

TRƯỜNG ĐẠI HỌC PHẠM VĂN ĐỒNG
KHOA CÔNG NGHỆ THÔNG TIN

PHẠM KHÁNH BẢO

BÀI GIẢNG
SQL SERVER

Quảng Ngãi, tháng 12 năm 2016

TRƯỜNG ĐẠI HỌC PHẠM VĂN ĐỒNG
KHOA CÔNG NGHỆ THÔNG TIN

PHẠM KHÁNH BẢO

BÀI GIẢNG
SQL SERVER

Dành cho sinh viên bậc đại học
ngành Công nghệ thông tin và Sư phạm tin

----TÀI LIỆU LƯU HÀNH NỘI BỘ----

LỜI NÓI ĐẦU

SQL Server là học phần cơ sở ngành 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. Từ năm học 2015-2016, môn học này là học phần tích hợp theo chương trình hợp tác đào tạo giữa trường ĐH Phạm Văn Đồng và công ty phần mềm FPT Software.

Bài giảng được thiết kế dành cho sinh viên đại học các ngành Công nghệ thông tin và Sư phạm tin học. Nội dung được xây dựng theo đúng chương trình chi tiết của học phần SQL Server, được áp dụng từ năm học 2016-2017.

Để thuận tiện cho việc giảng dạy và học tập, đầu mỗi chương sẽ mô tả thời lượng giảng dạy và mục tiêu tối thiểu mà sinh viên cần đạt được. Đối với các chương có phần thực hành, bài tập thực hành sẽ được trình bày cụ thể sau mỗi chương. Để thuận tiện cho việc thực hành của sinh viên, các bài thực hành sẽ tập trung vào một cơ sở dữ liệu duy nhất. Do đó, để học tập tốt học phần này, sinh viên cần phải thực hiện tất cả các bài tập thực hành ở chương trước trước khi tiếp tục học lý thuyết trong chương mới.

Hy vọng bài giảng này sẽ là tài liệu bổ ích dành cho sinh viên ngành Công nghệ thông tin và Sư phạm tin học của trường Đại học Phạm Văn Đồng.

Chương 1: TỔNG QUAN VỀ SQL SERVER

Thời lượng: 02 tiết lý thuyết

Kết thúc chương này, sinh viên có thể:

- ☞ *Hiểu được mô hình Client/Server*
- ☞ *Hiểu được khái niệm CSDL và chức năng của hệ quản trị cơ sở dữ liệu*
- ☞ *Biết được các tiện ích và tiện ích trong SQL Server*
- ☞ *Biết được các thành phần trong SQL Server*
- ☞ *Làm quen với giao diện công cụ SQL Server Management Studio*

1.1. MÔ HÌNH CLIENT/SERVER

Mô hình Client/Server (mô hình khách/chủ) là một hệ thống mạng máy tính được kết nối với nhau, bao gồm các thành phần sau:



Hình 1.1. Mô hình Client/Server

- **Server (máy chủ):** Là máy tính có bộ xử lý với tốc độ cao, tài nguyên bộ nhớ lớn để hoạt động được tốt bởi cùng lúc sẽ có nhiều người dùng mạng truy xuất đến server thông qua các máy client. Trong một hệ thống mạng máy tính có thể có nhiều server với các chức năng độc lập riêng biệt khác nhau.

- **Client (máy khách):** là các máy tính được phép truy xuất đến các tài nguyên được chia sẻ trên mạng.

- **Dây cáp mạng:** là hệ thống dây kim loại hoặc quang học kết nối các máy tính, máy in lại với nhau.

- **Dữ liệu chung:** là các tập tin, thư mục mà người sử dụng trong hệ thống mạng có thể truy xuất đến server từ máy client.

Trong mô hình client/server, việc tổ chức các xử lý phải đảm bảo các yêu cầu (request) từ client phải được server hồi đáp (response) một cách nhanh chóng và không làm tắc nghẽn hệ thống mạng máy tính.

Ưu điểm của việc phát triển ứng dụng trên mô hình Client/Server:

- Giảm chi phí
- Tốc độ nhanh
- Tính tương thích cao

1.2. CƠ SỞ DỮ LIỆU VÀ HỆ QUẢN TRỊ CƠ SỞ DỮ LIỆU

1.2.1. Cơ sở dữ liệu

Trong thực tế cuộc sống, dữ liệu được mô tả dưới nhiều dạng khác nhau (ký tự, ký số, hình ảnh, ký hiệu, âm thanh...). Mỗi cách mô tả như vậy được gắn với một ngữ nghĩa nào đó.

Ví dụ, một đối tượng sinh viên thực tế có rất nhiều thông tin khác nhau liên quan đến bản thân sinh viên đó như: tên, địa chỉ, ngày sinh, lớp, điểm số ... sở thích, tác phong, quê quán, gia đình, quan hệ cộng đồng, cân nặng, chiều cao... Tuy nhiên, với mục đích quản lý (thông thường) của nhà trường thì không phải tất cả các thông tin kia đều được lưu trữ. Tùy mục đích, nhà trường có thể cần lưu trữ:

- tên, địa chỉ, ngày sinh, lớp, điểm số.
- hoặc tên, địa chỉ, ngày sinh, lớp, điểm số, quê quán, họ tên cha mẹ,...
- hoặc một tập hợp thông tin khác.

Để quản lý thông tin của sinh viên bằng cách tin học hóa, Nhà trường cần thiết kế một cơ sở dữ liệu quản lý sinh viên.

Cơ sở dữ liệu (database), viết tắt là CSDL hoặc DB, là một tập hợp các dữ liệu có quan hệ logic với nhau, có thể dễ dàng chia sẻ và được thiết kế nhằm đáp ứng các nhu cầu sử dụng của một tổ chức, cá nhân nào đó.

Một cách định nghĩa khác dễ hiểu hơn, CSDL là một tập hợp có cấu trúc của những dữ liệu có liên quan với nhau được lưu trữ trong máy tính (bảng chấm công

nhân viên, danh sách các đề án, niên giám điện thoại, kết quả học tập của sinh viên, các mặt hàng trong siêu thị...). Một CSDL được thiết kế, xây dựng và lưu trữ với một mục đích xác định như phục vụ lưu trữ, truy xuất dữ liệu cho các ứng dụng hay người dùng.

1.2.2. Hệ quản trị cơ sở dữ liệu

Hệ quản trị cơ sở dữ liệu (Database Management System – DBMS) là một phần mềm hay hệ thống được thiết kế để quản trị cơ sở dữ liệu. Các hệ quản trị CSDL thực hiện các chức năng lưu trữ, sửa chữa, xóa, tìm kiếm thông tin trong một CSDL. Có rất nhiều loại hệ quản trị CSDL khác nhau, từ phần mềm chạy trên máy tính cá nhân đến các hệ thống lớn chạy trên các siêu máy tính.

Đa số các hệ quản trị CSDL đều sử dụng ngôn ngữ có cấu trúc SQL (Structured Query Language). Các hệ quản trị CSDL phổ biến được nhiều người biết đến là MySQL, Oracle, SQL Server, DB2...

Ưu điểm của hệ quản trị CSDL:

- Đảm bảo tính nhất quán và toàn vẹn dữ liệu
- Hạn chế dữ liệu dư thừa
- Có khả năng chia sẻ dữ liệu rất lớn

Nhược điểm của hệ quản trị CSDL:

- Hệ quản trị CSDL tốt thì khá phức tạp và tốn nhiều bộ nhớ

1.3. HỆ QUẢN TRỊ CSDL MICROSOFT SQL SERVER

Microsoft SQL Server là một hệ quản trị cơ sở dữ liệu quan hệ của hãng phần mềm Microsoft, hoạt động theo mô hình Client/Server, cho phép nhiều người truy xuất đến dữ liệu cùng lúc, quản lý việc truy nhập hợp lệ và các quyền hạn của từng người dùng trên CSDL.

SQL Server có 7 phiên bản:

- **Enterprise:** Chứa đầy đủ các đặc trưng của SQL Server và có thể chạy tốt trên hệ thống lên đến 32 CPUs và 64 GB RAM. Thêm vào đó nó có các dịch vụ giúp cho việc phân tích dữ liệu rất hiệu quả (Analysis Services)

- **Standard:** Rất thích hợp cho các công ty vừa và nhỏ vì giá thành rẻ hơn nhiều so với Enterprise Edition, nhưng lại bị giới hạn một số chức năng cao cấp (advanced features) khác, edition này có thể chạy tốt trên hệ thống lên đến 4 CPU và 2 GB RAM.

- **Personal:** được tối ưu hóa để chạy trên PC nên có thể cài đặt trên hầu hết các phiên bản windows kể cả Windows 98.

- **Developer:** Có đầy đủ các tính năng của phiên bản Enterprise nhưng được chế tạo đặc biệt như giới hạn số lượng người kết nối vào Server cùng một lúc.... Đây là phiên bản mà các bạn muốn học SQL Server cần có. Chúng ta sẽ dùng edition này trong suốt khóa học. Phiên bản này có thể cài trên Windows 2000 Professional hay Win NT Workstation.

- **Desktop Engine (MSDE):** Đây chỉ là một engine chạy trên desktop và không có user interface (giao diện). Thích hợp cho việc triển khai ứng dụng ở máy client. Kích thước database bị giới hạn khoảng 2 GB.

- **Win CE:** Dùng cho các ứng dụng chạy trên Windows CE

- **Trial:** Có các tính năng của phiên bản Enterprise, download free, nhưng giới hạn thời gian sử dụng.

1.4. CÁC THÀNH PHẦN TRONG SQL SERVER

SQL Server được cấu thành bởi nhiều thành phần khác nhau, các thành phần có mối quan hệ trong một hệ thống, phối hợp với nhau để tạo thành một giải pháp hoàn chỉnh, nâng cao hiệu quả quản trị, phân tích, lưu trữ dữ liệu.

- **SQL Server Database:** Là cỗ máy CSDL bao gồm Database Engine, lõi dịch vụ cho việc lưu trữ, xử lý và bảo mật dữ liệu, sao lưu và đồng bộ, tìm kiếm toàn văn (Full-text Search) và các công cụ cho việc quản trị dữ liệu quan hệ và XML.

- **Relational Database Engine:** Đây là một engine có khả năng chứa dữ liệu dưới nhiều quy mô khác nhau, theo dạng bảng, hỗ trợ nhiều phương thức kết nối thông dụng của Microsoft như ADO, OLE DB, ODBC.

- **Replication:** Là công cụ dùng nhân bản dữ liệu, người dùng có thể tạo một Server khác với bộ dữ liệu giống bộ dữ liệu trên Server chính. Công cụ tạo cơ chế tự đồng bộ dữ liệu giữa Server chính và Server nhân bản. Mục đích của việc tạo Server nhân bản là giảm tải cho Server chính, nâng cao hiệu quả phục vụ với số lượng người, phiên giao dịch lớn.

- **Data Transformation Service (DTS):** Là công cụ giúp bạn chuyển dữ liệu giữa các Server quản trị CSDL khác nhau, DTS có thể chuyển dữ liệu từ SQL Server sang Oracle, Access, DB,... trước khi chuyển dữ liệu DTS định dạng kiểu dữ liệu để chuyển sang hệ quản trị CSDL khác.

- **Analysis service:** Là công cụ giúp khai thác phân tích dữ liệu, hay khai phá dữ liệu theo phương thức đa chiều. Từ một tập dữ liệu sẵn có bạn có thể khai phá rồi từ đó đưa ra những nhận định, phân tích, đánh giá và dự đoán theo lĩnh vực nào đó, mỗi chiều trong ngữ cảnh này được coi là một tiêu chí xem xét của dữ liệu.

- **English query:** Đây là công cụ tra cứu dữ liệu bằng tiếng Anh, cú pháp có thể sử dụng theo văn phạm tiếng Anh thông thường.

- **SQL Server tools:** Là bộ công cụ cung cấp giao diện cho người quản trị như Enterprise manager, Query Analyzer ,...SQL Server sau khi cài đặt SQL Server group gồm những thành phần cơ bản trong group như sau:

1.5. CÁC TIỆN ÍCH TRONG SQL SERVER

- **Tiện ích Books Online:** cho phép người dùng tra cứu trực tuyến tất cả các thông tin liên quan đến Microsoft SQL Server một cách đầy đủ với các tính năng tìm kiếm dễ dàng và một giao diện dễ sử dụng.

- **Tiện ích SQL Native Client Configuration:** cho phép người sử dụng thay đổi, tạo mới các giao thức kết nối mạng mặc định của máy client khi thực hiện kết nối vào SQL Server tại các máy chủ.

- **Tiện ích SQL Server Management Studio:** giúp người sử dụng quản trị một hoặc nhiều Server khác nhau. Với giao diện đồ họa thân thiện, tiện ích này giúp người sử dụng tạo CSDL và các thành phần khác bên trong SQL Server một cách dễ dàng.

- **Tiện ích Import and Export Data:** cho phép nhập (import), xuất (export) và chuyển đổi dữ liệu qua lại giữa SQL Server và những loại cơ sở dữ liệu khác như Access, Visual Foxpro, Excel...

- **Tiện ích Performance Monitor:** thể hiện biểu đồ hoạt động của các tài nguyên trên máy chủ.

- **Tiện ích Profiler:** thể hiện các biến cố đã xảy ra khi SQL Server đang thực hiện một xử lý nào đó trên máy chủ. Các biến cố này được ghi lại trong một tập tin lưu vết (trace file) để sau này sử dụng lại cho việc phân tích nhằm phát hiện ra những vấn đề khi thực hiện các câu lệnh truy vấn trong xử lý đó.

- **Tiện ích Query Editor:** giúp người sử dụng soạn thảo các câu lệnh SQL. Các câu lệnh này sẽ được truy vấn trực tiếp trên CSDL SQL Server và nhận được kết quả ngay sau khi thực hiện truy vấn đó.

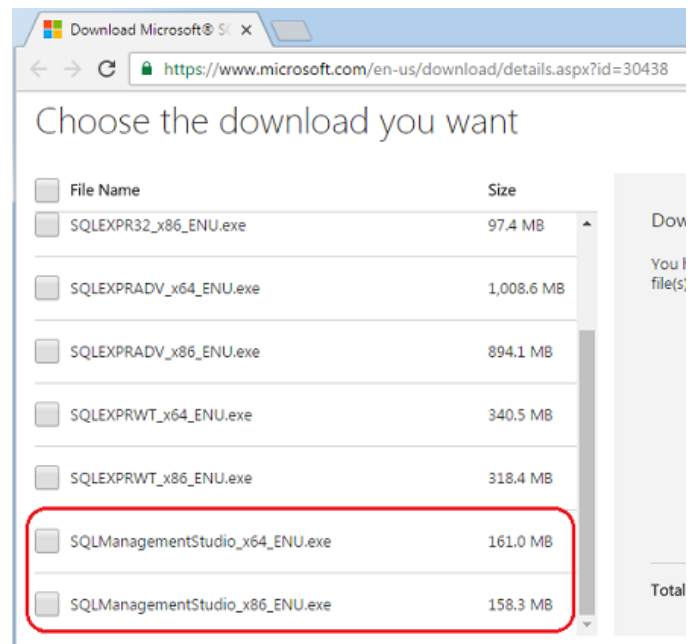
- **Tiện ích SQL Server Network Configuration:** dùng để quản lý các thư viện giao thức kết nối mạng của các máy server dùng để lắng nghe các yêu cầu từ máy client.

- **Tiện ích SQL Server Services:** dùng để quản lý các dịch vụ liên quan đến SQL Server. Có thể thực hiện việc khởi động (start), tạm dừng (pause), và dừng (stop) các dịch vụ đó. Các dịch vụ (service) này được xem như là các ứng dụng chạy ngầm định bên dưới hệ thống trong môi trường Windows.

1.6. SỬ DỤNG TIỆN ÍCH SQL SERVER MANAGEMENT STUDIO

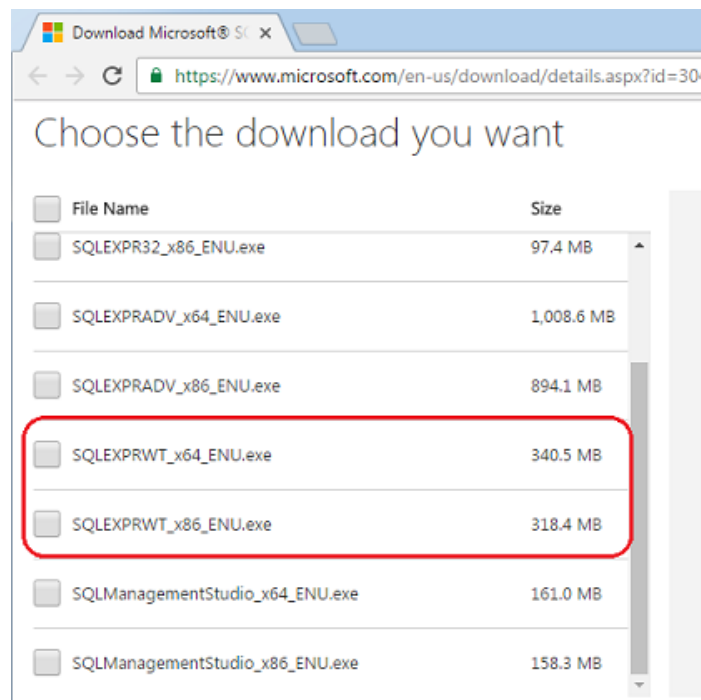
SQL Server Management Studio là một công cụ trực quan để quản lý SQL Server. Với SQL Server Management Studio, người sử dụng có thể thực hiện các tương tác với database trên giao diện người dùng hoặc bằng câu lệnh. SQL Server Management Studio được thiết kế với giao diện thân thiện và dễ sử dụng.

SQL Server có thể được cài đặt riêng sau khi cài đặt SQL Server bằng cách download file cài đặt *SQLManagementStudio_x64_ENU.exe* hoặc *SQLManagementStudio_x86_ENU.exe* để tiến hành cài đặt.



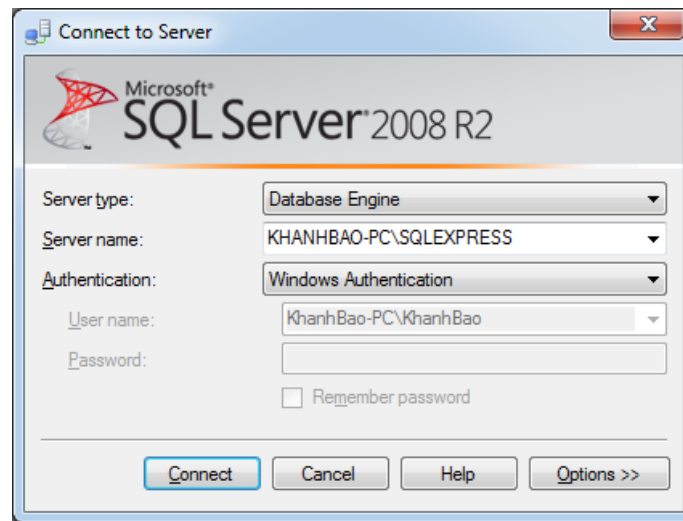
Hình 1.2. Các file cài đặt SQL Server Management Studio

Ngoài ra, người sử dụng có thể cài đặt chung công cụ này trong khi cài SQL Server bằng cách download file cài đặt *SQLEXPRT_x64_ENU.exe* hoặc *SQLEXPRT_x86_ENU.exe*.



Hình 1.3. Các file cài đặt SQL Server và SQL Server Management Studio

Sau khi mở SQL Server Management Studio, việc đầu tiên là tiến hành kết nối đến Server bằng cách chọn Server type, Server name và chế độ chứng thực (Authentication) phù hợp. Sau đó nhấn nút Connect.



Hình 1.4. Giao diện đăng nhập SQL Server

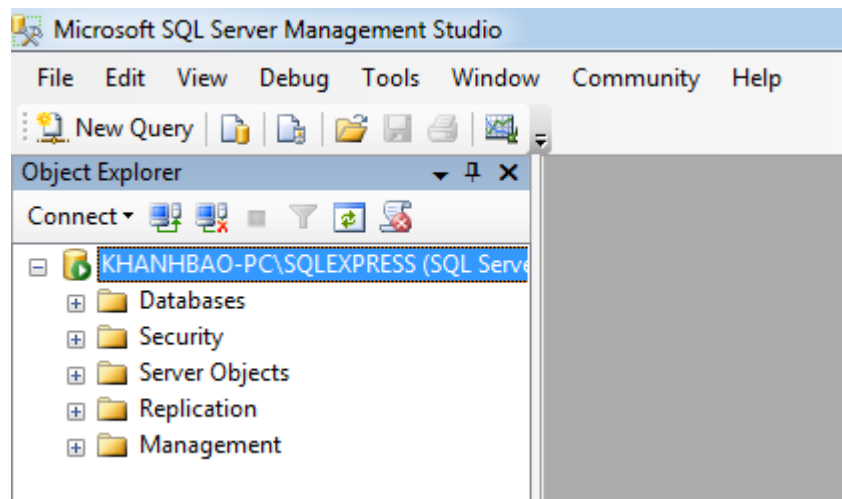
Lưu ý:

SQL Server có hai kiểu chứng thực người dùng (authentication):

- **Windows authentication mode:** Việc kiểm tra người dùng của SQL Server sẽ phụ thuộc vào kiểm tra người dùng của Windows. Khi người dùng có quyền đăng nhập vào Windows, người đó sẽ có quyền đăng nhập vào SQL Server. Kiểu kiểm tra người dùng này thường được sử dụng khi ứng dụng khai thác dữ liệu và SQL Server được cài chung trên một máy tính.

- **SQL Server authentication mode:** Việc kiểm tra người dùng của SQL Server sẽ không phụ thuộc vào việc kiểm tra người dùng của Windows. Khi người dùng có quyền đăng nhập vào Windows, người dùng đó chưa chắc có quyền đăng nhập vào SQL Server. Để đăng nhập vào SQL Server, người dùng này phải có một bộ username và password do SQL Server quản lý. Kiểu kiểm tra người dùng này thường được sử dụng khi ứng dụng khai thác dữ liệu và SQL Server không được cài trên cùng một máy.

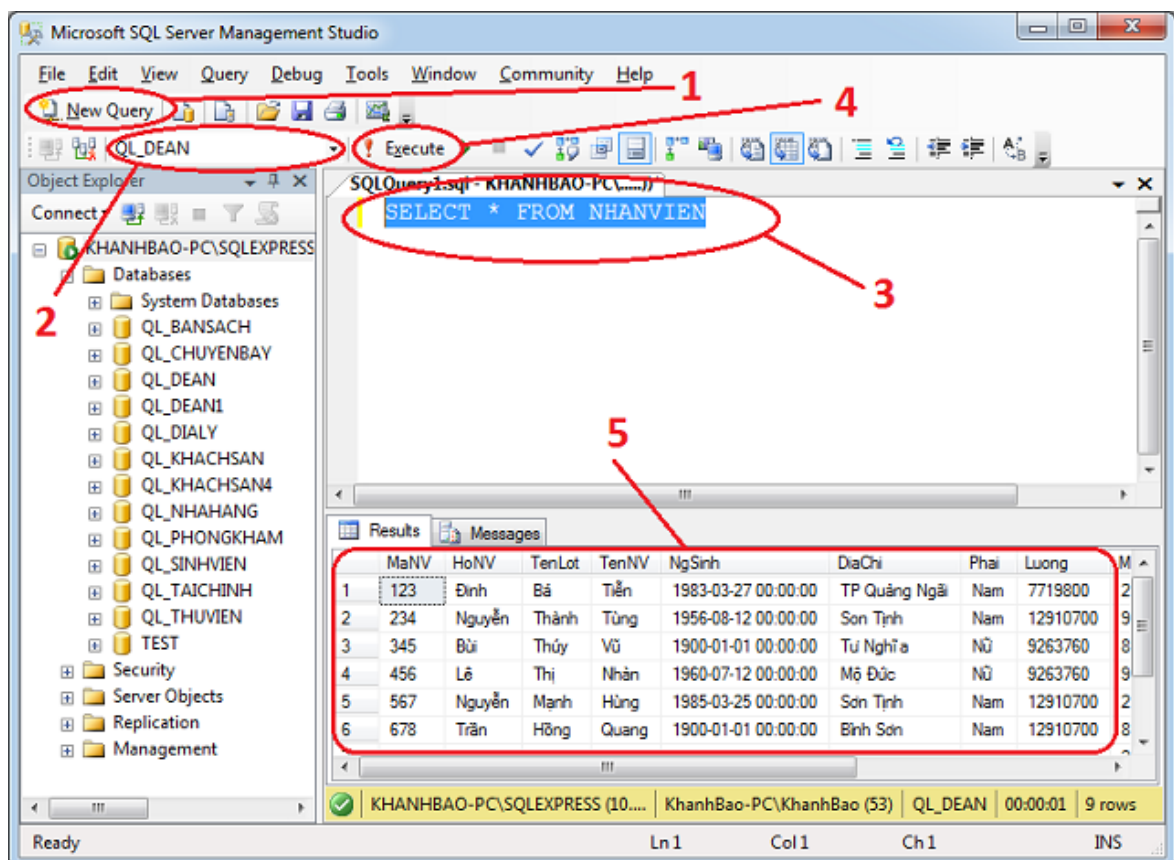
Sau khi kết nối thành công, giao diện công cụ SQL Management Studio như sau:



Hình 1.5. Giao diện SQL Server Management Studio

Cửa sổ Object Explorer là cửa sổ chính để người dùng tương tác với các cơ sở dữ liệu theo hình thức giao diện trực quan. Việc thực hiện các thao tác trên cửa sổ này sẽ được trình bày cụ thể ở các chương sau.

Để tương tác với các cơ sở dữ liệu thông qua câu lệnh SQL, có thể click vào biểu tượng New Query trên thanh công cụ, sau đó thực hiện các bước sau:



Hình 1.6. Truy vấn dữ liệu bằng câu lệnh SQL

- Chọn CSDL cần thao tác
- Soạn thảo câu lệnh SQL tại khung soạn thảo
- Chọn (bôi đen) câu lệnh cần thực thi
- Click vào nút Excute trên thanh công cụ
- Kết quả sẽ được hiển thị bên dưới khung soạn thảo

BÀI TẬP KẾT THÚC CHƯƠNG

LÝ THUYẾT

1. Anh/chị hiểu như thế nào về mô hình Client/Server? Cho một ví dụ về mô hình Client/Server mà anh/chị biết.
2. Hệ quản trị cơ sở dữ liệu là gì? Chức năng của nó. Nêu một số hệ quản trị cơ sở dữ liệu đang được sử dụng hiện nay?
3. Đối với hệ thống dữ liệu như thế nào thì nên dùng hệ quản trị CSDL SQL Server?

Chương 2: CÁC THAO TÁC VỚI CƠ SỞ DỮ LIỆU

Thời lượng: 02 tiết lý thuyết + 02 tiết thực hành

Kết thúc chương này, sinh viên có thể:

- ☞ *Hiểu một cách khái quát về khái niệm Cơ sở dữ liệu.*
- ☞ *Biết được các CSDL hệ thống trong SQL Server và chức năng của chúng.*
- ☞ *Biết được cách thức lưu trữ CSDL trong SQL Server*
- ☞ *Thực hiện được một số thao tác liên quan đến CSDL (tạo mới, attach, detach, xóa) trong SQL Server bằng hai cách.*

2.1. CƠ SỞ DỮ LIỆU TRONG SQL SERVER

2.1.1. Tổng quan

Một cơ sở dữ liệu của Microsoft SQL Server là một tập hợp các đối tượng: table, view, stored procedure... cho phép người sử dụng lưu trữ và khai thác các thông tin dữ liệu được tổ chức và lưu trữ bên trong đó.

Có hai loại cơ sở dữ liệu: CSDL hệ thống và CSDL do người dùng tự định nghĩa.

Một CSDL của Microsoft SQL Server chỉ do một người dùng tạo ra, tuy nhiên, nhiều người có thể truy cập vào CSDL này.

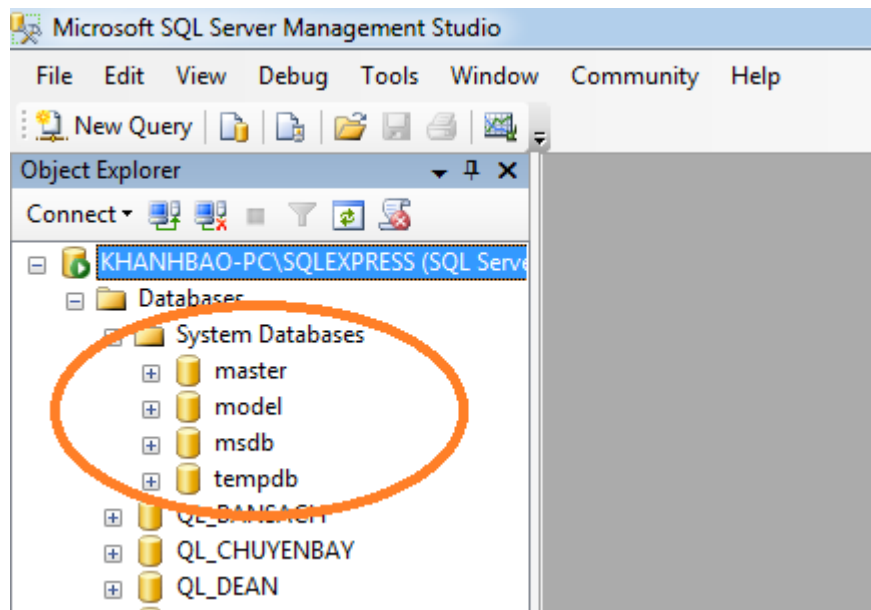
2.1.2. Các CSDL hệ thống

CSDL hệ thống (System Database) là CSDL được tạo ra trong quá trình cài đặt SQL Server, được dùng để lưu trữ các thông tin liên quan đến hoạt động của SQL Server.

SQL Server chứa 4 CSDL hệ thống, bao gồm:

- **Master:** Là cơ sở dữ liệu chính yếu dùng để chứa thông tin của các bảng hệ thống nhằm quản lý tất cả các cơ sở dữ liệu khác hiện có trong máy chủ Microsoft SQL Server.

- **Model:** Là cơ sở dữ liệu khuôn dạng mẫu (template) mà Microsoft SQL Server sử dụng để tạo lập ra các cơ sở dữ liệu mới. Trong đó, cơ sở dữ liệu mới sẽ chứa các đối tượng, quyền hạn hoàn toàn giống các đối tượng, quyền hạn hiện có trong cơ sở dữ liệu Model này.



Hình 2.1. Các CSDL hệ thống

- **Msdb:** Là cơ sở dữ liệu được dùng cho dịch vụ SQL Server Agent, dịch vụ này sẽ tự động thực hiện các hành động mà người quản trị cơ sở dữ liệu đã đưa ra trong thời gian biểu như sao chép an toàn dữ liệu (backup), đồng bộ hóa dữ liệu (replicate), ...

- **Tempdb:** Là cơ sở dữ liệu tạm thời (temporary database) mà Microsoft SQL Server sử dụng để chứa các bảng tạm do người dùng tạo ra một cách minh bạch hoặc là nơi lưu trữ các kết quả trung gian trong quá trình thực hiện các xử lý cho các truy vấn hoặc sắp xếp dữ liệu.

2.1.3. CSDL do người dùng tự định nghĩa

Là CSDL do người dùng tạo ra hoặc import vào SQL Server nhằm mục đích quản lý, khai thác và chia sẻ dữ liệu.

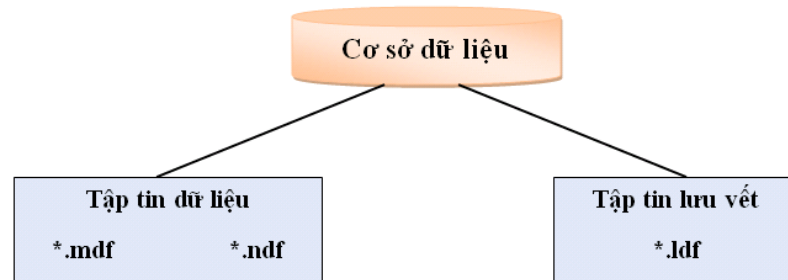
Nội dung các phần sau tập trung giới thiệu các thao tác liên quan đến CSDL do người dùng tự định nghĩa.

Cách thức lưu trữ CSDL của SQL Server:

Mặc dù phải quản lý nhiều đối tượng bên trong cơ sở dữ liệu nhưng Microsoft SQL Server chỉ tổ chức một vài tập tin dữ liệu (data files) vật lý để lưu trữ, đó là điều cải tiến từ phiên bản 7.0 trở lên của Microsoft SQL Server.

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)



Hình 2.2. Các tập tin dữ liệu trong SQL Server

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.

2.2. TẠO CƠ SỞ DỮ LIỆU

2.2.1. Các thuộc tính của một cơ sở dữ liệu trong Microsoft SQL Server

- **Tên CSDL (Database Name):** mỗi CSDL trong SQL Server có một tên duy nhất, có độ dài tối đa là 123 ký tự. Tên cơ sở dữ liệu nên được đặt bằng những từ gợi nhớ như QL_BANHANG (Quản lý bán hàng), QL_HOCSINH (Quản lý học sinh) ...

- **Vị trí tập tin (File Location):** Là đường dẫn vật lý của các loại tập tin dữ liệu dùng để lưu trữ cơ sở dữ liệu của Microsoft SQL Server.

- **Tên tập tin (File Name):** Là tên luận lý của mỗi loại tập tin dữ liệu tương ứng mà hệ thống Microsoft SQL Server dùng để quản lý bên trong. Tương ứng mỗi loại tập tin dữ liệu sẽ có một tên tập tin riêng biệt.

- **Kích thước ban đầu (Initial size):** Là kích thước khởi tạo của tập tin dữ liệu khi cơ sở dữ liệu mới được tạo lập. Đơn vị tính là megabyte (MB). Thông thường kích thước ban đầu của một cơ sở dữ liệu mới tối thiểu phải bằng kích thước của cơ sở dữ liệu Model, bởi vì Microsoft SQL Server sẽ lấy cơ sở dữ liệu Model làm khuôn dạng mẫu khi tạo lập một cơ sở dữ liệu mới.

- **Việc tăng trưởng kích thước tập tin dữ liệu (File growth):** Là các quy định cho việc tăng trưởng tự động kích thước tập tin dữ liệu, bởi vì các dữ liệu sẽ được người dùng nạp vào để lưu trữ ngày càng nhiều hơn so với kích thước ban đầu khi tạo lập. Việc tăng trưởng sẽ tự động làm tăng kích thước tập tin dữ liệu liên quan theo từng MB (in megabytes) hoặc theo tỷ lệ phần trăm (by percent) của kích thước tập tin hiện hành khi các dữ liệu bên trong Microsoft SQL Server lưu trữ gần đầy so với kích thước tập tin vật lý hiện thời. Mặc định kích thước tập tin dữ liệu sẽ được tăng tự động 10% khi dữ liệu lưu trữ bị đầy.

- **Kích thước tối đa tập tin dữ liệu (Maximum file size):** Là việc quy định sự tăng trưởng tự động kích thước của các tập tin dữ liệu nhưng có giới hạn (restrict file growth) đến một số MB nào đó hoặc là không có giới hạn (unrestrict file growth). Trong trường hợp nếu người dùng chọn có giới hạn kích thước của tập tin dữ liệu, dữ liệu sẽ được thêm vào các tập tin dữ liệu mới khi dữ liệu lưu trữ đã bằng với kích thước tối đa của tập tin dữ liệu. Các tập tin dữ liệu mới này chính là loại tập tin thứ yếu (Secondary data file) và chúng có thể được lưu trữ tại các ổ cứng khác có bên trong Microsoft SQL Server. Đây cũng là một nét đặc trưng của mô hình cơ sở dữ liệu phân tán (distributed database) mà Microsoft SQL Server hỗ trợ.

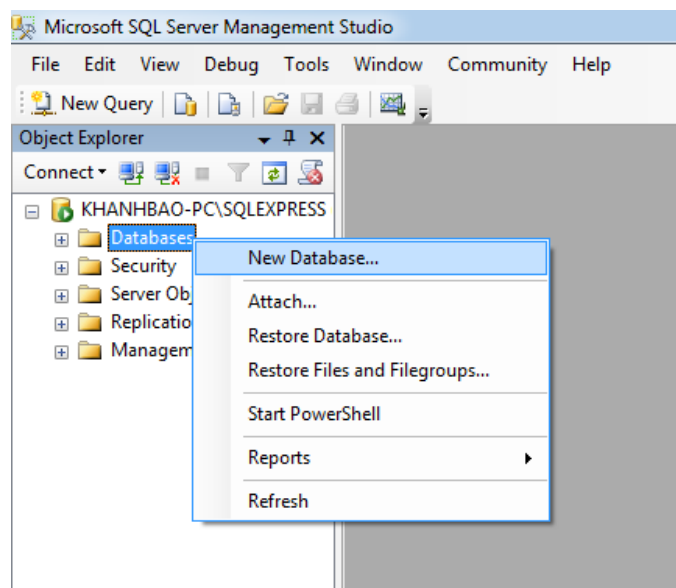
2.2.2. Tạo CSDL trong SQL Server

Trong SQL Server, hầu hết các thao tác với CSDL đều có thể được thực hiện bằng hai cách: sử dụng giao diện trực quan bằng tiện ích SQL Server Management Studio hoặc sử dụng câu lệnh T-SQL. Để tạo CSDL, người dùng cũng có thể sử dụng hai cách nói trên.

a. Tạo CSDL sử dụng tiện ích SQL Server Management Studio

Khởi động tiện ích SQL Server Management Studio, kết nối đến một instance của SQL Server Database Engine, click vào biểu tượng dấu (+) phía trước instance này để mở rộng chúng. Quá trình tạo CSDL được thực hiện như sau:

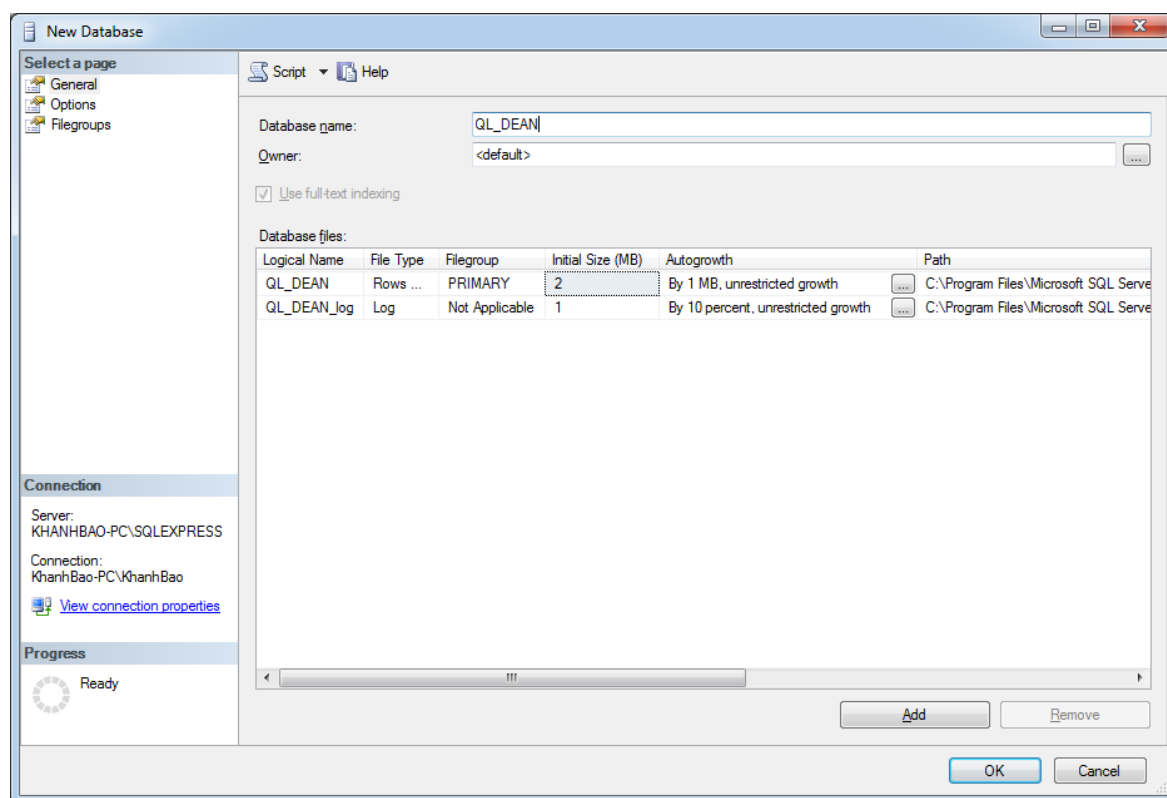
Bước 1: Click chuột phải trên đối tượng **Databases**, chọn **New Database...**



Hình 2.3. Tạo CSDL bằng tiện ích SQL Server Management Studio

Bước 2: Trong hộp thoại New Database, nhập tên CSDL vào mục **Database name**

Bước 3: Nhấn OK nếu muốn tạo CSDL bằng các giá trị mặc định. Nếu không, tiếp tục bước nhập các thông số liên quan đến các tập tin dữ liệu vào mục **Database files**. Bấm vào nút Add để tạo tập tin dữ liệu thứ yếu (*.ndf). Nhấn OK để tiến hành khởi tạo CSDL.



Hình 2.4. Cửa sổ New Database

b. Tạo CSDL bằng cách sử dụng câu lệnh T-SQL

Lưu ý: Các câu lệnh T-SQL và cách soạn thảo, thực thi câu lệnh T-SQL sẽ được trình bày chi tiết tại chương 5 của bài giảng này.

Việc tạo mới một cơ sở dữ liệu còn có thể được thực hiện bằng câu lệnh **CREATE DATABASE**. Các bước thực hiện như sau:

Bước 1: Kết nối đến **Database Engine**

Bước 2: 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:

```

CREATE DATABASE QL_DEAN
ON PRIMARY
( NAME = QLDeAN,
  FILENAME = 'D:\HocSQL\QLDeAn.mdf',
  SIZE = 10MB,
  MAXSIZE = 50MB,
  FILEGROWTH = 5% )
LOG ON
( NAME = QLDeAn_log,
  FILENAME = 'D:\HocSQL\QLDeAn_log.ldf',
  SIZE = 5MB,
  MAXSIZE = 25MB,
  FILEGROWTH = 5MB ) ;
GO

```

Bước 3: Nhấn vào nút Excute để thực thi

Lưu ý: Sinh viên có thể tham khảo hướng dẫn của Microsoft về việc tạo CSDL tại đường dẫn sau:

[https://msdn.microsoft.com/en-us/library/ms186312\(v=sql.110\).aspx](https://msdn.microsoft.com/en-us/library/ms186312(v=sql.110).aspx)

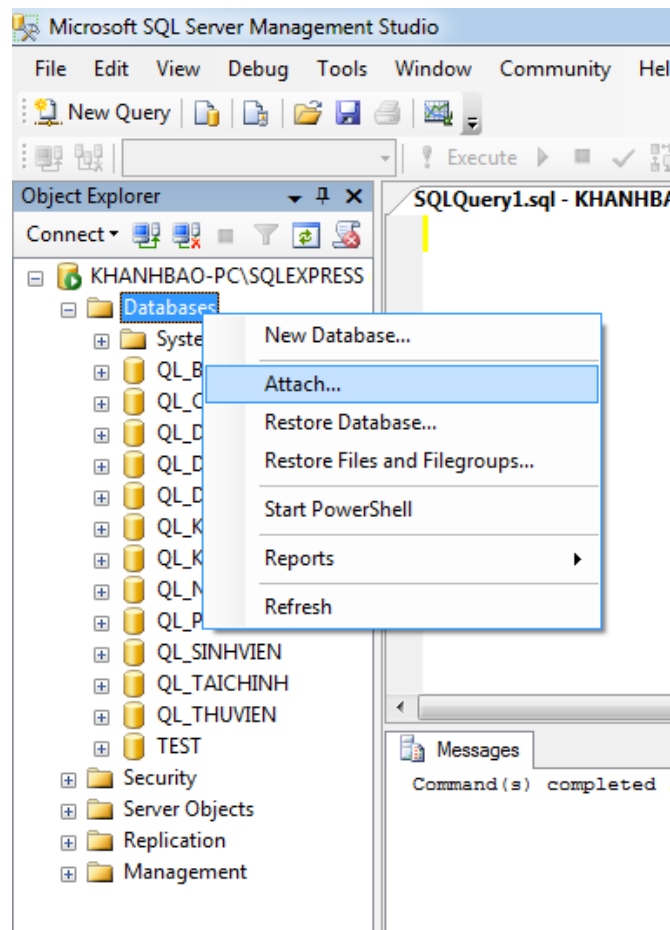
2.3. GẮN (ATTACH) CƠ SỞ DỮ LIỆU VÀO SQL SERVER

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.

a. 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



Hình 2.5. Attach Database bằng tiện ích SQL Server Management Studio

b. 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

Lưu ý: Sinh viên có thể tham khảo hướng dẫn của Microsoft về việc attach CSDL tại đường dẫn sau:

[https://msdn.microsoft.com/en-us/library/ms190209\(v=sql.110\).aspx](https://msdn.microsoft.com/en-us/library/ms190209(v=sql.110).aspx)

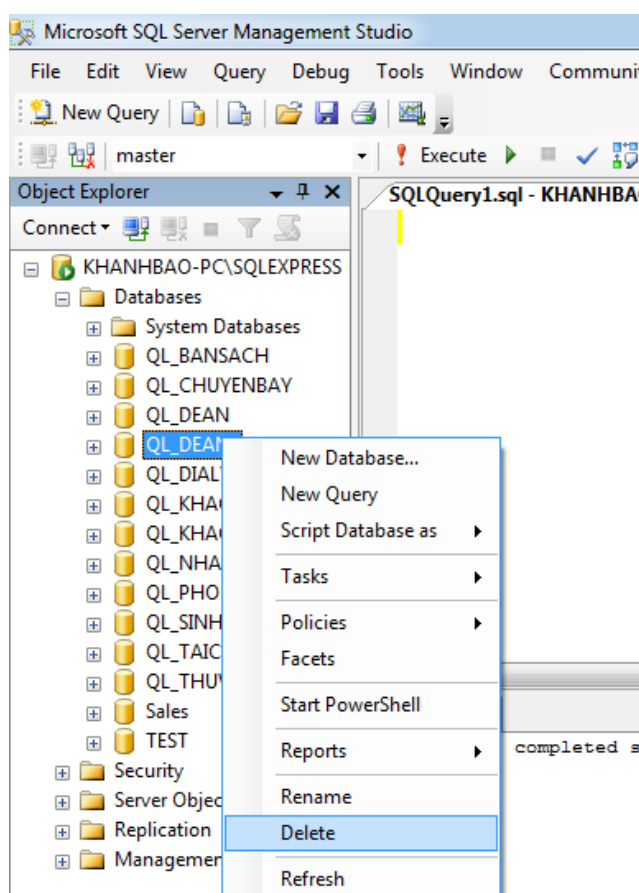
2.4. XÓA CƠ SỞ DỮ LIỆU

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.

a. 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



Hình 2.6. Xóa Database bằng tiện ích SQL Server Management Studio

b. 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

Lưu ý: Sinh viên có thể tham khảo hướng dẫn của Microsoft về việc xóa CSDL tại đường dẫn sau:

[https://msdn.microsoft.com/en-us/library/ms177419\(v=sql.110\).aspx](https://msdn.microsoft.com/en-us/library/ms177419(v=sql.110).aspx)

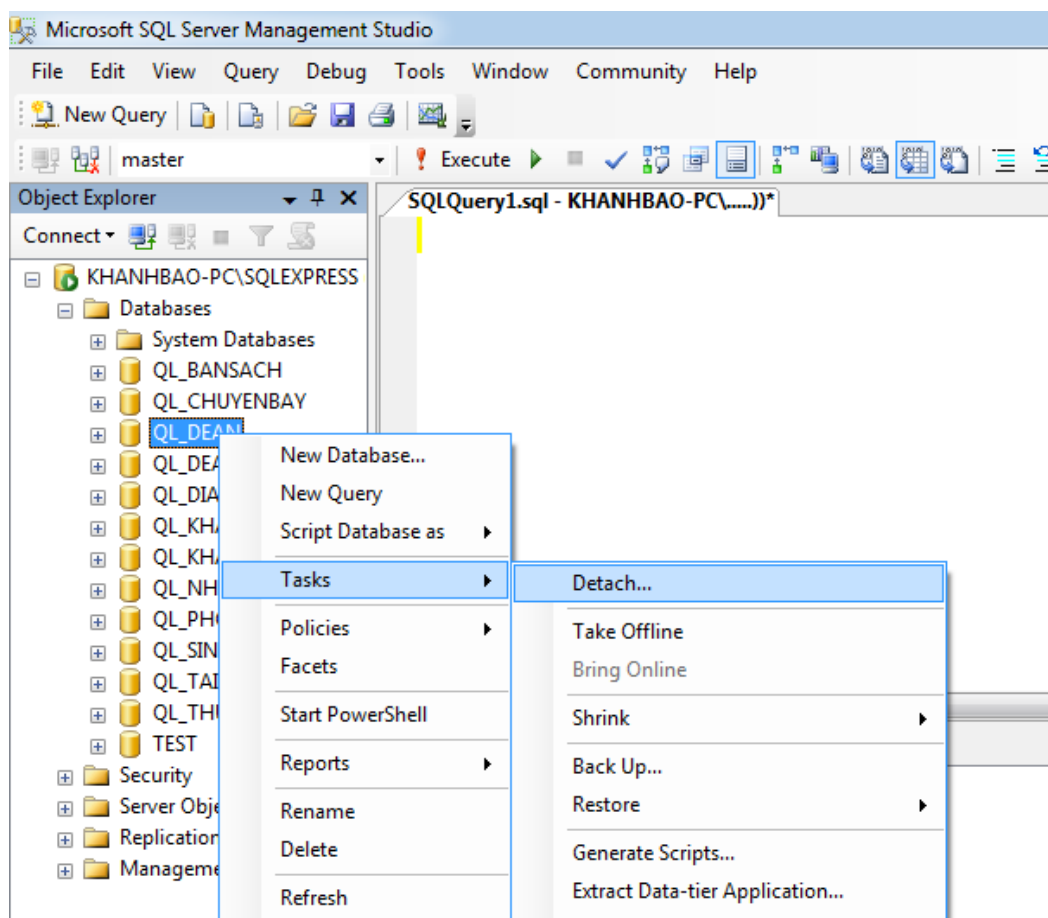
2.5. GỠ (DETACH) CƠ SỞ DỮ LIỆU KHỎI SQL SERVER

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.

a. 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.



Hình 3.7. Detach Database bằng tiện ích SQL Server Management Studio

b. 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

Lưu ý: Sinh viên có thể tham khảo hướng dẫn của Microsoft về việc detach CSDL tại đường dẫn sau:

[https://msdn.microsoft.com/en-us/library/ms191491\(v=sql.110\).aspx](https://msdn.microsoft.com/en-us/library/ms191491(v=sql.110).aspx)

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 MB

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 3: TẠO VÀ QUẢN LÝ TABLE (BẢNG)

Thời lượng: 04 tiết lý thuyết + 02 tiết thực hành

Kết thúc chương này, sinh viên có thể:

- ☞ *Hiểu được chức năng của bảng trong cơ sở dữ liệu*
- ☞ *Biết được cấu trúc của bảng và các thuộc tính liên quan*
- ☞ *Thực hiện các thao tác tạo và quản lý bảng bằng hai cách*

3.1. KHÁI NIỆM TABLE (BẢNG)

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...

3.1.1. Cấu trúc Table

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.

3.1.2. 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

MSSV	HoDem	Ten	NgaySinh	GioiTinh	Lop
SV01	Cao	Minh	3/2/1994	Nam	L1
SV02	Võ Tấn	Hùng	30/4/1995	Nam	L2
SV03	Cao Thanh	Loan	19/8/1995	Nữ	L2
SV04	Hồ	Cường	2/9/1996	Nam	L1

Table LOP

MaLop	TenLop	Khoa
L1	Lớp thứ 1	16
L2	Lớp thứ 2	16

3.2. CÁC KIỂU DỮ LIỆU TRONG SQL SERVER

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.

3.2.1. Kiểu dữ liệu số chính xác (Exact Numerics)

Bảng 4.1. Các kiểu dữ liệu số chính xác

Kiểu dữ liệu	Kích thước	Miền giá trị lưu trữ dữ liệu
bit	1 bit	Các giá trị 0, 1 hoặc NULL
tinyint	1 byte	Các số nguyên từ 0 đến 255
smallint	2 byte	Các số nguyên từ -32.768 đến 32.767

int	4 byte	Các số nguyên từ -2.147.483.648 đến 2.147.483.648
bigint	8 byte	Các số nguyên từ -2^{63} đến $2^{63} - 1$
decimal(p,s), numeric(p,s)	17 byte	Các số thập phân từ -10^{38} đến 10^{38} <i>p: số lượng tối đa tất cả các chữ số</i> <i>s: số lượng tối đa các chữ số phần thập phân</i>
smallmoney	4 byte	Các số dạng tiền tệ từ - 214748,3648 đến 214748,3647
money	8 byte	Các số dạng tiền tệ từ -922.337.203.685.477,5808 đến 922.337.203.685.477,5807

3.2.2. Kiểu dữ liệu số xấp xỉ (Approximate Numerics)

Bảng 4.2. Các kiểu dữ liệu số xấp xỉ

Kiểu dữ liệu	Kích thước	Miền giá trị lưu trữ dữ liệu
real	4 byte	Các số thực từ $-1.79E + 308$ đến $1.79E + 308$
float(n)		Các số thực từ $-1.79E + 308$ đến $1.79E + 308$ (tùy vào n) <i>Nếu $n = [1, ..., 24]$: kích thước 4 byte</i> <i>Nếu $n = [25, ..., 53]$: kích thước 8 byte</i> <i>Giá trị mặc định là 53</i>

3.2.3. Kiểu dữ liệu ngày giờ (Date and Time)

Bảng 4.3. Các kiểu dữ liệu ngày giờ

Kiểu dữ liệu	Kích thước	Miền giá trị lưu trữ dữ liệu
date	3 byte	Các giá trị ngày từ 0001/01/01 đến 9999/12/31
time	5 byte	Các giá trị giờ từ 00:00:00.000 đến 23:59:59.999
smalldatetime	4 byte	Các giá trị ngày giờ từ 1900/01/01 00:00:00 đến 2079/06/06 23:59:59

3.2.4. Kiểu dữ liệu chuỗi ký tự

Bảng 4.4. Các kiểu dữ liệu chuỗi ký tự

Kiểu dữ liệu	Kích thước	Miền giá trị lưu trữ dữ liệu
char(n)	n byte	Chuỗi ký tự có n ký tự
varchar(n)	n byte	Chuỗi ký tự có tối đa n ký tự

nchar(n)	2*n byte	Chuỗi ký tự Unicode có n ký tự
nvarchar(n)	2*n byte	Chuỗi ký tự Unicode có tối đa n ký tự

Ngoài ra còn có một số kiểu dữ liệu cơ sở khác. Sinh viên có thể tham khảo chi tiết tại website hướng dẫn của Microsoft tại đường dẫn sau:

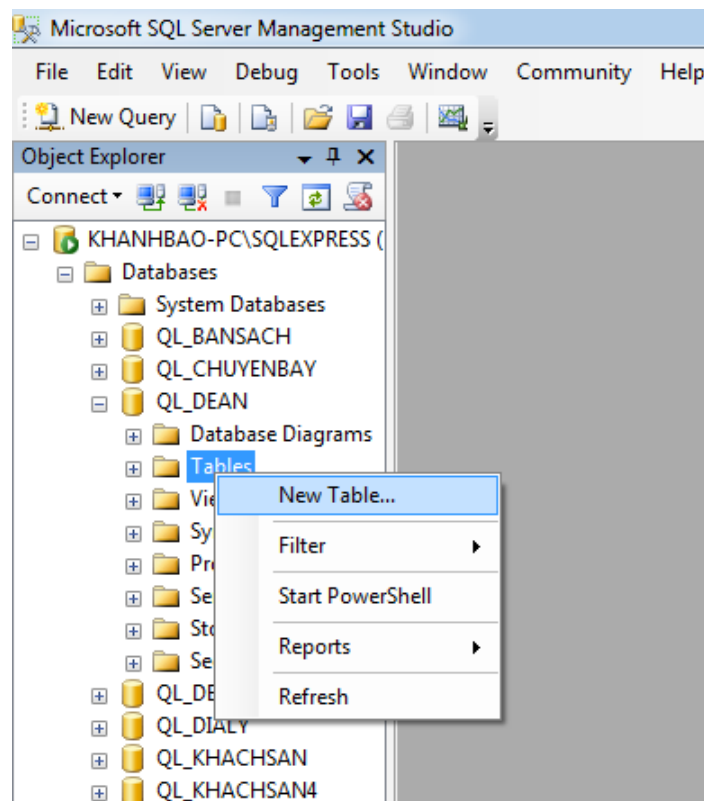
<https://msdn.microsoft.com/en-us/library/ms187752.aspx>

Lưu ý: Khi tiến hành tạo bảng, cần phải xác định kiểu dữ liệu của các cột dựa trên các dữ liệu dự định lưu trữ vào bảng.

3.3. TẠO TABLE

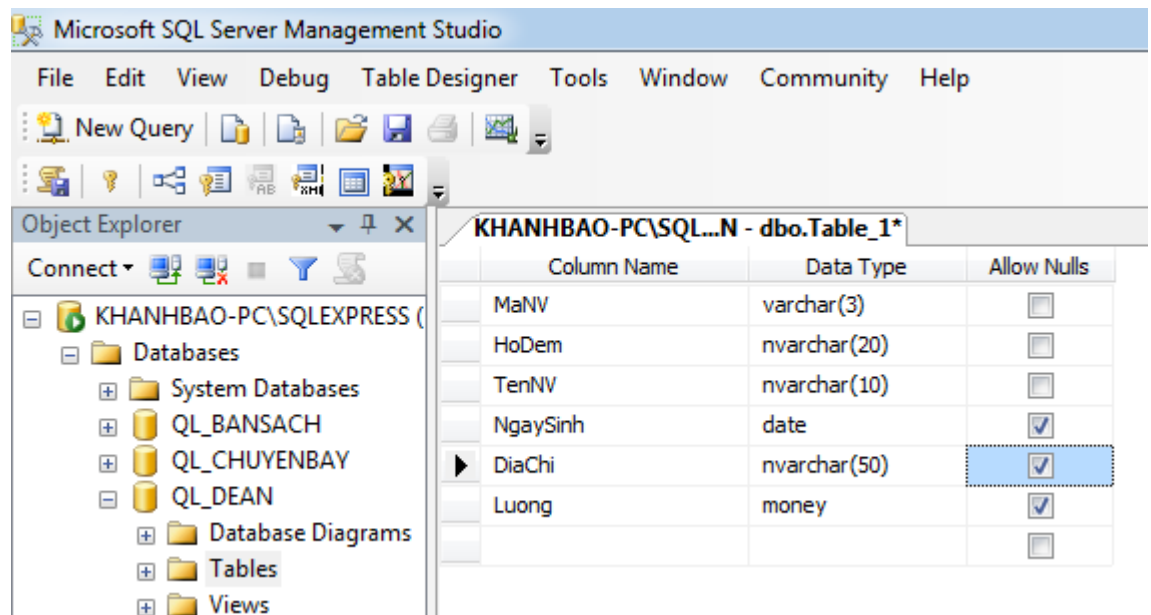
a. Tạo bảng bằng tiện ích SQL Server Management Studio

Bước 1: Trong cửa sổ Object Explorer, click vào biểu tượng dấu (+) phía trước tên CSDL cần thao tác, sau đó R-click vào thư mục **Tables**, chọn **New Table...**



Hình 4.1. Tạo Database bằng tiện ích SQL Server Management Studio

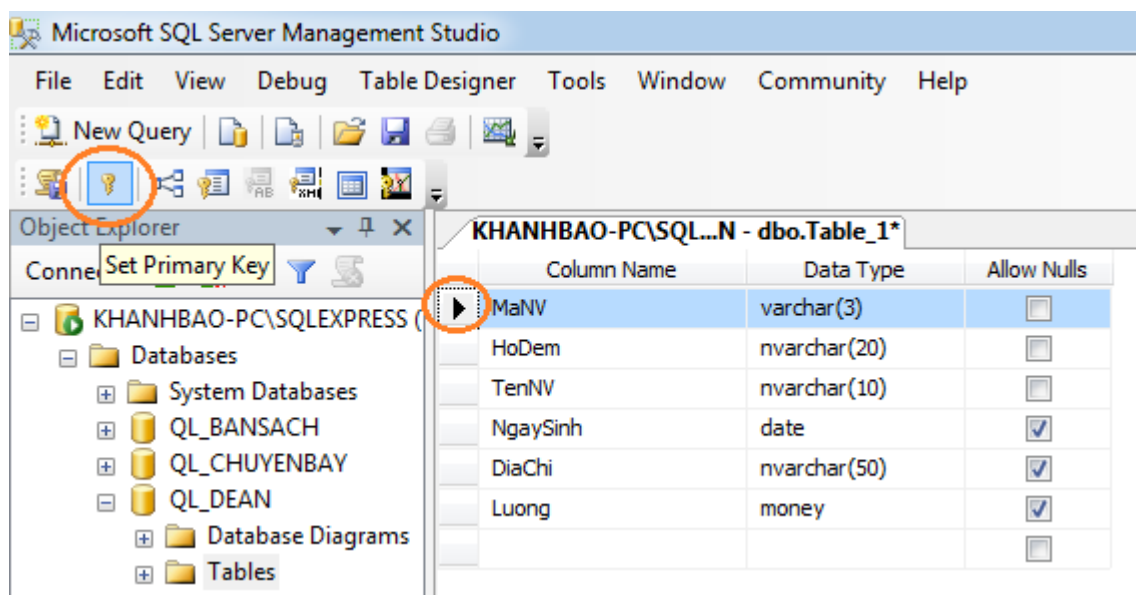
Bước 2: Nhập tên các cột và chọn kiểu dữ liệu tương ứng cho các cột. Click chọn vào ô **Allow Nulls** nếu giá trị dữ liệu của cột đó có thể được cho phép để trống.



Hình 3.2. Giao diện tạo các cột của bảng

Bước 3: Thiết lập khóa chính.

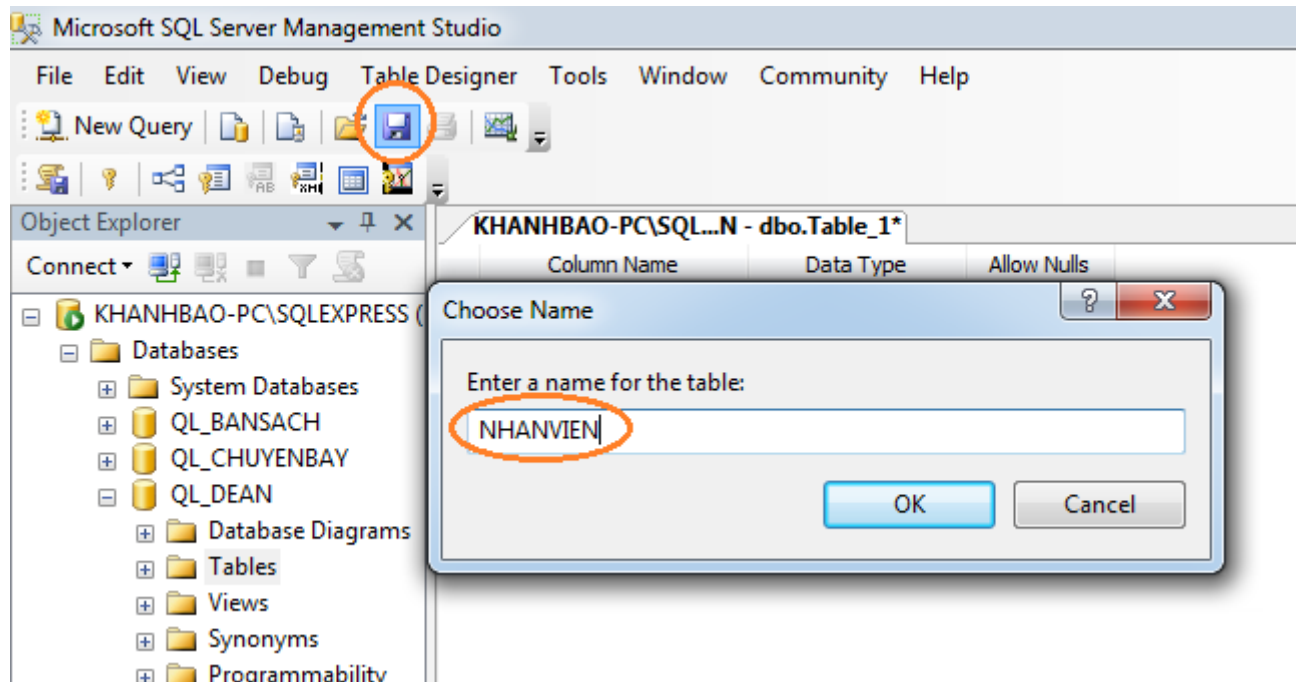
Click vào cột hoặc nhóm các cột được chọn làm khóa chính, bấm vào biểu tượng Set Primary Key trên thanh công cụ.



Hình 4.3. Thiết lập khóa chính cho bảng

Bước 4: Lưu bảng

Click vào biểu tượng **Save Table** trên thanh công cụ và nhập tên bảng vào hộp thoại **Choose Name**.



Hình 4.4. Lưu bảng

Một số lưu ý khi tạo bảng:

- Tên bảng, tên cột là chuỗi ký tự không dấu và không có dấu cách.
- Không thể thực hiện tạo bảng nếu đã tồn tại một bảng cùng tên trong CSDL.
- Không được thiết lập thuộc tính Allow Null cho cột được chọn làm khóa chính.

b. Tạo bảng bằng câu lệnh T-SQL

Có thể thực hiện tạo bảng bằng cách viết và thực thi câu lệnh T-SQL trong cửa sổ soạn thảo câu lệnh.

Cú pháp câu 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 date,  
    DiaChi nvarchar(150),  
    Phai nvarchar(3),  
    Luong numeric(18,0),  
    MaNQL varchar(9),  
    Phong varchar(2)  
)
```

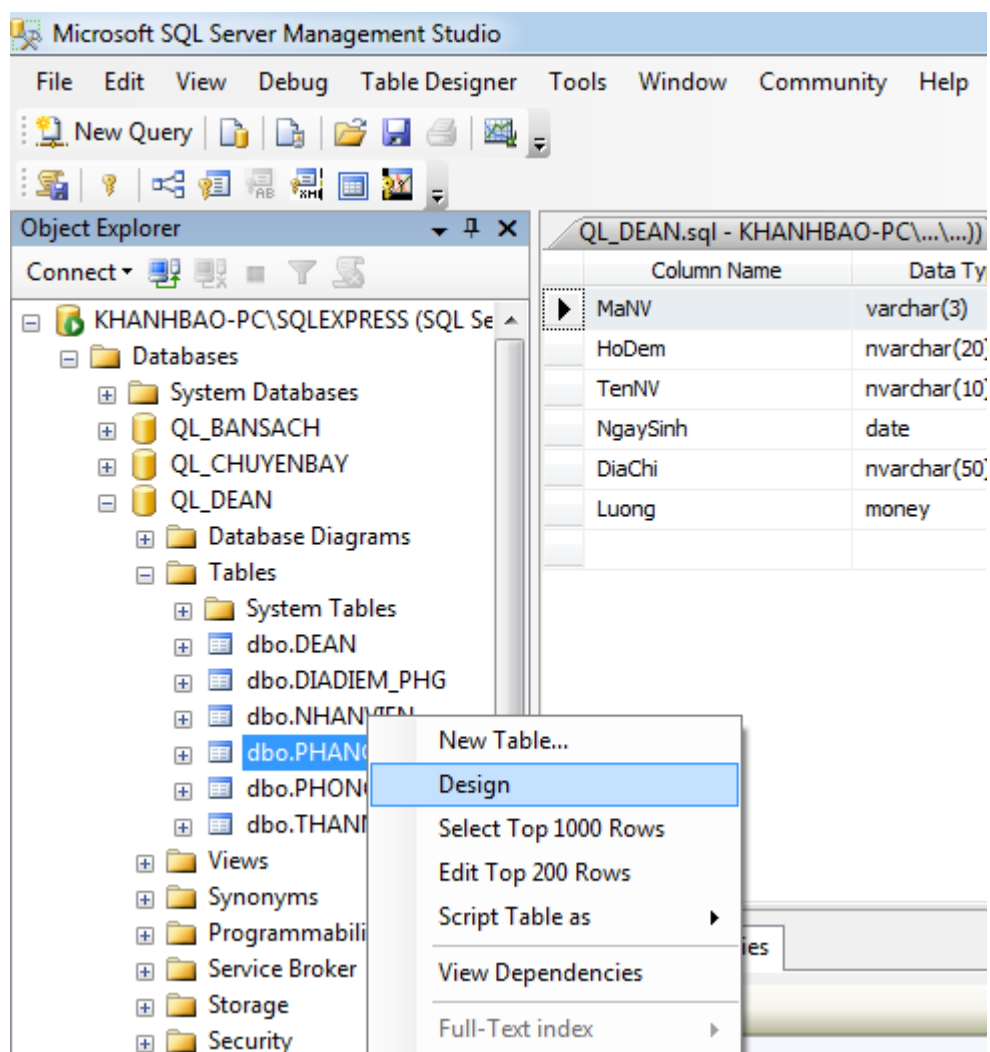

3.4. CHỈNH SỬA CẤU TRÚC TABLE

Việc chỉnh sửa cấu trúc bảng bao gồm một số thao tác sau:

- Thêm, xóa cột
- Thay đổi tên cột
- Thay đổi kiểu dữ liệu của cột
- Tạo, thay đổi khóa chính, khóa ngoại

a. Chỉnh sửa cấu trúc bảng bằng tiện ích SQL Server Management Studio

Để chỉnh sửa cấu trúc bảng, R_click vào tên bảng cần chỉnh sửa, sau đó chọn Design. Các thao tác còn lại làm tương tự như tạo bảng.



Hình 4.5. Chỉnh sửa cấu trúc bảng

b. Chỉnh sửa cấu trúc bảng bằng câu lệnh T-SQL

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

a) Thêm cột

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

b) Xóa cột

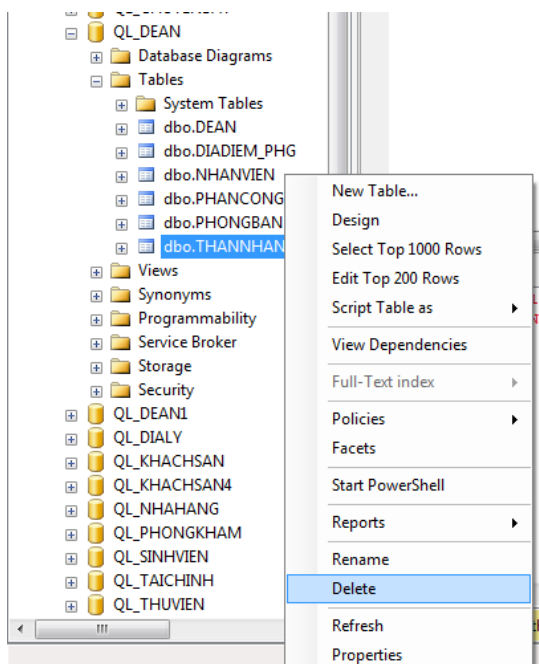
```
ALTER TABLE TableName  
DROP COLUMN ColumnName
```

c) Thay đổi kiểu dữ liệu của cột

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

3.5. XÓA TABLE

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.



Hình 4.6. 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
```

3.6. TẠO RÀNG BUỘC DỮ LIỆU TRONG TABLE

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.

Nội dung bài giảng sẽ trình bày việc tạo các Constraint bằng câu lệnh T-SQL.

a) 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.

b) 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 Luong (Lương) được tạo như sau:

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

c) 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
```

d) 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)
```

e) Kiểm tra ràng buộc toàn vẹn khóa ngoại

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)
```

Một số thao tác với Constraint

- Tạm thời vô hiệu Constraint

```
ALTER TABLE TableName
NOCHECK CONSTRAINT ConstraintName
```

Nếu muốn vô hiệu tất cả các constraint trong bảng, thay thế tên ConstraintName bằng từ khóa ALL.

- Kích hoạt lại Constraint

```
ALTER TABLE TableName  
CHECK CONSTRAINT ConstraintName
```

Nếu muốn kích hoạt tất cả các constraint trong bảng, thay thế tên ConstraintName bằng từ khóa ALL.

Hủy bỏ Constraint

```
ALTER TABLE TableName  
DROP CONSTRAINT ConstraintName
```

BÀI TẬP KẾT THÚC CHƯƠNG

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
NgayNhanChuc	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”.

Chương 4: BIỂU ĐỒ CƠ SỞ DỮ LIỆU (DATABASE DIAGRAM)

Thời lượng: 02 tiết lý thuyết + 02 tiết thực hành

Kết thúc chương này, sinh viên có thể:

- ☞ *Hiểu khái niệm Database Diagram*
- ☞ *Tạo và quản lý các Database Diagram*

4.1. KHÁI NIỆM BIỂU ĐỒ CƠ SỞ DỮ LIỆU

Thông thường, các Table (bảng) trong cơ sở dữ liệu có mối quan hệ với nhau. Việc tạo mối quan hệ giữa các bảng nhằm trao đổi thông tin giữa chúng, đồng thời giúp kiểm tra tính tồn tại của dữ liệu (khóa ngoại).

Thực tế, trong quá trình tạo bảng, người sử dụng cũng có thể tạo ra mối quan hệ dữ liệu ngầm định giữa các bảng thông qua quy tắc kiểm tra ràng buộc khóa ngoại (foreign key constraint). Tuy nhiên, các mối quan hệ này không thể hiện trực quan giúp người sử dụng dễ dàng quan sát.

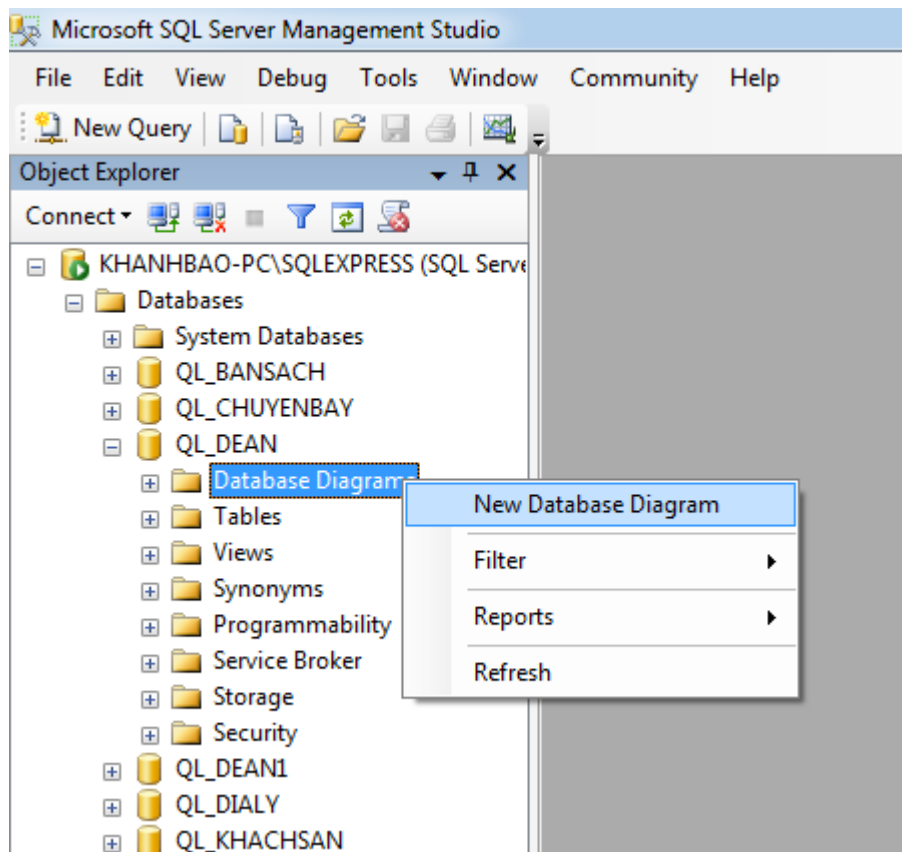
Biểu đồ cơ sở dữ liệu (Database Diagram) là một đối tượng trong CSDL giúp người sử dụng có thể dễ dàng nhìn thấy mối quan hệ dữ liệu giữa các bảng có liên quan với nhau, thực hiện một số hành động tác động trực tiếp ngay trong các bảng như: tạo cấu trúc bảng mới, xóa cấu trúc bảng đã có, chỉ định khóa chính cho bảng, thêm các bảng hiện có vào bên trong mô hình quan hệ dữ liệu.

4.2. TẠO BIỂU ĐỒ CƠ SỞ DỮ LIỆU

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

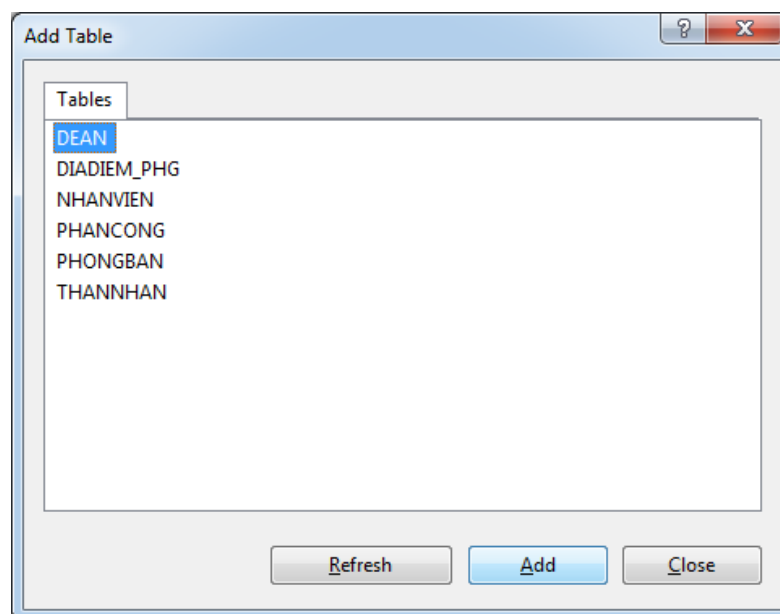
Các bước tạo mới biểu đồ cơ sở dữ liệu bên trong một cơ sở dữ liệu:

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**:



Hình 4.1. Tạo biểu đồ CSDL

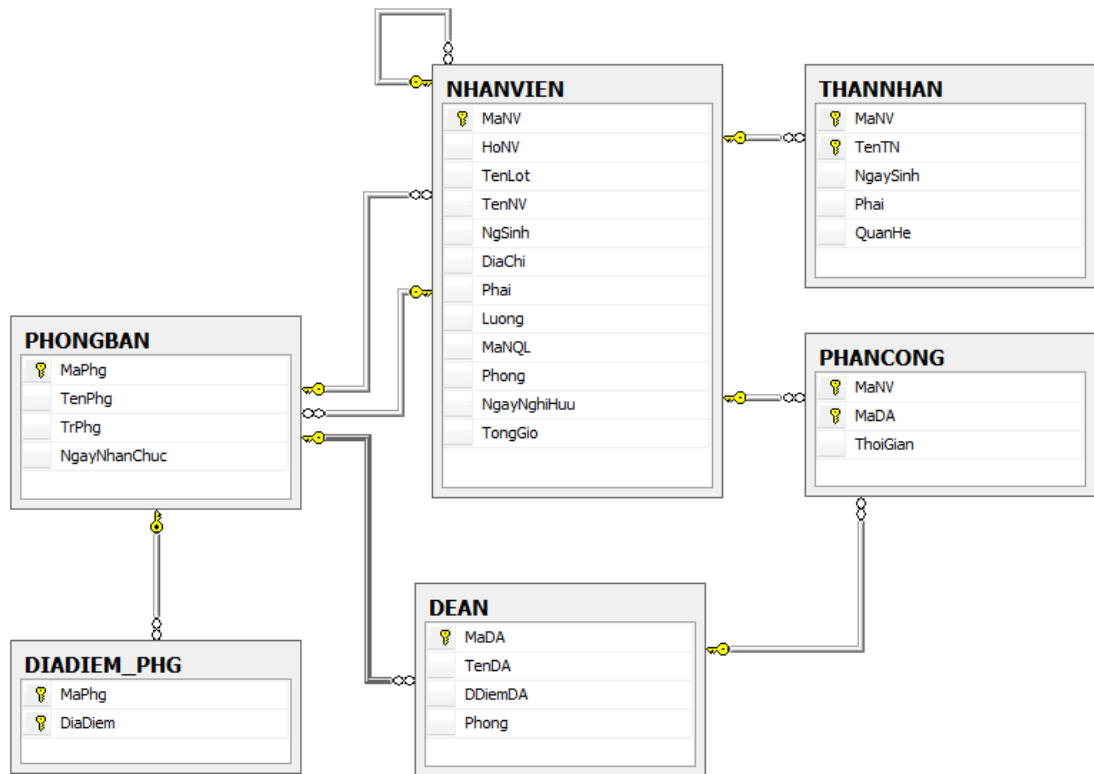
Bước 2: Trong cửa sổ **Add Table**, chọn các bảng cần đưa vào mô hình quan hệ dữ liệu, sau đó nhấn nút **Add**.



Hình 4.2. Cửa sổ Add Table

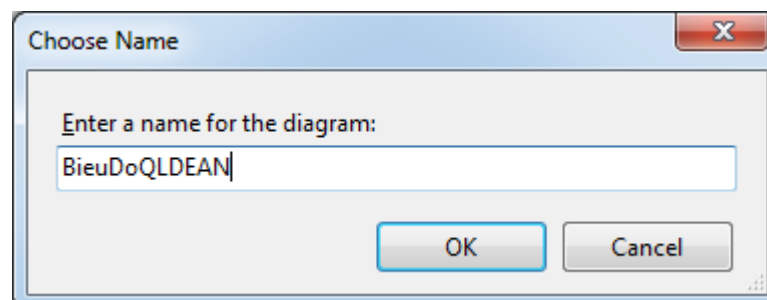
Bước 3: Sau khi chọn đủ các bảng, nhấn nút Close để đóng cửa sổ Add Table.

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.



Hình 4.3. Mô hình CSDL được tạo ra

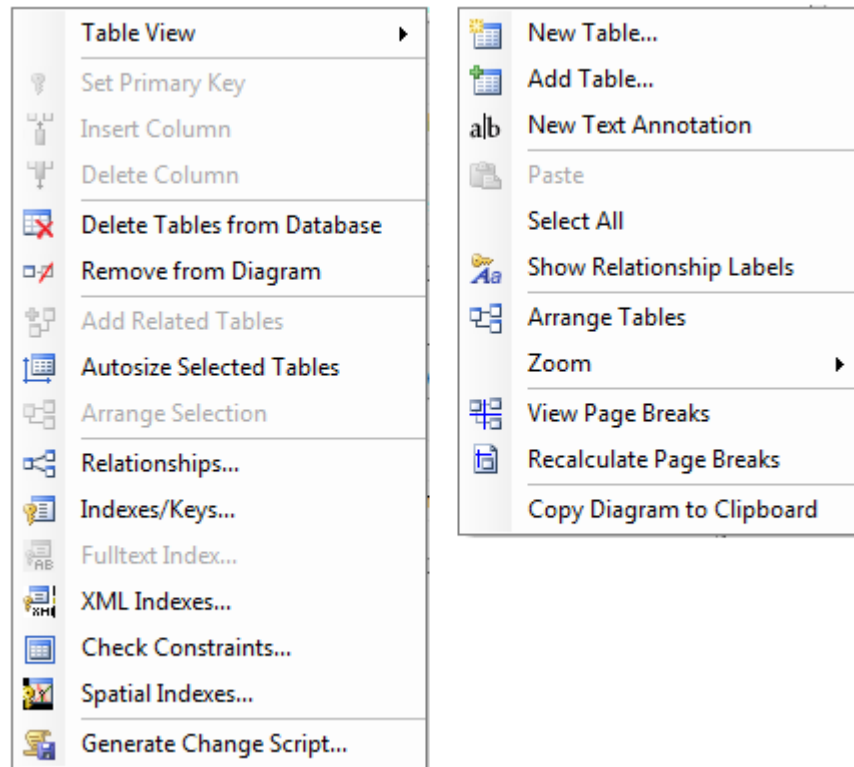
Bước 4: Nhấn nút Save trên thanh công cụ, nhập tên của mô hình để lưu lại mô hình quan hệ dữ liệu vừa tạo.



Hình 5.3. Hộp thoại lưu biểu đồ CSDL

4.3. QUẢN LÝ CÁC ĐỐI TƯỢNG TRONG BIỂU ĐỒ

Trong biểu đồ cơ sở dữ liệu có rất nhiều chức năng để người sử dụng có thể thực hiện khi đang làm việc bên trong mô hình này. Các chức năng sẽ thực hiện thông qua thực đơn tắt khi người dùng R-click lên tên bảng hoặc trên một vùng trống trong mô hình quan hệ dữ liệu.



Hình 5.4. Thực đơn tắt trên biểu đồ CSDL

4.3.1. Sửa đổi cấu trúc bảng hiện có

Để thực hiện việc sửa đổi cấu trúc bảng, R-click vào bảng cần chỉnh sửa, các lệnh liên quan đến việc chỉnh sửa cấu trúc bảng sẽ nằm trong menu tắt này:

- Insert Column: thêm cột vào bảng
- Delete column: xóa cột khỏi bảng
- Set Primary Key: định nghĩa khóa chính

Trong trường hợp thêm vào các cột mới, để có thể chỉ định được các thuộc tính liên quan đến các cột mới như là tên cột, kiểu dữ liệu, độ rộng, dữ liệu tại cột có cho phép dữ liệu bỏ trống..., người dùng nên chọn chức năng Properties để có

thể thấy được đầy đủ các thuộc tính liên quan đến các cột dữ liệu giống như màn hình thiết kế bảng trong tiện ích SQL Server Management Studio. Mặc định trong mô hình quan hệ dữ liệu chỉ hiển thị ra tên các cột có bên trong bảng.

4.3.2. Tạo mới, hủy bỏ bảng trong CSDL

Để tạo mới bảng trong CSDL, chọn chức năng **New Table**. Bảng mới được tạo sẽ tự động được chèn vào trong mô hình quan hệ dữ liệu

Để hủy bỏ bảng khỏi CSDL, bấm chọn bảng cần hủy bỏ, sau đó nhấn **Delete Tables From Database**.

Lưu ý: Sau khi đồng ý hủy bỏ bảng khỏi CSDL, dữ liệu và cấu trúc của bảng sẽ không được lưu trữ bên trong CSDL nữa.

4.3.3. Chèn, xóa bảng trong mô hình CSDL

Để chọn các bảng đã tồn tại trong CSDL và chèn chúng vào mô hình quan hệ, chọn chức năng **Add Table**, chọn các bảng trong hộp thoại **Add Table** và chèn chúng vào mô hình CSDL.

Chức năng **Add Related Tables** cho phép SQL Server tự động chèn các bảng có quan hệ với các bảng hiện hành đang chọn vào bên trong mô hình quan hệ dữ liệu.

Để xóa bảng được chọn ra khỏi mô hình quan hệ dữ liệu, chọn chức năng **Remove From Diagram**. Khác với chức năng **Delete Tables From Database**, cấu trúc và dữ liệu của bảng hoàn toàn không hề mất đi trong CSDL.

4.3.4. Thay đổi cách trình bày

Chức năng **Arrange Tables** cho phép hệ thống tự động sắp xếp lại vị trí hiển thị của các bảng hiện có trong mô hình quan hệ dữ liệu. Việc sắp xếp lại các bảng này nhằm giúp cho người sử dụng dễ dàng thấy rõ mối liên kết giữa các bảng.

Đối với một mô hình quan hệ dữ liệu lớn có chứa rất nhiều bảng thì người sử dụng có thể chọn chức năng **Zoom** với các tỉ lệ khác nhau (10%, 25%, 50%, 75%, 100%, 150% và 200%) để có thể thu nhỏ hoặc phóng to các bảng và các mối liên kết của chúng cho dễ nhìn.

Ngoài ra, chức năng **Show Relationship Labels** để hiển thị tên của các quy tắc kiểm tra dữ liệu khóa ngoại liên kết quan hệ giữa các bảng.

4.3.5. In mô hình quan hệ dữ liệu

Chức năng **View Page Breaks** giúp người sử dụng có thể thấy được các bảng dữ liệu trong mô hình quan hệ dữ liệu lớn được trình bày trên nhiều trang giấy khác nhau.

Ngoài ra, để in mô hình quan hệ dữ liệu ra máy in, người sử dụng có thể sử dụng chức năng **Print** hoặc click vào biểu tượng của nút **Print** trên thanh công cụ.

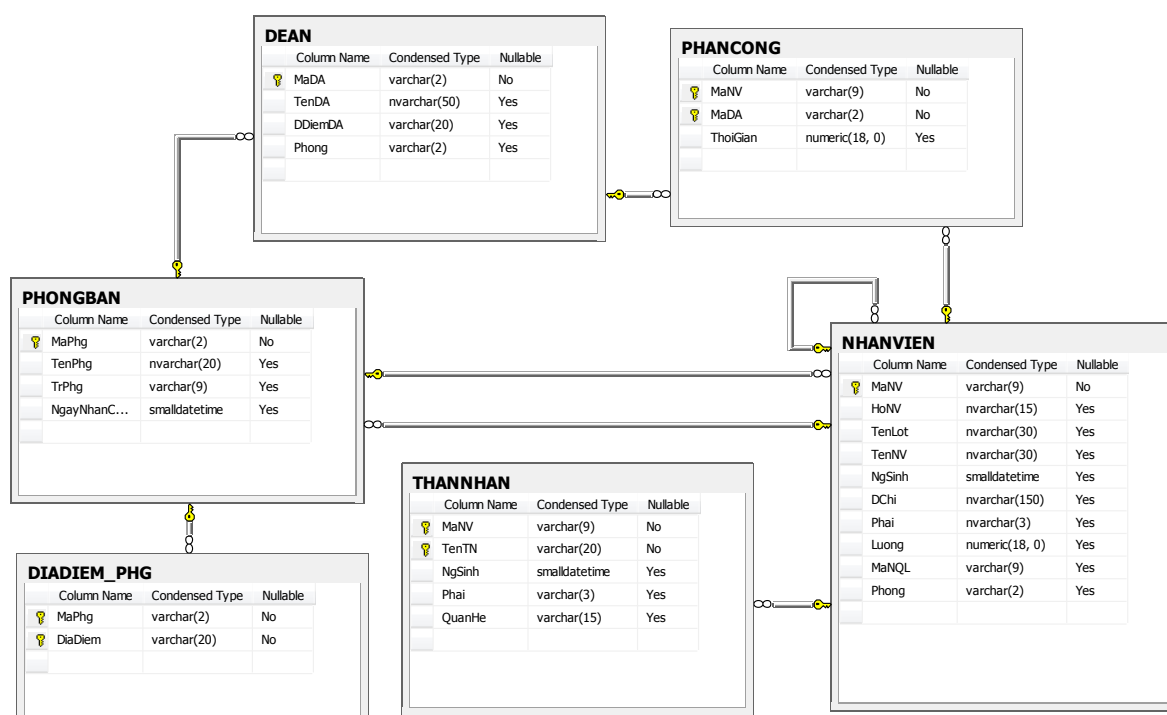
BÀI TẬP KẾT THÚC CHƯƠNG

LÝ THUYẾT

1. Trình bày khái niệm biểu đồ cơ sở dữ liệu (Database Diagram)
2. Trình bày các thao tác có thể thực hiện trên biểu đồ CSDL.

THỰC HÀNH

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 5: NGÔN NGỮ TRUY VẤN DỮ LIỆU T-SQL

Thời lượng: 06 tiết lý thuyết + 07 tiết thực hành

Kết thúc chương này, sinh viên có thể:

- ☞ *Hiểu được khái niệm cơ bản về T-SQL*
- ☞ *Phân biệt được các loại câu lệnh T-SQL*
- ☞ *Viết các câu lệnh T-SQL từ các yêu cầu cụ thể*
- ☞ *Biết được các hàm thường dùng trong ngôn ngữ T-SQL*

5.1. KHÁI NIỆM CƠ BẢN VỀ T-SQL

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

Tùy vào chức năng của câu lệnh, T-SQL được chia làm 4 nhóm:

- **DDL (Data Definition Language):** là những lệnh dùng để quản lý các thuộc tính của một CSDL như định nghĩa các hàng hoặc cột của một table, hay vị trí data file của một database...

- **DML (Data Manipulation Language):** là những lệnh phổ biến dùng để xử lý dữ liệu như SELECT, UPDATE, INSERT, DELETE.

- **DCL (Data Control Language):** là những lệnh quản lý quyền truy cập trên từng đối tượng của CSDL.

Mỗi câu lệnh T-SQL có thể được viết trên nhiều dòng và kết thúc lệnh bởi dấu chấm phẩy (;), tuy nhiên từ khoá, tên hàm, tên thuộc tính, tên bảng, tên đối tượng thì không được phép viết tách xuống hàng.

Phạm vi chương trình môn SQL Server chỉ trình bày một số câu lệnh trong nhóm lệnh DDL và DML. Các câu lệnh trong nhóm lệnh DCL, sinh viên sẽ được học trong học phần Quản trị hệ thống Cơ sở dữ liệu.

5.2. CÁC CÂU LỆNH ĐỊNH NGHĨA DỮ LIỆU

Các câu lệnh định nghĩa dữ liệu bao gồm tạo bảng (CREATE TABLE), xóa bảng (DROP TABLE), chỉnh sửa cấu trúc bảng (ALTER TABLE).

5.2.1. Câu lệnh tạo bảng

Để tạo bảng trong CSDL, sử dụng câu lệnh CREATE TABLE. Cú pháp câu lệnh CREATE TABLE đã được trình bày tại mục 3.3.

5.2.2. Câu lệnh thay đổi cấu trúc bảng

Để thay đổi cấu trúc bảng, sử dụng câu lệnh ALTER TABLE. Cú pháp câu lệnh ALTER TABLE đã được trình bày tại mục 3.4.

5.2.3. Câu lệnh xóa bảng

Để xóa bảng khỏi CSDL, sử dụng câu lệnh DROP TABLE. Cú pháp câu lệnh DROP TABLE đã được trình bày tại mục 3.5.

5.3. CÁC CÂU LỆNH THAO TÁC VỚI DỮ LIỆU

5.3.1. Câu lệnh nhập dữ liệu vào bảng

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

Cú pháp:

```
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.
- 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.

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

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

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

```
INSERT INTO NHANVIEN VALUES
('123', N'Đinh', N'Bá', N'Tiến', '1982/02/27', N'TP
Quảng Ngãi', N'Nam', NULL, NULL, '4')
```

5.3.2. Câu lệnh xóa dữ liệu ra khỏi bảng

Để xóa dữ liệu (mẫu tin) trong bảng, sử dụng câu lệnh DELETE

Cú pháp:

```
DELETE FROM table_name
WHERE some_column=some_value;
```

Trong đó:

- **table_name**: tên bảng có dữ liệu cần xóa
- **some_column=some_value**: điều kiện các mẫu tin được xóa

Lưu ý:

- Nếu không sử dụng mệnh đề điều kiện WHERE, tất cả các mẫu tin trong bảng đều sẽ bị xóa.

Ví dụ: Xóa các nhân viên nữ làm việc ở phòng số 2

```
DELETE FROM NHANVIEN
WHERE MaPhong = 2 AND Phai = N'Nữ'
```

5.3.3. Câu lệnh chỉnh sửa dữ liệu trong bảng

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'
```

5.4. CÂU LỆNH TRUY VẤN DỮ LIỆU

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.

5.4.1. Cú pháp chung

```
SELECT select_list  
[FROM table_source ]  
[WHERE search_condition ]  
[GROUP BY group_by_expression ]  
[HAVING search_condition ]  
[ORDER BY order_expression [ ASC | DESC ] ]
```

Trong đó:

- Mệnh đề **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.

- Mệnh đề **FROM**: danh sách các bảng là tên các bảng. Có thể dùng bí danh cho bảng.

- Mệnh đề **WHERE**: điều kiện lọc.

- Mệnh đề **GROUP BY**: dùng để nhóm các bản ghi theo một số trường nào đó. Kết quả có thể được lọc bởi mệnh đề HAVING .

- Mệnh đề **ORDER BY**: sắp xếp kết quả theo các trường.

Để minh họa các ví dụ, ta sử dụng lược đồ cơ sở dữ liệu quản lý đề án trong các bài thực hành trước.

+ Mệnh đề SELECT - Tìm thông tin từ các cột của bảng

Những thuộc tính được liệt kê trong mệnh đề SELECT sẽ là các thuộc tính có trong quan hệ đích. Các thuộc tính cách nhau bởi dấu phẩy và thứ tự này cũng là thứ tự của các thuộc tính trong quan hệ đích.

Quan hệ nguồn liên quan đến câu truy vấn được đặt sau mệnh đề FROM.

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

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

FROM NHANVIEN

Kết quả truy vấn:

	MaNV	HoNV	TenLot	TenNV	NgSinh	DiaChi	Phai	Luong	MaNQL	Phong
1	123	Đinh	Bá	Tiến	1982-02-27 00:00:00	TP Quảng Ngãi	Nam	5800000	234	4
2	222	Trần	Quốc	Huy	1981-12-27 00:00:00	Đức Phổ	Nam	7200000	901	2
3	234	Nguyễn	Thanh	Tùng	1956-08-12 00:00:00	Sơn Tịnh	Nam	10200000	901	5
4	333	Lê	Ngọc	Hân	1981-12-27 00:00:00	Mộ Đức	Nữ	5300000	222	2
5	345	Bùi	Thúy	Vũ	1985-01-01 00:00:00	Tư Nghĩa	Nữ	6090000	890	4
6	444	Trần	Ngọc	Thúy	1981-12-27 00:00:00	Sơn Tịnh	Nữ	4100000	222	2
7	456	Lê	Thị	Nhàn	1962-07-12 00:00:00	Mộ Đức	Nữ	6590000	901	4
8	567	Nguyễn	Mạnh	Hùng	1985-03-25 00:00:00	Sơn Tịnh	Nam	9700000	234	5
9	678	Trần	Hồng	Quang	1985-01-01 00:00:00	Bình Sơn	Nam	9700000	890	5
10	789	Trần	Thanh	Tâm	1972-06-17 00:00:00	TP Quảng Ngãi	Nam	10200000	234	5
11	890	Cao	Thanh	Huyền	1985-01-01 00:00:00	Tư Nghĩa	Nữ	6510000	901	1
12	901	Vương	Ngọc	Quyên	1987-12-12 00:00:00	Mộ Đức	Nam	6200000	NULL	1

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:

SELECT *
FROM NHANVIEN

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

Kết quả truy vấn:

	Tên đề án	Địa điểm
1	Nâng cao chất lượng muối	Sa Huỳnh
2	Xây dựng nhà máy chế biến thủy sản	Dung Quất
3	Phát triển hạ tầng mạng	TP Quảng Ngãi
4	Truyền tải cấp quang	TP Quảng Ngãi
5	Tin học hóa trường học	Ba Tư
6	Đào tạo nhân lực	Tịnh Phong

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 Tuổi
FROM NHANVIEN
```

Kết quả truy vấn:

	HoVaTen	Tuoi
1	Đinh Bá Tiến	34
2	Trần Quốc Huy	35
3	Nguyễn Thanh Tùng	60
4	Lê Ngọc Hân	35
5	Bùi Thúy Vũ	31
6	Trần Ngọc Thúy	35
7	Lê Thị Nhân	54
8	Nguyễn Mạnh Hùng	31
9	Trần Hồng Quang	31
10	Trần Thanh Tâm	44
11	Cao Thanh Huyền	31
12	Vương Ngọc Quyền	29

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.

+ Mệnh đề WHERE - chọn các mẫu tin thỏa mãn điều kiện

Các biểu thức sau từ khóa WHERE chính là điều kiện chọn các mẫu tin của bảng.

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

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

Kết quả truy vấn:

	MaNV	HoNV	TenLot	TenNV	NgSinh	DiaChi	Phai	Luong	MaNQL	Phong
1	234	Nguyễn	Thanh	Tùng	1956-08-12 00:00:00	Sơn Tịnh	Nam	10200000	901	5
2	567	Nguyễn	Mạnh	Hùng	1985-03-25 00:00:00	Sơn Tịnh	Nam	9700000	234	5
3	678	Trần	Hồng	Quang	1985-01-01 00:00:00	Bình Sơn	Nam	9700000	890	5
4	789	Trần	Thanh	Tâm	1972-06-17 00:00:00	TP Quảng Ngãi	Nam	10200000	234	5

Các điều kiện có thể phối hợp với nhau bởi các toán tử logic (NOT, AND, OR) để tạo nên những biểu thức logic phức tạp hơn.

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

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

Kết quả truy vấn:

	MaNV	HoNV	TenLot	TenNV	NgSinh	DiaChi	Phai	Luong	MaNQL	Phong
1	222	Trần	Quốc	Huy	1981-12-27 00:00:00	Đức Phổ	Nam	7200000	901	2
2	234	Nguyễn	Thanh	Tùng	1956-08-12 00:00:00	Sơn Tịnh	Nam	10200000	901	5
3	333	Lê	Ngọc	Hân	1981-12-27 00:00:00	Mộ Đức	Nữ	5300000	222	2
4	444	Trần	Ngọc	Thúy	1981-12-27 00:00:00	Sơn Tịnh	Nữ	4100000	222	2
5	567	Nguyễn	Mạnh	Hùng	1985-03-25 00:00:00	Sơn Tịnh	Nam	9700000	234	5
6	678	Trần	Hồng	Quang	1985-01-01 00:00:00	Bình Sơn	Nam	9700000	890	5
7	789	Trần	Thanh	Tâm	1972-06-17 00:00:00	TP Quảng Ngãi	Nam	10200000	234	5
8	890	Cao	Thanh	Huyền	1985-01-01 00:00:00	Tứ Nghĩa	Nữ	6510000	901	1
9	901	Vương	Ngọc	Quyên	1987-12-12 00:00:00	Mộ Đức	Nam	6200000	NULL	1

Có thể sử dụng toán tử IN để thay thế toán tử OR để làm biểu thức logic đơn giản hơn.

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

Kết quả truy vấn:

	MaNV	HoNV	TenLot	TenNV	NgSinh	DiaChi	Phai	Luong	MaNQL	Phong
1	222	Trần	Quốc	Huy	1981-12-27 00:00:00	Đức Phổ	Nam	7200000	901	2
2	234	Nguyễn	Thanh	Tùng	1956-08-12 00:00:00	Sơn Tịnh	Nam	10200000	901	5
3	333	Lê	Ngọc	Hân	1981-12-27 00:00:00	Mộ Đức	Nữ	5300000	222	2
4	444	Trần	Ngọc	Thúy	1981-12-27 00:00:00	Sơn Tịnh	Nữ	4100000	222	2
5	567	Nguyễn	Mạnh	Hùng	1985-03-25 00:00:00	Sơn Tịnh	Nam	9700000	234	5
6	678	Trần	Hồng	Quang	1985-01-01 00:00:00	Bình Sơn	Nam	9700000	890	5
7	789	Trần	Thanh	Tâm	1972-06-17 00:00:00	TP Quảng Ngãi	Nam	10200000	234	5
8	890	Cao	Thanh	Huyền	1985-01-01 00:00:00	Tứ Nghĩa	Nữ	6510000	901	1
9	901	Vương	Ngọc	Quyên	1987-12-12 00:00:00	Mộ Đức	Nam	6200000	NULL	1

Toán tử LIKE được sử dụng để so sánh các chuỗi có nét tương đồng. Khi sử dụng toán tử LIKE, có hai ký tự đại diện cần quan tâm:

- Ký tự %: đại diện cho một chuỗi ký tự bất kỳ

- Ký tự _ : đại diện cho một ký tự bất kỳ

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

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

Kết quả truy vấn:

	MaNV	HoNV	TenLot	TenNV	NgSinh	DiaChi	Phai	Luong	MaNQL	Phong
1	123	Đinh	Bá	Tiến	1982-02-27 00:00:00	TP Quảng Ngãi	Nam	5800000	234	4
2	234	Nguyễn	Thanh	Tùng	1956-08-12 00:00:00	Sơn Tịnh	Nam	10200000	901	5
3	789	Trần	Thanh	Tâm	1972-06-17 00:00:00	TP Quảng Ngãi	Nam	10200000	234	5

Mệnh đề ORDER BY – Quy định thứ tự sắp xếp các mẫu tin

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.

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
```

Kết quả truy vấn:

	HoNV	TenLot	TenNV	Luong
1	Trần	Thanh	Tâm	10200000
2	Nguyễn	Thanh	Tùng	10200000
3	Nguyễn	Mạnh	Hùng	9700000
4	Trần	Hồng	Quang	9700000
5	Lê	Thị	Nhàn	6590000
6	Cao	Thanh	Huyền	6510000
7	Vương	Ngọc	Quyên	6200000
8	Bùi	Thúy	Vũ	6090000
9	Đinh	Bá	Tiến	5800000

5.4.2. Truy vấn thông tin từ nhiều bảng

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

Kết quả truy vấn:

	HoNV	TenLot	TenNV	TenPhg
1	Đinh	Bá	Tiến	Điều hành
2	Nguyễn	Thanh	Tùng	Nghiên cứu
3	Bùi	Thúy	Vũ	Điều hành
4	Lê	Thị	Nhàn	Điều hành
5	Nguyễn	Mạnh	Hùng	Nghiên cứu
6	Trần	Hồng	Quang	Nghiên cứu
7	Trần	Thanh	Tâm	Nghiên cứu
8	Cao	Thanh	Huyền	Quản lý
9	Vương	Ngọc	Quyên	Quản lý

Việc liên kết hai hay nhiều bảng trong câu lệnh SELECT bằng cách trên 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
```

Kết quả truy vấn:

	HoNV	TenLot	TenNV	TenPhg
1	Đinh	Bá	Tiến	Điều hành
2	Nguyễn	Thanh	Tùng	Nghiên cứu
3	Bùi	Thúy	Vũ	Điều hành
4	Lê	Thị	Nhàn	Điều hành
5	Nguyễn	Mạnh	Hùng	Nghiên cứu
6	Trần	Hồng	Quang	Nghiên cứu
7	Trần	Thanh	Tâm	Nghiên cứu
8	Cao	Thanh	Huyền	Quản lý
9	Vương	Ngọc	Quyên	Quản lý

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.

Lưu ý: Sinh viên có thể tham khảo cách dùng các từ khóa INNER JOIN, LEFT JOIN, RIGHT JOIN, FULL JOIN tại các trang web có đường dẫn sau:

http://www.w3schools.com/sql/sql_join_inner.asp

http://www.w3schools.com/sql/sql_join_left.asp

http://www.w3schools.com/sql/sql_join_right.asp

http://www.w3schools.com/sql/sql_join_full.asp

5.4.3. Gom nhóm dữ liệu

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.

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
```

Kết quả truy vấn:

	TenDA	TongNhanVien
1	Nâng cao chất lượng muối	1
2	Xây dựng nhà máy chế biến thủy sản	2
3	Phát triển hạ tầng mạng	3
4	Truyền tải cáp quang	1
5	Tin học hóa trường học	4

+ Mệnh đề HAVING – điều kiện lọc sau khi gom nhóm

Khi cần lọc các mẫu tin thỏa 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
```

Kết quả truy vấn:

	TenDA	TongNhanVien
1	Phát triển hạ tầng mạng	3
2	Tin học hóa trường học	4

5.4.4. Câu lệnh truy vấn lồng nhau

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.

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)
```

Kết quả truy vấn:

	MaNV	HoNV	TenLot	TenNV	NgSinh	DiaChi	Phai	Luong	MaNQL	Phong
1	123	Đinh	Bá	Tiến	1982-02-27 00:00:00	TP Quảng Ngãi	Nam	5800000	234	4
2	234	Nguyễn	Thanh	Tùng	1956-08-12 00:00:00	Sơn Tịnh	Nam	10200000	901	5
3	345	Bùi	Thúy	Vũ	1985-01-01 00:00:00	Tư Nghĩa	Nữ	6090000	890	4
4	456	Lê	Thị	Nhàn	1962-07-12 00:00:00	Mộ Đức	Nữ	6590000	901	4
5	901	Vương	Ngọc	Quyền	1987-12-12 00:00:00	Mộ Đức	Nam	6200000	NULL	1

5.5. CÁC TOÁN TỬ TRONG T-SQL

5.5.1. Toán tử số học

Bảng 5.1. Các toán tử số học trong T-SQL

Ký hiệu	Ý nghĩa
+	Thực hiện phép cộng
-	Thực hiện phép trừ
*	Thực hiện phép nhân
/	Thực hiện phép chia
^	Thực hiện phép lũy thừa
%	Thực hiện phép chia lấy dư

5.5.2. Toán tử logic

Bảng 5.2. Các toán tử logic trong T-SQL

Ký hiệu	Ý nghĩa
NOT	Thực hiện phép phủ định
AND	Thực hiện phép hội
OR	Thực hiện phép tuyển

5.5.3. Toán tử so sánh

Bảng 5.3. Các toán tử so sánh trong T-SQL

Ký hiệu	Ý nghĩa
=	Thực hiện phép so sánh bằng
>	Thực hiện phép so sánh lớn hơn

<	Thực hiện phép so sánh nhỏ hơn
>=	Thực hiện phép so sánh lớn hơn hoặc bằng
<=	Thực hiện phép so sánh nhỏ hơn hoặc bằng
<>	Thực hiện phép so sánh khác
!=	Thực hiện phép so sánh khác
!>	Thực hiện phép so sánh không lớn hơn
!<	Thực hiện phép so sánh không nhỏ hơn
BETWEEN ... AND ...	Thực hiện phép so sánh trong khoảng

5.5.4. Toán tử tập hợp

Bảng 5.4. Các toán tử tập hợp trong T-SQL

Ký hiệu	Ý nghĩa
IN	Thực hiện phép kiểm tra phần tử thuộc tập hợp
LIKE	Thực hiện phép kiểm tra chuỗi tương đương
UNION	Thực hiện phép hợp hai tập hợp
INTERSECT	Thực hiện phép giao hai tập hợp
MINUS	Thực hiện phép trừ hai tập hợp

5.6. CÁC HÀM THƯỜNG DÙNG TRONG T-SQL

5.6.1. 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

5.6.2. 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 trống dành để chứa các ký số sau khi chuyển đổi sang kiểu dữ liệu chuỗi.
- **Số_lẻ**: chỉ định số thập phân

5.6.3. Các hàm ngày giờ

Các hàm này thường có thông số vào là kiểu dữ liệu ngày giờ và giá trị trả về của chúng có thể là kiểu dữ liệu số, chuỗi hoặc ngày giờ

Hàm DATE

Trả về giá trị là ngày, tháng, năm của hệ thống.

Cú pháp:

```
DATE ()
```

Hàm TIME

Trả về giá trị là giờ, phút, giây của hệ thống.

Cú pháp:

```
TIME ()
```

Hàm GETDATE

Giá trị trả về là ngày giờ hiện hành của hệ thống Microsoft SQL Server

Cú pháp:

```
GETDATE ()
```

Các hàm DAY, MONTH, YEAR

Giá trị trả về là một số nguyên chỉ định ngày (day), tháng (month), năm (year) của một ngày bất kỳ.

Cú pháp:

DAY (Ngày) MONTH (Ngày) YEAR (Ngày)

Trong đó:

- **Ngày**: là một biểu thức, biến, tên cột dữ liệu hoặc giá trị cụ thể có kiểu dữ liệu ngày.

5.6.4. 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)
```

5.6.4. 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)
```

5.7. BIẾN TRONG NGÔN NGỮ T-SQL

5.7.1. Biến cục bộ và biến hệ thống

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:

Bảng 5.4. Một số 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.

5.7.2. Làm việc với biến cục bộ

a. 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
```

b. 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
```

c. 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
```

5.8. CÁC CẤU TRÚC ĐIỀU KHIỂN VÀ CÂU LỆNH TRONG T-SQL

5.8.1. 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"
```

5.8.2. Cấu trúc lặp WHILE

Cú pháp:

```
WHILE Biểu_thức_luận_lý  
BEGIN  
    Các lệnh lặp  
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.

5.8.3. Câu lệnh GOTO

Cú pháp:

```
GOTO Tên_nhãn
```

Chức năng: Chuyển luồng thực thi câu lệnh đến nhãn đã được định nghĩa mà bỏ qua các câu lệnh sau câu lệnh GOTO.

Ví dụ:

```
DECLARE @Counter int;  
SET @Counter = 1;  
WHILE @Counter < 10  
BEGIN  
    SELECT @Counter  
    SET @Counter = @Counter + 1  
    IF @Counter = 4 GOTO Branch_One --Jumps to the  
first branch.
```

```

        IF @Counter = 5 GOTO Branch_Two  --This will
never execute.
END
Branch_One:
        SELECT 'Jumping To Branch One.'
        GOTO Branch_Three; --This will prevent
Branch_Two from executing.
Branch_Two:
        SELECT 'Jumping To Branch Two.'
Branch_Three:
        SELECT 'Jumping To Branch Three.';

```

5.8.4. Câu lệnh RETURN

Cú pháp:

```
RETURN Biểu_thức_số_nguyên
```

Chức năng:

Trả về một giá trị số nguyên cho một hàm hoặc một thủ tục, đồng thời, kết thúc việc thực hiện hàm hoặc thủ tục chứa câu lệnh RETURN này.

BÀI TẬP KẾT THÚC CHƯƠNG

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 $\leq$ 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.
28. Cho biết số lượng đề án của công ty
29. Liệt kê danh sách các phòng ban có tham gia chủ trì các đề án
30. Cho biết số lượng các phòng ban có tham gia chủ trì các đề án
31. Cho biết số lượng đề án do phòng Nghiên Cứu chủ trì
32. Cho biết lương trung bình của các nữ nhân viên
33. Cho biết số thân nhân của nhân viên Đinh Bá Tiến
34. Liệt kê danh sách 3 nhân viên lớn tuổi nhất, danh sách bao gồm họ tên và năm sinh.
35. 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.
36. Với mỗi đề án, cho biết có bao nhiêu nhân viên tham gia đề án đó
37. Cho biết mã đề án có đông nhân viên tham gia nhất.
38. Với mỗi nhân viên, cho biết họ và tên nhân viên và số lượng thân nhân của nhân viên đó.
39. Với mỗi nhân viên, cho biết họ tên của nhân viên và số lượng đề án mà nhân viên đó đã tham gia.
40. Với mỗi nhân viên, cho biết số lượng nhân viên mà nhân viên đó quản lý trực tiếp.
41. Với mỗi phòng ban, liệt kê tên phòng ban và lương trung bình của những nhân viên làm việc cho phòng ban đó.

42. 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.
43. Với mỗi phòng ban, cho biết tên phòng ban và số lượng đề án mà phòng ban đó chủ trì
44. 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
45. 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.
46. Cho biết số đề án diễn ra tại từng địa điểm
47. Với mỗi đề án, cho biết tên đề án và số lượng công việc của đề án này.
48. Cho biết danh sách các nhân viên có tối thiểu một thân nhân (dùng INNER JOIN, IN, EXISTS).
49. Cho biết danh sách các nhân viên không có thân nhân nào (dùng LEFT JOIN, NOT IN, NOT EXISTS).
50. Cho biết họ tên các nhân viên có trên 2 thân nhân.
51. Cho biết họ tên những trưởng phòng có tối thiểu một thân nhân.
52. Cho biết danh sách những trưởng phòng chưa có gia đình.
53. Cho biết họ tên các nhân viên phòng Quản lý có mức lương trên mức lương trung bình của phòng Quản lý.
54. Cho biết họ tên nhân viên có mức lương trên mức lương trung bình của phòng mà nhân viên đó đang làm việc
55. Cho biết tên phòng ban và họ tên trưởng phòng của phòng ban có đông nhân viên nhất.
56. Cho biết danh sách các đề án mà nhân viên có mã là 456 chưa tham gia.
57. 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.
58. 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.

59. 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.
60. Liệt kê danh sách các đề án mà cả hai nhân viên có mã số 123 và 234 cùng làm.
61. Liệt kê danh sách các đề án mà cả hai nhân viên Đinh Bá Tiến và Lê Thị Nhân cùng làm.
62. Danh sách những nhân viên (bao gồm mã nhân viên, họ tên, phái) làm việc trong mọi đề án của công ty
63. 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.
64. Tìm những nhân viên (bao gồm mã nhân viên, họ tên) được phân công tất cả đề án mà nhân viên Lê Thị Nhân làm việc
65. Cho biết danh sách nhân viên tham gia vào tất cả các đề án ở TP Quảng Ngãi
66. Cho biết phòng ban chủ trì tất cả các đề án ở TP Quảng Ngãi
67. Cho biết những phòng ban có nhân viên tham gia cả 2 dự án ở Dung Quất và Sa Huỳnh.
68. Cho biết những phòng ban có nhân viên tham gia dự án ở Dung Quất hoặc Sa Huỳnh.
69. Hãy xóa các nhân viên chưa tham gia đề án nào.
70. Hãy xóa các nhân viên không có thân nhân.

Chương 6: VIEW (KHUNG NHÌN)

Thời lượng: 02 tiết lý thuyết + 02 tiết thực hành

Kết thúc chương này, sinh viên có thể:

- ☞ *Hiểu được chức năng của view trong SQL Server*
- ☞ *Thực hiện tạo view trong SQL Server*

6.1. TỔNG QUAN VỀ VIEW

6.1.1. Khái niệm 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.

6.1.2. 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.

6.2. TẠO VIEW

Có hai cách dùng để tạo View: sử dụng giao diện View Designer hoặc sử dụng câu lệnh T-SQL.

Cách 1: Sử dụng giao diện View Designer

1. Tại cửa sổ **Object Explorer**, click phải vào mục **Views** của CSDL cần tạo, chọn **New View**
2. Trong hộp thoại **Add Table**, chọn các table cơ sở của view cần tạo.
3. Chọn các trường trong các table cơ sở vừa thêm vào.
4. Chọn **Execute SQL** để xem kết quả.
5. Nhấn **Ctrl-S** để lưu view

Cách 2: Sử dụng câu lệnh T-SQL

Cú pháp để tạo view bằng câu lệnh T-SQL:

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

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

Cách 2:

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

*** 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 vw_NhanVienPhong5, ta thực thi thủ tục *sp_helptext* như sau:

```
EXEC sp_helptext vw_NhanVienPhong5
```

Kết quả trả về như sau:

Text

```

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

```

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

```

thì khi sử dụng câu lệnh

```
EXEC sp_helptext vw_NhanVienPhong5
```

kết quả sẽ là:

```
The text for object 'vw_NhanVienPhong5' is encrypted.
```

****Từ khóa WITH CHECK OPTION***

Để hiểu rõ ý nghĩa từ khóa WITH CHECK OPTION, ta xem xét ví dụ sau:

Ví dụ: Tạo view vw_NhanVienNu để liệt kê danh sách các nhân viên nữ:

```

CREATE VIEW vw_NhanVienNu
AS
SELECT * FROM NHANVIEN WHERE Phai = N'Nữ'

```

Để thêm mới một nhân viên nam vào table NHANVIEN thông qua view vw_NhanVienNu, ta sử dụng câu lệnh INSERT như sau:

```

INSERT INTO vw_NhanVienNu (MaNV, HoNV, TenLot,
TenNV, Phai)
VALUES ('246', N'Cao', N'Đức', N'Hùng', N'Nam')

```

Nhận xét: mặc dù vw_NhanVienNu dùng để liệt kê các nhân viên nữ nhưng khi chèn một mẫu tin có thuộc tính Phai là “Nam” thông qua view vẫn được chấp nhận.

Để tránh tình trạng nhập nhầm trên, khi sử dụng câu lệnh tạo view, có thể thêm từ khóa WITH CHECK OPTION. Tác dụng của từ khóa này là ngăn cản các thao tác thay đổi dữ liệu thông qua view nhưng không đúng với điều kiện của mệnh đề WHERE bên trong câu lệnh SELECT định nghĩa View.

Ví dụ: Tạo view vw_NhanVienNu để liệt kê danh sách các nhân viên nữ, có sử dụng từ khóa WITH CHECK OPTION:

```
CREATE VIEW vw_NhanVienNu
AS
SELECT * FROM NHANVIEN WHERE Phai = N'Nữ'
WITH CHECK OPTION
```

Nếu thực hiện lệnh chèn dữ liệu:

```
INSERT INTO vw_NhanVienNu (MaNV, HoNV, TenLot,
TenNV, Phai)
VALUES ('246', N'Cao', N'Đức', N'Hùng', N'Nam')
```

thì kết quả sẽ là:

```
Msg 550, Level 16, State 1, Line 1
The attempted insert or update failed because the target view either specifies WITH
CHECK OPTION or spans a view that specifies WITH CHECK OPTION and one or more rows
resulting from the operation did not qualify under the CHECK OPTION constraint.
The statement has been terminated.
```

6.3. XEM VÀ CẬP NHẬT DỮ LIỆU THÔNG QUA 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.

6.4. THAY ĐỔI ĐỊNH NGHĨA VÀ HỦY BỎ VIEW

6.4.1. 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
```

6.4.2. 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 CHƯƠNG

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 7: STORED PROCEDURE (THỦ TỤC LƯU TRỮ)

Thời lượng: 02 tiết lý thuyết + 04 tiết thực hành

Kết thúc chương này, sinh viên có thể:

- ☞ *Hiểu được chức năng của các đối tượng Stored Procedure trong SQL Server*
- ☞ *Phân tích được yêu cầu, từ đó tạo được Stored Procedure hoàn chỉnh*

7.1. TỔNG QUAN VỀ STORED PROCEDURE

7.1.1. Khái niệm 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.

Mỗi Stored Procedure được đặc trưng bởi một tên nhất định. Một Stored Procedure có thể nhận các tham số truyền vào cũng như có thể trả về các giá trị thông qua các tham số (giống như trong ngôn ngữ lập trình). Khi một Stored Procedure đã được định nghĩa, nó có thể được gọi thông qua tên, nhận các tham số truyền vào, thực thi các câu lệnh T-SQL bên trong Stored Procedure và trả về các giá trị sau khi thực hiện.

Một Stored Procedure có thể được gọi bên trong một Stored Procedure khác. Phạm vi hoạt động của các Stored Procedure do người dùng tạo ra chỉ có tính cục bộ bên trong CSDL lưu trữ Stored Procedure đó.

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.

7.1.2. Các Stored Procedure hệ thống

Một số Stored Procedure hệ thống thường được sử dụng:

System Stored Procedure	Chức năng
sp_databases	Danh sách những Database có thể (available) trên Server (Danh sách này sẽ là khác nhau tùy thuộc vào quyền của người sử dụng)
sp_server_info	Chi tiết những thông tin về Server, ví dụ như tập các đặc tính, phiên bản...
sp_stored_procedures	Danh sách tất cả các thủ tục có thể trên môi trường hiện tại
sp_tables	Danh sách tất cả các bảng có thể trên môi trường hiện tại
sp_start_job	Khởi động tất cả các automated task ngay lập tức
sp_stop_job	Ngừng lại tất cả các automated task đang chạy
sp_password	Thay đổi password cho login account
sp_configure	Thay đổi lựa chọn cấu hình chung của SQL SERVER. Khi người sử dụng không lựa chọn thì hệ thống sẽ hiển thị cấu hình mặc định.
sp_help	Hiển thị thông tin về bất kỳ đối tượng nào trong Database
sp_helptext	Hiển thị nội dung (text) của các đối tượng

7.1.3. 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.

7.2. TẠO VÀ THỰC THI STORED PROCEDURE

7.2.1. 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 usp_ 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.

Trước khi tiến hành tạo Stored Procedure, việc đầu tiên là phân tích yêu cầu và xác định các thành phần sau:

- *Tên Stored Procedure*: tên gọi ngắn gọn và mang tính gợi nhớ về chức năng của Stored Procedure. Tên Stored Procedure thường được đặt với tiếp đầu ngữ usp_
- *Danh sách các tham số đầu vào*: số lượng các tham số đầu vào cùng với kiểu dữ liệu của từng tham số.
- *Danh sách các tham số đầu ra*: số lượng các tham số đầu ra cùng với kiểu dữ liệu của từng tham số.
- Nội dung các câu lệnh T-SQL bên trong Stored Procedure.

7.2.2. 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

7.2.3. Tham số trong Stored Procedure

a) Stored Procedure không chứa tham số

Ví dụ 7.1: 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
```

b) Stored Procedure có một tham số

Ví dụ 7.2: 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'
```

c) Stored Procedure có nhiều tham số

Ví dụ 7.3: 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'
```

c) *Stored Procedure có tham số đầu ra*

Ví dụ 7.4: 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
```

7.3. THAY ĐỔI NỘI DUNG VÀ HỦY STORED PROCEDURE

7.3.1. 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.

7.3.2. 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
```

BÀI TẬP KẾT THÚC CHƯƠNG

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.

Chương 8: FUNCTION (HÀM)

Thời lượng: 02 tiết lý thuyết + 04 tiết thực hành

Kết thúc chương này, sinh viên có thể:

- ☞ *Hiểu được chức năng của đối tượng Function trong SQL Server*
- ☞ *Phân tích được yêu cầu, từ đó tạo được Function hoàn chỉnh*

8.1. TỔNG QUAN VỀ FUNCTION

Function (hàm) được dùng tương tự như Stored Procedure giúp tối ưu hoạt động của CSDL, giảm thời gian viết lại các lệnh T-SQL thường dùng. Ta có thể truyền vào các tham số cho Function. Ngoài các function được SQL Server định nghĩa sẵn (Built-in Function) đã được trình bày trong mục 5.6, còn có các function do người dùng tạo ra (User-defined Function). Nội dung trong phần này sẽ đề cập đến các thao tác làm việc với User-defined Function.

Một số điểm khác biệt của Function so với Stored Procedure:

- Function luôn trả về một giá trị.
- Function phải có tham số kèm theo khi thực hiện lời gọi.
- Function có thể được gọi bên trong câu lệnh SELECT.

User-defined function (UDF) được chia làm 3 loại: **scalar function**, **inline table function** và **multi statement function**.

- **Scalar function**: trả về duy nhất một giá trị dựa trên các tham số truyền vào.
- **Inline table function**: trả về một bảng dựa trên một câu lệnh SQL duy nhất.
- **Multistatement table function**: trả về một bảng nhưng dựa trên một tập nhiều câu lệnh SQL.

Gọi thực hiện Function

- **Scalar Function**: có thể được gọi thực hiện tại bất kỳ nơi nào mà một biểu thức vô hướng có kiểu dữ liệu tương đương được chấp nhận trong câu lệnh T-SQL.
- **Inline Table Function** và **Multi Statement Function**: được sử dụng giống như view.

8.2. TẠO VÀ GỌI THỰC HIỆN FUNCTION

8.2.1. Scalar Function

Scalar Function là hàm trả về một giá trị cụ thể.

Cú pháp:

```
CREATE FUNCTION Tên_Function (Danh sách tham số)
RETURNS Kiểu dữ liệu trả về
AS
BEGIN
    Các câu lệnh T-SQL
    RETURN Biểu thức
END
```

Trong đó:

- **Tên_Function**: tên Function, người ta thường sử dụng tiếp đầu ngữ **udf_** để đặt tên cho Function.

- **Danh sách tham số**: danh sách tham số đầu vào của Function. Các tham số chỉ có phạm vi cục bộ bên trong Function mà nó được khai báo. Tên của mỗi tham số là duy nhất.

- **Biểu thức**: Giá trị của biểu thức phải có kiểu dữ liệu đúng với kiểu dữ liệu sau từ khóa RETURN trong phần khai báo.

Ví dụ: Viết function trả về tổng thời gian làm việc của một nhân viên bất kỳ, với tham số truyền vào là mã nhân viên.

```
CREATE FUNCTION udfTongGio (@manv varchar(3))
RETURNS int
AS
BEGIN
    DECLARE @tong int
    SELECT @tong = SUM(ThoiGian)
    FROM PHANCONG
    WHERE MaNV = @manv
```

```
RETURN @tong  
END
```

Sử dụng function ***udfTongGio*** để liệt kê danh sách các nhân viên cùng với tổng số giờ làm việc của từng nhân viên như sau:

```
SELECT MaNV, HoNV, TenLot, TenNV, udfTongGio(MaNV)  
FROM NHANVIEN
```

Để hiển thị tổng số giờ làm việc của nhân viên mang mã số 123, ta thực hiện như sau:

```
PRINT udfTongGio('123')
```

8.2.2. Inline Table Function

Inline Table Function là function có kiểu dữ liệu trả về là một table, dựa trên một câu lệnh SQL duy nhất.

Cú pháp:

```
CREATE FUNCTION Tên_Function (Danh sách tham số)  
RETURNS Table  
AS  
RETURN Câu lệnh SELECT
```

Ví dụ: Viết function trả về danh sách nhân viên của từng phòng, với tham số truyền vào là mã phòng

```
CREATE FUNCTION udfNhanVien (@maphong varchar(3))  
RETURNS TABLE  
AS  
RETURN (SELECT *  
FROM NHANVIEN  
WHERE Phong = @maphong)
```

Sử dụng function vừa tạo để liệt kê danh sách nhân viên ở phòng 5 như sau:

```
SELECT * FROM udfNhanVien('5')
```

Nhận xét: Inline Table Function khắc phục được nhược điểm không có tham số của View.

8.2.3. Multistatement Table Function

Giống như Inline Table Function, Multistatement Table Function cũng có kiểu dữ liệu trả về là một table. Tuy nhiên, nó cho phép thực hiện các câu lệnh SELECT phức tạp và các câu lệnh khác như UPDATE, INSERT liên quan đến table trả về... Đồng thời, cấu trúc bảng trả về có thể được thiết lập trong cặp dấu ngoặc đơn ngay sau câu lệnh RETURNS.

Cú pháp:

```
CREATE FUNCTION Tên_Function (Danh sách tham số)
RETURNS Tên_biến Table
(
    Danh sách các trường
)
AS
BEGIN
    Các câu lệnh T-SQL
    RETURN
END
```

Ví dụ: Viết function trả về danh sách các nhân viên trên 50 tuổi của từng phòng, với tham số truyền vào là mã phòng. Thông tin bao gồm mã nhân viên, họ tên, phụ cấp. Biết Phụ cấp = 20% * lương.

```
CREATE FUNCTION udfNhanVien (@maphong varchar(3))
RETURNS @tblNhanVien TABLE
(
    MaNV varchar(5),
    HoTen nvarchar(50),
    PhuCap numeric(18,0)
```

```

)
AS
BEGIN
    INSERT INTO @tblNhanVien
    SELECT MaNV, HoNV + ' ' + TenLot + ' ' + TenNV
AS HoTen, Luong
FROM NHANVIEN
WHERE Phong = @maphong
AND YEAR(GETDATE()) - YEAR(NgSinh) > 50

UPDATE @tblNhanVien
SET PhuCap = PhuCap * 0.2

RETURN
END

```

8.3. THAY ĐỔI NỘI DUNG VÀ XÓA FUNCTION

8.3.1. Thay đổi nội dung Function

Có thể thay đổi nội dung một User Defined Function đã được tạo ra bằng câu lệnh ALTER FUNCTION.

Cú pháp của câu lệnh ALTER FUNCTION cũng giống như câu lệnh CREATE FUNCTION.

8.3.2. Xóa Function

Để xóa một User Defined Function không còn sử dụng, ta sử dụng câu lệnh DROP FUNCTION có cú pháp như sau:

```
DROP FUNCTION Tên_Function
```

BÀI TẬP KẾT THÚC CHƯƠNG

LÝ THUYẾT

1. Trình bày chức năng của Function (hàm) trong cơ sở dữ liệu.
2. Có bao nhiêu loại Function? Trường hợp nào thì nên sử dụng loại Function nào?
3. Khi nào nên sử dụng hàm (Function), khi nào nên sử dụng Stored Procedure?
4. So sánh sự giống và khác nhau về chức năng giữa Function trong SQL Server và hàm trong ngôn ngữ lập trình C.

THỰC HÀNH

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

1. Viết hàm trả về tổng số dự án theo phòng ban chủ trì (tham số truyền vào là mã phòng ban).
2. Viết hàm trả về tổng số giờ tham gia đề án (Time_Total) của nhân viên (tham số truyền vào là mã nhân viên)
3. Viết hàm trả về tổng tiền thưởng cho nhân viên dựa vào tổng số giờ tham gia đề án (Time_Total) như sau:
 - + Nếu Time_Total ≥ 30 và ≤ 60 thì tổng tiền thưởng = 1.000.000 đ
 - + Nếu Time_Total > 60 và < 100 thì tổng tiền thưởng = 1.500.000 đ
 - + Nếu Time_Total ≥ 100 và < 150 thì tổng tiền thưởng = 2.000.000 đ
 - + Nếu Time_Total ≥ 150 thì tổng tiền thưởng = 2.500.000 đ
4. Viết hàm trả về tổng lương nhận được của nhân viên theo đề án (tham số truyền vào là mã nhân viên và mã đề án). Biết:
 - + tổng lương = lương + lương đề án;
 - + lương đề án = 100.000 đ * thời gian.
5. Viết hàm trả về tiền lương trung bình của một phòng ban tùy ý (tham số truyền vào là mã phòng ban)
6. Viết hàm trả về tổng tiền lương trung bình của tất cả các phòng ban.

7. Viết hàm trả về danh sách các nhân viên theo từng phòng ban thông tin gồm: MaNV, HoTen, NgaySinh, NguoiThan, Luong (*viết bằng hai cách: Inline Table-Valued Functions và Multistatement Table-Valued*)

8. Viết hàm trả về danh sách các nhân viên trên 50 tuổi của từng phòng, với tham số truyền vào là mã phòng. Thông tin bao gồm mã nhân viên, họ tên, phụ cấp.

Biết Phụ cấp = 20% * lương.

Chương 9: TRIGGER

Thời lượng: 02 tiết lý thuyết + 04 tiết thực hành

Kết thúc chương này, sinh viên có thể:

- ☞ *Hiểu được chức năng và cơ chế hoạt động của Trigger*
- ☞ *Phân tích vấn đề, tạo các Trigger theo yêu cầu*

9.1. TỔNG QUAN VỀ TRIGGER

9.1.1. 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.

9.1.2. 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.

9.1.3. 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.

9.2. TẠO TRIGGER

9.2.1. 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.

9.2.2. Các lệnh, biến hệ thống và các hàm thường sử dụng trong trigger

Mệnh đề UPDATE

Cú pháp:

```
UPDATE (Tên_cột)
```

Ý nghĩa: Xác định có hay không một biến cố insert hoặc update xảy ra trên cột đã chỉ ra tại mục tham số **Tên_cột**. Mệnh đề UPDATE sẽ trả về giá trị TRUE nếu có biến cố xảy ra, ngược lại, trả về giá trị FALSE.

Ví dụ: Tạo trigger cho thao tác UPDATE trên bảng NHANVIEN. Nếu có sự thay đổi tên nhân viên thì hiện ra thông báo: “Đã sửa tên nhân viên”

```
CREATE TRIGGER tgCapNhatNhanVien
ON NHANVIEN
FOR UPDATE
AS
BEGIN
    IF UPDATE(TenNV)
    BEGIN
        PRINT N'Dã sửa tên nhân viên'
    END
END
```

BIẾN @@ROWCOUNT

Ý nghĩa: Trả về số dòng bị ảnh hưởng bởi câu lệnh T-SQL ngay trước đó trong Trigger.

Ví dụ: Tạo trigger cho thao tác UPDATE trên bảng NHANVIEN. Sau khi thực hiện chỉnh sửa, trả về câu thông báo: “Có ... nhân viên đã được cập nhật thông tin”

```
CREATE TRIGGER tgCapNhatNhanVien2
ON NHANVIEN
FOR UPDATE
AS
BEGIN
    PRINT N'Có ' + CAST(@@rowcount AS varchar(5)) +
    N' nhân viên đã được cập nhật thông tin'
END
```

Sau khi tạo trigger, nếu thực hiện câu lệnh UPDATE để tăng lương cho nhân viên nữ

```
UPDATE NHANVIEN SET Luong = Luong * 1.2 WHERE Phai =
N'Nữ'
```

sẽ hiện ra câu thông báo như sau:

```
Có 3 nhân viên đã được cập nhật thông tin
(3 row(s) affected)
```

CÂU LỆNH RETURN

Ý nghĩa: được dùng để chấm dứt việc thi hành không cần thiết các câu lệnh trong trigger.

CÂU LỆNH RAISERROR

Ý nghĩa: được dùng để hiển thị thông báo lỗi hoặc in ra màn hình câu thông báo mang tính chất cảnh báo. Chuỗi thông báo khi sử dụng RAISERROR sẽ được gửi về cho client từ server.

CÂU LỆNH ROLLBACK TRANSACTION

Ý nghĩa: được dùng để bỏ qua toàn bộ thao tác trước đó là nguyên nhân kích hoạt trigger.

Ví dụ: Tạo trigger cho thao tác UPDATE trên bảng NHANVIEN nhằm mục đích không cho người sử dụng chỉnh sửa mã nhân viên. Nếu có thao tác chỉnh sửa mã nhân viên, hiện ra câu thông báo: “Không thể chỉnh sửa mã nhân viên” và hủy thao tác cập nhật trước đó.

```
CREATE TRIGGER tgCapNhatNhanVien3
ON NHANVIEN
FOR UPDATE
AS
BEGIN
    IF UPDATE (MaNV)
    BEGIN
        RAISERROR (N'Không được sửa mã nhân viên',10,1)
        ROLLBACK TRANSACTION
    END
END
```

Sau khi tạo trigger, nếu thực hiện thay đổi mã nhân viên bằng câu lệnh sau:

```
UPDATE NHANVIEN SET MaNV = '777' WHERE MaNV = '666'
```

thì sẽ hiện ra thông báo:

```
Không được sửa mã nhân viên
```

và mã nhân viên sẽ không được cập nhật.

Lưu ý: không nên đặt câu lệnh ROLLBACK TRANSACTION trong trigger nếu mục tiêu là hoàn tất giao tác trong tất cả các trường hợp.

9.3. THAY ĐỔI NỘI DUNG VÀ XÓA TRIGGER

9.3.1. 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.

9.3.2. Làm cho Trigger mất hiệu lực/có hiệu lực

a. 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
```

b. 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
```

9.3.3. 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 CHƯƠNG

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: $YEAR(NgayBatDau) - YEAR(NgSinh) \geq 18$

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.

TÀI LIỆU THAM KHẢO

- [1] Trần Xuân Hải, Nguyễn Tiến Dũng, *Giáo trình SQL Server 2005*, NXB ĐH Quốc gia TP Hồ Chí Minh, 2009
- [2] *Giáo trình SQL Server 2008*, Trung tâm tin học Nhất Nghệ, TP. Hồ Chí Minh
- [3] Mike Hoteck, *Microsoft SQL Server 2008 Step by Step*, Microsoft Press, 2008
- [4] Hà Văn Lâm, Bài giảng Hệ quản trị CSDL SQL Server, Trường ĐH Phạm Văn Đồng, 2013

MỤC LỤC

LỜI NÓI ĐẦU	1
Chương 1: TỔNG QUAN VỀ SQL SERVER.....	2
1.1. MÔ HÌNH CLIENT/SERVER.....	2
1.2. CƠ SỞ DỮ LIỆU VÀ HỆ QUẢN TRỊ CƠ SỞ DỮ LIỆU	3
1.2.1. Cơ sở dữ liệu	3
1.2.2. Hệ quản trị cơ sở dữ liệu	4
1.3. HỆ QUẢN TRỊ CSDL MICROSOFT SQL SERVER	4
1.4. CÁC THÀNH PHẦN TRONG SQL SERVER.....	5
1.5. CÁC TIỆN ÍCH TRONG SQL SERVER.....	6
1.6. SỬ DỤNG TIỆN ÍCH SQL SERVER MANAGEMENT STUDIO	7
BÀI TẬP KẾT THÚC CHƯƠNG	11
Chương 2: CÁC THAO TÁC VỚI CƠ SỞ DỮ LIỆU	12
2.1. CƠ SỞ DỮ LIỆU TRONG SQL SERVER	12
2.1.1. Tổng quan.....	12
2.1.2. Các CSDL hệ thống	12
2.1.3. CSDL do người dùng tự định nghĩa.....	13
2.2. TẠO CƠ SỞ DỮ LIỆU.....	15
2.2.1. Các thuộc tính của một cơ sở dữ liệu trong Microsoft SQL Server	15
2.2.2. Tạo CSDL trong SQL Server.....	16
2.3. GẮN (ATTACH) CƠ SỞ DỮ LIỆU VÀO SQL SERVER.....	18
2.4. XÓA CƠ SỞ DỮ LIỆU	20
2.5. GỠ (DETACH) CƠ SỞ DỮ LIỆU KHỎI SQL SERVER	21
BÀI TẬP KẾT THÚC CHƯƠNG	22

Chương 3: TẠO VÀ QUẢN LÝ TABLE (BẢNG).....	24
3.1. KHÁI NIỆM TABLE (BẢNG).....	24
3.1.1. Cấu trúc Table	24
3.1.2. Khóa chính, khóa ngoại	24
3.2. CÁC KIỂU DỮ LIỆU TRONG SQL SERVER	25
3.2.1. Kiểu dữ liệu số chính xác (Exact Numerics)	25
3.2.2. Kiểu dữ liệu số xấp xỉ (Approximate Numerics).....	26
3.2.3. Kiểu dữ liệu ngày giờ (Date and Time)	26
3.2.4. Kiểu dữ liệu chuỗi ký tự.....	26
3.3. TẠO TABLE.....	27
3.4. CHỈNH SỬA CẤU TRÚC TABLE.....	31
3.5. XÓA TABLE	32
3.6. TẠO Ràng BUỘC DỮ LIỆU TRONG TABLE	33
BÀI TẬP KẾT THÚC CHƯƠNG	35
Chương 4: BIỂU ĐỒ CƠ SỞ DỮ LIỆU (DATABASE DIAGRAM)	39
4.1. KHÁI NIỆM BIỂU ĐỒ CƠ SỞ DỮ LIỆU.....	39
4.2. TẠO BIỂU ĐỒ CƠ SỞ DỮ LIỆU.....	39
4.3. QUẢN LÝ CÁC ĐỐI TƯỢNG TRONG BIỂU ĐỒ	42
4.3.1. Sửa đổi cấu trúc bảng hiện có	42
4.3.2. Tạo mới, hủy bỏ bảng trong CSDL.....	43
4.3.3. Chèn, xóa bảng trong mô hình CSDL.....	43
4.3.4. Thay đổi cách trình bày.....	43
4.3.5. In mô hình quan hệ dữ liệu	44
BÀI TẬP KẾT THÚC CHƯƠNG	44

Chương 5: NGÔN NGỮ TRUY VẤN DỮ LIỆU T-SQL	45
5.1. KHÁI NIỆM CƠ BẢN VỀ T-SQL	45
5.2. CÁC CÂU LỆNH ĐỊNH NGHĨA DỮ LIỆU	46
5.2.1. Câu lệnh tạo bảng.....	46
5.2.2. Câu lệnh thay đổi cấu trúc bảng.....	46
5.2.3. Câu lệnh xóa bảng.....	46
5.3. CÁC CÂU LỆNH THAO TÁC VỚI DỮ LIỆU	46
5.3.1. Câu lệnh nhập dữ liệu vào bảng.....	46
5.3.2. Câu lệnh xóa dữ liệu ra khỏi bảng	47
5.3.3. Câu lệnh chỉnh sửa dữ liệu trong bảng.....	48
5.4. CÂU LỆNH TRUY VẤN DỮ LIỆU	48
5.4.1. Cú pháp chung.....	49
5.4.2. Truy vấn thông tin từ nhiều bảng.....	54
5.4.3. Gom nhóm dữ liệu	55
5.4.4. Câu lệnh truy vấn lồng nhau	56
5.5. CÁC TOÁN TỬ TRONG T-SQL.....	57
5.5.1. Toán tử số học	57
5.5.2. Toán tử logic	57
5.5.3. Toán tử so sánh	57
5.5.4. Toán tử tập hợp	58
5.6. CÁC HÀM THƯỜNG DÙNG TRONG T-SQL	58
5.6.1. Các hàm tính toán theo nhóm	58
5.6.2. Các hàm chuyển đổi kiểu dữ liệu.....	59
5.6.3. Các hàm ngày giờ.....	60

5.6.4. Các hàm toán học	61
5.6.4. Các hàm xử lý chuỗi	62
5.7. BIẾN TRONG NGÔN NGỮ T-SQL	63
5.7.1. Biến cục bộ và biến hệ thống	63
5.7.2. Làm việc với biến cục bộ	64
5.8. CÁC CẤU TRÚC ĐIỀU KHIỂN VÀ CÂU LỆNH TRONG T-SQL	64
5.8.1. Cấu trúc rẽ nhánh IF...ELSE...	64
5.8.2. Cấu trúc lặp WHILE	66
5.8.3. Câu lệnh GOTO	66
5.8.4. Câu lệnh RETURN	67
BÀI TẬP KẾT THÚC CHƯƠNG	67
Chương 6: VIEW (KHUNG NHÌN)	72
6.1. TỔNG QUAN VỀ VIEW	72
6.1.1. Khái niệm View	72
6.1.2. Lợi ích của việc sử dụng View.....	72
6.2. TẠO VIEW	72
6.3. XEM VÀ CẬP NHẬT DỮ LIỆU THÔNG QUA VIEW.....	76
6.4. THAY ĐỔI ĐỊNH NGHĨA VÀ HỦY BỎ VIEW	77
6.4.1. Thay đổi định nghĩa View.....	77
6.4.2. Hủy bỏ View	78
BÀI TẬP KẾT THÚC CHƯƠNG	78
Chương 7: STORED PROCEDURE (THỦ TỤC LƯU TRỮ).....	81
7.1. TỔNG QUAN VỀ STORED PROCEDURE	81
7.1.1. Khái niệm Stored Procedured	81

7.1.2. Các Stored Procedure hệ thống	82
7.1.3. Lợi ích của việc sử dụng Stored Procedure	82
7.2. TẠO VÀ THỰC THI STORED PROCEDURE.....	83
7.2.1. Tạo Stored Procedure	83
7.2.2. Thực thi Stored Procedure	84
7.2.3. Tham số trong Stored Procedure	84
7.3. THAY ĐỔI NỘI DUNG VÀ HỦY STORED PROCEDURE	87
7.3.1. Thay đổi nội dung Stored Procedure.....	87
7.3.2. Hủy Stored Procedure	87
BÀI TẬP KẾT THÚC CHƯƠNG	87
Chương 8: FUNCTION (HÀM)	89
8.1. TỔNG QUAN VỀ FUNCTION	89
8.2. TẠO VÀ GỌI THỰC HIỆN FUNCTION.....	90
8.2.1. Scalar Function.....	90
8.2.2. Inline Table Function	91
8.2.3. Multistatement Table Function	92
8.3. THAY ĐỔI NỘI DUNG VÀ XÓA FUNCTION	93
8.3.1. Thay đổi nội dung Function	93
8.3.2. Xóa Function.....	93
BÀI TẬP KẾT THÚC CHƯƠNG	94
Chương 9: TRIGGER	96
9.1. TỔNG QUAN VỀ TRIGGER	96
9.1.1. Khái niệm Trigger	96
9.1.2. Các trường hợp nên dùng Trigger.....	96

9.1.3. Cơ chế hoạt động của Trigger	96
9.2. TẠO TRIGGER	97
9.2.1. Cú pháp	97
9.2.2. Các lệnh, biến hệ thống và các hàm thường sử dụng trong trigger	98
9.3. THAY ĐỔI NỘI DUNG VÀ XÓA TRIGGER	100
9.3.1. Thay đổi nội dung Trigger	100
9.3.2. Làm cho Trigger mất hiệu lực/có hiệu lực.....	101
9.3.3. Xóa Trigger	102
BÀI TẬP KẾT THÚC CHƯƠNG	102
TÀI LIỆU THAM KHẢO.....	104