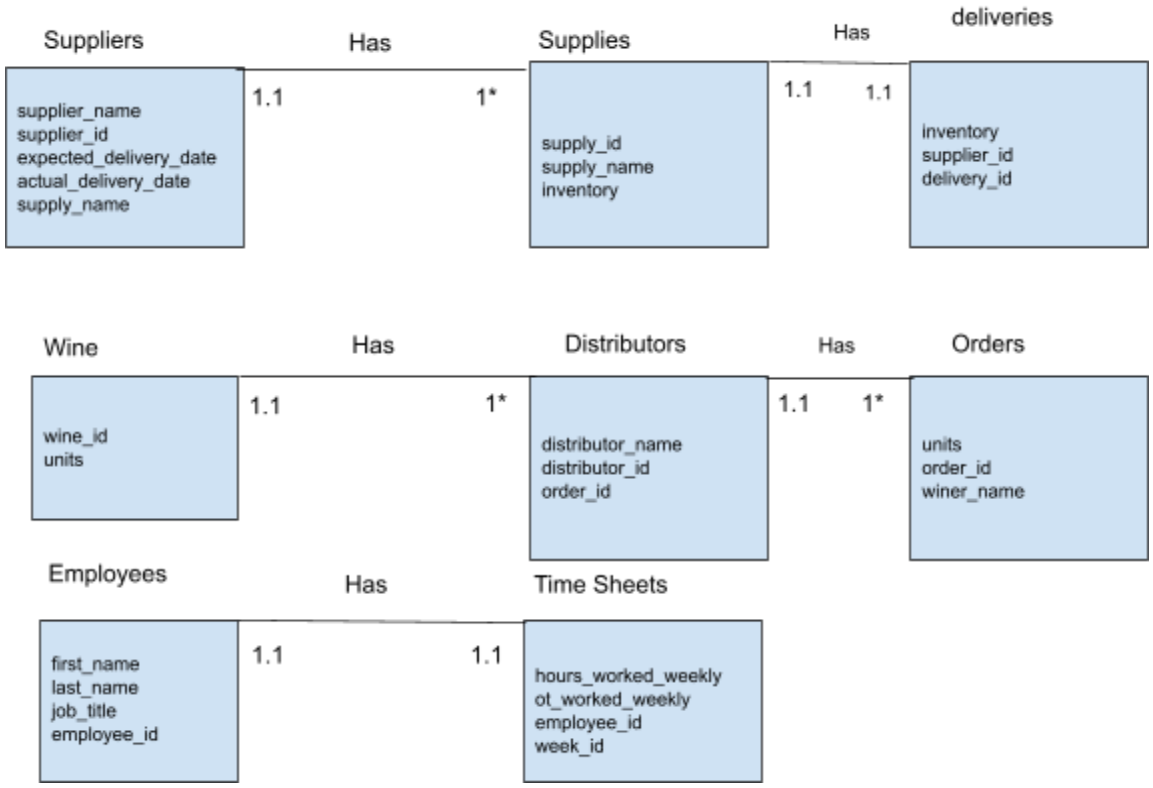
Rachel Cewe, Elizabeth Fung, Scott Jenks, Clint Steadman
12/1/2022
Module 10.1 - Milestone #2

Group 2
Bacchus Winery Case Study

Revised ERD



Rachel Cewe, Elizabeth Fung, Scott Jenks, Clint Steadman
12/1/2022
Module 10.1 - Milestone #2

Created a new database called bacchus_winery in MySQL Command Line Client

INITIAL PYTHON SCRIPT

The following code is from the example from
https://dev.mysql.com/doc/connector-python/en/connector-python-example-ddl.html

```
from __future__ import print_function
import mysql.connector
from mysql.connector import errorcode

config = {
   'user': 'root',
   'password': 'P!aybill!985',
   'host': 'localhost',
   'database': 'bacchus_winery',
   'raise_on_warnings': True
}
try:
   mydb = mysql.connector.connect(**config)

   print("\n Database user {} connected to MySQL on host {} with database {}".format(
      config["user"], config["host"], config["database"]))
   input("\n\n Press any key to continue...\n")

except mysql.connector.Error as err:
   if err.errno == errorcode.ER_ACCESS_DENIED_ERROR:
      print(" The supplied username or password are invalid")
   elif err.errno == errorcode.ER_BAD_DB_ERROR:
      print(" The specified database does not exist")
   else:
      print(err)

DB_NAME = 'bacchus_winery'

myCursor = mydb.cursor()


# Function to erase tables if they exist to start with a clean database
def drop_tables(myCursor):
   tables = "DROP TABLE wine"
```

```
    myCursor.execute(tables)
    mydb.commit()


drop_tables(myCursor)


TABLES = {}
TABLES['wine'] = (
    "CREATE TABLE wine"
    "  (wine_id int NOT NULL AUTO_INCREMENT KEY,"
    "  wine_name varchar(25) NOT NULL,"
    "  units int NOT NULL,"
    "  batch_month varchar(25) NOT NULL )")

"'TABLES['orders'] = (
    "CREATE TABLE orders"
    "  (order_id int NOT NULL AUTO_INCREMENT PRIMARY KEY,"
    "  units int NOT NULL,"
    "  wine_nam varchar(25) NOT NULL,"
    "  CONSTRAINT fk_wine FOREIGN KEY (units) REFERENCES wine(units))")

TABLES['distributors'] = (
    "CREATE TABLE distributors"
    "  (distributor_id int NOT NULL AUTO_INCREMENT PRIMARY KEY,"
    "  distributor_name varchar(25) NOT NULL,"
    "  order_id int NOT NULL,"
    "  CONSTRAINT fk_orders FOREIGN KEY (order_id) REFERENCES orders (order_id))")

TABLES['suppliers'] = (
    "CREATE TABLE suppliers"
    "  (supplier_id int NOT NULL AUTO_INCREMENT PRIMARY KEY,"
    "  supplier_name varchar(25) NOT NULL,"
    "  expected_delivery_date DATE NOT NULL,"
    "  actual_delivery_date DATE NOT NULL,"
    "  supply_name varchar(25) NOT NULL)")

TABLES['deliveries'] = (
    "CREATE TABLE deliveries"
    "  (delivery_id int NOT NULL AUTO_INCREMENT PRIMARY KEY,"
    "  supplier_id int NOT NULL,"
    "  inventory int NOT NULL,"
```

```
    "  CONSTRAINT fk_suppliers FOREIGN KEY (supplier_id) REFERENCES
suppliers(supplier_id))")

TABLES['supplies'] = (
    "CREATE TABLE supplies"
    "  (supply_id int NOT NULL PRIMARY KEY,"
    "  supply_name varchar(25) NOT NULL,"
    "  inventory int NOT NULL,"
    "  CONSTRAINT fk_suppliers FOREIGN KEY (supply_name) REFERENCES
suppliers(supplier_name),"
    "  CONSTRAINT fk_deliveries FOREIGN KEY (inventory) REFERENCES
deliveries(inventory))")

TABLES['employees'] = (
    "CREATE TABLE employees"
    "  (employee_id int NOT NULL AUTO_INCREMENT PRIMARY KEY,"
    "  first_name varchar(25) NOT NULL,"
    "  last_name varchar(25) NOT NULL,"
    "  job_title varchar(25) NOT NULL)")

TABLES['time_sheet'] = (
    "CREATE TABLE time_sheet"
    "  (week_id int NOT NULL PRIMARY KEY,"
    "  hours_worked_weekly int NOT NULL,"
    "  ot_worked_weekly int NOT NULL,"
    "  employee_id int NOT NULL,"
    "  CONSTRAINT fk_employees FOREIGN KEY (employee_id) REFERENCES employees
(employee_id))")
'''
for table_name in TABLES:
    table_description = TABLES[table_name]
    try:
        print("Creating table {}: ".format(table_name), end='')
        myCursor.execute(table_description)
    except mysql.connector.Error as err:
        if err.errno == errorcode.ER_TABLE_EXISTS_ERROR:
            print("already exists.")
        else:
            print(err.msg)
    else:
        print("OK\n")

# Inset values into wine table
```

```
sql1 = "INSERT INTO wine (wine_name, units, batch_month) VALUE ('Merlot', 50, 'January')"
sql2 = "INSERT INTO wine (wine_name, units, batch_month) VALUE ('Cabernet', 40,
'February')"
sql3 = "INSERT INTO wine (wine_name, units, batch_month) VALUE ('Chablis', 60, 'April')"
sql4 = "INSERT INTO wine (wine_name, units, batch_month) VALUE ('Chardonnay', 50,
'March')"
myCursor.execute(sql1)
myCursor.execute(sql2)
myCursor.execute(sql3)
myCursor.execute(sql4)
mydb.commit()

#Print out values
print("-Displaying Wine Records-")
query1 = "SELECT wine_name, units, batch_month FROM wine"
myCursor.execute(query1)
wines = myCursor.fetchall()
for wine in wines:
   print("Wine Name: {}\nNumber of units: {}\nBatch Month: {}\n".format(wine[0], wine[1],
                                                                        wine[2]))
```

REVISED PYTHON SCRIPT
(WORK IN PROGRESS)

```
from __future__ import print_function
import mysql.connector
from mysql.connector import errorcode

config = {
   'user': 'root',
   'password': 'Studman081!',
   'host': 'localhost',
   'database': 'bacchus_winery',
   'raise_on_warnings': True
}
try:
   mydb = mysql.connector.connect(**config)

   print("\n Database user {} connected to MySQL on host {} with database
{}".format(
       config["user"], config["host"], config["database"]))
   input("\n\n Press any key to continue...\n")

except mysql.connector.Error as err:
```

```
    if err.errno == errorcode.ER_ACCESS_DENIED_ERROR:
        print(" The supplied username or password are invalid")
    elif err.errno == errorcode.ER_BAD_DB_ERROR:
        print(" The specified database does not exist")
    else:
        print(err)

DB_NAME = 'bacchus_winery'

myCursor = mydb.cursor()

# Function to erase tables if they exist to start with a clean database
'''def drop_tables(myCursor):
    tables1 = "DROP TABLE wine"
    myCursor.execute(tables1)
    mydb.commit()


drop_tables(myCursor)'''

TABLES = {}
TABLES['wine'] = (
    "CREATE TABLE wine"
    "  (wine_id int NOT NULL AUTO_INCREMENT PRIMARY KEY,"
    "  wine_name varchar(25) NOT NULL,"
    "  units int NOT NULL,"
    "  batch_month varchar(25) NOT NULL)")

TABLES['orders'] = (
    "CREATE TABLE orders"
    "  (order_id int NOT NULL AUTO_INCREMENT PRIMARY KEY,"
    "  units int NOT NULL,"
    "  wine_name varchar(25) NOT NULL)")

TABLES['distributors'] = (
    "CREATE TABLE distributors"
    "  (distributor_id int NOT NULL AUTO_INCREMENT PRIMARY KEY,"
    "  distributor_name varchar(25) NOT NULL,"
    "  order_id int NOT NULL)")

TABLES['suppliers'] = (
    "CREATE TABLE suppliers"
    "  (supplier_id int NOT NULL AUTO_INCREMENT PRIMARY KEY,"
    "  supplier_name varchar(25) NOT NULL,"
    "  expected_delivery_date DATE NOT NULL,"
    "  actual_delivery_date DATE NOT NULL,"
    "  supply_name varchar(25) NOT NULL)")
```

```
TABLES['deliveries'] = (
   "CREATE TABLE deliveries"
   "  (delivery_id int NOT NULL AUTO_INCREMENT PRIMARY KEY,"
   "  supplier_id int NOT NULL,"
   "  inventory int NOT NULL)")

TABLES['supplies'] = (
   "CREATE TABLE supplies"
   "  (supply_id int NOT NULL PRIMARY KEY,"
   "  supply_name varchar(25) NOT NULL,"
   "  inventory int NOT NULL)")

TABLES['employees'] = (
   "CREATE TABLE employees"
   "  (employee_id int NOT NULL AUTO_INCREMENT PRIMARY KEY,"
   "  first_name varchar(25) NOT NULL,"
   "  last_name varchar(25) NOT NULL,"
   "  job_title varchar(25) NOT NULL)")

TABLES['time_sheet'] = (
   "CREATE TABLE time_sheet"
   "  (week_id int NOT NULL PRIMARY KEY,"
   "  hours_worked_weekly int NOT NULL,"
   "  ot_worked_weekly int NOT NULL,"
   "  employee_id int NOT NULL)")

for table_name in TABLES:
   table_description = TABLES[table_name]
   try:
       print("Creating table {}: ".format(table_name), end='')
       myCursor.execute(table_description)
   except mysql.connector.Error as err:
       if err.errno == errorcode.ER_TABLE_EXISTS_ERROR:
           print("already exists.")
       else:
           print(err.msg)
   else:
       print("OK\n")

# Insert values into wine table
sql1 = "INSERT INTO wine (wine_name, units, batch_month) VALUE ('Merlot', 50,
'January')"
sql2 = "INSERT INTO wine (wine_name, units, batch_month) VALUE ('Cabernet', 40,
'February')"
sql3 = "INSERT INTO wine (wine_name, units, batch_month) VALUE ('Chablis', 60,
'April')"
sql4 = "INSERT INTO wine (wine_name, units, batch_month) VALUE ('Chardonnay',
50, 'March')"
```

```
myCursor.execute(sql1)
myCursor.execute(sql2)
myCursor.execute(sql3)
myCursor.execute(sql4)
mydb.commit()

# Print out values
print("-Displaying Wine Records-")
query1 = "SELECT wine_name, units, batch_month FROM wine"
myCursor.execute(query1)
wines = myCursor.fetchall()
for wine in wines:
   print("Wine Name: {}\nNumber of units: {}\nBatch Month:
{}\n".format(wine[0], wine[1],

wine[2]))

# Insert values into Orders table
orders1 = "INSERT INTO orders (units, wine_name) VALUE (10, 'Merlot')"
orders2 = "INSERT INTO orders (units, wine_name) VALUE (15, 'Cabernet')"
orders3 = "INSERT INTO orders (units, wine_name) VALUE (12, 'Chablis')"
orders4 = "INSERT INTO orders (units, wine_name) VALUE (10, 'Chardonnay')"
myCursor.execute(orders1)
myCursor.execute(orders2)
myCursor.execute(orders3)
myCursor.execute(orders4)
mydb.commit()

print("-Displaying Orders-")
query2 = "SELECT units, wine_name FROM orders"
myCursor.execute(query2)
orders = myCursor.fetchall()
for order in orders:
   print("Number of units: {}\nWine Name: {}\n".format(order[0], order[1]))

# Insert values into Distributors table
dis1 = "INSERT INTO distributors (distributor_name, order_id) VALUE ('Wines to
GO', 2)"
dis2 = "INSERT INTO distributors (distributor_name, order_id) VALUE ('Partners
in Wine', 3)"
dis3 = "INSERT INTO distributors (distributor_name, order_id) VALUE ('Sip
Happens Co.', 1)"
myCursor.execute(dis1)
myCursor.execute(dis2)
myCursor.execute(dis3)
mydb.commit()

print("-Displaying Distributors-")
```

```
query3 = "SELECT distributor_name, order_id FROM distributors"
myCursor.execute(query3)
distributors = myCursor.fetchall()
for distributor in distributors:
   print("Distributor Name: {}\nOrder ID: {}\n".format(distributor[0],
distributor[1]))

# Insert values into Suppliers table
sup1 = "INSERT INTO suppliers (supplier_name, expected_delivery_date,
actual_delivery_date, supply_name)" \
      "VALUE ('The New Corker', 20220110, 20220115, 'Corks')"
sup2 = "INSERT INTO suppliers (supplier_name, expected_delivery_date,
actual_delivery_date, supply_name)" \
      "VALUE ('Boxes n Stuff', 20220215, 20220215, 'Boxes')"
sup3 = "INSERT INTO suppliers (supplier_name, expected_delivery_date,
actual_delivery_date, supply_name)" \
      "VALUE ('Wine Supply', 20220320, 20220415, 'Tubing')"
myCursor.execute(sup1)
myCursor.execute(sup2)
myCursor.execute(sup3)
mydb.commit()

# Print out values
print("-Displaying Suppliers Records-")
query4 = "SELECT supplier_name, expected_delivery_date, actual_delivery_date,
supply_name FROM suppliers"
myCursor.execute(query4)
suppliers = myCursor.fetchall()
for supplier in suppliers:
   print("supplier_name: {}\nexpected_delivery_date: {}\nactual_delivery_date:
{}\nsupply_name: {}\n".format(
       supplier[0], supplier[1], supplier[2], supplier[3]))

# Insert values into deliveries table
del1 = "INSERT INTO deliveries (supplier_id, inventory) VALUE (1, 50)"
del2 = "INSERT INTO deliveries (supplier_id, inventory) VALUE (2, 40)"
del3 = "INSERT INTO deliveries (supplier_id, inventory) VALUE (3, 60)"
myCursor.execute(del1)
myCursor.execute(del2)
myCursor.execute(del3)
mydb.commit()

# Print out values
print("-Displaying deliveries Records-")
query5 = "SELECT supplier_id, inventory FROM deliveries"
myCursor.execute(query5)
deliveries = myCursor.fetchall()
for delivery in deliveries:
```

```
   print("Supplier ID: {}\nInventory: {}\n".format(delivery[0], delivery[1]))

# Insert values into supplies table
spl1 = "INSERT INTO supplies (supply_name, inventory) VALUE ('Corks', 50)"
spl2 = "INSERT INTO supplies (supply_name, inventory) VALUE ('Boxes', 40)"
myCursor.execute(spl1)
myCursor.execute(spl2)
mydb.commit()

# Print out values
print("-Displaying Supplies Records-")
query6 = "SELECT supply_name, inventory FROM supplies"
myCursor.execute(query6)
supplies = myCursor.fetchall()
for supply in supplies:
   print("Supply Name: {}\nInventory: {}\n".format(supply[0], supply[1]))

# Insert values into employees table
emp1 = "INSERT INTO employees (first_name, last_name, job_title) VALUE ('Stan',
'Bacchus', 'Co-Owner')"
emp2 = "INSERT INTO employees (first_name, last_name, job_title) VALUE
('Davis', 'Bacchus', 'Co-Owner')"
emp3 = "INSERT INTO employees (first_name, last_name, job_title) VALUE
('Janet', 'Collins', 'Finances and Payroll')"
emp4 = "INSERT INTO employees (first_name, last_name, job_title) VALUE ('Roz',
'Murphy', 'Marketing Department')"
emp5 = "INSERT INTO employees (first_name, last_name, job_title) VALUE
('Henry', 'Doyle', 'Production Line')"
emp6 = "INSERT INTO employees (first_name, last_name, job_title) VALUE
('Maria', 'Costanza', 'Distribution')"
myCursor.execute(emp1)
myCursor.execute(emp2)
myCursor.execute(emp3)
myCursor.execute(emp4)
myCursor.execute(emp5)
myCursor.execute(emp6)
mydb.commit()

# Print out values
print("-Displaying Employee Records-")
query7 = "SELECT first_name, last_name, job_title FROM employees"
myCursor.execute(query7)
employees = myCursor.fetchall()
for employee in employees:
   print("First Name: {}\nLast Name: {}\nJob Title: {}\n".format(employee[0],
employee[1],
                                                                employee[2]))
```

```
# Insert values into time sheet table
time1 = "INSERT INTO time_sheet (week_id, hours_worked_weekly,
ot_worked_weekly, employee_id) VALUE (1,45, 5, 1)"
time2 = "INSERT INTO time_sheet (week_id, hours_worked_weekly,
ot_worked_weekly, employee_id) VALUE (2, 50, 10, 2)"
time3 = "INSERT INTO time_sheet (week_id, hours_worked_weekly,
ot_worked_weekly, employee_id) VALUE (3, 32, 0, 3)"
time4 = "INSERT INTO time_sheet (week_id, hours_worked_weekly,
ot_worked_weekly, employee_id) VALUE (4, 45, 5, 4)"
time5 = "INSERT INTO time_sheet (week_id, hours_worked_weekly,
ot_worked_weekly, employee_id) VALUE (5, 60, 20, 5)"
time6 = "INSERT INTO time_sheet (week_id, hours_worked_weekly,
ot_worked_weekly, employee_id) VALUE (6, 40, 0, 6)"
myCursor.execute(time1)
myCursor.execute(time2)
myCursor.execute(time3)
myCursor.execute(time4)
myCursor.execute(time5)
myCursor.execute(time6)
mydb.commit()

# Print out values
print("-Displaying Time Sheet Records-")
query8 = "SELECT week_id, hours_worked_weekly, ot_worked_weekly, employee_id
FROM time_sheet"
myCursor.execute(query8)
time_sheet = myCursor.fetchall()
for time in time_sheet:
   print("Week ID: {}\nHours worked weekly: {}\nOT hours worked weekly:
{}\nemployee ID: {}\n".format(time[0], time[1],

time[2],

time[3]))
```

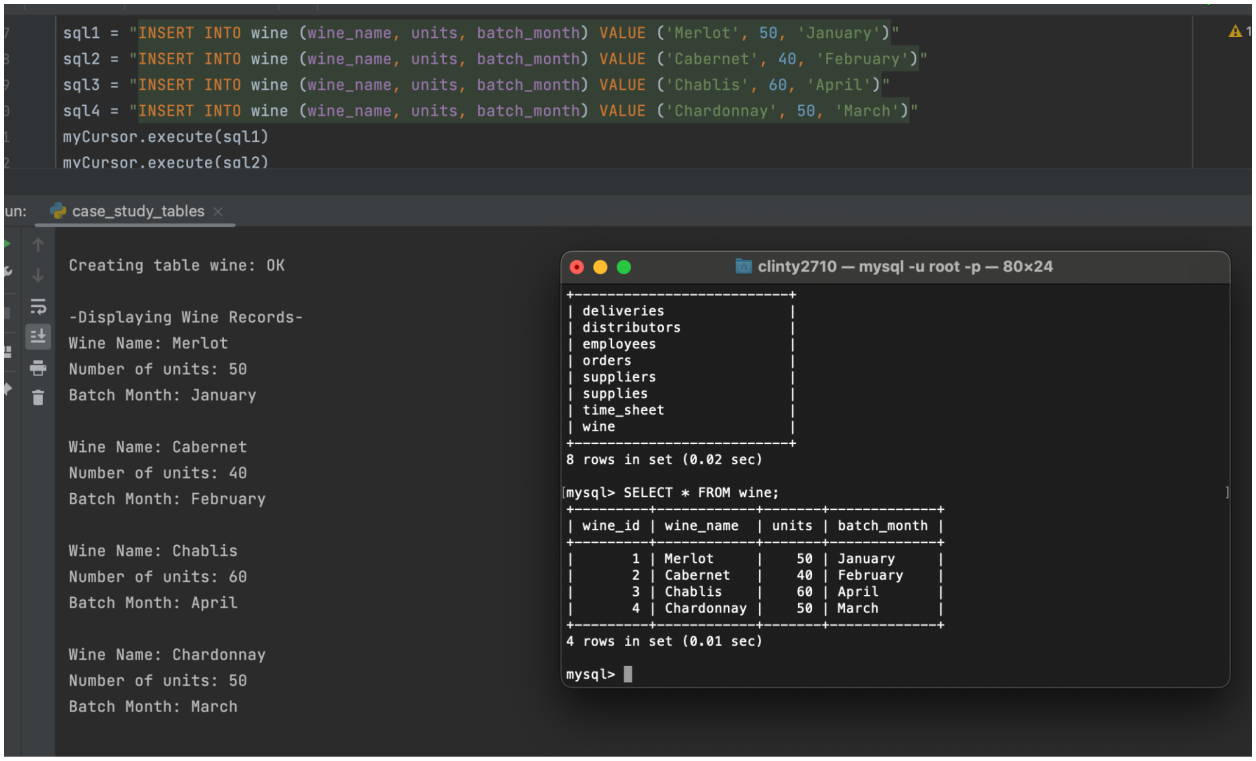
Rachel Cewe, Elizabeth Fung, Scott Jenks, Clint Steadman
12/1/2022
Module 10.1 - Milestone #2

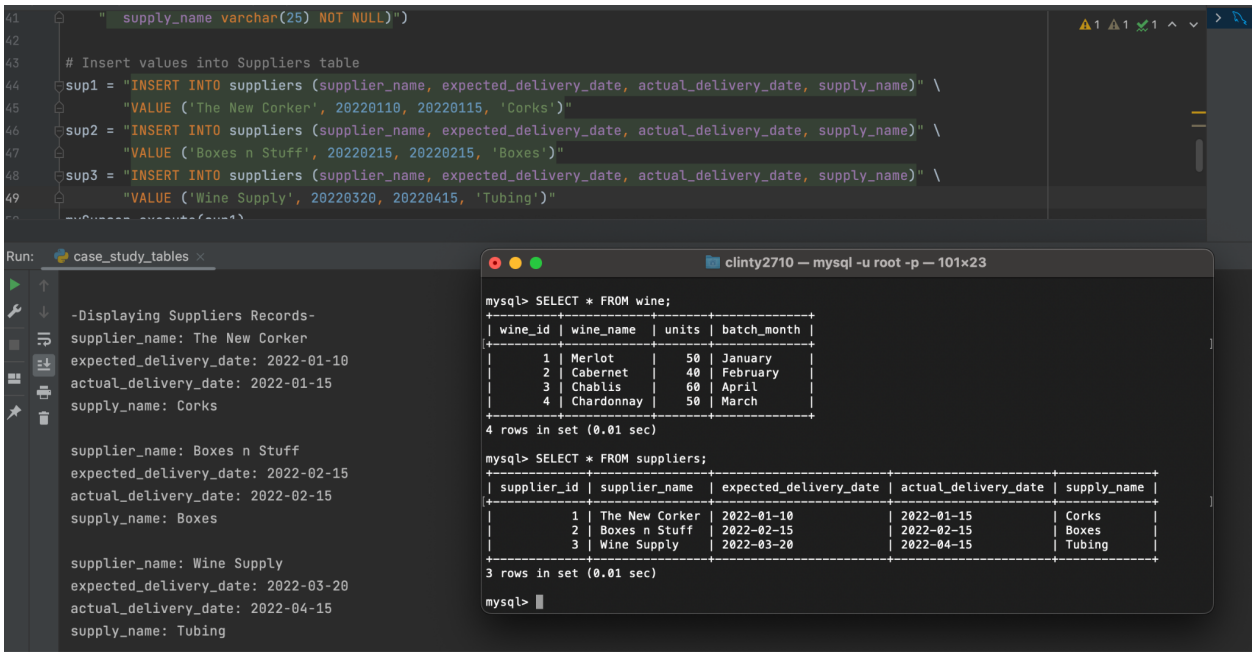
```
+--------------------------+
| Tables_in_bacchus_winery |
+--------------------------+
| deliveries               |
| distributors             |
| employees                |
| orders                   |
| suppliers                |
| supplies                 |
| time_sheet               |
| wine                     |
+--------------------------+
8 rows in set (0.01 sec)

mysql>
```

```
Run:  case_study_tables
Creating table time_sheet: already exists.
-Displaying Employee Records-
First Name: Stan
Last Name: Bacchus
Job Title: Co-Owner

First Name: Davis
Last Name: Bacchus
Job Title: Co-Owner

First Name: Janet
Last Name: Collins
Job Title: Finances and Payroll

First Name: Roz
Last Name: Murphy
Job Title: Marketing Department

First Name: Henry
Last Name: Doyle
Job Title: Production Line

First Name: Maria
Last Name: Costanza
Job Title: Distribution


Process finished with exit code 0
```

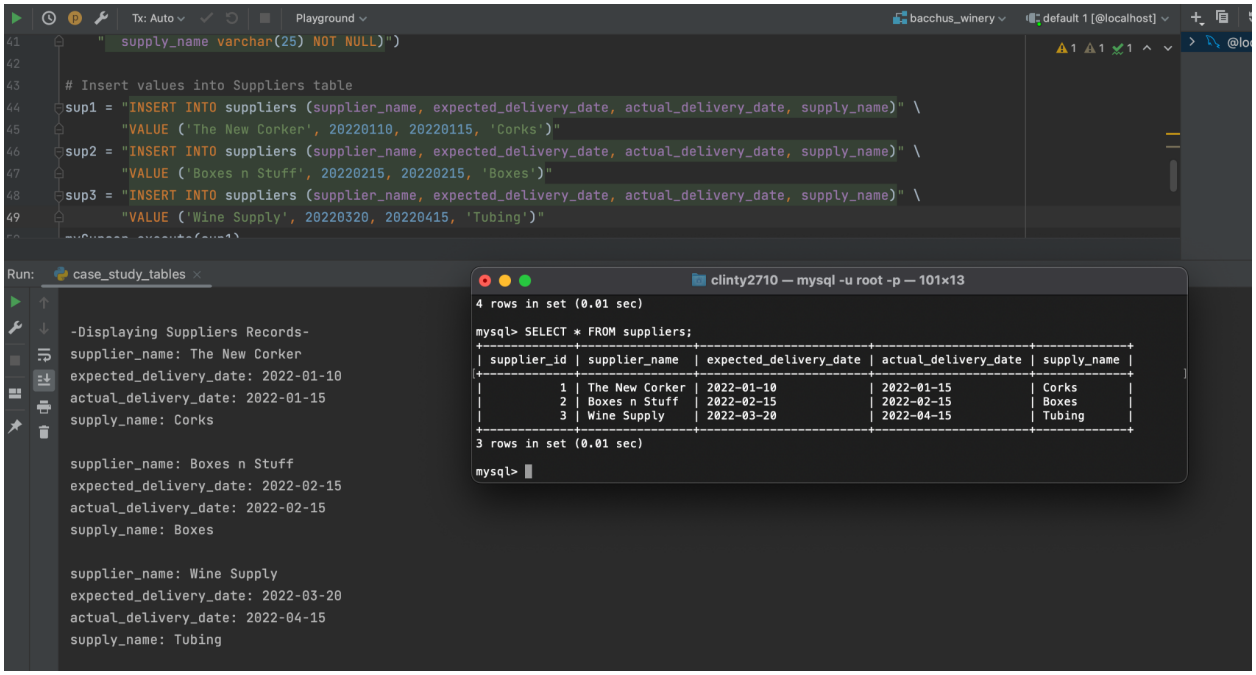
```
|          23 | Henry       | Doyle     | Production Line      |
|          24 | Maria       | Costanza  | Distribution         |
+-------------+-------------+-----------+----------------------+
24 rows in set (0.00 sec)

mysql> SELECT * FROM employees;
+-------------+-------------+-----------+----------------------+
| employee_id | first_name  | last_name | job_title            |
+-------------+-------------+-----------+----------------------+
|           1 | Stan        | Bacchus   | Co-Owner             |
|           2 | Davis       | Bacchus   | Co-Owner             |
|           3 | Janet       | Collins   | Finances and Payroll |
|           4 | Roz         | Murphy    | Marketing Department |
|           5 | Henry       | Doyle     | Production Line      |
|           6 | Maria       | Costanza  | Distribution         |
+-------------+-------------+-----------+----------------------+
6 rows in set (0.00 sec)

mysql>
```

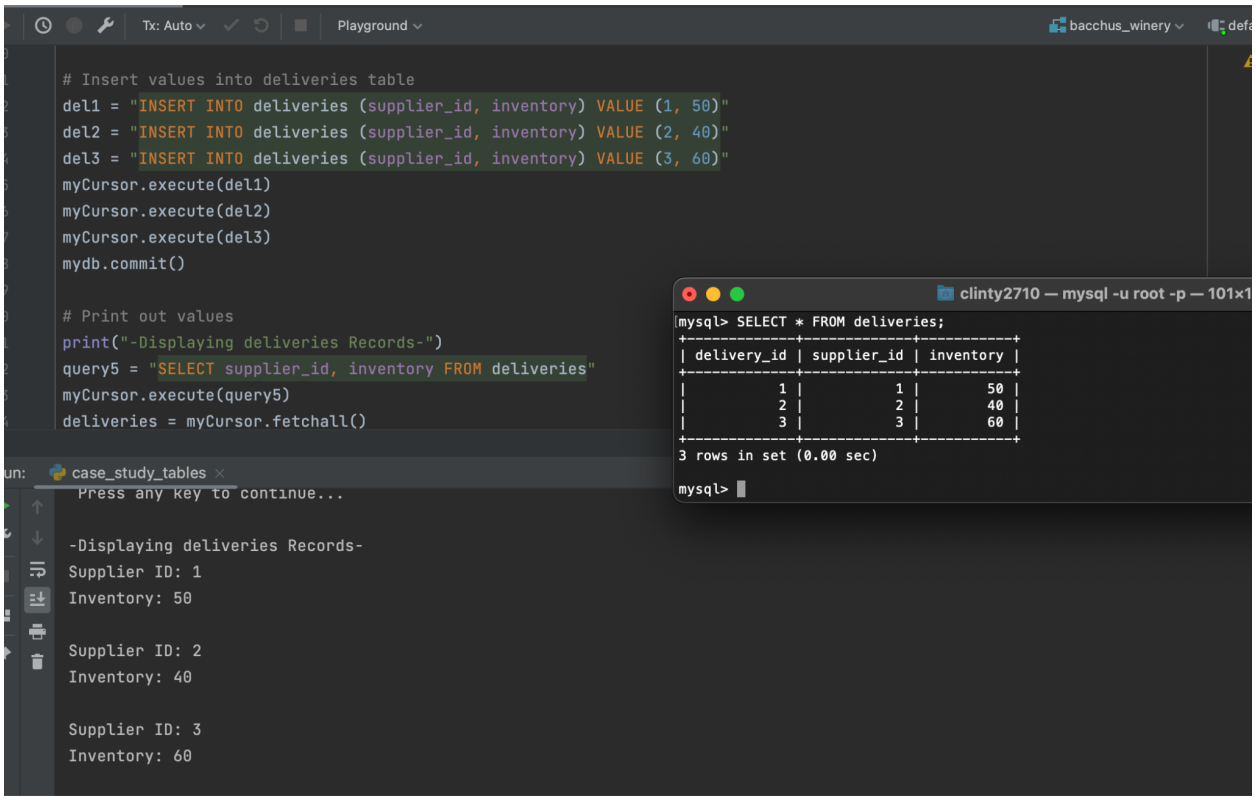
Rachel Cewe, Elizabeth Fung, Scott Jenks, Clint Steadman
12/1/2022
Module 10.1 - Milestone #2



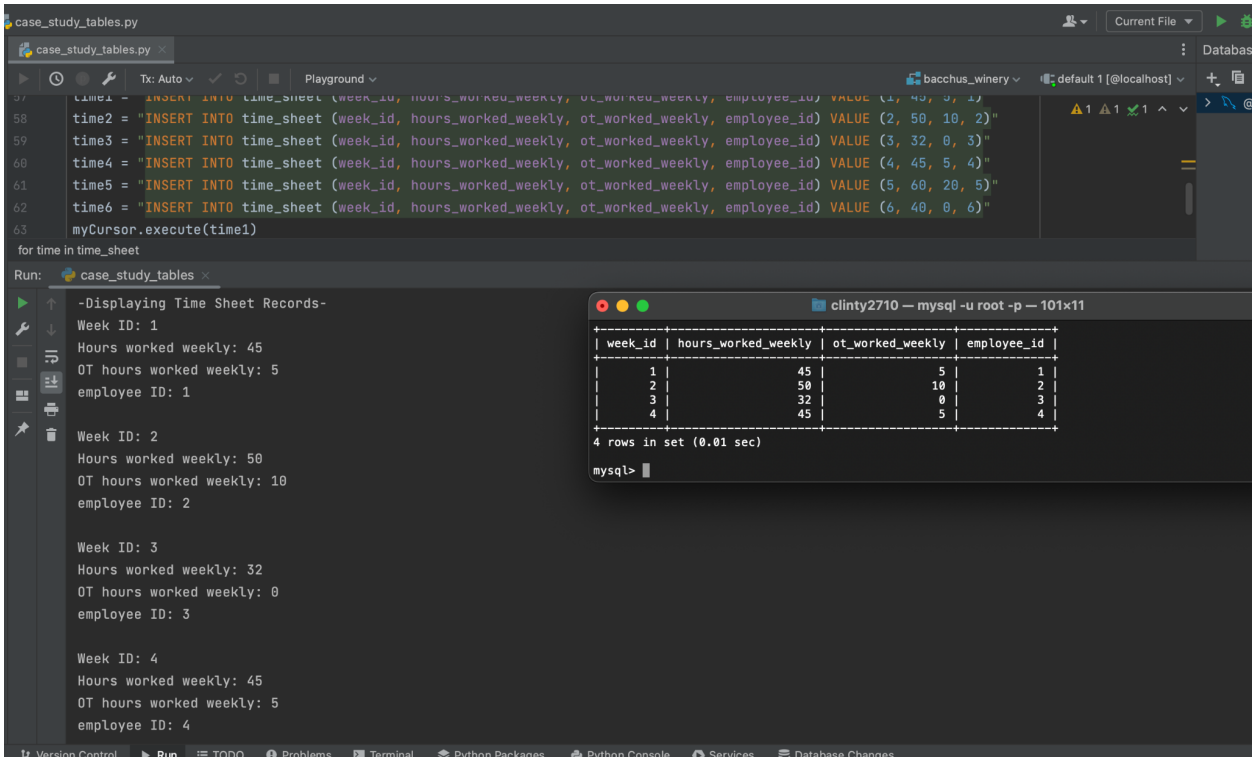


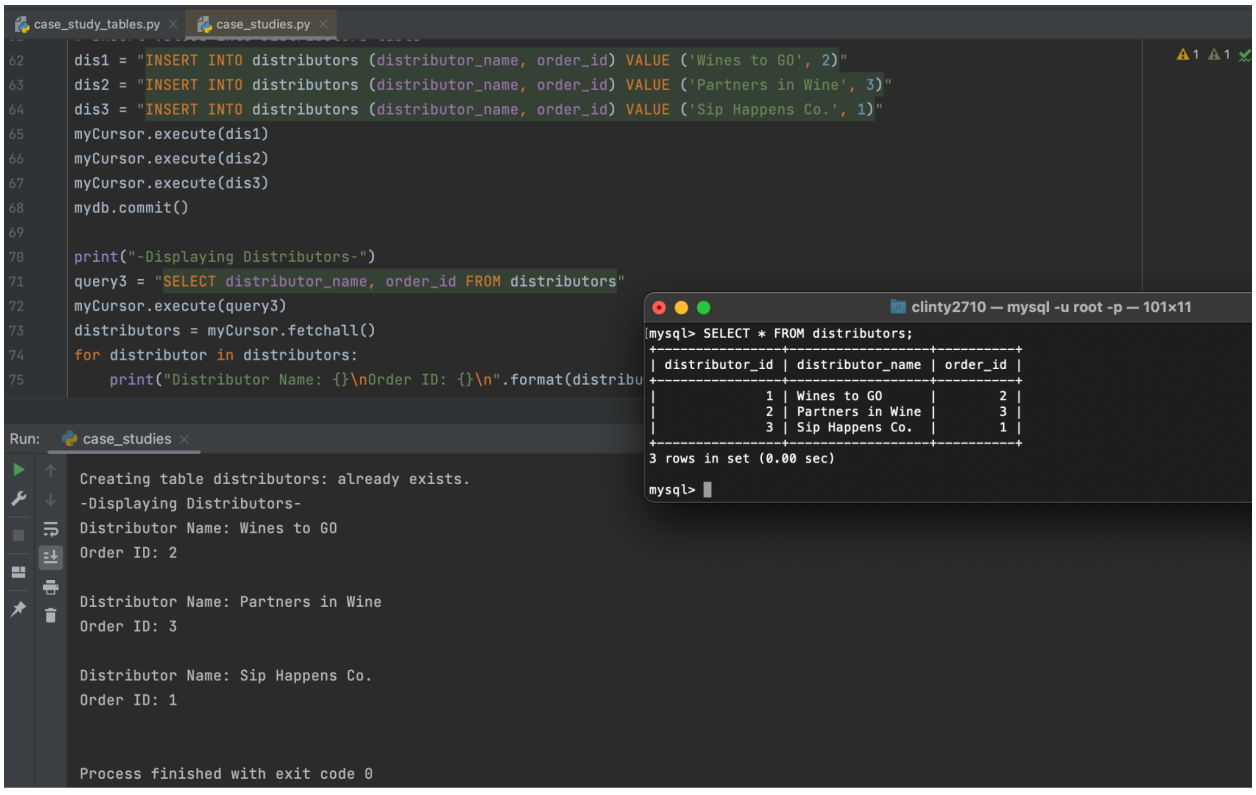
Rachel Cewe, Elizabeth Fung, Scott Jenks, Clint Steadman
12/1/2022
Module 10.1 - Milestone #2





Rachel Cewe, Elizabeth Fung, Scott Jenks, Clint Steadman
12/1/2022
Module 10.1 - Milestone #2





Rachel Cewe, Elizabeth Fung, Scott Jenks, Clint Steadman
12/1/2022
Module 10.1 - Milestone #2