Smart Pokédex

A Database Design Solution

By John Randis

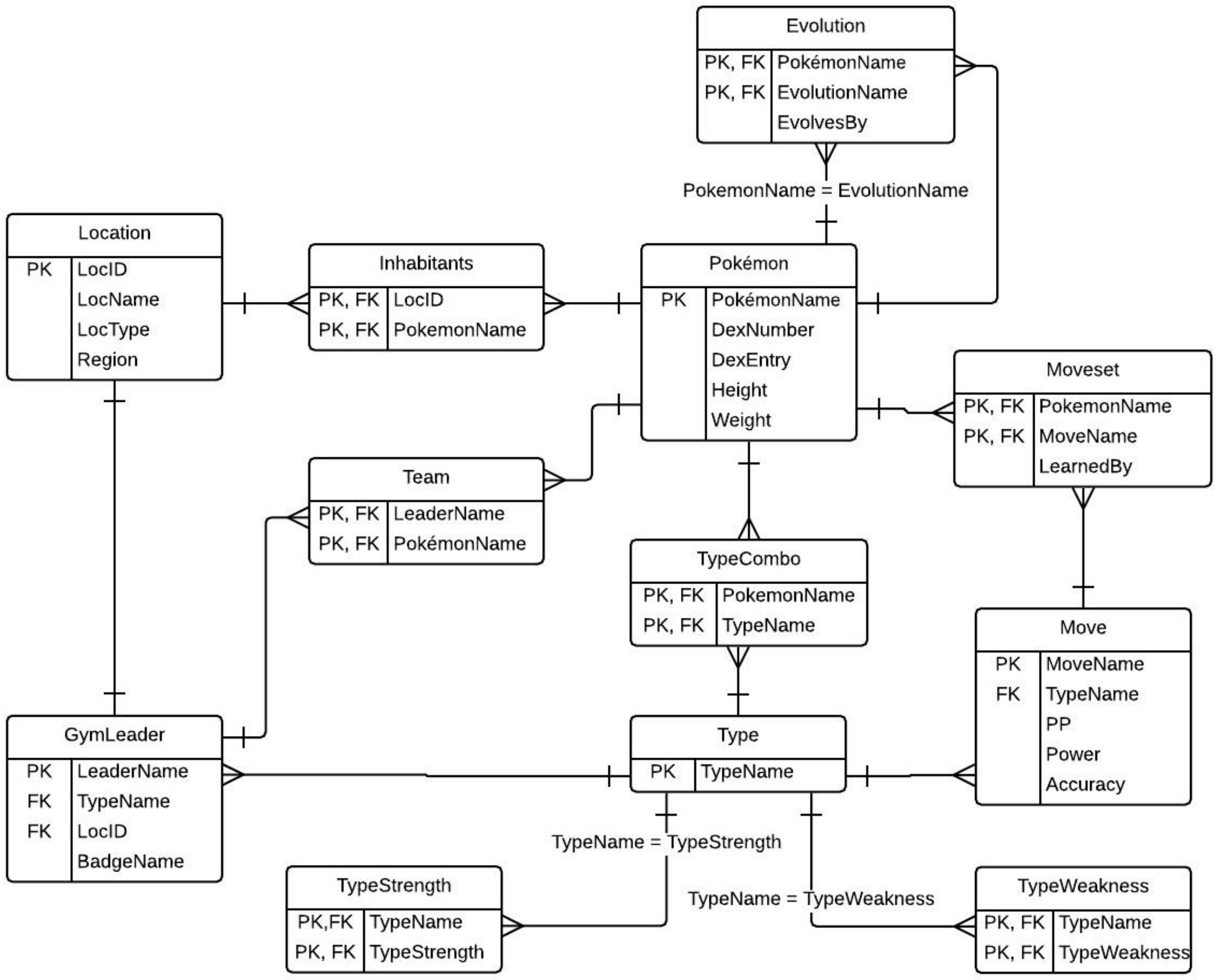
Contents

Executive Summary....3
Entity Relationship Diagram....4
Tables....5
Views....17
Reports....20
Stored Procedures....21
Triggers....22
Security....23
Issues/Future Considerations....24

Executive Summary

The purpose of this database is to act as a resource for Pokémon trainers to manage discovered Pokémon. This database will include locations of Pokémon, their weaknesses, evolutions, and moves they learn, making it an invaluable resource no trainer should be caught without. Trainers can also store their Pokémon teams, as well as view and catalog the teams of gym leaders in order to be better prepared for a battle. This document includes the tables, functional dependencies, views, reports, procedures, and triggers that went in to creating the database. Using this Smart Pokédex, trainers will be able to keep an accurate and precise record of their Pokémon in order to become the very best.

Entity Relationship Diagram



Pokémon Table

There are hundreds of known Pokémon across the world and even more yet to be discovered. This table holds a record of each Pokémon, it's dex number (order of discovery), a dex entry that provides a brief description of the Pokémon, and it's height (m) and weight (kg).

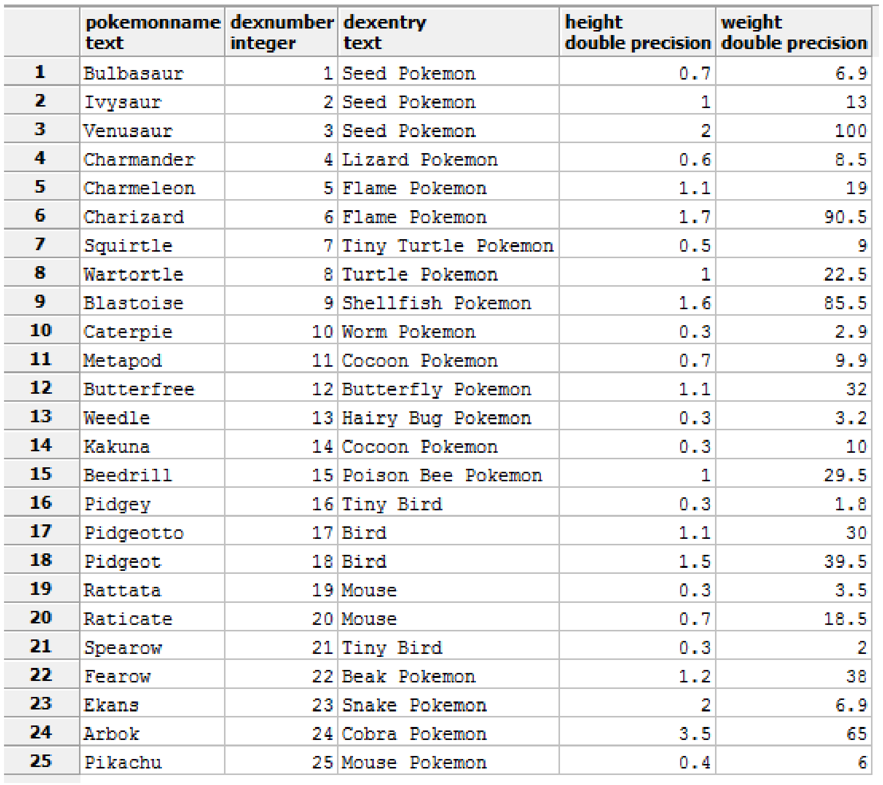
Create Statement:

```
CREATE TABLE Pokemon (
  PokemonName text not null,
  DexNumber integer not null check (DexNumber > 0),
  DexEntry text not null,
  Height float not null check (Height > 0),
  Weight float not null check (Weight > 0),
  PRIMARY KEY (PokemonName)
);
```

Functional Dependencies:

PokemonName → DexNumber, DexEntry, Height, Weight

Sample Data:



	pokemonname	dexnumber	dexentry	height	weight
	text	integer	text	double precision	double precision
1	Bulbasaur	1	Seed Pokemon	0.7	6.9
2	Ivysaur	2	Seed Pokemon	1	13
3	Venusaur	3	Seed Pokemon	2	100
4	Charmander	4	Lizard Pokemon	0.6	8.5
5	Charmeleon	5	Flame Pokemon	1.1	19
6	Charizard	6	Flame Pokemon	1.7	90.5
7	Squirtle	7	Tiny Turtle Pokemon	0.5	9
8	Wartortle	8	Turtle Pokemon	1	22.5
9	Blastoise	9	Shellfish Pokemon	1.6	85.5
10	Caterpie	10	Worm Pokemon	0.3	2.9
11	Metapod	11	Cocoon Pokemon	0.7	9.9
12	Butterfree	12	Butterfly Pokemon	1.1	32
13	Weedle	13	Hairy Bug Pokemon	0.3	3.2
14	Kakuna	14	Cocoon Pokemon	0.3	10
15	Beedrill	15	Poison Bee Pokemon	1	29.5
16	Pidgey	16	Tiny Bird	0.3	1.8
17	Pidgeotto	17	Bird	1.1	30
18	Pidgeot	18	Bird	1.5	39.5
19	Rattata	19	Mouse	0.3	3.5
20	Raticate	20	Mouse	0.7	18.5
21	Spearow	21	Tiny Bird	0.3	2
22	Fearow	22	Beak Pokemon	1.2	38
23	Ekans	23	Snake Pokemon	2	6.9
24	Arbok	24	Cobra Pokemon	3.5	65
25	Pikachu	25	Mouse Pokemon	0.4	6

Evolution Table

Pokémon that have the ability to evolve into new Pokémon will be added to this table. Users will be able to query the Pokémon's name and get their evolution, as well as see what Pokémon come before in the evolutionary chain. Also includes the method by which the Pokémon evolves.

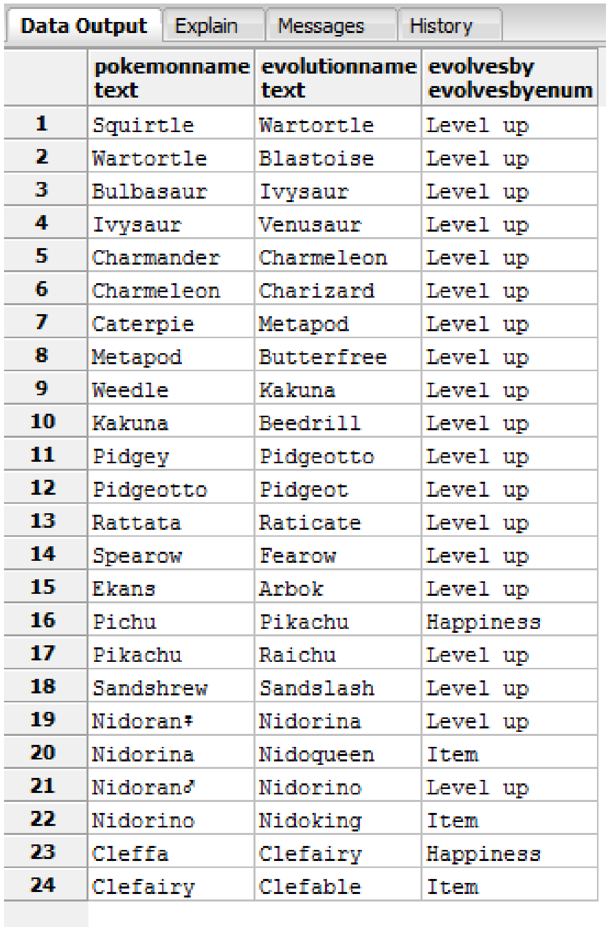
Create Statement:

```
CREATE TABLE Evolution (
  PokemonName text references Pokemon(PokemonName) not null,
  EvolutionName text references Pokemon(PokemonName) not null unique,
  EvolvesBy EvolvesByEnum not null,
  PRIMARY KEY (PokemonName, EvolutionName)
);
```

Functional Dependencies:

PokemonName → EvolutionName, EvolvesBy

Sample Data:



	pokemonname text	evolutionname text	evolvesby evolvesbyenum
1	Squirtle	Wartortle	Level up
2	Wartortle	Blastoise	Level up
3	Bulbasaur	Ivysaur	Level up
4	Ivysaur	Venusaur	Level up
5	Charmander	Charmeleon	Level up
6	Charmeleon	Charizard	Level up
7	Caterpie	Metapod	Level up
8	Metapod	Butterfree	Level up
9	Weedle	Kakuna	Level up
10	Kakuna	Beedrill	Level up
11	Pidgey	Pidgeotto	Level up
12	Pidgeotto	Pidgeot	Level up
13	Rattata	Raticate	Level up
14	Spearow	Fearow	Level up
15	Ekans	Arbok	Level up
16	Pichu	Pikachu	Happiness
17	Pikachu	Raichu	Level up
18	Sandshrew	Sandslash	Level up
19	Nidoran♀	Nidorina	Level up
20	Nidorina	Nidoqueen	Item
21	Nidoran♂	Nidorino	Level up
22	Nidorino	Nidoking	Item
23	Cleffa	Clefairy	Happiness
24	Clefairy	Clefable	Item

Location Table

The main purpose of the location table is so that users can query the Pokémon inhabitants at each location, in order to see where in the world a specific Pokémon resides.

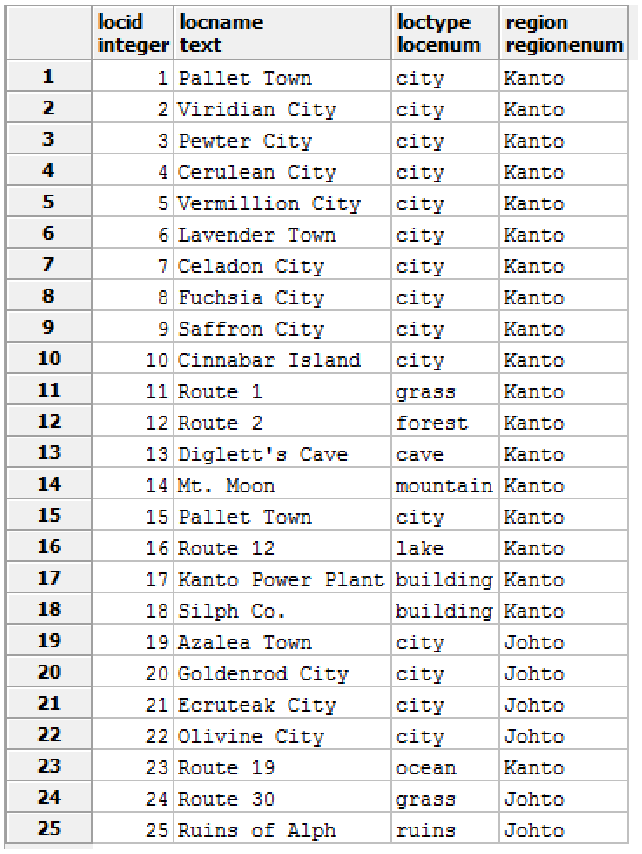
<u>Create Statement</u>:

```
CREATE TABLE Location (
  LocID serial not null,
  LocName text not null,
  LocType LocEnum,
  Region RegionEnum,
  PRIMARY KEY (LocID)
);
```

<u>Functional Dependencies:</u>

LocID → LocName, LocType, Region

<u>Sample Data:</u>



	locid	locname	loctype	region
	integer	text	locenum	regionenum
1	1	Pallet Town	city	Kanto
2	2	Viridian City	city	Kanto
3	3	Pewter City	city	Kanto
4	4	Cerulean City	city	Kanto
5	5	Vermillion City	city	Kanto
6	6	Lavender Town	city	Kanto
7	7	Celadon City	city	Kanto
8	8	Fuchsia City	city	Kanto
9	9	Saffron City	city	Kanto
10	10	Cinnabar Island	city	Kanto
11	11	Route 1	grass	Kanto
12	12	Route 2	forest	Kanto
13	13	Diglett's Cave	cave	Kanto
14	14	Mt. Moon	mountain	Kanto
15	15	Pallet Town	city	Kanto
16	16	Route 12	lake	Kanto
17	17	Kanto Power Plant	building	Kanto
18	18	Silph Co.	building	Kanto
19	19	Azalea Town	city	Johto
20	20	Goldenrod City	city	Johto
21	21	Ecruteak City	city	Johto
22	22	Olivine City	city	Johto
23	23	Route 19	ocean	Kanto
24	24	Route 30	grass	Johto
25	25	Ruins of Alph	ruins	Johto

Inhabitants Table

This table provides a link between Pokémon and Location. It stores all the locations of where a given Pokémon lives and where in the world it can be found.

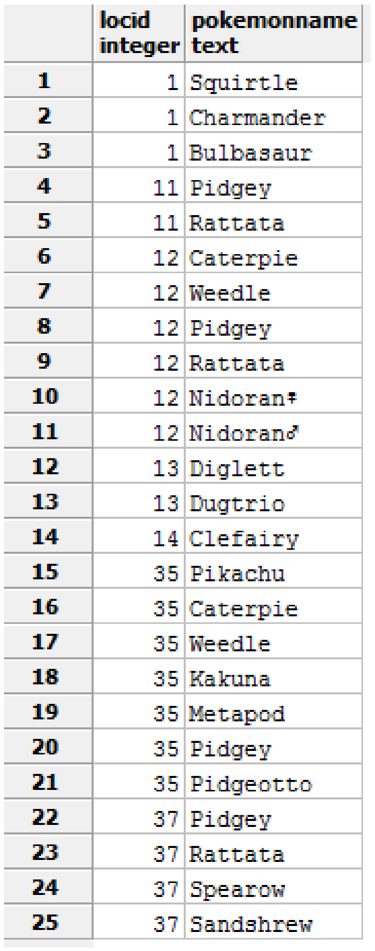
Create Statement:

```
CREATE TABLE Inhabitants (
  LocID integer not null references Location(LocID) not null,
  PokemonName text references Pokemon(PokemonName) not null,
  PRIMARY KEY (LocID, PokemonName)
);
```

Functional Dependencies:

LocID → PokemonName

Sample Data:



	locid integer	pokemonname text
1	1	Squirtle
2	1	Charmander
3	1	Bulbasaur
4	11	Pidgey
5	11	Rattata
6	12	Caterpie
7	12	Weedle
8	12	Pidgey
9	12	Rattata
10	12	Nidoran♀
11	12	Nidoran♂
12	13	Diglett
13	13	Dugtrio
14	14	Clefairy
15	35	Pikachu
16	35	Caterpie
17	35	Weedle
18	35	Kakuna
19	35	Metapod
20	35	Pidgey
21	35	Pidgeotto
22	37	Pidgey
23	37	Rattata
24	37	Spearow
25	37	Sandshrew

Type Table

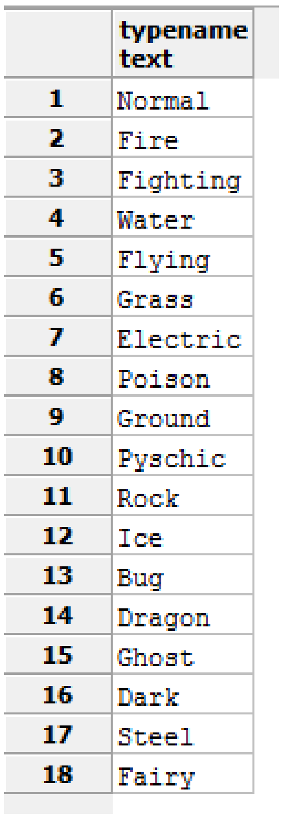
An understanding of the elemental types is key for any trainer. Each type its own unique set of strengths and weaknesses to other types. Each Pokémon possesses at least one type, and each move has a type associated with it. Types are also adopted by gym leaders.

Create Statement:

```
CREATE TABLE Type (
  TypeName text not null unique,
  PRIMARY KEY (TypeName)
);
```

Functional Dependencies: N/A

Sample Data:



	typename text
1	Normal
2	Fire
3	Fighting
4	Water
5	Flying
6	Grass
7	Electric
8	Poison
9	Ground
10	Pyschic
11	Rock
12	Ice
13	Bug
14	Dragon
15	Ghost
16	Dark
17	Steel
18	Fairy

Type Strength Table

A given type has its own specific set of types that it is strong against. That is, moves that of this type will be super-effect on Pokémon that belong to the types it is "strong" against. For instance, by querying the strengths of fire, we can see that fire-type moves will be super effective on grass, bug, ice, and steel type Pokémon.

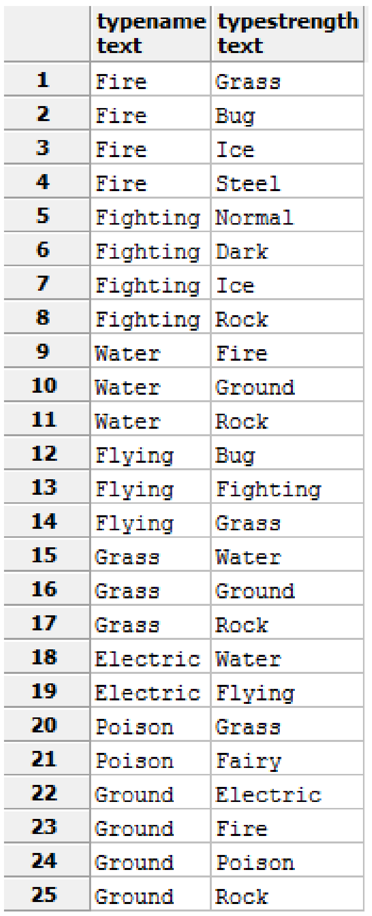
<u>Create Statement:</u>

```
CREATE TABLE TypeStrength (
  TypeName text not null references Type(TypeName) not null,
  TypeStrength text not null references Type(TypeName) not null
);
```

<u>Functional Dependencies</u>:

TypeName → TypeStrength

<u>Sample Data</u>:



	typename text	typestrength text
1	Fire	Grass
2	Fire	Bug
3	Fire	Ice
4	Fire	Steel
5	Fighting	Normal
6	Fighting	Dark
7	Fighting	Ice
8	Fighting	Rock
9	Water	Fire
10	Water	Ground
11	Water	Rock
12	Flying	Bug
13	Flying	Fighting
14	Flying	Grass
15	Grass	Water
16	Grass	Ground
17	Grass	Rock
18	Electric	Water
19	Electric	Flying
20	Poison	Grass
21	Poison	Fairy
22	Ground	Electric
23	Ground	Fire
24	Ground	Poison
25	Ground	Rock

Type Weakness Table

A given type has its own specific set of types that it is weak against. Pokémon that belong to this type will be especially susceptible to moves possessing the type it is weak against. For instance, by querying the weaknesses of fire, we can see that water-type moves will be super effective on fire type Pokémon.

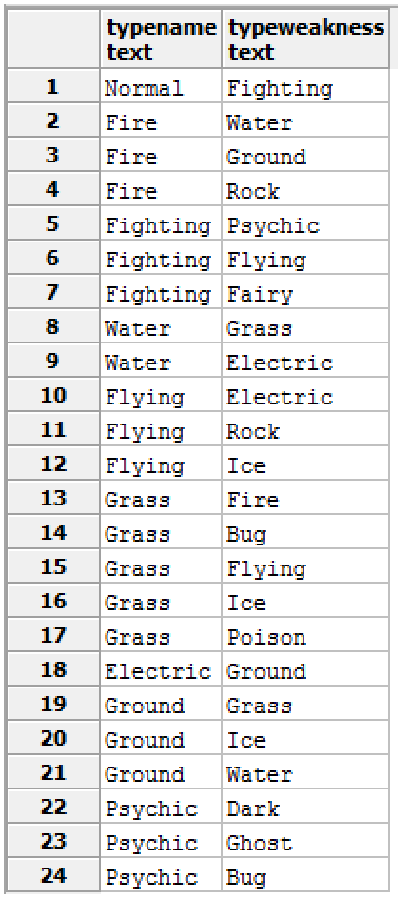
Create Statement:

```
CREATE TABLE TypeWeakness (
  TypeName text not null references Type(TypeName) not null,
  TypeWeakness text not null references Type(TypeName) not null
);
```

Functional Dependencies:

TypeName → TypeWeakness

Sample Data:



	typename text	typeweakness text
1	Normal	Fighting
2	Fire	Water
3	Fire	Ground
4	Fire	Rock
5	Fighting	Psychic
6	Fighting	Flying
7	Fighting	Fairy
8	Water	Grass
9	Water	Electric
10	Flying	Electric
11	Flying	Rock
12	Flying	Ice
13	Grass	Fire
14	Grass	Bug
15	Grass	Flying
16	Grass	Ice
17	Grass	Poison
18	Electric	Ground
19	Ground	Grass
20	Ground	Ice
21	Ground	Water
22	Psychic	Dark
23	Psychic	Ghost
24	Psychic	Bug

Type Combo Table

Each Pokémon can have up to two types. This table keeps a record every Pokémon in the database and the type or types they are affiliated with.

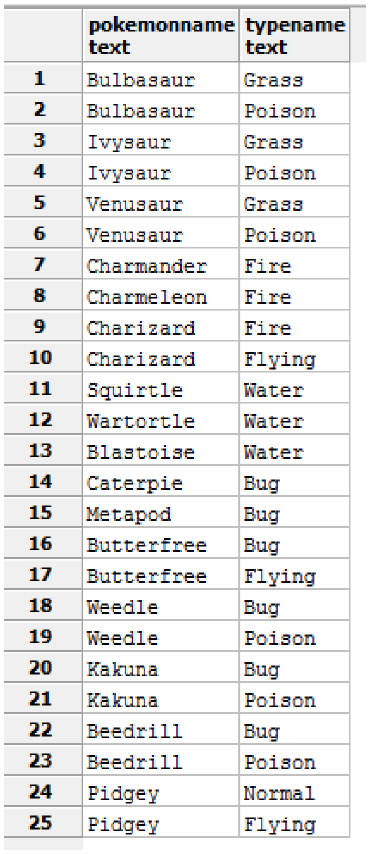
Create Statement:

```
CREATE TABLE TypeCombo (
  PokemonName text references Pokemon(PokemonName) not null,
  TypeName text not null references Type(TypeName) not null,
  PRIMARY KEY (PokemonName, TypeName)
);
```

Functional Dependencies:

PokemonName → TypeName

Sample Data:



	pokemonname text	typename text
1	Bulbasaur	Grass
2	Bulbasaur	Poison
3	Ivysaur	Grass
4	Ivysaur	Poison
5	Venusaur	Grass
6	Venusaur	Poison
7	Charmander	Fire
8	Charmeleon	Fire
9	Charizard	Fire
10	Charizard	Flying
11	Squirtle	Water
12	Wartortle	Water
13	Blastoise	Water
14	Caterpie	Bug
15	Metapod	Bug
16	Butterfree	Bug
17	Butterfree	Flying
18	Weedle	Bug
19	Weedle	Poison
20	Kakuna	Bug
21	Kakuna	Poison
22	Beedrill	Bug
23	Beedrill	Poison
24	Pidgey	Normal
25	Pidgey	Flying

Move Table

Pokémon are able to learn a variety of moves to be used in battle. Moves all belong to an elemental type, and have base power and a base accuracy. Moves also possess PP, or power points, which determine how many times the move can be used.

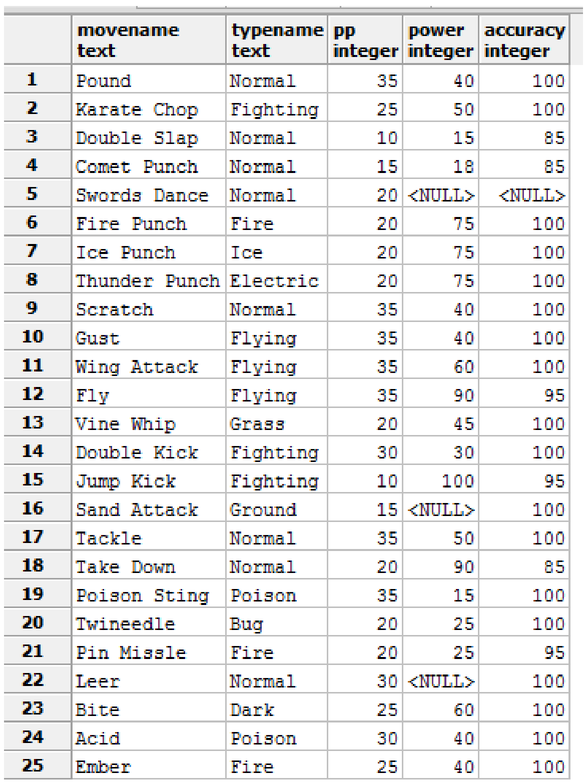
Create Statement:

```
CREATE TABLE Move (
  MoveName text not null,
  TypeName text not null references Type(TypeName),
  PP integer not null check (PP > 0),
  Power integer check (Power > 0),
  Accuracy integer check (Accuracy >= 0 and Accuracy <= 100),
  PRIMARY KEY (MoveName)
);
```

Functional Dependencies:

MoveName → TypeName, PP, Power, Accuracy

Sample Data:



	movename text	typename text	pp integer	power integer	accuracy integer
1	Pound	Normal	35	40	100
2	Karate Chop	Fighting	25	50	100
3	Double Slap	Normal	10	15	85
4	Comet Punch	Normal	15	18	85
5	Swords Dance	Normal	20	<NULL>	<NULL>
6	Fire Punch	Fire	20	75	100
7	Ice Punch	Ice	20	75	100
8	Thunder Punch	Electric	20	75	100
9	Scratch	Normal	35	40	100
10	Gust	Flying	35	40	100
11	Wing Attack	Flying	35	60	100
12	Fly	Flying	35	90	95
13	Vine Whip	Grass	20	45	100
14	Double Kick	Fighting	30	30	100
15	Jump Kick	Fighting	10	100	95
16	Sand Attack	Ground	15	<NULL>	100
17	Tackle	Normal	35	50	100
18	Take Down	Normal	20	90	85
19	Poison Sting	Poison	35	15	100
20	Twineedle	Bug	20	25	100
21	Pin Missle	Fire	20	25	95
22	Leer	Normal	30	<NULL>	100
23	Bite	Dark	25	60	100
24	Acid	Poison	30	40	100
25	Ember	Fire	25	40	100

Moveset Table

Each Pokemon has a wide set of moves it is able to learn, they either possess at birth by default, possess at birth by selective breeding, learn by leveling up, from a technical machine or hidden machine (TM/HM), or from a move tutor.

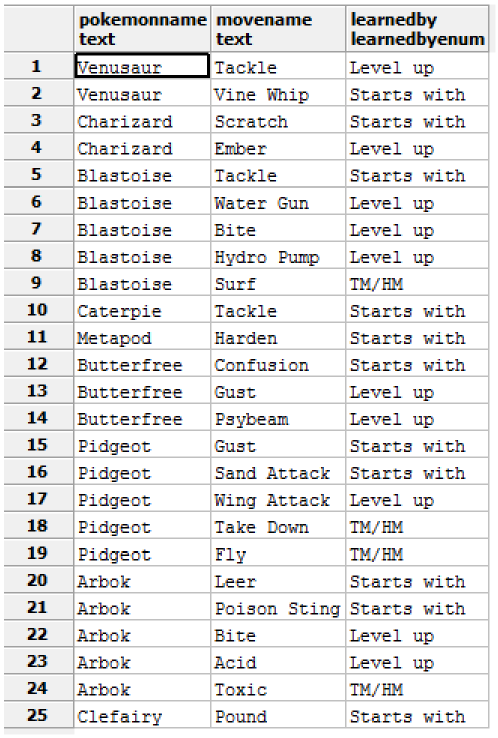
Create Statement:

```
CREATE TABLE Moveset (
  PokemonName text not null references Pokemon(PokemonName),
  MoveName text not null references Move(MoveName),
  LearnedBy LearnedByEnum not null,
  PRIMARY KEY (PokemonName, MoveName)
);
```

Functional Dependencies:

PokemonName → MoveName, LearnedBy

Sample Data:



	pokemonname text	movename text	learnedby learnedbyenum
1	Venusaur	Tackle	Level up
2	Venusaur	Vine Whip	Starts with
3	Charizard	Scratch	Starts with
4	Charizard	Ember	Level up
5	Blastoise	Tackle	Starts with
6	Blastoise	Water Gun	Level up
7	Blastoise	Bite	Level up
8	Blastoise	Hydro Pump	Level up
9	Blastoise	Surf	TM/HM
10	Caterpie	Tackle	Starts with
11	Metapod	Harden	Starts with
12	Butterfree	Confusion	Starts with
13	Butterfree	Gust	Level up
14	Butterfree	Psybeam	Level up
15	Pidgeot	Gust	Starts with
16	Pidgeot	Sand Attack	Starts with
17	Pidgeot	Wing Attack	Level up
18	Pidgeot	Take Down	TM/HM
19	Pidgeot	Fly	TM/HM
20	Arbok	Leer	Starts with
21	Arbok	Poison Sting	Starts with
22	Arbok	Bite	Level up
23	Arbok	Acid	Level up
24	Arbok	Toxic	TM/HM
25	Clefairy	Pound	Starts with

Gym Leader Table

Gym leaders are the highest ranking trainers across the world, who exist for trainers to challenge and test their skill in order to obtain gym badges. Each gym leader affiliates themselves with a type. They only use Pokémon of this type and theme their gym and environment around this type in order to reflect their battling style. This table stores the leader's name, their type, the location of their gym, and the name of their badge.

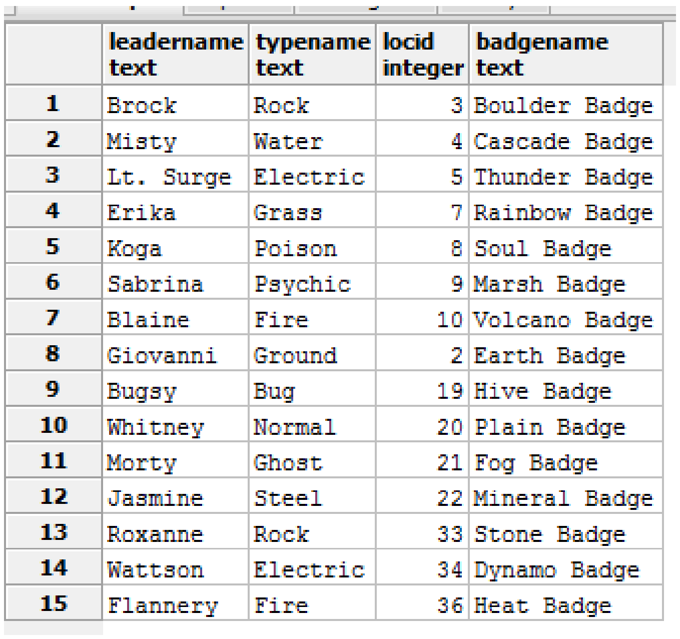
Create Statement:

```
CREATE TABLE GymLeader (
  LeaderName text not null,
  TypeName text references Type(TypeName),
  LocID integer references Location(LocID) not null,
  BadgeName text not null,
  PRIMARY KEY (LeaderName)
);
```

Functional Dependencies:

LeaderName → TypeName, LocID, BadgeName

Sample Data:



	leadername text	typename text	locid integer	badgename text
1	Brock	Rock	3	Boulder Badge
2	Misty	Water	4	Cascade Badge
3	Lt. Surge	Electric	5	Thunder Badge
4	Erika	Grass	7	Rainbow Badge
5	Koga	Poison	8	Soul Badge
6	Sabrina	Psychic	9	Marsh Badge
7	Blaine	Fire	10	Volcano Badge
8	Giovanni	Ground	2	Earth Badge
9	Bugsy	Bug	19	Hive Badge
10	Whitney	Normal	20	Plain Badge
11	Morty	Ghost	21	Fog Badge
12	Jasmine	Steel	22	Mineral Badge
13	Roxanne	Rock	33	Stone Badge
14	Wattson	Electric	34	Dynamo Badge
15	Flannery	Fire	36	Heat Badge

Team Table

Each gym leader can have up to six Pokémon on their team. This table keeps track of the set of Pokémon that each leader will use in battle.

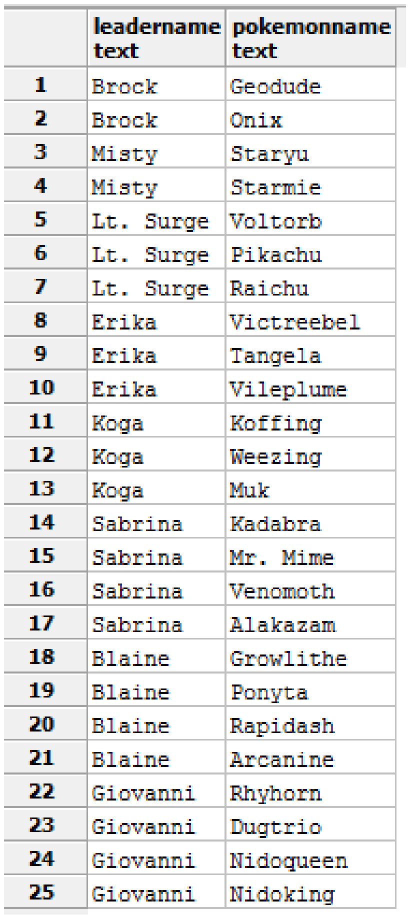
Create Statement:

```
CREATE TABLE Team (
  LeaderName text not null references GymLeader(LeaderName) not null ,
  PokemonName text references Pokemon(PokemonName) not null,
  PRIMARY KEY (LeaderName, PokemonName)
);
```

Functional Dependencies:

LeaderName → PokemonName

Sample Data:



	leadername text	pokemonname text
1	Brock	Geodude
2	Brock	Onix
3	Misty	Staryu
4	Misty	Starmie
5	Lt. Surge	Voltorb
6	Lt. Surge	Pikachu
7	Lt. Surge	Raichu
8	Erika	Victreebel
9	Erika	Tangela
10	Erika	Vileplume
11	Koga	Koffing
12	Koga	Weezing
13	Koga	Muk
14	Sabrina	Kadabra
15	Sabrina	Mr. Mime
16	Sabrina	Venomoth
17	Sabrina	Alakazam
18	Blaine	Growlithe
19	Blaine	Ponyta
20	Blaine	Rapidash
21	Blaine	Arcanine
22	Giovanni	Rhyhorn
23	Giovanni	Dugtrio
24	Giovanni	Nidoqueen
25	Giovanni	Nidoking

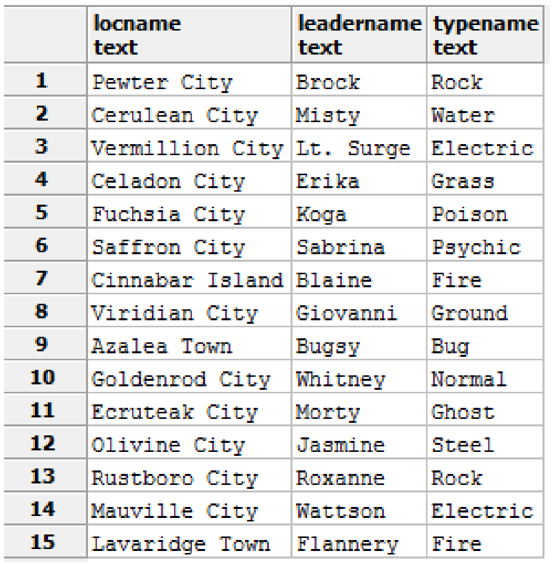
Gym Leader Locations View

This view shows all cities that contain a gym, and the gym leader that runs the gym in that city.

Create Statement:

```
CREATE VIEW GymLeaderLocations AS
  SELECT Location.LocName,
  GymLeader.LeaderName,
  GymLeader.TypeName
  FROM GymLeader
  INNER JOIN Location
  ON GymLeader.LocID = Location.LocID;
```

Sample Data:



	locname text	leadername text	typename text
1	Pewter City	Brock	Rock
2	Cerulean City	Misty	Water
3	Vermillion City	Lt. Surge	Electric
4	Celadon City	Erika	Grass
5	Fuchsia City	Koga	Poison
6	Saffron City	Sabrina	Psychic
7	Cinnabar Island	Blaine	Fire
8	Viridian City	Giovanni	Ground
9	Azalea Town	Bugsy	Bug
10	Goldenrod City	Whitney	Normal
11	Ecruteak City	Morty	Ghost
12	Olivine City	Jasmine	Steel
13	Rustboro City	Roxanne	Rock
14	Mauville City	Wattson	Electric
15	Lavaridge Town	Flannery	Fire

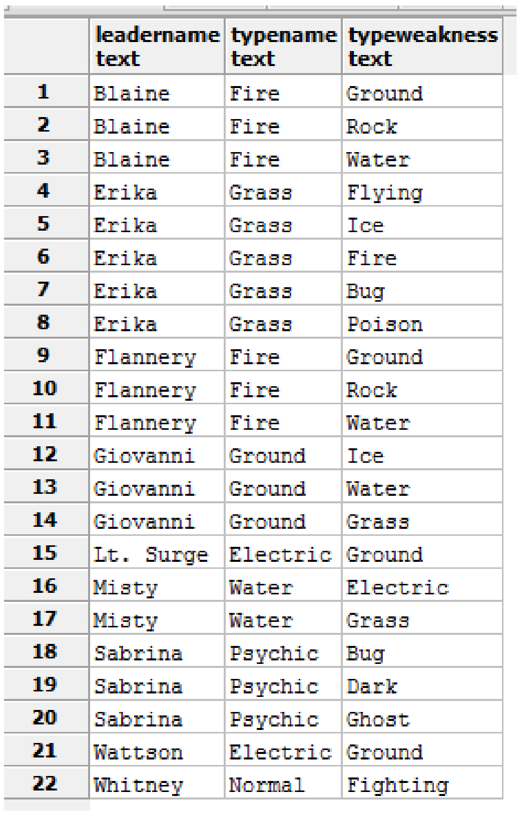
Gym Leader Weakness View

This view will show each gym leader, their type, and all the weaknesses they have. This is essential for a trainer to be prepared for a battle with a gym leader.

Create Statement:

```
CREATE VIEW GymLeaderWeakness AS
  Select GymLeader.LeaderName,
  GymLeader.TypeName,
  TypeWeakness.TypeWeakness
  FROM GymLeader
  INNER JOIN TypeWeakness
  ON GymLeader.TypeName = TypeWeakness.TypeName
  ORDER BY LeaderName ASC;
```

Sample Data:



	leadername text	typename text	typeweakness text
1	Blaine	Fire	Ground
2	Blaine	Fire	Rock
3	Blaine	Fire	Water
4	Erika	Grass	Flying
5	Erika	Grass	Ice
6	Erika	Grass	Fire
7	Erika	Grass	Bug
8	Erika	Grass	Poison
9	Flannery	Fire	Ground
10	Flannery	Fire	Rock
11	Flannery	Fire	Water
12	Giovanni	Ground	Ice
13	Giovanni	Ground	Water
14	Giovanni	Ground	Grass
15	Lt. Surge	Electric	Ground
16	Misty	Water	Electric
17	Misty	Water	Grass
18	Sabrina	Psychic	Bug
19	Sabrina	Psychic	Dark
20	Sabrina	Psychic	Ghost
21	Wattson	Electric	Ground
22	Whitney	Normal	Fighting

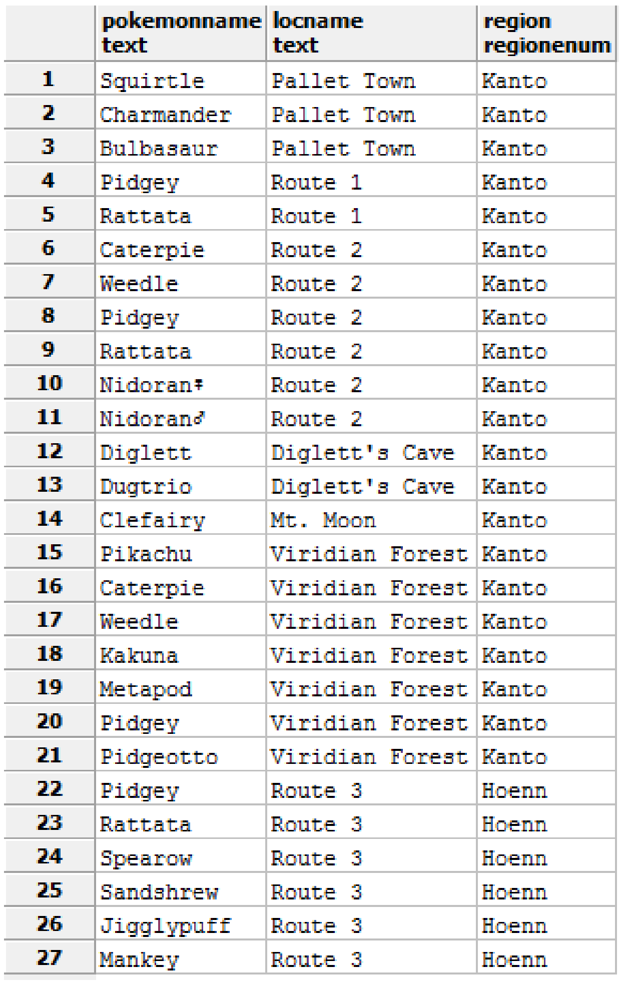
Pokémon Location View

This view will display each Pokémon and the known locations they can be found. Useful resource for trainers who are looking where to find specific Pokémon.

<u>Create Statement</u>:

```
CREATE VIEW PokemonLocations AS
  SELECT Inhabitants.PokemonName,
  Location.LocName,
  Location.Region
  FROM Inhabitants
  INNER JOIN Location
  ON Inhabitants.LocID = Location.LocID;
```

<u>Sample Data</u>:



	pokemonname text	locname text	region regionenum
1	Squirtle	Pallet Town	Kanto
2	Charmander	Pallet Town	Kanto
3	Bulbasaur	Pallet Town	Kanto
4	Pidgey	Route 1	Kanto
5	Rattata	Route 1	Kanto
6	Caterpie	Route 2	Kanto
7	Weedle	Route 2	Kanto
8	Pidgey	Route 2	Kanto
9	Rattata	Route 2	Kanto
10	Nidoran♀	Route 2	Kanto
11	Nidoran♂	Route 2	Kanto
12	Diglett	Diglett's Cave	Kanto
13	Dugtrio	Diglett's Cave	Kanto
14	Clefairy	Mt. Moon	Kanto
15	Pikachu	Viridian Forest	Kanto
16	Caterpie	Viridian Forest	Kanto
17	Weedle	Viridian Forest	Kanto
18	Kakuna	Viridian Forest	Kanto
19	Metapod	Viridian Forest	Kanto
20	Pidgey	Viridian Forest	Kanto
21	Pidgeotto	Viridian Forest	Kanto
22	Pidgey	Route 3	Hoenn
23	Rattata	Route 3	Hoenn
24	Spearow	Route 3	Hoenn
25	Sandshrew	Route 3	Hoenn
26	Jigglypuff	Route 3	Hoenn
27	Mankey	Route 3	Hoenn

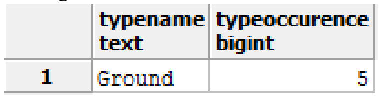
Strongest Type Report

This query will return the type which has the most appearances in the TypeStrengths table. In other words, the type which has the most strengths against other types.

Query:

```
SELECT TypeName,
COUNT(TypeName) AS TypeOccurence
FROM TypeStrength
GROUP BY TypeName
ORDER BY TypeOccurence DESC
LIMIT 1;
```

Output:



	typename text	typeoccurence bigint
1	Ground	5

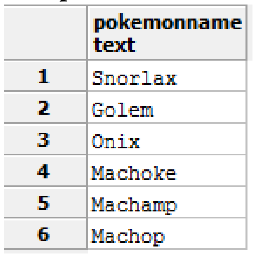
Strongest Heavy Slam Users Report

Heavy Slam is a move that does damage based on the weight of the Pokémon. This query displays all the Pokémon that learn heavy slam, in order of heaviest to lightest.

Query:

```
SELECT PokemonName
FROM Pokemon
WHERE PokemonName in (
  SELECT PokemonName
  FROM Moveset
  WHERE MoveName = 'Heavy Slam')
ORDER BY Weight DESC;
```

Output:



	pokemonname text
1	Snorlax
2	Golem
3	Onix
4	Machoke
5	Machamp
6	Machop

Moves of Prior Evolution Stored Procedure

Upon Evolution, Pokémon are able to retain any moves that were learned by their previous evolution. This function allows the user to specify an evolved Pokémon and returns all the moves that were able to be learned by its previous evolutions.

Function:

```
CREATE OR REPLACE FUNCTION get_moves_by_prior_evolution(text, REFCURSOR)
RETURNS REFCURSOR AS
$$
DECLARE
   evo text := $1;
   resultset REFCURSOR := $2;
BEGIN
   open resultset FOR
      SELECT MoveName
      FROM Moveset
      WHERE PokemonName in (
         SELECT PokemonName
         FROM Evolution
         WHERE EvolutionName = evo)
      UNION
      SELECT MoveName
      FROM Moveset
      WHERE PokemonName IN (
         SELECT PokemonName
         FROM Evolution
         WHERE EvolutionName IN (
            SELECT PokemonName
            FROM Evolution
            WHERE EvolutionName = evo));
   RETURN resultset;
END;
$$
Language plpgsql;
```

Insert Moves Upon Evolution Trigger

When a Pokémon is inserted into the evolution table, we can display the moves of their previous evolution. This will streamline inserting values into the moveset table.

```
CREATE TRIGGER set_moves_upon_evolution
AFTER UPDATE ON Evolution
  FOR EACH ROW EXECUTE PROCEDURE(PokemonName);
```

Trainer Role

While this database is designed as a resource to be used by trainers, they have no insertion privileges anywhere in the database. Trainers are able to query from any tables they like, but are not given access to update anything.

Role Creation:

```
CREATE ROLE Trainer;
GRANT SELECT ON
Pokemon, Evolution,
Inhabitants, Location,
Team, GymLeader,
Move, Moveset,
Type, TypeCombo,
TypeStrength, TypeWeakness
TO Trainer;
```

User Creation:

```
CREATE USER Red;
CREATE USER Blue;
CREATE USER Joey;
GRANT Trainer TO Red, Blue;
```

Leader Role

Gym Leaders have all the privileges trainers have, except they are also allowed the update the GymLeader and Team tables. They are allowed to insert into Team, in case a new Pokémon is added to their team.

Role Creation:

```
CREATE ROLE Leader;
GRANT SELECT ON
Pokemon, Evolution,
Inhabitants, Location,
Team, GymLeader,
Move, Moveset,
Type, TypeCombo,
TypeStrength, TypeWeakness
TO Leader;
GRANT UPDATE ON
GymLeader
TO Leader;
GRANT INSERT, UPDATE ON
Team
TO Leader;
```

User Creation:

```
CREATE USER Brock;
CREATE USER Misty;
CREATE USER Surge;
GRANT Leader TO Brock, Misty, Surge;
```

Administrator Role

Administrators are the creators of the Pokedex – Professor Oak, Professor Elm, and Bill. They have full privileges over everything in the database.

Role Creation:

```
CREATE ROLE Administrator;
GRANT ALL PRIVILEGES ON ALL TABLES IN SCHEMA public TO Administrator;
```

User Creation:

```
CREATE USER ProfessorOak;
CREATE USER ProfessorElm;
CREATE USER Bill;
GRANT Administrator TO ProfessorOak, ProfessorElm, Bill;
```

Known Issues

The following Issues are known in this database:

- While the trigger for setting moves upon evolution properly queries all the necessary moves, it will not execute themselves. To fix this one would have to edit or create a new stored procedure that uses insert statements into the moveset table in addition to the queries already in place.
- Some gym leaders possess more than one Pokémon on the same team. However, in the team table, LeaderName and PokemonName are both primary keys, so duplicate instances are not allowed. For example, while Koga has two Koffings, the database only records that he has one because the record for the second Koffing would not be unique. In order to fix this, we would need to create an arbitrary primary key, perhaps TeamID, to remove the primary key from PokemonName in the team table.
- Leaders are able to edit information not only about themselves, but other gym leaders. In order to fix this each gym leader would need his or her own table.

Future Enhancements

There are many features that I would have liked to include in this database, but could not due to time limitations. The following are implementations that could be included in the future:

- In its current form, the database shows the weaknesses for each type. However, an advanced implementation could show the weaknesses for each Pokémon. Pokémon with more than one type must have the aggregate strengths and weaknesses balanced out in order to display a dynamic and accurate record of strengths/weaknesses for each Pokémon. For instance, a user who owns a Bulbasaur may query the grass type Weakness and see that ground is weak to grass. However, Bulbsaur's secondary typing, Poison, is weak to ground, making Bulbasaur take neutral damage from ground type moves.
- Another table that would be nice to be added in the future is an Abilities table. Pokémon can have up to 3 abilities each, each with their own specific effects.