16	209
16	210
16	211

- **Moves**



move_id	name	type	base_dmg	power_pts	description
462	Absorb	8	20	20	Restores the user's HP by 1/2 of the damage inflicted on the target.
463	Absorb	8	20	20	Restores the user's HP by 1/2 of the damage inflicted on the target.
464	Acid	12	40	30	Has a ~10% chance to lower the target's Defense by 1 stage.
465	Acid Armor	12	*NULL*	40	Raises the user's Defense by 2 stages.
466	Agility	13	*NULL*	30	Raises the user's Speed by 2 stages.
467	Amnesia	13	*NULL*	20	Raises the user's Special by 2 stages.
468	Aurora Beam	10	65	20	Has a ~10% chance to lower the target's Attack by 1 stage.
469	Barrage	11	15	20	Attacks 2-5 times in one turn; if one of these attacks breaks a target's Substitute, the target will take damage for the rest of the hits. This move has a 3/8 chance to hit twice, a 3/8 chance to hit three times, a 1/8 chance to hit four times and a 1/8 chance to hit fiv
470	Barrier	13	*NULL*	30	Raises the user's Defense by 2 stages.

- **Pokemon_moves**



pokemon_id	move_id
1	542
1	621
1	553
1	557
1	555
2	598
2	584
2	535
2	616
2	464
3	463
3	462