|  |
| --- |
| 2. CÁC MÔ HÌNH DỮ LIỆU |
| Một mô hình dữ liệu là một tập tích hợp các khái niệm nhận thức để mô tả và thao tác dữ liệu, các mối quan hệ giữa dữ liệu và các ràng buộc trên dữ liệu của một tổ chức. Một mô hình dữ liệu là một cách biểu diễn các đối tượng trong thế giới thực và các sự kiện cũng như các liên hệ giữa chúng. Nó là một khái niệm trừu tượng tập trung vào những khía cạnh cần thiết, sống còn của một tổ chức và cần bỏ qua những thuộc tính ngẫu nhiên. |
| 2.1. Giới thiệu các mô hình dữ liệu |
| Mô hình dữ liệu phải cung cấp các khái niệm cơ bản và các ký pháp cho phép người thiết kế cơ sở dữ liệu và người dùng trao đổi với nhau những hiểu biết về dữ liệu của tổ chức một cách chính xác và không đa nghĩa. Một mô hình dữ liệu có thể được cho rằng cấu thành bởi ba thành phần:   1. Thành phần cấu trúc bao gồm một tập các luật mà dựa trên đó các cơ sở dữ liệu được thiết lập. 2. Thành phần thao tác, định nghĩa các loại thao tác được phép thực hiện trên dữ liệu (bao gồm cả những thao tác được dùng để cập nhật hoặc thu thập dữ liệu từ cơ sở dữ liệu và cho việc thay đổi cấu trúc cơ ở dữ liệu) 3. Có thể có một tập các luật thể hiện tính toàn vẹn để đảm bảo rằng dữ liệu được lưu trữ trong cơ sở dữ liệu là hoàn toàn chính xác tuyệt đối.   Dựa vào kiến trúc trừu tượng ba lớp được trình bày ở trên, chúng ta có thể xác định ba loại mô hình dữ liệu khác nhau tương ứng.   1. Mô hình dữ liệu mức ngoài để biểu diễn khung nhìn của từng người dùng trong tổ chức. 2. Mô hình dữ liệu mức khái niệm để biểu diễn khung nhìn mức logic, độc lập với hệ quản trị cơ sở dữ liệu cụ thể. 3. Mô hình dữ liệu bên trong để biểu diễn lược đồ mức khái niệm theo một cách thức mà hệ quản trị cơ sở dữ liệu có thể hiểu được.   Nhiều loại mô hình dữ liệu khác nhau tồn tại, chúng đã được lý thuyết hóa, được sử dụng, được phát triển và được cài đặt qua nhiều năm. Chúng được phân chia thành ba loại chính: hướng đối tượng, hướng bản ghi và vật lý. Có nhiều mô hình dữ liệu hướng bản ghi bao gồm mô hình dữ liệu quan hệ, mô hình dữ liệu mạng, mô hình dữ liệu phân cấp. Chúng ta chỉ tập trung một mô hình dữ liệu quan hệ trong bài giảng này. Các mô hình dữ liệu ngữ nghĩa thường cố gắng thể hiện ý nghĩa của một cơ sở dữ liệu. Thực tế, chúng cung cấp một cách tiến cận tới việc mô hình hóa dữ liệu mức khái niệm. Qua nhiều nhiều năm, đã có một vài mô hình dữ liệu ngữ nghĩa khác nhau được đề xuất. Trong số những mô hình đó, mô hình dữ liệu thực thể liên kết được sử dụng phổ biến nhất, thường được biết đến với cái tên đơn giản là mô hình dữ liệu E-R. Mô hình E-R thường được sử dụng làm phương tiện trao đổi giữa những người thiết kế cơ sở dữ liệu và những người sử dụng cuối trong các quá trình phát triển một cơ sở dữ liệu. Nó bao gồm một tập các công cụ thiết lập mô hình mở rộng, mà một số trong đó chúng ta sẽ không tìm hiểu như mục tiêu chính của bài học. Mô hình này thể hiện cho chúng ta một bức tranh bên trong về thiết kế cơ sở dữ liệu khái niệm, chúng ta sẽ không nhắc đến các đầu vào và đầu ra của mô ình E-R. Một mô hình khái niệm khác cũng đang trở nên phổ biến hơn là ngôn ngữ định nghĩa đối tượng (ODL), một cách tiếp cận hướng đối tượng để thiết kế cơ sở dữ liệu, và đang phát triển thành một chuẩn cho các hệ thống cơ sở dữ liệu hướng đối tượng. |
| 2.2. Mô hình thực thể liên kết |
| Mô hình này sử dụng ba khái niệm cơ bản là: tập các thực thể, tập các liên kết (mối quan hệ) và các thuộc tính. Một thực thể là một điều gì đó hoặc một đối tượng nào đó trong thế giới thực mà có thể phân biệt được được với các đối tượng khác. Một thực thể có thể hiện thực như là một người hoặc một quyển sách hoặc có thể trừu tượng như một khoản vay từ nhà băng hoặc một khái niệm nào đó. Một thực thể được biểu diễn bởi một tập các thuộc tính. Các thuộc tính  là các thuộc tính mô tả hoặc các đặc điểm được sở hữu bởi một thực thể. Một tập các thực thể là một tập hợp các thực thể cùng loại, có chung các thuộc tính. Ví dụ, tập hợp tất cả các người là khách hàng của một ngân hàng nào đó có thể được định nghĩa như một tập hợp các khách hàng.   |  | | --- | |  | |  |   Các tập thực thể không nhất thiết phải không giao nhau. Ví dụ, chúng ta có thể định nghĩa tập các thực thể của tất cả những người là khách hàng của một ngân hàng. Một thực thể người nào đó có thể là một nhân viên, một khách hàng, hoặc cả hai hoặc không phải cả hai. Với mỗi một thuộc tính, tập các giá trị cho phép của thuộc tính đó được gọi là miền giá trị của thuộc tính (đôi khi còn được gọi là tập giá trị). Một cách hình thức hơn, một thuộc tính của một tập các thực thể là một hàm ánh xạ từ một tập các thực thể vào một miền giá trị. Vì một tập  thực thể có thể có vài thuộc tính, mỗi thực thể trong tập có thể được mô tả bởi một tập có dạng một cặp <thuộc tính, giá trị dữ liệu>, mỗi cặp này cho mỗi thuộc tính của tập thực thể. Một cơ sở dữ liệu bao gồm một tập các thực thể. |
| 2.2.1. Các ký pháp sử dụng trong mô hình E-R |
| Các ký pháp cho mô hình E-R được thể hiện trong hình vẽ dưới đây:   |  | | --- | |  | |  |  |  | | --- | |  | |  |  |  | | --- | |  | |  |  |  | | --- | |  | |  | |
| 2.2.2. Một số ví dụ về lược đồ E-R |
| Một số ví dụ về lược đồ E-R (ERD) được thể hiện trong hình vẽ dưới đây: Ví dụ 1:   |  | | --- | |  | |  |   Ví dụ 2:   |  | | --- | |  | |  | |
| 2.2.3. Các thuộc tính trong mô hình E-R |
| Như được sử dụng trong mô hình E-R, một thuộc tính có thể được phân chia thành bốn loại:   * Thuộc tính đơn hoặc thuộc tính ghép: một thuộc tính đơn không bao gồm các thành phần cấu thành trong khi các thuộc tính ghép bao gồm các phần con cấu thành. Ví   dụ, xem xét một thuộc tính tên. Nếu tên biểu diễn một thuộc tính đơn thì chúng ta có thể coi bộ ba cấu thành tên là họ, tên đệm và tên gọi là một thuộc tính nguyên tố, không phân chia được nữa. Mặt khác, nếu coi tên là một thuộc tính ghép thì chúng ta có lựa chọn thao tác với thuộc tính này là một tên đầy đủ hoặc có thể thao tác với từng thành phần cấu thành của tên. Ví dụ, chúng ta có thể chỉ thao tác với tên họ, cái mà không thể thực hiện với một thuộc tính đơn. * Thuộc tính đơn trị hoặc đa trị: Một thuộc tính đơn trị có thể có nhiều nhất một giá trị tại một thời điểm thể hiện cụ thể. Một thuộc tính đa trị có thể có nhiều giá trị khác nhau tại một thời điểm cụ thể. Ví dụ, xem xét một lớp học tại  Học viện công nghệ bưu chính viễn thông (giả sử học theo hệ thống tín chỉ, mỗi môn học được tổ chức thành một lớp trong một kỳ học nào đó). Tại một thời điểm nào đó, số sinh viên đăng ký học một lớp (theo hệ thống học tín chỉ) là đơn trị, tạm cho là nhận giá trị 100, nhưng không thể nhận giá trị đa trị vừa là 100, 80 và 45! Mặt khác, một vài thuộc tính có thể chứa nhiều giá trị cùng một lúc. Một ví dụ là thuộc tính miêu tả số điện thoại của một sinh viên, tại một thời điểm nào đó một sinh viên có thể có một vài số điện thoại khác nhau vì vậy thuộc tính đa trị là tốt nhất mô tả chính xác mô hình này. Thường thì hay thiết lập cả cận trên và cận dưới của các giá trị cho một thuộc tính tại một thời điểm. * Thuộc tính phái sinh: Đây là một thuộc tính mà giá trị của nó được phát sinh (hay được tính toán) từ các giá trị của các thuộc tính liên quan hoặc  các thực thể liên quan. Ví du, giả sử rằng thực thể khách hàng của một ngân hàng bao gồm một thuộc tính tên là loans-held, chứa số lượng các khoản vay của một khách hàng từ một ngân hàng. Giá trị của thuộc tính này có thể được tính bằng cách đếm số lượng thực thể các khoản vay liên quan tới mỗi khách hàng. * Thuộc tính rỗng: một thuộc tính nhận giá trị rỗng khi một thực thể không có giá trị cho nó. Giá trị rỗng thường là các trường hợp đặc biệt có thể kiểm soát theo các cách khác nhau dựa vào từng tình huống. Ví dụ có thể phiên dịch ngữ nghĩa của giá trị rỗng này như sau: thuộc tính không được áp dụng cho thực thể này hoặc thực thể này có giá trị cho thuộc tính đó nhưng chúng ta không biết giá trị đó. Chúng ta sẽ xem xét các hệ thống khách nhau xử lý các giá trị rỗng như thế nào và các  cách hiểu khác nhau liên quan tới giá trị đặc biệt này. |
| 2.2.4. Các liên kết trong mô hình E-R |
| Một mối quan hệ hay liên kết là một sự liên hệ giữa một vài thực thể. Ví dụ chúng ta có thể định nghĩa một mối liên kết thể hiện bạn là một sinh viên của một lớp học nào đó. Mối quan hệ này xác nhận rằng bạn đã đăng ký học lớp đó. Về mặt hình thức, một mối quan hệ là một tập các quan hệ cùng loại. Nó là một quan hệ toán học của N tập thực thể (có thể giao nhau, N>1). Nếu E1, E2, …, En là các tập thực thể thì một tập quan hệ *R* là một tập con của:{(e1,e2..en) | e1 \epsilonE1,e2 \epsilon E2 ... en \epsilon En} Với (e1,e2..en) là một quan hệ. Mối liên hệ giữa các tập thực thể được gọi là sự tham gia; có nghĩa là các tập thực thể E1, E2, …, En tham gia vào quan hệ R. Một thể hiện quan hê trong một lược đồ E-R biểu diễn một mối liên hệ giữa các thực thể được định danh trong một tập đoàn đang được mô hình hóa trong thực tế. Một mối quan hệ cũng có thể có các thuộc tính được gọi là các thuộc tính mô tả. Ví dụ, xem xét lại ngữ cảnh ngân hàng, giả sử rằng chúng ta có một tập các mối quan hệ tên là depositor (đặt cọc) với các tập thực thể customer (khách hàng) và account (tài khoản). Chúng ta có thể muốn liên hệ với mối quan hệ depositor một thuộc tính mô tả có tên là access-date (ngày truy nhập) để mô tả ngày gần nhất mà một khách hàng truy nhập vào tài khoản của họ. Như đã đề cập đến trước đây, một giá trị được lưu trữ trong một cơ sở dữ liệu nào đó thường có những ràng buộc được thiết lập để đảm bảo rằng chúng mô hình hóa toàn bộ thế giới thực của tập đoàn đang được thể hiện trong cơ sở dữ liệu một cách chính xác. Mô hình E-R có khả năng mô hình các loại ràng buộc này. Chúng ta sẽ tập trung vào hai loại ràng buộc quan trọng là ánh xạ lực lượng liên kết giữa các tập thực thể trong mối quan hệ và ràng buộc về số lượng thực thể tham gia vào mỗi quan hệ. |
| 2.2.5. Các ràng buộc ánh xạ lực lượng |
| Các ràng buộc ánh xạ lực lượng liên kết (mapping cardinality) trong mô hình E-R thể hiện số lượng các thực thể mà một thực thể khác có thể liên hệ với thông qua một tập quan hệ (liên kết). Ràng buộc này có ích nhất trong khi mô tả các mối quan hệ hai ngôi, mặc dù chúng có thể có ích trong mô tả các quan hệ liên quan tới nhiều hơn hai tập thực thể (quan hệ nhiều ngôi). Chúng ta sẽ chỉ tập trung vào các mối quan hệ hai ngôi tại thời điểm này. Với một tập quan hệ hai ngôi R giữa tập thực thể A và B, ánh xạ lực lượng sẽ rơi vào một trong các loại sau: |
| 2.2.6. Các ràng buộc tham gia trong mô hình E-R |
| Sự tham gia của một tập thực thể E trong một tập các mối quan hệ R được cho là đầy đủ nếu mọi thực thể của E tham gia vào ít nhất một mối quan hệ của R. Nếu chỉ có một vài thực thể của E tham gia vào một mối quan hệ trong R thì sự tham gia của tập thực thể E tới tập các mối quan hệ R được cho là một phần. Ví dụ lại xem xét hệ thống ngân hàng, chúng ta thấy rằng mối thực thể khoản vay có liên hệ tới ít nhất một khách hàng qua mối quan hệ borrower (vay mượn). Vì vây, sự tham gia của thực thể *loan* trong tập quan hệ borrower là đầy đủ. Ngược lại, một cá nhân có thể là một khách hàng của ngân hàng không liên quan tới chuyện họ có khoản vay ngân hàng này không. Vì vậy có thể nói rằng chỉ một số các thực thể customer (khách hàng) có liên hệ tới một thực thể *loan* thông qua mối quan hệ borrowers. Vì vậy, sự tham gia của thực thể customer trong mối quan hệ borrower là một phần. |
| 2.2.7. Khóa của một tập thực thể |
| Chúng ta phải có một cơ chế để phân biệt các thực thể trong một tập các thực thể nào đó. Về mặt khái niệm, mỗi thực thể đơn đều khác nhau, từ một khía cạnh một cơ sở dữ liệu, tuy nhiên, các sự khác nhau giữa chúng phải được thể hiện thông qua các thuộc tính của chúng. Vì vậy, các giá trị của thuộc tính của một thực thể phải được xác định sao cho chúng có thể xác định duy nhất thực thể đó. Nói một cách khác, không có hai thực thể nào trong một tập các thực thể được phép có giá trị của tất cả các thuộc tính là hoàn toàn giống nhau. Một khóa cho phép chúng ta xác định một tập các thuộc tính đủ để phân biệt các thực thể với nhau. Khóa cũng giúp cho việc xác định duy nhất các mối quan hệ và vì vậy phân biệt các mối quan hệ với nhau. Có một số khái niệm khóa bao gồm: Khóa chính, siêu khóa và khóa dự phòng. Một siêu khóa là một tập gồm một hoặc nhiều thuộc tính, được lựa chọn, cho phép chúng ta xác định duy nhất một thực thể trong một tập các thực thể. Giả sử rằng chúng ta có một tập thực thể mô hình các sinh viên trong một lớp học với lược đồ sau:  Students(SS#, name, address, age, major, minor, gpa, spring-sch)  Trong số các thuộc tính liên quan tới mỗi sinh viên, ta phải xác định một tập các thuộc tính xác định duy nhất mỗi sinh viên. Ta định nghĩa tập thuộc tính này là (SS#, name, major, minor). Tập các thuộc tính này sẽ xác định một siêu khóa cho tập thực thể Students. Lưu ý rằng tập các thuộc tính (SS#, name) cũng xác định một siêu khóa cho tập thực thể này bởi vì với bộ thuộc tính thứ hai này chúng ta vẫn có thể xác định duy nhất mỗi sinh viên trong tập thực thể. Khái niệm về một siêu khóa là một định nghĩa không đầy đủ của một khóa bởi vì siêu khóa này còn chứa nhiều thuộc tính dư thừa, như trong ví dụ vừa xét. Nếu một tập K là một siêu khóa của tập thực thể E thì mọi tập cha của K cũng là siêu khóa. Chúng ta chỉ quan tâm tới những siêu khóa thỏa mãn không có tập con nào của K là siêu khóa. Những siêu khóa nhỏ nhất kiểu như vậy được gọi là khóa dự bị (candidate key). Với mỗi một tập thực thể E cho trước, nhiều tập khác nhau các thuộc tính tồn tại, thỏa mãn là khóa dự bị. Cho dù có một khóa dự bị hay có nhiều khóa dự bị, người thiết kế cơ sở dữ liệu chỉ chọn một khóa dự bị làm khóa chính, khái niệm mà mọi người thường gọi là khóa của một tập thực thể. Một khóa (khóa chính, khóa dự bị hay siêu khóa) là một thuộc tính của tập thực thể chứ không phải của một thực thế nào đó. Bất kể hai thực thể khác nhau trong một tập thực thể sẽ không thể có cùng giá trị trên tất cả các thuộc tính cấu thành các thuộc tính khóa tại cùng một thời điểm. Ràng buộc này thể hiện trên các giá trị cho phép của một thực thể trong một tập được gọi là **ràng buộc khóa.** Người thiết kế cơ sở dữ liệu phải cẩn thận khi lựa chọn tập các thuộc tính cấu thành khóa của một tập thực thể để đảm bảo (1) chắn chắn rằng một tập các thuộc tính xác định duy nhất thực thể và (2) rằng tập thuộc tính khóa sẽ không bao giờ hoặc rất hiếm khi bị thay đổi. |
| 2.2.8. Tập các mối quan hệ |
| Khóa chính của một tập thực thể cho phép bạn phân biệt các thực thể khác nhau trong một tập. Chúng ta phải có một cơ chế tương tự để cho phép phân biệt các mối quan hệ trong một tập các mối quan hệ. Cho R là một tập các mối quan hệ liên quan tới các tập thực thể *E1, E2, …, En.* Gọi Ki là tập các thuộc tính cấu thành khóa chính của tập thực thể *Ei.* Chúng ta giả thiết rằng:   * Tên của tất cả các thuộc tính trong tất cả các khóa chính là duy nhất, điều này sẽ khiến cho ký pháp dễ hiểu hơn và trong thực tế, nếu các tên không duy nhất thì cũng không có vấn đề gì nghiêm trọng cả. * Mỗi tập thực thể sẽ chỉ tham dự một lần duy nhất trong mối quan hệ   Việc cấu thành khóa chính cho một tập các mối quan hệ phụ thuộc vào tập các thuộc tính liên quan tới tập các mối quan hệ R theo các cách sau đây:  Trong cả hai trường hợp, tập các thuộc tính *K1* \cup*K2*  \cup... \cup*Kn*  đều hình thành một siêu khóa cho tập các mối quan hệ.  Trong cả hai trường hợp, tập các thuộc tính *K1* \cup*K2*  \cup... \cup*Kn* đều hình thành một siêu khóa cho tập các mối quan hệ |
| 2.2.9. Một số vấn đề cần quan tâm khi thiết kế mô hình E-R |
| Ảnh hưởng của ràng buộc ánh xạ lực lượng liên kết lên các khóa Đặt vị trí cho các thuộc tính của mối quan hệ |
| 2.2.9.1. Ảnh hưởng của ràng buộc ánh xạ lực lượng liên kết lên các khóa |
| Cấu trúc của khóa chính cho tập các mối quan hệ phụ thuộc vào sự ánh xạ lực lượng liên kết. Xem xét trường hợp được thể hiện trong lược đồ sau đây:   |  | | --- | |  | |  |   Lược đồ E-R này thể hiện một quan hệ N-N cho quan hệ depositor với một thuộc tính access-date liên quan tới tập mối quan hệ giữa hai thực thể customer và account. Khóa chính của mối quan hệ này sẽ bao gồm hợp của các khóa chính của hai tập thực thể customer và account. Để làm rõ hơn nữa tình huống này, chúng ta xem xét đến hai lược đồ dữ liệu của hai tập thực thể này như sau:  Customer (customer-id, customer-name, address, city) Account (account-number, balance)  Một quan hệ N-N giữa hai tập thực thể này có nghĩa là một khách hàng có thể có nhiều tài khoản và tương tự như vậy một tài khoản có thể được quản lý bởi nhiều khách hàng. Để xác định duy nhất một mối quan hệ giữa hai thực thể trong customer và account, một phép hợp các khóa chính của cả hai tập thực thể cần thiết lập. Để nhìn thấy lần chuyển  tiền cuối cùng (last deposit) vào một tài khoản cụ thể nào đó, chúng ta cần xác định người chuyển tiền vì với tài khoản này có một số người có thể chuyển tiền vào. Lược đồ cho mối quan hệ depositor như sau:  Depositor (customer-id, account-number, access-date)  Bây giờ chúng ta xem xét trường hợp một khách hàng chỉ được phép chuyển tiền vào một tài khoản duy nhất. Điều đó có nghĩa là mối quan hệ depositor sẽ là quan hệ N-1 từ customer tới account  thể hiện trong hình vẽ dưới đây. Trong trường hợp này khóa chính của mối quan hệ depositor sẽ chỉ bao gồm khóa chính của tập thực thể customer. Để cho rõ hơn, xem xét lược đồ của các tập thực thể dưới đây:  Customer (customer-id, customer-name, address, city)  Account (account-number, balance)   |  | | --- | |  | |  |   Một mối quan hệ N-1 có nghĩa là một khách hàng nào đó chỉ có thể có duy nhất một tài khoản nên khóa chính của mối quan hệ depositor chỉ đơn giản là khóa chính của thực thể customer bởi vì với một khách hàng xác định họ chỉ có thể thực hiện một lần chuyển tiền gần nhất tới tài khoản duy nhất họ có thể truy nhập, vì thể việc chỉ ra số tài khoản cụ thể đó là không cần thiết để xác định một lần chuyển tiền duy nhất bởi người khách hàng đó. Lược đồ của tập các mối quan hệ depositor đó trong trường hợp này như sau:  Depositor (customer-id, access-date)  Bây giờ xét đến trường hợp mối quan hệ depositor này là N-1 từ account tới customer.   |  | | --- | |  | |  |   Một mối quan hệ N-1 từ account tới customer có nghĩa là mỗi tài khoản được làm chủ bởi nhiều nhất là một khách hàng nhưng mỗi khách hàng có thể có nhiều hơn một tài khoản. Trong tình huống này, khóa chính của mối quan hệ depositor chỉ đơn giản bao gồm khóa chính của thực thể account bởi vì chỉ có thể có nhiều nhất một lần chuyển tiền gần nhất tới một tài khoản nào đó xác định và chỉ có nhiều nhất một khách hàng có thể thực hiện việc chuyển tiền. Chúng ta không cần xác định duy nhất khách hàng nào đã thực hiện chuyển tiền vì chỉ có thể có một và chỉ một khách hàng có khả năng chuyển tiền vào một tài khoản xác định mà thôi. Lược đồ cho mối quan hệ depositor như sau:  Depositor (account-id, access-date) |
| 2.2.9.2. Vấn đề đặt vị trí cho các thuộc tính của mối quan hệ |
| Phần trước đã xem xét sự ảnh hưởng của lực lượng liên kết của một tập các mối quan hệ ảnh hưởng tới việc cấu thành một khóa chính cho tập các mối quan hệ đó. Tương tự trong phần này,chúng ta sẽ xem xét sự ảnh hưởng của yếu tố này tới việc đặt vị trí của các thuộc tính liên quan với mối quan hệ. Các thuộc tính của một tập quan hệ dạng 1-1 hoặc 1-N có thể đặt vào một trong các tập thực thể tham gia quan hệ liên kết, hơn là được đặt vào bản thân tập các mối quan hệ đó. Ví dụ với quan hệ depositor thể hiện trong hình vẽ   |  | | --- | |  | |  |   Thuộc tính access-date có thể được đặt liên quan tới thực thể acccount mà không làm tổn thất thông tin như hình vẽ dưới đây. Sở dĩ làm được như vậy bởi vì một tài khoản cụ thể nào đó có thể thuộc sở hữu của nhiều nhất là một khách hàng, và tài khoản đó có thể có nhiều nhất một access-date, ngày được lưu trữ trong account.   |  | | --- | |  | |  |   Tiếp đến xét đến trường hợp sau đây:   |  | | --- | |  | |  |   Thuộc tính access-date có thể gắn vào hoặc thực thể customer hoặc tập thực thể account mà không làm tổn thất thông tin. Trong tình huống này một tài khoản cụ thể nào đó có thể được làm chủ bởi nhiều nhất một khách hàng và một khách hàng nào đó có thể sở hữu duy nhất một tài khoản. Vì vậy, nếu thuộc tính access-date được lưu trữ với tập customer thì nó phải tham chiếu tới lần truy nhập cuối cùng của khách hàng đó tới tài khoản duy nhất mà họ có. Tương tự, nếu thuộc tính access-date được lưu trữ trong thực thể account thì nó sẽ tham chiếu tới lần truy nhập cuối cùng  tới tài khoản đó bởi người khách hàng duy nhất sở hữu nó. Vì vậy, một trong hai lược đồ sau đây đều đúng trong tình huống này.   |  | | --- | |  | |  | |  | |  |   Khi tập các mối quan hệ này có ràng buộc N-N, tình huống này sẽ rõ ràng hơn nhiều. Tình huống được thể hiện trong hình vẽ dưới đây:   |  | | --- | |  | |  |   Một tài khoản có thể được sở hữu bởi nhiều khách hàng, chúng ta thấy rằng gắn thuộc tính access-date với tập thực thể tham gia nào cũng không mô hình hóa được tình huống này mà không làm tổn thất thông tin. Nếu chúng ta cần lưu trữ ngày truy nhập cuối cùng của một khách hàng cụ thể nào đó tới một tài khoản cụ thể thì thuộc tính access-date này nhất thiết ohair là một thuộc tính của tập mối quan hệ depositor, chứ không thể là thuộc tính của bất kể tập thực thể tham gia nào. Ví dụ, nếu access-date là một thuộc tính của account thì chúng ta không thể xác định được khách hàng nào đã thực hiện việc chuyển tiền vào tài khoản đó. Còn nếu access-date là một thuộc tính của customer, chúng ta cũng không thể xác định được tài khoản nào khách hàng đã truy nhập vào lần cuối. |
| 2.2.9.3. Các vấn đề thiết kế khác |
| Khái niệmcủa một tập thực thể và một tập các mối quan hệ là chưa chính xác. Chúng ta có thể định nghĩa một tập các thực thể và các mối quan hệ giữa chúng theo nhiều cách khác nhau. Ta sẽ cùng xem qua một số cách tiếp cận khác nhau để mô hình hóa dữ liệu. Trong một phạm vi nào đó, đây là chỗ “nghệ thuật” thiết kế cơ sở dữ liệu thể hiện sự hiểm hóc của nó. Đôi khi một số kịch bản thiết kế khác nhau nhìn qua có vẻ như giống nhau và thậm chí  cải thiện và chuẩn hóa vẫn phù hợp để tiến hành nhưng đôi khi không còn giống nhau nữa. Chỉ một thiết kế cẩn thận mới có thể loại bỏ những vấn đề mà chúng ta sẽ bàn tới sau đây.  **Vấn đề tập thực thể hay các thuộc tính** Xem xét một tập thực thể  *Employee(emp-name, telephone-number, age)*  Việc một máy điện thoại sẽ được coi là một thực thể (với các thuộc tính T*elephone-number, location, manufacturer, serial-num*, và một số thuộc tính khác nữa) sẽ dễ dàng gây ra tranh cãi. Trong tinh huống này thì thực thể Employee phải được định nghĩa lại như sau: *Employee (emp-name, age).*Sau đó phải tạo ra một tập thực thể mới *Telephone(telephone-number, location, manufacturer, serial-num,…)* và một tập các mối quan hệ phải được tạo ra để xác định mối liên hệ giữa các nhân viên và các máy điện thoại mà họ sở hữu *Emp-Phone(emp-name, telephone-number, age, location, manufacturer, serial-num).*  Bây giờ chúng ta. coi máy điện thoại như một thuộc tính *telephone-number* ngụ ý rằng mỗi nhân viên sẽ có một số điện thoại duy nhất (chú ý rằng điều này là đúng đắn nếu không thuộc tính số điện thoại sẽ phải là một phần của khóa chính cho mỗi nhân viên và ở đây không đề cập tới các thuộc tính đa trị). Việc coi máy điện thoại như một thực thể cho phép các nhân viên có thể sở hữu vài cái điện thoại (bao gồm cả 0 điện thoại). Tuy nhiên, chúng ta có thể dễ dàng chuyển thuộc tính telephone-number thành thuộc tính đa trị cho phép một nhân viên có nhiều máy điện thoại. Vì thế rõ ràng đây không phải là điểm khác biệt chính trong hai cách biểu diễn này. Sự khác nhau chính ở đây là việc coi máy điện thoại như một thực thể mô hình hóa tình huống này tốt hơn vì như vậy chúng ta có thể muốn lưu trữ thêm thông tin về máy điện thoại đó như được thể hiện trong ví dụ trên. Nếu chúng ta sử dụng cách tiếp cận ban đầu và muốn biến điện thoại thành một thuộc tính của một nhân viên và chúng ta muốn duy trì những thông tin thêm này về máy điện thoại của họ thì thực thể Employee sẽ có lược đồ mới như sau:  Employee(emp-name, telephone-number, age, location, manufacturer,…)  Đây không phải là một lược đồ tốt vì thuộc tính age (tuổi) trong lược đồ trên liệu liên quan tới employee (nhân viên) hay tới telephone (máy điện thoại)? Trong tình huống này chúng ta đang cố gắng mô hình hóa hai tập thực thể khác nhau trong cùng một tập thực thể. Ngược lại, nếu coi thuộc tính emp-name như một thực thể thì lại không thích hợp; khó có thể cho rằng tên của một nhân viên lại là một thực thể . Vì vậy, việc coi tên của một nhân viên emp\_name như một thuộc tính của tập thực thể Employee là hoàn toàn thích hợp. Vậy cái gì nên coi là một thuộc tính và cái gì nên coi là một thực thể? Thật không may, vấn đề này không có một câu trả lời đơn giản. Sự phân biệt giữa hai vai trò này phụ thuộc chính vào cấu trúc của ngữ cảnh trong thế giới thực cần mô hình hóa dữ liệu và dựa vào ngữ nghĩa liên quan tới các thuộc tính trong bài toán. Một lỗi chung thường gặp là việc sử dụng khóa chính của một tập thực thể như một thuộc tính của một thực thể khác để thể hiện mối liên kết giữa hai thực thể thay vì sử dụng một mối quan hệ. Quay lại ví dụ về ngân hàng, việc coi customer\_id là một thuộc tính của thực thể loan là không hợp lý thậm chí ngay cả khi mỗi loan chỉ có duy nhất một khách hàng liên quan tới nó. Quan hệ borrower là một cách đúng đắn để biểu diễn mối quan hệ giữa loan  và customer vì nó sẽ tạo liên kết một cách tường minh hơn là cách thể hiện không tường minh thông qua một thuộc tính.  **Coi một đối tượng là một tập thực thể hay tập mối quan hệ** Việc thể hiện một một đối tượng tốt nhất dưới dạng một tập thực thể hay một tập mối quan hệ thường không rõ ràng. Xem xét ví dụ về ngân hàng. Chúng ta đã mô hình một khoản vay như một thực thể. Một lựa chọn khác là mô hình khoản vay loan như một mối quan hệ giữa khách hàng và các chi nhánh branches của ngân hàng với loan-number và amount là các thuộc tính mô tả. Mỗi khoản vay được biểu diễn như một quan hệ giữa một khách hàng và một chi nhánh ngân hàng. Nếu mỗi khoản vay chỉ được sở hữu bởi duy nhất một khách hàng và có liên hệ tới duy nhất một chi nhánh ngân hàng thì chúng ta có thể mô hình khoản vay là một mối quan hệ. Tuy nhiên, với thiết kế kiểu này chúng ta không thể biểu diễn thuận tiện cho tình huống mà nhiều khách hàng có thể cùng chung sở hữu một khoản vay. Để giải quyết tình huống này, chúng ta cần định nghĩa một mối quan hệ riêng cho mỗi người sở hữu khoản vay chung. Sau đó chúng ta cần dùng lại tất cả các giá trị cho các thuộc tính loan-number và amount trong mỗi mối quan hệ loại này. Hiển nhiên mỗi quan hệ đó phải có cùng giá trị cho các thuộc tính mô tả. Hai vấn đề phát sinh như kết quả của việc dùng lặp lại các giá trị là:   1. Dữ liệu được lưu trữ ở nhiều nơi (đúng nghĩa của khái niệm dùng lặp lại) 2. Các câp nhật nhiều khả năng làm dữ liệu trong một trạng thái không đồng nhất nơi mà các giá trị trong hai tập khác nhau không có giá trị giống nhau mặc dù chúng phải giống hệt nhau. Chúng ta sẽ xem nhưng phức tạp mà việc dùng lặp lại giá trị này gây ra cũng như các kỹ thuật giải pháp (lý thuyết chuẩn hóa) sau đó trong bài giảng này. Chú ý là vấn đề này không xuất hiện trong phiên bản ban đầu bởi vì loan được biểu diễn bởi một tập thực thể trong trường hợp này.   Một hướng dẫn có thể có trong việc quyết định nên sử dụng một tập thực thể hay một tập các mối quan hệ là cần thiết kế một tập thực thể để mô tả một hành động xảy ra giữa các thực thể. Cách tiếp cận này cũng có ích trong việc quyết định các thuộc tính nào đó có thể thích hợp hơn với cách biểu diễn như là mối quan hệ.  **Coi một đối tượng là tập thực thể yếu hay tập thực thể mạnh** Một ví dụ về khái niệm này: cân nhắc một thực thể payment, có ba thuộc tính: payment-number, payment date và payment amount. Payment number thường là các số liên tiếp, bắt đầu từ 1 và được sinh ra riêng rẽ cho mỗi khoản nợ, Vì vậy, mặc dù mỗi thực thể payment là khác nhau, việc trả tiền cho các khoản nợ khác nhau có thể chung cùng một payment number (mã số trả tiền), vì vậy tập này không có một khóa chính và chỉ là một tập thực thể yếu. Để một tập thực thể yếu trở nên có ý nghĩa, nó phải liên hệ với một tập thực thể khác được gọi là tập thực thể xác định hay tập thực thể sở hữu. Mỗi thực thể yếu phải được liên quan tới một tập thực thể xác định như vậy. Thực thể yếu được nói là tồn tại phụ thuộc vào tập thực thể xác định. Tập xác định này được gọi là tập sở hữu thực thể yếu mà nó xác định. Mối quan hệ liên kết tập thực thể yếu với tập thực thể xác định được gọi là mối quan hệ xác định. Mối quan hệ này là quan hệ N-1 từ tập thực thể yếu tới tập xác định và sự tham gia của tập thực thể yếu trong quan hệ này là đầy đủ. Mặc dù một tập thực thể yếu không có khóa chính, nhưng chúng ta cần một phương thức để phân biệt tất cả các thực thể này trong một tập thực thể yếu mà phụ thuộc vào một thực thể khỏe mạnh cụ thể nào đó. Tập các thuộc tính của một thực thể yếu cho phép phân biệt các thực thể được gọi là thuộc tính phân biệt (đôi khi được gọi là khóa bán phần). Ví dụ, thuộc tính phân biệt của tập thực thể yếu payment ở trên là thuộc tính payment-number, vì với mỗi khoản nợ,một payment number xác định duy nhất một việc trả tiền riêng biệt cho khoản nợ này. Khóa chính của một tập thực thể yếu được cấu thành bởi khóa chính của tập thực thể xác định công với thuộc tính phân biệt của tập thực thể yếu. Với trường hợp ở trên, khóa chính của tập thực thể payment sẽ là {loan-number, payment-number}, trong đó loan-number là khóa chính của tập thực thể xác định và payment-number là thuộc tính phân biệt của tập thực thể yếu payment. Trong một lược đồ E-R, một tập thực thể yếu được biểu diễn bằng một hình chữ nhật với đường viền kép và quan hệ xác định cho một tập thực thể yếu được biểu diễn bằng một hình thoi có đường viền kép. Một ví dụ về tập thực thể yếu được thể hiện trong hình vẽ dưới đây   |  | | --- | |  | |  | |
| 2.2.10. Mô hình thực thể liên kết mở rộng |
| Một số đặc tính của tình huống trong thế giới thực có thể khó mô hình hóa nếu chỉ sử dụng những đặc tính của mô hình E-R đã được xem xét từ trước đến nay. Một số khái niệm chung đòi hỏi phải mở rộng mô hình E-R để thực hiện cơ chế mô hình hóa cho các đặc tính này. Một lần nữa, chúng ta sẽ không bàn luận đến tất cả chúng mà chỉ xem xét mức tổng quan về một số các đặc tính mở rộng quan trọng hơn trong chúng. |
| 2.2.10.1. Cụ thể hóa |
| Một tập thực thể bao gồm các phân nhóm thực thể mà được phân biệt với các thực thể khác theo một cách nào đó. Cụ thể hơn, một tập con các thực thể trong một tập thực thể có thể có các thuộc tính mà không chung với các thực thể khác trong tập thực thể đó. Một ví dụ minh họa, xem xét một tập thực thể *person* (người) với các thuộc tính *name, street, và city.* Một person có thể được phân chia nhỏ hơn thành một trong những loại sau: *student* (sinh viên) hoặc *instructor* (giảng viên). Mỗi loại người này được mô tả bởi một tập các thuộc tính bao gồm tất cả các thuộc tính của tập thực thể *person*, cộng với một vài các thuộc tính bổ sung thêm vào nữa. Thực thể *student* có thể mô tả thêm bởi thuộc tính *gpa (điểm trung bình)*, và *credit-hours-earned(số lượng tín chỉ đã tíchl*  trong khi đó thực thể *instructor* không được thể hiện bằng các thuộc tính này mà bằng một tập thuộc tính khác như *salary (lương)* và *years-employed (số năm làm việc)*. Quá trình thiết kế các phân nhóm trong một tập thực thể được gọi là quá trình cụ thể hóa. Việc cụ thể hóa của thực thể *person* cho phép chúng ta phân biệt giữa các loại người theo tiêu chí họ là sinh viên hay là giảng viên. Việc cụ thể hóa này có thể được lặp đi lặp lại nhiều lần vì vậy có thể có cụ thể hóa trong một quá trình cụ thể hóa khác. Trong phạm vi một lược đồ E-R, việc cụ thể hóa được biểu diễn bởi một hình tam giác với nhãn là ISA (là một), là một cách viết tắt của quan hệ “là một” (“is-a”) giữa nhóm con và nhóm cha. Mối quan hệ ISA được mô tả trong lược đồ như trong hình vẽ sau đây:   |  | | --- | |  | |  | |
| 2.2.10.2. Tổng quát hóa |
| Việc phát triển từ một tập thực thể ban đầu tới các mức tiếp theo là các phân nhóm các thực thể thể hiện các tiếp cận thiết kế từ trên-xuống trong đó các sự khác biệt được thể hiện tường minh. Quá trình thiết kế cùng một bài toán như vậy có thể được tiến hành theo cách tiếp cận từ dưới lên, trong đó nhiều tập thực thể được đồng bộ vào một thực thể ở mức cao hơn trên cơ sở các thuộc tính chung. Nói một cách khác, đầu tiên chúng ta có thể xác định các thực thể s*tudents(name, address, city, gpa, credit-hours-earned)* và một tập thực thể *instructors(name, address, city, salary, years-employed)*. Việc tìm điểm chung giữa các thuộc tính được gọi là tổng quát hóa, nó là một quan hệ tồn tại giữa một tập thực thể ở mức cao hơn với một hoặc nhiều tập thực thể ở mức thấp hơn. Trong ví dụ chúng ta đang xét, *person* là tập thực thể ở mức cao và *instructor* và *student* là các tập thực thể ở mức thấp. Thực thể ở mức cao biểu diễn một lớp cha và thực thể ở mức thấp biểu diễn một lớp con, Vì vậy, *person* là một lớp cha của lớp con *instructor* và *student*. Cho tất cả các mục đích thực tế, việc tổng quát hóa chỉ là một khái niệm ngược với việc cụ thể hóa và cả hai quá trình này đều có thể được dùng (hầu như là có thể hoán đổi cho nhau được) trong việc thiết kế các lược đồ cho một kịch bản có thực. Lưu ý trong lược đồ E-R, không thấy có sự khác biệt giữa tổng quát hóa và cụ thể hóa mà chỉ khác nhau ở cách bạn xem hình vẽ đó từ trên xuống hay từ dưới lên. Sự khác nhau của hai cách tiếp cận này thường được thể hiện bởi những điểm bắt đầu và mục đích chung của chúng: Cụ thể hóa xuất phát từ một tập thực thể riêng biệt; nó nhấn mạnh sự khác nhau giữa các thực thể trong cùng một tập bằng cách tạo ra các tập thực thể phân biệt nhau ở mức thấp hơn. Những tập thực thể mức thấp này có thể có các thuộc tính hoặc tham gia vào các mối quan hệ mà không áp dụng đối với tất cả các thực thể trong tập thực thể ở mức cao hơn. Trên thực tế, lý do mà một người thiết kế có thể cần để sử dụng quá trình cụ thể hóa là để thể hiện những đặc tính riêng của kịch bản trong thế giới thực. Ví dụ, nếu *instructor* và *student* không có các thuộc tính mà thực thể *person* có hoặc không tham gia vào các quan hệ  mà *person*  tham gia thì sẽ không cần phải thực hiện việc cụ thể hóa.  Tổng quát hóa xuất phát từ việc nhận ra rằng một số các tập thực thể có chung một số các đặc tính (về tên gọi, chúng được mô tả bởi cùng các thuộc tính và tham gia vào cùng các tập mối quan hệ). Trên cơ sở tính chung, tổng quát hóa tổng hợp những tập thực thể này thành một tập thực thể ở mức cao hơn. Quá trình tổng quát hóa được sử dụng để nhấn mạng tính tương đồng giữa các tập thực thể ở mức thấp hơn và giấu đi những sự khác nhau. Nó cũng cho phép một sự thể hiện kinh tế trong đó các thuộc tính chung không bị lặp lại. |
| 2.2.10.3. Sự kế thừa thuộc tính |
| Một tính chất không thể thiếu được của các thực thể mức thấp hơn, những thực thể được tạo ra bởi sự cụ thể hóa hoặc tổng quát hóa, là sự kế thừa thuộc tính. Thuộc tính của các tập thực thể ở mức cao hơn được nói là kế thừa bởi các tập thực thể ở mức thấp hơn. Trong ví dụ đã xét ở trên, *instructor* và *student*  đều kế thừa tất cả các thuộc tính của *person*. Một tập thực thể mức thấp hơn cũng kế thừa sự tham gia trong các tập mối quan hệ trong đó các tập thực thể ở mức cao hơn tham gia. Nó cũng kế thừa tất cả các thuộc tính và các mối quan hệ thuộc về tập thực thể mức cao hơn định nghĩa nó. Các tập thực thể ở mức cao sẽ không kế thừa bất kỳ một thuộc tính nào hoặc mối quan hệ nào được định nghĩa trong tập thực thể ở mức thấp hơn. Thông thường, một phân cấp các tập thực thể sẽ được phát triển trong đó thực thể ở mức cao nhất xuất hiện trên đỉnh của phân cấp. Nếu, trong một phân cấp như vậy, một tập thực thể nào đó có thể liên quan tới với vai trò một thực thể ở mức thấp hơn trong một mối quan hệ ISA thì sự kế thừa được gọi là kế thừa đơn. Mặt khác, nếu một thực thể nào đó liên quan như một tập thực thể mức thấp  trong nhiểu hơn một mối quan hệ ISA thì sự kế thừa đó được coi là đa kế thừa. |
| 2.2.10.4. Các ràng buộc trên việc tổng quát hóa |
| Để mô hình hóa các tình huống trong thực tế chính xác hơn, người thiết kế dữ liệu có thể thiết lập các ràng buộc trên một tổng quát hóa (hoặc cụ thể hóa). Loại đầu tiên của ràng buộc liên quan tới việc xác định các thực thể nào có thể là thành viên của tập thực thể mức thấp hơn. Thành viên này có thể được định nghĩa theo một trong hai cách sau đây:   * **Mệnh đề xác định**: Trong loại này, thành viên được đánh giá trên cơ sở xác định xem một thực thể có thỏa mãn một mệnh đề (điều kiện) tường minh nào đó không. Ví dụ, giả sử rằng tập thực thể ở mức cao account có thuộc tính account-type. Tất cả các thuộc tính được đánh giá dựa trên việc định nghĩa thuộc tính account-type. Chỉ những thực thể thỏa mãn mệnh đề account-type = “savings account” sẽ được phép thuộc vào tập thự thể mức thấp hơn savings-account. Vì tất cả các thực thể mức thấp hơn được đánh giá dựa trên cùng thuộc tính, loại tổng quát hóa này được gọi là thuộc tính xác định. * **Người dùng xác định**: Tập các thực thể mức thấp do người dùng xác định không bị ràng buộc bởi các điều kiện thành viên mà người dùng cơ sở dữ liệu gán các thực thể tới các tập thực thể nào đó. Ví dụ: giả sử sau 3 tháng làm việc tại một ngân hàng, một nhân viên được gán tới một trong năm nhóm khác nhau. Các đội sẽ được biểu diễn như năm tập thực thể mức thấp hơn của tập thực thể mức cao hơn là Employee. Một nhân viên nào đó sẽ không được gán tới một nhóm làm việc cụ thể nào đó  một cách tự động trên cơ sở một việc định nghĩa một điều kiện tường minh. Thay vào đó, người sử dụng có trách nhiệm tạo ra các nhóm dựa trên từng cá nhân các thành viên, có thể là tùy ý.   Một loại ràng buộc tổng quát hóa thứ hai liên quan tới việc liệu các thực thể có thuộc vào nhiều hơn một tập thực thể ở mức thấp hơn không trong một quá trình tổng quát hóa. Các tập thực thể ở mức thấp hơn có thể là một trong các loại sau:   * **Không giao nhau**: Một ràng buộc không giao nhau đòi hỏi rằng một thực thể không thể thuộc vào nhiều hơn một tập thực thể ở mức thấp hơn. Trong ví dụ trên, một thực thể account có thể thỏa mãn một điều kiện cho thuộc tính account-type tại một thời điểm. Ví dụ, một account-type có thể hoặc là checking account (tài khoản thường dùng) hoặc một saving account (tài khoản tiết kiệm) nhưng không thể  là cả hai được. * **Giao nhau**: Trong tổng quát hóa có giao nhau, một thực thể có thể thuộc vào nhiều hơn một tập thực thể ở mức thấp hơn trong cùng một quá trình tổng quát hoa. Lấy ví dụ, xem xét nhóm làm việc trong ngân hàng từ phần trước. Giả sử rằng các quản lý nào đó có thể tham gia vào nhiều hơn một nhóm làm việc. Một nhân viên nào đó (hay một quản lý nào đó) vì thể có thể xuất hiện trong nhiều hơn một tập thực thể nhóm mà là tập thực thể ở mức thấp hơn của employee. Chú ý là việc giao nhau của thực thể ở mức thấp là trường hợp thông thường hay nhắc đến nếu như không chú thích gì thêm, một ràng buộc không giao nhau phải được thể hiện một cách tường minh trong quá trình tổng quát hóa (hay cụ thể hóa). Trong mô hình E-R một ràng buộc không giao nhau được thể hiện bằng từ “disjoint” đặt cạnh biểu tượng hình tam giác như trong hình vẽ minh họa dưới đây. Ý nghĩa của lược đồ bây giờ đã rõ ràng: các nhân viên và các khách hàng là các phân nhóm cụ thể hóa của tập người nói chung và ràng buộc không giao nhau ngụ ý rằng một nhân viên cũng không phải là một khách hàng. Nếu ràng buộc không giao nhau này được bỏ đi thì một nhân viên có thể là một khách hàng (hoặc nhìn từ một hướng khác, một người trong công ty này có thể vừa là khách hàng vừa là một nhân viên).   **Ràng buộc dựa trên tính toàn bộ**: đây là loại ràng buộc cuối cùng của quá trình tổng quát hóa hoặc cụ thể hóa, xác định liệu một thực thể của tập thực thể ở mức cao phải thuộc vào ít nhất một tập trong các tập thực thể ở mức thấp hơn hay không. Loại ràng buộc này có thể được rơi vào một trong hai dạng dưới đây:   * **Tổng quát hóa/cụ thể hóa toàn bộ**: mỗi thực thể ở mức cao phải thuộc vào một thực thể ở mức thấp hơn. * **Tổng quát hóa/cụ thể hóa một phần**: Một vài thực thể ở mức cao có thể không thuộc vào một tập thực thể nào mức thấp hơn nào. Tổng quát hóa một phần là trường hợp ngầm định nếu như không có phát biều gì về loại ràng buộc này trong lược đồ. (Nhắc lại là sự tham gia toàn bộ được thể hiện trong lược đồ E-R bằng một đường kết nối kép, vì vậy nó cũng sẽ được sử dụng để thể hiện sự tổng quát hóa toàn bộ. Trong ví dụ dưới đây tổng quát hóa là toàn bộ và có giao nhau có nghĩa là mỗi người trong công ty phải xuất hiện như là một nhân viên hoặc là một khách hàng và cũng có thể là cả hai.   Ví dụ với các ràng buộc này thể hiện trong hình vẽ dưới đây:   |  | | --- | |  | |  |   **Tích hợp**: Một trong những hạn chế của mô hình E-R là nó không thể biểu diễn các mối quan hệ trong các mối quan hệ. Để hiểu được tại sao việc này lại quan trọng chúng ta xem xét mối quan hệ ba ngôi works-on giữa employee (nhân viên), branch (chi nhánh ngân hàng-văn phòng) và job (công việc) được thể hiện trong lược đồ E-R dưới đây:   |  | | --- | |  | |  |    Với ngữ cảnh này, giả sử rằng chúng ta muốn ghi nhận các quản lý cho mỗi nhiệm vụ được thực hiện bởi một nhân viên tại một văn phòng chi nhánh nào đó; có nghĩa là chúng ta muốn lưu trữ các người quản lý cho quan hệ ba ngôi (*employee, branch, job)*. Giả thiết rằng chỉ có một tập thực thể *manager.* Một cách để thực hiện việc này là tạo ra một mối quan hệ bốn ngoi như hình vẽ sau đây.  Câu hỏi đặt ra là tại sao không thể hiện mối quan hệ hai ngôi giữa *manager* và *employee?* Câu trả lời như sau: một mối quan hệ sẽ không cho phép chúng ta thể hiện  (*branch, job)* của một *employee (*nhân viên) được quản lý bởi người quản lý nào.   |  | | --- | |  | |  |   Khi bạn nhìn vào lược đồ E-R mô hình hóa tình huống này, dường như tập các mối quan hệ *works-on* và *manages* có thể kết hợp thành một tập mối quan hệ duy nhất. Tuy nhiên, chúng ta không thể thực hiện điều đó bởi vì một bộ ba *employee, branch, job* có thể không có một người quản lý. Tuy nhiên, rõ ràng là có một thông tin dư thừa trong hình vẽ trên vì mỗi bộ ba *employee, branch, job* của mối quan hệ *manages*  cũng giống của *works-on*. Nếu người quản lý là một giá trị chứ không phải là một thực thể, ta có thể biến *manager* thành một thuộc tính đa trị của mối quan hệ *works-on*. Tuy nhiên, việc biến đổi đó sẽ làm việc tìm kiếm khó khăn hơn (cả về logic lẫn về chi phí thực hiện), lấy ví dụ, bộ ba employee-branch-job sẽ do người quản lý nào chịu trách nhiệm. Tuy nhiên, sự lựa chọn này không sẵn có trong trường hợp người quản lý là một thực thể *manager.* Cách tốt nhất để mô hình loại tình huống này là sử dụng **tích hợp (hay kết hợp).**  Kết hợp là một sự trừu tượng thông qua việc các mối quan hệ được coi như các thực thể ở mức cao. Vì vậy, trong ví dụ trên, chúng ta có thể voi tập quan hệ *works-on* (liên quan tới các tập thực thể *employee, branch,* và *job*) như một tập thực thể ở mức cao được gọi tên là *works-on*. Một tập thực thể như vậy được đối xử theo cách thức giống như các tập thực thể khác. Ta có thể tạo một quan hệ hai ngôi *manages* giữa *works-on* và *manager* để thể hiện ai quản lý các nhiệm vụ này. Lược đồ E-R dưới đây sẽ thể hiện một sự tích hợp trong mô hình E-R.   |  | | --- | |  | |  | |
| 2.2.10.5. Các mối quan hệ nhiều ngôi |
| Hầu hết các mối quan hệ chúng ta đã xem xét từ trước đến giờ đều là các quan hệ hai ngôi, là những quan hệ liên quan tới hai tập thực thể. Mọi mối quan hệ liên quan tới nhiều hơn hai tập thực thể có thể được chuyển đổi tới một tập các mối quan hệ hai ngôi, nhiều-1. Điều này là rất có ích trong khi mô hình E-R không hạn chế các quan hệ tới quan hệ hai ngôi nhưng các mô hình dữ liệu khác ví dụ như ngôn ngữ định nghĩa đối tượng có hạn chế này. Để mô tả cho sự chuyển đổi từ mối quan hệ nhiều ngôi sang một tập các mối quan hệ hai ngôi, xem xét một lược đồ E-R ví dụ sau đây   |  | | --- | |  | |  |   Được chuyển thành một tập các quan hệ hai ngôi như hình vẽ dưới đây   |  | | --- | |  | |  | |
| 2.2.10.6. Lược đồ E-R với các chỉ thị vai trò |
| Các vai trò trong một lược đồ E-R được biểu diễn bằng bởi việc gán nhãn lên các đường kết nối giữa các tập thực thể với các tập các mối quan hệ. các vai trò này có thể được xác định cho các mối quan hệ đệ quy, nhị phân, và không nhị phân.   |  | | --- | |  | |  | |  | |  | |
| 2.3. Mô hình dữ liệu quan hệ |
| 2.3.1. Giới thiệu mô hình dữ liệu quan hệ |
| Mô hình dữ liệu quan hệ được phát triển dựa trên khái niệm về quan hệ toán học. Nhà khoa học đề xuất ra mô hình quan hệ này, có tên là Codd, là một nhà toán học có sư phạm và ông đã sử dụng một thuật ngữ từ lĩnh vực này, liên quan chủ yếu đến lý thuyết tập hợp và logic mệnh đề để làm tên của mô hình này. Mô hình dữ liệu quan hệ bao gồm những khái niệm cơ bản sau:   * Quan hệ: một quan hệ là một bảng (ma trận) với các hàng và các cột. Các quan hệ sẽ lưu giữ thông tin về các đối tượng được mô hình hóa trong một cơ sở dữ liệu. * Thuộc tính: một thuộc tính là một cột được đặt tên của một quan hệ. Một thuộc tính là một đặc tính nào đó của một thực thể (hay một mối quan hệ) được mô hình hóa trong cơ sở dữ liệu. Các thuộc tính có thể xuất hiện theo bất kỳ trật tự nào trong một quan hệ. * Miền giá trị: Một miền giá trị là một tập các giá trị có thể của một hoặc nhiều thuộc tính. Mỗi thuộc tính được xác định trên một miền giá trị nào đó. Các miền giá trị có thể khác nhau cho mỗi thuộc tính hoặc hai hay nhiều thuộc tính có thể được xác định trên cungd một miền giá trị. * Bộ: Một bộ là một hàng của một quan hệ. Các bộ có thể xuất hiện theo bất ký trật tự nào trong một quan hệ và quan hệ sẽ vẫn giống nhau vì vậy thể hiện cùng ý nghĩa. * Cấp:  của  một quan hệ là số lượng các thuộc tính mà nó có * Lực lượng: Lực lượng của một quan hệ là số lượng các bộ mà nó có. * Cơ sở dữ liệu quan hệ: Một tập hợp các quan hệ được chuẩn hóa với các tên phân biệt nhau.   Một ví dụ về quan hệ thể hiện ở hình vẽ dưới đây:   |  | | --- | |  | |  |   Các định nghĩa miền giá trị cho ví dụ trên   |  |  |  |  | | --- | --- | --- | --- | | **Thuộc tính** | **Tên miền** | **Ý nghĩa** | **Định nghĩa miền** | | staffNo | Staffnumbers | Tập của tất cả các số hiệu có thể của nhân viên | Ký tự: kích cỡ 4, phải bắt đầu bằng chữ cái. | | fName, lName | Name | Tập tất cả các tên có thể của một người | Ký tự: kích cỡ 20 | | DOB | Date | Ngày sinh của một người | Ngày: trong khoảng  từ 01-Jan-20,  Định dạng: dd-mmm-yy | | salary | Salaries | Các giá trị có thể của lượng nhân viên | Dạng tiền tệ: 7 chữ số, trong khoảng 10,000-9,999,999 | | position | Alljobs | Tập tất cả các vị trí có thể có của nhân viên trong một công ty | Lựa chọn một trong tập: {ceo, cfo, coo,manager, asst. manager, driver, secretary} |   Các thuật ngữ tương đương có thể thay thế được lẫn nhau trong mô hình quan hệ   |  |  |  | | --- | --- | --- | | **Thuật ngữ chính thống** | **Lựa chọn 1** | **Lựa chọn 2** | | Quan hệ | Bảng | Tệp | | Bộ | Hàng | Bản ghi | | Thuộc tính | Cột | Trường | |
| 2.3.1.1. Định nghĩa quan hệ |
| Để hiểu được ý nghĩa thực sự của khái niệm quan hệ, chúng ta cần nhắc lại một số khái niệm toán học cơ bản.   * Cho hai tập D1 và D2 với  D1 = {2, 4} và D2 = {1, 3, 5}, tích Đề các của hai tập này được kí hiệu là D1 x D2, là tập của tất cả các cặp có thứ tự với phần tử đầu tiên là một phần tử của D1 và phần tử thứ hai là của D2   D1 x D2  = {(2, 1), (2, 3), (2, 5), (4, 1), (4, 3), (4, 5)}  Mọi tập con của tích Đề các này là một quan hệ vì vậy chúng ta có thể sinh ra một quan hệ R sao cho R = {(2, 3), (4, 3)} Ngoài ra chúng ta có thể xác định một điều kiện nào đó  để lựa chọn các phần tử từ D1 x D2 cho quan hệ R ví dụ  R = {(x, y) | x \epsilon D1, y \epsilonD2, và y = 3}   * Cho ba tập D1,  D2, và D3 với    D1 = {2, 4}, D2 = {1, 3}, và D3 = {3, 6}, tích Đề các của ba tập này được kí hiệu là D1 ´ D2´ D3, là tập của tất cả các cặp có thứ tự với phần tử đầu tiên là một phần tử của D1 và phần tử thứ hai là của D2 , phần tử thứ ba là của D3.   D1 x D2 x D3  = {(2, 1, 3), (2, 1, 6), (2, 3, 3), (2, 3, 6), (4, 1, 3), (4, 1, 6), (4, 3, 3), (4, 3, 6)}  Mọi tập con của tích Đề các này là một quan hệ.   * Một cách tổng quát, nếu D1,D2,..,Dn là *n* tập thì tích Đề các của chúng được định nghĩa là   D1 x D2 x,.., x Dn = {(d1,d2,...,dn)| d1 \epsilonD1, d2 \epsilonD2,..., dn\epsilonDn} và nhìn chung được viết với ký pháp:   |  | | --- | |  | |  |  * Một lược đồ quan hệ là một quan hệ có tên được xác định bởi một tập các thuộc tính và các cặp tên miền   Ri = {A1:d1, A2:d2, ..., An:dn | d1 \epsilonD1, d2 \epsilonD2, ..., dn \epsilonDn}   * Một lược đồ cơ sở dữ liệu quan hệ là một tập các lược đồ quan hệ, mỗi lược đồ có một tên riêng R = {R1, R2, ..., Rn}   Một quan hệ có các đặc tính sau:   1. Quan hệ có một định danh hay tên phân biệt với tên của các quan hệ khác trong lược đồ quan hệ. 2. Mỗi thuộc tính chứa một giá trị nguyên tố (không phân chia được nữa). 3. Mỗi thuộc tính có một tên riêng biệt. 4. Các giá trị của một thuộc tính đều xuất phát từ một miền giá trị. 5. Mỗi bộ là phân biệt nhau; không có hai bộ nào giống hệt nhau. 6. Trật tự của các thuộc tính không quan trọng. 7. Trật tự của các bộ cũng không quan trọng trên lý thuyết. (Tuy nhiên, trong thực tế, trật tự này có thể ảnh hưởng đến hiệu quả truy nhập vào các bộ, Chi tiết sẽ xem xét trong những phần sau). |
| 2.3.1.2. Lược đồ quan hệ và các thể hiện |
| Khái niệm về lược đồ quan hệ và thể hiện (instance) của quan hệ là khác nhau và cần được phân biệt. Lược đồ bao gồm tên và các thuộc tính cho quan hệ và khá bất biến. Còn một thể hiện của quan hệ là một tập các bộ của quan hệ đó và thể hiện này có thể thay đổi thường xuyên. Hầu hết  các cập nhật và mỗi quá trình chèn thêm hay xóa các bộ sẽ làm thay đổi thể hiện của quan hệ đó. Một cơ sở dữ liệu hiện thời (snapshot database) thể hiện trạng thái hiện tại của thế giới thực được lưu trữ trong cơ sở dữ liệu đó. Tại một thời điểm bất kỳ, nó mô hình hóa thể hiện hiện tại của thế giới thực. Nếu thế giới thực thay đổi, cơ sở dữ liệu cũng sẽ thay đổi để duy trì cách biểu diễn cái thể hiện thê giới thực hiện tại đó. Ví dụ về các quan hệ giống nhay và khác nhau thể hiện trong hình vẽ dưới đây:   |  | | --- | |  | |  | |
| 2.3.1.3. Thiết kế logic |
| Trong quá trình thiết kế logic bạn chuyển đổi thiết kế mức khái niệm sang các lược đồ cơ sở dữ liệu quan hệ. Như vậy đầu vào cho quá trình này chính là lược đồ thực thể liên kết (E-R diagrams) và đầu ra của quá trình này là các lược đồ quan hệ. Việc ánh xạ các lược đồ thực thể liên kết sang các quan hệ là một quá trình khá dễ dàng với một tập các luật được định nghĩa khá rõ. Trong thực tế nhiều công cụ CASE (công cụ trợ giúp cho công nghệ phần mềm) có thể thực hiện tự động nhiều bước của quá trình chuyển đổi. Tuy nhiên, việc bạn hiểu rõ các bước này thực sự quan trọng bởi ba lý do sau đây:   1. Các công cu CASE thường không thể mô hình hóa các quan hệ dữ liệu phức tạp ví dụ như các quan hệ ba ngôi và các quan hệ giữa các lớp cha/lớp con. Các bước này sẽ được thực hiện thủ công bằng tay. 2. Có rất nhiều lựa chọn hợp lý nên bạn phải lựa chọn một cách thủ công bằng tay 3. Bạn cần được chuẩn bị để thực hiện việc kiểm tra chất lượng đối với những kết quả thu được sau khi chạy các công cụ CASE.   Trong các bước bạn sẽ cần phải tuân thủ theo để ánh xạ các lược đồ thực thể liên kết sang các lược đồ quan hệ, rất có ích nếu bạn nhớ rằng chúng ta đã định nghĩa ba loại thực thể sau:   * Thực thể thông thường là các thực thể có thể tồn tại độc lập và thường thể hiện các đối tượng của thế giới thực ví dụ như con người hoặc sản phẩm. Nó được thể hiện trong mô hình thực thể liên keeys bằng các hình chữ nhât với một đường viền đơn. * Thực thể yếu là các thực thể không thể tồn tại một mình mà phải đi cùng với một mối quan hệ xác định bởi một loại thực thể xác định nó (thực thể chủ- khỏe). Các thực thể yếu được biểu diễn bởi một hình chữ nhất với đường viền kép. * Thực thể kết hợp (hay còn được gọi là các danh động từ) được hình thành từ những mối quan hệ nhiều-nhiều giữa các loại thực thể khác nhau. Các thực thể kết hợp này được biểu diễn bởi một hình chữ nhật với đường viền đơn và được bao quanh bởi một biểu tượng quan hệ hình thoi. |
| 2.3.2. Ánh xạ mô hình thực thể liên kết sang mô hình quan hệ |
| Bước 1: dùng cho việc ánh xạ các thực thể thông thường (thực thể khỏe)   * Mỗi thực thể thông thường trong mô hình thực thể liên kết sẽ được chuyển đổi thành một lược đồ quan hệ. * Tên của quan hệ thường là tên của loại thực thể. * Mỗi thuộc tính đơn của loại thực thể trở thành một thuộc tính của lược đồ quan hệ * Thuộc tính định danh trở thành khóa chính của quan hệ tương ứng   ***Ví dụ cho bước 1***   |  | | --- | |  | |  |   Các thuộc tính ghép: khi một loại thực thể thường có một thuộc tính ghép, chỉ những thuộc tính đơn của thuộc tính ghép này được đưa vào lược đồ quan hệ mới. Ví dụ thể hiện trong hình dưới đây   |  | | --- | |  | |  |   Với trường hợp có thuộc tính đa trị: khi  một thực thể thường có một thuộc tính đa trị, hai lược đồ quan hệ mới được tạo ra (không phải một cái). Lược đồ quan hệ đầu tiên chứa tất cả các thuộc tính của thực thể loại trừ thuộc tính đa trị. Lược đồ thứ hai sẽ bao gồm hai thuộc tính cấu thành khóa chính của lược đồ quan hệ thứ hai. Thuộc tính thứ nhất trong khóa chính này là khoá chính của lược đồ thứ nhất, cái trở thành khóa ngoại trong lược đồ thứ hai. Thuộc tính thứ hai là thuộc tính đa trị này. Tên của lược đồ thứ hai nên thể hiện ngữ nghĩa của thuộc tính đa trị. Ví dụ cho trường hợp thuộc tính đa trị được thể hiện trong hình vẽ dưới đây   |  | | --- | |  | |  |   Lưu ý trong các lược đồ quan hệ trước đó được tạo ra bởi thuộc tính đa trị *skill,* lược đồ quan hệ kết quả *employee-skill*  chỉ có các thuộc tính khóa. Mỗi bộ đơn giản lưu lại thực tế là một nhân viên nào đó sẽ có một kỹ năng tương ứng. Điều này cung cấp cho người thiết kế cơ sở dữ liệu cơ hội để gợi ý cho người dùng rằng các thuộc tính mới có thể được thêm vào quan hệ này. Ví dụ, các thuộc tính *years-experience* và/hoặc *certificate-date* có thể là các giá trị mới thích hợp với quan hệ này. |
| **Bước 2: Ánh xạ các thực thể yếu**   * Nhắc lại rằng một thực thể yếu không thể tồn tại độc lập, chỉ có thể tồn tại thông qua một quan hệ xác định với một loại thực thể khác được gọi là chủ thể. * Một thực thể yếu không có một định danh đầy đủ mà phải có một thuộc tính được gọi là định danh một pần cho phép phân biệt các thể hiện khác nhau của loại thực thể yếu này cho mỗi thể hiện của thực thể chủ. * Thủ tục sau đây giả sử rằng bạn đã tạo một lược đồ quan hệ liên quan tới loại thực thể xác định này. Nếu bạn chưa thực hiện- cần làm ngay bây giờ trước khi đi tiếp. * Đối với mỗi loại thực thể yếu, tạo một lược đồ quan hệ mới và đưa tất cả các thuộc tính đơn (hoặc các thành phần đơn của các thuộc tính ghép) vào thành thuộc tính của lược đồ quan hệ này. * Sau đó thêm khóa chính của quan hệ xác định vào thành một thuộc tính khóa ngoài trong lược đồ quan hệ mới. * Khóa chính của lược đồ quan hệ mới là sự kết hợp của khoá chính của quan hệ xác định và định danh một phần của loại thực thể yếu.   ***Ví dụ về ánh xạ các thực thể yếu được thể hiện ở hình vẽ dưới đây***   |  | | --- | |  | |  | |
| **Bước 3: Ánh xạ các quan hệ hai ngôi** Thủ tục cho việc ánh xạ các quan hệ thành mô hình quan hệ phụ thuộc vào số ngôi của quan hệ (một ngôi, hai ngôi, ba ngôi, v.v…) và loại của các mối quan hệ (quan hệ 1-1, 1-nhiều hay nhiều-nhiều). Chúng ta sẽ xem xét việc ánh xạ cho các loại quan hệ đó trong phần này. Lưu ý là quan hệ lạo 1-nhiều và nhiều-1 là đối xứng nhau. ***Ánh xạ các quan hệ 1-nhiều hai ngôi***   * Với mỗi quan hệ 1-nhiều hai ngôi, đầu tiên tạo một lược đồ quan hệ cho mỗi loại thực thể tham gia vào mối quan hệ sử dụng các thủ tục ở bước 1 * Sau đó, thêm thuộc tính khóa chính (hoặc các thuộc tính) của thực thể bên phía 1 của mối quan hệ thành một khóa ngoại cho quan hệ  nằm ở bên phía nhiều của mối quan hệ (khóa chính lấy từ bên phía nhiều của mối quan hệ)   ***Ví dụ về cho quan hệ hai ngôi 1-nhiều***   |  | | --- | |  | |  |   ***Ánh xạ cho các quan hệ nhiều-nhiều hai ngôi***  Cho mỗi quan hệ hai ngôi nhiều-nhiều giữa hai thực thể loại A và B, đầu tiên phải tạo thêm một lược đồ quan hệ mới C. Khóa chính của một lược đồ C là sự kết hợp của các khóa chính của các tập thực thể tham gia quan hệ và các khóa chính này cũng là khóa ngoại của C. Các thuộc tính không phải là khóa mà liên quan tới quan hệ nhiều-nhiều giữa A và B cũng được đưa vào lược đồ quan hệ C.  ***Ví dụ việc chuyển đổi sang lược đồ quan hệ cho loại quan hệ nhiều-nhiều***   |  | | --- | |  | |  |  |  | | --- | |  | |  |   ***Ánh xạ sang lược đồ quan hệ cho loại quan hệ 1-1***: quá trình này cần qua hai bước   1. Hai quan hệ được tạo ra, mỗi quan hệ liên quan tới một loại thực thể tham gia  mối quan hệ đó. 2. Khóa chính của một quan hệ sẽ thành khóa ngoài trong quan hệ còn lại.   Trong một quan hệ 1-1, sự tham gia vào liên kết  trong một hướng thường là một lựa chọn, trong khi với hướng kia là bắt buộc (nhắc lại loại ràng buộc tham gia liên kết). Bạn nên thêm vào quan hệ của bên có tham gia không bắt buộc khóa ngoài của tập thực thể còn lại mà có tham gia bắt buộc vào mối liên kết  1-1. Cách này sẽ tránh việc lưu trữ các giá trị rỗng cho thuộc tính khóa ngoài. Mọi thuộc tính liên quan tới bản thân quan hệ cũng được đưa vào cùng quan hệ đó như là khóa ngoài. ***Ví dụ về việc chuyển đổi quan hệ hai ngôi 1-1 thể hiện ở hình vẽ dưới đây***   |  | | --- | |  | |  | |
| **Bước 4: Ánh xạ các thực thể liên kết (hay thực thể kết hợp)** Ánh xạ một thực thể liên kết sang một lược đồ quan hệ tương tự thủ tục chuyển đổi một quan hệ nhiều-nhiều. Hai bước sau cần được thực hiện:   1. Ba quan hệ được tạo ra, trong đó hai quan hệ liên quan tới mỗi loại thực thể tham gia  mối quan hệ đó và quan hệ thứ ba cho thực thể liên kết. Quan hệ được hình thành từ thực thể liên kết được gọi là quan hệ liên kết. 2. Hành động trong bước này phụ thuộc vào việc có gán một định danh cho thực thể liên kết trong lược đồ E-R hay không. Hai trường hợp sau xảy ra:   a.       Một định danh không được gán b.      Một định danh được gán  Chúng ta sẽ xem xét từng trường hợp riêng biệt.  ***Trường hợp không gán định danh***: khóa chính ngầm định cho quan hệ liên kết này bao gồm các thuộc tính khóa chính từ hai quan hệ còn lại. Những thuộc tính này sẽ là khóa ngoài tham chiếu tới hai quan hệ đó. Một ví dụ về trường hợp này được thể hiện dưới đây nhưng lưu ý sự giống nhau của ví dụ này với ví dụ của trường hợp chuyển đổi cho quan hệ nhiều-nhiều.   |  | | --- | |  | |  | |  | |  |   Đôi khi người xây dựng mô hình dữ liệu sẽ gán một định danh tới loại thực thể liên kết trong lược đồ E-R. Có hai lý do chính có thể xảy ra:   1. Thực thể liên kết có một định danh tự nhiên mà gần gũi với người sử dụng cuối. 2. Định danh ngầm định (bao gồm các định danh cho mỗi loại thực thể tham gia liên kết) có thể không xác định các thể hiện của thực thể liên kết một cách duy nhất.   Dù trong trường hợp nào trong hai trường hợp trên, quá trình để ánh xạ thực thể liên kết sẽ được thay đổi như sau.  ***Trường hợp một định danh được gán***: như trên, một quan hệ liên kết mới được tạo ra để thể hiện thực thể liên kết. Khóa chính cho quan hệ này là một định danh được gán trên lược đồ E-R (hơn là giá khóa ngầm định như trong trường hợp trước). Khóa chính cho hai loại thực thể tham gia sau đó sẽ được thêm vào làm khóa ngoài trong quan hệ liên kết này. Một ví dụ về trường hợp này xuất hiện như trong hình vẽ dưới đây.   |  | | --- | |  | |  |  |  | | --- | |  | |  | |
| **Bước 5: Ánh xạ các quan hệ một ngôi (đệ quy)** Nhắc lại rằng một quan hệ đệ quy được định nghĩa như một quan hệ giữa các thể hiện của cùng một loại thực thể. Hai trường hợp quan trọng nhất của các mối quan hệ một ngôi này là loại quan hệ 1-nhiều và nhiều-nhiều. Chúng ta sẽ bàn luận đến hai loại này tách biệt nhay vì chúng được xử lý khác nhau.  ***Ánh xạ quan hệ đệ quy loại 1-nhiều***  Tập thực thể trong quan hệ một ngôi được ánh xạ thành một lược đồ quan hệ sử dụng một thủ tục được mô tả trong bước 1. Tiếp đến một thuộc tính khóa ngoại được thêm vào cùng quan hệ có tham chiếu tới các giá trị của khóa chính (khóa ngoại này phải có cùng miền giá trị với khóa chính). Một khóa ngoại đệ quy là một khóa ngoại của một quan hệ mà tham chiếu tới giá trị khóa chính của cùng quan hệ đó. Một ví dụ về trường hợp ánh xạ loại này được thể hiện trong hình vẽ dưới đây:   |  | | --- | |  | |  | |  | |  |   ***Ánh xạ quan hệ đệ quy loại nhiều-nhiều***  Với loại quan hệ đệ quy loại này, hai lược đồ quan hệ sẽ được tạo ra: một thể hiện tập thực thể và lược đồ quan hệ liên kết để thể hiện mối quan hệ nhiều-nhiều. Khóa chính của quan hệ liên kết bao gồm hai thuộc tính. Những thuộc tính này (không nhất thiết phải có cùng tên) đều lấy giá trị của chúng từ các khóa chính của quan hệ còn lại. Các thuộc tính không khóa của quan hệ được thêm vào quan hệ liên kết. Ví dụ của trường hợp này về mối quan hệ phát sinh hóa đơn cho các nhiên vật liệu của các linh kiện được lắp ráp từ những linh kiện khác được thể hiện trong  hình vẽ dưới đây.   |  | | --- | |  | |  | |  | |  | |
| **Bước 6: Ánh xạ các quan hệ nhiều ngôi** Nhắc lại rằng một mối quan hệ ba ngôi được định nghĩa như một mối quan hệ giữa ba tập thực thể như được biểu diễn trong hình vẽ dưới đây   |  | | --- | |  | |  |   Một nhắc nhở về quan hệ nhiều ngôi: tất cả nên chuyển đổi về các thực thể liên kết trước khi được xử lý sâu hơn. Một ví dụ về việc chuyển đổi về thực thể liên kết cho lược đồ E-R trên thể hiện trong hình vẽ dưới đây.   |  | | --- | |  | |  |   Để ánh xạ một thực thể liên kết kết nối ba tập thực thể loại thường, ta cần tạo ra một quan hệ liên kết mới. Khóa chính ngầm định cho quan hệ này bao gồm các thuộc tính khóa chính của các loại thực thể tham gia liên kết (trong một số trường hợp cần thêm các thuộc tính khác để hình thành một khóa chính duy nhất). Những thuộc tính này hoạt động với vai trò các khóa ngoại tham chiếu tới từng khóa chính  của các tập thực thể tham gia liên kết. Mỗi thuộc tính của loại thực thể liên kết này trở thành thuộc tính trong quan hệ liên kết mới. Ví dụ của quá trình ánh xạ các quan hệ nhiều ngôi được thể hiện trong hình vẽ dưới đây.   |  | | --- | |  | |  | |  | |  | |
| **Bước 7: Ánh xạ các mối liên kết lớp cha/lớp con** Mô hình dữ liệu quan hệ không hỗ trợ trực tiếp các mối liên kết loại này. May thay, có nhiều chiến lược khác nhau mà người thiết kế cơ sở dữ liệu có thể dùng để thể hiện loại mối liên kết này trong mô hình dữ liệu quan hệ. Chúng ta sẽ xem xét một trong những kỹ thuật chung nhât được sử dụng để mô hình hóa loại liên kết cha/con. Kỹ thuật đó được thể hiện qua các bước dưới đây   1. Tạo ra một lược đồ quan hệ riêng biệt cho tập thực thể lớp cha và cho mỗi tập thực thể con. 2. Gán cho lược đồ quan hệ tương ứng với lớp cha các thuộc tính chung của tất cả các thành viên lớp con bao gồm cả khóa chính. 3. Gán cho lược đồ quan hệ của mỗi lớp con khóa chính của lớp cha và chỉ những thuộc tính duy nhất cho từng loại tập thực thể con. 4. Gán một (hoặc nhiều) thuộc tính của tập thực thể lớp cha thực hiện chức năng như một chỉ thị phân biệt các loại lớp con.   Xác định các thuộc tính phân biệt các lớp con: Với một mối quan hệ cha/con nào đó, nếu muốn thêm một thể hiện của lớp cha vào cơ sở dữ liệu thì thể hiện này sẽ nên được thêm vào lớp con nào? Một cách tiếp cận chung sử dụng một đối tượng phân biệt. Một đối tượng phân biệt các tập thực thể con là một thuộc tính của tập thực thể cha mà giá trị của nó xác định được các loại tập thực thể con (được sử dụng khi việc xác định các lớp con dựa trên các mệnh đề). Có hai trường hợp xảy ra: loại tập thực thể con không giao nhau và loại các tập thực thể con giao nhau. ***Ví dụ về đối tượng phân biệt trong lược đồ E-R cho loại tập thực thể không giao nhau***   |  | | --- | |  | |  |   ***Ví dụ về đối tượng phân biệt trong lược đồ E-R cho loại tập thực thể giao nhau***   |  | | --- | |  | |  | |  | |  | |
| 2.5. Giới thiệu về ngôn ngữ tạo mô hình thống nhất UML |
| Một số các thành phần của UML là   1. Lược đồ lớp (class): một lược đồ lớp tương tự như một lược đồ E-R 2. Lược đồ use case: được sử dụng để thể hiện sự tương tác giữa người sử dụng và hệ thống, cụ thể là các bước của công việc mà người dùng cần thực hiện (ví dụ như việc rút tiền từ một tài khoản ngân hàng hoặc đăng ký một khóa học) 3. Lược đồ hoạt động (activity): thể hiện luồng công việc giữa các thành phần của một hệ thống 4. Lược đồ cài đặt (implementation) thể hiện các thành phần của hệ thống và sự kết nối giữa chúng ở cả mức độ phần mềm và phần cứng   Sự liên quan giữa lược đồ lớp của UML và lược đồ E-R được thể hiện trong các hình vẽ dưới đây   |  | | --- | |  | |  | |  | |  |  |  | | --- | |  | |  | |  | |  | |
| 2.6. Ràng buộc toàn vẹn tham chiếu |
| Các ràng buộc tham chiếu có thể được hiểu đơn giản là việc đảm bảo rằng một thuộc tính nào đó có một giá trị khác rỗng. Tuy nhiên, các ràng buộc toàn vẹn tham chiếu thường liên quan tới các mối quan hệ giữa các tập thực thể. Chúng ta cùng xem xét ví dụ về ngân hàng và một quan hệ nhiều-một giữa khách hàng và tài khoản được biểu diễn như hình vẽ dưới đây.   |  | | --- | |  | |  |   Quan hệ *depositor* này thể hiện một thực tế là không có tài khoản nào có thể được chuyển tiền vào bởi nhiều hơn một khách hàng (và một khách hàng cũng chỉ có thể chuyển tiền vào nhiều tài khoản khác nhau). Quan trọng hơn, quan hệ này không phát biểu rằng một tài khoản phải được chuyển tiền vào bởi một khách hàng cũng như không phát biểu rằng một khách hàng phải thực hiện một việc chuyển tiền vào một tài khoản. Hơn nữa, nó cũng không thể hiện rằng nếu một tài khoản được chuyển tiền vào bởi một khách hàng thì một khách hàng sẽ được lưu trữ trong cơ sở dữ liệu. Một ràng buộc toàn vẹn tham chiếu yêu cầu rằng mỗi thực thể “được tham chiếu tới” bởi một mối quan hệ phải tồn tại trong cơ sở dữ liệu. Có một số phương pháp được sử dụng để đảm bảo tính ràng buộc toàn vẹn tham chiếu:   1. Việc xóa bỏ một thực thể được tham chiếu đến là không được phép. Nói một cách khác, nếu Kristi thực hiện một lần chuyển tiền vào tài khoản số hiệu 456 thì hệ quả là chúng ta sẽ không thể xóa thông tin liên quan tới Kristi hoặc của tài khoản 456. 2. Nếu một thực thể được tham chiếu được xóa bỏ thì tất cả các bản ghi tham chiếu tới thực thể bị xóa bỏ đó cũng bị xóa. Nói một cách khác, nếu chúng ta xóa thông tin về Kristi thì chúng ta phải xóa tất cả các thông tin về tài khoản của những tài khoản mà cô ta (một mình) đã chuyển tiền vào đó. Lưu ý la trong một ví dụ cụ thể mà chúng ta đang xem xét quan hệ là nhiều-một có nghĩa là nếu Kristi đã chuyển tiền vào một tài khoản thì cô ta sẽ là khách hàng duy nhaart làm việc đó. Trường hợp này không giống với quan hệ nhiều-nhiều.   Các ràng buộc toàn vẹn tham chiếu có thể được mô hình trong E-R. Thông thường chúng được thể hiện bằng một mũi tên đầu cong như trong hình vẽ dưới đây. Mũi tên cong thể hiện tồn tại một ràng buộc trên tập thực thể *account* thông quan mối quan hệ của nó *depositor* với tập thực thể *customers*   |  | | --- | |  | |  | |