

Chương5: SQL SERVER

Bài 8: SQL SERVER

I. Tổng quan vềSQL

1. Khái niệm SQL

- SQL (Structured Query Language – ngôn ngữ hỏi có câu trúc) là công cụ sử dụng để tổ chức, quản lý và truy xuất dữ liệu được lưu trữ trong các cơ sở dữ liệu.
- SQL là một hệ thống ngôn ngữ bao gồm tập các câu lệnh sử dụng để tương tác với cơ sở dữ liệu quan hệ.
- SQL được sử dụng để điều khiển tất cả các chức năng mà một hệ quản trị cơ sở dữ liệu cung cấp cho người dùng bao gồm:
 - **Định nghĩa dữ liệu:** SQL cung cấp khả năng định nghĩa các cơ sở dữ liệu, các cấu trúc lưu trữ và tổ chức dữ liệu cũng như mối quan hệ giữa các thành phần dữ liệu.
 - **Truy xuất và thao tác dữ liệu:** Với SQL, người dùng có thể dễ dàng thực hiện các thao tác truy xuất, bổ sung, cập nhật và loại bỏ dữ liệu trong các cơ sở dữ liệu.
 - **Điều khiển truy cập:** SQL có thể được sử dụng để cấp phát và kiểm soát các thao tác của người sử dụng trên dữ liệu, đảm bảo sự an toàn cho cơ sở dữ liệu.
 - **Đảm bảo toàn vẹn dữ liệu:** SQL định nghĩa các ràng buộc toàn vẹn trong cơ sở dữ liệu nhằm đảm bảo tính hợp lệ và chính xác của dữ liệu trước các thao tác cập nhật cũng như các lỗi của hệ thống.

2. Vai trò của SQL

- SQL không phải là một hệ quản trị cơ sở dữ liệu, do nó không thể tồn tại độc lập.
- SQL là một phần của hệ quản trị cơ sở dữ liệu, nó xuất hiện trong các hệ quản trị cơ sở dữ liệu với vai trò ngôn ngữ và là công cụ giao tiếp giữa người sử dụng và hệ quản trị cơ sở dữ liệu.
- SQL có những vai trò như sau:
 - **SQL là ngôn ngữ hỏi có tính tương tác:** Người sử dụng có thể dễ dàng thông qua các trình tiện ích để gửi các yêu cầu dưới dạng các câu lệnh SQL đến cơ sở dữ liệu và nhận kết quả trả về từ cơ sở dữ liệu.
 - **SQL là ngôn ngữ lập trình cơ sở dữ liệu:** Các lập trình viên có thể nhúng các câu lệnh SQL vào trong các ngôn ngữ lập trình để xây dựng nên các chương trình ứng dụng giao tiếp với cơ sở dữ liệu.
 - **SQL là ngôn ngữ quản trị cơ sở dữ liệu:** Thông qua SQL, người quản trị cơ sở dữ liệu có thể quản lý được cơ sở dữ liệu, định nghĩa các cấu trúc lưu trữ dữ liệu, điều khiển truy cập cơ sở dữ liệu, ...
 - **SQL là ngôn ngữ cho các hệ thống khách/chủ (client/server):** Trong các hệ thống cơ sở dữ liệu khách/chủ, SQL được sử dụng như là công cụ để giao tiếp giữa các trình ứng dụng phía máy khách với máy chủ cơ sở dữ liệu.
 - **SQL là ngôn ngữ truy cập dữ liệu trên Internet:** Cho đến nay, hầu hết các máy chủ Web cũng như các máy chủ trên Internet sử dụng SQL với vai trò là ngôn ngữ để tương tác với dữ liệu trong các cơ sở dữ liệu.
 - **SQL là ngôn ngữ cơ sở dữ liệu phân tán:** Đối với các hệ quản trị cơ sở dữ liệu phân tán, mỗi một hệ thống sử dụng SQL để giao tiếp với các hệ thống khác trên mạng, gửi và nhận các yêu cầu truy xuất dữ liệu với nhau.

- SQL là ngôn ngữ sử dụng cho các công giao tiếp cơ sở dữ liệu:** Trong một hệ thống mạng máy tính với nhiều hệ quản trị cơ sở dữ liệu khác nhau, SQL thường được sử dụng như là một chuẩn ngôn ngữ để giao tiếp giữa các hệ quản trị cơ sở dữ liệu.

II. Tổng quan về cơ sở dữ liệu (CSDL) quan hệ

1. Mô hình dữ liệu quan hệ

- CSDL quan hệ là một CSDL trong đó tất cả dữ liệu được tổ chức trong các bảng (table) có mối quan hệ với nhau. Mỗi bảng (table) bao gồm các dòng (record/bản ghi/bộ) và các cột (field/trường/thuộc tính).

- Tóm lại, một CSDL bao gồm nhiều bảng (table) có mối quan hệ với nhau (relationship).

Ví dụ:

MADONVI	TENDONVI	DIENTHOAI
01	Phòng Kế toán	821451
02	Phòng Tổ chức	831414
03	Phòng điều hành	823351
04	Phòng đối ngoại	841457
05	Phòng Tài vụ	821451

MANV	HOTEN	NGAYSINH	DIACHI	DIENTHOAI	HSLUONG	MADONVI
NV01001	Nguyễn Thị Hoa	05/05/1976	56 Lê Duẩn	521304	2.11	01
NV02001	Lê Hoài Nam	03/05/1976	32 Trần Phú	823145	1.86	02
NV02002	Hoàng Nam Phong	05/08/1971	66 Hoàng Diệu	521247	3.21	02
NV03001	Trần Nguyên Phong	20/12/1976	7 Hà Nội	849290	1.92	03
NV03002	Nguyễn Hữu Tình	18/08/1976	7 Hà Nội	849290	1.92	03
NV05001	Nguyễn Trung Kiên	14/05/1972	77 Nguyễn Huệ	823236	1.86	05

2. Bảng (Table)

Bảng (table) bao gồm các yếu tố sau:

- Tên của bảng: được xác định duy nhất.
- Cấu trúc của bảng: tập hợp các cột (field/trường/thuộc tính).
- Dữ liệu của bảng: tập hợp các dòng (record/bản ghi/bộ) hiện có trong bảng.

Ví dụ: Table **DONVI**

MADONVI	TENDONVI	DIENTHOAI
01	Phòng Kế toán	821451
02	Phòng Tổ chức	831414
03	Phòng điều hành	823351
04	Phòng đối ngoại	841457
05	Phòng Tài vụ	821451

3. Khóa chính của bảng (Primary Key)

- Mỗi bảng phải có một cột (hoặc một tập các cột) mà giá trị dữ liệu của nó xác định duy nhất một dòng trong tập hợp các dòng trong bảng.

- Một cột (hoặc một tập các cột) có tính chất này gọi là khóa chính của bảng (Primary Key).

Ví dụ: Table **DONVI**

MADONVI	TENDONVI	DIENTHOAI
01	Phòng Kế toán	821451
02	Phòng Tổ chức	831414
03	Phòng điều hành	823351
04	Phòng đối ngoại	841457
05	Phong Tài vụ	821451

(có khóa chính là **MADONVI**)

4. Mối quan hệ (Relationship) và khóa ngoại (Foreign Key)

- Mối quan hệ (Relationship) được thể hiện thông qua ràng buộc *giá trị dữ liệu xuất hiện ở bảng này phải có xuất hiện trước ở một bảng khác.*

MADONVI	TENDONVI	DIENTHOAI
01	Phòng Kế toán	821451
02	Phong Tổ chức	831414
03	Phòng điều hành	823351
04	Phòng đối ngoại	841457
05	Phong Tài vụ	821451

MANV	HOTEN	NGAYSINH	DIACHI	DIENTHOAI	HSLUONG	MADONVI
NV01001	Nguyễn Thị Hoa	05/05/1976	56 Lê Duẩn	521304	2.11	01
NV02001	Lê Hoài Nam	03/05/1976	32 Trần Phú	823145	1.86	02
NV02002	Hoàng Nam Phong	05/08/1971	66 Hoàng Diệu	521247	3.21	02
NV03001	Trần Nguyên Phong	20/12/1976	7 Hà Nội	849290	1.92	03
NV03002	Nguyễn Hữu Tình	18/08/1976	7 Hà Nội	849290	1.92	03
NV05001	Nguyễn Trung Kiên	14/05/1972	77 Nguyễn Huệ	823236	1.86	05

- Một cột (hoặc tập hợp các cột) (field/trường/thuộc tính) trong một bảng mà giá trị của nó được xác định từ khóa chính (Primary Key) của một bảng khác được gọi là khóa ngoại (Foreign Key).

5. Sơ lược về câu lệnh SQL

Câu lệnh	Chức năng
<i>Thao tác dữ liệu</i>	
SELECT	Truy xuất dữ liệu
INSERT	Bổ sung dữ liệu
UPDATE	Cập nhật dữ liệu
DELETE	Xóa dữ liệu
TRUNCATE	Xóa toàn bộ dữ liệu trong bảng
<i>Định nghĩa dữ liệu</i>	
CREATE TABLE	Tạo bảng
DROP TABLE	Xóa bảng
ALTER TABLE	Sửa cấu trúc bảng
CREATE FUNCTION	Tạo hàm (do người sử dụng định nghĩa)
ALTER FUNCTION	Sửa đổi hàm
DROP FUNCTION	Xóa hàm

CREATE TRIGGER	Tạo trigger
ALTER TRIGGER	Sửa trigger
DROP TRIGGER	Xóa trigger

6. Quy tắc sử dụng tên trong SQL

- Trong câu lệnh SQL, nếu ta cần chỉ đến một bảng *do một người dùng khác sở hữu* (hiện nhiên là phải được phép) thì *tên của bảng* phải được viết sau *tên của người sở hữu* và *phân cách* với tên người sở hữu *bởi dấu chấm* theo công thức: *tên_người_sở_hữu.tên_bảng*

- Trong câu lệnh SQL, nếu có sử dụng từ *hai cột* trở lên có *cùng tên* trong các bảng khác nhau thì *bắt buộc* phải chỉ định thêm *tên bảng trước tên cột; tên bảng và tên cột* được *phân cách* nhau *bởi dấu chấm* theo công thức: *tên_bảng.tên_cột*

7. Kiểu dữ liệu

Tên kiểu	Mô tả
CHAR (n)	Kiểu chuỗi với độ dài cố định
NCHAR (n)	Kiểu chuỗi với độ dài cố định hỗ trợ UNICODE
VARCHAR (n)	Kiểu chuỗi với độ dài chính xác
NVARCHAR (n)	Kiểu chuỗi với độ dài chính xác hỗ trợ UNICODE
INTEGER	Số nguyên có giá trị từ -2^{31} đến $2^{31} - 1$
INT	Như kiểu Integer
TINYINT	Số nguyên có giá trị từ 0 đến 255
SMALLINT	Số nguyên có giá trị từ -2^{15} đến $2^{15} - 1$
BIGINT	Số nguyên có giá trị từ -2^{63} đến $2^{63} - 1$
NUMERIC (p,s)	Kiểu số với độ chính xác cố định
DECIMAL (p,s)	Tương tự kiểu Numeric
FLOAT	Số thực có giá trị từ $-1.79E+308$ đến $1.79E+308$
REAL	Số thực có giá trị từ $-3.40E + 38$ đến $3.40E + 38$
MONEY	Kiểu tiền tệ
BIT	Kiểu bit (có giá trị 0 hoặc 1)
DATETIME	Kiểu ngày giờ (chính xác đến phần trăm của giây)
SMALLDATETIME	Kiểu ngày giờ (chính xác đến phút)
BINARY	Dữ liệu nhị phân với độ dài cố định (tối đa 8000 bytes)
VARBINARY	Dữ liệu nhị phân với độ dài chính xác (tối đa 8000 bytes)
IMAGE	Dữ liệu nhị phân với độ dài chính xác ($\leq 2,147,483,647$ bytes)
TEXT	Dữ liệu kiểu chuỗi với độ dài lớn (tối đa 2,147,483,647 ký tự)
NTEXT	Dữ liệu kiểu chuỗi với độ dài lớn và hỗ trợ UNICODE (tối đa 1,073,741,823 ký tự)

8. Toán tử

Toán tử	Ý nghĩa
a) Logic	
AND / OR	Và / Hoặc
b) So sánh	
=	Bằng
>	Lớn hơn
<	Nhỏ hơn
\geq	Lớn hơn hoặc bằng

\leq	Nhỏ hơn hoặc bằng	
\neq	Khác	
$>!$	Không lớn hơn	
$<!$	Không nhỏ hơn	
c) Danh sách		
IN	Nằm trong danh sách	
NOT IN	Không nằm trong danh sách	
d) Giới hạn dữ liệu		
BETWEEN	Giá trị BETWEEN a AND b nghĩa là ($a \leq$ Giá trị $\leq b$)	
NOT BETWEEN	Giá trị NOT BETWEEN a AND b nghĩa là (Giá trị $< a$) AND (Giá trị $> b$)	
LIKE	Mô tả khuôn dạng dữ liệu cần tìm kiếm có sử dụng ký tự đại diện:	
	%	Chuỗi ký tự bất kỳ
	_	Một ký tự bất kỳ
	[]	Ký tự bất kỳ trong giới hạn được chỉ định
	[^]	Ký tự bất kỳ không nằm trong giới hạn được chỉ định

III. Table (Bảng)

* Quy tắc khi viết câu lệnh SQL:

- Các câu lệnh SQL không phân biệt chữ hoa và chữ thường. Tuy nhiên, để dễ đọc nên viết hoa từ khóa trong mệnh đề.
- Câu lệnh SQL có thể viết trên một dòng hay nhiều dòng. Nhưng nên viết mỗi dòng một mệnh đề.
- Không tách từ khóa trên nhiều dòng, không viết tắt từ khóa.
- Câu lệnh SQL kết thúc bằng dấu chấm phẩy (;

1. Tạo bảng

Cú pháp:

```
CREATE TABLE tên_bảng
(
    tên_cột      thuộc_tính_cột      các_ràng_buộc
    [ , ...
     , tên_cột_n  thuộc_tính_cột_n  các_ràng_buộc_cột_n ]
    [ , các_ràng_buộc_trên_bảng ]
)
```

Trong đó:

tên_bảng	Tên của bảng cần tạo (<=128 ký tự)
tên_cột	Tên của cột (field / trường) cần định nghĩa
thuộc_tính_cột	Gồm: <ul style="list-style-type: none"> • Kiểu dữ liệu của cột (field). • Giá trị mặc định của cột (filed). • IDENTITY - giá trị tự động tăng, dùng với field kiểu số. • NULL / NOT NULL
các_ràng_buộc	Các ràng buộc được sử dụng trên mỗi cột (field) hoặc trên bảng.

Ví dụ 8.1:

Tạo bảng NHANVIEN gồm các field MANV (mã nhân viên), HOTEN (họ và tên), NGAYSINH (ngày sinh của nhân viên), DIACHI (địa chỉ của nhân viên), HSLUONG (hệ số lương), MADONVI (mã đơn vị).

* Hướng dẫn:

```
CREATE TABLE nhanvien
```

```
(  
    manv      NVARCHAR(10)      NOT NULL,  
    hoten     NVARCHAR(50)      NOT NULL,  
    ngaysinh   DATE             NULL,  
    diachi    NVARCHAR(100)     NULL,  
    dienthoai NVARCHAR(10)      NULL,  
    hsluong   DECIMAL(3,2)      DEFAULT (1.92)  
    madonvi   NVARCHAR(10)      NOT NULL  
)
```

2. Tạo ràng buộc

a. Ràng buộc CHECK:

- Chỉ định điều kiện hợp lệ đối với dữ liệu khi có sự thay đổi dữ liệu trên bảng.
- Dùng với các lệnh INSERT, UPDATE.

Cú pháp:

```
[CONSTRAINT tên_ràng_buộc]  
CHECK (điều_kiện)
```

Ví dụ 8.2:

Tạo bảng NHANVIEN như ví dụ 8.1, trong đó:

- NGAYSINH < '1/1/1990'
- DIENTHOAI của nhân viên là một chuỗi 6 chữ số.

* Hướng dẫn:

```
CREATE TABLE nhanvien
```

```
(  
    manv      NVARCHAR(10)      NOT NULL,  
    hoten     NVARCHAR(50)      NOT NULL,  
    ngaysinh   DATE             NULL  
        CONSTRAINT CK_nhanvien_ngaysinh  
        CHECK (ngaysinh < '1/1/1990'),  
    diachi    NVARCHAR(100)     NULL,  
    dienthoai NVARCHAR(10)      NULL,  
    CONSTRAINT CK_nhanvien_dienthoai  
    CHECK (dienthoai LIKE '[0-9][0-9][0-9][0-9][0-9][0-9]____')  
    hsluong   DECIMAL(3,2)      DEFAULT (1.92)  
    madonvi   NVARCHAR(10)      NOT NULL  
)
```

b. Ràng buộc PRIMARY KEY:

- Chỉ định khoá chính của bảng.

Cú pháp:

```
[CONSTRAINT tên_ràng_buộc]  
PRIMARY KEY [(danh_sách_cột)]
```

Lưu ý:

- Mỗi bảng có nhiều nhất một ràng buộc PRIMARY KEY.
- Một khoá chính có thể bao gồm nhiều cột nhưng không vượt quá 16 cột.

Ví dụ 8.3:

Tạo bảng NHANVIEN như ví dụ 8.2, với khoá chính là MANV

* Hướng dẫn:

```
CREATE TABLE nhanvien
(
    manv      NVARCHAR(10)      NOT NULL,
    hoten     NVARCHAR(50)      NOT NULL,
    ngaysinh   DATE             NULL
        CONSTRAINT CK_nhanvien_ngaysinh
        CHECK (ngaysinh < '1/1/1990'),
    diachi    NVARCHAR(100)      NULL,
    dienthoai NVARCHAR(10)      NULL,
    CONSTRAINT CK_nhanvien_dienthoai
    CHECK (dienthoai LIKE '[0-9][0-9][0-9][0-9][0-9]_____')
    hsluong   DECIMAL(3,2)      DEFAULT (1.92)
    madonvi   NVARCHAR(10)      NOT NULL,
    CONSTRAINT PK_nhanvien_manv PRIMARY KEY
)
```

c. Ràng buộc UNIQUE:

- Chỉ định khoá phụ cho bảng.

Cú pháp:

```
[CONSTRAINT tên_ràng_buộc]
    UNIQUE [(danh_sách_cột)]
```

Ví dụ 8.4:

Tạo bảng NHANVIEN như ví dụ 8.3, trong đó không cho phép các nhân viên khác nhau được trùng điện thoại với nhau.

* Hướng dẫn:

```
CREATE TABLE nhanvien
(
    manv      NVARCHAR(10)      NOT NULL,
    hoten     NVARCHAR(50)      NOT NULL,
    ngaysinh   DATE             NULL
        CONSTRAINT CK_nhanvien_ngaysinh
        CHECK (ngaysinh < '1/1/1990'),
    diachi    NVARCHAR(100)      NULL,
    dienthoai NVARCHAR(10)      NULL,
    CONSTRAINT CK_nhanvien_dienthoai
    CHECK (dienthoai LIKE '[0-9][0-9][0-9][0-9][0-9]_____')
    hsluong   DECIMAL(3,2)      DEFAULT (1.92)
    madonvi   NVARCHAR(10)      NOT NULL,
    CONSTRAINT PK_nhanvien_manv PRIMARY KEY,
    CONSTRAINT UNIQUE_nhanvien_dienthoai UNIQUE(dienthoai)
)
```

d. Ràng buộc FOREIGN KEY (khoá ngoại)

- Một cột (hay một tập các cột) trong một bảng được gọi là khoá ngoại (ràng buộc FOREIGN KEY) nếu giá trị của nó được xác định từ khoá chính (PRIMARY KEY) hoặc khoá phụ (UNIQUE) của một bảng dữ liệu khác.

Cú pháp:

```
[CONSTRAINT tên_ràng_buộc]
```

```

FOREIGN KEY [(danh_sách_cột)]
REFERENCES tên_bảng_tham_chiếu(danh_sách_cột_tham_chiếu)
[ON DELETE CASCADE | NO ACTION | SET NULL | SET DEFAULT]
[ON UPDATE CASCADE | NO ACTION | SET NULL | SET DEFAULT]

```

Trong đó:

- CASCADE: Tự động xoá (cập nhật) nếu bản ghi được tham chiếu bị xoá (cập nhật).
- NO ACTION (Mặc định): Nếu bản ghi trong bảng tham chiếu đang được tham chiếu bởi một bản ghi bất kỳ trong bảng được định nghĩa thì bản ghi đó không được phép xoá hoặc cập nhật (đối với cột được tham chiếu).
- SET NULL: Cập nhật lại khoá ngoài của bản ghi thành giá trị NULL (nếu cột cho phép nhận giá trị NULL).
- SET DEFAULT: Cập nhật lại khoá ngoài của bản ghi nhận giá trị mặc định (nếu cột có qui định giá trị mặc định).

Ví dụ 8.5:

Tạo bảng NHANVIEN như ví dụ 8.4, trong đó khoá ngoài trên cột MADONVI (bảng DONVI). Giả sử rằng bảng DONVI đã được định nghĩa.

* Hướng dẫn:

```

CREATE TABLE nhanvien
(
    manv      NVARCHAR(10)      NOT NULL,
    hoten     NVARCHAR(50)      NOT NULL,
    ngaysinh   DATE             NULL
        CONSTRAINT CK_nhanvien_ngaysinh
        CHECK (ngaysinh < '1/1/1990'),
    diachi    NVARCHAR(100)     NULL,
    dienthoai NVARCHAR(10)      NULL,
    CONSTRAINT CK_nhanvien_dienthoai
    CHECK (dienthoai LIKE '[0-9][0-9][0-9][0-9][0-9]____')
    hsluong   DECIMAL(3,2)      DEFAULT (1.92)
    madonvi   NVARCHAR(10)      NOT NULL,
    CONSTRAINT PK_nhanvien_manv PRIMARY KEY,
    CONSTRAINT UNIQUE_nhanvien_dienthoai UNIQUE(dienthoai),
    CONSTRAINT FK_nhanvien_madonvi
        FOREIGN KEY(madonvi)
        REFERENCES donvi(madonvi)
        ON DELETE CASCADE
        ON UPDATE CASCADE
)

```

3. Sửa cấu trúc bảng

a. Cú pháp

```

ALTER TABLE tên_bảng
    ADD định_nghĩa_cột |
    ALTER COLUMN tên_cột kiểu_dữ_liệu [NULL | NOT NULL] |
    DROP COLUMN tên_cột |
    ADD CONSTRAINT tên_ràng_buộc định_nghĩa_ràng_buộc |
    DROP CONSTRAINT tên_ràng_buộc

```

b. Ví dụ 8.6

Bổ sung vào bảng DONVI ở ví dụ 8.5, cột GHICHU

* Hướng dẫn:

```
ALTER TABLE nhanvien  
ADD  
    ghichu NVARCHAR(50)
```

4. Xóa bảng

a. Cú pháp

```
DROP TABLE tên_bảng
```

b. Ví dụ 8.7:

Xoá bảng DONVI ra khỏi CSDL.

Lưu ý: Cột MADONVI trong bảng DONVI đang được tham chiếu bởi khoá ngoài FK_nhanvien_madonvi trong bảng NHANVIEN.

* Hướng dẫn:

- Xoá bỏ ràng buộc FK_nhanvien_madonvi khỏi bảng NHANVIEN

```
ALTER TABLE nhanvien
```

```
DROP CONSTRAINT FK_nhanvien_madonvi
```

- Xoá bảng DONVI:

```
DROP TABLE donvi
```

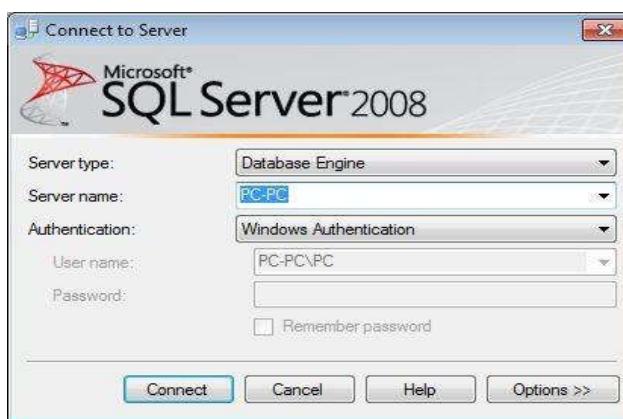
Thực hành

1. Khởi động SQL SERVER :

a. **Start à All programs à Microsoft SQL Server à SQL Server Management Studio**.

Chứng thực: chọn 1 trong 2 chế độ

- ✓ Là của Windows
- ✓ Là của SQL Server



c. Chọn **Connect**

2. Tạo database:

Ở cửa sổ Object Explorer, click chuột phải vào **Database** sau đó chọn **New database**

- Database name: **banhang**
- Chọn **Ok**

3. Tạo cấu trúc table:

a. Tạo table **Nhanvien**

* Ở cửa sổ Object Explorer:

- Nhấp dấu + trước database **banhang** (thành dấu -)
- Click chuột phải vào **Table** sau đó chọn **New table**
- Khai báo cấu trúc: **Ví dụ 8.1**

- Lưu: nhấp 

b. Tạo table Donvi : tự thực hiện.

4. Tạo ràng buộc cho table

* Chọn table Nhanvien

a. Ràng buộc CHECK

Ví dụ 8.2

* Ở cửa sổ hiển thị cấu trúc của table Nhanvien

- Click chuột phải vào field **ngaysinh**, chọn **CheckConstraints ...**

- Nhấp **Add**

+ (Name): gõ CK_nhanvien_ngaysinh

Expression: gõ **([ngaysinh] < '1/1/1990')**

- Nhấp **Add** (lần 2)

+ (Name): gõ CK_nhanvien_dienthoai

+ Expression: gõ **(dienthoai LIKE '[0-9][0-9][0-9][0-9][0-9][0-9]_____')**

- Nhấp **Close**

* Chọn table Donvi: tự thực hiện cho field dienthoai.

b. Ràng buộc PRIMARY KEY

* Chọn table Nhanvien

Ví dụ 8.3

* Ở cửa sổ hiển thị cấu trúc của table Nhanvien

- Click phải chuột vào field **manv** và chọn **Set Primary Key**

* Cho table Donvi: tự thực hiện cho field madonvi .

c. Ràng buộc UNIQUE

* Chọn table Nhanvien

Ví dụ 8.4

* Ở cửa sổ hiển thị cấu trúc của table Nhanvien

- Click phải chuột vào field **dienthoai** và chọn **Indexes / Keys...**

- Nhấp **Add**

+ (Name): gõ **UNIQUE_nhanvien_dienthoai**

+ Columns: dienthoai (ASC)

+ Type: Unique Key

- Nhấp **Close**

5. Tạo quan hệ (relationship) cho các table: Donvi, Nhanvien

Ví dụ 8.5

* Ở cửa sổ Object Explorer

- Click chuột phải vào **Database Diagrams** sau đó chọn **New database Diagram**

- Chọn các table tham gia tạo quan hệ, nhấp **Close**.

- Rê field madonvi (từ table nhanvien) thả vào field madonvi (của table donvi).

- Khai báo:

Primary key table	Foreign key table
Donvi	Nhanvien
madonvi	madonvi

6. Thêm field GHICHU vào table DONVI

Ví dụ 8.6

* Ở cửa sổ hiển thị cấu trúc của table Donvi

- Khai báo (ở cuối) thêm field: ghichu, nvarchar(50), NULL

7. Nhập dữ liệu cho các table: Donvi, Nhanvien

* Ở cửa sổ Object Explorer:

- Click chuột phải vào **dbo.donvi / dbo.nhanvien** sau đó chọn **Edit top 200 Rows**
- Nhập dữ liệu

MADONVI	TENDONVI	DIENTHOAI				
01	Phòng Kế toán	821451				
02	Phòng Tổ chức	831414				
03	Phòng điều hành	823351				
04	Phòng đối ngoại	841457				
05	Phòng Tài vụ	821451				
MANV	HOTEN	NGAYSINH	DIACHI	DIENTHOAI	HSLUONG	MADONVI
NV01001	Nguyễn Thị Hoa	05/05/1976	56 Lê Duẩn	521304	2.11	01
NV02001	Lê Hoài Nam	03/05/1976	32 Trần Phú	823145	1.86	02
NV02002	Hoàng Nam Phong	05/08/1971	66 Hoàng Diệu	521247	3.21	02
NV03001	Trần Nguyên Phong	20/12/1976	7 Hà Nội	849290	1.92	03
NV03002	Nguyễn Hữu Tinh	18/08/1976	7 Hà Nội	849290	1.92	03
NV05001	Nguyễn Trung Kiên	14/05/1972	77 Nguyễn Huệ	823236	1.86	05

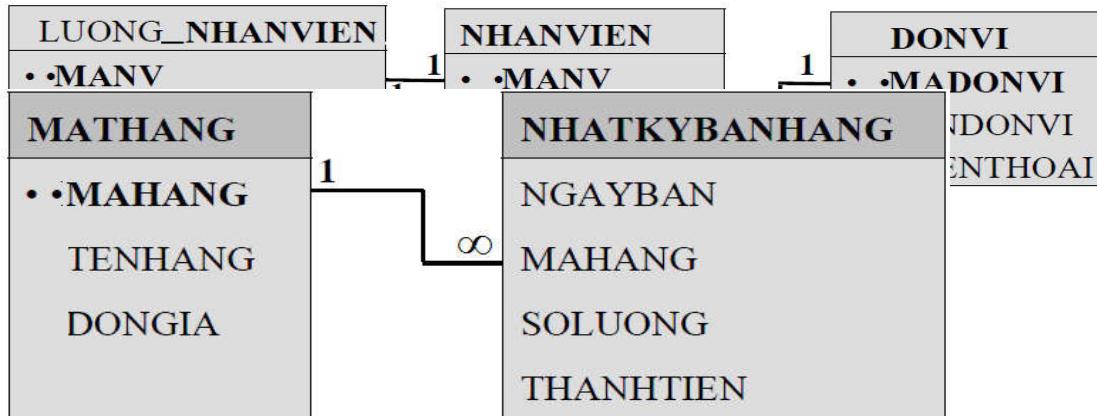
-- oOo --

Bài 9: CÁC CÂU LỆNH TRUY VẤN

* Xét CSDL **banhang** gồm các table:

NHANVIEN, DONVI, MATHANG, NHATKYBANHANG

- Câu trúc:



- Dữ liệu:

Bảng NHANVIEN

MANV	HOTEN	NGAYSINH	DIACHI	DIENTHOAI	HSLUONG	MADONVI
NV01001	Nguyễn Thị Hoa	05/05/1976	56 Lê Duẩn	521304	2.11	01
NV02001	Lê Hoài Nam	03/05/1976	32 Trần Phú	823145	1.86	02
NV02002	Hoàng Nam Phong	05/08/1971	66 Hoàng Diệu	521247	3.21	02
NV03001	Trần Nguyên Phong	20/12/1976	7 Hà Nội	849290	1.92	03
NV03002	Nguyễn Hữu Tinh	18/08/1976	7 Hà Nội	849290	1.92	03
NV05001	Nguyễn Trung Kiên	14/05/1972	77 Nguyễn Huệ	823236	1.86	05

Bảng DONVI

MADONVI	TENDONVI	DIENTHOAI
01	Phòng Kế toán	821451
02	Phòng Tổ chức	831414
03	Phòng điều hành	823351
04	Phòng đối ngoại	841457
05	Phòng Tài vụ	821451

IV. Các câu lệnh truy vấn

1. Câu lệnh SELECT

Công dụng:

- Chọn các field từ bảng.

Cú pháp:

```
SELECT [ALL | DISTINCT][TOP n] danh_sách_chọn
      [INTO tên_bảng_mới]
      FROM danh_sách_bảng [ | khung_nhìn]
      [WHERE điều_kiện]
      [GROUP BY danh_sách_cột]
      [HAVING điều_kiện]
      [ORDER BY cột_sắp_xếp]
      [COMPUTE danh_sách_hàm_gộp [BY danh_sách_cột]]
```

Chú ý: Các thành phần trong câu lệnh SELECT phải được sử dụng theo đúng thứ tự trên.

c. Danh sách chọn trong câu lệnh SELECT

- Chọn tất cả các field

Ví dụ 9.1a:

Câu lệnh sau đây cho biết thông tin của các nhân viên trong bảng NHANVIEN.

```
SELECT * FROM nhanvien n1
```

- Đổi tên các cột trong kết quả

Ví dụ 9.1b:

Câu lệnh sau đây cho biết HOTEN (họ tên) được đổi thành **Họ và Tên**, DIACHI (địa chỉ) được đổi thành **Địa Chỉ** của các nhân viên trong bảng NHANVIEN.

```
SELECT 'Họ và Tên' = hoten, 'Địa Chỉ' = diachi
      FROM nhanvien
```

Hoặc:

```
SELECT hoten 'Họ và Tên', diachi 'Địa Chỉ'
      FROM nhanvien
```

- Sử dụng cấu trúc CASE để đổi tên các cột trong kết quả

Ví dụ 9.1c:

Câu lệnh sau đây cho biết HOTEN (họ tên), HSLUONG (hệ số lương) và (xếp loại lương) của các nhân viên trong bảng NHANVIEN theo HSLUONG (hệ số lương).

```
SELECT 'Họ và Tên'= hoten, 'Hệ số lương' = hsluong, 'Hệ số lương' =
CASE
    WHEN hsluong = NULL          THEN 'Không xác định'
    WHEN hsluong <= 1.92         THEN 'Lương thấp'
    WHEN hsluong <= 3.11         THEN 'Lương trung bình'
    WHEN hsluong <= 5.2          THEN 'Lương cao'
    ELSE 'Lương rất cao'
END
```

```
FROM nhanvien
```

- Thêm chuỗi ký tự trong kết quả

Ví dụ 9.1d:

Câu lệnh sau đây sẽ cho thêm chuỗi 'Hệ số lương là:' ở trước cột HSLUONG (hệ số lương) trong từng dòng kết quả.

```
SELECT 'Họ và Tên'= hoten, 'Hệ số lương là:', 'Hệ số lương' = hsluong
      FROM nhanvien
```

- Tính toán các giá trị trong câu lệnh SELECT

Ví dụ 9.1e:

Câu lệnh sau đây sẽ cho HOTEN (họ tên) và LUONG (lương) của nhân viên theo công thức LUONG = HSLUONG * 730000.

```
SELECT 'Họ và Tên' = hoten, 'Lương' = hsluong * 730000  
FROM nhanvien
```

- Từ khóa DISTINCT: dùng để loại bỏ những dòng dữ liệu có kết quả giống nhau

Ví dụ 9.1f:

Câu lệnh sau sẽ cho các giá trị hsluong khác nhau trong bảng NHANVIEN

```
SELECT hsluong  
FROM nhanvien
```

- Tạo bảng mới bằng câu lệnh SELECT ... INTO

Ví dụ 9.1g:

Câu lệnh sau sẽ tạo bảng có tên NHANVIEN_LUU gồm các field HOTEN (họ tên), DIACHI (địa chỉ) của các nhân viên có HSLUONG > 1.92 từ bảng NHANVIEN.

```
SELECT hoten, diachi  
INTO nhanvien_luu  
FROM nhanvien  
WHERE hsluong > 1.92
```

- Sắp xếp kết quả (ASC: tăng, DESC: giảm) bằng ORDER BY

Ví dụ 9.1h:

Câu lệnh sau đây sẽ sắp xếp các nhân viên theo thứ tự giảm dần của HSLUONG (hệ số lương), nếu HSLUONG bằng nhau thì sắp xếp kết quả theo thứ tự tăng dần của NGAYSINH (ngày sinh)

```
SELECT hoten, ngaysinh, hsluong  
FROM nhanvien  
ORDER BY hsluong DESC, ngaysinh ASC
```

d. Xác định bảng bằng mệnh đề FROM

```
FROM danh_sách_bảng [ | khung_nhìn]
```

Ví dụ 9.1i:

Câu lệnh sau đây cho biết HOTEN (họ tên), DIACHI (địa chỉ) của các nhân viên bằng cách gán bí danh (alias) cho bảng NHANVIEN.

```
SELECT hoten, diachi  
FROM nhanvien n1
```

e. Đặt điều kiện truy xuất dữ liệu bằng mệnh đề WHERE

```
WHERE điều_kiện
```

điều_kiện: sử dụng các phép toán sau

- + So sánh: = , > , < , >= , <= , <> , != , !=
- + Giới hạn: BETWEEN ... AND ... , NOT BETWEEN ... AND ...
- + Danh sách: IN , NOT IN
- + Khuôn dạng: LIKE , NOT LIKE
 - With các ký tự đại diện: % , _ , [] , [^]
- + Các giá trị chưa biết: IS NULL , IS NOT NULL
- + Kết hợp các điều kiện: AND , OR

Ví dụ 9.1j:

Câu lệnh sau đây cho biết HOTEN (họ tên), DIACHI (địa chỉ) và DIENTHOAI (điện thoại) của các nhân viên có HSLUONG (hệ số lương) lớn hơn 1.92

```
SELECT hoten, diachi  
FROM nhanvien  
WHERE hsluong > 1.92
```

Ví dụ 9.1k:

Câu lệnh sau đây cho biết HOTEN (họ tên), DIACHI (địa chỉ) và DIENTHOAI (điện thoại) của các nhân viên có HSLUONG (hệ số lương) trong khoảng 1.92 đến 3.11

```
SELECT hoten, diachi  
FROM nhanvien  
WHERE hsluong BETWEEN 1.92 AND 3.11
```

Ví dụ 9.1l:

Câu lệnh sau đây cho biết HOTEN (họ tên), DIACHI (địa chỉ) và DIENTHOAI (điện thoại) của các nhân viên có HSLUONG (hệ số lương) là 1.86, 1.92, 2.11

```
SELECT hoten, diachi  
FROM nhanvien  
WHERE hsluong IN (1.86, 1.92, 2.11)
```

* Hoặc:

```
SELECT hoten, diachi  
FROM nhanvien  
WHERE hsluong = 1.86 OR hsluong = 1.92 OR hsluong = 2.11
```

Ví dụ 9.1m:

Câu lệnh sau đây cho biết thông tin của nhân viên có tên là NAM.

```
SELECT *  
FROM nhanvien  
WHERE hoten LIKE '%NAM'
```

Ví dụ 9.1n:

Câu lệnh sau đây cho biết thông tin của nhân viên không có điện thoại.

```
SELECT *  
FROM nhanvien  
WHERE dienthoai IS NULL
```

2. Thêm dữ liệu INSERT

Công dụng:

- Thêm dòng dữ liệu (mẫu tin/record) vào bảng.

Cú pháp:

```
INSERT INTO tên_bảng  
[(danh_sách_cột)] VALUES(danh_sách_trị)
```

Ví dụ 9.2a:

Câu lệnh sau đây thêm một dòng dữ liệu vào bảng DONVI

```
INSERT INTO donvi  
VALUES('06', 'Phòng CTCT-HSSV', '821460')
```

Ví dụ 9.2b:

Câu lệnh sau đây thêm một dòng dữ liệu vào bảng NHANVIEN

```
INSERT INTO nhanvien  
VALUES('NV02003', 'Lê Thị Mai', '23/05/1972', NULL, '523312', 1.92, '02')
```

Ví dụ 9.2c:

Câu lệnh sau đây thêm một dòng dữ liệu vào bảng NHANVIEN nhưng chỉ điền dữ liệu vào một số cột.

```
INSERT INTO nhanvien(manv, hoten, diachi, madonvi)  
VALUES('NV05002', 'Nguyễn Thị Hạnh Dung', '56 Trần Phú', '05')
```

Ví dụ 9.2d: Thêm dữ liệu vào bảng với dữ liệu lấy từ bảng khác

Câu lệnh sau đây thêm dữ liệu vào bảng LUONG_NHANVIEN với dữ liệu lấy từ bảng NHANVIEN.

```
INSERT INTO luong_nhanvien  
SELECT manv, hoten, hsluong*730000 FROM nhanvien
```

3. Cập nhật dữ liệu UPDATE

Công dụng:

- Cập nhật dữ liệu trong các bảng.

Cú pháp:

```
UPDATE tên_bảng  
SET tên_cột = biểu_thức  
[ , ...  
, tên_cột_k = biểu_thức_k]  
[FROM danh_sách_bảng]  
[WHERE điều_kiện]
```

Ví dụ 9.3a:

Câu lệnh sau đây tăng HSLUONG (hệ số lương) thêm 0.2 cho các nhân viên có MADONVI là 04.

```
UPDATE nhanvien  
SET hsluong = hsluong + 0.2  
WHERE madonvi = '04'
```

Ví dụ 9.3b:

Câu lệnh sau đây sẽ cập nhật giá trị cho field THANHTIEN (thành tiền) trong bảng NHATKYBANHANG theo công thức THANHTIEN = SOLUONG * DONGIA.

```
UPDATE nhatkybanhang  
SET thanhtien = soluong * MATHANG.dongia  
FROM MATHANG  
WHERE nhatkybanhang.mahang = MATHANG.mahang
```

4. Xóa dữ liệu DELETE

Công dụng:

- Để xóa dữ liệu trong bảng.

Cú pháp:

```
DELETE FROM tên_bảng  
[FROM danh_sách_bảng]  
[WHERE điều_kiện]
```

Ví dụ 9.4:

Câu lệnh sau đây xoá khỏi bảng NHANVIEN những nhân viên làm tại đơn vị có SODIENTHOAI (số điện thoại) là '848484'

```
DELETE FROM nhanvien  
FROM donvi  
WHERE nhanvien.madonvi = donvi.madonvi AND donvi.dienthoai = '848484'
```

5. Xóa toàn bộ dữ liệu TRUNCATE

Công dụng:

- Để xóa toàn bộ dữ liệu trong bảng.

Cú pháp:

```
TRUNCATE TABLE tên_bảng
```

Ví dụ 9.5:

Câu lệnh sau xoá toàn bộ dữ liệu trong bảng LUONG_NHANVIEN

```
DELETE FROM luong_nhanvien
```

Tương đương câu lệnh
TRUNCATE TABLE luong_nhanvien

V. Một số hàm thường dùng trong SQL Server

1. Hàm ngày – giờ

a. Hàm DATEADD

Cú pháp:

DATEADD(datepart, number, date)

Datepart: tham số chỉ định thành phần sẽ được cộng thêm vào ngày *date*.

DatePart	Viết tắt
year	yy, yyyy
quarter	qq, q
month	mm, m
dayofyear	dy, y
day	dd, d
week	wk, ww
hour	hh
minute	mi, n
second	ss, s
milisecond	ms

Công dụng: Hàm trả về một giá trị kiểu DateTime bằng cách cộng thêm một khoảng giá trị là *number* vào ngày *date* được chỉ định.

b. Hàm DATEDIFF

Cú pháp:

DATEDIFF(datepart, startdate, enddate)

Công dụng: Hàm trả về khoảng thời gian giữa 2 giá trị kiểu ngày *startdate* và *enddate* tùy thuộc vào *datepart*.

Ví dụ:

DateDiff(year, '3/10/2003', '6/15/2010') à kết quả: 7

c. Hàm DATEPART

Cú pháp:

DATEPART(datepart, date)

Công dụng: Hàm trả về một số nguyên được trích ra từ thành phần được chỉ định bởi *datepart* trong giá trị ngày *date*.

Ví dụ:

DatePart(year, '6/15/2010') à kết quả: 2010

d. Hàm GETDATE

Cú pháp:

GETDATE()

Công dụng: Hàm trả về giá trị là ngày hiện tại.

e. Hàm DAY, MONTH, YEAR

Cú pháp:

DAY(date) / MONTH(date) / YEAR(date)

Công dụng: Hàm trả về giá trị là ngày / tháng / năm của ngày *date*.

Ví dụ:

Day('6/15/2010') à kết quả: 15

Month('6/15/2010') à kết quả: 6

Year('6/15/2010') à kết quả: 2010

2. Hàm chuỗi

a. Hàm LEFT

Cú pháp:

LEFT(string, n)

Công dụng: Hàm trích từ chuỗi *string* n ký tự tính từ bên trái.

b. Hàm RIGHT

Cú pháp:

RIGHT(string, n)

Công dụng: Hàm trích từ chuỗi *string* n ký tự tính từ bên phải.

c. Hàm SUBSTRING

Cú pháp:

SUBSTRING(string, m, n)

Công dụng: Hàm trích từ chuỗi *string* n ký tự tính từ ký tự thứ *m*.

d. Hàm LTRIM

Cú pháp:

LTRIM(string)

Công dụng: Hàm cắt bỏ khoảng trắng thừa bên trái chuỗi *string*.

e. Hàm RTRIM

Cú pháp:

RTRIM(string)

Công dụng: Hàm cắt bỏ khoảng trắng thừa bên phải chuỗi *string*.

f. Hàm LEN

Cú pháp:

LEN(string)

Công dụng: Hàm trả về độ dài của chuỗi *string*.

Thực hành

1. Khởi động SQL SERVER :

a. **Start** à All programs à Microsoft SQL Server à SQL Server Management Studio b.

Chứng thực

c. Chọn **Connect**

2. Ở cửa sổ Object Explorer, click phải chuột lên **banhang** và chọn lệnh **New Query**

3. Ở cửa sổ query, thực hiện các câu lệnh trong các Ví dụ ở trên.

-- oOO --