Database Systems

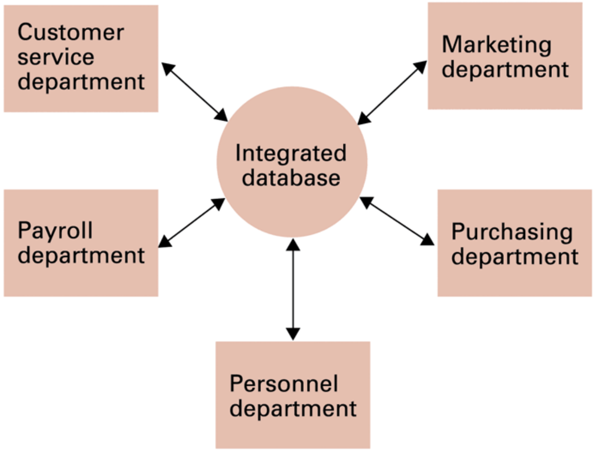
National Chiao Tung University

Chun-Jen Tsai

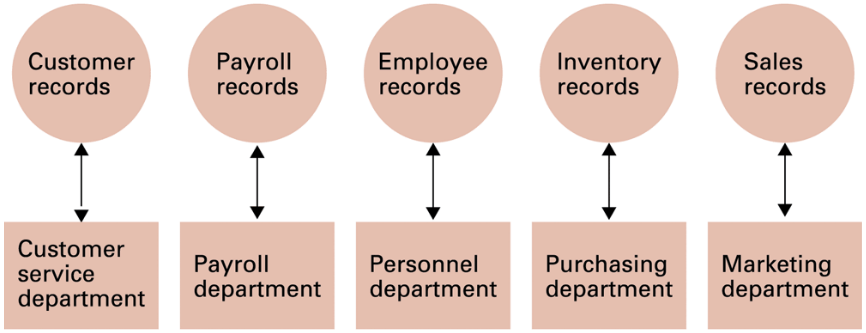
05/30/2012

Definition of a Database

- **Database System**
  - A multidimensional data collection, internal links between its entries make the information accessible from a variety of perspectives

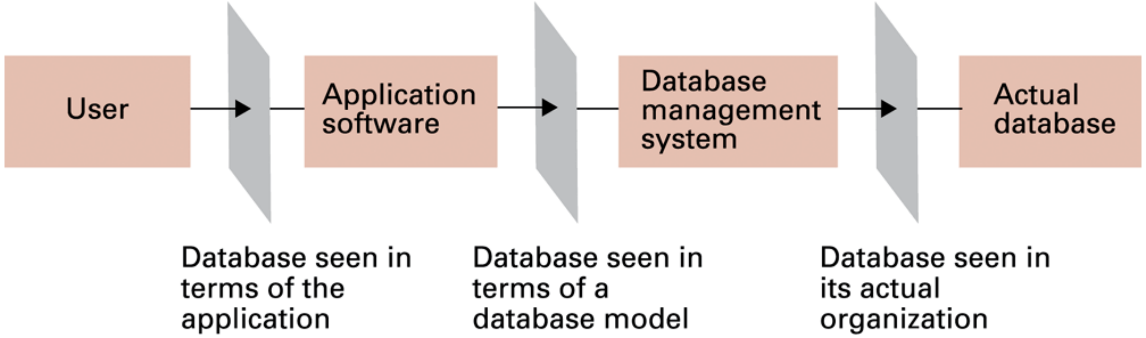


- **Flat File System**
  - One-dimensional file storage system that presents its information from a single point of view



Schemas

- **Schema**
  - A description of the structure of an entire database, used by database software to maintain the database
- **Sub-schema**
  - A description of only a portion of the database pertinent to a particular user's needs, used to prevent sensitive data from being accessed by unauthorized personnel


User
Application software
Database management system
Actual database
Database seen in terms of the application
Database seen in terms of a database model
Database seen in its actual organization

Database Management Systems

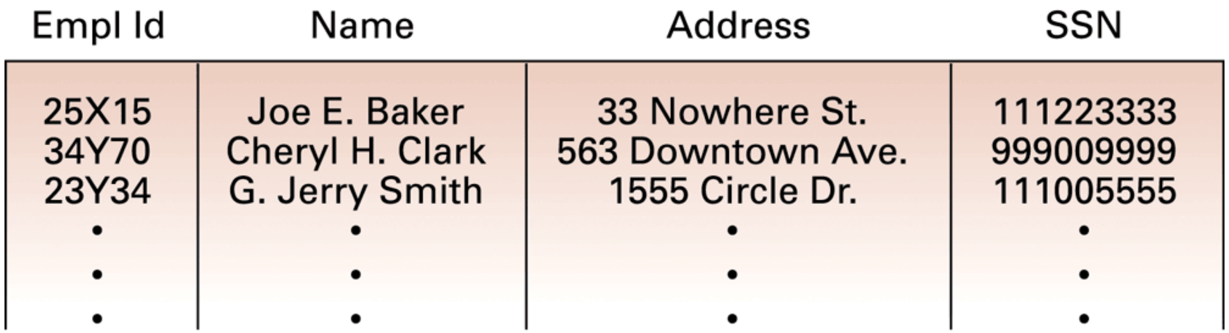
- **Database Management System** (DBMS)
  - A software layer that maintains a database and manipulates it in response to requests from applications
- **Distributed Database**
  - A database stored on multiple machines; the DBMS will mask this organizational detail from its users
- **Data independence**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

Database Models

- Database models:
  - Relational model
  - Object-oriented model
  - Hierarchical model
- Relational model is the most popular model
  - The database is a collection of tables of information
  - Each table is called a “**Relation**”
  - Each column in the table records an **attribute**
  - A row in the table is called a **tuple**

Example of a Relation

- ❑ A relation containing employee information:



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
•	•	•	•
•	•	•	•
•	•	•	•

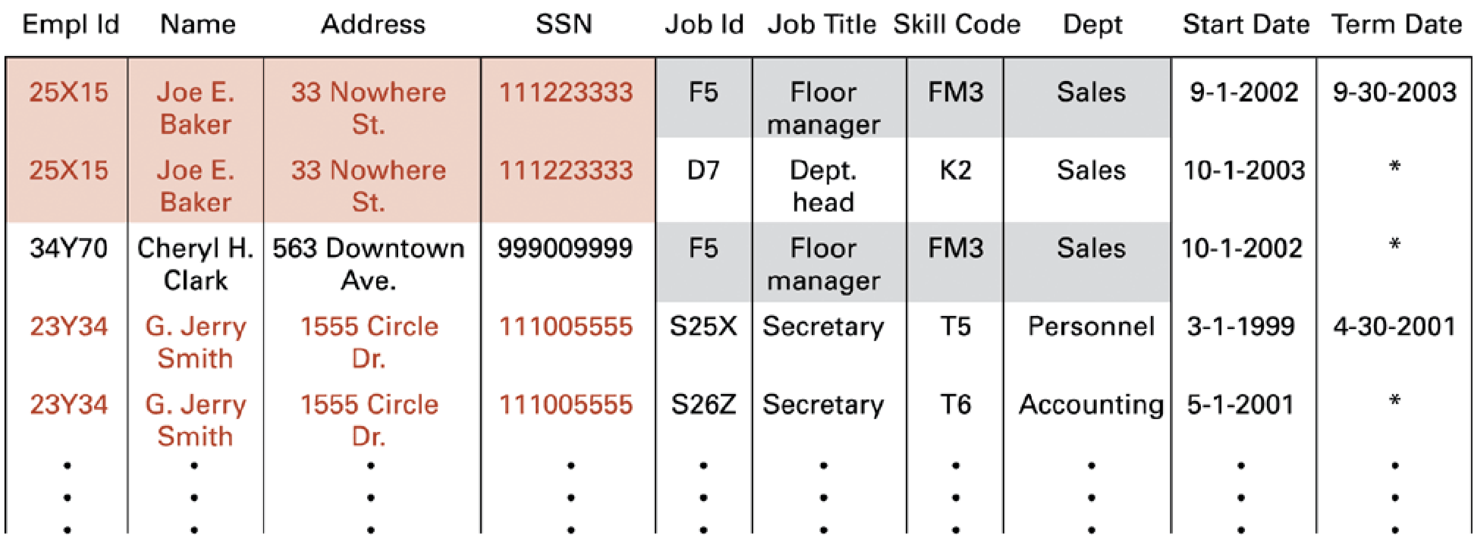
- ❑ The schema can also be denoted as follows

```
employee_info
{
    Empl_Id: string
    Name: string
    Address: string
    SSN: int
} primary key (Empl_Id, Name, SSN)
```

Relational Design Issues

❑ In general, we want to avoid multiple concepts within one relation; because:

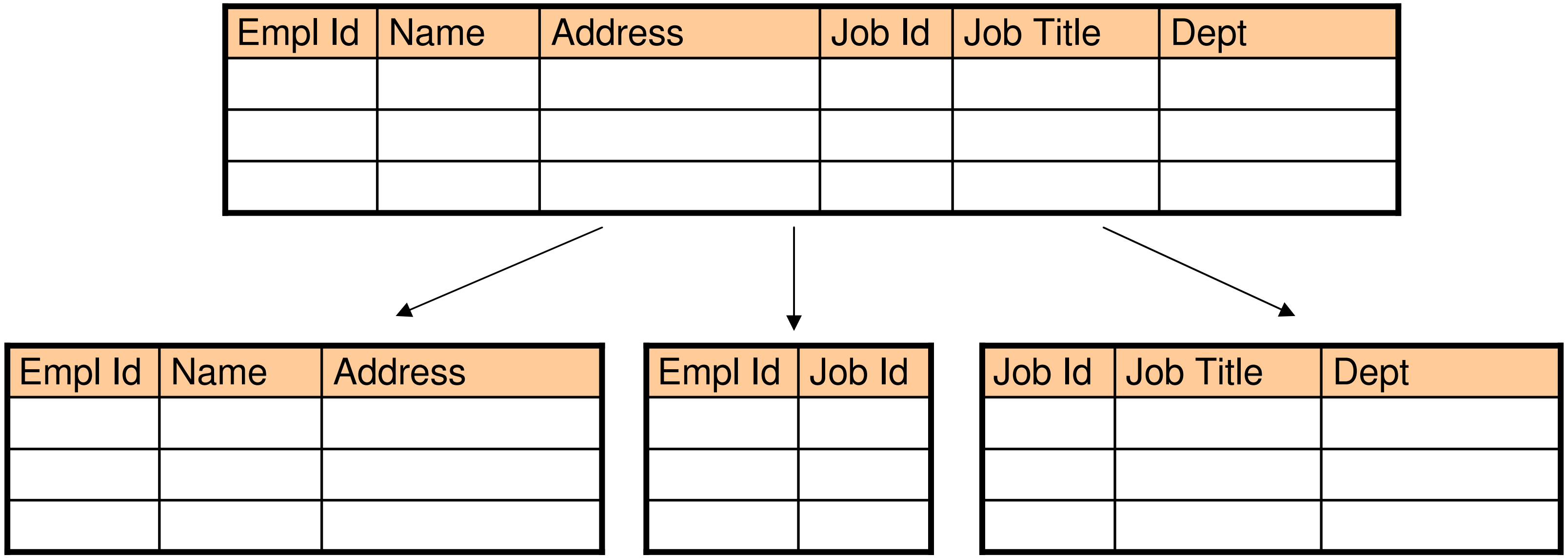
- It can lead to redundant data
- Deleting a tuple could also delete necessary but unrelated information



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	9-1-2002	9-30-2003
25X15	Joe E. Baker	33 Nowhere St.	111223333	D7	Dept. head	K2	Sales	10-1-2003	*
34Y70	Cheryl H. Clark	563 Downtown Ave.	999009999	F5	Floor manager	FM3	Sales	10-1-2002	*
23Y34	G. Jerry Smith	1555 Circle Dr.	111005555	S25X	Secretary	T5	Personnel	3-1-1999	4-30-2001
23Y34	G. Jerry Smith	1555 Circle Dr.	111005555	S26Z	Secretary	T6	Accounting	5-1-2001	*
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

Decomposition

- ❑ We can divide the columns of a relation into two or more relations, duplicating those columns necessary to maintain relationships; this techniques is called **decomposition**



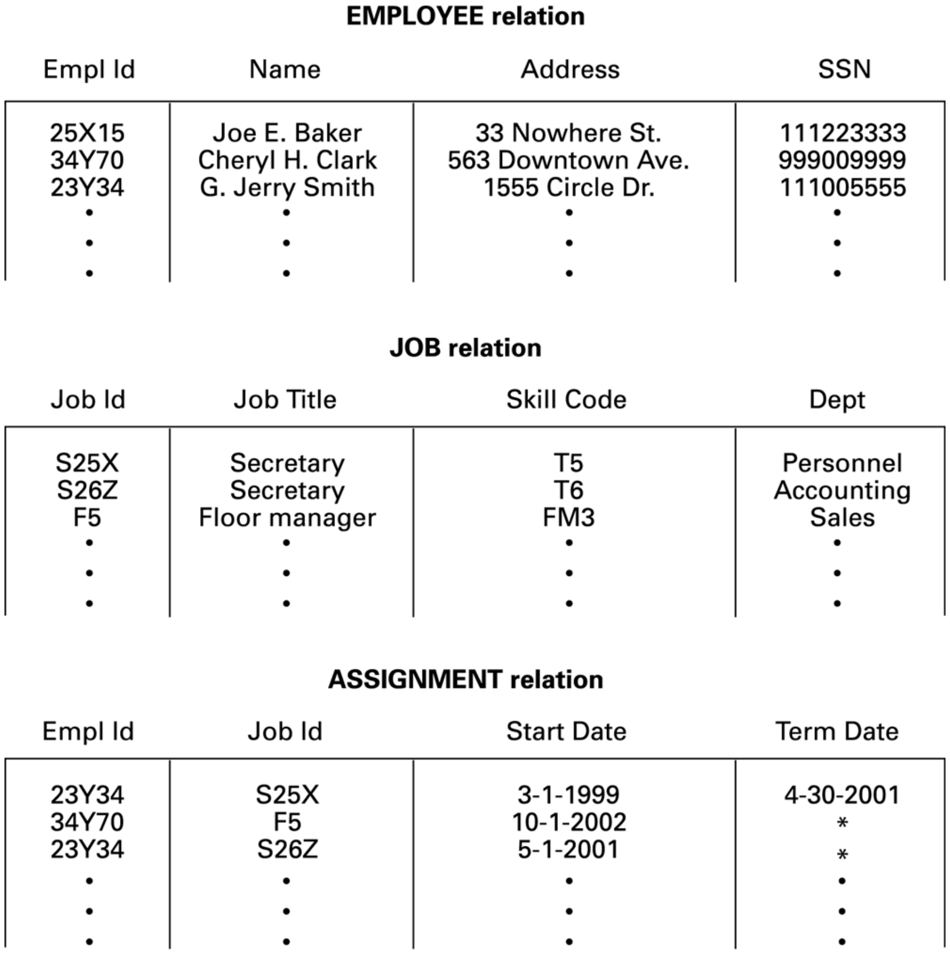
Empl Id	Name	Address	Job Id	Job Title	Dept

Empl Id	Name	Address

Empl Id	Job Id

Job Id	Job Title	Dept

Example of Decomposition



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

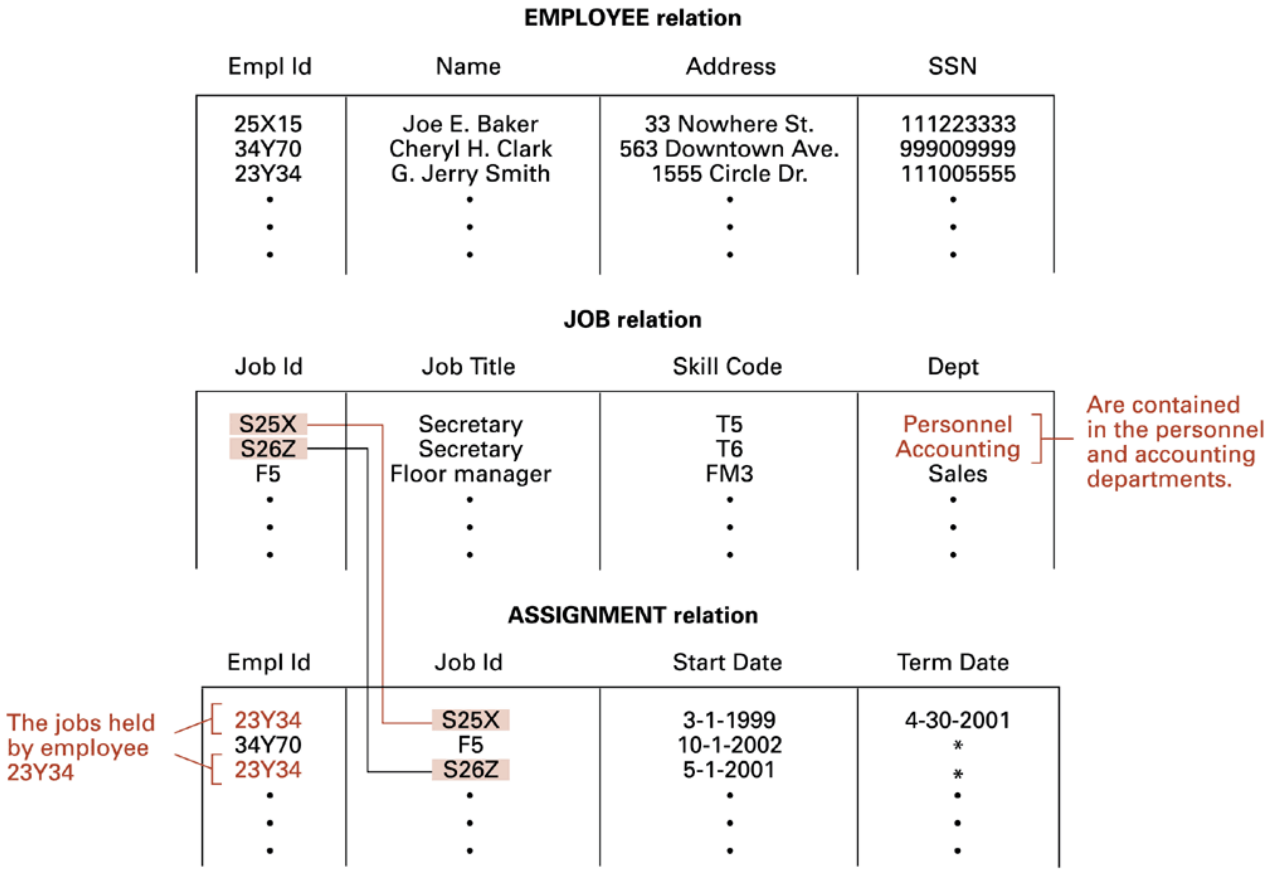
Job Id	Job Title	Skill Code	Dept
S25X	Secretary	T5	Personnel
S26Z	Secretary	T6	Accounting
F5	Floor manager	FM3	Sales
•	•	•	•
•	•	•	•
•	•	•	•

ASSIGNMENT relation

Empl Id	Job Id	Start Date	Term Date
23Y34	S25X	3-1-1999	4-30-2001
34Y70	F5	10-1-2002	*
23Y34	S26Z	5-1-2001	*
•	•	•	•
•	•	•	•
•	•	•	•

Example of Information Retrieval

- ❑ Finding the departments in which 23Y34 has worked



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

Are contained in the personnel and accounting departments.

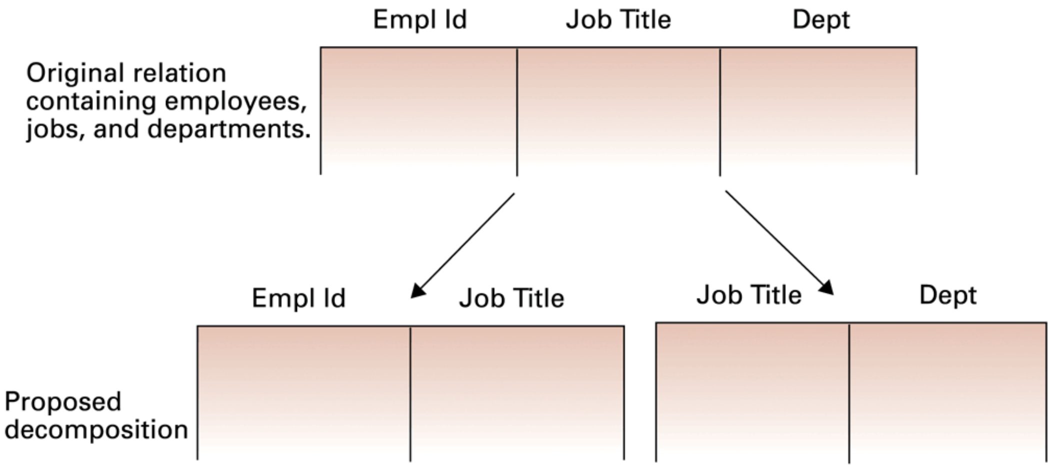
ASSIGNMENT relation

Empl Id	Job Id	Start Date	Term Date
23Y34	S25X	3-1-1999	4-30-2001
34Y70	F5	10-1-2002	*
23Y34	S26Z	5-1-2001	*
•	•	•	•
•	•	•	•
•	•	•	•

The jobs held by employee 23Y34

Lossless Decomposition

- ❑ Sometimes, decomposition can cause loss of information

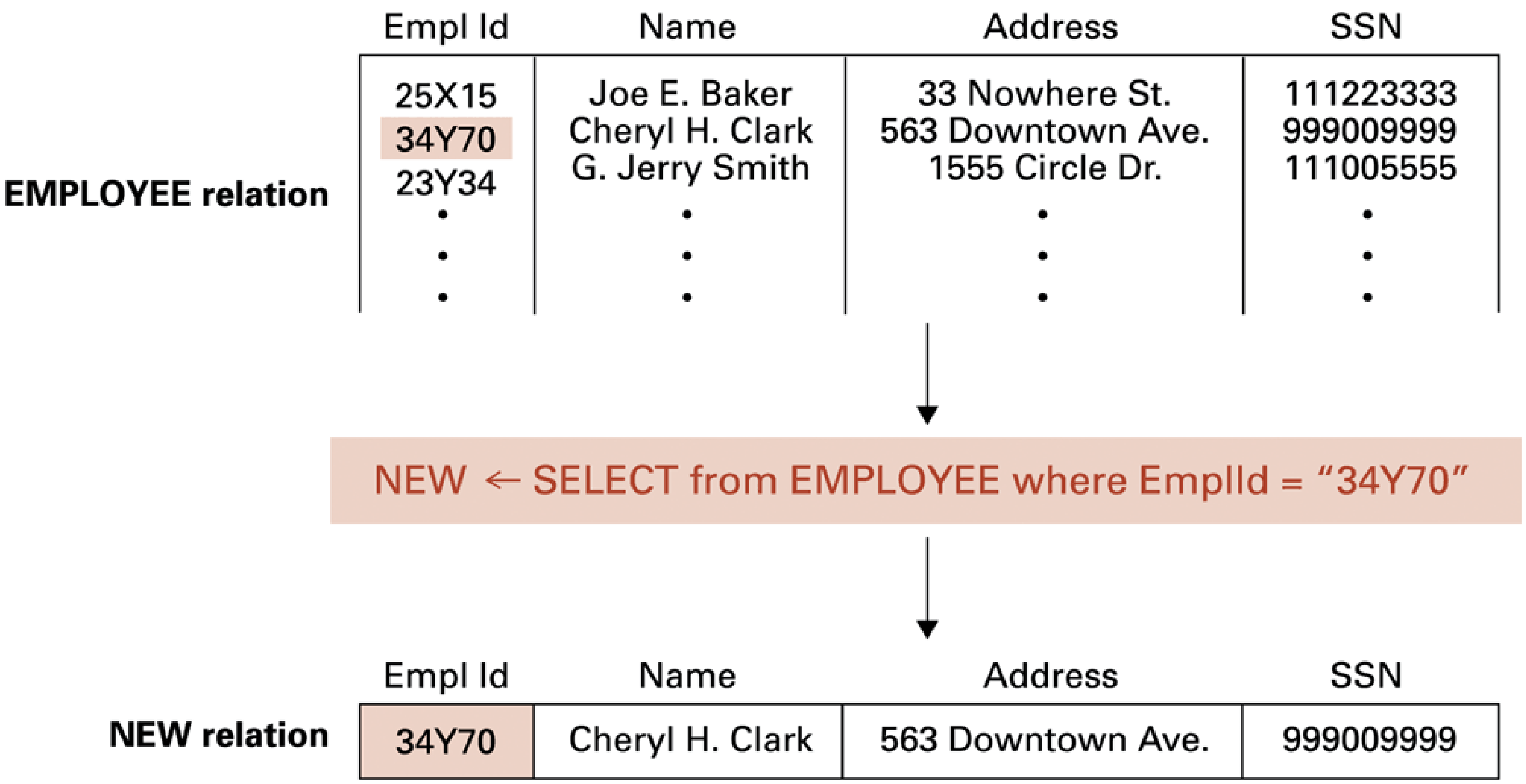


- ❑ A correct decomposition that does not lose info. is called lossless (non-loss) decomposition

Relational Operations

- ❑ **Select**: choose rows
- ❑ **Project**: choose columns
- ❑ **Join**: assemble information from two or more relations

The SELECT Operation



EMPLOYEE relation

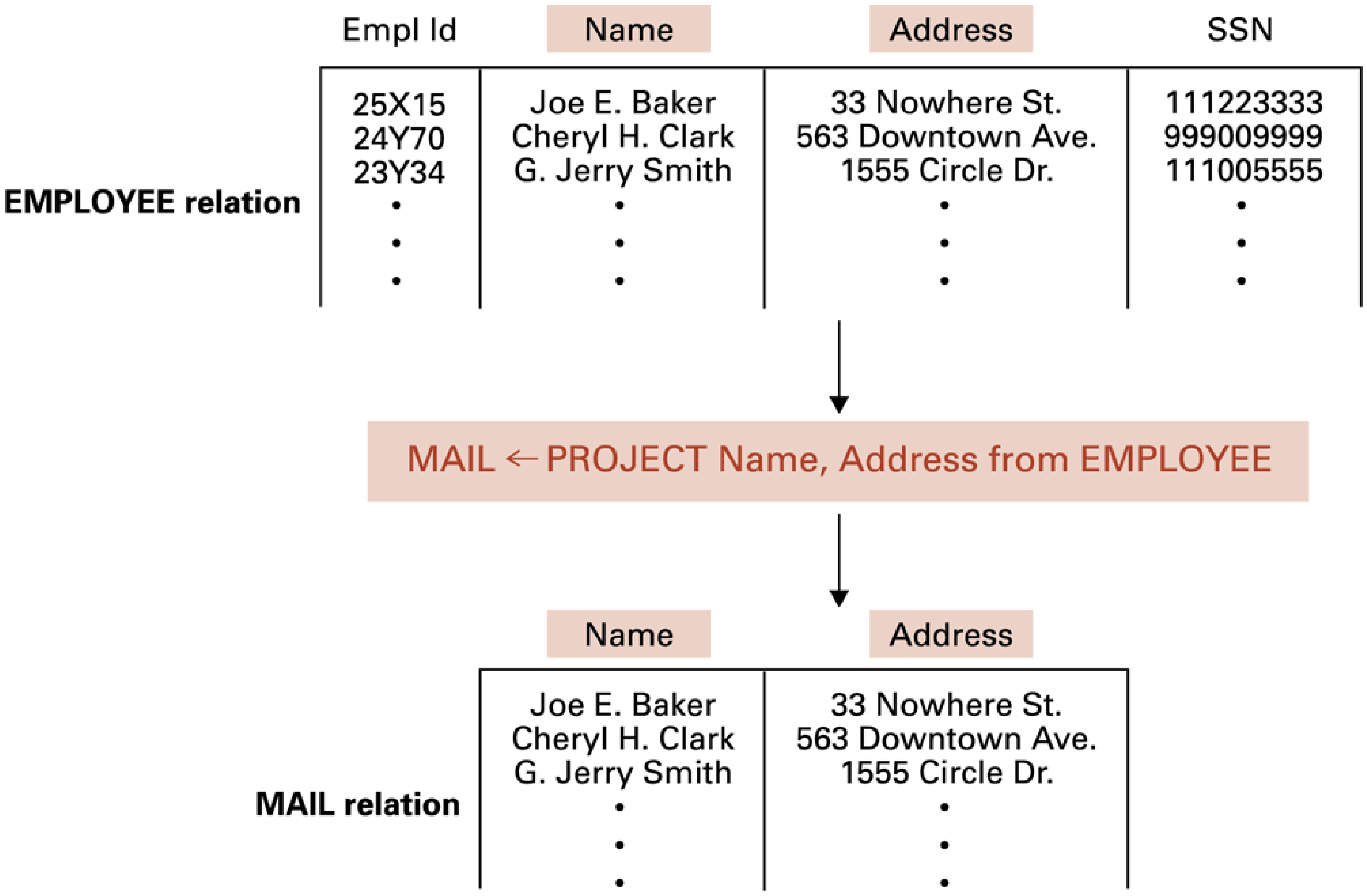
Empl Id	Name	Address	SSN
25X15	Joe E. Baker	33 Nowhere St.	111223333
34Y70	Cheryl H. Clark	563 Downtown Ave.	999009999
23Y34	G. Jerry Smith	1555 Circle Dr.	111005555
•	•	•	•
•	•	•	•
•	•	•	•

NEW ← SELECT from EMPLOYEE where EmplId = “34Y70”

NEW relation

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

The PROJECT Operation



EMPLOYEE relation

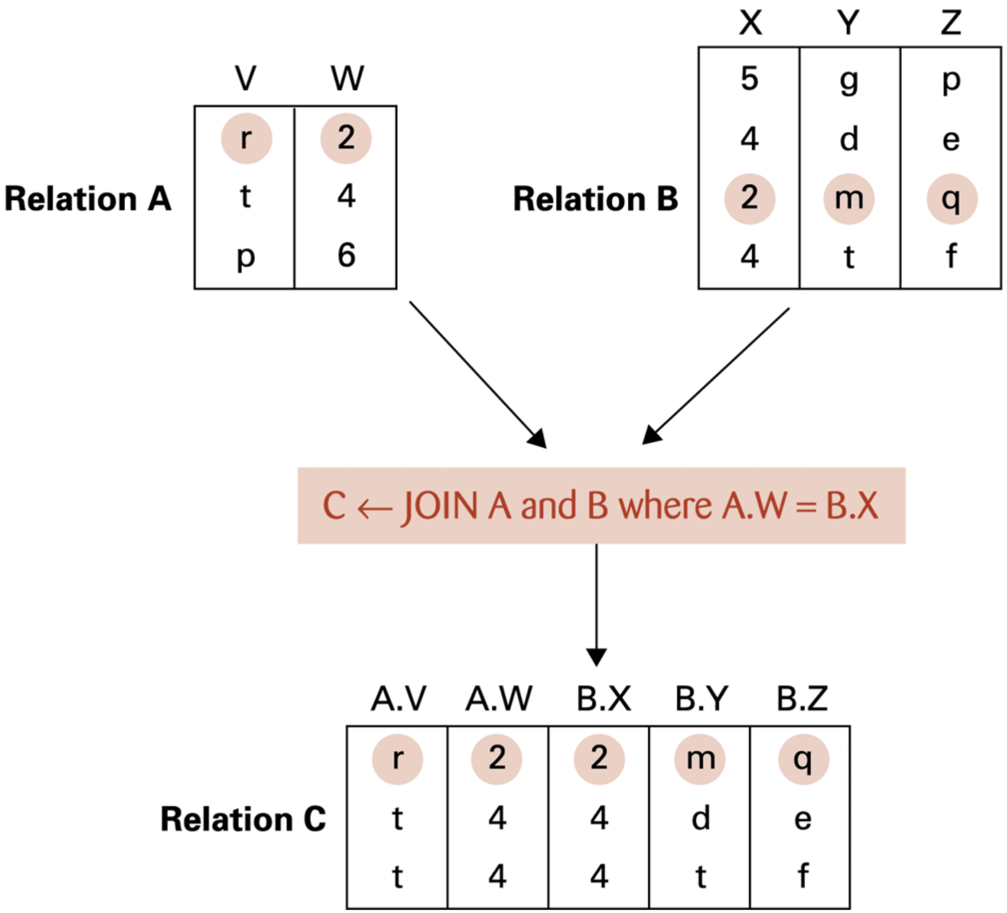
Empl Id	Name	Address	SSN
25X15	Joe E. Baker	33 Nowhere St.	111223333
24Y70	Cheryl H. Clark	563 Downtown Ave.	999009999
23Y34	G. Jerry Smith	1555 Circle Dr.	111005555
•	•	•	•
•	•	•	•
•	•	•	•

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 (1/2)



Relation A

V	W
r	2
t	4
p	6

Relation B

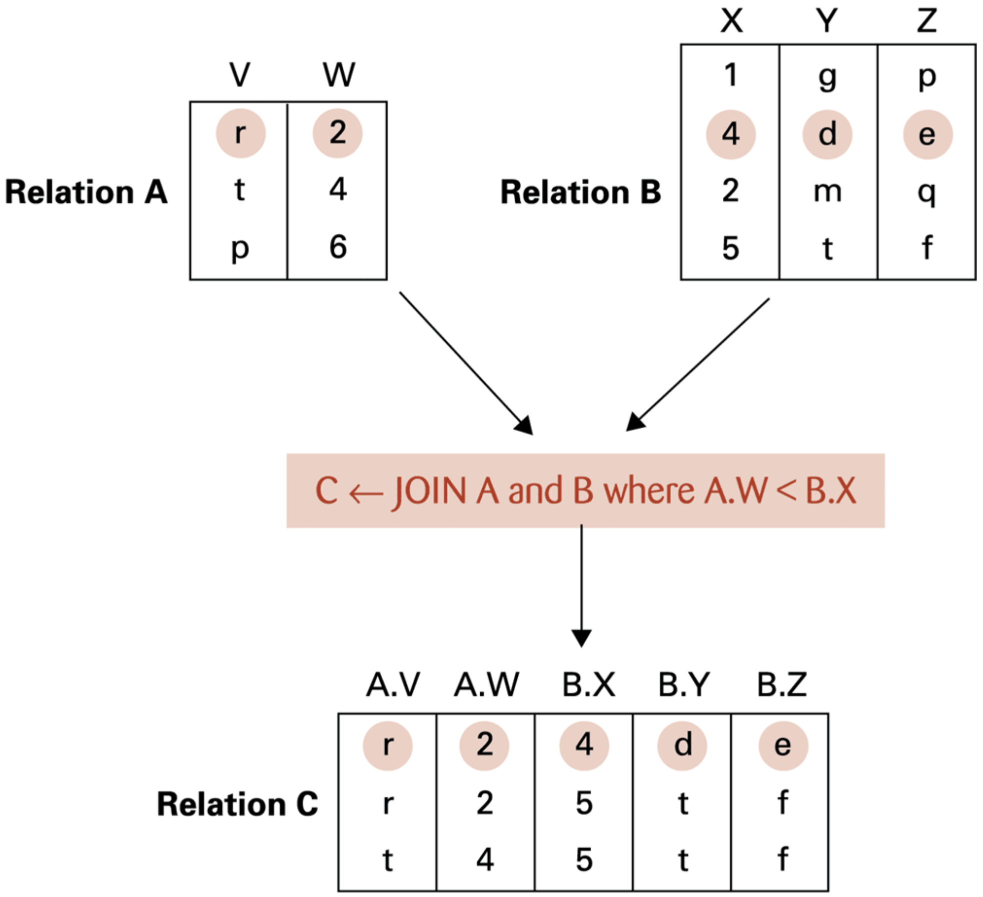
X	Y	Z
5	g	p
4	d	e
2	m	q
4	t	f

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

The JOIN Operation (2/2)



Relation A

V	W
r	2
t	4
p	6

Relation B

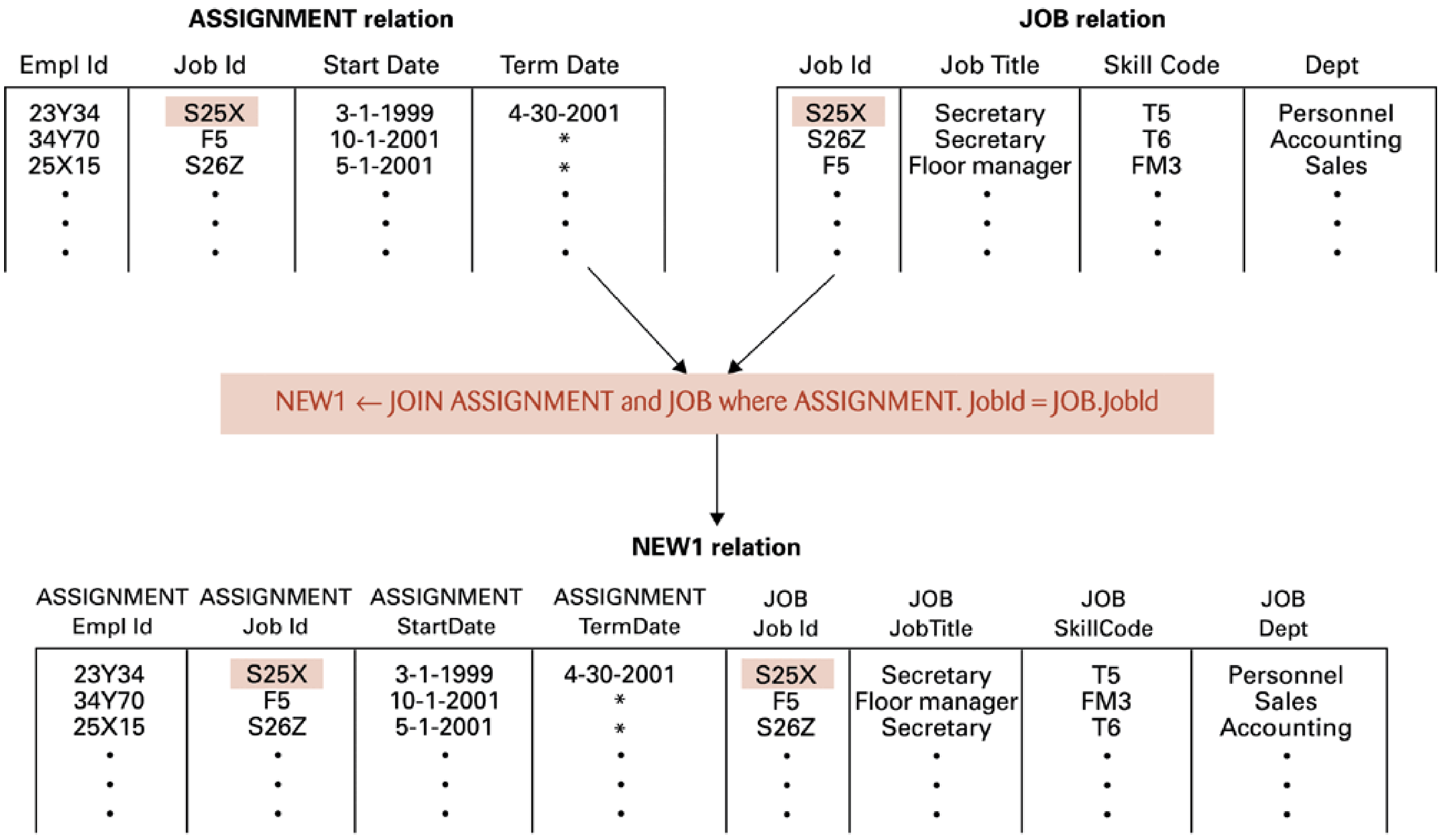
X	Y	Z
1	g	p
4	d	e
2	m	q
5	t	f

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



ASSIGNMENT relation

Empl Id	Job Id	Start Date	Term Date
23Y34	S25X	3-1-1999	4-30-2001
34Y70	F5	10-1-2001	*
25X15	S26Z	5-1-2001	*
•	•	•	•
•	•	•	•
•	•	•	•

JOB relation

Job Id	Job Title	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	4-30-2001	S25X	Secretary	T5	Personnel
34Y70	F5	10-1-2001	*	F5	Floor manager	FM3	Sales
25X15	S26Z	5-1-2001	*	S26Z	Secretary	T6	Accounting
•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•

Structured Query Language (SQL)

- SQL is the most popular language used to create, modify, retrieve and manipulate information from relational database management systems
- SQL was originally designed by IBM in 1970s, and became an ISO international standard in 1987
- In SQL, some operations to manipulate tuples are as follows:
  - `insert`
  - `update`
  - `delete`
  - `select`

SQL Examples

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

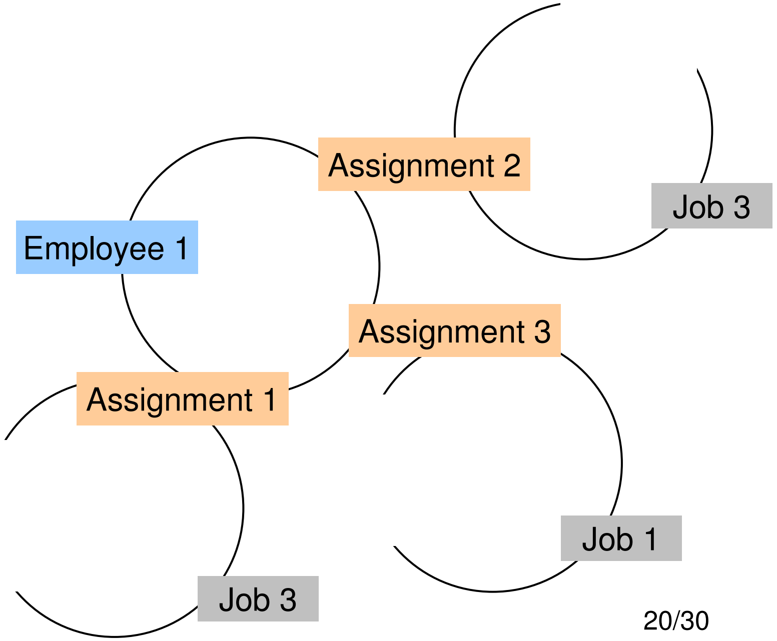
```
insert into EMPLOYEE
values (‘43212’, ‘Sue A. Burt’, ‘33 Fair St.’,
‘444661111’)
```

```
delete from EMPLOYEE
where Name = ‘G. Jerry Smith’
```

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

Object-Oriented Databases

- A database constructed by applying the object-oriented paradigm
  - Each data entity stored as a persistent object
  - Relationships indicated by links between objects
  - DBMS maintains inter-object links
- Example classes of objects:
  - Employee (ID, name, address)
  - Assignment (start/end dates)
  - Job (title, skills)



Advantages of OO Databases

- Many database applications are designed using OO paradigm, why not the database itself?
- OO design allows hiding of the implementation details of attributes
  - Example: “name” attribute has different formats, implement name attribute as an object is more flexible
- Can handle exotic data types
  - Example: a multimedia data item is often composed of several attributes (audio, video, graphics, descriptions), OO design concept can encapsulate them into one data object
- Can store intelligent entities
  - Intelligence is inside the methods of the object
    → if database is smart, DBMS can be simpler

Maintaining Database Integrity (1/2)

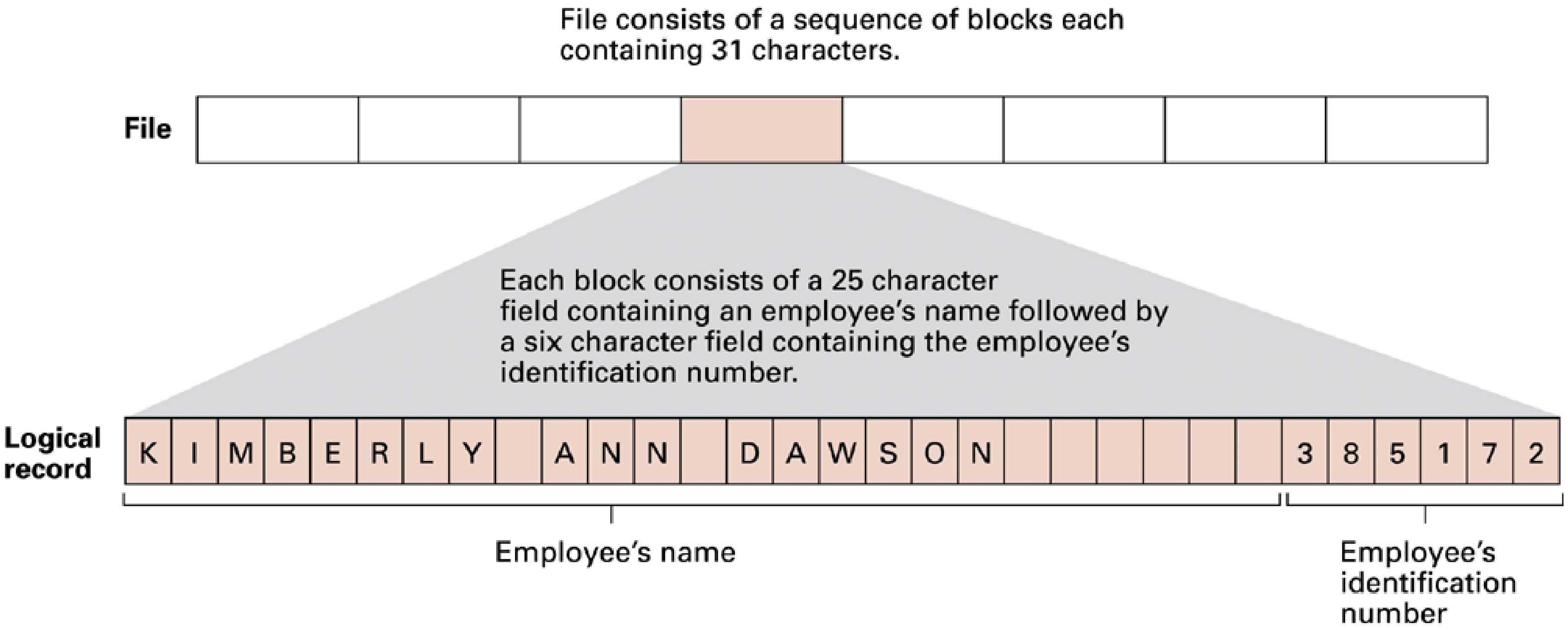
- A transaction is a sequence of operations that must all happen together
  - Example: transferring money between bank accounts
- Transaction log is non-volatile record of each transaction’s activities, built before the transaction is allowed to happen
  - **Commit point** is the point at which transaction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

Maintaining Database Integrity (2/2)

- ❑ Simultaneous access problems
  - Incorrect summary problem
  - Lost update problem
- ❑ To preventing others from accessing data being used by a transaction, a locking mechanism is required
  - **Shared** lock: used when reading data
  - **Exclusive** lock: used when altering data
- ❑ A common way to resolve deadlock in DBMS is the *wound-wait protocol*:
  - In a hold-and-wait deadlock situation, the data item held by the younger transaction will be forcibly retrieved by the older transaction

File Structure for Databases

- ❑ The structure of a simple employee file can be implemented as a text file



File consists of a sequence of blocks each containing 31 characters.

File

Each block consists of a 25 character field containing an employee's name followed by a six character field containing the employee's identification number.

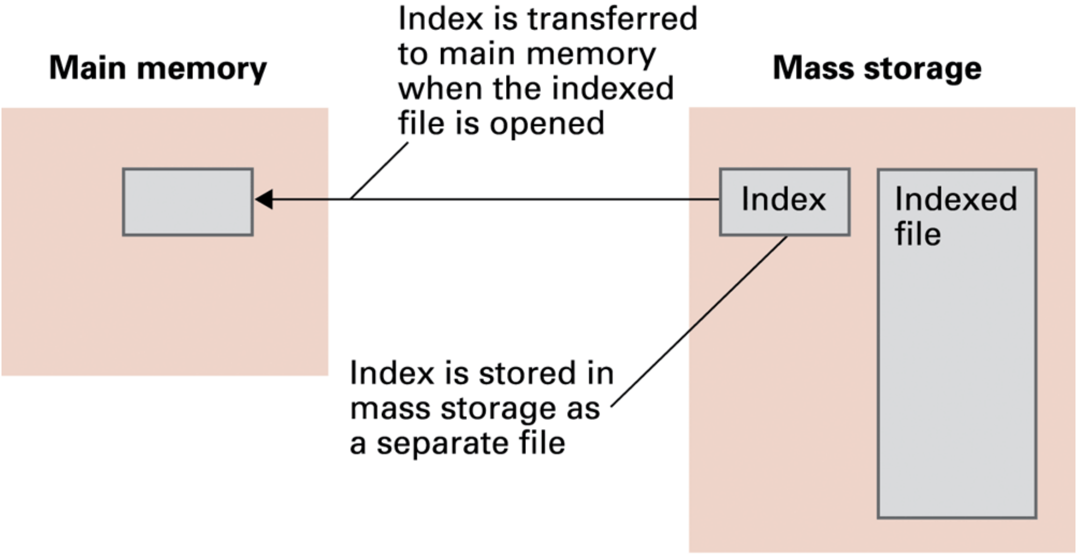
Logical record	K	I	M	B	E	R	L	Y		A	N	N		D	A	W	S	O	N							3	8	5	1	7	2

Employee's name

Employee's identification number

Indexed Files

- An index is a list of (*key*, *location*) pairs
  - Sorted by key values
  - *location* = where the record is stored
- An index file is for fast access to items in a file
- Open of an indexed file

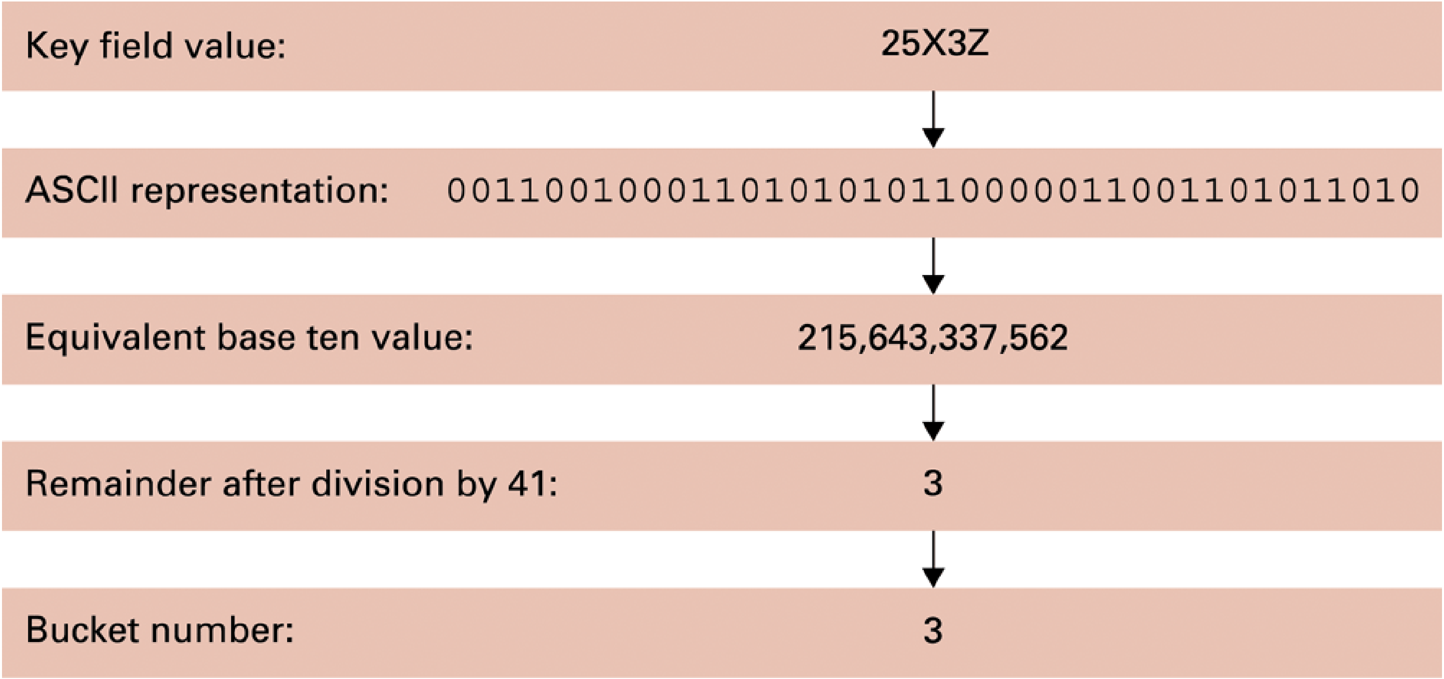

Main memory
Index is transferred to main memory when the indexed file is opened
Mass storage
Index
Indexed file
Index is stored in mass storage as a separate file

Hash Files

- Another technique for fast accessing of a file is called hashing
  - Each record has a **key**
  - The master file 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 Example

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



Key field value: 25X3Z

ASCII representation: 0011001000110101010110000011001101011010

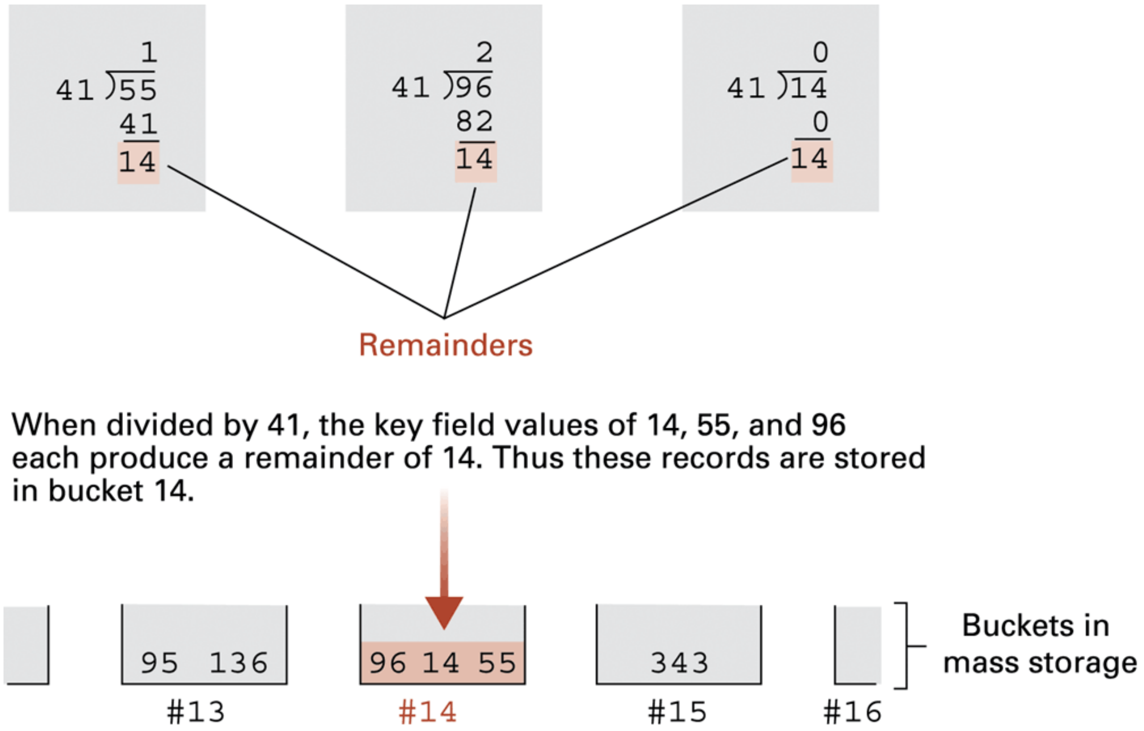
Equivalent base ten value: 215,643,337,562

Remainder after division by 41: 3

Bucket number: 3

Collisions in Hashing

- ❑ Collision happens if two keys hash to the same bucket
  - ■ Major problem when table is over 75% full
  - ■ Solution: increase number of buckets and rehash all data



When divided by 41, the key field values of 14, 55, and 96 each produce a remainder of 14. Thus these records are stored in bucket 14.

Data Mining

- Data mining is a set of techniques for discovering patterns in collections of data
  - Relies heavily on statistical analyses
- Data warehouse is the static data collection to be mined
  - Data cube is the data presented from many perspectives to enable mining
- Raises significant ethical issues when it involves personal information

Data Mining Strategies

- ❑ Class description
- ❑ Class discrimination
- ❑ Cluster analysis
- ❑ Association analysis
- ❑ Outlier analysis
- ❑ Sequential pattern analysis