CS143: Homework #2 (SQL)

1. Assume the following tables for this problem:

```
Employee(person-name, age, street, city)
Work(person-name, company-name, salary)
Company(company-name, city)
Manage(person-name, manager-name)
```

   A person's name is unique, but a person may work for more than one company. A company name is unique, but a company may be located in more than one city.

   (a) Write a query in SQL to find the names of such companies that all of their employees have salaries higher than $150,000.
   ANSWER:
   `SELECT company-name FROM Company C WHERE 150000 < ALL (SELECT salary FROM Work W2 WHERE C.company-name = W2.company-name)`

   (b) Write the same query in Relational Algebra.
   ANSWER:
   $\Pi_{company-name}(Company) - \Pi_{company-name}(\sigma_{salary \leq 150000}(Work))$

   (c) Compare the results of (a) and (b), are they the same? Why?
   ANSWER:
   `No, because there might be duplicates in the result of the SQL query.`

2. Assume the database of the previous problem and write the following queries in SQL. You should use at least one subquery in each of your answers and write each query in two significantly different ways (e.g., using different operators such as EXISTS, IN, and SOME)

   (a) Find the name(s) of the employee(s) whose *total* salary is higher than those of all employees living in Los Angeles.
   ANSWER:

```
SELECT person-name
FROM   Work
GROUP BY person-name
HAVING SUM(salary) >ALL
     (SELECT SUM(salary)
      FROM Work W, Employee E
      WHERE W.person-name=E.person-name AND city='Los Angeles'
      GROUP BY W.person-name)

SELECT person-name
FROM (SELECT person-name, SUM(salary) total-salary
      FROM Work
      GROUP BY person-name) TotalSalary
WHERE NOT EXISTS(
            SELECT W.person-name
```

```
                FROM Employee E, Work W
                WHERE W.person-name=E.person-name AND city='Los Angeles'
                GROUP BY W.person-name
                HAVING SUM(salary) >= TotalSalary.total-salary
              )
```

(b) Find the name(s) of the manager(s) whose *total* salary is higher than that of at least one employee that they manage.

ANSWER:

```
SELECT person-name
FROM   Work W1
GROUP BY person-name
HAVING SUM(salary) >SOME
     (SELECT SUM(salary)
      FROM Work W2, Manage M
      WHERE W1.person-name=M.manager-name AND M.person-name=W2.person-name
      GROUP BY W2.person-name)

SELECT person-name
FROM (SELECT person-name, SUM(salary) total-salary
      FROM Work
      GROUP BY person-name) TotalSalary
WHERE EXISTS(
        SELECT W.person-name
        FROM Work W, Manage M
        WHERE TotalSalary.person-name=M.manager-name AND M.person-name=W.person-name
        GROUP BY W.person-name
        HAVING SUM(salary) < TotalSalary.total-salary
      )
```

3. Assume the following tables for this problem:

   `MovieStar(name, address, gender)`
   `MovieExec(name, address, company, netWorth)`

   Assume that everyone's name is unique and if a person is both a movie star and executive, the person's address is the same in the two tables.

   (a) We want to find the names and addresses of all female movie stars (`gender = 'F'` in the `MovieStar` relation) who are also movie executives with a net worth over $1,000,000 (`netWorth > 1000000` in the `MovieExec` relation).

   i. Write the query using `INTERSECT` operator.
   ANSWER:
   `(SELECT name, address FROM MovieStar WHERE gender='F') INTERSECT (SELECT name, address FROM MovieExec WHERE netWorth>1000000)`

   ii. Write the query without using `INTERSECT` operator.
   ANSWER:
   `SELECT name, address FROM MovieStar WHERE gender='F' AND (name, address) IN (SELECT name, address FROM MovieExec WHERE netWorth>1000000)`

   (b) We want to find the movie stars who are not movie executives.

   i. Write the query using `EXCEPT` operator.
   ANSWER:
   `(SELECT name FROM MovieStar) EXCEPT (SELECT name FROM MovieExec)`

ii. Write the query without using `EXCEPT` operator.
ANSWER:
```
SELECT name FROM MovieStar WHERE name NOT IN (SELECT name FROM MovieExec)
```

4. Assume the following tables for this problem:

```
ComputerProduct(manufacturer, model, price)
Desktop(model, speed, ram, hdd)
Laptop(model, speed, ram, hdd, weight)
```

A computer product is either a desktop or a laptop.

(a) Find the average speed of all desktop computers.
ANSWER:
```
SELECT AVG(speed) FROM Desktop
```

(b) Find the average price of PC's and laptops made by "Dell."
ANSWER:
```
SELECT AVG(price) FROM ComputerProduct WHERE manufacturer=‘DELL’
```

(c) Find the average price of all laptops with weight above 3kg.
ANSWER:
```
SELECT AVG(price) FROM ComputerProduct CP, Laptop L WHERE CP.model=L.model AND weight>=3
```

(d) For each different CPU speed, find the average price of a laptop.
ANSWER:
```
SELECT AVG(price) FROM Laptop GROUP BY speed
```

(e) Find the manufacturers that make at least three different computer models.
ANSWER:
```
SELECT manufacturer FROM ComputerProduct GROUP BY manufacturer HAVING COUNT(model)>=3
```

5. Assume the computer-product database of the previous problem, and write the following database modifications.

(a) Using two `INSERT` statements, insert a desktop computer manufactured by HP, with model number 1100, price $1000, speed 1.2Ghz, 256MB RAM, and an 40GB hard drive.
ANSWER:
```
INSERT INTO ComputerProduct VALUES (‘HP’, 1100, 1000); INSERT INTO Desktop VALUES
(1100, ‘1.2GHz’, ‘256MB’, ‘40GB’);
```

(b) Using two `DELETE` statements, delete all desktops manufactured by IBM with price below $1000. (*Comments: Be careful with the order of your two* `DELETE` *statements.*)
ANSWER:
```
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) For each laptop made by Gateway, subtract 1GB from the HDD. (*Hint: The* `WHERE` *clause in a* `UPDATE` *statement may contain complex conditions, including subqueries.*)
ANSWER:
```
UPDATE Laptop SET hdd=hdd-1 WHERE model IN (SELECT model FROM ComputerProduct WHERE
manufacturer=‘Gateway’);
```