3) PostGreSQL Exercise

1. How can you produce a list of the start times for bookings by members named 'David Farrell'?

Query:-

**select** bks.starttime

**from**

cd.bookings bks,

cd.members mems

**where**

mems.firstname='David'

**and** mems.surname='Farrell'

**and** mems.memid = bks.memid;

1. How can you produce a list of the start times for bookings for tennis courts, for the date '2012-09-21'? Return a list of start time and facility name pairings, ordered by the time.

Query:-

**select** bks.starttime **as** **start**, facs.name **as** name

**from**

cd.facilities facs

**inner** **join** cd.bookings bks

**on** facs.facid = bks.facid

**where**

facs.facid **in** (0,1) **and**

bks.starttime >= '2012-09-21' **and**

bks.starttime < '2012-09-22'

**order** **by** bks.starttime;

1. How can you output a list of all members who have recommended another member? Ensure that there are no duplicates in the list, and that results are ordered by (surname, firstname).

Query:-

**select** **distinct** recs.firstname **as** firstname, recs.surname **as** surname

**from**

cd.members mems

**inner** **join** cd.members recs

**on** recs.memid = mems.recommendedby

**order** **by** surname, firstname;

1. How can you output a list of all members, including the individual who recommended them (if any)? Ensure that results are ordered by (surname, firstname).

Query:-

**select** mems.firstname **as** memfname, mems.surname **as** memsname,

recs.firstname **as** recfname, recs.surname **as** recsname

**from**

cd.members mems

**left** **outer** **join** cd.members recs

**on** recs.memid = mems.recommendedby

**order** **by** memsname, memfname;

1. How can you 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.

Query:-

**select** **distinct** mems.firstname || ' ' || mems.surname **as** member,

facs.name **as** facility

**from**

cd.members mems

**inner** **join** cd.bookings bks

**on** mems.memid = bks.memid

**inner** **join** cd.facilities facs

**on** bks.facid = facs.facid

**where**

bks.facid **in** (0,1)

**order** **by** member;

1. How can you 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 is always ID 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.

Query:-

**select** mems.firstname || ' ' || mems.surname **as** member,

facs.name **as** facility,

**case**

**when** mems.memid = 0 **then**

bks.slots\*facs.guestcost

**else**

bks.slots\*facs.membercost

**end** **as** cost

**from**

cd.members mems

**inner** **join** cd.bookings bks

**on** mems.memid = bks.memid

**inner** **join** cd.facilities facs

**on** bks.facid = facs.facid

**where**

bks.starttime >= '2012-09-14' **and**

bks.starttime < '2012-09-15' **and** (

(mems.memid = 0 **and** bks.slots\*facs.guestcost > 30) **or**

(mems.memid != 0 **and** bks.slots\*facs.membercost > 30)

)

**order** **by** cost **desc**;

1. How can you output a list of all members, including the individual who recommended them (if any), without using any joins? Ensure that there are no duplicates in the list, and that each firstname + surname pairing is formatted as a column and ordered.

Query:-

**select** **distinct** mems.firstname || ' ' || mems.surname **as** member,

(**select** recs.firstname || ' ' || recs.surname **as** recommender

**from** cd.members recs

**where** recs.memid = mems.recommendedby

)

**from**

cd.members mems

**order** **by** member;

1. The [Produce a list of costly bookings](https://pgexercises.com/questions/joins/threejoin2.html) exercises contained some messy logic: we had to calculate the booking cost in both the WHERE clause and the CASE statement. Try to simplify this calculation using subqueries.

Query:-

**select** member, facility, cost **from** (

**select**

mems.firstname || ' ' || mems.surname **as** member,

facs.name **as** facility,

**case**

**when** mems.memid = 0 **then**

bks.slots\*facs.guestcost

**else**

bks.slots\*facs.membercost

**end** **as** cost

**from**

cd.members mems

**inner** **join** cd.bookings bks

**on** mems.memid = bks.memid

**inner** **join** cd.facilities facs

**on** bks.facid = facs.facid

**where**

bks.starttime >= '2012-09-14' **and**

bks.starttime < '2012-09-15'

) **as** bookings

**where** cost > 30

**order** **by** cost **desc**;

4) Hackerrank MySQL Challenge

