CHƯƠNG 6: TỔNG QUAN VỀ MySQL

CHƯƠNG 6: TỔNG QUAN VỀ MySQL

6.1 GIỚI THIỆU VỀ MySQL
6.2 TẠO CƠ SỞ DỮ LIỆU VÀ NGƯỜI DÙNG
6.3 CÁC KIỂU DỮ LIỆU TRONG MySQL
6.4 CÁC CÂU LỆNH SQL

6.1. GIỚI THIỆU MySQL

- MySQL là:
  - Một hệ quản trị CSDL
  - Một hệ quản trị CSDL quan hệ
  - Được phát triển, phân phối và hỗ trợ bởi MySQL AB.
- Để làm việc với MySQL cần đăng ký kết nối, tạo CSDL, quản lý người dùng, phân quyền sử dụng, thiết kế đối tượng Table của CSDL và xử lý dữ liệu.
- Để quản lý và thao tác trên CSDL ta có thể sử dụng giao diện đồ họa hoặc dạng Command line.

6.1. GIỚI THIỆU MySQL

- MySQL cũng giống như các hệ quản trị CSDL: Access, SQL Server, PostgreSQL, Oracle, …
- Phần mềm mã nguồn mở do đó có thể tải miễn phí từ trang chủ.
- Nó có nhiều phiên bản cho các hệ điều hành khác nhau: phiên bản Win32 cho các hệ điều hành dòng Windows, Linux, Mac OS X, Unix, Solaris…

6.1. GIỚI THIỆU MySQL

- MySQL là cơ sở dữ liệu tốc độ cao, ổn định và dễ sử dụng, có tính khả chuyển, hoạt động trên nhiều hệ điều hành cung cấp một hệ thống lớn các hàm tiện ích rất mạnh.
- Với tốc độ và tính bảo mật cao, MySQL rất thích hợp cho các ứng dụng có truy cập CSDL trên internet.
- MySQL server hoạt động trong các hệ thống nhúng hoặc client/server.

6.2. TẠO CSDL VÀ NGƯỜI DÙNG

- Kết nối và tạo CSDL
- Quản lý người dùng
- Cấp quyền cho người dùng
- Xóa quyền của người dùng

Kết nối và tạo CSDL

- Để kết nối cơ sở dữ liệu ta có thể thực hiện theo hai cách:
  - Kết nối và tạo CSDL bằng Command line
  - Kết nối và tạo CSDL bằng giao diện đồ họa: MySQL Administrator hoặc phpmyadmin

Quản lý người dùng

- Để đăng nhập vào MySQL ta có thể sử dụng user là root và pass là rỗng
- Ngoài tài khoản này ta có thể tạo thêm các tài khoản cho người dùng với các users và pass khác nhau.

Cấp quyền cho người dùng

- Với quyền root ta có thể thực hiện mọi thao tác trên CSDL: select, update, insert, delete, …
- Tuy nhiên, khi tạo quyền người dùng ta cũng có thể hạn chế bớt một số quyền nhất định nào đó

Xóa quyền hoặc tài khoản người dùng

- Sau khi cấp quyền cho người dùng ta có thể thêm hoặc loại bỏ một số quyền nào đó.
- Với việc truy cập vào tài khoản root ta có thể xóa các tài khoản người dùng đã được tạo ra.

6.3. CÁC KIỂU DỮ LIỆU TRONG MySQL

- Dữ liệu kiểu numeric
- Dữ liệu kiểu date and time
- Dữ liệu kiểu string

Dữ liệu kiểu numeric

- Dữ liệu kiểu numeric gồm các kiểu sau:
  - Tinyint: Lưu các số nguyên không dấu (unsigned) từ 0-255 hoặc các số nguyên có dấu từ -128 -> 127
  - Mediumint: Lưu các số nguyên từ 0 đến 16.777.215 hoặc từ -8.388.608 đến 8.388.607
  - Int: Lưu các số nguyên từ 0 đến 4.294.967.295 hoặc từ -2.147.483.648 đến 2.147.483.647
  - Bigint
  - Float
  - Doube
  - Decimal/real

Dữ liệu kiểu date and time

- Dữ liệu kiểu date and time gồm các kiểu sau:
  - Date: Lưu trữ ngày dạng yyy-mm-dd. Cho phép giá trị từ 1000-01-01 đến 9999-12-31
  - Datetime: Lưu trữ dạng yyyy-mm-dd hh:mm:ss
  - Timestamp: Tự động ghi nhận thời gian thay đổi gần nhất. Tùy thuộc vào độ rộng của cột nằm trong khoảng từ 2 đến 14 (yy đến yyyy-mm-dd hh:mm:ss)
  - Time: Dùng để lưu trữ giờ định dạng hh:mm:ss.
  - Year: Dùng để lưu năm bắt đầu từ 1970

Dữ liệu kiểu string

- Dữ liệu kiểu string gồm các kiểu sau:
  - Char: Chiều dài tối đa 255 ký tự, đây là kiểu có chiều dài cố định.
  - Varchar: Cũng tương tự kiểu char có chiều dài tối đa 255 ký tự, song có điểm khác là có chiều dài thay đổi, các giá trị sẽ không bị nối thêm ký tự trắng.
  - Tinytext: là kiểu ký tự văn bản nhị phân. Có chiều dài tối đa 255.
  - Text: có chiều dài 65.535 ký tự. Các chỉ mục có thể được tạo trên 255 ký tự đầu của cột text
  - Mediumtext: có chiều dài 16.777.215
  - Longtext: có chiều dài >4 tỉ ký tự
  - Enum, Set, …

6.4. CÁC CÂU LỆNH SQL

- **Ngôn ngữ SQL**
- **Các câu lệnh SQL cơ bản**

Ngôn ngữ SQL

- SQL là ngôn ngữ dùng để truy vấn CSDL
- Được chia làm 4 loại:
  - DDL (Data Definition Language)
  - DML (Data Manipulationn Language)
  - DCL (Data Control Language)
  - TCL (Transaction Control Language)
- Các câu lệnh SQL thông dụng:
  - Câu lệnh Select
  - Câu lệnh Insert
  - Câu lệnh Update
  - Câu lệnh Delete
  - Câu lệnh Join

Câu lệnh SELECT

- Dùng để truy vấn dữ liệu từ một hay nhiều bảng khác nhau và trả về kết quả là một tập mẫu tin thỏa mãn điều kiện nào đó
- Cú pháp:

SELECT <Danh sách các cột>
[FROM <danh sách các bảng>]
[WHERE <các điều kiện ràng buộc>]
[GROUP BY <tên cột/ biểu thức trong SELECT>]
[HAVING <đk bắt buộc của GROUP BY>]
[ORDER BY <danh sách cột>]
[LIMIT FromNumber | ToNumber]

Câu lệnh SELECT

- Trong đó, danh sách các cột: Tên các cột, biểu thức kết hợp giữa các cột của bảng
- Trường hợp truy vấn tất cả các cột của bảng ta sử dụng toán tử * thay vì chỉ ra danh sách tất cả các cột
- Trường hợp, có các cột cùng tên ở các bảng khác nhau thì ta cần chỉ ra tên bảng đi trước theo cú pháp:

  Tên_bảng.Tên_cột

Câu lệnh SELECT

- Câu lệnh SELECT với mệnh đề FROM: dùng để truy vấn dữ liệu từ các cột hoặc biểu thức cho cột đó từ bảng được chỉ ra sau mệnh đề FROM

Ví dụ:

```
SELECT * FROM Sinhvien;
SLECT Masv, HoTen FROM Sinhvien;
SELECT * FROM Sinhvien LIMIT 0, 10;
```

Câu lệnh SELECT

- Câu lệnh SELECT với mệnh đề WHERE: dùng để truy vấn dữ liệu từ các cột hoặc biểu thức cho cột đó từ bảng được chỉ ra sau mệnh đề FROM và thỏa mãn điều kiện nào đó được chỉ ra sau mệnh đề WHERE

Ví dụ:

SELECT * FROM Sinhvien WHERE conditions;

SLECT Masv, HoTen FROM Sinhvien WHERE conditions;

Lưu ý: các phép toán được sử dụng để thiết lập các điều kiện sau WHERE là các phép toán Quan hệ và Logic

Câu lệnh SELECT

- Các phép toán quan hệ: >, >=, <, <=, =, !=, <>
- Các phép toán Logic: and, or, not, not in, between, like, not like, in

Ví dụ:

SELECT * FROM Sinhvien WHERE Tongdiem > 2.0;

SELECT * FROM Sinhvien WHERE Hoten like ‘%Hoa’;

Câu lệnh SELECT

- Câu lệnh SELECT với mệnh đề ORDER BY:
  - Dùng để truy vấn dữ liệu và kết quả trả về được sắp xếp tăng dần (ASC) hoặc giảm dần (DESC) trên cột nào đó.
  - Trường hợp sắp xếp theo nhiều cột thì các cột được phân cách nhau bởi dấu phẩy (,)

Ví dụ:

```
SELECT * FROM Sinhvien ORDER BY Tongdiem DESC;
SELECT * FROM Sinhvien ORDER BY Hoten ASC;
```

Câu lệnh SELECT

- Câu lệnh SELECT với mệnh đề GROUP BY: dùng để truy vấn dữ liệu và kết quả trả về được nhóm lại theo một cột nào đó.

Ví dụ:

SELECT Tongdiem, count(Tongdiem) FROM Sinhvien GROUP BY Tongdiem ORDER BY Tongdiem;

Câu lệnh SELECT

- Câu lệnh SELECT với mệnh đề AS: sử dụng khi cần phải thay đổi tên cột nào đó trong câu truy vấn.

Ví dụ:

SELECT Tongdiem, count(Tongdiem) AS Sosv FROM Sinhvien GROUP BY Tongdiem ORDER BY Tongdiem;

Câu lệnh SELECT

- Câu lệnh SELECT với mệnh đề LIMIT N, M: dùng để giới hạn số mẫu tin cần truy vấn từ vị trí thứ N đến vị trí thứ M.

Ví dụ:

```
SELECT * FROM Sinhvien LIMIT 0, 10;
SELECT * FROM Sinhvien ORDER BY Tongdiem DESC LIMIT 0, 10;
```

Câu lệnh SELECT

- Câu lệnh SELECT với mệnh đề DISTINCT: dùng để truy vấn dữ liệu và kết quả trả về được nhóm lại theo một cột nào đó.

Ví dụ:

SELECT Tongdiem, count(Tongdiem) FROM Sinhvien GROUP BY Tongdiem ORDER BY Tongdiem;

Câu lệnh INSERT

- Được sử dụng khi cần thêm mẫu tin vào bảng trong CSDL MySQL.
- Khi thêm dữ liệu, cần chú ý đến kiểu dữ liệu của các cột mình cần thêm dữ liệu.
- Cần quan tâm đến quyền của User đăng nhập có được phép Insert hay không
- Khi Insert dữ liệu vào bảng có 3 trường hợp:
  - Insert từ giá trị cụ thể
  - Lấy giá trị từ một hoặc nhiều bảng khác
  - Bao gồm cả hai trường hợp

Câu lệnh INSERT

Ví dụ:

- Insert vào bảng từ giá trị cụ thể.

  INSERT INTO Sinhvien (Masv, Hoten, Tongdiem) VALUES (‘0073’, ‘Lê Anh Ngọc’, 2.37);

- Insert vào bảng từ giá trị của bảng khác

  INSERT INTO Sinhvien (Masv, Hoten, Tongdiem) SELECT Mahs, Hoten, Tongdiem FROM Hocsinh;

- Insert vào bảng từ giá trị cụ thể, bảng khác:

  INSERT INTO <bảng 1> [<danh sách các cột>] SELECT [danh sách các cột], danh sách giá trị FROM <bảng 2> WHERE <conditions> …

Câu lệnh UPDATE

- Dùng để cập nhật lại dữ liệu đã tồn tại trong bảng.
- Nếu cập nhật giá trị cụ thể:

UPDATE <tên bảng> SET <cột> = <giá trị>, [<cột> = <giá trị>] [WHERE <conditions>]

- Cập nhật giá trị từ bảng khác:

UPDATE <tên bảng> SET <tên cột>=<SELECT … FROM tên bảng …> …

- UPDATE có thể ảnh hưởng đến nhiều bảng nhưng cập nhật giá trị chỉ có hiệu lực trên bảng đó.

Câu lệnh UPDATE

Ví dụ:

- UPDATE Sinhvien SET Hoten=‘Le Thi B’ WHERE Masv = ‘003’;
- UPDATE Sinhvien SET Hoten= (SELECT Hoten FROM Hocsinh WHERE Mahs = Sinhvien.Masv)

Câu lệnh DELETE

- Dùng để xóa mẫu tin trong bảng được chỉ ra bởi tên bảng và mệnh đề WHERE (nếu có) nhằm xác định mẫu tin cần xóa theo một điều kiện nào đó.
- Cú pháp:

DELETE FROM <tên bảng> WHERE <conditions>

- Conditions: có thể là phép toán giữa các cột và giá trị hoặc giá trị là kết quả trả về của một câu lệnh SELECT khác
- Lưu ý: không có khái niệm xóa giá trị trong một cột, vì xóa giá trị một cột đồng nghĩa với cập nhật cột đó bằng giá trị rỗng

Câu lệnh DELETE

- **Ví dụ:**

DELETE FROM Sinhvien WHERE Masv = ‘001’;

- **Lưu ý:** trong trường hợp có ràng buộc về quan hệ của dữ liệu, thì việc xóa mẫu tin cần được thực hiện ở bảng con trước bảng cha.

Câu lệnh JOIN

- Dùng để kết hợp dữ liệu trên hai hay nhiều bảng lại với nhau
- Cần xác định cột nào trong bảng này có quan hệ với cột nào trong bảng kia
- Các dạng của JOIN
  - Inner Join
  - Left Join
  - Right Join

Câu lệnh JOIN

- Inner Join: Dùng để kết hợp các bảng dữ liệu
- Cú pháp:

SELECT [các cột] FROM <bảng 1> INNER JOIN <bảng 2> ON <điều kiện kết hợp> WHERE …. ODER BY …

Ví dụ:

SELECT Hoten, Tongdiem FROM Sinhvien INNER JOIN Hocsinh ON Sinhvien.Masv = Hocsinh.Masv WHERE Tongdiem >2.0 ORDER BY Tongdiem ASC

Lưu ý: nếu cần trả về kết quả là tất cả các cột của các bảng tham gia Inner Join ta áp dụng cú pháp:

Câu lệnh JOIN

Lưu ý: nếu cần trả về kết quả là tất cả các cột của các bảng tham gia Inner Join ta áp dụng cú pháp:

SELECT bảng 1.*, bảng 2.* [, bảng n.*] FROM bảng 1 INNER JOIN bảng 2 ON <điều kiện> …

Nếu trong các bảng cần kết nối có tên cột giống nhau thì câu lệnh SQL dạng SELECT cần chỉ rõ cột thuộc bảng nào. Trường hợp cả hai cùng lấy dữ liệu ra thì cần chuyển ánh xạ tên khác cho cột thông qua mệnh đề AS

Câu lệnh JOIN

- Left Join: Dùng để kết hợp các bảng dữ liệu khi muốn trả về kết quả là những mẫu tin của bảng bên trái tồn tại ứng với những mẫu tin ở bảng bên phải không tồn tại.
- Cú pháp: SELECT [các cột] FROM <bảng trái> LEFT JOIN <bảng phải> ON <điều kiện kết hợp> WHERE …. ODER BY …

Câu lệnh JOIN

- Right Join: Dùng để kết hợp các bảng dữ liệu khi muốn trả về kết quả là những mẫu tin của bảng bên phải tồn tại dù bảng bên trái không tồn tại
- Cú pháp:

SELECT [các cột] FROM <bảng trái> RIGHT JOIN <bảng phải> ON <điều kiện kết hợp> WHERE …. ODER BY …