Chapter 9: Database Systems

Computer Science: An Overview
Tenth Edition

by
J. Glenn Brookshear

Presentation files modified by Farn Wang

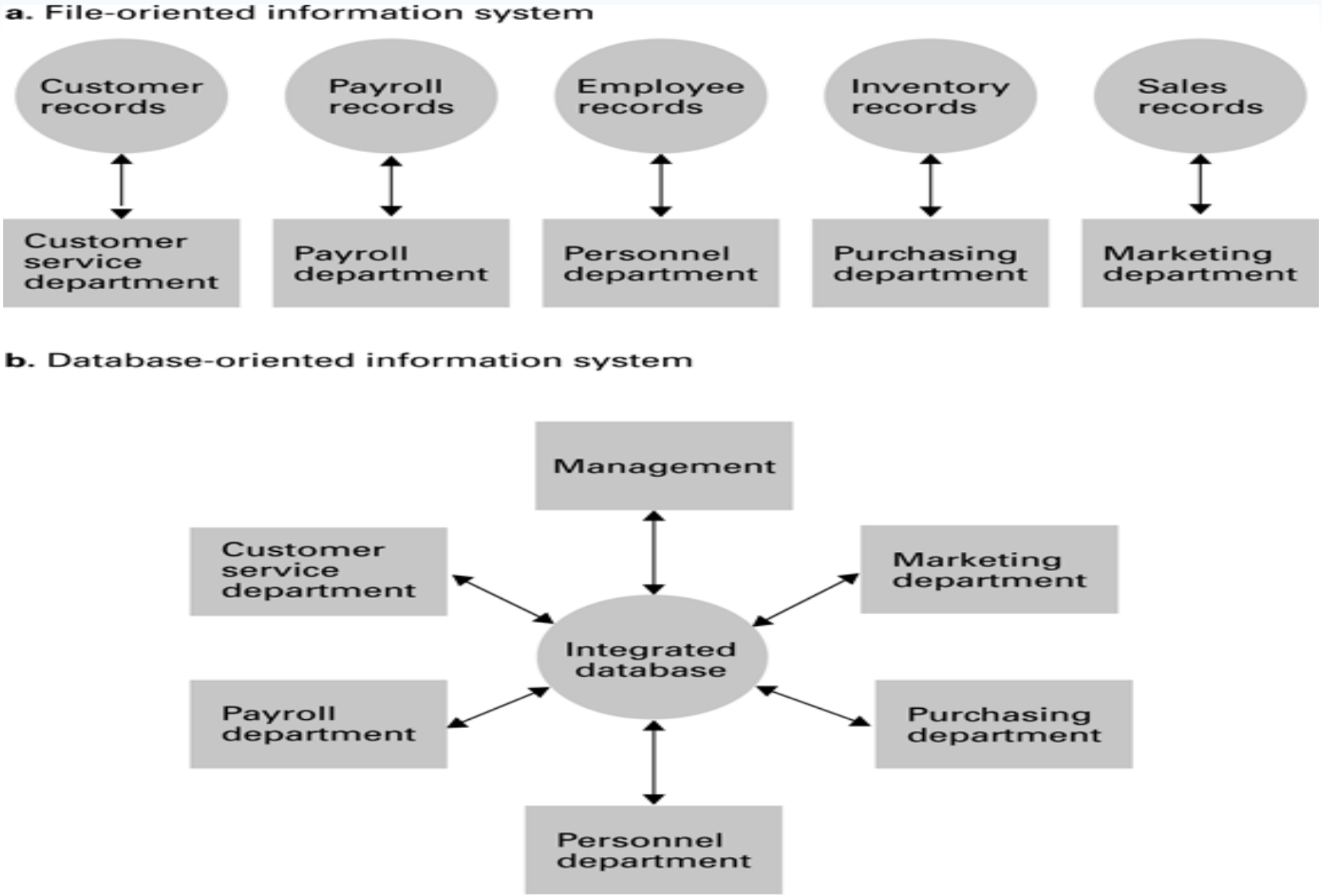
Chapter 9: Database Systems

- 9.1 Database Fundamentals
- 9.2 The Relational Model
- 9.3 Object-Oriented Databases
- 9.4 Maintaining Database Integrity
- 9.5 Traditional File Structures
- 9.6 Data Mining
- 9.7 Social Impact of Database Technology

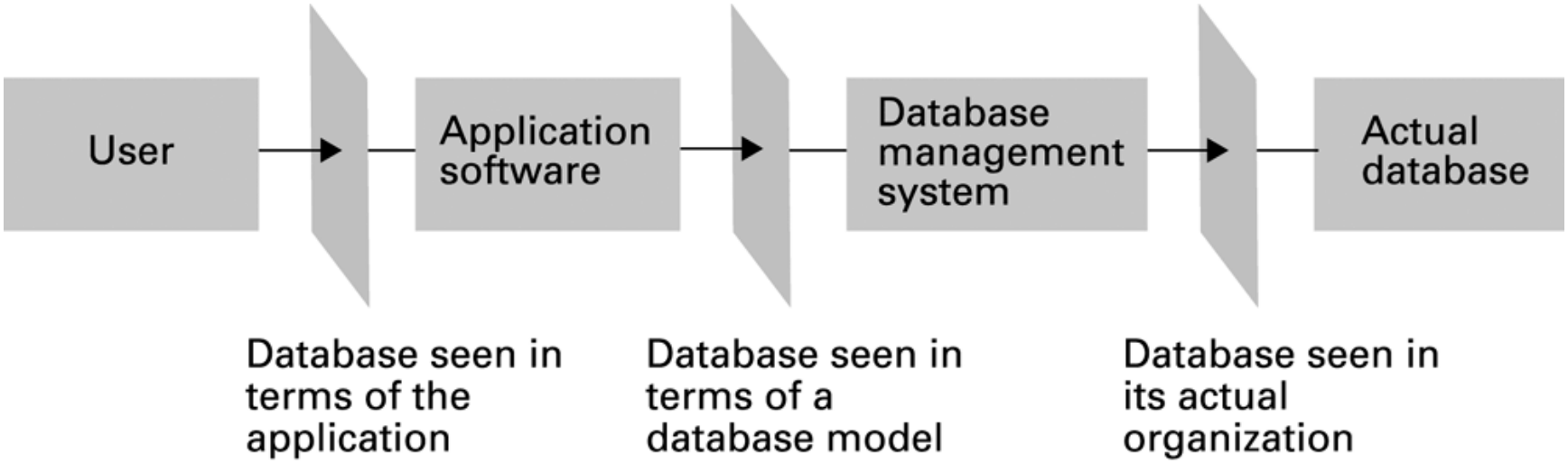
Database

A collection of data that is multidimensional in the sense that internal links between its entries make the information accessible from a variety of perspectives

A file versus a database organization



The conceptual layers of a database implementation



Schemas 馬詠治
- descriptions of database structure

- **Schema:**
  - of an entire database,
  - used by database software to maintain the database
- **Subschema:**
  - of only that portion of the database pertinent to a particular user's needs,
  - used to prevent sensitive data from being accessed by unauthorized personnel

Database Management Systems 張瑞

- **Database Management System** (DBMS)
  - A software layer that manipulates a database in response to requests from applications
- **Distributed Database**
  - A database stored on multiple machines
  - DBMS will mask this organizational detail from its users
- **Data independence**
  - The ability to change the organization of a database without changing the application software that uses it

Database Models 陳楷訓

- A conceptual view of a database
- Two popular models
  - Relational database model
  - Object-oriented database model

Relational Database Model 吳宇

Relation, in mathematics,

- an n-ary relation is a set of n-dimensional tuples (vectors)

Example: relations of

- parent: {(**陳幸妤**,**陳水扁**),(**馬唯中**,**周美青**),…}
- **系友**: {(**吳瑞北**,EE,NTU),(**周美青**,**法律**,NCCU),…}
- **顧客**:{(**陳水扁**,Sogo,coat),(**周美青**,**都蘭**,bag),…}
- **直屬單位**:{(**中央研究院**,**總統府**),(**教育部**,**行政院**),…}

Relational Database Model 張致綱

Relation, in practice,

- an *n*-ary relation is a rectangular table with *n* columns
- **Attribute, property:** A column in the table
- **Tuple, record:** A row in the table

A relation containing employee information 古行涵

Empl Id	Name	Address	SSN
25X15	Joe E. Baker	33 Nowhere St.	111223333
34Y70	Cheryl H. Clark	563 Downtown Ave.	999009999
23Y34	G. Jerry Smith	1555 Circle Dr.	111005555
•	•	•	•
•	•	•	•
•	•	•	•

Relational Design 楊其昇

- Avoid multiple concepts within one relation
  - Can lead to redundant data
  - Deleting a tuple could also delete necessary but unrelated information

Improving a Relational Design

- **Decomposition:** Dividing the columns of a relation into two or more relations, duplicating those columns necessary to maintain relationships
  - **Lossless** or **nonloss** decomposition: A "correct" decomposition that does not lose any information

A relation containing redundancy

personal data | job data | assignment

Empl Id	Name	Address	SSN	Job Id	Job Title	Skill Code	Dept	Start Date	Term Date
25X15	Joe E. Baker	33 Nowhere St.	111223333	F5	Floor manager	FM3	Sales	9-1-2007	9-30-2008
25X15	Joe E. Baker	33 Nowhere St.	111223333	D7	Dept. head	K2	Sales	10-1-2008	*
34Y70	Cheryl H. Clark	563 Downtown Ave.	999009999	F5	Floor manager	FM3	Sales	10-1-2007	*
23Y34	G. Jerry Smith	1555 Circle Dr.	111005555	S25X	Secretary	T5	Personnel	3-1-1999	4-30-2006
23Y34	G. Jerry Smith	1555 Circle Dr.	111005555	S26Z	Secretary	T6	Accounting	5-1-2006	*
.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.

An employee database consisting of three relations

EMPLOYEE relation

Empl Id	Name	Address	SSN
25X15	Joe E. Baker	33 Nowhere St.	111223333
34Y70	Cheryl H. Clark	563 Downtown Ave.	999009999
23Y34	G. Jerry Smith	1555 Circle Dr.	111005555

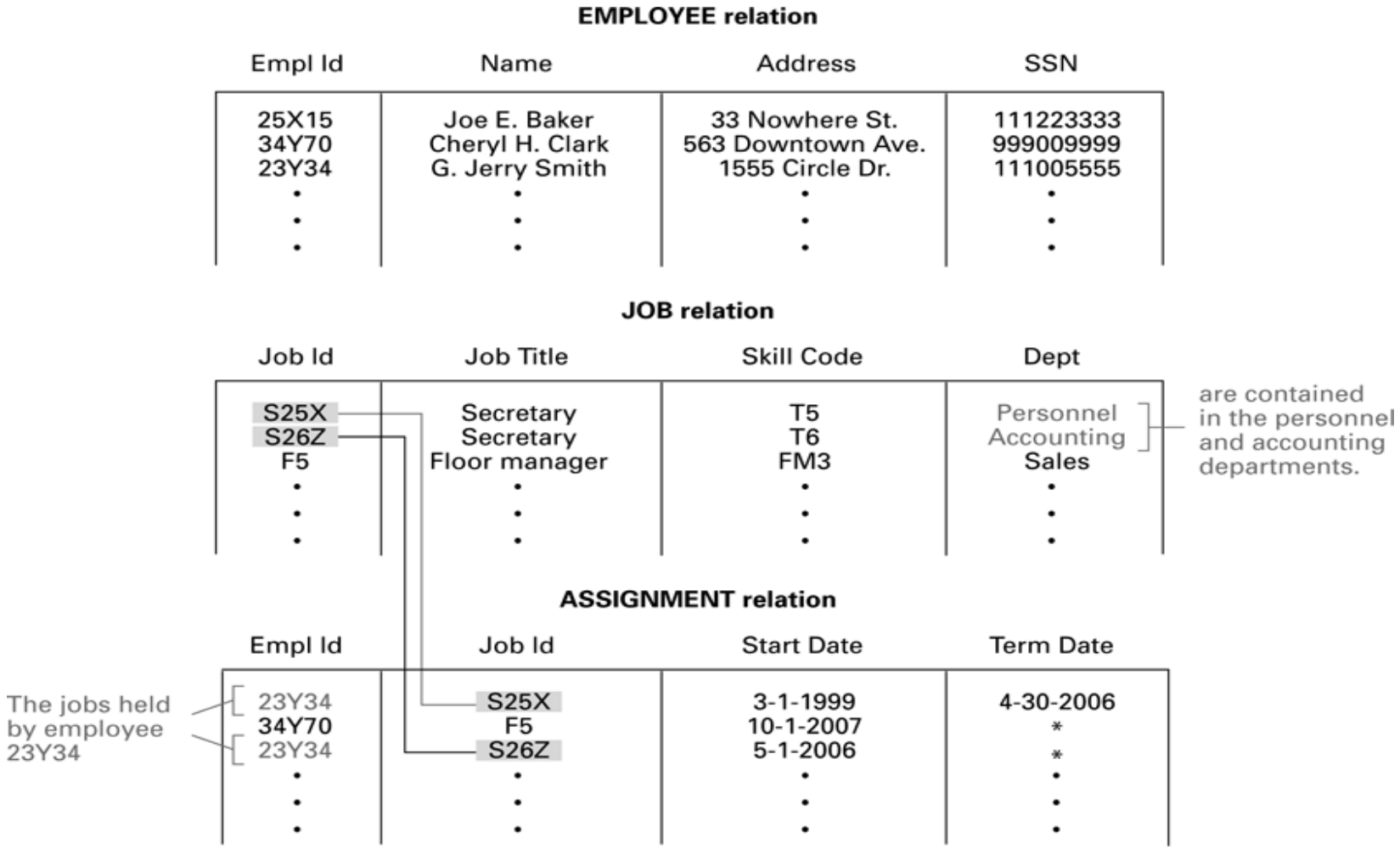
JOB relation

Job Id	JobTitle	Skill Code	Dept
S25X	Secretary	T5	Personnel
S26Z	Secretary	T6	Accounting
F5	Floor manager	FM3	Sales
•	•	•	•
•	•	•	•
•	•	•	•

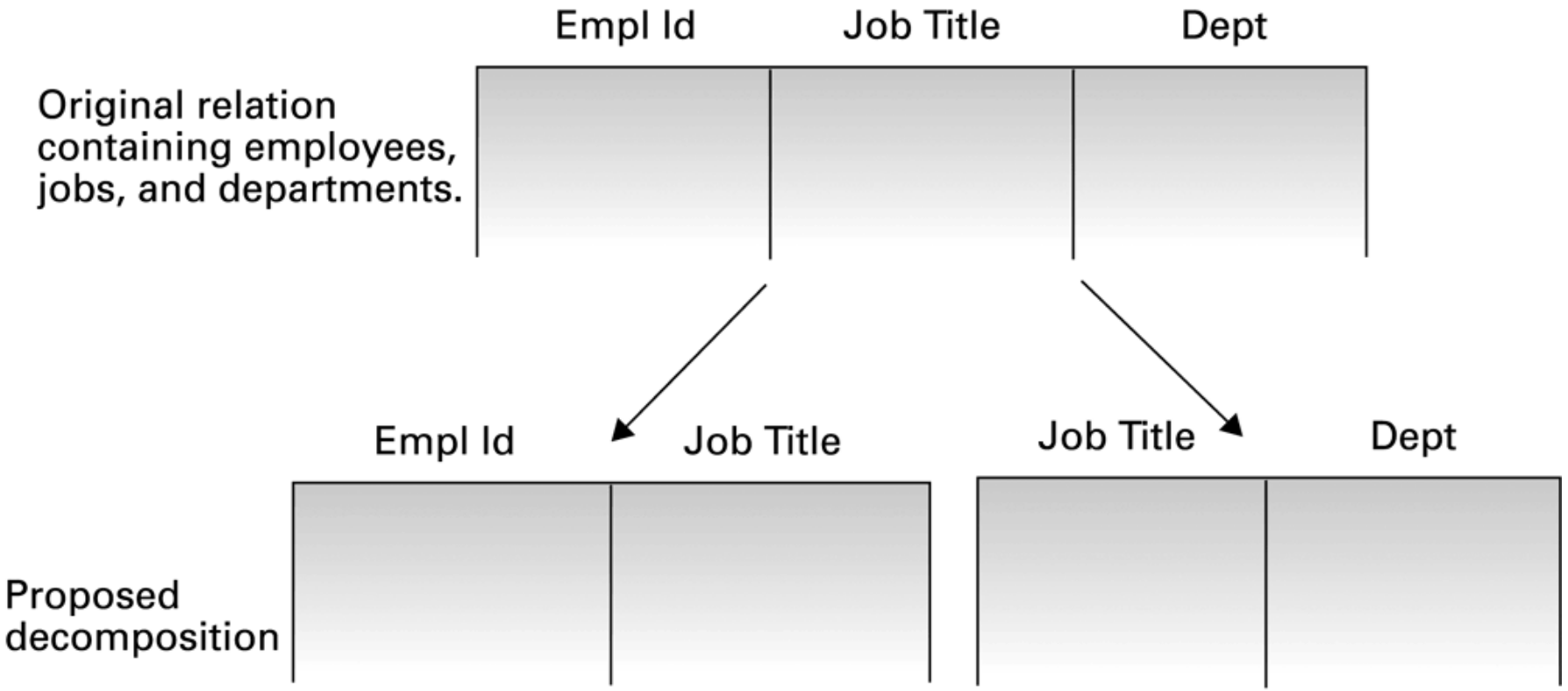
ASSIGNMENT relation

Empl Id	Job Id	Start Date	Term Date
23Y34	S25X	3-1-1999	4-30-2006
34Y70	F5	10-1-2007	*
23Y34	S26Z	5-1-2006	*
•	•	•	•
•	•	•	•
•	•	•	•

Finding the departments in which employee 23Y34 has worked



A relation and a proposed decomposition



2014/05/20 stopped here.

Relational Operations

Construct relations from relations

- **Select:** R α R
  - Pick a subset of a relation
  - Choose rows from a relations
- **Project:** R α R
  - Project to dimension
  - Choose columns,
- **Join:** R×R×… ×R α R
  - Product of relations
  - Assemble information from two or more relations

The SELECT operation (楊貽潯)

EMPLOYEE relation

Empl Id	Name	Address	SSN
25X15	Joe E. Baker	33 Nowhere St.	111223333
34Y70	Cheryl H. Clark	563 Downtown Ave.	999009999
23Y34	G. Jerry Smith	1555 Circle Dr.	111005555
•	•	•	•
•	•	•	•
•	•	•	•

NEW ← SELECT from EMPLOYEE where EmpId = "34Y70"

NEW relation

Empl Id	Name	Address	SSN
34Y70	Cheryl H. Clark	563 Downtown Ave.	999009999

The PROJECT operation （黃宇平）

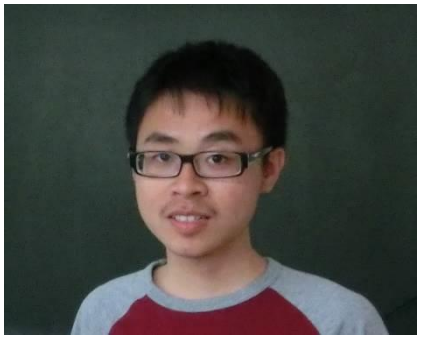
EMPLOYEE relation

Empl Id	Name	Address	SSN
25X15	Joe E. Baker	33 Nowhere St.	111223333
24Y70	Cheryl H. Clark	563 Downtown Ave.	999009999
23Y34	G. Jerry Smith	1555 Circle Dr.	111005555
•	•	•	•
•	•	•	•
•	•	•	•

MAIL ← PROJECT Name, Address from EMPLOYEE

MAIL relation

Name	Address
Joe E. Baker	33 Nowhere St.
Cheryl H. Clark	563 Downtown Ave.
G. Jerry Smith	1555 Circle Dr.
•	•
•	•
•	•



The JOIN operation （鍾勝隆）

Relation A

V	W
r	2
t	4
p	6

Relation B

X	Y	Z
5	g	p
4	d	e
2	m	q
4	t	f

C ← JOIN A and B where A.W = B.X

Relation C

A.V	A.W	B.X	B.Y	B.Z
r	2	2	m	q
t	4	4	d	e
t	4	4	t	f

Another example of the JOIN operation（王士元）

Relation A

V	W
r	2
t	4
p	6

Relation B

X	Y	Z
1	g	p
4	d	e
2	m	q
5	t	f

C ← JOIN A and B where A.W < B.X

Relation C

A.V	A.W	B.X	B.Y	B.Z
r	2	4	d	e
r	2	5	t	f
t	4	5	t	f

An application of the JOIN operation（賴慶旻）

ASSIGNMENT relation

Empl Id	Job Id	Start Date	Term Date
23Y34	S25X	3-1-1999	4-30-2006
34Y70	F5	10-1-2007	*
25X15	S26Z	5-1-2006	*
•	•	•	•
•	•	•	•
•	•	•	•

JOB relation

Job Id	JobTitle	Skill Code	Dept
S25X	Secretary	T5	Personnel
S26Z	Secretary	T6	Accounting
F5	Floor manager	FM3	Sales
•	•	•	•
•	•	•	•
•	•	•	•

NEW1 ← JOIN ASSIGNMENT and JOB where ASSIGNMENT. JobId = JOB.JobId

NEW1 relation

ASSIGNMENT Empl Id	ASSIGNMENT Job Id	ASSIGNMENT StartDate	ASSIGNMENT TermDate	JOB Job Id	JOB JobTitle	JOB SkillCode	JOB Dept
23Y34	S25X	3-1-1999	4-30-2006	S25X	Secretary	T5	Personnel
34Y70	F5	10-1-2007	*	F5	Floor manager	FM3	Sales
25X15	S26Z	5-1-2006	*	S26Z	Secretary	T6	Accounting
•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•

Structured Query Language (SQL)（李孟學）

- Operations to manipulate tuples
  - insert
  - update
  - delete
  - select
- A query language based on FOL (first-order logic)

SQL Examples

An implicit project operation.

An implicit join operation.

```
select EmplId, Dept
  from ASSIGNMENT, JOB
  where ASSIGNMENT.JobId = JOB.JobId
        and ASSIGNMENT.TermData = "*"
```

In mathematics,

$$\{(\text{EmplId}(t_1), \text{Dept}(t_2)) \mid t_1 \in \text{ASSIGNMENT}, t_2 \in \text{JOB}, \text{JobId}(t_1) = \text{JobId}(t_2), \text{TermData}(t_1) = \text{“*”}\}$$

SQL Examples

```
insert into EMPLOYEE
  values ('43212', 'Sue A. Burt',
          '33 Fair St.', '444661111')
```

In mathematics,

```
EMPLOYEE
= EMPLOYEE
∪ {('43212', 'Sue A. Burt',
          '33 Fair St.', '444661111')}
```

SQL Examples (continued)

```
delete from EMPLOYEE
  where Name = 'G. Jerry Smith'
```

In mathematics,

$$EMPLOYEE = \{t \mid t \in EMPLOYEE, Name(t) \neq \text{'G. Jerry Smith'}\}$$

```
select from EMPLOYEE
  where Name != 'G. Jerry Smith'
```

SQL Examples (continued)

```
update EMPLOYEE
  set Address = ‘1812 Napoleon Ave.’
  where Name = ‘Joe E. Baker’
```

In mathematics,

?

Relational database

In retrospection,

- does not quite allow variable-length records
- No heirarchy (class inheritance)
- No protection

Object-oriented (OO) Databases

A database constructed by applying the object-oriented paradigm

- Each entity stored as a persistent object
- Relationships indicated by links between objects
- DBMS maintains inter-object links

OO Databases

2 features

- the integration of data value and operations
- class inheritance

OO Databases

class inheritance

- 古細菌
- 真細菌（原核）
- 真核
  - 生物 class
    - attribute: DNA sequence
    - method: breathe()
    - 動物 class
      - method: move()
      - 哺乳類 class
        - » attribute: hair
        - » methods: 胎生(), 哺乳()
      - 鳥類class
        - » attribute: feather
        - » methods: fly(), 產卵()
    - 植物 class

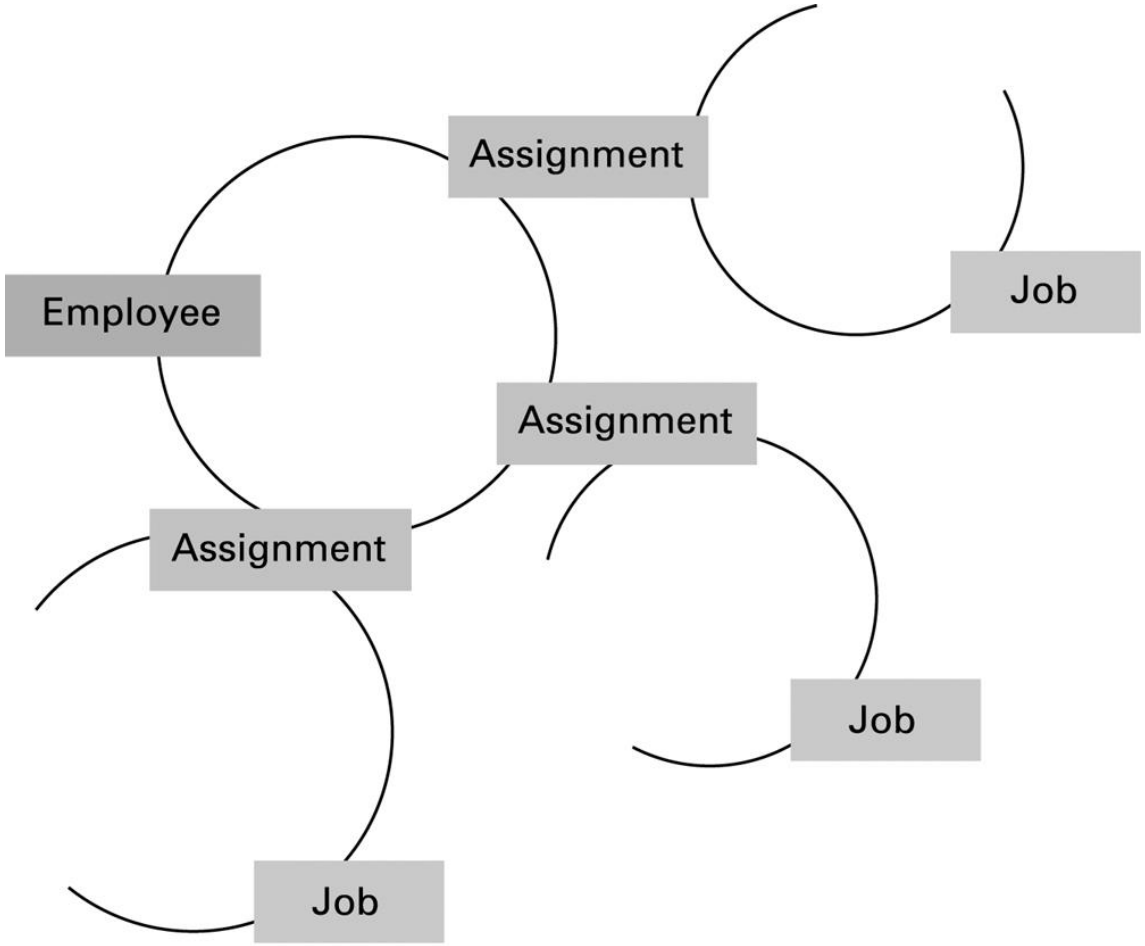
動物 is a subclass of **生物**。
All **動物** have DNA sequence and breathe.

哺乳類 is a subclass of **動物** and **生物**。
All **哺乳類** have DNA sequence, breathe, move, have hair, **胎生**, and **哺乳**.

鳥類 is a subclass of **動物** and **生物**。
All **鳥類** have DNA sequence, breathe, move, has feather, fly, and **產卵**.

植物 is a subclass of **生物**。
All **植物** have DNA sequence and breathe.

The associations between objects in an object-oriented database



Advantages of Object-oriented Databases

- Matches design paradigm of object-oriented applications
- Intelligence can be built into attribute handlers
- Can handle exotic data types
  - Example: multimedia

Maintaining Database Integrity

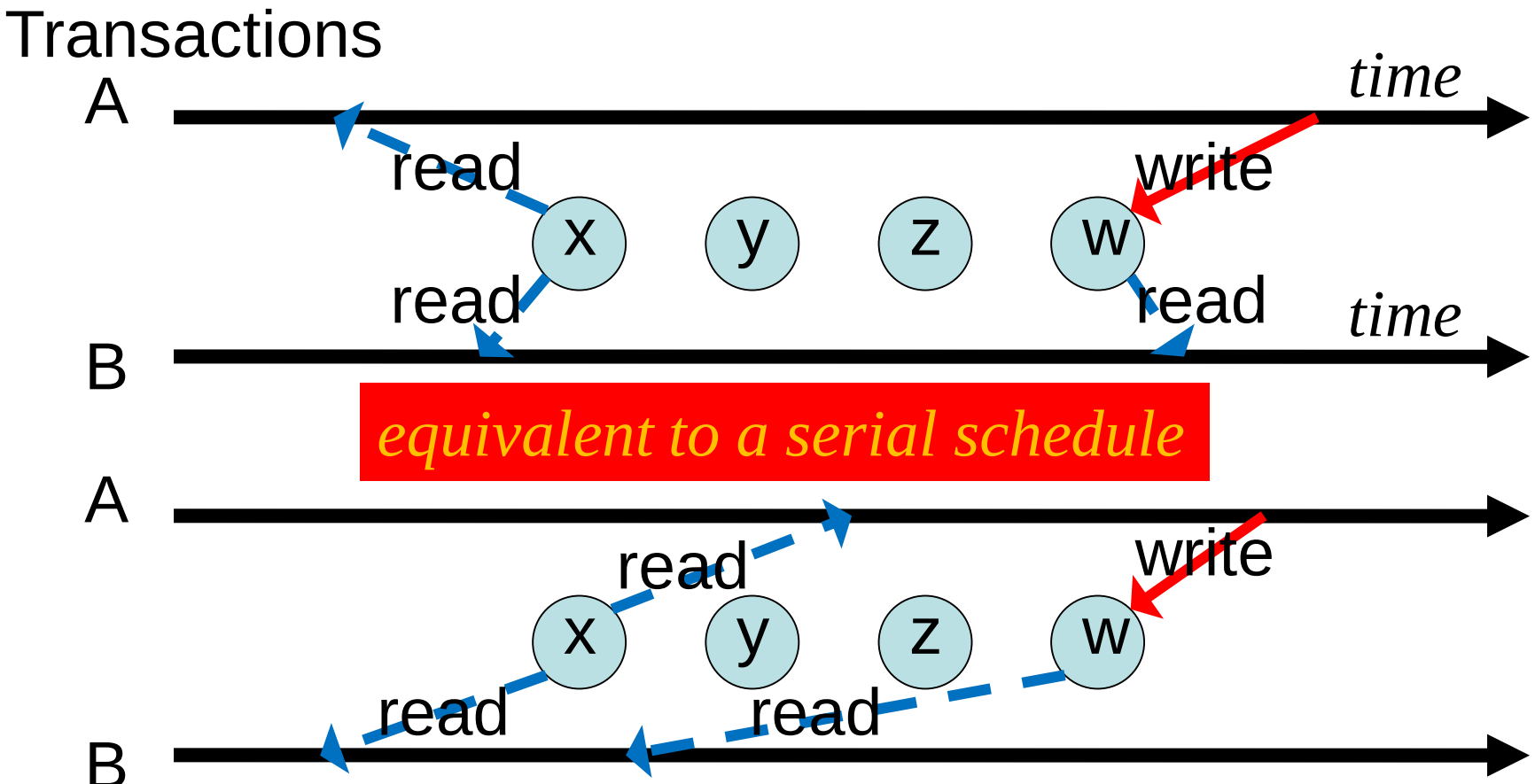
- **Transaction:** A sequence of operations that must all happen together
  - Example: transferring money between bank accounts
- **Transaction log:** A non-volatile record of each transaction's activities, built before the transaction is allowed to execute
  - **Commit point:** The point at which a transaction has been recorded in the log
  - **Roll-back:** The process of undoing a transaction

Maintaining database integrity (continued)

- Simultaneous access problems
  - Incorrect summary problem
  - Lost update problem
  - (the same as the synchronization problem of OS, except that DB transactions are longer and on sets of distributed data.)
- **Locking** = preventing others from accessing data being used by a transaction
  - **Shared** lock: used when reading data
  - **Exclusive** lock: used when altering data

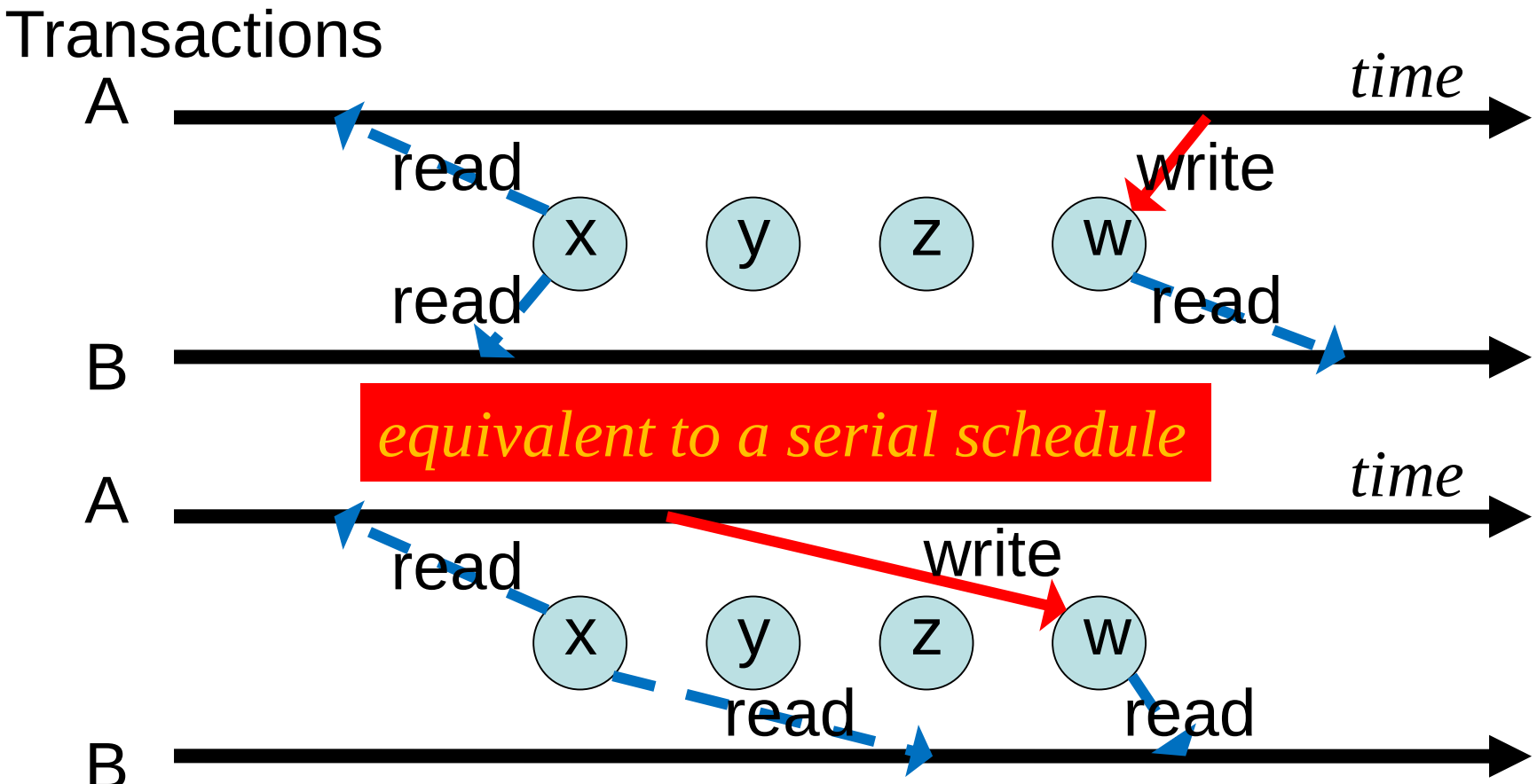
Transaction scheduling - managed by DBMS

A schedule to be admitted.



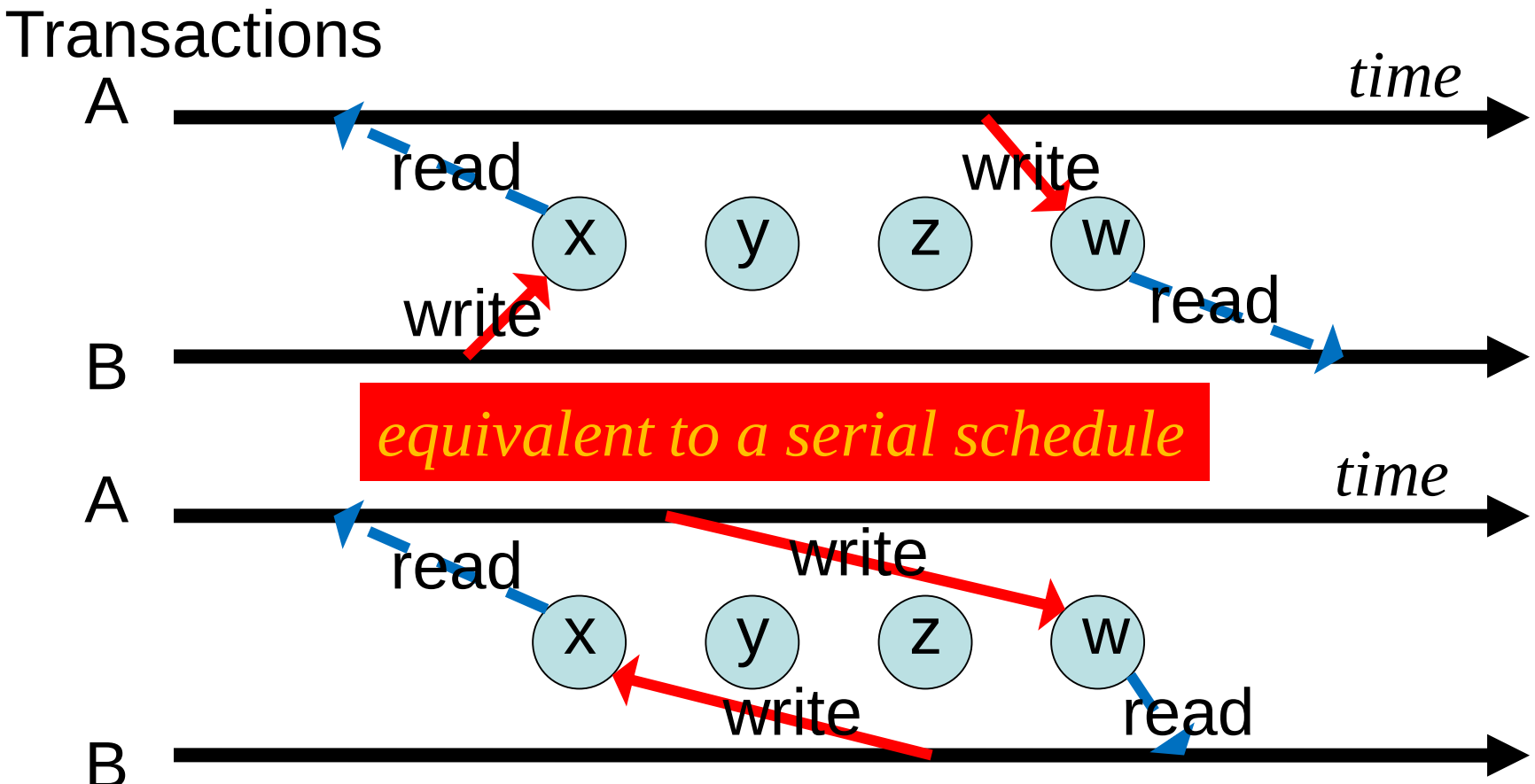
Transaction scheduling - managed by DBMS

A schedule to be admitted.



Transaction scheduling - managed by DBMS

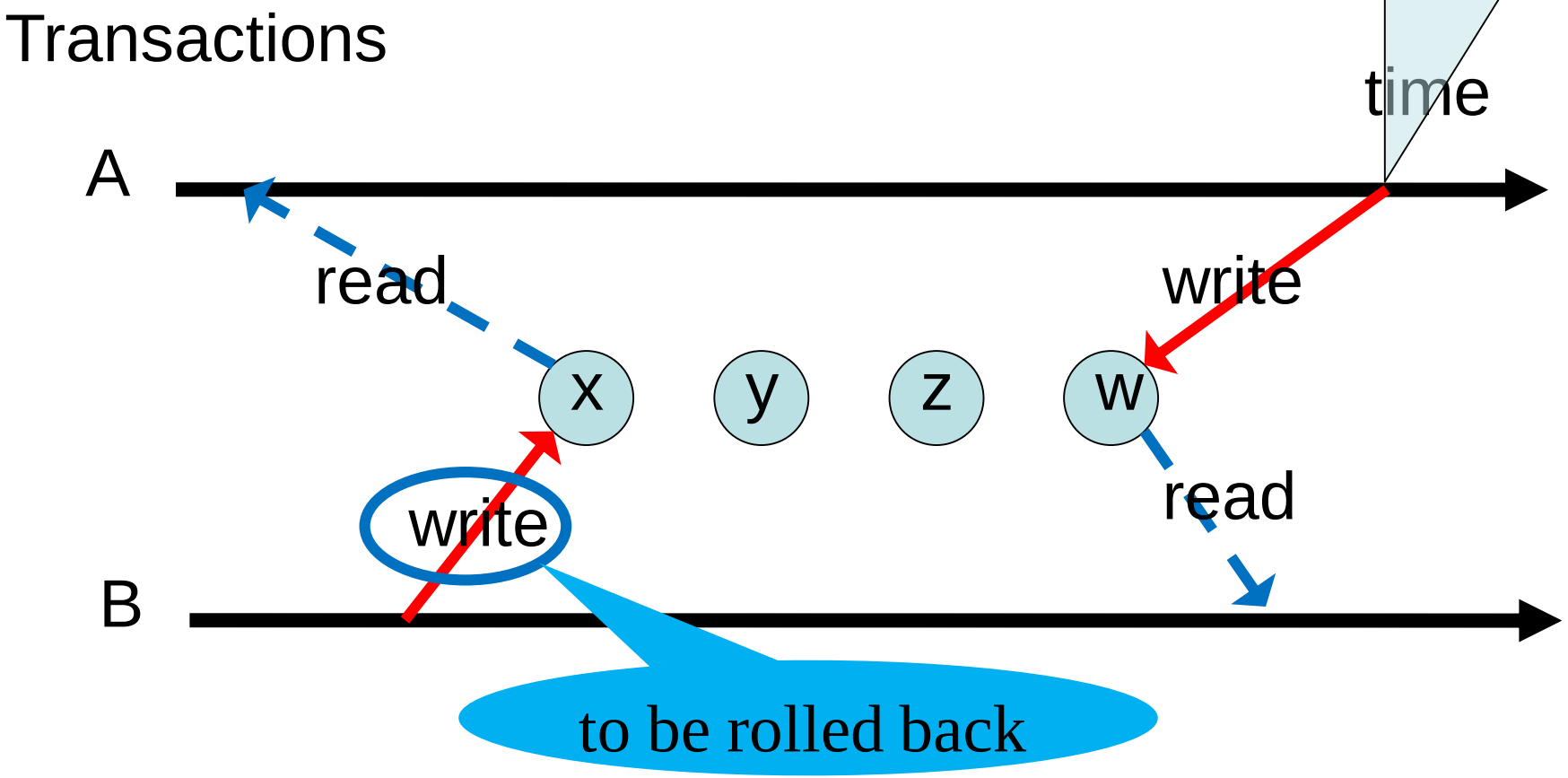
A schedule to be admitted.



Transaction scheduling - managed by DBMS

DBMS finds a read-write conflict, rolls the B back.

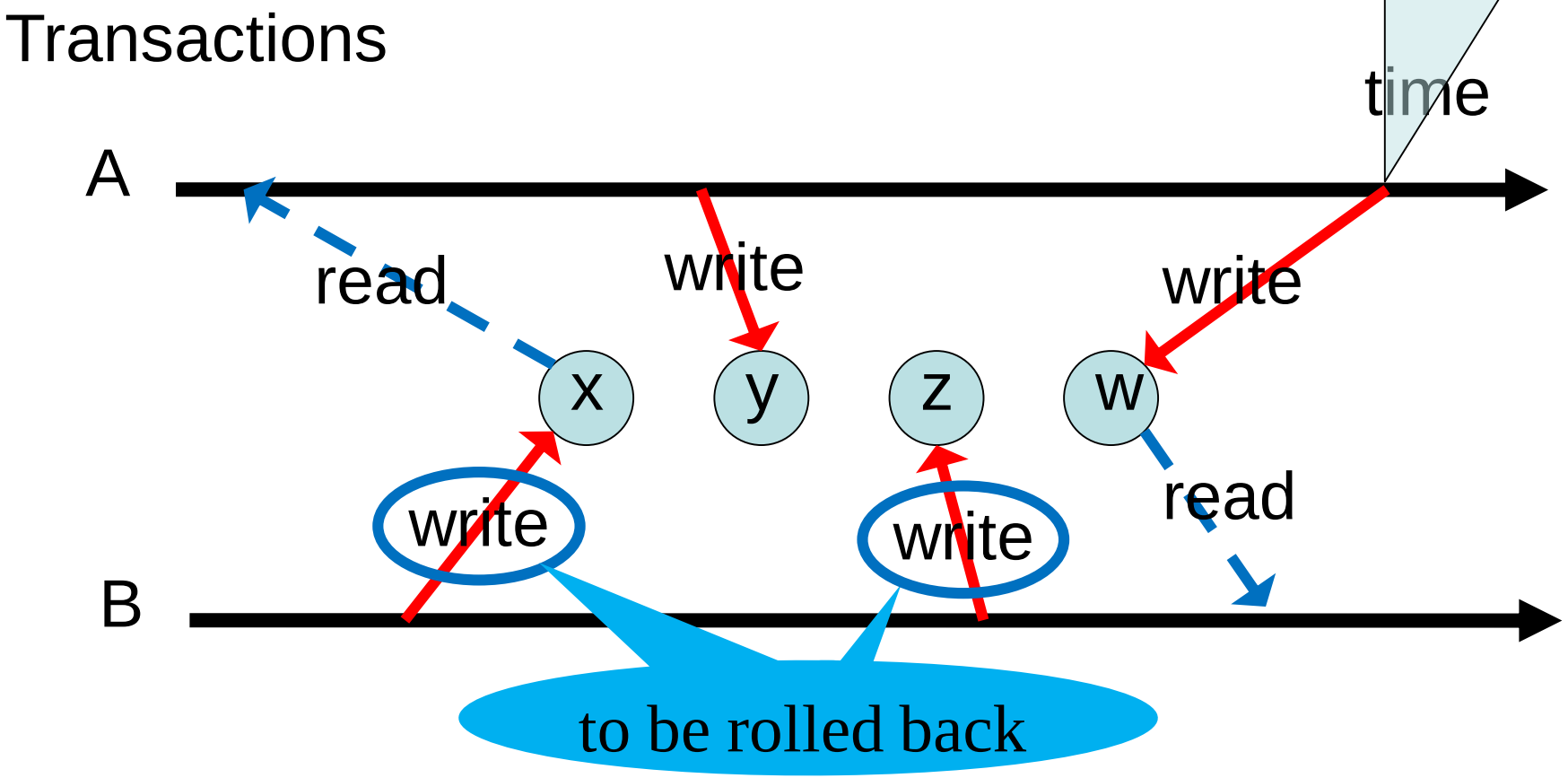
A schedule to be rejected.



Transaction scheduling - managed by DBMS

DBMS finds a read-write conflict, rolls the B back.

A schedule to be rejected.



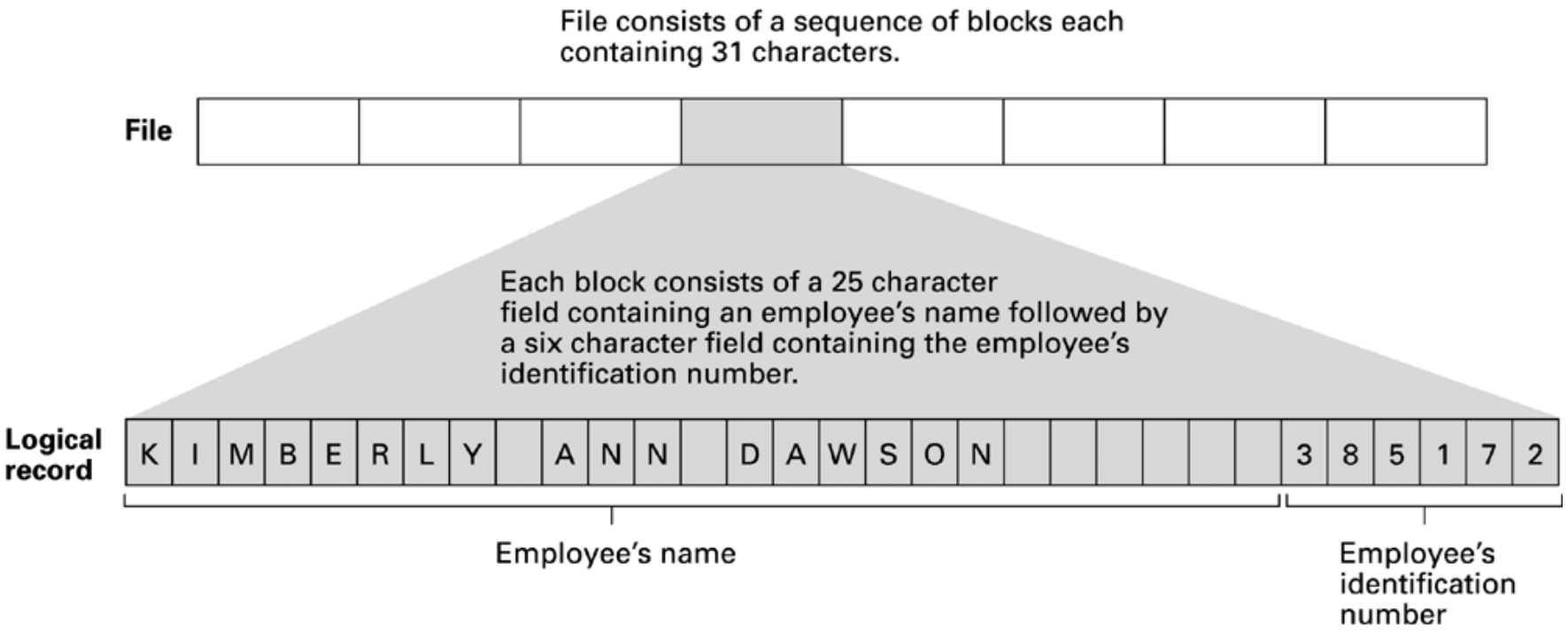
Traditional file structures

- Historical beginning of data storage and retrieval systems
- Important tools in the constructions of massive and complex DB.

Sequential Files

- **Sequential file:** A file whose contents can only be read in order
  - Reader must be able to detect end-of-file (EOF)
  - Data can be stored in logical records, sorted by a key field (master key)
    - *the primary key*
    - Greatly increases the speed of batch updates
    - Can sort the batch operations according to the key ordering.

The structure of a simple employee file implemented as a text file

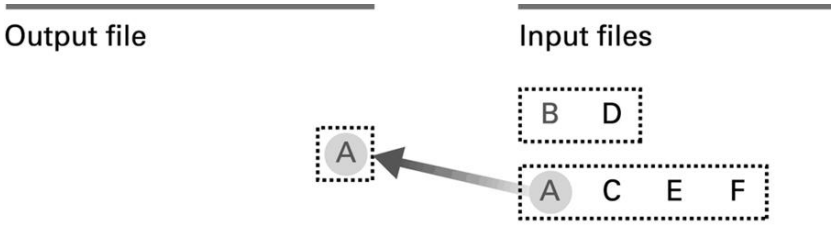


A procedure for merging two sequential files

```
procedure MergeFiles (InputFileA, InputFileB, OutputFile)

if (both input files at EOF) then (Stop, with OutputFile empty)
if (InputFileA not at EOF) then (Declare its first record to be its current record)
if (InputFileB not at EOF) then (Declare its first record to be its current record)
while (neither input file at EOF) do
    (Put the current record with the "smaller" key field value in OutputFile;
     if (that current record is the last record in its corresponding input file)
         then (Declare that input file to be at EOF)
          else (Declare the next record in that input file to be the file's current record)
    )
Starting with the current record in the input file that is not at EOF,
     copy the remaining records to OutputFile.
```

Applying the merge algorithm



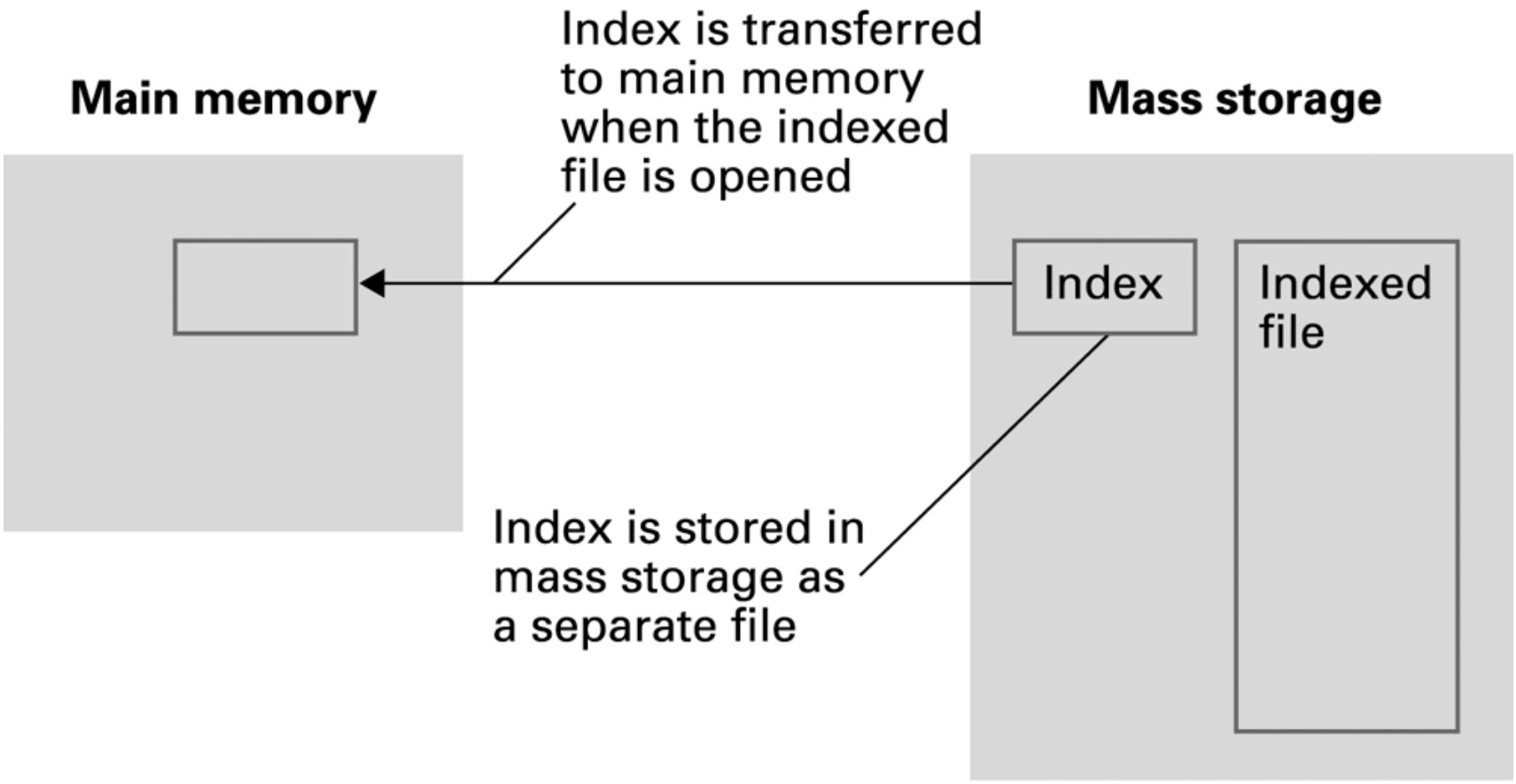
Letters are used to represent entire records.

The particular letter indicates the value of the record's key field.

Indexed Files

- **Index:** A list of key values and the location of their associated records
- Can support multiple indices
  - other than the master key
- Can save space and processing time

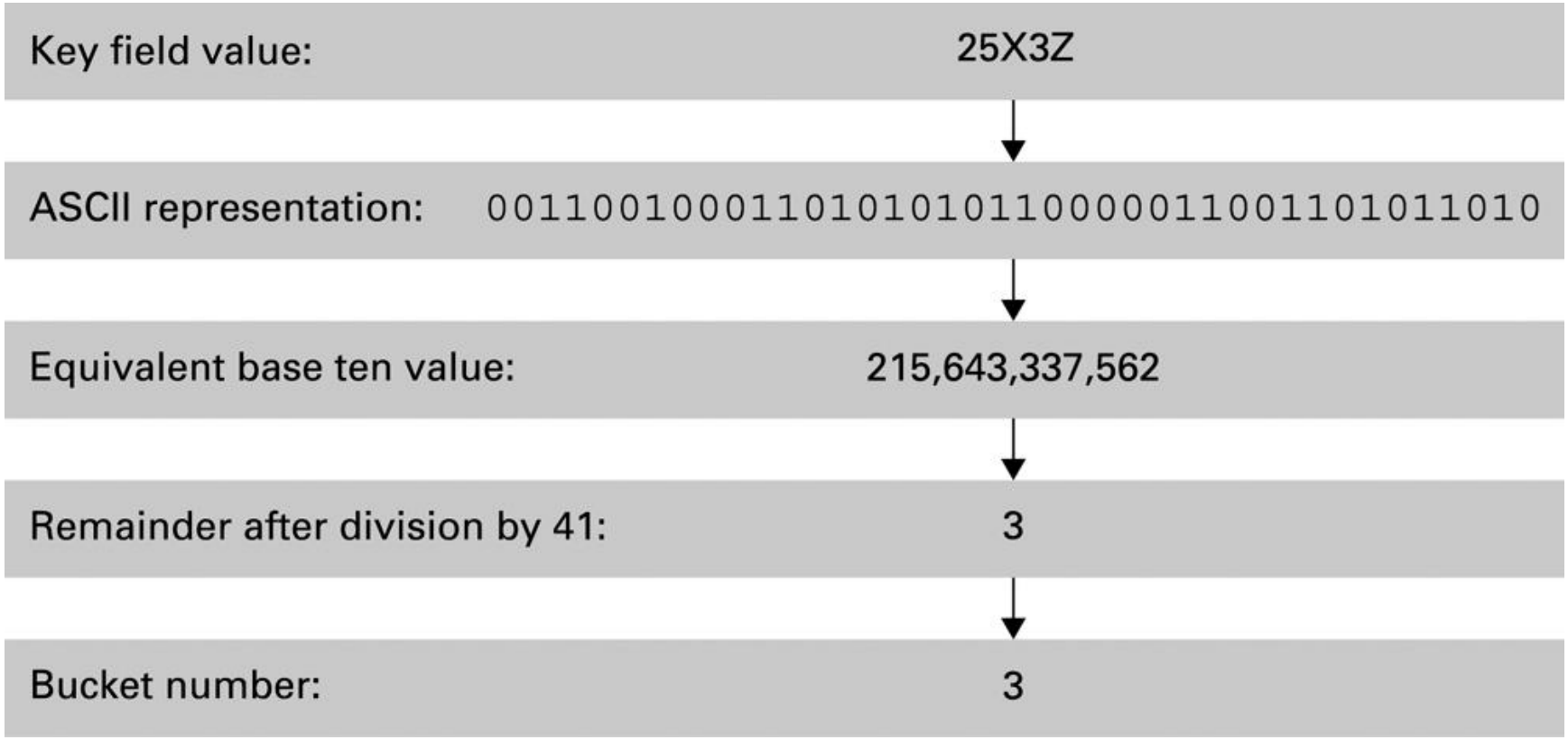
Opening an indexed file



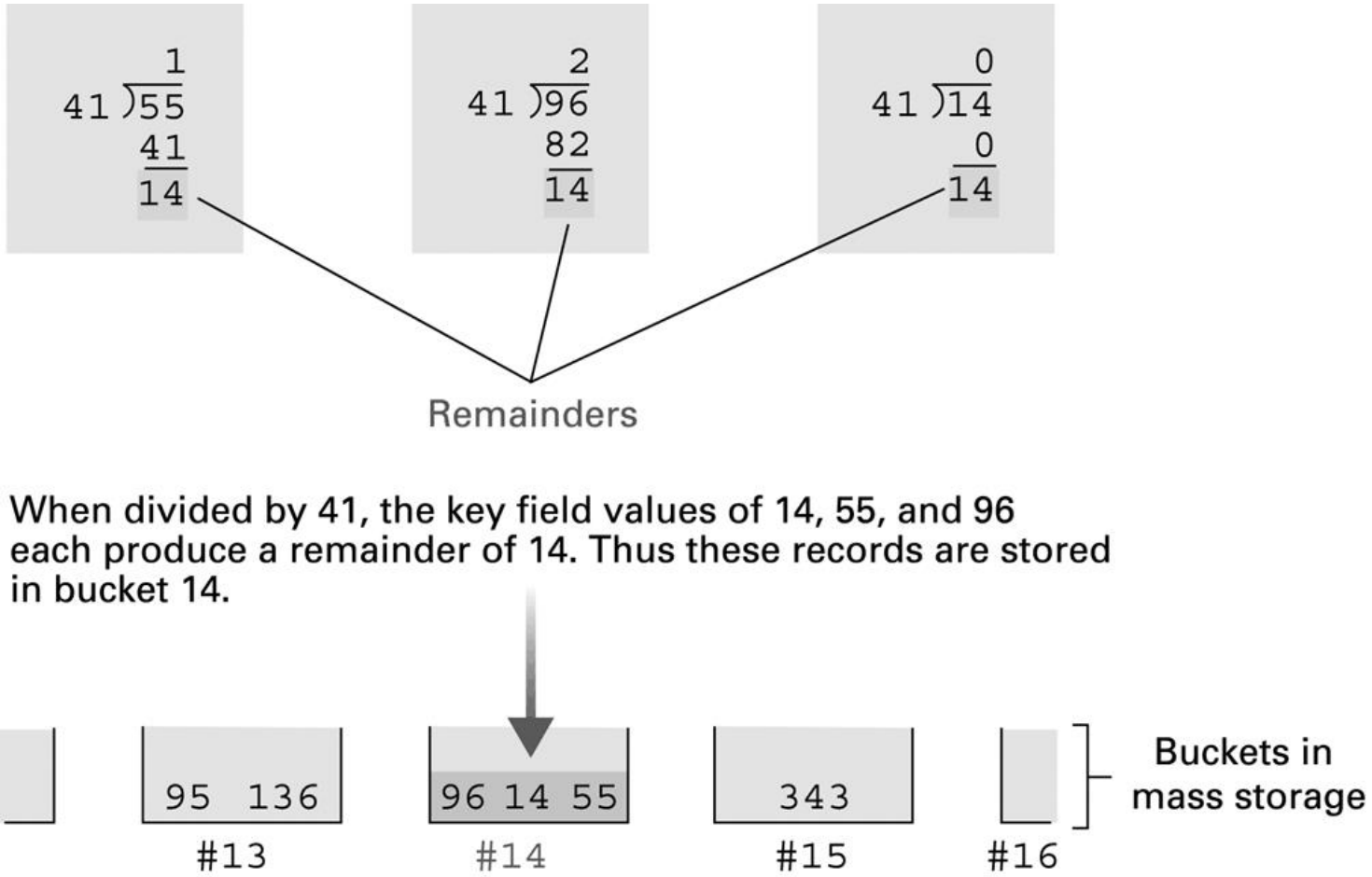
Hashing

- Each record has a key field
- The storage space is divided into **buckets**
- A **hash function** computes a bucket number for each key value
- Each record is stored in the bucket corresponding to the hash of its key

Hashing the key field value 25X3Z to one of 41 buckets



The rudiments of a hashing system



Hash function - typical example

remainder operations with large primes

……

102023 102031 102043 102059 102061 102071 102077
102079 102101 102103 102107 102121 102139 102149
102161 102181 102191 102197 102199 102203 102217
102229 102233 102241 102251 102253 102259 102293
102299 102301 102317 102329 102337 102359 102367
102397 102407 102409 102433 102437 102451 102461
102481 102497 102499 102503 102523 102533 102539
102547 102551 102559 102563 102587 102593 102607
102611 102643 102647 102653 102667 102673 102677
102679 102701 102761 102763 102769 102793 102797
102811 102829 ……

Hash function - typical example

prime[0]=102023, prime[1] = 102031,
prime[2]= 102043, prime[3]= 102059,
prime[4]=102481, prime[5]= 102497,
prime[6]=102499, prime[7]=102503,
prime[8]=102523, prime[9]=102533,
prime[10]=102539, prime[11]= 102547,
prime[12]=102551, prime[13]= 102559,
prime[14]=102563, prime[15]= 102587,
prime[16]=102593, prime[17]=102607,
……

```
struct job_description_type {
  char title[25];
  int    jobid;
  int    skill_code;
}
hash_key (r, table_size) {
   key = r.skill_code;
   key = key * prime[0] + r.jobid;
   for (k = 0; k < 25; k++) {
      key = (key * prime[k + 1])+r.title[k];
   }
  return (key % table_size);
}
```

Collisions in Hashing

- **Collision:** The case of two keys hashing to the same bucket
  - Major problem when table is over 75% full
  - Solution: increase number of buckets and rehash all data

Data Mining

- **Data Mining**
  - The area of computer science that deals with discovering patterns in collections of data
- **Data warehouse:** A static data collection to be mined
  - **Data cube:** Data presented from many perspectives to enable mining

Data Mining Strategies

- Class description
  - Identifying properties that characterize a group of data items
  - *What ages are those ladies with LV bags ?*
- Class discrimination
  - Identifying properties that divide two groups.
- Cluster analysis
  - Identifying groups

Data Mining Strategies

- Association analysis
  - revealing links between data groups
  - *people watch DVDs a lot also eat chips a lot.*
- Outlier analysis
  - Looking for abnormals
  - *A lot of purchases from a seldom used credit card.*
- Sequential pattern analysis
  - Looking for patterns over time

Social Impact of Database Technology

- Problems
  - Massive amounts of personal data are being collected
    - Often without knowledge or meaningful consent of affected people
  - Data merging produces new, more invasive information
  - Errors are widely disseminated and hard to correct
- Remedies
  - Existing legal remedies often difficult to apply
  - Negative publicity may be more effective