Logical Database Design

I. Requirements

a. This database shall be able to store all pertinent information of the “pokemon”
   i. “pokemon” data
   ii. Moves the “pokemon” may learn
   iii. Type effectiveness
   iv. A sprite to identify a visual for the “pokemon”
   v. Evolution sets that relate specific “pokemon”

b. The data stored shall be protected from redundant entries and transparent dependencies through the use of Primary Keys, Foreign Keys, and the separation of data into various tables

c. Data relevant to a specific “pokemon” can be viewed by the use through the use of table relations in an SQL query

Deferred Requirements

d. The database will be remotely accessed via a mobile application

e. The end-user will use the mobile application to access a web-server that will control database queries

f. Only the web-server scripts will directly access the database, under a user that has restricted privileges

g. The mobile application will call PHP scripts via WWW, and the PHP script will return values back to the application

h. The mobile application will display the database query results and database navigation through the use of a GUI

i. The GUI shall be able to do a web service request without halting the entire application

II. Use Cases

a. The user shall be able to call views to navigate the database
   i. A view that shows all possible “pokemon”
   ii. A short view of “pokemon” data
   iii. A long view of “pokemon” data
   iv. A view of moves
   v. A view of effectiveness for a give type against a given “pokemon”

b. The user will be able to “select” a view with parameters given from a previous view
   i. User uses view (a.i.) to show a listing of “pokemon”, and then “selects” to use view (a.ii) to show data, with parameter from “pokemon” list to uniquely identify which data to pull (as a foreign key)
   ii. User uses view (a.iii.) to show detailed view of “pokemon” data, and can include secondary parameter of given type, and using current pokemon and secondary parameter, the user “selects” to view (a.v.) effectiveness derived from secondary parameter and current “pokemon” type

Deferred Use Cases (In completion with deferred requirements)

c. The mobile application shall interactively show the navigation of the database to the user through a GUI

        i. A list of “pokemon” will be shown as a list showing minimal data to identify

    d. The application will emulate the “selection” to change navigation through the use of some interactive element (button/point and click)
        i. The user scrolls down the list and clicks on No.004 of the “pokemon” list, the GUI then “selects” (as previously described by Use Case (b.i.), taking the user to a new screen showing the query results
    e. The application will complete database queries via the use of PHP scripts, store the PHP script results in the application to show back to the user, never itself making an SQL query, only emulating it
        i. The user “selects” the “pokemon,” (Use Case d) and hands the PHP script an argument of value No.004 to specify which unique “pokemon” to view. The PHP script will execute a connect to the database, select either a predefined view or construct its own projection via an SQL SELECT using an SQL query, and return values back to the application. The application will then repopulate the GUI with a new outlay to accommodate the results, and populate the GUI with the results

III. Tasks (Use of Application)

a. The user should be able to lookup all data relevant to a “pokemon” as long as it exists within the database
b. The user should be able to lookup all moves a “pokemon” may learn, or inversely see all “pokemon” that can learn a specific move
c. The user should be able to see the effectiveness of a type (either user input or derived from a move type) against a pokemon.
d. The user shall only use the database to lookup “pokemon” data, and cannot input or modify data
    i. Deferred: Make the database support user login to track users and allow users to input data corresponding to their account, thus protecting the main data in the database by separating user data and main data