

## Session - 2

# Entity-Relationship (E-R) Model and Normalization

Welcome to the Session, **Entity-Relationship (E-R) Model and Normalization**.

This session talks about Data Modeling, the E-R model, its components, symbols, diagrams, relationships, Data Normalization, and Relational Operators.

In this Session, you will learn to:

- Define and describe data modeling
- Identify and describe the components of the E-R model
- Identify the relationships that can be formed between entities
- Explain E-R diagrams and their use
- Describe an E-R diagram, the symbols used for drawing, and show the various relationships
- Describe the various Normal Forms
- Outline the uses of different Relational Operators



## 2.1 Giới thiệu

Một mô hình dữ liệu là một nhóm các công cụ khái niệm mô tả dữ liệu, mối quan hệ của nó, và các ý nghĩa. Nó cũng bao gồm các ràng buộc nhất quán, mà dữ liệu tuân thủ. Các mô hình quan hệ thực thể (Entity-Relationship), mô hình dữ liệu quan hệ (Relational), mô hình dữ liệu mạng (Network), và các mô hình phân cấp (Hierarchical) là những ví dụ của mô hình dữ liệu. Sự phát triển của tất cả các CSDL bắt đầu với các bước cơ bản của việc phân tích dữ liệu của nó để xác định mô hình dữ liệu tốt nhất sẽ đại diện cho nó. Khi bước này hoàn tất, mô hình dữ liệu sẽ được áp dụng cho các dữ liệu.

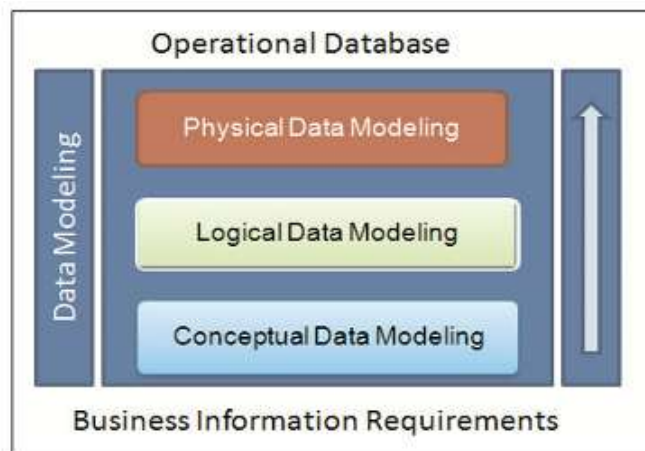
## 2.2 Data Modeling

Quá trình áp dụng một mô hình dữ liệu thích hợp với dữ liệu, để tổ chức và cấu trúc nó được gọi là mô hình hóa dữ liệu (data modeling)

Mô hình hóa dữ liệu là cần thiết để phát triển CSDL như là lập kế hoạch và thiết kế cho bất kỳ dự án phát triển. Xây dựng CSDL mà không có một mô hình dữ liệu, thì tương tự như phát triển một dự án mà không có kế hoạch và thiết kế của nó. Mô hình dữ liệu giúp các nhà phát triển cơ sở dữ liệu xác định các bảng quan hệ, các khóa chính và khóa ngoài, các thủ tục lưu trữ, và trigger cần thiết trong cơ sở dữ liệu.

Mô hình hóa dữ liệu có thể chia thành ba bước chính:

- **Conceptual Data Modeling ( Khái niệm mô hình hóa dữ liệu)**  
Người xây dựng mô hình dữ liệu xác định mức cao nhất của các mối quan hệ trong dữ liệu.
- **Logical Data Modeling**  
Việc xây dựng mô hình dữ liệu mô tả dữ liệu và các mối quan hệ của nó trong chi tiết. Người xây dựng mô hình dữ liệu tạo ra một mô hình logic của CSDL
- **Physical Data Modeling**  
Người xây dựng mô hình dữ liệu quy định cách xây dựng mô hình logic. Hình 2.1 cho thấy các bước khác nhau trong việc mô hình hóa dữ liệu



**Hình 2.1: Data Modeling Steps**

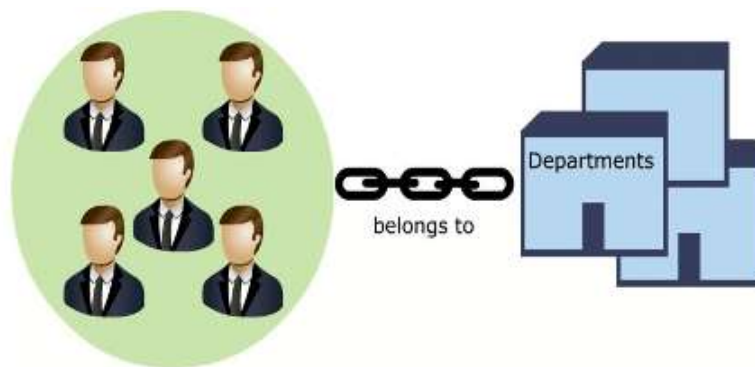
## 2.3 Mô hình quan hệ thực thể (E-R)

Các mô hình dữ liệu có thể được phân loại thành 3 nhóm khác nhau:

- Object-based logical models ( Mô hình logic dựa trên đối tượng)
- Record-based logical models ( Mô hình logic dựa trên bản ghi)
- Physical models ( Mô hình vật lý )

Mô hình quan hệ thực thể (Entity-Relationship) thuộc về phân loại đầu tiên. Mô hình này dựa trên một ý tưởng đơn giản. Dữ liệu có thể được coi là đối tượng thực trong thế giới, được gọi là thực thể và các mối quan hệ tồn tại giữa chúng. Ví dụ, dữ liệu về các nhân viên làm việc cho một tổ chức có thể được coi là một tập các nhân viên, và một tập các phòng ban khác nhau để tạo thành một tổ chức. Cả nhân viên, và phòng ban là các đối tượng trong thế giới thực. Một nhân viên thuộc về một bộ phận. Như vậy, mối quan hệ 'belong to' liên kết một nhân viên đến một phòng ban cụ thể.

Mối quan hệ nhân viên – Phòng ban có thể được mô hình như được thấy trong Hình 2.2.



Hình 2.2: E-R Model Mô tả một Tổ chức

Một mô hình quan hệ thực thể (E-R) bao gồm năm thành phần cơ bản sau:

- **Entity (Thực thể)**

Một thực thể là một đối tượng trong thế giới thực, có tồn tại về thể chất, và khác biệt rõ ràng với các đối tượng khác. Ví dụ, nhân viên, phòng ban, sinh viên, khách hàng, xe, và tài khoản là các thực thể.

- **Relationship (Quan hệ)**

Một mối quan hệ là một sự kết hợp, hoặc liên kết tồn tại giữa một hoặc nhiều thực thể. Ví dụ, thuộc về, sở hữu, làm việc cho, lưu trữ tại, được mua....vv

- **Attributes (Thuộc tính)**

Thuộc tính là các tính năng mà một thực thể có. Thuộc tính giúp phân biệt một thực thể này với một thực thể khác. Ví dụ, các thuộc tính của một Sinh viên sẽ là roll\_number , name, stream, semester.

Các thuộc tính của một xe hơi sẽ là registration\_number, model, manufacturer, color, price, owner.

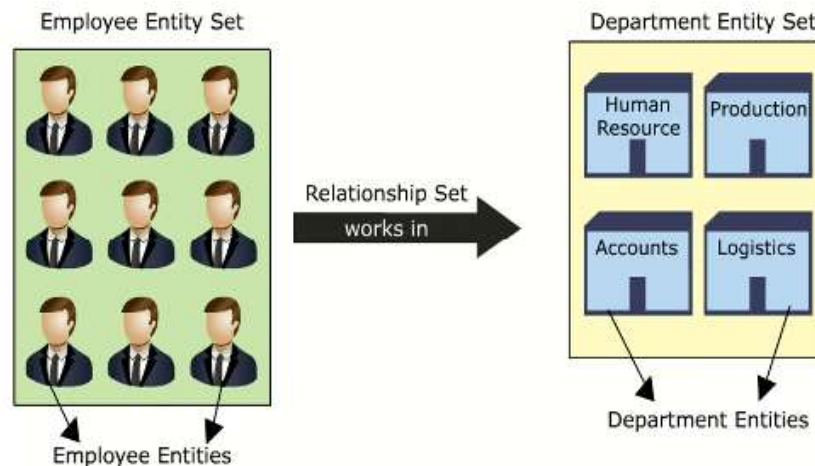
➤ **Entity Set ( Tập thực thể )**

Một tập thực thể là một tập các thực thể giống nhau. Ví dụ, các nhân viên của một tổ chức tạo thành một tập thực thể được gọi là tập thực thể nhân viên

➤ **Relationship Set ( Tập quan hệ )**

Một tập các mối quan hệ tương tự giữa hai, hoặc nhiều tập thực thể được gọi là một tập quan hệ. Ví dụ, các nhân viên làm việc trong một phòng ban cụ thể. Tập quan hệ 'work in' (làm việc tại) tồn tại giữa các nhân viên và các phòng ban được gọi là tập quan hệ 'work in'.

Các thành phần khác nhau của mô hình E-R có thể được xem trong Hình 2.3.



Hình 2.3: Components of the E-R Model

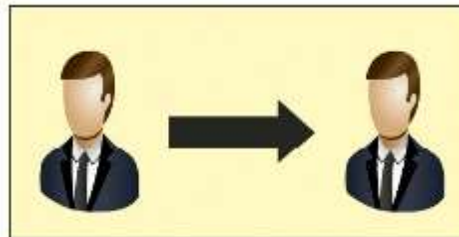
Các mối quan hệ liên kết một, hoặc nhiều thực thể, và có thể có ba loại sau:

➤ **Self-relationships (Quan hệ một ngôi )**

Các mối quan hệ giữa các thực thể của tập thực thể giống nhau được gọi là tự quan hệ. Ví dụ, một người quản lý và thành viên trong nhóm của mình, cả hai thuộc về tập thực thể nhân viên. Các thành viên trong nhóm làm việc cho người quản lý. Như vậy, mối quan hệ 'work for' (làm việc cho), tồn tại giữa hai thực thể nhân viên khác nhau của cùng một tập thực thể nhân viên

Mối quan hệ có thể được thấy trong Hình 2.4.

Tập thực thể Nhân viên

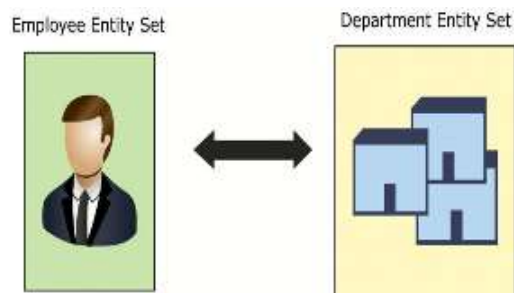


Hình 2.4: Self-Relationship

### ➤ Quan hệ hai ngôi

Các mối quan hệ tồn tại giữa các thực thể, của hai tập thực thể khác nhau được gọi là mối quan hệ nhị phân. Ví dụ, một nhân viên thuộc về một phòng ban. Mối quan hệ tồn tại giữa hai thực thể khác nhau, thuộc về hai tập thực thể khác nhau. Thực thể nhân viên thuộc về một tập thực thể nhân viên. Thực thể phòng ban thuộc về một tập thực thể Phòng ban.

Mối quan hệ có thể thấy trong Hình 2.5.



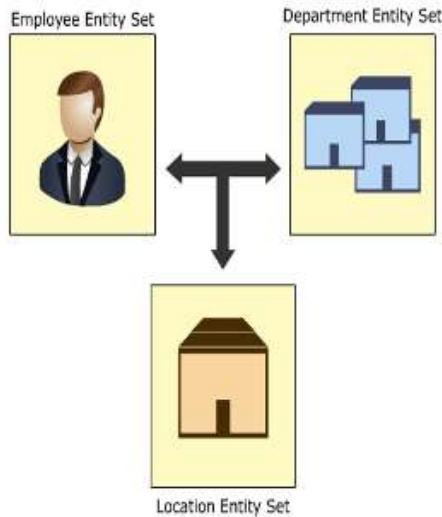
Hình 2.5: Binary Relationship

### ➤ Quan hệ ba ngôi

Các mối quan hệ tồn tại giữa 3 thực thể thuộc các tập thực thể khác nhau được gọi là mối quan hệ tam phân. Ví dụ, một nhân viên làm việc trong phòng kế toán, tại chi nhánh khu vực. Mối quan hệ 'works' (làm việc) tồn tại giữa tất cả ba tập thực thể, nhân viên, phòng ban, và địa điểm.



Mối quan hệ có thể được thấy trong Hình 2.6.



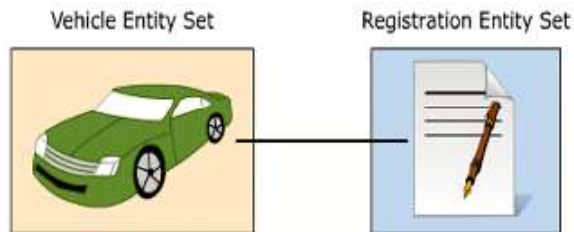
Hình 2.6: Ternary Relationship

Các mối quan hệ cũng có thể được chia theo tỷ lệ bản số ánh xạ. Các bản số ánh xạ khác nhau là như sau:

➤ **Một – Một (1 – 1)**

Loại ánh xạ này tồn tại khi một thực thể của một tập thực thể, có thể liên kết với duy nhất một thực thể của một tập thực thể khác.

Hãy xem xét mối quan hệ giữa một chiếc xe và giấy đăng ký nó. Mỗi một chiếc xe có một giấy đăng ký duy nhất. Không có hai chiếc xe nào, có thể có các chi tiết đăng ký giống nhau. Mối quan hệ này là một- trong-một (1 – 1), đó là, một chiếc xe chỉ có một đăng ký. Bản số ánh xạ có thể được nhìn thấy trong Hình 2.7.



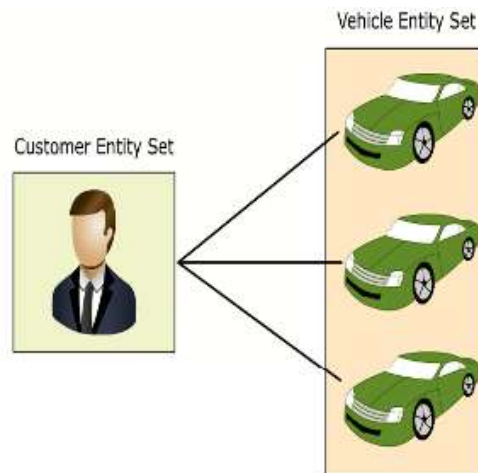
Hình 2.7: One-to-One Mapping Cardinality

➤ **Một – Nhiều ( 1 – n )**

Liên kết một – nhiều. Đây là loại ánh xạ tồn tại khi một thực thể của một tập thực thể có thể được liên kết với nhiều hơn một thực thể thuộc một tập thực thể khác

Hãy xem một mối quan hệ giữa một khách hàng, và các xe hơi của khách hàng. Một khách hàng có thể có nhiều ô tô. Do đó, ánh xạ là một – nhiều ánh xạ, đó là quan hệ một khách hàng – một, hoặc với nhiều xe hơi.

Bản số ánh xạ có thể được thấy trong Hình 2.8.



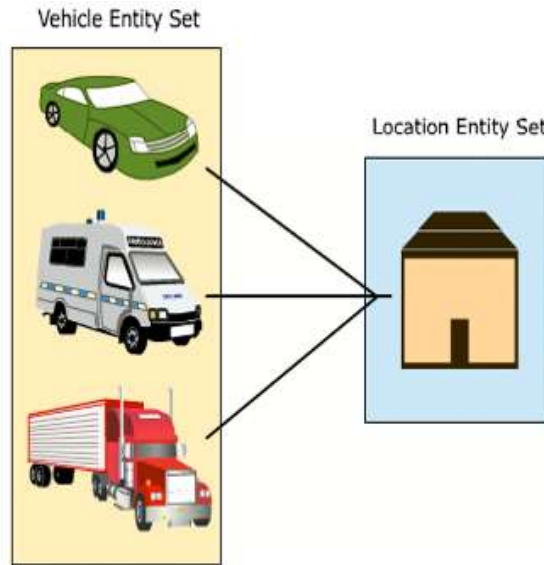
**Hình 2.8: One-to-Many Mapping Cardinality**

➤ **Nhiều – một ( n – 1 )**

Liên kết nhiều – một. Đây là loại ánh xạ tồn tại khi nhiều thực thể của một tập thực thể được liên kết với một thực thể của một tập thực thể khác. Sự liên kết này được thực hiện không phân biệt thực thể sau, đã được liên kết với các thực thể khác, hoặc liên kết với nhiều thực thể của tập thực thể cũ.

Hãy xem mối quan hệ giữa một chiếc xe, và nhà sản xuất của nó. Mỗi chiếc xe có duy nhất một nhà sản xuất, hoặc một liên doanh có liên quan đến nó trong mối quan hệ, 'được sản xuất bởi', nhưng cùng một nhà sản xuất, hoặc liên doanh có thể sản xuất được nhiều hơn một loại xe.

Sơ đồ có thể được thấy trong Hình 2.9.



**Hình 2.9: Many-to-One Mapping Cardinality**

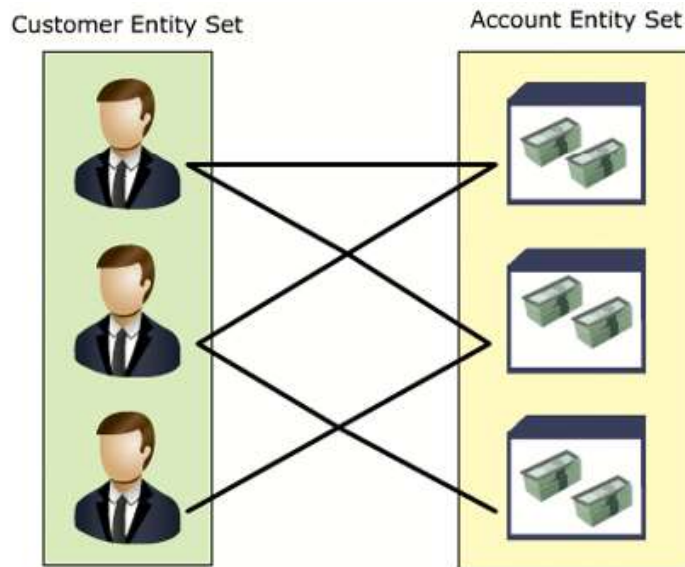
➤ **Nhiều – Nhiều ( n – n )**

Liên kết nhiều – nhiều. Đây là loại ánh xạ tồn tại khi bất kỳ số lượng nào của các thực thể thuộc một tập thực thể, có thể được liên kết với bất kỳ số lượng nào của các thực thể thuộc một tập thực thể khác.

Hãy xem mối quan hệ giữa một khách hàng của ngân hàng, và tài khoản của khách hàng. Một khách hàng có thể có nhiều tài khoản, và một tài khoản có thể có nhiều hơn một khách hàng được liên kết với nó, trong trường hợp nó là một tài khoản chung, hoặc tương tự. Do đó, ánh xạ của liên kết là nhiều – nhiều. Đó là một hoặc nhiều khách hàng được liên kết với một, hoặc nhiều tài khoản.



Bản số ảnh xạ được cho thấy trong Hình 2.10.



Hình 2.10: Many-to-Many Mapping Cardinality

Một số khái niệm bổ sung trong mô hình E-R là như sau:

➤ **Primary keys ( Khóa chính )**

Một primary key ( Khóa chính) là một thuộc tính duy nhất, có thể xác định một thực thể trong một tập thực thể. Hãy xem table 2.1 có chứa các chi tiết về các Sinh viên trong một trường học.

Enrollment_number	Name	Grade	Division
786	Ashley	Seven	B
957	Joseph	Five	A
1011	Kelly	One	A

Table 2.1: Student Details

Trong một trường học, mỗi sinh viên có một mã sinh viên duy nhất ( như là Enrollment\_number trong table 2.1). Bất kỳ sinh viên nào cũng có thể được xác định dựa trên mã số Sinh viên. Do đó, thuộc tính enrollment\_number đóng một vai trò là một khóa chính trong bảng Student Details

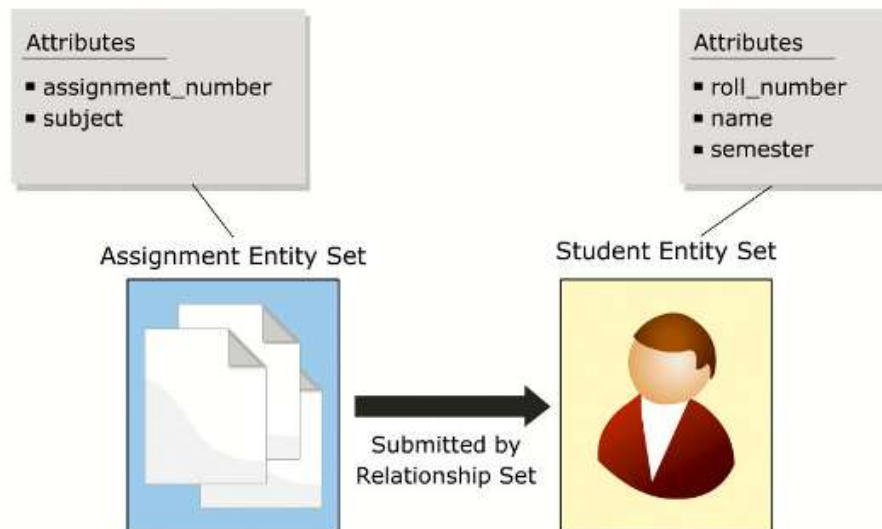
➤ **Weak entity sets ( Tập thực thể yếu)**

Các tập thực thể không có đủ các thuộc tính để thiết lập một primary key (khóa chính) bị gọi là các tập thực thể yếu.

➤ **Strong entity sets ( Tập thực thể mạnh )**

Các tập thực thể có đầy đủ các thuộc tính để thiết lập một primary key được gọi là các tập thực thể mạnh.

Hãy xem một kịch bản của một cơ sở giáo dục, nơi mà vào cuối mỗi học kỳ, các sinh viên được yêu cầu phải hoàn thành, và giao nộp một tập các bài tập ( assignment). Giáo viên sẽ lưu ý đến những bài tập được nộp bởi các sinh viên. Bây giờ, một bài tập và sinh viên có thể được coi là hai thực thể riêng biệt. Thực thể bài tập được mô tả bởi các thuộc tính **assignment\_number** và **subject**. Thực thể sinh viên được mô tả bởi **roll\_number**, **name**, và **semester**. Các thực thể bài tập có thể được nhóm lại để tạo thành một tập thực thể bài tập, và các thực thể sinh viên có thể được nhóm lại để tạo thành một tập thực thể sinh viên. Các tập thực thể được kết hợp bởi mối quan hệ 'submitted by' (nộp bởi). Mối quan hệ này được mô tả trong Hình 2.11.



**Hình 2.11: Assignment Student Relation**

Các thuộc tính, *assignment\_number*, và *subject* là không đủ để xác định một thực thể *assignment* duy nhất. Một mình thuộc tính *roll\_number* là đủ để xác định bất kỳ thực thể sinh viên nào. Do đó, *roll\_number* là primary key cho tập thực thể Sinh viên. Tập thực thể *assignment* (bài tập) là một tập thực thể yếu, vì nó thiếu một primary key (khóa chính). Tập thực thể *student* (sinh viên) là một tập thực thể mạnh, vì có sự hiện diện của thuộc tính *roll\_number*.

### 2.3.1 Biểu đồ quan hệ thực thể

Biểu đồ E-R là một cách trình bày bằng đồ họa của mô hình quan hệ thực thể. Biểu đồ E-R với sự trợ giúp của các biểu tượng khác nhau, đã trình bày được các thành phần khác nhau của mô hình E-R một cách hiệu quả.

Các biểu tượng được sử dụng cho các thành phần khác nhau có thể được thấy trong table 2.2

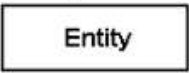
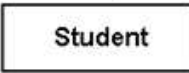







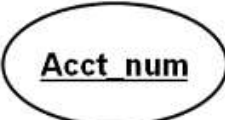
Component	Symbol	Example
Entity		
Weak Entity		
Attribute		
Relationship		
Key Attribute		

Table 2.2: E-R Diagram Symbols

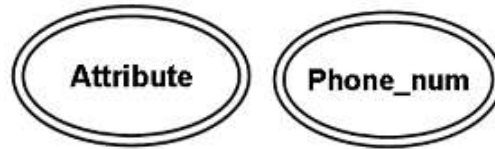
Các thuộc tính trong mô hình E-R có thể được phân thành nhiều loại như sau:

➤ **Multi-valued ( Đa giá trị )**

Thuộc tính đa trị là một thuộc tính chứa nhiều giá trị khác nhau thuộc một miền trị, được biểu diễn bằng hình bầu dục nét đôi.

Thuộc tính điện thoại của một cá nhân có thể có một hoặc nhiều giá trị, và một cá nhân có thể có một, hoặc nhiều số điện thoại. Do đó, thuộc tính điện thoại là một thuộc tính đa giá trị.

Biểu tượng và ví dụ của một thuộc tính đa giá trị có thể được thấy trong Hình 2.12.

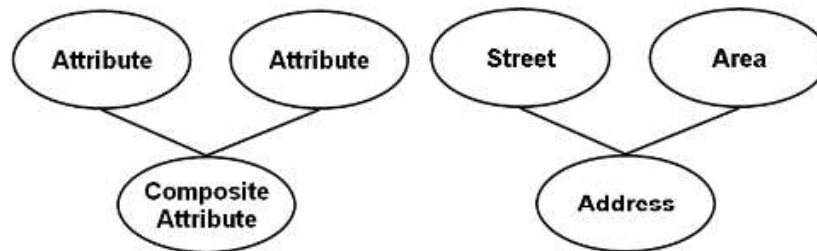


Hình 2.12: Symbol and Example of Multi-valued Attribute

➤ **Composite ( Phức hợp )**

Một thuộc tính phức hợp là một thuộc tính bị phân rã thành nhiều thuộc tính khác

Thuộc tính địa chỉ thường là một thuộc tính phức hợp, bao gồm các thuộc tính như đường phố, khu vực, và ..vv. Biểu tượng và ví dụ của một thuộc tính phức hợp có thể được thấy trong Hình 2.13

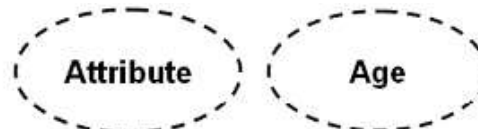


Hình 2.13: Symbol and Example of Composite Attribute

➤ **Derived ( Dẫn xuất )**

Thuộc tính dẫn xuất là thuộc tính mà giá trị của nó được suy dẫn từ các thuộc tính khác. Nó được biểu diễn bằng hình bầu dục nét đứt

Thuộc tính tuổi của một người là ví dụ tốt nhất cho các thuộc tính dẫn xuất. Đối với một thực thể người cụ thể, tuổi của một người có thể được xác định từ ngày hiện tại và ngày sinh của người đó.



Hình 2.14: Symbol and Example of Derived Attribute

Các bước để xây dựng một biểu đồ E-R là như sau:

1. Thu thập tất cả các dữ liệu cần được mô hình hóa
2. Xác định dữ liệu có thể được mô hình hóa như các thực thể trong thế giới thực
3. Xác định các thuộc tính cho từng thực thể

4. Sắp xếp các tập thực thể là yếu, hoặc mạnh.
5. Sắp xếp các thuộc tính của thực thể như các thuộc tính chính, thuộc tính đa giá trị, thuộc tính phức hợp, thuộc tính dẫn xuất, và ..vv
6. Xác định các mối quan hệ giữa các thực thể khác nhau

Sử dụng các biểu tượng để vẽ các thực thể, các thuộc tính và các mối quan hệ của chúng. Sử dụng các biểu tượng thích hợp trong khi vẽ các thuộc tính

Hãy xem một kịch bản của một ngân hàng, với các khách hàng và các tài khoản. Biểu đồ E-R cho kịch bản sẽ được xây dựng như sau:

#### Step 1: Thu thập thông tin

Ngân hàng là một tập các tài khoản được sử dụng bởi các khách hàng để tiết kiệm tiền.

#### Step 2: Xác định các thực thể

1. Khách hàng (Customer )
2. Tài khoản (Account )

#### Step 3: Xác định các thuộc tính

1. Customer: customer\_name, customer\_address, customer\_contact
2. Account: account\_number, account\_owner, balance\_amount

#### Step 4: Sắp xếp các tập thực thể

1. Tập thực thể Customer: tập thực thể yếu
2. Tập thực thể Account: tập thực thể mạnh

#### Step 5: Sắp xếp các thuộc tính

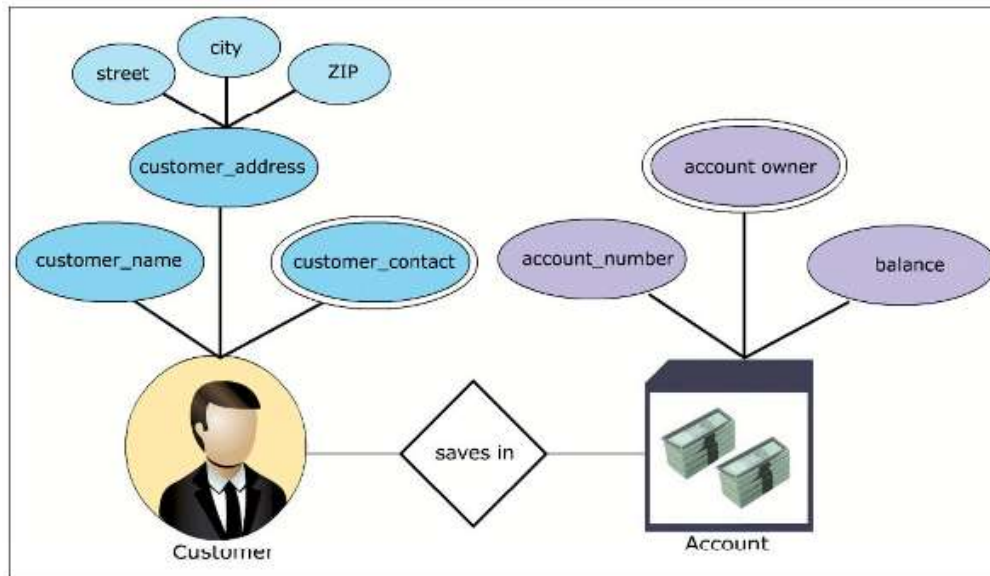
1. Customer entity set: customer\_address- composite, customer\_contact- multi-valued
2. Account entity set: account\_number -7 primary key, account\_owner- multi-valued

#### Step 6: Xác định các mối quan hệ

A customer 'saves in' an account. The relation is 'saves in'. (Một khách hàng ' gửi tiền trong' một tài khoản. Mối quan hệ là ' gửi tiền trong')

### Step 7: Vẽ biểu đồ bằng các biểu tượng

Hình 2.15 cho thấy biểu đồ E-R cho ngân hàng.



Hình 2.15: E-R Diagram for the Bank

## 2.4 Normalization ( Chuẩn hóa )

Ban đầu, tất cả các CSDL được đặc trưng bởi một số lượng lớn các cột và các bản ghi. Cách tiếp cận này có những hạn chế nhất định. Hãy xem xét các chi tiết sau đây về các nhân viên trong một bộ phận phòng ban. Table 2.3 bao gồm các chi tiết của nhân viên cũng như các chi tiết của dự án mà họ đang làm.

Emp_no	Project_id	Project_name	Emp_name	Grade	Salary
142	113, 124	BLUE STAR, MAGNUM	John	A	20,000
168	113	BLUE STAR	James	B	15,000
263	113	BLUE STAR	Andrew	C	10,000
109	124	MAGNUM	Bob	C	10,000

Table 2.3: Department Employee Details

### ➤ Repetition anomaly (Bất thường do lặp lại)

Dữ liệu như là Project\_id, Project\_name, Grade, và Salary lặp lại nhiều lần. Sự lặp lại này cản trở hiệu suất trong khi truy xuất dữ liệu, và dung lượng lưu trữ. Sự lặp lại này của dữ liệu được gọi là sự lặp lại bất thường



Sự lặp lại được cho thấy trong table 2.4 với sự trợ giúp của các ô màu xám

Emp_no	Project_id	Project_name	Emp_name	Grade	Salary
142	113, 124	BLUE STAR, MAGNUM	John	A	20,000
168	113	BLUE STAR	James	B	15,000
263	113	BLUE STAR	Andrew	C	10,000
109	124	MAGNUM	Bob	C	10,000

Table 2.4: Department Employee Details

➤ Insertion anomaly ( Bất thường do chèn )

Giả sử bộ phận phòng ban tuyển dụng một nhân viên mới tên là Ann. Ann chưa được giao bất kỳ dự án nào. Chèn các chi tiết về cô ấy vào trong bảng sẽ làm các cột Project\_id và Project\_name bị bỏ trống. Nếu để các cột đó rỗng, thì sẽ gây nên các vấn đề về sau. Sự bất thường được tạo ra bởi các việc chèn dữ liệu như vậy được gọi là các bất thường do chèn. Sự bất thường có thể được thấy trong table 2.5

Emp_no	Project_id	Project_name	Emp_name	Grade	Salary
142	113, 124	BLUE STAR, MAGNUM	John	A	20,000
168	113	BLUE STAR	James	B	15,000
263	113	BLUE STAR	Andrew	C	10,000
109	124	MAGNUM	Bob	C	10,000
195	-	-	Ann	C	10,000

Table 2.5: Department Employee Details

➤ Deletion anomaly ( Xóa bất thường )

Giả sử, Bob được rút ra khỏi dự án MAGNUM. Việc xóa bản ghi cũng sẽ xóa các chi tiết Emp\_no, Grade, và Salary của Bob. Việc mất dữ liệu này sẽ có hại bởi tất cả các chi tiết cá nhân của Bob cũng bị mất như được thấy trong bảng 2.6. Loại mất dữ liệu này do việc xóa được gọi là sự xóa bất thường. Có thể thấy sự bất thường này trong hình 2.6.

Emp_no	Project_id	Project_name	Emp_name	Grade	Salary
142	113, 124	BLUE STAR, MAGNUM	John Smith	A	20,000
168	113	BLUE STAR	James Kilber	B	15,000
263	113	BLUE STAR	Andrew Murray	C	10,000

Table 2.6: Employee Project Details

➤ Updating anomaly ( Bất thường do cập nhật )

Giả sử John đã được tăng lương, hoặc John bị giáng chức. Sự thay đổi trong mức lương, hoặc cấp bậc của John cần được phản ánh trong tất cả các dự án mà John đang làm việc. Vấn đề này trong cập nhật tất cả các sự việc xảy ra được gọi là bất thường do cập nhật.

Table Department Employee Details được gọi là bảng chưa chuẩn hóa. Những hạn chế đó dẫn đến nhu cầu thiết yếu cho việc chuẩn hóa.

Việc chuẩn hóa là quá trình loại bỏ các dư thừa, và phụ thuộc không mong muốn

Ban đầu, Codd (1972) đưa ra ba dạng chuẩn hóa (1NF, 2NF, và 3NF), tất cả đều dựa trên các phụ thuộc giữa các thuộc tính của một mối quan hệ. Dạng thứ tư, và thứ năm được dựa trên đa giá trị, và ghép các dependencies (phụ thuộc), và đã được đề xuất sau.

### 2.4.1 Dạng chuẩn hóa thứ nhất ( 1NF )

Để đạt được dạng chuẩn hóa thứ nhất, thì phải thực hiện các bước sau:

- Tạo các bảng riêng biệt cho mỗi nhóm dữ liệu liên quan
- Các cột của bảng phải có các giá trị nguyên tử
- Các thuộc tính chính phải được xác định

Hãy xem chi tiết dự án nhân viên ( Employee Project Details) được cho thấy trong table 2.7

Emp_no	Project_id	Project_name	Emp_name	Grade	Salary
142	113, 124	BLUE STAR, MAGNUM	John	A	20,000
168	113	BLUE STAR	James	B	15,000
263	113	BLUE STAR	Andrew	C	10,000
109	124	MAGNUM	Bob	C	10,000

Table 2.7: Employee Project Details

Bảng có dữ liệu liên quan đến các dự án, và nhân viên. Bảng cần được chia ra thành hai bảng, đó là một bảng Project Details, và một bảng Employee Details. Các cột Project\_id và Project\_names có nhiều giá trị. Dữ liệu cần được chia qua nhiều hàng khác nhau. Các bảng kết quả là Project Details, và Employee Details được cho thấy trong các tables 2.8 và 2.9

Project_id	Project_name
113	BLUE STAR
124	MAGNUM

Table 2.8: Project Details

Emp_no	Emp_name	Grade	Salary
142	John	A	20,000
168	James	B	15,000
263	Andrew	C	10,000
109	Bob	C	10,000

Table 2.9: Employee Details

Thuộc tính **Project\_id** là primary key cho bảng **Project Details**.

Thuộc tính Emp\_no là primary key ( khóa chính) cho bảng Employee Details. Do đó, trong dạng chuẩn hóa thứ nhất (1NF), bảng Employee Project Details ban đầu đã bị giảm thành các bảng Project Details và Employee Details.

## 2.4.2 Dạng chuẩn hóa thứ hai (2NF)

Các bảng được cho là trong dạng chuẩn hóa thứ hai (2NF) nếu:

- Chúng đáp ứng các yêu cầu của dạng chuẩn thứ nhất
- Không các sự phụ thuộc một phần nào trong các bảng
- Các bảng có liên quan thông qua các khóa ngoài ( foreign keys)

Sự phụ thuộc một phần có nghĩa là một thuộc tính không khóa, không phải là một phụ thuộc một phần trên nhiều hơn một thuộc tính khóa. Các bảng Project Details và Employee Details không thể hiện bất kỳ các sự phụ thuộc một phần nào. Project\_name chỉ phụ thuộc vào Project\_id và Emp\_name, Grade và Salary chỉ phụ thuộc vào Emp\_no. Các bảng cũng cần được liên kết thông qua các khóa ngoài. Một bảng thứ ba tên là Employee Project Details được tạo với hai cột duy nhất là Project\_id và Emp\_no.

Vì vậy, các bảng Project và Employee details trên việc chuyển đổi sang dạng chuẩn hóa thứ hai (2NF) tạo ra các bảng Project Details, Employee Details, và Employee Project Details như được thấy trong các tables 2.10, 2.11, và 2.12.

Project_id	Project_name
113	BLUE STAR
124	MAGNUM

Table 2.10: Project Details

Emp_no	Emp_name	Grade	Salary
142	John	A	20,000
168	James	B	15,000
263	Andrew	C	10,000
109	Bob	C	10,000

Table 2.11: Employee Details

Emp_no	Project_id
142	113
142	124
168	113
263	113
109	124

Table 2.12: Employee Project Details

Các thuộc tính Emp\_no và Project\_id của bảng Employee Project Details kết hợp với nhau để tạo thành khóa chính. Các khóa chính được gọi là các khóa chính phức hợp (composite primary keys)

### 2.4.3 Dạng chuẩn hóa thứ ba (3NF)

Để đạt dạng chuẩn hóa thứ ba:

- Các bảng phải đáp ứng các yêu cầu của dạng chuẩn hóa thứ hai
- Các bảng không nên có các sự phụ thuộc bắc cầu trong chúng

Các bảng Project Details, Employee Details, và Employee Project Details là trong dạng chuẩn hóa thứ hai. Nếu một thuộc tính có thể được xác định bởi một thuộc tính không khóa khác, nó được gọi là một sự phụ thuộc bắc cầu. Để đơn giản hóa, từng thuộc tính không khóa nên được xác định bởi một thuộc tính chính. Nếu một thuộc tính không khóa có thể được xác định bởi một thuộc tính không khóa khác, nó cần được đưa vào trong một bảng khác.

Theo quan sát trên các bảng khác nhau, nó được thấy rằng các bảng Project Details và Employee Project Details không thể hiện bất kỳ các sự phụ thuộc bắc cầu. Các thuộc tính không khóa được xác định hoàn toàn bởi các thuộc tính chính. Project\_name chỉ được xác định bởi Project\_number. Tiếp tục rà soát bảng Employee Details, một mâu thuẫn nhất định được thấy. Thuộc tính Salary được xác định bởi thuộc tính Grade, mà không phải thuộc tính chính Emp\_no. Do đó, sự phụ thuộc bắc cầu cần phải được loại bỏ.

Bảng Employee Details có thể được chia thành các bảng Employee Details và Grade Salary Details như được cho thấy trong các table 2.13 và 2.14.

Emp_no	Emp_name	Grade
142	John	A
168	James	B
263	Andrew	C
109	Bob	C

Table 2.13: Employee Details

Grade	Salary
A	20,000
B	15,000
C	10,000

Table 2.14: Grade Salary Details Table

Như vậy, vào cuối của ba giai đoạn chuẩn hóa. Bảng Employee Project Details đã bị gảm thành các bảng Project Details, Employee Project Details, Employee Details, và Grade Salary Details như được cho thấy trong các table 2.15, 2.16, 2.17, và 2.18.

Project_id	Project_name
113	BLUE STAR
124	MAGNUM

Table 2.15: Project Details

Emp_no	Project_id
142	113
142	124
168	113
263	113
109	124

Table 2.16: Employee Project Details

Emp_no	Emp_name	Grade
142	John	A
168	James	B
263	Andrew	C
109	Bob	C

Table 2.17: Employee Details

Grade	Salary
A	20,000
B	15,000
C	10,000

Table 2.18: Grade Salary Details

#### 2.4.4 Denormalization ( Phi chuẩn hóa )

Bằng cách chuẩn hóa một cơ sở dữ liệu, sự dư thừa được giảm đi. Điều này cũng làm giảm các yêu cầu lưu trữ cho cơ sở dữ liệu, và bảo đảm tính toàn vẹn dữ liệu. Tuy nhiên, nó có một số hạn chế như sau:

- Các truy vấn ghép phức tạp có thể phải được ghi thường xuyên để kết hợp các dữ liệu trong nhiều bảng
- Các liên kết có thể liên quan nhiều hơn ba bảng, tùy thuộc vào nhu cầu thông tin

Nếu như các liên kết được sử dụng rất thường xuyên, hiệu quả hoạt động của các cơ sở dữ liệu sẽ trở nên rất kém. Thời gian CPU cần để giải quyết các truy vấn như vậy sẽ rất lớn. Trong các trường hợp như vậy, việc lưu trữ một vài trường dữ liệu dư thừa có thể được bỏ qua, để tăng hiệu suất của CSDL. Cơ sở dữ liệu mà có những số lượng dư thừa nhỏ như vậy, để tăng hiệu suất được gọi là cơ sở dữ liệu phi chuẩn hóa, và quá trình làm việc như vậy được gọi là denormalization (Phi chuẩn hóa dữ liệu).

## 2.5 Các toán tử quan hệ

Mô hình quan hệ được dựa trên nền tảng vững chắc của Đại số quan hệ. Đại số quan hệ gồm một tập hợp các toán tử để tính toán trên các quan hệ. Mỗi toán tử lấy một hoặc hai quan hệ như là đầu vào và tạo ra một quan hệ mới như là đầu ra.

Hãy xem xét bảng **Branch Reserve Details** như được trình bày trong bảng 2.19.

Branch	Branch_id	Reserve (Billion €)
London	BS-01	9.2
London	BS-02	10
Paris	BS-03	15
Los Angeles	BS-04	50
Washington	BS-05	30

Table 2.19: Branch Reserve Details

### ➤ SELECT

Toán tử SELECT được sử dụng để trích xuất dữ liệu thỏa mãn một điều kiện nhất định. Chữ cái Hy Lạp sigma viết thường, 'σ', được sử dụng để biểu thị lựa chọn. Hoạt động lựa chọn, trên bảng **Branch Reserve Details**, để hiển thị chi tiết của các chi nhánh ở London sẽ cho kết quả trong table 2.20.

Branch	Branch_id	Reserve (Billion €)
London	BS-01	9.2
London	BS-02	10

Table 2.20: Details of Branches in London

Một lựa chọn trên bảng **Branch Reserve Details** để hiển thị các chi nhánh với trữ lượng lớn hơn 20 tỷ Euro sẽ cho kết quả trong table 2.21.

Branch	Branch_id	Reserve (Billion €)
Los Angeles	BS-04	50
Washington	BS-05	30

Table 2.21: Details of Branches with Reserves Greater Than 20 Billion Euros



➤ **PROJECT**

Toán tử PROJECT được sử dụng để đặt kế hoạch một số chi tiết của bảng quan hệ. Toán tử PROJECT chỉ hiển thị các chi tiết cần thiết để lại một số cột nhất định. Toán tử PROJECT được ký hiệu bằng chữ pi tiếng Hy Lạp, 'I'. Giả định rằng chỉ có số lượng **Branch\_id** và **Reserve** cần phải được hiển thị.

Để một phép toán lập kế hoạch làm như vậy, trên bảng **Branch Reserve Details**, sẽ cho kết quả trong bảng 2.22.

Branch_id	Reserve (Billion €)
BS-01	9.2
BS-02	10
BS-03	15
BS-04	50
BS-05	30

Table 2.22: Resultant Table with Branch\_id And Reserve Amounts

➤ **PRODUCT**

Toán tử PRODUCT, được ký hiệu bằng 'x' giúp kết hợp thông tin từ hai bảng quan hệ. Xem xét table 2.23.

Branch_id	Loan Amount ( Billion €)
BS-01	0.56
BS-02	0.84

Table 2.23: Branch Loan Details

Phép tính product trên bảng **Branch Reserve Details** và **Branch Loan Details** sẽ cho ra kết quả trong table 2.24.

Branch	Branch_id	Reserve (Billion €)	Loan Amount ( Billion €)
London	BS-01	9.2	0.56
London	BS-01	9.2	0.84
London	BS-02	10	0.56
London	BS-02	10	0.84
Paris	BS-03	15	0.56
Paris	BS-03	15	0.84
Los Angeles	BS-04	50	0.56
Los Angeles	BS-04	50	0.84
Washington	BS-05	30	0.56
Washington	BS-05	30	0.84

Table 2.24: Product of Branch Reserve Details and Branch Loan Details

Phép tính tích số kết hợp từng bản ghi từ bảng thứ nhất với tất cả bản ghi trong bảng thứ hai, đến lúc tạo ra tất cả các tổ hợp có thể giữa các bản ghi của bảng.

### ➤ UNION ( Phép tính tích số )

Giả sử một quan chức của ngân hàng có dữ liệu được đưa ra trong bảng 2.19 và 2.23 muốn biết chi nhánh nào có lượng dự trữ dưới 20 tỷ Euro hoặc cho vay. Bảng kết quả sẽ bao gồm các chi nhánh với lượng dự trữ dưới 20 tỷ Euro, cho vay hoặc cả hai.

Điều này cũng tương tự như sự kết hợp của hai bộ dữ liệu; đầu tiên, tập hợp các chi nhánh với lượng dự trữ ít hơn 20 tỷ Euro và thứ hai, các chi nhánh có cho vay. Chi nhánh với cả hai, dự trữ dưới 20 tỷ Euro và cho vay sẽ được hiển thị chỉ một lần. Toán tử UNION không chỉ như vậy, nó còn thu thập các dữ liệu từ các bảng khác nhau và trình bày một phiên bản hợp nhất của dữ liệu hoàn chỉnh. Phép tính hợp nhất được thể hiện bằng biểu tượng, 'U'. Phép tính union trên bảng **Branch Reserve Details** và **Branch Loan Details** sẽ tạo ra bảng 2.25.

Branch	Branch_id
London	BS-01
London	BS-02
Paris	BS-03

Table 2.25: Unified Representation of Branches with Less Reserves or Loans

### ➤ INTERSECT ( Phép giao )

Giả sử cùng một viên chức sau khi nhìn thấy dữ liệu này muốn biết những chi nhánh nào trong số này có cả lượng dự trữ thấp và cho vay. Câu trả lời sẽ là phép tính quan hệ giao cắt. Toán tử INTERSECT tạo ra dữ liệu giữ đúng trong tất cả các bảng được áp dụng trên đó. Nó dựa trên lý thuyết tập giao điểm và được thể hiện bằng biểu tượng '∩'. Kết quả của giao điểm của bảng **Branch Reserve Details** và **Branch Loan Details** sẽ là danh sách các chi nhánh có cả lượng dự trữ dưới 20 tỷ Euro và cho vay trong tài khoản của họ. Bảng kết quả được tạo ra là bảng 2.26.

Branch	Branch_id
London	BS-01
London	BS-02

Table 2.26: Branches with Low Reserves and Loans

### ➤ DIFFERENCE ( Khác biệt )

Nếu cùng một viên chức bây giờ muốn danh sách các chi nhánh có lượng dự trữ thấp, nhưng không cho vay, thì sau đó viên chức này sẽ phải sử dụng phép toán khác biệt. Toán tử DIFFERENCE, được ký hiệu là '-', tạo ra dữ liệu từ các bảng khác nhau, nhưng nó tạo ra dữ liệu giữ true trong một bảng này, và không phải trong bảng kia. Do đó, chi nhánh sẽ phải có lượng dự trữ thấp, và các khoản cho vay không được hiển thị

Table 2.27 là kết quả được tạo ra.

Branch	Branch_id
Paris	BS-03

**Table 2.27: Branches with Low Reserves but No Loans**

### ➤ JOIN ( nối )

Phép toán JOIN là phần nâng cao của phép toán sản phẩm. Nó cho phép một lựa chọn được thực hiện trên tích số của các bảng. Ví dụ, nếu các giá trị dự trữ và các khoản cho vay của chi nhánh có trữ lượng thấp và giá trị cho vay là cần thiết, sản phẩm của **Branch Reserve Details** và **Branch Loan Details** sẽ được yêu cầu. Một khi sản phẩm của bảng 2.19 và 2.23 được tạo ra, chỉ có những chi nhánh này sẽ được liệt kê trong đó có cả dự trữ dưới 20 tỷ Euro và cho vay. Bảng 2.28 được tạo ra như là kết quả của phép tính JOIN.

Branch	Branch_id	Reserve (Billion €)	Loan Amount (Billion €)
London	BS-01	9.2	0.56
London	BS-02	10	0.84

**Table 2.28: Detailed List of Branches with Low Reserve and Loans**

### ➤ DIVIDE

Giả sử một viên chức muốn xem tên các chi nhánh và lượng dự trữ của tất cả các chi nhánh có tiền vay. Quy trình này có thể được thực hiện rất dễ dàng bằng cách sử dụng toán tử DIVIDE. Tất cả những gì viên chức cần phải làm là chia bảng **Branch Reserve Details** (được trình bày trước đây trong bảng 2.19) bằng danh sách các chi nhánh, có nghĩa là, cột **Branch Id** của bảng **Branch Loan Details** (được trình bày trước đó trong bảng 2.23). Bảng 2.29 là kết quả được tạo ra.

Branch	Reserve (Billion €)
London	9.2
London	10

**Table 2.29: Resultant Table of Division Operation**

Lưu ý là các thuộc tính của bảng chia sẽ luôn là tập hợp phụ của bảng bị chia. Bảng kết quả sẽ luôn luôn không có giá trị của các thuộc tính của bảng số chia và các bản ghi không phù hợp với các bản ghi trong bảng số chia.

## 2.6 Kiểm tra tiến độ của bạn

1. Một hoặc nhiều thuộc tính có thể định nghĩa duy nhất một thực thể từ một tập thực thể được gọi là một \_\_\_\_\_ key

(A)	Primary	(C)	Alternate
(B)	Foreign	(D)	Super

2. Một thuộc tính chứa hai, hoặc nhiều giá trị thuộc tính trong nó được gọi là một thuộc tính \_\_\_\_\_.

(A)	Derived ( dẫn xuất )	(C)	Multi-valued ( đa giá trị )
(B)	Composite ( phức hợp )	(D)	Network ( mạng )

3. Phụ thuộc bắc cầu bị loại bỏ trong dạng chuẩn \_\_\_\_\_.

(A)	First ( đầu tiên )	(C)	Third ( thứ ba )
(B)	Second ( thứ hai )	(D)	Fourth ( thứ tư )

4. Phép tính nào được tăng cường hơn nữa trong phép tính Product ( Sản phẩm ) ?

(A)	Divide ( chia )	(C)	Difference ( khác biệt )
(B)	Intersection ( Giao cắt )	(D)	Join ( nối )

5. Điều nào sau đây là những thành phần cơ bản của mô hình E-R?
- a. Thực thể
  - b. Mối quan hệ
  - c. Các thuộc tính
  - d. Biểu đồ mối quan hệ
  - e. Tập quan hệ

(A)	a, b, c	(C)	a, c, e
(B)	a, d, c	(D)	a, b, c, e

### 2.6.1 Đáp án

1.	A
2.	B
3.	C
4.	D
5.	D



## Tóm tắt

- Việc tạo mô hình dữ liệu là quá trình áp dụng một mô hình dữ liệu thích hợp cho dữ liệu có sẵn.
- Mô hình E-R xem thế giới thực như là một tập các đối tượng cơ bản và các mối quan hệ giữa chúng
- Thực thể, thuộc tính, tập thực thể, mối quan hệ, và tập mối quan hệ hình thành năm thành phần cơ bản của mô hình E-R.
- Ánh xạ các bản số thể hiện số lượng các thực thể mà một thực thể có liên quan đến.
- Quá trình gỡ bỏ dữ liệu dư thừa khỏi các bảng của một cơ sở dữ liệu quan hệ được gọi là tiêu chuẩn hóa.
- Đại số quan hệ bao gồm tập hợp các toán tử để giúp truy vấn dữ liệu từ các cơ sở dữ liệu quan hệ.
- **SELECT, PRODUCT, UNION, và DIVIDE** là một số toán tử đại số quan hệ.



## *Entity-Relationship (E-R) Model and Normalization*



The background of the slide is a grayscale, high-contrast image. It features a close-up of a computer keyboard with keys labeled "CTRL", "SHIFT", and "BACK TAB" visible. Overlaid on the keyboard is a large, semi-transparent magnifying glass. Below the keyboard, there is a faint, detailed image of a computer circuit board with various components and traces. The overall composition suggests a focus on technology and learning.

*Learning how to learn is  
life's most important skill.*