資料庫
理論與實務
Access 2007

第 1 章

認識資料庫系統

本章提要

- 1 - 1 資料庫系統簡介
- 1 - 2 資料庫的類型
- 1 - 3 資料庫系統的處理架構
- 1 - 4 資料庫管理系統的基本功能

1-1 資料庫系統簡介

- 資料庫系統 (Database System) 是電腦化的資料儲存系統, 使用者可透過各種應用程式來存取其中的資料。
- 對公司或企業來說, 使用資料庫系統的好處如下：
  - 能透過電腦化的資料儲存及管理, 減少人力及空間的浪費。
  - 能迅速、即時地提供使用者所需要的資料, 大幅降低公司的成本。

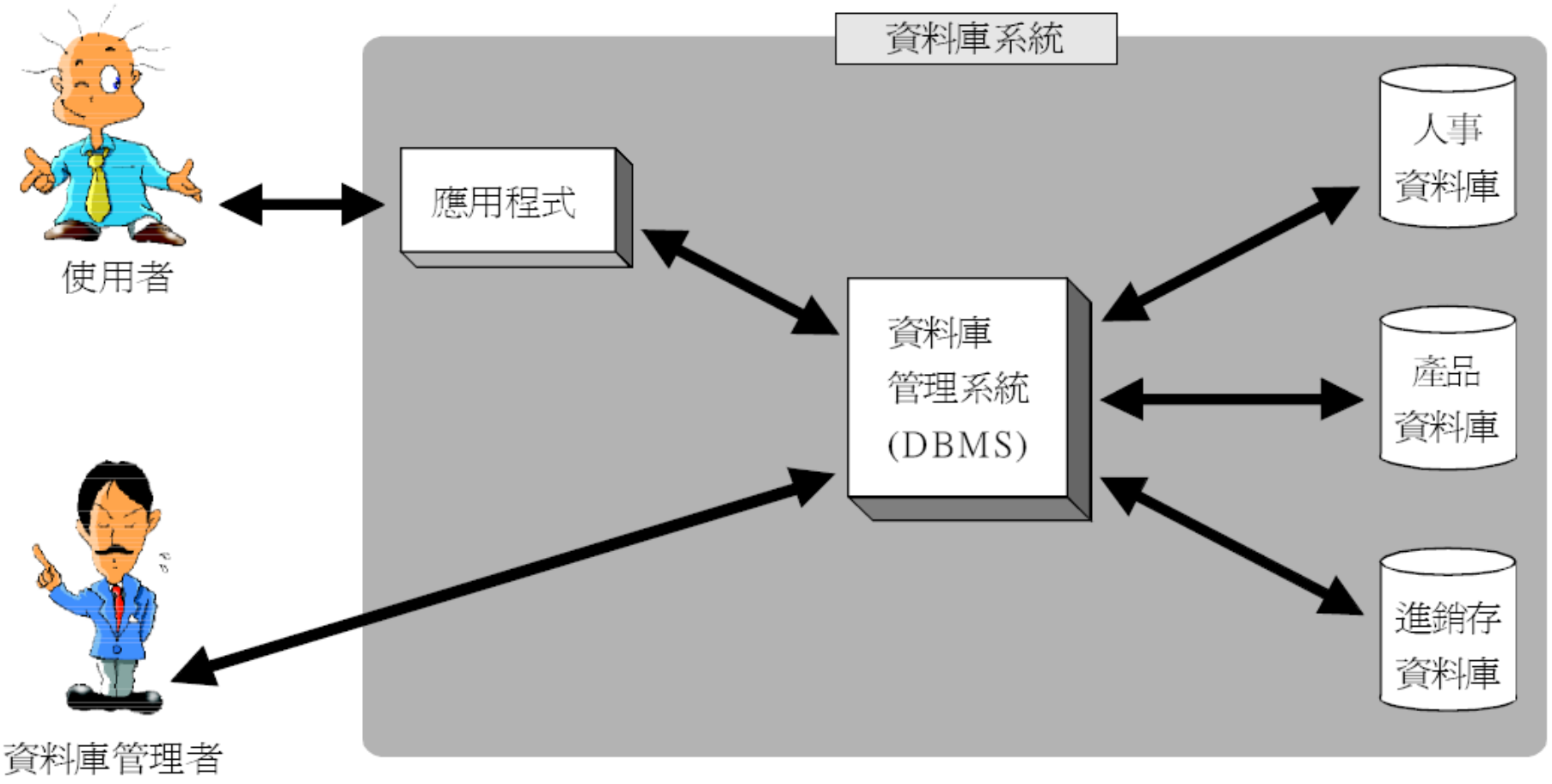
資料庫系統簡介

- 能集中管理公司所有的資料，並藉由設定使用者權限，加強資料的保密性及安全性。
- 可減少儲存重複的資料，相對地也加強了資料的一致性。

資料庫系統的組成

- 資料庫系統可分為三個部份：資料庫 (Database)、資料庫管理系統 (DataBase Management System, DBMS) 與應用程式 (Application)：

資料庫系統的組成



資料庫 (Database)

- 資料庫是儲存資料的地方。
- 一個資料庫系統中可以有多個資料庫, 每個資料庫都是經過整理的資料集合。
- 一般, 我們會將資料庫想像成是一個存放資料的容器, 但其真實型態其實是一個個的電子檔案 (file)。

資料庫管理系統 (DataBase Management System, DBMS)

- 資料庫管理系統則是指管理資料庫的軟體, 負責使用者與資料庫之間的溝通, 如存取資料庫中的資料、以及管理資料庫的各項事務等。
- Microsoft 的 Access, 還有許多用在大型資料庫系統上的 Microsoft SQL Server、Oracle、SyBase、Informix、MySQL、PostgreSQL... 等皆是資料庫管理系統。

應用程式 (Application)

- 應用程式是指自行開發的使用者介面, 因為並非每個使用者都會操作複雜的資料庫, 所以必須藉由另外設計的程式, 來提供較簡單且人性化的操作介面。
- 例如我們利用 Visual Basic 開發的人事管理系統、進銷存管理系統...等。
- 這些應用程式都必須透過資料庫管理系統才能存取及管理資料庫內的資料。

資料庫系統的使用者

- 資料庫系統從設計、建立、操作、到管理階段, 都會有不同的使用者參與, 我們可以大致區分出四種類型：
  - 資料庫設計者 (Database Designer)
  - 資料庫管理者 (DataBase Administrator, DBA)
  - 應用程式設計者 (Application Designer)
  - 一般使用者 (End user)

資料庫設計者 (Database Designer)

- 資料庫設計者負責整個資料庫系統的設計，依據使用者的需求設計適當的格式來存放資料；同時對於整個資料庫的使用者存取權限也需要做規劃。
- 設計完成後就可交由資料庫管理者負責管理維護的工作。
- 在一般中小型企業中，資料庫的設計者與管理者有可能就是同一個人；若是大型企業，則可能設計者是一組人，而管理者又是另外一組人。

資料庫管理者 (DataBase Administrator, DBA)

- 資料庫建好之後, 便可以交給資料庫管理者來負責管理及維護。
- DBA 最主要的任務, 就是要維護資料庫的有效運作, 並監督、記錄資料庫的操作狀況, 必要時還得修改資料庫的資料結構或各項設定, 以符合實際需求或提升運作效率。
- 由於資料庫中的資料對企業非常重要, 而資料庫系統難免會碰到人為疏失、硬體或作業系統的問題而損壞,

資料庫管理者 (DataBase Administrator, DBA)

- 所以 DBA 必須設定資料庫備份的方法和時機, 並且在資料庫受損時儘速讓資料庫回復原狀。
- 除此之外, DBA 也要負責資料庫的帳戶管理, 決定哪些人有權利登入資料庫, 哪些人有權執行哪些動作。
- 例如最基本的使用者可能只有查詢的權利, 需要輸入資料的使用者則具有寫入資料的權限... 等等。

應用程式設計者 (Application Designer)

- 應用程式設計者負責撰寫存取資料庫的用戶端應用程式, 讓使用者得以透過方便的操作介面來使用資料庫。
- 可用來開發應用程式的語言很多, 早期的程式設計師可能用 C、C++ 或 PASCAL 等語言, 現今的的程式設計師則多採用 Visual Basic、JAVA、C# 等程式語言。

一般使用者 (End user)

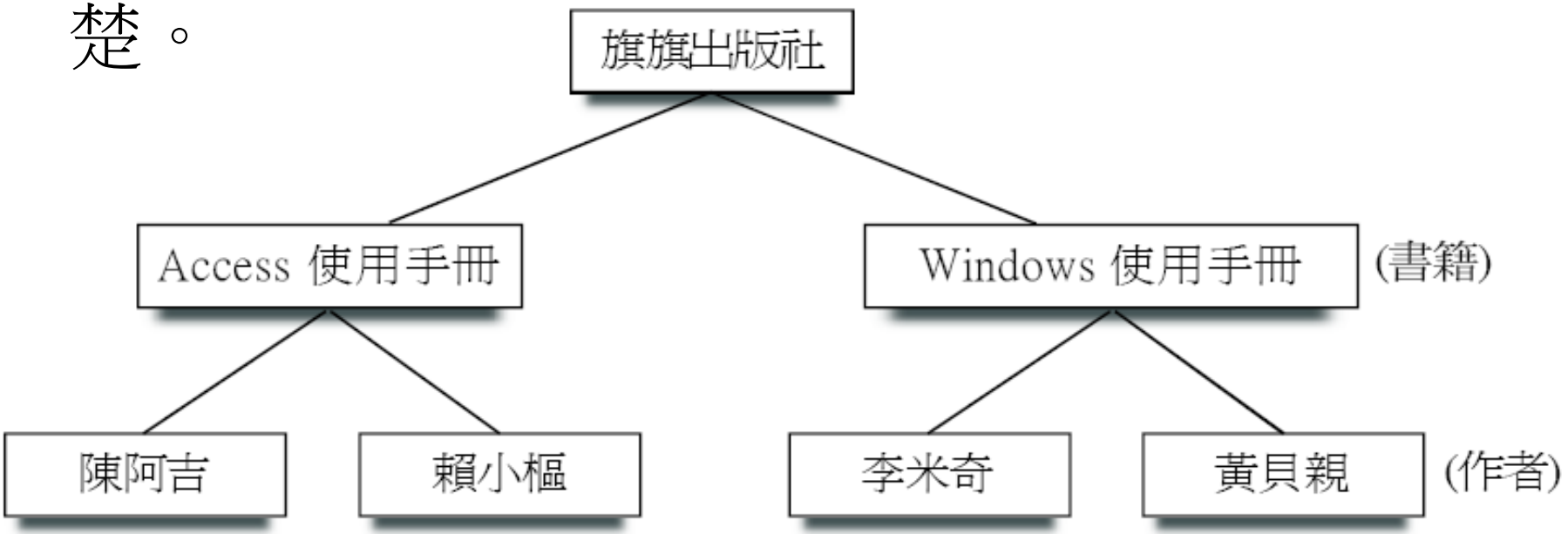
- 一般使用者就是真正經常在存取資料庫的使用者, 他們只需要學會用戶端的應用程式, 不需要擔心資料庫的維護或管理方面的任何問題。
- 若遇到問題, 只要請 DBA 處理即可。

1-2 資料庫的類型

- 前一節我們曾經提到資料庫是儲存資料的地方，就資料庫中儲存資料的架構來看，又可分為多種類型，
- 常見的有階層式資料庫、網狀式資料庫、關聯式資料庫及物件導向式資料庫，底下我們就針對這 4 種資料庫類型做簡單的介紹。

階層式資料庫 (Hierarchical Database)

- 階層式資料庫是採用樹狀的結構，將資料分門別類儲存在不同的階層下。
- 此類型的優點是資料結構類似金字塔，對於在同一類型中不同階層資料的描述非常簡單且清楚。



階層式資料庫 (Hierarchical Database)

- 以上圖為例, 我們可以很清楚地描述出版社和作者的關係。而它的缺點在於當資料的關係變得複雜時, 會造成管理及維護的不便。
- 例如：我們要描述學生及老師的關係, 一個學生可受教於多個老師, 而一個老師又能教導多個學生, 此種情況下, 資料重複出現的機率很高, 會造成管理及維護上的不便。
- 例如 IBM 公司在 1968 年推出的第一個資料庫管理系統—IMS 即屬於此類。

網狀式資料庫 (Network Database)

- 網狀式資料庫其實就是階層式資料庫的擴充, 我們可將每筆記錄想像成一個節點, 節點與節點間可以建立關聯 (也就是建立記錄和記錄間的關聯), 形成一個複雜的網狀架構。
- 它的優點是避免了階層式資料庫中資料重複的問題, 缺點是關聯比較複雜, 尤其當資料庫的內容愈來愈多時, 要維護之間的關聯性就會變得非常複雜。
- 例如 1970 年代, Computer Associates 發展的 IDMS 即是屬於此類的資料庫管理系統。

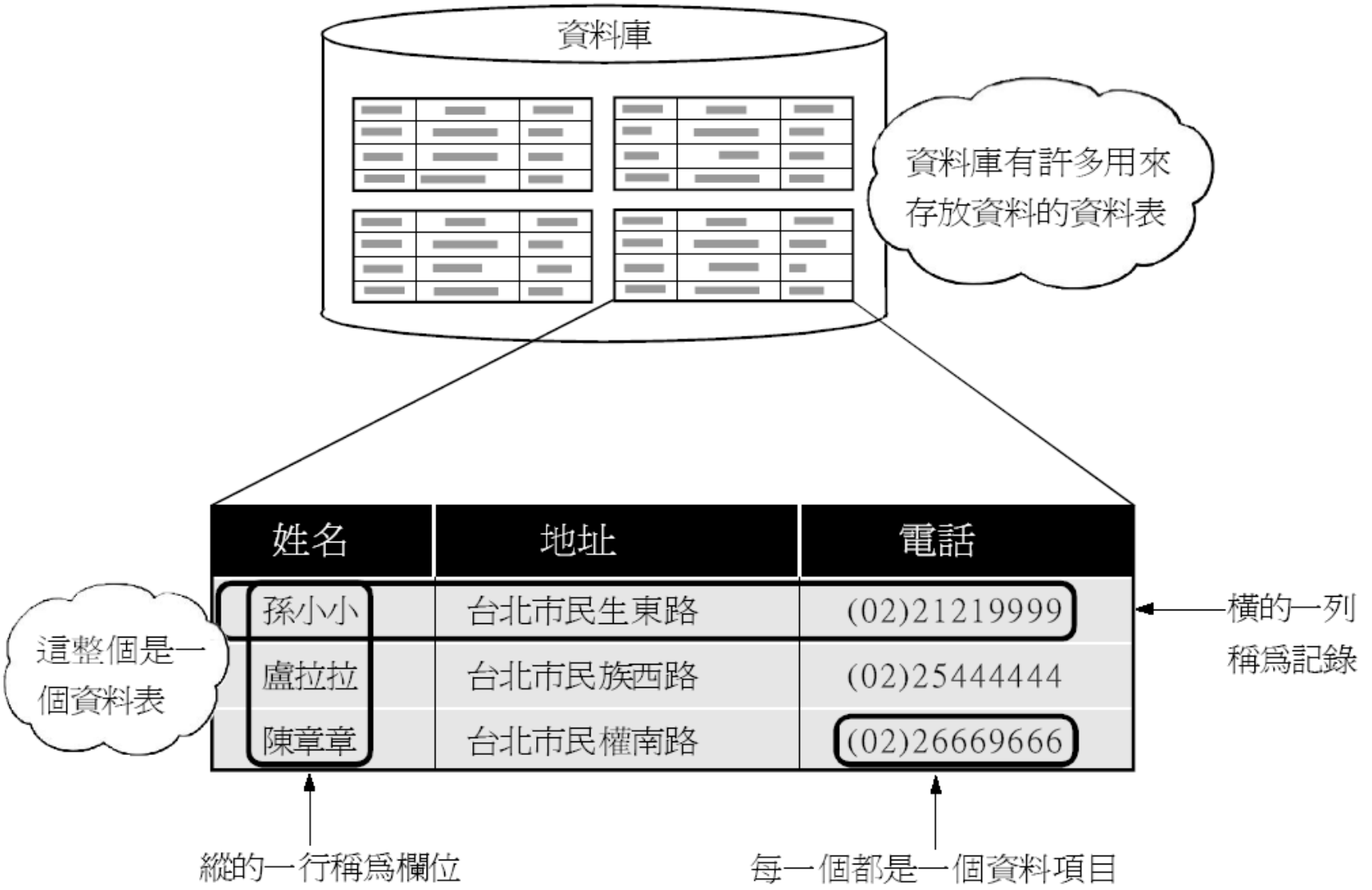
網狀式資料庫 (Network Database)

- 在上圖, 我們利用作者姓名可查到他寫過的書, 這些書又由哪些出版社出版的關係, 當記錄的數量增加, 彼此的關係就容易變得牽扯不清。

關聯式資料庫 (Relational Database)

- 關聯式資料庫是以 2 維的矩陣來儲存資料 (可以說是將資料儲存在表格的欄、列之中), 而儲存在欄、列裡的資料必會有所 “關聯”, 所以這種儲存資料的方式才會稱為關聯式資料庫, 而儲存資料的表格則稱為 “資料表” 。
- 舉例來說, 通訊錄資料表的每一欄可以劃分為『姓名』、『地址』、『電話』:

關聯式資料庫 (Relational Database)



關聯式資料庫 (Relational Database)

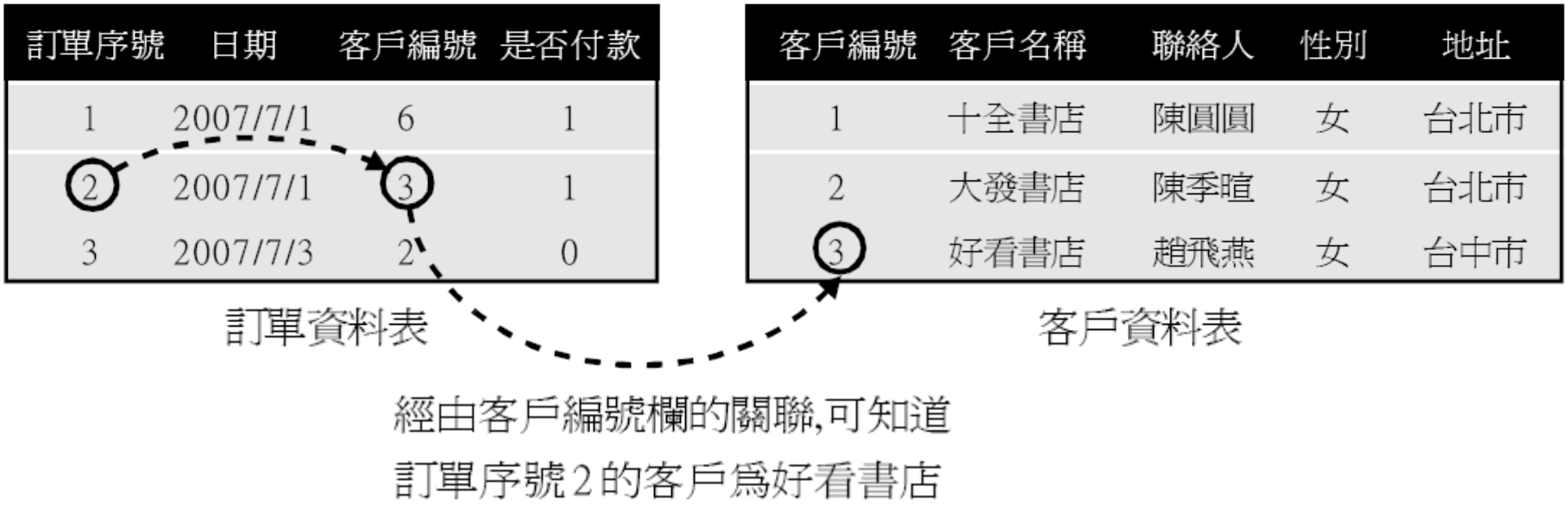
- 假如我們要從以上的資料表尋找 “盧拉拉”的地址, 則是由橫向的『盧拉拉』與縱向的『地址』, 交相關聯而得來：

姓名	地址	電話
孫小小	台北市民生東路	(02)21219999
盧拉拉	台北市民族西路	(02)25444444
陳章章	台北市民權東路	(02)26669666

- 除了儲存在資料表行與列會有所關聯, 關聯式資料庫裡面的資料表之間通常也會互有關聯。

關聯式資料庫 (Relational Database)

- 這種方式的優點是可以從一個資料表中的欄位，透過資料表的關聯，而找到另一個資料表中的資料：



關聯式資料庫 (Relational Database)

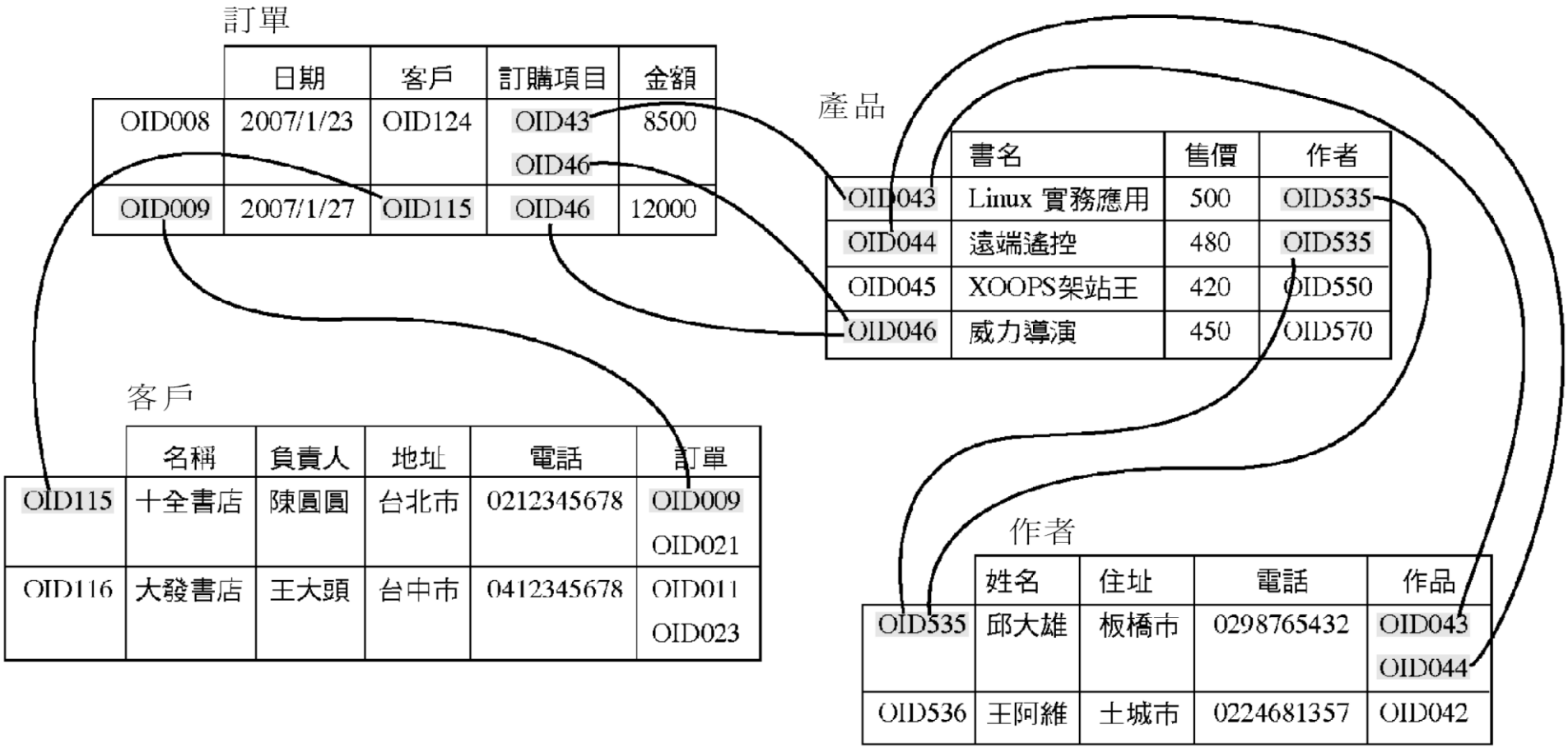
- 目前市場上是以關聯式資料庫使用最廣泛, 像 Microsoft SQL Server、SyBase、Informix、MySQL、PostgreSQL、Access...等, 都是屬於關聯式資料庫管理系統 (Relational DBMS, RDBMS)。

物件導向式資料庫 (Object-Oriented Database)

- 物件導向資料庫是以物件導向的方式來設計資料庫, 其中包含了物件的屬性、方法、類別、繼承等特性。
- 屬於這類的資料庫管理系統有 Computer Associates 公司的 Jasmine、Eastman Kodak 公司的 Alltalk、Servio 公司的 GemStone、O2 Technology 的 O2 ...等資料庫管理系統。
- 此外也有關聯式資料庫為主, 再於其上架設物件導向概念的資料庫, 如 PostgreSQL。

物件導向式資料庫 (Object-Oriented Database)

- 底下是一個物件導向式資料庫的結構示意圖：



物件導向式資料庫 (Object-Oriented Database)

- 上列的示意圖中有幾個重點, 說明如下：
  - 每一個橫列即為一個物件：
    - 以訂單為例, 每一個物件包含了日期、客戶、訂購項目、金額等屬性 (OID 是產生物件時的 ID, 不是物件的屬性, 說明如後)。
    - 這些屬性可以是文字資料、數值資料, 甚至是另一個物件, 而且一個屬性不必是唯一的值, 如上圖的訂單資料庫中, OID008 的物件, 其訂購項目屬性就包含 OID43 及 OID46 兩個物件。

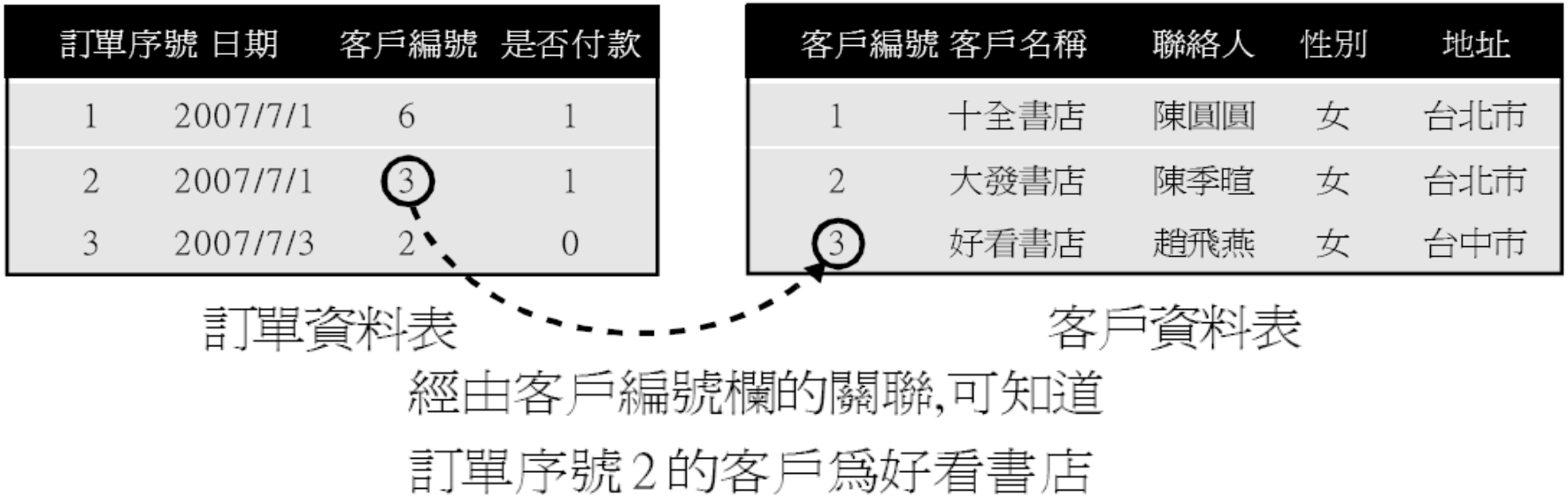
物件導向式資料庫 (Object-Oriented Database)

- 每個物件擁有唯一的 Object IDentity (OID)：
  - 同樣以訂單為例, 每個物件的第一欄就是物件的 OID。
  - OID 並不是資料庫設計者賦予的, 而是該物件成立時, 便自動產生一個 OID；要特別注意的是, OID 並不是物件的屬性, 實際上我們是看不到 OID 的。
  - 當物件內有包含其它物件時, 就能透過這個獨一無二的 OID 來快速找到對應用的物件。

物件導向式資料庫 (Object-Oriented Database)

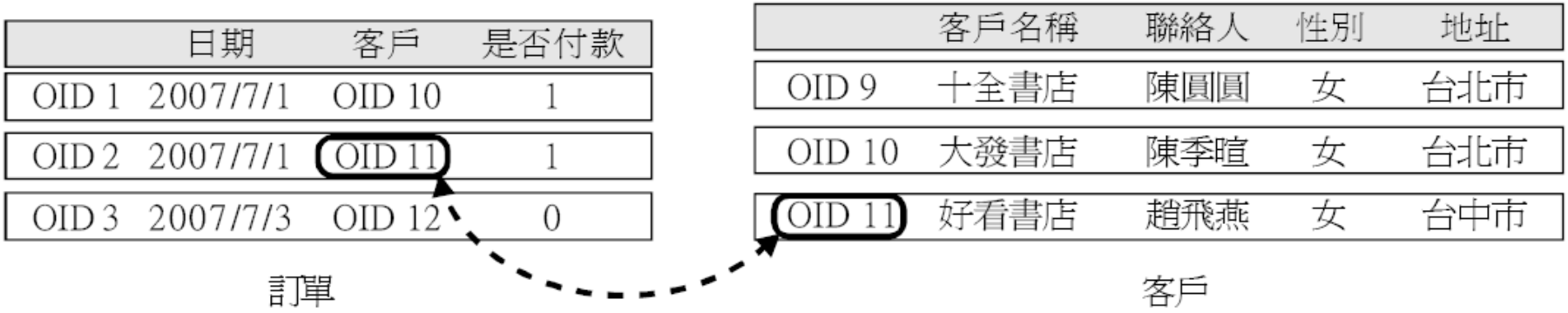
- 若以關聯式資料庫和物件導向式資料庫來做比較, 關聯式資料庫必須由資料庫設計者來設計、建立、及管理關聯。
- 但物件導向式資料庫中, 物件和物件之間的連繫, 是因其屬性而必然發生的。
- 我們先看下面這張關聯式資料庫的資料表：

物件導向式資料庫 (Object-Oriented Database)



- 由上圖可知, 兩個資料表是藉由客戶編號來達成關聯的, 而這個關聯性在關聯式資料庫中, 必須由設計者自行建立才會真正產生關聯。
- 接著看下面的物件導向式資料庫：

物件導向式資料庫 (Object-Oriented Database)



	日期	客戶	是否付款
OID 1	2007/7/1	OID 10	1
OID 2	2007/7/1	OID 11	1
OID 3	2007/7/3	OID 12	0

訂單

	客戶名稱	聯絡人	性別	地址
OID 9	十全書店	陳圓圓	女	台北市
OID 10	大發書店	陳季暄	女	台北市
OID 11	好看書店	趙飛燕	女	台中市

客戶

- 上圖中，兩個物件是透過 OID 來連繫起來的。
- 簡單地說，在關聯式資料庫中資料表間的關係必須靠設計者自行建立來產生關聯，而物件導向式資料庫中，各物件之間的關係則是在物件建立之時，便會自行連繫起來。

1-3 資料庫系統的處理架構

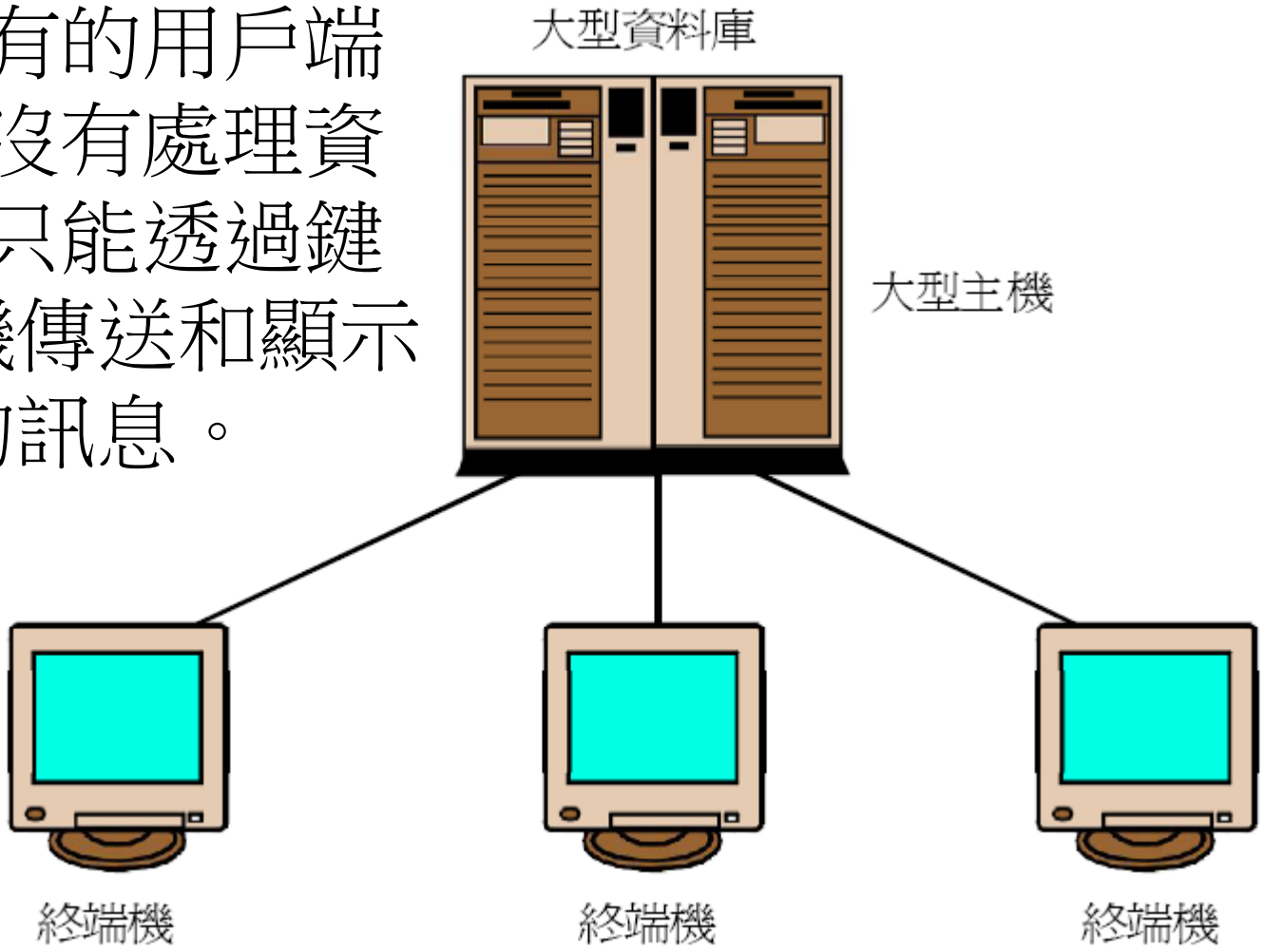
- 了解資料庫的類型後, 接下來我們要介紹如何部署您的資料庫系統, 通常我們會依照組織的規模、資料量的多寡、使用的人數及軟硬體的設備等條件來考量, 常見的有 4 種架構。
  - 單機架構
  - 大型主機 / 終端機架構
  - 主從式架構 (Client / Server)
  - 分散式架構

單機架構

- 在此架構中, 利用一台電腦完成所有的工作, 包含使用者存取資料、DBA 管理及維護資料庫...等, 適合在使用者少且資料量也不多的環境下使用, 例如小公司或個人使用者建立的資料庫。

大型主機 / 終端機架構

- 在這種架構中，由一台大型主機負責儲存及處理資料，所有的用戶端僅供操作，沒有處理資料的能力，只能透過鍵盤及終端機傳送和顯示大型主機的訊息。



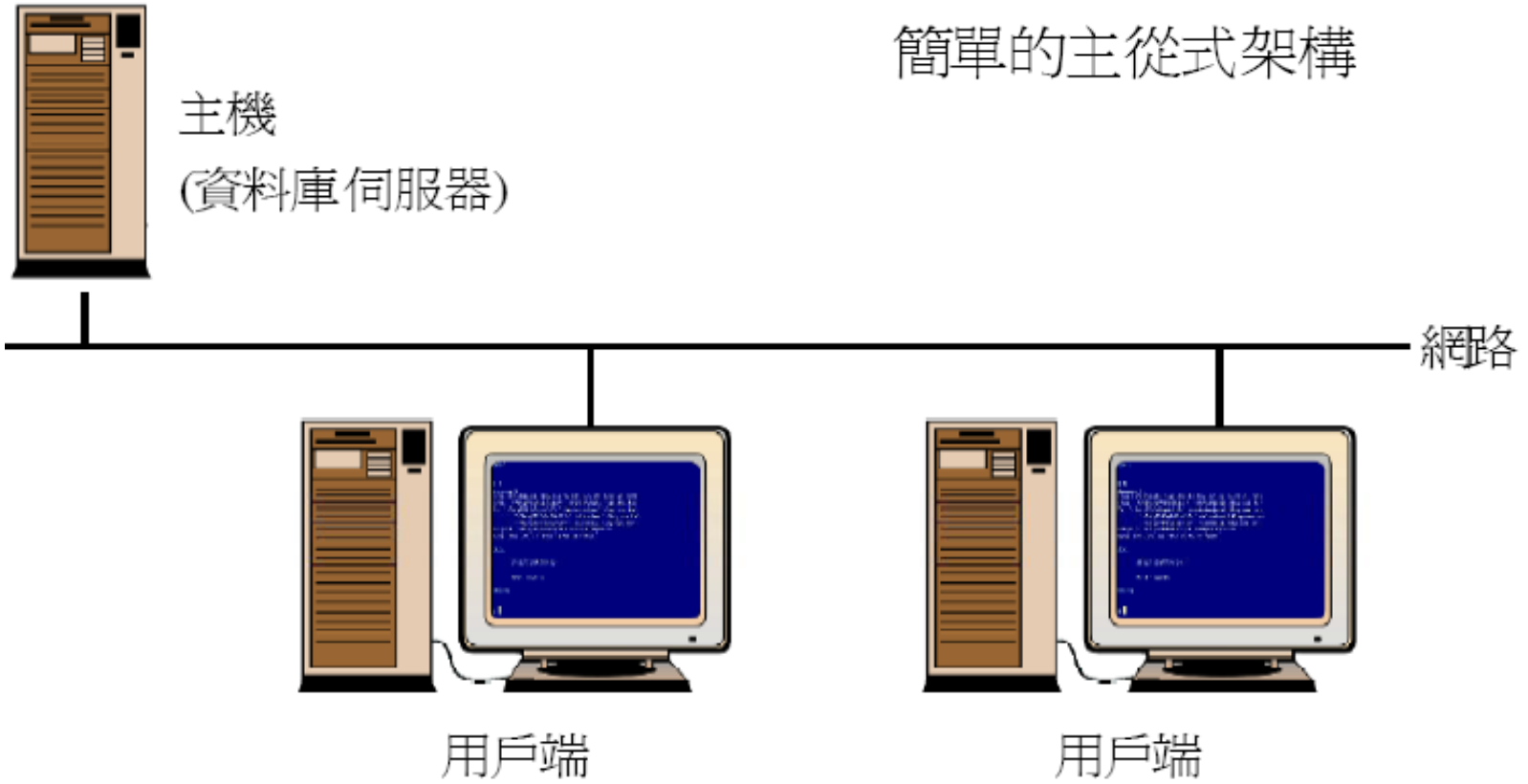
大型主機 / 終端機架構

- 這種 “集中式” 管理的優點在於主機完全掌控系統的資源, 所有管理及維護工作都只要針對主機即可, 環境較單純；
- 但缺點則是只有一台主機執行工作, 當連線的使用者增加時, 會因為處理的工作增加而降低執行的效率。
- 目前除了一些大型的機構外, 中小企業甚少使用此架構, 且因為大型主機的價格都很昂貴, 一般較無能力負擔。

主從式架構 (Client / Server)

- 近年來, 由於個人電腦的技術突飛猛進、軟體的功能愈來愈強以及網路的普遍, 因此不再由單一的大型主機來負責所有的工作。
- 基於分工的原則, 於是利用一台處理效能較強的電腦作為主機, 來維護資料庫及處理使用者提出的要求, 再利用使用者的個人電腦來分擔部分主機的工作 (例如提供操作介面及應用程式), 這就是主從式架構。

主從式架構 (Client / Server)

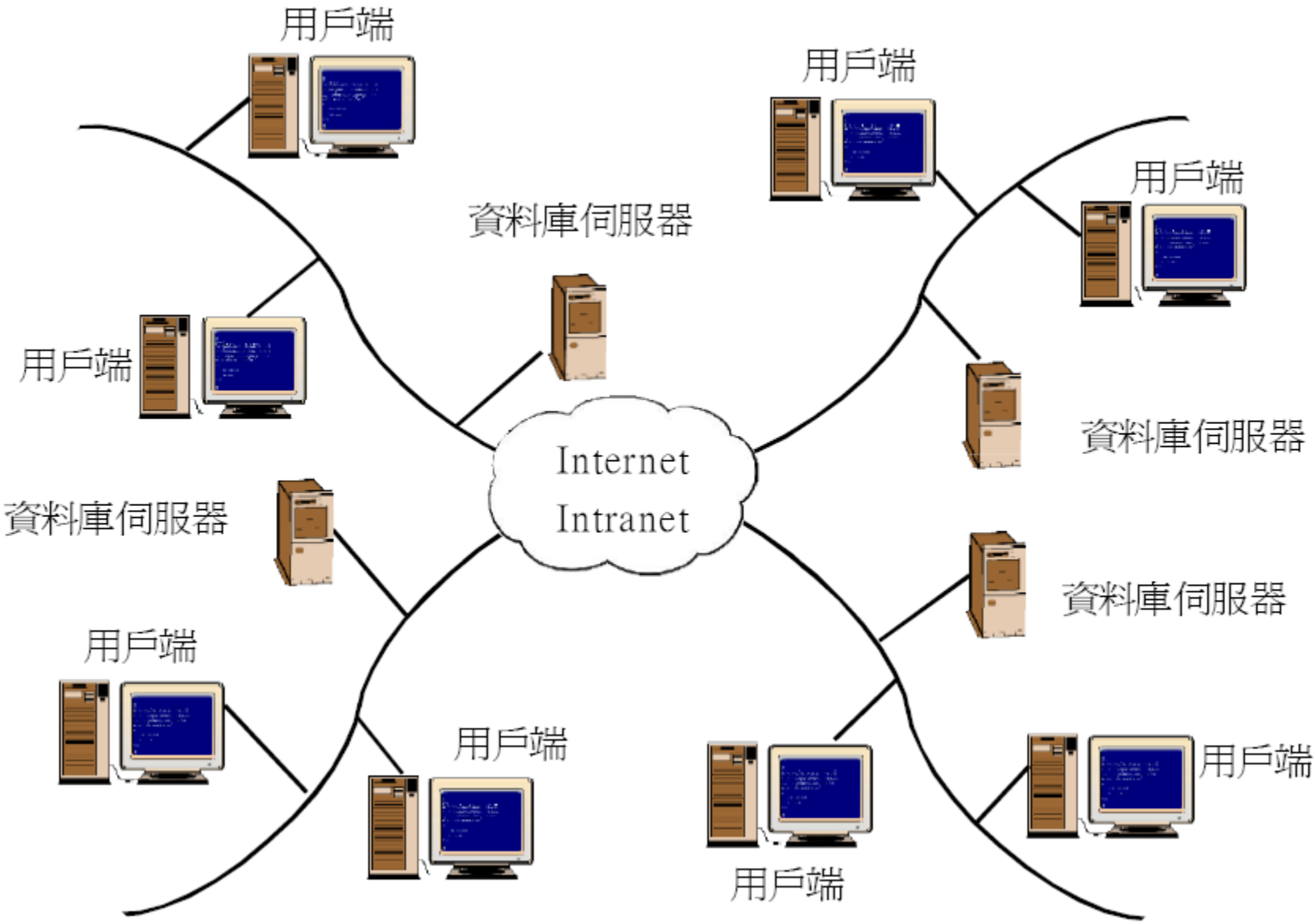


- 在這種架構下，主機能保留更多的效能來處理更多的使用者的連線。同時也不需要花費大量的金錢在購置大型主機上。

分散式架構

- 分散式架構是利用數台資料庫伺服器來分別處理使用者的連線, 其實這種架構也反映到現實社會中, 因為許多企業本身就是分散的, 不論是分散在各地的分公司或是總公司中的各部門。
- 基本上, 他們的資料都是各自獨立在各處。

分散式架構


用戶端
用戶端
資料庫伺服器
用戶端
用戶端
資料庫伺服器
Internet
Intranet
資料庫伺服器
資料庫伺服器
用戶端
用戶端
用戶端
用戶端

分散式架構

- 以上圖為例, 使用者透過網路存取資料, 這些資料可能分別來自不同的主機, 如此分擔了一台主機的工作, 執行起來效能會更佳。
- 看完上述資料庫架構, 您或許會問 Access 到底適合在哪一種架構中使用？
- 原則上 Access 適合在單機架構、主從式架構或分散式架構中使用, 由於它只是小型的資料庫管理系統, 能處理的資料量有限, 且處理效能較低於一般大型資料庫, 所以並不適合資料量大的大型公司。

1-4 資料庫管理系統的基本功能

- 在第 1-1 節中我們提過, 資料庫管理系統就是管理資料庫的軟體系統, 它們負責資料庫的建立、資料存取、權限設定、資料備份、操作的監督與記錄...等工作,
- 底下我們就再進一步詳述資料庫管理系統所應具備的基本功能。

資料庫管理系統的基本功能

- 資料定義：
  - DBMS 必須能夠充分定義並管理各種類型的資料項目。
  - 例如關聯式資料庫管理系統必須具備建立資料庫、資料表、定義各欄位的資料型別、限制, 以及資料表之間的關聯...等等的能力才行。
- 資料處理：
  - DBMS 必須提供使用者對資料庫存取的能力, 包括新增、修改、查詢與刪除的基本功能。
  - 有時 DBMS 提供的功能雖然完善, 但並不適合一般使用者操作, 這時就需要設計師另外撰寫用戶端的應用程式, 以供一般使用者操作。

資料庫管理系統的基本功能

- 資料安全：
  - DBMS 應該具備設定使用者帳戶、密碼及權限的功能，讓每一個使用者只能存取授權範圍內的資料，以防止機密資料外洩或資料遭受任何有意或無意的破壞。
- 資料備份：
  - DBMS 必須提供方便的資料備份功能，如此在資料庫不幸意外毀損時，還可還原到備份資料時的狀況，以減少損失。

資料庫管理系統的基本功能

- 此外, 維護資料庫的效率也非常重要, 尤其是在資料量很大或使用者很多的情況, 資料庫若因效率不佳而導致存取速度變慢, 亦會嚴重影響到操作人員的工作效率。