Chapter 1: Introduction

Database System Concepts, 6th ed.

Chapter 1: Introduction

- 1.1 Database-System Applications
- 1.2 Purpose of Database Systems
- 1.3 View of Data
- 1.4 Database Languages
- 1.5 Relational Databases
- 1.6 Database Design
- 1.7 Data Storage and Querying
- 1.8 Transaction Management
- 1.9 Database Architecture
- 1.10 Data Mining and Analysis
- 1.11 Specialty Databases
- 1.12 Database Users and Administrators
- 1.13 History of Database Systems

1.1. Database System Applications

Database Management System (DBMS)

- DBMS contains information about a particular enterprise
  - Collection of interrelated data
  - Set of programs to access the data
  - An environment that is both *convenient* and *efficient* to use
- Database Applications:
  - Banking: transactions
  - Airlines: reservations, schedules
  - Universities: registration, grades
  - Sales: customers, products, purchases
  - Online retailers: order tracking, customized recommendations
  - Manufacturing: production, inventory, orders, supply chain
  - Human resources: employee records, salaries, tax deductions
- Databases can be very large.
- Databases touch all aspects of our lives

University Database Example

- Application program examples
  - Add new students, instructors, and courses
  - Register students for courses, and generate class rosters
  - Assign grades to students, compute grade point averages (GPA) and generate transcripts
  - Deposit money into a bank account
  - Assign seats in a flight of an airplane
  - Buy and sell stocks in an home trading system
  - Run an internet portal system based on keyword queries

- In the early days, database applications were built directly on top of OS file systems

1.2 Purpose of Database Systems
Limits of File Systems to Store Data [1/3]

- Data redundancy and inconsistency
  - Multiple file formats, duplication of information in different files
  - Ex. Customer file has records (name, phone_no, address)

    Customer_Auto file has records (name, job, auto_info, address)
- Difficulty in accessing and manipulating data
  - Need to write a new program to carry out each new task
  - Application programming with data scattered in multiple files and formats is difficult
- Integrity problems
  - Semantics (integrity) of data
  - 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

Limits of File Systems to Store Data [2/3]

Files supported by OS

Branch File having Variable-length records

	Branch name	account		account		account	
0	Perryridge	A-102	400	A-201	900	A-218	700
1	Round Hill	A-305	350	$\perp$	$\perp$	$\perp$	$\perp$
2	Mianus	A-215	700	$\perp$	$\perp$	$\perp$	$\perp$
3	Downtown	A-101	500	A-110	600	$\perp$	$\perp$
4	Redwood	A-222	700	$\perp$	$\perp$	$\perp$	$\perp$
5	Brighton	A-217	750	$\perp$	$\perp$	$\perp$	$\perp$

Customer File having fixed-length records

customer	Account number
Hayes	A-102
Johnson	A-101
Johnson	A-201
Jones	A-217
Lindsay	A-222
Smith	A-215
Turner	A-305

Limits of File Systems to Store Data [3/3]

- 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
    - Example: 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

1.3 View of Data
Levels of Abstraction

- **Physical level:** describes how a record (e.g., customer) is stored.
  - File level
- **Logical level:** describes data stored in database, and the relationships among the data.
  - Relation level (= schema level)

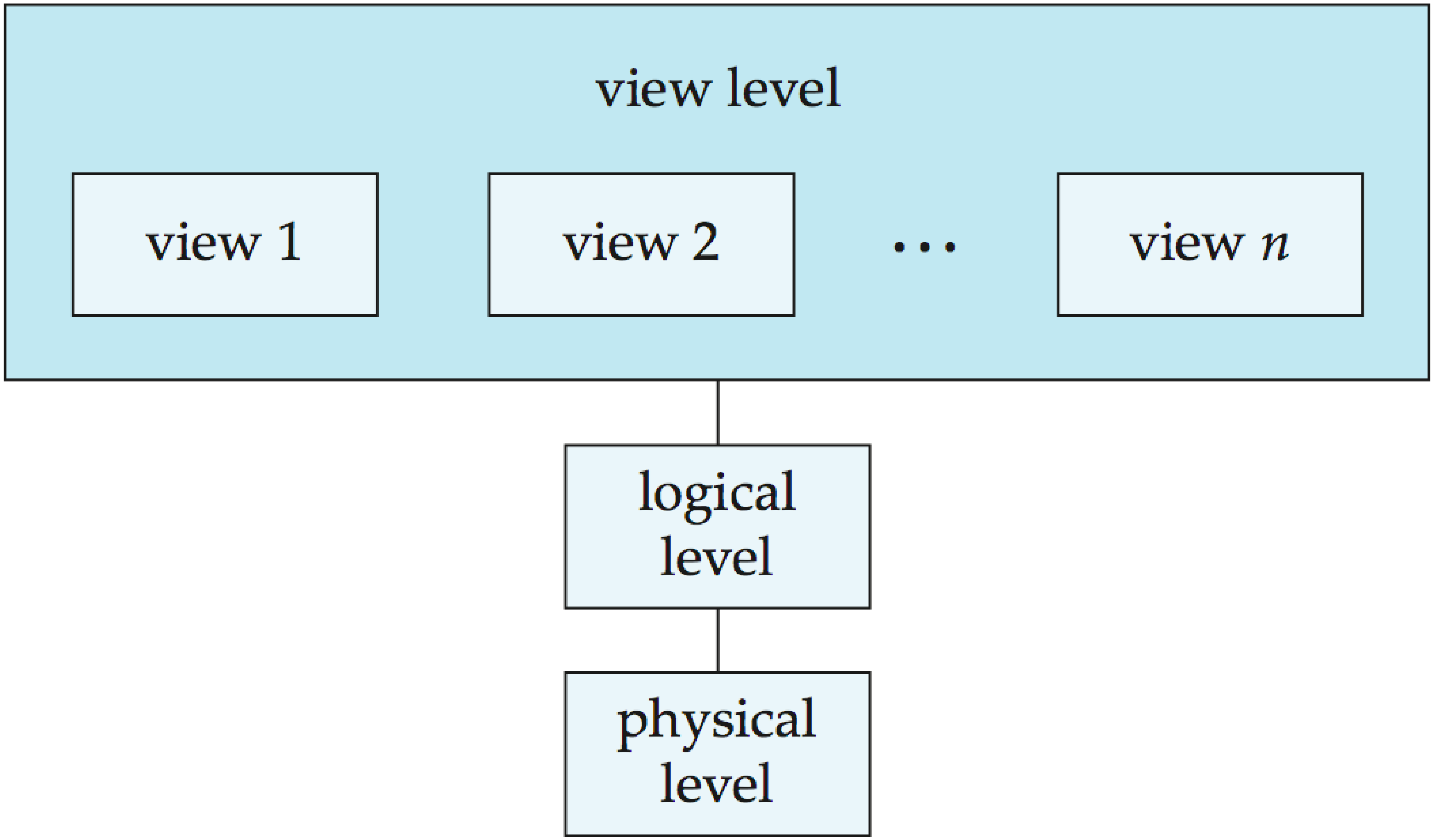
  type *instructor* = **record**

  ID : string;
  name : string;
  dept_name : string;
  salary : integer;

  end;
- **View level:** application programs hide details of data types.
  - Application level
  - Views can also hide information (such as an employee's salary) for security purposes.

View of Data

An architecture for a database system



Instances and Schemas

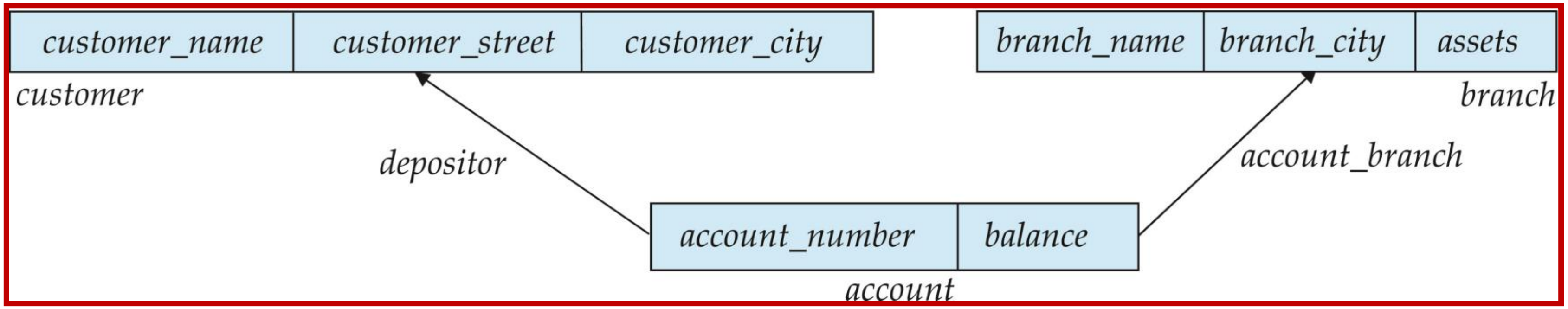
- Schema = Similar to types and variables in programming languages
- **Schema** – the logical structure of the database
  - Example: The database consists of information about a set of customers and accounts and the relationship between them
  - Analogous to type information of a variable in a program
  - **Physical schema**: database design at the physical level
  - **Logical schema**: database design at the logical level
- **Instance** – the actual content of the database at a particular point in time
  - Analogous to the value of a variable
- **Physical Data Independence** – the ability to modify the physical schema without changing the logical schema
  - Applications depend on the logical schema
  - In general, the interfaces between the various levels and components should be well defined so that changes in some parts do not seriously influence others.

Data Models

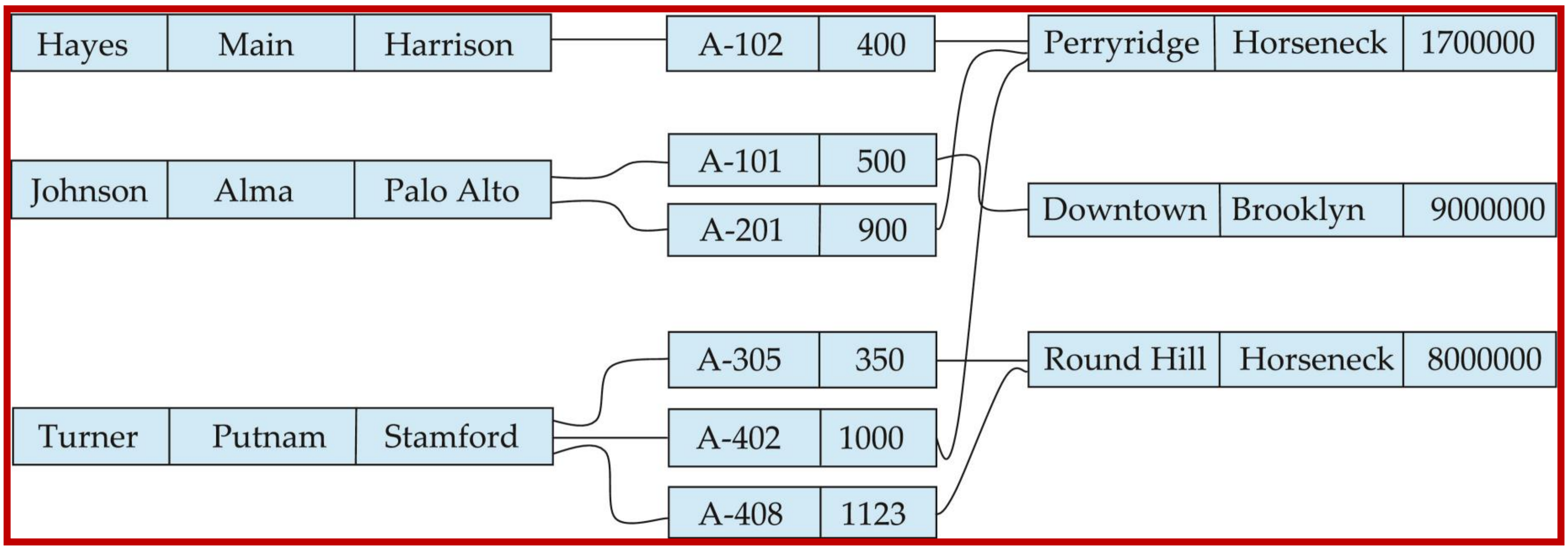
- Data Model: A collection of tools for describing
  - Data
  - Data relationships
  - Data semantics
  - Data constraints
- Relational model
- Entity-Relationship data model (mainly for database design)
- Object-based data models (Object-oriented and Object-relational)
- Semistructured data model (XML)
- Other older models:
  - Network model
  - Hierarchical model

Network Data Model (before 1980)

schema



data



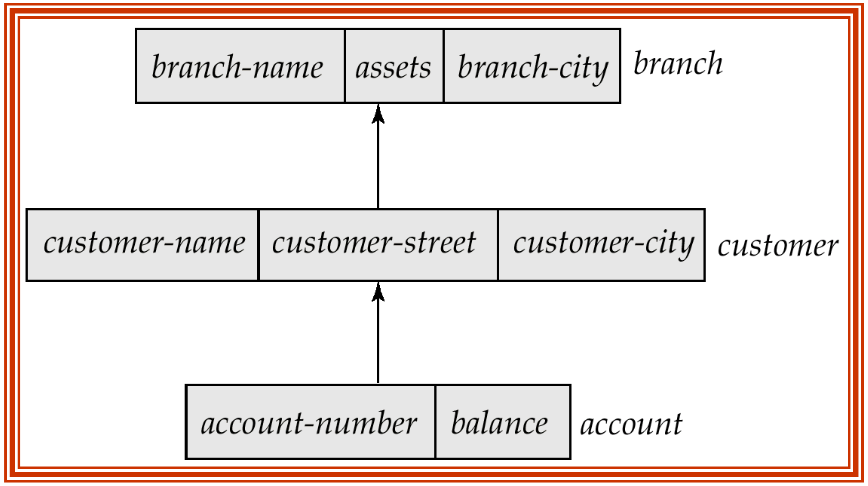
Example Query in Network Data Model

- Print the total number of accounts in the Perryridge branch with a balance greater than $10,000.

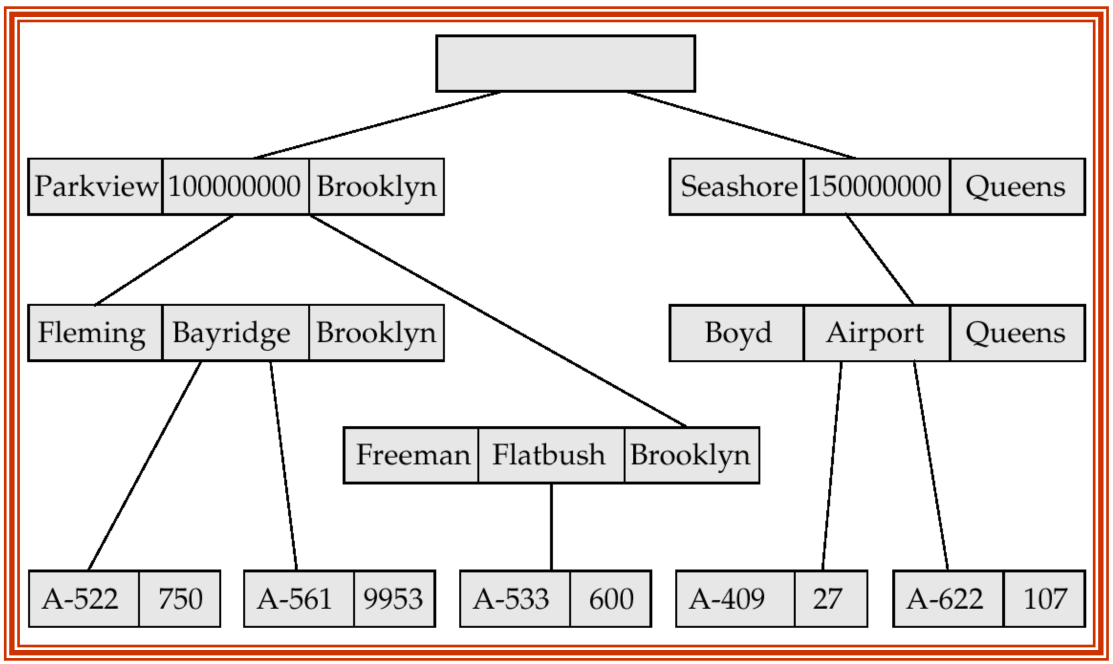
  count := 0;
  branch.branch-name := “Perryridge”;
  find any *branch* **using** *branch-name;*
  find first *account* **within** *account-branch;*
  while DB-status = 0 **do**
      begin
          get *account*
          if *account.balance* > 10000 **then** *count* := *count* + 1;
          find next *account* **within** *account-branch;*
      end
  print *(count);*

Hierarchical Data Model (before 1980)

schema



data



Example Query in Hierarchical Data Model

- Print the account number of all the accounts that have a balance greater than $500

```
get first account
      where account.balance > 500;
while DB-status = 0 do
      begin
             print (account.account-number);
             get next account
                     where account.balance > 500;
end
```

  - When **while** loop returns *DB-status* ≠ 0, we exhausted all account records with *account.balance* > 500.

Relational Data Model
(born in 1970, commercialized 1980)

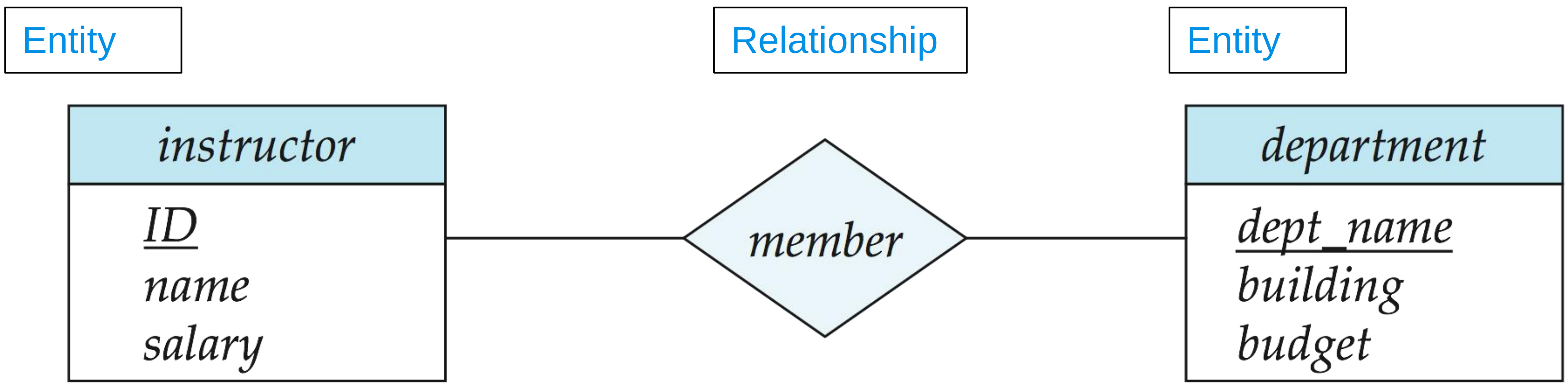
ID	*name*	*dept_name*	*salary*
22222	Einstein	Physics	95000
12121	Wu	Finance	90000
32343	El Said	History	60000
45565	Katz	Comp. Sci.	75000
98345	Kim	Elec. Eng.	80000
76766	Crick	Biology	72000
10101	Srinivasan	Comp. Sci.	65000
58583	Califieri	History	62000
83821	Brandt	Comp. Sci.	92000
15151	Mozart	Music	40000
33456	Gold	Physics	87000
76543	Singh	Finance	80000

(a) The *instructor* table

dept_name	*building*	*budget*
Comp. Sci.	Taylor	100000
Biology	Watson	90000
Elec. Eng.	Taylor	85000
Music	Packard	80000
Finance	Painter	120000
History	Painter	50000
Physics	Watson	70000

(b) The *department* table

E-R Data Model
(Conceptual Model for Database Table Design)



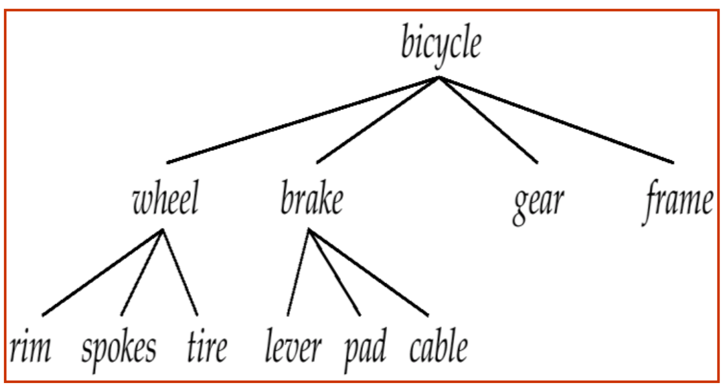
OR (Object-Relational) Data Model:
OO Extension of Relational Data Model

name	street	city	amount
Lowerly	Maple	Queens	900
Shiver	North	Bronx	556
Shiver	North	Bronx	647
Hodges	SideHill	Brooklyn	801
Hodges	SideHill	Brooklyn	647

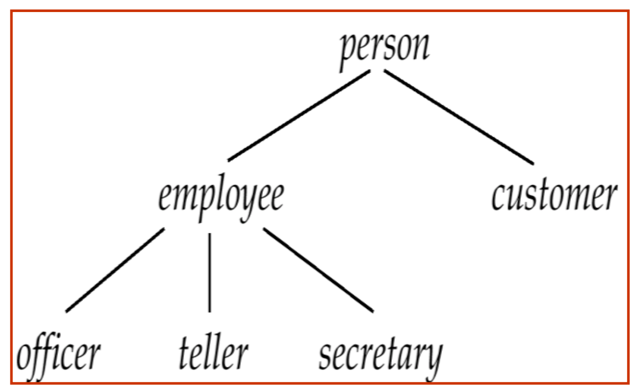
Only Tuples of Non-Decomposable Values

title	*author-set*	*publisher*	*keyword-set*
		(name, branch)	
Compilers	{Smith, Jones}	(McGraw-Hill, New York)	{parsing, analysis}
Networks	{Jones, Frick}	(Oxford, London)	{Internet, Web}

Set valued attributes
Relation valued attributes



Is-part-of relationship



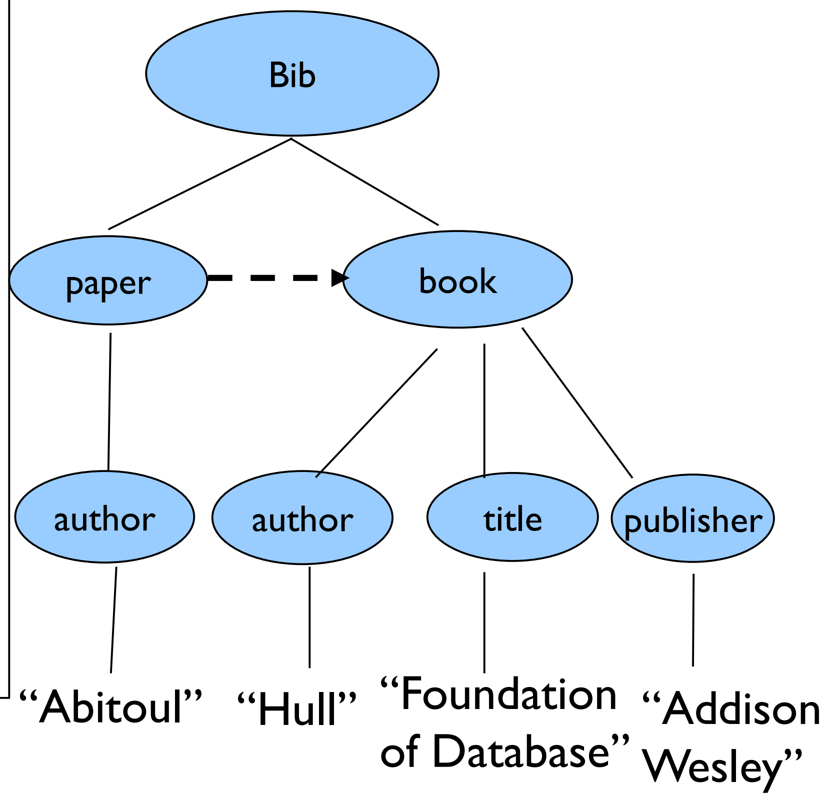
ISA relationship

XML Data Model (Modeling Document Data)

There is a paper by “Abitoul” referencing a book “Foundations of Data Bases” by Hull, from “Addison Wesley”

```
There is a paper by
<Bib>
   <paper id=“o2”    references=“o3”>
      <author> “Abitoul” </author>
    </paper>
   referencing a book
    <book id=“o3”>
       <title> “Foundations of Data Bases” </title>
       by <author> “Hull” </author>
       from <publisher> “Addison Wesley” </publisher>
     </book>
</Bib>
```

XML data



DOM Model

JavaScript Object Notation (JSON)

By Prof J. Widom

- Human-readable, useful for data interchange
- Modeling for representing & storing semistructured data

```
{ "Books":
  [
    { "ISBN":"ISBN-0-13-713526-2",
      "Price":85,
      "Edition":3,
      "Title":"A First Course in Database Systems",
      "Authors":[ {"First_Name":"Jeffrey", "Last_Name":"Ullman"},
                  {"First_Name":"Jennifer", "Last_Name":"Widom"} ] }
    ,
    { "ISBN":"ISBN-0-13-815504-6",
      "Price":100,
      "Remark":"Buy this book bundled with 'A First Course' - a great deal!",
      "Title":"Database Systems:The Complete Book",
      "Authors":[ {"First_Name":"Hector", "Last_Name":"Garcia-Molina"},
                  {"First_Name":"Jeffrey", "Last_Name":"Ullman"},
                  {"First_Name":"Jennifer", "Last_Name":"Widom"} ] }
  ]
```

And Many New Trend "NoSQL" Data Models

- NoSQL key-value stores
  - Cassandra (Apache)
  - Dynamo (Amazon)
  - Project Voldemort (LinkedIn)
  - Redis
- Document-based NoSQL systems
  - CouchDB (Apache)
  - MongoDB (10gen)
- Column-based NoSQL systems
  - Bigtable (Google)
  - Hbase(Apache)
  - Cassandra (Facebook -> Apache)
- Graph-based NoSQL systems
  - Neo4j
  - AllegroGraph
  - ArangoDB

Data Manipulation Language (DML)

- Language for accessing and manipulating the data organized by the appropriate data model
  - DML also known as query language
- Two classes of languages
  - **Procedural** – user specifies what data is required and how to get those data
  - **Declarative (nonprocedural)** – user specifies what data is required without specifying how to get those data
- SQL is the most widely used query language
  - 1980년, IBM Researcher, Don Chamberlain

Data Definition Language (DDL)

- Specification notation for defining the database schema

  Example: **create table** *instructor* (
  ID **char**(5),
  name **varchar**(20)**,**
  dept_name **varchar**(20),
  salary **numeric**(8,2))

- DDL compiler generates a set of table templates stored in a ***data dictionary***
- Data dictionary contains metadata (i.e., data about data)
  - Database schema
  - Integrity constraints
    - Primary key (ID uniquely identifies instructors)
    - Referential integrity (**references** constraint in SQL)
      - e.g. *dept_name* value in any *instructor* tuple must appear in *department* relation
  - Authorization

Relational Model

- Relational model (Chapter 2)
- Example of tabular data in the relational model

Columns

ID	*name*	*dept_name*	*salary*
22222	Einstein	Physics	95000
12121	Wu	Finance	90000
32343	El Said	History	60000
45565	Katz	Comp. Sci.	75000
98345	Kim	Elec. Eng.	80000
76766	Crick	Biology	72000
10101	Srinivasan	Comp. Sci.	65000
58583	Califieri	History	62000
83821	Brandt	Comp. Sci.	92000
15151	Mozart	Music	40000
33456	Gold	Physics	87000
76543	Singh	Finance	80000

Rows

(a) The *instructor* table

A Sample Relational Database and SQL

ID	*name*	*dept_name*	*salary*
22222	Einstein	Physics	95000
12121	Wu	Finance	90000
32343	El Said	History	60000
45565	Katz	Comp. Sci.	75000
98345	Kim	Elec. Eng.	80000
76766	Crick	Biology	72000
10101	Srinivasan	Comp. Sci.	65000
58583	Califieri	History	62000
83821	Brandt	Comp. Sci.	92000
15151	Mozart	Music	40000
33456	Gold	Physics	87000
76543	Singh	Finance	80000

(a) The *instructor* table

dept_name	*building*	*budget*
Comp. Sci.	Taylor	100000
Biology	Watson	90000
Elec. Eng.	Taylor	85000
Music	Packard	80000
Finance	Painter	120000
History	Painter	50000
Physics	Watson	70000

(b) The *department* table

- **SQL**: widely used non-procedural language
  - Example: Find the name of the instructor with ID 22222

    select *name*
    from *instructor*
    where *instructor.ID* = '22222'

  - Example: Find the ID and building of instructors in the Physics dept.

    select *instructor.ID*, *department.building*
    from *instructor*, *department*
    where *instructor.dept name* = "physics"

- Application programs generally access databases through one of
  - Language extensions to allow embedded SQL
  - Application program interface (e.g., ODBC/JDBC) which allow SQL queries to be sent to a database

- Chapters 3, 4 and 5

1.6 Database Design

The process of designing the general structure of the database:

- Logical Design – Deciding on the database schema.
  - Database design requires that we find a “good” collection of relation schemas.
  - Business decision – What attributes should we record in the database?
  - Computer Science decision – What relation schemas should we have and how should the attributes be distributed among the various relation schemas?
- Physical Design – Deciding on the physical layout of the database
  - Deciding types of attributes
  - Partition the relation into several small pieces vertically or horizontally
  - Index assignment on attributes: B-tree index or Hashing or something else..

Logical Database Design?

- Is there any problem with this design?

ID	*name*	*salary*	*dept_name*	*building*	*budget*
22222	Einstein	95000	Physics	Watson	70000
12121	Wu	90000	Finance	Painter	120000
32343	El Said	60000	History	Painter	50000
45565	Katz	75000	Comp. Sci.	Taylor	100000
98345	Kim	80000	Elec. Eng.	Taylor	85000
76766	Crick	72000	Biology	Watson	90000
10101	Srinivasan	65000	Comp. Sci.	Taylor	100000
58583	Califieri	62000	History	Painter	50000
83821	Brandt	92000	Comp. Sci	Taylor	100000
15151	Mozart	40000	Music	Packard	80000
33456	Gold	87000	Physics	Watson	70000
76543	Singh	80000	Finance	Painter	120000

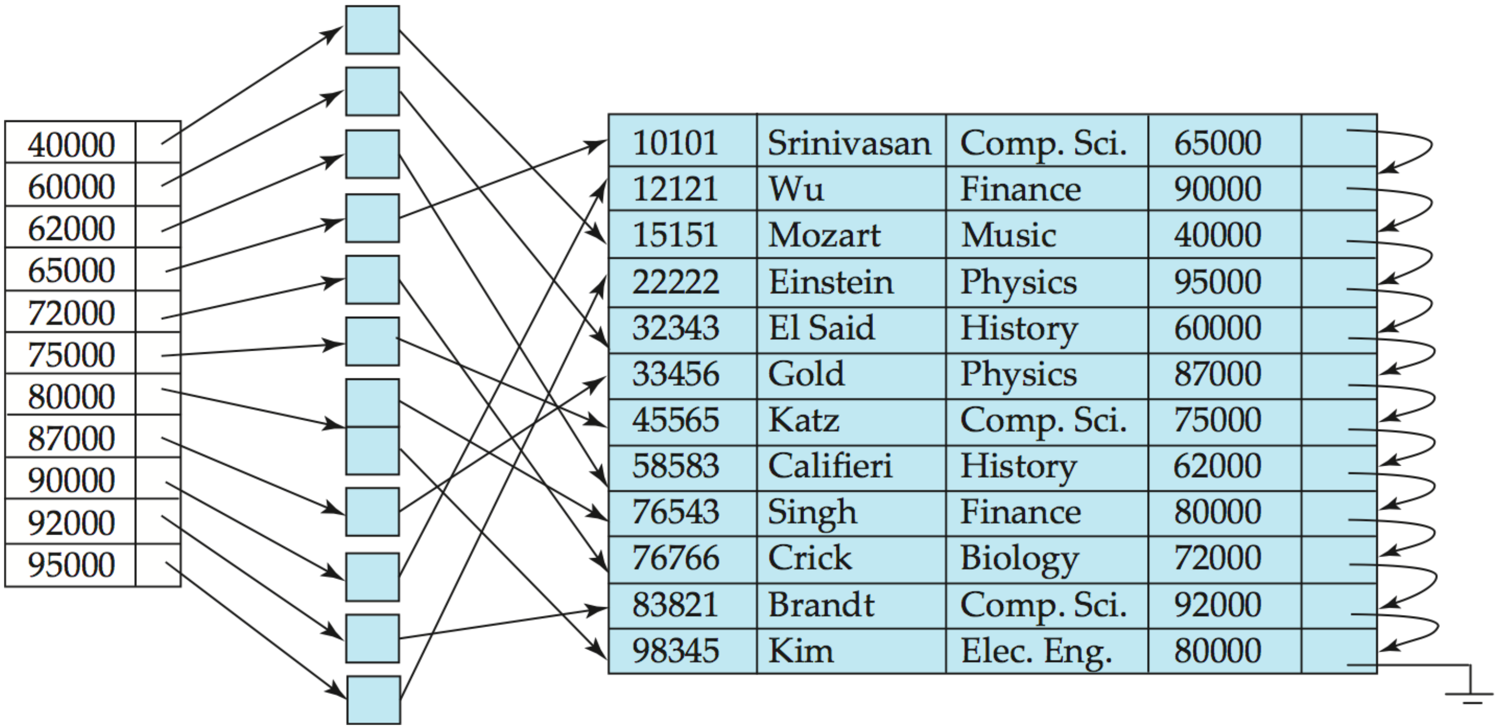
ID	*name*	*dept_name*	*salary*
22222	Einstein	Physics	95000
12121	Wu	Finance	90000
32343	El Said	History	60000
45565	Katz	Comp. Sci.	75000
98345	Kim	Elec. Eng.	80000
76766	Crick	Biology	72000
10101	Srinivasan	Comp. Sci.	65000
58583	Califieri	History	62000
83821	Brandt	Comp. Sci.	92000
15151	Mozart	Music	40000
33456	Gold	Physics	87000
76543	Singh	Finance	80000

(a) The *instructor* table

dept_name	*building*	*budget*
Comp. Sci.	Taylor	100000
Biology	Watson	90000
Elec. Eng.	Taylor	85000
Music	Packard	80000
Finance	Painter	120000
History	Painter	50000
Physics	Watson	70000

(b) The *department* table

Physical Database Design?



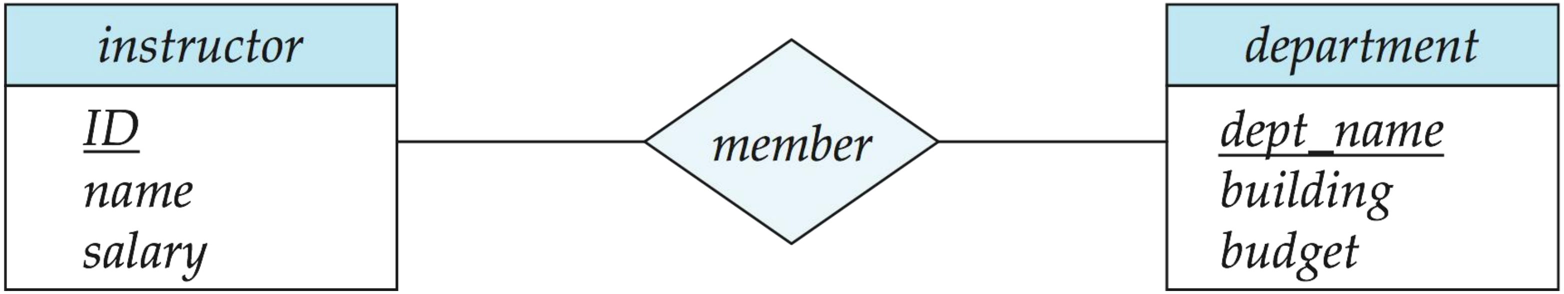
Design Approaches

- Normalization Theory (Chapter 8)
  - Formalize what designs are bad, and test for them

- Entity Relationship Model (Chapter 7)
  - Models an enterprise as a collection of *entities* and *relationships*
    - Entity: a “thing” or “object” in the enterprise that is distinguishable from other objects
      - Described by a set of *attributes*
    - Relationship: an association among several entities
  - Represented diagrammatically by an *entity-relationship diagram*

The Entity-Relationship Model

- Models an enterprise as a collection of *entities* and *relationships*
  - Entity: a "thing" or "object" in the enterprise that is distinguishable from other objects
    - Described by a set of *attributes*
  - Relationship: an association among several entities
- Represented diagrammatically by an *entity-relationship diagram:*



What happened to dept_name of instructor and student?

1.7 Data Storage and Querying

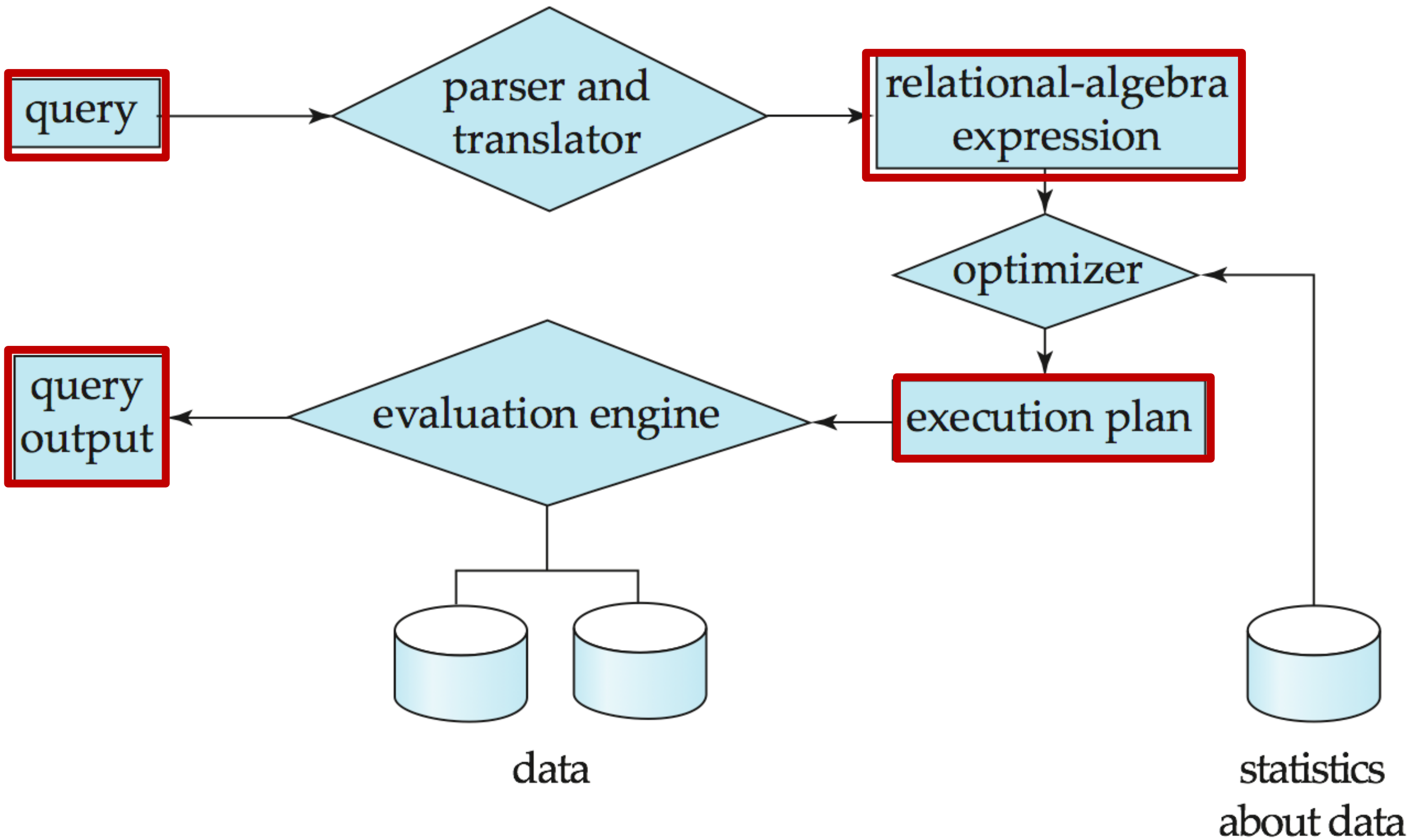
Storage Management

- **Storage manager** is lower-half of DBMS engine
  - a program module that provides the interface between the low-level data stored in the database and the application programs and queries submitted to the system
- The storage manager is responsible to the following tasks:
  - Interaction with the OS files
  - Efficient storing, retrieving and updating of data
- Modules
  - Buffer Manager
  - File Manager
  - Authorization Manager
  - Transaction Manager
- Maintaining Data Structures: Data files, Data Dictionary, Indices

Query Processing [1/2]

Query Processor is a higher half of DBMS engine

1. Parsing and translation
2. Optimization
3. Evaluation



Query Processing [2/2]

- Alternative ways of evaluating a given query
  - Equivalent expressions
  - Different algorithms for each operation
- Cost difference between a good and a bad way of evaluating a query can be enormous
- Need to estimate the cost of operations
  - Depends critically on statistical information about relations which the database must maintain
  - Need to estimate statistics for intermediate results to compute cost of complex expressions

1.8 Transaction Management

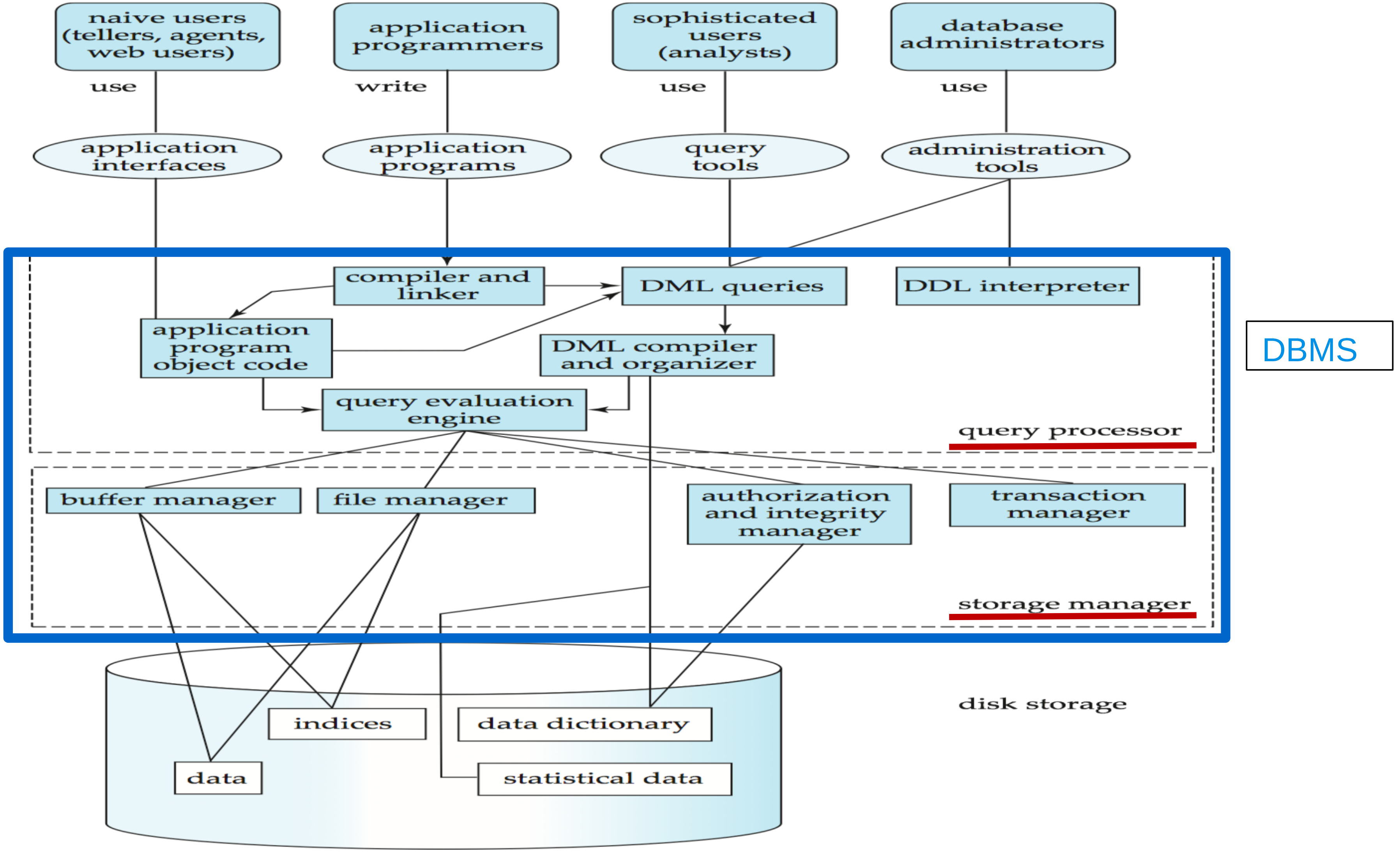
- What if the system fails?
- What if more than one user is concurrently updating the same data?
- A **transaction** is a collection of operations that performs a single logical function in a database application
- **Transaction-management component** ensures that the database remains in a consistent (correct) state despite system failures (e.g., power failures and operating system crashes) and transaction failures.
- **Concurrency-control manager** controls the interaction among the concurrent transactions, to ensure the consistency of the database.
- On-Line Transaction Processsing (OLTP)

Example of Transactions and Concurrent Access

- Transaction to transfer \$50 from account *A* to account *B*:
  1. **read**(*A*)
  2. *A* := *A* – 50
  3. **write**(*A*)
  4. **read**(*B*)
  5. *B* := *B* + 50
  6. **write**(*B)*

- Two people P1 and P2 are using two company debit cards for business
  - There is \$1000 in the company account
  - P1 is trying to retrieve \$500
  - P2 is trying to retrieve \$300

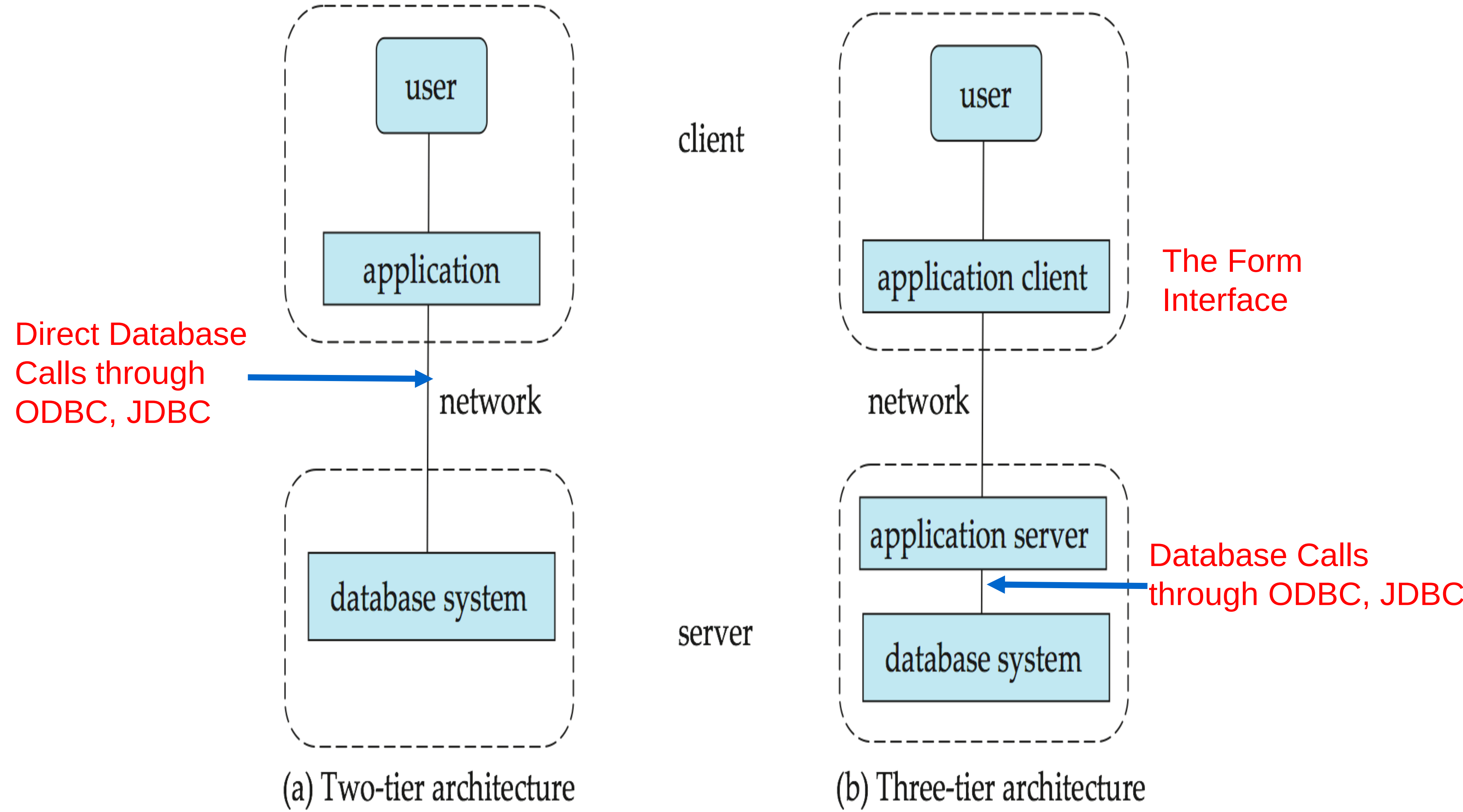
Overall Database System Structure



Types of Database Architecture

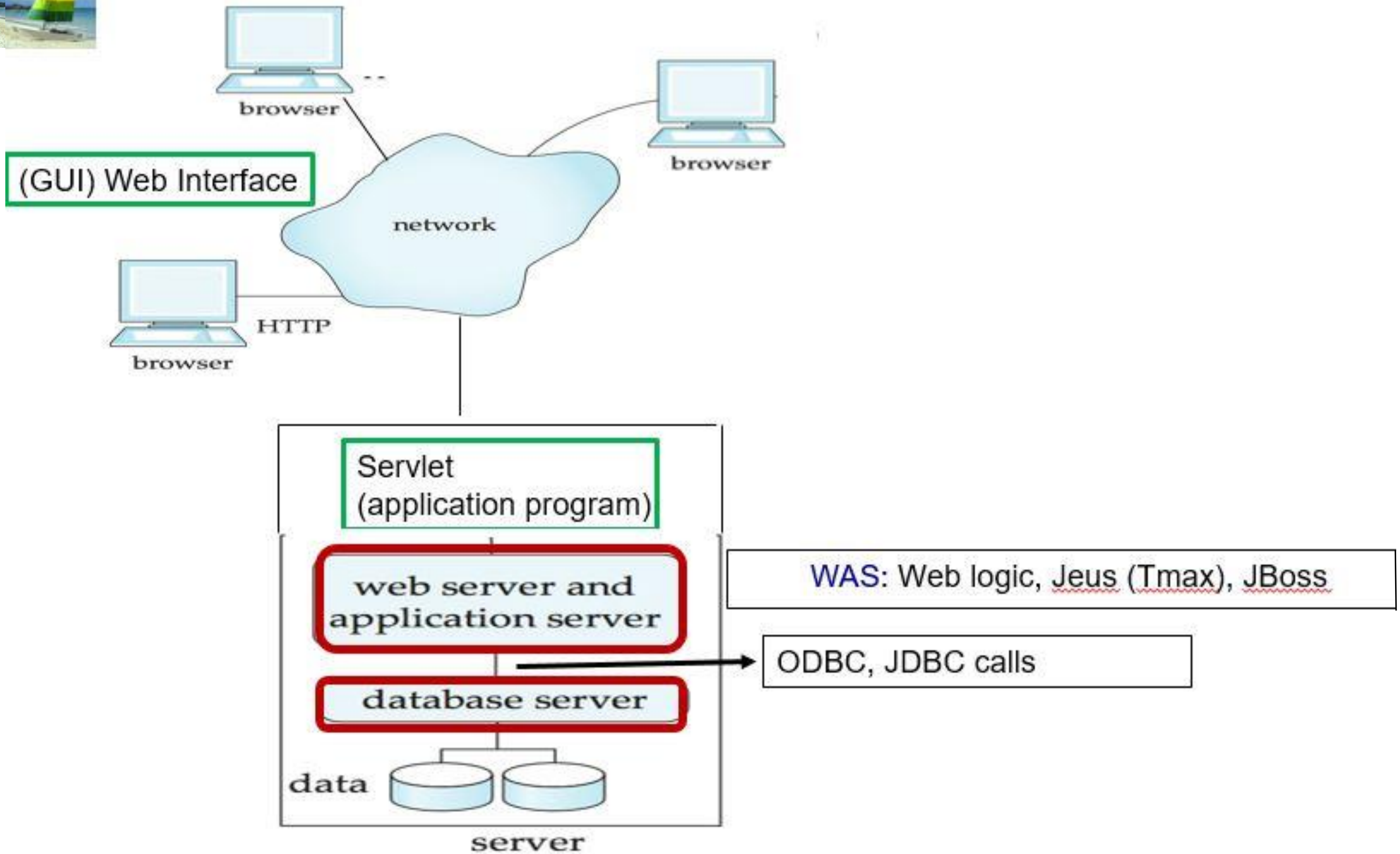
- The architecture of a database systems is greatly influenced by the underlying computer system on which the database is running:
  - Centralized
  - Client-server
  - Parallel (multi-processor)
  - Distributed

Figure 1.06: Client-Server DBMS Architecture



Web Application Architecture

참고사항



Parallel Database Architectures

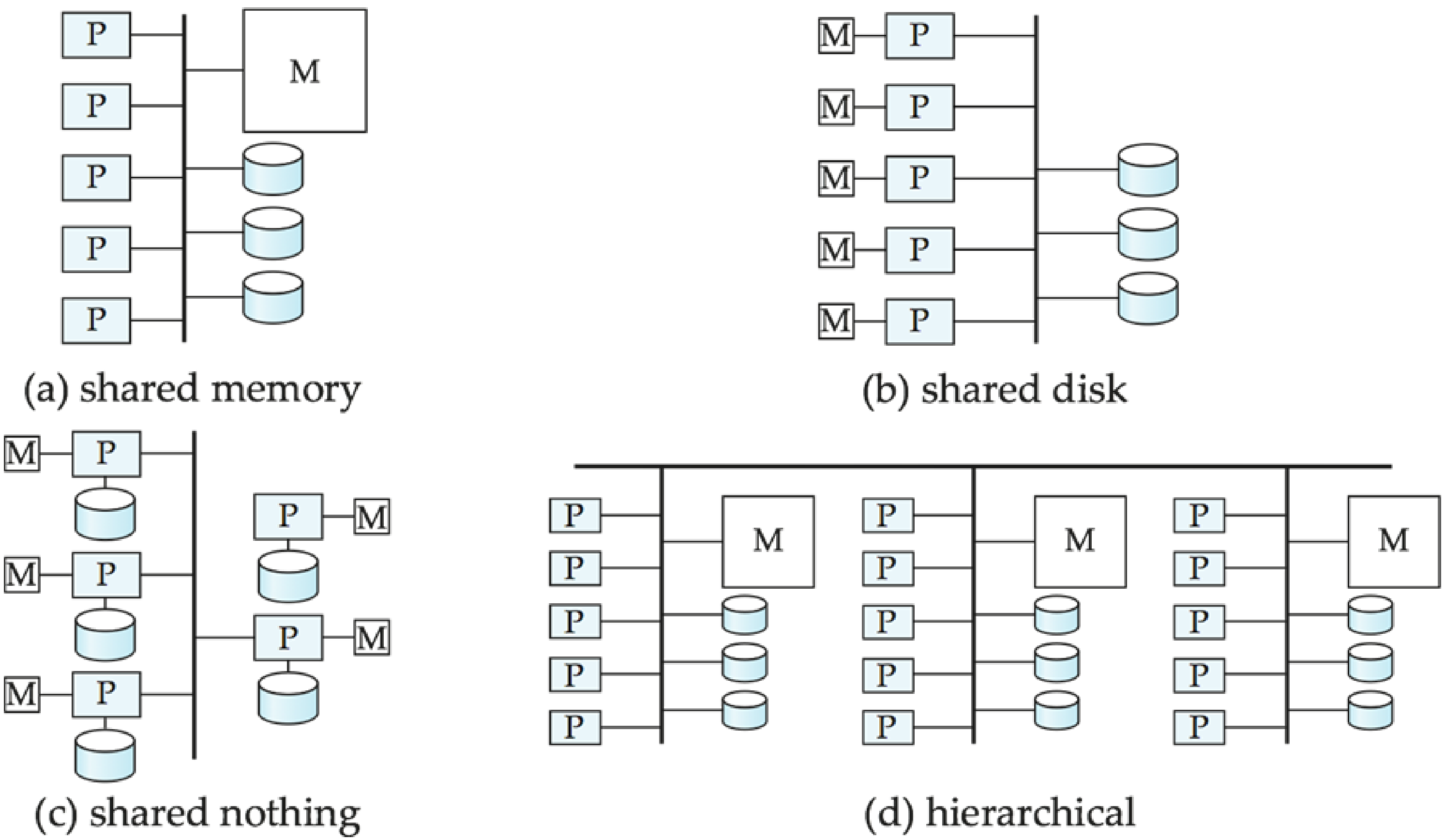


Fig 17.08

Distributed Database Systems

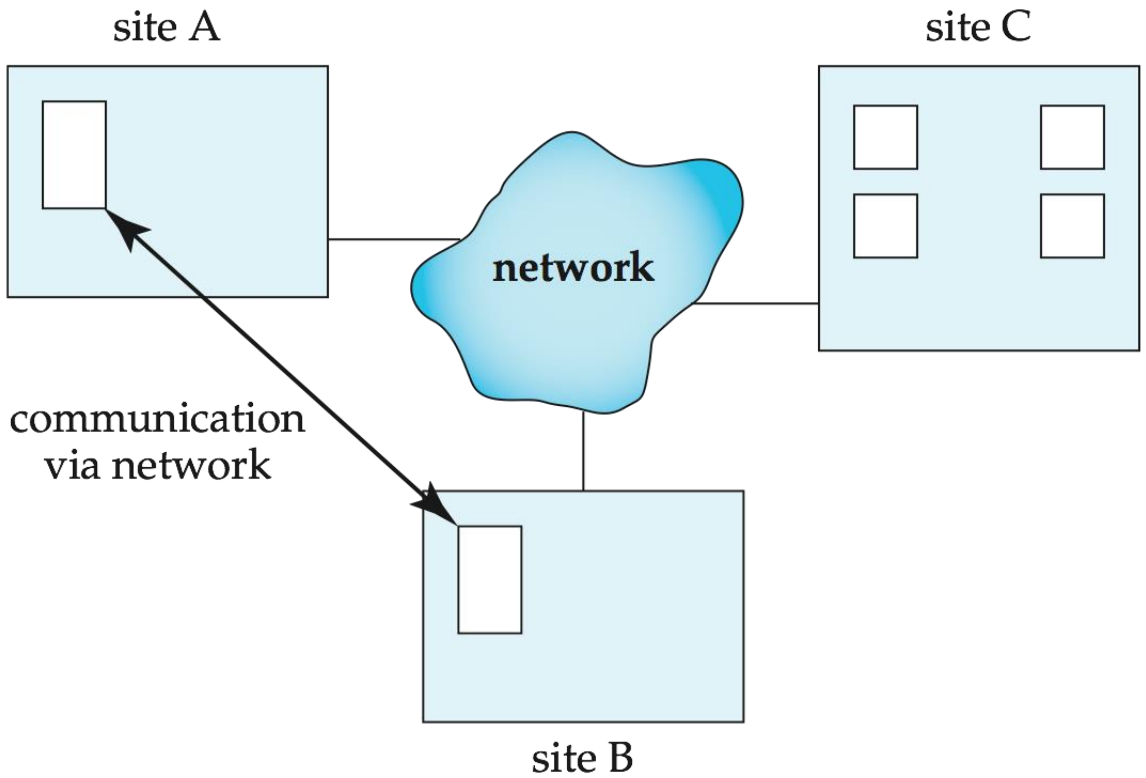


Fig 17.09

1.10 Data Mining and Information Retrieval

- Data Mining
  - The process of semiautomatically analyzing large databases to find useful patterns and rules
  - Similar to Knowledge Discovery in AI (also called Machine Learning), but dealing with very large database
  - Typical Application: Decision Support System for Business
    - Data-Warehouse (DW)
    - On-Line Analytical Processing (OLAP)
- Information Retrieval
  - From unstructured textual data
  - Keyword search
  - Now migrate to natural language processing
- Text mining, Machine Learning, Statistical Analysis are all strongly connected

Object-Relational Data Models

- Relational model: flat, “atomic” values

- Object Relational Data Models
  - Extend the relational data model by including object orientation and constructs to deal with added data types
  - Allow attributes of tuples to have complex types, including non-atomic values such as nested relations
  - Preserve relational foundations, in particular the declarative access to data, while extending modeling power
  - Provide upward compatibility with existing relational languages

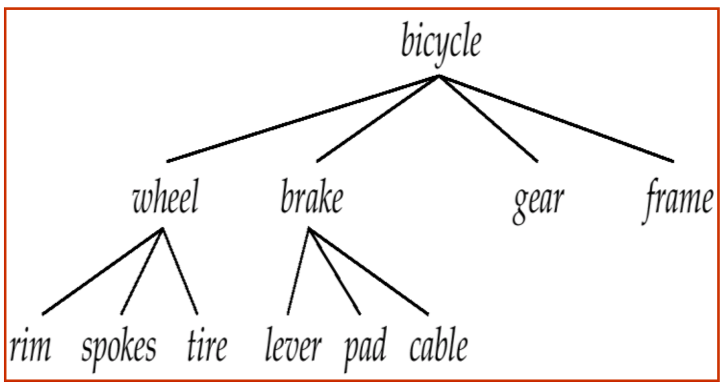
- Chapter 22: Object-based Databases

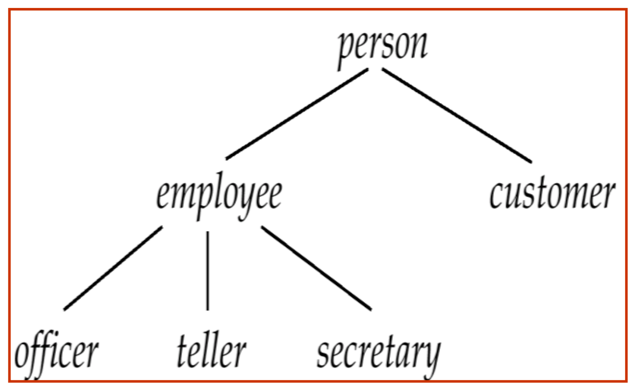
OR (Object-Relational) Data Model

name	street	city	amount
Lowerly	Maple	Queens	900
Shiver	North	Bronx	556
Shiver	North	Bronx	647
Hodges	SideHill	Brooklyn	801
Hodges	SideHill	Brooklyn	647

title	*author-set*	*publisher*	*keyword-set*
		(*name*, *branch*)	
Compilers	{Smith, Jones}	(McGraw-Hill, New York)	{parsing, analysis}
Networks	{Jones, Frick}	(Oxford, London)	{Internet, Web}

Set valued attributes
Relation valued attributes





Is-part-of relationship

ISA relationship

1.11 Specialty Databases
XML: Extensible Markup Language

- Defined by the WWW Consortium (W3C)
- Originally intended as a document markup language not a database language
- The ability to specify new tags, and to create nested tag structures made XML a great way to exchange **data**, not just documents
- XML has become the basis for all new generation data interchange formats.
- A wide variety of tools is available for parsing, browsing and querying XML documents/data

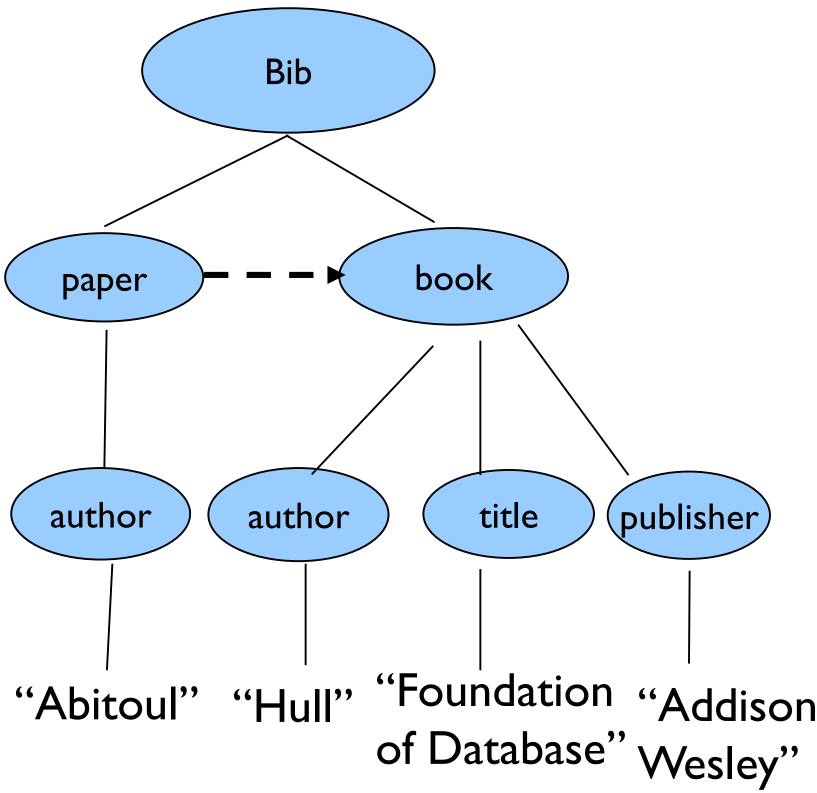
Chapter 23: XML

XML Data Model

There is a paper by “Abitoul” referencing a book “Foundations of Data Bases” by Hull, from “Addison Wesley”

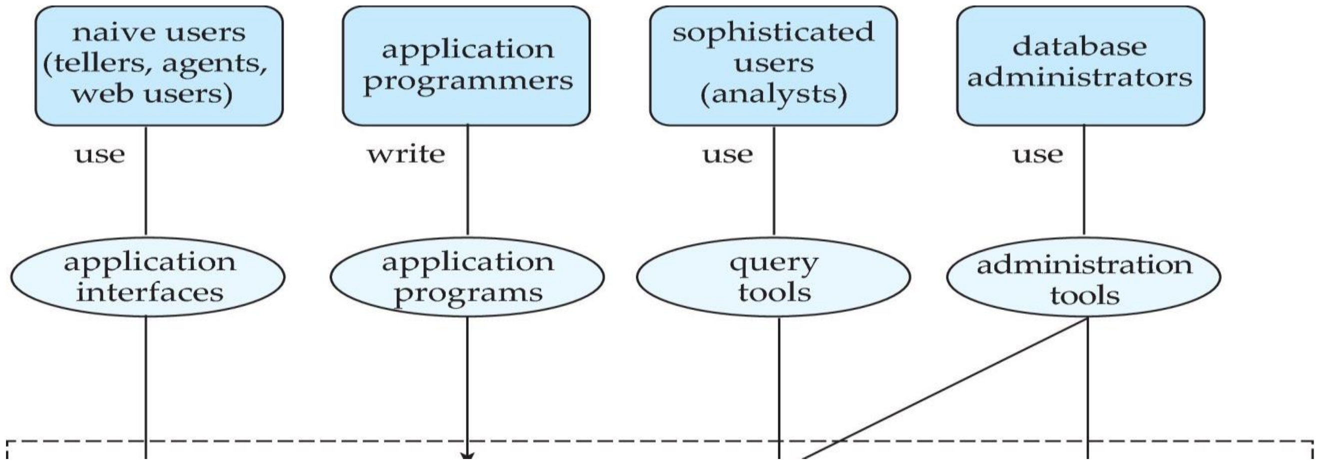
There is a paper by
<Bib>
<paper id=“o2” references=“o3”>
<author> “Abitoul” </author>
</paper>
referencing a book
<book id=“o3”>
<title> “Foundations of Data Bases” </title>
by <author> “Hull” </author>
from <publisher> “Addison Wesley” </publisher>
</book>
</Bib>

XML data



DOM Model

1.12 Database Users and Administrators



DataBase Management System

Users are differentiated by the way they expect to interact with the system

- **Application programmers** – interact with system through DML calls
- **Sophisticated users** – form requests in a database query language
- **Specialized users** – write specialized database applications that do not fit into the traditional data processing framework
- **Naïve users** – invoke one of the permanent application programs that have been written previously
  - Examples, people accessing database over the web, bank tellers, clerical staff

Database Administrator (DBA)

- Coordinates all the activities of the database system
- Must have a good understanding of the enterprise's information resources and needs
- Acts as liaison with users

- Database administrator's duties include:
  - Schema definition
    - Specifying integrity constraints
  - Storage structure and access method definition
  - Schema and physical organization modification
  - Granting user authority to access the database
  - Routine Maintenance
    - Periodic Back-Up
    - Monitoring performance and responding to changes in requirements

1.13 History of Database Systems [1/2]

- 1950s and early 1960s:
  - Data processing using magnetic tapes for storage
    - Tapes provided only sequential access
  - Punched cards for input
- Late 1960s and 1970s:
  - Hard disks allowed direct access to data
  - Network and hierarchical data models in widespread use
  - 1970: Ted Codd defines the relational data model
    - Would win the ACM Turing Award for this work
    - 1974: IBM Research begins System R prototype
    - 1974: UC Berkeley begins Ingres prototype
  - High-performance (for the era) transaction processing
- 1980s:
  - Research relational prototypes evolve into commercial systems
    - SQL becomes industrial standard
  - Parallel and distributed database systems
  - Object-oriented database systems

History of Database Systems [2/2]

- 1990s:
  - Large decision support and data-mining applications
  - Large multi-terabyte data warehouses
  - 1994: Emergence of Web!
  - XML and XQuery standards
- 2000s:
  - Automated database administration
  - Semantic Web (Ontology, RDF)
  - Giant data storage systems
    - Google BigTable, Yahoo PNuts, Amazon, ..
- 2010s ~~ Now
  - Big Data Analytics (convergence with Statistics)
  - DBMS support for Machine Learning / Data Mining
  - Streaming Data
  - Data from Internet of Things