Họ tên: Nguyễn Huỳnh Minh Tiến

MSSV: 18110377

**Review questions:**

1. **Describe the three-schema architecture.**

* **External view:** Liên quan đến người dùng, là lập trình viên ứng dụng hoặc người dùng cuối. Mỗi người dùng sẽ có một khung nhìn (view) riêng, cung cấp một phần CSDL có liên quan đến họ. Do đó cũng giúp che giấu dữ liệu khỏi người dùng (người dùng ở mức độ nào thì sẽ có thể thấy được dữ liệu liên quan đến khung hình của họ).
* **Conceptual level:** Mô tả cấu trúc logic của CSDL. Mô hình dữ liệu và lược đồ quan hệ là những giải thích về CSDL ở mức quan niệm. DBA và trợ lý duy trì lược đồ và định nghĩa CSDL bằng cách dùng ngôn ngữ DDL.
* **Internal level:** mô tả cấu trúc lưu trữ vật lý của CSDL. Được mô tả bằng cách sử dụng các bytes cụ thể và các từ ngữ ở mức máy tính. Do phần mềm hệ quản trị CSDL đảm nhiệm.

1. **Define the following terms and give an example for each term: an attribute, the domain of an attribute, a relation schema, a relation, n-tuple, degree of a relation, a relationship, a relation instance (state), a relational database schema, a relational database state, integrity constraints.**

|  |  |  |
| --- | --- | --- |
| an attribute  (thuộc tính) | * Chỉ các loại thông tin mô tả 1 loại đối tượng cụ thể nào đó (có thể là trừu tượng), đặc trưng bởi tên gọi, kiểu giá trị (Data Type) và miền giá trị (domain).   VD: Mỗi sinh viên có thể có các thuộc tính như mã sinh viên (MaSV), tên (Ten\_SV), ngày sinh (Ng\_Sinh), mã lớp học (Ma\_lop), tiền học bổng (Hoc\_bong).   * Phân loại: * Thuộc tính đơn (thuộc tính nguyên tố): có kiểu dữ liệu vô hướng * Thuộc tính kép: có kiểu dữ liệu có cấu trúc | |
| the domain of an attribute  (miền giá trị của thuộc tính) | * Tập hợp các giá trị mà các dữ liệu có thể lấy. Kí hiệu MGT(A) hoặc Dom(A). * VD1: MGT(MÃ\_NV): số nguyên dương; * VD2: MGT(TTGĐ)={“Độc thân”; “Có gia đình”; “Ly dị”} * Miền giá trị của một thuộc tính kép bằng tích Descartes các miền giá trị thành phần. VD: MGT(Ngày\_sinh)=MGT(Ngày) x MGT(Tháng) x MGT(Năm). * Miền giá trị có thể là NULL, tùy ngữ cảnh mà nó mang nghĩa là không xác định hoặc có 1 giá trị chưa xác định tại thời điểm đó. | |
| a relation schema  (lược đồ quan hệ) | * Là tập hữu hạn các thuộc tính. Kí hiệu: R(A1, A2, ..., An), tạo thành từ một quan hệ tên là R và danh sách các thuộc tính Ai, i=1..n, có MGT(Ai)=Di tương ứng. * VD: SINHVIEN(MaSV, Ten\_SV, Ng\_Sinh, Ma\_lop, Hoc\_bong). * Một lược đồ quan hệ R(A1:D1,A2:D2,..,An:Dn) bao gồm: * Tên quan hệ R: cụ thể R=SINHVIEN * Tập thuộc tính A1, A2, ..., An (tất cả đều khác nhau trong R):   A=< MaSV, Ten\_SV, Ng\_Sinh, Ma\_lop, Hoc\_bong>   * Miền giá trị (D1,D2,…,Dn) = tập hợp các giá trị atomic:   D1=Dom(A1)=Dom(MaSV), …, D5=Dom(A5)=Dom(Hoc\_bong)   * Cấu trúc (A1:D1,A2:D2,..,An:Dn): mọi bộ trong quan hệ đều là n-tuples (cụ thể trong quan hệ SINHVIEN là 5-tuples). Mỗi bộ có n=5 entries, mỗi entry ứng với một cột của bảng. | |
| a relation  (quan hệ) | * Một quan hệ r định nghĩa trên lược đồ quan hệ R(A1, A2, ..., An), là tập con bất kỳ của tích Descartes các miền giá trị của các thuộc tính trong lược đồ quan hệ.   .   * Tính chất của quan hệ: * Mỗi quan hệ có một tên duy nhất. * Mỗi thuộc tính của một quan hệ đều có tên khác nhau. * Tất cả các dòng của một bảng đều có cùng định dạng và cùng số lượng entries * Tất cả các giá trị của một thuộc tính có cùng một miền giá trị. * Mỗi thành phần của một bộ hoặc một filed là một giá trị atomic. * Thứ tự các bộ không quan trọng. * Mỗi bộ là duy nhất, không có 2 bộ nào trùng nhau - do quan hệ là *set* (tập hợp các bộ). Trong SQL quan hệ là *bag* (có thể xuất hiện dòng trùng). | |
| n-tuple (bộ) | * Cho , gọi là một bộ được thành lập trên các thuộc tính của lược đồ quan hệ Q(R), nếu và q thỏa . * Bộ giá trị còn được gọi là mẫu tin hay bản ghi (record) hay dòng – khác dòng tiêu đề của bảng (Row). Mỗi bộ của một quan hệ R bắt buộc phải có đủ n entries tương ứng với n thuộc tính của quan hệ. * VD: (18110377, Nguyễn Huỳnh Minh Tiến, 2000, 181101B, 5500000) là một bộ của lược đồ quan hệ SINHVIEN ở trên với 5 entries tương ứng với 5 thuộc tính (MaSV, Ten\_SV, Ng\_Sinh, Ma\_lop, Hoc\_bong). | |
| degree of a relation (bậc của quan hệ) | Là số thuộc tính của lược đồ quan hệ tương ứng.  VD: Quan hệ SINHVIEN(MaSV, Ten\_SV, Ng\_Sinh, Ma\_lop, Hoc\_bong) có bậc là 5 vì quan hệ này có tất cả 5 thuộc tính. | |
| a relationship (quan hệ) | Là một trạng thái tồn tại giữa 2 bảng CSDL quan hệ khi một bảng có khóa ngoại tham chiếu khóa chính của bảng khác. Các mối quan hệ cho phép CSDL quan hệ có thể tách và lưu trữ ở các bảng khác nhau, trong khi liên kết các mục dữ liệu.  VD: 2 bảng COURSE và PREREQUISITE được gọi là có quan hệ với nhau. Vì thuộc tính Course\_number là khóa ngoại ở bảng PREREQUISITE, tham chiếu đến thuộc tính Course\_number ở bảng COURSE. | |
| a relation instance (state) | Một thể hiện của quan hệ, kí hiệu r(R), trên một lược đồ quan hệ R(A1, A2, ..., An), là một tập n-tuples <d1,d2,…,dn> với di (i=1..n) là một thành phần của Dom(Ai) hoặc là NULL.  VD: Quan hệ DEPT\_LOCATIONS và các thể hiện của nó: | |
| a relational database schema | Một lược đồ CSDL quan hệ là tập hợp các lược đồ quan hệ, kí hiệu S = {R1, R2, ..., Rm} và một tập các ràng buộc. Trong đó, Ri (i=1..m) là một lược đồ quan hệ, kí hiệu là R(A1,A2,…,An), với mỗi thuộc tính Aj (j=1..n) biến thiên trên miền giá trị Dom(Aj). |  |
| a relational database state | Trạng thái của một CSDL quan hệ, kí hiệu DB, của một lược đồ CSDL quan hệ S, là một tập các thể hiện của quan hệ DB = {r1, r2, ..., rm} với mỗi ri là một trạng thái của quan hệ Ri, thỏa mãn các ràng buộc toàn vẹn.  VD: Đây là một trạng thái của một CSDL quan hệ: |  |
| integrity constraints  (ràng buộc toàn vẹn) | * Là những quy tắc hoặc hạn chế được áp dụng trên CSDL, cần tuân theo khi nhập liệu vào CSDL, để hạn chế miền giá trị của thuộc tính, giới hạn các giá trị dữ liệu mà nó có thể thay đổi (thêm, xóa, sửa), nhằm nhấn mạnh sự chắc chắn, chính xác và đúng đắn của dữ liệu. * Ràng buộc toàn vẹn được mô tả khi định nghĩa lược đồ bởi người thiết kế CSDL, được kiểm tra khi cập nhật các quan hệ và được chia sẻ bởi tất cả các ứng dụng sử dụng CSDL đó. * Đặc trưng: * Bối cảnh: một hay nhiều quan hệ cần sử dụng để kiểm tra RBTV. Các quan hệ này có khả năng làm vi phạm RBTV khi thực hiện thao tác thêm, xóa, sửa dữ liệu. * Nội dung: Một phát biểu về điều kiện của RBTV. Điều kiện này phải được kiểm tra trên mọi thể hiện (mọi thay đổi của thể hiện) của (các) quan hệ. * Bảng tầm ảnh hưởng: xác định thao tác nào (thêm, xóa, sửa dữ liệu) cần kiểm tra trên các quan hệ nào. * Hành động: hành động phản hồi phải được thực hiện nếu RBTV bị vi phạm. * Một thể hiện hợp lệ của một quan hệ là thể hiện thỏa mãn tất cả các ràng buộc toàn vẹn được mô tả. Hệ quản trị CSDL sẽ không chấp nhận các thể hiện không hợp lệ.   VD: trong quan hệ SINHVIEN, thì mỗi MaSV trong bảng là duy nhất và khác NULL | |

1. **Why are tuples in a relation not ordered?**

Một quan hệ (table) được định nghĩa như một tập của các tuples (dòng). Về mặt toán học, giữa các thành phần của tập trên không được sắp xếp thứ tự; vì thế, các tuples trong một quan hệ cũng không có bất kỳ một thứ tự cụ thể nào.

1. **Why are duplicate tuples not allowed in a relation?**

Bởi vì nó vi phạm các ràng buộc toàn vẹn của quan hệ, tạo ra các dữ liệu dư thừa tồn tại trong CSDL, làm tốn dung lượng lưu trữ, khiến cho việc thao tác trên dữ liệu (truy vấn, chèn, xóa, cập nhật…) trở nên chậm chạm và gặp nhiều khó khăn để xử lý.

1. **What is the difference between a key and a superkey?**

* Với mỗi quan hệ ta có:
* Siêu khóa (superkey – SK): là một tập thuộc tính (có thể gồm nhiều thuộc tính) của quan hệ R. Đặc điểm của siêu khóa là không có 2 bộ nào trong r(R) có cùng giá trị tại các thuộc tính SK này, giúp xác định duy nhất một bộ của một quan hệ. Nghĩa là cho t1, t2 là các bộ khác nhau trong R, ta có ràng buộc t1[SK] ≠ t2[SK].
* Để hiệu quả hơn, ta thường quan tâm đến siêu khóa tối thiếu (minimal superkey) hay khóa (key). Khóa được dùng để xác định duy nhất một bộ trong quan hệ. Khóa chỉ áp dụng trên lược đồ quan hệ, chứ không phải trên thể hiện của quan hệ. Nghĩa là nhìn vào thể hiện hiện thời, không thể nói chắc chắn rằng tập các thuộc tính đó là khóa. Đặc điểm:
  + Khóa (key) cũng là một siêu khóa.
  + Sau khi loại bỏ bớt một số thuộc tính từ tập ban đầu, không tìm thấy tập con nào là siêu khóa.

Như vậy, một tập fields được gọi là khóa của một quan hệ nếu: Không có 2 bộ khác nhau nào có thể có cùng giá trị ở tất cả các field khóa. Điều này không đúng với bất kỳ tập còn nào của khóa.

* Để tìm siêu khóa tối thiểu (minimal superkey hoặc key): Ta loại bỏ các thuộc tính và kiểm tra RBTV nêu trên có còn được thỏa mãn (loại bỏ cho tói khi ta không thể loại bỏ thuộc tính được nữa).
* Nếu quan hệ có nhiều hơn một siêu khóa tối thiểu, thì một trong số các khóa được chọn bởi DBA trở thành khóa chính, các khóa còn lại trở thành khóa ứng cử.
* Khóa ứng cử (candidate key) là một siêu khóa tối thiểu: một tập thuộc tính tối thiểu xác định duy nhất một bộ của một quan hệ.
* Khóa chính (primary key) là một khóa ứng cử được lựa chọn như là một khóa riêng của quan hệ. Khóa chính không thể mang giá trị NULL.

1. **Discuss the entity integrity and referential integrity constraints. Why is each considered important?**

* Ràng buộc toàn vẹn thực thể (entity integrity constraints): khóa chính (primary key) của quan hệ không thể chấp nhận giá trị NULL. Giúp xác định tính duy nhất của bộ trong quan hệ theo một hay nhiều thuộc tính.
* Ràng buộc toàn vẹn tham chiếu (referential integrity constraints): nếu một khóa ngoại tồn tại trong một quan hệ, thì giá trị khóa ngoại phải khớp với giá trị khóa chính của một bộ trong quan hệ được tham chiếu, hoặc mang giá trị NULL. Ràng buộc này là đặc trưng của CSDL quan hệ.
* Lý do cần thiết phải có RBTV (về thực thể và tham chiếu):
  + Bảo đảm tính kết dính (coherence) của các bộ phận cấu thành nên CSDL.
  + Bảo đảm tính nhất quán (consistency) về mặt logic của CSDL.
  + Bảo đảm CSDL luôn thể hiện đúng ngữ nghĩa thực tế.
  + Trong quá trình phân tích - thiết kế cơ sở dữ liệu, nếu không quan tâm đúng mức đến những vấn đề trên, thì có thể dẫn đến những hậu quả rất nghiêm trọng về tính an toàn và toàn vẹn dữ liệu, đặc biệt trong những CSDL tương đối lớn.

**Exercises**

1. ***[COMPANY]***

* ***Suppose that each of the following Update operations is applied directly to the database state shown in Figure 3.6. Discuss all integrity constraints violated by each operation, if any, and the different ways of enforcing these constraints.***

1. **Insert <‘Robert’, ‘F’, ‘Scott’, ‘943775543’, ‘1972-06-21’, ‘2365 Newcastle Rd, Bellaire, TX’, M, 58000, ‘888665555’, 1> into EMPLOYEE.**

* Về miền giá trị: Tất cả dữ liệu trong câu lệnh đều thỏa RBTV này tương ứng trong các cột.
* Ràng buộc khóa ngoại (còn gọi là ràng buộc tham chiếu):
  + Cột Super\_ssn là khóa ngoại tham chiếu đến cột Ssn. Trong câu lệnh Insert ở trên, người lãnh đạo có mã ‘888665555’ đã có trong bảng. Vậy ràng buộc này thỏa.
  + Cột Dno là khóa ngoại, tham chiếu đến cột Dnumber của bảng Department. Trong câu lệnh Insert trên, giá trị cột Dno được chèn vào là 1 (đã có bên bảng Department). Như vậy, ràng buộc này thỏa.

≫ Kết luận: Câu lệnh trên có thể thực hiện mà không vi phạm các RBTV.

1. **Insert <‘ProductA’, 4, ‘Bellaire’, 2> into PROJECT.**

* Về miền giá trị: Tất cả dữ liệu trong câu lệnh đều thỏa RBTV này tương ứng trong các cột.
* Về khóa ngoại: Cột Dnum trong bảng PROJECT là khóa ngoại tham chiếu qua cột Dnumber bên bảng DEPARTMENT. Trong câu lệnh trên, giá trị cột Dnum là 2, nhưng tìm trong cột Dnumber trong bảng DEPARTMENT ta không thấy giá trị này. Như vậy câu lệnh trên không thực hiện được vì vi phạm ràng buộc này.

≫ Muốn thực hiện được thao tác của câu lệnh này ta phải thực hiện chèn thêm 1 department (phòng ban) có Dnumber (mã phòng ban) là 2 vào trong bảng DEPARTMENT, sau đó ta mới thực hiện được câu lệnh trên.

1. **Insert <‘Production’, 4, ‘943775543’, ‘2007-10-01’> into DEPARTMENT.**

* Về miền giá trị: Tất cả dữ liệu trong câu lệnh đều thỏa RBTV này tương ứng trong các cột.
* Về khóa chính: Lệnh insert này vi phạm ràng buộc khóa chính (cột Dnumber) vì đã tồn tại một bộ (tuple) khác, đã có trong bảng với giá trị thuộc tính Dnumber là 4.

≫ Để có thể thực hiện được thao tác của lệnh này, ta phải đổi giá trị tương ứng ở cột Dnumber thành một giá trị khác chưa tồn tại trong bảng DEPARTMENT.

1. **Insert** **<‘677678989’, NULL, ‘40.0’> into** **WORKS\_ON.**

* Lệnh insert này vi phạm ràng buộc toàn vẹn về thực thể (khóa ngoại Pno tham chiếu đến cột Pnumber bảng PROJECT không thể nhận giá trị NULL).
* Cột ESSN bảng WORKS\_ON là khóa ngoại tham chiếu qua cột SSN ở bảng EMPLOYEE. Câu lệnh trên, giá trị cột Essn là ‘677678989’. Nhưng vì bảng EMPLOYEE không tồn tại tuple nào mang giá trị này ở cột Ssn. Vậy nên, nó cũng vi phạm ràng buộc về tham chiếu.

≫ Để có thể thực hiện được thao tác của lệnh này, ta phải đổi giá trị tương ứng ở cột Pno thành giá trị hợp lệ (khác NULL và đã tồn tại trong cột Pnumber của bảng PROJECT). Đồng thời với công việc trên, đó là:

* + Cách 1: đổi giá trị Essn=‘677678989’ thành một giá trị hợp lệ khác (một giá trị nào đó đã tồn tại trong cột Ssn của bảng EMPLOYEE).
  + Cách 2: chèn trước một bộ có Ssn=‘677678989’ vào bảng EMPLOYEE.

Sau đó, ta mới có thể thực hiện lệnh insert nêu trên.

1. **Insert** **<‘453453453’, ‘John’, ‘M’, ‘1990-12-12’, ‘spouse’> into** **DEPENDENT.**

* Về miền giá trị: Tất cả dữ liệu trong câu lệnh đều thỏa RBTV này tương ứng trong các cột.
* Về khóa: Khóa ngoại Essn của bảng DEPENDENT tham chiếu đến cột Ssn của bảng EMPLOYEE. Ta thấy giá trị ‘453453453’ tương ứng cột Essn đã có trong cột Ssn bảng EMPLOYEE. Do vậy thỏa mã ràng buộc toàn vẹn về tham chiếu.

≫ Kết luận: Lệnh Insert này có thể thao tác mà không vi phạm các ràng buộc toàn vẹn.

1. **Delete the** **WORKS\_ON tuples with** **Essn = ‘333445555’.**

≫ Ta thấy, giá trị Essn = ‘333445555’ có tồn tại trong bảng WORKS\_ON. Do đó, ta có thể thực hiện lệnh delete này.

1. **Delete the EMPLOYEE tuple with** **Ssn = ‘987654321’.**

Về khóa: Ssn của bảng EMPLOYEE là một khóa chính, có tham chiếu đến các bảng DEPARTMENT, WORKS\_ON và DEPENDENT. Khi xóa một tuple Ssn = ‘987654321’ ở bảng EMPLOYEE sẽ vi phạm ràng buộc toàn vẹn về tham chiếu.

≫ Để có thể thực hiện được thao tác delete này, ta có thể:

* + Cách 1: Xóa bỏ các bộ dữ liệu ở các bảng có tham chiếu đến tuple bị xóa này.
  + Cách 2: Thay vì xóa bỏ, ta sửa đổi thuộc tính tham chiếu thành giá trị khác, chẳng hạn như null hoặc một giá trị hợp lệ khác.

1. **Delete the PROJECT tuple with** **Pname = ‘ProductX’.**

Tuple có Pname = ‘ProductX’ ở bảng PROJECT, tức là bộ (‘ProductX’, 1, ‘Bellaire’, 5), ta thấy giá trị Pnumber=1 trên được các bộ ở bảng WORKS\_ON tham chiếu đến. Do vậy, nếu xóa đi bộ này, sẽ vi phạm ràng buộc toàn vẹn về tham chiếu.

≫ Để có thể thực hiện được thao tác delete này, ta có thể:

* + Cách 1: xóa bỏ tất cả các bộ (cụ thể là các bộ ở bảng ở bảng WORKS\_ON có Pno=1), tham chiếu đến bộ bị xóa ở trên.
  + Cách 2: thay vì xóa bỏ, ta có thể đổi những giá trị trên thành NULL hoặc một giá trị hợp lệ khác (tồn tại trong bảng PROJECT, và khác 1).

1. **Modify the** **Mgr\_ssn and Mgr\_start\_date of the DEPARTMENT tuple with** **Dnumber = 5 to** **‘123456789’ and ‘2007-10-01’, respectively.**

* Ta thấy, tuple có Dnumber = 5 ở bảng DEPARTMENT có tồn tại.
* Về miền giá trị: ‘123456789’ và ‘2007-10-01’ đều thỏa mãn miền giá trị tương ứng của Mgr\_ssn và Mgr\_start\_date.

≫ Kết luận: Ta có thể thực hiện lệnh nêu trên mà không vi phạm các ràng buộc toàn vẹn.

1. **Modify the Super\_ssn attribute of the EMPLOYEE tuple with Ssn = ‘999887777’ to** **‘943775543’.**

Trong bảng EMPLOYEE, Super\_Ssn là khóa ngoại tham chiếu đến Ssn.

* ***Trường hợp 1:*** Nếu các câu hỏi là độc lập, nghĩa là câu (j) sẽ thao tác trực tiếp lên trạng thái đề cho lúc ban đầu:

Ta thấy trong cột Ssn chưa có bộ nào mang giá trị ‘943775543’ này. Do đó, việc chỉnh sửa như trên vi phạm ràng buộc toàn vẹn về tham chiếu.

≫ Để thực hiện được lệnh này, ta có thể:

* + Cách 1: Thay ‘943775543’ bằng một giá trị hợp lệ khác (những giá trị đã có trong cột Ssn).
  + Cách 2: Chèn thêm một bộ với giá trị Ssn=‘943775543’ vào bảng EMPLOYEE. Sau đó, mới thực hiện lệnh chỉnh sửa nêu trên.
* ***Trường hợp 2:*** Nếu câu (a) thực hiện trước, chèn thành công Ssn=‘943775543’ và quan hệ EMPLOYEE:

Khi đó, trong bảng EMPLOYEE, cột Ssn đã tồn tại giá trị Ssn=‘943775543’. Do đó, ta có thể thực hiện thao tác chỉnh sửa thuộc tính Super\_ssn như trên mà không vi phạm các RBTV.

1. **Modify the Hours attribute of the WORKS\_ON tuple with** **Essn = ‘****999887777’ and Pno = 10 to ‘5.0’**

Ta thấy, có tồn tại một bộ (‘999887777’, 10, 10.0) trong bảng WORKS\_ON, với các thuộc tính Essn = ‘999887777’ và Pno = 10, theo đúng như lệnh chỉnh sửa đã cho. Mặt khác, giá trị mới 5.0 cũng thuộc miền dữ liệu của cột Hours.

≫ Kết luận: có thể thực thi lệnh trên màng không vi phạm các ràng buộc toàn vẹn.

* ***Which of the following operations are not correct? Why? Show the results of the correct operations:***

1. Π**Ssn, Fname, Lname, Salary(Employee)**

* Câu lệnh này đúng. Đúng về cú pháp của lệnh chiếu và các cột Ssn, Fname, Lname, Salary đều có trong bảng EMPLOYEE.

|  |  |  |  |
| --- | --- | --- | --- |
| **Ssn** | **Fname** | **Lname** | **Salary** |
| 123456789 | John | Smith | 30000 |
| 333445555 | Franklin | Wong | 40000 |
| 999887777 | Alicia | Zelaya | 25000 |
| 987654321 | Jennifer | Wallace | 43000 |
| 666884444 | Ramesh | Narayan | 38000 |
| 453453453 | Joyce | English | 25000 |
| 987987987 | Ahmad | Jabbar | 25000 |
| 888665555 | James | Borg | 55000 |

1. σ**Sex = F and Salary > 30000(Employee)**

* Câu lệnh này đúng.

≫ Kết quả là một bảng hiển thị bộ dữ liệu thỏa mãn điều kiện Sex = F and Salary > 30000.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Fname** | **Minit** | **Lname** | **Ssn** | **Bdate** | **Address** | **Sex** | **Salary** | **Super\_ssn** | **Dno** |
| Jennifer | S | Wallace | 987654321 | 1941-06-20 | 291 Berry, Bellaire, TX | F | 43000 | 888665555 | 4 |

1. σ**Sex = F and Salary > 30000 (**Π**Ssn, Fname, Lname, Salary(Employee))**

* Câu lệnh này đúng về cú pháp nhưng sai về logic.
* Vì phép chiếu ở trong ngoặc được thực hiện trước và cho ra kết quả là một bảng gồm 4 cột Ssn, Fname, Lname, Salary. Sau đó phép chọn được thực hiện trên bảng kết quả của phép chiếu với điều kiện Sex = F and Salary > 30000. Nhưng trong bảng kết quả không có cột Sex.

1. Π**Ssn, Fname, Lname, Salary (**σ**Sex = F and Salary > 30000(Employee))**

* Câu lệnh này đúng.
* Phép chọn ở trong ngoặc được thực hiện trước và cho ra kết quả là một bộ duy nhất thỏa mãn điều kiện Sex = F and Salary > 30000, tương ứng với nữ nhân viên Jennifer S. Wallace với mức lương 43000. Sau đó thực hiện phép chiếu và cho ra kết quả là một bảng gồm 4 cột Ssn, Fname, Lname, Salary tương ứng của nữ nhân viên trên.

|  |  |  |  |
| --- | --- | --- | --- |
| **Ssn** | **Fname** | **Lname** | **Salary** |
| 987654321 | Jennifer | Wallace | 43000 |

1. **Department × Dept\_Location**

* Câu lệnh này đúng.

≫ Kết quả của phép tích Descartes giữa 2 bảng DEPARTMENT và DEPT\_LOCATION là một bảng mới. Do cả hai bảng có cột cùng tên là Dnumber, nên phải đổi tên cột này ở 1 bảng. Cụ thể, Dnumber ở bảng DEPT\_LOCATIONS, có thể đổi thành Dnumber2 để phân biệt.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Dname** | **Dnumber** | **Mgr\_ssn** | **Mgr\_start\_date** | **Dnumber2** | **Dlocation** |
| Research | 5 | 333445555 | 1988-05-22 | 1 | Houston |
| Research | 5 | 333445555 | 1988-05-22 | 4 | Stafford |
| Research | 5 | 333445555 | 1988-05-22 | 5 | Bellaire |
| Research | 5 | 333445555 | 1988-05-22 | 5 | Sugarland |
| Research | 5 | 333445555 | 1988-05-22 | 5 | Houston |
| Administration | 4 | 987654321 | 1995-01-01 | 1 | Houston |
| Administration | 4 | 987654321 | 1995-01-01 | 4 | Stafford |
| Administration | 4 | 987654321 | 1995-01-01 | 5 | Bellaire |
| Administration | 4 | 987654321 | 1995-01-01 | 5 | Sugarland |
| Administration | 4 | 987654321 | 1995-01-01 | 5 | Houston |
| Headquarters | 1 | 888665555 | 1981-06-19 | 1 | Houston |
| Headquarters | 1 | 888665555 | 1981-06-19 | 4 | Stafford |
| Headquarters | 1 | 888665555 | 1981-06-19 | 5 | Bellaire |
| Headquarters | 1 | 888665555 | 1981-06-19 | 5 | Sugarland |
| Headquarters | 1 | 888665555 | 1981-06-19 | 5 | Houston |

1. **Department**  **Dept\_Location**

* Câu lệnh trên đúng.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Dname** | **Dnumber** | **Mgr\_ssn** | **Mgr\_start\_date** | **Dlocation** |
| Research | 5 | 333445555 | 1988-05-22 | Bellaire |
| Research | 5 | 333445555 | 1988-05-22 | Sugarland |
| Research | 5 | 333445555 | 1988-05-22 | Houston |
| Administration | 4 | 987654321 | 1995-01-01 | Stafford |
| Headquarters | 1 | 888665555 | 1981-06-19 | Houston |

1. **Compare the results of question e and f**

* Ở phép tích Descartes **×**, số cột ở bảng kết quả bằng tổng số cột ở các bảng thành phần (kể cả các cột trùng tên, nó phải đổi tên thành tên khác để phân biệt). Ngược lại, số cột ở bảng kết quả của phép ghép tự nhiên ⋈, là tổng số cột ở các bảng thành phần, có loại đi những cột trùng tên. Ở ghép tự nhiên ⋈, phải có ít nhất 1 hàng ở bảng này trùng tên với 1 hàng ở bảng kia.
* Số hàng của bảng kết quả phép ghép tự nhiên ⋈ luôn ít hơn số hàng của bảng kết quả phép tích Descartes **×**.
* Các hàng của phép ghép tự nhiên ⋈ được chọn từ các hàng của phép tích Descartes **×** với điều kiện là giá trị ở các cột trùng tên phải như nhau.

1. **Employee – Dependent**

* Câu lệnh trên không đúng.
* Vì về cấu trúc, 2 bảng trên rất khác nhau. Bảng EMPLOYEE gồm 10 thuộc tính (Fname, Minit, Lname Ssn, Bdate, Address, Sex, Salary, Super\_ssn, Dno), còn bảng DEPENDENT gồm 5 thuộc tính (Essn, Dependent\_name, Sex, Bdate, Relationship). Do đó, ta không thể thực hiện phép hiệu này.

1. Π**Ssn(Employee) -** Π**Essn(Dependent)**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Câu lệnh trên đúng. Vì bảng EMPLOYEE có cột Ssn, bảng DEPENDENT có cột Essn, nên có thể thực hiện các phép chiếu. Mỗi phép chiếu cho ra 1 bảng gồm 1 thuộc tính duy nhất, có cùng miền giá trị (là Ssn và Essn). Do đó, có thể thực hiện phép hiệu. | |  | | --- | | **Ssn** | | 999887777 | | 666884444 | | 453453453 | | 987987987 | | 888665555 | |

1. σ**Sex = F and Salary > 30000(Employee)** **∪** σ**Sex = M and Salary < 30000(Employee)**

* Câu lệnh trên đúng.
* Ở phép chọn σ**Sex = F and Salary > 30000(Employee),** kết quả là một bộ tương ứng với nữ nhân viên có lương trên 30000 (Jennifer S. Wallace). Ở phép chọn σ**Sex = M and Salary < 30000(Employee),** kết quả là một bộ tương ứng với nam nhân viên có lương dưới 30000 (Ahmad V. Jabbar).

≫ Kết quả cuối cùng là hợp của 2 phép chọn trên: hiển thị tất cả thuộc tính có trong bảng EMPLOYEE của 2 nhân viên là Jennifer S. Wallace và Ahmad V. Jabbar.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Fname** | **Minit** | **Lname** | **Ssn** | **Bdate** | **Address** | **Sex** | **Salary** | **Super\_ssn** | **Dno** |
| Jennifer | S | Wallace | 987654321 | 1941-06-20 | 291 Berry, Bellaire, TX | F | 43000 | 888665555 | 4 |
| Ahmad | V | Jabbar | 987987987 | 1969-03-29 | 980 Dallas, Houston, TX | M | 25000 | 987654321 | 4 |

1. Π**Ssn(**σ**Dno=5(Employee)) ∩** ρ**Ssn(**Π**Essn(**σ**Pno =1(Works\_on)))**

* Câu lệnh trên đúng.
* Đầu tiên là các phép tính thành phần:
  + Phép chọn σ**Dno=5(Employee)** đúng cú pháp (bảng EMPLOYEE có thuộc tính Dno).

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Fname** | **Minit** | **Lname** | **Ssn** | **Bdate** | **Address** | **Sex** | **Salary** | **Super\_ssn** | **Dno** |
| John | B | Smith | 123456789 | 1965-01-09 | 731 Fondren, Houston, TX | M | 30000 | 333445555 | 5 |
| Franklin | T | Wong | 333445555 | 1955-12-08 | 638 Voss, Houston, TX | M | 40000 | 888665555 | 5 |
| Ramesh | K | Narayan | 666884444 | 1962-09-15 | 975 Fire Oak, Humble, TX | M | 38000 | 333445555 | 5 |
| Joyce | A | English | 453453453 | 1972-07-31 | 5631 Rice, Houston, TX | F | 25000 | 333445555 | 5 |

* + Phép chiếu Π**Ssn(**σ**Dno=5(Employee))** đúng cú pháp (kết quả phép chọn ở trên có Ssn).

|  |
| --- |
| **Ssn** |
| 123456789 |
| 333445555 |
| 666884444 |
| 453453453 |

|  |  |  |
| --- | --- | --- |
| σ**Pno=1(Works\_on)**  Phép chọn đúng cú pháp (WORKS\_ON có cột Pno): | Π**Essn(**σ**Pno=1(Works\_on))**  Phép chiếu đúng cú pháp (kết quả phép chọn trên có Essn): | ρ**Ssn(**Π**Essn(**σ**Pno=1(Works\_on)))**  Phép đổi tên đúng cú pháp: |
| |  |  |  | | --- | --- | --- | | **Essn** | **Pno** | **Hours** | | 123456789 | 1 | 32.5 | | 453453453 | 1 | 20.0 | | |  | | --- | | **Essn** | | 123456789 | | 453453453 | | |  | | --- | | **Ssn** | | 123456789 | | 453453453 | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ≫ Kết luận:  Phép giao cho ra kết quả là các hàng có trùng giá trị ở 2 bảng: | |  | | --- | | **Ssn** | | 123456789 | | 453453453 | |

1. **Explain the sense of the result of question j and k.**

* Để có thể thực hiện các phép **∪** và **∩** thì các bảng kết quả trung gian khi thực hiện phải có cùng cấu trúc với nhau.

1. **Given a University database as follows:**

* ***Identify the column or the group of columns in the other tables that must be unique across all rows in the table*.**

|  |  |
| --- | --- |
| **Relational schema** | **Các cột có ràng buộc unique** |
| STUDENT | Student\_number |
| COURSE | Course\_name, Course\_number |
| PREREQUISITE | Prerequisite\_number, Course\_number |
| SECTION | Section\_identifier, Course\_number, Semester, Year |
| REPORT | Student\_number, Section\_identifier |

* ***Identify the primary key, foreign key in each relational schema.***

|  |  |  |
| --- | --- | --- |
| **Relational schema** | **Primary key** | **Foreign key** |
| STUDENT | Student\_number |  |
| COURSE | Course\_number |  |
| PREREQUISITE | Prerequisite\_number | Course\_number tham chiếu đến cột Course\_number của bảng COURSE |
| SECTION | Section\_identifier | Course\_number (tham chiếu đến cột Course\_number của bảng COURSE) |
| GRADE\_REPORT | Grade | * Student\_number (tham chiếu đến cột Student\_number của bảng STUDENT) * Section\_identifier (tham chiếu đến cột Section\_identifier của bảng SECTION) |

* ***Specify the following queries on the University database schema:***
  1. **Retrieve the student\_number and the name of all students in the university.**
  2. **Retrieve the student\_number and the name of all students in “Computer Science” major.**
  3. **Retrieve the prerequisite course number and course name of the course which has course number “1111’’.**
  4. **Retrieve the section identifier, course number and course name of courses which are opened in semester 1, year 2019.**

Π Section\_identifier, Course\_number(σSemester = 1 and Year = *2019*(SECTION)) |X| Π Course\_name, Course\_number (COURSE)

* 1. **Retrieve the section identifier, course number and course name of courses which are not opened in semester 1, year 2019.**

Π Section\_identifier, Course\_number(SECTION - σSemester = 1 and Year = *2019*(SECTION)) |X|

Π Course\_name, Course\_number (COURSE)

* 1. **Retrieve the section identifier, course number and course name of courses which are opened in year 2018 or 2019.**

Π Section\_identifier, Course\_number(σYear = 2018 or Year = *2019*(SECTION)) |X| Π Course\_name, Course\_number (COURSE)

* 1. **Retrieve the section identifier, course number and course name of courses which are opened in year 2018 and 2019.**

Π Section\_identifier, Course\_number(σYear = 2018(SECTION) ∩ σYear = *2019*(SECTION)) |X| Π Course\_name, Course\_number (COURSE)