Databasesystems, Fall 2006

Rasmus Pagh
Esben Rune Hansen

Exercises for the lecture the 28. November 2006

1 Authentication (Exam question worth 15%)

The user alice has just created a relation R(user,info), and issues the following SQL commands:

```
GRANT INSERT ON R TO bob WITH GRANT OPTION;
GRANT SELECT ON R TO bob WITH GRANT OPTION;
GRANT SELECT ON R TO claire;
```

Consider the following SQL commands:

1. SELECT * FROM alice.R WHERE user='claire';
2. INSERT INTO alice.R VALUES ('claire','clairvoyant');
3. GRANT SELECT ON alice.R TO dorothy;

a) State for each of the three users bob, claire, and dorothy, which of the above SQL commands he/she has autorization to execute.

Now assume that bob executes the command

```
   GRANT INSERT ON R TO claire;
```

and alice then executes the command

```
   REVOKE INSERT ON R FROM bob CASCADE;
```

b) Again, state for each of the three users bob, claire, and dorothy, which of the above SQL commands he/she has autorization to execute at this point.

2 More authentication (Exam question worth 15%)

Consider the relation BedBookings from Problem 1. Suppose that it is created by the user dba, who executes the following statements:

```
GRANT SELECT ON BedBookings TO adm WITH GRANT OPTION;
GRANT UPDATE ON BedBookings TO adm WITH GRANT OPTION;
GRANT DELETE ON BedBookings TO adm;
```

Subsequently, the user adm executes these statements (some of which may result in error messages from the DBMS):

```
GRANT SELECT ON BedBookings TO doc;
GRANT UPDATE(from_date,to_date) ON BedBookings TO doc WITH GRANT OPTION;
GRANT DELETE ON BedBookings TO doc;
```

a)State what kinds of rights (SELECT, UPDATE, DELETE) the user doc has on the relation BedBookings. Now assume that the user dba executes the following statements (some of which may result in error messages from the DBMS):

```
REVOKE SELECT ON BedBookings FROM adm CASCADE;
REVOKE UPDATE(from_date) ON BedBookings FROM adm CASCADE;
```

b)State the rights of the user doc after the above REVOKE statements.
The following SQL query returns all tuples in BedBookings concerning female patients, omitting the patient_cpr attribute. (It uses the fact that females have even CPR numbers.)

```
SELECT room_id,bed_number,from_date,to_date
FROM BedBookings
WHERE (patient_cpr%2=0);
```

c)Write SQL statements that, if executed by the user dba, allows the user public to retrieve the information produced by the above query, but does *not* allow public to access any CPR numbers, or any tuples concerning males. **Hint:** First define a view.