

資料庫設計(RDB)

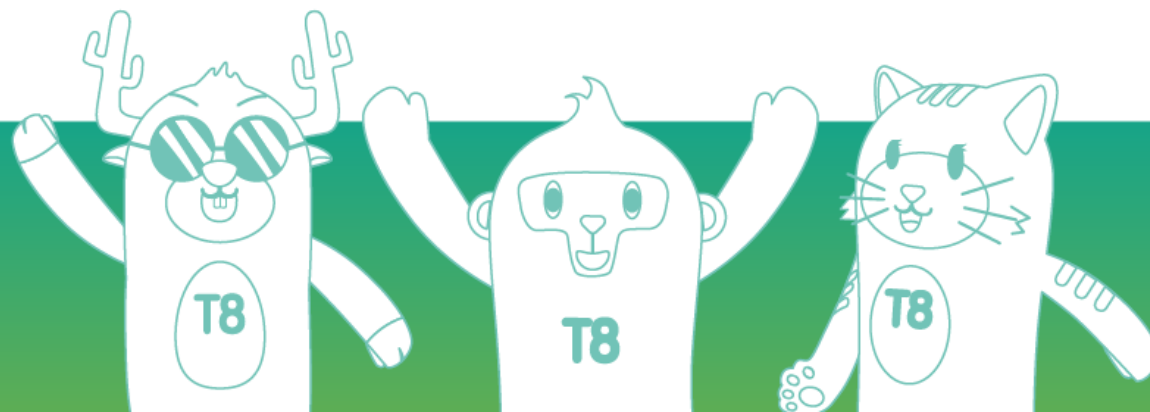
■ 授課講師 郭惠民

■ 教材編寫 郭惠民

緯育 *TibaMe*

即學・即戰・即就業

<https://www.tibame.com/>



- ◆ 模組1：關聯式資料庫設計程序
- ◆ 模組2：ER模型
- ◆ 模組3：ER模型實體塑模
- ◆ 模組4：ER模型關係塑模
- ◆ 模組5：正規化
- ◆ 模組6：資料模型轉換及實體資料庫設計

教材編著者 & 授課講師介紹



授課講師

郭惠民

isaachmkuo@gmail.com

老師的話

學習本課程須知

先備知識

任一種關聯式資料庫之架構與SQL指令之操作

學習目標

- A. 了解關聯式資料庫特性及設計之步驟
- B. 了解概想資料庫設計之原理及ER模型之符號
- C. 了解關聯正規化原則與處理方法
- D. 了解資料模型轉換及實體資料庫設計之原理與步驟

模組1：關聯式資料庫設計程序

1-1 關聯式資料庫之特性

1-2 關聯式資料庫設計步驟簡介

1-3 概想資料庫設計與實體資料庫設計之比較

1.使用SQL標準語言

- 關聯式資料庫使用 SQL 或結構式查詢語言做為主要的通訊介面。美國國家標準協會 (ANSI) 在 1986 年將 SQL 納入標準。所有常見的關聯式資料庫引擎都支援標準 **ANSI SQL**，而且有些引擎還提供 ANSI SQL 延伸功能，以支援該引擎的專屬功能。
- 您可以利用 SQL 新增、更新、刪除多個橫列的資料、擷取資料子集供交易處理和分析應用程式使用，以及管理資料庫的所有面向。

2.強調資料完整性 (Data Integrity)

- 資料完整性是指 **資料整體完備無缺、準確而且一致**。關聯式資料庫利用一套限制條件讓資料庫達成資料完整性。其中包括主索引鍵、外部索引鍵、「非空白值」限制條件、「唯一」限制條件、「預設」限制條件、「檢查」限制條件。
- 這些完整性限制條件對表格中的資料實施商業規則，確保資料的準確性和可靠性。除此之外，大部分關聯式資料庫也允許在觸發中嵌入自訂程式碼，當資料庫發生某個動作時便會執行。

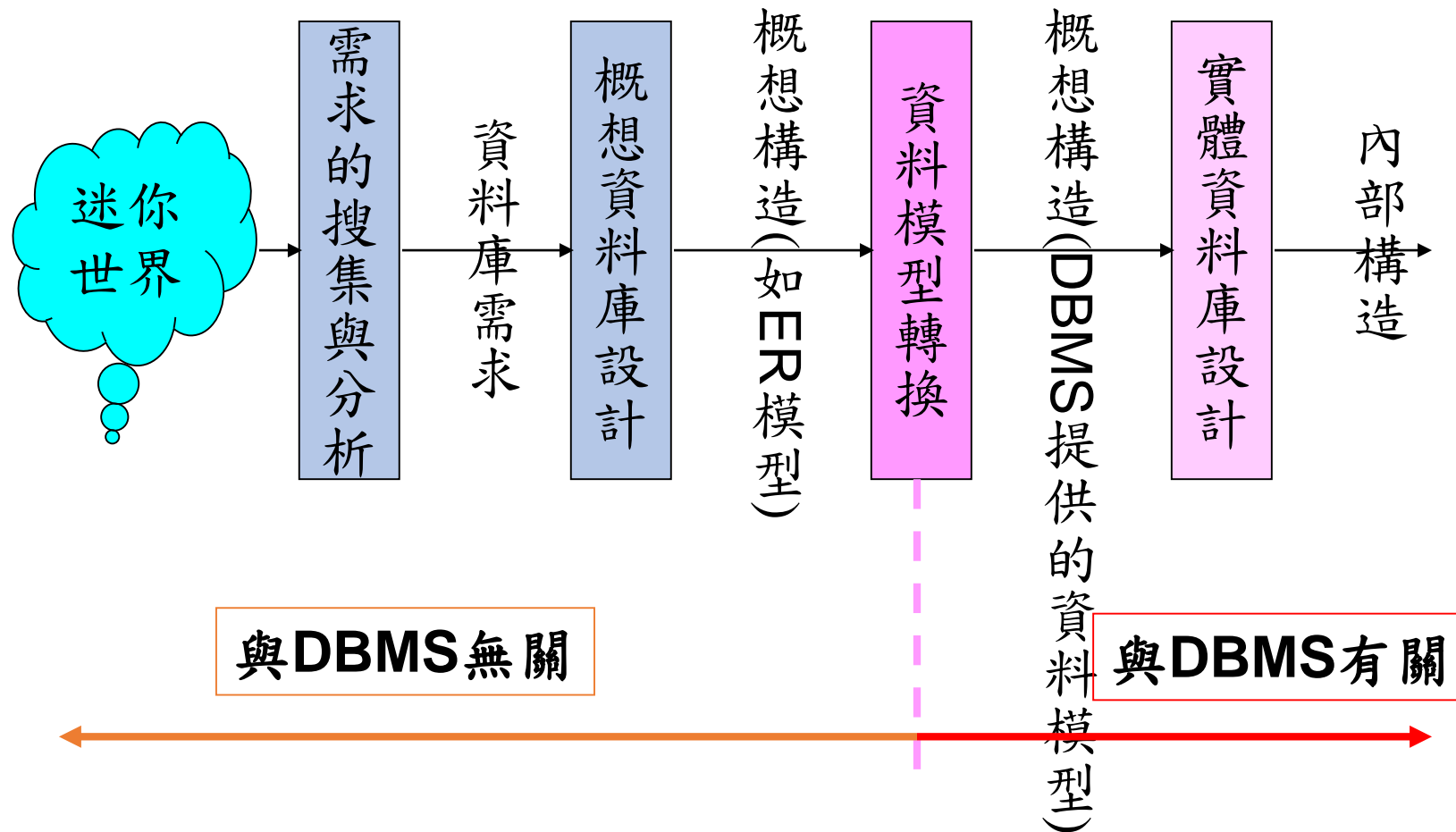
3.提供交易處理 (Transaction)

- 資料庫交易是指透過一連串的操作執行的一或多個 SQL 陳述式，形成單一**邏輯工作單位**。
- 交易的原則是「**全有或全無**」，表示整個交易必須做為單一單位完成並寫入資料庫，否則交易的任何個別元件都不應該通過。在關聯式資料庫的術語中，交易的結果包括「認可」(Commit)或「轉返」(Rollback)。處理每一個交易皆採用一致可靠的方式，和其他交易互不相干。

4. 具有ACID的特性

- 所有資料庫交易必須符合 ACID，即不可分割性 (Atomic)、一致性 (Consistent)、獨立性 (Isolated) 和耐用性 (Durable)，以確保資料完整性。
 - **不可分割性** 要求交易必須整體成功執行，若是交易有一部分操作失敗，整個交易都會失效。
 - **一致性** 要求做為交易的一部分寫入資料庫的資料，必須遵守所有明定規則以及約束，包括限制條件、級聯、觸發。
 - **獨立性** 是達成並行控制的重要關鍵，可以確保每一個交易都是獨立的。
 - **持久性** 要求在一個交易成功完成後，對資料庫所做的變更都是永久性的。

資料庫的設計步驟



資料庫設計步驟說明

●需求的搜集與分析

- 與資料庫的使用者面談以了解並記錄其需求
- 搜集資料要詳細與完整

●概想資料庫設計

- 以高階的、概想的資料模型來產生概想的資料庫(schema)
- 將使用者需求以簡明、一致的方式表達
- 確定能符合使用者的模型
- 例如：使用ER模型

資料庫設計步驟說明

●資料模型轉換

- 選定某一個DBMS
- 將概想資料庫構造轉換為選定的DBMS所提供的實行資料庫構造

●實體資料庫設計

- 決定資料庫裡的儲存結構與檔案組織
- 根據選定的DBMS所提供的

模組2：ER模型

2-1 ER模型之緣起與用途

2-2 ER模型基本符號與用法

2-3 各類ER模型表示法與工具

概想資料模型

●ER模型

- 實體－關係模型 (Entity-Relationship model)
- 是一種概想資料模型
- 目前大都用在資料庫的設計過程中
- P. Chen在1976年提出

●ER模型的觀念

- 以儘量接近使用者的觀點來描述資料，而非著重在資料的存放方式

ER 模型的觀念

名詞譯義

●實體與屬性(Entity and Attribute)

- 實體

- ER Model裡的基本個體
- 代表真實世界裡的“世界”
例如：汽車、房子、員工.....

- 屬性

- 描述實體的性質
- 實體裡每個屬性都有個值，稱之為屬性值
- 屬性值是存在資料庫裡主要的資料
例如：員工這個實體有底下這些屬性：姓名、性別、薪資...

ER 模型的觀念

●實體型別、值域與鍵值屬性(Entity type, value set and key attribute)

－實體型別

- 用來定義具有相同屬性串列的實體
- 包括一個名稱與屬性串列
- 以實型別構造(entity type schema)描述之

Employee
Name,age,salary

E1*
(john,55,80k)
E2*
(Judy,24,29k)

Company
Name,place,president

C1*
(sunco,loden,smith)
C2*
(fast,newyork,mary)

ER 模型的觀念

- 值域

- 屬性所可能包括的範圍值

例如：員工的年齡是介於15到70的整數

- 鍵值屬性

- 用來區別各個實體(同樣型別)的屬性
- 例如：employee裡的員工編號
- 用來保持唯一性的性質
- 鍵值屬性可能不只一個

例如：汽車的引擎號碼和牌照號碼

一個例子Company資料庫

- 底下我們舉一個例子來說明
 - 簡化資料庫設計步驟
 - ER Model
- 這是一個公司的資料庫(稱為Company)，所要記錄的資料有員工資料、部門資料與專案資料。

資料庫需求

茲將資料庫設計師經過搜集與分析使用者的需求後，所整理出的條目列舉如后：

1. 公司有數個部門，每個部門有一個名稱、一個編號、與一位經理、我們要記錄該經理何時出掌該部門。一個部門可能分散數地。
2. 一個部門底下有數個專案。每個專案有一個名稱、一個編號、與一個執行地點
3. 員工的資料包括：姓名、編號(ssn)、住址、性別、薪資與生日。每位員工屬於一個部門，但可能參與多個專案，參與的專案不一定是所屬的部門所負責。我們要記錄每星期、員工在每個專案所花的時間，也要記錄該員工的頂頭上司是誰。
4. 記錄員工的家屬資料：姓名、性別、生日與稱謂

Company資料庫的初步構造

- 先定義Company資料庫的實體型別，根據使用者的需求我定義了四個實體型別

DEPARTMENT

Name, Number, {Locations}, Manager, ManagerStartDate

PROJECT

Name, Number, Locations, ControllingDepartment

EMPLOYEE

Name(Fname, Lname), SSN, Address, Sex, Salary, Birthday, Department, Supervisor, {WorkOn(Project, Hours)}

DEPENDENT

Employee, DependentName, Sex, Birthdate, Relationship

多值屬性用{ }括起來、合成屬性用()括起來

Company資料庫的初步構造

- 每位員工參與專案的描述是在EMPLOYEE型別之WorkOn(Project, Hours)這個多值屬性中。
- 儘量在定義實體型別時讓它的鍵值為基本屬性。
- 在上述的定義裡我們並沒有描述出實體間的關係，像DEPARTMENT裡的manager是EMPLOYEE裡的一員，PROJECT裡的ControllingDepartment是DEPARTMENT裡的一員.....等。

Company資料庫的進階構造

有了關係這個觀念後，新的資料庫構造如下述：

- 實體型別有：



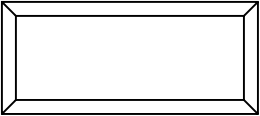
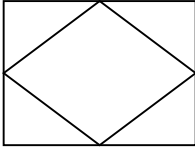
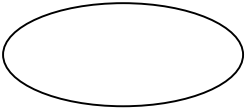
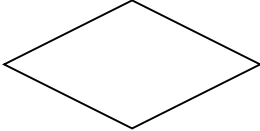
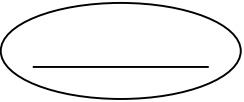
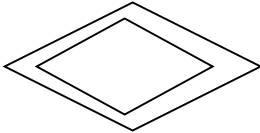
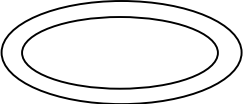
E1. 實體型別**DEPARTMENT**有屬性：名稱、編號和位置；鍵值屬性是名稱和編號

E2. 實體型別**PROJECT**有屬性：名稱、編號和位置；鍵值屬性是名稱和編號

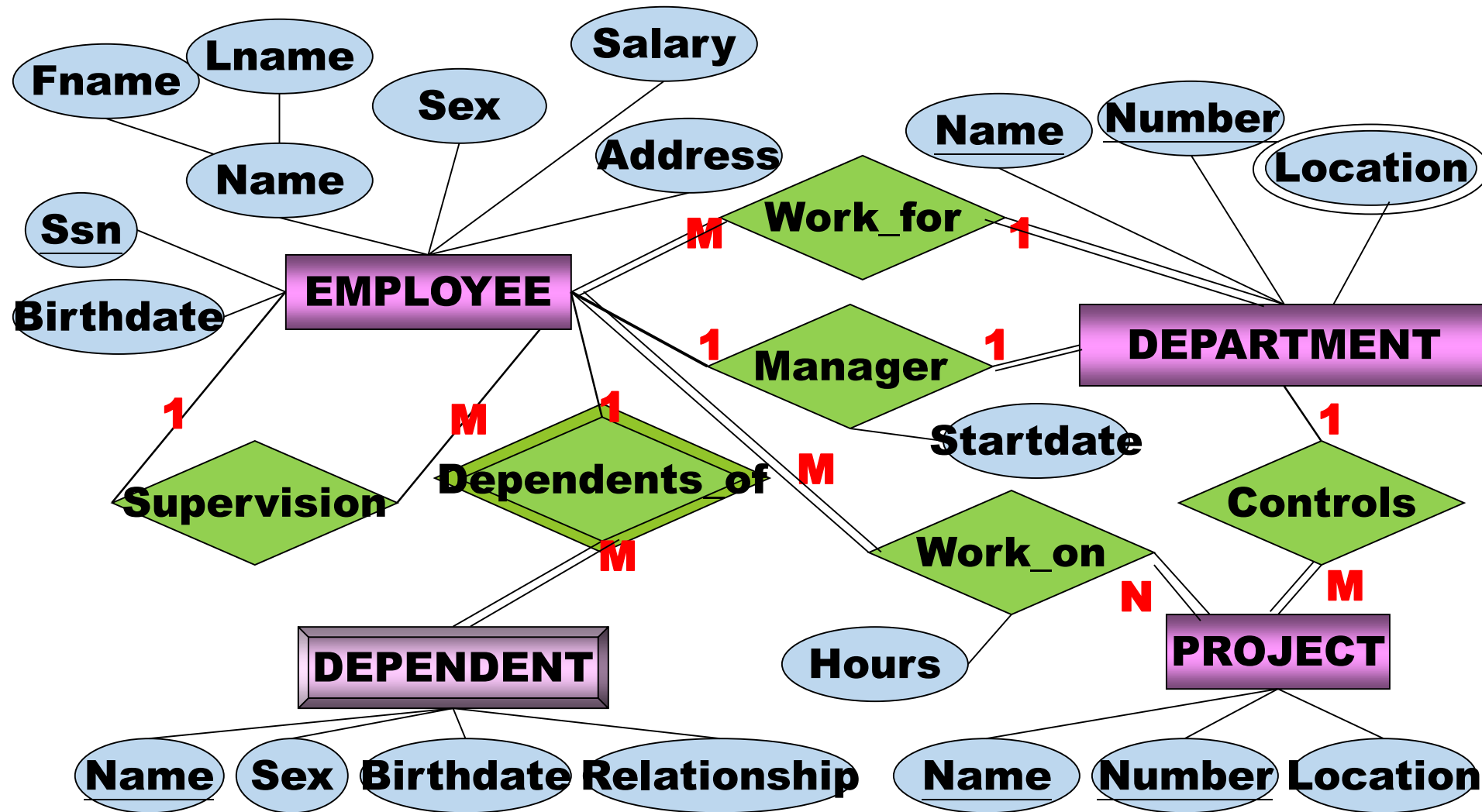
E3. 實體型別**EMPLOYEE**有屬性：姓名、編號、性別、位址、薪資和生日；鍵值屬性是編號

E4. 實體型別**DEPENDENT**有屬性：親屬名稱，性別、生日和稱謂；鍵值屬性是親屬名稱

ER 模型的符號

符號	意義	符號	意義
	實體		衍生屬性
	弱實體		結合實體
	屬性		關係
	鍵值屬性		確定關係
	多值屬性		

Company 資料庫ER模型圖



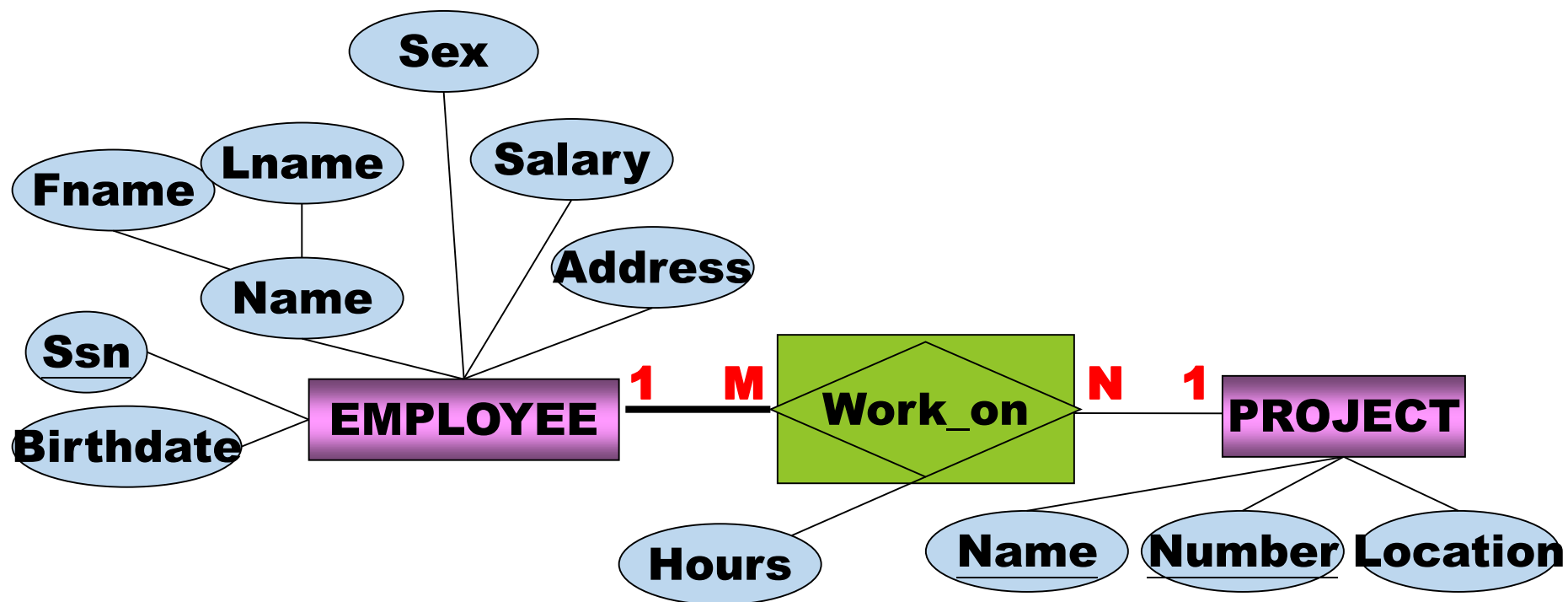
ER模型轉成關聯型別

實體EMPLOYEE與實體PROJECT關係為多對多

將關係Work_on轉為結合實體Work_on

讓實體EMPLOYEE對結合實體Work_on關係為一對多

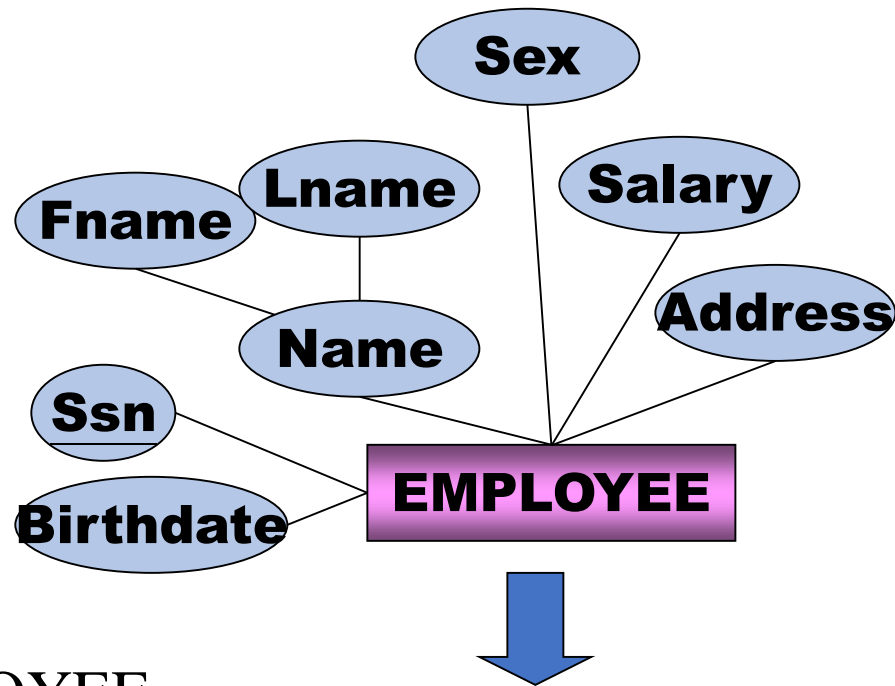
讓實體PROJECT對結合實體Work_on關係為一對多



ER模型轉成關聯型別

- 選定特定種類之資料庫管理系統(DBMS)
- 將實體關係模型轉換成此種類DBMS之模型
- 若選定的是關聯式模型則步驟如下:
 - － 實體 ⇨ 表格
 - － 屬性 ⇨ 欄位
 - － 關係 ⇨ FK

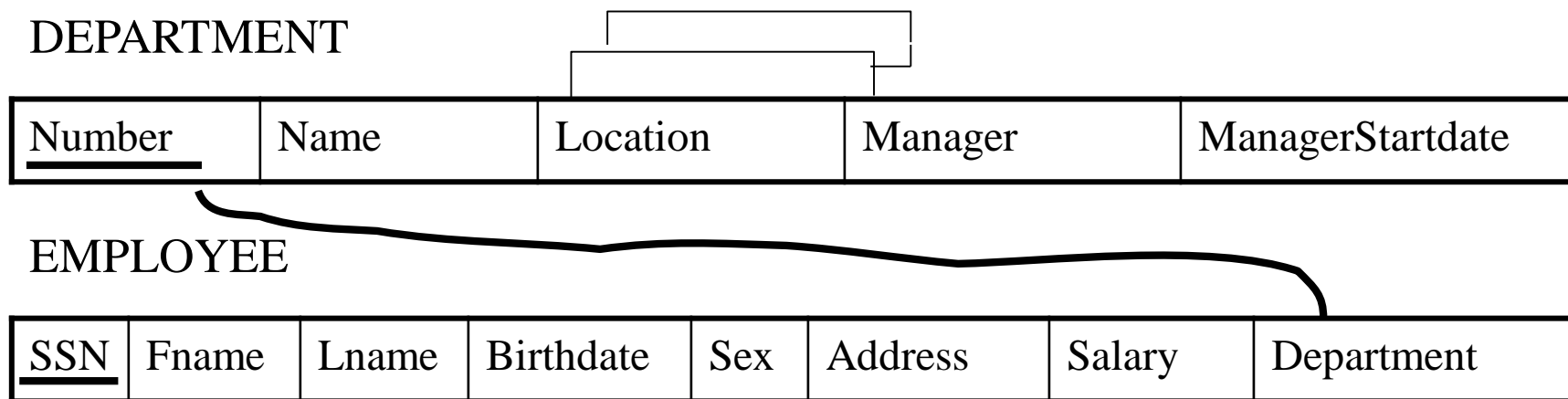
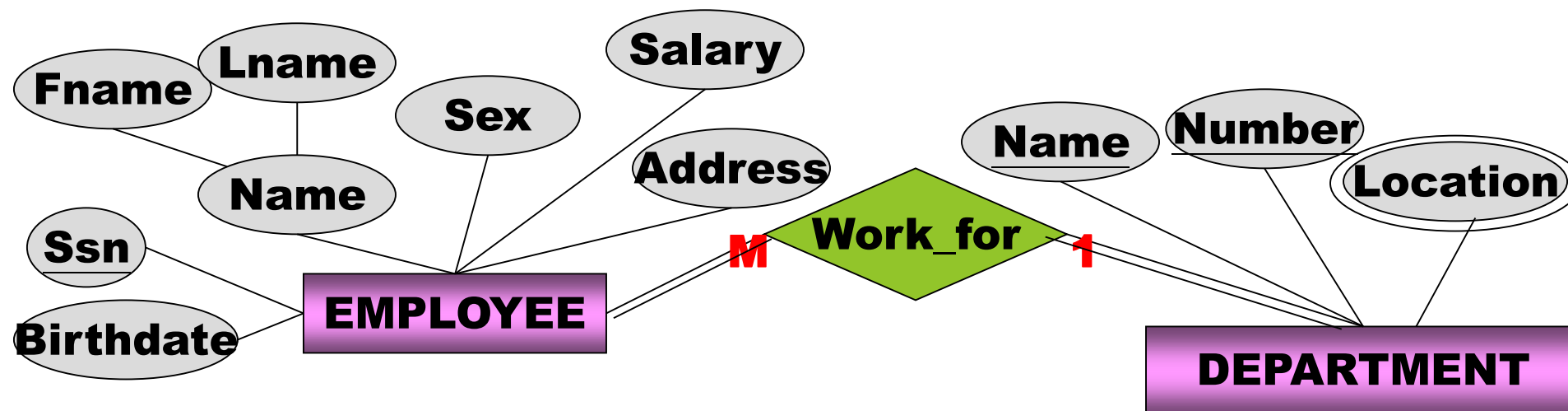
ER模型轉成關聯型別



EMPLOYEE

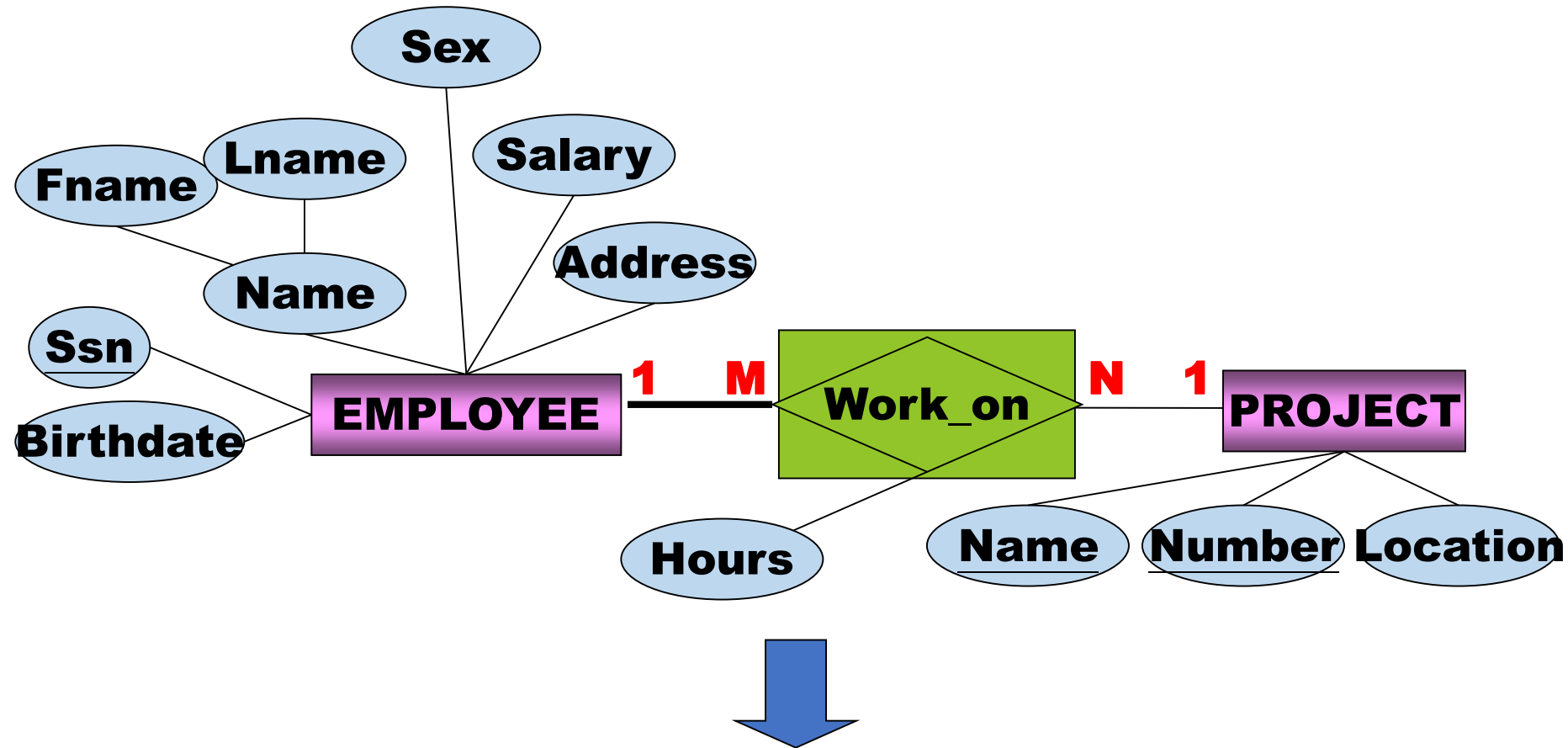
SSN	Fname	Lname	Birthdate	Sex	Address	Salary
-----	-------	-------	-----------	-----	---------	--------

ER模型轉成關聯型別



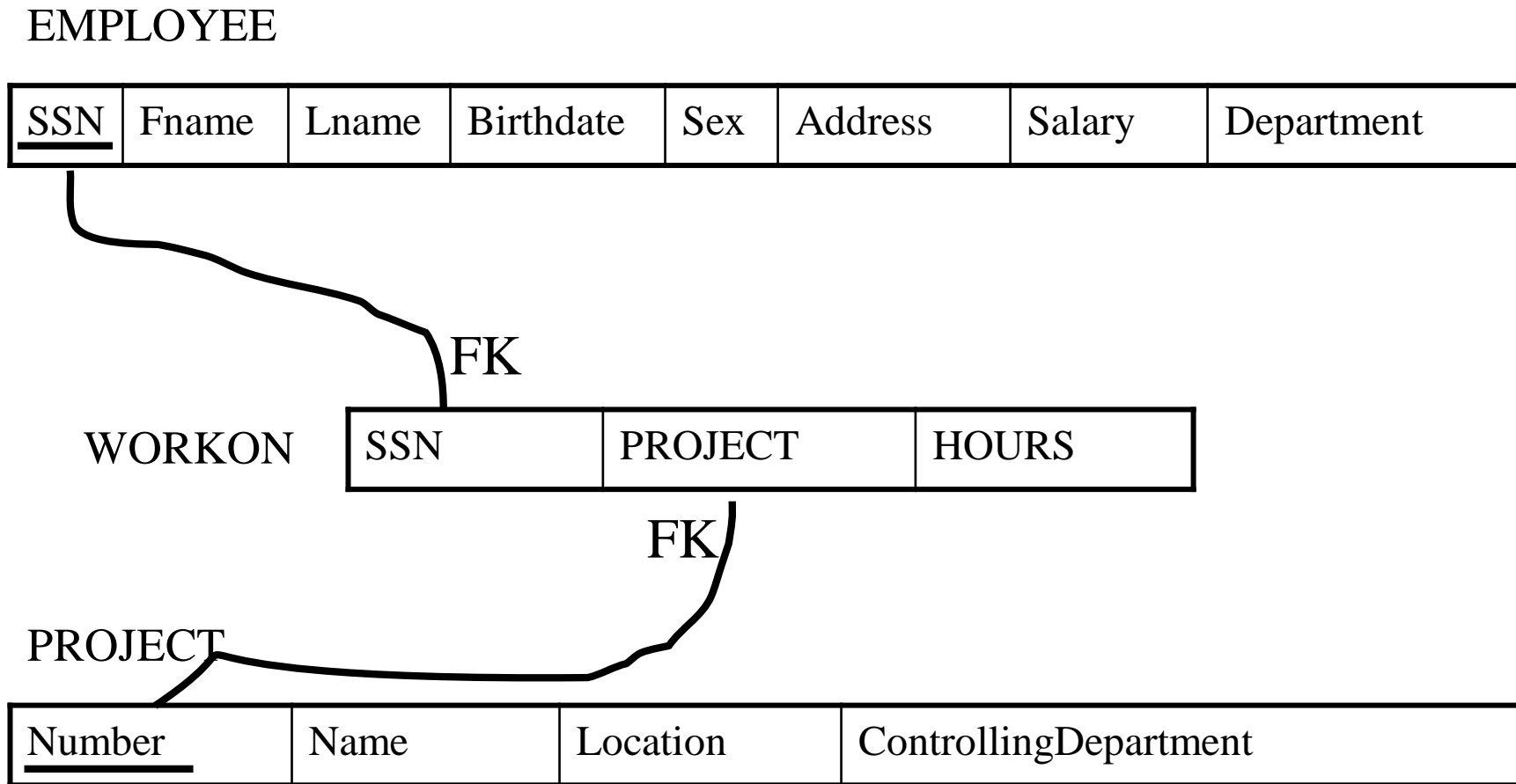
FK

ER模型轉成關聯型別

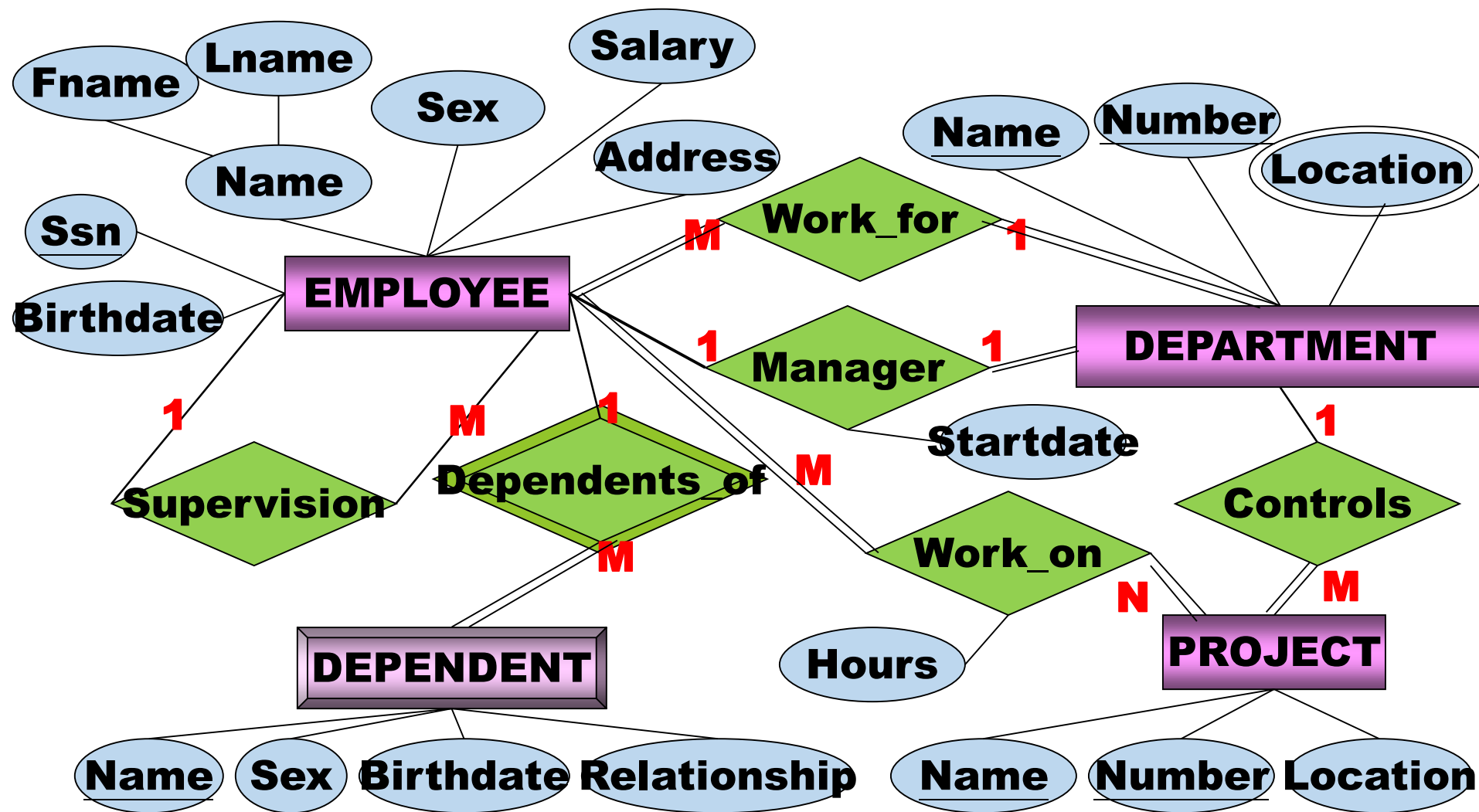


依據二個實體型別與一個結合實體關係轉成關聯模型

ER模型轉成關聯型別



ER模型轉成關聯型別



ER模型轉成關聯型別

DEPENDENT

Employee	Name	Sex	Birthdate	Relationship
----------	------	-----	-----------	--------------

DEPARTMENT

<u>Number</u>	Name	Location	Manager	ManagerStartdate
---------------	------	----------	---------	------------------

EMPLOYEE

<u>SSN</u>	Fname	Lname	Birthdate	Sex	Address	Salary	Department	Supervisor
------------	-------	-------	-----------	-----	---------	--------	------------	------------

WORKON

SSN	PROJECT	HOURS
-----	---------	-------

PROJECT

<u>Number</u>	Name	Location	ControllingDepartment
---------------	------	----------	-----------------------

以UML之類別圖表現ER模型

- UML：Unified Modeling Language。
- UML 是一套標準的圖形及符號，用來畫軟體藍圖。
- 市面上有許多CASE Tool支援UML, 例: Astah, Visual Paradigm 等。
- 可以用UML之類別圖表現ER模型。

模組3：ER模型實體塑模

3-1 辨認出實體及屬性

3-2 實體之圖形表現方式

3-3 屬性之圖形表現方式

物件(Object)與類別(Class)

- **物件**：真實世界中的事物與觀念
 - 特徵(identity)：name
 - 狀態(state)：attributes and values
 - 行為(behavior)：methods
- **類別**：一群具有共同特性的物件
 - 特徵(identity)：name
 - 屬性(attributes)
 - 行為(behavior)：operations

類別 (Class)

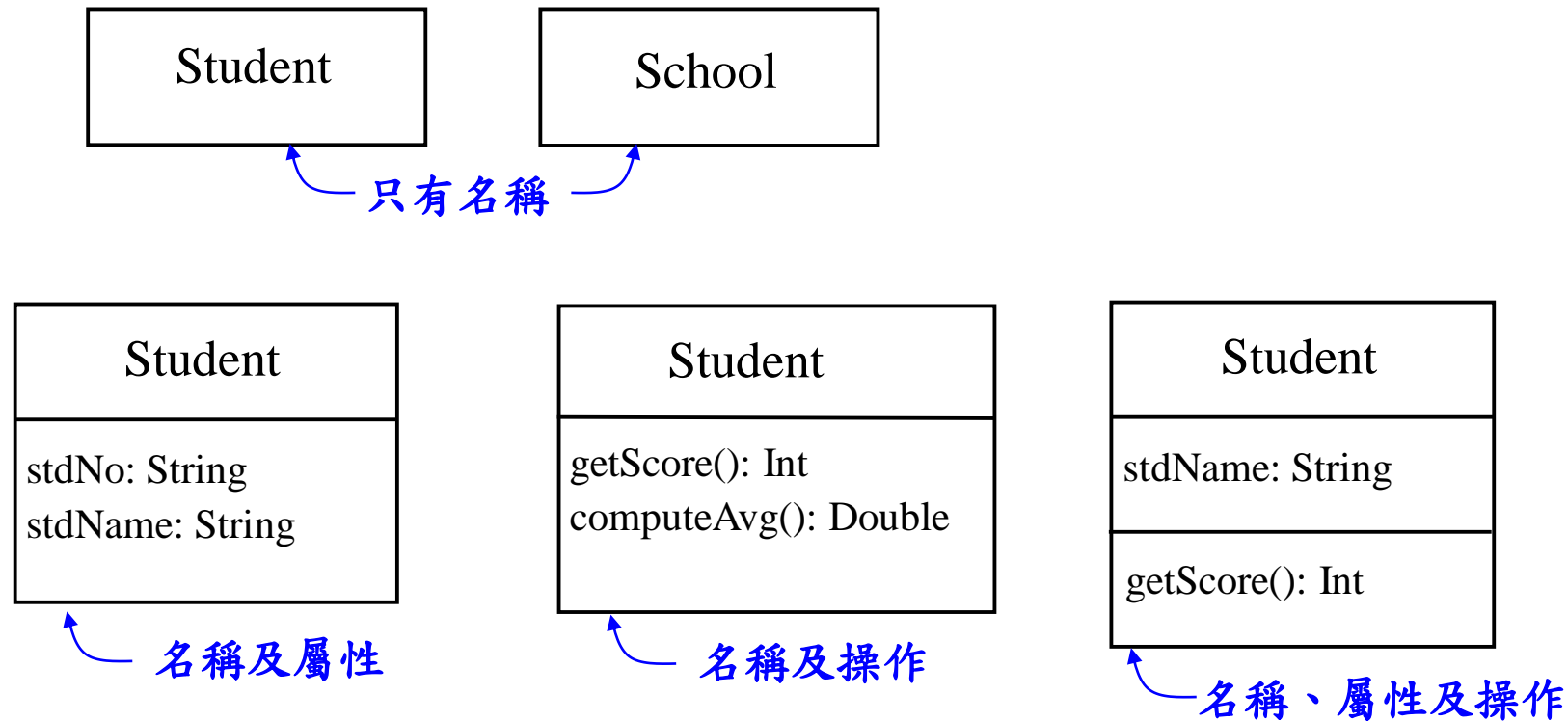
名 稱	Name
屬 性	Attribute
操 作	Operation

(1) Name : Book, Check Account, ATM Card

(2) Attribute : author, amount
bookTitle, serialNumber
ID, PIN

(3) Operation : check(), verify()
verifyPassword(), assignRating(rating : Int)
computeAvgRating() : Double

類別 (Class)



- 視狀況或需要，顯示類別之不同詳細度

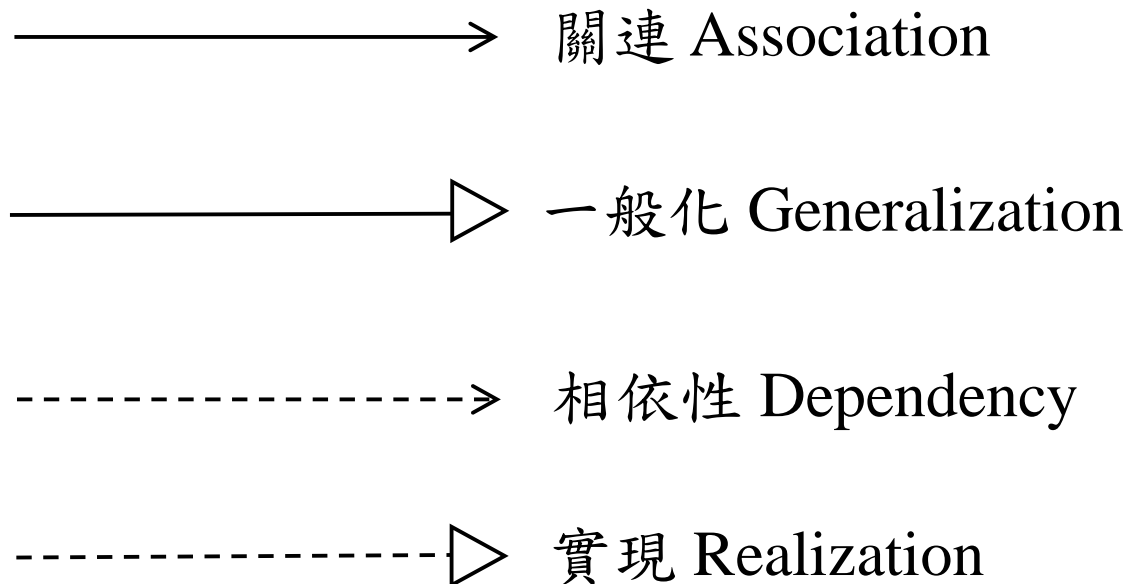
模組4：ER模型關係塑模

4-1 關係之圖形表現方式

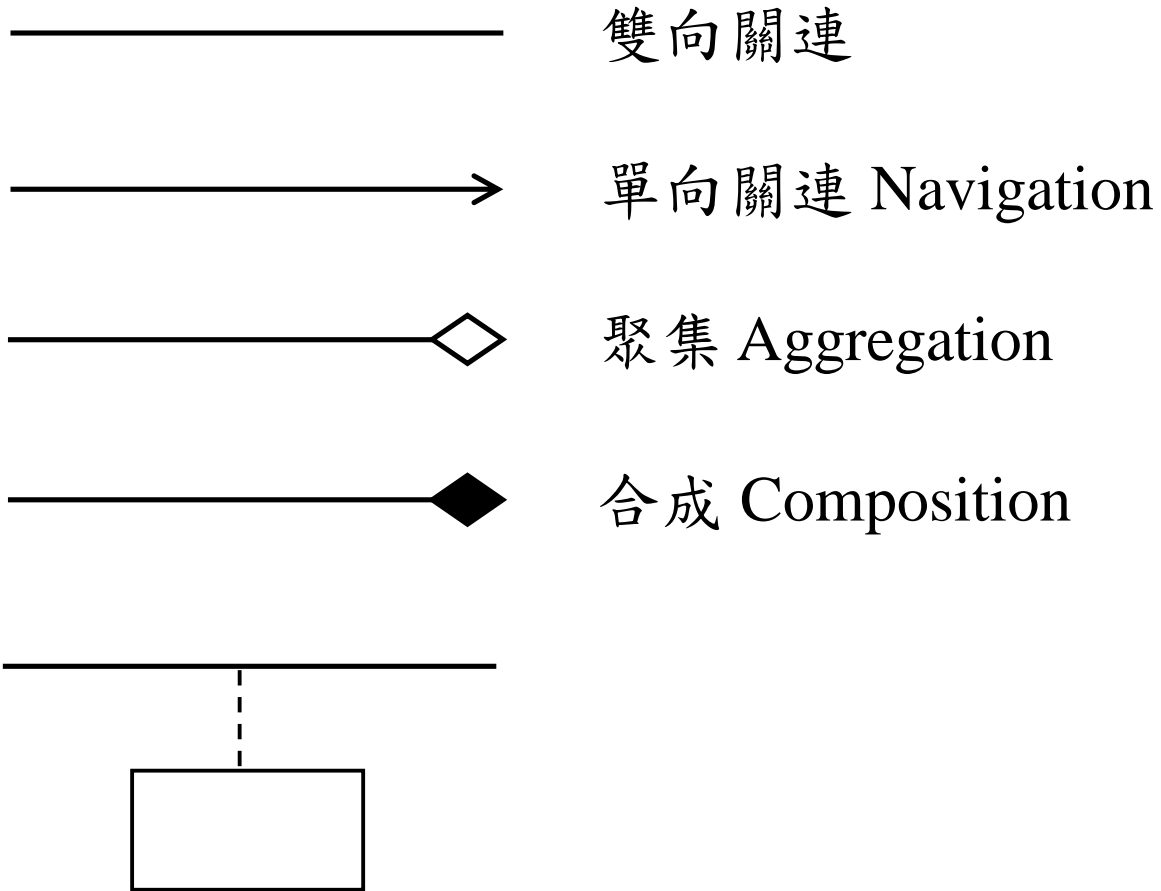
4-2 決定關係的種類

4-3 處理一對一關係及多對多關係

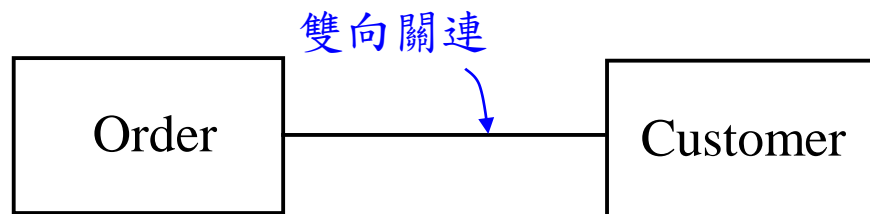
關係 (Relationships)



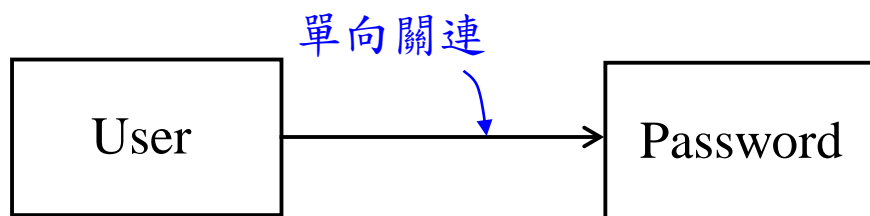
關連 (Association)



關連 - 航向 (Navigation)

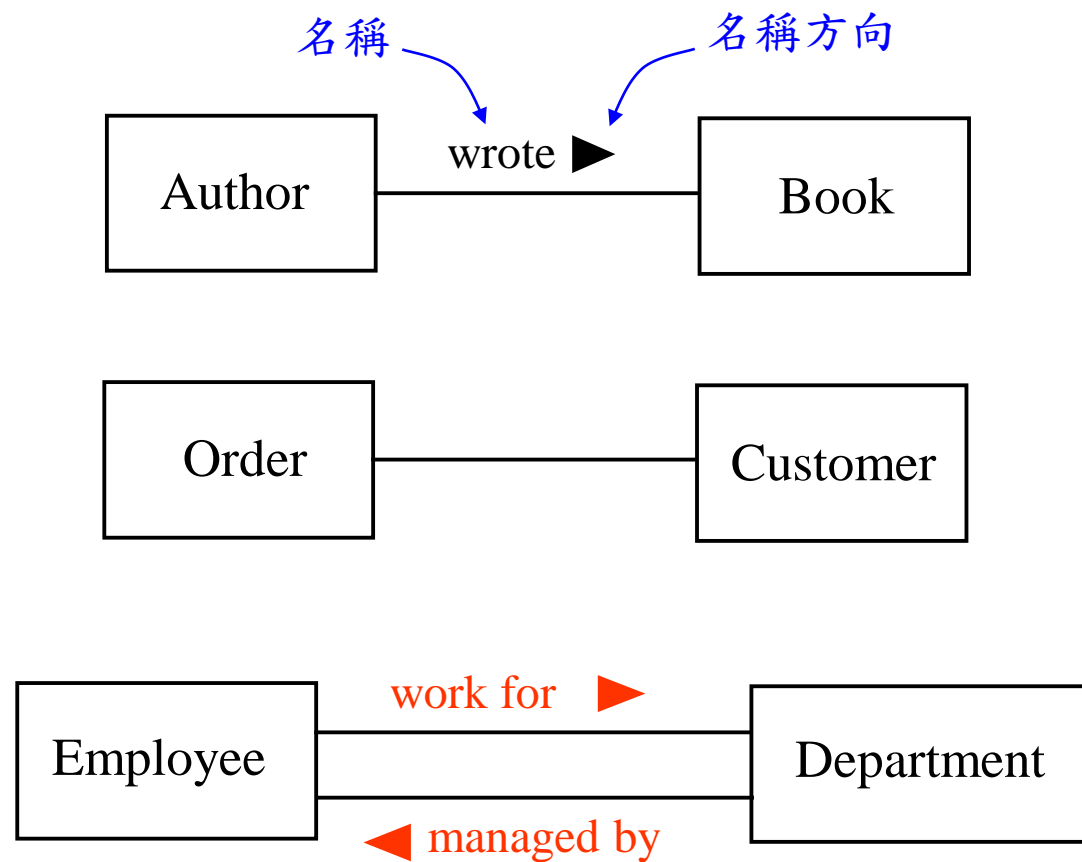


- 由任一方之物件可以找到另一方相對應之物件
 - Order → Customer 且 Customer → Order

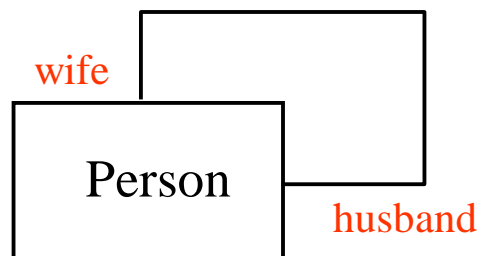
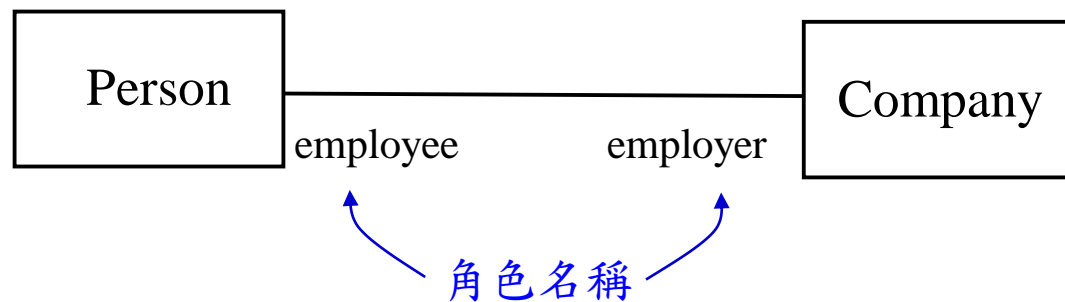


- 只能由一方之物件找到另一方相對應之物件，反之不然
 - User → Password 但 Password ✗ User

關連 - 名稱 (Name)

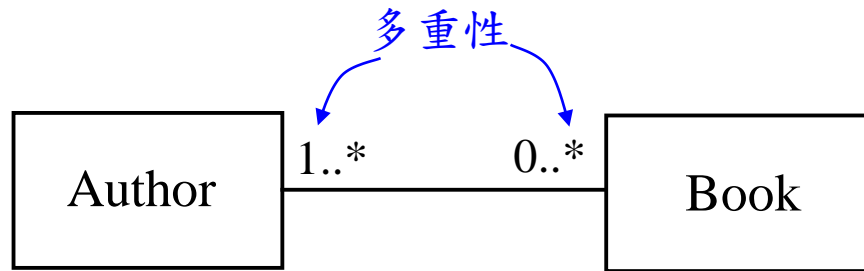


關連 - 角色 (Role)



關連 - 多重性 (Multiplicity)

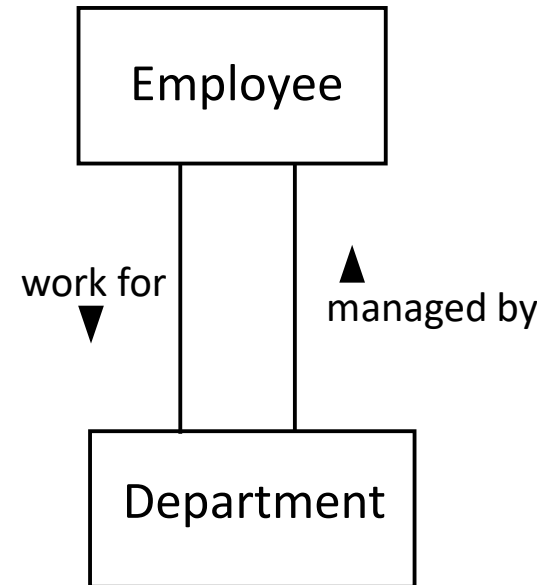
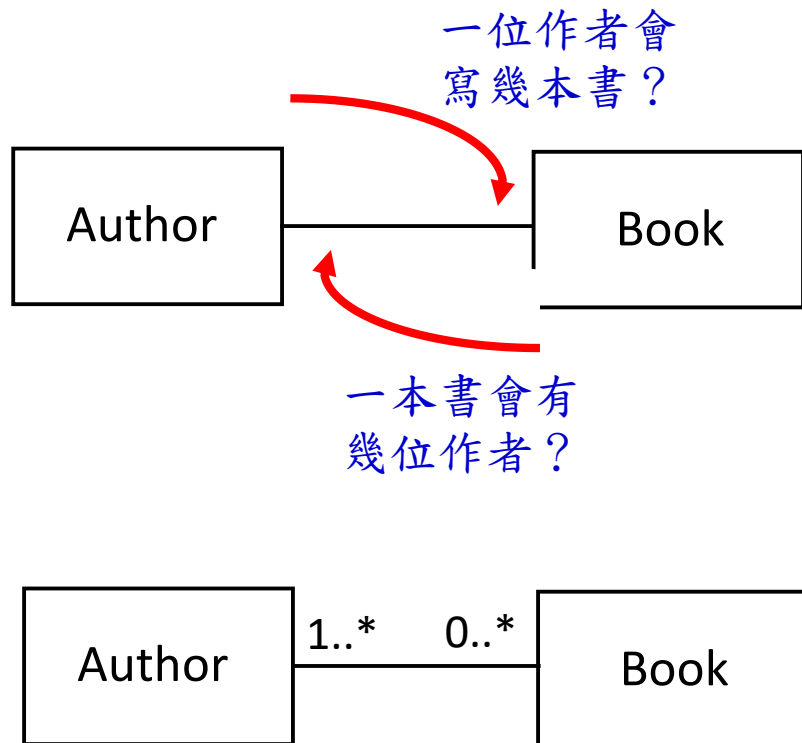
- 意義：對此關連，每一個類別中到底有多少個物件參與其中。



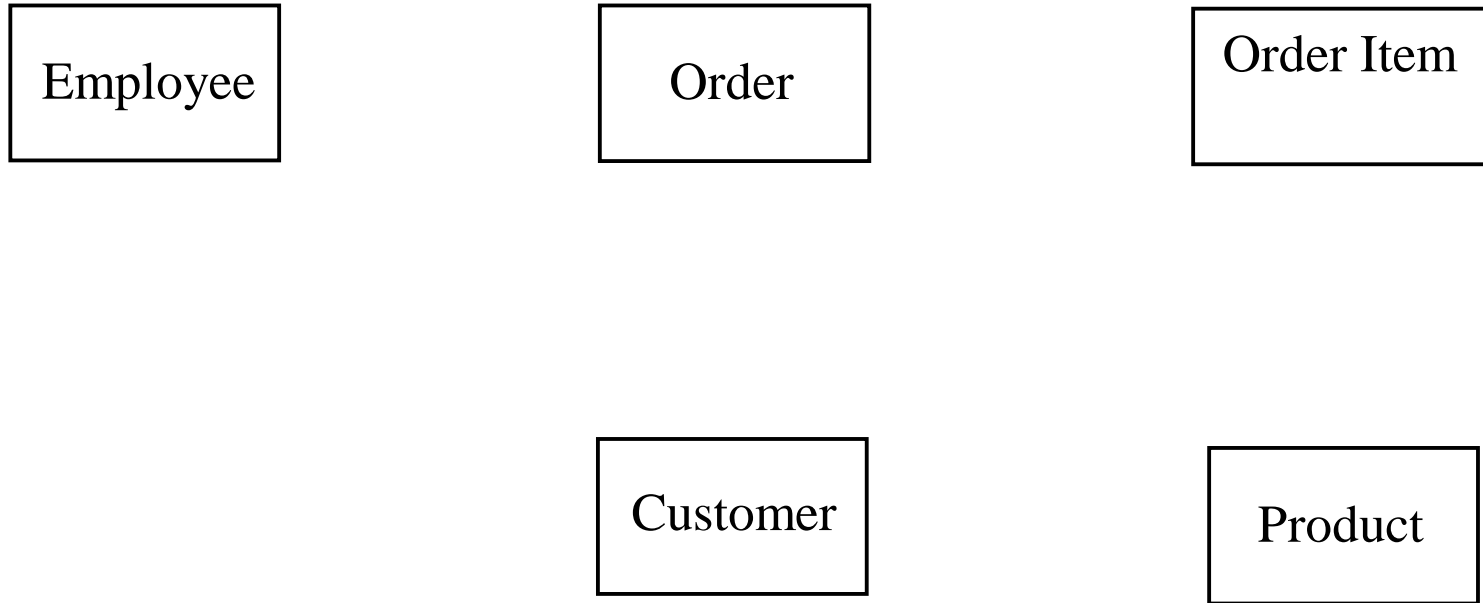
- 多重性：
 - 2..4
 - 1..1 : 1
 - 1..*
 - 0..* : *

關連 - 多重性 (Multiplicity)

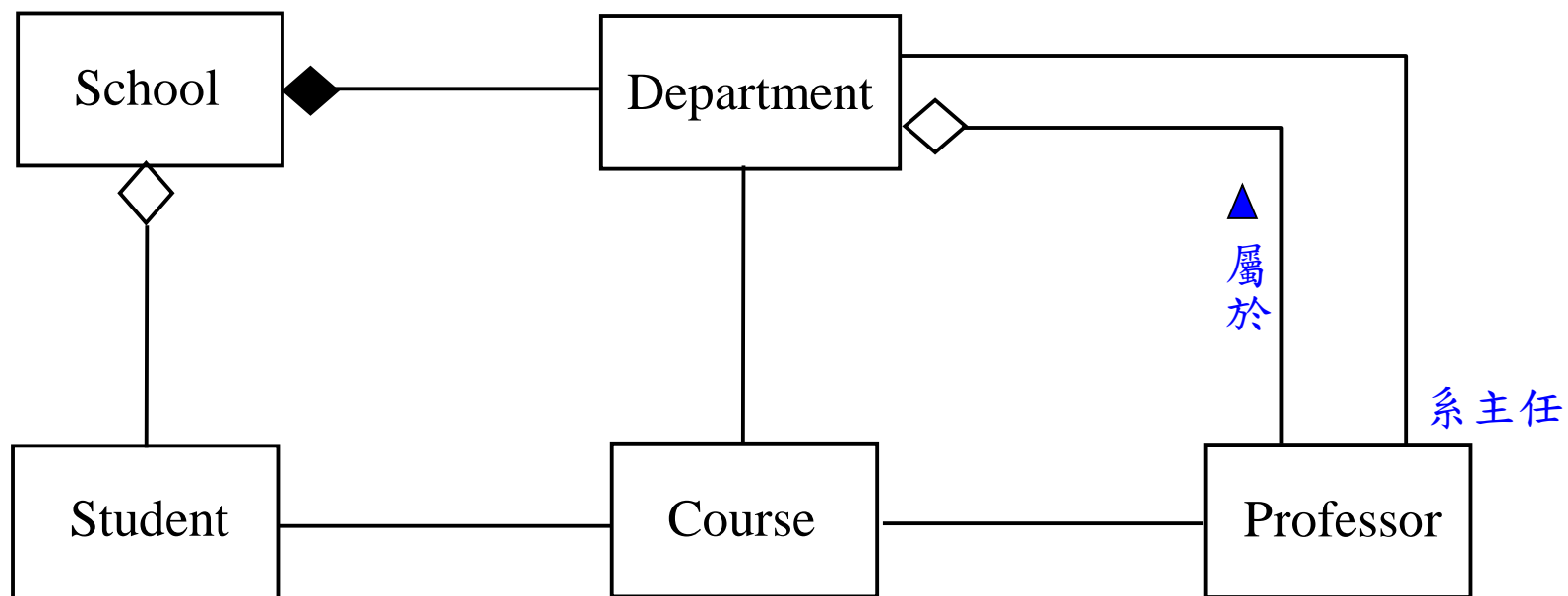
- 如何決定多重性：



關連 - 多重性 (Multiplicity)



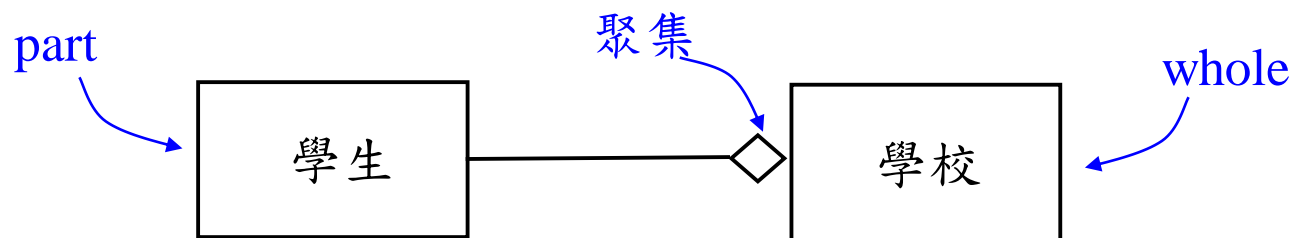
關連 - 多重性 (Multiplicity)



4-1 關係之圖形表現方式

關連 - 聚集 (Aggregation)

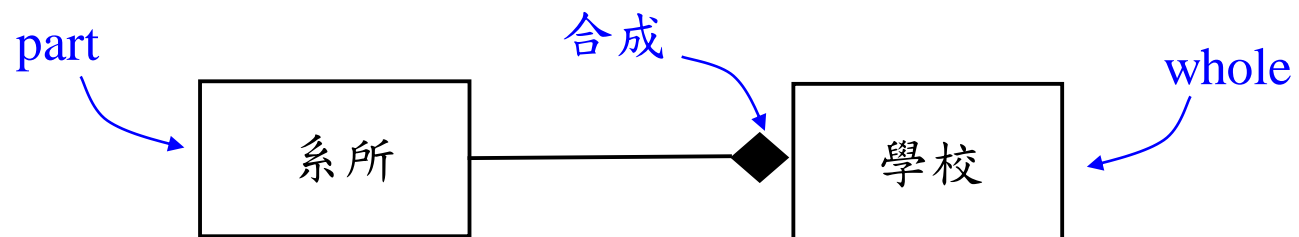
- 聚集與合成均用以表示 whole/part 關係
- 聚集關係較為鬆散，合成關係較為緊密



- 聚集關係較為鬆散，當學校不存在，學生依然不受影響

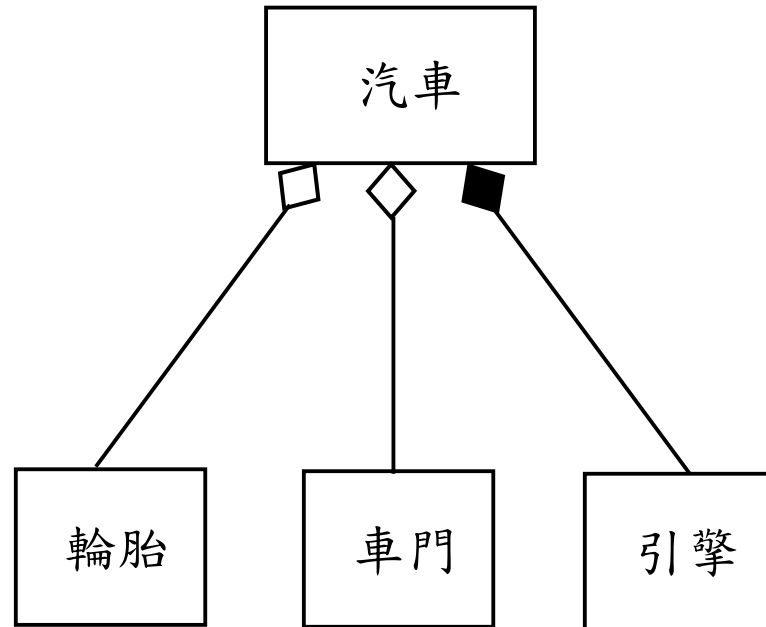
關連 - 合成 (Composition)

- 聚集與合成均用以表示 whole/part 關係
- 聚集關係較為鬆散，合成關係較為緊密



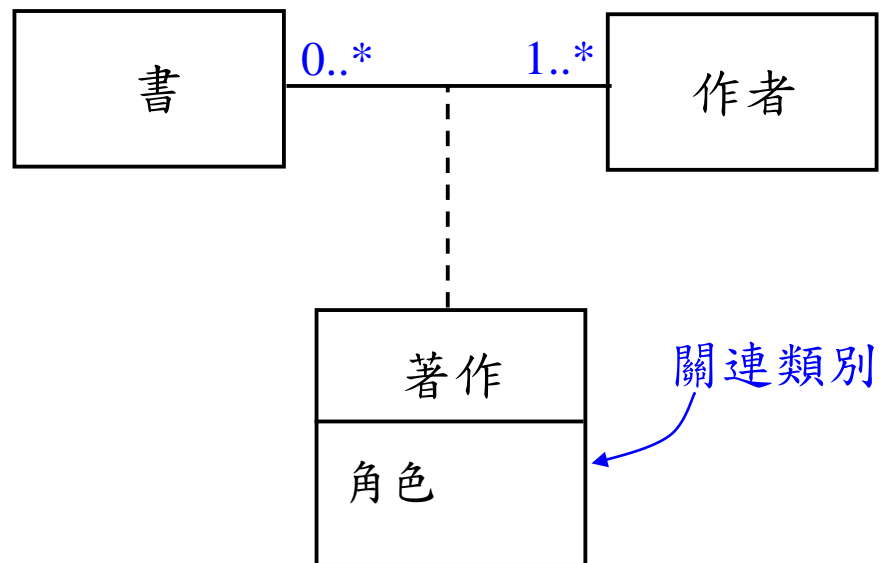
- 合成關係較為緊密，當學校不存在，校內系所也不存在

關連 - 聚集與合成



關連 - 關連類別 (Association Class)

- 對於關連之額外說明



資料庫設計程序

- 選出最初之實體
- 找出直接關接
- 繪出初步E-R模式圖形
- 決定E-R圖形之關係種類
- 解決一對一及多對多之關係
- 正規化處理
- 確認E-R模式圖形的正確性
- 資料模型轉換
- 表格(Table)規格描述
- SQL命令建構資料庫的Table

顧問公司之計畫規劃範例

本例為一顧問公司，有關規劃及計畫控制之流程，以下為一描述：

由客戶方面所發出之查詢將經由公司之地區辦公室傳達至公司之服務部門。服務部門將根據現行之成本及過去相似之計畫等資料產生一報價單。此報價單將送至地區辦公室，由地區辦公室轉寄各客戶，同時等待客戶之答覆，若報價單客戶接受，地區辦公室將此訊息傳到服務部以產生計畫規劃書。

計畫規劃書之產生過程中，將參考過去曾有過相似計畫之報價單及公司內具有執行此計畫之能力與時間之顧問。某些顧問將被指派於此計畫中，一個計畫將被區分為若干個小的活動，以便執行。當這些活動被鍵入電腦後，計畫才得以正式展開。

每週每個顧問將自己工作於某計畫之工作時數及交通住宿等費用送到公司之會計部門，會計部門將此些成本做一記錄，並和預估成本做一比對，每月中，電腦部門將為服務部門產製一活動報告書，服務部將根據此報告產生估價單，並送至會計部門。會計部門將根據與客戶間之合約，將屬於同一計畫之估價單，計價後，送至發票部門，由發票部門開立發票後送給客戶。

步驟一、選出最初之實體

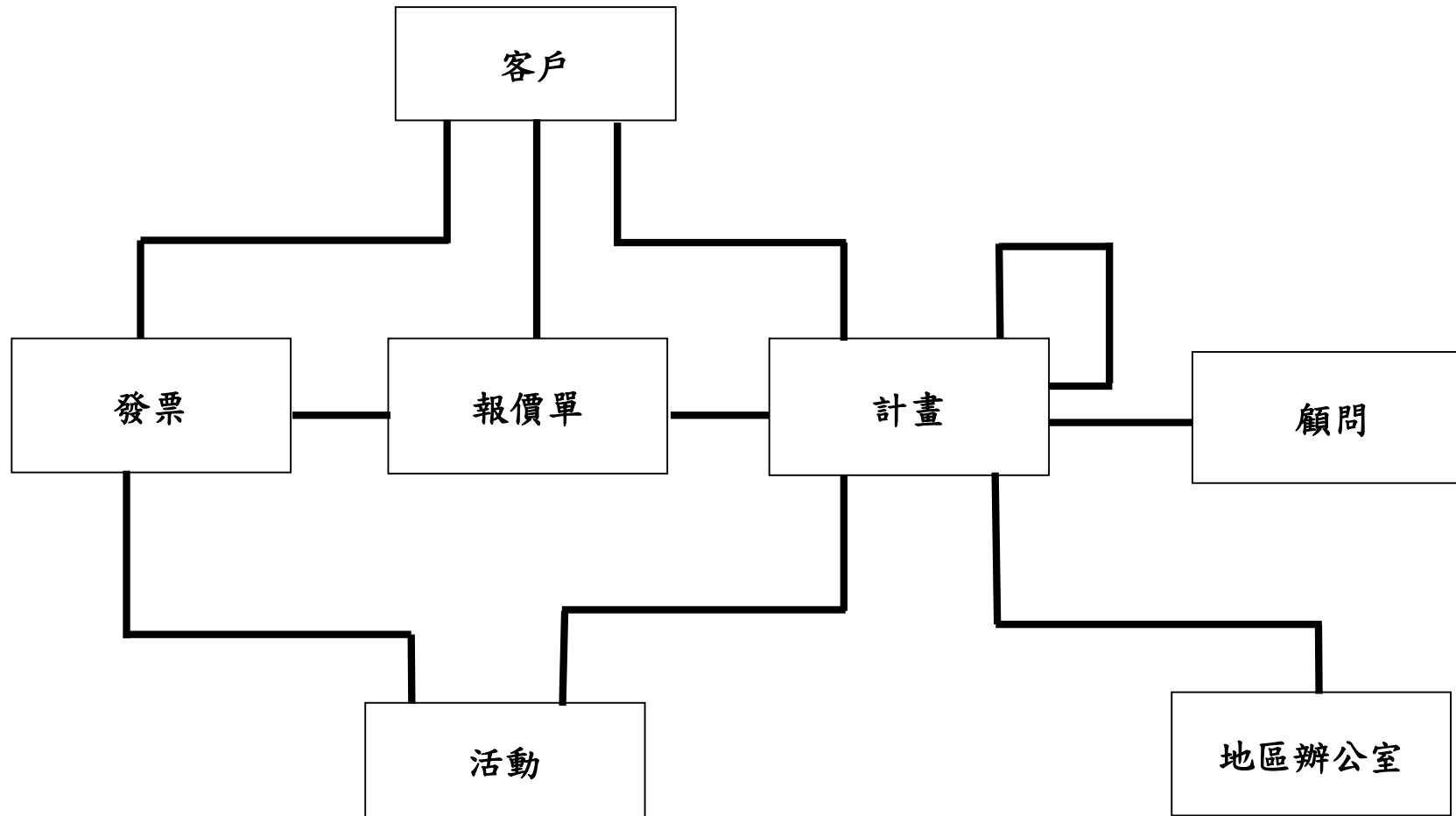
- 顧問
- 計畫
- 客戶
- 地區辦公室
- 發票
- 活動
- 報價單

4-2 決定關係的種類

步驟二、找出直接關係

	報價單	活動	發票	地區辦公室	客戶	計畫	顧問
顧問						X	
計畫	X	X		X	X	X	
客戶	X		X				
地區辦公室							
發票	X	X					
活動							
報價單							

步驟三、繪出初步E-R模式圖形

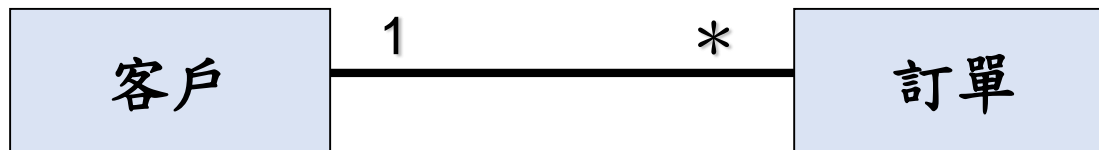


步驟四、決定E-R圖形之關係種類

一對一



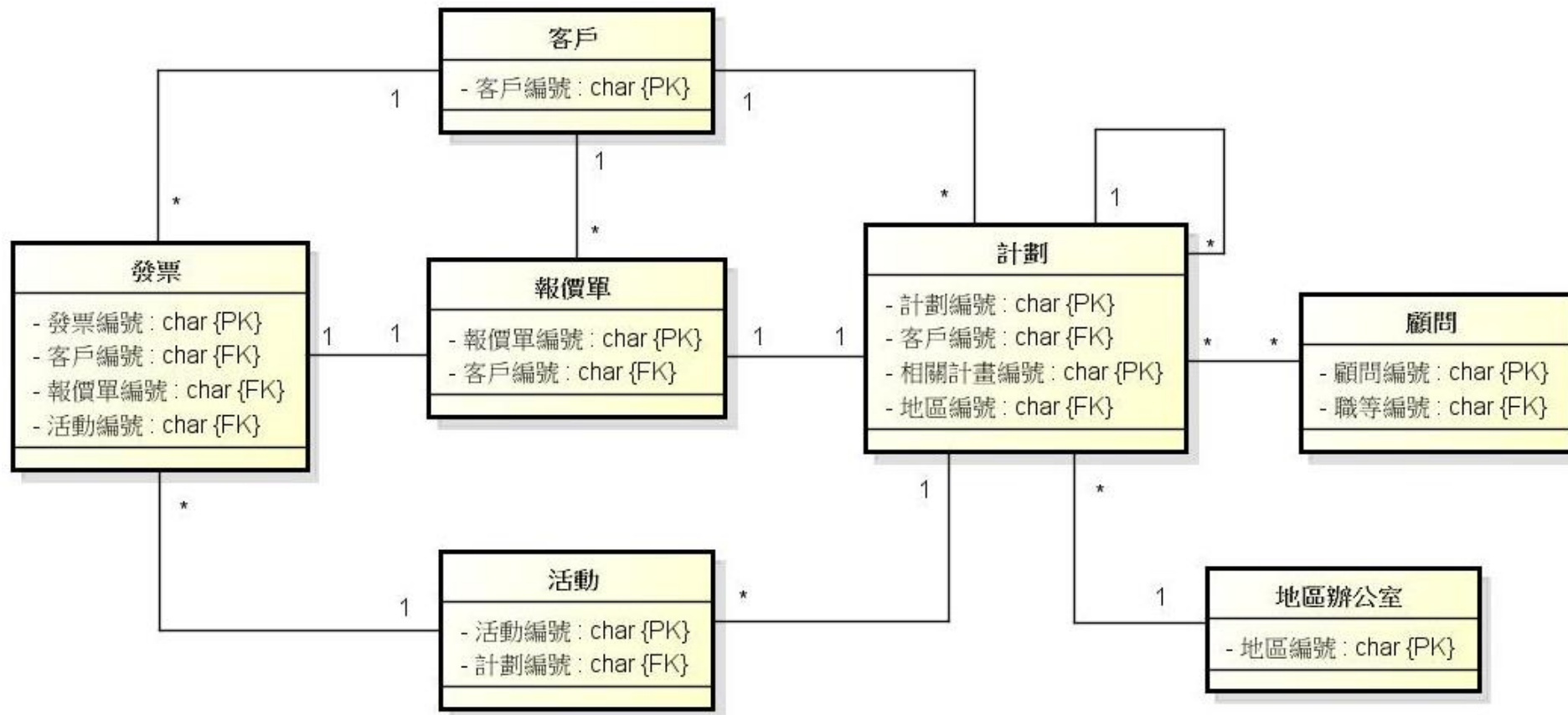
一對多



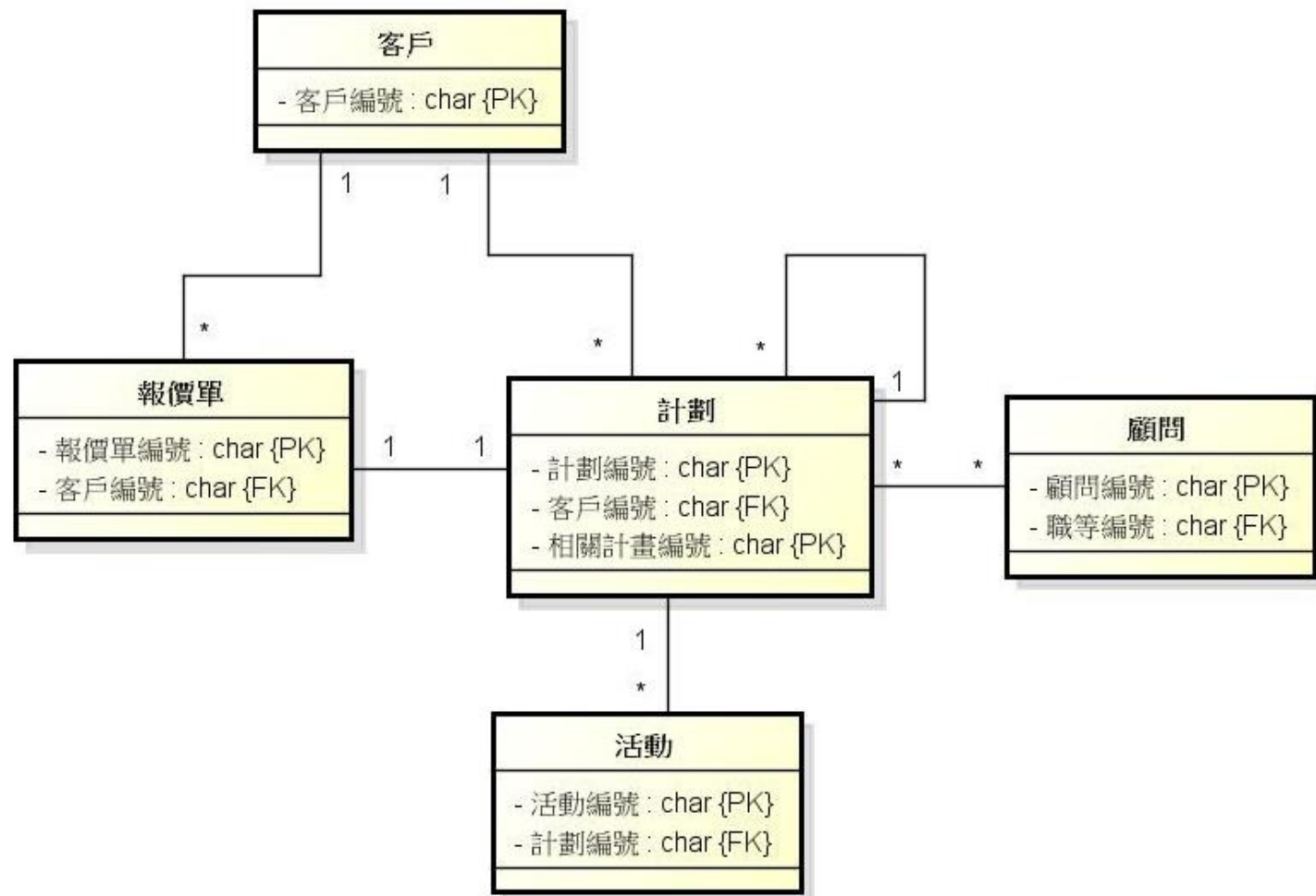
多對多



決定E-R圖形之關係種類



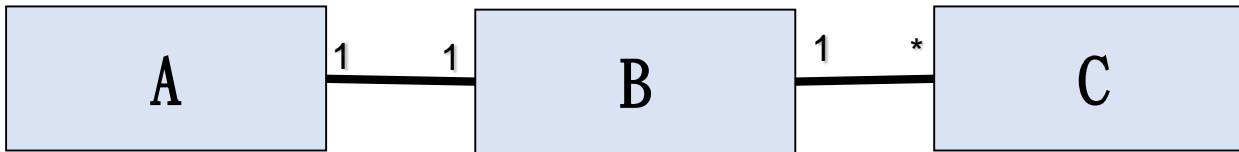
檢查系統範圍內之關係



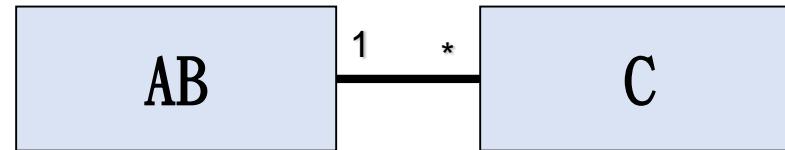
4-3 處理一對一關係及多對多關係

步驟五、解決一對一及多對多之關係

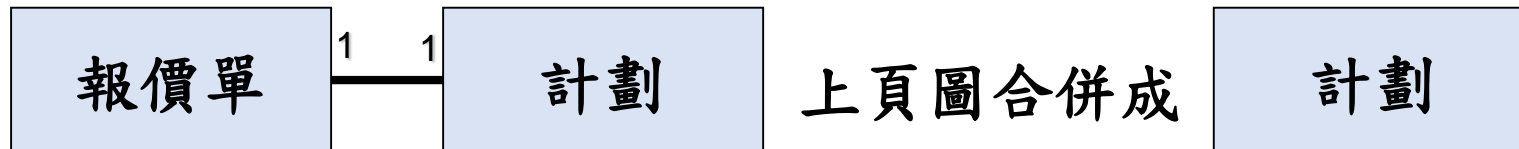
一對一



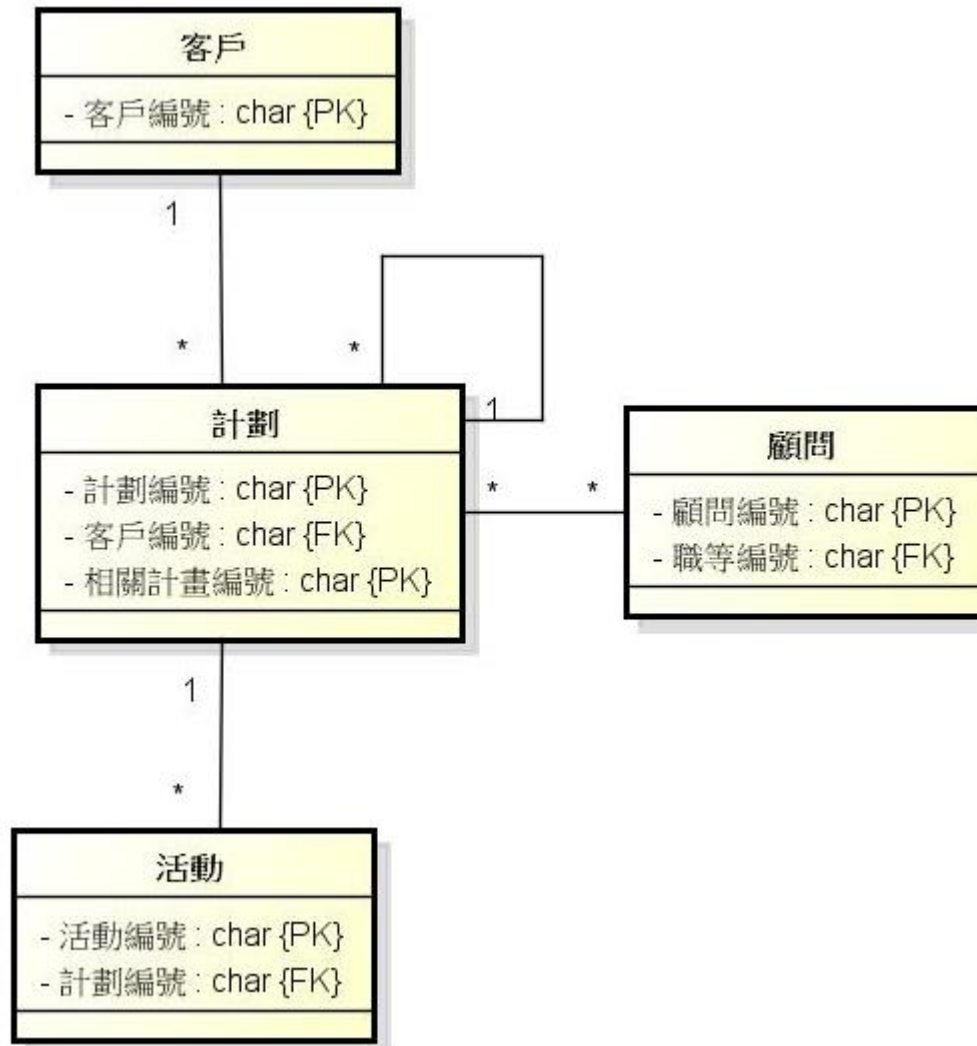
變成



顧問例



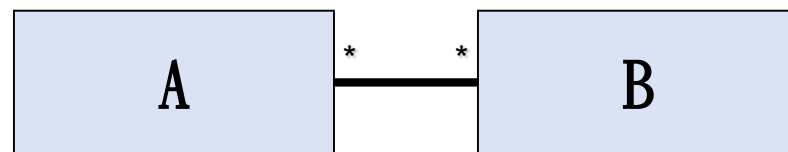
一對一關係解決後之圖形



4-3 處理一對一關係及多對多關係

多對多之關係

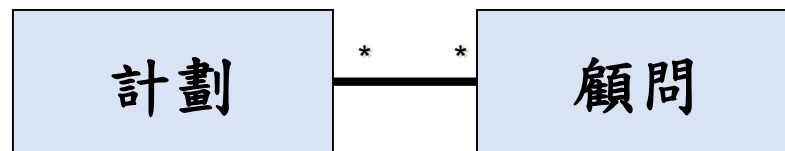
多對多



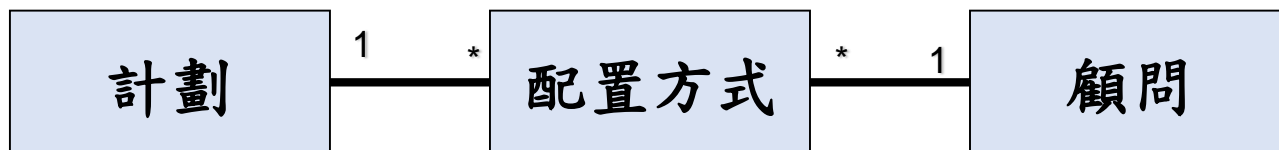
變成



顧問例

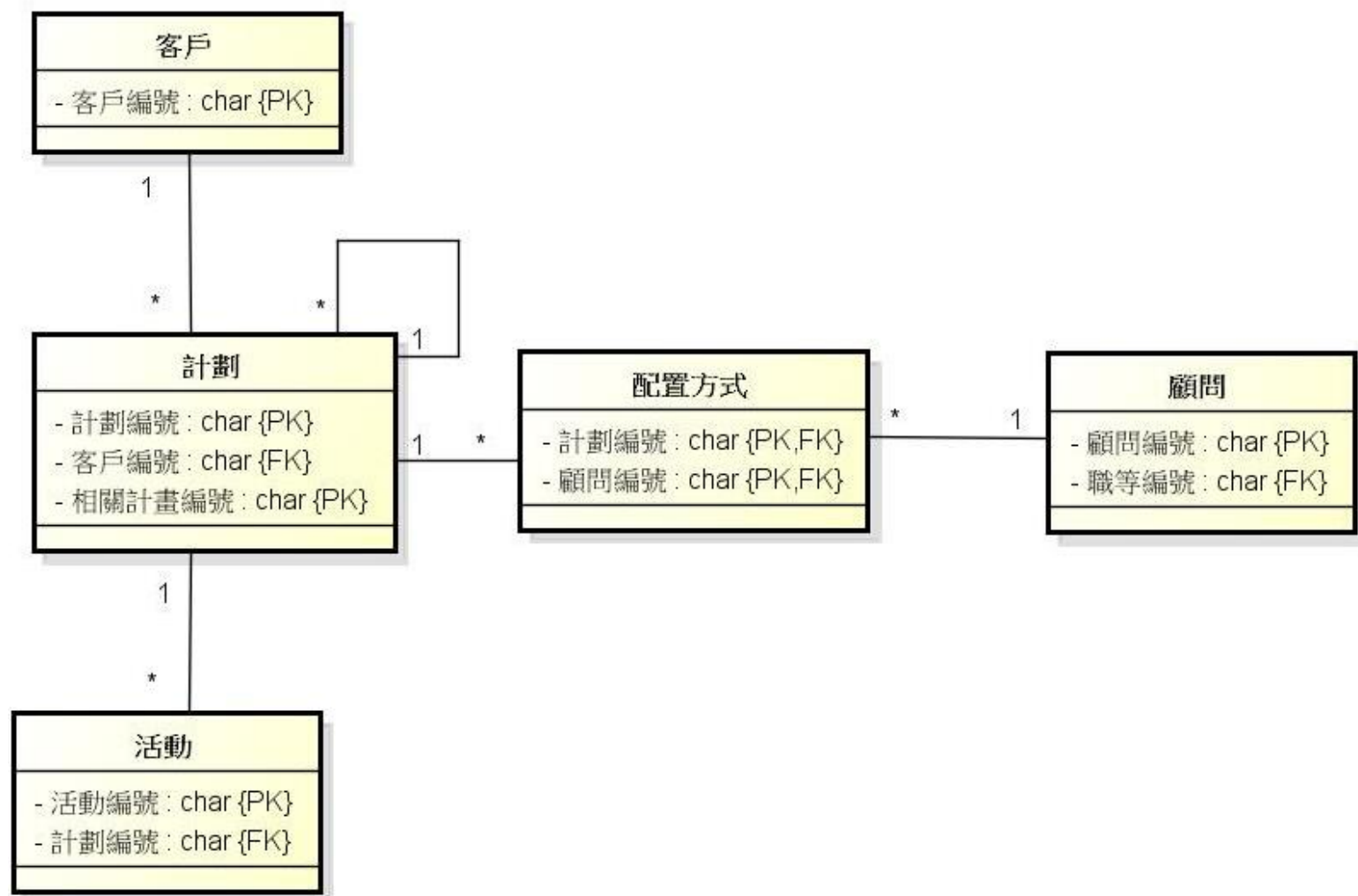


上頁圖合併成



4-3 處理一對一關係及多對多關係

多對多關係解決後之圖形



模組5：正規化

5-1 正規化概念

5-2 第一正規化

5-3 第二正規化及第三正規化

步驟六、正規化處理

- 透過一連串的檢視，讓每一個資料項目歸屬到適當的資料體中。
- 讓資料在新增、修改或刪除時，整個系統相關資料仍能維持正確良好的狀態。
- 具有關聯式之特性，但仍適用於其他資料模式。
- 正規化步驟：
 - 第一正規化
 - 第二正規化
 - 第三正規化

顧問明細表

編號	姓名	職等	薪資等級	座車等級	地址	專長代號	專長說明	專長資格
004	陳小期	D	S4	A	桃園	SK01	DB	DBA
						SK10	NW	CNI
						SK15	JAVA	JCP
009	張小誠	B	S2	F	台北	SK06	WCS	WCS
						SK10	NW	CNI
011	孫小華	E	S5	F	桃園	SK06	WCS	WCA
015	林小敏	E	S5	F	台北	SK01	DB	DBO
019	王小漢	B	S2	F	永和	SK05	NT	MCSE

未正規化形式UNF (Un-Normalized Form)

顧問編號

姓名

地址

職等

薪資級數

座車等級

專長代號

專長說明

專長資格

第一正規化形式FNF (First-Normalized Form)

第一正規化的規則：去除多值屬性

顧問編號

姓名

地址

職等

薪資級數

座車等級

專長代號

專長說明

專長資格

非多值部分

顧問編號

姓名

地址

職等

薪資級數

座車等級

多值部分

顧問編號

專長代號

專長說明

專長資格

重覆部分群由
原來PK加上
專長代號為PK

第二正規化形式SNF (Second-Normalized Form)

第二正規化的規則：去除只與部分主鍵相關欄位

非多值部分

顧問編號

姓名

地址

職等

薪資級數

座車等級

多值部分

顧問編號

專長代號

專長說明

專長資格

完全相依

顧問編號

專長代號

專長資格

部份相依

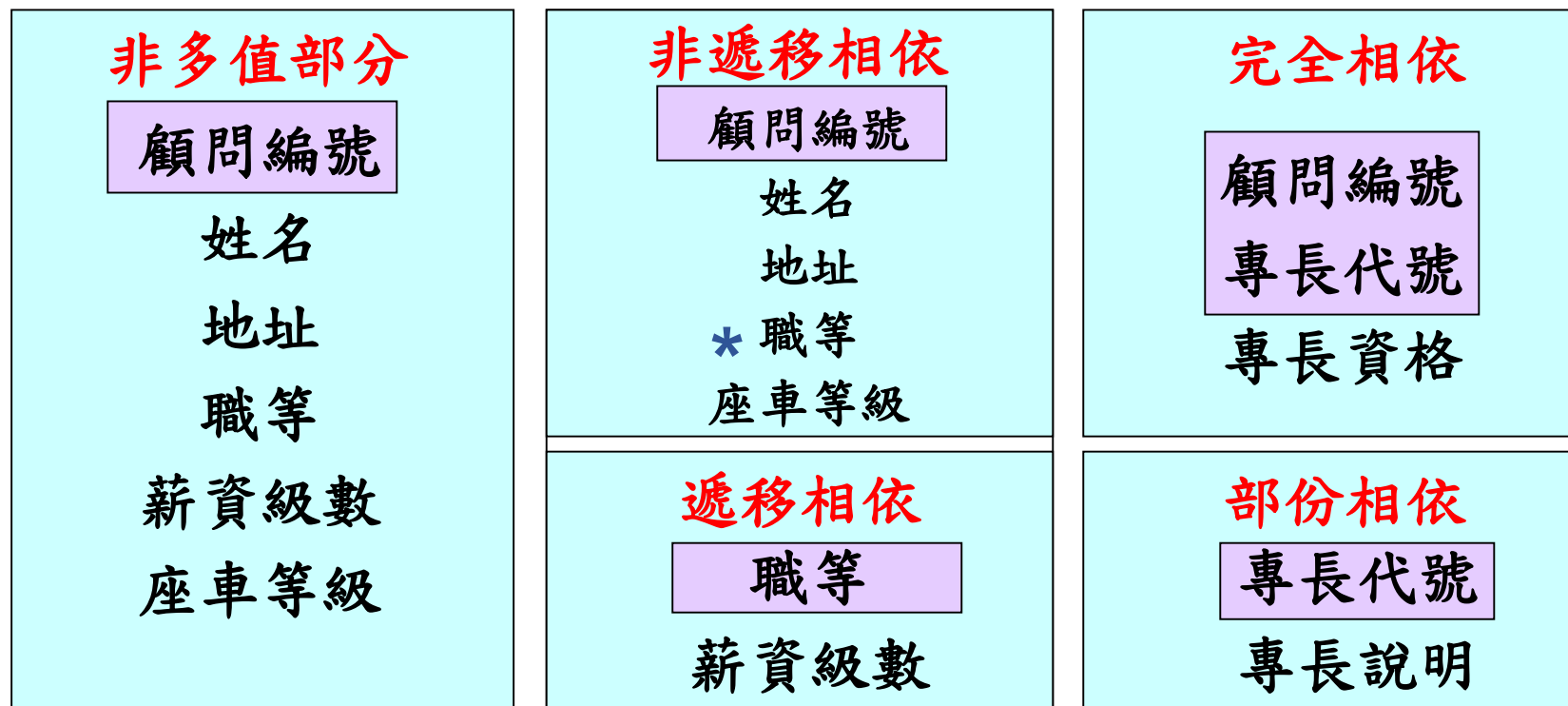
專長代號

專長說明

只與專長有關

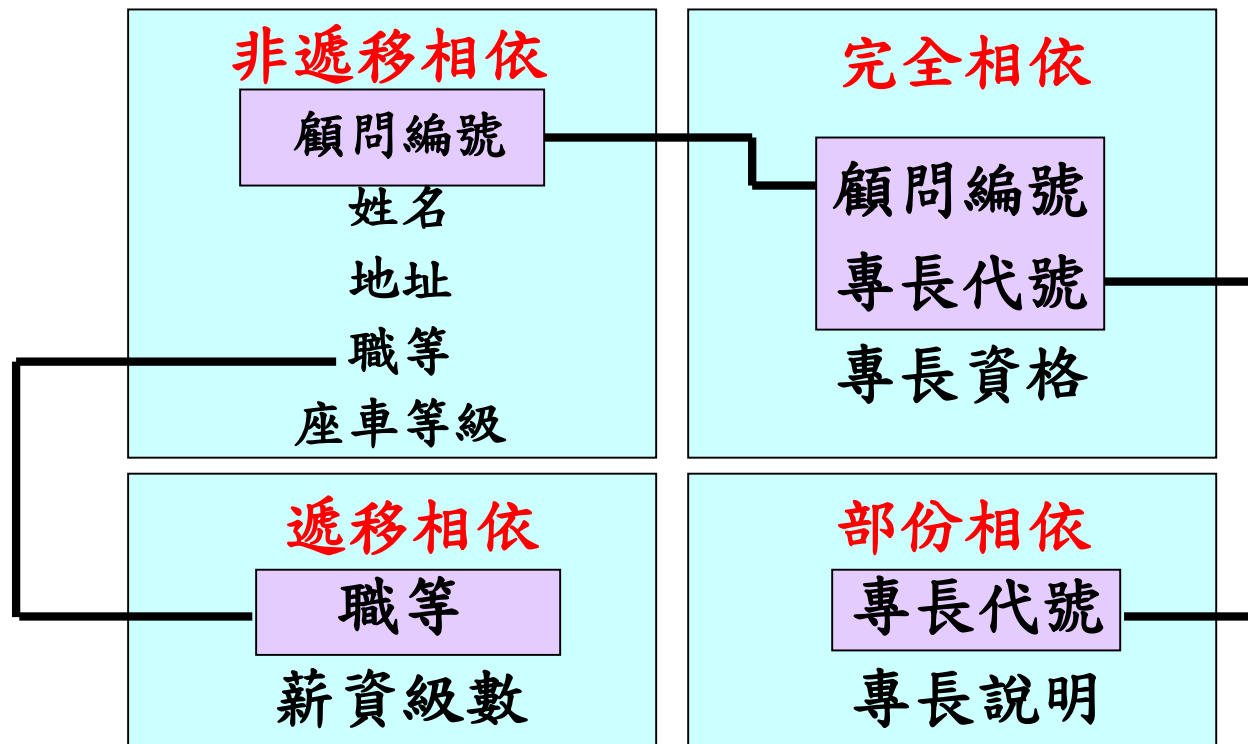
第三正規化形式TNF (Third-Normalized Form)

第三正規化的規則：去除除主鍵外有相依關係的欄位

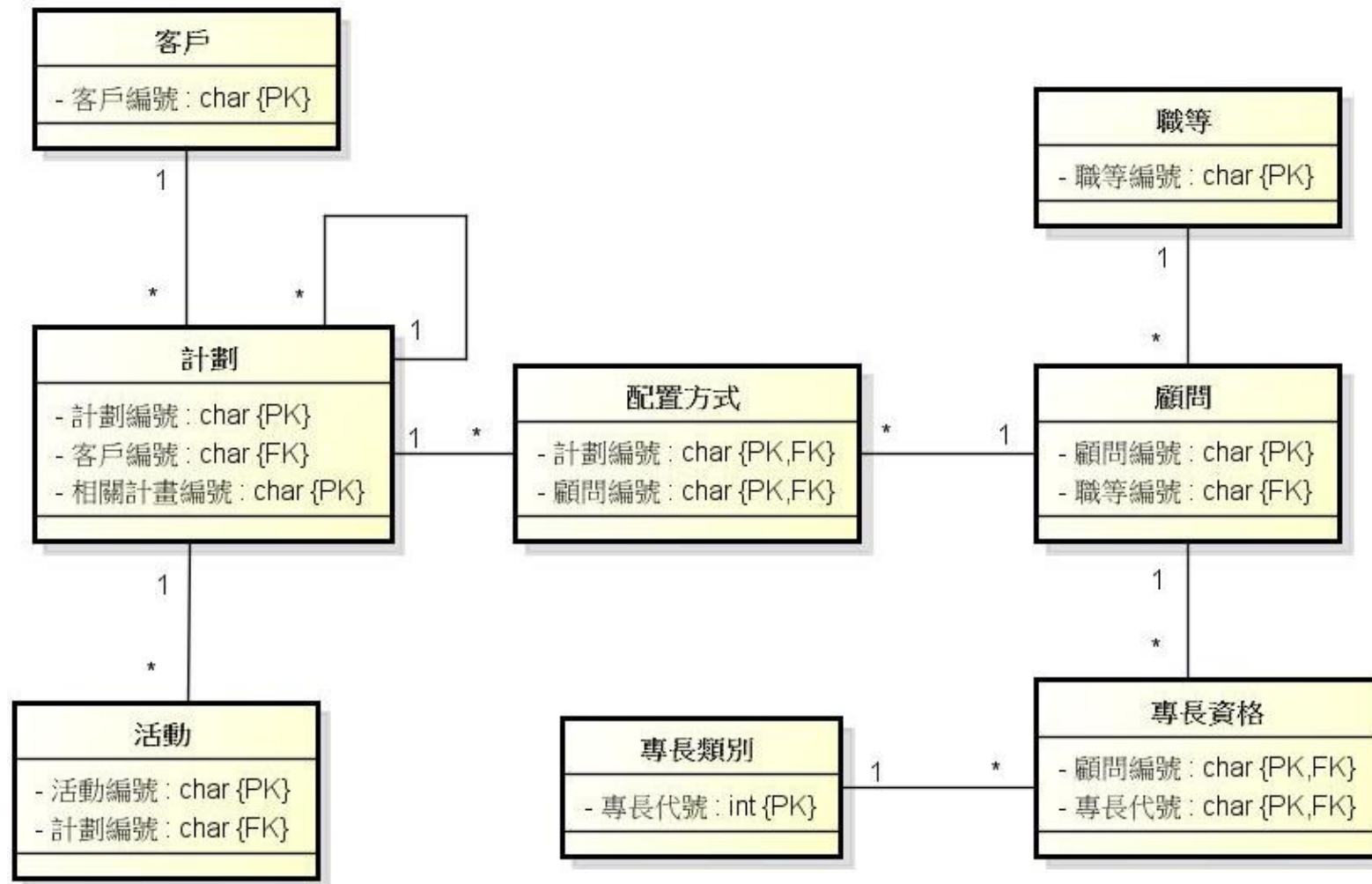


*表示FK

完成第三正規化形式



步驟七、確認 E-R 圖形的正確性



模組6：資料模型轉換 及 實體資料庫設計

6-1：資料模型轉換

6-2：撰寫表格規格

6-3：以SQL指令建置資料庫

步驟八、資料模型轉換

- 選定特定種類之資料庫管理系統(DBMS)
- 將實體關係模型轉換成此種類DBMS之模型
- 若選定的是關聯式模型則步驟如下：

實體 ⇨ 表格

屬性 ⇨ 欄位

關係 ⇨ FK (將「一」所代表表格之PK，複製到「多」所代表表格中，並設定成FK)

步驟九、表格規格描述

表格名稱	中文：職等 英文：OCCUPATION		索引鍵		
主鍵	OCC_NO		外來鍵		
欄位編號	欄位名稱	欄位敘述	資料型態	欄位長度	備註
1	OCC_NO	職等	Char	5	Not Null (PK)
2	OCC_SALARY	薪資級數	Int		Not Null

步驟九、表格規格描述

表格名稱	中文：顧問 英文：CONSULTANT		索引鍵		
主鍵	CON_NO		外來鍵	CON_OCCNO→OCCUPATION	
欄位編號	欄位名稱	欄位敘述	資料型態	欄位長度	備註
1	CON_NO	顧問編號	Char	10	Not Null (PK)
2	CON_NAME	姓名	Char	20	
3	CON_ADRS	地址	Varchar	65	
4	CON_OCCNO	職等	Char	5	(FK)
5	CON_CAR	座車級數	Char	3	

步驟九、表格規格描述

表格名稱	中文：專長資格 英文：EXPERT		索引鍵		
主鍵	EXP_CONNO+EXP_NO		外來鍵	EXP_CONNO→CONSULTANT EXP_NO→EXPSPECT	
欄位編號	欄位名稱	欄位敘述	資料型態	欄位長度	備註
1	EXP_CONNO	顧問編號	Char	10	Not Null (FK)
2	EXP_NO	專長代號	Char	6	Not Null (FK)
3	EXP_OWN	專長資格	Varchar	30	

步驟九、表格規格描述

表格名稱	中文：專長說明 英文：EXPSPECT		索引鍵		
主鍵	SPE_NO		外來鍵		
欄位編號	欄位名稱	欄位敘述	資料型態	欄位長度	備註
1	SPE_NO	專長代號	Char	6	Not Null (PK)
2	SPE_SPEC	專長說明	Char	30	

步驟十、SQL命令建構資料庫的Table

建立職等Table

```
CREATE TABLE OCCUPATION (  
    OCC_NO    CHAR(5) NOT NULL ,  
    OCC_SALARY INT,  
    PRIMARY KEY(OCC_NO) );
```

建立顧問Table

```
CREATE TABLE CONSULTANT (  
    CON_NO    CHAR (10) NOT NULL ,  
    CON_NAME  CHAR (20),  
    CON_ADRS  VARCHAR (65) ,  
    CON_OCCNO CHAR (5) ,  
    CON_CAR   CHAR (3) ,  
    FOREIGN KEY (CON_OCCNO) REFERENCES OCCUPATION(OCC_NO)  
    ON DELETE RESTRICT ON UPDATE RESTRICT ,  
    PRIMARY KEY (CON_NO) );
```

建立之後資料庫管理系統對資料異動會自動做
資參考完整性檢查

步驟十、SQL命令建構資料庫的Table

建立專長說明Table

```
CREATE TABLE EXPSPECT (  
    SPE_NO    CHAR (6) NOT NULL ,  
    SPE_SPECE CHAR (30),  
    PRIMARY KEY (SPE_NO)    );
```

建立專長資格Table

```
CREATE TABLE EXPERT (  
    EXP_CONNO CHAR (10) NOT NULL ,  
    EXP_NO     CHAR (6)  NOT NULL ,  
    EXP_OWN    VARCHAR (30),  
    FOREIGN KEY (EXP_CONNO) REFERENCES CONSULTANT(CON_NO)  
    ON DELETE RESTRICT ON UPDATE RESTRICT ,  
    FOREIGN KEY (EXP_NO) REFERENCES EXPSPECT(SPE_NO)  
    ON DELETE RESTRICT ON UPDATE RESTRICT ,  
    PRIMARY KEY(EXP_CONNO,EXP_NO)    );
```