Introduction to Databases
95.305*

Louis D. Nel
ldnel@scs.carleton.ca

Scenarios

- A company wishes to store and access information about its employees, departments and projects.
- The data will be accessed by different departments for various reasons

Scenarios

- Students (and Instructors) would like to know how assignment marking is progressing.
- Students should be able to determine the progress on their assignments
- Instructors should be able to generate reports about all of the assignment grades in the class.

File Systems as Database

- Consider using the fact that we can write programs which open files, write to files, and close files
- We could simply write programs to store data and retrieve it from the appropriate files when needed

...File Systems as Database

- Disk files accomplish one very important aspect of database systems:
- Data lives a long time -longer than the life of the program accessing it

...File Systems as Database

- Goal: creating a new database of student marks
- soln: create a new file, decide on the format of data, write programs which access data based on known format
- Problems:

  -data format is implicit and hard coded
  -data is probably duplicated -exists in
  files as well, or duplicated in single file
  -no explicit notion of data schema

...File Systems as Database

- Goal: query the database to print names of students with A+
- soln: write a procedure to search file for students with mark=A+
- Problems:

  -query is hard-coded, requires program
  -new queries require re-programming
  -no theory for query optimization

...File Systems as Database

- Goal: Data should be free of inconsistencies (You cannot get a B and A- in the same course)
- soln: make sure procedures verify data is valid
- Problems:

  -integrity constraints cannot be found or changed without re-programming
  -integrity constraints cannot be changed

...File Systems as Database

- Goal: Data should be secure from unauthorized users
- soln: use file access permissions
- Problems:
  - -file access permission provided by host operating system will be inadequate
  - -host OS may not provide any file access restrictions

...File Systems as Database

- Goal: Data should be usable by several user concurrently (Two agents book a seat on a plane simultaneously)
- soln: Custom locking code
- Problems:
  - -Must write explicit locking code
  - -Host OS file system will not provide basic concurrency control

...File Systems as Database

- Goal: Data invariants must be maintained (During an account transfer the power fails after the withdrawal but before the deposit)

- soln: no easy solution
- Problems:
  - -Must write explicit code to ensure the a program failure will allow an Undo on data modified so far

Database Solution

- Create an application to which we input:
  - -what our data will look like
  - -what constraints apply to the data
  - -allows us to pose arbitrary queries
  - -allows us to specify access controls
  - -monitors modification of data
  - -provides transaction control

- The database is generic -works with anyone’s data or queries -at run time

Database management System

DBMS allows users to:

- create new databases, specifying their schema (using a DDL)
- Query and modify data (using DML)
- Keep large amounts of data secure from accidents or unauthorized users
- Allow many users access to data at once without corruption

Problems to Avoid

- Difficult to access data
- Data Redundancy and Inconsistency
- Structure and Data Dependence
- Data Isolation
- Concurrent Access Anomalies
- Security Problems
- Integrity Problems

Difficult to access Data

- Does not anticipate new types of requests or reports
- How can you get the database to handle query its never done before without requiring re-programming?
- Database should anticipate new queries and provide a way of navigating data
- Provide a specific Query Language

Data Redundancy and Inconsistency

- e.g. Customer name and address appears in several files: CUSTOMER_FILE, ACCOUNTS_FILE
- Wasteful
- Allows for possible inconsistencies (changed in one file but not the other
  e.g. for change of address -this really happens)

Structure and Data Dependence

- In file processing systems the programs are written to expect a certain file structure and data type
- In a database we want the file formats (schemas) to be changeable.
- Programs will first read the schema then use this schema to access the data files

Data Isolation

- Lack of architecture for data
- In file processing systems data can be scattered in various files or different formats
- Programmer has to hunt for both information and formats

Concurrent Access Anomalies

- Databases need to be multi-user, multi-access
- What happens when two people try to change the same data
- e.g.

  withdrawals from a bank account
  reserving a seat on an airplane

Security Problems

- Nobody wants to see all of the data
- Employees need not have access to all personnel records
- Need access policies that can be enforced and managed

Integrity Problems

- Data has meaning and the values stored should make sense
- Type constraints:
  -account balance cannot be “Friday”
  -customer name cannot be “Lou3is Nel”
   (the 3 is silent)
- Policy Constraints
  -account balance cannot be < $100.00

Architecture of Databases

- Ultimately using a database will involve (at some stage) coming up with a data architecture
- The users model their enterprise data
- Done using data abstractions and modeling languages
- Data models describe the database Schema

Schemas and Instances

- Schema

  -description of the data
  -contains no data values
  -also called “Meta-data”
  -created with the database DDL
  -created when database is designed
  -should not change frequently

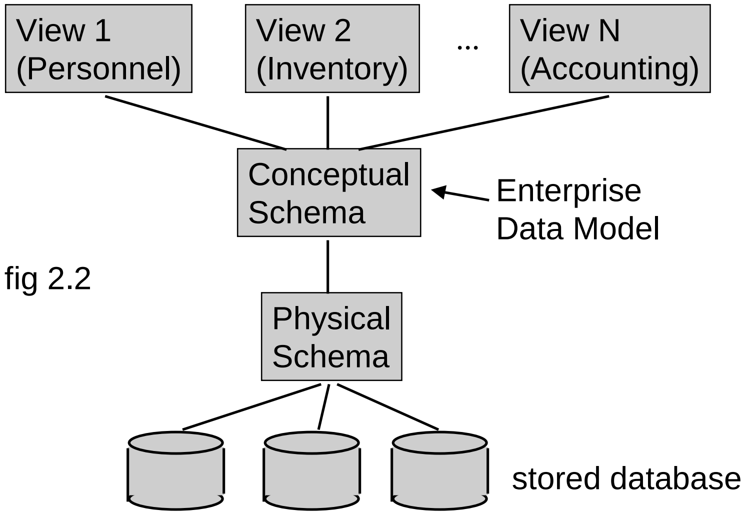
STUDENT		
Name	StudentNumber	Major

- Instances

  -actual data
  -changes frequently
  -also called database “state”
  -manipulated with the DML

Lou	1223	Comp Sci
Mary	5432	Math
Roger	3478	Comp Sci
Sue	1221	Physics

Three Schema Architecture



see fig 2.2

Data Independence

- **Logical Data Independence**
  being able to change the conceptual schema without affecting external views or application programs

- **Physical Data Independence**
  being able to change the physical data schema without affecting the conceptual schema

Classification of DBMS

Data Model
relational
network
hierarchical
object-oriented

Users
single-user
multi-user

Purpose
general purpose
special purpose
-telephone directory
-airline reservation
(online transaction processing)

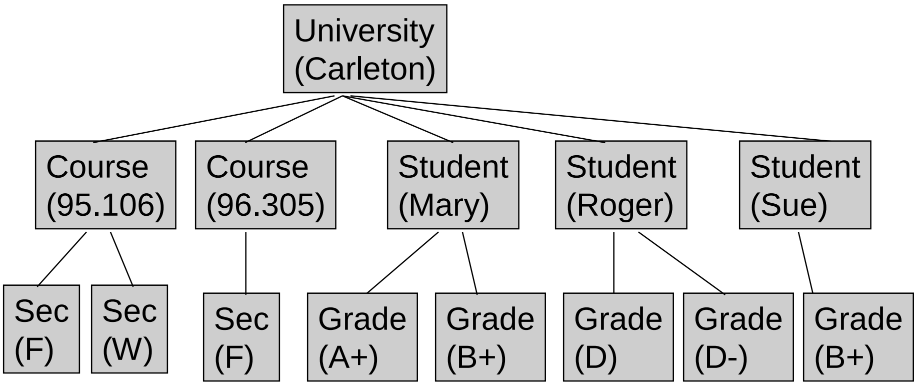
Cost
$10,000
$100,000
$100
$1,000

Sites
centralized
distributed
-homogeneous
-heterogeneous

Classification of DBMS:

Hierarchical Model

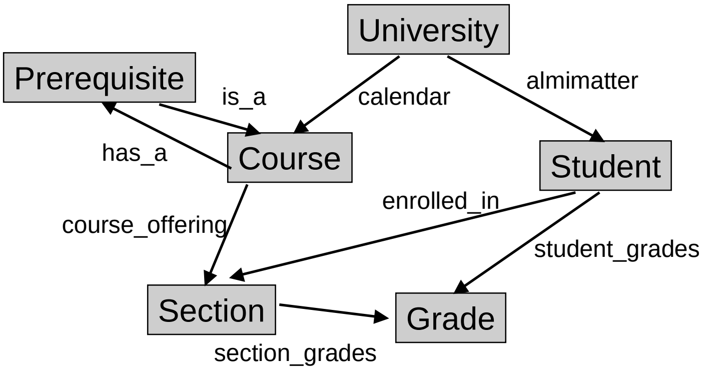
- Represents database as a hierarchical tree


University (Carleton)
Course (95.106)
Course (96.305)
Student (Mary)
Student (Roger)
Student (Sue)
Sec (F)
Sec (W)
Sec (F)
Grade (A+)
Grade (B+)
Grade (D)
Grade (D-)
Grade (B+)

Classification of DBMS:

Network Model

- Represents data as records with relationships captured as record sets


University
Prerequisite
is_a
calendar
almimatter
has_a
Course
Student
enrolled_in
course_offering
student_grades
Section
Grade
section_grades

Hierarchical and Network Models

- Used in 60’s early 70’s
- Mostly of historical interest
- Did not support high level queries
- Users wrote programs which navigated the data’s structure

Classification of DBMS: Relational model

- Represents database as collection of tables each stored in separate file, linked through keys

STUDENT

Name	StudentNumber	Major
Lou	1223	Comp Sci
Mary	5432	Math
Roger	3478	Comp Sci
Sue	1221	Physics

SECTION

SecNo	Course	Term
121	CS95.305	F
113	CS95.305	W
100	MA69.107	F
121	CS95.106	F

GRADES

StdNo	Section	Grade
1223	113	A
1223	100	B-
5432	121	A
3478	121	C

Relational Model

- Proposed by Codd, 1970
- User would not be concerned with storage structure
- Queries expressing in high level language (SQL -most important)
- Users avoid navigating through the database -like people now search the web
- Based on very simple models of relations

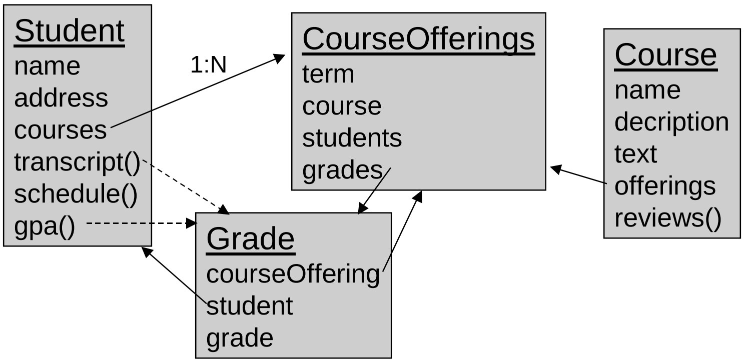
Relational Model -sample query

```
SELECT grade
FROM GRADES
WHERE StdNo = 1223 AND Section = 100
```

- We are asking the database to:
  -examine all tuples in GRADES
  -pick those satisfying some criteria
  -produce a report on the grades field of
   the tuples which satisfy the criteria
- Notice we are not navigating the structure of the data -left up to system to do it efficiently

Classification of DBMS:
Object-Oriented Model

- Data is stored as objects, relationships are stored as object references to other objects
- Objects have: attributes <u>and operations</u>



<u>Student</u>
name
address
courses
transcript()
schedule()
gpa()

1:N

<u>CourseOfferings</u>
term
course
students
grades

<u>Course</u>
name
decription
text
offerings
reviews()

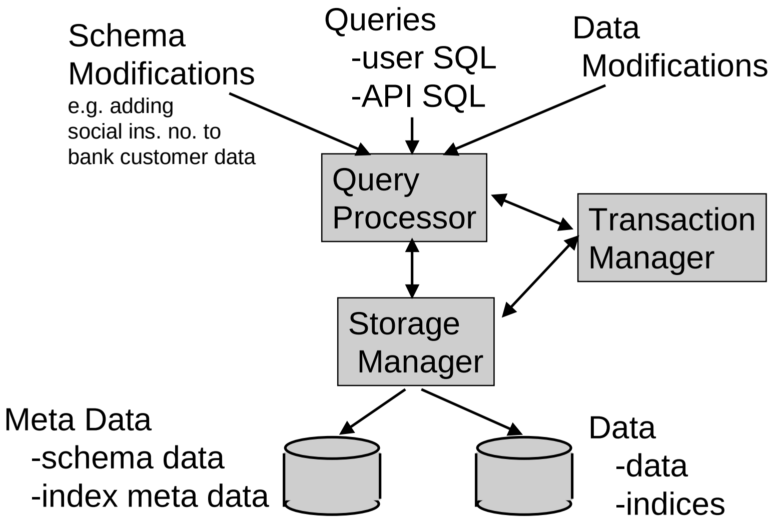
<u>Grade</u>
courseOffering
student
grade

Data and Memory

- A key feature that distinguishes DBMS’s from other computer applications is:

- A DBMS implicitly assumes data cannot fit in main memory -it must be out there in secondary store (Hard Disk) or tertiary store (CD)
- Remember in algorithms class they said accessing an array element was O(1),
  -they lied, it’s O(log n)
  (now try writing your O(nlogn) sort)

DBMS Components



Schema Modifications
e.g. adding social ins. no. to bank customer data

Queries
-user SQL
-API SQL

Data Modifications

Query Processor

Transaction Manager

Storage Manager

Meta Data
-schema data
-index meta data

Data
-data
-indices

Storage Manager

- Worries about how data is stored on disk (block size, cache size, ...)
- Shields other components from knowledge of how data is stored (allow transparent upgrades)
- Usually by-passes host operating system for accessing disks -for efficiency
- Small DBMS may just make use of host OS - like on your PC.

Query Manager

- Implements a query posed in high level (turns it into an data navigation strategy)
- Must optimize the query, come up with a query plan (use index if it exists, perhaps create one)
- Tricky subject, especially for federated databases

Transaction Manager

- Groups actions an ensures their completion as a unit
- Enforces integrity constraints on database
- Arbitrates over concurrent transactions
- Ensures ACID properties
  - -Atomicity of transactions
  - -Consistency of data (invariants)
  - -Isolation of transactions from each other
  - -Durability of transaction effects -they're not lost

We will Study

- Database modeling and design
  - -how to model enterprise data for a database
- Database use and programming
  - -how to access and manipulate data
- Database implementation
  - -how databases and DBMS’s can be implemented