



# Quản Trị Dữ Liệu Với Microsoft SQL Server

## Chương: 2

### Mô hình E-R và Chuẩn hóa (E-R Model and Normalization)

- Định nghĩa và mô tả về việc mô hình hoá dữ liệu
- Xác định và mô tả các thành phần của mô hình E-R (mô hình quan hệ thực thể) .
- Xác định mối quan hệ giữa các thực thể.
- Giải thích các sơ đồ E-R và các hữu ích của chúng.
- Mô tả biểu đồ, các ký hiệu được sử dụng để vẽ và chỉ ra các mối quan hệ khác nhau.
- Mô tả các dạng chuẩn khác nhau
- Tổng quan sử dụng các Toán tử quan hệ khác nhau

- Mô hình dữ liệu
  - Là nhóm các công cụ lý thuyết dùng để mô tả dữ liệu, các mối quan hệ và ngữ nghĩa giữa chúng.
  - Nó cũng bao gồm cả mô tả các ràng buộc toàn vẹn mà dữ liệu phải tuân theo.
- Mô hình thực thể liên kết, mô hình mạng, mô hình phân cấp, mô hình quan hệ là ví dụ về các mô hình dữ liệu.
- Sự phát triển của mỗi csdl bắt đầu từ bước là phân tích dữ liệu của nó và chọn ra một mô hình dữ liệu tốt nhất để biểu diễn nó.
- Khi bước này được hoàn thành, mô hình dữ liệu được áp dụng tới dữ liệu.

# Mô hình hóa dữ liệu 1-2

Quá trình áp dụng một mô hình dữ liệu phù hợp tới dữ liệu, để tổ chức và cấu trúc nó, được gọi là mô hình hóa dữ liệu.

Mô hình hoá dữ liệu được chia nhỏ thành ba bước:

## Mô hình mức khái niệm (Conceptual Data Modeling)

- Người mô hình hóa dữ liệu xác định mức cao nhất của các mối quan hệ trong dữ liệu.

## Mô hình mức luận lý (Logical Data Modeling)

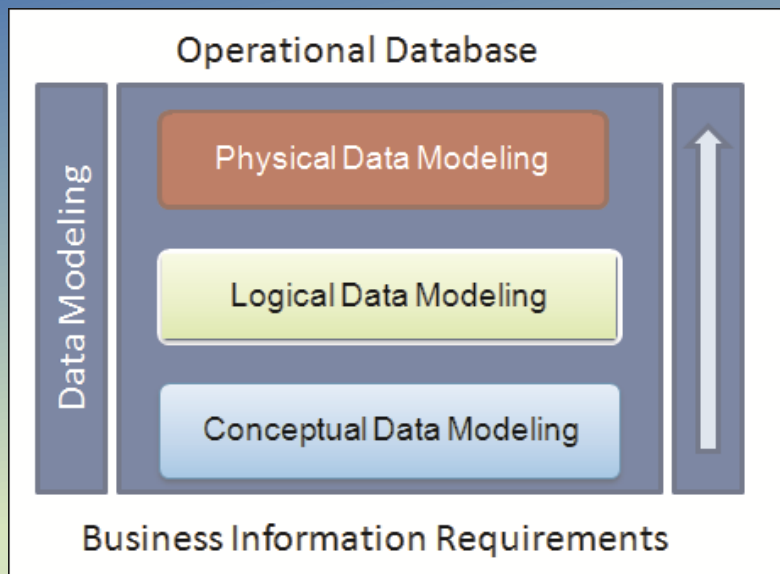
- Người mô hình hóa dữ liệu mô tả dữ liệu và các mối quan hệ giữa chúng một cách chi tiết
- Người mô hình hóa dữ liệu tạo một mô hình luận lý của csdl.

## Mô hình mức vật lý (Physical Data Modeling)

- Người mô hình hóa chỉ ra cách mô hình luận lý được hiện thực hóa một cách vật lý.

# Mô hình hóa dữ liệu 2-2

- **Mô hình hoá dữ liệu được chia nhỏ thành ba bước**



- **Các mô hình có thể phân thành các nhóm khác nhau như sau:**

Các mô hình  
luận lý dựa  
trên đối tượng

Các mô hình  
luận lý dựa trên  
bản ghi

Các mô  
hình vật lý

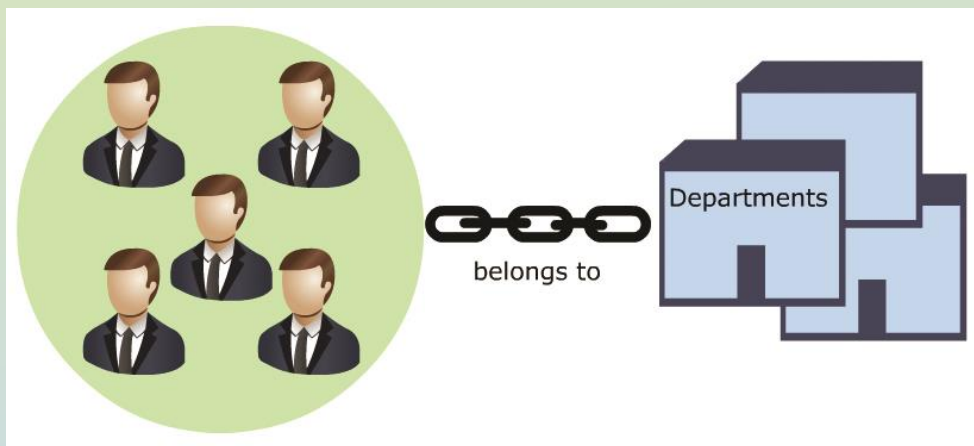
# Mô hình thực thể - liên kết (E-R) 1-11

Mô hình thực thể - liên kết (E-R) thuộc phân loại đầu tiên

Dữ liệu có thể được hiểu như là các đối tượng có trong thế giới thực, còn gọi là thực thể và tồn tại các mối quan hệ giữa các thực thể này.

Ví dụ, trong một tổ chức, nhân viên và phòng ban là các đối tượng trong thế giới thực. Một nhân viên thuộc về một phòng ban.

Do đó, mỗi quan hệ 'thuộc về' liên kết (links) một nhân viên tới một phòng ban cụ thể. Mỗi quan hệ Nhân viên – phòng ban có thể được mô hình hóa như trình bày trong hình dưới đây:



# Mô hình thực thể - liên kết (E-R) 2-11

➤ Mô hình E-R bao gồm năm thành phần cơ bản:

## Thực thể

- Một thực thể là một đối tượng trong thế giới thực, tồn tại khách quan và có thể phân biệt với các đối tượng khác.
- Ví dụ: nhân viên, phòng ban, phương tiện giao thông và tài khoản.

## Mối quan hệ

- Một mối quan hệ là một sự kết hợp hay ràng buộc mà tồn tại giữa hai hay nhiều thực thể.
- Ví dụ mối quan hệ: thuộc về, sở hữu, làm việc cho, tiết kiệm(, saves in), vv...

## Thuộc tính

- Thuộc tính là các đặc điểm mà thực thể có. Thuộc tính giúp nhận dạng/phân biệt được thực thể này với các thực thể khác.
- Ví dụ: các thuộc tính để nhận diện sinh viên là mã sinh viên, tên, ngày sinh,...

## Tập thực thể

- Một tập thực thể là một tập của các thực thể giống nhau (có các thuộc tính giống nhau).
- Ví dụ, các nhân viên của một tổ chức tạo thành một tập thực thể được gọi là tập nhân viên.

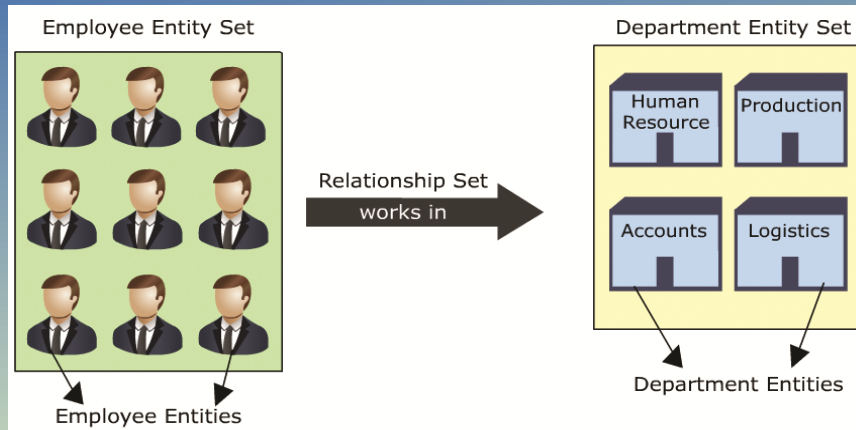
## Tập mối quan hệ

- Một tập các mối quan hệ giống nhau giữa hai hay nhiều tập thực thể được gọi là tập mối quan hệ.
- Ví dụ: tập tất cả các mối quan hệ 'làm việc ở' giữa các nhân viên và phòng ban được gọi là tập mối quan hệ 'work in'.



# Mô hình thực thể - liên kết (E-R) 3-11

- Các thành phần khác nhau của mô hình E-R được nhìn thấy trong hình dưới đây:

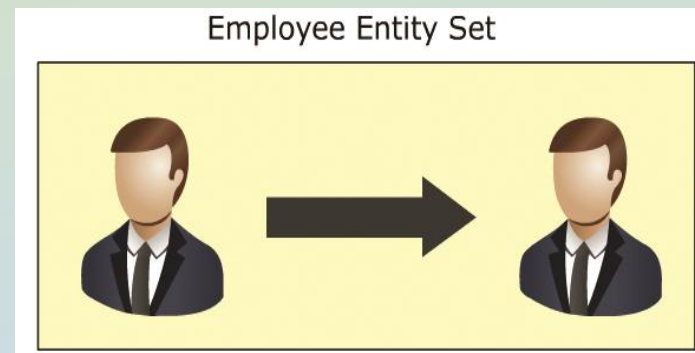


- Các mối quan hệ liên kết (associate) một hoặc nhiều thực thể có thể chia thành ba kiểu như sau:

## Mối quan hệ đệ quy

- Mối quan hệ giữa các thực thể trong cùng một tập thực thể được gọi là mối quan hệ đệ quy.
- Ví dụ một thành viên của nhóm làm việc cho người quản lý.
- Quan hệ 'làm việc cho' tồn tại giữa hai thực thể nhân viên khác nhau trong cùng một tập nhân viên.

- Có thể xem mối quan hệ trong hình sau:



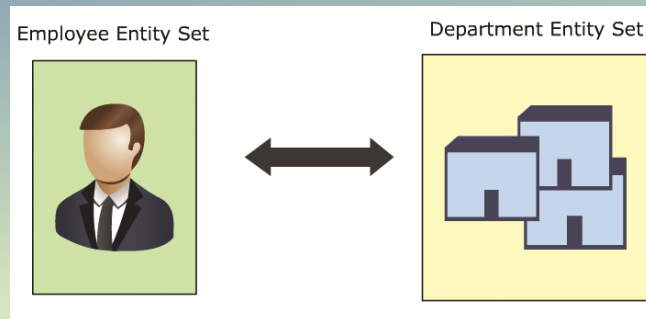


# Mô hình thực thể - liên kết (E-R) 4-11

## Mối quan hệ nhị phân

- Các mối quan hệ giữa các thực thể của hai tập thực thể được gọi là quan hệ nhị phân. (có 2 tập thực thể tham gia trong mối quan hệ)
- Ví dụ, nhân viên Hoa thuộc về một phòng Kế toán.
- Nhân viên Hoa là một thực thể thuộc về tập thực thể Nhân viên. Phòng Kế toán là một thực thể thuộc về tập thực thể Phòng ban.

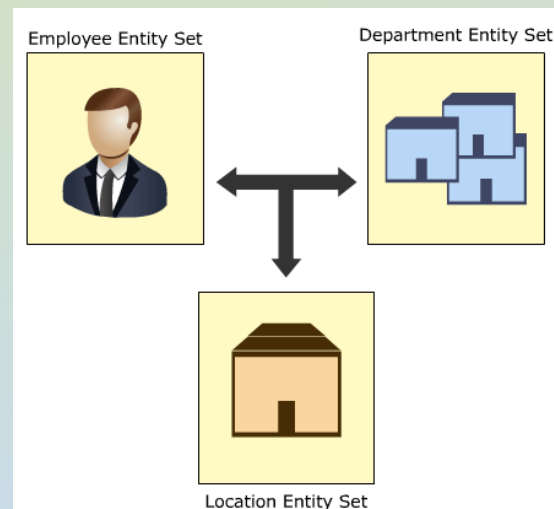
- Mỗi quan hệ nhị phân được minh họa trong hình hình sau:



## Mối quan hệ tam phân

- Mỗi quan hệ tồn tại giữa ba thực thể của tập thực thể khác nhau được gọi là mối quan hệ tam phân.
- Ví dụ, một nhân viên làm việc trong bộ phận kế toán tại chi nhánh khu vực.
- Mỗi quan hệ, 'làm việc' tồn tại giữa cả ba, nhân viên, các bộ phận, và vị trí.

- Mỗi quan hệ tam phân được minh họa trong hình hình sau:

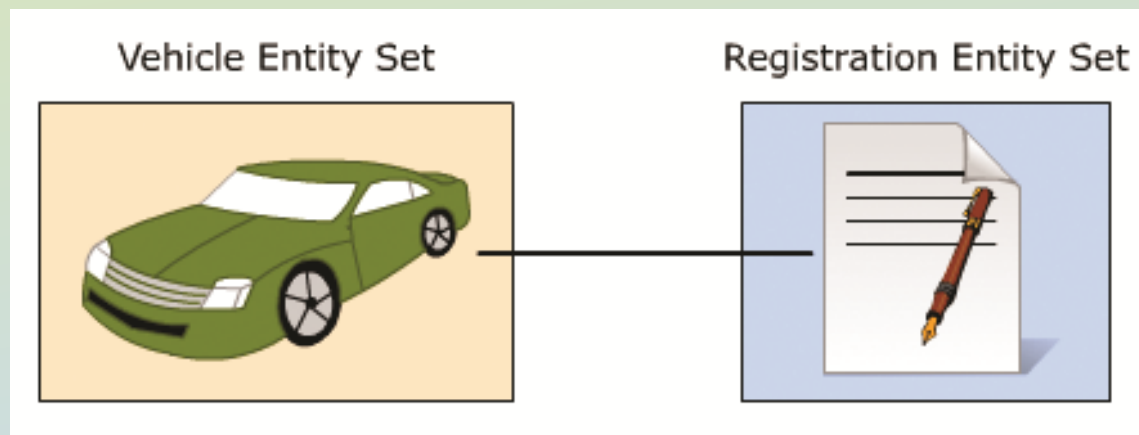


# Mô hình thực thể - liên kết (E-R) 5-11

- Các mối quan hệ có thể được phân loại là ánh xạ các lực lượng:

## Một – Một

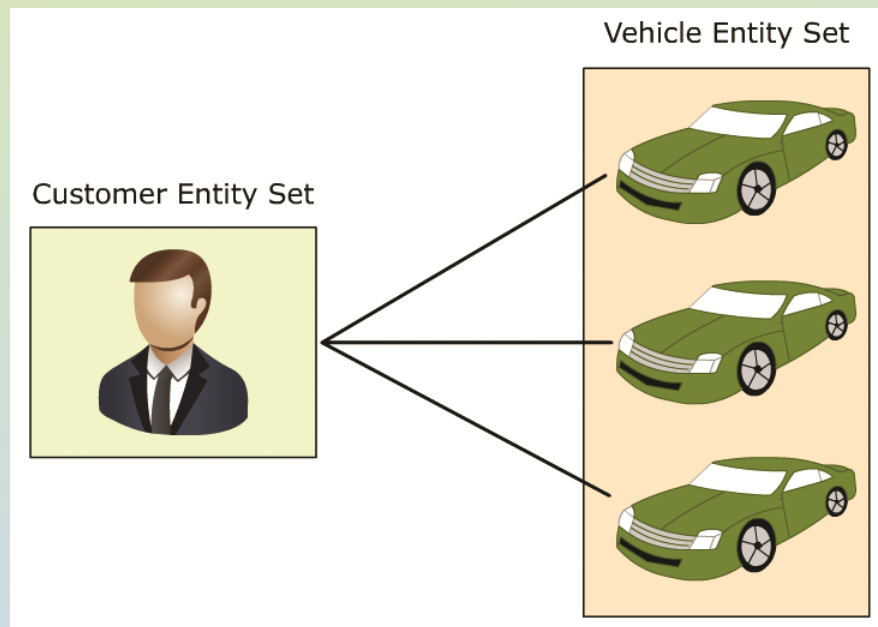
- Kiểu ánh xạ này tồn tại khi một thực thể của tập thực thể A có thể có liên quan(associated) đến một và chỉ một thực thể trong tập thực thể B.
- Ví dụ mỗi xe(vehicle) chỉ có duy nhất một đăng ký.
- Không thể có hai xe có cùng một thông tin đăng ký.
- Mối quan hệ là một – một, đó là một xe – một đăng ký.



# Mô hình thực thể - liên kết (E-R) 6-11

## Một - Nhiều

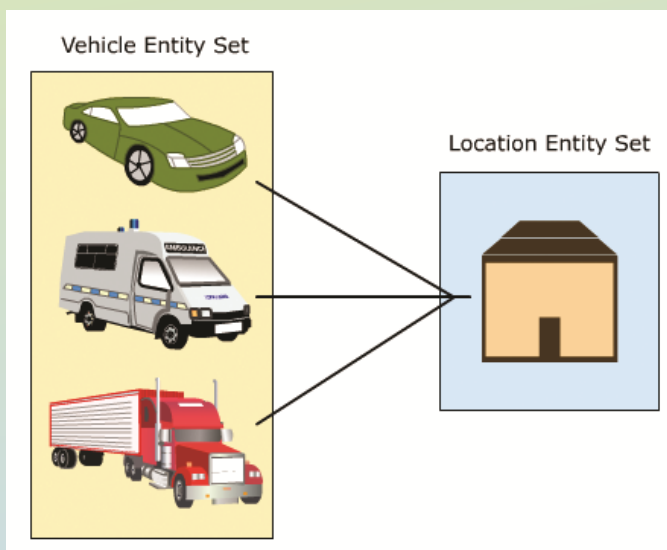
- Đây là kiểu ánh xạ này tồn tại khi một thực thể của tập thực thể A có thể có liên quan(associated) đến một hoặc nhiều thực thể trong tập thực thể B.
- Ví dụ một khách hàng có thể có nhiều một xe (vehicle).
- Do vậy, việc ánh xạ này là ánh xạ một tới nhiều, đó là một khách – (có) – một hoặc nhiều xe.
- Xem ví dụ mối quan hệ một- nhiều trong hình bên:



# Mô hình thực thể - liên kết (E-R) 7-11

## Nhiều - Một

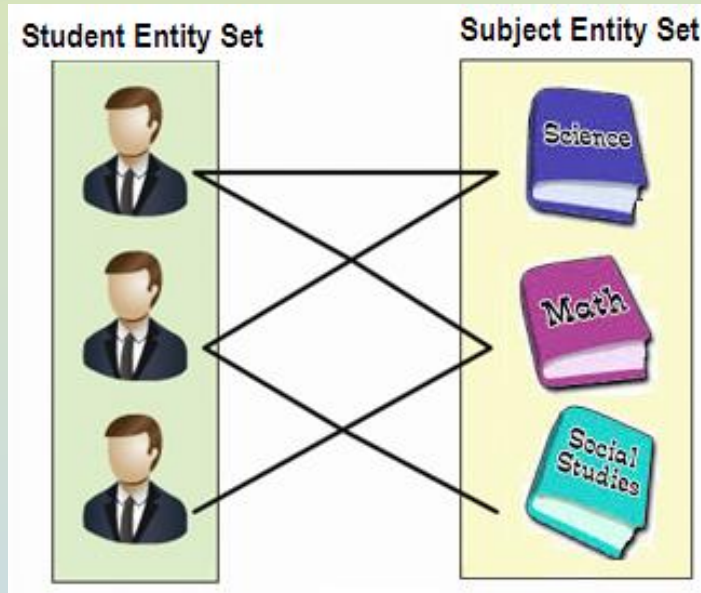
- Đây là kiểu ánh xạ tồn tại khi có nhiều thực thể của một tập thực thể A có liên quan đến một thực thể của tập thực thể B.
- Ví dụ, mỗi chiếc xe chỉ do một công ty sản xuất, nhưng công ty đó có thể sản xuất nhiều loại xe.
- Hình sau đây minh họa ánh xạ Nhiều - Một:



# Mô hình thực thể - liên kết (E-R) 8-11

## Nhiều – Nhiều

- Đây là kiểu ánh xạ tồn tại khi có nhiều thực thể của một tập thực thể A có liên quan đến nhiều thực thể của tập thực thể B.
- Ví dụ, một sinh viên có thể học nhiều môn và một môn có thể có nhiều sinh viên học.
- Do đó, ánh xạ Nhiều – Nhiều là một hoặc nhiều khách hàng liên kết(associated )với một hoặc nhiều môn học.
- Hình sau đây minh họa ánh xạ Nhiều – Nhiều:



# Mô hình thực thể - liên kết (E-R) 9-11

- Sau đây là một số khái niệm bổ sung trong mô hình:

## Khóa chính – Primary Keys

- Khóa chính là một thuộc tính hay một tập hợp các thuộc tính của tập thực thể E có thể dùng để phân biệt hai thực thể bất kỳ trong tập thực thể E. Nói cách khác khóa là cơ sở để nhận dạng cho mỗi thực thể bên trong loại thực thể
- Dưới đây là bảng chứa thông tin chi tiết của các sinh viên trong một trường học.

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

### Student Details

- Mỗi sinh viên có một mã số (như là enrollment\_number), là mã duy nhất cấp cho mỗi sinh viên.
- Sinh viên bất kỳ có thể được nhận diện dựa trên mã số.
- Do đó, thuộc tính enrollment\_number đóng vai trò là khóa chính (primary key) trong bảng Student Details.



## Tập thực thể yếu

- Tập thực thể không có đủ các thuộc tính để thiết lập một khóa chính được gọi là tập thực thể yếu.

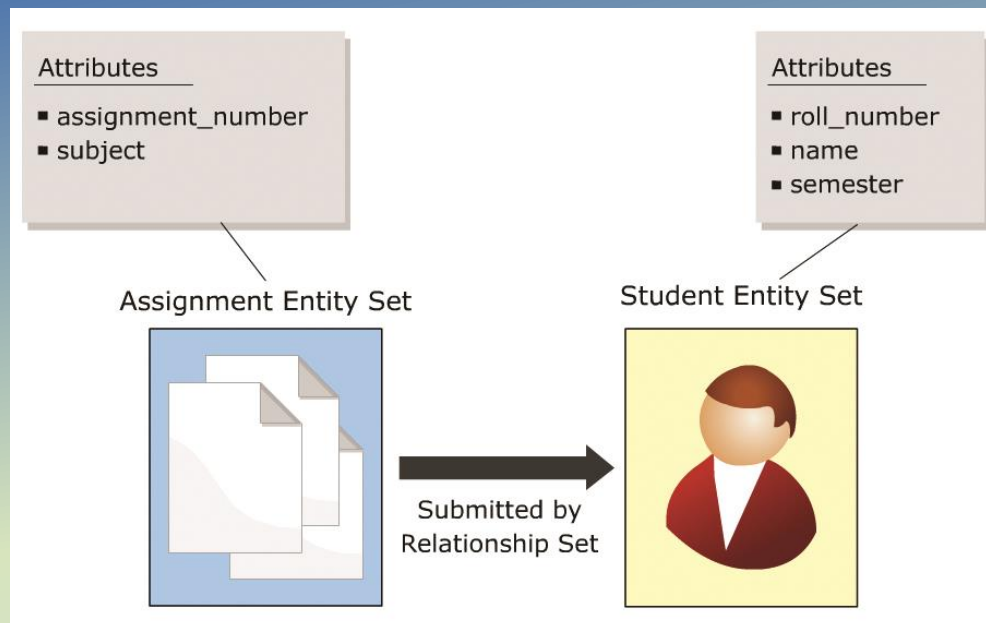
## Tập thực thể mạnh

- Tập thực thể có đủ các thuộc tính để thiết lập một khóa chính được gọi là tập thực thể mạnh.
- Xem xét kịch bản ở một cơ sở giáo dục, là nơi vào cuối mỗi học kỳ, sinh viên được yêu cầu phải hoàn thành và nộp một bài tập lớn(assignment).
- Giáo viên theo dõi những bài tập được nộp bởi các sinh viên.
- Một bài tập lớn và một sinh viên có thể được coi như 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 các thuộc tính **roll\_number**, **name**, và **semester**(học kỳ).
- 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 thành một tập thực thể sinh viên.
- Các tập thực thể có mối liên kết là 'nộp bởi'.



# Mô hình thực thể - liên kết (E-R) 11-11











➤ Mỗi quan hệ được mô tả như hình sau:



- Các thuộc tính, **assignment\_number** và **subject**, không đủ để nhận diện một thực thể bài tập (assignment) là duy nhất.
- Chỉ cần một thuộc tính **roll\_number** cũng đủ để nhận diện một thực thể sinh viên bất kỳ là duy nhất. Do đó, **roll\_number** là khóa chính cho tập thực thể sinh viên (student).
- Tập thực thể bài tập (assignment) là một tập thực thể yếu do thiếu khóa chính.
- Tập thực thể sinh viên (student) là tập thực thể mạnh do có sự hiện diện của thuộc tính **roll\_number** attribute.

# Sơ đồ thực thể - liên kết (ERD) 1-7

- Sơ đồ E-R (ERD) là sự biểu diễn đồ họa của mô hình E-R.
- Sơ đồ E-R (ERD) sử dụng nhiều kí hiệu hình vẽ khác nhau để giúp cho việc biểu diễn các thành phần khác nhau của mô hình thực thể - liên kết.
- Bảng liệt kê các hình vẽ ký hiệu dùng để biểu diễn cho các thành phần khác nhau trong sơ đồ:

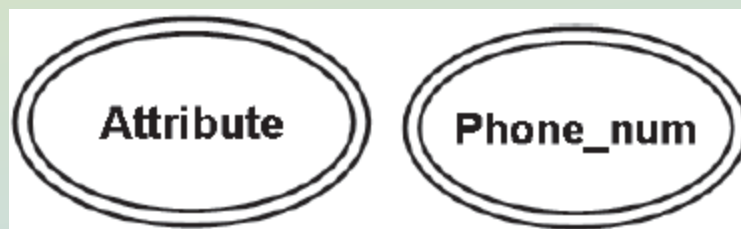
Component	Symbol	Example
Entity		
Weak Entity		
Attribute		
Relationship		
Key Attribute		

# Sơ đồ thực thể - liên kết (ERD) 2-7

- Thuộc tính trong mô hình ER có thể được phân thành nhiều loại như:

## Đa trị (Multi-valued)

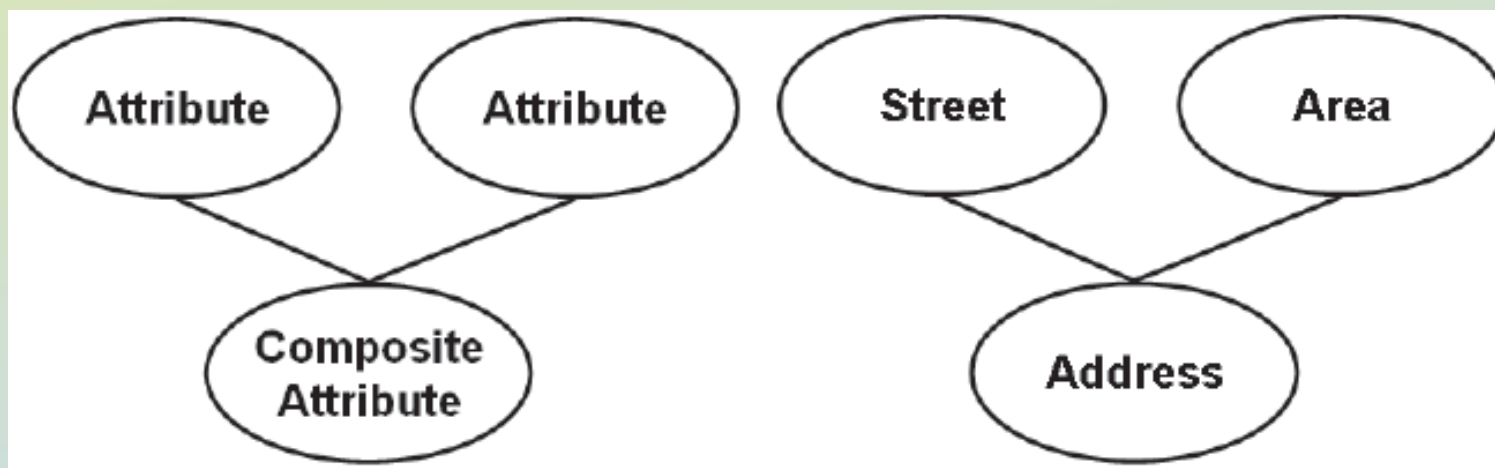
- Một thuộc tính đa trị (multi-valued) là thuộc tính có nhiều hơn một giá trị cho một thực thể. Nó được minh họa bằng một hình elip nét đôi.
- Thuộc tính này có thể có giới hạn(cận) trên, giới hạn(cận) dưới cho bất cứ giá trị thực thể nào
- Thuộc tính số điện thoại của một cá nhân có thể có nhiều hơn một giá trị, vì một cá nhân có một hoặc nhiều số điện thoại.
- Do vậy, thuộc tính số điện thoại là thuộc tính đa trị.
- Kí hiệu và ví dụ về thuộc tính đa trị:



# Sơ đồ thực thể - liên kết (ERD) 3-7

## Kết hợp (Composite)

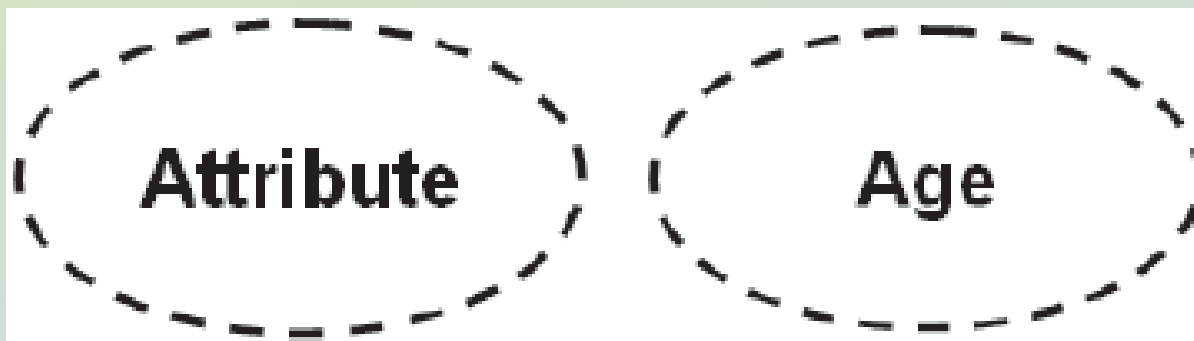
- Một thuộc tính kết hợp, tự nó có thể chứa hai hoặc nhiều thuộc tính, đại diện cho các thuộc tính cơ bản có ý nghĩa độc lập của riêng chúng.
- Thuộc tính địa chỉ là một thuộc tính kết hợp, nó được kết hợp từ các thuộc tính số nhà, phố, phường,...
- Sau đây là ví dụ và kí hiệu về thuộc tính kết hợp:



# Sơ đồ thực thể - liên kết (ERD) 4-7

## Dẫn xuất - Derived

- Thuộc tính dẫn xuất là các thuộc tính mà giá trị của nó phụ thuộc hoàn toàn các thuộc tính khác (giá trị của nó được tính toán từ các thuộc tính khác). Thuộc tính dẫn xuất được biểu diễn bằng các hình elip có nét đứt.
- Thuộc tính tuổi của một người là một ví dụ về thuộc tính dẫn xuất.
- Với mỗi thực thể con người cụ thể, tuổi có thể được xác định dựa trên ngày hiện tại với ngày sinh của người đó.
- Sau đây là ví dụ và kí hiệu về thuộc tính dẫn xuất:



# Sơ đồ thực thể - liên kết (ERD) 5-7

➤ Sau đây là các bước xây dựng một sơ đồ thực thể:

1

- Thu thập tất cả các dữ liệu mà cần được mô hình.

2

- Nhận diện dữ liệu là các thực thể trong thế giới thực có thể được mô hình.

3

- Xác định các thuộc tính cho mỗi thực thể.

4

- Sắp xếp các tập thực thể yếu, các tập thực thể mạnh.

5

- Sắp xếp các thuộc tính thực thể như là các thuộc tính khoá, các thuộc tính đa trị, các thuộc tính kết hợp, các thuộc tính dẫn suất.

6

- Xác định các mối quan hệ giữa các thực thể.

# Sơ đồ thực thể - liên kết (ERD) 6-7

- Xem xét kịch bản của một ngân hàng, với khách hàng và tài khoản. Sơ đồ ER cho các kịch bản có thể được xây dựng như sau:

## Bước 1: Thu thập dữ liệu

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

## Bước 2: Nhận diện các thực thể

- Khách hàng (Customer)
- Tài khoản (Account)

## Bước 3: Nhận diện các thuộc tính

- Khách hàng(Customer): customer\_name, customer\_address, customer\_contact
- Tài khoản(Account): account\_number, account\_owner, balance\_amount

## Bước 4: Sắp xếp các tập thực thể

- Tập thực thể Khách hàng (Customer): là tập thực thể yếu
- Tập thực thể Tài khoản(Account): là tập thực thể mạnh

## Bước 5: Sắp xếp các thuộc tính

- Tập thực thể Khách hàng: có customer\_address – là loại thuộc tính kết hợp, và customer\_contact – là thuộc tính đa trị.
- Tập thực thể Tài khoản: có account\_number → là khóa chính, và account\_owner – là đa trị



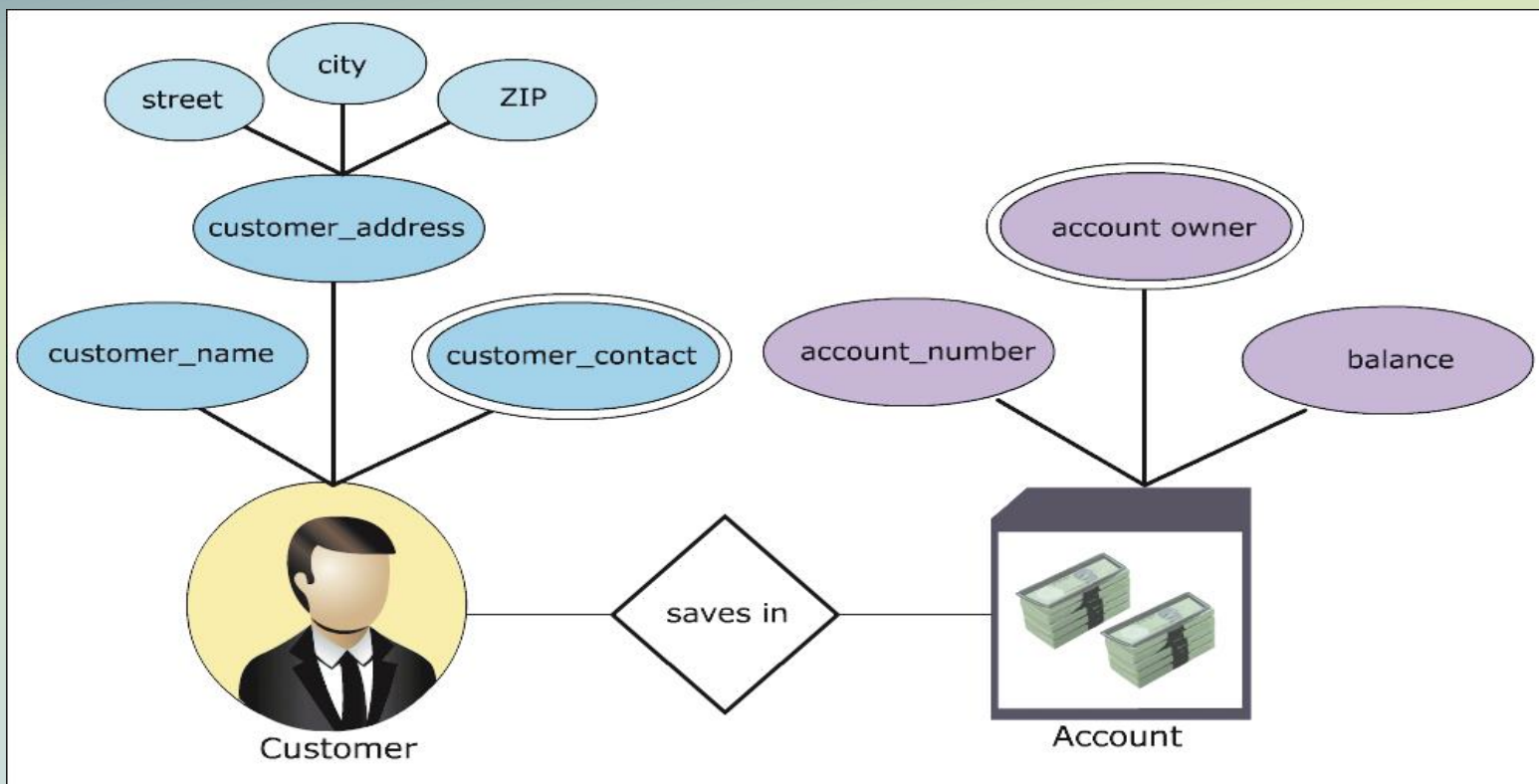
# Sơ đồ thực thể - liên kết (ERD) 7-7

**Bước 6: Nhận diện các quan hệ**

- Một khách hàng 'gửi tiết kiệm trong' một tài khoản. Vậy mối quan hệ giữa thực thể trong tập Khách hàng với thực thể trong tập Tài khoản là mối quan hệ 'gửi tiết kiệm' (saves in).

**Bước 7: Sử dụng kí hiệu vẽ sơ đồ**

- Sơ đồ ER (ERD):



# Chuẩn hóa (Normalization) 1-4

- Ban đầu, tất cả các csdl được đặc trưng bởi số lượng lớn các cột và các bản ghi.
- Phương pháp này có một số hạn chế.
- Bảng dưới đây chứa thông tin chi tiết của các nhân viên và dự án mà họ đang tham gia.

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

- Một số vấn đề dị thường (anomaly) của csdl chưa chuẩn hóa:

## Dị thường trùng lặp

- Dữ liệu về **Project\_id**, **Project\_name**, **Grade**, và **Salary** lặp đi lặp lại nhiều lần.
- Việc trùng lặp này làm ảnh hưởng đến hiệu suất của cả hai là quá trình là lấy dữ liệu và khả năng lưu trữ dữ liệu.
- Sự trùng lặp dữ liệu này được gọi là dị thường trùng lặp.

# Chuẩn hóa (Normalization) 2-4

- Sự trùng lặp dữ liệu được thể hiện ở bảng dưới đây, được đánh dấu bằng 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

## Dị thường khi chèn

- Giả sử phòng ban tuyển một nhân viên mới có tên là **Ann**.
- **Ann** vẫn chưa được phân công tham gia dự án nào. Khi chèn thông tin của **Ann** vào bảng sẽ để trống các cột **Project\_id** và **Project\_name**.
- Việc để các cột trống có thể dẫn đến các vấn đề sau này.
- Các dị thường phát sinh khi chèn dữ liệu được gọi là dị thường khi chèn , được trình bày trong bảng dưới đây:

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

# Chuẩn hóa (Normalization) 3-4

## Dị thường khi xóa

- Giả sử Bob rời khỏi dự án MAGNUM (do nghỉ việc).
- Việc xóa bản ghi cũng đồng nghĩa là xóa thông tin **Emp\_no**, **Grade**, và **Salary** của Bob.
- Việc mất dữ liệu này làm mất toàn bộ thông tin cũ nhân của Bob.
- Mất dữ liệu khi xóa được gọi là dị thường khi xóa, có thể xem trong bảng sau :

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

## Dị thường khi cập nhật

- Giả sử John được tăng hoặc giảm lương.
- Sự thay đổi lương và cấp bậc của John trong cột **Salary** hoặc **Grade** cần được phản ánh tới tất cả các dự án mà John đang làm việc.
- Vấn đề của việc phải cập nhật tất cả những nơi mà John xuất hiện được gọi là dị thường khi cập nhật.

# Chuẩn hóa (Normalization) 4-4

Bảng Department Employee Details được gọi là bảng chưa chuẩn hóa. Các hạn chế dần dần tới việc cần phải chuẩn hóa.

Chuẩn hóa là quá trình loại bỏ sự dư thừa và phụ thuộc không mong muốn.

Ban đầu, Codd (1972) trình bày 3 dạng chuẩn (1NF, 2NF and 3NF), tất cả dựa trên sự phụ thuộc giữa các thuộc tính của quan hệ.

Các dạng chuẩn thứ tư và thứ năm được dựa trên đa giá trị và ghép các phụ thuộc và đã được đề xuất sau đó.

# Dạng Chuẩn 1 (1NF) 1-2

➤ Để đạt được chuẩn một, cần thực hiện các bước sau:

1

- Tạo ra các bảng riêng biệt cho mỗi nhóm các dữ liệu liên quan.

2

- Các cột của bảng phải có giá trị nguyên tố (đơn trị).

3

- Tất cả các thuộc tính khóa phải được nhận diện.

➤ Xem bảng **Employee Project Details** dưới đây:

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

- Bảng trên có chứa các dữ liệu liên quan đến các dự án và các nhân viên.
- Cần tách bảng trên thành hai, một là bảng **Project Details** và một là bảng **Employee Details**
- Các cột **Project\_id** and **Project\_names** của bảng trên là các các cột đa trị



# Dạng Chuẩn 2 (2NF) 1-2

- Các bảng được gọi là chuẩn hai nếu:

Chúng thoả mãn các yêu cầu của dạng chuẩn một.

Không có phụ thuộc hàm bộ phận. (Tất cả các thuộc tính không khóa đều phụ thuộc hàm đầy đủ vào khóa)

Các bảng được thiết lập quan hệ thông qua các khóa ngoại.

- Phụ thuộc hàm bộ phận có nghĩa là một thuộc tính không khóa không nên phụ thuộc bộ phận vào một hoặc nhiều thuộc tính khóa .
- Bảng **Project Details** và **Employee Details** không có bất kỳ một phụ thuộc bộ phận nào.
- **Project\_name** chỉ phụ thuộc vào **Project\_id**, và **Emp\_name**, **Grade**, **Salary** chỉ phụ thuộc vào **Emp\_no**.
- Các bảng được thiết lập quan hệ thông qua các khóa ngoại.
- Bảng thứ ba, có tên **Employee Project Details**, được tạo nên bởi hai cột **Project\_id** và **Emp\_no**.



# Dạng Chuẩn 2 (2NF) 1-2

- Dữ liệu cần phải được phân chia trên các hàng khác nhau.
- Các bảng kết quả là **Project Details** và **Employee Details** như sau:

Project_id	Project_name
113	BLUE STAR
124	MAGNUM

**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

**Employee Details**

- Thuộc tính **Project\_id** là khóa chính của bảng **Project Details**.
- Thuộc tính **Emp\_no** là khóa chính của bảng **Employee Details**.
- Do đó, trong dạng chuẩn một, từ bảng ban đầu **Employee Project Details** sinh ra các bảng là **Project Details** và **Employee Details**.

# Dạng Chuẩn 2 (2NF) 2-2

- Khi chuyển sang dạng chuẩn 2, từ hai bảng **Project Details** và **Employee Details** sinh ra các bảng **Project Details**, **Employee Details** và bảng **Employee Project Details** như sau:

Project_id	Project_name
113	BLUE STAR
124	MAGNUM

**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

**Employee Details**

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

**Employee Project Details**

- Các thuộc tính, **Emp\_no** và **Project\_id**, của bảng **Employee Project Details** được kết hợp cùng nhau để thành lập một khóa chính (primary key).
- Các khóa chính như vậy được gọi là khóa phức hợp.

# Dạng chuẩn 3 (3NF) 1-3

## ➤ Để đạt dạng chuẩn 3:

Các bảng thỏa mãn yêu cầu dạng chuẩn hai

Không tồn tại phụ thuộc bắc cầu trong các bảng.

- Các bảng **Project Details**, **Employee Details**, và **Employee Project Details** ở dạng chuẩn hai.
- Nếu một thuộc tính có thể được xác định bởi một thuộc tính không khóa khác, nó được gọi là phụ thuộc bắc cầu.
- Như vậy, mỗi thuộc tính không khóa chỉ nên được xác định bởi thuộc tính khóa.
- Nếu một thuộc tính không khóa có thể được xác định bởi một thuộc tính không khóa khác, nó cần phải đặt vào một bảng khác.

## Dạng chuẩn 3 (3NF) 2-3

- Quan sát các bảng ta thấy, hai bảng **Project Details** và **Employee Project Details** không tồn tại bất kỳ phụ thuộc hàm bậc cao nào.
- Các thuộc tính không khóa phụ thuộc hoàn toàn vào các thuộc tính khóa chính.
- Thuộc tính **Project\_name** chỉ được xác định bởi **Project\_number**.
- Xem xét kỹ bảng **Employee Details**, chắc chắn thấy có sự không nhất quán.
- Thuộc tính **Salary** được xác định bởi thuộc tính **Grade** và không có thuộc tính khóa **Emp\_no**.
- Như vậy, đây là một phụ thuộc hàm bậc cao cần phải loại bỏ.
- Bảng **Employee Details** có thể chia thành hai bảng là **Employee Details** và **Grade Salary Details** như sau:

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

**Employee Details**

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

**Grade Salary Details**

# Dạng chuẩn 3 (3NF) 3-3

- Như vậy, kết thúc sau 3 giai đoạn chuẩn hóa, từ bảng **Employee Project Details** ban đầu đã sinh ra các bảng **Project Details**, **Employee Project Details**, **Employee Details**, and **Grade Salary Details** như sau:

Project_id	Project_name
113	BLUE STAR
124	MAGNUM

**Project Details**

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

**Employee Project Details**

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

**Employee Details**

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

**Grade Salary Details**

# Denormalization

Bằng việc chuẩn hóa một csdl, dữ liệu dư thừa sẽ được giảm thiểu.

Điều này dẫn đến giảm đòi hỏi không gian lưu trữ cho csdl và đảm bảo tính toàn vẹn.

Tuy nhiên, nó có một số hạn chế:

Thêm phức tạp khi thường xuyên phải các truy vấn kết nối(join) để ghép các dữ liệu từ trong nhiều bảng.

Việc ghép nối có thể có liên quan đến nhiều hơn ba bảng phụ thuộc vào nhu cầu thông tin.

- Nếu như thường xuyên phải sử dụng ghép nối (joins), hiệu suất (thao tác) với csdl sẽ trở lên rất tồi.
- Trong trường hợp cần để tăng hiệu suất của csdl, việc lưu trữ thêm một vài trường dư thừa cũng có thể bỏ qua được.
- Các cơ sở dữ liệu có dư thừa nhỏ như vậy để tăng hiệu suất được gọi là cơ sở dữ liệu không chuẩn và quá trình làm như vậy được gọi là phi chuẩn.

# Các toán tử quan hệ

- Mô hình quan hệ được dựa hoàn toàn trên cơ sở của đại số quan hệ.
- Đại số quan hệ(Relational Algebra) bao gồm tập hợp các toán tử thao tác trên các quan hệ.
- Mỗi toán tử cần có một hoặc hai quan hệ xem như là đầu vào và tạo ra kết quả đầu ra là một quan hệ mới.
- Hãy xem xét **Branch Reserve Details** được trình bày trong bảng dưới đây

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

**Branch Reserve Details**



# Phép chọn - SELECT

- Phép chọn SELECT được sử dụng để lấy dữ liệu thỏa mãn điều kiện đưa ra.
- Kí tự sigma ' $\sigma$ ' được sử dụng làm kí hiệu cho phép toán chọn.
- Phép toán chọn trên bảng **Branch Reserve Details** để hiển thị thông tin các chi nhánh ở London, kết quả nhận được trong bảng sau:

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

**Thông tin các chi nhánh ở London**

- Phép toán chọn trên bảng **Branch Reserve Details** để hiển thị thông tin các chi nhánh với reserve > 10 tỷ Euro, kết quả nhận được trong bảng sau:

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

**Thông tin các chi nhánh có reserves > 20 tỷ Euro**

# Phép chiếu - PROJECT

- Phép chiếu PROJECT được sử dụng để chiếu các chi tiết của một bảng quan hệ.
- Phép chiếu PROJECT chỉ hiển thị thông tin cần thiết loại bỏ một số cột.
- Phép chiếu PROJECT được kí hiệu bởi kí tự pi, ' $\pi$ '.
- Giả sử chỉ cần hiển thị Branch\_id và Reserve
- Thực hiện phép chiếu trên bảng **Branch Reserve Details** cho kết quả như sau:

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

**Bảng kết quả với Branch\_id và Reserve Amounts**

# Phép tính đề các - PRODUCT

- Phép tính đề các (PRODUCT), được kí hiệu bằng 'x', nó ghép mỗi dòng trong bảng thứ nhất với tất cả các dòng trong bảng thứ hai. Xét bảng dưới đây:

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

**Branch Loan Details**

- Phép tích đề các trên bảng **Branch Reserve Details** và **Branch Loan Details** cho kết quả như sau:

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

**Kết quả tích đề các của Branch Reserve Details và Branch Loan Details**

# Phép hợp – UNION

- Giả sử một văn phòng của ngân hàng với dữ liệu cho trong các bảng **Branch Reserve Details** và **Branch Loan Details**, muốn biết những chi nhánh có dự trữ (reserves) hoặc cho vay(loans) dưới 20 tỷ Euro.
- Bảng kết quả sẽ gồm có các chi nhánh(branche) có dự trữ (reserves) hoặc khoản vay(loans) hoặc cả hai dưới 20 tỷ Euro.
- Nó tương tự như hợp(union) của hai tập dữ liệu: tập thứ nhất là các chi nhánh có dự trữ(reserve) nhỏ hơn 20 tỷ Euro, tập thứ hai là các chi nhánh với khoản vay(loan) nhỏ hơn 20 tỷ Euro.
- Các chi nhánh có cả dự trữ và khoản vay dưới 20 tỷ sẽ chỉ hiển thị một lần.
- Phép toán hợp không chỉ thực hiện vậy, nó thu thập dữ liệu từ nhiều bảng khác nhau và biểu diễn dưới một phiên bản dữ liệu hoàn chỉnh duy nhất.
- Phép toán hợp được biểu diễn bằng kí hiệu 'U'.
- Hợp của hai bảng **Branch Reserve Details** và **Branch Loan Details** sẽ cho ra kết quả như sau:

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

# Phép giao - INTERSECT

- Giả sử giám đốc sau khi xem dữ liệu này muốn biết các chi nhánh có cả dự trữ và cho vay quá thấp.
- Câu trả lời là phép giao (intersect relational operation) .
- Phép Giao (INTERSECT) lấy ra những dữ liệu mà có ở trong tất cả các bảng mà toán tử được áp dụng.
- Nó được dựa trên phép giao nhau của lý thuyết tập hợp và được kí hiệu là ' $\cap$ '.
- Kết quả phép giao của các bảng **Branch Reserve Details** và **Branch Loan Details** sẽ liệt kê các chi nhánh có cả dự trữ và khoản vay dưới 20 tỷ Euro trong tài khoản của họ.
- Bảng kết quả được sinh ra như sau:

Branch	Branch_id
London	BS-01
London	BS-02

Các chi nhánh có cả dự trữ và khoản vay  
dưới 20 tỷ Euro

# Phép trừ - DIFFERENCE

- Nếu bây giờ giám đốc muốn danh sách các chi nhánh có dự trữ thấp nhưng không có cho vay(loan), thì giám đốc sẽ phải sử dụng phép toán (difference operation).
- Toán tử khác(DIFFERENCE) , kí hiệu là '-', cũng phát sinh ra dữ liệu từ nhiều bảng khác nhau, nhưng nó phát sinh ra dữ liệu có trong bảng này nhưng không có trong bảng khác.
- Do đó, các chi nhánh mà có dự trữ thấp nhưng không có các khoản vay sẽ được hiển thị.
- Dưới đây là bảng kết quả được sinh ra

Branch	Branch_id
Paris	BS-03

**Chi nhánh mà có dự trữ thấp nhưng không  
có các khoản vay**

# Phép ghép - JOIN

- Phép ghép nối(JOIN) là sự tác cải tiến của phép tích đề các (product).
- Nó cho phép lựa chọn được thực hiện trên tích đề các các bảng.
- Ví dụ, nếu cần các giá trị dự trữ(reserve) và khoản cho vay của các chi nhánh với các giá trị dự trữ và cho vay thấp, cần tích đề các hai bảng là **Branch Reserve Details** và **Branch Loan Details**.
- Khi tích đề các của hai bảng được sinh ra, chỉ những chi nhánh nào có cả dự trữ(reserve) và khoản vay(loan) dưới 20 tỷ Euro được liệt kê
- Bảng dưới đây là kết quả được sinh ra của phép JOIN:

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

Detailed List of Branches with Low Reserve and Loans



# Phép chia - DIVIDE

- Giả sử một nhà quản lý muốn xem tên chi nhánh và dự trữ của tất cả các chi nhánh đã có các khoản vay.
- Có thể thực hiện quá trình này dễ dàng bằng toán tử `DIVIDE`.
- Nhà quản lý cần thực hiện là chia bảng **Branch Reserve Details**, là bảng danh sách tên các chi nhánh, theo cột **Branch\_Id**
- Bảng dưới đây là kết quả được sinh ra.

Branch	Reserve (Billion €)
London	9.2
London	10

Bảng kết quả của phép chia

- Các thuộc tính (cột) của bảng chia luôn luôn là tập con của bảng bị chia.



# Summary

- Data modeling is the process of applying an appropriate data model to the data at hand.
- E-R model views the real world as a set of basic objects and relationships among them.
- Entity, attributes, entity set, relationships, and relationship sets form the five basic components of E-R model.
- Mapping cardinalities express the number of entities that an entity is associated with.
- The process of removing redundant data from the tables of a relational database is called normalization.
- Relational Algebra consists of a collection of operators that help retrieve data from the relational databases.
- SELECT, PRODUCT, UNION, and DIVIDE are some of the relational algebra operators.