

Database Management Systems: An Architectural View

Lecture 1



Outline

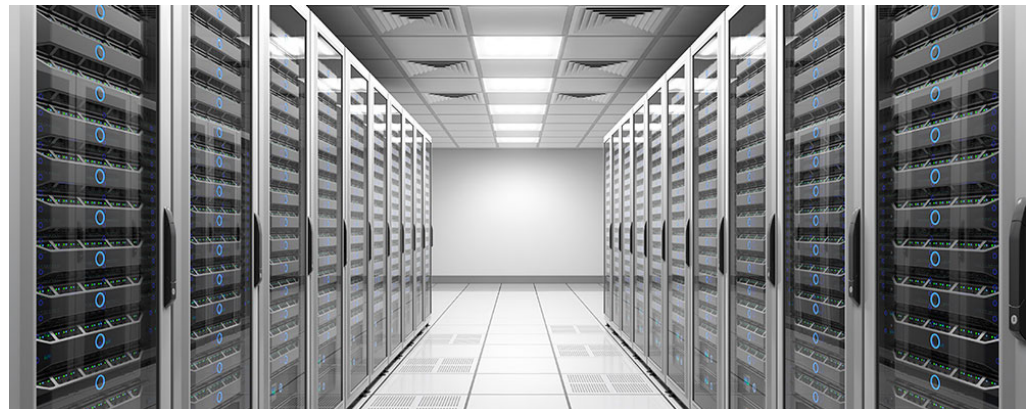
1. What is a Database? A DBMS?
2. Why use a DBMS?
3. Databases in Context
4. Design and Implementation Process



What is a Database?



What is a Database?



What is a Database?

A collection of related data, most often...

- reflects some aspect of the real world
- logically coherent with inherent meaning
- designed, built, and populated with data for a specific purpose
 - intended group of users
 - some preconceived applications with which these users are interested
 - application requirements in terms of performance, security, redundancy, concurrency, etc.



Database Management System

DBMS

A collection of programs that enables users to create and maintain a database

- Supports specifying the data types, structures, and constraints of the data
- Stores the data on some medium under control of the DBMS
- Supports querying and updating the database
- Protects data against malfunction and unauthorized access



Why use a DBMS?

Common tradeoff in CS:

A. Code from scratch

- Pros: you know your problem best (so fast, customized)
- Cons: slow, labor intensive, need to add/change features?

B. Find a library/tool that solves [part of] your problem

- Pros: fast via bootstrapping, better designed?
- Cons: understand the tool, may not be efficient, support?

DBMSs adopt some set of limiting assumptions in order to efficiently support a useful feature set over a wide class of possible databases




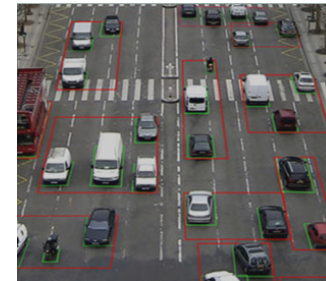
Example: Student Records

- Given a school with MANY students (NEU: ~25k, UM: ~45k), each with some data (name, ID, DOB, classes)
- Write a program that can efficiently...
 - Retrieve a random student
 - Retrieve the first/last student, according to...
 - Last name
 - DOB
 - Retrieve a student by...
 - ID
 - Name (with *'s)
 - Retrieve a class roster (all students in class X)
 - Handles adding/removing/editing students/classes
 - Handles multiple simultaneous reads/writes
 - Provides differing access rights
 - Handles OS faults/power outages
 - ...



Many Kinds of DBMSs (1)

- Graph databases  Neo4j
 - Create nodes, edges, labels
 - Query about relationships and paths
 - Find your friends
 - Find someone that can help you learn databases
- Spatial databases
 - Objects in 2D/3D
 - Query locations, relations
 - Collision detection



Many Kinds of DBMSs (2)

- Document stores
 - Create dynamic documents
 - Query about contents
 - Find by author, title, content, etc. patterns



- Key-Value stores
 - Associative array
 - Scalable, fault-tolerant
 - Query



Relational DBMS

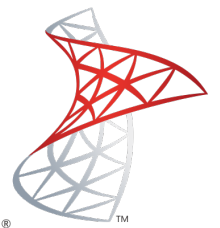
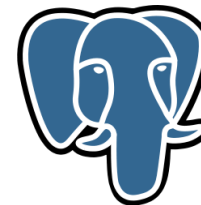
We focus on **relational** databases

Based on the relational *data model*

- Researched ~45 years, widely used
 - Free/paid implementations for personal use, embedded systems, small/large enterprise



FileMaker

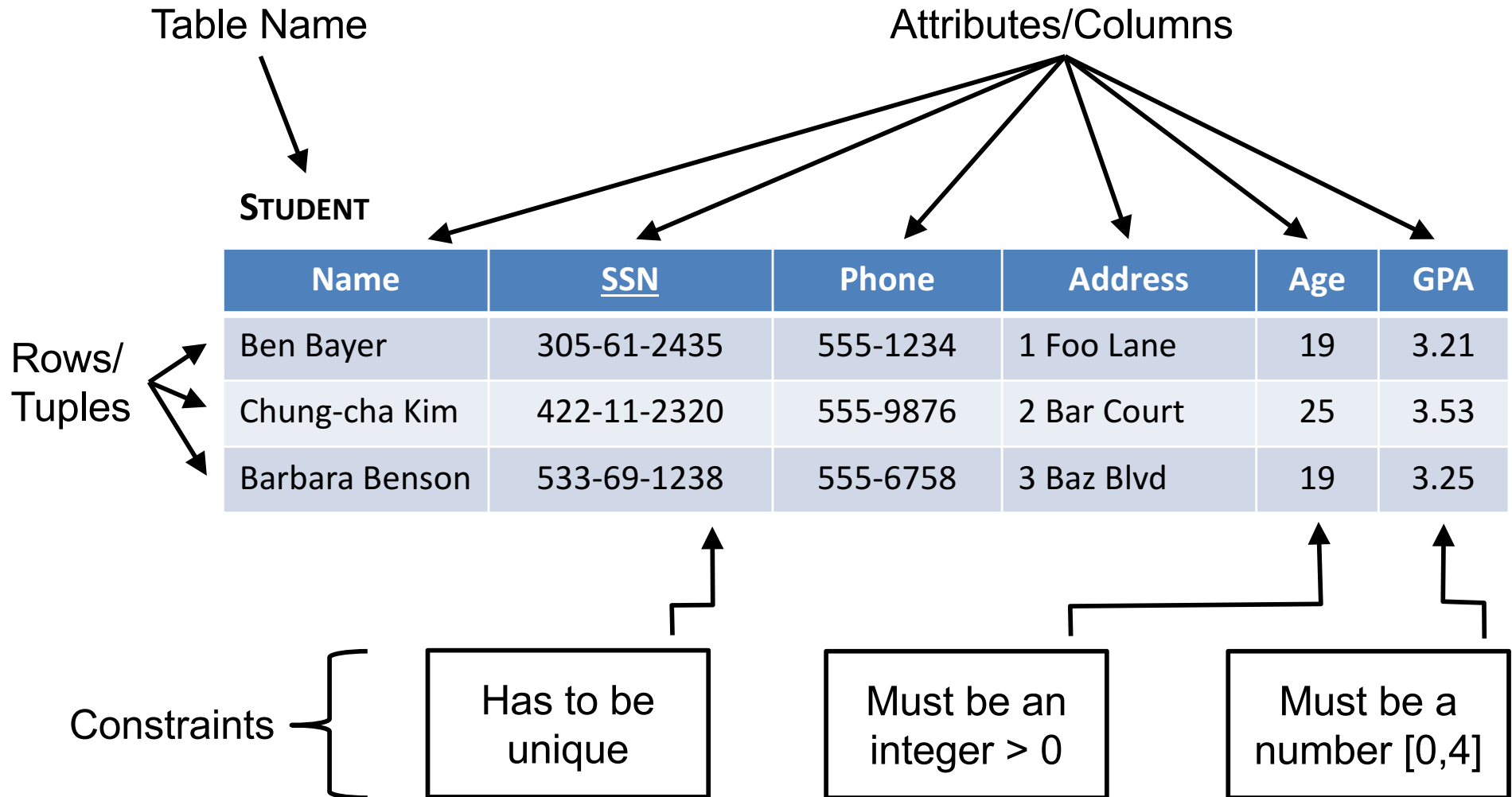


Microsoft®
SQL Server®



Relational Databases (1)

Table or “Relation”



Relational Databases (2)

More Tables!

STUDENT

Name	<u>SSN</u>	Phone	Address	Age	GPA
Ben Bayer	305-61-2435	555-1234	1 Foo Lane	19	3.21
Chung-cha Kim	422-11-2320	555-9876	2 Bar Court	25	3.53
Barbara Benson	533-69-1238	555-6758	3 Baz Blvd	19	3.25

CLASS

<u>SSN</u>	<u>Class</u>
305-61-2435	COMP355
422-11-2320	COMP355
533-69-1238	MATH650
305-61-2435	MATH650
422-11-2320	BIOL110

DORM

<u>SSN</u>	Dorm
305-61-2435	555 Huntington
422-11-2320	Baker
533-69-1238	555 Huntington

Values in one table can be forced to come from another (“Referential Integrity”)



Relational Databases (3)

Queries!

STUDENT

Name	<u>SSN</u>	Phone	Address	Age	GPA	
Ben Bayer	305-61-2435	555-1234	1 Foo Lane	19	3.21	Result
Chung-cha Kim	422-11-2320	555-9876	2 Bar Court	25	3.53	3.23
Barbara Benson	533-69-1238	555-6758	3 Baz Blvd	19	3.25	

What is the average GPA of students in MATH650?

- Find all SSN in table Class where Class=MATH650
- Find all GPA in table Student where SSN=#1
- Average GPA in #2

DORM

<u>SSN</u>	Dorm
305-61-2435	555 Huntington
422-11-2320	Baker
533-69-1238	555 Huntington

CLASS

<u>SSN</u>	<u>Class</u>
305-61-2435	COMP355
422-11-2320	COMP355
533-69-1238	MATH650
305-61-2435	MATH650
422-11-2320	BIOL110



Relational Databases (3)

Queries!

STUDENT

Name	<u>SSN</u>	Phone	Address	Age	GPA
Ben Bayer	305-61-2435	555-1234	1 Foo Lane	19	3.21
Chung-cha Kim	422-11-2320	555-9876	2 Bar Court	25	3.53
Barbara Benson	533-69-1238	555-6758	3 Baz Blvd	19	3.25

Result

3.23

CLASS

<u>SSN</u>	<u>Class</u>
305-61-2435	COMP355
422-11-2320	COMP355
533-69-1238	MATH650
305-61-2435	MATH650
422-11-2320	BIOL110

DORM

<u>SSN</u>	Dorm
305-61-2435	555 Huntington
422-11-2320	Baker
533-69-1238	555 Huntington

```

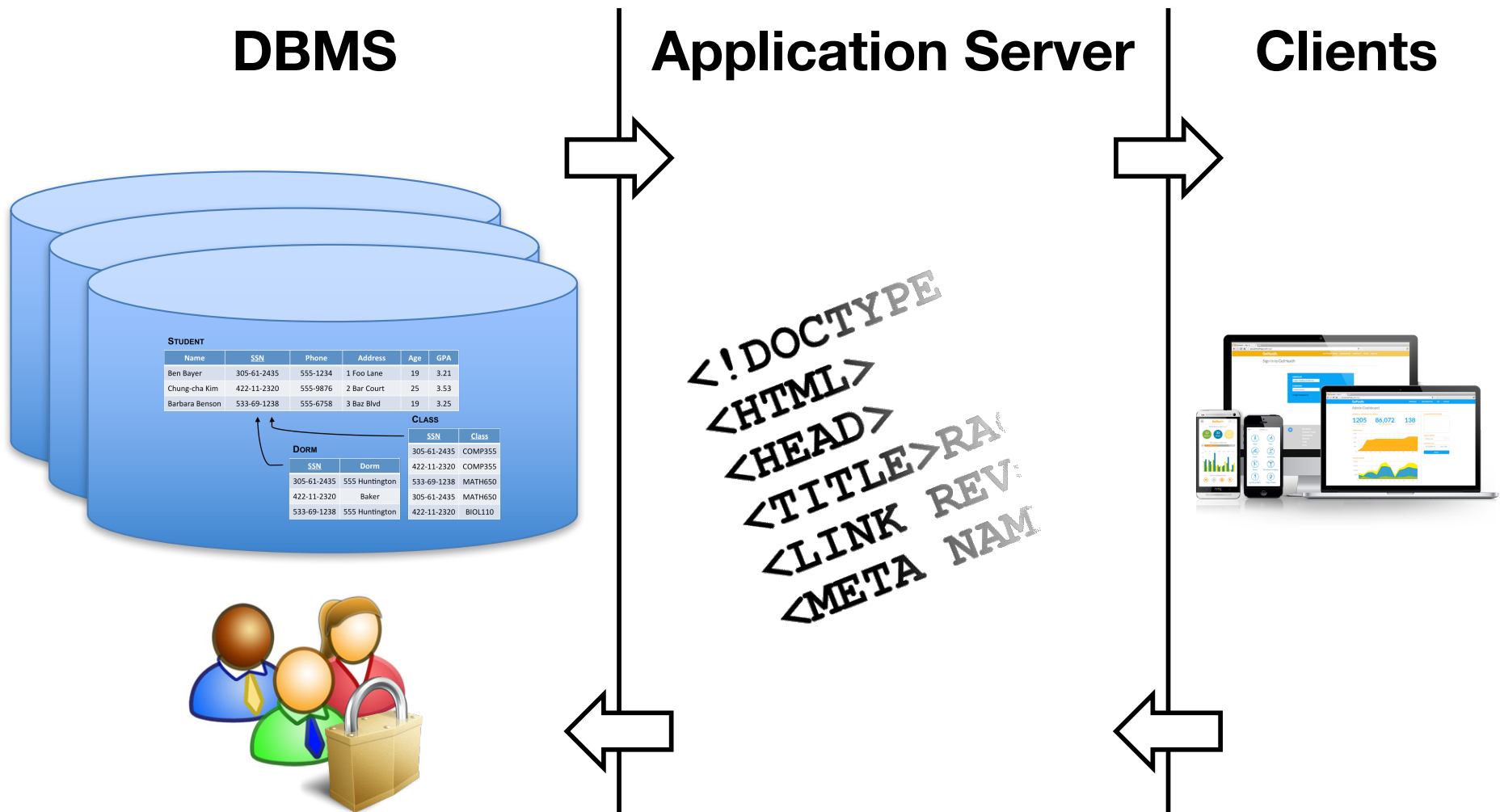
SELECT AVG(STUDENT.GPA)
FROM
    STUDENT INNER JOIN CLASS
    ON STUDENT.SSN=CLASS.SSN
WHERE CLASS.Class= 'MATH650';

```



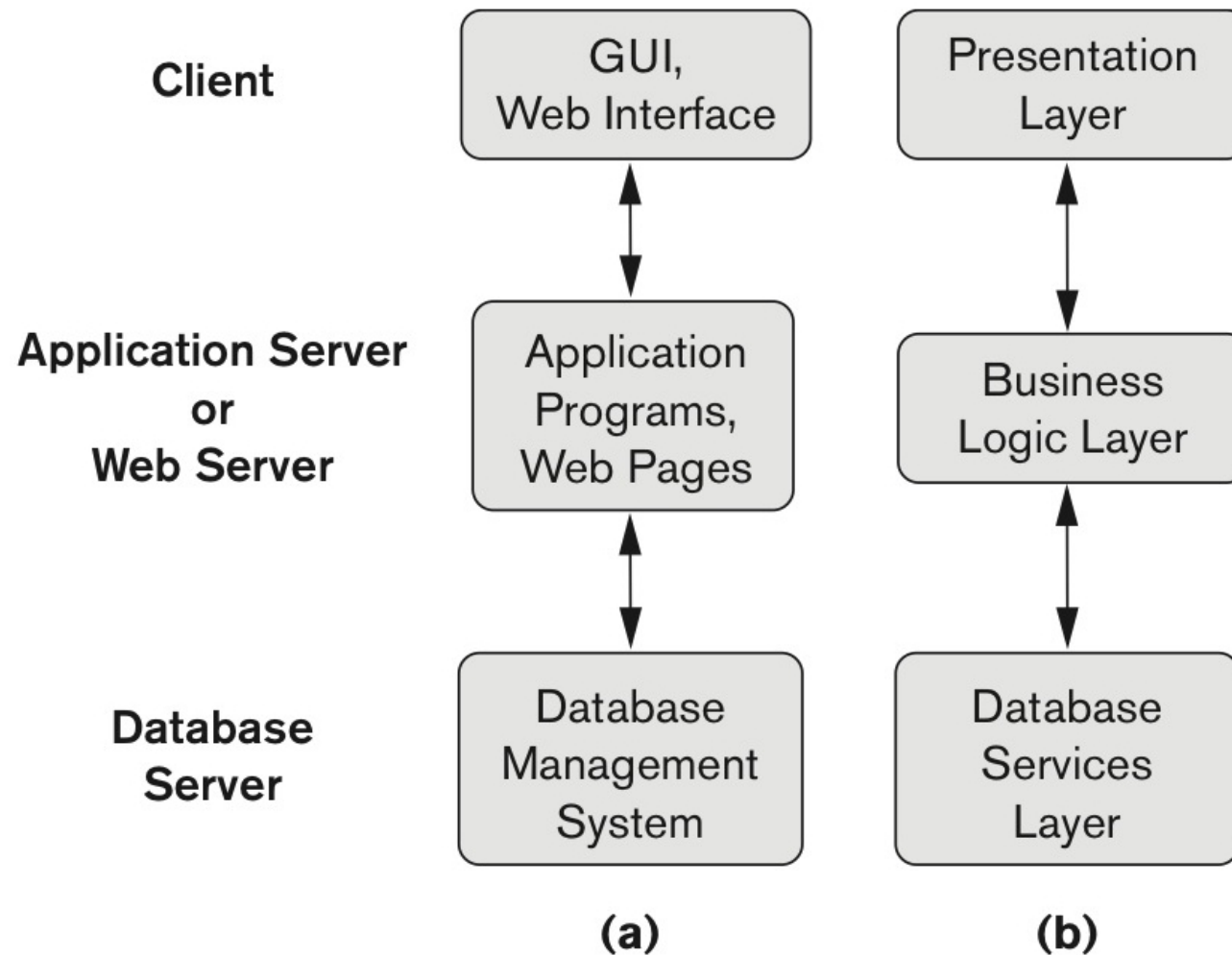
Relational Databases (4)

Users!



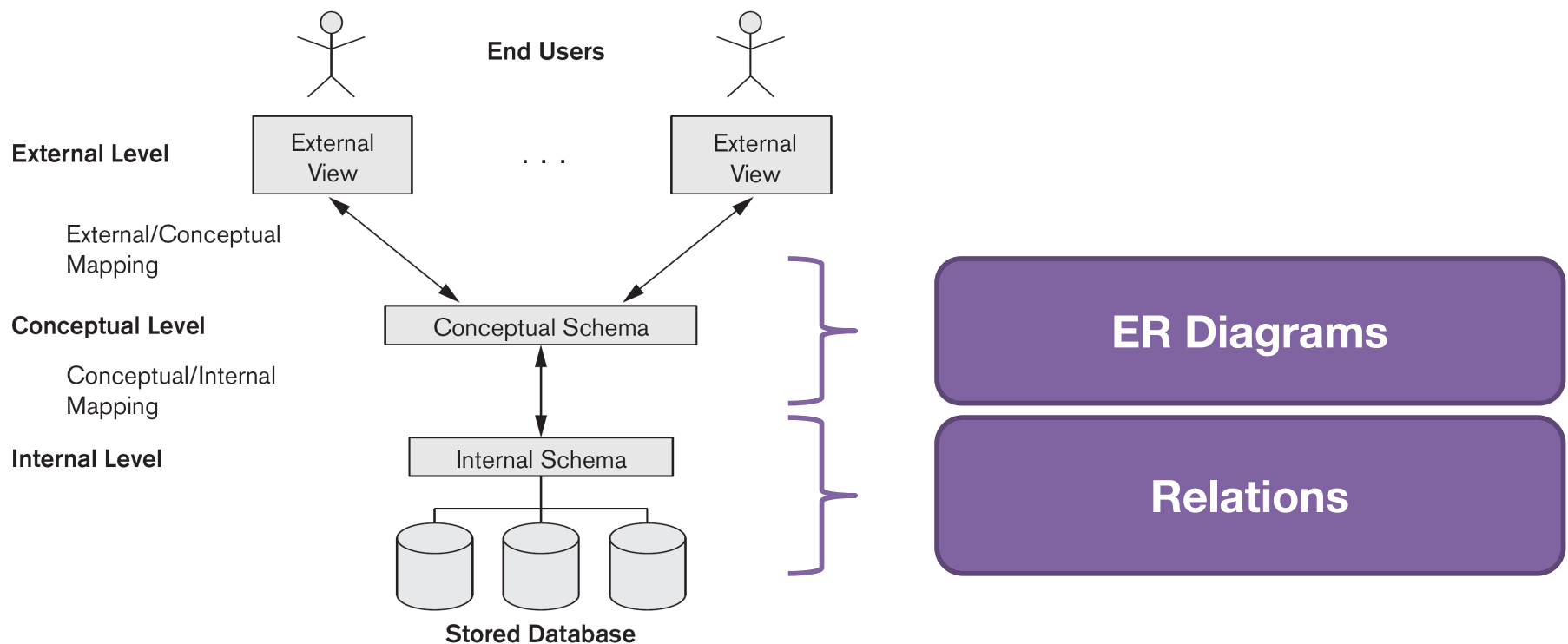
Databases in Context

Three-Tier Architecture



Relational DBMS Features (1)

- Data *independence* via **data models**
 - Conceptual representation independent of underlying storage or operation implementation



Relational DBMS Features (2)

- Operation abstraction via...
 - Declarative languages
 - Structured Query Language (SQL)
 - Data... definition, manipulation, query
 - Programmatic APIs
 - Function libraries (focus), embedded languages, stored procedures, etc.



Relational DBMS Features (3)

- Reliable concurrent transactions
 - (A)tomicity: “all or nothing”
 - (C)onsistency: valid \rightarrow valid’
 - (I)solation: parallel execution, serial result
 - (D)urability: once it is written, it is so
- High performance
 - Buffering, caching, locking (like a mini OS)
 - Query optimization, redundant data structures (e.g. indexes, materialized views)



Relational DBMS Features (4)

- Authentication and authorization
 - Discussed in context of other security concerns/techniques
- Backup and recovery
 - Logging, replication, migration



Why NOT to use a DBMS

Your application...

- involves a single user
- has simple/well-defined data/operations
 - DBMS may be overkill

However, DBMS techniques may be useful

- We will discuss useful and scalable indexing structures and processes



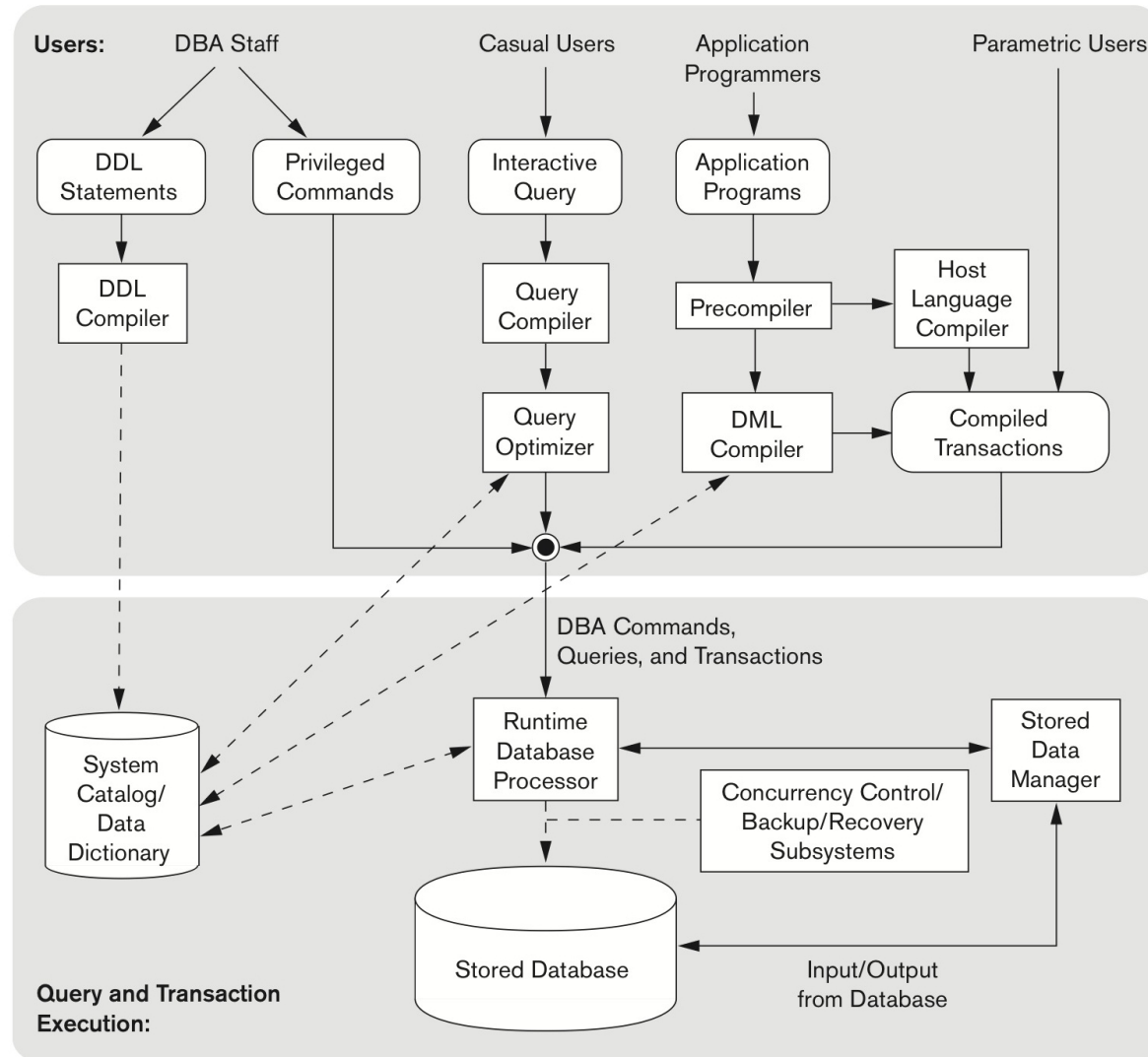
Databases in Context

People

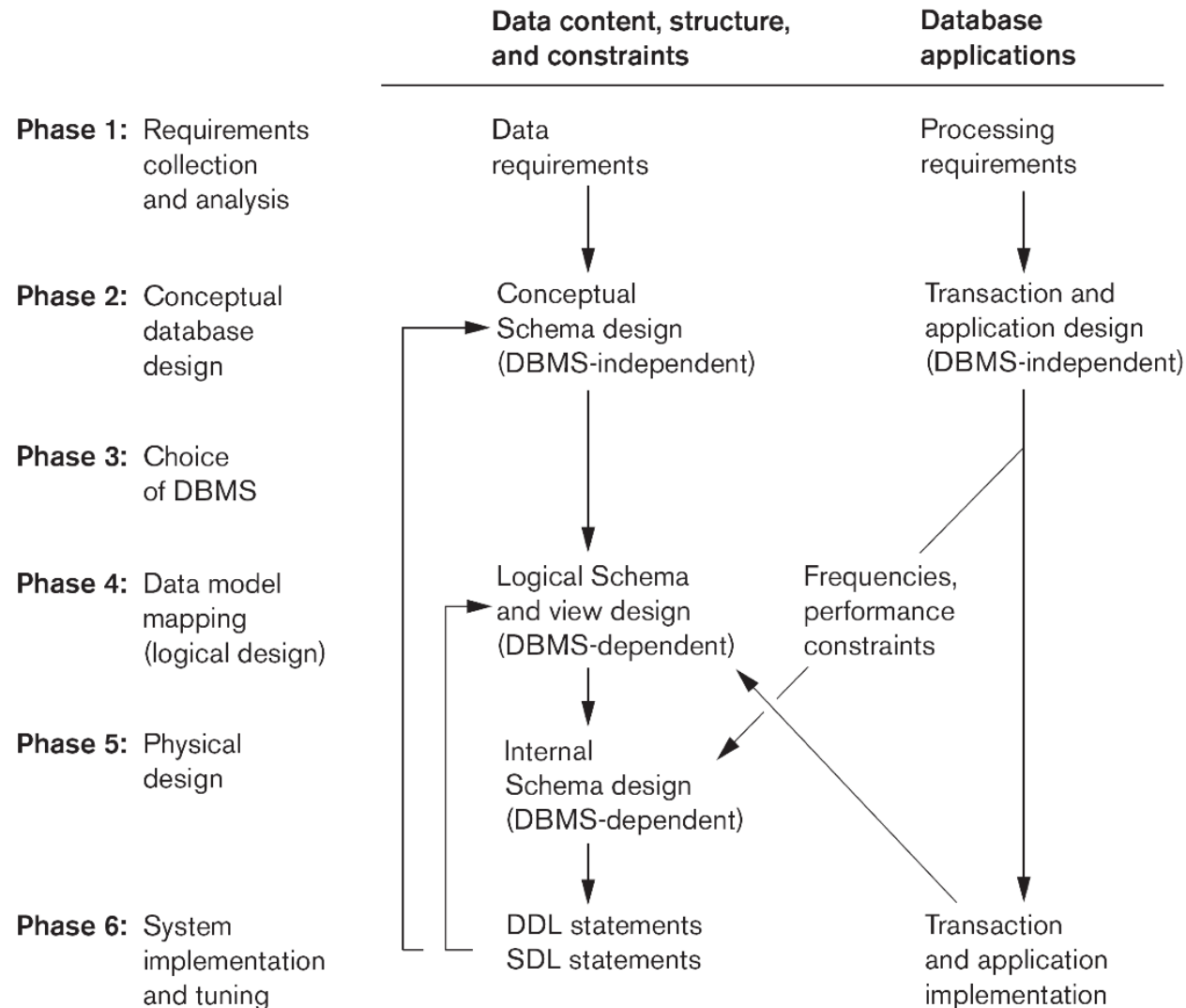
- 1. Database designers**
- 2. System analysts & application programmers**
- 3. Database administrators**
- 4. End users**
5. Back-end
 - a. DBMS designer/implementer
 - b. Tool developers
 - c. SysAdmins



Relational DBMS



Database Design and Implementation Process



Requirements Collection & Analysis

- Data/Constraints

“The company is organized into departments. Each department has a unique name, number, and a particular employee who manages the department. We keep track...”



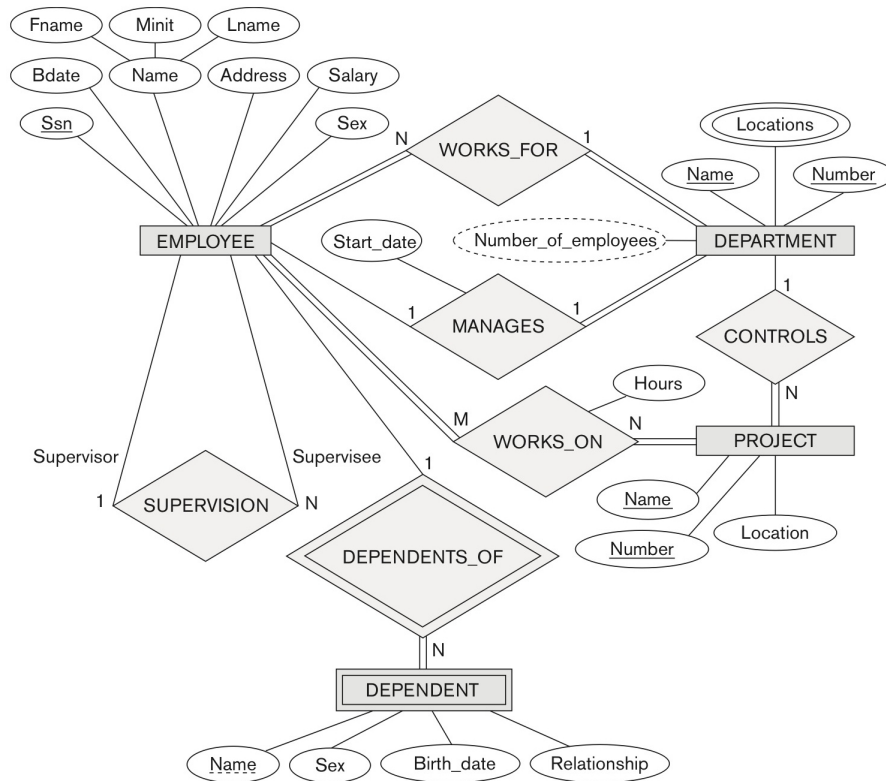
- Functional Needs

- Operations/queries/reports
 - Frequency
- Performance, security, etc.



Conceptual Design

Data



Application

- Software
 - UML
 - Form design
- Database
 - Transaction design
 - Report design



Logical Design

Data

EMPLOYEE

Fname	Minit	Lname	<u>Ssn</u>	Bdate	Address	Sex	Salary	Super_ssn	Dno
-------	-------	-------	------------	-------	---------	-----	--------	-----------	-----

DEPARTMENT

Dname	<u>Dnumber</u>	Mgr_ssn	Mgr_start_date
-------	----------------	---------	----------------

DEPT_LOCATIONS

<u>Dnumber</u>	<u>Dlocation</u>
----------------	------------------

PROJECT

Pname	<u>Pnumber</u>	Plocation	Dnum
-------	----------------	-----------	------

WORKS_ON

<u>Essn</u>	<u>Pno</u>	Hours
-------------	------------	-------

DEPENDENT

<u>Essn</u>	<u>Dependent_name</u>	Sex	Bdate	Relationship
-------------	-----------------------	-----	-------	--------------

Application

- Supporting code (that does not depend upon database)
 - Possibly using techniques from databases (e.g. indexing)

- Normalization



Physical Design

Data

- Index, materialized view selection and analysis

Application

- Implementing operations as queries
- Implementing constraints as keys, triggers, views
- Implementing multi-user security as grants



Implementation and Tuning

Data

- DDL statements
- De-normalization, updating indexes/materialized views

Application

- Query integration
- Profiling queries/operations
- Security, concurrency, performance, etc. analysis



Summary

- A **database** is a collection of related data that reflects some aspect of the real world; is logically coherent with inherent meaning; and is designed, built, and populated with data for a specific purpose
- A **database management system** (DBMS) is a collection of programs that enables users to create and maintain a database
- There are many types – we will focus on **relational** databases (RDBMS)
- The typical database design process is an iterative process of requirements collection/analysis, conceptual design, logical design, physical design, and system implementation/tuning

