Chapter 6 homework

Name: Zhang Yupeng

Student ID: 5130309468

Exercise 6.2.2

Write the following queries, based on the database schema

```
Product(maker, model, type)
PC(model, speed, ram, hd, price)
Laptop(model, speed, ram, hd, screen, price)
Printer(model, color, type, price)
```

of Exercise 2.4.1, and evaluate your queries using the data of that exercise.

e) Find those pairs of PC models that have both the same speed and RAM. A pair should be listed only once; e.g., list(i, j) but not (j, i);

Solution:

```
SELECT  P1.model, P2.model
FROM    PC P1, PC P2
WHERE   P1.speed = P2.speed
    AND P1.ram   = P2.ram
    AND P1.model < P2.model ;
```

Exercise 6.3.1

Write the following queries, based on the database schema

```
Product(maker, model, type)
PC(model, speed, ram, hd, price)
Laptop(model, speed, ram, hd, screen, price)
Printer(model, color, type, price)
```

of Exercise 2.4.1. You should use at least one subquery in each of your answers and write each query in two significantly differently ways (e.g., using different sets of the operators EXISTS, IN, ALL, and ANY).

c) Find the laptops whose speed is slower than that of any PC;

Solution:

```
SELECT  L.model
FROM    Laptop L
WHERE   L.speed < ANY(SELECT P.speed FROM PC P);

SELECT  L.model
FROM    Laptop L
WHERE   EXISTS(SELECT P.speed FROM PC
                WHERE P.speed >= L.speed);
```

Exercise 6.4.6

Write the following queries, based on the database schema

```
Product(maker, model, type)
PC(model, speed, ram, hd, price)
Laptop(model, speed, ram, hd, screen, price)
Printer(model, color, type, price)
```

of Exercise 2.4.1, and evaluate your queries using the data of that exercise.

d) Find, for each speed of PC above 2.0, the average price;

Solution:

```
SELECT  speed, AVG(price) AS Avg_Price
FROM    PC
WHERE   speed > 2.0
GROUP BY speed;
```

Exercise 6.6.4

Suppose we have a transaction T that is a function which runs “forever,” and at each hour checks whether there is a PC that has a speed of 3.5 or more and sells for under $1000. If it finds one, it prints the information and terminates. During this time, other transactions that are executions of one of the four programs described in Exercise 6.6.1 may run. For each of the four isolation levels – serializable, repeatable read, read committed, and read uncommitted – tell what the effect on T of running at this isolation level is.

Solution:

Serializable: T will never see changes to the database and keep printing the same list of PCs. This does not serve any useful purpose. Application may need to periodically stop T and then restart it to see data committed in the meantime.

Repeatable Read: T will continue to see the list of PCs it saw once. However, T will also see any new PCs that are inserted in the database. Locking issues can occur if another transaction such as 6.6.1 (b) or (c) tries to update/delete the rows read by T. 6.6.1 (d) inserts a new row and thus can run concurrently with T.

Read Committed: Perhaps the best option. T can see new or updated rows after other transactions such as 6.6.1 (c) or (d) commit. However, if T reads the same table twice, the results are not consistent because some rows may have been updated (6.6.1 (c) or deleted(6.6.1 (b)) by other transaction. Moreover, if T reads a row and based on the result then tries to read/update/delete the row; the state of row may have changed in the meantime.

Read Uncommitted: T will not cause any locking (high concurrency) but uncommitted PC data might be printed out due to insert/update by other transaction e.g. 6.6.1 (c) or (d). However, the other transaction might rollback resulting in wrong reports.