Databasesystemer, forår 2005

IT Universitetet i København

Forelæsning 1: Introduktion

3. februar 2005

Forelæser: Rasmus Pagh

Lidt om jeres undervisere

Rasmus Pagh (forelæser):

- Ph.d. i datalogi fra Aarhus Universitet, 2002.
- Forskning inden for algoritmik, bl.a. i spørgsmål omkring effektiv implementation af DBMSer, softwaren bag databaser.
- Har undervist i databaser på ITU siden foråret 2003, på kurserne *Introduction to databases* og *Advanced database technology*.

Hjælpelærere:

- Tøger Gralle Nørgaard (stud.dat. på KU).
- Lise Gregersen (INT stud. på ITU).

Desuden vil der blive en række gæsteforelæsere, der vil fortælle om konkrete eksempler fra det virkelige liv, med relation til kursusmaterialet.

Kursussprog: Danglish

- Kursusbog på engelsk.
- Forelæsninger på dansk.
- Tekniske termer uden entydig oversættelse dog altid på engelsk.
- Slides, opgaver, etc. på blandet dansk og engelsk.

Forelæsningen i dag

- Hvorfor studere databaser?
- Målsætning og indhold.
- Praktisk information.
- Introduktion til kursets indhold.
- Øvelser: Brug af ITUs Oracle database.

Hvorfor studere databaser?

Akademikerens grund:
Databaser har berøring med mange fascinerende emner indenfor datalogi.

Programmørens grund:
Har brug for databaser til at udvikle programmer.

Informationspilotens grund:
Vil arbejde med, og trække information ud fra, store, dynamiske datamængder.

Kapitalistens grund:
Alle har brug for databaser – her er der penge at tjene!

Målsætning for kurset

Du får kendskab til de grundlæggende regler for datamodellering og datamanipulation i forbindelse med databaser for at kunne forstå opbygningen af systemer, hvor databaser indgår.

Den relationelle model til design af databaser mødes overalt i kommercielle produkter, og vil være hovedfokus for kurset.

Kursusindhold

Kursusindhold i hovedpunkter:

- Grundlæggende databasebegreber og database management.
- Datamodellering: E/R modellen, normalisering og andre datamodellerings-metoder.
- Structured Query Language (SQL): Forespørgsler, datadefinition, domæne- og integritetsspørgsmål, samt datamanipulation.
- Transaktioner.
- Objekt-orientering i databaser.
- Data warehousing og data mining.
- XML og datahåndtering.

Kursusformat

Forelæsninger: Torsdag 17.00-ca. 19.00, Aud. 3.

Blanding af forelæsninger og korte problemsessioner uden forberedelse.

Øvelser: Torsdag 19.00-21.00 (samt måske 15.00-17.00?)

Bortset fra første gang forudsætter øvelserne, at deltagerne har set på, og forsøgt at løse opgaverne i forvejen.

Obligatoriske afleveringer: 3 individuelle og 3 gruppeopgaver.

Eksamen: Skriftlig, 4 timer, d. 7. juni 2005. Tidligere eksamenssæt bliver gjort tilgængelige via kursushjemmesiden.

Overlevelse 1: Obligatoriske afleveringer

- Afleveres som hovedregel til hjælpelæreren i forbindelse med øvelserne. Anden afleveringsmåde skal aftales med hjælpelæreren.
- **Alle opgaver bortset fra højst 1 individuel opgave skal godkendes for at kunne gå til eksamen.**
- Der er mulighed for 1 genaflevering af hver opgave, men kun hvis der er gjort et seriøst forsøg på at besvare den til afleveringsfristen.
- Opgaver ligger på kursushjemmesiden mindst 2 uger før fristen.
- Gruppeopgaver **skal** laves i grupper med 3 eller 4 studerende (se dagens øvelsesark for detaljer).

Overlevelse 2: Mad

- Nærmeste åbne kantine er på KUA (2 min. gang fra ITU) – åbent til 19.30.
- Diverse slikautomater på ITU.
- Mad og drikke må **ikke** indtages i auditoriet
- . . . men pauserne burde række til en sandwich.

Kursusmateriale

- Kursushjemmeside: Nyheder, læseanvisninger, slides, øvelser, afleveringsopgaver.
  URL: `www.itu.dk/people/pagh/DBS05/`
- Lærebog: Modern Database Managament, 7th ed, af Hoffer, Prescott og McFadden.
- Supplerende materiale on-line på
  `www.itu.dk/people/pagh/DBS05/Intranet/`
  (kræver brugernavn og password, som I vil få tilsendt per e-mail).

Lærebogen sælges i Samfundslitteratur (ved siden af ITUs information).

Efter pausen: Kursusoverblik

- What is a database?
- What is a database management system (DBMS)?
- What is a relational database?
- How are databases designed?
- How are databases programmed?
- . . . and other subjects of the course.

What is a database?

According to Webster's dictionary:

> **da·ta·base**
> a usually large collection of data organized especially for rapid search and retrieval (as by a computer)

Remark:
The need for (and the ability to give) rapid answers to a multitude of **queries** about data is increasing. Databases have thus grown to perform much more advanced processing than search and retrieval.

What is a database management system?

Database management system (DBMS):

Software system used when implementing databases

more precisely

System for providing **efficient**, **convenient**, and **safe** storage of and **multi-user** access to (possibly **massive**) amounts of **persistent** data.

What is a relational database?

All major general purpose DBMSs are based on the so-called **relational data model**. This means that all data is stored in a number of tables (with named columns), such as:

accountNo	*balance*	*type*
12345	1000.00	`savings`
67890	2846.92	`checking`
32178	-3210.00	`loan`
...	...	...

For historical, mathematical reasons such tables are referred to as **relations**. This course is mainly on relational databases, and on relational database management systems (RDBMSs).

How are relational databases designed?

It is often far from obvious to decide how to store data from a application as relations. A considerable part of the course will deal with a methodology for good relational database design.

Problem session: (5 minutes, discuss in groups of 2-4 students)

Suggest how to represent the following types of data as one or more relations:

- An address book.
- A phone operator's record of phone calls.

Can you avoid (or reduce) duplication of information?

Database design methodology

We will cover the dominant design methodology for relational databases, which consists of three steps:

1. Identify all relevant **E**ntities and **R**elationships, and describe them using so-called **E/R model notation**. (Lecture 3.)
2. Convert the E/R model to a number of relations. (Lecture 3.)
3. Eliminate (or reduce) redundancy by splitting relations. This process is called **normalization**. (Lecture 4.)

In real, complex systems the process is often iterated several times before a final design is reached.

How are relational databases programmed?

The success of relational databases is largely due to the existence of powerful **programming languages** for writing database queries.

The most important such language is **SQL** ("Structured Query Language", sometimes pronounced "sequel").

Important properties:

- **convenient**: queries can be written with little effort
- **efficient**: even for large data sets, a good DBMS can answer queries written in SQL quickly (compared to the fastest possible)

SQL example

Consider the following relation, which we give the name `Accounts`:

accountNo	*balance*	*type*
12345	1000.00	`savings`
67890	2846.92	`checking`
32178	-3210.00	`loan`

SQL to get the balance of account 67890:

```
SELECT balance
FROM Accounts
WHERE accountNo = 67890;
```

More SQL examples

```
SELECT accountNo, balance
FROM Accounts
WHERE type = 'loan' AND balance < -10000;
```

```
SELECT *
FROM Accounts
WHERE accountNo > balance;
```

Even more SQL

Suppose we have a relation `Holders` related (!) to `Accounts`:

accountNo	*name*	*address*
12345	Scrooge	Money Tank
67890	Donald Duck	Apple Rd 13
67890	Daisy Duck	Apple Rd 13
32178	Gyro Gearloose	Inventor's lane 1

SQL to get names of all holders of checking accounts:

```
SELECT name
FROM Accounts, Holders
WHERE Accounts.accountNo = Holders.accountNo;
```

— General form of "simple" SELECT-FROM-WHERE

```
SELECT name1, name2, ...
FROM relation1, relation2, ...
WHERE <some condition>;
```

The * is a shorthand for the list of all column names (or **attributes**).

You will learn next week what happens when there is more than one relation involved in the SELECT-FROM-WHERE.

Problem session

Consider again the relation `Accounts`:

accountNo	*balance*	*type*
12345	1000.00	`savings`
67890	2846.92	`checking`
32178	-3210.00	`loan`

Problem session: (5 minutes, discuss in groups of 2-4 students)

Write an SQL query that finds all accounts (i.e. account number and type) that have positive balance.

"Joining" information in two relations

SQL's `NATURAL JOIN` operator combines information from two relations, by merging tuples that agree on the common attributes.

Example:

cpr	*street*	*city*
300266-3278	Louis Ln 1	Louisiana
300266-3278	Blixen Park 4	Ørestad
310671-2343	Glentevej 67	København
311177-2342	Grønager 222	Vejle

cpr	*course*	*grade*
300266-3278	DBS	9
300266-3278	ADBT	8
310671-2343	OOP	10
310671-2343	IPBR	9
310671-2343	IDBI	6
311177-2342	IPBR	7

The natural join of these relations results in a new relation with 8 tuples, where each tuple contains cpr number, an address, and a grade of a student.

Syntax and semantics of SQL

As you have seen, SQL queries resemble questions in English. Often, the effect of an SQL query can easily be intuitively understood.

During the course you will learn how to program much more complex queries in SQL. To be able to do that you need a precise understanding of SQL's:

- **Syntax.** The *way* SQL may be written.
- **Semantics.** The *meaning* of an SQL statement.

Other aspects of SQL

In addition to queries, SQL can be used to express many types of database operations:

- Define new relations.
- Perform changes to data.
- Set up **constraints** and **triggers**.
- Manage users, security, rights, etc.
- Control **transactions** in a multi-user environment.
- ...

These aspects are covered later in the course.

Other subjects treated in the course

Towards the end of the course we will consider:

- Data warehousing and OLAP ("organizational intelligence").
- Database efficiency.
- Transactions.
- XML and data interchange.

Most important points in this lecture

As a minimum, you should after this lecture:

- Know how to qualify for the exam.
- Know a little about some key concepts: Relation, (R)DBMS, SQL queries, normalization, relational algebra, and know how they fit into this course.
- Understand how a subset of a relation `R` can be obtained using "`SELECT ... FROM R WHERE ...`".

Læsevejledning

Skemaet angiver MDM kapitel 1 og 2 som litteratur til denne forelæsning.

Disse kapitler giver et bredt billede af databasesystemers rolle i virksomheder, og af de processer, der bruges når systemer skal udvikles, altså også emner jeg ikke har berørt ved forelæsningen.

Vi er startet på materiale fra MDM kapitel 7, og resten af kapitlet gennemgås næste gang. Desuden har vi set kort på "natural join" som er beskrevet i MDM kapitel 8.

Næste uge

Næste uge ser vi mere på relationer og SQL:

- Hvad er den relationelle datamodel?
- Hvad er en relation helt præcist?
- Hvad er de grundliggende måder at skrive SQL udtryk på?
- Hvordan opdaterer man data i en relation?
- Hvordan skaber man nye relationer i SQL?