DATA BASE MANAGEMENT SYSTEM BASIC CONCEPT

ERI PRASETYO

http://pusatstudi.gunadarma.ac.id/pscitra

http://eri.staffsite.gunadarma.ac.id

Sources :

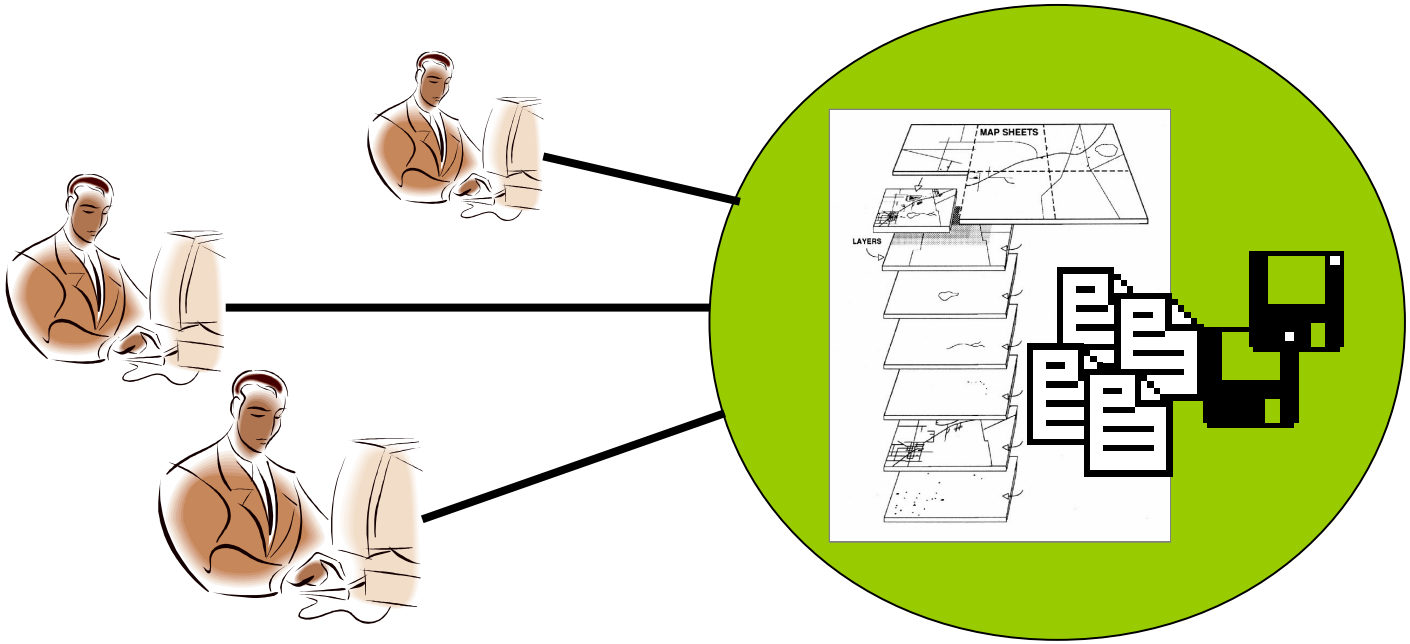
-Database Systems: Design, Implementation & management, 5th, edition, Rob & Coronel

- Database System Concept, Silberschatz, Korth and Sudarshan

- RDBMS untuk mendukung apalikasi web database, Achmad Maududie, UNEJ

Database

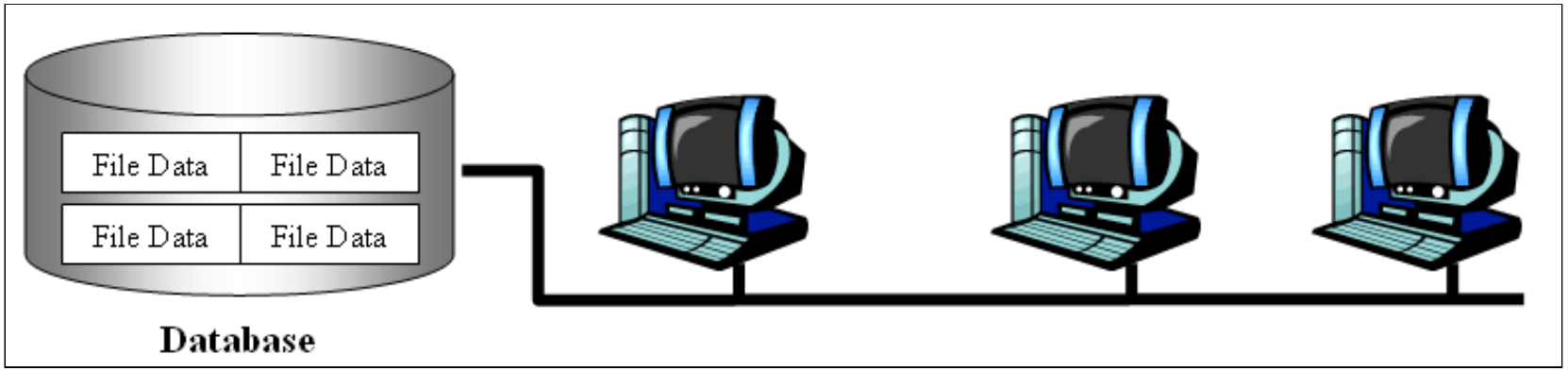
- Kumpulan data
- Tersimpan dalam suatu tempat
- Dapat digunakan secara bersama



Database

Secara digital

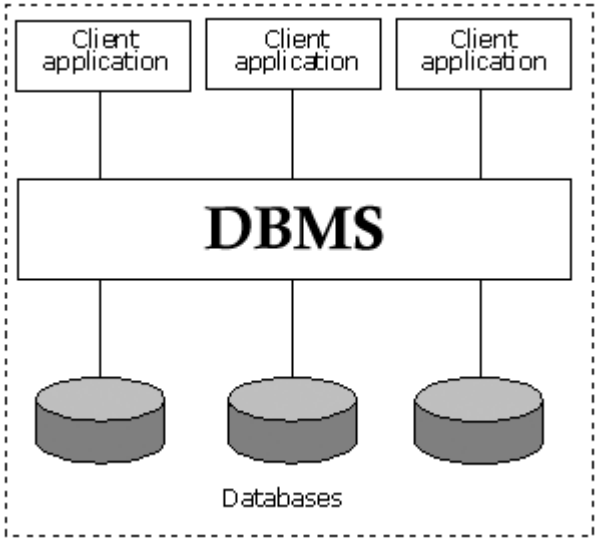
- tersimpan dalam bentuk file
- tersimpan dalam database server
- diakses secara bersama melalui jaringan komputer



Database Management Systems

Database Management System = DBMS

system basisdata yang memiliki kemampuan menajemen untuk menjamin ketersediaan, keamanan, reliabilitas, konsistensi dan validitas data



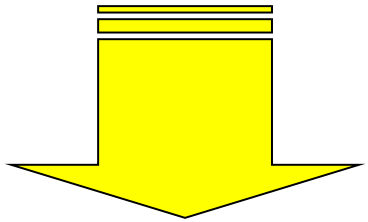
Tujuan DBMS

- Menghindari duplikasi data
- Memudahkan akses data
- Meningkatkan keamanan

Database Management System

Bagaimana cara mengorganisasikan data dalam database?

Salah satu metode yg digunakan

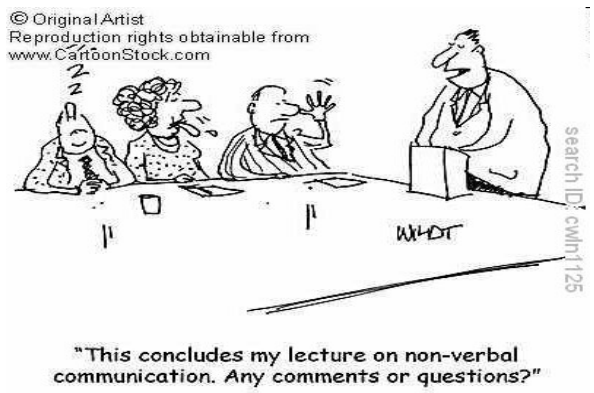


Menggunakan metode relasi yang didasarkan pada teori himpunan matematika

Database Management System

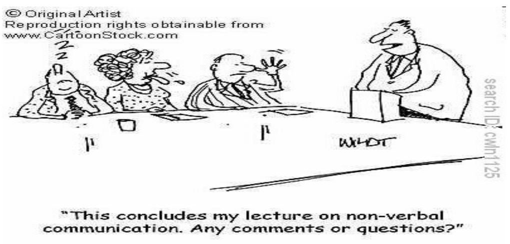
Himpunan

Kumpulan dari object yang berbeda (diskrit) yg digunakan untuk mengelompokkan sejumlah objek (yg disebut dg elemen, unsur atau anggota)

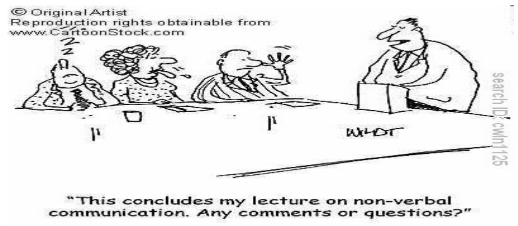


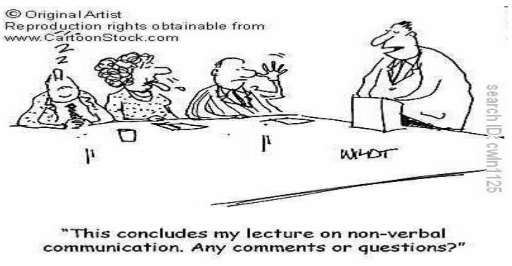
Database Management System

Himpunan



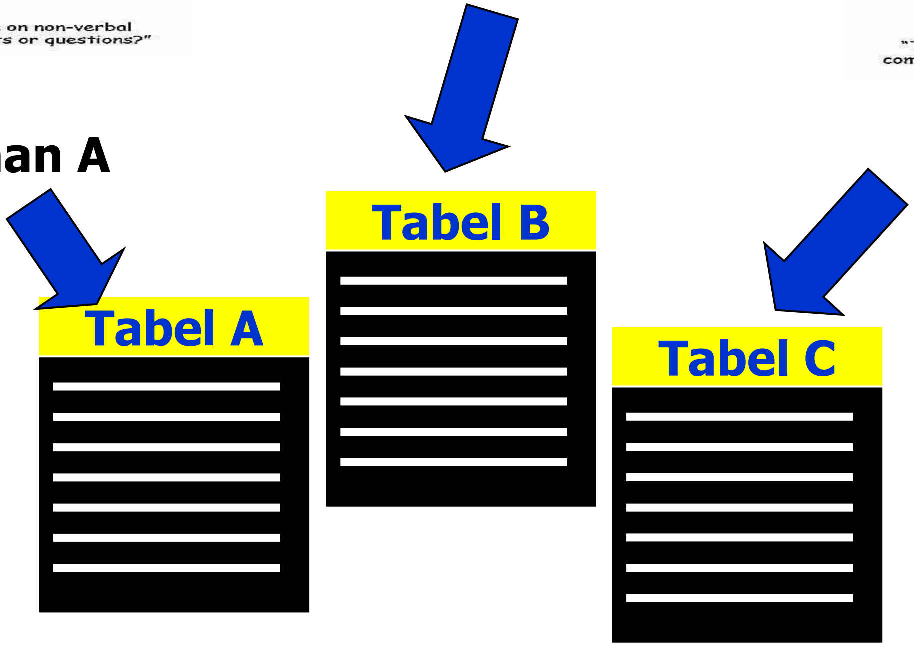
Himpunan B





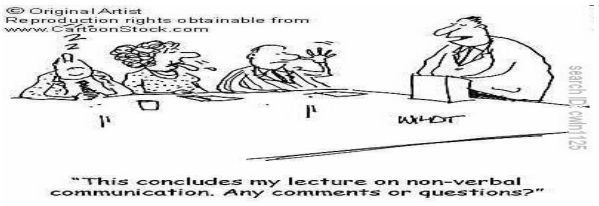
Himpunan A

Himpunan C



Masing-masing Tabel memiliki relasi dengan tabel lain

Database Management System



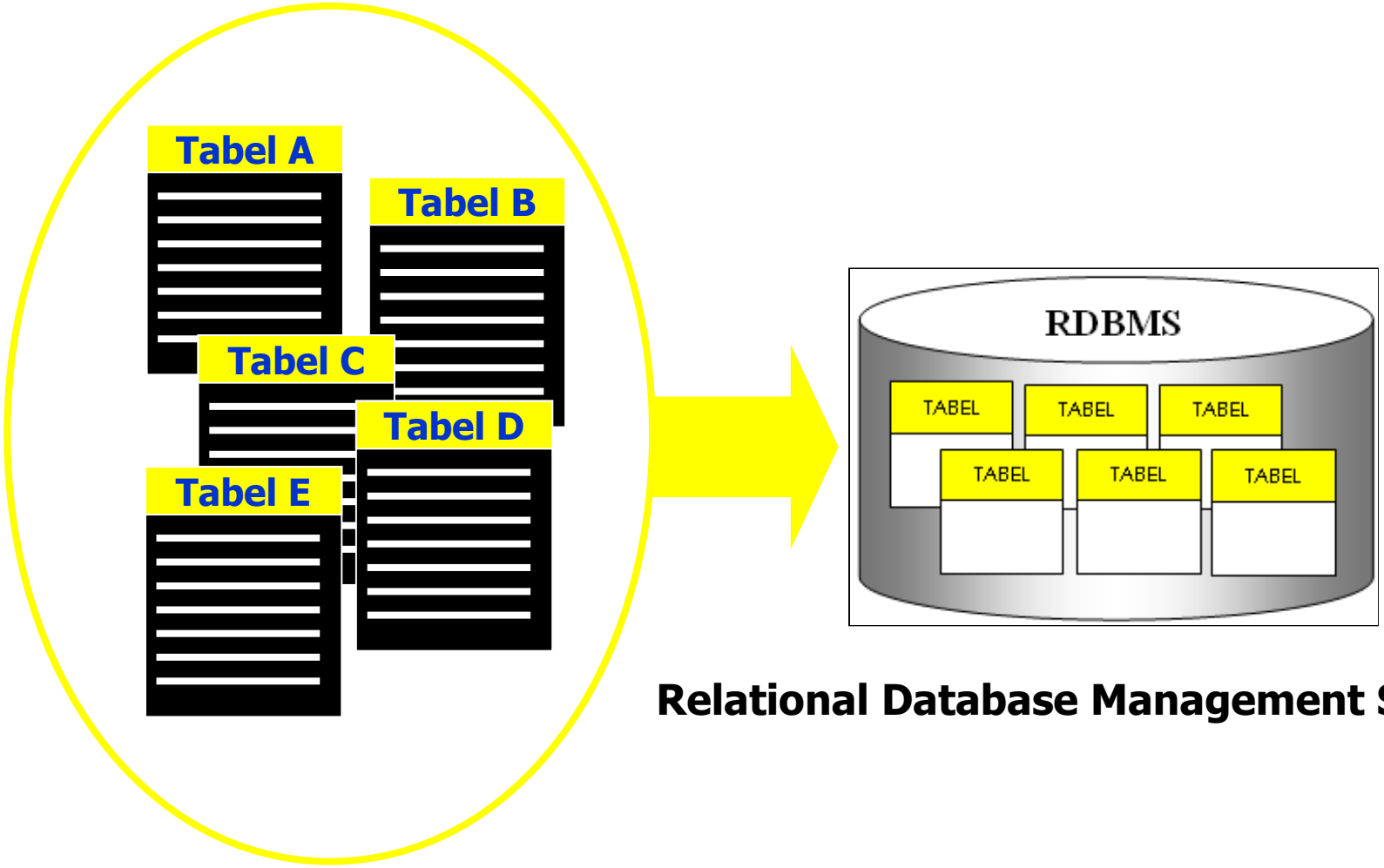
Himpunan Pekerja

Field

Employee number	Employee name	Rate category
11	Vincent Radebe	A
12	Pauline James	B
16	Charles Ramoraz	C
17	Monique Williams	B

Record

Database Management System



Relational Database Management System

Apa Yang Penting Didalam DBMS

- Menjadikan management data lebih efficient dan effective
- Bahasa query memungkinkan menjawab dengan cepat untuk query ad hoc
- Menyediakan akses yang lebih baik untuk pengelolaan data yang lebih besar
- Mengurangi kemungkinan in-konsistensi data

DBMS Manages Interaction

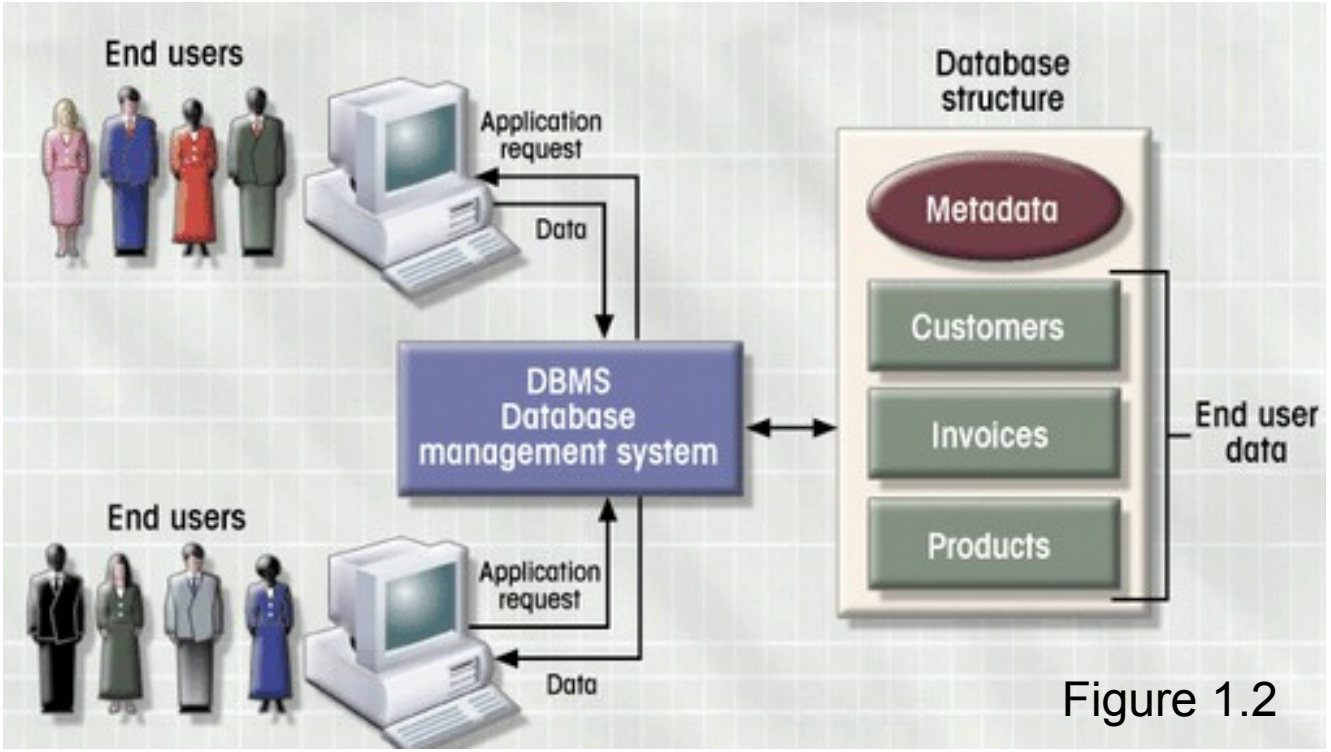


Figure 1.2

Where are RDBMS used ?

- Backend for traditional “database” applications
  - administration(Gunadarma)
- Backend for large Websites
  - Immosearch (http://www.robtex.com/dns/www.immosearch.ch.html)
- Backend for Web services
  - Amazon

Example of a Traditional Database Application

Suppose we are building a system

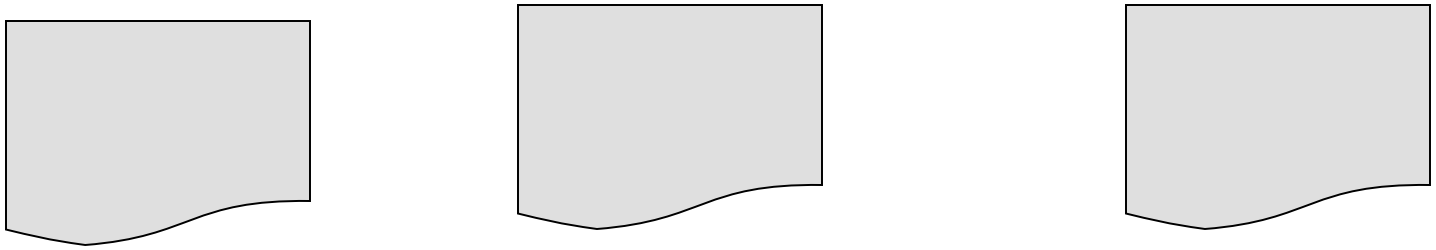
to store the information about:

- students
- courses
- professors
- who takes what, who teaches what

Can we do it without a DBMS ?

Sure we can! Start by storing the data in files:

students.txt courses.txt professors.txt



Now write C or Java programs to implement specific tasks

Doing it without a DBMS...

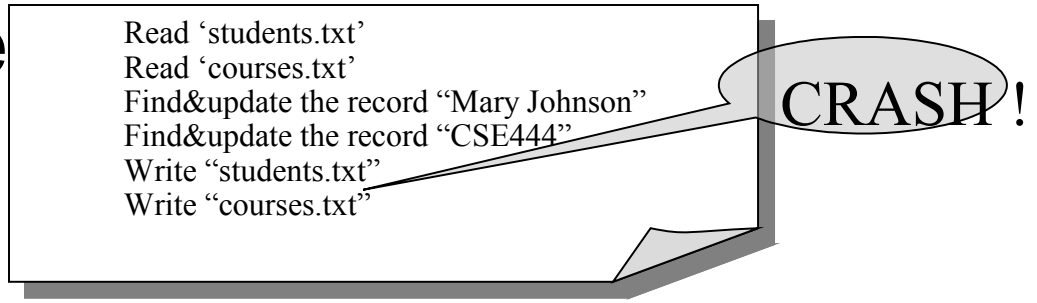
- Enroll “Mary Johnson” in “CSE444”:

Write a C/Java program to do the following:

Read ‘students.txt’
Read ‘courses.txt’
Find&update the record “Mary Johnson”
Find&update the record “CSE444”
Write “students.txt”
Write “courses.txt”

Problems without an DBMS...

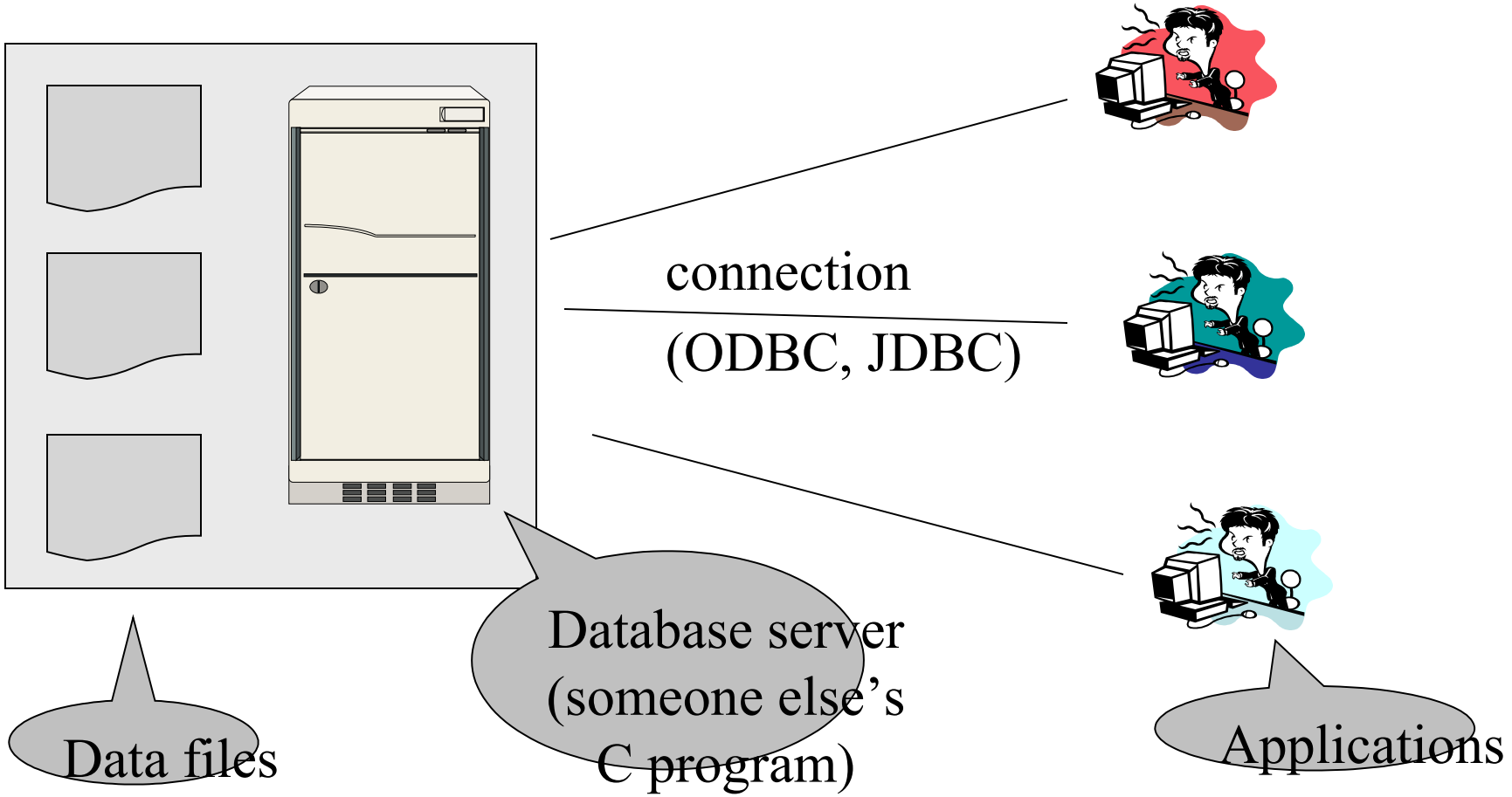
- System crashe



  - What is the problem ?
- Large data sets (say 50GB)
  - Why is this a problem ?
- Simultaneous access by many users
  - Lock students.txt – what is the problem ?

Enters a DBMS

"Two tier system" or "client-server"



Functionality of a DBMS

The programmer sees SQL, which has two components:

- Data Definition Language - DDL
- Data Manipulation Language - DML
  - query language

Behind the scenes the DBMS has:

- Query engine
- Query optimizer
- Storage management
- Transaction Management (concurrency, recovery)

How the Programmer Sees the DBMS

- Start with DDL to *create tables*:

```
CREATE TABLE Students (
        Name CHAR(30)
        SSN CHAR(9) PRIMARY KEY NOT NULL,
        Category CHAR(20)
)   . . .
```

- Continue with DML to *populate tables:*

```
INSERT INTO Students
VALUES('Charles', '123456789', 'undergraduate')
. . . .
```

How the Programmer Sees the DBMS

- Tables:

Students:

SSN	Name	Category
123-45-6789	Charles	undergrad
234-56-7890	Dan	grad
	…	…

Takes:

SSN	CID
123-45-6789	CSE444
123-45-6789	CSE444
234-56-7890	CSE142
	…

Courses:

CID	Name	Quarter
CSE444	Databases	fall
CSE541	Operating systems	winter

- Still implemented as files, but behind the scenes can be quite complex

"*data independence*" = separate *logical* view from *physical implementation*

Transactions

- Enroll “Mary Johnson” in “CSE444”:

```
BEGIN TRANSACTION;

INSERT INTO Takes
    SELECT Students.SSN, Courses.CID
    FROM Students, Courses
    WHERE Students.name = ‘Mary Johnson’ and
                Courses.name = ‘CSE444’

-- More updates here....

IF everything-went-OK
     THEN COMMIT;
ELSE ROLLBACK
```

If system crashes, the transaction is still either committed or aborted

Transactions

- A *transaction* = sequence of statements that either all succeed, or all fail
- Transactions have the ACID properties:

  A = atomicity (a transaction should be done or undone completely)

  C = consistency (a transaction should transform a system from one consistent state to another consistent state)

  I = isolation (each transaction should happen independently of other transactions)

  D = durability (completed transactions should remain permanent)

Queries

- Find all courses that “Mary” takes

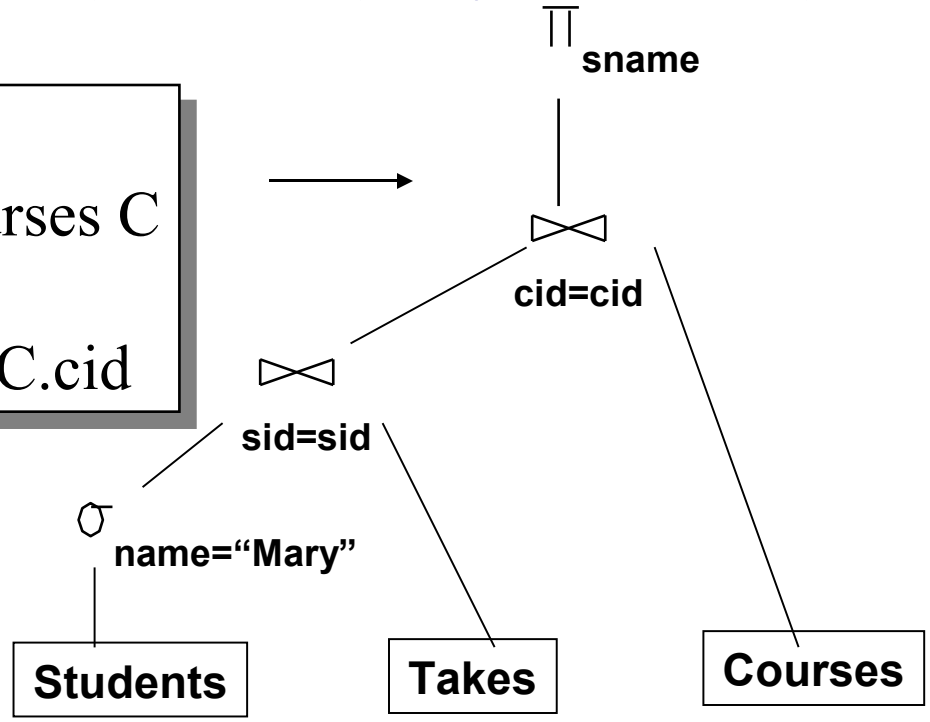
```
SELECT  C.name
FROM    Students S, Takes T, Courses C
WHERE   S.name=“Mary” and
        S.ssn = T.ssn and T.cid = C.cid
```

- What happens behind the scene ?
  - Query processor figures out how to answer the query efficiently.

Queries, behind the scene

Declarative SQL query → ***Imperative query execution plan:***

```
SELECT  C.name
FROM Students S, Takes T, Courses C
WHERE S.name="Mary" and
        S.ssn = T.ssn and T.cid = C.cid
```



The **optimizer** chooses the best execution plan for a query

Database Systems

- The big commercial database vendors:
  - Oracle
  - IBM (with DB2)
  - Microsoft (SQL Server)
  - Sybase
- Some free database systems (Unix) :
  - Postgres
  - MySQL
  - Predator

Databases and the Web

- Accessing databases through web interfaces
  - Java programming interface (JDBC)
  - Embedding into HTML pages (JSP)
  - Access through http protocol (Web Services)
- Using Web document formats for data definition and manipulation
  - XML, Xquery, Xpath
  - XML databases and messaging systems

Database Integration

- Combining data from different databases
  - collection of data (wrapping)
  - combination of data and generation of new views on the data (mediation)
- Problem: heterogeneity
  - access, representation, content
- Example revisited
  - http://immo.search.ch/
  - http://www.swissimmo.ch

Other Trends in Databases

- Industrial
  - Object-relational databases
  - Main memory database systems
  - Data warehousing and mining
- Research
  - Peer-to-peer data management
  - Stream data management
  - Mobile data management