



CHALMERS
UNIVERSITY OF TECHNOLOGY



UNIVERSITY OF GOTHENBURG

Introduction to Databases

LECTURE 1

Dr. Philipp Leitner



philipp.leitner@chalmers.se



@xLeitix

GENERAL COURSE INFORMATION

Usual Time: 13:15 - 14:45

Lecture TUE, THU

Supervision WED, FRI

Language: entirely in English

(please also send mails etc. in English)

Credits: 7.5 HEC

1 HEC ~ 30 hours of effort (225 hours total)

Web Page

All information etc. is on GUL:

<https://gul.gu.se/courseld/82750/content.do?id=39332486>

Lecture list:

<https://gul.gu.se/courseld/82750/content.do?id=39765276>

Discussion Forum:

<https://gul.gu.se/courseld/82750/courseDiscussListForums.do>

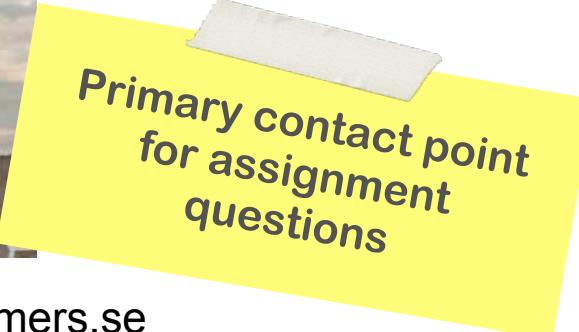
Teachers



philipp.leitner@chalmers.se



scheuner@chalmers.se



TAs

Justinas Stirbys
gusstirju@student.gu.se

Andres De Biase
gusdebana@student.gu.se

Renjith Sebastian
renjith@student.chalmers.se

Weekly Schedule

Lectures two times a week

(almost) every week TUE, THU (13:15 - 14:45)

Alfons

Supervision twice a week

Every week while the assignments are running WED, FRI (13:15 - 14:45)

Mållgan

Regularly Check Lecture List on GUL

Event

Content

- Welcome
- ▼ Course Information
 - Course Syllabus
 - Schedule**
 - Literature List
 - Room Booking
 - Course PM
- ▶ Course Evaluation

Documents

Surveys

Assignments

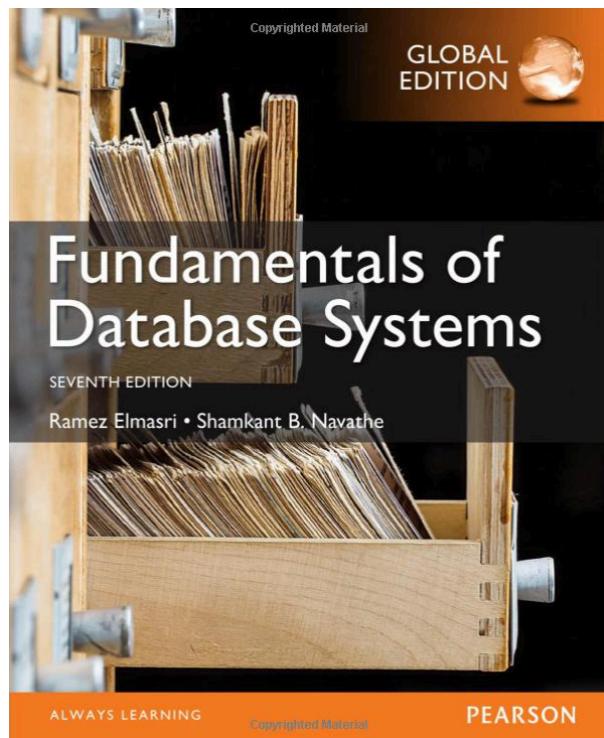
◀ Previous ▶ Next ▶ Search + Maximize the content ↗ Save as PDF

Schedule

Below is the **Class Schedule** for this course. This is in addition to the official [Room Booking](#) as published by the university. Consider the booking schedule the authoritative source for information, as we may have to cancel classes (which does not show up in the room reservation list):

Week	Date & Time	Event	Teacher	Room	Resources
Week 1					
	January 16th, 13:15 - 14:45	Lecture 1 - Kick-Off and Introduction to Databases	Philipp Leitner Alfons	TBA	
	January 17th, 13:15 - 14:45	<i>No supervision</i>			
	January 18th, 13:15 - 14:45	Lecture 2 - Entity Relationship Diagrams	Philipp Leitner Alfons	TBA	
	January 19th, 13:15 - 14:45	<i>No supervision</i>			
Week 2					

Text Book



Fundamentals of Database Systems

by Ramez Elmasri and Shamkant B. Navathe

Pearson Global Edition

7th Edition

(older editions are fine for most of the lecture, but the NOSQL and Map/Reduce part is not covered)

<https://www.adlibris.com/se/bok/fundamentals-of-database-systems-global-edition-9781292097619>

Grading

Written Exam:

4.5 hec

GU students:

Pass with Distinction (**VG**), Pass (**G**) and Fail (**U**)

Chalmers students:

Five, Four, Three, Fail

3-hour written exam in the week of March 13th

Assignments:

3 hec

Pass (**G**) and Fail (**U**)

Detailed Grading Scheme

Written Exam:

VG / Five: $\geq 90\%$ of exam points

Four: $\geq 70\%$ of exam points

G / Three: $\geq 50\%$ of exam points

U / Fail: $< 50\%$ of exam points

Assignments:

G: $\geq 70\%$ of assignment points in all three assignments

U: $< 70\%$ of assignment points in at least one assignment

Assignments

All assignments are done in **teams of two**. Choose your own partner and register like that in GUL.

Assignment 1:

Entity Relationship Diagrams and Relational Algebra

Due Feb. 5th, Midnight

Assignment 2:

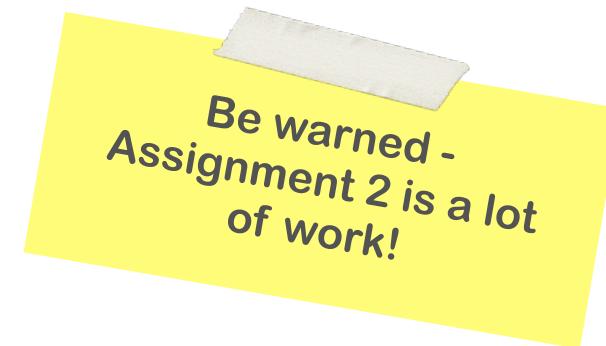
SQL and JDBC

Due Feb. 19th, Midnight

Assignment 3:

MongoDB and Map/Reduce

Due Mar. 5th, Midnight





Next Written Exam Dates

Main exam date: **March 13th, 2018**

(after that: June 7th and August 22nd, 2018)

Forum

We will be operating a discussion forum on GUL:

<https://gul.gu.se/courseld/82750/courseDiscussListForums.do>

This will be the best way to get answers to your questions

We will check the forum frequently, and so should you.

Students are encouraged to answer questions as well!

Self- and Peer-Assessment Forms

There's a form on GUL that every student should submit for every assignment

Three questions:

What did you do?

What did your partner do?

How much time did you invest (in hours)?

The last question is mainly to improve our assignments for the next years - this has no impact on grading.

Print and sign the form, and bring it into the lecture, supervision, or my office hour.

Office Hour

I hold office hours

THU, 09:00 - 10:00

in my office (Jupiter, 4th floor, room 411).

Outside of office hours **you will need an appointment**. Drop me an email if something urgent comes up.

Student Representatives

We need 2 - 5 student representatives

- Feedback during 2 meetings (one midway, one after the lecture)
- Can raise issues at any time

Volunteers - please contact me after the lecture

Bug Bounty Programme

You can get a small number of extra points for the **next assignment grading** for each unique bug you report

Bugs can be:

- Errors in the slides
- Errors in assignment texts
- Errors in assignment solutions

Bug Bounty Programme - Guidelines

We'll reserve the right to assign points based on the severity of the bug:

Roughly:

- Typos, grammar errors, other inconsequential stuff: *0 - 1 points*
- Inelegant / inefficient solutions: *2 - 3 points*
- Proposed solution does not work at all: *4 - 5 points*

Bug reports should come with an error case and/or proposed improvement.

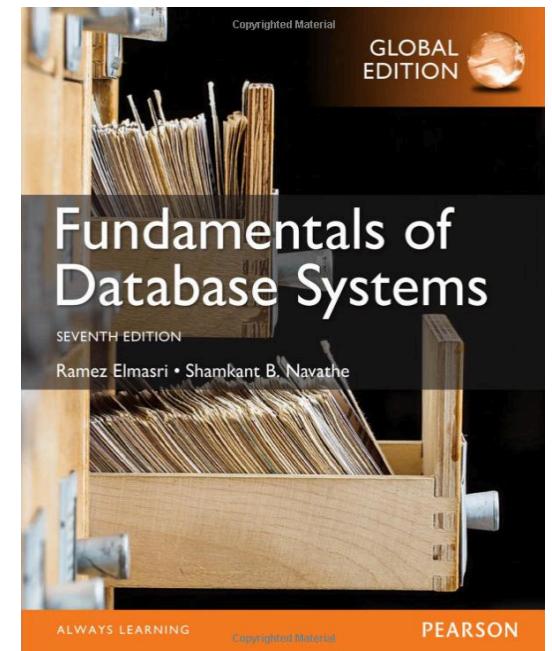
Submit bugs via forum:

First come / first serve (each bug will only be awarded a bounty once)

LECTURE 1

Covers ...

**Chapter 1 and Chapter 2
(+ short information theory detour)**



What we will be covering

About data, information, and knowledge

Types of databases

What's a database, really?

Database architecture and concepts

What we will be covering



About data, information, and knowledge

Types of databases

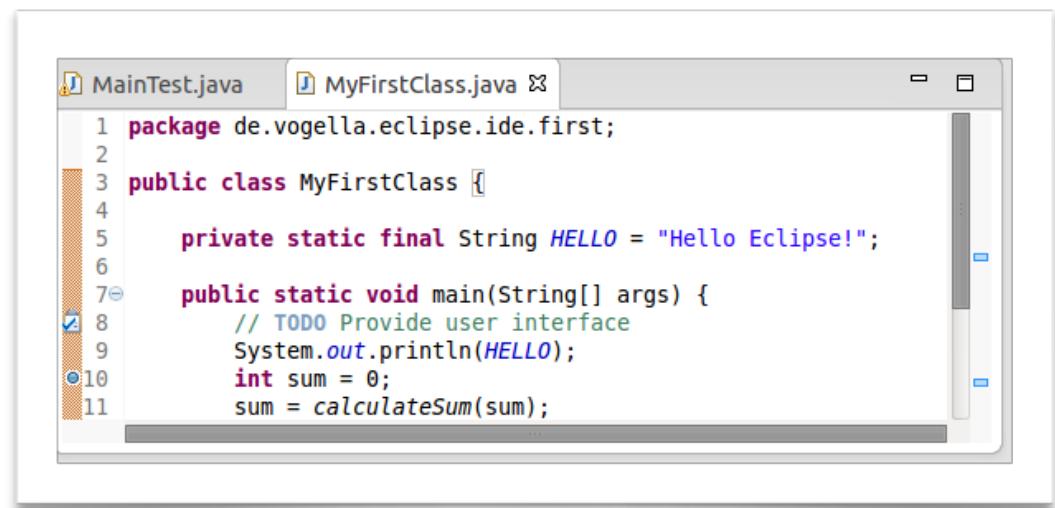
What's a database, really?

Database architecture and concepts

That's the information theory part

Most programming is about *data*

Simplest cases:
Variables
Text files

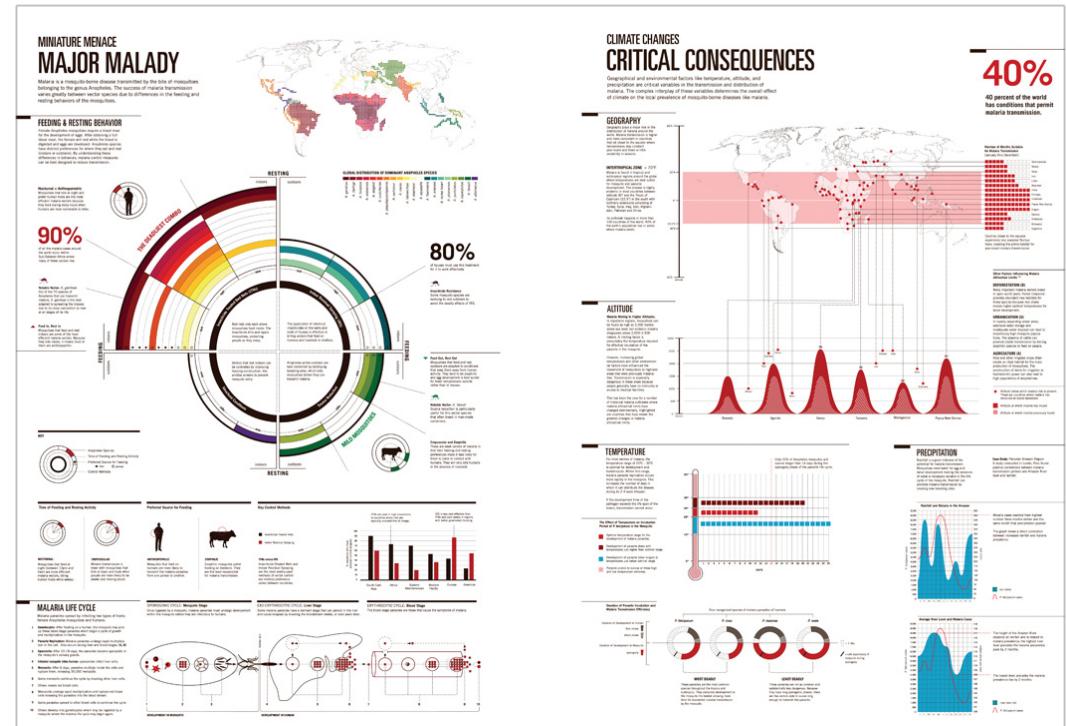
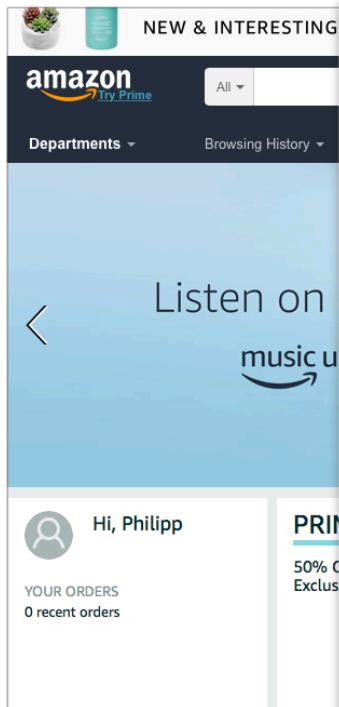


```
MainTest.java
1 package de.vogella.eclipse.ide.first;
2
3 public class MainTest {
4
5     public static void main(String[] args) {
6         // TODO Provide user interface
7         System.out.println("Hello Eclipse!");
8     }
9 }
10
11

MyFirstClass.java
1 package de.vogella.eclipse.ide.first;
2
3 public class MyFirstClass {
4
5     private static final String HELLO = "Hello Eclipse!";
6
7     public static void main(String[] args) {
8         // TODO Provide user interface
9         System.out.println(HELLO);
10        int sum = 0;
11        sum = calculateSum(sum);
12    }
13 }
```

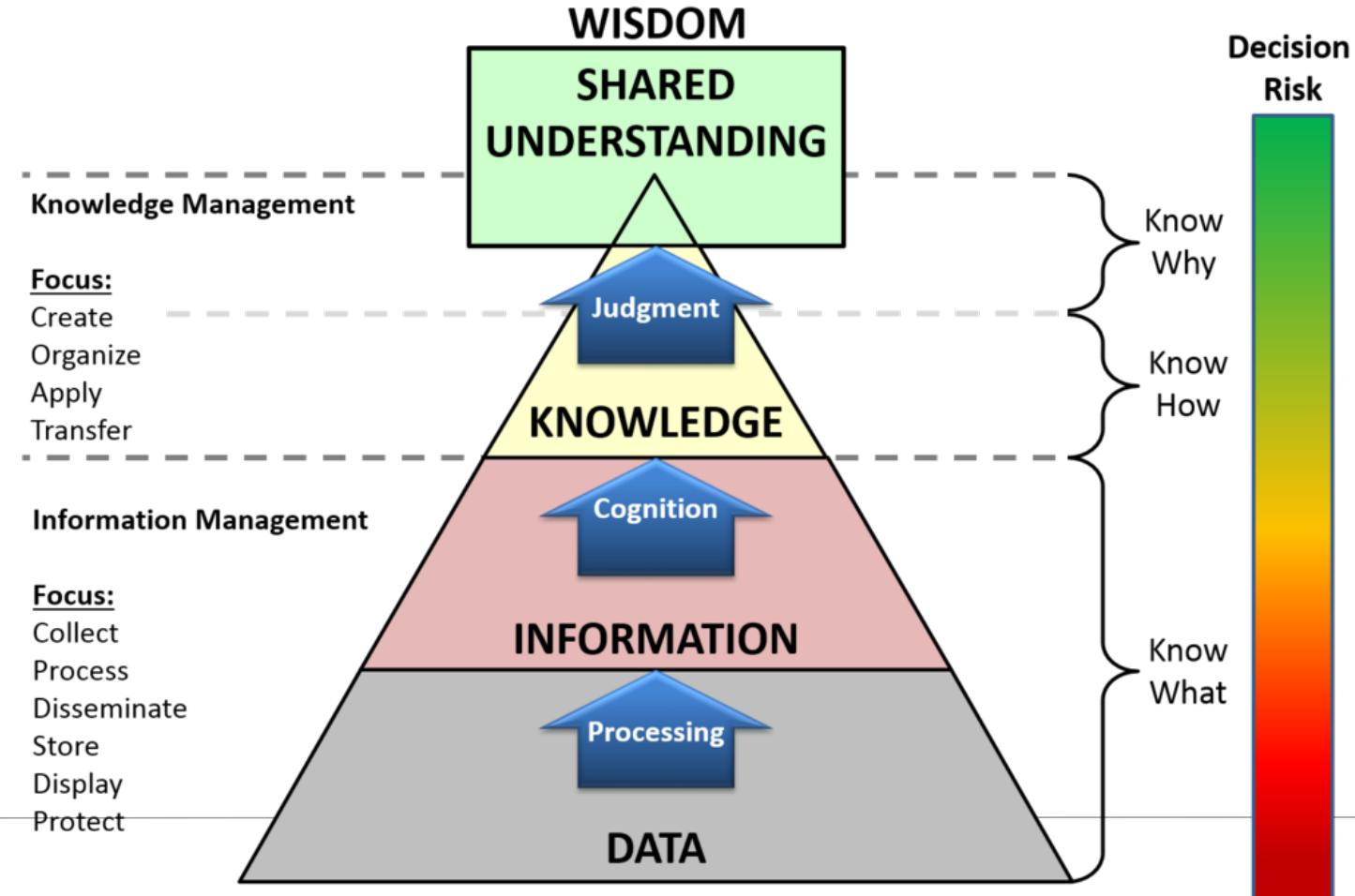
... not sufficient for many use cases

Most programming is about *data*



https://en.wikipedia.org/wiki/DIKW_pyramid

Knowledge Management Cognitive Pyramid



Data

Signals that are stored (on disk, in memory, ...)

No “**meaning**” in itself

May or may not be **correct** (“noise”)

May or may not be **redundant**



Information

Data with **labels** (== some sort of description)

Can be a **schema**

Or just implicit labels

Generally **assumed to be** not redundant nor wrong

Data and information is what we primarily deal with!

(but the meaning will need to come from your programs)

Knowledge / Wisdom

Less well-defined concepts

Generally:

Information + meaning

Typically **not the realm of computer programmes**

But AI (artificial intelligence) is stretching the boundaries

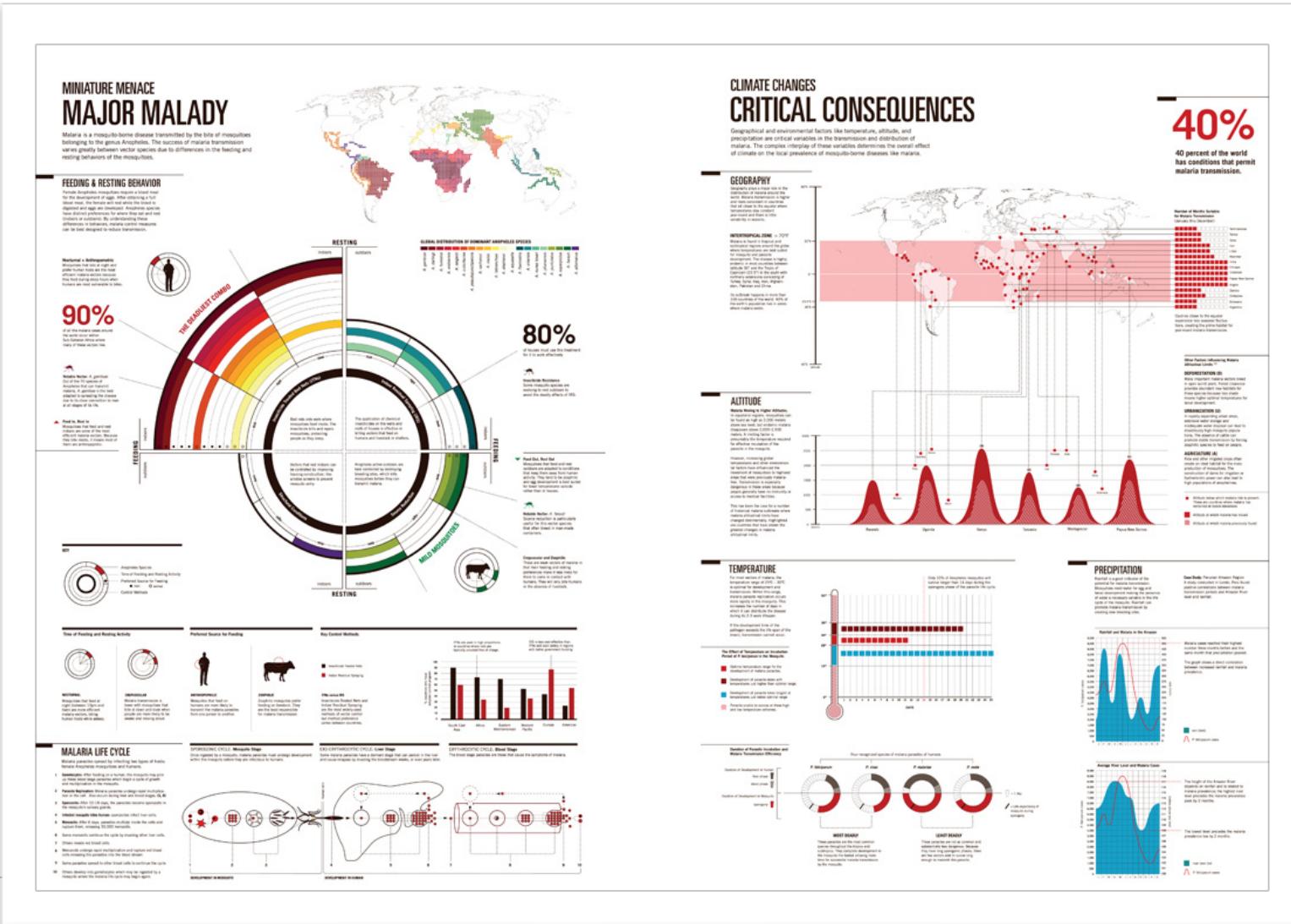
Data Visualization

Common (human) way to **go from information to knowledge**
Good visualisation helps understanding

Quick overview of lots of data

Easier to establish relationships

Easier to compare data



Types of Visualizations

Textual (e.g., printing to `System.out`)

Tables

MS Excel style visualization

Plots

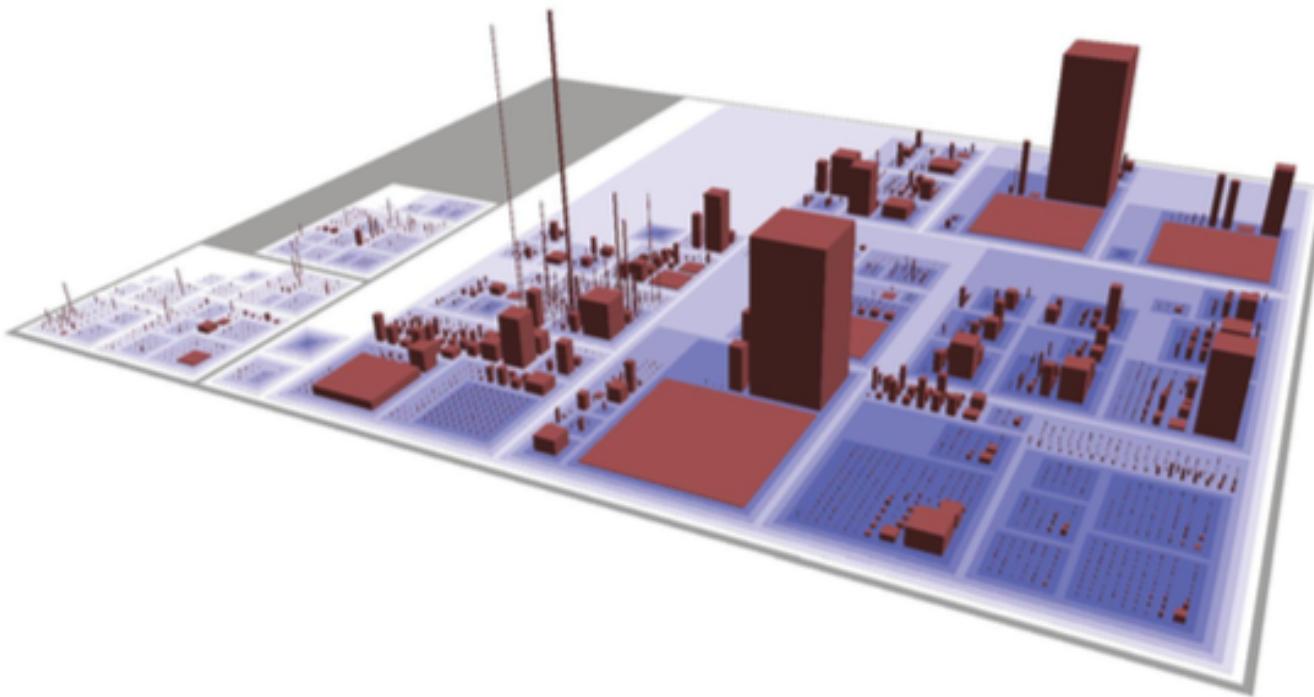
Line charts, bar charts, boxplots, pie charts, ...

Custom

E.g., the “city metaphor” for visualization of source code

City Metaphor

Wettel and Lanza. Software systems as cities. VISSOFT 2007.



Types of Visualizations

Textual (e.g., printing to `System.out`)

Tables

MS Excel style visualization

Plots

Line charts, bar charts, boxplots, pie charts, ...

Custom

E.g., the “city metaphor” for visualization of source code

What we will be covering

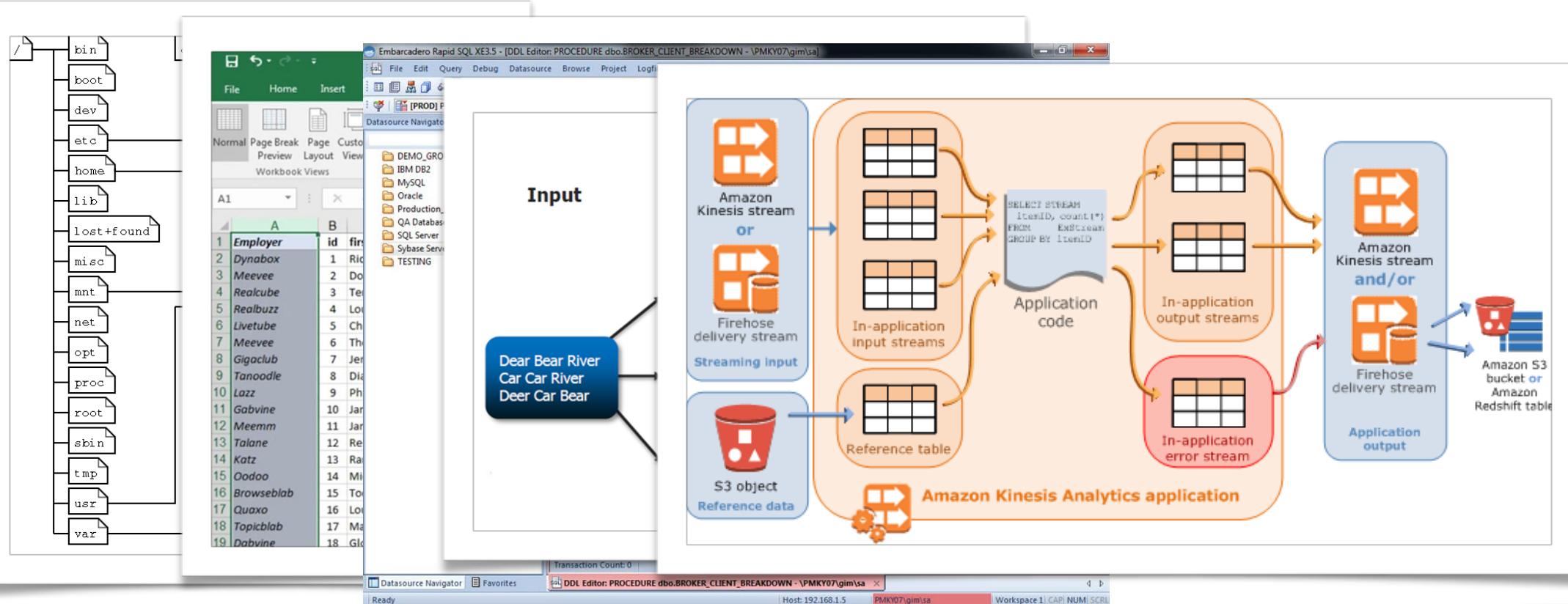
About data, information, and knowledge

 **Types of databases**

What's a database, really?

Database architecture and concepts

Types of databases





Types of databases

Files and File Systems

Spreadsheets

SQL

NOSQL

Cloud services

Basic data operations - CRUD

Create
Read
Update
Delete



*Fundamental Operations Most
Databases Will Support*

Querying

Querying is in reality where the “meat” in most real database systems is

Special form of “read”

Whereas “read” delivers exactly one entry, a query is used to **retrieve**:

- More than one entry

- Parts of one or more entries (e.g., only the IDs)

- Based on complex constraints

Examples:

- Give me all files that end with “*.exe”

- Give me the last names of all students enrolled in DIT032

What we will be covering

About data, information, and knowledge

Types of databases



What's a database, really?

Database architecture and concepts

Database

Collection of related data

Miniworld or **universe of discourse** (UoD)

Represents **some aspect** of the real world

Logically coherent collection of data with inherent meaning

Built for a **specific purpose**

An Example

UNIVERSITY database

Information concerning students, courses, and grades in a university environment

Data records

STUDENT

COURSE

SECTION

GRADE_REPORT

PREREQUISITE

An Example

Database > Records > Data Items

'chalmers' > 'course' > course_name: 'Database'

'chalmers' > 'student' > student_number: 17



An Example

Database > Records > Data Items

'chalmers' > 'course' > course_name: 'Database'

'chalmers' > 'student' > student_number: 17



Meta-data



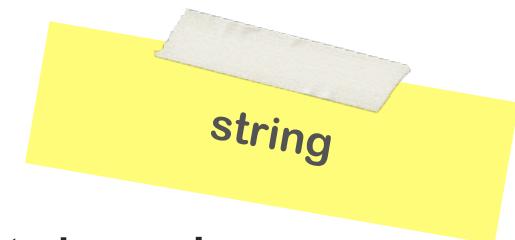
Data

An Example

Database > Records > Data Items

'chalmers' > 'course' > course_name: 'Database'

'chalmers' > 'student' > student_number: 17

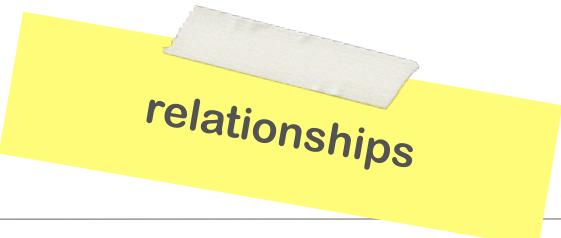


An Example

Database > Records > Data Items

'chalmers' > 'course' > course_name: 'Database'

'chalmers' > 'student' > student_number: 17  *Enrolled In*



relationships

Meta-data

“Data about data”

Data that is used to describe what the “actual” data is allowed to look like

Think Java classes and how they relate to object instances

Database Management Systems

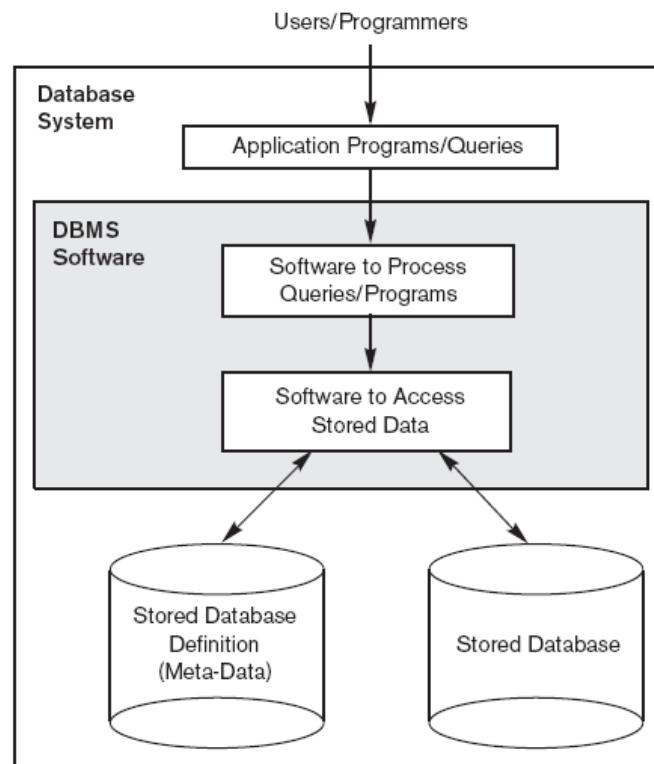


Figure 1.1
A simplified database system environment.

Advantages of (many) DBMS

Providing **persistent storage** for program objects

But: Impedance **mismatch problem**

Providing **efficient querying** capabilities (search)

Indexing / optimized data structures

Query optimization

Caching

Controlling **redundancy**

Data normalization

Denormalization

Advantages of (many) DBMS

Ensuring data integrity

- Data types

- Referential integrity

- Uniqueness constraints

Ensuring transactional properties

- “ACID” properties

Restricting unauthorized access

- Access control

Backup and recovery

ACID

Atomicity

All changes are applied, or nothing is applied

Consistency

Database is always in a consistent state (no “in-between” time)

Isolation

Concurrency control - guarantees that concurrent transactions are not interfering

Durability

Once a change is applied, it remains even through failures

What we will be covering

About data, information, and knowledge

Types of databases

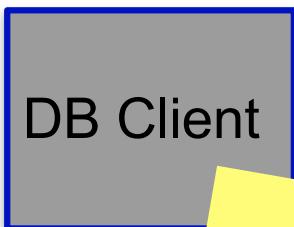
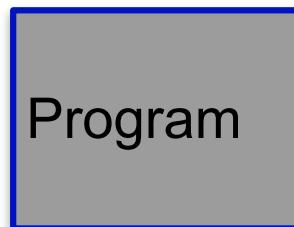
What's a database, really?



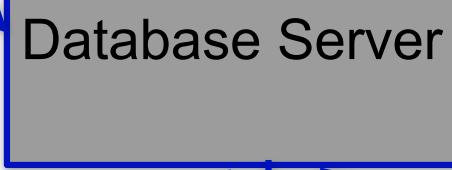
Database architecture and concepts

Client/Server Architecture

(diff. kinds of) **Clients**



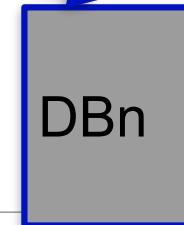
actual DBMS



different Databases

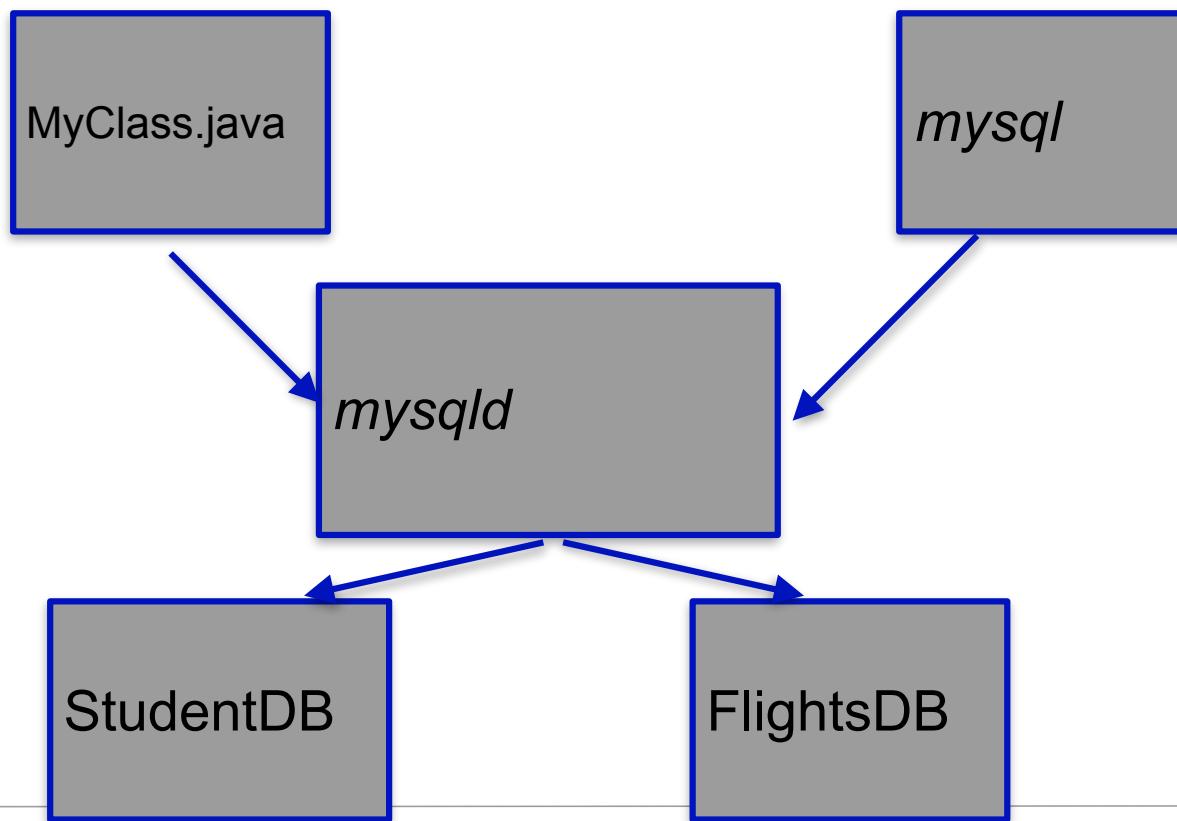


...



*nothing magical
about the cmdline DB
client - just another
program*

Example: MySQL



Roles in a Database World - User Side

Database Administrator

Manages the DBMS itself

Database Designer

Manages the meta-data (schema), but not the data itself

Database User

Developers writing programs using DBs

“Naive” end users (use pre-canned queries)

Sophisticated end users (write custom, interactive queries)

Roles in a Database World - Backend Side

Database Developers

Build the DBMS itself

Database Tool Developers

Responsible for building dev. tools, e.g., ORM libraries

Operators and Maintenance

Manage hardware and software that the DBMS is running on



We will focus on the user side!

Some Basic Concepts and Terminology

Database schema

Description of a database

Entities

“Things” that exist in the schema (students, courses, ...)

Attributes

Describe entities

Instances

Concrete objects matching an entity

Cp.: instances in OOP

Some Basic Concepts and Terminology

Schema diagram

Displays selected aspects of schema

Schema construct

Each object in the schema

Database state or snapshot

Data in database at a particular moment in time

Change in the data represented by series of snapshots

Schema Evolution

Some Basic Concepts and Terminology

Figure 2.1

Schema diagram for the database in Figure 1.2.

STUDENT

Name	Student_number	Class	Major
------	----------------	-------	-------

COURSE

Course_name	Course_number	Credit_hours	Department
-------------	---------------	--------------	------------

PREREQUISITE

Course_number	Prerequisite_number
---------------	---------------------

SECTION

Section_identifier	Course_number	Semester	Year	Instructor
--------------------	---------------	----------	------	------------

GRADE_REPORT

Student_number	Section_identifier	Grade
----------------	--------------------	-------

⁶Schema changes are usually needed as the requirements of the database applications change. Newer database systems include operations for allowing schema changes, although the schema change process is more involved than simple database updates.

⁷It is customary in database parlance to use *schemas* as the plural for *schema*, even though *schemata* is the proper plural form. The word *schema* is also sometimes used to refer to a schema.

Schema Evolution

In practice, not only the data but also the **schema** changes over time

E.g., a new version of your program uses some extra attributes

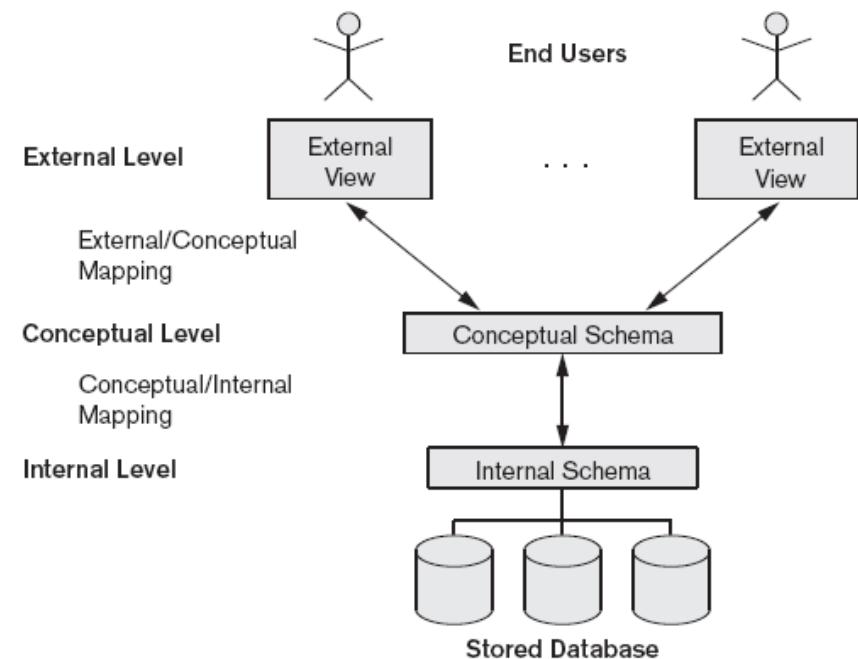
Schema Evolution

Difficult because old data needs to be **migrated**

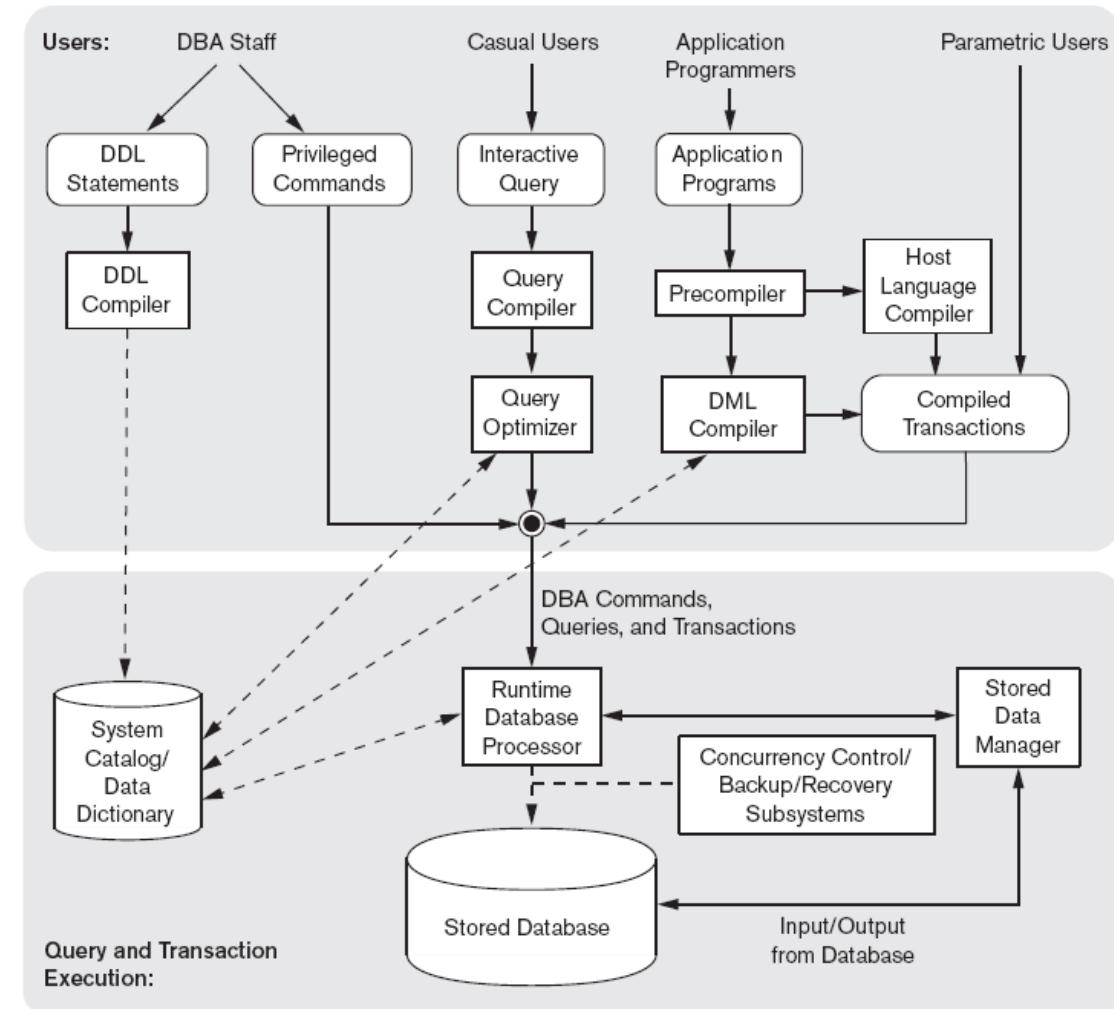
Three-Schema Architecture

ANSI/SPARC
Standard

Figure 2.2
The three-schema architecture.



A More Comprehensive View on a Relational Database



© **Figure 2.3**
 Component modules of a DBMS and their interactions.

A More Comprehensive View on a Relational Database

Different tools for different users

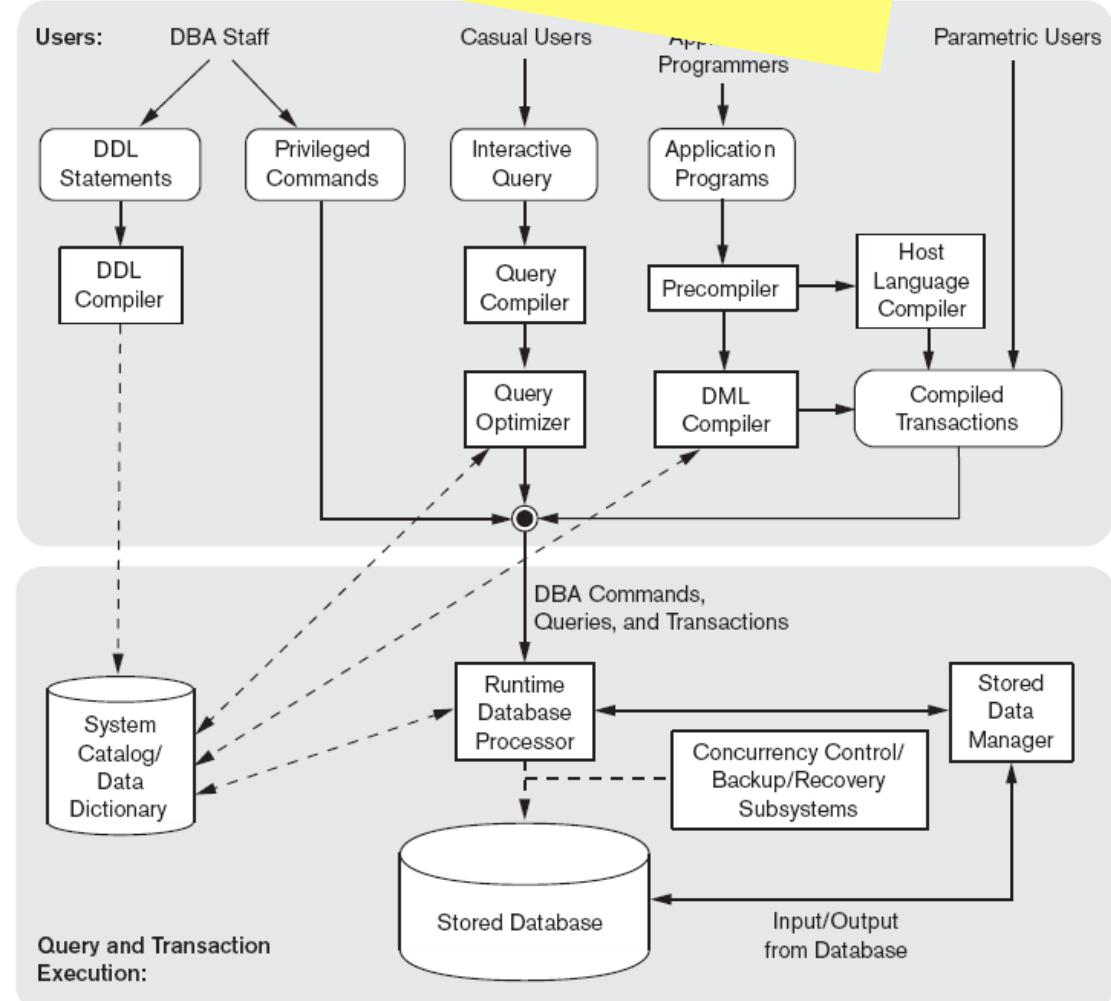


Figure 2.3
 Component modules of a DBMS and their interactions.

A More Comprehensive View on a Relational Database

Client/Server Architecture

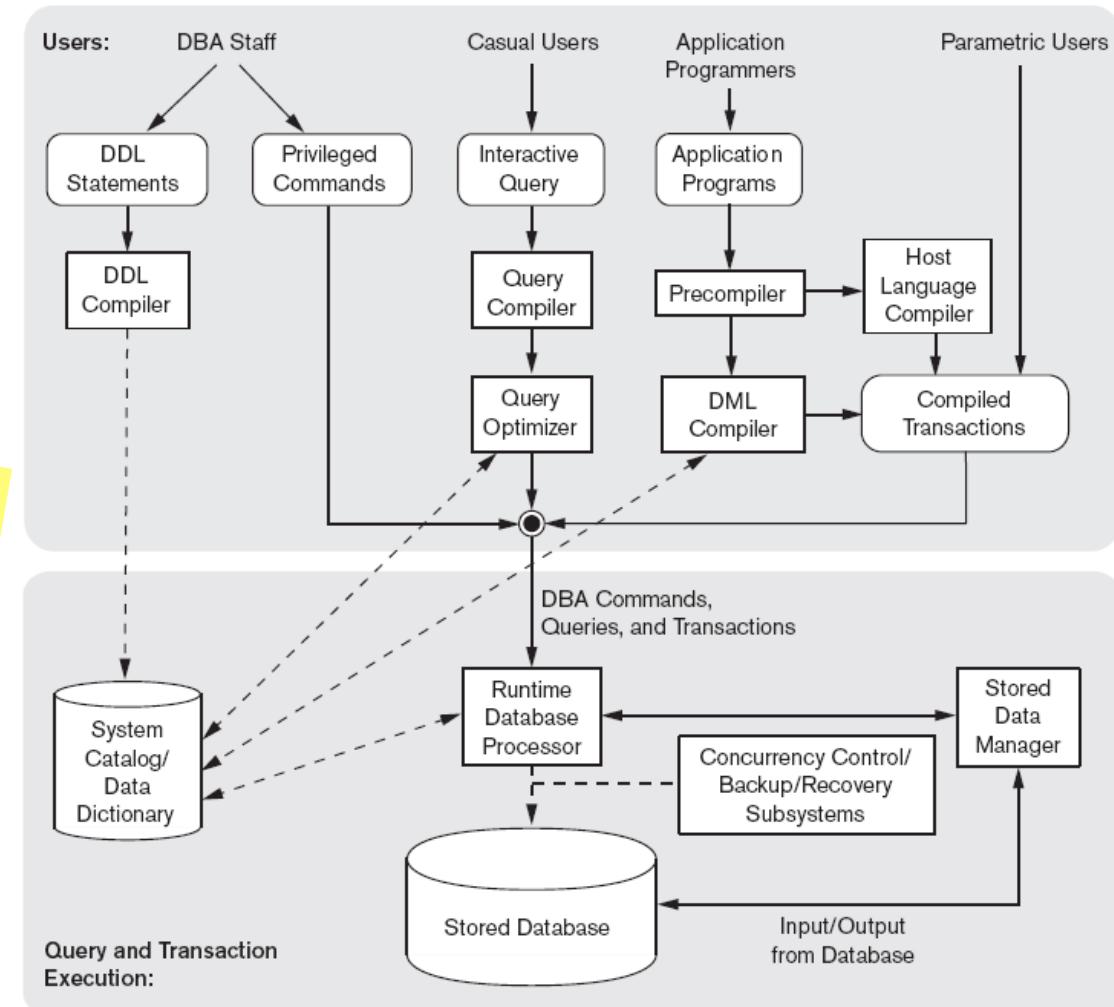


Figure 2.3
 Component modules of a DBMS and their interactions.

A More Comprehensive View on a Relational Database

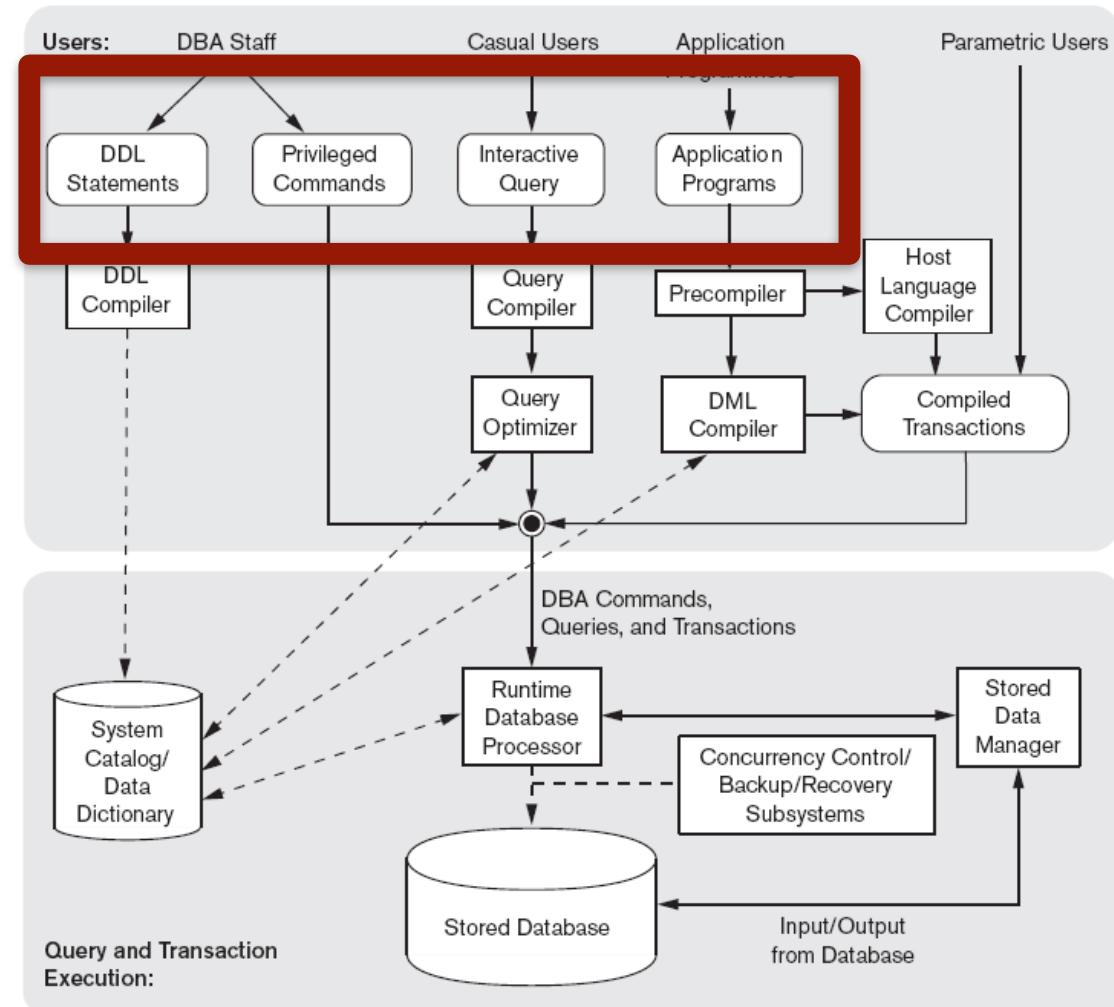


Figure 2.3
 Component modules of a DBMS and their interactions.

Interaction with the DBMS

Menu-based interfaces for Web clients or browsing

Forms-based interfaces

Graphical user interfaces (GUI)

Natural language interfaces

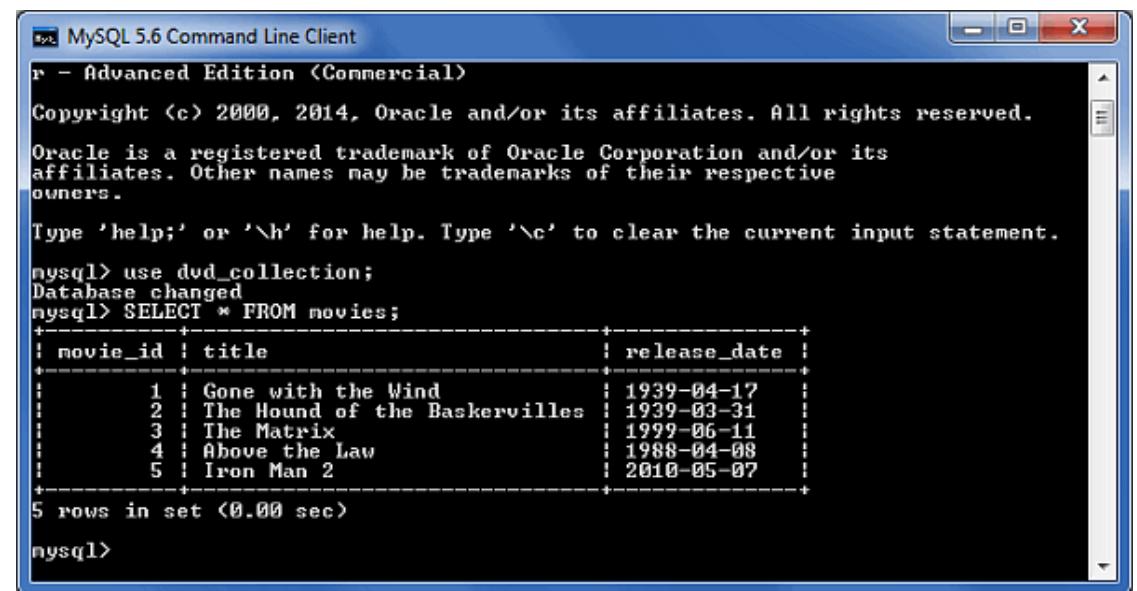
Speech input and output

Interfaces for parametric users

Interfaces for the DBA



This what we will use



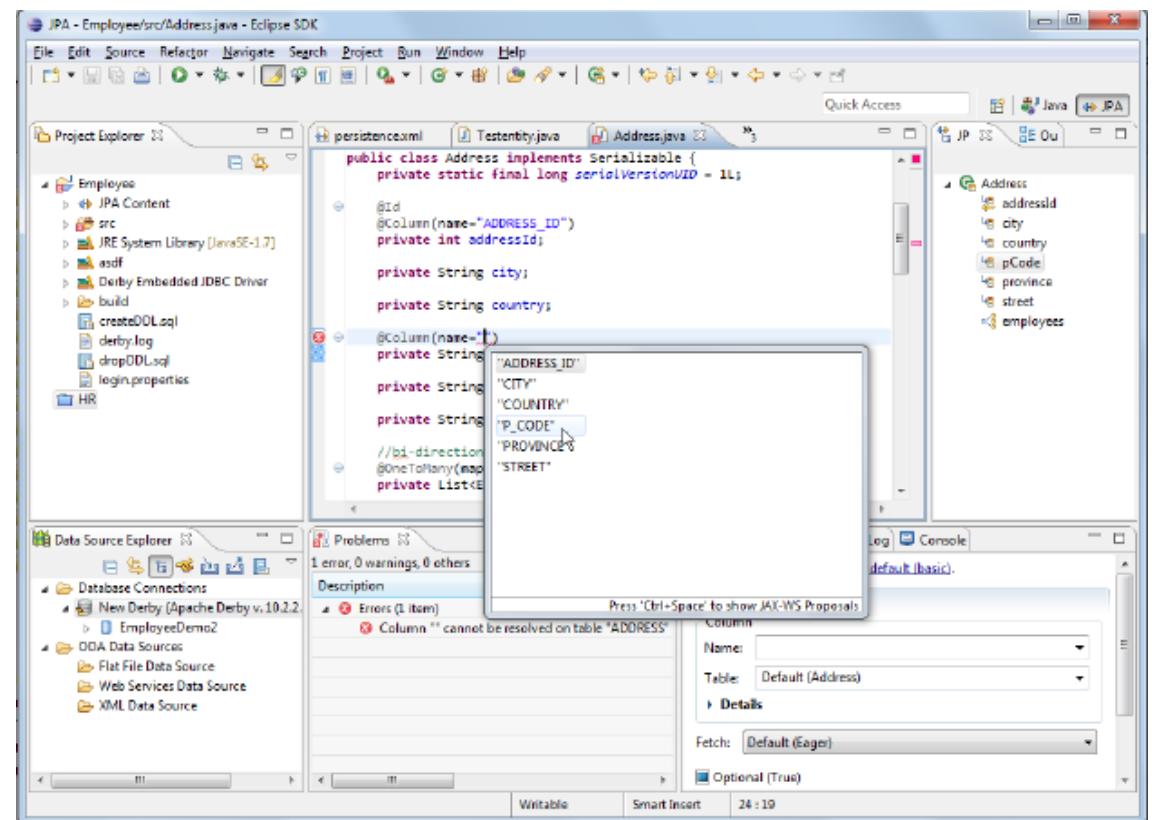
The screenshot shows a Windows command-line interface window titled "MySQL 5.6 Command Line Client - Advanced Edition (Commercial)". It displays the results of a SQL query:

```
Copyright (c) 2000, 2014, Oracle and/or its affiliates. All rights reserved.  
Oracle is a registered trademark of Oracle Corporation and/or its  
affiliates. Other names may be trademarks of their respective  
owners.  
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.  
mysql> use dvd_collection;  
Database changed  
mysql> SELECT * FROM movies;  
+-----+-----+  
| movie_id | title | release_date |  
+-----+-----+  
| 1 | Gone with the Wind | 1939-04-17 |  
| 2 | The Hound of the Baskervilles | 1939-03-31 |  
| 3 | The Matrix | 1999-06-11 |  
| 4 | Above the Law | 1988-04-08 |  
| 5 | Iron Man 2 | 2010-05-07 |  
+-----+-----+  
5 rows in set (0.00 sec)  
mysql>
```

Interaction with the DBMS

Over an API (e.g., Java)

We will do
that as well



Key Takeaways

Data, information, and knowledge

What ways do we have to visualize data?

Basic database operations

CRUD - create, read, update, delete

The importance of querying

Databases as **collections of meta-data and data**

The concept of “**Database Management Systems**”

Key Takeaways

Advantages of using a DBMS

Incl. relational integrity, ACID properties, powerful queries, ...

Basic database operations

CRUD - create, read, update, delete

The importance of querying

Roles in a database application

The ANSI/SPARC Three-Schema Architecture