應用本體論於主檔資料管理之研究

翁頌舜
國立台北科技大學資訊與運籌管理研究所
wengss@ntut.edu.tw

王貞淑
國立台北科技大學資訊與運籌管理研究所
wangcs@ntut.edu.tw

謝孟原
國立台北科技大學資訊與運籌管理研究所
norm0412@gmail.com

摘要

近年來學術界與產業界正積極發展以語意為導向的資料存取與整合之應用，若將本體論(Ontology)整合應用到資料中，將使資料容易分享與明確規範，並使機器能夠理解，如此達到快速資料整合，獲得單一事實的資料，並提高企業彈性與簡化流程，避免資訊錯誤情形發生。本研究發展一套以本體論為基礎之三階段資料整合模式，以協助企業進行主檔資料管理。第一階段中，本模式透過以逆向工程為基礎的轉換規則演算法，將關聯式資料表轉換為本體論；第二階段中，將轉換後的本體論透過合併工具 PROMPT 進行整併，產生全局本體論(Global Ontology)；最後於第三階段中，利用本體論查詢語言 SPARQL，使用者可以針對個別或全局本體論進行查詢，產生有用的資訊。主檔資料為企業常用之核心資料，企業若以本體論將主檔資料明確定義與形式化，藉以正確、快速的整合與管理主檔資料，企業可有效利用與分享這些資料，作為行銷與分析的來源，達到獲利。

關鍵詞：本體論、資料整合、逆向工程、主檔資料管理、主檔資料

壹、 導論

隨著企業不斷發展、外部競爭提升與全球經濟危機的威脅，使企業面臨越來越多的挑戰，為了更靈活與彈性，滿足發展的各種需求，永續的企業須不停變革來符合市場環境的變化，已是不可避免的趨勢，因此以各種相關資料為核心競爭力的企業，必須仰賴與利用正確與容易分享的資料來分析，進而提供顧客服務，達到企業獲利與企業永續經營。根據調查指出，企業組織的資料錯誤率高達 30%，另一調查指出 83%的受訪公司的主檔資料品質明顯不好[14]，推估其成因，主要是缺乏共同標準，加上事實資料在許多不同系統間被存取多次，導致須經常性的人工介入來核對與調整資料，使得資料定義、資料格式與資料數值彼此不一致，關鍵資料品質若下降，很難被企業使用者所理解

和使用[14]。

主檔資料（Master Data, MD）是指企業跨越供應商、客戶、產品或組織單位等企業流程和系統且不斷被使用的核心企業實體[4]，它提供企業各種內容豐富的資料，因此核心的主檔資料管理變得越來越重要。主檔資料管理（Master Data Management, MDM）旨在提供企業組織一種全面與廣泛的方法，無論資料儲存在何處，它能使企業組織有能力整合、分析和開發企業組織資料資產的價值[14]。在大型企業中，主檔資料儲存在許多獨立的系統中，且品質往往是未知的，所以主檔資料管理需導入企業，以提供企業正確與單一事實的資料，卑利決策者做出正確的企業策略[7]。主檔資料管理通常與企業資源規劃系統（Enterprise Resource Planning System, ERP System）共同發展，由 ERP 系統來提供資料來源，而 MDM 提供 ERP 系統單一事實版本的資料，彼此相輔相成[11]。企業所用的企業資源規劃系統、顧客關係管理系統（Customer Relationship Management System, CRM System）、供應鏈管理系統（Supply Chain Management System, SCM System）與商業智慧系統（Business Intelligence System, BI System）等資訊系統，通常是分散式運作的獨立系統，因此儲存於個別資料庫的主檔資料是需要有效管理與利用。主檔資料管理也用來輔助商業智慧系統中的萃取、轉換與載入資料，與提供商業智慧系統中資料倉儲的資料管理[7]，鑒於這些因素，本研究將利用本體論的特性來有效管理這些資料。

本體論在電腦科學領域裡是表達形式詞彙之涵義與原意的邏輯方法[9]，也是一種概念化的形式和明確的規範[12]，消除非正式符號（如自然語言）的模糊性提供能使資料被機器閱讀和理解的方法，因此它提高系統間的互操作性和知識分享[8]。一般而言，本體論形式地描述真實世界各個關心的領域，通常本體論由術語（Terminology）的有限清單和術語之間的關係（Relationships）所組成[1]。術語表示領域中的重要概念（類別 Classes 與物件 Objects）。關係包括類別的階層（Hierarchies Of Classes），也就是父類別（Superclass）和子類別（Subclass）之間的關係等。在語意網領域中，本體論創造使電腦與機器能懂的語意資料，來協助資訊網搜尋、解釋與溝通所取得的資訊[5]。資料若以本體論形式表示，將可發揮其同等的特性，使資料更容易管理與應用，因此本研究透過逆向工程流程來實現完整且無遺失資料相關性的轉換。

逆向工程目的在於分析系統來確定所有系統元件和元件之間的關係[12]，逆向工程也可藉由分析資料庫，來萃取實體關係和物件模型。早期從資料庫綱要萃取語意的方法，會遺失許多必要的語意[13]，由於本體論的特性，是理想的資料庫整合解決方案，因此近年利用逆向工程將關聯式資料庫轉換成本體論的方法，以保留較充足的資料語意與資料間的關係[6]。逆向工程方法就是將資料庫轉換為本體論的方法，其輸出目標是建立語意本體，而基於資料來源的類型可大致分為五類[2]，本研究採用「以分析值組（Tuples）為基礎的方法」。逆向工程主要由兩個流程所組成，一個是模型轉換（Schema Transformation）流程，另一個是資料轉移（Data Migration）流程。模型轉換流程是分析關聯式資料庫，將資料表轉換與映對成為本體論。首先分析資料庫的資訊來獲得含有資料庫語意與描述資料本身意義的概念綱要（Conceptual Schema），而分析的資訊包含鍵值、屬性與資料，再來將獲得的概念綱要轉換成本體論；資料轉移流程則是將實例

(Instance)加入至轉換後的本體論中，並分配屬性值給實例[3]。

本研究將主檔資料管理與本體論之相關轉換、合併與對齊等技術相結合，提出主檔資料整合模式，發展關聯式資料表轉換成本體論之轉換規則，並提出合併、整合本體論，進而有效利用整合後的本體論之三階段架構。本研究之主檔資料整合模式所轉換的對象是關聯式資料庫中的資料表，在第一階段中，將其以逆向工程的概念，透過本研究進一步發展的專換規則演算法，轉換成為本體論。在第二階段中，將兩個轉換後的本體論進行合併。在第三階段中，透過查詢語法來找出有用資訊，並驗證本研究之可行性。本研究目的有以下幾點：

1. 提出以本體論為基礎之主檔資料整合模式，以建立彈性與可擴展的主檔資料管理。
2. 進一步發展以逆向工程為基礎的轉換規則之演算法，將關聯式資料表轉換成為本體論。

貳、 研究架構

本研究提出以本體論為基礎之三階段資料整合模式，此處將詳述說明流程與內容。

本研究發展一套如圖 2 所示的以本體論為基礎之資料整合模式，此模式分為三個階段，包括第一階段的關聯式資料表轉換本體論之轉換流程、第二階段的本體論對齊與合併之整合流程、以及第三階段的本體論查詢應用流程，其中第一階段轉換流程如圖 3 所示，又分為三個子流程。本資料整合模式是採取自下而上的本體論整合方法，其特性是建立本體論之間的連結，尚能保留各個資料庫的原本獨立性，使資料既容易維護，又易於在不同的組織間與異質系統間運作；使用者可對整合後的本體論來進行相關查詢，以獲得使用者所需要的資料，甚至找出資料之間潛在的關係。此三個階段分別說明如下：

第一階段：本體論建立

本研究以本體論的形式來表達被儲存在關聯式資料庫中的主檔資料內容，採用「基於分析資料庫綱要的方法」，分析資料庫的鍵值、屬性與資料的相關性、與它們之間的結合，來萃取可以表達關聯式資料庫之語意的關聯式綱要（Relational Schema）或概念綱要（Conceptual Schema），再轉換為本體論表達的格式，如 RDF 和 OWL，其中轉換流程為圖 2 所示之流程圖，按此流程將關聯式資料表轉換為本體論。

第二階段：本體論映對

將資料以本體論表達之後，接下來對這些轉換後的本體論進行整併。本體論整併是透過本體論整合工具「PROMPT」來生一個整合的本體論，依據本體論中語意重疊的程度與相似程度，該工具會建議使用者應合併之處，使用者也可自定義應合併之處。

第三階段：本體論連結

將整合後的本體論透過本體論查詢語法「SPARQL」對本體論做查詢，來得到所需的應用資訊。

圖 3 所示之轉換流程圖分為流程一、二、三（Process I、II、III），分別說明，並描述其概念、邏輯表示與演算法如下。

一、 流程一（Process I）

流程一的輸入為關聯式資料表，輸出為一元關係轉換規則（Unary Relationship Rule）

或基底關聯表轉換規則（Base Relation Rule），其中包含判別式 1（Decision 1）和判別式 2（Decision2）如下：

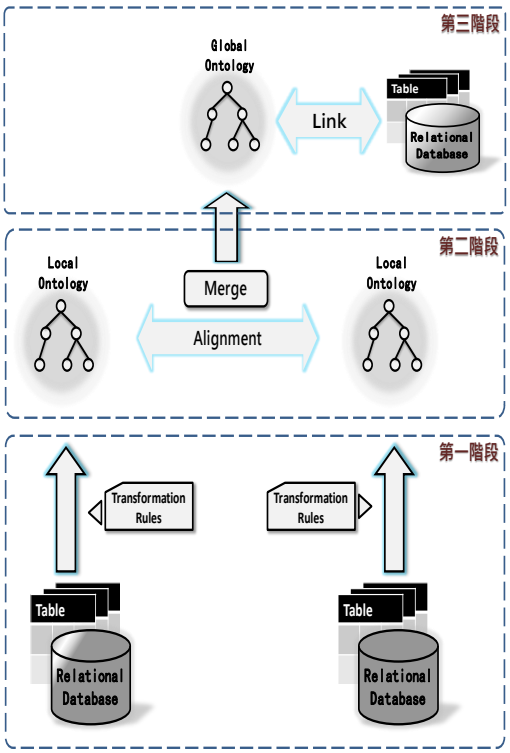

圖 2 以本體論為基礎之資料整合模式（資料來源：本研究整理）

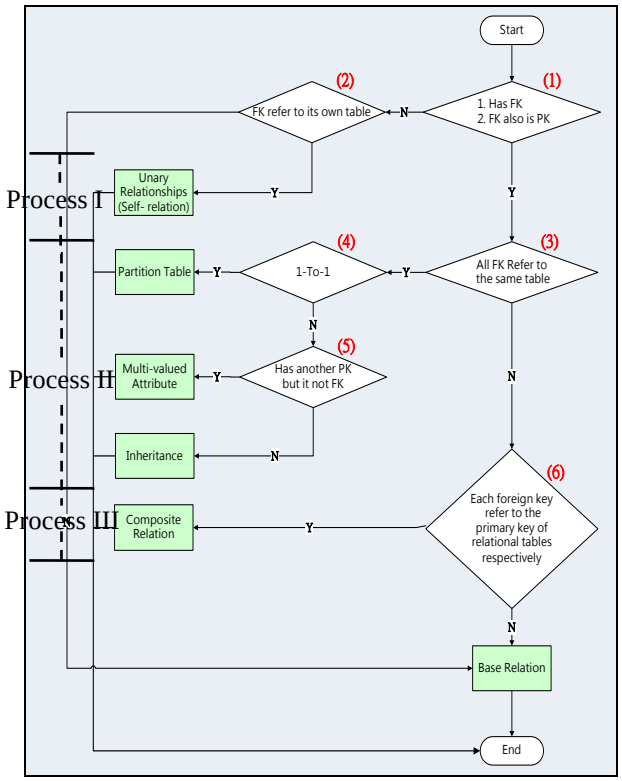

圖 3 關聯式資料庫轉換 OWL 本體論之流程圖（資料來源：本研究整理）

（一） 判別式 1（Decision 1）：

對於關聯表進行判別，該關聯表有外鍵（Foreign Key），或該關聯表有外鍵且同時是該關聯表的主鍵（Primary Key），若判別為真，則進入判別式 3；若判別為假，則進入判別式 2。虛擬碼如圖 4 所示。

```
decision 1
{   Begin
      If    Table ¡ has any   foreign key     Or
            Table ¡ has foreign key while this foreign key also is primary key
        Then   execute next decision 3
      Else   execute next decision 2
   End                                                              }
```

圖 4 判別式 1 之虛擬碼

（二） 判別式 2（Decision 2）：

對關聯表進行判斷，主鍵與外鍵不為同一屬性，且該外鍵參考到本身關聯表的主鍵，圖 5a 所示為一元關係的資料庫綱要圖，若判別為真，則關聯表轉換規則為一元關係規則；若判別為假，則關聯表轉換規則為基底關聯表規則。圖 5b 所示為判別式 2 之虛擬碼。

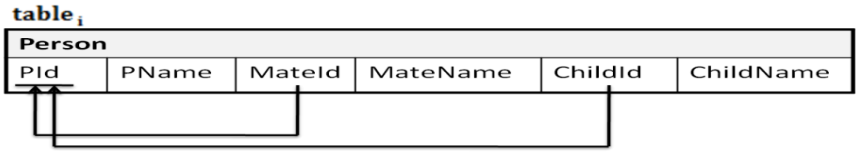

圖 5a 一元關係的資料庫綱要圖

```
decision 2
{   Begin
      If    Table i  has a primary key and a foreign key while disjoint each other    And
            foreign key of Table i refer to primary key of Table i
        Then   execute self-relation rule
      Else     execute base relation rule
   End                                                                              }
```

圖 5b 判別式 2 之虛擬碼

圖 6a 一元關係的實體關聯圖

```
self-relation rule
{   Begin
        create the class
                name of class = name of Table i     //   table transfer to class in OWL
   For   2   object property                                  //   create 2 object property
   {      create object property
          < owl:ObjectPropertyrdf：ID = name of relationship   >
   {   If    Cardinality =1
         Then     <rdf：type rdf:resource=”&owl；FunctionalProperty”/>          }
          < rdfs：domain   rdf：resource =   “# corresponding name of class”/ >
          < rdfs：range    rdf：resource =   “# corresponding name of class  “/ >
                //   in self-relation rule , the range and domain of object property are the same
      < owl：inverseOf   rdf：resource="# name of another object property"/ >     }
End                                                                                 }
```

圖 6b 一元關係規則之虛擬碼

（三） 一元關係規則（Unary Relationship Rule or Self-Relation Rule）

一元關係（Unary Relationship），指實體關聯模型中，該關係中參與的實體數目只有一個，且一元關係可能涉及一對一和一對多基數限制。例如圖 6a 為兩種基數限制的一元關係實體關聯圖，右邊為一對一基數限制的一元關係，人類同時只能有一個配偶；左邊為一對多基數限制的一元關係，人們可生產零到多個後代，但每個後代只會有一個生產者，就是親生母親。

一元關係轉換規則是直接將該關聯表轉為類別（Class），並以關聯表的名稱作為類別的名稱。接著，建立兩個逆向物件性質（Inverse Object Properties），而其中的定義域（Domain）與值域（Range）皆為該關聯表轉換後的類別，若關聯表的基數限制為 1，必須再額外添加「FunctionalProperty」的 OWL 語句到該基數限制為 1 的物件性質（Object Properties）中，最後再為兩物件性質建立逆向關係。一元關係規則之虛擬碼如圖 6b 所示。

（四） 基底關聯表規則（Base Relation Rule）

若關聯表無外鍵，或關聯表有外鍵但並不是該表的主鍵，則稱為基底關聯表（Base Relation），如圖 7a 基底關聯表的資料庫綱要圖所示，其轉換規則是直接將關聯表轉為類別，並以關聯表的名稱作為類別的名稱。所有關聯表的屬性皆轉為資料型別性質（Datatype Property），而資料型別性質中的值域（Range）轉換為與該屬性相關的 XML 資料型別（XML Data Type）；定義域（Domain）則是轉換為與該屬性相關的類別。圖 7b 所示為基底關聯表規則之虛擬碼。

圖 7a 基底關聯表的資料庫綱要圖

```
base relation rule   {
Begin
      create class     name of class = name of   Table i        // table transfer to class in OWL
   For ∀all attribute in Table i{                                 // create Datatype property
       < owl：DatatypePropertyrdf：ID =   “ name of attribute”  >
       < rdfs：domain   rdf：resource =   “# corresponding name of table”  />
       < rdfs：range    rdf：resource =   “&xsd; XML data type of attribute  “ />
       < /owl：DatatypeProperty>
                          //   each attribute in   Table i   transfer to Datatype property   in OWL
End                                                                                          }
```

圖 7b 基底關聯表規則之虛擬碼

二、 流程二（Process I I）

流程二的輸入為判別式 1（Decision 1）的輸出，輸出為多值關聯表轉換規則（Multi-Valued Relation Rule）、階層轉換規則（Hierarchy Rule）或分區表規則（Partitioning Of Table Rule），其中包含判別式 3（Decision3）、判別式 4（Decision4）和判別式 5（Decision5）如下：

（一） 判別式 3（Decision 3）：

判別式 3 主要判斷關聯表的主鍵是否完全只參考到同一張關聯表，如圖 8a 相依關聯表的資料庫綱要圖所示，若判別為真，則進入判別式 4；若判別為假，則進入判別式 6。圖 8b 所示為判別式 3 之虛擬碼。

圖 8a 相依關聯表的資料庫綱要圖

```
decision 3
{   Begin
      If   Table j has a primary key which also is a foreign key          And
           This foreign key of   Table j refer to primary key of   Table i
        Then   execute next decision rule 4
      Else   execute next decision rule 6
   End                                                                    }
```

圖 8b 判別式 3 之虛擬碼

（二） 判別式 4（Decision 4）：

判別式 4 用於判斷兩個各只包含一個主鍵的相依關聯表（判別式 3），且同時是一對一基數限制的關係，若判別為真，則兩關聯表轉換規則為分區表規則；若判別為假，則進入判別式 5。圖 9 所示為判別式 4 之虛擬碼。

```
decision 4
{   Begin
      If     the reference relationship between   Table i   to Table j   is   1          And
             the reference relationship between   Table j   to Table i   is   1          And
             Number of primary key in   Table i = 1                                     And
             Number of primary key in   Table j = 1
        Then   execute partitioning of table rule
      Else   execute next decision rule 5
   End                                                                              }
```

圖 9 判別式 4 之虛擬碼

（三） 判別式 5（Decision 5）：

判別式 5 再於分辨資料表是採用多值關聯表規則，還是階層規則，若判別為真，則兩個關聯表轉換規則為多值關聯表規則；若判別為假，則兩個關聯表轉換規則為階層規則。圖 10 所示為判別式 5 之虛擬碼。

```
decision 5
{   Begin
      If     Table j   has 2 primary key , one is foreign key ,another is not foreign key
        Then   execute multi-valued relation rule
      Else   execute inheritance rule
   End                                                                    }
```

圖 10 判別式 5 之虛擬碼

（四） 多值關聯表規則（Multi-Valued Relation Rule）

多值關聯表是指包含多值屬性的關聯表，多值屬性則是指該屬性有多個可能的值，若同一關聯表中包含多值屬性將違反資料庫正規化的第一階正規化，因此將其分割成兩個透過外鍵參考關係的關聯表，其中多值屬性與外鍵共為主鍵而成一個關聯表，圖 11a 所示為多值關聯表的資料庫綱要圖。

圖 11a 多值關聯表的資料庫綱要圖

多值關聯表其轉換規則是將兩個關聯表合併並轉為類別，另將兩個關聯表除了外鍵以外的所有屬性皆轉為資料型別性質（DatatypeProperty），而重複的主鍵整合成一個資料型別性質，而其中值域（Range）則是與該屬性相關的 XML 資料型別（XML Data Type）；定義域（Domain）則是與該屬性相關的 class。此規則不須表達物件性質，因為兩個關聯表已被整合成為一個本體論的類別。多值關聯表規則之虛擬碼如圖 11b 所示。

```
multi-valued relation rule
{   Begin     create the class
                    name of class = name of Table i              //   2 table transfer to 1 class in OWL
     For ∀ ((  all attribute in Table i and Table j  ) − (  the foreign key in Table j )) : i ≠ j
                /*   every attribute transfer to datatype property , two identical primary key become one   */
        {   create datatype property                                    //   create Datatype properties
          < owl：DatatypePropertyrdf：ID =    “ name of attribute”   >
          < rdfs：domain    rdf：resource =    “# corresponding name of class”   />
          < rdfs：range     rdf：resource =    “&xsd; XML data type of attribute   “ />
     /*   each attribute except the foreign key in Table i and Table j transfer to Datatype properties in OWL */
          <   /owl：DatatypeProperty>                                                    }
   End                                                                                    }
```

圖 11b 多值關聯表規則之虛擬碼

（五） 分區表規則（partitioning of table rule）

分區技術主要有兩種形式：垂直分區（Vertical Partitioning）與水平分區（Horizontal Partitioning）。

垂直分區表，透過對關聯表的垂直劃分，來減少關聯表的寬度，使某些特定且相關的列被劃分到特定的分區中，也就是把記錄分開儲存在多個分區表中，因此每個分區表的列都包含相同的行數。垂直分區表如圖 12b 所示。

水平分區表，這種形式分區是對關聯表的行來進行分區，也就是把記錄分割並儲存到不同的分區表中，因此每個分區表的行都包含相同的列數。所有在關聯表中定義的列在每個分區中都能找到，所以表的特性依然得以保持。

其轉換規則是將兩個關聯表合併並轉為類別，兩個關聯表除了外鍵以外的所有屬性

皆轉為資料型別性質（DatatypeProperty），重複的主鍵整合成一個資料型別性質，而其中值域(Range)則是與該屬性相關的 XML 資料型別(XML Data Type)；定義域(Domain)則是與該屬性相關的類別。此規則不須表達物件性質，因為兩個關聯表已被整合成為一個本體論的類別。圖 12c 所示為分區表規則之虛擬碼。

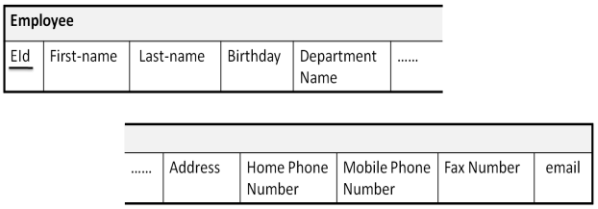

圖 12a Employee 關聯表

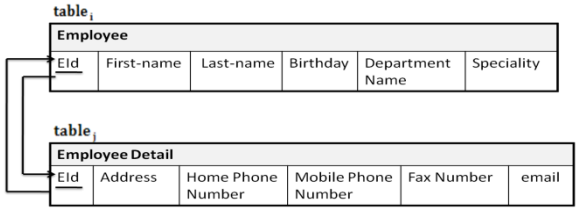

圖 12b 採用垂直分區的 Employee 關聯表

partitioning of table rule
{ Begin
create the class
name of class = name of Table i // 2 table transfer to 1 class in OWL
For ∀all attribute in Table i and Table j , except the foreign key in Table j：i≠j
/ every attribute transfer to datatype property , two identical primary key become one */*
{ create datatype property
< rdf：ID = name of attribute >
< rdfs：domain rdf：resource = "# corresponding name of table" >
< rdfs：range rdf：resource = "&xsd; XML data type of attribute " >
/ each attribute in Table i and Table j transfer to Datatype properties in OWL , and 2 primary key integrate to 1*/ }*
End }

圖 12c partitioning of table rule 之虛擬碼

（六） 階層規則（Hierarchy rule）

階層規則，也就是關聯表之間有「IS-A」繼承關係所使用的轉換規則，在增強型實體關係模型（Enhanced Entity-Relationship Model）或稱擴充實體關係模型（Extended Entity-Relationship Model）中，會在原本的實體關係模型上加入階層關係，以表達關聯表之間的繼承關係。圖 13a 所示為繼承關係的資料庫綱要圖。

圖 13a 繼承關係的資料庫綱要圖

關聯表間繼承關係的轉換規則是將兩個關聯表分別轉為類別，一個為父類別（Supclass），另一個為子類別（Subclass），且在子類別中加入「SubClass」的 OWL 本體論語言，而兩個關聯表除了外鍵以外的所有屬性皆轉為資料型別性質（Datatype

property），重複的主鍵整合成一個資料型別性質，其中值域（Range）則是與該屬性相關的 XML data type；定義域（Domain）則是與該屬性相關的類別。此規則不須表達物件性質（Object Property），因為所建立的 SubClass 已表達。圖 13b 所示階層規則之虛擬碼。

```
hierarchy rule
{   Begin
    For ∀   Table i , Table j ：i ≠j
    {     create the class              name of class = name of    Table (i and j)
                        /* 2 table transfer to 2 class in OWL , one is supclass and another one is subclass */
     If     Table i or Table j has foreign key
       Then     < rdfs : subClassOf rdf : resource = "# corresponding name of class" />
                                        //   when table is subtype, then add subclass sentence
     End   if
    Next                                                                    }
     For ∀all attribute in Table i and Table j , except the foreign key ：i ≠j
     {     create datatype properties                          //   create Datatype properties
     < owl ：DatatypePropertyrdf ：ID =    " name of attribute "   >
     < rdfs ：domain    rdf ：resource =  "# corresponding name of class" />
     < rdfs ：range     rdf ：resource = "&xsd; XML data type of attribute" />
     < /owl ：DatatypeProperty>
 // each attribute except the foreign key in Table i and Table j transfer to Datatype properties in OWL        }
   End                                                                      }
```

圖 13b 階層規則之虛擬碼

流程三（Process III）

流程三的輸入為判別式 3（Decision 3）的輸出，輸出為二元關聯表轉換規則（Binary Relation Rule）或基底關聯表轉換規則（Base Relation Rule），其中包含判別式 6（Decision 6）如下：

（一）　判別式 6（Decision 6）：

判別式 6 在判別關聯表是否為二元關係，若判別為真，則兩個關聯表轉換規則為二元關聯表規則；若判別為假，則兩個關聯表轉換規則為基底規則。圖 14a 所示為二元關係的資料庫綱要圖，圖 14b 所示為判別式 6 之虛擬碼。

上述之規則能透過分析資料庫綱要與資料表之間的關係，來判斷關聯式資料表的轉換規則，依循本研究發展之轉換規則演算法，可將關聯表轉換為本體論表示的形式（OWL），其中包括本體論的類別（Class）、物件性質（Object Properties）、資料型別性質（Datatype Property）、子類別（Subclass）關係、逆向（Inverse）關係與功能型性質（Functional Property）等本體論表示形式。透過本體論的表示方式，可以將資料表之間的語意表示出來，因此將資料庫轉換本體論後，資料將明確的定義、形式化、概念化與分享，再經過半自動化地整合與對齊，最後應用整合後的本體論，並查詢有用的資訊。

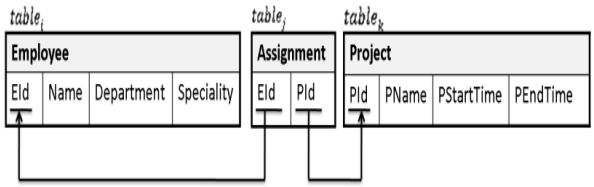

圖 14a 二元關係的資料庫綱要圖

decision 6

{ Begin

If Table $_j$ has 2 primary key which is also foreign key

Then execute binary relation rule

Else execute base relation rule

End }

圖 14b 判別式 6 之虛擬碼

肆、 模型驗證與展示

本研究以一跨國企業之訂單主檔來驗證本研究提出的資料整合模型，其中分為美國總公司與台灣分公司，圖 15a 與圖 15b 為該企業的美國總公司與台灣分公司之訂單主檔資料庫綱要模型。本研究使用 MySQL 為資料庫管理工具，來建立資料庫與關聯式資料表；使用 Protégé 本體論編輯工具，作為建立、整合與查詢本體論之平台。

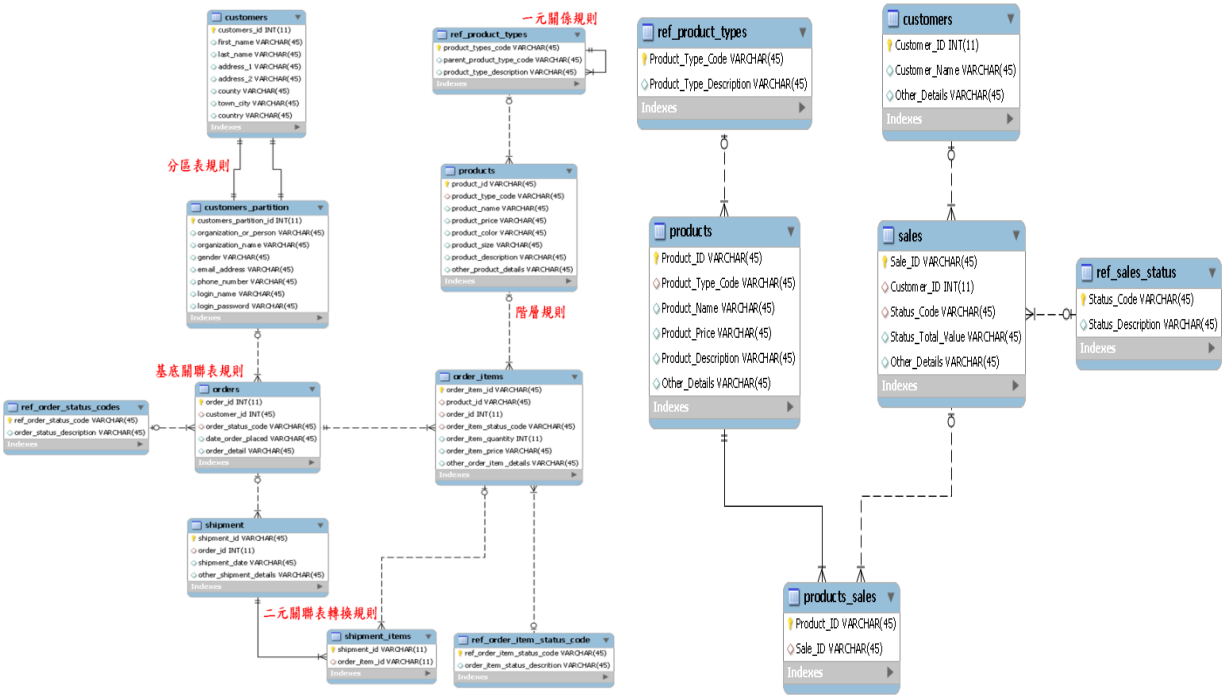

圖 15a 美國總公司之資料庫綱要模型

圖 15b 台灣分公司之資料庫綱要模型

一、 第一階段驗證：本體論建立

此階段將透過本研究發展之轉換規則來轉換關聯式資料表為本體論，輸入為美國總公司或台灣分公司之訂單主檔資料庫綱要模型，以本研究發展之轉換規則為處理流程，輸出則為美國總公司或台灣分公司之訂單主檔本體論。經過轉換規則的判斷，美國總公司之訂單主檔資料庫轉換會用到的轉換規則有一元關係規則、基底關聯表規則、分區表規則、階層規則與二元關係規則，接著將資料表裡的資料以本體論實例方式轉移到本體論中，轉換後的部分本體論如下圖 16a，其中類別名稱「Order_Items」與「Shipments」所顯示的就是二元關係規則所轉換後的本體論。台

灣分公司之訂單主檔資料庫轉換會用到的轉換規則有基底關聯表規則與二元關係規則，同樣也將資料表裡的資料以本體論實例方式轉移到本體論中，轉換後的部分本體論如下圖 16b。其中虛線鍵頭代表「物件性質」的關係，實線鍵頭代表「包含實例」的關係，表 1 和表 2 則分別說明圖中 A 和 B 部分。

表 1 本體論類別的說明

類別名稱(Class Name)	
資料型別性質 (Datatype Property)	值域
資料型別性質 (Datatype Property)	值域

表 2 本體論實例的說明

實例名稱(Instance Name)	
實例的資料型別性質 (Datatype Property Of Instance)	值域
實例的資料型別性質 (Datatype Property Of Instance)	值域

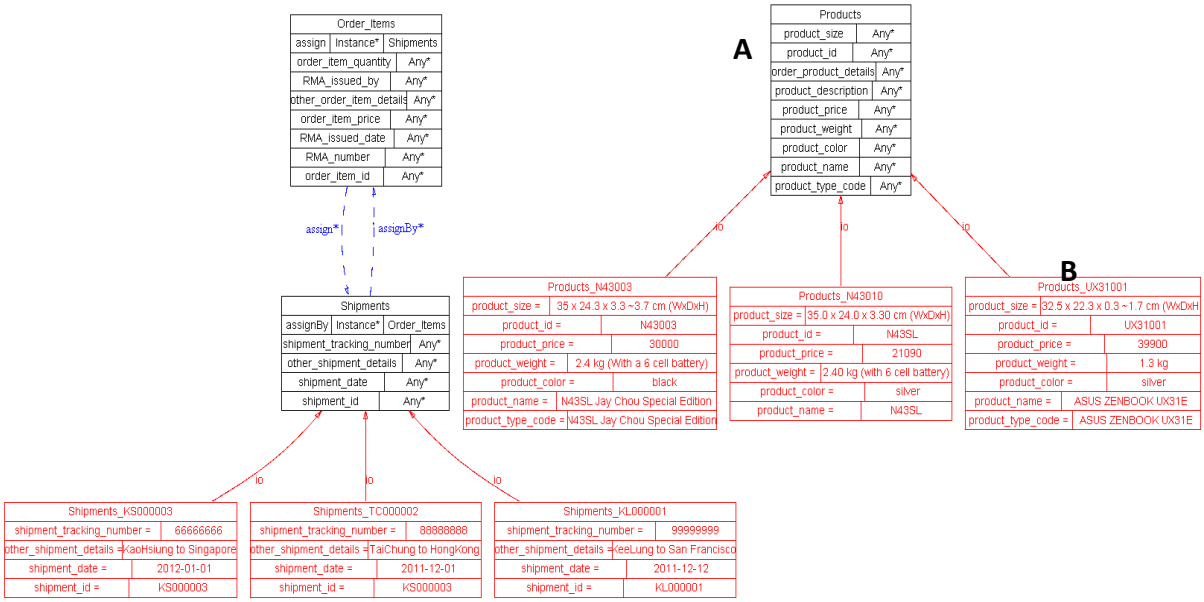

圖 16a 轉換後的部分本體論之概念圖(美國)

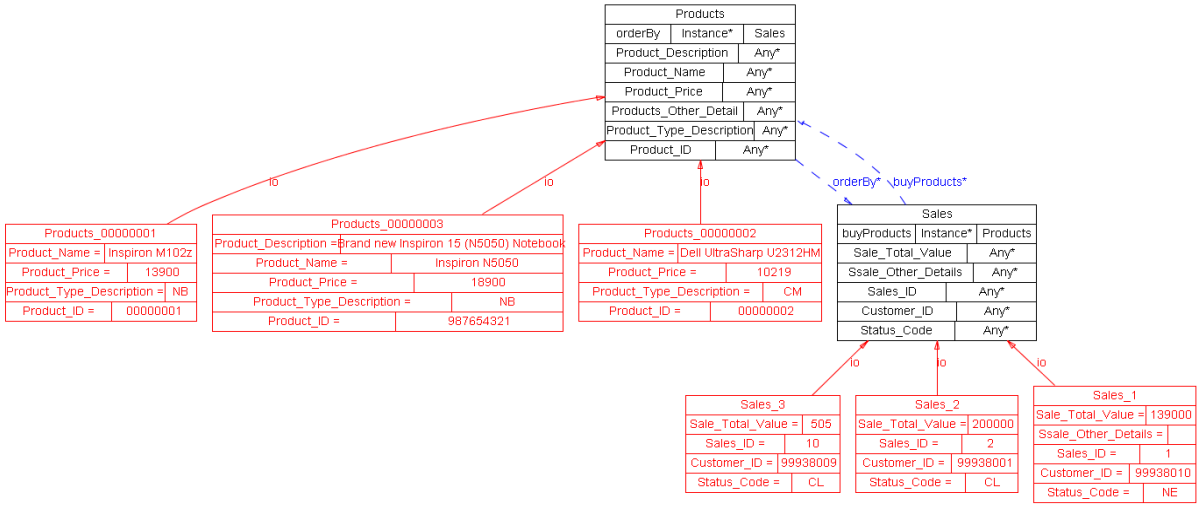

圖 16b 轉換後的部分本體論之概念圖(台灣)

二、 第二階段：本體論映對

此階段使用 Protégé 的外掛工具「PROMPT」來合併兩個本體論成為一個新的全

局本體論，該工具在輸入兩個本體論後，會建議應該合併或直接複製的類別、物件屬性與資料型別物件，也可以由使用者自己定義哪些該合併或直接複製到新的全局本體論中，如圖 16c7 所示。

三、　第三階段：本體論連結

此階段使用本體論查詢語言「SPARQL」來對整合後的全局本體論取得所需的資訊，如圖 16d 所示，其中左欄 Query 為查詢語句，右欄 Result 為查詢結果。例如可查詢住址相同但資料原本儲存在不同資料庫中的兩個顧客，其實是家人關係，如此可合併帳單或交叉行銷等。

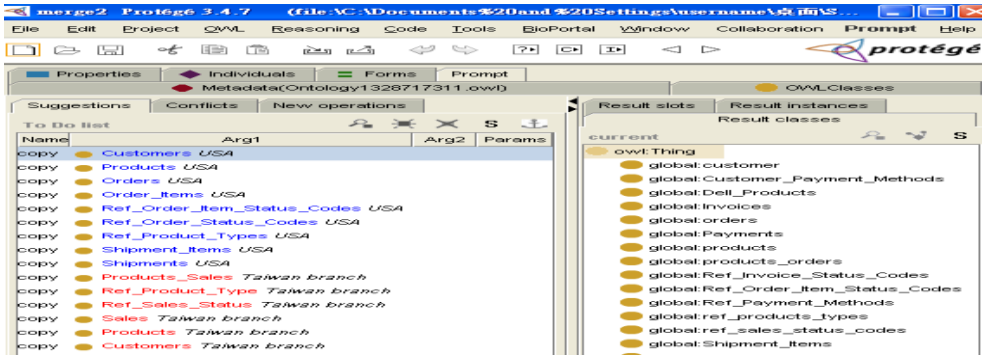

圖 16c 外掛工具 PROMPT 建議合併或複製類別到新的全局本體論

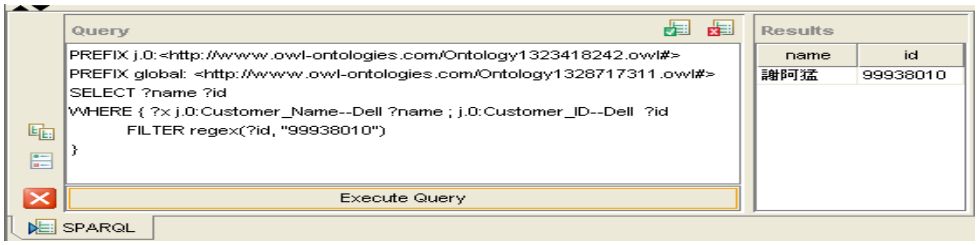

圖 16d 本體論查詢語言 SPARQL 之查詢

伍、　結論

本研究以逆向工程為基礎，進一步發展轉換關聯式資料表到本體論之轉換規則演算法，此演算法能透過分析資料表間關係來決定轉換的方式；另外本研究還提出以本體論為基礎之三階段資料整合模式，並以跨國企業的總公司和分公司之資料庫為範例，經過轉換、整併與查詢應用，驗證此模型實際用於企業的可行性。許多企業以資料為核心資產，而主檔資料尤其重要，因其經常被企業所使用，但不同系統擁有不同的資料庫，其中往往改了其中之一的資料，但並未更改另一系統中的資料，導致資料更新不同步，甚至「一步錯，步步錯」，造成「垃圾進，垃圾出」的錯誤情形發生。本研究發展一套以本體論為基礎之三階段資料整合模式，並將該模式套用在企業主檔資料管理之中，將異質系統中的主檔資料加入語意技術，以本體論形式定義與表示，透過三個階段的轉換、整合與應用，將資料表中的資料化為物件形式，不但改進傳統關聯式資料庫在查詢時，使用 JOIN 所耗費的大量資源與時間，還建立彈性與可擴展的主檔資料整合與管理，期望達到簡化企業流程、發揮與找出企業資料的最大價值與增加行銷機會，最後達到企業獲利。

參考文獻

1. 屠名正譯，Antoniou, G. andHarmelen, F.V. 著，《語意網技術導論 A Semantic Web Primer》，碁峰資訊股份有限公司，2006 年。
2. Alalwan, N.A., “Ontological Approach for Database Integration”, De Montfort University ,http://www.dora.dmu.ac.uk, 2011.
3. Astrova, I. “Reverse engineering of relational databases to ontologies”, lecture notes in computer science, Springer, vol 3053, pp 327–341, 2004.
4. Brunner, J-S., Ma, L., Wang, C., Zhang, L., Wolfson, D. C., Pan, Y., and Srinivas , K., “Explorations in the use of semantic web technologies for product information management”, Conference on World Wide Web, 2007.
5. Berners-Lee, T., Hendler, J. and Lassila, O., “The semantic web”, Scientific American, Vol.284, No.5,pp. 28-43 ,2001.
6. Behm A., Geppert A. and Dittrich K., “On the migration of relational database schemas and data to object-oriented database systems”, In Proceedings of the 5th International Conference on Re-Technologies for Information Systems, 1997.
7. Dreibelbis, A., Hechler, E., Milman I., Oberhofer, M., van Run, P. and Wolfson, D., “Enterprise Master Data Management: An SOA Approach to Managing Core Information”, Published by IBM Press, June 15, 2008.
8. Dorion, E. and Fortin, S., “Multi-Source Semantic Integration - Revisiting the Theory of Signs and Ontology Alignment Principles”, Information Fusion, 2007 10th International Conference on, July 2007.
9. Guarino, N., “Formal Ontology and Information Systems : proceedings of the first international conference （FOIS’98）”, Published by IOS Press , pp. 3-15, Trento, Italy, June 1998.
10. Lubyte, L. andTessaris, S., “Extracting ontologies from relational databases,” In Proc. of DL 2007, ser. CEUR Electronic Workshop Proceedings, http://ceur-ws.org/Vol-250/, vol. 250,, pp. 387–394, 2007.
11. Maedche, A., “An ERP-centric Master Data Management Approach”,Americas Conference on Information Systems, 2010.
12. McGuinness, D.L. and Harmelen, F.V., “OWL Web Ontology Language Overview”, Technical report, W3C, http://www.w3.org/TR/owl-features/, February 2004.
13. Ogden, C.K. and Richards, I.A., “The Meaning of Meaning: A Study of the Influence of Language Upon Thought and of the Science of Symbolism“, London: RoutledgeandKegan Paul, 1989.
14. Cleven,A., Wortmann,F., “Uncovering four strategies to approach master data management”, System Sciences (HICSS), 2010 43rd Hawaii International Conference on, 5-8 Jan. 2010.

A study of applying ontology on master data management model

Sung-Shun Weng
Graduate Institute of Information and Logistics Management
National Taipei University Technology
wengss@ntut.edu.tw

Chen-Shu Wang
Graduate Institute of Information and Logistics Management
National Taipei University Technology
wangcs@ntut.edu.tw

Meng-Yuan Hsieh
Graduate Institute of Information and Logistics Management
National Taipei University Technology
norm0412@gmail.com

Abstract

In recent years, the academia and industry is developing a semantic-oriented data management and data integration, if we use the ontology in data integration, that would be make information sharing easy, specification clearly and make the machine understandable. We would to achieve data integration efficient, information access accurately and make business flexibility, process simplify. There are so many subsidiaries in large enterprises or multinational corporations, including information systems independent of each other, there are own database respective, the enterprise resource planning system just only integrate the information for internal business units. But with the business expanding and needs increasing, providing data in holistic and single factual views to large enterprises can be avoided error that called "garbage in, garbage out". Our paper designs a ontology-based data integration model in three stage, and develops transformation rules algorithm that use reverse engineering to transform relational table to ontology. After transforming, we use the tool to ontology alignment and merging, and the use ontology query command to generate useful information. Our paper uses the three-stage ontology-based data management mode in the master data, because it is the most core data and be used commonly in the organization. The ontology-based data integration in master data not only makes data defined explicitly and formalization, but also integrates, shares and manages master data effectively.

Keyword：Ontology、Data Integration、Reverse Engineering、Master Data Management、Master Data