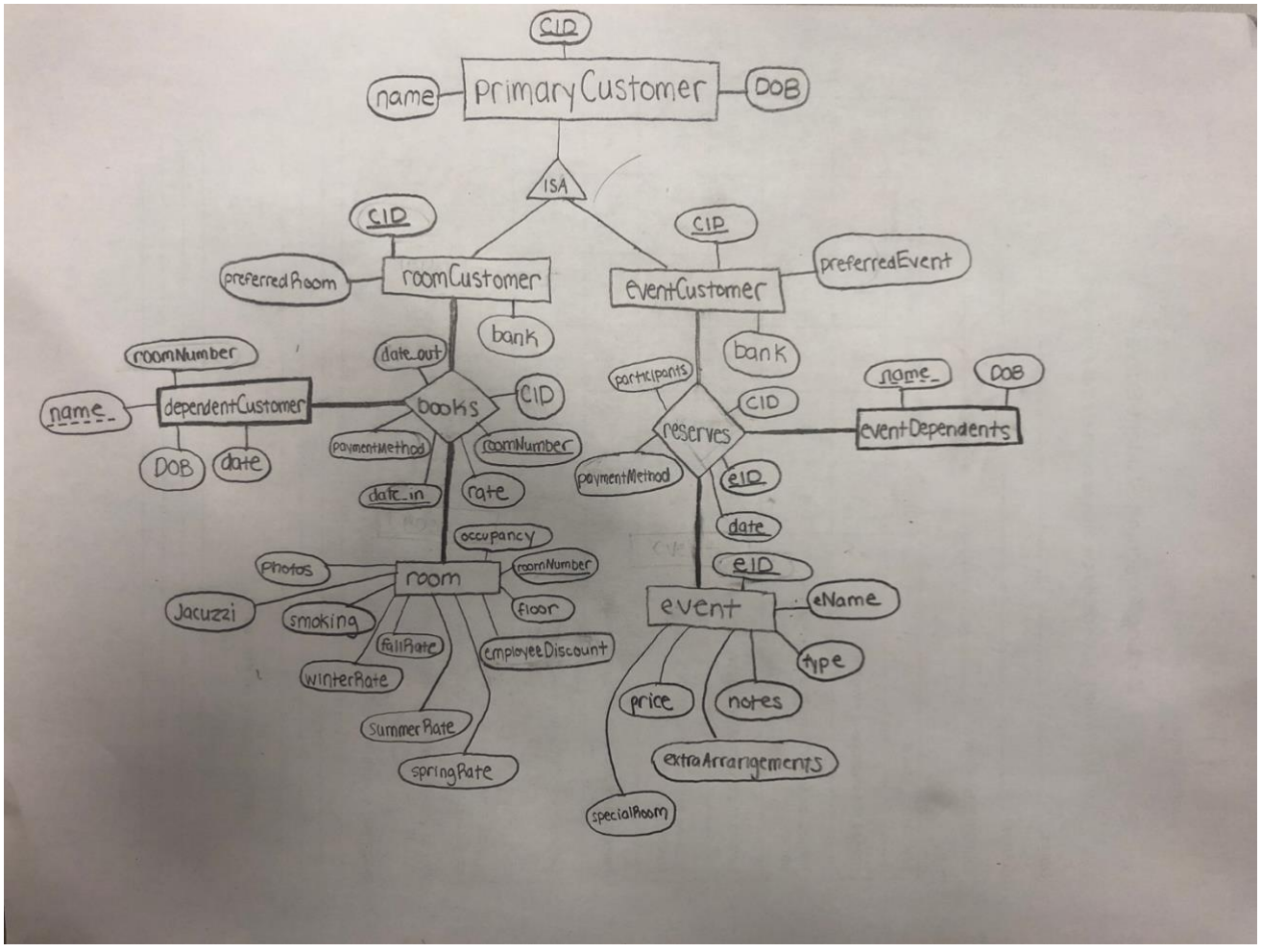
I decided to store the customers in a hierarchical structure. At the top there are all customers responsible for a payment called primaryCustomers. Using the “ISA” connection I made two subtables that are children of primaryCustomers: one called eventCustomer and one called roomCustomer. For all three of these tables the primary key is a field called “cID” which stands for customer ID. For roomCustomer, I established a relationship between roomCustomer, dependentCustomer, and room, as a relationship table called books. The dependentCustomer table is a weak entity that engages in total participation with the relationship table (books). Its key is dependentName. The room table is used to store information about the rooms, and its primary key is the roomNumber. The books table uses information from roomCustomer and room to store information about room bookings. On the other side of the hierarchical structure there is a symmetrical flow. The event table, the eventDependents, and eventCustomer tables all engage with a relationship table called reserves. The event table stores information about events and is identified by eID (event ID). The eventDependents table stores information about the people who attend the event but do not pay. These three tables store information about each reservation between an eventCustomer and an event.



```
CREATE DATABASE hotel;

CREATE TABLE primaryCustomer(
    cID INT NOT NULL,
    name VARCHAR(50) NOT NULL,
    DOB DATE,
    PRIMARY KEY(cID)
    );

INSERT INTO primaryCustomer(cID,name,DOB) VALUES
    (1, "John Abraham", '1987-02-23'),
    (2, "Alex Smith", '1998-03-31'),
    (3, "Mike Tirico", '1987-02-23'),
    (4, "Dana Olive", '1987-02-23'),
    (5, "Rebecca Martinez", '1997-08-07'),
    (6, "Stanley Hudson", '1987-02-23'),
    (7, "Meek Mill", '1987-02-23'),
    (8, "Michael Jordan", '1987-02-23'),
    (9, "Lebron James", '1987-02-23'),
    (10, "Eric Plesko", '1987-02-23'),
    (11, "Helicopter McFlurry", '1987-02-23'),
    (12, "Alligator Miami", '1987-02-23');

CREATE TABLE eventCustomer(
    cID INT NOT NULL,
    preferredEvent VARCHAR(25) NOT NULL,
    bank VARCHAR(25) NOT NULL,
    PRIMARY KEY(cID),
    FOREIGN KEY(cID) REFERENCES primaryCustomer(cID));

INSERT INTO eventCustomer (cID, preferredEvent, bank) VALUES
    (6, "Brunch Banquet", 'TD'),
    (7, "Dinner Banquet", 'Chase'),
    (8, "Bar Mitzvah", 'Mariners'),
    (9, "Quinceanera", 'PNC'),
    (10, "Brunch Banquet", 'Wells Fargo'),
    (11, "Wedding Rehearsal", 'TD'),
```

```
    (12, "Dinner Banquet", 'Chase');

CREATE TABLE roomCustomer(
    cID INT NOT NULL,
    preferredRoom VARCHAR(25) NOT NULL,
    bank VARCHAR(25) NOT NULL,
    PRIMARY KEY(cID),
    FOREIGN KEY(cID) REFERENCES primaryCustomer(cID));

INSERT INTO roomCustomer (cID, preferredRoom, bank) VALUES
    (1, "Single", 'State Street'),
    (2, "Double", 'Bank of America'),
    (3, "Triple", 'JP Morgan'),
    (4, "Quad", 'PNC'),
    (5, "Queen", 'TD'),
    (6, "Single", 'Chase'),
    (7, "King", 'Wells Fargo');

CREATE TABLE books(
    roomNumber INT NOT NULL,
    date_in TIMESTAMP NOT NULL,
    date_out TIMESTAMP,
    cID INT NOT NULL,
    paymentMethod VARCHAR(12) NOT NULL,
    rate DEC(7,2) NOT NULL,
    FOREIGN KEY (roomNumber) REFERENCES room(roomNumber),
    FOREIGN KEY (cID) REFERENCES primaryCustomer(cID),
    PRIMARY KEY (roomNumber,date_in)
);
INSERT INTO books VALUES
    (212,'2019-03-31 4:30:07','2019-04-02 9:50:27',4,"Check",350),
    (387,'2019-03-31 4:00:19','2019-04-03 10:12:27',7,"Cash",350),
    (420,'2019-03-28 4:30:07','2019-04-02 10:50:22',3,"Credit",350),
    (212,'2019-03-22 4:29:11','2019-03-29 8:21:33',2,"Check",350),
    (510,'2019-02-28 4:30:07','2019-03-02 11:26:26',1,"Cash",350),
    (919,'2019-02-14 4:30:07','2019-03-04 9:33:47',7,"Debit",350),
    (660,'2019-02-12 4:30:07','2019-02-17 8:09:23',6,"Check",350),
```

```
    (387,'2019-01-31 4:30:07','2019-02-02 5:11:45',5,"Credit",350),
    (101,'2019-01-18 4:30:07','2019-01-19 6:48:15',2,"Check",350),
    (660,'2019-03-26 4:30:07','2019-03-31 7:13:51',4,"Check",350),
    (919,'2019-04-12 4:30:07','2019-04-20 6:56:34',3,"Cash",350),
    (420,'2019-04-11 4:30:07','2019-04-15 7:12:44',1,"Cash",350),
    (212,'2019-03-04 4:30:07','2019-03-12 9:41:51',3,"Credit",350),
    (101,'2019-03-12 4:30:07','2019-03-17 5:34:11',4,"Debit",350),
    (510,'2019-03-14 4:30:07','2019-03-25 6:31:43',5,"Check",350);




CREATE TABLE room(
    roomNumber /INT NOT NULL,
    occupancy /INT NOT NULL,
    floor /INT NOT NULL,
    employeeDiscount DEC(2,2) NOT NULL,
    springRate DEC(7,2) NOT NULL,
    fallRate DEC(7,2) NOT NULL,
    summerRate DEC(7,2) NOT NULL,
    winterRate DEC(7,2) NOT NULL,
    smoking BIT NOT NULL,
    jacuzzi BIT NOT NULL,
    photos VARCHAR(80),
    PRIMARY KEY(roomNumber)
);
INSERT INTO room VALUES
(101,2,1,.85,300,250,310,200,0,0,"..."),
(212,2,2,.90,350,300,380,240,0,1,"..."),
(420,4,4,.75,290,250,305,190,1,0,"..."),
(387,4,3,.70,234,200,260,210,1,0,"..."),
(510,2,5,.82,330,300,400,200,0,0,"..."),
(660,2,6,.85,305,260,315,200,0,1,"..."),
(919,2,9,.97,1000,800,1200,750,0,1,"...");

CREATE TABLE dependentCustomer(
```

```
name VARCHAR(45) NOT NULL,
roomNumber INT NOT NULL,
dob DATE NOT NULL,
FOREIGN KEY (roomNumber) REFERENCES room(roomNumber)
);

INSERT INTO dependentCustomer VALUES
    ("Jack Alabama", 212,'1985-04-28'),
    ("Milly TheSavage", 919,'1985-12-31'),
    ("Jack Alabama", 101,'1985-03-28'),
    ("National Sapio", 420,'1985-03-19'),
    ("HotDog Malibu", 387,'1985-03-11'),
    ("Nick Ciambrone", 510,'1985-03-17'),
    ("Paul Licini", 387,'1985-03-09'),
    ("Scott Wills", 212,'1985-08-01'),
    ("Yhago Silva", 101,'1985-03-02'),
    ("Roxburio MatSlammer", 919,'1985-03-05'),
    ("Starvation Runner", 387,'1985-09-25'),
    ("Biceps McGee", 101,'1985-02-12'),
    ("Jill Fun", 420,'1985-06-18'),
    ("Roxanne East", 919,'1985-03-31');



CREATE TABLE event(
    eID INT NOT NULL,
    eName VARCHAR(25) NOT NULL,
    type VARCHAR(10) NOT NULL,
    notes VARCHAR(100) NOT NULL,
    extraArrangements VARCHAR(100) NOT NULL,
    price DEC(8,2) NOT NULL,
    specialRoom BIT NOT NULL,
    PRIMARY KEY(eID)
);

INSERT INTO event VALUES
    (1, "Brunch Banquet", "Feast", "Eggs pancakes and more, big enough for 300 people", "Chefs", 1499.99,0),
    (2, "Dinner Banquet", "Feast", "Steak Lobster and more, big enough for 300 people", "Chefs", 2499.99,0),
```

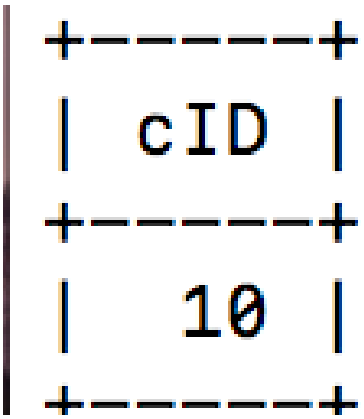
```
    (3, "Quinceanera", "Party", "Spanish celebration for 15th birthday", "Mexican Music, Chefs", 1750.00,1),
    (4, "Wedding Rehearsal", "Party", "Rehearse for the wedding", "Chefs, candles", 2650.00,1),
    (5, "Bar Mitzvah", "Party", "Jewish celebration on becoming a man", "Chefs, Yamikahs, Chair", 1250.00,1),
    (6, "Prom", "Party", "High School Prom, lots of fun", "DJ, poster", 2000.00,0);

CREATE TABLE eventDependents(
    name VARCHAR(40) NOT NULL,
    eID INT NOT NULL,
    DOB DATE NOT NULL,
    PRIMARY KEY(name)
);
INSERT INTO eventDependents VALUES
    ('John Lewinsky', 1,'1988-09-19'),
    ('PJ Tucker', 2,'1988-09-19'),
    ('Rip Panorama', 3,'1988-09-19'),
    ('Pterodactyl Jones', 4,'1988-09-19'),
    ('Lasquarius VonHampton', 5,'1988-09-19'),
    ('Roqualiacous Simple', 6,'1988-09-19'),
    ('Instario Sabadaba', 1,'1988-09-19'),
    ('Mike Allen', 2,'1988-09-19'),
    ('JarHead McBeastly', 6,'1988-09-19'),
    ('Probation Godzilla', 5,'1988-09-19'),
    ('MaryJane Johnson', 3,'1988-09-19'),
    ('Attack Milloy', 4,'1988-09-19'),
    ('Rachel Moore', 1,'1988-09-19');



CREATE TABLE reserves(
    eID INTEGER NOT NULL,
    date TIMESTAMP NOT NULL,
    cID INT NOT NULL,
    participants INT NOT NULL,
    paymentMethod VARCHAR(10),
    PRIMARY KEY(eID, date),
    FOREIGN KEY (cID) REFERENCES primaryCustomer(cID),
    FOREIGN KEY (eID) REFERENCES event(eID)
);
```

```
INSERT INTO reserves VALUES
  (2,'2019-02-01 12:11', 10, 280, "Check"),
  (1,'2019-03-30 12:00', 7, 280, "Check"),
  (2,'2019-03-13 12:00', 8, 280, "Credit"),
  (3,'2019-03-11 12:00', 9, 280, "Check"),
  (3,'2019-02-02 12:00', 10, 280, "Check"),
  (3,'2019-01-19 12:00', 10, 280, "Check"),
  (4,'2019-04-24 12:00', 12, 280, "Credit"),
  (6,'2019-04-25 12:00', 11, 280, "Check"),
  (4,'2019-05-01 12:00', 10, 280, "Check"),
  (5,'2019-04-30 12:00', 9, 280, "Debit"),
  (1,'2019-03-06 12:00', 6, 280, "Check");
```

```
SELECT DISTINCT r.cID
FROM reserves r
GROUP BY r.cID
HAVING count(r.cID)= (SELECT DISTINCT MAX(r.maxRes)
  FROM (SELECT DISTINCT cID, COUNT(cID) AS maxRes
        FROM  reserves
        WHERE date BETWEEN '2019-01-01 00:01:00' AND '2019-12-31 23:59:59'
        GROUP BY cID) r);
```

This Query selects the customerID column from the reserves relation, grouped by cID having count(r.cID) = a nested relation that finds the maximum value from the count(cID) coulumn in that same relation:

The result is a table with one row: the customer ID of the customer who booked the most rooms:



cID
10

All the relations before adding anything via an HTML file linked to a python file:

primaryCustomer:

cID	name	DOB
1	John Abraham	1987-02-23
2	Alex Smith	1998-03-31
3	Mike Tirico	1987-02-23
4	Dana Olive	1987-02-23
5	Rebecca Martinez	1997-08-07
6	Stanley Hudson	1987-02-23
7	Meek Mill	1987-02-23
8	Michael Jordan	1987-02-23
9	Lebron James	1987-02-23
10	Eric Plesko	1987-02-23
11	Helicopter McFlurry	1987-02-23
12	Alligator Miami	1987-02-23

roomCustomer:

cID	preferredRoom	bank
1	Single	State Street
2	Double	Bank of America
3	Triple	JP Morgan
4	Quad	PNC
5	Queen	TD
6	Single	Chase
7	King	Wells Fargo

eventCustomer:

cID	preferredEvent	bank
6	Brunch Banquet	TD
7	Dinner Banquet	Chase
8	Bar Mitzvah	Mariners
9	Quinceanera	PNC
10	Brunch Banquet	Wells Fargo
11	Wedding Rehearsal	TD
12	Dinner Banquet	Chase

rooms:

```
mysql> SELECT * FROM room;
```

roomNumber	occupancy	floor	employeeDiscount	springRate	fallRate	summerRate	winterRate	smoking	jacuzzi	photos
101	2	1	0.85	300.00	250.00	310.00	200.00			...
212	2	2	0.90	350.00	300.00	380.00	240.00			...
387	4	3	0.70	234.00	200.00	260.00	210.00			...
420	4	4	0.75	290.00	250.00	305.00	190.00			...
510	2	5	0.82	330.00	300.00	400.00	200.00			...
660	2	6	0.85	305.00	260.00	315.00	200.00			...
919	2	9	0.97	1000.00	800.00	1200.00	750.00			...

event:

```
mysql> select * from event;
+-----+------------------+-------+-------------------------------------------------+----------------------+---------+-------------+
| eID | eName            | type  | notes                                           | extraArrangements    | price   | specialRoom |
+-----+------------------+-------+-------------------------------------------------+----------------------+---------+-------------+
|   1 | Brunch Banquet   | Feast | Eggs pancakes and more, big enough for 300 people | Chefs                | 1499.99 |             |
|   2 | Dinner Banquet   | Feast | Steak Lobster and more, big enough for 300 people | Chefs                | 2499.99 |             |
|   3 | Quinceanera      | Party | Spanish celebration for 15th birthday           | Mexican Music, Chefs | 1750.00 |             |
|   4 | Wedding Rehearsal | Party | Rehearse for the wedding                        | Chefs, candles       | 2650.00 |             |
|   5 | Bar Mitzvah      | Party | Jewish celebration on becoming a man            | Chefs, Yamikahs, Chair | 1250.00 |             |
|   6 | Prom             | Party | High School Prom, lots of fun                   | DJ, poster           | 2000.00 |             |
+-----+------------------+-------+-------------------------------------------------+----------------------+---------+-------------+
```

books:

```
mysql> select * from books
    -> ;
+------------+---------------------+---------------------+-----+---------------+--------+
| roomNumber | date_in             | date_out            | cID | paymentMethod | rate   |
+------------+---------------------+---------------------+-----+---------------+--------+
|        101 | 2019-01-18 04:30:07 | 2019-01-19 06:48:15 |   2 | Check         | 350.00 |
|        101 | 2019-03-12 04:30:07 | 2019-03-17 05:34:11 |   4 | Debit         | 350.00 |
|        212 | 2019-03-04 04:30:07 | 2019-03-12 09:41:51 |   3 | Credit        | 350.00 |
|        212 | 2019-03-22 04:29:11 | 2019-03-29 08:21:33 |   2 | Check         | 350.00 |
|        212 | 2019-03-31 04:30:07 | 2019-04-02 09:50:27 |   4 | Check         | 350.00 |
|        387 | 2019-01-31 04:30:07 | 2019-02-02 05:11:45 |   5 | Credit        | 350.00 |
|        387 | 2019-03-31 04:00:19 | 2019-04-03 10:12:27 |   7 | Cash          | 350.00 |
|        420 | 2019-03-28 04:30:07 | 2019-04-02 10:50:22 |   3 | Credit        | 350.00 |
|        420 | 2019-04-11 04:30:07 | 2019-04-15 07:12:44 |   1 | Cash          | 350.00 |
|        510 | 2019-02-28 04:30:07 | 2019-03-02 11:26:26 |   1 | Cash          | 350.00 |
|        510 | 2019-03-14 04:30:07 | 2019-03-25 06:31:43 |   5 | Check         | 350.00 |
|        660 | 2019-02-12 04:30:07 | 2019-02-17 08:09:23 |   6 | Check         | 350.00 |
|        660 | 2019-03-26 04:30:07 | 2019-03-31 07:13:51 |   4 | Check         | 350.00 |
|        919 | 2019-02-14 04:30:07 | 2019-03-04 09:33:47 |   7 | Debit         | 350.00 |
|        919 | 2019-04-12 04:30:07 | 2019-04-20 06:56:34 |   3 | Cash          | 350.00 |
+------------+---------------------+---------------------+-----+---------------+--------+
15 rows in set (0.00 sec)
```

reserves:

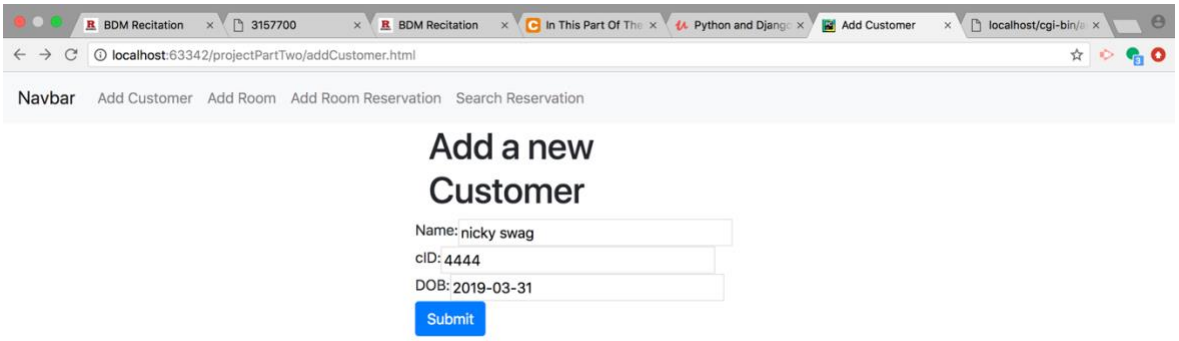
```
mysql> select * from reserves;
+-----+---------------------+-----+--------------+---------------+
| eID | date                | cID | participants | paymentMethod |
+-----+---------------------+-----+--------------+---------------+
|   1 | 2019-03-06 12:00:00 |   6 |          280 | Check         |
|   1 | 2019-03-30 12:00:00 |   7 |          280 | Check         |
|   2 | 2019-02-01 12:11:00 |  10 |          280 | Check         |
|   2 | 2019-03-13 12:00:00 |   8 |          280 | Credit        |
|   3 | 2019-01-19 12:00:00 |  10 |          280 | Check         |
|   3 | 2019-02-02 12:00:00 |  10 |          280 | Check         |
|   3 | 2019-03-11 12:00:00 |   9 |          280 | Check         |
|   4 | 2019-04-24 12:00:00 |  12 |          280 | Credit        |
|   4 | 2019-05-01 12:00:00 |  10 |          280 | Check         |
|   5 | 2019-04-30 12:00:00 |   9 |          280 | Debit         |
|   6 | 2019-04-25 12:00:00 |  11 |          280 | Check         |
+-----+---------------------+-----+--------------+---------------+
11 rows in set (0.00 sec)
```

eventDependents:


```
mysql> select * from reserves;
+-----+---------------------+-----+--------------+---------------+
| eID | date                | cID | participants | paymentMethod |
+-----+---------------------+-----+--------------+---------------+
|   1 | 2019-03-06 12:00:00 |   6 |          280 | Check         |
|   1 | 2019-03-30 12:00:00 |   7 |          280 | Check         |
|   2 | 2019-02-01 12:11:00 |  10 |          280 | Check         |
|   2 | 2019-03-13 12:00:00 |   8 |          280 | Credit        |
|   3 | 2019-01-19 12:00:00 |  10 |          280 | Check         |
|   3 | 2019-02-02 12:00:00 |  10 |          280 | Check         |
|   3 | 2019-03-11 12:00:00 |   9 |          280 | Check         |
|   4 | 2019-04-24 12:00:00 |  12 |          280 | Credit        |
|   4 | 2019-05-01 12:00:00 |  10 |          280 | Check         |
|   5 | 2019-04-30 12:00:00 |   9 |          280 | Debit         |
|   6 | 2019-04-25 12:00:00 |  11 |          280 | Check         |
+-----+---------------------+-----+--------------+---------------+
11 rows in set (0.00 sec)
```

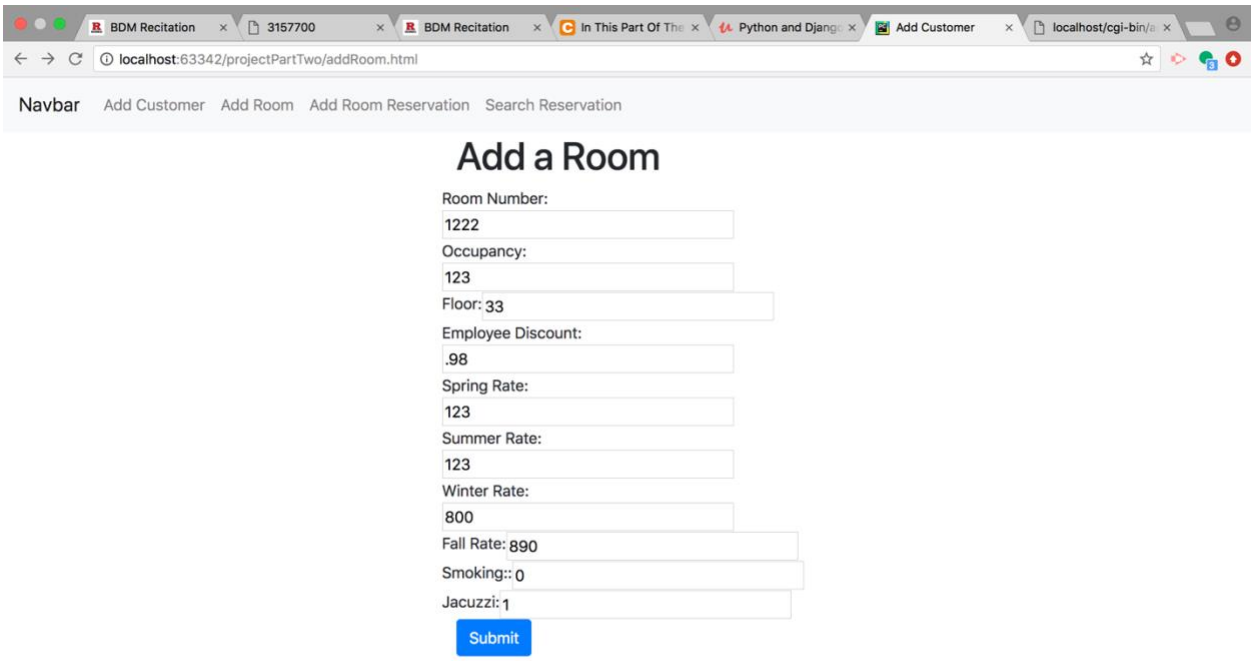

roomDependents:

```
mysql> select * from dependentCustomer;
+--------------------+------------+------------+
| name               | roomNumber | dob        |
+--------------------+------------+------------+
| Jack Alabama       |        212 | 1985-04-28 |
| Milly TheSavage    |        919 | 1985-12-31 |
| Jack Alabama       |        101 | 1985-03-28 |
| National Sapio     |        420 | 1985-03-19 |
| HotDog Malibu      |        387 | 1985-03-11 |
| Nick Ciambrone     |        510 | 1985-03-17 |
| Paul Licini        |        387 | 1985-03-09 |
| Scott Wills        |        212 | 1985-08-01 |
| Yhago Silva        |        101 | 1985-03-02 |
| Roxburio MatSlammer|        919 | 1985-03-05 |
| Starvation Runner  |        387 | 1985-09-25 |
| Biceps McGee       |        101 | 1985-02-12 |
| Jill Fun           |        420 | 1985-06-18 |
| Roxanne East       |        919 | 1985-03-31 |
+--------------------+------------+------------+
14 rows in set (0.00 sec)
```



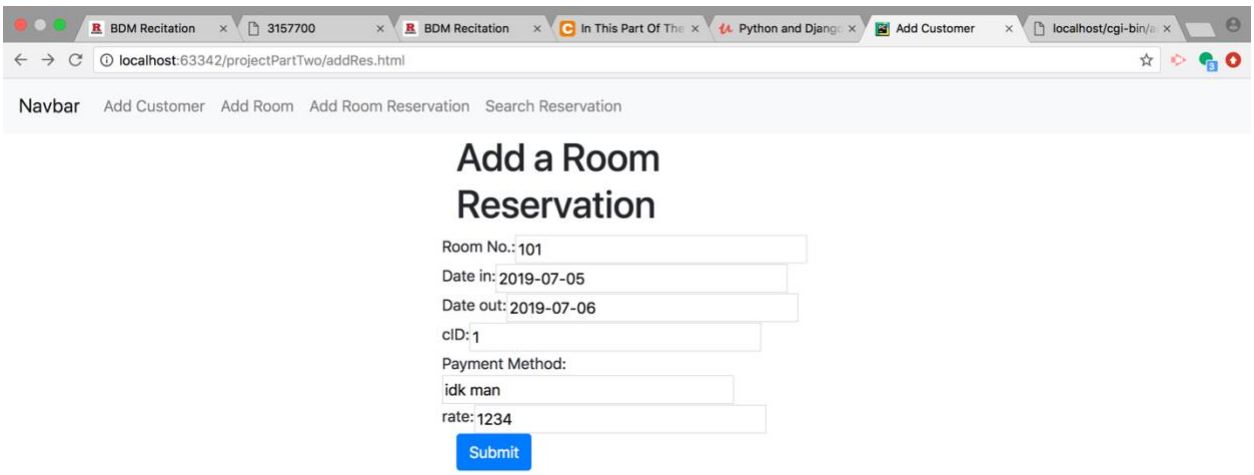
The python file adds the user with name nicky swag, cid 444, and dob 2019-03-31. It then returns the resulting relation in tabular form, and you can see that nicky swag is now the newest customer in the primaryCustomer table.

((1, 'John Abraham', datetime.date(1987, 2, 23)), (2, 'Alex Smith', datetime.date(1998, 3, 31)), (3, 'Mike Tirico', datetime.date(1987, 2, 23)), (4, 'Dana Olive', datetime.date(1987, 2, 23)), (5, 'Rebecca Martinez', datetime.date(1997, 8, 7)), (6, 'Stanley Hudson', datetime.date(1987, 2, 23)), (7, 'Meek Mill', datetime.date(1987, 2, 23)), (8, 'Michael Jordan', datetime.date(1987, 2, 23)), (9, 'Lebron James', datetime.date(1987, 2, 23)), (10, 'Eric Plesko', datetime.date(1987, 2, 23)), (11, 'Helicopter McFlurry', datetime.date(1987, 2, 23)), (12, 'Alligator Miami', datetime.date(1987, 2, 23)), (4444, 'nicky swag', datetime.date(2019, 3, 31)))



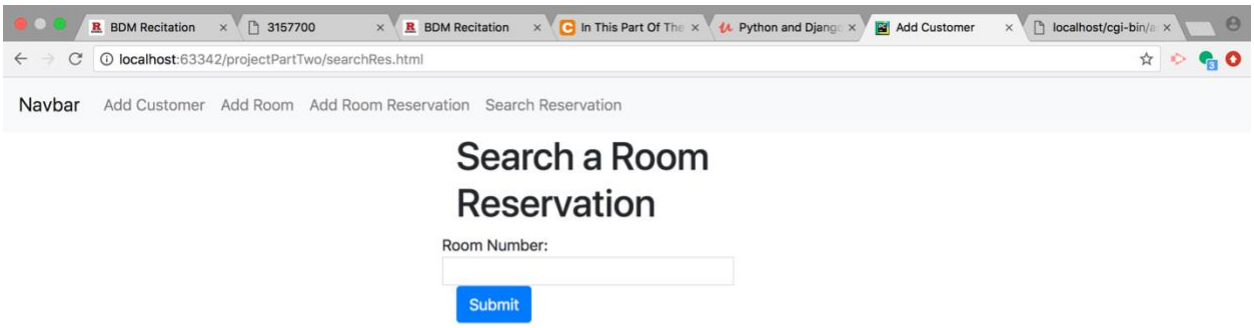
The python file takes the info from the html file and creates a new room in the room table. It then prints the room table and you can see the newest room (with room number 1222) is on the end.

((101, 2, 1, Decimal('0.85'), Decimal('300.00'), Decimal('250.00'), Decimal('310.00'), Decimal('200.00'), b'\x00', b'\x00', '...'), (212, 2, 2, Decimal('0.90'), Decimal('350.00'), Decimal('300.00'), Decimal('380.00'), Decimal('240.00'), b'\x00', b'\x01', '...'), (387, 4, 3, Decimal('0.70'), Decimal('234.00'), Decimal('200.00'), Decimal('260.00'), Decimal('210.00'), b'\x01', b'\x00', '...'), (420, 4, 4, Decimal('0.75'), Decimal('290.00'), Decimal('250.00'), Decimal('305.00'), Decimal('190.00'), b'\x01', b'\x00', '...'), (510, 2, 5, Decimal('0.82'), Decimal('330.00'), Decimal('300.00'), Decimal('400.00'), Decimal('200.00'), b'\x00', b'\x00', '...'), (660, 2, 6, Decimal('0.85'), Decimal('305.00'), Decimal('260.00'), Decimal('315.00'), Decimal('200.00'), b'\x00', b'\x01', '...'), (919, 2, 9, Decimal('0.97'), Decimal('1000.00'), Decimal('800.00'), Decimal('1200.00'), Decimal('750.00'), b'\x00', b'\x01', '...'), (1222, 123, 33, Decimal('0.98'), Decimal('123.00'), Decimal('123.00'), Decimal('890.00'), Decimal('800.00'), b'\x01', b'\x00', '..'))



The python file takes all of the information from the html file and creates a new reservation then prints the reservation table. As you can see on the second line of the printed data, there is a new reservation with room Number 101 and payment method of ‘idk man’

((101, datetime.datetime(2019, 1, 18, 4, 30, 7), datetime.datetime(2019, 1, 19, 6, 48, 15), 2, 'Check', Decimal('350.00')), (101, datetime.datetime(2019, 3, 12, 4, 30, 7), datetime.datetime(2019, 3, 17, 5, 34, 11), 4, 'Debit', Decimal('350.00')), (101, datetime.datetime(2019, 7, 5, 0, 0), datetime.datetime(2019, 7, 6, 0, 0), 1, 'idk man', Decimal('1234.00')), (212, datetime.datetime(2019, 3, 4, 4, 30, 7), datetime.datetime(2019, 3, 12, 9, 41, 51), 3, 'Credit', Decimal('350.00')), (212, datetime.datetime(2019, 3, 22, 4, 29, 11), datetime.datetime(2019, 3, 29, 8, 21, 33), 2, 'Check', Decimal('350.00')), (212, datetime.datetime(2019, 3, 31, 4, 30, 7), datetime.datetime(2019, 4, 2, 9, 50, 27), 4, 'Check', Decimal('350.00')), (387, datetime.datetime(2019, 1, 31, 4, 30, 7), datetime.datetime(2019, 2, 2, 5, 11, 45), 5, 'Credit', Decimal('350.00')), (387, datetime.datetime(2019, 3, 31, 4, 0, 19), datetime.datetime(2019, 4, 3, 10, 12, 27), 7, 'Cash', Decimal('350.00')), (420, datetime.datetime(2019, 3, 28, 4, 30, 7), datetime.datetime(2019, 4, 2, 10, 50, 22), 3, 'Credit', Decimal('350.00')), (420, datetime.datetime(2019, 4, 11, 4, 30, 7), datetime.datetime(2019, 4, 15, 7, 12, 44), 1, 'Cash', Decimal('350.00')), (510, datetime.datetime(2019, 2, 28, 4, 30, 7), datetime.datetime(2019, 3, 2, 11, 26, 26), 1, 'Cash', Decimal('350.00')), (510, datetime.datetime(2019, 3, 14, 4, 30, 7), datetime.datetime(2019, 3, 25, 6, 31, 43), 5, 'Check', Decimal('350.00')), (660, datetime.datetime(2019, 2, 12, 4, 30, 7), datetime.datetime(2019, 2, 17, 8, 9, 23), 6, 'Check', Decimal('350.00')), (660, datetime.datetime(2019, 3, 26, 4, 30, 7), datetime.datetime(2019, 3, 31, 7, 13, 51), 4, 'Check', Decimal('350.00')), (919, datetime.datetime(2019, 2, 14, 4, 30, 7), datetime.datetime(2019, 3, 4, 9, 33, 47), 7, 'Debit', Decimal('350.00')), (919, datetime.datetime(2019, 4, 12, 4, 30, 7), datetime.datetime(2019, 4, 20, 6, 56, 34), 3, 'Cash', Decimal('350.00')))



The python file takes the string from the textbox and runs an sql query where it searches the "books" table WHERE roomNumber = that value from the textbox. When we type 101 and hit submit we get the two reservation entries with roomNumber 101.

((101, datetime.datetime(2019, 1, 18, 4, 30, 7), datetime.datetime(2019, 1, 19, 6, 48, 15), 2, 'Check', Decimal('350.00')), (101, datetime.datetime(2019, 3, 12, 4, 30, 7), datetime.datetime(2019, 3, 17, 5, 34, 11), 4, 'Debit', Decimal('350.00')))