資訊管理學系 陳士杰老師

資料庫系統管理
Database System Management

正規化的概念與應用
Normalization

國立聯合大學
NATIONAL UNITED UNIVERSITY

Outlines

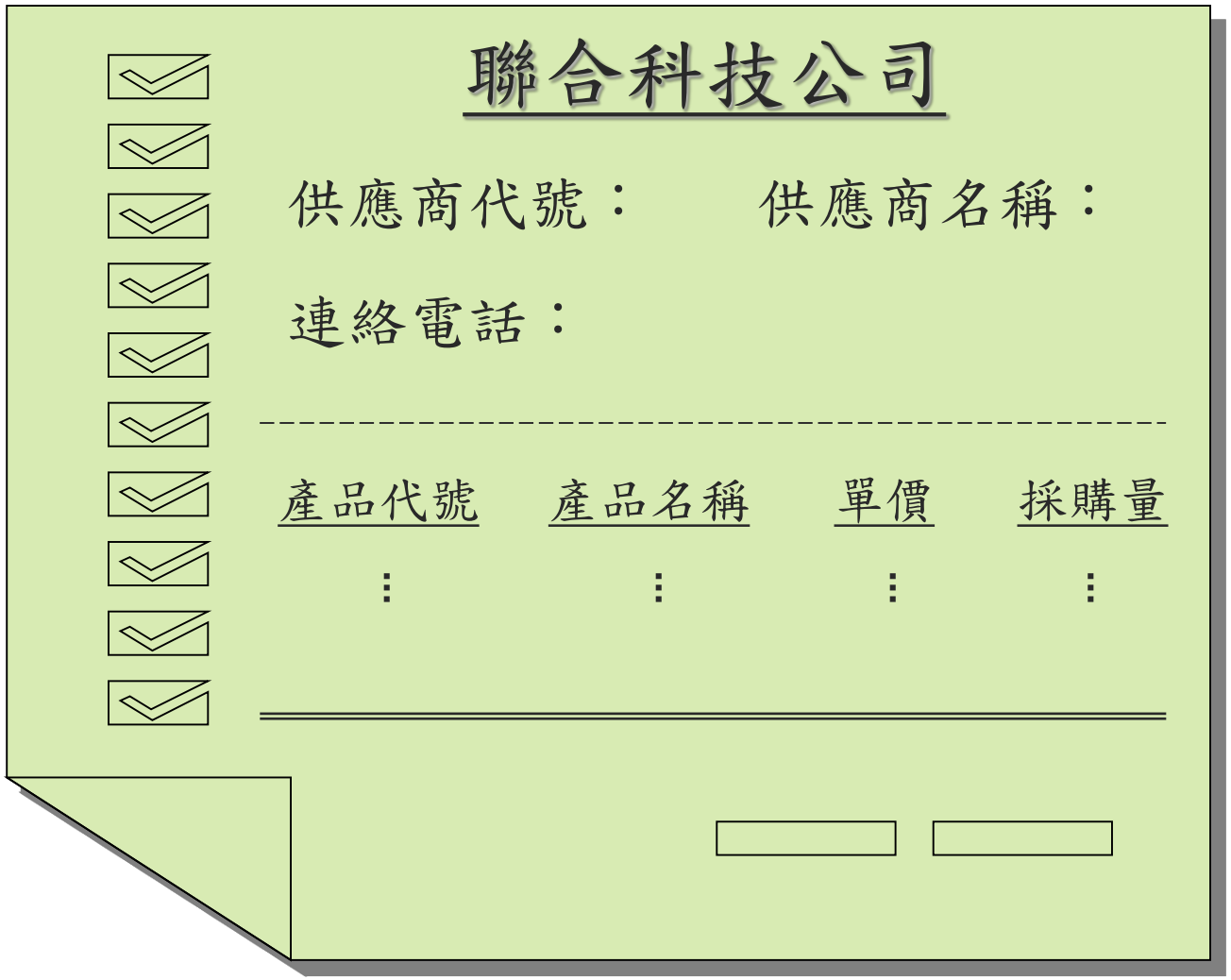
- 為什麼要正規化(Normalization)？
- 功能相依性 (Functional Dependency, FD)
- 正規化(Normalization)
- 正規化的過程
- 如何由E-R Model從事正規化
- 總結

【講義：Ch. 4】
【原文：Ch. 10】

■ 為什麼要正規化(Normalization)？

- 設計資料庫的相關表格時，常見的兩種錯誤作法：
  - 直接將公司表單設計成表格
  - 將E-R Diagram中的所有個體及其屬性直接設計成表格
- 上述兩種作法所設計出來的表格皆未經正規化，實際存放或操作資料時，會產生下述問題：
  - 容易有不必要的資料重覆儲存情形
  - 資料在做插入、刪除或更新動作時產生異常(Anomalies)情形

某公司採購資料表單



公司對某供應商之採購資料表

供應商代號	供應商名稱	聯絡電話	產品代號	產品名稱	單價	採購量
12345	連店	0800000XXX	p147 s201	X檯燈 A-101電池	1000 150	10 300
02314	台雞店	0800111△△△	s201 s199	A-101電池 插座	160 20	100 100
22138	廣答				750	20

- **更新**產品 」。但因為資料不只一筆 **不一致**。
- **插入**代號 允許!! 除非此供應商已賣
- **刪除**「u 答」之資訊。

- 上述的問題，主要是因為同一份表格內的所有Attribute之間，有許多不同類型的**功能相依性(Functional Dependence; FD)**存在。
- 正規化主要是對表格做**分割**的動作
  - 在資料正規化的過程中，每個階段都是以不同類型的**相依性(Dependence)**做為**表格處理與切割**的依據。
  - 將Attribute之間的關係，分散到不同表格。
- 正規化的目的：
  - 降低**資料重覆性(Data Redundancy)**
  - 避免產生插入、刪除或更新可能的**異常(Anomalies)**

功能相依性 (Functional Dependency, FD)

- 假如 R(A1, A2,…, An) 代表一個關聯，其中X 與 Y 為關聯 R 中屬性的子集合，則Y功能相依於X可以利用符號寫成：

$Y \propto X$ (Y 功能相依於 X)

或

$X \rightarrow Y$ (X 決定 Y)

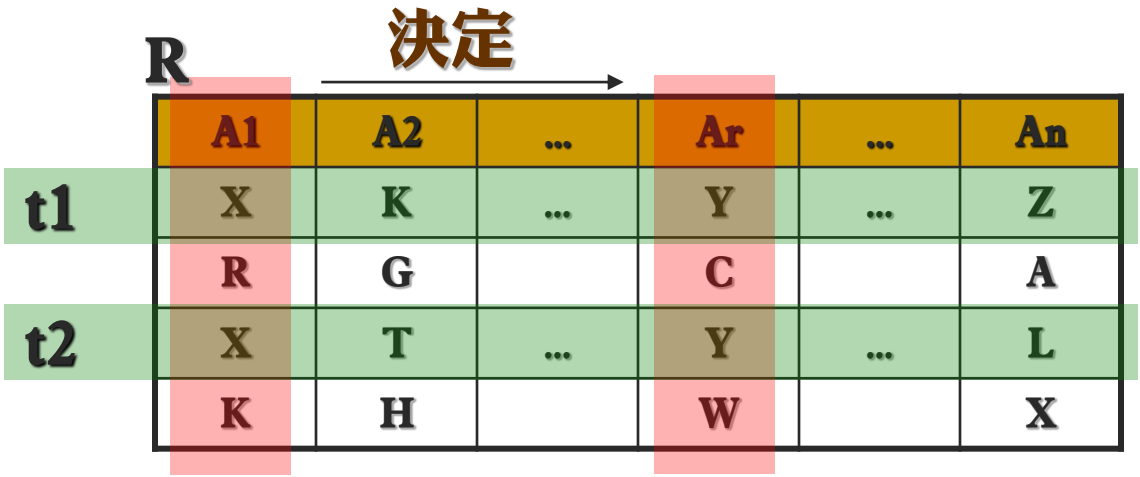
R

A1	A2	…	Ar	…	An
X	K	…	Y	…	Z
R	G		C		A
X	T	…	Y	…	L
K	H		W		X

其中，X為決定因素 (Determinant)或left-hand side，Y為相依因素 (Dependent)或right-hand side

- 一個關聯表格中的數個屬性間之功能相依性可能如:
  - 所在地可決定分公司電話區碼 (即: 所在地→區碼)
  - 地址所在之區域可決定郵遞區號
  - 員工編號可決定員工姓名 (即: 員工編號→姓名)

- **Def**：給定一個關聯R，R的屬性子集Y功能相依於R的屬性子集X, 則：
  - 若且唯若(if and only if) 無論何時R的兩個Tuples若**有相同的X值**時，則**必有相同的Y值**。
  - 對於關聯R中的每個X值，均有唯一的Y值來對應。
- Ex: Ar功能相依於A1 (A1→Ar)

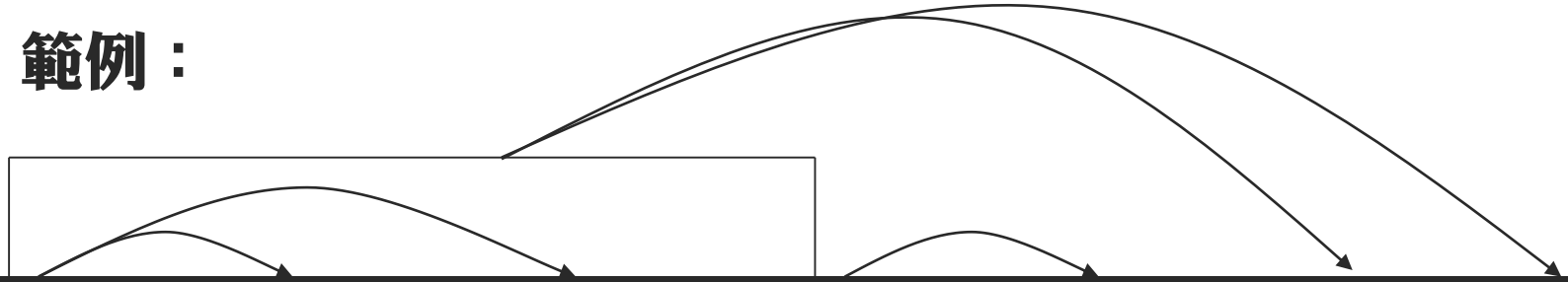


- 特性：
  - 若在關聯R中X→Y成立，並不表示Y→X也成立
  - 功能相依性是一個多對一的關係
    - 多個不同的決定因素X可能決定出一個相依因素，反之則否。
  - 關聯R中的X若為主鍵(候選鍵)，則關聯R中所有其它屬性必功能相依於X
- 找出功能相依性是從事正規化的第一步。

- 在一個關聯上，功能相依的表示方式：

屬性y**功能相依於**屬性x，或稱x**決定了**y (x→y)

- 範例：



供應商代號	供應商名稱	聯絡電話	產品代號	產品名稱	單價	採購量
12345	連店	0800000XXX	p147 s201	X檯燈 A-101電池	1000 150	10 300
02314	台雞店	0800111△△△	s201 s199	A-101電池 插座	160 20	100 100
22138	廣答	0800123〇〇〇	u001	電鍋	750	20

- {供應商代號}→{供應商名稱，聯絡電話}
- {產品代號}→{產品名稱}
- {供應商代號，產品代號}→{單價，採購量}

Trivial(沒價值)及Non-Trivial(非沒價值)相依性

- **沒價值 (Trivial)**
  - 若功能相依右邊之相依因素，為左邊決定因素的部份集合時，稱此功能相依為沒價值的。
  - Ex: {供應商代號，產品代號}→供應商代號
- **非沒價值 (Non-Trivial)**
  - 若功能相依右邊之相依因素，不為左邊決定因素的部份集合時，稱此功能相依為非沒價值的。
  - Ex: {供應商代號，產品代號}→單價
  - Non-Trivial的功能相依性，在從事正規化時才具有意義。

功能相依性的種類

- 功能相依性有幾種型式：
  - 完全功能相依 (Full Functional Dependency)
  - 部份功能相依 (Partial Functional Dependency)
  - 遞移相依 (Transitive Dependency)
- 鍵值屬性(Key Attribute)
  - 能構成主鍵或候選鍵的所有屬性。反之，則稱為非鍵值屬性(Non-Key Attribute)。
  - 在實務上，鍵值屬性通常是指主鍵。

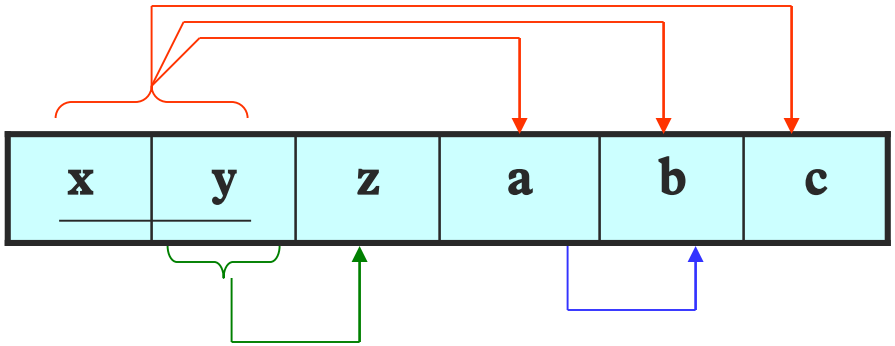
- Key Attribute範例：

學號	姓名	系別	年級	生日	地址	身份証字號
001	張三	資管	2	3.18	台北	F11111111
002	李四	企管	3	3.19	台中	M22222222
003	王五	人管	4	3.20	台南	K33333333

  - **鍵值屬性**：學號，身份証字號
  - **非鍵值屬性**：姓名，系別，年級，生日，地址

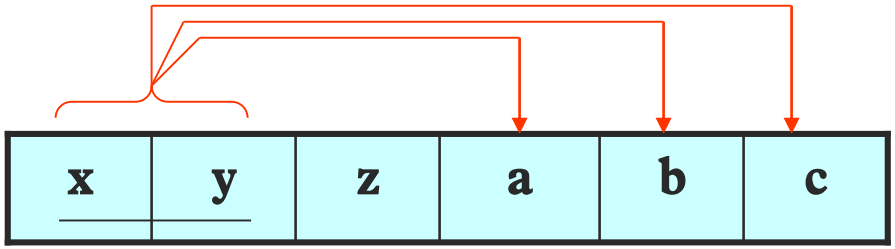
功能相依性範例

- 假設現在有下面的相依性範例：



完全功能相依 (Full Functional Dependency)

- 若主鍵是由**多個屬性**組成，且某非鍵值屬性依賴主鍵之**全部**而非部分時，則稱該非鍵值屬性**“完全相依”於主鍵**。
- 例如：

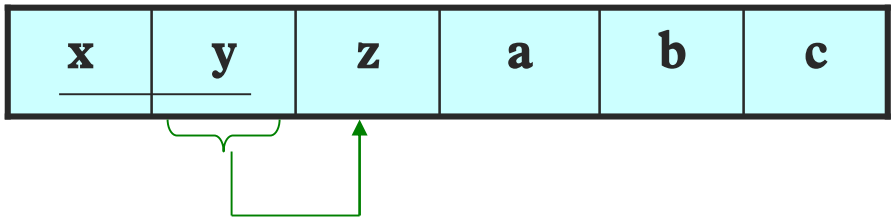


屬性a, b, c**完全相依**於由屬性x, y所共同構成的主鍵。

- 若主鍵**僅由一個屬性**所組成，則任一非鍵值屬性必“完全相依”於主鍵。

部份功能相依 (Partial Functional Dependency)

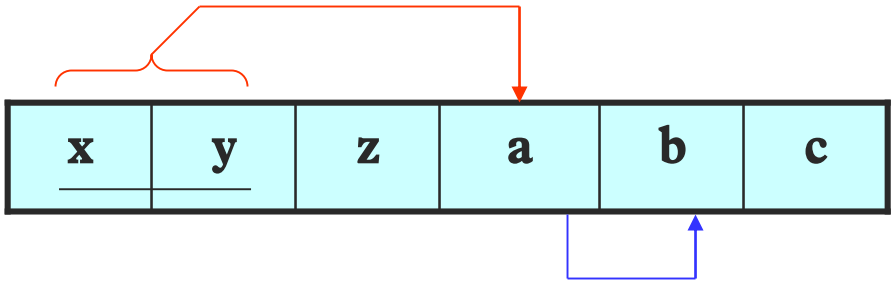
- 若主鍵是由多個屬性組成，而某非鍵值屬性是依賴主鍵之部分屬性時，則稱該非鍵值屬性“部份相依”於主鍵。
- 例如：



屬性z與屬性x, y所構成的主鍵部份相依。

遞移相依 (Transitive Dependency)

- 若存在一個**非鍵值屬性 T**，使得 **I→T** 且 **T→J** 的功能相依性均成立，則稱屬性 I與屬性 J之間具有**遞移相依**。
- 例如：

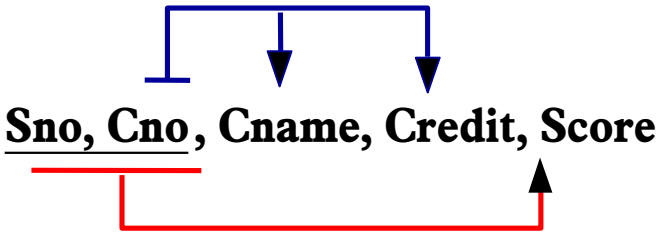


{x,y} → b是透過非鍵值屬性a所產生的遞移

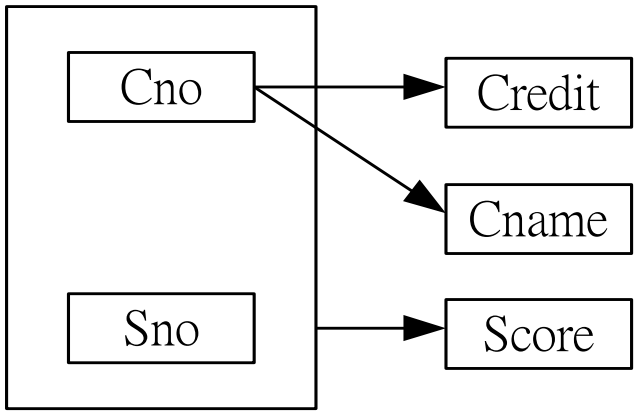
功能相依圖(FD Diagram)

- 學生選課資料的功能相依性(FD) 為：

(Sno, Cno) → Score
Cno → Cname
Cno → Credit



其功能相依圖如圖:



學生選課資料的功能相依圖

功能相依性的推導法則－阿姆斯壯定理 (Armstrong's Axioms)

- 目的：利用已知的功能相依性，透過一些性質，推導出以下可能項目：
  - 其它**隱含(Implicit) 的功能相依性**。
  - 一個關聯中的**候選鍵**或**主鍵**、
  - **最簡功能相依性(Irreducible Functional Dependence)**
- 最簡功能相依性的特性：
  1. 相依因素僅有一個屬性
  2. 沒有多餘的決定因素
  3. 沒有多餘的功能相依性

- 假設A, B, C, D為關聯R的四個任意屬性子集，且AB表示A聯集B。則阿姆斯壯定理如下：
  - 反身性(Reflexivity)：若B是A的子集合，則A→B
  - 擴張性(Augmentation)：若A →B，則AC →BC
  - 遞移性(Transitivity)：若A →B且B →C，則A →C
  - 自身決定性(Self-determination)：A →A
  - 分解性(Decomposition)：若A →BC，則A →B且A →C
  - 聯集性(Union)：若A →B且A →C，則A →BC
  - 合成性(Composition)：若A →B且C →D，則AC →BD
  - 虛擬遞移性(Pseudo-transitivity): 若A →B且BC →D，則AC →D

- 例：某一關聯R={A, B, C, D}，且有以下功能相依：

  {C→BD, B→D, C→B, BC→D, CD→A}

  請找出最簡功能相依性，並決定R的鍵值為何。

- Ans:

  ① 相依因素僅有一個屬性

  - 透過分解性，將帶有一個以上屬性之相依因素分解掉。故可得到

    {C→B, C→D , B→D, C→B, BC→D, CD→A}

  - 上述功能相依集合中， C→B有重覆，保留一個即可!!故可得到

    {C→B, C→D , B→D, BC→D, CD→A}

② **沒有多餘的決定因素**

- ∵ C→D與CD→A呈虛擬遞移，可得到C→A 。∴上述功能相依可改為：

  {C→B, C→D , B→D, BC→D, C→A}

- ∵ C→B與BC→D呈虛擬遞移，可得到C→D 。∴上述功能相依可改為：

  {C→B, C→D , B→D, C→D, C→A}

- 上述功能相依集合中， C→D有重覆，保留一個即可!!故可得到

  {C→B, C→D , B→D, C→A}

③ **去除多餘的功能相依**

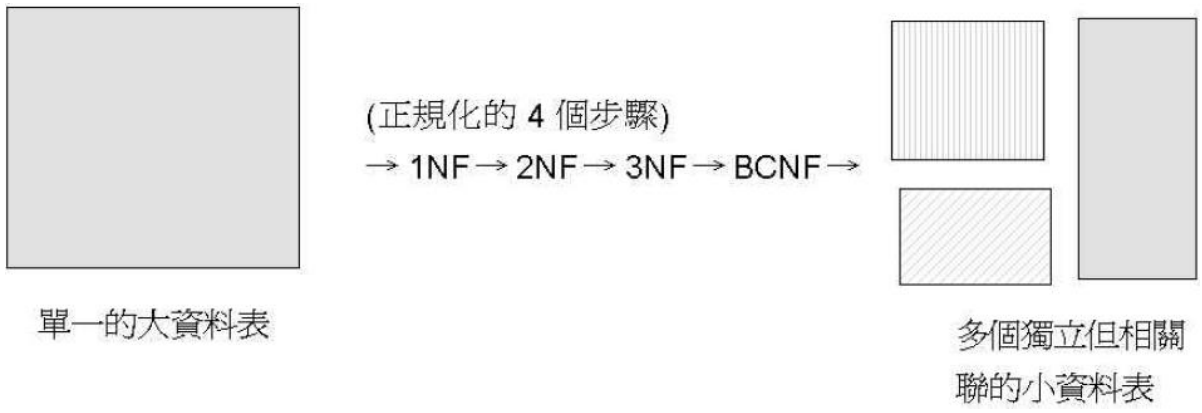
- ∵由遞移性得知，C→D可由C→B與B→D得到。∴上述功能相依可改為:

  {C→B, B→D, C→A}

- 由上述推導得知，最簡相依性是{C→B, B→D, C→A}。且因為屬性C可以直接或間接決定其它所有屬性， ∴C是鍵值。

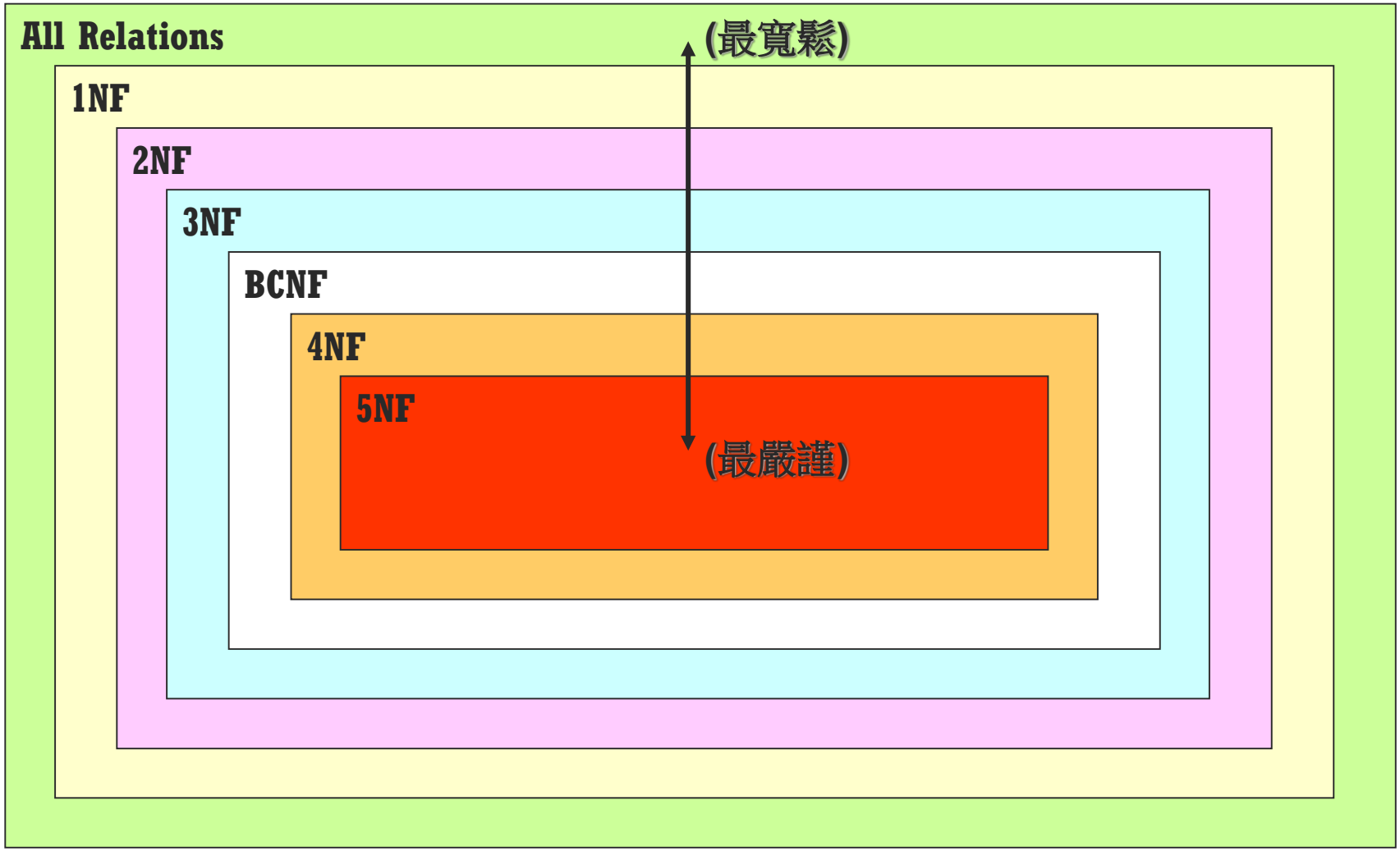
■ 正規化(Normalization)

- 正規化的理論首先由 E. F. Codd 於 1971 年提出，目的是用來設計「良好」的關聯式資料模式。
  - 原因：解決**資料重覆**及一些**異常**現象。
  - 方法：根據不同的相依性問題來**分割關聯**
  - 前題：分割後的關聯不能有**資訊遺失**的情況(無損失分解；Lossless decomposition)



正規化的類型

- 基本上，正規化是以漸進的方式逐步地進行表格分割，每一個步驟都會滿足某一項「正規化型式」的條件。
- 正規化的型式有以下幾種：
  - 低階正規化
    - 第一正規化型式 (1NF; First Normal Form)
    - 第二正規化型式 (2NF; Second Normal Form)
    - 第三正規化型式 (3NF; Third Normal Form)
    - BC正規化型式 (BCNF; Boyce-Codd Normal Form)
  - 高階正規化 (比較少用)
    - 第四正規化型式 (4NF; Forth Normal Form)
    - 第五正規化型式 (5NF; Fifth Normal Form)

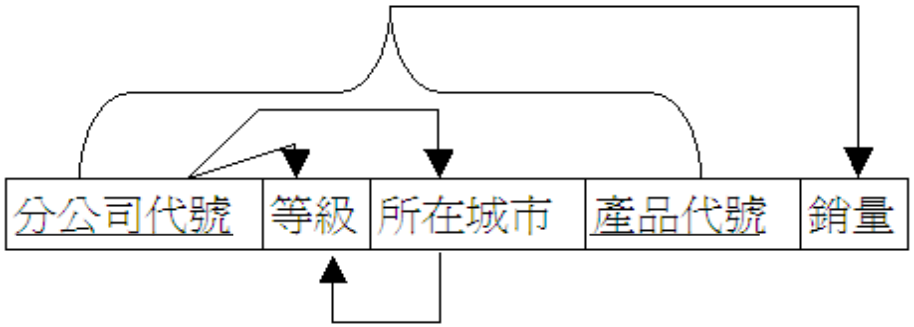

All Relations
(最寬鬆)
1NF
2NF
3NF
BCNF
4NF
5NF
(最嚴謹)

正規化過程


未正規化的關聯
【除去非基元性(Non-atomic)資料項目】
第一正規化 1NF
所有記錄中的項目都是基元值
【除去部份功能相依性】
第二正規化 2NF
滿足1NF且每一個非鍵屬性完全相依於主鍵
【除去遞移相依性】
第三正規化 3NF
滿足2NF且每一個非鍵屬性非遞移相依於主鍵
【除去其它功能相依性所引起的異常】
Boyce-Codd正規化 BCNF
每一個決定因素(Determinant)皆是候選鍵
【除去多值相依性】
第四正規化 4NF
滿足BCNF且無Multivalued Dependency現象
【除去合併相依性】
第五正規化 5NF
又稱為Projection-Join Normal Form; PJ/NF

- 理論上，表格經由正規化分割得愈細，愈可避免資料重覆及一些更新上的異常現象。然而，在實務上，分割太細的表格，可能會產生以下問題：
  - 在查詢資料時，可能會用到許多**合併 (Join)** 運算，此運算易造成查詢動作非常沒有效率。
  - 可能產生**有損分解**。
- 因此，在實作上，為**“保持執行效率”** 與 **“避免異常”** 間的平衡，通常正規化進行到**3NF** (最多至**BCNF**) 即可。

- **假設有一個非正規化表格與它的屬性間之相依關係如下：**



分公司代號	等級	所在城市	產品代號	銷量
S1	20	台北	P1	300
			P2	200
			P3	400
			P4	200
			P5	100
			P6	100
S2	10	高雄	P1	300
			P2	400
S3	10	高雄	P2	200
S4	20	台北	P2	200
			P4	300
			P5	400

第一正規化(First Normal Form，1NF)

- 定義：一個關聯表為**第一正規化表格**，若且唯若關聯表中的每一個屬性其值皆為**基元值 (Atomic Value)**。
- 違反基元值可能情況：
  - 多值屬性
  - 複合屬性
- 作法：
  - 多值屬性：分解成多個值組(Tuples)
  - 複合屬性：拆為簡單屬性

1個Tuple

分公司代號	等級	所在城市	產品代號	銷量
S1	20	台北	P1 P2 P3 P4 P5 P6	300 200 400 200 100 100
S2	10	高雄	P1 P2	300 400
S3	10	高雄	P2	200
S4	20	台北	P2 P4 P5	200 300 400

多值屬性

- **問題：**
  - **產生資料重複性**
- **需要靠2NF, 3NF, BCNF來降低重複性。**

6個Tuple

分公司代號	等級	所在城市	產品代號	銷量
S1	20	台北	P1	300
S1	20	台北	P2	200
S1	20	台北	P3	400
S1	20	台北	P4	200
S1	20	台北	P5	100
S1	20	台北	P6	100
S2	10	高雄	P1	300
S2	10	高雄	P2	400
S3	10	高雄	P2	200
S4	20	台北	P2	200
S4	20	台北	P4	300
S4	20	台北	P5	400

- 【補充範例】複合屬性執行第一正規化：

學號	姓名	系別	年級	生日	地址	身份証字號
001	張三	資管	2	3.18	台北	F11111111
002	李四	企管	3	3.19	台中	M22222222
003	王五	人管	4	3.20	台南	K33333333

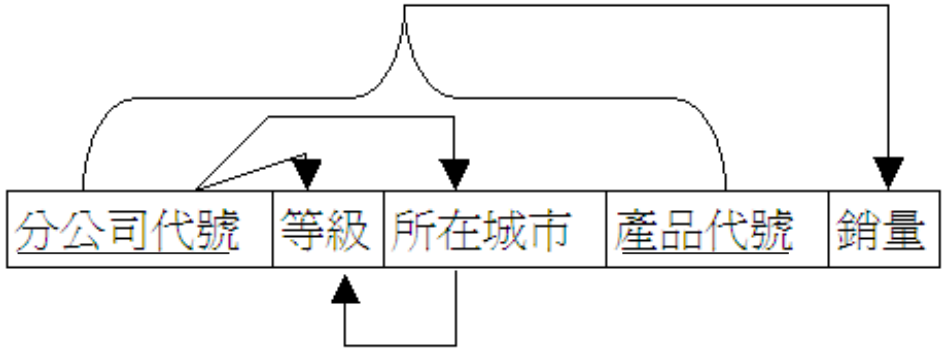
學號	姓名	系別	年級	B_月	B_日	地址	身份証字號
001	張三	資管	2	3	18	台北	F11111111
002	李四	企管	3	3	19	台中	M22222222
003	王五	人管	4	3	20	台南	K33333333

第一正規化型式所產生的異常 (Anomalies)

- **新增記錄時 (輸入資料時，必須等主鍵輸入才可進行)**
  - 當有一新的分公司加入時，但未有任何交易經歷時，將造成**產品代號為NULL**。
- **更新資料時 (需要一起修改許多相關的值組)**
  - 若單一分公司有多筆記錄在檔案中，此分公司**若有資料異動，則必須更動多筆資料**。如此不僅浪費空間，而且更新時亦浪費時間，也**容易造成資料不一致**的情況。
- **刪除記錄時 (刪除資料時，會把過多的資訊刪掉)**
  - 若某分公司只有一筆交易經歷，若我們要刪除此分公司的交易資料時，則將連帶刪除該分公司資訊。

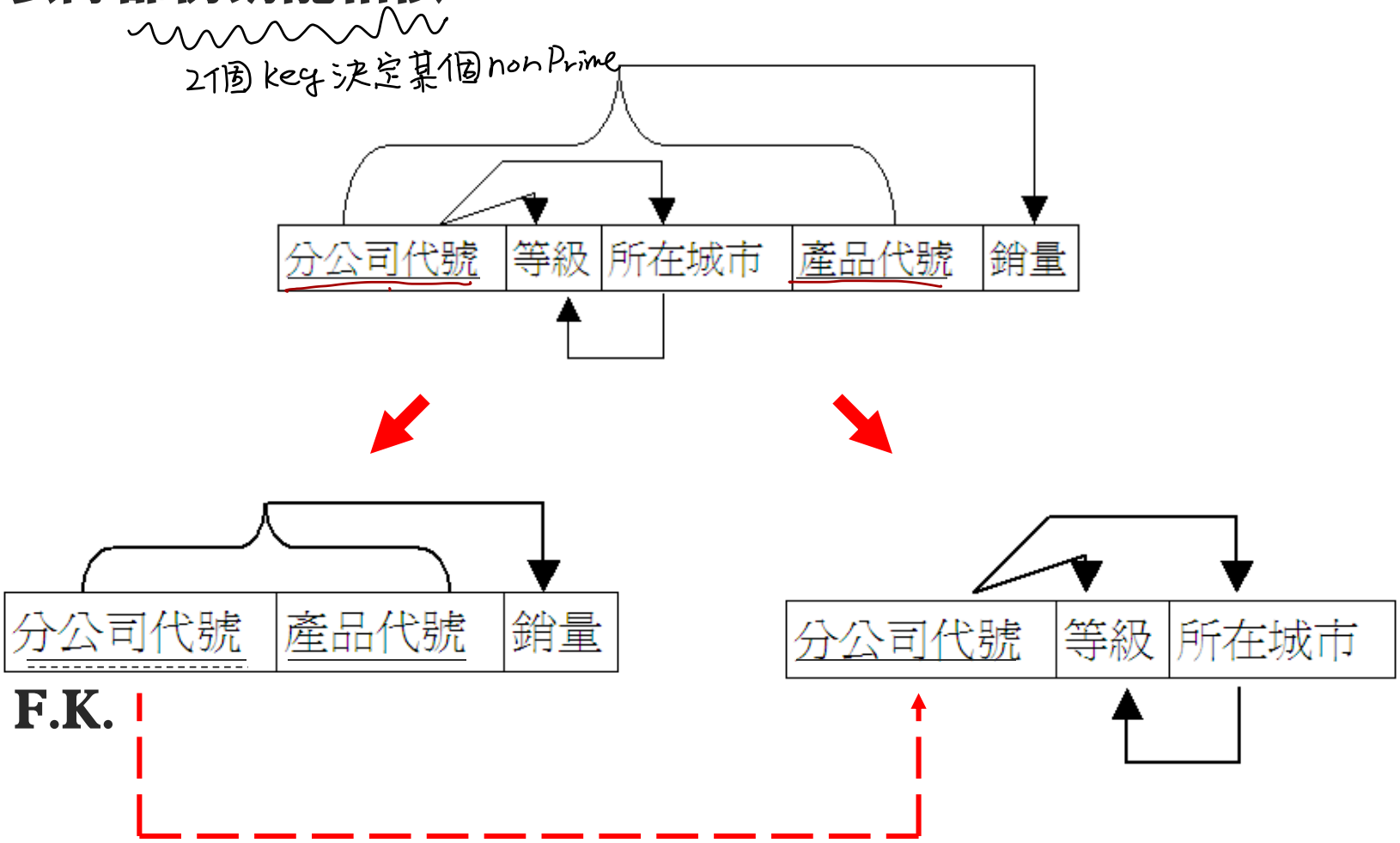
第二正規化(Second Normal Form，2NF)

- 定義：一個關聯表為第二正規化表格，若且唯若關聯表中，所有非鍵值屬性皆完全功能相依於主鍵



- 作法：
  - 拆解關聯：將部份相依於主鍵之非鍵值屬性，與所屬之決定因素複製以建立新的關聯；另外，將部份相依性之相依因素自原關聯刪除。
  - 決定外來鍵

去除部份功能相依


2個 key 決定某個 non Prime
分公司代號
等級
所在城市
產品代號
銷量
分公司代號
產品代號
銷量
F.K.
分公司代號
等級
所在城市

分公司代號	等級	所在城市
S1	20	台北
S2	10	高雄
S3	10	高雄
S4	20	台北

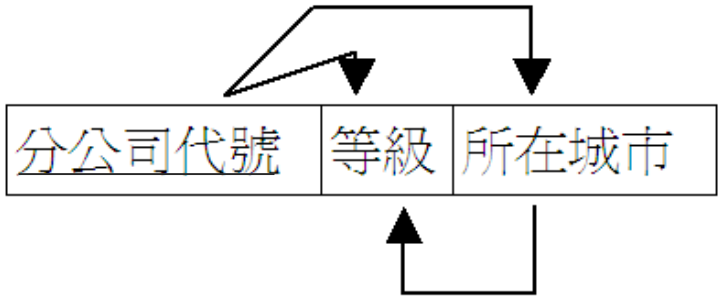
分公司代號	產品代號	銷量
S1	P1	300
S1	P2	200
S1	P3	400
S1	P4	200
S1	P5	100
S1	P6	100
S2	P1	300
S2	P2	400
S3	P2	200
S4	P2	200
S4	P4	300
S4	P5	400

第二正規化型式所產生的異常 (Anomalies)

- **新增記錄時 (輸入時必須等主鍵輸入才可進行)**
  - 當公司想評估一個新的點時(如：苗栗)，無法加入這個新的點的評估等級，除非已經在那裡設分公司了(∵在當地沒有實際設公司，沒有主鍵值可以對應!!)。
- **更新資料時 (需要一起修改許多相關的值組)**
  - 如果要把高雄的評等做更動，就要先將所有高雄的記錄給找出來(∵高雄的資料有很多筆，怕有些會沒改到，產生資料不一致)。
- **刪除記錄時 (刪除時會把過多的資訊刪掉)**
  - 假設高雄分公司的評等資料只有一筆。如果要刪除高雄分公司的資料時，則必須連帶地刪除高雄的評估資料 (∵遞移相依)，如果以後還想在高雄設點，就要重新做評估了。

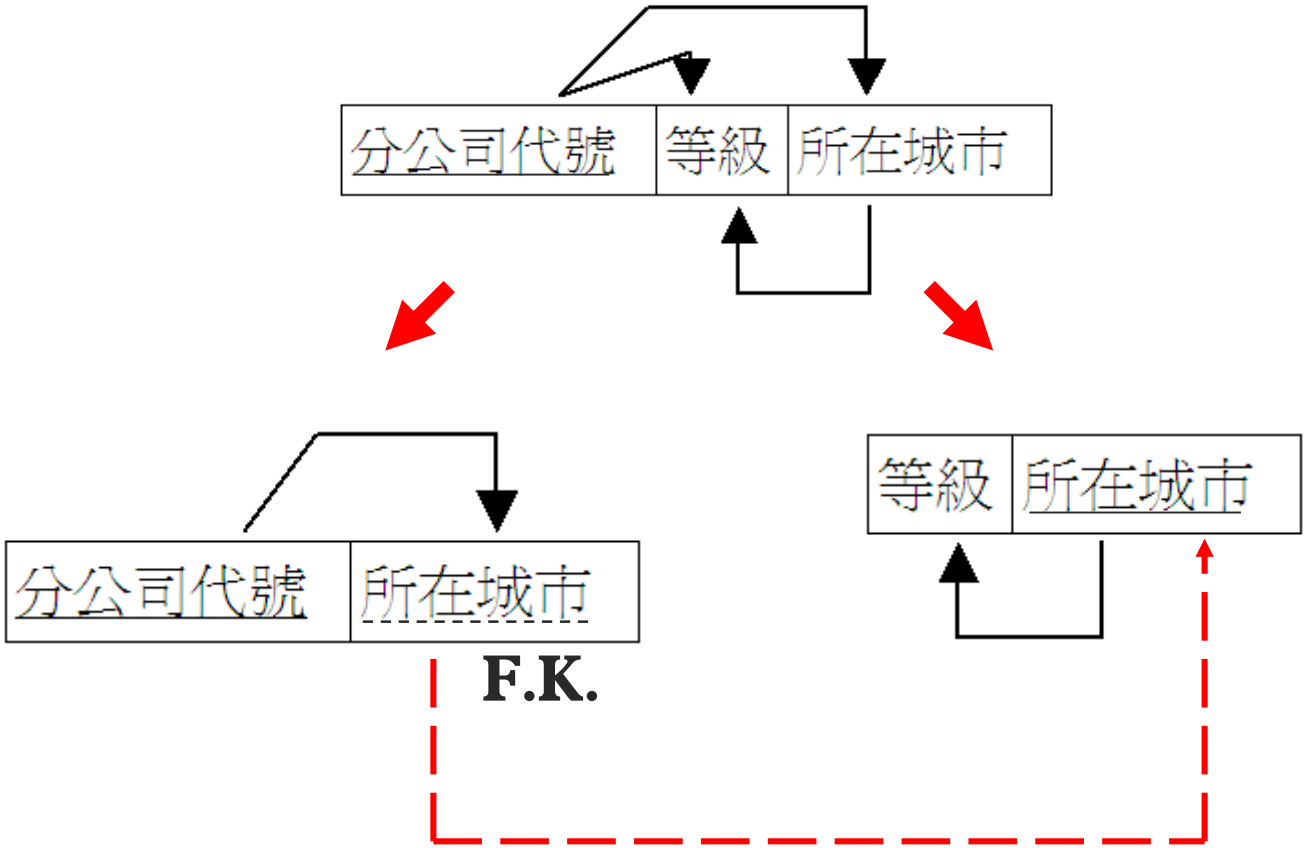
第三正規化(Third Normal Form，3NF)

- 定義：一個關聯表為第三正規化表格，若且唯若該關聯表中，不存在非鍵值屬性遞移相依於主鍵
  - (即：非鍵值屬性間，不存在著功能相依性)。



- 作法：
  - 拆解關聯：將遞移相依中，非鍵值之決定因素與相依因素複製，以建立新的關聯；另外，將前述之非鍵值相依因素，自原關聯中刪除。
  - 決定外來鍵

- 去除(非主鍵屬性)遞移功能相依。



分公司代號	所在城市
S1	台北
S2	高雄
S3	高雄
S4	台北

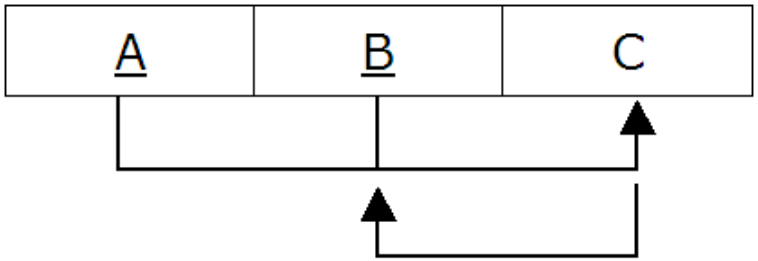
等級	所在城市
20	台北
10	高雄

分公司代號	產品代號	銷量
S1	P1	300
S1	P2	200
S1	P3	400
S1	P4	200
S1	P5	100
S1	P6	100
S2	P1	300
S2	P2	400
S3	P2	200
S4	P2	200
S4	P4	300
S4	P5	400

Boyce-Codd Normal Form(BCNF)

- 是由Boyce和Codd於1974年所提出來的3NF的改良式。其條件比3NF更加嚴苛。因此每一個符合BCNF的關聯一定也是3NF。
- 一個關聯R為 BCNF 表格，若且唯若該關聯表中的功能相依性之決定因素都是候選鍵。
  - 即：關聯R中，每個功能相依Xi→Yi中，Xi為R之候選鍵。
- 如果主鍵 (Primary Key)為單一屬性時，就不需要考慮這個狀況。

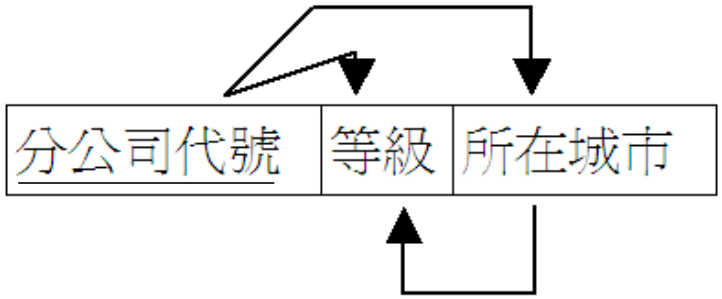
- **範例 (滿足1NF, 2NF, 3NF，但不滿足BCNF的樣式)：**

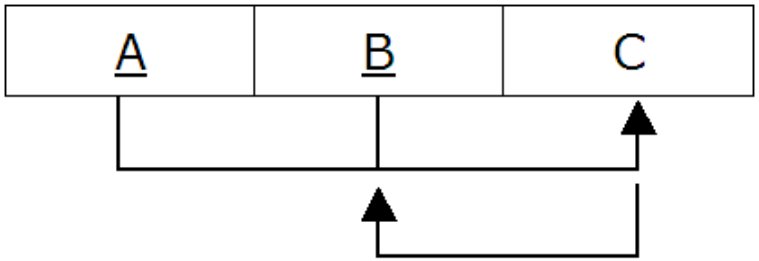


  - **滿足3NF**
    - **(∵B是主鍵的一部份) 沒有 "非鍵值屬性遞移相依於主鍵" 的問題**
  - **不滿足BCNF**
    - **(∵有一個決定因素C不為候選鍵) 候選鍵需要能決定其它所有屬性**

- **決定因素非候選鍵時，可能**
  - ① **有非鍵值屬性之遞移相依性 (非3NF，非BCNF)**
  - ② **沒有非鍵值屬性之遞移相依性 (符合3NF，非BCNF)**

∴BCNF比3NF更嚴謹!!





Client	Problem	Consultant
宋小于	財務	李大輝
宋小于	人事	黃小州
陳圓圓	財務	劉太音
…	…	…
呂小蓮	金融	陳火扁
張大宏	人事	黃小州

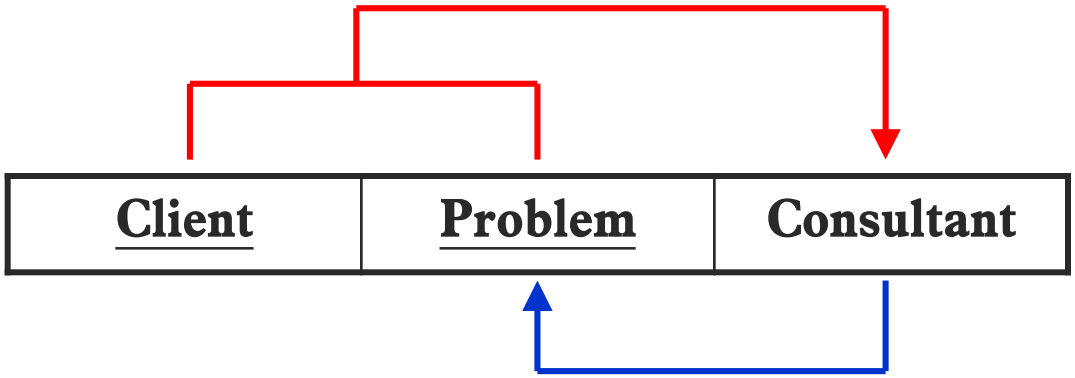
顧問表格 Advisor

- **上述表格中的每一個屬性，皆無法獨自成為主鍵**
- **某一個客戶就某一個問題，只能接受一位顧問的輔導。**
- **每一個顧問只擅長某類型問題，然而擅長某類問題的顧問可能不只一位。**

- **該表格有以下關係**

  {Client, Problem}→Consultant

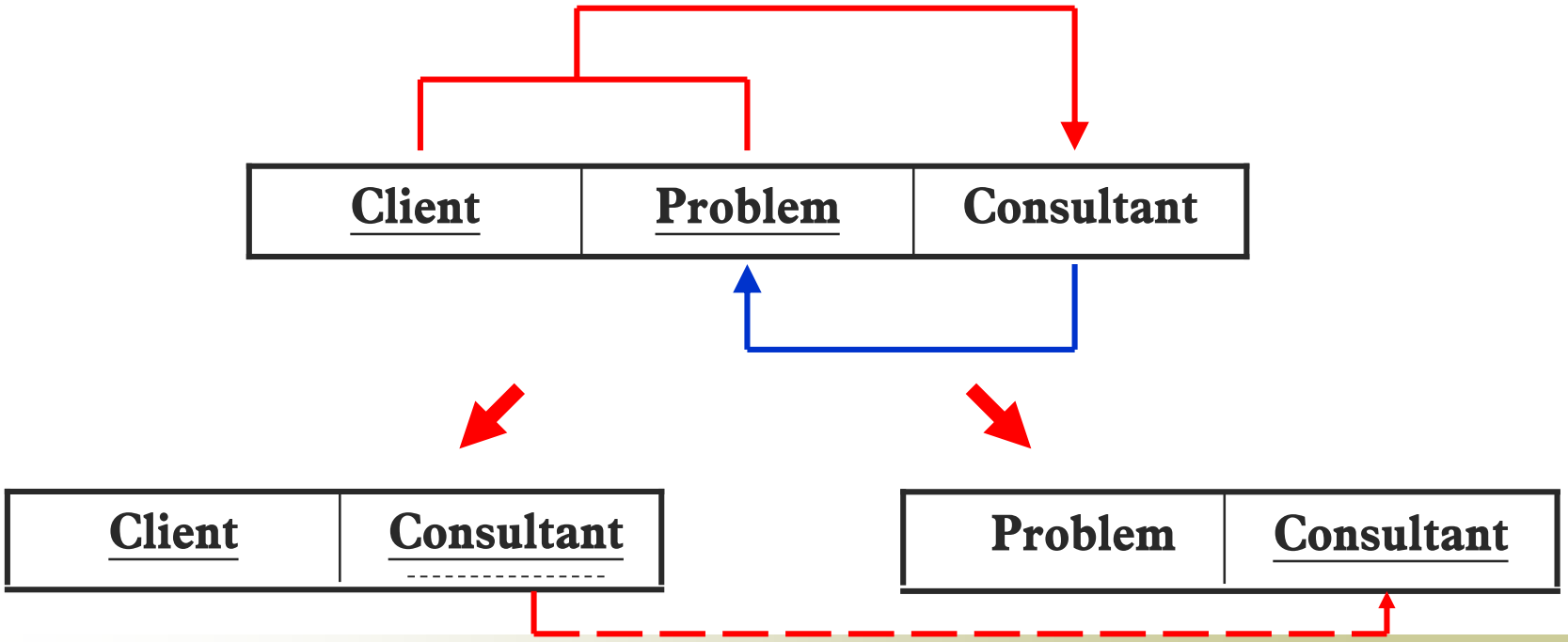
  Consultant→Problem



- **檢視此關聯：**
  - 沒有存在非鍵值屬性**部份功能相依**於主鍵，所以為2NF的關聯
  - 非鍵值屬性間不存在**遞移相依性**，所以為3NF的關聯

- 但是，上例3NF表格可能仍存在一些異常：
  - 輸入時必須等主鍵完整輸入才可進行。
    - 新增(Insert)一名新顧問為吳小雄，專長為金融。但由於他目前沒有客戶，所以Client欄為NULL。然而，因為Client是主鍵的一部份，不允許為NULL，因此違反了個體完整性限制 (Entity Integrity Constraint)。
  - 刪除時會把過多資訊刪掉。
    - 因為呂小蓮撤回委託的業務，因此必須刪除(Delete)這筆記錄。然而，如果陳火扁只有呂小蓮這一個客戶，會使得陳火扁的資料一併被刪除，造成資料的遺失。
  - 修改時需要一起修改許多相關值組。
    - 如果黃小州顧問覺得自已的名字與八字不合，因此更改(Update)名字成黃小洲。然而在表格中他有多筆記錄存在，也必須一併更改，若不小心只更改了部份資料，便會造成資料不一致的情況。

- 作法：
  - **拆解關聯**：將不為候選鍵的決定因素與其相依因素取出，建立新的關聯，並將相依因素自原關聯中刪除。
  - **決定外來鍵**



無損分解與有損分解

- 假如舊關聯表中所有的功能相依性都可以由分解後所產生的新關聯中不多不少地保留著(即：可由阿姆斯壯定理推回)，這種分解稱之為無損分解(Lossless Decomposition)，反之稱為有損分解。
- 無損分解是我們所期望的。但若得到的結果是有損分解，則須退回到前一步驟的結果 (即：反正規化)。
  - 如：顧問表格的例子，我們拆解成兩個關聯後，雖然滿足了BCNF的要求，但是我們發現原先存在的功能相依性{Client, Problem}→Consultant 永遠消失了。
  - 此例由BCNF所分解出來的結果為有損分解，而前面3NF所分解出來的結果為無損分解，故此例做到3NF即可，雖然在使用資料時可能會造成一些異常現象，但可選用其它方式降低這些異常現象的發生。

分解表格為何要著重於“無損分解” ？

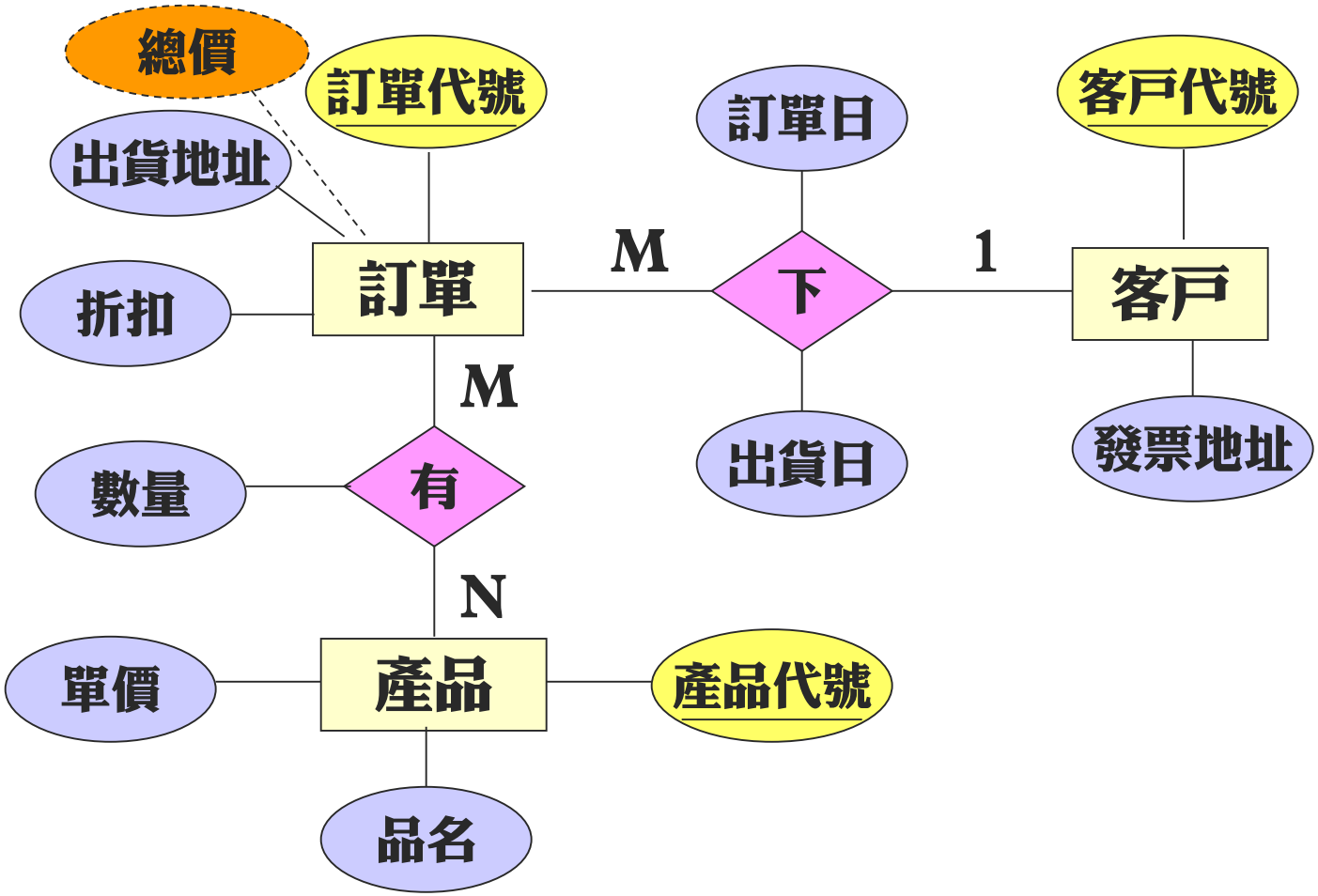
- 表格中的每一個功能相依關係，皆表示著**各個欄位之間的關係**，而這些關係是會發生在企業日常的資料運作當中。
- 但是，當有多個功能相依發生在同一個二維表格上，往往會引發一些操作上的異常情況，如：
  - 因**部份功能相依**所引發的異常
  - 因**遞移相依**所引發的異常
  - 因**多值相依**所引發的異常
  - …
- 因此，為了解決這些異常情況，利用正規化**將表格適度分解**，讓會引發上述異常之數個功能相依不要出現在同一個二維表格上。但是，這些表示著欄位間關係的功能相依性我們仍希望能保留著。

反正規化（De-normalization）

- 有時，在詳細的正規化之後，反而會造成資料處理速度上的困擾，或是其它因為表格分割而造成的損失。
- 因此在執行效率考量之下，有時候不得不做適當的反向正規化。但必須小心控制反正規化所造成的資料重覆性問題或一些更新上的異常，以期達成最佳的資料庫設計

■ 如何由E-R Model從事正規化

- E-R Model→Relation →Normalization

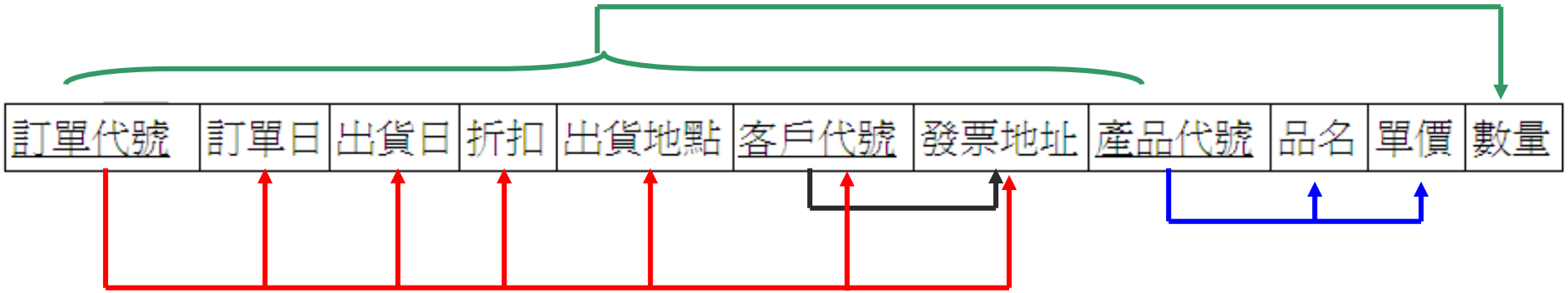


關係上的屬性可由誰決定？

1. 多對多關係：由兩個個體之主鍵共同決定
2. 1 對多關係：由多端個體之主鍵所決定（同時可決定1端的所有屬性）
3. 1 對 1 關係：
   (1) 一端完全參與、一端部份參與：由完全參與之個體主鍵決定
   （同時可決定部份參與端個體之主鍵）
   (2) 兩端皆部份參與：由實例數較少之個體主鍵決定
   （同時可決定實例數較多之個體的主鍵）
   (3) 兩端皆完全參與：由兩個個體之主鍵共同決定

任挑其中一個個體，以該個體之主鍵決定。

E-R Model→Relation



Relation → Normalization

- 2NF →去除部份相依性
  - 訂單表：訂單代號、訂單日、出貨日、折扣、出貨地點、客戶代號、發票地址
  - 產品表：產品代號、品名、單價
  - 產品訂單表：產品代號、訂單代號、數量
- 3NF →去除遞移相依性
  - 訂單表：訂單代號、訂單日、出貨日、折扣、出貨地點、客戶代號
  - 客戶表：客戶代號、發票地址

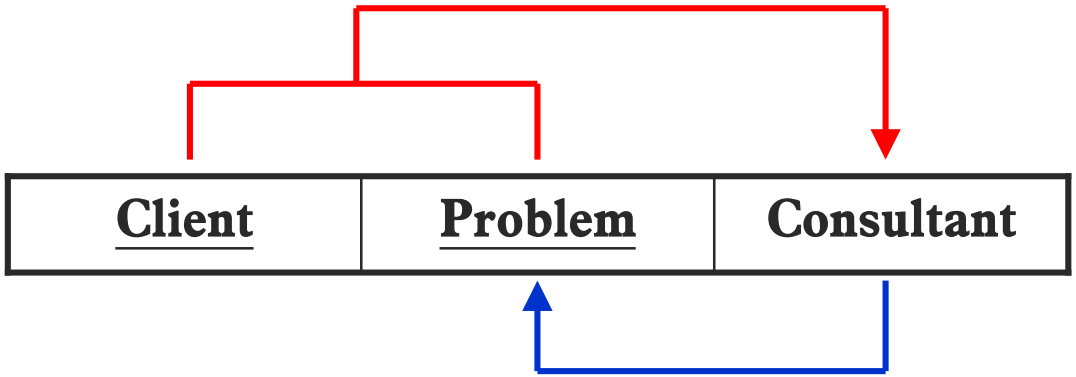
總結

- 資料庫正規化可降低資料重複性 (Data Redundancy)。
- 正規化亦可消除某些在資料插入、刪除或更新時所衍生的異常(Anomalies)問題。

補　充

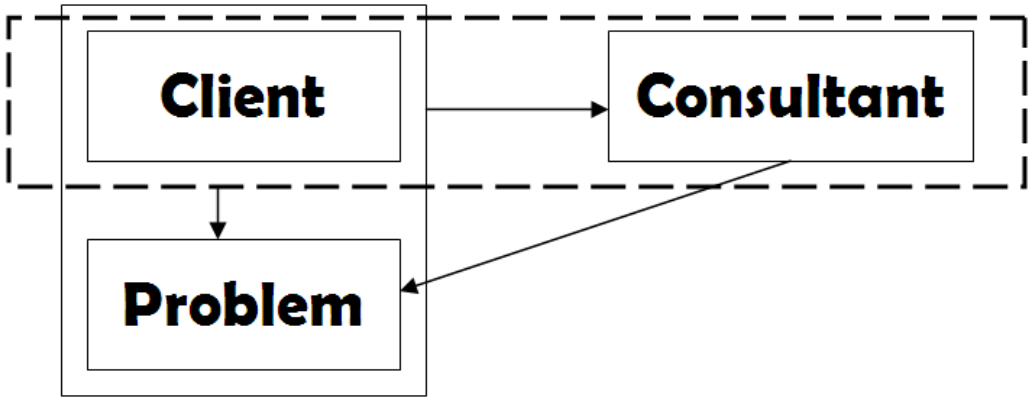
■ 深入討論BCNF

- 承繼先前“顧問表格” 的範例：



- 已知此關聯：
  - 沒有存在非鍵值屬性**部份功能相依**於主鍵，所以為2NF的關聯
  - 非鍵值屬性間不存在**遞移相依性**，所以為3NF的關聯

- 然而，上述的關係是否還有隱含其它的關係??
  - 有， {Client, Consultant} → Problem

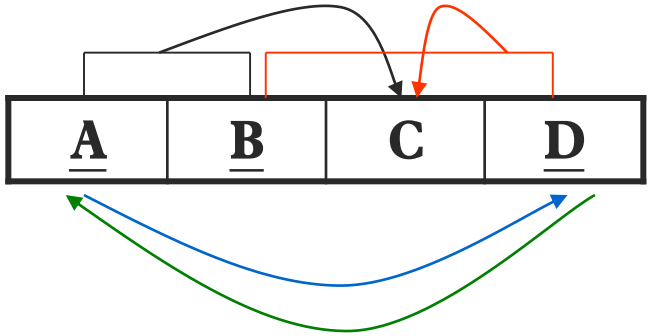


  - 如何得知？
    - 利用反身性，則： {Client, Consultant} → Consultant
    - 已知Consultant → Problem
    - 利用遞移性，則： {Client, Consultant} → Consultant 與 Consultant → Problem 可推得{Client, Consultant} → Problem

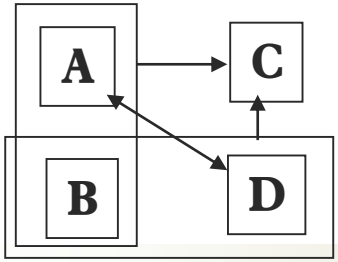
- 由BCNF的基本定義，再加上所推導出的隱含關係後，我們可以得知當有下述情況時，就必須採行BCNF正規化：
  - 關聯R有**兩個(以上) 候選鍵**，且這些候選鍵是由**複合屬性**所構成
  - 這些候選鍵的**某些屬性**是**共有**的 (重疊；Overlapped)

範例

- 假設關聯R(A, B, C, D)的功能相依性如下：



- 檢視此關聯：
  - 沒有存在非鍵值屬性**部份功能相依**於主鍵，所以為2NF的關聯
  - 非鍵值屬性間不存在**遞移相依性**，所以為3NF的關聯
  - 有多個複合屬性的候選鍵，且為**重疊**，所以不為BCNF的關聯

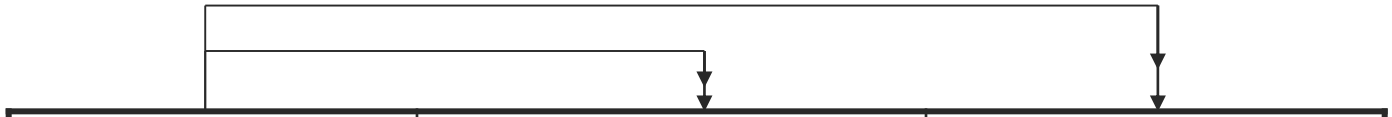


- 解1：
  - 可分解成 R1(A, D)和R2(A, B, C)
    - R1的功能相依性有：A→D與D→A
    - R2的功能相依性有：AB →C
  - 此為無損分解 (∵ D→A 且AB →C ，∴BD →C)
- 解2：
  - 可分解成 R1(A, D)和R2(B, C，D)
    - R1的功能相依性有：A→D與D→A
    - R2的功能相依性有：BD →C
  - 此為無損分解 (∵ A→D 且BD →C ，∴BD →C)

■ 多值相依性與第四正規化 (4NF)

- 第四階正規化型式的基礎是「多值相依」（Multi-valued Dependency, MVD），這是在1977年所提出，屬於功能相依的廣義解釋。或者，功能相依是多值相依的一種特例。
- 多值相依性 (Multi-valued Dependency, MVD)：
  - 假設關聯R(A, B, C)中，每個A值可以對應到多個B值，但與C完全無關，則稱A→→B (A多值決定B)。
  - 或每個A值可以對應到多個C值，但與B完全無關，則稱A→→C (A多值決定C)
- 多值相依的先決條件是關聯擁有3個(以上)的屬性。

- 例如：一門課可能由多位老師授課，每一門課可以使用多本教課書，儲存這些資料的Course_Instructor_Textbook關聯表，簡稱為CIT，如下圖所示(在此的相依不是指功能相依)：



Course	Instructor	Textbook
物件導向程式設計	溫老師 陳老師	Java2程式設計教學 物件導向程式設計
程式語言	溫老師	C++程式設計教學 Java2程式設計教學 VB.net程式設計教學

- 上述CIT關聯可透過對應關係改寫：

Course	Instructor	Textbook
物件導向程式設計	溫老師 陳老師	Java2程式設計教學 物件導向程式設計
程式語言	溫老師	C++程式設計教學 Java2程式設計教學 VB.net程式設計教學

Course	Instructor	Textbook
物件導向程式設計	溫老師	Java2程式設計教學
物件導向程式設計	溫老師	物件導向程式設計
物件導向程式設計	陳老師	Java2程式設計教學
物件導向程式設計	陳老師	物件導向程式設計
程式語言	溫老師	C++程式設計教學
程式語言	溫老師	Java2程式設計教學
程式語言	溫老師	VB.net程式設計教學

多值相依所產生的異常 (Anomalies)

- **新增記錄時**
  - 由前面的對應關係，可得知若新增老師 “陳老師” 教授 “程式語言”，雖然只有一門課，但是需要同時新增3筆值組，每一個值組對應到一本教課書，否則就會產生資料不一致的問題。

Course	Ins
物件導向程式設計	溫老 陳老
程式語言	溫老 陳老

Course	Instructor	Textbook
物件導向程式設計	溫老師	Java2程式設計教學
物件導向程式設計	溫老師	物件導向程式設計
物件導向程式設計	陳老師	Java2程式設計教學
物件導向程式設計	陳老師	物件導向程式設計
程式語言	溫老師	C++程式設計教學
程式語言	溫老師	Java2程式設計教學
程式語言	溫老師	VB.net程式設計教學
程式語言	陳老師	C++程式設計教學
程式語言	陳老師	Java2程式設計教學
程式語言	陳老師	VB.net程式設計教學

第四階正規化型式

- 作法：
  - 將每個MVD拆解成新的關聯 (理由：這種一對多的多值關係，永遠也不知道這個 “多” 到底是多少。留多了不好，留少了不夠。乾脆獨立出去，要多少任君選擇。)

Course	Instructor	Textbook
物件導向程式設計	溫老師	Java2程式設計教學
物件導向程式設計	溫老師	物件導向程式設計
物件導向程式設計	陳老師	Java2程式設計教學
物件導向程式設計	陳老師	物件導向程式設計
程式語言	溫老師	C++程式設計教學
程式語言	溫老師	Java2程式設計教學
程式語言	溫老師	VB.net程式設計教學

Course	Instructor
物件導向程式設計	溫老師
物件導向程式設計	陳老師
程式語言	溫老師

Course	Textbook
物件導向程式設計	Java2程式設計教學
物件導向程式設計	物件導向程式設計
程式語言	C++程式設計教學
程式語言	Java2程式設計教學
程式語言	VB.net程式設計教學

■ 合併相依性與第五正規化 (5NF)

- 第五階正規化型式的基礎是「**合併相依**」(Join Dependency, JD)。
  - 合併相依是指當關聯表分割成**3個或更多關聯**後，一定能夠透過**多次合併運算**恢復成**原來的關聯表**。
- 例如：每個科系（department）開多門課，課程（course）可以給多位學生修，學生（student）可以修不同科系的課，這3個屬性循環關聯儲存在同一個關聯表DCS，如右圖所示：

DCS

department	course	student
資訊系	101	陳會安
資訊系	202	江小魚
資管系	202	江小魚
資管系	101	張三丰
資管系	222	江小魚

- 將上例的關聯表使用投影運算分割成3個關聯表：DC (department, course)、CS (course, student)、SD (student, department)

DC

department	course
資訊系	101
資訊系	202
資管系	202
資管系	101
資管系	222

CS

course	student
101	陳會安
202	江小魚
101	張三丰
222	江小魚

SD

student	department
陳會安	資訊系
江小魚	資訊系
江小魚	資管系
張三丰	資管系

- 經過多次合併運算，我們會發現這三個關聯會恢復成原來的關聯。因此，原範例的關聯表具有合併相依性。

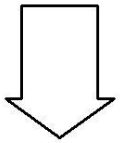
DC

department	course
資訊系	101
資訊系	202
資管系	202
資管系	101
資管系	222

⋈

CS

course	student
101	陳會安
202	江小魚
101	張三丰
222	江小魚



DCS

department	course	student
資訊系	101	陳會安
資訊系	101	張三丰
資訊系	202	江小魚
資管系	202	江小魚
資管系	101	陳會安
資管系	101	張三丰
資管系	222	江小魚

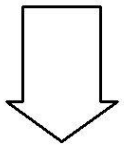
DCS_Temp

department	course	student
資訊系	101	陳會安
資訊系	101	張三丰
資訊系	202	江小魚
資管系	202	江小魚
資管系	101	陳會安
資管系	101	張三丰
資管系	222	江小魚

▷◁

SD

student	department
陳會安	資訊系
江小魚	資訊系
江小魚	資管系
張三丰	資管系



DCS

department	course	student
資訊系	101	陳會安
資訊系	202	江小魚
資管系	202	江小魚
資管系	101	張三丰
資管系	222	江小魚

合併相依所產生的異常 (Anomalies)

- 如果一個關聯擁有合併相依，在新增和刪除時就會產生異常情況。
- 例如：DCS關聯擁有合併相依。假設在DCS關聯中，新增一筆值組{資管系, 101, 江小魚} 如下圖所示：

DCS

department	course	student
資訊系	101	陳會安
資訊系	202	江小魚

DCS

department	course	student
資訊系	101	陳會安
資訊系	202	江小魚
資管系	101	江小魚

- 將上述關聯表以投影運算分割成DC、CS和SD三個關聯表，再將它使用合併運算結合起來，可以發現多了一筆值組，如下圖所示：

DCS

department	course	student
資訊系	101	陳會安
資訊系	101	江小魚
資訊系	202	江小魚
資管系	101	江小魚

- 如果新增一筆值組{資訊系, 101, 江小魚}，在分割和合併後，就不會多出一筆值組。

第五階正規化型式

- DCS關聯表擁有合併相依，並不符合5NF，所以需要執行第五階正規化。
- 使用投影運算分割成3個關聯表，如下圖所示：

DC

department	course
資訊系	101
資訊系	202
資管系	202
資管系	101
資管系	222

CS

course	student
101	陳會安
202	江小魚
101	張三丰
222	江小魚

SD

student	department
陳會安	資訊系
江小魚	資訊系
江小魚	資管系
張三丰	資管系

- 實務上，由於關聯上通常有大量的屬性，從中找出JD十分困難，因此目前實際資料庫的設計甚少提及。