/\* Welcome to the SQL mini project. You will carry out this project partly in

the PHPMyAdmin interface, and partly in Jupyter via a Python connection.

URL: https://sql.springboard.com/

Username: student

Password: learn\_sql@springboard

The data you need is in the "country\_club" database. This database

contains 3 tables:

i) the "Bookings" table,

ii) the "Facilities" table, and

iii) the "Members" table.

/\* QUESTIONS

/\* 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, membercost

FROM Facilities

WHERE membercost <> 0.0;

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

4

SELECT COUNT(membercost)

FROM Facilities

WHERE membercost = 0.0;

/\* 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 facid, name, membercost, monthlymaintenance

FROM Facilities

WHERE membercost <> 0.0 and membercost < .2\*monthlymaintenance;

/\* 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);

/\* 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,

CASE WHEN monthlymaintenance < 100 THEN 'cheap'

ELSE 'expensive' END AS cheapexpensive

FROM Facilities;

/\* 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 surname, firstname, joindate

FROM Members

ORDER BY joindate DESC;

/\* 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 DISTINCT f.name,

CONCAT(surname, ', ', firstname) as membername

FROM Bookings as b

INNER JOIN Members as m

ON b.memid = m.memid

INNER JOIN Facilities as f

ON b.facid = f.facid

WHERE b.facid IN (0,1)

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.\*/

SELECT f.name as facilityname,

CONCAT (surname,', ',firstname) as bookingname,

CASE WHEN b.memid = 0 THEN guestcost\*b.slots

ELSE membercost\*slots

END AS booking\_cost

FROM ((Bookings as b INNER JOIN Facilities as f

on b.facid = f.facid)

INNER JOIN Members as m

on b.memid = m.memid)

WHERE starttime LIKE '2012-09-14%'

HAVING booking\_cost > 30

ORDER BY booking\_cost DESC;

/\* Q9: 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 facilityname,

SUM(CASE WHEN b.memid = 0 THEN guestcost\*b.slots

ELSE membercost\*slots

END) AS total\_revenue

FROM ((Bookings as b INNER JOIN Facilities as f

on b.facid = f.facid)

INNER JOIN Members as m

on b.memid = m.memid)

GROUP BY facilityname

HAVING total\_revenue < 1000

ORDER BY total\_revenue DESC;

/\* Q10: Produce a report of members and who recommended them in alphabetic surname,firstname order \*/

SELECT CONCAT(A.surname, ', ', A.firstname) as membername,

A.recommendedby,

CONCAT(B.surname, ', ', B.firstname) as recommendername

FROM Members A, Members B

WHERE A.recommendedby <> ' '

ORDER BY recommendername;

/\* Q11: Find the facilities with their usage by member, but not guests \*/

SELECT

firstname, surname,

f.name as facilityname,

count(b.bookid) as facililtyusage

FROM Bookings as b INNER JOIN Facilities as f

on b.facid = f.facid

INNER JOIN Members as m

on b.memid = m.memid

WHERE b.memid <> 0

GROUP BY b.memid, facilityname

ORDER BY b.memid;

/\* Q12: Find the facilities usage by month, but not guests \*/

SELECT

f.name as facilityname,

EXTRACT(MONTH FROM starttime) as month,

count(b.bookid) as facililtyusage

FROM Bookings as b INNER JOIN Facilities as f

on b.facid = f.facid

INNER JOIN Members as m

on b.memid = m.memid

WHERE b.memid <> 0

GROUP BY month, facilityname

ORDER BY facilityname, month;