1. **Normalization**

* Là một process phân tích lược đồ quan hệ được cho dựa trên các phụ thuộc hàm và các khóa (primary) để đạt được những đặc tính sau:

+ Giảm thiểu dư thừa dữ liệu

+ Giảm thiểu những kỳ dị dữ liệu khi insert, delete, và update

* Dạng chuẩn của một quan hệ: là điều kiện chuẩn cao nhất mà quan hệ đó đạt được
* Quá trình chuẩn hóa thông qua phân rã sinh ra lược độ quan hệ mới phải đảm bảo:

+ The lossless join (*kết không có mất thông tin*) hoặc là non-additive join (*Kết không có thêm thông tin*). -> đặc tính bảo toàn thông tin

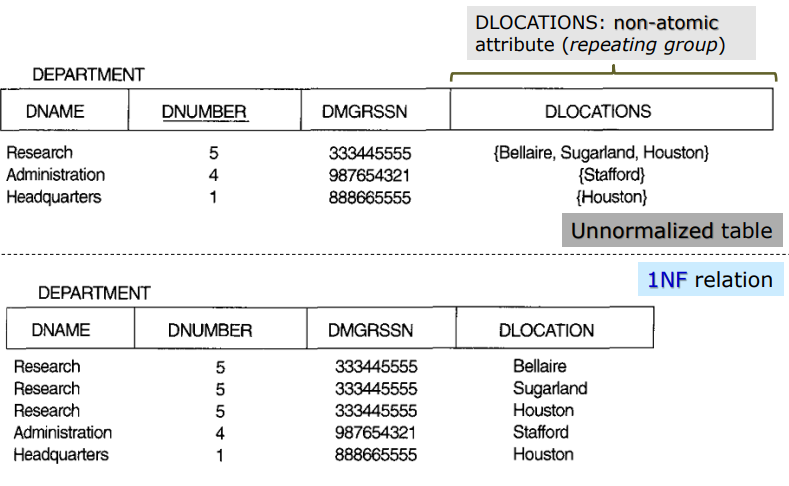
+ The dependency preservation property: đặc tính bảo toàn phụ thuộc hàm

* **Normal Form Based on Primary Keys**

|  |  |  |
| --- | --- | --- |
| **Normal Form** | **Test** | **Remedy (Normaliztion)** |
| First (1NF) | Relation phải thỏa mãn là atomic attribute và không có quan hệ lồng nhau | Hình thành các relations mới cho mỗi nonatomic attribute hoặc là những quan hệ lồng nhau |
| Second (2NF) | Các Relations mà primary key chứa nhiều attributes, không có thuộc tính không khóa nào phụ thuộc bán phần vào primary key | Phân rả và hình thành relation mới cho mỗi partial key với những attributes phụ thuộc nó. Phải đảm bảo giữ original primary key và tất cả những thuộc tính phụ thuộc hoàn toàn vào nó |
| Third (3NF) | Relation không được có thuộc tính không khóa phụ thuộc vào một thuộc tính không khóa khác (hoặc là một tập các thuộc tính không khóa). Không có transitive dependency của thuộc tính không khóa trên khóa chính | Phân rả và hình thành quan hệ mới bao gồm các thuộc tính không khóa mà nó xác định cho các thuộc tính không khóa khác |

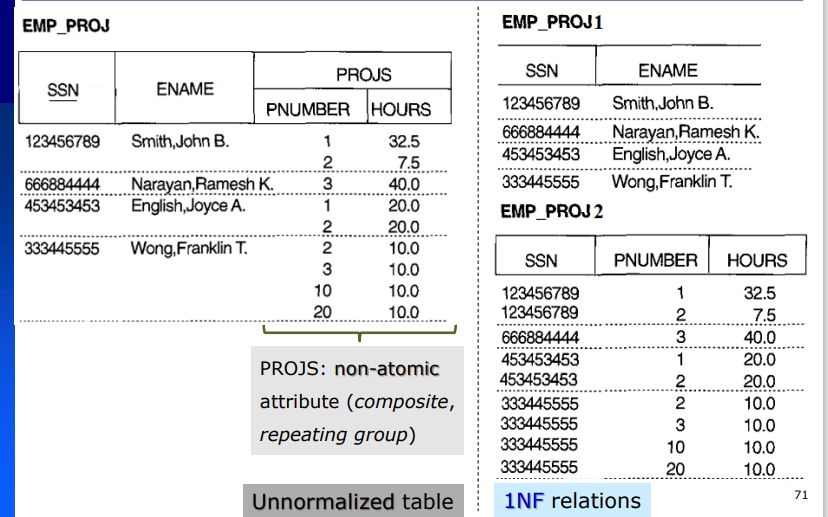
* **Pure relations dựa trên mô hình relational data đều là** ***1NF***
* **Atomic attributes**: không có nhóm lặp hay quan hệ lồng nhau

Ví dụ 1: cho trường hợp không thỏa mãn 1NF:



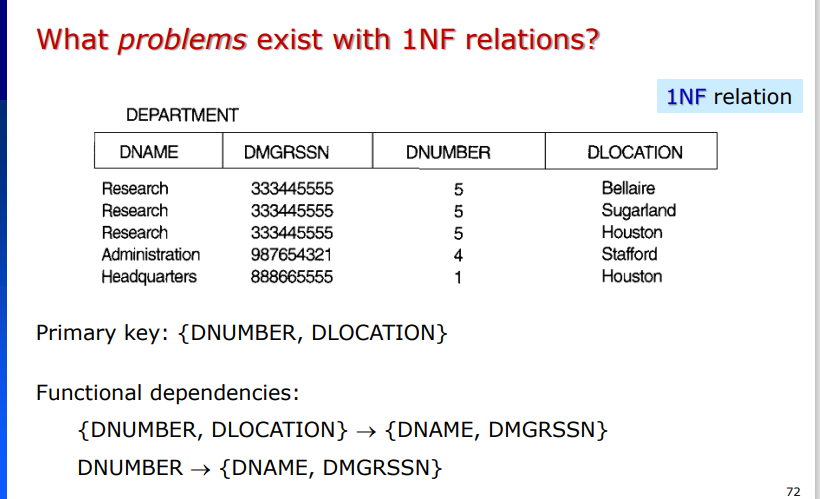
* Ở thuộc tính DLOCATION ở relation ban đầu chứa những giá tị tập hợp không atomic. Chuẩn hóa bằng cách tách mỗi giá trị trong Dlocation thành một hàng mới.

Ví dụ 2: cho trường hợp không thỏa mãn 1NF:



* Trong quan hệ EMP\_PROJ, attribute PROJS là một composite attribute (thuộc tính gộp), nên được tách ra thành một relation mới khi mapping với mối quan hệ là 1-N
* **Relations in *2NF* tất cả thuộc tính không khóa phải phụ thuộc hàm trên *whole key.***
* Atomic attributes: không có nhóm lặp hay quan hệ lồng nhau
* Thuộc tính không khóa KHÔNG có phụ thuộc hàm riêng phần vào khóa

Ví dụ: trường hơp không thỏa mãn 2NF



* Với bảng DEPARTMENT trên, đã thỏa mãn 1NF tất cả các attribute đều là atomic.
* Nhưng có phụ thuộc hàm bán phần vào khóa:

+ Primary key: {DNUMBER, DLOCATION}

+ {DNUMBER, DLOCATION} -> {DNAME, DMGRSSN}

+ DNUMBER -> {DNAME, DMGRSSN}: DNUMBER là key-attrinute là con của primary key không phải là key nên là phụ thuộc hàm bán phần vào khóa.

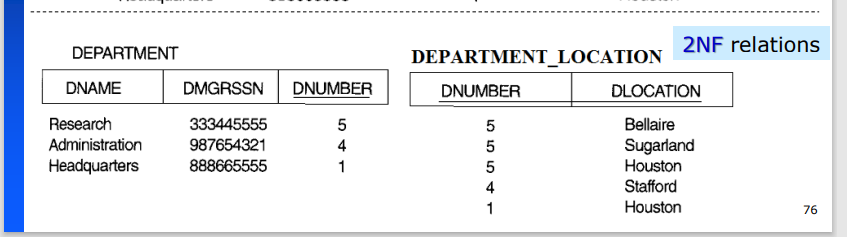
* Chính vị vậy mà có thể thấy DNAME và DMGRSSN sẽ có nhiều hàng lặp lại giá trị -> redundancy.
* Sẽ gây ra một số kỳ dị dữ liệu khi thao tác Insert, Delete, Modify

+ **Modification anomalies**: DMGRSSN of Department 5 to 123456789? DNAME of Department 5 to “Research & Development”? -> phải điều chỉnh trên nhiều rows của table. Nếu như table chứa hàng triệu records thì việc sửa đổi này tốn thời thời gian và chi phí đáng kể.

+ **Insertion anomalies**: (4, Houston)? (Sales, 123456789, 10)? -> khi muốn thêm một một vị trí khác cho phòng ban Administration (4, Houston) lại phải ghi lại DNAME, DMGRSSN -> redundancy. Và khi muốn thêm một phòng ban nhưng chưa có vị trí sẽ đặt phòng ban đó ở đâu (Sales, 123456789, 10)? Thì không thể insert được.

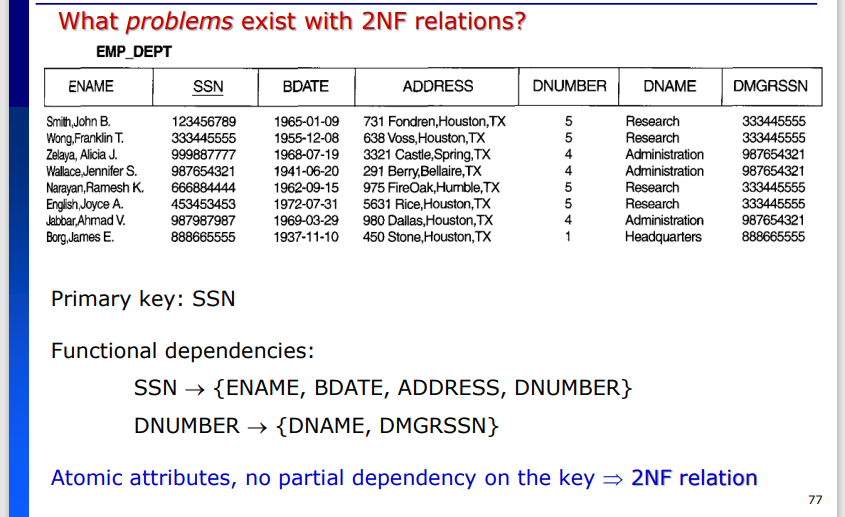
+ **Deletion anomalies:** (1, Houston)? (Research, 333445555, 5)? -> Khi muốn xóa vị trí của một phòng ban trong bảng thì dữ liệu về phòng ban đó (DNAME, DMGRSSN – mã SSN người quản lý phòng ban) bị mất luôn.

* Chính vì thế mà phải tách ra một new relation mới

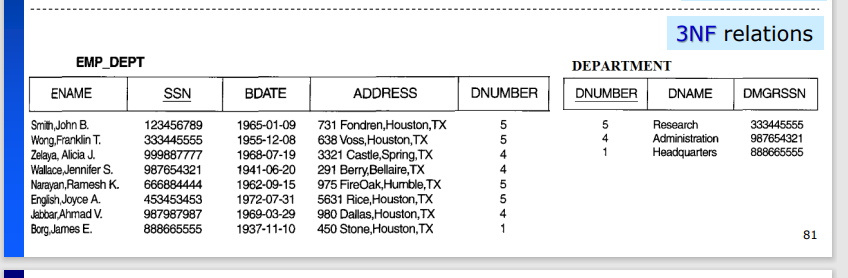
****

* Relations in ***3NF*** các thuộc tính không khóa **chỉ** phụ thuộc hàm vào **whole key**
* Atomic attributes: không có nhóm lặp hay quan hệ lồng nhau
* Thuộc tính không khóa KHÔNG có phụ thuộc hàm riêng phần vào khóa
* Thuộc tính không khóa KHÔNG phụ thuộc hàm bắc cầu vào khóa

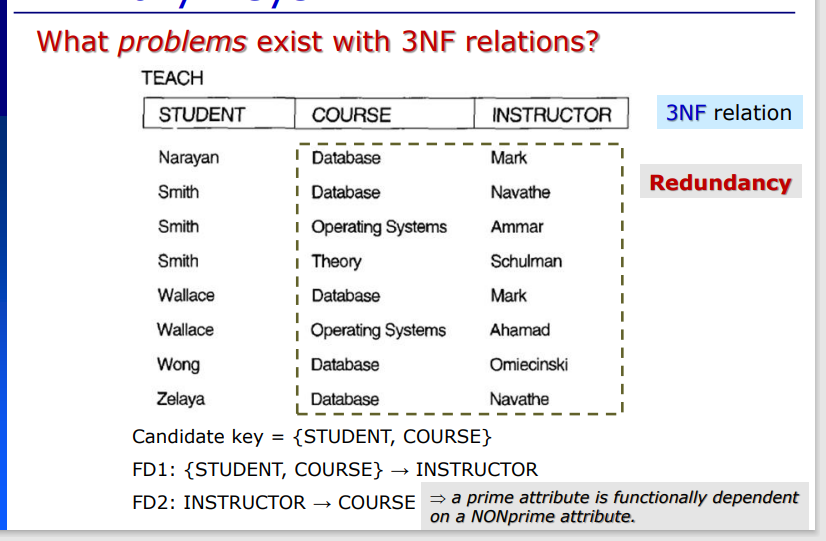
Ví dụ về trường hợp không thỏa 3NF:

****

* Với bảng trên có thể thấy là thỏa mãn 2NF. Tuy nhiên có phụ thuộc hàm bắt cầu (thuộc tính không khóa phụ thuộc hàm vào một thuộc tính không khóa khác) là DNUMBER -> {DNAME, DMGRSSN}.
* Có thể thấy dư thừa dữ liệu trên thuộc tính DNAME, DMGRSSN, gây ra những kì dị dữ liệu khi thao tác dữ liệu Insert, Delete và modificate
* Chính vì thế nên tách phụ thuộc hàm bắc cầu thành một relation mới

****

* **Tuy nhiên vẫn có một vài vấn đề với 3NF, có thể tham khảo table sau:**

****

* Đó chính là thuộc tính khóa phụ thuộc hàm vào một thuộc tính không khóa: INTRUCTOR -> COURSE
* Cũng sẽ gây ra những kỳ dị dữ liệu khi thao tác dữ liệu insert, delete, modify:

+ Insertion anomalies: Elmasri teaches Database? -> không thể insert vì thuộc tính Student not null (thuộc tính khóa - prime)

+ Deletion anomalies: Wong studies Database? -> mất luôn dữ liệu Omiecinski dạy môn Database

+ Modification anomalies: Mark teaches Algorithms instead of Database?

* **Boyce-Codd Normal Form**
* Một giản đơn giản hơn 3NF, nhưng lại chặt chẽ hơn 3NF
* Mỗi relation trong BCNF thì cũng là 3NF, tuy nheien một relation tỏng 3NF thì không nhất thiết là BCNF.

+ Cho một relation trong BCNF, mỗi thuộc tính không khóa/khóa phải ***phụ thuộc hàm vào superkeys***:

* Mỗi X->A trong R và X là một siêu khóa -> R in BCNF
* Bất cứ X->A nào trong R mà X không phải là siêu khóa -> R không phải là BCNF
* **Đặc tính bảo toàn phụ thuộc hàm khi phân rã**
* Mỗi phụ thuộc hàm X->Y xách định trong F phải xuất hiện trực tiếp một trong các lược đồ relation schemas Ri trong phân rã D hoặc có thể **suy diễn** từ các phụ thuộc hàm xuất hiện trong các Ri
* Nếu như phân rã không bảo toàn phụ thuộc, một vài phụ thuộc sẽ mất khi phân rã

+ Để kiểm tra, kết các relations lại để có quan hệ bao gồm tất cả thuộc tính bên trái và bên phải của phụ thuộc hàm bị mất, và kiểm tra xem phụ thuộc hàm đó có xuất hiện trong kết quả phép kết không.

**Cái ở trên là lý thuyết mà em học từ trường về Normalization, dĩ nhiên em đã có kiến thức để tìm tất cả khóa, tìm siêu khóa tối tiểu từ các phụ thuộc hàm, tìm tập phủ tối tiểu. Ngoài ra còn một số giải thuật để đảm bảo bảo toàn phụ thuộc hàm và chuyển đổi từ 3NF sang BCNF.**