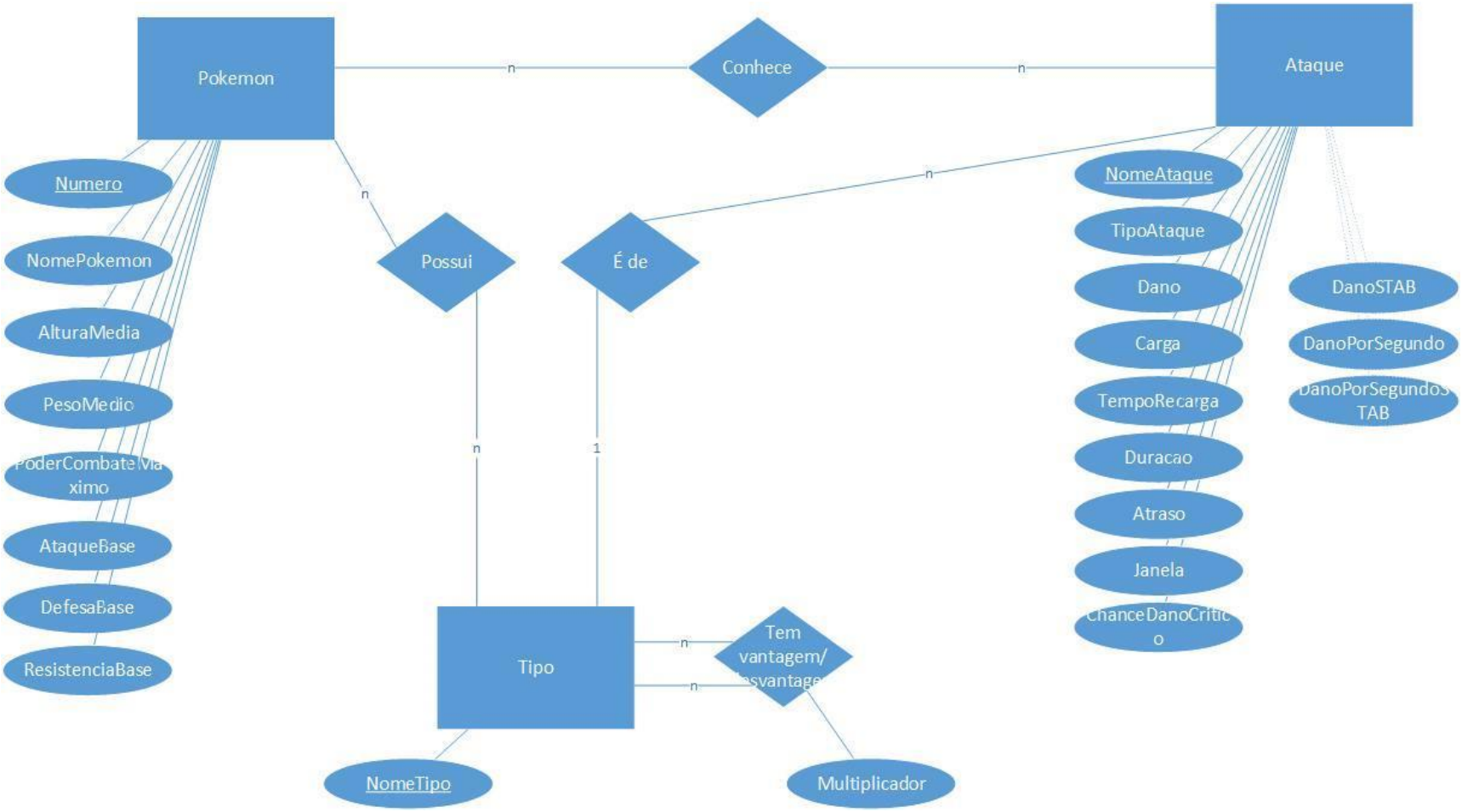
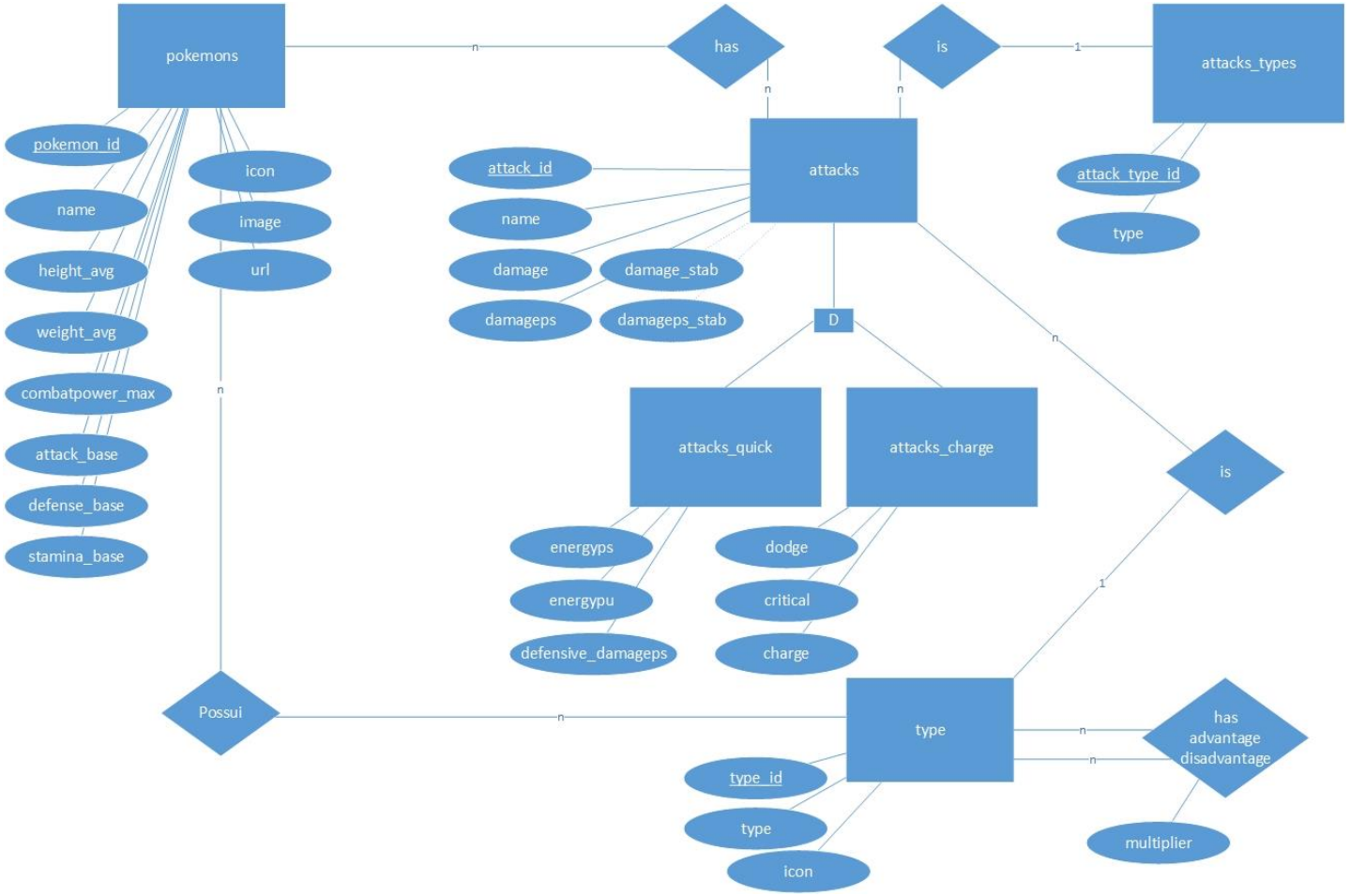
PoGym

DER-1


Pokemon
Conhece
Ataque
n
n
Numero
NomePokemon
AlturaMedia
PesoMedio
PoderCombateMaximo
AtaqueBase
DefesaBase
ResistenciaBase
n
Possui
É de
n
1
NomeAtaque
TipoAtaque
Dano
Carga
TempoRecarga
Duracao
Atraso
Janela
ChanceDanoCritico
DanoSTAB
DanoPorSegundo
DanoPorSegundoSTAB
Tipo
n
n
Tem vantagem/
NomeTipo
Multiplicador

DER-2


pokemons
has
is
attacks_types
pokemon_id
name
height_avg
weight_avg
combatpower_max
attack_base
defense_base
stamina_base
icon
image
url
attack_id
name
damage
damageps
damage_stab
damageps_stab
attacks
attack_type_id
type
D
attacks_quick
attacks_charge
energyps
energypu
defensive_damageps
dodge
critical
charge
is
Possui
type
type_id
type
icon
has advantage disadvantage
multiplier
n
1

DER-2

- Os nomes de atributos foram utilizados no seu original em inglês para facilitar o trabalho de mapeamento dos atributos da tabela com o script que lê os dados diretamente da página da gamepress, esta alteração facilitou também o trabalho de checagem dos dados no pós importação;
- Os nomes de tabelas servem como prefixos das relacionadas, para facilitar a compreensão. Por exemplo “Pokemons (pokemons)” e a relacionada “Tipos dos Pokemons (pokemons_types)”;
- Foi adotada uma nova formatação de nome de atributo:
  - Todo em minúsculas;
  - Com nomes sem abreviatura divididos por “_” para fins de tornar mais claro seu uso;
  - Chaves estrangeiras mantém o nome de atributo original da chave primária de origem para facilitar a localização;
- Foi feita a especialização da tabela de ataques em uma superclasse “ataque” (attack) e duas subclasses uma para “ataques rápidos” (attacks_quick) e outra para “ataques especiais” (attacks_charge);
- Atributos adicionais incluídos no site posteriormente foram adicionados com base em sua utilidade, por exemplo:
  - Atributos “charge” e “energypu” nas sub-classes de “attack”;
  - Atributos que guardam informação de ícone, imagem e url em entidades como “pokemons” e “types”;

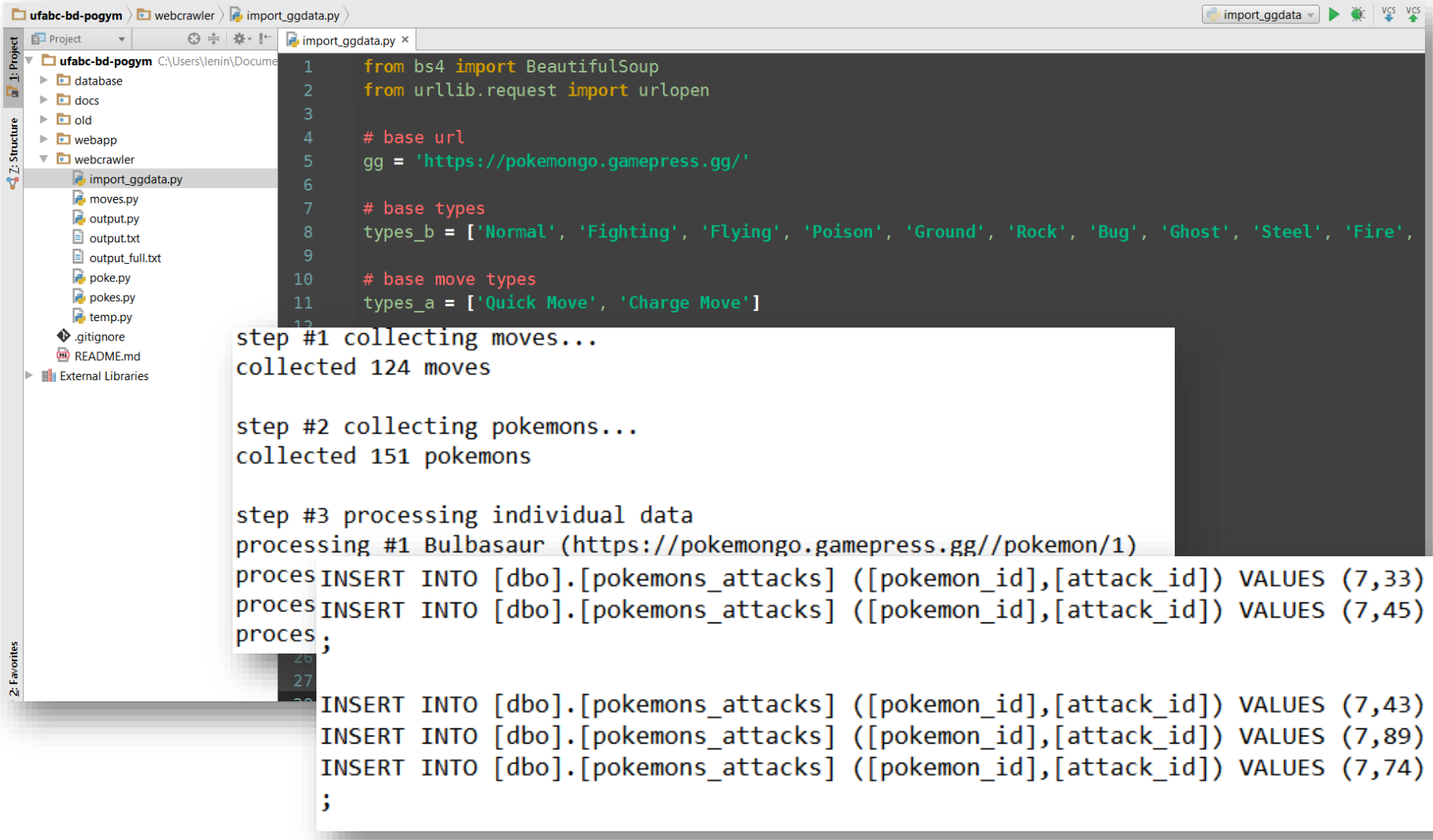
MER

- **attacks**(attack_id,attacks_types.attack_type_id,name,types.type_id,damage,damage_stab,damageps,damageps_stab,move_cooldown);
- **attacks_charge**(attacks.attack_id,charge,critical,dodge)
- **attacks_quick**(attacks.attack_id,energyps,energypu,defensive_damageps)
- **attacks_types**(attack_type_id,type)
- **pokemons**(pokemon_id,pokemons.evolvefrom_pokemon_id,name,icon,image,url,height_avg,weight_avg,combatpower_max,attack_base,defense_base,stamina_base)
- **pokemons_attacks**(pokemon_attack_id,pokemons.pokemon_id,attacks.attack_id,true_damageps)
- **pokemons_types**(pokemon_type_id,pokemons.pokemon_id,types.type_id)
- **types**(type_id,type,icon)
- **types_types**(type_type_id,types.attacker_type_id,types.defender_type_id,multiplier)

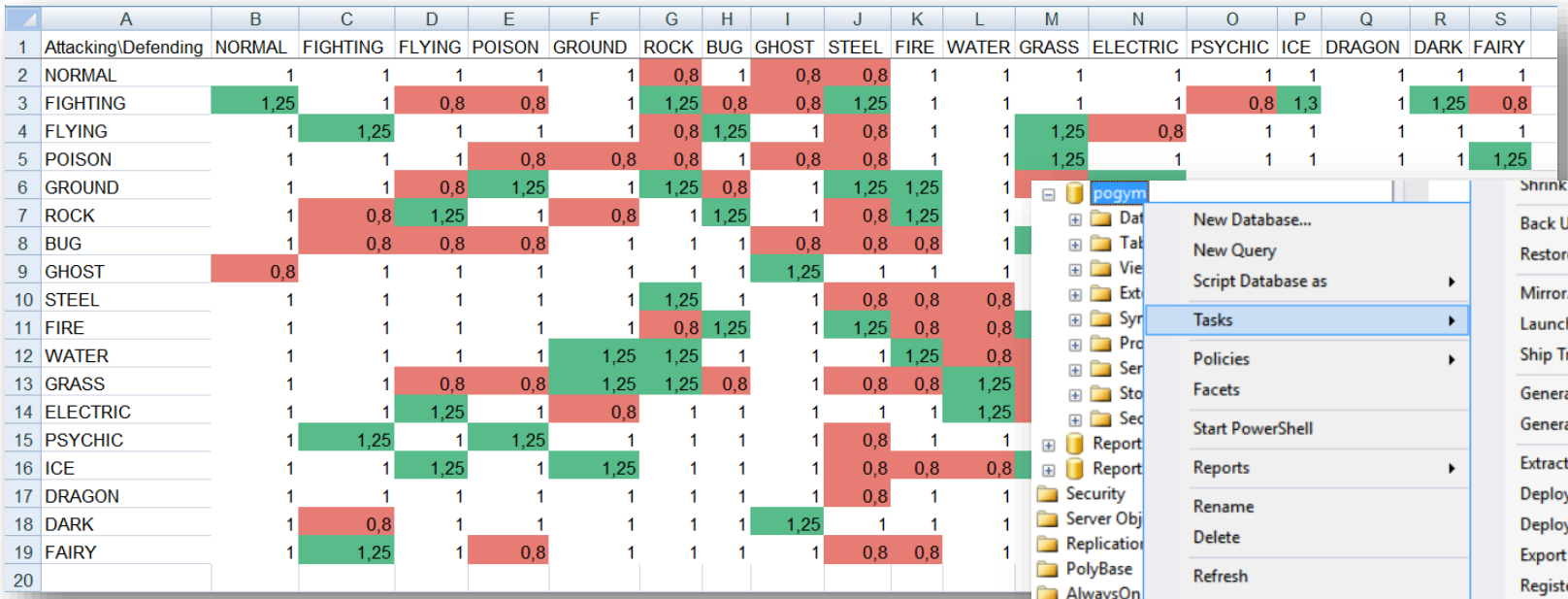
Normalização

- Ao analisar as entidades com relação a dependência funcional por exemplo, entendemos que os atributos nas tabelas não só eram funcionalmente dependentes da sua chave como no nosso entendimento não estavam presentes atributos funcionalmente dependentes de outros atributos não presentes nas chaves. Mesmo redundâncias e campos multivalorados presentes no primeiro DER foram eliminados no segundo.
- A tabela “Ataque (attack)” possuía um atributo chamado “Tipo de ataque” que poderia assumir dois valores: “Ataque rápido” ou “Ataque carregado” no primeiro diagrama. No segundo, essa redundância foi resolvida com uma nova entidade separada “Tipo de ataque (attacks_types)” e o atributo se tornou a chave estrangeira “attack_type_id”;
- A tabela “Ataque (attack)” possuía atributos utilizados somente em ataques rápidos, atributos utilizados somente em ataques carregados e atributos utilizados nos dois casos, um caso típico na nossa avaliação no qual poderíamos empregar a especialização, geramos assim duas subclasses da entidade “Ataque (attack)”: “Ataque rápido (attack_quick)” e “Ataque carregado (attack_charge)”;

Carga


ufabc-bd-pogym
webcrawler
import_ggdata.py
import_ggdata
Project
import_ggdata.py
ufabc-bd-pogym C:\Users\lenin\Docume
database
docs
old
webapp
webcrawler
import_ggdata.py
moves.py
output.py
output.txt
output_full.txt
poke.py
pokes.py
temp.py
.gitignore
README.md
External Libraries
1: Project
7: Structure
2: Favorites
1 from bs4 import BeautifulSoup
2 from urllib.request import urlopen
3
4 # base url
5 gg = 'https://pokemongo.gamepress.gg/'
6
7 # base types
8 types_b = ['Normal', 'Fighting', 'Flying', 'Poison', 'Ground', 'Rock', 'Bug', 'Ghost', 'Steel', 'Fire',
9
10 # base move types
11 types_a = ['Quick Move', 'Charge Move']
step #1 collecting moves...
collected 124 moves
step #2 collecting pokemons...
collected 151 pokemons
step #3 processing individual data
processing #1 Bulbasaur (https://pokemongo.gamepress.gg//pokemon/1)
INSERT INTO [dbo].[pokemons_attacks] ([pokemon_id],[attack_id]) VALUES (7,33)
INSERT INTO [dbo].[pokemons_attacks] ([pokemon_id],[attack_id]) VALUES (7,45)
;
INSERT INTO [dbo].[pokemons_attacks] ([pokemon_id],[attack_id]) VALUES (7,43)
INSERT INTO [dbo].[pokemons_attacks] ([pokemon_id],[attack_id]) VALUES (7,89)
INSERT INTO [dbo].[pokemons_attacks] ([pokemon_id],[attack_id]) VALUES (7,74)
;

Carga



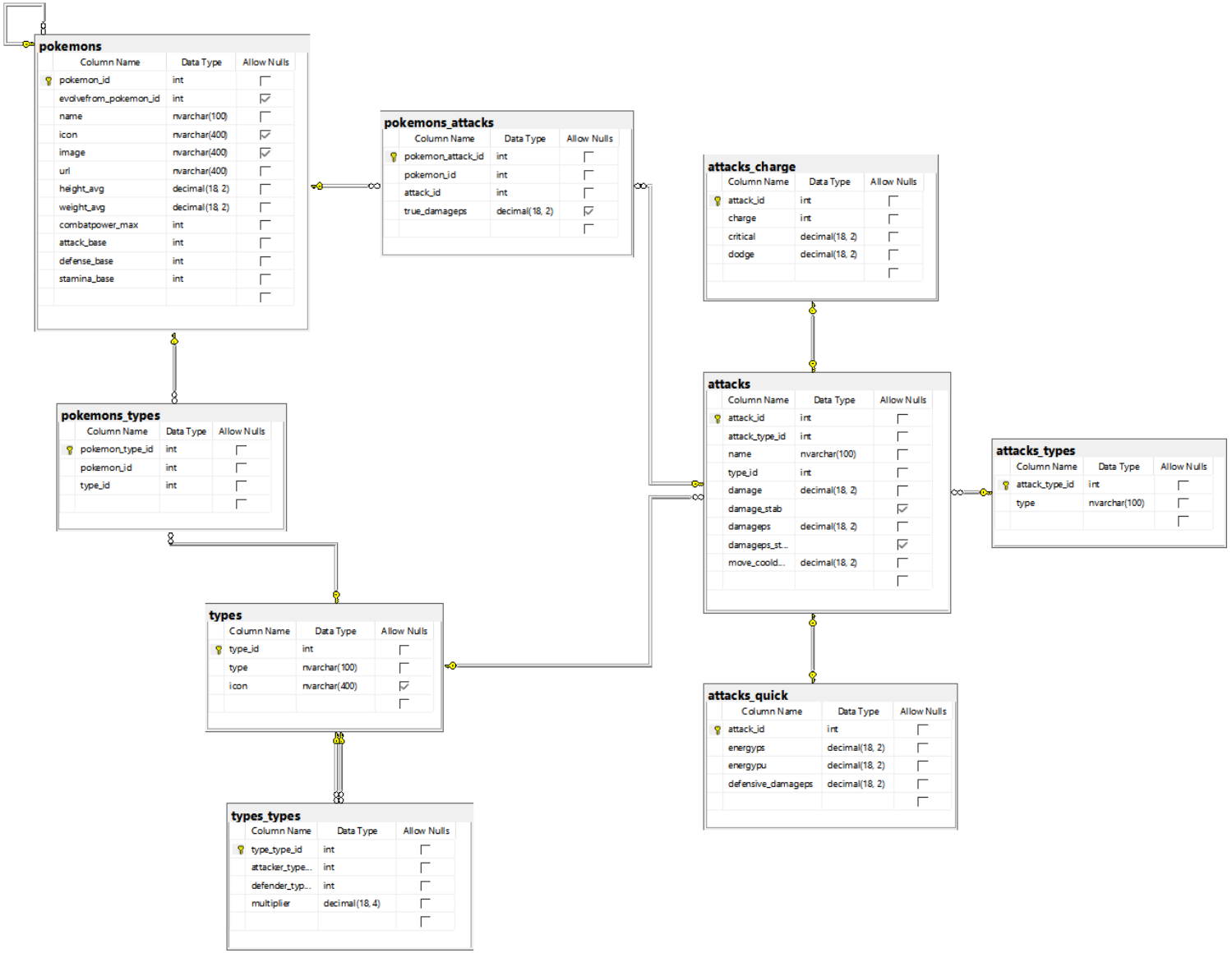
Attacking\Defending	NORMAL	FIGHTING	FLYING	POISON	GROUND	ROCK	BUG	GHOST	STEEL	FIRE	WATER	GRASS	ELECTRIC	PSYCHIC	ICE	DRAGON	DARK	FAIRY
NORMAL	1	1	1	1	1	0,8	1	0,8	0,8	1	1	1	1	1	1	1	1	1
FIGHTING	1,25	1	0,8	0,8	1	1,25	0,8	0,8	1,25	1	1	1	1	0,8	1,3	1	1,25	0,8
FLYING	1	1,25	1	1	1	0,8	1,25	1	0,8	1	1	1,25	0,8	1	1	1	1	1
POISON	1	1	1	0,8	0,8	0,8	1	0,8	0,8	1	1	1,25	1	1	1	1	1	1,25
GROUND	1	1	0,8	1,25	1	1,25	0,8	1	1,25	1,25	1	1						
ROCK	1	0,8	1,25	1	0,8	1	1,25	1	0,8	1,25	1							
BUG	1	0,8	0,8	0,8	1	1	1	0,8	0,8	0,8	1							
GHOST	0,8	1	1	1	1	1	1	1,25	1	1	1							
STEEL	1	1	1	1	1	1,25	1	1	0,8	0,8	0,8							
FIRE	1	1	1	1	1	0,8	1,25	1	1,25	0,8	0,8							
WATER	1	1	1	1	1,25	1,25	1	1	1	1,25	0,8							
GRASS	1	1	0,8	0,8	1,25	1,25	0,8	1	0,8	0,8	1,25							
ELECTRIC	1	1	1,25	1	0,8	1	1	1	1	1	1,25							
PSYCHIC	1	1,25	1	1,25	1	1	1	1	0,8	1	1							
ICE	1	1	1,25	1	1,25	1	1	1	0,8	0,8	0,8							
DRAGON	1	1	1	1	1	1	1	1	0,8	1	1							
DARK	1	0,8	1	1	1	1	1	1,25	1	1	1							
FAIRY	1	1,25	1	0,8	1	1	1	1	0,8	0,8	1							


```
select 'INSERT INTO [types_types] ([attacker_type_id],[defender_type_id],[multiplier])
 VALUES ('+convert(nvarchar(50),id)+',1,'+convert(nvarchar(50),Normal)+')'
from TypesMultiplier

union

select 'INSERT INTO [types_types] ([attacker_type_id],[defender_type_id],[multiplier])
 VALUES ('+convert(nvarchar(50),id)+',2,'+convert(nvarchar(50),Fighting)+')'
from TypesMultiplier
```

Modelo Físico - SQL Server



Queries

Queries

```
6  SELECT t.type_id as id,t.type as tipo
7  FROM types as t
8  ORDER BY id
9  go
```

Results | Messages

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

$t \leftarrow types$

$R(id,tipo) \leftarrow \Pi_{type\_id,type}(t)$

τ id asc (R)

Queries

```
17 declare @id int = 1
18
19 select t.type as tipo
20 from pokemons_types as pa inner join types as t on pa.type_id = t.type_id
21 where pokemon_id = @id
22
23 select p.pokemon_id as pokedex,p.name as nome,weight_avg as peso,height_avg as altura
24       ,p.attack_base as atk,p.defense_base as def,p.stamina_base as sta
25 from pokemons as p
26 where p.pokemon_id = @id
27 go
```

$@id \leftarrow 1$

$pa \leftarrow pokemons\_types$

$t \leftarrow types$

$tipo \leftarrow \Pi_{type}(t)$

$\sigma_{pokemon\_id=@id}(pa \bowtie pa.type\_id = t.type\_id\ t)$

$p \leftarrow pokemons$

$R(pokedex, nome, peso, altura, atk, def, sta) \leftarrow \Pi_{pokemon\_id,\ name,\ weight\_avg,\ height\_avg,\ attack\_base,\ defense\_base,\ stamina\_base}(p)$

$\sigma_{p.pokemon\_id=@id}(R)$

Results | Messages

	tipo
1	Grass
2	Poison

	pokedex	nome	peso	altura	atk	def	sta
1	1	Bulbasaur	6.90	0.70	126	126	90

Objetos

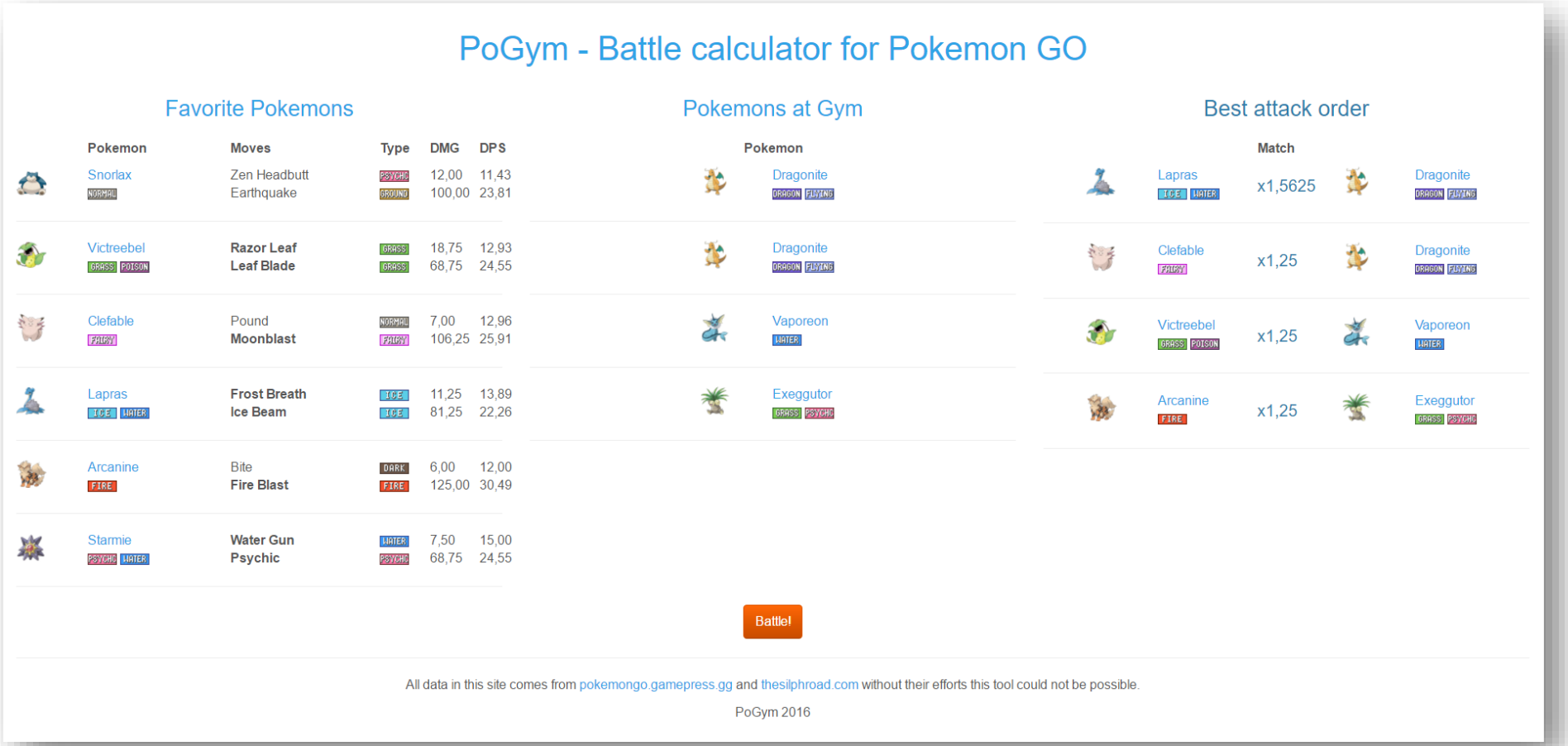
```
14 ALTER PROCEDURE [dbo].[sp_battle]
15     @type1 int = 0
16    ,@type2 int = 0
17 AS
```

```
13 ALTER TRIGGER [dbo].[attacks_charge_disjoint]
14    ON [dbo].[attacks_charge]
15    AFTER INSERT
16 AS
17 BEGIN
18     SET NOCOUNT ON;
19
20 declare @id int = (select i.attack_id from inserted as i)
21
22 IF EXISTS (SELECT *
23             FROM dbo.attacks_quick as a
24             WHERE a.attack_id = @id)
25 BEGIN
26     RAISERROR ('This attack already exists on ''Quick Attacks'' table.', 16, 1);
27     ROLLBACK TRANSACTION;
28     RETURN
29 END
30
31 END
```

```
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31     RETURN @out
32 END
```

Web App

Web App



https://github.com/lnncrs/PoGym