**Trường Đại Học Khoa Học Tự Nhiên TP.HCM**

**Báo Cáo Đồ Án**

**Cơ Sở Sữ Liệu**

**Tìm Hiểu MySQL**

**Giới thiệu về MySQL**

Trong thế giới máy tính, chúng ta cần liên tục lưu trữ và truy xuất dữ liệu của mình đến và đi từ cơ sở dữ liệu. Dữ liệu có thể được lưu trữ dưới dạng cơ sở dữ liệu, cấu trúc tệp hoặc thư mục.

Chúng tôi sử dụng các phần mềm cơ sở dữ liệu khác nhau để lưu trữ và trích xuất các kết quả đến và từ cơ sở dữ liệu của chúng tôi. Chúng bao gồm MySQL, MS Access, Oracle, PostgreSQL, IBM DB2, MongoDB trong số những phần mềm khác. Trong số này, MySQL là hệ thống cơ sở dữ liệu được sử dụng phổ biến nhất được sử dụng để lưu trữ và truy xuất dữ liệu.

**Lịch sử của MySQL**

MySQL được thành lập vào năm 1995 bởi MySQL AB, một doanh nghiệp Thụy Điển. Michael Widenius (Monty), David Axmark và Allan Larsson là những người tạo ra nền tảng. Mục tiêu chính là cung cấp cho người tiêu dùng các giải pháp quản lý dữ liệu hiệu quả và đáng tin cậy, cả ở nhà và tại cơ quan. Đến năm 2000, hơn nửa tá phiên bản alpha và beta của nền tảng này đã được tung ra. Hầu như tất cả các nền tảng chính đều được hỗ trợ bởi các phiên bản này.

**1) Nguồn mở:**

Ban đầu thuộc sở hữu của MySQL AB, nền tảng này trở thành mã nguồn mở vào năm 2000 và bắt đầu tuân theo các điều kiện GPL. Chuyển sang sử dụng mã nguồn mở dẫn đến doanh thu giảm đáng kể, doanh thu này sau đó đã được lấy lại. Do tính chất nguồn mở của MySQL, nó có sẵn cho sự đóng góp của các nhà phát triển bên thứ ba.

**2) Mở rộng kinh doanh:**

MySQL ngày càng trở nên phổ biến đối với cả người dùng gia đình và chuyên nghiệp, với 2 triệu lượt cài đặt đang hoạt động vào năm 2001. Ngoài trụ sở chính ở Thụy Điển, công ty đã mở rộng phạm vi hoạt động vào năm 2002 bằng cách thiết lập sự hiện diện tại Hoa Kỳ. Trong cùng năm, có thông báo rằng số lượng thành viên của nền tảng đã vượt qua 3 triệu người, với tổng doanh thu là 6,5 triệu đô la.

**3) Vụ kiện đầu tiên:**

Vào tháng 6 năm 2001, NuSphere đã kiện MySQL AB tại Tòa án Quận Hoa Kỳ ở Boston, đây là trận chiến pháp lý lớn đầu tiên của công ty. Các vi phạm hợp đồng của bên thứ ba và cạnh tranh không lành mạnh nằm trong số các cáo buộc. MySQL AB đã kiện NuSphere vì vi phạm bản quyền và nhãn hiệu vào năm 2002. Sau phiên điều trần sơ bộ vào ngày 27 tháng 2 năm 2002, cả hai công ty đã đạt được thỏa thuận.

**4) Thay đổi trong chiến lược:**

Nền tảng này ngày càng trở nên phổ biến và đến cuối năm 2003, nó đã tạo ra doanh thu 12 triệu đô la và có 4 triệu lượt cài đặt đang hoạt động. Năm 2004, công ty quyết định tập trung nhiều hơn vào doanh thu liên tục của người dùng cuối thay vì phí giấy phép một lần. Ý tưởng này đã được đền đáp với doanh thu ròng 20 triệu đô la vào cuối năm.

Những gì bạn có thể tìm thấy trong bài viết này:

* Việc mua lại Innobase của Oracle
* Mua lại Oracle tiếp theo
* MySQL được mua lại bởi Sun Microsystems
* Việc mua lại Sun và MySQL của Oracle
* MySQL Forks
* MySQL và Điện toán đám mây

**Cơ sở dữ liệu là gì?**

Trước khi thành thạo MySQL, điều quan trọng là phải hiểu cơ sở dữ liệu. Cơ sở dữ liệu là một chương trình sắp xếp và lưu một tập hợp các bản ghi. Nó rất đơn giản để người dùng truy cập và quản lý. Nó cho phép chúng ta truy cập nhanh chóng thông tin cần thiết bằng cách tổ chức dữ liệu thành các bảng, hàng, cột và chỉ mục. Mỗi cơ sở dữ liệu có API của nó để thực hiện các tác vụ cơ sở dữ liệu như tạo, quản lý, truy cập và tìm kiếm thông tin mà nó nắm giữ. Nhiều cơ sở dữ liệu có sẵn ngày nay, bao gồm MySQL, Sybase, Oracle, MongoDB, PostgreSQL, SQL Server và những cơ sở khác. Trong phần này, chúng ta sẽ chủ yếu tập trung vào MySQL.

**Thuật ngữ RDBMS:**

RDBMS là viết tắt của Relational Database Management System, và nó là một hệ thống quản lý cơ sở dữ liệu dựa trên mô hình quan hệ của EF Codd. Dữ liệu được giữ trong các quan hệ (bảng) và được biểu diễn dưới dạng các bộ giá trị trong mô hình quan hệ (các hàng).

RDBMS là một hệ quản trị cơ sở dữ liệu quan hệ. Cơ sở dữ liệu quan hệ là một tập hợp các bảng được tổ chức tốt được liên kết với nhau và từ đó có thể dễ dàng truy cập dữ liệu. Ngày nay, cơ sở dữ liệu được sử dụng rộng rãi nhất là cơ sở dữ liệu quan hệ.

**Bảng là gì?**

Bảng là tập hợp các thành phần dữ liệu được tổ chức theo hàng và cột ở định dạng cơ sở dữ liệu quan hệ. Một bảng cũng có thể được coi là một cách thuận tiện để hiển thị các mối quan hệ. Mặt khác, một bảng có thể có các hàng dữ liệu trùng lặp, trong khi một quan hệ thực sự thì không thể. Hình thức lưu trữ dữ liệu cơ bản nhất là bảng. Dưới đây là một ví dụ về bảng Nhân viên.

|  |  |  |  |
| --- | --- | --- | --- |
| **TÔI** | **Tên** | **Tuổi** | **Lương** |
| 1 | Adam | 34 | 13000 |
| 2 | Alex | 28 | 15000 |
| 3 | Stuart | 20 | 18000 |
| 4 | Ross | 42 | 19020 |

**Tuple là gì?**

Tuple, Record hoặc Row là một mục nhập trong một bảng. Một bộ trong bảng là một tập hợp các dữ liệu có liên quan. Ví dụ, bảng Nhân viên có bốn bộ dữ liệu / bản ghi / hàng.

Dưới đây là một ví dụ về một bản ghi hoặc bộ ghi.

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | Adam | 34 | 13000 |

**Thuộc tính là gì?**

Một bảng được tạo thành từ nhiều bản ghi (hàng), mỗi bản ghi có thể được chia thành Thuộc tính, là những phần dữ liệu nhỏ hơn. ID, Tên, Tuổi và Mức lương là bốn thuộc tính trong bảng Nhân viên ở trên.

Miền thuộc tính

Miền thuộc tính được xác định khi một thuộc tính trong một quan hệ (bảng) được xác định để chỉ chứa một loại giá trị cụ thể.

Kết quả là, đối với mỗi bộ, thuộc tính Name sẽ lưu trữ tên của nhân viên. Sẽ là vi phạm khái niệm Cơ sở dữ liệu quan hệ nếu chúng ta lưu địa chỉ của nhân viên ở đó.

|  |
| --- |
| **Tên** |
| Adam |
| Alex |
| Stuart - 9/401, OC Street, Amsterdam |
| Ross |

**Lược đồ quan hệ là gì?**

Một lược đồ quan hệ giải thích cấu trúc của mối quan hệ, bao gồm tên của kết nối (tên bảng), đặc điểm của nó, cũng như tên và kiểu của nó.

**Khóa quan hệ là gì?**

Khóa quan hệ là một thuộc tính có thể được sử dụng để xác định một bộ (hàng) cụ thể trong một quan hệ (bảng).

**Ràng buộc về tính toàn vẹn quan hệ**

Khóa quan hệ là một thuộc tính có thể được sử dụng để xác định một bộ (hàng) cụ thể trong một quan hệ (bảng).

Để trở thành một kết nối hợp pháp trong mô hình cơ sở dữ liệu quan hệ, mỗi quan hệ phải tuân theo hoặc tuân theo một tập hợp các ràng buộc được gọi là Ràng buộc toàn vẹn quan hệ.

Sau đây là ba Ràng buộc Chính trực:

1. Những ràng buộc chính
2. Ràng buộc miền
3. Các ràng buộc về tính toàn vẹn tham chiếu

**Những hạn chế chính:**

Chúng tôi giữ dữ liệu trong các bảng để có thể truy cập sau khi cần. Để truy xuất dữ liệu từ bảng, một hoặc nhiều thuộc tính được kết hợp trong mỗi bảng. Ràng buộc Chính nói rằng bất kỳ mối quan hệ (bảng) nào phải có một thuộc tính (cột) có thể được sử dụng để lấy dữ liệu cho bất kỳ bộ (hàng) nào.

Đối với hai hàng dữ liệu khác nhau, thuộc tính Khóa không bao giờ được NULL hoặc giống nhau.

Ví dụ, trong cơ sở dữ liệu Nhân viên, ID thuộc tính có thể được sử dụng để truy xuất dữ liệu cho từng nhân viên. ID không có giá trị rỗng và là duy nhất cho mỗi hàng, vì vậy nó có thể được sử dụng làm thuộc tính Khóa của chúng tôi.

**Ràng buộc miền:**

Ràng buộc miền là các quy tắc chi phối loại giá trị có thể được lưu trữ cho một thuộc tính nhất định.

 Như đã nêu trước đây, chúng tôi không thể lưu trữ địa chỉ của nhân viên trong cột Tên.

Tương tự, một số điện thoại di động không được có nhiều hơn mười chữ số.

**Ràng buộc về tính toàn vẹn tham chiếu:**

Chúng ta sẽ xem xét kỹ hơn điều này sau. Hiện tại, hãy nhớ rằng nếu tôi nói Supraja là bạn gái của tôi, thì một cô gái tên là Supraja cũng phải tồn tại để mối quan hệ đó tồn tại.

Nếu một bảng tham chiếu đến dữ liệu từ một bảng khác, thì bảng và dữ liệu đó phải có mặt để lưu giữ ràng buộc toàn vẹn tham chiếu.

**MYSQL là gì:**

MySQL là mã nguồn mở hệ quản trị cơ sở dữ liệu SQL quan hệ được sử dụng rộng rãi nhất. MySQL là một hệ quản trị cơ sở dữ liệu quan hệ (RDBMS) phổ biến để xây dựng các ứng dụng phần mềm dựa trên web.

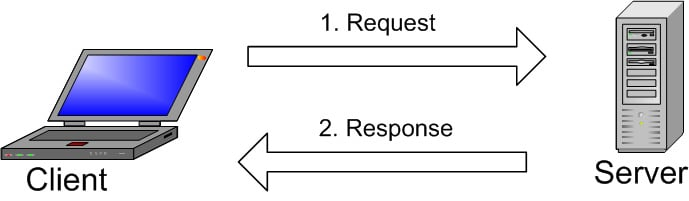
 Bài học này sẽ giúp bạn tăng tốc độ với MySQL và giúp bạn lập trình thoải mái với nó.

**Cơ sở dữ liệu MySQL:**

MySQL là một hệ quản trị cơ sở dữ liệu quan hệ (RDBMS) nhanh chóng, dễ sử dụng được nhiều doanh nghiệp lớn và nhỏ sử dụng. MySQL AB, một doanh nghiệp Thụy Điển, chịu trách nhiệm phát triển, tiếp thị và hỗ trợ. MySQL đang trở nên phổ biến vì nhiều lý do.

* MySQL là một cơ sở dữ liệu mã nguồn mở và miễn phí. Kết quả là, không có chi phí để sử dụng nó.
* MySQL là một chương trình cực kỳ mạnh mẽ. Nó có thể xử lý một phần đáng kể các tính năng được tìm thấy trong các giải pháp cơ sở dữ liệu phức tạp và đắt tiền nhất.
* MySQL sử dụng phiên bản chuẩn của ngôn ngữ dữ liệu SQL được sử dụng rộng rãi.
* MySQL tương thích với một loạt các hệ điều hành và ngôn ngữ, bao gồm PHP, PERL, C, C ++, JAVA và các ngôn ngữ khác.
* MySQL là một cơ sở dữ liệu nhanh, hoạt động tốt ngay cả khi xử lý một lượng lớn dữ liệu.
* MySQL đặc biệt thân thiện với PHP, ngôn ngữ phát triển web phổ biến nhất.
* MySQL có thể xử lý cơ sở dữ liệu khổng lồ với tối đa 50 triệu hàng trong một bảng. Giới hạn kích thước tệp mặc định cho bảng là 4GB, tuy nhiên, bạn có thể mở rộng nó lên giới hạn lý thuyết là 8 triệu gigabyte (giả sử hệ điều hành của bạn có thể xử lý nó) (TB).
* MySQL có thể thích ứng. Phần mềm MySQL có thể được sửa đổi để phù hợp với nhiều ngữ cảnh khác nhau nhờ vào giấy phép GPL nguồn mở.

**SQL của tôi hoạt động như thế nào?**



Sơ đồ mô tả kiến ​​trúc máy khách-máy chủ ở dạng cơ bản nhất của nó. Thông qua một mạng chuyên biệt, một hoặc nhiều thiết bị (máy khách) kết nối với một máy chủ. Mọi máy khách đều có thể gửi yêu cầu bằng giao diện người dùng đồ họa (GUI) trên màn hình của họ và máy chủ sẽ phản hồi với kết quả dự kiến ​​miễn là cả hai bên đều hiểu yêu cầu. Không cần quá kỹ thuật, sau đây là các quy trình chính xảy ra trong môi trường MySQL:

* MySQL tạo cơ sở dữ liệu để lưu trữ và thao tác dữ liệu, với mối quan hệ của mỗi bảng được xác định.
* Khách hàng có thể gửi yêu cầu đến MySQL bằng cách nhập các câu lệnh SQL được chỉ định.
* Thông tin được yêu cầu sẽ được ứng dụng máy chủ trả về và hiển thị ở phía khách hàng.

Thats tất cả để có nó. Khách hàng thường nhấn mạnh MySQL GUI nào nên được sử dụng. Hoạt động quản lý dữ liệu của họ càng nhanh và dễ dàng thì GUI càng nhẹ và thân thiện với người dùng. MySQL WorkBench, SequelPro, DBVisualizer và Công cụ quản trị Navicat DB là một số GUI của MySQL phổ biến nhất. Một số là miễn phí, trong khi những người khác phải trả phí; một số chỉ dành riêng cho macOS, trong khi một số khác tương thích với tất cả các hệ điều hành chính. Khách hàng nên chọn GUI phù hợp nhất với nhu cầu của họ. Cách tốt nhất để bảo trì cơ sở dữ liệu web, bao gồm cả trang WordPress, là phpMyAdmin.

**Các tính năng chính của MYSQL:**

MySQL là một hệ quản trị cơ sở dữ liệu quan hệ (RDBMS) sử dụng các truy vấn SQL (Ngôn ngữ truy vấn có cấu trúc) để thực hiện các hoạt động. Đây là một trong những ngôn ngữ được sử dụng rộng rãi nhất để truy cập và thao tác với các bản ghi bảng. Theo giấy phép GNU, MySQL là phần mềm mã nguồn mở và miễn phí. Tập đoàn Oracle ủng hộ nó.

Sau đây là một số tính năng cần thiết nhất của MySQL:

1. **Hệ thống quản lý cơ sở dữ liệu quan hệ (RDBMS):**

MySQL là một hệ quản trị cơ sở dữ liệu sử dụng mô hình quan hệ. Các truy vấn SQL được sử dụng để truy xuất và quản lý các bản ghi của bảng bằng ngôn ngữ cơ sở dữ liệu này.

2. **Dễ sử dụng:**

MySQL là một cơ sở dữ liệu đơn giản để làm việc. Chúng ta chỉ cần học các nguyên tắc cơ bản của SQL. Chỉ cần một số câu lệnh SQL đơn giản để xây dựng và tương tác với MySQL.

3. **Bảo mật:**

MySQL có một lớp bảo mật dữ liệu mạnh để giữ an toàn cho dữ liệu nhạy cảm khỏi tin tặc. Ngoài ra, MySQL mã hóa mật khẩu.

4. **Kiến trúc Máy khách / Máy chủ:**

Kiến trúc máy khách / máy chủ được theo sau bởi MySQL. Máy chủ cơ sở dữ liệu (MySQL) và số lượng không giới hạn máy khách (ứng dụng ứng dụng) giao tiếp với máy chủ, cho phép chúng truy vấn dữ liệu, thực hiện thay đổi, v.v.

5. **mã nguồn mở:**

MySQL được sử dụng miễn phí, do đó chúng ta có thể tải miễn phí từ trang web chính thức của MySQL.

6. **Có thể mở rộng:**

MySQL có thể mở rộng vì nó hỗ trợ đa luồng. Nó có thể xử lý hầu như mọi lượng dữ liệu, lên đến 50 triệu hàng hoặc hơn. Khoảng 4 GB là giới hạn kích thước tệp mặc định. Tuy nhiên, về mặt lý thuyết, chúng ta có thể tăng con số này lên 8 TB dữ liệu.

**7. Tốc độ:**

MySQL được coi là một trong những ngôn ngữ cơ sở dữ liệu nhanh nhất, được chứng minh bằng nhiều bài kiểm tra điểm chuẩn.

**8. Tính linh hoạt cao:**

MySQL đặc biệt thích ứng vì nó hỗ trợ một loạt các ứng dụng nhúng.

**9. Tương thích trên nhiều hệ điều hành:**

MySQL có thể chạy trên nhiều hệ điều hành khác nhau, bao gồm Novell NetWare, Windows \* Linux \*, UNIX \* (bao gồm Sun \* Solaris \*, AIX và DEC \* UNIX), OS / 2, FreeBSD \* và các hệ điều hành khác. MySQL cũng cho phép các máy khách chạy trên cùng một máy tính với máy chủ hoặc một máy riêng biệt (giao tiếp qua mạng cục bộ hoặc Internet).

10. **Cho phép khôi phục:**

MySQL cho phép bạn khôi phục, cam kết và phục hồi sau sự cố.

**11 Hiệu quả bộ nhớ:**

Hiệu quả của nó là tốt vì rò rỉ bộ nhớ là tối thiểu.

**12. Hiệu suất cao:**

Do cấu trúc công cụ lưu trữ độc đáo của nó, MySQL nhanh hơn, đáng tin cậy hơn và ít tốn kém hơn. Nó đạt được kết quả hiệu suất đặc biệt tốt khi so sánh với các cơ sở dữ liệu khác mà không phải hy sinh bất kỳ tính năng quan trọng nào của phần mềm. Do bộ nhớ đệm khác nhau nên nó có tiện ích tải nhanh.

**13. Năng suất cao:**

Các trình kích hoạt, thủ tục được lưu trữ và chế độ xem đều được sử dụng trong MySQL để giúp các nhà phát triển làm việc hiệu quả hơn.

14. **Nền tảng độc lập:**

Nó có thể được tải xuống, cài đặt và chạy trên hầu hết các hệ điều hành **.**

**15. Phân vùng:**

Tính năng này tăng cường hiệu suất và cho phép xử lý nhanh chóng các cơ sở dữ liệu khổng lồ.

**16. Hỗ trợ GUI:**

MySQL cung cấp “MySQL Workbench”, một công cụ giao diện người dùng đồ họa cơ sở dữ liệu trực quan thống nhất dành cho kiến ​​trúc sư cơ sở dữ liệu, nhà phát triển và quản trị viên cơ sở dữ liệu. MySQL Workbench bao gồm phát triển SQL, lập mô hình dữ liệu, di chuyển dữ liệu và cấu hình máy chủ, quản trị người dùng, sao lưu và các tiện ích quản trị khác. Từ phiên bản MySQL Server 5.6 trở lên, có đầy đủ GUI hỗ trợ17.

**17. Hỗ trợ mật khẩu kép:**

MySQL phiên bản 8.0 hỗ trợ mật khẩu kép: một là mật khẩu hiện tại và mật khẩu còn lại là mật khẩu dự phòng mà chúng ta có thể sử dụng để chuyển sang mật khẩu mới.

**Phiên bản MySQL:**

Tạo phiên bản là quá trình gán tên hoặc số phiên bản riêng biệt cho một nhóm chương trình phần mềm khi chúng được tạo và phân phối. Thuật ngữ “phiên bản 1.0” được sử dụng rộng rãi để chỉ bản phát hành ban đầu của phần mềm hoặc chương trình. Không có hướng dẫn tiêu chuẩn ngành nào điều chỉnh cách định dạng số phiên bản. Do đó, mỗi công ty đều có hệ thống đặt tên cho các phiên bản phần mềm. Khi các tính năng mới trong phần mềm và chương trình được thêm vào, các lỗi được sửa và các lỗ hổng bảo mật được cắm vào, số phiên bản sẽ được thay đổi để phản ánh những tiến bộ này.

Phiên bản hỗ trợ MySQL gần đây nhất là phiên bản v5.8. Nhiều sửa đổi quan trọng đã được thực hiện, bao gồm việc bổ sung và loại bỏ các tính năng mới, sửa lỗi và các vấn đề bảo mật, v.v. Phiên bản này cung cấp lịch sử phát hành MySQL 8.0 đến MySQL 8.0.21. Nó sẽ có sẵn bắt đầu từ tháng 4 năm 2018 và sẽ được hỗ trợ đến hết tháng 4 năm 2026.

 Khi cài đặt MySQL trên PC, bạn phải chọn phiên bản và định dạng phân phối mà bạn muốn sử dụng. MySQL có sẵn trong hai phiên bản, phiên bản đầu tiên là bản phát triển phát triển và phiên bản thứ hai là bản phát hành Tính khả dụng chung (GA). Bản phát triển chứa tính năng mới nhất và không được sử dụng trong sản xuất. Bản phát hành GA, thường được gọi là bản sản xuất hoặc bản phát hành ổn định, chủ yếu được sử dụng để sản xuất. Do đó, phải chọn bản phát hành Tính sẵn có chung gần đây nhất.

Hãy xem có gì mới trong MySQL 8.0.

**Các tính năng được thêm vào MySQL 8.0:**

Các tính năng sau đã được thêm vào MySQL 8.0:

Từ điển dữ liệu: Từ điển dữ liệu giao dịch được sử dụng để chứa thông tin về các đối tượng cơ sở dữ liệu. Dữ liệu trước đây được lưu giữ trong tệp siêu dữ liệu và bảng không giao dịch trong các phiên bản trước.

Tuyên bố DDL nguyên tử: Đó là một tuyên bố DDL nguyên tử kết hợp các hoạt động của công cụ lưu trữ, các thay đổi từ điển dữ liệu và nhật ký nhị phân được liên kết với hoạt động DDL thành một giao dịch nguyên tử duy nhất.

Quy trình nâng cấp: Trước đây, khi phiên bản MySQL mới được cài đặt, bảng từ điển dữ liệu sẽ tự động được nâng cấp vào lần khởi động tiếp theo và DBA dự kiến ​​sẽ hoàn thành thủ công quá trình nâng cấp bằng lệnh nâng cấp MySQL. Sau MySQL 8.0.16, DBA không còn bắt buộc phải sử dụng lệnh nâng cấp MySQL để hoàn tất nâng cấp.

Bảo mật và quản lý tài khoản: Một số cải tiến đã được thực hiện để tăng cường bảo mật và giúp DBA linh hoạt hơn trong việc quản lý tài khoản.

Quản lý tài nguyên: MySQL hiện cho phép bạn xây dựng và hỗ trợ các nhóm tài nguyên, cũng như chỉ định các luồng cho mỗi nhóm để nó có thể thực thi dựa trên các tài nguyên có sẵn cho nó. Việc sử dụng tài nguyên của các luồng trong một nhóm có thể được kiểm soát bởi các đặc tính của nhóm.

Quản lý bảng mã hóa: Mã hóa bảng hiện được kiểm soát ở cấp hệ thống bằng cách thiết lập và thực thi các mặc định mã hóa. Khi xây dựng một lược đồ và không gian bảng chung, biến mã hóa bảng mặc định hoặc mệnh đề DEFAULT ENCRYPTION cung cấp mặc định mã hóa.

Các cải tiến của InnoDB: Bộ đếm tự động tăng, tham nhũng cây chỉ mục, plugin Memcached, phát hiện bế tắc InnoDB, tính năng mã hóa vùng bảng, công cụ lưu trữ, máy chủ chuyên dụng InnoDB, thư viện Zlib và nhiều cải tiến khác của InnoDB đã được bao gồm.

Hỗ trợ bộ ký tự: latin1 đã được thay thế bằng utf8mb4 làm bộ ký tự mặc định. Nhiều ảnh ghép mới được đưa vào bộ ký tự mới, đáng chú ý là utf8MB JA 0900 dưới dạng cs.

Hỗ trợ kiểu dữ liệu: Nó có thể hỗ trợ việc sử dụng các biểu thức làm giá trị mặc định trong các đặc tả kiểu dữ liệu.

Cải tiến của Trình tối ưu hóa: Phiên bản này bao gồm các cải tiến đối với trình tối ưu hóa, chẳng hạn như chỉ mục vô hình, chỉ mục giảm dần và khả năng tạo chỉ mục chức năng. Nó có thể so sánh một cột với một giá trị không đổi bằng cách sử dụng cách gấp liên tục.

Chức năng cửa sổ: Nhiều chức năng cửa sổ mới, chẳng hạn như RANK (), LAG () và NTILE, được hỗ trợ trong phiên bản này ().

Cải tiến JSON: Trong chức năng json của MySQL, các cải tiến hoặc bổ sung sau được thực hiện: Toán tử đường dẫn nội tuyến (- >>), hàm tổng hợp json JSON ARRAYAGG () và JSON OBJECTAGG (), hàm tiện ích JSON PRETTY (), JSON STORAGE SIZE () , JSON STORAGE MIỄN PHÍ (), JSON STORAGE FREE (), JSON STORAGE FREE (), JSON STORAGE FREE (), JSON STORAGE FREE (), JSON STORAGE FREE () (). Thay vì kích thước 1K đã đặt, mỗi giá trị hiện được biểu thị bằng một phần có độ dài thay đổi của khóa sắp xếp khi sắp xếp các giá trị json. Nó cũng bao gồm hàm hợp nhất JSON MERGE PATCH để kết hợp hai đối tượng json, cũng như hàm JSON TABLE ().

**Ai sử dụng MYSQL:**

Nhiều chương trình web hướng cơ sở dữ liệu, chẳng hạn như Drupal, Joomla, phpBB và WordPress, sử dụng MySQL. Nhiều trang web lớn, chẳng hạn như Facebook, Flickr, MediaWiki, Twitter và YouTube, sử dụng MySQL. Tất cả các công ty lớn đều sử dụng MySQL.

**Các nền tảng được hỗ trợ MySQL:**

Phiên bản hiện tại của MySQL Workbench là 8.0 và nó được khuyến nghị cho MySQL 8.0. Nó cũng hỗ trợ MySQL 5.6 và 5.7.

Xin lưu ý rằng MySQL Workbench cho Linux là một ứng dụng Gnome chỉ được hỗ trợ chính thức trên máy tính để bàn Gnome, tuy nhiên, nó cũng sẽ hoạt động bình thường trên các môi trường máy tính để bàn khác.

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | **8.0** | **5,7** |
| **Hệ điều hành** | **Ngành kiến ​​trúc** |  |  |
| **Oracle Linux / Red Hat / CentOS** |  |  |  |
| Oracle Linux 8 / Red Hat Enterprise Linux 8 / CentOS 8 | x86\_64, ARM 64 | • |  |
| Oracle Linux 7 / Red Hat Enterprise Linux 7 / CentOS 7 | ARM 64 | • |  |
| Oracle Linux 7 / Red Hat Enterprise Linux 7 / CentOS 7 | x86\_64 | • | • |
| Oracle Linux 6 / Red Hat Enterprise Linux 6 / CentOS 6 | x86\_32, x86\_64 | • | • |
| **Oracle Solaris** |  |  |  |
| Solaris 11 (Cập nhật 4+) | SPARC\_64 | • | • |
| **Canonical** |  |  |  |
| Ubuntu 21.04 | x86\_64 | • |  |
| Ubuntu 20.04 LTS | x86\_64 | • |  |
| Ubuntu 18.04 LTS | x86\_32, x86\_64 | • | • |
| **SUSE** |  |  |  |
| SUSE Enterprise Linux 15 / OpenSUSE 15 (15.2) | x86\_64 | • |  |
| SUSE Enterprise Linux 12 (12.5+) | x86\_64 | • | • |
| **Debian** |  |  |  |
| Debian GNU / Linux 10 | x86\_64 | • | • |
| **Microsoft Windows Server** |  |  |  |
| Máy chủ Microsoft Windows 2019 | x86\_64 | • |  |
| Máy chủ Microsoft Windows 2016 | x86\_64 | • | • |
| Microsoft Windows 2012 Server R2 | x86\_64 | • | • |
| **Microsoft Windows** |  |  |  |
| Microsoft Windows 10 | x86\_64 | • | • |
| **quả táo** |  |  |  |
| macOS 11 | x86\_64, ARM\_64 | • |  |
| macOS 10.15 | x86\_64 | • |  |
| **Linux khác nhau** |  |  |  |
| Linux chung (định dạng tar) | x86\_32, x86\_64, glibc 2.12, libstdc ++ 4.4 | • | • |
| [Yum Repo](https://dev.mysql.com/downloads/repo/yum/) | • | • |  |
| [APT Repo](https://dev.mysql.com/downloads/repo/apt/) | • | • |  |
| [SUSE Repo](https://dev.mysql.com/downloads/repo/suse/) |  |  |  |

Yêu cầu phần cứng tối thiểu:

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  | **Tối thiểu** | **Khuyến khích** |
| CPU | CPU x86 64bit | CPU đa lõi 64bit x86, RAM 8 GB |
| RAM | 4 GB | 8 GB trở lên |
| Trưng bày | 1024 × 768 | 1920 × 1200 trở lên |

**Yêu cầu hệ thống tối thiểu cho MySQL Enterprise Monitor Service Manager:**

* một CPU có hai hoặc nhiều lõi
* Đĩa có hệ thống con I / O RAM từ 2 GB trở lên thích hợp cho việc cài đặt cơ sở dữ liệu yêu cầu ghi nhiều.

Yêu cầu hệ thống được khuyến nghị: (Nếu bạn đang xem hơn 100 máy chủ MySQL):

* CPU có bốn lõi trở lên
* Đĩa có ít nhất 8 GB RAM RAID 10, RAID 0 + 1) Hệ thống con I / O phù hợp để cài đặt cơ sở dữ liệu chuyên sâu
* \* Sử dụng Tác nhân tích hợp để giám sát từ xa các Phiên bản MySQL hoạt động trên các hệ thống không được hỗ trợ. Lưu ý rằng giám sát từ xa không nắm bắt thông tin hệ điều hành; để biết thêm chi tiết, hãy đọc tài liệu trực tuyến.

**Các công cụ bên thứ ba được hỗ trợ của MySQL:**

4.9 Công cụ của bên thứ ba và cấu hình MySQL

Phần mềm của bên thứ ba có thể xác định phiên bản MySQL từ nguồn MySQL bằng cách đọc tệp VERSION trong thư mục nguồn cấp cao nhất. Các phần của phiên bản được liệt kê riêng trong tệp. Ví dụ: nếu phiên bản là MySQL 5.7.4-m14, tệp sẽ giống như sau:

PHIÊN BẢN MYSQL MAJOR = 5

PHIÊN BẢN MYSQL MINOR = 7

MYSQL VERSION PATCH = 4

PHIÊN BẢN MYSQL EXTRA = -m14.

Giá trị MYSQL VERSION EXTRA không có giá trị gì nếu nguồn không dành cho bản phát hành Tính khả dụng chung của Máy chủ MySQL (GA). Giá trị tương ứng với Mốc 14 trong ví dụ trước.

Đối với các bản phát hành NDB Cluster (bao gồm cả các bản phát hành GA của NDB Cluster), MYSQL VERSION EXTRA cũng không giống như vậy, như được minh họa ở đây:

PHIÊN BẢN MYSQL MAJOR = 5

PHIÊN BẢN MYSQL MINOR = 7

MYSQL VERSION PATCH = 32

MYSQL VERSION EXTRA = -ndb-7.5.21

Sử dụng công thức sau để nhận một số có năm chữ số từ các thành phần phiên bản:

MYSQL\_VERSION\_MAJOR \* 10000 + MYSQL\_VERSION\_MINOR \* 100 + MYSQL\_VERSION\_PATCH

**Thiết lập tài khoản người dùng MySQL:**

Xin vui lòng đăng nhập.

Đi tới máy chủ đám mây của bạn và đăng nhập.

Để kết nối với MySQL, hãy nhập lệnh sau:

MySQL -u root -p

Bạn sẽ được yêu cầu nhập mật khẩu gốc MySQL của mình (lưu ý rằng mật khẩu này không giống với mật khẩu gốc của Máy chủ đám mây).

Tạo một tài khoản mới cho chính bạn:

Bạn có thể tạo người dùng mới và định cấu hình mật khẩu của họ cùng lúc, như được minh họa trong lệnh ví dụ bên dưới, lệnh này tạo người dùng với kiểm tra tên người dùng:

TẠO NGƯỜI DÙNG 'test' @ 'localhost' VỚI 'mật khẩu mới' NHƯ IDENTIFIER;

Sau đó, bạn phải xóa các đặc quyền, điều này khiến MySQL tải lại bảng người dùng. Bước này phải được hoàn thành mỗi khi bạn thêm hoặc chỉnh sửa người dùng.

Lệnh xóa đặc quyền được hiển thị trong ví dụ sau:

QUYỀN RIÊNG TƯ FLUSH;

Xong.

Quyền - Chọn

Tạo một tài khoản mới cho chính bạn:

Đặt quyền của người dùng mới.

Người dùng mới (thử nghiệm) của bạn không có quyền và không thể làm bất cứ điều gì tại thời điểm này. Bắt đầu với quyền SELECT (chỉ đọc) trên tất cả các cơ sở dữ liệu có sẵn, bạn có thể muốn bắt đầu thiết lập quyền. Bằng cách sử dụng lệnh sau, bạn có thể cấp các quyền cụ thể:

CẤP CHỌN BẬT \*. \* ĐẾN 'test' @ 'localhost'; Quyền - Tất cả

Thêm cơ sở dữ liệu mới và cấp toàn quyền truy cập để kiểm tra, cho phép họ tạo, đọc, chỉnh sửa và xóa bản ghi, như được minh họa trong ví dụ dưới đây:

TẠO CƠ SỞ DỮ LIỆU mytestdb1; Bây giờ chúng ta có cơ sở dữ liệu và người dùng, chúng ta có thể gán các đặc quyền: CẤP TẤT CẢ CÁC QUYỀN RIÊNG TƯ TRÊN `mytestdb`. \* ĐẾN 'test' @ 'localhost';

Khi tạo người dùng và cơ sở dữ liệu cho ứng dụng web, đây là loại quyền bạn sẽ muốn sử dụng. Không có yêu cầu đối với người dùng để truy cập vào bất kỳ cơ sở dữ liệu nào khác.

Chạy lệnh sau để xóa các đặc quyền:

QUYỀN RIÊNG TƯ FLUSH;

Đăng nhập bằng tài khoản người dùng mới mà bạn đã tạo…

Đăng nhập vào MySQL với tư cách người dùng mới và chạy lệnh sau để xác minh rằng các quyền bạn đặt là chính xác:

MySQL -u test -p

Điền mật khẩu cho người dùng thử nghiệm khi được nhắc.

Sau khi bạn đã đăng nhập với tư cách người dùng thử nghiệm, hãy thử sử dụng lệnh sau để tạo cơ sở dữ liệu mới:

Thông báo lỗi sau xuất hiện khi bạn chạy TẠO CƠ SỞ DỮ LIỆU mytestdb2:

LỖI 1044 (42000): Quyền truy cập vào cơ sở dữ liệu'mytestdb2 ′ bị từ chối đối với người dùng 'test' @ 'localhost'.

Bởi vì thử nghiệm sử dụng chỉ có TẤT CẢ CÁC QUYỀN RIÊNG TƯ cho cơ sở dữ liệu 'mytestdb' và chỉ CHỌN đặc quyền cho mọi thứ khác, lỗi này xảy ra.

Xóa / loại bỏ một người dùng:

Quy trình xóa người dùng tương tự như quy trình xóa cơ sở dữ liệu. Ví dụ bên dưới minh họa cách xóa người dùng thử nghiệm:

DROP USER 'test' @ 'localhost';

**Lệnh MySQL quản trị:**

Đây là tập hợp các lệnh MySQL quan trọng nhất mà bạn có thể gặp phải khi làm việc với cơ sở dữ liệu MySQL.

* SỬ DỤNG tên cơ sở dữ liệu Để chọn một cơ sở dữ liệu trong vùng làm việc MySQL, hãy nhập tên cơ sở dữ liệu.
* HIỂN THỊ DỮ LIỆU hiển thị danh sách cơ sở dữ liệu mà MySQL DBMS có thể truy cập.
* SHOW TABLES hiển thị các bảng cơ sở dữ liệu sau khi sử dụng lệnh sử dụng để chọn cơ sở dữ liệu.
* HIỂN THỊ CÁC CỘT TỪ tên bảng: Hiển thị các thuộc tính của bảng, loại thuộc tính, thông tin khóa, nếu giá trị NULL được cho phép, giá trị mặc định và dữ liệu khác.
* HIỂN THỊ CHỈ SỐ TỪ tên bảng hiển thị thông tin về tất cả các chỉ mục của bảng, bao gồm cả KHÓA CHÍNH.
* HIỂN THỊ TÌNH TRẠNG BẢNG NHƯ tablenameG hiển thị thông tin về hiệu suất và thống kê của hệ thống quản lý cơ sở dữ liệu MySQL.

**Các kiểu dữ liệu MySQL:**

Định nghĩa thích hợp về các trường của bảng rất quan trọng đối với việc tối ưu hóa tổng thể cơ sở dữ liệu của bạn. Bạn chỉ nên sử dụng loại và kích thước của trường mà bạn thực sự yêu cầu. Ví dụ: nếu bạn biết mình sẽ chỉ sử dụng hai ký tự, đừng tạo trường lớn 10 ký tự. Sau khi sắp xếp dữ liệu, bạn sẽ lưu trữ trong các trường đó, các trường (hoặc cột) này được gọi là kiểu dữ liệu.

MySQL sử dụng nhiều kiểu dữ liệu khác nhau được chia thành ba nhóm.

* Số
* Ngày và giờ
* Các loại chuỗi.

**Kiểu dữ liệu số:**

MySQL hỗ trợ tất cả các kiểu dữ liệu số ANSI SQL tiêu chuẩn, do đó, các định nghĩa này sẽ dễ nhận biết nếu bạn đến từ một hệ thống cơ sở dữ liệu khác.

Các kiểu dữ liệu số phổ biến và mô tả của chúng được liệt kê bên dưới.

INT: là một số nguyên có dấu hoặc không dấu có kích thước chuẩn. Phạm vi được phép, nếu có dấu, là -2147483648 đến 2147483647. Nếu giá trị không có dấu, phạm vi là 0 đến 4294967295. Có thể chỉ định độ rộng của tối đa 11 chữ số.

 TINYINT: là một số nguyên có dấu hoặc không dấu rất nhỏ. Phạm vi được phép, nếu có dấu, là -128 đến 127. Nếu giá trị không có dấu, phạm vi là 0 đến 255. Có thể chỉ định độ rộng của tối đa bốn chữ số.

SMALLINT: Đây là một số nguyên nhỏ có thể có dấu hoặc không dấu. Phạm vi được phép, nếu có dấu, là -32768 đến 32767. Nếu giá trị không được ký, phạm vi là 0 đến 65535. Có thể chỉ định độ rộng của tối đa 5 chữ số.

MEDIUMINT: Đây là một số nguyên cỡ vừa có dấu hoặc không dấu. Phạm vi được phép, nếu có dấu, là -8388608 đến 8388607. Nếu giá trị không được ký, phạm vi là 0 đến 16777215. Có thể chỉ định độ rộng của tối đa 9 chữ số.

BIGINT: Kích thước của một số nguyên rất lớn, có thể có dấu hoặc không dấu. Phạm vi được phép nếu có dấu là -9223372036854775808 đến 9223372036854775807. Nếu giá trị không được ký, phạm vi là 0 đến 18446744073709551615. Có thể chỉ định độ rộng của tối đa 20 chữ số

FLOAT (M, D): Đây là một số dấu phẩy động có dấu không thể không dấu. Cả chiều dài hiển thị (M) và số lượng số thập phân đều có thể được tùy chỉnh (D). Đây là tùy chọn; giá trị mặc định là 10,2, trong đó 2 biểu thị số lượng phần thập phân và 10 biểu thị tổng số chữ số (bao gồm cả số thập phân). Đối với FLOAT, độ chính xác thập phân có thể lên đến 24 chữ số.

DOUBLE (M, D): Một số dấu phẩy động có độ chính xác gấp đôi không thể không dấu. Cả chiều dài hiển thị (M) và số lượng số thập phân đều có thể được tùy chỉnh (D). Đây là tùy chọn; giá trị mặc định là 16,4, trong đó 4 là số thập phân. Đối với DOUBLE, độ chính xác thập phân có thể lên tới 53 chữ số. DOUBLE là một từ đồng nghĩa với REAL.

DECIMAL (M, D): Một số dấu phẩy động có dấu đã được giải nén. Mỗi số thập phân trong số thập phân được giải nén tương ứng với một byte. Cần xác định độ dài hiển thị (M) và số phần thập phân (D). DECIMAL là từ đồng nghĩa với NUMERIC.

**Loại ngày và giờ:**

Sau đây là các kiểu dữ liệu ngày và giờ của MySQL:

DATE: Ngày từ 1000-01-01 đến 9999-12-31 ở định dạng YYYY-MM-DD. Ví dụ, ngày 30 tháng 12 năm 1973 sẽ được giữ là 1973-12-30.

DATETIME: Ngày và giờ kết hợp giữa 1000-01-01 00:00:00 và 9999-12-31 23:59:59 ở định dạng YYYY-MM-DD HH: MM: SS. Ví dụ: vào ngày 30 tháng 12 năm 1973, 3:30 chiều sẽ được giữ là 1973-12-30 15:30:00.

TIMESTAMP: Ngày từ nửa đêm ngày 1 tháng 1 năm 1970 đến năm 2037. Định dạng này có vẻ giống với định dạng DATETIME trước đó, nhưng không có dấu gạch ngang giữa các chữ số; ví dụ: 3:30 chiều ngày 30 tháng 12 năm 1973 sẽ được giữ là 19731230153000. (YYYYMMDDHHMMSS).

TIME: TIME lưu thời gian ở định dạng HH: MM: SS.

YEAR (M): Lưu trữ một năm dưới dạng số có hai chữ số hoặc bốn chữ số. YEAR có thể là bất kỳ năm nào trong khoảng từ 1970 đến 2069 với điều kiện độ dài được chỉ định là 2 (ví dụ, YEAR (2)) (70 đến 69). YEAR có thể nằm trong khoảng từ 1901 đến 2155 nếu độ dài được cho là 4. Độ dài mặc định là 4 ký tự.

**STRING LOẠI:**

Mặc dù các định dạng số và ngày đều mang tính giải trí nhưng phần lớn dữ liệu của bạn sẽ được lưu trữ ở định dạng chuỗi. Sau đây là danh sách các kiểu dữ liệu chuỗi phổ biến nhất trong MySQL.

CHAR (M): Một chuỗi có độ dài cố định (ví dụ: CHAR (5)) có độ dài từ 1 đến 255 ký tự, được đệm bên phải với các khoảng trắng có độ dài được chỉ định khi được lưu trữ. Tuy nhiên, không cần thiết phải chỉ định độ dài, mặc định là 1.

VARCHAR (M): Một chuỗi có độ dài thay đổi với độ dài từ 1 đến 255 ký tự. VARCHAR, chẳng hạn (25). Khi tạo trường VARCHAR, bạn phải chỉ định độ dài.

BLOB HOẶC VĂN BẢN: Một trường có thể chứa tối đa 65535 ký tự. “Đối tượng khổng lồ nhị phân” hay BLOB được sử dụng để lưu trữ khối lượng lớn dữ liệu nhị phân, chẳng hạn như ảnh hoặc các loại tệp khác. Các trường TEXT cũng có thể mang nhiều thông tin. Sự khác biệt giữa hai điều này là trên BLOB, việc sắp xếp và so sánh trên dữ liệu được lưu trữ có phân biệt chữ hoa chữ thường, trong khi trong trường TEXT thì không. Với BLOB và TEXT, bạn không cần phải cung cấp độ dài.

TINYBLOB hoặc TINY TEXT: Độ dài tối đa 255 ký tự cho cột BLOB hoặc TEXT. Với TINYBLOB và TINYTEXT, bạn không phải cung cấp độ dài.

MEDIUM BLOB hoặc MEDIUM TEXT: Độ dài tối đa của cột BLOB hoặc TEXT là 16777215 ký tự. Với MEDIUMBLOB hoặc MEDIUM TEXT, bạn không phải chỉ định độ dài.

DÀI BLOB hoặc DÀI TEXT: Độ dài tối đa của cột BLOB hoặc TEXT là 4294967295 ký tự. Với LONGBLOB và LONGTEXT, bạn không phải xác định độ dài

ENUM: Một danh sách được gọi là một kiểu liệt kê. Bạn đang thiết lập một danh sách các đối tượng mà từ đó giá trị phải được chọn khi bạn xác định một ENUM (hoặc nó có thể là NULL). Nếu bạn muốn trường của mình giữ “A,” “B” hoặc “C”, bạn sẽ xác định ENUM của mình là ENUM ('A,' B, 'C,)

Trước đó, dữ liệu được lưu trữ trong cấu trúc DBMS nhưng bây giờ dữ liệu được lưu trữ trong RDBMS. Sự khác biệt chính giữa DBMS và RDBMS là định dạng dữ liệu. Trong DBMS, dữ liệu được lưu trữ ở định dạng tệp trong khi trong RDBMS, dữ liệu được lưu trữ ở định dạng bảng. Với việc dữ liệu được lưu trữ dưới dạng bảng, dữ liệu trở nên có cấu trúc hơn và dễ đọc và dễ hiểu hơn.

Tất cả các hệ thống cơ sở dữ liệu được đề cập ở trên bao gồm cả MySQL là một dạng cơ sở dữ liệu RDBMS.

Trong MySQL, chủ yếu có 4 loại lệnh cơ sở dữ liệu để lưu trữ và trích xuất dữ liệu.

**Bao gồm các:**

1. [DDL - Ngôn ngữ định nghĩa dữ liệu](https://www.mygreatlearning.com/blog/mysql-tutorial/#DDL)
2. [DML - Ngôn ngữ thao tác dữ liệu](https://www.mygreatlearning.com/blog/mysql-tutorial/#DML)
3. [DCL - Ngôn ngữ điều khiển dữ liệu](https://www.mygreatlearning.com/blog/mysql-tutorial/#DCL)
4. [TCL - Ngôn ngữ kiểm soát giao dịch](https://www.mygreatlearning.com/blog/mysql-tutorial/#TCL)

**DDL:**

DDL hoặc Ngôn ngữ Định nghĩa Dữ liệu bao gồm các lệnh bao gồm việc xác định cấu trúc của lược đồ cơ sở dữ liệu và bảng cơ sở dữ liệu. Nó chủ yếu liên quan đến cấu trúc của lược đồ cơ sở dữ liệu và bảng.

Các lệnh bao gồm trong DDL là:

# Tạo - Được sử dụng để tạo lược đồ bảng

# Drop - Bỏ cơ sở dữ liệu khỏi bộ nhớ

# Alter - Thay đổi cấu trúc của lược đồ bảng

# Cắt ngắn - Xóa tất cả dữ liệu khỏi lược đồ cơ sở dữ liệu

# Nhận xét - Các câu lệnh này chỉ nhằm mục đích hiểu cấu trúc lược đồ. Chúng không đóng góp vào cấu trúc cơ sở dữ liệu thực tế

# Rename - Được sử dụng để đổi tên bảng cơ sở dữ liệu

**DML:**

DML hoặc Ngôn ngữ thao tác dữ liệu bao gồm các lệnh bao gồm thao tác dữ liệu của bảng cơ sở dữ liệu. Nó liên quan đến việc tạo và thao tác dữ liệu của bảng cơ sở dữ liệu.

Các lệnh có trong DML là:

# Chọn: Được sử dụng để trích xuất thông tin từ bảng cơ sở dữ liệu

# Chèn: Dùng để chèn dữ liệu vào bảng cơ sở dữ liệu

# Cập nhật: Cập nhật dữ liệu hiện có trong bảng cơ sở dữ liệu nơi một điều kiện cụ thể được xác định.

# Xóa: Xóa dữ liệu khỏi bảng cơ sở dữ liệu

**DCL:**

DCL hoặc Ngôn ngữ điều khiển dữ liệu bao gồm các lệnh đó để xử lý các quyền, quyền hạn hoặc các điều khiển khác của hệ thống cơ sở dữ liệu. Nó liên quan đến các quyền cụ thể đối với những người dùng cụ thể để truy cập vào cơ sở dữ liệu.

Các lệnh có trong DCL là:

# Grant: Được sử dụng để cấp quyền cho một người dùng cụ thể truy cập vào cơ sở dữ liệu với quyền đọc hoặc ghi.

# Thu hồi: Được sử dụng để xóa quyền truy cập cơ sở dữ liệu của một người dùng cụ thể.

**TCL:**

TCL hoặc Ngôn ngữ điều khiển giao dịch bao gồm các lệnh xử lý các giao dịch của cơ sở dữ liệu, bao gồm tập hợp các câu lệnh giữa máy khách và máy chủ cụ thể.

Các lệnh có trong TCL là:

# Cam kết: Dùng để lưu trữ dữ liệu trước đó.

# Rollback: Được sử dụng để khôi phục dữ liệu hoặc bảng đã xóa trước đó.

# Savepoint: Dùng để tạo một điểm trong cơ sở dữ liệu hay còn gọi là checkpoint. Nó được sử dụng để khôi phục giao dịch.

**TỪ KHÓA:**

Bất cứ khi nào chúng tôi tạo dữ liệu trong cơ sở dữ liệu, sẽ có khả năng xảy ra các mục trùng lặp trong bảng. Chúng tôi có thể xác định duy nhất các giá trị / bản ghi này từ cơ sở dữ liệu bằng cách sử dụng các cột cụ thể hoặc kết hợp các cột có bản chất duy nhất còn được gọi là 'khóa'.

**Chủ yếu có 5 loại khóa trong cơ sở dữ liệu:**

1. **Khóa chính:** Một thuộc tính có thể được sử dụng để xác định từng bộ giá trị duy nhất trong bảng kết quả được gọi là khóa chính. Có thể chỉ có một khóa chính duy nhất trong bảng.
2. **Khóa ứng viên:** Một tập hợp tối thiểu các thuộc tính có thể xác định duy nhất các bộ giá trị trong bản ghi được gọi là khóa ứng viên. Có thể có nhiều hơn một khóa ứng viên trong bảng còn được gọi là khóa tổng hợp.
3. **Siêu khóa:** Một tập hợp các thuộc tính có thể xác định duy nhất một bộ dữ liệu trong bản ghi được gọi là siêu khóa. Vì vậy, một khóa ứng viên là một siêu khóa nhưng ngược lại không phải lúc nào cũng đúng.
4. **Khóa thay thế: Khóa** ứng viên không phải là khóa chính được gọi là khóa thay thế.
5. **Khóa ngoại:** Một thuộc tính chỉ có thể nhận các giá trị hiện tại dưới dạng giá trị hiện tại làm giá trị của một số thuộc tính khác, là khóa ngoại của thuộc tính mà nó tham chiếu đến.

**HẠN CHẾ:**

Bất cứ khi nào chúng ta xác định cấu trúc của cơ sở dữ liệu, chúng ta có thể tạo các ràng buộc trên cột để đảm bảo các điều kiện nhất định được đáp ứng trước khi giá trị được chấp nhận trong cột đó.

**Dưới đây được đề cập là một số ràng buộc:**

1. **NOT NULL:** Được sử dụng để đảm bảo rằng giá trị null không thể được lưu trữ trong một cột
2. **DUY NHẤT:** Được sử dụng để đảm bảo rằng tất cả các giá trị được lưu trữ trong một cột là duy nhất.
3. **KIỂM TRA:** Được sử dụng để đảm bảo rằng tất cả các giá trị thỏa mãn một điều kiện cụ thể trước khi chúng có thể được chấp nhận trong một cột
4. **DEFAULT:** Được sử dụng để xác định một tập hợp các giá trị mặc định khi không có giá trị nào cho một cột được chỉ định.
5. **INDEX:** Được sử dụng để lưu trữ và truy xuất dữ liệu đến và đi từ cơ sở dữ liệu rất nhanh chóng.

**Sắp xếp kết quả:**

Có nhiều lý do tại sao chúng ta cần sắp xếp kết quả như sau:

Dữ liệu, mà bạn muốn sắp xếp, sẽ không phải lúc nào cũng có trong cơ sở dữ liệu (SQL trong trường hợp này). Nếu bạn lấy một số dữ liệu từ người dùng làm đầu vào nhưng muốn thực hiện sắp xếp trước khi thực hiện bất kỳ thao tác nào thì bạn phải sắp xếp tại chỗ trong mã của chúng tôi.

Đôi khi logic sắp xếp của bạn có thể rất khó (kết hợp nhiều trường và hoạt động). Dữ liệu của chúng tôi có thể trải dài trên nhiều bảng. Trong các vấn đề của thế giới thực, loại dữ liệu mà chúng ta sẽ nhận được sẽ không đơn giản. Nó sẽ không được sắp xếp theo thứ tự tăng dần hoặc giảm dần. Nó sẽ rất nhiều không được sắp xếp. Đôi khi chúng tôi có thể phải thực hiện một số hành động trước khi sắp xếp. Logic sắp xếp của chúng tôi có thể cần nhiều trường và nhiều thao tác trước khi sắp xếp. Hãy lấy một ví dụ. Chúng ta có 2 cột trong bảng A (col1, col2) và 1 cột trong bảng B (col3). Vì một số lý do chúng ta cần sắp xếp thứ tự dựa trên (col1 \* Col2 + col3). (Như bài toán về phương trình đường thẳng mx + c). Bây giờ chúng ta không thể thực hiện bằng cách sử dụng các lệnh của SQL một cách dễ dàng. Đó là nơi chúng tôi sẽ cần phân loại trong mã.

Trong một số trường hợp, việc sắp xếp sẽ nhanh hơn nếu được thực hiện bằng mã dựa trên tập dữ liệu của chúng tôi. Chúng tôi có thể cần các thuật toán sắp xếp khác nhau cho hiệu suất dựa trên tập dữ liệu của chúng tôi. Mọi thứ phụ thuộc vào dữ liệu của chúng tôi.

Chúng tôi cần nghiên cứu các thuật toán sắp xếp để cải thiện việc thiết kế thuật toán của bạn và tìm hiểu các cách tiếp cận và mẫu khác nhau mà bạn sẽ thực sự cần trong thế giới CNTT. Chúng tôi sẽ không bao giờ viết sắp xếp bong bóng hoặc sắp xếp đống trong bất kỳ công ty nào như nó vốn có. Bạn chỉ cần biết logic. Tất cả các ngôn ngữ lập trình đều có các hàm sắp xếp đã được triển khai mà bạn chỉ cần gọi cho mục đích sắp xếp. Chúng sẽ nhanh hơn và được kiểm tra tốt.

Vì vậy, chúng ta cần biết tất cả các cách tiếp cận khác nhau và các giải pháp khả thi để giải quyết một vấn đề. Một giải pháp không thể tốt nhất trong mọi trường hợp. Tùy thuộc vào dữ liệu và bối cảnh, bạn sẽ cần các cách tiếp cận khác nhau. Vì vậy, chúng ta cần khám phá tất cả các giải pháp khả thi và sử dụng chúng dựa trên nhu cầu của mình.

* **Lựa chọn:**

Bất cứ khi nào chúng tôi tìm nạp kết quả từ cơ sở dữ liệu bằng cách sử dụng truy vấn SQL Select, chúng tôi sẽ nhận được kết quả theo kiểu không có thứ tự.

**Ví dụ.**Chọn \* từ nhân viên;

Câu lệnh này tìm nạp tất cả các kết quả từ bảng nhân viên nhưng theo kiểu đặt hàng trước. Nếu chúng ta muốn kết quả được hiển thị theo thứ tự, trong trường hợp đó, chúng ta phải thực hiện các bước tiếp theo.

* **Đặt bởi:**

Chúng ta có thể sắp xếp kết quả trong SQL Select bằng mệnh đề ORDER BY và xác định tên cột trên cơ sở chúng ta muốn sắp xếp kết quả.

**Ví dụ:** Chọn \* từ các nhân viên đặt hàng theo emp\_id;

Câu lệnh này tìm nạp kết quả từ bảng nhân viên và sắp xếp kết quả trên cơ sở cột emp\_id theo thứ tự tăng dần.

Theo mặc định, kết quả thu được theo thứ tự tăng dần.

* **Mô tả:**

Chúng tôi cũng có thể sắp xếp các kết quả theo thứ tự giảm dần bằng cách sử dụng từ khóa “desc”. Sử dụng từ khóa desc, tất cả các kết quả được hiển thị theo thứ tự giảm dần bắt đầu từ giá trị cao nhất được hiển thị ở trên cùng và giá trị thấp nhất được hiển thị ở dưới cùng.

**Ví dụ:** Chọn \* từ các nhân viên đặt hàng theo emp\_id desc;

* **Sắp xếp theo nhiều cột:**

Nếu có hai hoặc nhiều kết quả giống nhau thu được trong một nhóm, trong trường hợp đó, chúng ta có thể sử dụng lập chỉ mục ưa thích trong đó kết quả thu được bằng cách sắp xếp kết quả trên cơ sở cột đầu tiên và nếu kết quả trùng nhau, thì chúng ta có thể sắp xếp kết quả trên cơ sở của cột thứ hai.

**Ví dụ:** Chọn \* từ thứ tự của nhân viên theo emp\_id, lương;

Khi chúng tôi cố gắng sắp xếp kết quả dựa trên 2 cột, trong trường hợp đó, kết quả được sắp xếp ban đầu trên cơ sở emp\_id cột đầu tiên và sau đó dựa trên mức lương của cột thứ hai.

* **Giới hạn:**

Chúng tôi cũng có thể hiển thị kết quả bằng cách giới hạn số hàng được hiển thị bằng cách sử dụng hàm LIMIT.

**Ví dụ:** chọn \* từ các nhân viên đặt hàng theo giới hạn 3;

Câu lệnh này hiển thị 3 hàng trên cùng từ bảng nhân viên sắp xếp kết quả trên cơ sở bảng emp\_id theo thứ tự tăng dần.

* **Bù lại:**

Bất cứ khi nào chúng tôi sử dụng hàm LIMIT, chúng tôi sẽ hiển thị các cột 'n' trên cùng trong kết quả của chúng tôi. Chúng ta có thể hiển thị số hàng chính xác bỏ qua một tập hợp hàng cụ thể từ trên cùng bằng cách sử dụng hàm offset.

**Vd** . Chọn \* từ nhân viên đặt hàng theo giới hạn emp\_id 3 bù 2;

Câu lệnh này sẽ hiển thị kết quả của tất cả nhân viên từ bảng nhân viên được sắp xếp theo emp\_id theo thứ tự tăng dần hiển thị 3 hàng trên cùng sau khi bỏ qua 2 hàng đầu tiên.

Chúng tôi sắp xếp kết quả từ cơ sở dữ liệu của mình bất cứ khi nào chúng tôi muốn hiển thị kết quả từ định dạng không có cấu trúc sang định dạng có cấu trúc.

Bất cứ khi nào chúng tôi hiển thị kết quả trong một trang web hoặc một trang web bằng cách trích xuất kết quả từ Cơ sở dữ liệu SQL, thì chúng tôi phải hiển thị kết quả theo cách có thứ tự, ví dụ. hiển thị kết quả của học sinh.

Ở đó, chúng tôi sử dụng chức năng sắp xếp để hiển thị kết quả một cách có thứ tự.

Vì không phải tất cả dữ liệu đều được lưu trữ trong Cơ sở dữ liệu SQL, một số được lưu trữ trong hệ thống tệp và một số cũng được lưu trữ trong thư mục, vì vậy chúng ta cần tìm hiểu các thuật toán sắp xếp khác nhau để hiển thị kết quả theo thứ tự.

**THAM GIA**

Bất cứ khi nào chúng ta muốn hiển thị kết quả từ nhiều bảng với các giá trị chung, chúng ta có thể nối các bảng trong SQL.

**SQL hỗ trợ các kiểu nối sau trong bảng:**

**1. Tham gia bên trong:**

Tham gia bên trong hoặc liên kết ngầm là cách đơn giản nhất để nối hai bảng. Bất cứ khi nào chúng ta muốn nối hai bảng, chúng ta chỉ cần chọn các cột chung giữa hai bảng mà các bảng có thể được nối với nhau.

Phép nối bên trong trả về các bản ghi có giá trị phù hợp trong cả hai bảng.

**Truy vấn sau đây hiển thị liên kết bên trong:**

|  |  |
| --- | --- |
| 1  2  3 | SELECT Orders.OrderID, Customers.CustomerName  FROM Orders  INNER JOIN Customers ON Orders.CustomerID = Customers.CustomerID; |

**2. Tham gia bên trái:**

Kết nối trái hoặc Nối ngoài trái là một kiểu nối ngoài trả về tất cả các bản ghi từ bảng bên trái và các bản ghi đã so khớp từ bảng bên phải.

**Truy vấn sau đây hiển thị kết nối bên trái:**

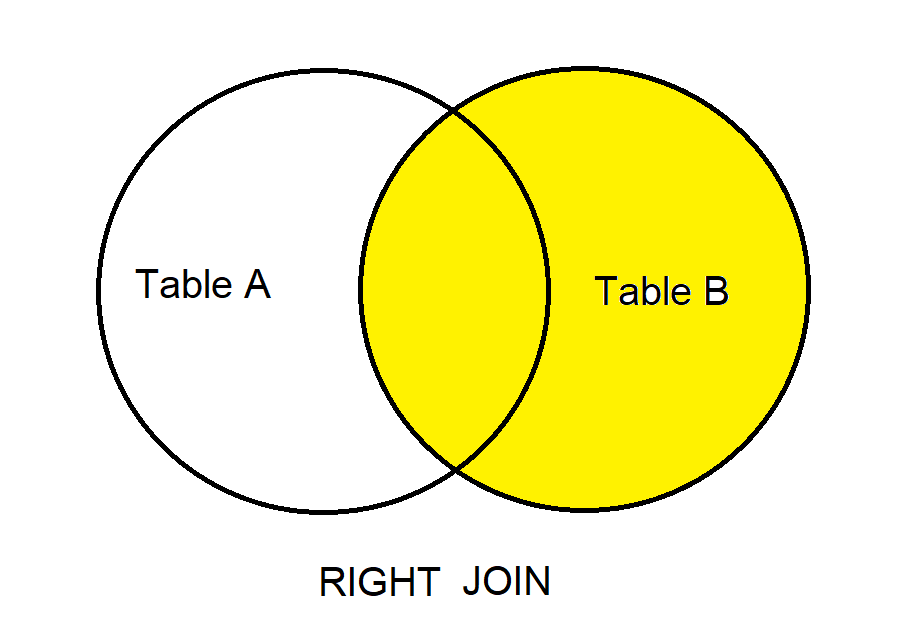
|  |  |
| --- | --- |
| 1  2  3  4 | SELECT Customers.CustomerName, Orders.OrderID  FROM Customers  LEFT JOIN Orders ON Customers.CustomerID = Orders.CustomerID  ORDER BY Customers.CustomerName; |

**3. Tham gia đúng:**

Phép nối bên phải hoặc phép nối ngoài bên phải là một loại phép nối trả về tất cả các bản ghi từ bảng bên phải và các bản ghi đã so khớp từ bảng bên trái

**Truy vấn sau đây hiển thị liên kết phải:**

|  |  |
| --- | --- |
| 1  2  3  4 | SELECT Orders.OrderID, Employees.LastName, Employees.FirstName  FROM Orders  RIGHT JOIN Employees ON Orders.EmployeeID = Employees.EmployeeID  ORDER BY Orders.OrderID; |

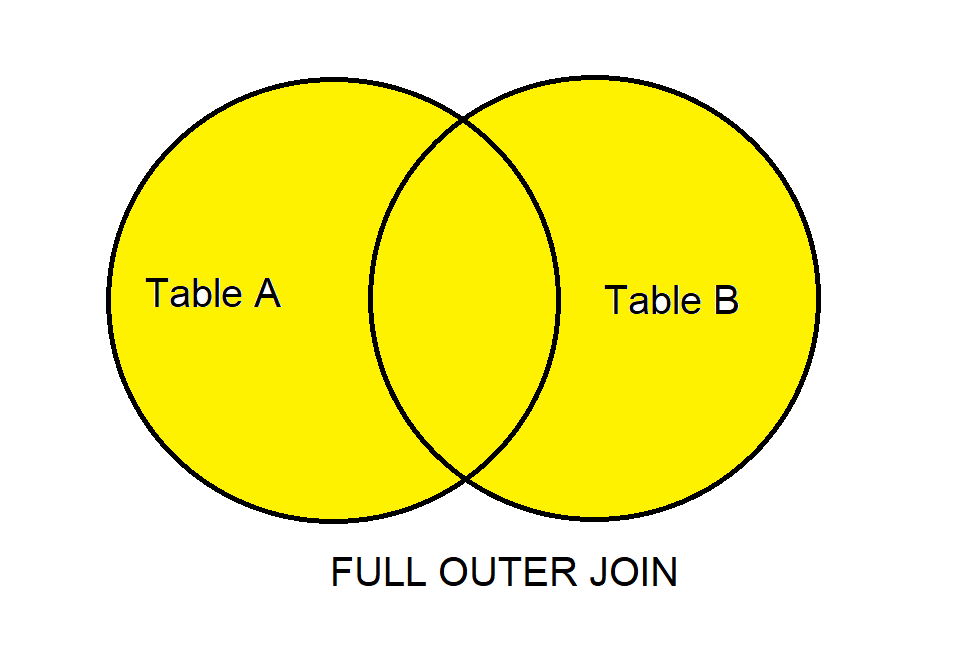


**4. Tham gia đầy đủ:**

Tham gia đầy đủ hoặc tham gia ngoài đầy đủ là một loại liên kết trả về tất cả các bản ghi khi có một kết quả phù hợp trong bảng bên trái hoặc bên phải

**Truy vấn sau đây hiển thị tham gia đầy đủ:**

|  |  |
| --- | --- |
| 1  2  3  4 | SELECT Customers.CustomerName, Orders.OrderID  FROM Customers  FULL OUTER JOIN Orders ON Customers.CustomerID=Orders.CustomerID  ORDER BY Customers.CustomerName; |

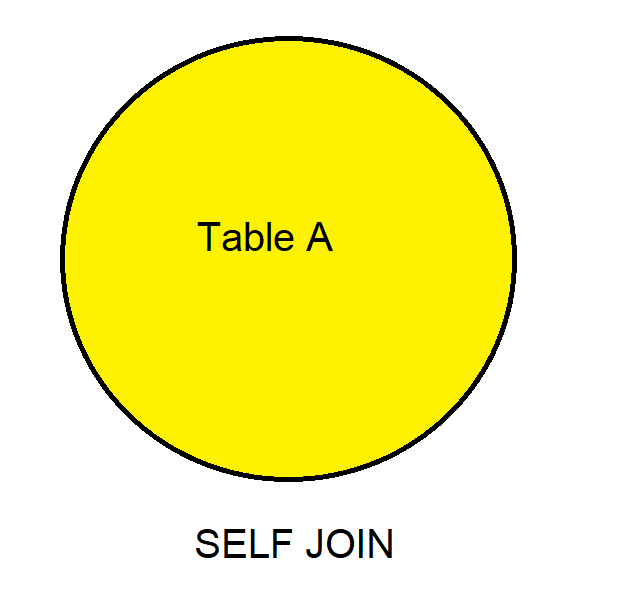


**5. Tự tham gia:**

Tự nối là một kiểu liên kết trong đó một bảng được nối với chính nó trong SQL.

**Truy vấn sau đây hiển thị tự tham gia:**

|  |  |
| --- | --- |
| 1  2  3  4  5 | SELECT A.CustomerName AS CustomerName1, B.CustomerName AS   CustomerName2, A.City  FROM Customers A, Customers B  WHERE A.CustomerID <> B.CustomerID  AND A.City = B.City  ORDER BY A.City; |

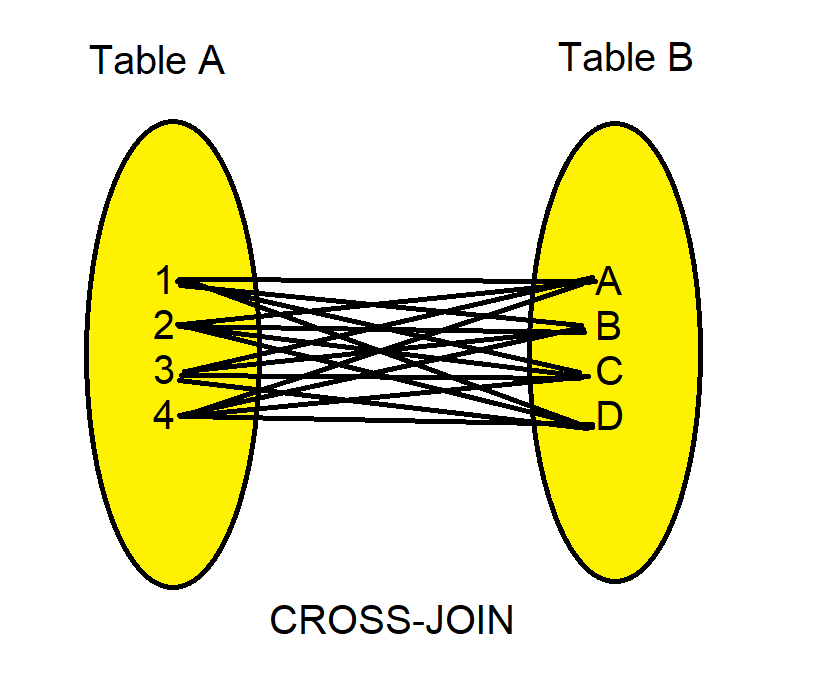


**6. Tham gia chéo:**

Nối chéo là kiểu liên kết trong SQL nơi mọi giá trị từ bảng 1 được nối với mọi giá trị khác trong bảng 2.

**Truy vấn sau đây hiển thị kết hợp chéo:**

|  |  |
| --- | --- |
| 1  2  3  4 | SELECT foods.item\_name,foods.item\_unit,  company.company\_name,company.company\_city  FROM foods  CROSS JOIN company; |



**7. Tham gia trang bị:**

Trong SQL, Equi-join thực hiện phép nối với giá trị cột bằng nhau hoặc khớp với (các) cột của các bảng được liên kết. Trong phép nối tương đương, dấu bằng (=) được sử dụng như một toán tử so sánh trong mệnh đề where để chỉ sự bình đẳng của các bản ghi.

Chúng tôi cũng có thể thực hiện phép nối tương đương bằng cách sử dụng từ khóa JOIN và sau đó là từ khóa ON, sau đó chỉ định tên của các cột cùng với các bảng được liên kết của chúng để kiểm tra tính bình đẳng.

**Truy vấn sau đây hiển thị liên kết tương đương:**

|  |  |
| --- | --- |
| 1  2 | SELECT column\_list FROM table1, table2....WHERE table1.column\_name =  table2.column\_name; |

**Tham gia trang bị vs Tự tham gia:**

Equijoin là kiểu nối với điều kiện nối có chứa toán tử đẳng (=). Một phép nối tương đương chỉ trả về những hàng có giá trị tương đương cho tập hợp cột được chỉ định.

Trong khi đó, phép nối bên trong là phép nối của hai hoặc nhiều bảng chỉ trả về các hàng đó (được so sánh bằng cách sử dụng toán tử so sánh) thỏa mãn điều kiện nối.

**8. Tham gia tự nhiên:**

Trong SQL, Natural Join là một loại Equijoin và được cấu trúc theo cách mà các cột có cùng tên với các bảng được liên kết sẽ chỉ xuất hiện một lần.

**Tham gia tự nhiên và Tham gia bên trong:**

Sự khác biệt giữa phép nối bên trong và phép nối tự nhiên là số cột được trả về.

**Giá trị NULL**

Trong SQL, một cột có thể chứa giá trị NULL. Điều này có nghĩa là không tồn tại giá trị cho bản ghi cụ thể đó trong bảng. Chúng ta có thể kiểm tra sự tồn tại của giá trị NULL trong SQL bằng các lệnh sau:

1. IS NULL - Truy vấn này trả về true nếu giá trị cột là NULL.
2. IS NOT NULL - Truy vấn này trả về true nếu giá trị cột không phải là NULL. Ngược lại, nếu giá trị cột là NULL, giá trị false được trả về.
3. <=> - Toán tử này so sánh các giá trị và trả về true cho 2 giá trị KHÔNG ĐỦ.

**Truy vấn sau đây hiển thị các giá trị NULL trong SQL:**

|  |  |
| --- | --- |
| 1 | SELECT \* FROM table1 WHERE tutorial\_count = NULL; |

Tại đây, truy vấn trả về tất cả các bản ghi đó trong đó giá trị của cột tutorial\_count = NULL

**REGEXPS trong SQL :**

Trong SQL, chúng ta có thể sử dụng Regexps để so khớp chuỗi với một chuỗi cụ thể hoặc một tập hợp con của chuỗi. Nó được sử dụng để đối sánh với mẫu để tìm một chuỗi cụ thể với mẫu đó.

**Có 3 cách để sử dụng so sánh regexp trong SQL:**

1. Như
2. Tương tự với
3. Bộ so sánh Posix

Giống như và tương tự với được sử dụng cho các so sánh cơ bản trong đó một chuỗi cụ thể được sử dụng để khớp với chuỗi phù hợp. Nếu mẫu được khớp, kết quả là chuỗi tương ứng được trả về.

**Truy vấn sau đây cho thấy việc sử dụng toán tử LIKE:**

|  |  |
| --- | --- |
| 1 | Select \* from employees where emp\_name LIKE ‘a%a’; |

Với toán tử Posix, chúng tôi sử dụng các toán tử sau để khớp với kết quả:

1. ~: Khớp, phân biệt chữ hoa chữ thường
2. ~ \*: Khớp, không phân biệt chữ hoa chữ thường
3. ! ~: Không khớp, phân biệt chữ hoa chữ thường
4. ! ~ \*: Không trùng khớp, không phân biệt chữ hoa chữ thường

Chúng tôi cũng có thể sử dụng toán tử REGEXP để khớp với mẫu. Dưới đây là bảng xác định danh sách các toán tử được sử dụng với các toán tử REGEXP để khớp với chuỗi mong muốn.

1. ^: Bắt đầu của chuỗi
2. $: Cuối chuỗi
3. . : Bất kỳ ký tự đơn nào
4. […]: Bất kỳ ký tự nào được liệt kê giữa 2 dấu ngoặc vuông
5. [^…]: Bất kỳ ký tự nào không được liệt kê giữa 2 dấu ngoặc vuông
6. p1 | p2 | p3: Thay thế, khớp với bất kỳ mẫu nào p1 hoặc p2 hoặc p3
7. \*: 0 hoặc nhiều trường hợp của phần tử trước
8. +: 1 hoặc nhiều trường hợp của phần tử trước
9. {x}: x phiên bản của phần tử trước
10. {x, y}: x đến y các trường hợp của phần tử trước

Chúng ta có thể sử dụng truy vấn SQL sau để hiển thị việc sử dụng toán tử REGEXP:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | SELECT name FROM person\_tbl WHERE name REGEXP '^st';    SELECT name FROM person\_tbl WHERE name REGEXP 'ok$';    SELECT name FROM person\_tbl WHERE name REGEXP 'mar';    SELECT FirstName FROM intque.person\_tbl WHERE FirstName REGEXP '^[aeiou].\*ok$'; |

**LỆNH ALTER :**

Lệnh Alter trong SQL dùng để thay đổi cấu trúc cơ sở dữ liệu. Chúng ta có thể thêm hoặc bớt các cột. Chúng ta có thể thay đổi lược đồ của cột bảng trong SQL bằng cách sử dụng lệnh Alterna. Lệnh Alter cũng có thể được sử dụng để đổi tên bảng. Với sự trợ giúp của lệnh thay đổi, chúng ta cũng có thể định vị lại các cột.

1. **Thêm cột:**

Với sự trợ giúp của truy vấn sau, chúng ta có thể thêm một cột trong bảng SQL:

Nhân viên thay thế bảng thêm tên varchar (30);

Lệnh này thêm một cột tên kiểu dữ liệu varchar có độ dài 30 ký tự vào bảng nhân viên.

1. **Thả cột:**

Với sự trợ giúp của truy vấn sau, chúng ta có thể bỏ một cột từ các bảng SQL:

Thay đổi tên nhân viên bảng;

Lệnh này xóa cột tên khỏi bảng nhân viên

1. **Định vị lại một cột:**

Với sự trợ giúp của các truy vấn sau, chúng ta có thể định vị lại một cột cụ thể trong bảng SQL:

Nhân viên thay đổi bảng thêm lương int đầu tiên;

Lệnh này thêm cột lương kiểu int vào bảng nhân viên ở vị trí đầu tiên.

Nhân viên thay đổi bảng thêm lương int sau tên;

Lệnh này thêm cột lương kiểu int vào bảng nhân viên sau cột tên

1. **Đặt giá trị mặc định:**  
   Lệnh sau giúp đặt giá trị mặc định cho một cột cụ thể nếu giá trị rõ ràng không được cung cấp.

Nhân viên thay đổi bảng thay đổi lương đặt mặc định 1000;

Lệnh này đặt giá trị mặc định của cột lương thành 1000 nếu giá trị rõ ràng của cột lương không được cung cấp.

1. **Thay đổi định nghĩa cột:**

Nhân viên thay đổi bảng sửa đổi tên varchar (20);

Lệnh này thay đổi cột tên trong bảng nhân viên và sửa đổi kiểu dữ liệu của nó thành

varchar (20) bằng cách ghi đè kiểu dữ liệu trước đó của nó.

1. **Đổi tên một cột:**

* Lệnh sau thay đổi tên của bảng trong cơ sở dữ liệu SQL.
* Thay đổi tên nhân viên bảng thành trống;
* Lệnh này thay đổi tên của bảng nhân viên và đổi tên nó thành emp.

Lệnh Alter chỉ cần cập nhật cấu trúc của bảng cơ sở dữ liệu. Nó có thể thêm, bớt, đặt lại vị trí, đổi tên các cột hoặc đơn giản là đổi tên toàn bộ bảng của cơ sở dữ liệu.

**CÁC CHỈ SỐ TRONG SQL**

Chỉ mục SQL là cấu trúc dữ liệu được sử dụng để cải thiện tốc độ hoạt động trong bảng. Chỉ mục có thể được tạo trên một hoặc nhiều cột, do đó cung cấp cơ sở cho việc tra cứu bảng nhanh chóng và hiệu quả.

Chỉ mục cũng là một loại bảng lưu giữ bản ghi các con trỏ trỏ đến mỗi bản ghi trong bảng thực tế.

Chỉ mục có thể có hai loại:

1. Chỉ mục đơn giản
2. Chỉ mục duy nhất

Lập chỉ mục đơn giản cho phép 2 hàng có cùng giá trị chỉ mục trong khi Unique không cho phép 2 hàng của bảng không có cùng giá trị chỉ mục.

Lệnh sau giúp tạo chỉ mục trong bảng cơ sở dữ liệu.

**Chỉ mục đơn giản:**

|  |  |
| --- | --- |
| 1 | Create index my\_index on employee( emp\_name, salary); |

**Chỉ số duy nhất:**

|  |  |
| --- | --- |
| 1 | Create unique index my\_index on employee(emp\_name, salary); |

Các truy vấn trên tạo một chỉ mục my\_index trên bảng nhân viên với các cột tên\_và và lương;

Chúng ta có thể sử dụng lệnh thay đổi để thêm hoặc xóa các chỉ mục trên một bảng hiện có.

* Nhân viên thay thế bảng thêm khóa chính (tên\_tên, tiền lương)

Lệnh này thêm chỉ mục khóa chính trên bảng nhân viên

* Nhân viên thay thế bảng thêm my\_index duy nhất (tên\_tên, tiền lương)

Lệnh này thêm chỉ mục duy nhất my\_index trên bảng nhân viên trong đó 2 hàng không được có cùng giá trị chỉ mục (lập chỉ mục duy nhất)

* Nhân viên thay thế bảng thêm chỉ mục my\_ind (tên\_tên, tiền lương)

Lệnh này thêm một chỉ mục bình thường my\_ind trên bảng nhân viên trong đó 2 hàng có thể có cùng giá trị chỉ mục (lập chỉ mục đơn giản)

* Nhân viên thay đổi bảng thả chỉ mục my\_index

Lệnh này loại bỏ chỉ mục my\_index khỏi bảng nhân viên.

Hiển thị các chỉ mục:

Lệnh sau có thể được sử dụng để hiển thị tất cả các chỉ mục trong bảng SQL.

Hiển thị chỉ mục từ nhân viên \ G

Ở đây \ G được sử dụng để hiển thị các kết quả theo thứ tự dọc, do đó tránh các kết quả dài theo chiều ngang.

**GIAO DỊCH TRONG CƠ SỞ DỮ LIỆU**

Trong cơ sở dữ liệu, một giao dịch là một nhóm tuần tự các câu lệnh, được thực hiện như thể nó là một đơn vị duy nhất.

Cơ sở dữ liệu phải tuân theo các thuộc tính giao dịch sau:

**AXIT**

1. **(A) tomicity:** Nó đảm bảo rằng các giao dịch được hoàn thành đầy đủ hoặc hoàn toàn bị hủy bỏ. Không nên có giao dịch từng phần diễn ra trong SQL. Ví dụ. Nếu chuyển tiền được thực hiện từ tài khoản A sang tài khoản B, thì quá trình chuyển tiền hoàn tất sẽ được thực hiện hoặc không được thực hiện. Không được có giao dịch từng phần diễn ra ghi nợ tiền từ tài khoản A và không ghi có tiền vào tài khoản B.
2. **(C) tính nhất quán:** Cơ sở dữ liệu phải luôn duy trì ở trạng thái nhất quán. Cơ sở dữ liệu phải nhất quán sau truy vấn SQL vì nó nhất quán trước truy vấn SQL. Ví dụ. Khi chuyển tiền diễn ra từ tài khoản A sang tài khoản B, thì tổng số tiền trong tài khoản A và tài khoản B phải luôn bằng nhau trước và sau khi giao dịch.
3. **(I) solation:** Nó đảm bảo rằng nếu nhiều giao dịch đang chạy thành công trong cơ sở dữ liệu, thì mỗi giao dịch sẽ không bị ảnh hưởng bởi các giao dịch khác đang chạy trong cơ sở dữ liệu. Ví dụ. nếu có 2 thao tác ghi được thực hiện trong một cơ sở dữ liệu, thì sẽ không có sự can thiệp nào giữa hai thao tác đang được thực hiện. Chúng nên được thực hiện một cách cô lập.
4. **(D) tính khả thi:** Cơ sở dữ liệu phải chống được lỗi. Các thay đổi được thực hiện trong hệ thống cơ sở dữ liệu phải được lưu trữ vĩnh viễn ngay cả khi hệ thống bị lỗi phần mềm. Ví dụ. Nếu tài khoản A đã được ghi có với số tiền Rs. 50000, thì thay đổi đó sẽ tồn tại vĩnh viễn trong tài khoản ngay cả khi hệ thống hoặc phần mềm bị lỗi vì nó chứa thông tin nhạy cảm liên quan đến tài khoản của khách hàng.

**CAM KẾT, ROLLBACK VÀ SAVEPOINT**

Bất cứ khi nào chúng tôi sử dụng các giao dịch trong cơ sở dữ liệu, chúng tôi tạo các điểm kiểm tra còn được gọi là điểm lưu để đánh dấu điểm trong cơ sở dữ liệu nơi các giao dịch có thể được lưu trữ.

**Cam kết:** Lệnh này cho phép chúng tôi lưu các giao dịch trong cơ sở dữ liệu uptol một điểm cụ thể trong cơ sở dữ liệu. Tất cả các truy vấn hoặc giao dịch cho đến một điểm cụ thể đều được lưu trong cơ sở dữ liệu.

**Điểm lưu :** Một điểm kiểm tra được tạo ra để đánh dấu một điểm tối đa nơi các giao dịch có thể được lưu hoặc khôi phục trong trường hợp có lỗi.

**Roll-back:** Lệnh này đảm bảo với chúng tôi rằng một tập hợp các câu lệnh cụ thể có thể được hoàn tác trong trường hợp xảy ra lỗi đối với một điểm lưu cụ thể được tạo. Chúng tôi không cần đảo ngược toàn bộ giao dịch đã thực hiện. Chỉ một tập hợp giao dịch cụ thể mới được hoàn nguyên.

MySQL cung cấp một tính năng của AUTOCOMMIT. Chúng ta có thể đặt AUTOCOMMIT = 1, để đảm bảo rằng tự động gửi được BẬT. Khi tự động gửi được BẬT, thì mỗi và mọi giao dịch sẽ tự động được cam kết trong bộ nhớ ngay sau khi nó được thực thi.

**BẢNG TẠM THỜI TRONG SQL**

Bảng tạm thời được sử dụng để lưu trữ dữ liệu tạm thời trong SQL. Các bảng này vẫn còn trong bộ nhớ cho đến khi một phiên cụ thể còn tồn tại hoặc cho đến khi chúng bị hủy theo cách thủ công.

Bảng tạm thời chỉ có sẵn trong phiên bản SQL mới nhất. Nếu chúng ta đang sử dụng các phiên bản SQL cũ hơn thì không thể sử dụng các bảng tạm thời. Trong trường hợp đó, bảng đống có thể được sử dụng.

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | CREATE TEMPORARY TABLE SalesSummary (     product\_name VARCHAR(50) NOT NULL     , total\_sales DECIMAL(12,2) NOT NULL DEFAULT 0.00     , avg\_unit\_price DECIMAL(7,2) NOT NULL DEFAULT 0.00     , total\_units\_sold INT UNSIGNED NOT NULL DEFAULT 0  ); |

Lệnh trên tạo bảng tạm thời SalesSummary với các tên cột đã xác định ở trên và các kiểu dữ liệu tương ứng.

Các giá trị trong bảng tạm thời có thể được chèn bằng lệnh chèn đơn giản.

Nếu chúng ta sử dụng lệnh 'Show Tables', thì các bảng tạm thời sẽ không được hiển thị.

Loại bỏ các bảng tạm thời khỏi bộ nhớ:

Mặc dù các bảng tạm thời sẽ tự động bị xóa khi phiên kết thúc, nhưng nếu chúng ta muốn xóa chúng, chúng ta có thể sử dụng lệnh drop table.

Thả bảng SalesSummary;

Lệnh trên xóa bảng tạm thời SalesSummary khỏi cơ sở dữ liệu.

* **SQL so với MySQL: Sự khác biệt là gì?**

Mô hình quan hệ lần đầu tiên được mô tả trong một bài báo năm 1970 bởi Edgar F. Codd. Một trong những ngôn ngữ lập trình thương mại đầu tiên liên quan đến mô hình, SQL, được phát triển ngay sau đó tại IBM. Trong một thời gian, SQL là ngôn ngữ cơ sở dữ liệu được sử dụng rộng rãi nhất, được chấp nhận như một tiêu chuẩn ANSI vào năm 1986 và trong ISO một năm sau đó.

SQL bao gồm bốn ngôn ngữ con, mỗi ngôn ngữ có một phạm vi khác nhau.

* **DQL:** Ngôn ngữ truy vấn dữ liệu (DQL) là ngôn ngữ quen thuộc nhất và được sử dụng để chạy các truy vấn trên cơ sở dữ liệu và trích xuất thông tin từ dữ liệu được lưu trữ. Ví dụ: chọn và trả về giá trị lớn nhất trong một cột.
* **DDL:** Một ngôn ngữ định nghĩa dữ liệu (DDL) được sử dụng để mã hóa các cấu trúc và lược đồ cụ thể của cơ sở dữ liệu. Tạo bảng hoặc xác định kiểu dữ liệu là một ví dụ.
* **DCL:** Một ngôn ngữ kiểm soát dữ liệu (DCL) xác định quyền truy cập, ủy quyền và quyền cho người dùng và các quá trình truy cập cơ sở dữ liệu, bao gồm cả việc cấp đặc quyền quản trị viên hoặc hạn chế người dùng chỉ có các đặc quyền chỉ đọc.
* **DML:** Và cuối cùng, ngôn ngữ thao tác dữ liệu (DML) được sử dụng để thực hiện sửa đổi trên các thành phần hiện có của cơ sở dữ liệu, như chèn bản ghi, cập nhật giá trị trong ô hoặc xóa dữ liệu.

Công ty Thụy Điển MySQL AB lần đầu tiên phát hành MySQL vào năm 1995. Giống như hầu hết các phần mềm cơ sở dữ liệu theo sau sự nổi lên ban đầu của các hệ thống quan hệ, [MySQL chỉ đơn giản là một phần mở rộng của tiêu chuẩn SQL ban đầu](https://www.talend.com/resources/sql-vs-nosql/) , bổ sung thêm nhiều tính năng, hỗ trợ, lập trình thủ tục, cơ chế luồng điều khiển, và nhiều hơn nữa.

* **Đám mây và tương lai của MySQL**

MySQL ban đầu được hình dung để quản lý cơ sở dữ liệu khổng lồ, nhanh hơn các phần mềm cơ sở dữ liệu hiện có. Được sử dụng trong các môi trường hoạt động, giao dịch và sản xuất đòi hỏi khắt khe trong nhiều thập kỷ, MySQL đã phát triển cùng với việc chuyển tính toán và [lưu trữ vào đám mây](https://www.talend.com/resources/cloud-storage/) .

Mặc dù thường được cài đặt trên các máy riêng lẻ, MySQL hiện bao gồm hỗ trợ sâu cho các ứng dụng phân tán và đưa vào hầu hết các nền tảng dữ liệu đám mây.

So với nhiều giải pháp lưu trữ và xử lý dữ liệu trên thị trường hiện nay, MySQL là một công nghệ cũ hơn, nhưng nó không có dấu hiệu gắn cờ về mức độ phổ biến hay tiện ích. Trên thực tế, MySQL đã có một sự hồi sinh gần đây so với các hệ thống lưu trữ hiện đại chuyên biệt hơn, do tốc độ, độ tin cậy, dễ sử dụng và khả năng tương thích rộng của nó.