CSCI4333 Database Design & Implement

Lecture Two – Intro to Database (cont.)

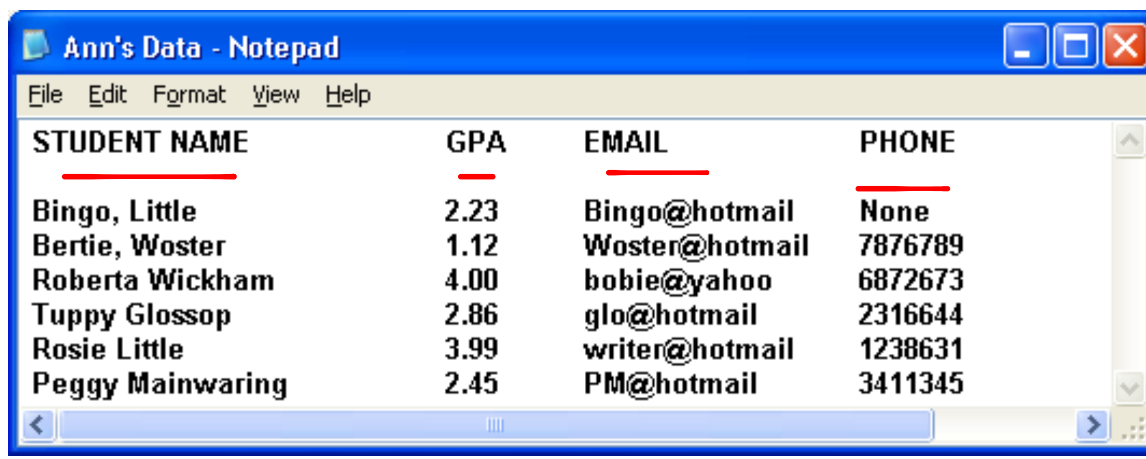
Instructor: Dr. Yifeng Gao

Why do we need database management systems?

• A **Database Management System** (DBMS) is a tool that allows us to store, modify, and query data.

However, I can store, modify and query data in a text file!

What can a DBMS do that I can't do with my text file solution?



STUDENT NAME	GPA	EMAIL	PHONE
Bingo, Little	2.23	Bingo@hotmail	None
Bertie, Woster	1.12	Woster@hotmail	7876789
Roberta Wickham	4.00	bobie@yahoo	6872673
Tuppy Glossop	2.86	glo@hotmail	2316644
Rosie Little	3.99	writer@hotmail	1238631
Peggy Mainwaring	2.45	PM@hotmail	3411345

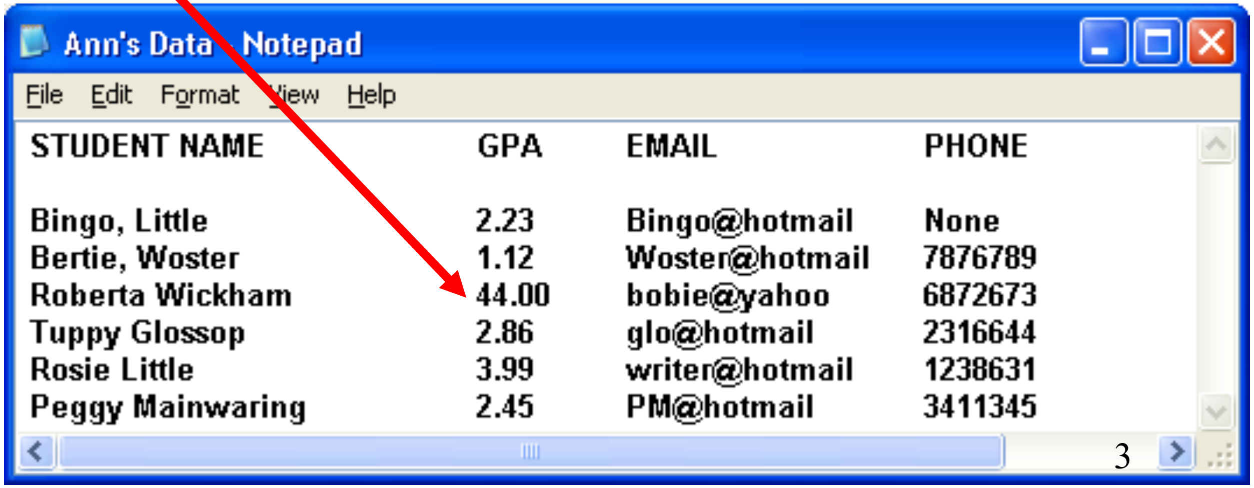
A simple solution to manage data:- stick them all in a text file!

Enforcing Constraints

• With the text file solution, there is no way to enforce integrity constraints on the data. In other words people can put bad data into the text file.

• In contrast, a DBMS allows us to enforce all kinds of constraints. This really helps (but does not guarantee) that our data is correct.

A typo gives Roberta Wickham a GPA of 44.00

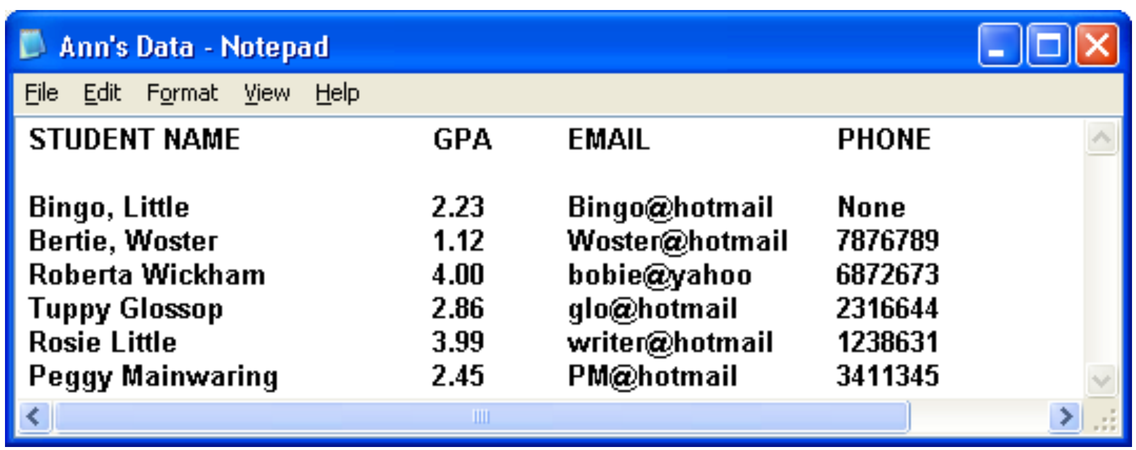


STUDENT NAME	GPA	EMAIL	PHONE
Bingo, Little	2.23	Bingo@hotmail	None
Bertie, Woster	1.12	Woster@hotmail	7876789
Roberta Wickham	44.00	bobie@yahoo	6872673
Tuppy Glossop	2.86	glo@hotmail	2316644
Rosie Little	3.99	writer@hotmail	1238631
Peggy Mainwaring	2.45	PM@hotmail	3411345

Scalability

• The text file solution might work for small datasets. What happens when we have big datasets?

• Most real world datasets are so large that we can only have a small fraction of them in main memory at any time, the rest has to stay on disk.

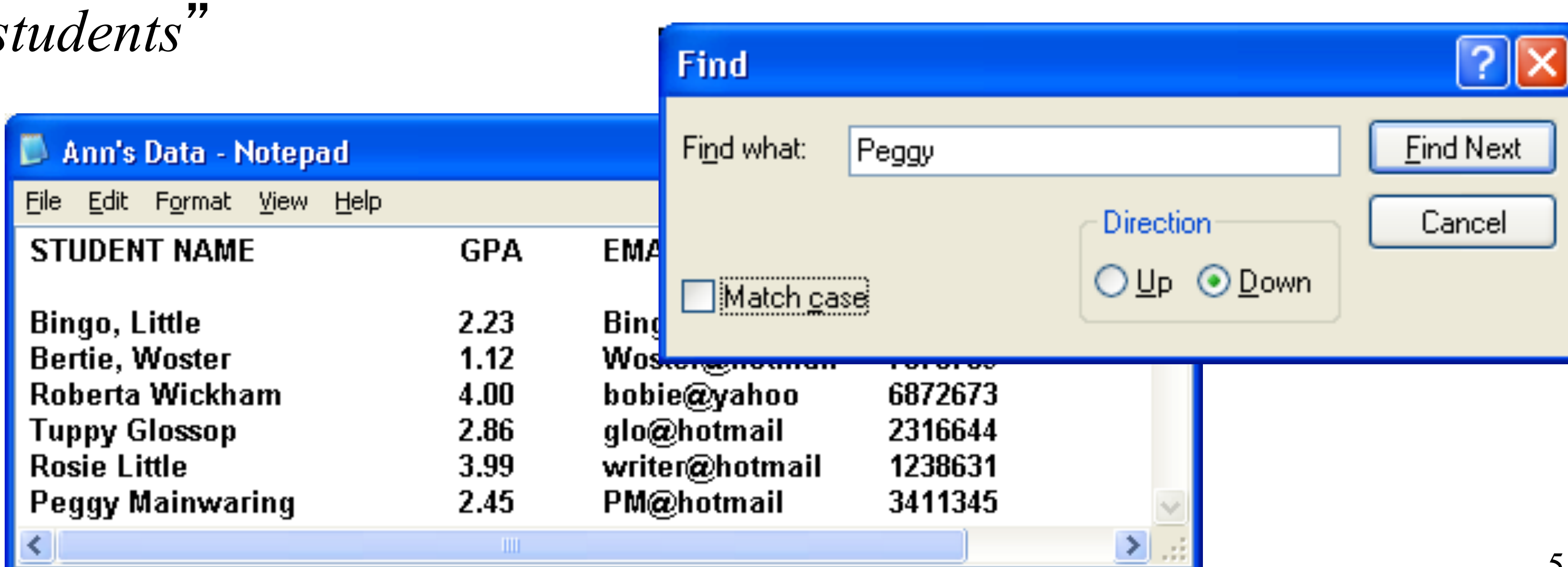

Ann's Data - Notepad
File Edit Format View Help

STUDENT NAME	GPA	EMAIL	PHONE
Bingo, Little	2.23	Bingo@hotmail	None
Bertie, Woster	1.12	Woster@hotmail	7876789
Roberta Wickham	4.00	bobie@yahoo	6872673
Tuppy Glossop	2.86	glo@hotmail	2316644
Rosie Little	3.99	writer@hotmail	1238631
Peggy Mainwaring	2.45	PM@hotmail	3411345

Query Expressiveness

- The text file solution would allow me to search for keywords or certain numbers (slowly).

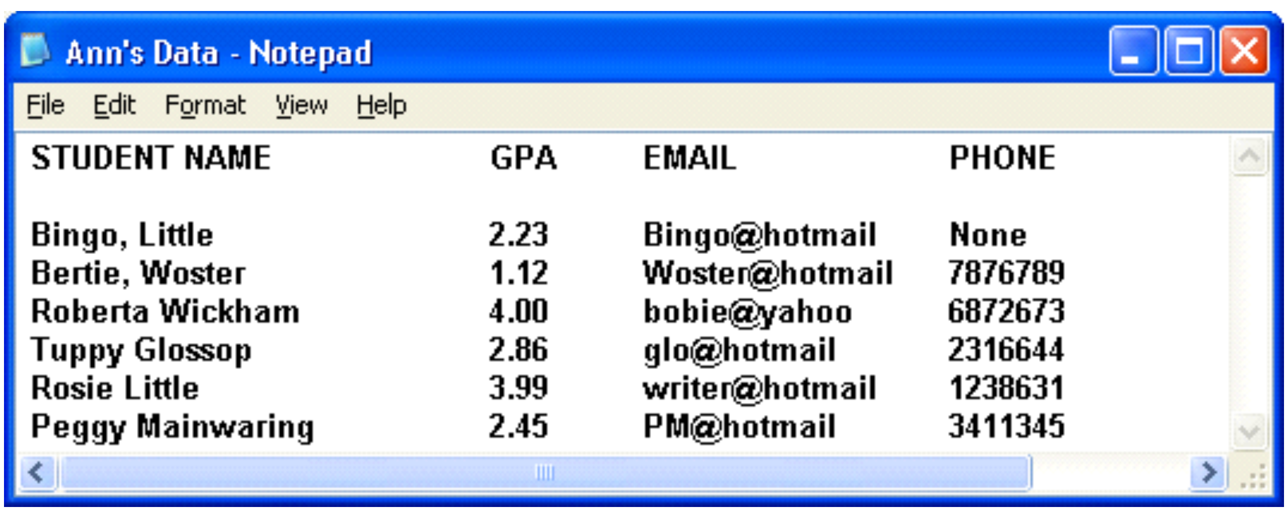
- With a DBMS I can search with much more expressive queries. For example I can ask.. "*Find all students whose GPA is greater than 2.5, and who don't own a phone*" or "*what is the average GPA of the students*"



Different Views

• The text file solution only allows one view of the data.

• With a DBMS I can arrange for different people to have different views of the data. For example, I can see everything, a student can see only his/her data, the TA can see data for students in his/her section, etc.


Ann's Data - Notepad

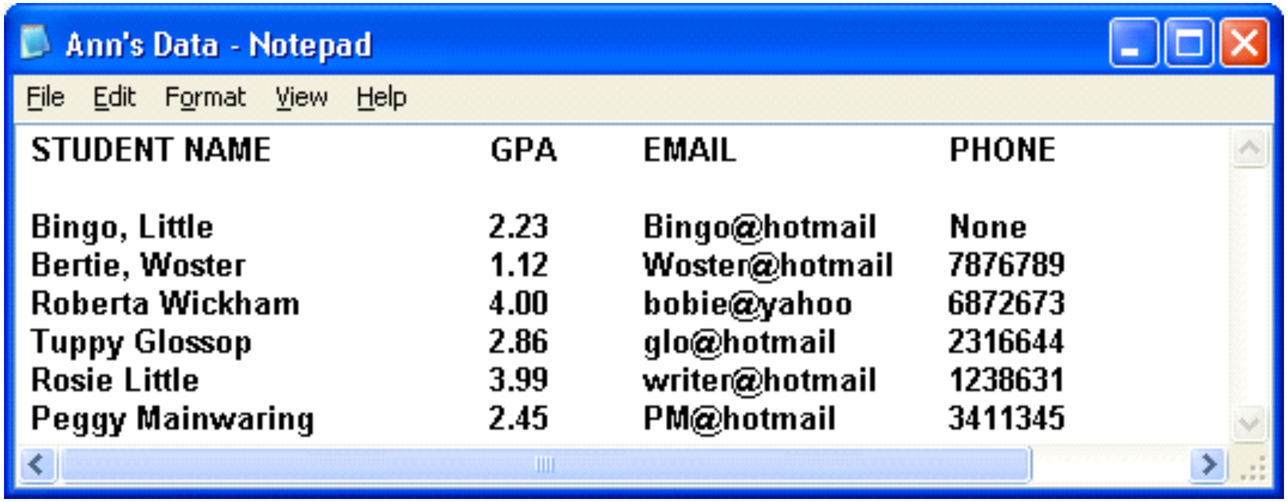
File Edit Format View Help

STUDENT NAME	GPA	EMAIL	PHONE
Bingo, Little	2.23	Bingo@hotmail	None
Bertie, Woster	1.12	Woster@hotmail	7876789
Roberta Wickham	4.00	bobie@yahoo	6872673
Tuppy Glossop	2.86	glo@hotmail	2316644
Rosie Little	3.99	writer@hotmail	1238631
Peggy Mainwaring	2.45	PM@hotmail	3411345

Concurrency

• Suppose I leave my text file on UNIX account, and I log in and begin to modify it at the same time my TA is modifying it!

• A DBMS will automatically make sure that this kind of thing cannot happen.

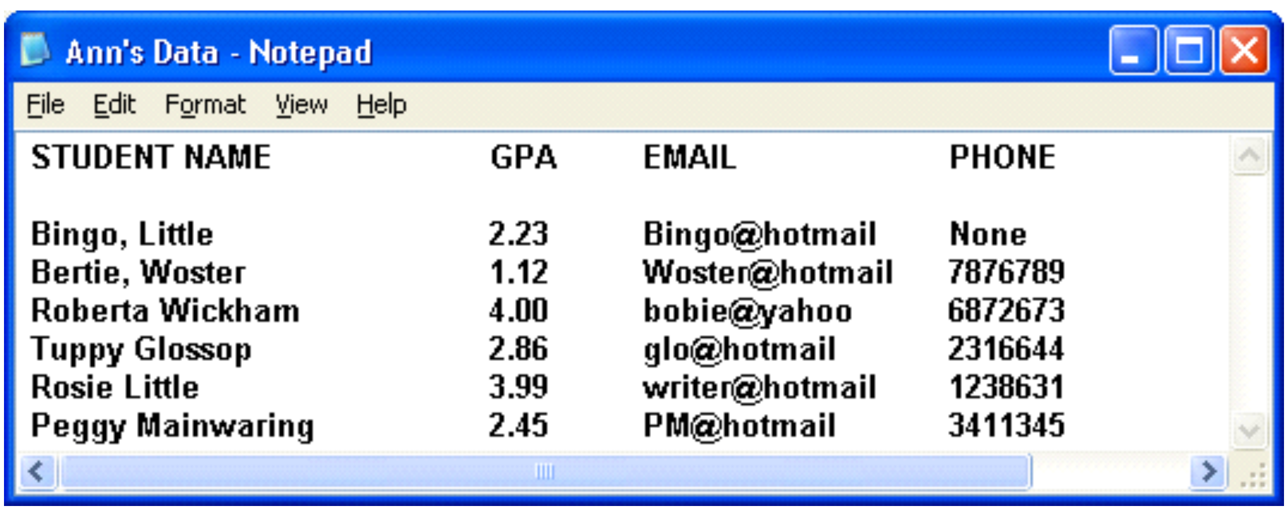


STUDENT NAME	GPA	EMAIL	PHONE
Bingo, Little	2.23	Bingo@hotmail	None
Bertie, Woster	1.12	Woster@hotmail	7876789
Roberta Wickham	4.00	bobie@yahoo	6872673
Tuppy Glossop	2.86	glo@hotmail	2316644
Rosie Little	3.99	writer@hotmail	1238631
Peggy Mainwaring	2.45	PM@hotmail	3411345

Security

• Suppose I leave my text file on UNIX account, and a student hacks in and changes their grades…

• A DBMS will allow multiple levels of security.

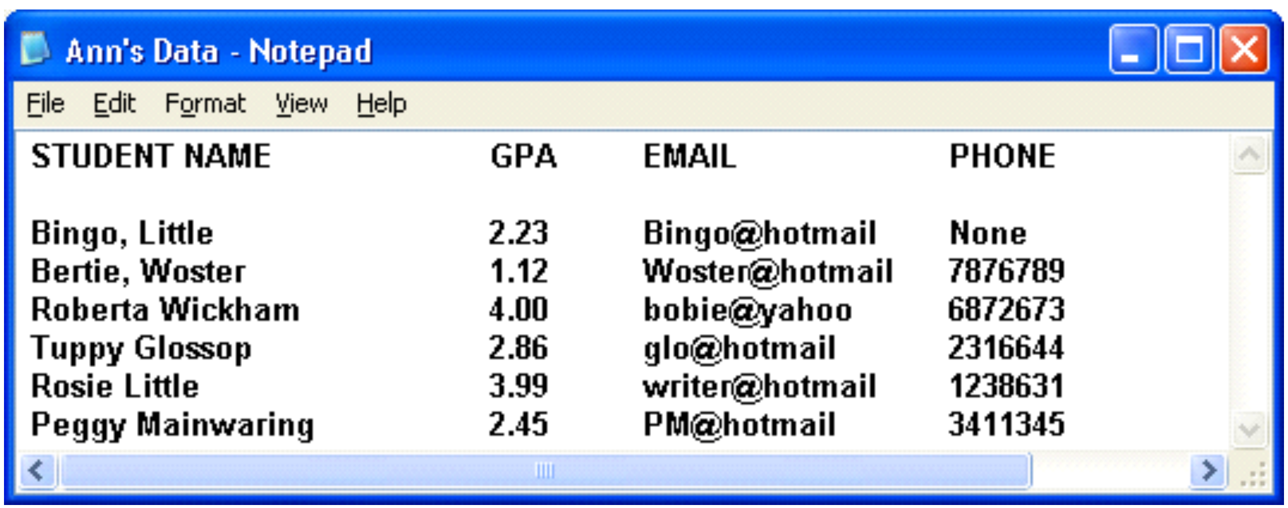


STUDENT NAME	GPA	EMAIL	PHONE
Bingo, Little	2.23	Bingo@hotmail	None
Bertie, Woster	1.12	Woster@hotmail	7876789
Roberta Wickham	4.00	bobie@yahoo	6872673
Tuppy Glossop	2.86	glo@hotmail	2316644
Rosie Little	3.99	writer@hotmail	1238631
Peggy Mainwaring	2.45	PM@hotmail	3411345

Crash Recovery

- Suppose I am editing my text file and the system crashes!

- A DBMS is able to guarantee 100% recovery from system crashes (to a consistent state).



STUDENT NAME	GPA	EMAIL	PHONE
Bingo, Little	2.23	Bingo@hotmail	None
Bertie, Woster	1.12	Woster@hotmail	7876789
Roberta Wickham	4.00	bobie@yahoo	6872673
Tuppy Glossop	2.86	glo@hotmail	2316644
Rosie Little	3.99	writer@hotmail	1238631
Peggy Mainwaring	2.45	PM@hotmail	3411345

Summary: Plain Text vs. Database

- When design a reliable database:
  - Verify Valid Input
  - Improve Scalability (Reading/Writing)
  - Support Comprehensive Query / Formal Expression
  - Prevent Data Loss
  - Prevent Unauthorized Access
  - Prevent Conflict Action
  - …

Don’t need to consider about these issues!

Not so good example: C++ vs. Python

Data Independence

- Applications are insulated from how data is structured and stored.
- *Logical data independence*: Protection from changes in *logical* structure of data.
- *Physical data independence*: Protection from changes in *physical* structure of data.

☛ *One of the most important benefits of using a DBMS!*

Purposes of DBMS

- Provide support for “easy-to-use” data
  - Data model (data)
  - Transaction model (operation)
- Provide efficient storage and access of the data in terms of the data model and transactional model.

To sum up: Why Use a DBMS?

Easier and More Efficient

- Data independence and efficient access.
- Query expressiveness
- Reduced application development time.
- Data integrity and security.
- Concurrent access, recovery from crashes.

- Any reasons to NOT use a DBMS?

Database Users

- End users (or DB application users)
- DB application programmers (more precisely, they are *DBMS* users)
  - E.g. webmasters
- *Database administrator (DBA)*
  - Designs logical /physical schemas
  - Handles security and authorization
  - Data availability, crash recovery
  - Database tuning as needs evolve

Must understand how a DBMS works!

Data Models

- Data model: A class of mathematical structures, with description and operations
- Necessary to be *general* and *intuitive.*
- Conceptual data model: Just structural description
  - Identifies the highest-level relationships between different entities
  - Features include the main entities and the relationships among them

Something we need to consider

- **High Availability**: must be operational while enterprise is functioning
- **High Reliability**: correctly tracks state, does not lose data, controlled concurrency
- **High Throughput:** many users => many operations/sec
- **Low Response Time**: users are waiting

Database System Requirement

- Long Lifetime: complex systems are not easily replaced
  - Must be designed so they can be easily extended as the needs of the enterprise change
- Security: sensitive information must be carefully protected since system accessible to many users

Database System

- Applications interact with a database by generating:

- **Queries**: that access different parts of data and formulate the result of a request.
  - report grade of all students in CSCI4333 of Spring 2023

- **Transactions**: that may read some data and "update" certain values or generate new data and store that in the database

Transaction

- A transaction is an application program with special properties to guarantee it maintains database correctness

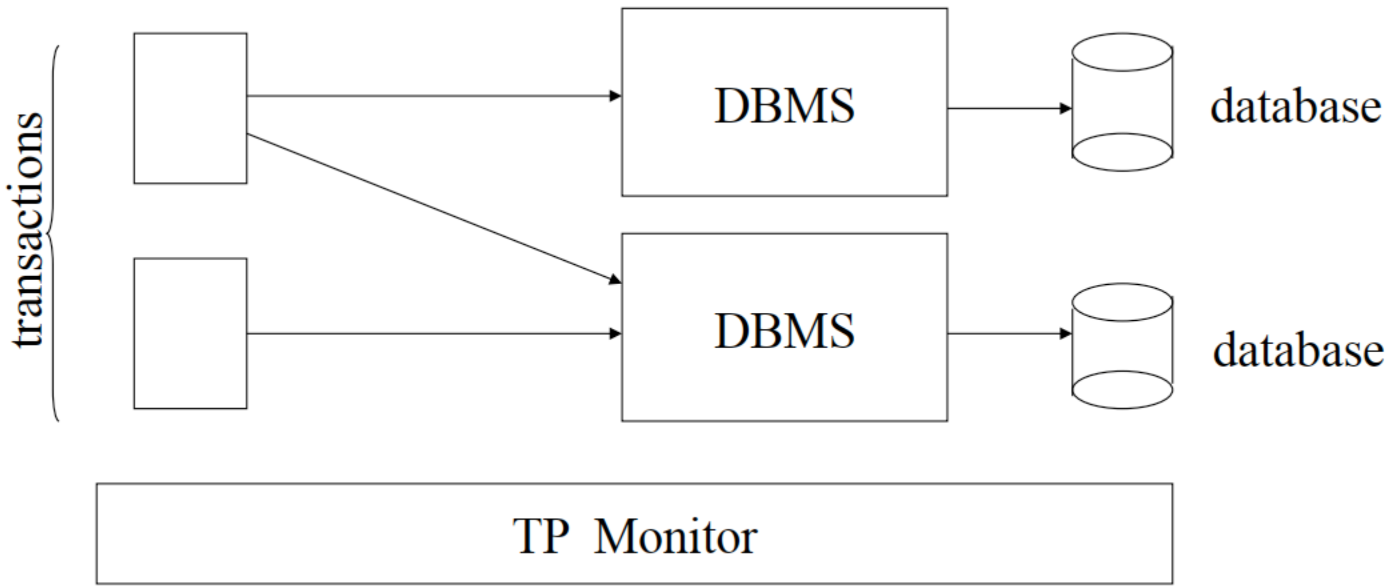
Transaction

- When an event in the real world changes the state of the enterprise, a transaction is executed to cause the corresponding change in the database state
  - With an on-line database, the event causes the transaction to be executed in real time

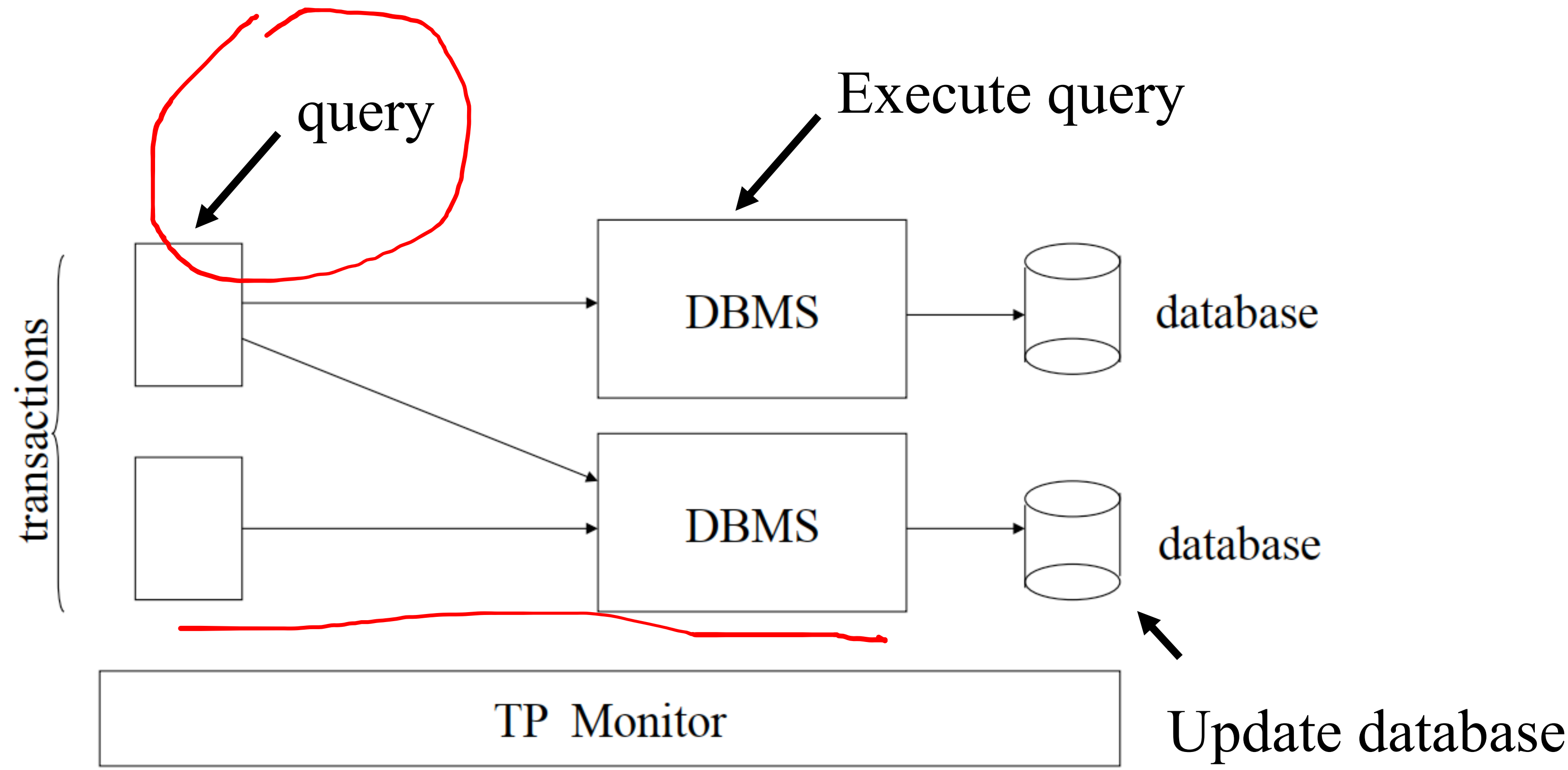
Transaction Processing System

- Transaction execution is controlled by a TP monitor
  - Creates the abstraction of a transaction, analogous to the way an operating system creates the abstraction of a process
  - TP monitor and DBMS together guarantee the special properties of transactions
- A Transaction Processing System consists of TP monitor, databases, and transactions

Transaction Processing System



Transaction Processing System



OLTP vs. OLAP

- **On-line Transaction Processing** (OLTP)
  - Day-to-day handling of transactions that result from enterprise operation
  - Maintains correspondence between database state and enterprise state
- **On-line Analytic Processing** (OLAP)
  - Analysis of information in a database for the purpose of making management decisions

OLAP

- Analyzes historical data (terabytes) using complex queries
- Due to volume of data and complexity of queries, OLAP often uses a data warehouse
- **Data Warehouse -** (offline) repository of historical data generated from OLTP or other sources
- **Data Mining** - use of warehouse data to *discover* relationships that might influence enterprise strategy

Example: Supermarket

- OLTP
  - Event is 3 cans of soup and 1 box of crackers bought; update database to reflect that event
- OLAP
  - Last winter in all stores in northeast, how many customers bought soup and crackers together?
- Data Mining
  - Are there any interesting combinations of foods that customers frequently bought together?

Overview of Database Design

- *Conceptual design*
  - Use *ER Model*: E- *Entities* and R-*Relationships*
  - Decide the *entities* and *relationships* in the enterprise.
  - Decide what information about these entities and relationships should we store in the database.
  - Decide the *integrity constraints* or *business rules*.
- *Implementation (logical design)*
  - Map an ER model into a relational schema.

Example: University Database

Students(*sid*, *name*, *login*, *age*)

Faculty(*fid*, *fname*, *sal*)

Courses(*cid*, *cname*, *credits*, semester)

Enrolled(*sid*, *cid*)

Teaches(*fid*, *cid*)

Grades(sid, *cid*, gpa)

Example: University Database

Students(*sid*:string, *name*:string, *login*:string, *age*:integer)

Faculty(*fid*:string, *fname*:string, *sal*:real)

Courses(*cid*:string, *cname*:string, *credits*:integer, semester:string)

Enrolled(*sid*:string, *cid*:string)

Teaches(*fid*:string, *cid*:string)

Grades(sid:string, *cid*:string, gpa:real)

Example: University Database

Students(*sid*:string, *name*:string, *login*:string, *age*:integer)

Faculty(*fid*:string, *fname*:string, *sal*:real)

Courses(*cid*:string, *cname*:string, *credits*:integer, semester:string)

Enrolled(*sid*:string, *cid*:string)

Teaches(*fid*:string, *cid*:string)

Grades(sid:string, *cid*:string, gpa:real)

Entities:

Students

Faculty

Courses

…

Example: University Database

Students(*sid*:string, *name*:string, *login*:string, *age*:integer)

Faculty(*fid*:string, *fname*:string, *sal*:real)

Courses(*cid*:string, *cname*:string, *credits*:integer, semester:string)

Enrolled(*sid*:string, *cid*:string)

Teaches(*fid*:string, *cid*:string)

Grades(sid:string, *cid*:string, gpa:real)

Relation:

Students take **Courses (Enrolled)**

Faculty teach **Courses (Teaches)**

…

Example: University Database

Students(*sid*:string, *name*:string, *login*:string, *age*:integer)

Faculty(*fid*:string, *fname*:string, *sal*:real)

Courses(*cid*:string, *cname*:string, *credits*:integer, semester:string)

Enrolled(*sid*:string, *cid*:string)

Teaches(*fid*:string, *cid*:string)

Grades(sid:string, cid:string, gpa:real)

Constraint:

We only have three semester {Fall, Summer, Spring}

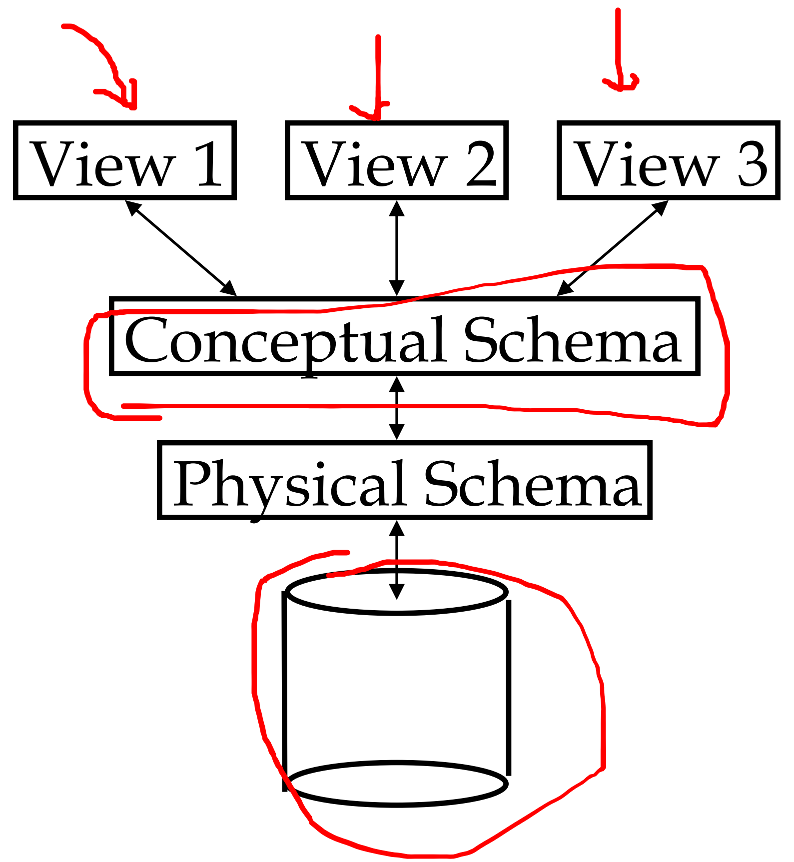
A faculty teach 1 to 3 courses

A student can take no more than 6 courses

…

Levels of Abstraction

- Many *views*, single *conceptual (logical) schema* and *physical schema*.
  - Views describe how users see the data.
  - Conceptual schema defines logical structure
  - Physical schema describes the files and indexes used.



Back to Example: University Database

Students(*sid*:string, *name*:string, *login*:string, *age*:integer)

Faculty(*fid*:string, *fname*:string, *sal*:real)

Courses(*cid*:string, *cname*:string, *credits*:integer, semester:string)

Enrolled(*sid*:string, *cid*:string)

Teaches(*fid*:string, *cid*:string)

Grades(sid:string, cid:string, gpa:real)

View:

CourseInfo (*cid*:string, *cname*:string, *credits*:integer, semester:string, *fname*:string)

Transcript (*sid*:string, *name*:string, *cname*:string, *credits*:integer, gpa:real)

Summary

- DBMS used to maintain, query large datasets.
- Benefits include recovery from system crashes, concurrent access, quick application development, data integrity and security.
- Levels of abstraction give data independence.
- We will learn how to
  - *Design* and set up a database
    - Design (ER and Relational Models), and refine (Relational Normalization Theory)
  - *Query* the database
    - Relational Algebra and SQL
  - *Implement* database applications