Short Question

Question 1. [10 marks]

Define $R \otimes S$ (R XOR S) as the relation which contains tuples that belong to either R or S but to both R and S. Assume that both R and S have the same schema, (A, B) and they are "XOR compatible", wherever applicable:

(a) Can you express $R \otimes S$ in terms of the basic relational operations? If yes, then show your expressions. If no, explain why and what is missing.

Solution:

$$R \otimes S = R \cup S - R \cap S$$

(b) Can you express $R \otimes S$ in SQL? If yes, then show your SQL. If no, explain why and what is missing.

Solution:

```
(select * from R
where (R.A, R.B) not in (select * from S))
UNION
(select * from S
where (S.A, S.B) not in (select * from R))
```

Schema for Relational Algebra
and SQL queries:

Consider following schema for *sailors-boats-reserves.*
The database stores the information about sailors who have reserved boats.
The database has following three relations with primary keys underlined and attribute domain listed besides the attribute name:

sailors (sid: integer, sname: string, rating: real, age: real)
boats (bid: integer, bname: string, color: string)
reserves (sid: integer, bid: string, day: date)

Relational Algebra

Answer the following queries in Relational Algebra. If the query cannot be expressed in Relational Algebra, briefly explain the reason for it.

Question 2. [5 marks] Find the id of the sailors whose rating is better than some sailor called 'Bob'.

Solution:

$$\pi_{s2.sid}(\sigma_{s2.rating>sailors.rating}(\rho_{s2}(sailors) \times \sigma_{sname='Bob'}(sailors)))$$

Question 3. [5 marks] Find the names of the sailors who have reserved all boats called 'BigBoat'.

Solution:

$$\pi_{sname}((\pi_{sid,bid}reserves \,/\, \pi_{bid}boats) \bowtie sailors)$$

Question 4. [3 marks] Find the ids of the sailors who have reserved the highest number of green boats.

This cannot be expressed in relational algebra because there is no operator to count, and this query requires the ability to count up to a number that depends on the data.

Question 5. [7 marks] Find the names of the sailors
who have reserved every boat reserved by those with a lower rating.

Solution:

$$sailorsAndLesserSailors := \pi_{sailors.sid,s2.sid}\left(\sigma_{sailors.rating>s2.rating}(\rho_{s2}(sailors) \times sailors)\right)$$

$$shouldHaveReserved := \pi_{sailors.sid,bid}(sailorsAndLesserSailors \bowtie_{s2.sid=r.sid} reserves)$$

$$witnessOfDisqualification := shouldHaveReserved - \pi_{sid,bid} reserves$$

$$answer := \pi_{s2.sname}((\pi_{sid} sailors - \pi_{sid} witnessofDisqualification) \bowtie \rho_{s2}(sailors))$$

Question 6. [5 marks] Find the name and age of the oldest sailor.

Solution:

$$\pi_{sname,age}(\left(\pi_{sid} sailors - \pi_{s2.sid}\left(\sigma_{s2.age<s.age}(\rho_{s2} sailors \times sailors)\right)\right) \bowtie sailors)$$

SQL

Answer the following queries in SQL.

Question 7. [3 marks] Find the names of sailors who have reserved a red boat, and list in the descending order of age.

Solution:

```
SELECT S.sname, S.age
FROM Sailors S, Reserves R, Boats B
WHERE S.sid = R.sid AND R.bid = B.bid AND B.color = 'red'
ORDER BY S.age DESC
```

Question 8. [5 marks] Find the names of sailors who have reserved all the boats.

Solution:

```
SELECT S.sname
FROM Sailors S
WHERE NOT EXISTS (
   SELECT B.bid
   FROM Boats B
   WHERE NOT EXISTS (
      SELECT R.bid
      FROM Reserves R
      WHERE R.sid = S.sid AND R.bid = B.bid
   )
);
```

Question 9. [5 marks] For each boat which was reserved by at least 5 distinct sailors, find the boat id and the average age of sailors who reserved it. **For this question only, Do Not use subquery**.

Solution:

```
SELECT  b.bid, AVG(age)
FROM    Sailors s, Reserves r, Boats b
WHERE  b.bid=r.bid AND r.sid=s.sid
GROUP BY bid
HAVING 5<=COUNT(DISTINCT r.sid)
```

Question 10. [7 marks] Create a view named `RatingDiffrence`. In this view, store the difference between the average rating of sailors who reserved red boat and the average rating of all sailors (including ones who reserved red boat). Using this view, find the ***name and rating of all the sailors*** who have ***not*** reserved a boat whose name includes string “thunder” and whose rating is more than `RatingDiffrence`.

Solution:

```
CREATE VIEW RatingDifference (diff) AS
SELECT TEMP1.avg - TEMP2.avg
FROM (SELECT avg(S1.rating) AS avg
        FROM Sailors S1, Boats B1, Reserves R1
        WHERE S1.sid = R1.sid AND R1.bid = B1.bid AND B1.color = 'red')
        AS TEMP1,
         (SELECT avg(S2.rating) AS avg
        FROM Sailors S2) AS TEMP2;

SELECT S1.sname, S1.rating
FROM Sailors S1, RatingDifference
WHERE S1.sid NOT IN(SELECT DISTINCT s.sid
                        FROM Reserves r, Sailors s, Boats b
                          WHERE r.bid = b.bid AND r.sid = s.sid AND b.bname LIKE '%storm%' ) AND
S1.rating > RatingDifference.diff;
```