**CÂU LỆNH TRONG MYSQL - DinhNT**

--- **NỘI DUNG SLIDE 4 - PHẦN 1 (Ngôn ngữ định nghĩa dữ liệu DDL)** ---

--Tạo database

CREATE DATABASE <tên\_database>;

**ví dụ:**

    CREATE DATABASE qlnhanvien;

--Tạo databse có charset

CREATE DATABASE <tên\_database> CHARACTER SET <...> COLLATE <...>;

**ví dụ:**

    CREATE DATABASE mydatabase CHARACTER SET utf8 COLLATE utf8\_unicode\_ci;

--Tạo bảng

CREATE TABLE  <tên\_bảng> (

    <thuộc\_tính\_1>  <kiểu\_dữ\_liệu[miền\_giá\_trị]>,

    <thuộc\_tính\_2>  <kiểu\_dữ\_liệu[miền\_giá\_trị]>,

    ...

);

**ví dụ:**

    CREATE TABLE NHAN\_VIEN (

        ID\_NHANVIEN  INT            NOT NULL,

        HO\_NV        VARCHAR(20)    NULL,

        TEN\_NV       VARCHAR(25)    NOT NULL,

        NGAY\_SINH    DATE,

        LUONG        INT            NULL,

        PHG          CHAR(5)        NULL

    );

--Tạo bảng có khóa chính (PK) và khóa ngoại (FK)

CREATE TABLE  <tên\_bảng> (

    <thuộc\_tính\_1>  <kiểu\_dữ\_liệu[miền\_giá\_trị]>,

    <thuộc\_tính\_2>  <kiểu\_dữ\_liệu[miền\_giá\_trị]>,

    ...,

    PRIMARY KEY (<thuộc\_tính\_làm\_khóa\_chính>),

    FOREIGN KEY (<thuộc\_tính\_làm\_khóa\_ngoại>)

    REFERENCES <tên\_bảng\_liên\_kết\_khóa\_ngoại> (<thuộc\_tính>)

);

**ví dụ:**

    CREATE TABLE NHAN\_VIEN (

        ID\_NHANVIEN  INT            NOT NULL,

        HO\_NV        VARCHAR(20)    NULL,

        TEN\_NV       VARCHAR(25)    NOT NULL,

        NGAY\_SINH    DATE,

        LUONG        INT            NULL,

        PHG          CHAR(5)        NULL,

        PRIMARY KEY (ID\_NHANVIEN),

        FOREIGN KEY (PHG) REFERENCES PHONG\_BAN(MA\_PHONG)

    );

--- **NỘI DUNG SLIDE 4 - PHẦN 2 (Các câu lệnh thay đổi và xoá bảng)** ---

--Thêm MỘT cột vào bảng có sẵn

ALTER TABLE <tên\_bảng>

ADD COLUMN <tên\_thuộc\_tính> <kiểu\_dữ\_liệu[miền\_giá\_trị]>;

**ví dụ:**

    ALTER TABLE NHAN\_VIEN

    ADD COLUMN EMAIL VARCHAR(20);

--(\*)Thêm NHIỀU cột vào bảng có sẵn

ALTER TABLE <tên\_bảng>

ADD COLUMN <tên\_thuộc\_tính\_1> <kiểu\_dữ\_liệu[miền\_giá\_trị]>,

ADD COLUMN <tên\_thuộc\_tính\_2> <kiểu\_dữ\_liệu[miền\_giá\_trị]>,

...;

**ví dụ:**

    ALTER TABLE NHAN\_VIEN

    ADD COLUMN EMAIL VARCHAR(20),

    ADD COLUMN SDT VARCHAR(20);

--(\*)Chỉnh sửa thông tin MỘT cột trong bảng có sẵn

ALTER TABLE <tên\_bảng>

MODIFY COLUMN <tên\_thuộc\_tính> <kiểu\_dữ\_liệu[miền\_giá\_trị]>;

**ví dụ:**

    ALTER TABLE PHONG\_BAN

    MODIFY COLUMN tenPB varchar(50);

--(\*)Chỉnh sửa thông tin NHIỀU cột trong bảng có sẵn

ALTER TABLE <tên\_bảng>

MODIFY COLUMN <tên\_thuộc\_tính\_1> <kiểu\_dữ\_liệu[miền\_giá\_trị]>,

MODIFY COLUMN <tên\_thuộc\_tính\_2> <kiểu\_dữ\_liệu[miền\_giá\_trị]>,

...;

**ví dụ:**

    ALTER TABLE PHONG\_BAN

    MODIFY COLUMN email varchar(100),

    MODIFY COLUMN tenPB varchar(50);

--Xóa MỘT cột trong bảng có sẵn

ALTER TABLE <tên\_bảng>

DROP COLUMN <tên\_cột\_cần\_xóa>;

**ví dụ:**

    ALTER TABLE PHONG\_BAN

    DROP COLUMN EMAIL;

--(\*)Xóa NHIỀU cột trong bảng có sẵn

ALTER TABLE <tên\_bảng>

DROP COLUMN <tên\_cột\_cần\_xóa\_1>,

DROP COLUMN <tên\_cột\_cần\_xóa\_2>,

...;

**ví dụ:**

    ALTER TABLE PHONG\_BAN

    DROP COLUMN EMAIL,

    DROP COLUMN DIACHI;

--Thêm ràng buộc KIỂM TRA vào bảng

ALTER TABLE <tên\_bảng>

ADD CONSTRAINT <tên\_ràng\_buộc>

CHECK (<điều\_kiện>);

**ví dụ:**

    ALTER TABLE NHAN\_VIEN

    ADD CONSTRAINT CHK\_SALARY\_MIN

    CHECK (LUONG >= 100);

    (\*) ***Ràng buộc trong khoảng:***

    ALTER TABLE NHAN\_VIEN

    ADD CONSTRAINT CHK\_SALARY\_MIN

    CHECK (contact\_id BETWEEN 100 AND 200);

--LOẠI BỎ ràng buộc

ALTER TABLE <tên\_bảng>

DROP CONSTRAINT <tên\_ràng\_buộc\_cần\_xóa>;

**ví dụ:**

    ALTER TABLE NHAN\_VIEN

    DROP CONSTRAINT CHK\_SALARY\_MIN;

--Thêm ràng buộc KHÓA CHÍNH vào bảng

ALTER TABLE <tên\_bảng>

ADD CONSTRAINT <tên\_ràng\_buộc>

PRIMARY KEY (<thuộc\_tính\_khóa\_chính>,[<thuộc\_tính\_khóa\_chính\_2>]);

**ví dụ:**

    ALTER TABLE PHONG\_BAN

    ADD CONSTRAINT PRI\_PHONGBAN

    PRIMARY KEY (MaPB);

    ALTER TABLE CTHD

    ADD CONSTRAINT PRI\_CTHD

    PRIMARY KEY (MaHD, MaSP);

--LOẠI BỎ ràng buộc KHÓA CHÍNH trong bảng

ALTER TABLE <tên\_bảng\_cần\_xóa\_khóa\_chính>

DROP PRIMARY KEY;

**ví dụ:**

    ALTER TABLE NHAN\_VIEN

    DROP PRIMARY KEY;

--Thêm ràng buộc KHÓA NGOẠI vào bảng

ALTER TABLE <tên\_bảng>

ADD CONSTRAINT <tên\_ràng\_buộc>

FOREIGN KEY (<thuộc\_tính\_làm\_khóa\_ngoại>)

REFERENCES <tên\_bảng\_liên\_kết>(<khóa\_chính\_liên\_kết\_với\_khóa\_ngoại>);

**ví dụ:**

    ALTER TABLE NHAN\_VIEN

    ADD CONSTRAINT FK\_PHONGBAN\_NHANVIEN

    FOREIGN KEY (PHG)

    REFERENCES PHONG\_BAN(MAPB);

--LOẠI BỎ ràng buộc KHÓA NGOẠI trong bảng

ALTER TABLE <tên\_bảng\_cần\_xóa\_khóa\_chính>

DROP FOREIGN KEY <tên\_ràng\_buộc\_cần\_xóa>;

và

ALTER TABLE <tên\_bảng\_cần\_xóa\_khóa\_chính>

DROP INDEX <tên\_ràng\_buộc\_cần\_xóa>;

**ví dụ:**

    ALTER TABLE NHAN\_VIEN

    DROP FOREIGN KEY FK\_PHONGBAN\_NHANVIEN;

và

    ALTER TABLE NHAN\_VIEN

    DROP INDEX FK\_PHONGBAN\_NHANVIEN;

--Thêm ràng buộc UNIQUE

ALTER TABLE <tên\_bảng>

ADD CONSTRAINT <tên\_ràng\_buộc>

UNIQUE (<tên\_cột\_cần\_ràng\_buộc>);

**ví dụ:**

    ALTER TABLE NHAN\_VIEN

    ADD CONSTRAINT NHANVIEN\_UNQ\_EMAIL

    UNIQUE (EMAIL);

--LOẠI BỎ ràng buộc UNIQUE

ALTER TABLE <tên\_bảng>

DROP CONSTRAINT <tên\_ràng\_buộc\_cần\_xóa>;

**ví dụ:**

    ALTER TABLE NHAN\_VIEN

    DROP CONSTRAINT NHANVIEN\_UNQ\_EMAIL;

--(\*) Thay đổi tên bảng

RENAME TABLE `<tên\_bảng\_cần\_thay\_đổi>` TO  `<tên\_bảng\_mới>`;

**ví dụ:**

    RENAME TABLE `PHOGNN\_BAN` TO `PHONG\_BAN`;

--(\*) Thay đổi thông tin của MỘT cột trong bảng

ALTER TABLE <tên\_bảng>

CHANGE COLUMN `<tên\_cột\_cần\_thay\_đổi>` `<tên\_mới>` <kiểu\_dữ\_liệu>;

**ví dụ:**

    ALTER TABLE NHANVIEN

    CHANGE COLUMN `CMND` `CCCD` int;

--Xóa bảng

DROP TABLE <tên\_bảng>;

**ví dụ:**

    DROP TABLE NHAN\_VIEN;

--Xóa bảng và tự động loại bỏ các ràng buộc tham chiếu trong bảng

**(chạy từng câu lệnh bên dưới)**

SET GLOBAL FOREIGN\_KEY\_CHECKS = 0;

DROP TABLE <tên\_bảng>;

SET GLOBAL FOREIGN\_KEY\_CHECKS = 1;

**ví dụ:**

    SET GLOBAL FOREIGN\_KEY\_CHECKS = 0;

    DROP TABLE NHAN\_VIEN;

    SET GLOBAL FOREIGN\_KEY\_CHECKS = 1;

--Xóa database

DROP DATABASE <tên\_database\_cần\_xóa>;

**ví dụ:**

    DROP DATABASE QLNHANVIEN;

--- **NỘI DUNG SLIDE 5 - PHẦN 1 (Truy vấn dữ liệu trên MỘT bảng)** ---

**Cú pháp:**

SELECT [DISTINCT] Column(s)

FROM <tên\_bảng>, <views>

[WHERE  Conditions]

[ORDER BY Column(s) [asc|desc]]

[GROUP BY Row(s)]

(Các mệnh đề trong cặp dấu **[]** không bắt buộc)

**Trong đó:**

**DISTINCT** trả về các bản ghi **không trùng lặp nhau**

**WHERE** cho phép truy vấn lựa chọn **theo hàng**

**ORDER BY** cho phép sắp xếp dữ liệu **theo cột**

**GROUP BY** cho phép nhóm dữ liệu **theo hàng**

--Truy vấn lựa chọn tất cả các hàng và cột

SELECT \* FROM <tên\_bảng>;

**ví dụ:**

    SELECT \* FROM NHAN\_VIEN;

--Truy vấn lựa chọn một số cột

SELECT <tên\_cột\_1>, <tên\_cột\_2>, ... FROM <tên\_bảng>;

**ví dụ:**

    SELECT MaNV, HoTenNV, LUONG FROM NHAN\_VIEN;

--Thay đổi tên cột hiển thị trong tập kết quả trả về

SELECT <tên\_cột\_1> AS `<tên\_cần\_đổi\_cột\_1>`,

       <tên\_cột\_2> AS `<tên\_cần\_đổi\_cột\_2>`,

       ...

FROM <tên\_bảng>;

**ví dụ:**

    SELECT MaNV AS `Mã Nhân Viên`,

           HoTenNV AS `Tên Nhân Viên`,

           LUONG AS `Lương`

    FROM NHAN\_VIEN;

--Mệnh đề **CONCAT** - Nối nội dung của những cột với nhau

SELECT CONCAT(<tên\_cột\_1>,<tên\_cột\_2>,<tên\_cột\_3>,...)

FROM <tên\_bảng>;

hoặc

SELECT <tên\_cột\_1> + <tên\_cột\_2> + <tên\_cột\_3> + ...

FROM <tên\_bảng>;

**ví dụ:**

    SELECT CONCAT(HoNV,' ',TenNV)

    FROM NHAN\_VIEN;

    hoặc

    SELECT HoNV + ' ' + TenNV

    FROM NHAN\_VIEN;

--Mệnh đề **DISTINCT** - Loại bỏ các hàng trùng nhau trong tập kết quả

SELECT DISTINCT <tên\_cột>

FROM <tên\_bảng>;

**ví dụ:**

    SELECT DISTINCT HoNV

    FROM NHAN\_VIEN;

--Mệnh đề **LIMIT** - dùng để hiển thị N hàng đầu tiên trong bảng

SELECT \* FROM <tên\_bảng> LIMIT **<N>**;

hoặc

SELECT <tên\_cột\_1>, <tên\_cột\_2>, … FROM <tên\_bảng> LIMIT **<N>**;

Trong đó **<N>** là số hàng đầu tiên cần hiển thị

**ví dụ:**

    SELECT \* FROM NHAN\_VIEN LIMIT **5**;

    hoặc

    SELECT MaNV, HoTenNV, LUONG FROM NHAN\_VIEN LIMIT **5**;

--Mệnh đề **WHERE** - Câu điều kiện

**Cú pháp:**

SELECT [DISTINCT] **Column(s)**

FROM <tên\_bảng>

[WHERE  <điều\_kiện>]

Một số toán tử (Operator) sử dụng trong biểu thức Conditions:

* Toán tử so sánh : **>** , **<** , **>=** , **<=** , **<>**
* Toán tử logic : **AND**, **OR**, **NOT**
* So sánh chuỗi dùng toán tử **LIKE**
* **BETWEEN** … **AND** , **IN**

So sánh dùng toán tử **LIKE**

|  |  |  |
| --- | --- | --- |
| **Kí tự đại diện** | **Mô tả** | **Ví dụ** |
| \_ | Đại diện cho 1 kí tự | SELECT \*  FROM NHAN\_VIEN  WHERE HO\_NV LIKE 'H\_' |
| % | Đại diện cho một chuỗi kí tự có độ dài bất kì | SELECT \*  FROM NHAN\_VIEN  WHERE TEN\_NV LIKE 'B%' |

--Làm việc với Ngày/Tháng/Năm (Date)

Hiển thị tất cả các thông tin dự án có ngày bắt đầu từ ngày 01/01/2017

SELECT \*

FROM du\_an

WHERE DATE(ngay\_batdau) >= '2017-01-01'

--Lấy ngày, tháng, năm trong cột

* DAY(<tên\_cột\_>) : lấy ngày
* MONTH(<tên\_cột\_>) : lấy tháng
* YEAR(<tên\_cột\_>) : lấy năm
* CURDATE() hoặc CURRENT\_DATE() để lấy ngày hiện tại

**ví dụ:**

    Hiển thị danh sách các hoá hơn có ngày mua hàng trong năm 2016

SELECT \*

FROM HOA\_DON

WHERE YEAR(ngay\_mua) = '2016'

    Hiển thị danh sách các hoá hơn có ngày mua bằng với ngày hiện tại

SELECT \*

FROM HOA\_DON

WHERE CURRENT\_DATE() = ngay\_mua

--- **NỘI DUNG SLIDE 5 - PHẦN 2 (Truy vấn dữ liệu trên MỘT bảng)** ---

1. **MAX** - Tìm giá trị **LỚN** nhất trong cột

**ví dụ:**

    Hiển thị mức lương CAO NHẤT trong bảng nhân viên

    SELECT **MAX**(Luong)

    FROM NHAN\_VIEN

2. **MIN** - Tìm giá trị **NHỎ** nhất trong cột

**ví dụ:**

    Hiển thị mức lương THẤP NHẤT trong bảng nhân viên

    SELECT **MIN**(Luong)

    FROM NHAN\_VIEN

3. **AVG** - Tìm giá trị trung bình của cột

**ví dụ:**

    Hiển thị mức lương TRUNG BÌNH của nhân viên trong công ty

    SELECT **AVG**(Luong)

    FROM NHAN\_VIEN

4. **SUM** - Tính tổng giá trị của cột

**ví dụ:**

    Hiển thị tổng tất cả số lượng sản phẩm có trong bảng sản phẩm

    SELECT **SUM**(SoLuong)

    FROM SAN\_PHAM

5. **COUNT** - Đếm số lượng giá trị trong cột

**ví dụ:**

    Đếm số lượng nhân viên trong công ty

    SELECT **COUNT**(\*)

    FROM NHAN\_VIEN

*hoặc*

    SELECT **COUNT**(MaNV)

    FROM NHAN\_VIEN

6. (\*) **ROUND** - Làm tròn số

**ví dụ:**

    Tính tổng lương của tất cả nhân viên trong mỗi phòng ban, **làm tròn 2 số thập phân**

    SELECT PHG, **ROUND**(SUM(Luong),**2**)

    FROM NhanVien

    GROUP BY PHG

--Mệnh đề **GROUP BY**

Mệnh đề **GROUP BY** cho phép nhóm các hàng dữ liệu có giá trị giống nhau thành một nhóm

Các tính toán (thường sử dụng các hàm tổng hợp) sẽ được tính trên mỗi nhóm.

**ví dụ:**

    Đếm số lượng nhân viên trong mỗi phòng

    SELECT COUNT(\*) AS 'Số lượng',  PHG AS 'Mã PB'

    FROM NHAN\_VIEN

**GROUP BY** PHG

    Tính tổng lương của tất cả nhân viên trong mỗi phòng ban

    SELECT PHG, SUM(Luong)

    FROM NhanVien

**GROUP BY** PHG

--Mệnh đề **HAVING**

Mệnh đề **HAVING** đi kèm với **GROUP BY** giúp **loại bỏ các nhóm không thoã mãn điều kiện**

**ví dụ:**

    Hiển thị mức lương cao nhất của mỗi phòng ban. Chỉ hiển thị mức lương lớn hơn 1000$.

    SELECT PHG AS 'Mã PB', MAX(LUONG) AS 'Lương cao nhất' FROM NHAN\_VIEN

**GROUP BY** PHG

**HAVING** MAX(LUONG) > 1000

    Hiển thị mã hoá đơn và số sản phẩm được mua trong từng hoá đơn. Yêu cầu chỉ hiển thị

    hàng nào có số loại sản phẩm được mua >=5

    SELECT MaHD, COUNT(MaSP) FROM ChiTietHoaDon

**GROUP BY** MaHD

**HAVING** COUNT(MaSP) >= 5

--Mệnh đề **ORDER BY** - Cho phép sắp xếp kết quả truy vấn theo cột

Có thể sắp xếp kết quả theo chiều:Tăng dần (**ASC**) Giảm dần (**DESC**)

**ví dụ:**

    Hiển thị danh sách nhân viên theo thứ tự **TĂNG DẦN** của mã nhân viên

    SELECT \* FROM NHAN\_VIEN

**ORDER BY** ID\_NhanVien

*hoặc*

    SELECT \* FROM NHAN\_VIEN

**ORDER BY** ID\_NhanVien **ASC**

    Hiển thị danh sách nhân viên theo thứ tự **GIẢM DẦN** của mã nhân viên

    SELECT \* FROM NHAN\_VIEN

**ORDER BY** ID\_NhanVien **DESC**

--- **NỘI DUNG SLIDE 6 - PHẦN 1 (Truy vấn dữ liệu trên NHIỀU bảng)** ---

--Sử dụng **PHÉP TÍCH**

Sử dụng điều kiện **kết bằng** trong mệnh đề WHERE

Nếu xuất hiện tên cột trùng nhau trong nhiều bảng thì bắt buộc phải sử dụng **tên bảng** hoặc **bí danh** bảng trước tên cột

**Cú pháp:**

        SELECT <tên\_bảng\_1>.<tên\_thuộc\_tính>, <tên\_bảng\_2>.<tên\_thuộc\_tính>,...

        FROM <tên\_bảng\_1>, <tên\_bảng\_2>

        WHERE <tên\_bảng\_1>.<tên\_thuộc\_tính> = <tên\_bảng\_2>.<tên\_thuộc\_tính>

**ví dụ:**

    Hiển thị dữ liệu của 2 bảng **Nhân viên** và **Phòng ban** bao gồm: *mã nhân viên*,

*tên nhân viên*, *lương* và *tên phòng ban đang làm việc*

    SELECT NHAN\_VIEN.ID\_NhanVien, NHAN\_VIEN.Ten\_NV, NHAN\_VIEN.Luong, PHONG\_BAN.Ten\_PB

    FROM NHAN\_VIEN, PHONG\_BAN

    WHERE **NHAN\_VIEN.PHG** **=** **PHONG\_BAN.MA\_PB**;

    Hiển thị dữ liệu trong 3 bảng: **NHAN\_VIEN**, **DU\_AN** và **QL\_DUAN** bao gồm: *tên dự án*,

*tên nhân viên*, *ngày tham gia dự án* và *ngày kết thúc dự án*

    SELECT DU\_AN.Ten\_DuAn, NHAN\_VIEN.Ten\_NV, QL\_DUAN.Ngay\_Tham\_Gia, QL\_DUAN.Ngay\_Ket\_Thuc

    FROM NHAN\_VIEN, DU\_AN, QL\_DUAN

    WHERE **DU\_AN.MA\_DUAN =** **QL\_DUAN.MA\_DUAN** AND **NHAN\_VIEN.ID\_NhanVien =** **QL\_DUAN.MA\_NHANVIEN**;

--Dùng **BÍ DANH** cho tên bảng

Đơn giản hóa các câu truy vấn khi cần sử dụng tên bảng cho việc truy xuất các cột

**ví dụ:**

    Hiển thị dữ liệu của 2 bảng **Nhân viên** và **Phòng ban** bao gồm: *mã nhân viên*,

*tên nhân viên*, *lương* và *tên phòng ban đang làm việc*

    SELECT **nv**.ID\_NhanVien, **nv**.Ten\_NV, **nv**.Luong, **pb**.Ten\_PB

    FROM NHAN\_VIEN **nv**, PHONG\_BAN **pb**

    WHERE **nv**.PHG **=** **pb**.MA\_PB;

    Hiển thị dữ liệu trong 3 bảng: **NHAN\_VIEN**, **DU\_AN** và **QL\_DUAN** bao gồm: *tên dự án*,

*tên nhân viên*, *ngày tham gia dự án* và *ngày kết thúc dự án*

    SELECT **da**.Ten\_DuAn, **nv**.Ten\_NV, **qlda**.Ngay\_Tham\_Gia, **qlda**.Ngay\_Ket\_Thuc

    FROM NHAN\_VIEN **nv**, DU\_AN **da**, QL\_DUAN **qlda**

    WHERE **da**.MA\_DUAN = **qlda**.MA\_DUANAND **nv**.ID\_NhanVien = **qlda**.MA\_NHANVIEN;

--**INNER JOIN**

INNER JOIN trả về kết quả là các bản ghi mà trường được join ở **hai bảng khớp nhau**, các bản ghi **chỉ xuất hiện ở một trong hai bảng sẽ bị loại**

Diagram, venn diagram

Description automatically generated

**Cú pháp:**

        SELECT <thuộc\_tính\_1>, <thuộc\_tính\_2>,...

        FROM <tên\_bảng\_1> [INNER] JOIN <tên\_bảng\_2> ON <điều\_kiện\_join\_1>

                          [[INNER] JOIN <tên\_bảng\_3> ON <điều\_kiện\_join\_2>]

                          ...

**ví dụ:**

    Sử dụng câu lệnh JOIN để hiển thị dữ liệu của 2 bảng **Nhân viên** và **Phòng ban**

    SELECT ID\_NhanVien, Ho\_NV, Ten\_NV, Ten\_PB

**FROM PHONG\_BAN PB INNER JOIN NHAN\_VIEN NV**

**ON PB.Ma\_PB = NV.PHG**

    Sử dụng mệnh đề JOIN để hiển thị dữ liệu trong 3 bảng: **NHAN\_VIEN**, **DU\_AN** và

**QUANLY\_DUAN**

    SELECT da.Ten\_DaAn, nv.Ho\_NV, nv.Ten\_NV, qlda.Ngay\_Tham\_Gia, qlda.Ngay\_Ket\_Thuc

**FROM NHAN\_VIEN nv INNER JOIN QUANLY\_DUAN qlda ON nv.ID\_NhanVien = qlda.Ma\_NhanVien**

**INNER JOIN DU\_AN da ON da.Ma\_DuAn = qlda.Ma\_DuAn;**

**So sánh INNER JOIN với PHÉP TÍCH**

* **Phép tích** sẽ nhân số lượng bản ghi 2 bảng, sau đó loại bỏ các bản ghi không thỏa mãn điều kiện

**Ví dụ:** Bảng A có 3 bản ghi, bảng B có 4 bản ghi -> tích sẽ cho ra **12** bản

            ghi, sau đó sẽ loại bỏ các bản ghi không thoã mãn điều kiện: **A.MA\_PB = B.PHG**

* Với **INNER JOIN** thì trong quá trình thực hiện tích 2 bảng nó sẽ kiểm tra điều kiện ở **ON** luôn, nếu đúng thì chọn, sai thì bỏ qua.
* Như vậy xét về tốc độ truy vấn thì trường hợp sử dụng **INNER JOIN** sẽ nhanh hơn rất nhiều so với sử dụng **phép tích**

**ví dụ (NỘI DUNG SLIDE 6 - PHẦN 1 - PHÉP TÍCH VÀ INNER JOIN):**

Sử dụng JOIN hoặc phép tích để hiển thị thông tin gồm: họ và tên nhân viên, lương, tên phòng ban mà nhân viên thuộc về, tên dự án, ngày bắt đầu tham gia vào dự án và số giờ làm của nhân viên trong dự án.

**Phép tích:**

    SELECT Ho\_NV, Ten\_NV, Luong, Ten\_PB, Ten\_DuAn, Ngay\_BatDau, So\_Gio

    FROM PHONG\_BAN pb, NHAN\_VIEN nv, DU\_AN da, QUANLY\_DUAN qlda

**WHERE pb.Ma\_PB = nv.PHG AND nv.ID\_NhanVien = qlda.Ma\_NhanVien**

**AND da.Ma\_DuAn = qlda.Ma\_DuAn**

**INNER JOIN:**

    SELECT Ho\_NV, Ten\_NV, Luong, Ten\_PB, Ten\_DuAn, Ngay\_BatDau, So\_Gio

    FROM **PHONG\_BAN pb INNER JOIN NHAN\_VIEN nv ON pb.Ma\_PB = nv.PHG**

**INNER JOIN QUANLY\_DUAN qlda ON nv.ID\_NhanVien = qlda.Ma\_NhanVien**

**INNER JOIN DU\_AN da ON da.Ma\_DuAn = qlda.Ma\_DuAn**

Viết câu truy vấn hiển thị các thông tin bao gồm họ, tên, lương của nhân viên, tên dự án với điều kiện nhân viên thuộc phòng ban có tên “Thiết kế”, tham gia vào các dự án có ngày bắt đầu 01/01/2016

**Phép tích:**

    SELECT Ho\_NV, Ten\_NV, Ten\_DuAn

    FROM PHONG\_BAN pb, NHAN\_VIEN nv, DU\_AN da, QUANLY\_DUAN qlda

**WHERE pb.Ma\_PB = nv.PHG AND nv.ID\_NhanVien = qlda.Ma\_NhanVien**

**AND da.Ma\_DuAn = qlda.Ma\_DuAn**

AND pb.Ten\_PB LIKE 'Thiết kế'

AND DATE(da.Ngay\_BatDau) >= '2016-01-01'

**INNER JOIN:**

    SELECT Ho\_NV, Ten\_NV, Ten\_DuAn

    FROM **PHONG\_BAN pb INNER JOIN NHAN\_VIEN nv ON pb.Ma\_PB = nv.PHG**

**INNER JOIN QUANLY\_DUAN qlda ON nv.ID\_NhanVien = qlda.Ma\_NhanVien**

**INNER JOIN DU\_AN da ON da.Ma\_DuAn = qlda.Ma\_DuAn**

    WHERE pb.Ten\_PB LIKE 'Thiết kế' AND DATE(da.Ngay\_BatDau) >= '2016-01-01'

--- **NỘI DUNG SLIDE 6 - PHẦN 2 (Truy vấn dữ liệu trên NHIỀU bảng)** ---

\*\*MySQL không có **FULL OUTER JOIN**

--**LEFT/RIGHT JOIN**

Bảng **A** **LEFT/RIGHT JOIN** với **bảng B** thì *kết quả gồm có bản ghi* trong **bảng A**, *với các bảng ghi không có mặt* trong **bảng B** thì **các cột từ B được ghi NULL**. Các bảng ghi *chỉ có trong bảng B mà không có trong bảng A* sẽ **không được trả về**

Sử dụng **LEFT JOIN** hay **RIGHT JOIN** phụ thuộc vào vị trí của bảng trong câu truy vấn

Diagram

Description automatically generatedA yellow circle with black text

Description automatically generated with low confidence

**Cú pháp:**

        1. **LEFT JOIN**

        SELECT <thuộc\_tính\_1>, <thuộc\_tính\_2>,...

        FROM <tên\_bảng\_1> LEFT JOIN <tên\_bảng\_2> ON <điều\_kiện\_join\_1>

**2. RIGHT JOIN**

        SELECT <thuộc\_tính\_1>, <thuộc\_tính\_2>,...

        FROM <tên\_bảng\_1> RIGHT JOIN <tên\_bảng\_2> ON <điều\_kiện\_join\_1>

**ví dụ:**

Ta có dữ liệu 2 bảng **NHÂN VIÊN** và **PHÒNG BAN** như sau:

* Bảng **PHÒNG BAN**:

Table

Description automatically generated with medium confidence

* Bảng **NHÂN VIÊN**:

Table

Description automatically generated with medium confidence

Khi sử dụng **RIGHT JOIN** ta được kết quả như sau:

**Truy vấn:**

    Viết câu truy vấn hiển thị các thông tin bao gồm mã nhân viên, họ tên nhân viên, và

    tên phòng ban mà nhân viên trực thuộc. Nếu nhân viên chưa được phân bổ vào phòng nào

    thì cột tên phòng để trống

    SELECT ID\_NhanVien, Ho\_NV, Ten\_NV, Ten\_PB

**FROM PHONG\_BAN PB RIGHT JOIN NHAN\_VIEN NV**

**ON PB.Ma\_PB = NV.PHG**

**Kết quả:** Trên câu truy vấn ta thấy bảng **PHÒNG BAN** nằm bên trái và bảng **NHÂN VIÊN** nằm bên phải so với **JOIN**, nên khi sử dụng **RIGHT JOIN** ta phải lấy ***toàn bộ dữ liệu*** từ bảng **NHÂN VIÊN** (vì đang sử dụng **RIGHT JOIN** và bảng **NHÂN VIÊN** đang nằm bên phải). Khi đó những nhân viên không thuộc bất kỳ một phòng ban nào thì cột Ten\_PB sẽ để NULL còn những nhân viên thuộc một phòng ban nào đó thì cột TenPB sẽ hiển thị ra tên phòng ban tương ứng.

**Table

Description automatically generated**

Khi sử dụng **LEFT JOIN** ta được kết quả như sau:

**Truy vấn:**

    Viết câu truy vấn hiển thị các thông tin bao gồm mã nhân viên, họ tên nhân viên và

    tên phòng ban mà nhân viên trực thuộc. Nếu phòng ban nào chưa có nhân phân bổ vào

    phòng thì cột mã nhân viên, họ tên nhân viên và tên phòng ban để trống

    SELECT ID\_NhanVien, Ho\_NV, Ten\_NV, Ten\_PB

**FROM PHONG\_BAN PB LEFT JOIN NHAN\_VIEN NV**

**ON PB.Ma\_PB = NV.PHG**

**Kết quả:** Trên câu truy vấn ta thấy bảng **PHÒNG BAN** nằm bên trái và bảng **NHÂN VIÊN** nằm bên phải so với **JOIN**, nên khi sử dụng **LEFT JOIN** ta phải lấy ***toàn bộ dữ liệu*** từ bảng **PHÒNG BAN** (vì đang sử dụng **LEFT JOIN** và bảng **PHÒNG BAN** đang nằm bên trái). Khi đó những phòng ban không có bất kỳ nhân viên nào thì các cột ID\_NhanVien, Ho\_NV, Ten\_NV sẽ để NULL còn những phòng ban có nhân viên làm việc trong đó thì các cột ID\_NhanVien, Ho\_NV, Ten\_NV sẽ hiển thị ra thông tin nhân viên tương ứng làm việc trong phòng ban đó.

Table

Description automatically generated

--**SELF JOIN**

Một bảng kết nối với chính nó

Diagram, venn diagram

Description automatically generated

**ví dụ:**

    Ta có bảng dữ liệu sau:

**Table

Description automatically generated**

    Hiển thị tên nhân viên và tên người Quản Lý của nhân viên đó

    SELECT **nv1**.Ma\_NV, **nv1**.Ten\_NV, **nv2**.Ten\_NV

    FROM NHAN\_VIEN **nv1** INNER JOIN NHAN\_VIEN **nv2**

    ON **nv2**.Ma\_NV = **nv1**.Ma\_QuanLy;

--**TRUY VẤN CON**

* Là câu truy vấn **SELECT** nằm lồng bên trong một câu truy vấn khác
* Câu truy vấn con có thể được sử dụng:
  + Trong mệnh đề **WHERE** như một điều kiện tìm kiếm
  + Trong mệnh đề **HAVING** như một điều kiện tìm kiếm
  + Trong mệnh đề **FROM** như một đặc tả bảng
  + Trong mệnh đề **SELECT** như một đặc tả cột

**ví dụ:**

    Hiển thị thông tin nhân viên có mức lương lớn hơn mức lương trung bình trong công ty

    SELECT \* FROM NHAN\_VIEN

    WHERE LUONG > (**SELECT AVG(LUONG) FROM NHAN\_VIEN**)

**SO SÁNH JOIN VÀ CÂU TRUY VẤN CON**

|  |  |
| --- | --- |
| **JOIN** | **CÂU TRUY VẤN CON** |
| **Kết quả có thể bao gồm các cột của cả 2 bảng** | **Không thể bao gồm các cột của câu truy vấn con** |
| **Sử dụng mối quan hệ giữa 2 bảng** |  |
| **Chạy nhanh hơn** |  |
|  | **Có thể chuyển 1 giá trị tính toán ra câu truy vấn bên ngoài** |
|  | **Dễ viết code và dễ hiểu** |

--**IN / NOT IN**

**ví dụ: MỆNH ĐỀ IN**

    Hiển thị toàn bộ thông tin của những khách hàng có họ thuộc 1 trong 3 họ sau: *Smith,*

*Anderson, Johnson*

    SELECT \*

    FROM NHAN\_VIEN

    WHERE Ho\_NV **IN** ('Smith','Anderson','Johnson');

*Câu truy vấn trên tương đương với:*

    SELECT \*

    FROM NHAN\_VIEN

    WHERE Ho\_NV LIKE'Smith' **OR** Ho\_NV LIKE'Anderson' **OR** Ho\_NV LIKE'Johnson';

    Hiển thị toàn bộ thông tin nhân viên thuộc phòng ban chứa cụm từ “Sản xuất”

    SELECT \*

    FROM NHAN\_VIEN

    WHERE PHG **IN** (SELECT MA\_PB FROM PHONG\_BAN WHERE TEN\_PB LIKE '%Sản xuất%')

**ví dụ: MỆNH ĐỀ NOT IN**

    Hiển thị toàn bộ thông tin của những khách hàng có họ **KHÔNG** thuộc những họ sau:

*Sarah, Johnny, Dale*

    SELECT \*

    FROM NHAN\_VIEN

    WHERE Ho\_NV **NOT IN** ('Sarah','Johnny','Dale');

*Câu truy vấn trên tương đương với:*

    SELECT \*

    FROM NHAN\_VIEN

    WHERE Ho\_NV **<>** 'Sarah' **AND** Ho\_NV **<>** 'Johnny' **AND** Ho\_NV **<>** 'Dale';

--**ANY**

|  |  |  |
| --- | --- | --- |
| **ĐIỀU KIỆN** | **MÔ TẢ** | **Ví dụ** |
| X **>** **ANY**(truy\_vấn\_con) | X phải **lớn hơn** ít nhất 1 giá trị trả về từ câu truy vấn con | *Điều kiện*: **X > ANY(1,2,0)**  *Kết quả*: **X > 0**  **~> Lấy giá trị nhỏ nhất** |
| X **<** **ANY**(truy\_vấn\_con) | X phải **nhỏ hơn** ít nhất 1 giá trị trả về từ câu truy vấn con | *Điều kiện*: **X < ANY(1,2,-1)**  *Kết quả*: **X < 2**  **~> Lấy giá trị lớn nhất** |
| X = **ANY**(truy\_vấn\_con) | X **IN** (1,2,5) | *Điều kiện*: **X = ANY(1,2,5)**  *Kết quả*: **X=1 or X=2 or X=5** |

**ví dụ:**

    SELECT \*

    FROM NHAN\_VIEN

    WHERE LUONG > **ANY** (SELECT LUONG

FROM NHAN\_VIEN

WHERE PHG LIKE 'PB002')

--**EXISTS / NOT EXISTS**

**ví dụ: MỆNH ĐỀ EXISTS - tồn tại**

    SELECT \*

    FROM NHAN\_VIEN

    WHERE **EXISTS** (SELECT \*

FROM PHONG\_BAN

WHERE NHAN\_VIEN.PHG = PHONG\_BAN.MA\_PB)

**ví dụ: MỆNH ĐỀ NOT EXISTS - không tồn tại**

    SELECT \*

    FROM NHAN\_VIEN

    WHERE **NOT** **EXISTS** (SELECT \*

    FROM PHONG\_BAN

    WHERE NHAN\_VIEN.PHG = PHONG\_BAN.MA\_PB)

--- **NỘI DUNG SLIDE 7 - PHẦN 1 (Ngôn ngữ thao tác dữ liệu)** ---

--Sao chép TOÀN BỘ thông tin (thuộc tính, ràng buộc, dữ liệu,...) từ một bảng

CREATE TABLE <tên\_bảng\_mới> LIKE <tên\_bảng\_cần\_sao\_chép>;

INSERT INTO <tên\_bảng\_mới> SELECT \* FROM <tên\_bảng\_cần\_sao\_chép>;

**ví dụ:**

    CREATE TABLE NHAN\_VIEN\_CAO\_CAP LIKE NHAN\_VIEN;

    INSERT INTO NHAN\_VIEN\_CAO\_CAP SELECT \* FROM NHAN\_VIEN;

--Sao chép thông tin (chỉ sao chép thuộc tính và dữ liệu) từ một bảng

CREATE TABLE <tên\_bảng\_mới> AS SELECT \* FROM <tên\_bảng\_cần\_sao\_chép>;

**ví dụ:**

    CREATE TABLE NHAN\_VIEN\_CAO\_CAP AS SELECT \* FROM NHAN\_VIEN;

--Thêm một hàng mới vào bảng KHÔNG LIỆT KÊ danh sách các cột (đầy đủ thông tin)

INSERT INTO <tên\_bảng>

VALUES (<giá\_trị\_1>, <giá\_trị\_2>, <giá\_trị\_3>,...)

Lưu ý: Giá trị các cột truyền vào phải theo thứ tự các cột trong bảng

**ví dụ:**

    INSERT INTO PHONG\_BAN

    VALUES ('PB007', 'Truyen Thong', null);

--Thêm một hàng mới vào bảng LIỆT KÊ danh sách các cột

INSERT INTO <tên\_bảng>(<tên\_cột\_1>, <tên\_cột\_2>, <tên\_cột\_3>,...)

VALUES (<giá\_trị\_cột\_1>, <giá\_trị\_cột\_2>, <giá\_trị\_cột\_3>,...)

**ví dụ:**

    INSERT INTO PHONG\_BAN(MA\_PB, TEN\_PB, MA\_TP)

    VALUES ('PB007', 'Truyen Thong', null);

--- **NỘI DUNG SLIDE 7 - PHẦN 2 (Ngôn ngữ thao tác dữ liệu)** ---

--**Cập nhật** dữ liệu ở trong bảng

UPDATE <tên\_bảng>

SET <tên\_cột\_1> = <biểu\_thức\_1> [, <tên\_cột\_2> = <biểu\_thức\_2>] . . .

[WHERE  <điều\_kiện>]

**ví dụ:**

    Thay đổi tên phòng ban có mã phòng ban PB001 thành Phòng phát triển công nghệ

**UPDATE** PHONG\_BAN

**SET** TEN\_PB = 'Phòng phát triển công nghệ'

    WHERE MA\_PB = 'PB001'

    Cập nhật lại lương tăng thêm 10% cho những nhân viên thuộc phòng ban Thiết kế

**UPDATE** NHAN\_VIEN

**SET** LUONG = LUONG + LUONG \* 0.1

    WHERE PHG = (SELECT MA\_PB FROM PHONG\_BAN WHERE TEN\_PB LIKE 'Thiết kế')

    Cập nhật lại lương cho các nhân viên có tham gia vào dự án có mã DA001 lên 50$

**UPDATE** nhan\_vien

**SET** LUONG = LUONG + 50

    WHERE ID\_NhanVien IN (SELECT ma\_nhanvien

                          FROM quanly\_duan

                          WHERE ma\_duan LIKE 'DA001')

    Cập nhật lại cột mã trưởng phòng cho Phòng ban có tên “Sản xuất” với giá trị

    mới là mã nhân viên có tên “Huỳnh Anh Tuấn”

**UPDATE** phong\_ban

**SET** ma\_truongphong = (SELECT id\_nhanvien

                          FROM nhan\_vien

                          WHERE ho\_nv = 'Huỳnh' AND ten\_nv = 'Anh Tuấn')

    WHERE tenn\_pb = 'Sản xuất'

--**Xóa** dữ liệu ở trong bảng

DELETE FROM <tên\_bảng>

[WHERE <điều\_kiện>]

**ví dụ:**

    Xóa toàn bộ dữ liệu có trong bảng phòng ban

**DELETE** FROM PHONG\_BAN

    Xóa phòng ban có mã PB007

**DELETE** FROM PHONG\_BAN

    WHERE MA\_PB LIKE 'PB007'

    Xoá các Phòng ban không có nhân viên nào

**DELETE** FROM PHONG\_BAN

    WHERE MA\_PB NOT IN (SELECT DISTINCT PHG FROM NHAN\_VIEN)