**TÌM HIỂU VỀ UNIQUE TRONG SQL**

[Primary Key](https://freetuts.net/tao-khoa-chinh-primary-key-trong-mysql-316.html)  trong một bảng sẽ tồn tại một khóa chính (có thể có nhiều fields) và giá trị của nó sẽ là duy nhất, nghĩa là dữ liệu sẽ không được trùng lặp. Vậy thì UNIQUEcũng có tính chất tương tự như vậy, tức là nếu bạn xác định một field nào đó là UNIQUE dữ liệu của mỗi record không được trùng lặp nhau.

UNIQUE thực chất là một index KEY nên khi bạn thiết lập một field nào đó là UNIQUE thì việc truy vấn dữ liệu trên field đó sẽ nhanh hơn. Nó được dùng trong trường hợp bạn muốn tạo thêm một field có giá trị là duy nhất.

Ví dụ trong bảng Users gồm các fields như sau: Users(id, username, email) thì với id ta sẽ chọn làm khóa chính. Nhưng mà tôi muốn email và username của người dùng phải là duy nhất, nghĩa là người dùng không được sử dụng một email hoặc username để đăng ký nhiều tài khoản. Để giải quyết thì ta sẽ thiết lập cho hai fields này là UNIQUE.

Như vậy sự khác biệt giữa UNIQUE và Primary Key là:

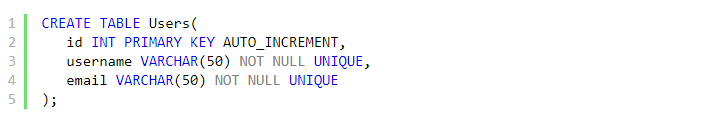
 Primary Key có cấp cao hơn UNIQUE, nó  thường dùng để phân biệt giữa các record.

Trong mỗi bảng chỉ có có một Primary Key, còn UNIQUE thì có thể có nhiều

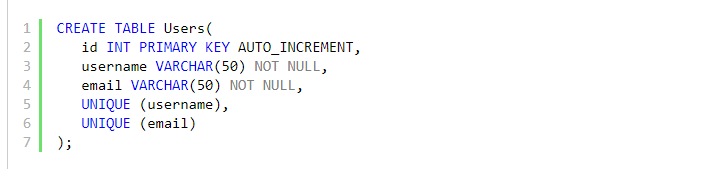
Để tạo UNIQUE ta có hai cách, cách thứ nhất là tạo trực tiếp trong lệnh CREATE TABLE và cách thứ hai là sử dụng lệnh ALTER TABLE.

### \*Tạo trong lệnh tạo bảng

Chúng ta sẽ thêm từ khóa UNIQUE vào field mà ban muốn thiết lập nó là UNIQUE.

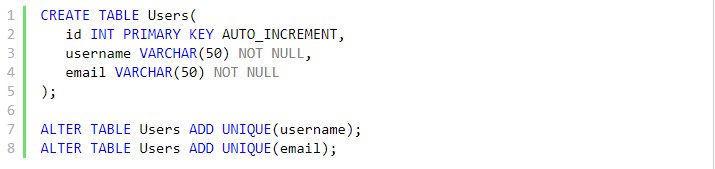


Hoặc ta có thể thêm nó ở dưới danh sách các fields (giống Primary Key).



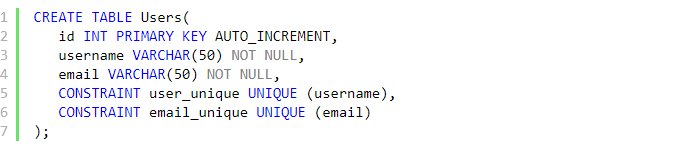
### \* Sử dụng Alter Table để tạo

Trong trường hợp này chúng ta sẽ tạo bảng trước, sau đó sử dụng lệnh Alter Table để thêm UNIQUE.



### \* Sử dụng CONSTRAINT để tạo tên cho UNIQUE

Cách này khuyến khích sử dụng vì mỗi index chúng ta nên đặt cho nó một cái tên để sau này muốn thực hiện thao tác delete unique thì sẽ dựa vào tên constraint đó.  Ví dụ:



Hoặc



### \* Xóa (Drop) Unique trong MySQL

Sau khi tạo UNIQUE xong nếu bạn muốn bỏ nó đi thì sẽ sử dụng lệnh ALTER TABLE kết hợp với DROP. Tuy nhiên để xóa UNIQE nào thì chúng ta phải biết được tên của nó, vì vậy ta nên dùng CONSTRAINT để tạo unique nhé. Sau đây là ví dụ xóa UNIQUE.



# Tìm hiểu them về Insert, Delete và Update bị ảnh hưởng như thế nào khi đánh Index ?

Trong các bài viết trước về chỉ mục, chúng ta chỉ bàn về phần hiệu suất, tối ưu truy vấn, nhưng SQL không chỉ là về truy vấn. Nó cũng hỗ trợ thao tác dữ liệu. Các lệnh tương ứng là INSERT, DELETE, và UPDATE, cái gọi là “data manipulation language - ngôn ngữ thao tác dữ liệu” (DML) - một phần của chuẩn SQL. Hiệu suất của các lệnh trên đa phần bị ảnh hưởng tiêu cực bởi việc đánh chỉ mục.

Một chỉ mục đơn thuần nó là dư thừa. Nó chỉ chứa dữ liệu mà cũng được cất trữ trong các bảng. Trong các hoạt động thay đổi dữ liệu, cơ sở dữ liệu phải giữ các dư thừa đó luôn nhất quán (giống nhau trong index và trong bảng). Cụ thể, nó có nghĩa là INSERT, DELETE, hay UPDATE không chỉ ảnh hưởng đến bản chính mà các chỉ mục cũng bị ảnh hưởng.

* INSERT

Số lượng chỉ mục trên một bảng là hệ số chi phối nhiều nhất tới hiệu suất của INSERT. Chỉ mục của một bảng càng nhiều, việc thực thi càng chậm. Câu lệnh INSERT là hoạt động duy nhất mà không thể trực tiếp hưởng lợi từ việc đánh chỉ mục vì nó không có mệnh đề WHERE.Thêm một hàng mới vào một bảng bao gồm một vài bước.Trước tiên, database phải tìm nơi để cất trữ hàng. Đối với các bảng heap thông thường - không có thứ tự hàng cụ thể nào - cơ sở dữ liệu có thể lấy bất kì block nào có đủ không gian trống cho bảng này. Đây là một quá trình rất đơn giản và nhanh chóng, hầu hết được hoàn thành trong main memory. Tất cả những việc database làm sau đó là thêm dữ liệu mới vào khối dữ liệu tương ứng.Nếu có nhiều index trên một bảng, CSDL phải đảm bảo các hàng mới thêm vào cũng được tìm thấy thông qua các chỉ mục. Vì lý do này mà nó phải thêm phần tử chỉ mục cho hàng mới trên mọi index. Do đó mà số chỉ mục càng nhiều sẽ khuếch đại chi phí cho mỗi thao tác INSERT.Hơn nữa, thêm một phần tử mới vào một chỉ mục đắt hơn nhiều do với việc thêm vào một cấu trúc heap bởi vì CSDL phải giữ thứ tự các chỉ mục và cây cân bằng. Điều đó nghĩa là phần tử mới không thể được ghi cho bất kì block nào, mà nó sẽ thuộc về một nút lá (block) cụ thể. Mặc dù CSDL sử dụng cây chỉ mục để tìm đúng nút lá, nó vẫn phải đọc một vài các nút lá khác cho việc duyệt cây.Chỉ khi nút lá đúng được xác định, database sẽ xác nhận rằng có đủ không gian trống trong nút này. Nếu không, cơ sở dữ liệu sẽ chia tách nút lá và phân phối lại các phần tử giữa nút cũ và nút mới. Quá trình này cũng ảnh hưởng tới cả nút mới và nút cũ, tức ảnh hưởng bị nhân đôi. Xa hơn, nút nhánh có thể cũng bị đầy vì vậy nó có thể bị phân tách. Trong trường hợp xấu nhất, cơ sở dữ liệu sẽ tách tất cả các nút. Đây là trường hợp duy nhất mà cây được thêm tầng và lớn hơn về độ sâu. Sau tất cả, việc duy trì index là phần đắt nhất của hoạt động INSERT. Việc thêm một chỉ mục đơn là đủ để tăng thời gian thực hiện. Mỗi index được thêm lại làm chậm các việc thực thi hơn nữa.

Hiệu suất của INSERT theo số chỉ mục :

Chú ý: Các chỉ mục đầu tiên tạo ra sự khác biệt lớn nhất.Để tối ưu hóa hiệu suất insert thì rất rất quan trọng việc giữ số index nhỏ.

Mẹo: Sử dụng index cần xem xét kĩ lưỡng và cần tiết kiệm số index, và tránh các index rườm rà bất cứ khi nào có thể. Điều này cũng có lợi với các câu lệnh DELETEhay UPDATE.

Xem xét riêng câu lệnh INSERT, tốt nhất nên tránh hoàn toàn các chỉ mục - điều này sẽ có hiệu quả tốt nhất cho hiệu suất INSERT. Tuy nhiên các bảng không có chỉ mục là không thực tế trong các ứng dụng. Bạn luôn cần truy vấn dữ liệu vì vậy bạn cần đánh chỉ mục để cải thiên tốc độ truy vấn.

* DELETE

Không giống như câu lệnh INSERT, câu lệnh DELETE có mệnh đề WHERE mà có thể sử dụng tất cả các phương pháp mô tả ở các bài viết trước để hưởng lợi trực tiếp từ việc đánh chỉ mục. Trên thực tế, lệnh DELETE làm việc giống như một việc lựa chọn các hàng xác định để xóa.

Việc xóa một hàng thực tế là một quá trình tương tự với việc INSERT một hàng - đặc biệt là về việc loại bỏ các tham chiếu/con trỏ của index và các hoạt động để giữ cho cậy index cân bằng. Do đó mà hiệu suất gần giống với hiệu suất cho hành động INSERT.

Hiệu suất của DELETE theo số chỉ mục :

Về lý thuyết, chúng ta mong đợi hiệu suất của lệnh DELETE là tốt nhất trong trường hợp không có index - như với lệnh INSERT. Tuy nhiên, nếu không có index, cơ sở dữ liệu phải đọc toàn bộ bảng để tìm ra hàng cần xóa. Điều này có nghĩa rằng việc xóa thì nhanh hơn nhưng việc tìm ra hàng để xóa thì lại rất chậm.

Tuy nhiên có thể có khả năng là câu lệnh DELETE thực thi tốt hơn khi không có index như là việc có thể lệnh INSERT thực thi khi không có index tốt hơn nếu truy vấn tác động đến phần lớn của bảng.

Một lệnh DELETE không có mệnh đề WHERE là một ví dụ rõ ràng mà database không sử dụng index, mặc dù đây là trường hợp đặc biệt mà có lệnh SQL riêng: truncate table.

* UPDATE

Một lệnh UPDATE phải di rời các phần tử chỉ mục thay đổi để duy trì thứ tự index. Để làm điều này thì cơ sở dữ liệu phải xóa phần tử chỉ mục cũ ban đầu và thêm vào phần tử chỉ mục mới tại vị trí mới. Thời gian phản hổi cơ bản là giống như là riêng từng lệnh DELETE và INSERT cùng thực hiện vậy.

Như lệnh INSERT hay DELETE thì hiệu suất của lệnh UPDATE cũng phụ thuộc vào số index trên bảng. Sự khác biệt duy nhất là câu lệnh UPDATE không nhất thiết phải ảnh hưởng tới tất cả các cột vì chúng chỉ sửa đổi một số cột được chọn. Hệ quả là lệnh UPDATE không nhất thiết ảnh hưởng đến tất cả các index trên bảng nhưng vẫn ảnh hưởng tới các index có chứa trong các cột được cập nhật.

Thời gian cho hai lệnh UPDATE: một lệnh thì update tất cả các cột và ảnh hưởng tới tất cả các index, một lệnh chỉ update một cột và chỉ ảnh hưởng tới một index.

Hiệu suất của UPDATE theo số chỉ mục :

Lệnh UPDATE trên tất cả các cột cho thấy cùng một pattern ta đã quan sát được trong các phần trước: thời gian phản ứng tăng lên với mỗi một index được thêm, thời gian phản hổi với UPDATE trên một cột không tăng nhiều bởi hầu hết các chỉ mục không phải cập nhật.

Để tối ưu hóa hiệu suất của lệnh UPDATE, bạn chỉ cần phải lưu tâm tới những cột bị cập nhật. Điều này là hiển nhiên nếu chạy các câu lệnh cập nhật thủ công. Tuy nhiên nếu sử dụng công cụ ORM thì câu lệnh UPDATE có thể được tạo ra luôn luôn cập nhật tất cả các cột.

Câu hỏi đặt ra là: Bạn có thể nghĩ ra trường hợp nào mà câu lệnh INSERT hoặc DELETEkhông ảnh hưởng tới tất cả các index của một bảng.

Kết luận : Ở bài viết này, ta có thể thấy rằng việc cập nhật dữ liệu sẽ bị ảnh hưởng hiệu suất khá nhiều khi chúng ta sử dụng index. Chính vì điều này, việc cân nhắc khi bạn cần sử dụng index cho một bảng nào đó là không hề thừa bởi nó sẽ làm giảm hiệu năng chương trình của bạn nếu nó là dư thừa, không cần thiết.

Tham khảo :

<https://freetuts.net/tao-unique-trong-mysql-317.html>

<https://viblo.asia/p/insert-delete-va-update-bi-anh-huong-nhu-the-nao-khi-danh-index-oOVlYW3zK8W>