Pokemon Trainer Database

Kaitlyn Lawlor, Zachary Purdy

Table of Contents

Definition of Business Problem

Pokémon Trainer Database
Avid players of Pokémon could use a Pokémon Trainer database in order to help them play the game by giving them useful information of the names of the important trainers, what games they are from, and the type of Pokémon they use.

****
****

Since Pokémon GO was released this past Summer, the Pokémon craze is bigger now than it has been in recent time. Sales of merchandise is up, old trading cards are being reprinted, and with two new games on the horizon, gamers will be reliving their childhoods all over again and playing through their old games while they await the new releases. Trainers who are a little more serious than others will need all the help they can get to grind through each generation on their quest to become a Pokémon master.

The Pokémon Trainer Database will be a useful tool to aid gamers in beating the toughest opponents in the games. Essentially, it is a Pokédex of people, rather than Pokémon. Anyone can go to a popular website such as Serebii or Bulbapedia, but navigating through a fan site could be rather cumbersome for a more casual player. This database should provide an extremely quick and easy solution to questions as by a user to prepare for an upcoming battle.

The database will include entities such as:

- Gym leaders
- The Pokémon that are a part of each trainer's teams
- Pokemon Types
- Regions and towns
- Each game and generation

Questions a user of the database could potentially ask:

- List the 8 Kanto gym leaders
- What games are Maylene in?
- List all the trainers that use Bug type or Water type
- Who are all the trainers with last names?
- Any name starting with F
- How many gym leaders are in Gen IV?
- Who are the Gym Leaders of Gen VI and what are Pokemon Type do they possess?
- Return the Pokemon Types from most used to least used
- What Pokemon types have the highest frequency?
- What are the Town Names in Generation II?
- List all the trainers with a town, if they have one

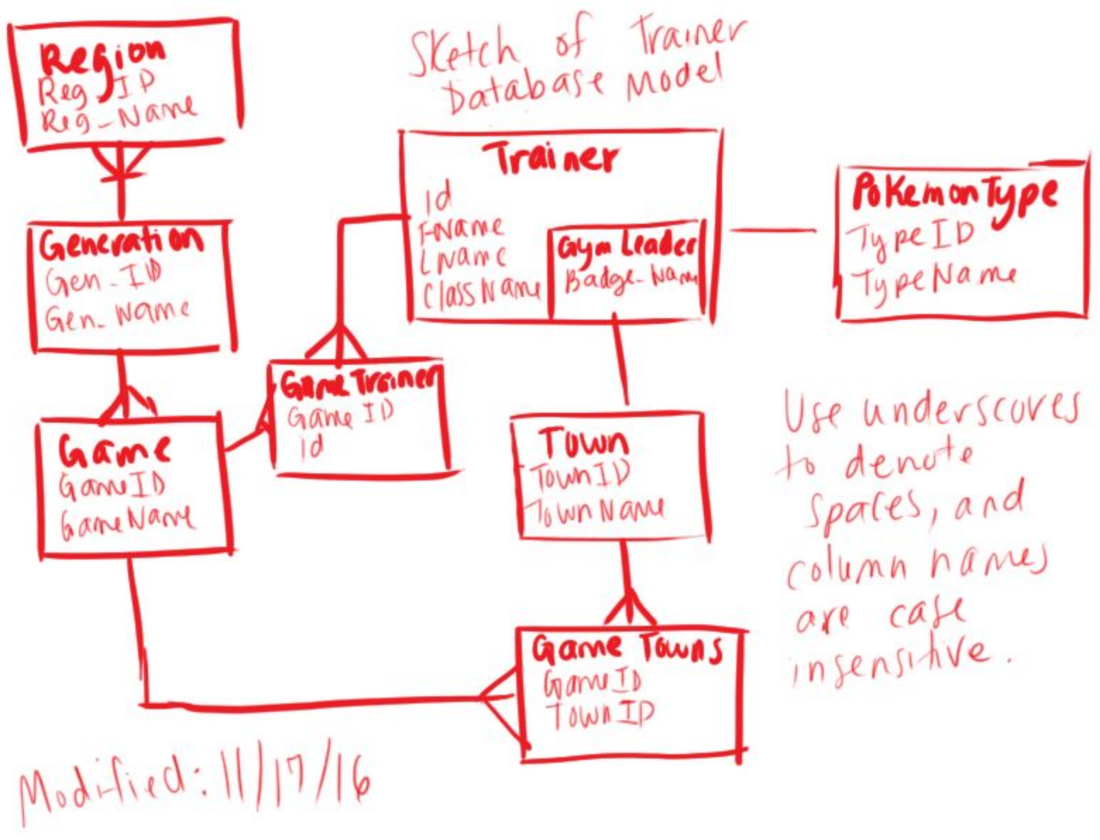
Business Rules:

1. The primary goal here is to create a database of the most important trainers in the Pokemon handheld games. There are 3 types of trainers we are concerned about (Gym leaders, Elite 4, and Champion). However, due to the scope of this, we are going to limit to just gym leaders and expand to Elite 4 and Champions if time allows it this semester.
2. Each Generation of Pokemon games has multiple games. Each of these games can differ in some aspects, so there needs to be a way to distinguish exactly which game each trainer is a part of.
3. Every trainer is skilled in a certain type of Pokemon. Therefore, each trainer must have a pokemon type associated with them in the database. For example, it would be good to be able to know all the trainers that specialize in fire types and where they are from.
4. The gym leaders each "rule over" a town by themselves. The town they are from should be known.

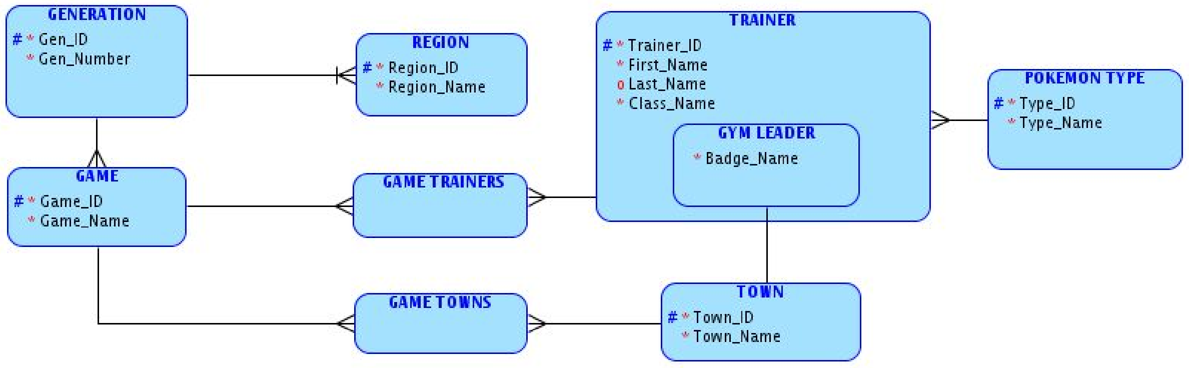
Trainer: Trainer ID (Primary Key), Trainer Name

Gym Leaders: Should have knowledge of the Trainer ID
Knowledge of what **Region** they are from.
Knowledge of what **City** they are from.
Knowledge of what **Badge** they possess.

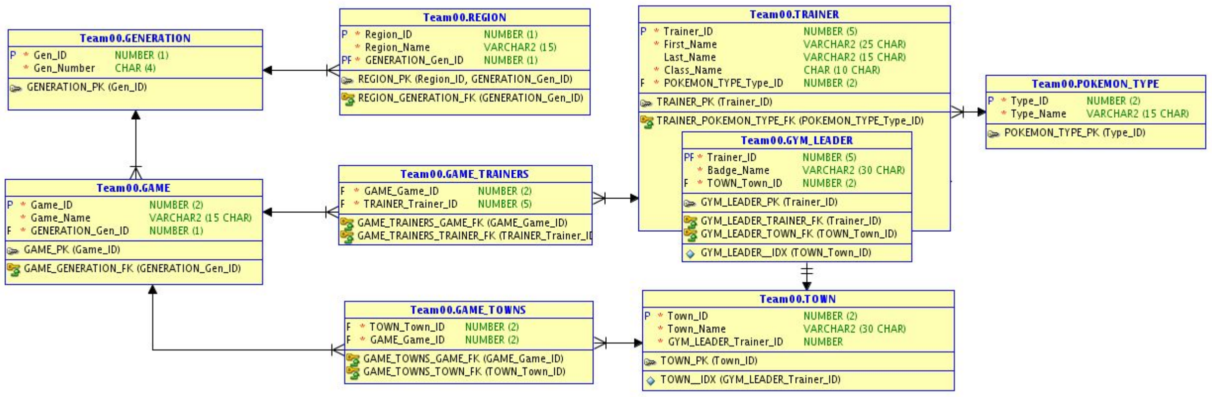
Outline of Model



Logical Model



Relational Model


Team00.GENERATION
P * Gen_ID NUMBER (1)
* Gen_Number CHAR (4)
GENERATION_PK (Gen_ID)
Team00.REGION
P * Region_ID NUMBER (1)
* Region_Name VARCHAR2 (15)
PF * GENERATION_Gen_ID NUMBER (1)
REGION_PK (Region_ID, GENERATION_Gen_ID)
REGION_GENERATION_FK (GENERATION_Gen_ID)
Team00.TRAINER
P * Trainer_ID NUMBER (5)
* First_Name VARCHAR2 (25 CHAR)
Last_Name VARCHAR2 (15 CHAR)
* Class_Name CHAR (10 CHAR)
F * POKEMON_TYPE_Type_ID NUMBER (2)
TRAINER_PK (Trainer_ID)
TRAINER_POKEMON_TYPE_FK (POKEMON_TYPE_Type_ID)
Team00.POKEMON_TYPE
P * Type_ID NUMBER (2)
* Type_Name VARCHAR2 (15 CHAR)
POKEMON_TYPE_PK (Type_ID)
Team00.GYM_LEADER
PF * Trainer_ID NUMBER (5)
* Badge_Name VARCHAR2 (30 CHAR)
F * TOWN_Town_ID NUMBER (2)
GYM_LEADER_PK (Trainer_ID)
GYM_LEADER_TRAINER_FK (Trainer_ID)
GYM_LEADER_TOWN_FK (TOWN_Town_ID)
GYM_LEADER__IDX (TOWN_Town_ID)
Team00.GAME
P * Game_ID NUMBER (2)
* Game_Name VARCHAR2 (15 CHAR)
F * GENERATION_Gen_ID NUMBER (1)
GAME_PK (Game_ID)
GAME_GENERATION_FK (GENERATION_Gen_ID)
Team00.GAME_TRAINERS
F * GAME_Game_ID NUMBER (2)
F * TRAINER_Trainer_ID NUMBER (5)
GAME_TRAINERS_GAME_FK (GAME_Game_ID)
GAME_TRAINERS_TRAINER_FK (TRAINER_Trainer_ID
Team00.GAME_TOWNS
F * TOWN_Town_ID NUMBER (2)
F * GAME_Game_ID NUMBER (2)
GAME_TOWNS_GAME_FK (GAME_Game_ID)
GAME_TOWNS_TOWN_FK (TOWN_Town_ID)
Team00.TOWN
P * Town_ID NUMBER (2)
* Town_Name VARCHAR2 (30 CHAR)
* GYM_LEADER_Trainer_ID NUMBER
TOWN_PK (Town_ID)
TOWN__IDX (GYM_LEADER_Trainer_ID)

http://i.imgur.com/UtEd3Ry.png (for a larger scale)

Physical Model Considerations

For a physical model, some Performance considerations that would need to be taken into account are to make sure that the Database is normalized. The data portion of the class hierarchy needs to conform to the rules for entities in an E-R model. The model needs to be able to fit the guidelines of the Business Rules.

Another performance consideration would be Primary Keys being integers instead of character types.

Query performance can be improved by including indexes, which we have included.

Merging tables is also a way to increase performance.

Link to Data Dictionary

https://docs.google.com/spreadsheets/d/1O95qdlpKIX4pGNu-0YRTIpFSo9Ua--yOEk590CrPwyk/edit?usp
=sharing

This is our data dictionary.

Link to Data

https://docs.google.com/spreadsheets/d/1oiGVuhRb7TARd-JFXVI-7vrrTjn94Q0Hv00Ix6Gs604/edit?usp=
sharing

Modified from the Data Dictionary.
The data has been formatted into multiple Spreadsheets.
Trainer IDs have been assigned to all the Gym Leaders.
Additional Unique IDs have been assigned to *each* TOWN, REGION, and GENERATION.

Link to Implemented SQL Statements

https://docs.google.com/document/d/1iv2ZtBu9etej5XMSCaxNfQqtTRiI5DtijlxSNtK1Uww/edit?usp=sharin
g

The statements that populate our tables.

Link to Implemented SQL Queries

https://docs.google.com/document/d/11xDzs8juoDvvRBYcAbHOtiSuYaDLbjAxEWN5Z_ajnA0/edit?usp=s
haring

These are the SQL statements that answer our questions.

Changelog

12/04/2016

- Removed '/' from Game Trainer table
- Fully populated tables with data.
- Having trouble with working on each other's files. We realized that the Team00 schema should be added to the tables but would not work on Kaitlyn's version properly. Also, the competing 1 to 1 relationships with Gym Leader and Town had to stay as the queries were working with it. We didn't want to break our code last minute before submission. On Zack's version, these changes worked, however the tables weren't being populated.
- Despite these last minute obstacles, we are happy with what we are turning in.

11/17/2016

- Changed Gen Number precision to 4 from 2.

11/15/2016

Delivery 3 of the project was to finish up the Physical model, tying up any loose ends.

- Removed MS Paint diagram from design document. This was our original model. After undergoing many changes, it no longer reflects what we have in SQL Developer.
- The data has been formatted into multiple Spreadsheets, making it easier to create Insert statements.
- Trainer IDs have been assigned to all the Gym Leaders.
- Additional Unique IDs have been assigned to *each* TOWN, REGION, and GENERATION.
- **https://docs.google.com/spreadsheets/d/1oiGVuhRb7TARd-JFXVl-7vrrTjn94Q0Hv00lx6Gs604/edit?usp=sharing Spreadsheet of Data**
- https://docs.google.com/document/d/1dnC1pKuy_6FUp1v_SfnfrXGW8tSwKktH4iZ-HKUOKIM/edit?usp=sharing **Insert Statements (will be being updated)**
- We know that we cannot use a backslash in the Entity names, and we know that we need to change it.

10/23/2016

After meeting with Professor DePratti, this changelog was developed to help streamline our model. Two big changes stand out. How Generation, Game, and Region relate to each other before connecting to Trainer, and the removal of the Elite 4 and Champion. To properly implement Elite 4 would mean representing a group of people and not just individuals. This would complicate our database building goal and be too ambitious for the time we have this semester. If time allows it, we will incorporate more trainer types into the model. We will probably work on the changes together.

- Remove Elite 4 and Champion from Trainer supertype. It's added complexity that we don't need. Focusing on Gym Leader for now.

- Remove Game/Region associative entity.
- Remove Game subtype from Generation entity.
- From the last two changes, make Generation a one to many relationship to Region. Doing this will help represent that in each generation there is a unique region, and 3 games are produced for each generation.
- Also make Generation a one to many non-identifying relationship to Games.
- Game and Trainer would be a many to many relationship that will now require an associative entity.
- For Trainer entity, apply specialization discriminator.
- For Pokemon Type, use a code table to help with associating a Pokemon type to each trainer.
- Add more attributes to each entity. Right now almost exclusively PKs and something to identify instance, like a name. Still need to incorporate badges, Pokemon types, game versions, etc.

In the model, we also corrected the current PKs and changed the relationships from optional to mandatory. However, some of these are going to be altered since we are now going to shift our entities around. Moving forward, we now know to make sure whether the relationships are supposed to be optional or mandatory and to double check the flow of the PKs and FKs. Also, the numeric attribute values had a precision value rather than a scale value. Those were corrected as well. Now we know precision is for decimal values.