Nome: Davi Augusto Neves Leite

RA: 191027383

OBS: resolvido detalhadamente até o exercício 3. A partir do exercício 4, a resolução foi realizada a parte e colocado as principais anotações neste arquivo.

Resolução – Quarta Lista de Exercícios – Dependências Funcionais

1) As **chaves candidatas** são os atributos que, dadas as dependências funcionais, se localizam somente no lado **esquerdo** da relação. Além disso, é levado em conta os atributos **isolados** (certamente chaves candidatas), realizando uma **união** entre os **isolados** com os da **esquerda**.

Dessa forma, dado o conjunto de dependências funcionais F, tem-se as seguintes chaves candidatas: {ISBN, Autor}; tendo "ISBN" um atributo presente somente ao lado esquerdo das dependências funcionais e "Autor" um atributo isolado. Com esse conjunto {ISBN, Autor}, é possível encontrar **todos** os outros atributos de Livro.

2) Para encontrar um conjunto (ou cobertura) mínimo basta realizar quatro passos: decompor F com vários atributos do lado direito; eliminar as dependências funcionais triviais (aquelas que levam nela mesmo, por exemplo A->A ou DA->A); eliminar os atributos redundantes do lado esquerdo de cada dependência funcional restante e; eliminar as dependências funcionais implicadas logicamente por outras presentes no conjunto restante.

Dessa forma:

I) F' = {G->D, C->A, BC->G, CD->B, CD->G, CE->A, CE->E, ACG->B}

II) F'' = {G->D, C->A, BC->G, CD->B, CD->G, CE->A, ACG->B}

III) ACG->B (C->A)

F''' = {G->D, C->A, BC->G, CD->B, CD->G, CE->A, **CG->B**}

IV) CE->A (C->A => AE->A, trivial);

CD->G (CD->B => B->G ==> BC->G => trivial);

CG->B (G->D e CD->B)

Portanto, um conjunto mínimo para F é: G = {G->D, C->A, BC->G, CD->B}

3)

A)

Outra maneira de encontrar **um conjunto mínimo é aplicando o algoritmo** descrito por Navathe (Fundamentos do Sistema de Banco de Dados) que consiste em:

- Entrada(s): um conjunto de dependências funcionais E.

1) Defina F = E.
2) Substitua cada dependência funcional X → {A1, A2, ..., An} em F pelas n dependências funcionais X →A1, X →A2, ..., X → An.
3) Para cada dependência funcional X → A em F para cada atributo B que é um elemento de X
   se {{F – {X → A}} ∪ {(X – {B}) → A}} for equivalente a F
   então substitua X → A por (X – {B}) → A em F.
4) Para cada dependência funcional restante X → A em F
   se {F – {X → A}} for equivalente a F,
   então remova X → A de F.

Dessa forma, considerando o conjunto de dependências funcionais F como entrada, tem-se:

1) Vamos chamar de G a estrutura. Ou seja: G = F.
2) Não há atributos a serem decompostos (por exemplo: A->AB => A->A e A->B)
3) Até então: G = {A->B, B->A, B->C, C->A, C->B, C->D}
   Realizando o passo 3 (desconsidera-se a dependência funcional ao realizar a operação):
   A->B (é desconsiderado ao lado direito) => A+ = A; como não há B (pela relação), então B não pode ser removido
   B->A => B+ = ABCD; como há A (pela relação), então A pode ser removido
   B->C => B+ = B; como não há C, C não pode ser removido
   C->A => C+ = BCD; como não há A, A não pode ser removido
   C->B => C+ = ABCD; como há B, B pode ser removido
   C->D => C+ = ABC; como não há D, D não pode ser removido
   Dessa forma: G = {A->B, B->, B->C, C->A, C->, C->D}
4) Removendo as vazias, tem-se: G = {A->B, B->C, C->A, C->D}

Portanto, um conjunto mínimo para F é: G = {A->B, B->C, C->A, C->D}.

B)

Assim como no exercício anterior, Navathe apresentou **um algoritmo para determinar uma chave de uma relação R, dado um conjunto F de dependências funcionais**. O algoritmo é dado por:

- Entrada(s): uma relação R e um conjunto de dependências funcionais F nos atributos de R.

1) Defina CHAVE = R
2) Para cada atributo A em CHAVE:

   {calcule (CHAVE – A)+ em relação a F;

   se (CHAVE – A)+ contiver todos os atributos em R,

   então defina CHAVE = CHAVE – {A}};

Dessa forma, considerando a relação R dada na questão e o conjunto de dependências funcionais F como entradas, tem-se:

1) Vamos chamar de C a estrutura. Ou seja: C = R.
2) A+ = ABCD = R;
   B+ = ABCD = R;
   C+ = ABCD = R;
   D+ = NULL (não possui dependentes/não pode gerar ninguém).

Dessa forma, como A+ = B+ = C+ = ABCD (possuem todos os atributos dependentes deles), então as chaves candidatas de R são: A, B e C.

4)

A) G = {A->BCD}

B) Chave(s) candidata(s): A

5)

A) G = {A->B, B->C, C->D}

B) Chave(s) candidata(s): A

6)

A) G = {A->B, B->C, C->D, D->A}

B) Chave(s) candidata(s): A, B, C, D

7)

A) G = {A->B, C->D}

B) Chave(s) candidata(s): {A, C}

8)

A) G = {A->B, B->C, C->D}

B) Chave(s) candidata(s): A

9)

A) G = {A->B, B->C, C->D}

B) Chave(s) candidata(s): A

C) Levando em consideração o exercício 9, letra A e B, sabe-se que G = {A->B, B->C, C->D} e que A é a única chave candidata.

Diante disso, basta verificar a aplicabilidade das formas normais em R. Em outras palavras:

1) Primeira Forma Normal (1FN, a qual exprime que todos os atributos de R devem ser atômicos e monovalorados. Para normalizar para 1FN, basta separar os atributos multivalorados em tabelas separadas, ligadas pela chave primária): como R = {<u>A</u>, B, C, D} e, desta forma, não é possível estimar se os atributos são atômicos e monovalorados, suponha-se que R está na 1FN.
2) Segunda Forma Normal (2FN, a qual exprime que todos os atributos não-chave devem ser totalmente dependentes da chave primária. Para normalizar para 2FN, basta realizar uma decomposição de projeção sem perda de dados): como R está na 1FN, pode-se verificar se ele está na 2FN. Desta forma, sabendo que B, C, D são atributos não-chave (ou não primos), verifica-se:
    - A->B: válido, B depende totalmente de A

- B->C: não válido, pois C depende de um atributo não-chave
- C->D: não válido, pois D depende de um atributo não-chave.

Dessa forma, sabe-se que R está na 1FN apenas.

3) Terceira Forma Normal (3FN, a qual exprime que todos os atributos não-chave são mutuamente independentes, ou seja, quando nenhum deles é funcionalmente dependente de nenhuma combinação dos outros, exceto da chave primária. Mais uma vez, para normalizar para a 3FN deve-se realizar uma decomposição de projeção sem perda de dados): não é possível verificar, pois R não está na 2FN.
4) Forma Normal de Boyce-Codd (FNBC, a qual exprime que todo determinante deve ser chave candidata. Esse determinante nada mais é do que qualquer atributo do qual álbum outro atributo é funcionalmente dependente. Semelhante a 2FN e 3FN, para normalizar para FNBC basta realizar uma decomposição sem perdas): não é possível verificar, pois R não está na 3FN.

Portanto, a forma normal mais alta de R é a 1FN.

D) Tendo em vista o exercício anterior, a normalização de R é realizada da seguinte maneira:

1) Segunda Forma Normal (2FN): para cada subconjunto de atributos da chave primária, neste caso apenas A, deve-se gerar uma relação com esse subconjunto como sua chave primária, incluindo os atributos da relação original na relação correspondente à chave primária (A).

   R (<u>A</u>, B, C, D) será transformado em:

   ➔ R1 (<u>A</u>, B)
   ➔ R2 (<u>B</u>, C)
   ➔ R3 (<u>C</u>, D)

2) Terceira Forma Normal (3FN): neste caso, todos os atributos não-chave devem possuir dependência total e não transitiva da chave primária. Ou seja, o resultado obtivo acima já está na 3FN, uma vez que todos os atributos não-chave dependem diretamente da sua chave primária (B depende de A em R1; C depende de B em R2 e; D depende de C em R3).

3) Forma Normal de Boyce-Codd (FNBC): neste caso, todos os determinantes são chaves candidatas. Esses determinantes representam qualquer atributo do qual algum outro atributo é funcionalmente dependente. Dessa forma, o resultado obtido na 2FN, que está na 3FN, também está em FNBC, uma vez que todos os determinantes são chaves candidatas (novamente, em R1 tem-se B depende do determinante A que é a chave candidata; o mesmo ocorre em R2 e R3).

Portanto, a relação R está normalizada para FNBC com:

- ➔ R1 (A, B)
- ➔ R2 (B, C)
- ➔ R3 (C, D)