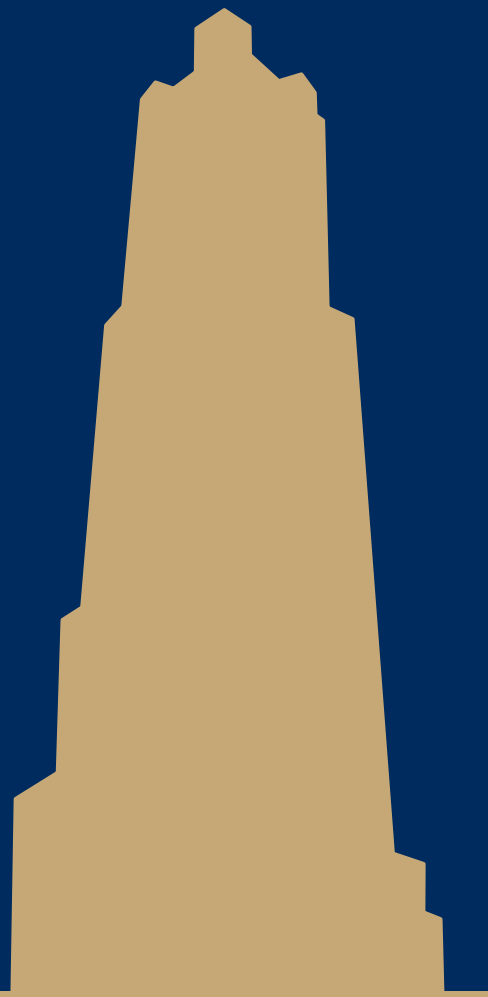
CS 1555

www.cs.pitt.edu/~nlf4/cs1555/

Introduction



Meta-notes

- These notes are intended for use by students in CS1555 at the University of Pittsburgh. They are provided free of charge and may not be sold in any shape or form.
- These notes are incomplete and NOT a substitute for material covered during course lectures. If you miss a lecture, you should definitely obtain both these notes and notes written by a student who attended the lecture.
- Material from these notes is obtained from various sources, including, but not limited to, the following:
  - Fundamentals of Database Systems by Elmasri and Navathe
  - Various online resources (see notes for specifics)

Instructor Info

- Nicholas Farnan (nlf4@pitt.edu)
  Office: 6313 Sennott Square

- All other info appears on the class website
  - E.g., office hours, TA contact info/office hours

A note about email

- Prefix all email subjects with [CS1555]
- Address all emails to both the instructor and the TA

Course Info

- Website:
  - www.cs.pitt.edu/~nlf4/cs1555/

- **Review the Course Information and Policies**

- **Assignments will not be accepted after the deadline**
  - No late assignment submissions
  - If you do not submit an assignment by the deadline, you will receive a **0** for that assignment

Term project

- Small group based
- Will require the use of git for development/submission
  - You will NEED a private repository on GitHub to host your project
    - Provided for free to students, see
      - https://education.github.com/
    - Apply for a private repository NOW
  - Git documentation:
    - https://git-scm.com/documentation

Data management

- How long does it take Amazon to give your order history?
- How long does Facebook take look up a friend’s friends?
- How long does Google spend accumulating all of your Google+ posts?

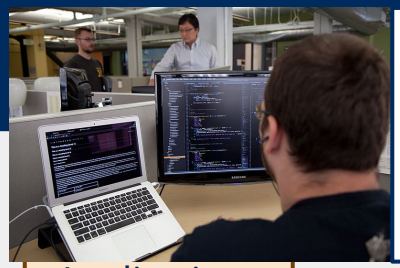
Simple approaches to data management

- Write everything to text files!
  - Have your applications directly read/write to these files
  - Problems?
    - Slow!
    - Have to constantly enforce layout of data
    - What if multiple instances of your application need to read/write to the same file at the same time?
    - What if one of them crashes?

DBMSs can address all of these problems

- DataBase Management System
  - A *general-purpose software system* that facilitates the process of *defining, constructing, manipulating,* and *sharing* databases among various users and applications

Database systems



Application Developer

Applications

Web forms

Applications

Shell

DBMS

Database Developer

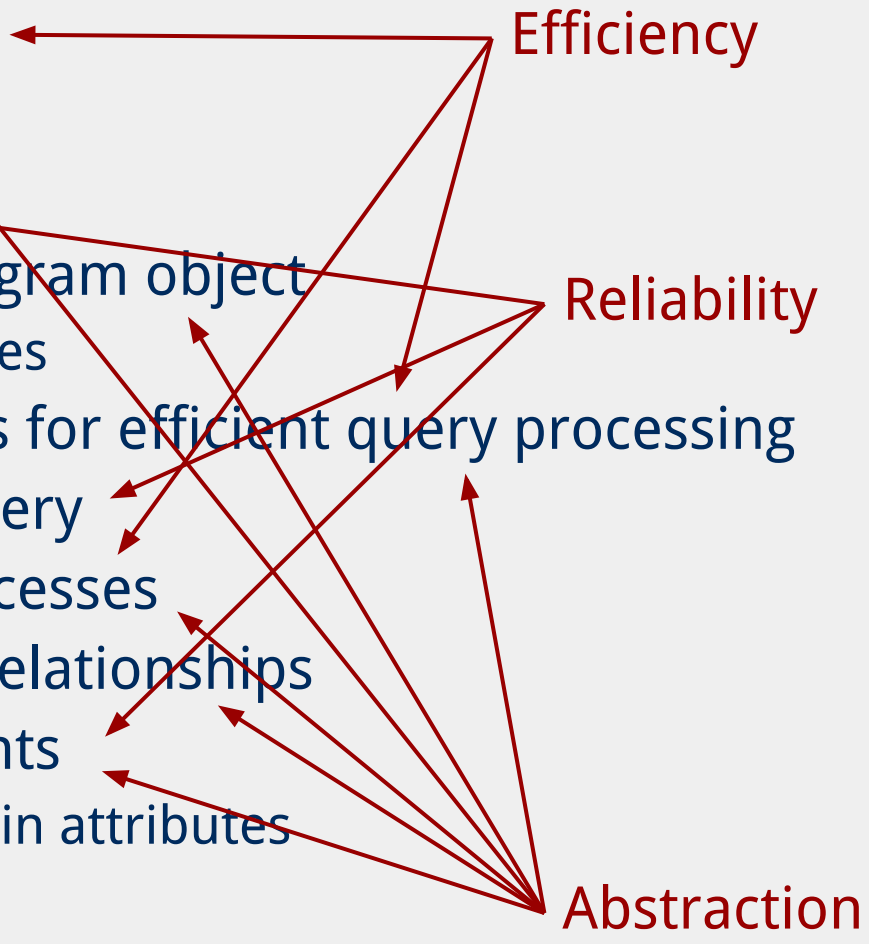
Database

Data

Metadata

DBMSs provide:

- Control over redundancy
  - Via *data normalization*
- Access controls
- Persistent storage for program object
  - Object-oriented databases
- Structures and techniques for efficient query processing
- Provide backup and recovery
- Provide for concurrent accesses
- Represent complex data relationships
- Enforce integrity constraints
  - e.g., uniqueness of certain attributes



Efficiency

- Controlling redundancy
  - Saves space on physical storage
- Queries are *optimized*
  - Returns a response to the user in a short or bounded time
  - DBMS works to find the best way to answer user questions

Reliability

- Access control
  - Subjects - who? (e.g., users)
    - Requires authentication
  - Objects - what? (e.g., stored data)
  - Authorizations - how? (e.g., read, write)
- Integrity constraints
  - Data type
  - Data relationships
- Ensure integrity despite failure
  - Data cannot be lost if the system or program fails for any reason

Execution Abstraction

- *Transactions*
  - A transaction is a logical unit of work in DBMSs
  - A transaction is an executing program or process that includes one or more database accesses such as reading or updating of database *records*
  - Examples:
    - Transferring money between bank accounts
    - Inventory updates

ACID

- **A**tomicity
  - Either all the operations associated with a transaction happen or none of them happens
- **C**onsistency Preservation
  - A transaction is a correct program segment. It satisfies the database's integrity constraints at its boundaries
- **I**solation
  - Transactions are independent, the result of the execution of concurrent transactions is the same as if transactions were executed serially, one after the other
- **D**urability (a.k.a. Permanency)
  - The effects of completed transactions become permanent surviving any subsequent failure(s)

Data Abstraction

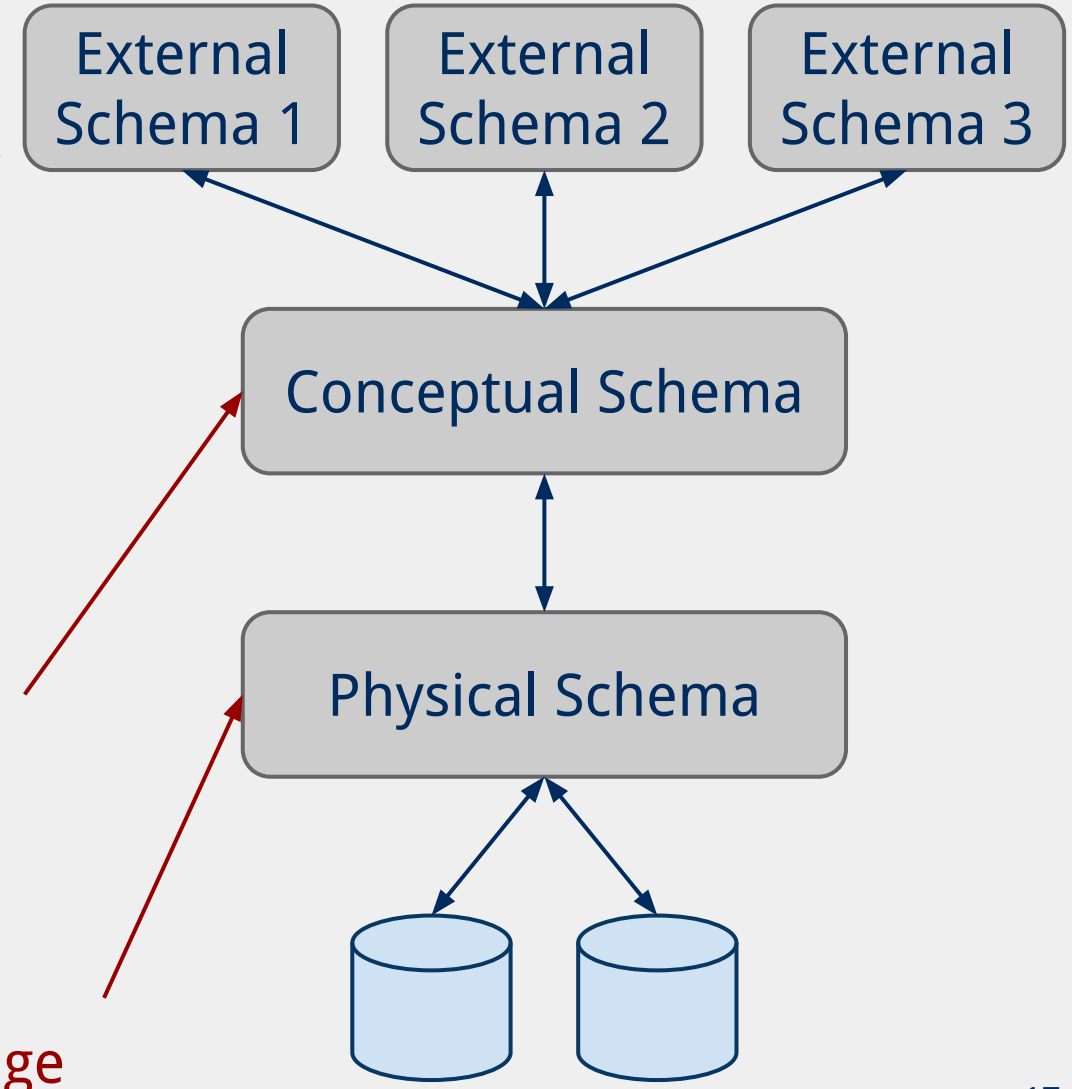
- *Data Models*
  - A collection of high-level data description constructs that hide low-level storage details
- In this class, we will be primarily concerned with the *object-relational* data model
  - Relational data model with concepts inherited from object-oriented databases
  - Primary construct is the *relation*
    - A.k.a. *table*
    - Made up of individual *records*
  - Every relation has a *schema* or description of the data it contains

DBMS 3-level architecture

Provides different *views* of the data depending on user requirements and permissions, computed as needed and specified using:
VDL: view definition language

Logical level, where base relations are specified using:
DDL: data definition language

Physical level, specifies how tables are stored using:
SDL: storage definition language



Those languages all specify schemas...

- What about actually adding data to the database and issuing queries over it?
  - Well you'd need a DML (data manipulation language) for that of course!
- Fortunately, for relational/object-relational database management systems, one language can play the role of DDL, VDL, and DML:
  - SQL (Structured Query Language)

Data management

- This class will focus on SQL and transactional object-relational database management systems
  - Is this a one-stop shop for all of your data management needs? No!
- These systems can be expensive
  - E.g., Oracle software/hardware (RIP Sun)
- Maybe they're overkill
  - Maybe you don't need concurrent, multi-user data access
  - Maybe your application is fault tolerant
  - Maybe your application/data needs are incredibly simple
- Maybe they're insufficient
  - DBMS is a general-purpose solution, maybe your application would benefit from a purpose-built solution

Topics for the term

- The Relational Model
- DDL
- Normalization
- ER Modeling
- Relational Algebra
- DML
- VDL
- Access Control
- Transactions
- Integrity Contraints
- Database Programming
- File Storage and Indexing
- Query Processing
- Concurrency Control
- Recovery

Theory

SQL/Programming

Systems