Databasesystemer, forår 2005
IT Universitetet i København

Forelæsning 12, del 2: Eksamensforberedelse

28. april 2005

Forelæser: Rasmus Pagh

Today's lecture, part II

- What you should be able to do after this course.
- Suggestions on how to study for the exam.
- Facts about the exam.
- Tips on writing a good exam.

Next: What you should be able to do after this course.

Desired effects of your study in databases

Concrete effects:
You are able to carry out certain **tasks** related to relational databases:

- Design using E/R modeling
- Database programming in SQL
- Analyze the possible behavior of transactions, create indexes, etc.

Less concrete (but not less important):

- You have improved your ability to make ideas and concepts precise.
- You have improved your ability of abstract thinking.

Tasks in database design

Some basic tasks in database design:

- Given a description in words of a data set, draw a corresponding E/R diagram.
- Given an E/R diagram, perform the conversion into relations.
- Add to an E/R diagram certain multiplicity or referential integrity constraints.
- Add key constraints or referential integrity constraints to an SQL database schema.
- Given a relation instance what are the possible keys?
- Find functional or multivalued dependencies in a relation schema.
- Determine whether a relation schema is in BCNF or 3NF.
- Decompose a relation into BCNF or 3NF.

Example problem 1: Normalization (8%)

Consider the following instance of relation R:

`student`	`course`	`grade`	`address`
William Gates	Operating systems	6	Microsoft Way 1
Jakob Nielsen	User interfaces	8	Silicon Valley 22
Jakob Nielsen	User interfaces	8	Glentevej 38
William Gates	Intro. to databases	7	Microsoft Way 1
Steve Jobs	Intro. to databases	10	Apple Drive 10
Steve Jobs	Operating systems	10	Apple Drive 10

a) Which of the following functional dependencies can be seen to **not** hold for R: `student` $\rightarrow$ `address`, `address` $\rightarrow$ `student`, `course` $\rightarrow$ `grade`. Argue in each case why a functional dependency is possible or impossible.

Answer to problem 1

a)

- `student` → `address` is no FD since `Jakob Nielsen` occurs in tuples with two different addresses.
- `address` → `student` is a possible FD: For each of the two addresses that occur more than once, the name in each tuple is the same.
- `course` → `grade` is no FD because the course `Intro. to databases` occurs in tuples with two different grades.

Tasks in database programming

Some basic tasks in database programming:

- Given an SQL statement, explain in words what it does.
- Given a database schema and a query described in words, write the query in SQL.
- Explain the difference between two SQL queries.
- Rewrite an SQL query such that it does not use some specific feature of SQL (e.g., with no subquery).
- Given a sequence of `GRANT` and `REVOKE` statements, state the privileges of each user.
- Write SQL to grant a user the right to access certain information.

Example problem 2: SQL programming (15%)

Again consider the relation R from Problem 1. The following SQL query is used in the statistics office:

```
SELECT student, avg(grade) AS GPA
FROM R
HAVING count(*) > 1
GROUP BY student;
```

a) What is the result of the query when run on R? Describe in words what the query computes in general.

b) Rewrite the query to avoid the `HAVING` clause. That is, write an equivalent SQL query that does not contain the keyword `HAVING`.

Answer to problem 2

a) When run on R, the query returns:

student	GPA
Jakob Nielsen	8
Steve Jobs	10
William Gates	6.5

In general, it computes the average grade of all students with names occurring in several different tuples.

b) The following query is equivalent to the query given:

```
SELECT student, GPA FROM (
   SELECT student, avg(grade) AS GPA, count(*) AS c
   FROM R
   WHERE c > 1
   GROUP BY student);
```

Other kinds of tasks

Some basic tasks in other areas:

- Given a data warehouse design, write the corresponding star schema.
- Given the description of a transaction, state what SQL isolation level would be appropriate for it.
- Given the description of two transactions, state what undesired event could happen if they were run at isolation level `READ COMMITTED`.
- Given some SQL query, state what kind of index could be used to speed it up.
- Given the time to execute certain queries with and without an index, and the frequency of updates to the underlying relations, determine what indexes should be created to minimize the total time.

Example problem 3: DB efficiency (10%)

You are appointed the administrator of a new DBMS that is used to register business transactions of a large multinational company, for use by the top management. Every day about 10,000 new business transactions are registered, and there are about 10 queries for old business transactions (identified by a transaction code). Having learnt about indexes on DBS, you consider placing an index on the transaction code to speed up queries. A full table scan processes 10,000 business transactions per second, while an index lookup can be done in 100 ms. The time used to insert is 40 ms without an index, and 100 ms with an index.

a) With 100,000 business transactions registered, what is the daily processing time (registering new + looking up old business transactions) with and without an index?

b) When will it become an advantage (in terms of total processing time) to use an index?

Answer to problem 3

a)
- **Without index:** The 10 full table scans will take 100 seconds, and the insertions will take 400 seconds. 500 seconds in total.
- **With index:** The 10 index lookups will take 1 second, and the insertions will take 1,000 seconds. 1,001 seconds in total.

b) The daily time using an index will remain 1,001 seconds. If no index is used, the time for insertions is 400 seconds, which means that not having an index is a bad idea once the table scans take more than 601 seconds. This will happen when the number of business transactions exceeds $10,000 \cdot 601/10 = 601,000$, i.e., after roughy 60 days of use.

Next: Suggestions on how to study for the exam.

The exam curriculum

The curriculum consists of:

- All parts of MDM and supplementary material written on the course schedule (you will have the list emailed soon).
- Regular lecture slides (not guest lectures).
- The example runs corresponding to the lecture slides.

Focus your effort using the lecture slides

The lecture slides focus on:

- The most *important* aspects of the course material, and
- The most *difficult* aspects of the course material.

In other words, you should spend most of your preparation time on getting a full understanding of the material on the slides (the book and supplementary material will be needed for this).

Things on the lecture slides you should *not* focus on:

- Information specific to Oracle: This was included for the sake of the exercises and will not be tested at the exam.

Reading the book

If there are parts of the curriculum that you haven't read, do so.

Also, you might want to re-read material in the course book.

I recommend that you:

- Read in the order of the book,
- *not* in the order suggested by the course schedule.

Don't spend time on memorizing the book's definitions of terms! The important thing is that you can use your knowledge in connection with specific tasks, not that you know all things by heart.

How to use the examples

The book contains a wealth of examples. It is a good idea to read an example:

- If you remember things better when they are made concrete.
- If you are not quite sure that you have properly understood the material it exemplifies.

But: The exam will not require knowledge of any particular example.

- You can safely skip an example if you have understood the material it exemplifies.

How to use the SQL example runs

There are files with SQL example runs for many lectures. They were designed to illustrate many of the points in the book and lectures.

Going through them is a good check to see if you understood all aspects properly.

Again: The exam will not require knowledge of any particular example run.

- You can safely skip an SQL example run if you have understood the material it exemplifies.

How to use exercises and previous exams

Since the exam is written, you may want to review some of the exercises and hand-ins. Most relevant for the exam are (the parts of) exercises that do not involve using a computer.

Note: Tøger has posted several example solutions on the course news group.

It is *highly* recommended that you try solving the problems from previous exams (available on course home page). All problems from previous DBS exams, and most problems from previous IDB exams are relevant.

How to ask questions

What to do if you encounter a question related to the course:

1. Write it down.
2. Discuss it with your study group (it is recommended to find someone to study for the exam with).
3. If still in doubt, ask your teacher:
   - Send e-mail to `pagh@itu.dk`.
   - There will be a question answering session on Monday June 6, 9 AM, where questions asked by e-mail will be answered.

Next: Facts about the exam.

Exam format

- Written exam, 4 hours, June 7, 9.00-13.00.
- "Open book" with all written aids allowed. You should bring:
  - The course curriculum.
  - Your notes.
  - Your answers to exercises and hand-ins.
- You may not bring a computer, but you may use a calculator (not vital).
- Remember to bring your study card (i.e., ITU key card).
- It is OK to write in Danish and use English technical terms.

The structure of the exam

- There will be 4-6 problems, each focusing on a specific topic.
- Each problem will be marked with a percentage, indicating its weight in the grade.
- Each problem will be split into 1-4 questions.
- Questions will have varying difficulty, with a tendency that the last questions in each problem have the highest difficulty.
- Difficult questions do not necessarily count more than easy ones.

Start with the questions you find easiest (quickest) to answer!

How the grading is done

- A problem marked with X% is worth X points.
- The maximum possible score is thus 100 points.
- The questions in a problem are worth roughly the same amount of points, depending on the "size" of the question.
- The passed/not passed boundary is 50 points.
- To get an average grade (8) you should get around 70 points.
- To get a top grade (10–13) you must get more than 80 points.

Next: Tips on writing a good exam.

Understand the question, answer the question

- Take your time to make sure you understand the question! Failing to do so is a common cause of losing points.
- Answer the question, the whole question, and nothing but the question.

It will usually not give any (extra) points to:

- Answer a more general question than what is posed, or
- Write something that seems related to the question.

Write short

Questions will usually be posed with relatively short answers in mind.

If you find yourself embarking on a long answer, you may have overlooked something.

A verbose answer is not necessarily bad, but takes longer to write.

Explain the non-obvious, state your assumptions

Any fully correct answer will receive full points. However, it is a good idea to add some explanation to show your understanding:

- It can make it clearer that the answer is correct.
- If there is some error in the answer, the explanation might show that this is not due to lack of understanding.

If you need to make any assumptions to answer the question, you should state them explicitly.

Order, please!

- Start the answer to every problem on a new sheet.
- Do not write on the back of the sheet.
- Number the sheets in an order consistent with that of the problems.