Bài 16
Thiết kế và tạo CSDL

Module: Advanced Programming with PHP

Mục tiêu

- Trình bày được mục đích của việc thiết kế CSDL
- Mô tả được các bước để thiết kế cơ sở dữ liệu
- Thiết kế được cơ sở dữ liệu đơn giản
- Trình bày được các ràng buộc trong CSDL
- Trình bày được ý nghĩa của khoá chính
- Tạo được khoá chính trong một bảng
- Tạo được khoá ngoại trong một bảng
- Sử dụng được các kiểu dữ liệu cơ bản của MySQL
- Trình bày được các thao tác CRUD
- Thực hiện được các thao tác CRUD

Mục tiêu

- Trình bày được cú pháp câu lệnh INSERT
- Trình bày được cú pháp câu lệnh SELECT
- Trình bày được cú pháp câu lệnh UPDATE
- Trình bày được cú pháp câu lệnh DELETE
- Trình bày được cú pháp câu lệnh WHERE cơ bản

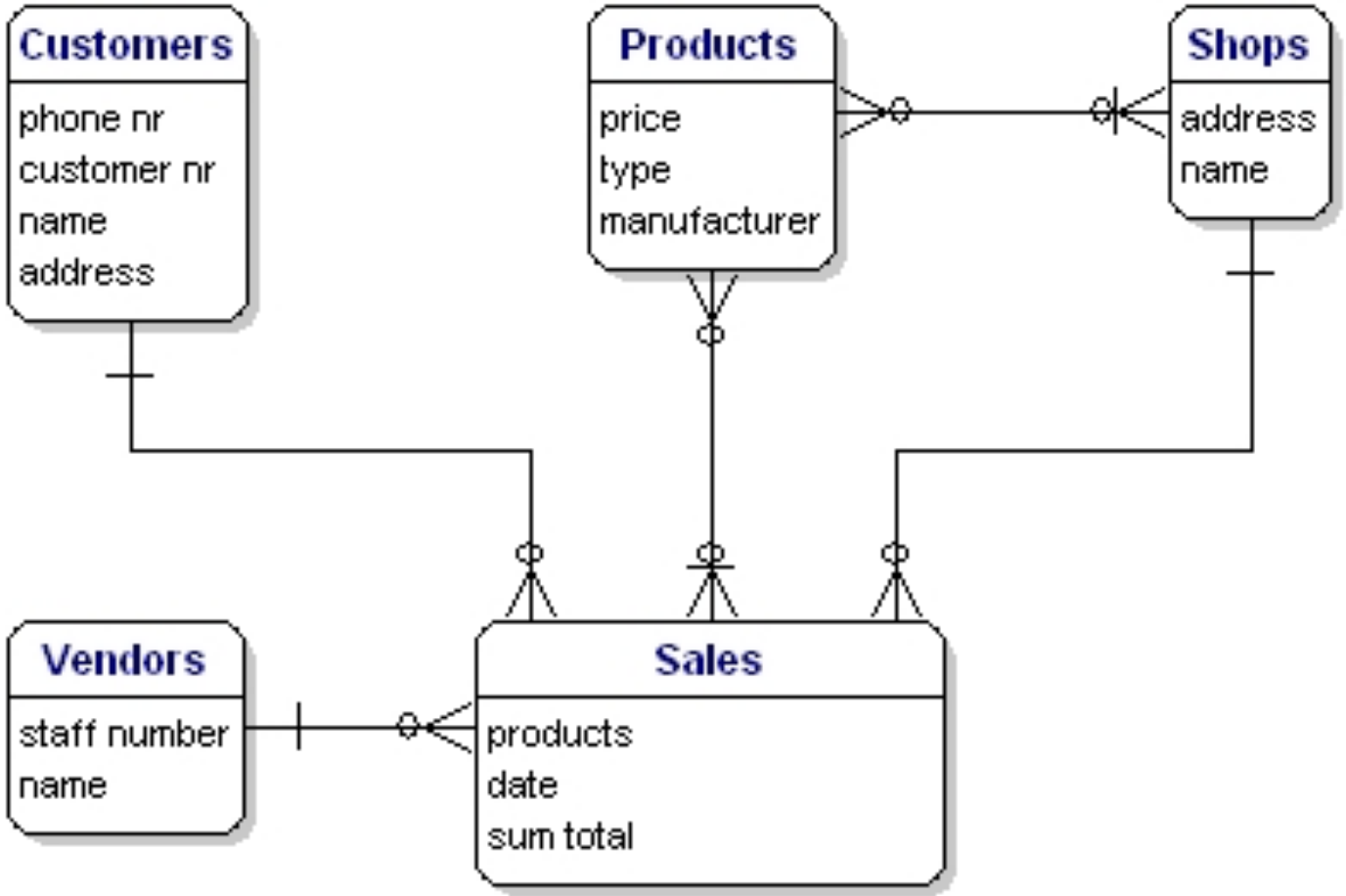
Thảo luận

Thiết kế CSDL

Phân tích và Thiết kế CSDL

- Phân tích và Thiết kế CSDL là các thao tác được thực hiện để tìm ra một mô hình CSDL trong một tình huống nhất định
- Kết quả của bước này là một bản thiết kế của CSDL
- Dựa vào bản thiết kế của CSDL, chúng ta có thể viết các câu lệnh để tạo ra CSDL đó
- Bản thiết kế của CSDL thường được biểu diễn dưới dạng một Lưu đồ Thực thể Quan hệ (ERD - Entity-Relationship Diagram)

ERD: Ví dụ


Customers
phone nr
customer nr
name
address
Products
price
type
manufacturer
Shops
address
name
Vendors
staff number
name
Sales
products
date
sum total

Các bước phân tích và thiết kế CSDL Quan hệ

1. Xác định mục đích của CSDL
2. Tìm hiểu và tổ chức các thông tin cần lưu trữ
3. Phân chia thông tin vào trong các bảng
4. Xác định các trường dữ liệu của từng bảng
5. Xác định khóa chính của các bảng
6. Xác định mối quan hệ giữa các bảng
7. Làm mịn thiết kế
8. Áp dụng các quy tắc chuẩn hóa

Ví dụ về hệ thống quản lý bán hàng

- Danh sách các cửa hàng
- Danh sách các sản phẩm
- Danh sách các nhà cung cấp
- Danh sachs các nhà sản xuất
- Danh sách khách hàng
- Danh sách hóa đơn

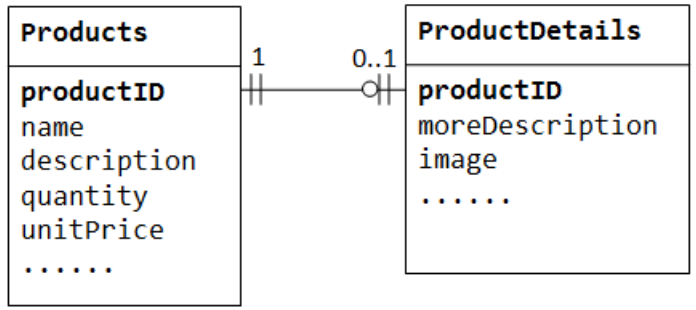
Tập hợp Dữ liệu, Tổ chức các bảng

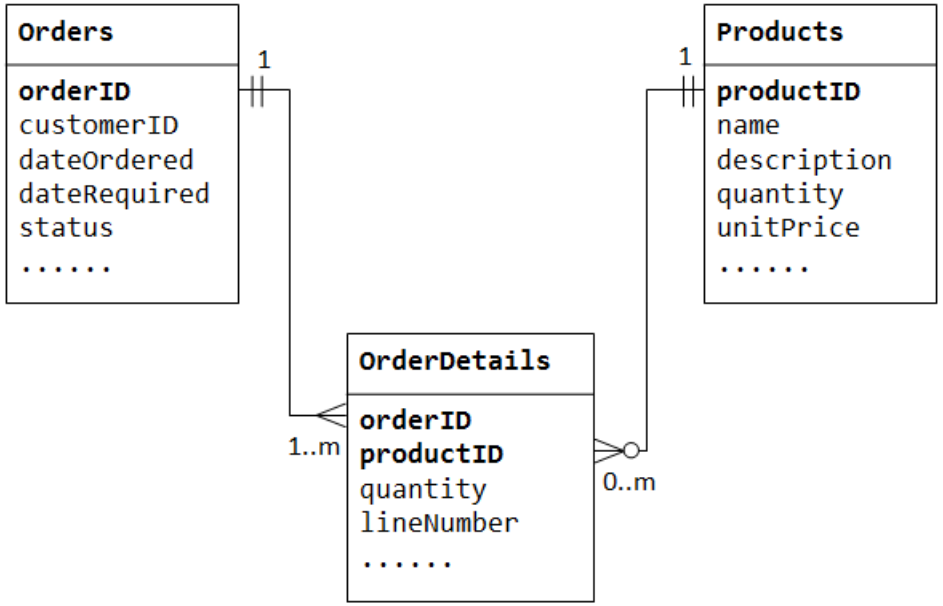
- Tìm hiểu và tổ chức các thông tin cần lưu trữ
- Phân chia thông tin vào trong các bảng
- Xác định các trường dữ liệu của từng bảng
- Xác định khóa chính của các bảng

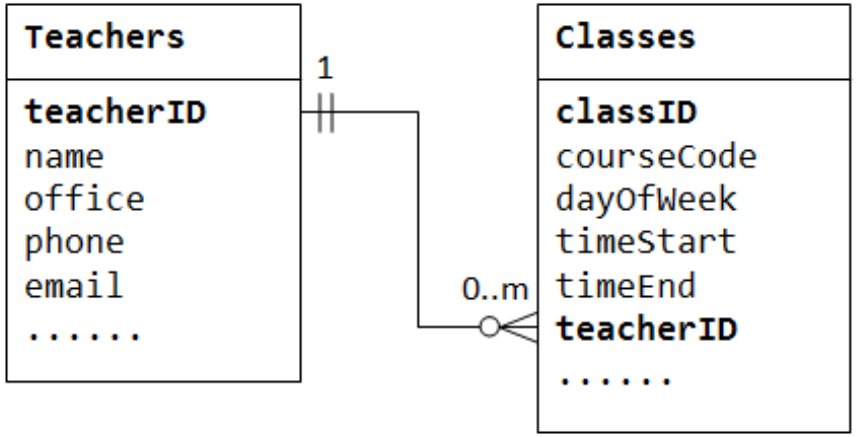
Tạo mối quan hệ giữa các bảng

Các loại quan hệ gồm:

- one-to-many(một ~ nhiều)
- many-to-many(nhiều~ nhiều)
- one-to-one(một~một)



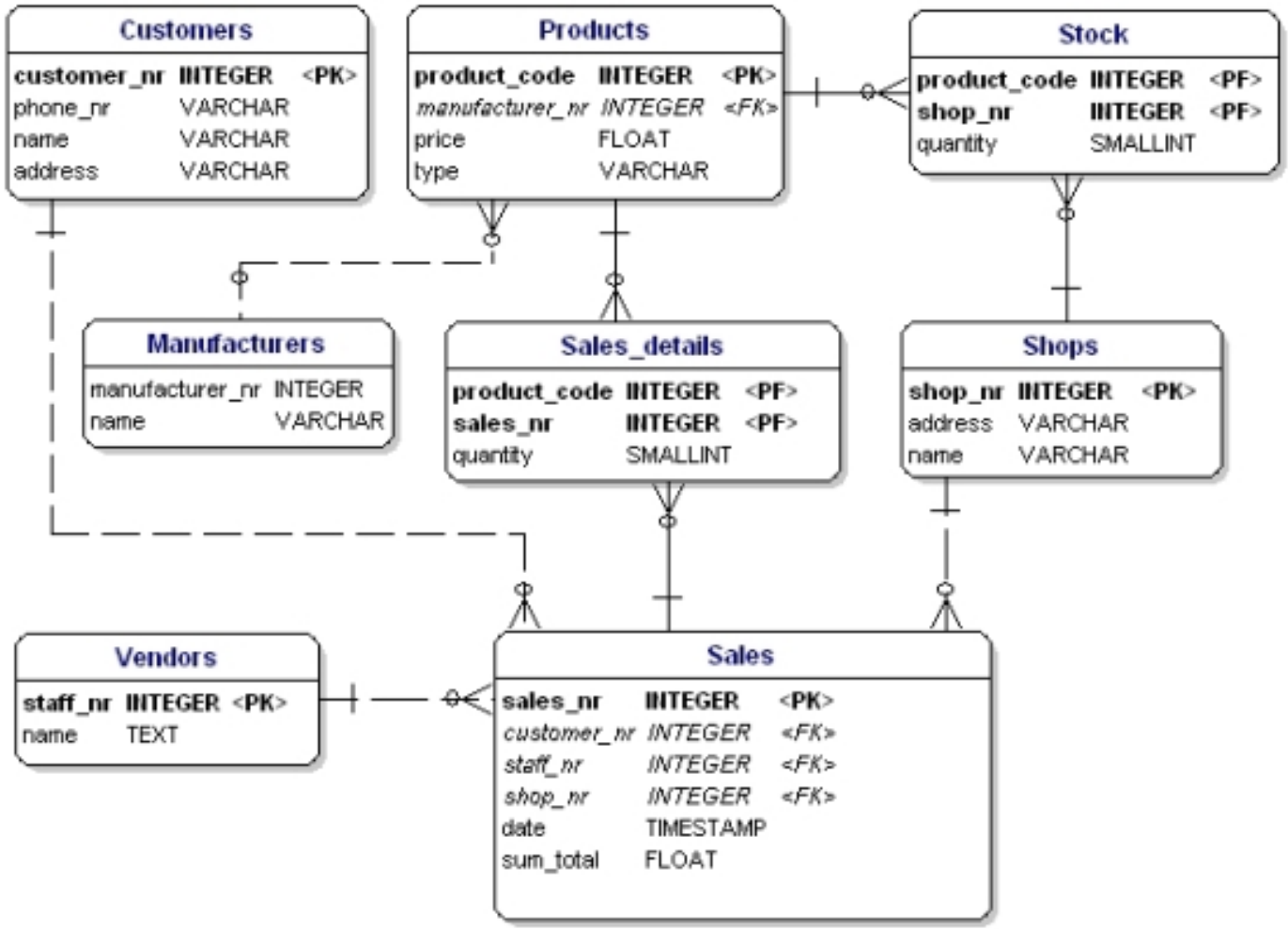




Tinh chỉnh và chuẩn hóa thiết kế

- Chuẩn hóa
  - Chuẩn thứ nhất(1NF)
  - Chuẩn thứ hai (2NF)
  - Chuẩn thứ ba (3NF)
- Quy tắc về tính toàn vẹn của Entity: Khóa chính không được phép NULL.

ERD Website bán hàng



Thảo luận

Tạo CSDL

Tạo CSDL

- Câu lệnh tạo CSDL:
  CREATE DATABASE

- Câu lệnh tạo bảng:
  CREATE TABLE

- Cấu trúc của một bảng cần tuân theo các ràng buộc

Constraint (ràng buộc)

- Constraint là các quy tắc (rule) được quy định cho bảng
- Sử dụng constraint, chúng ta có thể hạn chế những dữ liệu có thể đưa vào trong bảng
- Constraint giúp cho dữ liệu chính xác, tin cậy, toàn vẹn
- Dữ liệu đưa vào trong bảng cần tuân thủ các constraint
- Có thể quy định constraint cho bảng hoặc cột

Một số ràng buộc thông dụng

- NOT NULL: Không cho phép giá trị NULL
- UNIQUE: Mỗi giá trị là duy nhất
- PRIMARY KEY: Khóa chính (Không NULL và là UNIQUE)
- FOREIGN KEY: Khóa ngoại (tham chiếu sang bảng khác)
- CHECK: Kiểm tra dựa vào một điều kiện
- DEFAULT: Quy định giá trị mặc định cho trường (nếu không có giá trị nào được nhập vào)
- INDEX: Giúp tăng tốc độ truy vấn dữ liệu

NOT NULL

- Mặc định thì các trường trong bảng có thể chứa giá trị NULL
- Sử dụng từ khóa NOT NULL để bắt buộc các trường phải có giá trị khác NULL
- Ví dụ:

```
CREATE TABLE persons (
    id int NOT NULL,
    last_name varchar(255) NOT NULL,
    first_name varchar(255) NOT NULL,
    age int
);
```

UNIQUE

- Mặc định thì các trường trong bảng có thể chứa giá trị giống nhau
- Sử dụng từ khóa UNIQUE để bắt buộc các giá trị trong cột phải khác nhau
- Chẳng hạn: Số điện thoại, email...
- Ví dụ:

```
CREATE TABLE persons (
    id int NOT NULL UNIQUE,
    name varchar(255) NOT NULL,
    email varchar(100) UNIQUE,
    phone varchar(13) UNIQUE
);
```

CHECK

- Từ khóa CHECK được sử dụng để quy định điều kiện ràng buộc cho giá trị của một cột
- Ví dụ:

```
CREATE TABLE persons (
    id int NOT NULL,
    last_name varchar(255) NOT NULL,
    first_name varchar(255),
    age int,
    CHECK (age>=18)
);
```

DEFAULT

- Từ khóa DEFAULT được sử dụng để quy định giá trị mặc định cho một trường
- Giá trị mặc định sẽ được sử dụng nếu không NULL được nhập vào
- Ví dụ:

```
CREATE TABLE persons (
    id int NOT NULL,
    last_name varchar(255) NOT NULL,
    first_name varchar(255),
    age int DEFAULT 0,
    city varchar(255) DEFAULT 'Sandnes'
);
```

Primary Key (Khóa chính)

- Khóa chính là một cột (hoặc nhiều cột) được sử dụng để xác định một bản ghi duy nhất trong bảng
- Khóa chính là UNIQUE và NOT NULL
- Mỗi bảng chỉ có thể có 1 khóa chính
- Từ khóa PRIMARY KEY được sử dụng để định nghĩa khóa chính

Primary Key: Ví dụ

```
CREATE TABLE users(
  id INT AUTO_INCREMENT PRIMARY KEY,
  username VARCHAR(40),
  password VARCHAR(255),
  email VARCHAR(255)
);
```

- Hoặc

```
CREATE TABLE roles(
  id INT AUTO_INCREMENT,
  name VARCHAR(50),
  PRIMARY KEY(id)
);
```

Tạo khóa chính phức hợp

- Khóa chính phức hợp (composite) là khóa được tạo nên từ 2 hoặc nhiều cột:
- Ví dụ:

```
CREATE TABLE user_roles(
   user_id INT NOT NULL,
   role_id INT NOT NULL,
   PRIMARY KEY(user_id,role_id),
   ...
);
```

Tạo khóa chính sau khi tạo bảng

- Nếu một bảng đã tồn tại và chưa có khóa chính, chúng ta có thể bổ sung khóa chính thông qua câu lệnh ALTER TABLE.
- Ví dụ:

```
ALTER TABLE users
    ADD PRIMARY KEY(id);
```

Khóa ngoại (Foreign Key)

- Khóa ngoại là cơ chế để tạo liên kết giữa 2 bảng trong cùng CSDL
- Khóa ngoại được đặt trên một cột của bảng này và tham chiếu đến khóa chính của một bảng khác
- Kiểu dữ liệu của khóa chính và khóa ngoại phải giống nhau

Khóa ngoại: Ví dụ

- CSDL bán hàng: Bảng **customers** và bảng **orders** có liên kết với nhau

```
CREATE TABLE customers(
  id INT AUTO_INCREMENT PRIMARY KEY,
  name VARCHAR(40),
  address VARCHAR(255),
  email VARCHAR(255)
);
```

- Cột **customer_id** trong bảng **orders** là khóa ngoại, tham chiếu đến cột **id** của bảng **customers**

```
CREATE TABLE orders(
  id INT AUTO_INCREMENT,
  staff VARCHAR(50),
  PRIMARY KEY(id),
  customer_id INT, FOREIGN KEY (customer_id) REFERENCES customers(id)
);
```

Các kiểu dữ liệu thông dụng

- CHAR
- VARCHAR
- TEXT
- LONGTEXT
- INT
- BIGINT
- FLOAT
- DOUBLE
- DATE
- DATETIME
- TIMESTAMP
- TIME

Thảo luận

CRUD

CRUD

- CRUD (Create-Read-Update-Delete) là tên gọi ngắn gọn để chỉ đến 4 thao tác cơ bản của một hệ thống có lưu trữ dữ liệu:
  - Create: Tạo dữ liệu
  - Read (hoặc Retrieve): Đọc dữ liệu
  - Update: Cập nhật dữ liệu
  - Delete (hoặc Destroy): Xóa dữ liệu
- Ví dụ: Trang web tin tức, trang web bán hàng, trang web chăm sóc khách hàng, trang web quản lý nhân viên...

Ứng dụng quản lý cơ bản

- Một ứng dụng quản lý (chẳng hạn như: quản lý sản phẩm) cơ bản sẽ có những chức năng sau:
    1. Hiển thị danh sách các thực thể
    2. Hiển thị thông tin chi tiết một thực thể
    3. Thêm một thực thể mới
    4. Cập nhật thông tin của một thực thể
    5. Xóa một thực thể

[1] Hiển thị danh sách các thực thể

GET: http://example.com/entities

Bấm vào đây sẽ đi đến trang Thêm thực thể mới

Thêm

Bấm vào đây sẽ đi đến trang Chỉnh sửa thực thể

Bấm vào đây sẽ đi đến trang Xóa thực thể

Hiển thị từ 100-120 trong tổng số 1000 thực thể

	Thuộc tính 1	Thuộc tính 2	Thuộc tính 3	Sửa	
1	....	....	....	sửa	xóa
2	....	....	.....	sửa	xóa
3	....	....	....	sửa	xóa
4	....	....	....	sửa	xóa
5	....	...	....	sửa	xóa

Bấm vào đây sẽ đi đến trang chi tiết thực thể

<< < > >>

Bấm vào đây sẽ hiển thị các phần dữ liệu tiếp theo

[2] Hiển thị chi tiết thực thể

GET: http://example.com/entities/1 ← Đây là id của thực thể đang xem

Thông tin chi tiết của thực thể

Thuộc tính 1: .....
Thuộc tính 2: ....
Thuộc tính 3: ....
.....
Thuộc tính n: ....

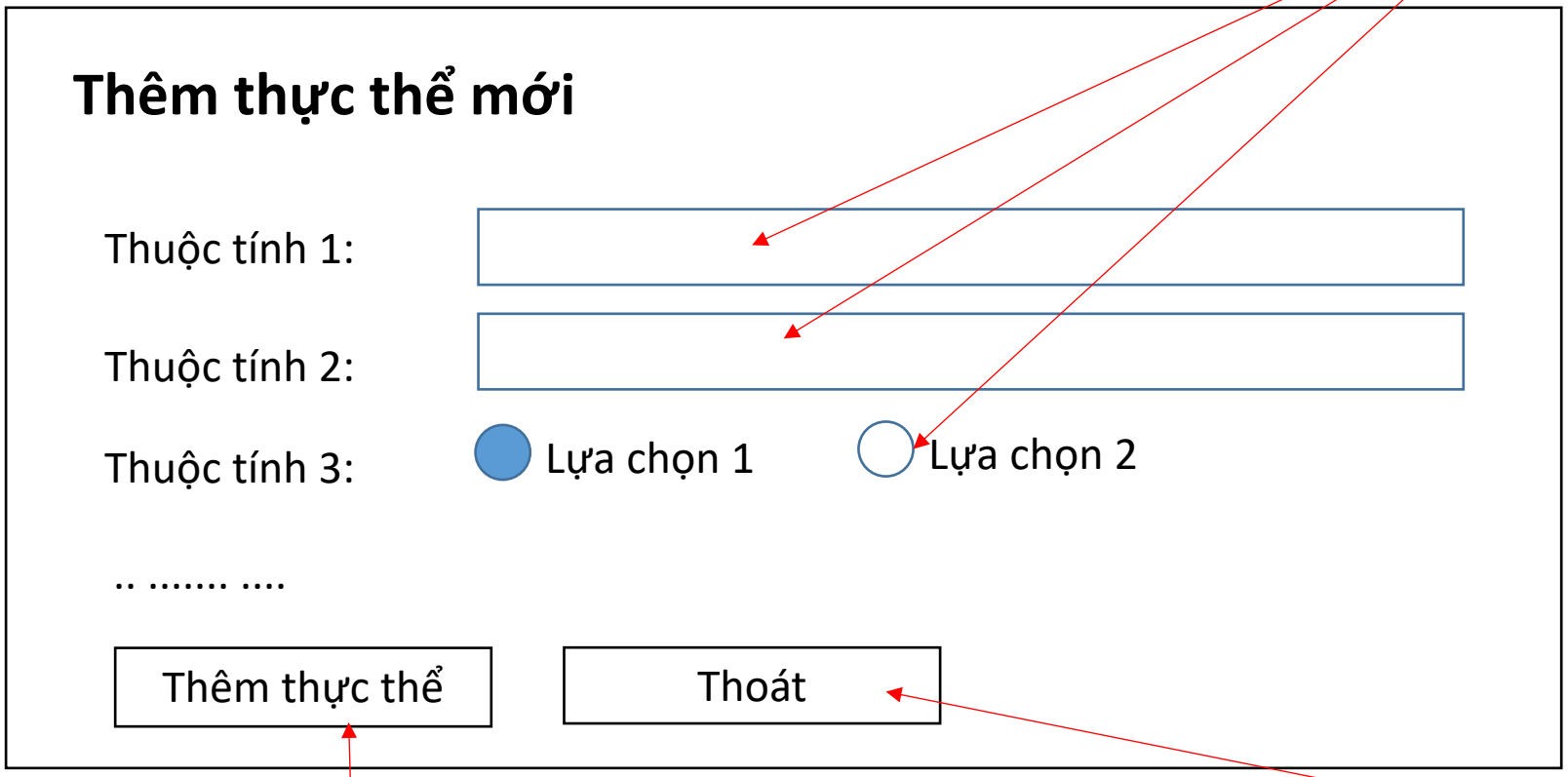
<<Quay lại trang danh sách thực thể>>

Bấm vào đây sẽ đi đến trang Danh sách các thực thể

[3] Thêm thực thể mới

GET: http://example.com/entities/new

Nhập giá trị của các thuộc tính của thực thể mới



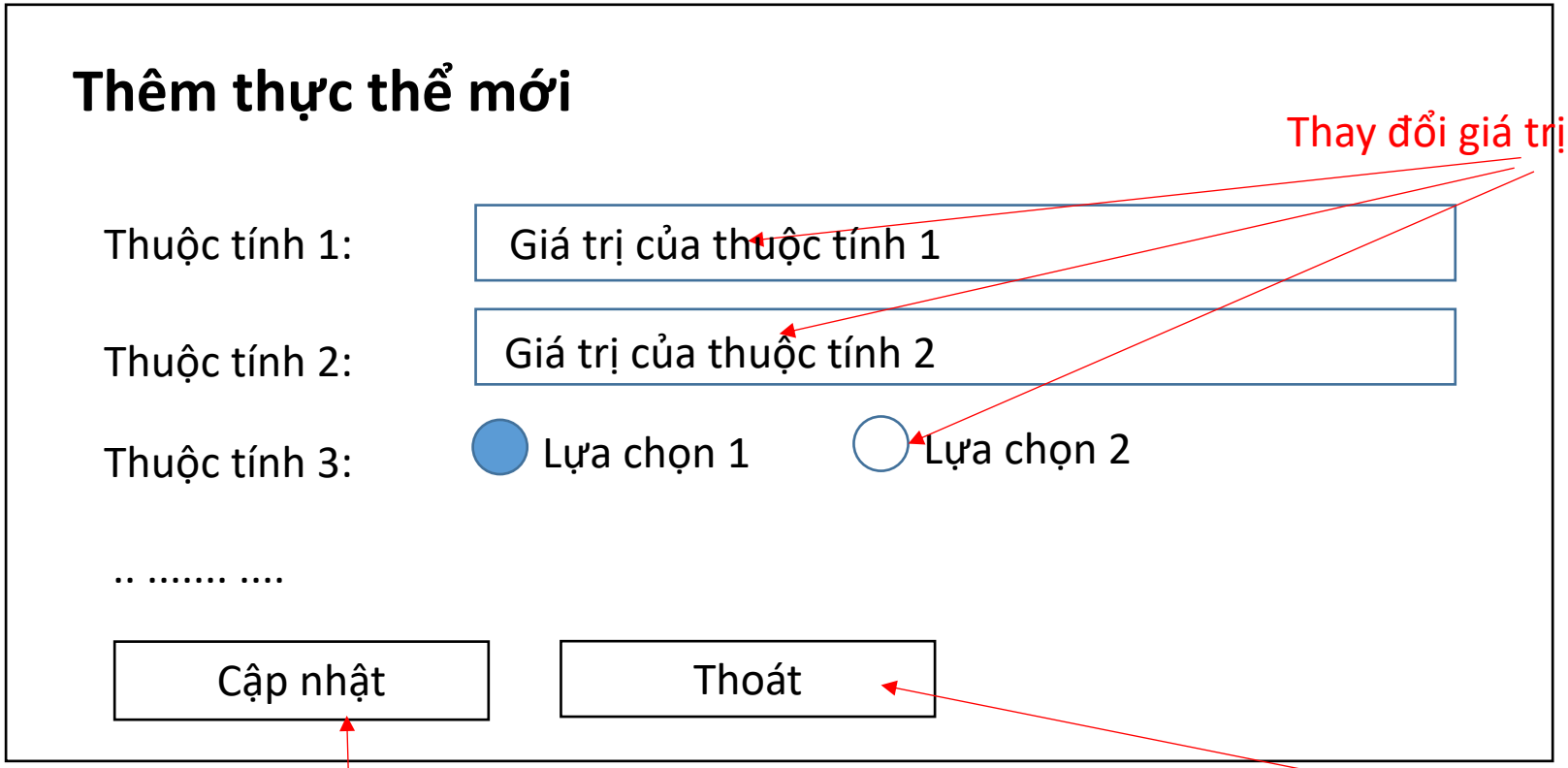
Bấm vào đây sẽ tạo thực thể mới [**POST**]

Bấm vào đây sẽ quay về trang hiển thị danh sách các thực thể

[4] Cập nhật thông tin thực thể

GET: http://example.com/entities/1/edit

Đây là id của thực thể đang chỉnh sửa

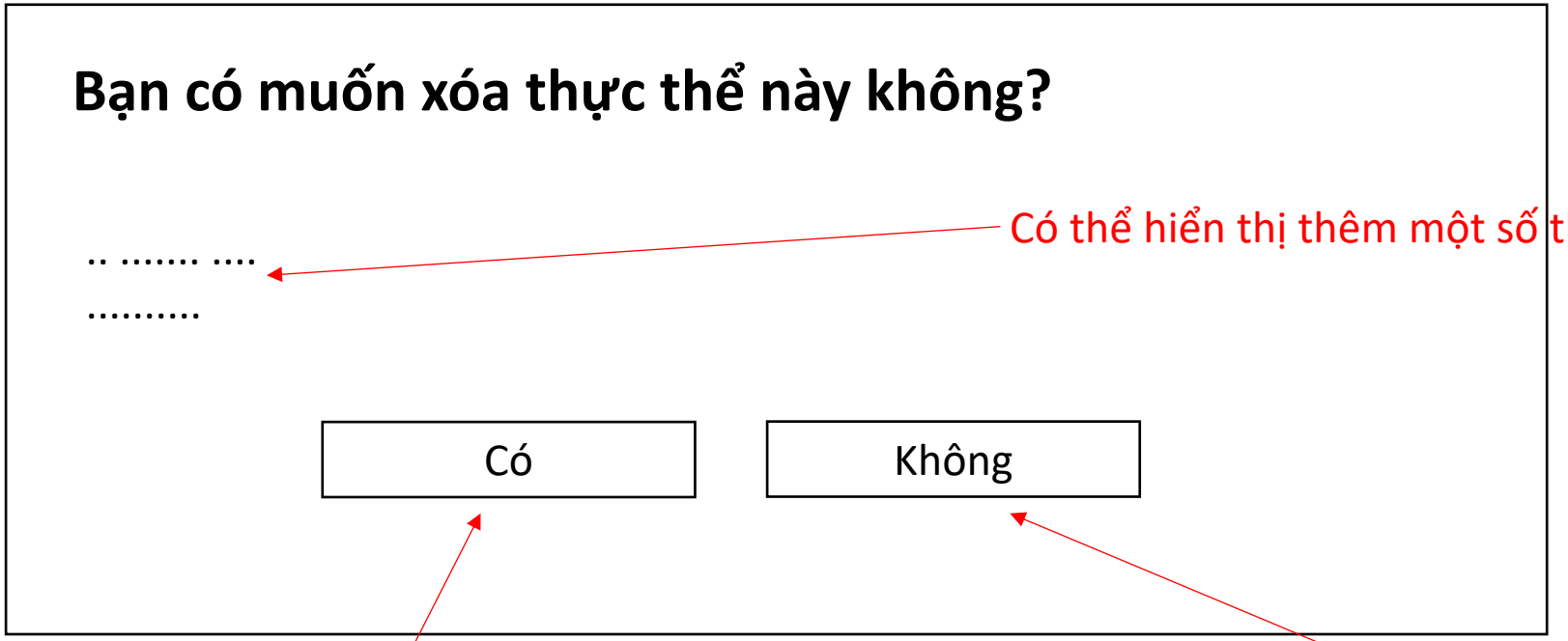


Bấm vào đây sẽ cập nhật thông tin thực thể [**POST**]

Bấm vào đây sẽ quay về trang hiển thị danh sách các thực thể

[5] Xóa một thực thể

GET: http://example.com/entities/1/delete ← Đây là id của thực thể đang muốn xóa



Có thể hiển thị thêm một số thông tin của thực thể

Bấm vào đây sẽ xóa thực thể [**POST**]

Bấm vào đây sẽ quay về trang hiển thị danh sách các thực thể

Demo: Ứng dụng quản lý cơ bản

- http://demo.codegym.vn/lamp/eshop-dashboard

Thảo luận

Các câu lệnh CRUD trong SQL

Các câu lệnh CRUD trong SQL

- Create: **INSERT INTO**

```
INSERT INTO customers (name, city, country) VALUES ('Cardinal', 'Stavanger', 'Norway');
```

- Read: **SELECT/FROM**

```
SELECT name, city FROM customers;
```

- Update: **UPDATE/SET**

```
UPDATE customers SET contact_name = 'Alfred Schmidt', city= 'Frankfurt‘ WHERE id = 1;
```

- Delete: **DELETE FROM**

```
DELETE FROM customers WHERE name='Alfreds Futterkiste';
```

Câu lệnh INSERT/INTO

- Cú pháp:

```
INSERT INTO table_name (column1, column2, column3, ...)
VALUES (value1, value2, value3, ...);
```

  - **Hoặc** (chèn dữ liệu cho tất cả các cột và đúng thứ tự cột):

```
INSERT INTO table_name
VALUES (value1, value2, value3, ...);
```

- Ví dụ:

```
INSERT INTO customers (name, contact_name, address, city, postal_code, country)
VALUES ('Cardinal', 'Tom B. Erichsen', 'Skagen 21', 'Stavanger', '4006', 'Norway');
```

Câu lệnh SELECT/FROM

- Cú pháp:

```
SELECT column1, column2, ...
FROM table_name;
```

  - Hoặc (đọc tất cả các cột có trong bảng):

```
SELECT * FROM table_name;
```

- Ví dụ:

```
SELECT name, City FROM customers;
```

  - Hoặc:

```
SELECT * FROM customers;
```

Câu lệnh UPDATE/SET

- Cú pháp:

```
UPDATE table_name
SET column1 = value1, column2 = value2, ...
WHERE condition;
```

- Ví dụ:

```
UPDATE customers
SET contact_name = 'Alfred Schmidt', city= 'Frankfurt'
WHERE id = 1;
```

- Câu lệnh **WHERE** là không bắt buộc

Câu lệnh DELETE

- Cú pháp:

```
DELETE FROM table_name
WHERE condition;
```

- Ví dụ:

```
DELETE FROM customers
WHERE name='Alfreds Futterkiste';
```

- Câu lệnh **WHERE** là không bắt buộc

Câu lệnh WHERE

- Cú pháp:

```
SELECT column1, column2, ...
FROM table_name
WHERE condition;
```

- Ví dụ:

```
SELECT * FROM Customers
WHERE Country='Mexico';
```

Các toán tử trong câu lệnh WHERE

Toán tử	Mô tả
=	So sánh bằng
<>	Khác nhau
>	Lớn hơn
<	Nhỏ hơn
>=	Lớn hơn hoặc bằng
<=	Nhỏ hơn hoặc bằng
BETWEEN	Nằm trong khoảng (bao gồm cả 2 giá trị biên)
LIKE	So sánh theo mẫu (pattern)
IN	So sánh theo một danh sách các giá trị

Kết nối đến CSDL

- Ứng dụng Web cần kết nối đến CSDL để thực hiện các thao tác
- Ứng dụng Web và CSDL có thể được cài đặt trên cùng Server hoặc khác server
- Để kết nối CSDL, cần biết các thông tin:
  - Địa chỉ của Database Server, chẳng hạn: **127.0.0.1**
  - Tên người dung, chẳng hạn: **root**
  - Mật khẩu, chẳng hạn: **secret**

Tóm tắt bài học

- Constraint là các quy tắc (rule) được quy định cho bảng
- Dữ liệu đưa vào trong bảng cần tuân thủ các constraint
- PRIMARY KEY: Khóa chính (Không NULL và là UNIQUE)
- FOREIGN KEY: Khóa ngoại (tham chiếu sang bảng khác)
- Phân tích và Thiết kế CSDL là các thao tác được thực hiện để tìm ra một mô hình CSDL trong một tình huống nhất định
- Kết quả của bước này là một bản thiết kế của CSDL

Hướng dẫn

- Hướng dẫn làm bài thực hành và bài tập
- Chuẩn bị bài tiếp: ***Thao tác với CSDL***