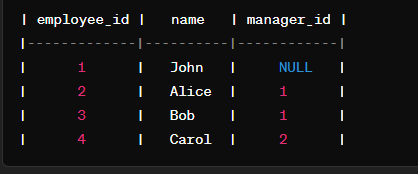
* **NULL** được sử dụng trong cơ sở dữ liệu để đại diện cho giá trị không tồn tại hoặc không biết



***SELECT name FROM students WHERE student\_id = NULL;***

* ***Ko trả về kq***
* **Lệnh IS NULL** được sử dụng để kiểm tra xem một giá trị có là NULL hay không *(vd: Lấy tên của nhân viên ko có quản lý)*

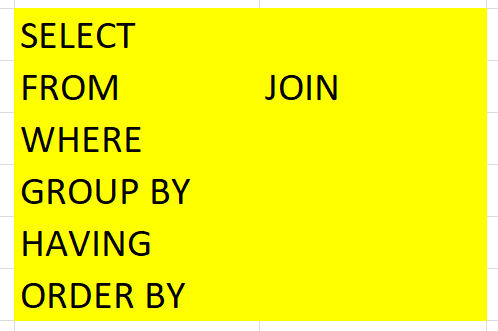
***SELECT name FROM employees WHERE manager\_id IS NULL;***

* ***John***

**BÀI GIẢNG MÔN SQL CƠ BẢN CHO LẬP TRÌNH VIÊN**

**VÀ KIỂM THỬ VIÊN**

Bí kíp số 1: Cú pháp của lệnh SELECT 6 chữ (có bổ sung thêm 1 JOIN)



Bí kíp số 2: Chỉ cần nhớ 1 chữ duy nhất là chữ DÒNG (ROW = dòng, Record = bảng ghi)

Bí kíp số 3: Chỉ cần nhớ 1 cụm từ nữa là BẢNG TẠM.

Rồi, bây giờ áp dụng bí kíp vào các câu lệnh SQL.

Câu hỏi 1: Lệnh SELECT dùng làm gì?

Trả lời: Lấy ra các dòng phù hợp với một số tiêu chí nào đó trong 1 hoặc nhiều bảng.

Câu hỏi 2: WHERE dùng làm gì?

Trả lời: Dùng để kết hợp với SELECT, mục đích là chọn ra được các dòng theo những điều kiện nào đó, các điều kiện đó được đặt trong WHERE. Nói ngắn gọn thì WHERE là nơi để đặt điều kiện LỌC DÒNG.

Câu hỏi 3: Câu lệnh SELECT \* FROM HOCSINH WHERE 1 = 1 có nghĩa là gì?

Câu này sẽ trả về tất cả các dòng của bảng HOCSINH.

Nhưng nó sẽ khác câu sau chỗ nào?

SELECT \* FROM HOCSINH

Có bạn trả lời như sau: Câu trên phải chạy qua các dòng, còn câu dưới không cần chạy luôn mà lấy tất cả các dòng ra.

Đúng là chỉ người bị thần kinh mới viết câu SELECT \* FROM HOCSINH WHERE 1 = 1. Câu này sẽ làm cho chương trình SQL chạy chậm.

Cách thực hiện 1 câu lệnh SELECT sẽ là:

Duyệt qua từng dòng trong bảng, cứ mỗi lần duyệt qua thì điều kiện trong WHERE sẽ được kiểm tra 1 lần.

WHERE 1 = 1 là điều kiện luôn luôn đúng, nên việc kiểm tra chỉ làm mất thời gian, chứ không có ý nghĩa gì cả.

Hãy nhớ là, trong SQL tất cả mọi xử lý đều là xử lý trên DÒNG, chứ không phải trên cột. Rất nhiều người hiểu nhầm là xử lý trên CỘT.

Câu hỏi 4: Giải thích câu lệnh sau



Câu này sẽ được thực hiện từng bước như sau:

Duyệt qua từng dòng trong bảng SINHVIEN, mỗi lần duyệt thì kiểm tra điều kiện quequan <> 'Phuong Hoang Co Tran' hay không. Nếu điều kiện đúng thì lấy dòng đó ra.

À, lấy hết các dòng thỏa điều kiện trong WHERE đã, sau đó mới đi sắp xếp theo GioiTinh giảm dần.

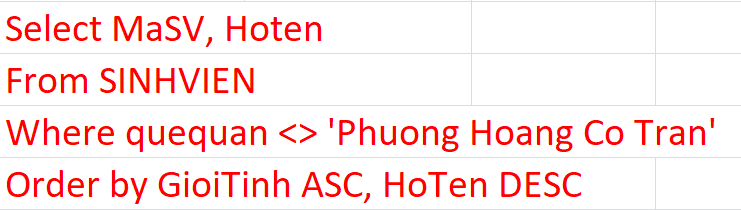
Như vậy, nhiệm vụ của ORDER BY là làm gì?

ORDER BY là sắp xếp các dòng của bảng tạm sinh ra sau khi WHERE.

Bảng tạm là bảng sinh ra tạm thời trong quá trình xử lý câu lệnh SQL.

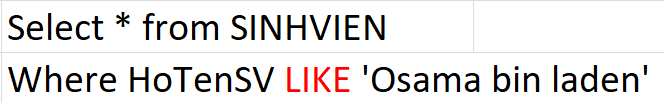
Một câu lệnh SQL có thể tạo ra nhiều bảng tạm trung gian, trong quá trình xử lý.

Câu hỏi 5: Giải thích câu lệnh sau



Cũng tương tự như ở câu 4, nhưng sẽ ưu tiên sắp xếp giới tính trước, khi giới tính bằng nhau thì mới xét đến họ tên.

Câu hỏi 6: Giải thích câu lệnh sau



Đây là 1 trong những câu dễ bị lừa nhất khi đi phỏng vấn.

Họ hay hỏi là: câu SQL trên sẽ liệt kê những người có họ tên là Osama bin laden hay là họ tên có chứa chuỗi là Osama bin laden?

Nói tóm lại là sinh viên Osama bin laden bin ladu có xuất hiện trong kết quả trả về không?

Câu trả lời là KHÔNG CÓ.

Từ khóa LIKE dùng để tìm kiếm chuỗi 1 cách tương đối (tức là tìm không chính xác hoàn toàn), nhưng với 1 điều kiện là nó phải đi kèm với các ký tự đại diện (%, \_, [], [^]).

Nếu không có các ký tự đại diện thì LIKE tương đương với = .

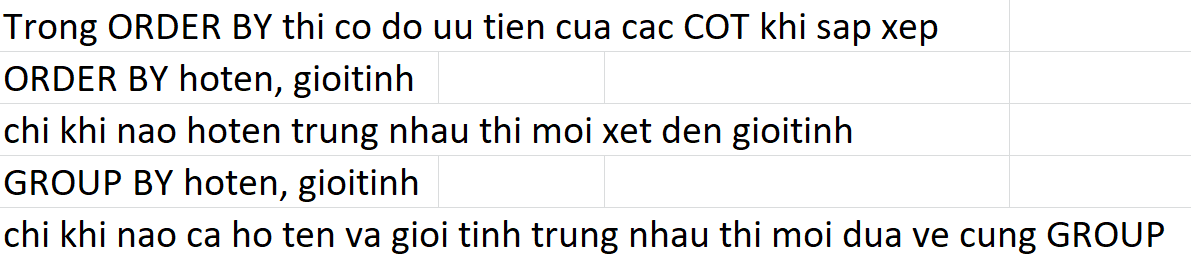
Câu số 7: Trình bày ý nghĩa của 3 cụm từ sau ORDER BY, GROUP BY, DISTINCT?

ORDER BY: Sap xep cac DONG trong BANG TAM dua vao su tang dan hoac giam dan cua cua 1 hoac nhieu cot

GROUP BY: Nhom cac DONG trong BANG TAM dua vao su giong nhau (ve gia tri) cua 1 hoac nhieu cot

DISTINCT: LIET KE KHONG TRUNG LAP CAC DONG TRONG BANG TAM TRA VE (CHU KO PHAI LA TRONG BANG GOC). Có rất nhiều người phát biểu là DISTINCT dùng để loại bỏ các dòng trùng lặp. Câu này đúng hay sai? Không đúng, vì vẫn giữ lại 1 dòng chứ đâu xóa hết các dòng giống nhau.

Câu số 8: Trình bày sự khác nhau giữa ORDER BY và GROUP BY



Câu số 9: Câu SQL nào sau đây bị lỗi?

Select GioiTinh, count(MaHS) From HOCSINH

Select count(MaHS) From HOCSINH

Select GioiTinh, count(MaHS) From HOCSINH Group by GioiTinh

Select count(MaHS) From HOCSINH Group by GioiTinh

Để trả lời được câu hỏi này, mọi người cần nhớ lại các kiến thức sau đây:

Hàm gộp thường đi với GROUP BY

Không được SELECT các cột không có trong mệnh đề GROUP BY (nếu câu SELECT có GROUP BY).

Khi chạy thì ta sẽ thấy câu 1 lỗi, câu 2 không lỗi.

Câu 2 không lỗi vì nếu như không có group by thì nó xem như cả bảng SINHVIEN là 1 nhóm. Nghĩa là, không có GROUP BY thì không có nghĩ là không có nhóm. Cho nên count(MaHS) không bị lỗi.

Còn câu số 1, chúng ta tưởng tượng là tương đương với câu sau:

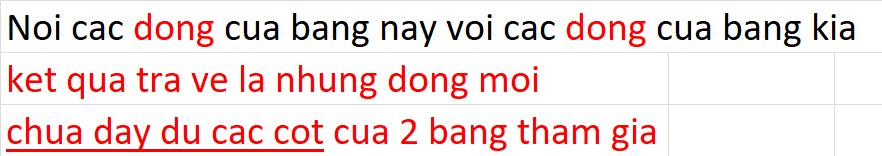
Select GioiTinh, count(MaHS) From HOCSINH GROUP BY (không có cột nào trong GROUP BY)

Nhớ là trong mệnh đề SELECT mà có hàm gộp thì SQL nó sẽ hiểu là chúng ta đang muốn GROUP BY. Nếu chúng ta có hàm gộp trong SELECT mà không có GROUP BY thì SQL vẫn hiểu là chúng ta đang GROUP BY cả bảng về thành 1 GROUP.

Câu thứ 3 và 4 đương nhiên là không có lỗi.

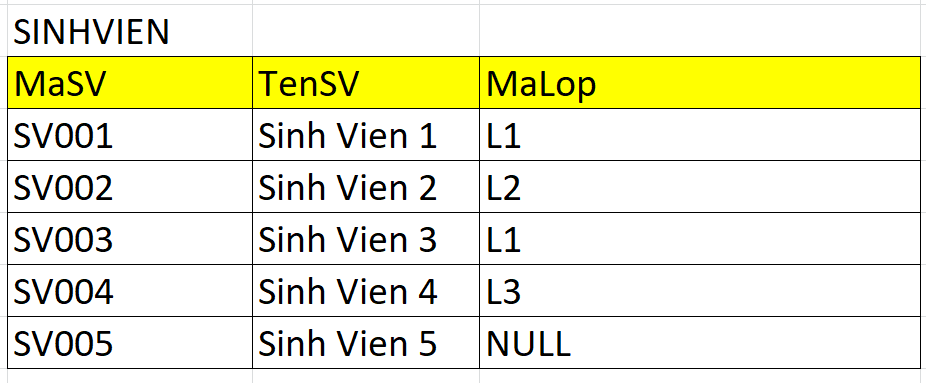
Câu số 10: Bản chất của phép nối (JOIN) là gì?

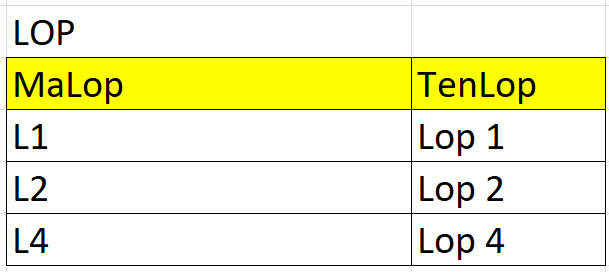
Trả lời:



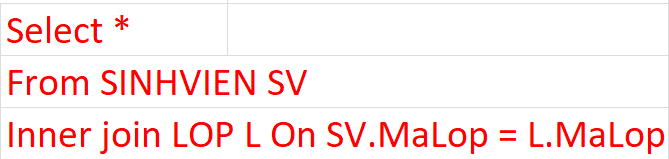
Lưu ý: Cách trả lời toàn dân (bình dân) thì JOIN nối 2 bảng, hoặc nối bảng. Trả lời cho vui thì được, chứ trả lời xong rồi hiểu theo cách đó luôn là tầm bậy.

Giả sử có 2 bảng:





Và có câu SQL là:





Có quá nhiều người hiểu nhầm là khi nối bảng xong, thì chỉ có 1 cột MaLop (tức là nghĩ nhầm kết quả trả về chỉ có 4 cột). Thực tế cột SV.MaLop và L.MaLop là 2 cột hoàn toàn khác nhau. Vì nó trùng tên cho nên phải ghi thêm nick name của bảng ở phía trước.

Câu số 11: Cùng dữ liệu như câu 10, 2 câu SQL sau câu nào bị lỗi không?

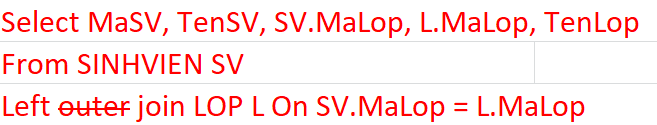
Select SV.MaSV, L.TenLop From SINHVIEN SV Inner join LOP L On SV.MaLop = L.MaLop

Select MaSV, TenLop From SINHVIEN SV Inner join LOP L On SV.MaLop = L.MaLop

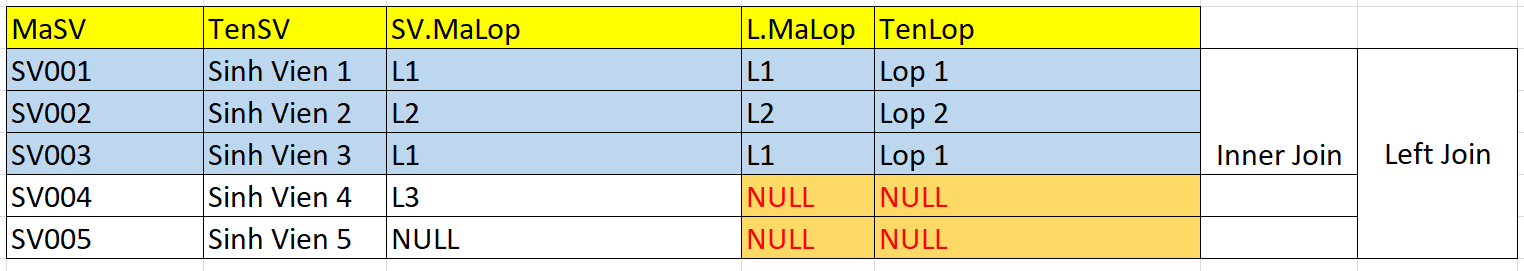
Cả 2 câu ở trên đều không có lỗi.

Cái SV. và L. khi nào thì cần? Khi mà có sự trùng tên của cột trong 2 bảng tham gia vào phép nối thì ta mới cần thêm vào. Ta thích thêm vào cũng chẳng sao, không chết ai hết.

Câu số 12: Phân biệt giữa INNER JOIN và LEFT JOIN?



Chữ outer có cũng được, không có cũng được.

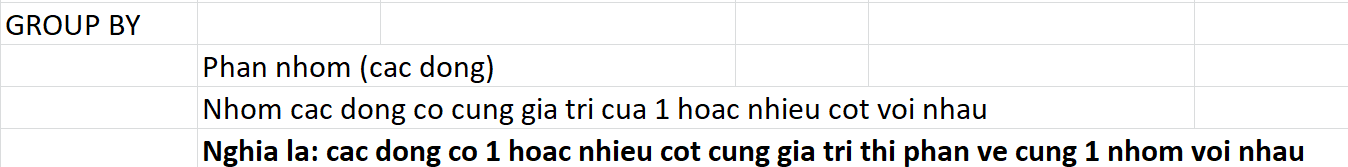


Khi ta dùng phép Left hoặc Right Join thì ta mới thấy rõ là cột MaLop của SV.MaLop khác hẳn với L.MaLop.

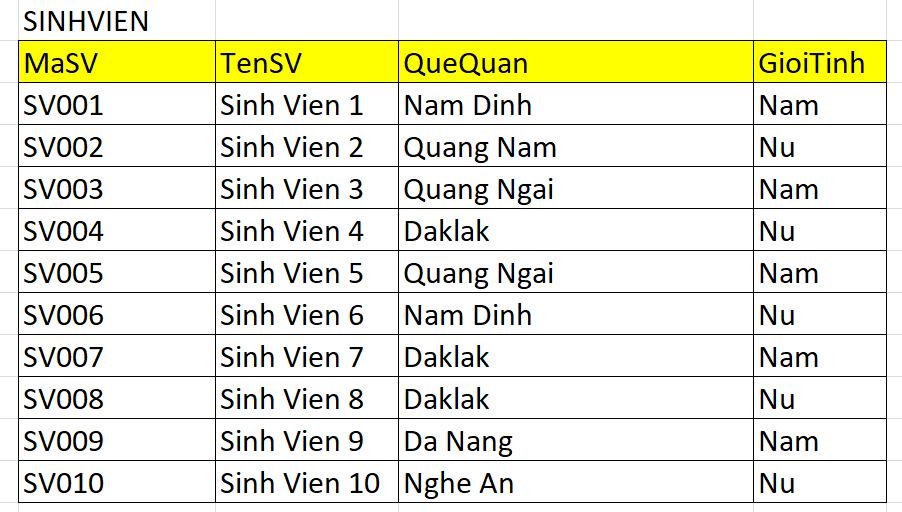
Câu SQL trên nếu mô tả bằng lời thì ta sẽ nói là: Liệt kê TẤT CẢ sinh viên (kể cả sinh viên chưa được phân lớp).

Chữ TẤT CẢ tương đương với chữ MỌI, MỖI MỘT….

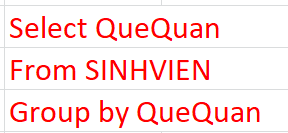
Câu số 13: Trình bày cơ chế hoạt động của GROUP BY

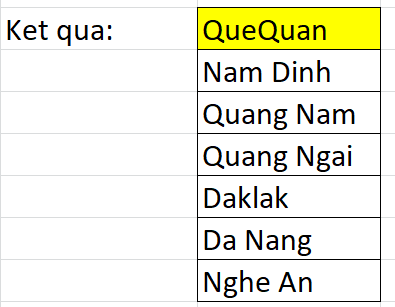


Giả sử ta có bảng dữ liệu như sau:

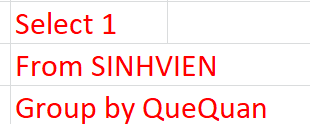


Sau đó, ta có câu SQL:





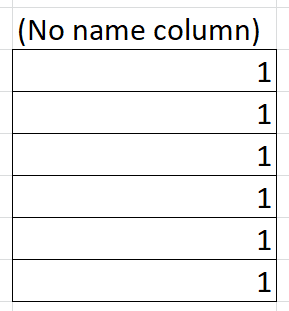
Cũng dữ liệu như trên, nên ta thay đổi lại câu SQL 1 chút:



Đố mọi người chuyện gì xảy ra?

Có lỗi không?

Kết quả sẽ là:



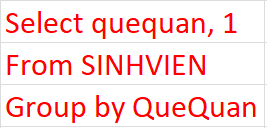
Qua ví dụ này, chúng ta học được những kiến thức như sau:

Một là, chúng ta có quyền select những hằng số, để tạo ra những cột giả (cột không tên). Dữ liệu ở cột giả này sẽ giống nhau ở tất cả các hàng.

Hai là, sau khi GROUP BY, mỗi GROUP sẽ trả về đúng 1 dòng kết quả duy nhất. Cho nên trong ví dụ trên nó có 6 dòng.

Ba là, không cần thiết phải SELECT các cột có trong GROUP BY.

Tức là nếu ta viết như thế này:



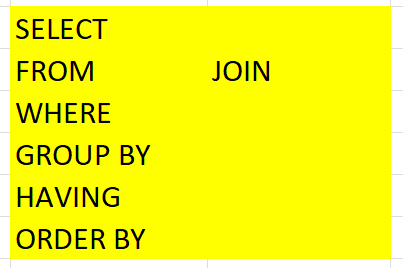
Thì kết quả sẽ là như thế này:



Hằng số có thể là 1 số hoặc 1 chuỗi hoặc 1 chuỗi ngày tháng năm…

Đến đây, mọi người bắt đầu cảm giác được về SELECT rồi.

Quay lại bí kíp số 1, ta sẽ hiểu là, ngoài cú pháp của lệnh SELECT là để ta học thuộc thứ tự thì ta phải hiểu là, nó còn cho ta biết thứ tự thực hiện câu lệnh SELECT.



FROM chạy đầu tiên, sau đó tới JOIN là tạo được các bảng tạm, sau đó mới WHERE, sau WHERE lại tạo bảng tạm, sau đó mới GROUP, sau đó mới là HAVING, rồi ORDER BY. Cuối cùng mới là SELECT.

Trong quá trình chạy từ FROM cho tới ORDER BY, thì xử lý diễn ra hoàn toàn ở trên các DÒNG. Nghĩa là các cột gần như vẫn còn tồn tại. Chỉ khi nào SELECT chạy thì mới bỏ bớt các cột.

Cho nên khi ta GROUP BY GioiTinh, ta vẫn count(DiaChi) được mà không lỗi, vì thật ra cột DiaChi vẫn còn sống sốt sau khi GROUP BY.

Tóm lại, phải nhớ là SELECT chạy sau cùng. Trong quá trình xử lý, SQL chỉ quan tâm đến xử lý các dòng. Các cột chỉ biến mất sau khi chúng ta chạy đến SELECT.

Câu số 14: Vì sao không được phép SELECT những cột không có trong mệnh đề GROUP BY?

Câu lệnh sau sẽ lỗi:



Nhưng vì sao lại lỗi?

Vì HoTenSV không có hay vì có nhưng lấy không được?

Chắc chắn là HoTenSV vẫn còn, vì mình chưa hoàn thành SELECT. Khi nào SELECT xong thì mới mất bớt cột đi. Còn đang SELECT thì vẫn còn tất cả các cột.

Vậy vì sao có mà không lấy được?

Vì mỗi GROUP chỉ được trả về 1 dòng kết quả.

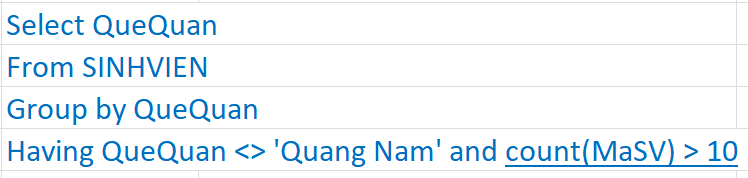
Mà mỗi nhóm thì có rất nhiều SV, mỗi SV lại có HoTenSV khác nhau. Cho nên đâu biết lấy HoTenSV của SV nào để trả về.

Giống như là khi lớp đứng xếp hàng, mỗi hàng chỉ có 1 cái ghế, thì không biết cho học sinh nào trong hàng đó được ngồi vào ghế đó.

Tức là, sau GROUP BY mình tưởng tượng giống như là các nhóm đang đứng xếp hàng sẵn để chờ chúng ta quyết định sẽ SELECT cái gì.

Câu số 15: Trong 2 câu SQL sau, câu nào sẽ bị lỗi?





Màu xanh hay màu đỏ sẽ bị lỗi?

Câu đỏ sai vì hàm gộp không được nằm trong WHERE. Vì WHERE chạy trước GROUP BY. Mà GROUP BY xong rồi mới được gọi hàm GỘP.

Vì sao trong SELECT lại có hàm GỘP? Vì SELECT chạy sau cùng.

Câu số 16: Trong 2 câu SQL sau, câu nào sẽ bị lỗi?



Xanh hay đỏ bị lỗi?

Câu nào cũng chạy được.

Kết quả ra giống hệt nhau. Nhưng cách thực hiện thì khác nhau.

Câu đỏ: Loại ngay từ đầu những sinh viên có quê quán là Quảng Nam. Sau đó thì không có nhóm Quảng Nam.

Câu xanh: Vẫn tạo ra nhóm Quảng Nam nhưng loại nhóm đó.

Kết quả vẫn là không có nhóm Quảng Nam.

Câu nào sẽ tối ưu hơn? Câu đỏ. Nếu không yêu người ta thì đừng có chờ đến sau 14/2 mới trả lời hỉ.