**1. Table**

Create database myDB;

Use myDB;

CREATE TABLE employees(

employee\_id INT,

first\_name VARCHAR(50),

last\_name VARCHAR(50),

hourly\_day DECIMAL(5, 2),

hire\_day DATE

);

Select \*from employees;

Rename table employees to workers;

Alter table employees

Add phone\_number varchar(15);

Alter table employees

Rename Column phone\_number to email;

Alter table employees

Modify column email varchar(50);

Alter table employees

Modify email varchar(50)

After last\_name;

Alter table employees

Drop column email;

**2. INSERT ROWS**

use myDB;

insert into employees

value(1, "Duc", "Pham", 23.50,"2000-8-30");

select \*from employees;

**3. SELECT**

select last\_name, first\_name

from employees;

select \*

from employees

where employee\_id = 1;

select \*

from employees

where first\_name = "…";

(<=; >=; !=; is null; is not null)

**4. UPDATE & DELETE**

update employees

set hourly\_day = "10.11"

where employee\_id = 3;

select \*from employees;

delete from employees;

where employee\_id = 2;

select \* from employees;

**5. AUTOCOMMIT, COMMIT, ROLLBACK**

Set autocommit = off;

Rollback;

Commit;

**6. CURRENT\_DATE() & CURRENT\_TIME()**

create table test(

my\_date date,

my\_time time,

my\_datetime datetime

);

Select \*from test;

insert into test

value(current\_date(), current\_time(), now());

value(current\_date() +- 1, null, null);

**7. UNIQUE**

Ràng buộc UNIQUE đảm bảo rằng tất cả các giá trị trong một cột là khác nhau.

Cả ràng buộc UNIQUE và PRIMARY KEY đều đảm bảo tính duy nhất cho một cột hoặc tập hợp các cột.

Một PRIMARY KEY ràng buộc tự động có một UNIQUE ràng buộc.

Tuy nhiên, bạn có thể có nhiều UNIQUE ràng buộc trên mỗi bảng, nhưng chỉ có một PRIMARY KEY ràng buộc trên mỗi bảng.

create table product(

product\_id int,

product\_name varchar(50) unique,

price decimal(4, 2)

);

alter table product

add constraint

unique(product\_name);

insert into product

value (100, "a", 3.99),

(101, "b", 4.00),

(102, "c", 4.01),

(103, "b", 4.02);

=> Error Code: 1062. Duplicate entry 'b' for key 'product.product\_name'

**8. NOT NULL**

Theo mặc định, một cột có thể chứa giá trị NULL.

Ràng buộc NOT NULL buộc cột KHÔNG chấp nhận giá trị NULL.

Điều này buộc một trường phải luôn chứa một giá trị, nghĩa là bạn không thể chèn bản ghi mới hoặc cập nhật bản ghi mà không thêm giá trị vào trường này.

CREATE TABLE Persons (  
    ID int NOT NULL,  
    LastName varchar(255) NOT NULL,  
    FirstName varchar(255) NOT NULL,  
    Age int  
);

Để tạo NOT NULLràng buộc trên cột "Tuổi" khi bảng "Người" đã được tạo, hãy sử dụng câu lệnh SQL sau:

ALTER TABLE Persons  
MODIFY COLUMN Age int NOT NULL;

**9. CHECK**

Ràng buộc CHECK được sử dụng để giới hạn phạm vi giá trị có thể được đặt trong một cột.

Nếu bạn xác định một ràng buộc CHECK trên một cột, nó sẽ chỉ cho phép một số giá trị nhất định cho cột này.

Nếu bạn xác định một ràng buộc CHECK trên một bảng, nó có thể giới hạn các giá trị trong các cột nhất định dựa trên các giá trị trong các cột khác trong hàng.

CREATE TABLE Persons (  
    ID int NOT NULL,  
    LastName varchar(255) NOT NULL,  
    FirstName varchar(255),  
    Age int,  
    CHECK (Age>=18)  
);

Để cho phép đặt tên cho một ràng buộc CHECK và để xác định một ràng buộc CHECK trên nhiều cột, hãy sử dụng cú pháp SQL sau:

CREATE TABLE Persons (  
    ID int NOT NULL,  
    LastName varchar(255) NOT NULL,  
    FirstName varchar(255),  
    Age int,  
    City varchar(255),  
    CONSTRAINT CHK\_Person CHECK (Age>=18 AND City='Sandnes')  
);

Để tạo một ràng buộc CHECK trên cột "Tuổi" khi bảng đã được tạo, hãy sử dụng câu lệnh SQL sau:

ALTER TABLE Persons  
ADD CHECK (Age>=18);

Để cho phép đặt tên cho một ràng buộc CHECK và để xác định một ràng buộc CHECK trên nhiều cột, hãy sử dụng cú pháp SQL sau:

ALTER TABLE Persons  
ADD CONSTRAINT CHK\_PersonAge CHECK (Age>=18 AND City='Sandnes');

Để loại bỏ một CHECKràng buộc, hãy sử dụng SQL sau:

ALTER TABLE Persons  
DROP CHECK CHK\_PersonAge;

**10. DEFAULT**

Ràng buộc DEFAULT được sử dụng để đặt giá trị mặc định cho một cột.

Giá trị mặc định sẽ được thêm vào tất cả các bản ghi mới, nếu không có giá trị nào khác được chỉ định.

SQL sau đặt giá trị DEFAULT cho cột "Thành phố" khi bảng "Người" được tạo:

CREATE TABLE Persons (  
    ID int NOT NULL,  
    LastName varchar(255) NOT NULL,  
    FirstName varchar(255),  
    Age int,  
    City varchar(255) DEFAULT 'Sandnes'  
);

Ràng buộc DEFAULT cũng có thể được sử dụng để chèn các giá trị hệ thống, bằng cách sử dụng các hàm như : [GETDATE()](https://www.w3schools.com/sql/func_sqlserver_getdate.asp)

CREATE TABLE Orders (  
    ID int NOT NULL,  
    OrderNumber int NOT NULL,  
    OrderDate date DEFAULT GETDATE()  
);

Để tạo ràng buộc DEFAULT trên cột "Thành phố" khi bảng đã được tạo, hãy sử dụng câu lệnh SQL sau:

ALTER TABLE Persons  
ALTER City SET DEFAULT 'Sandnes';

Để loại bỏ một ràng buộc DEFAULT, hãy sử dụng SQL sau:

ALTER TABLE Persons  
ALTER City DROP DEFAULT;

**11. PRIMARY KEYS**

Ràng buộc PRIMARY KEY xác định duy nhất từng bản ghi trong một bảng.

Các khóa chính phải chứa các giá trị UNIQUE và không được chứa các giá trị NULL.

Một bảng chỉ có thể có **MỘT** khóa chính; và trong bảng, khóa chính này có thể bao gồm một hoặc nhiều cột (trường).

SQL sau đây tạo một PRIMARY KEY cột "ID" khi bảng "Người" được tạo:

CREATE TABLE Persons (  
    ID int NOT NULL,  
    LastName varchar(255) NOT NULL,  
    FirstName varchar(255),  
    Age int,  
    PRIMARY KEY (ID)  
);

Để cho phép đặt tên cho một ràng buộc PRIMARY KEY và để xác định một ràng buộc PRIMARY KEY trên nhiều cột, hãy sử dụng cú pháp SQL sau:

CREATE TABLE Persons (  
    ID int NOT NULL,  
    LastName varchar(255) NOT NULL,  
    FirstName varchar(255),  
    Age int,  
    CONSTRAINT PK\_Person PRIMARY KEY (ID,LastName)  
);

**Lưu ý:** Trong ví dụ trên chỉ có MỘT PRIMARY KEY(PK\_Person). Tuy nhiên, GIÁ TRỊ của khóa chính được tạo thành từ HAI CỘT (ID + Họ).

Để tạo ràng buộc PRIMARY KEY trên cột "ID" khi bảng đã được tạo, hãy sử dụng câu lệnh SQL sau:

ALTER TABLE Persons  
ADD PRIMARY KEY (ID);

Để cho phép đặt tên cho một ràng buộc PRIMARY KEY và để xác định một ràng buộc PRIMARY KEY trên nhiều cột, hãy sử dụng cú pháp SQL sau:

ALTER TABLE Persons  
ADD CONSTRAINT PK\_Person PRIMARY KEY (ID,LastName);

**Lưu ý:** Nếu sử dụng ALTER TABLE để thêm khóa chính, (các) cột khóa chính phải được khai báo là không chứa giá trị NULL (khi bảng được tạo lần đầu tiên).

Để loại bỏ một ràng buộc PRIMARY KEY, hãy sử dụng SQL sau:

ALTER TABLE Persons  
DROP PRIMARY KEY;

**12. AUTO\_INCREMENT**

Tự động tăng(auto increment) cho phép một số duy nhất được tạo tự động khi một bản ghi mới được chèn vào bảng.

Thường thì đây là trường khóa chính mà ta muốn được tạo tự động mỗi khi một bản ghi mới được chèn vào.

Câu lệnh SQL sau xác định cột "Personid" là trường khóa chính tăng tự động trong bảng "Persons":

CREATE TABLE Persons (  
    Personid int NOT NULL AUTO\_INCREMENT,  
    LastName varchar(255) NOT NULL,  
    FirstName varchar(255),  
    Age int,  
    PRIMARY KEY (Personid)  
);

MySQL sử dụng từu khóa AUTO\_INCREMENT để thực hiện tính năng tự động tăng.

Theo mặc định, giá trị bắt đầu cho AUTO\_INCREMENT là 1 và giá trị này sẽ tăng thêm 1 cho mỗi bản ghi mới.

Để chuỗi AUTO\_INCREMENT bắt đầu với một giá trị khác, hãy sử dụng câu lệnh SQL sau:

ALTER TABLE Persons AUTO\_INCREMENT=100;

Để chèn một bản ghi mới vào bảng "Persons", chúng ta KHÔNG phải chỉ định giá trị cho cột "Personid" (một giá trị duy nhất sẽ được thêm tự động):

INSERT INTO Persons (FirstName,LastName)  
VALUES ('Lars','Monsen');

Câu lệnh SQL trên sẽ chèn một bản ghi mới vào bảng "Persons". Cột "Personid" sẽ được gán một giá trị duy nhất. Cột "FirstName" sẽ được đặt thành "Lars" và cột "LastName" sẽ được đặt thành "Monsen".

**13. FOREIGN KEYS**

Ràng buộc FOREIGN KEY được sử dụng để ngăn chặn các hành động phá hủy liên kết giữa các bảng.

FOREIGN KEY là một trường (hoặc tập hợp các trường) trong một bảng, tham chiếu đến một bảng [PRIMARY KEY](https://www.w3schools.com/sql/sql_primarykey.asp) khác.

Bảng có khóa ngoại được gọi là bảng con và bảng có khóa chính được gọi là bảng được tham chiếu hoặc bảng cha.

SQL sau tạo một FOREIGN KEYcột "PersonID" khi bảng "Orders" được tạo:

CREATE TABLE Orders (  
    OrderID int NOT NULL,  
    OrderNumber int NOT NULL,  
    PersonID int,  
    PRIMARY KEY (OrderID),  
    FOREIGN KEY (PersonID) REFERENCES Persons(PersonID)  
);

Để tạo ràng buộc FOREIGN KEY trên cột "PersonID" khi bảng "Orders" đã được tạo, hãy sử dụng câu lệnh SQL sau:

ALTER TABLE Orders  
ADD FOREIGN KEY (PersonID) REFERENCES Persons(PersonID);

Để cho phép đặt tên cho một ràng buộc FOREIGN KEY và để xác định một ràng buộc FOREIGN KEY trên nhiều cột, hãy sử dụng cú pháp SQL sau:

ALTER TABLE Orders  
ADD CONSTRAINT FK\_PersonOrder  
FOREIGN KEY (PersonID) REFERENCES Persons(PersonID);

Để loại bỏ một ràng buộc FOREIGN KEY, hãy sử dụng SQL sau:

ALTER TABLE Orders  
DROP FOREIGN KEY FK\_PersonOrder;

**14. JOINS**

Một mệnh đề JOIN được sử dụng để kết hợp các hàng từ hai hoặc nhiều bảng, dựa trên một cột có liên quan giữa chúng.

Dưới đây là các loại JOINS khác nhau trong SQL:

* (INNER) JOIN: Trả về các bản ghi có giá trị khớp trong cả hai bảng
* LEFT (OUTER) JOIN: Trả về tất cả các bản ghi từ bảng bên trái và các bản ghi phù hợp từ bảng bên phải
* RIGHT (OUTER) JOIN: Trả về tất cả các bản ghi từ bảng bên phải và các bản ghi phù hợp từ bảng bên trái
* FULL (OUTER) JOIN: Trả về tất cả các bản ghi khi có sự trùng khớp trong bảng bên trái hoặc bên phải

      

SELECT Orders.OrderID, Customers.CustomerName, Orders.OrderDate  
FROM Orders  
INNER JOIN Customers ON Orders.CustomerID=Customers.CustomerID;