CS 3200 Database Design Introduction

Kathleen Durant PhD

Lesson 1

Northeastern University

Lecture 1 Outline

- Course Logistics
- Goals of the course
- Outline of the Course
- The 5 W's
  - What, why, where, who, when of DBs
  - Brief History and fun facts of DBs
- The Relational Model
- Properties of a RDMS

Course Information

- Website:
  - http://www.ccs.neu.edu/home/kathleen/classes/cs3200/index.htm
- Professor:
  - k.durant@neu.edu
  - 460A WVH
- Teaching Assistant
  - Apurva Narasimhan
    - Narasimhan dot A at husky dot neu dot edu
  - Nisha Kanani
    - Kanani dot N at husky dot neu dot edu
  - Priyank Kumar
    - Kumar dot pr at husky dot neu dot edu
- Prerequisites
- Grading
  - Homework
    - Project
    - 5 assignments
  - In class Midterm
  - Final exam

Class Assessment

- Prerequisite
  - CS 2510 (CS U213)
- Point Distribution
  - Homework (25%)
    - 5 assignments
  - In class Midterm (20%)
  - Final exam (25%)
  - Project (25%)
  - Class participation (5%)

Average	Grade
< 60	F
< 63 and >= 60	D-
< 67 and >= 63	D
< 60 and >= 67	D+
< 70 and >= 67	C-
< 77 and >= 73	C
< 80 and >=77	C+
< 83 and >= 80	B-
< 87 and >= 83	B
<90 and >= 87	B+
< 94 and >= 90	A-
>= 94	A

Goals of the Course

- Learn the theory behind relational databases
  - Relational model, relational algebra, relational calculus
- Learn to design and represent a data schema
  - Entity relational model
- Given a data schema, create and manipulate a database using SQL
  - Translate an ERM schema into SQL
- Become familiar with the functionality provided by SQL
  - Strengths as well as its limitations
- Understand the internal workings as well as the functionality provided by a database management system
  - Concurrency-control, transactions, indexes
- Gain some industry knowledge and a historical perspective on databases
  - Codd, Stonebraker, Ellison, Date, Boyce, Wong, Chamberlin
  - Oracle, Ingres, IBM, Teradata, MySql, Postgres
  - Bachman vs. Codd debate

Outline of Course

	Lectures
Introduction, Entity Relationship Modeling	1,2, 3
Normal Forms	4 , 5
Relational Algebra & Calculus	6, 7
SQL Language, Triggers, Embedded SQL	8 - 12
Transactions, Concurrency	13 - 15
Recoverability	16-17
Midterm review & Midterm	18 – 19
Storage & Buffer Management	20 - 21
Indexing Methods	22 - 24
Query evaluation and optimization	25 – 29
No SQL Systems, Final Review	30 – 32
Project Presentations	33 – 36

5W's: **What** is a database?

"an organized set of data that is stored in a computer and can be looked at and used in various ways"

Oxford English Dictionary

"one or more large structured sets of persistent data, usually associated with software to update and query the data"

Free On-Line Dictionary of Computing

" a comprehensive collection of related data organized for convenient access, generally in a computer. "

Dictionary.com

Good start but not very precise !!

5 W's: **What** is a DB, DBMS?

Database = very large, integrated collection of data.

- Entities (e.g., students, courses)
- Relationships (e.g., Jill *is taking* CS 3200)

Database Management System (DBMS) = software package designed to store and manage databases

Definitions will be more precise as semester progresses!!

5W's: **Where** are databases?

- History
  - Database development analogous to the development of written language: humans needed a reliable means for transmitting information, maintaining financial accounts, keeping historical records (i.e. business transactions)
    - Where's the money?
  - Traditionally databases were found on mainframes at large entities such as corporations, hospitals and the government
    - Airline Reservation Systems – Data items are: single passenger reservations; Information about flights and airports; Information about ticket prices and tickets restrictions.
    - Banking Systems – Data items are accounts, customers, loans, mortgages, balances, etc.
    - Corporate Records – Data items are: sales, accounts, bill of materials records, employee and their dependents
  - Failures are not tolerable. Concurrent access must be provided
- Today
  - Databases are behind almost everything you do on the Web.
    - Google searches.
    - Queries at Amazon, eBay, etc.
  - Databases can exist on any computer and at no cost!
    - Personal computer: ACCESS, MySQL
    - Servers: Oracle, SQL Server
    - Data Warehouses: Teradata

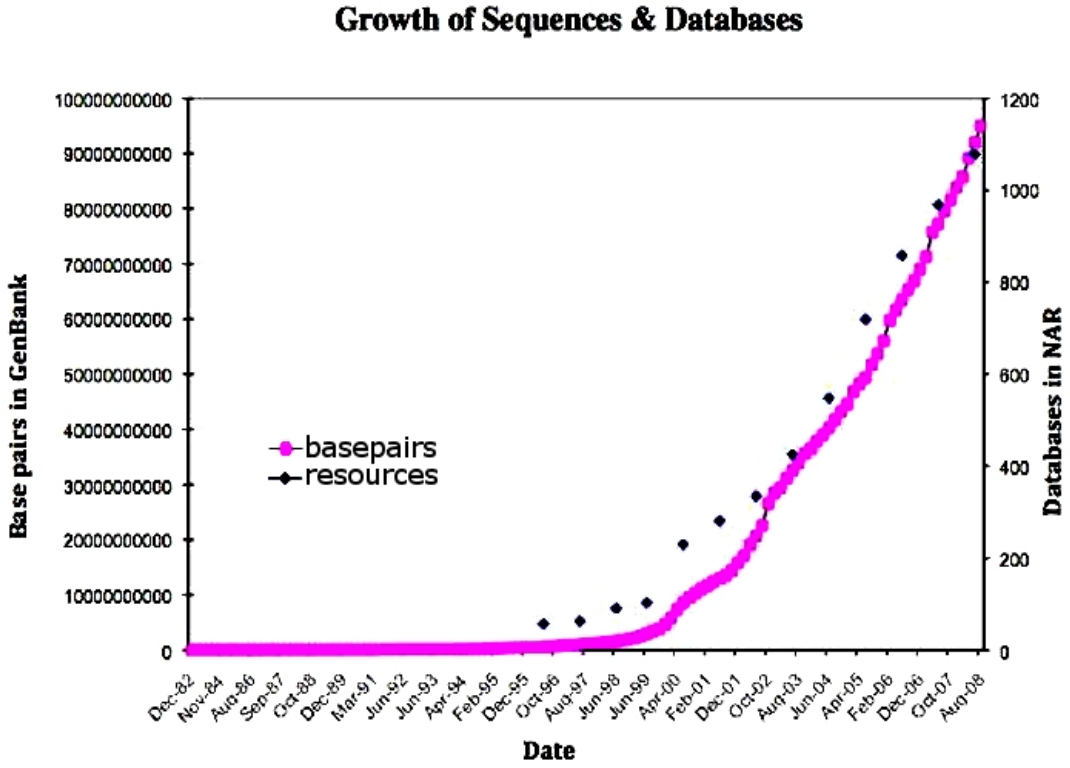
5W's: **Where** are the largest DB's

- World Data Centre for Climate – 6 Petabytes
- National Energy Research Scientific Computing Center (NERSC) – 2.8 petabytes
- AT&T – 312 Terabytes
- Google – 91 million searches per day
- Sprint - 70,000 call record insertions per second
- ChoicePoint – 250 Terabytes on Americans
- YouTube – 45 Terabtyes
- Amazon - 42 terabytes of data on 59 million
- CIA – all content digitized: growth rate of 100 articles per month
- Library of Congress – index for the library: growth rate of 10,000 items per day

http://www.comparebusinessproducts.com/fyi/10-largest-databases-in-the-world

5W's **Where**: Growth in Genetic data

- Explosion of available genetic data through advances in automatic genetic sequencing techniques
- Amount of genetic data doubling every 18 months

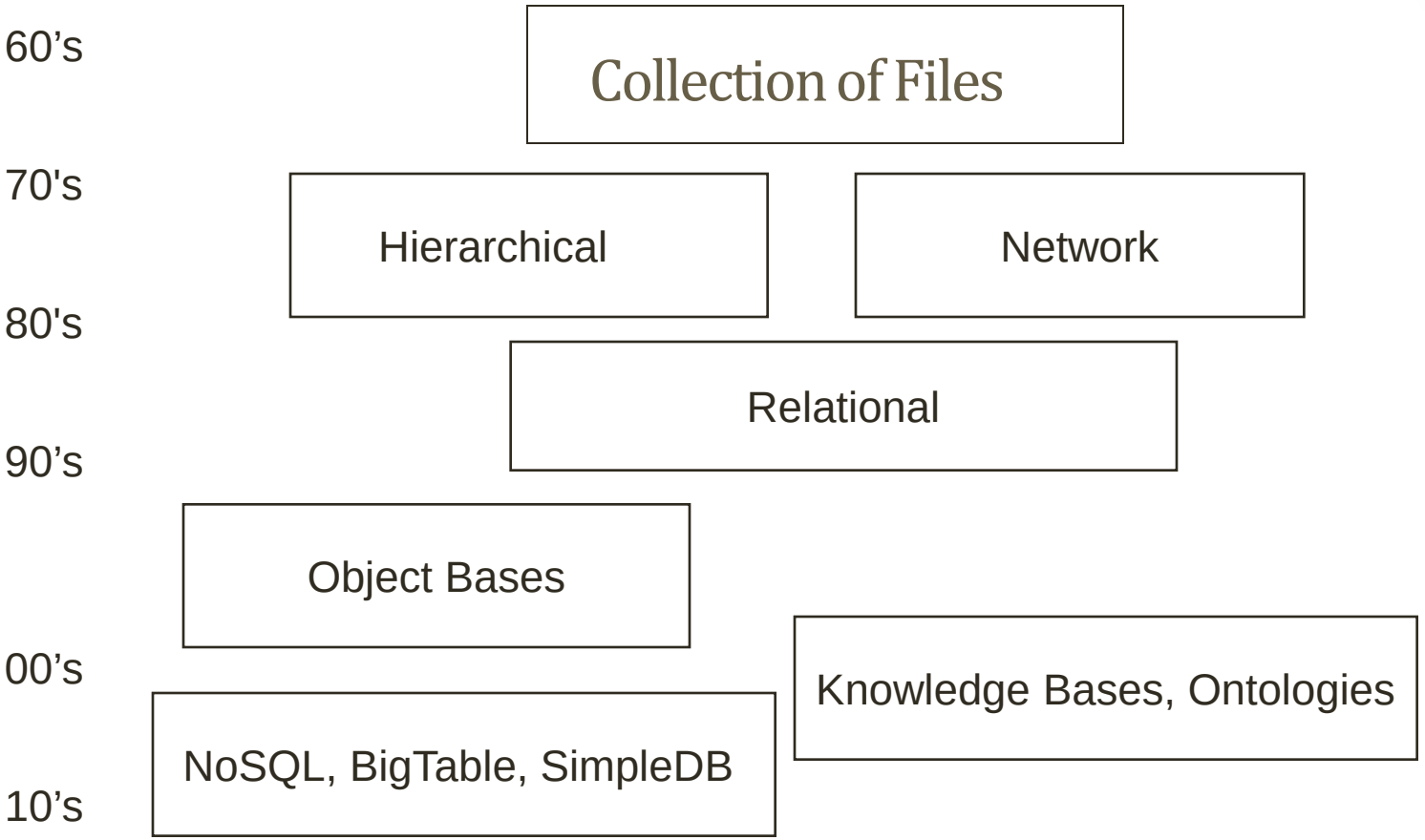


http://www.ncbi.nlm.nih.gov/genbank/
http://www.nature.com/scitable/topicpage/genomic-data-resources-challenges-and-promises-743721

5 W's: **When** did DBs get their start?

- Three Eras of Database Technology
  - Pre-relational
    - File systems
    - Hierarchical and network systems
  - The revolution: relational database technology
    - Codd's 1970 paper + 10 years
  - Post-relational era
    - New organizations of data, more complex data
    - Influence of object technology
    - More complex applications (e.g., distributed, web-based, and parallel)
    - No SQL DBs

5W's When: Storage of data



When:
Pre-RDB File-based data

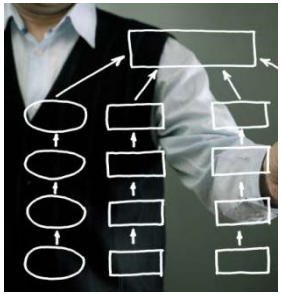
- File technology was the first attempt to automate manual filing systems
- Collection of applications that each define and manage their own files
- File: collection of records with structure linked to a specific application and a computer language or library that provides data access
- Data storage and retrieval must be coded explicitly in each application

When: Pre-RDB – Hierarchical and Network Systems

- Software systems without underlying organized discipline
  - **Lack of data independence**
  - File-based view of data structures as records (on disk storage) + links: mixture of physical and logical worlds
  - User view dependent on physical details
    - Not relevant for information needs or data concept
  - Modern view: links = semantic relationships + physical access paths
- Navigation through the data via programming
  - **No high-level query language** - all database accesses imply procedural programming
  - Programming = navigation governed by physical access paths
    - long and complex programs
  - Oriented towards one-record-at-a-time navigational programming
    - complex, necessarily-manual performance optimization
- Example: Integrated Data Services (IDS) developed at GE by Charlie Bachman in the 1960s

5W's: **When**: RDB revolution

- Codd's paper: 'A relational model of data for large shared data banks' (1970)
  - Linked data representation/operations to set theory
  - Just a theory no database implementation

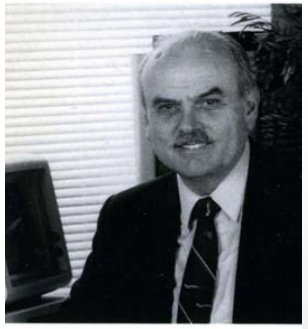


- Date @ IBM starts development of R
  - Boyce & Chamberlin develop SEQUEL based on relational algebra
- Stonebraker @ UC Berkeley starts development of Ingres
  - Stonebraker, Wong develops QUEL based on relational calculus

IDS Database Architect & Guru

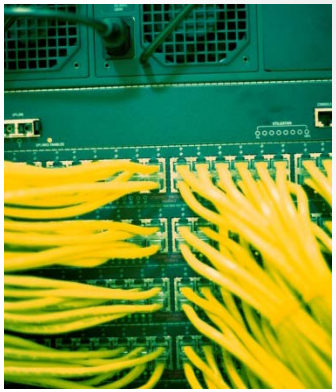
Bachman vs. Codd debate

1974 @ ACM SIGMOD



Mathematician

5W's: **When**: Post-revolution



- New organizations of data
  - Non-normalized relations
  - Object technology and object-relational systems
    - Hierarchical data encapsulation for an object (multiple tables for one object)
  - Semi-structured and unstructured data (XML)
  - Vertical databases
- New functionality
  - Distribution
  - Heterogeneity (multi-databases, interoperability)
  - Active databases (triggers) and deduction
  - ERP packages (application-oriented tasks common to many organizations)
  - Data analysis (data warehouses and data mining)
- More complex data domains (e.g., design, geography, molecular biology, electronic commerce)
- Relaxation of the ACID test for DBMS

http://slashdot.org/topic/bi/sql-vs-nosql-which-is-better/

5 W's: **Who** are the corporate players?

Open Source

RDB Companies

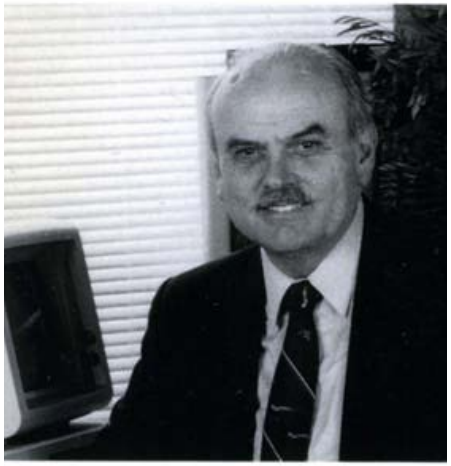
Computer Companies

5W's: **Who** are the RDB pioneers?

E. Codd

IBM

relational model



M. Stonebraker

UC Berkleley Ingres DB

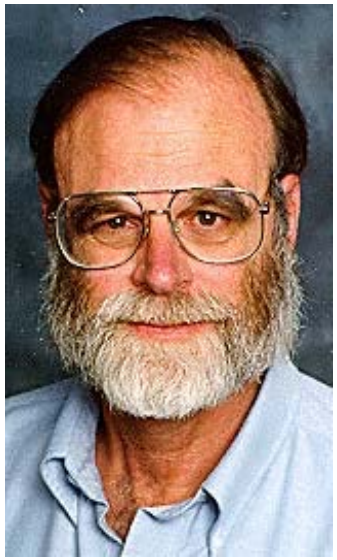
C. Date

IBM

System R

5W's: **Who** are the RDB pioneers?

J. Gray
IBM
Transactions
Db locking



Ray Boyce
IBM
SEQUEL
Normal form

D. Chamberlin
IBM
SEQUEL ->SQL
XQUERY

5 W's: **Why** databases?

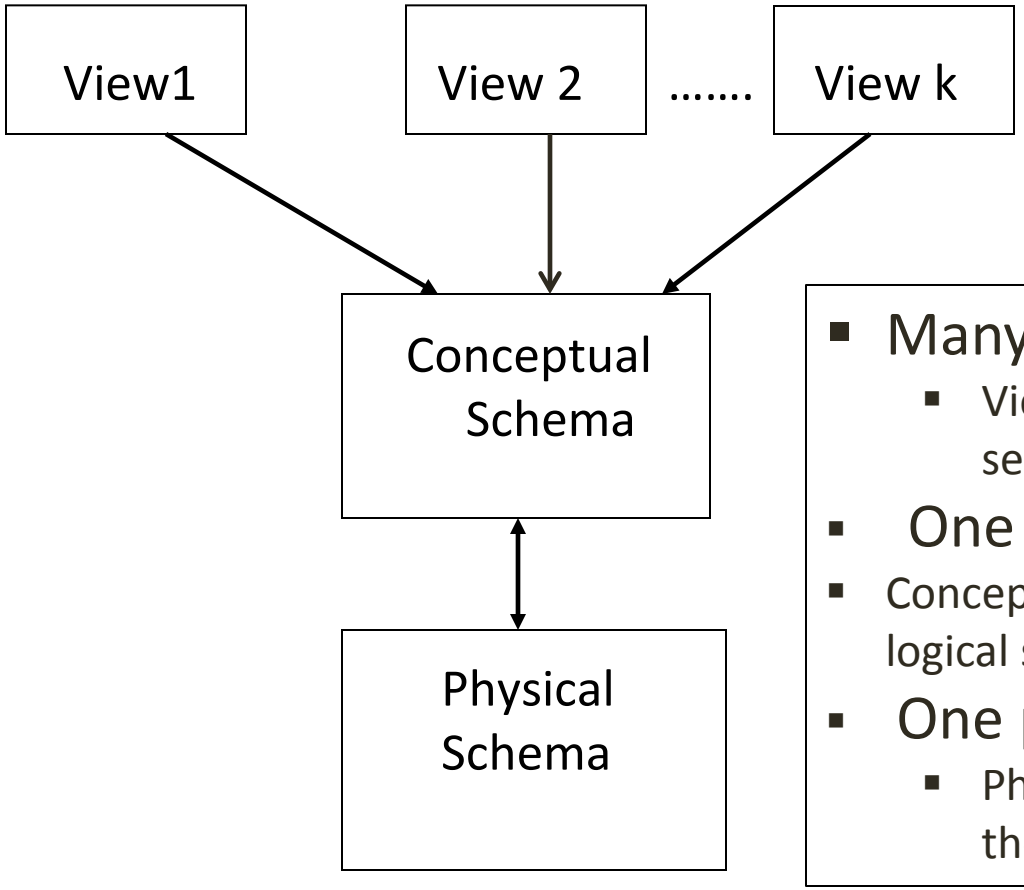
- ***Persistent Storage –*** Database not only provides persistent storage but also efficient access to large amounts of data
  - Reduced processing time
- ***Programming Interface –*** Database allows users to access and modify data using powerful query language. It provides flexibility in data management
  - Reduced application development time
- ***Transaction Management –*** Database supports a concurrent access to the data
  - Data independence and efficient access
  - Data integrity and security
  - Uniform data administration
  - Concurrent access, recovery from crashes

Data Models: Representation

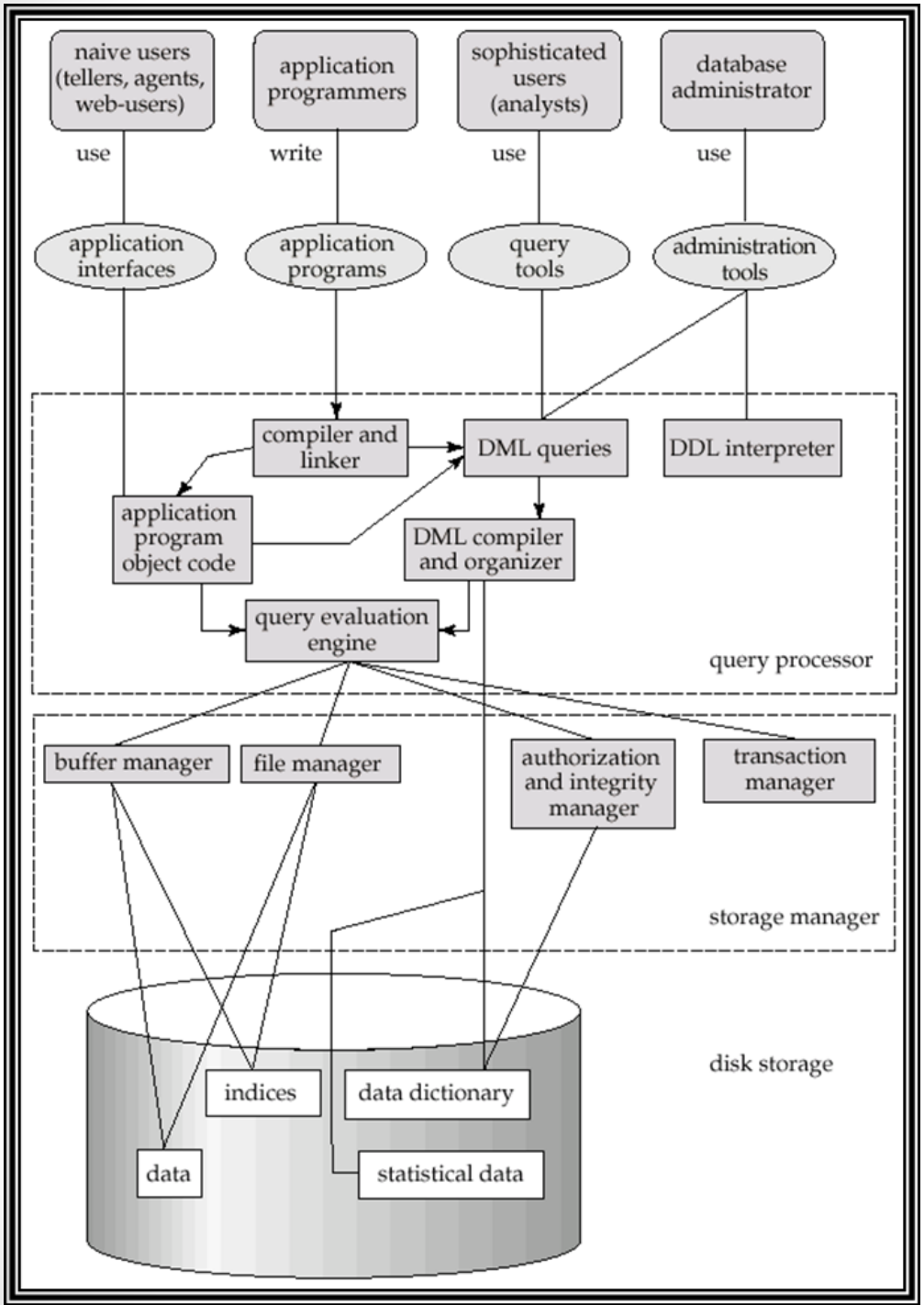
- Data model = collection of concepts for describing data.
  - Relational data model
  - Object oriented database model (OODMS)
  - XML data model
  - NoSQL database model
- Schema = description of a particular collection of data, using a given data model.
- The **relational data model** is the most widely used model today.
  - Main concept: relation= table
    - rows and columns
  - Every relation has a schema
    - Describes the columns or fields
  - Strength is in its simplicity

	Name	Age	…
Person1			
Person2			
Person3			
…			

Levels of abstraction



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



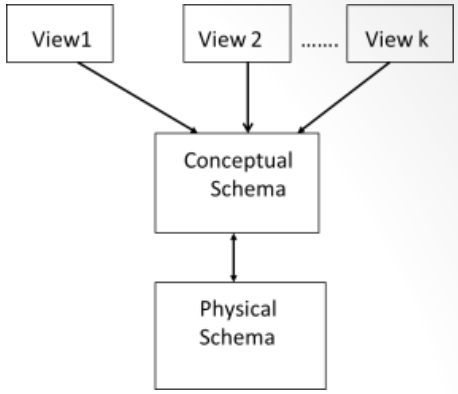
Layered Architecture of a DBMS

Properties of a RDBMS

- Atomicity
  - All or no changes done by a transaction
- Consistency
  - Only valid data written to the DB
- Isolation
  - Multiple transactions happening at the same time do not effect each other
- Durability
  - Changes committed to the DB are not lost

ACID test: important attributes of a RDBMS

Data Independence



- Applications are insulated from the physical layout of the data
- Logical data independence: Protection from changes in *logical* structure of data
  - Define a ‘view’ with the old logical structure
- Physical data independence: Protection from changes in *physical* structure of data.
  - Query and update logical structure, not physical structure

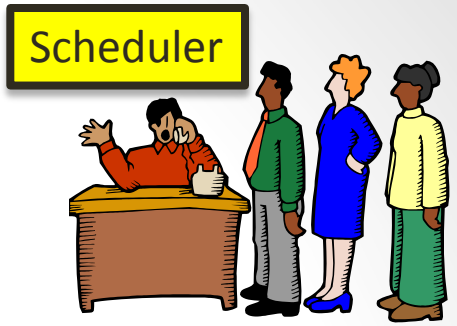
Concurrency control

- Concurrent execution of user programs is essential for good DBMS performance
  - Because disk accesses are frequent and relatively slow, it is important to keep the CPU humming by working on several user programs concurrently.
  - Interleaving actions of different user programs can lead to inconsistency in the database
    - E.g., check is cleared while account balance is being computed.
- **DBMS ensures such problems do not arise: users and programmers can pretend they are using a single-user system**.

Transaction processing

- Transaction = a*tomic* sequence of database actions (costly operations: reads/writes or disk access).
- Each transaction, executes completely, must leave the DB in a consistent state if DB is consistent when the transaction begins.
  - Users can specify integrity constraints on the data, and the DBMS will enforce these constraints.
    - Beyond this, the DBMS does not really understand the semantics of the data.
      - E.g., it does not understand how the interest on a bank account is computed.
  - Thus, ensuring that a single running transaction preserves the database consistency is ultimately the user's responsibility!

Scheduling concurrent transactions

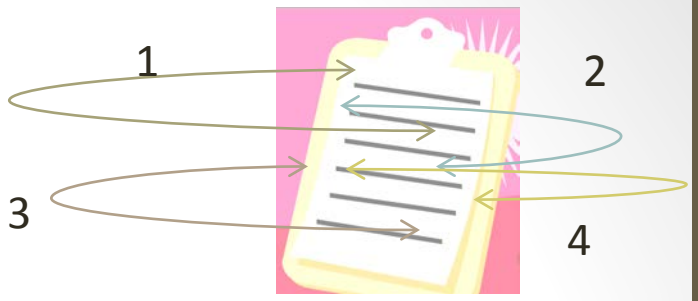


- DBMS ensures that the execution of {T1,..., Tn} is equivalent to some serial execution of T1',..., Tn'.
  - Before reading/writing an object, a transaction requests a lock on the object, and waits till the DBMS gives it the lock.
  - All locks are released at the end of the transaction
    - Strict 2 Phase locking protocol (2PL).
  - 2PL = If an action of Ti (say, writing X) affects Tj (which perhaps reads X), one of them, say Ti, will obtain the lock on X first and Tj is forced to wait until Ti completes; this effectively orders the transactions.
  - What if Tj already has a lock on Y and Ti later requests a lock on Y? (Deadlock!) Ti or Tj is aborted and restarted
    - Important to identify deadlocks and restart one of the transactions

Ensuring Atomicity

- DBMS ensures atomicity (all-or-nothing property) even if system crashes in the middle of a transaction
  - How?: Keeps a **log** (history) of all actions carried out by the DBMS while executing a set of transactions
  - Before a change is made to the database, the corresponding log entry is forced to a safe location. Write ahead logging protocol (WAL protocol)
  - After a crash, the effects of partially executed transactions are undone using the log.
    - Thanks to WAL, if a log entry was not saved before the crash, corresponding data change was not applied to the database

Log file operations



- Log is necessary to implement transaction processing
- The following actions are recorded in the log:
  - Ti writes an object: The old value and the new value
    - Log record must go to disk before the changed page
  - Ti commits/aborts: A log record indicating this action
- Log records chained together by transaction id
  - Quick/easy to undo a specific transaction
    - e.g., to resolve a deadlock
- Log is often *duplexed* and *archived* on "stable" storage.
- All log related work is handled transparently by the DBMS activities
  - Includes: all concurrency control related activities such as lock/unlock, dealing with deadlocks etc.

Conclusion

- Data and databases are ubiquitous
- DBMS are 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
- A DBMS typically has a layered architecture
- Data models and DBMS have been evolving for over 50 years
  - Mathematical theory led the way to a powerful relational data model
    - Relational revolution started with Codd's paper
      - Data represented as tables – operations defined by set theory
  - ACID properties necessary requirements for a RDMS
  - Relaxation or additional properties give rise to new data models
    - Object oriented data model (OODMS)
    - XML data model
    - NoSQL data model
    - New data models developed to address the evolution of data

Discussion

- Questions on class content or schedule?