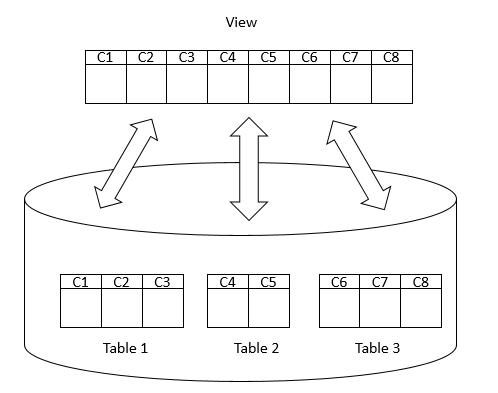
VIEW + FUNCTION SQL

1. View

VIEW là một bảng ảo (Virtual Table) không thực sự tồn tại trong SQL Server. Nó được tạo ra bởi truy vấn kết hợp 1 hoặc nhiều bảng. Thực chất, Một view là không gì khác ngoài môt lệnh SQL mà được lưu giữ trong Database với một tên liên kết. Một view thực sự là một thành phần của một bảng trong form của một truy vấn SQL đã được định nghĩa trước.

Tương tự như table, một view bao gồm nhiều dòng và nhiều cột.



Dùng View để làm gì?

* Truy vấn đơn giản, ngắn gọn hơn
* Bảo mật: để giới hạn quyền truy cập vào dữ liệu. Ví như một người dùng chỉ có thể xem và đôi khi sửa đổi chính xác những cột mà tồn tại trong view. Hoặc có user thì chỉ được xem dữ liệu mà không thể biết chính xác bảng cụ thể có các thông tin nào.
* View để tổng hợp dữ liệu từ các bảng khác nhau và có thể được sử dụng để tạo các báo cáo.

1. Lệnh tạo View:

|  |
| --- |
| **CREATE VIEW tên\_view AS**  **SELECT column1, column2.....**  **FROM tên\_bảng**  **[WHERE điều\_kiện];** |

VD: Tạo view để lấy thông tin sinh viên, gồm cả thông tin lớp:

|  |
| --- |
| CREATE VIEW v\_sinhvien\_lop  AS  SELECT masv, tensv, ngaysinh, Gioitinh, s.malop, l.tenlop FROM sinhvien s  LEFT JOIN lop l ON s.malop = l.malop; |

* Truy vấn dữ liệu từ view:

VD:

|  |
| --- |
| Select \* from v\_sinhvien\_lop where malop is not null; |

1. Sửa view:

|  |
| --- |
| **ALTER VIEW tên\_view AS**  **SELECT column1, column2.....**  **FROM tên\_bảng**  **[WHERE điều\_kiện];** |

VD: sửa view

|  |
| --- |
| ALTER VIEW v\_sinhvien\_lop  AS  SELECT masv, tensv, ngaysinh, Gioitinh, s.malop, l.tenlop FROM sinhvien s  LEFT JOIN lop l on s.malop = l.malop  WHERE ngaysinh is not null and gioitinh is not null  WITH CHECK OPTION; |

1. With check option:

* Thêm điều kiện with check option vào lệnh tạo/ sửa view để check điều kiện ở biểu thức WHERE, các cột tương ứng theo điều kiên khi INSERT/ UPDATE dữ liệu từ view vào bảng.

VD: WHERE ngaysinh is not null and gioitinh is not null

* Thì insert/ update 2 cột ngaysinh, gioitinh không được truyền null;

1. INSERT/ UDPATE/ DELETE qua view:  
   Thực hiện cú pháp DML dữ liệu vào View tương tự vào bảng.

|  |
| --- |
| **insert into tên\_view …**  **update tên\_view set …**  **delete from tên\_view …** |

Chú ý:

* Chỉ được thêm dữ liệu cho các cột tương ứng ở các bảng gọi trong view
* Chỉ thực hiện được DML cho view không có từ khóa DISTINCT, ORDER BY trong lệnh tạo view (SELECT).
* Mệnh đề SELECT không thể chứa các hàm tổng, các hàm tập hợp, toán tử tập hợp.
* Tất cả các cột NOT NULL từ bảng ban đầu phải được bao trong view để cho truy vấn INSERT vận hành.

VD:

|  |
| --- |
| insert into v\_sinhvien\_lop(masv, tensv, ngaysinh, Gioitinh) values ('1004',N'Huy','1995-02-15', 0);  update v\_sinhvien\_lop set malop='T2007E' where masv='1004'; |

1. Hủy View:

|  |
| --- |
| **DROP VIEW tên\_view** |

1. Function

* Có 2 loại:

+ Hàm có sẵn. VD: getdate(), year() …

+ Hàm do người dùng định nghĩa (user-defined).

* Hàm thuộc database và bao gồm một hoặc nhiều câu lệnh SQL và được tạo ra với mục đích sử dụng lại (reuse).
* Đối với hàm tự định nghĩa bao gồm 2 loại: Scalar Functions và Table-Valued Functions

1. Scalar Functions

Một scalar function là hàm **trả về một giá trị** với kiểu dữ liệu được khai báo trong RETURNS lúc tạo hàm.

|  |
| --- |
| **CREATE FUNCTION tên\_hàm ([@parameter\_name parameter\_data\_type [ = default ] [ ,...n ] ] )**  **RETURNS kiểu\_dữ\_liệu AS**  **BEGIN**  **//thân hàm**  **// khai bao biến**  declare @ten\_bien kiểu\_biến;  // lenh thuc thi  **RETURN value**  **END** |

VD:

|  |
| --- |
| create function fn\_dem\_sv\_lop (  @malop char(6)  )  returns int  as  begin  declare @cc int;  select @cc = count(\*) from sinhvien where malop = @malop;  return @cc;  end; |

Cách gọi:

|  |
| --- |
| select dbo.fn\_dem\_sv\_lop('T0001'); |

* Sử dụng alter, drop để thực hiện sửa hoặc xóa function.

1. Table-Valued Functions

Scalar function trả về một giá trị (single value) thì table-valued function trả về một table chứa những giá trị từ câu lệnh SELECT.

|  |
| --- |
| **CREATE FUNCTION tên\_hàm ([@parameter\_name parameter\_data\_type [ = default ] [ ,...n ] ] )**  **RETURNS TABLE AS**  **RETURN**  **câu\_lệnh\_select;** |

VD:

|  |
| --- |
| create function fn\_getList\_sinhvien\_bylop(  @malop char(6)  ) returns table as  return (  select \* from sinhvien where malop = @malop  ); |

Cách gọi:

|  |
| --- |
| select \* from dbo.fn\_getList\_sinhvien\_bylop('T0001'); |

1. Bài tập

Bài 1: view

Câu 1. Tạo view có tên vw\_CTPNHAP bao gồm các thông tin sau: số phiếu nhập hàng, mã vật tư, số lượng nhập, đơn giá nhập, thành tiền nhập.

Câu 2. Tạo view có tên vw\_CTPNHAP\_VT bao gồm các thông tin sau: số phiếu nhập hàng, mã vật tư, tên vật tư, số lượng nhập, đơn giá nhập, thành tiền nhập.

Câu 3. Tạo view có tên vw\_CTPNHAP\_VT\_PN bao gồm các thông tin sau: số phiếu nhập hàng, ngày nhập hàng, số đơn đặt hàng, mã vật tư, tên vật tư, số lượng nhập, đơn giá nhập, thành tiền nhập.

Câu 4. Tạo view có tên vw\_CTPNHAP\_VT\_PN\_DH bao gồm các thông tin sau: số phiếu nhập hàng, ngày nhập hàng, số đơn đặt hàng, mã nhà cung cấp, mã vật tư, tên vật tư, số lượng nhập, đơn giá nhập, thành tiền nhập.

Câu 5. Tạo view có tên vw\_CTPNHAP\_loc bao gồm các thông tin sau: số phiếu nhập hàng, mã vật tư, số lượng nhập, đơn giá nhập, thành tiền nhập. Và chỉ liệt kê các chi tiết nhập có số lượng nhập > 5.

Câu 6. Tạo view có tên vw\_CTPNHAP\_VT\_loc bao gồm các thông tin sau: số phiếu nhập hàng, mã vật tư, tên vật tư, số lượng nhập, đơn giá nhập, thành tiền nhập. Và chỉ liệt kê các chi tiết nhập vật tư có đơn vị tính là Bộ.

Câu 7. Tạo view có tên vw\_CTPXUAT bao gồm các thông tin sau: số phiếu xuất hàng, mã vật tư, số lượng xuất, đơn giá xuất, thành tiền xuất.

Câu 8. Tạo view có tên vw\_CTPXUAT\_VT bao gồm các thông tin sau: số phiếu xuất hàng, mã vật tư, tên vật tư, số lượng xuất, đơn giá xuất.

Câu 9. Tạo view có tên vw\_CTPXUAT\_VT\_PX bao gồm các thông tin sau: số phiếu xuất hàng, tên khách hàng, mã vật tư, tên vật tư, số lượng xuất, đơn giá xuất.

Bài 2: function

Câu 1. Viết function tên fn\_ThongKeSLDatTheoNhaCungCap, kiểu trả về là table: dùng để lấy tổng số lượng đặt theo từng nhà cung cấp. Ví dụ:

|  |  |
| --- | --- |
| MaNhaCC | Tổng số lượng đặt |
| C01 | 30 |
| C03 | 25 |

Câu 2. Viết function tên fn\_LayThongTinNhaCungCap, kiểu trả về là int và nhận tham số @MaNhaCC: dùng để đếm số hóa đơn của nhà cung cấp. Ví dụ:

|  |  |
| --- | --- |
| MaNhaCC | Tổng số hóa đơn |
| C01 | 1 |
| C02 | 2 |

Câu 3. Viết function tên fn\_LayThongTinPhieuNhap, kiểu trả về float và nhận tham số @SoPN: dùng để tính tổng số tiền nhập của @SoPN. Ví dụ:

|  |  |
| --- | --- |
| SoPN | Tổng tiền |
| N001 | 6000000 |
| N002 | 6000000 |

Câu 4. Viết function tên fn\_LayThongTinPhieuXuat, kiểu trả về int và nhận tham số @year: dùng để đếm số phiếu xuất được lập trong năm @year. Ví dụ:

|  |  |
| --- | --- |
| Năm | Tổng số phiếu xuất |
| 2007 | 3 |
| 2016 | 1 |

Câu 5. Viết function tên fn\_LayThongTinVatTuTheoPhieuXuat, kiểu trả về table: dùng để đếm số vật tư theo từng phiếu xuất. Ví dụ:

|  |  |
| --- | --- |
| SoPX | Số vật tư |
| X001 | 1 |
| X002 | 2 |

Bài tập làm trên lớp:

**Câu 1.** Tạo function cho biết số lượng sinh viên của một lớp với mã lớp là tham số.

**Câu 2.** Tạo function cho biết mã sinh viên, tên sinh viên, tên lớp, mã môn học, tên môn học, lần thi và điểm thi của từng sinh viên.

**Câu 3.** Tạo function cho biết mã khoa, tên khoa và số lượng giảng viên của từng khoa.

**Câu 4.** Tạo function hiển thị mã sinh viên, tên sinh viên, giới tính, ngày sinh, mã lớp, tên lớp, mã môn học, tên môn học, lần thi và điểm thi. Function này nhận tham số là mã sinh viên.