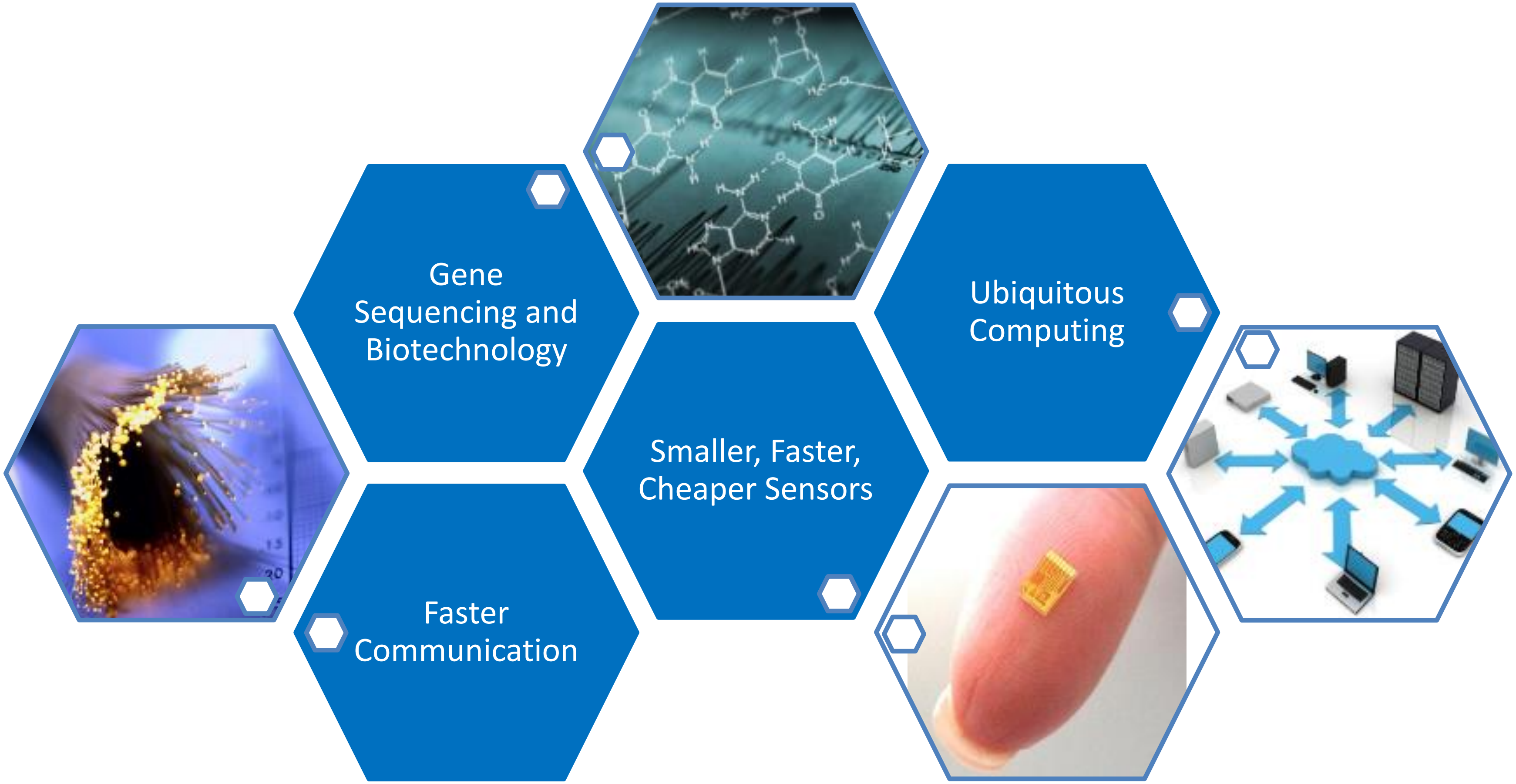
On the Verge of A Disruptive Century: Breakthroughs



Gene Sequencing and Biotechnology

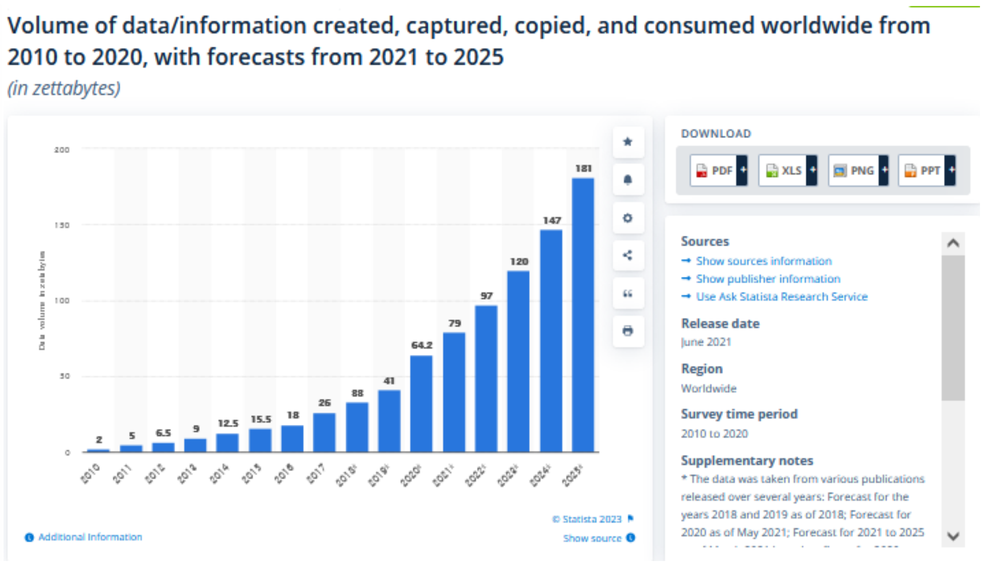
Ubiquitous Computing

Smaller, Faster, Cheaper Sensors

Faster Communication

A Common Theme is Data

The amount of data is only growing…
1.2 Zettabytes (1ZB = 10^{21} B or 1 Billion TB) in 2010



Volume of data/information created, captured, copied, and consumed worldwide from 2010 to 2020, with forecasts from 2021 to 2025

(in zettabytes)

Sources

- Show sources information
- Show publisher information
- Use Ask Statista Research Service

Release date

June 2021

Region

Worldwide

Survey time period

2010 to 2020

Supplementary notes

* The data was taken from various publications released over several years: Forecast for the years 2018 and 2019 as of 2018; Forecast for 2020 as of May 2021; Forecast for 2021 to 2025

We Live in a World of Data

- Nearly 500 Exabytes per day are generated by the Large Hadron Collider experiments (not all recorded!)

- 2.9 million emails are sent every second

- 20 hours of video are uploaded to YouTube every minute

- 24 PBs of data are processed by Google every day

- 50 million tweets are generated per day

- 700 billion total minutes are spent on Facebook each month

- 72.9 items are ordered on Amazon every second

Data and *Big* Data

- The value of data as an organizational asset is widely recognized

- Data is literally exploding and is occurring along three main dimensions
  - "Volume" or the amount of data
  - "Velocity" or the speed of data
  - "Variety" or the range of data types and sources

- What is Big Data?
  - It is the proliferation of data that floods organizations on a daily basis

  - It is *high* volume, *high* velocity, and/or *high* variety information assets

  - It requires new forms of processing to enable *fast* mining, enhanced decision-making, insight discovery and process optimization

What Do We Do With Data and Big Data?

Store

Share

Query

Mine

Encrypt

…. and more!

We want to do these *seamlessly* and *fast*...

Using Diverse Interfaces & Devices

Computers

Mobile Devices

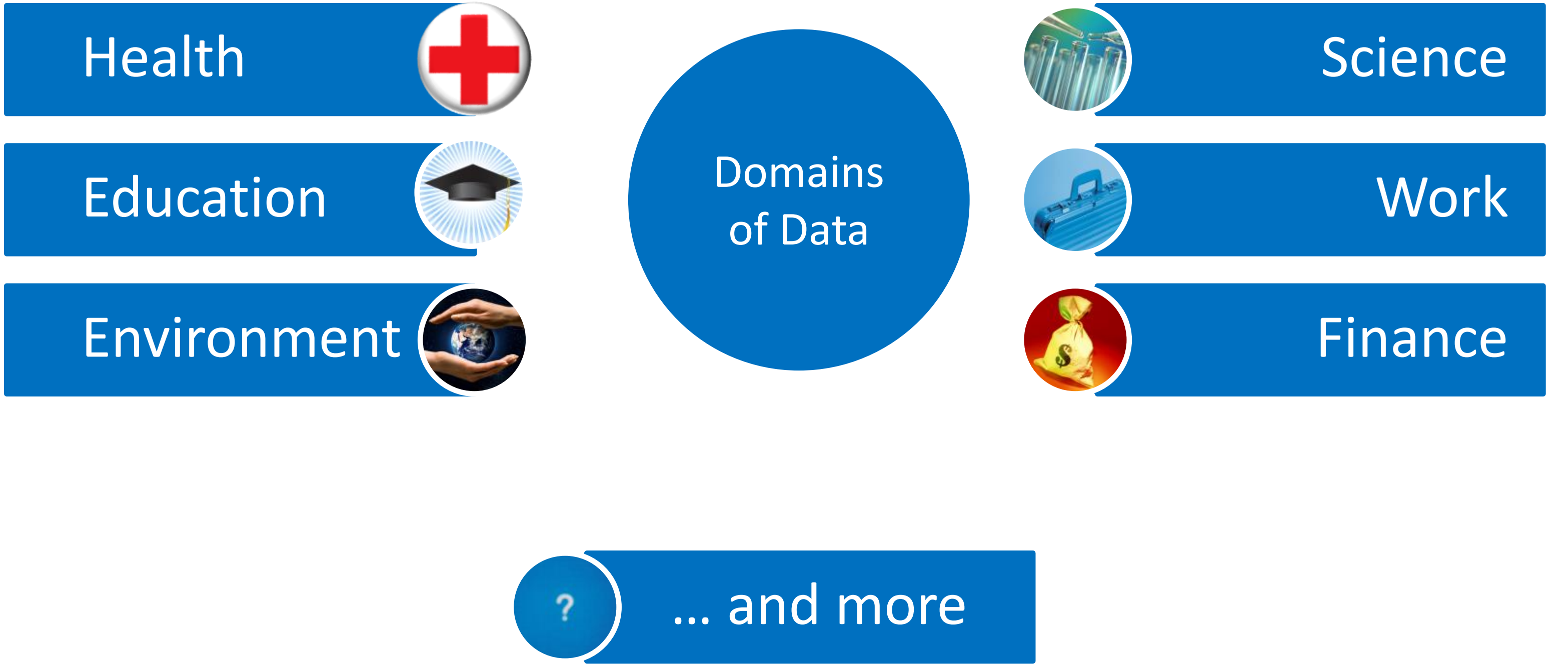
Consumer Electronics

Personal Monitors and Sensors

…and even appliances

We also want to access, share and process our data from all of our devices, **anytime, anywhere**!

Data is Becoming Critical to Our Lives



Domains of Data

Health

Education

Environment

Science

Work

Finance

… and more

Why Studying Databases?

- Data is *everywhere* and is *critical* to our lives

- Data need to be recorded, maintained, accessed and manipulated *correctly*, *securely*, *efficiently* and *effectively*
  - At the “low end”: scramble to web-scale (a mess!)
  - At the “high end”: scientific applications

- Database management systems (DBMSs) are indispensable software for achieving such goals

- The principles and practices of DBMSs are now an integral part of computer science curricula
  - They encompass OS, languages, theory, AI, multimedia, and logic, among others

As such, the study of database systems can prove to be richly rewarding in more ways than one!

Purpose of Database Systems

In the early days, database applications were built directly on top of file systems, which leads to:

- Data redundancy and inconsistency: data is stored in multiple file formats resulting induplication of information in different files
- Difficulty in accessing data
  - Need to write a new program to carry out each new task
- Data isolation
  - Multiple files and formats
- Integrity problems
  - Integrity constraints (e.g., account balance > 0) become "buried" in program code rather than being stated explicitly
  - Hard to add new constraints or change existing ones

Purpose of Database Systems (Cont.)

- Atomicity of updates
  - Failures may leave database in an inconsistent state with partial updates carried out
  - Example: Transfer of funds from one account to another should either complete or not happen at all
- Concurrent access by multiple users
  - Concurrent access needed for performance
  - Uncontrolled concurrent accesses can lead to inconsistencies
    - Ex: Two people reading a balance (say 100) and updating it by withdrawing money (say 50 each) at the same time
- Security problems
  - Hard to provide user access to some, but not all, data

Database systems offer solutions to all the above problems

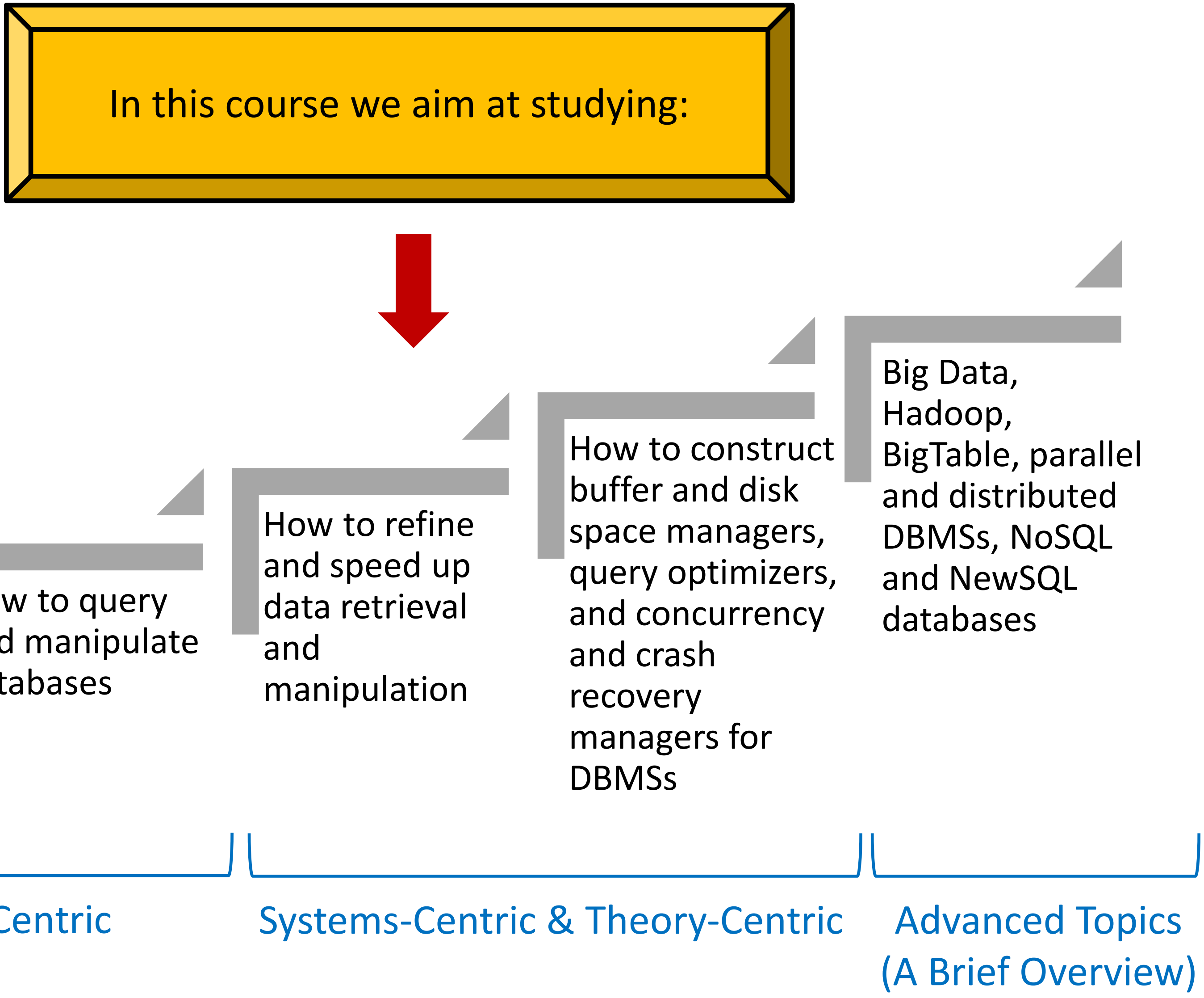
Database Applications Examples

- Enterprise Information
  - Sales: customers, products, purchases
  - Accounting: payments, receipts, assets
  - Human Resources: Information about employees, salaries, payroll taxes.
- Manufacturing: management of production, inventory, orders, supply chain.
- Banking and finance
  - customer information, accounts, loans, and banking transactions.
  - Credit card transactions
  - Finance: sales and purchases of financial instruments (e.g., stocks and bonds; storing real-time market data
- Universities: registration, grades

Database Applications Examples (Cont.)

- Airlines: reservations, schedules
- Telecommunication: records of calls, texts, and data usage, generating monthly bills, maintaining balances on prepaid calling cards
- Web-based services
  - Online retailers: order tracking, customized recommendations
  - Online advertisements
- Document databases
- Navigation systems: For maintaining the locations of varies places of interest along with the exact routes of roads, train systems, buses, etc.

Course Objectives



In this course we aim at studying:

How to design and implement databases from ‘cradle-to-grave’

How to query and manipulate databases

How to refine and speed up data retrieval and manipulation

How to construct buffer and disk space managers, query optimizers, and concurrency and crash recovery managers for DBMSs

Big Data, Hadoop, BigTable, parallel and distributed DBMSs, NoSQL and NewSQL databases

Application-Centric

Systems-Centric & Theory-Centric

Advanced Topics (A Brief Overview)

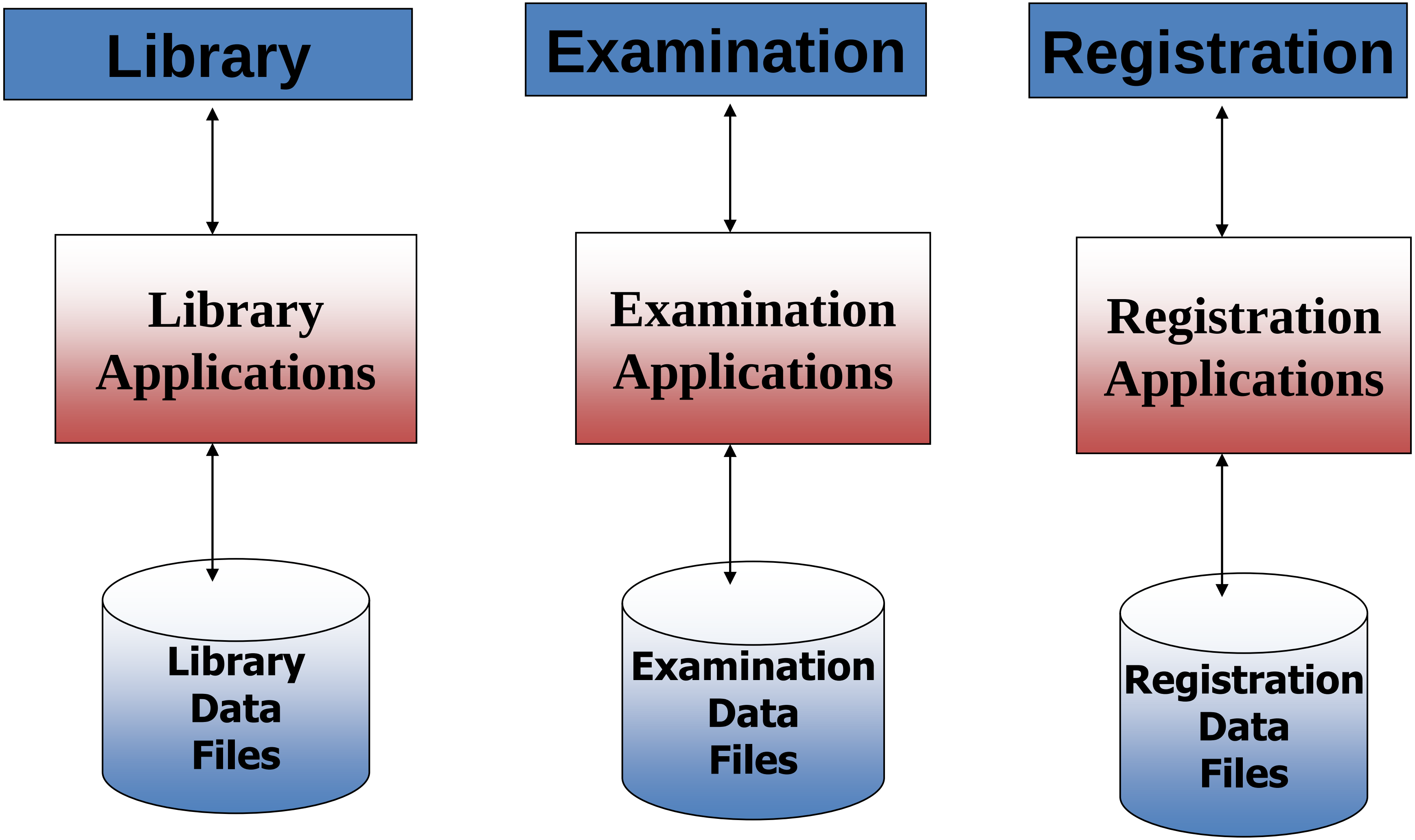
A bit of History

- Computer initially used for computational/ engineering purposes
- Commercial applications introduced File Processing System

File Processing System

- A collection of application programs that perform services for the end-users such as production of reports
- Each program defines and manages its own data

File Processing Systems



Program and Data Interdependence

File Processing Systems

Library
Reg_Number
Name
Father Name
Books Issued
Fine

Examination
Reg_Number
Name
Address
Class
Semester
Grade

Registration
Reg_Number
Name
Father Name
Phone
Address
Class

Files Based Processing

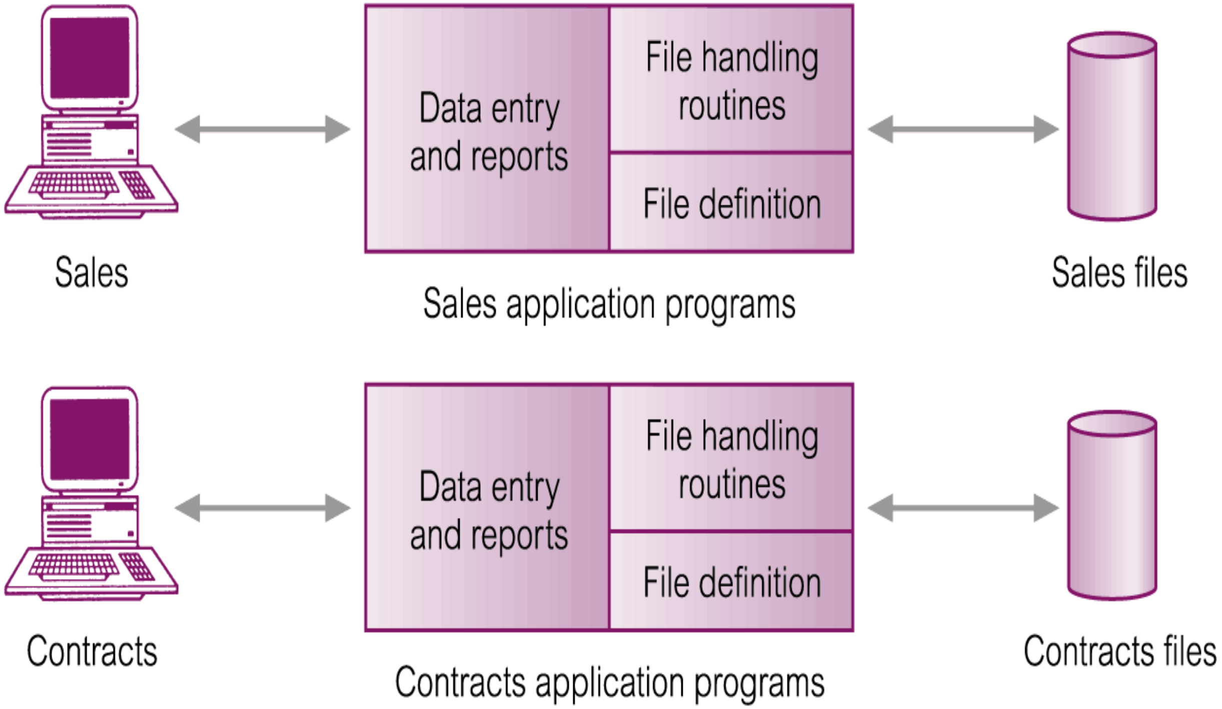


Figure 1.5
File-based processing.

Disadvantages of File Processing

- **Program-Data Dependence**
  - File structure is defined in the program code.
  - All programs maintain metadata for each file they use
- **Duplication of Data (Data Redundancy)**
  - Different systems/programs have separate copies of the same data
  - Same data is held by different programs.
  - Wasted space and potentially different values and/or different formats for the same item.
- **Limited Data Sharing**
  - No centralized control of data
  - Programs are written in different languages, and so cannot easily access each other’s files.

Disadvantages of File Processing

- **Lengthy Development Times**
  - Programmers must design their own file formats
- **Excessive Program Maintenance**
  - 80% of of information systems budget
- **Vulnerable to Inconsistency**
  - Change in one table need changes in corresponding tables as well otherwise data will be inconsistent

Problems with Data Dependency

- Each application programmer must maintain their own data
- Each application program needs to include code for the metadata of each file
- Each application program must have its own processing routines for reading, inserting, updating and deleting data
- Lack of coordination and central control
- Non-standard file formats

Problems with Data Redundancy

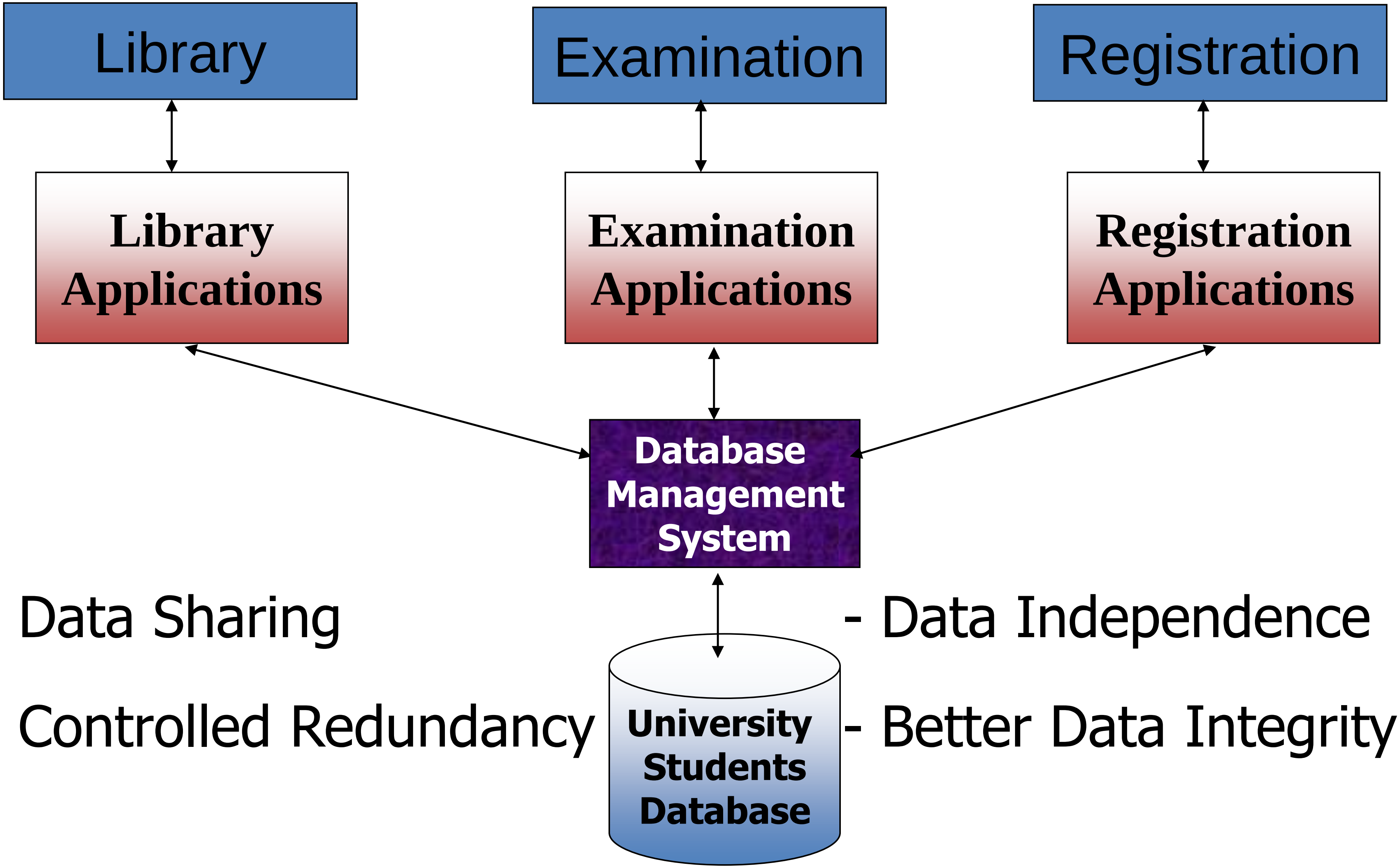
- Waste of space to have duplicate data
- Causes more maintenance headaches
- The biggest problem:
  - **When data changes in one file, could cause inconsistencies (Vulnerable to Inconsistency)**
  - Compromises ***data integrity (data reliability)***

SOLUTION:
The DATABASE Approach

- Central repository of shared data
- Data is managed by a controlling agent
- Stored in a standardized, convenient form

This requires a
Database and Database Management System (DBMS)

Advantages of Database Approach



Library

Examination

Registration

Library Applications

Examination Applications

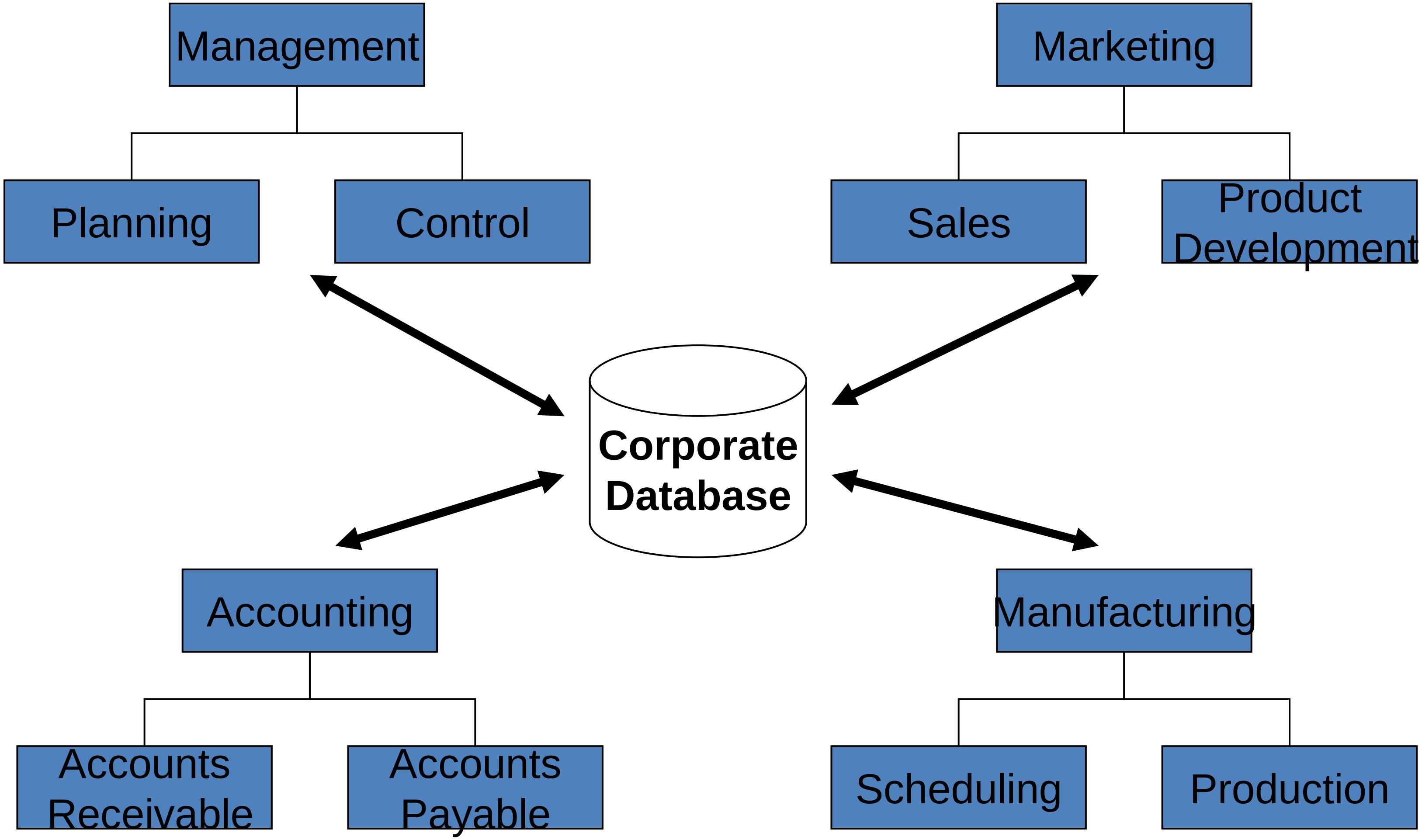
Registration Applications

Database Management System

University Students Database

- Data Sharing
- Controlled Redundancy
- Data Independence
- Better Data Integrity

The concept of a shared organizational database



A Motivating Scenario

- PSTU has a “large” collection of data (say 500GB) on employees, students, universities, research centers, etc.,
- **Performance (Concurrency Control)**
- **Performance (Response Time)**
- **Correctness (Consistency)**
- **Correctness (Security)**
- **Correctness (Durability and Atomicity)**

Managing Data using File Systems

- What about managing PSTU data using local file systems?
  - Files of fixed-length and variable-length records as well as formats
  - Main memory vs. disk
  - Computer systems with 32-bit addressing vs. 64-bit addressing schemes
  - Special programs (e.g., C++ and Python programs) for answering user questions
  - Special measures to maintain atomicity
  - Special measures to maintain consistency of data
  - Special measures to maintain data isolation
  - Special measures to offer software and hardware fault-tolerance
  - Special measures to enforce security policies in which different users are granted different permissions to access diverse subsets of data

This becomes tedious and inconvenient, especially at large-scale, with evolving/new user queries and higher probability of failures!

List of Topics

○ **Considered:** a reasonably critical and comprehensive understanding.

◐ **Thoughtful:** fluent, flexible and efficient understanding.

● **Masterful:** a powerful and illuminating understanding.

● .1. The Entity-Relationship Model

● .2. The Relational Model

● .3. Relational Algebra and Calculus

● .4. SQL

● .5. Data Storage and Organization

● .6. Tree-Based and Hash-Based Indexing

◐ .7. Query Evaluation and Optimization

● .9. Concurrency Control and Crash Recovery

○ .10. Advanced Topics: Distributed Databases, Hadoop, and NoSQL and NewSQL Databases

Learning Outcomes

❖ After finishing this course you will be able to:

1. Describe a wide range of data involved in real-world organizations using the entity-relationship (ER) data model

2. Explain how to translate an ER diagram into a relational database

3. Analyze and apply a formal query language, relational calculus and algebra

4. Indicate how SQL builds upon relational calculus and algebra and effectively apply SQL to create, query and manipulate relational databases

5. Design and develop multi-tiered, full-fledged standalone and web-based applications with back-end databases

6. Appreciate how DBMSs create, manipulate and manage files of fixed-length and variable-length records on disks

Learning Outcomes

- After finishing this course you will be able to:

7. Create and operate various static and dynamic tree-based (e.g., ISAM and B+ trees) and hash-based (e.g., extendable and linear hashing) indexing schemes

8. Explain and evaluate various algorithms for relational operations (e.g., join) using techniques such as iteration, indexing and partitioning

9. Analyze and apply different query evaluation plans and describe the various tasks of a typical relational query optimizer

10. Describe how transactions can be interleaved correctly, and indicate how a DBMS can ensure atomicity and durability when systems fail or entirely crash

11. Identify alternative architectures for distributed databases, and describe how data can be partitioned and distributed across networked nodes of a DBMS

12. Appreciate the scale of Big Data, discuss some popular analytics engines for Big Data processing and denote the applicability of NoSQL databases for Big Data storage

Teaching Methods, Assignments and Projects

Lectures

- Motivate learning
- Provide a framework or roadmap to organize the information of the course
- Explain subjects and reinforce the critical big ideas

Recitations

- Get you to reveal what you do not understand, so we can help you
- Allow you to practice skills you will need to become an expert

5 Assignments

- We will have 5 assignments which involve problem solving and span most of the topics that we discuss in the class

Projects

- We will have 3 projects which involve using SQL, Oracle

Data Base Management Systems

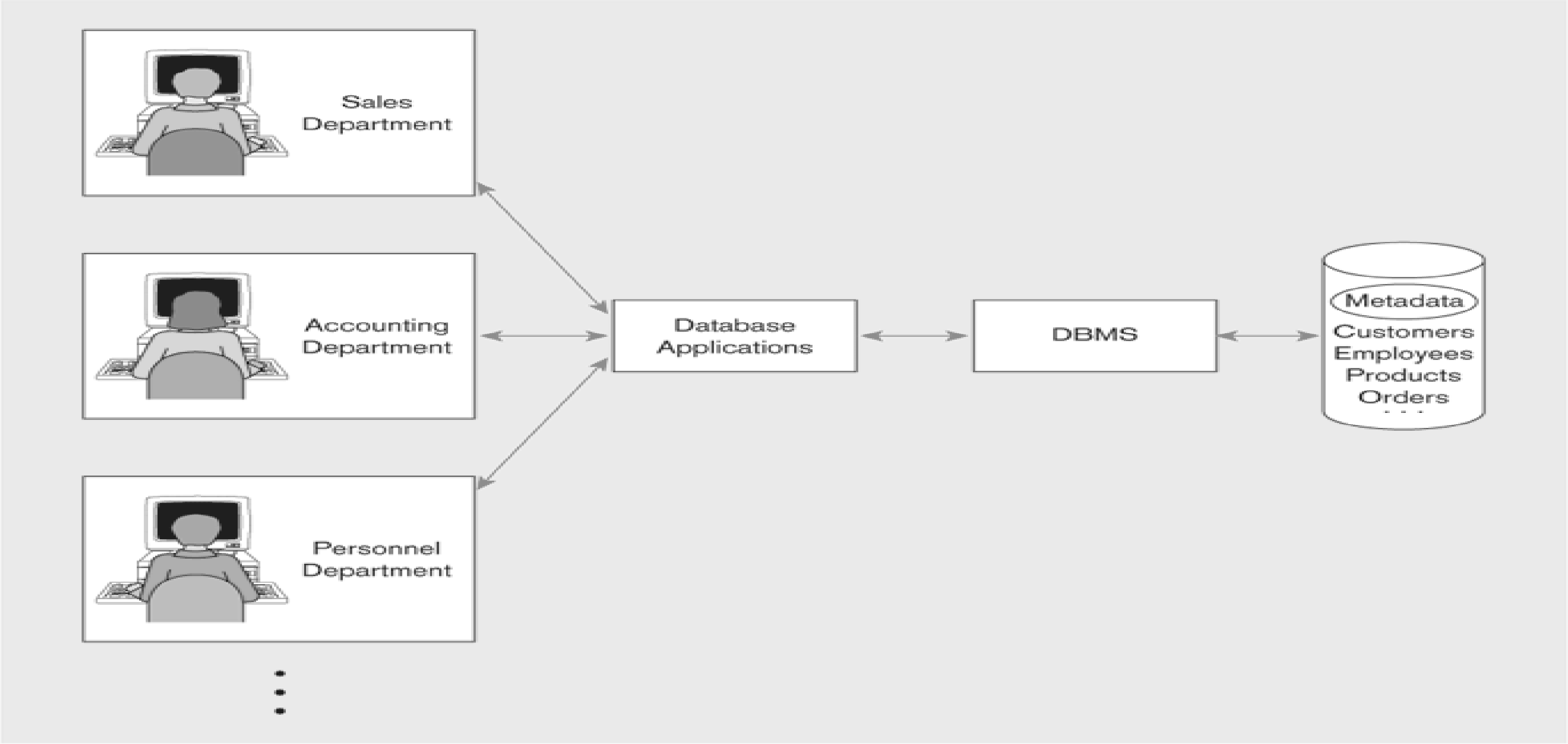
- A special software is accordingly needed to make the preceding tasks easier

- This software is known as Data Base Management System (DBMS)

- DBMSs provide automatic:
  - Data independence
  - Efficient data access
  - Data integrity and security
  - Data administration
  - Concurrent access and crash recovery
  - Reduced application development and tuning time

Database Management System

Figure 1-3 Database approach at Pine Valley Furniture Company



DBMS manages data resources like an operating system manages hardware resources

An Example

- UNIVERSITY database
  - Information concerning students, courses, and grades in a university environment
- **Data records**
  - STUDENT
  - COURSE
  - SECTION
  - GRADE_REPORT
  - PREREQUISITE

An Example (cont'd.)

- Specify structure of records of each file by specifying **data type** for each **data element**
  - String of alphabetic characters
  - Integer
  - Etc.

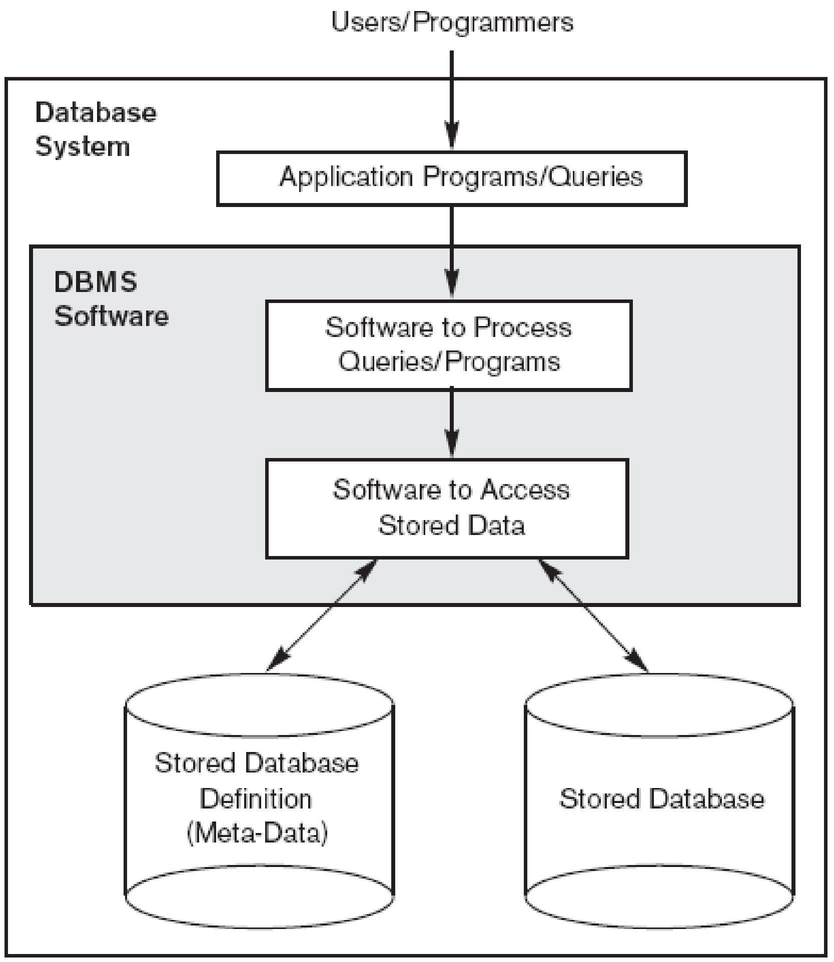

Users/Programmers
Database System
Application Programs/Queries
DBMS Software
Software to Process Queries/Programs
Software to Access Stored Data
Stored Database Definition (Meta-Data)
Stored Database

Figure 1.1
A simplified database system environment.

Addison-Wesley
is an imprint of
PEARSON

An Example (cont'd.)

- Construct UNIVERSITY database
  - Store data to represent each student, course, section, grade report, and prerequisite as a record in appropriate file
- Relationships among the records
- Manipulation involves querying and updating

An Example (cont'd.)

- Examples of queries:
  - Retrieve the transcript
  - List the names of students who took the section of the 'Database' course offered in fall 2008 and their grades in that section
  - List the prerequisites of the 'Database' course

Addison-Wesley
is an imprint of
PEARSON

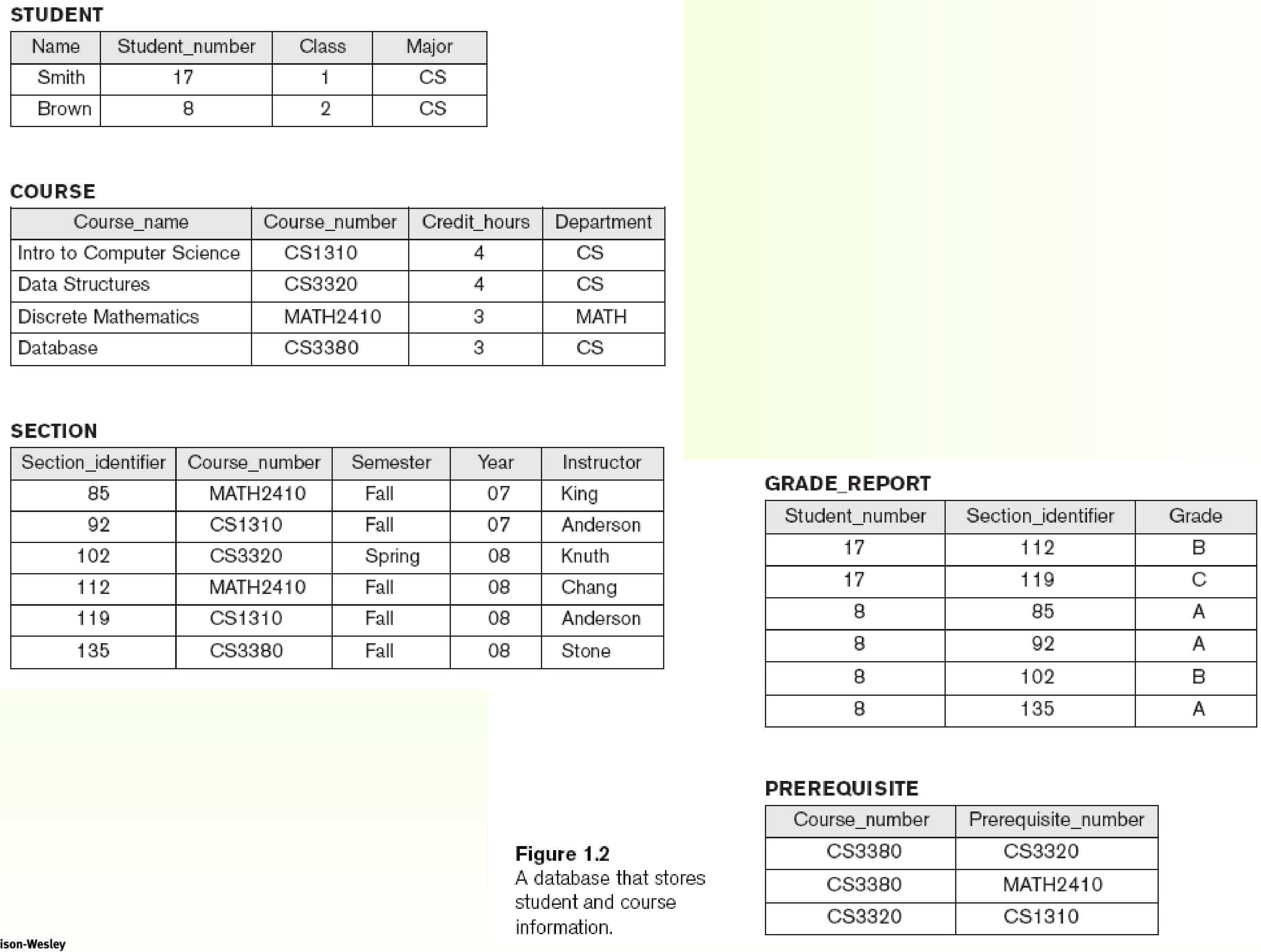
An Example (cont'd.)

- Examples of updates:
  - Change the class of ‘Smith’ to sophomore
  - Create a new section for the ‘Database’ course for this semester
  - Enter a grade of ‘A’ for ‘Smith’ in the ‘Database’ section of last semester

Addison-Wesley
is an imprint of
PEARSON

An Example (cont'd.)

- Phases for designing a database:
  - **Requirements specification and analysis**
  - **Conceptual design**
  - **Logical design**
  - **Physical design**



STUDENT

Name	Student_number	Class	Major
Smith	17	1	CS
Brown	8	2	CS

COURSE

Course_name	Course_number	Credit_hours	Department
Intro to Computer Science	CS1310	4	CS
Data Structures	CS3320	4	CS
Discrete Mathematics	MATH2410	3	MATH
Database	CS3380	3	CS

SECTION

Section_identifier	Course_number	Semester	Year	Instructor
85	MATH2410	Fall	07	King
92	CS1310	Fall	07	Anderson
102	CS3320	Spring	08	Knuth
112	MATH2410	Fall	08	Chang
119	CS1310	Fall	08	Anderson
135	CS3380	Fall	08	Stone

GRADE_REPORT

Student_number	Section_identifier	Grade
17	112	B
17	119	C
8	85	A
8	92	A
8	102	B
8	135	A

PREREQUISITE

Course_number	Prerequisite_number
CS3380	CS3320
CS3380	MATH2410
CS3320	CS1310

Figure 1.2
A database that stores student and course information.

Addison-Wesley
is an imprint of
PEARSON

Some Definitions

- A database is a collection of data which describes one or many real-world enterprises
  - E.g., a university database might contain information about entities like students and courses, and relationships like a student enrollment in a course

- A DBMS is a software package designed to store and manage databases
  - E.g., DB2, Oracle, MS SQL Server, MySQL and Postgres

- A database system = (Big) Data + DBMS + Application Programs

Definitions of Database

- ***Def 1:*** Database is an *organized collection* of *logically related data*
- ***Def 2:*** A database is a *shared* collection of *logically related* *data* that is stored to meet the requirements of *different users* of an organization
- ***Def 3:*** A database is a *self-describing* collection of integrated records
- ***Def 4:*** A database models a *particular real world system* in the computer in the form of data

Definitions

- ***Data***: stored representations of meaningful objects and events or
- Referred to facts concerning objects and events that could be recorded and stored on computer media
  - Structured: numbers, text, dates
  - Unstructured: images, video, documents
- ***Information***: data processed to increase knowledge in the person using the data
- ***Metadata:*** data that describes the properties and context of user data

Data Models

- The user of a DBMS is ultimately concerned with some real-world enterprises (e.g., a University)

- The data to be stored and managed by a DBMS *describes* various aspects of the enterprises
  - E.g., The data in a university database describes students, faculty and courses entities and the relationships among them

- A data model is a collection of high-level data description constructs that hide many low-level storage details

- A widely used data model called the entity-relationship (ER) model allows users to pictorially denote entities and the relationships among them

The Relational Model

- The relational model of data is one of the most widely used models today

- The central data description construct in the relational model is the relation

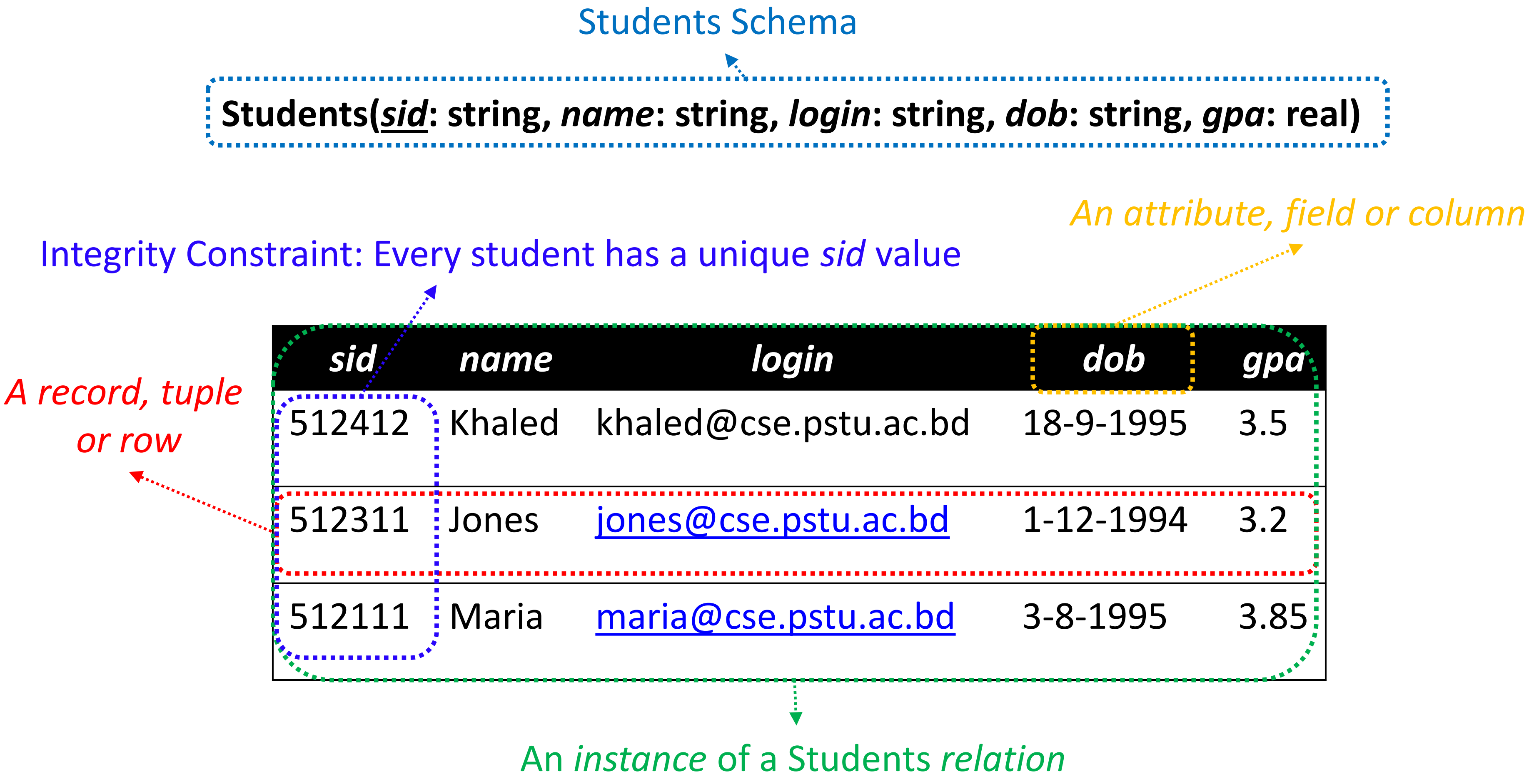
- A relation is basically a table (or a set) with rows (or records or tuples) and columns (or fields or attributes)

- Every relation has a schema, which describes the columns of a relation

- Conditions that records in a relation must satisfy can be specified
  - These are referred to as integrity constraints

The Relational Model: An Example

- Let us consider the student entity in a university database



Students Schema

Students(*sid*: string, *name*: string, *login*: string, *dob*: string, *gpa*: real)

An attribute, field or column

Integrity Constraint: Every student has a unique *sid* value

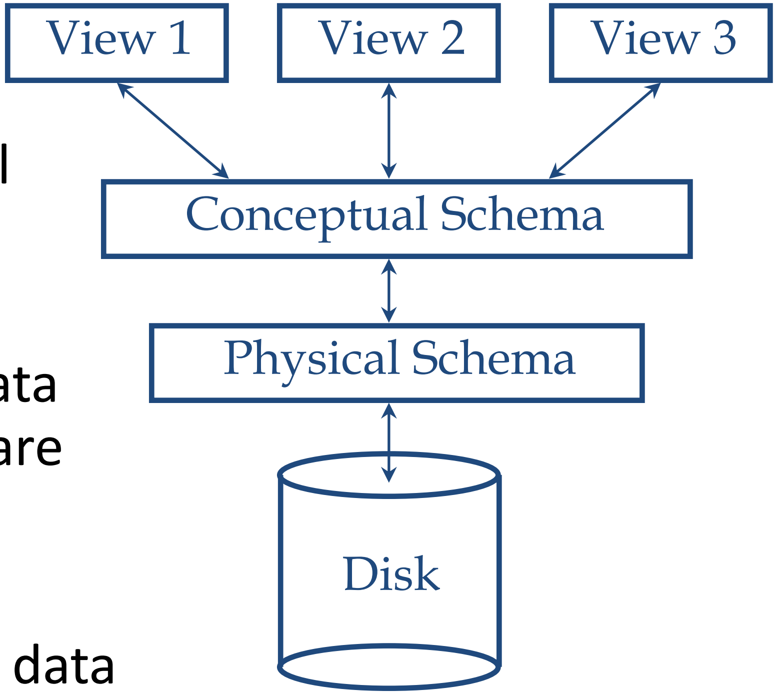
A record, tuple or row

sid	***name***	***login***	***dob***	***gpa***
512412	Khaled	khaled@cse.pstu.ac.bd	18-9-1995	3.5
512311	Jones	jones@cse.pstu.ac.bd	1-12-1994	3.2
512111	Maria	maria@cse.pstu.ac.bd	3-8-1995	3.85

An *instance* of a Students *relation*

Levels of Abstraction

- The data in a DBMS is described at three levels of abstraction, the conceptual (or logical), physical and external schemas



- The conceptual schema describes data in terms of a specific data model (e.g., the relational model of data)
- The physical schema specifies how data described in the conceptual schema are stored on secondary storage devices
- The external schema (or views) allow data access to be customized at the level of individual users or group of users (views can be 1 or many)

Views

- A view is conceptually a relation

- Records in a view are computed as needed and usually not stored in a DBMS

- Example: University Database

Conceptual Schema	Physical Schema	External Schema (View)
• Students(sid: string, name: string, login: string, dob: string, gpa:real) • Courses(cid: string, cname:string, credits:integer) • Enrolled(sid:string, cid:string, grade:string)	• Relations stored as heap files • Index on first column of Students	Students can be allowed to find out course enrollments: • Course_info(cid: string, enrollment: integer)

Can be computed from the relations in the conceptual schema (so as to avoid data redundancy and inconsistency).

Iterating: Data Independence

- One of the most important benefits of using a DBMS is data independence

- With data independence, application programs are insulated from how data are structured and stored

- Data independence entails two properties:
  - Logical data independence: users are shielded from changes in the conceptual schema (e.g., add/drop a column in a table)
  - Physical data independence: users are shielded from changes in the physical schema (e.g., add index or change record order)

Queries in a DBMS

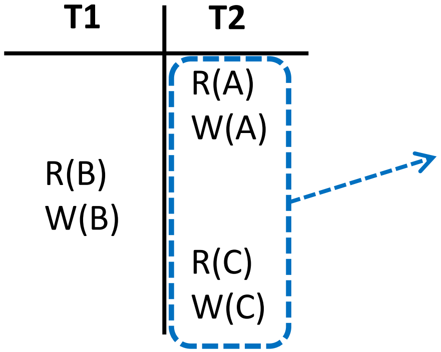
- The ease with which information can be queried from a database determines its value to users

- A DBMS provides a specialized language, called the query language, in which queries can be posed

- The relational model supports powerful query languages
  - Relational calculus: a formal language based on mathematical logic
  - Relational algebra: a formal language based on a collection of operators (e.g., selection and projection) for manipulating relations
  - Structured Query Language (SQL):
    - Builds upon relational calculus and algebra
    - Allows creating, manipulating and querying relational databases
    - Can be embedded within a host language (e.g., Java)

Concurrent Execution and Transactions

- An important task of a DBMS is to *schedule* concurrent accesses to data so as to improve performance



T1	T2
	R(A)
	W(A)
R(B)	
W(B)	
	R(C)
	W(C)

An *atomic* sequence of database actions (read/writes) is referred to as “*transaction*”

- When several users access a database *concurrently*, the DBMS must order their requests carefully to avoid conflicts
  - E.g., A check might be cleared while account balance is being computed!

- DBMS ensures that conflicts do not arise via using a locking protocol
  - Shared vs. Exclusive locks

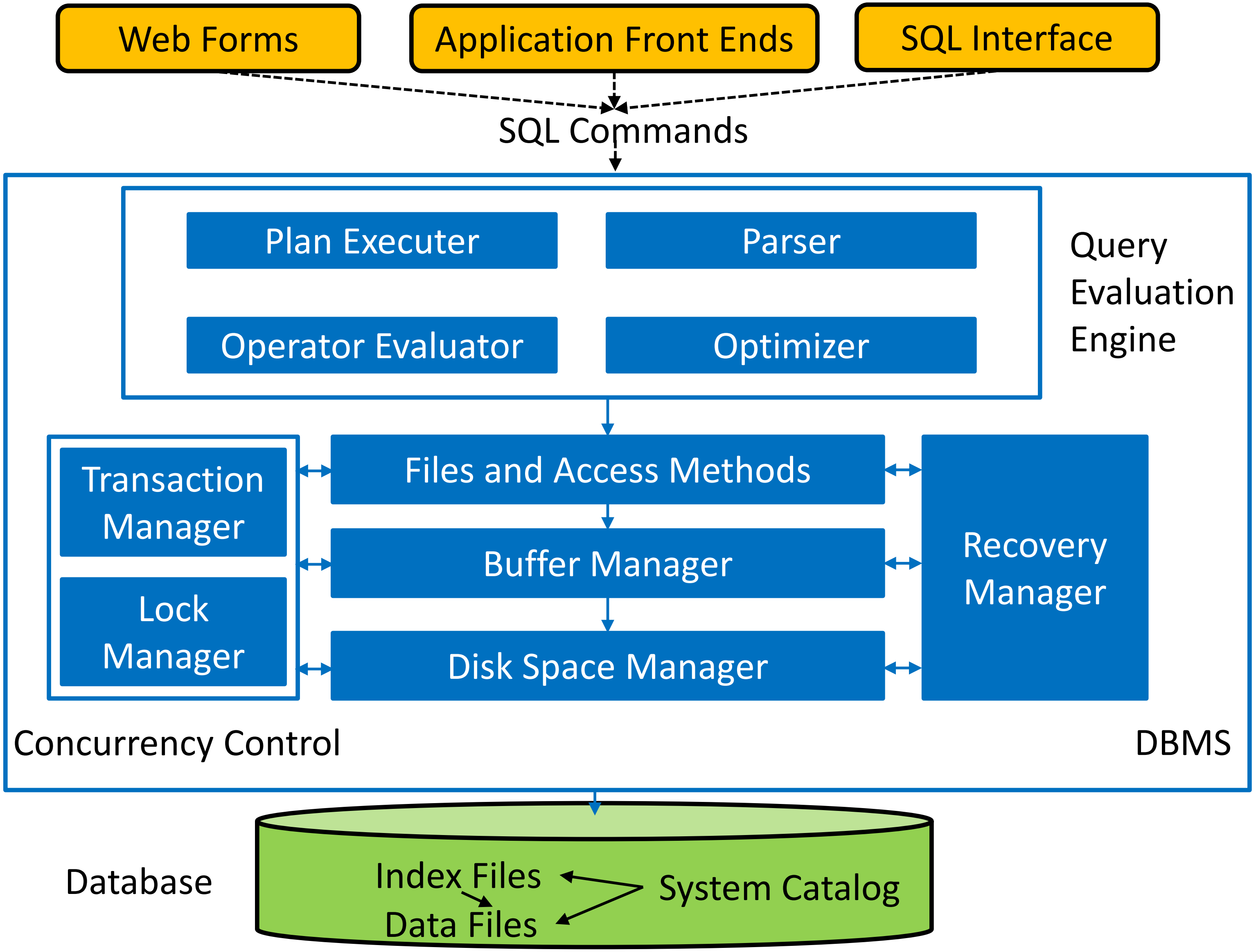
Ensuring Atomicity

- Transactions can be interrupted before running to completion for a variety of reasons (e.g., due to a system crash)

- DBMS ensures atomicity (all-or-nothing property) even if a crash occurs in the middle of a transaction

- This is achieved via maintaining a log (i.e., history) of all writes to the database
  - *Before* a change is made to the database, the corresponding log entry is forced to a safe location (this protocol is called Write-Ahead Log or WAL)
  - After a crash, the effects of partially executed transactions are *undone* using the log

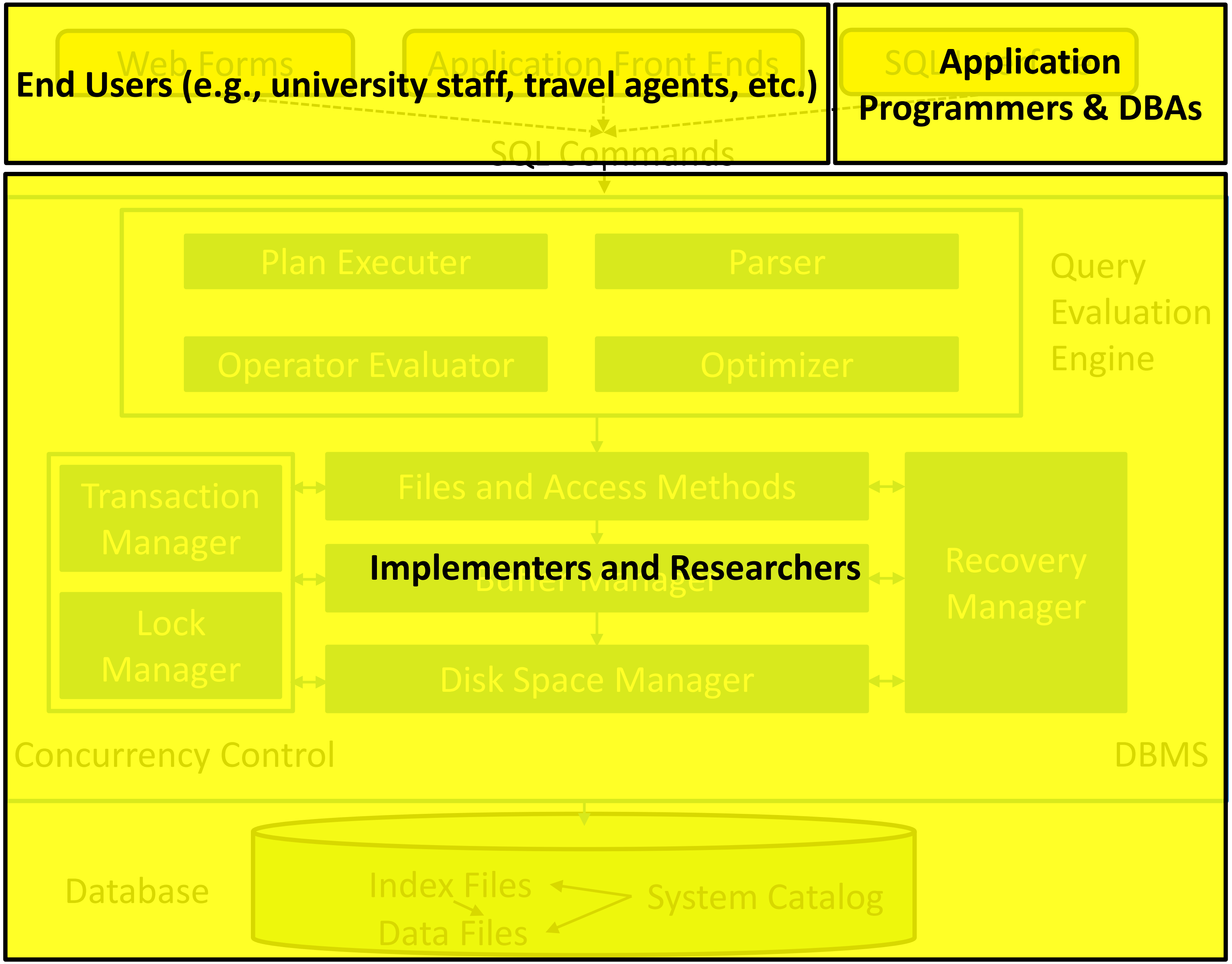
The Architecture of a Relational DBMS



People Who Work With Databases

- There are five classes of people associated with databases:
  1. End users
     - Store and use data in DBMSs
     - Usually not computer professionals
  2. Application programmers
     - Develop applications that facilitate the usage of DBMSs for end-users
     - Computer professionals who know how to leverage host languages, query languages and DBMSs altogether
  3. Database Administrators (DBAs)
     - Design the conceptual and physical schemas
     - Ensure security and authorization
     - Ensure data availability and recovery from failures
     - Perform database tuning
  4. Implementers
     - Build DBMS software for vendors like IBM and Oracle
     - Computer professionals who know how to build DBMS internals
  5. Researchers
     - Innovate new ideas which address evolving and new challenges/problems

The Architecture of a Relational DBMS



End Users (e.g., university staff, travel agents, etc.)

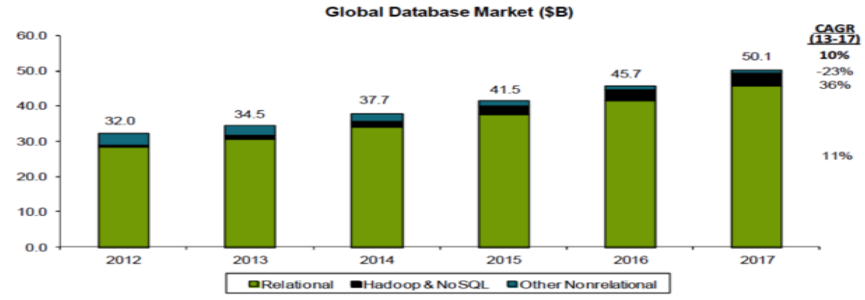
Application Programmers & DBAs

Implementers and Researchers

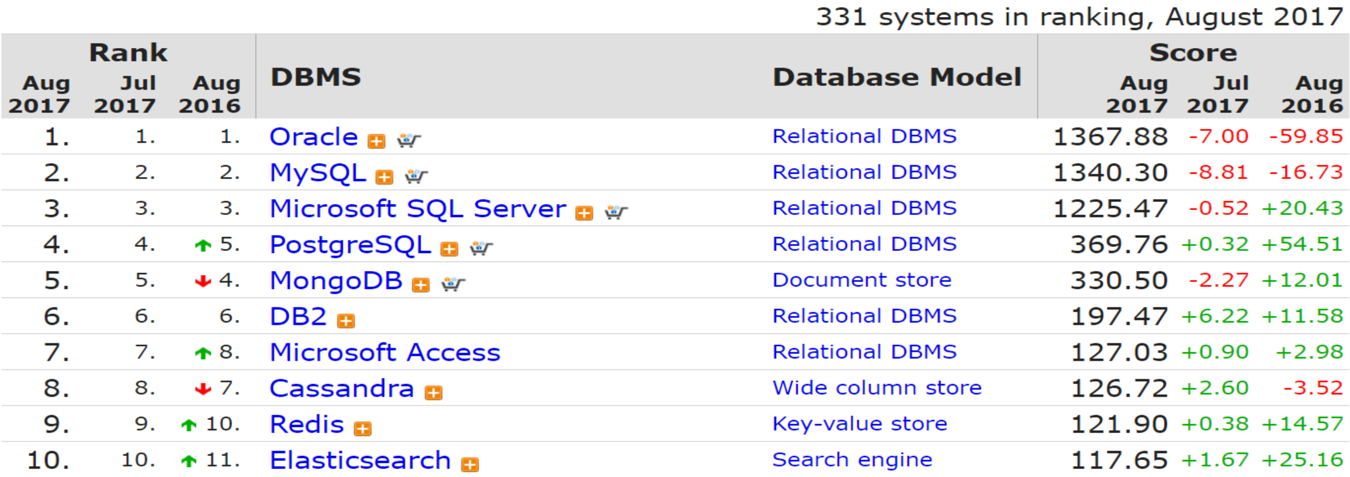
What are we still working on?

- Why Are You Taking this Course?
- Relational DBMS was invented in early 70's, and now 50+ billion mature industry
- *What are we still working on?*
  - **Database**
    - http://www.youtube.com/watch?v=Q2GMtIuaNzU
  - **Big data**
    - http://www.youtube.com/watch?v=LrNlZ7-SMPk
- Are you interested more in being
  - An **IT guru** at Goldman-Sachs or Boeing?
  - A **system developer** at Oracle or Google?
  - A **data scientist** at Facebook or Uber?
  - A **DB pro or researcher** in Microsoft research or IBM research?
  - A **professor** exploring the most exciting, and fastest growing area in CS?

In Industry



Source: IDC, Bernstein analysis



331 systems in ranking, August 2017

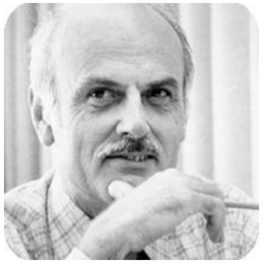
Rank			DBMS	Database Model	Score		
Aug 2017	Jul 2017	Aug 2016			Aug 2017	Jul 2017	Aug 2016
1.	1.	1.	Oracle	Relational DBMS	1367.88	-7.00	-59.85
2.	2.	2.	MySQL	Relational DBMS	1340.30	-8.81	-16.73
3.	3.	3.	Microsoft SQL Server	Relational DBMS	1225.47	-0.52	+20.43
4.	4.	↑ 5.	PostgreSQL	Relational DBMS	369.76	+0.32	+54.51
5.	5.	↓ 4.	MongoDB	Document store	330.50	-2.27	+12.01
6.	6.	6.	DB2	Relational DBMS	197.47	+6.22	+11.58
7.	7.	↑ 8.	Microsoft Access	Relational DBMS	127.03	+0.90	+2.98
8.	8.	↓ 7.	Cassandra	Wide column store	126.72	+2.60	-3.52
9.	9.	↑ 10.	Redis	Key-value store	121.90	+0.38	+14.57
10.	10.	↑ 11.	Elasticsearch	Search engine	117.65	+1.67	+25.16

In Science – Turing Awardees

The **ACM A.M. Turing Award** is an annual prize given by the Association for Computing Machinery (ACM) to an individual selected for contributions "of lasting and major technical importance to the computer field"

CHARLES BACHMAN, 1973

- **Known for his work in the early development of database management systems.**



EDGAR CODD, 1981

-Invented Relational model (**RM**), the theoretical basis for relational databases and relational database management systems.

JAMES GRAY, 1998

-For seminal contributions to database and transaction processing research

MICHAEL STONEBRAKER, 2014

-Research and products are central to many relational database systems

The Grand Challenges of Data Management

- What is the ultimately advanced DB?
  - Data of **all sorts-**-- Prevalent on the Web!
  - What have you been *searching* lately?
  - What you search is what you want?
- New challenges naturally arise
  - structured vs. unstructured data
  - querying vs. analysis vs. mining vs. learning
  - closed “base” vs. the open Web

DBMS examples

- Some of the major players:

IBM
DB2
Microsoft
SQL Server
MySQL
PostgreSQL
ORACLE

- also various “NoSQL” systems, such as Cassandra, CouchDB, MongoDB, Neo4j, . . .

Summary

- We live in a world of data

- The explosion of data is occurring along the 3Vs dimensions

- DBMSs are needed for ensuring logical and physical data independence and ACID properties, among others

- The data in a DBMS is described at three levels of abstraction

- A DBMS typically has a layered architecture

Summary

- Studying DBMSs is one of the broadest and most exciting areas in computer science!

- This course provides an in-depth treatment of DBMSs with an emphasis on how to *design*, *create*, *refine*, *use* and *build* DBMSs and real-world enterprise databases

- Various classes of people who work with databases hold responsible jobs and are well-paid!