Buổi 3
Ràng buộc trong CSDL
Truy vấn đơn giản với SELECT

Trong khuôn khổ tài liệu buổi 3, chúng ta sử dụng CSDL QLVatTu:

- File *[QLVatTu] Mo ta CSDL.pdf*: Mô tả cấu trúc bảng, khóa ngoại, dữ liệu của CSDL.
- File *[QLVatTu] CSDL.sql*: Phát sinh CSDL.

I. Ràng buộc trong CSDL

Các HQT CSDL cung cấp các ràng buộc (*constraint*) để đảm bảo tính đúng đắn của dữ liệu.

Các loại ràng buộc thường gặp:

- `NOT NULL`: Không cho phép giá trị `NULL`.
- `UNIQUE`: Không cho phép giá trị trùng lặp.
- `PRIMARY KEY`: Ràng buộc khóa chính.
- `FOREIGN KEY`: Ràng buộc khóa ngoại.
- `CHECK`: Ràng buộc điều kiện.
- `DEFAULT`: Giá trị mặc định.

1. Ràng buộc NOT NULL

Ràng buộc `NOT NULL` không cho phép 1 thuộc tính nhận giá trị `NULL`, hay nói cách khác, người dùng bắt buộc phải nhập giá trị cho thuộc tính đó.

Cú pháp:

- Trong câu lệnh `CREATE TABLE`, thêm từ khóa `NOT NULL` vào sau câu khai báo thuộc tính[1].
- Trong câu lệnh `ALTER TABLE`, dùng cú pháp sau:

```
ALTER TABLE <Tên bảng>
MODIFY <Tên thuộc tính> <Kiểu dữ liệu> NOT NULL;
```

2. Ràng buộc UNIQUE

Ràng buộc `UNIQUE` không cho phép 1 thuộc tính nhận giá trị trùng lặp, hay nói cách khác, tất cả các giá trị của thuộc tính đó bắt buộc phải khác nhau.

Một thuộc tính là khóa chính ngay lập tức có ràng buộc `UNIQUE`, nhưng ngược lại, một thuộc tính có ràng buộc `UNIQUE` chưa chắc đã là khóa chính.

Mỗi bảng chỉ có thể có đúng 1 khóa chính, nhưng có thể có nhiều ràng buộc `UNIQUE`.

Cú pháp:

- Trong câu lệnh `CREATE TABLE`, thêm từ khóa `UNIQUE` vào câu khai báo thuộc tính.
- Trong câu lệnh `CREATE TABLE`, nếu muốn đặt tên cho ràng buộc hoặc tạo ràng buộc `UNIQUE` trên nhiều thuộc tính:

```
CONSTRAINT <Tên ràng buộc>[2] UNIQUE (<Danh sách thuộc tính>)
```

- Trong câu lệnh `ALTER TABLE`, dùng cú pháp sau:

---

[1] Xem tài liệu buổi 2, phần II.

[2] Tên của ràng buộc UNIQUE thường có dạng UC_<Tên bảng>

```
ALTER TABLE <Tên bảng>
ADD UNIQUE (<Tên thuộc tính>);
```

- Trong câu lệnh ALTER TABLE, nếu muốn đặt tên cho ràng buộc hoặc tạo ràng buộc UNIQUE trên nhiều thuộc tính:

```
ALTER TABLE <Tên bảng>
ADD CONSTRAINT <Tên ràng buộc> UNIQUE (<Danh sách thuộc tính>);
```

3. Ràng buộc PRIMARY KEY

Ràng buộc PRIMARY KEY xác định khóa chính, định danh cho từng dòng dữ liệu trong bảng.

Một thuộc chính là khóa chính ngay lập tức có ràng buộc UNIQUE và NOT NULL.

Mỗi bảng chỉ có thể có đúng 1 khóa chính.

Cú pháp:

- Trong câu lệnh CREATE TABLE, thêm từ khóa PRIMARY KEY vào câu khai báo thuộc tính[3].
- Trong câu lệnh CREATE TABLE, nếu muốn đặt tên cho ràng buộc hoặc tạo ràng buộc PRIMARY KEY trên nhiều thuộc tính:

```
CONSTRAINT <Tên ràng buộc>[4] PRIMARY KEY (<Danh sách thuộc tính>)
```

- Trong câu lệnh ALTER TABLE, dùng cú pháp sau:

```
ALTER TABLE <Tên bảng>
ADD PRIMARY KEY (<Tên thuộc tính>);
```

- Trong câu lệnh ALTER TABLE, nếu muốn đặt tên cho ràng buộc hoặc tạo ràng buộc FOREIGN KEY trên nhiều thuộc tính:

```
ALTER TABLE <Tên bảng>
ADD CONSTRAINT <Tên ràng buộc> PRIMARY KEY (<Danh sách thuộc tính>);
```

4. Ràng buộc FOREIGN KEY

Ràng buộc FOREIGN KEY xác định khóa ngoại trỏ đến 1 bảng khác.

Mỗi bảng có thể có nhiều khóa ngoại.

Cú pháp:

- Trong câu lệnh CREATE TABLE, thêm cú pháp sau vào câu khai báo thuộc tính[5]:

```
FOREIGN KEY REFERENCES <Tên bảng đích>(<Khóa chính của bảng đích>)
```

- Trong câu lệnh CREATE TABLE, nếu muốn đặt tên cho ràng buộc hoặc tạo ràng buộc FOREIGN KEY trên nhiều thuộc tính:

```
CONSTRAINT <Tên ràng buộc>[6] FOREIGN KEY (<Danh sách thuộc tính>)
REFERENCES <Tên bảng đích>(<Khóa chính của bảng đích>)
```

- Trong câu lệnh ALTER TABLE, dùng cú pháp sau:

```
ALTER TABLE <Tên bảng>
ADD FOREIGN KEY (<Tên thuộc tính>);
```

- Trong câu lệnh ALTER TABLE, nếu muốn đặt tên cho ràng buộc hoặc tạo ràng buộc FOREIGN KEY trên nhiều thuộc tính:

```
ALTER TABLE <Tên bảng>
ADD CONSTRAINT <Tên ràng buộc> FOREIGN KEY (<Danh sách thuộc tính>)
REFERENCES <Tên bảng đích>(<Khóa chính của bảng đích>);
```

---

[3] Xem tài liệu buổi 2, phần II

[4] Tên của ràng buộc PRIMARY KEY thường có dạng PK_<Tên bảng>

[5] Xem tài liệu buổi 2, phần II

[6] Tên của ràng buộc FOREIGN KEY thường có dạng FK_<Tên bảng hiện tại>_<Tên bảng đích>

5. Ràng buộc CHECK

Ràng buộc CHECK chỉ cho phép nhập giá trị cho thuộc tính theo 1 điều kiện nhất định.

Cú pháp:

- Trong câu lệnh CREATE TABLE, thêm cú pháp sau vào câu khai báo thuộc tính:

```
CHECK (<Biểu thức điều kiện>)
```

- Trong câu lệnh CREATE TABLE, nếu muốn đặt tên cho ràng buộc hoặc tạo ràng buộc CHECK trên nhiều thuộc tính:

```
CONSTRAINT <Tên ràng buộc>[7] CHECK (<Biểu thức điều kiện>)
```

- Trong câu lệnh ALTER TABLE, dùng cú pháp sau:

```
ALTER TABLE <Tên bảng>
ADD CHECK (<Biểu thức điều kiện>);
```

- Trong câu lệnh ALTER TABLE, nếu muốn đặt tên cho ràng buộc hoặc tạo ràng buộc CHECK trên nhiều thuộc tính:

```
ALTER TABLE <Tên bảng>
ADD CONSTRAINT <Tên ràng buộc> CHECK (<Biểu thức điều kiện>);
```

Ví dụ: Biểu thức điều kiện có thể là:

- SLTON >= 0
- SLTON >= 0 AND GIAMUA >= 0
- …

6. Ràng buộc DEFAULT

Ràng buộc DEFAULT quy định giá trị mặc định cho 1 thuộc tính nếu người dùng không điền giá trị cho thuộc tính đó.

Cú pháp:

- Trong câu lệnh CREATE TABLE, thêm cú pháp sau vào câu khai báo thuộc tính

```
DEFAULT <Giá trị>
```

- Trong câu lệnh ALTER TABLE, dùng cú pháp sau:

```
ALTER TABLE <Tên bảng>
ADD CONSTRAINT <Tên ràng buộc>
DEFAULT <Giá trị> FOR <Tên thuộc tính>;
```

7. Hủy bỏ ràng buộc

Để hủy bỏ 1 ràng buộc đã đặt tên (ngoại trừ ràng buộc DEFAULT), dùng cú pháp sau:

```
ALTER TABLE <Tên bảng>
DROP CONSTRAINT <Tên ràng buộc>;
```

Để hủy bỏ ràng buộc DEFAULT trên 1 thuộc tính của bảng, dùng cú pháp sau:

```
ALTER TABLE <Tên bảng>
ALTER COLUMN <Tên thuộc tính> DROP DEFAULT;
```

II. Cú pháp câu lệnh SELECT

Ngôn ngữ SQL cung cấp câu lệnh SELECT để lấy dữ liệu từ các bảng.

Cú pháp của 1 câu lệnh SELECT đơn giản:

```
SELECT <Danh sách thuộc tính>
FROM <Danh sách bảng>
```

[7] Tên của ràng buộc CHECK thường có dạng CHK_<Tên bảng>

Trong đó: <Danh sách thuộc tính> là tên các thuộc tính thuộc <Danh sách bảng>, hoặc là * nếu muốn lấy hết tất cả các thuộc tính.

Ví dụ:

- Liệt kê danh sách vật tư có trong cửa hàng. Gồm: Tất cả thuộc tính:

```
SELECT *
FROM VATTU
```

- Liệt kê danh sách vật tư có trong cửa hàng. Gồm: Mã vật tư, tên vật tư:

```
SELECT MAVT, TENVT
FROM VATTU
```

III. Mệnh đề điều kiện WHERE

1. Cú pháp

Thông thường, chúng ta sẽ cần phải lấy dữ liệu phù hợp với 1 tiêu chí nào đó. Ngôn ngữ SQL cung cấp mệnh đề WHERE để quy định biểu thức điều kiện cho câu truy vấn[8].

Trong câu lệnh SELECT, mệnh đề WHERE nằm ngay sau mệnh đề FROM.

Biểu thức điều kiện trong mệnh đề WHERE thường sử dụng các toán tử so sánh như =, <>, >, >=, <, <= hoặc các toán tử như BETWEEN, LIKE, IN, kèm theo các toán tử logic như AND, OR, NOT để nối các biểu thức điều kiện đơn.

Ví dụ:

- Liệt kê những vật tư có số lượng tồn từ 100.000 trở lên. Gồm: Tất cả thuộc tính:

```
SELECT *
FROM VATTU
WHERE SLTON >= 100000
```

- Liệt kê những vật tư có số lượng tồn từ 100.000 trở lên và giá mua từ 30.000 trở lên. Gồm: Tất cả thuộc tính:

```
SELECT *
FROM VATTU
WHERE SLTON >= 100000 AND GIAMUA >= 30000
```

- Liệt kê những khách hàng sống ở Bình Chánh và Tân Bình. Gồm: Tên khách hàng, địa chỉ:

```
SELECT TENKH, DIACHI
FROM KHACHHANG
WHERE DIACHI = N'Bình Chánh' OR DIACHI = N'Tân Bình'
```

- Liệt kê những khách hàng không sống ở Tân Bình. Gồm: Tên khách hàng, địa chỉ:

```
SELECT TENKH, DIACHI
FROM KHACHHANG
WHERE NOT DIACHI = N'Tân Bình'
```

2. Toán tử BETWEEN

Toán tử BETWEEN dùng để quy định 1 thuộc tính nằm trong 1 khoảng giá trị nào đó.

Ví dụ: Liệt kê danh sách vật tư có giá từ 20.000 đến 40.000. Gồm: Tên vật tư, giá mua:

```
SELECT TENVT, GIAMUA
FROM VATTU
WHERE GIAMUA BETWEEN 20000 AND 40000
```

[8] Thường dùng cho câu SELECT, UPDATE và DELETE

3. Toán tử LIKE

Toán tử LIKE dùng để viết biểu thức điều kiện theo dạng gần đúng (dành cho chuỗi). SQL Server hỗ trợ một số kí tự đại diện (*wildcard character*) như sau:

Kí tự đại diện	Ý nghĩa	Ví dụ
_	1 kí tự	h_t
%	0, 1 hoặc nhiều kí tự	black%
[]	Bất kì kí tự nào trong []	h[aio]t
^	Bất kì kí tự nào không nằm trong []	h[^aio]t
-	Một khoảng kí tự	h[a-e]t

Ví dụ: Liệt kê số lượng tồn kho của từng loại gạch. Gồm: Tên vật tư, số lượng tồn:

```
SELECT TENVT, SLTON
FROM VATTU
WHERE TENVT LIKE N'Gạch%'
```

4. Toán tử IN

Toán tử IN dùng để quy định giá trị 1 thuộc tính nằm trong 1 tập hợp nào đó.

Ví dụ: Liệt kê những vật tư có đơn vị tính là bao, viên hoặc cái. Gồm: Tên vật tư, đơn vị tính:

```
SELECT TENVT, DVT
FROM VATTU
WHERE DVT IN (N'Bao', N'Viên', N'Cái')
```

5. Mệnh đề WHERE trong câu lệnh UPDATE và DELETE

Tương tự câu lệnh SELECT, mệnh đề WHERE cũng có thể được sử dụng trong câu lệnh UPDATE và DELETE để chọn ra những dòng nào sẽ được cập nhật giá trị hoặc xóa bỏ.

Trong câu lệnh UPDATE, mệnh đề WHERE nằm ngay sau mệnh đề SET.

Ví dụ: Cập nhật địa chỉ của khách hàng Lê Hoàng Nam thành Quận 3:

```
UPDATE KHACHHANG
SET DIACHI = N'Quận 3'
WHERE TENKH = N'Lê Hoàng Nam'
```

Trong câu lệnh DELETE, mệnh đề WHERE nằm ngay sau mệnh đề FROM.

Ví dụ: Xóa khách hàng có mã KH05:

```
DELETE FROM KHACHHANG
WHERE MAKH = 'KH05'
```

IV. Các hàm có sẵn của ngôn ngữ SQL

Ngôn ngữ SQL cung cấp 1 số hàm có sẵn[9] (*built-in function*) để hỗ trợ cho việc tính toán:

Hàm	Ý nghĩa
SUM(), AVG()	Tính tổng và trung bình cộng
MAX(), MIN()	Tìm giá trị lớn nhất và nhỏ nhất
COUNT()	Đếm số dòng kết quả
DAY(), MONTH(), YEAR()	Lấy ngày, tháng, năm của 1 giá trị kiểu date
GETDATE()	Lấy ngày hiện tại của hệ thống
CAST(), CONVERT()	Chuyển đổi kiểu dữ liệu

[9] Tham khảo cách sử dụng và các hàm khác tại link: https://www.w3schools.com/sql/sql_ref_sqlserver.asp

V. Bài tập

Thực hiện các yêu cầu truy vấn sau theo định dạng dưới đây:

```
-- Câu 1
SELECT ……
-- Câu 2
SELECT ……
```

1. Liệt kê danh sách vật tư có trong cửa hàng. Gồm: Tất cả thuộc tính.
2. Liệt kê danh sách vật tư có trong cửa hàng. Gồm: Mã vật tư, tên vật tư.
3. Liệt kê danh sách vật tư có số lượng tồn từ 100.000 trở lên. Gồm: Tất cả thuộc tính.
4. Liệt kê danh sách vật tư có số lượng tồn từ 100.000 trở lên và giá mua từ 30.000 trở lên. Gồm: Tất cả thuộc tính.
5. Liệt kê những khách hàng sống ở Bình Chánh và Tân Bình. Gồm: Tên khách hàng, địa chỉ.
6. Đếm số lượng hóa đơn có trong hệ thống.
7. Đếm số lượng hóa đơn của khách hàng có mã là KH01.
8. Cho biết giá mua cao nhất trong bảng Vật tư.
9. Cho biết số lượng tồn kho ít nhất trong bảng Vật tư.
10. Tính tổng số lượng đã bán của vật tư có mã là VT01.
11. Liệt kê danh sách hóa đơn được lập trong ngày 25/05/2015. Gồm: Tất cả thuộc tính.
12. Liệt kê danh sách hóa đơn được lập trong tháng 05 và tháng 06/2015. Gồm: Tất cả thuộc tính.
    - Cách 1: Dùng phép so sánh ngày
    - Cách 2: Dùng toán tử `BETWEEN`
    - Cách 3: Dùng hàm `MONTH()` và `YEAR()`
13. Tính tổng tiền hàng còn tồn kho.
14. Tính tổng tiền hàng còn tồn kho với các vật tư là gạch.
15. Tính tổng doanh số của cửa hàng.
16. Tính tổng doanh thu của cửa hàng.
17. Liệt kê danh sách khách hàng không có SĐT hoặc Email. Gồm: Tên khách hàng, ĐT, Email.
18. Liệt kê danh sách khách hàng có tên lót là "Thị". Gồm: Tên khách hàng.
19. Cho biết tổng tiền của hóa đơn HD005.
20. Cho biết ngày gần nhất mà cửa hàng bán được hàng.