

Data Storage and Formats

Lecture 1
Fall 2008
Anna Pagh

Some figures are borrowed from the ppt slides from the book used in the course, Database systems by Kiefer, Bernstein, Lewis Copyright © 2006 Pearson, Addison-Wesley, all rights reserved.



Today's lecture

- Introduction
 - Overview of content
 - Learning outcome/course goals
- How the course/teaching is organized
- Practical information
- Modeling exercise



"Problem session"

What are the (3) most important things you learned in the course?



Course goals

From course description:

The main part of the course deals with relational databases, including theory and practice for modeling and querying a database. The course also considers storage and manipulation of data in XML format.



Database

- "a usually large collection of data organized especially for rapid search and retrieval (as by a computer)" - from m-w.com
- "a collection of data items related to some enterprise" – from KBL
- Other more involved queries than just search and retrieval



DBMS

- DataBase Management System
- Software system used when implementing databases.
- Provides efficient, convenient, and safe storage of, and multiuser access to (possible massive) amounts of persistent data
- Supports a high-level language for access (queries and updates) to the data.



XML

- eXtensible Markup Language
- A format for semistructured data
- Framework for defining markup languages
- Resembles HTML, but you decide the tags
- XML describes content, while HTML describes appearance



XML example

```
<dictionary>
  <entry id=31>
    <word>banana</word>
    <meaning>yellow fruit</meaning>
  </entry>
  <entry id=83>
    <word>milk</word>
    <meaning>white fluid</meaning>
  </entry>
</dictionary>
```



Course goal

- After the course the students should be able to:
- write SQL queries, involving multiple relations, compound conditions, grouping, aggregation, and subqueries.



SQL

- The most important programming language for databases
- Structured Query Language ("sequel")
- Declarative: specify what you want, not how to get it
- Takes one or more tables as arguments and produces a table as a result
- Not only queries, also updates and schema definition



Relational databases - some terminology

- In relational databases data is stored in tables (aka. relations)
- A table is a set of rows (aka. tuples)
- Columns (aka. attributes) have a name and a type

Id	Name	Address	Status
111111111	John Doe	123 Main St.	Freshman
666666666	Joseph Public	666 Hollow Rd.	Sophomore
111223344	Mary Smith	1 Lake St.	Freshman
987654321	Bart Simpson	Fox 5 TV	Senior
023456789	Homer Simpson	Fox 5 TV	Senior
123454321	Joe Blow	6 Yard Ct.	Junior

FIGURE 2.1 The table STUDENT. Each row describes a single student.

Figure: Copyright © 2006 Pearson Addison-Wesley. All rights reserved.



SELECT statement

```
SELECT Name
FROM STUDENT
WHERE Id=987654321
```

returns a table with one row and one column

Id	Name	Address	Status
111111111	John Doe	123 Main St.	Freshman
666666666	Joseph Public	666 Hollow Rd.	Sophomore
111223344	Mary Smith	1 Lake St.	Freshman
987654321	Bart Simpson	Fox 5 TV	Senior
023456789	Homer Simpson	Fox 5 TV	Senior
123454321	Joe Blow	6 Yard Ct.	Junior

FIGURE 2.1 The table STUDENT. Each row describes a single student.

Figure: Copyright © 2006 Pearson Addison-Wesley. All rights reserved.



SELECT statement

```
SELECT *
FROM STUDENT
WHERE Id=987654321
```

* means "all columns" - but is not a "wildcard" character

returns a table with one row and all 4 columns

Id	Name	Address	Status
111111111	John Doe	123 Main St.	Freshman
666666666	Joseph Public	666 Hollow Rd.	Sophomore
111223344	Mary Smith	1 Lake St.	Freshman
987654321	Bart Simpson	Fox 5 TV	Senior
023456789	Homer Simpson	Fox 5 TV	Senior
123454321	Joe Blow	6 Yard Ct.	Junior

FIGURE 2.1 The table STUDENT. Each row describes a single student.

Figure: Copyright © 2006 Pearson Addison-Wesley. All rights reserved.



SELECT statement

```
SELECT Id, Name
FROM STUDENT
WHERE Status='Senior'
```

Id	Name
987654321	Bart Simpson
023456789	Homer Simpson

Id	Name	Address	Status
111111111	John Doe	123 Main St.	Freshman
666666666	Joseph Public	666 Hollow Rd.	Sophomore
111223344	Mary Smith	1 Lake St.	Freshman
987654321	Bart Simpson	Fox 5 TV	Senior
023456789	Homer Simpson	Fox 5 TV	Senior
123454321	Joe Blow	6 Yard Ct.	Junior

FIGURE 2.1 The table STUDENT. Each row describes a single student.

Figure: Copyright © 2006 Pearson Addison-Wesley. All rights reserved.



SELECT statement

```
SELECT COUNT(*)
FROM STUDENT
WHERE Status='Senior'
```

returns a table with the value 2, i.e. #rows with 'Senior'
COUNT is an aggregate function

Id	Name	Address	Status
111111111	John Doe	123 Main St.	Freshman
666666666	Joseph Public	666 Hollow Rd.	Sophomore
111223344	Mary Smith	1 Lake St.	Freshman
987654321	Bart Simpson	Fox 5 TV	Senior
023456789	Homer Simpson	Fox 5 TV	Senior
123454321	Joe Blow	6 Yard Ct.	Junior

FIGURE 2.1 The table STUDENT. Each row describes a single student.

Figure: Copyright © 2006 Pearson Addison-Wesley. All rights reserved.



SELECT statement

```
SELECT *
FROM STUDENT
WHERE Id < 666666666 AND NOT (Status='Senior')
```

returns a table with three rows
A condition in WHERE can be any Boolean expression

Id	Name	Address	Status
111111111	John Doe	123 Main St.	Freshman
666666666	Joseph Public	666 Hollow Rd.	Sophomore
111223344	Mary Smith	1 Lake St.	Freshman
987654321	Bart Simpson	Fox 5 TV	Senior
023456789	Homer Simpson	Fox 5 TV	Senior
123454321	Joe Blow	6 Yard Ct.	Junior

FIGURE 2.1 The table STUDENT. Each row describes a single student.

Figure: Copyright © 2006 Pearson Addison-Wesley. All rights reserved.



General form of simple SELECT statements

```
SELECT A1, A2, ...
FROM R1, R2, ...
WHERE <condition>
```

where

- A₁, A₂, ... are column names,
- R₁, R₂, ... are tables,
- condition is a boolean expression involving columns from R₁, R₂, ...



Multi-table SELECT

```
SELECT *
FROM Car, Owner
WHERE Id=Ownerid
```

Join

- The FROM clause produces the Cartesian product of the tables Car and Owner.
- Cartesian product contains all combinations of rows from Car and Owner.
- If Car has n rows with i columns and Owner has m rows with j columns the Cartesian product is a table with n*m rows with i+j columns.



Course goal

After the course the students should be able to:

- suggest a database design according to the relational model, and present it as an SQL schema, using the concepts key, type, and constraint.

Problem session

(In small groups, 10 minutes)

There are several ways to design a relational database for a given application.

Discuss and suggest how to represent an *address book* as one or more relations. You need to make several assumptions.

Can you avoid (or reduce) duplication of information?

SQL schema

- Defines the relations in a database, i.e. specifies the tables, column names and types
- An integrity constraint is a statement about legal instances of a database
 - All students have unique ids (a key)
 - A student can't change status from senior to freshman
 - Enrolment date is before graduation date

Course goal

After the course the students should be able to:

- find functional dependencies in a relation and perform decomposition to eliminate unwanted dependencies.

Normalization

Redundant information is a problem:

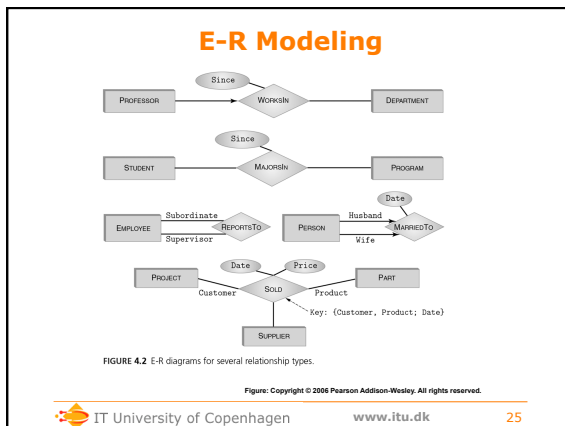
- Extra storage
- Hard to update

Normalization theory helps to refine the design to get a more efficient way to organize data in relations.

Course goal

After the course the students should be able to:

- define a database design by E-R modeling, using the concepts entity, attribute, key, cardinality, and relationship



Course goal

After the course the students should be able to:

- express simple relational expressions using the relational algebra operators select, project, join, intersection, union, set difference, and cartesian product.

IT University of Copenhagen www.itu.dk 26

Relational algebra

The mathematical basis of SQL.

SQL expressions can be translated into relational algebra expressions and vice versa.

```
SELECT Lastname, Regnr, Color
FROM Car, Owner
WHERE Id=Ownerid AND Color='Pink'
```

$$\pi_{\text{Lastname, Regnr, Color}}(\sigma_{\text{Color}='Pink'}(\text{Car} \bowtie_{\text{Id}=\text{Ownerid}} \text{Owner}))$$

IT University of Copenhagen www.itu.dk 27

Relational algebra

SQL is declarative (what)
 Relational algebra is procedural (how)

The DBMS translates SQL to relational algebra.

The query optimizer translates the expression to an equivalent expression that can be evaluated more efficiently.

IT University of Copenhagen www.itu.dk 28

Course goals

After the course the students should be able to:

- decide if a given index is likely to improve performance for a given query.

IT University of Copenhagen www.itu.dk 29

Problem session

Assume that we have a large table storing information of all persons in Denmark with 5.5 million rows storing cpr number, name, and income.

We write the following SQL query:

```
SELECT name, income
FROM Person
WHERE income > 1 500 000
```

- How long does it take to find the resulting table? What does the time depend on?
- Using your knowledge from the algorithms course, how can you improve the query time?

IT University of Copenhagen www.itu.dk 30

Course goals

After the course the students should be able to:

- identify possible problems in transaction handling, related to consistency, atomicity, and isolation.
- apply a simple technique for avoiding deadlocks



Transaction

A transaction is a sequence of operations on a database that belong together.
Useful when multiple users update the database in parallel.

Example:

Two persons with a shared bank account try to withdraw 100 kr at the same time.

Transaction:

- 1) read balance and store in variable B
- 2) if $B \geq 100$ then $B := B - 100$
- 3) write B to balance



ACID Properties

Atomicity: Transactions runs to completion or has no effect at all

Consistency: After a transaction completes, the integrity constraints are satisfied

Isolation: Transactions executed in parallel has the same effect as if they were executed sequentially

Durability: The effect of a committed transaction remains in the database even if the computer crashes.



Course goals

After the course the students should be able to:

- use SQL in applications (Java).



Course goals

- write simple XML Schemas and simple XQuery.
- explain the meaning of a DTD, and the effect of simple XSLT transformations.



XML Schema

Data definition language for XML documents, i.e. describes the structure of an XML document.

Describes tag names and types, constraints and more.

DTD is another (more limited) data definition language for XML documents.



XQuery

A query language for XML.

Similar to SQL:

FOR variable declaration
WHERE condition
RETURN result

New: "Path expressions" are used to extract data from the XML document.



Teaching

- Lectures, 3 h
 - No preparation expected
 - But expected that you have studied last week's text
 - Problem sessions
- Afternoon session, 2 h
 - Exercises
 - Current week, no preparation
 - Previous week, homework
 - Project
- Hand-ins
 - Feedback



Homepage

- www.itu.dk/courses/BDLF/E2008/
 - Slides
 - Exercises
 - Reading directions
 - Hand-ins
 - Links to recourses
 - News



Modeling exercise

- <http://www.itu.dk/people/pagh/IDB05/exam05.pdf>, Question 1, Database design
- Suggest a design by describing the relations, including attributes and types, and how the different relations reference to each other

