

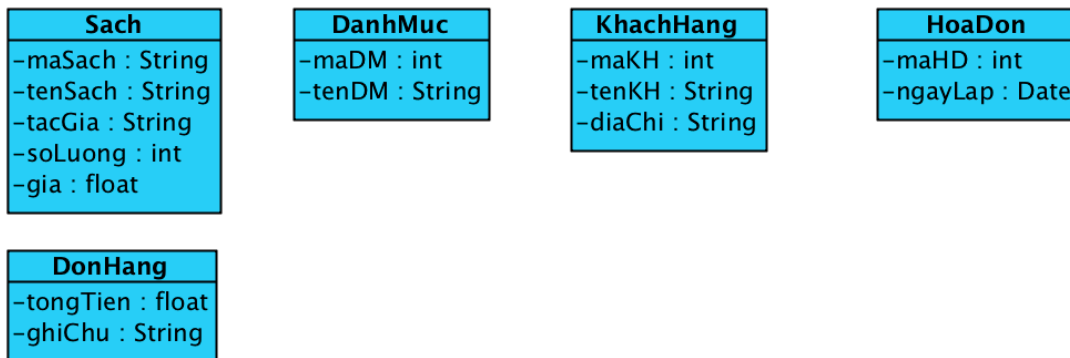
HƯỚNG DẪN CHI TIẾT CÁCH CHUYỂN BIỂU ĐỒ LỚP SANG LƯỢC ĐỒ CƠ SỞ DỮ LIỆU TRONG VISUAL PARADIGM

Người viết báo cáo: Kiều Thanh Sơn

Giảng viên phụ trách: thầy Trần Đình Quế

Bước 1 :

Chúng ta cần tạo ra một biểu đồ lớp. Ở đây , là một biểu đồ lớp gồm các thuộc tính của lớp mà chưa có các phương thức

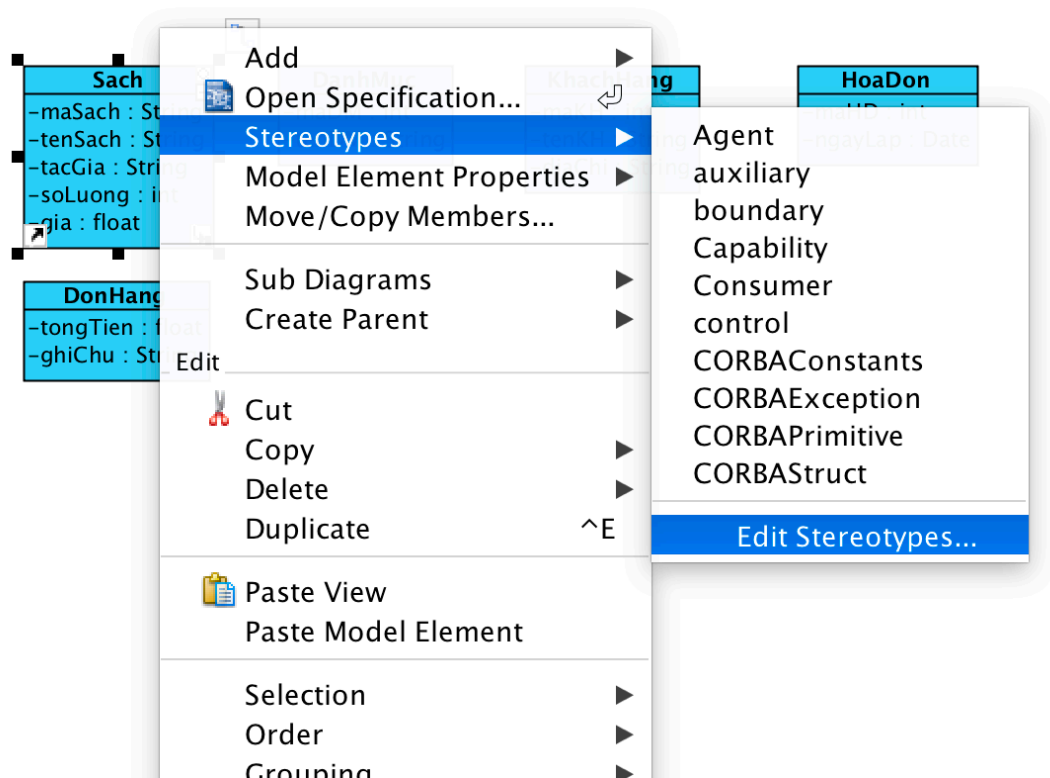


Bước 2 :

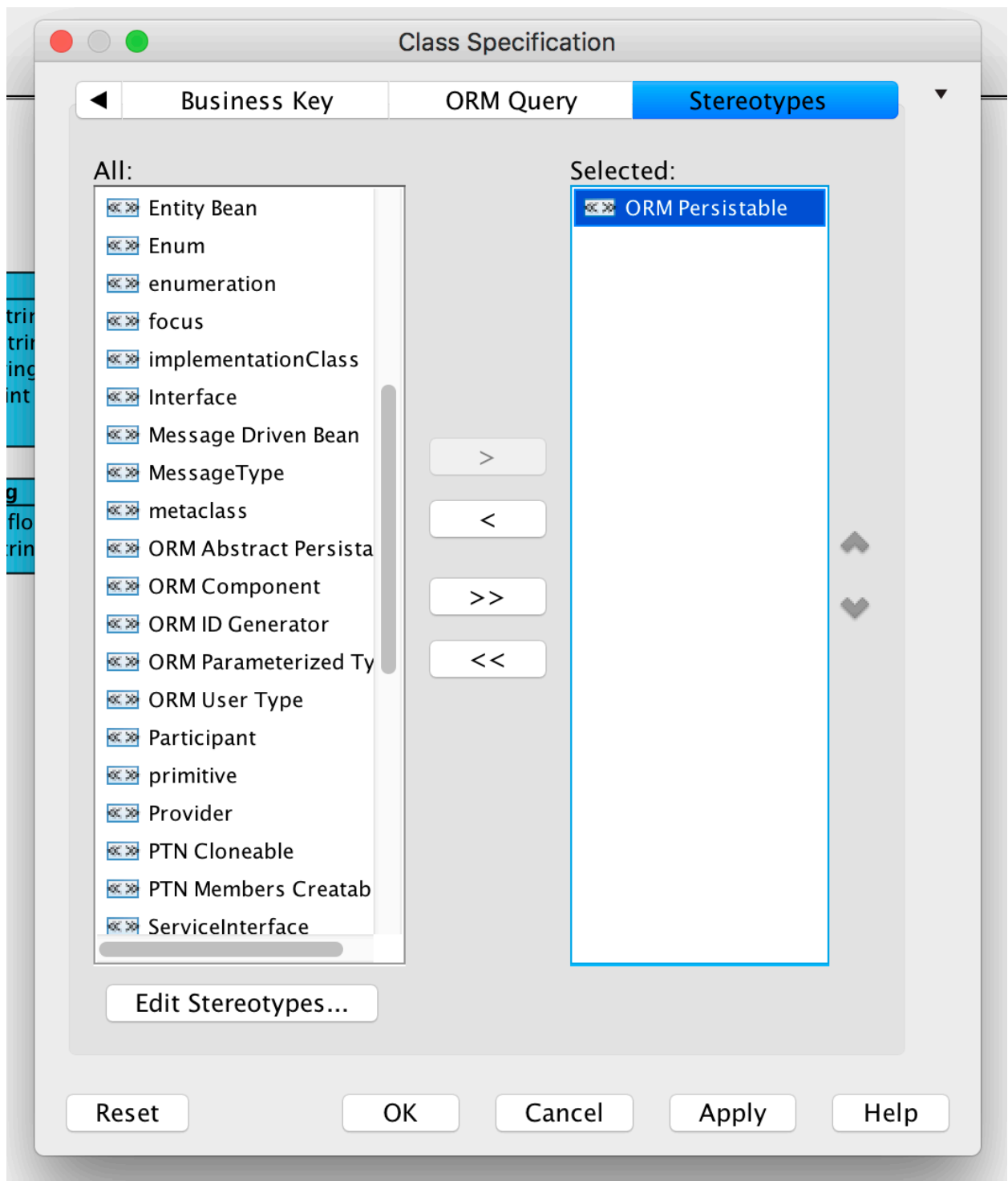
Khi ánh xạ mô hình đối tượng thành Bảng, ta có thể bắt đầu với biểu đồ lớp phân tích hoặc biểu đồ lớp thiết kế. Biểu đồ lớp phân tích gần gũi với mô hình quan hệ hơn vì nó không chỉ ra hướng liên kết. Tuy nhiên, biểu đồ lớp thiết kế có kiểu gán cho các trường nên tiện lợi khi thiết kế bảng. Vì vậy, chúng ta sẽ sử dụng mô hình phân tích để thiết kế các bảng và lấy kiểu từ mô hình thiết kế.

Tuy nhiên, để có thể ánh xạ được đối tượng với các bảng và quan hệ của các database được ánh xạ với sự ràng buộc liên quan bên trong đối tượng , ta cần sử dụng mô hình ORM Persitable

1. Chọn 1 bảng và làm như sau:

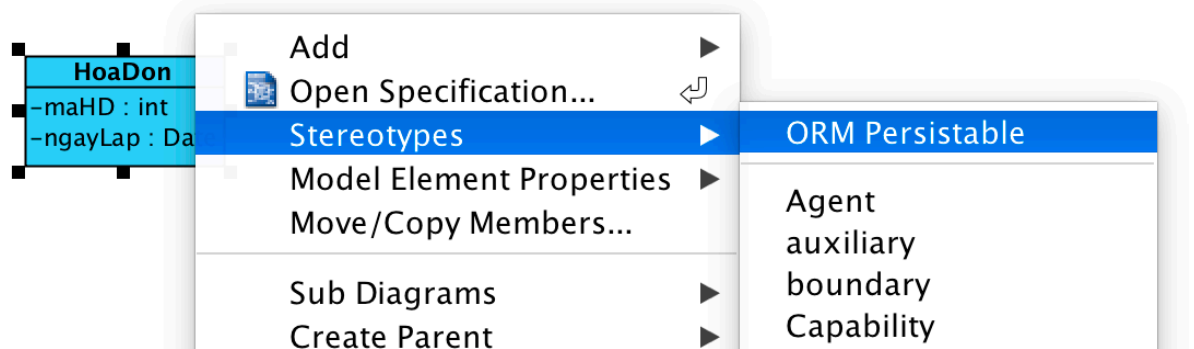


2. Chọn Edit Stereotypes -> Stereotypes và chọn ORM Persitable -> Apply



3. Ta set ORM persistable cho toàn bộ các bảng

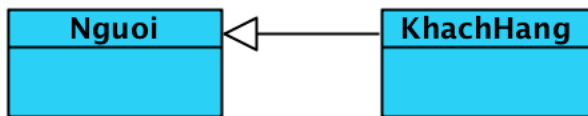
Lưu ý: Ta chỉ cần set ORM persistable 1 lần như trên, sau đó nó sẽ hiện trên đầu trong Stereotypes



Bước 3 : Ta tạo các quan hệ cho các bảng, cũng như kiểu liên kết giữa các bảng với nhau.

Có 4 mối quan hệ:

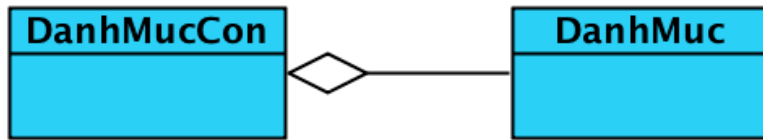
- Kế Thừa (Inheritance) một lớp con kế thừa tất cả các thuộc tính và hành vi của lớp cha của nó



- Liên kết (Association) các đối tượng của một lớp được liên kết với các đối tượng của lớp khác



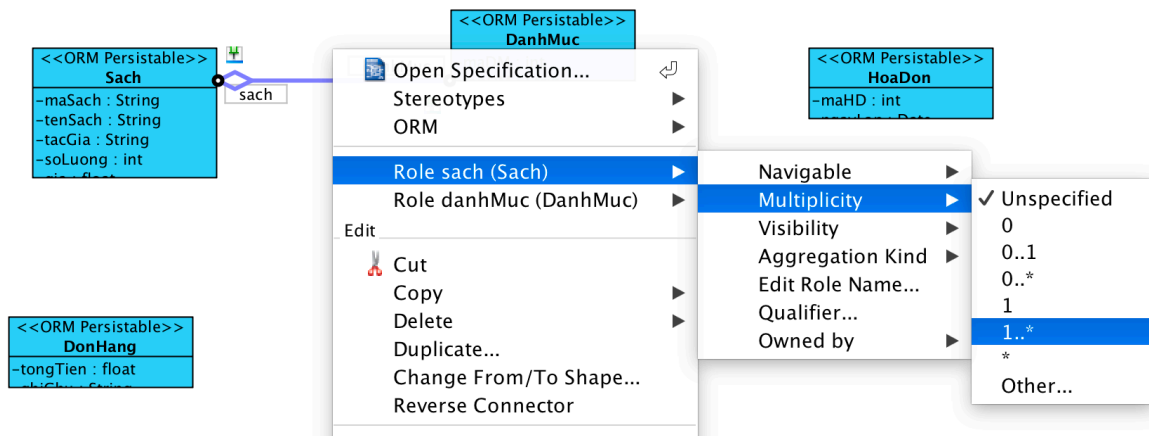
- Kết hợp (Aggregation) một loại liên kết mạnh, nghĩa là một thể hiện của một lớp được tạo ra bởi các thể hiện của một lớp khác



- Hợp thành (Composition) quan hệ kết hợp mạnh, nghĩa là một đối tượng hợp thành không thể được dùng chung với các đối tượng khác và nó sẽ mất cùng với các đối tượng được hợp thành

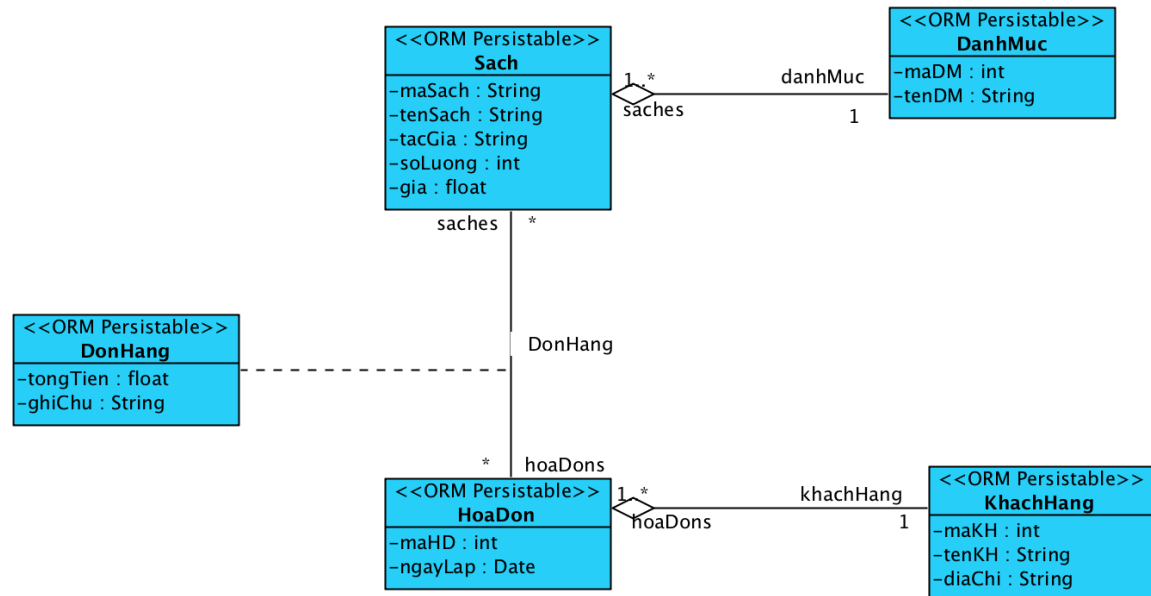


Ngoài ra, ta cần chọn kiểu liên kết cho mỗi quan hệ (n-m, 1-1, 1-n ..). Ta click chuột phải vào mỗi quan hệ và chọn như hình sau:

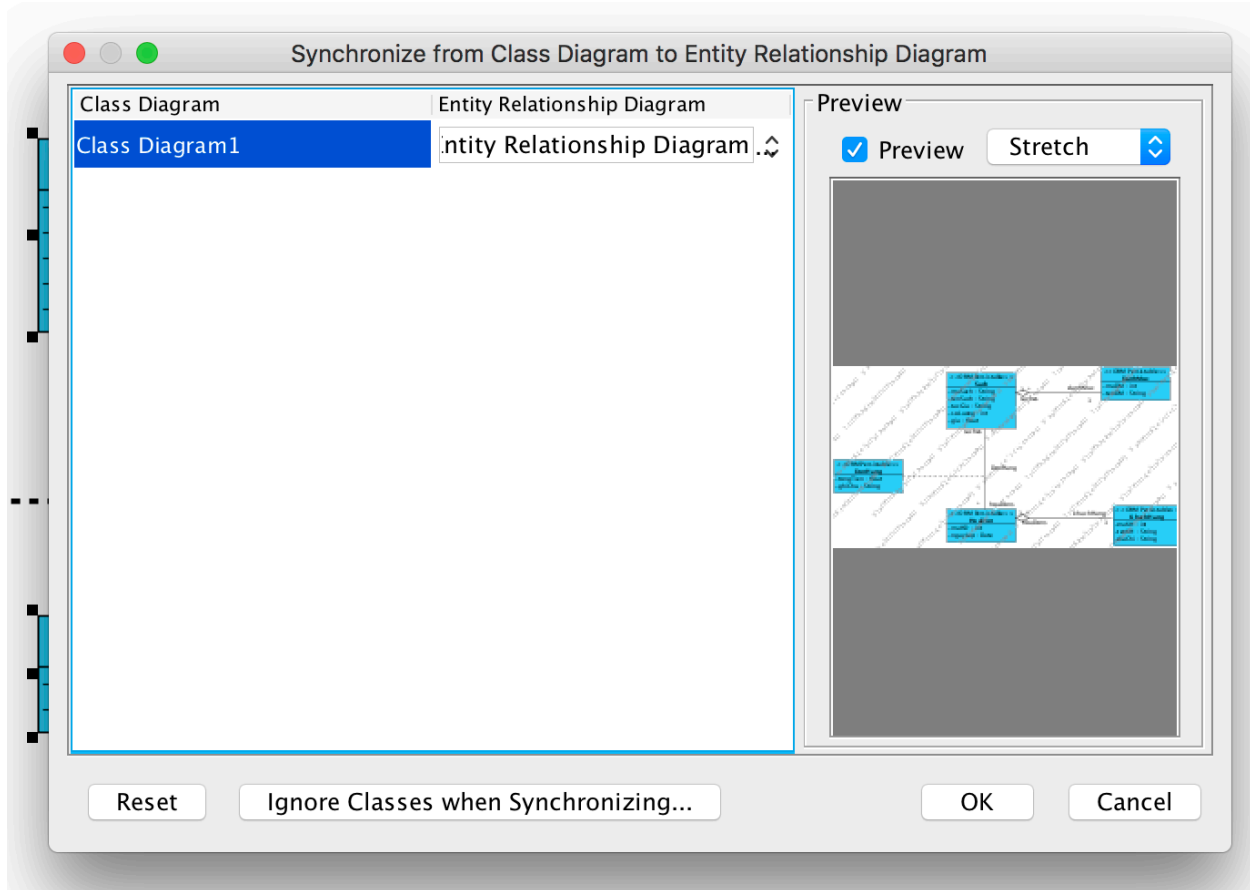


Lưu ý Ta phải set kiểu liên kết cho cả 2 đầu mỗi quan hệ. VD: Role sach & Role danhMuc

Bước 4: Ta có một mô hình đối tượng hoàn chỉnh:



Bước 5: Ta chọn các lớp cần chuyển sang ER và chọn Tools -> Hibernate -> Synchronize to Entity Relationship Diagram



Bước 6: Click Ok. Chọn khóa chính cho các bảng. Có thể chọn *Auto Generate*, VP sẽ tự sinh ra 1 khóa cho bảng đó. Sau đó click OK

Synchronize to Entity Relationship Diagram

Target Parent : Same as source model element

Select Primary Key

Select primary key:

Entity	Primary Key
DonHang	<i>Auto Generate</i>
HoaDon	MaHD
KhachHang	MaKH
DanhMuc	MaDM
Sach	MaSach

Reset all to Auto Generate

Reset all to Do Not Generate

Object-and-Relational Mapping

Class / Attribute	Entity / Column
Sach	Sach
maSach	MaSach
tenSach	TenSach
tacGia	TacGia
soLuong	SoLuong
gia	Gia
DanhMuc	DanhMuc
maDM	MaDM
tenDM	TenDM

Export Mapping

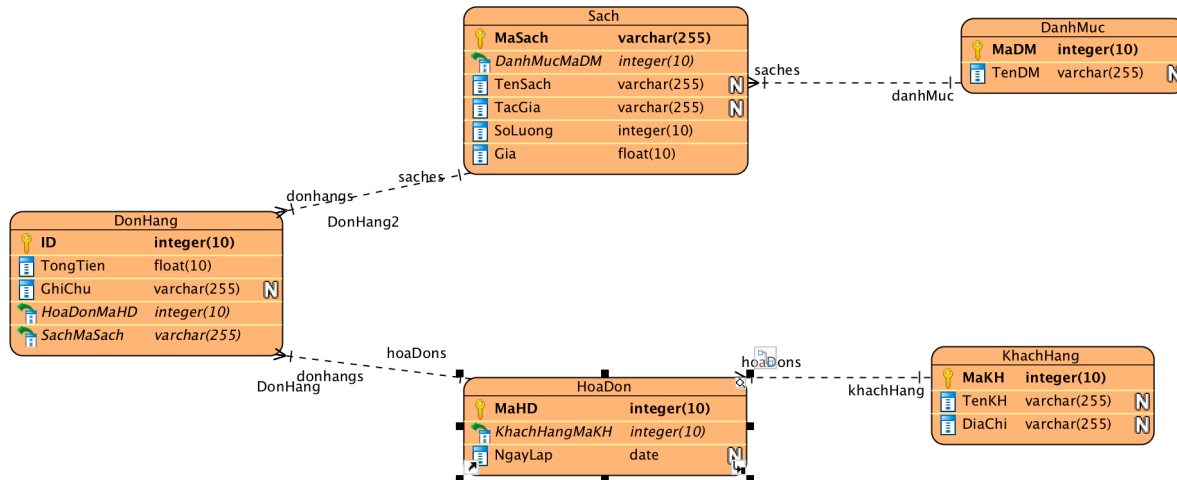
Import Mapping

☐ Add <<PK>> stereotype to ID attribute

OK

Cancel

Bước 7: VP sẽ sinh ra cho ta một mô hình ER với đầy đủ thuộc tính cũng như các quan hệ mà ta đã tạo ra từ các bước trên



Bước 8: Ta chọn những thực thể cần chuyển sang CSDL, sau đó chọn Tools -> DB -> Genarate Database

Database Generation

Output Path : ...

Generate Database :

Schema:

☐ Export to database ☒ Generate DDL ☒ Generate C...

☒ Upper Case SQL ☐ Formatted SQL

☐ Generate Individual DDL ☐ Separate Create/Drop DDL

Generate Sample Data:

Quote SQL Identifier:

Column Order:

DDL Extension:

Connection Provider Class:

Connection :

JDBC

Connection Pool Options

Database Configuration

Driver :

Driver file :

Connection URL :

User : Password :

Database Options

OK Cancel Help

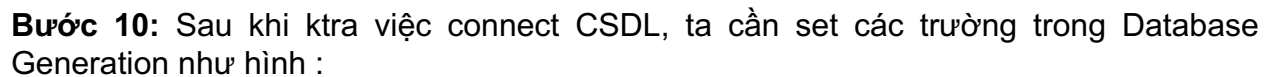
Chon CSDL mà ta sử dụng trong máy

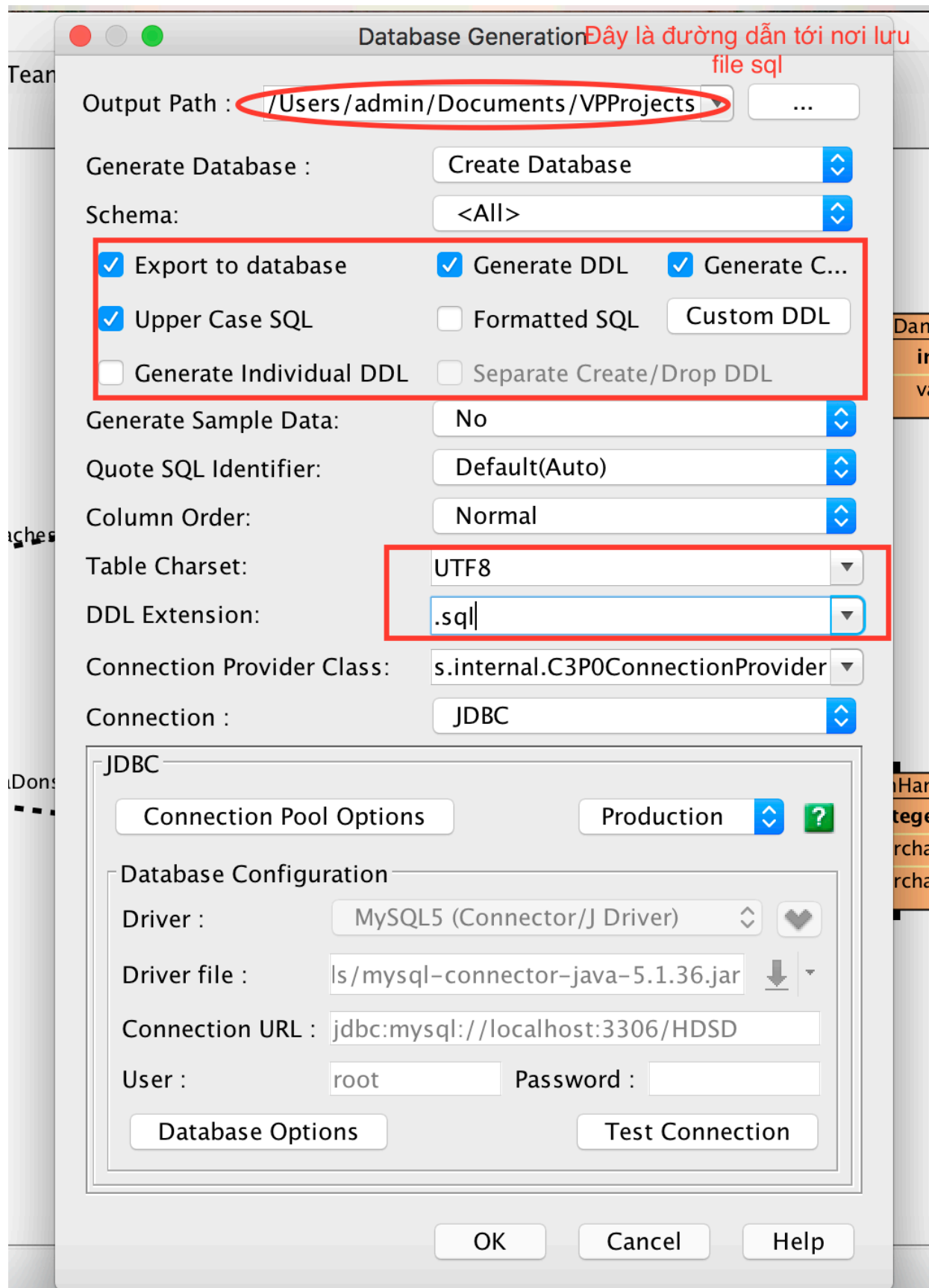
Chọn file Driver

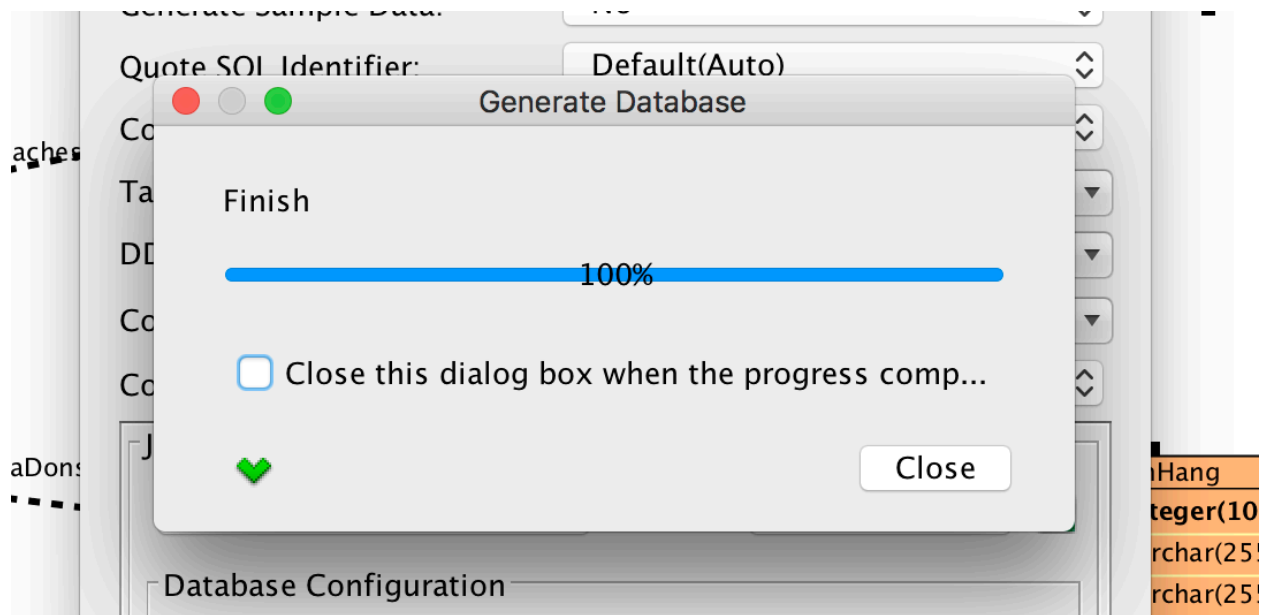
Điền hostname và cổng theo máy các bạn, và tạo 1 database trong CSDL

Kiểm tra lại User password cho khớp với CSDL, và chọn InnoDB

Kiểm tra đã truy cập được Database

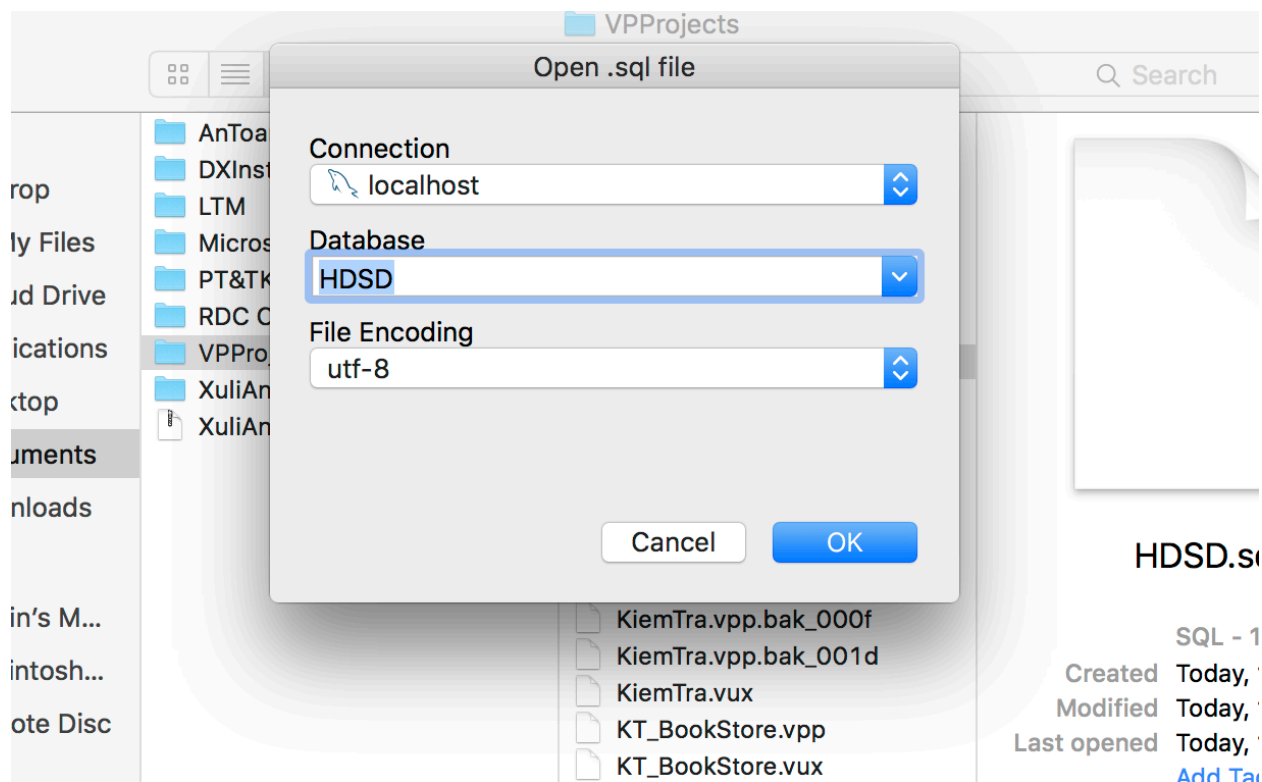






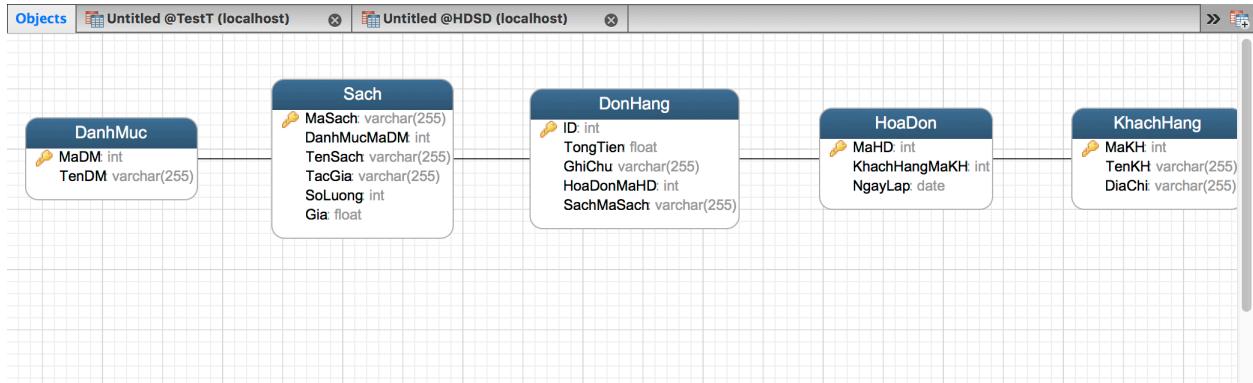
VP thông báo việc generate Database thành công, sinh ra file sql lưu trong đường dẫn phía trên **Users/admin/Documents/VPProjects/**

Bước 11: Việc cuối cùng chúng ta cần làm là import file sql lưu trong đường dẫn ở trên vào CSDL.

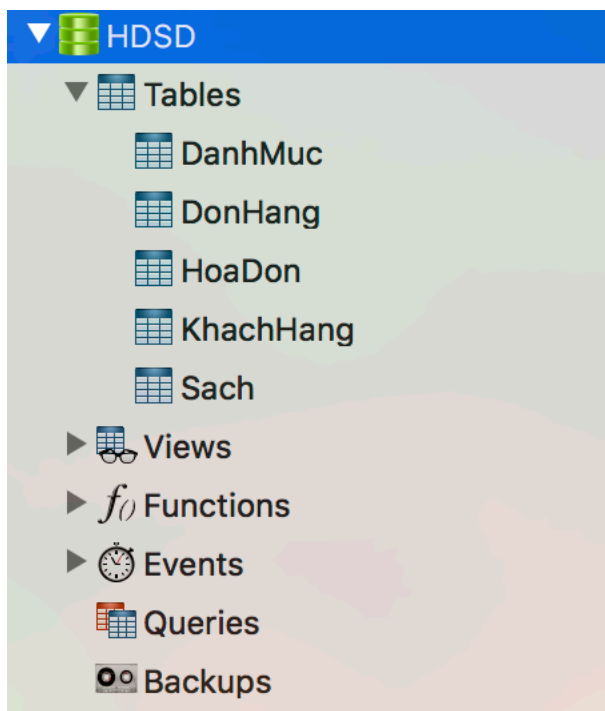


Tùy vào CSDL bạn dùng mà cách import sẽ khác nhau, ở đây mình dùng navicat premium chạy trên Mac OS X

Ta được một lược đồ CSDL như hình :



Các bảng đã được tạo ra :



Objects	(lo... ✕)	Untitled @HDSD (lo... ✕)	DanhMuc @HDSD (l... ✕)	HoaDon @HDSD (lo... ✕)	Khac
MaSach	DanhMucMaDM	TenSach	TacGia	SoLuong	Gia

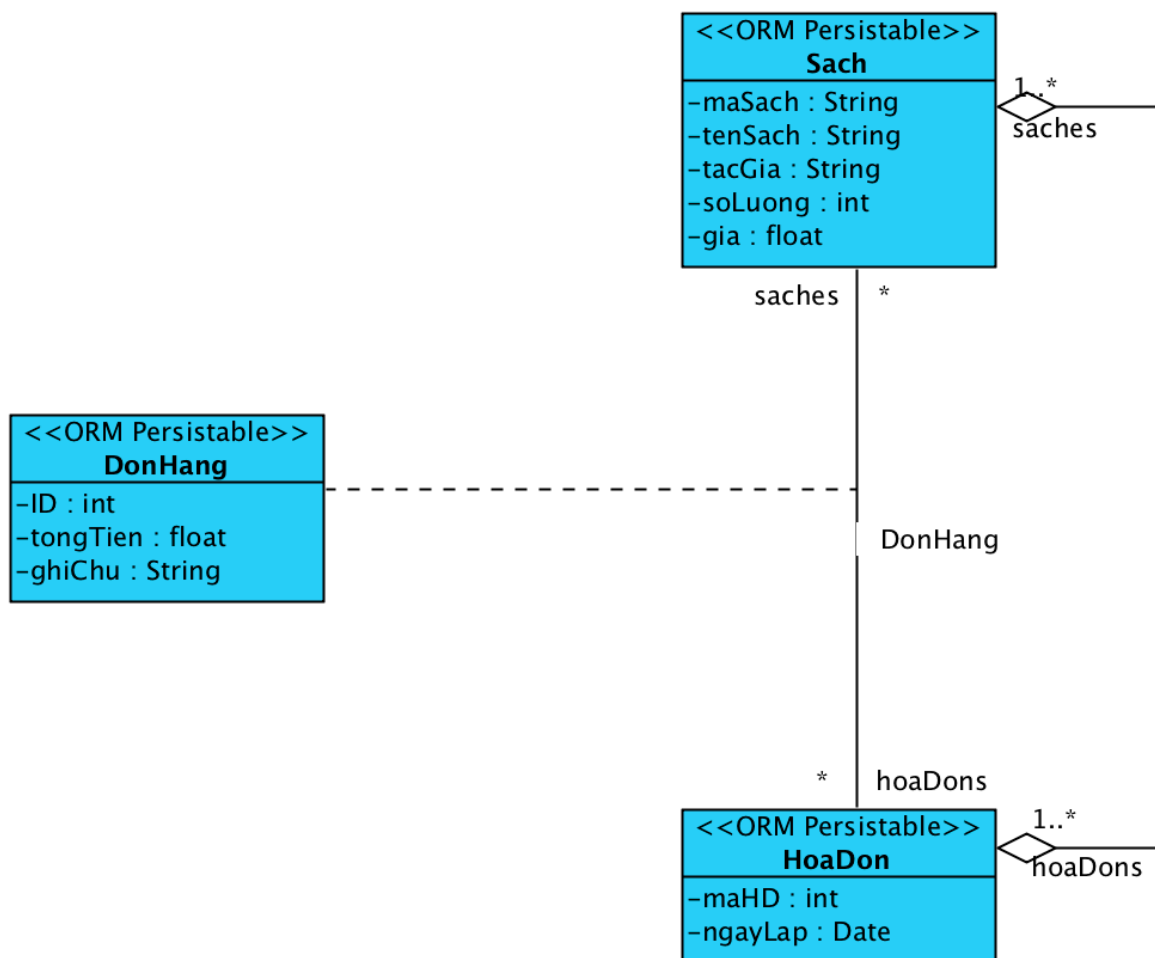
MỘT SỐ TRƯỜNG HỢP ĐẶC BIỆT

1. Mỗi liên kết nhiều- nhiều

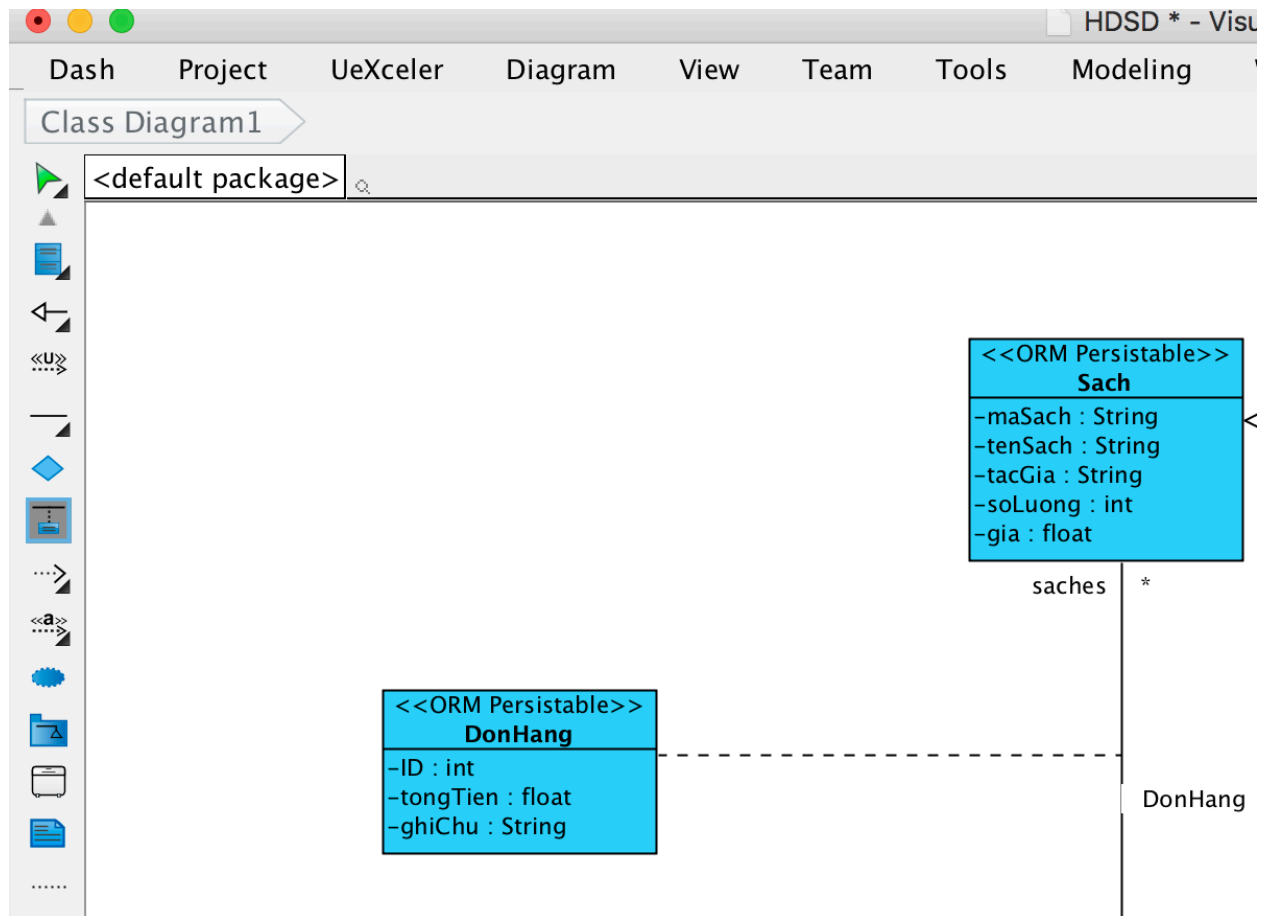
Mỗi liên kết này trong quá trình tạo CSDL sẽ sinh ra 1 bảng phụ, được gọi là một thực thể liên kết

Trong VP, khi chuyển từ mô hình đối tượng (Class Diagram) sang mô hình thực thể ER (Entity Relationship Diagram), hệ thống sẽ tự tạo ra một thực thể liên kết nếu tồn tại 2 đối tượng có mối liên kết nhiều nhiều với nhau.

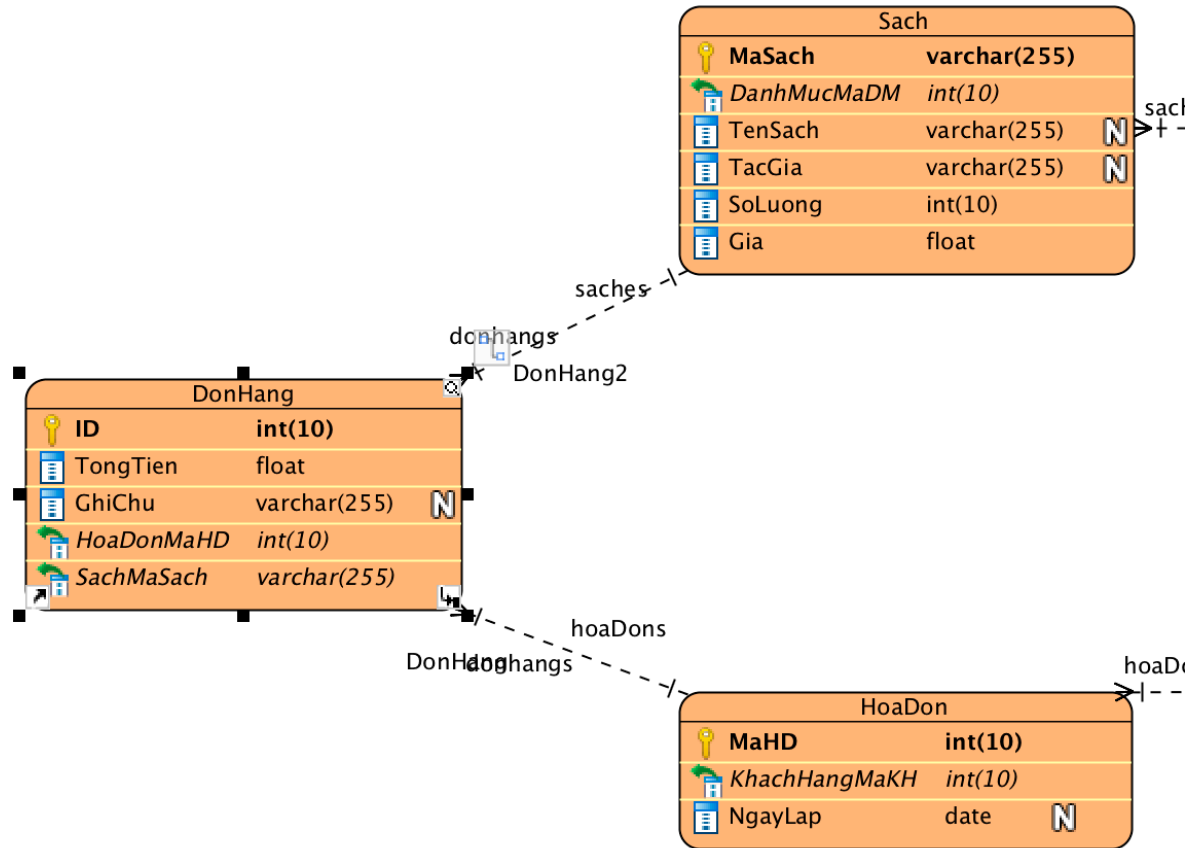
Tuy nhiên, nếu chúng ta muốn thực thể liên kết có thêm những thuộc tính mình mong muốn, VP hỗ trợ cho chúng ta tạo ra một thực thể liên kết với những thuộc tính chúng ta cần.



Để làm được như trên, chúng ta chọn Association Class, kéo từ bảng phụ vào liên kết nhiều nhiều.

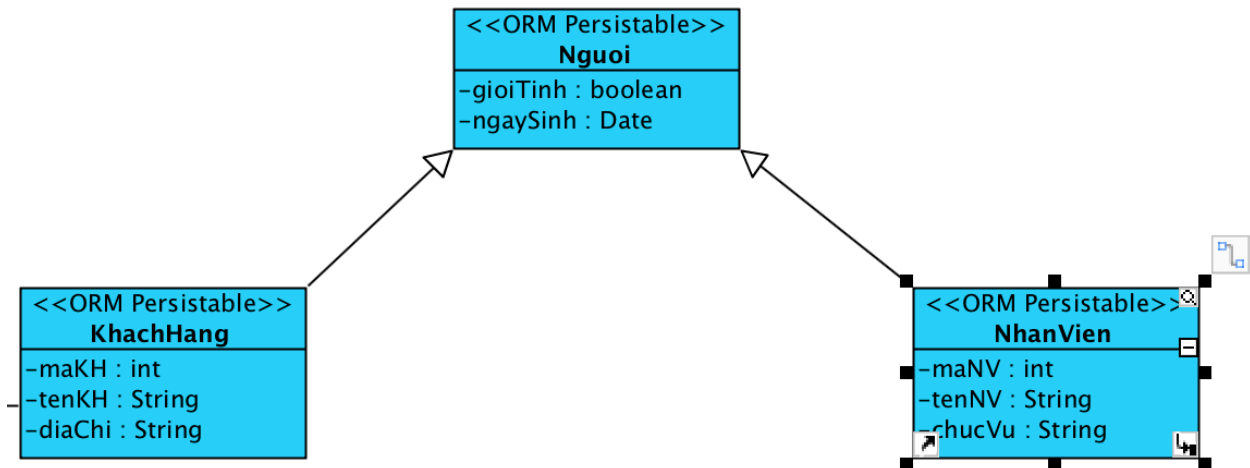


Khi chuyển sang ER :

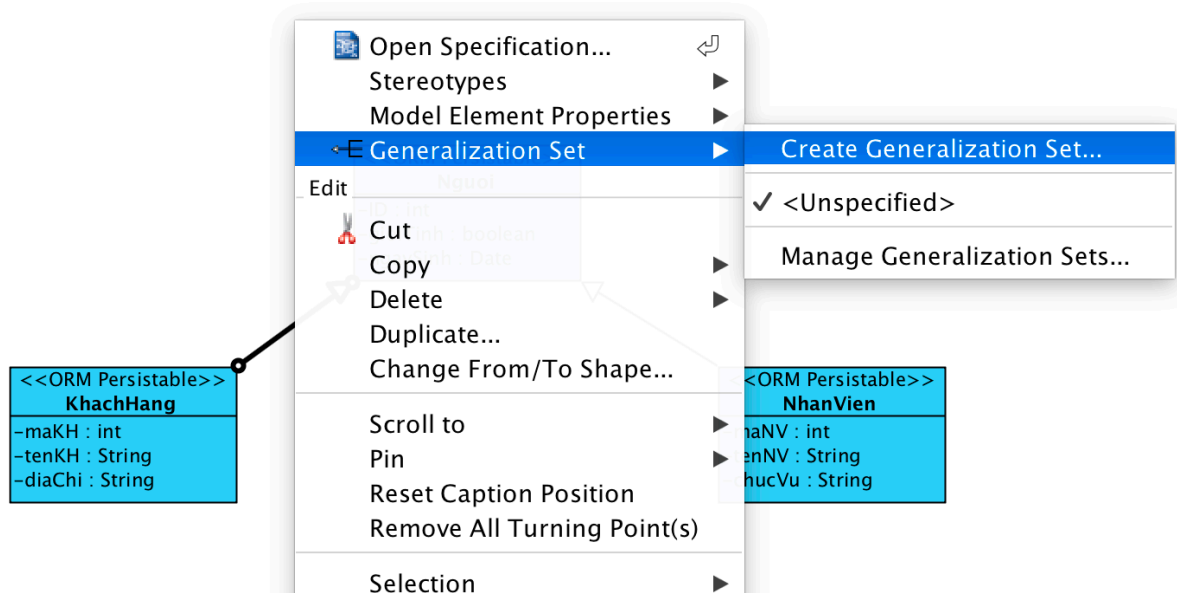


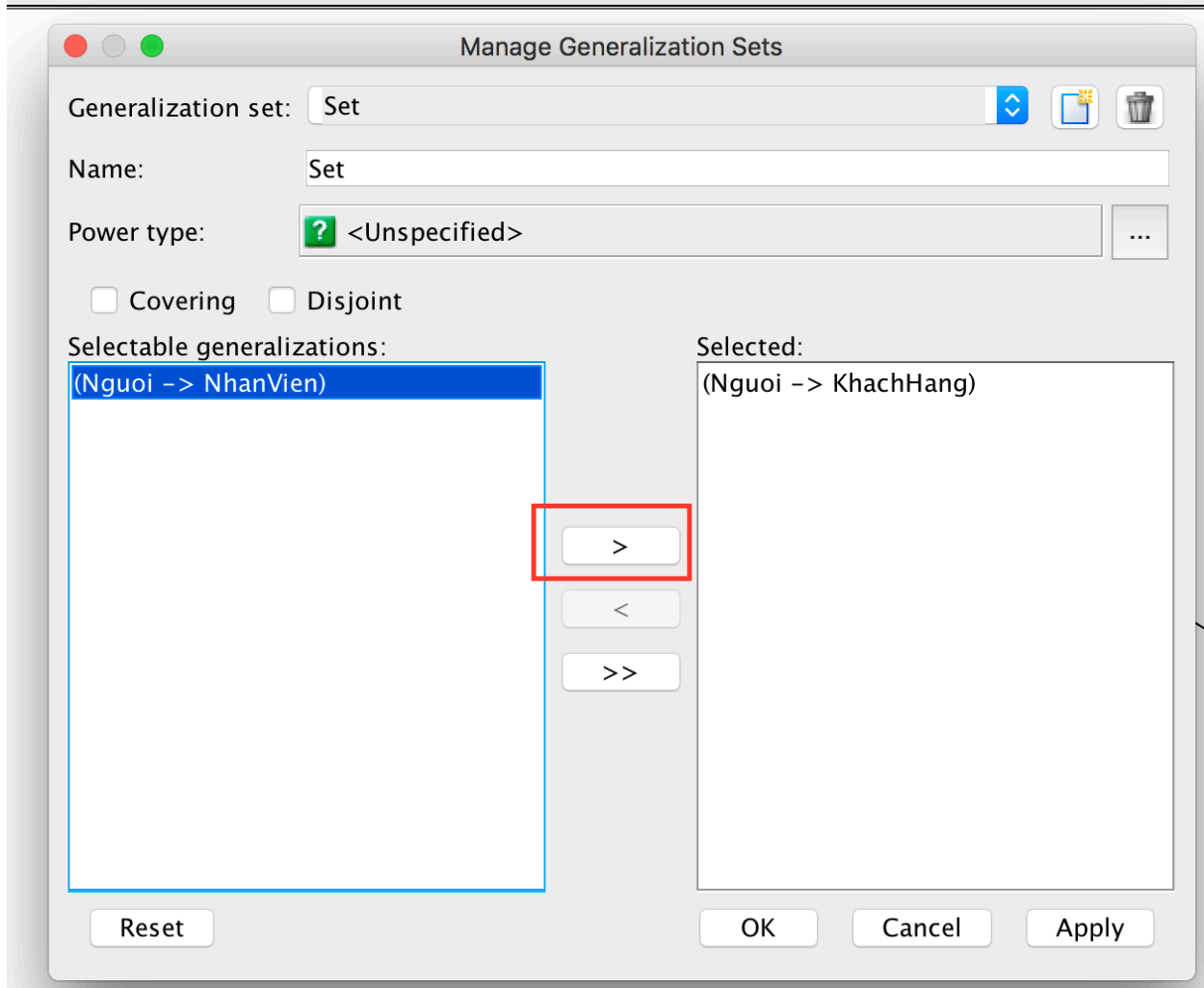
có thể thấy, VP đã tự tạo ra 2 khóa ngoại từ DonHang tham chiếu đến Sach và HoaDon (HoaDonMaHD, SachMaSach)

2. Quan hệ kế thừa

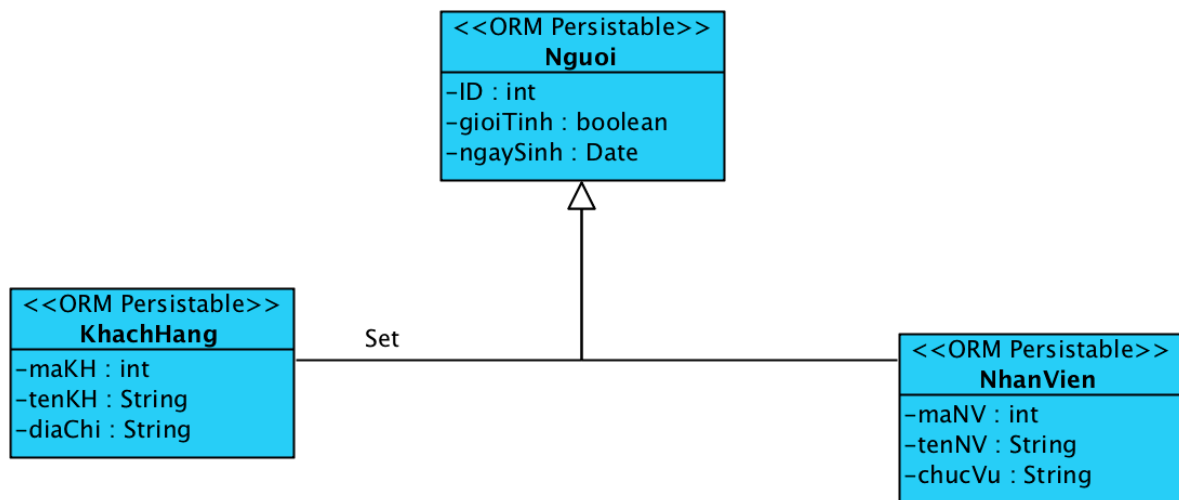


Click chuột phải vào quan hệ kế thừa



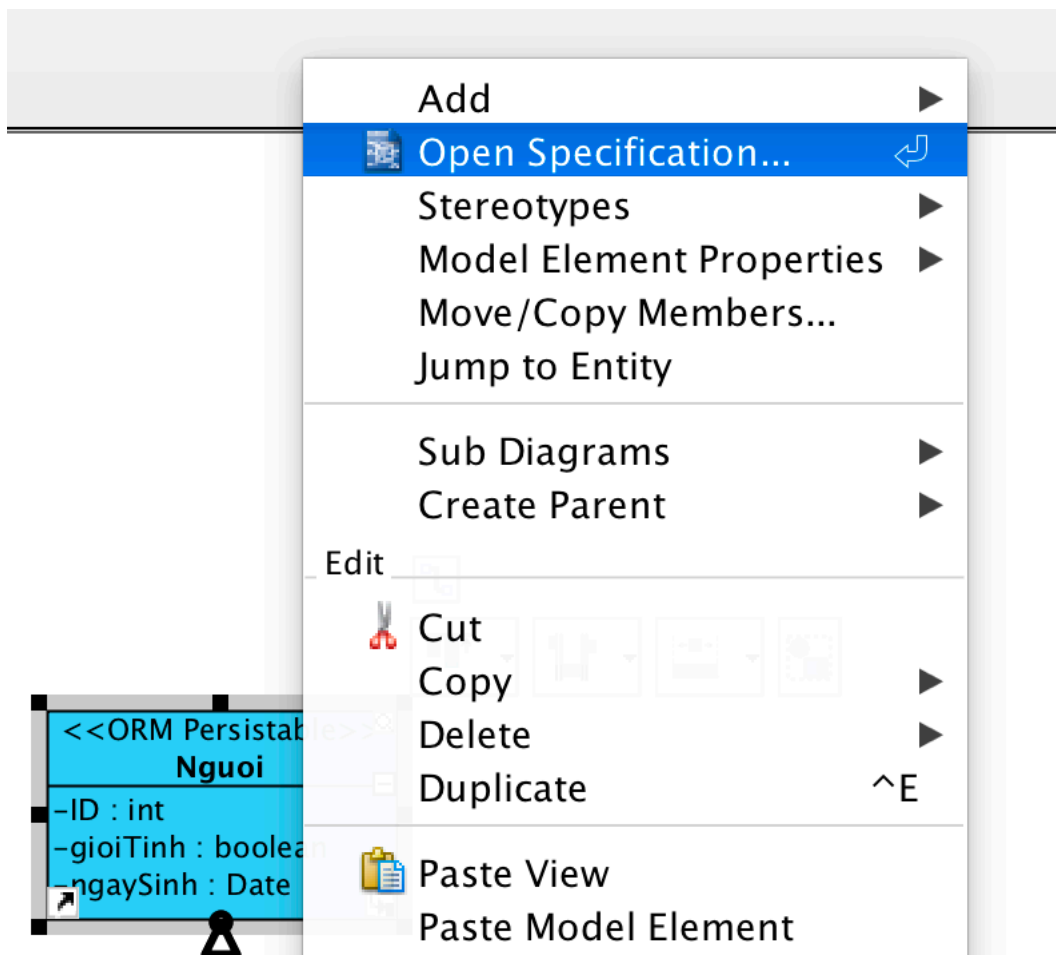


Ta sẽ được cái nhìn dễ dàng về mô hình phân cấp kế thừa:



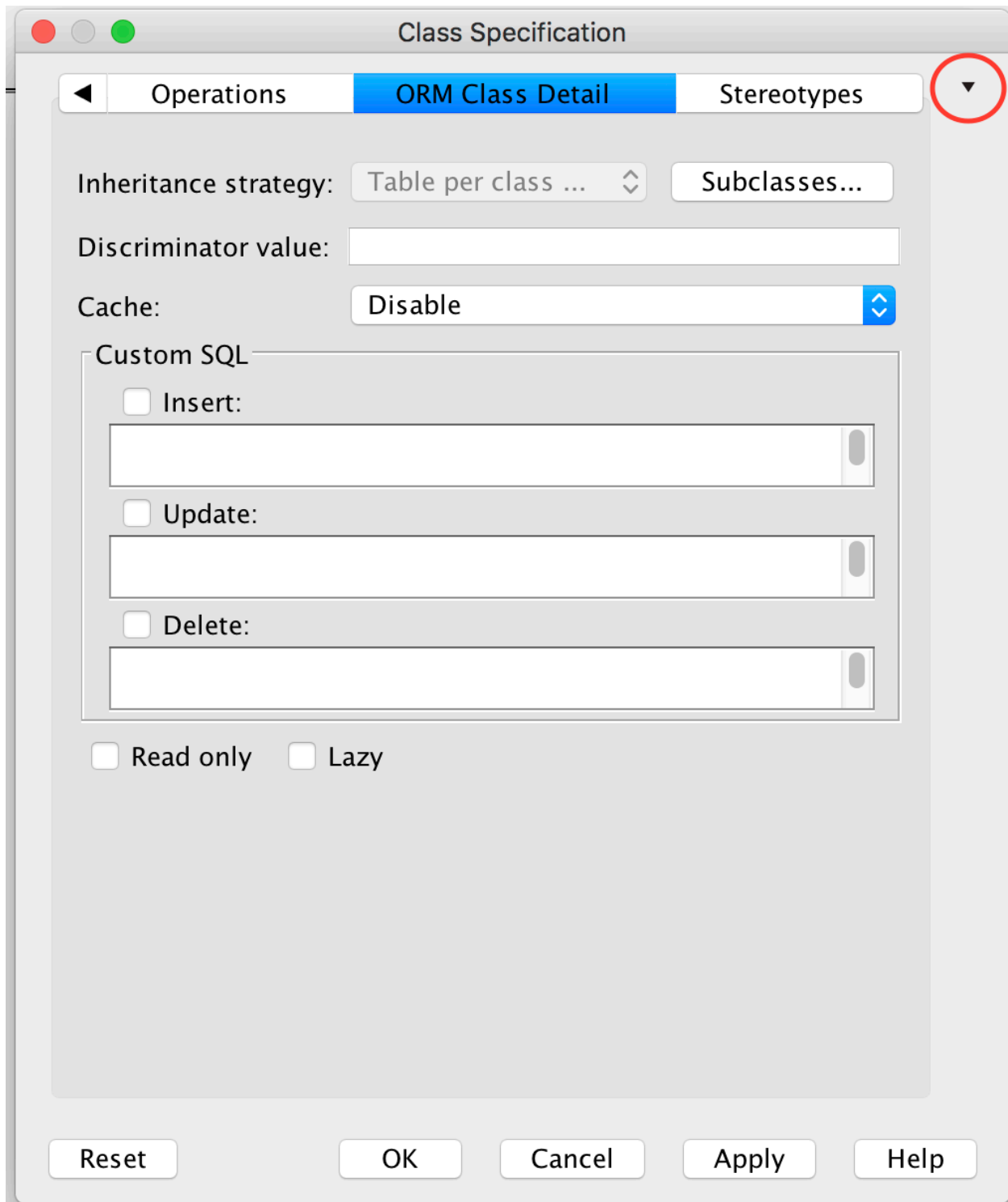
VP hỗ trợ cho chúng ta 3 kiểu ánh xạ kế thừa, tùy theo nhu cầu và cách chúng ta sử dụng csdl.

Đầu tiên, chọn các class cha và class con, chuột phải và chọn *Open Specification*



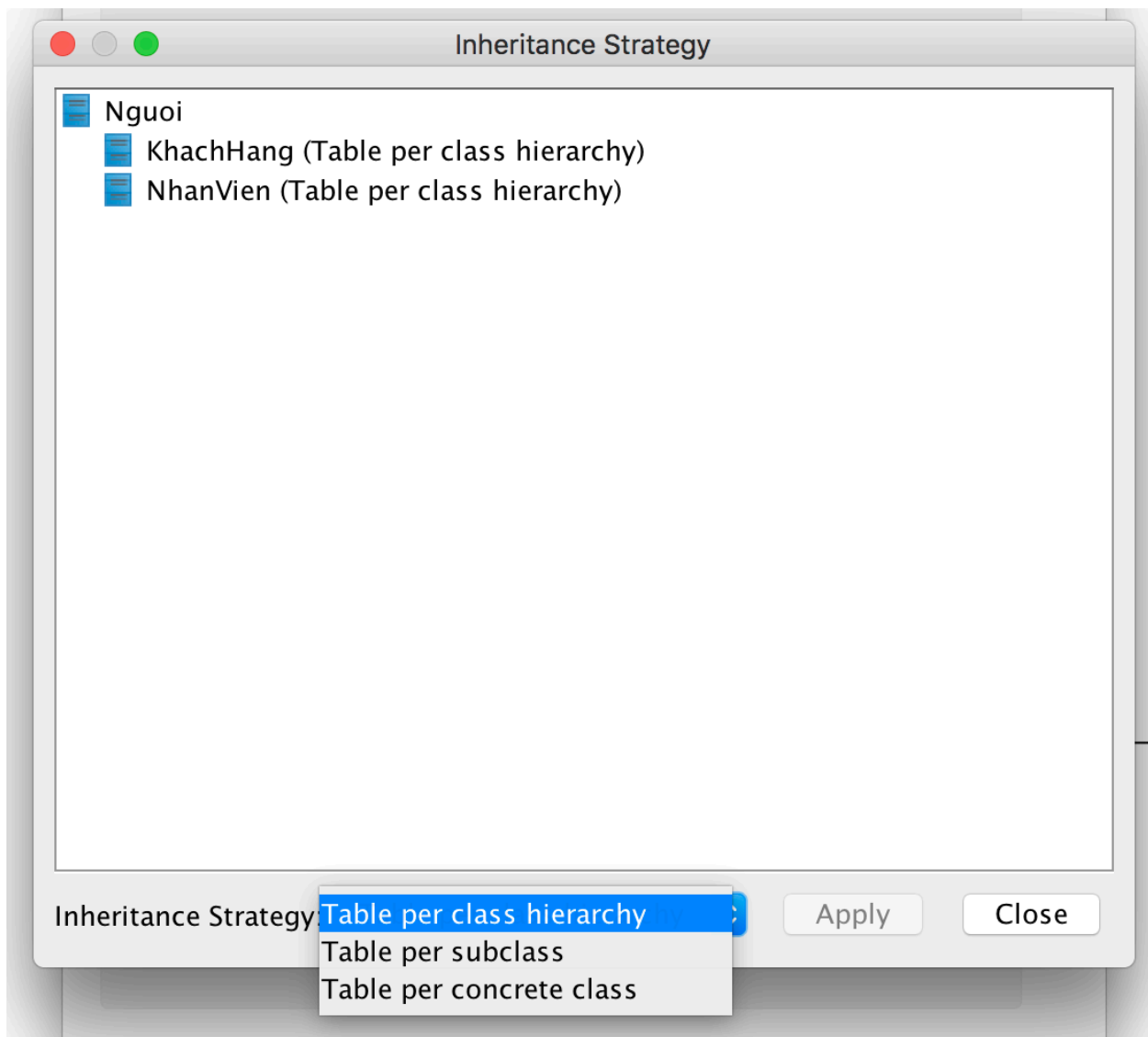
Trong giao diện **Class Specification** , chọn *ORM Class Detail*.

Lưu ý: ORM Class Detail có thể bị ẩn, có thể tìm trong phần đã đánh dấu trong ảnh sau



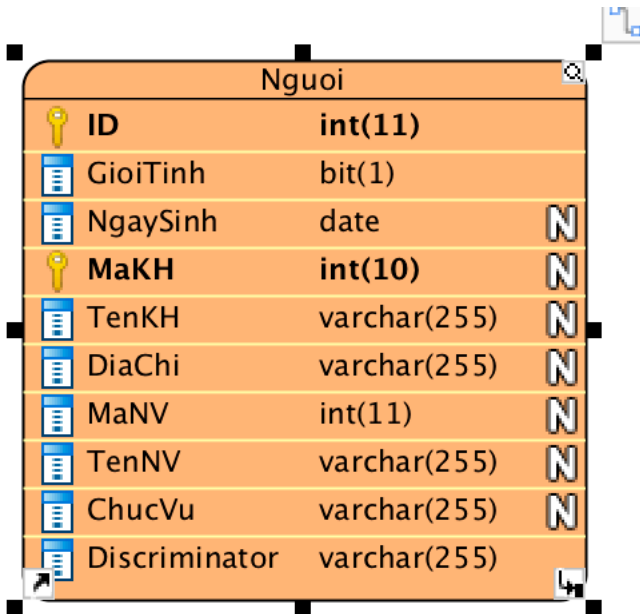
Trong bảng *ORM Class Detail* , ta chọn Subclasses và có 3 cách để ánh xạ kế thừa cho chúng ta lựa chọn











- Table per class hierarchy
- Table per subclass
- Table per concrete class



Apply 3 lựa chọn trên và chuyển sang ER

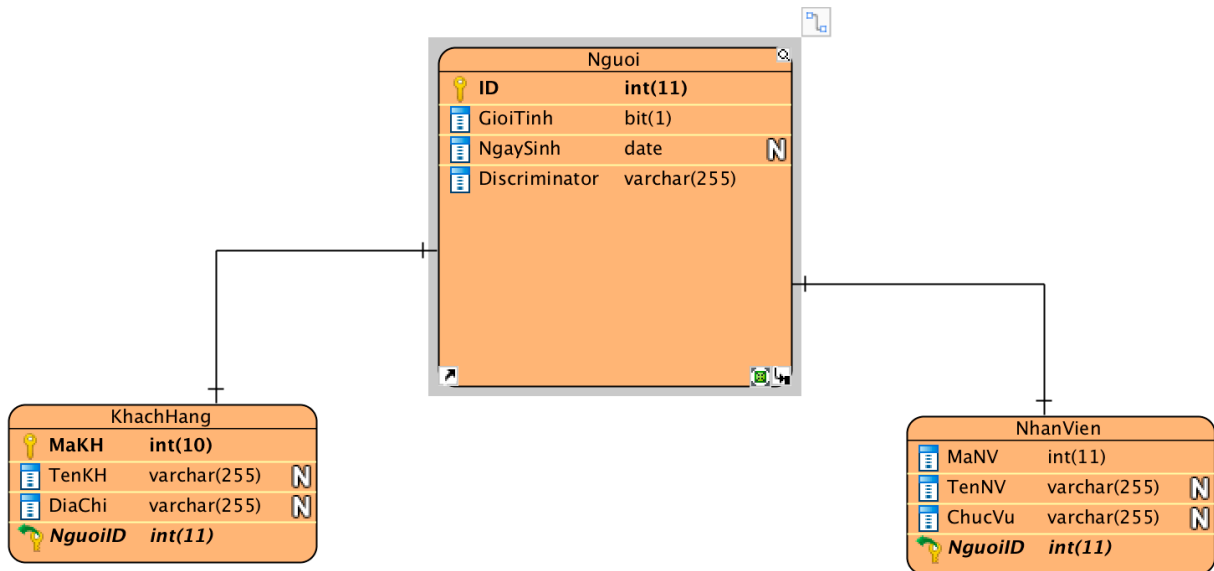
- Cách 1: Sử dụng một bảng cho lớp thực thể cha và tất cả các thuộc tính của lớp con được lưu trữ trong một bảng (Table per class hierarchy)



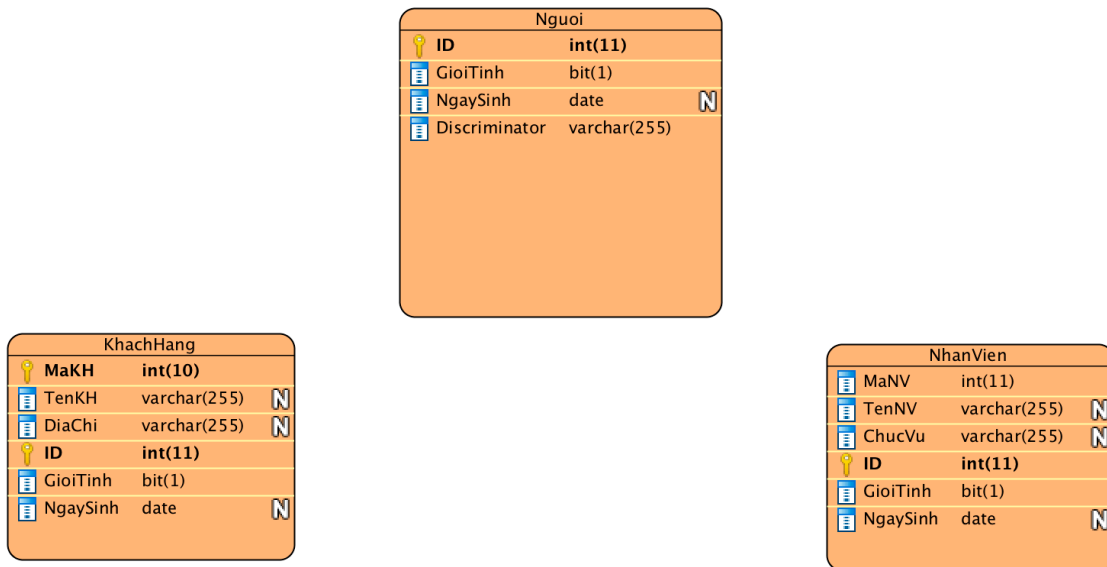
Nguoi		
	ID	int(11)
	GioiTinh	bit(1)
	NgaySinh	date
	MaKH	int(10)
	TenKH	varchar(255)
	DiaChi	varchar(255)
	MaNV	int(11)
	TenNV	varchar(255)
	ChucVu	varchar(255)
	Discriminator	varchar(255)

Cách này có ưu điểm là đơn giản và tính đa hình được hỗ trợ khi một người có thể thay đổi vai trò hoặc có nhiều vai trò (ví dụ 1 người có thể vừa là giảng viên vừa là nghiên cứu sinh, một người có thể vừa là nhân viên vừa là khách hàng...). Tuy nhiên, nó có nhược điểm là mỗi lần thêm một thuộc tính mới vào bất kỳ bảng nào trong quan hệ kế thừa đều phải thêm một cột vào bảng và do đó dễ tăng sự trùng lặp trong phân cấp lớp.

- Cách 2: Sử dụng một bảng cho một lớp con cụ thể, mỗi bảng chứa thuộc tính của chúng và có khóa ngoại tham chiếu đến lớp cha (Table per subclass)



- Cách 3: Sử dụng mỗi bảng cho mỗi lớp (Table per concrete class)



Ưu điểm của cách này là thỏa mãn khái niệm hướng đối tượng nhất và hỗ trợ tính đa hình, hơn nữa dễ dàng sửa lớp cha và thêm các lớp con mới . Tuy nhiên, có nhược điểm có quá nhiều bảng và mất nhiều thời gian để đọc và ghi dữ liệu do phải xử lí trên nhiều bảng