Daniel Reyes
Josue Loya

DinoDatabase: CSE 111 Final Project Report

Project Description

Our project database revolves around dinosaurs, in which we create a database that has both general and specific information about dinosaurs. Data includes: fossil location, time period, taxonomy, physical traits, and more. The database along with the attributes were created by hand. This proved to be a bit of a tough task, but we decided to do it this way in order to be as accurate as we can be. The sources used vary from books to national history websites. The database is accompanied by two java files, one named **Dino.java,** that contains the jdbc code and the other named **DinoQueries.java,** containing strings of all the queries that are used and called upon in the main file, data modification functions, and the code used for the UI. The program gives 3 different users the ability to interact with the database in different ways. A student or regular user will not have access to a certain menu that both an expert and admins will have. In addition, the expert will not have the full access that the administrator will have. The specifics of these privileges are shown in the user case diagram in **Figure 1**.

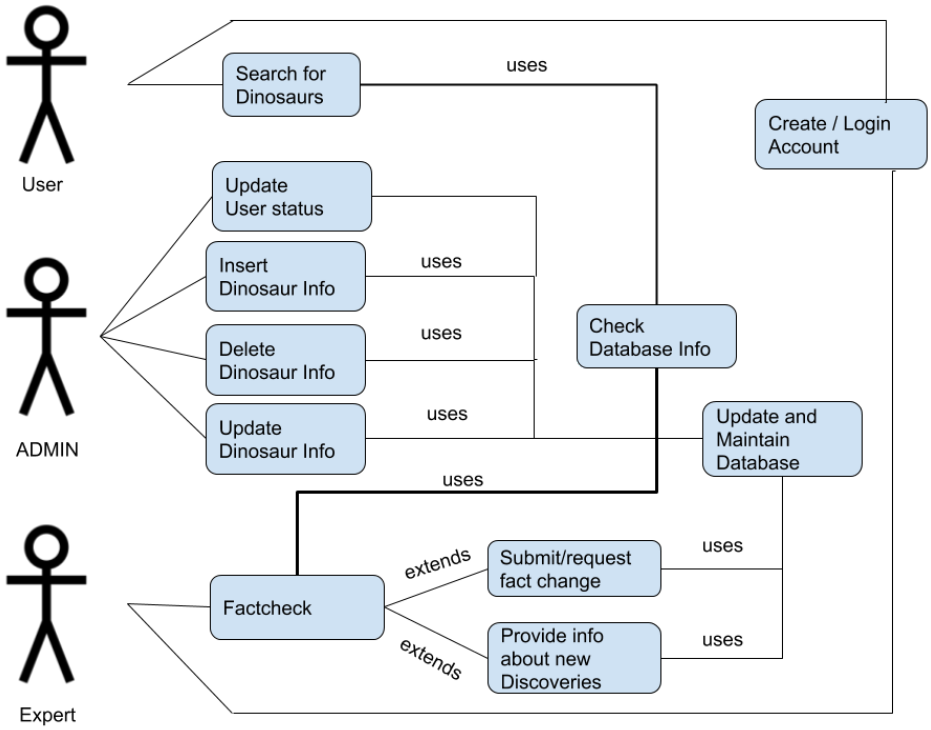

Figure 1: Use Case Diagram

As shown above, the regular user can search for information in the database and can also create an account but will have no privileges. The expert will take the role of a historian which means he is in charge of making sure the correct information is displayed or that new discoveries are known. This requires the expert to have access and the ability to update the request table that is hidden to regular users. The most important user is the administrator which has the power to alter any information on the database. He has the ability to delete, update, or insert data within the program. When the expert makes a request, it will prompt the admin to make those changes on the following table seen below. **Figure 2** shows how all the tables are connected and how they interact with each other.

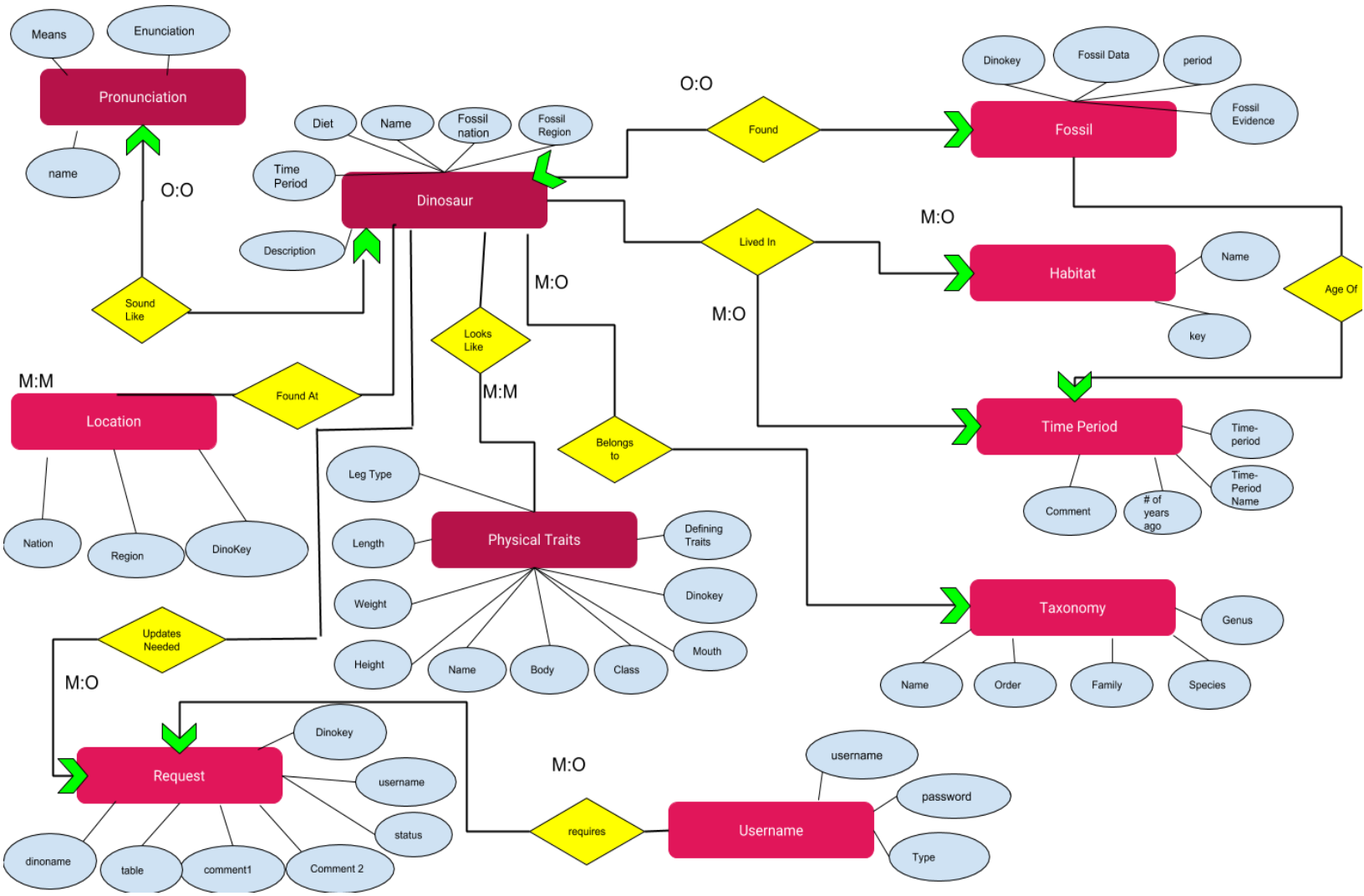

Figure 2: ER Diagram (Red Tables, Blue attributes, Yellow diamond)

Figure 2 shows our E/R diagram for our database. The database consists of 10 tables, each with a varying amount of attributes and different relationships. 8 of our tables revolve around the dinosaurs in the database, with some of the tables being: **Pronunciation, fossil, habitat, etc.** There are two tables that have many to many relationships, these tables being location, and physical traits. The other 6 tables vary between many to one or one to one relationships. These tables helped provide the foundation for our data, and were instrumental in properly organizing all the data that we inputted. The other two tables were mainly to help in implementing the use cases that we outlined back in phase 1. **Userbase** is used to store the usernames and passwords of the user base and to also store their status. There are three statuses: ADMIN, HIST, and USER. ADMIN and HIST get more options, while USER only has available the search queries in the database. **Requests** is used for HIST, as they are fact checkers, and they can submit requests in order to submit any errors/new information they might have. The use of these 10 tables were instrumental in us implementing all the use cases outlined in our phase 1 report.

Tables

- **Dinosaur** (key, name, diet, timeperiod, description, type, habitatkey)
- **Fossil**(key, fossilData, FossilEvidence, period)
- **Habitat** (habitatname, key)
- **Location**(nation, region, dinokey)
- **TimePeriod** (comment, years ago , name)
- **Taxonomy** (name, order, family, genus, species)
- **Phys_Traits**(name, body, weight, length, height, defining_traits, mouth_type, leg_type)
- **Pronunciation** (enunciation, meaning, name)
- **Request**(name, table, comment1, comment2, status, username)
- **Userbase**(username, password, type)

Relations

- **FoundAt** (name, nation, region)
- **Found** (dinokey, fossil evidence , name)
- **LivedIn** (name, #of years ago, timePeriod_name, type, region)
- **BelongsTo** (name, order, family genus, species)
- **LooksLike** (name,length, weight, body, definingTraits)
- **SoundLike** (name, enunciation)
- **AgeOf** (nation, region, time period, # of years ago)
- **Request** (username, password, type, dinokey, status)
- **UpdatesNeeded** (dinokey, username, status, comments)

Project Implementation

For this project, the language used was Java, and the database engine used was SQLite. We were able to use Java and SQLite together by using Java DataBase Connectivity(JDBC). The reason we used this method was because it was something that a foundation was already built through lab 7 in the course. We felt comfortable using these resources so we decided on continuing to use Java, SQLite, and JDBC, and build upon what we made. Due to this, we were able to built a fairly robust Terminal Menu, that allowed the user to log in, see their user options, and have a menu that changes based on the log-in status. While there may be some limitations to having a program of this style(UI isn't as good as something web based, Java GUI implementations such as Swing, etc.), we felt that we did the best we could, and took advantage of our knowledge to build something that looks clean and easy to use for the user. We even implemented a color mode to make it pop out more, but unfortunately did not show it for the presentation in order to prevent the risk of washed out colors that projectors on campus may have. **Figure 3** below shows the UI without color mode enabled, while **Figure 4** shows the UI with color mode enabled.

Figure 3: DinoDatabase UI with color mode disabled and ADMIN functionality.

Figure 4: DinoDatabase UI with color mode enabled and USER functionality.

As you can see the Menus are fairly clean considering our limitations, we also added things such as color, and ASCII art, in order to have a good look. Also looking at the figures, you can see how the menu changes based on the login status, with the ADMIN being able to see all the options, the USER only able to see the search options, and EXPERT only able to see the search options and request options, but not data modification options. All of this is implemented with two java files, **Dino.java** and **DinoQueries.java**, which contain information like the search functions, menus, expert functions, admin functions, and the code needed to implement stuff such as color and ASCII art.

Most of the challenges faced were dealing with how to best implement the SQL queries, and how to best print them out. For this, we decided on printing each column by line, in order to prevent formatting issues that come with printing columns side to side. Another issue that was faced was implementing Data modification statements without the user possibly doing some sort

of SQL injection. To do this, we used a combination of table names stored in an array and table metadata in order to ask the user where they'd like to place their modification. This prevented the user from getting free reign to type in whatever they want and potentially causing something to crash or go wrong. We hope that our method prevents this from happening, and so far it looks like it does. The code size overall is pretty large, as there are a lot of SQL statements to be run and a lot of input asked for to fill in prepared statements and stuff of that nature.This makes sense considering the point of the project is to provide search options for the user to able to find dinosaurs based on the option they pick.

Conclusion

Through each phase of this project, we were able to see and track our progress and be able to make changes based on new findings we made. We were able to work together and plan out a program that features different use cases, a database that features a good amount of dinosaurs and their respective information, and a Java/JDBC implementation that's simple and hopefully nice to use. We were able to supply a robust number of search options for the user to be able to search based on a number of factors, and also included a couple of fun ones, such as a query that returns a random dinosaur and allows you to learn more about it. The goal of this project was to be able to learn how to create and maintain a database, and how to create a connection to said database using programs such as Java and JDBC. The goal of the program itself was for the user to be able to learn more about dinosaurs and to come away learning something new. Over the course of the semester, the project went from an idea to a program that people are able to use, and be able to learn about dinosaurs.