Jason Lai
204995126
Dis 1B

CS 143
January 25, 2021

Homework 2

1 Problem 1

a) `SELECT company-name FROM Work EXCEPT SELECT company-name FROM Work WHERE salary <= 150000;`

b) $(\pi_{company-name}(Work)) - (\pi_{company-name}(\sigma_{salary<=150000}(Work)))$

c) I believe the results of (a) and (b) are not the same, because the SQL query will return duplicate company names, whereas the relational algebra query will return distinct company names. This is because in relational algebra, the PROJECT operator will remove duplicates as a result of set semantics, whereas the SELECT operator in SQL will not do that.

Work:

a)
First `SELECT` the names of companies with at least one employee that has a salary less than or equal to $150,000:

```
SELECT company-name FROM Work WHERE salary <= 150000;
```

Now, we SELECT all company names not in the result we got above:

```
SELECT company-name FROM Work EXCEPT SELECT company-name FROM Work WHERE salary <= 150000;
```

b)
Use the $Work$ relation:

$Work$

Select the tuples from $Work$ where the salaries are less than $150,000$

$\sigma_{salary<=150000}(Work)$

Now we filter down to just company names:

$\pi_{company-name}(\sigma_{salary<=150000}(Work))$

Now use the $Work$ relation again:

$Work$

Filter down to just the company names:

$\pi_{company-name}(Work)$

Now take all the names that are in $Company$ but not the result from $Work$

$(\pi_{company-name}(Work)) - (\pi_{company-name}(\sigma_{salary<=150000}(Work)))$

2 Problem 2

a)

1st Way:

```
SELECT person-name FROM Work GROUP BY person-name HAVING SUM(salary) > ALL (SELECT
SUM(salary) FROM Work WHERE person-name IN (SELECT person-name FROM Employee WHERE
city = 'Los Angeles') GROUP BY person-name);
```

2nd Way:

```
SELECT person-name FROM Work EXCEPT SELECT person-name FROM Work GROUP BY person-name
HAVING SUM(salary) <= SOME (SELECT SUM(salary) FROM Work WHERE person-name IN
(SELECT person-name FROM Employee WHERE city = 'Los Angeles') GROUP BY person-name);
```

b)

1st Way:

```
SELECT DISTINCT MT.MN FROM
      (SELECT MT.MN, MT.MS, ET.ES FROM
           (SELECT M1.manager-name AS MN, SUM(W1.salary) AS MS FROM Work W1, Manage
M1 WHERE M1.manager-name = W1.person-name GROUP BY W1.person-name) AS MT,
           (SELECT M2.manager-name AS EMN, SUM(W2.salary) AS ES FROM Work M2, Manage
M2 WHERE M2.person-name = W2.person-name GROUP BY W2.person-name) AS ET
           WHERE MT.MN = ET.EMN)
      WHERE MT.MS > ET.ES;
```

2nd Way:

```
SELECT manager-name AS MN FROM Work GROUP BY person-name HAVING SUM(salary) >
SOME (SELECT SUM(salary) FROM Work INNER JOIN Manage ON Manage.person-name =
Work.person-name WHERE MN = Manage.manager-name GROUP BY Work.person-name);
```

Work:

a)
1st Way:
First `SELECT` the names of the employees that live in Los Angeles:

```
SELECT person-name FROM Employee WHERE city = 'Los Angeles';
```

Now get the total salaries of these people:

```
SELECT SUM(salary) FROM Work WHERE person-name IN (SELECT person-name FROM
Employee WHERE city = 'Los Angeles') GROUP BY person-name;
```

Now get the name of employees who have total salaries greater than those of the people in the result above:

```
SELECT person-name FROM Work GROUP BY person-name HAVING SUM(salary) > ALL
```

```
(SELECT SUM(salary) FROM Work WHERE person-name IN (SELECT person-name FROM Employee
WHERE city = 'Los Angeles') GROUP BY person-name);
```

2nd Way:
First `SELECT` the names of the employees that live in Los Angeles:

```
SELECT person-name FROM Employee WHERE city = 'Los Angeles';
```

Now get the total salaries of these people:

```
SELECT SUM(salary) FROM Work WHERE person-name IN (SELECT person-name FROM
Employee WHERE city = 'Los Angeles') GROUP BY person-name;
```

Now get the name of employees who have total salaries less than or equal to those of the people in the result above:

```
SELECT person-name FROM Work GROUP BY person-name HAVING SUM(salary) <=
SOME (SELECT SUM(salary) FROM Work WHERE person-name IN (SELECT person-name FROM
Employee WHERE city = 'Los Angeles') GROUP BY person-name);
```

Finally get the difference:

```
SELECT person-name FROM Work EXCEPT SELECT person-name FROM Work GROUP BY
person-name HAVING SUM(salary) <= SOME (SELECT SUM(salary) FROM Work WHERE person-name
IN (SELECT person-name FROM Employee WHERE city = 'Los Angeles') GROUP BY person-name);
```

b)
1st Way:
Use the `Work` relation:

```
Work;
```

Select the total salaries and names of all managers:

```
SELECT M1.manager-name AS MN, SUM(W1.salary) AS MS FROM Work W1, Manage
M1 WHERE M1.manager-name = W1.person-name GROUP BY W1.person-name;
```

Now select the total salaries, names, and manager's names of all employees:

```
SELECT M2.manager-name AS EMN, W2.person-name AS EN, SUM(W2.salary) AS ES
FROM Work M2, Manage M2 WHERE M2.person-name = W2.person-name GROUP BY W1.person-name;
```

Combine the two subqueries into one query:

```
SELECT MT.MN, MT.MS, ET.ES FROM (SELECT M1.manager-name AS MN, SUM(W1.salary)
AS MS FROM Work W1, Manage M1 WHERE M1.manager-name = W1.person-name GROUP BY
W1.person-name) AS MT, (SELECT M2.manager-name AS EMN, SUM(W2.salary) AS ES FROM
Work M2, Manage M2 WHERE M2.person-name = W2.person-name GROUP BY W1.person-name)
AS ET WHERE MT.MN = ET.EMN;
```

Select the names of managers with higher salaries than their employees:

```
SELECT DISTINCT MT.MN FROM
     (SELECT MT.MN, MT.MS, ET.ES FROM
          (SELECT M1.manager-name AS MN, SUM(W1.salary) AS MS FROM Work W1,
```

```
Manage M1 WHERE M1.manager-name = W1.person-name GROUP BY W1.person-name) AS MT,
               (SELECT M2.manager-name AS EMN, SUM(W2.salary) AS ES FROM Work
M2, Manage M2 WHERE M2.person-name = W2.person-name GROUP BY W2.person-name) AS
ET
               WHERE MT.MN = ET.EMN)
          WHERE MT.MS > ET.ES;
```

2nd Way:
Use the `Work` relation:

```
Work;
```

Select the total salaries and names of all employees with a manager's name after join with `Manage` relation:

```
SELECT SUM(salary) FROM Work INNER JOIN Manage ON Manage.person-name =
Work.person-name WHERE MN = Manage.manager-name GROUP BY Work.person-name;
```

Now select the manager names who have a total salary greater than at least one person they manage:

```
SELECT manager-name AS MN FROM Work GROUP BY person-name HAVING SUM(salary)
> SOME (SELECT SUM(salary) FROM Work INNER JOIN Manage ON Manage.person-name =
Work.person-name WHERE MN = Manage.manager-name GROUP BY Work.person-name);
```

3 Problem 3

a)
i.

```
SELECT name, address FROM MovieStar WHERE gender = 'F' INTERSECT SELECT name,
address FROM MovieExec WHERE netWorth > 1000000;
```

ii.

```
SELECT name, address FROM MovieExec WHERE netWorth > 1000000 AND name IN (SELECT
name FROM MovieStar WHERE gender = 'F');
```

b)
i.

```
SELECT * FROM MovieStar WHERE name IN (SELECT name FROM MovieStar EXCEPT SELECT
name FROM MovieExec);
```

ii.

```
SELECT * FROM MovieStar WHERE name NOT IN (SELECT name FROM MovieExec);
```

Work:

a)
i.
Use the `MovieStar` relation:

```
MovieStar;
```

Select the names and addresses of female movie stars:

```
SELECT name, address FROM MovieStar WHERE gender = 'F';
```

Use the `MovieExec` relation:

```
MovieExec;
```

Select the names and address of movie execs with networths $>$ 1000000:

```
SELECT name, address FROM MovieExec WHERE netWorth > 1000000;
```

Intersect the two results:

```
SELECT name, address FROM MovieStar WHERE gender = 'F' INTERSECT SELECT
name, address FROM MovieExec WHERE netWorth > 1000000;
```

ii.
Use the `MovieStar` relation:

```
MovieStar;
```

Select the names of female movie stars:

```
SELECT name FROM MovieStar WHERE gender = 'F';
```

Use the `MovieExec` relation:

```
MovieExec;
```

Select the names of movie execs that are also female movie stars, and have a networth > 1000000:

```
SELECT name, address FROM MovieExec WHERE netWorth > 1000000 AND name IN (SELECT name FROM MovieStar WHERE gender = 'F');
```

b)
i.
Use the `MovieExec` relation:

```
MovieExec;
```

Select the names of movie execs:

```
SELECT name FROM MovieExec;
```

Get the names of movie stars that are not movie execs:

```
SELECT name FROM MovieStar EXCEPT SELECT name FROM MovieExec;
```

Get the full tuples of movie stars that are not movie execs:

```
SELECT * FROM MovieStar WHERE name IN (SELECT name FROM MovieStar EXCEPT SELECT name FROM MovieExec);
```

ii.
Use the `MovieExec` relation:

```
MovieExec;
```

Select the names of movie execs:

```
SELECT name FROM MovieExec;
```

Get the tuples of movie stars that are not movie execs:

```
SELECT * FROM MovieStar WHERE name NOT IN (SELECT name FROM MovieExec);
```

4 Problem 4

a) `SELECT AVG(speed) FROM Desktop;`

b) `SELECT AVG(price) FROM ComputerProduct WHERE manufacturer = 'Dell';`

c) `SELECT AVG(price) FROM ComputerProduct WHERE model IN (SELECT model FROM Laptop WHERE weight > 3);`

d) `SELECT AVG(ComputerProduct.price) FROM ComputerProduct INNER JOIN Laptop ON ComputerProduct.model = Laptop.model GROUP BY Laptop.speed;`

e) `SELECT manufacturer FROM ComputerProduct GROUP BY manufacturer HAVING COUNT(model) >= 3;`

Work:

a)
Use the `Desktop` relation:

```
Desktop;
```

Select the average speed of desktop computers:

```
SELECT AVG(speed) FROM Desktop;
```

b)
Use the `ComputerProduct` relation:

```
ComputerProduct;
```

Select the average price of the models produced by Dell:

```
SELECT AVG(price) FROM ComputerProduct WHERE manufacturer = 'Dell';
```

c)
Use the `Laptop` relation:

```
Laptop;
```

Select the models of laptops with weight greater than 3kg:

```
SELECT model FROM Laptop WHERE weight > 3;
```

Select the average price of models in the ComputerProduct relation that are in the result above:

```
SELECT AVG(price) FROM ComputerProduct WHERE model IN (SELECT model FROM Laptop WHERE weight > 3);
```

d)
Select the average price of laptops grouped by speed after a left join:

```
SELECT AVG(ComputerProduct.price) FROM ComputerProduct INNER JOIN Laptop
ON ComputerProduct.model = Laptop.model GROUP BY Laptop.speed;
```

e)
Use the `ComputerProduct` relation:

```
ComputerProduct;
```

Select the manufacturers that make at least three different computer models:

```
SELECT manufacturer FROM ComputerProduct GROUP BY manufacturer HAVING COUNT(model)
>= 3;
```

5 Problem 5

a)

```
INSERT INTO ComputerProduct VALUES ('HP', 1100, 1000);
INSERT INTO Desktop VALUES (1100, 1.2, 256, 40);
```

b)

```
DELETE FROM Desktop WHERE model IN (SELECT model FROM ComputerProduct WHERE
manufacturer = 'IBM' AND price < 1000);
DELETE FROM ComputerProduct WHERE manufacturer = 'IBM' AND price < 1000 AND model
NOT IN (SELECT model FROM Laptop);
```

c)

```
UPDATE Laptop SET hdd = hdd - 1 WHERE model IN (SELECT model FROM ComputerProduct
WHERE manufacturer = 'Gateway');
```

Work:

a)
Insert into `ComputerProduct` relation:

```
INSERT INTO ComputerProduct VALUES ('HP', 1100, 1000);
```

Insert into `Desktop` relation:

```
INSERT INTO Desktop VALUES (1100, 1.2, 256, 40);
```

b)
First select the models of IBM desktops that are priced below $1000:

```
SELECT model FROM ComputerProduct WHERE manufacturer = 'IBM' AND price <
1000;
```

Delete all rows from `Desktop` that are models in the result above:

```
DELETE FROM Desktop WHERE model IN (SELECT model FROM ComputerProduct WHERE
manufacturer = 'IBM' AND price < 1000);
```

Then get the models of laptops:

```
SELECT model FROM Laptop;
```

Delete all rows from `ComputerProduct` that are not in the result above:

```
DELETE FROM ComputerProduct WHERE manufacturer = 'IBM' AND price < 1000
AND model NOT IN (SELECT model FROM Laptop);
```

c)
First select the models of Gateway computers:

```
SELECT model FROM ComputerProduct WHERE manufacturer = 'Gateway';
```

Update the Gateway laptops:

```
UPDATE Laptop SET hdd = hdd - 1 WHERE model IN (SELECT model FROM ComputerProduct
WHERE manufacturer = 'Gateway');
```