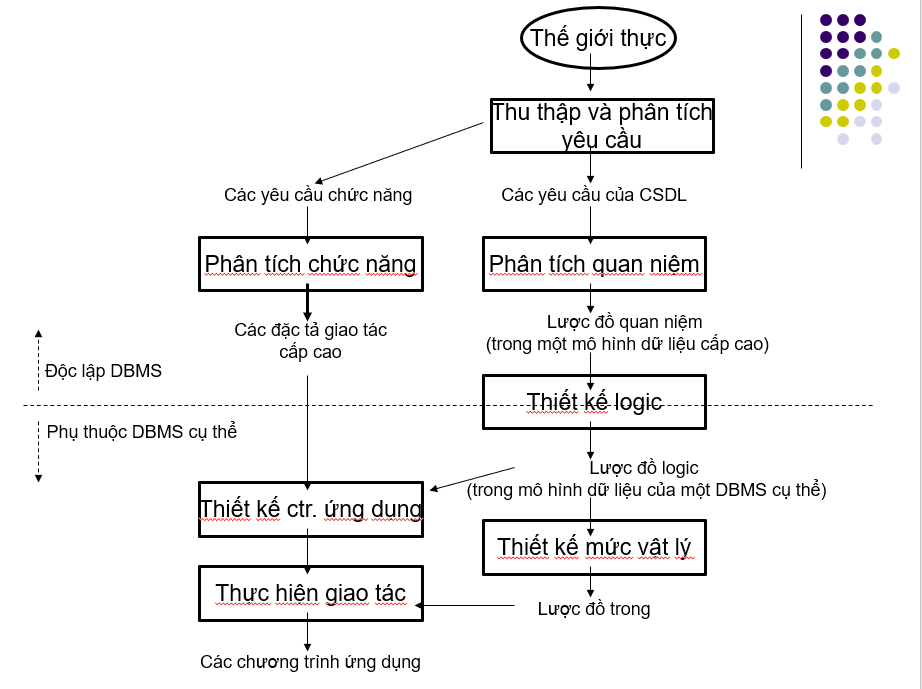
## Tổng hợp môn Cơ sở dữ liệu (Database)

# Quy trình thiết kế CSDL

1. Thu thập và phân tích các yêu cầu
2. Thiết kế mức quan niệm
3. Ánh xạ thiết kế logic/mô hình dữ liệu
4. Thiết kế mức vật lý



# Các khái niệm:

* Entity: “thing” or object: Xác định 1 bộ trong quan hệ
* Entity Set: collection of similar entities (similar to a class in OOP): Một tập hợp tất cả các bộ ở một thời điểm bất kỳ.
* Attribute: property of entity
  + Generally, all entities in a set have the same properties.
  + Attribute are simple values, e.g., integers or character strings…

# Các thuộc tính

* Simple
* Multi-valued
* Derived
* NULL

# Thuộc tính Khóa

* Các thực thể của 1 tập thực thể phải được phân biệt
* Tập thực thể bắt buộc phải có 1 giá trị duy nhất gọi là Thuộc tính Khóa – Key attribute – của tập thực thể đó để phân biệt các thực thể với nhau.
* Giá trị của thuộc tính Khóa:
  + Phải khác NULL
  + Xác định duy nhất cho từng thực thể
* Unique key:
  + Có thể = NULL
  + Xác định duy nhất cho từng thực thể
* Tập thực thể có thể có:
  + 1 Khóa chính
  + Nhiều hơn 1 Khóa
  + Không có thuộc tính Khóa => Loại thực thể yếu

# Mối kết hợp và loại mối kết hợp

* Relationship: liên hệ với 2 hay nhiều thực thể khác nhau mang một ý nghĩa nào đó
* Relationship type: các thực thể được nhóm lại hay phân loại thành một loại mối kết hợp
* Có nhiều hơn 1 loại mối kết hợp có thể có trong cùng loại thực thể:
  + Ex: QUANLY và PHANCONG là các mối kết hợp khác nhau giữa NHANVIEN và PHONGBAN
* Relationship degree: số lượng các loại thực thể tham gia vào mối kết hợp đó
  + Mối kết hợp cấp 2, 3,…, n
  + Cả 2 mối kết hợp QUANLY và PHANCONG là kết hợp cấp 2.

# Thuộc tính của Relationship type

* Một loại mối kết hợp có thể có các thuộc tính để mô tả tính chất của một mối kết hợp
* Các thuộc tính này không gắn liền với các thực thể tham gia, nó chỉ có ý nghĩa trong ngữ cảnh của một mối kết hợp

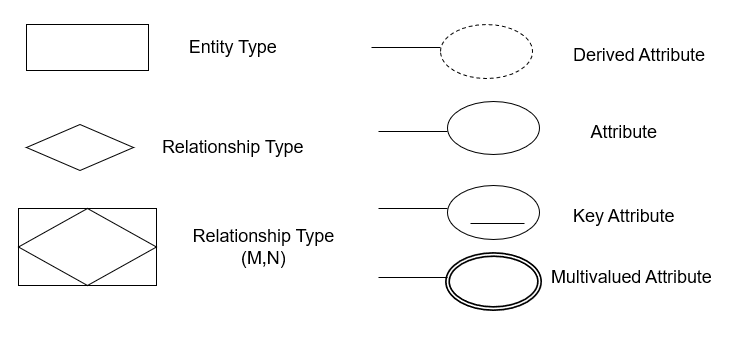
# Ràng buộc trên mối kết hợp cấp 2: Cardinality Ratio

* Many-to-Many
* One-to-Many
* One-to-One

# Các bước thiết kế ER

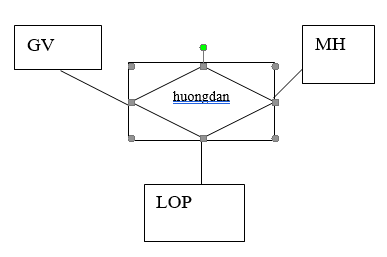
1. Xác định các loại thực thể, và thuộc tính Khóa
2. Xác định loại mối kết hợp
3. Xác định và gắn thuộc tính với loại thực thể và loại mối kết hợp
4. Quyết định miền giá trị của thuộc tính
5. Gắn (tinh chế) cardinality ratio(s) vào loại mối kết hợp
6. Thiết kế phân cấp chuyên biệt hóa/tổng quát hóa (specialization/generalization) trong các ràng buộc

# Các ký hiệu của mô hình ER

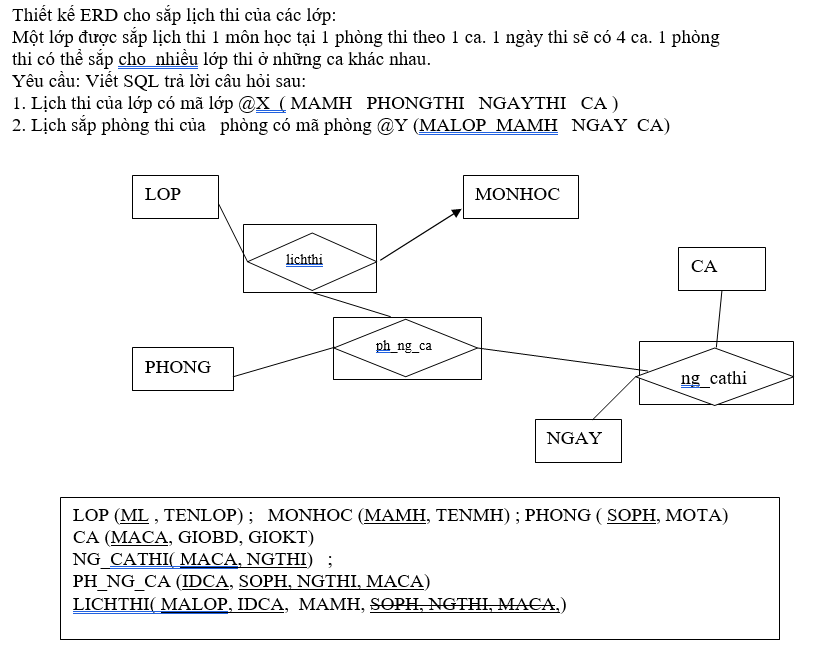


# Các trường hợp liên kết đặc biệt:

* Mối liên kết 3: có 2 trường hợp 1-N-N, N-N-N



* Liên kết giữa Thực thể với 1 Mối liên kết

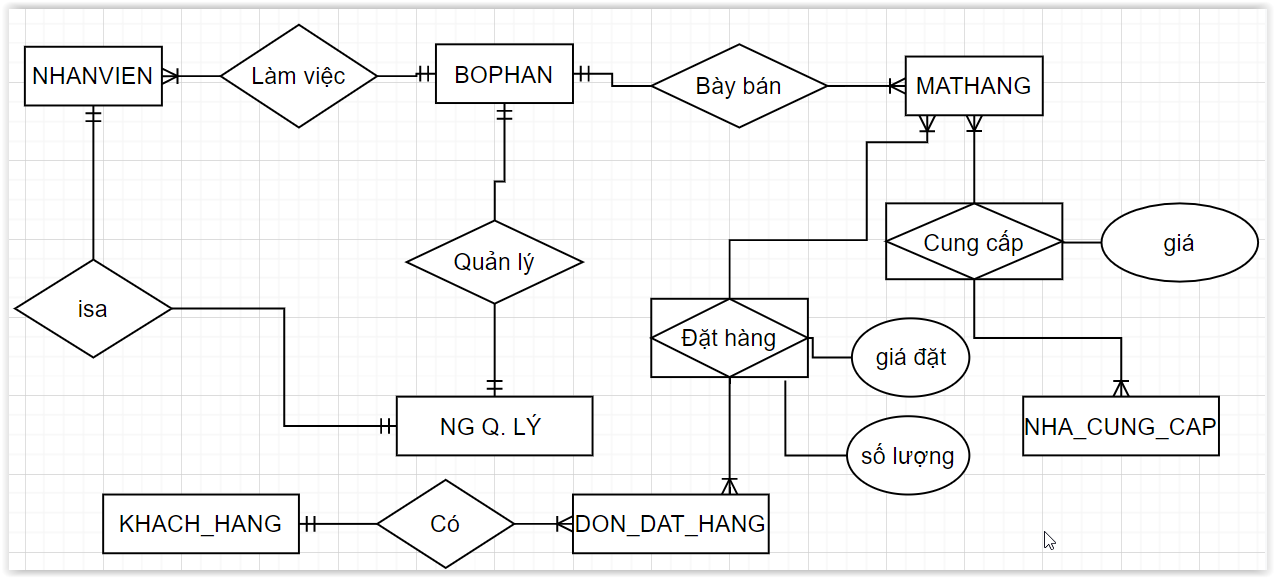


# Ví dụ về ERD của siêu thị

* Một siêu thị có nhiều nhân viên. Thông tin về nhân viên: MaNV, TênNV, Luong. Mỗi nhân viên làm việc ở 1 bộ phận, 1 bộ phận có thể có nhiều nhân viên. Thông tin về bộ phận: MaBP và tênBP. Trong các nhân viên có người làm quản lý, một người chỉ quản lý tối đa 1 bộ phận, và 1 bộ phận chỉ có 1 người quản lý.
* Ở 1 bộ phận có nhiều mặt hàng, nhưng 1 mặt hàng chỉ được đặt ở 1 bộ phận mà thôi. Thông tin về mặt hàng: MAMH, TenMH. 1 mặt hàng do nhiều người cung cấp bán, 1 người cung cấp có thể cung cấp nhiều mặt hàng, mỗi mặt hàng sẽ có giá khác nhau. Thông tin về người cung cấp: MaNCC,TenCC, Diachi.
* Siêu thị có nhiều khách hàng. Thông tin về khách hàng: MAKH, TenKH, Diachi. Mỗi khách hàng có nhiều đơn đặt hàng nhưng 1 đơn đặt hàng thì chỉ thuộc 1 khách hàng mà thôi. Thông tin về đơn đặt hàng: SoHieuDDH, NgayDH. Mỗi đơn đặt hàng bao gồm nhiều mặt hàng với mỗi mặt hàng có số lượng mua và đơn giá khác nhau.

**Xác định thực thể:**

* NHANVIEN (MANV, HO, TEN, LUONG)
* BOPHAN (MABP, TENBP)
* NGUOIQUANLY (MANQL)
* MATHANG (MAMH, TENMH)
* NGUOICC (MANCC, TENNCC, DIACHI)
* KHACHHANG (MAKH, TENKH, DIACHI)
* DONDATHANG (SOHIEUDDH, NGAYDH)



# Chuyển đổi ERD sang TABLE

* Mỗi thực thể sẽ được chuyển thành Quan hệ thực thể
* Mối liên kết 1-1: quan hệ này sẽ chứa khóa chính của quan hệ kia, và ngược lại
* Mối liên kết 1-N: quan hệ đầu nhiều sẽ chứa khóa chính của quan hệ đầu 1
* Mối liên kết N-N: sẽ có thêm quan hệ mới với các thuộc tính là khóa chính ở các thực thể, và các thuộc tính riêng của nó. Khóa chính của Quan hệ mới này sẽ là tổ hợp (khóa chính của các thực thể đầu nhiều), có thể có thêm thuộc tính riêng trong khóa chính (khi không thể xác định được thực thể thông qua khóa chính).
* Tổ hợp các quan hệ có cùng khóa chính

**Ex:**

* NHANVIEN (MANV, HO, TEN, LUONG, MABP)
* BOPHAN (MABP, TENBP, MANQL)
* MATHANG (MAMH, TENMH, MABP)
* NGUOICC (MANCC, TENNCC, DIACHI)
* KHACHHANG (MAKH, TENKH, DIACHI)
* DONDATHANG (SOHIEUDDH, NGAYDH, MAKH)
* CTDDH (MAMH, SOHIEUDDH, GIADAT, SOLUONG)
* CTCC (MANCC, MAMH, GIACC)

# Phụ thuộc hàm

* Định nghĩa: Cho R(U) là 1 lược đồ quan hệ với U = {A1, A2, …, An} là tập thuộc tính.
  + r là 1 thể hiện (record) của quan hệ R(U)
  + X, Y là tập con của U (attribute của U)

Ta nói X -> Y là phụ thuộc hàm trên R(U) nếu t1, t2 ∈ r (2 records) mà t1[X]=t2[X] thì t1[Y]=t2[Y].

* Ví dụ: Trong quan hệ SINHVIEN nếu X là MASV và Y là HOTEN thì X → Y là phụ thuộc hàm và Y → X không phải phụ thuộc hàm vì trùng họ tên nhưng chưa chắc đã xác định trùng mã sinh viên.

# Phụ thuộc hàm đầy đủ

* Định nghĩa: Ta nói X->Y là PTH đầy đủ trên R(U) nếu không tồn tại X’⊂ X sao cho X’->Y (tức là X phải là bé nhất)
* Mệnh đề:
  + X->Y là phụ thuộc hàm trên r nếu và chỉ nếu X là siêu khóa của r[XY]
  + X->Y là phụ thuộc hàm đầy đủ trên r nếu và chỉ nếu X là khóa (siêu khóa bé nhất) của r[XY]

# Hệ luật dẫn ARMSTRONG

* Cho R(U) là 1 lược đồ quan hệ với U = {A1, A2, …, An} là tập thuộc tính.
* Xét X, Y, Z, W là tập con của U.
  + Luật phản xạ: Y ⊆ X => X-> Y
  + Luật thêm vào: X->Y và Z ⊆ W (W->Z) => XW->YZ   
    X->Y thì XZ -> YZ (tăng trưởng) (hay dùng) (1 chiều)
  + Luật bắc cầu: X->Y và Y->Z => X->Z
  + Luật tựa bắc cầu: X->Y và WY->Z => XW->Z (dùng luật tăng trưởng c/m)
  + Luật phân rã: X->Y và Z ⊆ Y => X->Z
  + Luật hợp: X->Y và X->Z => X->YZ
* Ex: Cho quan hệ R (A, B, C) và tập phụ thuộc hàm F = {AB->C, C->A}. Dựa vào hệ luật dẫn ArmStrong, hãy chứng minh: BC->ABC.

Có 2 cách tiếp cận: 1 là đi ngược lại dò trong đề bài xem có cái nào có A hay không? Nếu có A thì phải có C muốn có C thì cần có gì? Và đi ngược lại, 2 là cố gắng từ đề bài làm sao xuất hiện vế trái trước rồi tính tiếp.

* + Ta có: C->A (giả thiết)
  + BC->BA (Luật tăng trưởng)
  + BC->ABC (Luật tăng trưởng)
* Ex: Cho quan hệ R (A, B, C, D, E, G, F) và tập phụ thuộc hàm F = {B->CD, E->F, D->E, B->A, AD->B, F->G}. Chứng minh: B->G.
  + Ta có: B->CD (giả thiết)
  + B->D (Luật phân rã)
  + Và D->E (giả thiết)
  + B->E (Luật bắc cầu)
  + Và E->F (giả thiết)
  + B->F (Luật bắt cầu)
  + Và F->G (giả thiết)
  + B->G (Luật bắt cầu)
* Ex: Cho lược đồ quan hệ R (A, B, C, D, E, G, H, K) và tập phụ thuộc hàm

F = {C → AD; E→ BH; B→ K; CE→ G}. Kiểm tra xem các phụ thuộc hàm E->K (được); E->G (không) có được suy từ F hay không?

* Ta có: E->BH (giả thiết)
* E->B (Luật phân rã)
* Và B->K (giả thiết)
* E->K (Luật bắc cầu) (đpcm)

# Bao đóng: (Bao đóng của tập phụ thuộc hàm F và bao đóng của tập thuộc tính)

**Định nghĩa Phụ thuộc hàm suy dẫn:** Cho tập các PTH F. X, Y **⊆** U. Ta nói:

f: X->Y là PTH suy dẫn từ F nếu nó được suy từ F bằng các luật dẫn của hệ tiên đề Armstrong.

* Kí hiệu: F |= f
* Tập F+ = {f/ F |= f} được gọi là bao đóng của F

**Mệnh đề:** Cho tập các PTH F, ta có:

1. F ⊆ F+
2. f là hệ quả của F ⬄ F |= f ⬄ f ∈ F+

**Bao đóng của tập thuộc tính:**

Cho lược đồ quan hệ Q với tập PTH F ={f1, f2 , …., fn }. Giả sử X ⊆ Q+

Bao đóng của X dựa trên F:XF+={ Y ∈ Q+ / X-> Y ∈ F+}

**Nhận xét:**

* X ⊆ X+
* f: X->Y ∈ F+ ⬄ Y ⊆ XF+

Ex: Cho lược đồ quan hệ Q (A, B, C, D, E, G) với: F = {AE-> C, CG -> A , BD -> G, GA->E}. Chứng minh BCD -> Q+ ∈ F+

* BD->G (giả thiết)
* BCD->CG (tăng trưởng)
* BCD->G (phân rã) (1)
* CG->A (giả thiết)
* BCD->CG (đã chứng minh)
* BCD->A (bắc cầu) (2)
* (1) và (2): BCD->GA (hợp) (3)
* GA->E (giả thiết)
* BCD->E (bắc cầu) (4)
* (3) và (4): BCD->GAE (hợp)
* BDC->ABCDEG (tăng trưởng)
* Vậy BDC->Q+

Ex: Q là tập các thuộc tính với Q = {A, B, C, D, E, G} và tập F các phụ thuộc hàm với F={AG→E; A→B; C→A; C→D; AG→C}. Xác định phụ thuộc hàm CG->BE ∈ F+ ?

* AG->C (giả thiết)
* C->A (giả thiết)
* AG->A (bắt cầu)
* A->B (giả thiết)
* AG->B (bắc cầu)
* AG->E (giả thiết)
* AG->BE (hợp) (1)
* C->A (giả thiết)
* CG->AG (tăng trưởng) (2)
* CG->BE (bắc cầu)

# Khóa của quan hệ. Xác định khóa

* + Tìm các nút gốc X = U – {vế phải}
  + Tính bao đóng của các nút gốc X
* Nếu XF+ = U thì X là khóa
* Ngược lại X = X U Y với Y **∈** U
* Quay lại bước 2

Ex: Cho Q (A B C D E H) và F ={AB->C, CD -> E, EC->A, CD-> H , H->B}. Tìm 1 khóa của Q.

* Nút gốc = {A B C D E H} – {C E A H B} = {D}

Gọi X = {D}

* Vì XF+ = {D} nên ta chọn X = X U {C} = {CD}

Vì XF+ = {CD}F+ = {C D E H B A} = Q+ nên {CD} là khóa.

# Xác định tất cả khóa chính

Nhận xét: Nếu K là khóa của Q thì K chứa tất cả các thuộc tính nguồn và không chứa bất kỳ thuộc tính đích nào.

B1: Tập gốc N = U – {vế phải}

Tập trung gian L = U – N

Xây dựng 2k tập con của L: L1, L2,…,Lk bằng Phương pháp đường chạy nhị phân.

B2: Xây dựng tập K chứa siêu khóa.

K := i

For I = 1 to 2k do.

Begin

Xi := N U Li.

Tính bao đóng (Xi)+F

If (Xi)+F = Q+ theo K = K U Xi

End

B3: Loại bỏ dần các siêu khóa lớn trong tập hợp các siêu khóa K = {K1, K2, …, K3}

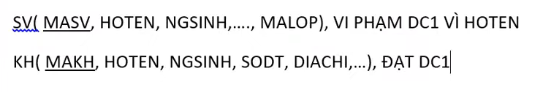
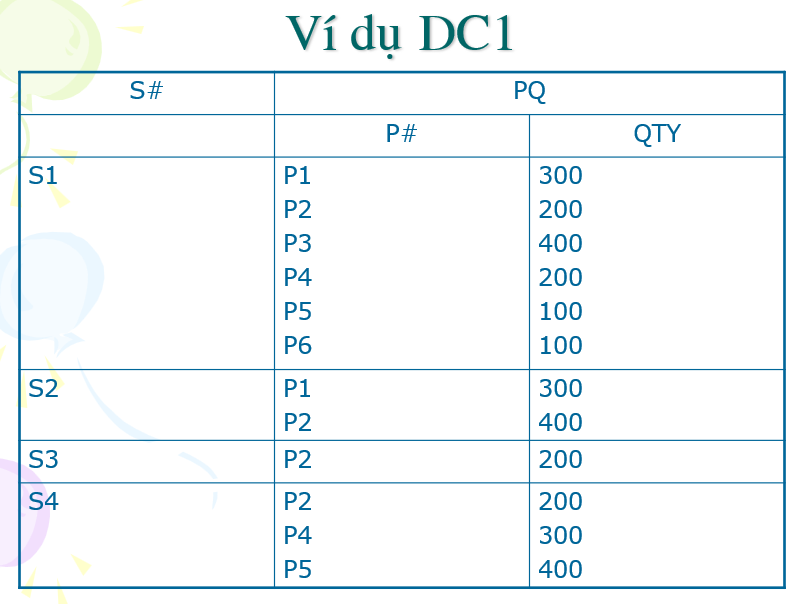
# Xác định dạng chuẩn CSDL

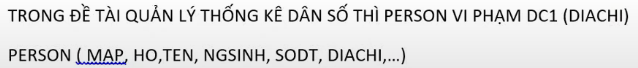
Mục đích: Thiết kế CSDL sao cho đạt tối thiểu DC3 để đảm bảo không dư thừa dữ liệu

**Dạng chuẩn 1: 1NF (Normal Form)**

Một quan hệ R được xem ở dạng chuẩn 1 nếu và chỉ nếu tất cả các thuộc tính trong R đều chứa các giá trị nguyên tố (giá trị không tách ra được nữa).

Ví dụ vi phạm chuẩn 1

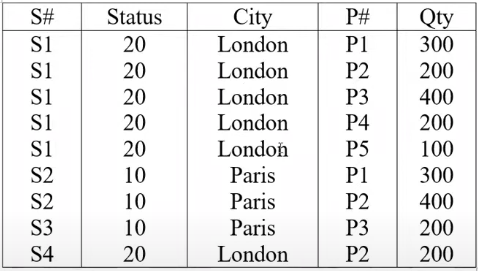
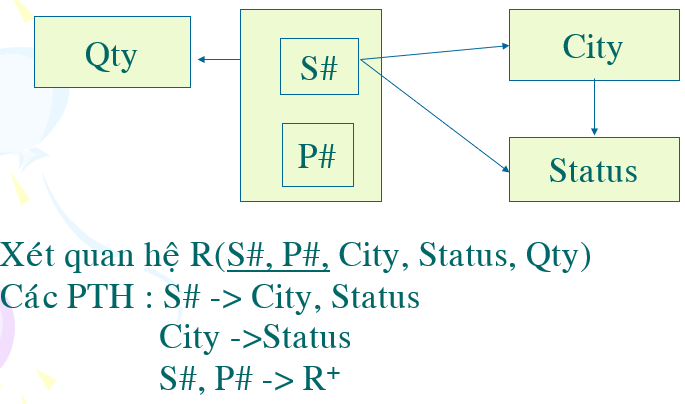
=> Sinh viên khi sắp theo tên nếu trùng tên sắp theo họ. Nếu gom chung họ tên thì sẽ bị sắp theo họ không theo tên.

 => Địa chỉ bao gồm cả xã quận huyện thì sao mà thống kê theo từng xã, huyện, tỉnh được?

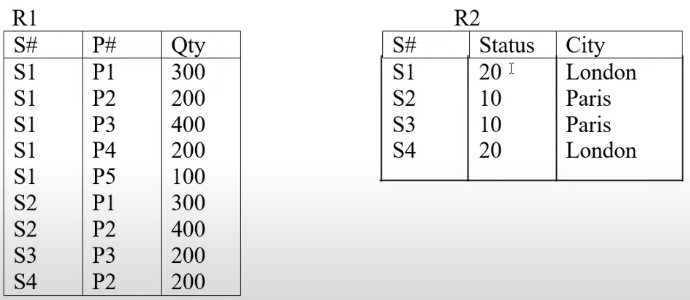
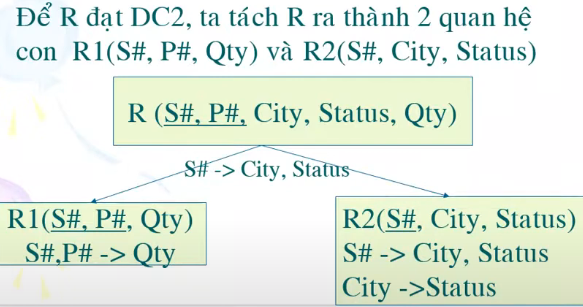
**Dạng chuẩn 2:**

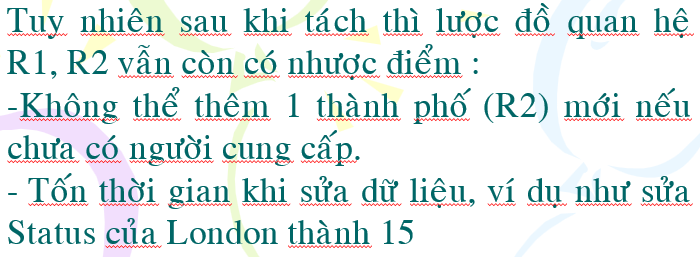
Một quan hệ R được xem ở dạng chuẩn 2 nếu và chỉ nếu R đạt DC1 và các thuộc tính không khóa phụ thuộc hàm đầy đủ vào khóa chính.

Vi phạm chuẩn 2



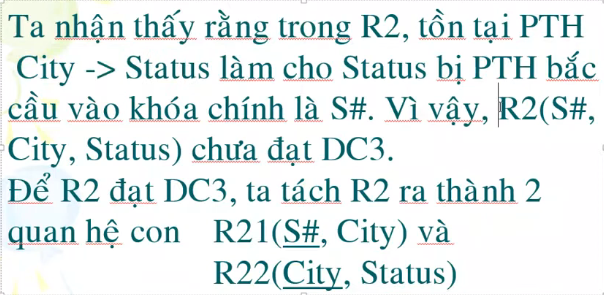
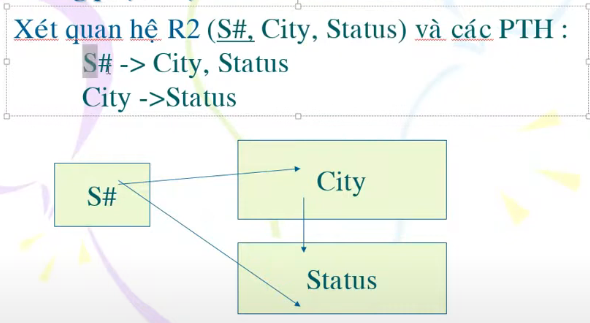
Chuẩn hóa:

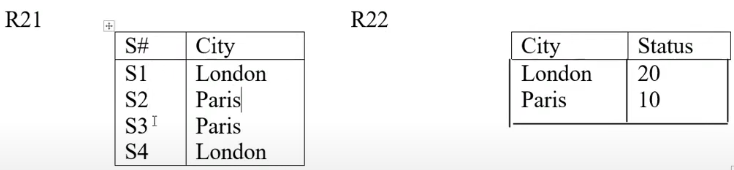




**Dạng chuẩn 3:**

Một quan hệ R được xem ở dạng chuẩn 3 nếu và chỉ nếu R đạt DC2 và các thuộc tính không khóa không phụ thuộc hàm bắc cầu vào khóa chính.

****



**Trick:** Nếu khóa chính chỉ có 1 thuộc tính => Đạt chuẩn 2

Nếu thuộc tính không khóa chỉ còn 1 => Đạt chuẩn 3.

Nếu tất cả các quan hệ của lược đồ cơ sở dữ liệu đạt DCi thì t nói lược đồ cơ sở dữ liệu đạt DCi.

# Ngôn ngữ đại số

**Các phép toán quan hệ:**

* Phép chọn (Selection Operation): Dùng để trích chọn các bộ từ quan hệ r. Các bộ được trích chọn phải thỏa mã điều kiện P.
* Ký hiệu: **σP(r) = {t | t ∈ r ∧ P(t)}**
* Op: =, !=, <=, >=, <, >, …
* Các đk có thể liên kết với nhau thông qua ∧ (and)2, ∨ (or)3, ¬ (not)1
* Phép chọn có tính chất giao hoán: σP1(σP2(*r*)) = σP2(σP1(*r*)) = σP1 ∧ P2(*r*)
* Phép chiếu (Project Operation): Dùng để trích chọn 1 vài cột của quan hệ r được chỉ ra trong danh sách thuộc tính.
  + Ký hiệu: πA1,A2, …,Ak(*r*)
  + Số lượng các bộ trong một quan hệ kết quả của phép chiếu luôn luôn ít hơn hoặc bằng số lượng các bộ trong *r*. Các dòng trùng nhau sẽ được loại bỏ khỏi quan hệ kết quả.
  + Phép chiếu không có tính chất giao hoán.
* Phép gán (Assignment Operation)
  + Cung cấp 1 cách thức để diễn tả câu truy vấn phức tạp
  + Ký hiệu: ←
* Phép đổi tên
  + Cho phép đặt tên, và do đó để tham chiếu tới kết quả của biểu thức đại số quan hệ
  + Cho một biểu thức đại số quan hệ *E* có n thuộc tính, biểu thức ρχ(A1, A2, …, An)(*E*) trả về kết quả biểu thức *E* dưới tên χ, và với các tên thuộc tính đã được thay đổi thành A1, A2, …, An.
* Các phép toán được hình thành từ lý thuyết tập hợp toán học:
  + Phép hội (Union Operation): r ∪ s (Dùng Union all để lấy tất cả dòng kể cả trùng)
  + Phép giao (Set-Intersection Operation): r ∩ s
  + Phép trừ (Set Difference Operation): r – s

SQL: Not in, Not exist, Outer join left

* + Phép tích (Cartesian Product Operation): r×s
    - KQ(A1, A2, …, Anr, B1, B2, …, Bms) ← r(A1, A2, …, Anr) × s(B1, B2, …, Bms)
    - Nếu *r* có nr bộ và *s* có ms bộ thì r × s có nr × ms bộ.
* Phép kết
* Phép chiếu tổng quát
* Phép chia (Division Operator):
  + Áp dụng cho câu truy vấn có từ “tất cả”
  + Mục tiêu: Đưa ra các bộ quan hệ r, sao cho khớp với tất cả cá bộ trong quan hệ khác, s.
  + Ký hiệu: r ÷ s
* Các phép toán quan hệ khác: OUTER JOIN, AGGREGATE FUNCTION (sum, max, min, average, count) và GROUPING
* Ex: Tìm họ tên và lương nhân viên làm việc ở phòng số 4

πHOTEN, LUONG(σPHONG=4(NHANVIEN))

# SQL PRIMER: Data definition language (DDL)

1. **Tạo CSDL:**

Cú pháp: Create database <tenCSDL>

Trên dĩa sẽ có 2 files: .mdf và .ldf

Ngoài ra, ta có thể chỉ định các tham số cần thiết cho việc tạo cơ sở dữ liệu, chẳng hạn như:

**CREATE DATABASE QLVT**

On Primary

(Name = Qlvt1,

Filename = ‘c:\ruffa\qlvt1.mdf’,

Size = 10MB,

MaxSize = 100MB,

FileGrowth = 10MB)

Log On

(Name = Qlvt1Log,

Filename = ‘c:\ruffa\qlvt1\_log.ldf’,

Size = 10MB,

MaxSize = 100MB,

FileGrowth = 10MB)

Các giá trị mặc định:

* + Size: Dựa vào size của file model.mdf
  + MaxSize: không giới hạn
  + FileGrowth: 10%

Sau khi tạo xong 1 csdl, ta có thể:

* + Thêm 1 file vào csdl đã được tạo:

**Alter Database QLVT Add File**

(Name = Qlvtt2,

Filename = ‘c:\ruffa\qlvt2**.Ndf**’, (từ file thứ 2 là .ndf)

Size = 10MB,

MaxSize = 100MB,

FileGrowth = 10MB)

* + Xóa 1 file khởi cơ sở dữ liệu QLVT:

**Alter database QLVT**

Remove File Qlvt2

* + Hiệu chỉnh lại các thùy chọn đã định:

**Alter database QLVT**

ModifyFile

(Name = Qlvt1,

FileGrowth = 5MB)

* + Xóa toàn bộ cơ sở dữ liệu: **Drop Database QLVT**

1. **Tạo TABLE:**

* Ví dụ:

**USE QLVT**

**Create table QLVT.dbo.VatTu**

(MAVT nChar(4) NOT NULL,

TENVT nVarChar(30) unique NOT NULL,

DVT nVarChar(15) Default ‘ ‘

**Constraint** PK\_VatTu **Primary Key** (MAVT))

**Create table NhanVien**

(MANV int **Primary key**,

HO nvarchar(40) NOT NULL,

TEN nvarchar(10) NOT NULL,

PHAI nvarchar(3) Default ‘NAM’ Check ( PHAI = ‘NAM’ or Phai = N’NỮ’),

NGAYSINH Date,

LUONG Money Default 5000000 CHECK (LUONG Between 5000000 AND 50000000),

GHICHU nTEXT)

* Trong lệnh Create Table, ta có thể:

1. **Tạo ràng buộc về khóa ngoại và field tự động tính.**

Ví dụ: Xem ví dụ tạo table CT\_PhatSinh, trong đó có thêm field

TRIGIA = SOLUONG \* DONGIA:

Create table QLVT.dbo.CT\_PhatSinh

(PHIEU nchar(8) NOT NULL,

MAVT nChar(4) NOT NULL,

MAKHO nChar(2) NOT NULL,

SOLUONG int NOT NULL CHECK (SOLUONG > 0),

DONGIA Money NOT NULL CHECK (DONGIA > 0),

LYDO nvarchar(30)

Constraint PK\_CT\_PhatSinh Primary key (PHIEU, MAVT),

Constraint FK\_CT\_PhatSinh\_Kho Foreign Key (MAKHO)

References dbo.KHO (MAKHO)

ON DELETE CASCADE ON UPDATE CASCADE,

…)

1. **Tạo Unique Key:**

**Constraint UK\_TENKHO UNIQUE (TENKHO)**

1. **Kiểm tra dữ liệu ở mức record:**

**Constraint <tênRB> check (đk)**

<đk> trong Constraint Check chỉ được thực hiện khi nào ta ghi dữ liệu đang nhập vào table.

1. **Bỏ qua việc kiểm tra các ràng buộc trong nhân bản dữ liệu:**

**Constraint <tênRB> Check not for replication (đk)**

Sau khi tạo xong Table, ta có thể:

* + **Thêm hay xóa 1 ràng buộc:**

**Alter table QLVT.dbo.CT\_PhatSinh**

**Drop Constraint FK\_CT\_PhatSinh\_KHO**

* + **Thêm ràng buộc về khóa ngoại trên field MAKHO của tabke CT\_PhatSinh**

**Alter table QLVT.dbo.CT\_PhatSinh**

**Add Constraint FK\_CTPS\_KHO foreign key (MAKHO)**

**Reference dbo.KHO(MAKHO)**

* + **Lưu ý:** Để kiểm tra tất cả các records đã có trong table phải thỏa 1 ràng buộc nào đó thì ta thêm vào tùy chọn **WITH CHECK**

**Alter table QLVT.dbo.CT\_PhatSinh**

**With check**

**Check constraint FK\_CTPS\_KHO**

* + **Thêm, Xóa, thay đổi field trong Table:**
    - **Thêm 1 field:**

**Alter table QLVT.dbo.VATTU**

**ADD DONGIA int Check (Dongia >0)**

* + - **Thay đổi kiểu của field**

**Alter table QLVT.dbo.VATTU**

**Alter column TENVT varchar(40) NOT NULL**

* + - **Xóa 1 field:**

**Alter table QLVT.dbo.VATTU**

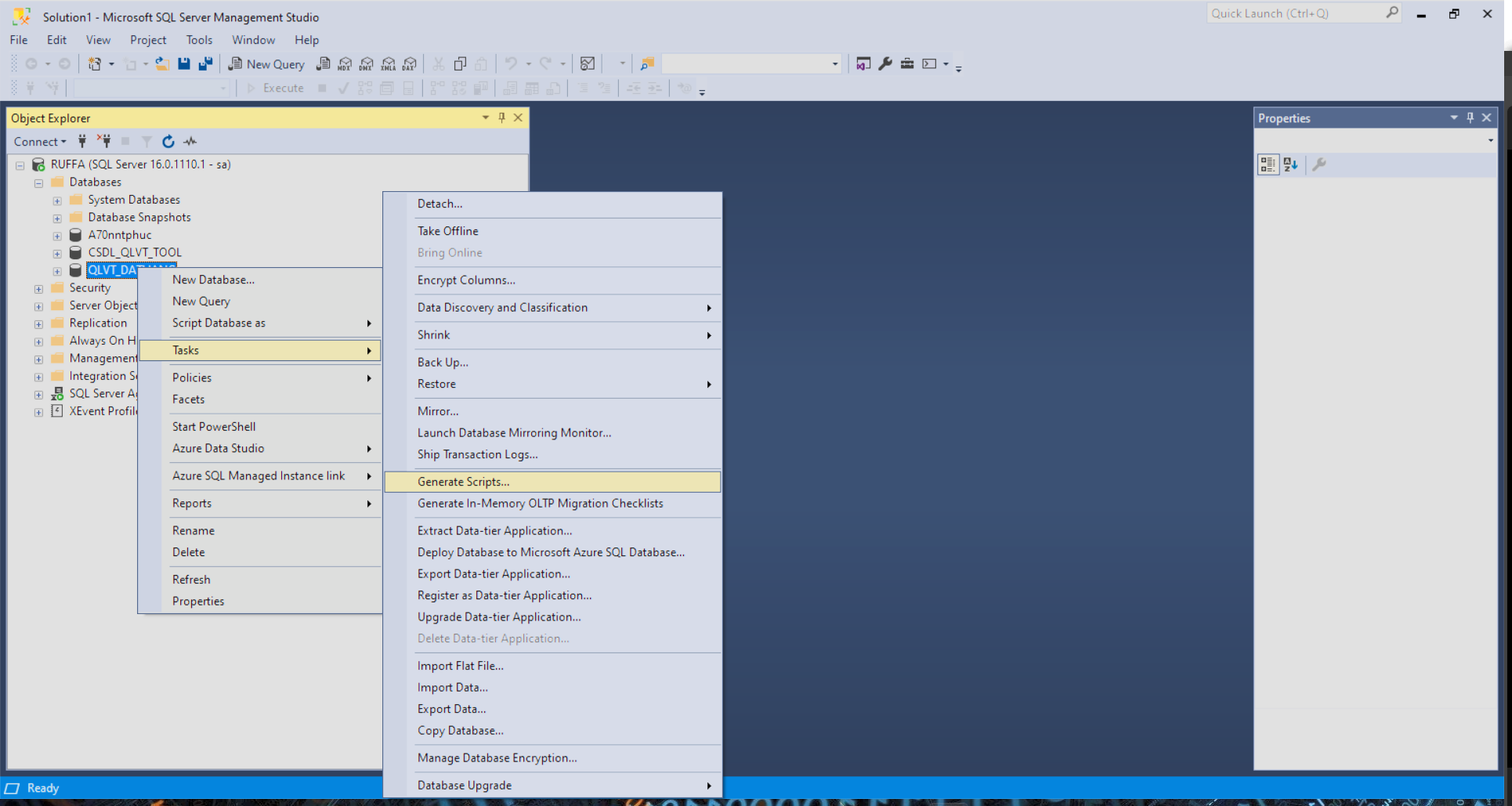
**Drop column DONGIA**

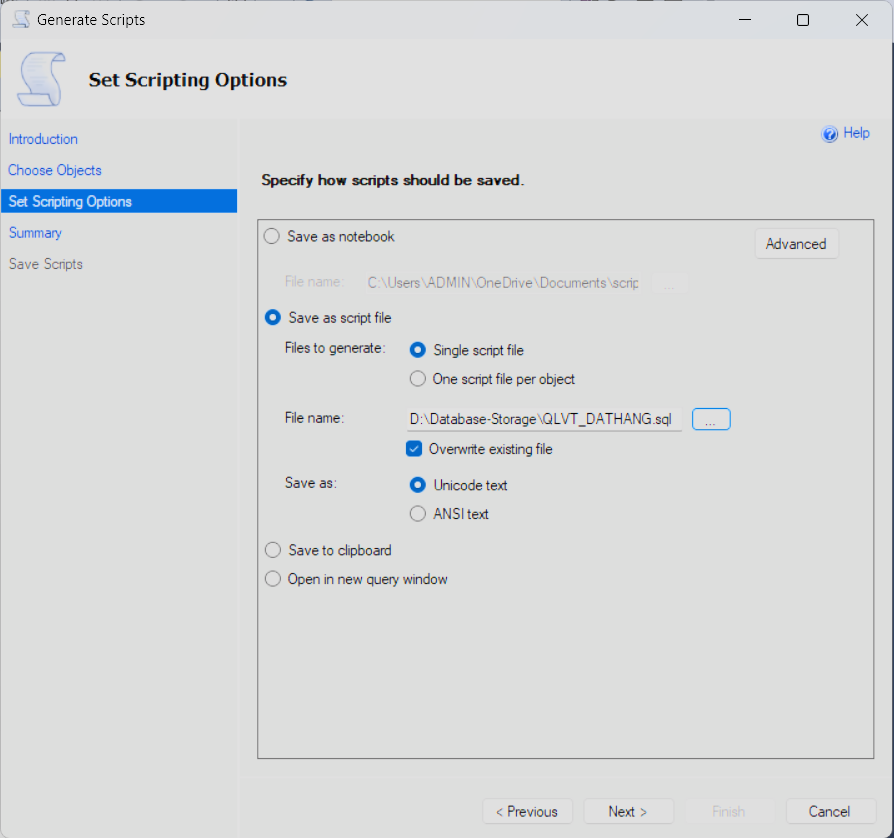
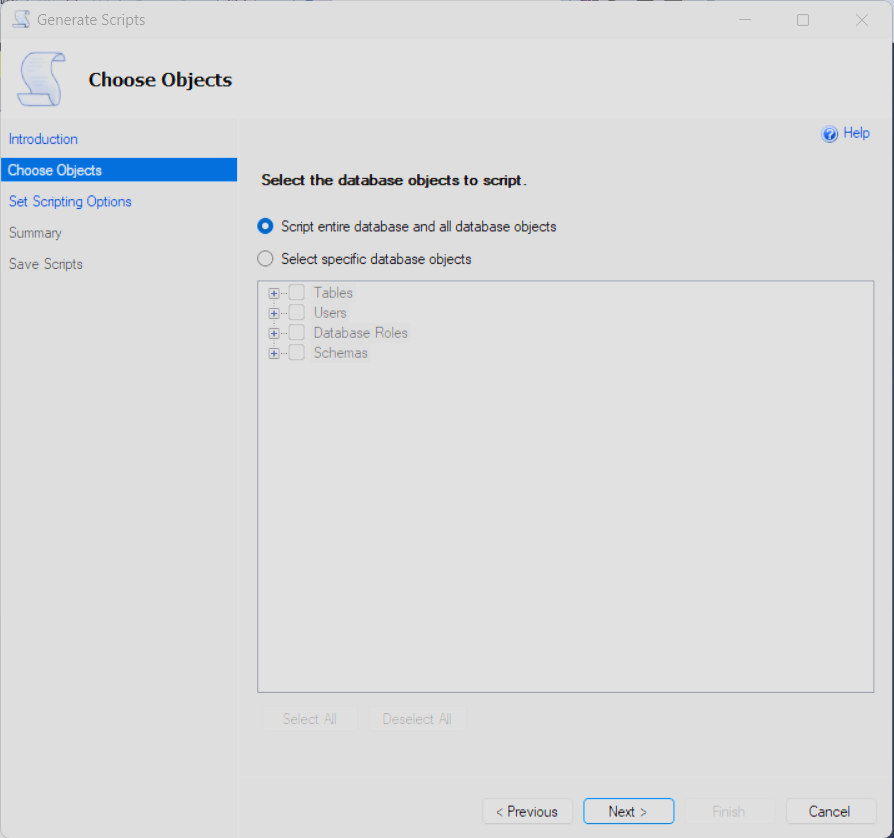
* + **Xóa Table:**

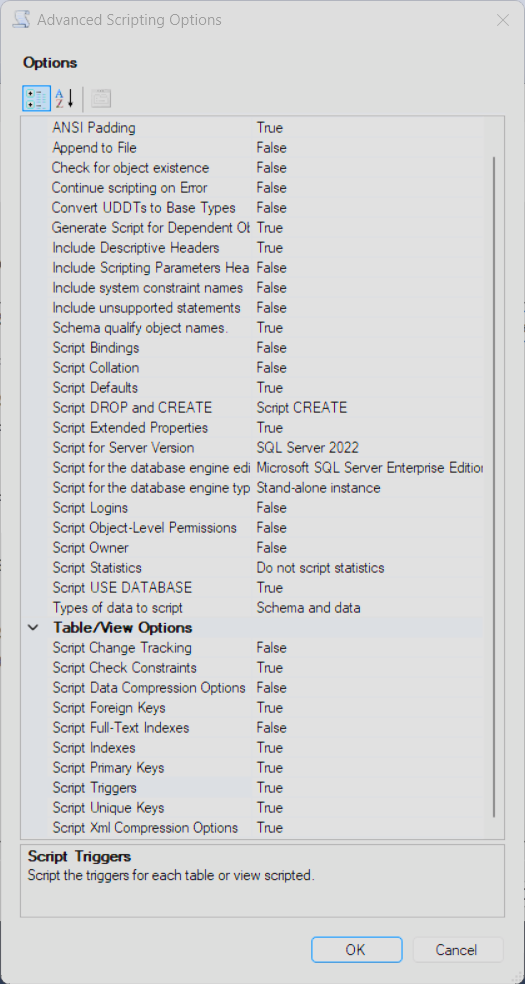
**Drop table QLVT.dbo.VATTU (xóa tất cả)**

**Delete table QLVT.dbo.VATTU (xóa dữ liệu)**

# Cách lưu cấu trúc và dữ liệu thành 1 file script (nên tạo database bằng giao diện sau đó lưu thành script)

****

****

****

# SQL Primer: Data manipulation language (DML)

* **Ngôn ngữ thao tác dữ liệu có các lệnh chính:**
  + Select
  + Insert
  + Update
  + Delete
  + Merge
* **Kiểu dữ liệu (data type):**
  + Nchar(…) lưu dữ liệu ít biến động như mã sinh viên,…
  + Nvarchar(…) lưu dữ liệu biến động
  + nText lưu trữ nhiều đoạn văn dài, 1 paragraph, … chiều dài tối đa 1GB
  + double
  + bit
  + image (không nên dùng mà nên dùng nvarchar)
  + tinyInt 1 byte
  + SmallInt 2 bytes
  + Int 4 bytes
  + BigInt 8bytes
  + Money
  + Date
  + Time
* **Ràng buộc (Constraint):** Theo quan niệm của thầy, có 2 loại ràng buộc:
  + Ràng buộc cứng: định nghĩa được trong lòng database, khi nhập liệu từ bất cứ cách nào thì ràng buộc cứng đều hoạt động.
    - Primary key
    - Foreign key
    - Unique key
    - Miền giá trị
    - Not NULL
  + Ràng buộc mềm: không thể định nghĩa trong lòng database => phải viết code kiểm tra (ví dụ: số lượng xuất hàng phải <= số lượng tồn) có thể dùng trigger để bắt trên database.
* **LỆNH SELECT:** Chọn ra các mẫu tin từ 1 hay nhiều table.

Cú pháp:

Select [Distinct] [Top n/ Top n percent]

FROM Danh sách\_cột

[INTO # | ## <table ảo]

From danhsách\_nguồndữliệu

[Where điều kiện]

[Group by cột\_nhóm [Having điều kiện]] (những field nào không nằm trong hàm thống kê phải group by).

[Order By <cột> [DESC] [, <cột> [DESC]]…]

[Compute AggregateFunction (cột)]

* **Các toán tử dùng trong điều kiện:**

> >= < <= = <>(!=+

Is NULL : WHERE Ghichu IS NULL

IS NOT NULL : WHERE Ghichu IS NOT NULL  
Between … And … : WHERE LUONG BETWEEN 5000 AND 500000

IN (danh sách giá trị) : WHERE Loai IN (‘N ’, ‘X’)

LIKE: \_ đại diện 1 ký tự : WHERE SDT LIKE ‘091%5’

% đại diện 1 string

Not And Or: thứ tự ưu tiên: Not, kế tiếp And, cuối cùng là OR

* Một cột trong lệnh Select có thể là: 1 field trong table, 1 hằng, 1 biểu thức, hay 1 hàm aggregate function (Count, Sum, Avg, Max, Min), Select-Stmt
* Một table trong danhsách\_table của lệnh Select có thể là 1: table hệ thống, user table, table ảo, view, UDF, Select-Stmt.
* **SubQuery:** Để thi hành 1 SubQuery ta dùng các toán tử so sánh, và một số toán tử sau: Exists, Not Exists, all, any, in.
* **Biểu thức Case:** SQL cung cấp cấu trúc Case để thay thế 1 trị có sẵn trong csdl bằng 1 biểu thức khác. Cấu trúc Case có dạng sau:

**Case**

**When đk1 then Expr1** (đặt đk có xác suất đúng cao I trước để không phải chạy nhiều case).

**When đk2 then Expr2**

**…**

**When đkn then Exprn**

**Else Exprx**

**End**

* **Dùng DML để hiệu chỉnh dữ liệu:**
  + **Lệnh INSERT:** Thêm 1 hoặc nhiều record mới vào table

**Insert into <table> (danh sách field)**

**Values (danh sách các giá trị)**

Lệnh Insert còn cho phép lấy dữ liệu từ các table khác chuyển vào qua cú pháp:

**Insert into <table> (ds field)**

**Select <ds cột> from <table\_x>**

* + **Lệnh update:** để thay đổi giá trị của 1 hay nhiều cột trong table thỏa điều kiện

**Update NhanVien**

**Set Name = N’Nguyễn Ngọc Thiên Phúc’**

**Where MaNV = 16**

Lưu ý: Không được update cùng 1 lúc các field có liên quan với nhau.

**Update CTPX**

**Set Dongia = dongia \* 1.1**

**Trigia = dongia \* soluong \* 1.1**

**Where mapx = ‘px01’**

* + **Lệnh Delete:** Xóa các mẫu tin thỏa điều kiện

**Delete from NhanVien**

**Where MaNV = 2**

Để xóa toàn bộ dữ liệu trong table:

**Delete from <table>**

**Hay**

**Truncate Table <table>**

Muốn xóa hẳn dữ liệu lẫn cáu trúc của Table:

**Drop Table <table>**

* + **Lệnh Merge:** Cho phép chèn hoặc cập nhật dữ liệu vào table Target từ dữ liệu trong source
    - Ví dụ sau đây sẽ chèn vật tư ‘Đường Biên Hòa’ mã ‘VT20’ DVT là ‘Kg’ (Source) table VATTU (target) nếu mã vật tư ‘VT20’ chưa có trong table VATTU; ngược lại lệnh merge sẽ Update SOLUONGTON = 20
    - MERGE INTO dbo.VATTU AS Target
    - USING (SELECT MAVT='VT20', TENVT=N'ĐƯỜNG BIÊN HÒA', DVT=N'KG', SLT= 0 ) AS Source
    - ON Target.MAVT= Source.MAVT
    - WHEN MATCHED THEN
    - UPDATE SET TARGET.SOLUONGTON =20
    - WHEN NOT MATCHED THEN
    - INSERT (MAVT, TENVT, DVT, SOLUONGTON)
    - VALUES (Source.MAVT, Source.TENVT, Source.DVT, Source.SLT) ;
    - ALTER PROC GHISV
    - @MASV NVARCHAR( 12), @HO NVARCHAR( 40), @TEN NVARCHAR( 12), @MALOP NVARCHAR(10), @PHAI NVARCHAR(6),@NGAYSINH NVARCHAR(12) , @NOISINH NVARCHAR(40), @DIACHI NVARCHAR( 100)
    - AS
    - BEGIN
    - SET DATEFORMAT DMY
    - MERGE INTO dbo.SINHVIEN AS Target
    - USING (SELECT MASV=@MASV , HO=@HO , TEN=@TEN , MALOP=@MALOP , PHAI=@PHAI,
    - NGAYSINH=@NGAYSINH , NOISINH=@NOISINH , DIACHI=@DIACHI ) AS Source
    - ON Target.MASV= Source.MASV
    - WHEN MATCHED THEN
    - UPDATE SET TARGET.HO=@HO ,
    - TARGET.TEN=@TEN ,
    - TARGET.MALOP =@MALOP ,
    - TARGET.PHAI=@PHAI ,
    - TARGET.NGAYSINH=@NGAYSINH ,
    - TARGET.NOISINH=@NOISINH ,
    - TARGET.DIACHI=@DIACHI
    - WHEN NOT MATCHED THEN
    - INSERT (MASV , HO , TEN , MALOP , PHAI , NGAYSINH , NOISINH , DIACHI)
    - VALUES (Source.MASV, Source.HO, Source.TEN, Source.MALOP,Source.PHAI, Source.NGAYSINH , Source.NOISINH, Source.DIACHI ) ;
    - END
    - **Lưu ý :** Nếu source là 1 table thì câu lệnh lấy từng mẫu tin trong Source để Insert/Update vào Target.
  + **@@Rowcount:** trả về những dòng đã chịu sự tác động của DML gần nhất
* **Các loại Join trong SQL Server:**
  + **Phải đặt 2 câu hỏi: Kết nối trong hay ngoài? Cú pháp ntn? Các xử lý của hệ thống trên từng loại kết nối?**
  + **INNER JOIN (chiếm hơn 90% các dự án):** Trả về kết quả là các bản ghi mà trường được join ở 2 bảng là khớp nhau, các bản ghi chỉ xuất hiện ở 1 trong 2 bảng sẽ bị loại.



* + **OUTER JOIN:** nới lỏng hơn, lấy về các bản ghi có mặt trong cả 2 bảng và cả các bản ghi chỉ xuất hiện ở 1 trong 2 bảng. Kiểu Join này được chia làm 2 loại:
    - FULL OUTER JOIN: kết quả gồm tất cả các bản ghi của cả hai bảng. Với các bản ghi chỉ xuất hiện trong một bảng thì các cột dữ liệu từ bảng kia được điền giá trị



* + - HALF OUTER JOIN (LEFT hoặc RIGHT): Nếu bảng A LEFT JOIN bảng B thì kết quả gồm các bản ghi có trong bảng A, và các bản ghi không có mặt trong bảng B thì các cột từ B được điền NULL. Các bản ghi chỉ có trong B mà không có trong A sẽ không được trả về.



* + **CROSS JOIN:** mỗi bản ghi của bảng A được kết hợp với tất cả bản ghi của bảng B để tạo thành 1 tích Đề-các giữa 2 bảng (số bản ghi trả về bằng tích của số bản ghi trong 2 bản)

# Cách viết View, SP, UDF:

* **Khi nào dùng View, SP, UDF?**
  + **Nếu câu lệnh truy vấn không có tham số, vd in ra số lượng tồn kho tất cả vật tư => view (cho phép kế thừa kết quả từ view)**
  + **Sp: chấp nhận tham số và trả về 1 bảng kết quả, tuy nhiên SP vẫn chấp nhận không tham số, không kế thừa được kết quả của Sp để chạy tiếp.**
  + **UDF (Hàm): vừa chấp nhận tham số như Sp và vẫn kế thừa kết quả và tận dụng chúng như View, Không chấp nhận tham biến, Trả về hoặc là 1 table ảo hoặc là 1 giá trị rời rạc.**
* **Tạo View:** Trong SQL, 1 view là 1 đối tượng xuất hiện như table ảo để user thực hiện 1 query hay thực hiện 1 chương trình ứng dụng. Tuy nhiên, view không chứa dữ liệu. Thay vì vậy, view được định nghĩa trên 1 hay nhiều table cơ sở (hay trên các view khác), và vì vậy ta có thể xem như view là 1 phương tiện để truy nhập tới các table cơ sở. Ta có thể dùng View thực hiện các việc sau:
  + Chọn ra 1 tập các records trong 1 table cơ sở
  + Tạo ra các cột mới dựa trên các cột đã có trong các table cơ sở
  + Kết nối nhiều records từ các table cơ sở thành 1 record trong view
  + Nối các record từ nhiều table
  + Thay đổi dữ liệu trên table
  + **Nếu mệnh đề Select trong 1 view có dùng Group by thì view đó có thuộc tính Read Only.**
  + **View chứa Subquery cũng có thuộc tính ReadOnly**
  + **With Check Option:** Nếu trong View có thêm tùy chọn With Check Option thì các lệnh Insert, Update trên view này sẽ không thực hiện trên các Field làm vi phạm điều kiện của view.
  + **Derived Column:** field mới được xây dựng từ các field đã có.
  + **Xóa 1 View:** Drop View <tên view>
* **STORED PROCEDURE:**Một SP là 1 nhóm các câu lệnh Transact-SQL đã được biên dịch và chứa trong SQL dưới 1 tên nào đó và được xử lý như 1 đơn vị (chứ không phải nhiều câu lệnh SQL riêng lẻ)

Một SP có thể chứa tất cả các lệnh SQL (ngoại trừ các lệnh CREATE DEFAULT, CREATE PROCEDURE, CREATE RULE, CREATE TRIGGER, CREATE VIEW, USE). Trong SP có các tham số đầu vào, các tham số đầu ra, biến cục bộ, lệnh gán, các thao tác trên cơ sở dữ liệu và cấu trúc điều khiển việc thực thi.

Các SP của Microsoft® SQL Server™ trả về dữ liệu theo 4 dạng:

* + Các **output paramenter** có thể là:
    - Dữ liệu (ký tự hay số nguyên)
    - Một biến con trỏ cursor (các con trỏ là các tập kết quả được rút trích mỗi lần 1 dòng).
    - Các **mã trả về,** là các số nguyên
    - **Một tập kết quả** cho mỗi câu lệnh SELECT chứa trong SP hay trong những SP khác được gợi bởi SP.
    - **Một con trỏ toàn cục** có thể được tham khảo bên ngoài SP.

**Ưu điểm của SP:**

1. **Hiệu quả thực thi (performance)**
2. **Tạo khung sườn trong lập trình (Programming Framework)**
3. **Bảo mật**