

LỜI NÓI ĐẦU

Hệ quản trị cơ sở dữ liệu là học phần cơ sở ngành bắt buộc trong chương trình đào tạo ngành Công nghệ thông tin giúp sinh viên có được một kiến thức toàn diện trong việc hiểu và sử dụng hệ quản trị cơ sở dữ liệu SQL Server.

Microsoft SQL Server là một hệ quản trị cơ sở dữ liệu mô hình quan hệ (Relational Database Management System – RDBMS) được phát triển bởi Microsoft dùng để quản lý và lưu trữ dữ liệu theo mô hình Client/Server. Đây là một trong những hệ quản trị cơ sở dữ liệu có đầy đủ các tính năng của một hệ quản trị cơ sở dữ liệu cơ bản, được nhiều nhà phát triển ứng dụng lựa chọn như một giải pháp lưu trữ dữ liệu cho các ứng dụng.

Giáo trình Hệ quản trị cơ sở dữ liệu được biên soạn theo chương trình đào tạo của Khoa Công nghệ thông tin – Trường Cao Đẳng Lý Tự Trọng TPHCM. Giáo trình này là tài liệu cho sinh viên và những người tự học nắm được các khái niệm và ý nghĩa của hệ quản trị cơ sở dữ liệu, làm quen và thực hành trên một hệ quản trị cơ sở dữ liệu cụ thể là Microsoft SQL Server.

Hy vọng giáo trình này sẽ là tài liệu bổ ích dành cho sinh viên ngành Công nghệ thông tin của trường Cao Đẳng Lý Tự Trọng.

TPHCM, ngày ... tháng ... năm 2020

Tham gia biên soạn
Ths. Lê Hồng Thái

MỤC LỤC

LỜI NÓI ĐẦU	1
MÔN HỌC: HỆ QUẢN TRỊ CƠ SỞ DỮ LIỆU	6
Mã môn học: TTC1261	6
CHƯƠNG 1: TỔNG QUAN VỀ MICROSOFT SQL SERVER.....	8
BÀI 1: TỔNG QUAN VỀ MICROSOFT SQL SERVER	8
1. Giới thiệu.....	8
2. Cài đặt SQL Server.....	8
3. SQL Server Management Studio.....	13
4. Tạo cơ sở dữ liệu	14
5. Xóa cơ sở dữ liệu	16
6. Attach /detach cơ sở dữ liệu	17
7. Backup/restore cơ sở dữ liệu.....	19
8. Import/export dữ liệu	20
CHƯƠNG 2: NGÔN NGỮ ĐỊNH NGHĨA DỮ LIỆU.....	24
BÀI 1: TẠO VÀ THAY ĐỔI CẤU TRÚC BẢNG	24
1. Khái niệm về bảng dữ liệu.....	24
1.1. Các kiểu dữ liệu	25
1.2. Tạo bảng	28
2. Chỉnh sửa cấu trúc bảng	29
2.1. Thêm cột.....	29
2.2. Xóa cột.....	29
2.3. Sửa đổi kiểu dữ liệu của cột	30
3. Xóa bảng	30
BÀI 2: TÍNH TOÀN VỆ TRONG CƠ SỞ DỮ LIỆU	31
1. Tạo ràng buộc dữ liệu trong bảng.....	31
1.1. Kiểm tra tính duy nhất của dữ liệu	31
1.2. Kiểm tra miền giá trị.....	31
1.3. Thiết lập giá trị mặc định.....	31
1.4. Kiểm tra ràng buộc toàn vẹn khóa chính.....	32
1.5. Kiểm tra ràng buộc toàn vẹn khóa ngoại.....	32

2. Tạo lược đồ CSDL.....	32
CHƯƠNG 3: NGÔN NGỮ THAO TÁC DỮ LIỆU	38
BÀI 1: CÁC LỆNH THAO TÁC VỚI DỮ LIỆU.....	38
1. Nhập dữ liệu.....	38
1.1. Cú pháp nhập một dòng dữ liệu vào một bảng.....	38
1.2. Nhập dữ liệu chuỗi, ngày tháng.....	39
1.3. Nhập dữ liệu khi có ràng buộc khóa ngoại.....	39
2. Xem và xóa dữ liệu.....	41
3. Cập nhật dữ liệu.....	41
BÀI 2: TRUY VẤN ĐƠN GIẢN	45
1. Cú pháp câu truy vấn tổng quát.....	45
2. Câu truy vấn đơn giản.....	46
3. Tìm kiếm có sắp xếp.....	47
4. Tìm kiếm với điều kiện đơn giản.....	48
5. Tìm kiếm với điều kiện liên quan đến chuỗi ký tự.....	49
6. Tìm kiếm với điều kiện liên quan đến ngày giờ.....	50
7. Sử dụng các hàm khi tìm kiếm.....	50
8. Phép kết.....	53
9. Sử dụng ALIAS.....	54
10. Sử dụng JOIN.....	54
BÀI 3: TRUY VẤN NÂNG CAO.....	57
1. Sử dụng các hàm kết hợp khi truy vấn.....	57
2. Truy vấn gom nhóm.....	57
3. Truy vấn theo điều kiện gom nhóm.....	58
4. Truy vấn con.....	58
4.1. Cú pháp.....	58
4.2. Truy vấn con trả về một giá trị.....	59
4.3. Truy vấn con trả về nhiều giá trị.....	59
5. Hàm Case.....	59
BÀI 4: BẢNG ẢO (VIEW)	63
1. Khái niệm về view.....	63
2. Lợi ích của việc sử dụng view.....	63

3. Tạo view	63
4. Xem và cập nhật view	65
5. Thay đổi và hủy bỏ view	66
BÀI 1: BIẾN VÀ CẤU TRÚC ĐIỀU KHIỂN.....	70
1. Khái niệm về biến.....	70
2. Làm việc với biến cục bộ	71
2.1. Khai báo biến.....	71
2.2. Gán giá trị cho biến.....	71
2.3. Xem giá trị hiện hành của biến.....	71
3. Các cấu trúc điều khiển	71
3.1. Beginend	71
3.2. Cấu trúc rẽ nhánh IF.....ELSE.....	72
3.3 Cấu trúc CASE	73
3.4. Cấu trúc lặp WHILE	74
BÀI 2: THỦ TỤC NỘI TẠI (STORED PROCEDURE)	75
1. Khái niệm về Stored Procedure.....	75
2. Lợi ích của việc sử dụng Stored Procedure	75
3. Các hành động cơ bản với Stored Procedure	75
3.1. Tạo Stored Procedure	75
3.2. Thực thi Stored Procedure	76
3.3. Thay đổi nội dung Stored Procedure	77
4. Tham số trong Stored Procedure	77
4.1. Stored Procedure không chứa tham số	77
4.2. Stored Procedure có một tham số	77
4.3. Stored Procedure có nhiều tham số	78
4.4. Stored Procedure có tham số đầu ra	79
5. Trả về giá trị trong Stored Procedure.....	79
5.1. Trả về giá trị từ câu lệnh Return.....	79
5.2. Trả về dữ liệu từ câu lệnh	80
BÀI 3: BÃY (TRIGGER).....	82
1. Khái niệm trigger	82
2. Các trường hợp nên dùng trigger.....	82

3. Cơ chế hoạt động của trigger	82
4. Tạo trigger	83
5. Các kiểu trigger	84
6. Thay đổi nội dung trigger.....	84
7. Xóa trigger	85

MÔN HỌC: HỆ QUẢN TRỊ CƠ SỞ DỮ LIỆU

Mã môn học: TTC1261

Thời gian thực hiện môn học: 75 giờ; (Lý thuyết: 13 giờ; Thực hành, thí nghiệm, thảo luận, bài tập: 60 giờ; Kiểm tra: 2 giờ)

Vị trí, tính chất của môn học:

- Vị trí: Học sau với học phần Cơ sở dữ liệu
- Tính chất: Là môn học bắt buộc nằm trong các học phần cơ sở. Môn học này yêu cầu phải có kiến thức về cơ sở dữ liệu

Mục tiêu học phần:

- Về kiến thức: Sau khi học xong sinh viên có các kiến thức cơ bản về:
 - + Tổng quan về Microsoft SQL Server
 - + Tạo và quản trị cơ sở dữ liệu
 - + Truy xuất và hiệu chỉnh cơ sở dữ liệu
 - + Thủ tục nội tại (Stored Procedure)
 - + Bẫy (Trigger)
- Về kỹ năng:
 - + Biết các kiến thức cơ bản và việc quản lý cơ sở dữ liệu bằng hệ quản trị CSDL SQL Server như tạo lập cơ sở dữ liệu quan hệ (CSDL) có các thành phần chính : các bảng (TABLE), các ràng buộc toàn vẹn (CONSTRAINT) trên CSDL, các bảng ảo (VIEW),..
 - + Các kỹ năng quản lý bảng (TABLE) như thao tác (thêm/xóa/sửa) dữ liệu trên bảng, quản lý dữ liệu, tạo các câu query có chọn lọc ,có nhóm, có thống kê, có khả năng tạo các Stored Procedure, Trigger để làm cơ sở cho môn lập trình cơ sở dữ liệu.
 - + Lập trình với cơ sở dữ liệu bằng ngôn ngữ lập trình T-SQL để viết các xử lý tính toán, các xử lý kiểm tra tính đúng đắn của dữ liệu và quản lý bảo mật dữ liệu trên SQL Server
- Về năng lực tự chủ và trách nhiệm:

- + Có khả năng tự tìm tòi, học hỏi, tích lũy kinh nghiệm liên quan đến môn học.
- + Tạo, truy xuất, hiệu chỉnh dữ liệu bằng ngôn ngữ T-SQL trong hệ quản trị SQL Server
- + Sử dụng được thủ tục,... trong quản trị cơ sở dữ liệu.

Nội dung môn học:

CHƯƠNG 1: TỔNG QUAN VỀ MICROSOFT SQL SERVER

🎯 MỤC TIÊU

Học xong chương này, người học biết một số khái niệm cơ bản:

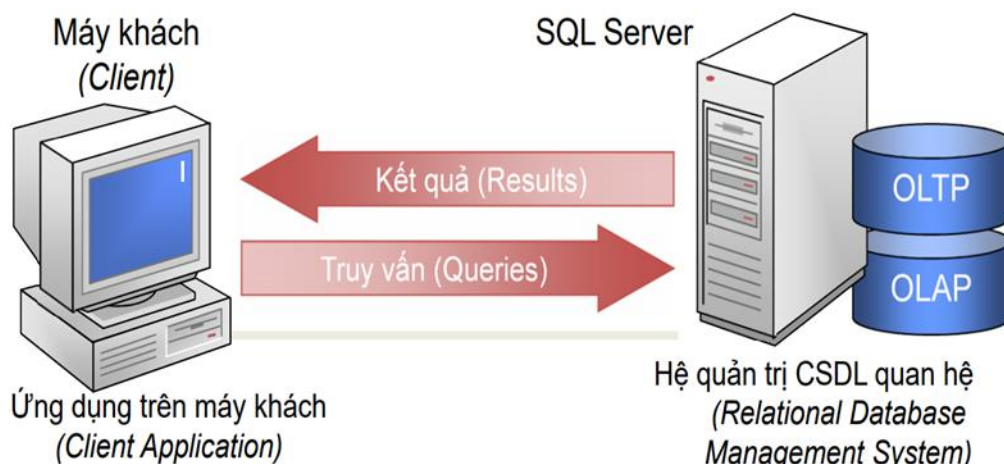
- Hiểu được Microsoft SQL Server là gì
- Cài đặt Microsoft SQL Server
- Làm quen với SQL Server Manager Studio
- Các thao cơ bản : tạo/ xóa một CSDL

Chương này giới thiệu về hệ quản trị cơ sở dữ liệu Microsoft SQL Server, trình bày các bước để cài đặt Microsoft SQL Server, làm quen với công cụ SQL Server Manager Studio. Sau đó sẽ thực hiện các thao tác cơ bản trên với công cụ SQL Server Manager Studio .

BÀI 1: TỔNG QUAN VỀ MICROSOFT SQL SERVER

1. Giới thiệu

Microsoft SQL Server (hay còn gọi SQL Server) là một hệ thống quản lý cơ sở dữ liệu quan hệ được phát triển bởi Microsoft, có chức năng chính là lưu trữ, truy xuất dữ liệu và thông qua câu lệnh SQL nó có thể trao đổi dữ liệu với các ứng dụng trên máy Client. Trong môi trường Client/Server cơ sở dữ liệu được lưu trên máy Server, người dùng truy cập dữ liệu từ các máy Client qua một hệ thống mạng.



2. Cài đặt SQL Server

SQL Server 2008 có nhiều phiên bản khác nhau, trong đó bản Express là bản thấp nhất, được Microsoft cung cấp miễn phí cho người dùng với mục đích học tập

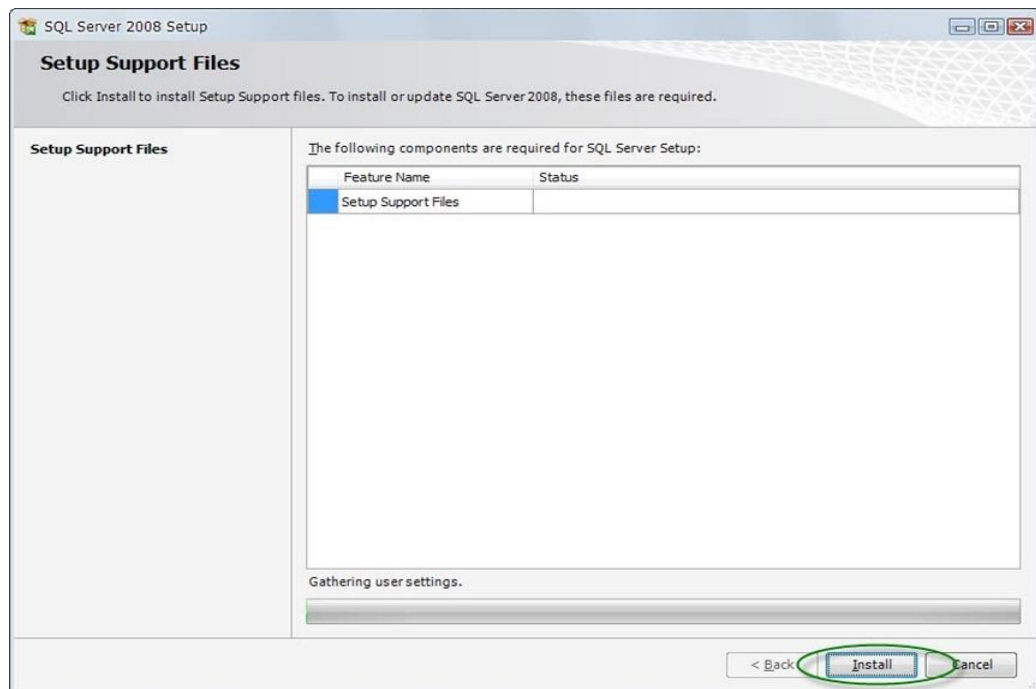
và ứng dụng vào những ứng dụng nhỏ, không yêu cầu cao về các tính năng khác ngoài việc lưu trữ và xử lý đơn giản.

Các bước cài đặt SQL Server 2008 Express

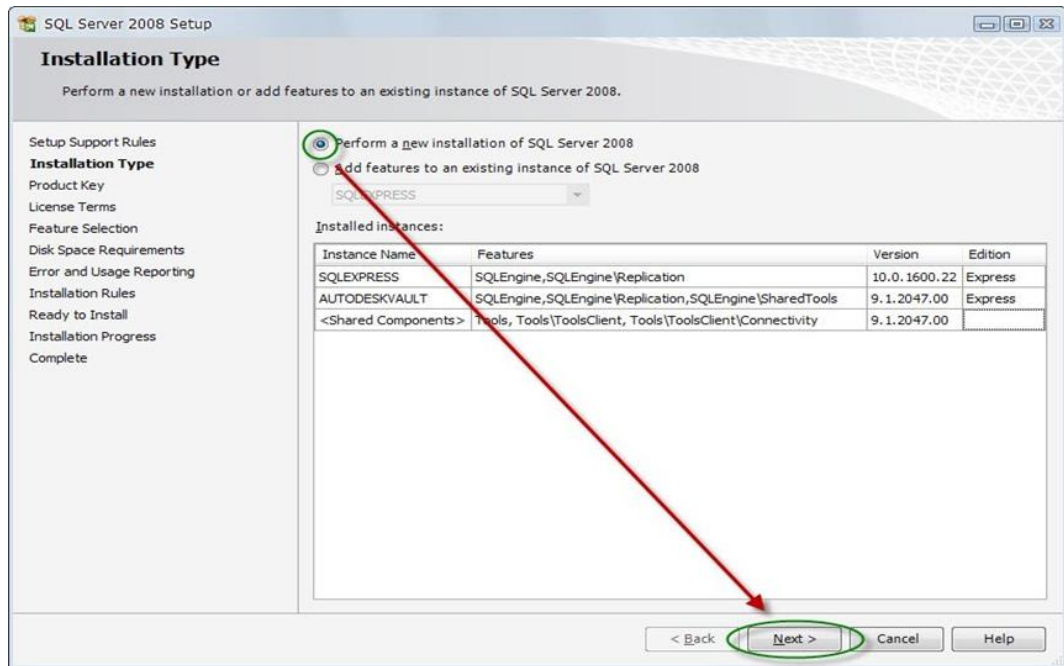
Bước 1: Chạy file setup.exe để cài đặt, chọn Installation -> New SQL Server stand-alone ...installation



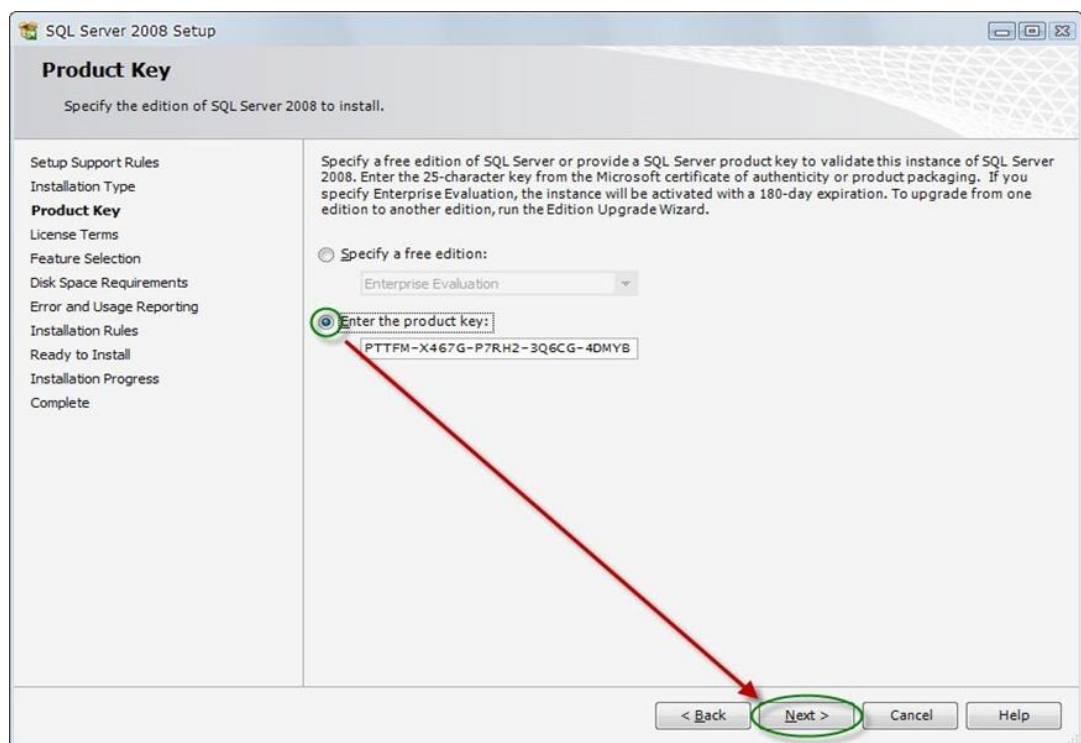
Bước 2: Chọn Ok -> Next



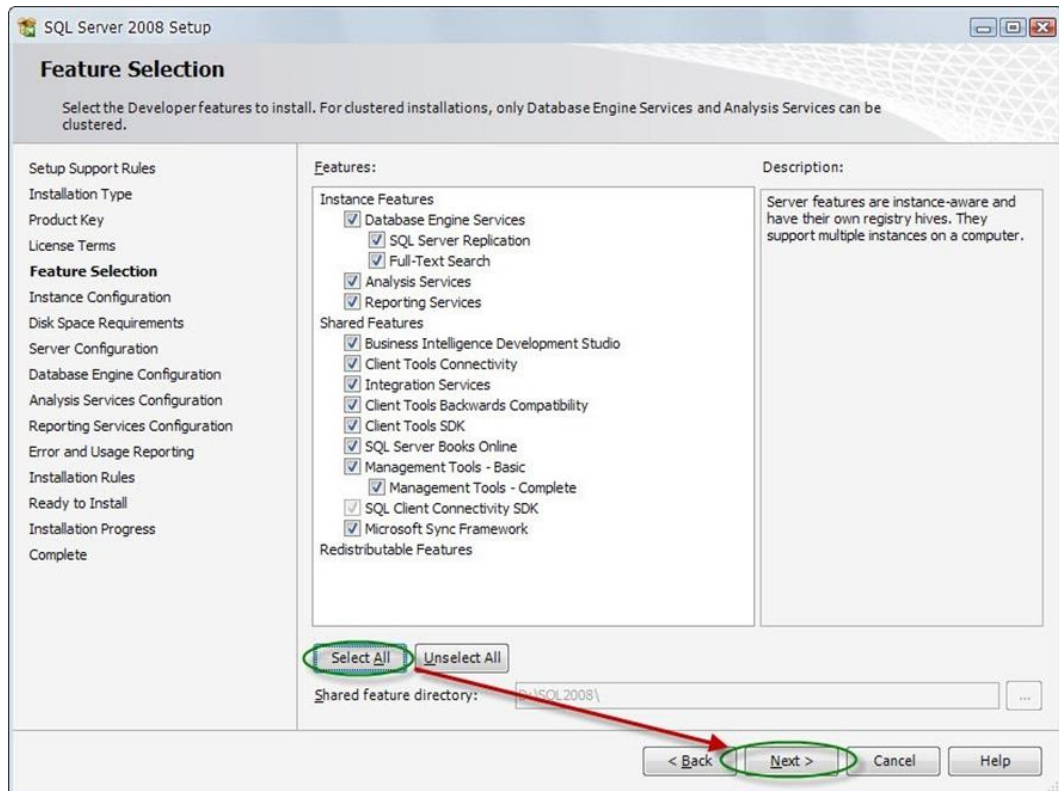
Bước 3: Chọn kiểu cài đặt mới



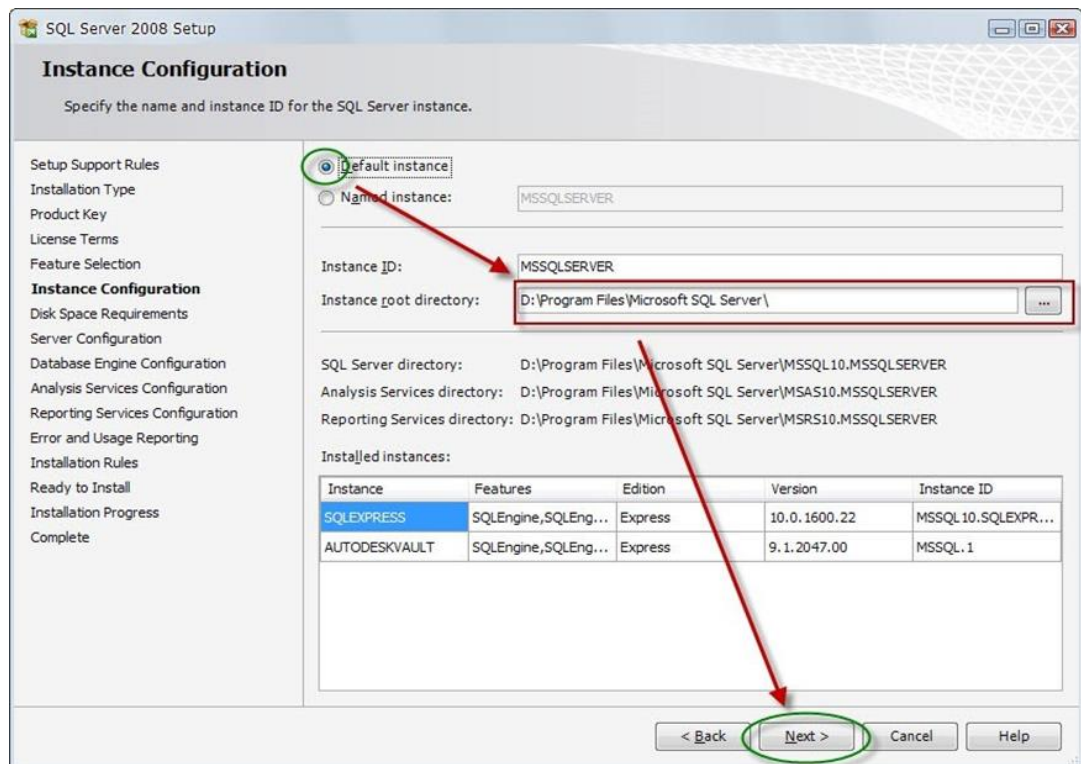
Bước 4: Nhập product key



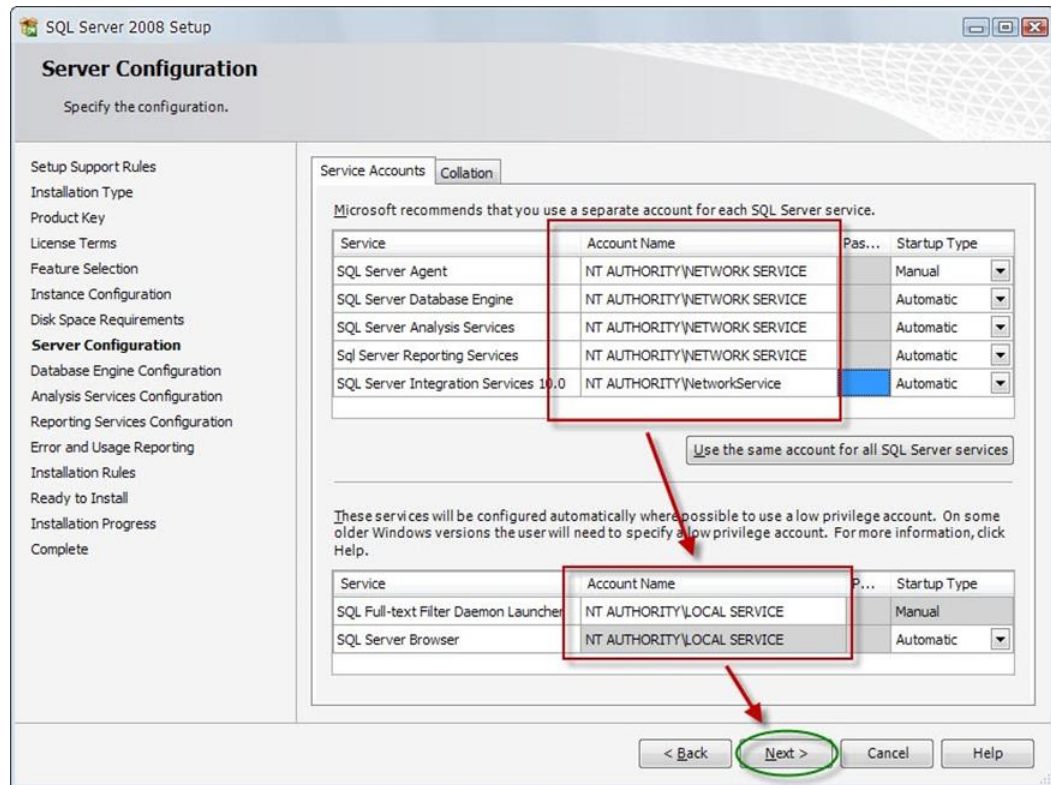
Bước 5: Sau khi đồng ý License Terms, chọn các thành phần cài đặt



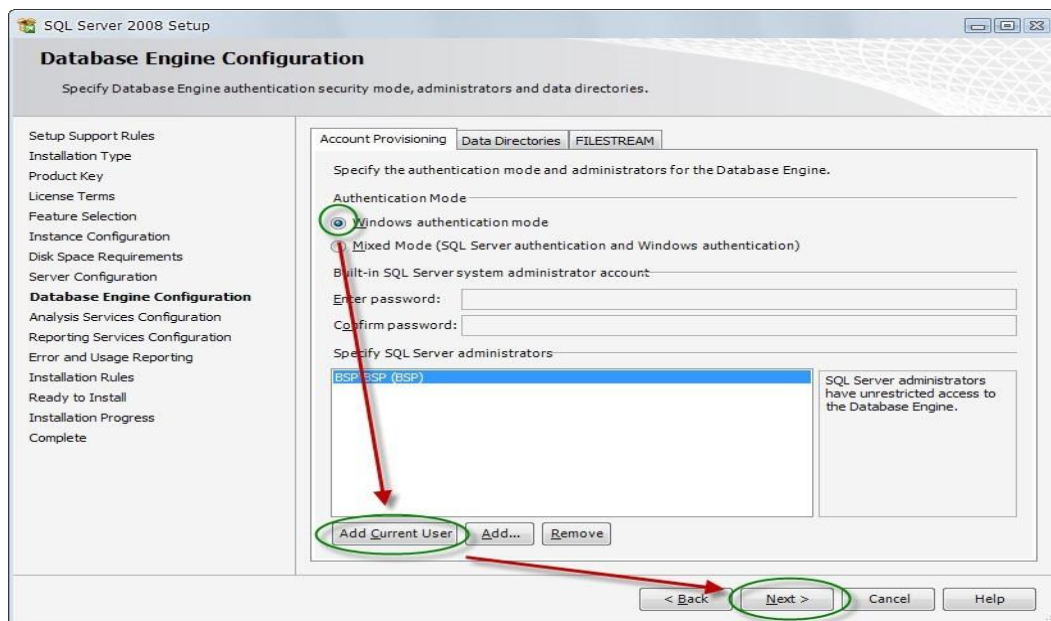
Bước 6: Thiết lập cài đặt chọn Default instance



Bước 7: Cấu hình Server

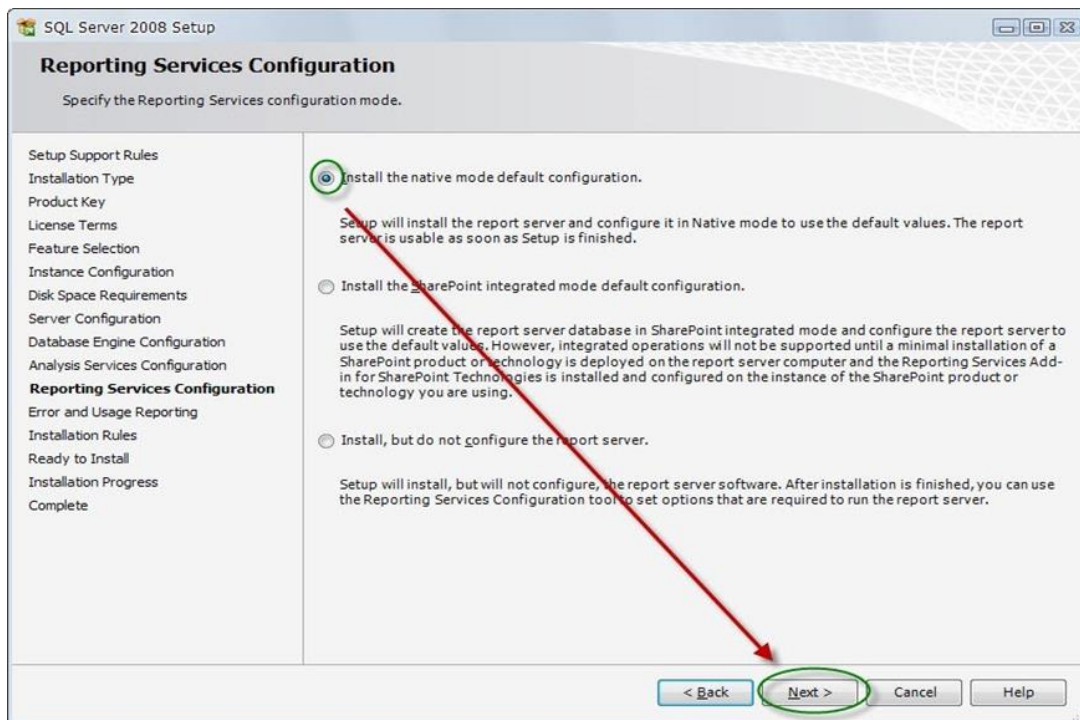


Bước 8: Cấu hình dữ liệu chọn Window Authentication và Add current User



Bước 9: Cấu hình analysis services Add Current User

Bước 10: Cấu hình report chọn option như hình nhấn Next, Next ... Cho đến khi hoàn tất

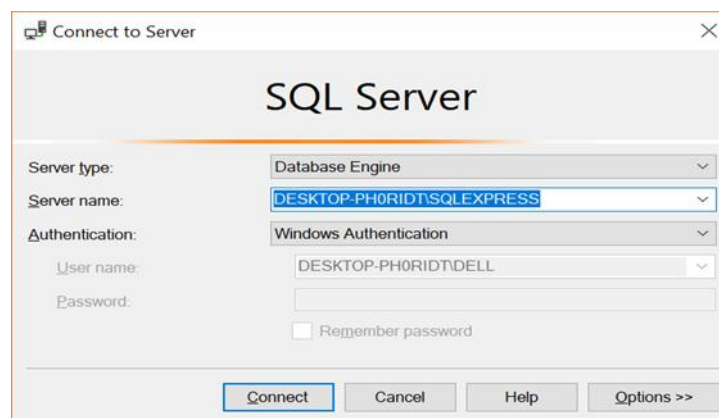


3. SQL Server Management Studio

SQL Server được kết hợp bởi nhiều thành phần riêng biệt. Các thành phần này khi phối hợp với nhau sẽ tạo thành một giải pháp hoàn chỉnh giúp cho việc lưu trữ và phân tích dữ liệu trở nên dễ dàng hơn. Một trong những công cụ quan trọng của SQL Server là SQL Server Management Studio.

Với SQL Server Management Studio chúng ta có thể thực hiện được các thao tác với CSDL bằng câu lệnh hoặc trên giao diện người dùng. SQL Server Management Studio được thiết kế đơn giản và dễ sử dụng.

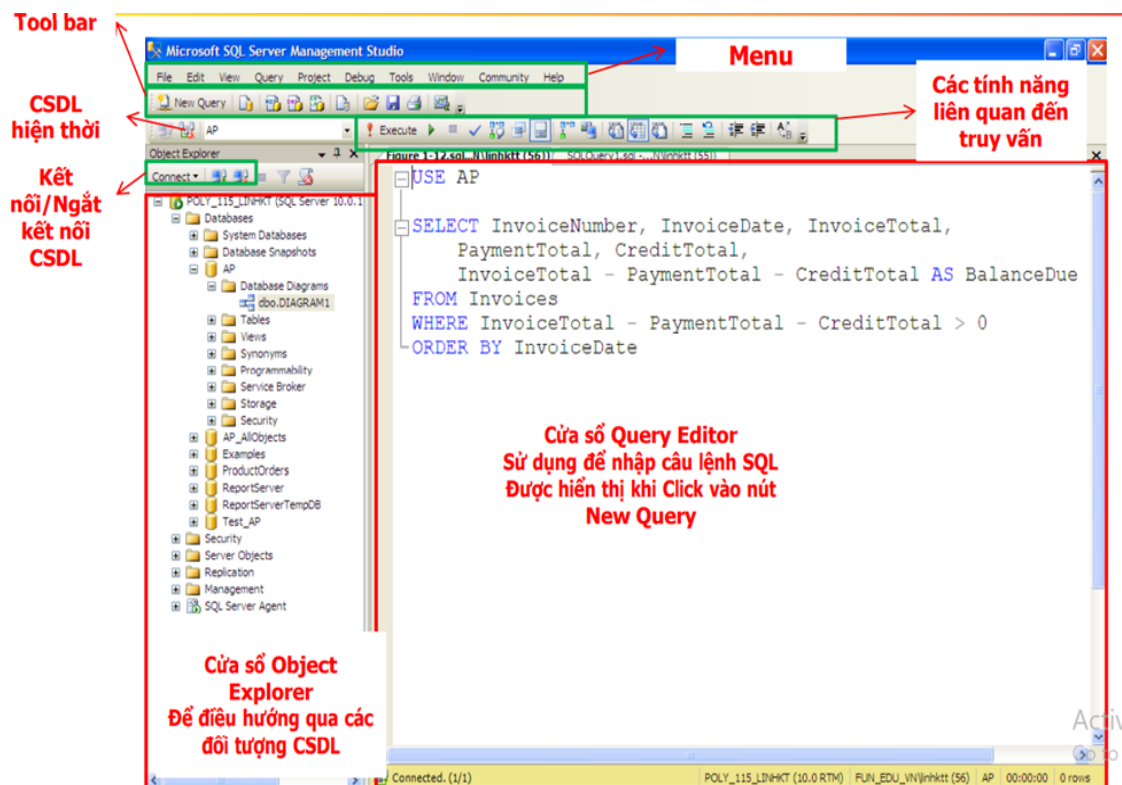
Mở SQL Server Management Studio ta làm như sau: Vào start -> Chọn program -> chọn Microsoft SQL Server -> chọn SQL Server Management Studio



Trong đó:

- **Server Type:** thường chọn là Database Engine. Các tùy chọn khác là kiểu kết nối khác đến máy chủ SQL Server.
- **Server Name:** chọn tên máy chủ SQL Server sẽ kết nối.
- **Authentication:** xác định cách xác thực đăng nhập. Khi cài đặt ở phần chọn quyền login, ta chọn Mixed Mode để có thể cho phép login bằng cả 2 quyền đó là Windows và SQL Server.
- **Chọn Connect:** Kết nối
- **Cancel:** Hủy bỏ thao tác

Sau khi nhấn nút Connect sẽ xuất hiện màn hình sau:

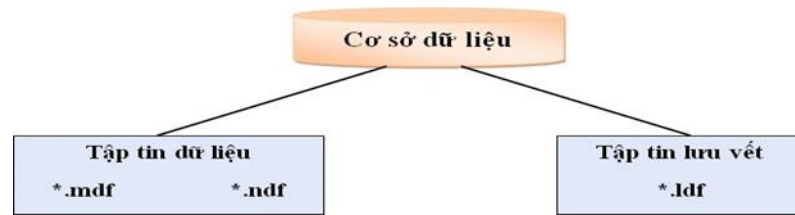


4. Tạo cơ sở dữ liệu

Một cơ sở dữ liệu trong Microsoft SQL Server tối thiểu phải bao gồm hai tập tin vật lý để lưu trữ dữ liệu.

- Một dùng để lưu trữ dữ liệu (data file).

- Một dùng để lưu vết các giao tác mà người dùng đã thực hiện (transaction log file)



Các tập tin lưu trữ cơ sở dữ liệu bên trong Microsoft SQL Server được phân chia thành ba loại tập tin vật lý khác nhau:

Tập tin dữ liệu chính (Primary Data File)

Đây là tập tin chính dùng để lưu trữ các thông tin hệ thống của cơ sở dữ liệu và phần còn lại dùng lưu trữ một phần dữ liệu. Phần mở rộng của tập tin này thông thường là *.mdf.

Tập tin dữ liệu thứ yếu (Secondary Data Files)

Đây là tập tin dùng để lưu trữ các đối tượng dữ liệu không nằm trong tập tin dữ liệu chính. Loại tập tin này không bắt buộc phải có khi tạo mới cơ sở dữ liệu. Phần mở rộng của tập tin này thông thường là *.ndf.

Tập tin lưu vết (Log Files)

Đây là tập tin dùng để lưu vết các giao tác, là những hành động cập nhật dữ liệu (thêm, sửa, xóa) vào các bảng do người dùng tác động trên cơ sở dữ liệu. Tập tin này hỗ trợ cho phép bạn có thể hủy bỏ (Rollback) các thao tác cập nhật dữ liệu vừa mới thực hiện. Điều này giúp cho dữ liệu không hề bị thay đổi. Phần mở rộng của tập tin này thông thường là *.ldf.

Mặc định các tập tin dữ liệu của cơ sở dữ liệu Microsoft SQL Server sẽ được lưu trữ trong thư mục C:\Program Files\Microsoft SQL Server\MSSQL\Data hoặc đường dẫn mà người dùng đã chỉ định lại trong quá trình cài đặt Microsoft SQL Server.

Người sử dụng cũng có thể thiết lập vị trí lưu trữ các tập tin dữ liệu trong lúc tạo mới CSDL.

❖ Tạo CSDL đơn giản không tham số:

Create Database <Tên CSDL>

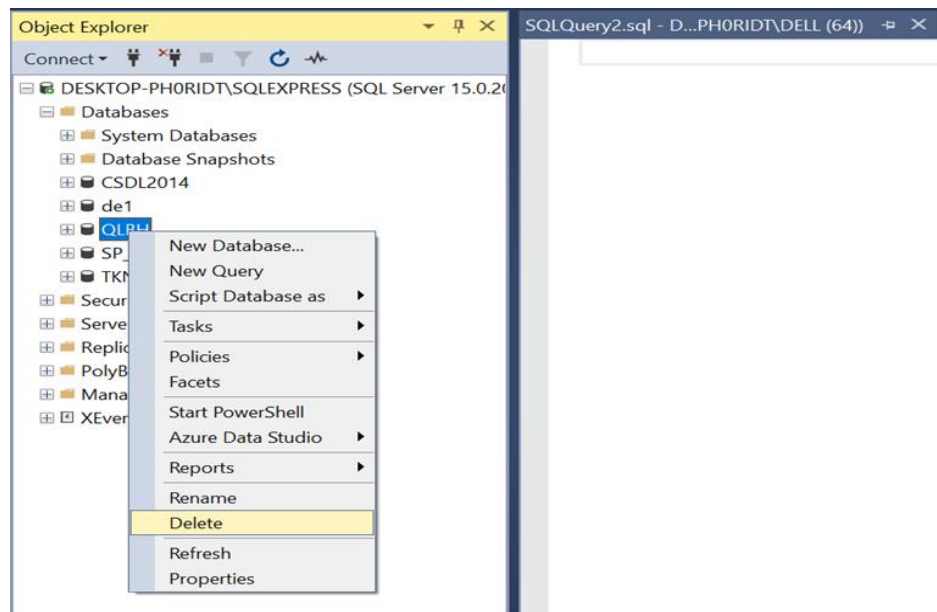
❖ Tạo CSDL chỉ định đường dẫn nơi chứa CSDL

Cú pháp	Ví dụ
CREATE DATABASE <Tên CSDL> on primary -- <i>Tạo tập tin data</i> (name = tên_csdl, filename = 'đường_dẫn\tên_csdl.mdf', size = kích_thước_ban_đầu, maxsize = kích_thước_tối_đa, filegrowth = kích_thước_tăng,) log on -- <i>Tạo tập tin log</i> (name = tên_csdl_log, filename = 'đường_dẫn\tên_csdl.ldf', size = kích_thước_ban_đầu, maxsize = kích_thước_tối_đa, filegrowth = kích_thước_tăng)	CREATE DATABASE QLBH ON (NAME = 'QLBH_Data', FILENAME = 'D:\QLBH_Data.mdf', SIZE = 10MB, MAXSIZE = UNLIMITED, FILEGROWTH = 5MB) LOG ON (NAME = 'QLBH_Log', FILENAME = 'D:\QLBH_Log.ldf', SIZE = 5MB, MAXSIZE = UNLIMITED, FILEGROWTH = 2MB)

5. Xóa cơ sở dữ liệu

Xóa CSDL là hình thức gỡ bỏ CSDL ra khỏi SQL Server, đồng thời, các tập tin lưu trữ cũng sẽ bị xóa khỏi bộ nhớ.

❖ Xóa CSDL bằng tiện ích SQL Server Management Studio



- **Bước 1:** Click chuột phải vào tên CSDL muốn xóa, sau đó click Delete
- **Bước 2:** Xác nhận CSDL đã chọn để xóa, sau đó nhấn OK

❖ Xóa CSDL bằng câu lệnh T-SQL

Click vào biểu tượng New Query trên tiện ích SQL Server Management Studio, tại cửa sổ soạn thảo, nhập dòng lệnh có cấu trúc như sau và nhấn nút Execute để thực thi.

DROP DATABASE Tên_CSDL

Trong đó: Tên_CSDL chính là tên CSDL cần xóa

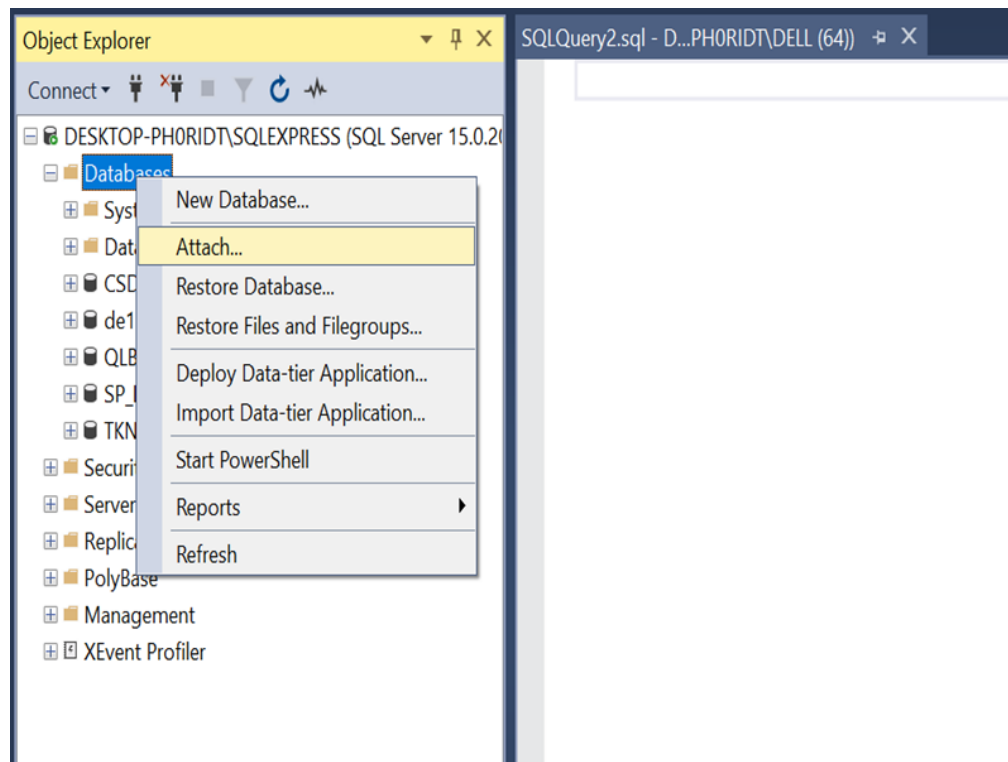
6. Attach /detach cơ sở dữ liệu

❖ **Attach File:** Tạo một CSDL mới sử dụng file CSDL đã có.

Attach CSDL là hình thức gắn CSDL vào SQL Server bằng các tập tin dữ liệu *.mdf và *.ldf có sẵn. Khi đó, một CSDL mới sẽ được tạo ra trên SQL Server với dữ liệu đã được lấy từ file *.mdf.

▪ **Attach CSDL bằng tiện ích SQL Server Management Studio**

- Bước 1: Click chuột phải vào Databases, sau đó click Attach...
- Bước 2: Tại hộp thoại Attach Databases, nhấn nút Add, chọn file *.mdf cần Attach, sau đó nhấn OK



▪ **Attach CSDL bằng câu lệnh T-SQL**

Click vào biểu tượng New Query trên tiện ích SQL Server Management Studio, tại cửa sổ soạn thảo, nhập dòng lệnh có cấu trúc như sau và nhấn nút Execute để thực thi.

```
CREATE DATABASE Tên_CSDL
ON (FILENAME = 'Path1\*.mdf'),
(FILENAME = 'Path2\*.ldf')
FOR ATTACH;
```

Trong đó:

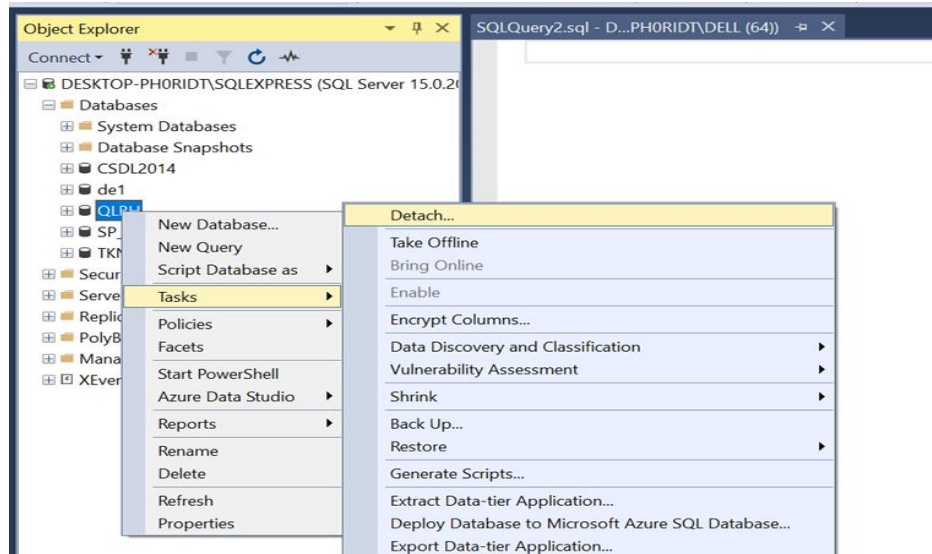
- Tên_CSDL chính là tên CSDL cần xóa
- Path1*.mdf: đường dẫn đến file mdf
- Path2*.ldf: đường dẫn đến file ldf

❖ **Detach File:** Gỡ (detach) CSDL là hình thức gỡ bỏ CSDL ra khỏi SQL Server, tuy nhiên, các tập tin lưu trữ vẫn còn tồn tại trong bộ nhớ.

▪ **Gỡ CSDL bằng tiện ích SQL Server Management Studio**

Bước 1: Click chuột phải vào tên CSDL muốn gỡ, sau đó click Tasks / Detach

Bước 2: Trong hộp thoại Detach Database, nhấn OK để tiến hành gỡ bỏ CSDL, khỏi SQL Server.



▪ **Gỡ CSDL bằng câu lệnh T-SQL**

Click vào biểu tượng New Query trên tiện ích SQL Server Management Studio, tại cửa sổ soạn thảo, nhập dòng lệnh có cấu trúc như sau và nhấn nút Execute để thực thi.

```
EXEC sp_detach_db 'Tên_CSDL', 'true';
```

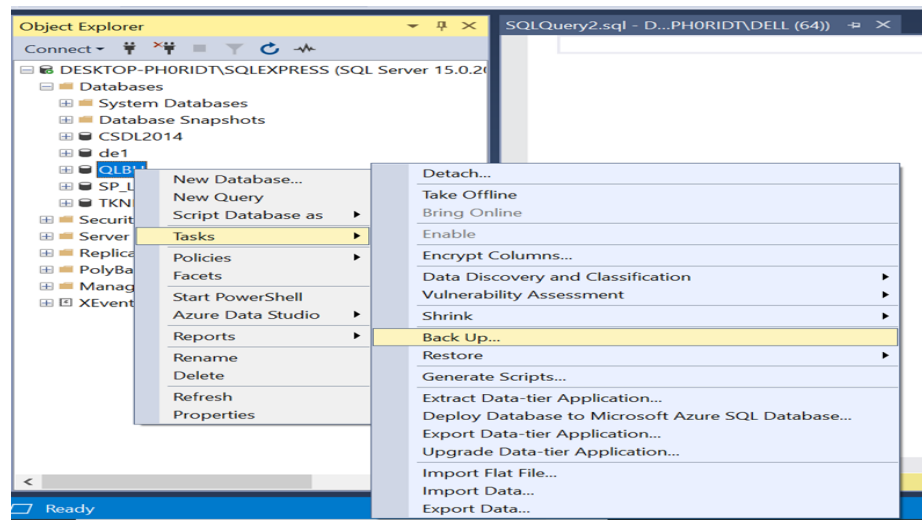
Trong đó: **Tên_CSDL** chính là tên CSDL cần detach

7. Backup/restore cơ sở dữ liệu

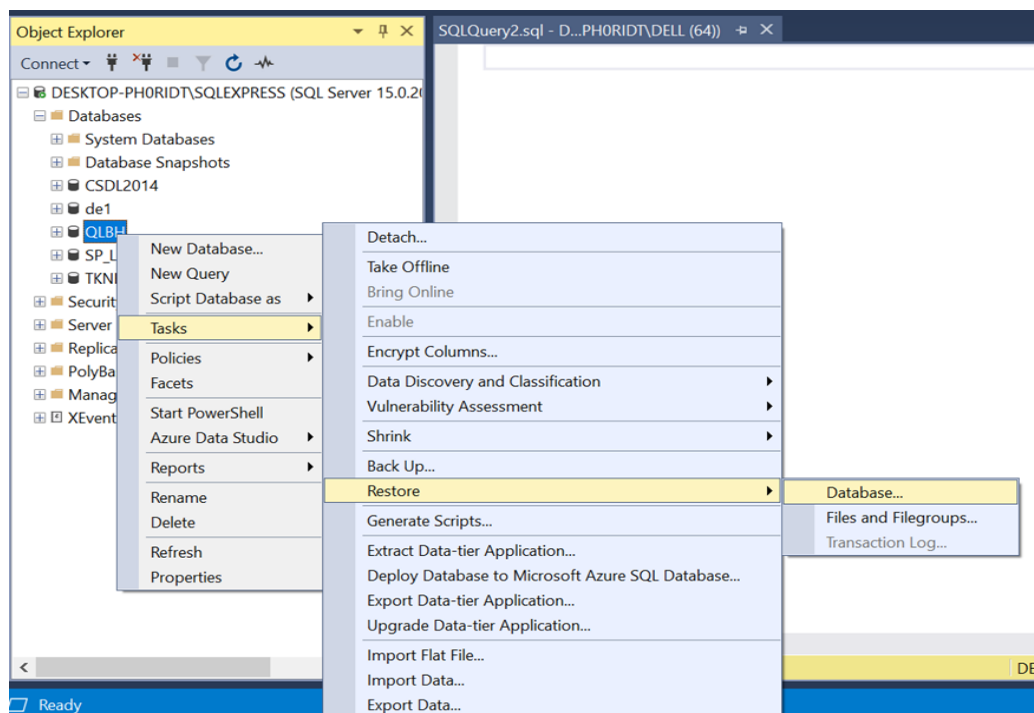
❖ **Backup** (sao lưu): SQL Server cung cấp 3 loại Back up:

- Full backup: Là backup toàn bộ dữ liệu tại thời điểm thực hiện.
- Differential backup: Là backup các trang dữ liệu mới được cập nhật kể từ lần backup full trước đó.
- File or File Group Backup: Copy một data file/một file group hay Standby thì mặc định Recovery sẽ được thực hiện.

Click phải vào CSDL cần back up -> Chọn Tasks -> Chọn Back up...



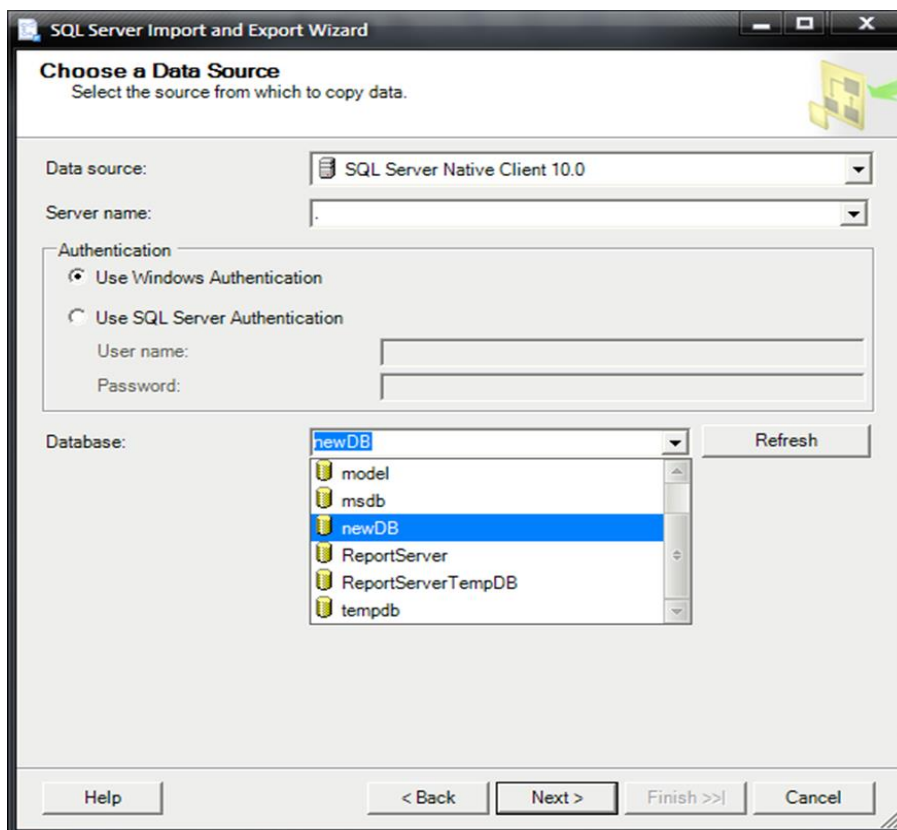
❖ **Restore** (Phục hồi):Nhấn chuột phải vào CSDL cần Restore -> Chọn Tasks -> Chọn Restore -> Database...



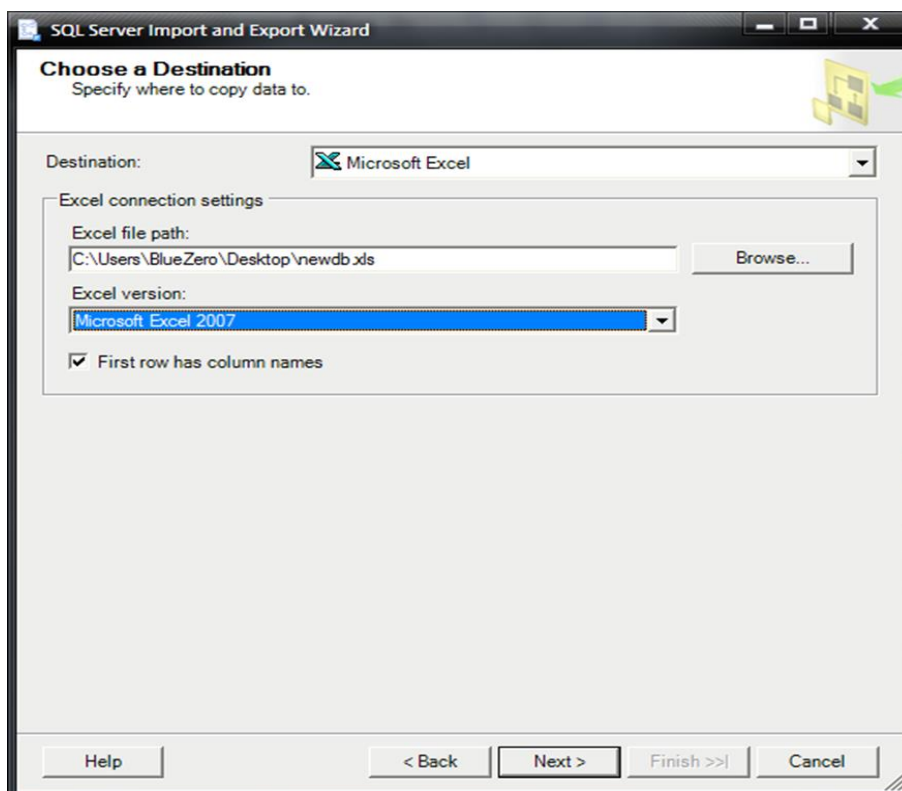
8. Import/export dữ liệu

Tính năng này cho phép xuất dữ liệu ra nhiều định dạng khác nhau (text file, access, excel, sql server...)

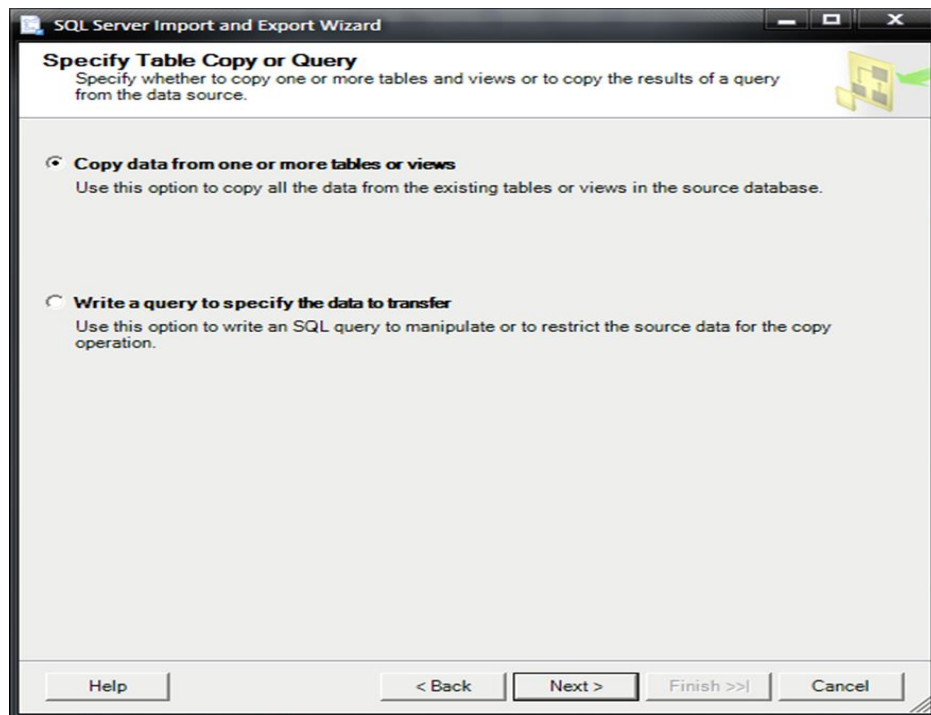
❖ **Export**: Click phải vào database> task> export. Nhập thông tin kết nối và chọn cơ sở dữ liệu cần export



Chọn và mô tả loại, nơi lưu cơ sở dữ liệu đích



Nhấn next, chọn phương thức sao chép từ bảng/view trong cơ sở dữ liệu nguồn hoặc phương thức viết câu truy vấn để lấy dữ liệu từ nguồn, ở đây ta chọn phương thức thứ nhất.



Sau đó nhấn next, chọn các bảng hoặc view cần export. Và đây là kết quả dữ liệu được xuất ra tập tin excel (bao gồm 2 sheet tương ứng với các bảng đã chọn export)

	A	B	C	D	E	F	G
1	mahs	tenhs	diachi	malop			
2		1 Nguyen A	DBP	1			
3		2 Tran B	NTMK	2			
4							
5							

❖ **Import:** thực hiện tương tự.

BÀI TẬP KẾT THÚC CHƯƠNG

LÝ THUYẾT

1. Trong SQL Server có bao nhiêu cơ sở dữ liệu hệ thống? Trình bày chức năng của từng cơ sở dữ liệu hệ thống.
2. Cơ sở dữ liệu được lưu trữ trong SQL Server như thế nào? Chức năng của tập tin dữ

liệu chính (mdf), tập tin dữ liệu thứ cấp (ndf), tập tin lưu vết (ldf).

3. Trong trường hợp nào, người sử dụng nên thực hiện chức năng attach, detach cơ sở dữ liệu?

THỰC HÀNH

1. Tạo CSDL Quản lý đề án có các tham số như sau:

THAM SỐ	GIÁ TRỊ
Database name	QuanLyDeAn
Tên logic của data file chính	QuanLyDeAn_Data
Tên tập tin và đường dẫn của data file chính	D:\HoTenSV\QuanLyDeAn_Data.mdf
Kích cỡ khởi tạo của CSDL	20 MB
Kích cỡ tối đa của CSDL	40 MB
Kích thước gia tăng tập tin CSDL	1 M B
Tên logic của transaction log	QuanLyDeAn_Log
Tên tập tin và đường dẫn của transaction log	D:\HoTenSV\QuanLyDeAn_Log.ldf
Kích cỡ khởi tạo của transaction log	6 MB
Kích cỡ tối đa của transaction log	8 MB
Kích thước gia tăng tập tin transaction log	1 MB

2. Sử dụng Windows Explorer để kiểm tra các tập tin dữ liệu của CSDL vừa tạo
3. Thực hiện xóa CSDL QL_DEAN và kiểm tra sự tồn tại các tập tin dữ liệu của CSDL vừa xóa
4. Thực hiện Attach một CSDL với các tập tin dữ liệu được giáo viên cung cấp.
5. Thực hiện Detach CSDL vừa tạo, kiểm tra sự tồn tại của CSDL vừa detach trên SQL Server.

CHƯƠNG 2: NGÔN NGỮ ĐỊNH NGHĨA DỮ LIỆU



MỤC TIÊU

Học xong chương này, người học có khả năng:

- Tạo bảng
- Thay đổi cấu trúc bảng
- Tạo các ràng buộc trong bảng

Chương này gồm các câu lệnh cho phép người sử dụng định nghĩa CSDL và các đối tượng trong CSDL như các bảng, ...

BÀI 1: TẠO VÀ THAY ĐỔI CẤU TRÚC BẢNG

1. Khái niệm về bảng dữ liệu

Table (Bảng) là một đối tượng trong cơ sở dữ liệu SQL Server, dùng để lưu trữ các thông tin dữ liệu của những đối tượng, thực thể trong thế giới thực vào máy tính. Ví dụ như các thông tin về khách hàng, nhà cung cấp, hóa đơn...

❖ **Cấu trúc bảng:** Các thông tin sẽ được tổ chức thành các dòng (row) và các cột (column) giống như định dạng của bảng tính Excel.

Khi làm việc với bảng, cần phải quan tâm đến các thuộc tính sau:

- Tên bảng (table name): có độ dài không quá 128 ký tự.
- Cột (column): tùy vào lượng thông tin cần lưu trữ, mỗi bảng sẽ có một số lượng cột nhất định, mỗi cột có một tên (column name) và kiểu dữ liệu (data type) xác định.
- Kiểu dữ liệu (data type): quy định kiểu dữ liệu mà cột sẽ lưu trữ ở bên trong bảng.

❖ **Khóa chính, khóa ngoại:**

Mỗi dòng dữ liệu trong Table thường phải là duy nhất, do đó sẽ có một hoặc nhiều cột bên trong Table sẽ tham gia làm khóa chính (primary key). Giá trị dữ liệu tại các cột tham gia khóa chính sẽ không được trùng lặp bên trong Table.

Ví dụ:

Mỗi sinh viên đều có một mã số sinh viên duy nhất. Do đó, đối với Table được thiết kế để lưu thông tin sinh viên, cột Mã số sinh viên thường được sử dụng làm khóa chính.

Hầu như các bảng trong một CSDL sẽ có quan hệ với các bảng khác nhằm kiểm tra tính tồn tại dữ liệu và trao đổi, chia sẻ thông tin với nhau. Mỗi bảng sẽ có một hoặc một vài cột được xác định làm khóa ngoại (foreign key) tham chiếu đến khóa chính của một bảng khác.

Ví dụ:

Cột Lop trong table SINHVIEN là khóa ngoại tham chiếu đến khóa chính của bảng LOP.

Table SINHVIEN

<u>MSS</u> <u>V</u>	HoDem	Ten	NgaySinh	GioiTinh	<u>L</u> <u>o</u> <u>p</u>
SV01	Cao	Mìn h	3/2/1994	Nam	L 1
SV02	Võ Tấn	Hùn g	30/4/1995	Nam	L 2
SV03	Cao Thanh	Loan	19/8/1995	Nữ	L 2
SV04	Hồ	Cườ ng	2/9/1996	Nam	L 1

Table LOP

<u>Ma</u> <u>Lo</u> <u>p</u>	TenLop	K ho a
L1	Lớp thứ 1	16
L2	Lớp thứ 2	16

1.1. Các kiểu dữ liệu

Có hai loại kiểu dữ liệu: kiểu dữ liệu cơ sở và kiểu dữ liệu do người dùng định nghĩa. Kiểu dữ liệu cơ sở là kiểu dữ liệu mà SQL Server cung cấp. Nội dung phần này trình bày một số kiểu dữ liệu cơ sở thường dùng.

Mục	Kiểu dữ liệu	Mô tả
Exact Numbers	int	Một cột được khai báo kiểu dữ liệu này sẽ sử dụng 4 byte trong bộ nhớ máy tính. Nó thường được sử dụng để lưu trữ giá trị số nguyên.
	smallint	Một cột được khai báo kiểu dữ liệu này sẽ sử dụng 2 byte trong bộ nhớ máy tính. Nó có thể lưu trữ các số nguyên từ -32768 đến 32767.
	tinyint	Một cột của kiểu này chiếm 1 byte trong bộ nhớ. Có giá trị từ 0 đến 255
	bigint	Một cột được khai báo kiểu dữ liệu này sẽ sử dụng 8 byte trong bộ nhớ máy tính. Nó có thể lưu trữ các số nguyên từ -2^{63} (-9223372036854775807) đến $2^{63}-1$.
	numeric	Một cột được khai báo kiểu dữ liệu này sẽ có độ chính xác cao và có thể co giãn kích thước lưu trữ trong bộ nhớ.
	money	Một cột được khai báo kiểu dữ liệu này sẽ sử dụng 8 byte trong bộ nhớ máy tính. Biểu diễn giá trị dữ liệu tiền tệ từ $(-2^{63}/10000)$ đến $(2^{63}-1)$.
Approximate numerics	float	Một cột được khai báo kiểu dữ liệu này sẽ sử dụng 8 byte trong bộ nhớ máy tính. Biểu diễn các số chấm động từ $-1.79E+308$ đến $1.79E+308$.
	real	Một cột được khai báo kiểu dữ liệu này sẽ sử dụng 4 byte trong bộ nhớ máy tính. Biểu diễn các số chấm động có độ chính xác từ $-3.4E+38$ đến $3.40E+38$.
Date and time	datetime	Biểu diễn ngày và giờ. Được lưu trữ như là 2 số integer, chiếm 8 byte.
	smalldatetime	Biểu diễn ngày và time.
Character String	char	Lưu trữ dữ liệu ký tự, nó được cố định kích thước và không hỗ trợ Unicode.
	varchar	Lưu trữ dữ liệu ký tự, độ dài có thể thay đổi và không hỗ trợ Unicode.

	text	Lưu trữ dữ liệu kí tự, độ dài có thể thay đổi và không hỗ trợ Unicode
Unicode Types	nchar	Lưu trữ dữ liệu kí tự, nó được cố định kích thước và có hỗ trợ Unicode.
	nvarchar	Lưu trữ dữ liệu kí tự, độ dài có thể thay đổi và có hỗ trợ Unicode.
Các kiểu dữ liệu khác	Timestamp	Một cột được khai báo kiểu dữ liệu này sẽ sử dụng 8 byte trong bộ nhớ máy tính. Nó chứa các số binary tự động phát sinh (mỗi hàng là một số duy nhất).
	binary(n)	Lưu trữ dữ liệu binary có độ dài cố định với độ dài tối đa là 8000byte
	varbinary(n)	Lưu trữ dữ liệu binary có độ dài thay đổi với độ dài tối đa là 8000byte.
	image	Lưu trữ dữ liệu binary có độ dài thay đổi với độ dài tối đa là $(2^{30}-1)$ byte.
	uniqueidentifie	Một cột được khai báo kiểu dữ liệu này sẽ sử dụng 16 byte trong bộ nhớ máy tính. Ngoài ra nó lưu trữ một GUID (Globally Unique Identifier)

1.2. Tạo bảng

❖ Cú pháp lệnh tạo bảng:

```
CREATE TABLE Table_Name
(
    Column_Name_1    Data_Type_1[NOT NULL],
    Column_Name_2    Data_Type_2[NOT NULL],
    [...],
    PRIMARY KEY (Column_Name)
)
```

Trong đó:

- Table_Name: Tên bảng

- Column_Name_n: Tên cột thứ n trong bảng
- Column_Name: tên cột được chọn làm khóa chính

❖ **Một số lưu ý khi tạo bảng bằng câu lệnh T-SQL:**

- Thêm từ khóa NOT NULL đối với các cột bắt buộc phải chứa dữ liệu (không được để trống).
- Đối với khóa chính gồm nhiều cột, các cột được liệt kê sau từ khóa PRIMARY KEY và cách nhau bởi dấu phẩy.
- Đối với khóa chính chỉ chứa một cột, có thể đặt từ khóa PRIMARY KEY ngay sau khi khai báo.

Ví dụ: câu lệnh tạo bảng NHANVIEN

```
CREATE TABLE NHANVIEN
(
    MaNV varchar(9) NOT NULL PRIMARY KEY,
    HoNV nvarchar(15),
    TenLot nvarchar(30),
    TenNV nvarchar(30),
    NgSinh datetime,
    DiaChi nvarchar(150),
    Phai nvarchar(3),
    Luong numeric(18,0),
    MaNQL varchar(9),
    Phong varchar(2)
)
```

2. Chỉnh sửa cấu trúc bảng

Có thể sử dụng câu lệnh ALTER TABLE để chỉnh sửa cấu trúc bảng.

2.1. Thêm cột

```
ALTER TABLE TableName
ADD ColumnName DataType [NOT NULL]
```

2.2. Xóa cột

```
ALTER TABLE TableName
DROP COLUMN ColumnName
```

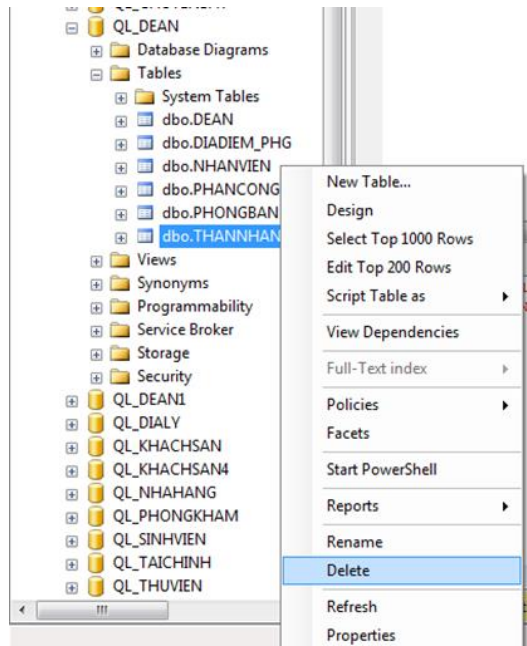
2.3. Sửa đổi kiểu dữ liệu của cột

3. Xóa bảng

```
ALTER TABLE TableName
ALTER COLUMN ColumnName NewDataType
```

Để xóa bảng ra khỏi CSDL, R-click vào tên bảng cần xóa, sau đó chọn Delete.

Nhấn OK trong hộp thoại Delete Object để xác nhận việc xóa bảng khỏi CSDL.



Có thể xóa bảng bằng câu lệnh DROP TABLE. Câu lệnh DROP TABLE có cấu trúc như sau:

```
DROP TABLE TableName
```

BÀI 2: TÍNH TOÀN VỆ TRONG CƠ SỞ DỮ LIỆU

1. Tạo ràng buộc dữ liệu trong bảng

Ràng buộc (Constraint) toàn vẹn dữ liệu là các quy tắc trong CSDL nhằm kiểm tra các giá trị dữ liệu trước khi được lưu trữ phải đảm bảo tính chính xác và hợp lý. Một thao tác thay đổi dữ liệu (thêm, xóa, cập nhật) sẽ không được thực thi nếu thao tác đó tác động đến dữ liệu làm mất đi tính toàn vẹn dữ liệu.

Constraint là một đối tượng trong CSDL có tác dụng kiểm tra tính toàn vẹn dữ liệu trong CSDL. Việc tạo một Constraint có thể được thực hiện trong quá trình tạo bảng hoặc chỉnh sửa bảng.

1.1. Kiểm tra tính duy nhất của dữ liệu

Ví dụ: Hai nhân viên trong công ty không thể có cùng số CMND. Một Constraint trong bảng NHANVIEN để kiểm tra thuộc tính SoCMND (Số CMND) là duy nhất được tạo như sau:

```
ALTER TABLE NHANVIEN  
ADD CONSTRAINT UNQ_NHANVIEN_SoCMND UNIQUE (SoCMND)
```

Trong đó: UNQ_NHANVIEN_SoCMND là tên Constraint, được đặt tùy ý.

1.2. Kiểm tra miền giá trị

Ví dụ: Lương nhân viên phải là một số nguyên dương từ 2000000 đến 15000000. Một Constraint trong bảng NHANVIEN để kiểm tra miền giá trị của thuộc tính Lương (Lương) được tạo như sau:

```
ALTER TABLE NHANVIEN  
ADD CONSTRAINT CHK_NHANVIEN_Luong  
CHECK (Luong BETWEEN 2000000 TO 15000000)
```

1.3. Thiết lập giá trị mặc định

Ví dụ: Một nhân viên có thể không có số điện thoại. Một Constraint trong bảng NHANVIEN để thiết lập giá trị mặc định cho thuộc tính SoDT là “Chưa có” được tạo như sau:

```
ALTER TABLE NHANVIEN  
ADD CONSTRAINT DEF_NHANVIEN_SoDT  
DEFAULT 'Chưa có' FOR SoDT
```

1.4. Kiểm tra ràng buộc toàn vẹn khóa chính

Ràng buộc toàn vẹn khóa chính được thiết lập trong quá trình tạo bảng. Nếu chưa thiết lập khóa chính, có thể tạo một Constraint để kiểm tra ràng buộc toàn vẹn khóa chính.

Ví dụ: Một Constraint kiểm tra ràng buộc toàn vẹn khóa chính trong bảng NHANVIEN được thực hiện như sau:

```
ALTER TABLE NHANVIEN  
ADD CONSTRAINT PK_NHANVIEN_MaNV  
PRIMARY KEY (MaNV)
```

1.5. Kiểm tra ràng buộc toàn vẹn khóa ngoại

Ví dụ: Một Constraint để kiểm tra ràng buộc toàn vẹn khóa ngoại MaPhong của bảng NHANVIEN tham chiếu đến khóa chính MaPhong của bảng PHONGBAN như sau:

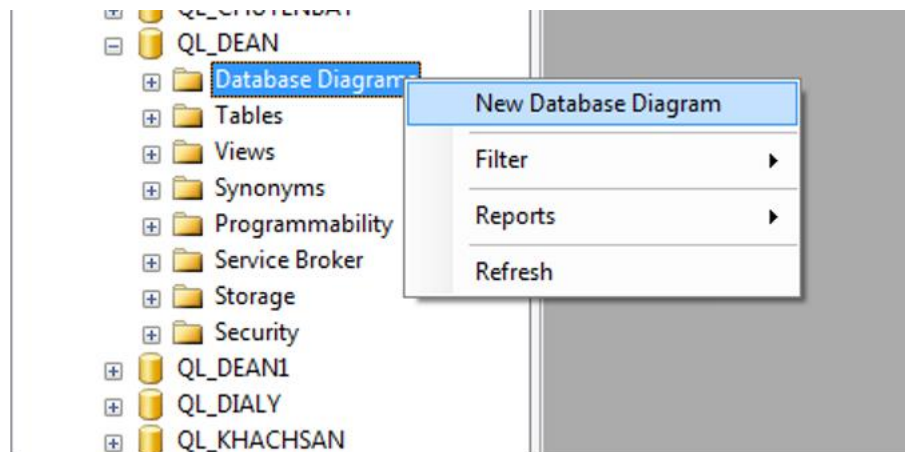
```
ALTER TABLE NHANVIEN  
ADD CONSTRAINT FK_NHANVIEN_PHONGBAN  
FOREIGN KEY (MaPhong) REFERENCES PHONGBAN (MaPhong)
```

2. Tạo lược đồ CSDL

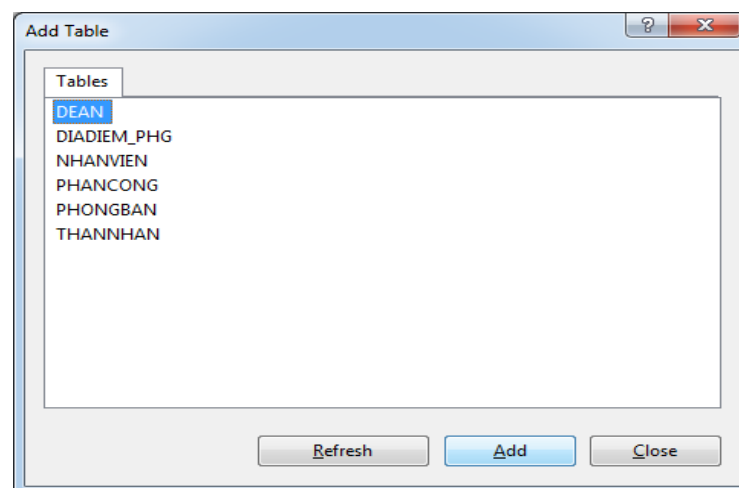
Đối với các bảng đã tạo ràng buộc khóa ngoại, các mối kết hợp bên trong lược đồ cơ sở dữ liệu sẽ được tự động tạo ra khi người dùng chèn các bảng vào biểu đồ này.

❖ **Các bước tạo mới biểu đồ cơ sở dữ liệu bên trong một cơ sở dữ liệu:**

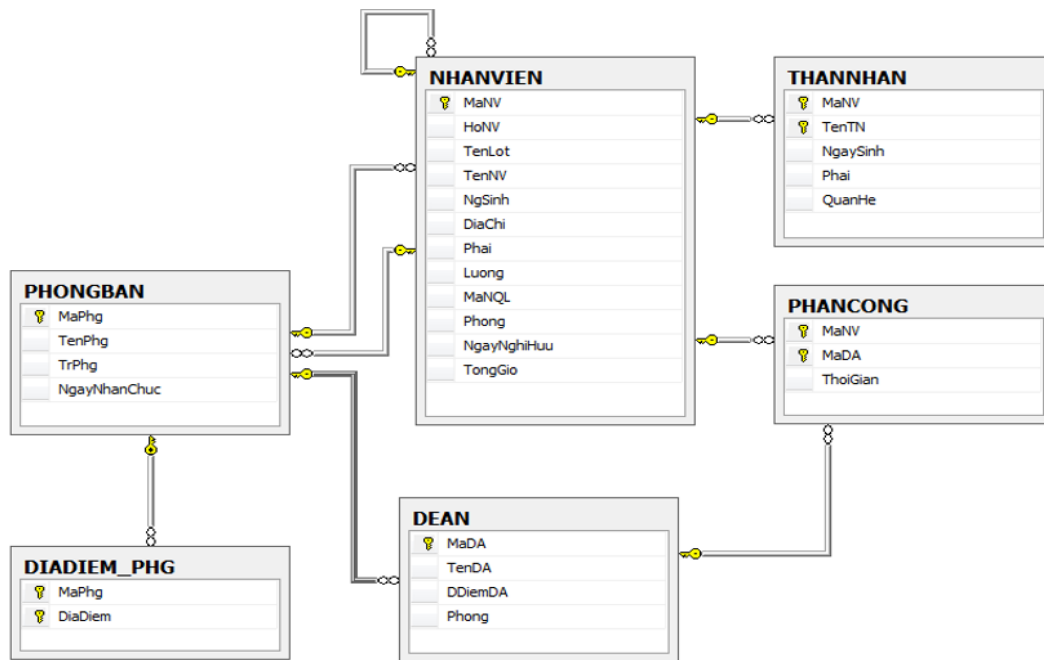
Bước 1: Tại cửa sổ Object Explorer của tiện ích Microsoft SQL Server Management Studio, chọn CSDL chứa các bảng cần tạo mô hình, R-click vào mục **Database Diagrams**, chọn **New Database Diagram**:



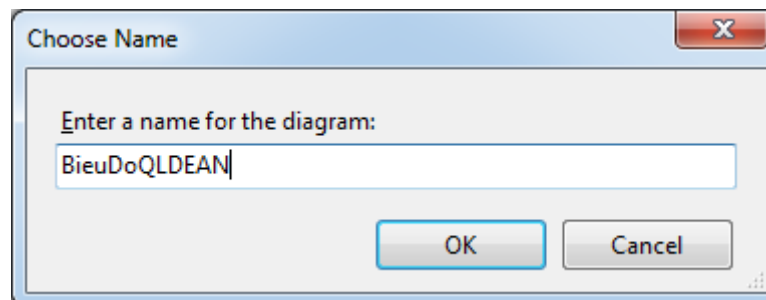
Bước 2: Trong cửa sổ Add Table, chọn các bảng cần đưa vào mô hình quan hệ dữ liệu, sau đó nhấn nút Add.



Bước 3: Sau khi chọn đủ các bảng, nhấn nút Close để đóng cửa sổ Add Table. Nếu các bảng đã được tạo ràng buộc khóa ngoại (foreign key constraint) từ trước, hệ thống sẽ tự động tạo ra các mối liên kết giữa các bảng đã chọn và sắp xếp vào bên trong biểu đồ cơ sở dữ liệu.



Bước 4: Nhấn nút Save trên thanh công cụ, nhập tên của mô hình để lưu lại mô hình quan hệ dữ liệu vừa tạo.



BÀI TẬP KẾT THÚC

LÝ THUYẾT

1. Trình bày chức năng của table (bảng) trong cơ sở dữ liệu.
2. Trình bày khái niệm khóa chính, khóa ngoại. Nếu một bảng không tồn tại khóa chính, trường hợp nào sẽ xảy ra?
3. Trình bày các kiểu dữ liệu trong SQL Server.

THỰC HÀNH

1. Trong CSDL QL_DEAN đã tạo tại bài trước, hãy thiết kế cấu trúc các bảng có khóa chính, khóa ngoại và kiểu dữ liệu như sau:

Bảng **NHANVIEN**: dùng để lưu thông tin của các nhân viên

Tên cột	Kiểu dữ liệu	Ý nghĩa	Nullable
---------	--------------	---------	----------

<u>MaNV</u>	varchar(9)	Mã số của nhân viên, mỗi nhân viên có một mã số duy nhất	No
HoNV	nvarchar(15)	Họ của nhân viên	Yes
TenLot	nvarchar(30)	Tên lót của nhân viên	Yes
TenNV	nvarchar(30)	Tên của nhân viên	Yes
NgSinh	smalldatetime	Ngày sinh của nhân viên	Yes
DChi	nvarchar(150)	Địa chỉ của nhân viên	Yes
Phai	nvarchar(3)	Giới tính của nhân viên	Yes
Luong	numeric(18,0)	Lương của nhân viên	Yes
MaNQL	varchar(9)	Mã nhân viên của người quản lý. Mỗi nhân viên chỉ có một người quản lý. Đây là khóa ngoại tham chiếu đến khóa chính của chính bảng NHANVIEN	Yes
Phong	varchar(2)	Mã phòng nhân viên đang làm việc Đây là khóa ngoại tham chiếu đến khóa chính của chính bảng PHONGBAN	Yes

Bảng **PHONGBAN**: dùng để lưu thông tin của các phòng ban

Tên cột	Kiểu dữ liệu	Ý nghĩa	Nullable
<u>MaPhg</u>	varchar(2)	Mã phòng ban	No
TenPhg	nvarchar(20)	Tên phòng ban	Yes
TrPhg	varchar(9)	Mã nhân viên của trưởng phòng Đây là khóa ngoại tham chiếu đến khóa chính của chính bảng NHANVIEN	Yes
NgayNhan Chuc	smalldatetime	Ngày nhận chức của trưởng phòng	Yes

Bảng **THANNHAN**: dùng để lưu thông tin thân nhân của các nhân viên. Mỗi nhân viên có thể có nhiều thân nhân nhưng cũng có thể không có nhân viên nào

Tên cột	Kiểu dữ liệu	Ý nghĩa	Nullable
<u>MaNV</u>	varchar(9)	Mã nhân viên của nhân viên có thân nhân. Đây là khóa ngoại tham chiếu đến khóa chính của chính bảng NHANVIEN	No
<u>TenTN</u>	varchar(20)	Tên thân nhân	No
NgSinh	smalldatetime	Ngày sinh của thân nhân	Yes

Phai	varchar(3)	Giới tính của thân nhân	Yes
QuanHe	varchar(15)	Quan hệ của thân nhân đối với nhân viên (con trai, con gái, vợ chồng...)	

Bảng **DIADIEM_PHG**: dùng để lưu thông tin địa điểm làm việc của các phòng ban. Mỗi phòng ban có thể có nhiều địa điểm làm việc.

Tên cột	Kiểu dữ liệu	Ý nghĩa	Nullable
<u>MaPhg</u>	varchar(2)	Mã phòng Đây là khóa ngoại tham chiếu đến khóa chính của chính bảng PHONGBAN	No
<u>DiaDiem</u>	varchar(20)	Địa điểm làm việc của phòng ban	No

Bảng **DEAN**: dùng để lưu thông tin các đề án đang được thực hiện.

Tên cột	Kiểu dữ liệu	Ý nghĩa	Nullable
<u>MaDA</u>	varchar(2)	Mã đề án	No
TenDA	nvarchar(50)	Tên đề án	Yes
DDiemDA	varchar(20)	Địa điểm thực hiện đề án	Yes
Phong	varchar(2)	Mã phòng của phòng ban chủ trì đề án Đây là khóa ngoại tham chiếu đến khóa chính của chính bảng PHONGBAN	Yes

Bảng **PHANCONG**: dùng để lưu thông tin phân công nhân viên tham gia đề án.

Tên cột	Kiểu dữ liệu	Ý nghĩa	Nullable
<u>MaNV</u>	varchar(2)	Mã nhân viên tham gia đề án Đây là khóa ngoại tham chiếu đến khóa chính của chính bảng NHANVIEN	No
<u>MaDA</u>	varchar(2)	Mã đề án Đây là khóa ngoại tham chiếu đến khóa chính của chính bảng DEAN	Yes
ThoiGian	numeric(18,0)	Số giờ/tuần mà nhân viên tham gia đề án	Yes

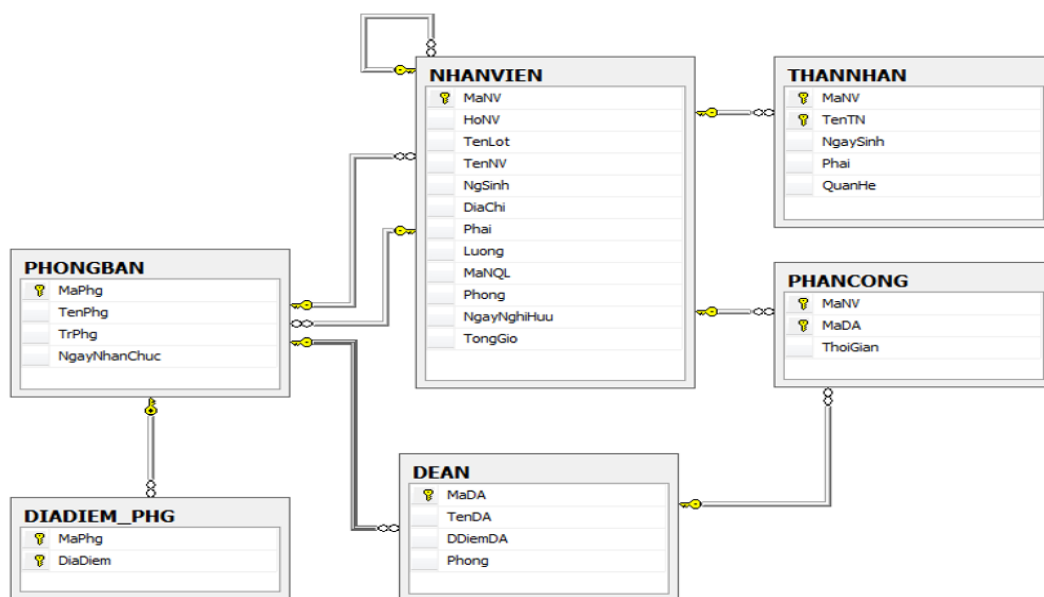
2. Thực hiện các yêu cầu sau:

- Trong bảng PHONGBAN, chỉnh sửa kiểu dữ liệu của cột **TenPhg** thành *nvarchar(30)*
- Trong bảng DIADIEM_PHG, chỉnh sửa kiểu dữ liệu của cột **DiaDiem** thành *nvarchar(20)*
- Trong bảng DEAN, chỉnh sửa kiểu dữ liệu của cột **DDiemDA** thành *nvarchar(20)*
- Trong bảng THANNHAN, chỉnh sửa kiểu dữ liệu của cột **TenTN** thành

nvarchar(20), kiểu dữ liệu của cột **Phai** thành *nvarchar(3)*, kiểu dữ liệu của cột **QuanHe** thành *nvarchar(15)*.

- Tạo ràng buộc cho cột **Phai** của bảng NHANVIEN chỉ mang một trong hai giá trị “**Nam**” hoặc “**Nữ**”
- Thiết lập giá trị mặc định cho cột DDiemDA trong bảng DEAN là “TP Quảng Ngãi”.

3. Trong CSDL QL_DEAN đã tạo tại bài trước, hãy tạo một mô hình quan hệ dữ liệu từ các bảng đã tạo như sau:



CHƯƠNG 3: NGÔN NGỮ THAO TÁC DỮ LIỆU



MỤC TIÊU

Học xong chương này, người học có khả năng:

- Nhập dữ liệu vào bảng, xem và xóa dữ liệu, cập nhật dữ liệu
- Thực hiện rút trích dữ liệu ở mức cơ bản, nâng cao
- Hiểu được chức năng của view trong SQL Server
- Tạo và sử dụng view trong SQL Server.

T-SQL (Transact-SQL) là ngôn ngữ SQL mở rộng dựa trên SQL chuẩn của ISO (International Organization for Standardization) và ANSI (American National Standards Institute) được sử dụng trong SQL Server. Đây là ngôn ngữ được sử dụng trong SQL Server để thao tác với dữ liệu: nhập dữ liệu, rút trích dữ liệu,....

BÀI 1: CÁC LỆNH THAO TÁC VỚI DỮ LIỆU

1. Nhập dữ liệu

1.1. Cú pháp nhập một dòng dữ liệu vào một bảng

Để nhập dữ liệu (mẫu tin) vào một bảng, sử dụng câu lệnh INSERT

❖ Cú pháp nhập dữ liệu tường minh:

```
INSERT INTO table_name (column1, column2, column3, ...)
VALUES (value1, value2, value3, ...);
```

Trong đó:

- **table_name:** tên bảng cần chèn dữ liệu
- **column1, column2, ...:** tên các cột trong bảng sẽ chèn dữ liệu
- **value1, value2, ...:** danh sách các giá trị được chèn vào, được ngăn cách bởi dấu phẩy

Lưu ý: Số lượng và thứ tự các giá trị được chèn vào phải đúng với số lượng và thứ tự các cột được liệt kê.

Ví dụ: Chèn một nam nhân viên có tên Đinh Bá Tiến, sinh ngày 27/7/1982, có địa chỉ tại TP Hồ Chí Minh, làm việc tại phòng 4 vào bảng NHANVIEN

```
INSERT INTO NHANVIEN (MaNV, HoNV, TenLot, TenNV,
NgSinh, DiaChi, GioiTinh, MaPhong)
VALUES ('123', N'Dinh', N'Bá', N'Tiến', '1982/02/27',
N'TP Hồ Chí Minh', N'Nam', '4')
```

❖ **Cú pháp nhập dữ liệu không tường minh:**

```
INSERT INTO table_name
VALUES (value1,value2,value3,...);
```

Lưu ý:

- Có thể không cần chỉ định ra tên các cột, khi đó danh sách các giá trị cần chèn vào phải theo đúng thứ tự các cột bên trong bảng.
- Đối với các cột không có giá trị, SQL Server sẽ tự động chèn giá trị mặc định (được quy định khi tạo bảng) hoặc giá trị NULL.

Nếu không liệt kê tên các cột, thực hiện câu lệnh SQL như sau:

```
INSERT INTO NHANVIEN
VALUES ('123', N'Dinh', N'Bá', N'Tiến', '1982/02/27', N'TP
Hồ Chí Minh', N'Nam', NULL, NULL, '4')
```

1.2. Nhập dữ liệu chuỗi, ngày tháng

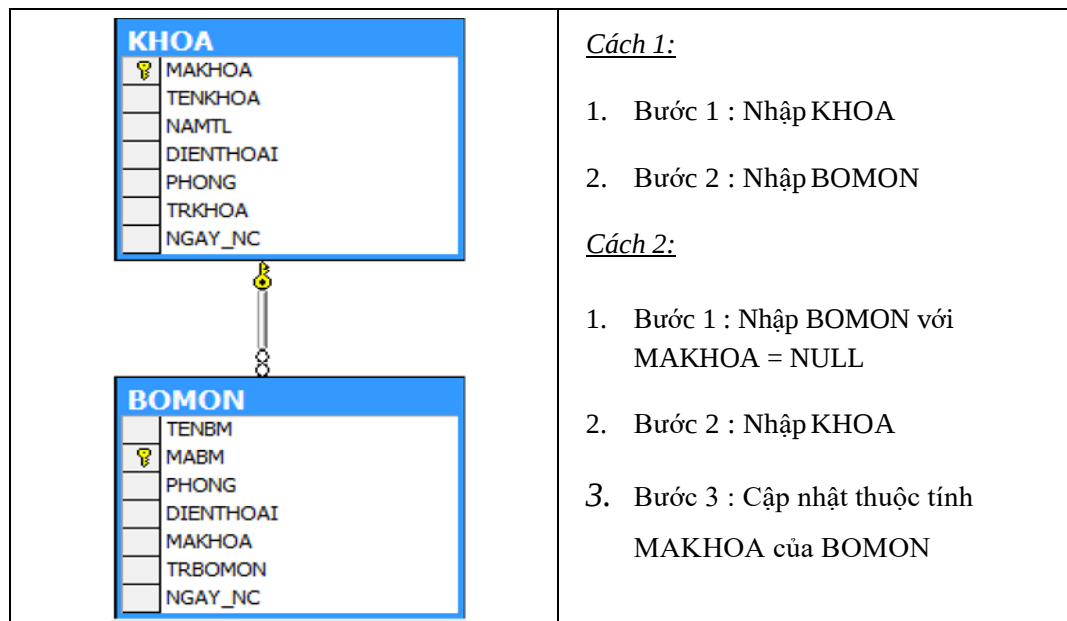
Cú pháp

- **Nhập dữ liệu Unicode :** Thêm kí tự **N** trước chuỗi Unicode
- **Nhập dữ liệu ngày tháng:** Định dạng nhập năm tháng ngày mặc định : 'yyyy/mm/dd' hoặc set dateformat dmy để nhập ngày tháng năm.
- **Nhập một bộ dữ liệu có 1 giá trị NULL (rỗng):** Dùng từ khóa *null*

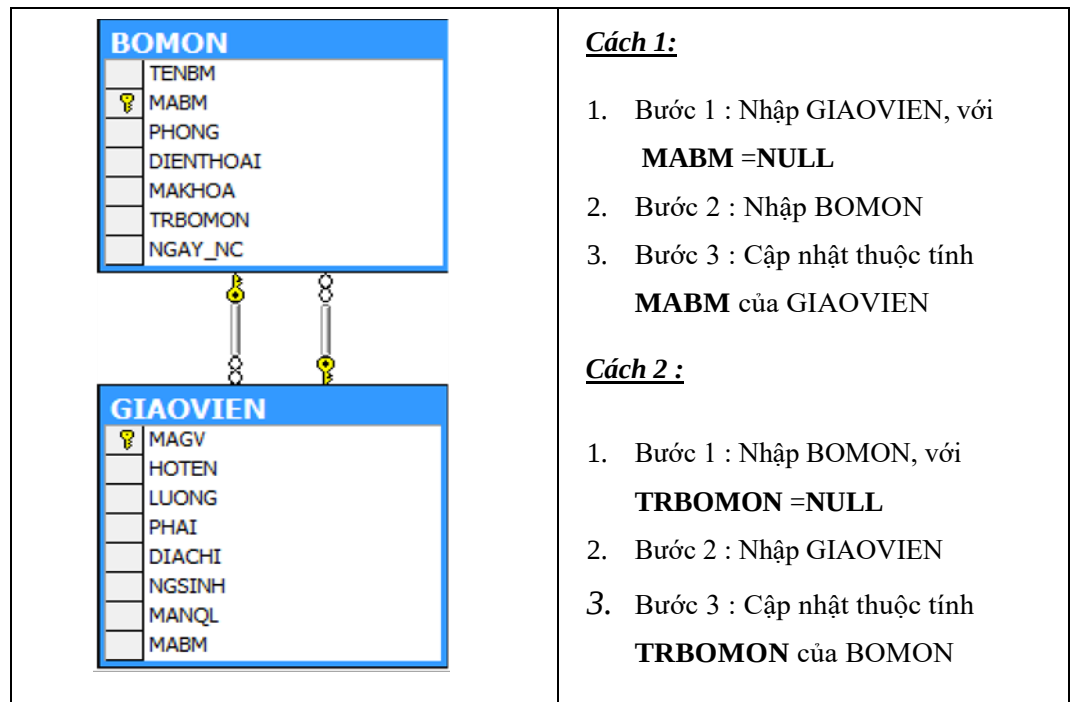
Lưu ý: Nếu thuộc tính được khai báo trong cú pháp tạo bảng là NOT NULL thì bắt buộc phải có giá trị khi nhập 1 bộ vào bảng.

1.3. Nhập dữ liệu khi có ràng buộc khóa ngoại

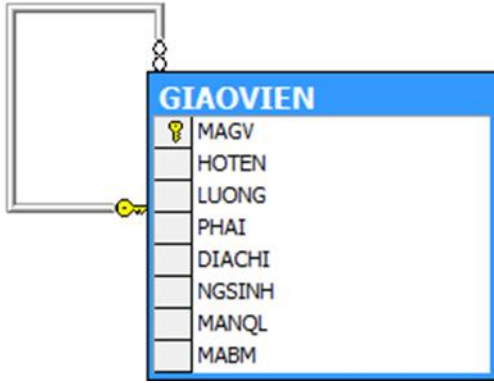
❖ Trường hợp 1:



❖ Trường hợp 2:



❖ Trường hợp 3:

	Cách 1 : Những GIAOVIEN mà có MANQL là null thì nhập trước Sau đó nhập những GIAOVIEN mà đã nhập thông tin về NQL của giáo viên đó. Cách 2 : Bước 1. Nhập tất cả các GIAOVIEN, đặt thuộc tính MANQL là null Bước 2. Cập nhật MANQL của GIAOVIEN
---	--

2. Xem và xóa dữ liệu

❖ Xem nội dung của một bảng:

```
SELECT *
FROM table_name
```

3. Cập nhật dữ liệu

Để cập nhật dữ liệu (mẫu tin) trong bảng, sử dụng câu lệnh UPDATE

Cú pháp:

```
UPDATE table_name
SET column1=value1,column2=value2,...
WHERE some_column=some_value;
```

Trong đó:

- **table_name:** tên bảng cần cập nhật dữ liệu
- **column1, column2,...:** tên các cột trong bảng sẽ được cập nhật giá trị
- **value1, value2, ...:** danh sách các giá trị được cập nhật mới, tương ứng với các cột column1, column2,...
- **some_column=some_value:** điều kiện các mẫu tin được cập nhật

Lưu ý:

- Các mẫu tin được cập nhật là các mẫu tin có giá trị dữ liệu thỏa mãn mệnh đề WHERE
- Trong câu lệnh UPDATE, có thể không cần mệnh đề điều kiện WHERE. Khi đó, tất cả các mẫu tin trong bảng đều sẽ được cập nhật.

Ví dụ: Tăng lương lên 5% cho các nhân viên nữ

```
UPDATE NHANVIEN SET Luong = 1.05 * Luong
WHERE Phai = N'Nữ'
```

BÀI TẬP KẾT THÚC

Thực hiện nhập liệu vào các bảng như sau:

Bảng NHANVIEN:

MaNV	HoNV	TenLot	TenNV	NgSinh	Dchi	Phai	Luong	MaNQL	Phong
123	Đinh	Bá	Tiến	27/2/1982	Mộ Đức	Nam	NULL	234	4
234	Nguyễn	Thanh	Tùng	12/8/1956	Sơn Tịnh	Nam	NULL	901	5
345	Bùi	Thúy	Vũ	NULL	Tư Nghĩa	Nữ	NULL	890	4
456	Lê	Thị	Nhàn	12/7/1962	Mộ Đức	Nữ	NULL	901	4
567	Nguyễn	Mạnh	Hùng	25/3/1985	Sơn Tịnh	Nam	NULL	234	5
678	Trần	Hồng	Quang	NULL	Bình Sơn	Nam	NULL	890	5
789	Trần	Thanh	Tâm	17/6/1972	Tp Quảng Ngãi	Nam	NULL	234	5
890	Cao	Thanh	Huyền	NULL	Tư Nghĩa	Nữ	NULL	901	1
901	Vương	Ngọc	Quyền	12/12/1987	Mộ Đức	Nam	NULL	NULL	1

Bảng PHONGBAN:

MaPhg	TenPhg	TrPhg	NgayNhanChuc
1	Quản lý	890	06/03/2003
4	Điều hành	456	05/11/2014
5	Nghiên cứu	234	05/02/2015

Bảng DEAN:

MaDA	TenDA	DDiemDA	Phong
1	Nâng cao chất lượng muối	Sa Huỳnh	1
10	Xây dựng nhà máy chế biến thủy sản	Dung Quát	4
2	Phát triển hạ tầng mạng	Tp Quảng Ngãi	4
20	Truyền tải cáp quang	Tp Quảng Ngãi	5
3	Tin học hóa trường học	Ba Tơ	1
30	Đào tạo nhân lực	Tỉnh Phong	5

Bảng PHANCONG:

MaNV	MaDA	ThoiGian
123	1	33
123	2	8
345	10	10
345	20	10
345	3	10
456	1	20
456	2	20
678	3	40
789	10	35
789	20	30
789	30	5

Bảng THANNHAN:

MaNV	TenTN	NgSinh	Phai	QuanHe
123	Châu	30/10/2005	Nữ	Con gái
123	Duy	25/10/2001	Nam	Con trai

123	Phuong	30/10/1985	Nữ	Vợ chồng
234	Thanh	05/04/1980	Nữ	Con gái
345	Dương	30/10/1956	Nam	Vợ chồng
345	Khang	25/10/1982	Nam	Con trai
456	Hùng	01/01/1987	Nam	Con trai
901	Thương	05/04/1989	Nữ	Vợ chồng

Bảng DIADIEM_PHG:

MaPhg	DiaDiem
1	Tp Quảng Ngãi
4	Tp Quảng Ngãi
5	Tp Quảng Ngãi
5	Dung Quất
5	Sa Huỳnh

BÀI 2: TRUY VẤN ĐƠN GIẢN

Ngôn ngữ truy vấn SQL có tập lệnh khá phong phú để thao tác trên cơ sở dữ liệu. Trong đó, lệnh quan trọng nhất là lệnh SELECT - lệnh hỏi - tìm kiếm dữ liệu. Kết quả của lệnh SELECT là một quan hệ, quan hệ kết quả này có thể kết xuất ra màn hình, máy in, hoặc các thiết bị lưu trữ thông tin khác.

Câu lệnh SELECT được dùng để truy xuất dữ liệu từ các dòng, các cột của một hay nhiều bảng. Câu lệnh này có thể thực hiện các phép chiếu, chọn, nối. Kết

quả trả về của câu lệnh SELECT có cấu trúc giống một bảng, gọi là bảng kết quả đích.

1. Cú pháp câu truy vấn tổng quát

```
SELECT [tính chất] <danh sách các thuộc tính_1>
FROM <danh sách các table hoặc query/view [as alias] >
[WHERE <điều kiện_1>]
[GROUP BY <danh sách các thuộc tính_2>]
[HAVING <điều kiện_2>]
[ORDER BY <danh sách các thuộc tính_3 [ASC | DESC]>]
```

Trong đó:

- **SELECT:** danh sách trường gồm tên các trường của các bảng trong mệnh đề FROM và các trường mới... Giá trị của trường mới có thể được tạo thành bằng các biểu thức, hàm được xây dựng sẵn.
- **FROM:** danh sách các bảng là tên các bảng. Có thể dùng bí danh cho bảng.
- **WHERE:** điều kiện lọc.
- **GROUP BY:** dùng để nhóm các bản ghi theo một số trường nào đó. Kết quả có thể được lọc bởi mệnh đề HAVING .
- **ORDER BY:** sắp xếp kết quả theo các trường.

Lưu ý:

- **Tính chất** : Một trong các từ khóa: **ALL** (chọn ra tất cả các dòng trong bảng), **DISTINCT** (loại bỏ các dòng trùng lặp thông tin), **TOP <n>** (chọn n dòng đầu tiên thỏa mãn điều kiện).
- **Danh sách các thuộc tính_1**: tên các thuộc tính cho biết thông tin cần lấy.
Chú ý: Các thuộc tính cách nhau bởi dấu phẩy (,)
 - Nếu lấy tất cả các thuộc tính của 1 bảng tbl thì dùng: tbl.*
 - Nếu sau FROM chỉ có 1 table và lấy tất cả các field của table đó thì dùng select *
 - Nếu tồn tại 1 thuộc tính sau select xuất hiện ở 2 table sau FROM thì phải chỉ định rõ thuộc tính đó thuộc table nào.
- **Danh sách các table**: các table chứa thông tin cần lấy. Khi tìm kiếm thông tin trên nhiều hơn 2 table thì phải kết các table lại với nhau (điều kiện kết đặt sau where)
- **Alias**: bí danh (tên tắt) của bảng dùng cho các bảng có tên quá dài hoặc bắt buộc trong trường hợp sử dụng bảng hơn 1 lần khi truy vấn.
- **Điều kiện_1**: là điều kiện để lọc dữ liệu.
- **Danh sách các thuộc tính_2**: dữ liệu sẽ được gom nhóm theo các cột này.
- **Điều kiện_2**: điều kiện lọc lại dữ liệu sau khi đã thực hiện gom nhóm trên dữ liệu. Điều kiện này được áp dụng trên dữ liệu thỏa mãn điều kiện_1.
- **Danh sách các thuộc tính_3**: sắp xếp dữ liệu theo cột nào, thứ tự là tăng (ASC) hoặc giảm (DESC). Mặc định là dữ liệu được sắp theo thứ tự tăng dần. Việc sắp xếp được thực hiện theo thứ tự ưu tiên từ trái qua phải.

2. Câu truy vấn đơn giản

Cú pháp:

```
SELECT <danh sách các thuộc tính>

FROM <tên bảng >
```

Ví dụ: Liệt kê danh sách tất cả các nhân viên

```
SELECT MaNV, HoNV, TenLot, TenNV, NgSinh, DiaChi,
       Phai, Luong, MaNQL, Phong
FROM NHANVIEN
```

❖ Ký hiệu * theo sau từ khóa SELECT dùng để chỉ tất cả các thuộc tính của quan hệ nguồn sẽ là thuộc tính của quan hệ đích. Câu lệnh trên tương đương với câu lệnh sau:

❖ Nếu muốn đặt tên khác cho các cột trong bảng (còn gọi là bí danh – ALIAS), ta thêm từ khóa AS, theo sau là tên mới. Nếu tên chứa các ký tự đặc biệt hoặc khoảng trắng, thì viết tên đó trong cặp dấu ngoặc vuông ([])

Ví dụ: Liệt kê danh sách các đề án, thông tin bao gồm tên đề án và địa điểm

```
SELECT TenDA AS [Tên đề án], DdiemDA AS [Địa điểm]
FROM DEAN
```

```
SELECT *
FROM NHANVIEN
```

❖ Câu lệnh SELECT không chỉ thực hiện việc trích thông tin từ các cột đơn lẻ của bảng mà còn có thể thực hiện các tính toán theo công thức hay biểu thức bất kỳ dựa trên giá trị của các cột trên từng mẫu tin.

Ví dụ: Liệt kê danh sách các nhân viên, thông tin bao gồm họ tên và tuổi

```
SELECT HoNV + ' ' + TenLot + ' ' + TenNV AS HoVaTen,
       YEAR(GETDATE()) - YEAR(NgSinh) AS Tuoi
FROM NHANVIEN
```

❖ Từ khóa **DISTINCT** được đặt theo sau từ khóa SELECT nhằm loại bỏ bớt các mẫu tin trùng nhau trong bảng kết quả của lệnh truy vấn.

```
SELECT DISTINCT MLUONG
FROM NHANVIEN
```

3. Tìm kiếm có sắp xếp

Quan hệ đích có thể được sắp xếp tăng/giảm theo một hoặc nhiều thuộc tính nào đó

bằng cách sử dụng mệnh đề ORDER BY (độ ưu tiên giảm dần từ trái sang phải). Từ khóa DESC được dùng nếu muốn sắp xếp giảm dần. Nếu không có DESC, mặc định quan hệ đích sẽ được sắp xếp tăng dần ASC theo các thuộc tính đã chỉ ra.

Cú pháp:

```
SELECT <danhsach các thuộc tính>

FROM <tên bảng >

ORDER BY < thuộc_ tính_1 [ASC | DESC], thuộc_ tính_2 [ASC | DESC], .>
```

Ví dụ: Liệt kê danh sách họ tên các nhân viên, thứ tự sắp xếp giảm dần theo mức lương, đối với những người có mức lương bằng nhau thì sắp xếp theo tên nhân viên theo thứ tự ABC.

```
SELECT HoNV, TenLot, TenNV, Luong
FROM NHANVIEN
ORDER BY Luong DESC, TenNV ASC
```

4. Tìm kiếm với điều kiện đơn giản

Cú pháp:

```
SELECT <danhsach các thuộc tính>

FROM <tên bảng >

WHERE < (điều kiện 1) AND/OR (điều kiện 2) > (điều kiện n)
```

❖ Các toán tử so sánh :

SQL hỗ trợ các toán tử so sánh sau: > (lớn hơn), < (nhỏ hơn), = (bằng), != (khác), <> (khác), >= (lớn hơn hoặc bằng), <= (nhỏ hơn hoặc bằng)

Ví dụ: Liệt kê danh sách các nhân viên làm việc ở phòng số 5

```
SELECT * FROM NHANVIEN
WHERE Phong = 5
```

❖ Các toán tử logic (NOT, AND, OR) :

SQL hỗ trợ các toán tử logic sau: **AND** (và), **OR** (hoặc) và **NOT** (phủ định)

Ví dụ: Liệt kê danh sách các nhân viên làm việc ở phòng 1 hoặc phòng 5

```
SELECT *
FROM NHANVIEN
WHERE Phong = 1 OR Phong = 2 OR Phong = 5
```

❖ BETWEEN...AND , NOT BETWEEN ... AND

Ví dụ: Liệt kê danh sách các nhân viên có mức lương từ 5 triệu đến 7 triệu

```
SELECT *
FROM NHANVIEN
WHERE luong between 5000000 and 7000000
```

❖ IS NULL và IS NOTNULL

Ví dụ: Liệt kê danh sách các nhân viên không có người quản lý

```
SELECT *
FROM NHANVIEN
WHERE MaNQL is null
```

❖ IN và NOT IN

Ví dụ: Liệt kê danh sách các nhân viên làm việc ở phòng 1, phòng 2 hoặc phòng 5

```
SELECT *
FROM NHANVIEN
WHERE Phong IN ('1', '2', '5')
```

5. Tìm kiếm với điều kiện liên quan đến chuỗi ký tự

Để xử lý với các dữ liệu thuộc dạng xâu ký tự, ngoài toán tử so sánh bằng = để sử dụng so sánh chuỗi tuyệt đối thì ngôn ngữ SQL có hỗ trợ toán tử so sánh chuỗi LIKE để so sánh chuỗi tương đối.

- Ký tự %: đại diện cho một chuỗi ký tự bất kỳ
- Ký tự _: đại diện cho một ký tự bất kỳ
- Ký tự []: đại diện cho các ký tự nằm trong một khoảng nào đó.

Chú ý:

- LIKE 'ab\%cd%' cho ra những chuỗi bắt đầu với 'ab%cd'

- LIKE 'ab\\cd%' cho ra những chuỗi bắt đầu với 'ab\\cd'

Ví dụ: Liệt kê danh sách các nhân viên có tên bắt đầu bằng chữ T

```
SELECT *
FROM NHANVIEN
WHERE TenNV LIKE 'T%'
```

6. Tìm kiếm với điều kiện liên quan đến ngày giờ

Một số hàm liên quan đến ngày giờ mà SQL hỗ trợ:

- **datediff**: Hàm tính khoảng cách (hiệu) 2 thời gian theo một đơn vị nào đó (đơn vị ngày, tháng, năm, giờ, phút, giây)
- **datepart**: Hàm lấy một phần trong giá trị thời gian (ngày, tháng, năm, giờ, phút, giây)
- **year, month, day**: Hàm lấy năm, tháng, ngày của một giá trị thời gian truyền vào.
- **getdate**: Hàm lấy ngày hiện hành của hệ thống

Ví dụ: Cho biết năm thực hiện dự án 'Dao tao'

```
SELECT datediff(year, ngbd_dk, ngkt_dk) as 'nam thuc hien'
FROM DuAn
WHERE MADA = 'DT001'
```

7. Sử dụng các hàm khi tìm kiếm

❖ Các hàm tính toán theo nhóm

- **SUM(thuộc tính)**: tính tổng giá trị các bộ theo thuộc tính đã chỉ ra
- **MAX(thuộc tính)**: cho biết giá trị lớn nhất của các bộ theo thuộc tính đã chỉ ra
- **MIN(thuộc tính)**: cho biết giá trị nhỏ nhất của các bộ theo thuộc tính đã chỉ ra
- **AVG(thuộc tính)**: cho biết giá trị trung bình của các bộ theo thuộc tính đã chỉ ra
- **COUNT(*)**: đếm tất cả các bộ

- **COUNT(thuộc tính):** đếm những bộ mà giá trị của thuộc tính là khác NULL

❖ Các hàm chuyển đổi kiểu dữ liệu

Công dụng của các hàm này là chuyển đổi qua lại giữa các kiểu dữ liệu tương thích với nhau bên trong SQL Server. Thông thường, người ta thường chuyển đổi các kiểu dữ liệu số hoặc kiểu dữ liệu ngày giờ về kiểu dữ liệu chuỗi để hiển thị ra màn hình.

- **Hàm CAST:** Dùng để chuyển đổi một biểu thức, một biến sang kiểu dữ liệu bất kỳ mong muốn.

Cú pháp:

```
CAST (Biểu_thức AS Kiểu_dữ_liệu)
```

Trong đó:

- **Biểu_thức:** tên một cột trong bảng, một biến hoặc một biểu thức tính toán cần chuyển sang kiểu dữ liệu mới.
- **Kiểu_dữ_liệu:** tên kiểu dữ liệu mới mà biểu thức sẽ được chuyển đổi sang.
- **Hàm CONVERT:** Dùng để chuyển đổi một biểu thức, một biến sang kiểu dữ liệu bất kỳ mong muốn nhưng có thể theo một định dạng nào đó.

Cú pháp:

```
CONVERT (Kiểu_dữ_liệu, Biểu_thức [,Định_dạng])
```

Trong đó:

- **Kiểu_dữ_liệu:** tên kiểu dữ liệu mà biểu thức sẽ được chuyển đổi sang
- **Biểu_thức:** tên một cột trong bảng, một biến hoặc một biểu thức tính toán cần chuyển sang kiểu dữ liệu mới.
- **Định_dạng:** con số chỉ định việc định dạng cho việc chuyển đổi dữ liệu từ dạng ngày sang dạng chuỗi
- **Hàm STR:** Dùng để chuyển đổi một biểu thức, một biến có kiểu dữ liệu số sang kiểu dữ liệu chuỗi. Phải đảm bảo đủ vùng trống để chứa các ký số khi chuyển đổi sang kiểu dữ liệu chuỗi.

Cú pháp:

```
STR(Số_thực, Số_ký_tự [,Số_lẻ])
```

Trong đó:

- **Số_thực**: một biến hoặc một biểu thức tính toán có kiểu dữ liệu số thực
- **Số_ký_tự**: số vùng trắng dành để chứa các ký số sau khi chuyển đổi sang kiểu dữ liệu chuỗi.
- **Số_lẻ**: chỉ định số thập phân

❖ Các hàm toán học

Các hàm này thường có tham số vào là kiểu dữ liệu số và giá trị trả về của chúng cũng là kiểu dữ liệu số.

- **Hàm ABS**: Giá trị trả về là trị tuyệt đối của một số bất kỳ

Cú pháp:

```
ABS (Biểu_thức_số)
```

- **Hàm POWER**: Giá trị trả về là phép tính lũy thừa của một số bất kỳ nào đó theo một số mũ chỉ định

Cú pháp:

```
POWER (Biểu_thức_số, Số_mũ)
```

- **Hàm RAND**: Giá trị trả về là một số thực ngẫu nhiên mà SQL Server tự động tạo ra đảm bảo không trùng lặp

Cú pháp:

```
RAND ([Số_nguồn])
```

Trong đó:

- **Số_nguồn**: là một giá trị số nguyên có phạm vi không vượt quá phạm vi của kiểu dữ liệu int làm giá trị nguồn cho hệ thống tạo ra số ngẫu nhiên.
- **Hàm ROUND**: Giá trị trả về là một số đã được làm tròn

Cú pháp:

```
ROUND (Biểu_thức_số, Vị_trí_làm_tròn)
```

Trong đó:

- **Biểu_thức_số**: là một biểu thức có kiểu dữ liệu là số thực.
- **Vị_trí_làm_tròn**: là một số nguyên âm hoặc dương dùng để chỉ định vị trí muốn làm tròn, được tính từ vị trí dấu chấm thập phân

- **Hàm SQRT:** Giá trị trả về là căn bậc hai của một số dương bất kỳ

Cú pháp:

```
SQRT(Biểu_thức_số)
```

❖ Các hàm xử lý chuỗi

- **Hàm UPPER, LOWER:** Giá trị trả về là một chuỗi sau khi đã chuyển đổi các ký tự bên trong chuỗi thành chữ in (upper) hoặc chữ thường (lower)

Cú pháp:

```
UPPER(Biểu_thức_chuỗi)
```

```
LOWER(Biểu_thức_chuỗi)
```

- **Hàm LEFT, RIGHT, SUBSTRING:** Giá trị trả về là một chuỗi con được trích ra từ chuỗi nguồn. Chuỗi con được trích ra tại vị trí bắt đầu từ bên trái (left), bên phải (right) hoặc tại bất kỳ vị trí nào (substring) và lấy ra bao nhiêu ký tự.

Cú pháp:

```
LEFT(Chuỗi_nguồn, Số_ký_tự)
```

```
RIGHT(Chuỗi_nguồn, Số_ký_tự)
```

```
SUBSTRING(Chuỗi_nguồn, Số_ký_tự)
```

8. Phép kết

Một câu truy vấn có thể lấy thông tin từ nhiều bảng. Các bảng này được liệt kê trong mệnh đề FROM, thứ tự các bảng không quan trọng. Trong mệnh đề WHERE cần có điều kiện để kết nối các bảng với nhau.

Nếu thuộc tính ở mệnh đề SELECT hoặc WHERE là thuộc tính của nhiều hơn một bảng thì cần phải chỉ rõ thuộc tính đó thuộc về bảng nào theo cú pháp: *tên bảng.tên thuộc tính*.

Ví dụ: Liệt kê danh sách họ tên các nhân viên cùng với tên phòng ban mà họ đang làm việc.

```
SELECT NHANVIEN.HoNV, NHANVIEN.TenLot, NHANVIEN.TenNV,
PHONG.TenPhg
FROM NHANVIEN , PHONGBAN
WHERE NHANVIEN.Phong = PHONG.MaPhg
```

9. Sử dụng ALIAS

Khi tìm kiếm trên nhiều bảng để làm câu truy vấn dễ hiểu và ngắn gọn hoặc giữa các bảng có thuộc tính trùng tên, hoặc sử dụng một bảng nhiều hơn 1 lần trong câu truy vấn đặt tên lại các bảng để có thể phân biệt các bảng, các thuộc tính.

Ví dụ: Liệt kê danh sách họ tên các nhân viên cùng với tên phòng ban mà họ đang làm việc. (sử dụng **ALIAS**)

```
SELECT N.HoNV, N.TenLot, N.TenNV, P.TenPhg
FROM NHANVIEN N, PHONGBAN P
WHERE N.Phong = P.MaPhg
```

Lưu ý: Khi truy vấn trên các bảng có các thuộc tính cùng tên thì việc sử dụng ALIAS sẽ giúp tránh được việc nhập nhầm khi sử dụng các thuộc tính. Ngoài ra, việc sử dụng ALIAS trong những câu truy vấn trên nhiều bảng sẽ làm cho câu truy vấn dễ đọc hơn.

10. Sử dụng JOIN

Việc liên kết hai hay nhiều bảng trong câu lệnh SELECT bằng cách trên (phép kết) có một nhược điểm là tốn nhiều thời gian nếu lượng dữ liệu lớn. Do đó, còn có một cách để liên kết các bảng là sử dụng từ khóa INNER JOIN.

Cú pháp chung:

```
SELECT column_name(s)
FROM table1
INNER JOIN table2
ON table1.column_name=table2.column_name;
```

Khi sử dụng từ khóa INNER JOIN như trên, hệ thống sẽ trả về tất cả các mẫu tin từ hai bảng table1 và table2 có giá trị giống nhau ở cột column_name.

Ví dụ: Liệt kê danh sách họ tên các nhân viên cùng với tên phòng ban mà họ đang

làm việc.

```
SELECT N.HoNV, N.TenLot, N.TenNV, P.TenPhg
FROM NHANVIEN N INNER JOIN PHONGBAN P
ON N.Phong = P.MaPhg
```

Ngoài việc sử dụng từ khóa INNER JOIN, còn có thể sử dụng các từ khóa LEFT JOIN, RIGHT JOIN hoặc FULL JOIN để kết nối các bảng. Kết quả trả về sẽ có những khác biệt.

BÀI TẬP KẾT THÚC

LÝ THUYẾT

1. Các câu lệnh T-SQL được chia làm bao nhiêu loại? Trình bày tên các loại và cho ví dụ.
2. Trình bày các mệnh đề trong câu lệnh SELECT.

THỰC HÀNH

Sử dụng CSDL quản lý đề án đã tạo ở phần trước, thực hiện các yêu cầu sau bằng ngôn ngữ SQL:

1. Cập nhật ngày sinh cho những nhân viên có ngày sinh là NULL là ngày 01/01/1985.
2. Cập nhật Lương cho các nhân viên phòng Quản lý là 6.200.000 đồng, cho các nhân viên phòng Điều hành là 5.800.000 đồng, cho các nhân viên phòng Nghiên cứu là 9.700.000 đồng.
3. Tăng lương lên 5% cho các nhân viên nữ, tăng 500.000 đồng tiền lương cho các nhân viên trên 40 tuổi.
4. Tăng thời gian làm việc đối với các nhân viên tham gia đề án có mã số 10 lên thêm 5 giờ.
5. Thêm vào bảng NHANVIEN cột NgayBatDau (Ngày bắt đầu làm việc) với kiểu dữ liệu là smalldatetime. Sau đó, cập nhật ngày bắt đầu làm việc cho toàn bộ các nhân viên là ngày 01/01/2010.
6. Hãy tạo ràng buộc cho cột Phai của bảng NHANVIEN chỉ có “Nam” hoặc “Nữ”
7. Hãy tạo ràng buộc cho dữ liệu của cột NgaySinh <= Ngayhientai -18(năm) trong bảng NHANVIEN.
8. Tạo 1 default có giá trị là ‘TP Quảng Ngãi’ , sau đó gắn vào cột DDiemDA của bảng DIADIEM.

9. Liệt kê danh sách tất cả các nhân viên
10. Tìm các nhân viên làm việc ở phòng số 5
11. Tìm các nhân viên có mức lương trên 6.000.000 đồng
12. Tìm các nhân viên có mức lương trên 6.500.000 ở phòng 1 hoặc các nhân viên có mức lương trên 5.000.000 ở phòng 4
13. Cho biết họ tên đầy đủ của các nhân viên ở TP Quảng Ngãi
14. Cho biết họ tên đầy đủ của các nhân viên có họ bắt đầu bằng ký tự 'N'
15. Cho biết ngày sinh và địa chỉ của nhân viên Cao Thanh Huyền
16. Cho biết các nhân viên có năm sinh trong khoảng 1955 đến 1975
17. Cho biết các nhân viên và năm sinh của nhân viên
18. Cho biết các nhân viên và tuổi của nhân viên
19. Với mỗi phòng ban, cho biết tên phòng ban và địa điểm phòng
20. Tìm tên những người trưởng phòng của từng phòng ban
21. Tìm tên và địa chỉ của tất cả các nhân viên của phòng "Điều hành".
22. Với mỗi đề án ở Tp Quảng Ngãi, cho biết tên đề án, tên phòng ban, họ tên và ngày nhận chức của trưởng phòng của phòng ban chủ trì đề án đó.
23. Tìm tên những nữ nhân viên và tên người thân của họ
24. Với mỗi nhân viên, cho biết họ tên nhân viên và họ tên người quản lý trực tiếp của nhân viên đó
25. Với mỗi nhân viên, cho biết họ tên của nhân viên đó, họ tên người trưởng phòng và họ tên người quản lý trực tiếp của nhân viên đó.
26. Tên những nhân viên phòng số 5 có tham gia vào đề án "Xây dựng nhà máy chế biến thủy sản" và tên người quản lý trực tiếp.
27. Cho biết tên các đề án mà nhân viên Trần Thanh Tâm đã tham gia.
- .

BÀI 3: TRUY VẤN NÂNG CAO

1. Sử dụng các hàm kết hợp khi truy vấn

Hàm kết hợp: hàm kết hợp là hàm được sử dụng khi truy vấn, và kết quả trả về của hàm chỉ có được khi kết hợp nhiều giá trị lại với nhau.

Các hàm kết hợp: SQL hỗ trợ các hàm kết hợp sau:

- COUNT : đếm số dòng dữ liệu hoặc đếm số lượng giá trị của một thuộc tính.
- AVG: tính giá trị trung bình
- MAX: tính giá trị lớn nhất
- MIN: tính giá trị nhỏ nhất
- SUM: tính tổng

Lưu ý: Ngoài hàm COUNT(*) sử dụng để lấy số dòng dữ liệu, các hàm kết hợp còn lại được tính toán trên một thuộc tính của tất cả các dòng dữ liệu trong kết quả trả về.

Ví dụ: Cho biết số lượng nhân viên của toàn công ty.

```
SELECT COUNT(*)
FROM NHANVIEN
```

Giải thích: Hàm COUNT sẽ thực hiện đếm số lượng dòng dữ liệu của bảng NHANVIEN, và số lượng dòng này chính là số lượng nhân viên của toàn công ty.

2. Truy vấn gom nhóm

Khi cần tính toán trên các mẫu tin theo một nhóm nào đó, ta dùng mệnh đề GROUP BY. Mệnh đề GROUP BY dùng để phân nhóm dữ liệu. Những mẫu tin của bảng có cùng giá trị trên các thuộc tính này sẽ tạo thành một nhóm.

GROUP BY: được sử dụng khi có nhu cầu gom nhóm dữ liệu để thực hiện các thao tác tính toán. Do đó GROUP BY thường được sử dụng kèm với các hàm kết hợp.

Khi sử dụng GROUP BY với các hàm kết hợp, các hàm kết hợp chỉ tính toán trên các dòng cùng một nhóm dữ liệu.

Ví dụ: Với mỗi đề án, cho biết có bao nhiêu nhân viên tham gia đề án đó.

```
SELECT D.TenDA, COUNT(P.MaNV) AS TongNhanVien
FROM PHANCONG P INNER JOIN DEAN D ON P.MaDA = D.MaDA
GROUP BY P.MaDA, D.TenDA
```

3. Truy vấn theo điều kiện gom nhóm

Khi cần lọc các mẫu tin thoả mãn điều kiện sau khi gom nhóm, ta sử dụng mệnh đề HAVING.

Ví dụ: Liệt kê các đề án có tổng số nhân viên tham gia từ 3 người trở lên

```
SELECT D.TenDA, COUNT(P.MaNV) AS TongNhanVien
FROM PHANCONG P INNER JOIN DEAN D ON P.MaDA = D.MaDA
GROUP BY P.MaDA, D.TenDA
HAVING COUNT(P.MaNV) >= 3
```

4. Truy vấn con

Là những câu lệnh mà trong thành phần WHERE có chứa thêm câu lệnh SELECT khác nữa. Câu lệnh này thường gặp khi dữ liệu cần thiết phải duyệt qua nhiều lần.

4.1. Cú pháp

```
SELECT A
FROM X
WHERE ... ( SELECT B FROM Y WHERE ... ) ...
```

Một số quy tắc khi sử dụng truy vấn con:

- Câu truy vấn con phải nằm trong cặp ngoặc đơn.
- Đặt truy vấn con bên phải điều kiện so sánh.
- Mệnh đề ORDER BY trong truy vấn con là không cần thiết ngoại trừ khi có sử dụng mệnh đề TOP.
- Sử dụng các toán tử một dòng với các truy vấn con trả về một dòng và sử dụng các toán tử nhiều dòng với các truy vấn con trả về nhiều dòng.

Ví dụ: Cho biết danh sách các nhân viên có tối thiểu một thân nhân

```
SELECT * FROM NHANVIEN
WHERE MaNV IN (SELECT MaNV FROM THANNHAN)
```

4.2. Truy vấn con trả về một giá trị

- ❖ Trả về duy nhất 1 dòng
- ❖ Sử dụng các toán tử so sánh một dòng

Toán tử	Ý nghĩa
=	Bằng
>	Lớn hơn
>=	Lớn hơn hoặc bằng
<	Nhỏ hơn
<=	Nhỏ hơn hoặc bằng
<>	Không bằng

4.3. Truy vấn con trả về nhiều giá trị

- ❖ Trả về nhiều dòng
- ❖ Sử dụng các toán tử so sánh nhiều dòng

Toán tử	Ý nghĩa
IN	Bằng một trong các giá trị
ANY	Chỉ cần thỏa một trong các giá trị trả về bởi truy vấn con
ALL	Phải thỏa tất cả các giá trị trả về bởi truy vấn con

Ví dụ: Liệt kê danh sách nhân viên có mức lương nhỏ nhất

```
SELECT * FROM NHANVIEN
WHERE MLUONG <= ALL (SELECT MIN (MLUONG) FROM NHANVIEN)
```

5. Hàm Case

Cú pháp

```

CASE <biểu thức>
    WHEN <gtri_1> THEN <kquả_1>
    [WHEN <gtri_2> THEN <kquả_2>
    ...
    ]
    [ELSE <kquả_n>]
END
    
```

Ví dụ: Cho biết họ tên các nhân viên và năm về hưu của họ

```

SELECT HONV, TENNV,
    (CASE PHAI
        WHEN 'Nam' THEN YEAR(NGSINH) + 60
        WHEN 'Nu' THEN YEAR(NGSINH) + 55
        END ) AS NAMVEHUU
FROM NHANVIEN
    
```

BÀI TẬP KẾT THÚC

1. Cho biết số lượng đề án của công ty
2. Liệt kê danh sách các phòng ban có tham gia chủ trì các đề án
3. Cho biết số lượng các phòng ban có tham gia chủ trì các đề án
4. Cho biết số lượng đề án do phòng Nghiên Cứu chủ trì
5. Cho biết lương trung bình của các nữ nhân viên
6. Cho biết số thân nhân của nhân viên Đinh Bá Tiến
7. Liệt kê danh sách 3 nhân viên lớn tuổi nhất, danh sách bao gồm họ tên và năm sinh.
8. Với mỗi đề án, liệt kê tên đề án và tổng số giờ làm việc một tuần của tất cả các nhân viên tham gia đề án đó.
9. Với mỗi đề án, cho biết có bao nhiêu nhân viên tham gia đề án đó
10. Cho biết mã đề án có đông nhân viên tham gia nhất.
11. Với mỗi nhân viên, cho biết họ và tên nhân viên và số lượng thân nhân của nhân viên đó.
12. Với mỗi nhân viên, cho biết họ tên của nhân viên và số lượng đề án mà nhân viên đó đã tham gia.
13. Với mỗi nhân viên, cho biết số lượng nhân viên mà nhân viên đó quản lý trực tiếp.
14. Với mỗi phòng ban, liệt kê tên phòng ban và lương trung bình của những nhân viên làm việc cho phòng ban đó.

15. Với các phòng ban có mức lương trung bình trên 5.200.000, liệt kê tên phòng ban và số lượng nhân viên của phòng ban đó.
16. Với mỗi phòng ban, cho biết tên phòng ban và số lượng đề án mà phòng ban đó chủ trì
17. Với mỗi phòng ban, cho biết tên phòng ban, họ tên người trưởng phòng và số lượng đề án mà phòng ban đó chủ trì
18. Với mỗi phòng ban có mức lương trung bình lớn hơn 6.000.000, cho biết tên phòng ban và số lượng đề án mà phòng ban đó chủ trì.
19. Cho biết số đề án diễn ra tại từng địa điểm
20. Với mỗi đề án, cho biết tên đề án và số lượng công việc của đề án này.
21. Cho biết danh sách các nhân viên có tối thiểu một thân nhân (dùng INNER JOIN, IN, EXISTS).
22. Cho biết danh sách các nhân viên không có thân nhân nào (dùng LEFT JOIN, NOT IN, NOT EXISTS).
23. Cho biết họ tên các nhân viên có trên 2 thân nhân.
24. Cho biết họ tên những trưởng phòng có tối thiểu một thân nhân.
25. Cho biết danh sách những trưởng phòng chưa có gia đình.
26. Cho biết họ tên các nhân viên phòng Quản lý có mức lương trên mức lương trung bình của phòng Quản lý.
27. Cho biết họ tên nhân viên có mức lương trên mức lương trung bình của phòng mà nhân viên đó đang làm việc
28. Cho biết tên phòng ban và họ tên trưởng phòng của phòng ban có đông nhân viên nhất.
29. Cho biết danh sách các đề án mà nhân viên có mã là 456 chưa tham gia.
30. Danh sách nhân viên gồm mã nhân viên, họ tên và địa chỉ của những nhân viên không sống tại TP Quảng Ngãi nhưng làm việc cho một đề án ở TP Quảng Ngãi.
31. Tìm họ tên và địa chỉ của các nhân viên làm việc cho một đề án ở một địa điểm nhưng lại không sống tại địa điểm đó.
32. Cho biết danh sách các mã đề án có: nhân công với họ là Lê hoặc có người trưởng phòng chủ trì đề án với họ là Lê.
33. Liệt kê danh sách các đề án mà cả hai nhân viên có mã số 123 và 234 cùng làm.
34. Liệt kê danh sách các đề án mà cả hai nhân viên Đinh Bá Tiến và Lê Thị Nhân cùng làm.
35. Danh sách những nhân viên (bao gồm mã nhân viên, họ tên, phái) làm việc trong mọi đề án của công ty

36. Danh sách những nhân viên (bao gồm mã nhân viên, họ tên) được phân công tất cả đề án do phòng số 5 chủ trì.
37. Tìm những nhân viên (bao gồm mã nhân viên, họ tên) được phân công tất cả đề án mà nhân viên Lê Thị Nhàn làm việc
38. Cho biết danh sách nhân viên tham gia vào tất cả các đề án ở TP Quảng Ngãi
39. Cho biết phòng ban chủ trì tất cả các đề án ở TP Quảng Ngãi
40. Cho biết những phòng ban có nhân viên tham gia cả 2 dự án ở Dung Quất và Sa Huỳnh.
41. Cho biết những phòng ban có nhân viên tham gia dự án ở Dung Quất hoặc Sa Huỳnh.
42. Hãy xóa các nhân viên chưa tham gia đề án nào.
43. Hãy xóa các nhân viên không có thân nhân

BÀI 4: BẢNG ẢO (VIEW)

1. Khái niệm về view

View (bảng ảo, khung nhìn) là một đối tượng thuộc CSDL, có cấu trúc giống một table nhưng không có chức năng lưu trữ dữ liệu. Thực chất, view chỉ lưu trữ một câu lệnh SELECT dùng để truy vấn dữ liệu từ các table. Các table được sử dụng để lấy dữ liệu cho view được gọi là table cơ sở (underlying table) của view đó.

Người sử dụng vẫn có thể cập nhật dữ liệu (thêm, xóa, sửa) trên view. Đây là một hình thức cập nhật gián tiếp dữ liệu ở các table cơ sở.

2. Lợi ích của việc sử dụng view

Đảm bảo tính an toàn, bảo mật của dữ liệu và không ảnh hưởng đến dữ liệu gốc: view được sử dụng để khai thác dữ liệu như table, chia sẻ cho nhiều người dùng. View không hiện ra tất cả thông tin dữ liệu mà chỉ hiển thị ra các thông tin tối thiểu mà người sử dụng cần.

Thông qua việc truy vấn trực tiếp đến view, người sử dụng có thể dễ dàng truy xuất đến thông tin mà họ cần mà không cần quan tâm đến thông tin này đang được lưu trữ ở table nào.

Trong thực tế, view thường được tạo ra để biểu diễn các thông tin được truy xuất từ nhiều bảng với điều kiện phức tạp.

3. Tạo view

```
CREATE VIEW Tên_View [(Danh sách các trường)]  
[WITH ENCRYPTION]  
AS  
    Câu lệnh SELECT  
[WITH CHECK OPTION]
```

Cú pháp:

Trong đó:

- **Tên_View**: tên view, được đặt tuân theo quy tắc đặt tên trong SQL Server, nên sử dụng tiếp đầu ngữ vw_
- **Danh sách các trường**: danh sách các trường được chỉ định cho view,

tên các trường được ngăn cách bởi dấu phẩy.

- Từ khóa **WITH ENCRYPTION**: dùng để mã hóa câu lệnh SELECT bên trong view
- Từ khóa **WITH CHECK OPTION**: dùng để ngăn cản các thao tác cập nhật dữ liệu đối với các view có sử dụng mệnh đề WHERE trong câu lệnh SELECT.

Một số lưu ý trong việc tạo view bằng câu lệnh T-SQL:

- Các trường được chỉ định cho view phải có cùng số lượng với các trường trong kết quả trả về của câu lệnh SELECT.
- Nếu không chỉ định danh sách các trường cho view: tên các trường trong view sẽ chính là tiêu đề của các trường trong kết quả câu lệnh SELECT. Do đó, nếu kết quả của câu lệnh SELECT có ít nhất một trường được sinh ra bởi một biểu thức hoặc tồn tại hai trường trùng tên thì cần phải đặt tên cho trường trong câu lệnh
- SELECT (sử dụng từ khóa AS).
- Câu lệnh SELECT không chứa các từ khóa ORDER BY (sắp xếp dữ liệu), COMPUTE (thống kê dữ liệu cuối cùng), COMPUTE BY (thống kê dữ liệu theo nhóm), SELECT INTO (sao chép dữ liệu sang table mới).

Ví dụ: Tạo view có tên vw_NhanVienPhong5 chứa danh sách nhân viên phòng 5, thông tin bao gồm mã nhân viên, họ tên và tuổi.

Cách 1:

```
CREATE VIEW vw_NhanVienPhong5 (MaNV, HoVaTen, Tuoai) AS
SELECT MaNV, HoNV + ' ' + TenLot + ' ' + TenNV,
YEAR(GETDATE()) - YEAR(NgSinh)
FROM NHANVIEN
WHERE Phong = 5
```

Cách 2:

❖ **Từ khóa WITH ENCRYPTION**

Khi muốn xem nội dung của câu lệnh SELECT bên trong view, có thể sử dụng thủ tục

của hệ thống là `sp_helptext`.

Ví dụ: Để xem nội dung câu lệnh SELECT bên trong view `w_NhanVienPhong5`, ta

```
CREATE VIEW vw_NhanVienPhong5
AS
SELECT MaNV, HoNV + ' ' + TenLot + ' ' + TenNV AS
HoVaTen, YEAR(GETDATE()) - YEAR(NgSinh) AS Tuoi
FROM NHANVIEN
WHERE Phong = 5
```

thực thi thủ tục `sp_helptext` như sau:

```
EXEC sp_helptext vw_NhanVienPhong5
```

Để giấu nội dung câu lệnh SELECT định nghĩa view, người ta sử dụng từ khóa `WITH ENCRYPTION`.

Ví dụ: Nếu `vw_NhanVienPhong5` được định nghĩa như sau:

```
CREATE VIEW vw_NhanVienPhong5
WITH ENCRYPTION
AS
SELECT MaNV, HoNV + ' ' + TenLot + ' ' + TenNV AS
HoVaTen, YEAR(GETDATE()) - YEAR(NgSinh) AS Tuoi
FROM NHANVIEN
WHERE Phong = 5
```

4. Xem và cập nhật view

Người sử dụng có thể thực hiện các thao tác xem, thêm, sửa, xóa dữ liệu trên view. Thực chất, những thao tác này sẽ chuyển thành các thao tác trên table cơ sở và tác động đến dữ liệu trên table cơ sở.

Để xem dữ liệu trên view, có thể sử dụng câu lệnh `SELECT` giống hoàn toàn như xem dữ liệu trên table.

Ví dụ: Để xem dữ liệu trong view `vw_NhanVienPhong5`, thực hiện câu lệnh

sau:

```
SELECT * FROM vw_NhanVienPhong5
```

Việc cập nhật dữ liệu trên view có thể được thực hiện bằng các câu lệnh

INSERT, UPDATE, DELETE thông qua việc tham chiếu đến tên view. Mặc dù dữ liệu trong view có thể được lấy ra từ nhiều table nhưng việc cập nhật dữ liệu trên view chỉ được phép tác động trên một và chỉ một bảng mà thôi.

Lưu ý: Chỉ có thể thực hiện thao tác cập nhật dữ liệu (thêm, xóa, sửa) trên view khi view đó phải thỏa mãn các điều kiện sau:

- Câu lệnh SELECT định nghĩa view không chứa các từ khóa DISTINCT, TOP, GROUP BY và UNION
- Các thành phần xuất hiện trong danh sách chọn của câu lệnh SELECT phải là các cột trong table cơ sở. Trong danh sách chọn không chứa các biểu thức tính toán và các hàm thống kê (AVG, MAX, MIN, COUNT, SUM...)

Ngoài những điều kiện trên, các thao tác thay đổi dữ liệu thông qua view còn phải đảm bảo các ràng buộc trên table cơ sở, tức là vẫn đảm bảo tính toàn vẹn dữ liệu.

Thông thường, việc thực hiện các thao tác thay đổi dữ liệu thông qua view chỉ được thực hiện với các view đơn giản. Đối với các view phức tạp thì không thực hiện được, hay nói khác hơn, dữ liệu trong view chỉ để đọc.

5. Thay đổi và hủy bỏ view

❖ Thay đổi định nghĩa View

Người sử dụng có thể thay đổi định nghĩa View bằng cách sử dụng câu lệnh ALTER VIEW. Cú pháp câu lệnh ALTER VIEW như sau:

```
ALTER VIEW Tên_View[(Danh sách các trường)]  
[WITH ENCRYPTION]  
AS  
    Câu lệnh SELECT mới  
[WITH CHECK OPTION]
```

Ví dụ: Trong vw_NhanVienPhong5, muốn lấy thêm thông tin về địa chỉ của các nhân viên, ta có thể định nghĩa lại như sau:

```
ALTER VIEW vw_NhanVienPhong5
SELECT MaNV, HoNV + ' ' + TenLot + ' ' + TenNV AS
HoVaTen, YEAR(GETDATE()) - YEAR(NgSinh) AS Tuoi,
DiaChi
FROM NHANVIEN
WHERE Phong = 5
```

❖ Hủy bỏ View

Sau khi không còn được sử dụng, có thể thực hiện xóa view bằng câu lệnh DROP như sau:

```
DROP VIEW Tên_View
```

Ví dụ: Để xóa vw_NhanVienPhong5, thực hiện câu lệnh:

```
DROP VIEW vw_NhanVienPhong5
```

BÀI TẬP KẾT THÚC

LÝ THUYẾT

1. Trình bày chức năng của view trong cơ sở dữ liệu.
2. Trình bày chức năng của các từ khóa WITH ENCRYPTION và WITH CHECK OPTION.

THỰC HÀNH

1. Tạo view có tên **vwNhanVienNam** dùng để liệt kê thông tin mã nhân viên, họ, tên lót, tên, ngày tháng năm sinh, mã phòng của các nhân viên nam. Sau đó sử dụng câu lệnh SELECT để xem dữ liệu hiển thị trong view này.
2. Tạo view có tên **vwNhanVienPhong5** dùng để liệt kê thông tin mã nhân viên, họ, tên lót, tên, tuổi, lương của các nhân viên làm việc ở phòng số 5. Sau đó sử dụng câu lệnh SELECT để xem dữ liệu hiển thị trong view này.
3. Tạo view có tên **vwThanNhanNVNu** dùng để liệt kê các nhân viên nữ (mã nhân viên, tên nhân viên) cùng với tên những người thân của họ.

4. Tạo view có tên **vwPhongLuongCao** để liệt kê tên các phòng ban có mức lương trung bình trên 7.000.000 đồng. Thông tin bao gồm tên phòng ban và số lượng đề án mà phòng ban đó chủ trì. Thực hiện truy vấn dữ liệu thông qua view.

5. Thông qua view **vwNhanVienNam**, thêm nhân viên với thông tin như sau:

MaNV	HoNV	TenLot	TenNV	NgSinh	Phong
246	Trần	Công	Minh	12/12/1992	1

Kiểm tra dữ liệu vừa nhập ở bảng NHANVIEN bằng câu lệnh SELECT.

6. Thông qua view **vwNhanVienNam**, chuyển nhân viên vừa mới nhập ở trên sang phòng 5.

7. Thông qua view **vwNhanVienNam**, xóa thông tin nhân viên vừa nhập.

8. Thông qua view **vwNhanVienPhong5**, thêm một nhân viên có thông tin như sau:

MaNV	HoNV	TenLot	TenNV	Tuoi	Luong
247	Lê	Minh	Hoàng	40	6200000

Việc thêm có thực hiện được hay không? Vì sao?

9. Sử dụng thủ tục **sp_helptext** để xem nội dung câu lệnh SELECT định nghĩa view **vw_NhanVienPhong5**.

10. Định nghĩa lại view **vwNhanVienPhong5** để giấu nội dung câu lệnh SELECT định nghĩa view này. Kiểm tra kết quả với thủ tục **sp_helptext**.

11. Định nghĩa lại view **vwThanNhanNVNu** để thông qua view, có thể biết được thông tin về giới tính của thân nhân các nhân viên nữ.

12. Nhân viên nữ có mã số 890 vừa mới kết hôn với người tên An. Hãy bổ sung thông tin này thông qua view **vwThanNhanNVNu**. Kiểm tra dữ liệu vừa nhập trong view **vwThanNhanNVNu** bằng câu lệnh SELECT.

13. Nhân viên nam có mã số 567 vừa mới kết hôn với người tên Thúy. Hãy bổ sung thông tin này thông qua view ***vw_ThanNhanNVNu***. Kiểm tra dữ liệu vừa nhập trong view ***vw_ThanNhanNVNu*** và trong table **THANNHAN** bằng câu lệnh **SELECT**. Sau đó đưa ra nhận xét.

14. Định nghĩa lại view ***vw_ThanNhanNVNu*** để ngăn cản việc thay đổi dữ liệu thân nhân cho các nhân viên nam thông qua view này. Sau đó, thông qua view, thử thêm một thân nhân có tên là Khang (là con trai của nhân viên nam có mã số 567). Việc này có thực hiện được hay không?

CHƯƠNG 4: LẬP TRÌNH T-SQL



MỤC TIÊU

Học xong chương này, người học có khả năng:

- Dùng biến trong SQL Server, các cấu trúc điều khiển và các câu lệnh trong SQL Server
- Tạo và sử dụng thủ tục trong SQL Server
- Hiểu được chức năng và cơ chế hoạt động của Trigger
- Tạo và sử dụng Trigger trong SQL Server

BÀI 1: BIẾN VÀ CẤU TRÚC ĐIỀU KHIỂN

1. Khái niệm về biến

Trong T-SQL có hai loại biến: biến cục bộ và biến hệ thống.

Cũng giống như các ngôn ngữ lập trình khác, biến cục bộ được khai báo bởi người lập trình và có phạm vi hoạt động giới hạn. Giá trị của biến cục bộ có thể được điều chỉnh bởi người lập trình. Tên biến cục bộ trong SQL Server luôn bắt đầu bằng ký tự @

Biến hệ thống là các biến do SQL Server cung cấp. Giá trị mà chúng đang lưu giữ là do hệ thống SQL Server cung cấp. Người lập trình không thể can thiệp trực tiếp để gán giá trị vào các biến hệ thống. Tên biến hệ thống trong SQL Server luôn bắt đầu bằng hai ký tự @@.

Danh sách các biến hệ thống thường dùng:

Tên biến	Kiểu trả về	Ý nghĩa
@@RowCount	số nguyên	tổng số mẫu tin được tác động vào câu lệnh truy vấn gần nhất
@@ServerName	chuỗi	tên của máy tính cục bộ được cài đặt SQL Server
@@ServiceName	chuỗi	tên dịch vụ chạy kèm bên dưới SQL Server
@@Fetch_Status	số nguyên	trạng thái của việc đọc dữ liệu trong bảng theo cơ chế từng dòng mẫu tin.

2. Làm việc với biến cục bộ

2.1. Khai báo biến

Để khai báo biến cục bộ, ta dùng lệnh DECLARE với cú pháp như sau:

```
DECLARE @tên_biến kiểu_dữ_liệu
```

Ví dụ: Để khai báo biến @tennv dùng để lưu trữ tên nhân viên, biến @tongnv dùng để lưu trữ tổng số lượng nhân viên, ta khai báo như sau:

```
DECLARE @tennv nvarchar(10)
DECLARE @tongnv int
```

2.2. Gán giá trị cho biến

Để gán giá trị cho biến, ta dùng lệnh SET hoặc SELECT cùng với phép gán (=). Thông thường, lệnh SET thường được dùng để gán giá trị cụ thể hoặc các biểu thức tính toán, còn lệnh SELECT thường được dùng để gán giá trị được lấy ra từ các cột trong bảng dữ liệu.

Ví dụ:

```
SET @tennv = N'Lê Văn Nam'
SELECT @tongnv = COUNT(*) FROM NHANVIEN
```

2.3. Xem giá trị hiện hành của biến

Để xem giá trị hiện hành của biến, ta có thể dùng lệnh PRINT

Ví dụ:

```
PRINT @tennv
```

3. Các cấu trúc điều khiển

3.1. Beginend

Một tập lệnh | khối lệnh SQL được thực thi sẽ được đặt trong BEGIN...END

Cú pháp:

```
BEGIN
    Lệnh/Khối lệnh
END
```

3.2. Cấu trúc rẽ nhánh IF.....ELSE

Cú pháp:

```
IF Biểu_thức_luận_lý
    BEGIN
        Khối_lệnh_1
    END
ELSE
    BEGIN
        Khối_lệnh_2
    END
```

Trong đó:

- **Biểu_thức_luận_lý:** trả về giá trị là đúng hoặc sai. Thông thường là một biểu thức so sánh
- **Khối_lệnh_1:** tập hợp các lệnh được thực hiện khi biểu_thức_luận_lý trả về giá trị đúng.
- **Khối_lệnh_2:** tập hợp các lệnh được thực hiện khi biểu_thức_luận_lý trả về giá trị sai.

Lưu ý:

- Nếu khối lệnh chỉ có duy nhất 1 câu lệnh thì không cần đặt trong cặp từ khóa BEGIN...END.
- Có thể kết hợp cấu trúc IF với từ khóa EXISTS để kiểm tra sự tồn tại của dữ liệu bên trong bảng.

Ví dụ: Liệt kê danh sách các nhân viên có mức lương trên 10 triệu đồng. Nếu không có, hiện ra câu thông báo: “không có nhân viên nào có mức lương trên 10 triệu đồng”.

```

IF EXISTS (SELECT MaNV FROM NHANVIEN WHERE Luong >
10000000)
BEGIN
    SELECT HoNV, TenLot, TenNV FROM NHANVIEN
    WHERE Luong > 10000000
END
ELSE
    PRINT "Không có nhân viên nào có mức lương trên
10 triệu đồng"

```

3.3 Cấu trúc CASE

- ❖ Trong T-SQL biểu thức CASE vô cùng hữu ích, nó giống như cú pháp SWITCH...CASE trong C.
- ❖ Được dùng để xử lý điều kiện cho cột được chọn trong câu lệnh truy vấn dữ liệu.
- ❖ Cú pháp của biểu thức CASE có 2 dạng.

Cú pháp dạng 1:

```

CASE <biểu thức>
    WHEN <gtri_1> THEN <kquả_1>
    [WHEN <gtri_2> THEN <kquả_2>
    ...
    ]
    [ELSE <kquả_n>]
END

```

Ví dụ: Cho biết họ tên các nhân viên và năm về hưu của họ

```

SELECT HONV, TENNV,
(CASE PHAI
    WHEN 'Nam' THEN YEAR(NGSINH) + 60
    WHEN 'Nu' THEN YEAR(NGSINH) + 55
END ) AS NAMVEHUU
FROM NHANVIEN

```

Cú pháp dạng 2:

```

CASE
    when <bt_logic_1> then <kquả_1>
    [when <bt_logic_2> then <kquả_2>
    ...
    ]
[ELSE <kquả_n>]
END

```

Ví dụ: Cho biết họ tên các nhân viên đã đến tuổi về hưu (nam 60 tuổi, nữ 55 tuổi)

```

SELECT HONV, TENNV
FROM NHANVIEN
WHERE YEAR(GETDATE()) - YEAR(NGSINH) >= ( CASE PHAI
                                            WHEN 'Nam' THEN 60
                                            WHEN 'Nu' THEN 55
                                            END )

```

3.4. Cấu trúc lặp WHILE

Cú pháp:

```

WHILE <Biểu thức luận lý>
BEGIN
    <Câu lệnh>
END

```

Trong đó:

- **Biểu thức luận lý:** thông thường là một biểu thức so sánh, các lệnh sẽ được lặp khi mà giá trị biểu thức vẫn còn đúng.

BÀI 2: THỦ TỤC NỘI TẠI (STORED PROCEDURE)

1. Khái niệm về Stored Procedure

Stored Procedure (thủ tục nội tại, thủ tục lưu trữ) là một đối tượng trong CSDL, là một tập hợp chứa các dòng lệnh, các biến và các cấu trúc điều khiển trong ngôn ngữ T-SQL, được dùng để thực hiện một hành động nhất định nào đó. Tất cả nội dung của Stored Procedure sẽ được lưu trữ tại CSDL của SQL Server.

Có hai loại Stored Procedure:

- ❖ **Stored Procedure của hệ thống:** là các Stored Procedure được SQL Server cung cấp sẵn, được dùng để thực hiện các xử lý trong việc quản trị CSDL. Hầu hết các Stored Procedure hệ thống được lưu trữ bên trong CSDL Master.
- ❖ **Stored Procedure do người dùng định nghĩa:** là các Stored Procedure được định nghĩa bởi người sử dụng để thực hiện một công việc cụ thể, được lưu trữ trong một CSDL nhất định và chỉ có phạm vi hoạt động bên trong CSDL đó. Trong phạm vi chương trình, ta chỉ tập trung tìm hiểu và thao tác với các Stored Procedure do người dùng tự định nghĩa.

2. Lợi ích của việc sử dụng Stored Procedure

Tốc độ xử lý của các Stored Procedure sẽ rất nhanh vì nội dung của Stored Procedure được lưu trữ và thực hiện ngay trên máy chủ. SQL Server chỉ biên dịch các Stored Procedure một lần và sử dụng lại kết quả biên dịch này trong các lần tiếp theo.

Việc sử dụng Stored Procedure sẽ làm giảm lưu lượng dữ liệu lưu thông trên mạng. Để thực hiện một yêu cầu, người sử dụng chỉ cần thực hiện một câu lệnh đơn giản thay vì phải sử dụng một tập câu lệnh T-SQL.

Giúp tăng cường khả năng bảo mật đối với hệ thống bằng việc phân quyền cho người sử dụng thông qua các Stored Procedure.

Khắc phục nhược điểm không có tham số của view.

3. Các hành động cơ bản với Stored Procedure

3.1. Tạo Stored Procedure

Cú pháp câu lệnh T-SQL để tạo mới một Stored Procedure:

```
CREATE PROCEDURE Tên_Stored_Procedure[(danh sách tham
số) ]
AS
BEGIN
    Các câu lệnh T-SQL
END
```

Trong đó:

- Tên_Stored_Procedure: tên Stored Procedure, người ta thường sử dụng tiếp đầu ngữ sp_ để đặt tên cho Stored Procedure.
- Danh sách tham số: danh sách tham số đầu vào và đầu ra của Stored Procedure. Các tham số chỉ có phạm vi cục bộ bên trong Stored Procedure mà nó được khai báo. Tên của mỗi tham số là duy nhất.
- ❖ Một số lưu ý khi tạo Stored Procedure bằng câu lệnh T-SQL:
 - Có thể thay thế từ khóa PROCEDURE bằng từ khóa viết tắt PROC.
 - Một Stored Procedure có thể không có tham số hoặc có nhiều tham số.
 - Các tham số được ngăn cách nhau bởi dấu phẩy. Riêng đối với các tham số đầu ra, phải đặt từ khóa OUTPUT ở phía sau.
 - Mỗi tham số bao gồm hai thành phần: tên tham số (theo quy cách đặt tên biến trong T-SQL) và kiểu dữ liệu của tham số đó.
 - Giá trị các tham số đầu ra sẽ được thiết lập bên trong nội dung các câu lệnh T-SQL bằng câu lệnh SELECT hoặc SET.

3.2. Thực thi Stored Procedure

Sau khi một Stored Procedure được tạo ra, có thể sử dụng lệnh EXECUTE để thực thi Stored Procedure.

Cú pháp:

```
EXECUTE Tên_Stored_Procedure [danh sách các tham số]
```

Lưu ý:

- Có thể thay thế từ khóa EXECUTE bằng từ khóa viết tắt là EXEC
- Danh sách các tham số truyền vào phải đúng với số lượng và thứ tự khai báo các tham số khi định nghĩa Stored Procedure

- Các tham số được truyền vào có thể là các biến hoặc các giá trị phù hợp với kiểu dữ liệu được khai báo trong lệnh tạo Stored Procedure.
- Đối với các tham số đầu ra, giá trị truyền vào là các biến kèm theo từ khóa OUTPUT

3.3. Thay đổi nội dung Stored Procedure

Có thể thay đổi nội dung một Stored Procedure đã được tạo ra bằng câu lệnh ALTER PROCEDURE.

Cú pháp:

```
ALTER PROCEDURE Tên_Stored_Procedure[ (danh sách tham
số) ]
AS
    Các câu lệnh T-SQL
```

Câu lệnh này được sử dụng tương tự như câu lệnh CREATE PROCEDURE. Việc thay đổi nội dung một Stored Procedure đã có không làm thay đổi các quyền đã cấp phát trên Stored Procedure cũng như không tác động đến các Stored Procedure khác.

3.4. Hủy Stored Procedure

Để hủy một Stored Procedure không còn sử dụng, ta sử dụng câu lệnh DROP PROCEDURE có cú pháp như sau:

```
DROP PROCEDURE Tên_Stored_Procedure
```

4. Tham số trong Stored Procedure

4.1. Stored Procedure không chứa tham số

Ví dụ: Tạo Stored Procedure liệt kê danh sách tất cả các nhân viên của công ty.

```
CREATE PROCEDURE sp_LietKeNhanVien
AS
SELECT * FROM NHANVIEN
```

Thực thi Stored Procedure:

```
EXEC sp_LietKeNhanVien
```

4.2. Stored Procedure có một tham số

Ví dụ: Tạo Stored Procedure liệt kê danh sách các nhân viên của từng phòng với mã phòng được nhập vào khi gọi Stored Procedure.

```
CREATE PROCEDURE sp_LietKeNhanVienTheoPhong @maphong
varchar(3)
AS
SELECT * FROM NHANVIEN WHERE Phong = @maphong
```

Để xem danh sách các nhân viên phòng 1, ta thực thi Stored Procedure như sau:

```
EXEC sp_LietKeNhanVienTheoPhong '1'
```

4.3. Stored Procedure có nhiều tham số

Ví dụ: Tạo Stored Procedure cho phép thêm một đề án mới với tham số truyền vào là các thông tin về mã đề án, tên đề án, địa điểm làm việc của đề án và mã phòng chủ trì đề án đó. Trước khi thêm, cần kiểm tra ràng buộc khóa chính và khóa ngoại đối với các table liên quan.

```
CREATE PROCEDURE sp_ThemDeAn (@mada varchar(2), @tenda
nvarchar(50), @ddiemda nvarchar(20), @maphong
varchar(2))
AS
--kiểm tra việc trùng khóa chính
IF EXISTS (SELECT * FROM DEAN WHERE MaDA = @mada)
    PRINT(N'Mã đề án đã tồn tại trong bảng DEAN')
--kiểm tra việc tồn tại mã phòng trong bảng
PHONGBAN
ELSE IF NOT EXISTS (SELECT * FROM PHONGBAN WHERE
MaPhg = @maphong)
    PRINT(N'Mã phòng chưa tồn tại trong
bảng PHONGBAN')
ELSE
    INSERT INTO DEAN VALUES (@mada,
@tenda, @ddiemda, @maphong)
```

Để thêm một đề án mới, ta thực thi Stored Procedure như sau:

```
EXEC sp_ThemDeAn '22', N'ABCD', N'Trà Bồng', '7'
```

4.4. Stored Procedure có tham số đầu ra

Ví dụ : Tạo Stored Procedure để tính tổng số giờ làm việc của nhân viên với mã nhân viên được truyền vào khi gọi Stored Procedure.

```
CREATE PROCEDURE sp_TinhTongGio @manv varchar(3),
@tonggio int OUTPUT
AS
BEGIN
    --kiểm tra mã nhân viên đã tồn tại trong bảng
    NHANVIEN
    IF NOT EXISTS (SELECT * FROM NHANVIEN WHERE MaNV =
@manv)
        PRINT(N'Mã nhân viên chưa tồn tại trong bảng
NHANVIEN')
    ELSE
        SELECT @tonggio = SUM(ThoiGian)
        FROM PHANCONG
        WHERE MaNV = @manv
END
```

Để xem tổng số giờ làm việc của nhân viên mang mã số 123, ta thực thi Stored Procedure như sau:

```
DECLARE @tong int
EXEC sp_TinhTongGio '321', @tong OUTPUT
PRINT @tong
```

5. Trả về giá trị trong Stored Procedure

5.1. Trả về giá trị từ câu lệnh Return

Lệnh RETURN được sử dụng để trả về giá trị từ stored procedure mà không cần sử dụng tham số đầu ra. Giá trị trả về này có một số đặc điểm:

- Giá trị trả về chỉ có thể là số nguyên. Nếu trả về các loại giá trị khác thì lúc thực thi stored procedure sẽ báo lỗi (ngoại trừ 1 số kiểu dữ liệu được tự động chuyển đổi sang kiểu số nguyên như: float, double,...).
- Giá trị trả về mặc định là 0.
- Có thể nhận giá trị trả về này bằng 1 biến.

- Sau khi gọi RETURN, stored procedure sẽ trả về giá trị và kết thúc xử lý.

5.2. Trả về dữ liệu từ câu lệnh

Mỗi lệnh SELECT đặt trong stored procedure sẽ trả về 1 bảng.

```
CREATE PROC
TestSelect AS

BEGIN

    SELECT * FROM SINHVIEN

    SELECT * FROM LOP

END

GO

EXEC TestSelect
```

BÀI TẬP KẾT THÚC

LÝ THUYẾT

1. Trình bày chức năng của Stored Procedure trong cơ sở dữ liệu.
2. Phân biệt tham số input và tham số output.
3. So sánh sự giống và khác nhau về chức năng giữa Stored Procedure trong SQL Server và hàm trong ngôn ngữ lập trình C.

THỰC HÀNH

1. Viết SP **spTangLuong** dùng để tăng lương lên 10% cho tất cả các nhân viên.
2. Thêm vào cột NgayNghỉHuu (ngày nghỉ hưu) trong bảng NHANVIEN. Viết SP **spNghỉHuu** dùng để cập nhật ngày nghỉ hưu là ngày hiện tại cộng thêm 100 (ngày) cho những nhân viên nam có tuổi từ 60 trở lên và nữ từ 55 trở lên.
3. Tạo SP **spXemDeAn** cho phép xem các đề án có địa điểm đề án được truyền vào khi gọi thủ tục.
4. Tạo SP **spCapNhatDeAn** cho phép cập nhật lại địa điểm đề án với 2 tham số truyền vào là diadiem_cu, diadiem_moi.
5. Viết SP **spThemDeAn** để thêm dữ liệu vào bảng DEAN với các tham số vào là các trường của bảng DEAN.
6. Cập nhật SP **spThemDeAn** ở câu trên để thỏa mãn ràng buộc sau: kiểm tra mã đề án cần chèn có rỗng hoặc trùng với các mã đề án khác không. Nếu có thì

thông báo lỗi “Mã đề án rỗng hoặc trùng, đề nghị chọn mã đề án khác”. Thực thi thủ tục với 1 trường hợp đúng và 2 trường hợp sai để kiểm chứng.

7. Tạo SP **spXoaDeAn** cho phép xóa các đề án với tham số truyền vào là Mã đề án. Lưu ý trước khi xóa cần kiểm tra mã đề án có tồn tại trong bảng PHANCONG hay không, nếu có thì viết ra thông báo và không thực hiện việc xóa dữ liệu.

8. Tạo SP **spTongGioLamViec** có tham số truyền vào là MaNV, tham số ra là tổng thời gian (tính bằng giờ) làm việc ở tất cả các dự án của nhân viên đó.

9. Viết SP **spTongTien** để in ra màn hình tổng tiền phải trả cho nhân viên với tham số truyền vào là mã nhân viên. (Tổng tiền phải trả cho nhân viên = lương + lương đề án; lương đề án = 100000 đ x thời gian). Kết quả của thủ tục là dòng chữ: “Tổng tiền phải trả cho nhân viên ‘333’ là 1200000 đồng.

10. Viết SP **spThemPhanCong** để thêm dữ liệu vào bảng PHANCONG thỏa mãn yêu cầu sau: ThoiGian phải là một số dương và MaDA phải tồn tại ở bảng DEAN. Nếu không thỏa mãn phải thông báo lỗi tương ứng và không được phép thêm dữ liệu.

BÀI 3: BẦY (TRIGGER)

1. Khái niệm trigger

Trigger là đối tượng trong CSDL, sẽ được thực thi tự động khi có sự thay đổi dữ liệu trên một bảng cụ thể. Trigger được sử dụng để đảm bảo tính toàn vẹn dữ liệu (Data Integrity) hay thực hiện các ràng buộc dữ liệu (Business Rule) nào đó.

Một số tính chất của Trigger:

- Trigger luôn gắn liền với một hoặc nhiều thao tác thay đổi dữ liệu trong một Table cụ thể.
- Trigger được thực thi tự động khi có sự thay đổi dữ liệu tương ứng trong Table.
- Như vậy, trước khi tạo một Trigger, cần phải xác định:
- Trigger được sử dụng cho Table nào.
- Các hành vi thay đổi dữ liệu nào (thêm, xóa, sửa) cần phải kích hoạt Trigger.

2. Các trường hợp nên dùng trigger

- ❖ Khi người dùng muốn tự kiểm soát các Constraint và cho ra các câu thông báo thích hợp khi có sự thay đổi dữ liệu vi phạm Constraint.
- ❖ Khi có sự thay đổi dữ liệu trên một Table thì dữ liệu trên một hay nhiều Table khác cũng tự động thay đổi theo cho phù hợp.
- ❖ Khi có thao tác thay đổi dữ liệu (thêm, xóa, sửa) thì một hay một vài sự kiện được thực hiện tự động phía server.

3. Cơ chế hoạt động của trigger

❖ *Bảng Inserted và bảng Deleted:*

- Là hai bảng trong bộ nhớ chính.
- Khi có thao tác thêm dữ liệu vào một bảng, các mẫu tin sẽ được lưu trữ vào bảng, đồng thời chúng sẽ được lưu trữ vào bảng Inserted.
- Khi có thao tác xóa dữ liệu từ một bảng, các mẫu tin sẽ được xóa ra khỏi bảng, đồng thời chúng sẽ được lưu trữ vào bảng Deleted.
- Hai bảng Inserted và Deleted có cấu trúc giống với cấu trúc của bảng dữ liệu liên quan đến trigger khi tạo ra.

- Thao tác cập nhật dữ liệu chính là hai thao tác xóa dữ liệu cũ và thêm dữ liệu mới được thực hiện liên tiếp nhau. Khi đó, bảng Deleted sẽ lưu trữ các mẫu tin cũ trước khi sửa, bảng Inserted sẽ lưu trữ các mẫu tin mới sau khi sửa.
- Hai bảng Inserted và Deleted chỉ tồn tại trong thời gian mà trigger đang xử lý.

❖ **Cơ chế hoạt động:**

- Khi thực hiện thêm mới mẫu tin vào một table, thao tác này sẽ kích hoạt một trigger, trigger sẽ lưu trữ dữ liệu của mẫu tin vừa thêm mới vào table Inserted. Tương tự, khi thực hiện việc xóa mẫu tin của table, thao tác này sẽ kích hoạt một trigger, trigger sẽ lưu trữ dữ liệu của mẫu tin vừa xóa vào table Deleted.
- Khi có một biến cố xảy ra, trigger sẽ được thực thi một cách tự động. Các câu lệnh bên trong trigger có nhiệm vụ lấy thông tin dữ liệu từ các table Inserted và Deleted để thực hiện các công việc liên quan.

4. Tạo trigger

Cú pháp:

```
CREATE TRIGGER Tên_Trigger
ON {Tên_Table | Tên_View}
{FOR | AFTER | INSTEAD OF} {[INSERT][,] [DELETE][,]
[UPDATE]}
AS
BEGIN
    Các câu lệnh T-SQL
END
```

Trong đó:

- Tên_Table, Tên_View: Table hoặc View chịu sự tác động của Trigger
- FOR | AFTER | INSTEAD OF:
 - Nếu sử dụng FOR hoặc AFTER: trigger sẽ được thực thi sau khi các thao tác thay đổi dữ liệu được thực hiện thành công.
 - Nếu sử dụng INSTEAD OF: trigger được thực thi thay cho các thao tác thay đổi dữ liệu.

5. Các kiểu trigger

Trigger chia thành 2 loại: INSTEAD OF và AFTER:

❖ **INSTEAD OF:** là trigger mà thao tác kích hoạt trigger sẽ bị bỏ qua và thay vào đó là các lệnh trong trigger được thực hiện.

❖ **AFTER:** là loại trigger mặc định, trigger này thực hiện các lệnh bên trong trigger sau khi đã thực hiện thao tác kích hoạt trigger.

6. Thay đổi nội dung trigger

Để thay đổi nội dung các câu lệnh bên trong trigger, người dùng có thể xóa trigger và tạo lại. Tuy nhiên, người sử dụng có thể thay đổi nội dung thông qua câu lệnh ALTER TRIGGER.

Cú pháp của câu lệnh ALTER TRIGGER cũng giống như câu lệnh CREATE TRIGGER, nhưng ALTER TRIGGER không gỡ bỏ trigger khỏi CSDL.

❖ Làm cho trigger mất hiệu lực (disable trigger)

Trong một số trường hợp, người sử dụng cần tạm thời làm mất hiệu lực các trigger như cô lập vấn đề cần gỡ rối, sửa đổi dữ liệu, chuyển dữ liệu giữa các CSDL. Có thể sử dụng câu lệnh ALTER TABLE để thực hiện việc làm mất hiệu lực tạm thời của các trigger.

Cú pháp:

```
ALTER TABLE Tên_Table
DISABLE TRIGGER Tên_Trigger
```

Lưu ý:

Trong trường hợp cần tắt tất cả các trigger trên một table, có thể thay tên của trigger bằng từ khóa ALL

Ví dụ: để làm mất hiệu lực tạm thời tất cả các trigger trên bảng NHANVIEN, ta thực hiện câu lệnh như sau:

```
ALTER TABLE NHANVIEN
DISABLE TRIGGER ALL
```

❖ Làm cho trigger có hiệu lực (enable trigger)

Đối với các trigger đang tạm thời mất hiệu lực, có thể làm cho chúng có hiệu lực trở lại bằng câu lệnh ALTER TABLE.

Cú pháp:

```
ALTER TABLE Tên_Table
ENABLE TRIGGER Tên_Trigger
```

Lưu ý:

Trong trường hợp cần bật tất cả các trigger trên một table, có thể thay tên của trigger bằng từ khóa ALL

Ví dụ: để làm có hiệu lực tất cả các trigger trên bảng NHANVIEN, ta thực hiện câu lệnh như sau:

```
ALTER TABLE NHANVIEN
ENABLE TRIGGER ALL
```

7. Xóa trigger

Có thể sử dụng câu lệnh DROP TRIGGER để xóa một trigger ra khỏi hệ thống.

Cú pháp:

```
DROP TRIGGER Tên_Trigger
```

BÀI TẬP KẾT THÚC

LÝ THUYẾT

1. Trình bày khái niệm Trigger. Tạo sao nói Trigger là một dạng đặc biệt của Stored Procedure?
2. Hai bảng INSERTED là DELETED là gì? Chức năng của hai bảng nói trên.
3. Khi nào nên sử dụng từ khóa FOR, AFTER, INSTEAD OF?
4. Trình bày chức năng của câu lệnh ROLLBACK TRANSACTION.

THỰC HÀNH

Sử dụng cơ sở dữ liệu QL_DEAN đã thực hiện ở các chương trước, thực hiện các yêu cầu sau:

1. Tạo trigger trên bảng NHANVIEN cho thao tác UPDATE. Khi có thao tác cập nhật dữ liệu xảy ra trên cột TenNV, thông báo cho người dùng “Không được phép cập nhật” và hủy thao tác.
2. Thêm cột TongGio vào trong bảng NHANVIEN. Viết trigger cho các thao tác INSERT, UPDATE, DELETE trên bảng PHANCONG. Khi

có mẫu tin được thêm vào, cập nhật hay xóa thì TongGio được tính lại tương ứng cho nhân viên được phân công.

Lưu ý:

- Ban đầu, TongGio = 0
- TongGio là tổng thời gian phân công tham gia vào các dự án cho các nhân viên.

3. Tạo các trigger để kiểm tra ràng buộc liên thuộc tính giữa NgSinh và NgayBatDau trên bảng NHANVIEN.

Trong đó: $\text{YEAR}(\text{NgayBatDau}) - \text{YEAR}(\text{NgSinh}) \geq 18$

4. Tạo trigger để kiểm tra ràng buộc trên bảng THANNHAN sao cho số lượng thân nhân của một nhân viên không quá 05 người.

5. Tạo trigger cho thao xóa trên bảng DEAN để đảm bảo nguyên tắc: Mã đề án sẽ không thể được xóa khi còn mẫu tin chứa mã đề án đó trên bảng PHANCONG.

6. Tạo trigger trên bảng PHONGBAN cho thao tác UPDATE. Khi có thao tác cập nhật dữ liệu xảy ra trên cột MaPhong, tất cả dữ liệu trên các bảng có liên quan cũng phải thay đổi theo.