```
Introduction to Databases                               Fall-Winter 2009/10
---------------------------------------------------------------------------
Exercises                                                Michail Kazimianec
                                                                Werner Nutt
                                                            Damiano Somenzi
```

SQL Queries (II)

-- Sample Solutions --

As the name says, these are sample solutions, that is, they are only some of the many different possible solutions.

QUERIES OVER THE SAILOR DATABASE

For each of the queries ... find at least 2 different but equivalent formulations.

- 1. Select, for each boat, the sailor who made he highest number of reservations for that boat.

```
CREATE VIEW resbyboat(bid,bname,sid,sname,numberOfRes) AS
SELECT b.id, b.name, s.id, s.name, COUNT(*)
FROM   boat b JOIN reservation r ON b.id = r.bid
              JOIN sailor s ON s.id = r.sid
GROUP BY b.id, b.name, s.id, s.name;

SELECT *
FROM   resbyboat r
WHERE  r.numberOfRes =
       (SELECT MAX(r0.numberOfRes)
        FROM   resbyboat r0
        WHERE  r0.bid = r.bid);


SELECT DISTINCT b.name, s.name, COUNT(*)
FROM   boat b JOIN reservation r ON b.id = r.bid
              JOIN sailor s ON s.id = r.sid
GROUP BY b.id, b.name, s.id, s.name
HAVING COUNT(*) >= ALL
       (SELECT COUNT(*)
        FROM   reservation r0
        WHERE  r0.bid = b.id
        GROUP BY r0.sid)
ORDER BY b.name, s.name;
```

- 2. List, for every boat, the number of times it has been reserved, including those boats that have never been reserved (list the id and the name).

```
SELECT b.id, b.name, COUNT(r.sid)
FROM   boat b LEFT JOIN reservation r ON b.id = r.bid
GROUP BY b.id, b.name
ORDER BY b.id


-- Note: For a boat for which there are no reservations, the attributes of
         r have the value NULL, e.g., r.sid. The function COUNT takes only
         attribute values into account that are distinct from NULL.


SELECT b.id as bid, b.name as bname, COUNT(*) as numberOfRes
FROM   boat b JOIN reservation r ON b.id = r.bid
GROUP BY b.id, b.name
```

```
UNION
SELECT b.id, b.name, 0
FROM   boat b
WHERE  NOT EXISTS
       (SELECT *
        FROM   reservation r0
        WHERE  r0.bid = b.id)
```

- 3. List those sailors who have reserved every red boat (list the id and the name).

```
SELECT s.id, s.name
FROM   sailor s
WHERE  NOT EXISTS
       (SELECT *
        FROM   boat b
        WHERE  b.colour = 'red' AND
               NOT EXISTS
               (SELECT *
                FROM   reservation r
                WHERE  r.sid = s.id AND
                       r.bid = b.id))

-- A sailor has made reservations for every red boat if there is no
-- red boat for which he/she has not made reservation


SELECT s.id, s.name
FROM   sailor s
WHERE  (SELECT COUNT(DISTINCT b.id)
        FROM   reservation r JOIN boat b ON r.bid=b.id
        WHERE  r.sid=s.id AND
               b.color = 'red')
       =
       (SELECT COUNT(b.id)
        FROM   boat b
        WHERE  b.color = 'red')
```

- 4. List those sailors who have reserved only red boats.

```
SELECT *
FROM   sailor S
WHERE  NOT EXISTS
       (SELECT *
        FROM   reservation r JOIN boat b ON r.bid=b.id
        WHERE  b.colour <> 'red' AND
               r.sid=s.id)

-- We are looking for sailors for whom there does not exist a
-- reservation for a non-red boat.

SELECT DISTINCT *
FROM   sailor s
WHERE  'red'= ALL (SELECT b.colour
                   FROM   reservation r JOIN boat b ON r.bid=b.id
                   WHERE  r.sid = s.id)

-- We are looking for sailors such that 'red' is the only colour that shows
-- up in their boat reservations
```

- 5. For which boat are there the most reservations?

```
SELECT b.id, b.name, COUNT(*)
FROM   boat b JOIN reservation r ON b.id=r.bid
GROUP BY b.id, b.name
HAVING COUNT(*) >= ALL
       (SELECT COUNT(*)
        FROM   reservation r0
        GROUP BY r0.bid)
```

```
-- Alternatively, we can support the query by a view

CREATE VIEW resPerBoat(name,id,noRes) AS
SELECT b.name, b.id, COUNT(*)
FROM   boat b JOIN reservation r on b.id=r.bid
GROUP BY b.name, b.id

SELECT *
FROM   resPerBoat rpb
WHERE  rpb.noRes = (SELECT MAX(noRes)
                    FROM   resPerBoat rpb0)



 - 6. Delete all sailors who have never reserved a red boat.

DELETE
FROM   sailor s
WHERE  NOT EXISTS
       (SELECT *
        FROM   reservation r JOIN boat b ON r.bid=b.id
        WHERE  b.colour='red' AND
               r.sid=s.id)

DELETE
FROM   sailor s
WHERE  s.id NOT IN
       (SELECT r.sid
        FROM   reservation r JOIN boat b ON r.bid=b.id
        WHERE  b.colour='red')


 - 7. For each sailor, who has at least ten reservations for red boats,
   increase the ranking by 1.

UPDATE sailor
SET    ranking = ranking + 1
WHERE  10 <= (SELECT COUNT(*)
              FROM   reservation r JOIN boat b ON r.bid=b.id
              WHERE  b.colour='red' AND
                     r.sid=b.id)



 - 8. Create a table LastReservation(sid,sname,day,bid, bname) that
   should contain for every sailor the most recent date when the sailor has
   made a reservation and the boat he/she has reserved. Write an insert
   statement that fills the table with the information that can be inferred
   from the current state of the database.


CREATE TABLE LastReservation
    (sid   INT,
     sname VARCHAR(20),
     day   DATE,
     bid   INT,
     bname VARCHAR(20),
     PRIMARY KEY (sid, bid),
     FOREIGN KEY (sid)
         REFERENCES sailor(id),
     FOREIGN KEY (bid)
     REFERENCES boat(id));


INSERT INTO LastReservation
SELECT s.sid, s.name, r.day, b.bid, b.name
FROM   sailor s JOIN reservation r ON s.id=r.sid AND
                JOIN boat b ON b.id=r.bid
WHERE  r.day = (SELECT MAX(r0.day)
                FROM   reservation r0
                WHERE  r0.sid = s.id)
```

- 9. Find the sailors who have made, for each boat colour, at least one reservation for a boat with that colour.

```
SELECT *
FROM   sailor s
WHERE  NOT EXISTS
       (SELECT *
        FROM   boat bc
        WHERE  NOT EXISTS
               (SELECT *
                FROM   reservation r JOIN boat b ON r.bid=b.id
                WHERE  r.sid=s.id AND
                       b.colour=bc.colour))
```

VIEWS TO SUPPORT QUERIES

- 1. Which is the department with the lowest maximal salary?

```
CREATE VIEW maxsal(deptno, dname, maxsalary) AS
SELECT d.deptno, d.dname, MAX (e.sal)
FROM   emp e NATURAL JOIN dept d
GROUP BY d.deptno, d.dname

SELECT ms.deptno, ms.dename, ms.maxsalary
FROM   maxsal ms
WHERE  ms.maxsalary =
          (SELECT MIN(maxsalary)
           FROM   maxsal)
```

- 2. For each job type, find the department that has the highest number of employees with that type of job.

```
CREATE VIEW deptjobs(deptno, job, jcount) AS
SELECT e.deptno, e.job, COUNT(*)
FROM   emp e
GROUP BY e.deptno, e.job

SELECT dj.job, dj.deptno, dj.jcount
FROM   deptjobs dj
WHERE  dj.jcount >= ALL
          (SELECT dj0.jcount
           FROM   deptjobs dj0
           WHERE  dj0.job = dj.job)
```