Introduction to Databases, ITU, Fall 2005

Rasmus Pagh

October 17, 2005

Group project – part 3

This is the third part of the **mandatory** group project. It should be handed in **by e-mail** to the teaching assistant associated with your group, no later than:

Monday November 7, 23.59 PM.

Learning outcomes

The third group hand-in develops competences in three areas mentioned in the goals of the course description. After working on this hand-in you should have improved your ability to:

- program complex queries in SQL
- use [...] relational algebra
- create database constraints such as referential integrity

Observe that the background material on relational algebra and database constraints will be treated in the lectures on October 24 and October 31.

Case description – continued

The hand-in concerns the continued development of the new database of IT@. At this point you should have a final E/R diagram for your system, and have created the corresponding normalized relations in Oracle. If your group has 3 or 4 members, you will also have some data in your tables. In any case, to get more data to work on, you should write a collection of SQL statements that import the data entered by another group. To be able to do this, you should:

- Look at the reports of the groups having 3 or 4 students (available on the internal pages of the course), and choose a group whose data you would like to import. There may be modeling decisions done by this group which mean that there is some data you cannot import. This is OK, but you should try to import as much as possible. Note that you refer to the relation `R` of user `pagh` by `pagh.R`, and so on.
- Contact the target group to let them know that you are importing their data. In this way they can inform you when they have imported new data, so that you even get data entered by a third group, and so on.
- Write and run the SQL statements. Make sure that duplicate tuples are eliminated, so that you can run the statements several times if needed. (To do this you could create relations with schemas identical to the ones you have, and move non-duplicate data from one table to the other using statements of the form `INSERT INTO R1 (SELECT DISTINCT * FROM R2)`.)

To shed light on the future use of the system, the management at IT@ have described a number of "use cases". To see if your database design supports these, without implementing a complete system, you are given the task to program the queries specified below. If you did not make a data model for the whole system, ignore the queries that do not make sense in your data model. Note that a number of the queries are quite challenging. If some query is difficult (or impossible) to write, maybe because of the way you represented data, you should explain the difficulty. You may want to add or change data in order to test your queries properly. Five of your queries should additionally be stated as *relational algebra expressions*.

Queries on the common part for all groups:

1. "How many students do we have of each nationality?"
2. "What is the average total number of ECTS for students in each study line?"
3. "How many persons are registered in each of the different groups of persons?" (this may require several queries).
4. "Who in the faculty lives on the same address as another faculty?"
5. "What was the total number of ECTS for courses in the fall of 2005, specified for each study line?"
6. "Which students have taken a course (s)he did not have the competences for?"
7. "How many students need more competences to get a degree from their study line?"
8. "What students did not pass any course in 2005?"
9. "Who in the faculty are also alumni, and what were their average grade?"
10. "How many of our students live in an area with postal code above 2999?"

Queries on the optional parts:

1. "What is the average grade in courses and projects, respectively, for each student?"
2. "Is it true that grades in projects are higher than grades in courses?"
3. "How many people were employed each year (give a list of years and numbers)?"
4. "What is the size of the different departments?"
5. "How much does each department cost in salary?"
6. "Who was in the various bodies at the beginning of 2004?"
7. "Which body has had the highest number of different members?"
8. "What students enrolled in 2005 are lacking a tutor?"
9. "What tutors had less than three students assigned?"
10. "How much time has been spent in meetings (past or planned)?"

Extra credit: For queries that you have written with a correlated subquery, try to come up with an alternative query that has no correlated subquery.

Finally, you should create database constraints. You may use the syntax `ALTER TABLE R ADD CONSTRAINT ...`. You should at least create the referential integrity and primary key constraints implied by your E/R diagram, but may also add `CHECK` constraints such as checking that the grades entered into the system are valid. If some constraint cannot be created because of it is violated by the data, you should try to correct the data (within reason).

What must be handed in

You must hand in:

- **The SQL statements used for importing**. Specify what group you are importing from.
- **Your SQL queries, and the corresponding query results**. Use the `spool` feature of SQL*Plus specified in group hand-in 2. For queries that you could not complete, you should explain what the difficulty is, and perhaps give some partial result (e.g. a query that seems useful as a subquery, or computes part of the result).
- **Relational algebra expressions corresponding to 5 of your queries**.
- **The SQL statements creating the constraints**.

One the first page, clearly specify the members of your group (if someone dropped the project, don't include him/her). The project should be sent as a **single file in PDF format** to your TA, either Simer Sawhney (simer@aii-tee-yuu.dk) or Priya Seetharaman (itspri@aii-tee-yuu.dk). (Note that the domain name must be changed to itu.dk.)