SE240: Introduction to Database Systems

Lecture 01:

Introduction

Overview

- **Database & Database Systems**
- Data Models
- Application & History

What is a Database?

- A collection of data, typically describing the activities of one or more related organizations.
- Models real-world enterprise
  - Entities
    - e.g. students, professors, courses, classrooms
  - Relationships between entities
    - e.g. student's enrollment in courses, professor teaching courses, and use of room for courses.

What is a Database?

- Databases play a critical role in almost all areas
  - **Banking:** all transactions
  - **Airline:** reservation, schedules
  - **Universities:** registration, grades
  - **Sales:** customers, products, purchases
  - **Manufacturing:** production, inventory, orders, supply chain
  - **Human resources:** employee records, salaries, tax deductions

What is a Database?

- A database can be of **any size** and of **varying complexity**.
  - For example, the list of names and address of friends
  - The book catalog of a large library may contain half a million records
  - A database of much greater size and complexity is maintained by the government to keep track of the tax information filed by taxpayers.

What is a Database?

Student Name	ID	Age	Gender	Entrance Year	Grade
Chan Mei Yee	A34455	20	F	1998	A
Lee Wai Man	C23444	19	M	1999	B
Wong Wing Nam	C73334	19	M	2000	C

- A **schema** is the ***definition*** of a database. It defines the ***meaning*** of data.
- An **instance** of a database is ***the collection of data*** in the database at a particular point of time (snapshot).
- For example, in the above, the schema is "*Student Name, ID, Age, Gender, Entrance Year, Grade*". The remaining rows in the table make up an instance of the database.

What is a DBMS?

- **DBMS – Database Management System**
- A DBMS is a collection of ***software programs*** to enable users to ***create***, ***maintain*** and ***utilize*** a ***database***.

What is a DBMS?

Student Name	ID	Age	Gender	Entrance Year	Grade
Chan Mei Yee	A34455	20	F	1998	A
Lee Wai Man	C23444	19	M	1999	B
Wong Wing Nam	C73334	19	M	2000	C

- DBMS
  - Insert records
  - Delete records
  - Update records
  - Query records

What is a DBMS?

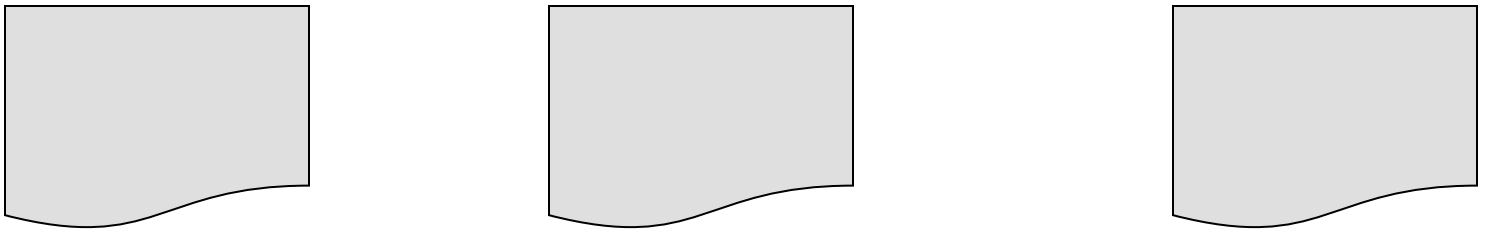
- Commercial DBMS

Company	Product
Oracle	Oracle 8i, 9i, 10g
IBM	DB2, Universal Server
Microsoft	Access, SQL Server
Sybase	Adaptive Server
Informix	Dynamic Server

Can we do without a DBMS?

- Sure! Start by storing the data in files:

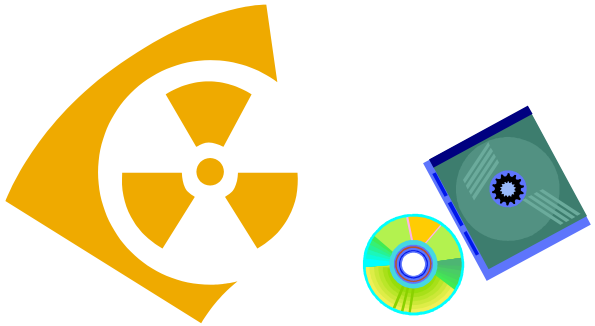
student.txt course.txt teacher.txt



- Now write C or Java programs to implement specific tasks…

Why do we need a DBMS?

- To reduce application development time
- Suppose we are given a collection of raw files which occupy 500GB
- Need to store in disk or tape, and bring relevant parts into main memory for processing as needed.



Why do we need a DBMS?

1. Data redundancy and inconsistency
   - E.g., consider a bank application
     - *Address of a customer* in
       - the file of “saving-accounts” and
       - the file of “checking-accounts”

A good design of DBMS can avoid data redundancy and inconsistency.

Why do we need a DBMS?

2. Difficulty in accessing data

- Need to write a *new program* to carry out each *new task*

Student Name	ID	Age	Gender	Entrance Year	Grade
Chan Mei Yee	A34455	20	F	1998	A
Lee Wai Man	C23444	19	M	1999	B
Wong Wing Nam	C73334	19	M	2000	C

It is easy to obtain data with DBMS

Why do we need a DBMS?

3. Integrity problems

- E.g., consider a bank application
  - The balance *cannot be* below $1000
  - The day of a month *cannot exceed* 31

DBMS can check the integrity automatically

Why do we need a DBMS?

4. Atomicity of updates

- E.g., consider a bank application
  - We want to transfer $100 from account A to account B
  - Steps:
    - **Step 1:** We deduct $100 from account A
    - **Step 2:** Then, we increment $100 in account B
  - If the system *crashes* at Step 1, then Step 2 cannot be executed

DBMS makes sure that Step 1 and Step 2 can be executed together even with a crash (We call the execution is **atomic**.)

Why do we need a DBMS?

5. Concurrent Access by multiple users

- Uncontrolled concurrent accesses can lead to inconsistencies
- E.g., consider a bank application
  - There is an account shared by 2 customers A and B
  - Customers A and B withdraw $1000 concurrently

A	B
Read 5000	
	Read 5000
5000 - 1000	
	5000 - 1000
Write 4000	
	Write 4000

DBMS makes sure that the concurrent access cannot lead to this problem

Why do we need a DBMS?

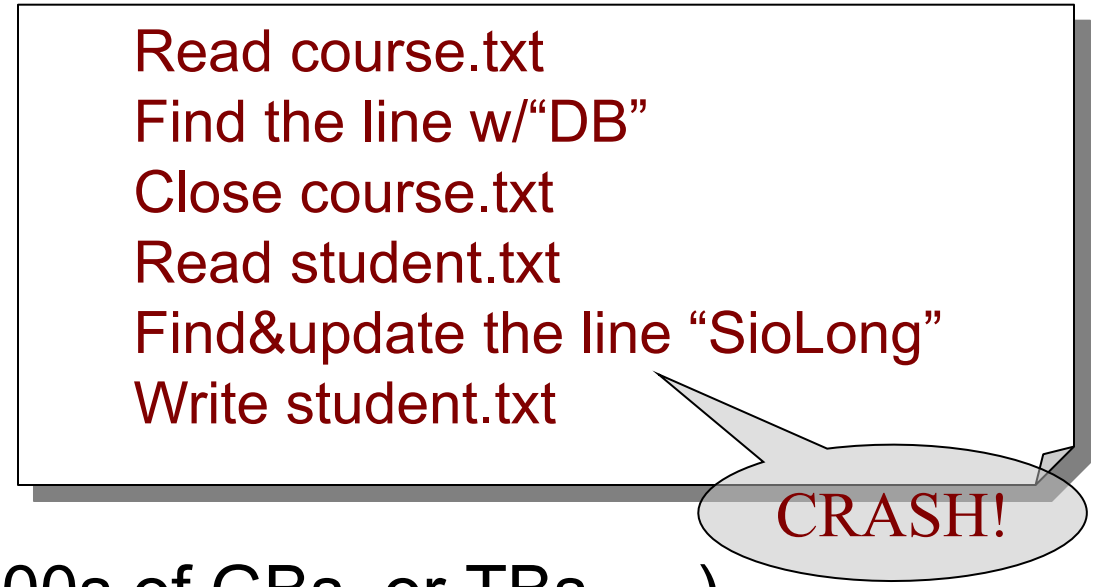
6. Security Problems

- E.g., consider a bank application
  - We do not want system programmers to have permissions to read some data (e.g., Andy Lau's saving account and Joey Yung's saving account)
- Need a lot of effort to re-write a program for this permission system

DBMS can enforce that different users have different permissions to access different parts of the data

Other problems without an DBMS...

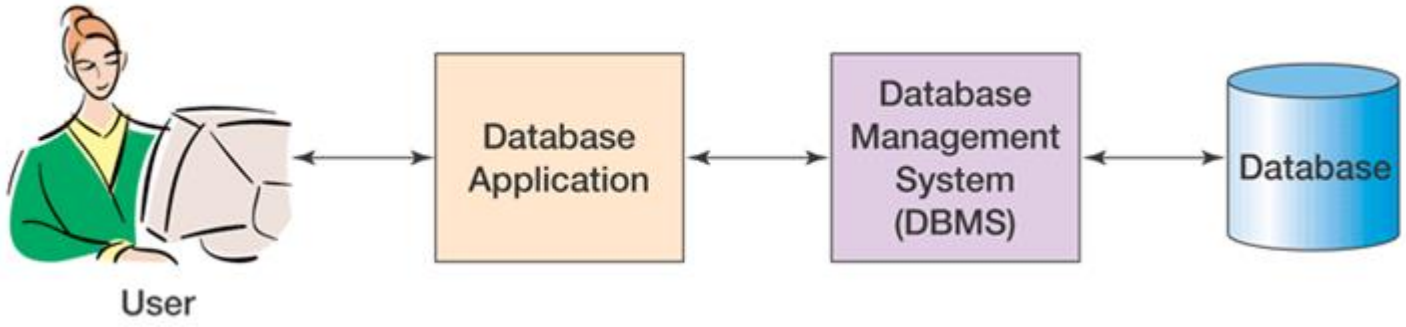
7. System crashes:



8. Large data sets (100s of GBs, or TBs, …)
   - No indices
     - Finding “SioLong” in huge flatfile is expensive
   - Modifications intractable without better data structures
9. Application programming interface (API)?
   - Interfaces, interoperability

Advantages of DMBS

- With the use of DBMS, we have the following advantages
  - Data independence
  - Efficient data access
  - Data integrity and security
  - Data administration
  - Concurrent access and crash recovery
- **Overall:** Reduced application development time

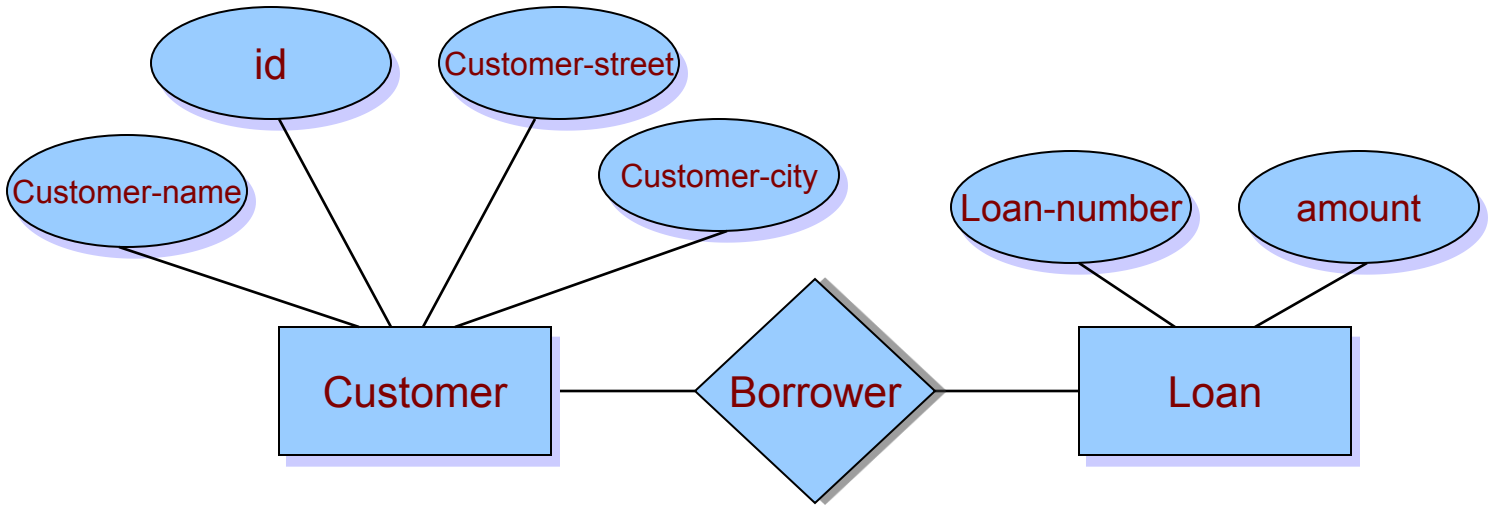


Overview

- Database & Database Systems
- **Data Models**
- Application & History

Data Models

- A data model is a collection of tools or concepts for describing data, the meaning of data, data relationships and data constraints.
- Object-based Logical Models
  - **Entity-Relationship Model (ER Model)**



Data Models

- Record-based Logical Models
  - **Relational Model**

Customer-Name	ID	customer-street	customer-city

Main concept: **relation**, basically a table with rows and columns. A column is also called a **field** or **attribute**

  - Other models such as the Network Model, Hierarchical Model, Objected-relational Model

We will focus on the dominant ***Relational model***.

A description of data in terms of a data model is called a **schema**.

Data Abstraction

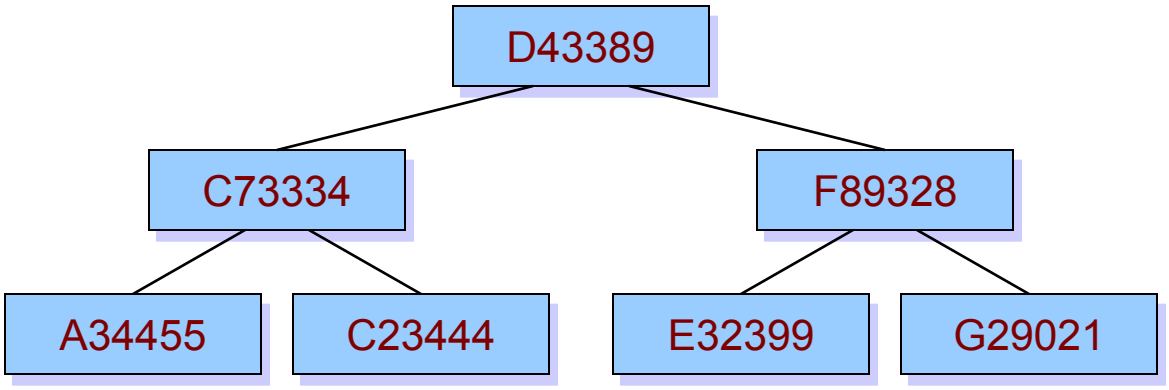
- Hide certain details of how data is stored and maintained
  - **Physical level:** how and where data are actually stored, low level data structures are specified at this level
  - **Conceptual level (logical level):** describes what data should be stored in the database, and relationship and semantics of the data
  - **View level:** Relevant partial view of the database to be particular type of users

Data Abstraction

- **Conceptual Level**

Student Name	ID	Age	Gender	Entrance Year	Grade
Chan Mei Yee	A34455	20	F	1998	A
Lee Wai Man	C23444	19	M	1999	B
Wong Wing Nam	C73334	19	M	2000	C
Cheung Nam	E32399	21	F	1998	B+
Fung Yu Siu	D43389	22	M	1997	B-
Chau Man Yi	G29021	18	M	2000	C
Yeung Chi	F89328	19	F	1999	B

- **Physical** arrangement of records by an index tree



Database Languages

- **Data Definition Language (DDL)**
  - A language that specifies data schemas
- **Data Manipulation Language (DML)**
  - A language to facilitate the retrieval, update of data in the database
- For retrieval, we query the database with the **query language**, which is part of the DML

What languages does the computer speak?

- Start with SQL DDL to *create tables*:

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

- Continue with SQL to *populate tables:*

```
INSERT INTO Students
VALUES('Hillary', '123456789', 'undergraduate');
```

Querying: Structured Query Language

- Find all the students who have taken CS101:

```
SELECT SSN
FROM   Takes
WHERE  CID='CS101';
```

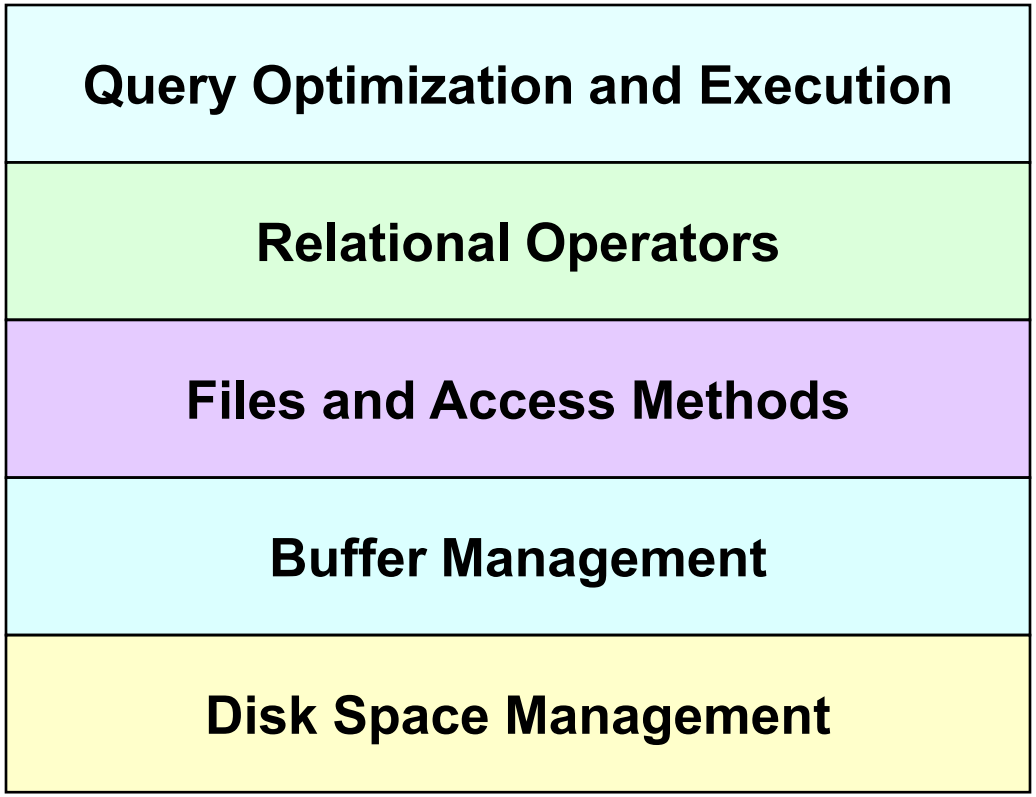
- Find all the students who CS101 *previously*:

```
SELECT SSN
FROM   Takes
WHERE  CID='CS101' AND Semester='2010';
```

- Find the students' *names*:

```
SELECT Name
FROM   Students, Takes
WHERE  Students.SSN = Takes.SSN AND
       CID='CS101' AND Semester='2010';
```

Structure of a DBMS



Overview

- Database & Database Systems
- Data Models
- **Application & History**

People who Deal With Databases

- **Database Administrator (DBA):** Person(s) who has central control over the database and is responsible for the following tasks:
  - Schema definition/modification
  - Storage structure definition/modification
  - Authorization of data access
  - Integrity constraint specification
  - Monitoring performance
  - Responding to changes in requirements

People who Deal With Databases

- **Application Programmers**
  - Embed DML calls in program written in a host language (e.g., Cobol, C, Java). (DML stands for data manipulation language)
  - e.g., programs that generates payroll checks, transfer funds between accounts
- **Sophisticated Users**
  - Form request in database query language
- **Naïve users**
  - Invokes one of the permanent application programs that have been written previously
  - e.g. transfer – transfer fund between accounts

Types of Databases and Database Applications

- Traditional Applications:
  - Numeric and Textual Databases
- More Recent Applications:
  - Multimedia Databases
  - Geographic Information Systems (GIS)
  - Data Warehouses
  - Real-time and Active Databases
  - Many other applications

History

- The first DBMS was designed by Bachman at GE in early 1960s
- In 1970 Codd at IBM proposed a new data representation framework called the **relational data model**.
- The SQL query for relational databases, developed as part of IBM's System R project, was standardized in the late 1980s.
- The current standard, SQL-92, was adopted by ANSI (American National Standards Institute) and ISO (International Standards Organization).

History

- In late 1980s and 1990s, several vendors (e.g., IBM's DB2, Oracle 8) have extended their systems with the ability to store new data types such as images and text.
- Specialized systems developed for **data warehouses**, consolidating data from several databases.
- Entering the Internet Age, a new markup language **XML** is proposed for data access through a Web browser.
- As more and more data are collected, companies are also interested to **mine** useful information from their data.

History of Database Processing

Timeframe	Technology	Remarks
Pre–1968	File Processing	Predecessor of database processing. Data maintained in lists. Processing characteristics determined by common use of magnetic tape medium.
1968–1980	Hierarchical and network models	Era of non-relational database processing. Prominent hierarchical data model was DL/I, part of IBM's first DBMS called IMS. Prominent network data model was CODASYL DBTG model; IDMS was most popular network DBMS.
1980 to present	Relational data model	Relational data model, first published in 1970; began to see commercial application in 1980. IBM endorsed it with DB2; other vendors followed by modifying their DBMS products or by creating new ones. Oracle achieved prominence. SQL became standard relational language.
1982	First microcomputer DBMS products	Ashton-Tate developed dBase products; Microrim created R:Base; Borland followed with Paradox.
1985	Interest in object-oriented DBMS (OODBMS) develops	With advent of object-oriented programming, OODBMS were proposed. Little success commercially, primarily because advantages did not justify the cost of converting billions of bytes of organizations' data to new format. Under development today.
1991	Microsoft ships Access	Personal DBMS created as element of Windows. Gradually supplanted all other personal DBMS products.
1995	First Internet database applications	Databases become key component of Internet applications. Popularity of the Internet greatly increases need and demand for database expertise.
1997	XML applied to database processing	Use of XML solves long-standing database problems. Major vendors begin to integrate XML into DBMS products.

Summary

- What is a Database?
- What is a DBMS?
- Why do we need a DBMS?
- Data models, data abstraction and Data Independence
- DBMS languages and people with DBMS
- Read Chapter 1