**Phủ và các vấn đề liên quan**

**Thuật toán Member**

* Khi bao đóng của 1 tập các thuộc tính X được tạo ra, ta dễ dàng kiểm tra được X 🡪 Y có thuộc F+ hay không
* Thuật toán Member sẽ xác định tập các phụ thuộc hàm F có suy diễn ra 1 phụ thuộc hàm

X 🡪 Y nào đó hay không

* Đầu vào là tập các phụ thuộc hàm F và 1 phụ thuộc hàm X 🡪 Y
* Đầu ra : nếu X 🡪 Y được suy diễn F thì trả về true, ngược lại trả về falseText

  Description automatically generated

**Phủ và sự tương đương**

* Một tập phụ thuộc hàm F được phủ bởi một tập phụ thuộc hàm G ( hay nói cách khác G phủ F ) nếu mọi phụ thuộc hàm trong F đều nằm trong G+

=> F được phủ nếu mọi phụ thuộc hàm trong có thể được suy diễn từ G

* Hại tập phụ thuộc hàm F và G là tương đương nếu F+ = G+

=> Mọi phụ thuộc hàm trong F có thể được suy diễn từ G và mọi phụ thuộc hàm trong G có thể được suy diễn từ F

* Để xác định xem G có phủ F hay không: tính X+ trên G cho mỗi phụ thuộc hàm X 🡪 Y trong F, nếu Y thuộc X+ cho mọi X 🡪 Y thì G phủ F

**Phủ không dư thừa**

* Một tập các phụ thuộc hàm G được gọi là không dư thừa nếu như không có tập con G’ nào của G thỏa mãn G’ tương đương G
* Nếu có tập con như vậy thì G được gọi là dư thừa
* G là 1 phủ không dư thừa của F khi G không dư thừa và G phủ F

**Thuật toán xác định phủ không dư thừa : trả về một phủ không dư thừa**

Text

Description automatically generated

Giải thích :

* Một tập các phụ thuộc hàm F giống G
* Với mỗi phụ thuộc hàm X 🡪 Y của G
* Nếu Y thuộc X+ ( với X+ được suy diễn ra từ F’ = F – { X 🡪 Y }
* Member = true
* F sẽ bỏ đi X 🡪 Y và tiếp tục với các phụ thuộc hàm còn lại

Text

Description automatically generated with medium confidence

Chú ý : Nếu thứ tự các phụ thuộc hàm trong G khác với ví dụ trên thì kết quả của F thu được cũng sẽ khác ( nguyên nhân là do khi tìm thấy phụ thuộc hàm nào dư thừa ta sẽ bỏ đi phụ thuộc hàm đó và điều này ảnh hưởng đến việc tìm bao đóng của các X trong các phụ thuộc hàm X 🡪 Y sau đó).

Điều này cũng chứng tỏ rằng với 1 tập các phụ thuộc hàm cho trước ta có thể có nhiều hơn 1 phủ không dư thừa. Cũng có trường hợp, trong đó các phủ không dư thừa của 1 tập các phụ thuộc hàm G chứa những phụ thuộc hàm không nằm trong G.

Vd: với G = { A 🡪 B, B 🡪 A, B 🡪 C, A 🡪 C} thì F = { A 🡪 B, B 🡪 A, AB 🡪 C } cũng là 1 phủ không dư thừa của G

**Các thuộc tính dư thừa**

* Nếu F là 1 tập các phụ thuộc hàm không dư thừa, F sẽ không thể thu nhỏ hơn bằng cách loại bỏ các phụ thuộc hàm
* Tuy nhiên vẫn có một cách để giảm kích cỡ của F, đó là loại bỏ các thuộc tính từ các phụ thuộc hàm trong F
* Thuộc tính A được gọi là dư thừa trong phụ thuộc hàm X 🡪 Y nếu:

A picture containing text

Description automatically generated

Nói cách khác A bị coi là dư thừa nếu như bỏ đi A cũng không làm thay đổi F+

**Tập phụ thuộc hàm tối giản trái và tối giản phải**

* X 🡪 Y được gọi là tối giản trái nếu X không chứa thuộc tính dư thừa
* X 🡪 Y được gọi là tối phải trái nếu Y không chứa thuộc tính dư thừa

**Từ đây ta có thuật toán Left – Reduce trả về phiên bản tối giản trái của phủ F**

Text

Description automatically generated

**Và thuật toán tối giản phải để trả về phiên bản tối giản phải của phủ F**

Text, letter

Description automatically generated

**Thuật toán tối giản phụ thuộc hàm : trả về một phụ thuộc hàm tối giản ( cả trái cả phải )**

**Text, letter

Description automatically generated**

Chú ý : thứ tự thực hiện trong thuật toán Reduce cần đảm bảo tối giản trái trước rồi mới tối giản phải. Điều này rất quan trọng, nếu làm ngược lại, F cuối cùng thu được có thể không tối giản phải

**Phủ tối thiểu : một tập phụ thuộc hàm được coi là tối thiểu khi**

* Mọi phụ thuộc hàm trong tập đều có vế phải là một thuộc tính
* F là không dư thừa
* F tối giản trái

**Thuật toán tìm phủ tối thiếu**

**Graphical user interface, text, application, email

Description automatically generated**

**Chuẩn hóa**

1. Các khái niệm cần lưu ý

* Thuộc tính nguyên tố: là những thuộc tính chứa giá trị đơn và không thể phân rã
* Các quan hệ chưa được chuẩn hóa đồng nghĩa với việc chúng chưa ở dạng chuẩn 1
* Các quan hệ chưa ở dạng chuẩn 1 là các quan hệ có 1 hoặc 1 vài thuộc tính chưa nguyên tố, hoặc bị lặp thuộc tính hoặc thuộc tính là thuộc tính dẫn xuất
* Khóa

+ Siêu khóa: là tập các thuộc tính giúp xác định duy nhất 1 thực thể

Vd : {id, tên} và {id} đều có thể gọi là siêu khóa vì đều có thể xác định được thực thể sinh viên

+ Khóa: là tập các thuộc tính vừa đủ để xác định duy nhất 1 thực thể, nó khác siêu khóa ở chỗ, các thuộc tính trong khóa không có thuộc tính nào dư thừa

Vd : {id} là khóa còn {id, tên} thì không vì chỉ cần id ta đã có thể xác định được 1 thực thể sinh viên, không nhất thiết cần thêm tên

+ Khóa dự bị: là tập hợp gồm các khóa

+ Khóa chính: từ các khóa dự bị, người thiết kế CSDL chọn ra một khóa làm khóa chính

+ Thuộc tính khóa: là thuộc tính của 1 khóa dự bị nào đó

+ Thuộc tính không khóa: ngược lại

1. Các dạng chuẩn hóa
   1. Dạng chuẩn 1 (1NF)

* Moi giá trị thuộc tính phải là nguyên tố
* Không có thuộc tính đa trị
* Không có thuộc tính dẫn xuất ( không có thuộc tính nào được tính toán từ một hay nhiều thuộc tính khác )
  1. Dạng chuẩn 2 (2NF)

Phụ thuộc hàm đầy đủ: 1 phụ thuộc hàm X 🡪 Y được gọi là đầy đủ nếu ta loại bỏ bất kì thuộc tính A (A là thuộc tính trong tập các thuộc tính cấu tạo nên X) nào đi thì X 🡪 Y sẽ sai.

Lúc này X 🡪 Y được gọi là tối giản hay tối giản trái( khái niệm tối giản trái sẽ giải thích sau )

* Là 1NF
* Các thuộc tính không khóa phụ thuộc hàm đầy đủ vào thuộc tính khóa chính

**Chuẩn hóa từ 1NF sang 2NF:**

* Loại bỏ các thuộc tính không khóa phụ thuộc 1 phần vào khóa chính để tách thành bảng riêng. Bảng này sẽ có khóa chính là bộ phận khóa mà chúng phụ thuộc
* Các thuộc tính còn lại thành một bảng với khóa là khóa chính ban đầu

VD : Cho R = { A, D, P, G} và F = { AD 🡪 P, A 🡪 G} với k = {AD} là 1 khóa

R đã ở dạng 1NF

R chưa ở dạng 2NF vì G ( thuộc tính không khóa ) không phụ thuộc đầy đủ vào khóa AD

Quá trình chuẩn hóa diễn ra như sau

Phân rã R thành

R1 = { A, G} , F1 = { A 🡪 G } với k1 = {A}

R2 = { A, D, P} , F2 = { AD 🡪 P } với k2 = {AD}

* 1. Dạng chuẩn 3 (3NF)
* Là 2 NF
* Tất cả các thuộc tính không khóa phải được suy ra trực tiếp từ các thuộc tính khóa

**Chuẩn hóa từ 2NF sang 3NF:**

Graphical user interface, text, application, email

Description automatically generated

**Ví dụ**

* B1: Tìm toàn bộ khoá dự bị
* B2: Sử dụng thuật toán tìm phủ tối thiểu 🡺 output: tập phụ thuộc hàm F\*
* B3: Thực hiện phân tách với F\* vừa tính được
  + Gộp các phụ thuộc hàm có cùng vế trái trong F\* (luật suy diễn IR5: cộng thêm)
  + Với mỗi phụ thuộc hàm trong F\* lập một bảng mới
  + Nếu trong số các bảng vừa lập không có bảng nào chứa bất kỳ 1 khoá dự bị nào (ở B1) thì phải tạo thêm bảng mới có chứa 1 khoá dự bị nào đó

VD minh hoạ: Text

Description automatically generated

B1: Tìm các khoá dự bị

{B}+ = {B}, {C}+ = {C}, {D}+ = {D}, {E}+ = {E, D, B, C, G}, {G}+ = {G}

Vậy khoá dự bị là E

B2: Thuật toán tìm phủ tối thiểu:

- Phân tách các phụ thuộc hàm sao cho vế phải là thuộc tích đơn (luật suy diễn IR4: chiếu)

F = {BC 🡪 G, E 🡪 DE, E 🡪 B, E 🡪 CD, E 🡪 G}

Tách thành (loại bỏ trùng lặp và phụ thuộc hàm hiển nhiên):

F = {BC 🡪 G, E 🡪 D, E 🡪 B, E 🡪 C, E 🡪 G}

- Sử dụng thuật toán tối giản trái: chỉ áp dụng với phụ thuộc hàm có vế trái > 1 thuộc tính

+ BC 🡪 G:

- bỏ B: tính {C}+ = {C} => không có G nên B không thừa

- bỏ C: tính {B}+ = {B} => không có G nên C không thừa

Vậy F vẫn giữ nguyên

- Sử dụng thuật toán tìm phủ không dư thừa:

- bỏ {BC 🡪 G} tính {BC}+ với F = {E 🡪 D, E 🡪 B, E 🡪 C, E 🡪 G}:

{BC}+ = {B, C} => không chứa G nên BC 🡪 G không thừa

- bỏ {E 🡪 D} tính {E}+ với F = {BC 🡪 G, E 🡪 B, E 🡪 C, E 🡪 G}:

{E}+ = {E, B, C, G} => không chứa D nên E 🡪 D không thừa

- bỏ {E 🡪 B} tính {E}+ với F = {BC 🡪 G, E 🡪 D, E 🡪 C, E 🡪 G}:

{E}+ = {E, D, C, G} => không chứa B nên E 🡪 B không thừa

- bỏ {E 🡪 C} tính {E}+ với F = {BC 🡪 G, E 🡪 D, E 🡪 B, E 🡪 G}:

{E}+ = {E, D, B, G} => không chứa C nên E 🡪C không thừa

- bỏ {E 🡪 G} tính {E}+ với F = {BC 🡪 G, E 🡪 D, E 🡪 B, E 🡪C}:

{E}+ = {E, D, B, C, G} => chứa G nên E 🡪 G thừa -> bỏ {E 🡪 G} khỏi F

Cuối cùng thu được F\* = {BC 🡪 G, E 🡪 D, E 🡪 B, E 🡪 C}

B3: Thực hiện chuẩn hoá

- Gộp các phụ thuộc hàm cùng vế phải (luật cộng thêm) ta được F\* = {BC 🡪 G, E 🡪 BCD}

- Tách thành 2 bảng:

R1 = (BCG), K = {BC}, F1 = {BC 🡪 G}

R2 = (EBCD), K = {E}, F2 = {E 🡪 BCD}

Khoá dự bị E đã có trong R2 nên k cần tạo thêm bảng mới

Chữa bài ktra:

Text, letter

Description automatically generated

- Đã đạt chuẩn 1NF (thật ra phải nhìn dữ liệu thực tế mới biết được :v)

- Tìm các khoá dự bị:

{A}+ = {A}, {B}+ = {B}, {C}+ = {C}, {D}+ = {D, C, B}

{AB}+ = {A, B, C}, {AC}+ = {AC}, {AD}+ = {A, D, C, B}

{BC}+ = {B, C}, {BD}+ = {B, D, C}

{CD}+ = {C, D}

Vậy chỉ có 1 khoá dự bị AD

R không thuộc chuẩn 2NF vì có D 🡪 C, C phụ thuộc một phần vào khoá

Thực hiện thuật toán tách 3NF:

V2: Tìm phủ tối thiểu

- Thực hiện phân tách và loại bỏ phụ thuộc hàm hiển nhiên:

F = {CD 🡪 B, D 🡪 C, D 🡪 B, AB 🡪 C}

- Thực hiện thuật toán tối giản trái:

+ CD 🡪 B:

- bỏ C: tính {D}+ = {D, C, B} => có B nên C thừa -> bỏ C khỏi CD 🡪 B

- bỏ D: tính {C}+ = {C} => không có B nên D không thừa

+ AB 🡪 C:

- bỏ A: tính {B}+ = {B} => không có C nên A không thừa

- bỏ B: tính {A}+ = {A} => không có C nên B không thừa

Vậy F = {D 🡪 C, D 🡪 B, AB 🡪 C}

- Sử dụng thuật toán tìm phủ không dư thừa:

- bỏ {D 🡪 C} tính {D}+ với F = {D 🡪 B, AB 🡪 C}:

{D}+ = {D, B} => không chứa C nên D 🡪 C không thừa

- bỏ {D 🡪 B} tính {D}+ với F = {D 🡪 C, AB 🡪 C}:

{D}+ = {D, C} => không chứa B nên D 🡪 B không thừa

- bỏ {AB 🡪 C} tính {AB}+ với F = {D 🡪 C, D 🡪 B}:

{AB}+ = {A, B} => không chứa C nên AB 🡪 C không thừa

Cuối cùng thu được F\* = {D 🡪 C, D 🡪 B, AB 🡪 C}

B3: Thực hiện chuẩn hoá

- Gộp các phụ thuộc hàm cùng vế phải (luật cộng thêm) ta được F\* = {D 🡪 BC, AB 🡪 C}

- Tách thành 2 bảng:

R1 = (DBC), K = {D}, F1 = {D 🡪 BC}

R2 = (ABC), K = {AB}, F2 = {AB 🡪 C}

Khoá dự bị AD không có trong bảng nào nên cần tạo bảng mới

R3 = (AD), K = {AD}, F3 = {AD 🡪 AD}

- Kiểm tra đạt chuẩn:

- đã đạt 1NF

- các thuộc tính không khoá trong các bảng đã phụ thuộc hàm đầy đủ vào khoá => đạt 2NF

- các thuộc tính không khoá đã được suy ra trực tiếp từ thuộc tính khoá trong các bảng => đạt 3NF

- không có thuộc tính khoá nào phụ thuộc vào thuộc tính không khoá => đạt BCNF

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated