CS6083-A Final Project Part2

Principles of Database Systems, Fall 2021

Ryan Cheng(kc4532) Wei-Neng Hsu(wnh215)

Note: It is run on wnh215@jedi.poly.edu (mailto:wnh215@jedi.poly.edu) with port 8618

Business Rules

We create a database that stores some useful information in the anime Pokemon in the final project.

There are seven entity sets and ten relationship sets in our E-R model.

Pokemon

The main character, Pokemon, will have its id (positive integer), name (string), hit point (HP, positive integer), attack (Atk, positive integer), defense (Def, positive integer), special attack (Sp. Atk, positive integer), special defense (Sp. Def, positive integer) and speed (positive integer). All of these cannot be null.

Skill

Every Pokemon will learn at least one skill, and every skill will be learned by at least one Pokemon, we need to meet a requirement (string) when Pokemon learn a skill. Every skill will have its name (string), chance to hit (positive integer), and damage(positive float). Chance to hit and damage may be null since some skills will not do damage to enemies.

Type

There are eighteen types in the world of Pokemon, every skill should have exactly one type, and every Pokemon should have one or two types. The type has its name (string) and some types are not introduced in the very beginning, so we will record its introduced generation.

Damage

Every type has its advantages and disadvantages, like an electricity type attack will do no damage to ground type, so will be 0x in this case, we will also record this in one relationship set.

Evolve

Some Pokemon will evolve. One Pokemon may evolve to different Pokemon in the same or different requirements, but one pokemon cannot be evolved from different Pokemon. Some Pokemons have different types, so we will also record what is the basic type of it. Therefore, for every pokemon, there is at most one base type.

Episodes

Pokemon cartoons make Pokemon more interesting! There are hundreds (or even thousands) of episodes so far, each episode has its id (positive integer), generation(positive integer), and broadcast date (date). In every episode, some Pokemon will appear. Since data may become too complex, we will only record the first episode of every Pokemon that appears. However, not every Pokemon appears in at least one episode. Each episode will be only played in one place.

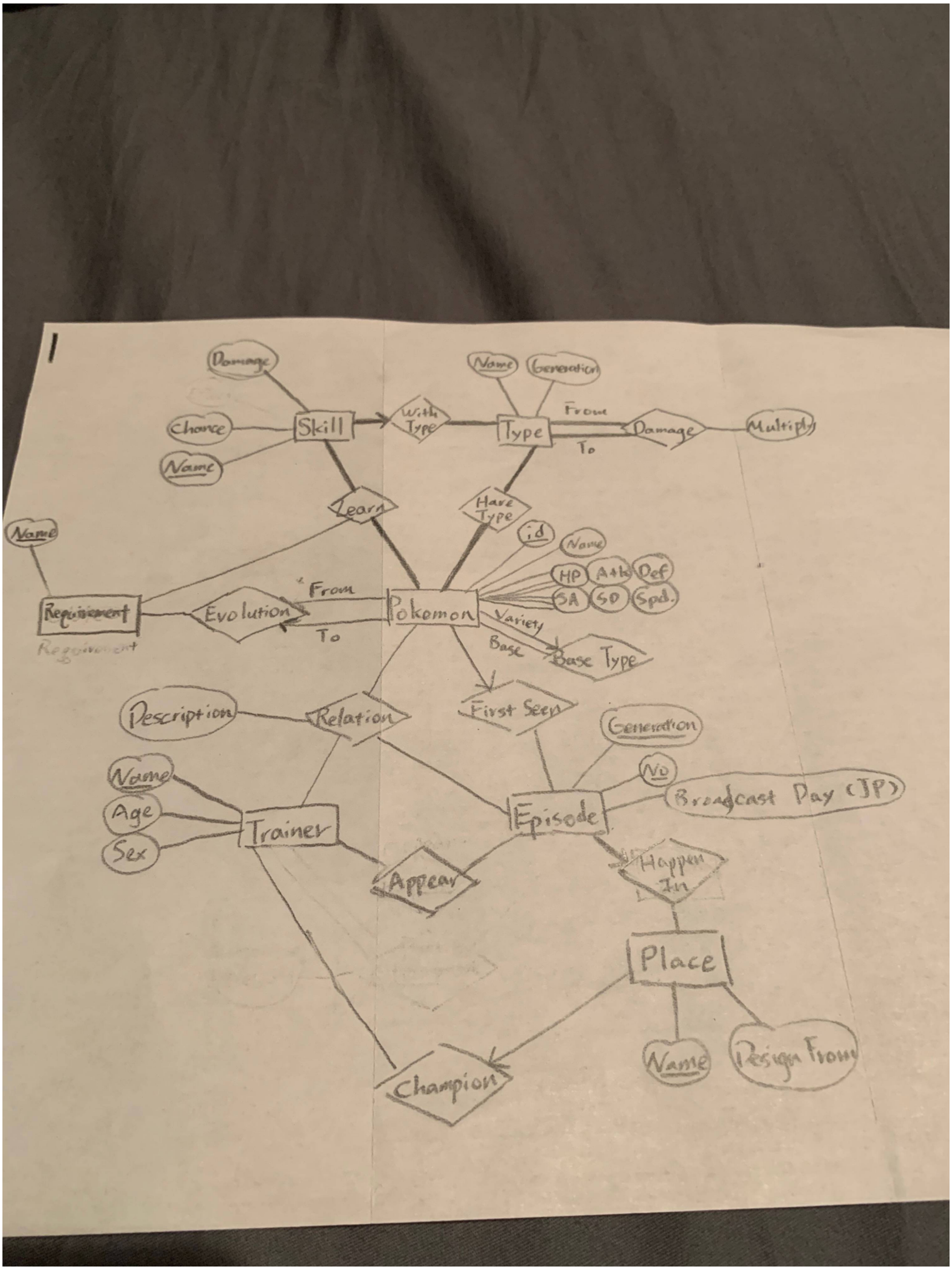
Trainer

Another important thing in Pokemon cartoons is the trainer. The trainer has its name (string), age (positive integer), and sex (string). A trainer will have some relations with some Pokemons in some episodes, the relations can be owning the Pokemon or releasing the Pokemon, to make it simpler, if a Pokemon evolves, we will record it as releasing the old one and owning a new one. We will also record if a character appears in an episode. Some episodes may have no recorded trainer participate in, and some trainers never appeared on the cartoons (for example, they only appear on video games).

Region

These cartoons take place in some virtual world, and every region in the world has its name(string). Each region is designed from a real place in reality. Each episode plays in exactly one region. Each region has no or one champion, and the champion is a trainer.

E-R diagram



Data Acquisition

While we can’t watch every episode and play every Pokemon game to collect information, here are some websites that contain data that is useful for us. Like “Pokemondb” has some data like skill list and Pokemon list. We can find how Pokemon learn a skill from the website. “Bulbapedia” has the data of every episode, like which character is played, and which Pokemon is participating. It may still be something enormous, so we may only record the data for some parts of Pokemon and some parts of Pokemon anime. (Like only data in the first generation)

We have put our data on Google sheet to have it become easier to cooperate. We have designed a ahell script `xlsx2csv.sh` to transfer data from `xlsx` to `csv` (We should install `xlsx2csv` first) and a shell script `csv2database.sh` to load these data into database server. You may need to change some configuration in this shell script to change the destination database to your own db.

User Interaction

We have designed several interactions in our project.

Read Table

Input: None
Output: Every table in database
This is an interaction that moved from the project demo in class. This table can enable the user to know about the table and give us a chance to know whether the program runs in the right way.

Pokemon with Skills

Input: Skill name
Output: Pokemons with that skill
This interaction will return a series of pokemon and requirements which can learn a specific skill.

Pokemon Evolution

Input: Pokemon
Output: Pokemon after evolving with their data
This interaction will return the Pokemon that the specific pokemon evolve to. The output will print with those pokemons’ data.

Analyzing different types in customized data

Input: Minimum requirements for Pokemon
Output: Every Pokemon that match the requirements
This interaction allows users to decide the minimum requirements for a Pokemon, like the minimum HP or minimum species strength, and our program will return those Pokemons to match these requirements.

What will the type be multiplied when encountering enemy

Input: type name, Pokemon
Output: Damage multiply to that Pokemon
This interaction will return the number that should be multiplied when a specific type of skill hits a Pokemon. It will be achieved in two SQL command because Pokemon will have one or two types.

What skill should you learn to beat an enemy

Input: My Pokemon and enemies Pokemon
Output: The best skill to beat enemies Pokemon
This interaction is an extension of the previous interaction. We will combine the previous interaction with the skill which my Pokemon can learn, and choose the best skill from those skills. Since some Pokemons won’t learn a skill, those pokemon won’t be available to choose from in this part.

Duel of champions

Input: Two different champions then two Pokemon that owned by them
Output: Battle result after one round
This interaction will implement a round of battle between two champions. This interaction can give users a brief view of how the battle performed (Although it is simpler than the real battle). Pokemon with higher $HP + Def$ will win after the round.

Who would you choose???

Input: The first pokemon chosen
Output: Analyze your choice
This interaction will help you to analyze the choice we made in choosing pokemon. It will print out the data of the potential ability of the Pokemon, and how good it is after evolution.

SQL code

```sql
drop table if exists Skill_with_Type cascade;
drop table if exists Type cascade;
drop table if exists Damage cascade;
drop table if exists Pokemon_first_seen cascade;
drop table if exists Pokemon_have_type cascade;
drop table if exists Requirement cascade;
drop table if exists Pokemon_learn_skill cascade;
drop table if exists Pokemon_evolve cascade;
drop table if exists Pokemon_base_type cascade;
drop table if exists Place_champion cascade;
drop table if exists Episode_happen_in cascade;
drop table if exists Trainer cascade;
drop table if exists Pokemon_Episode_Trainer cascade;
drop table if exists Trainer_Appear cascade;

create table Type(
        Name varchar(42) primary key,
        Generation integer not null
);

create table Trainer(
        Name varchar(42) primary key,
        Age integer,
        Gender varchar(42)
);

create table Skill_with_Type(
        Name varchar(42) primary key,
        Chance integer,
        Damage integer,
        Type_name varchar(42) not null,
        foreign key (Type_name) references Type(Name)
);

create table Damage(
        Type_name_from varchar(42) not null,
        Type_name_to varchar(42) not null,
        Multiply float not null,
        primary key (Type_name_from, Type_name_to),
        foreign key (Type_name_from) references Type(Name),
        foreign key (Type_name_to) references Type(Name)
);

create table Place_champion(
        Name varchar(42) primary key,
        Design_from varchar(42),
        Champion varchar(42),
        foreign key (Champion) references Trainer(Name)
```

```
);

create table Episode_happen_in(
        Generation integer,
        Number  integer,
        Broadcast Date not null,
        Happen_in varchar(42),
        primary key    (Generation, Number),
        foreign key (Happen_in) references Place_champion(Name)
);

create table Pokemon_first_seen(
        id integer primary key,
        Name varchar(42) not null unique,
        HP integer  not null,
        Atk integer not null,
        Def integer not null,
        Special_Atk integer not null,
        Special_Def integer not null,
        Speed integer not null,
        Fisrt_seen_generation integer,
        Fisrt_seen_no integer,
        foreign key (Fisrt_seen_generation, Fisrt_seen_no) referen
);

create table Pokemon_have_type(
        id integer,
        Type_name varchar(42),
        primary key (id, Type_name),
        foreign key (Type_name) references Type(name),
        foreign key (id) references Pokemon_first_seen(id)
);

create table Requirement(
        Name varchar(42) primary key
);

create table Pokemon_learn_skill(
        id integer,
        Skill_name varchar(42),
        Requirement_name varchar(42),
        primary key (id, Skill_name, Requirement_name),
        foreign key (Skill_name) references Skill_with_Type(name),
        foreign key (id) references Pokemon_first_seen(id),
        foreign key (Requirement_name) references Requirement(name
);

create table Pokemon_evolve(
        Evolve_from integer,
        Evolve_to integer,
        Requirement_name varchar(42),
        primary key (Evolve_from, Evolve_to,Requirement_name),
        foreign key (Evolve_from) references Pokemon_first_seen(id
        foreign key (Evolve_to) references Pokemon_first_seen(id),
        foreign key (Requirement_name) references Requirement(name
);

create table Pokemon_base_type(
        Base integer,
        Variety integer,
        primary key (Base, Variety),
        foreign key (Base) references Pokemon_first_seen(id),
        foreign key (Variety) references Pokemon_first_seen(id)
);

create table Pokemon_Episode_Trainer(
        Pokemon_id integer,
        Episode_generation integer,
        Episode_Number integer,
        Trainer_Name varchar(42),
        primary key(Pokemon_id, Episode_generation, Episode_Number
        foreign key (Pokemon_id) references Pokemon_first_seen(id)
        foreign key (Trainer_Name) references Trainer(Name),
        foreign key (Episode_generation, Episode_Number) reference
);

Create table Trainer_Appear(
        Trainer_name varchar(42),
        Episode_generation integer,
        Episode_number integer,
        primary key(Trainer_name, Episode_generation, Episode_numb
        foreign key (Trainer_Name) references Trainer(Name),
        foreign key (Episode_generation, Episode_number) reference
);

```