Database Applications

- Traditional Database Applications
- Basic Definitions
- DBMS functionality
- Database System
- Example of a Database
- Database Users
- Advantages of using DBMS
- Data Models and 3 Schema Architecture
- Database Schema vs Database State
- DB System Architectures

Prepared By: Dr. Ashraf Odeh

Reference to: Fundamentals of Database Systems , Ramez Elmasri and Shamkant B. Navathe, 7th Edition.

Traditional Database Applications

Traditional Database Applications are DB applications in which accessed or retrieved data is either textual or numeric

Examples:

- Bank Systems
- Hotel Reservations
- Airline Reservations
- Computerized Library Catalog
- E-commerce websites

Basic Definitions

- **Database (DB)**: A collection of **related** data

- **Mini World (Universe of Discourse (UoD))**: Part of the world about which a database stores data.
  - Example: A database that stores data about students, instructors, departments. This database is concerned about university related information. In this case, the mini world of this database is “University Environment”.

Basic Definitions

- **Database Management System (DBMS)**: A collection of programs that enable users to create and maintain a database. Examples:
  - Oracle (Popular with most programming languages)
  - SQL Server (Most popular when working with .Net languages)
  - MySQL (Most popular when working with php or java languages)
  - MS Access (Most popular with .Net language. It is better to use it for simple applications)

- **Database System**: DBMS + DB

- **Meta-Data**: is a description of database

Typical DBMS Functionality

1) **Defining a database** (data types, structure, constraints).
   - Eg. student ID + address + major are all related together.
   - Eg. Student id should be numeric.
   - Eg. Student grade > 0 and < 100.

2) **Constructing a database**: storing actual data on a storage medium controlled by the DBMS.
   - Eg. Stores data on secondary storage medium such as hard disk.

3) **Manipulating a database**:
   - Querying (retrieving) data from DB.
   - Inserting data in DB.
   - Updating data in DB.
   - Generating reports from data.

Typical DBMS Functionality

4) **Sharing a database**: allows multiple users and programs to access the DB "**<u>concurrently</u>**". At the same time data should remain valid and consistent.

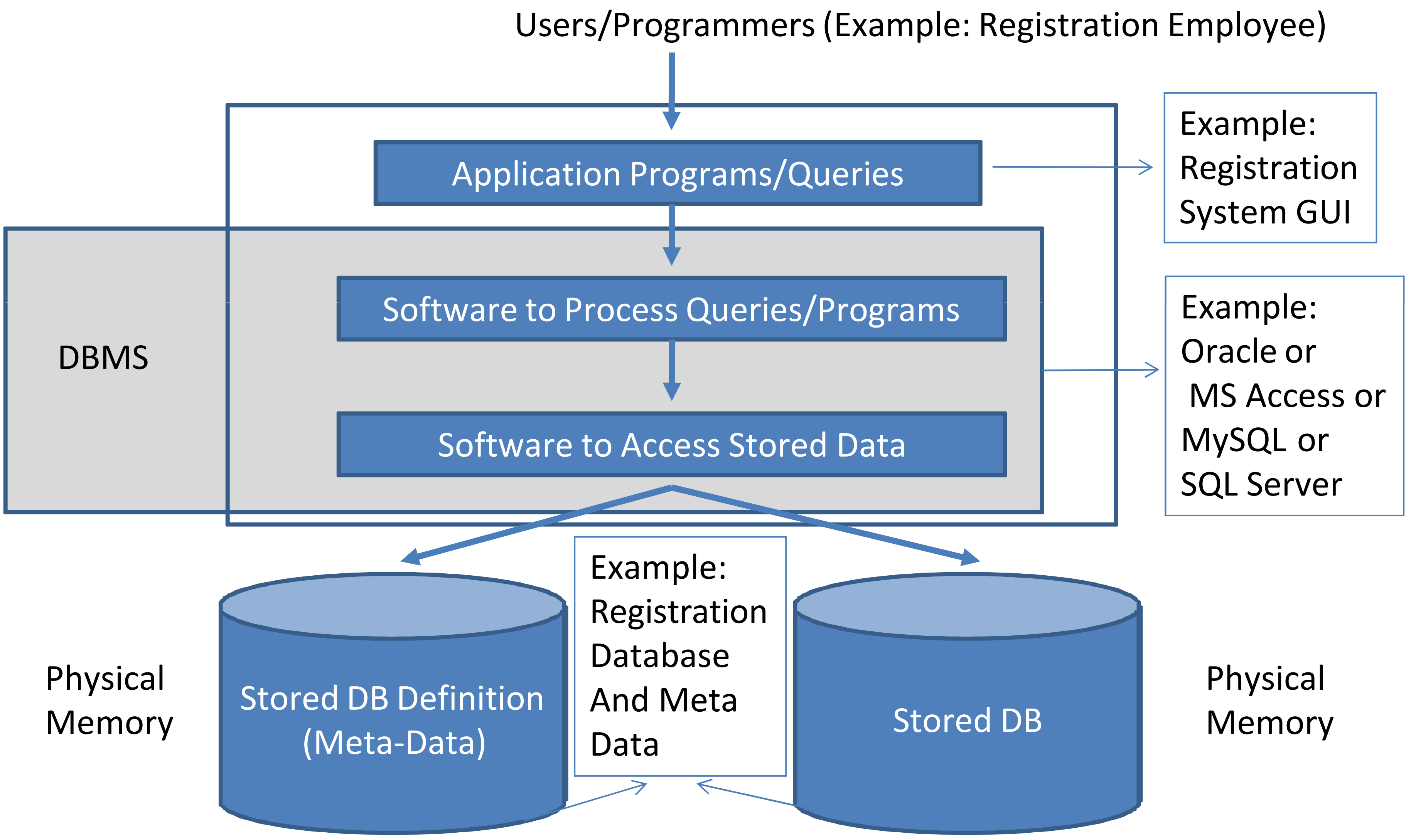
5) **Protection**: protecting the DB from:

- Software or Hardware malfunction or crashes.
- Unauthorized or malicious access.

6) Presentation and visualization of data

- Presents data for users in away they can easily understand.

A simplified DB System Environment



Users/Programmers (Example: Registration Employee)

Application Programs/Queries

Example: Registration System GUI

DBMS

Software to Process Queries/Programs

Software to Access Stored Data

Example: Oracle or MS Access or MySQL or SQL Server

Physical Memory

Stored DB Definition (Meta-Data)

Example: Registration Database And Meta Data

Stored DB

Physical Memory

Example of a DB

- Mini world of the DB: University Environment

- Mini world entities in the Example:
  - Students
  - Courses
  - Sections (of courses)
  - Academic Departments
  - Instructors

Example of a DB

- Some mini world relationships:
  - Courses **have** sections
  - Students **register for** a section
  - Instructors **teach** sections
  - Courses **have** prerequisite courses
  - Courses are **offered by** departments
  - Students **major in** departments

Note: This Conceptual data model can be translated into entity-relationship data model (will be explained later).

Example of a DB

Student	Name	Student Number	Class	Major
	Ahmad	45	1	CS
	Kamal	76	2	CIS

Course	Course Name	Course Number	Credit Hours	Department
	Data Structures	CS456	3	CS
	Database 1	CS467	3	CS
	Calculus 1	Math547	3	Math

Section	Section Identifier	Course Number	Semester	Year	Instructor
	3	CS456	First	2010	Dr. Ahmad
	1	Math547	Summer	2008	Dr. Sami

Example of a DB

Grade Report	Student Number	Section Identifier	Grade
	45	1	90
	45	3	78

Prerequisite	Course Number	Prerequisite Number
	CS456	CS783
	CS467	CS542
	Math547	Math339

Database Users

1) Database Administrator (DBA)

- Authorizes access to the DB.
- Coordinates and monitors DB usage.
- Acquires software and hardware resources as needed.

2) Database Designer

- Identifies data to be stored in the DB.
- Chooses appropriate structure to represent and store data.
- Communicates with different DB users to come up with the best design for the DB.

3) End user

- Users who access the DB to query, update, and generate reports from the DB.

Advantages of using DBMS Approach

1) Controlling Redundancy: Data is stored once in most cases and multiple times in few cases.

2) Restricting unauthorized access

3) Providing Persistent Storage

4) Providing Backup and Recovery

Problems of Redundancy

a) Repeating actions multiple times (duplication of effort).

- Eg. Registration people add new student and
- Accounting people add the same new student

b) Waste of storage space

c) Inconsistent Data: different value for the data that is supposed to be identical

- Eg. Registration people enter a student birth date as Jan-29-1999 and
- Accounting people enter the same date as Jan-19-1999

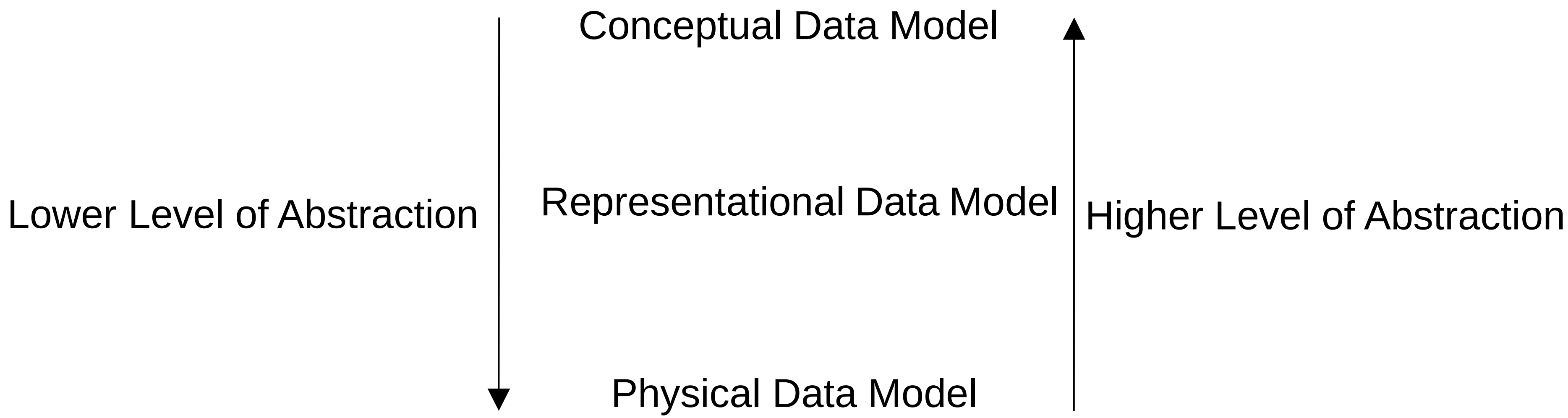
Categories of Data Models
3 Schema Architecture

Data Model	Corresponding Schema
Conceptual (High Level)	External Schema
Representational (Implementation)	Conceptual Schema
Physical (Low Level)	Internal Schema

Data Models

- **Conceptual Model**: eg. Registration database consists of information related students, instructors, courses, … etc
- **Representational Model**: eg. implementing registration database as tables using DBMS such as Oracle
- **Physical Model**: eg. Defining an index on an attribute to make search more efficient

Abstraction in Data Models



Conceptual Data Model

Representational Data Model

Physical Data Model

Lower Level of Abstraction

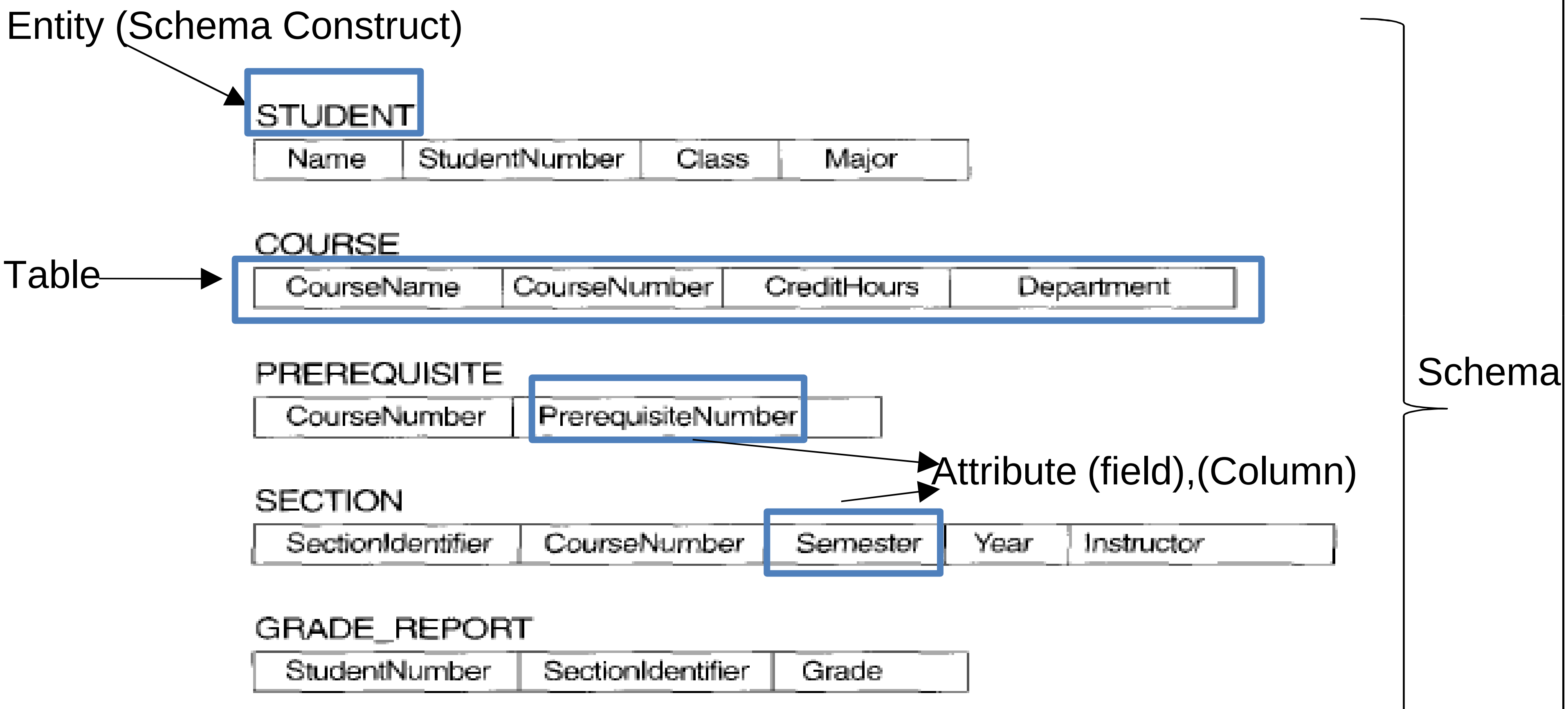
Higher Level of Abstraction

If we are hiding more information, level of data abstraction increases

Schema

- DB Schema:
  - Description of the DB.
  - It is usually specified during **design stage** and is **not expected to change** frequently.

Schema



Entity (Schema Construct)

Table

Attribute (field),(Column)

Schema

STUDENT

Name	StudentNumber	Class	Major

COURSE

CourseName	CourseNumber	CreditHours	Department

PREREQUISITE

CourseNumber	PrerequisiteNumber

SECTION

SectionIdentifier	CourseNumber	Semester	Year	Instructor

GRADE_REPORT

StudentNumber	SectionIdentifier	Grade

FIGURE 2.1 Schema diagram for the database in Figure 1.2.

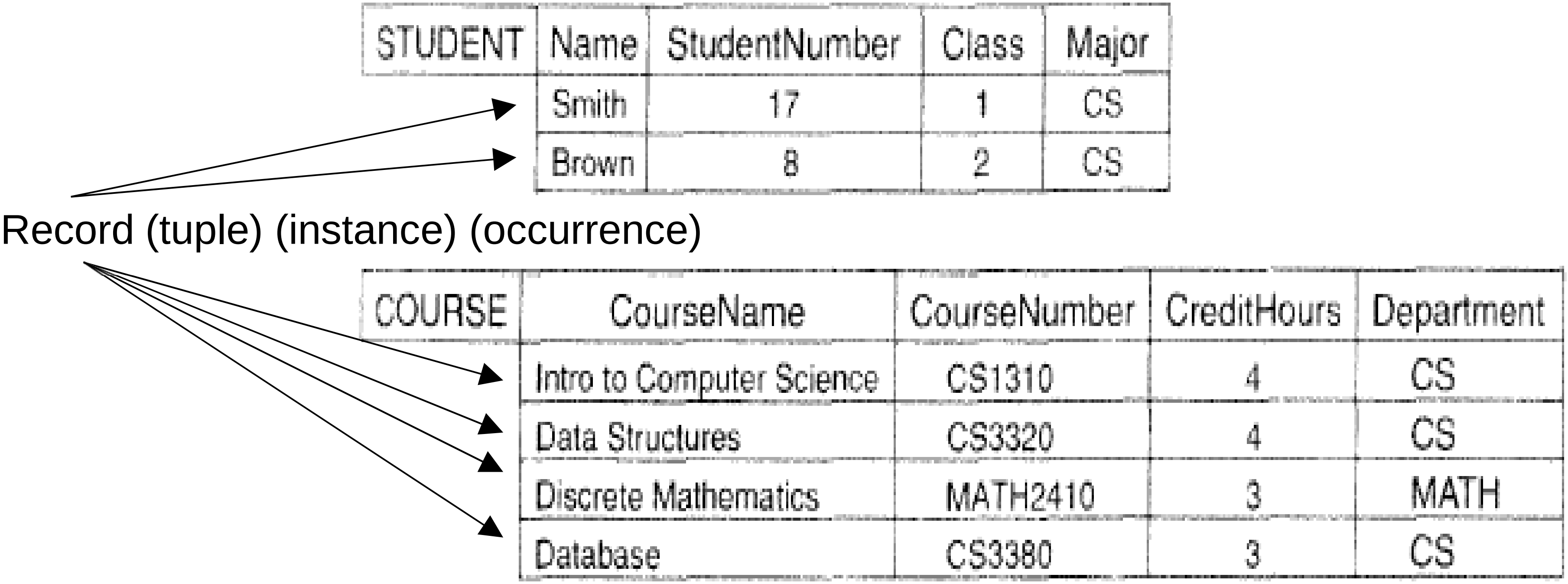
Database State

- Database State (Snapshot): It is the data in the DB at a particular moment in time.

- Database state changes frequently while DB schema does not change frequently.

- Database state contains instances (occurrences) of data.

Database State



STUDENT	Name	StudentNumber	Class	Major
	Smith	17	1	CS
	Brown	8	2	CS

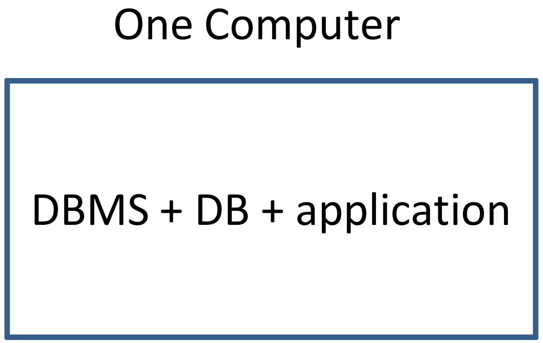
Record (tuple) (instance) (occurrence)

COURSE	CourseName	CourseNumber	CreditHours	Department
	Intro to Computer Science	CS1310	4	CS
	Data Structures	CS3320	4	CS
	Discrete Mathematics	MATH2410	3	MATH
	Database	CS3380	3	CS

- Every time we perform an update operation, we get a new DB state.

DBMS Architectures (Centralized Architecture)

- The DBMS, DB, and applications to access DB all exist on one machine.

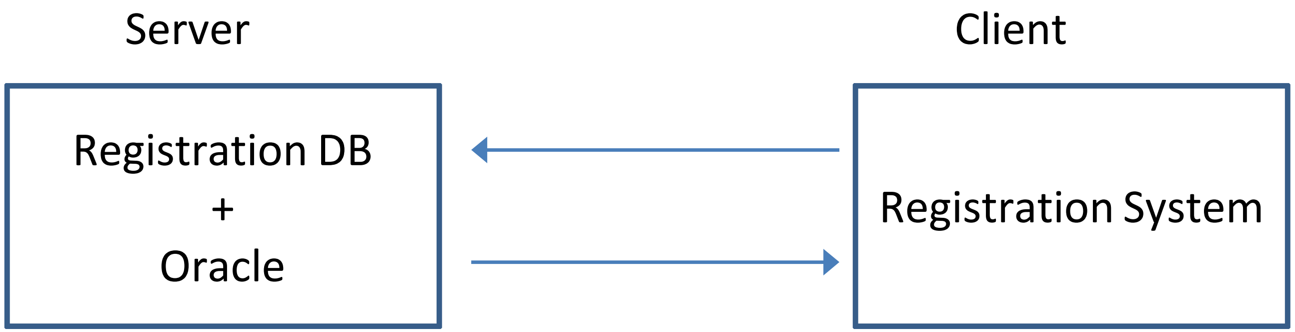


One Computer

DBMS + DB + application

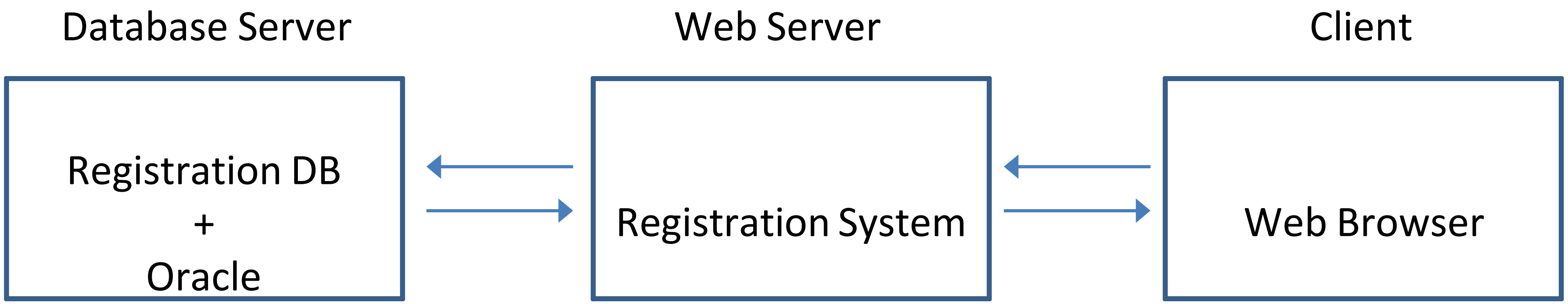
DBMS Architectures (2-tier Architecture)

- The DBMS and DB exist on Server.
- The application exists on Client
- This is very popular when the application is a desktop application (can be developed using .Net, java, … etc)



DBMS Architectures (3-tier Architecture)

- The DBMS and DB exist on Database Server.
- The application exists on Web Server (Application Server)
- The Client uses a browser (eg. Google Chrome) to access system
- This is very popular with Web Applications (It can be designed using ASP.Net, php, JSP+Servlets, Ruby on Rails, … etc)



Why tier architectures are good design to follow

- Separation between user interface, business logic rules, and data (DB).

- The ability to change any tier without having to change the other tiers.

- Enhancing security because the client is not directly connected to database.