<<會計資訊系統課程講義>>

關聯式資料庫與REA概念資料模式
(RDB & REA)

周國華
國立屏東大學會計學系

初版：2007/7/24
本次修訂：2023/2/20

智慧財產權聲明

- 本文件係由周國華老師獨自撰寫，除引用之概念屬於原文作者外，其餘文字及圖形內容之智慧財產權當然屬於周老師獨有。
- 任何機構或個人，在未取得周老師同意前，不得直接以本文件做為學校、研究機構、企業、會計師事務所、政府機關或財團法人機構舉辦教學或進修課程之教材，否則即屬侵權行為。
- 任何機構或個人，在未取得周老師同意前，不得在自行編撰的教材中直接大量引用本文件的內容。若屬單頁內部分內容之引用，亦請註明出處。

第一部份

關聯式資料庫(RDB) 基本概念

RDBMS

- RDBMS (Relational Database Management System)：關聯式資料庫管理系統。使用者根據RDB model建立資料庫。
- RDB model係由E. F. Codd在1970年創建。資料儲存在二維表格中，每份表格有一欄為**主索引**(primary key)，兩份表格藉由**外來鍵**(foreign key)建立彼此間的關聯。
  - **主索引**的值可唯一識別資料表的特定資料列，因此不可重複。主索引亦可由兩個或多個欄位共同組成，稱為**複合主索引**(composite primary key)。
  - 當資料表的某個欄位值，是另一個資料表的主索引值時，該欄位稱為**外來鍵**。表格之間藉由外來鍵建立關聯性，而不由儲存表格的實體位置來代表。
- RDBMS使用SQL語言建立、修改、移除資料庫**物件**及新增、刪除、修改**資料**。

RDBMS產品

- 根據db-engines.com的資料，2023/2資料庫軟體的市場排名如下：

Rank			DBMS	Database Model	Score		
Feb 2023	Jan 2023	Feb 2022			Feb 2023	Jan 2023	Feb 2022
1.	1.	1.	Oracle	Relational, Multi-model	1247.52	+2.35	-9.31
2.	2.	2.	MySQL	Relational, Multi-model	1195.45	-16.51	-19.23
3.	3.	3.	Microsoft SQL Server	Relational, Multi-model	929.09	+9.70	-19.96
4.	4.	4.	PostgreSQL	Relational, Multi-model	616.50	+1.65	+7.12
5.	5.	5.	MongoDB	Document, Multi-model	452.77	-2.42	-35.88
6.	6.	6.	Redis	Key-value, Multi-model	173.83	-3.72	-1.96
7.	7.	7.	IBM Db2	Relational, Multi-model	142.97	-0.60	-19.91
8.	8.	8.	Elasticsearch	Search engine, Multi-model	138.60	-2.56	-23.70
9.	↑ 10.	↑ 10.	SQLite	Relational	132.67	+1.17	+4.30
10.	↓ 9.	↓ 9.	Microsoft Access	Relational	131.03	-2.33	-0.23
11.	↑ 12.	11.	Cassandra	Wide column	116.22	-0.09	-7.76
12.	↓ 11.	↑ 15.	Snowflake	Relational	115.65	-1.60	+32.47
13.	13.	↓ 12.	MariaDB	Relational, Multi-model	96.81	-2.55	-10.30
14.	14.	↓ 13.	Splunk	Search engine	87.08	-1.32	-3.73
15.	15.	↑ 17.	Amazon DynamoDB	Multi-model	79.69	-1.87	-0.67
16.	16.	↓ 14.	Microsoft Azure SQL Database	Relational, Multi-model	78.75	-1.62	-6.20
17.	17.	↓ 16.	Hive	Relational	72.12	-2.22	-9.76
18.	18.	18.	Teradata	Relational, Multi-model	63.03	-2.40	-5.54
19.	19.		Databricks	Multi-model	60.33	-0.49	
20.	20.	20.	Neo4j	Graph	55.43	-0.41	-2.81
21.	↑ 22.	↑ 22.	FileMaker	Relational	52.80	-0.28	-1.33
22.	↓ 21.	↑ 24.	Google BigQuery	Relational	52.45	-1.97	+7.35
23.	23.	↓ 21.	SAP HANA	Relational, Multi-model	49.67	-1.67	-6.64

ORDBMS

- ORDBMS：物件關聯式資料庫管理系統。
- 特性：
  - 與RDBMS概念完全相容。
  - 支援使用者定義資料類型。
  - 支援多媒體和其他大型物件，可在單一資料欄中儲存影音、圖片、大量文字。

XML DB

- 一般而言，RDB處理XML文件可分成三種模式：
  - Textual fidelity: 將XML文件內容以純文字型態儲存在RDB的欄位內。
  - Relational fidelity (shredding + XML publishing): 將XML文件的內容及格式儲存在RDB的欄位內。
  - XML fidelity: 將XML文件按照schema結構以樹狀方式儲存。此模式稱為Native XML。
- 目前，主要的DBMS軟體都已在RDB中提供處理XML文件的機制，可透過一般SQL語法查詢XML文件；某些高階產品進一步提供符合XQuery標準的資料庫查詢功能。
  - XBRL文件也屬於XML格式，實務上，XBRL文件有兩種儲存方式，其一為單純以檔案方式存在資料夾內，其二則是以Native XML方式儲存在XML DB內，但此時XML DB需具有處理XBRL格式的能力。
  - 以Oracle為例，其XML DB可外加Oracle特別開發的XBRL Extension模組，XML DB就能處理XBRL格式文件。

JSON 文件的儲存

- JSON (JavaScript Object Notation)是一種輕量級的資料傳輸格式，目前已成為主流的文件交換格式。
- JSON將資料以key:value的配對方式表達，和Python語言的dict資料結構很像。它使用大括號{ }包住物件內容。

  JSON範例：

  {“city”:”台北市”, “mayor”:”柯文哲”, “population”:2608332}
- 目前，主要的DBMS軟體都已在RDB中提供處理JSON文件的機制，可以輕易地將文字資料輸出成JSON格式或是將JSON文件轉存進RDB內。
  - 大部分DB以純文字型態把JSON格式文件內容儲存在RDB的欄位內。
  - 某些DB將JSON的key和value分開儲存，以提高處理效率。
  - Oracle DB 21c有一個內建的json資料型態，可以更有效率處理JSON文件。

第二部份

個體關係模式 及 個體關係圖

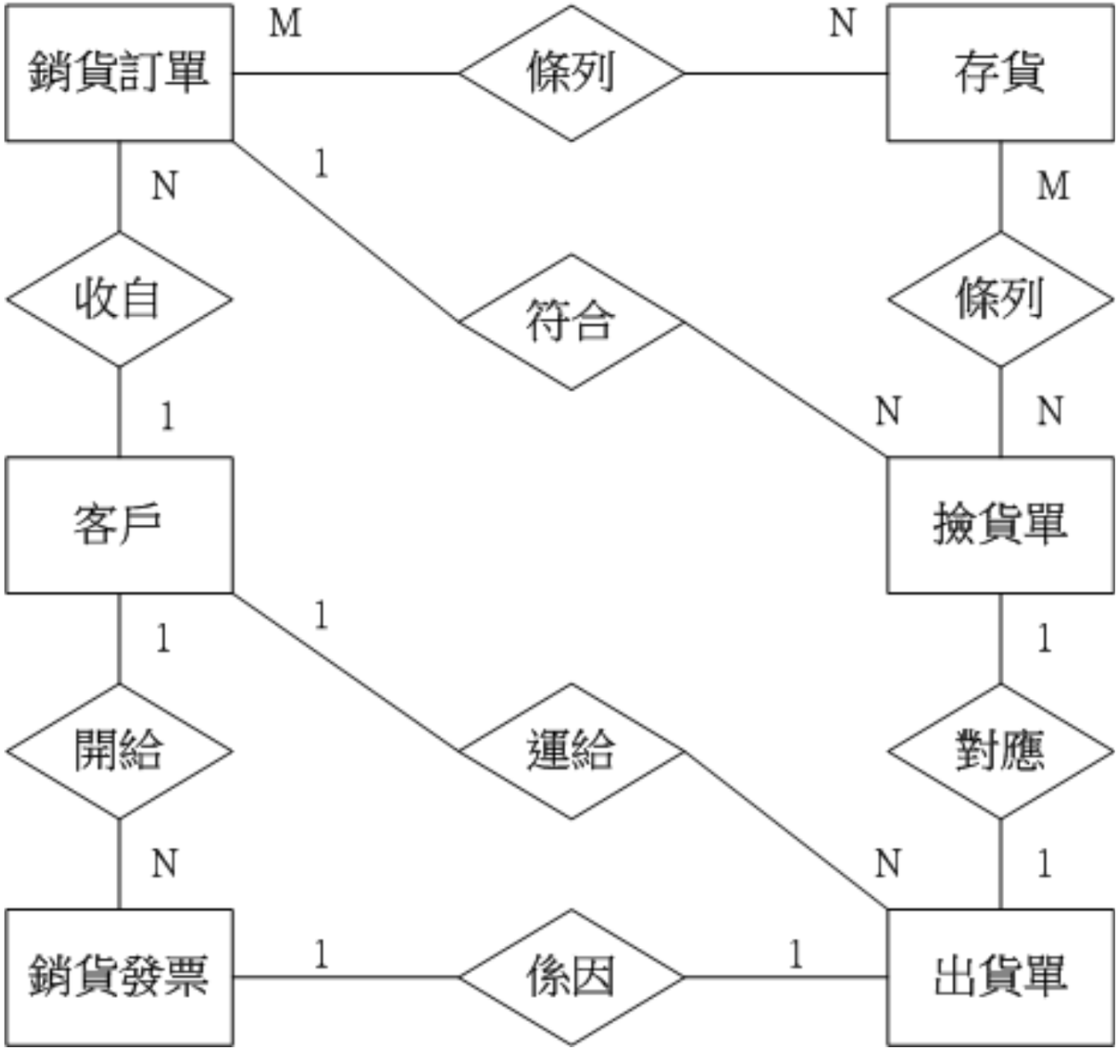
Data Modeling

- Data Modeling: 資料塑模，係指依據任何一種資料模式理論來建立資料模式(data model)的過程。
- Data model：資料模式，又稱為綱要(schema)，可分成三種
  - 概念資料模式(conceptual data model)：以簡化的模型描述組織的運作內涵，例如，以ERD的個體(E)來代表欲蒐集資料的對象，並在兩兩個體之間建立關係(R)，以描述個體之間的互動。
  - 邏輯資料模式(logical data model)：以詳細的書面表格將概念資料模式中的個體結構呈現出來，並藉由表格內的主索引及外來鍵呈現表格之間的關係。
  - 實體資料模式(physical data model)：在特定RDBMS軟體中實作出邏輯資料模式之內涵。

ERD & ER Model

- ERD：個體關係圖(entity-relationship diagram)。
- ER model是由Peter Chen在1976年發表，是一種概念資料模式(conceptual data model)。根據ER model所建立的概念圖形稱為ERD。
- ERD內通常包含四種元素：
  - 個體(entity, 即E)：代表欲蒐集資料的對象，以矩形表示。
  - 關係(relationship, 即R)：在兩兩個體之間建立合乎邏輯的關係，以菱形表示。在簡化的ERD中，通常會將關係的菱形圖形省略。
    - 關係的實質內容是以基數性(cardinality)來表示，可分為1對1、 1對多及 多對多三種。
  - 屬性(attributes)：個體或關係內的資料項目(即表格中的欄位)，以圓矩形表示。當描述稍微複雜的系統時，ERD內通常會省略此元素。

ERD範例：銷貨系統


銷貨訂單
M
條列
N
存貨
1
N
M
收自
符合
條列
1
N
N
客戶
撿貨單
1
1
1
開給
運給
對應
N
N
1
銷貨發票
1
係因
1
出貨單

基數性：1對1

- 基數性(cardinality)：兩個個體之間若具有特定關係，可用基數性表示其關係之內涵。
- 1對1關係 (1:1)：表示兩個個體都僅能以一個實例(instance)參與此關係。
- 例如：在前述ERD範例之「銷貨發票」與「出貨單」之間的基數性關係為1對1，意含「一張銷貨發票的內容對應一張出貨單，一張出貨單只能開一張銷貨發票」。

基數性：1對多

- 1對多關係 (1:N 或 1:∞ 或 1: *)：1方的一個實例，可以面對多方的多個實例；多方的一個實例，只能面對1方的一個實例。
- 例如：在前述ERD範例之「客戶」與「銷貨訂單」之間的基數性關係為 1對多，意涵「一位客戶可發出多次(張)銷貨訂單[客戶發出的進貨訂單，對供應商而言是銷貨訂單]，一張銷貨訂單只能來自一位客戶」。

基數性：多對多

- 多對多關係 (M:N)： 一邊多方的一個實例，可以面對另邊多方的多個實例；反之亦然。
- 例如：在前述ERD範例之「銷貨訂單」與「存貨」之間的關係為多對多，意涵「一張銷貨訂單內可包含多種存貨，一種存貨可出現在多張銷貨訂單內」。

基數性的相對性

- 兩個個體之間的基數性，並非絕對的。其內涵受到企業政策之左右。
- 例如：「公務車」與「業務員」之間
  - 1對1：表示「一部公務車只能讓一位業務員使用，一位業務員只能使用一部公務車」。
  - 1對多：表示「一部公務車可讓多位業務員使用，每位業務員只能使用一部公務車」。
  - 多對1：表示「一部公務車只能讓一位業務員使用，一位業務員可使用多部公務車」。
  - 多對多：表示「一部公務車可讓多位業務員使用，一位業務員可使用多部公務車」。

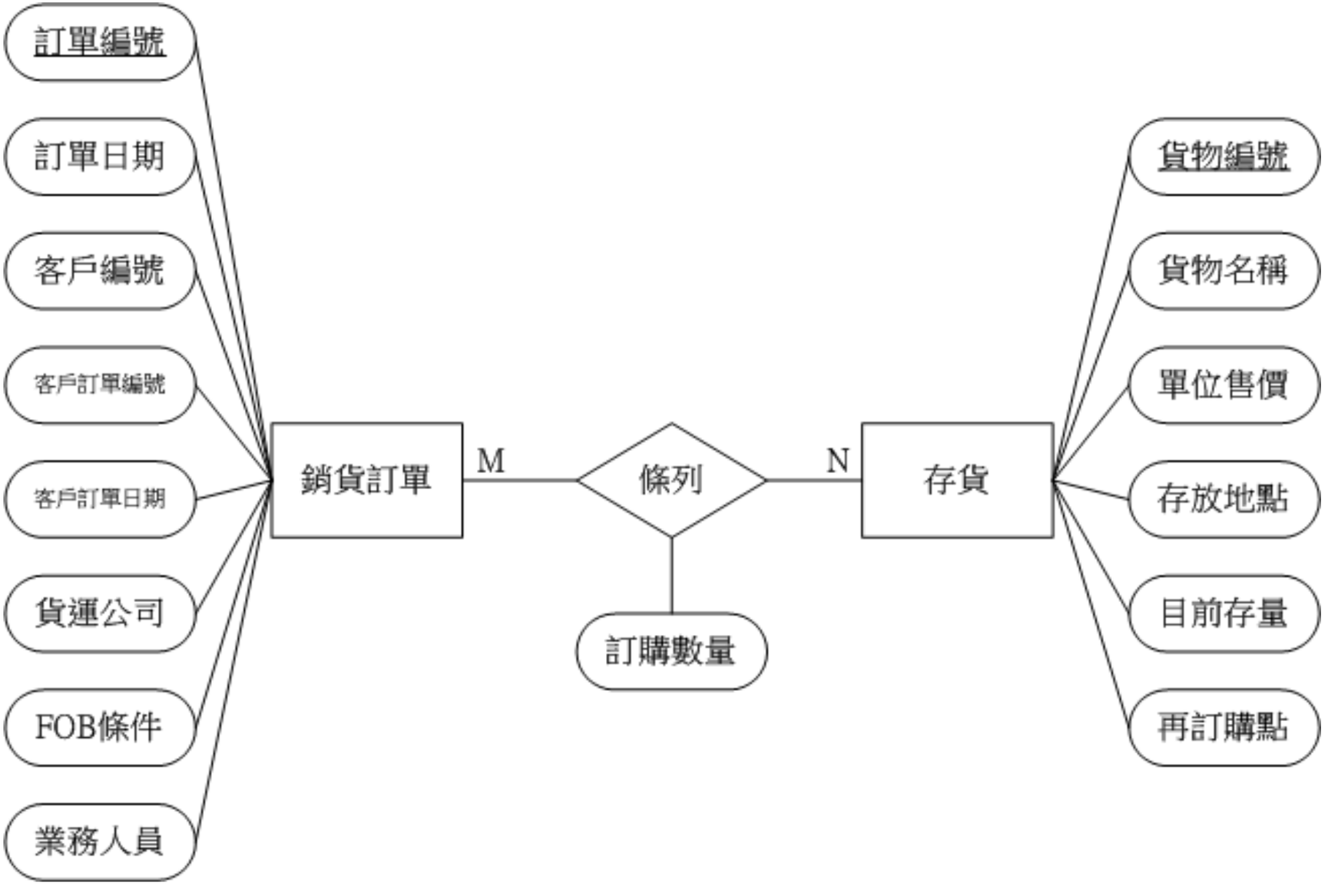
最小及最大基數性

- 最小及最大基數性(minimum & maximum cardinality)又稱為參與限制(participating constraint)，用以指明個體在特定關係中的最低及最高參與程度。
- 例如：
  - 「員工」(1, 1)對「已完成工作」(0, N)：表示一名員工可能沒完成任何工作(新進員工)，也可能已完成許多工作；一份工作僅能由一位員工完成(無法自動完成、也不能由超過一位員工合作完成)。
  - 「學生」(10, 60)對「課程」(2, 5)：表示一名學生至少需修2門課、至多能修5門課；一門課至少需有10名學生選修，至多不能超過60名學生。

屬性

- 屬性是個體或關係內的資料項目。在將個體或關係轉成資料表後，屬性就是資料表內的各個欄位。
  - 一對一 或 一對多 的關係沒有屬性，只有多對多之間的關係才有屬性，這些屬性將成為關係資料表(通常會把兩個多方的主索引納入，成為複合主索引)的一部份。
- 每個個體都有多個屬性，有些屬性的值可決定其他數個屬性的值，此種屬性稱為決定性屬性(determinant)。資料表經過正規化(normalization)後，剩下的決定性屬性稱為**候選鍵**(candidate key)，此屬性的值具有不重複性、且可決定所有其他屬性的值。候選鍵若不只一個，資料庫管理師(DBA)須挑選一個做為**主索引**(primary key，或稱**主鍵**)。
  - 例如，「員工編號」及「身分證字號」都是「員工」資料表的候選鍵，可任選一個做為主索引。在ERD中主鍵名稱應加上底線。

屬性標示範例：訂單&存貨


訂單編號
訂單日期
客戶編號
客戶訂單編號
客戶訂單日期
貨運公司
FOB條件
業務人員
銷貨訂單
M
條列
N
存貨
訂購數量
貨物編號
貨物名稱
單位售價
存放地點
目前存量
再訂購點

將ERD轉換成邏輯模式

- ERD描繪完成後，可按以下步驟轉成邏輯資料模式：
  1. 為每個個體編製一張資料表(稱為relation或table)，ERD內之屬性即為資料表內的各個欄位，且每張資料表內須包含一個主索引欄位。
  2. 在1對多關係中， 把1方資料表的主索引放在多方資料表內，此欄位稱為多方的外來鍵(foreign key)。表格間的關聯即藉此建立。
  3. 在多對多關係中，另外增加一個關係資料表，將兩個個體的主索引納入此資料表，成為複合主索引(composite PK)。亦即，將原本的一組多對多關係，轉成兩組1對多關係(關係資料表為多方)。
  4. 在1對1關係中，把未來最有可能轉成多方的個體暫時當成多方，然後按照1對多關係處理。若難以判定，亦可互將對方的主索引納入成為外來鍵。
- 例如：若將前述銷貨系統ERD範例轉成邏輯模式，需包含8張資料表，其中兩張為關係資料表。

邏輯資料模式範例

- 將前述銷貨系統ERD範例轉成邏輯資料模式如下(共8張表格)：

客戶

客戶編號	客戶名稱	地址	城市	國家	郵遞區號	收貨人	送貨地址	送貨城市	送貨國家	送貨郵區	信用額度	最後修正	信用條件
A001	上華公司	民和路54號	高雄	中華民國	800	相同	相同	相同	相同	相同	5,000,000.00	20071012	2/10,n/30
A002	永康公司	健康路185號	屏東	中華民國	900	大方公司	馬甲路8號	上海	中國	n/a	850,000.00	20070630	n/60

銷貨訂單

訂單編號	訂單日期	客戶編號	客戶訂單編號	客戶訂單日期	貨運公司	FOB條件	業務人員
B001	20071013	A001	C00123	20070930	新竹貨運	起運點	李麗華
B002	20071014	A002	H34892	20071001	DHL	目的地	范小文

銷貨訂單 條列 存貨

銷貨訂單編號	貨物編號	訂購數量
B001	D001	150
B001	D003	80
B002	D004	90

揀貨單

揀貨單編號	揀貨日期	揀貨員	銷貨訂單編號
C001	20071015	張五哥	B001
C002	20071017	王唯一	B002

撿貨單 條列 存貨

撿貨單編號	貨物編號	揀貨磅數
C001	D001	150
C001	D003	80
C002	D004	90

存貨

貨物編號	貨物名稱	單位售價(磅)	存放地點	目前存量(磅)	再訂購點
D001	摩卡咖啡豆	500.00	倉一	1000	500
D002	爪哇咖啡豆	560.00	倉一	1500	600
D003	曼特寧咖啡豆	650.00	倉二	2300	800
D004	藍山咖啡豆	890.00	倉二	800	400

出貨單

出貨單編號	出貨日期	出貨人員	撿貨單編號	客戶編號
E001	20071018	陳錦芳	C001	A001
E002	20071020	伍瑞于	C002	A002

銷貨發票

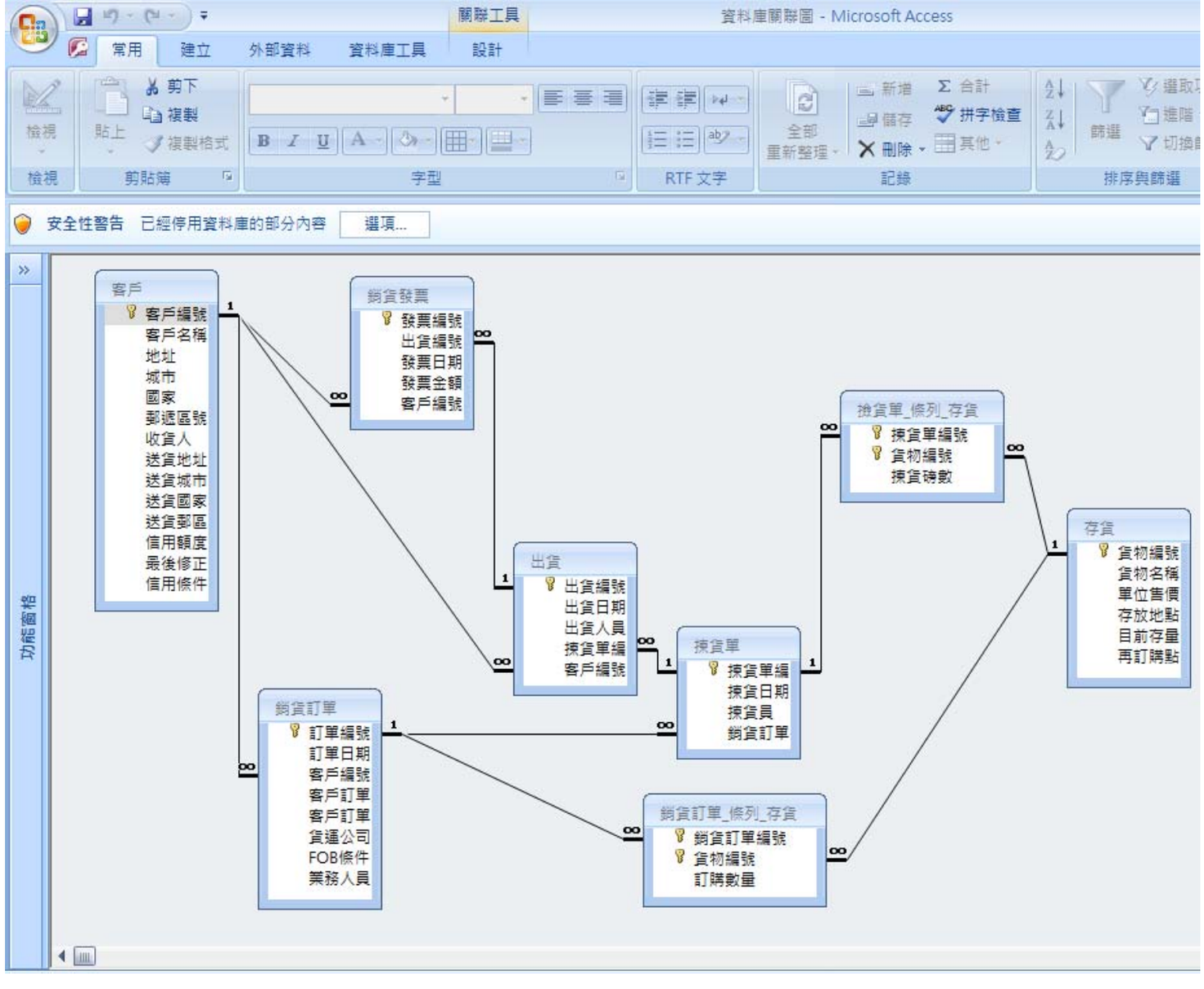
發票編號	出貨單編號	發票日期	發票金額	客戶編號
F001	E001	20071018	127,000.00	A001
F002	E002	20071021	80,100.00	A002

外來鍵 & 參考完整性

- 多方資料表內任一筆資料列的外來鍵值，可連結至一方資料表內的特定資料列。為確保此外來鍵值所指向的資料列確實存在、未被刪除，可在DBMS內設定外來鍵的參考完整性(referential integrity)。一經設定後，一方的特定資料列的主索引值只要被多方特定資料列的外來鍵引用，該一方特定資料列即無法被刪除(除非修改多方的外來鍵值)。
- 有時，多方特定資料列的外來鍵的確無法在一方找到對應的主索引值，此外來鍵可設定為空值(null)。
  - 例如，在前述銷貨系統ERD範例之「銷貨發票」與「客戶」之間的關係為多對1，因此，銷貨發票資料表內會有一欄位為客戶編號(客戶資料表之主索引)。在現銷的情境下，許多客戶並不願留下任何記錄，故此類交易之銷貨發票資料表內客戶編號欄會留白。

實體資料模式：MS Access

- 將前述邏輯資料模式實作在MS Access資料庫所產生之資料庫關聯圖如右：



第三部份

REA概念資料模式及企業本體論

REA 會計資料模式

- REA (Resource [資源] -- Event [事件] -- Agent [代理人]) 會計資料模式是由William E. McCarthy創建，他根據企業交易的特性將ER model中的個體(entity)區分為資源、事件、代理人三大類，以事件為中心，三大類個體彼此環環相扣。
- REA模式最早發表於The Accounting Review這份頂尖會計期刊 (1979,1982)，由於模式中強調「會計科目及借貸法則是會計系統中不必要的人為設計」，觸怒了不少會計界大老，McCarthy的文章此後多年不再見容於頂尖會計期刊。
- 隨著過去20年來ERP系統的發展及逐漸普及，使得原生概念與ERP頗為契合的REA模式鹹魚翻身，目前已成為AIS領域的重要教學及研究主題。

REA 企業本體論

- REA模式的原始設計可說是ER模式的會計改良版。
- Geerts and McCarthy從1990年代開始對原始REA模式進行擴充，讓REA發展成一套可據以建構整合性企業資訊系統的企業領域本體論(enterprise domain ontology)。
  - 本體論(ontology)：探討事物的存在本質。
  - 領域本體論(domain ontology)：對攸關特定領域的事項做定義。
  - REA企業本體論：針對適用於所有企業的構念(construct)做定義，並說明這些構念如何表達在整合性企業資訊系統內。
  - 整合性企業資訊系統：即通稱的ERP (enterprise resource planning)系統。O'Leary (1999)針對REA本體論與SAP ERP系統的內涵進行比對研究，發現兩者間有許多相似性，但SAP ERP系統內仍有一些不符合REA標準的設計。

企業本體論：描述與樣式

- 真實的企業是十分複雜的，必須透過簡化的模型來描述重要的運作內涵。描述(representation)的層次可以由簡至繁，且可遵循某些通用的樣式。
- 樣式(pattern)可分成兩大類：
  - 物件樣式(object pattern)：包含物件及其間的關係。例如：企業有董事會、經理人、員工、辦公室、營業所、設備…等物件，各物件間可能有管理、服從、使用、維修…等關係。
  - 腳本樣式(script pattern)：指一系列前後攸關的事件。例如：企業從融資循環取得資金、進貨付款循環取得商品存貨、薪工循環募集人力並支付薪資、銷貨收款循環賣出商品取回貨款、再回到融資循環償還資金，為完整的營業循環腳本樣式。

※ 設計樣式(design pattern)原本是物件導向(OO)領域內的程式設計方法論，此概念逐漸普及至企業流程分析，形成各種企業樣式。

REA 本體論的三個導向

Dunn and McCarthy (1997)提出REA本體論的三個導向：

- 資料庫導向(database-oriented)：資料必須以最原始的層次儲存至少一段期間；相同資料必須只儲存一次，且能讓所有授權者存取；資料必須能讓使用者以所需格式擷取。
- 語意導向(semantic-oriented)：系統內建的企業程序應盡可能符合個別企業的實況 [所以外購的ERP套裝軟體並不符合此概念]；系統描述的物件應盡可能符合其真實情況，人工化的構念(artificial construct)應盡可能排除 [所以最純粹的REA資訊系統不採用借貸法則及會計科目]。
- 結構化導向(structure-oriented)：企業資訊系統應根據樣式來建立。

REA 的爭議：會計科目

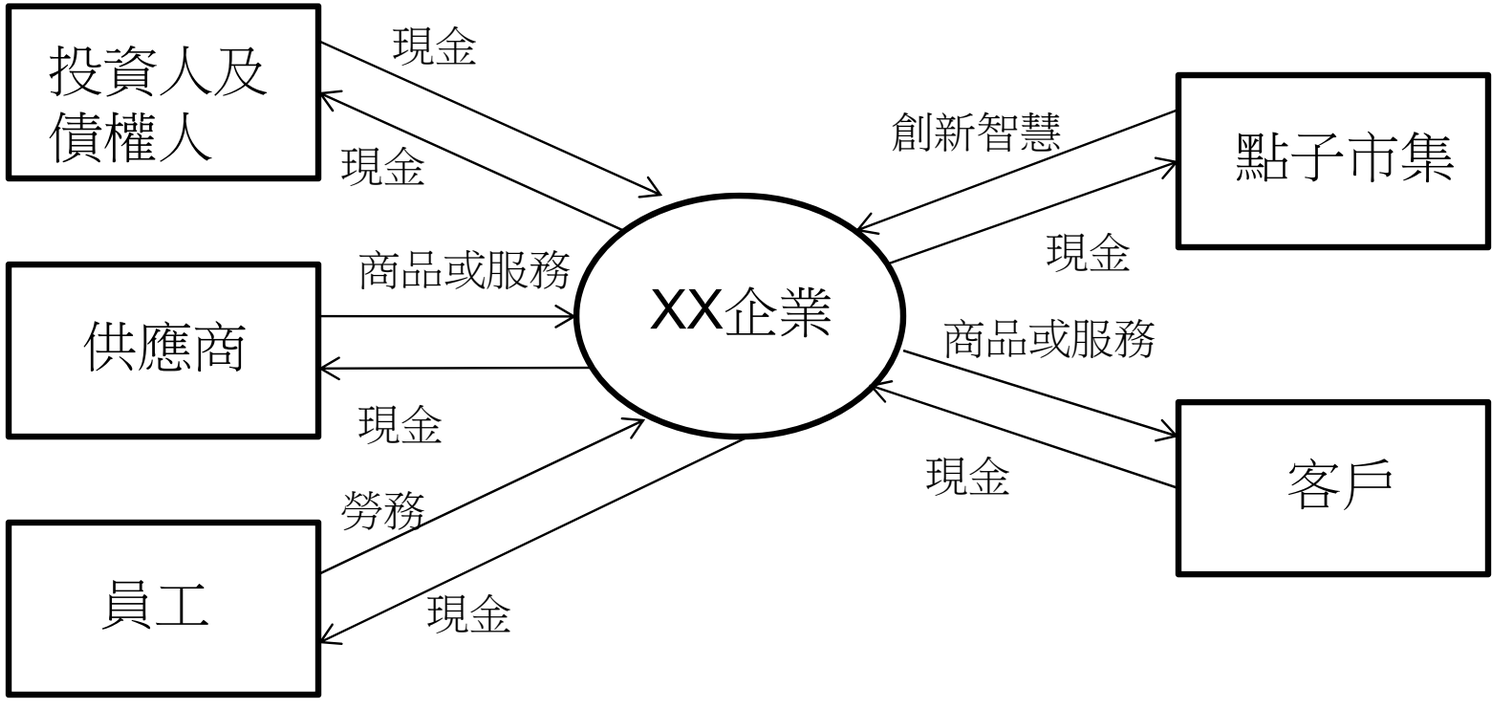
- REA模式強調會計系統不必使用會計科目及借貸法則，因為它們並非實際交易的核心，而僅是為了會計報告的目的所做的額外記錄。
  - 例：爸媽給兒女生活費，雖然沒有「借：長期投資 貸：現金」，並不改變給生活費這件事實。
  - 在資料庫導向下，只要資料以最原始的層次(實際的交易細節)儲存，並讓使用者以所需格式擷取，即可在需要編製會計報告時以動態查詢方式產生報表上所需要的每一項數據。
  - 換言之，REA模式認為會計科目的數據值可由資料庫內原始層次資料項的值推論而得，因此會計科目屬於可推論屬性(derivable attributes)，不應列入資料表內，應改以查詢方式動態取得數據。
- 但實務上，即使系統內涵最接近REA模式的ERP軟體，都設有會計科目表。REA學者認為這是一種向傳統作業習慣妥協(compromise)的設計，而非流程上的真正需求。

REA 本體論的四個層次

- Dunn et al. (EIS, 2005)把REA企業本體論由簡至繁分成四個層次：
  - 價值系統層(value system level)：描述資源如何在企業及其外部伙伴(包含供應商、客戶、投資人/債權人、員工等)之間流動。
  - 價值鏈層(value chain level)：描述資源如何在企業內的不同企業程序(進貨付款、銷貨收款...)之間流動。
  - 企業程序層(business process level)：描述特定企業程序內的資源、事件、代理人等個體及彼此間的關係。
  - 工作層(task level)：描述特定企業程序的工作流程細節。

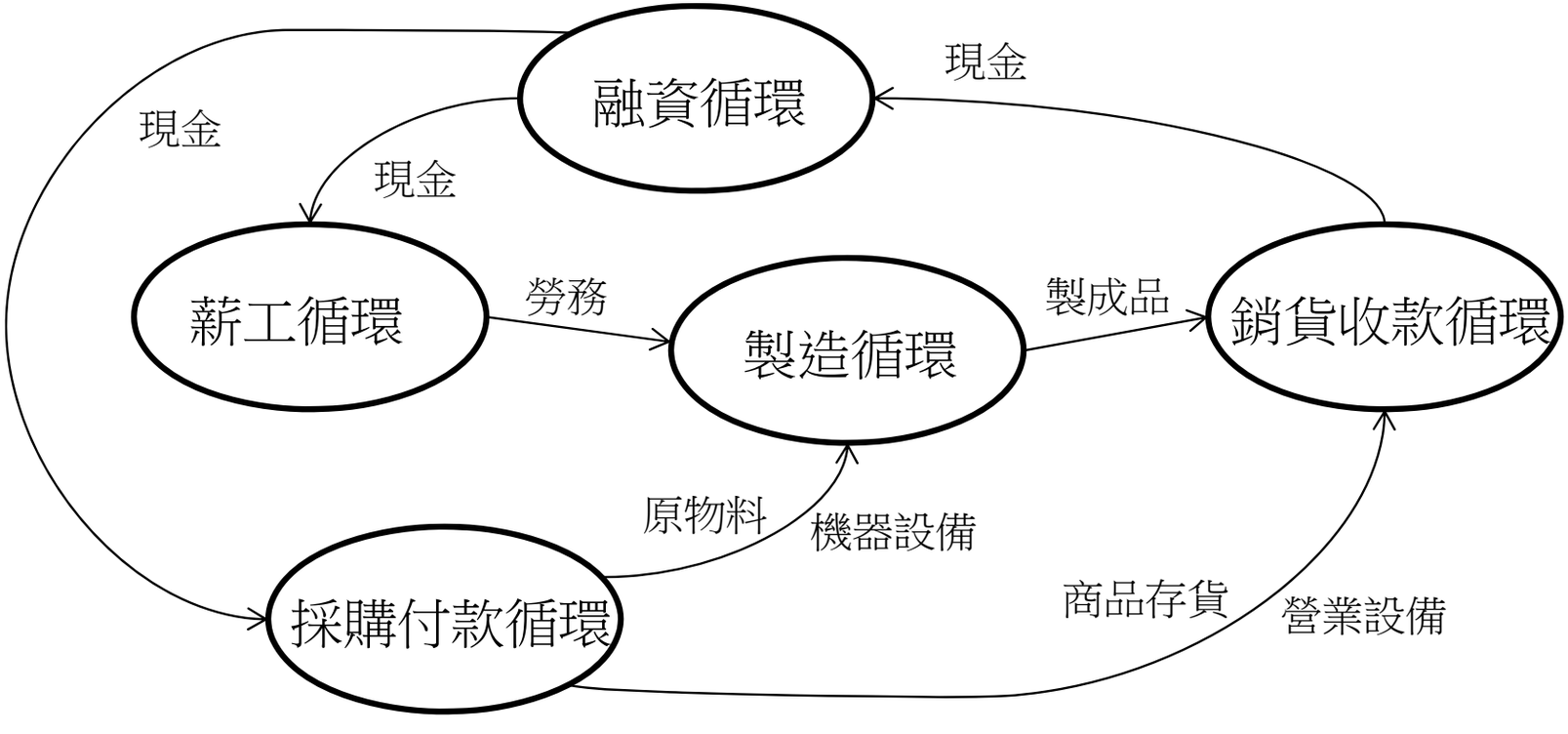
REA：價值系統層

- 價值系統層描述資源如何在企業及其外部伙伴之間流動，是一個物件樣式的REA模型。通用樣式如下：



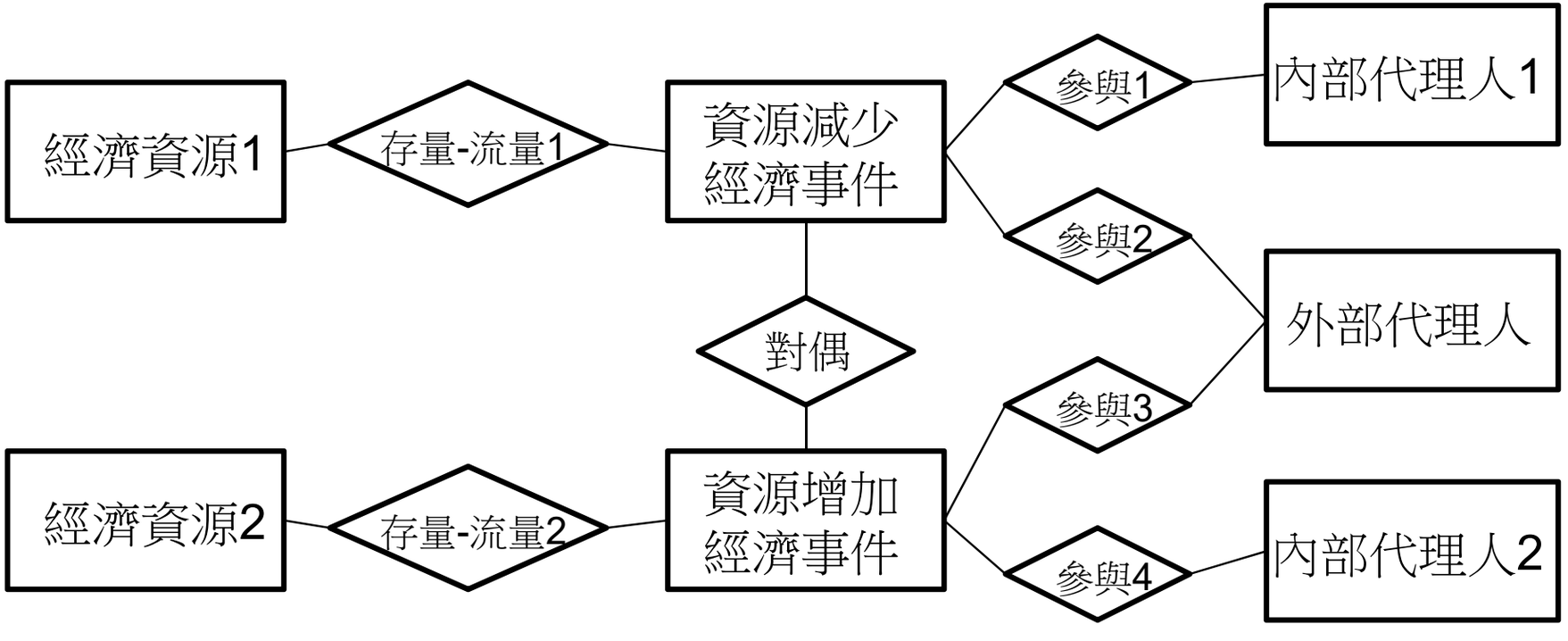
REA：價值鏈層

- 價值鏈層描述資源如何在企業內的不同企業程序之間流動，是一個腳本樣式的REA模型。通用樣式如下：



REA：企業程序層

- 企業程序層描述特定企業程序內的資源、事件、代理人等個體(entity)及彼此間的關係(relationship)，是一個物件樣式的模型(註：若向上延伸增加契約事件及帶頭事件，也可以當成腳本樣式)，也是REA企業本體論的核心所在。基本樣式如下：



REA：工作層

- 工作層描述特定企業程序的工作流程(workflow)細節，由於流程細節的選擇眾多，因此並無一定樣式可循。
- 除了可用文字敘述外，許多系統分析的方法及工具都可用來描述工作流程細節，包含系統流程圖、資料流程圖、程序圖、魚骨圖以及UML的活動圖、順序圖等。

※ Hruby (2006)的進階REA教科書，使用REA模式描述各種企業程序的工作流程細節，證明REA也能成為建構工作層企業樣式(business pattern)的基礎工具。

企業程序層：事件

- REA企業程序層係以事件(event)為核心，左接增減的資源(resource)、右連參與事件的內外部代理人(agent)。
- REA模式內的事件可分成以下幾種：
  - 經濟事件(economic event)：是REA樣式圖的核心事件，每一種企業程序內通常包含兩個或多個具有對偶關係(duality)的資源增加事件及資源減少事件。例如，銷貨(資源減少)與收款(資源增加)、採購(資源增加)與付款(資源減少)都是典型的對偶事件。
  - 契約/相互承諾事件(contract/mutual commitment event)：是促成經濟事件發生的契約事件。例如，「客戶下訂、企業接訂」後，才有後續的銷貨及收款，因此前者為相互承諾事件。
  - 帶頭事件(instigation event)：是促成相互承諾事件發生的事件。例如，企業的行銷活動，以及客戶詢價、企業報價，都算是促成訂單簽訂的帶頭事件。
  - 轉迴性經濟事件(reversal economic event)：是「非現金資源」增減事件的轉迴事件，亦可算是經濟事件的一種。例如，銷貨是資源減少經濟事件，銷貨退回則是前項事件的轉迴事件。

企業程序層：資源

- REA模型內的資源可分成以下兩種：
  - 資源(resource)：可個別辨識且有獨立編號的資源，通常適用於量少、價高的物件，例如：鋼琴、汽車等。
  - 資源型態(resource type)：大量生產、使用共同產品編號的資源，例如：超商或量販店中販售的大部分貨品(例如：瓶裝鮮奶、速食麵、洋芋片..等)。

企業程序層：代理人

- REA模型內的代理人分成以下兩類：
  - 內部代理人(internal agent)：為參與特定事件的企業內部員工。
    - 員工不一定是內部代理人。在薪工循環中，員工也可能是外部代理人。
  - 外部代理人(external agent)：為參與特定事件的企業外部伙伴(例：客戶、供應商，以及薪工循環中的員工)。
- 因為代理人是參與特定事件的人或機構，因此也可把agent 翻譯成「參與者」。
  - 註：Agent 這個字在美式英文中不一定表示代理人。例如，MLB 的 free agent，中文翻譯成「自由球員」，就比「自由代理人」更為貼切。

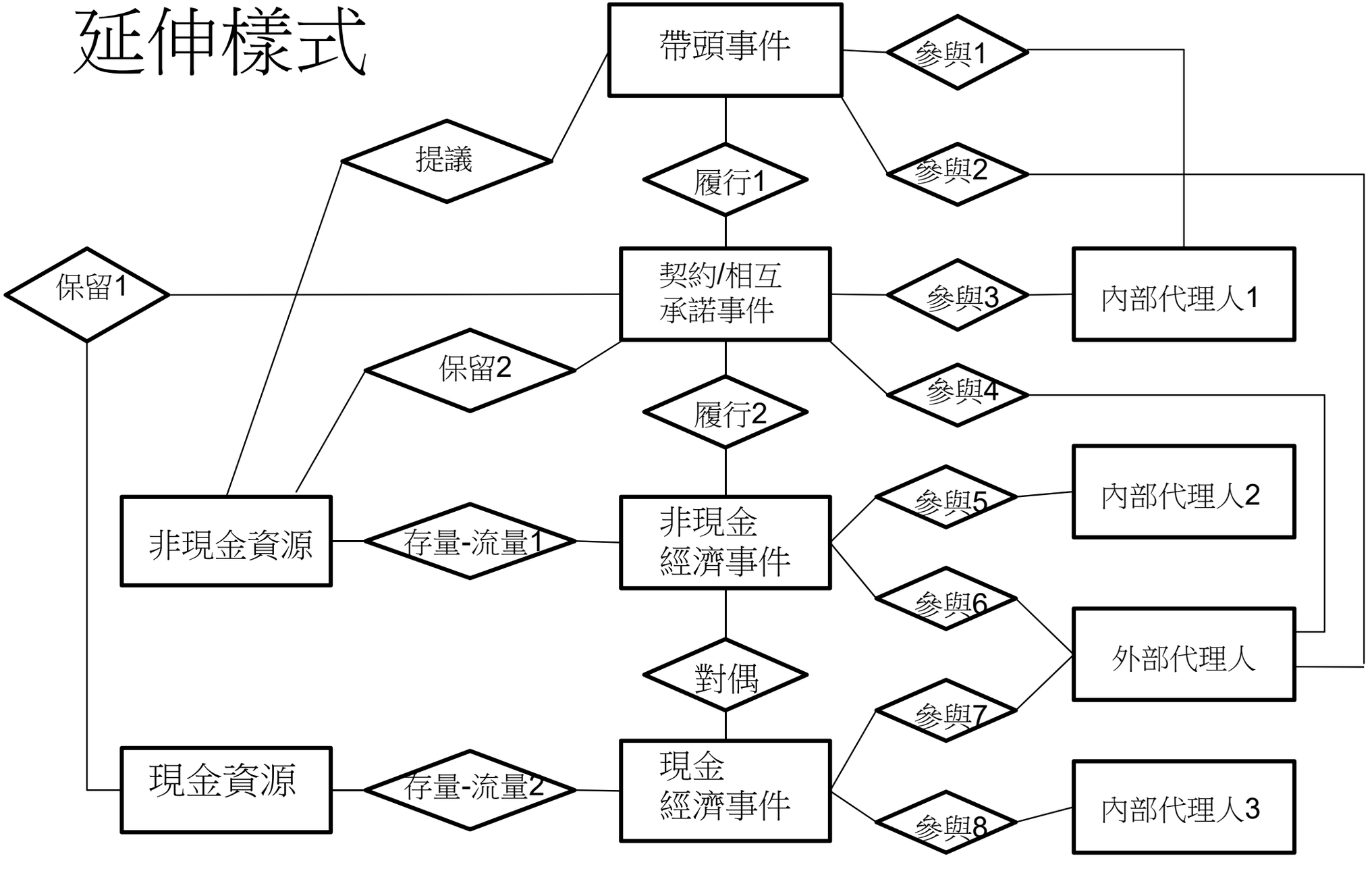
企業程序層：關係 2-1

- REA企業程序層兩個個體之間有以下多種關係：
  - 事件之間的關係：
    - 對偶關係(duality)：存在於兩個資源增減經濟事件之間。
    - 相對關係(reciprocal)：存在於兩個分別連接不同資源增減事件的相互承諾事件之間。因REA樣式圖通常只為個別企業程序繪製一個相互承諾事件，因此相對關係並不常見。
    - 履行關係(fulfillment)：存在於相互承諾事件與資源增減經濟事件之間，以及帶頭事件與相互承諾事件之間。
    - 轉迴關係(reversal)：存在於非現金資源增減經濟事件與其轉迴事件之間。
  - 事件與代理人之間的關係：
    - 參與關係(participation)：存在於事件與內外部代理人之間。

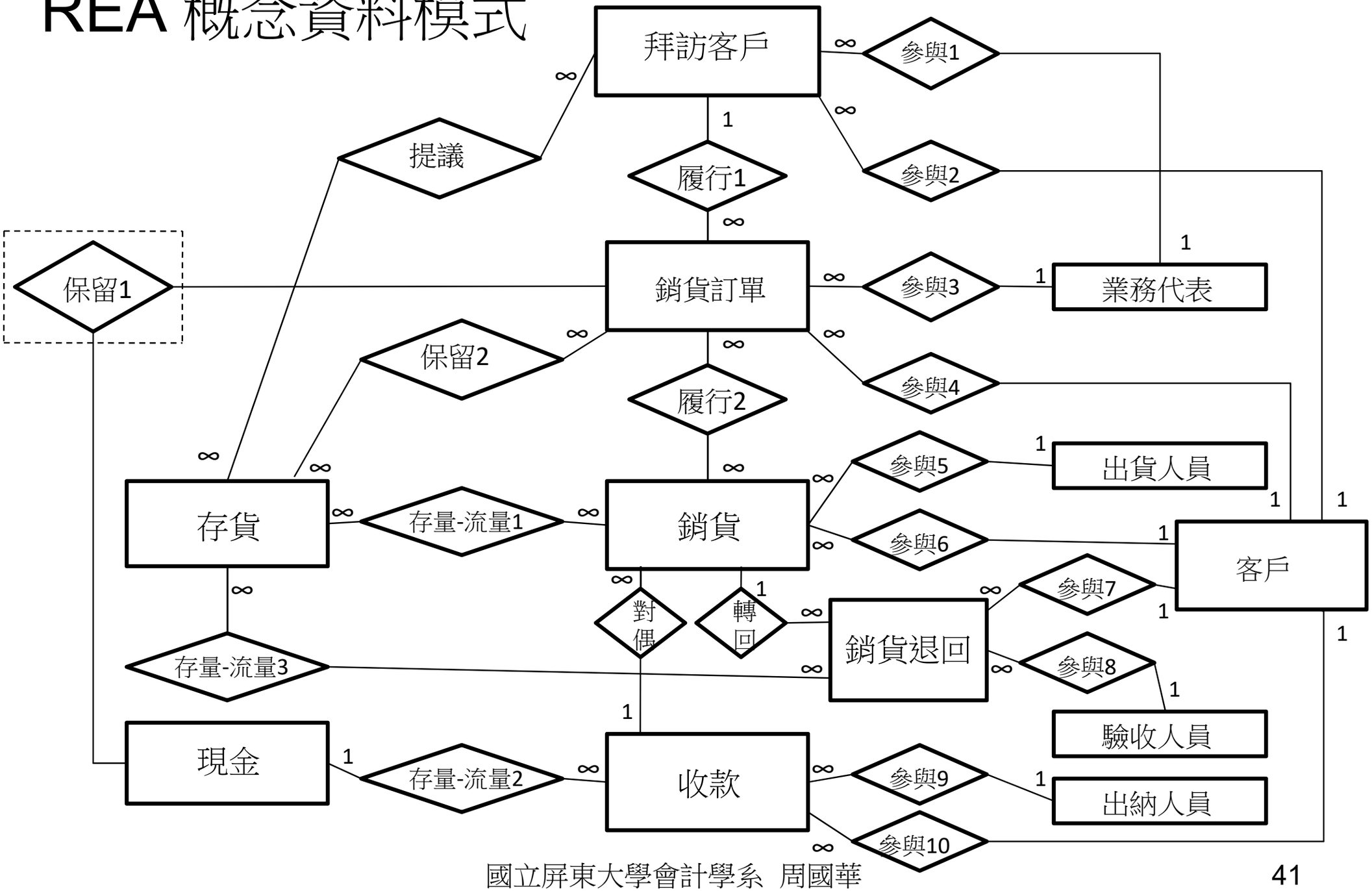
企業程序層：關係 2-2

- 事件與資源之間的關係：
  - 存量-流量關係(stock-flow)：存在於經濟事件與資源之間。
  - 保留關係(reservation)：存在於相互承諾事件與資源之間。
  - 提議關係(proposition)：存在於帶頭事件與資源之間。
- 代理人之間的關係：
  - 指派關係(assignment)：存在於內部與外部代理人之間。
  - 責任關係(responsibility)：存在於兩個內部代理人之間。
- 其他關係：
  - 監管關係(custody)：存在於內部代理人與資源之間。
  - 連結關係(linkage)：存在於兩種資源之間。
  - 型態化關係(typification)：存在於資源與資源型態之間。
  - 一般化關係(generalization)：存在於父子個體之間。

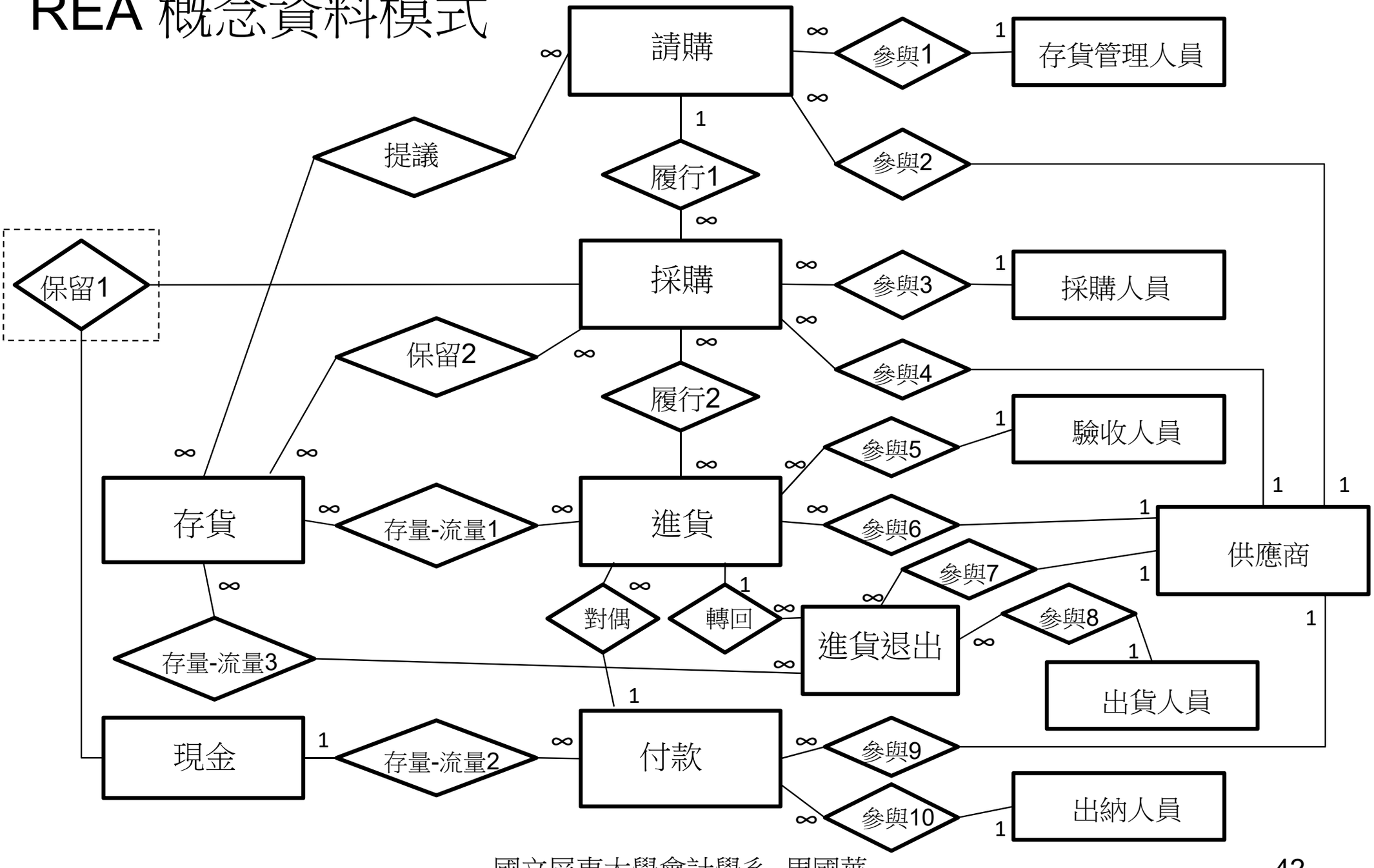
企業程序層延伸樣式


帶頭事件
參與1
提議
履行1
參與2
保留1
契約/相互承諾事件
參與3
內部代理人1
保留2
履行2
參與4
非現金資源
存量-流量1
非現金經濟事件
參與5
內部代理人2
參與6
外部代理人
對偶
參與7
現金資源
存量-流量2
現金經濟事件
參與8
內部代理人3

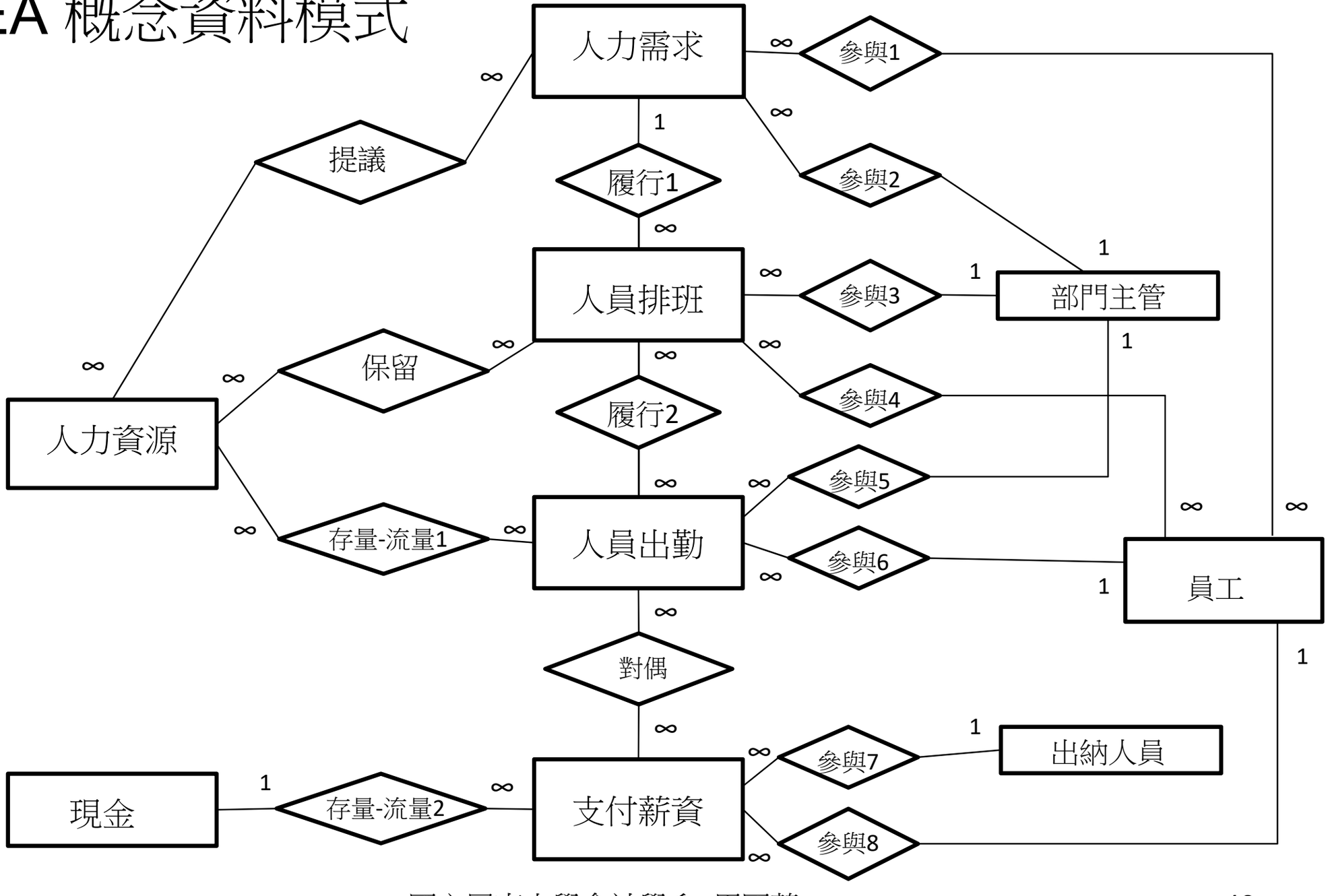
銷售收款循環
REA 概念資料模式



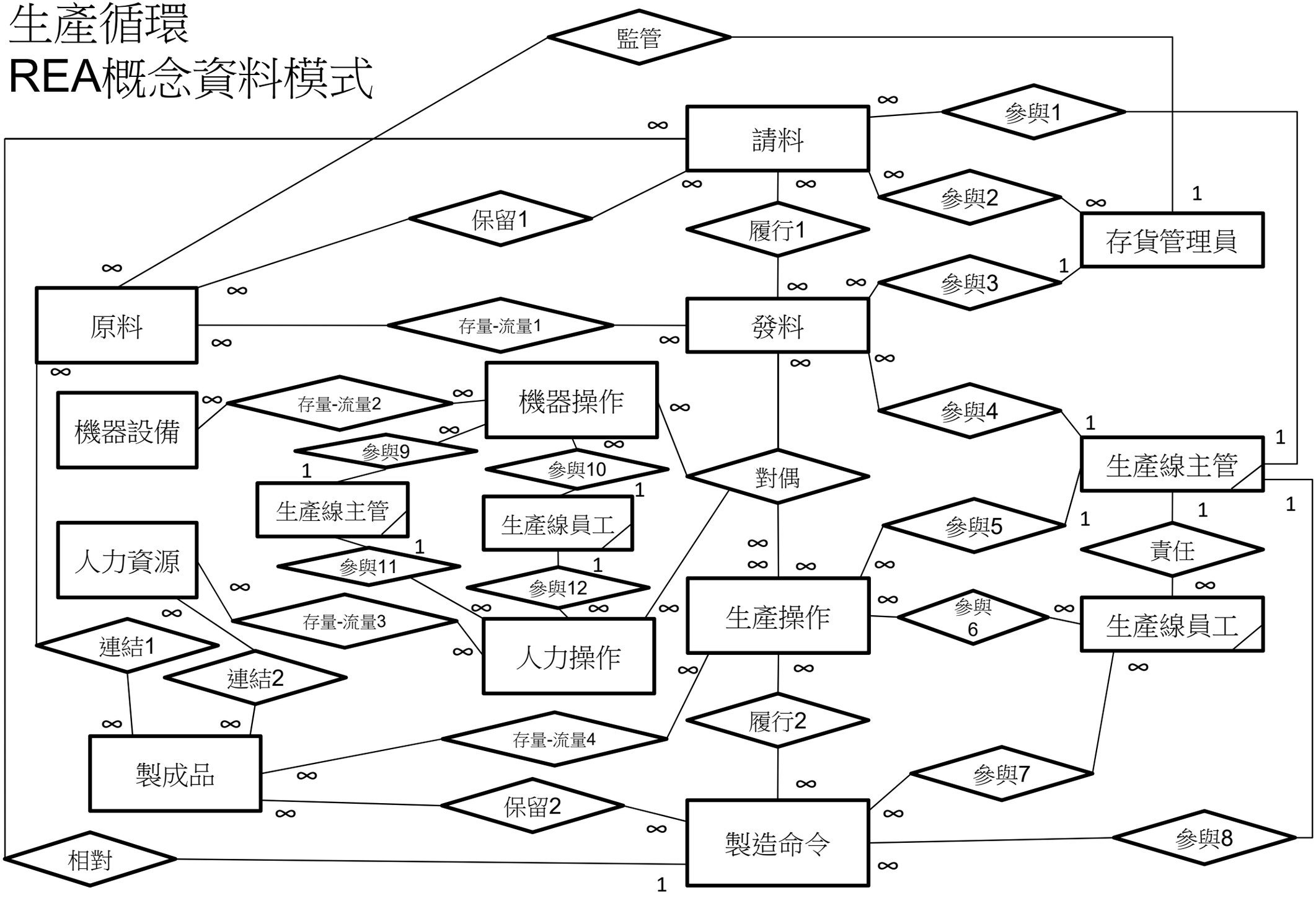
採購付款循環
REA 概念資料模式



薪工循環
REA 概念資料模式



生產循環
REA概念資料模式



融資循環
REA 概念資料模式