SoftSys LKM 6 Report

Alison Berkowitz and Hayley Hansson

May 2014

1 Topics Explored

For our final Loadable Kernel Module, we decided to learn more about database systems. We did this by learning about the MySQL database management system and it's C API. This not only allowed us to explore the topic of databases, but also practice programming in C.

2 Resources

We followed two tutorials before beginning our final product. In the first tutorial, we became familiar with MySQL as a database management system. We issued MySQL commands through the command line to learn how to add records to a table and retrieve records from a table. This tutorial can be found at: http://www.elated.com/articles/mysql-for-absolute-beginners/
The second tutorial we followed demonstrated how to use the C programming API for MySQL. We learned how to write a C program using MySQL. This tutorial can be found at: http://zetcode.com/db/mysqlc/

3 Learning Goals

In our project proposal, we stated that our final deliverable should be a functional boxer database with criteria boxers would use to find sparring partners. We wanted to work on database design, firstly, because we both wished we had gone into a bit more depth with LMK 3 and wished to further our understanding of DBMS's. We also wanted to become more comfortable with writing programs in C by getting more experience.

4 Acheivement

We succeeded in creating this database using C and MySQL like we intended. We used C (coupled with MySQL) for the entire project, and learned a lot by writing some programs from nothing and debugging. We were able to incorporate our previous knowledge from Head First C with some new things we picked

up that were more specific to our project. We do feel more comfortable writing our own programs in C, and are happy to have some useful insights about database design.

5 Final Product

Our final product consists of five different C programs that interact with our Boxer Database. It can be found here: https://github.com/alisonberkowitz/SoftwareSystems/tree/master/l
To run our programs, change all instances of

```
mysql_real_connect(con, "localhost", "root", "51Perkins",
        "boxerdb", 0, NULL, 0)
```

Replace "51Perkins" with your own password.

5.1 createdb.c

The first C program we wrote creates our Boxer database. It simply initiates a MySQL connection and issues a query to create a new database using

```
mysql_query(con, "CREATE DATABASE boxerdb")
```

This program is only run once at the initial set up of the system.

5.2 populatedb.c

We added populatedb.c in order to create a table inside the database. To add the table, we defined the columns by what criteria we would use to match sparring partners in this function.

```
mysql_query(con, "CREATE TABLE Boxers(Id INT PRIMARY KEY
    AUTO_INCREMENT, FirstName varchar(255) NOT NULL,
    LastName varchar(255) NOT NULL, Sex varchar(2) NOT
    NULL, Age INT NOT NULL, Weight varchar(50) NOT NULL,
    Bouts INT NOT NULL)")
```

In order to figure out syntax, we put a few people and their information into the database after creating the table.

```
if (mysql_query(con, "INSERT INTO Boxers(FirstName,
    LastName, Sex, Age, Weight, Bouts) VALUES('Hayley',
    'Hansson', 'F', 20, 'Super Lightweight', 2)")) {
     finish_with_error(con);
}
```

This program is only run once at the initial set up of the system, after createdb.c.

5.3 retrieve_data.c

We created this function in order to easily check the contents of the database. When run, it prints out everyone in the database and their information. This was most helpful when we were debugging addtodb.c because we could easily check whether or not we had successfully added a person and their correct information.

5.4 addtodb.c

This program allows the user to add themselves and their information to the database from the command line. We use scanf to get user input and make a query to the database to add a new row to the Boxer table.

5.5 querydb.c

This program allows a user to find their matches in our database from the command line. We use scanf to get their own name, and a list of every other boxer of the same gender in the database in the same weight class is printed. This is done through two MySQL queries. The first finds the boxer in the database matching the inputed first and last name and stores it as user.

```
char prefix[1000] = "SELECT * FROM Boxers WHERE
    FirstName = '";
strcat(prefix, FirstName);
strcat(prefix, "' AND LastName = '");
strcat(prefix, LastName);
strcat(prefix, "' LIMIT 1");

if (mysql_query(con, prefix)) {
    finish_with_error(con);
}

MYSQL_RES *user = mysql_store_result(con);
```

The second finds every other boxer in the database who's gender and weight class matches.