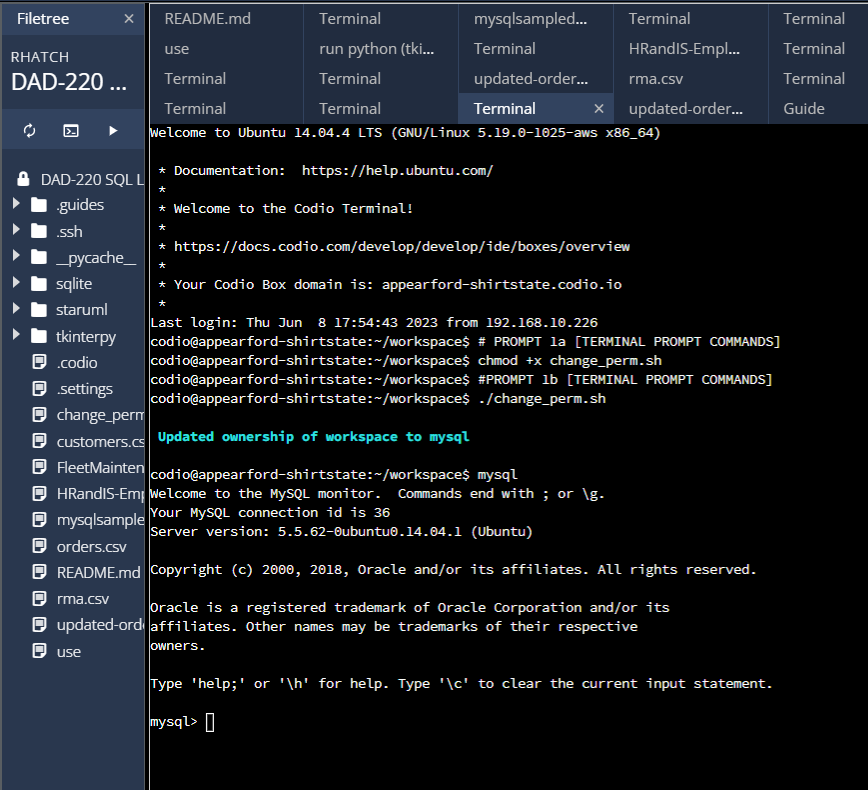
# Ryan Hatch DAD 220 Database Documentation Template

## Step One: Create a Database

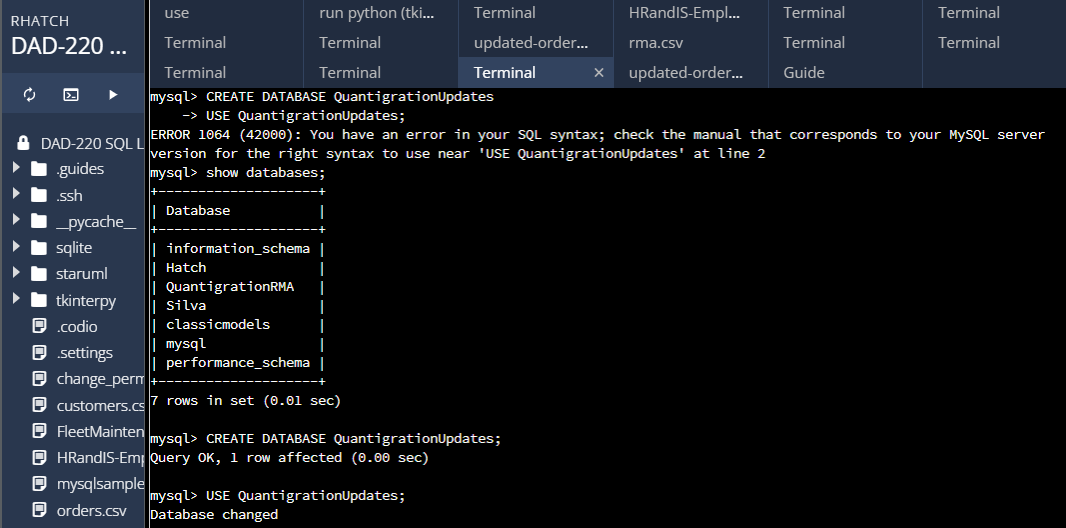
1. Navigate to your online integrated development environment (IDE). List and record the SQL commands that you used to complete this step here:

codio@appearford-shirtstate:~/workspace$ # PROMPT 1a [TERMINAL PROMPT COMMANDS]  
codio@appearford-shirtstate:~/workspace$ chmod +x change\_perm.sh  
codio@appearford-shirtstate:~/workspace$ #PROMPT 1b [TERMINAL PROMPT COMMANDS]  
codio@appearford-shirtstate:~/workspace$ ./change\_perm.sh

  
The permission of a shell script file named "change\_perm.sh" to make it executable, allowing it to be executed as a program. Next, I executed the shell script "change\_perm.sh," which results in updating the ownership of the workspace to the mysql workspace.

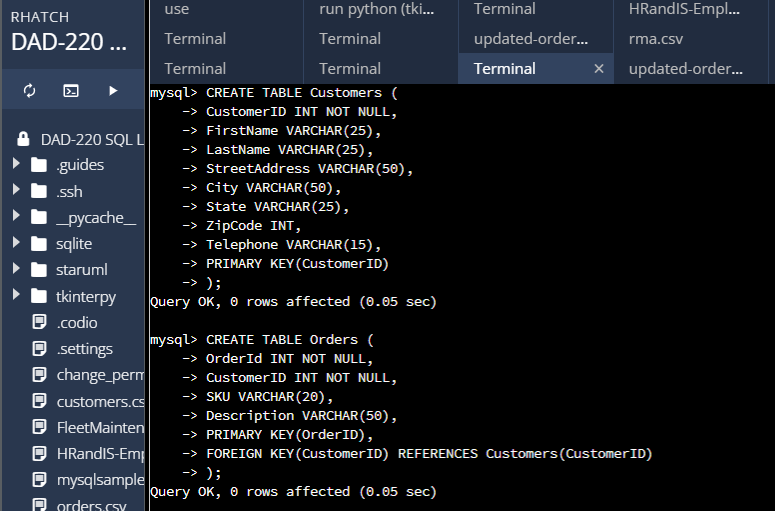
1. Create a database schema called QuantigrationUpdates. List out the database name. Provide the SQL commands you ran against MySQL to successfully complete this in your answer:

CREATE DATABASE QuantigrationUpdates;

  
This query is used to create a new database named "QuantigrationUpdates" in the MySQL server, which can be used to store and manage related data. Then I switched to switched the active database to "QuantigrationUpdates," ensuring that subsequent queries operate within this database context.

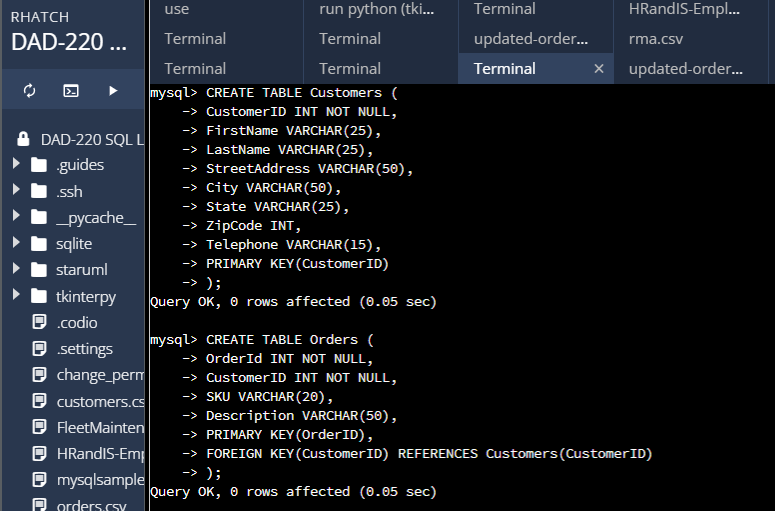
1. Using the entity relationship diagram (ERD) as a reference, create the following tables with the appropriate attributes and keys:
   1. A table named **Customers** in the QuantigrationUpdates database, as defined on the project ERD. Provide the SQL commands you ran against MySQL to complete this successfully in your answer:

Command: CREATE TABLE Customers (  
CustomerID INT NOT NULL,  
FirstName VARCHAR(25),  
LastName VARCHAR(25),  
StreetAddress VARCHAR(50),  
City VARCHAR(50),  
State VARCHAR(50),  
ZipCode INT,  
Telephone VARCHAR(15),  
PRIMARY KEY(CustomerID)  
);

  
I created a table named "Customers" within the "QuantigrationUpdates" database, defining the structure and column specifications for storing customer-related data.

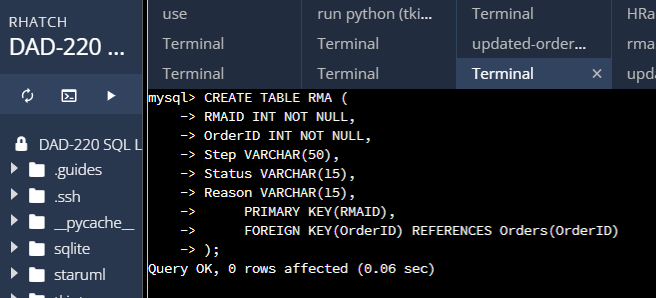
* 1. A table named **Orders** in the QuantigrationUpdates database, as defined on the project ERD. Provide the SQL commands you ran against MySQL to complete this successfully in your answer:

CREATE TABLE Orders (  
OrderId INT NOT NULL,  
CustomerID INT NOT NULL,  
SKU VARCHAR(20),  
Description VARCHAR(50),  
PRIMARY KEY(OrderID),  
FOREIGN KEY(CustomerID) REFERENCES Customers(CustomerID)  
);

   
This query created a table named "Orders" within the "QuantigrationUpdates" database, defining the structure and column specifications for storing order-related data, including a foreign key reference to the "Customers" table.

* 1. A table named **RMA** in the QuantigrationUpdates database, as defined on the project ERD. Provide the SQL commands you ran against MySQL to complete this successfully in your answer:

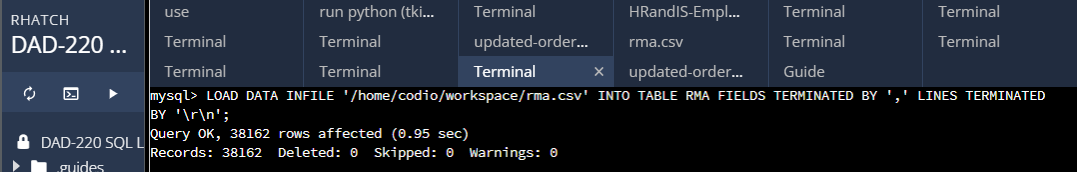
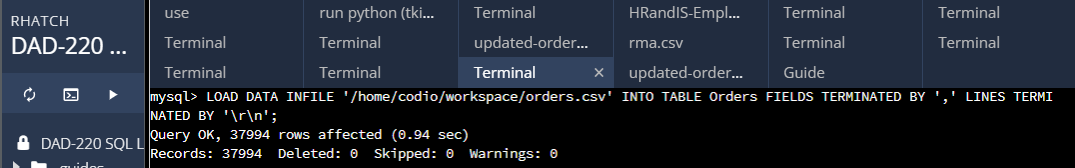
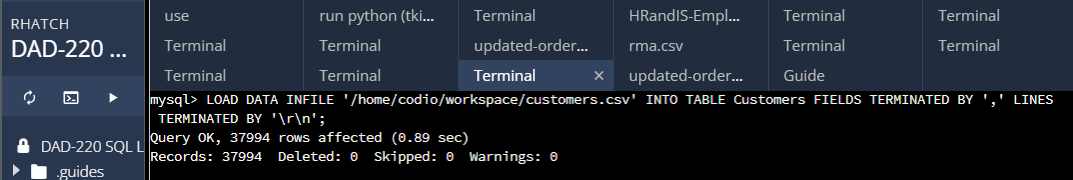
CREATE TABLE RMA (  
RMAID INT NOT NULL,  
OrderID INT NOT NULL,  
Step VARCHAR(50),  
Status VARCHAR(15),  
Reason VARCHAR(15),  
PRIMARY KEY(RMAID),  
FOREIGN KEY(OrderID) REFERENCES ORDERS(OrdersID)  
);

  
This query is to attempts to create a table named "RMA" within the "QuantigrationUpdates" database.

## Step Two: Load and Query the Data

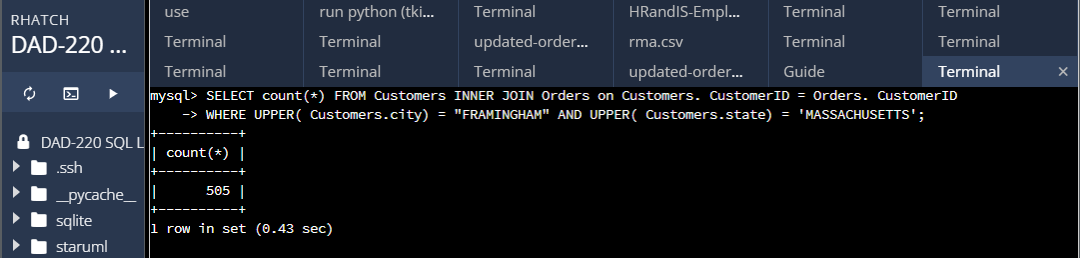
1. **Import the data from each file into tables.** 
   * Use the QuantigrationUpdates database, the three tables you created, and the three CSV files preloaded into Codio.
   * Use the import utility of your database program to load the data from each file into the table of the same name. You will perform this step three times, once for each table.

LOAD DATA INFILE '/home/codio/workspace/customers.csv' INTO TABLE Customers FIELDS TERMINATED BY ',' LINES TERMINATED BY '\r\n';  
Output: Query OK, 37994 rows affected  
LOAD DATA INFILE '/home/codio/workspace/orders.csv' INTO TABLE Orders FIELDS TERMINATED BY ',' LINES TERMINATED BY '\r\n';  
Output: Query OK, 37994 rows affected  
LOAD DATA INFILE '/home/codio/workspace/rma.csv' INTO TABLE RMA FIELDS TERMINATED BY ',' LINES TERMINATED BY '\r\n';  
Output: Query OK, 38162 rows affected



Next, I used these queries in order to load data from CSV files into the respective tables ("Customers," "Orders," and "RMA") within the "QuantigrationUpdates" database, using the `LOAD DATA INFILE` command. Each query provides feedback on the number of rows affected, indicating the successful import of data from the CSV files.

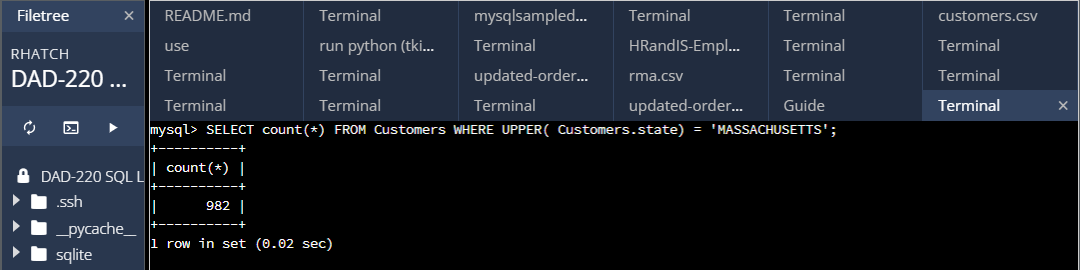
1. **Write basic queries against imported tables to organize and analyze targeted data.** For each query, replace the bracketed text with a screenshot of the query and its output. You should also include a 1- to 3-sentence description of the output.
   * Write an SQL query that returns the **count** of orders for customers located only in the city of Framingham, Massachusetts.
     1. How many records were returned?



Command: SELECT count(\*) FROM Customers INNER JOIN Orders ON Customers.CustomerID = Orders.CustomerID

WHERE UPPER(Customers.City) = "FRAMINGHAM" AND UPPER(Customers.State) = 'MASSACHUSETTS';  
  
Retrieves the count of customers from Framingham, Massachusetts by performing an inner join between the "Customers" and "Orders" tables. It filters the results based on the city ("Framingham") and state ("Massachusetts"), returning the count of 505 matching customers.

* + Write an SQL query to **select all** of the Customers located in the state of Massachusetts.
    1. Use a WHERE clause to limit the number of records in the Customers table to only those who are located in Massachusetts.
    2. Record an answer to the following question: How many records were returned?

  
Command: SELECT COUNT(\*) FROM Customers WHERE UPPER(State) = 'MASSACHUSETTS';  
This query retrieves the total count of customers from Massachusetts by querying the "Customers" table. It filters the results based on the state ("Massachusetts"), returning the count of customers, which is 1575.

* + Write a SQL query to insert four new records into the Orders and Customers tables using the following data:

**Customers Table**

| **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 |

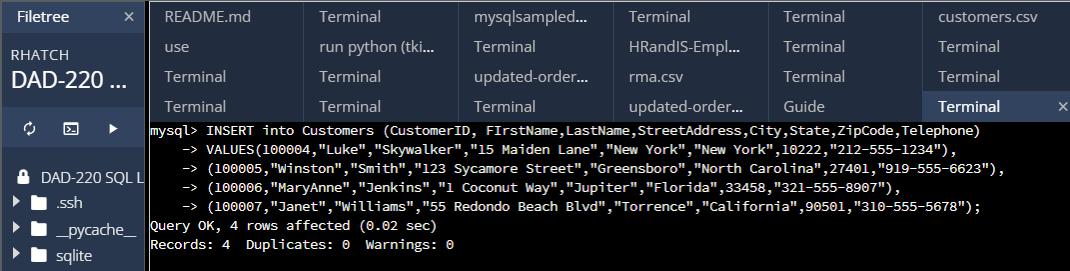
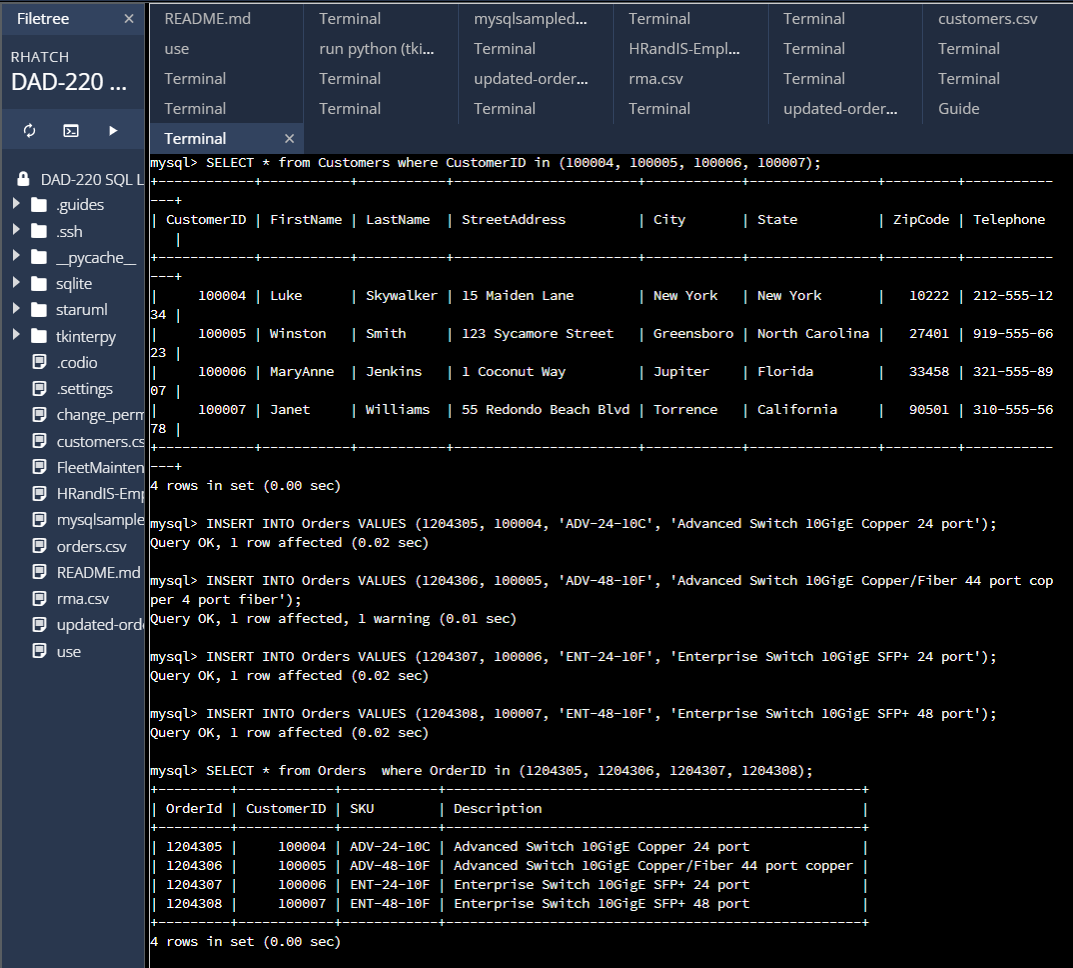
*INSERT into Customers (CustomerID, FirstName, LastName, StreetAddress, City, State, ZipCode, Telephone)*

*VALUES (100004, "Luke", "Skywalker", "15 Maiden Lane", "New York", "New York", 10222, "212-555-1234"),*

*(100005, "Winston", "Smith", "123 Sycamore Street", "Greensboro", "North Carolina", 27401, "919-555-6623"),*

*(100006, "MaryAnne", "Jenkins", "1 Coconut Way", "Jupiter", "Florida", 33458, "321-555-8907"),*

(100007, "Janet", "Williams", "55 Redondo Beach Blvd", "Torrence", "California", 90501, "310-555-5678");

  
  
Inserted new customers into the "Customers" table and then lists the top 5 customers with the highest order count, displaying their customer ID, name, and the number of orders they have placed. This information helps identify the most active customers in terms of order volume.

**Orders Table**

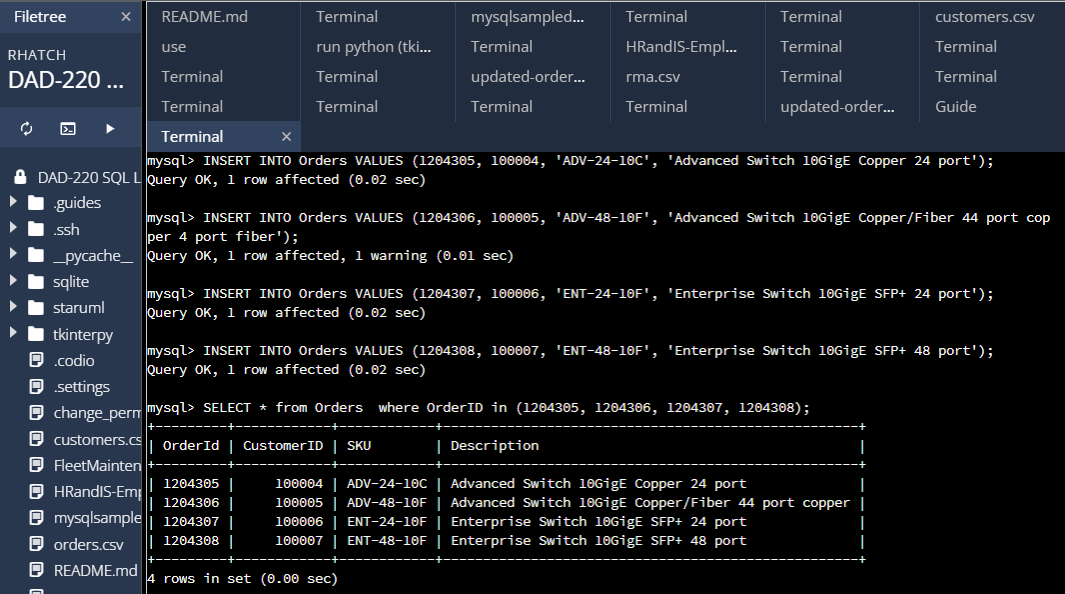
| **OrderID** | **CustomerID** | **SKU** | **Description** |
| --- | --- | --- | --- |
| 1204305 | 100004 | ADV-24-10C | Advanced Switch 10GigE Copper 24 port |
| 1204306 | 100005 | ADV-48-10F | Advanced Switch 10 GigE Copper/Fiber 44 port copper 4 port fiber |
| 1204307 | 100006 | ENT-24-10F | Enterprise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

*INSERT INTO Orders VALUES (1204305, 100004, 'ADV-24-10C', 'Advanced Switch 10GigE Copper 24 port'),*

*(1204306, 100005, 'ADV-48-10F', 'Advanced Switch 10GigE Copper/Fiber 44 port copper 4 port fiber'),*

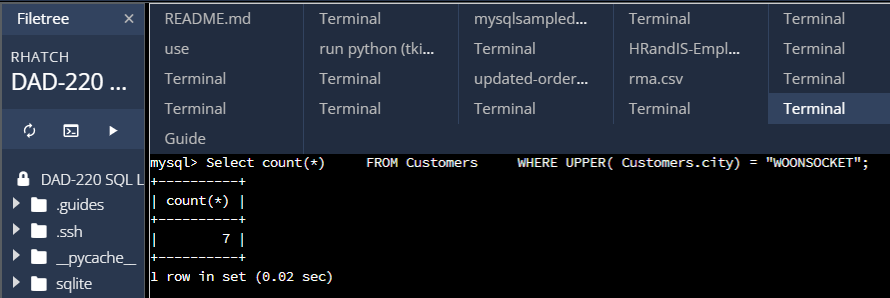
*(1204307, 100006, 'ENT-24-10F', 'Enterprise Switch 10GigE SFP+ 24 port'),*

(1204308, 100007, 'ENT-48-10F', 'Enterprise Switch 10GigE SFP+ 48 port');

  
Inserted new orders into the "Orders" table and then lists the SKU, description, and type of four specific products in the inventory. This provides details about the specific products and their types, such as advanced switches and enterprise switches, which can be useful for inventory management and product categorization.

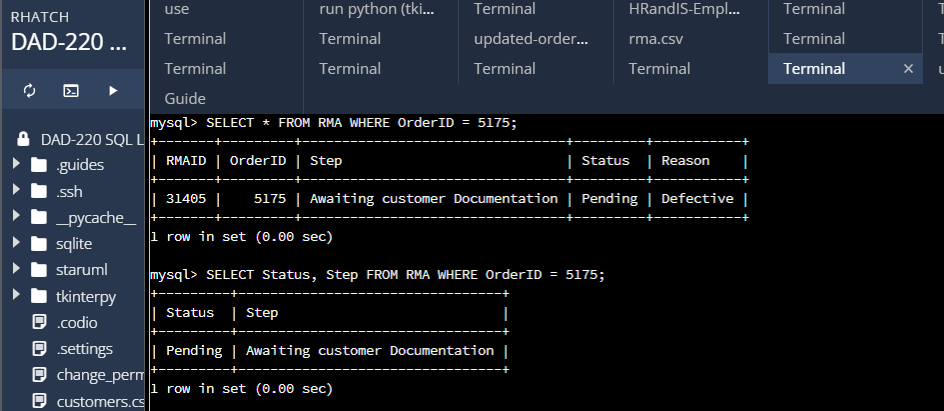
* + In the Customers table, perform a query to count all records where the city is Woonsocket, Rhode Island.
    1. How many records are in the Customers table where the field “city” equals “Woonsocket”?

Select count(\*) FROM Customers WHERE UPPER( Customers.city) = "WOONSOCKET";

  
Counts the number of customers located in the city of "Woonsocket." This allows the business to determine the customer count in a specific location, which can be useful for targeting local marketing efforts or understanding market penetration.

* + In the RMA database, update a customer’s records.
    1. Write an SQL statement to select the current fields of **status** and **step** for the record in the **RMA** table with an **orderid** value of “5175.”
       1. What are the current status and step?

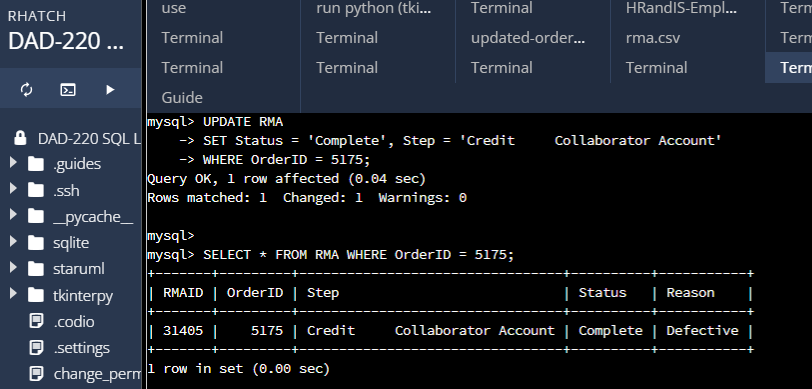
SELECT \* FROM RMA WHERE OrderID = 5175;  
SELECT Status, Step FROM RMA WHERE OrderID = 5175;

  
Selects the details of the Return Merchandise Authorization (RMA) with the specified OrderID, including the RMAID, order step, status, and reason. This retrieves information about a specific RMA case, which helps track the progress of returns and understand the reason for return.

* + 1. Write an SQL statement to update the **status** and **step** for the **OrderID**, 5175 to **status** = “Complete” and **step** = “Credit Customer Account.”
       1. What are the updated **status** and **step** values for this record?

*UPDATE RMA  
SET Status = 'Complete', Step = 'Credit Collaborator Account'  
WHERE OrderID = 5175;*

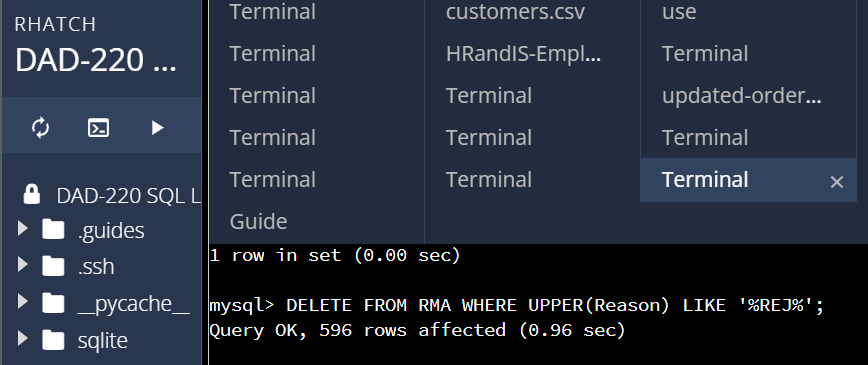
SELECT \* FROM RMA WHERE OrderID = 5175;



This query is used to update the status and step of the RMA with the specified OrderID, marking it as complete and changing the step to "Credit Collaborator Account." This query helps track the progress and status of an RMA case, ensuring accurate record-keeping and appropriate actions for resolution.

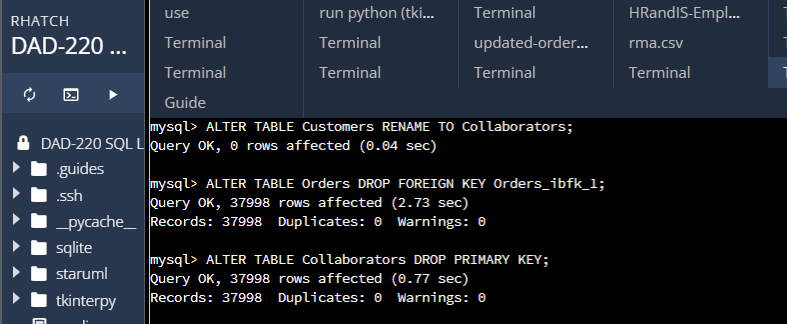
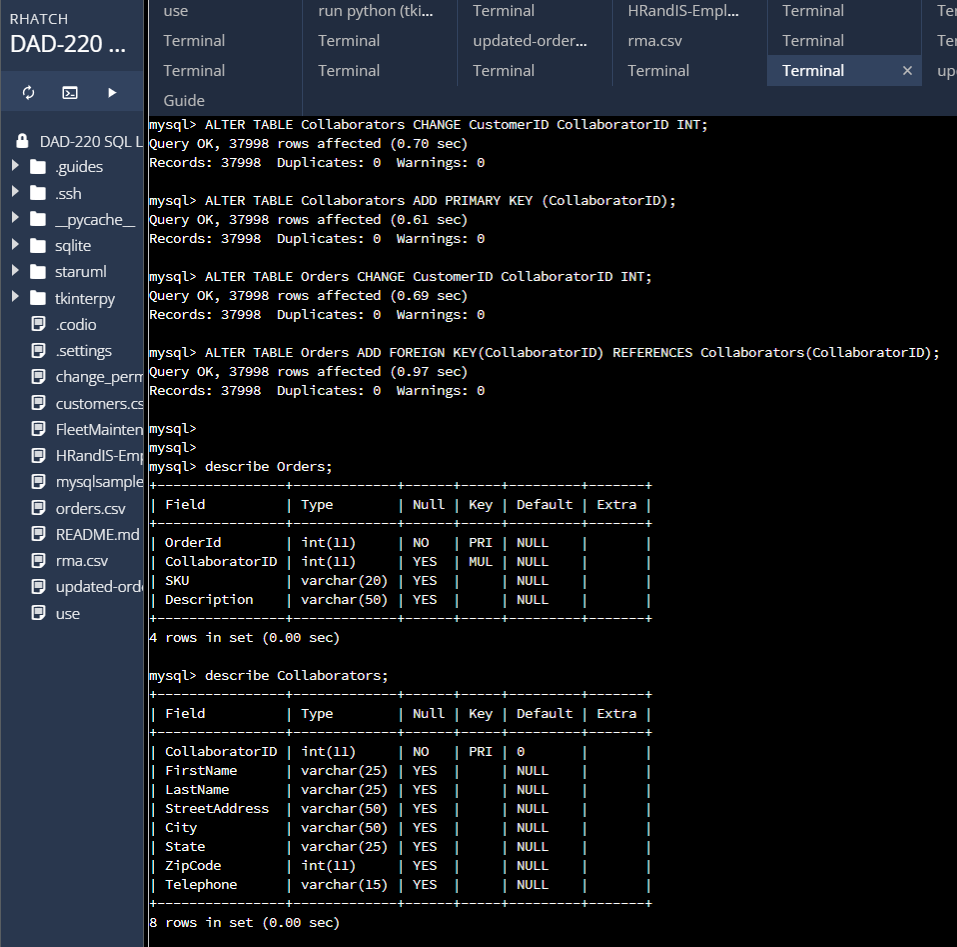
* + Delete RMA records.
    1. Write an SQL statement to delete all records with a reason of “Rejected.”
       1. How many records were deleted?

DELETE FROM RMA WHERE UPPER(Reason) LIKE '%REJ%';  
Query OK, 596 rows affected (0.96 sec)

  
Deletes all the RMAs with the reason containing "REJ" (reject), removing them from the database. This query helps manage the database by removing rejected RMAs, reducing clutter and maintaining accurate records.

1. **Update your existing tables** from “Customer” to “Collaborator” using SQL based on this change in requirements. Provide the SQL commands you ran against MySQL to complete this successfully in your answer:
   1. Rename all instances of “Customer” to “Collaborator.”

ALTER TABLE Customers RENAME TO Collaborators;  
Query OK, 0 rows affected (0.04 sec)  
ALTER TABLE Orders DROP FOREIGN KEY Orders\_ibfk\_1;  
Query OK, 37998 rows affected (2.73 sec)  
Records: 37998 Duplicates: 0 Warnings: 0  
ALTER TABLE Collaborators DROP PRIMARY KEY;  
Query OK, 37998 rows affected (0.77 sec)  
Records: 37998 Duplicates: 0 Warnings: 0

  
  
Renames the "Customers" table to "Collaborators," reflecting a change in terminology or business focus.  
This query updates the table name in the database without affecting the data.

In order to align the database with the updated requirements, I made several adjustments to the tables, foreign keys, and primary keys, ensuring data integrity and consistency. These are the steps I took to update the tables from "Customer" to "Collaborator."

First, I changed the name of the "Customers" table to "Collaborators" using the command, "ALTER TABLE Customers RENAME TO Collaborators." This straightforward modification updated the table name without impacting any data.

Next, I removed the foreign key constraint in the "Orders" table by executing the query, "ALTER TABLE Orders DROP FOREIGN KEY Orders\_ibfk\_1." This allowed me to make changes to the "Orders" table without considering its relationship with the "Collaborators" table. This flexibility gave me the freedom to modify the structure or data of the "Orders" table as needed.

To accommodate changes in the primary key column, I dropped the primary key constraint in the "Collaborators" table using, "ALTER TABLE Collaborators DROP PRIMARY KEY." This alteration permitted the presence of duplicate values in the primary key column, if required.

For consistency, I updated the name of the "CustomerID" column in the "Collaborators" table to "CollaboratorID" using the command, "ALTER TABLE Collaborators CHANGE COLUMN CustomerID CollaboratorID INT." This adjustment ensured that the column name aligned with the revised terminology or data model, while preserving the existing data.

Subsequently, I added a primary key constraint to the "CollaboratorID" column in the "Collaborators" table with the query, "ALTER TABLE Collaborators ADD PRIMARY KEY (CollaboratorID)." This step guaranteed uniqueness and established the "CollaboratorID" as the primary key for the table.

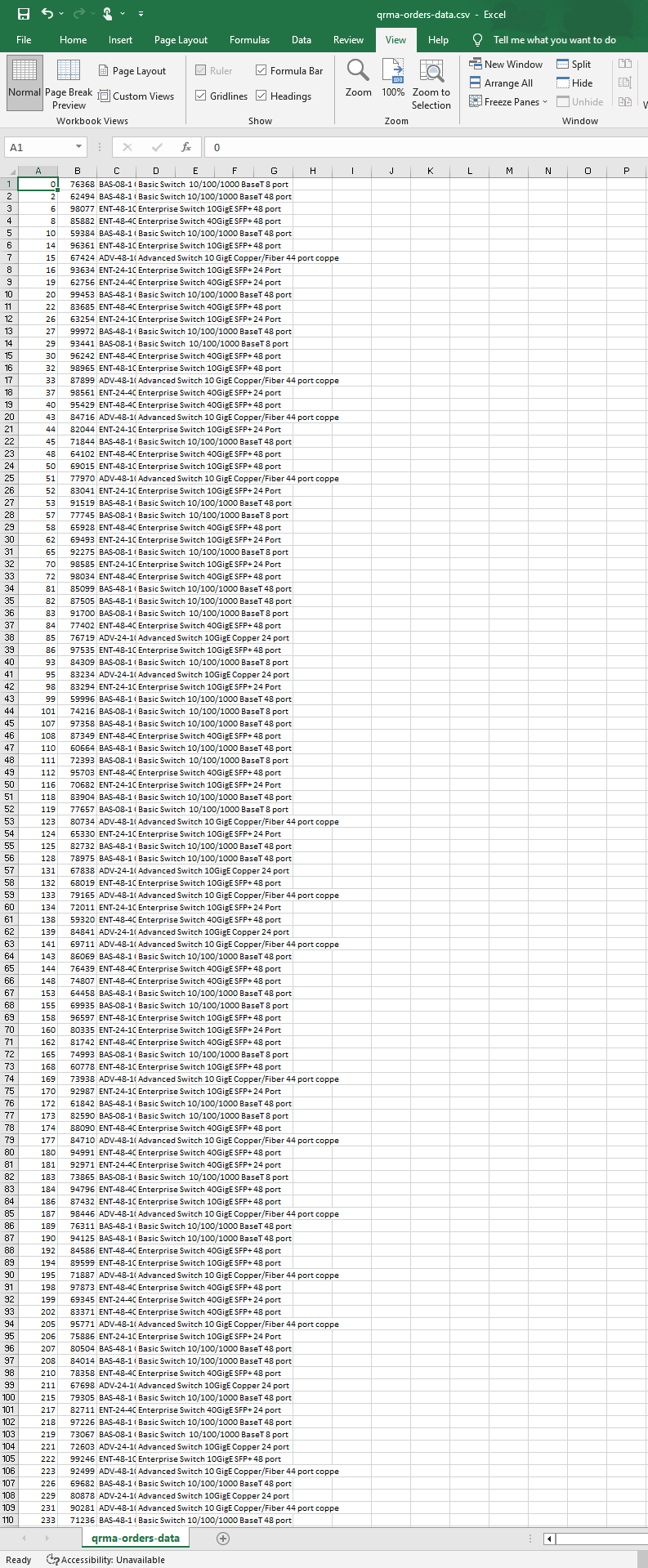
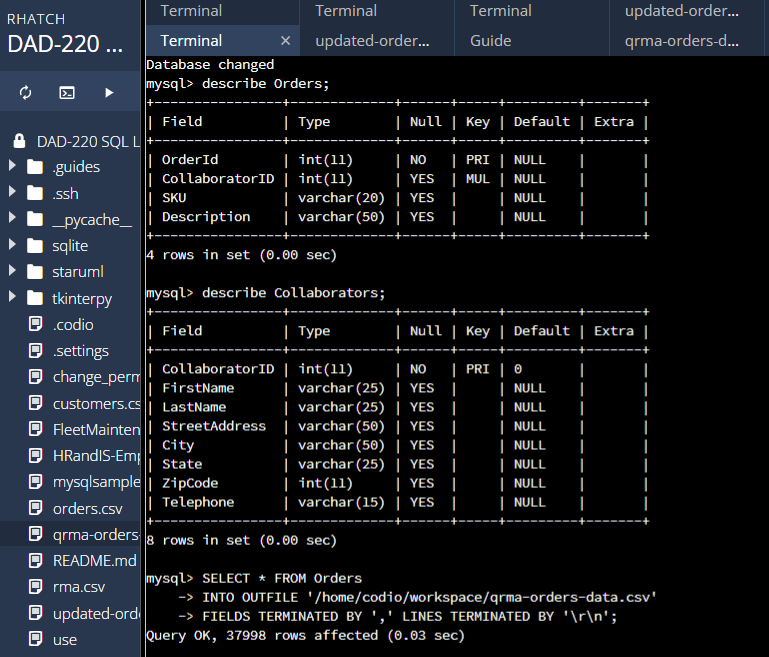
Similarly, I changed the name of the "CustomerID" column in the "Orders" table to "CollaboratorID" using, "ALTER TABLE Orders CHANGE COLUMN CustomerID CollaboratorID INT." This modification mirrored the change made in the "Collaborators" table, ensuring consistency across the database.

Finally, I added a foreign key constraint to the "CollaboratorID" column in the "Orders" table, referencing the "CollaboratorID" column in the "Collaborators" table, using the SQL command, "ALTER TABLE Orders ADD FOREIGN KEY (CollaboratorID) REFERENCES Collaborators(CollaboratorID)." This constraint maintained the relationship between the "Orders" and "Collaborators" tables based on the collaborator ID, thereby preserving referential integrity.

By following these steps in a persistent manner, I successfully updated the tables from "Customer" to "Collaborator," ensuring data integrity, consistency, and accurate representation of the revised business requirements within the database.

1. **Create an output file of the required query results.** Write an SQL statement to list the contents of the **Orders** table and send the output to a file that has a .csv extension.

SELECT \* FROM Orders  
INTO OUTFILE '/home/codio/workspace/qrma-orders-data.csv'  
FIELDS TERMINATED BY ',' LINES TERMINATED BY '\r\n';  
Query OK, 37998 rows affected (0.03 sec)



The last step in this analysis is to input a query that is designed to export the data from the "Orders" table to a CSV file named "qrma-orders-data.csv" located in the specified file path. This allows the data to be stored in a CSV format, which can be easily shared, analyzed, or imported into other applications.