CS143: Database Systems
Homework #1 SOLUTION

1. $(R - S) \cup (S - R)$ is:

A	B	C
1	2	6
2	5	4
4	5	6

2. $R \bowtie_{R.A<S.C \wedge R.B<S.D} S$ is:

A	R.B	S.B	C	D
1	2	2	4	6
3	4	2	4	6
1	2	8	6	8
3	4	8	6	8
5	6	8	6	8
1	2	7	5	9
3	4	7	5	9

3. (a)

$$\pi_{customer-name}(\sigma_{branch-name='Region12'}(Account))$$

(b)

$$\pi_{customer-name}(\sigma_{A.city<>B.city \wedge A.branch-name=B.branch-name}(\rho_B(Branch) \times \rho_A(Customer \bowtie Account)))$$

(c)

$$\pi_{branch-name}(Branch) - \pi_{branch-name}(Account)$$

(d)

$$\pi_{customer-name}(Customer) - \pi_{customer-name}(\sigma_{branch-name='Region12'}(Account))$$

(e)

$$\pi_{customer-name}(Customer) - \pi_{customer-name}(\pi_{customer-name}(Customer) \times \pi_{branch-name}(\sigma_{city='LosAngeles'}(Branch)) - \pi_{customer-name,branch-name}(Account))$$

(f)

$$\pi_{customer-name}(Customer) - \pi_{A.customer-name}(\sigma_{A.branch-name<>B.branch-name \vee A.account-number<>B.account-number) \wedge A.customer-name=B.customer-name}(\rho_A(Account) \times \rho_B(Account)))$$

4. $\pi_{sid}(Student) - \pi_{A.sid}(\sigma_{A.GPA>B.GPA \wedge A.sid<>B.sid}(\rho_A(Student) \times \rho_B(Student)))$

5. Write the queries of Exercises 3. and 4. in SQL .

[SQL] 3. Here's the tables again for reference

Customer(customer-name, street, city)
Branch(branch-name, city)
Account(customer-name, branch-name, account-number)

(a) Find the names of all customers who have an account in the 'Region12' branch.

```
SELECT DISTINCT customer-name
FROM Account
WHERE branch-name = 'Region12'
```

Distinct optional since question didn't specify unique.

(b) Find the names of all customers who have an account in a branch NOT located in the same city that they live in.

Two example solutions, one using implicit cross join and one using explicit JOIN.

Implicit cross join:

```
SELECT DISTINCT A.customer-name
FROM Account A,
    Branch B,
    Customer C
WHERE A.customer-name = C.customer-name
  AND A.branch-name = B.branch-name
  AND B.city <> C.city
```

Explicit JOIN:

```
SELECT DISTINCT A.customer-name
FROM Account A
JOIN Branch B ON A.branch-name = B.branch-name
JOIN Customer C ON A.customer-name = C.customer-name
WHERE B.city <> C.city
```

Distinct again optional.

(c) Find the branches that do not have any accounts.

Can be written using either EXCEPT or NOT IN:

NOT IN	EXCEPT
SELECT DISTINCT branch-name FROM Branch WHERE branch-name NOT IN (SELECT branch-name FROM Account)	SELECT branch-name FROM Branch EXCEPT SELECT branch-name FROM Account

Note: **DISTINCT** is required for NOT IN to return the same results as EXCEPT. The question didn't explicitly require it though.

(d) Find the customer names who do not have any account in the 'Region12' branch.

EXCEPT	NOT IN
SELECT customer-name FROM Branch EXCEPT SELECT customer-name FROM Account WHERE branch-name = 'Region12'	SELECT DISTINCT branch-name FROM Branch WHERE branch-name NOT IN (SELECT customer-name FROM Account WHERE branch-name = 'Region12')

Note that the underlined portions ('subquery') are identical and come from 3(a).

(e) Find the customer names who have accounts in all the branches located in 'Los Angeles'. You are not allowed to use the division operator directly for this question.

Idea: Cross product Branch + Customer and find combinations that don't exist (for LA), then take the set difference.

EXCEPT	NOT IN
SELECT customer-name FROM Customer EXCEPT SELECT customer-name FROM FROM Branch B, Customer C WHERE B.city = 'Los Angeles' AND (C.customer-name, B.branch-name) NOT IN (SELECT customer-name, branch-name FROM Account)	SELECT DISTINCT customer-name FROM Customer WHERE customer-name NOT IN (SELECT customer-name FROM FROM Branch B, Customer C WHERE B.city = 'Los Angeles' AND (C.customer-name, B.branch-name) NOT IN (SELECT customer-name, branch-name FROM Account))

Note: EXCEPT cannot be used in the inner query, as it only applies at the top level.

Aggregates weren't covered for hw1, but this query can be solved using count distinct:

```
SELECT customer-name
FROM Account AS A,
    Branch AS B
WHERE A.branch-name=B.branch-name
  AND B.city = 'Los Angeles'
GROUP BY customer-name
HAVING count(DISTINCT B.branch-name) =
  (SELECT count(DISTINCT branch-name)
   FROM Branch
   WHERE city='Los Angeles')
```

(f) Find the customer names who have only one account.

Using EXCEPT:

```
SELECT customer-name
FROM Customer
EXCEPT
SELECT A.customer-name
FROM Account A,
    Account B
WHERE (A.branch-name <> B.branch-name
      OR A.account-number <> B.account-number)
 AND A.customer-name = B.customer-name
```

Using NOT IN:

```
SELECT DISTINCT customer-name
FROM Customer
WHERE customer-name NOT IN
   (SELECT A.customer-name
    FROM Account A,
        Account B
    WHERE (A.branch-name <> B.branch-name
          OR A.account-number <> B.account-number)
     AND A.customer-name = B.customer-name)
```

Again, this can also be solved with aggregates:

```
SELECT customer-name
FROM Account
GROUP BY customer-name
HAVING count(DISTINCT account-number)=1
```

*[SQL] 4. The relation **Student(sid, GPA)** captures the student-GPA information, where **sid** is the id of a student and GPA is the student's GPA. Write a relational algebra that finds the ids of the students with the lowest GPA.*
(Hint: When a query is difficult to write, think of its complement.)

EXCEPT	**NOT IN**
SELECT sid FROM Student EXCEPT SELECT A.sid FROM Student A, Student B WHERE A.GPA > B.GPA AND A.sid <> B.sid	SELECT DISTINCT sid FROM Student WHERE sid NOT IN (SELECT A.sid FROM Student A, Student B WHERE A.GPA > B.GPA AND A.sid <> B.sid)

This is much easier with aggregates.

```
SELECT sid
FROM Student
WHERE GPA = (SELECT MIN(GPA)  FROM Student )
```