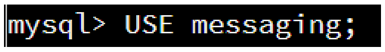
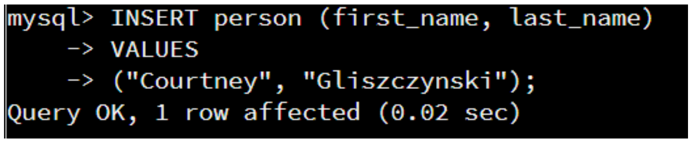
Courtney Gliszczynski

Final Project



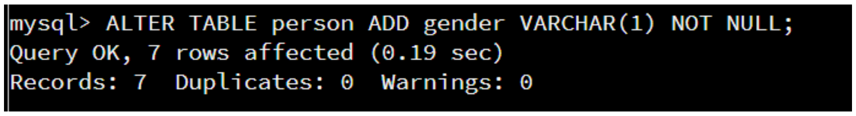
```
mysql> USE messaging;
```

I began my project by first assigning which database I wanted to work in by using the USE function.



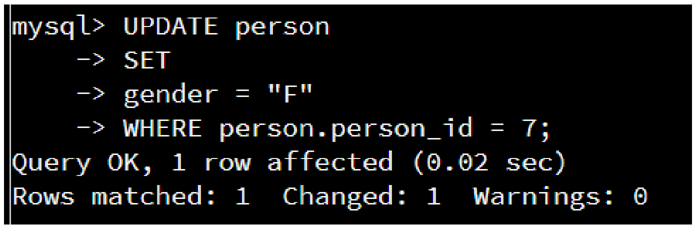
```
mysql> INSERT person (first_name, last_name)
    -> VALUES
    -> ("Courtney", "Gliszczynski");
Query OK, 1 row affected (0.02 sec)
```

The first task I had to do was add my information into the 'person' table. I did this by using the INSERT function, and then listing which columns I wanted to insert the data into. I didn’t include the person_id column because that is an auto generated column. I then used the VALUES function to add the desired information the respective columns.



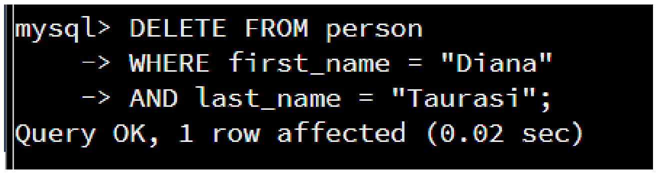
```
mysql> ALTER TABLE person ADD gender VARCHAR(1) NOT NULL;
Query OK, 7 rows affected (0.19 sec)
Records: 7  Duplicates: 0  Warnings: 0
```

The next task was to add a column into the 'person' table. I used the ALTER TABLE function, which I listed which table I was editing, followed by the ADD function, which I inserted what column I wanted to add to the table. I decided on gender, which only required 1 character and was required.



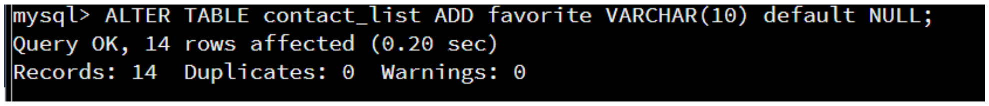
```
mysql> UPDATE person
    -> SET
    -> gender = "F"
    -> WHERE person.person_id = 7;
Query OK, 1 row affected (0.02 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

I then had to update the 'person' table. I used the UPDATE function, listed the table I wanted to update. I then used the SET function which I told the statement what needed to be changed. In order to change the value for my column I used the WHERE function to set the person_id to my id number.



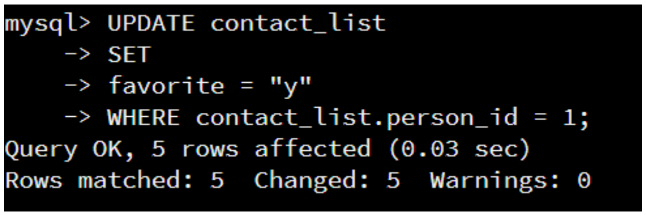
```
mysql> DELETE FROM person
    -> WHERE first_name = "Diana"
    -> AND last_name = "Taurasi";
Query OK, 1 row affected (0.02 sec)
```

I then used the DELETE FROM function, and then listed the 'person' table since that is the table that I have to remove a record from. I also used the WHERE function to help narrow down which record I wanted removed. Since I needed to use the first and last name as parameters, I also used the AND function so I could include both.



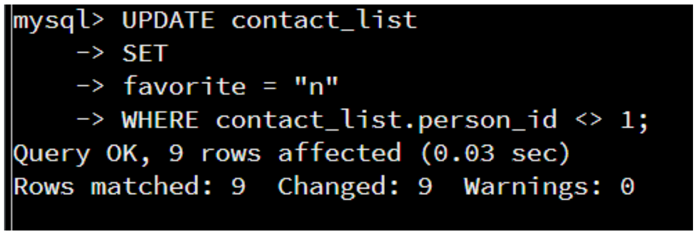
```
mysql> ALTER TABLE contact_list ADD favorite VARCHAR(10) default NULL;
Query OK, 14 rows affected (0.20 sec)
Records: 14  Duplicates: 0  Warnings: 0
```

I used the same functions for altering the 'contact_list' table that I did for when I altered the 'person' table above. However, for the column I was adding, I made it so it could only have 10 characters by using the VARCHAR function. I also made it optional, by setting the default to NULL.



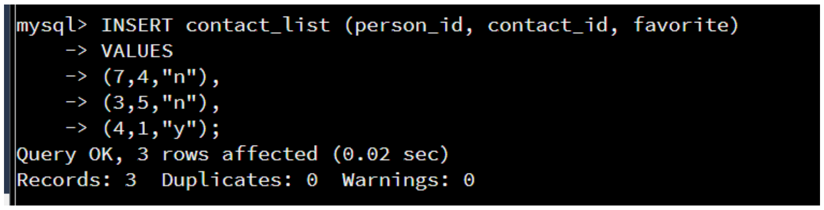
```
mysql> UPDATE contact_list
    -> SET
    -> favorite = "y"
    -> WHERE contact_list.person_id = 1;
Query OK, 5 rows affected (0.03 sec)
Rows matched: 5  Changed: 5  Warnings: 0
```

For my next task I had to update the 'contact_list' table so that favorite was set to 'y' when the person_id was Michael Phelps. I used the UPDATE function, listed the table I wanted to update and then used the SET function. I listed favorite as equaling 'y' because that's what I was changing in the table. I also used the WHERE function to set some parameters. I looked up Michael Phelps person_id number by using SELECT * FROM person; I then set that as the person_id that needed to be listed for favorite to be 'y'.



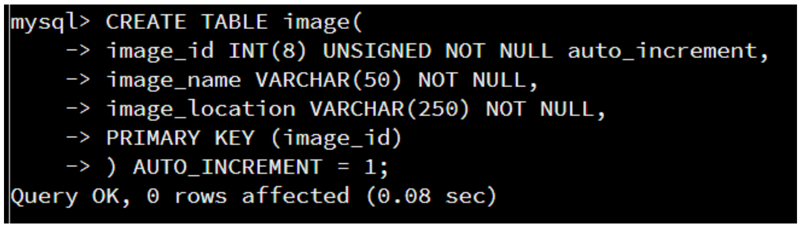
```
mysql> UPDATE contact_list
    -> SET
    -> favorite = "n"
    -> WHERE contact_list.person_id <> 1;
Query OK, 9 rows affected (0.03 sec)
Rows matched: 9  Changed: 9  Warnings: 0
```

The next part of this was updating the 'contact_list' so everyone else had 'n' listed under favorite. I copied the SQL statement used previously, however instead of setting person_id equal to 1, I did the opposite, I set it unequal to 1. That way every person_id that is not 1, will have 'n' for their record in the favorite column.



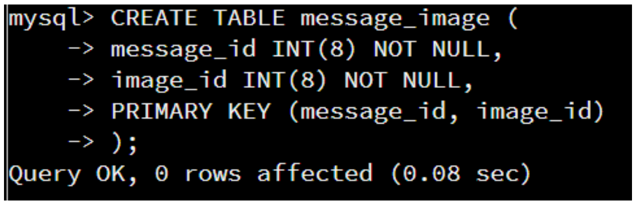
```
mysql> INSERT contact_list (person_id, contact_id, favorite)
    -> VALUES
    -> (7,4,"n"),
    -> (3,5,"n"),
    -> (4,1,"y");
Query OK, 3 rows affected (0.02 sec)
Records: 3  Duplicates: 0  Warnings: 0
```

In order to add 3 new records to the 'contact_list' table I used the INSERT function. I listed the table I wanted to insert the data into, and in the parenthesis, I listed the columns I wanted to include. After that I used the VALUES function to start adding the records. I used parenthesis for inputting the data and listed the data to match how the corresponding columns were listed. After every record I put a comma afterwards in order to continue writing the code. Once I was finished adding records, I used the semi-colon.



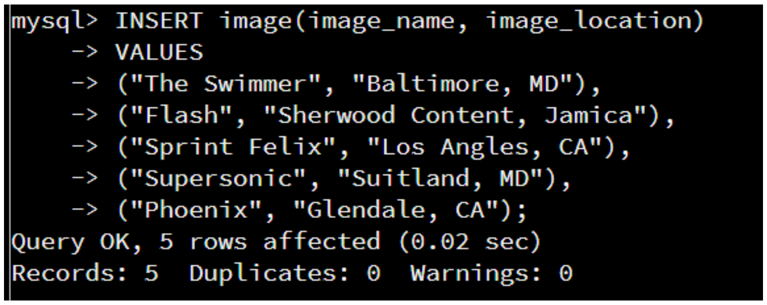
```
mysql> CREATE TABLE image(
    -> image_id INT(8) UNSIGNED NOT NULL auto_increment,
    -> image_name VARCHAR(50) NOT NULL,
    -> image_location VARCHAR(250) NOT NULL,
    -> PRIMARY KEY (image_id)
    -> ) AUTO_INCREMENT = 1;
Query OK, 0 rows affected (0.08 sec)
```

The next task was to create a whole new table. I used the CREATE TABLE function, followed by the name on the new table, 'image'. The first column I made it as an integer with only a size of 8 numbers long, unsigned, required and made it so it would automatically change its number with every record. Next, I create a column where it could only have 50 characters and used NOT NULL to make it required. The third column I set to have a maximum number of 250 characters, and also set to be required. I used the PRIMARY KEY function to set the image_id column to be the primary key. I also set the increment size to be one, so the image_id will only add one after every record inputted.



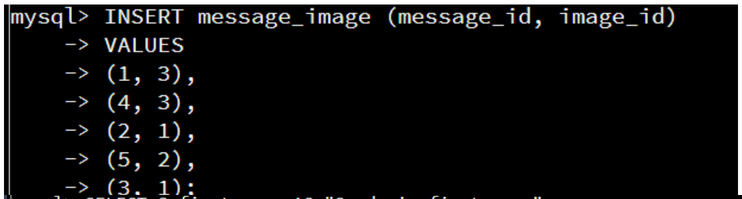
```
mysql> CREATE TABLE message_image (
    -> message_id INT(8) NOT NULL,
    -> image_id INT(8) NOT NULL,
    -> PRIMARY KEY (message_id, image_id)
    -> );
Query OK, 0 rows affected (0.08 sec)
```

I had to create another table in this step, so I used a similar statement to the one above. Each column was a required integer with a limited size of 8 numbers. However, for this task both columns needed to be primary keys. I did this by simply using the PRIMARY KEY function and listing both columns inside the parenthesis.



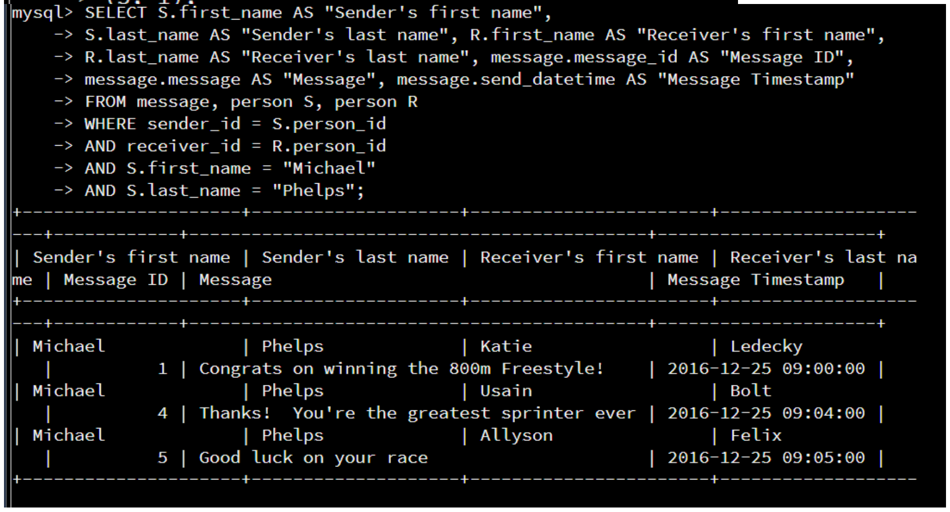
```
mysql> INSERT image(image_name, image_location)
    -> VALUES
    -> ("The Swimmer", "Baltimore, MD"),
    -> ("Flash", "Sherwood Content, Jamica"),
    -> ("Sprint Felix", "Los Angles, CA"),
    -> ("Supersonic", "Suitland, MD"),
    -> ("Phoenix", "Glendale, CA");
Query OK, 5 rows affected (0.02 sec)
Records: 5  Duplicates: 0  Warnings: 0
```

To insert records into the 'image' table I used the INSERT function, followed by the desired table and a set of parenthesis' with the desired columns listed within. I then used the VALUES function and inputted the data below.



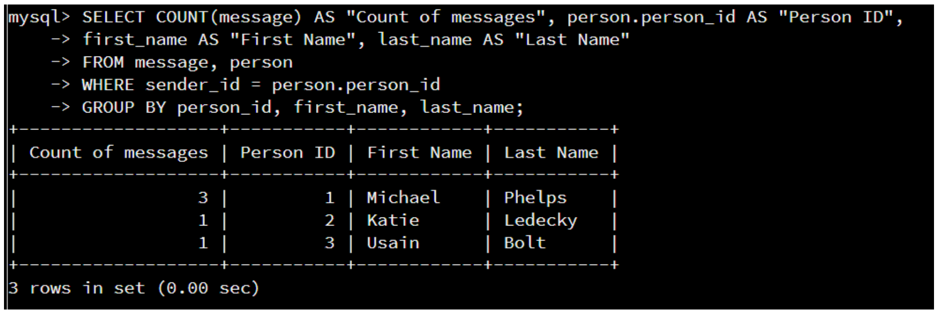
```
mysql> INSERT message_image (message_id, image_id)
    -> VALUES
    -> (1, 3),
    -> (4, 3),
    -> (2, 1),
    -> (5, 2),
    -> (3, 1);
```

I did the same method to insert new records into the 'message_image' table. I just changed the columns listed and the data inputted.



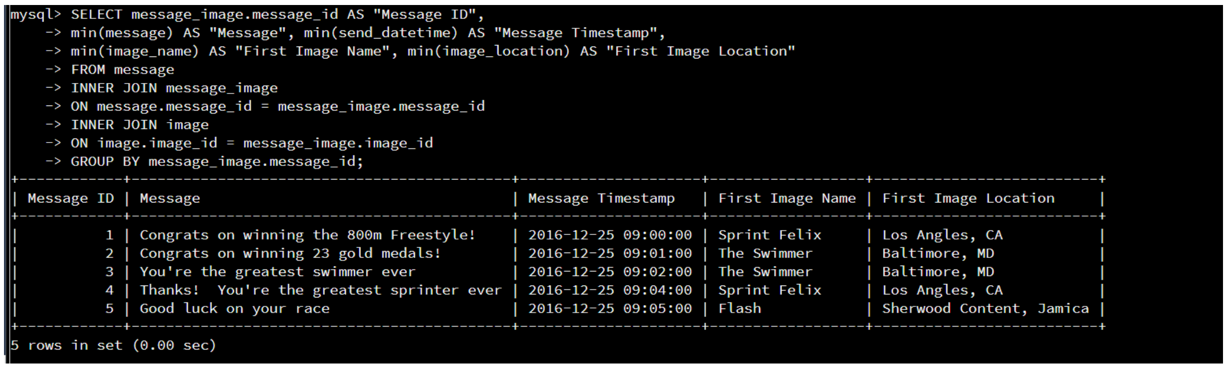
```
mysql> SELECT S.first_name AS "Sender's first name",
    -> S.last_name AS "Sender's last name", R.first_name AS "Receiver's first name",
    -> R.last_name AS "Receiver's last name", message.message_id AS "Message ID",
    -> message.message AS "Message", message.send_datetime AS "Message Timestamp"
    -> FROM message, person S, person R
    -> WHERE sender_id = S.person_id
    -> AND receiver_id = R.person_id
    -> AND S.first_name = "Michael"
    -> AND S.last_name = "Phelps";
+---------------------+--------------------+-----------------------+-------------------
---+------------+--------------------------------------------+---------------------+
| Sender's first name | Sender's last name | Receiver's first name | Receiver's last na
me | Message ID | Message                                    | Message Timestamp   |
+---------------------+--------------------+-----------------------+-------------------
---+------------+--------------------------------------------+---------------------+
| Michael             | Phelps             | Katie                 | Ledecky
   |          1 | Congrats on winning the 800m Freestyle!    | 2016-12-25 09:00:00 |
| Michael             | Phelps             | Usain                 | Bolt
   |          4 | Thanks!  You're the greatest sprinter ever | 2016-12-25 09:04:00 |
| Michael             | Phelps             | Allyson               | Felix
   |          5 | Good luck on your race                     | 2016-12-25 09:05:00 |
+---------------------+--------------------+-----------------------+-------------------
```

For this next task I used the SELECT function and listed all the columns I wanted to include in my output. For first_name and last_name I put a S. if they were for the sender's name and a R. if they were for the receivers' name. While listing the columns I used the AS function to rename what I wanted the column to be. I then used the FROM function to indicate which 'tables' I was getting these columns from. I included a person S and person R to differentiate the two first_name, last_name columns. Next, I set the parameters using the WHERE function. So, I started off setting the correct ids equal to each other, using the AND function because there were multiple parameters. I then set the senders first name to be equal to Michael, and last name equal to Phelps.



```
mysql> SELECT COUNT(message) AS "Count of messages", person.person_id AS "Person ID",
    -> first_name AS "First Name", last_name AS "Last Name"
    -> FROM message, person
    -> WHERE sender_id = person.person_id
    -> GROUP BY person_id, first_name, last_name;
+-------------------+-----------+------------+-----------+
| Count of messages | Person ID | First Name | Last Name |
+-------------------+-----------+------------+-----------+
|                 3 |         1 | Michael    | Phelps    |
|                 1 |         2 | Katie      | Ledecky   |
|                 1 |         3 | Usain      | Bolt      |
+-------------------+-----------+------------+-----------+
3 rows in set (0.00 sec)
```

For this next task, I used the same functions and method as the previous task. Some differences in this statement though is that I used the COUNT function to count the number of messages. I used the SELECT function again to list the desired columns, the AS function to rename them, the FROM function to indicate what tables were used and the WHERE function to help make a connection within the tables. This time I did use the GROUP BY function to help reduce the amount of repeating records and help condense the size of the output.



```
mysql> SELECT message_image.message_id AS "Message ID",
    -> min(message) AS "Message", min(send_datetime) AS "Message Timestamp",
    -> min(image_name) AS "First Image Name", min(image_location) AS "First Image Location"
    -> FROM message
    -> INNER JOIN message_image
    -> ON message.message_id = message_image.message_id
    -> INNER JOIN image
    -> ON image.image_id = message_image.image_id
    -> GROUP BY message_image.message_id;
+------------+--------------------------------------------+---------------------+------------------+-------------------------+
| Message ID | Message                                    | Message Timestamp   | First Image Name | First Image Location    |
+------------+--------------------------------------------+---------------------+------------------+-------------------------+
|          1 | Congrats on winning the 800m Freestyle!    | 2016-12-25 09:00:00 | Sprint Felix     | Los Angles, CA          |
|          2 | Congrats on winning 23 gold medals!        | 2016-12-25 09:01:00 | The Swimmer      | Baltimore, MD           |
|          3 | You're the greatest swimmer ever           | 2016-12-25 09:02:00 | The Swimmer      | Baltimore, MD           |
|          4 | Thanks!  You're the greatest sprinter ever | 2016-12-25 09:04:00 | Sprint Felix     | Los Angles, CA          |
|          5 | Good luck on your race                     | 2016-12-25 09:05:00 | Flash            | Sherwood Content, Jamica |
+------------+--------------------------------------------+---------------------+------------------+-------------------------+
5 rows in set (0.00 sec)
```

For the last task I used the SELECT function again, but when I was listing my functions I used the min() function in order to get the “first” image, location and such. I then used the FROM function and INNER JOIN function to show that we are joining the 'message_image' table to the 'message' table. I used the ON function to indicate how the tables were connected. I did this again to connect the 'message' and 'image' table. Finally, I used the GROUP BY function to group the data by the message_id.