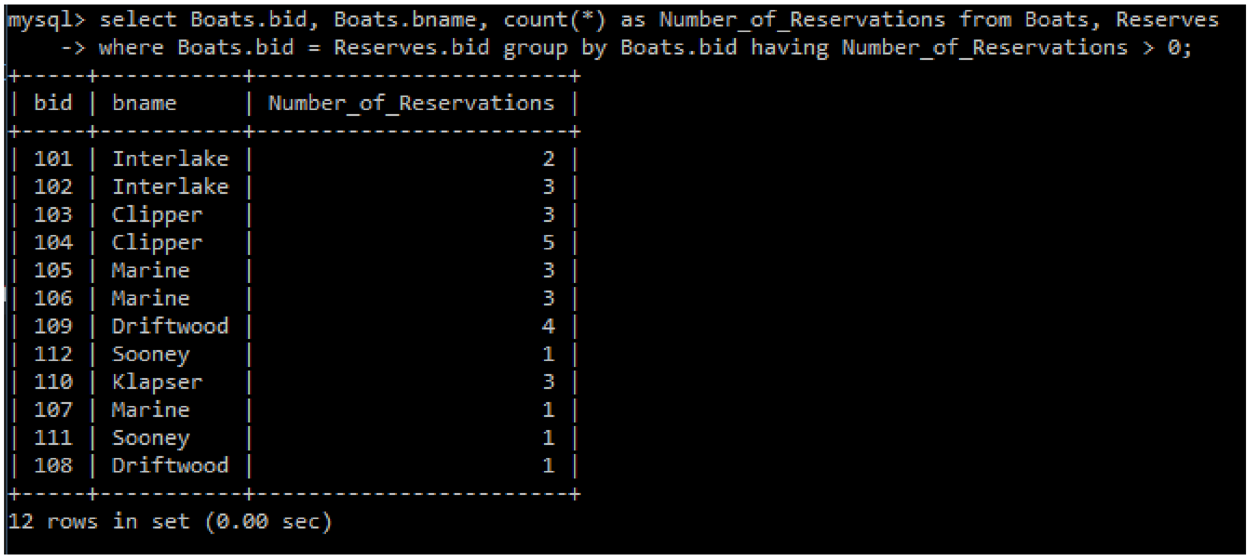
Danny Hong ECE-464 Databases Problem Set 1

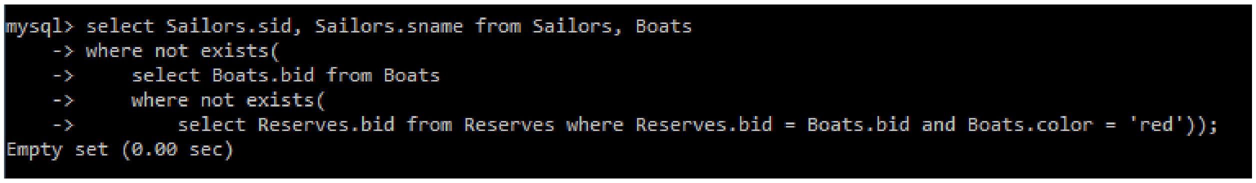
PART 1

1.) List, for every boat, the number of times it has been reserved, excluding those boats that have never been reserved (list the id and the name).



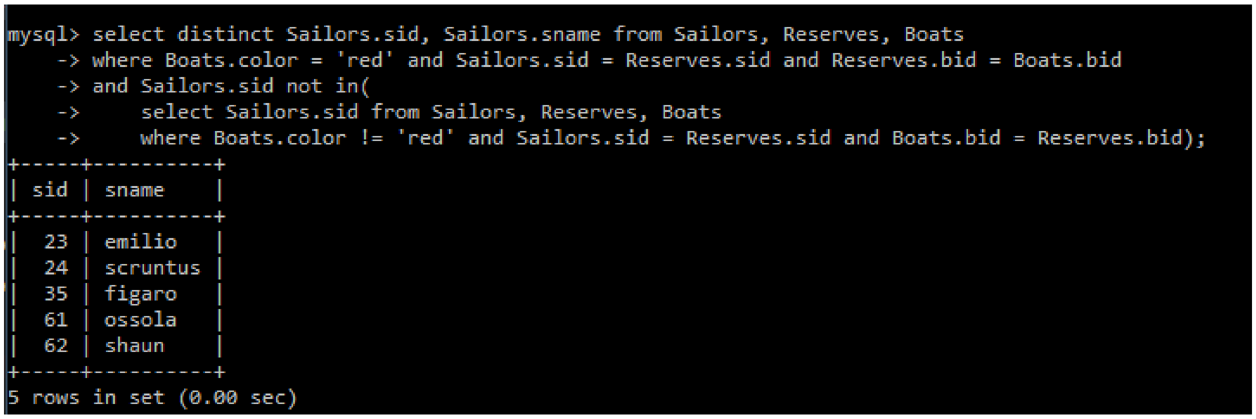
```
mysql> select Boats.bid, Boats.bname, count(*) as Number_of_Reservations from Boats, Reserves
    -> where Boats.bid = Reserves.bid group by Boats.bid having Number_of_Reservations > 0;
+-----+-----------+------------------------+
| bid | bname     | Number_of_Reservations |
+-----+-----------+------------------------+
| 101 | Interlake |                      2 |
| 102 | Interlake |                      3 |
| 103 | Clipper   |                      3 |
| 104 | Clipper   |                      5 |
| 105 | Marine    |                      3 |
| 106 | Marine    |                      3 |
| 109 | Driftwood |                      4 |
| 112 | Sooney    |                      1 |
| 110 | Klapser   |                      3 |
| 107 | Marine    |                      1 |
| 111 | Sooney    |                      1 |
| 108 | Driftwood |                      1 |
+-----+-----------+------------------------+
12 rows in set (0.00 sec)
```

2.) List those sailors who have reserved every red boat (list the id and the name).



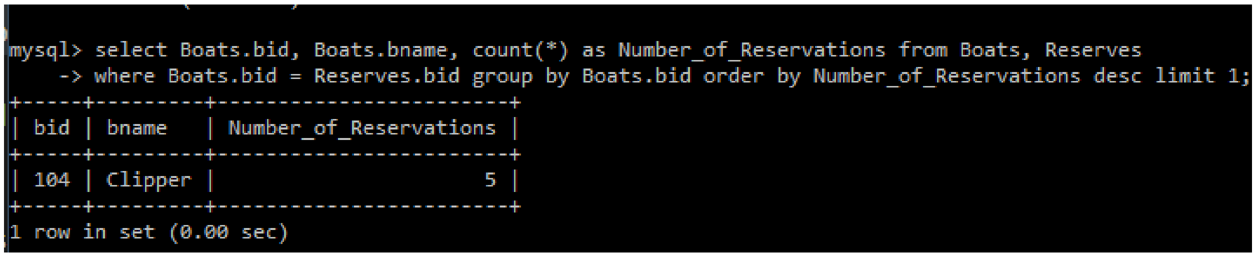
```
mysql> select Sailors.sid, Sailors.sname from Sailors, Boats
    -> where not exists(
    ->     select Boats.bid from Boats
    ->     where not exists(
    ->         select Reserves.bid from Reserves where Reserves.bid = Boats.bid and Boats.color = 'red'));
Empty set (0.00 sec)
```

3.) List those sailors who have reserved only red boats.

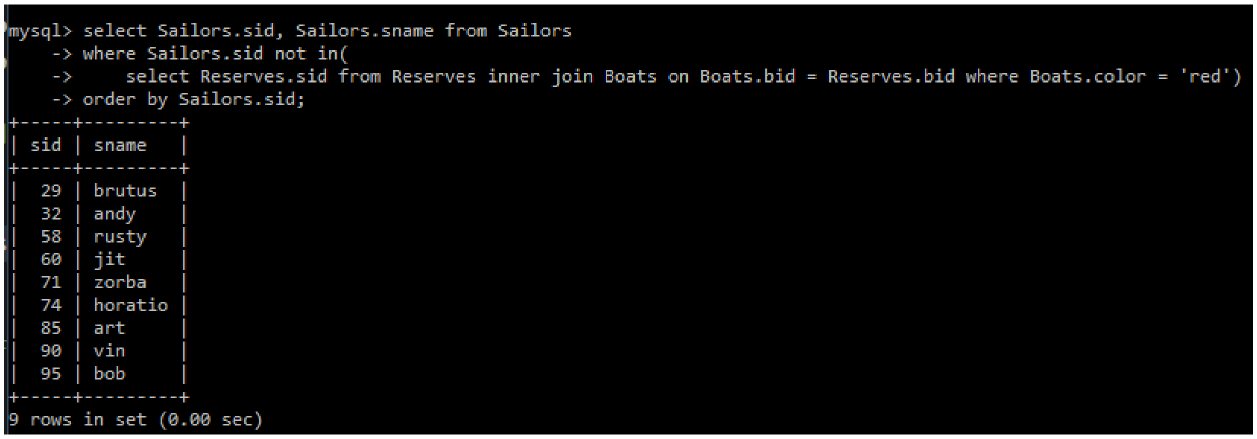


```
mysql> select distinct Sailors.sid, Sailors.sname from Sailors, Reserves, Boats
    -> where Boats.color = 'red' and Sailors.sid = Reserves.sid and Reserves.bid = Boats.bid
    -> and Sailors.sid not in(
    ->     select Sailors.sid from Sailors, Reserves, Boats
    ->     where Boats.color != 'red' and Sailors.sid = Reserves.sid and Boats.bid = Reserves.bid);
+-----+----------+
| sid | sname    |
+-----+----------+
|  23 | emilio   |
|  24 | scruntus |
|  35 | figaro   |
|  61 | ossola   |
|  62 | shaun    |
+-----+----------+
5 rows in set (0.00 sec)
```

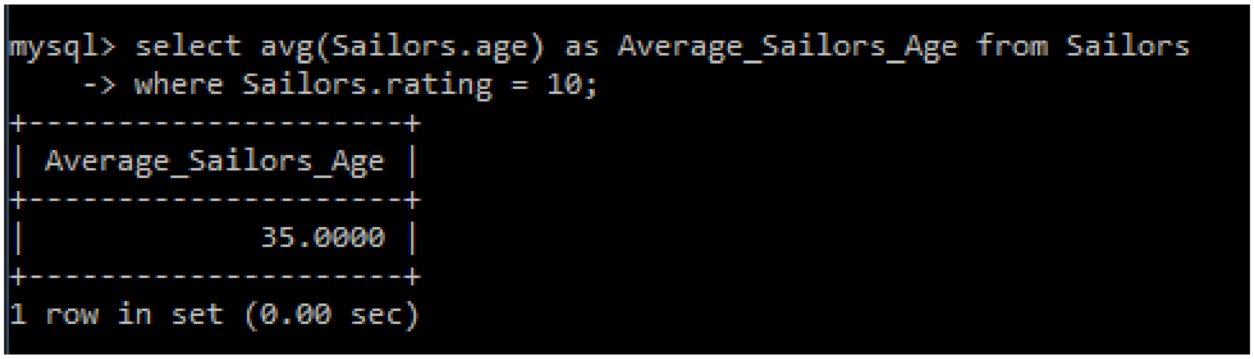
4.) For which boat are there the most reservations?


```
mysql> select Boats.bid, Boats.bname, count(*) as Number_of_Reservations from Boats, Reserves
    -> where Boats.bid = Reserves.bid group by Boats.bid order by Number_of_Reservations desc limit 1;
+-----+---------+------------------------+
| bid | bname   | Number_of_Reservations |
+-----+---------+------------------------+
| 104 | Clipper |                      5 |
+-----+---------+------------------------+
1 row in set (0.00 sec)
```

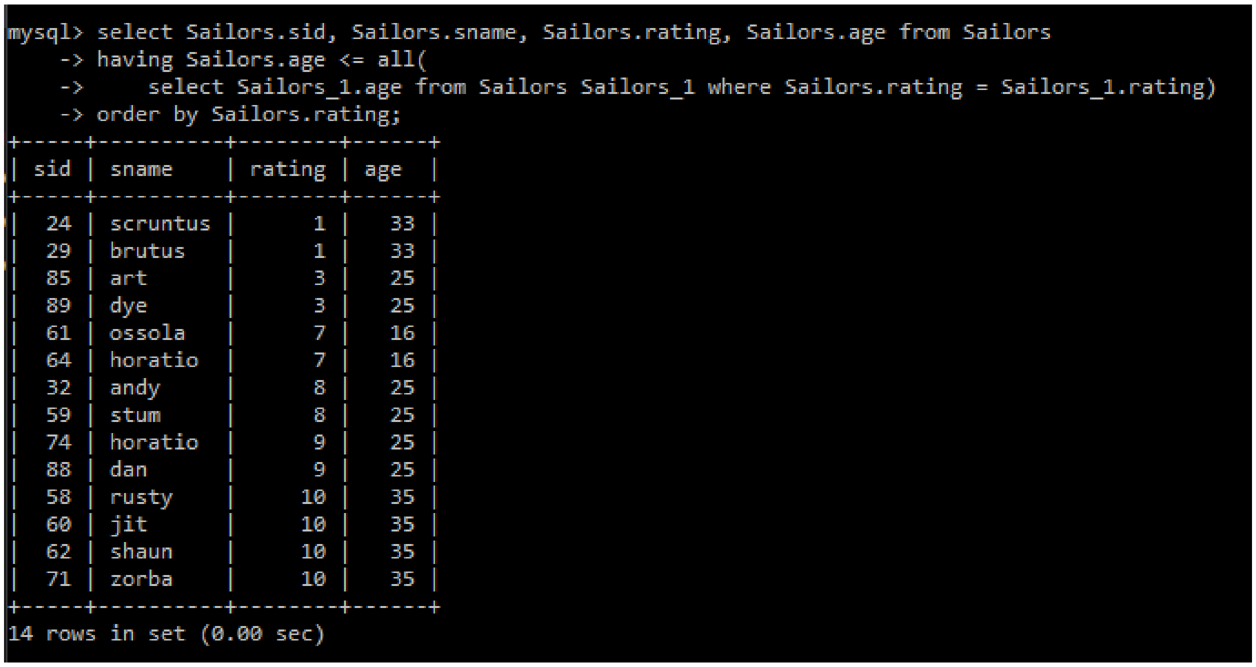
5.) Select all sailors who have never reserved a red boat.


```
mysql> select Sailors.sid, Sailors.sname from Sailors
    -> where Sailors.sid not in(
    ->     select Reserves.sid from Reserves inner join Boats on Boats.bid = Reserves.bid where Boats.color = 'red')
    -> order by Sailors.sid;
+-----+---------+
| sid | sname   |
+-----+---------+
|  29 | brutus  |
|  32 | andy    |
|  58 | rusty   |
|  60 | jit     |
|  71 | zorba   |
|  74 | horatio |
|  85 | art     |
|  90 | vin     |
|  95 | bob     |
+-----+---------+
9 rows in set (0.00 sec)
```

6.) Find the average age of sailors with a rating of 10.

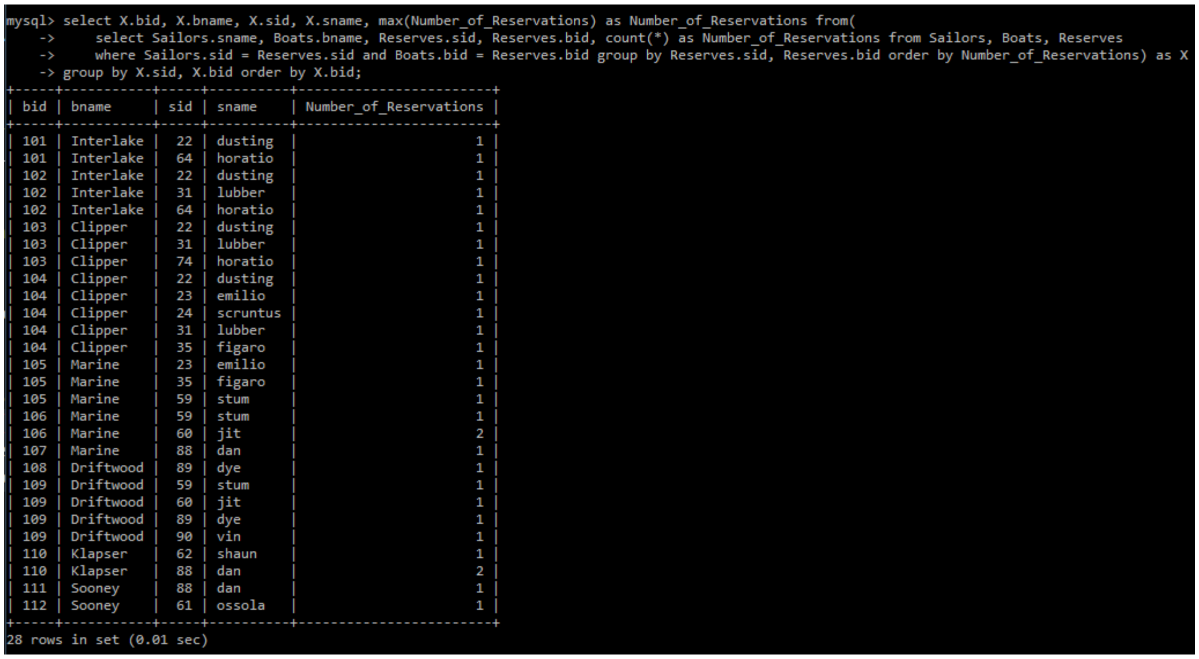

```
mysql> select avg(Sailors.age) as Average_Sailors_Age from Sailors
    -> where Sailors.rating = 10;
+---------------------+
| Average_Sailors_Age |
+---------------------+
|             35.0000 |
+---------------------+
1 row in set (0.00 sec)
```

7.) For each rating, find the name and id of the youngest sailor.



```
mysql> select Sailors.sid, Sailors.sname, Sailors.rating, Sailors.age from Sailors
    -> having Sailors.age <= all(
    ->     select Sailors_1.age from Sailors Sailors_1 where Sailors.rating = Sailors_1.rating)
    -> order by Sailors.rating;
+-----+---------+--------+------+
| sid | sname   | rating | age  |
+-----+---------+--------+------+
|  24 | scruntus |      1 |   33 |
|  29 | brutus  |      1 |   33 |
|  85 | art     |      3 |   25 |
|  89 | dye     |      3 |   25 |
|  61 | ossola  |      7 |   16 |
|  64 | horatio |      7 |   16 |
|  32 | andy    |      8 |   25 |
|  59 | stum    |      8 |   25 |
|  74 | horatio |      9 |   25 |
|  88 | dan     |      9 |   25 |
|  58 | rusty   |     10 |   35 |
|  60 | jit     |     10 |   35 |
|  62 | shaun   |     10 |   35 |
|  71 | zorba   |     10 |   35 |
+-----+---------+--------+------+
14 rows in set (0.00 sec)
```

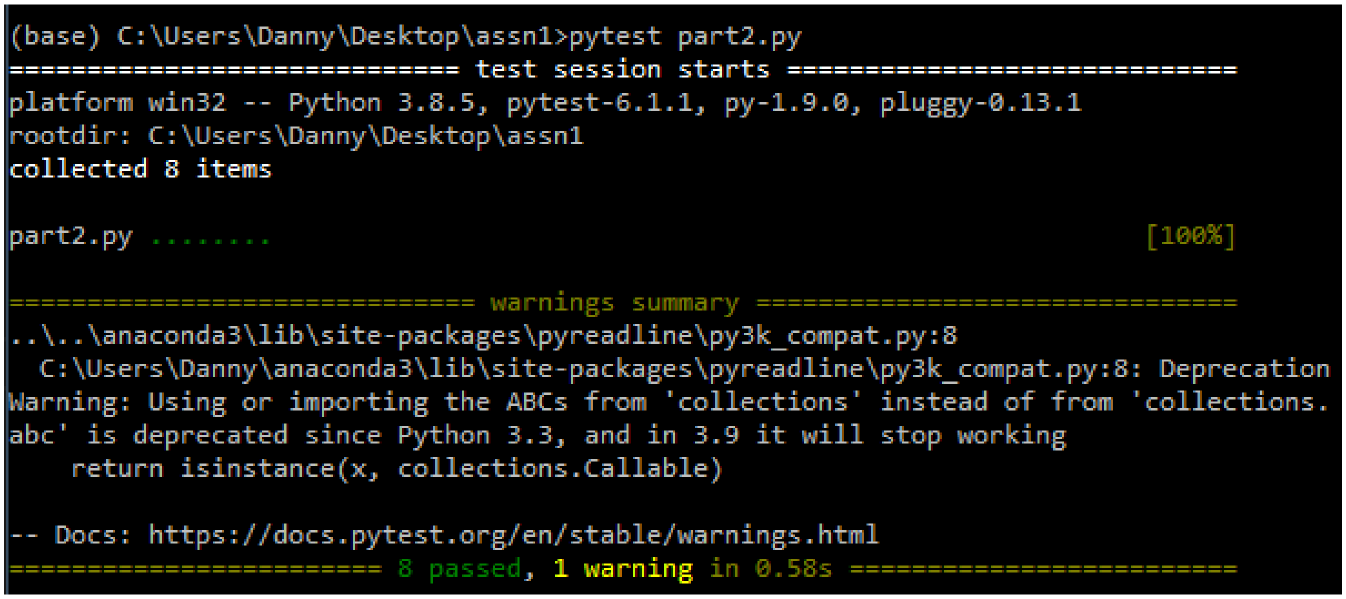
8.) Select, for each boat, the sailor who made the highest number of reservations for that boat.



```
mysql> select X.bid, X.bname, X.sid, X.sname, max(Number_of_Reservations) as Number_of_Reservations from(
    ->     select Sailors.sname, Boats.bname, Reserves.sid, Reserves.bid, count(*) as Number_of_Reservations from Sailors, Boats, Reserves
    ->     where Sailors.sid = Reserves.sid and Boats.bid = Reserves.bid group by Reserves.sid, Reserves.bid order by Number_of_Reservations) as X
    -> group by X.sid, X.bid order by X.bid;
+-----+-----------+-----+----------+------------------------+
| bid | bname     | sid | sname    | Number_of_Reservations |
+-----+-----------+-----+----------+------------------------+
| 101 | Interlake |  22 | dusting  |                      1 |
| 101 | Interlake |  64 | horatio  |                      1 |
| 102 | Interlake |  22 | dusting  |                      1 |
| 102 | Interlake |  31 | lubber   |                      1 |
| 102 | Interlake |  64 | horatio  |                      1 |
| 103 | Clipper   |  22 | dusting  |                      1 |
| 103 | Clipper   |  31 | lubber   |                      1 |
| 103 | Clipper   |  74 | horatio  |                      1 |
| 104 | Clipper   |  22 | dusting  |                      1 |
| 104 | Clipper   |  23 | emilio   |                      1 |
| 104 | Clipper   |  24 | scruntus |                      1 |
| 104 | Clipper   |  31 | lubber   |                      1 |
| 104 | Clipper   |  35 | figaro   |                      1 |
| 105 | Marine    |  23 | emilio   |                      1 |
| 105 | Marine    |  35 | figaro   |                      1 |
| 105 | Marine    |  59 | stum     |                      1 |
| 106 | Marine    |  59 | stum     |                      1 |
| 106 | Marine    |  60 | jit      |                      2 |
| 107 | Marine    |  88 | dan      |                      1 |
| 108 | Driftwood |  89 | dye      |                      1 |
| 109 | Driftwood |  59 | stum     |                      1 |
| 109 | Driftwood |  60 | jit      |                      1 |
| 109 | Driftwood |  89 | dye      |                      1 |
| 109 | Driftwood |  90 | vin      |                      1 |
| 110 | Klapser   |  62 | shaun    |                      1 |
| 110 | Klapser   |  88 | dan      |                      2 |
| 111 | Sooney    |  88 | dan      |                      1 |
| 112 | Sooney    |  61 | ossola   |                      1 |
+-----+-----------+-----+----------+------------------------+
28 rows in set (0.01 sec)
```

PART 2

See the attached python file part2.py for the ORM code. ORM queries were written and tested against all 8 SQL queries from Part 1. Below is the output after using pytest.

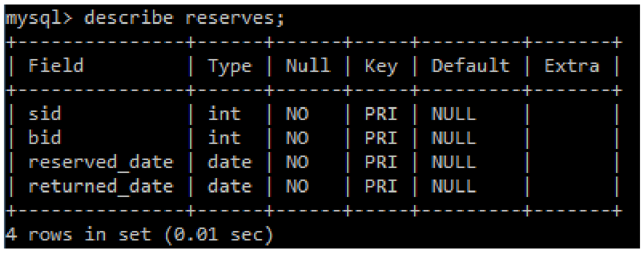


```
(base) C:\Users\Danny\Desktop\assn1>pytest part2.py
============================= test session starts =============================
platform win32 -- Python 3.8.5, pytest-6.1.1, py-1.9.0, pluggy-0.13.1
rootdir: C:\Users\Danny\Desktop\assn1
collected 8 items

part2.py ........                                                        [100%]

============================== warnings summary ===============================
..\..\anaconda3\lib\site-packages\pyreadline\py3k_compat.py:8
  C:\Users\Danny\anaconda3\lib\site-packages\pyreadline\py3k_compat.py:8: Deprecation
Warning: Using or importing the ABCs from 'collections' instead of from 'collections.
abc' is deprecated since Python 3.3, and in 3.9 it will stop working
    return isinstance(x, collections.Callable)

-- Docs: https://docs.pytest.org/en/stable/warnings.html
======================== 8 passed, 1 warning in 0.58s =========================
```

PART 3

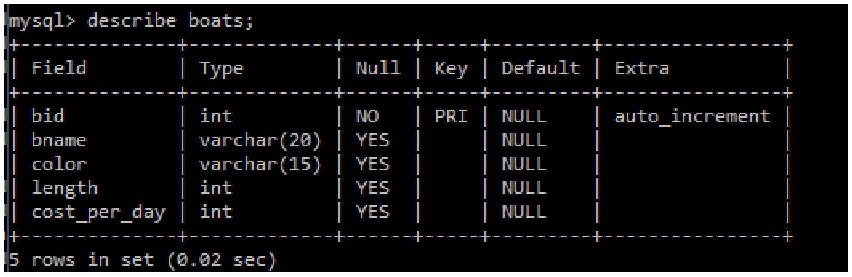
In order to expand the codebase from part 2, the following changes were made:

1.) The ‘reserves’ table was edited so that it contained ‘reserved_date’ and ‘returned_date’ attributes, which would be later used to determine the number of days that a particular boat had been leased out and used for. Shown below is its schema.



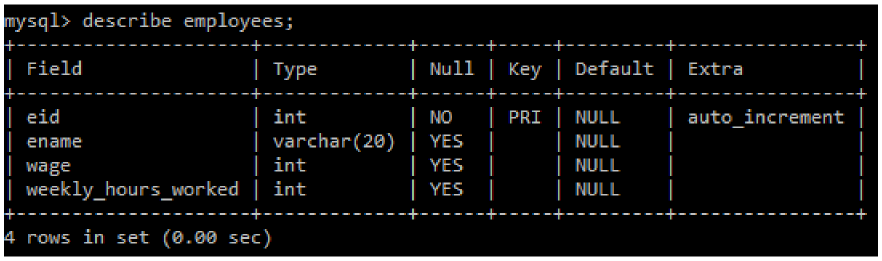
```
mysql> describe reserves;
+---------------+------+------+-----+---------+-------+
| Field         | Type | Null | Key | Default | Extra |
+---------------+------+------+-----+---------+-------+
| sid           | int  | NO   | PRI | NULL    |       |
| bid           | int  | NO   | PRI | NULL    |       |
| reserved_date | date | NO   | PRI | NULL    |       |
| returned_date | date | NO   | PRI | NULL    |       |
+---------------+------+------+-----+---------+-------+
4 rows in set (0.01 sec)
```

2.) The ‘boats’ table was edited so that it contained a ‘cost_per_day’ attribute, which would represent how much per day it would be to rent a particular boat. Shown below is its schema.



```
mysql> describe boats;
+--------------+-------------+------+-----+---------+----------------+
| Field        | Type        | Null | Key | Default | Extra          |
+--------------+-------------+------+-----+---------+----------------+
| bid          | int         | NO   | PRI | NULL    | auto_increment |
| bname        | varchar(20) | YES  |     | NULL    |                |
| color        | varchar(15) | YES  |     | NULL    |                |
| length       | int         | YES  |     | NULL    |                |
| cost_per_day | int         | YES  |     | NULL    |                |
+--------------+-------------+------+-----+---------+----------------+
5 rows in set (0.02 sec)
```

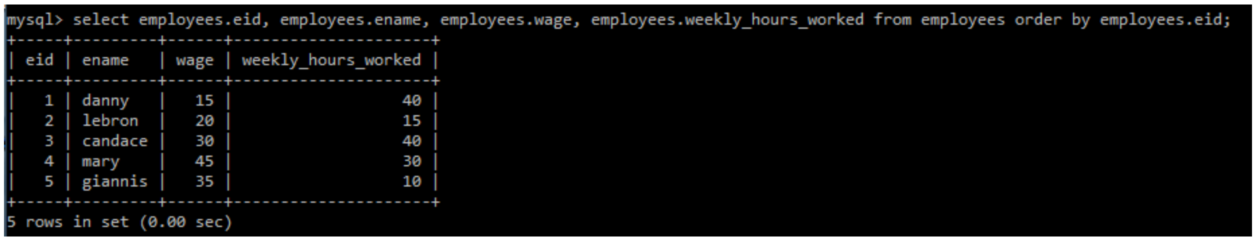
3.) Finally, an ‘employee’ table was added, consisting of an employee’s id, name, wage, and weekly hours worked. It is assumed that the number of hours that each employee works per week is a fixed and unchanged amount. Shown below is its schema.



```
mysql> describe employees;
+---------------------+-------------+------+-----+---------+----------------+
| Field               | Type        | Null | Key | Default | Extra          |
+---------------------+-------------+------+-----+---------+----------------+
| eid                 | int         | NO   | PRI | NULL    | auto_increment |
| ename               | varchar(20) | YES  |     | NULL    |                |
| wage                | int         | YES  |     | NULL    |                |
| weekly_hours_worked | int         | YES  |     | NULL    |                |
+---------------------+-------------+------+-----+---------+----------------+
4 rows in set (0.00 sec)
```

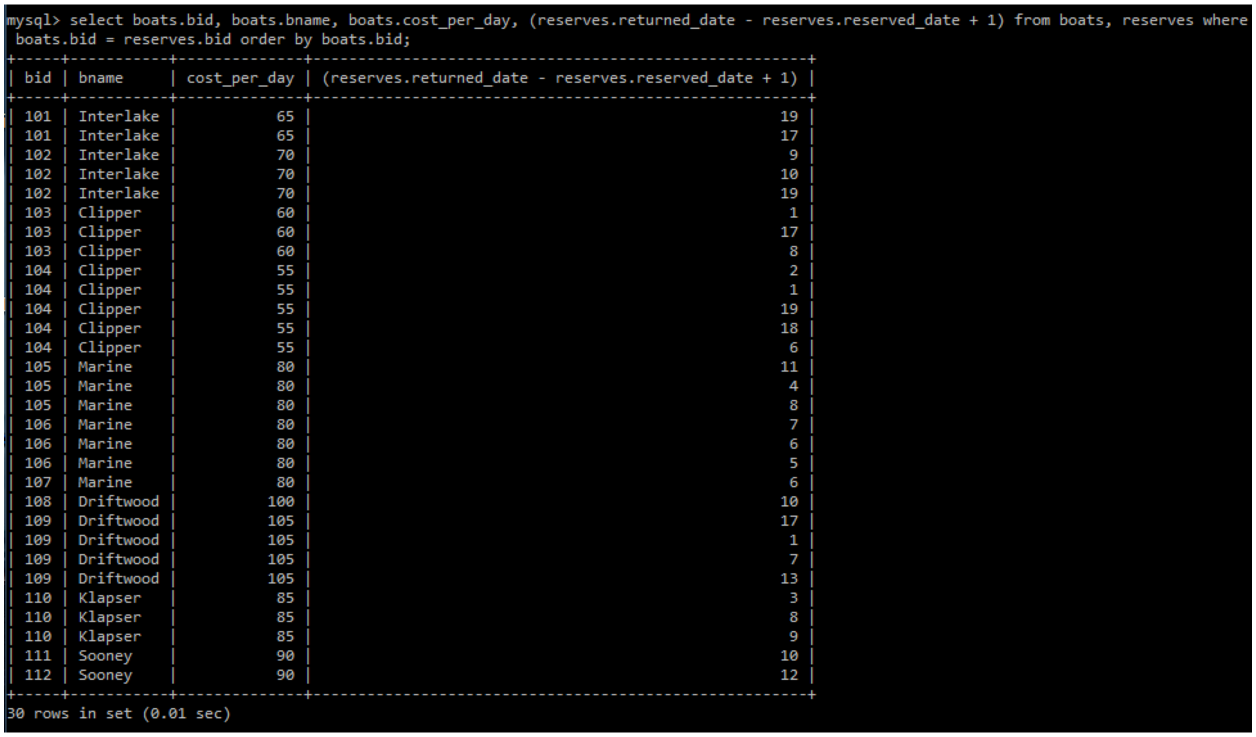
The originally provided sailors.sql file was edited as well to account for these changes. A reserved date and returned date were added to each reservation in ‘reserves’, and for each boat in ‘boats’, a cost value was assigned to it. Five employees were created as well.

The information from the new codebase will allow the company to be able to calculate the total amount that it will pay its employees per week. Using the table shown below, each employee’s weekly earnings is determined by multiplying his/her weekly hours worked by his/her hourly wage. The values are then summed together to obtain the total amount. These calculations were also done externally, and so pytest was later used to assert if the values matched.



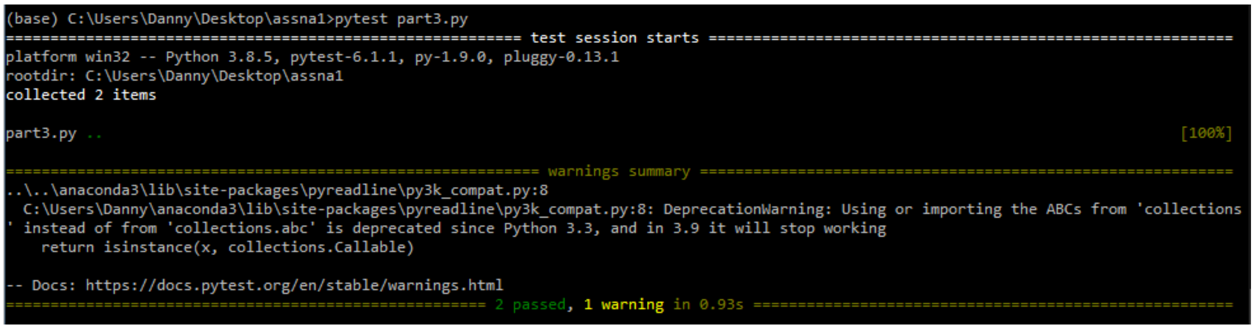
```
mysql> select employees.eid, employees.ename, employees.wage, employees.weekly_hours_worked from employees order by employees.eid;
+-----+---------+------+---------------------+
| eid | ename   | wage | weekly_hours_worked |
+-----+---------+------+---------------------+
|   1 | danny   |   15 |                  40 |
|   2 | lebron  |   20 |                  15 |
|   3 | candace |   30 |                  40 |
|   4 | mary    |   45 |                  30 |
|   5 | giannis |   35 |                  10 |
+-----+---------+------+---------------------+
5 rows in set (0.00 sec)
```

The information will also allow the company to be able to calculate the total earnings from their boat rental service. Using the table shown below, the number of days each boat was rented and used for is calculated by subtracting the reserved date from the returned date and then adding that result to 1. The addition of 1 is important in this instance since when counting the number of days through subtraction, the first day is always unaccounted for, so adding a 1 to the result would fix this. Then, this number is multiplied by the boat's respective cost per day for usage. After this was done for each reserved boat, the resulting values are summed together to obtain the total boat earnings. Similarly, these calculations were also done externally, and so pytest was later used to assert if the values matched.



```
mysql> select boats.bid, boats.bname, boats.cost_per_day, (reserves.returned_date - reserves.reserved_date + 1) from boats, reserves where
 boats.bid = reserves.bid order by boats.bid;
+-----+-----------+--------------+-----------------------------------------------------+
| bid | bname     | cost_per_day | (reserves.returned_date - reserves.reserved_date + 1) |
+-----+-----------+--------------+-----------------------------------------------------+
| 101 | Interlake |           65 |                                                  19 |
| 101 | Interlake |           65 |                                                  17 |
| 102 | Interlake |           70 |                                                   9 |
| 102 | Interlake |           70 |                                                  10 |
| 102 | Interlake |           70 |                                                  19 |
| 103 | Clipper   |           60 |                                                   1 |
| 103 | Clipper   |           60 |                                                  17 |
| 103 | Clipper   |           60 |                                                   8 |
| 104 | Clipper   |           55 |                                                   2 |
| 104 | Clipper   |           55 |                                                   1 |
| 104 | Clipper   |           55 |                                                  19 |
| 104 | Clipper   |           55 |                                                  18 |
| 104 | Clipper   |           55 |                                                   6 |
| 105 | Marine    |           80 |                                                  11 |
| 105 | Marine    |           80 |                                                   4 |
| 105 | Marine    |           80 |                                                   8 |
| 106 | Marine    |           80 |                                                   7 |
| 106 | Marine    |           80 |                                                   6 |
| 106 | Marine    |           80 |                                                   5 |
| 107 | Marine    |           80 |                                                   6 |
| 108 | Driftwood |          100 |                                                  10 |
| 109 | Driftwood |          105 |                                                  17 |
| 109 | Driftwood |          105 |                                                   1 |
| 109 | Driftwood |          105 |                                                   7 |
| 109 | Driftwood |          105 |                                                  13 |
| 110 | Klapser   |           85 |                                                   3 |
| 110 | Klapser   |           85 |                                                   8 |
| 110 | Klapser   |           85 |                                                   9 |
| 111 | Sooney    |           90 |                                                  10 |
| 112 | Sooney    |           90 |                                                  12 |
+-----+-----------+--------------+-----------------------------------------------------+
30 rows in set (0.01 sec)
```

Shown below is the result after using pytest. As it can be seen, both tests successfully passed.



```
(base) C:\Users\Danny\Desktop\assna1>pytest part3.py
========================================================== test session starts ===========================================================
platform win32 -- Python 3.8.5, pytest-6.1.1, py-1.9.0, pluggy-0.13.1
rootdir: C:\Users\Danny\Desktop\assna1
collected 2 items

part3.py ..                                                                                                                        [100%]

============================================================ warnings summary ============================================================
..\..\anaconda3\lib\site-packages\pyreadline\py3k_compat.py:8
  C:\Users\Danny\anaconda3\lib\site-packages\pyreadline\py3k_compat.py:8: DeprecationWarning: Using or importing the ABCs from 'collections
' instead of from 'collections.abc' is deprecated since Python 3.3, and in 3.9 it will stop working
    return isinstance(x, collections.Callable)

-- Docs: https://docs.pytest.org/en/stable/warnings.html
===================================================== 2 passed, 1 warning in 0.93s ======================================================
```