|  |  |
| --- | --- |
| What is Database? What is Database Management system | * Database : là cơ sở dữ liệu lưu trữ dữ liệu * Database Management system : là gói các phần mềm dùng để thiết kế, truy xuất và quản lý dữ liệu trong csdl * => dựa trên mô hình quan hệ * **Relational** Database là cơ sở dữ liệu được sử dụng phổ biến nhất. Nó chứa các bảng và mỗi bảng có Primary Key riêng. * <https://vietjack.com/sql/khai_niem_rdbms.jsp> |
| How many kind of database? | * CSDL Tập trung * CSDL Phân Phối * CSDL Cá Nhân * CSDL Người Dùng * CSDL Thương mại * CSDL NoSQL * CSDL quan hệ * CSDL đám mây * CSDL biểu đồ * <https://websolutions.com.vn/database-la-gi-cac-loai-database-hien-nay/> |
| What is Relational database? Relational database management system? | * RDBMS là viết tắt của **R**elational **D**atabase **M**anagement **S**ystem ((Hệ thống quản lý cơ sở dữ liệu quan hệ)). Tất cả các hệ thống quản lý cơ sở dữ liệu hiện đại như SQL, MS SQL Server, ORACLE, … là dựa trên RDBMS. Nó được gọi là RDBMS bởi vì nó dựa trên Relational Model (Mô hình quan hệ) * dữ liệu được biểu diễn bởi các hàng. Relational Database là cơ sở dữ liệu được sử dụng phổ biến nhất. Nó chứa các bảng và mỗi bảng có Primary Key riêngx. Bởi vì các bảng này được tổ chức chặt chẽ nên việc truy cập dữ liệu trở nên dễ dàng hơn trong RDBMS. |
| What is table? column? and row? | * Table : lưu trữ dữ liệu * Column(Cột) : là thực thể dọc trong bảng, chứa thông tin liên kết với 1 trường cụ thể trong bảng * Row(hàng) : dc coi là 1 bản ghi. Chứa thoogn tin cụ thể của các Field ( trường : id,ten,msv…) * Database -> Table -> Column, Row |
| How many type to store a number in MySQL? | * <https://o7planning.org/vi/10321/du-lieu-va-cau-truc-trong-mysql#a206009> -> all in here * Kiểu số nguyên : TINYINT, SMALLINT,MEDIUMINT, INT, BIGINT * Kiểu số thực : FLOAT,DOUBLE * Nhị phân : DECIMAL * Kiểu : BIT |
| How many way to store string into MySQL? | * Char,varchar,binary,varbinary,blob,text,enum,set |
| Diffrent between CHAR(255), VARCHAR(255) and TEXT, MEDIUMTEXT, LONGTEXT? | * Char(255) : Khoảng trắng phía trước dc loại bỏ – có giá trị từ 0-255 * Varchar(255) : Giống như lưu trữ, giá trị từ 0-65,535 * Text: lưu trữ giá trị dạng chuỗi có kỹ tự mã hóa * Medium text : độ dài tối đa hơn TEXT * Longtext : độ dài nhiều nhất |
| How to store date time value in MySQL? Diffrent between date/time types | * Date, time, datetime,timestamp, year * Date: Chỉ Lưu trũ thông tin ngày tháng(YYYY-MM-DD) * DateTime : thông tin ngày tháng và thời gian (YYYY-MM-DD HH:MM:SS) * Timestamp : thông tin ngày tháng thời gian nhưng chuyển từ múi giờ hiện tại sang UTC khi lưu trữ, khi lấy giữ liệu ra sẽ trở lại múi giờ hiện tại * Year(4) – YYYY : phạm vi từ năm 1901 tới 2155 |
| Is it possible to store binary (like Image file) into MySQL? How to do that? | * Có thể – Lưu và kiểu Bolb   CREATE TABLE 'test'.'pic' (  'idpic' INTEGER UNSIGNED NOT NULL AUTO\_INCREMENT,  'caption' VARCHAR(45) NOT NULL,  'img' LONGBLOB NOT NULL,  PRIMARY KEY ('idpic')  ) |
| Diffrent the BLOB vs CLOB? | * BOLB : BLOB là một đối tượng nhị phân lớn (Binary Large OBject) có thể chứa một lượng lớn dữ liệu. * Ví dụ : data (**videos, images, documents, other**) * CLOB : cho dữ liệu văn bản lớn ( TEXT) |
| What is Primary key? Why do we need it? How to define the PK for Entity? | * Primary key là một trường hoặc kết hợp nheieuf trường và được xác định là 1 bản ghi duy nhất. Mỗi table chỉ có một primary key duy nhất, mỗi primary key có thể có một hoặc nhiều column. * 2 way : a natural key or a surrogate key. * Natural key : định danh của bản ghi theo logic của dữ liệu bảng SanPham (sản phẩm) có khóa tự nhiên là kết hợp của HangSX (hãng sản xuất) và TenSP (tên sản phẩm). * Surrogate :  là khóa không có liên hệ nào với dữ liệu mà hoàn toàn chỉ được dùng làm định danh cho bản ghi. ( ví dụ là thuộc tính tự tăng,, ko dính dáng gì với bản ghi |
| What is Foregin key? Why do we need it? Which value can be stored in FK column? Is it possible to store NULL value in FK column? | * Foregin key:  **là mối quan hệ giữa hai bảng và mối quan hệ này ta hay gọi là cha - con**, nghĩa là nếu bảng A có một thuộc tính liên kết tới bảng B thì lúc này bảng B đóng vai trò là cha và bảng A đóng vai trò là con. * SET NULL : Có thể xảy ra khi **FK** là 1 phần của **PK** của 1 **Table** |
| What is Contraints? Why/when we use it? Give some example of contraints? | * Contraints : Ràng buộc -  **là những quy tắc được áp dụng trên các cột dữ liệu, trên bảng**. Được sử dụng để kiểm tra tính hợp lệ của dữ liệu vào, đảm bảo tính chính xác, tính toàn vẹn của dữ liệu. * Ví dụ : Thiết lập điều kiên **NOTNULL**( để cột k nhận giá trị null) , **PK,FK**, **Check**(ktra tính hợp lệ) , **unique** ( đảm bảo dữ liệu của cột là duy nhất) |
| - What is Index? Why/when should we need index? The advantages/disadvantages of index? | * **Index** : một chỉ mục là một con trỏ chỉ tới từng giá trị xuất hiện trong bảng/cột được đánh chỉ mục. * Advantage **:** tăng tốc câu lệnh truy vấn – tạo một hàng duy nhất ko trùng - Nếu chỉ mục được đặt thành chỉ mục điền văn bản, thì chúng ta có thể tìm kiếm theo các giá trị chuỗi lớn. ví dụ để tìm một từ trong câu, v.v. * Disadvantage :tốn tài nguyên - **indexes slow down** INSERT,UPDATE and DELETE, **but will speed up** UPDATE if the WHERE condition has an **indexed field.** (INSERT, UPDATE and DELETE **becomes slower because** on each operation the indexes **must also be updated. )** |
| How many kind of index? | * Clustered index : tự tạo khi lập khi bảng có PK, các dữ liệu dc sắp xếp theo bảng chữ cái, 1 bảng có duy nhất 1 Clus(chính là unique index) * Non Clustered index : ko sắp xếp theo bảng chữ cái mà bị rối tung lên miễn sao nó theo logic index quy định |
| - What is Entity? Entity Relational Diagram (ERD)? | * Entity : mô hình Mối quan hệ thực thể   Ví dụ: “Quản lý các dự án của công ty”   * Một nhân viên là một **thực thể** * Tập hợp các nhân viên là **tập thực thể** * Một dự án là một thực thể * Tập hợp các dự án là tập thực thể * Một phòng ban là một thực thể * Tập hợp các phòng ban là tập thực thể   **Ví dụ 1:** tập thực thể NHANVIEN có các thuộc tính   * Mã NV (MaNV: integer) * Họ tên (Hoten: string[50]) * Ngày sinh (ns:date) * Địa chỉ (diachi:string[100]) * Quê quán (quequan:string[30]) * Hệ số lương (hsluong:float) * Hệ số phụ cấp (hsphucap:float) * Tổng lương (tongluong:float)   ERD : là **bản vẽ thiết kế của database trong hệ thống**. Nó nói cho chúng ta biết **database sẽ có những table nào**, **có thuộc tính** gì **và mối quan hệ** của những table này ra sao. |
| How many type of relation between entities? | * 1:1( qhe 1-1) **ví dụ** một người đàn ông chỉ có thể có 1 vợ và ngược lại ) * 1:n (qhe một nhiều) :  **ví dụ** : một đứa trẻ chỉ có duy nhất 1 người cha nhưng người cha thì có thể có nheieuf đứa con ) * m : n (qhe nhiều nhiều) : ví dụ , một học sinh có thể ghi danh nhiều lớp học, và lớp học có thể ghi danh nhiều học sinh |
| How to design table for relation 1-1, 1-n and m-n? | * 1:1 : khi một hàng trong bảng A có thể **liên kết với chỉ một** hàng của bảng B( **ví dụ** : một đát nước chỉ có 1 thủ đô và ngược lại ) * 1:n : một hàng trong bảng A có thể liên kết với nhiều hàng trong bảng B , nhưng một hàng trong bảng B dược liên ketets với chỉ một hàng trong bảng A * M : n : Thường thực hiên bằng bảng kết hợp ( bảng nối ). AB với mối qhe 1: n và ngược lại. Trong trường hợp này, khóa chính của AB dc hình thành từ 2 khóa ngoại (bản sao của khóa chính bảng A,B) |
| What is normalization? | * Chuẩn hóa dữ liệu : Chuyển bảng có cấu trúc phức tạp thành nheieuf bảng có cấu trúc đơn giản, giảm dư thừa và giải phóng bộ nhớ, tăng tốc độ, độ chính xác và hiệu quả |
| How many level of Normalization? explains and sample them | * Càng chuẩn hóa cao thì dư thừa càng ít * 1NF : Phân tách lặp * 2NF: Loại bỏ phụ thuộc hàm không hoàn toàn vào kháo chính * 3NF : loại bỏ phụ thuộc hàm bắc cầu vào khóa chính * BCNF : các thuộc tính quyết định phỉa là khóa * <https://expressmagazine.net/development/698/chuan-hoa-co-so-du-lieu-database-normalization> |
| Create/alter/drop table/column/key | * Create : tạo bảng bằng cách xác định tất cả cột và loại data của nó   CREATE TABLE COURSE  (  Course\_ID Int,  Course\_Name Varchar(10)  )  Alter : thay đổi : Alter TABLE <Table name> ADD Column1 datatype, Column2 datatype;    Drop : delete table DROP TABLE <tableName>; |
| How to define default value for column? | * Cung cấp giá trị mặc định của cột , giá trị mặc định sẽ thêm vào tất cả bản ghi mới nếu ko có giá trị nào dc chỉ định ALTER TABLE Persons ADD CONSTRAINT df\_City DEFAULT 'Sandnes' FOR City; |
| How to insert data into table? | * INSERT INTO table\_name (column1, column2, column3, ...) VALUES (value1, value2, value3, ...); |
| When default value will be set to column? | * giá trị mặc định sẽ thêm vào tất cả bản ghi mới nếu ko có giá trị nào dc chỉ định |
| What is "Insert on duplicate" how to use it? | * Bản sao sẽ có tác dụng vừa cập nhật và thêm mới bản ghi mà không cần phải sử lệnh 2 lệnh: update và insert. Giúp cải thiện tốc độ sử lý sql.INSERT ON DUPLICATE KEY UPDATE * INSERT INTO ins\_duplicate VALUES (1,'Antelope'**) ON DUPLICATE KEY UPDATE** animal='Antelope'; |
| "REPLACE" syntax and how to use it? | * SELECT REPLACE('SQL Tutorial', 'T', 'M'); * => SQL MuMorial * Dùng để thay thế tất cả lần xuất hiện của một chuỗi con trong một chuỗi thành một chuỗi con mới |
| How to update value in database? | * UPDATE table\_name SET column1 = value1, column2 = value2, ... WHERE condition; |
| How to delete row in table? | DELETE FROM table\_name |  2) DELETE FROM table\_name WHERE |  example\_column\_id IN (1,2,3) |
| What happen if we delete row which has foregin key? | các hàng trong bảng tham chiếu sẽ tự động bị xóa khi hàng tham chiếu của chúng bị xóa. |
| What is the TRUNCATE syntax? when to use it? Diffrerent between DELETE all? | * Truncate : TRUNCATE TABLE table\_name; Xóa data trong bảng, ko xóa bảng * Drop table : xóa cả bảng * **Khác** **nhau** : **TRUNCATE** xóa tất cả hàng khỏi bảng, ko thể khôi phục 🡺 nhanh hơn và ko sử dụng nhiều ko gian như delete * **Delete** : thường dùng ddeerr xóa 1 hàng khỏi bảng với thêm câu lệnh WHERE( nếu ko dùng where thì cũng sẽ xóa hết ) |
| How to retrive data from database? | * SELECT \* FROM table\_name |
| What is condition? How many kind of condition? (IN, BETWEEN, equal, greater,...) | * **IN condition** : cho phép bạn chỉ định nhiều giá trị trong mệnh đề WHERE. * Ví dụ : * SELECT column\_name(s) FROM table\_name WHERE column\_name IN (value1, value2, ...); * BETWEEN : chọn các giá trị trong một phạm vi nhất định. Các giá trị có thể là số, văn bản hoặc ngày. * Ví dụ : SELECT column\_name(s) FROM table\_name WHERE column\_name BETWEEN value1 AND value2 * EQUAL : dấu = * GREATER : dấu lớn hơn > |
| The ordering in select query? | * **Theo thứ tự mà các dữ liệu dc tạo** |
| What is LIMIT, OFFSET, how and when to use it? | * **LIMIT : mệnh đề sử dụng với SELECT để hạn chế số lượng hàng khi in ra kết quả theo mong muốn** * **OFFSET : sử đụng để chỉ định phần bù của hàng đầu tiên trả về** * **COUNT : chỉ định số lượng hàng tối đa trả về**   SELECT column1, column2, ...  FROM table\_name  LIMIT offset, count;  **Ví dụ :** SELECT \* FROM Data LIMIT 3;  => sẽ trỏ tới 3 hàng đầu tiên của bảng DATA  SELECT \* FROM Data LIMIT 1, 2;  => trỏ tới hàng thứ 2,3 |
| What is GROUP BY? | * **Nhóm các hàng có cùng gia trị thành một hàng tổng quát** * SELECT column\_name(s) FROM table\_name WHERE condition GROUP BY column\_name(s)ORDER BY column\_name(s); * Ví dụ SELECT COUNT(CustomerID), Country FROM Customers GROUP BY Country; * <https://www.w3schools.com/sql/sql_groupby.asp> |
| Diffrerent between WHERE and HAVING? | * **HAVING : kiểm tra điều kiện sau khi GROUP BY diễn ra**   select City, CNT=Count(1)  From Address  Where State = Vietnam  Group By City  Having Count(1)>5 🡺 show ra 1 bảng với các thành phố ở Vietnam và hơn 5 điaj chỉ và số lượng của mỗi thành phố  **WHERE : kiểm tra điều kiện trước khi GROUP BY diễn ra**  select City, CNT=Count(1)  From Address  Where State = vietnam  Group By City 🡺 sẽ show ra thành phố ở Việt nam và địa chỉ của thành phố |
| When/why/how to use JOIN? | * **Join : dùng khi kết hợp các hàng từ 2 hoặc nhiều bảng, dựa trên 1 trường có liên quan tới nhau** * <https://www.w3schools.com/sql/sql_join.asp> * Ví dụ : SELECT Orders.OrderID, Customers.CustomerName, Orders.OrderDate FROM Orders INNER JOIN Customers ON Orders.CustomerID=Customers.CustomerID; |
| Different between INNER JOIN vs LEFT JOIN vs RIGHT JOIN vs FULL JOIN? | * **INNER JOIN : trả về các bản ghi có giá trị khớp trong cả 2 bảng** * **LEFT (OUTER) JOIN : trả vè tất cả các bản ghi từ bên trái và các bản ghi khớp từ bảng bên phải** * **RIGHT : ngược lại** * **FULL : trả veef tất cả các bản ghi khi có một giá trị khớp trong bảng bên trái hoặc phải** |
| What is sub query? give some examples? | * truy vấn trong một truy vấn SQL khác và được nhúng trong mệnh đề WHERE.. * Đặt trong dấu ngoặc đơn * Ví dụ : **dùng với SELECT** * SELECT \* * FROM CUSTOMERS * WHERE ID IN (SELECT ID * FROM CUSTOMERS * WHERE SALARY &amp;gt; 4500) ; * 🡺 trả về ID trong bảng có lương cao hơn 4500 * **Với update :**   UPDATE CUSTOMERS     SET SALARY = SALARY \* 0.25     WHERE AGE IN (SELECT AGE FROM CUSTOMERS\_BKP        WHERE AGE &amp;gt;= 27 )  **Với delete :** DELETE FROM CUSTOMERS     WHERE AGE IN (SELECT AGE FROM CUSTOMERS\_BKP        WHERE AGE &amp;gt;= 27 ); |
| What is function in MySQL? | * một đối tượng trong cơ sở dữ liệu bao gồm **một tập nhiều câu lệnh được nhóm lại với nhau và được tạo ra với mục đích sử dụng lại.**Create funciton :   CREATE FUNCTION function\_name [ (parameter datatype [, parameter datatype]) ]  RETURNS return\_datatype  BEGIN  declaration\_section  executable\_section  END;   * Drop funciton : xóa   DROP FUNCTION [ IF EXISTS ] function\_name;   * <https://www.techonthenet.com/mysql/functions.php> |
| - What is Transaction? | * **là tiến trình thực hiện một nhóm các câu lệnh SQL** * Có một số câu lệnh : * Commit: lưu thay đổi * Rollback : quay laij trạng thái trc khi có thay đổi * Savepoint : tạo điểm point để khi rollback về đúng điểm đó * Set transaction : đặt tên cho transaction * Ví dụ :   SQL> DELETE FROM NHANVIEN  WHERE TUOI = 25; SQL> COMMIT;   * <https://quantrimang.com/transaction-trong-sql-162443> |