Chương
Ngôn ngữ Transact - SQL

Nguyễn Đức Cương – cuongnguyenduc@gmail.com/
nguyenduccuong@iuh.edu.vn

Website: http://nguyenduccuong.com
http://elearningvn.org

Ngôn ngữ định nghĩa dữ liệu
(Data Definition Language - DDL)

- **DDL**: dùng định nghĩa và quản lý các thuộc tính của CSDL như: dòng(row), cột(column), khóa(key), vị trí lưu trữ tập tin.
- **DDL có 3 lệnh :**
  - CREATE object_name
  - ALTER object_name
  - DROP object_name

SQL

- Structure Query Language gồm
  - DDL (Data Definition Language)
  - DML (Data Manipulation Language)
  - DCL (Data Control Language)

Lệnh Create

- Tạo cơ sở dữ liệu và các đối tượng trong cơ sở dữ liệu.
  - Database
  - Table
  - View
  - Function, …

Lệnh Create

- Tạo cơ sở dữ liệu

```
CREATE DATABAS database_name
[ ON
[ PRIMARY ] [ <filespec> [ ,...n ]
[ , <filegroup> [ ,...n ] ]
[ LOG ON { <filespec> [ ,...n ] } ]
]
[ COLLATE collation_name ]
]
[;]
```

Lệnh Create

- Tạo CSDL Quanlysinhvien
  - Create database Quanlysinhvien

On

(filename = 'T:/CSDL/QLSV.mdf',

  name = 'QLSV',

  size = 2MB,

  maxsize = 10MB,

  filegrowth = 10%

)

Log On

(filename = 'T:/CSDL/QLSV_log.ldf',

  name = 'QLSV_log',

  size = 2MB,

  maxsize = 10MB,

  filegrowth = 10%

)

Tạo bảng

Sinhvien (MaSV, TenSv, Ngaysinh, Diachi)

```
Create table Sinhvien (
    MaSV char (10) Primary,
    TenSV varchar (50) not null,
    Ngaysinh smalldatetime,
    diachi nvarchar (50)
)
```

Tạo CSDL QLSV

- Tạo CSDL QLSV gồm các bảng sau:
  - KHOA (MaKhoa, TenKhoa, NgayThanhLap)
  - LOPHOC (MaLop, TenLop, NienKhoa, SoHocvien, MaKhoa)
  - MONHOC (MaMon, TenMon, SoTC)
  - HOCVIEN (MaHV, HoHV, TenHV, NgaySinh, QueQuan, MaLop)
  - KQUATHI (MaHV, MaMon, LanThi, NgayThi, DiemThi, GhiChu)

Lệnh Alter

- Hiệu chỉnh các đối tượng trong cơ sở dữ liệu
- **Cú pháp hiệu chỉnh cơ sở dữ liệu**:

```
ALTER DATABASE database_name
{
   <add_or_modify_files>
| <add_or_modify_filegroups>
| <set_database_options>
| MODIFY NAME = new_database_name
| COLLATE collation_name
}[;]
```

Lệnh Alter

- Hiệu chỉnh bảng - Thêm cột vào bảng

```
ALTER TABLE <table_name>
ADD <column_name1>  <data_type1>
[,<column_name2> <data_type2>, ...]
```

Lệnh Alter

- Hiệu chỉnh bảng - Thêm cột vào bảng

```
ALTER TABLE SINHVIEN
(ADD COLUMN DIENTHOAI VARCHAR (10)
)
```

Lệnh Alter

- Thêm ràng buộc vào bảng

```
ALTER TABLE <table_name>
ADD     CONSTRAINT     <constraint_name>
  <constraint_type1>    [,<column_name>
```

Lệnh Alter

- Thêm ràng buộc vào bảng

```
ALTER TABLE SINHVIEN
ADD CONSTRAINT FK_Lop
FOREIGN KEY (Malop)Reference Lop (malop)
```

Lệnh Alter

- Hiệu chỉnh kiểu dữ liệu

```
ALTER TABLE <table_name>
ALTER  COLUMN  <column_name1>  <data_type1>
  [,<column_name2> <data_type2>   ...]
```

Lệnh Drop

- Xóa cơ sở dữ liệu:

```
DROP DATABASE { database_name }
```

- Xóa bảng

```
DROP table { table_name }
```

- Xóa cột trong bảng

```
ALTER TABLE <table_name>
DROP COLUMN <column_name1>  [,<column_name2> ...]
```

Ngôn ngữ thao tác dữ liệu

- Truy vấn dữ liệu từ table (SELECT)
- Chèn dữ liệu vào table (INSERT)
- Cập nhật dữ liệu vào table(UPDATE)
- Xóa dữ liệu(DELETE)
- Fulltext Search

Cấu trúc lệnh truy vấn

```
SELECT Select_List
    FROM Table_List
    [WITH (BUFFERING = lExpr)]
    [WHERE Conditions]
    [GROUP BY Column_List]
    [UNION Clause]
    [HAVING Conditions]
    [ORDER BY Column_List]
    [INTO Clause | TO Clause ]
    [Additional_Display_Options]
```

Lệnh select

- **Câu truy vấn cơ bản**
- Ví dụ:
  - SELECT *
  - FROM Orders

  - SELECT OrderID, OrderDate, CustomerID
  - FROM Orders

Lệnh select

- Truy vấn loại bỏ các dòng bị trùng:
  - **Cú pháp: Select Distinct**
- Ví dụ:

```
SELECT DISTINCT Order_Date as "Date of Order"
FROM Orders
```

Lệnh select

- Truy vấn dùng các toán tử chuẩn trong biểu thức điều kiện:
- Ví dụ:
  - SELECT ProductID, UnitPrice
  - FROM Product_T
  - WHERE UnitPrice < 275;

Lệnh select

- Ví dụ:

```
SELECT ProductID, ProductName,UnitPrice
FROM Products
WHERE ProductName like 'N%'
AND UnitPrice > 300
```

Lệnh select

- Ví dụ: - Liệt kê các sản phẩm có giá bán >300$ và được bán trong tháng 7/2020
- SELECT ProductID, ProductName,UnitPrice
- FROM Products p, Orders o, [Order Details] od
- WHERE p.ProductId = od. ProductId and od.OrderID = o.OrderID and Year (Orderdate) = 2020 and Month (Orderdate) = 7 and ProductName like 'N%'
- AND UnitPrice > 300

Truy vấn trên nhiều bảng

- Ví dụ:

```
SELECT c.CustomerID, CompanyName, OrderID, OrderDate
FROM Customers C INNER JOIN Orders O
ON C.CustomerID = O.CustomerID
```

Truy vấn seft join

- Truy vấn trên một bảng liên kết với chính nó
- Ví dụ:

```
SELECT e.Firstname+' '+e.Lastname AS 'Employee',
m.Firstname+' '+m.Lastname AS 'Manager'
FROM Employees e JOIN Employees m
ON e.ManagerID = m.EmployeeID
```

Lệnh Union

- **Union** nối kết quả từ nhiều câu lệnh select
- Ví dụ:

```
SELECT  Firstname+' '+Lastname AS name, Homephone
FROM  Employees
UNION
SELECT  Companyname, Phone
FROM  Customers
```

Các hàm tổng hợp dữ liệu

- **Các hàm tổng hợp - Aggregate Functions**: tổng hợp thông tin từ nhiều bộ thành một bộ.
- Chức năng grouping được sử dụng để tạo nhóm trước khi thực hiện tổng hợp dữ liệu.
- Các hàm tổng hợp: **COUNT, SUM, MAX, MIN, AVG.**

Các hàm tổng hợp dữ liệu

- Ví dụ: *đếm số nhóm sách*

```
select count(*) as sonhom
from nhomsach
```

- *Ví dụ: tính tổng số cuốn sách đã bán*

```
select sum(soluong) as tongsoluong
from chitiethoadon
```

Mệnh đề GROUP BY

- **Mệnh đề GROUP BY:** chỉ định các thuộc tính kết nhóm xuất hiện trong mệnh đề **select**, kết quả của hàm thống kê được áp dụng cho các bộ trong cùng một nhóm.
- Ví dụ: *tính tổng số cuốn sách của mỗi nhóm sách*

```
select n.manhom, tennhom, sum(soluong) as tongsoluong
from nhomsach n join danhmucsach d on n.MaNhom=d.MaNhom
                join chitiethoadon c on d.MaSach=c.MaSach
group by n.manhom, tennhom
```

Mệnh đề HAVING

- **Mệnh đề Having**: xác định điều kiện lọc sau khi nhóm dữ liệu
- Ví dụ*: liệt kê các nhóm sách có tổng số sách >=30.*

```
select n.manhom, tennhom, sum(soluong) as tongsoluong
from nhomsach n join danhmucsach d on n.MaNhom=d.MaNhom
                join chitiethoadon c on d.MaSach=c.MaSach
group by n.manhom, tennhom
having sum(soluong)>=30
```

Lệnh select into

- Có thể tạo một bảng mới dựa vào kết quả của câu lệnh **select.**
- **Ví dụ:**

```
SELECT  C.CustomerID AS  NameId, OrderID, OrderDate
INTO  Customer_Order
FROM Customers C  INNER JOIN Orders  O
ON C.CustomerID = O.CustomerID
WHERE  month(OrderDate) = 7
```

Truy vấn con - Nested Queries

- **Nested query** là một query chứa một query khác, query được chứa bên trong gọi là *subquery* .
  - Subquery thường xuất hiện trong mệnh đề WHERE của query.
  - Ngoài ra Subquery cũng có thể xuất hiện trong mệnh đề FROM hoặc HAVING.

Các phép toán dùng trong nested query

- **IN**: so sánh một giá trị **v** với một tập giá trị **V,** kết quả là TRUE nếu **v** tồn tại trong **V.**
- Ví dụ: *liệt kê các sách thuộc nhóm sách 'Tin học'*

```
select masach, tensach
from danhmucsach
where MaNhom in (select MaNhom from nhomsach
                 where TenNhom =N'Tin học')
```

Các phép toán dùng trong nested query

- **NOT IN:** so sánh một giá trị **v** với một tập giá trị **V,** kết quả là TRUE nếu **v** không tồn tại trong **V**
- Ví dụ: *Tìm những quyển sách chưa bán*

```
select masach, tensach
from danhmucsach
where MaSach not in (select masach from chitiethoadon)
```

Các phép toán dùng trong nested query

- **ANY:** kết hợp với các phép toán **op** (>, >=, <, <=, and <>), kết quả là TRUE nếu và chỉ nếu các giá trị trong tập **v** thỏa mãn phép toán **op** với ít nhất là một giá trị trong **V**.
- **Ví dụ:** *liệt kê các sách có đơn giá lớn hơn đơn giá của ít nhất một sách trong nhóm sách 'N002'*

```
select masach, tensach, DonGia
from danhmucsach
where DonGia >any (select DonGia from danhmucsach
                   where MaNhom='N002')
```

Các phép toán dùng trong nested query

- **ALL**: kết hợp với các phép toán **op** (>, >=, <, <=, and <>), kết quả là TRUE nếu và chỉ nếu các giá trị trong tập **v** thỏa mãn phép toán **op** với tất cả giá trị trong **V**
- **Ví dụ:** *liệt kê các sách có đơn giá lớn hơn đơn giá của tất cả sách trong nhóm sách 'N002'*

```
select masach, tensach, DonGia
from danhmucsach
where DonGia >all (select DonGia from danhmucsach
                   where MaNhom='N002')
```

Các phép toán dùng trong nested query

- **EXISTS**: kiểm tra kết quả của subquery có rỗng hay không, exists trả về giá trị là TRUE nếu kết quả của subquery chứa ít nhất là một bộ giá trị.
- Ví dụ: *liệt kê các nhân viên lập hóa đơn*

```
select manv, TenNV
from nhanvien n
where exists (select MaNV from hoadon
                    where MaNV= n.MaNV)
```

Các phép toán dùng trong nested query

- **NOT EXISTS**: trả về giá trị là TRUE nếu kết quả của subquery không chứa bộ giá trị nào.

Ví dụ: *liệt kê các nhân viên không lập hóa đơn nào*

```
select manv, TenNV
from nhanvien n
where not exists (select MaNV from hoadon
                  where MaNV= n.MaNV)
```

Select With

- Trả ra một tập kết quả lưu trữ tạm thời trong common table expression (CTE).
- Tạo CTE

```
WITH name_CTE(fields_list)
AS
    (
        SELECT stmt
    )
```

Select With

- Sử dụng CTE

```
SELECT fields_list
FROM name_CTE
[WHERE]
[GROUP BY]
[ORDER BY]
[HAVING]
```

Select Merge

- Điểm mới trong SQL Server 2008 cho phép merge dữ liệu từ 2 table trở lên.
- Ví dụ:

```
CREATE TABLE T1(col1 Int Primary Key);
CREATE TABLE T2(col1 Numeric(12, 2) Primary Key);

SELECT T1.col1, T2.col1
FROM T1 Inner Merge Join T2
ON dbo.T1.col1 = dbo.T2.col1;
```

Merge query

- MERGE query là một loại query có thể thực hiện các hoạt động Insert, Update, Delete trên một bảng đích dựa trên các kết quả của một liên kết với một bảng nguồn
- Lợi thế quan trọng của câu lệnh MERGE là tất cả các dữ liệu được đọc và xử lý một lần.

Merge query

- **Cú pháp đơn giản**

```
Merge TargetTable
using SourceTable
on Joinconditions
[WHEN MATCHED THEN DML]
[WHEN NOT MATCHED BY TARGET THEN DML]
[WHEN NOT MATCHED BY SOURCE THEN DML]
```

Ví dụ - Merge query

```
CREATE TABLE dbo.BookInventory
(
  TitleID INT NOT NULL PRIMARY KEY,
  Title NVARCHAR(100) NOT NULL,
  Quantity INT NOT NULL
  CONSTRAINT Qutitydeft1 DEFAULT 0
);
```

Bảng Đích

```
CREATE TABLE dbo.BookOrder
(
  TitleID INT NOT NULL PRIMARY KEY,
  Title NVARCHAR(100) NOT NULL,
  Quantity INT NOT NULL
  CONSTRAINT Qutitydeflt DEFAULT 0
);
```

Bảng Nguồn

Ví dụ Merge - WHEN MATCHED

```
MERGE BookInventory bi
USING BookOrder bo
ON bi.TitleID = bo.TitleID
WHEN MATCHED THEN
   UPDATE
   SET bi.Quantity = bi.Quantity+bo.Quantity;
```

Ví dụ Merge - WHEN MATCHED

```
MERGE BookInventory bi
USING BookOrder bo
ON bi.TitleID = bo.TitleID
WHEN MATCHED AND
   bi.Quantity + bo.Quantity = 0 THEN
   DELETE
WHEN MATCHED THEN
   UPDATE
   SET bi.Quantity = bi.Quantity + bo.Quantity;
```

Ví dụ Merge - WHEN MATCHED

```
MERGE BookInventory bi
USING BookOrder bo
ON bi.TitleID = bo.TitleID
WHEN MATCHED AND
  bi.Quantity + bo.Quantity = 0 THEN
  DELETE
WHEN MATCHED THEN
  UPDATE
  SET bi.Quantity = bi.Quantity +
bo.Quantity;
```

Ví dụ Merge -WHEN NOT MATCHED BY TARGET

```
MERGE BookInventory bi
USING BookOrder bo
ON bi.TitleID = bo.TitleID
WHEN MATCHED AND
  bi.Quantity + bo.Quantity = 0 THEN DELETE
WHEN MATCHED THEN UPDATE
  SET bi.Quantity = bi.Quantity + bo.Quantity
WHEN NOT MATCHED BY TARGET THEN
  INSERT (TitleID, Title, Quantity)
  VALUES (bo.TitleID, bo.Title,bo.Quantity);
```

Ví dụ Merge -WHEN NOT MATCHED BY SOURCE

```
MERGE BookInventory bi
USING BookOrder bo
ON bi.TitleID = bo.TitleID
WHEN MATCHED AND
  bi.Quantity + bo.Quantity = 0 THEN DELETE
WHEN MATCHED THEN UPDATE
  SET bi.Quantity = bi.Quantity + bo.Quantity
WHEN NOT MATCHED BY TARGET THEN
  INSERT (TitleID, Title, Quantity)
  VALUES (bo.TitleID, bo.Title,bo.Quantity)
WHEN NOT MATCHED BY SOURCE
  AND bi.Quantity = 0 THEN DELETE;
```

Implementing the WHEN NOT MATCHED BY SOURCE

```
MERGE BookInventory bi
USING BookOrder bo
ON bi.TitleID = bo.TitleID
WHEN MATCHED AND
  bi.Quantity + bo.Quantity = 0 THEN DELETE
WHEN MATCHED THEN UPDATE
  SET bi.Quantity = bi.Quantity +
bo.Quantity
WHEN NOT MATCHED BY TARGET THEN
  INSERT (TitleID, Title, Quantity)
  VALUES (bo.TitleID, bo.Title,bo.Quantity)
WHEN NOT MATCHED BY SOURCE
  AND bi.Quantity = 0 THEN DELETE;
```

Pivot query

- Pivot query được dùng để tạo bảng thống kê dạng 2 chiều.
- Cú pháp:

pivoted_table ::=
table_source PIVOT (*aggregate_function* (
value_column)
PIVOT and UNPIVOT FOR *pivot_column*
IN (*column_list*) *table_alias*

Pivot query

- Ví dụ

```
SELECT empid, A, B, C, D
FROM (SELECT empid, custid, qty
FROM dbo.Orders) AS D
PIVOT(SUM(qty) FOR custid IN(A, B, C,
D)) AS P;
```

Pivot

- To use the PIVOT feature, you first decide which column contains the important values for the query.

```
SELECT empid, A, B, C, D
FROM (SELECT empid, custid, qty
FROM dbo.Orders) AS D
PIVOT(SUM(qty) FOR custid IN(A, B, C, D)) AS P;
```

Select - Compute By

- Trả về **tổng giá trị** của nhiều dòng tương ứng với field.
- Ví dụ:

```
SELECT SalesOrderID, UnitPrice, UnitPriceDiscount
FROM Sales.SalesOrderDetail
ORDER BY SalesOrderID
COMPUTE SUM(UnitPrice), SUM(UnitPriceDiscount);
```

```
SELECT SalesOrderID, UnitPrice, UnitPriceDiscount
FROM Sales.SalesOrderDetail
ORDER BY SalesOrderID
COMPUTE SUM(UnitPrice), SUM(UnitPriceDiscount) BY SalesOrderID;
```

Lệnh Insert

- **Cú pháp:**

```
Insert <Table Name>[field_List]
Values (value_1, value_2, …, value_n)
Values (…)…
```

Hoặc

```
Insert <Table Name>[field_List]
(Select statement)
```

Lệnh Delete

- **Cú pháp**:

```
Delete from <Table name>
[Where <condition>]
```

Hoặc

```
Delete Top [n] percent
From <Table name>
[Where <condition>]
```

Hoặc

```
Output DeletedDelete From
<Table name>
```

Lệnh Update

- **Cú pháp:**

```
Update <Table Name>
Set Field_Name = New_value
[Where <Condition>]
```

Câu lệnh TRUNCATE TABLE

- Dùng để xóa các dòng của table
- Nhanh hơn lệnh DELETE
- Không dùng với Trigger

Cú pháp: **TRUNCATE TABLE *table_name***

VD

TRUNCATE TABLE NewProducts

View

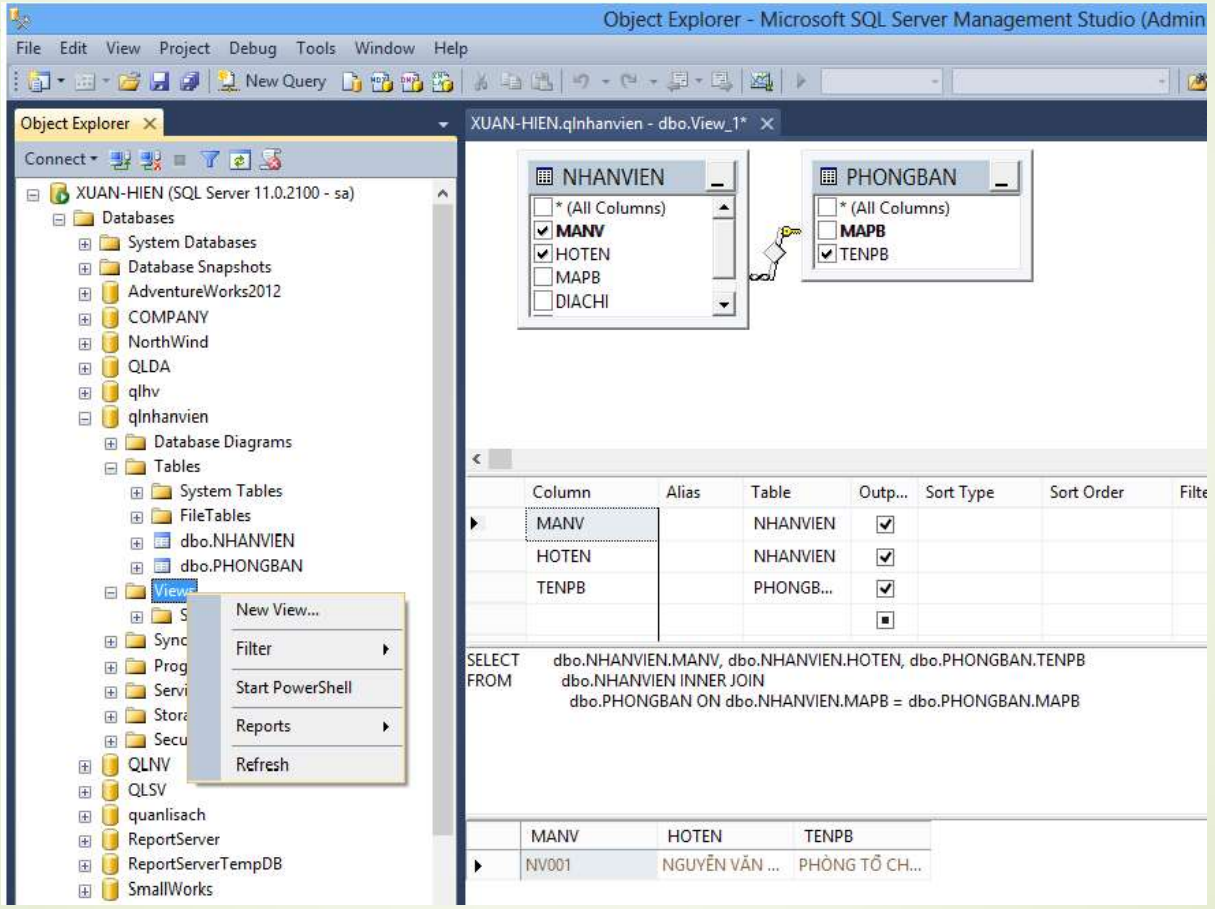
Khái niệm View

- **View** là đối tượng hoặc bảng ảo chứa kết quả từ một truy vấn.
- **Đặc trưng của views**:
  - Lọc dữ liệu từ các bảng.
  - Lọc dữ liệu cho mục đích bảo mật.
  - Tập trung dữ liệu phân tán từ nhiều máy chủ.
  - Tạo tập dữ liệu có khả năng tái sử dụng.

Tạo view

- **Sử dụng SSMS**
  - Trong Object Explorer, chọn cơ sở dữ liệu cần tạo view
  - Click phải trên folder Views → New View
  - Trong hộp thoại Add Table → chọn bảng cần lấy dữ liệu cho view.
  - Chọn các field → save

Tạo view


Object Explorer - Microsoft SQL Server Management Studio (Admin
File Edit View Project Debug Tools Window Help
New Query
Object Explorer
Connect
XUAN-HIEN (SQL Server 11.0.2100 - sa)
Databases
System Databases
Database Snapshots
AdventureWorks2012
COMPANY
NorthWind
QLDA
qlhv
qlnhanvien
Database Diagrams
Tables
System Tables
FileTables
dbo.NHANVIEN
dbo.PHONGBAN
Views
New View...
Filter
Start PowerShell
Reports
Refresh
QLNV
QLSV
quanlisach
ReportServer
ReportServerTempDB
SmallWorks
XUAN-HIEN.qlnhanvien - dbo.View_1*
NHANVIEN
* (All Columns)
MANV
HOTEN
MAPB
DIACHI
PHONGBAN
* (All Columns)
MAPB
TENPB
Column
Alias
Table
Outp...
Sort Type
Sort Order
MANV
NHANVIEN
HOTEN
NHANVIEN
TENPB
PHONGB...
SELECT dbo.NHANVIEN.MANV, dbo.NHANVIEN.HOTEN, dbo.PHONGBAN.TENPB
FROM dbo.NHANVIEN INNER JOIN
dbo.PHONGBAN ON dbo.NHANVIEN.MAPB = dbo.PHONGBAN.MAPB
MANV
HOTEN
TENPB
NV001
NGUYỄN VĂN ...
PHÒNG TỔ CH...

Tạo view

- Sử dụng T-SQL

```
CREATE VIEW  view_name   [(column[ ,...n ])]
AS select_statement  [ WITH CHECK OPTION ]
```

  - *Không dùng SELECT * trong view*
  - *Không sử dụng ORDER BY trong view, chỉ sử dụng với TOP, trong trường hợp này ORDER BY dùng chỉ định số record kết quả*

Tạo view

- Ex:

```
CREATE VIEW TongLuongPB
    (TenPB, TongSoNV, TongLuong)AS
    SELECT MaPB, COUNT (*), SUM (Luong)
    FROM PhongBan P, NhamVien N
    WHERE MaPB='P1'
    GROUP BY MaPB;
```

Tạo view

- **WITH CHECK OPTION**
  - Bắt buộc tất cả các thao tác hiệu chỉnh được thực hiện trên view phải thỏa điều kiện trong câu lệnh select

```
create view sachTH
as
    select masach, tensach, Manhom
    from danhmucsach
    where manhom='N001'
    with check option
```

Tạo view

- **WITH ENCRYPTION:**
  - Mã hóa câu lệnh tạo view.
- Ex:

```
create view empview with encryption
as
   select ssn, fname, minit, lname
   from employee
```

Tạo view

- **SCHEMABINDING:**
  - Không được hiệu chỉnh các bảng dùng trong câu lệnh tạo view.

```
create view VWSACH_sch with schemabinding
as
    select MaSach,TenSach,TacGia,MaNhom,DonGia
    from dbo.DanhMucSach
```

Thao tác trên view

- **DELETE VIEW:**

  DROP VIEW *view_name*

- **RENAME Views**:

  sp_rename *old_viewname*, *new_viewname*

- **CHECK VIEW:**

  sp_helptext *viewname*

- **MODIFY VIEW** *:*

  ALTER VIEW view_name (column_list)

  AS select_statement

Bài tập

- Câu 1: Thống kê số lượng khách hàng của từng thành phố
- Select City, Count (*) from Customers Group by City
- Câu 2: Liệt kê Danh sách nhân viên và doanh số bán hàng theo từng năm
- Select EmployeeID, Year (OrderDate), Sum (Unitprice*Quantity) From Orders o join [Order Detail] od on o.OrderId = od.OrderID Group by EmployeeID, Year (OrderDate)
- Câu 3: Liệt kê Danh sách nhân viên có doanh số bán hàng theo từng năm >10000$

CSDL - Northwind

- Customers (CustomerID, CompanyName, Address, City, ..) - thông tin khách hàng
- Categories (CategoryID, CategoryName,..)- Thông tin DM sản phẩm
- Products (ProductID, ProductName, UnitPrice, ..)- Thông tin sản phẩm
- Employees (EmployeeID, LastName, FirstName, City,..) - Thông tin nhân viên
- Orders (OrderID, Orderdate, CustID, EmpID) - Thông tin hóa đơn được lập
- Order Details (OrderID, ProID, Price, Quanlity)- Thông tin chi tiết của hóa đơn