

---

Introduction to Databases, Fall 2003  
IT University of Copenhagen

**Lecture 12, part II: Exam preparation**

November 18, 2003

Lecturer: Rasmus Pagh

---

## — Today's lecture, part II —

---

- Suggestions on how to study for the exam.
- What to expect at the exam.
- Exam-type problems.

---

**Next: Suggestions on how to study for the exam.**

---

# — The curriculum

---

The curriculum consists of:

- All parts of GUW and supplementary material written on the course schedule (you will have the list emailed soon).
- The lecture slides.

Material *not* in the curriculum:

- Literature in square brackets on the course schedule.
- The example files accompanying 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 or MySQL: 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.

## — How to use the examples —

---

The book contains a wealth of examples, and there are files with SQL examples for many lectures.

It is a good idea to read/try out 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 exercises —

---

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.
- Are of the exam-type form (to be discussed later).

**Note:** Many exercises in the textbook have solutions available on the web.

## — 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.
  - You may use the course newsgroup to find a group.
3. If still in doubt, ask the teachers on the course
  - There will be question hours in the week before the exam.
  - Questions may also be posted on the course newsgroup.

---

**Next: What to expect at the exam.**

---

## — Exam format —

---

- Written exam, 4 hours.
- “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).
- Your answer may be written in English or Danish. (In the latter case it is OK to use English terms if you don't know the translation.)

## — 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!

## — Focus is on “most important points” —

The exam will test whether you have a satisfactory knowledge of databases, as covered by the curriculum.

Therefore questions naturally focus on the most important aspects, and not on particular details, of the curriculum.

A large part of the exam will be about “most important points”, as described on the lecture slides.

## — Most questions will have relatively short answers

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

If you find yourself answering a question using more than one page, you may have overlooked something.

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

## — 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: Exam-type problems.**

---

# — Questions in database design —

---

Some possible types of questions 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, 3NF, or 4NF.
- Decompose a relation into BCNF, 3NF, or 4NF.

## — Problem 1: Normalization (15%) —

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

## — Answer to problem 1 —

---

- a)
- student  $\rightarrow$  address is no FD since Jakob Nielsen occurs in tuples with two different addresses.
  - address  $\rightarrow$  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  $\rightarrow$  grade is no FD because the course Intro. to databases occurs in tuples with two different grades.
- b)
- student  $\twoheadrightarrow$  address is a possible MVD: For every student, his addresses occur in all possible combinations with his courses and grades. To check this it suffices to see that only Jakob Nielsen has more than one address, and both addresses occur together with his only course, User interfaces.
  - address  $\twoheadrightarrow$  student is a possible MVD because it is a possible FD, and any FD is also an MVD.
  - course  $\twoheadrightarrow$  grade is not an MVD. If it was, R should, for example, have contained the tuple (William Gates, Operating systems, 10, Microsoft Way 1), because of the first and last tuple.

# — Questions in database programming —

Some possible types of questions in database programming:

- Given an SQL statement, explain in words what it does.
- Given a database schema and a query 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).
- Write an SQL query corresponding to a relational algebra expression.
- 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.

## — 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);
```

## — Questions in other areas —

---

Some possible types of questions in other areas:

- Explain the difference between two relational algebra expressions.
- 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.

## — Problem 3: Database 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 IDB, 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 roughly 60 days of use.

## — More sample problems

---

I expect to supplement the sample exam-type problems you have just seen with a few more problems, so that there will be an entire sample exam.

You will be notified by email when these additional problems are available.

Good luck at the exam!