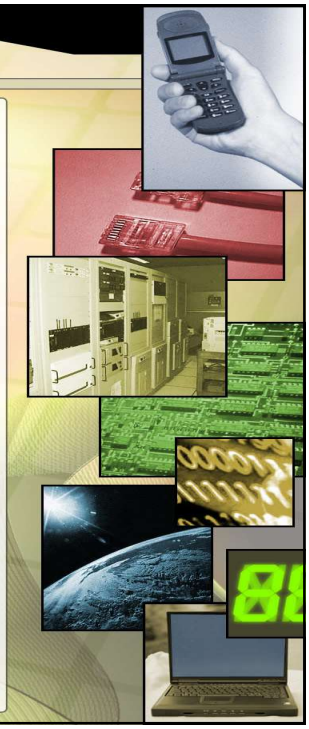
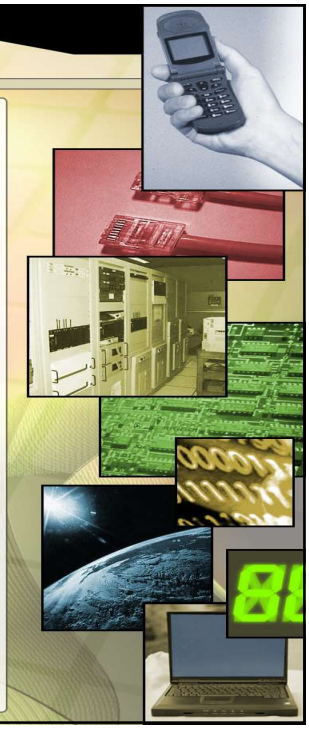
NGÔN NGỮ ĐỊNH NGHĨA DỮ LIỆU
DATA DEFINITION LANGUAGE - DDL



61

DDL

- **CREATE**: Tạo mới đối tượng CSDL
- **ALTER**: Thay đổi định nghĩa của đối tượng CSDL
- **DROP**: Xoá đối tượng CSDL



62

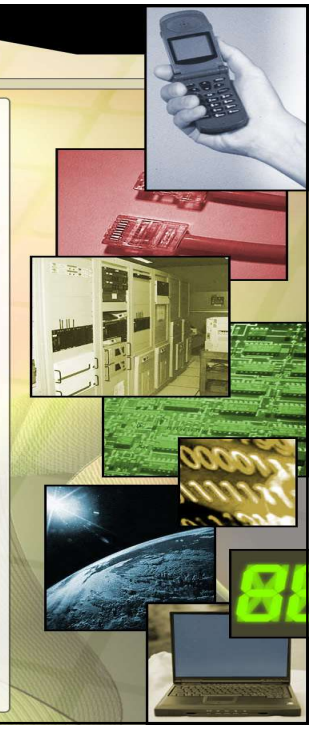
TẠO MỚI CSDL

- Tạo database mặc định

CREATE DATABASE *Tên_CSDL*;

Ví dụ:

```
IF EXISTS (SELECT *  FROM sys.DATABASES
           WHERE NAME = N'QLBanHang')
    --Xóa Database
    DROP DATABASE QLBanHang;
GO
--Tạo database mới
CREATE DATABASE QLBanHang;
```

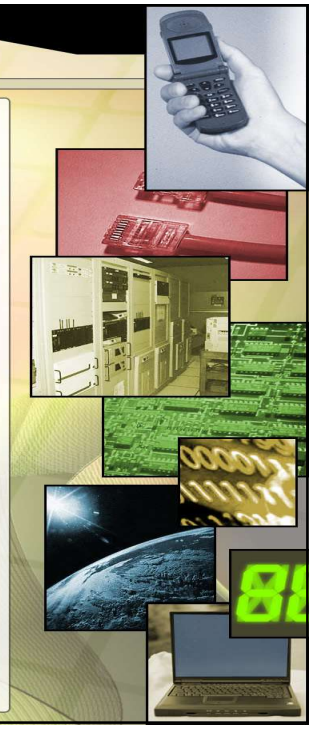


63

TẠO MỚI CSDL

- Tạo database theo yêu cầu

```
CREATE DATABASE Tên_CSDL
ON PRIMARY
(
    Name        = tên_file_logic
    , FileName  = 'tên_file_vật_lý'
    , Size      = kích_thước_ban_đầu [KB|MB|GB|TB]
    , MaxSize   = kích_thước_tối_đa [KB|MB|GB|TB]
    , FileGrowth= kích_thước_tăng_trưởng
)
LOG ON          -- Tạo tập tin log
(
    Name        = tên_file_logic
    , FileName  = 'tên_file_vật_lý'
    , Size      = kích_thước_ban_đầu [KB|MB|GB|TB]
    , MaxSize   = kích_thước_tối_đa [KB|MB|GB|TB]
    , FileGrowth= kích_thước_tăng_trưởng
);
```

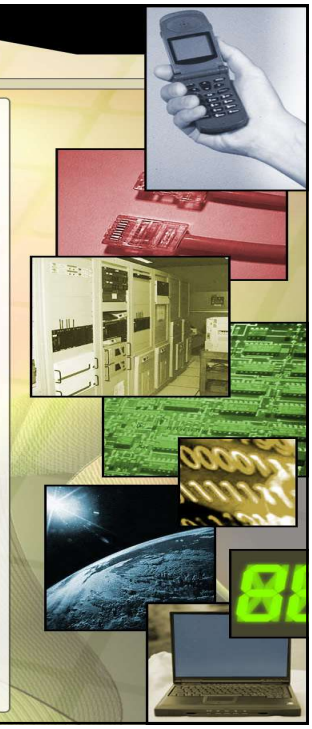


64

TẠO MỚI CSDL

Ví dụ:

```
create database QuanLyBanHang
on primary
(
  name = QuanLyBanHang_Data
  ,filename = 'c:\QuanLyBanHang.mdf'
  ,size = 200
  ,maxsize = Unlimited
  ,filegrowth = 10%      -- Tăng 10%
)
log on
(
  name = QuanLyBanHang_Log
  ,filename = 'c:\QuanLyBanHang.ldf'
  ,size = 100
  ,maxsize = 1GB
  ,filegrowth = 10       -- Tăng 10MB
);
```



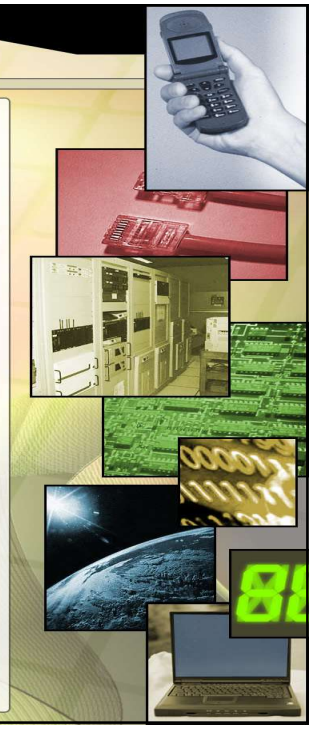
65

TẠO MỚI CSDL

Ví dụ:

```
create database Products ON
(
  name = prods_dat,
  filename = 'D:\BTSQL\prods.mdf',
  size = 5,
  maxsize = 500,
  filegrowth = 10
);
```

***Lưu ý**: Nếu không chỉ định transaction log file thì SQL Server sẽ tự động tạo ra 1 log file với kích thước ban đầu là 1MB*

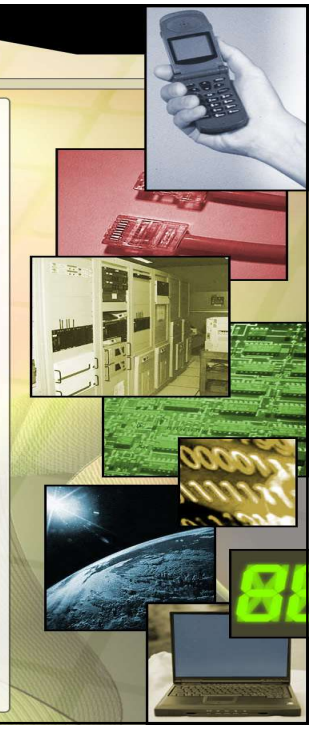


66

SỬ DỤNG CSDL

- Sử dụng (mở) CSDL để làm việc

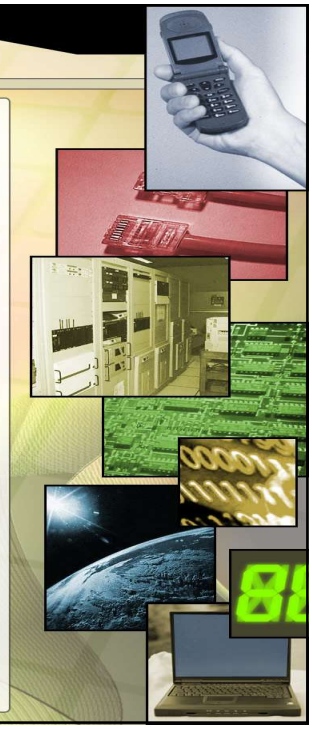
```
USE Tên_CSDL;
```



67

SỬA ĐỔI CSDL

```
ALTER DATABASE Tên_CSDL MODIFY FILE
(
   Name        = tên_file_logic
   [ , NewName  = 'tên_file_logic_sửa_đổi' ]
   [ , FileName = 'tên_file_vật_lý_sửa_đổi' ]
   [ , Size     = kích_thước_ban_đầu_sửa_đổi ]
   [ , MaxSize  = kích_thước_tối_đa_sửa_đổi ]
   [ , FileGrowth = kích_thước_tăng_trưởng_sửa_đổi ]
);
```



68

SỬA ĐỔI CSDL

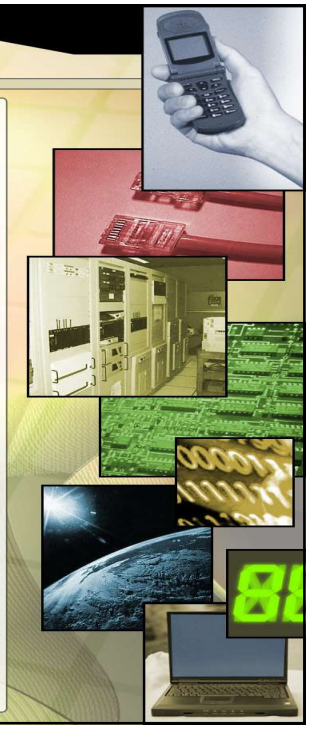
***Lưu ý**: Trong trường hợp sửa đổi file vật lý (đổi tên hoặc di chuyển file) thì cần làm theo các bước sau:*

- Sửa đổi thông số của CSDL như trên

```
ALTER DATABASE Tên_CSDL MODIFY FILE (...);
```

- Ngắt kết nối, thiết lập offline CSDL

```
ALTER DATABASE Tên_CSDL
  --đóng hết các connection tới database
  SET SINGLE_USER WITH ROLLBACK IMMEDIATE;
--đặt database thành offline
ALTER DATABASE Tên_CSDL SET OFFLINE;
```

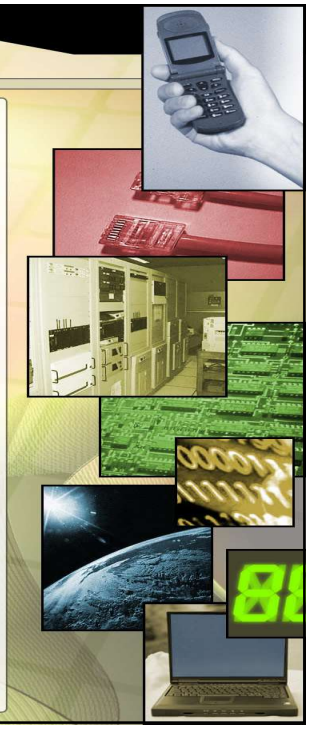


69

SỬA ĐỔI CSDL

- Sau khi sửa đổi CSDL (đổi tên, di chuyển file) thì kết nối CSDL với người dùng trở lại

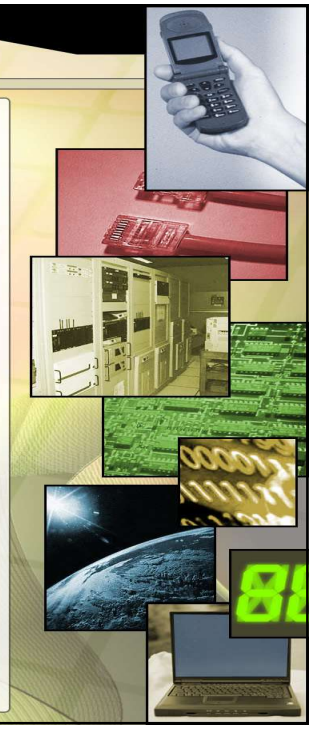
```
--đặt database trở lại online
ALTER DATABASE Tên_CSDL SET ONLINE;


--đặt trở lại chế độ nhiều người dùng
ALTER DATABASE Tên_CSDL SET MULTI_USER;
```



70

SỬA ĐỔI CSDL

Ví dụ: Di chuyển CSDL file (data file hoặc log file) sang vị trí mới.



71

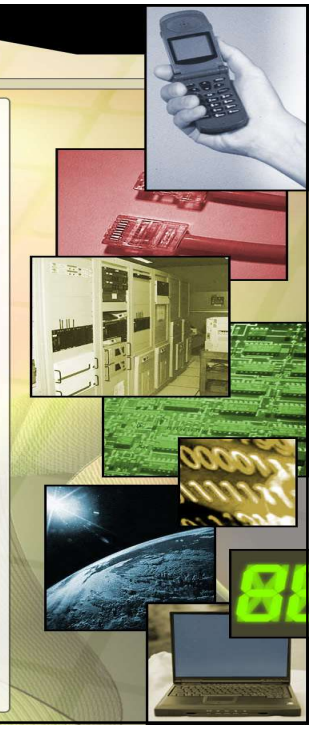
THU HẸP CSDL

- Thu hẹp toàn bộ CSDL

```
DBCC SHRINKDATABASE
( Tên_CSDL | id_CSDL | 0        -- 0 là CSDL mặc định
   [ , tỷ_lệ ]                  -- % khoảng trống còn lại
   [ , { NOTRUNCATE | TRUNCATEONLY } ]
);
```

Ví dụ:

```
dbcc shrinkdatabase(tempdb, 40);
```



72

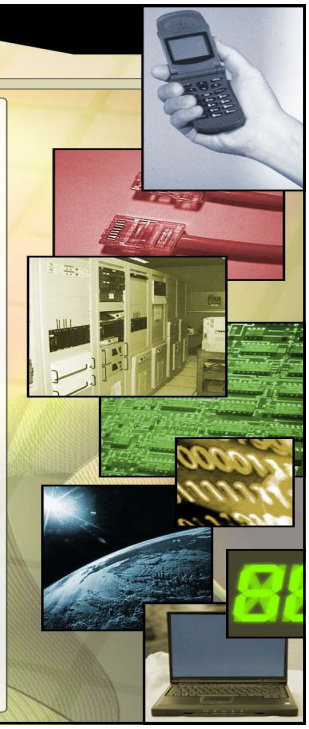
THU HẸP CSDL

- Thu hẹp file (cụ thể) trong CSDL

```
USE Tên_CSDL;
DBCC SHRINKFILE
( { tên_file | id_file }
  [ , EMPTYFILE ]
| [ [ , kích_thước_file ] [ , { NOTRUNCATE | TRUNCATEONLY } ] ]
) ;
```

Ví dụ:

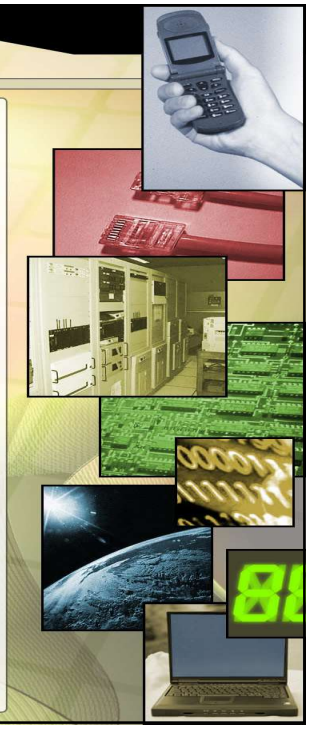
```
use QLSV;
dbcc shrinkfile (QLSV_Log, 1);
```



73

THÊM FILE VÀO CSDL

```
ALTER DATABASE Tên_CSDL
   ADD FILE [TO FILEGROUP Tên_nhóm] -- Thêm file dữ liệu .ndf
       | LOG FILE                   -- Thêm file log
   (
      Name       = tên_file_logic
      , FileName = 'tên_file_vật_lý'
      , Size     = kích_thước_ban_đầu [KB|MB|GB|TB]
      , MaxSize  = kích_thước_tối_đa [KB|MB|GB|TB]
      , FileGrowth = kích_thước_tăng_trưởng
   );
```

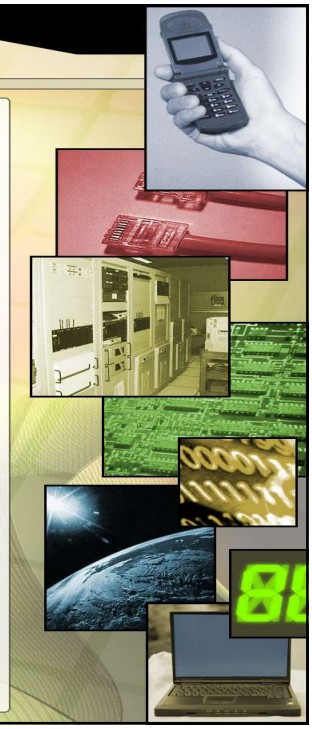


74

THÊM FILE VÀO CSDL

Ví dụ:

```
ALTER DATABASE QLSV
  ADD FILE
  (
      NAME = QLSV_data2,
      FILENAME = 'd:\qlsv_data2.ndf'
  );
```

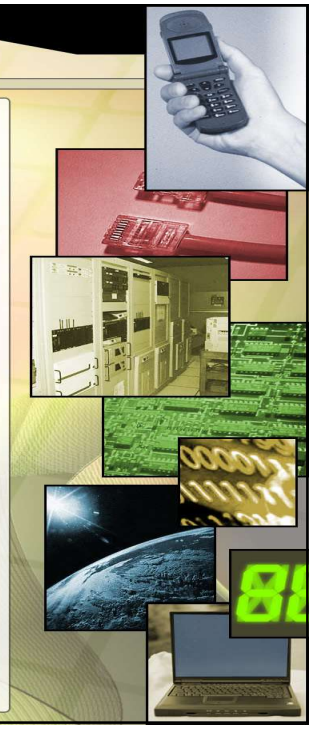


75

THÊM FILE VÀO CSDL

Ví dụ:

```
ALTER DATABASE QLSV
  ADD LOG FILE
  (
    NAME        = qlsvlog2,
    FILENAME    = 'D:\SQLServer\qlsvlog2.ldf',
    SIZE        = 5MB,
    MAXSIZE     = 100MB,
    FILEGROWTH  = 5%
  ),
  (
    NAME        = qlsvlog3,
    FILENAME    = 'D:\SQLServer\qlsvlog3.ldf',
    SIZE        = 5MB,
    MAXSIZE     = 100MB,
    FILEGROWTH  = 5MB
  );
```



76

GỠ BỎ FILE CSDL

USE *Tên_CSDL*;

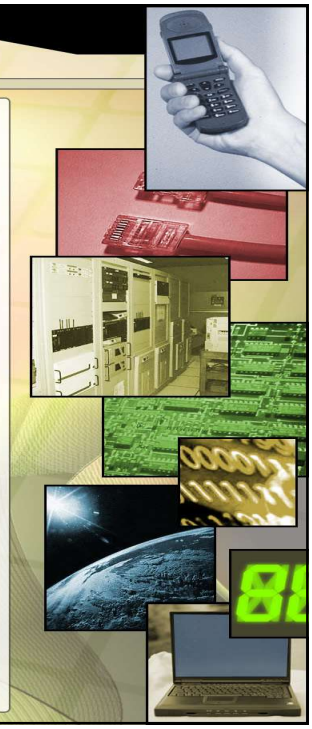
-- Làm trống file

DBCC SHRINKFILE (*Tên_file_logic*, **EMPTYFILE**);

-- Gỡ bỏ file

ALTER DATABASE *Tên_CSDL*

REMOVE FILE *Tên_file_logic*;



77

ĐỔI TÊN CSDL

- Cách 1

ALTER DATABASE *Tên_CSDL*

MODIFY NAME = *Tên_CSDL_mới*;

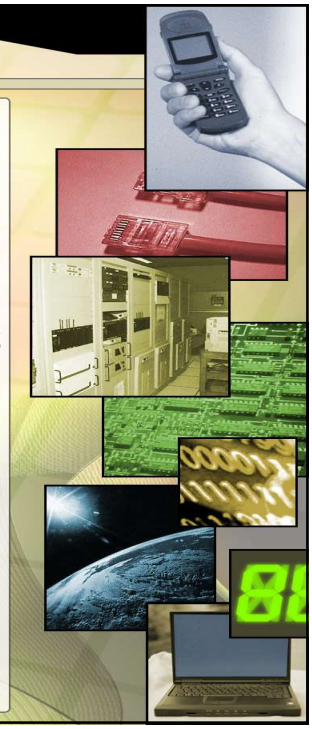
Ví dụ:
```
alter database QLSV modify name = QuanLySinhVien;
```

- Cách 2: sử dụng thủ tục **sp_renamedb** của hệ thống

sp_renamedb '*Tên_CSDL_cũ*' , '*Tên_CSDL_mới*'

Ví dụ:
```
sp_renamedb N'QLSV', N'QuanLySinhVien'
```

Lưu ý:
- *CSDL phải ở chế độ một người dùng*
- *Phải làm việc trong CSDL chính để thực thi thủ tục*

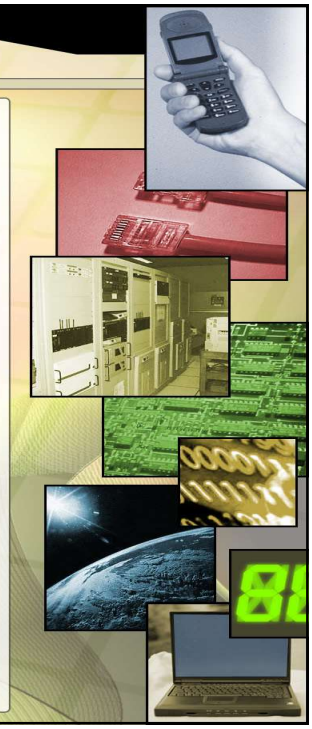


78

GỠ BỎ (XÓA) CSDL

- Gỡ bỏ thông tin của CSDL từ các bảng hệ thống và gỡ bỏ các tập tin nhật ký và dữ liệu từ hệ thống.
- CSDL bị xóa có thể khôi phục lại từ bản sao.
- Không một người dùng nào đang sử dụng CSDL tại thời điểm nó bị xóa.
- Phải là thành viên của db_owner (hoặc vai trò máy chủ sysadmin) để xóa CSDL
- **Cú pháp:**

DROP DATABASE *Tên_CSDL* [**,** *Tên_CSDL_2*] [**,** ...];

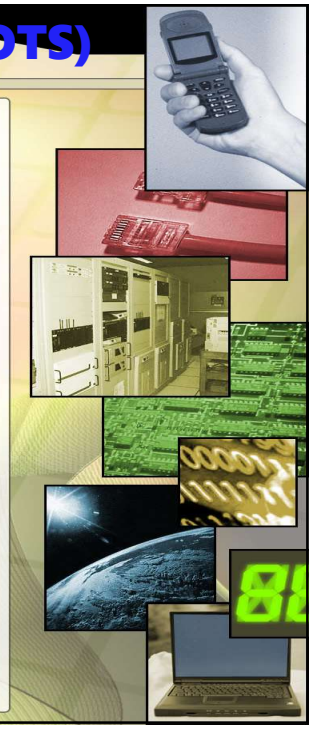


79

BẢN CHỤP CSDL (DATABASE SNAPSHOTS)

- Là bản chỉ đọc, khung nhìn tĩnh của 1 CSDL

CREATE DATABASE *Tên_bản_chụp_CSDL* **ON**
(
Name = tên_file_logic
, FileName = **'**tên_file_vật_lý**'**
) [, …] -- Các file bản chụp tiếp theo
AS SNAPSHOT OF *Tên_CSDL_gốc*;



80

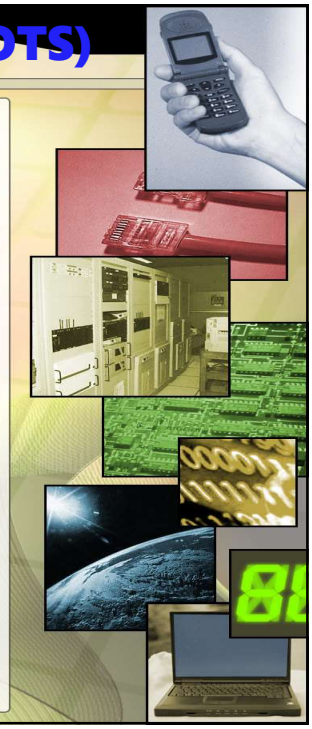
BẢN CHỤP CSDL (DATABASE SNAPSHOTS)

Ưu điểm

- Bản chụp cung cấp một bản copy chỉ đọc của dữ liệu.
- Khi một bản chụp được truy vấn, không làm giảm hiệu suất của đối tượng được quan sát.
- Các tập tin dữ liệu của bản chụp là nhỏ và được tạo ra rất nhanh. Nó chỉ lớn khi cơ sở dữ liệu là chủ thể thay đổi thường xuyên.
- Sử dụng chủ yếu cho mục đích báo cáo, bảo vệ dữ liệu khỏi các lỗi quản trị, các lỗi người dùng và quản lý các dữ liệu thử nghiệm

Nhược điểm

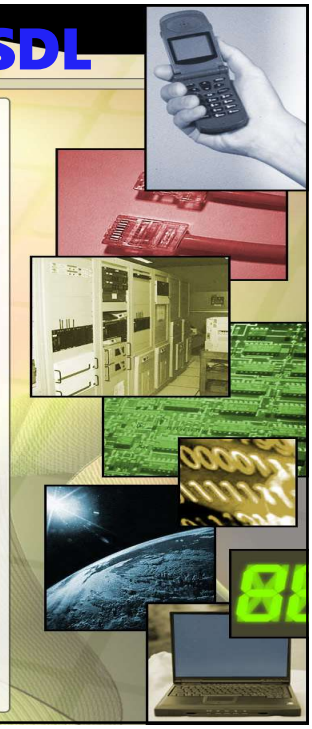
- Bản sao của bản chụp không thể được tạo
- Bản chụp phải tồn tại trên cùng máy chủ cơ sở dữ liệu cùng với cơ sở dữ liệu gốc.
- Một người sử dụng mới không thể được quyền truy cập vào dữ liệu trong bản chụp. Quyền được thừa kế từ cơ sở dữ liệu gốc khi nó đã tồn tại tại thời điểm tạo ra bản chụp.



81

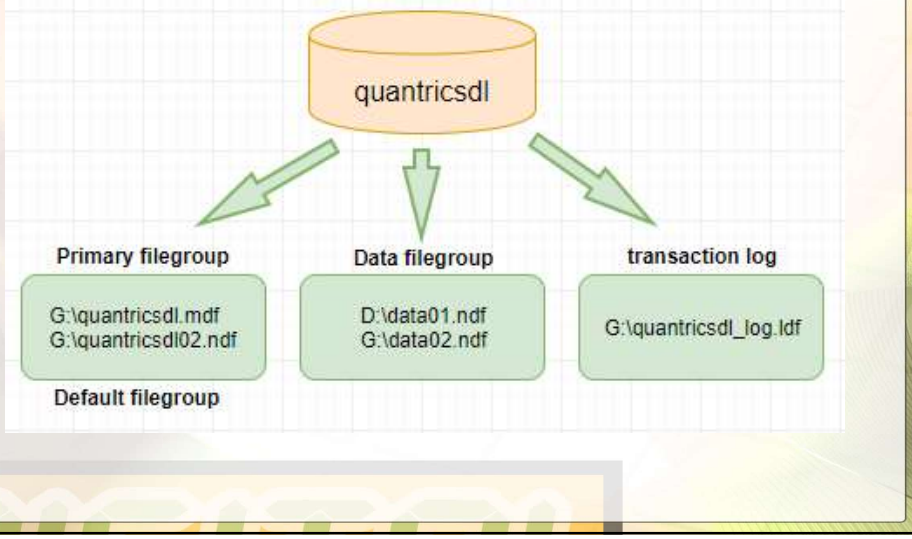
NHÓM TẬP TIN (FILEGROUP) CSDL

- Filegroup là tên đặt cho một nhóm data file trong SQL Server.
- Filegroup không chứa dữ liệu, mà chỉ định nghĩa ở mức logic các data file nằm trong đó (các data file mới thực sự chứa dữ liệu).
- Tương tự như tổ chức thư mục trong Windows.
- Filegroup mặc định trong SQL Server là **PRIMARY**



82

NHÓM TẬP TIN (FILEGROUP) CSDL



83

NHÓM TẬP TIN (FILEGROUP) CSDL

- Filegroup có thể được tạo cùng lúc với tạo CSDL hoặc thêm vào sau.
- Một file không thể thuộc nhiều hơn 1 filegroup tại cùng thời điểm.
- Không thể thay đổi filegroup của 1 file trong CSDL
- Filegroup chỉ có thể chứa file dữ liệu, KHÔNG chứa file log.

84

NHÓM TẬP TIN (FILEGROUP) CSDL

- Thêm filegroup khi tạo CSDL

CREATE DATABASE *Tên_CSDL*

[ON

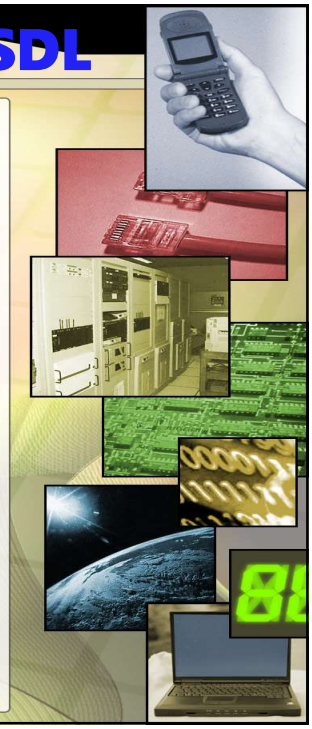
[PRIMARY] <*định_nghĩa_file_data*> **[, ...n]**

[, <FILEGROUP> [, ...n]]

[LOG ON <*định_nghĩa_file_log*> **[, ...n]]**

]

[;]

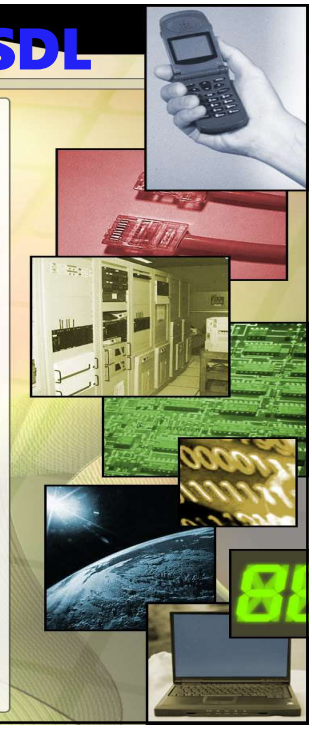


85

NHÓM TẬP TIN (FILEGROUP) CSDL

Ví dụ:

```
CREATE DATABASE MySQLDB ON
   PRIMARY
   (
      NAME           = 'MySQLDB_Data',
      FILENAME       = 'd:\baitap\mysqldb_data.mdf',
      SIZE           = 10,
      MAXSIZE        = UNLIMITED,
      FILEGROWTH     = 10%
   ),
   FILEGROUP my_Group
   (
      NAME           = 'Ext_Data',
      FILENAME       = 'd:\baitap\ext_data.ndf',
      SIZE           = 2 ,
      MAXSIZE        = UNLIMITED,
      FILEGROWTH     = 1
   )
   LOG ON
   (
      NAME           = 'MySQLDB_Log',
      FILENAME       ='d:\baitap\mysqldb_log.ldf',
   );
```



86

NHÓM TẬP TIN (FILEGROUP) CSDL

- Thêm filegroup vào CSDL đã có

ALTER DATABASE *Tên_CSDL*

ADD FILEGROUP *Tên_nhóm*

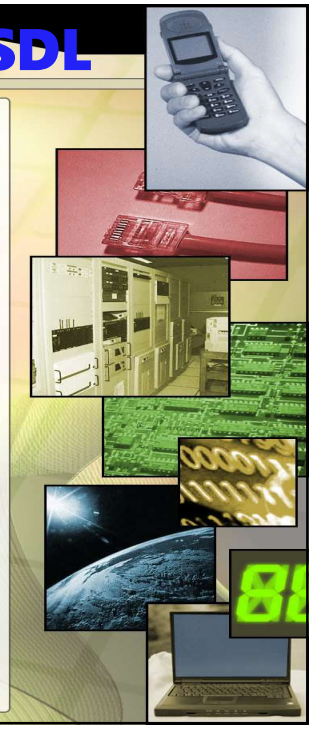
- Thêm file vào filegroup trong CSDL

ALTER DATABASE *Tên_CSDL*

ADD FILE

(<*định_nghĩa_file_data*> **[** , ...n **]** **)**

TO FILEGROUP *Tên_nhóm*;

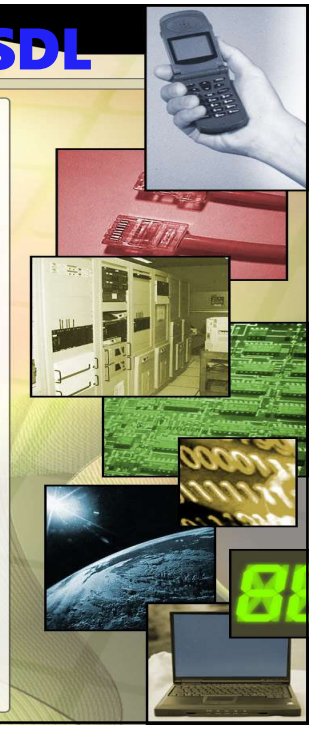


87

NHÓM TẬP TIN (FILEGROUP) CSDL

Ví dụ:

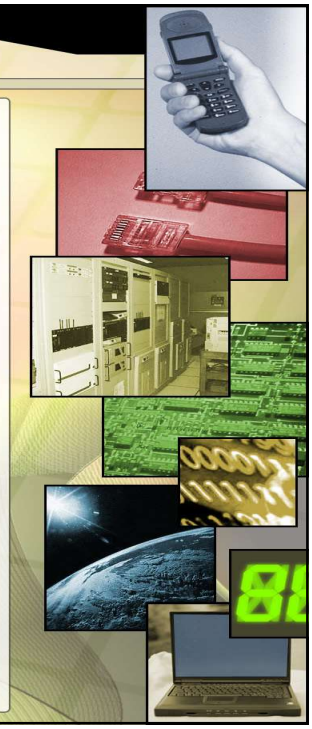
```
ALTER DATABASE QLSV
  ADD FILEGROUP DaiHoc;

ALTER DATABASE QLSV
  ADD FILE
  (
    NAME        = 'DaiHoc2019_Data',
    FILENAME    = 'd:\baitap\daihoc2019_data.ndf',
    SIZE        = 100,
    MAXSIZE     = UNLIMITED,
    FILEGROWTH  = 10%
  )
  TO FILEGROUP DaiHoc;
```



88

SCHEMA

- Một schema thuận lợi cho việc tách, quản lý, và quyền sở hữu của các đối tượng CSDL, nâng cao quản trị an ninh của các đối tượng CSDL.
- Có trong SQL Server 2005 trở lên
- Schema là một namespace đối với các đối tượng CSDL

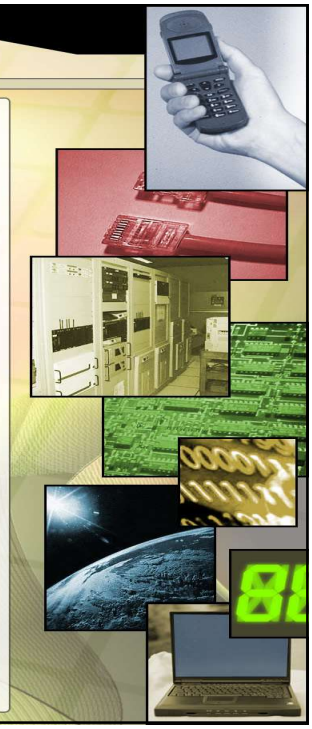


89

SCHEMA

- Trong một CSDL, tên của schema là duy nhất, luôn có tên với đường đầy đủ dạng

  server.database.schema.object
- Schema mặc định của các đối tượng trong CSDL là **dbo**
- **Ưu điểm**
  - Nhóm các đối tượng CSDL lại với nhau cho dễ quản lý
  - Cho phép phân quyền ở schema tăng tính bảo mật



90

SCHEMA

- Tạo schema

  CREATE SCHEMA *Tên_schema*;
- Xóa schema

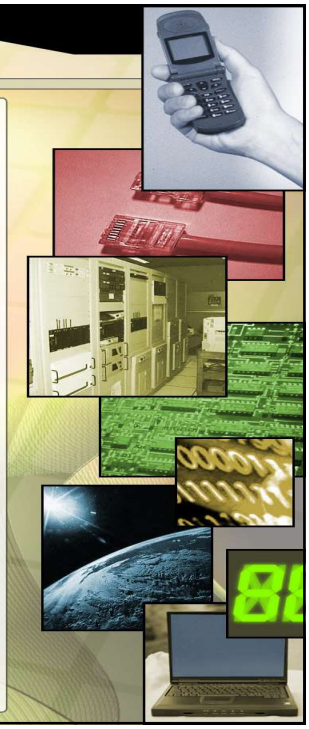
  DROP SCHEMA *Tên_schema*;
- Thay đổi schema cho đối tượng CSDL
  - Dành cho kiểu dữ liệu người dùng định nghĩa

    ALTER SCHEMA *Tên_schema_mới*

    TRANSFER TYPE :: *Tên_schema_cũ.User_Type*
  - Dành cho các đối tượng CSDL khác

    ALTER SCHEMA *Tên_schema_mới*

    TRANSFER OBJECT :: *Tên_schema_cũ.Đối_tượng*

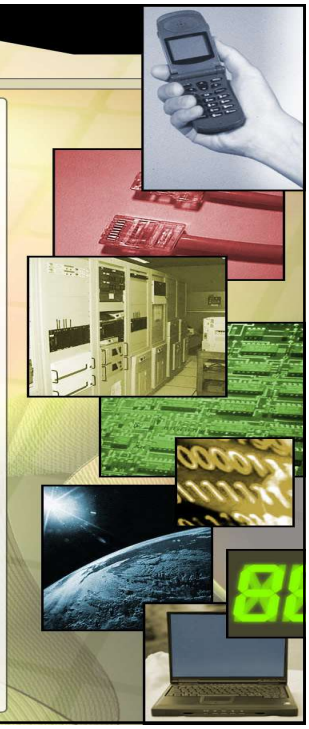


91

SCHEMA

Ví dụ:

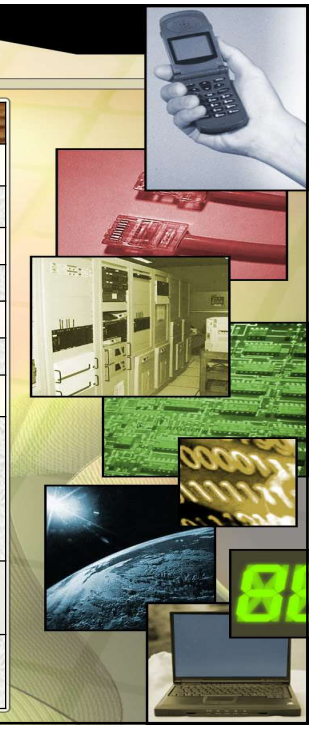
```
CREATE SCHEMA S1;
CREATE SCHEMA S2;
-- Tạo kiểu dữ liệu thuộc S1
CREATE TYPE S1.TestType FROM [varchar](10) NOT NULL;
-- Kiểm tra thông tin schema
SELECT sys.types.name, sys.types.schema_id, sys.schemas.name
    FROM sys.types JOIN sys.schemas
        ON sys.types.schema_id = sys.schemas.schema_id
    WHERE sys.types.name = 'TestType';
GO
-- Thay đổi kiểu dữ liệu TestType từ S1 sang S2
ALTER SCHEMA S2 TRANSFER type::S1.TestType;
```



92

KIỂU DỮ LIỆU

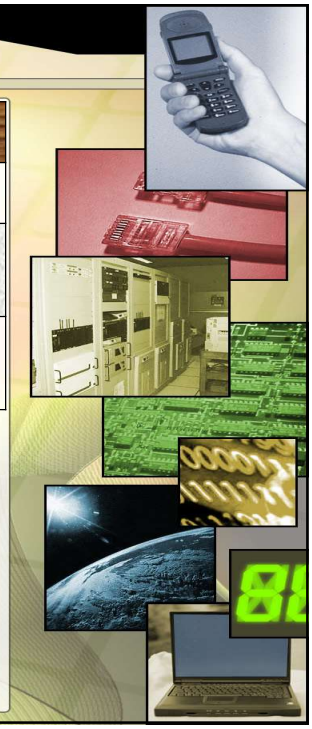
KIỂU DỮ LIỆU	KÍCH THƯỚC	MIỀN GIÁ TRỊ LƯU TRỮ
❖ Các kiểu dữ liệu dạng số nguyên		
BigInt	8 bytes	từ -2^{63} đến $2^{63}-1$
Int	4 bytes	từ -2^{31} đến $2^{31}-1$
SmallInt	2 bytes	từ -2^{15} đến $2^{15}-1$
TinyInt	1 byte	từ 0 đến 255
Bit	1 byte	0, 1 hoặc Null
❖ Các kiểu dữ liệu số thực		
Decimal[(p [, s])] **Dec[(p [,s])]** **Numeric[(p [, s])]**	5 – 17 bytes	▪ *p* là tổng số chữ số (mặc định là *18*), *s* là số chữ số thập phân (mặc định là *0*) ▪ Tối đa $-10^{38}+1$ đến $10^{38}-1$ (khi sử dụng 17 bytes)
Float[(n)]	4 – 8 bytes	Số thực dấu chấm (phẩy) động với phần định trị n bit (mặc định là *53* – tối đa)
Real	4 bytes	Tương đương với **Float(24)**. Số thực chính xác đơn.



93

KIỂU DỮ LIỆU

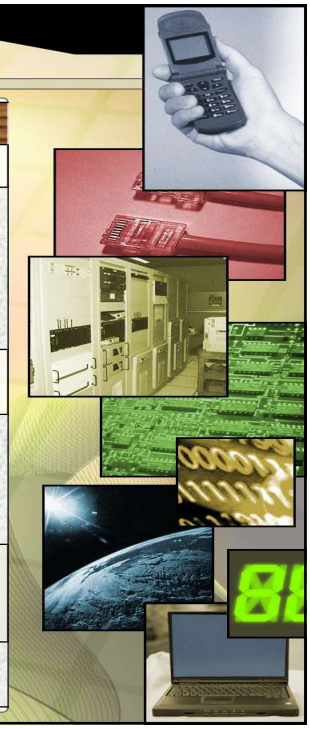
KIỂU DỮ LIỆU	KÍCH THƯỚC	MIỀN GIÁ TRỊ LƯU TRỮ
❖ Các kiểu dữ liệu tiền tệ (số)		
Money	8 bytes	từ – 922,337,203,685,477.5808 đến 922,337,203,685,477.5807
SmallMoney	4 bytes	từ – 214,748.3648 đến 214,748.3647



94

KIỂU DỮ LIỆU

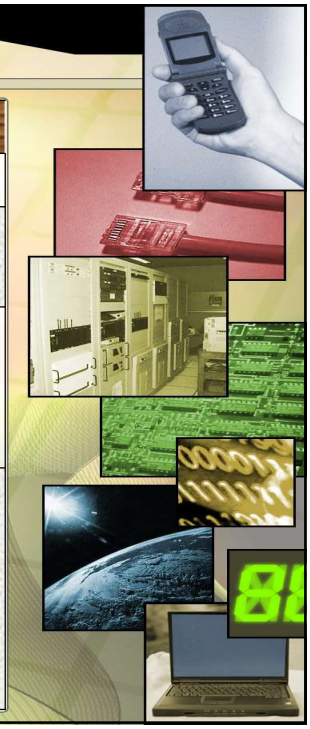
KIỂU DỮ LIỆU	KÍCH THƯỚC	MIỀN GIÁ TRỊ LƯU TRỮ
❖ **Các kiểu dữ liệu ký tự**		
Char [(n)]	Tối đa 8000 ký tự (mỗi ký tự 1 byte)	▪ **n** là số ký tự lưu trữ (mặc định **1**) ▪ Độ dài cố định ▪ Thêm dấu cách về bên phải để bù phần trống cho đủ số kí tự. ▪ Không chứa kí tự Unicode.
NChar [(n)]	Tối đa 4000 ký tự (mỗi ký tự 2 bytes)	Tương tự như **Char** nhưng chứa được ký tự Unicode
VarChar [(*n* | **max)]**	Tối đa 8000 ký tự (mỗi ký tự 1 byte) hoặc tính theo max	▪ **n** là số ký tự lưu trữ (mặc định **1**) ▪ Độ dài tùy biến. ▪ Nếu chỉ định **max** thì tối đa là 2GB ▪ Không chứa kí tự Unicode.
NVarChar [(*n* | **max)]**	Tối đa 4000 ký tự (mỗi ký tự 2 byte) hoặc tính theo max	Tương tự như **VarChar** nhưng chứa được ký tự Unicode
Binary [(n)] **NBinary [(n)]**	Tối đa 8000 ký tự (mỗi ký tự 1 byte)	Tương tự như **Char / VarChar** nhưng chỉ chứa giá trị nhị phân (mã của ký tự)



95

KIỂU DỮ LIỆU

KIỂU DỮ LIỆU	KÍCH THƯỚC	MIỀN GIÁ TRỊ LƯU TRỮ
❖ **Các kiểu dữ liệu thời gian**		
Date	3 bytes	▪ Hiển thị dưới dạng **YYYY-MM-DD** ▪ Ngày từ 0001-01-01 đến 9999-12-31
DateTime	8 bytes	▪ Hiển thị dưới dạng **YYYY-MM-DD hh:mm:ss[.mmm]** ▪ Ngày từ 1753-01-01 đến 9999-12-31 ▪ Giờ từ 00:00:00 đến 23:59:59.997
DateTime2 [(*ps*)]	Tối đa 8 bytes	▪ Hiển thị dưới dạng **YYYY-MM-DD hh:mm:ss[.***số_giây_thập_phân***]** ▪ Ngày từ 0001-01-01 đến 9999-12-31 ▪ Giờ từ 00:00:00 đến 23:59:59.9999999 ▪ **ps** – số chữ số chính xác của giây (mặc định là **7** – tối đa)



96

KIỂU DỮ LIỆU

KIỂU DỮ LIỆU	KÍCH THƯỚC	MIỀN GIÁ TRỊ LƯU TRỮ
❖ Các kiểu dữ liệu thời gian (tt)		
SmallDatetime	4 bytes	▪ Hiển thị dưới dạng **YYYY-MM-DD** ▪ Ngày từ 0001-01-01 đến 2079-06-06 ▪ Giờ từ 00:00:00 đến 23:59:59
Time [(*ps*)]	5 bytes	▪ **hh:mm:ss[.***số_giây_thập_phân***]** ▪ Giờ từ 00:00:00 đến 23:59:59.9999999 ▪ ps – số chữ số chính xác của giây (mặc định là **7** – tối đa)
DateTimeOffset [(*ps*)] [(*múi_giờ*)]	10 bytes	▪ Hiển thị dưới dạng **YYYY-MM-DD hh:mm:ss[.***số_giây_thập_phân***][{+|-}hh:mm]** ▪ Ngày từ 0001-01-01 đến 9999-12-31 ▪ Giờ từ 00:00:00 đến 23:59:59.9999999 ▪ **Múi giờ** từ -14:00 đến +14:00 ▪ ps – số chữ số chính xác của giây (mặc định là **7** – tối đa)

97

GIÁ TRỊ NULL

❖ Giá trị **NULL**: giá trị dữ liệu tồn tại trong CSDL có thể không xác định được do:

- Giá trị đó có tồn tại nhưng không biết.
- Không xác định được giá trị đó có tồn tại hay không.
- Tại một thời điểm nào đó giá trị chưa có nhưng rồi có thể sẽ có. Giá trị bị lỗi do tính toán (tràn số, chia cho không...).

➡ Được biểu diễn bởi các giá trị NULL.

✍ ***Lưu ý**: Phân biệt NULL với giá trị rỗng (kiểu ký tự - chuỗi) hoặc giá trị 0 (kiểu số)*

98

TẠO KIỂU DỮ LIỆU

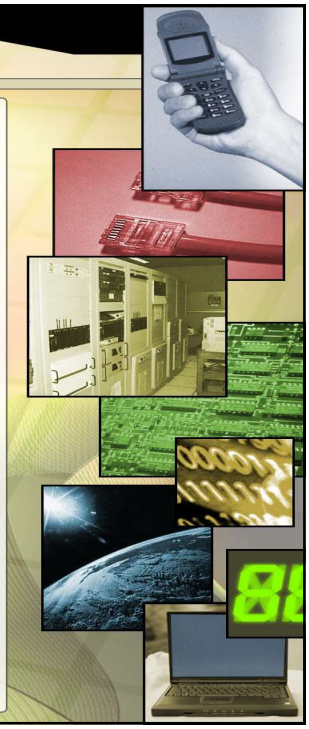
- Tạo kiểu dữ liệu đơn tự định nghĩa

CREATE TYPE [*Tên_schema.* **] Tên_kiểu**

FROM *Kiểu_dữ_liệu_gốc* [NULL | NOT NULL];

Ví dụ:

```
CREATE TYPE AgeType FROM TinyInt NOT NULL;
CREATE TYPE ntu.LocationType FROM VARCHAR(500);
```

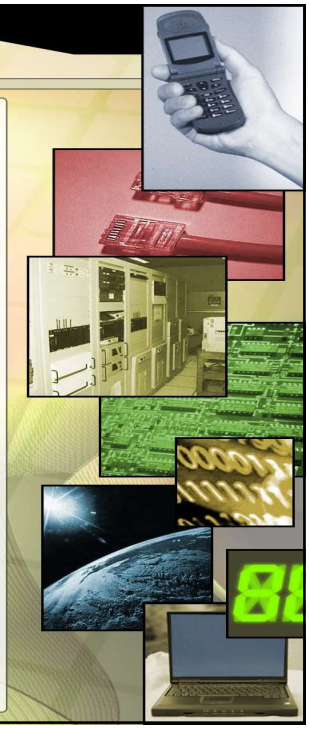


99

TẠO KIỂU DỮ LIỆU

- Tạo kiểu dữ liệu dạng bảng

CREATE TYPE [*Tên_schema.* **] Tên_kiểu**
AS TABLE (
Định_nghĩa_cột **[, ... n]**
[, { PRIMARY KEY | UNIQUE }
(*Tên_cột_khóa* [ASC | DESC] [, ...n])]
[, INDEX Tên_index
(*Tên_cột_index* [ASC | DESC] [, ...n])]
[, CHECK (*điều_kiện_logic*)]
);

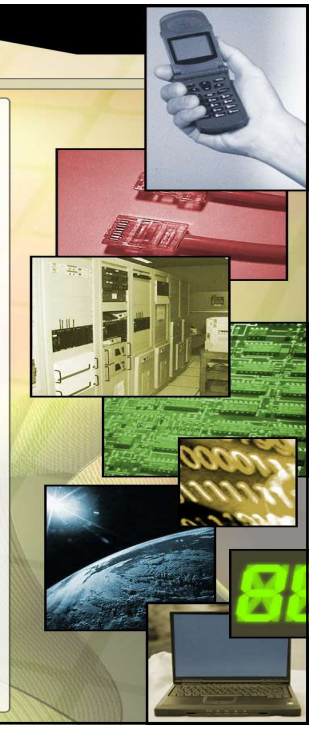


100

TẠO KIỂU DỮ LIỆU

Ví dụ:

```
CREATE TYPE Item_Trans AS TABLE
(
   ItemId NVARCHAR(10) NOT NULL,
   SupplierId INT NOT NULL,
   Price DECIMAL (18, 4) NULL,
   PRIMARY KEY (
     ItemId ASC,
     SupplierId DESC
   ),
   INDEX IX_Item_Price ( Price ),
   CHECK (Price >= 0)
);
```



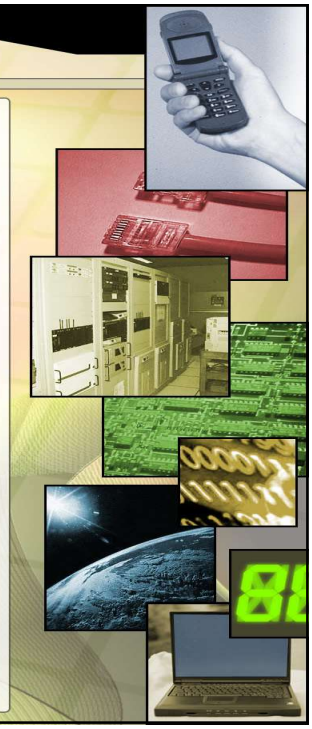
101

XÓA KIỂU DỮ LIỆU

DROP TYPE [IF EXISTS] [*Tên_schema.*] Tên_kiểu

;

Lưu ý*: *Không sử dụng ALTER để sửa đổi thông số trong kiểu dữ liệu tự định nghĩa. Để thay đổi kiểu dữ liệu thì ta phải XÓA kiểu dữ liệu cũ và tạo lại kiểu dữ liệu với thông số mới.



102

TẠO BẢNG DỮ LIỆU

CREATE TABLE tên_bảng

(

tên_cột_1 kiểu_dữ_liệu [thuộc_tính_cột_1] [các_ràng_buộc_cột_1] ,

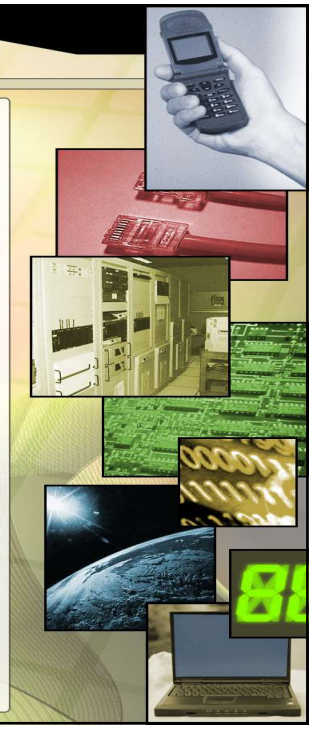
tên_cột_2 kiểu_dữ_liệu [thuộc_tính_cột_2] [các_ràng_buộc_cột_2] ,

… ,

tên_cột_n kiểu_dữ_liệu [thuộc_tính_cột_n] [các_ràng_buộc_cột_n]

[, các_ràng_buộc_trên_bảng]

) ;

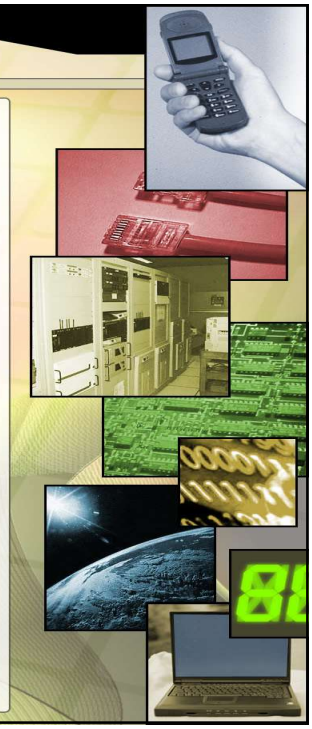


103

TẠO BẢNG DỮ LIỆU

Ví dụ:

```
CREATE TABLE NHANVIEN
(
  manv     VARCHAR(10) PRIMARY KEY,
  honv     NVARCHAR(30) NOT NULL,
  tennv    NVARCHAR(20) NOT NULL,
  gioitinh BIT DEFAULT(1),
  ngaysinh DATE NOT NULL,
  diachi   NVARCHAR(100) NOT NULL,
  dienthoai NVARCHAR(15) NULL
)
```



104

RÀNG BUỘC KHÓA CHÍNH

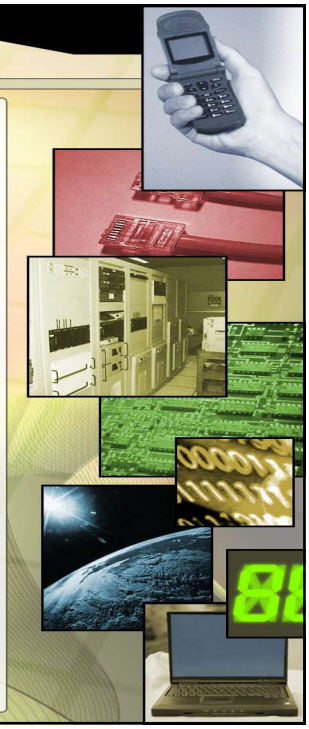
- **PRIMARY KEY** : định nghĩa **khoá chính** của bảng

  [CONSTRAINT tên_ràng_buộc]

  PRIMARY KEY (*danh_sách_cột*)

Ví dụ:

```
create table KetQuaHocTap
(
  masv varchar(10) not null,
  mamh varchar(10) not null,
  diem tinyint,
  constraint kequaHT_pk primary key(masv, mamh)
);
```



105

RÀNG BUỘC KHÓA NGOẠI

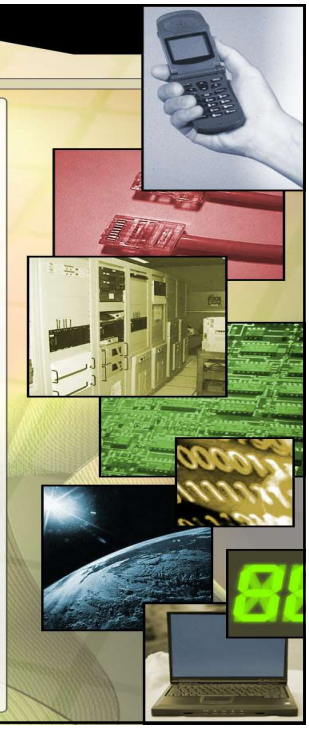
- **FOREIGN KEY** : được sử dụng để tạo ra mối liên kết dữ liệu giữa hai bảng.

[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]

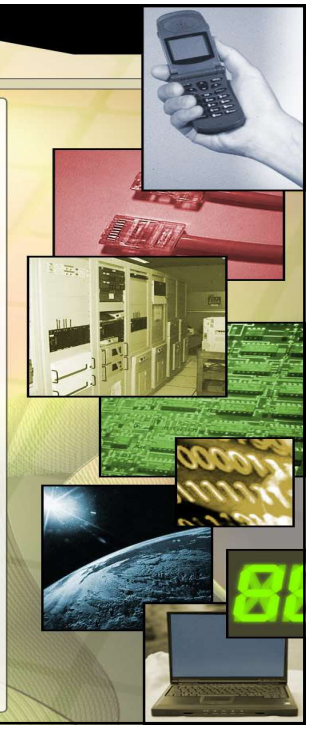
[ON UPDATE CASCADE | NO ACTION | SET NULL | SET DEFAULT]



106

RÀNG BUỘC KHÓA NGOẠI

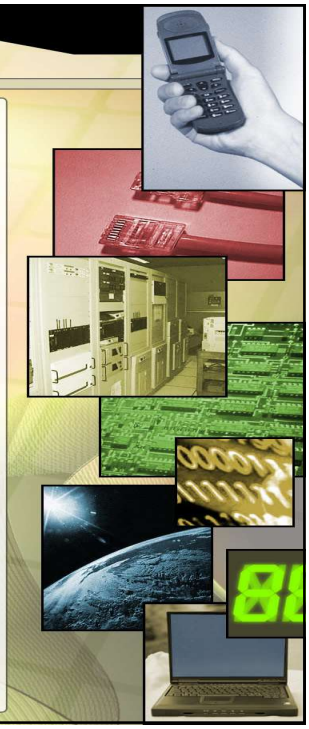
- Giá trị thêm vào cột FOREIGN KEY chỉ có thể nằm trong tập giá trị của PRIMARY KEY hoặc cột có ràng buộc UNIQUE của bảng mà FOREIGN KEY tham chiếu tới.
- Mặc định SQL Server không cho phép xóa dữ liệu trong bảng cha khi giá trị khóa chính đang được tham chiếu trong bảng con.



107

RÀNG BUỘC KHÓA NGOẠI

- **CASCADE** : Tự động xoá | cập nhật bản ghi trong bảng con nếu bản ghi được tham chiếu trong bảng cha bị xoá | cập nhật.
- **NO ACTION** : **Tùy chọn mặc định** khi tạo khóa ngoại trong SQL Server. Một dòng trong bảng cha đang được tham chiếu sẽ không thể bị xóa hay cập nhật.
- **SET NULL** : Cập nhật lại giá trị khoá ngoại của dòng trong bảng con thành NULL nếu dòng được tham chiếu trong bảng cha bị xóa.
- **SET DEFAULT** : Cập nhật lại giá trị khoá ngoại của dòng trong bảng con thành giá trị mặc định của cột nếu dòng được tham

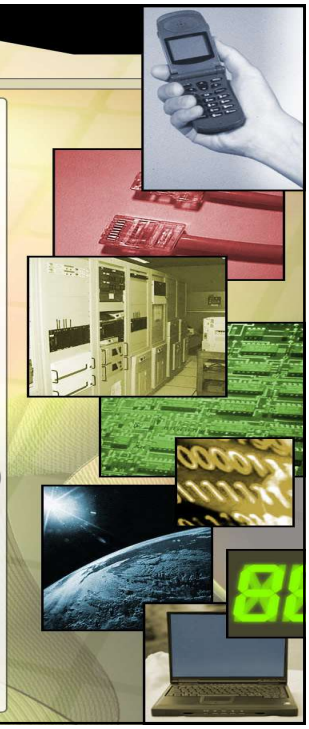


108

RÀNG BUỘC KHÓA NGOẠI

Ví dụ:

```
create table dondathang
(
   mahoadon varchar(10) primary key,
   makh varchar(10) not null,
   manv varchar(10) not null,
   ngaydathang datetime,
   constraint fk_dondathang_khachhang foreign key (makh)
   references khachhang (makh),
   constraint fk_dondathang_nhanvien foreign key (manv)
   references nhanvien (manv)
)
```



109

RÀNG BUỘC GIỚI HẠN DỮ LIỆU

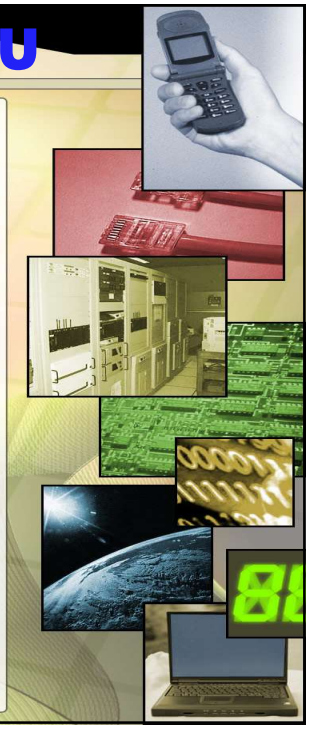
- **CHECK** : chỉ định điều kiện hợp lệ đối với dữ liệu

dữ liệu được **thêm** vào hoặc **cập nhật** trong bảng.

[CONSTRAINT tên_ràng_buộc] CHECK (*điều_kiện_logic*)

Ví dụ:

```
create table KetQuaHocTap
(
   masv varchar(10) not null,
   mamh varchar(10) not null,
   diem tinyint,
   constraint kequaHT_pk primary key(masv, mamh),
   constraint diem_ck check(diem >= 0 and diem <=10)
)
```



110

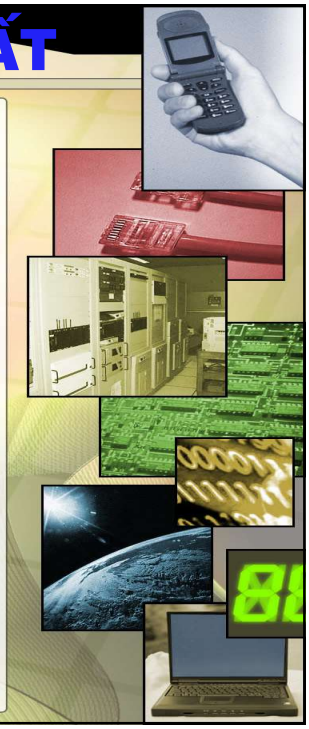
RÀNG BUỘC DỮ LIỆU DUY NHẤT

- **UNIQUE** : đảm bảo giá trị của một dòng tại một hay nhiều cột tham gia ràng buộc là **duy nhất**.

[CONSTRAINT tên_ràng_buộc] UNIQUE (*danh_sách_cột*)

Ví dụ:

```
create table hanghoa
(
   mahang    varchar(10) primary key,
   tenhang   nvarchar(100) not null,
   donvitinh nvarchar(10) not null,
   dongia    money check(dongia >= 0),
   constraint tenhang_unq unique (tenhang)
);
```



111

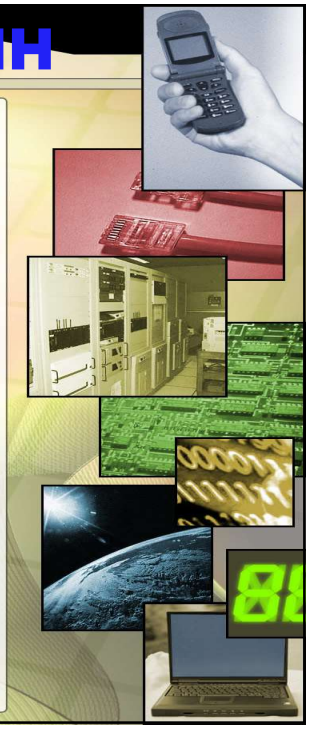
RÀNG BUỘC DỮ LIỆU MẶC ĐỊNH

- **DEFAULT** : cung cấp giá trị mặc định cho cột khi thêm dòng mới hoặc cập nhật dữ liệu mà giá trị cho cột này không được cung cấp.

Ví dụ:

```
CREATE TABLE MyTable
(
   column_a INT,
   column_b INT DEFAULT 50
);
```

```
CREATE TABLE MyTable
(
   column_a INT,
   column_b INT
    CONSTRAINT column_b_def DEFAULT 50
);
```



112

SỬA ĐỔI BẢNG

CÔNG VIỆC	CÚ PHÁP
Thêm cột vào bảng	ALTER TABLE tên_bảng ADD *định_nghĩa_cột_1* [, ...*n*]
Sửa cột trong bảng	ALTER TABLE tên_bảng ALTER COLUMN *tên_cột* kiểu_dữ_liệu [NULL | NOT NULL]
Xóa cột trong bảng	ALTER TABLE tên_bảng DROP COLUMN *tên_cột_1* [, ...*n*]
Thêm ràng buộc vào bảng	ALTER TABLE tên_bảng ADD CONSTRAINT *tên_ràng_buộc định_nghĩa_ràng_buộc*
Xóa ràng buộc	ALTER TABLE tên_bảng DROP CONSTRAINT *tên_ràng_buộc*

113

SỬA ĐỔI BẢNG

Ví dụ:

- Thêm cột EMAIL, DIACHI vào bảng NHANVIEN

```
alter table nhanvien
add email nvarchar(50) not null,
    diachi nvarchar(200);
```

- Thay đổi định nghĩa cột EMAIL

```
alter table nhanvien
alter column email nvarchar(40) null ;
```

- Xóa cột EMAIL

```
alter table nhanvien
drop column email ;
```

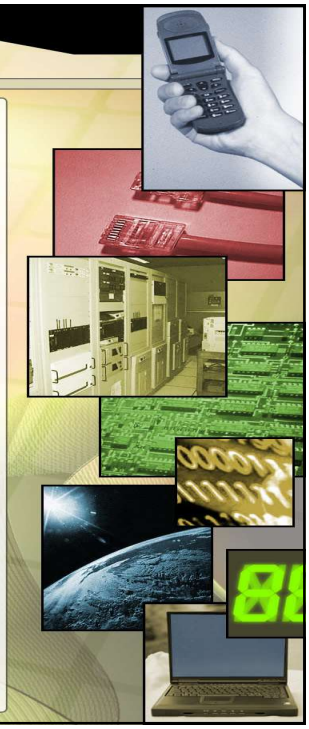
114

SỬA ĐỔI BẢNG

Ví dụ:

- Thêm cột EMAIL và xây dựng ràng buộc cho cột này

```
alter table nhanvien
add column email nvarchar(40) null,
constraint chk_email check
(email like  N'%_@_%.com');
```



115

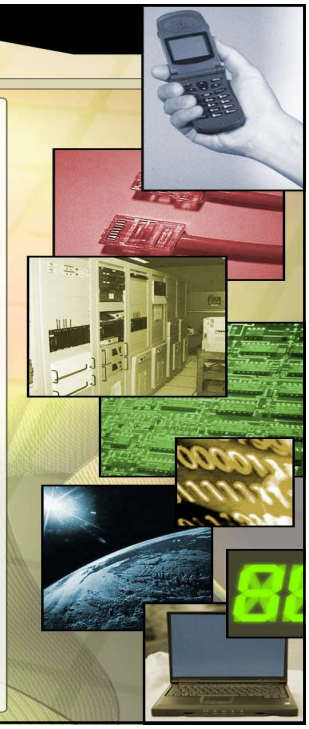
XÓA BẢNG

DROP TABLE Tên_bảng_1 [, ...n];

Ví dụ:

```
drop table hanghoa, MyTable;
```

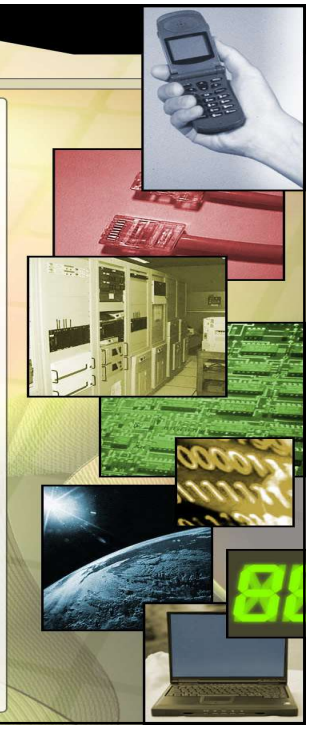
- Lệnh DROP TABLE không thể thực hiện nếu bảng cần xoá đang được tham chiếu bởi một ràng buộc FOREIGN KEY → cần xóa ràng buộc FOREIGN KEY hoặc bảng tham chiếu trước



116

VIEW

- Khung nhìn (view) là một thành phần của một bảng hoặc nhiều bảng được tạo thành bởi kết quả của một truy vấn SQL.
- View là bảng ảo cho phép
  - Cấu trúc dữ liệu theo cách của người dùng.
  - Giới hạn truy cập tới dữ liệu người dùng có thể thấy hoặc sửa đổi.
  - Tổng kết dữ liệu từ các bảng khác nhau để tạo ra bảng báo cáo.



117

TẠO VIEW

CREATE [OR ALTER] VIEW [tên_schema **.] tên_view**

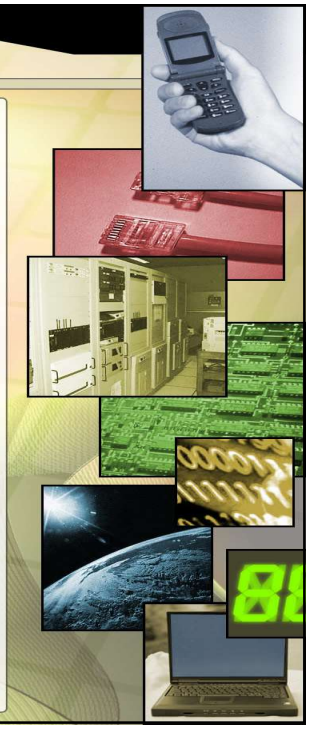
[***(tên_cột_đại_diện_1*** **[, …n])]**

AS

< lệnh SELECT >

[WITH CHECK OPTION] ;

Lưu ý: *Lệnh SELECT* ***không*** *chứa mệnh đề* ***ORDER BY***, ***OPTION***, *từ khóa* ***INTO***; ***không*** *chứa tham chiếu đến bảng tạm hoặc biến kiểu bảng*



118

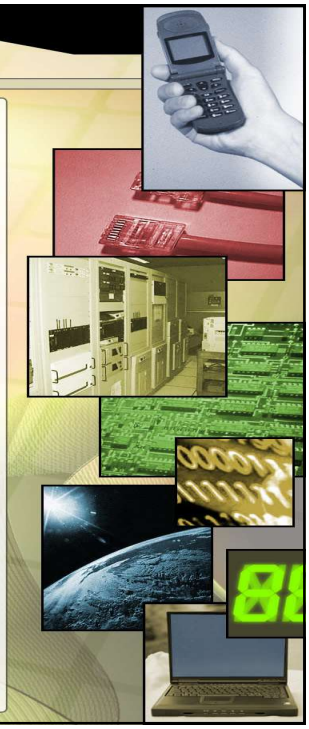
TẠO VIEW

Ví dụ:

```
create view sinhvien_view as
select ten, diachi
from  sinhvien;
```

Ví dụ:

```
create or alter view myview (khoa_name, tongsv) as
select k.khoa_name, COUNT(s.sv_id)
from khoa k , sinhvien s
where k.khoa_id = s.khoa_id
group by (k.khoa_name);
```



119

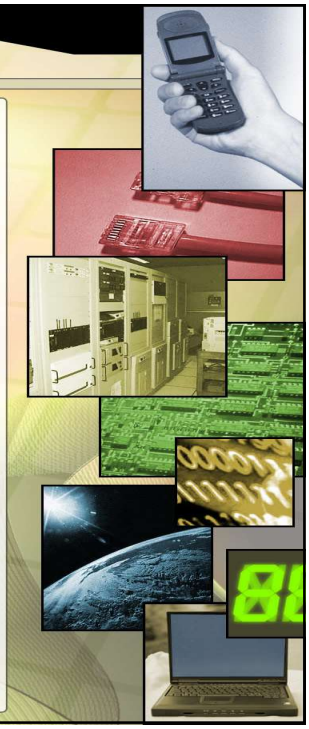
TẠO VIEW – CHECK OPTION

- **WITH CHECK OPTION** bảo đảm rằng tất cả UPDATE và INSERT thỏa mãn các điều kiện trong định nghĩa VIEW.

Ví dụ:

```
create view sinhvien_view as
select ten, diachi
from  sinhvien
where diachi is not null
with check option;
```

Với ví dụ trên, khi thực hiện insert hay update view thì trường *diachi* mới không được có giá trị NULL.

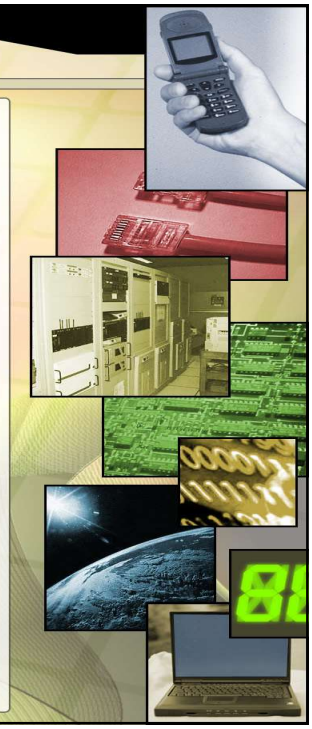


120

CẬP NHẬT VIEW

Điều kiện để cập nhật được view

- Mệnh đề SELECT không thể chứa từ khóa DISTINCT.
- Mệnh đề SELECT không thể chứa các hàm tổng.
- Mệnh đề SELECT không thể chứa các hàm và các toán tử tập hợp.
- Mệnh đề SELECT không thể chứa một mệnh đề ORDER BY.

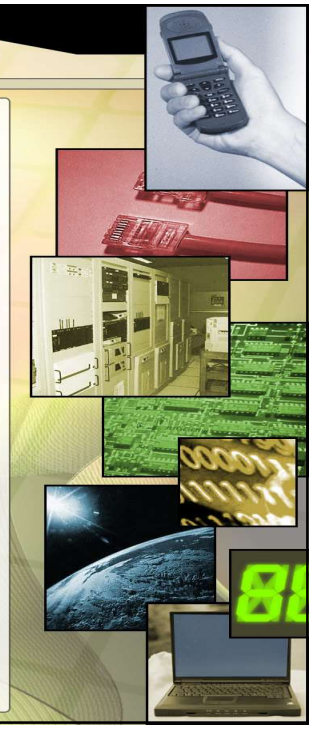


121

CẬP NHẬT VIEW

Điều kiện để cập nhật được view (tt)

- Mệnh đề FROM không thể chứa nhiều bảng.
- Mệnh đề WHERE không thể chứa các truy vấn phụ.
- Truy vấn không thể chứa GROUP BY hoặc HAVING.
- Các cột được ước lượng không thể bị cập nhật.
- Tất cả các cột NOT NULL từ bảng ban đầu phải được bao trong view để cho truy vấn INSERT vận hành.

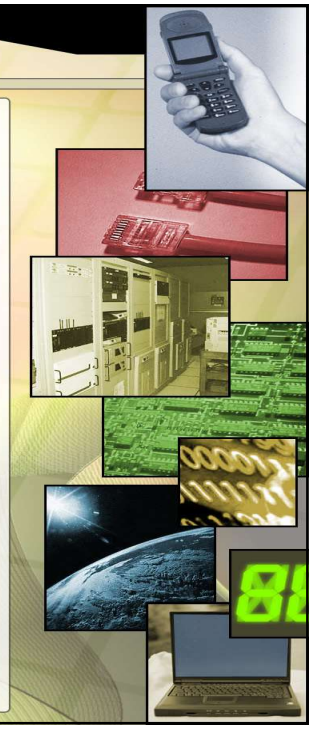


122

CẬP NHẬT VIEW

Điều kiện để cập nhật được view (tt)

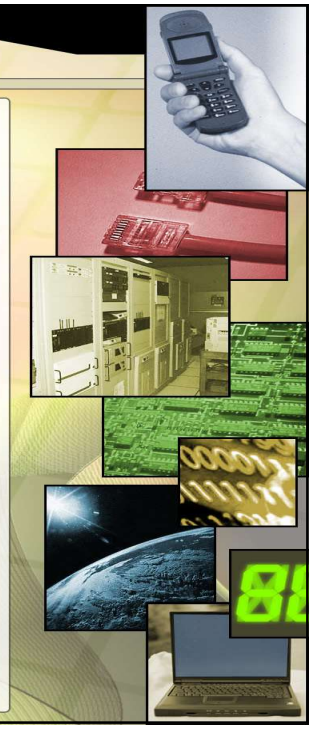
- Mệnh đề FROM không thể chứa nhiều bảng.
- Mệnh đề WHERE không thể chứa các truy vấn phụ.
- Truy vấn không thể chứa GROUP BY hoặc HAVING.
- Các cột được ước lượng không thể bị cập nhật.
- Tất cả các cột NOT NULL từ bảng ban đầu phải được bao trong view để cho truy vấn INSERT vận hành.



123

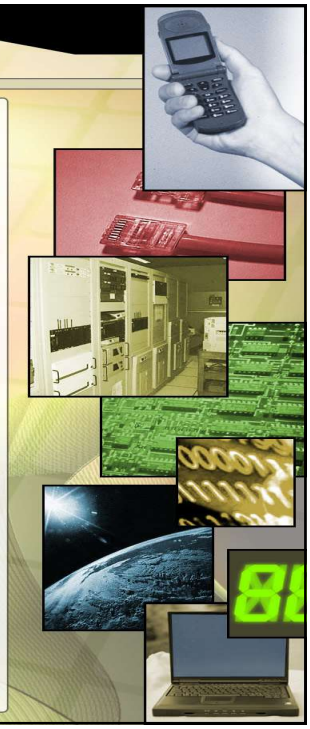
CẬP NHẬT VIEW

- INSERT
- UPDATE
- DELETE



124

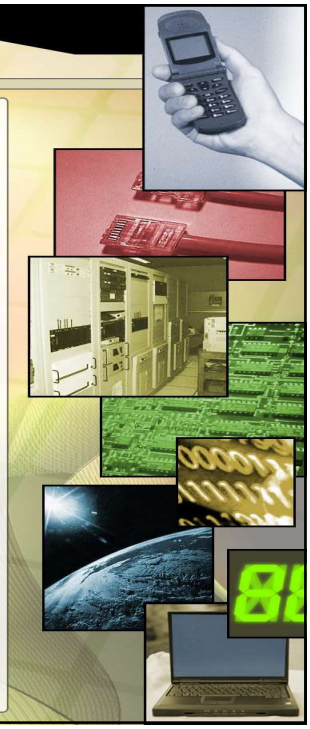
NGÔN NGỮ THAO TÁC DỮ LIỆU
DATA MANIPULATION LANGUAGE - DML



125

SELECT

```
SELECT [ ALL | DISTINCT ]
  [ Tên_đại_diện_bảng_1 . ] Mục_chọn_1 [ AS Tên_cột_hiển_thị_1 ]
  [ , ...n ]
FROM Bảng_1 [ Tên_đại_diện_bảng_1 ] [ , …n ]
[ WHERE Điều_kiện_1 [ AND | OR Điều_kiện_2… ] ]
[ GROUP BY Nhóm_cột_1 [ , ...n ] ]
[ HAVING Điều_kiện_nhóm ]
[ UNION [ ALL ] Lệnh_SELECT ]
[ ORDER BY Cột_sắp_xếp [ ASC | DESC ] [ , ... n ] ];
```



126

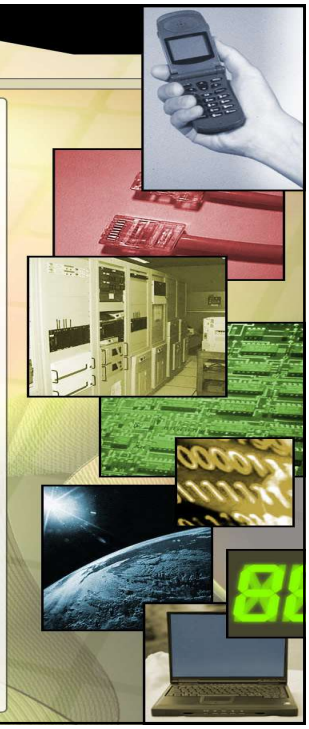
SELECT

Ví dụ:

```
select *
from sinhvien;
```

Ví dụ:

```
select khoa_name, COUNT(sv_id)
from khoa k , sinhvien s
where k.khoa_id = s.khoa_id
group by (khoa_name)
having COUNT(sv_id) <= 50;
```



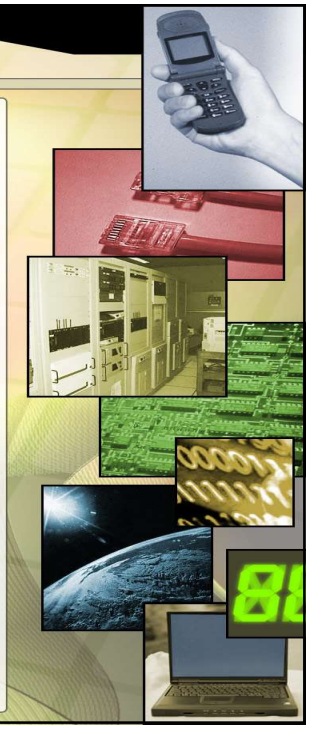
127

SELECT - ALIAS

ALIAS trong SQL Server là cách đổi tên một bảng hoặc một cột tạm thời bằng tên khác cho mục đích truy vấn (hiển thị hoặc rút gọn tên…). Tên ALIAS sẽ không ảnh hưởng (thay đổi) đến đối tượng trong CSDL và không thể dùng lại cho mục đích truy vấn hoặc tính toán

tên_cột | tên_bảng [AS] *tên_ALIAS*

***Lưu ý**: Nếu tên ALIAS có chứa khoảng trắng thì phải đặt trong dấu trích dẫn '' hoặc [ ]*

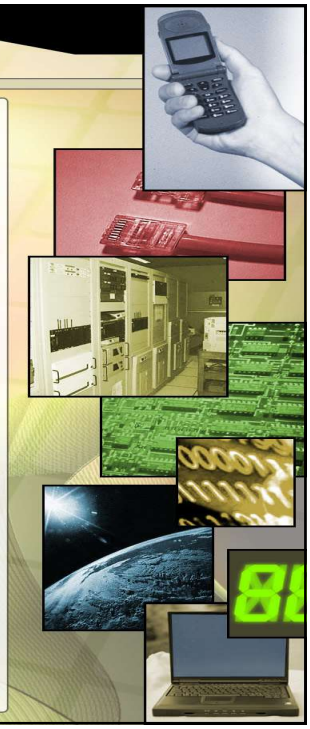


128

SELECT - ALIAS

Ví dụ:

```
select manv as [Mã nhân viên],
       honv + ' ' + tennv 'Họ tên nhân viên',
       diachi as 'Địa chỉ'
from nhanvien;
```

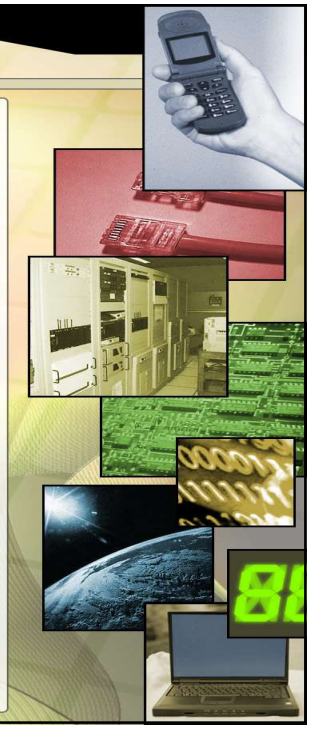


129

SELECT - ALIAS

Ví dụ:

```
select tmp.honv + ' ' + tmp.tennv as hoten,
       tmp.gioitinh, tmp.diachi
from ( select * from nhanvien
       where manv = 'nv001' or manv = 'nv002'
     ) as tmp;
```



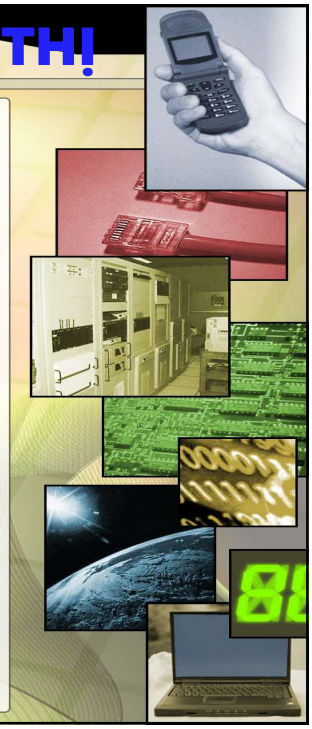
130

SELECT – THAY ĐỔI KẾT QUẢ HIỂN THỊ

Cấu trúc **CASE…WHEN...** được sử dụng nhằm thay đổi kết quả hiển thị của truy vấn tùy mục đích khác nhau nhưng KHÔNG làm thay đổi dữ liệu trong CSDL

- **Simple CASE**

```
CASE biểu_thức
   WHEN giá_trị_1 THEN kết_quả_1
   WHEN giá_trị_2 THEN kết_quả_2
   …
   [ ELSE kết_quả_còn_lại ]
END
```

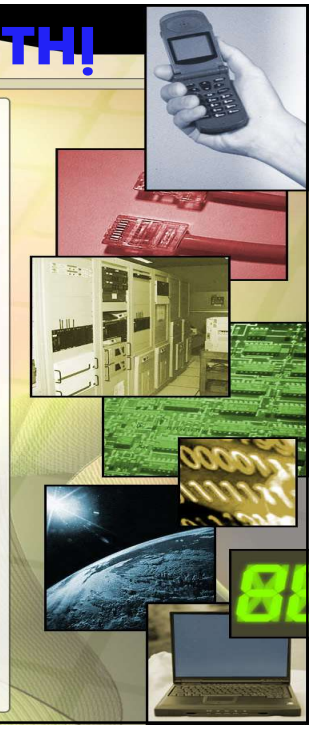


131

SELECT – THAY ĐỔI KẾT QUẢ HIỂN THỊ

- **Searched CASE**

```
CASE
   WHEN biểu_thức_logic_1 THEN kết_quả_1
   WHEN biểu_thức_logic_2 THEN kết_quả_2
   …
   [ ELSE kết_quả_còn_lại ]
END
```

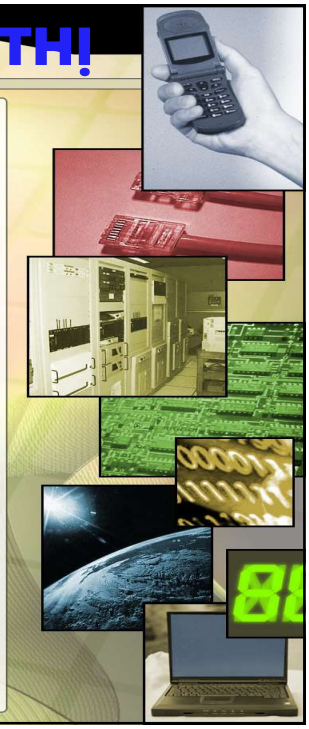


132

SELECT – THAY ĐỔI KẾT QUẢ HIỂN THỊ

Ví dụ:

```
select manv, honv, tennv,
       case gioitinh
          when 1 then N'nữ'
          else N'nam'
       end as gioitinh, diachi
from nhanvien;
```

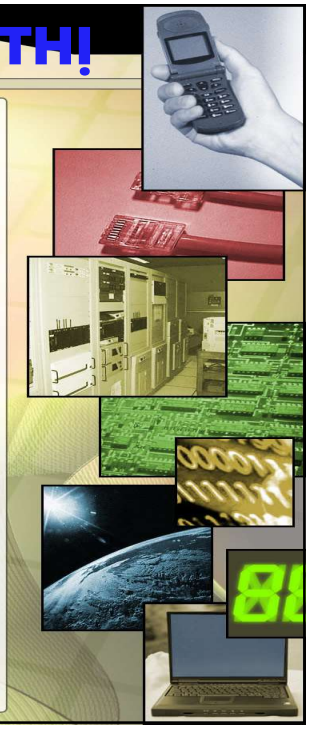


133

SELECT – THAY ĐỔI KẾT QUẢ HIỂN THỊ

Ví dụ:

```
select manv, honv, tennv,
       case
          when gioitinh=1 then N'nữ'
          else N'nam'
       end as gioitinh, diachi
from nhanvien;
```

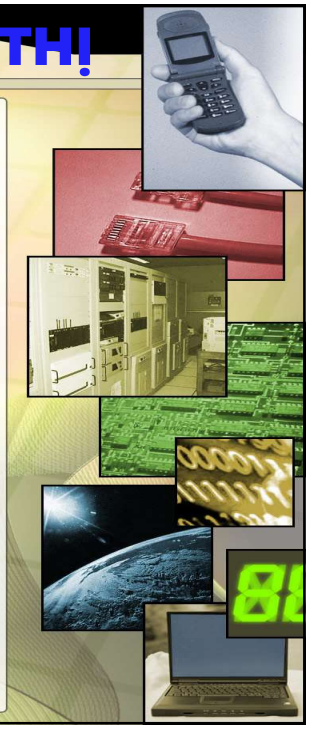


134

SELECT – THAY ĐỔI KẾT QUẢ HIỂN THỊ

Ví dụ:

```
select customername, city, country
from customers
order by
(case
    when city is null then country
    else city
end);
```



135

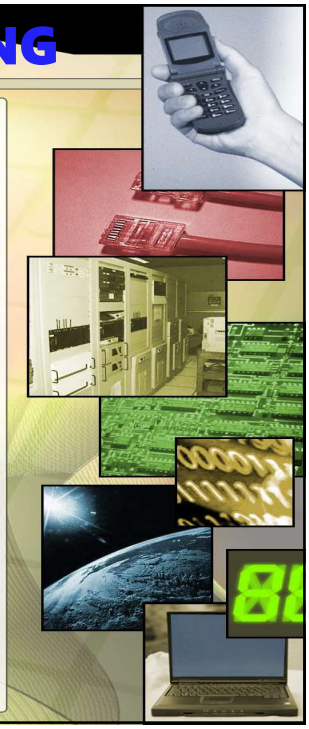
SELECT – HIỂN THỊ HỮU HẠN DÒNG

Ví dụ:

```
select TOP 2 manv, honv, tennv, diachi
from nhanvien;
```

Ví dụ:

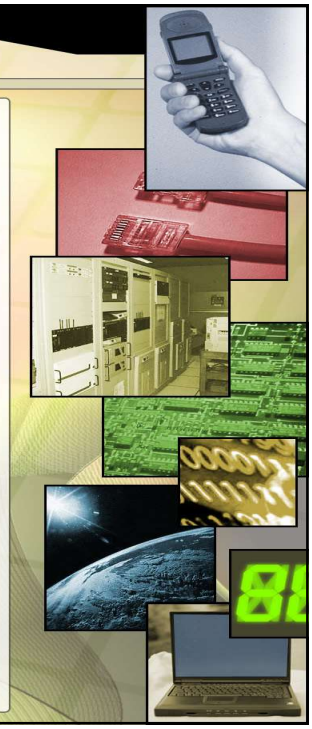
```
select TOP 10 PERCENT manv, honv, tennv, diachi
from nhanvien;
```



136

SELECT – TOÁN TỬ ĐIỀU KIỆN

- Toán tử kết nối điều kiện: **AND**, **OR**
- Toán tử so sánh giá trị: = , > , < , >= , <= , != , <> , !< , !>
- Toán tử so sánh giá trị tập hợp: **ALL | SOME | ANY**
- Toán tử kiểm tra giới hạn của dữ liệu: **BETWEEN | NOT BETWEEN**
- Toán tử làm việc với tập hợp: **IN | NOT IN**
- Toán tử kiểm tra khuôn dạng dữ liệu: **LIKE | NOT LIKE**
- Toán tử làm việc với giá trị NULL: **IS NULL | IS NOT NULL**
- Toán tử kiểm tra tồn tại dữ liệu: **EXISTS**

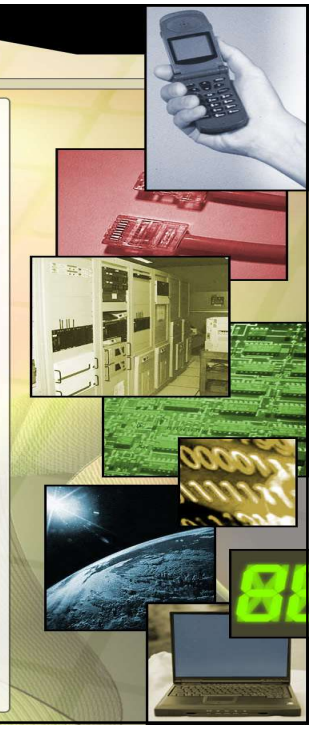


137

SELECT – TOÁN TỬ ĐIỀU KIỆN

Ví dụ: Liệt kê các nhân viên nữ có tuổi lớn hơn 40. Ngày sinh hiển thị theo định dạng dd/MM/yyyy

```
select manv, honv + ' ' + tennv as hoten,
convert(varchar(10), ngaysinh, 103) as ngaysinh
from nhanvien
where gioitinh = 0
and datediff(year, ngaysinh, getdate()) >= 40;
```

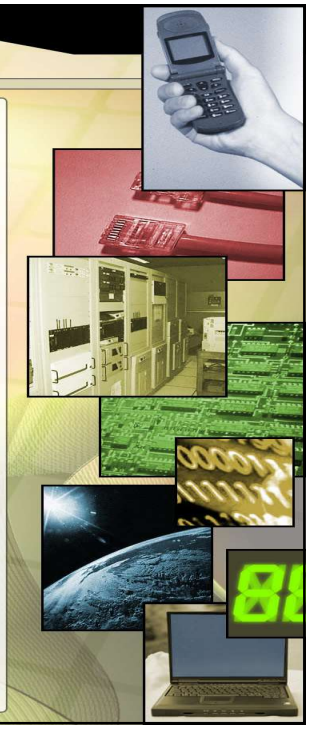


138

SELECT – TOÁN TỬ ĐIỀU KIỆN

Ví dụ: Liệt kê các nhân viên có tuổi thuộc tập hợp {50,55,60}

```
select *
from nhanvien
where datediff(year, ngaysinh, getdate())
      in (50,55,60);
```

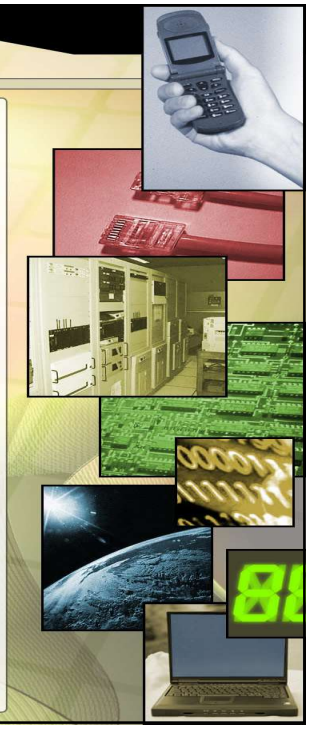


139

SELECT – TOÁN TỬ ĐIỀU KIỆN

Ví dụ: Liệt kê các nhân viên có lập hóa đơn cho khách hàng

```
select *
from nhanvien
where manv in (select manv from dondathang);
```

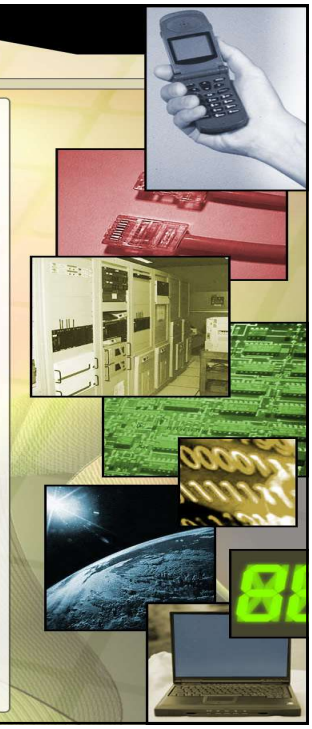


140

SELECT – TOÁN TỬ ĐIỀU KIỆN

Ví dụ: Liệt kê các phòng ban có nhân viên tên Châu

```
select * from phongban pb
where exists (
  select nv.manv
  from nhanvien nv
  where (nv.mapb = pb.mapb)
    and (nv.tennv like N'%Châu%')
);
```

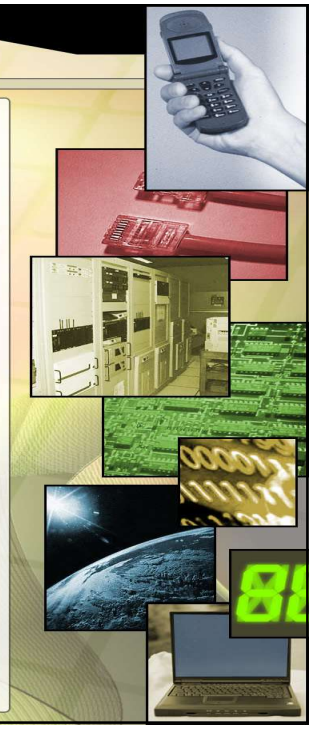


141

SELECT – TOÁN TỬ ĐIỀU KIỆN

Ví dụ: Liệt kê nhân viên có tuổi lớn nhất

```
select * from nhanvien
where datediff(year, ngaysinh,getdate())
>=all (select datediff(year, ngaysinh, getdate())
       from nhanvien);
```

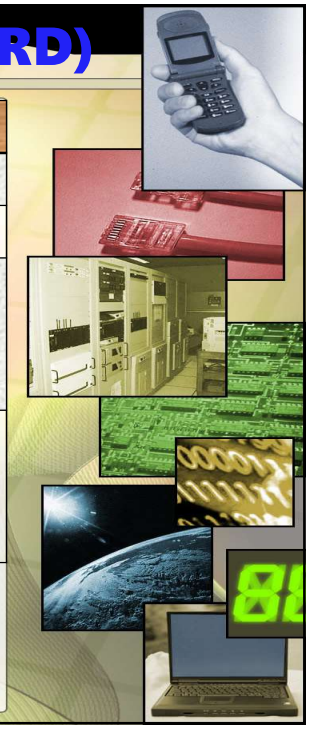


142

SELECT – KÝ TỰ ĐẠI DIỆN (WILDCARD)

Ký tự đại diện	Ý nghĩa
%	*Chuỗi ký tự bất kì bao gồm cả chuỗi rỗng*
_	*Một ký tự*
[ ]	*Ký tự nằm trong giới hạn được chỉ định.* *Ví dụ*: [a-f] hay [abcdef] hàm ý chỉ một trong các ký tự: a, b, c, d, e, f.
[^]	*Ký tự không nằm trong giới hạn được chỉ định.* *Ví dụ*: [^a-f] hay [^abcdef] hàm ý chỉ một ký tự khác tất cả các ký tự: a, b, c, d, e, f.

- **Ký tự thoát (escape character)**: là ký tự đặt trước ký tự đại diện để đánh dấu rằng ký tự đại diện sau nó được xem là một ký tự bình thường.



143

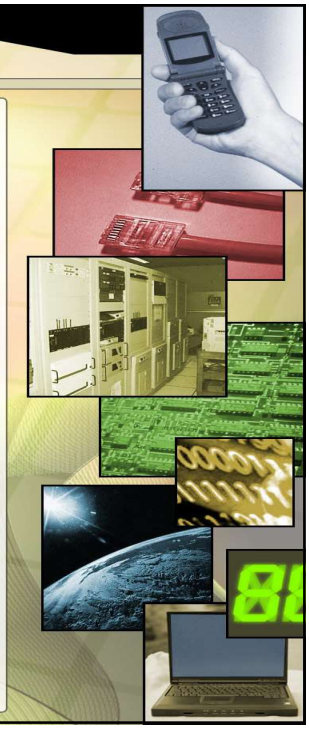
SELECT – WILDCARD

Ví dụ: Liệt kê những nhân viên họ Nguyễn

```
select *
from nhanvien
where honv like N'Nguyễn%';
```

Ví dụ:

```
select *
from tintuc
where noidung like N'%tăng trưởng 10-15!%%'
                ESCAPE '!';
```



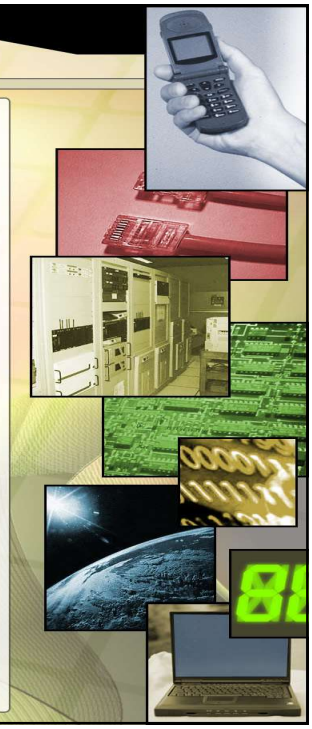
144

SELECT – PHÉP HỢP (UNION)

- **Kết quả truy vấn của hai hay nhiều câu truy vấn**

```
SELECT <Truy vấn Bảng 1>
UNION [ ALL ]
SELECT <Truy vấn Bảng 2>
UNION [ ALL ]
...
;
```

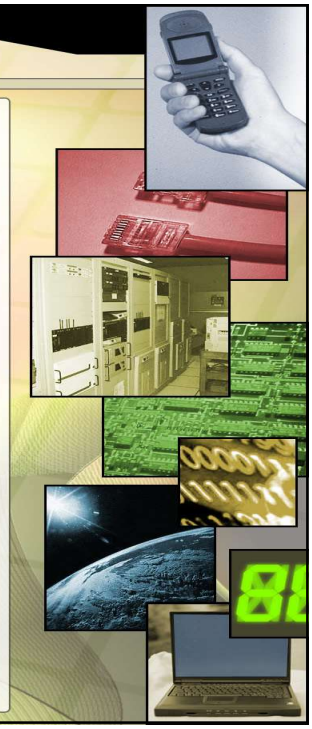
- UNION xóa các hàng trùng kết quả, UNION ALL sẽ lấy tất cả kết quả.



145

SELECT – PHÉP HỢP (UNION)

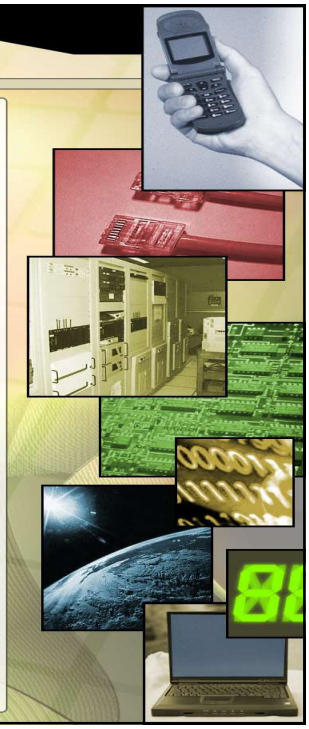
- Các cột trong các truy vấn thành phần phải tương ứng nhau.
- Các cột tương ứng trong tất cả các kết quả truy vấn phải có kiểu dữ liệu tương thích nhau.
- Các cột tương ứng trong bản thân từng truy vấn thành phần của một câu lệnh UNION phải xuất hiện theo thứ tự như nhau.



146

SELECT – PHÉP HỢP (UNION)

- Khi các kiểu dữ liệu khác nhau được kết hợp với nhau trong câu lệnh UNION, chúng sẽ được chuyển sang kiểu dữ liệu cao hơn (nếu có thể được).
- Tiêu đề cột trong kết quả của phép hợp sẽ là tiêu đề cột được chỉ định trong truy vấn đầu tiên.
- Mệnh đề ORDER BY chỉ được sử dụng ở sau câu lệnh SELECT cuối cùng.
- Mệnh đề GROUP BY và HAVING chỉ có thể được sử dụng trong bản thân từng truy vấn thành phần.



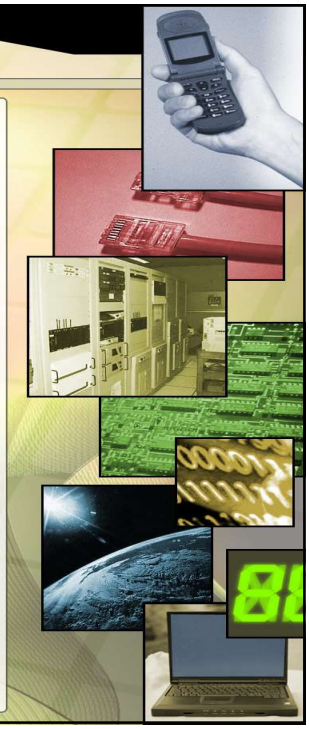
147

SELECT – PHÉP HỢP (UNION)

Ví dụ: Liệt kê thông tin khách hàng đặt hàng trong tháng 12/2009 và 1/2010.

Trước tiên chúng ta lấy thông tin khách hàng đặt mua hàng trong tháng 12/2009, sau đó lấy thông tin khách hàng đặt mua hàng trong tháng 1/2010 rồi hợp hai tập kết quả.

```
select distinct kh.makh, kh.hokh, kh.tenkh, kh.diachi
from khachhang kh join dondathang ddh
on kh.makh = ddh.makh where month(ddh.ngaydathang) = 12
and year(ddh.ngaydathang) = 2009
UNION
select distinct kh.makh, kh.hokh, kh.tenkh, kh.diachi
from khachhang kh join dondathang ddh
on kh.makh = ddh.makh where month(ddh.ngaydathang) = 1
and year(ddh.ngaydathang) = 2010;
```



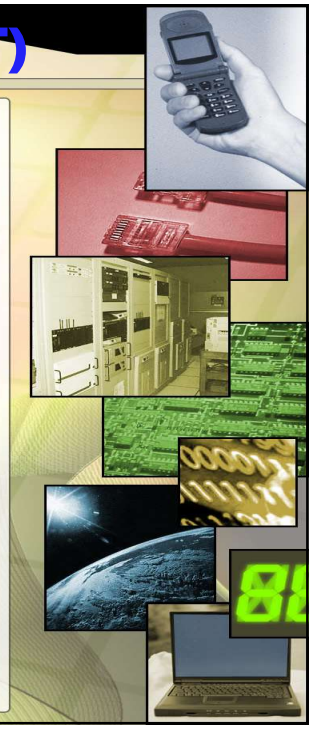
148

SELECT – PHÉP GIAO (INTERSECT)

- Tìm những dòng dữ liệu giống nhau trong nhiều kết quả truy vấn.

```
SELECT <Truy vấn Bảng 1>
INTERSECT
SELECT <Truy vấn Bảng 2>
INTERSECT
...
;
```

- Các điều kiện của INTERSECT tương tự như UNION

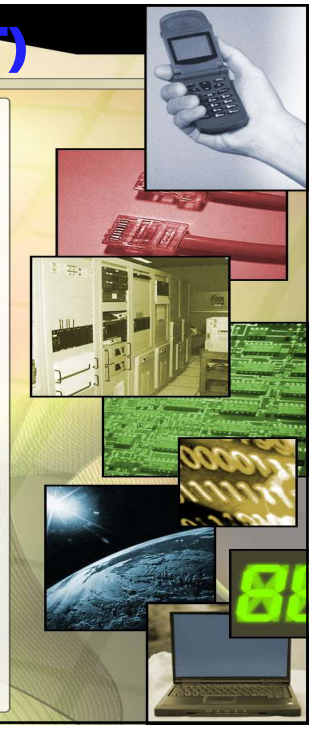


149

SELECT – PHÉP GIAO (INTERSECT)

Ví dụ: Liệt kê thông tin khách hàng đặt hàng trong cả 2 tháng 12/2009 và 1/2010.

```
select distinct kh.makh, hokh, tenkh, diachi
from khachhang kh join dondathang ddh
on kh.makh = ddh.makh where month(ngaydathang) = 12
and year(ngaydathang) = 2009
INTERSECT
select distinct kh.makh, hokh, tenkh, diachi
from khachhang kh join dondathang ddh
on kh.makh = ddh.makh where month(ngaydathang) = 1
and year(ngaydathang) = 2010;
```



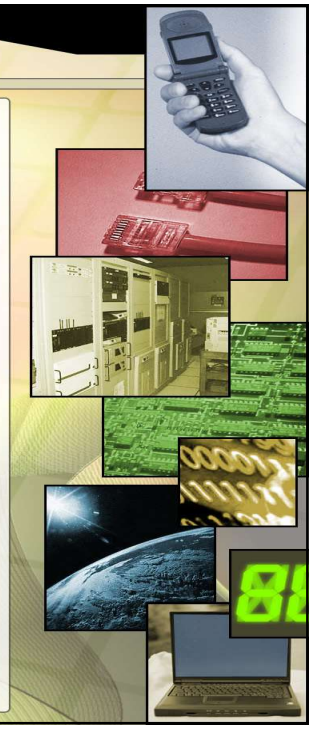
150

SELECT – PHÉP TRỪ (EXCEPT)

- **Xác định các bản ghi thuộc kết quả câu truy vấn này nhưng không thuộc kết quả câu truy vấn khác.**

```
SELECT <Truy vấn Bảng 1>
EXCEPT
SELECT <Truy vấn Bảng 2>
EXCEPT
...
;
```

- Các điều kiện của EXCEPT tương tự như UNION

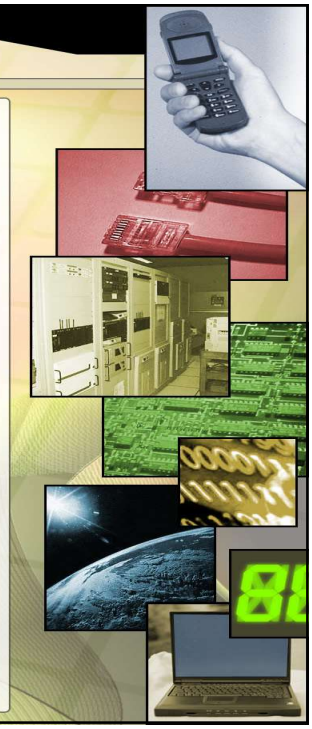


151

SELECT – PHÉP TRỪ (EXCEPT)

Ví dụ: Liệt kê thông tin khách hàng đặt hàng trong tháng 12/2009 không đặt hàng trong tháng 1/2010.

```
select distinct kh.makh, kh.hokh, kh.tenkh, kh.diachi
from khachhang kh join dondathang ddh
on kh.makh = ddh.makh where month(ddh.ngaydathang) = 12
and year(ddh.ngaydathang) = 2009
EXCEPT
select distinct kh.makh, kh.hokh, kh.tenkh, kh.diachi
from khachhang kh join dondathang ddh
on kh.makh = ddh.makh where month(ddh.ngaydathang) = 1
and year(ddh.ngaydathang) = 2010;
```

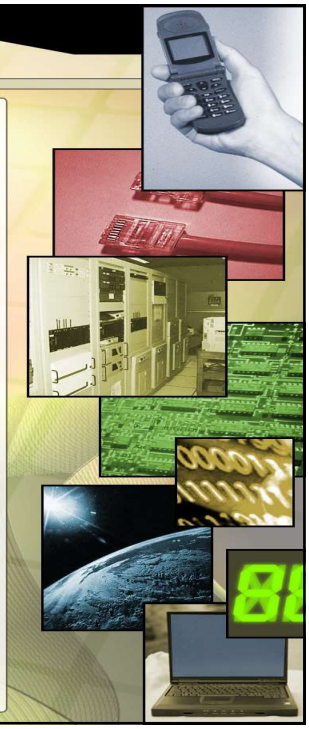


152

SELECT – PHÉP NỐI (JOIN)

- Xác định kết quả truy vấn từ hai hay nhiều bảng

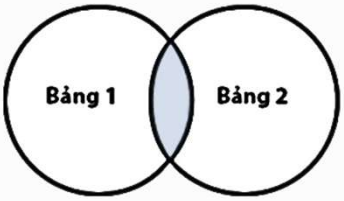
```
SELECT …
FROM  <Bảng_1> [ [ AS ] Alias_bảng_1 ]
<Kiểu_JOIN> <Bảng_2> [ [ AS ] Alias_bảng_2 ]
ON <Điều_kiện_JOIN>
…
;
```



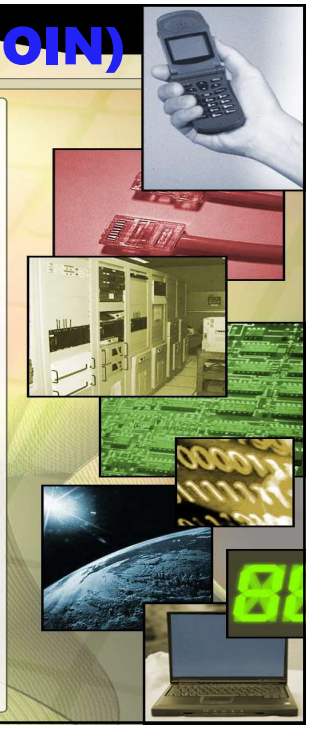
153

SELECT – PHÉP NỐI TRONG (INNER JOIN)

- Kết quả trả về là những bản ghi chung giữa các bảng




```
SELECT …
FROM  <Bảng_1> [ [ AS ] Alias_bảng_1 ]
[ INNER ] JOIN <Bảng_2> [ [ AS ] Alias_bảng_2 ]
ON <Điều_kiện_JOIN> …
WHERE …
;
```

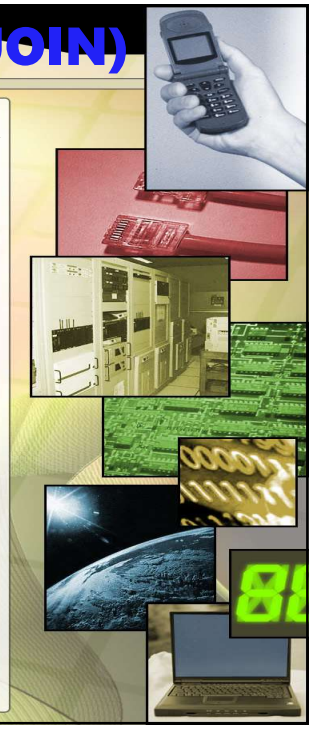


154

SELECT – PHÉP NỐI TRONG (INNER JOIN)

- Có thể sử dụng câu lệnh SQL sau thay cho INNER JOIN

```
SELECT …
FROM <Bảng_1> [ [ AS ] Alias_bảng_1 ] ,
        <Bảng_2> [ [ AS ] Alias_bảng_2 ]
WHERE <Điều_kiện_JOIN> AND …
…
;
```

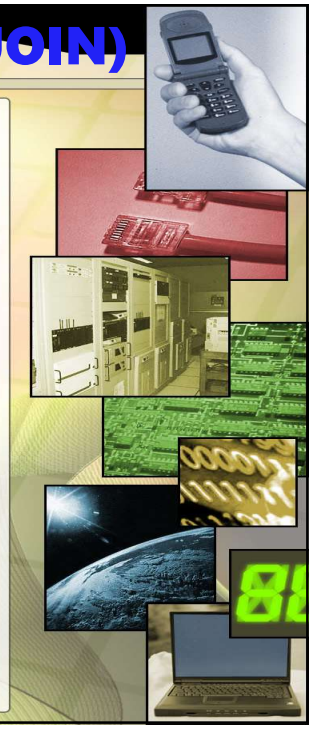


155

SELECT – PHÉP NỐI TRONG (INNER JOIN)

Ví dụ: Truy xuất thông tin các nhân viên đã lập hóa đơn trong tháng 12/2019

```
select nv.manv, honv, tennv, diachi,
       ngaydathang as ngaylaphoadon
from nhanvien nv join dondathang ddh
                  on nv.manv = ddh.manv
where month(ngaydathang) = 12
and year(ngaydathang) = 2019;
```



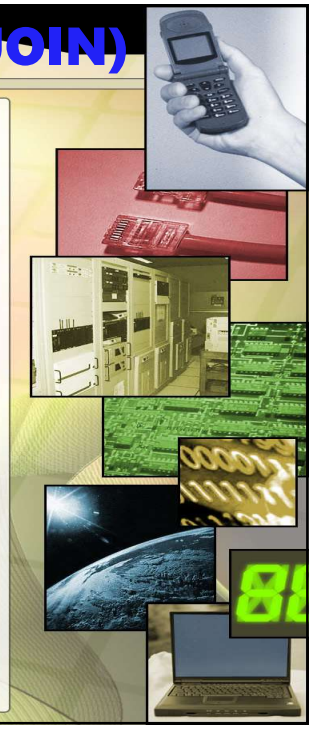
156

SELECT – PHÉP NỐI TRONG (INNER JOIN)

- **Phép tự nối** : điều kiện nối được chỉ định liên quan đến các cột của cùng một bảng. Các bảng phải được đặt tên alias

Ví dụ: Tìm các cặp nhân viên có cùng ngày sinh.

```
select nv1.manv, nv1.honv + ' ' + nv1.tennv
                    as hoten
from nhanvien nv1 join nhanvien nv2
on nv1.manv <> nv2.manv
and nv1.ngaysinh = nv2.ngaysinh;
```

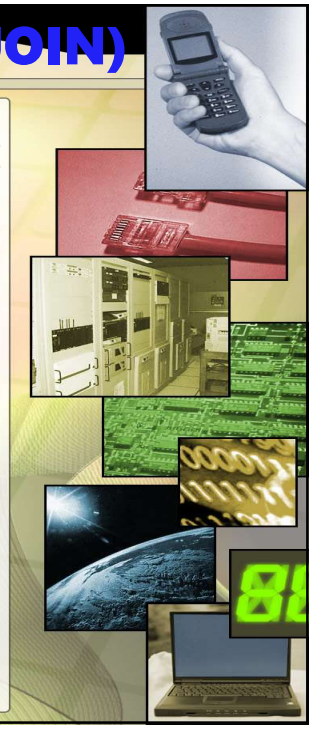


157

SELECT – PHÉP NỐI NGOÀI (OUTER JOIN)

- **Xác định kết quả truy vấn từ hai hay nhiều bảng kể cả những kết quả không thỏa mãn điều kiện nối**

SELECT …
FROM <*Bảng_1*> [[**AS**] Alias_bảng_1]
LEFT | **RIGHT** | **FULL** [**OUTER**] **JOIN**
<*Bảng_2*> [[**AS**] Alias_bảng_2]
ON <*Điều_kiện_JOIN*> …
WHERE …
;

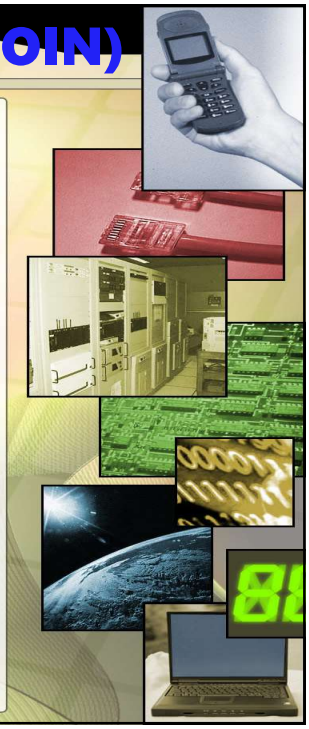


158

SELECT – PHÉP NỐI NGOÀI (OUTER JOIN)

Ví dụ: Xem tất cả thông tin nhân viên đã và chưa lập hóa đơn trong tháng 12/2019

```
select nv.manv, nv.honv, nv.tennv, nv.diachi,
       ddh.ngaydathang as ngaylaphoadon
from nhanvien nv left join dondathang ddh
                 on nv.manv = ddh.manv
where month(ddh.ngaydathang) = 12
and year(ddh.ngaydathang) = 2019;
```

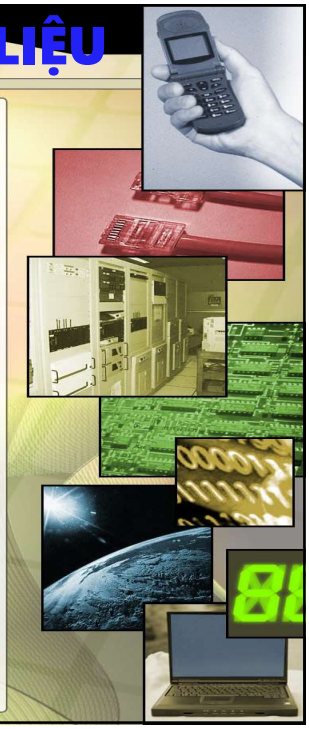


159

SELECT GROUP BY – THỐNG KÊ DỮ LIỆU

- Sử dụng mệnh đề GROUP BY để gom nhóm và thực hiện việc thống kê dữ liệu trên từng nhóm

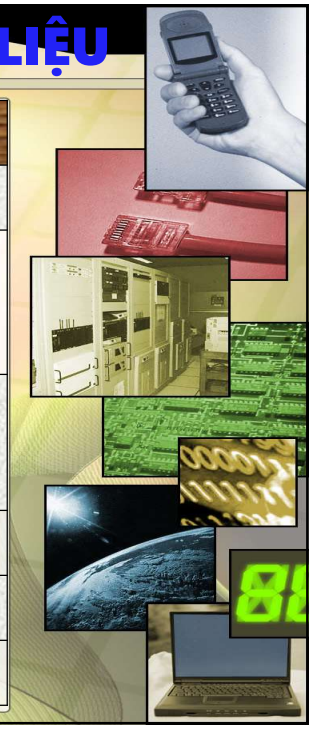
SELECT [<Danh_sách_chọn>],
<*Hàm_gộp_1*> [, <*Hàm_gộp_2*>…]
FROM …
WHERE …
[GROUP BY <Danh_sách_chọn>]
[HAVING <Điều_kiện_nhóm>]
;



160

SELECT GROUP BY – THỐNG KÊ DỮ LIỆU

HÀM GỘP	CHỨC NĂNG
SUM([ALL | DISTINCT] *biểu_thức*)	Tính tổng các giá trị.
AVG([ALL | DISTINCT] *biểu_thức*)	Tính trung bình của các giá trị
COUNT([ALL | DISTINCT] *biểu_thức*)	Đếm số các giá trị trong biểu thức
COUNT(*)	Đếm số các dòng được chọn
MAX(*biểu_thức*)	Tính giá trị lớn nhất
MIN(*biểu_thức*)	Tính giá trị nhỏ nhất

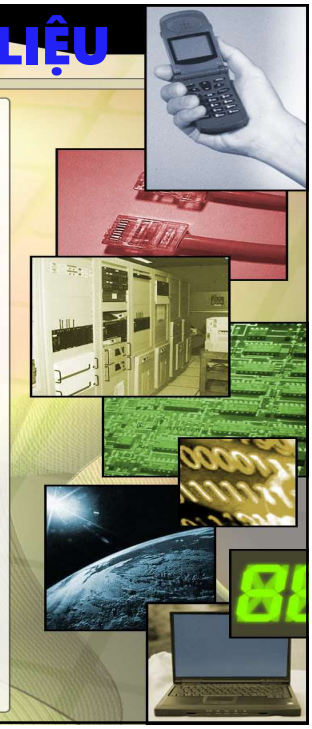


161

SELECT GROUP BY – THỐNG KÊ DỮ LIỆU

Ví dụ: Truy xuất tuổi lớn nhất, nhỏ nhất và tuổi trung bình của các nhân viên trong công ty.

```
select max(datediff(year, ngaysinh, getdate())) as tuoilonnhat,
       min(datediff(year, ngaysinh, getdate())) as tuoinhonhat,
       avg(datediff(year, ngaysinh, getdate())) as tuoitrungbinh
from nhanvien
```

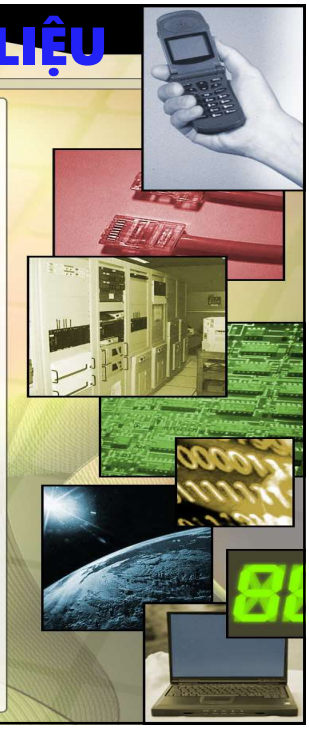


162

SELECT GROUP BY – THỐNG KÊ DỮ LIỆU

Ví dụ: Liệt kê thông tin nhân viên và số đơn đặt hàng của các nhân viên đó trong tháng 12/2019

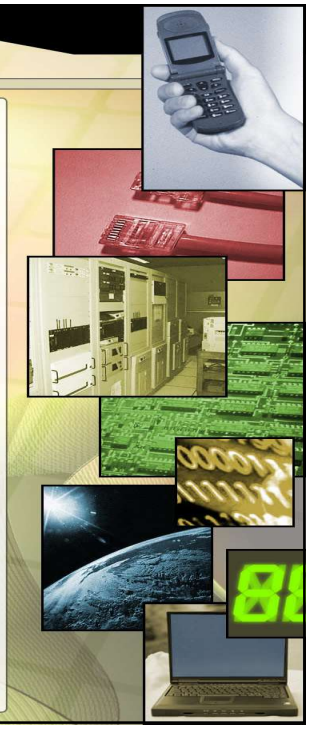
```
select nv.manv, nv.honv, nv.tennv, nv.diachi,
       count(ddh.manv) as N'Số đơn hàng'
from nhanvien nv left join dondathang ddh
                 on nv.manv = ddh.manv
where month(ddh.ngaydathang) = 12
   and year(ddh.ngaydathang) = 2019
group by nv.manv, nv.honv, nv.tennv, nv.diachi;
```



163

SELECT GROUP BY – HAVING

- Mệnh đề **WHERE** dùng để chỉ định điều kiện lọc áp dụng trên bất cứ dòng dữ liệu trong các bảng tham gia truy vấn.
- Mệnh đề **HAVING** chỉ có thể chỉ định điều kiện lọc đối với các dòng dữ liệu được liệt kê tại GROUP BY
- Mệnh đề HAVING cho phép sử dụng các hàm gộp trong biểu thức điều kiện còn WHERE thì không cho phép.

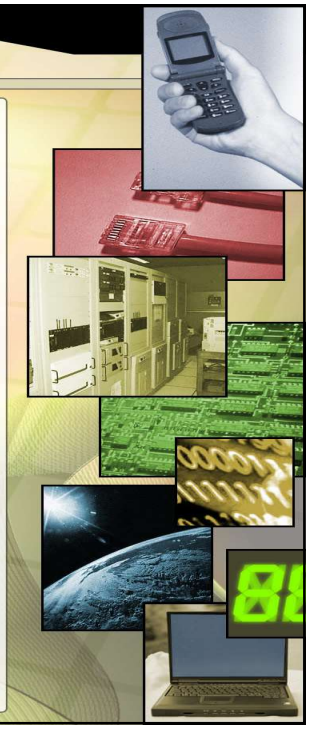


164

SELECT GROUP BY – HAVING

Ví dụ: Liệt kê thông tin nhân viên tên Châu có số đơn đặt hàng lớn hơn 10 trong năm 2019

```
select nv.manv, nv.honv, nv.tennv, nv.diachi
from nhanvien nv join dondathang ddh
                on nv.manv = ddh.manv
where year(ddh.ngaydathang) = 2019
group by nv.manv, nv.honv, nv.tennv, nv.diachi
having count(ddh.manv) > 10
and (nv.tennv like N'Châu');
```

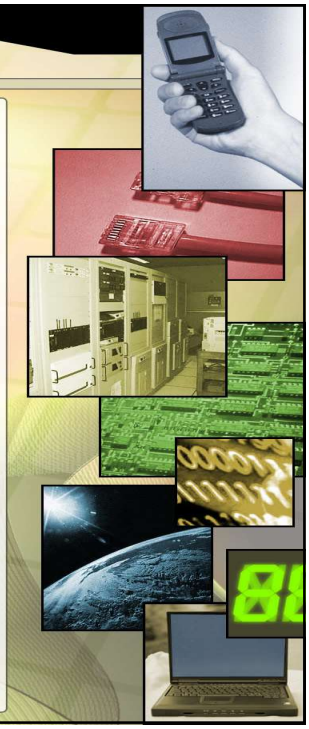


165

SELECT INTO

- Tạo một bảng mới có cấu trúc và dữ liệu được xác định từ kết quả của câu lệnh SELECT

SELECT … INTO [# | ##] tên_bảng
FROM …
;

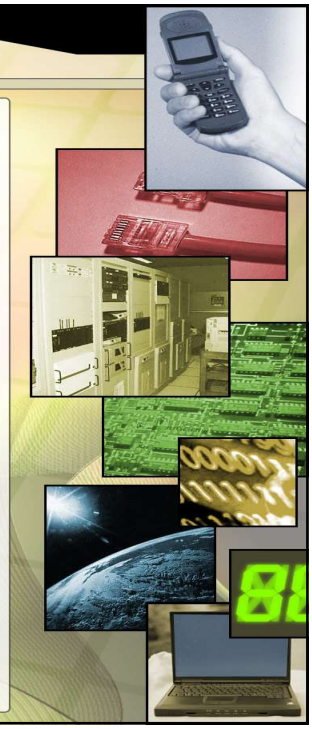


166

SELECT INTO

Ví dụ: Lưu thông tin các nhân viên đã lập hóa đơn trong tháng 12/2019 vào bảng có tên **baocao12_2019**

```
select nv.manv, honv, tennv, diachi,
       ngaydathang as ngaylaphoadon
into baocao12_2019
from nhanvien nv join dondathang ddh
                 on nv.manv = ddh.manv
where month(ngaydathang) = 12
and year(ngaydathang) = 2019;
```



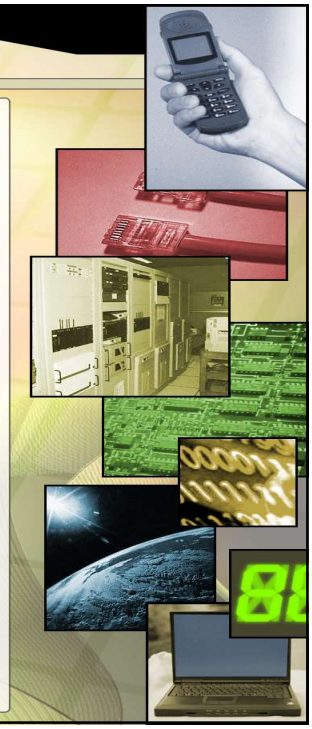
167

INSERT

INSERT INTO tên_bảng [(*danh_sách_cột*)]
VALUES (danh_sách_giá_trị_1) [, …n] ;

- Nếu không khai báo *danh_sách_cột* thì phải nhập danh_sách_giá_trị có đầy đủ các cột của bảng.
- Có thể sử dụng truy vấn SELECT để nhập danh_sách_giá_trị vào bảng.

INSERT INTO tên_bảng [(*danh_sách_cột*)]
truy_vấn_SELECT ;



168

INSERT

Ví dụ: Thêm giá trị vào bảng NhanVien (trường DiaChi có giá trị NULL)

```
insert into nhanvien (manv, honv, tennv,
                      gioitinh, ngaysinh)
values('nv001', N'Nguyễn Văn', N'An',
       1, '1990-09-19'),
('nv002', N'Trần Thị', N'Bình', 0, '1995-11-09'),
('nv003', N'Lê Hoàng', N'Châu', 1, '1993-02-24');
```

169

INSERT

Ví dụ: Tạo bảng tên **KhachHang_Backup** có cấu trúc giống hệt bảng ***KhachHang***.

```
insert into khachhang_backup
select * from khachhang;
```

hoặc

```
select * into khachhang_backup
from khachhang;
```

170

UPDATE

UPDATE tên_bảng

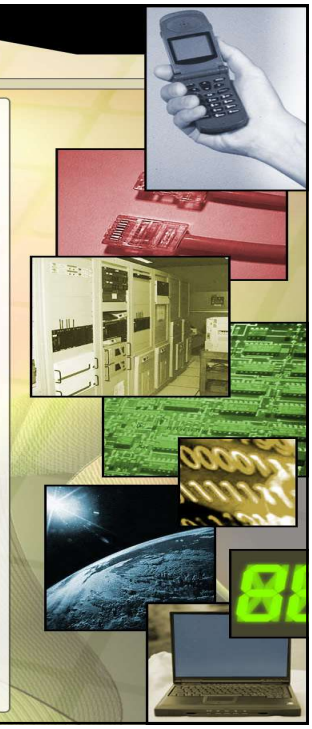
SET *tên_cột_1* = biểu_thức_1

[**,** ...n **]**

[FROM danh_sách_bảng_tác_động **]**

[WHERE điều_kiện]

- Nếu không chỉ định **điều_kiện** thì toàn bộ các bản ghi trong bảng sẽ được cập nhật.

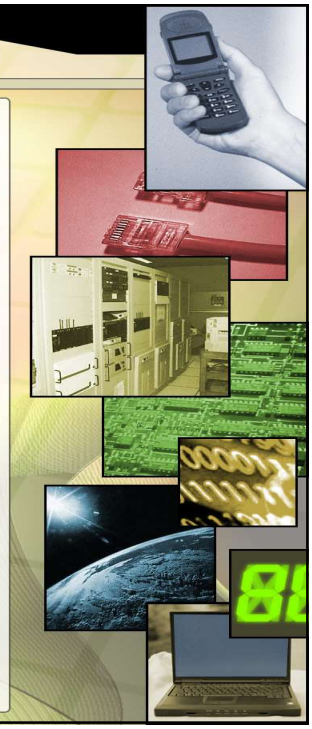


171

UPDATE

Ví dụ: Cập nhật tên hàng hóa

```
update hanghoa
set tenhh = N'Dell XPS 13'
where mahh = 'hh005';
```

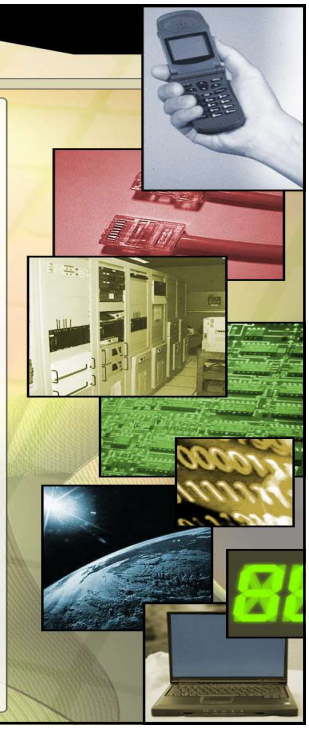


172

DELETE

DELETE [**TOP** *số_lượng* [**PERCENT**]]
FROM **tên_bảng**
[**FROM** danh_sách_bảng_tác_động]
[**WHERE** **điều_kiện**]
;

- Nếu không chỉ định **điều_kiện** thì toàn bộ các bản ghi trong bảng sẽ bị xóa.

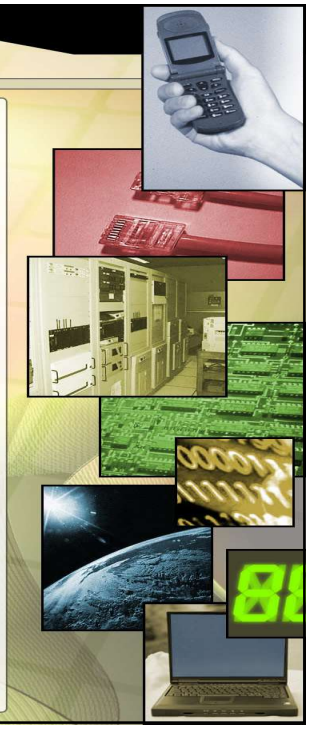


173

DELETE

Ví dụ: Xóa 2 bản ghi đầu tiên trong bảng **dondathang** của các nhân viên tên Châu

```
delete top 2 from dondathang
from dondathang as ddh join nhanvien as nv
on ddh.manv = nv.manv
where nv.tennv like N'Châu';
```



174

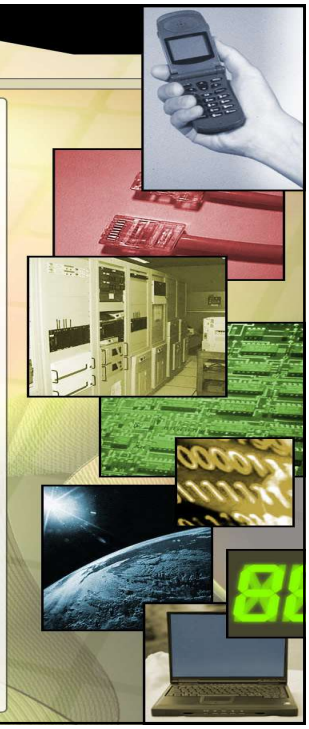
TRUNCATE

- TRUNCATE chỉ được sử dụng khi muốn xóa toàn bộ dữ liệu của bảng

  TRUNCATE TABLE tên_bảng;

- Dữ liệu bị xóa bởi DELETE có thể phục hồi được còn TRUNCATE thì không do dữ liệu bị xóa bởi TRUNCATE sẽ bị reset ở **Transaction Log** (do đó khi tạo 1 bản ghi mới, thứ tự bản ghi sẽ bắt đầu từ 1)

- TRUNCATE thực hiện nhanh hơn DELETE



175