CS143: Database Systems
Homework #3

1. 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) Using a `CHECK` constraint on the `Laptop` table, express the constraint that a laptop cannot have weight larger than 5kg. You do not need to show the entire `CREATE TABLE` statement. Show only the `CHECK` constraint part in the `CREATE TABLE` statement.

   ANSWER:

```
CHECK (weight<=5)
```

2. You are the DBA for the VeryFine Toy Company and create a relation called Employees(ename, dept, salary). For authorization reasons, you also define views EmployeeNames(ename) and DeptInfo(dept, avgsalary). The second column lists the average salary for each department.

   (a) Show the view definition statements for EmployeeNames and DeptInfo.

   ANSWER:

```
CREATE VIEW EmployeeNames AS
SELECT ename FROM Employees

CREATE VIEW DeptInfo AS
SELECT dept, AVG(salary) avgsalary
FROM Employees
GROUP BY dept
```

   (b) You want to authorize your secretary, Mike, to fire people (you will probably tell him whom to fire, but you want to be able to delegate this task), to check on who is an employee, and to check on average department salaries. What is the minimum set of privileges you should grant to Mike?

   ANSWER:

```
SELECT, DELETE on EmployeeNames.  SELECT on DeptInfo.
```

Problem 2 continued.

(c) Continuing with the preceding scenario, you do not want your secretary to be able to look at the salaries of individuals. Does your answer to the previous question ensure this? Be specific: Can your secretary possibly find out salaries of some individuals (depending on the actual set of tuples), or can your secretary always find out the salary of any individual he wants to?

ANSWER:
Yes, if he really wants to, he can get the salary of any employee. For example, to get the salary of John, he first delete all tuples in EmployeeNames except John's and look at the DeptInfo table. Since there is only one employee, John, the avgsalary of the single tuple in DeptInfo is John's salary.

(d) Give an example of a view update on the preceding views that cannot be translated into an update to Employees.

ANSWER:
```
UPDATE DeptInfo SET avgsalary = 10000 WHERE dept='Toy'
```

(e) You decide to go on an extended vacation, and to make sure that emergencies can be handled, you want to authorize your boss Joe to read and modify the Employees relation and the EmployeeNames relation (and Joe must be able to delegate authority, of course, since he is too far up the management hierarchy to actually do any work). Show the appropriate SQL statements. Can Joe read the DeptInfo view?

ANSWER:

```
GRANT SELECT, UPDATE ON Employees TO Joe WITH GRANT OPTION
GRANT SELECT, UPDATE ON EmployeeNames TO Joe WITH GRANT OPTION
No, Joe cannot because we did not give him SELECT privilege on DeptInfo.  Even through
Joe can SELECT on the base table of DeptInfo, it does not give him the SELECT privilege
on DeptInfo.  This should be given explicitly by the owner of DeptInfo.
```

(f) After you come back from your vacation, you realize that Joe has been quite busy. He has defined a view called AllNames using the view EmployeeNames, defined another relation called StaffNames that he has access to (but you cannot access), and given his secretary James the right to read from the AllNames view. James has passed this right on to his friend Susan. You decide that, even at the cost of annoying Joe by revoking some of his privileges, you simply have to take away some of Joe's privileges to prevent James and Susan from seeing your data. What REVOKE statement would you execute? What views remain after you execute this statement?

ANSWER:

```
REVOKE SELECT ON EmployeeNames FROM Joe CASCADE
Since AllNames is dependent on EmployeeNames, and Joe, the owner of AllNames, does
not have the right to read from EmployeeNames, AllNames will be automatically dropped
after the above REVOKE. If we want to prevent Joe from creating another view on Employees
and share it with others, we will have to ‘‘REVOKE SELECT ON Employees FROM Joe CASCADE’’
as well.
```

3. We want to store the table created by the following SQL statement into a disk.

```
CREATE TABLE Class(
  dept  CHAR(2),
  cnum  INTEGER,
  sec   INTEGER,
  unit  INTEGER,
  year  INTEGER,
  quarter INTEGER,
  title CHAR(30),
  instructor CHAR(20)
)
```

We need to store tuples for 1,000 classes that have been offered so far. 10 classes are offered every year. The tuples are stored in random order (i.e., they are not sequenced by any attribute). A disk of the following parameters is used for storing the table.

- 3 platters (6 surfaces)
- 10,000 cylinders
- 500 sectors per track
- 1024 bytes per sector
- 6,000 RPM rotational speed
- 10ms average seek time

(a) What is the capacity of this disk?

ANSWER:
6 surfaces/disk * 10,000 tracks/surface * 500 sectors/track * 1KB/sector = 30GB/disk

(b) What is the average time to read a random sector from the disk?

ANSWER:
(seek time) + (rotational delay) + (transfer time) = 10ms + 5ms + 0.02ms = 15.02ms

(c) Assume one disk block corresponds to one disk sector. How many disk blocks are needed to store the above table with 1,000 tuples?

ANSWER:
72 blocks. $\lfloor \frac{\text{1024 bytes/block}}{\text{1 tuple/72 bytes}} \rfloor = 14$ tuples/block. $\lceil \frac{\text{1000 tuples/table}}{\text{14 tuples/block}} \rceil = 72$ blocks/table.

(d) We want to run the following query by scanning the entire table:

```
SELECT * FROM Class WHERE year = 2005
```

Assuming that all blocks for the table is allocated sequentially, how long will it take to run the query? Assume that the disk head is not on the same track where the first block of the table is stored.

ANSWER:
`(seek time) + (rotational delay) + (transfer time) = 10ms + 5ms + 72*0.02ms = 16.44ms`

(e) Now assume that due to frequent updates to the table, disk blocks are allocated such that, on average, sequentiality is broken every three blocks. That is, the table is stored in 24 randomly located "clusters" of 3 consecutive blocks. Assuming that we scan the entire table to execute the above query, how long will it take?

ANSWER:
`24 * ((seek time) + (rotational delay) + (transfer time)) = 24 * (10ms + 5ms + 3*0.02ms) = 361.44ms`