

# Southern New Hampshire University

## DAD 220 Module 6-1

Prepared on: 10 Apr., 2022

Prepared for: Prof. Aastha Agarwal

Prepared by: Alexander Ahmann

## Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Procedure</b>	<b>2</b>
2.1	Creating the database and its tables . . . . .	2
2.2	Updating table labels . . . . .	7
2.3	Importing data to QuantigrationUpdates . . . . .	8
2.4	Basic data analysis . . . . .	10
2.5	Inserting new entries . . . . .	12
2.6	UPDATE and DELETE: Modifying records . . . . .	14
2.7	Exporting to .csv . . . . .	17
<b>3</b>	<b>Summary</b>	<b>18</b>

## 1 Introduction

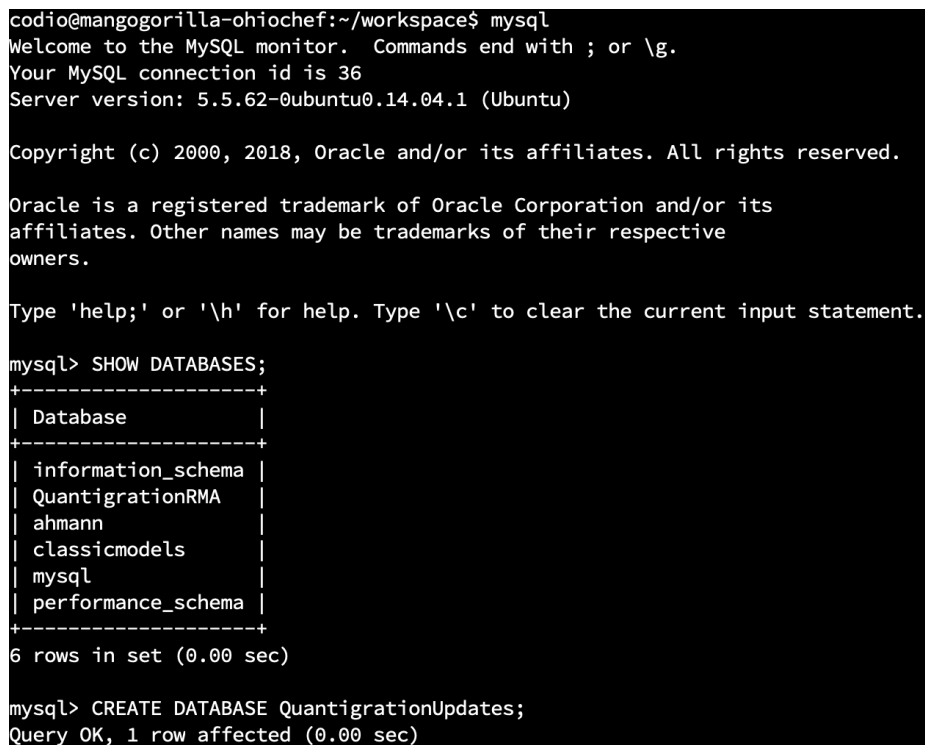
Our company, *Quantigration*, is experiencing rapid growth. While this is a good thing in terms of profit maximisation, new challenges regarding the storage and retrieval of information regarding our business have arisen. To attack this problem, we should make use of relational databases that are driven by the *Structured Query Language*—or simply SQL. In this report, I will discuss my efforts in engineering a relational database for handling our company information.

## 2 Procedure

### 2.1 Creating the database and its tables

I shall begin by logging on to our MySQL instance and creating the database QuantigrationUpdates<sup>1</sup> with the following SQL query:

```
CREATE DATABASE QuantigrationUpdates;
```

A terminal window with a black background and white text. The user 'codio' is logged into a machine named 'mangogorilla-ohiochef'. They run the command 'mysql'. The MySQL monitor displays a welcome message, connection ID 36, and server version 5.5.62-0ubuntu0.14.04.1. After typing 'SHOW DATABASES;', it lists six databases: information\_schema, QuantigrationRMA, ahmann, classicmodels, mysql, and performance\_schema. Then, after typing 'CREATE DATABASE QuantigrationUpdates;', it confirms 'Query OK, 1 row affected (0.00 sec)'.

```
codio@mangogorilla-ohiochef:~/workspace$ mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 36
Server version: 5.5.62-0ubuntu0.14.04.1 (Ubuntu)

Copyright (c) 2000, 2018, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> SHOW DATABASES;
+-----+
| Database                |
+-----+
| information_schema      |
| QuantigrationRMA        |
| ahmann                  |
| classicmodels           |
| mysql                   |
| performance_schema      |
+-----+
6 rows in set (0.00 sec)

mysql> CREATE DATABASE QuantigrationUpdates;
Query OK, 1 row affected (0.00 sec)
```

Figure 1: Logging into MySQL instance and creating the database.

Fig. 1 depicts a screenshot of the terminal output demonstrating that I managed to complete this task. Next, I shall proceed to create the tables based on an *Entity-Relations Diagram*—or E.R. Diagram— schematic that was provided to myself by an colleague. Fig. 2 depicts this E.R. Diagram— specifically depicting three tables: a Customers table, an Orders table and finally an RMA table. The connectors representing relationships show a one-to-many relationship between

---

<sup>1</sup>This answers Step 1, Questions 1 and 2

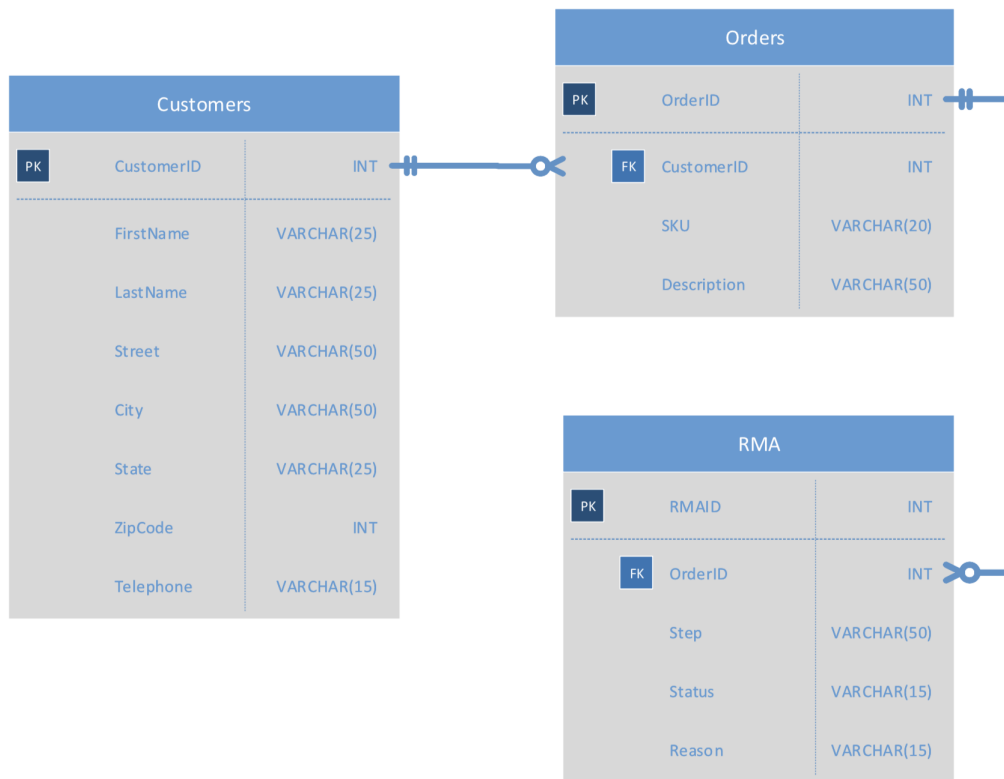


Figure 2: **The QuantigrationUpdates' E.R. diagram (Week 6 resources, n.d.).**

Customers and Orders, and another one-to-many relationship between Orders and RMA.

Regarding the Customers table, its attributes are defined by the E.R. Diagram as:

CustomerID [PK]	INT
FirstName	VARCHAR(25)
LastName	VARCHAR(25)
Street	VARCHAR(50)
City	VARCHAR(50)
State	VARCHAR(25)
ZipCode	INT
Telephone	VARCHAR(15)

Regarding the Orders table, its attributes are defined by the E.R. Diagram as:

OrderID [PK]	INT
CustomerID [FK]	INT
SKU	VARCHAR(20)
Description	VARCHAR(50)

Regarding the RMA table, its attributes are defined by the E.R. Diagram as:

RMAID [PK]	INT
OrderID [FK]	INT
Step	VARCHAR(50)
Status	VARCHAR(15)
Reason	VARCHAR(15)

Based on the E.R. diagram and the nature of the feature space, I have worked out that the Customers' table one-to-many relationship with Orders involves the foreign key CustomerID in Customers' which references Customers' primary key. A similar relationship between Orders and RMA is similar: in which the Orders' primary key is used referenced as a foreign key for RMA.

To create the Customers table, the following query will be used:<sup>2</sup>

```
CREATE TABLE Customers (
    CustomerID INT PRIMARY KEY,
    FirstName VARCHAR(25),
    LastName VARCHAR(25),
    Street VARCHAR(50),
    City VARCHAR(50),
    State VARCHAR(25),
    ZipCode INT,
    Telephone VARCHAR(15)
);
```

I shall use the SQL query DESCRIBE Customers; to confirm that the table was indeed created. Fig. 3 depicts a screenshot of the terminal output that demonstrates that I was able to complete these tasks.

Turning my attention to the Orders table, it references the Customers table's CustomerID attribute through a foreign key. So, the SQL statement that I will use to create this table is:<sup>3</sup>

<sup>2</sup>This is part of my answer to Step 1, Question 3.

<sup>3</sup>This is part of my answer to Step 1, Question 3.

```
mysql> CREATE TABLE Customers (
  -> CustomerID INT PRIMARY KEY,
  -> FirstName VARCHAR(25),
  -> LastName VARCHAR(25),
  -> Street VARCHAR(50),
  -> City VARCHAR(50),
  -> State VARCHAR(25),
  -> ZipCode INT,
  -> Telephone VARCHAR(15));
Query OK, 0 rows affected (0.05 sec)

mysql> DESCRIBE Customers;
```

Field	Type	Null	Key	Default	Extra
CustomerID	int(11)	NO	PRI	NULL	
FirstName	varchar(25)	YES		NULL	
LastName	varchar(25)	YES		NULL	
Street	varchar(50)	YES		NULL	
City	varchar(50)	YES		NULL	
State	varchar(25)	YES		NULL	
ZipCode	int(11)	YES		NULL	
Telephone	varchar(15)	YES		NULL	

```
8 rows in set (0.00 sec)
```

Figure 3: **Creating the Customers table.**

```
CREATE TABLE Orders (
  OrderID INT PRIMARY KEY,
  CustomerID INT,
  SKU VARCHAR(20),
  Description VARCHAR(50),
  FOREIGN KEY (CustomerID)
    REFERENCES Customers(CustomerID)
    ON UPDATE CASCADE
    ON DELETE RESTRICT
);
```

As with the Customers table, I will use `DESCRIBE Orders;` to confirm that the table was indeed created. Fig. 4 and fig. 5 depicts a screenshot of the terminal

output that demonstrates that I was able to complete these tasks.<sup>4</sup>

```
mysql> CREATE TABLE Customers (  
-> CustomerID INT PRIMARY KEY,  
-> FirstName VARCHAR(25),  
-> LastName VARCHAR(25),  
-> Street VARCHAR(50),  
-> City VARCHAR(50),  
-> State VARCHAR(25),  
-> ZipCode INT,  
-> Telephone VARCHAR(15)  
-> );  
Query OK, 0 rows affected (0.05 sec)
```

Figure 4: **Creating the Customers table.**

```
mysql> DESCRIBE Collaborators;  
+-----+-----+-----+-----+-----+-----+  
| Field      | Type      | Null | Key | Default | Extra |  
+-----+-----+-----+-----+-----+-----+  
| CustomerID | int(11)    | NO   | PRI | NULL    |       |  
| FirstName  | varchar(25)| YES  |     | NULL    |       |  
| LastName   | varchar(25)| YES  |     | NULL    |       |  
| Street     | varchar(50)| YES  |     | NULL    |       |  
| City       | varchar(50)| YES  |     | NULL    |       |  
| State      | varchar(25)| YES  |     | NULL    |       |  
| ZipCode    | int(11)    | YES  |     | NULL    |       |  
| Telephone  | varchar(15)| YES  |     | NULL    |       |  
+-----+-----+-----+-----+-----+-----+  
8 rows in set (0.01 sec)  
  
mysql> DESCRIBE Orders;  
+-----+-----+-----+-----+-----+-----+  
| Field      | Type      | Null | Key | Default | Extra |  
+-----+-----+-----+-----+-----+-----+  
| OrderID    | int(11)    | NO   | PRI | NULL    |       |  
| CustomerID | int(11)    | YES  | MUL | NULL    |       |  
| SKU        | varchar(20)| YES  |     | NULL    |       |  
| Description | varchar(50)| YES  |     | NULL    |       |  
+-----+-----+-----+-----+-----+-----+  
4 rows in set (0.00 sec)
```

Figure 5: **Creating the Customers and the Orders table.**

---

<sup>4</sup>Due to technical errors, I could not show the process of creating the Orders table. Furthermore, I am describing the Collaborators table, which is what I changed the Customers table to in a future step.

Finally I will proceed with building the RMA table. As with the Orders table, I need to reference the OrderID primary key in the Orders table with a foreign key. The SQL statement that I will use to accomplish this is similar to the one that I used to create the Orders table:<sup>5</sup>

```
CREATE TABLE RMA (  
    RMAID INT PRIMARY KEY,  
    OrderID INT,  
    Step VARCHAR(50),  
    Status VARCHAR(15),  
    Reason VARCHAR(15),  
    FOREIGN KEY (OrderID)  
        REFERENCES Orders(OrderID)  
        ON UPDATE CASCADE  
        ON DELETE RESTRICT  
);
```

As with the two previous tables, I will use the SQL statement `DESCRIBE RMA;` to confirm that this table was indeed created. Fig. 6 depicts a screenshot that demonstrates that I was able to complete these tasks.

## 2.2 Updating table labels

To demonstrate an advantage of relational databases, I will change references in table names that reference the word “Customer” to “Collaborator.” I will use the following SQL statement:<sup>6 7</sup>

```
RENAME TABLE Customers  
    TO Collaborators;
```

I will use the `SHOW TABLES;` to confirm that I indeed was able to change the Customers table’s name to Collaborators. Fig. 7 depicts a screenshot of terminal output that demonstrates that I was able to complete this task.

---

<sup>5</sup>This is part of my answer to Step 1, Question 3.

<sup>6</sup>This is part of my answer to Step 1, Question 4.

<sup>7</sup>Note that I could not work out a means to rename the CustomerID column for the (former) Customers table and the Orders table. Doing so outputs some kind of error regarding the foreign key, and I am not sure how to solve this.

```
mysql> CREATE TABLE RMA (
->     RMAID INT PRIMARY KEY,
->     OrderID INT,
->     Step VARCHAR(50),
->     Status VARCHAR(15),
->     Reason VARCHAR(15),
->     FOREIGN KEY (OrderID)
->         REFERENCES Orders(OrderID)
->         ON UPDATE CASCADE
->         ON DELETE RESTRICT
-> );
Query OK, 0 rows affected (0.07 sec)

mysql> DESCRIBE RMA;
+-----+-----+-----+-----+-----+-----+
| Field | Type      | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| RMAID | int(11)   | NO   | PRI | NULL    |       |
| OrderID | int(11)   | YES  | MUL | NULL    |       |
| Step   | varchar(50) | YES  |     | NULL    |       |
| Status | varchar(15) | YES  |     | NULL    |       |
| Reason | varchar(15) | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.00 sec)
```

Figure 6: **Creating the RMA table.**

```
mysql> RENAME TABLE Customers
->     TO Collaborators;
Query OK, 0 rows affected (0.01 sec)

mysql> SHOW TABLES;
+-----+
| Tables_in_QuantigrationUpdates |
+-----+
| Collaborators                    |
| Orders                          |
| RMA                             |
+-----+
3 rows in set (0.00 sec)
```

Figure 7: **Changing Customers to Collaborators**

## 2.3 Importing data to QuantigrationUpdates

Next, the QuantigrationUpdates needs to have its data populated with customer and transaction data. The raw data files are stored in a comma-separated



values format (.csv) file format—a text file representing a tabular dataset where the rows are usually separated with a line break and columns are typically separated by a column—hence its name. In particular, the files `rma.csv`, `customers.csv` and `orders.csv` are to be imported into the Collaborators, Orders and RMA tables respectively.

I will use the common Unix command `head` to examine the first ten rows of each table. Here are excerpts from each terminal output:

```
codio@mangogorilla-ohiochef:~/workspace$ head rma.csv
0,53832,Product replacement or account refund
    processed,Complete,Rejected
1,30050,Product replacement or account refund
    processed,Complete,Rejected
[... snip ...]
codio@mangogorilla-ohiochef:~/workspace$ head customers.csv
20225,Kelley,Rivas,888 White Fabien Blvd.,Austin,
    Connecticut,90064,709-851-1060
56261,Damian,Moyer,151 North Green Old Parkway,
    Anaheim,Minnesota,54268,3550179730
[... snip ...]
codio@mangogorilla-ohiochef:~/workspace$ head orders.csv
0,76368,BAS-08-1 C,Basic Switch  10/100/1000 BaseT 8 port
2,62494,BAS-48-1 C,Basic Switch 10/100/1000 BaseT 48 port
[... snip ...]
```

What I can tell from these outputs is that the first row does not label the data. This will be useful information when writing my SQL statements to import the data.

I will also use the Unix command `pwd` to get the working directory of where the CSV files are stored. The following output is what I got:

```
codio@mangogorilla-ohiochef:~/workspace$ pwd
/home/codio/workspace
```

The full path of the CSV file in question is the file name of the CSV file appended to the working directory. For example: the `rma.csv` CSV file would have a full path of:

```
/home/codio/workspace/rma.csv
```

This will be helpful when writing my SQL statements to import CSV files into their respective tables.

Now I can proceed with the query building. I will log back into the MySQL instance and then devise queries to import tables. Regarding the Collaborators table, I will use the following SQL query:<sup>8</sup>

```
LOAD DATA INFILE "/home/codio/workspace/customers.csv"
  INTO TABLE Collaborators
  FIELDS TERMINATED BY ","
  LINES TERMINATED BY "\n";
```

I will use a similar query for Orders...

```
LOAD DATA INFILE "/home/codio/workspace/orders.csv"
  INTO TABLE Orders
  FIELDS TERMINATED BY ","
  LINES TERMINATED BY "\n";
```

... and another query for RMA

```
LOAD DATA INFILE "/home/codio/workspace/rma.csv"
  INTO TABLE RMA
  FIELDS TERMINATED BY ","
  LINES TERMINATED BY "\n";
```

Fig. 8 depicts a screenshot of the terminal's output that demonstrates that I have indeed completed tasks.

## 2.4 Basic data analysis

Next, I shall demonstrate retrieving data and doing rudimentary data analysis on the relational data. For example, I have the following SQL statement that joins the Collaborators and Orders tables to get some of their personal information as well as their orders information from Framingham, Massachusetts:<sup>9</sup>

```
SELECT Collaborators.CustomerID, Orders.OrderID, Orders.SKU
  FROM Collaborators
 INNER JOIN Orders
    ON Collaborators.CustomerID = Orders.CustomerID
```

---

<sup>8</sup>This, and the next two queries, are my solution to Step 2, Question 1.

<sup>9</sup>This is part of my solution to Step 2, Question 2a.

```
mysql> LOAD DATA INFILE "/home/codio/workspace/customers.csv"
-> INTO TABLE Collaborators
-> FIELDS TERMINATED BY ","
-> LINES TERMINATED BY "\n";
Query OK, 37994 rows affected (0.26 sec)
Records: 37994 Deleted: 0 Skipped: 0 Warnings: 0

mysql> LOAD DATA INFILE "/home/codio/workspace/orders.csv"
-> INTO TABLE Orders
-> FIELDS TERMINATED BY ","
-> LINES TERMINATED BY "\n";
Query OK, 37994 rows affected, 65535 warnings (0.37 sec)
Records: 37994 Deleted: 0 Skipped: 0 Warnings: 75988

mysql> LOAD DATA INFILE "/home/codio/workspace/rma.csv"
-> INTO TABLE RMA
-> FIELDS TERMINATED BY ","
-> LINES TERMINATED BY "\n";
Query OK, 38162 rows affected (0.37 sec)
Records: 38162 Deleted: 0 Skipped: 0 Warnings: 0
```

Figure 8: Loading the CSV files into their respective tables.

```
WHERE (Collaborators.State = "Massachusetts"
      OR Collaborators.State = "MA")
AND State = "Framingham";
```

The query returned 0 rows.<sup>10</sup>

This SQL statement is similar to the previous one, but includes all of Massachusetts:<sup>11</sup>

```
SELECT Collaborators.CustomerID, Orders.OrderID, Orders.SKU
FROM Collaborators
INNER JOIN Orders
ON Collaborators.CustomerID = Orders.CustomerID
WHERE Collaborators.State = "Massachusetts"
OR Collaborators.State = "MA";
```

The query returns 982 rows.<sup>12</sup>

<sup>10</sup>This is part of my solution to Step 2, Question 2a.

<sup>11</sup>This is part of my solution to Step 2, Question 2b

<sup>12</sup>This is part of my solution to Step 2, Question 2b.

One may “take a peep” in other territories in the United States. For example, I have constructed the following query to get the records for the city “Woonsocket” in Rhode Island:<sup>13</sup>

```
SELECT Collaborators.CustomerID, Orders.OrderID, Orders.SKU
FROM Collaborators
INNER JOIN Orders
ON Collaborators.CustomerID = Orders.CustomerID
WHERE (Collaborators.State = "Rhode Island"
      OR Collaborators.State = "RI")
AND State = "Woonsocket";
```

This query returned 0 rows.<sup>14</sup>

Finally, one can “take a peep” into specific rows that meet some kind of criteria with the WHERE clause. As an example, I shall return an entry from the RMA table with the OrderID=5175. The following SQL statement demonstrates this:<sup>15</sup>

```
SELECT *
FROM RMA
WHERE OrderID = 5175;
```

As shown in Fig. 9, the current status and step are “Pending” and “Awaiting customer Documentation” respectively.

## 2.5 Inserting new entries

The relational database can also have new entries added into their tables with the INSERT statement. I shall demonstrate this by inserting the following information shown in Fig. 10 into their respective tables.

The first query submits new information to the Collaborators table:<sup>16</sup>

```
INSERT INTO Collaborators(CustomerID, FirstName, LastName,
                          Street, City, State, ZipCode, Telephone)
VALUES
    (100004, "Luke", "Skywalker", "15 Maiden Lane",
     "New York", "NY", 10222, "212-555-1234"),
```

<sup>13</sup>This is my solution to Step 2, Question 2d.

<sup>14</sup>Note that I executed this query **AFTER** inserting records in the next section.

<sup>15</sup>This is part of my solution to Step 2, Question 2e

<sup>16</sup>This is part of my solution to Step 2, Question 2c

```
mysql> SELECT RMA.*
      -> FROM Orders
      -> INNER JOIN RMA
      -> ON Orders.OrderID = RMA.OrderID
      -> WHERE RMA.OrderID = 5175;
```

RMAID	OrderID	Step	Status	Reason
31405	5175	Awaiting customer Documentation	Pending	Defective

1 row in set (0.00 sec)

Figure 9: Selecting a row where OrderID = 5175.

CustomerID	FirstName	LastName	StreetAddress	City	State	ZipCode	Telephone
100004	Luke	Skywalker	15 Maiden Lane	New York	NY	10222	212-555-1234
100005	Winston	Smith	123 Sycamore Street	Greensboro	NC	27401	919-555-6623
100006	MaryAnne	Jenkins	1 Coconut Way	Jupiter	FL	33458	321-555-8907
100007	Janet	Williams	55 Redondo Beach Blvd	Torrence	CA	90501	310-555-5678

OrderID	CustomerID	SKU	Description
1204305	100004	ADV-24-10C	Advanced Switch 10GigE Copper 24 port
1204306	100005	ADV-48-10F	Advanced Switch 10 GigE Copper/Fiber 44 port copper 4 port fiber
1204307	100006	ENT-24-10F	Enterprise Switch 10GigE SFP+ 24 Port
1204308	100007	ENT-48-10F	Enterprise Switch 10GigE SFP+ 48 port

Figure 10: Tables to import into their respective tables. (Week 6 resources, n.d.-b)

(100005, "Winston", "Smith", "123 Sycamore Street", "Greenboro", "NC", 27401, "919-555-6623"), (100006, "MaryAnne", "Jenkins", "1 Coconut Way", "Jupiter", "FL", 33458, "321-555-8907"),
---

```
(100007, "Janet", "Williams", "55 Redondo Beach Blvd.",  
"Torrence", "CA", 90510, "310-555-5678");
```

And the second query submits new information to the Orders table:<sup>17</sup>

```
INSERT INTO Orders(OrderID, CustomerID, SKU,  
Description)  
VALUES  
(1204305, 100004, "ADV-24-10C",  
"Advanced Switch 10GigE Copper 24 port"),  
(1204306, 100005, "ADV-48-10F",  
"Advanced Switch 10 GigE Copper/Fiber 44  
port copper 4 port fiber"),  
(1204307, 100006, "ENT-24-10F",  
"Enterprise Switch 10GigE SFP+ 24 Port"),  
(1204308, 100007, "ENT-48-10F",  
"Enterprise Switch 10GigE SFP+ 48 port");
```

Fig. 11 and fig. 12 depicts two screenshots that demonstrate that I was able to complete this task.

## 2.6 UPDATE and DELETE: Modifying records

Furthermore, it can be demonstrated that relational databases can make modifications to their datasets with the UPDATE and DELETE statements. Our company needs to make changes to the Orders database and then do away with rejected cases for the RMA.

First, I shall update the Orders database columns Status and Step with “Complete” and “Credit Customer Account” respectively. The exact query that I used to complete this task is:<sup>18</sup>

```
UPDATE RMA  
SET Status = "Complete",  
Step = "Credit Customer Account"  
WHERE OrderID = 5175;
```

---

<sup>17</sup>This is part of my solution to Step 2, Question 2c.

<sup>18</sup>This is my solution to Step 2, Question 2f.

```
mysql> INSERT INTO Collaborators(CustomerID, FirstName, LastName,
-> Street, City, State, ZipCode, Telephone)
-> VALUES
-> (100004, "Luke", "Skywalker", "15 Maiden Lane",
-> "New York", "NY", 10222, "212-555-1234"),
-> (100005, "Winston", "Smith", "123 Sycamore Street",
-> "Greenboro", "NC", 27401, "919-555-6623"),
-> (100006, "MaryAnne", "Jenkins", "1 Coconut Way",
-> "Jupiter", "FL", 33458, "321-555-8907"),
-> (100007, "Janet", "Williams", "55 Redondo Beach Blvd.",
-> "Torrence", "CA", 90510, "310-555-5678");
Query OK, 4 rows affected (0.02 sec)
Records: 4 Duplicates: 0 Warnings: 0
```

Figure 11: **Insert new data to Collaborators.**

```
mysql> INSERT INTO Orders(OrderID, CustomerID, SKU,
-> Description)
-> VALUES
-> (1204305, 100004, "ADV-24-10C",
-> "Advanced Switch 10GigE Copper 24 port"),
-> (1204306, 100005, "ADV-48-10F",
-> "Advanced Switch 10 GigE Copper/Fiber 44
-> port copper 4 port fiber"),
-> (1204307, 100006, "ENT-24-10F",
-> "Enterprise Switch 10GigE SFP+ 24 Port"),
-> (1204308, 100007, "ENT-48-10F",
-> "Enterprise Switch 10GigE SFP+ 48 port");
Query OK, 4 rows affected, 1 warning (0.08 sec)
Records: 4 Duplicates: 0 Warnings: 1
```

Figure 12: **Insert new data to Orders.**

Fig. 13 depicts a screenshot that demonstrates that I have indeed managed to complete this task:

Finally, the company needs to do away with rejected entries in the RMA database. I have constructed the following SQL statement to complete this task:

```
mysql> UPDATE RMA
  -> SET Status = "Complete",
  ->     Step = "Credit Customer Account"
  -> WHERE OrderID = 5175;
Query OK, 1 row affected (0.02 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> SELECT *
  -> FROM Orders
  -> WHERE OrderID = 5175;
+-----+-----+-----+-----+
| OrderID | CustomerID | SKU      | Description |
+-----+-----+-----+-----+
| 5175    | 20225      | ENT-24-10F | Enterprise Switch 10GigE SFP+ 24 Port |
+-----+-----+-----+-----+
1 row in set (0.00 sec)

mysql> SELECT * FROM RMA WHERE OrderID = 5175;
+-----+-----+-----+-----+-----+
| RMAID | OrderID | Step | Status | Reason |
+-----+-----+-----+-----+-----+
| 31405 | 5175    | Credit Customer Account | Complete | Defective |
+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)
```

Figure 13: Updating the RMA table.

```
DELETE FROM RMA
WHERE Reason = "Rejected";
```

Fig 14 depicts a screenshot of the terminal output of the SQL statement's execution. It reports that 0 tables were deleted.<sup>19</sup>

```
mysql> DELETE FROM RMA
  -> WHERE Reason = "Rejected";
Query OK, 0 rows affected (0.02 sec)
```

Figure 14: Deleting information from the RMA table.

<sup>19</sup>This is my solution to Step 2, Question 2g.



## 2.7 Exporting to .csv

Finally, this information can be exported into a .csv file for whatever purposes that are desired by the company. To demonstrate this, I have devised the following SQL statement to export the Orders table into a .csv file:

```
SELECT *
FROM Orders
INTO OUTFILE "/home/codio/workspace/Project6.csv"
FIELDS TERMINATED BY ","
LINES TERMINATED BY "\n";
```

To verify that the file got exported into .csv, I have use the Unix head command to output the first 10 lines of the result. Fig 15 depicts a screenshot of the terminal output that demonstrates that I have indeed completed this task.<sup>20</sup>

```
mysql> SELECT *
-> FROM Orders
-> INTO OUTFILE "/home/codio/workspace/Project6.csv"
-> FIELDS TERMINATED BY ","
-> LINES TERMINATED BY "\n";
Query OK, 37998 rows affected (0.03 sec)

mysql> exit
Bye
codio@mangogorilla-ohiochef:~/workspace$ head Project6.csv
0,76368,BAS-08-1 C,Basic Switch 10/100/1000 BaseT 8 port
2,62494,BAS-48-1 C,Basic Switch 10/100/1000 BaseT 48 port
6,98077,ENT-48-10F,Enterprise Switch 10GigE SFP+ 48 port
8,85882,ENT-48-40F,Enterprise Switch 40GigE SFP+ 48 port
10,59384,BAS-48-1 C,Basic Switch 10/100/1000 BaseT 48 port
14,96361,ENT-48-10F,Enterprise Switch 10GigE SFP+ 48 port
15,67424,ADV-48-10F,Advanced Switch 10 GigE Copper/Fiber 44 port coppe
16,93634,ENT-24-10F,Enterprise Switch 10GigE SFP+ 24 Port
19,62756,ENT-24-40F,Enterprise Switch 40GigE SFP+ 24 port
20,99453,BAS-48-1 C,Basic Switch 10/100/1000 BaseT 48 port
```

Figure 15: **Exporting Orders into .csv.**

---

<sup>20</sup>This is my solution to Step 2, Question 3.

### **3 Summary**

Hopefully I have convinced to whom may be of concern regarding the Quantigraton's operations that a relational database management system, MySQL in particular, would be of benefit to this organisation. Modern computer and information systems work a lot faster than the traditional "hand method" employed by clerical work defined in the twentieth century. Furthermore, a relational database format is preferable because it offers security and performance measures that will increase productivity in the already productive modern computer and information system.

### **References**

Week 6 resources (n.d.). *Quantigraton E.R. diagram*. DAD-220.

Week 6 resources (n.d.). *Project One Guidelines and Rubric*. DAD-220.