Unit 1
Introduction to DBMS
(Database Management Systems)
本課程講授內容

• **PART I: 入門與導論**
  - Overview
  - DB2系統及SQL語言
  - 闡述關係式資料模型(The Relational Model)
  - 階層式資料模型(The Hierarchical Model)簡介
  - 網狀式資料模型(The Network Model)簡介

• **PART II: 資料庫設計 (Database Design)**
  - 資料庫問題分析與 E-R Model
  - 資料庫的表格正規化
  - 設計介面增刪查改資料庫

• **PART III: 進階探討**
  - 快速存取方法(Access Methods)
  - 資料庫回復(Database Recovery)
  - 協同控制(Concurrency Control)
  - 資料安全與資料正確(Security and Integrity)
  - 查詢最佳化_Query Optimization_
  - 分散式資料庫系統(Distributed Database)
PART I: 入門與導論

- DB2系統及SQL語言:
  - 介紹最多人使用的查詢語言SQL
  - 配合實作習題，先試用一個免費的DBMS系統MySQL
  - 好比要學開車可先在大停車場讓你繞一圈
  - 這樣我們在介紹下單元關連式資料模型的設計原理時會比較容易想像

- 關連式資料模型(The Relational Model):
  - 闡述使用者視資料庫為許多表格(tables)組成的關連式資料庫之原始設計原理
  - 這是關連式資料庫的理論基礎

- 階層式資料模型(The Hierarchical Model)及網狀式資料模型(The Network Model):
  - 這是最早的二個資料模型
  - 介紹這二模型將有助於我們對資料庫模型更深入了解，並知其來龍去脈
  - 我們將簡要的說明其原始設計原理
Contents of PART I: 入門與導論

- Unit 1  Introduction to DBMS
- Unit 2  DB2 and SQL
- Unit 3  The Relational Model
- Unit 4  The Hierarchical Model
- Unit 5  The Network Model

References:
3. Cited papers
Outline of Unit 1

1.1 Information Systems
1.2 An Overview of a Database System
1.3 Why Database Systems?
1.4 An Architecture for a Database System
1.5 Data Models
1.6 Establish/Design a Database System
1.7 Extending Database Technology
1.8 Discussion and Remarks
1.1 Information Systems
Stages of Information System

- **Stage 0: Manual Information System**
  - Records
  - Files
  - Index Cards

- **Stage 1: Sequential Information Systems**
  - Tapes
  - Files
  - slow, non-interactive, redundancy,...

- **Stage 2: File Based Information Systems**
  - Disk (direct access)
  - application program has its own file → data dependence
  - data redundancy

- **Stage 3: DBMS based Information Systems**
  - Generalized data management software
  - Transaction processing
Stage 0: Manual Information System

- 圖書館index card
- 醫院診所病歷卡
Stage 1: Sequential Information Systems

- The old computer data center at NASA's Jet Propulsion Laboratory ...
Stage 2: File Based Information Systems

- Conventional **Data Processing** techniques:

![Diagram showing file based information systems]
Stage 2: File Based Information Systems (cont.)

Customer
- Customer No.
- Customer Name
- Customer Addr.
- Social Security ID

Invoice
- Customer No.
- Customer Name
- Part No.
- Quantities
- Unit Price

Inventory
- Part No.
- Part Description
- Unit Price
- Supplier
- Quantities Remain
- Quantities Ordered

Parts
- Part No.
- Part Description
- Supplier
- Quantities Ordered
- Customer Name
- Unit Price
Stage 2: File Based Information Systems (cont.)

- Advantages: File Systems are simple in design

- Disadvantages:
  - Data Redundancy:
    - a waste of memory
    - high update cost
    - data inconsistency
  - Data Incompleteness
  - Data Insecure
  - Application Program Unstable
    - file system (application program)
      - data changed  →  data structure changed
      - →  program changed
Solution: Database Systems!

- Objectives of Database Systems:
  - eliminate unnecessary data redundancy
  - maintain data integrity
  - control security
  - achieve data independence
  - add program stability
Stage 3: DBMS based Information Systems:

Basic Approach - Integration

- (1) Integration of Information
  - Description of the *integrated view* of data is the "Conceptual Schema" of the database
Stage 3: DBMS based Information Systems:

Basic Approach – Simple views and High level language

- (2) Provide simple views (External Schema) and high level language (e.g. SQL) for users to manipulate (handle) data
  - High level language: e.g. SQL (Structured Query Language)
    
    <e.g.>: SELECT SNAME 
    FROM S 
    WHERE S#= 'S4';

  - Description of user's view of data is the "external schema" or "subschema" or "view".

  - High-level languages (Query Language): SQL
    
    (1) Data Definition Language: define format
    (2) Data Manipulation Language: retrieve, insert, delete, update

  - Emphasize: EASE OF USE !!
Stage 3: DBMS based Information Systems: 
Basic Approach - Storage/Access Method

- (3) Efficient Storage/Access Techniques:
  - implemented once rather than duplicated in all application programs.

Diagram:

- User: query in SQL
  - Language Processor
    - Access Methods Calls
      - Access Method
        - (B+ tree, Dynamic Hashing)
      - I/O calls
  - DBMS
Stage 3: DBMS based Information Systems: Basic Approach - Transaction Management

- (4) Provide **Transaction Management**:
  - Concurrency Control
  - Recovery
  - Security
  -
Example: A Simple Query Processing

**Query in SQL** :

```sql
SELECT CUSTOMER.NAME 
FROM CUSTOMER, INVOICE 
WHERE REGION = 'N.Y.' AND 
AMOUNT > 10000 AND 
CUSTOMER.C#=INVOICE.C
```

**Internal Form** :

\[ \Pi(\sigma(S \bowtie SP)) \]

**Operator** :
- SCAN C using region index, create C
- SCAN I using amount index, create I
- SORT C? and I? on C#
- JOIN C? and I? on C#
- EXTRACT name field

**Calls to Access Method** :
- OPEN SCAN on C with region index
- GET next tuple
  - ...
- Calls to file system :
  - GET 10th to 25th bytes from block #6 of file #5

**Calls to Access Method**
- OPEN SCAN on C with region index
- GET next tuple
  - ...
- Calls to file system :
  - GET 10th to 25th bytes from block #6 of file #5
1.2 An Overview of a Database System
Database System: Introduction

- Database Management System (DBMS)
  - Contains a large bodies of information
  - Collection of interrelated data (database)
  - Set of programs to access the data

- Goal of a DBMS:
  - provides a way to store and retrieve database information
    - convenient and
    - efficient
Database System: Functions of DBMS

- Functions of DBMS: Management of Data (MOD)
  - Defining structure for storage data
  - Proving mechanisms for manipulation of data
  - Ensure safety of data (system crashes, unauthorized access, misused, …)
  - Concurrent control in multi-user environment

- Computer Scientists:
  - Developed a lot of concepts and techniques for MOD
  - Concepts and techniques form the focus of this course
Database System: Data Integrated and Shared

Application program

DBMS

End-user
Database System: Major components

- Data: integrated and shared
- Hardware: disk, CPU, Main Memory, ...
- Software: DBMS
- Users:
  1. Application programmers
  2. End users
  3. Database administrator (DBA)
     - Defining external schema
     - Defining conceptual schema
     - Defining internal schema
     - Liaison with users
     - Defining security and integrity checks
     - Defining backup and recovery procedures
     - Monitoring performance and changing requirements
An Example: Supplier-and-Parts Database

### Query:
- 列出住在London的供應商名字？
- Get the total number of suppliers.
- Total QTY of SP?
- Get supplier names for suppliers who supply part P2?
- Get supplier names for suppliers who supply red color parts?
- …
1.3 Why Database System?
Why Database System?

- Answer: **Easy to retrieve information!**

- Word, Excel vs. Access

  <e.g.> Supplier-and-Parts Database

<table>
<thead>
<tr>
<th>S#</th>
<th>SNAME</th>
<th>STATUS</th>
<th>CITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>Smith</td>
<td>20</td>
<td>London</td>
</tr>
<tr>
<td>S2</td>
<td>Jones</td>
<td>10</td>
<td>Paris</td>
</tr>
<tr>
<td>S3</td>
<td>Blake</td>
<td>30</td>
<td>Paris</td>
</tr>
<tr>
<td>S4</td>
<td>Clark</td>
<td>20</td>
<td>London</td>
</tr>
<tr>
<td>S5</td>
<td>Adams</td>
<td>30</td>
<td>Athens</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>P#</th>
<th>PNAME</th>
<th>COLOR</th>
<th>WEIGHT</th>
<th>CITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>Nut</td>
<td>Red</td>
<td>12</td>
<td>London</td>
</tr>
<tr>
<td>P2</td>
<td>Bolt</td>
<td>Green</td>
<td>17</td>
<td>Paris</td>
</tr>
<tr>
<td>P3</td>
<td>Screw</td>
<td>Blue</td>
<td>17</td>
<td>Rome</td>
</tr>
<tr>
<td>P4</td>
<td>Screw</td>
<td>Red</td>
<td>14</td>
<td>London</td>
</tr>
<tr>
<td>P5</td>
<td>Cam</td>
<td>Blue</td>
<td>12</td>
<td>Paris</td>
</tr>
<tr>
<td>P6</td>
<td>Cog</td>
<td>Red</td>
<td>19</td>
<td>London</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>S#</th>
<th>P#</th>
<th>QTY</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>P1</td>
<td>300</td>
</tr>
<tr>
<td>S1</td>
<td>P2</td>
<td>200</td>
</tr>
<tr>
<td>S1</td>
<td>P3</td>
<td>400</td>
</tr>
<tr>
<td>S1</td>
<td>P4</td>
<td>200</td>
</tr>
<tr>
<td>S1</td>
<td>P5</td>
<td>100</td>
</tr>
<tr>
<td>S1</td>
<td>P6</td>
<td>100</td>
</tr>
<tr>
<td>S2</td>
<td>P1</td>
<td>300</td>
</tr>
<tr>
<td>S2</td>
<td>P2</td>
<td>400</td>
</tr>
<tr>
<td>S3</td>
<td>P2</td>
<td>200</td>
</tr>
<tr>
<td>S4</td>
<td>P2</td>
<td>200</td>
</tr>
<tr>
<td>S4</td>
<td>P4</td>
<td>300</td>
</tr>
<tr>
<td>S4</td>
<td>P5</td>
<td>400</td>
</tr>
</tbody>
</table>

- Query:
  - 列出住在London的供應商名字?
  - Get the total number of suppliers.
  - Total QTY of SP?
  - Get supplier names for suppliers who supply part P2?
  - Get supplier names for suppliers who supply red color parts?
  - …

---

Wei-Pang Yang, Information Management, NDHU

Unit 1 Introduction to DBMS

1-26
Retrieval Operations

- Easy to retrieve information!

- Get color and city for "non-Paris" parts with weight greater than ten.

```sql
SELECT P.COLOR, P.CITY 
FROM P 
WHERE P.CITY <> 'Paris' 
AND P.WEIGHT > 10;
```

- DISTINCT

```sql
SELECT DISTINCT P.COLOR, P.CITY 
FROM P 
WHERE P.CITY <> 'Paris' 
AND P.WEIGHT > 10;
```

<table>
<thead>
<tr>
<th>P#</th>
<th>PNAME</th>
<th>COLOR</th>
<th>WEIGHT</th>
<th>CITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>Nut</td>
<td>Red</td>
<td>12</td>
<td>London</td>
</tr>
<tr>
<td>P2</td>
<td>Bolt</td>
<td>Green</td>
<td>17</td>
<td>Paris</td>
</tr>
<tr>
<td>P3</td>
<td>Screw</td>
<td>Blue</td>
<td>17</td>
<td>Rome</td>
</tr>
<tr>
<td>P4</td>
<td>Screw</td>
<td>Red</td>
<td>14</td>
<td>London</td>
</tr>
<tr>
<td>P5</td>
<td>Cam</td>
<td>Blue</td>
<td>12</td>
<td>Paris</td>
</tr>
<tr>
<td>P6</td>
<td>Cog</td>
<td>Red</td>
<td>19</td>
<td>London</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>COLOR</th>
<th>CITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>Red</td>
<td>London</td>
</tr>
<tr>
<td>Blue</td>
<td>Rome</td>
</tr>
<tr>
<td>Red</td>
<td>London</td>
</tr>
<tr>
<td>Red</td>
<td>London</td>
</tr>
</tbody>
</table>
Retrieval Operations  (cont.)

- For all parts, get the part number and the weight of that part in grams.

  \[ \text{SELECT P.P#, P.WEIGHT * 454 AS GMWT FROM P;} \]

- Get the maximum and minimum quantity for part P2.

  \[ \text{SELECT MAX (SP.QTY) AS MAXQ, MIN (SP.QTY) AS MINQ FROM SP WHERE SP.P# = 'P2';} \]

- For each part supplied, get the part number and the total shipment quantity.

  \[ \text{SELECT SP.P#, SUM (SP.QTY) AS TOTQTY FROM SP GROUP BY SP.P#;} \]
Retrieval Operations (cont.)

- Get part numbers for all parts supplied by more than one supplier.

```
SELECT SP.P#
FROM SP
GROUP BY SP.P#
HAVING COUNT (SP.S#) > 1;
```

- Get supplier names for suppliers who supply part P2.

```
SELECT DISTINCT S.SNAME
FROM S
WHERE S.S# IN
  (SELECT SP.S#
   FROM SP
   WHERE SP.P# = 'P2');
```

---

**<e.g.> Supplier-and-Parts Database**

<table>
<thead>
<tr>
<th>S#</th>
<th>SNAME</th>
<th>STATUS</th>
<th>CITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>Smith</td>
<td>20</td>
<td>London</td>
</tr>
<tr>
<td>S2</td>
<td>Jones</td>
<td>10</td>
<td>Paris</td>
</tr>
<tr>
<td>S3</td>
<td>Blake</td>
<td>30</td>
<td>Paris</td>
</tr>
<tr>
<td>S4</td>
<td>Clark</td>
<td>20</td>
<td>London</td>
</tr>
<tr>
<td>S5</td>
<td>Adams</td>
<td>30</td>
<td>Athens</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SP</th>
<th>S#</th>
<th>P#</th>
<th>QTY</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>P1</td>
<td>300</td>
<td></td>
</tr>
<tr>
<td>S1</td>
<td>P2</td>
<td>200</td>
<td></td>
</tr>
<tr>
<td>S1</td>
<td>P3</td>
<td>400</td>
<td></td>
</tr>
<tr>
<td>S1</td>
<td>P4</td>
<td>200</td>
<td></td>
</tr>
<tr>
<td>S1</td>
<td>P5</td>
<td>100</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>P#</th>
<th>PNAME</th>
<th>COLOR</th>
<th>WEIGHT</th>
<th>CITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>Nut</td>
<td>Red</td>
<td>12</td>
<td>London</td>
</tr>
<tr>
<td>P2</td>
<td>Bolt</td>
<td>Green</td>
<td>17</td>
<td>Paris</td>
</tr>
<tr>
<td>P3</td>
<td>Screw</td>
<td>Blue</td>
<td>17</td>
<td>Rome</td>
</tr>
<tr>
<td>P4</td>
<td>Screw</td>
<td>Red</td>
<td>14</td>
<td>London</td>
</tr>
<tr>
<td>P5</td>
<td>Cam</td>
<td>Blue</td>
<td>12</td>
<td>Paris</td>
</tr>
<tr>
<td>P6</td>
<td>Cog</td>
<td>Red</td>
<td>19</td>
<td>London</td>
</tr>
</tbody>
</table>
Why Database?

- Easy to retrieve information!
- Redundancy can be reduced
- Inconsistency can be avoided
- The data can be shared
- Standards can be enforced
- Security restrictions can be applied
- Integrity can be maintained
- Provision of *data independence* objective!
- Database Growth Fast!
Why Database: Redundancy can be reduced

Customer
- Customer No.
- **Customer Name**
- Customer Addr.
- Social Security ID

Invoice
- Customer No.
- **Customer Name**
- Part No.
- Quantities
- **Unit Price**

Inventory
- Part No.
- Part Description
- **Unit Price**
- Supplier
- Quantities Remain
- Quantities Ordered

Parts
- Part No.
- Part Description
- Supplier
- Quantities Ordered
- **Customer Name**
- **Unit Price**

Why Database: Inconsistency can be avoided
Why Database: The data can be shared

Database System: Data Integrated and Shared

Why Database: Standards can be enforced
Why Database: Security restrictions can be applied

- <e.g.1> [GRANT]
  
  GRANT SELECT ON TABLE S TO CHARLEY;
  GRANT SELECT, UPDATE (STATUS, CITY) ON TABLE S TO JUDY, JACK, JOHN;
  GRANT ALL ON TABLE S, P, SP TO FRED, MARY;
  GRANT SELECT ON TABLE P TO PUBLIC;
  GRANT INDEX ON TABLE S TO PHIL;
Why Database: Integrity can be maintained

- Consider Supplier-and-Parts Database,

  Assume the STATUS should always be positive value.

<table>
<thead>
<tr>
<th>S#</th>
<th>SNAME</th>
<th>STATUS</th>
<th>CITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>Smith</td>
<td>20</td>
<td>London</td>
</tr>
<tr>
<td>S2</td>
<td>Jones</td>
<td>10</td>
<td>Paris</td>
</tr>
<tr>
<td>S3</td>
<td>Blake</td>
<td>30</td>
<td>Paris</td>
</tr>
<tr>
<td>S4</td>
<td>Clark</td>
<td>20</td>
<td>London</td>
</tr>
<tr>
<td>S5</td>
<td>Adams</td>
<td>30</td>
<td>Athens</td>
</tr>
</tbody>
</table>

- CREATE INTEGRITY RULE

CREATE INTEGRITY RULE R1
  ON INSERT S.STATUS,
  UPDATE S.STATUS;
  CHECK FORALL S ( S.STATUS > 0 )
  ELSE REJECT;
Why Database: Provision of data independence

Database management system (DBMS)

- Conceptual schema
- External/conceptual mapping A
- External/conceptual mapping B
- Conceptual/internal mapping
- Stored database (Internal View)
- DDL (Data Definition Language)
- Dictionary (e.g., system catalog)
- DBA (Build and maintain schemas and mappings)

User A1
Host Language + DSL

User A2
Host Language + DSL

User B1
Host Language + DSL

User B2
Host Language + DSL

User B3
Host Language + DSL

External View @ # &

External schema A

External schema B

DSL (Data Sub. Language) e.g., SQL

C, C++
Data Independence

- Application Program
  ➔ Data Structure

- Immunity of application to change in storage structure and access strategy.
Data Dependence vs. Data Independence

- **Data Dependent**
  - e.g. SELECT CITY FROM S WHERE ITEM = 'X';
  - **Linked list:** TOP
    - S → s1 → s2 → ... → sn
    - Top if item = TOP . item then ...........
  - **Tree:**
    - if item < root.data then root := root . left ...........
  - **Array:**
    - if A[I] = item then ............

- **Storage structure changed ➔ program changed**
1.4 An Architecture for a Database System
Architecture for a Database System: view 1

Diagram:

DBMS

Query

Language Processor

Optimizer

Operation Processor

Access Method

File Manager

Database
Components of Database System

- **Query Processor**
  - Helps to simplify to access data
  - High-level view
  - Users are not be burdened unnecessarily with the physical details

- **Storage Manager**
  - Require a large amount of space
  - Can not store in main memory
  - Disk speed is slower
  - Minimize the need to move data between disk and main memory

**Goal of a DBMS:** provides a way to **store** and **retrieve** data that is both **convenient** and **efficient**.
Architecture for a Database System: view 2

User A1
Host Language + DSL

User A2
Host Language + DSL

User B1
Host Language + DSL

User B2
Host Language + DSL

User B3
Host Language + DSL

External View
@ # &

External/conceptual mapping A

Conceptual schema

External View

External/conceptual mapping B

Conceptual View

Conceptual/internal mapping

Database management system (DBMS)

Dictionary

e.g. system catalog

DSL (Data Sub. Language)
e.g. SQL

C, C++

External View

Stored database (Internal View)

DBA
(Build and maintain schemas and mappings)

Storage structure definition (Internal schema)
Overall System Structure
Data Dictionary in DBMS

```
CREATE TABLE S
(S# CHAR (5) NOT NULL,
 SNAME CHAR (20),
 STATUS SMALLINT,
 CITY CHAR (15));
```

```
SELECT S.SNAME
FROM S
WHERE S.S.CITY = "London";
```
Figure 2.2 An example of the difference between physical and logical data organization.
Figure 6.2 Two different programmers require different files. The top of this diagram shows their subschemas. Both subschemas are derived from the schema in Fig. 6.1 shown at the bottom of this diagram.
1.5 Data Models
Data Models:

- Hierarchical Data Model
- Network Data Model
- Relational Data Model
- Object-Oriented Data Model
- ...

...
Hierarchical Data Model

"Data Pile"

<table>
<thead>
<tr>
<th>course#</th>
<th>name</th>
<th>prereq.1</th>
<th>prereq.2</th>
<th>off.1</th>
<th>92/9/6</th>
<th>EE108</th>
<th>T003</th>
<th>Yang</th>
<th>..........</th>
</tr>
</thead>
</table>

Course

- Course#
- Title

PreReq

- Prereq.#

Offering

- Off.#
- Date
- Location

Teacher

- Emp.#
- Name

Student

- Emp.#
- Name
- Grade
Network Data Model

S

P

S-SP

P-SP

SP

QTY

(Links)
The Network Model: Sample Values
Relational Data Model: [Codd70]

- System R
- DB2
- INGRES
- Oracle
- Informix
- ACCESS
- mySQL
- ...

\[\text{<e.g.>}\] Supplier-and-Parts Database

<table>
<thead>
<tr>
<th>S#</th>
<th>SNAME</th>
<th>STATUS</th>
<th>CITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>Smith</td>
<td>20</td>
<td>London</td>
</tr>
<tr>
<td>S2</td>
<td>Jones</td>
<td>10</td>
<td>Paris</td>
</tr>
<tr>
<td>S3</td>
<td>Blake</td>
<td>30</td>
<td>Paris</td>
</tr>
<tr>
<td>S4</td>
<td>Clark</td>
<td>20</td>
<td>London</td>
</tr>
<tr>
<td>S5</td>
<td>Adams</td>
<td>30</td>
<td>Athens</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SP</th>
<th>S#</th>
<th>P#</th>
<th>QTY</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>P1</td>
<td></td>
<td>300</td>
</tr>
<tr>
<td>S1</td>
<td>P2</td>
<td></td>
<td>200</td>
</tr>
<tr>
<td>S1</td>
<td>P3</td>
<td></td>
<td>400</td>
</tr>
<tr>
<td>S1</td>
<td>P4</td>
<td></td>
<td>200</td>
</tr>
<tr>
<td>S1</td>
<td>P5</td>
<td></td>
<td>100</td>
</tr>
<tr>
<td>S1</td>
<td>P6</td>
<td></td>
<td>100</td>
</tr>
<tr>
<td>S2</td>
<td>P1</td>
<td></td>
<td>300</td>
</tr>
<tr>
<td>S2</td>
<td>P2</td>
<td></td>
<td>400</td>
</tr>
<tr>
<td>S3</td>
<td>P2</td>
<td></td>
<td>200</td>
</tr>
<tr>
<td>S4</td>
<td>P2</td>
<td></td>
<td>200</td>
</tr>
<tr>
<td>S4</td>
<td>P4</td>
<td></td>
<td>300</td>
</tr>
<tr>
<td>S4</td>
<td>P5</td>
<td></td>
<td>400</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>P#</th>
<th>PNAME</th>
<th>COLOR</th>
<th>WEIGHT</th>
<th>CITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>Nut</td>
<td>Red</td>
<td>12</td>
<td>London</td>
</tr>
<tr>
<td>P2</td>
<td>Bolt</td>
<td>Green</td>
<td>17</td>
<td>Paris</td>
</tr>
<tr>
<td>P3</td>
<td>Screw</td>
<td>Blue</td>
<td>17</td>
<td>Rome</td>
</tr>
<tr>
<td>P4</td>
<td>Screw</td>
<td>Red</td>
<td>14</td>
<td>London</td>
</tr>
<tr>
<td>P5</td>
<td>Cam</td>
<td>Blue</td>
<td>12</td>
<td>Paris</td>
</tr>
<tr>
<td>P6</td>
<td>Cog</td>
<td>Red</td>
<td>19</td>
<td>London</td>
</tr>
</tbody>
</table>
Object-Oriented Data Model

Person

name

owns

photo

Vehicle

video

model

manu.

Company

name

location

Employee

department

salary

face-image

name

hair-color

aggregation

inheritance

Vehicle

car-video

manu.

ownership

Vehicle

video

model

manu.

Company

name

location

Person

name

owns

photo

Vehicle

video

model

manu.

Company

name

location

Person

name

owns

photo

Vehicle

video

model

manu.

Company

name

location

Person

name

owns

photo

Vehicle

video

model

manu.

Company

name

location

Person

name

owns

photo

Vehicle

video

model

manu.

Company

name

location

Person

name

owns

photo

Vehicle

video

model

manu.

Company

name

location

Person

name

owns

photo

Vehicle

video

model

manu.

Company

name

location

Person

name

owns

photo

Vehicle

video

model

manu.

Company

name

location

Person

name

owns

photo

Vehicle

video

model

manu.

Company

name

location

Person

name

owns

photo

Vehicle

video

model

manu.

Company

name

location

Person

name

owns

photo

Vehicle

video

model

manu.

Company

name

location

Person

name

owns

photo

Vehicle

video

model

manu.

Company

name

location

Person

name

owns

photo

Vehicle

video

model

manu.

Company

name

location

Person

name

owns

photo

Vehicle

video

model

manu.
## Database Technology Trends

<table>
<thead>
<tr>
<th></th>
<th>1960s to Mid-1970s</th>
<th>1970s to Mid-1980s</th>
<th>Late 1980s</th>
<th>Future</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Data Model</strong></td>
<td>Network Hierarchical</td>
<td>Relational</td>
<td>Semantic Object-oriented Logic</td>
<td>Merging data models, knowledge representation, and programming languages</td>
</tr>
<tr>
<td><strong>Database Hardware</strong></td>
<td>Mainframes</td>
<td>Minis PCs</td>
<td>Faster PCs Workstations Database machines</td>
<td>Parallel processing Optical memories</td>
</tr>
<tr>
<td><strong>User Interface</strong></td>
<td>None Forms</td>
<td>Query languages - SQL, QUEL</td>
<td>Graphics Menus Query-by-forms</td>
<td>Natural language Speech input</td>
</tr>
<tr>
<td><strong>Program Interface</strong></td>
<td>Procedural</td>
<td>Embedded query language</td>
<td>4GL Logic programming</td>
<td>Integrated database and programming language</td>
</tr>
<tr>
<td><strong>Presentation and display processing</strong></td>
<td>Reports Processing data</td>
<td>Report generators Information and transaction processing</td>
<td>Business graphics Image output Knowledge processing</td>
<td>Generalized display managers Distributed knowledge processing</td>
</tr>
</tbody>
</table>
1.6 Establish/Design a Database System
PART II: 資料庫設計 (Database Design)

- 資料庫問題分析與架構規劃:
  - 若有一大量資料想利用DBMS建資料庫來管理。第一步要分析問題，找到使用者需求
  - 實體-關係模型(Entity-Relationship Model, 簡稱E-R Model)是一套資料庫的設計工具。我們可以利用E-R Model分析資料庫問題。它可以把真實世界中複雜的問題中的事物和關係轉化為資料庫中的資料架構
  - 由於利用實體-關係模型設計資料庫時，並不會牽涉到資料庫的操作、儲存方式等複雜的電腦運作。所以我們會把心力放在需求分析去規劃想要的資料庫，並以實體-關係圖(E-R Diagram)來呈現

- 資料庫的表格正規化:
  - 實體-關係圖很容易轉化為表格(Tables)，而資料庫就是由許多表格(tables)組成的
  - 這些表格要正規化(Normalization)才能避免將來操作時的異常現象發生

- 設計介面增刪查改資料庫:
  - 如何方便、又有效率的管理存取資料庫是使用者最關心的二個要素
  - 良好的介面設計，可以讓使用者方便的查詢、方便的新增、方便的刪除、方便的修改的處理資料庫
Database Design

- **Database Design** - The process of designing the general structure of the database:
  - Logical Design
  - Physical Design

- **Logical Design** – Deciding on the database schema.
  - To 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
Design Process

- **Phase I**
  - Specification of user requirement (with domain experts)

- **Phase II**
  - Conceptual design (unit 6)
  - Choose a data model
  - Design tables
  - Normalization (unit 7)

- **Phase III**
  - Specification of functional requirements

- **Phase IV**
  - User interface design (unit 8)
  - Implementation
Contents of PART II: 資料庫設計

- Unit 6  Database Design and the E-R Model
- Unit 7  Normalization (表格正規化)
- Unit 8  User Interfaces (使用者介面)
- Unit 9 實作範例一:
- Unit 10 實作範例二:

References:
4. Cited papers
How to Establish a Database System?

- **STEP 1: Database Design**
  - Logical database vs. physical database
  - Collect data of applications
  - Analyze data to eliminate redundancy (using normalization theory and E-R Model…)
  - Describe data in the specific Data Model the DBMS use.
  - Describe each schema in DDL

- **STEP 2: Implementation**
  - schema
  - data

- **STEP 3: Evaluation and Correction (by DBA)**
  - tuning
  - statistical analysis
Components of a Database System

- DDL (Data Definition Language)
- DML (Data Manipulation Language)
- Data Dictionary
- Utility Routines
Components of a Database System: DDL

**DDL (Data Definition Language)**

CREATE TABLE S
(S# CHAR(5) NOT NULL,
SNAME CHAR(20) NOT NULL,
STATUS SMALLINT NOT NULL,
CITY CHAR(15) NOT NULL,
PRIMARY KEY (S#));

ALTER TABLE S ADD DISCOUNT SMALLINT;

DROP TABLE S

CREATE INDEX XSC ON S (CITY);
CREATE UNIQUE INDEX X ON S (S#);

DROP INDEX XSC;

<table>
<thead>
<tr>
<th>S#</th>
<th>SNAME</th>
<th>STATUS</th>
<th>CITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>Smith</td>
<td>20</td>
<td>London</td>
</tr>
<tr>
<td>S2</td>
<td>Jones</td>
<td>10</td>
<td>Paris</td>
</tr>
<tr>
<td>S3</td>
<td>Blake</td>
<td>30</td>
<td>Paris</td>
</tr>
<tr>
<td>S4</td>
<td>Clark</td>
<td>20</td>
<td>London</td>
</tr>
<tr>
<td>S5</td>
<td>Adams</td>
<td>30</td>
<td>Athens</td>
</tr>
</tbody>
</table>
Components of a Database System: DML

DML (Data Manipulation Language)

- SELECT S#, STATUS
  FROM S
  WHERE CITY='PARIS'

- UPDATE S
  SET STATUS= 2*STATUS
  WHERE CITY='LONDON'

- DELETE
  FROM S
  WHERE S#='S5'

- INSERT
  INTO S (S#, SNAME, STATUS, CITY)
  VALUES('S6','TSENG',100,'HSINCHU')
Components of a Database System: Data Dictionary

SELECT S.SNAME
FROM S
WHERE S.S.CITY = "London";

CREATE TABLE S
(S# CHAR (5) NOT NULL,
SNAME CHAR (20),
STATUS SMALLINT,
CITY CHAR (15));
Components of a Database System: Utility Routines

- Loading Routines
- Reorganization Routines
- Journalizing routines (log)
- Database Dump Routines
- Recovery Routines
- Statistical Analysis Routines
- ...

Unit 1  Introduction to DBMS
1.7 Extending Database Technology
Extending Database Technology

• Expert Database Systems
  Knowledge-base Management Systems
  AI + DB

• Image Database Systems
  Intelligent Pictorial Databases
  Image + DB

• Object-Oriented Database Systems
  OO Programming + DB

• Multimedia Database
  Text + Voice + Image + …..+ DB

• Multidatabases
  Integrate heterogeneous /homogeneous
database systems
Extending Database Technology (cont.)

- Real-time Database Systems
  Real-time Tech. + DB

- Video Database Systems
  MPEG + DB

- Digital Library
  Library + DB

- Bioinformatics Database Systems
  Biological + DB

- ...

Wei-Pang Yang, Information Management, NDHU
Unit 1 Introduction to DBMS
Distributed Databases

- Distributed database is a database that is not stored in its entirety at a single physical location, but rather is spread across a network of computers.

< e.g.>
Distributed Databases (cont.)

- **Advantages:**
  - efficiency of local processing
  - data sharing

- **Disadvantages:**
  - communication overhead
  - implementation difficulties

- **Reference:**
  S. Ceri and G. Pelagatti
  "Distributed Databases: principles and systems"
Multi-Database/Heterogeneous Database

- semantic inconsistency
- data incompleteness
- global schema
DB + AI

Database

AI

Query

DBMS

Language Processor

Query Optimizer

Operator Processor

Access Method

File Manager

Distributed DB design

Logical DB design

Knowledge Base

Knowledge Base

Knowledge Base

Knowledge Base

Unit 1 Introduction to DBMS
KBMS

A Combined Model:
Logic Programming + Relational DB

Three layers:

- **User Program**

  Knowledge management program

  - IDB:
    - `ancestor(X,Y):- parent(X,Y)`
    - `ancestor(X,Y) :- parent(X,Z), ancestor(Z,Y)`
    - `parent(X,Y):-edb(father(X,Y))`  
    - `parent(X,Y):-edb(mother(X,Y))`  
    - `grandfather(X,Z):- father(X,Y) ^ father(Y,Z)`

  - EDB:
    - Father
      - Father
        - A
        - X
        - B
      - Son
        - B
        - C

- **Relational DB management program**

  - EDB:
  - Database Logic

Query:

- ? :- ancestor (taro, Y)
- ? :- grandfather (?, c)
A typical Document: MEMO [Woelk86, SIGMOD]

**MCC**

To: W. Kim  
From: D. Woolk  
Date: September 18, 1992  
Subject: Workstations

In the computer center of National Chiao-Tung University, there are a lot of workstations. There are HP RS serials, SUNs, Apollo, and so on. The students in NCTU learn to use workstation since they are freshmen. The configuration of the workstations follows:

In the course introduction to Computer Science, students do their homework's on workstations.
Use of a Database Management System in Design and Application

Database  Manufacturing

DBMS
Database Management System

Detailed Design  Design Release Control

Analysis Models  Preliminary Design

Fabrication Assembly Info  Test / Inspection

INTERFACE
Graphic Interface  Language Interface

APPLICATION
Design Analysis  Design Verification  Evaluation  Synthesis  Release

Manufacturing Planning  Production Control  Project Management
Fuzzy Database

- Fuzzy Query

\[<\text{e.g.}>\quad \text{SELECT} \quad \text{STUDENT\_NAME} \]
\[\text{FROM} \quad \text{STUDENT} \]
\[\text{WHERE} \quad \text{SEX} = \text{M} \]
\[\text{AND} \quad \text{HEIGH} = \text{TALLER} \]
\[\text{AND} \quad \text{WEIGH} = \text{SLIMMER} \]

\text{STUDENT:}

<table>
<thead>
<tr>
<th>NAME</th>
<th>SEX</th>
<th>HEIGHT</th>
<th>WEIGHT</th>
<th>IQ</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mary</td>
<td>F</td>
<td>158</td>
<td>55</td>
<td>High</td>
</tr>
<tr>
<td>Linda</td>
<td>F</td>
<td>165</td>
<td>55</td>
<td>Medium</td>
</tr>
<tr>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
</tbody>
</table>

\[<\text{e.g.}>\quad \text{SELECT} \quad \text{STUDENT\_NAME} \]
\[\text{FROM} \quad \text{STUDENT} \]
\[\text{WHERE} \quad \text{IQ} \geq 130 \]
1.8 Discussion and Remarks
Discussion and Remarks

- Advantages of database systems
  - Easy to retrieve information!
  - Redundancy can be reduced
  - Inconsistency can be avoided
  - Data can be shared
  - Standards can be enforced
  - Security restrictions can be applied
  - Integrity can be maintained
  - Provision of data independence

- Disadvantages of database systems
  - Database design and control are a complicated matter.
Contents of Part I: 入門與導論

- Unit 1 Introduction to DBMS
- Unit 2 DB2 and SQL
- Unit 3 The Relational Model
- Unit 4 The Hierarchical Model
- Unit 5 The Network Model

References:
3. Cited papers
Contents of PART II: 資料庫設計

- Unit 6  Database Design and the E-R Model
- Unit 7  Normalization (表格正規化)
- Unit 8  User Interfaces (使用者介面)
- Unit 9 實作範例一:
- Unit 10 實作範例二:

-----------------------------

References:
4. Cited papers
Contents of PART III: 進階探討

- Unit 11  Access Methods
- Unit 12  Database Recovery
- Unit 13  Concurrency Control
- Unit 14  Security and Integrity
- Unit 15  Query Optimization
- Unit 16  Distributed Database
- Unit 17  More on E-R Model
- Unit 18  More on Normalization
- Unit 19  More on User Interfaces
- Unit 20  More on X?

References:
3. Cited papers
Contents of PART VI: 主題研究

- Unit 21  Object-Oriented Database
- Unit 22  Logic-Based Database
- Unit 23  Image Database
- Unit 24  Multimedia Database
- Unit 25  Real-Time Database
- Unit 26  Parallel Database
- Unit 27  Temporal Database
- Unit 28  Active Database
- Unit 29  Bioinformatics Database
- Unit 30  ....

References:
1. Cited papers
Study and Research on Databases

Level 5: Doing Research

Level 4: Survey Papers: Special Topics (Unit 21 - )

Level 3: DBMS: Advanced Topics
(Unit 11 – 20)
Date, Vol. 1, 2
Ullman

Level 2: DBMS: Fundamentals
(Unit 1 – 10)
Date, Vol. 1
Using mySQL

Level 1: Using DBMS
end of unit 1