Big Data

Data dependencies and DBMS

- ✓ RDBMS design
  - – the normal forms 1NF, 2NF, 3NF, BCNF
  - – normal forms transformation
- ✓ DBMS: architecture

Database Management Systems (DBMS)

Database management systems (DBMS)

- ✓ A **Database** is a collection of stored operational data used by the application systems of some particular enterprise (C.J. Date)
- ✓ A **DBMS** is a computer software with the capability to 1) store data in an integrated, structured format and to 2) enable users to retrieve, manipulate and manage the data.
  - Paper "Databases"
    - Still contain a large portion of the world's knowledge
  - File-Based Data Processing Systems
    - Early batch processing of (primarily) business data
  - Database Management Systems (DBMS)

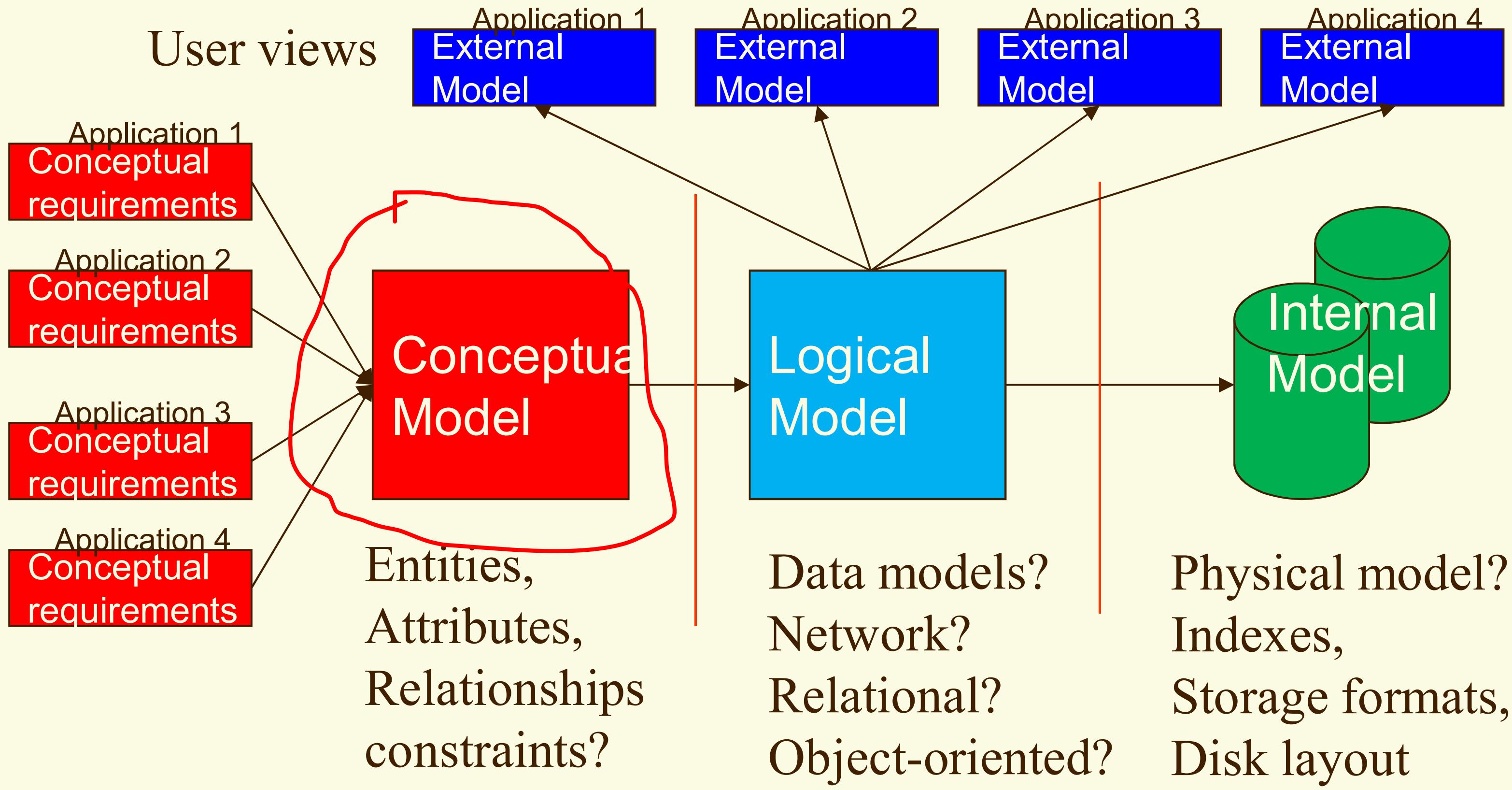
Data ≠ Information ≠ Knowledge!

Importance of DBMS

- It helps make data management more efficient and effective.
- Its query language allows quick answers to *ad hoc* queries.
- It provides end users better access to more and better-managed data.
- It promotes an integrated view of organization's operations -- "big picture."
- It reduces the probability of inconsistent data.

RDBMS design

DBMS design process



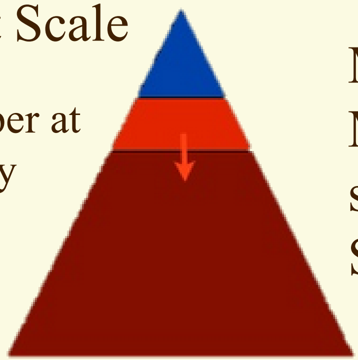
A Big data Fallacy

- "Database Design in the era of Big data is less important" (?)
  - New high-volume data streams
  - specialized hardware/softwares
  - Storage issues coped by hardware appliance
- **Fact:**
  - Most data is physically located in DBMS and new special-purpose appliance
  - Data loads, extract, transform, preprocessing ops continue as is
  - Database design for quality assurance

Big data a necessity at Largest Scale

A certain kind of developer at a certain kind of company

Most development still RDBMS



MySQL, Oracle, Mongo, Cassandra, some memcache, Some Hadoop…

Relational Databases Design

- Relational database design: The grouping of attributes to form "good" relation schemas
- Two levels of relation schemas:
  - The logical "user view" level
  - The storage "base relation" level
- Design is concerned mainly with base relations
- We have assumed schema *R* is given
  - *R* could have been generated when converting E-R diagram to a set of tables.
  - *R* could have been a single relation containing *all* attributes that are of interest (called **universal relation**).
  - **Normalization** breaks *R* into smaller relations.
  - *R* could have been the result of some ad hoc design of relations, which we then test/convert to normal form.

Database Tables and Normalization

Introduction to Normalization

- ✓ **Normalization**: Process of decomposing unsatisfactory "bad" relations by breaking up their attributes into smaller relations

- ✓ **Normal form**: Condition using keys and functional dependencies (FDs) of a relation to certify whether a relation schema is in a particular normal form
  - 2NF, 3NF, BCNF based on keys and FDs of a relation schema
  - 4NF based on keys, multi-valued dependencies

guarantees

Normal form (1NF)

Second normal form (2NF)

Third normal form (3NF)

Boyce-Codd Normal Form (BCNF)

stronger

The Need for Normalization

- ✓ Mixing attributes of multiple entities may cause problems
  - Information is stored redundantly wasting storage
  - Problems with update anomalies:
    - Insertion anomalies
    - Deletion anomalies
    - Modification anomalies
- ✓ The report may yield different results, depending on data anomaly
  - Primary keys?
  - Data redundancy
  - Possible data inconsistencies: JOB_CLASS: Elect.Engineer, Elect.Eng. El.Eng. EE

The Need for Normalization

- ✓ Example: company that manages building projects
  - Charges its clients by billing hours spent on each contract
  - Hourly billing rate is dependent on employee's position

TABLE 5.1 A SAMPLE REPORT LAYOUT

PROJ. NUM.	PROJECT NAME	EMPLOYEE NUMBER	EMPLOYEE NAME	JOB CLASS.	CHG/HOUR	HOURS BILLED	TOTAL CHARGE
15	Evergreen	103	June E. Arbough	Elec. Engineer	$84.50	23.8	$2,011.10
		101	John G. News	Database Designer	$105.00	19.4	$2,037.00
		105	Alice K. Johnson *	Database Designer	$105.00	35.7	$3,748.50
		106	William Smithfield	Programmer	$35.75	12.6	$450.45
		102	David H. Senior	Systems Analyst	$96.75	23.8	$2,302.65
				Subtotal			**$10,549.70**
18	Amber Wave	114	Annelise Jones	Applications Designer	$48.10	24.6	$1,183.26
		118	James J. Frommer	General Support	$18.36	45.3	$831.71
		104	Anne K. Ramoras *	Systems Analyst	$96.75	32.4	$3,135.70
		112	Darlene M. Smithson	DSS Analyst	$45.95	44.0	$2,021.80
				Subtotal			**$7,171.47**
22	Rolling Tide	105	Alice K. Johnson	Database Designer	$105.00	64.7	$6,793.50
		104	Anne K. Ramoras	Systems Analyst	$96.75	48.4	$4,682.70
		113	Delbert K. Joenbrood *	Applications Designer	$48.10	23.6	$1,135.16
		111	Geoff B. Wabash	Clerical Support	$26.87	22.0	$591.14
		106	William Smithfield	Programmer	$35.75	12.8	$457.60
				Subtotal			**$13,660.10**
25	Starflight	107	Maria D. Alonzo	Programmer	$35.75	24.6	$879.45
		115	Travis B. Bawangi	Systems Analyst	$96.75	45.8	$4,431.15
		101	John G. News *	Database Designer	$105.00	56.3	$5,911.50
		114	Annelise Jones	Applications Designer	$48.10	33.1	$1,592.11
		108	Ralph B. Washington	Systems Analyst	$96.75	23.6	$2,283.30
		118	James J. Frommer	General Support	$18.36	30.5	$559.98
		112	Darlene M. Smithson	DSS Analyst	$45.95	41.4	$1,902.33
				Subtotal			**$17,559.82**
				Total			**$48,941.09**

* Indicates project leader

A Table in the Report Format

FIGURE 5.1 A TABLE IN THE REPORT FORMAT

Table name: RPT_FORMAT | Database name: Ch05_ConstructCo

PROJ_NUM	PROJ_NAME	EMP_NUM	EMP_NAME	JOB_CLASS	CHG_HOUR	HOURS
15	Evergreen	103	June E. Arbough	Elect. Engineer	$84.50	23.8
		101	John G. News	Database Designer	$105.00	19.4
		105	Alice K. Johnson *	Database Designer	$105.00	35.7
		106	William Smithfield	Programmer	$35.75	12.6
		102	David H. Senior	Systems Analyst	$96.75	23.8
18	Amber Wave	114	Annelise Jones	Applications Designer	$48.10	24.6
		118	James J. Frommer	General Support	$18.36	45.3
		104	Anne K. Ramoras *	Systems Analyst	$96.75	32.4
		112	Darlene M. Smithson	DSS Analyst	$45.95	44.0
22	Rolling Tide	105	Alice K. Johnson	Database Designer	$105.00	64.7
		104	Anne K. Ramoras	Systems Analyst	$96.75	48.4
		113	Delbert K. Joenbrood *	Applications Designer	$48.10	23.6
		111	Geoff B. Wabash	Clerical Support	$26.87	22.0
		106	William Smithfield	Programmer	$35.75	12.8
25	Starflight	107	Maria D. Alonzo	Programmer	$35.75	24.6
		115	Travis B. Bawangi	Systems Analyst	$96.75	45.8
		101	John G. News *	Database Designer	$105.00	56.3
		114	Annelise Jones	Applications Designer	$48.10	33.1
		108	Ralph B. Washington	Systems Analyst	$96.75	23.6
		118	James J. Frommer	General Support	$18.36	30.5
		112	Darlene M. Smithson	DSS Analyst	$45.95	41.4

The Need for Normalization

- ✓ For most business database design, 3NF is highest we need to go in the normalization process
  - Highest level of normalization is not always most desirable
- ✓ Normalization should be part of design process
  - Make sure that proposed entities meet required normal form before table structures are created
  - Many real-world databases have been improperly designed or burdened with anomalies if improperly modified during course of time
  - You may be asked to redesign and modify existing databases

Functional Dependencies

- **Functional dependencies** (FDs) are used to specify formal measures of the "goodness" of relational designs
- FDs and keys are used to define normal forms for relations
- FDs are constraints that are derived from the meaning and interrelationships of the data attributes

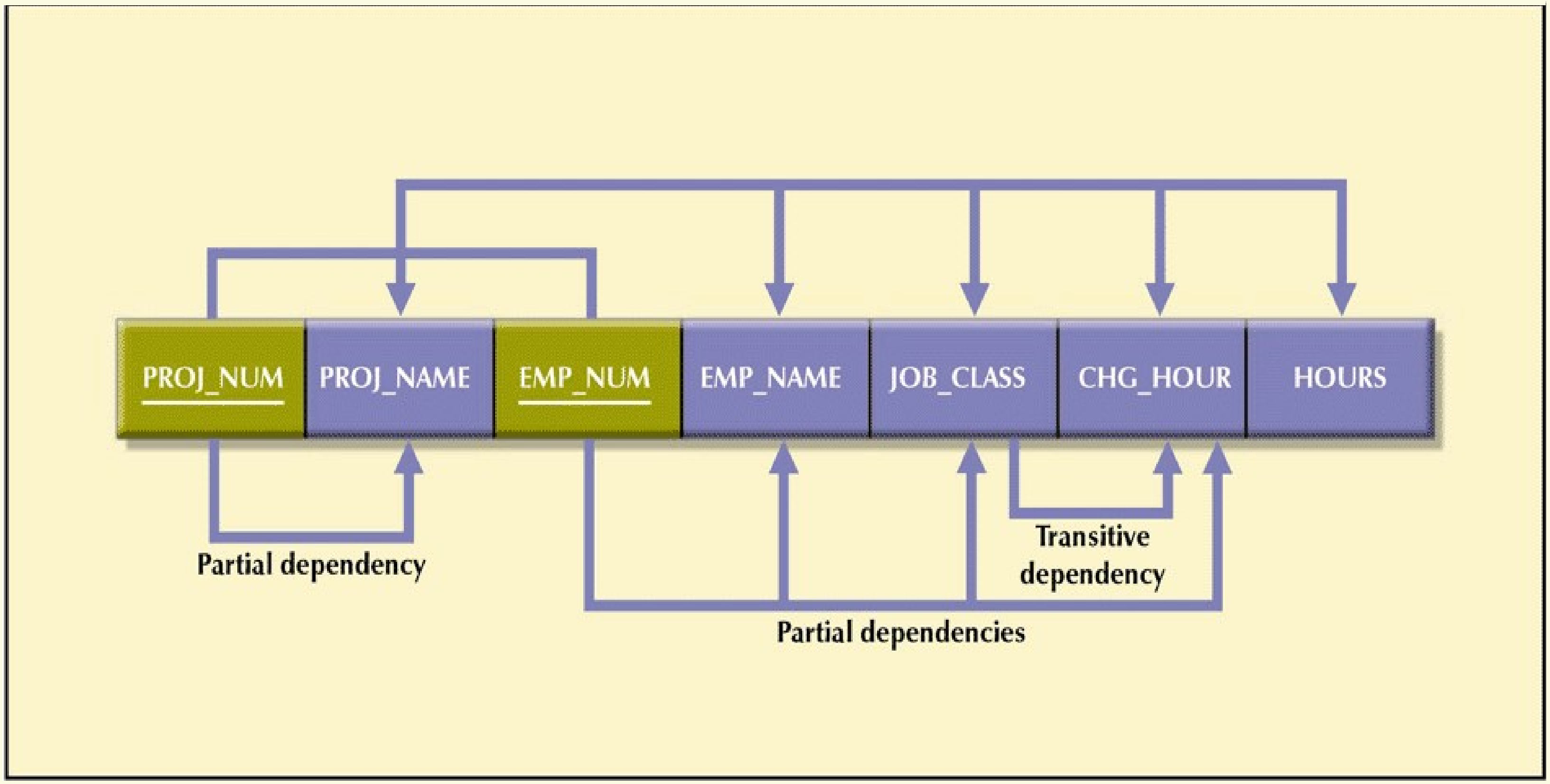
Functional Dependencies (2)

- A set of attributes X *functionally determines* a set of attributes Y if the value of X determines a unique value for Y
- X → Y holds if whenever two tuples have the same value for X, they *must have* the same value for Y

  If t1[X]=t2[X], *then* t1[Y]=t2[Y] in any relation instance r(R)
- a *constraint* on ***all*** relation instances r(R)
- FDs are derived from the real-world constraints on the attributes
  - SSN → ENAME
  - PNUMBER → {PNAME, PLOCATION}
  - {SSN, PNUMBER} → HOURS
  - what if LHS attributes is a **key**?

- A FD X → Y is a
  - **Full functional dependency** if removal of any attribute from X means the FD does not hold any more
  - **Transitive functional dependency**- if there a set of attributes Z that are neither a primary or candidate key and both X→ Z and Z → Y holds.

A Dependency Diagram



Inference Rules for FDs

- Given a set of FDs F, we can infer additional FDs that hold whenever the FDs in F hold

- Armstrong's inference rules

  A1. (Reflexive) If Y subset-of X, then X → Y

  A2. (Augmentation) If X → Y, then XZ → YZ
  (Notation: XZ stands for X U Z)

  A3. (Transitive) If X → Y and Y → Z, then X → Z

- A1, A2, A3 form a *sound and complete* set of inference rules

First Normal Form

- ✓ Tabular format in which:
  - All key attributes are defined
  - There are no repeating groups in the table
  - All attributes are dependent on primary key

- ✓ Disallows composite attributes, multivalued attributes, and **nested relations**; attributes whose values *for an individual tuple* are non-atomic

1NF deals with the "shape" of the tables

Transform to 1NF

- ✓ Step 1: Eliminate repeating groups
- ✓ Step 2: Identify primary key
- ✓ (Step 3: Identify all dependencies)

FIGURE 5.2 DATA ORGANIZATION: FIRST NORMAL FORM

Table name: DATA_ORG_1NF   Database name: Ch05_ConstructCo

PROJ_NUM	PROJ_NAME	EMP_NUM	EMP_NAME	JOB_CLASS	CHG_HOUR	HOURS
15	Evergreen	103	June E. Arbough	Elect. Engineer	$84.50	23.8
15	Evergreen	101	John G. News	Database Designer	$105.00	19.4
15	Evergreen	105	Alice K. Johnson *	Database Designer	$105.00	35.7
15	Evergreen	106	William Smithfield	Programmer	$35.75	12.6
15	Evergreen	102	David H. Senior	Systems Analyst	$96.75	23.8
18	Amber Wave	114	Annelise Jones	Applications Designer	$48.10	24.6
18	Amber Wave	118	James J. Frommer	General Support	$18.36	45.3
18	Amber Wave	104	Anne K. Ramoras *	Systems Analyst	$96.75	32.4
18	Amber Wave	112	Darlene M. Smithson	DSS Analyst	$45.95	44.0
22	Rolling Tide	105	Alice K. Johnson	Database Designer	$105.00	64.7
22	Rolling Tide	104	Anne K. Ramoras	Systems Analyst	$96.75	48.4
22	Rolling Tide	113	Delbert K. Joenbrood *	Applications Designer	$48.10	23.6
22	Rolling Tide	111	Geoff B. Wabash	Clerical Support	$26.87	22.0
22	Rolling Tide	106	William Smithfield	Programmer	$35.75	12.8
25	Starflight	107	Maria D. Alonzo	Programmer	$35.75	24.6
25	Starflight	115	Travis B. Bawangi	Systems Analyst	$96.75	45.8
25	Starflight	101	John G. News *	Database Designer	$105.00	56.3
25	Starflight	114	Annelise Jones	Applications Designer	$48.10	33.1
25	Starflight	108	Ralph B. Washington	Systems Analyst	$96.75	23.6
25	Starflight	118	James J. Frommer	General Support	$18.36	30.5
25	Starflight	112	Darlene M. Smithson	DSS Analyst	$45.95	41.4

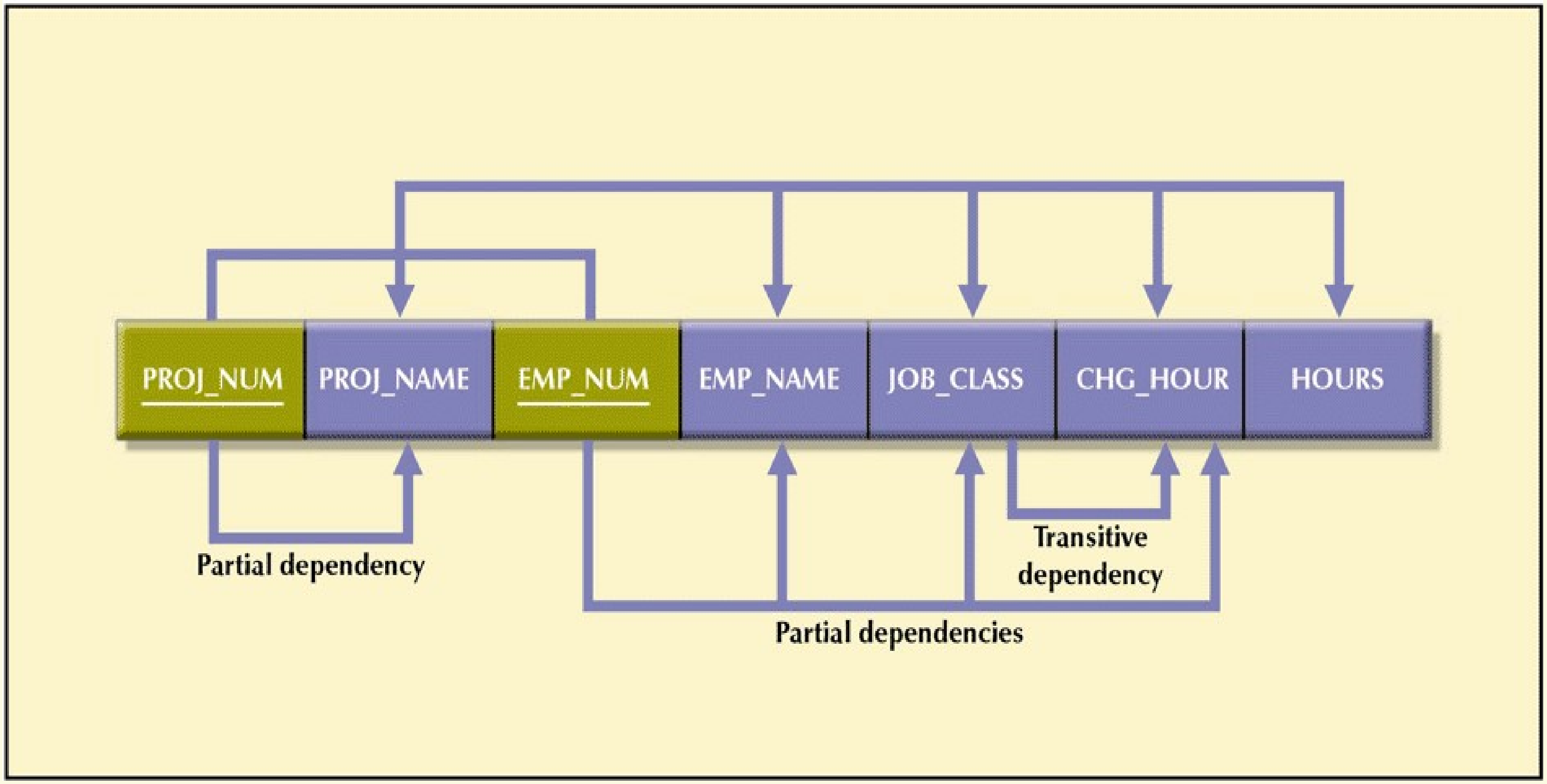
Second Normal Form

- ✓ Table is in second normal form (2NF) if:
  - It is in 1NF and
  - It includes no partial dependencies:
    - No attribute is dependent on only a portion of the primary key = every attribute A not in PK is fully functionally dependent on PK

2NF deals with the relationship between non-key and key attributes

A Dependency Diagram: First Normal Form (1NF)

FIGURE 5.3 A DEPENDENCY DIAGRAM: FIRST NORMAL FORM (1NF)

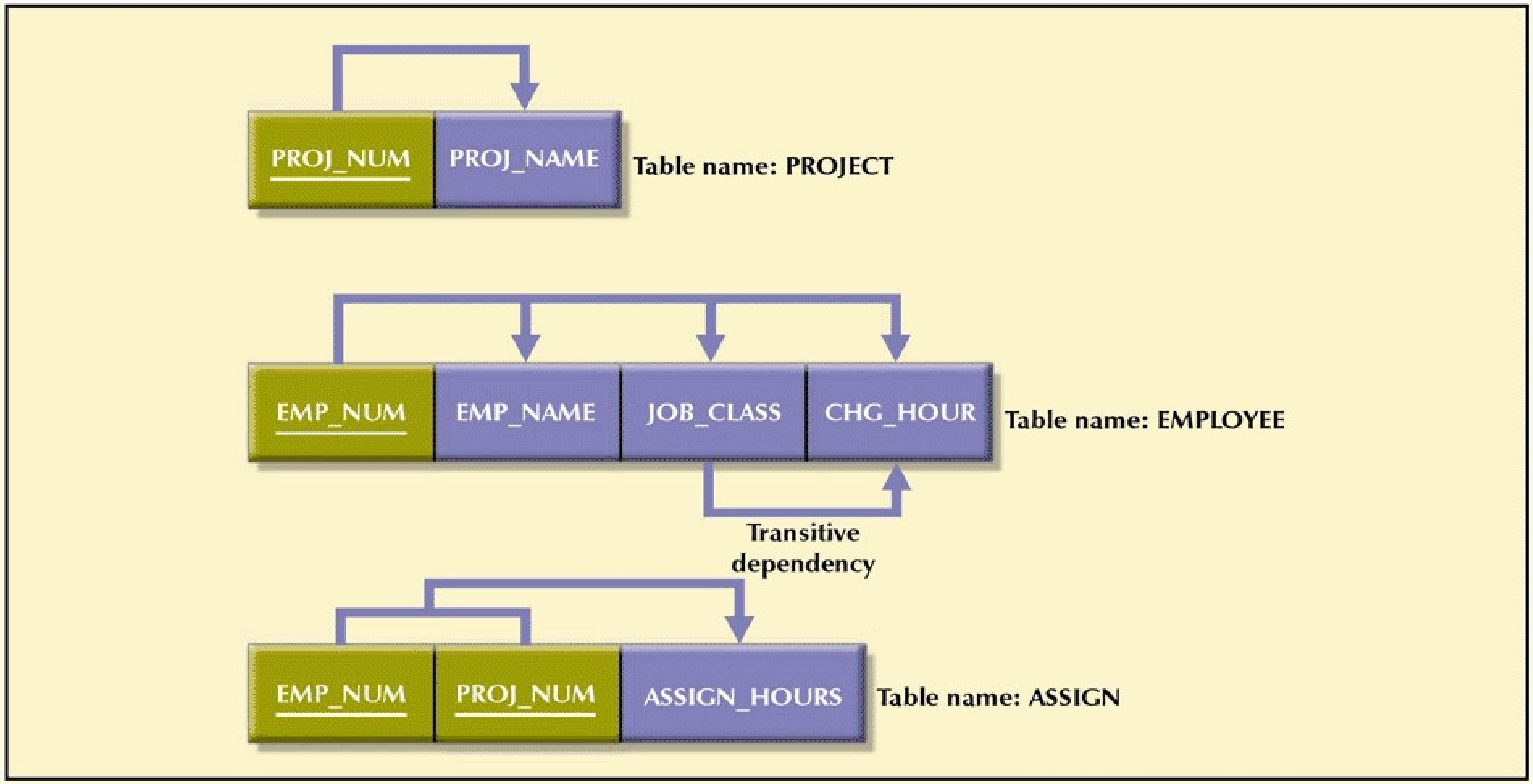


Conversion to Second Normal Form

- Relational database design can be improved by converting the database into second normal form (2NF)
- Two steps
  - Step 1: Write each key attribute on separate line, and then write the original (composite) key on the last line; Each component will become the key in a new table
  - Step 2: Determine which attributes are dependent on which other attributes (remove anomalies)

Second Normal Form (2NF) Conversion Results

FIGURE 5.4 SECOND NORMAL FORM (2NF) CONVERSION RESULTS



Third Normal Form

- A table is in third normal form (3NF) if:
  - It is in 2NF and
  - It contains no transitive dependencies

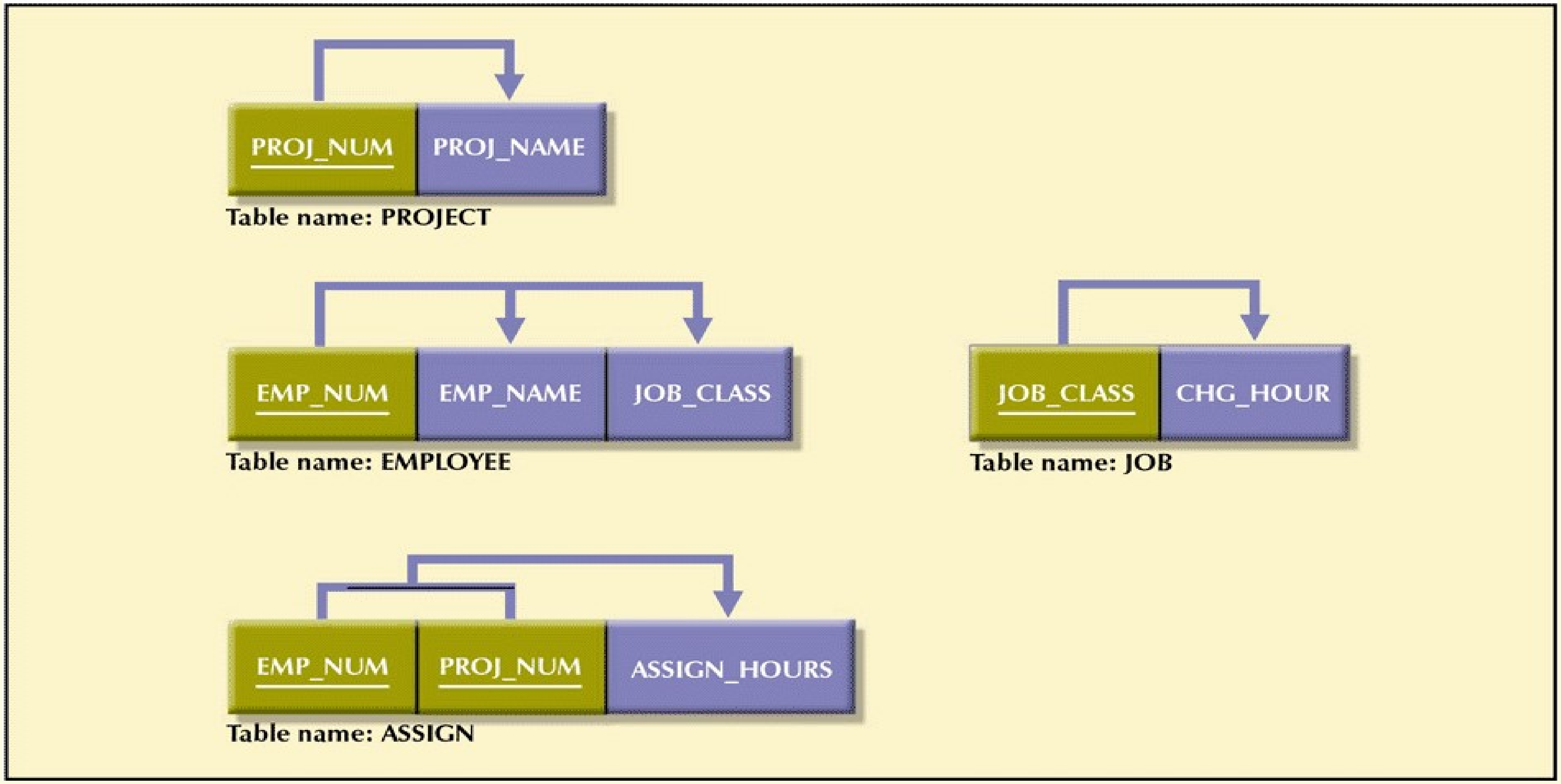
3NF removes transitive dependencies

Conversion to Third Normal Form

- ✓ Data anomalies created are easily eliminated by completing three steps
  - **Step 1: find new fact:** For every transitive dependency X-Y, write fact Z as a PK for a new table where X->Z and Z->Y
  - **Step 2: Identify the Dependent Attributes** - Identify the attributes dependent on each Z identified in Step 1 and identify the dependency; Name the table to reflect its contents and function
  - **Step 3: Remove X->Y from Transitive Dependencies**

Third Normal Form (3NF) Conversion Results

FIGURE 5.5 THIRD NORMAL FORM (3NF) CONVERSION RESULTS



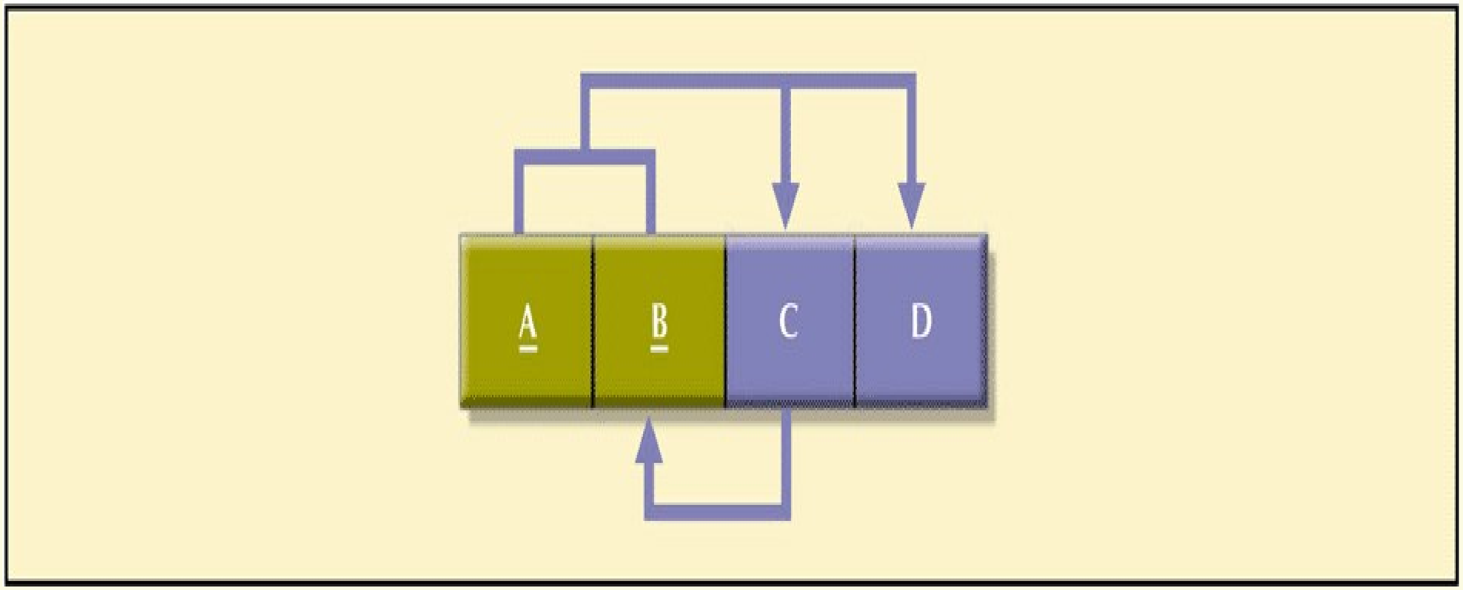
data depends on the key [1NF], the whole key [2NF] and nothing but the key [3NF]

BCNF (Boyce-Codd Normal Form)

- ✓ A relation schema R is in **Boyce-Codd Normal Form** (BCNF), aka 3.5NF, if whenever an FD $X \rightarrow A$ holds in R, then X is a superkey of R
  - Each normal form is strictly stronger than the previous one:
    - Every 2NF relation is in 1NF
    - Every 3NF relation is in 2NF
    - Every BCNF relation is in 3NF
  - There exist relations that are in 3NF but not in BCNF
  - The goal is to have each relation in BCNF (or 3NF)
  - Why not 4NF? 4 and 5NF deal with multi-valued attributes
  - Just slightly stricter than 3NF

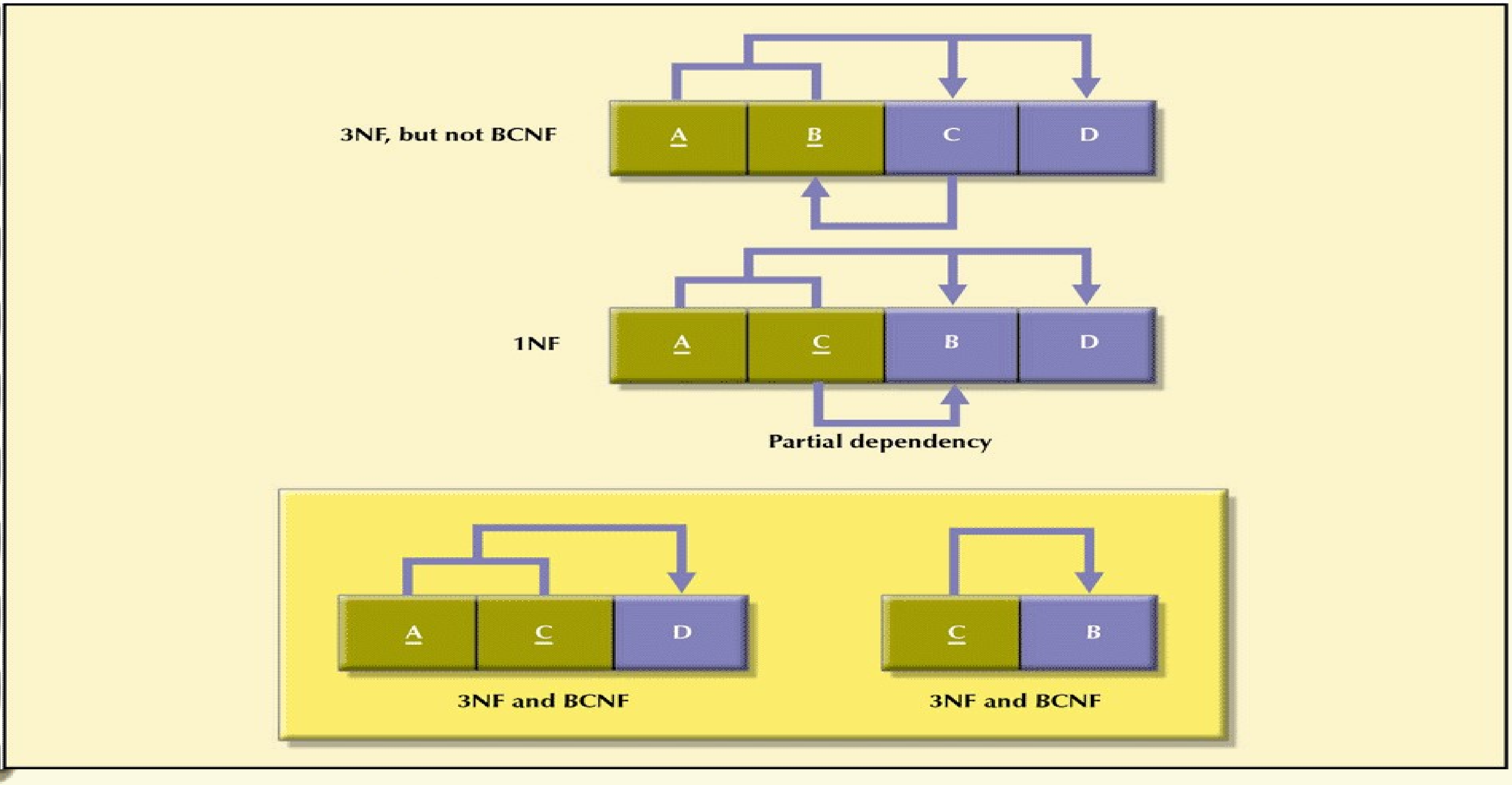
A Table That is in 3NF but not in BCNF

FIGURE 5.7 A TABLE THAT IS IN 3NF BUT NOT IN BCNF



Decomposition to BCNF

FIGURE 5.8 DECOMPOSITION TO BCNF



The Completed Database

FIGURE 5.6 THE COMPLETED DATABASE

Database name: Ch05_ConstructCo

Table name: PROJECT

PROJ_NUM	PROJ_NAME	EMP_NUM
15	Evergreen	105
18	Amber Wave	104
22	Rolling Tide	113
25	Starflight	101

Table name: JOB

JOB_CODE	JOB_DESCRIPTION	JOB_CHG_HOUR
500	Programmer	$35.75
501	Systems Analyst	$96.75
502	Database Designer	$105.00
503	Electrical Engineer	$84.50
504	Mechanical Engineer	$67.90
505	Civil Engineer	$55.78
506	Clerical Support	$26.87
507	DSS Analyst	$45.95
508	Applications Designer	$48.10
509	Bio Technician	$34.55
510	General Support	$18.36

FIGURE 5.6 The Completed Database (Continued)

Table name: ASSIGN

ASSIGN_NUM	ASSIGN_DATE	PROJ_NUM	EMP_NUM	ASSIGN_HOURS	ASSIGN_CHG_HOUR	ASSIGN_CHARGE

Table name: ASSIGN **Database name: Ch05_ConstructCo**

ASSIGN_NUM	ASSIGN_DATE	PROJ_NUM	EMP_NUM	ASSIGN_HOURS	ASSIGN_CHG_HOUR	ASSIGN_CHARGE
1001	04-Mar-04	15	103	2.6	$84.50	$219.70
1002	04-Mar-04	18	118	1.4	$18.36	$25.70
1003	05-Mar-04	15	101	3.6	$105.00	$378.00
1004	05-Mar-04	22	113	2.5	$48.10	$120.25
1005	05-Mar-04	15	103	1.9	$84.50	$160.55
1006	05-Mar-04	25	115	4.2	$96.75	$406.35
1007	05-Mar-04	22	105	5.2	$105.00	$546.00
1008	05-Mar-04	25	101	1.7	$105.00	$178.50
1009	05-Mar-04	15	105	2.0	$105.00	$210.00
1010	06-Mar-04	15	102	3.8	$96.75	$367.65
1011	06-Mar-04	22	104	2.6	$96.75	$251.55
1012	06-Mar-04	15	101	2.3	$105.00	$241.50
1013	06-Mar-04	25	114	1.8	$48.10	$86.58
1014	06-Mar-04	22	111	4.0	$26.87	$107.48
1015	06-Mar-04	25	114	3.4	$48.10	$163.54
1016	06-Mar-04	18	112	1.2	$45.95	$55.14
1017	06-Mar-04	18	118	2.0	$18.36	$36.72
1018	06-Mar-04	18	104	2.6	$96.75	$251.55
1019	06-Mar-04	15	103	3.0	$84.50	$253.50
1020	07-Mar-04	22	105	2.7	$105.00	$283.50
1021	08-Mar-04	25	108	4.2	$96.75	$406.35
1022	07-Mar-04	25	114	5.8	$48.10	$278.98
1023	07-Mar-04	22	106	2.4	$35.75	$85.80

Table name: EMPLOYEE

EMP_NUM	EMP_LNAME	EMP_FNAME	EMP_INITIAL	EMP_HIREDATE	JOB_CODE

Table name: EMPLOYEE

EMP_NUM	EMP_LNAME	EMP_FNAME	EMP_INITIAL	EMP_HIREDATE	JOB_CODE
101	News	John	G	08-Nov-98	502
102	Senior	David	H	12-Jul-87	501
103	Arbough	June	E	01-Dec-94	503
104	Ramoras	Anne	K	15-Nov-85	501
105	Johnson	Alice	K	01-Feb-91	502
106	Smithfield	William		22-Jun-03	500
107	Alonzo	Maria	D	10-Oct-91	500
108	Washington	Ralph	B	22-Aug-89	501
109	Smith	Larry	W	18-Jul-95	501
110	Olenko	Gerald	A	11-Dec-93	505
111	Wabash	Geoff	B	04-Apr-89	506
112	Smithson	Darlene	M	23-Oct-92	507
113	Joenbrood	Delbert	K	15-Nov-94	508
114	Jones	Annelise		20-Aug-91	508
115	Bawangi	Travis	B	25-Jan-90	501
116	Pratt	Gerald	L	05-Mar-95	510
117	Williamson	Angie	H	19-Jun-94	509
118	Frommer	James	J	04-Jan-04	510

DBMS architecture: overview

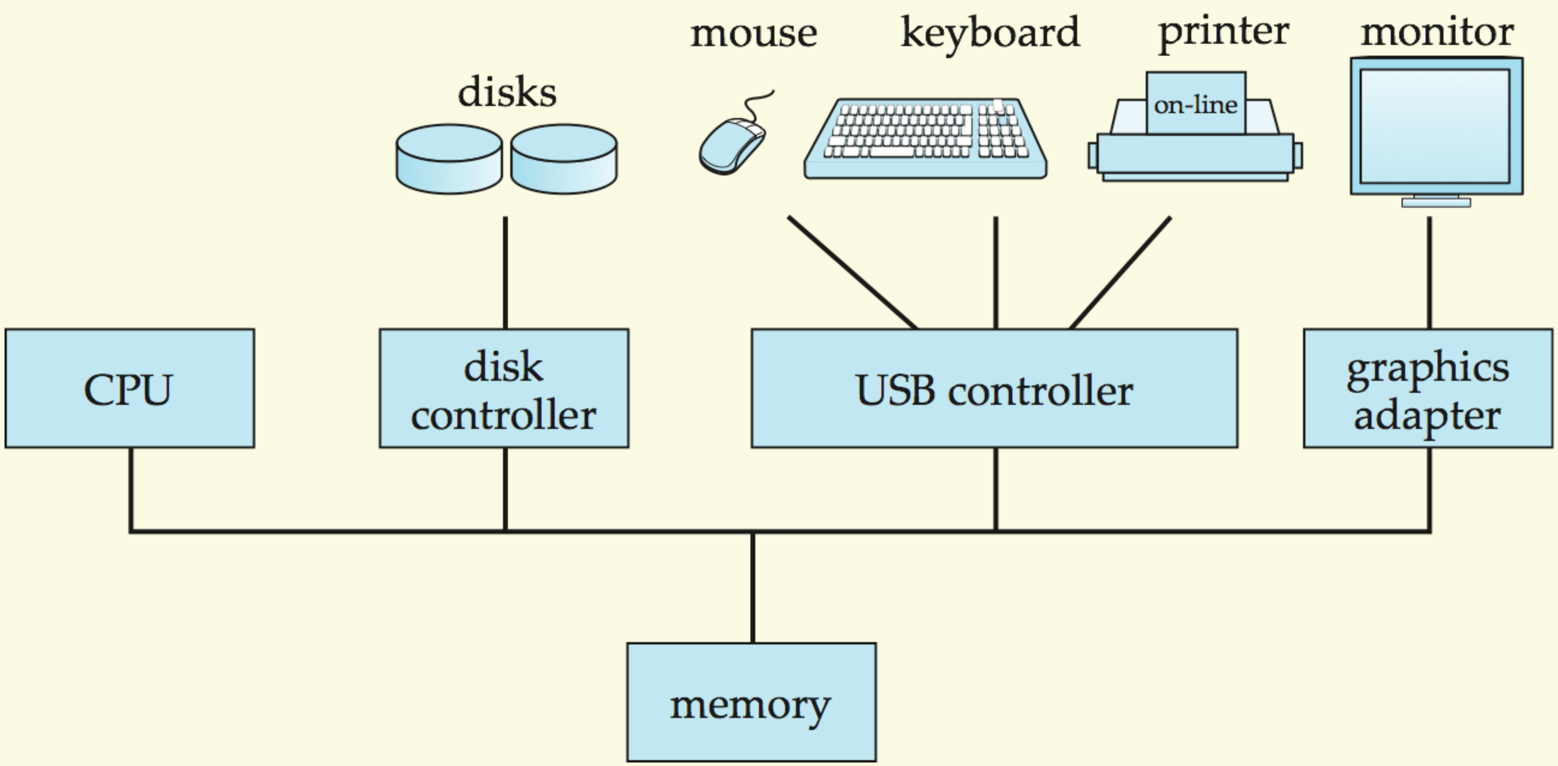
DBMS architectures

- Centralized and Client-Server Systems
- Server System Architectures
- Parallel Systems
- Distributed Systems
- Network Types

Centralized Systems

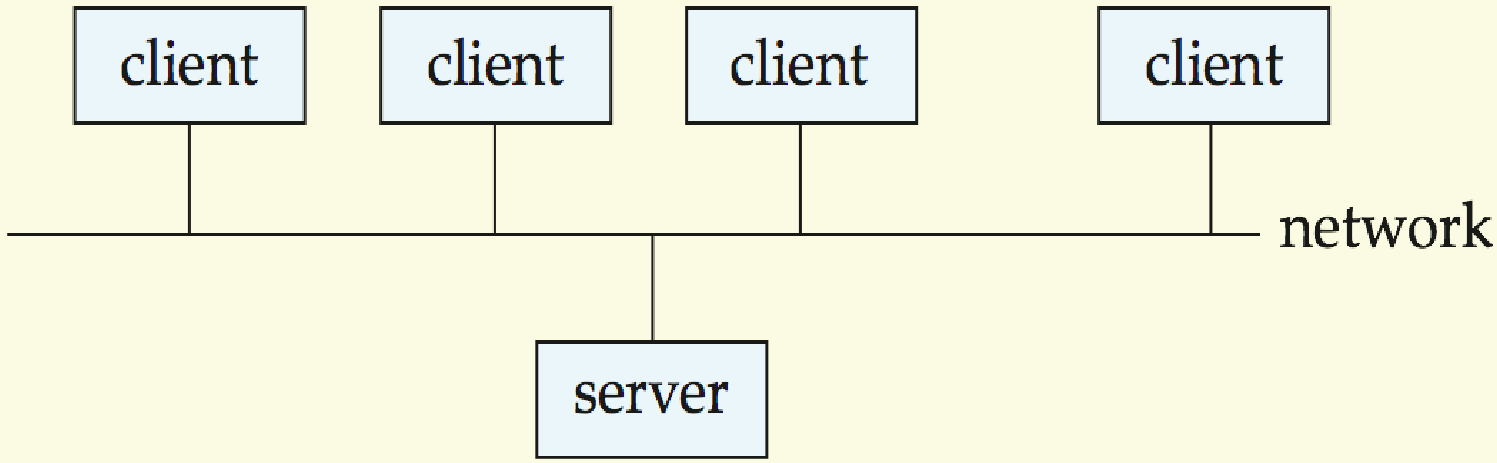
- Run on a single computer system and do not interact with other computer systems.
- Single-user system (e.g., personal computer or workstation): desk-top unit, single user, usually has only one CPU and one or two hard disks; the OS may support only one user.
- Multi-user system: more disks, more memory, multiple CPUs, and a multi-user OS. Serve a large number of users who are connected to the system vie terminals. Often called *server* systems.

A Centralized Computer System



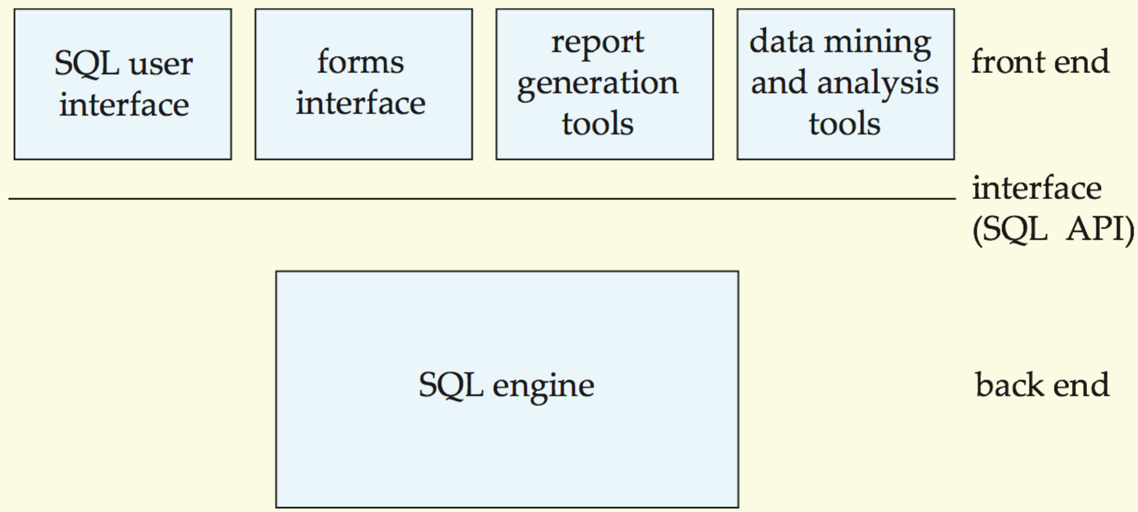
Client-Server Systems

- Server systems satisfy requests generated at *m* client systems



Client-Server Systems

- Database functionality can be divided into:
  - **Back-end**: manages access structures, query evaluation and optimization, concurrency control and recovery.
  - **Front-end**: consists of tools such as *forms*, *report-writers*, and graphical user interface facilities.
- The interface between the front-end and the back-end is through SQL or through an application program interface.



Client-Server Systems (Cont.)

- Advantages of replacing mainframes with networks of workstations or PCs connected to back-end server machines:
  - better functionality for the cost
  - flexibility in locating resources and expanding facilities
  - better user interfaces
  - easier maintenance

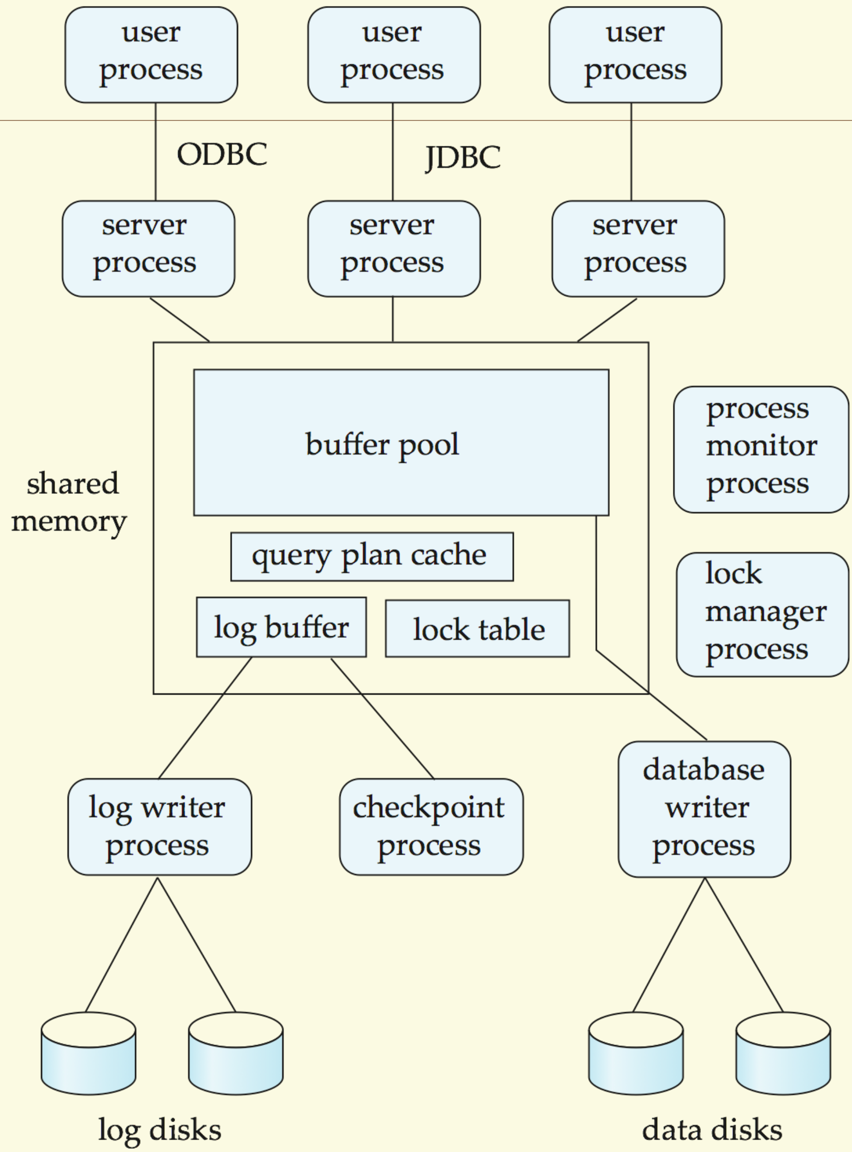
Server System Architecture

- ✓ Server systems can be broadly categorized into :
  - **transaction servers** which are widely used in relational database systems, and
  - **data servers**, used in object-oriented database systems

Transaction Servers

- Also called **query server** systems
  - Clients send requests to the server
  - Transactions are executed at the server
  - Results are shipped back to the client.
- Requests are specified in e.g., SQL, and communicated to the server through a *remote call* mechanism..
- *Open Database Connectivity* (ODBC) is a C language application program interface standard from Microsoft for connecting to a server, sending SQL requests, and receiving results.
- JDBC standard is similar to ODBC, for Java

Transaction System Processes (Cont.)



Data Servers

- Used in high-speed LANs, in cases where
  - The clients are comparable in processing power to the server
  - The tasks to be executed are compute intensive.
- Data are shipped to clients where processing is performed, and then shipped results back to the server.
- This architecture requires full back-end functionality at the clients.
- Used in many object-oriented database systems

Parallel Systems

- Parallel database systems consist of multiple processors and multiple disks connected by a fast interconnection network.
- A **coarse-grain parallel** machine consists of a small number of powerful processors
- A **massively parallel** or **fine grain parallel** machine utilizes thousands of smaller processors.
- Two main performance measures:
  - **throughput** --- the number of tasks that can be completed in a given time interval
  - **response time** --- the amount of time it takes to complete a single task from the time it is submitted

Speed-Up and Scale-Up

- **Speedup**: a fixed-sized problem executing on a small system is given to a system which is *N*-times larger.
  - Measured by:

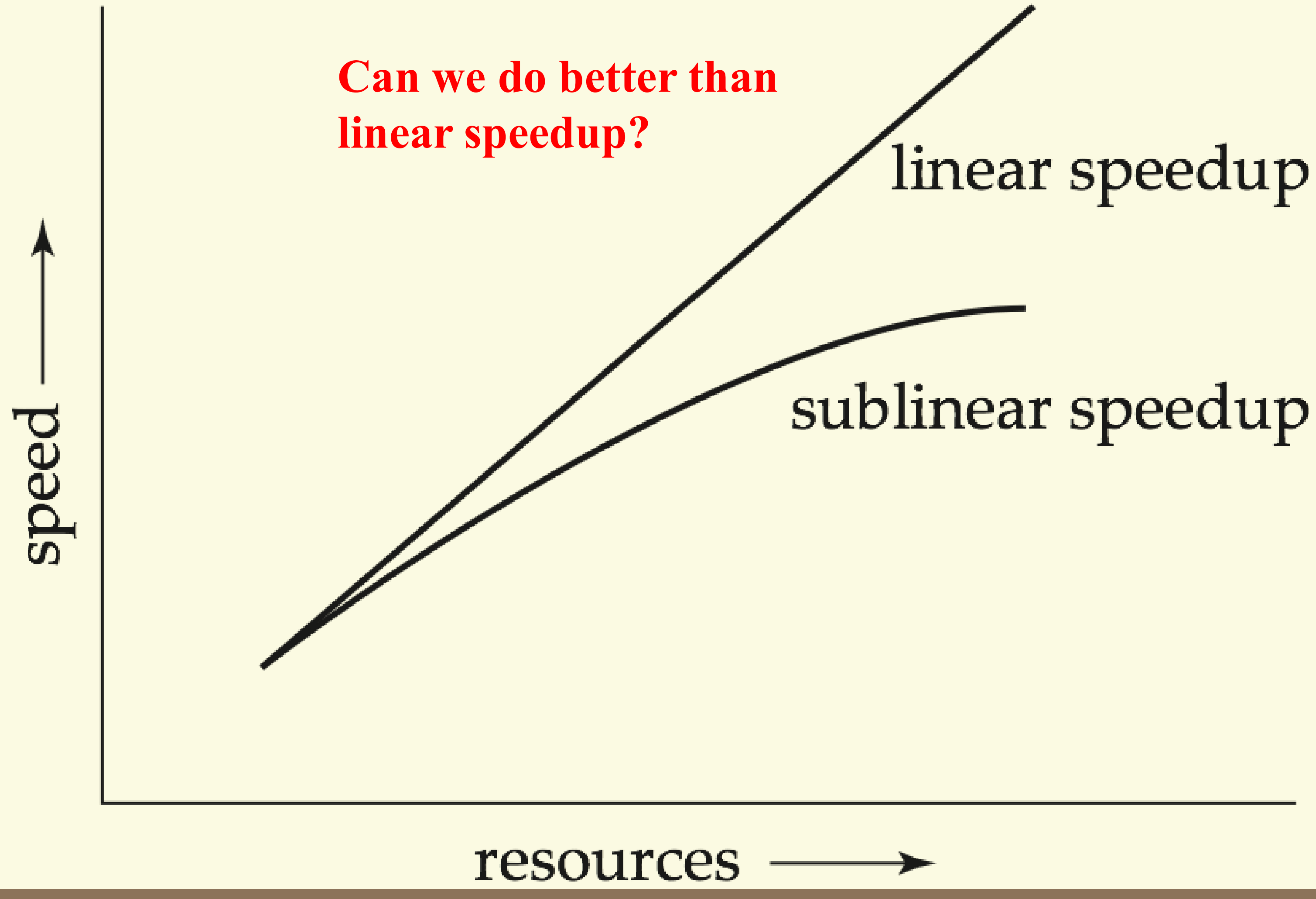
$$\text{speedup} = \frac{\text{small system elapsed time}}{\text{large system elapsed time}}$$

  - Speedup is **linear** if equation equals N.
- **Scaleup**: increase the size of both the problem and the system
  - *N*-times larger system used to perform *N*-times larger job
  - Measured by:

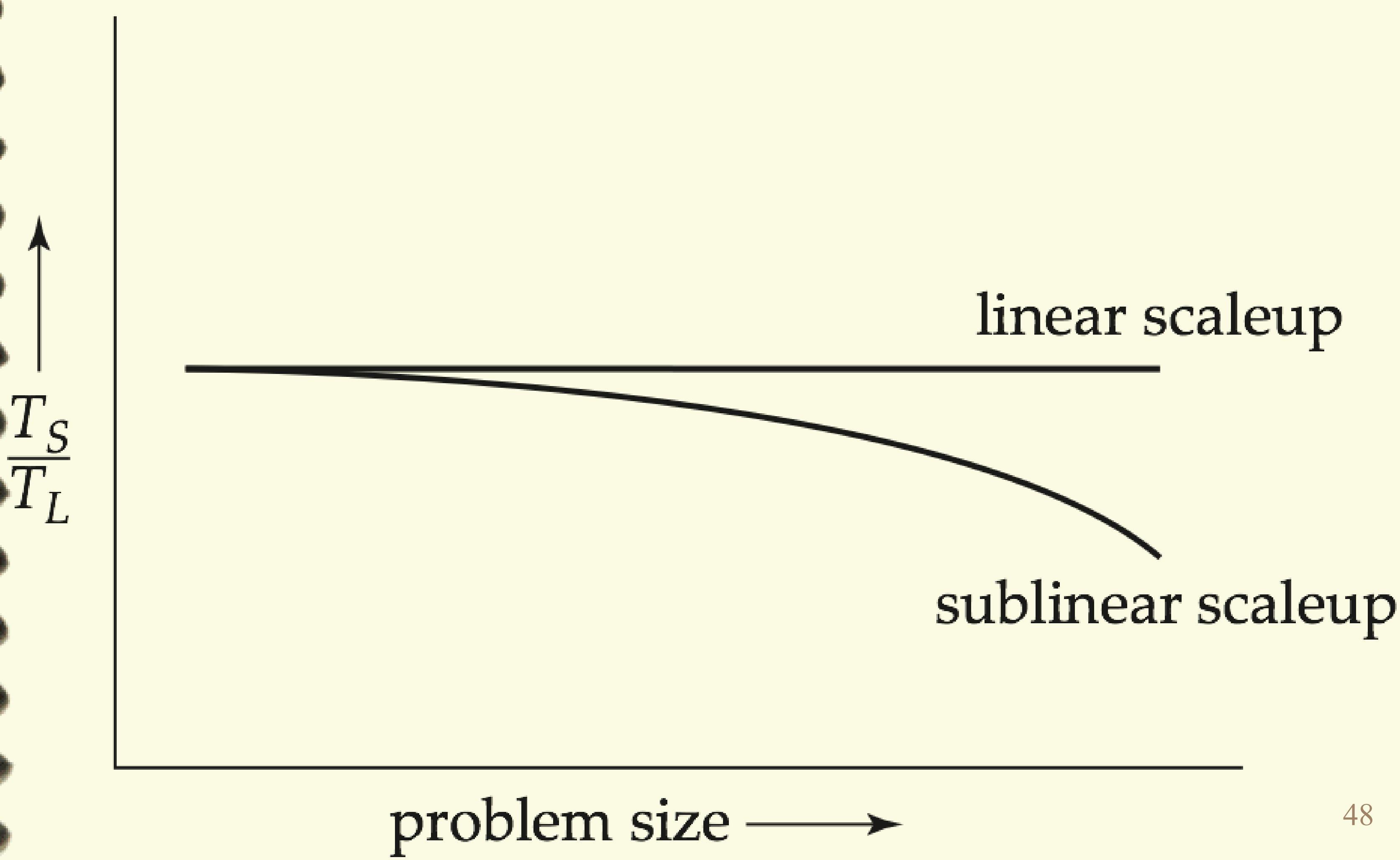
$$\text{scaleup} = \frac{\text{small system small problem elapsed time}}{\text{big system big problem elapsed time}}$$

  - Scale up is **linear** if equation equals 1.

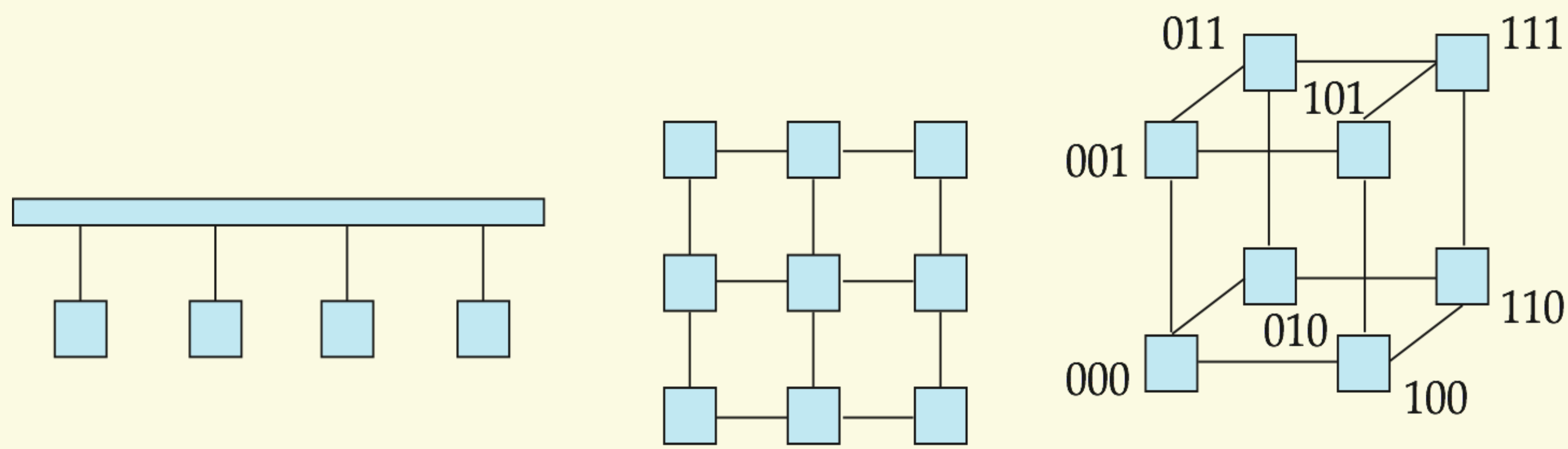
Speedup


Can we do better than linear speedup?
linear speedup
sublinear speedup
speed
resources

Scaleup



Interconnection Architectures



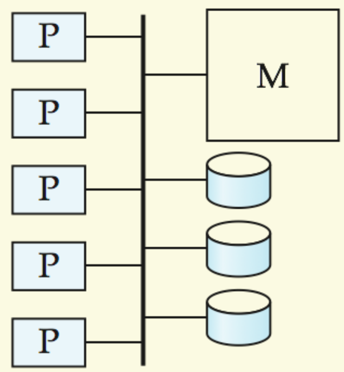
(a) bus (b) mesh (c) hypercube

send data on and receive data from a single communication bus; Does not scale well with increasing parallelism.

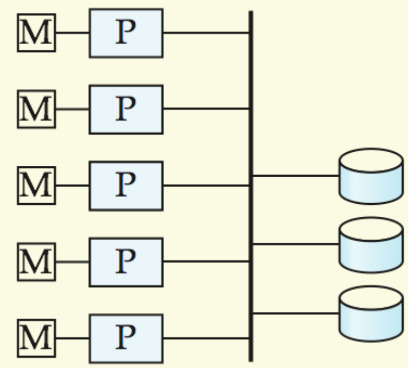
each connected to all adjacent components; scales better. But may require $2\sqrt{n}$ hops to send message to a node

numbered in binary; connected to one another if binary differ in exactly 1bit; *n components* connected to *log(n)* other components. can reach each other via at most *log(n)* links

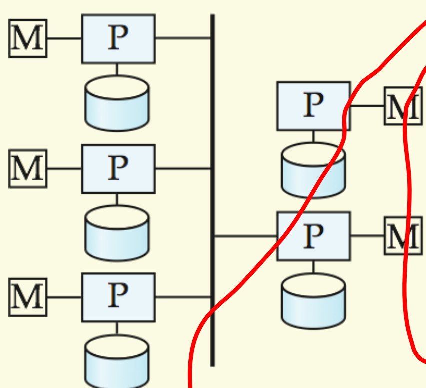
Parallel Database Architectures



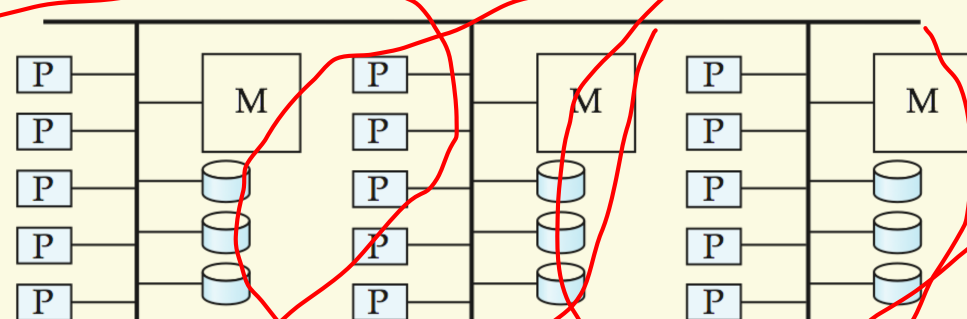
(a) shared memory



(b) shared disk



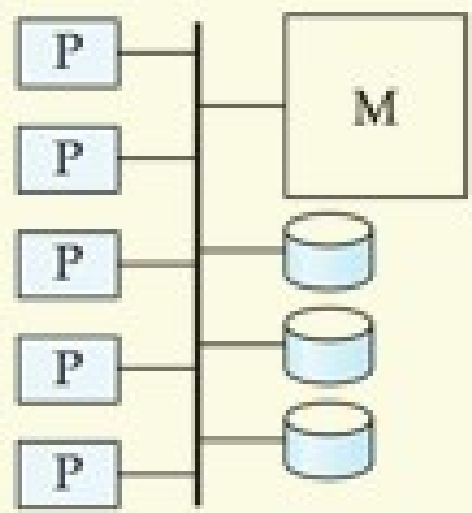
(c) shared nothing



(d) hierarchical

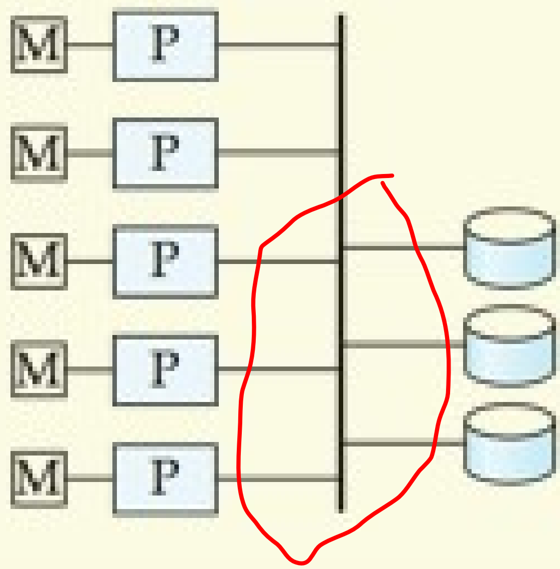
Shared Memory

- Processors and disks have access to a common memory, typically via a bus or through an interconnection network.
- Extremely efficient communication between processors — data in shared memory can be accessed by any processor without having to move it using software.
- Downside – architecture is not scalable beyond 32 or 64 processors since the bus or the interconnection network becomes a bottleneck
- Widely used for lower degrees of parallelism (4 to 8).



Shared Disk

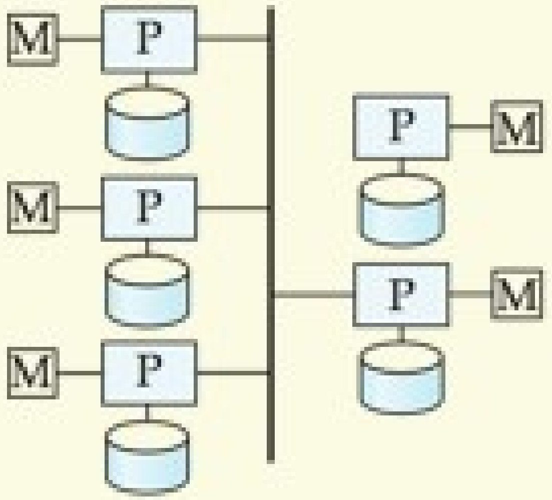
- All processors can directly access all disks via an interconnection network, but the processors have private memories.
  - The memory bus is not a bottleneck
  - Architecture provides a degree of **fault-tolerance**
- Downside: bottleneck now occurs at interconnection to the disk subsystem.
- Shared-disk systems can scale to a somewhat larger number of processors, but communication between processors is slower.



IBM Sysplex and DEC clusters (now part of Compaq) running Rdb (now Oracle Rdb

Shared Nothing

- ✓ Node consists of a processor, memory, and one or more disks. Processors at one node communicate with another processor at another node using an interconnection network.
- ✓ minimizing the interference of resource sharing.
- ✓ Shared-nothing multiprocessors can be scaled up to thousands of processors without interference.
- ✓ Main drawback: cost of communication and non-local disk access; sending data involves software interaction at both ends.



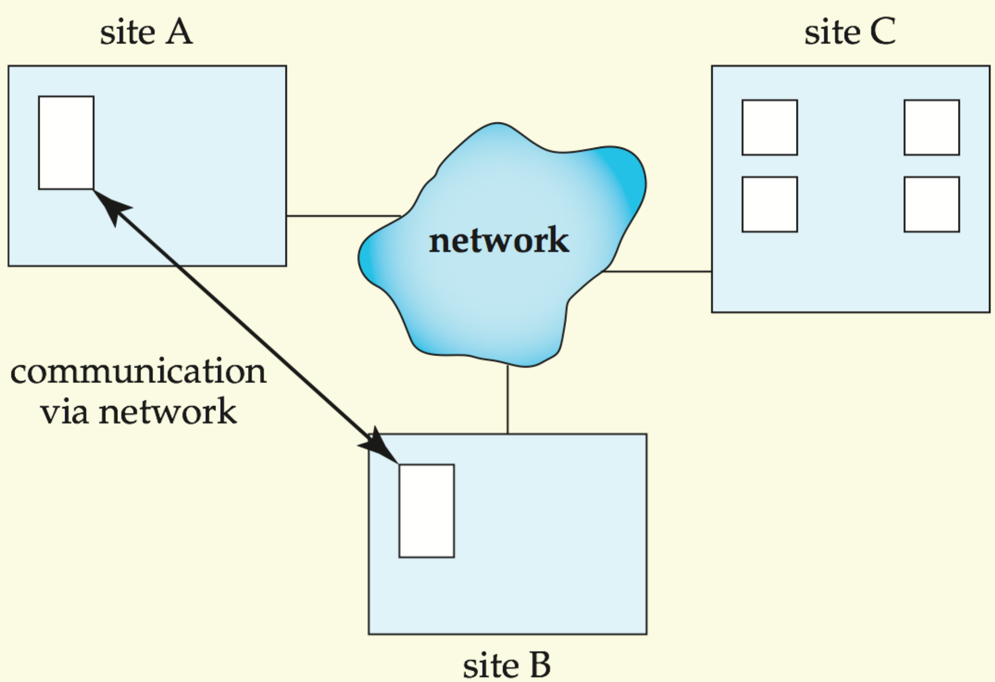
Teradata, Tandem, Oracle-n CUBE

Hierarchical

- ✓ Combines characteristics of shared-memory, shared-disk, and shared-nothing architectures.
- ✓ Top level is a shared-nothing architecture – nodes connected by an interconnection network, and do not share disks or memory with each other.
- ✓ Each node of the system could be a shared-memory system with a few processors.
- ✓ Alternatively, each node could be a shared-disk system, and each of the systems sharing a set of disks could be a shared-memory system.
- ✓ Reduce the complexity of programming such systems by **distributed virtual-memory** architectures
  - Also called **non-uniform memory architecture (NUMA)**

Distributed Systems

- Data spread over multiple machines (also referred to as **sites** or **nodes**).
- Network interconnects the machines
- Data shared by users on multiple machines



Distributed Databases

- Homogeneous distributed databases
  - Same software/schema on all sites, data may be partitioned among sites
  - Goal: provide a view of a single database, hiding details of distribution
- Heterogeneous distributed databases
  - Different software/schema on different sites
  - Goal: integrate existing databases to provide useful functionality
- Differentiate between *local* and *global* transactions
  - A **local transaction** accesses data in the *single* site at which the transaction was initiated.
  - A **global transaction** either accesses data in a site different from the one at which the transaction was initiated or accesses data in several different sites.

Trade-offs in Distributed Systems

- ✓ Sharing data – users at one site able to access the data residing at some other sites.
- ✓ Autonomy – each site is able to retain a degree of control over data stored locally.
- ✓ Higher system availability through redundancy — data can be replicated at remote sites, and system can function even if a site fails.
- ✓ Disadvantage: added complexity required to ensure proper coordination among sites.
  - Software development cost.
  - Greater potential for bugs.
  - Increased processing overhead.

Summary

- ✓ RDBMS design process
  - Normalization
  - the normal forms 1NF, 2NF, 3NF, BCNF, and 4NF
  - normal forms transformation
- ✓ DBMS architectures