WEB PROGRAMMING

BÀI 7.2

MYSQL

Kiểm dữ liệu - kiểu số

Kiểu dữ liệu	Mô tả
TINYINT	Lưu trữ một số nguyên có giá trị từ -128 đến -127 hoặc 0 đến 255
SMALLINT	Lưu trữ một số nguyên có giá trị từ -32768 đến 32767 hoặc 0 đến 65535
MEDIUMINT	Lưu trữ một số nguyên có giá trị từ -8388608 đến 8388607 hoặc 0 đến 16777215
INT	Lưu trữ một số nguyên có giá trị từ -2147483648 đến 2147483647 hoặc 0 đến 4294967295
BIGINT	Lưu trữ một số nguyên có giá trị từ -9223372036854775808 đến 9223372036854775807 hoặc 0 đến 18446744073709551615.
FLOAT	Lưu trữ một số thập phân loại nhỏ (Ví dụ: 567.25).
DOUBLE	Lưu trữ một số thập phân loại lớn.
DECIMAL	Lưu trữ như một chuỗi, cho phép một dấu thập phân cố định.

Kiểm dữ liệu - kiểu văn bản

Kiểu dữ liệu	Mô tả
CHAR(size)	Dùng để lưu trữ một chuỗi ký tự có chiều dài tối đa là 255 ký tự
VARCHAR(size)	Dùng để lưu trữ một chuỗi ký tự có chiều dài tối đa là 255 ký tự. Nếu đặt “size” lớn hơn 255 thì nó sẽ chuyển sang kiểu TEXT
TINYTEXT	Dùng để lưu trữ một chuỗi ký tự có chiều dài tối đa là 255 ký tự
TEXT	Dùng để lưu trữ một chuỗi ký tự có chiều dài tối đa là 65,535 ký tự
BLOB	Dùng để lưu trữ dữ liệu nhị phân tối đa là 65,535 byte
MEDIUMTEXT	Dùng để lưu trữ một chuỗi ký tự có chiều dài tối đa là 16,777,215 ký tự
MEDIUMBLOB	Dùng để lưu trữ dữ liệu nhị phân tối đa là 16,777,215 byte
LONGTEXT	Dùng để lưu trữ một chuỗi ký tự có chiều dài tối đa là 4,294,967,295 ký tự
LONGBLOB	Dùng để lưu trữ dữ liệu nhị phân tối đa là 4,294,967,295 byte

Kiểm dữ liệu - ngày tháng năm

Kiểu dữ liệu	Mô tả
DATE()	Lưu trữ một ngày theo định dạng YYYY-MM-DD (Ví dụ: 2016-09-12 tức là lưu ngày 12 tháng 9 năm 2016)
TIME()	Lưu trữ thời gian theo định dạng HH:MI:SS (Ví dụ 17:25:36 tức là lưu 17 giờ 25 phút 36 giây)
YEAR()	Lưu trữ một năm theo định dạng hai số hoặc bốn số
DATETIME()	Lưu trữ một ngày cùng với thời gian theo định dạng YYYY-MM-DD HH:MI:SS (Ví dụ: 2016-09-12 17:25:36 tức là lưu ngày 12 tháng 9 năm 2016 lúc 17 giờ 25 phút 36 giây)

Tạo CSDL

- CREATE DATABASE database_name;
- *Ví dụ*: CREATE DATABASE quanlynhanvien;

Tạo bảng

- CREATE TABLE table_name (
  column_name1data_type(size),
  column_name2 data_type(size),
  ....);
- *Ví dụ*: CREATE TABLE sinh_vien(
  Full_name VARCHAR(150),
  Gender VARCHAR(3),
  Age INT,
  City VARCHAR(50)
  );

Tạo khóa chính - cách 1

```
CREATE TABLE users (
  id INT(10) UNSIGNED NOT NULL PRIMARY KEY,
  email VARCHAR(70) NOT NULL
) ENGINE = INNODB;
```

Tạo khóa chính – cách 2

```
CREATE TABLE users (
  id INT(10) UNSIGNED NOT NULL,
  email VARCHAR(70) NOT NULL,
  PRIMARY KEY(id)
) ENGINE = INNODB;
```

Tạo khóa chính – cách 3

```
CREATE TABLE users (
  id INT(10) UNSIGNED NOT NULL,
  email VARCHAR(70) NOT NULL,
  PRIMARY KEY(id, email)
)
```

Tạo khóa chính – cách 4

ALTER TABLE table_name ADD PRIMARY KEY(primary_key_column);

Tạo khóa chính – sử dụng constraint

```
CREATE TABLE users (
  id INT(10) UNSIGNED NOT NULL,
  email VARCHAR(70) NOT NULL,
  CONSTRAINT pk_user PRIMARY KEY(id)
) ENGINE = INNODB;
```

Xóa khóa chính

```
ALTER TABLE users DROP PRIMARY KEY
```

Tạo khóa ngoại

```
CREATE TABLE Users(
    userid INT(11) NOT NULL PRIMARY KEY
AUTO_INCREMENT,
    username VARCHAR(50) NOT NULL ,
    email VARCHAR (50) NOT NULL ,
    groupid INT(11),
    FOREIGN KEY (groupid) REFERENCES
Groups(groupid)
);
```

Tạo khóa ngoại – dùng constraint

```
CREATE TABLE Users(
    userid INT(11) NOT NULL PRIMARY KEY
AUTO_INCREMENT,
    username VARCHAR(50) NOT NULL ,
    email VARCHAR (50) NOT NULL ,
    groupid INT(11),
    CONSTRAINT fk_group FOREIGN KEY (groupid)
REFERENCES Groups(groupid)
);
```

Tạo khóa ngoại – dùng ALTER

```
ALTER TABLE Users ADD FOREIGN KEY(groupid)
REFERENCES Groups(groupid);
```

- ***Hoặc***

```
ALTER TABLE Users ADD CONSTRAINT fk_group
FOREIGN KEY(groupid) REFERENCES Groups(groupid);
```

Xóa khóa ngoại

```
ALTER TABLE Users DROP FOREIGN KEY fk_group;
```

Sửa cấu trúc một bảng

- Thêm một cột mới vào bảng
- Xóa 1 cột trong bảng
- Sửa một cột trong bảng
- Đổi tên bảng

Thêm một cột mới

```
ALTER TABLE table_name
ADD column_name datatype;
```

Ví dụ:

```
ALTER TABLE sinh_vien
ADD SO_DIEN_THOAI INT;
```

Xóa một cột

ALTER TABLE table_name DROP COLUMN column_name;

Ví dụ:

ALTER TABLE sinh_vien
DROP COLUMN City;

Sửa một cột

ALTER TABLE table_name
MODIFY COLUMN column_name datatype;

Ví dụ:

ALTER TABLE sinh_vien
MODIFY COLUMN AGE SMALLINT;

Đổi tên bảng

```
ALTER TABLE table_name
RENAME TO new_table_name;
```

Ví dụ:

```
ALTER TABLE sinh_vien
RENAME TO sinh_vien_dai_hoc;
```

Lệnh insert – cú pháp 1

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

Ví dụ:

```
INSERT INTO sinh_vien VALUES ("Trinh Giao
Kim","Nam","44","Bình thuận");
```

Lệnh insert – cú pháp 2

```
INSERT INTO sinh_vien (Full_name, Gender, Age, City)
VALUES ("Trinh Giao Kim","Nam","44",”Bình thuận");
```

Lệnh select

```
SELECT * FROM name_table;
```

```
SELECT name_column1,
name_column2 FROM name_table;
```

Lệnh select - where

```
SELECT * FROM name_table
WHERE name_column operator value_column;
```

Các operator trong lệnh select

=	So sánh hai giá trị bằng nhau
<>	So sánh hai giá trị khác nhau
!=	So sánh hai giá trị khác nhau
>	Giá trị bên trái lớn hơn giá trị bên phải
<	Giá trị bên trái nhỏ hơn giá trị bên phải
>=	Giá trị bên trái lớn hơn hoặc bằng giá trị bên phải
<=	Giá trị bên trái nhỏ hơn hoặc bằng giá trị bên phải
BETWEEN	Giá trị nằm trong một khoảng nào đó (Sẽ nói rõ trong những bài hướng dẫn tiếp theo)
LIKE	Dùng trong tìm kiếm chuỗi ký tự (Sẽ nói rõ trong những bài hướng dẫn tiếp theo)
IN	Giá trị là một trong số các giá trị được nêu (Sẽ nói rõ trong những bài hướng dẫn tiếp theo)

Lệnh select – where - or và and

```
SELECT * FROM name_table
WHERE condition1 AND condition2 AND condition3;
```

Ví dụ 1:

```
SELECT * FROM sinh_vien
WHERE Gender="Nam" AND Age<=35;
```

Ví dụ 2:

```
SELECT Full_name FROM sinh_vien
WHERE (Gender="Nam" AND City="Can Tho") OR
(Gender="Nu" AND City="Soc Trang");
```

Lệnh order by

```
SELECT * FROM name_table
ORDER BY name_column ASC (hoặc DESC);
```

Ví dụ:

```
SELECT * FROM sinh_vien
 ORDER BY Age DESC;
```

Lệnh order by

SELECT * FROM name_table
ORDER BY name_column ASC (hoặc DESC);

Ví dụ:

SELECT * FROM sinh_vien
ORDER BY Age DESC;

Ví dụ 2:

SELECT * FROM sinh_vien
ORDER BY Gender DESC, Age ASC;

Lệnh DISTINCT

```
SELECT DISTINCT name_column
FROM name_table;
```

Ví dụ 1:

```
SELECT DISTINCT Age
FROM sinh_vien;
```

Ví dụ 2: lấy các cặp tuổi và giới tính khác nhau

```
SELECT DISTINCT Age, Gender
FROM sinh_vien;
```

Lệnh LIMIT

```
SELECT * FROM name_table
LIMIT num_ber;
```

Ví dụ: Lấy 3 dòng đầu

```
SELECT * FROM
sinh_vien LIMIT 3;
```

Ví dụ 2:

```
SELECT * FROM sinh_vien
WHERE Gender = 'Nam'
ORDER BY Age DESC
LIMIT 3;
```

Lệnh IN – NOT IN

```
SELECT * FROM name_table
WHERE name_column IN (value1, value2, value3);
```

Ví dụ:

```
SELECT *
FROM sinh_vien
WHERE Age IN (19, 22, 35);
```

Ví dụ: NOT IN

```
SELECT Full_name, City
FROM sinh_vien
WHERE Age NOT IN (19, 22);
```

Lệnh BETWEEN - NOT BETWEEN

```
SELECT * FROM name_table
WHERE name_column
BETWEEN value1 AND value2;
```

Ví dụ: BETWEEN

```
SELECT * FROM sinh_vien
WHERE Age BETWEEN 19 AND 22;
```

Vì dụ: NOT BETWEEN

```
SELECT Full_name, Age FROM sinh_vien
WHERE Age NOT BETWEEN 21 AND 36;
```

Lệnh LIKE và NOT LIKE

```
SELECT * FROM name_table
WHERE name_column LIKE chuỗi_muốn_tìm;
```

Ví dụ:

```
SELECT * FROM sinh_vien
WHERE Full_name LIKE "Tan Thuc Bao";
```

Ký tự đại diện	Mô tả
%	Đại diện cho *không* hoặc *nhiều* ký tự
_	Đại kiện cho *một* ký tự

Lệnh DELETE

```
DELETE FROM table_name
WHERE column_name = giá_trị;
```

Ví dụ:

```
DELETE FROM sinh_vien
WHERE Gender = "Nam";
```

Lệnh UPDATE

```
UPDATE table_name
SET column_name1=value1, column_name2=value2
WHERE column_name=value;
```

Ví dụ 1:

```
UPDATE sinh_vien
SET City="Hà Nội"
WHERE Gender = "Nam";
```

Ví dụ 2:

```
UPDATE sinh_vien SET Full_name="Bui
Nguyen Khanh", City="Tien Giang"
WHERE (Gender="Nam") AND (Age=35);
```

Lệnh INNER JOIN

```
SELECT *
FROM table1 INNER JOIN table2 ON
table1.column_name=table2.column_name;
```

Ví dụ:

```
SELECT hoadon.ID_hoadon,
khachhang.NAME_khachhang
FROM hoadon  INNER JOIN khachhang  ON
hoadon.ID_khachhang=khachhang.ID_khachhang;
```

Câu truy vấn con (1)

customers table

customer_id	last_name	first_name
4000	**Jackson**	**Joe**
5000	**Smith**	**Jane**
6000	**Ferguson**	**Samantha**
7000	**Reynolds**	**Allen**
8000	**Anderson**	**Paige**
9000	**Johnson**	**Derek**

Câu truy vấn con (2)

orders table

order_id	customer_id
1	7000
2	5000
3	8000
4	4000
5	7000

Câu truy vấn con (4)

In hoặc Exists

- **Ví dụ**: In ra thông tin của khách hàng có mua ít nhất 1 lần
- SELECT *

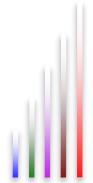
  FROM customers where customers.customer_id in

  (SELECT orders.customer_id

  FROM orders

)

Câu truy vấn con (5)



In hoặc Exists

- **Ví dụ**: In ra thông tin của khách hàng có mua ít nhất 1 lần
- SELECT *

```
FROM customers
where EXISTS
  (SELECT *
   FROM orders
   WHERE customers.customer_id = orders.customer_id)
```

Câu truy vấn con (5)

4000	Jackson	Joe
5000	Smith	Jane
7000	Reynolds	Allen
8000	Anderson	Paige

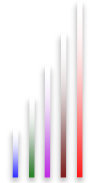
Câu truy vấn con (3)

In hoặc Exists

- **Ví dụ**: Tìm các số hóa đơn mua cùng lúc 2 sản phẩm có mã số “BB01” và “BB02”.
  - select distinct sohd
    from CTHD where masp='BB01' and sohd **IN**
    (select distinct sohd from CTHD where masp='BB02')

    select distinct A.sohd
    from CTHD A where A.masp='BB01' and
    EXISTS (select * from CTHD B
    where B.masp='BB02‘ and
    A.sohd=B.sohd)
  - A.sohd=B.sohd: Có nghĩa trên cùng một hóa đơn

Câu truy vấn con (6)



Not In hoặc Not Exists

- **Ví dụ**: Tìm những khách hàng không mua hàng
- SELECT * FROM customers
  where customers.customer_id not in
  (SELECT orders.customer_id
  FROM orders
)

Câu truy vấn con (7)

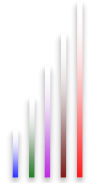
Not In hoặc Not Exists

- **Ví dụ**: Tìm những khách hàng không mua hàng
- SELECT * FROM customers
  WHERE NOT EXISTS
  (SELECT *
  FROM orders
  WHERE customers.customer_id = orders.customer_id);

Câu truy vấn con (8)

6000	Ferguson	Samantha
9000	Johnson	Derek

Câu truy vấn con (9)



Not In hoặc Not Exists

- **Ví dụ**: Tìm các số hóa đơn có mua sản phẩm mã số ‘BB01’ nhưng không mua sản phẩm mã số ‘BB02’.
  - select distinct sohd
    from CTHD where masp='BB01' and sohd **NOT IN**
    (select distinct sohd from CTHD where masp='BB02')
  - select distinct A.sohd
    from CTHD A where A.masp='BB01' and
    NOT EXITST (select * from CTHD B
    where B.masp='BB02‘ and
    A.sohd=B.sohd)

Nhận xét IN và EXISTS

- IN
- Thuộc tính ở mệnh đề SELECT của truy vấn con phải có cùng kiểu dữ liệu với thuộc tính ở mệnh đề WHERE của truy vấn cha
- EXISTS
- Không cần có thuộc tính,hằng số hay biểu thức nào khác đứng trước
- Không nhất thiết liệt kê tên thuộc tính ở mệnh đề SELECT của truy vấn con
- Những câu truy vấn có =ANY hay IN đều có thể chuyển thành câu truy vấn có EXISTS

ALL

- Toán tử ALL trả về TRUE nếu tất cả các giá trị truy vấn con đáp ứng điều kiện.

Cú pháp ALL

SELECT tên_cột(s) **FROM** tên_bảng

WHERE tên_cột **to**án_tử **ALL**

(**SELECT** tên_cột **FROM** tên_bảng **WHERE** điều_kiện);

Ví dụ - bảng sản phẩm

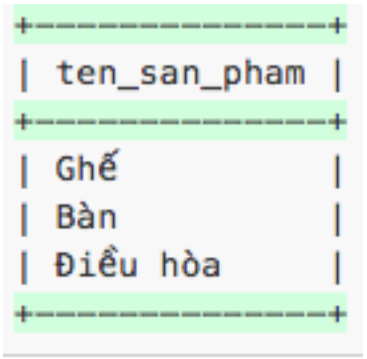
ID_san_pham	ten_san_pham	ID_nha_cung_cap	ID_danh_muc	gia
1	Ghế	1	1	180000
2	Bàn	1	1	190000
3	Quạt	1	2	500000
4	Điều hòa	2	2	1000000
5	Đèn	2	2	50000

Ví dụ - bảng chi tiết đơn hàng

ID_don_hang	ID_san_pham	so_luong
2	1	12
4	2	10
5	5	5
6	3	9
1	4	40

Ví dụ

```
SELECT ten_san_pham FROM san_pham
WHERE id_san_pham = ANY
    (SELECT id_san_pham FROM
    chi_tiet_don_hang
       WHERE so_luong > 9);
```



ten_san_pham
Ghế
Bàn
Điều hòa

Ví dụ

```
SELECT ten_san_pham FROM san_pham
WHERE id_san_pham = ALL
  (SELECT id_san_pham FROM chi_tiet_don_hang
   WHERE so_luong > 10);
```

UNION (hội)

- Trả về các bản ghi nằm trong lệnh Select 1 hoặc trong lệnh select 2
- ***Ví dụ***: In ra MANV của các nhân viên làm việc cho dự án có MADA =1 và các nhân viên làm việc cho dự án có MADA = 2

UNION (hội)

```
(SELECT MANV
FROM NHÂNVIÊN_DỰÁN WHERE MADA = 1)
UNION
(SELECT MANV
FROM NHÂNVIÊN_DỰÁN WHERE MADA = 2)
```

Các hàm tính toán và gom nhóm (1)

Các hàm tính toán cơ bản

- COUNT: Đếm số bộ dữ liệu của thuộc tính
- MIN: Tính giá trị nhỏ nhất
- MAX: Tính giá trị lớn nhất
- AVG: Tính giá trị trung bình
- SUM: Tính tổng giá trị các bộ dữ liệu

Dữ liệu

NHANVIEN					
MANV	**HOTEN**	**PHAI**	**MANQL**	**PHONG**	**LUONG**
NV001	Nguyễn Ngọc Linh	Nữ	Null	NC	2.800.000
NV002	Đinh Bá Tiến	Nam	NV002	DH	2.000.000
NV003	Nguyễn Văn Mạnh	Nam	NV001	NC	2.300.000
NV004	Trần Thanh Long	Nam	NV002	DH	1.800.000
NV005	Nguyễn Thị Hồng Vân	Nữ	NV001	NC	2.500.000
NV006	Nguyễn Minh	Nam	NV002	DH	2.000.000
NV007	Hà Duy Lập	Nam	NV003	NC	1.800.000
NV008	Trần Kim Duyên	Nữ	NV003	NC	1.800.000
NV009	Nguyễn Kim Anh	Nữ	NV003	NC	2.000.000

Ví dụ

1. Tính lương thấp nhất, cao nhất, trung bình và tổng lương của tất cả các nhân viên.
2. Có tất cả bao nhiêu nhân viên
3. Bao nhiêu nhân viên có người quản lý
4. Bao nhiêu phòng ban có nhân viên trực thuộc
5. Tính lương trung bình của các nhân viên
6. Tính lương trung bình của các nhân viên theo từng phòng ban

Giải

1. Tính lương thấp nhất, cao nhất, trung bình và tổng lương của tất cả các nhân viên.

```
SELECT   min(luong) as thapnhat,
         max(luong) as caonhat,
         avg(luong) as trungbinh,
         sum(luong) as tongluong

FROM     NhanVien
```

Giải

2. Có tất cả bao nhiêu nhân viên

SELECT **count(*)** FROM NhanVien

3. Bao nhiêu nhân viên có người quản lý

- Select count(*) FROM NhanVien WHERE manql is not null
- SELECT **count(Manql)** FROM NhanVien

4. Bao nhiêu phòng ban có nhân viên trực thuộc

SELECT **count(distinct phong)** FROM NhanVien

Các hàm tính toán và gom nhóm (2)

Gom nhóm: mệnh đề GROUP BY

- Sử dụng hàm gom nhóm trên các bộ trong quan hệ.
- Mỗi nhóm bộ bao gồm tập hợp các bộ có cùng giá trị trên các thuộc tính gom nhóm
- Hàm gom nhóm áp dụng trên mỗi bộ độc lập nhau.
- SQL có mệnh đề GROUP BY để chỉ ra các thuộc tính gom nhóm, các thuộc tính này phải xuất hiện trong mệnh đề SELECT

Giải

5. Tính lương trung bình của các nhân viên

```
SELECT      avg(LUONG) as LUONGTB
FROM        NhanVien
```

6. Tính lương trung bình của các nhân viên theo từng phòng ban.

```
SELECT      phong, avg(LUONG) as LUONGTB
FROM        NhanVien
GROUP BY    phong
```

Các hàm tính toán và gom nhóm (3)

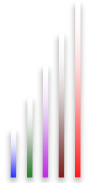
Điều kiện sau gom nhóm: mệnh đề HAVING

- Lọc kết quả theo điều kiện, sau khi đã gom nhóm
- Điều kiện ở HAVING được thực hiện sau khi gom nhóm, các điều kiện có liên quan đến thuộc tính Group By

- **Ví dụ**: tìm phòng có số lượng nhân viên "Nữ" trên 5 người

```
SELECT      phong
FROM        NhanVien
WHERE       phai = 'Nữ'
GROUP BY    phong
HAVING      count(manv) > 5
```

MySQL không hỗ trợ các lệnh



- ANY (một vài)
- SOME (một vài)
- TOP
- INTERSECT (Giao)
- EXCEPT (Trừ)