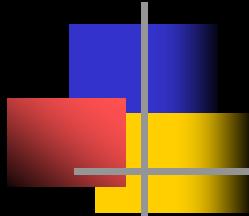




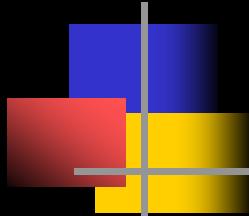
Golden Gate Bridge, USA by Frank S.C. Tseng
(<http://www2.nkfust.edu.tw/~imfrank>)

第十五章 商業智慧與資料倉儲



本章內容

- 15.1 商業智慧的興起
- 15.2 為什麼需要資料倉儲？
- 15.3 資料倉儲系統架構與元件
- 15.4 資料倉儲的設計—維度模式
- 15.5 資料倉儲與資料庫的差別
- 15.6 使用SQL Server 2008 建立資料倉儲
- 15.7 使用MDX 摳取 Cube 的資料
- 15.8 使用Excel 製作 Cube 的樞紐分析表
- 15.9 結論



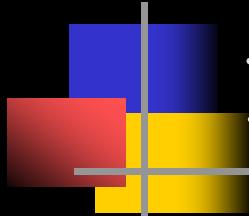
商業智慧(BI)的興起

"How you gather, manage, and use information will determine whether you win or lose."

-- Bill Gates, Microsoft

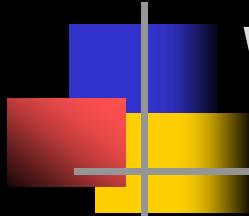
*"Business @ The Speed of Thought:
Using a Digital Nervous System"*

- 人類感受天氣冷⇒馬上知道要多加衣服；
- 企業感受公司營運困境或景氣變冷時⇒也應立刻知道如何進行防護與避險
- 商業智慧讓企業「立刻」感受到景氣的變化
- 以快如人類思考的速度來進行調適。



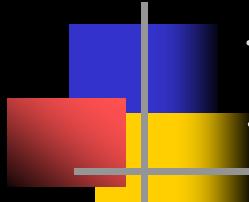
企業 e 化後仍面臨問題

- 系統各自獨立，無法提供全面性的整合分析資訊；
- 營運資料與日俱增，傳統 SQL 查詢效能低落；
- 管理階層無法透過多重角度的分析報表進行決策
- 資訊管理部門每天都有產生不完的報表；
- 報表之間的關聯性無法自動進行相互連結 (類似網頁的超連結)，需要承辦人員親自參與說明，
- 員工的日常業務一再重覆，自動化效果大打折扣。



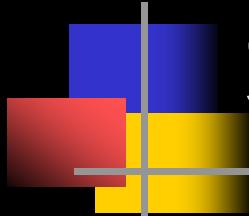
What is Business Intelligence?

- 「商業智慧」(Business Intelligence, BI) 是一種思考模式 (A Way of Thinking)，
 - 企業藉由經營過程收集資料，
 - 加上匯入外部環境的資料，進行整合、精煉，
 - 並透過制定各類「關鍵績效指標」(KPI, Key Performance Indicator) 的量化數據加以分析後，
 - 即時產生各類分析報表，適時警示管理階層進行未來績效預測與後續因應策略的方案制定。



如何達成商業智慧的目標

- 首先要將下列三者進行串聯才能實現：
 - 整合異質性資料軟體開發工具 (Integration Service)：讓各種資料格式被快速過濾，透過介面進行互相轉換、整合，產生整合性資料庫。
 - 因應快速分析需求的環境建置 (Analysis Service)：Data Warehouse (彈性儲存結構)+ On-Line Analytical Processing (線上分析處理) 模式 + Data Mining (歸納與預測未來趨勢)。
 - 產生報表的工具軟體 (Reporting Service)：擷取分析結果快速產生各類商業智慧報表，預先連結



SQL Server 2008 Solution

使用 SQL Server
Integration Service
進行資料整合

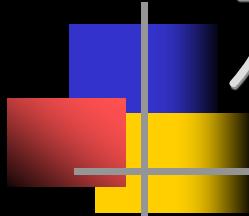
使用 SQL Server
Analysis Service
進行資料分析

使用 SQL Server
Reporting Service
進行報表呈現

- ◆ 整合異質性資料來源
- ◆ 資料過濾與精煉
- ◆ 資料轉換與清理

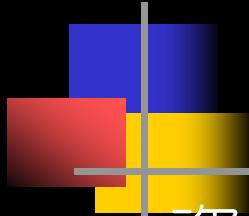
- ◆ 從商業邏輯角度分析
- ◆ 以資料倉儲進行觀察
- ◆ 以資料採礦進行預測

- ◆ 發佈分析結果
- ◆ 各種角度的報表呈現
- ◆ 使用者自訂報表格式



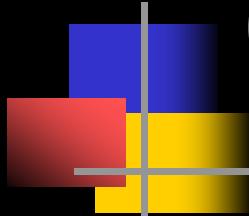
為什麼需要資料倉儲？

- OLTP 系統—資料庫正規化設計將資料的重覆性 (Redundancy) 降低，消除資料更新時的異常現象 (Update Anomalies)，提高運作效率。
- 目標：建立高效率的 Operational Database，很容易建立各式各樣的資料，以追蹤客戶、銷售、庫存等商業或行政管理上的狀態。
- 傳統上，企業所賴以成功的關鍵：妥善運用這類高效率的操作型資料庫。



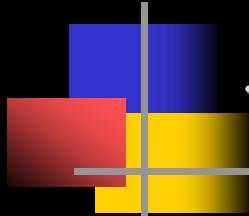
OLAP 系統

- 資料庫存放資料的最終目的在於希望用現有及歷史資料去產生一些有用的資訊，以作為協助我們制定未來決策方向的參考。
- 80年代「報表系統」(Reporting Systems)→「試算表」(Spreadsheet)→SAS、SPSS 統計套裝軟體→「決策支援系統」(DSS)
- 透過電腦化介面分析、了解過去，預測未來，以提供制定各種類型決策的參考依據。



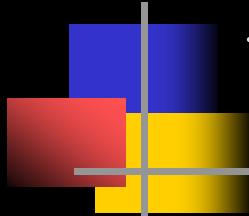
OLTP 與 OLAP 間的兩難

- OLTP Systems 上所組織出來的資料結構，重點在於將資料的儲存方式最佳化 (關聯表的正規化)，適合大量的異動處理，
- 無法滿足我們很方便地擷取各式各樣資訊的需求
- 例如：只為了要調閱某筆客戶與訂購的資料，需要將三、四個關聯表一起合併才能如願。



以 OLTP 架構做 OLAP ?

- 「本公司有堆積如山的資料，但我們不知道如何取得與應用它們。」
- 「我每天都有從各個構面（從時間、物品類別等構面）去擷取資料的需求，但是現有的資料庫結構讓我不曉得要如何下這些查詢。」
- 「我想同時比較去年度與今年度，在台灣北、中、南、東部的銷售情況在男、女方面的差異與對比情形，但是卻要花很多時間才能將這些統計資料組織起來。」



可以看成 N 個一維欄位的關聯表

購買日期	品名	客戶身份	價錢
1999/5/8	毛巾	外國男性	500
1999/5/9	香皂	本國女性	800
1999/6/1	保養品	本國女性	1200
1999/6/8	健康食品	本國男性	1500
1999/7/8	毛巾	外國女性	350
1999/8/8	健康食品	外國男性	1200
1999/9/10	香皂	外國女性	500
1999/10/21	健康食品	本國女性	1800

很難看出銷售狀況的各類分析結果

資料以多維度呈現

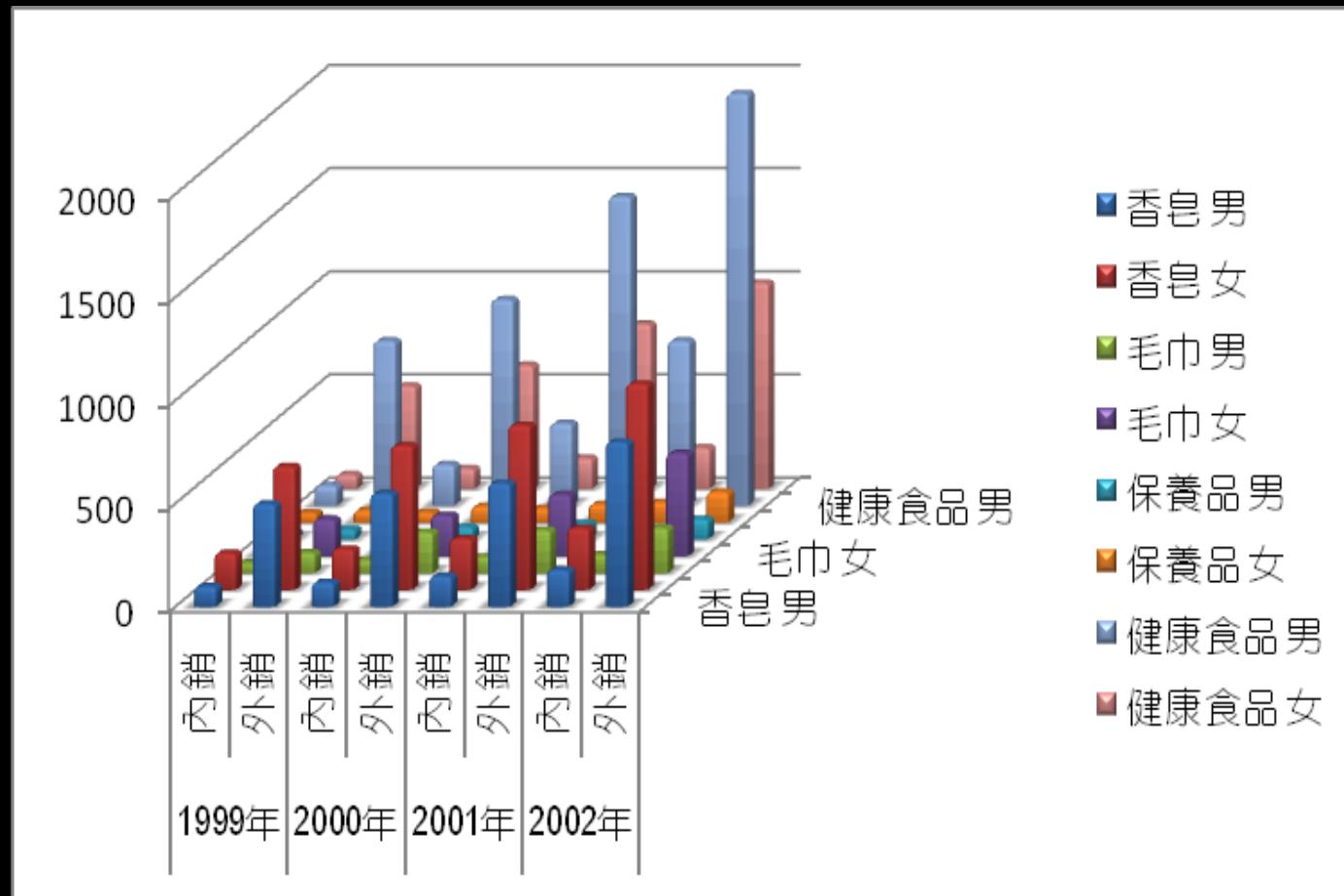
年度/銷別		1999年		2000年		2001年		2002年	
		內銷	外銷	內銷	外銷	內銷	外銷	內銷	外銷
產品/性別	男	100	500	120	550	150	600	180	800
	女	180	600	200	700	250	800	300	1000
毛巾	男	50	100	65	200	80	210	90	220
	女	80	180	90	200	100	300	150	500
保養品	男	38	50	45	60	50	80	70	100
	女	48	60	50	80	70	90	100	150
健康食品	男	100	800	200	1000	400	1500	800	2000
	女	70	500	100	600	150	800	200	1000

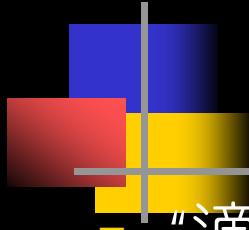
↑
維度 1
↑
維度 2

以 4 個維度看第 5 個維度 (銷售量)

← 維度 3
← 維度 4

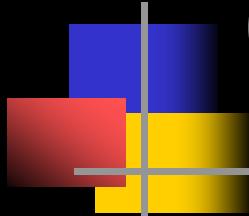
多維度配合統計圖表一目了然





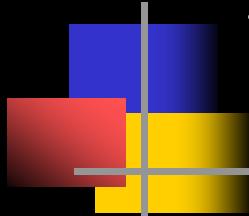
Data Warehouse 提供 OLAP

- “適合高效率線上處理的資料結構並不見得適合做大量資料分析與決策”。
- 資料庫中所放的內容就像倉庫中放置許多原料一樣，要做成成品要自行用 SQL 求得，其中的原料可能含有不良或不全的原料，需要進一步過濾。
- 「資料倉儲」(Data Warehouse) 概念將我們想分析的各種資料切面 (Slice) 做前置處理後，變成能很方便地產生各類資訊的型式，然後擺放在系統中供未來下決策時直接取用，就像倉儲中放置各類成品一樣，可以讓我們隨時取用。
- On-Line Analytical Processing Systems, OLAP Systems)



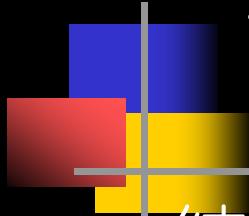
OLAP vs. 試算表軟體

- 資料數量很少時，以試算表軟體，如：Excel 或 Lotus 1-2-3 的「樞紐分析表」功能就可以達成目的
- 但是如果要分析的對象是整個企業的各種資料時，由於數量很大，而且效能以及各種整合性的要求高，所以就非建置「資料倉儲」系統不可了。
- 可以將兩者互相搭配，做最有效的利用。



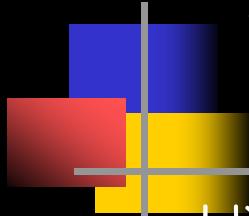
資料倉儲 VS. 資料採礦

- 「資料採礦」(Data Mining/Data Dredging)
- 「知識發掘」(Knowledge Discovery in Databases—KDD) 範圍比 Data Mining 還廣
- 「資料考古學」(Data Archaeology)、
- 「資料樣型分析」(Data Pattern Analysis) 或
- 「功能相依性分析」(Functional Dependency Analysis)
- 需要架在資料倉儲：Clean、Complete and Integrated



資料採礦技術

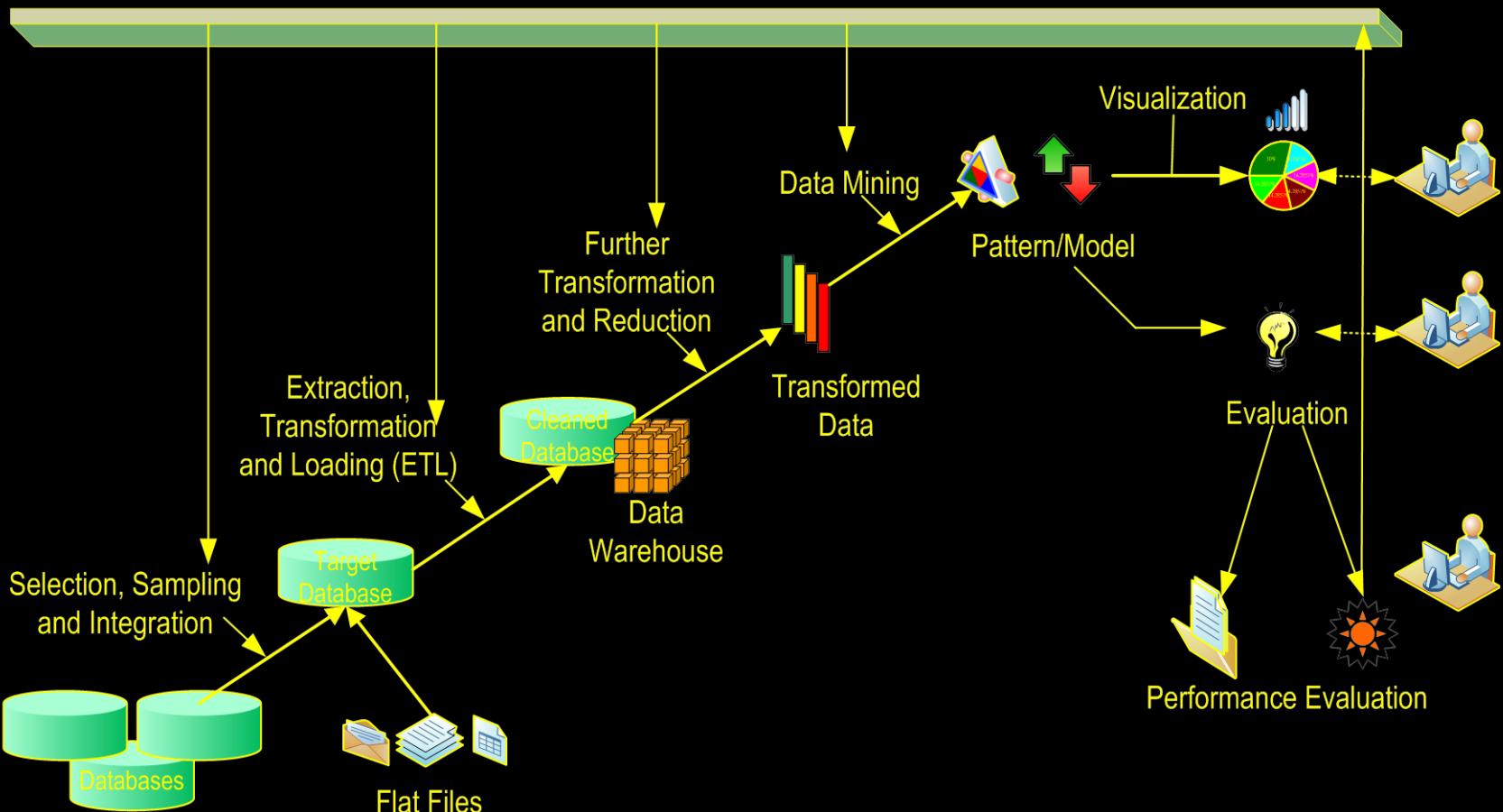
- 結合資料庫系統、機器學習、人工智慧、統計學、高效能運算架構、視覺化等技術相結合的一個重要研究領域
- 是一項增加各企業潛能的重要技術
- 因為：企業體經常會大量地收集市場、客戶、供應商、競爭對手以及未來趨勢等重要資訊
- 可從龐大的資料庫中，發掘出非常重要而且有用的資訊或知識，輔助我們做出重要的決策。

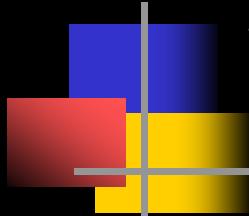


資料採礦技術類型

- 找出順序規則 (Sequential Rules)
- 找出關聯性規則 (Association Rules)
- 找出同質時間序列 (Similar Time Series)
- 找出分類的規則 (Classification Rules)：將以往的資料依照所找出的規則建成決策樹
- 找出群集規則 (Clustering Rules)：群間差異大、群內差異小。E.g. 將客戶分成數群

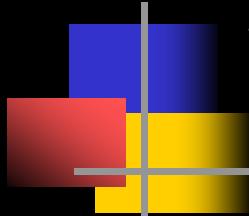
KDD 的步驟 [U.M. Fayyad (1996)]





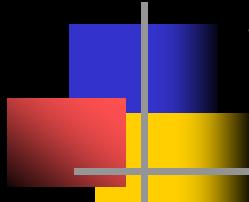
資料採礦的步驟（續）

- 最早由 U.M. Fayyad 在 (1996) *IEEE Expert Intelligent Systems & Their Applications* 中描繪出大綱
- 後續有些學者則認為在做資料採礦時應該先整合各種資料並架在資料倉儲上來實行
- 其步驟說明如上頁所示。



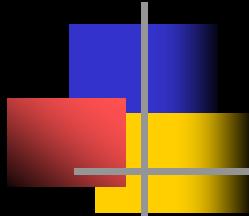
資料採礦的步驟（續）

- 針對應用領域，瞭解該領域的先導知識，及最終目標。
- 先將資料庫與檔案資料整合，並透過資料篩選、取樣以建立一個採擷目標資料集。
- 將資料做前置處理，去除錯誤或不一致的資料 (Data Cleaning and Preprocessing)。
- 此步驟有時候可以與上一個步驟互換順序。



資料採礦的步驟（續）

- 針對所要開發的應用領域，瞭解該應用領域、熟悉相關的先導知識，以及最終目標
- 資料篩選、取樣以建立一個採擷目標資料集
- 將資料做前置處理，去除錯誤/不一致的資料
- 對資料做簡化或轉換的工作
- 決定資料採礦的類型（關聯規則/群集規則...）
- 選擇演算法或處理模式
- 以上述演算法尋找「樣型」(Pattern)、做迴歸分析(Regression)或找出分類型態等
- 評估結果（需專家參與），以決定該結果是否有用。
- 這些步驟已經有標準化的規範了：稱為 CRISP-DM
(<http://www.crisp-dm.org>) 是由 NCR, SPSS, 等廠商所推動

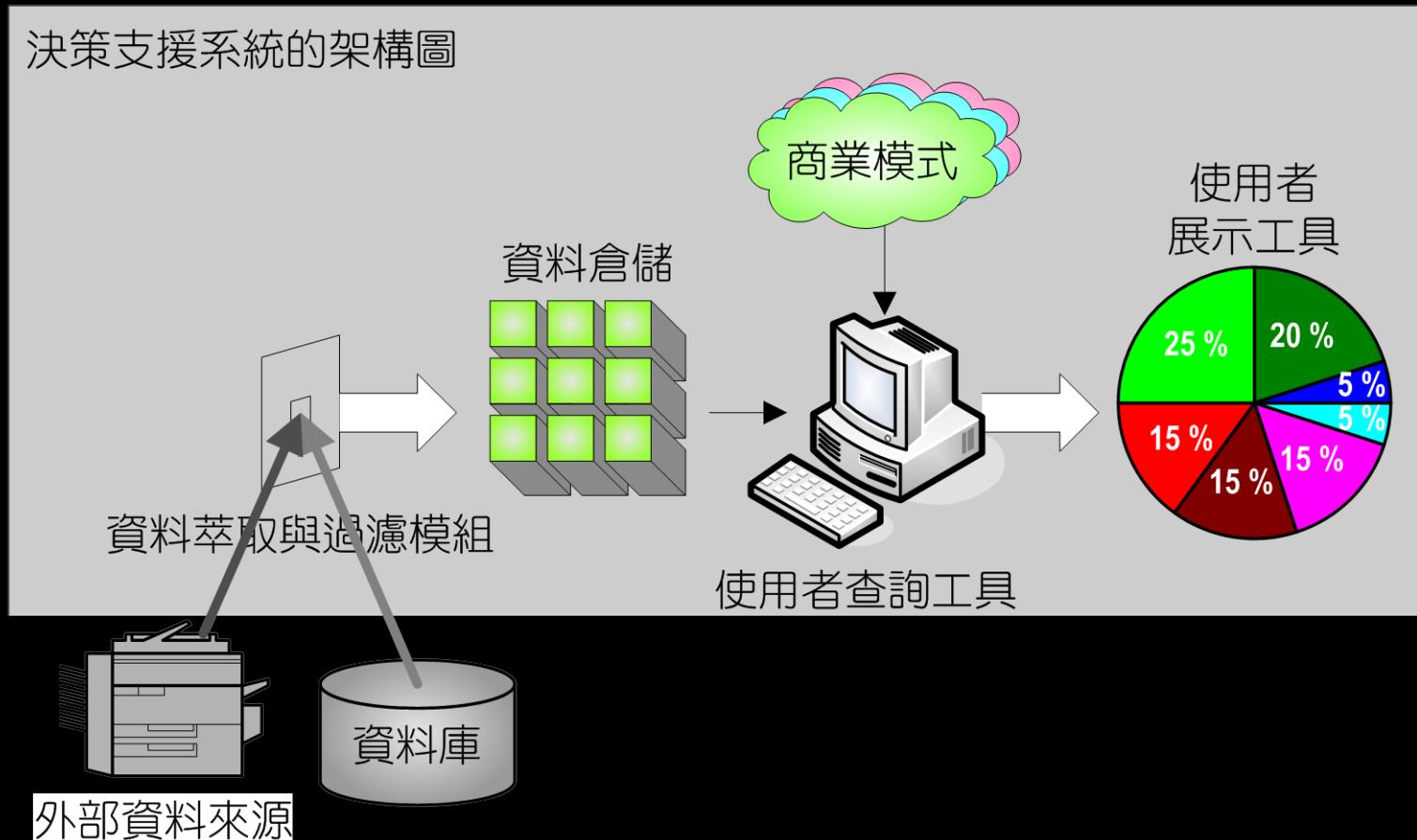


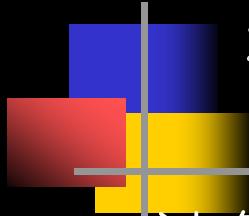
應用層面

- 超市客戶分析 (動線設計、貨架、音樂, ...)
- 美國 NBA、職棒對手與戰況分析
- 股市、期貨分析
- 痘情上的醫療分析
- 科學與工程上的分析
- 保險與信用卡市場，洗錢與逃、漏稅防制
- 電信資料的商機開發 (e.g. 與航空公司結盟)

資料倉儲 vs. 決策支援系統

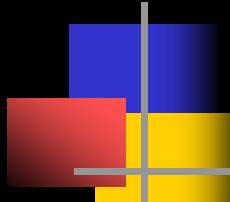
決策支援系統的架構圖





資料倉儲 vs. 決策支援系統

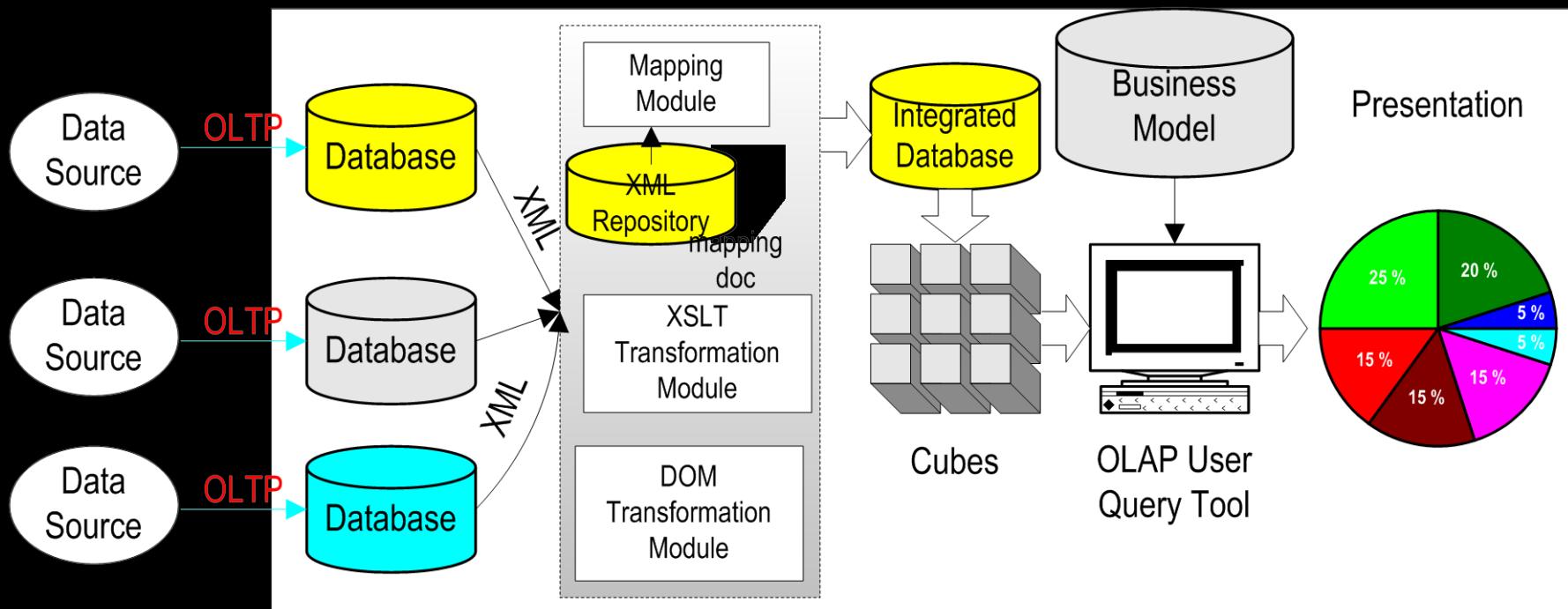
- 決策支援的有效與否？取決於「資料品質」(Quality of Data)的好壞。
- 因此電腦系統中有所謂的「垃圾進，垃圾出」(Garbage In, Garbage Out) 的說法。
- 資料儲存體 (Data Store Component)—存放：
 - 商業性資料 (Business Data)—包括由運算資料所匯總過來的資料，以及來自於其它資料來源的整合資料，如：競爭對手的資料，或政府所提供的產業資料等
 - 商業模式資料 (Business Model Data)—通常是用來描述該商業行為的線性迴歸模式 (Linear Regression Model)、線性規劃模式 (Linear Programming Model)、時間序列分析模式 (Time-Series Analysis Model)、BCG Model、Kano Model 等資料



資料倉儲 vs. 決策支援系統

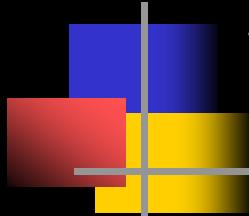
- 資料萃取與過濾模組 (Data Extracting and Filtering Component)—用來萃取、過濾並驗證資料庫的資料，以取得可以用來產生決策的資料。
 - 要能支援批次 (Batch) 或排程 (Scheduled) 的萃取作業，也要能支援異質性資料庫上的綱要整合問題，如下頁所示
 - 這個模組的執行程序簡稱為 ETL (Extraction, Transformation, and Loading)。
 - SQL Server 2008 上已經有類似功能，如：SQL Server Integration Service (SISS) 等 (早期稱為 DTS: Data Transformation Service)。
- 使用者查詢工具 (End-User Query Tool)—協助使用者建立查詢。
- 使用者展示工具 (End-User Presentation Tool)—呈現資料分析結果的型式，如：圓餅圖 (Pie Chart)、長條圖 (Bar Chart)

資料萃取需要異質性資料庫整合



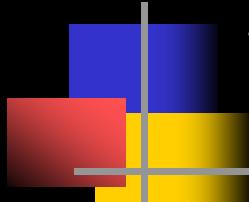
ETL : Extraction, Transformation, and Loading

可透過 XML 做異質性整合



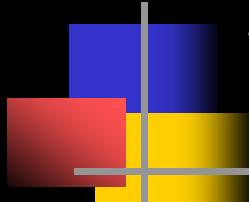
資料倉儲的十二條規則

- 資料倉儲應該與資料庫運作環境分開建置。
- 資料倉儲的資料要完全整合。
- 資料倉儲包含了長時間累積的歷史資料。
- 資料倉儲的資料是某個時間點所擷取到的狀態資料。
- 資料倉儲是以主題為導向 (Subject-Oriented)。
- 資料倉儲中的資料主要是用於讀取，並定時由運作資料庫做批次更新。不允許線上更新。



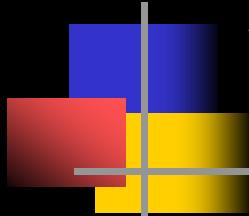
資料倉儲的十二條規則

- 資料倉儲的開發生命週期為資料驅動式 (Data-Driven) 的發展方式，有別於傳統程序驅動式 (Process-Driven) 的發展方式。
- 資料倉儲的資料包含了數個層次的細部資料：目前的細部資料、以往的細部資料、科子目的加總資料，以及全體的加總資料。
- 整個資料倉儲環境具有對非常大量的資料集做「唯讀式異動」 (Read-Only Transactions) 的特性。



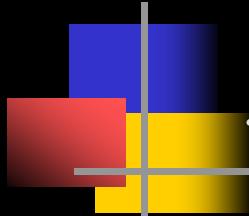
資料倉儲的十二條規則

- 資料倉儲環境具有追蹤資料來源、轉換與儲存的子系統。
- 資料倉儲中的 **Metadata** 是最重要的一部份，它標明並定義所有的資料元素，也提供每一個資料元素的來源、轉換、整合、儲存、與使用方式，以及相互間的關係，產生歷史等。
- 資料倉儲中應該包含一種資源使用的**收費機制**，以便強迫使用者能以最佳的運用方式來使用其中的資料。



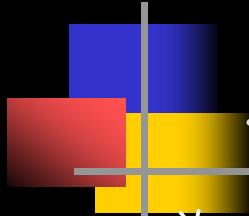
資料市集 (Data Mart)

- 資料倉儲的建立要耗費不少經費、時間、人力以及管理層面所須付出的心力
- 由於資料來源可能跨越部門，所以也必須面臨層層官僚式的溝通與協調才能完成，
- 在建置過程難免會有某些排斥與抗拒的阻力。
- 為了單一部門的決策參考，我們可能會建置一個規模較小的資料倉儲系統，稱為「資料市集」(Data Mart)。



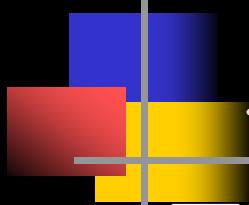
建置資料倉儲系統的好處

- 可以得到相當高的投資報酬率 (ROI, Return on Investment)：根據 [T. Connolly, et al. (2000)] 所指出在 IDC (International Data Corporation) 1996 年的研究顯示：三年平均 ROI 達 400%，90% 的公司超過 40% 的 ROI，一半以上超過 160%，有四分之一超過 600%
- 強化競爭優勢：容易取得以往無法取得的完整資料、或是無法知道的資料、甚至是未曾想過的資料
- 增強決策人員的生產力：準確的決策分析
- 對整體企業運作資料作徹底的整理



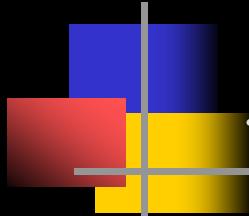
建置資料倉儲系統的好處

- 為企業內部的「知識管理」(Knowledge Management)預作準備：建置過程，讓決策人員全盤考慮其決策過程所需要的資訊，以及彼此間的聯繫關係，可以發現以往的盲點。
- 強化「客戶關係管理」(Customer Relationship Management, CRM)：將客戶的個人資料與消費記錄完整整合，提供更完善的服務。
 - R. Kimball 與 R. Merz 在合著的 The Data Webhouse Toolkit [R. Kimball and R. Merz (2000)] 一書中便倡導將資料倉儲技術與 WWW 技術整合，以便對客戶提供一致化且更完整的 Web 服務品質。



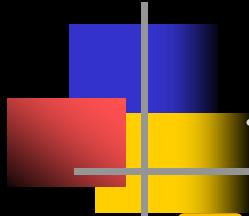
建置資料倉儲所引發的問題

- 不要低估資料整合 (Integration)、轉換 (Transformation) 與載入 (Data Loading) 所需的時間、人力與物力：大約要花費整個專案 80% 的人力與時間。
- 來源系統中可能含有隱藏的潛在問題。
- 可能遺漏所要分析的某些資料：有許多未在舊有系統上出現。
- 可考慮在來源系統上增加功能收集這些資料因應。
- 使用者的需求會急速增加
- 資料名詞與代碼可能沒有統一：異質性資料庫整合的問題有密切的關係。



建置資料倉儲所引發的問題

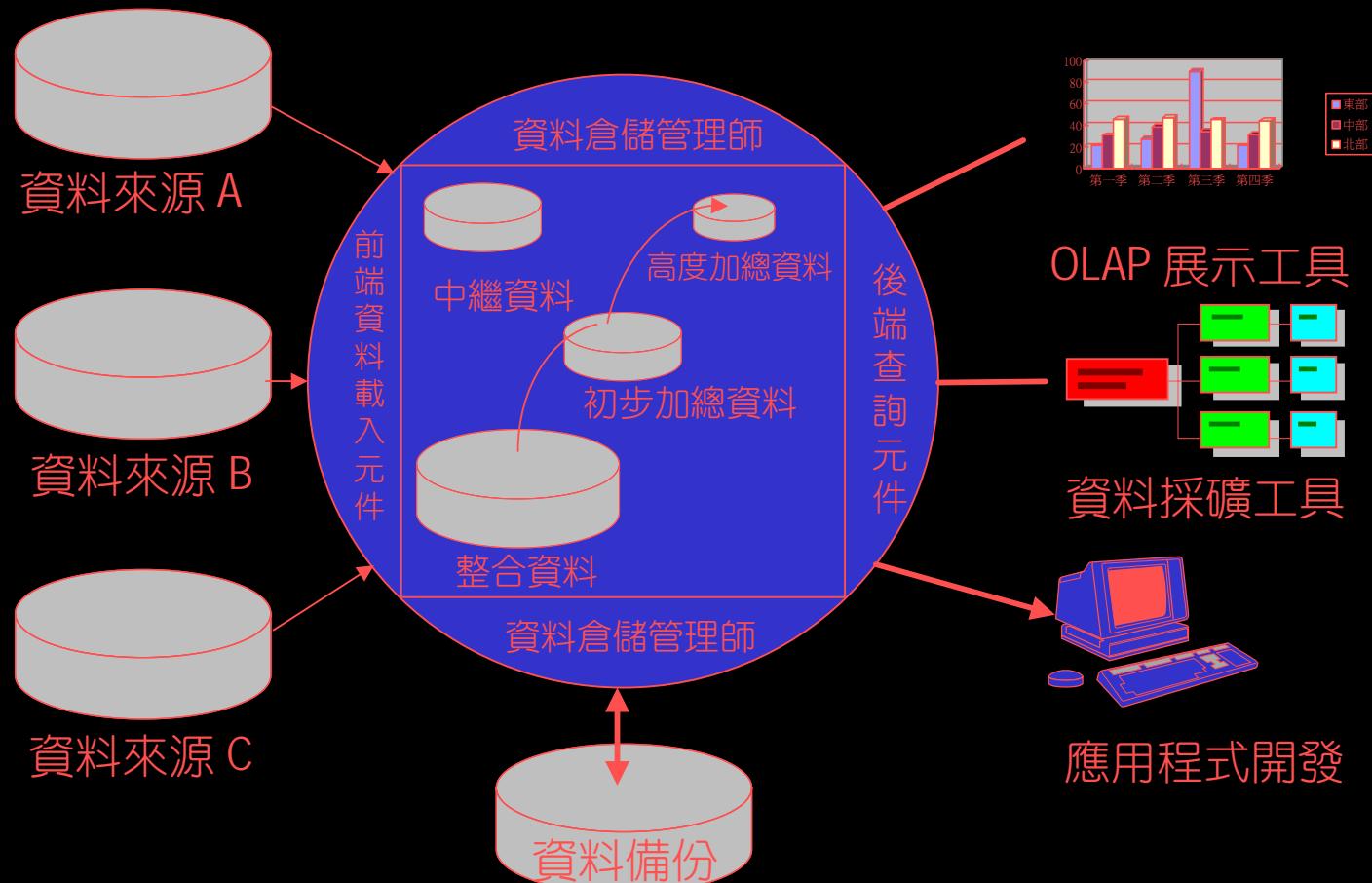
- 需要高速而昂貴的硬體與儲存空間配合：資料儲存空間往往數十甚至數百倍於資料庫所需的空間
- 資料的保密性可能要做某些妥協：多維度的資料呈現方式，使得以往不被允許公開給某些層級使用者的原始或統計資料，在不經意的情況下透露訊息
- 高昂的人員訓練與系統維護費用：一旦企業進行組織重整 (Re-organization) 或程序再造 (Business Process Reengineering) 後，要對資料倉儲做調整
- 專案拖得太長，失去競爭力：耗上兩、三年的時間，失去了先機，導致系統建置失敗

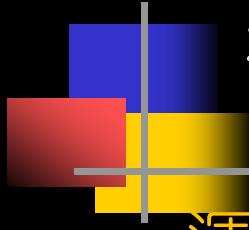


建置資料倉儲所引發的問題

- 異質性資料庫整合複雜度超過原本的預估：難以預估其真正的複雜度。
- 來源資料庫沒有時間上的考量，失去資料隨著時間變化的過程與歷史：傳統資料庫一旦更新資料後，舊有的資料便會被覆蓋，所以無法保存。
- 資料倉儲往往需要分析資料隨著時間的變化。

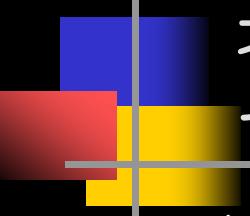
資料倉儲系統架構與元件





資料倉儲系統中的資料類型

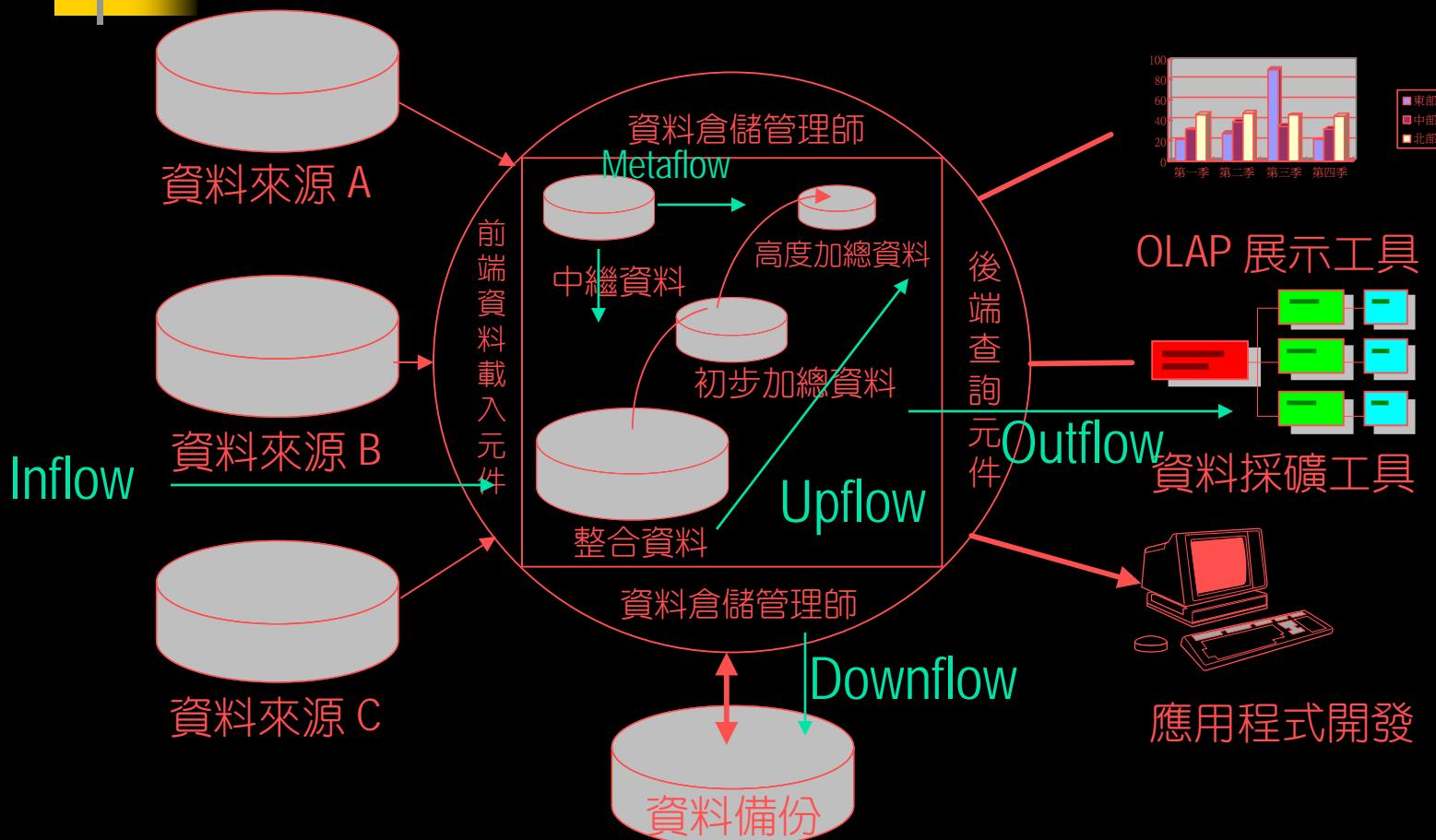
- 運算資料 (Operational Data)：以 On-Line Transaction Processing Systems (OLTP Systems) 所處理過資料組成。
- 整合後的詳細資料 (Integrated Detail Data)：將所需的運算資料整合、轉換而成整合後的詳細資料。
- 預先加總的資料 (Summarized Data)：對某些常查詢的統計資料做預先加總的動作。
- 中繼資料 (Meta-Data)：前述的整合資料必須配合能夠用來描述它們的資料。
- 備份資料 (Backup Data)：適時備份整合後的資料。



資料倉儲管理師

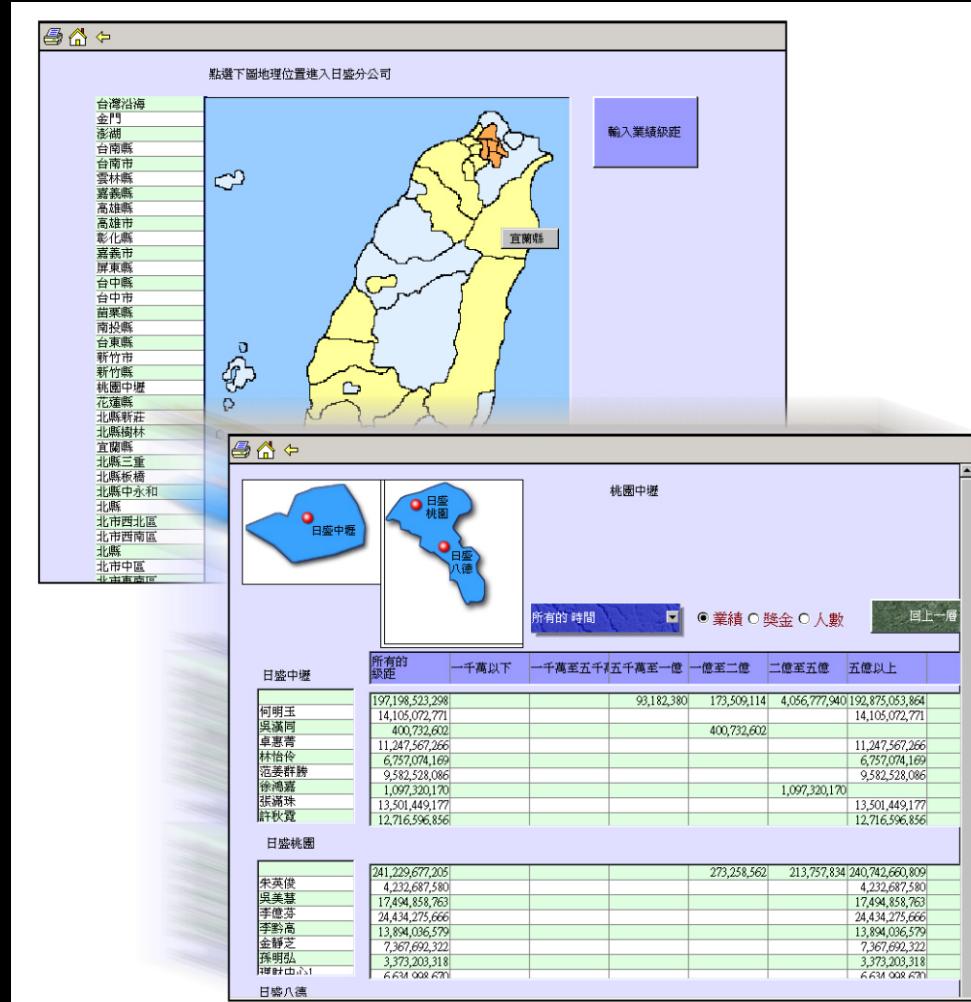
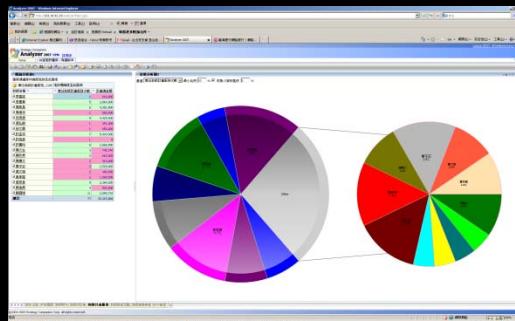
- 確保資料的一致性，
- 轉換與合併來源資料到資料倉儲的表格中，
- 建立視界 (Views) 或索引 (Indexes) 以應付使用者對資料內容與效能的需求，
- 產生預先加總的資料，
- 負責資料倉儲的資料備份

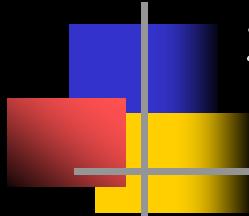
資料倉儲系統中的資訊流



資料倉儲展示與查詢工具

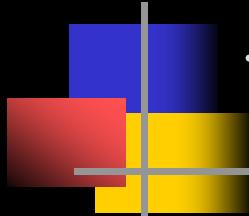
- Brio (<http://www.brio.com>)、
- 德國 Arcplan Insight 公司 (<http://www.arcplan.com>)
- Cizer (<http://www.cizer.com>)
- Analyzer (<http://www.analyzer.com.tw>) 台灣睿智)
- Cognos (<http://www.cognos.com/tw/index.html>)
- Oracle (JD Edwards, Peoplesoft)



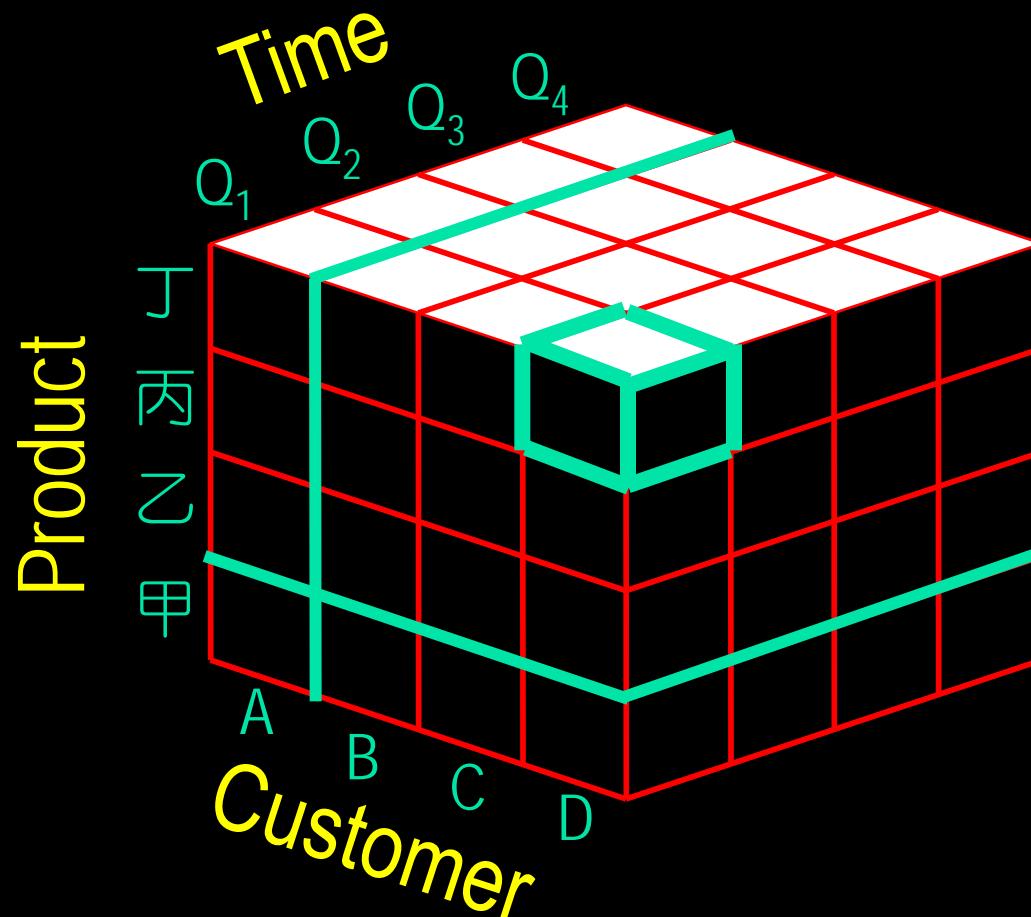


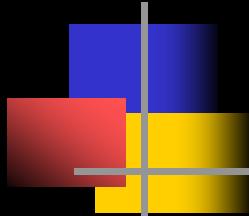
資料倉儲的設計—維度模式

- 「維度模式」(Dimensional Model)：資料庫被看成是一個二維、三維、或更多維資料的「方塊」(Cube)，可以沿著 Cube 的各個維度 (Dimension) 做切面 (Slicing)，將資料切割出來。
- 例如：將銷售資料看成是 Cube，其中有時間 (Time)、產品 (Product) 以及客戶 (Customer) 三種維度
- 則 Cube 中的任一小方塊代表某一類客戶在某個時間點購買了某樣產品的資料。

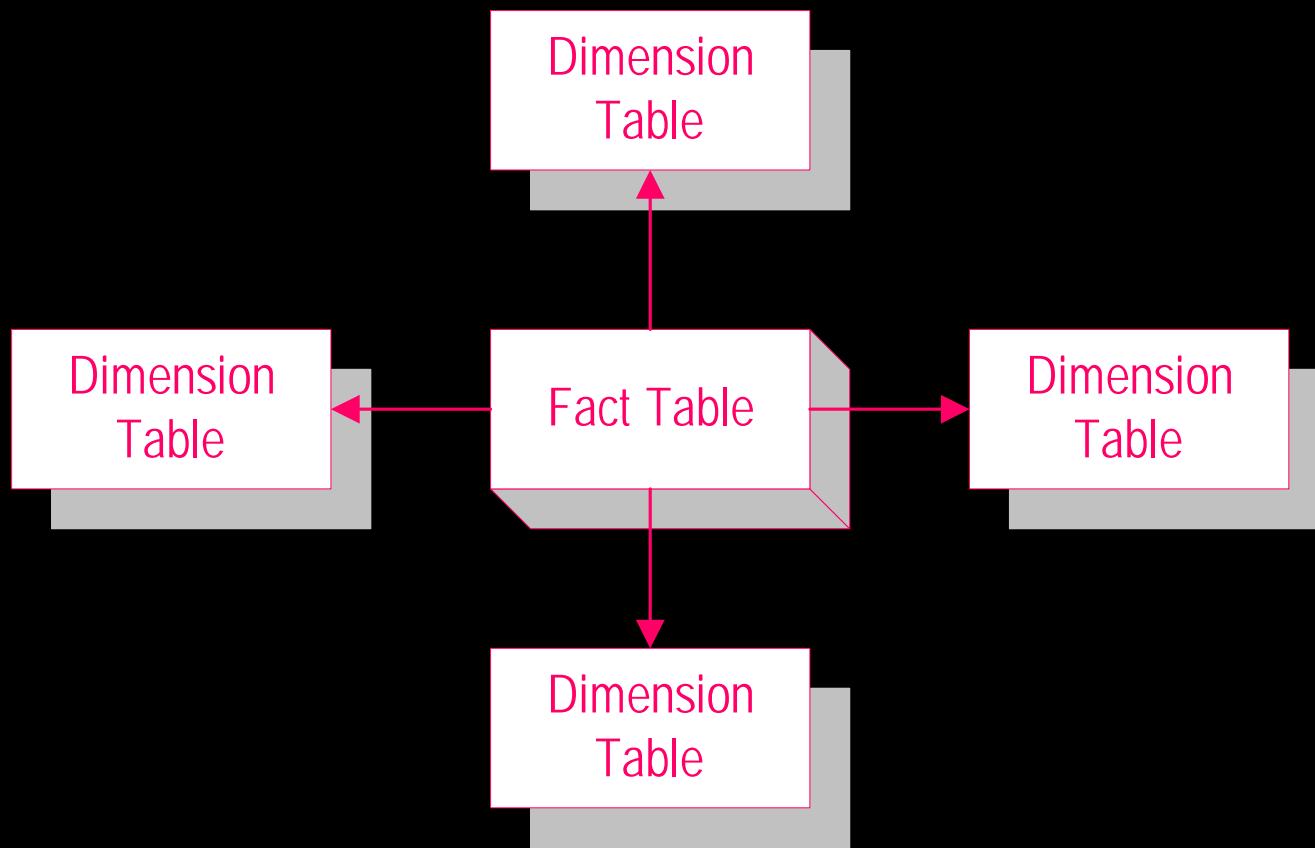


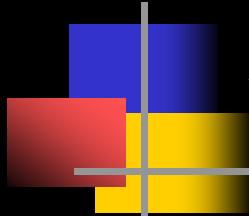
維度分析



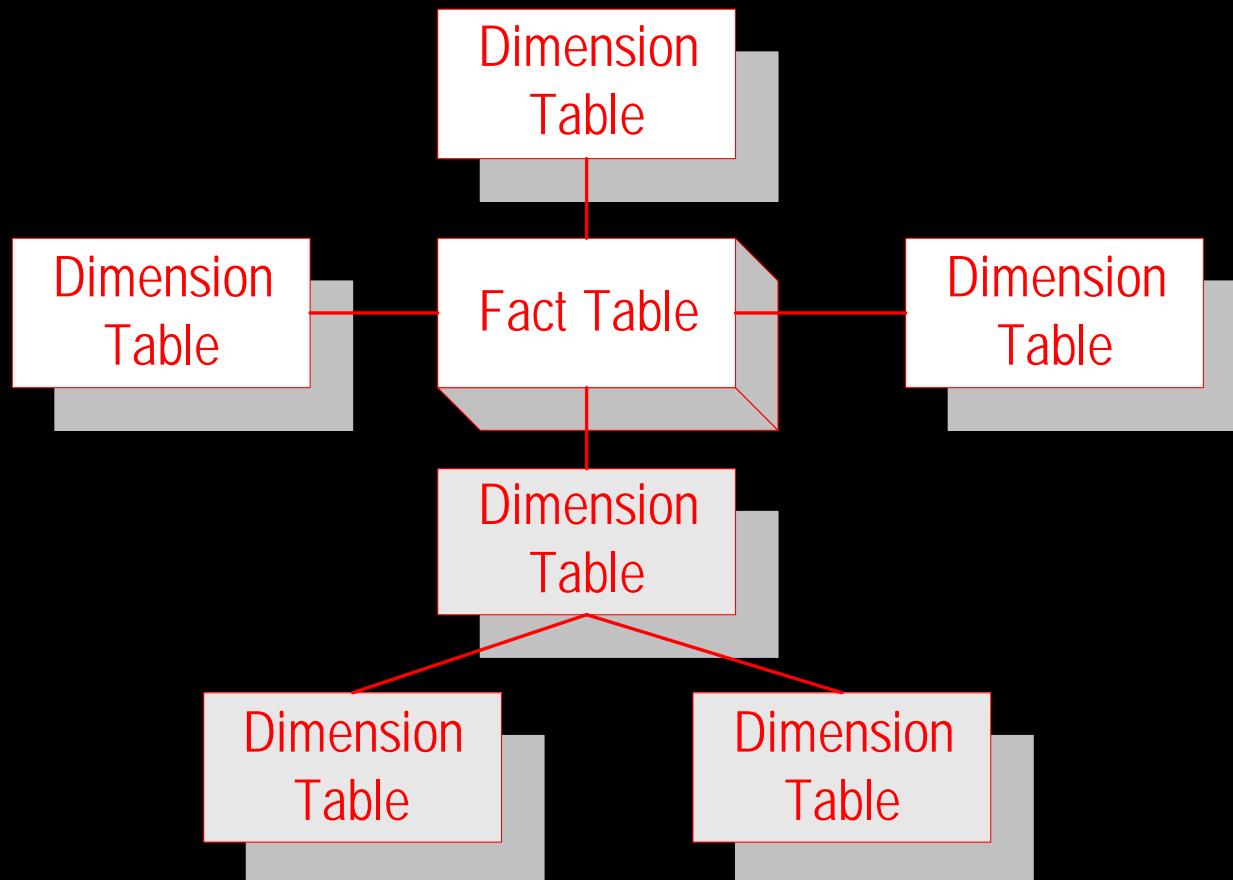


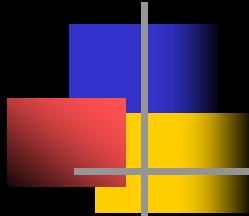
星狀結構的維度模式



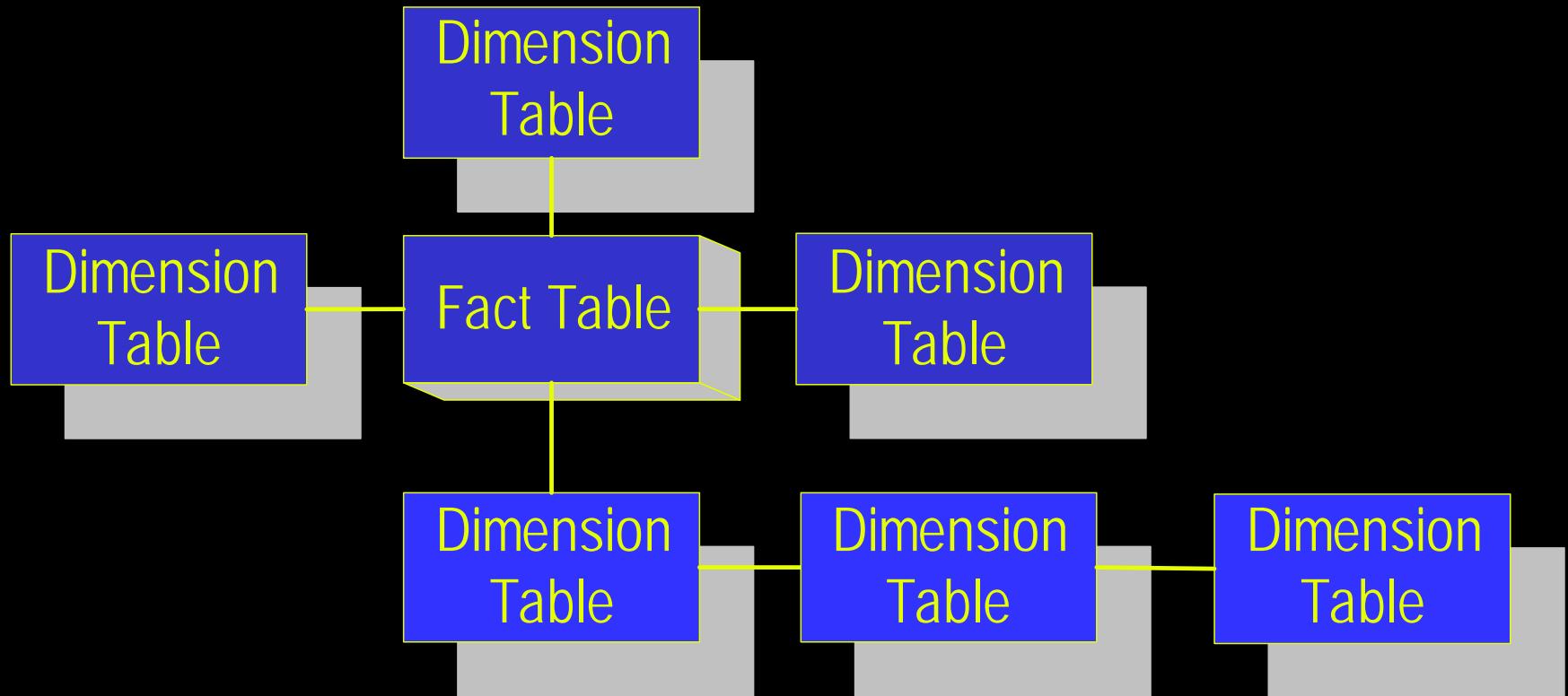


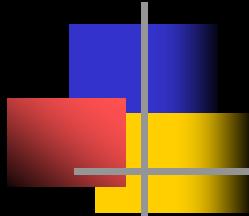
SnowFlake 雪花結構





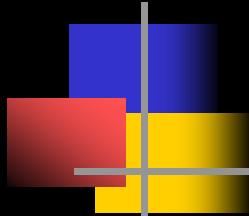
也是雪花結構





星座綱要 (Constellation Schema)

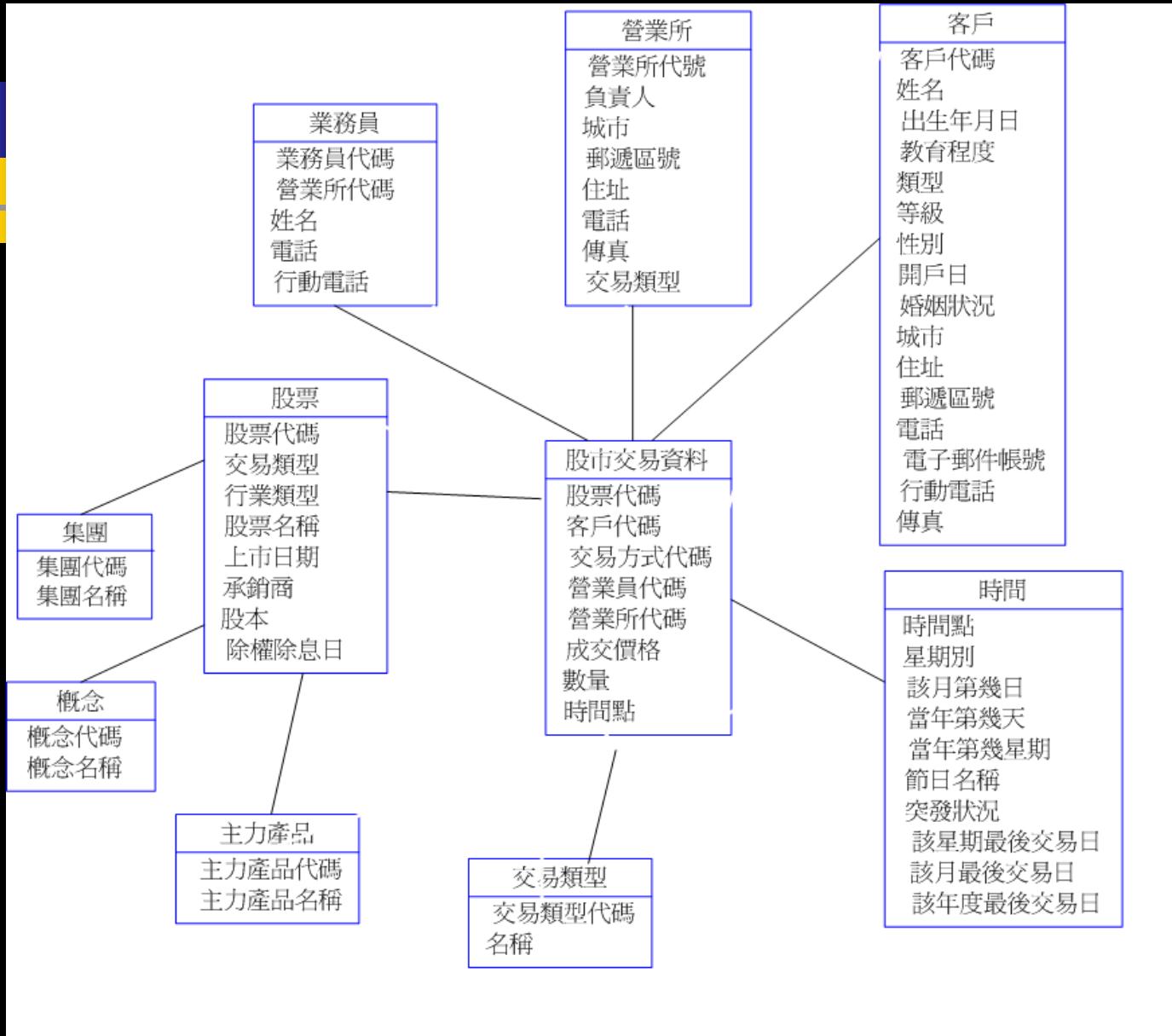
- SQL Server 2008 容許使用者將許多的 Star Schema 或 Snowflake 組成「星座綱要」(Constellation Schema) 的概念，
- 一次呈現在管理畫面中進行管理，
- 可以將整個星座綱要當成一個整合的 Cube 看待

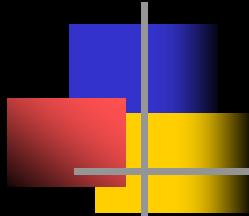


星狀綱要四大元件

- **事實表格** (Fact Tables)：儲存已量化的數值資料，如：銷售量、部門的預算等，是分析的主角，也是整個星狀綱要的中心表格。表格的內容通常非常大，通常已經是高度正規化或不必正規化。
- **維度表格** (Dimension Tables)：存放要對事實表格做分析的觀察特性。最好能鉅細靡遺地存放所有相關資訊。內容較少，可以進一步做正規化，形成雪花結構 (Snowflake)

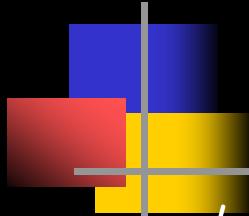
複雜的星狀綱要範例





星狀綱要四大元件

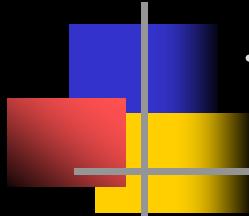
- **屬性** (Attributes)：用來藉以觀察事實表格的屬性，可以讓我們對事實表格做分類、過濾以及搜尋之用
- **屬性階層** (Attribute Hierarchies)：將屬性安排成完整的屬性階層，以便用來做深入探測 (Drill Down) 或提升 (Roll Up) 的分析
- Time 維度幾乎是必備維度。SQL Server 2008 的 Analysis Service 也有預設支援機制來建立此維度，不必自行建立 Time Dimension Table。



Drill Down VS. Roll Up

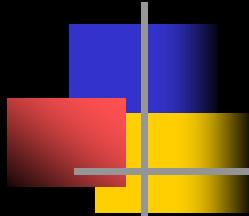
- 如果原資料庫中就有細分類別，則可以向下探測 (Drill Down) 或向上提升 (Roll Up)

Date	Customer	Product	Price
1998/7/18	Jesse	Car	500,000
1998/8/25	Frank	Phone	5,000
1998/9/4	Mike	Desktop Computer	80,000
1998/10/3	Mike	Car	530,000
1998/11/18	Jesse	Laptop Computer	70,000
1998/12/25	Frank	Desktop Computer	66,000



維度分析結果

Product (1998)	Jesse	Frank	Mike	Total
<i>Car</i>	500,000	0	530,000	1030,000
<i>Computer</i>	70,000	66,000	80,000	216,000
<i>Phone</i>	0	5,000	0	5,000
Total	570,000	71,000	610,000	1251,000



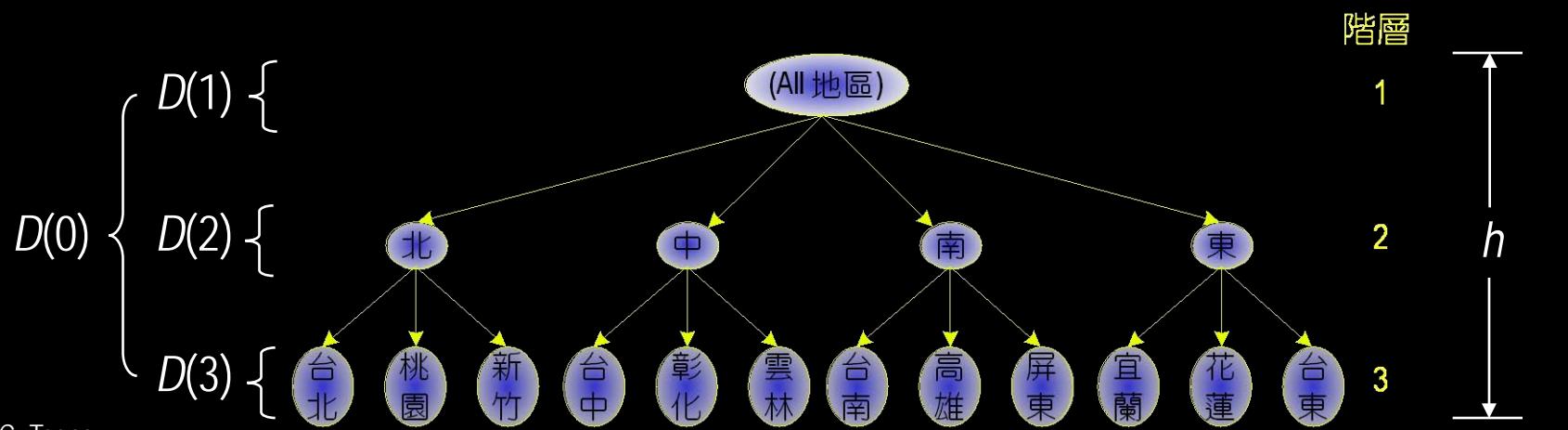
Drill Down VS. Roll Up

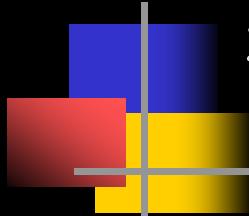
Computer 可以 Drill Down 分析 Laptop 與 Desktop 兩類

Product (1998)		Jesse	Frank	Mike	SubTotal	Total
Car		500,000	0	530,000	1030,000	1030,000
Computer	Laptop	70,000	0	0	70,000	216,000
	Desktop	0	66,000	80,000	146,000	
	Phone	0	5,000	0	5,000	5,000
Total		570,000	71,000	610,000	1251,000	1251,000

資料倉儲相關元素的正規定義

定義 16.1: [維度 (Dimension)] 一個「維度」(Dimension) D 是一個具有 h ($h \geq 1$) 個階層 (Level) 的樹狀結構，用來表示一群關鍵字彼此間的階層關係 (Hierarchical Relationships)。結構中的節點稱為「成員」(Member)，其根節點 (Root) 代表著整個維度下所有節點的整體概念，我們以 “(All D)” 或 $D(1) = \{(All D)\}$ 來表示；同理 $D(2)$ 則代表階層 2 的整體概念，也就是說 $D(2)$ 為階層 2 上的所有節點所組成的集合，依此類推。另外，我們以 $D(0)$ 表示該維度中所有節點的集合，即 $D(0) = \bigcup_{1 \leq i \leq h} D(i)$ ，其中 h 為該維度的樹狀結構高度。

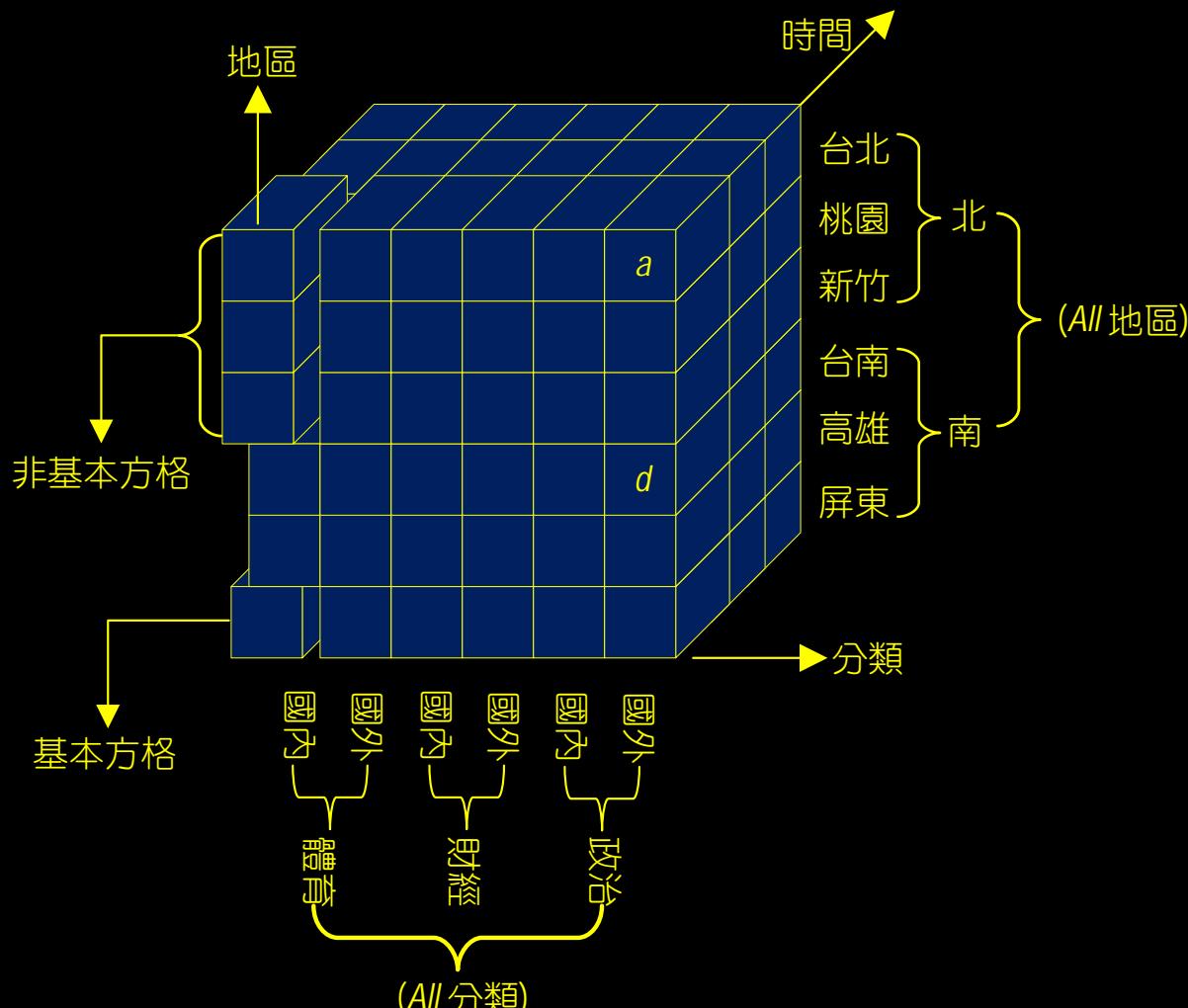


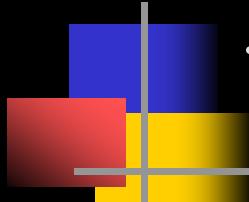


資料倉儲相關元素的正規定義

- 對於任一個維度 D 而言：
 - 由任一個內部節點向下展開所有子節點稱為「向下鑽研」(Drill-Down)。
 - 由任一節點往父節點收縮則稱為「向上提昇」(Roll-Up)。
- 下頁定義「方格」(Cell) 與「資料方塊」(Data Cube)...

資料倉儲相關元素的正規定義



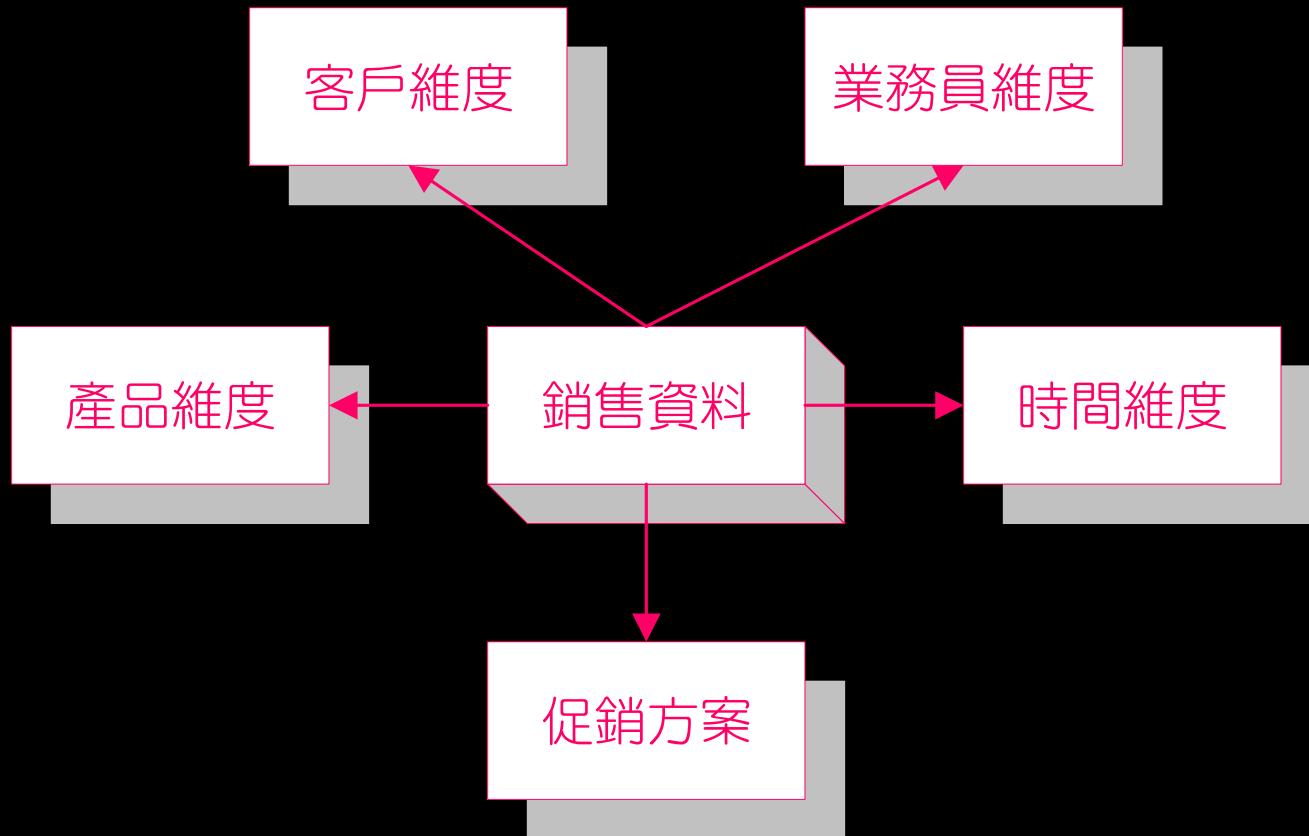


維度模式的設計步驟

- 選擇商業應用程序(Business Model) 來做為模式，例如：訂購的程序、盤點、運送的程序。
- 依照所選的商業應用程序，選擇所欲記錄的計算單位是以每天、週、月或季為基礎。
- 選擇所要的維度，並訂定維度表格鉅細靡遺的屬性，並予以妥善分類。如果屬性太多，予以切割，形成「雪花結構」的組織方式。
- 選擇所要可以累加的測量事實。

資料倉儲規劃

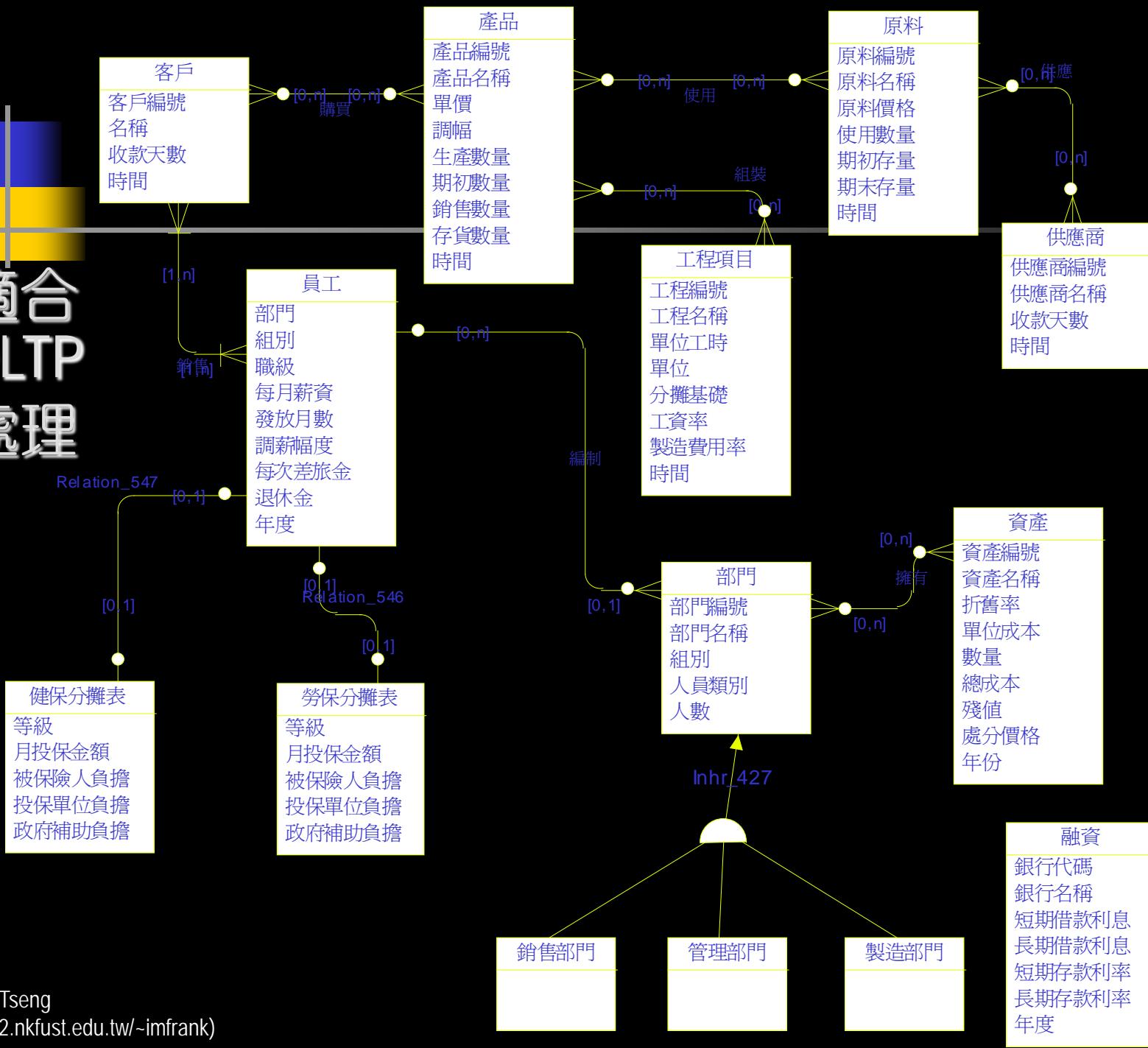
適合
OLAP
分析

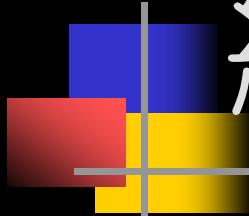


資料倉儲結構很簡單

傳統資料庫規劃很複雜

適合 OLTP 處理



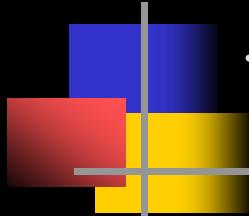


維度表 Bookstores_D

Bookstores_D

<u>no</u>	<i>name</i>	<i>rank</i>	<i>city</i>	<i>region</i>
1	巨蟹書局	20	臺北市	北
2	射手書局	10	高雄市	南
3	水瓶書局	30	新竹市	北
4	天秤書局	20	臺中市	中
5	獅子書局	30	臺南市	南
6	雙魚書局	20	臺東市	東
7	雙子書局	10	花蓮市	東

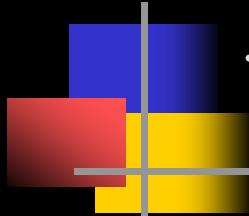
Region ⊃ city ⊃ name



維度表 Books_D

Books_D

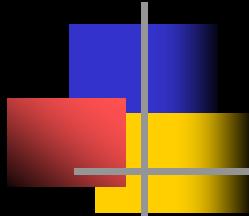
<i>id</i>	<i>bookname</i>	<i>category</i>	<i>author</i>	<i>price</i>	<i>publisher</i>
1	三國演義	文學類	羅貫中	120	古文出版社
2	水滸傳	文學類	施耐庵	170	中庸出版社
3	紅樓夢	文學類	曹雪芹	170	春秋出版社
4	西遊記	文學類	吳承恩	140	聊齋出版社
5	水經注	文學類	酈道元	120	易經出版社
6	UNIX系統	電腦類	周韻寰	190	中庸出版社
7	資料庫系統	電腦類	曾守正	300	春秋出版社
8	Web架站實務	電腦類	張三	250	中庸出版社
9	英美文學讀本	語言類	李四	250	易經出版社
10	日文讀本	語言類	王五	320	中庸出版社
11	德語入門	語言類	毛六	240	易經出版社
12	應用法語入門	語言類	李四	280	古文出版社



維度表 Time_D

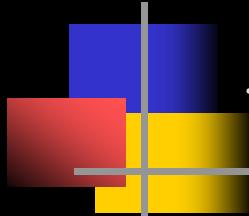
Time_D			
<i>tid</i>	<i>year</i>	<i>month</i>	<i>day</i>
1	1999	1	9
2	1999	2	7
3	1999	2	8
4	1999	2	9
5	1999	3	12
6	1999	3	18
7	1999	4	12
8	1999	4	15
9	1999	5	14
10	1999	7	10

11	1999	7	15
12	1999	8	11
13	1999	8	17
14	1999	9	14
15	1999	9	20
16	1999	10	20
17	1999	12	18
18	1999	12	19
19	1999	12	21



時間維度通常需要的屬性

- Time_key
- day_of_week
- day_number_in_month
- day_number_overall
- week_number_in_year
- week_number_overall
- month
- month_number_overall
- Quarter
- fiscal_period
- weekday_flag
- holiday_flag
- last_day_in_month_flag
- season (重要節日)
- event (奧運開幕, 總統大選, 王建民登板, ...)

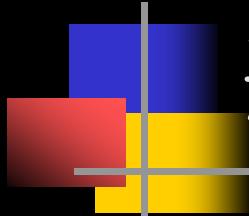


事實表 Order_F

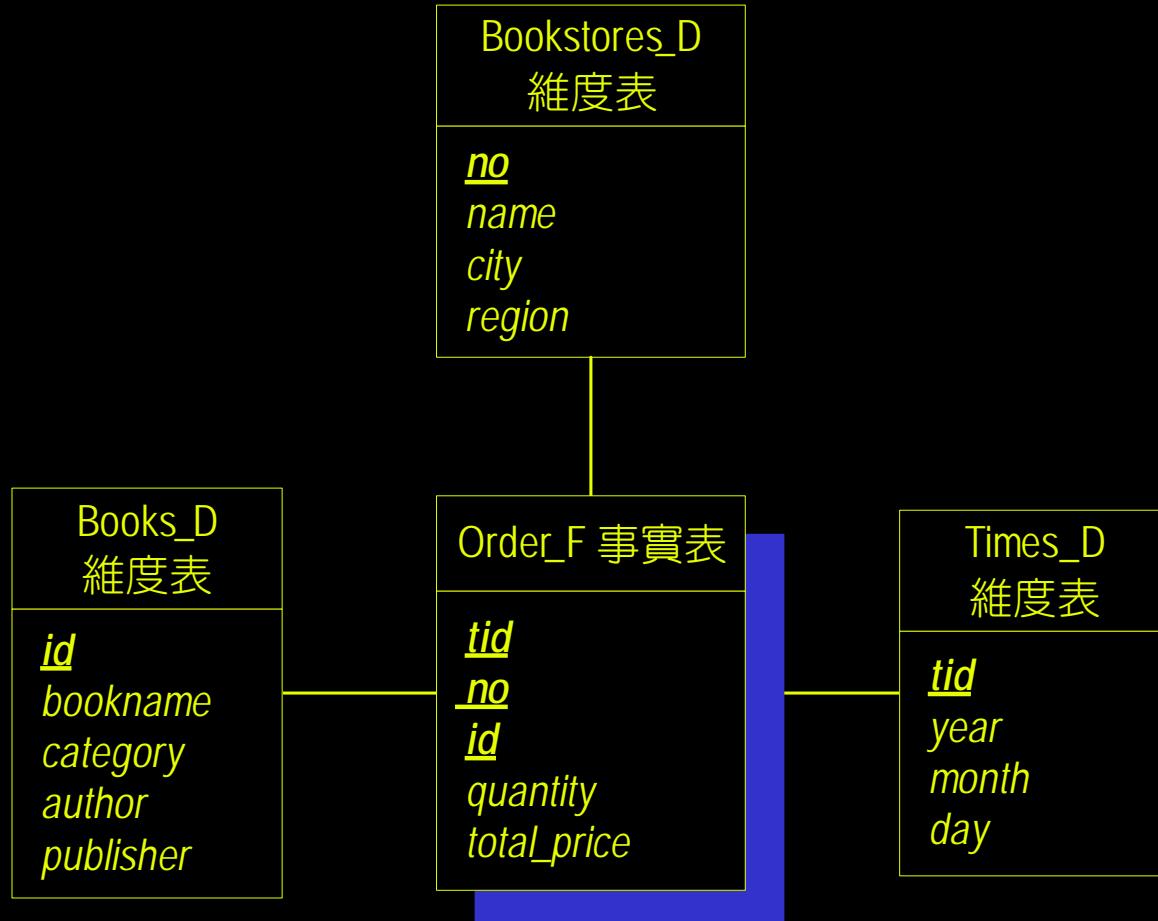
Order_F

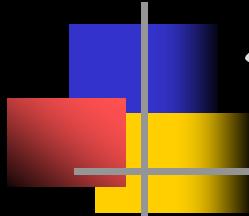
<u><i>tid</i></u>	<u><i>no</i></u>	<u><i>id</i></u>	<u><i>quantity</i></u>	<u><i>total_price</i></u>
1	2	1	30	3600
2	6	9	10	2500
3	1	1	30	3600
4	5	1	20	2400
5	1	2	20	3400
5	6	10	10	3200
6	4	2	20	3400
7	2	2	40	6800
8	6	11	10	2400
9	1	3	40	6800
10	5	10	20	6400

11	1	4	20	2800
11	3	2	20	3400
12	5	7	40	12000
13	1	5	10	1200
14	5	6	30	5700
15	1	6	10	1900
16	4	4	30	4200
17	5	8	60	15000
17	6	12	10	2800
18	5	12	10	2800
19	4	5	40	4800



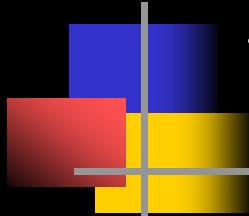
維度模式範例





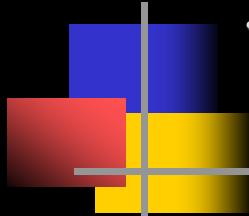
分析結果

	中	北	東	南
All Books	90	150	40	250
文學類	90	140		90
三國演義		30		50
水經注	40	10		
水滸傳	20	40		40
西遊記	30	20		
紅樓夢		40		
電腦類		10		130
UNIX 系統		10		30
Web 架站實務				60
資料庫系統				40
語言類			40	30
日文讀本			10	20
英美文學讀本			10	
德語入門			10	
應用法語入門			10	10



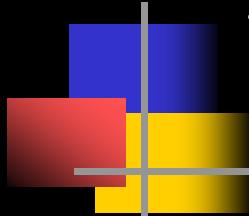
資料倉儲與資料庫的差別

- 資料庫：資料產生是透過人為的輸入、點銷售系統 (Point of Sale, POS)、刷卡資料等來取得
- 處理對象是當前的資料，尚未經過匯總與整理，會有許多的錯誤資料或虛值 (Null Values) 隱藏於其中。
- 一般查詢應用可以容忍這些錯誤資料的存在
- 以最新的資料覆蓋 (Overwrite) 舊有的資料，比較不會隨時間累積大量的資料



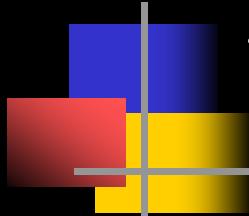
資料倉儲與資料庫的差別

- 資料倉儲：處理的對象是歷史資料
- 需經過匯總與整理，力求完整精確，因為絲毫的資料誤差或缺項就可能產生錯誤的決策
- 需要所謂的「資料品質管理師」(Data Quality Manager, DQM)，簡稱「資料品管師」負責資料倉儲的資料品質管制工作。
- 資料來源可能整合其它資料來源。如：競爭對手的資料，或政府所提供的產業資料等。



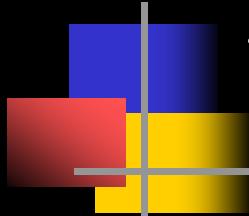
資料倉儲與資料庫的差別

- 資料倉儲：異質性資料庫整合。資料會隨著時間而漸漸累積，導致數量相當龐大。
- 傳統資料庫的應用重點在於：透過異動處理將資料輸入資料庫中 (Put Data In)
- 資料倉儲應用重點則在於：透過線上分析工具將資訊輸出到螢幕或報表上 (Get Information Out)。



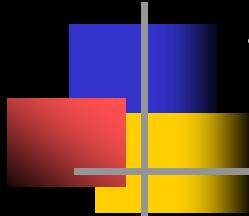
資料倉儲與資料庫的差別

- 兩者的資料內容將時間因素納入考慮的話是一樣的，差別在於在內部資料儲存結構與外部資料呈現方式上的不同。
- 衍生出來的軟、硬體架構、系統管理與行政管理方式也與傳統資料庫系統有所不同



資料倉儲與資料庫的差別

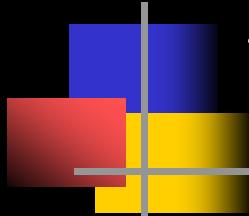
- 資料庫與資料倉儲兩類應資料分開存放
在不同的系統上，各別管理以避免混淆。
- 資料庫規劃通常以「個體-關係」(E-R) 模式正規化理論，是「由下往上」(Bottom-Up) 的「資料相依模式」(Data Dependencies Model)



資料倉儲與資料庫的差別

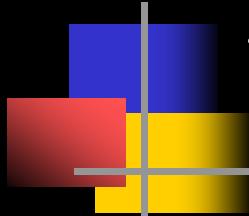
- 正規化的規劃造成了關聯表中的資料都是存放單一時間點的單筆異動資料

<i>Date</i>	<i>Customer</i>	<i>Product</i>	<i>Price</i>
1998/7/18	Jesse	Car	500,000
1998/8/25	Frank	Phone	5,000
1998/9/4	Mike	Computer	80,000
1998/10/3	Mike	Car	530,000
1998/11/18	Jesse	Computer	70,000
1998/12/25	Frank	Computer	66,000



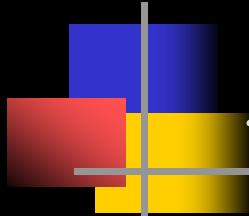
資料倉儲與資料庫的差別

- 資料倉儲的規劃是以所謂的「星狀綱要」(Star Schema)，將主要資料以單一個關聯表儲存(我們稱之為Fact Table)，並配合欲詳細分析的各種資料維度(Dimension)，以非正規化的關聯表儲存(我們稱這些關聯表為Dimension Table)
- Fact Tables 透過外來鍵參考 Dimension Table，形成以 Fact Table 為中心的外來鍵參考圖，故名「星狀綱要」。



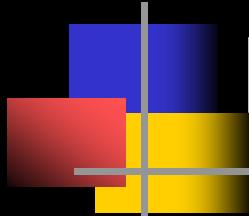
資料倉儲與資料庫的差別

- 星狀綱要的規劃方式可以說是「由上往下」(Top-Down) 的方式。目的在於將資料的各種維度展開，以利各種資料切面的分析。稱之為「維度模式」(Dimensional Model)
- 會得到少少的非正規化關聯表，但每個關聯表會有很多欄位的情況。
- 可以很快地以 [時間/產品/客戶] 做為切面，找出某個 [時間/產品/客戶] 角度的銷售狀況



建立資料倉儲所需的系統支援

- Multi-Dimensional Data Schema Support
- Multidimensional Query Language Support
- 多維度資料 Query Optimization
- 超大型資料庫的支援

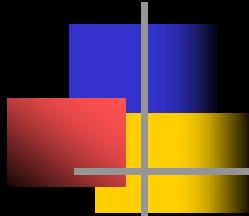


Multi-Dimensional Data Schema Support

- 提供「星狀綱要」(Star Schema) 的規劃，讓我們得以產生多維度的資料綱要，以利 OLAP 之運作。
- 這樣的綱要設計主要是在將查詢運算做最佳化
- 與傳統正規化綱要對更新運算做最佳化的目標有很大的區別。

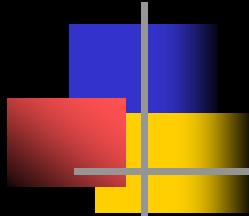
Multidimensional Query Language Support

- 大多數 OLAP 需要多重掃瞄式的查詢處理 (Multi-Pass) 以及多重巢狀的查詢處理 (Multiple Nested Query)，所以 SQL 並不適合用來做專業的資料分析。
- 通常都會在 SQL 之外再加上多維度查詢處理的擴充功能，如：MS SQL Server 2008 的 MDX (Multi-Dimensional eXpression) 擴充功能。



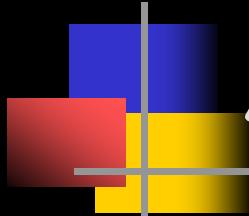
多維度資料 Query Optimization

- 星狀綱要 (Star Schema) 在使用上與傳統的關聯式查詢工具可能會有些格格不入
- 所以系統除了要另外支援資料分析的功能以外，還要對這類的查詢做最佳化。



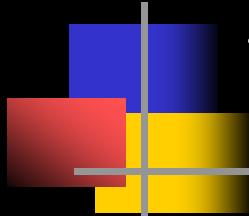
多維度資料的索引技術

- 傳統 B-tree 索引方式並無法讓系統做很有效率地處理，通常要配合
 - 「位元映對圖索引」(Bitmapped Indexes)
 - R-Tree 的索引方式方能加速此類查詢的速度。
- 目前已經被發展出來的方法可分成四類：
 - Multidimensional Array-Based Methods
 - Bit-Mapped Indices and Variations
 - Hierarchical Indexing Methods
 - Multidimensional Indices



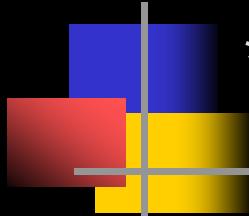
超大型資料庫的支援

- 能從各種不同格式的運作資料 (Operational Data) 將資料餵入 (Import) 到資料倉儲中，
- 提供介面將它們予以充分整合 (Integrate)。
- 例如：在 MS SQL Server 2008 的 SQL Server Integration Service (SSIS) 可以支援此一工作。
- 系統當然也要提供超大型的資料儲存空間與快速的存取效能。



資料倉儲規劃的困難點

- Slowly Changing Dimension : 資料會隨時間漸次變化，在時間序列的分析上造成困擾
- Heterogeneous data resources : 異質性的資料來源造成資料整合上的問題
- Accumulating Snapshot : 資料漸次累積需要漸進式的計算求解
- fact 與 dimension 之間的多對多關係 : 某股票同時是奧運概念股及高鐵概念股，也同時屬於國泰集團與統一集團

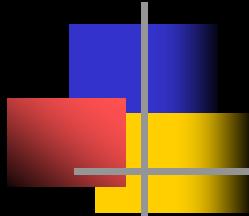


*緩變維度 (Slowly Changing Dimension)

- 緩變維度 (SCD): 某些維度的內容會隨著時間的變化，而緩慢改變其屬性內容
- 例：客戶維度的居住區域、地址、電話、婚姻狀況、公司頭銜、教育程度等，會隨著時間的改變而緩慢改變。
- 這樣的改變可能會導致某些分析結果失真。

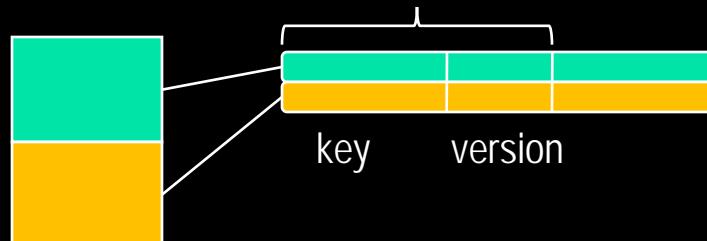


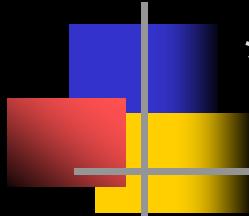
前半年在台北業績 (300 萬) 敬陪末座,
遠調屏東, 後半年業績也是敬陪末座 (3 萬),
但平均起來還有 151.5 萬, 在屏東名列前茅



*緩變維度 (Slowly Changing Dimension)

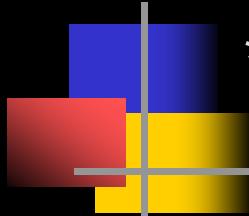
- Type I：直接用新的值將舊有的值 Overwrite
 - 作法簡單，但是會讓資料倉儲喪失追蹤歷史的能力，難以呈現過去引發該事實的維度內容。
 - 使用時機：當發現原先的維度屬性值有錯誤，因而採用正確的值來更新，與其歷史追蹤能力無關。
- Type II：維度屬性內容有新的值要修正時，則新增一筆記錄到維度表中，進行「歷史切割」(History Partition) 的作法。通常用在與人有關的維度上 (E.g., 客戶維度)





*緩變維度 (Slowly Changing Dimension)

- 如果緩變維度是指某類物品時 (像：產品維度)，則一般不會採用對事實表做「歷史切割」的作法，而是Type III 的做法
- 原因是產品通常會有生命週期，我們會在商品推陳出新時，會與舊商品同時販售一陣子，
- 兩者的銷售事實資料應該要並存於資料倉儲中，並不適合採用歷史切割的作法。



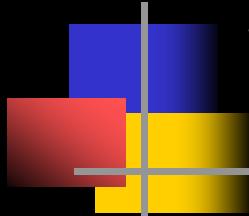
*緩變維度 (Slowly Changing Dimension)

- Type III : 只存放某個最新的維度屬性內容與其上一版的內容。

Newly Added

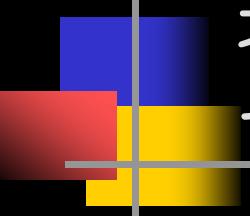
Last Value	Last Old Value	Effective Date
------------	----------------	----------------

- 只能保持資料倉儲的部份歷史追蹤能力
- 可用於產品維度的屬性值改變之上
(E.g., Honda CR-V II  CR-V III) ,
- 因為一般新推出的產品會有新的產品名稱並另外重編主鍵值，並與舊商品同時販售一陣子，而讓舊產品存續的必要性僅及於出清庫存之前的那段時間而已，所以只要存放兩個版本即足夠了。
- Type 6 (Type 1+2+3): See Wikipedia Examples



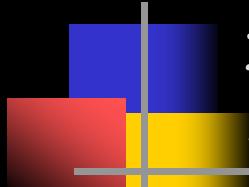
資料倉儲的四個要點

- 資料倉儲之父 W.H. Inmon (也稱 Bill Inmon) 定義 (1994)：
 - 整合的 (Integrated)：集中而統合的資料庫
 - 主題導向 (Subject-Oriented)：傳統資料庫的規劃大多以功能來劃分 (Function-Oriented)，整合後的資料則以主題 (Subject) 來區分。
 - 隨時間變動的 (Time-Variant)：資料倉儲儲存整個組織的運作歷程記錄，
 - 非暫存性 (Non-Volatile)：保存每個階段當時的狀態，所以資料一旦存入資料倉儲即被保留



資料倉儲與資料庫的比較表

特性	資料庫 (Database)	資料倉儲 (Data Warehouse)
資料的時間性	當時的運算資料	經過處理的歷史資料
資料庫的規劃方式	由下往上 (Bottom-Up)	由上往下 (Top-Down)
資料庫的綱要設計	個體-關係模式配合正規化	維度模式(e.g. Star Schema)
資料特性	無重覆儲存	大量重覆儲存，並預先加總
資料維護者	資料庫管理師 (DBA)	資料品管師 (DQM)
異動的頻率	經常異動 (故稱 OLTP)	少有異動，大多為查詢
異動的資料數量	平時均有大量的異動處理	定期大量載入並聚合加總
效能要求	須能承受大量的更新要求	查詢速度要夠快

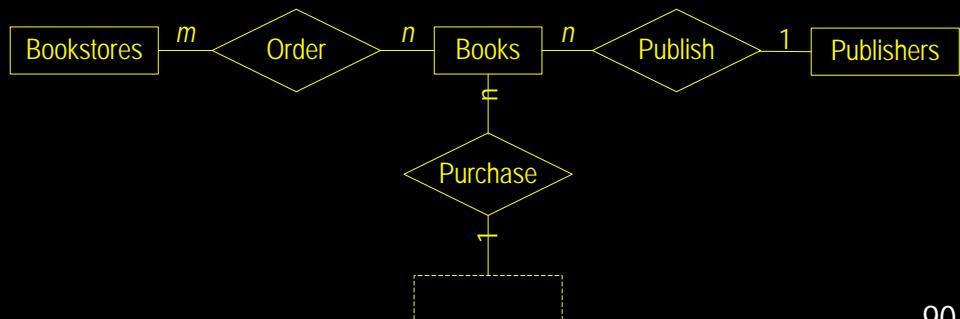
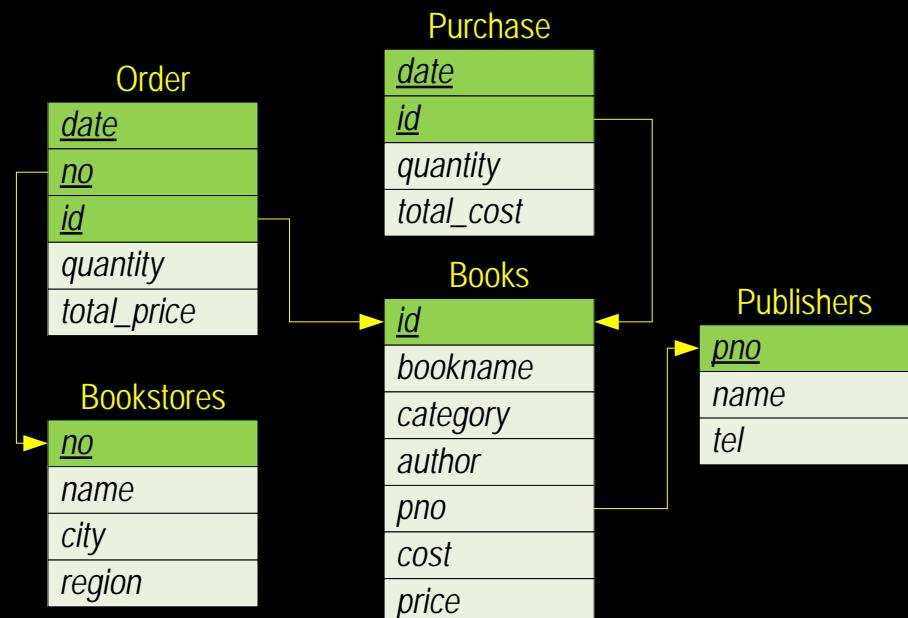
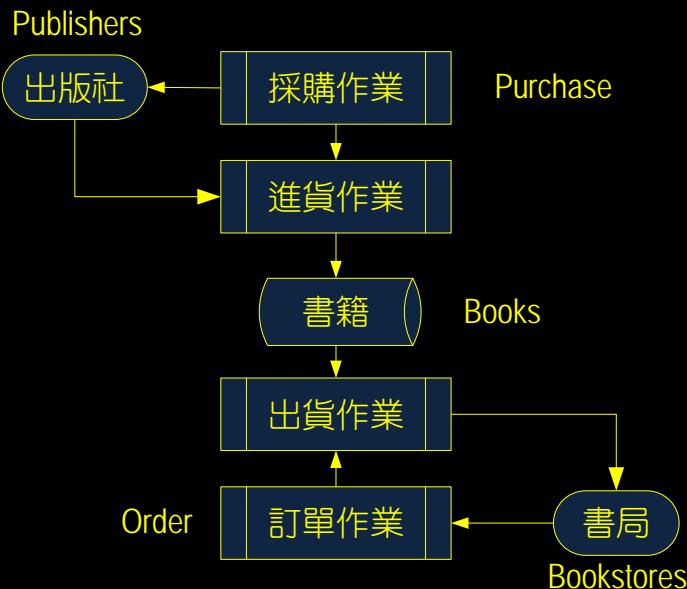


資料倉儲與資料庫的比較表

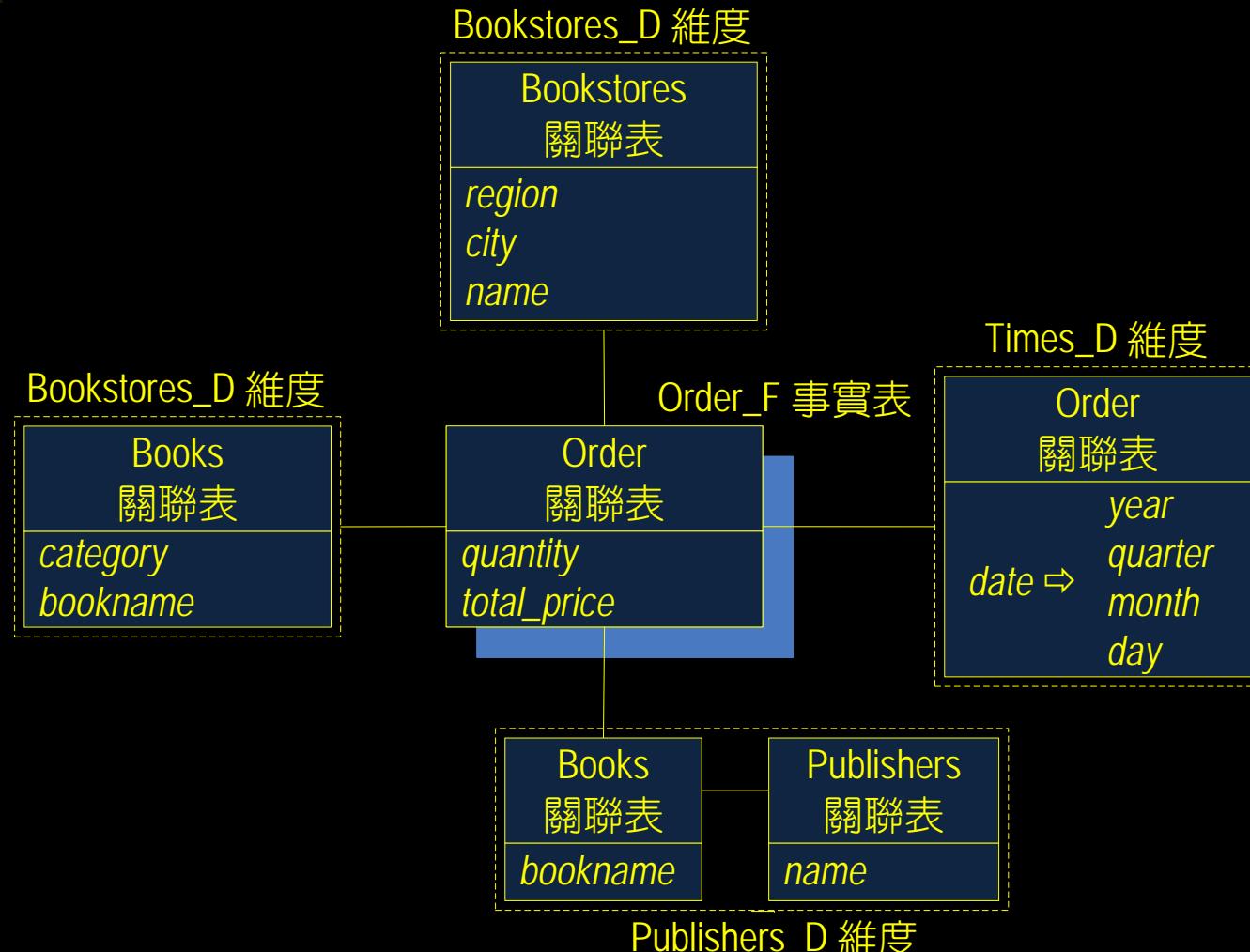
特性	資料庫 (Database)	資料倉儲 (Data Warehouse)
查詢的頻率	少量的需求	大量需求 (故稱 OLAP)
查詢的範圍	較狹隘	相當寬廣
查詢的複雜度	較單純	相當複雜
所內含的資料量	數百萬位元組 (Megabytes)	數百 Gigabytes 以上
內含資料的錯誤率	可以容忍錯誤與缺項存在	極少錯誤與資料缺項
資料的精細度	存放單一筆交易的詳細資料	存放大量加總過的資料
整合性	依功能分資料庫，未整合	整個組織的資料完全整合
主題性	依功能導向區分資料庫	依主題導向
隨時間變動的特性	少會依時間流逝增加內容	依時間的流逝而增加其內容
暫存性	只保留目前最新的資料	完整保留所有歷程資料
適合建置的系統	關聯式資料庫管理系統	多維度資料庫管理系統

以 SQL Server 2008 建立資料倉儲

一個維度模式範例



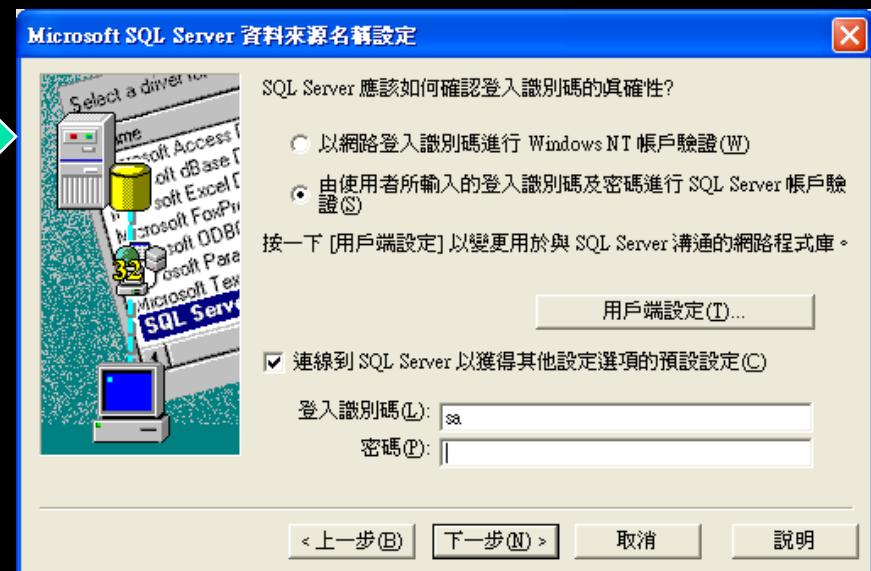
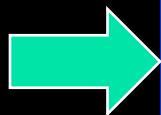
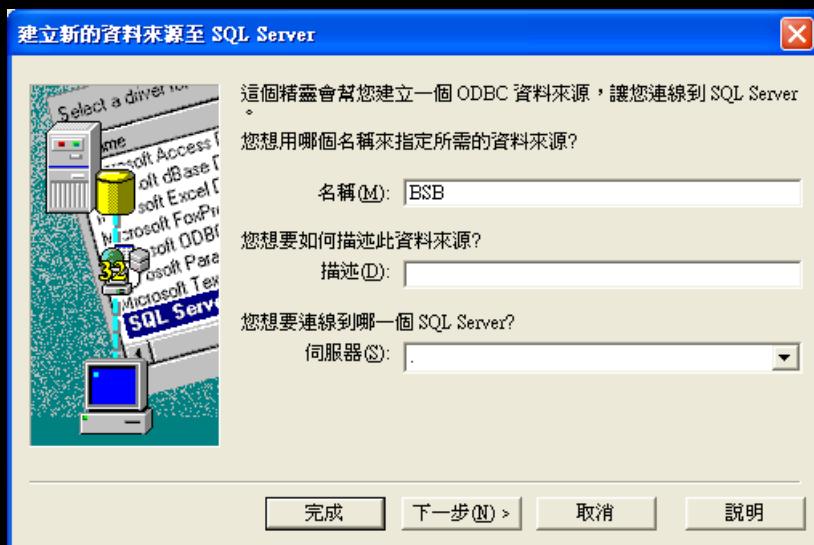
一個維度模式範例



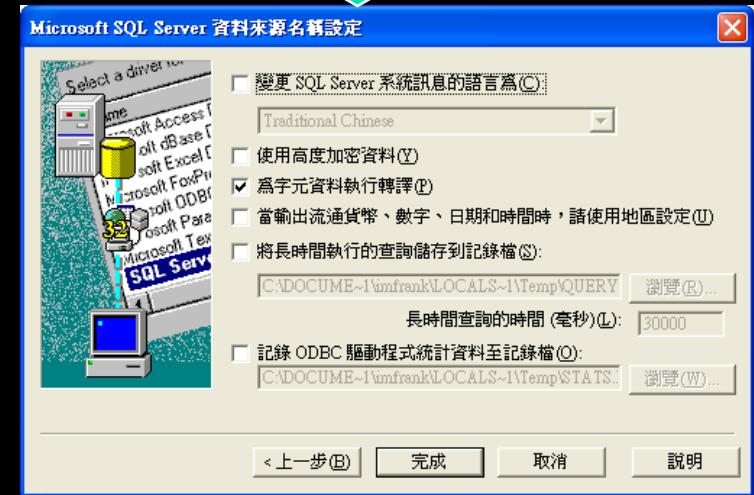
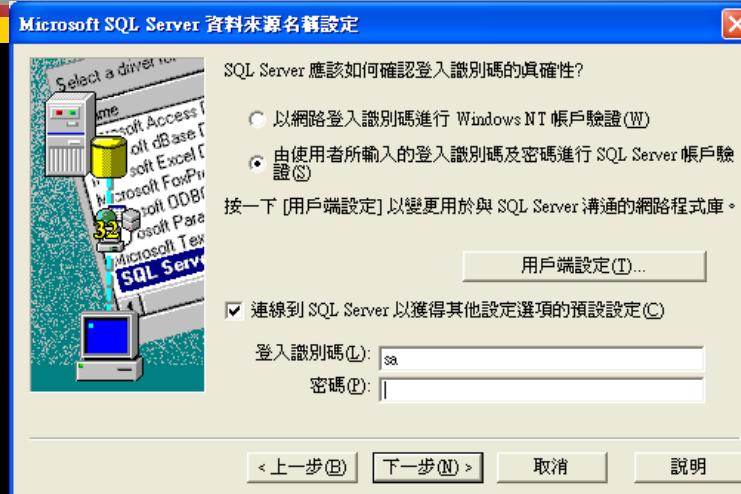
建立 ODBC 資料來源



建立 ODBC 資料來源

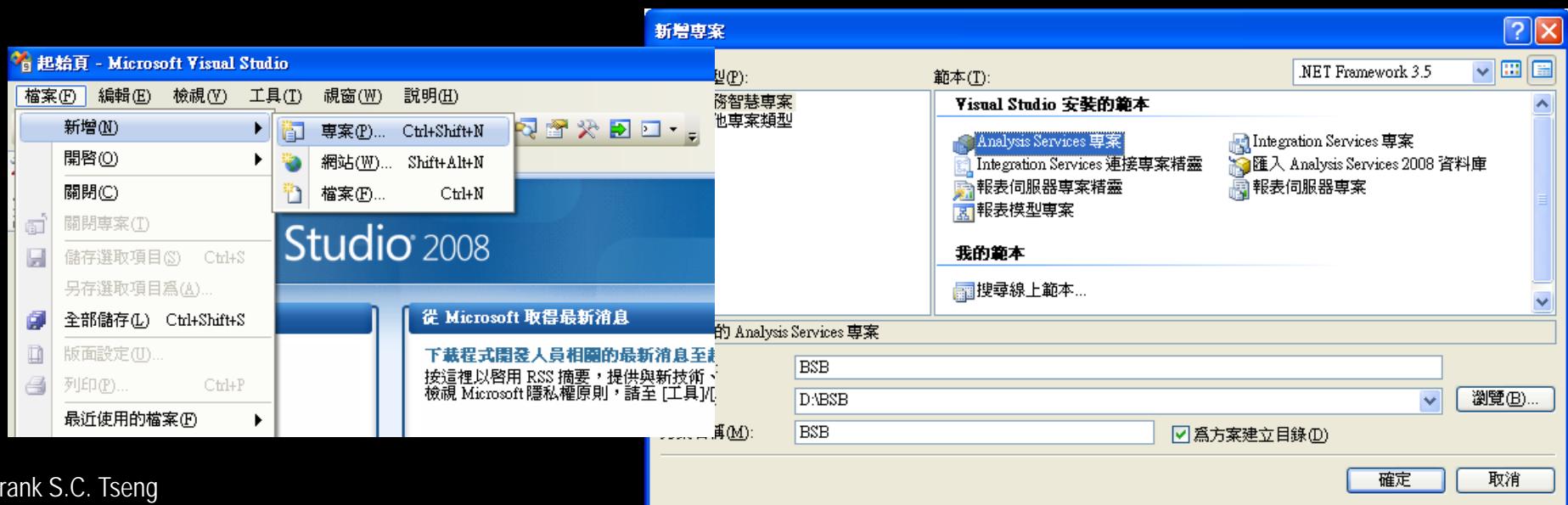


建立 ODBC 資料來源

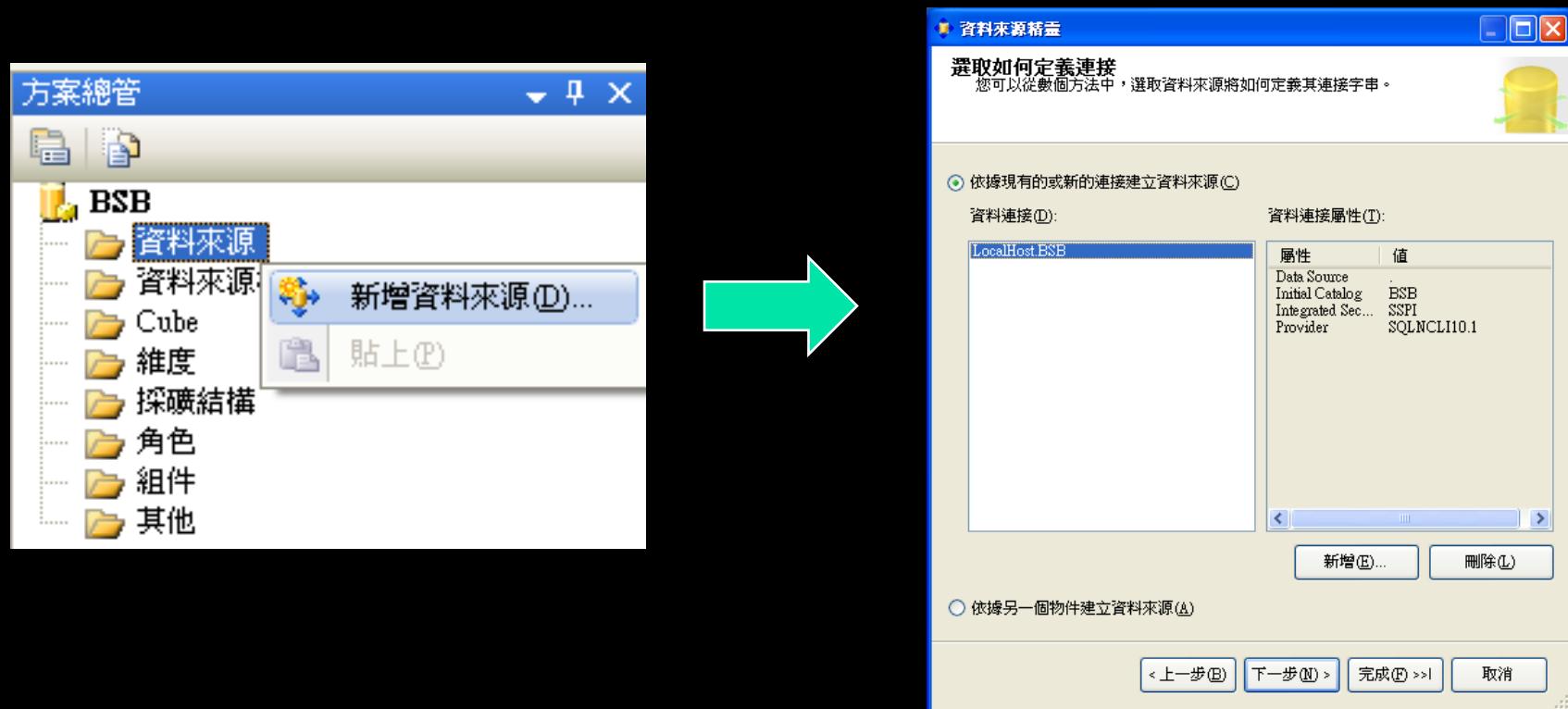


建立資料倉儲

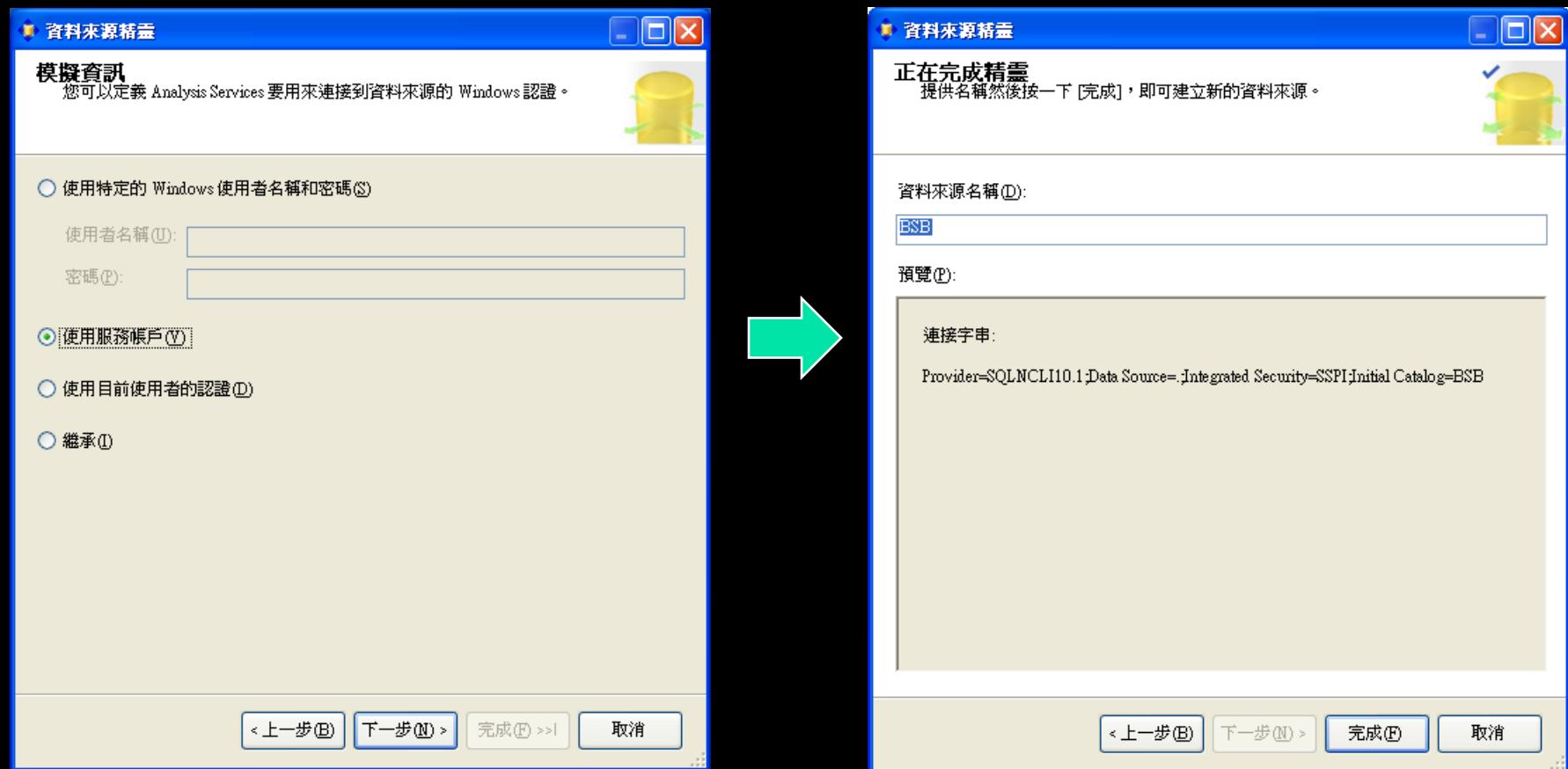
- 設定資料倉儲的資料來源與資料來源檢視
- 目的:讓使用者具備調整彈性，
因為不希望在建立資料倉儲時所進行的調整
去干擾來源資料庫的原有結構



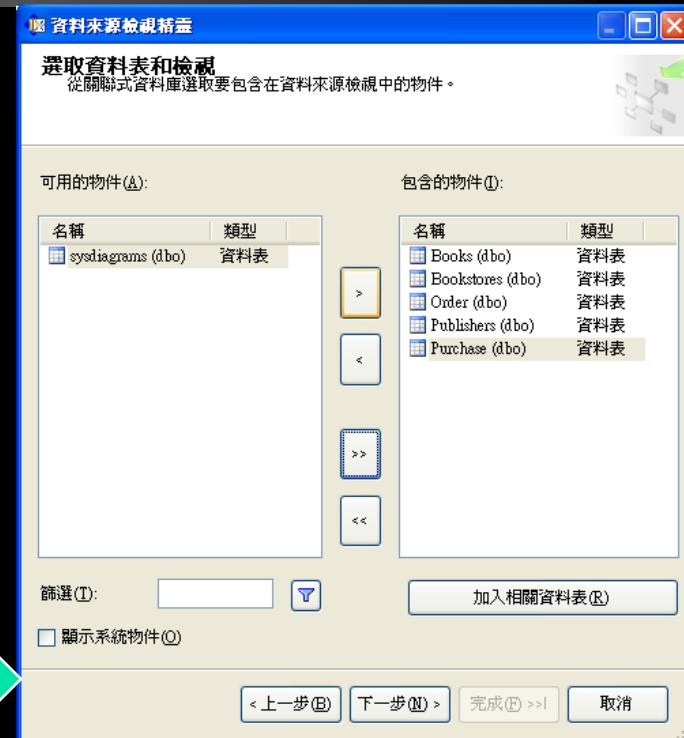
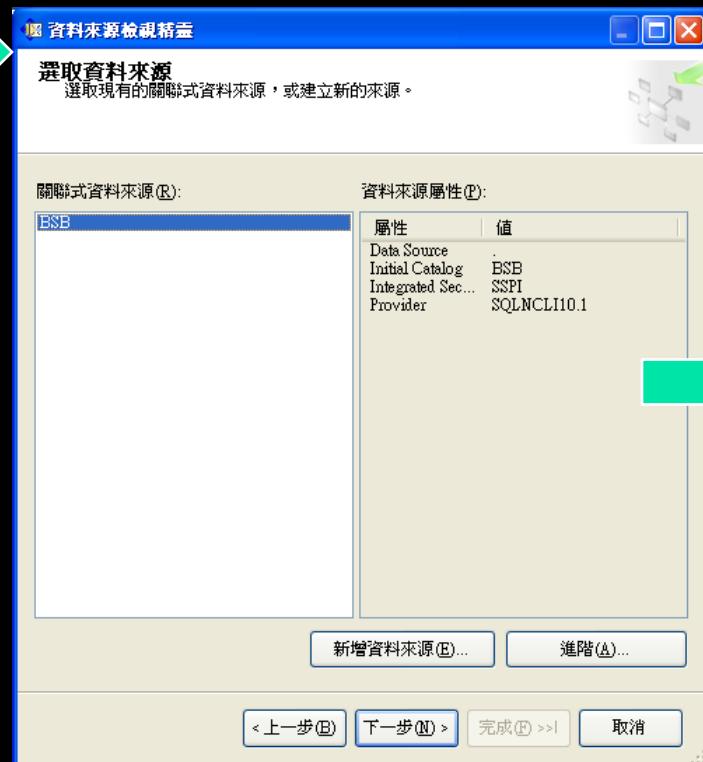
設定資料倉儲的資料來源



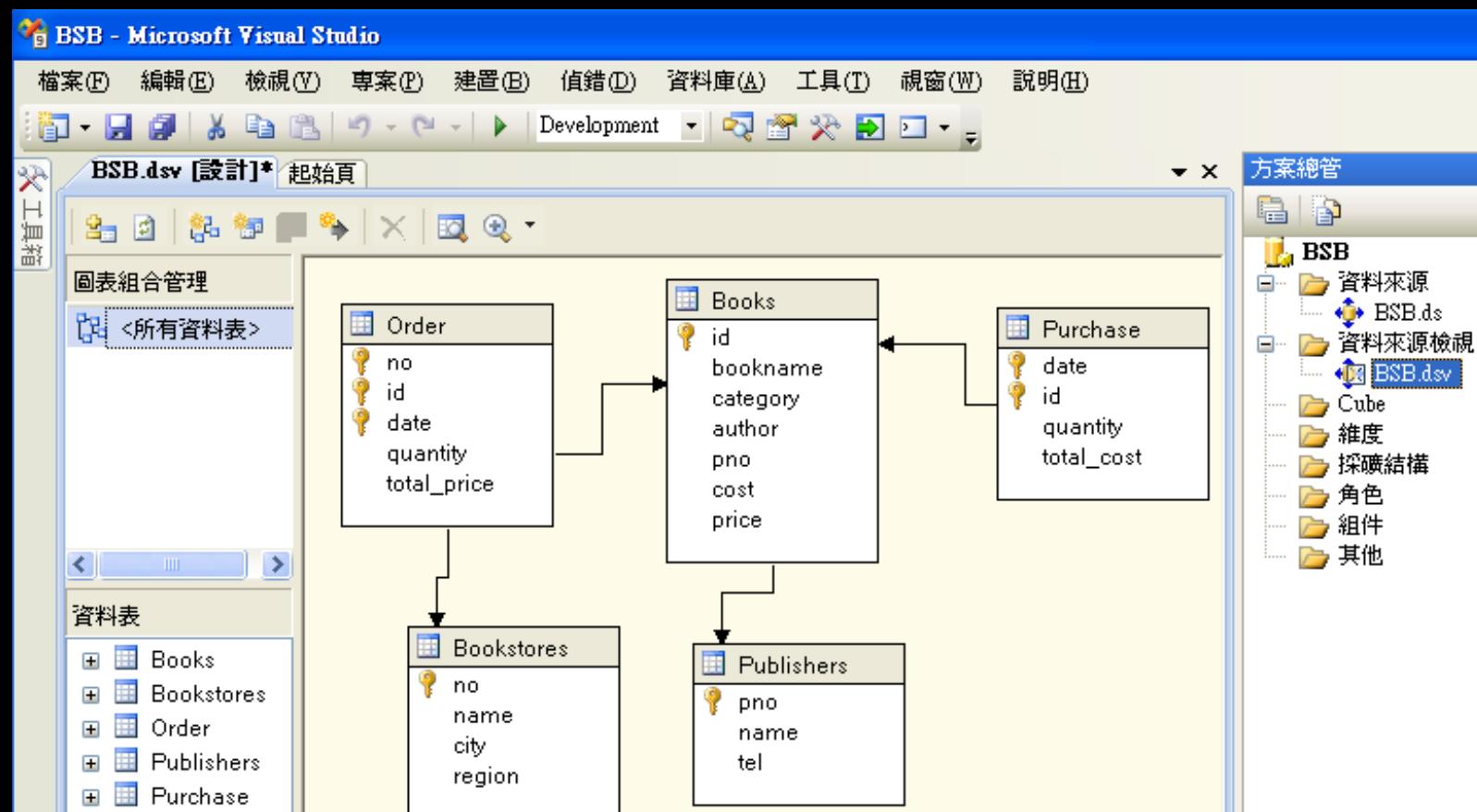
設定資料倉儲的資料來源



設定資料倉儲的資料來源檢視

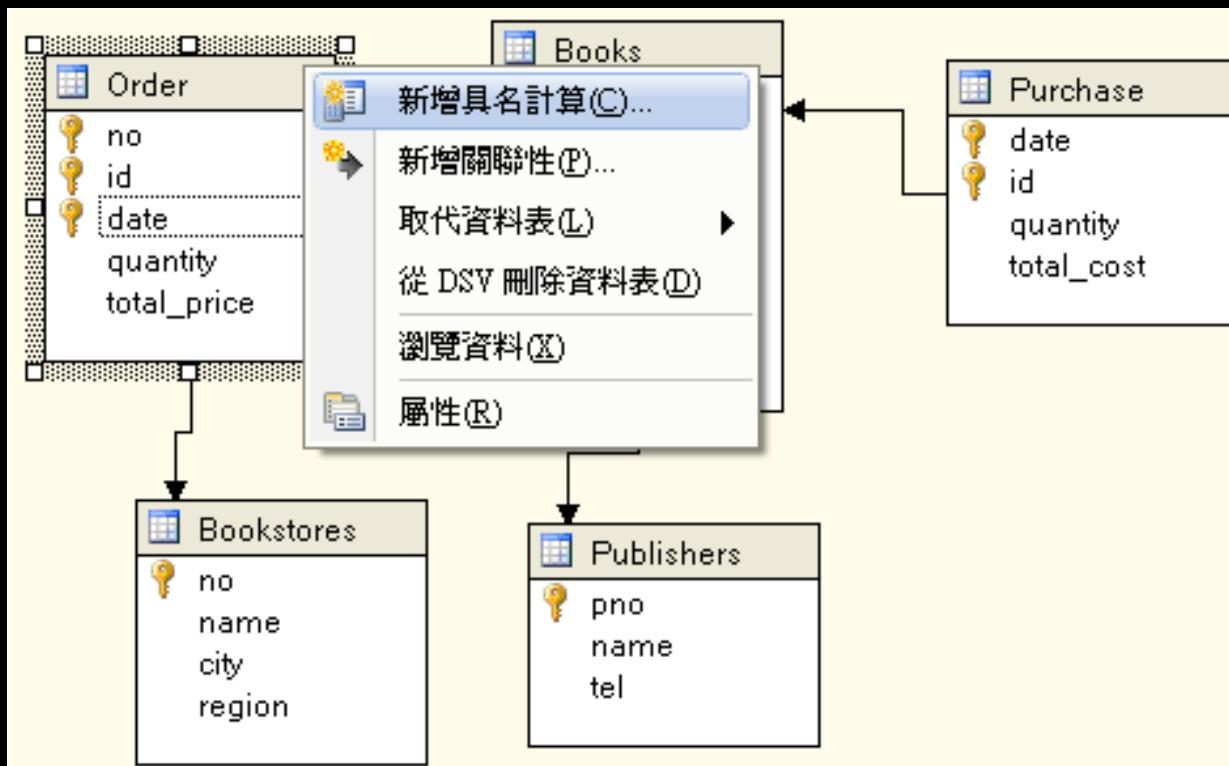


資料來源檢視設定完成

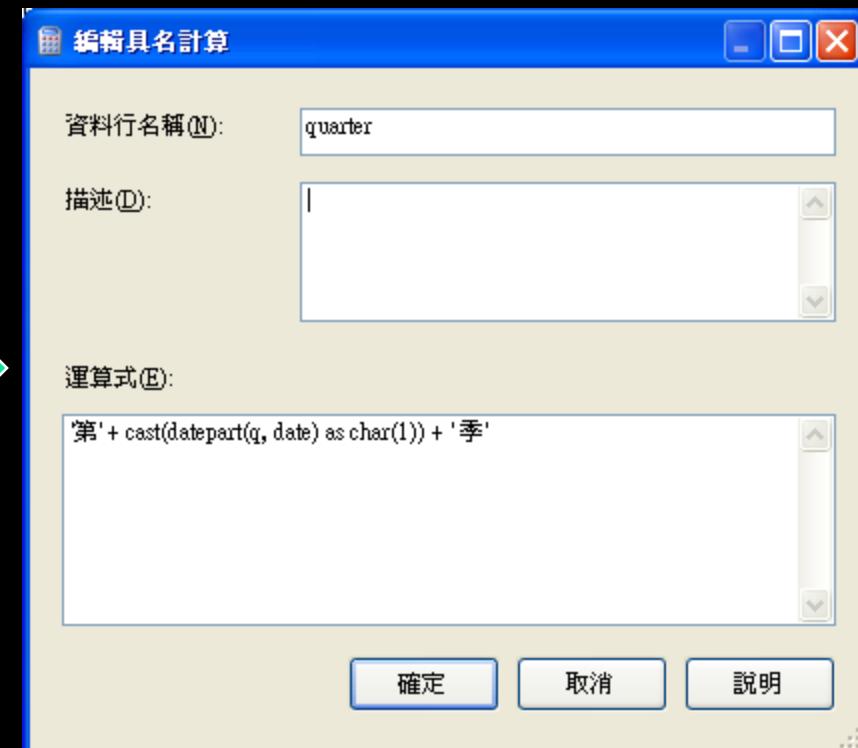


新增具名計算

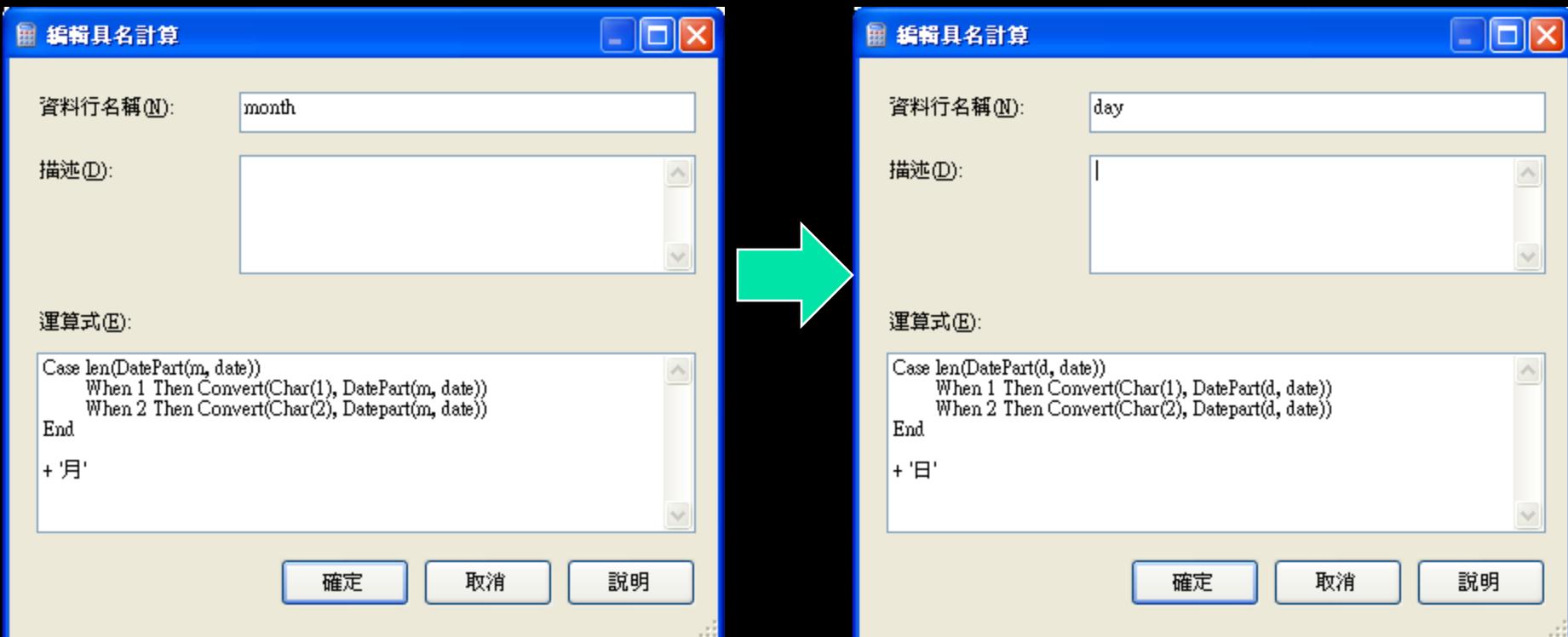
- 讓 Times_D 維度的結構完全掌控在我們自己



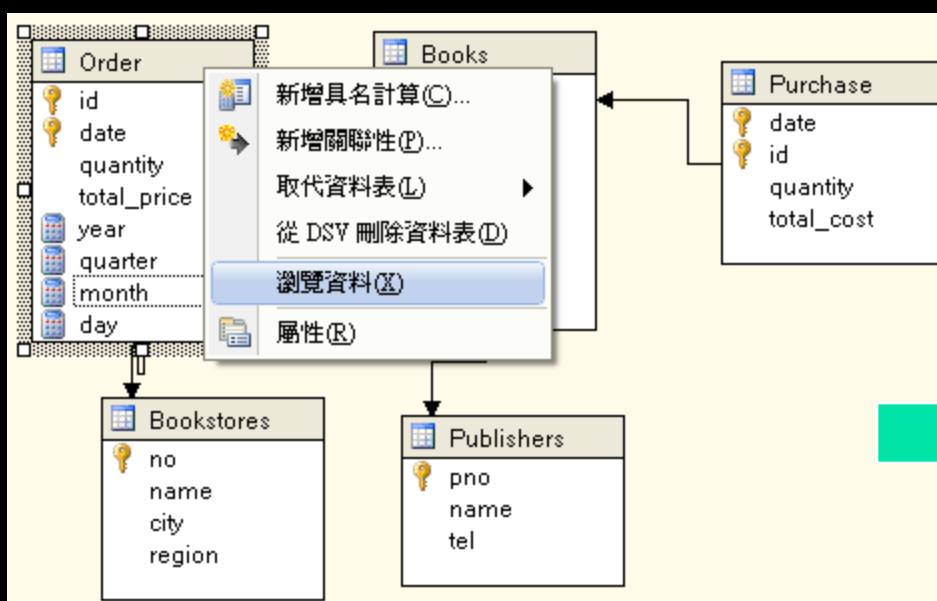
新增具名計算



新增具名計算



具名計算新增完成

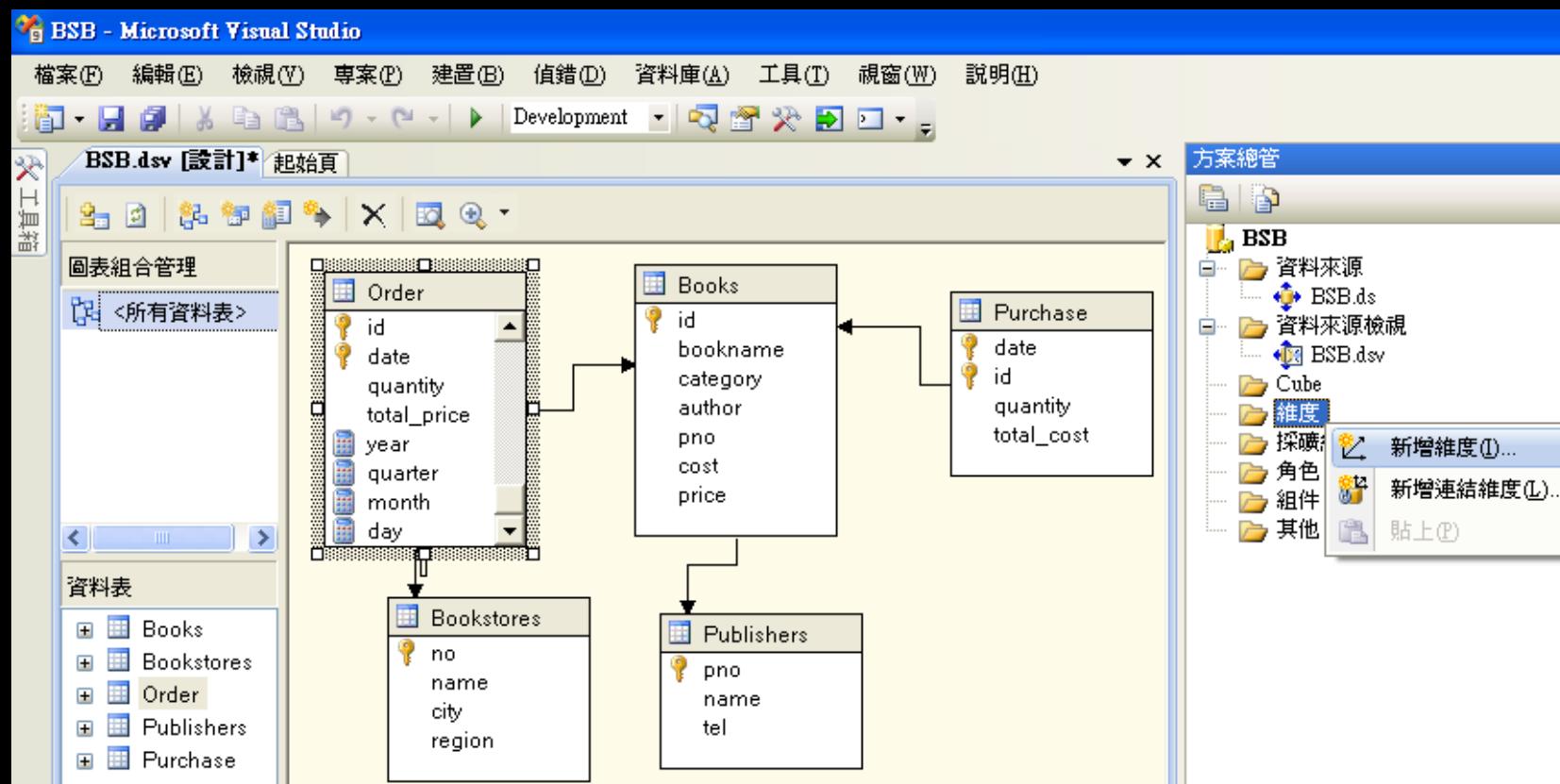


The screenshot shows the 'Order' table in the BSB.dsv design view. The table has columns: no, id, date, quantity, total_price, year, quarter, month, day. The data shows various purchase records with dates ranging from February 8, 1999, to December 21, 1999.

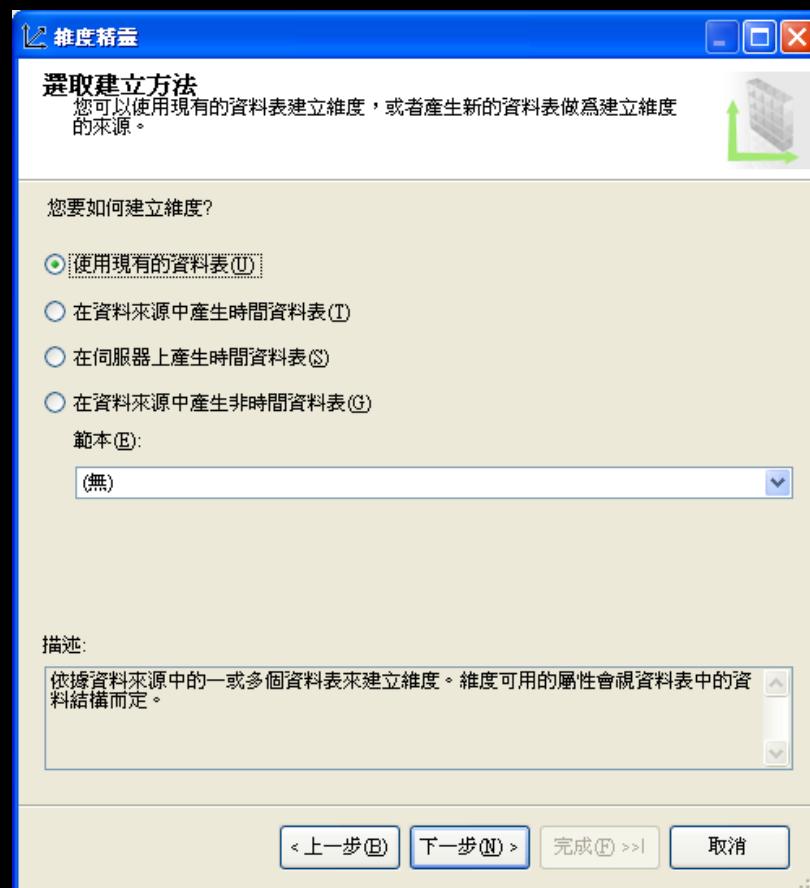
no	id	date	quantity	total_price	year	quarter	month	day
1	1	1999-02-08 00:00:00Z	30	3600	1999年	第1季	2月	8日
1	2	1999-03-12 00:00:00Z	20	3400	1999年	第1季	3月	12日
1	3	1999-05-14 00:00:00Z	40	6800	1999年	第2季	5月	14日
1	4	1999-07-15 00:00:00Z	20	2800	1999年	第3季	7月	15日
1	5	1999-08-17 00:00:00Z	10	1200	1999年	第3季	8月	17日
1	6	1999-09-20 00:00:00Z	10	1900	1999年	第3季	9月	20日
2	1	1999-01-09 00:00:00Z	30	3600	1999年	第1季	1月	9日
2	2	1999-04-12 00:00:00Z	40	6800	1999年	第2季	4月	12日
3	2	1999-07-15 00:00:00Z	20	3400	1999年	第3季	7月	15日
4	2	1999-03-18 00:00:00Z	20	3400	1999年	第1季	3月	18日
4	4	1999-10-20 00:00:00Z	30	4200	1999年	第4季	10月	20日
4	5	1999-12-21 00:00:00Z	40	4800	1999年	第4季	12月	21日

建立維度

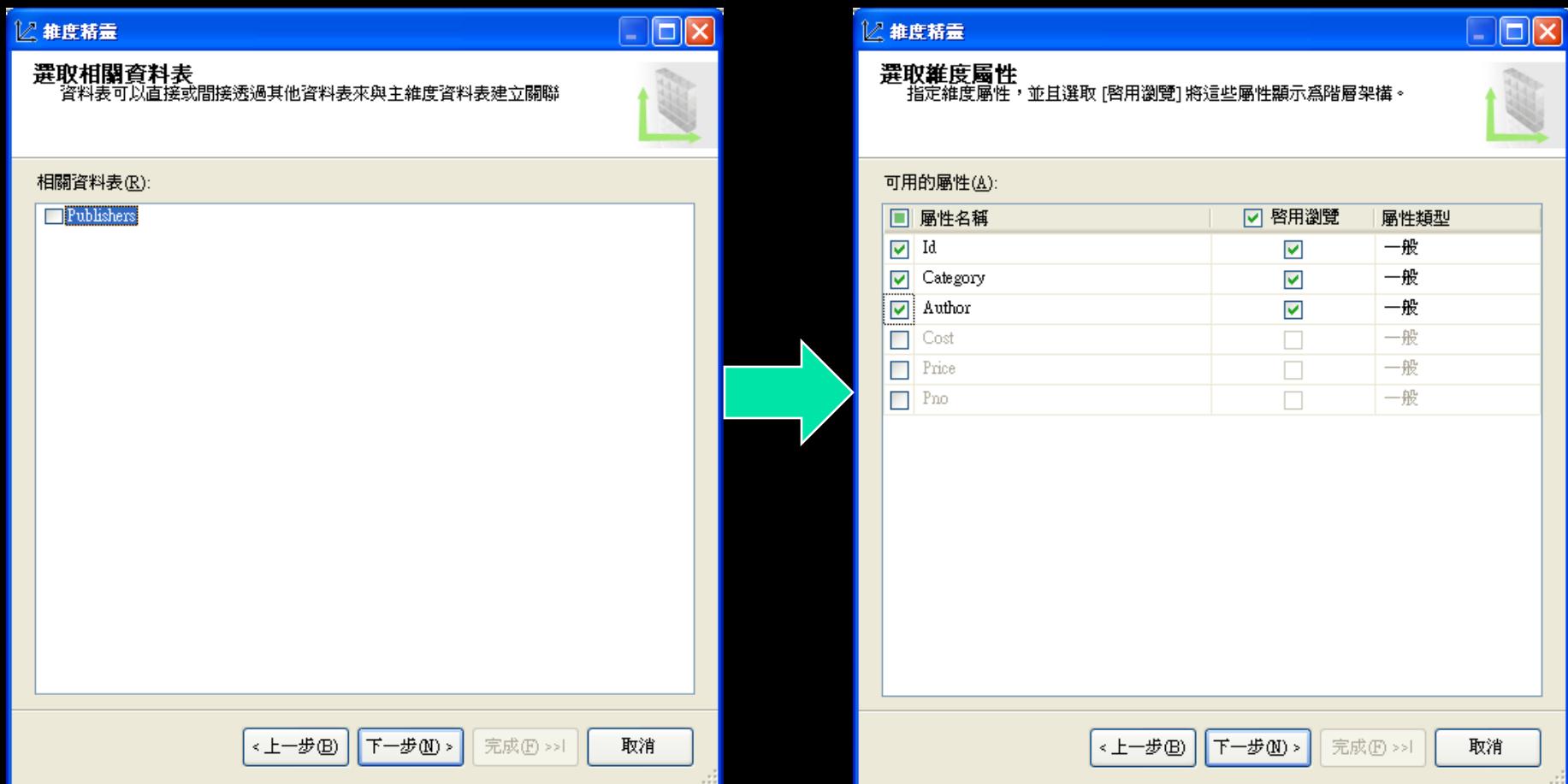
■ 建立 Books_D 維度



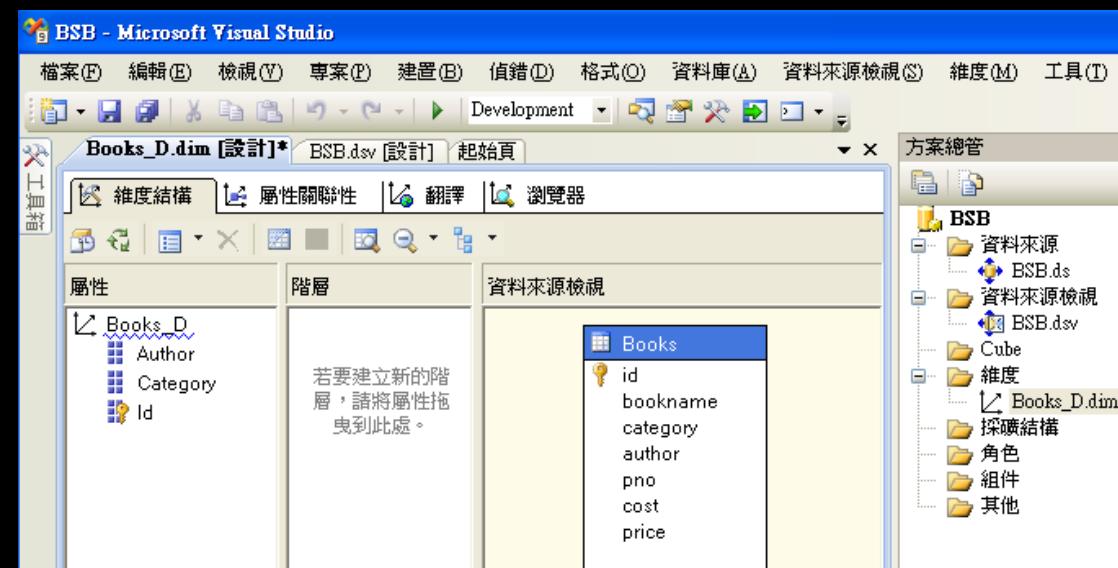
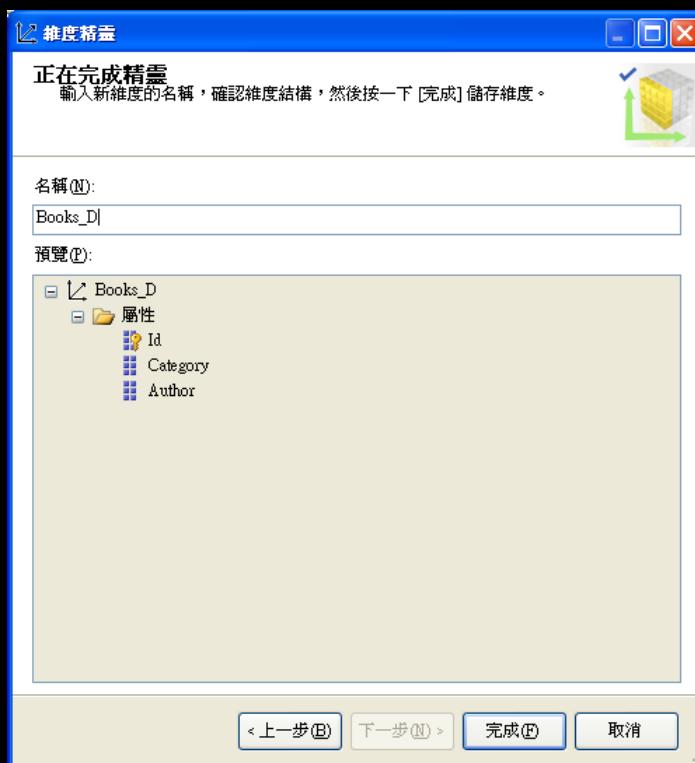
建立 Books_D 維度



建立 Books_D 維度



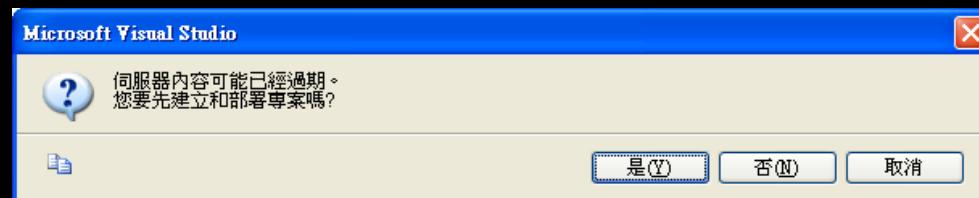
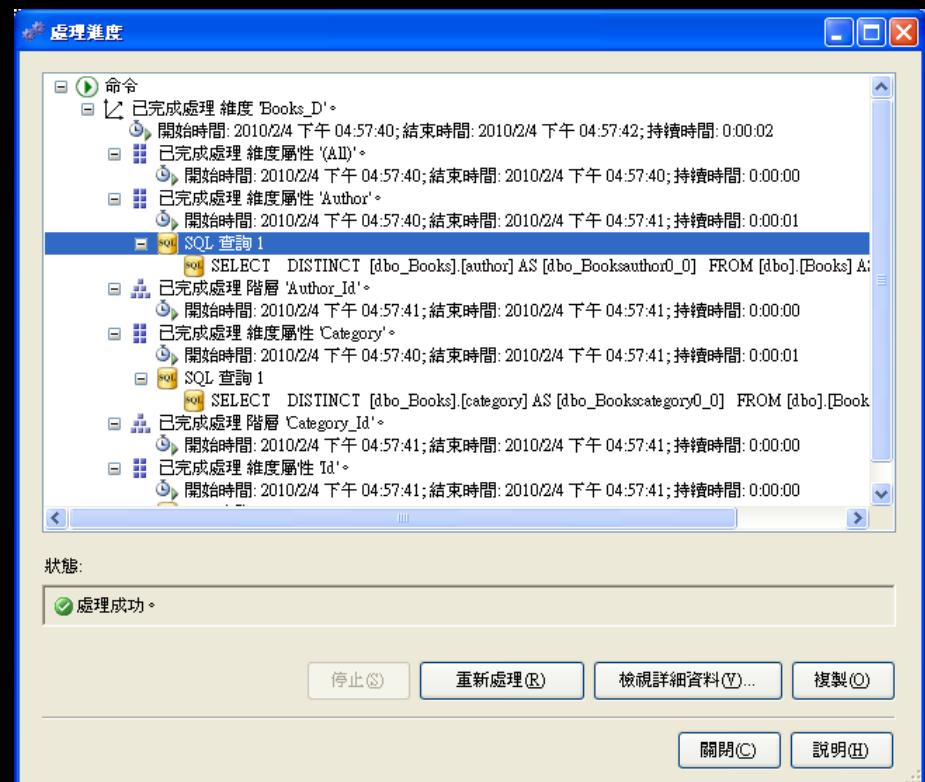
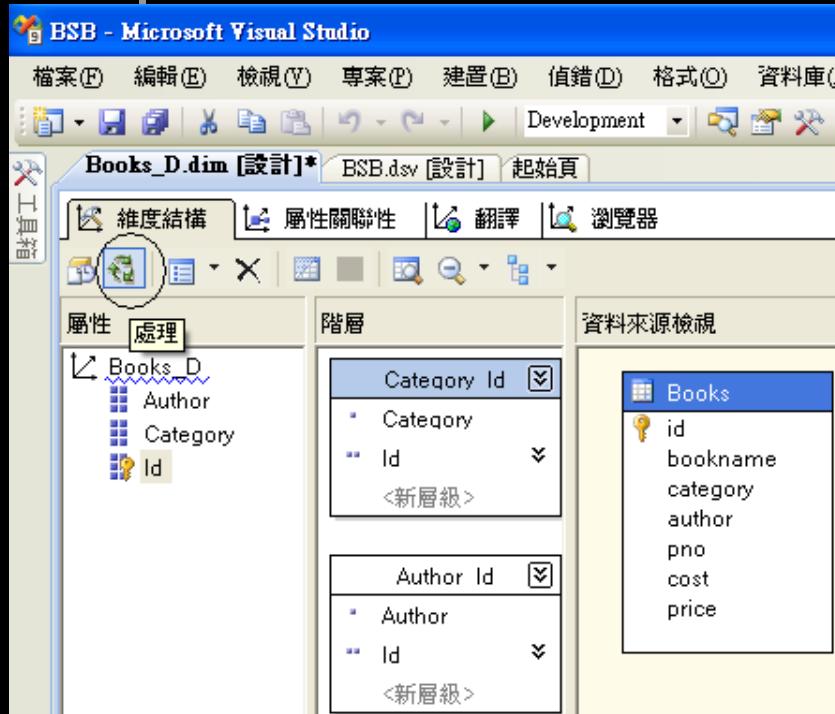
建立 Books_D 維度



建立階層關係

The screenshot shows the Microsoft Analysis Services (SSAS) Data Source Designer interface. The title bar reads "Books_D.dim [設計] * BSB.ds [設計] 起始頁". The left pane displays the "屬性" (Properties) and "階層" (Hierarchies) sections. In the "階層" section, there is a tree view under "Category" and a dropdown menu for "Id". A tooltip says: "若要建立新的階層，請將屬性拖曳到此處。" (To create a new hierarchy, drag the attribute here.) The right pane shows the "資料來源檢視" (Data Source View) with a table named "Books" containing columns: id, bookname, category, author, pno, cost, price. The bottom pane shows the "屬性" (Properties), "階層" (Hierarchies), and "資料來源檢視" (Data Source View) sections. A context menu is open over the "Category" hierarchy node, with options: 剪下 (Cut), 複製 (Copy), 貼上 (Paste), 刪除 (Delete), 重新命名 (Rename), and 屬性 (Properties). The "Rename" option is highlighted.

處理 Books_D 維度



Books_D 維度資料/階層切換

The screenshot displays the 'Books_D.dim [設計]' (Dimension Design) window in a software application. It features two main panes:

- Dimension Structure (維度結構) Pane:** Shows a tree view of dimensions:
 - 階層 (Hierarchy): Author_Id
 - 語言 (Language): Author_Id (selected), Category_Id, Author, Category, Id
 - 目前屬性 (Current Attribute): All, followed by a list of authors: 毛六, 王五, 吳承恩, 李四 (with sub-items: 英美文學讀本, 應用法語入門), 周韻寰, 施耐庵, 張三, 曹雪芹, 曾守正, 羅貫中, 龍道元.
- Hierarchy Browser (階層切換) Pane:** Shows a tree view of categories:
 - 階層 (Hierarchy): Category_Id (selected)
 - 語言 (Language): Category_Id (selected), Author_Id, Author, Category, Id
 - 目前屬性 (Current Attribute): All, followed by three category groups:
 - 文學類 (Literature): 三國演義, 水經注, 水許傳, 西遊記, 紅樓夢
 - 電腦類 (Computer): UNIX系統, Web架站實務, 資料庫系統
 - 語言類 (Language): 日文讀本, 英美文學讀本, 德語入門, 應用法語入門

Bookstores_D 與 Publishers_D 建立

- 步驟與前述類似，請參考書上畫面建立

The screenshot displays three windows from the Microsoft Analysis Services (SSAS) Management Studio:

- Dimension Designer (Left):** Shows the 'Books_D.dim [設計]' dimension. The 'Dimensions' pane lists 'RCN' as the selected dimension. The 'Hierarchy' pane shows a hierarchy structure under 'RCN' with levels: Region, No, City, and RCN. The 'Current Level' dropdown is set to 'All'. The 'Dimensions' tree view shows categories like All, 中 (Central), 北 (North), 東 (East), 南 (South), and their respective cities.
- Cube Designer (Middle):** Shows the 'BSB.dsv [設計]' cube. The 'Dimensions' pane lists 'Publisher_Book' as the selected dimension. The 'Hierarchy' pane shows a hierarchy structure under 'Publisher_Book' with levels: All, Publisher, and Book. The 'Current Level' dropdown is set to '預設值'. The 'Dimensions' tree view shows categories like All, 中庸出版社, 古文出版社, 易經出版社, 春秋出版社, and 聊齋出版社.
- Data Source Designer (Right):** Shows the 'BSB' data source. The 'Data Sources' pane lists 'BSB' as the selected data source. The 'Dimensions' pane shows a list of dimensions: Books_D.dim, Bookstores_D.dim, Publishers_D.dim, 探礦結構, 角色, 組件, and 其他. The 'Connections' pane shows connection details for 'BSB' to 'localhost' on 'BSB' with 'Deployment Progress - BSB' status.

Times_D 建立方式也請參考書本

系統發現此階層並非越下面越多成員
所以提出一個警告，
原因是來源資料庫內容筆數太少所導致，
在此可以不理它。

建立 Cube

The screenshot shows the Microsoft Analysis Services Cube Wizard interface, divided into two main windows:

方案總管 (Solution Explorer) on the left:

- 显示项目名称: BSB
- 文件夹结构:
 - 資料來源: BSB.ds
 - 資料來源檢視: BSB.dsv
 - Cube**: 新增 Cube(C)... (当前选中)
 - 維度: B (已粘贴), B, Publishers_D.dim, Times_D.dim
 - 採礦結構
 - 角色
 - 組件
 - 其他

Step 1: 選取建立方法 (Select Build Method) - 左侧窗口:

说明: 您可以用現有資料表做為 Cube 的依據、建立空白 Cube，或在資料來源中產生資料表。

您要如何建立 Cube?

使用現有的資料表(1)

建立空白 Cube(C)

在資料來源中建立資料表(G)

範本(I): (無)

描述:

依據資料來源中的一或多個資料表建立 Cube。

按钮: <上一步(B) | 下一步(N) > | 完成(F) >> | 取消

Step 2: 選取量值群組資料表 (Select Value Group Data Table) - 右侧窗口:

選取資料來源檢視或圖表，然後選取要使用於量值群組的資料表。

資料來源檢視(D): BSB

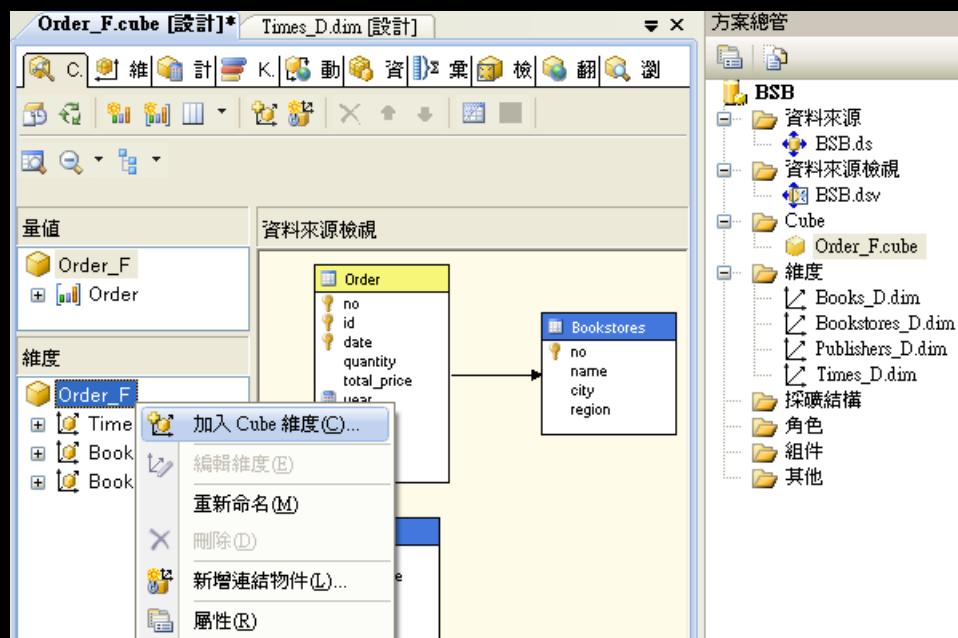
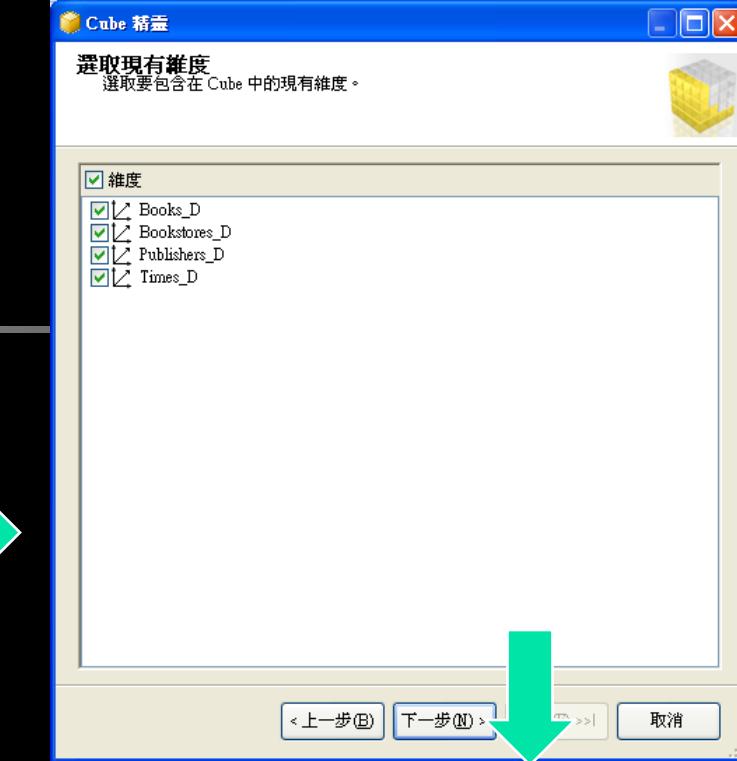
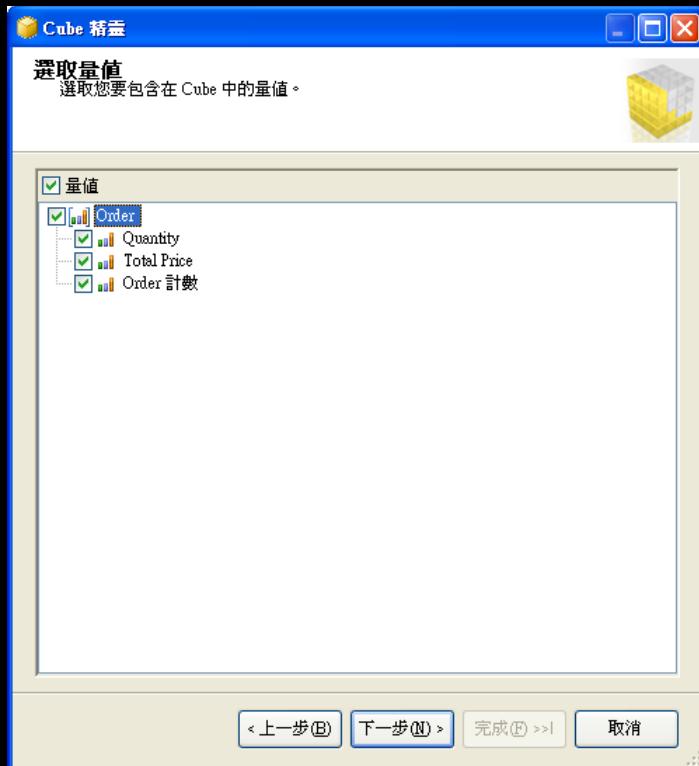
量值群組資料表(M):

Books
Bookstores
 Order
Publishers
Purchase

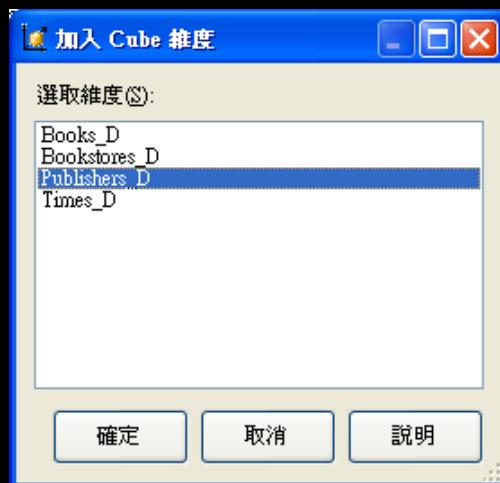
建議(S)

按钮: <上一步(B) | 下一步(N) > | 完成(F) >> | 取消

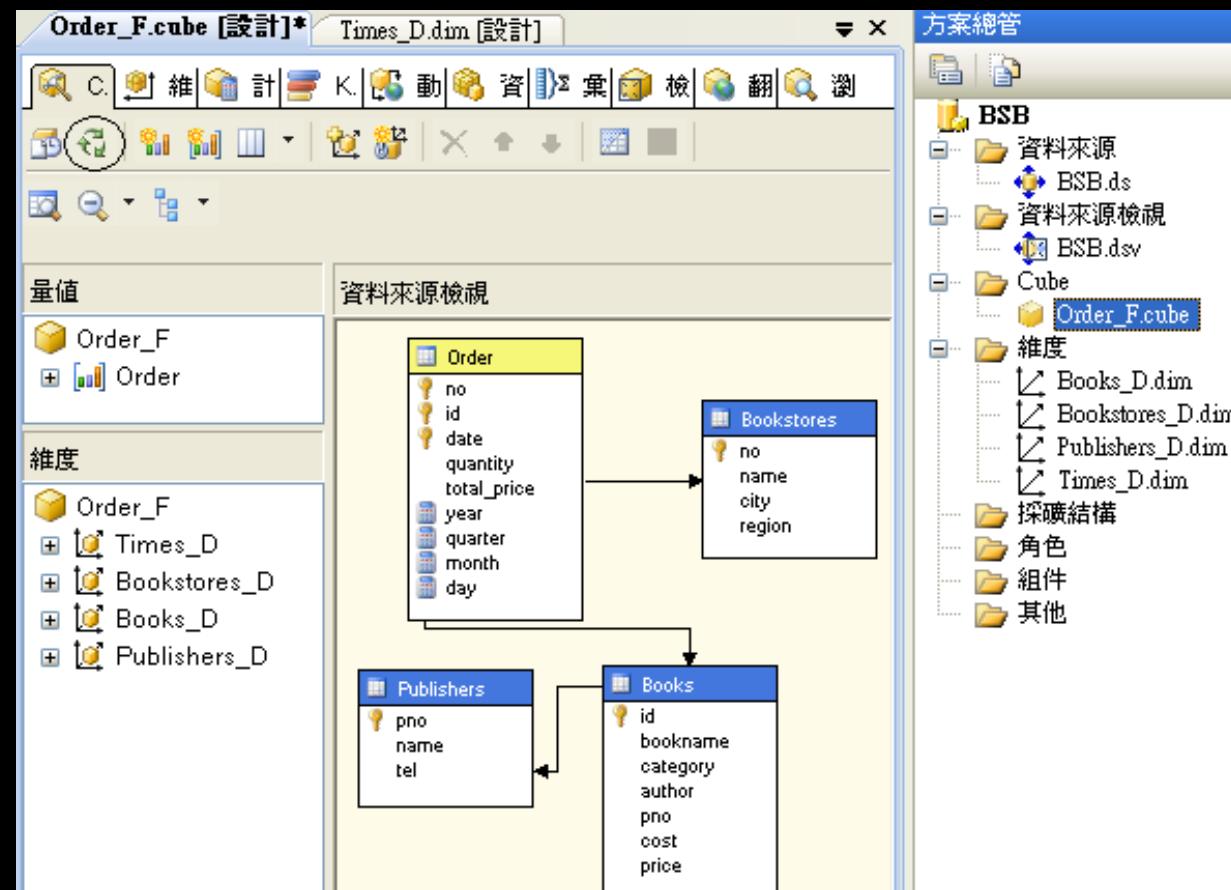
建立 Cube



建立 Cube



將漏掉的
Publishers_D 加回來



Cube 建立完成

量值群組: <全部>

維度	階層	運算子
<選取維度>		

請將篩選欄位拖曳到這裡
將欄位拖曳到這裡

方案總管

- BSB
 - 資料來源
 - BSB.ds
 - 資料來源檢視
 - BSB.dsv
 - Cube
 - Order_Fcube
 - 維度
 - Books_D.dim
 - Bookstores_D.dim
 - Publishers_D.dim
 - Times_D.dim
 - 探礦結構
 - 角色
 - 組件

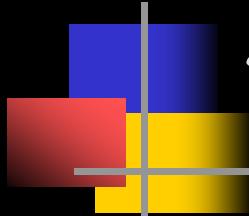
量值群組: <全部>

維度	階層	運算子	篩選運算式
<選取維度>			

請將篩選欄位拖曳到這裡

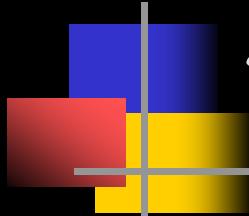
Region		中	北	東	南	總計				
Author	Quantity	Total Price	Quantity	Total Price	Quantity	Total Price				
毛六				10	2400	10	2400			
王五				10	3200	20	6400			
吳承恩	30	4200	20	2800			50	9600		
李四				20	5300	10	2800	30	8100	
周韻寰			10	1900		30	5700	40	7600	
施耐庵	20	3400	40	6800		40	6800	100	17000	
張三						60	15000	60	15000	
曹雪芹			40	6800				40	6800	
曾守正						40	12000	40	12000	
羅貫中			30	3600		50	6000	80	9600	
關曉彤	40	4800	10	1200				50	6000	
總計	90	12400	150	23100	40	10900	250	54700	530	101100

使用Drag-and-Drop 各維度
與Measures 就可以得到



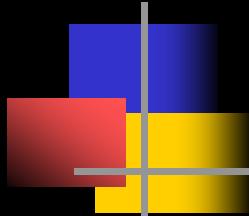
使用 MDX 摘取資料

- Multi-Dimensional eXpression
- 維持 SQL 的語法架構
- 傳統 SQL 語法查詢結果是不對稱 (Asymmetric) 的表格 (由 n 個單一維度組成)：
 - 橫向結構由固定屬性所定義，並不是無限延伸；
 - 直向結構會隨著資料量的多寡而變動，可以無限延伸的。
 - 很難對查詢結果做「轉置」(Transpose) 動作，也就是將列變成行、行變成列來看資料。



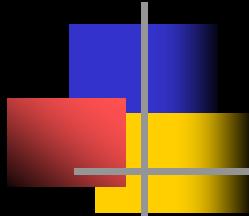
使用 MDX 摘取資料

- 對資料方塊 (Cubes) 做查詢，是以多「軸」 (Axes) 的方式來描述。
- 在查詢中指明分類資料的維度，放在那個軸上。
- 系統都可以將每個維度的資料變成各個軸上的標題 (類似屬性)，所以這些軸是完全對稱的，這些標題會隨資料的多寡而變動，不會像關聯表一樣都具有相同的固定行。
- 可以任意「轉置」 (Transpose) ，隨時可以將列變成行、行變成列來看待資料。



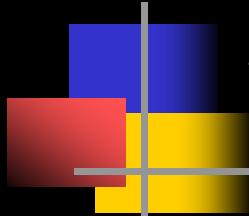
MDX 語法

- ```
SELECT column_set0 ON AXIS(0) [, column_set1 on
AXIS(1),...]
FROM cube_name
[WHERE (member-of-dim0, member-of-dim1,..., member-of-
dimn)]
```
- ```
SELECT column_set0 ON COLUMNS, column_set1 on  
ROWS  
FROM cube  
[WHERE (member-of-dim0, member-of-dim1,..., member-of-  
dimn) ]
```



前五個維度的名稱

- 不管你的維度有多少個，前五個維度 (`axis(0)`, `axis(1)`, ..., `axis(4)`) 也分別稱為
 - COLUMNS,
 - ROWS,
 - PAGES,
 - CHAPTERS,
 - SECTIONS ,
- 就好像書本的安排順序一般。順序最好不要弄反，否則可能會出現錯誤的情況。



指定階層屬性

- 如果只要指定維度表中的某一個階層 (Level) 的屬性時，則使用：

<dimension_name>.<level_name>.MEMBERS

- 如果要指定所有的維度成員，則使用：

<dimension_name>.<key>.MEMBERS

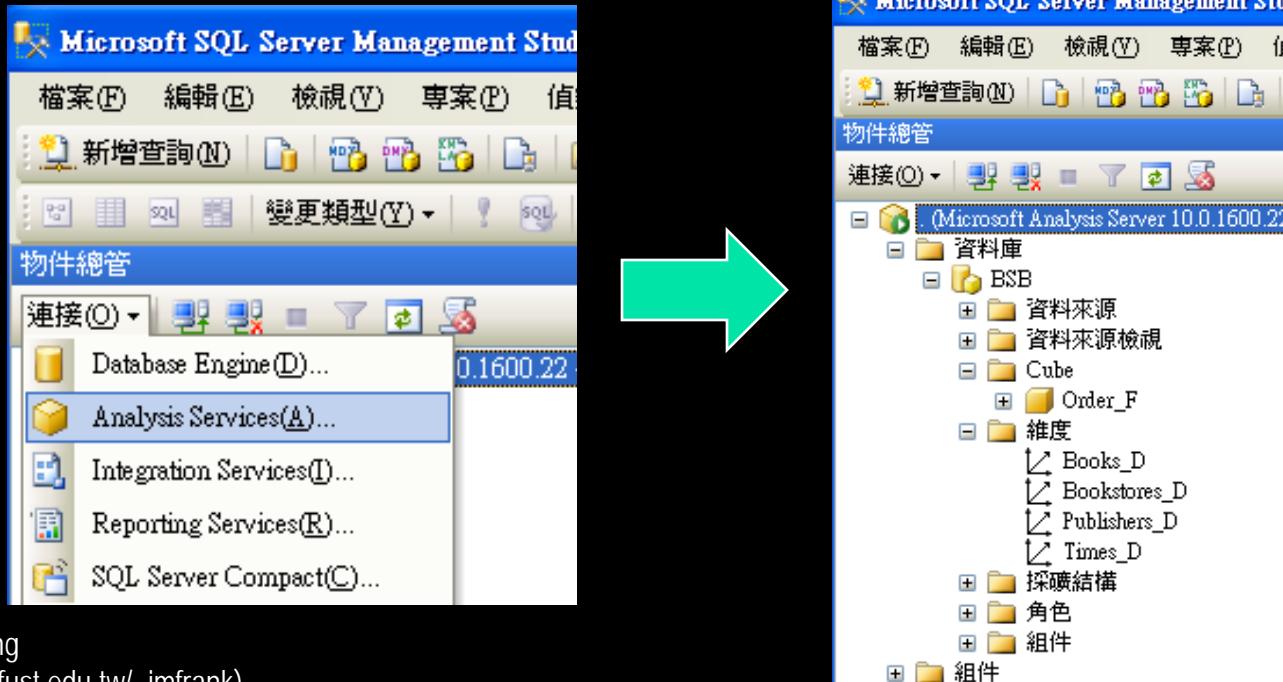
<key> 是指以主鍵所建立的階層

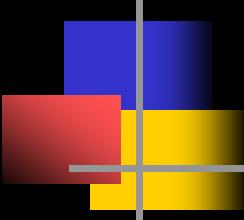
E.g. Books_D.[Id].members 表示 Books_D 所有成員：

{Books_D.[文學類], ..., Books_D.[語言類], Books_D.[三國演義], ..., Books_D.[應用法語入門]}

MDX 查詢介面

- MDX 查詢介面目前已經整合到 SQL Server Management Studio 中了
- 只要連到 Analysis Services 即可





範例說明

- select Bookstores_D.[Region].members on columns,
Books_D.[Id].members on rows
from Order_F

The screenshot shows the Microsoft SQL Server Management Studio interface. The title bar reads "Microsoft SQL Server Management Studio". The menu bar includes "檔案(F)", "編輯(E)", "檢視(V)", "查詢(Q)", "專案(P)", "值錯(D)", "工具(T)", "視窗(W)", "社群(C)", and "說明(H)". The toolbar contains icons for "新增查詢(N)", "執行(X)", "取消(C)", "重做(R)", "剪切(C)", "複製(C)", "貼上(P)", "刪除(D)", "插入(I)", "縮放(S)", and "放大(L)".

The left pane displays the "物件總管" (Object Explorer) with a tree structure:

- 連接(O) - Microsoft Analysis Server 10.
- 資料庫 - BSB
 - 中繼資料
 - 資料來源
 - 資料來源檢視
 - Cube
 - Order_F
 - 維度
 - Books_D
 - Bookstores_D
 - Publishers_D
 - Times_D
 - 探礦結構
 - 角色
 - 組件
- 組件

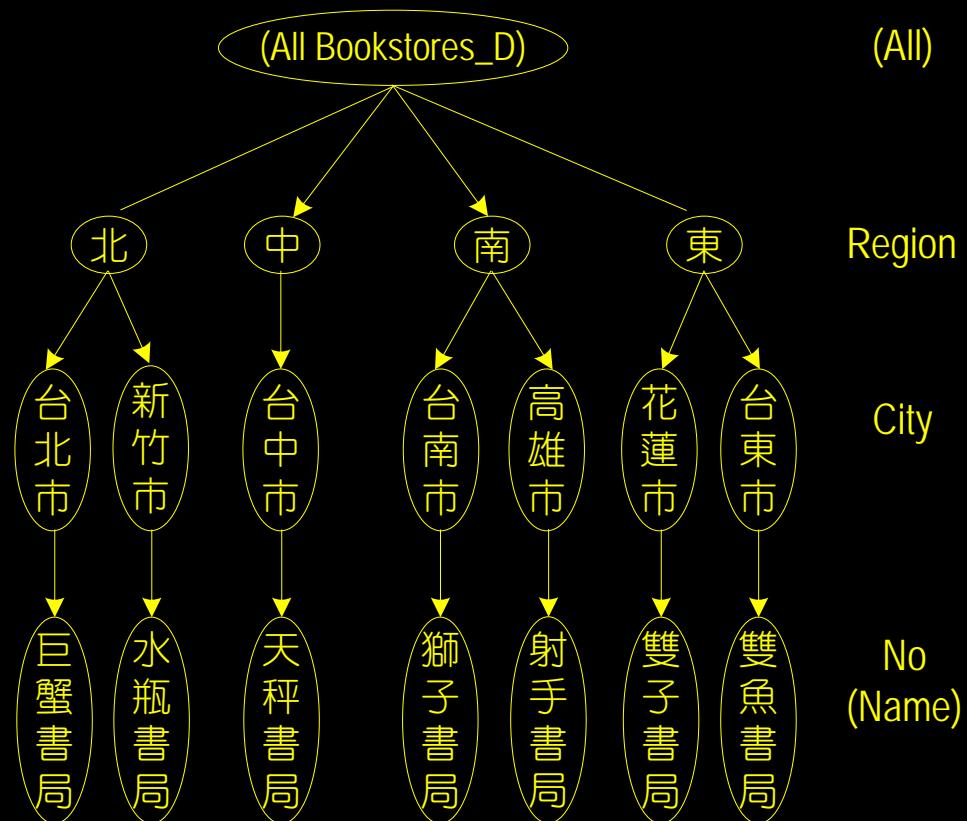
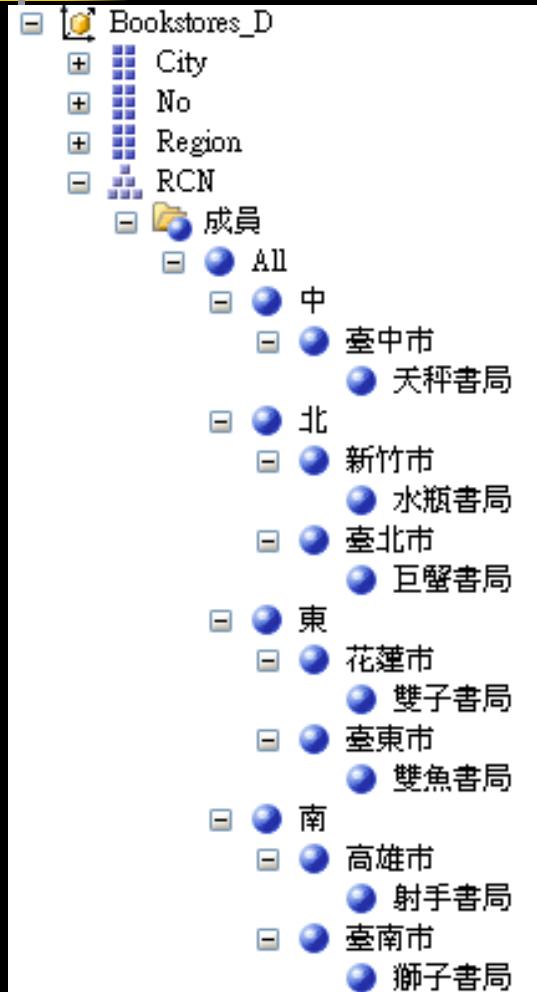
The central pane shows the MDX query results for "MDXQuery1.mdx - ...ISCHAIR\imfrank*". The query is:

```
select [Bookstores_D].[Region].members on column  
      ,[Books_D].[Id].members on rows  
from Order_F
```

The results are displayed in a grid:

	All	中	北	東	南
All	530	90	150	40	250
UNIX系統	40	(Null)	10	(Null)	30
Web架站實務	60	(Null)	(Null)	(Null)	60
三國演義	80	(Null)	30	(Null)	50
日文讀本	30	(Null)	(Null)	10	20
水經注	50	40	10	(Null)	(Null)
水滸傳	100	20	40	(Null)	40
西遊記	50	30	20	(Null)	(Null)
紅樓夢	40	(Null)	40	(Null)	(Null)
英美文學讀本	10	(Null)	(Null)	10	(Null)
資料庫系統	40	(Null)	(Null)	(Null)	40
德語入門	10	(Null)	(Null)	10	(Null)
應用法語入門	20	(Null)	(Null)	10	10

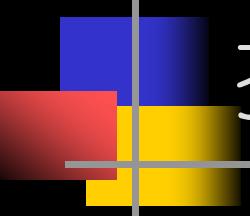
維度可以看成是一個樹狀結構



Region

City

No
(Name)

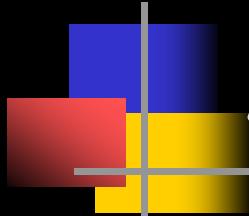


指定某個層級成員的 Children 範例

- select Bookstores_D.[南].**children** on rows,
Books_D.[Id].members on columns
from Order_F



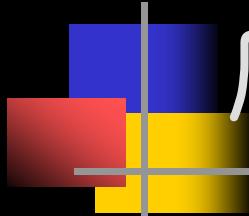
	高雄市	台南市
All	70	180
UNIX系統		30
Web架站實務		60
三國演義	30	20
日文讀本		20
水經注		
水滸傳	40	
西遊記		
紅樓夢		
英美文學讀本		
資料庫系統		40
德語入門		
應用法語入門		10



產生時間軸的範例

```
select Times_D.month.members on columns,  
      Bookstores_D.region.members on rows  
from Order_F
```

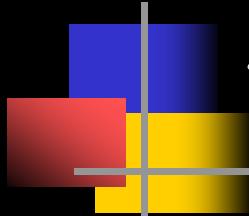
	All	1月	2月	3月	4月	5月	6月	8月	9月	10月	12月
All	530	30	60	50	50	40	60	50	40	30	120
中	90			20						30	40
北	150		30	20		40	40	10	10		
東	40		10	10	10						10
南	250	30	20		40		20	40	30		70



產生時間軸的範例

將前面指令的 region 改成 city：

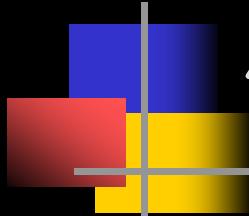
	All	1月	2月	3月	4月	5月	6月	8月	9月	10月	12月
All	530	30	60	50	50	40	60	50	40	30	120
花蓮市											
高雄市	70	30			40						
新竹市	20						20				
臺中市	90			20					30	40	
臺北市	130		30	20		40	20	10	10		
臺東市	40		10	10	10						10
臺南市	180		20				20	40	30		70



在資料軸上顯示特定成員

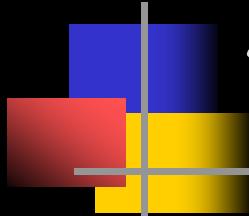
```
select {Books_D.[三國演義], Books_D.[資料庫系統]} on columns,  
Bookstores_D.region.members on rows  
from Order_F
```

	三國演義	資料庫系統
中		
北	60	
東		
南	100	80



使用完整成員名稱避免混淆

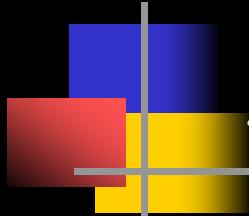
- [Times_D].[Month].&[3 月] 會混淆，因為每一年都有 [3 月]
- [Times_D].[date].[Year].&[1999 年].&[第1季].&[3 月] 才不會造成混淆
- 可以直接在查詢畫面中拖曳成員，系統會自動產生完整的名稱



使用 WHERE 子句

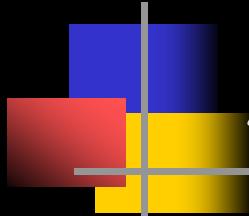
- Select {[Books_D].[三國演義], [Books_D].[資料庫系統]} on columns,
Bookstores_D.region.members on rows
From Order_F
where [Times_D].[Month].&[1 月]

	三國演義	資料庫系統
中		
北		
東		
南	30	



用 WHERE 查詢另一個觀察值

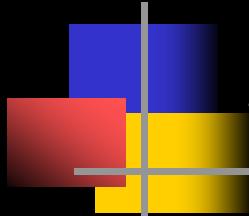
- 觀察值 (Measures) 與維度 (Dimensions) 在 MDX 中的地位是完全一樣的
- 任何可以出現 Dimension 的地方幾乎都可以出現 Measures。
- 要看另一個觀察值的話，就可以將該觀察值寫在 WHERE 子句中
- 見下頁範例...



用 Where 子句查詢其他觀察值

```
select {Books_D.[三國演義], Books_D.[資料庫系統]} on columns,  
Bookstores_D.region.members on rows  
from Order_F  
where Measures.[Total Price]
```

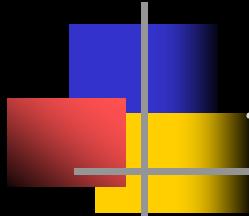
	三國演義	資料庫系統
中		
北	3,600	
東		
南	6,000	12,000



三維查詢

```
Select {Books_D.[三國演義], Books_D.[資料庫系統]} on axis(0),  
    Bookstores_D.region.members on axis(1),  
    {[Times_D].[Month].&[1 月], [Times_D].[Month].&[2 月]} on axis(2)  
from Order_F
```

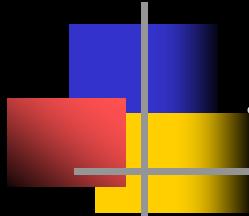
- 結果應該像圖 8.76 (如下頁) 一樣，是三維的表格。
但是 SQL Server Management Studio 不支援顯示三個維度以上的查詢結果



三維查詢應該顯示的結果

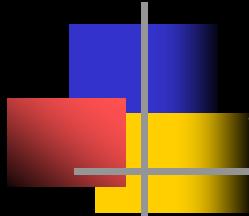
		三國演義	資料庫系統
中	January		
	February		
北	January		
	February	30	
東	January		
	February		
南	January	30	
	February	20	

無法顯示, 沒關係! 山不轉, 路轉! 路不轉, 水彎!
只要換成下頁的方式即可如願



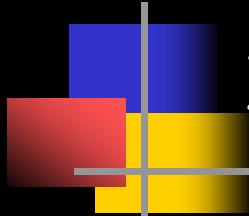
三維查詢替代作法

```
select {Books_D.[三國演義], Books_D.[資料庫系統]} on columns,  
{ (Bookstores_D.region.[中], [Times_D].[Month].&[1 月]),  
  (Bookstores_D.region.[中], [Times_D].[Month].&[2 月]),  
  (Bookstores_D.region.[北], [Times_D].[Month].&[1 月]),  
  (Bookstores_D.region.[北], [Times_D].[Month].&[2 月]),  
  (Bookstores_D.region.[東], [Times_D].[Month].&[1 月]),  
  (Bookstores_D.region.[東], [Times_D].[Month].&[2 月]),  
  (Bookstores_D.region.[南], [Times_D].[Month].&[1 月]),  
  (Bookstores_D.region.[南], [Times_D].[Month].&[2 月]) }  
on rows  
from Order_F
```



更簡化的作法

- 透過函數 Crossjoin(<Set1>, <Set2>) 完成。
- select {Books_D.[三國演義], Books_D.[資料庫系統]} on columns,
Crossjoin({Bookstores_D.region.[中], Bookstores_D.region.[北],
Bookstores_D.region.[東], Bookstores_D.region.[南]},
{Times_D.[month].[1 月], Times_D.[month].[2 月]}) on rows
from Order_F
- 或以 Bookstores_D.region.members 來取得 <Set1> 的資料：
- select {Books_D.[三國演義], Books_D.[資料庫系統]} on columns,
Crossjoin(Bookstores_D.region.members, {Times_D.[month].[1 月],
Times_D.[month].[2 月]}) on rows
from Order_F



利用Where 子句來完成

- 要分兩次，所以還要另外整合

1. select {Books_D.[三國演義], Books_D.[資料庫系統]} on columns,
Bookstores_D.region.members on rows

from Order_F

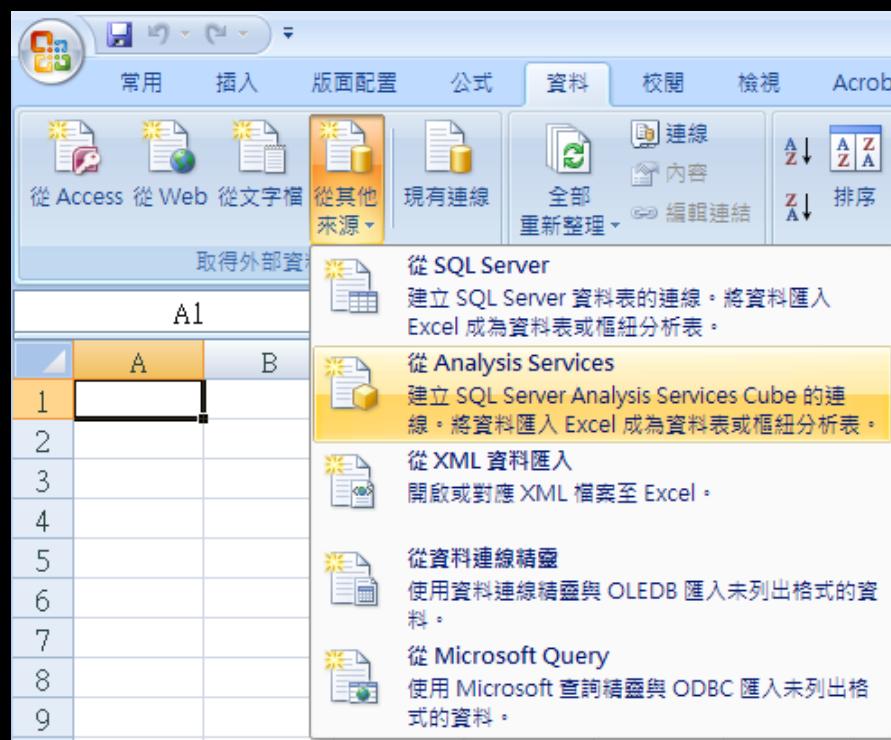
where Times_D.[month].[1 月]

2. Select {Books_D.[三國演義], Books_D.[資料庫系統]} on columns,
Bookstores_D.region.members on rows

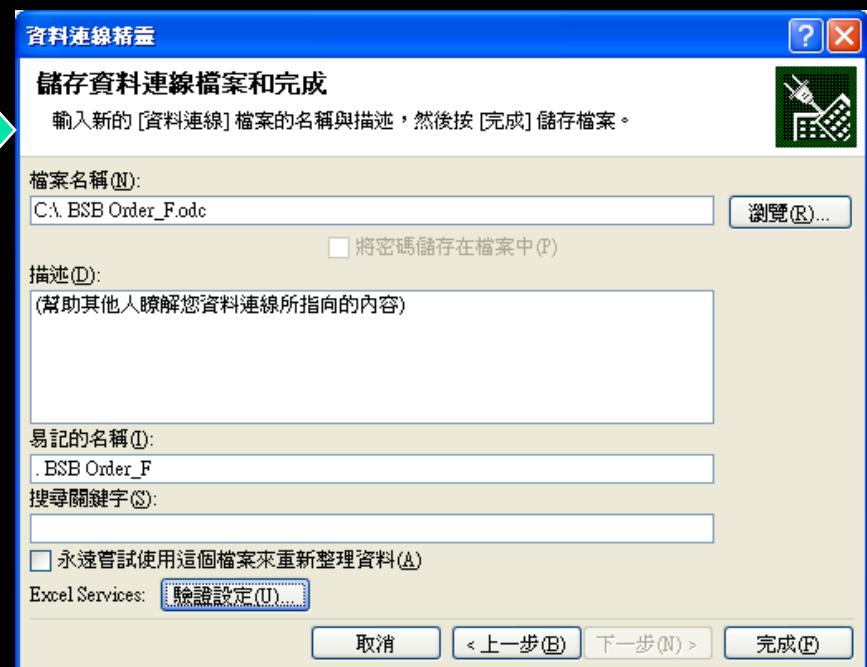
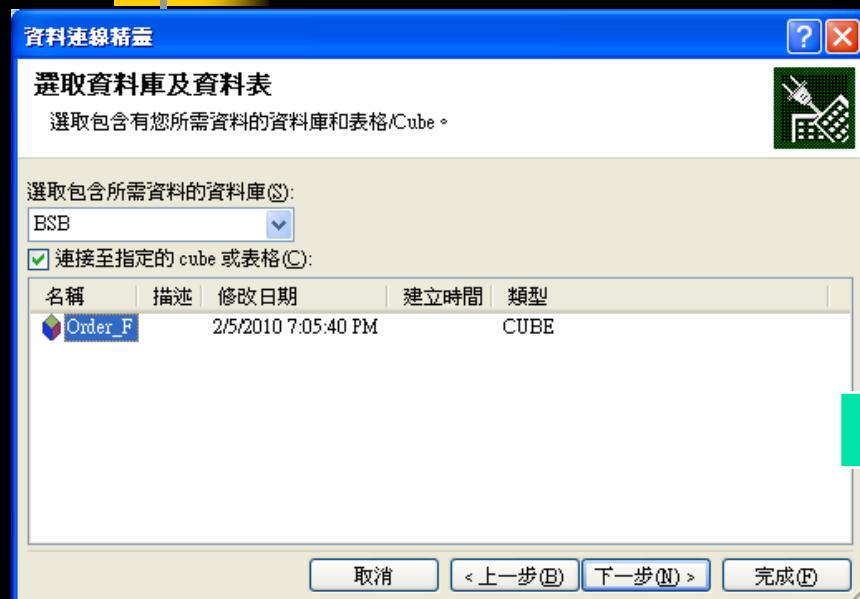
from Order_F

where Times_D.[month].[2 月]

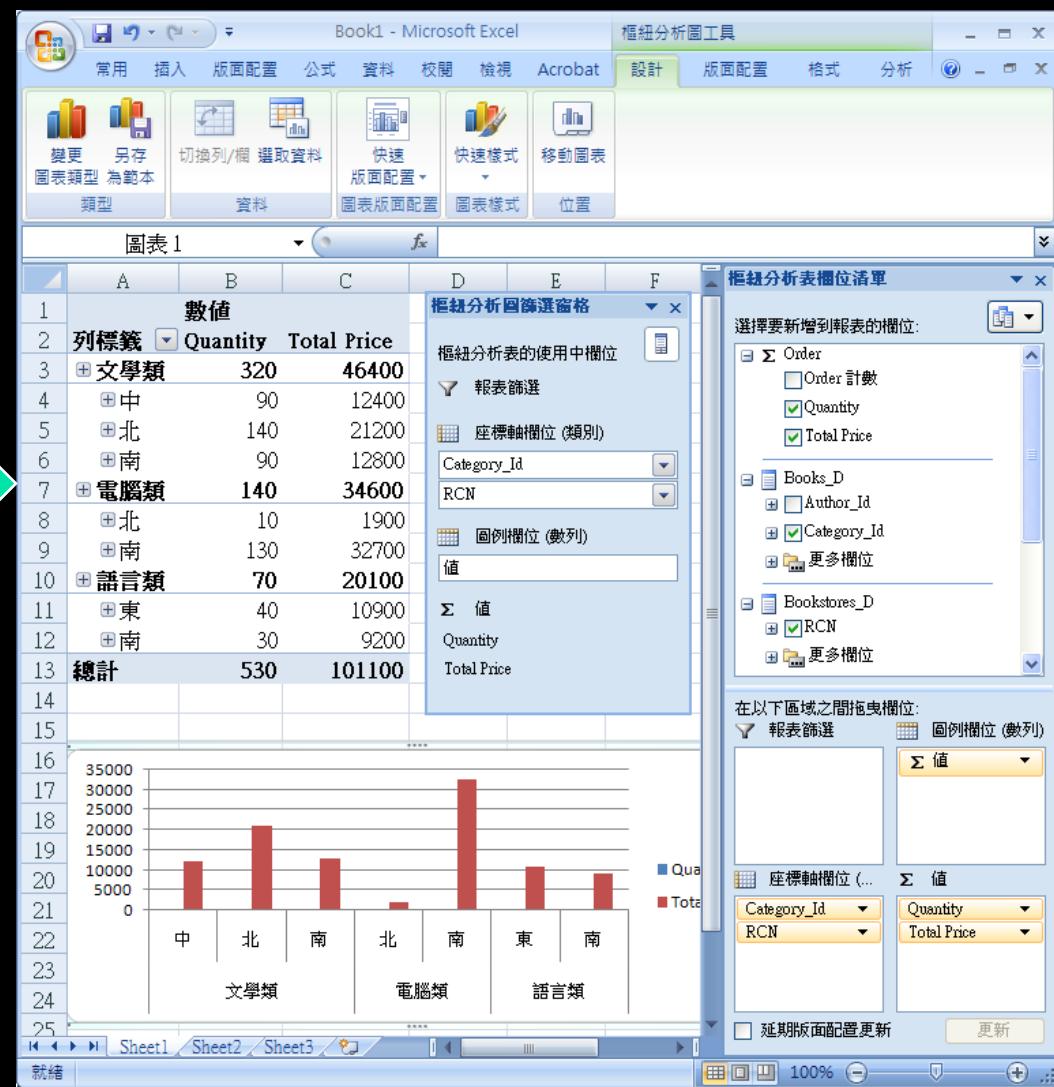
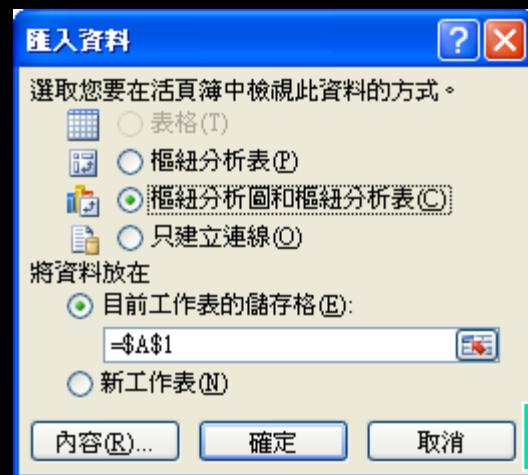
以 Excel 顯示 Cube 樞紐分析表

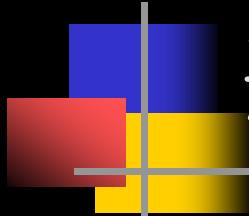


以 Excel 顯示 Cube 樞紐分析表



以 Excel 顯示 Cube 樞紐分析表



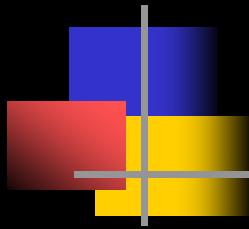


維度/資料方塊交互參考對照表

- 一般企業在資料倉儲系統的建置上，會有
 - 數十個維度，
 - 十多個甚至數十個資料方塊
- 所以在整個建置過程的報告撰寫上也格外重要
- 往往會藉助所謂的「維度/資料方塊交互參考對照表」(Dimension/Cube Cross-Reference Table) 來幫助記錄哪些維度用在哪些資料方塊上
- 能夠一目了然地看出：改變一個維度的結構後，會有多少資料方塊受到影響。

維度/資料方塊交互參考對照表

維度 (Dimension)	分公司	分公司 - 帳號	分公司 - 營業員	市場別	交易方式	交易年 - 月 - 日	交易金額級距	年齡層級距	券商 - 營業據點	券商區或	性別	縣市 - 鄉鎮	投資人身分	類股 - 登證券
資料方塊 (Cube)														
證券客戶結構分析				√				√			√	√	√	
分公司業務狀況分析				√	√	√						√		
市場總成交金額分析				√		√						√		
營業據點市場達成率分析				√		√			√					
券商營業據點業績分析						√	√	√						
歷史業績分析	√			√		√							√	
類股成交金額比重分析						√								√
融資融券維持率分佈	√	√			√	√								
市場融資融券分析				√		√								
客戶交易方式分析	√				√	√	√				√			
交易種類業績分析			√		√	√								√
投資類股比重分析	√				√	√								√



本章結束
The End.