Contents

SQL Constraints (ràng buộc)

Vd: tạo bảng trống

```
CREATE TABLE Persons (-- demo.Persons : tạo bảng trong database demo
    PersonID int,
    LastName varchar(255),
    FirstName varchar(255),
    Address varchar(255),
    City varchar(255)
);
INSERT INTO database.persons (PersonID, LastName, FirstName, Address, City)
VALUES ('1', 'Tom B', 'Erichsen', 'Stavanger', 'Norway');
--  ALTER TABLE Persons
-- MODIFY PersonID int NOT NULL;
```

=> ràng buộc:

Các ràng buộc được sử dụng để giới hạn loại dữ liệu có thể đi vào bảng. Điều này đảm bảo tính chính xác và độ tin cậy của dữ liệu trong bảng. Nếu có bất kỳ vi phạm nào giữa ràng buộc và hành động dữ liệu, hành động đó sẽ bị hủy bỏ.

Các ràng buộc sau thường được sử dụng trong SQL:

- `NOT NULL` - Đảm bảo rằng một cột không thể có giá trị NULL
- `UNIQUE` - Đảm bảo rằng tất cả các giá trị trong một cột là khác nhau
- `PRIMARY KEY`- Sự kết hợp của a `NOT NULL`và `UNIQUE`. Xác định duy nhất từng hàng trong bảng

- FOREIGN KEY - Ngăn chặn các hành động phá hủy liên kết giữa các bảng
- CHECK - Đảm bảo rằng các giá trị trong một cột thỏa mãn một điều kiện cụ thể
- DEFAULT - Đặt giá trị mặc định cho một cột nếu không có giá trị nào được chỉ định
- CREATE INDEX - Được sử dụng để tạo và lấy dữ liệu từ cơ sở dữ liệu rất nhanh chóng

SQL NOT NULL Constraint

Theo mặc định, một cột có thể chứa các giá trị NULL.
Các NOT NULLhạn chế thực thi một cột để không chấp nhận giá trị NULL.
Điều này bắt buộc trường luôn chứa giá trị, có nghĩa là bạn không thể chèn bản ghi mới hoặc cập nhật bản ghi mà không thêm giá trị vào trường này.

Vd:

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

SQL UNIQUE Constraint

Các UNIQUEhạn chế để đảm bảo rằng tất cả các giá trị trong một cột là khác nhau.
Cả ràng buộc UNIQUEvà PRIMARY KEYràng buộc đều đảm bảo tính duy nhất cho một cột hoặc tập hợp các cột.
Một PRIMARY KEYràng buộc tự động có một UNIQUEràng buộc.
Tuy nhiên, bạn có thể có nhiều UNIQUEràng buộc trên mỗi bảng, nhưng chỉ có một PRIMARY KEYràng buộc cho mỗi bảng.

Vd:

```
CREATE TABLE Persons (
```

```
    ID int NOT NULL,
    LastName varchar(255) NOT NULL,
    FirstName varchar(255),
    Age int,
    UNIQUE (ID)
);

SELECT * FROM databasename.persons;
-- Để tạo UNIQUEràng buộc trên cột "Age" khi bảng đã được tạo, hãy sử dụng SQL sau:
ALTER TABLE Persons
ADD UNIQUE (Age);
-- Để loại bỏ một UNIQUE ràng buộc, hãy sử dụng SQL sau:
ALTER TABLE Persons
DROP INDEX Age;
```

SQL PRIMARY KEY Constraint

Các `PRIMARY KEY`hạn chế nhận dạng duy nhất mỗi bản ghi trong một bảng.
Khóa chính phải chứa giá trị DUY NHẤT và không được chứa 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).

Vd:

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

-- ALTER TABLE Persons
-- ADD PRIMARY KEY (Age); => lỗi

-- Để loại bỏ một PRIMARY KEYràng buộc, hãy sử dụng SQL sau:
ALTER TABLE Persons
DROP PRIMARY KEY;
```

SQL FOREIGN KEY Constraint

Các `FOREIGN KEY`hạn chế được sử dụng để ngăn chặn những hành động đó sẽ phá hủy các liên kết giữa các bảng.
A `FOREIGN KEY`là một trường (hoặc tập hợp các trường) trong một bảng, tham chiếu đến `PRIMARY KEY`trong một bảng khác.
Bảng có khóa ngoại được gọi là bảng con và bảng có khóa chính được gọi là bảng tham chiếu hoặc bảng cha.

Persons Table

PersonID	LastName	FirstName	Age
1	Hansen	Ola	30
2	Svendson	Tove	23
3	Pettersen	Kari	20

Orders Table

OrderID	OrderNumber	PersonID
1	77895	3
2	44678	3
3	22456	2
4	24562	1

The "PersonID" column in the "Persons" table is the `PRIMARY KEY` in the "Persons" table.
The "PersonID" column in the "Orders" table is a `FOREIGN KEY` in the "Orders" table.

Vd:

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

```
);

CREATE TABLE Orders (
    OrderID int NOT NULL,
    OrderNumber int NOT NULL,
    PersonID int,
    PRIMARY KEY (OrderID),
    FOREIGN KEY (PersonID) REFERENCES Persons(PersonID)
);
--  Cách thêm FK
SELECT * FROM databasename.orders;
-- ALTER TABLE Orders
-- ADD FOREIGN KEY (PersonID) REFERENCES Persons(PersonID);
```

SQL CHECK Constraint

Các CHECKhạn chế được sử dụng để giới hạn phạm vi giá trị có thể được đặt trong một cột.
Nếu bạn xác định một CHECKràng buộc trên một cột, nó sẽ chỉ cho phép một số giá trị nhất định cho cột này.
Nếu bạn xác định một CHECKràng buộc 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.

Vd:

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

SELECT * FROM demo.persons;
INSERT INTO demo.persons (ID, LastName, FirstName, Age)
VALUES ('1', 'Tom B', 'Stavanger', '20');
```

⇨ Được add

```
INSERT INTO demo.persons (ID, LastName, FirstName, Age)
VALUES ('1', 'Tom B', 'Stavanger', '10');
```

⇨ Lỗi

SQL DEFAULT Constraint

Các DEFAULThạn chế được sử dụng để đặt một 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.

Vd:

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

INSERT INTO demo.persons (ID, LastName, FirstName, Age)
VALUES ('1', 'Tom B', 'Stavanger', '20');
```

⇨ Thiếu cột city nhưng city sẽ mặc định để là sadness

Để tạo DEFAULTràng buộc trên cột "Thành phố" khi bảng đã được tạo, hãy sử dụng SQL sau:

MySQL:

```
ALTER TABLE Persons
ALTER City SET DEFAULT 'Sandnes';
```

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

MySQL:

```
ALTER TABLE Persons
ALTER City DROP DEFAULT;
```

SQL CREATE INDEX Statement

Các chỉ mục được sử dụng để lấy dữ liệu từ cơ sở dữ liệu nhanh hơn so với cách khác. Người dùng không thể nhìn thấy các chỉ mục, chúng chỉ được sử dụng để tăng tốc độ tìm kiếm / truy vấn.

Cú Pháp CREATE INDEX
Tạo chỉ mục trên bảng. Các giá trị trùng lặp được phép:

```
CREATE INDEX index_name
ON table_name (column1, column2, ...);
```

Vd:
SELECT * FROM demo.persons;
CREATE INDEX idx_lastname
ON Persons (LastName);
CREATE INDEX idx_pname
ON Persons (LastName, FirstName);

Câu DROP INDEXlệnh được sử dụng để xóa một chỉ mục trong một bảng.

MySQL:

```
ALTER TABLE table_name
DROP INDEX index_name;
```