Database Systems, fall 2006

Esben Rune Hansen

Exercises for lecture on October 3

Joins

The exercises in this section are considering the relations `Product`, `PC` og `Laptop`, that are already in your database if you have inserted in relations in the file "all.rel" into your database as described on the first exercise sheet (Ex0). If you have not inserted the relations your should do it now.

The relations have the following schemas:

```
Product (id, maker, model, type)
PC (model, speed, ram, hd, rd, price)
Laptop (model, speed, ram, hd, screen, price)
```

Exercise 1

Write queries that return the following:

1. All laptops with a price less than 2500.
2. The modelnumber for all laptops with a price less than 2500 or with a `speed` on at least 800.
3. All tuples i `Product`, that concerns laptops.

The syntax `Product NATURAL JOIN Laptop` can be used to combine tuples in `Product` and `Laptop` with same value in the `model` attribute. Try this in Oracle by typing the following:

```
SELECT * FROM (Product NATURAL JOIN Laptop);
```

Exercise 2

Repeat the exercises from last section but now the related data concerning the data returned has to be included in the result. For instance the first query in Exercise 1 that before only returned information from `Laptop` now has to return information from `Product` as well.

Try now the following query:

```
SELECT * FROM (PC NATURAL JOIN Laptop);
```

Explain why the result is empty. Write then a less restrictive join on the form:

```
SELECT * FROM PC, Laptop where ... = ...
```

that does not give an empty result. Add a tuple to `Laptop`, that makes the query:

```
SELECT * FROM (PC NATURAL JOIN Laptop);
```

no longer return an empty result.

Sub-queries

Exercise 1

Write the following queries, based on the database schema:

```
Classes(class, type, country, numGuns, bore, displacement)
Ships(name, class, launched)
Battles(name, date)
OutComes(ship, battle, result)
```

You should use at least one sub-query in each of your answers and write each query in two significantly different ways (e.g. using different sets of the operator `EXISTS`, `IN`, `ALL` and `ANY`).

1. Find the countries whose ships had the largest number of guns.
2. Find the classes of the ships at least one of which was sunk in a battle
3. Find the names of the ships with a 16-inch bore.
4. Find the battles in which ships of the Kongo class participates

Test your queries in Oracle on the sample data you entered in the beginning of the course.

Exercise 2

Write the following queries, based on the database schema:

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

You should use at least one sub-query in each of your answers and write each query in two significantly different ways (e.g. using different sets of the operator `EXISTS`, `IN`, `ALL` and `ANY`).

1. find the printers with the highest price
2. find the laptops whose speed is slower than that of any PC.

Test your queries in Oracle on the sample data you entered in the beginning of the course.

Constraints & Triggers

Exercise 1

In this exercise you are going to work on the relations

```
Ships(name,class,launched)
Classes(class,type,country,numGuns,bore,displacement)
Outcomes(ship,battle,result)
Battles(name,date)
```

1. Propose appropriate primary keys and foreign key for the 4 relations. Make some changes in the SQL schema definitions, so that primary and foreign key are stated. Use the SET NULL policy in connection with foreign kays. (Oracle only supports SET NULL in connection with deletions, not in connection with updates). Test your new schema definition in Oracle, including insertion of more than one tuple with the same primary key, or with the value NULL on some key attribute. Test also deletion of a tuple that another tuple is referring to.

2. Add tuple base CHECK constraints to the relation schemas, using the syntax `ALTER TABLE ... ADD CONSTRAINT ...`:

   (a) No ship are allowed to have a `bore` (diameter on the canon) larger than 16 inches.

   (b) If a ship has more than 9 canons, its diameter has to be at most 14 inches.

   (c) No ship can be engaged in battle before it is (`launched`).

Exercise 2

3. Write a trigger that makes the following:

   (a) When a new class is inserted in `Classes`, there has to be inserted a ship in `Ships` which name is the name of the class and with NULL as the value of `launched`.

   (b) When a new class is inserted with `displacement` larger than 35.000 tons, the weight has to be changed to 35.000 tons.

   Test your triggers in Oracle. Remember to put semi colons in front of references to new as well as old rowvariables. If you try to modify a realtion in the "action" part of a trigger on the same relation you get a "table is mutation" error since this is not allowed in Oracle. The solution will in many cases be to use a `INSTEAD OF` trigger, where it is allowed to change the relation. If there are compilationerrors in the action part they can be shown by writting `show errors` (but do not trust the linenumbers).

4. Change the policy for foreign keys in exercise 1 to `CASCADE`. Test the new behaviour by deleting some selected tuples.