/* 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.
This is Tier 2 of the case study, which means that there'll be less guidance for you about how to setup
your local SQLite connection in PART 2 of the case study. This will make the case study more challenging for you:
you might need to do some digging, aand revise the Working with Relational Databases in Python chapter in the previous resource.
Otherwise, the questions in the case study are exactly the same as with Tier 1.
PART 1: PHPMyAdmin
You will complete questions 1-9 below in the PHPMyAdmin interface.
Log in by pasting the following URL into your browser, and
using the following Username and Password:
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.
In this case study, you'll be asked a series of questions. You can
solve them using the platform, but for the final deliverable,
paste the code for each solution into this script, and upload it
to your GitHub.
Before starting with the questions, feel free to take your time,
exploring the data, and getting acquainted with the 3 tables. */

/* 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 FROM Facilities
WHERE membercost !=0;

Output
Tennis Court 1
Tennis Court 2
Massage Room 1
Massage Room 2
Squash Court

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

```
SELECT count(membercost) FROM Facilities
WHERE membercost !=0;
```

Output:

```
count(membercost)
5
```

/* 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 and
membercost<0.2 * monthlymaintenance;
```

Output

facid	name	membercost	monthlymaintenance
0	Tennis Court 1	5.0	200
1	Tennis Court 2	5.0	200
4	Massage Room 1	9.9	3000
5	Massage Room 2	9.9	3000
6	Squash Court	3.5	80

/* 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);
```

Output

facid	name	membercost	guestcost	initialoutlay	monthlymaintenance
1	Tennis Court 2	5.0	25.0	8000	200
5	Massage Room 2	9.9	80.0	4000	3000

/* 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 'cheap'
else 'expensive'
end as type
from Facilities;

```
select name,monthlymaintenance,
case when monthlymaintenance <=100 then 'cheap'
else 'expensive'
end as type
from Facilities;
```

Output

name	monthlymaintenance	type
Tennis Court 1	200	expensive
Tennis Court 2	200	expensive
Badminton Court	50	cheap
Table Tennis	10	cheap
Massage Room 1	3000	expensive
Massage Room 2	3000	expensive
Squash Court	80	cheap
Snooker Table	15	cheap
Pool Table	15	cheap

/* 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 from Members

where joindate in (select max(joindate) from Members);

```
select firstname,surname from Members

where joindate in (select max(joindate) from
Members);
```

Output:

firstname	surname
Darren	Smith

/* 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 concat_ws(" ",s.first,s.last) as fullname,s.tenniscourt
from
(select distinct f.name as tenniscourt,m.firstname as first,m.surname as last
 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 f.name like 'Tennis%') as s
order by fullname;
```

```
select concat_ws(" ",s.first,s.last) as fullname,s.tenniscourt
from
(select distinct f.name as tenniscourt,m.firstname as first,m.surname as last
 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 f.name like 'Tennis%') as s
order by fullname;
```

Output

fullname	tenniscourt
Anne Baker	Tennis Court 2
Anne Baker	Tennis Court 1
Burton Tracy	Tennis Court 1
Burton Tracy	Tennis Court 2
Charles Owen	Tennis Court 2
Charles Owen	Tennis Court 1
Darren Smith	Tennis Court 2
David Farrell	Tennis Court 1
David Farrell	Tennis Court 2
David Jones	Tennis Court 2
David Jones	Tennis Court 1
David Pinker	Tennis Court 1
Douglas Jones	Tennis Court 1
Erica Crumpet	Tennis Court 1
Florence Bader	Tennis Court 2
Florence Bader	Tennis Court 1
Gerald Butters	Tennis Court 1

/* 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 distinct concat_ws(" ", m.firstname, m.surname) AS fullname, f.name AS facility,
case when b.memid =0
then f.guestcost * b.slots
else f.membercost * b.slots
end as 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 b.starttime like '12-09-14%'
and
(case when b.memid =0
then f.guestcost * b.slots
else f.membercost * b.slots end)>30
order by cost desc

```
select distinct concat_ws(" ", m.firstname, m.surname ) AS fullname, f.name AS facility,
case when b.memid =0
then f.guestcost * b.slots
else f.membercost * b.slots
end as 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 b.starttime like '12-09-14%'
and
(case when b.memid =0
then f.guestcost * b.slots
else f.membercost * b.slots end)>30
order by cost desc
/*
```

Output

fullname	facility	cost
GUEST GUEST	Massage Room 2	320.0
GUEST GUEST	Massage Room 1	160.0
GUEST GUEST	Tennis Court 2	150.0
GUEST GUEST	Tennis Court 2	75.0
GUEST GUEST	Tennis Court 1	75.0
GUEST GUEST	Squash Court	70.0
Jemima Farrell	Massage Room 1	39.6
GUEST GUEST	Squash Court	35.0

/* Q9: This time, produce the same result as in Q8, but using a subquery. */
select distinct concat_ws(" ", m.firstname, m.surname) AS fullname, f.name AS facility,
case when b.memid =0
then f.guestcost * b.slots
else f.membercost * b.slots
end as 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 b.starttime like '12-09-14%'
and b.memid in (select b.memid
from Bookings as b
inner join Facilities as f on b.facid = f.facid
where ((b.memid =0 and f.guestcost * b.slots >30) or (b.memid >0 and f.membercost * b.slots >30))
order by cost desc;

```
select distinct concat_ws(" ", m.firstname, m.surname ) AS fullname, f.name AS facility,
case when b.memid =0
then f.guestcost * b.slots
else f.membercost * b.slots
end as 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 b.starttime like '12-09-14%'
and b.memid in (select b.memid
from Bookings as b
inner join Facilities as f on b.facid = f.facid
where ((b.memid =0 and f.guestcost * b.slots >30) or (b.memid >0 and  f.membercost * b.slots >30))
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! */

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

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

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