SQL mini project for Springboard. Steven Bowler.

Link to repo for this project (https://github.com/stevenbowler/SpringboardSQLminiProject). Link to notebook for this project (https://github.com/stevenbowler/SpringboardSQLminiProject/blob/master/SQLminiProject.ipynb). Link to this file (https://github.com/stevenbowler/SpringboardSQLminiProject/blob/master/SQLminiProject.pdf).

In [156]:
```
# Import necessary module
import pandas as pd
import mysql.connector
from mysql.connector import errorcode
```

In [31]:
```
# from https://dev.mysql.com/doc/connector-python/en/connector-python-example-connecting.html
try:
    db = mysql.connector.connect(
        host="localhost",
        user="root",
        password="",
        database="country_club"
        )
except mysql.connector.Error as err:
  if err.errno == errorcode.ER_ACCESS_DENIED_ERROR:
    print("Something is wrong with your user name or password")
  elif err.errno == errorcode.ER_BAD_DB_ERROR:
    print("Database does not exist")
  else:
    print(err)
# else:
#   cnx.close()
# db.close()   # at some point need to close the connection with this instruction

cursor = db.cursor(buffered=True)  # to avoid [error](https://stackoverflow.com/questions/29772337/python-mysql-connector-unread-result-found-when-using-fetchone)
```

In [30]:
```
cursor.close()
db.close()
```

In [34]:
```
# Build select statement for members table: stmt
query = 'SELECT * FROM members'

# Execute the statement and fetch the results: results
cursor.execute(query)

rows = cursor.fetchall()    # get all selected rows
for r in rows:
    print(r)
```

```
(0, 'GUEST', 'GUEST', 'GUEST', 0, '(000) 000-0000', '', '2012-07-01 00:00:0
0')
(1, 'Smith', 'Darren', '8 Bloomsbury Close, Boston', 4321, '555-555-5555',
'', '2012-07-02 12:02:05')
(2, 'Smith', 'Tracy', '8 Bloomsbury Close, New York', 4321, '555-555-5555',
'', '2012-07-02 12:08:23')
(3, 'Rownam', 'Tim', '23 Highway Way, Boston', 23423, '(844) 693-0723', '',
'2012-07-03 09:32:15')
(4, 'Joplette', 'Janice', '20 Crossing Road, New York', 234, '(833) 942-471
0', '1', '2012-07-03 10:25:05')
(5, 'Butters', 'Gerald', '1065 Huntingdon Avenue, Boston', 56754, '(844) 078-
4130', '1', '2012-07-09 10:44:09')
(6, 'Tracy', 'Burton', '3 Tunisia Drive, Boston', 45678, '(822) 354-9973',
'', '2012-07-15 08:52:55')
(7, 'Dare', 'Nancy', '6 Hunting Lodge Way, Boston', 10383, '(833) 776-4001',
'4', '2012-07-25 08:59:12')
(8, 'Boothe', 'Tim', '3 Bloomsbury Close, Reading, 00234', 234, '(811) 433-25
47', '3', '2012-07-25 16:02:35')
(9, 'Stibbons', 'Ponder', '5 Dragons Way, Winchester', 87630, '(833) 160-390
0', '6', '2012-07-25 17:09:05')
(10, 'Owen', 'Charles', '52 Cheshire Grove, Winchester, 28563', 28563, '(855)
542-5251', '1', '2012-08-03 19:42:37')
(11, 'Jones', 'David', '976 Gnats Close, Reading', 33862, '(844) 536-8036',
'4', '2012-08-06 16:32:55')
(12, 'Baker', 'Anne', '55 Powdery Street, Boston', 80743, '844-076-5141',
'9', '2012-08-10 14:23:22')
(13, 'Farrell', 'Jemima', '103 Firth Avenue, North Reading', 57392, '(855) 01
6-0163', '', '2012-08-10 14:28:01')
(14, 'Smith', 'Jack', '252 Binkington Way, Boston', 69302, '(822) 163-3254',
'1', '2012-08-10 16:22:05')
(15, 'Bader', 'Florence', '264 Ursula Drive, Westford', 84923, '(833) 499-352
7', '9', '2012-08-10 17:52:03')
(16, 'Baker', 'Timothy', '329 James Street, Reading', 58393, '833-941-0824',
'13', '2012-08-15 10:34:25')
(17, 'Pinker', 'David', '5 Impreza Road, Boston', 65332, '811 409-6734', '1
3', '2012-08-16 11:32:47')
(20, 'Genting', 'Matthew', '4 Nunnington Place, Wingfield, Boston', 52365,
'(811) 972-1377', '5', '2012-08-19 14:55:55')
(21, 'Mackenzie', 'Anna', '64 Perkington Lane, Reading', 64577, '(822) 661-28
98', '1', '2012-08-26 09:32:05')
(22, 'Coplin', 'Joan', '85 Bard Street, Bloomington, Boston', 43533, '(822) 4
99-2232', '16', '2012-08-29 08:32:41')
(24, 'Sarwin', 'Ramnaresh', '12 Bullington Lane, Boston', 65464, '(822) 413-1
470', '15', '2012-09-01 08:44:42')
(26, 'Jones', 'Douglas', '976 Gnats Close, Reading', 11986, '844 536-8036',
'11', '2012-09-02 18:43:05')
(27, 'Rumney', 'Henrietta', '3 Burkington Plaza, Boston', 78533, '(822) 989-8
876', '20', '2012-09-05 08:42:35')
(28, 'Farrell', 'David', '437 Granite Farm Road, Westford', 43532, '(855) 755
-9876', '', '2012-09-15 08:22:05')
(29, 'Worthington-Smyth', 'Henry', '55 Jagbi Way, North Reading', 97676, '(85
5) 894-3758', '2', '2012-09-17 12:27:15')
(30, 'Purview', 'Millicent', '641 Drudgery Close, Burnington, Boston', 34232,
'(855) 941-9786', '2', '2012-09-18 19:04:01')
(33, 'Tupperware', 'Hyacinth', '33 Cheerful Plaza, Drake Road, Westford', 686
66, '(822) 665-5327', '', '2012-09-18 19:32:05')
(35, 'Hunt', 'John', '5 Bullington Lane, Boston', 54333, '(899) 720-6978', '3
```

```
0', '2012-09-19 11:32:45')
(36, 'Crumpet', 'Erica', 'Crimson Road, North Reading', 75655, '(811) 732-481
6', '2', '2012-09-22 08:36:38')
(37, 'Smith', 'Darren', '3 Funktown, Denzington, Boston', 66796, '(822) 577-3
541', '', '2012-09-26 18:08:45')
```

SQL mini project question #1

In [52]:
```
query = 'SELECT name FROM facilities WHERE membercost > 0'

cursor.execute(query)

q1result = cursor.fetchall()

print('These are the facilites that charge members:')

for r in q1result:
    facility = r[0]
    print(facility)
```

```
These are the facilites that charge members:
Tennis Court 1
Tennis Court 2
Massage Room 1
Massage Room 2
Squash Court
```

SQL mini project question #2

In [51]:
```
query = 'SELECT COUNT( name )FROM Facilities WHERE membercost = 0'

cursor.execute(query)

q2result = cursor.fetchall()

for r in q2result:
    num1 = r[0] # convert the tuple to an integer
    print('There are', num1, 'free facilities.')
```

```
There are 4 free facilities.
```

SQL mini project question #3

In [59]:
```
query = 'SELECT * FROM facilities WHERE membercost < ( monthlymaintenance * 0.
2 )'

cursor.execute(query)

q3result = cursor.fetchall()

print('Facilites where membercost < 20% of montlymaintenance:\n')
print('ID','Name')
for r in q3result:
    facid = r[0] # convert the tuple to an integer
    facility = r[1]
    print(facid, facility)
```

```
Facilites where membercost < 20% of montlymaintenance:

ID Name
0 Tennis Court 1
1 Tennis Court 2
2 Badminton Court
3 Table Tennis
4 Massage Room 1
5 Massage Room 2
6 Squash Court
7 Snooker Table
8 Pool Table
```

SQL mini project question #4

In [61]:
```
query = 'SELECT * FROM facilities WHERE facid BETWEEN 1 AND 5 '

cursor.execute(query)

q4result = cursor.fetchall()

print('Facilites where facid in 1 to 5:\n')
print('ID','Name')
for r in q4result:
    facid = r[0] # convert the tuple to an integer
    facility = r[1]
    print(facid, facility)
```

```
Facilites where facid in 1 to 5:

ID Name
1 Tennis Court 2
2 Badminton Court
3 Table Tennis
4 Massage Room 1
5 Massage Room 2
```

SQL mini project question #5

In [100]:
```
query = "SELECT name, monthlymaintenance, CASE WHEN monthlymaintenance > 100 T
HEN 'expensive' ELSE 'cheap' END AS 'cheapexpensive' FROM facilities;"

cursor.execute(query)

q5result = cursor.fetchall()

data = {}
df5 = pd.DataFrame(data)

print('Facilites where expensive is > $100 per month:\n')

# Both print result to screen and append to dataframe
for r in q5result:
    name = r[0] # convert the tuple to an integer
    monthlymaint = r[1]
    cheapexpensive = r[2]
    data = {
        'name': r[0],
        'monthlymaint': r[1],
        'cheapexpensive': r[2]
    }
    df5 = df5.append(data, ignore_index=True)
#     print(name, monthlymaint, cheapexpensive)

df5 = df5[['name','monthlymaint','cheapexpensive']]
df5
```

```
Facilites where expensive is > $100 per month:
```

Out[100]:

	name	monthlymaint	cheapexpensive
0	Tennis Court 1	200.0	expensive
1	Tennis Court 2	200.0	expensive
2	Badminton Court	50.0	cheap
3	Table Tennis	10.0	cheap
4	Massage Room 1	3000.0	expensive
5	Massage Room 2	3000.0	expensive
6	Squash Court	80.0	cheap
7	Snooker Table	15.0	cheap
8	Pool Table	15.0	cheap

SQL mini project question #6

In [99]:
```
query = 'SELECT firstname, surname, joindate FROM members ORDER BY joindate DE
SC'

cursor.execute(query)

q6result = cursor.fetchall()

data = {}
df6 = pd.DataFrame(data)

print('Members and their Join Date :\n')

# Both print result to screen and append to dataframe
for r in q6result:
    surname = r[1] # convert the tuple to an integer
    firstname = r[0]
    joindate = r[2]
    data = {
        'surname': surname,
        'firstname': firstname,
        'joindate': joindate
    }
    df6 = df6.append(data, ignore_index=True)
#     print(surname,',', firstname, joindate)

df6 = df6[['surname','firstname','joindate']]
df6
```

```
Members and their Join Date :
```

Out[99]:

	surname	firstname	joindate
0	Smith	Darren	2012-09-26 18:08:45
1	Crumpet	Erica	2012-09-22 08:36:38
2	Hunt	John	2012-09-19 11:32:45
3	Tupperware	Hyacinth	2012-09-18 19:32:05
4	Purview	Millicent	2012-09-18 19:04:01
5	Worthington-Smyth	Henry	2012-09-17 12:27:15
6	Farrell	David	2012-09-15 08:22:05
7	Rumney	Henrietta	2012-09-05 08:42:35
8	Jones	Douglas	2012-09-02 18:43:05
9	Sarwin	Ramnaresh	2012-09-01 08:44:42
10	Coplin	Joan	2012-08-29 08:32:41
11	Mackenzie	Anna	2012-08-26 09:32:05
12	Genting	Matthew	2012-08-19 14:55:55
13	Pinker	David	2012-08-16 11:32:47
14	Baker	Timothy	2012-08-15 10:34:25
15	Bader	Florence	2012-08-10 17:52:03
16	Smith	Jack	2012-08-10 16:22:05
17	Farrell	Jemima	2012-08-10 14:28:01
18	Baker	Anne	2012-08-10 14:23:22
19	Jones	David	2012-08-06 16:32:55
20	Owen	Charles	2012-08-03 19:42:37
21	Stibbons	Ponder	2012-07-25 17:09:05
22	Boothe	Tim	2012-07-25 16:02:35
23	Dare	Nancy	2012-07-25 08:59:12
24	Tracy	Burton	2012-07-15 08:52:55
25	Butters	Gerald	2012-07-09 10:44:09
26	Joplette	Janice	2012-07-03 10:25:05
27	Rownam	Tim	2012-07-03 09:32:15
28	Smith	Tracy	2012-07-02 12:08:23
29	Smith	Darren	2012-07-02 12:02:05
30	GUEST	GUEST	2012-07-01 00:00:00

SQL mini project question #7

In [116]:
```
query = (
    "select distinct b.facid, f.name, concat(m.firstname,' ',m.surname) as mem
bername "
    "from Bookings as b "
    "inner join Members as m "
        "on m.memid = b.memid "
    "inner join Facilities as f "
        "on f.facid = b.facid "
    "where f.facid = 0 or f.facid = 1 "
    "order by membername"
        )


cursor.execute(query)

q7result = cursor.fetchall()

data = {}
df7 = pd.DataFrame(data)

print('Members that have used a tennis court :\n')

# Both print result to screen and append to dataframe
for r in q7result:
    facid = r[0] # convert the tuple to an integer
    facility = r[1]
    membername = r[2]
    data = {
        'facid': facid,
        'facility': facility,
        'membername': membername
    }
    df7 = df7.append(data, ignore_index=True)
#     print(surname,',', firstname, joindate)

# df7['facid'] = df7['facid'].round(decimals=0)
df7['facid'] = df7['facid'].astype(int)
df7
```

Members that have used a tennis court :

Out[116]:

	facid	facility	membername
0	1	Tennis Court 2	Anne Baker
1	0	Tennis Court 1	Anne Baker
2	1	Tennis Court 2	Burton Tracy
3	0	Tennis Court 1	Burton Tracy
4	1	Tennis Court 2	Charles Owen
5	0	Tennis Court 1	Charles Owen
6	1	Tennis Court 2	Darren Smith
7	0	Tennis Court 1	David Farrell
8	1	Tennis Court 2	David Farrell
9	1	Tennis Court 2	David Jones
10	0	Tennis Court 1	David Jones
11	0	Tennis Court 1	David Pinker
12	0	Tennis Court 1	Douglas Jones
13	0	Tennis Court 1	Erica Crumpet
14	0	Tennis Court 1	Florence Bader
15	1	Tennis Court 2	Florence Bader
16	1	Tennis Court 2	Gerald Butters
17	0	Tennis Court 1	Gerald Butters
18	1	Tennis Court 2	GUEST GUEST
19	0	Tennis Court 1	GUEST GUEST
20	1	Tennis Court 2	Henrietta Rumney
21	1	Tennis Court 2	Jack Smith
22	0	Tennis Court 1	Jack Smith
23	0	Tennis Court 1	Janice Joplette
24	1	Tennis Court 2	Janice Joplette
25	0	Tennis Court 1	Jemima Farrell
26	1	Tennis Court 2	Jemima Farrell
27	0	Tennis Court 1	Joan Coplin
28	1	Tennis Court 2	John Hunt
29	0	Tennis Court 1	John Hunt
30	0	Tennis Court 1	Matthew Genting
31	1	Tennis Court 2	Millicent Purview
32	1	Tennis Court 2	Nancy Dare
33	0	Tennis Court 1	Nancy Dare
34	0	Tennis Court 1	Ponder Stibbons

	facid	facility	membername
35	1	Tennis Court 2	Ponder Stibbons
36	1	Tennis Court 2	Ramnaresh Sarwin
37	0	Tennis Court 1	Ramnaresh Sarwin
38	0	Tennis Court 1	Tim Boothe
39	1	Tennis Court 2	Tim Boothe
40	0	Tennis Court 1	Tim Rownam
41	1	Tennis Court 2	Tim Rownam
42	1	Tennis Court 2	Timothy Baker
43	0	Tennis Court 1	Timothy Baker
44	1	Tennis Court 2	Tracy Smith
45	0	Tennis Court 1	Tracy Smith

SQL mini project question #8

In [124]:
```
query = (
        "select b.facid, b.starttime, f.name as facility, concat(m.firstname,'
',m.surname) as membername, "
            "case when (m.firstname = 'GUEST' and f.guestcost * b.slots > 30)
 then f.guestcost * b.slots "
            "when (m.firstname != 'GUEST' and f.membercost * b.slots > 30) the
n f.membercost * b.slots "
            "end as cost "
        "from Bookings as b "
        "inner join Members as m "
            "on m.memid = b.memid "
        "inner join Facilities as f "
            "on f.facid = b.facid "
        "having b.starttime like '2012-09-14%' and cost is not null "
        "order by cost desc;"
)

cursor.execute(query)

q8result = cursor.fetchall()

data = {}
df8 = pd.DataFrame(data)

print('Members and guests that paid more than $30 for a tennis court :\n')

# Both print result to screen and append to dataframe
for r in q8result:
    facid = r[0] # convert the tuple to an integer
    starttime = r[1]
    facility = r[2]
    membername = r[3]
    cost = r[4]
    data = {
        'facid': facid,
        'starttime': starttime,
        'facility': facility,
        'membername': membername,
        'cost': cost
    }
    df8 = df8.append(data, ignore_index=True)
#     print(surname,',', firstname, joindate)

df8['facid'] = df8['facid'].astype(int)
df8
```

```
Members and guests that paid more than $30 for a tennis court :
```

Out[124]:

	cost	facid	facility	membername	starttime
0	320.0	5	Massage Room 2	GUEST GUEST	2012-09-14 11:00:00
1	160.0	4	Massage Room 1	GUEST GUEST	2012-09-14 13:00:00
2	160.0	4	Massage Room 1	GUEST GUEST	2012-09-14 16:00:00
3	160.0	4	Massage Room 1	GUEST GUEST	2012-09-14 09:00:00
4	150.0	1	Tennis Court 2	GUEST GUEST	2012-09-14 17:00:00
5	75.0	1	Tennis Court 2	GUEST GUEST	2012-09-14 14:00:00
6	75.0	0	Tennis Court 1	GUEST GUEST	2012-09-14 16:00:00
7	75.0	0	Tennis Court 1	GUEST GUEST	2012-09-14 19:00:00
8	70.0	6	Squash Court	GUEST GUEST	2012-09-14 09:30:00
9	39.6	4	Massage Room 1	Jemima Farrell	2012-09-14 14:00:00
10	35.0	6	Squash Court	GUEST GUEST	2012-09-14 12:30:00
11	35.0	6	Squash Court	GUEST GUEST	2012-09-14 15:00:00

SQL mini project question #9

In [137]:
```
query = (
    "select b.facid, b.starttime, f.name as facility, concat(m.firstname,' ', 
m.surname) as membername, "
        "(select distinct f.guestcost * b.slots from Facilities where f.guestc
ost * b.slots > 30 and m.firstname = 'GUEST') "
#         "(select distinct f.membercost * b.slots from Facilities where f.mem
bercost * b.slots > 30 and m.firstname != 'GUEST') "
    "as cost "
    "from Bookings as b "
    "inner join Members as m "
        "on m.memid = b.memid "
    "inner join Facilities as f "
        "on f.facid = b.facid "
    "having b.starttime like '2012-09-14%' and  cost is not null "
    "order by cost desc;"
        )


cursor.execute(query)

q9result = cursor.fetchall()

data = {}
df9 = pd.DataFrame(data)

print('Members and guests that paid more than $30 for a facility :\n')

# Both print result to screen and append to dataframe
for r in q9result:
    facid = r[0] # convert the tuple to an integer
    starttime = r[1]
    facility = r[2]
    membername = r[3]
    cost = r[4]
    data = {
        'facid': facid,
        'starttime': starttime,
        'facility': facility,
        'membername': membername,
        'cost': cost
    }
    df9 = df9.append(data, ignore_index=True)
#     print(surname,',', firstname, joindate)

df9['facid'] = df9['facid'].astype(int)
df9
```

```
Members and guests that paid more than $30 for a facility :
```

Out[137]:

	cost	facid	facility	membername	starttime
0	320.0	5	Massage Room 2	GUEST GUEST	2012-09-14 11:00:00
1	160.0	4	Massage Room 1	GUEST GUEST	2012-09-14 13:00:00
2	160.0	4	Massage Room 1	GUEST GUEST	2012-09-14 16:00:00
3	160.0	4	Massage Room 1	GUEST GUEST	2012-09-14 09:00:00
4	150.0	1	Tennis Court 2	GUEST GUEST	2012-09-14 17:00:00
5	75.0	0	Tennis Court 1	GUEST GUEST	2012-09-14 19:00:00
6	75.0	1	Tennis Court 2	GUEST GUEST	2012-09-14 14:00:00
7	75.0	0	Tennis Court 1	GUEST GUEST	2012-09-14 16:00:00
8	70.0	6	Squash Court	GUEST GUEST	2012-09-14 09:30:00
9	35.0	6	Squash Court	GUEST GUEST	2012-09-14 12:30:00
10	35.0	6	Squash Court	GUEST GUEST	2012-09-14 15:00:00

SQL mini project question #10

In [135]:
```
query = (
    "select f.name as facility, "
            "sum(case when m.firstname = 'GUEST' then f.guestcost * b.slots "
                    "when m.firstname != 'GUEST' then f.membercost * b.slots "
                    "end) as revenue "
    "from bookings as b "
    "inner join members as m "
        "on m.memid = b.memid "
    "inner join facilities as f "
        "on f.facid = b.facid "
    "group by f.name "
    "having revenue < 1000 "
    "order by revenue desc;"
        )


cursor.execute(query)

q10result = cursor.fetchall()

data = {}
df10 = pd.DataFrame(data)

print('Facilities with less than $1000 revenue :\n')

# Both print result to screen and append to dataframe
for r in q10result:
    facility = r[0]
    revenue = r[1]
    data = {
        'facility': facility,
        'revenue': revenue
    }
    df10 = df10.append(data, ignore_index=True)

df10
```

```
Facilities with less than $1000 revenue :
```

Out[135]:

	facility	revenue
0	Pool Table	270.0
1	Snooker Table	240.0
2	Table Tennis	180.0

SQL mini project question #11

In [141]:
```
query = (
        "select a.memid as memberID, "
            "concat(a.surname,', ',a.firstname) as member, "
            "b.memid as recommenderID, "
            "concat(b.surname,', ',b.firstname) as recommender "
        "from members a, members b "
        "where a.recommendedby = b.memid and a.memid != 0 and b.memid != 0 "
        "order by a.surname;"
        )


cursor.execute(query)

q11result = cursor.fetchall()

data = {}
df11 = pd.DataFrame(data)

print('Members and who recommended them :\n')

# Both print result to screen and append to dataframe
for r in q11result:
    memberID = r[0]
    member = r[1]
    recommenderID = r[2]
    recommender = r[3]
    data = {
        'memberID': memberID,
        'member': member,
        'recommenderID': recommenderID,
        'recommender': recommender
    }
    df11 = df11.append(data, ignore_index=True)


df11['memberID'] = df11['memberID'].astype(int)
df11['recommenderID'] = df11['recommenderID'].astype(int)
df11
```

```
Members and who recommended them :
```

Out[141]:

	member	memberID	recommender	recommenderID
0	Bader, Florence	15	Stibbons, Ponder	9
1	Baker, Timothy	16	Farrell, Jemima	13
2	Baker, Anne	12	Stibbons, Ponder	9
3	Boothe, Tim	8	Rownam, Tim	3
4	Butters, Gerald	5	Smith, Darren	1
5	Coplin, Joan	22	Baker, Timothy	16
6	Crumpet, Erica	36	Smith, Tracy	2
7	Dare, Nancy	7	Joplette, Janice	4
8	Genting, Matthew	20	Butters, Gerald	5
9	Hunt, John	35	Purview, Millicent	30
10	Jones, Douglas	26	Jones, David	11
11	Jones, David	11	Joplette, Janice	4
12	Joplette, Janice	4	Smith, Darren	1
13	Mackenzie, Anna	21	Smith, Darren	1
14	Owen, Charles	10	Smith, Darren	1
15	Pinker, David	17	Farrell, Jemima	13
16	Purview, Millicent	30	Smith, Tracy	2
17	Rumney, Henrietta	27	Genting, Matthew	20
18	Sarwin, Ramnaresh	24	Bader, Florence	15
19	Smith, Jack	14	Smith, Darren	1
20	Stibbons, Ponder	9	Tracy, Burton	6
21	Worthington-Smyth, Henry	29	Smith, Tracy	2

SQL mini project question #12

In [149]:
```
query = (
    "SELECT b.facid as facilityID, "
        "sum(b.slots) as memberusage "
    "FROM bookings as b "
    "WHERE b.memid != 0 "
    "group by b.facid "
    "order by b.facid;"
        )


cursor.execute(query)

q12result = cursor.fetchall()

data = {}
df12 = pd.DataFrame(data)

print('Facility ID and total usage :\n')

# Both print result to screen and append to dataframe
for r in q12result:
    facilityID = r[0]
    memberusage = r[1]
    data = {
        'facilityID': facilityID,
        'memberusage': memberusage
    }
    df12 = df12.append(data, ignore_index=True)

df12['facilityID'] = df12['facilityID'].astype(int)
df12
```

```
Facility ID and total usage :
```

Out[149]:

	facilityID	memberusage
0	0	957
1	1	882
2	2	1086
3	3	794
4	4	884
5	5	54
6	6	418
7	7	860
8	8	856

SQL mini project question #13

In [155]:
```
query = (
        "SELECT month(b.starttime) as month, "
            "sum(b.slots) as facilityusage "
        "FROM bookings as b "
        "WHERE b.memid != 0 "
        "group by month(b.starttime) "
        "order by month(b.starttime);"
        )

cursor.execute(query)

q13result = cursor.fetchall()

data = {}
df13 = pd.DataFrame(data)

print('Total usage by month :\n')

# Both print result to screen and append to dataframe
for r in q13result:
    month = r[0]
    facilityusage = r[1]
    data = {
        'month': month,
        'facilityusage': facilityusage
    }
    df13 = df13.append(data, ignore_index=True)

df13['month'] = df13['month'].astype(int)
df13 = df13[['month','facilityusage']]
df13
```

```
Total usage by month :
```

Out[155]:

	month	facilityusage
0	7	1061
1	8	2531
2	9	3199

/ *Q1: Some of the facilities charge a fee to members, but some do not. Write a SQL query to produce a list of the names of the facilities that do.* /

SELECT name FROM Facilities WHERE membercost >0 LIMIT 0 , 30

/ *Q2: How many facilities do not charge a fee to members?* /

SELECT COUNT(name) FROM Facilities WHERE membercost =0

/ *Q3: Write an SQL query to show a list of facilities that charge a fee to members, where the fee is less than 20% of the facility's monthly maintenance cost. Return the facid, facility name, member cost, and monthly maintenance of the facilities in question.* /

SELECT *FROM `Facilities` WHERE membercost < (monthlymaintenance* 0.2) LIMIT 0 , 30

/ *Q4: Write an SQL query to retrieve the details of facilities with ID 1 and 5. Try writing the query without using the OR operator.* /

SELECT * FROM `Facilities` WHERE facid IN (1, 5) LIMIT 0 , 30

/ *Q5: Produce a list of facilities, with each labelled as 'cheap' or 'expensive', depending on if their monthly maintenance cost is more than $100. Return the name and monthly maintenance of the facilities in question.* /

SELECT name, monthlymaintenance, CASE WHEN monthlymaintenance >100 THEN 'expensive' ELSE 'cheap' END AS cheapexpensive FROM `Facilities` LIMIT 0 , 30

/ *Q6: You'd like to get the first and last name of the last member(s) who signed up. Try not to use the LIMIT clause for your solution.* /

SELECT firstname, surname, joindate FROM `Members` ORDER BY joindate DESC LIMIT 0 , 30

/ *Q7: Produce a list of all members who have used a tennis court. Include in your output the name of the court, and the name of the member formatted as a single column. Ensure no duplicate data, and order by the member name.* /

select b.facid, f.name, concat(m.firstname,' ',m.surname) as membername, b.cost from Bookings as b inner join Members as m on m.memid = b.memid inner join Facilities as f on f.facid = b.facid where b.date = '2012-09-14' and f.facid == 0 or f.facid == 1 order by membername

/ *Q8: Produce a list of bookings on the day of 2012-09-14 which will cost the member (or guest) more than $30. Remember that guests have different costs to members (the listed costs are per half-hour 'slot'), and the guest user's ID is always 0. Include in your output the name of the facility, the name of the member formatted as a single column, and the cost. Order by descending cost, and do not use any subqueries.* /

select b.facid, b.starttime, f.name as facility, concat(m.firstname,' ',m.surname) as membername, case when (m.firstname = 'GUEST' and f.guestcost *b.slots > 30) then f.guestcost* b.slots when (m.firstname != 'GUEST' and f.membercost *b.slots > 30) then f.membercost* b.slots end as cost from Bookings as b inner join Members as m on m.memid = b.memid inner join Facilities as f on f.facid = b.facid having b.starttime like '2012-09-14%' and cost is not null order by cost desc;

/ *Q9: This time, produce the same result as in Q8, but using a subquery.* /

select b.facid, b.starttime, f.name as facility, concat(m.firstname,' ',m.surname) as membername, (select distinct f.guestcost *b.slots from Facilities where f.guestcost* b.slots > 30 and m.firstname = 'GUEST') + (select distinct f.membercost *b.slots from Facilities where f.membercost* b.slots > 30 and m.firstname != 'GUEST') as cost from Bookings as b inner join Members as m on m.memid = b.memid inner join Facilities as f on f.facid = b.facid having b.starttime like '2012-09-14%' and cost is not null order by cost desc;

/* PART 2: SQLite

Export the country club data from PHPMyAdmin, and connect to a local SQLite instance from Jupyter notebook for the following questions.

QUESTIONS: / *Q10: Produce a list of facilities with a total revenue less than 1000. The output of facility name and total revenue, sorted by revenue. Remember that there's a different cost for guests and members!* /

select f.name as facility,
sum(case when m.firstname = 'GUEST' then f.guestcost *b.slots when m.firstname != 'GUEST' then f.membercost* b.slots end) as revenue from bookings as b inner join members as m on m.memid = b.memid inner join facilities as f on f.facid = b.facid group by f.name having revenue > 1000 order by revenue desc;

/ *Q11: Produce a report of members and who recommended them in alphabetic surname,firstname order* /

select a.memid as memberID, concat(a.surname,', ',a.firstname) as member, b.memid as recommenderID, concat(b.surname,', ',b.firstname) as recommender from members a, members b where a.recommendedby = b.memid and a.memid != 0 and b.memid != 0 order by a.surname;

/ *Q12: Find the facilities with their usage by member, but not guests* /

SELECT b.facid as facilityID, sum(b.slots) as memberusage FROM bookings as b WHERE b.memid != 0 group by b.facid order by b.facid

/ *Q13: Find the facilities usage by month, but not guests* /

SELECT month(b.starttime) as month, sum(b.slots) as facilityusage FROM bookings as b WHERE b.memid != 0 group by month(b.starttime) order by month(b.starttime)