LẬP TRÌNH ỨNG DỤNG QUẢN LÝ TRÊN WEB

Bài 1: CSDL MySQL

1. Tổng quan
2. Bảng
3. Toán tử
4. Import và export dữ liệu
5. Phát biểu SQL

1. Tổng quan

- Giới thiệu Cơ sở dữ liệu (CSDL)
- CSDL MySQL

Giới thiệu CSDL

- Khái niệm
- Chức năng
- Các đối tượng chính của CSDL quan hệ
- Hệ quản trị CSDL
- SQL (Structure Query Language)

Giới thiệu CSDL

- Khái niệm
  - CSDL là một tập hợp dữ liệu được lưu trữ một cách có tổ chức nhằm giúp việc xem, tìm kiếm và lấy thông tin được nhanh chóng và chính xác, giúp giảm công sức và thời gian quản lý thông tin cần thiết

Giới thiệu CSDL

- Chức năng
  - Lưu trữ
    - Dữ liệu được lưu trên đĩa và người dùng có thể chuyên đổi dữ liệu từ CSDL này sang CSDL khác
    - Tùy theo quy mô của ứng dụng mà chúng ta có thể chọn CSDL lớn hay nhỏ
  - Truy cập
    - Tùy vào mục đích và yêu cầu mà có các mức độ truy cập: cục bộ, chia sẽ, truy cập dữ liệu giữa các CSDL khác nhau

Giới thiệu CSDL

- Chức năng
  - Tổ chức
    - Cách tổ chức tùy thuộc: mô hình CSDL, cách phân tích, thiết kế và đặc điểm riêng của từng ứng dụng
  - Xử lý
    - xử lý dữ liệu là việc sử dụng các truy vấn để truy xuất hay cập nhật dữ liệu theo yêu cầu của người dùng.

Giới thiệu CSDL

- Các đối tượng chính của CSDL quan hệ
  - Bảng dữ liệu (table)
  - Quan hệ

Giới thiệu CSDL

- Các đối tượng chính của CSDL quan hệ
  - Bảng dữ liệu (table)
    - Là thành phần trung tâm của CSDL, được dùng để lưu trữ thông tin của CSDL
    - Gồm hai thành phần: dòng và cột

Giới thiệu CSDL

- Các đối tượng chính của CSDL quan hệ
  - Bảng dữ liệu (table)
    - Cột: là một khối dữ liệu trong bảng, có cùng loại dữ liệu, có các thông tin chính:
      - Tên cột: dùng để phân biệt với các cột khác trong bảng. Tên cột trong bảng phải duy nhất và không dùng các ký tự đặc biệt.
      - Kiểu dữ liệu của cột: xác định loại giá trị nào được phép lưu trữ trong cột
    - Ví dụ: Bảng khách hàng KHACH_HANG có các cột sau: MKH (mã khách hàng), TEN_KH (tên khách hàng), PHAI (phái), …

Giới thiệu CSDL

- Các đối tượng chính của CSDL quan hệ
  - Bảng dữ liệu (table)
    - Dòng: là tập hợp các thông tin của tất cả cột dữ liệu trong bảng
    - Ví dụ: Bảng khách hàng KHACH_HANG có các dòng dữ liệu sau:

MKH	TEN_KH	PHAI	DIA_CHI	DT	EMAIL
KH001	**Trần Văn An**	**0**	**123 Nguyễn Du**	**8123456**	**tvan@yahoo.com**
KH002	**Nguyễn Thanh An**	**0**	**30 Lê Thánh Tôn**	**9852147**	**ntan@yahoo.com**
KH003	**Lê Thanh Thảo**	**1**	**22bis Pasteur**	**8976431**	**ltthao@gmail.com**

    - Mỗi dòng trong bảng khách hàng lưu trữ thông tin về một khách hàng trong thực tế.

Giới thiệu CSDL

- Các đối tượng chính của CSDL quan hệ
  - Quan hệ
    - Là thành phần được dùng để tạo mối liên kết giữa các bảng dữ liệu với nhau nhằm đảm bảo tính nhất quán, đúng đắn của dữ liệu trong CSDL.
    - Có ba loại quan hệ chính:
      - Quan hệ 1 – 1
      - Quan hệ 1 – nhiều
      - Quan hệ nhiều – nhiều

Giới thiệu CSDL

- Các đối tượng chính của CSDL quan hệ
  - Quan hệ
    - Quan hệ 1 – 1 (One to One)
      - Mô tả mối quan hệ giữa hai bảng mà trong đó một dòng dữ liệu bên bảng này có liên hệ với duy nhất với một dòng dữ liệu bên bảng kia và ngược lại
      - Ví dụ:
        - » **Một** nhân viên chỉ có **một** sơ yếu lý lịch
        - » **Một** sơ yếu lý lịch chỉ **thuộc về một** nhân viên

Giới thiệu CSDL

- Các đối tượng chính của CSDL quan hệ
  - Quan hệ
    - Quan hệ 1 – nhiều (One to Many)
      - Mô tả mối quan hệ giữa hai bảng mà trong đó một dòng dữ liệu bên bảng này có liên hệ với nhiều dòng dữ liệu bên bảng kia và một dòng dữ liệu bên bảng kia sẽ có liên hệ với duy nhất với một dòng dữ liệu bên bảng này. Quan hệ này thường gặp nhất trong CSDL
      - Ví dụ:
        - » **Một** phòng có **nhiều** nhân viên
        - » **Một** nhân viên chỉ **thuộc về một** phòng

Giới thiệu CSDL

- Các đối tượng chính của CSDL quan hệ
  - Quan hệ
    - Quan hệ nhiều – nhiều (Many to Many)
      - Mô tả mối quan hệ giữa hai bảng mà trong đó một dòng dữ liệu bên bảng này có liên hệ với nhiều dòng dữ liệu bên bảng kia và ngược lại. Trong CSDL không lưu trữ quan hệ nhiều nhiều vì vậy khi gặp quan hệ này, chúng ta sẽ chuyển thành các quan hệ một nhiều

Giới thiệu CSDL

- Các đối tượng chính của CSDL quan hệ
  - Quan hệ
    - Quan hệ nhiều – nhiều (Many to Many)
      - Ví dụ:
        - » **Một** giáo viên dạy **nhiều** lớp học.
        - » **Một** lớp học có **nhiều** giáo viên.

      - Quan hệ này tương đương với hai quan hệ một nhiều sau:

Giới thiệu CSDL

- Hệ quản trị CSDL
  - Quản lý các dữ liệu được lưu trữ bên trong các CSDL, giúp cho CSDL dễ dàng đến được với người dùng khi cần truy cập các thông tin khác nhau.
  - Có khả năng lưu trữ dữ liệu và cho phép dữ liệu có thể trao đổi với các CSDL khác, và có khả năng
    - Bảo vệ dữ liệu
    - Duy trì dữ liệu
    - Quản lý các giao dịch
    - …

Giới thiệu CSDL

- SQL
  - Là loại ngôn ngữ cho phép thực hiện các thao tác rút trích, tính toán, cập nhật trên các dữ liệu được lưu trữ trong CSDL

CSDL MySQL

- Giới thiệu
- Đặc điểm
- Các tập tin vật lý lưu trữ CSDL
- Quy tắc đặt tên
- Tạo CSDL
- Xóa CSDL

CSDL MySQL

- Giới thiệu
  - CSDL MySQL là tập hợp các đối tượng: bảng, bảng ảo,… cho phép người dùng lưu trữ và truy xuất các thông tin đã được tổ chức và lưu trữ bên trong đó.

CSDL MySQL

- Đặc điểm
  - Sử dụng cho các ứng dụng Web có quy mô vừa và nhỏ.
  - Để thực hiện các thao tác trên CSDL, có thể sử dụng giao diện đồ họa hay dùng dòng lệnh (command line)

CSDL MySQL

- Các tập tin vật lý lưu trữ CSDL
  - Mỗi bảng sẽ được lưu trữ dưới ba tập tin vật lý:
    - **.frm : lưu định dạng (cấu trúc) của bảng**
    - **.MYD : lưu nội dung của bảng**
    - **.MYI : lưu chỉ mục của bảng**
  - Các tập tin này sẽ được tự động lưu trữ trong thư mục: wamp\mysql\data\tên_CSDL

CSDL MySQL

- Chiều dài của tên CSDL, bảng, chỉ mục, cột, định danh

Loại	Chiều dài tối đa (bytes)	Chiều dài tối đa (ký tự không dấu)
CSDL (database)	64	64
Bảng (Table)	64	64
Chỉ mục (Index)	64	64
Cột (Column)	64	64
Định danh (Alias)	255	255

CSDL MySQL

- Quy tắc đặt tên
  - Tên không kết thúc bằng khoảng trắng
  - Tên CSDL không có các ký tự **/, \, ., :, *, ‘’, <, >**
  - Tên bảng không có các ký tự **/, \, ., :, *, ‘’, <, >, |**
  - Chiều dài của tên tối đa là 64 ký tự không dấu. Khi sử dụng các ký tự đa byte thì chiều dài sẽ dựa trên tổng số byte của tất cả các ký tự được dùng.

CSDL MySQL

- Tạo CSDL
  - Các thuộc tính của CSDL
    - **Tên CSDL: phải duy nhất trong một hệ quản trị CSDL MySQL**
    - **Vị trí lưu trữ: Khi tạo mới một CSDL hệ thống sẽ tự động tạo ra một thư mục có tên của CSDL và được lưu tại thư mục wamp\mysql\data**
  - Có hai cách để tạo một CSDL là dùng giao diện đồ họa hoặc dùng dòng lệnh

CSDL MySQL

- Tạo CSDL
  - Giao diện đồ họa
    - **Bước 1: khởi động PHPMyAdmin**

CSDL MySQL

- Tạo CSDL
  - Giao diện đồ họa
    - **Bước 2: nhập tên CSDL muốn tạo vào mục Create New Database và chọn các thông tin khác**
    - **Bước 3: Nhấn "Create" để hoàn thành việc tạo CSDL**

CSDL MySQL

- Tạo CSDL
  - Dùng câu lệnh CREATE DATABASE
    - **Vào giao diện dòng lệnh**

CSDL MySQL

- Tạo CSDL
  - Dùng câu lệnh CREATE DATABASE
    - **Cú pháp:**

      CREATE DATABASE Tên_CSDL

      [[DEFAULT] CHARACTER SET <character set name>]

      [[DEFAULT] COLLATE <collation name>]
    - **Với:**
      - CHARACTER SET: xác định bộ ký tự mặc định cho CSDL mới
      - COLLATE: xác định bộ collation
      - Character set name: tên của một bộ mã bao gồm các ký tự, ký số, và biểu tượng để lưu trữ các thông tin trong CSDL
      - Collation name: tên của bộ mã tùy theo từng khu vực dựa trên bộ mã chuẩn character set name

CSDL MySQL

- Tạo CSDL
  - Dùng câu lệnh CREATE DATABASE
    - **Ví dụ: Tạo CSDL ql_ban_sua**

```
CREATE DATABASE ql_ban_sua
CHARACTER SET utf-8
COLLATE utf8_unicode_ci
Hay
CREATE DATADASE ql_ban_sua
```

CSDL MySQL

- Xóa CSDL
  - Giao diện đồ họa
    - **Bước 1: Chọn CSDL cần xóa, sau đó nhấn DROP**

CSDL MySQL

- Xóa CSDL
  - Giao diện đồ họa
    - **Bước 2: Xác nhận việc xóa CSDL đã chọn, nhấn OK để xóa**

CSDL MySQL

- Xóa CSDL
  - Dùng câu lệnh DROP DATABASE
    - **Cú pháp:**

      DROP DATABASE TÊN_CSDL
    - **Ví dụ: Xóa CSDL ql_ban_sua**

```
DROP DATABASE ql_ban_sua
```

2. Bảng (table)

- Khái niệm
- Thuộc tính
- Thao tác với bảng

Bảng (table)

- Khái niệm
  - Dùng để lưu trữ thông tin của những đối tượng, thực thể trong thế giới thực muốn được lưu trữ vào máy tính.
  - Các thông tin trong bảng sẽ được tổ chức thành các dòng (row) và các cột (column).
  - Mỗi dòng thông tin trong bảng là duy nhất do có một hoặc nhiều cột làm khóa chính. Dữ liệu của cột làm khóa chính không trùng lắp trong bảng.
  - Các bảng thường có quan hệ với nhau giúp trao đổi và chia sẻ thông tin.

Bảng (table)

- Thuộc tính
  - Tên bảng (table name)
    - Do người dùng tạo ra
    - Duy nhất trong CSDL

Bảng (table)

- Thuộc tính
  - Các thuộc tính của cột trong bảng
    - Tên cột
      - Do người dùng tạo ra
      - Duy nhất trong bảng
      - Ví dụ: Đặt tên cho cột mã sữa trong bảng sữa là: Ma_sua

Bảng (table)

- Thuộc tính
  - Các thuộc tính của cột trong bảng
    - Kiểu dữ liệu (type)
      - Xác định loại dữ liệu được lưu trữ trên từng cột

Bảng (table)

- Thuộc tính
  - Các thuộc tính của cột trong bảng
    - Các kiểu dữ liệu số nguyên

Kiểu dữ liệu	Kích thước	Miền giá trị
Tinyint	1 byte	-127 => 128 hay 0..255
Smallint	2 bytes	-32768 => 32767 hay 0..65535
Mediumint	3 bytes	-8388608 => 838860 hay 0..16777215
Int	4 bytes	-2^{31} => $2^{31}-1$ hay 0..$2^{32}-1$
Bigint	8 bytes	-2^{63} => $2^{63}-1$ hay 0..$2^{64}-1$

Bảng (table)

- Thuộc tính
  - Các thuộc tính của cột trong bảng
    - Kiểu dữ liệu true/false

Kiểu dữ liệu	Kích thước	Miền giá trị
Bool / boolean	1 byte	Có hai giá trị là True và False

- Ví dụ

Select if(0, “true” , “false”) → false
Select if(2, “true” , “false”) → true
select if(0 = false, ‘true’, ‘false’) → true
select if(2 = false, ‘true’, ‘false’) → false

Bảng (table)

- Thuộc tính
  - Các thuộc tính của cột trong bảng
    - Kiểu dữ liệu dạng số thập phân: decimal và numeric
      - Là những kiểu dữ liệu được dùng để lưu trữ những giá trị số cụ thể.
      - Giá trị được lưu với định dạng nhị phân.
      - Cú pháp:

        Decimal(M[, N])
      - Trong đó: M là tổng ký số và N là số ký số thập phân

Kiểu dữ liệu	Kích thước	Miền giá trị
Decimal/ Numeric	4 bytes	

Bảng (table)

- Thuộc tính
  - Các thuộc tính của cột trong bảng
    - Các kiểu dữ liệu số thực

Kiểu dữ liệu	Kích thước	Miền giá trị
Float	4 bytes	-3.402823466E+38 => -1.175494351E-38; 0; 1.175494351E-38 => 3.402823466E+38
Double	8 bytes	-1.7976931348623157E+308 => -2.2250738585072014E-308; 0; 2.2250738585072014E-308 => 1.7976931348623157E+308

Bảng (table)

- Thuộc tính
  - Các thuộc tính của cột trong bảng
    - Các kiểu dữ liệu dạng ngày/giờ (date/time)

Kiểu dữ liệu	Miền giá trị	Diễn giải
Date	'1000-01-01' => '9999-12-31'	Ngày với định dạng yyyy-mm-dd
Datetime	'1000-01-01 00:00:00' => '9999-12-31 23:59:59'	Ngày giờ với định dạng yyyy-mm-dd hh:mm:ss
Time	'00:00:00' => '23:59:59'	Giờ với định dạng hh:mm:ss
Year[(2|4)]	4 ký số: '1901' => '2155' 2 ký số: '1970' => '2069'	Năm với định dạng 2 ký số hoặc 4 ký số
Timestamp [(kích cỡ định dạng)]	'1970-01-01 00:00:01'	Timestamp trình bày dưới dạng yyyy-mm-dd hh:mm:ss

Bảng (table)

- Thuộc tính
  - Các thuộc tính của cột trong bảng
    - Bảng kích cỡ định dạng

Kiểu dữ liệu	Định dạng
Timestamp	yyyymmddhhmmss
Timestamp(14)	yyyymmddhhmmss
Timestamp(12)	yymmddhhmmss
Timestamp(10)	yymmddhhmm
Timestamp(8)	yyyymmdd
Timestamp(6)	yymmdd
Timestamp(4)	yymm
Timestamp(2)	yy

Bảng (table)

- Thuộc tính
  - Các thuộc tính của cột trong bảng
    - Các kiểu dữ liệu kiểu chuỗi

Kiểu dữ liệu	Miền giá trị	Diễn giải
Char	1 => 255 ký tự	Chuỗi cố định
Varchar	1 => 255 ký tự	Chuỗi động
Tinyblob	1 => 2^8 -1 bytes (255 bytes)	Kiểu đối tượng nhị phân cỡ 255 ký tự
Tinytext	1 => 2^8-1 ký tự (255 ký tự)	Kiểu đối tượng chuỗi kích cỡ 255 ký tự
Blob	1 => 2^{16}-1 bytes (65535 bytes)	Kiểu blob cỡ 65535 ký tự
Text	1 => 2^{16}-1 ký tự (65535 ký tự)	Kiểu chuỗi dạng văn bản cỡ 65535 ký tự
Mediumblob	1 => 2^{24}-1 bytes (16777215 bytes)	Kiểu blob vừa cỡ 16777215 ký tự
Mediumtext	1 => 2^{24}-1 ký tự (16777215 ký tự)	Kiểu chuỗi dạng văn bản vừa 16777215 ký tự
Longblob	1 => 2^{32}-1 bytes (4GB)	Kiểu blob lớn khoảng 4GB ký tự
Longtext	1 => 2^{32}-1 ký tự (4GB)	Kiểu chuỗi dạng văn bản lớn khoảng 4GB ký tự

Bảng (table)

- Thuộc tính
  - Các thuộc tính của cột trong bảng
    - Các kiểu dữ liệu kiểu chuỗi
      - Sự khác nhau cơ bản giữa kiểu char và kiểu varchar

Giá trị	Char(4)	Số bytes	Varchar(4)	Số bytes
''	' '	4 bytes	''	1 byte
'ab'	'ab '	4 bytes	'ab'	3 bytes
'abcd'	'abcd'	4 bytes	'abcd'	5 bytes
'abcdefgh'	'abcd'	4 bytes	'abcd'	5 bytes

Bảng (table)

- Thuộc tính
  - Các thuộc tính của cột trong bảng
    - Độ dài dữ liệu (length/value):
      - Quy định độ dài dữ liệu mà cột sẽ lưu trữ đối với kiểu dữ liệu chuỗi hoặc số.
      - Ví dụ: độ dài dữ liệu của cột Ma_sua là 6
    - Kiểu hiển thị (collation)
      - Quy định bảng mã hiển thị cho dữ liệu trong cột.
      - Ví dụ: utf8_unicode_ci, utf8_bin

Bảng (table)

- Thuộc tính
  - Các thuộc tính của cột trong bảng
    - Thuộc tính (attribute):
      - Quy định thuộc tính cho cột, mặc định là không quy định.
    - Cho phép để trống dữ liệu (NULL)
      - Quy định cột có thể để trống hay không khi thêm, cập nhật dữ liệu.
    - Giá trị mặc định (default)
      - Là giá trị sẽ thêm vào cho cột khi thêm mới mẩu tin mà người dùng không nhập giá trị cho cột này

Bảng (table)

- Thuộc tính
  - Các thuộc tính của cột trong bảng
    - Thuộc tính mở rộng (extra)
      - Cho phép thiết lập thuộc tính auto increment (cột có giá trị tự động tăng dần khi thêm mới mẩu tin) cho khóa chính.
    - Ghi chú (comment)
      - Chuỗi chú thích cho cột

Bảng (table)

- Thao tác với bảng
  - Tạo bảng
  - Thay đổi cấu trúc bảng
  - Thêm cột mới trong bảng
  - Sửa đổi kiểu dữ liệu của cột
  - Hủy cột trong bảng
  - Xóa bảng

Bảng (table) – thao tác với bảng

- Tạo bảng – giao diện đồ họa

Bảng (table) – thao tác với bảng

- Tạo bảng – giao diện đồ họa
  - Bước 1:

Bảng (table) – thao tác với bảng

- Tạo bảng – giao diện đồ họa
  - Bước 2:

Field	Type	Length/Values[1]	Collation	Attributes
Ma_hang_sua	VARCHAR	20		
Ten_hang_sua	VARCHAR	100		
Dia_chi	VARCHAR	200		
Dien_thoai	VARCHAR	20		
Email	VARCHAR	100		

Bảng (table) – thao tác với bảng

- Tạo bảng – giao diện đồ họa
  - Bước 3:
    - Quy định cột nào phải nhập dữ liệu
    - Giá trị ban đầu của cột,…
    - Thiết lập khóa chính cho bảng
    - Ghi chú cho từng cột (nếu cần)

Bảng (table) – thao tác với bảng

- Tạo bảng – giao diện đồ họa
  - Bước 4:

Bảng (table) – thao tác với bảng

- Tạo bảng – CREATE TABLE
  - Tạo bảng đơn giản
    - Cú pháp:

```
CREATE TABLE Tên_bảng
(
   Tên_cột_1 kiểu_dữ_liệu[(kích_cỡ)] [NOT NULL],
   Tên_cột_2 kiểu_dữ_liệu[(kích cỡ)] [NOT NULL],
   ...
)
```

– NOT NULL: không cho phép để trống dữ liệu trong cột

Bảng (table) – thao tác với bảng

- Tạo bảng – CREATE TABLE
  - Tạo bảng đơn giản
    - Ví dụ: tạo bảng hang_sua với hai cột dữ liệu không được phép bỏ trống là Ma_hang_sua và Ten_hang_sua

```
CREATE TABLE hang_sua
(
        Ma_hang_sua varchar(20) NOT NULL,
        Ten_hang_sua varchar(100) NOT NULL,
        Dia_chi varchar(200),
        Dien_thoai varchar(20),
        Email varchar(100)
)
```

Bảng (table) – thao tác với bảng

- Tạo bảng – CREATE TABLE
  - Tạo bảng có giá trị mặc định
    - Cú pháp:

```
CREATE TABLE Tên_bảng
(
   Tên_cột_1 kiểu_dữ_liệu[(kích_cỡ)] DEFAULT giá_trị,
   Tên_cột_2 kiểu_dữ_liệu[(kích cỡ)] [NOT NULL],
   ...
)
```

Bảng (table) – thao tác với bảng

- Tạo bảng – CREATE TABLE
  - Tạo bảng có giá trị mặc định
    - Ví dụ: tạo bảng có tên là khach_hang theo cấu trúc sau với cột điện thoại có giá trị mặc định là (08)___:

Field Name	Field Type	Field Size	Description
Ma_Khach_Hang	varchar	5	Not null
Ten_Khach_Hang	varchar	100	Not null
Phai	tinyint (bool)	1	1: Nữ , 0: Nam
Dia_chi	varchar	200	
Dien_thoai	varchar	20	(08)___
Email	varchar	100	

Bảng (table) – thao tác với bảng

- Tạo bảng – CREATE TABLE
  - Tạo bảng có giá trị mặc định
    - Ví dụ: tạo bảng có tên là khach_hang theo cấu trúc sau với cột điện thoại có giá trị mặc định là (08)___:

```
CREATE TABLE khach_hang
(
        Ma_khach_hang VARCHAR(5) NOT NULL ,
        Ten_khach_hang VARCHAR(100) NOT NULL ,
        Phai BOOL ,
        Dia_chi VARCHAR(200),
        Dien_thoai VARCHAR(20) DEFAULT '(08)___',
        Email VARCHAR(100)
)
```

Bảng (table) – thao tác với bảng

- Tạo bảng – CREATE TABLE
  - Thiết lập khóa chính (primary key) và dữ liệu duy nhất (unique)
    - Để kiểm tra tính duy nhất của dữ liệu bên trong bảng ta có thể thiết lập khóa chính (primary key) hoặc có thể thiết lập duy nhất (unique) cho bảng.
    - Phân biệt giữa primary và unique

Primary key

Một bảng chỉ có một khóa chính, có thể có một hay nhiều cột tham gia làm khóa chính => Primary key chỉ xuất hiện một lần khi tạo cấu trúc bảng

Unique

Unique kiểm tra tính duy nhất của dữ liệu trên các cột trong bảng => Unique có thể được xuất hiện nhiều lần khi tạo cấu trúc bảng

Bảng (table) – thao tác với bảng

- Tạo bảng – CREATE TABLE
  - Thiết lập khóa chính (primary key) và dữ liệu duy nhất (unique)
    - Cú pháp tạo khóa chính

    PRIMARY KEY (Các_cột_tham_gia_làm_khóa_chính)
    - Đối với bảng mà khóa chính chỉ có một cột =>có thể khai báo khóa chính ngay sau khi khai báo cột.
    - Đối với bảng có nhiều cột tham gia làm khóa => thiết lập các cột trước rồi tạo khóa chính sau

Bảng (table) – thao tác với bảng

- Tạo bảng – CREATE TABLE
  - Thiết lập khóa chính (primary key) và dữ liệu duy nhất (unique)
    - Ví dụ: tạo bảng hang_sua như trên với khóa chính là cột Ma_hang_sua

```
CREATE TABLE hang_sua
(
        Ma_hang_sua varchar(20) NOT NULL PRIMARY KEY,
        Ten_hang_sua varchar(100) NOT NULL,
        Dia_chi varchar(200),
        Dien_thoai varchar(20),
        Email varchar(100),
)
```

Bảng (table) – thao tác với bảng

- Tạo bảng – CREATE TABLE
  - Thiết lập khóa chính (primary key) và dữ liệu duy nhất (unique)
    - Ví dụ: tạo bảng hang_sua như trên với khóa chính là cột Ma_hang_sua

```
CREATE TABLE hang_sua
(
        Ma_hang_sua varchar(20) NOT NULL,
        Ten_hang_sua varchar(100) NOT NULL,
        Dia_chi varchar(200),
        Dien_thoai varchar(20),
        Email varchar(100),
        PRIMARY KEY(Ma_hang_sua)
)
```

Bảng (table) – thao tác với bảng

- Tạo bảng – CREATE TABLE
  - Thiết lập khóa chính (primary key) và dữ liệu duy nhất (unique)
    - Cú pháp tạo cột duy nhất dữ liệu:

    UNIQUE (các_cột_duy_nhất)

    - Đối với bảng chỉ có một cột duy nhất dữ liệu => có thể khai báo duy nhất ngay sau khi khai báo cột.
    - Đối với bảng có nhiều cột duy nhất dữ liệu => thiết lập các cột trước rồi tạo duy nhất sau

Bảng (table) – thao tác với bảng

- Tạo bảng – CREATE TABLE
  - Thiết lập khóa chính (primary key) và dữ liệu duy nhất (unique)
    - Ví dụ: tạo bảng hang_sua như trên với cột Ten_hang_sua là duy nhất

```
CREATE TABLE hang_sua
(
        Ma_hang_sua varchar(20) NOT NULL PRIMARY KEY,
        Ten_hang_sua varchar(100) NOT NULL UNIQUE,
        Dia_chi varchar(200),
        Dien_thoai varchar(20),
        Email varchar(100)
)
```

Bảng (table) – thao tác với bảng

- Tạo bảng – CREATE TABLE
  - Thiết lập khóa chính (primary key) và dữ liệu duy nhất (unique)
    - Ví dụ: tạo bảng hang_sua như trên với cột Ten_hang_sua và email là duy nhất

```
CREATE TABLE hang_sua
(
        Ma_hang_sua varchar(20) NOT NULL PRIMARY KEY,
        Ten_hang_sua varchar(100) NOT NULL,
        Dia_chi varchar(200),
        Dien_thoai varchar(20),
        Email varchar(100),
        UNIQUE(Ten_hang_sua, Email)
)
```

Bảng (table) – thao tác với bảng

- Tạo bảng – CREATE TABLE
  - Xác định cột tự tăng giá trị Auto_Increment
    - Tạo ra cột có giá trị tự động tăng dần => auto_increment
    - Thuộc tính auto_increment chỉ có thể thiết lập cho cột có kiểu dữ liệu là kiểu số nguyên.
    - Trong một bảng, chỉ có thể có một cột có thuộc tính auto_increment, cột này phải là khóa và không thiết lập giá trị mặc định DEFAULT.

Bảng (table) – thao tác với bảng

- Tạo bảng – CREATE TABLE
  - Xác định cột tự tăng giá trị Auto_Increment
    - Ví dụ: tạo bảng loại (loai) trong đó cột Ma_loai kiểu int là khóa chính và có thuộc tính auto_increment

```
CREATE TABLE loai
(
        Ma_loai int NOT NULL AUTO_INCREMENT PRIMARY KEY,
        Ten_loai VARCHAR(30) NOT NULL
)
```

Bảng (table) – thao tác với bảng

- Thay đổi cấu trúc bảng
  - Có thể thay đổi lại cấu trúc bảng mà không làm mất đi dữ liệu đã có trong bảng bằng cách sử dụng câu lệnh ALTER TABLE
    - Thêm cột mới trong bảng
    - Sửa đổi kiểu dữ liệu của cột

Bảng (table) – thao tác với bảng

- Thay đổi cấu trúc bảng
  - Thêm cột mới trong bảng
    - Cú pháp:

```
ALTER TABLE Tên_bảng
  ADD
        Tên_cột kiểu_dữ_liệu[(kích_cỡ)] [...]
```

    - Chú ý: Tên cột mới thêm vào phải khác với tên các cột đã có trong bảng

Bảng (table) – thao tác với bảng

- Thay đổi cấu trúc bảng
  - Thêm cột mới trong bảng
    - Ví dụ: thêm cột mo_ta có kiểu varchar(255) cho bảng loai đã tạo ở trên

```
ALTER TABLE loai
        ADD mo_ta varchar(255)
```

Bảng (table) – thao tác với bảng

- Thay đổi cấu trúc bảng
  - Thêm cột mới trong bảng bằng giao diện đồ họa
    - **Bước 1: Chọn Structure để hiển thị cấu trúc của bảng cần thêm cột**

Bảng (table) – thao tác với bảng

- Thay đổi cấu trúc bảng
  - Thêm cột mới trong bảng bằng giao diện đồ họa
    - **Bước 2: nhập số cột muốn thêm vào ở ô Add và chọn vị trí cần thêm cột đó trong bảng là:**
      - At End of Table: thêm vào cuối bảng
      - At Beginning of Table: thêm vào đầu bảng
      - After <tên_cột>: thêm vào sau tên_cột được chọn

      sau đó nhấn Go

Bảng (table) – thao tác với bảng

- Thay đổi cấu trúc bảng
  - Thêm cột mới trong bảng bằng giao diện đồ họa
    - **Bước 3: điền các thông tin cho cột mới như tên cột (Field), kiểu dữ liệu (Type),… rồi nhấn Save để hoàn thành việc thêm cột mới vào bảng**

Bảng (table) – thao tác với bảng

- Thay đổi cấu trúc bảng
  - Sửa đổi kiểu dữ liệu của cột
    - Cú pháp

ALTER TABLE Tên_bảng

CHANGE *tên_cột_cũ* tên_cột_mới kiểu_dữ_liệu_mới [kích_cỡ]

- Chú ý: câu lệnh này có thể dùng để thay đổi tên của cột thành một tên mới

Bảng (table) – thao tác với bảng

- Thay đổi cấu trúc bảng
  - Sửa đổi kiểu dữ liệu của cột
    - Ví dụ: cột mo_ta của bảng loai có kiểu varchar(255) không đủ để lưu trữ thông tin, ta thay đổi kiểu của cột này thành text

```
alter table loai
change mo_ta mo_ta text
```

Bảng (table) – thao tác với bảng

- Thay đổi cấu trúc bảng
  - Sửa đổi kiểu dữ liệu của cột dùng giao diện đồ họa
    - **Bước 1: Chọn cột muốn sửa đổi dữ liệu trong bảng, sau đó nhấn chọn biểu tượng trên cột**

Bảng (table) – thao tác với bảng

- Thay đổi cấu trúc bảng
  - Sửa đổi kiểu dữ liệu của cột dùng giao diện đồ họa
    - **Bước 2: Chọn kiểu dữ liệu muốn sửa đổi, nhấn Save để hoàn thành việc sửa đổi**

Field	Type	Length/Values[1]	Collation	Attributes	Null
mo_ta	TEXT		latin1_swedish_ci		not null

Save

Bảng (table) – thao tác với bảng

- Hủy cột trong bảng
  - Khi không cần sử dụng cột trong bảng chúng ta có thể dùng cú pháp ALTER TABLE để hủy bỏ cột.
  - Cú pháp:

```
ALTER TABLE Tên_bảng
  DROP COLUMN Tên_cột, ...
```

Bảng (table) – thao tác với bảng

- Hủy cột trong bảng
  - Ví dụ: hãy hủy bỏ cột mo_ta trong bảng loai đã tạo ở trên

```
ALTER TABLE loai
        DROP COLUMN mo_ta
```

Bảng (table) – thao tác với bảng

- Hủy cột trong bảng dùng giao diện đồ họa
  - Bước 1: nhấn chọn các cột cần xóa, sau đó chọn biểu tượng

  - Bước 2: Xác nhận việc xóa cột bằng cách nhấn Yes để hoàn thành. Nếu không muốn xóa thì có thể nhấn No

Bảng (table) – thao tác với bảng

- Xóa bảng
  - Cú pháp:

    DROP TABLE Tên_bảng
  - Ví dụ: xóa bảng loai đã tạo

```
DROP TABLE loai
```

3. Import và export dữ liệu

- Import dữ liệu
- Export dữ liệu

Import dữ liệu

- Nhập dữ liệu từ bên ngoài vào Database trong MySQL

Import dữ liệu

- Bước 1: Vào Database muốn import dữ liệu

- Bước 2: Chọn chức năng import

Import dữ liệu

- Bước 3: chọn tập tin chứa dữ liệu cần import ở ô Location of the text file

- Bước 4: xác nhận (nhấn Go) để hoàn thành việc import dữ liệu

Export dữ liệu

- Xuất dữ liệu từ Database trong MySQL ra tập tin thuộc một trong các dạng sau:

Export dữ liệu

Structure | SQL | Search | Query | Export | Import | Operations | Privileges | Drop

View dump (schema) of database

Export

bang_tam
ct_hoadon
ct_hoadon_07_2007
hang_sua
hoa_don
khach_hang

Select All / Unselect All

- SQL
- LaTeX
- Microsoft Excel 2000
- Microsoft Word 2000
- CSV for MS Excel
- CSV
- XML

SQL options

Add custom comment into header (\n splits lines):

- [ ] Enclose export in a transaction
- [ ] Disable foreign key checks

SQL export compatibility: NONE

Structure

- [ ] Add DROP TABLE
- [ ] Add IF NOT EXISTS
- [x] Add AUTO_INCREMENT value
- [x] Enclose table and field names with backquotes

Add into comments:

- [ ] Creation/Update/Check dates
- [ ] Relations
- [ ] MIME type

Data

- [ ] Complete inserts
- [ ] Extended inserts

Maximal length of created query: 50000

- [ ] Use delayed inserts
- [ ] Use ignore inserts
- [x] Use hexadecimal for binary fields

Export type: INSERT

Save as file

File name template (1): __DB__ ([x] remember template)

Compression: None "zipped" "gzipped"

Go

Export dữ liệu

- Bước 1: Chọn Database muốn export dữ liệu

- Bước 2: Chọn chức năng export

Export dữ liệu

- Bước 3: Chọn các bảng trong Database cần export

- Bước 4: Chọn loại file để export

Export dữ liệu

- Bước 5: Thiết lập các thuộc tính cho file export
- Bước 6: Đặt tên cho file export (nhấn Go để qua phần chọn nơi lưu trữ)

Export dữ liệu

- Bước 7: Chọn nơi lưu trữ file export

Export dữ liệu

- Bước 8: Click “Save” để hoàn thành việc export dữ liệu

4. Toán tử

- Khái niệm
- Toán tử số học
- Toán tử so sánh
- Toán tử luận lý (logic)

Khái niệm

- MySQL cung cấp các toán tử như: toán tử số học, toán tử so sánh, toán tử luận lý (logic).
- Các toán tử này được kết hợp vào bên trong các mệnh đề WHERE, HAVING, IF, CASE, …

Toán tử số học

- Dùng để thực hiện các phép tính: cộng, trừ, nhân, chia, chia lấy phần dư với giá trị được đem tính toán là kiểu số.
- Khi có nhiều phép tính thì đưa từng biểu thức tính toán vào trong ngoặc đơn () để việc tính toán được tường minh.

Toán tử số học

- Bảng các toán tử số học

Toán tử	Mô tả
+	Thực hiện phép cộng hai số
-	Thực hiện phép trừ hai số
*	Thực hiện phép nhân hai số
/	Thực hiện phép chia hai số
%	Thực hiện phép chia lấy phần dư

Toán tử số học

- Ví dụ:

```
SELECT 10+3*5-6/2 → 22
SELECT 10+(3*5)-(6/2) → 22
SELECT 107%4 → 3
```

Toán tử so sánh

- Dùng để thực hiện các phép so sánh như: bằng, lớn hơn, nhỏ hơn, khác,… cho các biểu thức cần so sánh. Kết quả trả về của phép so sánh là TRUE (đúng) hoặc FALSE (sai)
- Toán tử so sánh được sử dụng cho nhiều kiểu dữ liệu khác nhau như kiểu số, kiểu chuỗi,…

Toán tử so sánh

- Bảng các toán tử so sánh

Toán tử	Mô tả
=	Thực hiện phép so sánh bằng khi hai giá trị (hoặc biểu thức) khác NULL
<=>	Thực hiện phép so sánh bằng ngay cả khi hai giá trị (hoặc biểu thức) đem so sánh đều là NULL
<> , !=	Thực hiện phép so sánh khác
<	Thực hiện phép so sánh nhỏ hơn
<=	Thực hiện phép so sánh nhỏ hơn hoặc bằng
>	Thực hiện phép so sánh lớn hơn
>=	Thực hiện phép so sánh lớn hơn hoặc bằng

Toán tử so sánh

- Ví dụ:

```
SELECT 1 = 0  → 0
SELECT '0.0' = 0  → 1
SELECT 1 <=> 1, NULL <=> NULL, 1 <=> NULL
→ 1    1    0
SELECT 'zapp' <> 'zappp' → 1
SELECT 2 >= 2 → 1
```

Toán tử luận lý (logic)

- Dùng để kết hợp các biểu thức so sánh đơn lẻ thành một biểu thức chung
- Bảng các toán tử luận lý

Toán tử	Mô tả
AND, &&	Khi tất cả các biểu thức đều có kết quả đúng thì biểu thức chung sẽ có kết quả đúng
OR, ||	Chỉ cần một biểu thức có kết quả đúng thì biểu thức chung sẽ có kết quả đúng
XOR	Khi một trong hai biểu thức có kết quả đúng thì biểu thức chung sẽ có kết quả đúng (nhưng khi cả hai biểu thức đều đúng thì biểu thức chung sẽ có kết quả sai)
NOT, !	Khi biểu thức có kết quả sai thì NOT(biểu thức) sẽ có kết quả đúng và ngược lại

Toán tử luận lý (logic)

- Ví dụ:

```
SELECT (1>2) && (1<3) → 0
SELECT (9+1=10) AND (10=10) → 1
SELECT 0 || 0 → 0
SELECT (3>2) OR (5>9) → 1
SELECT 1 XOR 0 → 1
SELECT 1 XOR 1 → 0
SELECT NOT 0 → 1
SELECT !(1+1) → 0
```

5. Phát biểu SQL

- Câu lệnh SELECT
- Truy xuất dữ liệu từ nhiều bảng
- Truy vấn con
- Câu lệnh thêm dữ liệu (INSERT INTO)
- Câu lệnh cập nhật dữ liệu (UPDATE)
- Câu lệnh xóa dữ liệu (DELETE)
- Sử dụng mệnh đề UNION trong truy vấn
- Sử dụng hàm trong SQL

Câu lệnh SELECT

- Truy vấn đơn giản select … from
- Truy vấn có sắp xếp dữ liệu
- Truy vấn có điều kiện where
- Nhóm dữ liệu group by
- Điều kiện lọc nhóm having
- Giới hạn mẩu tin limit

Câu lệnh SELECT

- Truy vấn đơn giản select … from
  - Chọn ra dữ liệu của các cột có trong một bảng
  - Cú pháp:

    SELECT Danh_sách_các_cột

    FROM Tên_bảng
  - Với tên các cột trong danh_sách_các_cột phải chính xác

Câu lệnh SELECT

- Truy vấn đơn giản select … from
  - Ví dụ: hiển thị thông tin của các hãng sữa có trong bảng hang_sua gồm có các cột như sau: mã hãng sữa, tên hãng sữa, điện thoại

```
SELECT Ma_hang_sua,
Ten_hang_sua, Dien_thoai
FROM hang_sua
```

Ma_hang_sua	Ten_hang_sua	Dien_thoai
VNM	Vinamilk	8794561
NTF	Nutifood	7895632
AB	Abbott	8741258
DS	Daisy	5789321
DL	Dutch Lady	7826451
DM	Dumex	6258943
MJ	Mead Jonhson	8741258

Câu lệnh SELECT

- Truy vấn đơn giản select … from
  - Dùng * khi muốn lấy dữ liệu từ tất cả các cột trong bảng
  - Ví dụ: hiển thị thông tin hóa đơn gồm toàn bộ các cột trong bảng hóa đơn

```
SELECT *
FROM HOA_DON
```

Ma_hoa_don	Ngay_HD	Ma_khach_hang	Tri_gia
D001	2007-07-31	kh001	1480000
D002	2007-07-30	kh002	1656000
D003	2007-08-10	kh003	2484500
D004	2007-08-11	kh002	5573000

Câu lệnh SELECT

- Truy vấn có sắp xếp dữ liệu
  - Câu lệnh SELECT … FROM kết hợp với mệnh đề ORDER BY giúp lấy dữ liệu của các cột bên trong bảng đồng thời sắp xếp lại dữ liệu theo thứ tự tăng dần hoặc giảm dần.
  - Cú pháp:

    SELECT Danh_sách_các_cột

    FROM Tên_bảng

    ORDER BY Tên_cột_sắp_xếp [DESC, …]
  - Mặc định, cột sẽ được sắp xếp tăng dần, nếu muốn sắp xếp cột theo thứ tự giảm dần thì phía sau tên cột sắp xếp cần thêm từ khóa DESC

Câu lệnh SELECT

- Truy vấn có sắp xếp dữ liệu
  - Ví dụ: hiển thị thông tin các hãng sữa trong bảng hang_sua gồm có: mã hãng sữa, tên hãng sữa, email có sắp xếp dữ liệu theo cột tên hãng sữa tăng dần

```
SELECT Ma_hang_sua,
Ten_hang_sua, Email
FROM hang_sua
ORDER BY Ten_hang_sua
```

Ma_hang_sua	Ten_hang_sua	Email
AB	Abbott	abbott@ab.com
DS	Daisy	daisy@ds.com
DM	Dumex	dumex@dm.com
DL	Dutch Lady	dutchlady@dl.com
MJ	Mead Jonhson	meadjohn@mj.com
NTF	Nutifood	nutifood@ntf.com
VNM	Vinamilk	vinamilk@vnm.com

Câu lệnh SELECT

- Truy vấn có sắp xếp dữ liệu
  - Ví dụ: hiển thị thông tin các hãng sữa trong bảng hang_sua gồm có: mã hãng sữa, tên hãng sữa, email có sắp xếp dữ liệu theo cột tên hãng sữa giảm dần

```
SELECT Ma_hang_sua,
Ten_hang_sua, Email
FROM hang_sua
ORDER BY
Ten_hang_sua DESC
```

Ma_hang_sua	Ten_hang_sua	Email
VNM	Vinamilk	vinamilk@vnm.com
NTF	Nutifood	nutifood@ntf.com
MJ	Mead Jonhson	meadjohn@mj.com
DL	Dutch Lady	dutchlady@dl.com
DM	Dumex	dumex@dm.com
DS	Daisy	daisy@ds.com
AB	Abbott	abbott@ab.com

Câu lệnh SELECT

- Truy vấn có điều kiện where
  - Câu lệnh SELECT … FROM kết hợp với mệnh đề WHERE giúp lọc các dòng dữ liệu bên trong bảng, dữ liệu này phải thỏa điều kiện đưa ra trong mệnh đề WHERE
  - Cú pháp:

```
SELECT Danh_sách_các_cột
FROM Tên_bảng
WHERE Điều_kiện_lọc
[ORDER BY Tên_cột_sắp_xếp [DESC, …]]
```

Câu lệnh SELECT

- Truy vấn có điều kiện where
  - Các phép toán thường gặp trong điều kiện lọc
    - So sánh: >, >=, <, <=, =, !=, <>
    - Logic: and, or, not, in, not in, between, like, not like

Câu lệnh SELECT

- Truy vấn có điều kiện where
  - Ví dụ: hiển thị thông tin của các sữa trong bảng sua gồm: mã sữa, tên sữa, mã hãng sữa, đơn giá sao cho chỉ chọn ra các sữa có mã hãng sữa là “AB”

```
SELECT Ma_sua, Ten_sua,
Ma_hang_sua, Don_gia
FROM sua
WHERE Ma_hang_sua = "AB"
```

Ma_sua	Ten_sua	Ma_hang_sua	Don_gia
AB0001	Gain Advance	AB	107000
AB0002	Gain IQ	AB	107000
AB0003	Abbott Grow	AB	87000
AB0004	Abbott Grow School	AB	87000
AB0005	Abbott Pedia Sure	AB	146000
AB0006	Similac Neo Sure	AB	145000

Câu lệnh SELECT

- Truy vấn có điều kiện where
  - Ví dụ: hiển thị các thông tin trong bảng sữa (sua) gồm mã sữa, tên sữa, trọng lượng sao cho chỉ chọn ra các sữa với tên sữa có từ “Dielac” hoặc tên sữa có từ “Enfa” và trọng lượng bằng 400gr, 500gr hoặc 900gr

```
SELECT Ma_sua, Ten_sua, Trong_luong
FROM sua
WHERE (Ten_sua LIKE "%Dielac%" OR Ten_sua LIKE "%Enfa%") AND Trong_luong IN(400, 500, 900)
```

Ma_sua	Ten_sua	Trong_luong
MJ0001	Enfa Mama A+	900
MJ0002	EnfaLac	400
MJ0003	EnfaGrow	400
MJ0004	EnfaPro	900
MJ0005	EnfaPro A+	900
VNM006	Dielac Alpha	900
VNM009	Dielac Canxi Premier 2000	400
VNM011	Dielac Sure	400
VNM012	Dielac Mamma	900

Câu lệnh SELECT

- Truy vấn có điều kiện where
  - Có thể sử dụng mệnh đề WHERE để liên kết dữ liệu của nhiều bảng trong truy vấn
  - Cú pháp:

```
SELECT Danh_sách_các_cột
FROM Tên_bảng_1, Tên_bảng_2, ...
WHERE Tên_bảng_1.tên_cột =
  Tên_bảng_2.tên_cột
[ORDER BY Tên_cột_sắp_xếp [DESC, …]]
```

Câu lệnh SELECT

- Truy vấn có điều kiện where
  - Ví dụ: hiển thị thông tin các sữa trong bảng sữa (sua) gồm mã sữa, tên sữa, tên hãng sữa sao cho chỉ chọn ra các sữa có mã hãng sữa khác “DL” và “DS” và “VNM”

```
SELECT Ma_sua, Ten_sua,
Ten_hang_sua
FROM hang_sua, sua
WHERE
hang_sua.Ma_hang_sua =
sua.Ma_hang_sua AND
sua.Ma_hang_sua NOT
IN("DL", "DS", "VNM")
```

Ma_sua	Ten_sua	Ten_hang_sua
AB0001	Gain Advance	Abbott
AB0002	Gain IQ	Abbott
AB0003	Abbott Grow	Abbott
AB0004	Abbott Grow School	Abbott
AB0005	Abbott Pedia Sure	Abbott
AB0006	Similac Neo Sure	Abbott
MJ0001	Enfa Mama A+	Mead Jonhson
MJ0002	EnfaLac	Mead Jonhson
MJ0003	EnfaGrow	Mead Jonhson
MJ0004	EnfaPro	Mead Jonhson
MJ0005	EnfaPro A+	Mead Jonhson
NTF001	Nuti Mum	Nutifood
NTF002	Obilac	Nutifood
NTF003	Nuti IQ	Nutifood
NTF004	Nuti Vita	Nutifood
NTF005	Sữa nguyên kem Nuti	Nutifood
NTF006	Nuti Vita	Nutifood
NTF007	DiabetCare	Nutifood

Câu lệnh SELECT

- Nhóm dữ liệu group by
  - Câu lệnh SELECT … FROM kết hợp với mệnh đề GROUP BY giúp nhóm dữ liệu của các dòng dữ liệu bên trong bảng và sử dụng thêm các hàm thống kê đi kèm để tính toán dữ liệu có tính chất thống kê
  - Cú pháp:

```
SELECT Danh_sách_các_cột, Hàm_thống_kê [as
  tên]
FROM Tên_bảng
[WHERE Điều_kiện_lọc]
GROUP BY Danh_sách_các_cột_nhóm_dữ_liệu
[ORDER BY Tên_cột_sắp_xếp [DESC, …]]
```

Câu lệnh SELECT

- Nhóm dữ liệu group by
  - Các hàm thống kê:
    - AVG: hàm trả về giá trị trung bình theo nhóm trong câu lệnh truy vấn trên bảng
    - COUNT: hàm trả về số lượng mẩu tin theo nhóm trong câu truy vấn trên bảng
    - MIN: hàm trả về giá trị nhỏ nhất theo nhóm
    - MAX: hàm trả về giá trị lớn nhất theo nhóm
    - SUM: hàm trả về tổng các giá trị theo nhóm

Câu lệnh SELECT

- Nhóm dữ liệu group by
  - Ví dụ: thống kê số lượng sữa của mỗi hãng sữa, sắp xếp dữ liệu tăng dần theo số lượng sữa

```
SELECT Hang_sua.Ma_hang_sua,
Ten_hang_sua, COUNT(sua.Ma_sua)
AS So_luong_sua
FROM hang_sua, sua
WHERE hang_sua.Ma_hang_sua =
sua.Ma_hang_sua
GROUP BY Ma_hang_sua,
Ten_hang_sua
ORDER BY So_luong_sua
```

Ma_hang_sua	Ten_hang_sua	So_luong_sua
DS	Daisy	2
MJ	Mead Jonhson	5
AB	Abbott	6
NTF	Nutifood	7
DL	Dutch Lady	11
VNM	Vinamilk	12

Câu lệnh SELECT

- Nhóm dữ liệu group by
  - Ví dụ: căn cứ vào bảng chi tiết hóa đơn để thống kê tổng số tiền của mỗi hóa đơn, sắp xếp dữ liệu tăng dần theo tổng số tiền

```
SELECT So_hoa_don,
Sum(So_luong*Don_gia) as
Tong_so_tien
FROM ct_hoadon
GROUP BY (So_hoa_don)
ORDER BY Sum(So_luong*Don_gia)
```

So_hoa_don	Tong_so_tien
D001	1480000
D002	1656000
D003	2484500
D004	5573000

Câu lệnh SELECT

- Điều kiện lọc nhóm having
  - Câu lệnh SELECT … FROM kết hợp với mệnh đề HAVING giúp lọc lại dữ liệu sau khi đã gom nhóm dữ liệu bằng mệnh đề GROUP BY
  - Cú pháp:

```
SELECT Danh_sách_các_cột, Hàm_thống_kê [as
  tên]
FROM Tên_bảng
[WHERE Điều_kiện_lọc]
GROUP BY Danh_sách_các_cột_nhóm_dữ_liệu
HAVING Điều_kiện_lọc_sau_khi_nhóm
[ORDER BY Tên_cột_sắp_xếp [DESC, …]]
```

Câu lệnh SELECT

- Điều kiện lọc nhóm having
  - Ví dụ: thống kê số lượng sữa của mỗi hãng sữa, sao cho chỉ lọc ra những hãng sữa có số lượng sữa >5

```
SELECT Hang_sua.Ma_hang_sua,
Ten_hang_sua, COUNT(sua.Ma_sua)
AS So_luong_sua
FROM hang_sua, sua
WHERE hang_sua.Ma_hang_sua =
sua.Ma_hang_sua
GROUP BY Ma_hang_sua,
Ten_hang_sua
HAVING COUNT(sua.Ma_sua)>5
```

Ma_hang_sua	Ten_hang_sua	So_luong_sua
AB	Abbott	6
DL	Dutch Lady	11
NTF	Nutifood	7
VNM	Vinamilk	12

Câu lệnh SELECT

- Giới hạn mẩu tin limit
  - Câu lệnh SELECT … FROM kết hợp với mệnh đề LIMIT n,m giúp lấy ra m mẩu tin trong bảng tính từ vị trí n, theo một tiêu chuẩn sắp xếp nào đó
  - Cú pháp:

```
SELECT Danh_sách_các_cột, Hàm_thống_kê [as tên]
FROM Tên_bảng
[WHERE Điều_kiện_lọc]
[GROUP BY Danh_sách_các_cột_nhóm_dữ_liệu]
[HAVING Điều_kiện_lọc_sau_khi_nhóm]
[ORDER BY Tên_cột_sắp_xếp [DESC, …]]
LIMIT n,m
```

Câu lệnh SELECT

- Giới hạn mẩu tin limit
  - Ví dụ: cho biết 3 sữa trong bảng sữa có đơn giá lớn nhất gồm các thông tin: mã sữa, tên sữa, đơn giá, trọng lượng

```
SELECT Ma_sua, Ten_sua, Don_gia,
Trong_luong
FROM sua
ORDER BY Don_gia DESC
LIMIT 0,3
```

Ma_sua	Ten_sua	Don_gia	Trong_luong
MJ0005	EnfaPro A+	241000	900
MJ0004	EnfaPro	198000	900
MJ0001	Enfa Mama A+	196000	900

Truy vấn dữ liệu từ nhiều bảng

- Khái niệm
- Inner join
- Left join, right join
- Mệnh đề liên kết dữ liệu nhiều bảng

Truy vấn dữ liệu từ nhiều bảng

- Khái niệm
  - Khi muốn liên kết các bảng có quan hệ với nhau để lấy ra dữ liệu chung → kết hợp SELECT … FROM với mệnh đề JOIN.
  - Khi sử dụng JOIN để nối các bảng cần phải lưu ý những bảng này phải có các cột liên hệ với nhau và thứ tự quan hệ được chỉ định giữa các bảng sẽ làm ảnh hưởng tới kết quả truy vấn.

Truy vấn dữ liệu từ nhiều bảng

- Khái niệm
  - Ví dụ:
    - Trong ví dụ quản lý bán sữa, khi muốn in một hóa đơn bán sữa cho khách hàng, theo thiết kế sẽ có một số bảng liên quan đến nhau.
    - Nếu muốn biết các sản phẩm sữa được bán cho ai thì liên quan tới bảng khách hàng, khách hàng mua những sản phẩm gì thì liên quan tới bảng chi tiết hóa đơn, sữa được mua có tên là gì thì liên quan tới bảng sữa, …
    - Để tạo được các mối quan hệ này và truy xuất dữ liệu → dùng JOIN để kết nối các bảng với nhau.

Truy vấn dữ liệu từ nhiều bảng

- Inner join
  - Khi kết nối các bảng dùng INNER JOIN → chỉ định việc so sánh giá trị trong các cột của các bảng là tương đương – dữ liệu đều có ở cả hai bảng.
  - Kết quả sau khi thực hiện câu lệnh truy vấn kết hợp INNER JOIN là các mẩu tin thỏa điều kiện quan hệ ở cả hai bảng
  - Cú pháp:

```
SELECT Danh_sách_các_cột
FROM Tên_bảng
INNER JOIN Tên_bảng_liên_kết ON
  Điều_kiện_liên_kết
[WHERE Điều_kiện_lọc]
[ORDER BY Danh_sách_các_cột_sắp_xếp [DESC]]
```

Truy vấn dữ liệu từ nhiều bảng

- Inner join
  - Ví dụ: in ra thông tin chi tiết của các hóa đơn trong tháng 7 năm 2007, gồm: số hóa đơn, ngày hóa đơn, mã sữa, số lượng

```
SELECT ct_hoadon.So_hoa_don,
Ngay_hd, Ma_sua, So_luong
FROM ct_hoadon INNER JOIN
hoa_don ON
ct_hoadon.So_hoa_don =
hoa_don.So_hoa_don
WHERE MONTH(Ngay_hd) = 7 AND
YEAR(Ngay_hd)=2007
```

So_hoa_don	Ngay_hd	Ma_sua	So_luong
D001	2007-07-31	AB0001	2
D001	2007-07-31	DL0001	12
D001	2007-07-31	NTF002	8
D001	2007-07-31	VNM012	4
D002	2007-07-30	DL0001	2
D002	2007-07-30	MJ0001	5
D002	2007-07-30	MJ0004	3

Truy vấn dữ liệu từ nhiều bảng

- Left join, right join
  - Khi kết nối các bảng dùng LEFT|RIGHT JOIN → chỉ định việc so sánh giá trị trong các cột của các bảng được ưu tiên cho mối quan hệ bên nhánh trái | nhánh phải. Việc thay đổi thứ tự ưu tiên này sẽ làm ảnh hưởng tới kết quả truy vấn.
  - Cú pháp:

```
SELECT Danh_sách_các_cột
FROM Tên_bảng
LEFT|RIGHT JOIN Tên_bảng_liên_kết
ON Điều_kiện_liên_kết
[WHERE Điều_kiện_lọc]
[ORDER BY Danh_sách_các_cột_sắp_xếp [DESC]]
```

Truy vấn dữ liệu từ nhiều bảng

- Left join, right join
  - Ví dụ: hiển thị danh sách tất cả các khách hàng hiện có trong bảng khách hàng và hóa đơn mua hàng trong bảng hóa đơn gồm: mã khách hàng, số hóa đơn, ngày hóa đơn.

```
SELECT
khach_hang.Ma_khach_hang,
So_hoa_don, Ngay_hd
FROM khach_hang LEFT JOIN
hoa_don ON
khach_hang.Ma_khach_hang =
hoa_don.Ma_khach_hang
```

Ma_khach_hang	So_hoa_don	Ngay_hd
kh001	D001	2007-07-31
kh002	D002	2007-07-30
kh002	D004	2007-08-11
kh003	D003	2007-08-10
kh004	NULL	NULL
kh005	NULL	NULL
kh006	NULL	NULL
kh008	NULL	NULL

Truy vấn dữ liệu từ nhiều bảng

- Mệnh đề liên kết dữ liệu nhiều bảng
  - Cú pháp:

```
SELECT Danh_sách_các_cột
FROM Tên_bảng_1
INNER|LEFT|RIGHT JOIN Tên_bảng_2
ON Điều_kiện_liên_kết_bang_1_2
INNER|LEFT|RIGHT JOIN Tên_bảng_3
ON Điều_kiện_liên_kết...
...
[WHERE Điều_kiện_lọc]
[ORDER BY Danh_sách_các_cột_sắp_xếp [DESC]]
```

Truy vấn dữ liệu từ nhiều bảng

- Mệnh đề liên kết dữ liệu nhiều bảng
  - Ví dụ: hiển thị thông tin chi tiết hóa đơn gồm có ngày hóa đơn, số hóa đơn, tên sữa và số lượng

```
SELECT Ngay_hd, ct_hoadon.So_hoa_don,
Sua.Ten_sua, So_luong
FROM ct_hoadon
INNER JOIN sua ON ct_hoadon.Ma_sua =
sua.Ma_sua
INNER JOIN hoa_don ON ct_hoadon.So_hoa_don =
hoa_don.So_hoa_don
```

2007-08-10	D003	Sữa đặc Trường Sinh	13
2007-08-11	D004	Gain Advance	15
2007-08-11	D004	Gain IQ	25
2007-08-11	D004	Nuti Mum	10
2007-08-11	D004	Dielac Mamma	8

Truy vấn con

- Truy vấn con trả về một giá trị
- Truy vấn con trả về danh sách các giá trị
- Làm việc với toán tử so sánh
- Làm việc với toán tử truy vấn con

Truy vấn con

- Khái niệm
  - Truy vấn con là một câu lệnh truy vấn SELECT được lồng vào các câu lệnh truy vấn khác nhằm thực hiện các truy vấn tính toán phức tạp.
  - **Lưu ý:** khi dùng truy vấn con cần tuân theo các quy tắc
    - Truy vấn con phải được đặt trong cặp ngoặc đơn ()
    - Truy vấn con chỉ có thể tham chiếu đến 1 cột hoặc 1 biểu thức.
  - Kết quả trả về của truy vấn con có thể là một giá trị hoặc một danh sách các giá trị.

Truy vấn con

- Truy vấn con trả về một giá trị
  - Là truy vấn mà kết quả trả về của nó là một giá trị duy nhất
  - Ví dụ: hãy cho biết sản phẩm sữa nào có trọng lượng lớn nhất.
    - Tìm trọng lượng lớn nhất

```
SELECT MAX(Trong_luong)
FROM sua
```

MAX(Trong_luong)
900

Truy vấn con

- Truy vấn con trả về một giá trị
  - Ví dụ: hãy cho biết sản phẩm sữa nào có trọng lượng lớn nhất.
    - Lọc ra các sữa có trọng lượng bằng với trọng lượng lớn nhất này

```
SELECT Ma_sua, Ten_sua,
Trong_luong
FROM sua
WHERE Trong_luong = (SELECT
MAX(Trong_luong) FROM sua)
```

Ma_sua	Ten_sua	Trong_luong
DL0002	Canximex	900
DS0001	Daisy Không Đường	900
MJ0001	Enfa Mama A+	900
MJ0004	EnfaPro	900
MJ0005	EnfaPro A+	900
VNM006	Dielac Alpha	900
VNM012	Dielac Mamma	900

Truy vấn con

- Truy vấn con trả về danh sách các giá trị
  - Là truy vấn con mà kết quả trả về là tập hợp các giá trị.
  - Toán tử IN hoặc NOT IN thường được dùng trong trường hợp này vì nó so sánh một phần tử có thuộc (hay không thuộc) tập hợp các giá trị hay không.

Truy vấn con

- Truy vấn con trả về danh sách các giá trị
  - Ví dụ: hãy cho biết các khách hàng nào chưa mua hàng (chưa có thông tin khách hàng trong bảng hóa đơn)
    - Tìm các khách hàng đã mua hàng

```
SELECT Ma_khach_hang
FROM hoa_don
```

Ma_khach_hang
kh001
kh002
kh003
kh002

Truy vấn con

- Truy vấn con trả về danh sách các giá trị
  - Ví dụ: hãy cho biết các khách hàng nào chưa mua hàng (chưa có thông tin khách hàng trong bảng hóa đơn)
    - Lọc ra những khách hàng chưa mua hàng

```
SELECT Ma_khach_hang
FROM khach_hang
WHERE Ma_khach_hang NOT IN
(SELECT Ma_khach_hang
FROM hoa_don)
```

Ma_khach_hang
kh004
kh005
kh006
kh008

Truy vấn con

- Làm việc với toán tử so sánh
  - Các toán tử so sánh thường được sử dụng trong truy vấn con có thể là: >, >=, <, <=, =, <>.
  - **Lưu ý:** Thông thường các toán tử so sánh được sử dụng khi dùng truy vấn con trả về một giá trị

Truy vấn con

- Làm việc với toán tử truy vấn con
  - Các toán tử truy vấn con thường hay sử dụng là: ANY, SOME, ALL, IN, NOT IN, EXISTS, NOT EXISTS.
  - Lưu ý: thông thường các toán tử truy vấn con được sử dụng khi dùng truy vấn con trả về tập hợp các giá trị.
  - Quy tắc:

    IN ⇔ =ANY

    NOT IN ⇔ <>ALL

Truy vấn con

- Làm việc với toán tử truy vấn con
  - Ví dụ: viết lại câu lệnh truy vấn cho biết các khách hàng chưa mua hàng

```
SELECT Ma_khach_hang
FROM khach_hang
WHERE Ma_khach_hang <>ALL
(SELECT Ma_khach_hang
FROM hoa_don)
```

Ma_khach_hang
kh004
kh005
kh006
kh008

Truy vấn con

- Làm việc với toán tử truy vấn con
  - Sử dụng từ khóa EXISTS hoặc NOT EXISTS để kiểm tra tính tồn tại (hay không tồn tại) của dữ liệu.
  - Sau EXISTS hoặc NOT EXISTS là câu lệnh truy vấn con mà kết quả trả về là một tập hợp trống hoặc có phần tử

Truy vấn con

- Làm việc với toán tử truy vấn con
  - Ví dụ: tìm những khách hàng chưa mua hàng theo cách dùng NOT EXISTS

```
SELECT Ma_khach_hang
FROM khach_hang
WHERE NOT EXISTS (SELECT *
FROM hoa_don
WHERE khach_hang.Ma_khach_hang =
hoa_don.Ma_khach_hang)
```

Ma_khach_hang
kh004
kh005
kh006
kh008

Câu lệnh thêm dữ liệu

- Câu lệnh INSERT INTO được dùng để thêm mới một hay nhiều dòng dữ liệu vào bên trong một bảng.
- Có thể thêm vào bảng:
  - Giá trị trực tiếp
  - Dữ liệu lấy từ các bảng khác

Câu lệnh thêm dữ liệu

- Giá trị trực tiếp
  - Cú pháp:

INSERT INTO Tên_bảng [(Danh sách các cột trong bảng)]

VALUES (Danh_sách_các_giá_trị)

Câu lệnh thêm dữ liệu

- Giá trị trực tiếp
  - Ví dụ: thêm mới một khách hàng có các thông tin sau vào bảng khach_hang:
    - Mã khách hàng: kh007
    - Họ tên: Phan Nam
    - Phái: 0 (nam)
    - Địa chỉ: 12A – Pasteur – Q.1 – TP. HCM
    - Điện thoại: 8497625
    - Email: phannam@yahoo.com

Câu lệnh thêm dữ liệu

- Giá trị trực tiếp
  - Cách 1:

```
INSERT INTO khach_hang(Ma_khach_hang,
Ten_khach_hang, Phai, Dia_chi, Dien_thoai, Email)
VALUES("kh001", "Phan Nam", 0, "12A – Pasteur – Q.1 – TP.
HCM", "8497625", "phannam@yahoo.com")
```

```
INSERT INTO khach_hang
VALUES("kh001", "Phan Nam", 0, "12A – Pasteur – Q.1 – TP.
HCM", "8497625", "phannam@yahoo.com")
```

Câu lệnh thêm dữ liệu

- Lấy dữ liệu từ các bảng khác
  - Bằng cách kết hợp giữa INSERT và SELECT
  - Cú pháp:

```
INSERT INTO Tên_bảng
[(Danh_sách_các_cột_cần_thêm_dữ_liệu)]
SELECT Danh_sách_các_cột_lấy_dữ_liệu
FROM Tên_bảng_nguồn
WHERE Điều_kiện_lọc_dữ_liệu
```

  - Danh_sách_các_cột_cần_thêm_dữ_liệu và Danh_sách_các_cột_ lấy_dữ_liệu phải tương ứng với nhau

Câu lệnh thêm dữ liệu

- Lấy dữ liệu từ các bảng khác
  - Ví dụ: thêm vào bảng ct_hoadon_07_2007 (bảng chưa có dữ liệu) các chi tiết hóa đơn bán sữa của tháng 07/2007

```
INSERT INTO ct_hoadon_07_2007
SELECT ct_hoadon.So_hoa_don, Ma_sua, So_luong,
Don_gia
FROM ct_hoadon, hoa_don
WHERE ct_hoadon.So_hoa_don = hoa_don.So_hoa_don
AND month(Ngay_HD) = 7 AND year(Ngay_hd)=2007
```

Thêm dữ liệu

- Sử dụng giao diện đồ họa
  - Bước 1: Chọn bảng muốn thêm dữ liệu, sau đó nhấn chọn

để vào màn hình thêm mới.

Thêm dữ liệu

- Sử dụng giao diện đồ họa
  - Bước 2: Trong màn hình cần thêm mới, lần lượt nhập thông tin cho từng mẩu tin (mặc định ban đầu là thêm vào 2 mẩu tin).
  - Bước 3: Khi hoàn thành việc nhập dữ liệu cho các mẩu tin chọn chức năng:

Thêm dữ liệu

- Sử dụng giao diện đồ họa
  - Bước 4: Chọn quay về trang trước:

  - Hay tiếp tục thêm các mẩu tin khác

  - Bước 5: Nhấn Go để hoàn thành việc thêm mới mẩu tin

Câu lệnh cập nhật dữ liệu

- Giá trị trực tiếp
- Lấy dữ liệu từ các bảng khác

Câu lệnh cập nhật dữ liệu

- Giá trị trực tiếp
  - Dùng câu lệnh UPDATE để cập nhật giá trị trực tiếp hay một biểu thức có giá trị trả về cho mẫu tin trong bảng
  - Cú pháp:

```
UPDATE Tên_bảng
SET Tên_cột = Giá_trị (hoặc biểu thức) [,...]
WHERE Điều_kiện_cập_nhật
```

Câu lệnh cập nhật dữ liệu

- Giá trị trực tiếp
  - Ví dụ: cập nhật lại đơn giá cho sữa có tên là ‘Abbott Grow’ với đơn giá mới là 95000vnđ

Ten_sua	Don_gia
Abbott Grow	87000

Đơn giá cũ

```
UPDATE sua
SET Don_gia = 95000
WHERE Ten_sua = "Abbott Grow"
```

Đơn giá mới

Ten_sua	Don_gia
Abbott Grow	95000

Câu lệnh cập nhật dữ liệu

- Giá trị trực tiếp
  - Ví dụ: cập nhật lại đơn giá cho các sữa có mã hãng sữa là “AB” với đơn giá mới = đơn giá cũ + 5%

Ten_sua	Don_gia	Ma_hang_sua
Gain Advance	107000	AB
Gain IQ	107000	AB
Abbott Grow	87000	AB
Abbott Grow School	87000	AB
Abbott Pedia Sure	146000	AB
Similac Neo Sure	145000	AB

Đơn giá cũ

```
UPDATE sua
SET Don_gia = Don_gia + Don_gia *
0.05
WHERE Ma_hang_sua = "AB"
```

Đơn giá mới

Ten_sua	Don_gia	Ma_hang_sua
Gain Advance	112350	AB
Gain IQ	112350	AB
Abbott Grow	91350	AB
Abbott Grow School	91350	AB
Abbott Pedia Sure	153300	AB
Similac Neo Sure	152250	AB

Câu lệnh cập nhật dữ liệu

- Lấy dữ liệu từ các bảng khác
  - Kết hợp giữa UPDATE và SELECT để lấy dữ liệu từ bảng khác cập nhật vào bảng
  - Cú pháp:

```
UPDATE Tên_bảng
SET Tên_cột = (SELECT ... FROM ... WHERE ...)
WHERE Điều_kiện_cập_nhật
```

Câu lệnh cập nhật dữ liệu

- Lấy dữ liệu từ các bảng khác
  - Ví dụ: cột đơn giá trong bảng ct_hoadon chưa có giá trị, hãy cập nhật đơn giá cho cột này với giá trị được lấy từ cột đơn giá của bảng sữa.

```
UPDATE ct_hoadon
SET Don_gia = (SELECT Don_gia
FROM sua WHERE ct_hoadon.Ma_sua
= sua.Ma_sua)
```

Ma_hoa_don	Ma_sua	So_luong	Don_gia
D001	VNM012	4	103500
D001	AB0001	2	107000
D002	DL0001	2	41000
D002	MJ0004	3	198000
D001	DL0001	12	41000
D003	AB0001	8	107000
D004	NTF001	10	46500
D004	VNM012	8	103500
D003	DL0006	13	11500
D004	AB0001	15	107000
D002	MJ0001	5	196000
D001	NTF002	8	45000
D003	AB0003	17	87000
D004	AB0002	25	107000

Cập nhật dữ liệu

- Sử dụng giao diện đồ họa
  - Bước 1: Chọn bảng cần cập nhật dữ liệu và mở bảng dưới chế độ

  - Bước 2: Khi các dòng dữ liệu hiển thị, tìm tới dòng muốn cập nhật dữ liệu và chọn biểu tượng

Cập nhật dữ liệu

- Sử dụng giao diện đồ họa
  - Màn hình cập nhật dữ liệu sẽ được hiển thị

  - Bước 3: Cập nhật giá trị mới sau đó nhấn “Go” để hoàn thành việc cập nhật.

Câu lệnh xóa dữ liệu

- Lệnh xóa dữ liệu đơn giản
- Lệnh xóa dữ liệu có điều kiện được lấy từ các bảng khác

Câu lệnh xóa dữ liệu

- Lệnh xóa dữ liệu đơn giản
  - Cú pháp:

    DELETE FROM Tên_bảng

    WHERE Điều_kiện_xóa

Câu lệnh xóa dữ liệu

- Lệnh xóa dữ liệu đơn giản
  - Ví dụ: hãy xóa khách hàng có mã khách hàng là “kh007” trong bảng khách hàng

```
DELETE FROM khach_hang
WHERE Ma_khach_hang = ‘kh007’
```

Câu lệnh xóa dữ liệu

- Lệnh xóa dữ liệu có điều kiện được lấy từ các bảng khác
  - Khi việc xóa phức tạp hơn vì có liên quan tới các quy tắc ràng buộc dữ liệu → kết hợp câu lệnh DELETE với SELECT
  - Cú pháp:

    DELETE FROM Tên_bảng

    WHERE Tên_cột toán_tử (SELECT ... FROM ... WHERE)

Câu lệnh xóa dữ liệu

- Lệnh xóa dữ liệu có điều kiện được lấy từ các bảng khác
  - Ví dụ: xóa các hãng sữa không có sản phẩm sữa nào

```
DELETE FROM hang_sua
WHERE Ma_hang_sua NOT IN (SELECT DISTINCT
Ma_hang_sua FROM sua)
```

Xóa dữ liệu

- Sử dụng giao diện đồ họa
  - Bước 1: Chọn bảng muốn xóa dữ liệu và mở bảng dưới chế độ

  - Bước 2: Khi các dòng dữ liệu hiển thị, chúng ta tìm tới dòng muốn xóa dữ liệu và chọn biểu tượng
  - Màn hình xác nhận việc xóa sẽ được hiển thị

  - Bước 3: Chọn “OK” để hoàn thành việc xóa mẩu tin. (nếu không muốn xóa thì nhấn “Cancel”)

Sử dụng mệnh đề Union trong truy vấn

- Mệnh đề UNION dùng để kết nối dữ liệu của các câu lệnh truy vấn lại với nhau.
- Cú pháp

```
SELECT Danh_sách_các_cột_1
FROM Tên_bảng_1
[WHERE ...]
[GROUP BY ...
[HAVING ...]]
UNION
SELECT Danh_sách_các_cột_2
FROM Tên_bảng_2
[WHERE ...]
[GROUP BY ...
[HAVING ...]]
[ORDER BY ...]
```

Sử dụng mệnh đề Union trong truy vấn

- Chú ý:
  - Với truy vấn sử dụng UNION thì danh sách các cột trong các câu truy vấn phải tương ứng với nhau về số lượng, thứ tự và kiểu dữ liệu của các cột.
  - Khi dùng UNION, việc đặt tiêu đề cột được thực hiện ngay truy vấn đầu tiên.
  - Với UNION có thể kết hợp nhiều truy vấn với nhau.

Sử dụng mệnh đề Union trong truy vấn

- Ví dụ: tạo truy vấn liệt kê danh sách các phiếu nhập xuất kho. Thông tin liệt kê gồm: Mã phiếu, mã sữa và số lượng.

Ma_phieu_nhap	Ma_sua	So_luong
PN001	AB0001	100
PN002	AB0002	150

Ma_phieu_xuat	Ma_sua	So_luong
PX001	AB0001	90
PX002	AB0002	140

Ma_phieu	Ma_sua	So_luong
PN001	AB0001	100
PN002	AB0002	150
PX001	AB0001	90
PX002	AB0002	140

```
SELECT Ma_phieu_nhap as
Ma_phieu, Ma_sua, So_luong
FROM nhap_kho
UNION
SELECT Ma_phieu_xuat, Ma_sua,
So_luong
FROM xuat_kho
```

Sử dụng hàm trong SQL

- Các hàm cấu trúc điều khiển
- Các hàm chuyển đổi kiểu dữ liệu
- Các hàm xử lý chuỗi
- Các hàm xử lý số
- Các hàm xử lý thời gian

Sử dụng hàm trong SQL

- Các hàm cấu trúc điều khiển
  - IF
  - IFNULL
  - NULLIF
  - CASE

Sử dụng hàm trong SQL

- Các hàm cấu trúc điều khiển
  - Hàm IF
    - Cú pháp:

IF (biểu_thức_so_sánh, biểu_thức_1, biểu_thức_2)

    - Khi biểu thức so sánh đúng thì kết quả trả về là biểu thức 1, ngược lại thì kết quả trả về là biểu thức 2
    - Ví dụ:

```
SELECT IF(1>2,2,3)
```

IF(1>2,2,3)
3

Sử dụng hàm trong SQL

- Các hàm cấu trúc điều khiển
  - Hàm IFNULL
    - Cú pháp:

    IFNULL(biểu thức 1, biểu thức 2)

    - Nếu biểu thức 1 khác NULL thì hàm IFNULL có kết quả trả về là biểu thức 1, ngược lại thì kết quả trả về là biểu thức 2
    - Hàm IFNULL trả về giá trị số hoặc chuỗi tùy thuộc vào nội dung trong các biểu thức.

Sử dụng hàm trong SQL

- Các hàm cấu trúc điều khiển
  - Hàm IFNULL
    - Ví dụ:

```
SELECT IFNULL(1,0)
```

IFNULL(1,0)
1

```
SELECT IFNULL(10*NULL, '10')
```

IFNULL(10*NULL, '10')
10

Sử dụng hàm trong SQL

- Các hàm cấu trúc điều khiển
  - Hàm NULLIF
    - Cú pháp:

      NULLIF(biểu_thức_1,biểu_thức_2)
    - Nếu biểu thức 1 bằng biểu thức 2 thì hàm NULLIF có kết quả trả về là NULL, ngược lại thì kết quả trả về là biểu thức 1

Sử dụng hàm trong SQL

- Các hàm cấu trúc điều khiển
  - Hàm NULLIF
    - Ví dụ:

```
SELECT NULLIF(1,1)
```

NULLIF(1,1)
NULL

```
SELECT NULLIF(1,2)
```

NULLIF(1,2)
1

Sử dụng hàm trong SQL

- Các hàm cấu trúc điều khiển
  - CASE: thực hiện việc so sánh một giá trị hay một biểu thức với hàng loạt các giá trị khác để đưa về một kết quả thích hợp với giá trị hay biểu thức đã đem so sánh

Sử dụng hàm trong SQL

- Các hàm cấu trúc điều khiển
  - CASE dạng đơn giản
    - Cú pháp:

```
CASE biểu_thức_giá_trị
WHEN giá_trị_so_sánh THEN kết_quả
[WHEN giá_trị_so_sánh THEN kết_quả ...]
[ELSE kết_quả]
END
```

Sử dụng hàm trong SQL

- Các hàm cấu trúc điều khiển
  - CASE dạng đơn giản
    - Ví dụ:

```
SELECT CASE 10*2
   WHEN 20 THEN 20
   WHEN 30 THEN 30
   WHEN 40 THEN 40
END
```

CASE 10*2 WHEN 20 THEN 20 WHEN 30 THEN 30 WHEN 40 THEN 40 END
20

Sử dụng hàm trong SQL

- Các hàm cấu trúc điều khiển
  - CASE dạng có điều kiện
    - Cú pháp:

```
CASE
  WHEN điều_kiện_1 THEN kết_quả_1
  WHEN điều_kiện_2 THEN kết_quả_2
  ...
  [WHEN điều_kiện_n-1 THEN kết_quả_n-1]
  [ELSE kết_quả_n]
END
```

Sử dụng hàm trong SQL

- Các hàm cấu trúc điều khiển
  - CASE dạng có điều kiện
    - Ví dụ:

```
SELECT CASE
        WHEN 10*2=30 THEN '30'
        WHEN 10*2=40 THEN '40'
        ELSE '10*2=20'
        END
```

CASE WHEN 10*2=30 THEN '30' WHEN 10*2=40 THEN '40' ELSE '10*2=20' END
10*2=20

Sử dụng hàm trong SQL

- Các hàm chuyển đổi kiểu dữ liệu
  - Hàm cast
  - Hàm convert

Sử dụng hàm trong SQL

- Các hàm chuyển đổi kiểu dữ liệu
  - Hàm cast
    - Dùng để chuyển đổi một giá trị hoặc biểu thức sang một kiểu dữ liệu khác.
    - Kết quả trả về là giá trị hoặc biểu thức với kiểu dữ liệu mới
    - Cú pháp:

CAST(biểu_thức AS kiểu_dữ_liệu)

    - Kiểu dữ liệu có thể là một trong các kiểu sau: BINARY[(*N*)], CHAR[(*N*)], DATE, DATETIME, DECIMAL, SIGNED [INTEGER], TIME, UNSIGNED [INTEGER]

Sử dụng hàm trong SQL

- Các hàm chuyển đổi kiểu dữ liệu
  - Hàm cast
    - Ví dụ:

```
SELECT CAST(20071212 AS DATE)
```

CAST(20071212 AS DATE)
2007-12-12

Sử dụng hàm trong SQL

- Các hàm chuyển đổi kiểu dữ liệu
  - Hàm convert
    - Dạng 1: dùng để chuyển đổi một giá trị hoặc biểu thức sang một kiểu dữ liệu khác.
    - Kết quả trả về là giá trị hoặc biểu thức với kiểu dữ liệu mới
    - Cú pháp:

CONVERT(biểu_thức, kiểu_dữ_liệu)

Sử dụng hàm trong SQL

- Các hàm chuyển đổi kiểu dữ liệu
  - Hàm convert
    - Ví dụ:

```
SELECT CONVERT(20071212, DATE)
```

CONVERT(20071212, DATE)
2007-12-12

Sử dụng hàm trong SQL

- Các hàm chuyển đổi kiểu dữ liệu
  - Hàm convert
    - Dạng 2: dùng để chuyển một giá trị hoặc một biểu thức sang một kiểu hiển thị khác
    - Kết quả trả về là giá trị hoặc biểu thức dưới dạng hiển thị mới
    - Cú pháp:

CONVERT(biểu_thức USING kiểu_hiển_thị)

    - Kiểu hiển thị có thể là một trong các kiểu sau: utf8, latin1.., latin7, ascii, binary...

Sử dụng hàm trong SQL

- Các hàm chuyển đổi kiểu dữ liệu
  - Hàm convert
    - Ví dụ:

```
SELECT CONVERT('cats and dogs' USING latin2)
```

↓

```
CONVERT("cats and dogs" USING latin2)
cats and dogs
```

Sử dụng hàm trong SQL

- Các hàm xử lý chuỗi
  - char_lenght(), character_length() và length()
  - concat() và concat_ws()
  - lower() và upper()
  - left() / right() / mid() / substring()
  - repeat()
  - reverse()
  - replace()
  - encode() và decode()
  - space()
  - strcmp()

Sử dụng hàm trong SQL

- Các hàm xử lý chuỗi
  - char_lenght(), character_length() và length()
    - Kết quả trả về là chiều dài của chuỗi (str) nhưng theo hai dạng là chiều dài tính theo ký tự (char_length(), character_length()) và chiều dài tính theo byte (length)

Sử dụng hàm trong SQL

- Các hàm xử lý chuỗi
  - char_lenght(), character_length() và length()
    - Cú pháp:

      CHAR_LENGTH(str)

      CHARACTER_LENGTH(str)
    - Kết quả trả về số ký tự có trong chuỗi bao gồm cả khoàng trắng

Sử dụng hàm trong SQL

- Các hàm xử lý chuỗi
  - char_lenght(), character_length() và length()
    - Ví dụ:

```
SELECT CHAR_LENGTH("anh và em")
```

CHAR_LENGTH("anh và em")
9

Sử dụng hàm trong SQL

- Các hàm xử lý chuỗi
  - char_lenght(), character_length() và length()
    - Cú pháp:

LENGTH(str)

    - Kết quả trả về chiều dài của chuỗi được tính bằng byte.
    - Ví dụ:

```
SELECT LENGTH("anh và em")
```

LENGTH("anh và em")
10

Sử dụng hàm trong SQL

- Các hàm xử lý chuỗi
  - concat() và concat_ws()
    - Nối các chuỗi lại với nhau thành một chuỗi mới

Sử dụng hàm trong SQL

- Các hàm xử lý chuỗi
  - concat() và concat_ws()
    - Cú pháp:

      CONCAT(str1, str2 [,…])
    - Kết quả trả về là một chuỗi mới kết hợp từ các chuỗi được liệt kê trong hàm CONCAT()
    - Ví dụ

```
SELECT CONCAT( "anh"," ", "và"," ", "em" )
```

concat("anh", " " , "và" ," ", "em")
anh và em

Sử dụng hàm trong SQL

- Các hàm xử lý chuỗi
  - concat() và concat_ws()
    - Hàm CONCAT() sẽ trả về giá trị NULL khi một chuỗi trong hàm CONCAT() có giá trị NULL
    - Ví dụ:

```
SELECT CONCAT('My', NULL, 'SQL')
```

CONCAT("My", NULL, "SQL")
NULL

Sử dụng hàm trong SQL

- Các hàm xử lý chuỗi
  - concat() và concat_ws()
    - Khi muốn sử dụng chỉ định dấu phân cách giữa các chuỗi khi kết hợp chúng với nhau → sử dụng hàm CONCAT_WS()
    - Cú pháp:

CONCAT_WS(chỉ_định_cách, str1, str2 [, …])

- Ví dụ:

```
SELECT CONCAT_WS(" ", "anh", "và", "em")
```

CONCAT_WS(" ", "anh", "và", "em")
anh và em

Sử dụng hàm trong SQL

- Các hàm xử lý chuỗi
  - lower() và upper()
    - Hàm lower() có kết quả trả về là một chuỗi sau khi đã chuyển các ký tự trong chuỗi thành chữ thường
    - Cú pháp:

LOWER(str)

- Ví dụ:

```
SELECT LOWER('WWw.hcmuns.EDU.vn')
```

LOWER('WWw.hcmuns.EDU.vn')
www.hcmuns.edu.vn

Sử dụng hàm trong SQL

- Các hàm xử lý chuỗi
  - lower() và upper()
    - Hàm upper() có kết quả trả về là một chuỗi sau khi đã chuyển các ký tự trong chuỗi thành chữ hoa
    - Cú pháp:

UPPER(str)

Sử dụng hàm trong SQL

- Các hàm xử lý chuỗi
  - lower() và upper()
    - Ví dụ: hiển thị thông tin trong bảng hang_sua: tên hãng sữa đổi thành chữ in, và email đổi thành chữ thường

```
SELECT
UPPER(Ten_hang_sua) as
'Tên viết hoa',
LOWER(Email) as 'Email viết
thường'
FROM hang_sua
```

Tên viết hoa	Email viết thường
VINAMILK	vinamilk@vnm.com
NUTIFOOD	nutifood@ntf.com
ABBOTT	abbott@ab.com
DAISY	daisy@ds.com
DUTCH LADY	dutchlady@dl.com
DUMEX	dumex@dm.com
MEAD JONHSON	meadjohn@mj.com

Sử dụng hàm trong SQL

- Các hàm xử lý chuỗi
  - left() / right() / mid() / substring()
    - Kết quả trả về là một chuỗi con được trích ra từ chuỗi gốc. Trong đó chuỗi con được trích ra có thể bắt đầu từ bên trái (LEFT()), bên phải (RIGHT()), hay ở một vị trí bất kỳ trong chuỗi (MID(), SUBSTRING()) và lấy ra bao nhiêu kí tự tính từ vị trí đó.
    - Cú pháp:

LEFT(str, số_kí_tự)

RIGHT(str, số_kí_tự)

MID(str, vị_trí, số_kí_tự)

Sử dụng hàm trong SQL

- Các hàm xử lý chuỗi
  - left() / right() / mid() / substring()
    - Ví dụ:

```
SELECT LEFT("anh và em", 3)
```

→

left("anh và em", 3)
anh

```
SELECT RIGHT("anh và em", 2)
```

→

RIGHT('anh và em', 2)
em

```
SELECT MID("anh và em", 4,3)
```

→

MID("anh và em", 4,3)
và

Sử dụng hàm trong SQL

- Các hàm xử lý chuỗi
  - left() / right() / mid() / substring()
    - Khi hàm MID() không có tham số số_byte thì chuỗi con mới sẽ có nội dung bắt đầu từ vị_trí đến hết chuỗi gốc
    - Ví dụ:

SELECT MID("anh và em",4) → **và em**

Sử dụng hàm trong SQL

- Các hàm xử lý chuỗi
  - left() / right() / mid() / substring()
    - Cú pháp:

SUBSTRING(str, vị_trí)

SUBSTRING(str, vị_trí, số_kí_tự)

SUBSTRING(str FROM vị_trí)

SUBSTRING(str FROM vị_trí FOR số_kí_tự)

Sử dụng hàm trong SQL

- Các hàm xử lý chuỗi
  - left() / right() / mid() / substring()
    - Ví dụ:

```
SELECT SUBSTRING(“Happy new year”, 7)
```

→

SUBSTRING("Happy new year", 7)
new year

```
SELECT SUBSTRING( "Happy new year" FROM 7 )
```

→

SUBSTRING("Happy new year" FROM 7)
new year

```
SELECT SUBSTRING("Happy new year", 7, 3)
```

→

SUBSTRING("Happy new year", 7, 3)
new

```
SELECT SUBSTRING( "Happy new year" FROM 7 FOR 3 )
```

→

SUBSTRING("Happy new year" FROM 7 FOR 3)
new

Sử dụng hàm trong SQL

- Các hàm xử lý chuỗi
  - repeat()
    - Được dùng để lặp lại nhiều lần một chuỗi.
    - Cú pháp:

REPEAT(str, số_lần_lặp)

    - Kết quả trả về là một chuỗi mới được tạo ra từ chuỗi được lặp lại.
    - Ví dụ:

```
SELECT REPEAT("Tôi đi học", 3)
```

→

REPEAT("Tôi đi học", 3)
Tôi đi họcTôi đi họcTôi đi học

Sử dụng hàm trong SQL

- Các hàm xử lý chuỗi
  - reverse()
    - Cú pháp:

      REVERSE(str)
    - Kết quả trả về là một chuỗi đảo ngược
    - Ví dụ:

SELECT REVERSE("anh và em") →

REVERSE("anh và em")
me àv hna

Sử dụng hàm trong SQL

- Các hàm xử lý chuỗi
  - replace()
    - Cú pháp:

REPLACE(chuỗi_nguồn, chuỗi_cần_tìm, chuỗi_thay_thế)

    - Kết quả trả về là một chuỗi mới sau khi tìm và thay thế một chuỗi con trong chuỗi nguồn bằng một chuỗi khác

```
SELECT REPLACE("See you again!", "again", "later")
```

REPLACE("See you again!", "again", "later")
See you later!

Sử dụng hàm trong SQL

- Các hàm xử lý chuỗi
  - encode() và decode()
    - Hàm ENCODE() dùng để mã hóa một chuỗi
    - Cú pháp:

ENCODE(str, khóa)

  - str: là chuỗi sẽ được mã hóa dưới dạng chuỗi nhị phân.
  - khóa: là password do chúng ta đặt ra để không cho phép người khác giải mã

- Ví dụ: khi có một người mới đăng ký vào hệ thống, họ sẽ nhập vào mật khẩu => cần phải mã hóa mật khẩu này

SELECT ENCODE("phuong", "nd1") → ENCODE("phuong", "nd1")

_□9□□s

Sử dụng hàm trong SQL

- Các hàm xử lý chuỗi
  - encode() và decode()
    - Hàm DECODE() dùng để giải mã thông tin đã bị mã hóa.
    - Cú pháp:

DECODE(str, khóa)

  - str: là chuỗi đã bị mã hóa
  - khóa: là mật khẩu được đặt ra khi tiến hành mã hóa. Không có mật khẩu này thì không thể giải mã.

Sử dụng hàm trong SQL

- Các hàm xử lý chuỗi
  - encode() và decode()
    - Ví dụ: cột mật khẩu bị mã hóa như sau:

ten_dang_nhap	encode(`mat_khau` , 'nd1')
phuong	□□□□X8
thien	□□□□X8
quy	□□□□X8
thy	□□□□X8

Giải mã cột mật khẩu bị mã hóa

```
SELECT ten_dang_nhap,
DECODE(mat_khau, 'nd1')
```

ten_dang_nhap	Giải mã mật khẩu
phuong	123456
thien	123456
quy	123456
thy	123456

Sử dụng hàm trong SQL

- Các hàm xử lý chuỗi
  - space()
    - Cú pháp:

      SPACE(N)
    - Kết quả trả về là một chuỗi có N khoảng trắng
    - Ví dụ: Xuất tên các hãng sữa và số điện thoại – trong đó tên cách số điện thoại 10 khoảng trắng

```
SELECT CONCAT( Ten_hs,
SPACE(10) , Dien_thoai )
FROM hang_sua
```

CONCAT(Ten_hs , SPACE(10) , Dien_thoai)
Vinamilk 8794561
Nutifood 7895632
Abbott 8741258
Daisy 5789321
Dutch Lady 7826451
Dumex 6258943
Mead Jonhson 8741258

Sử dụng hàm trong SQL

- Các hàm xử lý chuỗi
  - strcmp()
    - Dùng để so sánh chính xác hai chuỗi
    - Cú pháp:

    STRCMP(str1, str2)

    - Kết quả trả về bằng 0 nếu hai chuỗi giống nhau, trả về -1 nếu chuỗi 1 nhỏ hơn chuỗi 2, trả về 1 nếu chuỗi 1 lớn hơn chuỗi 2

Sử dụng hàm trong SQL

- Các hàm xử lý chuỗi
  - strcmp()
    - Ví dụ:

```
SELECT STRCMP('text', 'text2')
```

STRCMP('text', 'text2')
-1

```
SELECT STRCMP('text2', 'text')
```

STRCMP('text2', 'text')
1

```
SELECT STRCMP('text', 'text')
```

STRCMP('text', 'text')
0

Sử dụng hàm trong SQL

- Các hàm xử lý số
  - abs()
  - ceiling() / ceil()
  - floor()
  - mod()
  - pi()
  - pow() / power()
  - round()
  - sqrt()
  - sign()
  - rand()

Sử dụng hàm trong SQL

- Các hàm xử lý số
  - abs()
    - Cú pháp:

      ABS(số)
    - Kết quả trả về là giá trị tuyệt đối của một số
    - Ví dụ:

```
SELECT ABS(-32)
```

→

ABS(-32)
32

Sử dụng hàm trong SQL

- Các hàm xử lý số
  - ceiling() / ceil(): dùng để làm tròn số theo cận trên
    - Cú pháp:

      CEILING(X)

      CEIL(X)
    - Kết quả trả về là số nguyên nhỏ nhất có giá trị không nhỏ hơn X
    - Ví dụ:

```
SELECT CEILING(9.327)
```

→

CEILING(9.327)
10

Sử dụng hàm trong SQL

- Các hàm xử lý số
  - floor(): làm tròn theo cận dưới
    - Cú pháp:

      FLOOR(X)
    - Kết quả trả về là số nguyên lớn nhất có giá trị không lớn hơn X
    - Ví dụ:

```
SELECT FLOOR(9.327)
```

FLOOR(9.327)
9

Sử dụng hàm trong SQL

- Các hàm xử lý số
  - mod(): chia lấy phần dư
    - Cú pháp:

      MOD(số_thứ_1, số_thứ_2)
    - Kết quả trả về của hàm là phần dư của phép chia số thứ 1 cho số thứ 2
    - Ví dụ:

```
SELECT MOD(34.5,3)
```

MOD(34.5,3)
1.5

Sử dụng hàm trong SQL

- Các hàm xử lý số
  - pi()
    - Cú pháp:

      PI()
    - Kết quả trả về là giá trị của hằng số PI trong toán học
    - Ví dụ:

```
SELECT PI()
```

PI()
3.141593

Sử dụng hàm trong SQL

- Các hàm xử lý số
  - pow() / power(): tính lũy thừa của một số
    - Cú pháp

      POW(số, số_mũ)

      POWER(số, số_mũ)
    - Kết quả trả về là phép tính lũy thừa của một số bất kỳ theo một số mũ chỉ định
    - Ví dụ:

```
SELECT POW(2,2)
```

POW(2,2)
4

Sử dụng hàm trong SQL

- Các hàm xử lý số
  - round(): làm tròn
    - Cú pháp:

      ROUND(số [, vị_trí_làm_tròn])
      - Vị_trí_làm_tròn: có thể là số nguyên âm hoặc nguyên dương cho biết vị trí muốn làm tròn tính từ dấu chấm thập phân
    - Kết quả trả về là số đã được làm tròn

Sử dụng hàm trong SQL

- Các hàm xử lý số
  - round(): làm tròn
    - Ví dụ:

```
SELECT ROUND(4.27943, 2)
```

ROUND(4.27943, 2)
4.28

```
SELECT ROUND(-1.58)
```

ROUND(-1.58)
-2

```
SELECT ROUND(23.298, -1)
```

ROUND(23.298, -1)
20

Sử dụng hàm trong SQL

- Các hàm xử lý số
  - sqrt(): tính căn bậc hai của một số dương
    - Cú pháp:

      SQRT(số)
    - Ví dụ:

```
SELECT SQRT(4)
```

→

SQRT(4)
2

Sử dụng hàm trong SQL

- Các hàm xử lý số
  - sign(): xét dấu của số hay biểu thức
    - Cú pháp:

      SIGN(số)
    - Kết quả trả về là 1 nếu số hay biểu thức là số dương, -1 nếu số âm, 0 nếu số bằng 0
    - Ví dụ:

```
SELECT SIGN(-32)
```

→

SIGN(-32)
-1

Sử dụng hàm trong SQL

- Các hàm xử lý số
  - rand(): cấp phát ngẫu nhiên
    - Cú pháp:

      RAND([N])
      - N: là một số nguyên, được coi là giá trị nguồn giúp tạo ra các giá trị giống nhau khi gọi hàm RAND()
    - Kết quả trả về là một số thực ngẫu nhiên do MySQL tự động cấp phát có giá trị trong khoảng từ 0->1

Sử dụng hàm trong SQL

- Các hàm xử lý số
  - rand(): cấp phát ngẫu nhiên
    - Ví dụ:

```
SELECT RAND()
```

RAND()
0.10702744136288

```
SELECT RAND(20)
```

RAND(20)
0.15888261251047

Sử dụng hàm trong SQL

- Các hàm xử lý thời gian
  - adddate() / date_add() / subdate() / date_sub()
  - curdate() / current_date() / curtime() / current_time() / now()
  - date() / month() / monthname() / year()
  - day() / dayofmonth() / dayname() / dayofweek() / dayofyear()
  - second() / minute() / hour() / time()
  - datediff() / timediff()

Sử dụng hàm trong SQL

- Các hàm xử lý thời gian
  - adddate() / date_add() / subdate() / date_sub()
    - adddate() và date_add() là hai hàm có cùng kết quả trả về là một ngày mới sau khi đã cộng thêm một đơn vị thời gian
    - Cú pháp dạng 1:

ADDDATE(ngày, số_ngày)

DATE_ADD(ngày, số_ngày)

```
SELECT ADDDATE('2007-12-13', 31)
```

ADDDATE('2007-12-13', 31)
2008-01-13

Sử dụng hàm trong SQL

- Các hàm xử lý thời gian
  - adddate() / date_add() / subdate() / date_sub()
    - adddate() và date_add()
    - Cú pháp dạng 2:

ADDDATE(ngày, INTERVAL giá_trị kiểu)

DATE_ADD(ngày, INTERVAL giá_trị kiểu)

  - Kiểu: là kiểu của giá_trị (tham khảo thêm các kiểu trong giáo trình)

    - Ví dụ:

```
SELECT ADDDATE('2007-12-13',
INTERVAL 31 DAY)
```

→

ADDDATE('2007-12-13', INTERVAL 31 DAY)
2008-01-13

Sử dụng hàm trong SQL

- Các hàm xử lý thời gian
  - adddate() / date_add() / subdate() / date_sub()
    - subdate() và date_sub() là hai hàm có cùng kết quả trả về là một ngày mới sau khi đã trừ đi một đơn vị thời gian.
    - Cách sử dụng và cú pháp của hai hàm này tương tự như hai hàm adddate() và date_add()

```
SELECT SUBDATE('2007-12-13 23:59:59', 31)
```

→

SUBDATE('2007-12-13 23:59:59', 31)
2007-11-12 23:59:59

```
SELECT DATE_SUB(CURDATE(), INTERVAL 30 DAY)
```

→

DATE_SUB(CURDATE(),INTERVAL 30 DAY)
2007-11-13

Sử dụng hàm trong SQL

- Các hàm xử lý thời gian
  - curdate() / current_date() / curtime() / current_time() / now()
    - curdate(), current_date() có kết quả trả về là ngày hiện hành của hệ thống
    - Cú pháp:

CURDATE()

CURRENT_DATE()

Sử dụng hàm trong SQL

- Các hàm xử lý thời gian
  - curdate() / current_date() / curtime() / current_time() / now()
    - curdate(), current_date()
    - Ví dụ:

```
SELECT CURDATE()
```

CURDATE()
2007-12-13

```
SELECT CURRENT_DATE()
```

CURRENT_DATE()
2007-12-13

Sử dụng hàm trong SQL

- Các hàm xử lý thời gian
  - curdate() / current_date() / curtime() / current_time() / now()
    - curtime(), current_time() có kết quả trả về là giờ hiện hành của hệ thống
    - Cú pháp:

CURTIME()

CURRENT_TIME()

Sử dụng hàm trong SQL

- Các hàm xử lý thời gian
  - curdate() / current_date() / curtime() / current_time() / now()
    - curtime(), current_time()
    - Ví dụ:

```
SELECT CURTIME()
```

CURTIME()
16:45:23

```
SELECT CURRENT_TIME()
```

CURRENT_TIME()
16:46:23

Sử dụng hàm trong SQL

- Các hàm xử lý thời gian
  - curdate() / current_date() / curtime() / current_time() / now()
    - now(): có kết quả trả về là ngày giờ hiện hành của hệ thống
    - Cú pháp:

      NOW()

    - Ví dụ:

```
SELECT NOW()
```

NOW()
2007-12-13 16:49:04

Sử dụng hàm trong SQL

- Các hàm xử lý thời gian
  - date() / month() / monthname() / year()
    - date() có kết quả trả về là ngày-tháng-năm của một biểu thức thời gian bất kỳ
    - Cú pháp:

DATE(biểu thức thời gian)

- Ví dụ:

```
SELECT DATE('2007-12-14 07:58:59')
```

DATE('2007-12-14 07:58:59')
2007-12-14

Sử dụng hàm trong SQL

- Các hàm xử lý thời gian
  - date() / month() / monthname() / year()
    - month() có kết quả trả về là tháng của một biểu thức thời gian bất kỳ
    - Cú pháp:

MONTH(biểu thức thời gian)

- Ví dụ:

```
SELECT MONTH('2007-12-13 08:03:20')
```

→

MONTH('2007-12-13 08:03:20')
12

Sử dụng hàm trong SQL

- Các hàm xử lý thời gian
  - date() / month() / monthname() / year()
    - monthname() có kết quả trả về là tên của tháng (tiếng Anh) của của một biểu thức thời gian bất kỳ
    - Cú pháp:

MONTHNAME(biểu thức thời gian)

- Ví dụ:

```
SELECT
MONTHNAME(NOW())
```

MONTHNAME(NOW())
December

Sử dụng hàm trong SQL

- Các hàm xử lý thời gian
  - date() / month() / monthname() / year()
    - year() có kết quả trả về là năm của một biểu thức thời gian bất kỳ
    - Cú pháp:

YEAR(biểu thức thời gian)

- Ví dụ:

```
SELECT YEAR('2007-12-14 08:13:40')
```

YEAR('2007-12-14 08:13:40')
2007

Sử dụng hàm trong SQL

- Các hàm xử lý thời gian
  - day() / dayofmonth() / dayname() / dayofweek() / dayofyear()
    - day() và dayofmonth() có kết quả trả về là giá trị ngày của một biểu thức thời gian có kiểu ngày/ngày giờ bất kỳ
    - Cú pháp:

DAY(biểu thức thời gian)

DAYOFMONTH(biểu thức thời gian)

Sử dụng hàm trong SQL

- Các hàm xử lý thời gian
  - day() / dayofmonth() / dayname() / dayofweek() / dayofyear()
    - day() và dayofmonth()
    - Ví dụ:

```
SELECT DAY(NOW())
```

DAY(NOW())
14

```
SELECT DAYOFMONTH('2007-12-14 10:10:10')
```

DAYOFMONTH('2007-12-14 10:10:10')
14

Sử dụng hàm trong SQL

- Các hàm xử lý thời gian
  - day() / dayofmonth() / dayname() / dayofweek() / dayofyear()
    - dayname() có kết quả trả về là tên của ngày trong tuần của một biểu thức thời gian có kiểu ngày/ngày giờ bất kỳ
    - Cú pháp:

DAYNAME(biểu thức thời gian)

- Ví dụ:

```
SELECT DAYNAME('2007-12-24 23:59:59')
```

→

DAYNAME('2007-12-24 23:59:59')
Monday

Sử dụng hàm trong SQL

- Các hàm xử lý thời gian
  - day() / dayofmonth() / dayname() / dayofweek() / dayofyear()
    - dayofweek(): kết quả trả về là giá trị số tương ứng với ngày trong tuần
    - Cú pháp:

      DAYOFWEEK(biểu thức thời gian)
    - Kết quả trả về từ 1->7, trong đó 1 tương ứng với 'Sunday', 2 tương ứng với 'Monday', …
    - Ví dụ:

```
SELECT DAYOFWEEK
( '2007-12-24 23:59:59' )
```

DAYOFWEEK('2007-12-24 23:59:59')
2

Sử dụng hàm trong SQL

- Các hàm xử lý thời gian
  - day() / dayofmonth() / dayname() / dayofweek() / dayofyear()
    - dayofyear() có kết quả trả về là ngày trong năm của một biểu thức thời gian có kiểu ngày/ngày giờ bất kỳ
    - Cú pháp:

DAYOFYEAR(biểu thức thời gian)

- Ví dụ:

```
SELECT DAYOFYEAR
( '2007-12-24 23:59:59' )
```

→

DAYOFYEAR('2007-12-24 23:59:59')
358

Sử dụng hàm trong SQL

- Các hàm xử lý thời gian
  - second() / minute() / hour() / time()
    - second(), minute(), hour(), time() có kết quả trả về là một số nguyên chỉ định giây, phút, giờ và thời gian của một biểu thức thời gian có kiểu giờ:phút:giây hoặc kiểu ngày-tháng-năm giờ:phút:giây bất kỳ
    - Cú pháp:

SECOND(biểu thức thời gian)

MINUTE(biểu thức thời gian)

HOUR(biểu thức thời gian)

TIME(biểu thức thời gian)

Sử dụng hàm trong SQL

- Các hàm xử lý thời gian
  - second() / minute() / hour() / time()
    - Ví dụ:

```
SELECT SECOND(NOW())
```

SECOND(NOW())
40

```
SELECT MINUTE(NOW())
```

MINUTE(NOW())
4

```
SELECT HOUR('2007-12-14 09:06:15')
```

HOUR('2007-12-14 09:06:15')
9

```
SELECT TIME(NOW())
```

TIME(NOW())
09:07:46

Sử dụng hàm trong SQL

- Các hàm xử lý thời gian
  - datediff() / timediff()
    - DATEDIFF() có kết quả trả về là khoảng cách đại số giữa hai ngày bất kỳ
    - Cú pháp:

DATEDIFF(ngày_1, ngày_2)

    - Chú ý: Nếu ngày 1 nhỏ hơn ngày 2 thì kết quả sẽ là số nguyên âm, ngược lại thì kết quả sẽ là số nguyên dương.

Sử dụng hàm trong SQL

- Các hàm xử lý thời gian
  - datediff() / timediff()
    - DATEDIFF()
    - Ví dụ:

```
SELECT DATEDIFF('2007-12-14 23:59:59', '2008-02-13 23:59:59')
```

DATEDIFF('2007-12-14 23:59:59', '2008-02-13 23:59:59')
-61

```
SELECT DATEDIFF('2008-02-13 23:59:59', '2007-12-14 23:59:59')
```

DATEDIFF('2008-02-13 23:59:59', '2007-12-14 23:59:59')
61

Sử dụng hàm trong SQL

- Các hàm xử lý thời gian
  - datediff() / timediff()
    - TIMEDIFF() có kết quả trả về là khoảng cách đại số của hai biểu thức thời gian bất kỳ
    - Cú pháp:

TIMEDIFF(biểu_thức_thời_gian_1, biểu_thức_thời_gian_2)

- Ví dụ:

```
SELECT TIMEDIFF('23:59:59', '10:44:45')
```

TIMEDIFF('23:59:59', '10:44:45')
13:15:14