# GIỚI THIỆU

## Cơ sở dữ liệu (CSDL) là gì?

Bao gồm tập các dữ liệu được tổ chức/lưu trữ theo một lược đồ dữ liệu, sao cho nó có thể dễ dàng được quản trị và dễ dàng rút trích thông tin để trả lời các câu hỏi của người dùng.

**Database = Lược đồ dữ liệu + Dữ liệu**

## Hệ quản trị Cơ sở dữ liệu (HQT CSDL) là gì?

Database Management System (DBMS) là tập các chương trình máy tính nhằm:

* Quản lý cơ sở dữ liệu (tạo mới/xóa/thay đổi/rút trích lược đồ dữ liệu hoặc dữ liệu).
* Quản lý người sử dụng và quyền truy cập tới lược đồ dữ liệu/dữ liệu.
* Cung cấp một tập các câu lệnh (ngôn ngữ truy vấn SQL) để thực hiện các chức năng ở trên.

## Mô hình CSDL là gì?

Một mô hình CSDL là một lý thuyết mô tả cách biểu diễn, tổ chức và truy cập cơ sở dữ liệu.

Một số ví dụ về mô hình CSDL:

* **Mô hình quan hệ (Relational Model):** biểu diễn và tổ chức CSDL bằng các bảng (bảng còn có tên là quan hệ). SQL Server sử dụng mô hình cơ sở dữ liệu này.
* **Mô hình thực thể - liên kết (Entity-Relationship Model):** biểu điễn, tổ chức CSDL bằng các thực thể, quan hệ giữa các thực thể (thường được sử dụng trong giai đoạn thiết kế).
* **Mô hình mạng**.

## Lược đồ dữ liệu (Data Schema) là gì?

Lược đồ dữ liệu là các mô tả cấu trúc dữ liệu của CSDL trong một mô hình cụ thể.

Ví dụ: Lược đồ dữ liệu trong mô hình quan hệ:

* *Student (StudentID, FullName, Birthday, DepartmentID)*
* *Department (DepartmentID, Name, Addess)*

## Các thuật ngữ sử dụng

* **Thực thể (Entity)**: Là các bảng (hay còn gọi là các quan hệ).
* **Thuộc tính/ Trường (Attribute/ Field)**: Các thông tin cần lưu trữ liên quan đến một thực thể.
* **Bản ghi/ Bộ/ Dòng (Record/ Tuple/ Row)**: Một tập các giá trị cho mỗi thuộc tính của một thực thể cụ thể.
* **Khóa**: Một hoặc một tập thuộc tính dùng để nhận diện một bản ghi/bộ.
  + **Khóa chính (Primary Key)**: Khóa được sử dụng để nhận diện một bản ghi/bộ.
  + **Khóa ngoại (Foreign Key)**: Là một thuộc tính/trường trong một bảng mà là khóa chính trong một bảng khác.
* **Quan hệ (Relationship)**: Liên kết giữa 2 hay nhiều thực thể.

Ví dụ: Cho bảng Student

|  |  |  |  |
| --- | --- | --- | --- |
| *StudentID* | *FullName* | *Birthday* | *DepartmentID* |
| 1234567 | Nguyễn Thị B | 21/2/1998 | CNTT |
| … | … | … | … |

🡪 Thực thể là *Student*.

🡪 Thuộc tính là:

|  |  |  |  |
| --- | --- | --- | --- |
| *StudentID* | *FullName* | *Birthday* | *DepartmentID* |

🡪 Bản ghi là:

|  |  |  |  |
| --- | --- | --- | --- |
| *1234567* | *Nguyễn Thị B* | *21/2/1998* | *CNTT* |

🡪 Khóa chính là: *StudentID*; Khóa ngoại là *DepartmentID* (Tham chiếu đến bảng Department (DepartmentID)).

🡪 Quan hệ giữa bảng Student và Department là quan hệ Nhiều – Một (tức là 1 khoa có nhiều sinh viên)

## Cơ sở dữ liệu quan hệ là gì?

* Cơ sở dữ liệu quan hệ: Một tập các quan hệ (bảng).
* Một quan hệ:
  + Lược đồ quan hệ: *Student (StudentID: char(7), FullName: nvarchar(100), Birthday: DateTime, DepartmentID: char(4))*
  + Dữ liệu:

|  |  |  |  |
| --- | --- | --- | --- |
| *StudentID* | *FullName* | *Birthday* | *DepartmentID* |
| 1234567 | Nguyễn Thị B | 21/2/1998 | CNTT |
| … | … | … | … |

## Ngôn ngữ SQL

Structured Query Language (SQL) là một ngôn ngữ dành cho hệ quản trị cơ sở dữ liệu quan hệ và thao tác dữ liệu.

* DML là viết tắt của Ngôn ngữ Thao tác Dữ liệu (Data Manipulation Language): SELECT, INSERT, UPDATE và DELETE.
* DDL là viết tắt của Ngôn ngữ Định nghĩa Dữ liệu (Data Definition Language): CREATE, ALTER, DROP, RENAME,…

## Ví dụ HQT CSDL

MySQL, PostgreSQL, Microsoft Access, SQL Server, FileMaker, Oracle, RDBMS, dBASE, Clipper, FoxPro,…

## Các cơ sở dữ liệu hệ thống

* **Master**: Lưu trữ các thông tin login account, cấu hình hệ thống, thông tin quản trị các CSDL, là CSDL quan trọng nên thường được sao lưu để bảo đảm an toàn cho hệ thống.
* **Tempdb**: Chứa các table tạm thời và các thủ tục được lưu trữ tạm thời. Các table và thủ tục nói trên được lưu trữ trong CSDL này phục vụ cho các user.
* **Model**: Được sử dụng khi template được sử dụng cho các CSDL được tạo trên một hệ thống.
* **Msdb**: Sử dụng bởi SQL Agent.

## Các tập tin của CSDL

* Primary data file: Là file chính lưu trữ dữ liệu (\*.mdf = Master Data File), mỗi CSDL có một file primary, lưu trữ điểm bắt đầu của một CSDL và các điểm kết nối đến các file lưu trữ tiếp theo (sencondary).
* Secondary data file: Là tập tin lưu trữ dữ liệu sau Primary data file (\*.ndf), một CSDL có thể có nhiều tập tin sencondary. Loại tập tin này cho phép một CSDL có thể phân tán dữ liệu ở nhiều nơi trên máy tính hoặc trên mạng.
* Log file: Là loại tập tin lưu trữ thông tin nhật ký của CSDL (\*.ldf).

## Chuẩn hóa cơ sở dữ liệu (Database Normalization)

**\* Một cơ sở dữ liệu cần được chuẩn tắc khi:**

* Dữ liệu lớn, phân tán.
* Không xác định rõ nhóm dữ liệu.
* Dữ liệu phức tạp.
* Bước đầu tiên khi xây dựng ứng dụng.

**\* Khi nào không cần chuẩn hóa:**

Một số tình huống sẽ không cần chuẩn tắc hóa, nếu như thiết kế theo chuẩn thì việc đưa ra một mẫu tin truy vấn có thể phải thực hiện truy xuất từ nhiều bảng với nhau, điều này có nghĩa ta phải thực hiện kết hợp các bảng với nhau (tuy theo luật) nên thời gian truy xuất có thể rất lớn mà yêu cầu thực tế đặt ra trong tình huống này là phải nhanh, thì truy xuất theo một bảng đã có sẵn là nhanh hơn, sau đây là một số trường hợp không cần chuẩn tác hóa (tùy theo tình huống):

* Thông tin tính toán.
* Thông tin sự kiện.
* Sự phân hoạch.

## Các loại chuẩn hóa

* 1st Normal Form: Một bảng (quan hệ) được gọi là ở dạng chuẩn 1NF nếu và chỉ nếu toàn bộ các miền giá trị của các cột có mặt trong bảng (quan hệ) đều chỉ chứa các giá trị nguyên tử (nguyên tố).
* 2nd Normal Form: Định nghĩa Một quan hệ ở dạng chuẩn 2NF nếu quan hệ đó:
  + Là 1NF;
  + Các thuộc tính không khoá phải phụ thuộc hàm đầy đủ vào khoá chính.
* 3rd Normal Form: Định nghĩa Một quan hệ ở dạng chuẩn 3NF nếu quan hệ đó:
  + Là 2NF;
  + Các thuộc tính không khoá phải phụ thuộc trực tiếp vào khoá chính.
* BCNF: Định nghĩa Một quan hệ ở dạng chuẩn BCNF nếu quan hệ đó:
  + Là 3NF;
  + Không có thuộc tính khoá mà phụ thuộc hàm vào thuộc tính không khoá.
* 4th Normal Form

# NGÔN NGỮ ĐỊNH NGHĨA DỮ LIỆU

## Các đối tượng của CSDL

* Các bảng dữ liệu (Tables)
* Các chỉ mục (Index)
* Các khung nhìn (Views)
* Các thủ tục lưu trú (Stored Procedures)
* Các bẫy lỗi (Triggers)

## Tạo lập cơ sở dữ liệu

CREATE DATABASE QLNV;

Hoặc:

CREATE DATABASE QLNV

ON

(

NAME = QLNV,

FILENAME = 'D:\OneDrive\Thang\HOCTAP\2019-2020\HKI\HQTCSDL\Lab\QLNV.MDF',

SIZE = 10 MB,

MAXSIZE = 50 MB,

FILEGROWTH = 5 MB

)

LOG ON

(

NAME = QLNV\_LOG,

FILENAME = 'D:\OneDrive\Thang\HOCTAP\2019-2020\HKI\HQTCSDL\Lab\QLNV\_LOG.LDF',

SIZE = 5 MB,

MAXSIZE = 25 MB,

FILEGROWTH = 5 MB

)

GO

**\* Lưu ý:**

* Tên cơ sở dữ liệu là tên duy nhất trong SQL Server, tối đa 128 kí tự.
* On … : Xác định các tệp (primary file, secondary file) và nhóm tệp lưu trữ CSDL.
* Log On … : Xác định các log file, nếu mệnh đề này không có thì một log file sẽ tự động được tạo ra.

## Chỉnh sửa cơ sở dữ liệu

ALTER DATABASE QLNV

ADD FILE

(

NAME = QLNV,

FILENAME = 'D:\OneDrive\Thang\HOCTAP\2019-2020\HKI\HQTCSDL\Lab\QLNV.MDF',

SIZE = 5MB,

MAXSIZE = 100MB,

FILEGROWTH = 5MB

)

GO

## Xem thông tin cơ sở dữ liệu

exec sp\_helpdb qlnv

exec sp\_helpdb

## Xóa cơ sở dữ liệu

DROP DATABASE qlnv

## Kiểu dữ liệu hệ thống

* **Exact numeric**:

|  |  |  |
| --- | --- | --- |
| **DATA TYPE** | **FROM** | **TO** |
| bigint | -9,223,372,036,854,775,808 | 9,223,372,036,854,775,807 |
| int | -2,147,483,648 | 2,147,483,647 |
| smallint | -32,768 | 32,767 |
| tinyint | 0 | 255 |
| bit | 0 | 1 |
| decimal | -10^38 +1 | 10^38 -1 |
| numeric | -10^38 +1 | 10^38 -1 |
| money | -922,337,203,685,477.5808 | +922,337,203,685,477.5807 |
| smallmoney | -214,748.3648 | +214,748.3647 |

* **Approximate numeric**:

|  |  |  |
| --- | --- | --- |
| **DATA TYPE** | **FROM** | **TO** |
| float | -1.79E + 308 | 1.79E + 308 |
| real | -3.40E + 38 | 3.40E + 38 |

* **Date & Time**:

|  |  |  |
| --- | --- | --- |
| **DATA TYPE** | **FROM** | **TO** |
| datetime | Jan 1, 1753 | Dec 31, 9999 |
| smalldatetime | Jan 1, 1900 | Jun 6, 2079 |
| date | Stores a date like June 30, 1991 | |
| time | Stores a time of day like 12:30 P.M. | |

* **Character string**:

|  |  |
| --- | --- |
| **Sr.No.** | **DATA TYPE & Description** |
| 1 | **char**  Maximum length of 8,000 characters.( Fixed length non-Unicode characters) |
| 2 | **varchar**  Maximum of 8,000 characters.(Variable-length non-Unicode data). |
| 3 | **varchar(max)**  Maximum length of 2E + 31 characters, Variable-length non-Unicode data (SQL Server 2005 only). |
| 4 | **text**  Variable-length non-Unicode data with a maximum length of 2,147,483,647 characters. |

* **Unicode character string**:

|  |  |
| --- | --- |
| **Sr.No.** | **DATA TYPE & Description** |
| 1 | **nchar**  Maximum length of 4,000 characters.( Fixed length Unicode) |
| 2 | **nvarchar**  Maximum length of 4,000 characters.(Variable length Unicode) |
| 3 | **nvarchar(max)**  Maximum length of 2E + 31 characters (SQL Server 2005 only).( Variable length Unicode) |
| 4 | **ntext**  Maximum length of 1,073,741,823 characters. ( Variable length Unicode ) |

* **Binary character string**:

|  |  |
| --- | --- |
| **Sr.No.** | **DATA TYPE & Description** |
| 1 | **binary**  Maximum length of 8,000 bytes(Fixed-length binary data ) |
| 2 | **varbinary**  Maximum length of 8,000 bytes.(Variable length binary data) |
| 3 | **varbinary(max)**  Maximum length of 2E + 31 bytes (SQL Server 2005 only). ( Variable length Binary data) |
| 4 | **image**  Maximum length of 2,147,483,647 bytes. ( Variable length Binary Data) |

* **Khác**:

|  |  |
| --- | --- |
| **Sr.No.** | **DATA TYPE & Description** |
| 1 | **sql\_variant**  Stores values of various SQL Server-supported data types, except text, ntext, and timestamp. |
| 2 | **timestamp**  Stores a database-wide unique number that gets updated every time a row gets updated |
| 3 | **uniqueidentifier**  Stores a globally unique identifier (GUID) |
| 4 | **xml**  Stores XML data. You can store xml instances in a column or a variable (SQL Server 2005 only). |
| 5 | **cursor**  Reference to a cursor object |
| 6 | **table**  Stores a result set for later processing |

Ngoài các kiểu dữ liệu hệ thống ra thì còn có: **User-defined data type, Spatial data, Filestreams**.

## Toán tử số học và Toán tử so sánh

Giả sử a = 10, b = 20, ta có:

|  |  |  |
| --- | --- | --- |
| **Operator** | **Description** | **Example** |
| + (Addition) | Adds values on either side of the operator. | a + b will give 30 |
| - (Subtraction) | Subtracts right hand operand from left hand operand. | a - b will give -10 |
| \* (Multiplication) | Multiplies values on either side of the operator. | a \* b will give 200 |
| / (Division) | Divides left hand operand by right hand operand. | b / a will give 2 |
| % (Modulus) | Divides left hand operand by right hand operand and returns remainder. | b % a will give 0 |

|  |  |  |
| --- | --- | --- |
| **Operator** | **Description** | **Example** |
| = | Checks if the values of two operands are equal or not, if yes then condition becomes true. | (a = b) is not true. |
| != | Checks if the values of two operands are equal or not, if values are not equal then condition becomes true. | (a != b) is true. |
| <> | Checks if the values of two operands are equal or not, if values are not equal then condition becomes true. | (a <> b) is true. |
| > | Checks if the value of left operand is greater than the value of right operand, if yes then condition becomes true. | (a > b) is not true. |
| < | Checks if the value of left operand is less than the value of right operand, if yes then condition becomes true. | (a < b) is true. |
| >= | Checks if the value of left operand is greater than or equal to the value of right operand, if yes then condition becomes true. | (a >= b) is not true. |
| <= | Checks if the value of left operand is less than or equal to the value of right operand, if yes then condition becomes true. | (a <= b) is true. |
| !< | Checks if the value of left operand is not less than the value of right operand, if yes then condition becomes true. | (a !< b) is false. |
| !> | Checks if the value of left operand is not greater than the value of right operand, if yes then condition becomes true. | (a !> b) is true. |

|  |  |
| --- | --- |
| **Sr.No.** | **Operator & Description** |
| 1 | **ALL**  The ALL operator is used to compare a value to all values in another value set. |
| 2 | **AND**  The AND operator allows the existence of multiple conditions in an SQL statement's WHERE clause. |
| 3 | **ANY**  The ANY operator is used to compare a value to any applicable value in the list as per the condition. |
| 4 | **BETWEEN**  The BETWEEN operator is used to search for values that are within a set of values, given the minimum value and the maximum value. |
| 5 | **EXISTS**  The EXISTS operator is used to search for the presence of a row in a specified table that meets a certain criterion. |
| 6 | **IN**  The IN operator is used to compare a value to a list of literal values that have been specified. |
| 7 | **LIKE**  The LIKE operator is used to compare a value to similar values using wildcard operators. |
| 8 | **NOT**  The NOT operator reverses the meaning of the logical operator with which it is used. Eg: NOT EXISTS, NOT BETWEEN, NOT IN, etc. **This is a negate operator.** |
| 9 | **OR**  The OR operator is used to combine multiple conditions in an SQL statement's WHERE clause. |
| 10 | **IS NULL**  The NULL operator is used to compare a value with a NULL value. |
| 11 | **UNIQUE**  The UNIQUE operator searches every row of a specified table for uniqueness (no duplicates) |

## Tạo bảng

USE QLNV

GO

CREATE TABLE PHONG

(

MAPHONG CHAR(3),

TENPHONG NVARCHAR(40),

DIACHI NVARCHAR(50),

TEL CHAR(10),

CONSTRAINT [PK\_MAPHONG] PRIMARY KEY (MAPHONG)

)

GO

## Sửa đổi bảng (Thêm, xóa, sửa cột)

ALTER TABLE PHONG

ADD UNIQUE (TENPHONG)

GO

## Xem thông tin bảng

exec sp\_help 'phong'

Hoặc:

exec sp\_help phong

## Xóa bảng

drop table PHONG

# NGÔN NGỮ THAO TÁC DỮ LIỆU

## Chèn thêm bản ghi

INSERT INTO Khoa VALUES ('CNTT', N'Công nghệ thông tin', N'Nhà C', '37547100')

Hoặc:

INSERT INTO Khoa (MaKhoa,TenKhoa,DiaChi) VALUES ('DIAL', N'Địa lý', N'Nhà A1')

## Cập nhật bản ghi

UPDATE Khoa

SET TenKhoa = N'Công nghệ sinh học'

WHERE MaKhoa = 'CNSH'

## Xóa bản ghi

DELETE FROM Khoa

WHERE MaKhoa = 'CNSH'

## Xóa tất cả các bản ghi nhưng vẫn giữ lược đồ

TRUNCATE TABLE Khoa

## Truy vấn

SELECT [ALL | DISTINCT] [TOP (giá\_trị\_đầu) [PERCENT] [WITH TIES]] Biểu\_thức

FROM bảng

[WHERE điều\_kiện]

[GROUP BY biểu\_thức]

[HAVING điều\_kiện]

[ORDER BY biểu\_thức [ASC | DESC]];

Nghĩa là:

Lấy thuộc\_tính\_nào  
Từ bảng\_hoặc\_nhiều\_bảng\_nào  
[Thoả mãn điều\_kiện\_gì]

[Nhóm theo thuộc\_tính\_gì]

[Sắp xếp theo thuộc\_tính [Tăng | Giảm]]

## Một số ví dụ về truy vấn

* *Cho biết thông tin về các phòng ban?*

SELECT \* FROM Phong

* *Cho biết danh sách thưởng của các nhân viên bao gồm các thông tin mã nhân viên, họ tên, lương, và thưởng (bằng lương + 40% lương)?*

SELECT Manv, Hoten, Luong, 1.4 \* Luong as Thuong FROM Nhanvien

* *Cho biết mã các ngoại ngữ (không bị trùng) mà các nhân viên trong cơ quan đã học?*

Select Distinct Mann From TDNN

* *Đưa ra thông tin về 2 nhân viên đầu tiên trong bảng nhân viên?*

Select top 2\* From NHANVIEN

* *Cho biết danh sách các nhân viên làm việc ở phòng có mã số là HCA?*

Select \*

From Nhanvien

Where Maphong = 'HCA'

* *Cho biết mã nhân viên, họ tên, ngày sinh, lương của các nhân viên có lương nằm trong khoảng từ 2.000.000 đồng đến 3.000.000 đồng?*

Select Manv, Hoten, Ngaysinh, Luong

From Nhanvien

Where (Luong >= 2000000) and (Luong <= 3000000)

Hoặc:

Select Manv, Hoten, Ngaysinh, Luong

From Nhanvien

Where Luong Between 2000000 and 3000000

* *Cho biết các nhân viên làm việc ở phòng có mã phòng là hca hoặc KDA?*

Select \* From Nhanvien

Where Maphong = 'hca' or Maphong = 'KDA'

Hoặc:

Select \* From Nhanvien

Where Maphong in ('hca', 'KDA')

* *Cho biết danh sách các nhân viên có tên bắt đầu bằng ‘N’?*

Select \*

From Nhanvien

Where HoTen like 'N%'

* *Hãy đưa ra danh sách nhân viên theo theo chiều tăng dần của lương?*

Select Manv, Hoten, Ngaysinh, Luong

From NHANVIEN

Order By Luong Asc

* *Hãy đưa ra danh sách nhân viên theo theo chiều tăng dần của lương, nếu 2 nhân viên có cùng lương thì sắp xếp theo ngày sinh giảm dần?*

Select Manv, Hoten, Ngaysinh, Luong

From NHANVIEN

Order By Luong Asc, Ngaysinh Desc

* *Cho biết lương trung bình của phòng có mã KDA?*

Select Avg(Luong) as TBluong

From NHANVIEN

Where Maphong= 'KDA'

* *Cho biết trung bình lương của mỗi phòng?*

Select Maphong, avg(Luong) as TBLuong

From NHANVIEN

Group By Maphong

* *Cho biết các phòng có trung bình lương lớn hơn 2250000?*

Select Maphong, avg(Luong) as TBLuong

From NHANVIEN

Group By Maphong

Having avg(Luong) >=2250000

## Một số hàm xử lý

* ASCII(s): trả về giá trị mã ASCII của kí tự bên nhất trái của xâu
* Char(n): trả ra kí tự thứ n trong bảng mã ASCII
* Upper(s): chuyển đổi xâu sang chữ hoa
* Lower(s): chuyển đổi xâu sang chữ thường
* +: phép toán ghép các xâu
* Len(s): trả về độ dài của xâu
* LTRIM(s): loại bỏ các kí tự trắng bên trái xâu
* RTRIM(s): loại bỏ các kí tự trắng bên phải xâu
* Left(s, n): trả về n kí tự (xâu con) phía bên trái nhất xâu s
* Right(s, n): trả về n kí tự (xâu con) phía bên phải nhất xâu s
* SubString(s, m ,n): trả về xâu con dài n kí tự của xâu s bắt đầu từ ví trí thứ m
* Replace(s, s1, s2): thay thế các xâu con s1 xuất hiện trong xâu s bằng xâu s2
* Stuff (s, m, n, s1): xóa xâu con n kí tự bắt đầu từ vị trí thứ m trong xâu s và thay thế vào đó xâu s1
* Reverse(s): trả về xâu đảo ngược của xâu s
* Getdate(): trả về ngày tháng năm của hệ thống
* DatePart(tham số datepart,date): trả về giá trị nguyên là thành phần của giá trị date
* DateDiff(datepart, date1,date2): trả về khoảng ngày, tháng, năm,... giữa 2 ngày date1 và date2
* Dateadd(datepart,n,date): thêm một số n datepart vào date
* Day(d): trả về giá trị số nguyên của ngày
* Month(d): trả về giá trị số nguyên của tháng
* Year(d): trả về giá trị số nguyên của năm
* Square(n)
* Sqrt(n)
* Round(f,n)
* Power(f,n)
* Pi()
* Sign(f)

## Truy vấn liên kết

* Liên kết trong (inner join)

Select manv, hoten, nv.maphong, tenphong

From nhanvien nv

Inner join phong p

On nv.maphong=p.maphong

Hoặc:

Select manv, hoten, nhanvien.maphong, tenphong

From Phong, nhanvien

Where phong.maphong=nhanvien.maphong

* Liên kết ngoài (outer join)

SELECT \*

FROM Nhanvien nv

LEFT OUTER JOIN phong p

ON nv.maphong = p.maphong

SELECT \*

FROM Nhanvien nv

RIGHT OUTER JOIN phong p

ON nv.maphong = p.maphong

* Liên kết chéo (cross join)
* Tự liên kết (self\_join)

## Các loại ràng buộc dữ liệu

* **Ràng buộc khai báo:**
  + **Ràng buộc cột (miền giá trị - Domain integrity):** xác định miền giá trị mà thuộc tính sẽ nhận được.

Ví dụ: NOT NULL, Default giá\_trị, Unique, Check

* + **Ràng buộc về tính toàn vẹn của thực thể (**Entry integrity): chỉ rõ khóa của bảng

Ví dụ: Primary key

* + **Ràng buộc toàn vẹn tham chiếu (Referential integrity)**: liên quan đến việc quản lý các mối quan hệ giữa các bản ghi của hai bảng (mối quan hệ giữa khóa chính và khóa phụ).

Ví dụ: Foreign Key

* **Ràng buộc do người dùng định nghĩa** (User-defined integrity) **hay Ràng buộc phức tạp:** sự thay đổi dữ liệu của một bảng 🡪 sự thay đổi dữ liệu của một loạt các bảng khác mà phải thực hiện theo một giải thuật (trigger, stored procedure).

## So sánh hàm (UDF) và thủ tục (SP)

|  |  |
| --- | --- |
| **Hàm (UDF)** | **Thủ tục (SP)** |
| Giá trị các tham số không được truyền ra ngoài. | Có thể đưa giá trị của tham số ra ngoài bằng thuộc tính OUTPUT |
| Trả ra một giá trị bằng mệnh đề RETURN |  |
| Có thể trả về một giá trị vô hướng hoặc một bảng dữ liệu. | Chỉ trả về kiểu dữ liệu giá trị kiểu nguyên |

# TOÀN VẸN DỮ LIỆU

## Toàn vẹn dữ liệu là gì? Các kiểu toàn vẹn

* Tập các ràng buộc nhằm đảm bảo chất lượng dữ liệu trong các CSDL: dữ liệu là đúng đắn, tin cậy, nhất quán.
* Các kiểu toàn vẹn:

A screenshot of a cell phone

Description automatically generated

## Cài đặt tính toàn vẹn dữ liệu

* Cài đặt lúc khai báo lược đồ CSDL
  + Khai báo thuộc tính các cột: PRIMARY KEY, NOT NULL, UNIQUE, DEFAULT, FOREIGN KEY
  + Thêm vào các bảng: constraints, defaults, and rules
* Cài đặt bằng các thủ tục: scripts, triggers and stored procedures.

## Một số ví dụ về ràng buộc

* Ràng buộc khóa chính:

USE qlnv

CREATE TABLE DMNN

(

MaNN char(2) primary key,

TenNN nvarchar(20) NOT NULL

);

Hoặc:

USE qlnv

CREATE TABLE DMNN

(

MaNN char(2),

TenNN nvarchar(20) NOT NULL,

CONSTRAINT PK\_DMNN PRIMARY KEY (Mann)

);

* Ràng buộc Unique (duy nhất):

CREATE TABLE DMNN

(

MaNN char(2) primary key,

TenNN nvarchar(20) unique NONCLUSTERED

);

* Ràng buộc Check (Tính đúng đắn):

CREATE TABLE Nhanvien

(

Manv char (8) PRIMARY KEY,

HoTen nvarchar(50) NOT NULL,

Ngaysinh datetime CHECK (Ngaysinh BETWEEN '01-01-1900' AND '01-01-2012'),

MaPhong int

)

* Ràng buộc Default (Mặc định):

CREATE TABLE Phong

(

MaPhong int PRIMARY KEY,

TenPhong nvarchar(40) UNIQUE,

DiaChi nvarchar(50) DEFAULT N'123 - Đống Đa - HN',

Tel char (10)

)

* Ràng buộc khóa ngoại:

CREATE TABLE TDNN

(

MaNV char(8),

MaNN char(2),

TDo smallint,

PRIMARY KEY (MaNV, MaNN),

CONSTRAINT FK\_TDNN\_NhanVien

FOREIGN KEY (manv) REFERENCES Nhanvien(manv),

FOREIGN KEY (mann) REFERENCES DMNN(mann)

)

# CHỈ MỤC (INDEX)

## Chỉ mục là gì?

* Index được thực hiện theo cấu trúc cây B-Tree nhằm tăng tốc độ truy xuất dữ liệu đối với ổ đĩa (thiết bị thứ cấp).
* Ví dụ:

USE library

CREATE CLUSTERED INDEX cl\_lastname  
ON member(lastname)

Để xóa chỉ mục:

Drop Index Sach(sach\_idx)

## Chỉ mục kết hợp

A screenshot of a cell phone

Description automatically generated

USE library

CREATE UNIQUE INDEX loan\_ident

ON loan (isbn, copy\_no)

## Các loại chỉ mục

* **Clustered Index:** 
  + Khi khóa đặt thuộc tính Clustered, dữ liệu của bảng sẽ được sắp xếp vật lý trên đĩa, như vậy khi thiết kế khóa dạng này dữ liệu được chèn và sẽ tìm đúng vị trí trên địa để lưu trữ (vùng đĩa dành cho bảng dữ liệu), chính vì vậy mà có thể xảy ra trường hợp phải dịch chuyển danh sách các giá trị đã có ở đĩa. Những việc tạo khóa Index dạng này sẽ không cần sắp xếp giá trị ở dạng logic mà khi truy nhập đĩa đã bảo đảm dữ liệu được sắo xếp.
  + Bảng dữ liệu chỉ có thể tạo tối đa một khóa Clustered Index.
* **Nonclustered Index**: Dữ liệu Index không sắp xếp ở dạng vật lý mà chỉ sắp xếp logic, dữ liệu của bảng lưu trữ giá trị khóa Index được sắp xếp, nhanh trong nhập dữ liệu.
* **Unique Index**: Xác định dữ liệu của cột tham gia khóa Index không lặp lại.
* **Fill Factor**: Khi tạo khóa Index, dữ liệu tham gia tạo khóa Index sẽ được phân theo mức của B-Tree, các mức được phân theo page dữ liệu, giá trị Fill factor xác định phần khoảng trống tối đa của page theo tỷ lệ phần trăm. Nhờ khoảng trống này mà tốc độ bố trí cấu trúc Index, tốc độ truy lục thông tin trong cây được cải thiện

# LẬP TRÌNH T-SQL

## Bó (batch) là gì?

* Bó là một tập các câu lệnh T-SQL được biên dịch đồng thời bởi SQL Server (chú ý được biên dịch đồng thời, nếu một lệnh có lỗi cú pháp thì cả bó không biên dịch được, khác với transaction!)
* Câu lệnh **GO** đánh dấu kết thúc một bó.

## Giao dịch (Transaction) là gì?

* Nhóm các lệnh SQL thành một đơn vị thực hiện. Nếu một trong các lệnh của nhóm không thực hiện thành công thì kết quả thực hiện của tất cả các lệnh trong nhóm đều hủy.
* Cú pháp:

BEGIN TRAN

<Nhóm các lệnh của Transaction>

COMMIT TRAN

## Sử dụng Batch và Transaction

use QLSVC

GO

BEGIN TRAN

insert into Khoa values('KTHN', N'Kỹ Thuật Hạt Nhân', 'Nhà A11', '063323344')

insert into SinhVien values('1910000', N'Nguyễn Văn A', '11/11/2000', 'KTHN') insert into SinhVien values('1910001', N'Nguyễn Văn B', '11/11/2001', 'KTHN')

COMMIT TRAN

## Biến là gì?

* Biến cục bộ Transact-SQL là một đối tượng có thể chứa giá trị thuộc một kiểu dữ liệu nhất định, tên biến bắt đầu bằng một ký tự @.
* Ví dụ 1:

DECLARE @masv CHAR(8);

SET @masv = '1910001';

* Ví dụ 2:

DECLARE @sosv INT;

SELECT @sosv = COUNT(\*) FROM SinhVien;

PRINT N'Số sinh viên:' + CAST(@sosv AS CHAR(10));

* Biến hệ thống (do SQL server quản lý) có tên bắt đầu @@, là biến toàn cục.
* Một số biến hệ thống thường dùng
  + **@@error**: thông báo mã lỗi, nếu @@error = 0: thao tác thực hiện thành công.
  + **@@rowcount**: cho biết số dòng bị ảnh hưởng bởi lệnh cuối (insert, update, delete).
  + **@@trancount**: cho biết số giao dịch đang hoạt động trên kết nối hiện tại.
  + **@@fetch\_status**: cho biết thao tác lấy dữ liệu từ cursor có thành công không.

## Câu điều khiển If … else …

declare @Siso int

select @Siso = Siso from HocPhan

where MaHP = 'HP001'

if @Siso < 50

insert into DangKy values('1810222', 'HP001', 0)

print N'Đăng ký học phần thành công'

else

print N'Học phần đã đủ sinh viên đăng ký'

## Vòng lặp While

Declare @MaNS int

Set @MaNS = 1

While exists(select \* from NhanSu where MaNS = @MaNS)

set @MaNS = @MaNS + 1

Insert into NhanSu values (@MaNS, N'Nguyễn Thị Thu', 35)

## Câu điều khiển Case

* Dạng 1:

SELECT Hoten, Tennn, TrinhDo=

CASE TDo

WHEN 'A' THEN N'Yếu'

WHEN 'B' THEN N'Tạm được'

WHEN 'C' THEN N'Thành thạo'

WHEN 'D' THEN N'Rất tốt'

END

FROM Nhanvien, DMNN, TDNN

WHERE Nhanvien.manv=TDNN.manv and TDNN.mann=DMNN.mann

* Dạng 2:

SELECT Hoten, Phanloai =

CASE

WHEN Year(getdate()) - Year(Ngaysinh) < 30 THEN N'Trẻ'

WHEN Year(getdate()) - Year(Ngaysinh) > 30 and Year(getdate()) –

Year(Ngaysinh) < 40 THEN N'Trung niên'

ELSE N'Già'

END

FROM Nhanvien

## Con trỏ (Cursor) là gì?

* Là một cấu trúc dữ liệu ánh xạ đến một tập các dòng dữ liệu kết quả của một câu truy vấn (select).
* Cho phép duyệt tuần tự qua tập các dòng dữ liệu và đọc giá trị từng dòng.
* Lặp lại việc duyệt và sử dụng cursor, có thể sử dụng biến @@fetch\_status để biết đã duyệt qua hết cursor hay chưa.
  + **@@FETCH\_STATUS = 0**: lấy dữ liệu thành công.
  + **@@FETCH\_STATUS < 0**: không lấy được dữ liệu
* Ví dụ:

--Khai báo biến @id, @title để lưu nội dung đọc

DECLARE @id int

DECLARE @title nvarchar(200)

DECLARE cursorProduct CURSOR FOR -- khai báo con trỏ cursorProduct

SELECT id, title

FROM Product -- dữ liệu trỏ tới

OPEN cursorProduct -- Mở con trỏ

FETCH NEXT FROM cursorProduct -- Đọc dòng đầu tiên

INTO @id, @title

WHILE @@FETCH\_STATUS = 0 -vòng lặp WHILE khi đọc Cursor thành công

BEGIN

PRINT 'ID:' + CAST(@id as nvarchar) -- In kết quả

PRINT 'TITLE:'+ @title

FETCH NEXT FROM cursorProduct -- Đọc dòng tiếp

INTO @id, @title

END

CLOSE cursorProduct -- Đóng Cursor

DEALLOCATE cursorProduct -- Giải phóng tài nguyên

# KHUNG NHÌN (VIEW) VÀ THỦ TỤC (SP)

## Khung nhìn (View) là gì?

* View là bảng không chứa dữ liệu, nó chỉ là truy vấn kết hợp dữ liệu từ 1 hay nhiều bảng có quan hệ với nhau và được lưu thành một đối tượng của SQL Server.
* Người sử dụng có thể áp dụng ngôn ngữ thao tác dữ liệu trên các View giống như Table.
* Ví dụ:

CREATE VIEW nv\_kinhdoanh

AS

SELECT Manv, Hoten, Luong

FROM NHANVIEN

GO

SELECT \* FROM NV\_KINHDOANH

Xóa khung nhìn:

DROP VIEW nv\_kinhdoanh

## Mục đích dùng View

* Hạn chế tính phức tạp của dữ liệu đối với người sử dụng đơn giản.
* Tạo ra bảng ảo có dữ liệu theo yêu cầu cho người sử dụng và sử dụng trong thiết kế báo cáo.
* Hạn chế quyền truy cập dữ liệu của người sử dụng.
* View dùng để trình bày các thông tin dẫn xuất.

## Cập nhật dữ liệu cho View

* View định nghĩa dữ liệu trên một bảng thì có thể dùng Insert
* Nếu trong định nghĩa View có chứa mệnh đề Inner join thì không thể dùng các thao tác Insert hay Delete để thay đổi dữ liệu
* Nếu trong định nghĩa View có chứa mệnh đề With check option thì chỉ những bản ghi thỏa mãn điều kiện của View mới được Insert, Update.

## Thủ tục (Stored Procedure) là gì?

* Thủ tục được xây dựng từ các câu lệnh T-SQL và được lưu trú trên SQL server.
* Muốn thực hiện một thủ tục, người sử dụng chỉ cần thực hiện một lời gọi hàm.
* Khi SP được chạy lần đầu tiên nó sẽ được biên dịch qua 5 bước và sinh ra một mô hình truy vấn. Mô hình này sẽ được đặt trong một CSDL của SQL server, lần sau chạy lại thủ tục sẽ không phải dịch lại nữa.
* Ví dụ 1:

CREATE PROC DS\_Phong

@BienMaPhong int

AS

SELECT Manv, HoTen, NgaySinh

FROM NhanVien

WHERE MaPhong = @BienMaPhong

GO

Exec DS\_Phong 1

* Ví dụ 2:

CREATE PROC Soluong\_Phong

@BienMaPhong int,

@SL int OUTPUT

AS

SELECT @SL = COUNT (\*)

FROM NhanVien

WHERE MaPhong = @BienMaPhong

GO

# HÀM (USER DEFINED FUNCTION) VÀ TRIGGER

## Hàm trả về một số nguyên

CREATE FUNCTION fn\_Ham\_Soluong\_Phong (@BienMaPhong int)

RETURNS int

AS

BEGIN

DECLARE @SL int;

SELECT @SL = COUNT (\*)

FROM NhanVien

WHERE MaPhong=@BienMaPhong;

RETURN (@SL);

END

-- Sử dụng hàm (Hàm xuất hiện trong biểu mẫu)

SELECT dbo.fn\_Ham\_Soluong\_Phong(1);

## Hàm trả về một bảng

CREATE FUNCTION fn\_Ham\_DS\_NhanVien (@BienMaPhong int)

RETURNS @table\_KQ TABLE

(

Manv CHAR(8), HoTen nvarchar(50), gaySinh DATETIME

)

BEGIN

INSERT INTO @table\_KQ

SELECT Manv, HoTen, NgaySinh

FROM NhanVien

WHERE MaPhong = @BienMaPhong

RETURN

END

-- Sử dụng hàm trả về kết quả của một bảng

Select \*

from dbo.fn\_Ham\_DS\_NhanVien(1);

Go

Hoặc:

Create function fn\_DanhSachSVDK (@MaSV nchar(10)) returns Table

As

return

(

select sv.MaSV, sv.HoTen, dk.MaHP, hp.TenHP

From SinhVien sv, DangKy dk, HocPhan hp

Where sv.MaSV = dk.MaSV and

dk.MaHP = hp.MaHP and sv.MaSV = @MaSV

)

--Cách gọi hàm:

select \*

from dbo.fn\_DanhSachSVDK('1810278')

## Trigger là gì?

* Thủ tục lưu trú (Stored Procedure) đặc biệt
* Triger tự động được chạy mỗi khi có một hành động liên quan đến nó xảy ra
* Triger không có tham số, không trả ra giá trị như SP bình thường.
* Các loại trigger: Insert trigger, Delete trigger, Update trigger, trigger tổng hợp.

## Trigger chèn

CREATE TRIGGER trInsNV

ON NHANVIEN

FOR INSERT

AS

RAISERROR('%d hang da duoc them vao bang NHANVIEN', 0, 1, @@rowcount)

Trigger này tự động được thực hiện mỗi khi có bản ghi mới được chèn vào bảng NHANVIEN.

Ví dụ: Thêm nhân viên trong một phòng thì sẽ tăng số lượng nhân viên lên

CREATE TRIGGER trInsNV\_PHONG

ON NHANVIEN

FOR INSERT

AS

UPDATE PHONG SET SONV = SONV + 1 WHERE PHONG.MAPHONG = (SELECT MAPHONG FROM INSERTED)

## Trigger xóa

CREATE TRIGGER trDelNV

ON NHANVIEN

FOR DELETE

AS

RAISERROR('%d hàng bị xóa trong bảng NHANVIEN', 0, 1, @@rowcount)

Trigger này tự động được thực hiện mỗi khi có một hoặc nhiều bản ghi bị xóa ở bảng NHANVIEN.

Ví dụ: Xóa phòng xóa luôn nhân viên trong phòng

CREATE TRIGGER trDelPhong

ON PHONG

FOR DELETE

AS

DELETE NHANVIEN

FROM DELETED

WHERE DELETED.MAPHONG = NHANVIEN.MAPHONG

## Trigger cập nhật

CREATE TRIGGER trUpNV

ON NHANVIEN

FOR UPDATE

AS

RAISERROR('%d %d hàng đã được sửa ở bảng bảng NHANVIEN', 0, 1, @@rowcount)

Trigger này tự động được thực hiện mỗi khi có một câu lệnh Update được thực hiện trên bảng NHANVIEN.

## Trigger tổng hợp

CREATE TRIGGER trIns\_UpNV

ON NHANVIEN

FOR INSERT, UPDATE

AS

RAISERROR('%d hàng đã được thêm hoặc sửa ở bảng NHANVIEN', 0, 1, @@rowcount)

## Trigger instead of

* Cho phép cập nhật dữ liệu các bảng thông qua view có liên kết nhiều bảng.
* Ví dụ: Ta sẽ cần một trigger gán vào view vw\_brands, mỗi khi có thao tác insert vào view thì sẽ insert nó vào table brand\_approvals.

CREATE TRIGGER production.trg\_vw\_brands

ON production.vw\_brands

INSTEAD OF INSERT

AS

BEGIN

SET NOCOUNT ON;

INSERT INTO production.brand\_approvals (brand\_name)

SELECT i.brand\_name

FROM inserted i

WHERE i.brand\_name NOT IN

(

SELECT brand\_name

FROM production.brands

);

END

Contents

[I. GIỚI THIỆU 1](#_Toc25828126)

[1. Cơ sở dữ liệu (CSDL) là gì? 1](#_Toc25828127)

[2. Hệ quản trị Cơ sở dữ liệu (HQT CSDL) là gì? 1](#_Toc25828128)

[3. Mô hình CSDL là gì? 1](#_Toc25828129)

[4. Lược đồ dữ liệu (Data Schema) là gì? 1](#_Toc25828130)

[5. Các thuật ngữ sử dụng 1](#_Toc25828131)

[6. Cơ sở dữ liệu quan hệ là gì? 2](#_Toc25828132)

[7. Ngôn ngữ SQL 2](#_Toc25828133)

[8. Ví dụ HQT CSDL 2](#_Toc25828134)

[9. Các cơ sở dữ liệu hệ thống 3](#_Toc25828135)

[10. Các tập tin của CSDL 3](#_Toc25828136)

[11. Chuẩn hóa cơ sở dữ liệu (Database Normalization) 3](#_Toc25828137)

[12. Các loại chuẩn hóa 3](#_Toc25828138)

[II. NGÔN NGỮ ĐỊNH NGHĨA DỮ LIỆU 5](#_Toc25828139)

[1. Các đối tượng của CSDL 5](#_Toc25828140)

[2. Tạo lập cơ sở dữ liệu 5](#_Toc25828141)

[3. Chỉnh sửa cơ sở dữ liệu 5](#_Toc25828142)

[4. Xem thông tin cơ sở dữ liệu 6](#_Toc25828143)

[5. Xóa cơ sở dữ liệu 6](#_Toc25828144)

[6. Kiểu dữ liệu hệ thống 6](#_Toc25828145)

[7. Toán tử số học và Toán tử so sánh 8](#_Toc25828146)

[8. Tạo bảng 11](#_Toc25828147)

[9. Sửa đổi bảng (Thêm, xóa, sửa cột) 11](#_Toc25828148)

[10. Xem thông tin bảng 11](#_Toc25828149)

[11. Xóa bảng 11](#_Toc25828150)

[III. NGÔN NGỮ THAO TÁC DỮ LIỆU 12](#_Toc25828151)

[1. Chèn thêm bản ghi 12](#_Toc25828152)

[2. Cập nhật bản ghi 12](#_Toc25828153)

[3. Xóa bản ghi 12](#_Toc25828154)

[4. Xóa tất cả các bản ghi nhưng vẫn giữ lược đồ 12](#_Toc25828155)

[5. Truy vấn 12](#_Toc25828156)

[6. Một số ví dụ về truy vấn 12](#_Toc25828157)

[7. Một số hàm xử lý 14](#_Toc25828158)

[8. Truy vấn liên kết 14](#_Toc25828159)

[9. Các loại ràng buộc dữ liệu 15](#_Toc25828160)

[10. So sánh hàm (UDF) và thủ tục (SP) 15](#_Toc25828161)

[IV. TOÀN VẸN DỮ LIỆU 17](#_Toc25828162)

[1. Toàn vẹn dữ liệu là gì? Các kiểu toàn vẹn 17](#_Toc25828163)

[2. Cài đặt tính toàn vẹn dữ liệu 17](#_Toc25828164)

[3. Một số ví dụ về ràng buộc 17](#_Toc25828165)

[V. CHỈ MỤC (INDEX) 19](#_Toc25828166)

[1. Chỉ mục là gì? 19](#_Toc25828167)

[2. Chỉ mục kết hợp 19](#_Toc25828168)

[3. Các loại chỉ mục 19](#_Toc25828169)

[VI. LẬP TRÌNH T-SQL 21](#_Toc25828170)

[1. Bó (batch) là gì? 21](#_Toc25828171)

[2. Giao dịch (Transaction) là gì? 21](#_Toc25828172)

[3. Sử dụng Batch và Transaction 21](#_Toc25828173)

[4. Biến là gì? 21](#_Toc25828174)

[5. Câu điều khiển If … else … 22](#_Toc25828175)

[6. Vòng lặp While 22](#_Toc25828176)

[7. Câu điều khiển Case 22](#_Toc25828177)

[8. Con trỏ (Cursor) là gì? 23](#_Toc25828178)

[VII. KHUNG NHÌN (VIEW) VÀ THỦ TỤC (SP) 24](#_Toc25828179)

[1. Khung nhìn (View) là gì? 24](#_Toc25828180)

[2. Mục đích dùng View 24](#_Toc25828181)

[3. Cập nhật dữ liệu cho View 24](#_Toc25828182)

[4. Thủ tục (Stored Procedure) là gì? 24](#_Toc25828183)

[VIII. HÀM (USER DEFINED FUNCTION) VÀ TRIGGER 26](#_Toc25828184)

[1. Hàm trả về một số nguyên 26](#_Toc25828185)

[2. Hàm trả về một bảng 26](#_Toc25828186)

[3. Trigger là gì? 27](#_Toc25828187)

[4. Trigger chèn 27](#_Toc25828188)

[5. Trigger xóa 27](#_Toc25828189)

[6. Trigger cập nhật 28](#_Toc25828190)

[7. Trigger tổng hợp 28](#_Toc25828191)

[8. Trigger instead of 28](#_Toc25828192)