CAS CS 460/660
Introduction to Database Systems

Fall 2017

About the course – Administrivia

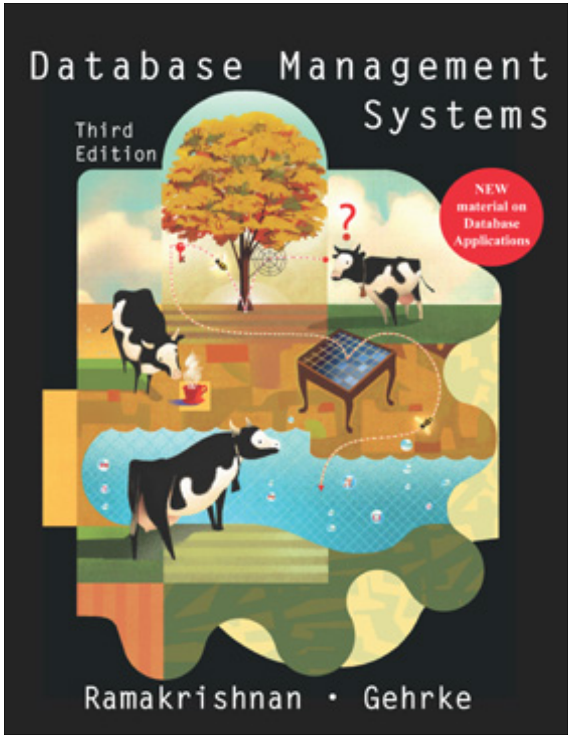
- Instructor:
  - George Kollios, gkollios@cs.bu.edu

    MCS 283, Mon 2:30-4:00 PM and Tue 1:00-2:30 PM
- Teaching Fellows:
  - Mona Jalal, jalal@bu.edu
    - EMA 309, Tue/Thu 2:00-3:15 PM and Fri 10:15-11:45 AM
  - Baichuan Zhou, baichuan@bu.edu
    - EMA 309, Wed 2:30-4:30 PM and Thu 2:30-4:30PM
- Home Page:
  - http://www.cs.bu.edu/fac/gkollios/cs460f17

    Check frequently! Syllabus, schedule, assignments, announcements…
  - Piazza site (you will be added soon)

Textbook

Raghu Ramakrishnan and Johannes Gehrke, "Database Management Systems", McGraw-Hill, Third Edition. 2002.


Database Management Systems
Third Edition
NEW material on Database Applications
Ramakrishnan • Gehrke

Grading

CS460

- Homeworks: 25%
- Midterm: 20%
- Final: 30%
- Programming Assignments: 25%

  examples:

  - Implement a Web application using a DBMS
  - Use a NoSQL system to analyze large datasets
  - (tentative) Use Amazon Cloud Services to perform data analysis on a large dataset

Grading

CS660

- Homeworks: 20%
- Midterm: 20%
- Final: 25%
- Programming Assignments: 25%
- Extra Assignments: 10%

What is a Database?

- Database:

  A very large collection (of files) of related data

- Examples: Accounts in a bank, BU's students database, Airline reservations… also, facebook pictures and comments, web logs, etc…
- Models a real world enterprise:
  - Entities (e.g., teams, games / students, courses)
  - Relationships (e.g., student takes CS460)
  - Even active components (e.g. "business logic")

What is a Data Base Management System?

- Data Base Management System (DBMS):

  A software package/system that can be used to store, manage and retrieve data from databases that persist for long periods of time!

- Examples: Oracle, IBM DB2, MS SQLServer, MySQL, PostgreSQL, SQLite,…

- Database System: DBMS+data (+ applications)

Why Study Databases??

- Shift from computation to data (information)
  - Always true for corporate computing
  - More and more true in the scientific world
  - and of course, Web
  - New trend: social media generate ever increasing amount of data, sensor devices generate also huge datasets
- DBMS encompasses much of CS in a practical discipline
  - OS, languages, theory, AI, logic

Why Databases??

- Why not store everything on flat files: use the file system of the OS, cheap/simple…

Name, Course, Grade

John Smith, CS112, B

Mike Stonebraker, CS234, A

Jim Gray, CS560, A

John Smith, CS560, B+

……………………

- Yes, but has many problems…

Problem 1

- Data Organization
  - redundancy and inconsistency
    - Multiple file formats, duplication of information in different files

Name, Course, Email, Grade

John Smith, js@cs.bu.edu, CS112, B

Mike Stonebraker, ms@cs.bu.edu, CS234, A

Jim Gray, CS560, jg@cs.bu.edu, A

John Smith, CS560, js@cs.bu.edu, B+

Why this is a problem?

- Wasted space
- Potential inconsistencies (multiple formats, John Smith vs Smith J.)

Problem 2

- Data retrieval:
  - Find the students registered for CS460
  - Find the students with GPA > 3.5

For every query we need to write a program!

- We need the retrieval to be:
  - Easy to write
  - Execute efficiently

Problem 3

- Data Integrity
  - No support for sharing:
    - Prevent simultaneous modifications
  - No coping mechanisms for system crashes
  - No means of Preventing Data Entry Errors (checks must be hard-coded in the programs)
  - Security problems
- Database systems offer solutions to all the above problems

Data Organization

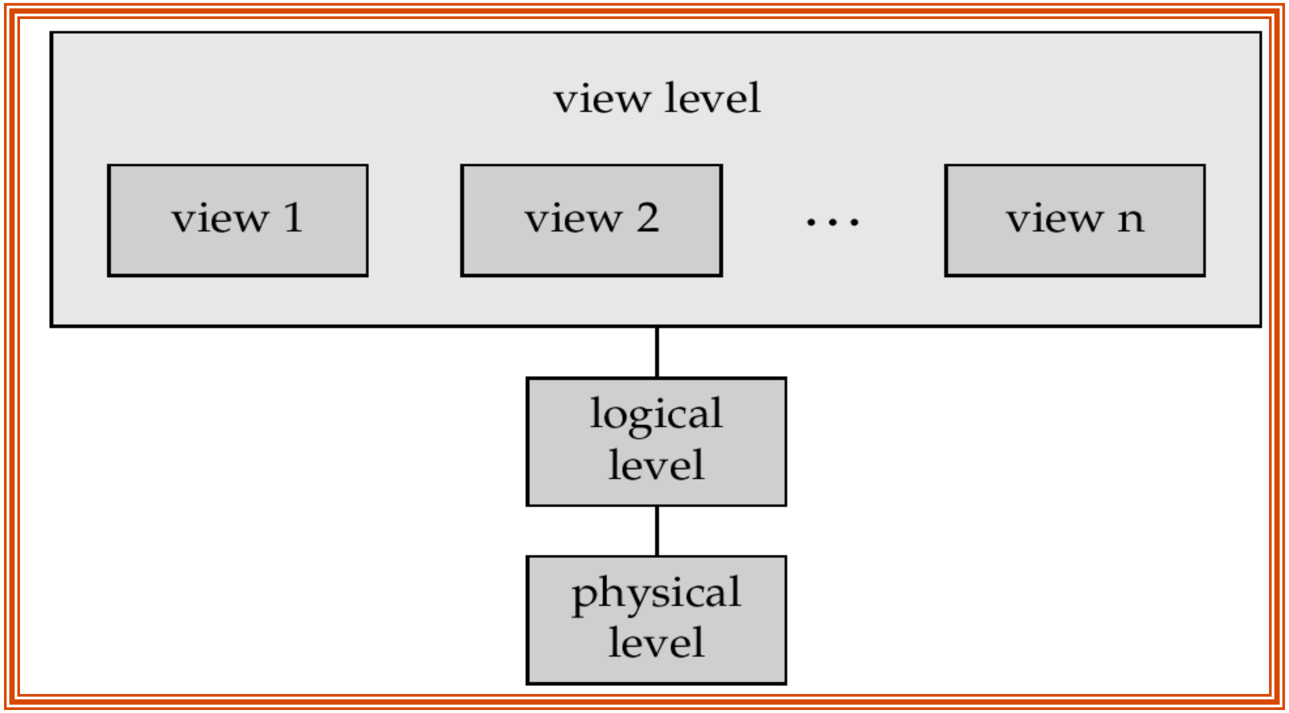
- Two levels of data modeling
- **Conceptual or Logical level**: describes data stored in database, and the relationships among the data.

```
type customer = record
                    name : string;
                    street : string;
                    city : integer;
                end;
```

- **Physical level:** describes how a record (e.g., customer) is stored.
- Also, **External (View) level**: application programs hide details of data types. Views can also hide information (e.g., salary) for security purposes.

View of Data

A logical architecture for a database system



Database Schema

- **Schema** – the structure of the database
  - e.g., the database consists of information about a set of customers and accounts and the relationship between them
  - Analogous to type information of a variable in a program
  - **Physical schema**: database design at the physical level
  - **Logical schema**: database design at the logical level

Data Organization

- **Data Models**: a framework for describing
  - data
  - data relationships
  - data semantics
  - data constraints
- Entity-Relationship model
- We will concentrate on Relational model
- Other models:
  - object-oriented model
  - semi-structured data models, XML

Relational Model

Attributes

- Example of tabular data in the relational model

Customer-id	*customer-name*	*customer-street*	*customer-city*	*account-number*
192-83-7465	Johnson	Alma	Palo Alto	A-101
019-28-3746	Smith	North	Rye	A-215
192-83-7465	Johnson	Alma	Palo Alto	A-201
321-12-3123	Jones	Main	Harrison	A-217
019-28-3746	Smith	North	Rye	A-201

Data Organization

- Data Storage

  Where can data be stored?

  - Main memory
  - Secondary memory (hard disks)
  - Optical storage (DVDs)
  - Tertiary store (tapes)
- Move data? Determined by *buffer manager*
- Mapping data to files? Determined by *file manager*

Data retrieval

- Queries

  Query = Declarative data retrieval

  *describes **what** data, not **how** to retrieve it*

  Ex. Give me the students with GPA > 3.5 vs

  Scan the student file and retrieve the records with gpa>3.5

- Why?

  1. Easier to write
  2. Efficient to execute (why?)

SQL

- SQL: widely used (declarative) non-procedural language
  - E.g. find the name of the customer with customer-id 192-83-7465

    select *customer.customer-name*
    from *customer*
    where *customer.customer-id* = '192-83-7465'

  - E.g. find the balances of all accounts held by the customer with customer-id 192-83-7465

    select *account.balance*
    from *depositor*, *account*
    where *depositor.customer-id* = '192-83-7465' **and**
    depositor.account-number = *account.account-number*

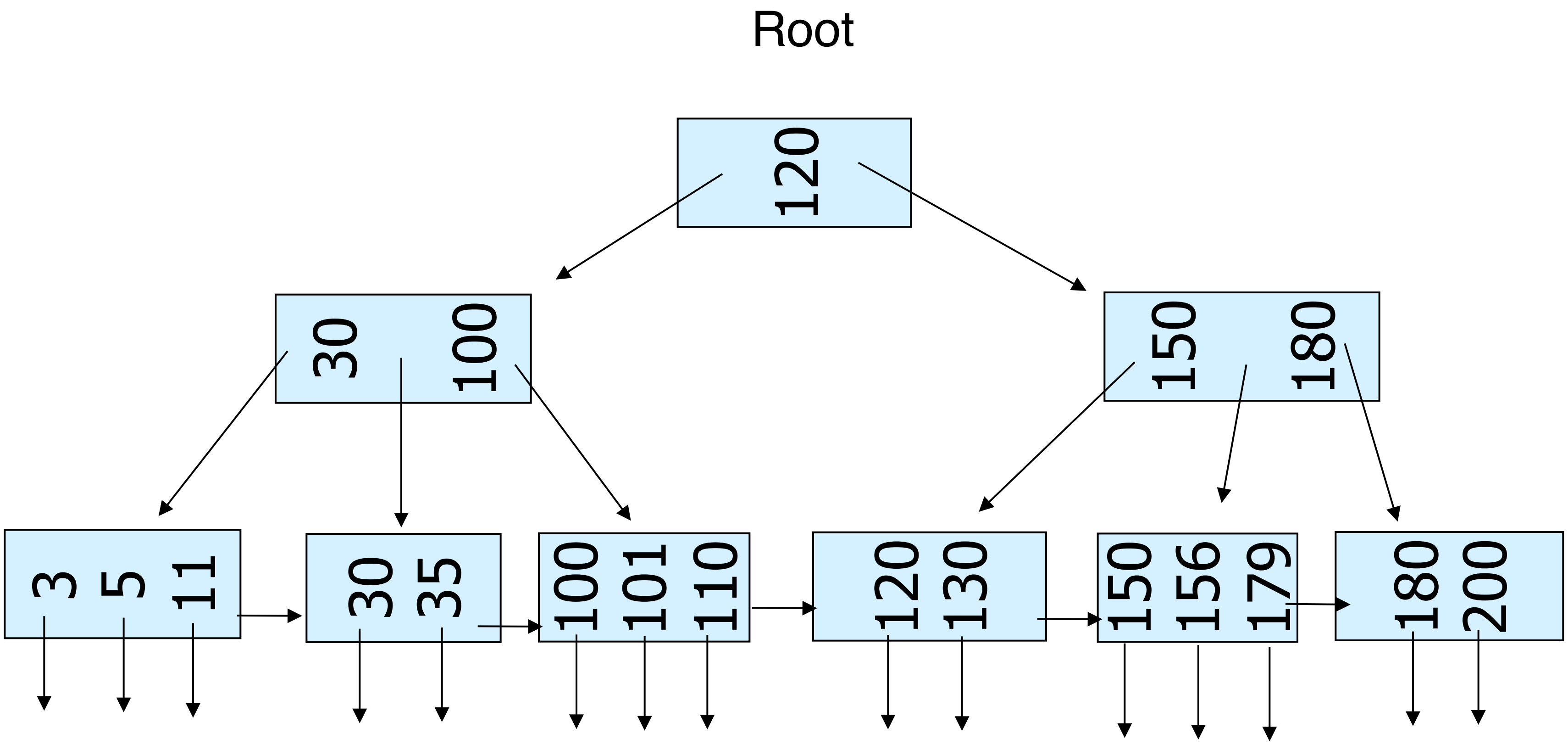
- Procedural languages: C++, Java, relational algebra

Data retrieval: Indexing

- How to answer fast the query: “Find the student with SID = 101”?
- One approach is to scan the student table, check every student, retrurn the one with id=101… very slow for large databases
- Any better idea?

1st keep student record over the SID. Do a binary search…. Updates…
2nd Use a dynamic search tree!! Allow insertions, deletions, updates and at the same time keep the records sorted! In databases we use the B+-tree (multiway search tree)
3rd Use a hash table. Much faster for exact match queries… but cannot support Range queries. (Also, special hashing schemes are needed for dynamic data)

B+Tree Example B=4



Data Integrity

Transaction processing

- Why Concurrent Access to Data must be Managed?

John and Jane withdraw $50 and $100 from a common account…

John:

1. get balance
2. if balance > $50
3. balance = balance - $50
4. update balance

Jane:

1. get balance
2. if balance > $100
3. balance = balance - $100
4. update balance

Initial balance $300. Final balance=?

It depends…

Data Integrity

Recovery

Transfer $50 from account A ($100) to account B ($200)

1. get balance for A
2. If balanceA > $50
3. balance$_A$ = balance$_A$ – 50

4.Update balance$_A$ in database

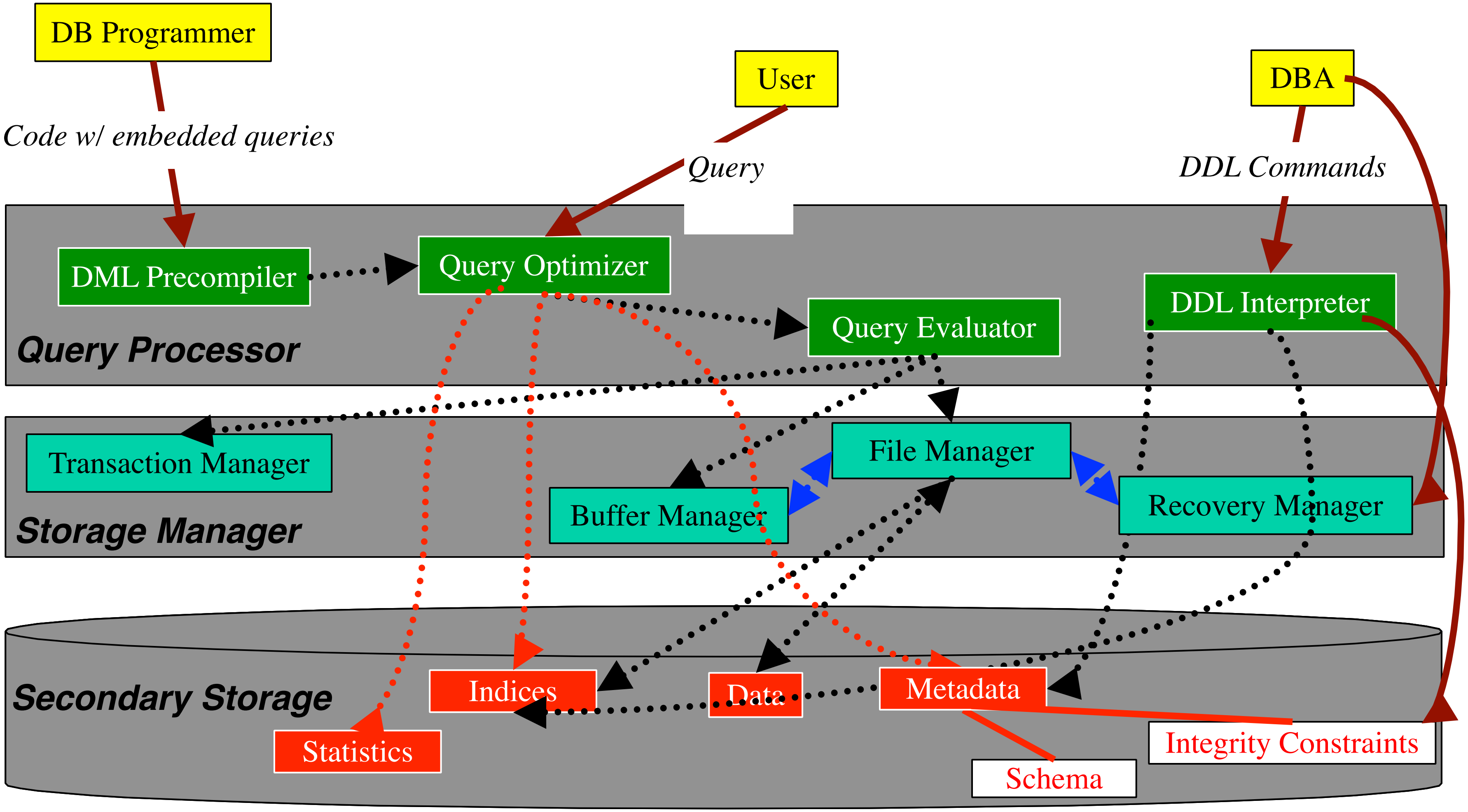
5. Get balance for B

System crashes….

6. balance$_B$ = balance$_B$ + 50
7. Update balance$_B$ in database

Recovery management

Database Architecture



Big Data and NoSQL

- Large amount of data are collected and stored everyday
  - Can come from different sources, huge amounts, large update rates
- Examples: facebook needs to handle: 2.7 billion “likes”, 400 million images, 500+ TB per day!!, Google receives more than 1 billion queries per day!
- Question: How to utilize these datasets in order to help us on our goals:
  - Data Analytics: Try to analyze the data in order to find useful, unknown and actionable information in the data
- Cluster based data analytics:
  - Map-Reduce, shared nothing DBs

- NoSQL: trade something for improved performance
    - (usually: ACID properties, flexibility, functionality)

Outline

- 1st half of the course: application-oriented
  - How to develop database applications: User + DBA

- 2nd part of the course: system-oriented
  - Learn the internals of a relational DBMS (developer for Oracle..)