[MAC0426] Sistemas de Bancos de Dados
[IBI5013] Bancos de Dados para Bioinformática
Aula 19
Normalização de Bancos de Dados Relacionais

Kelly Rosa Braghetto

DCC–IME–USP

6 de maio de 2016

Objetivos implícitos no projeto de BDs Relacionais

Preservação da informação

- Preservação de todos os conceitos capturados no projeto conceitual

Redundância mínima

- Diminuição do armazenamento redundante de informações e redução da necessidade de múltiplas atualizações para manter a consistência

Como medir a qualidade de um projeto de BD?

Porque um conjunto de agrupamentos de atributos em esquemas de relação é melhor do que outros?

Boas práticas se relacionam a dois níveis:

- **Nível lógico** (ou **conceitual**) – como os usuários interpretam os esquemas de relação e o significado de seus atributos
- **Nível de implementação** – como as tuplas em uma relação são armazenadas e atualizadas

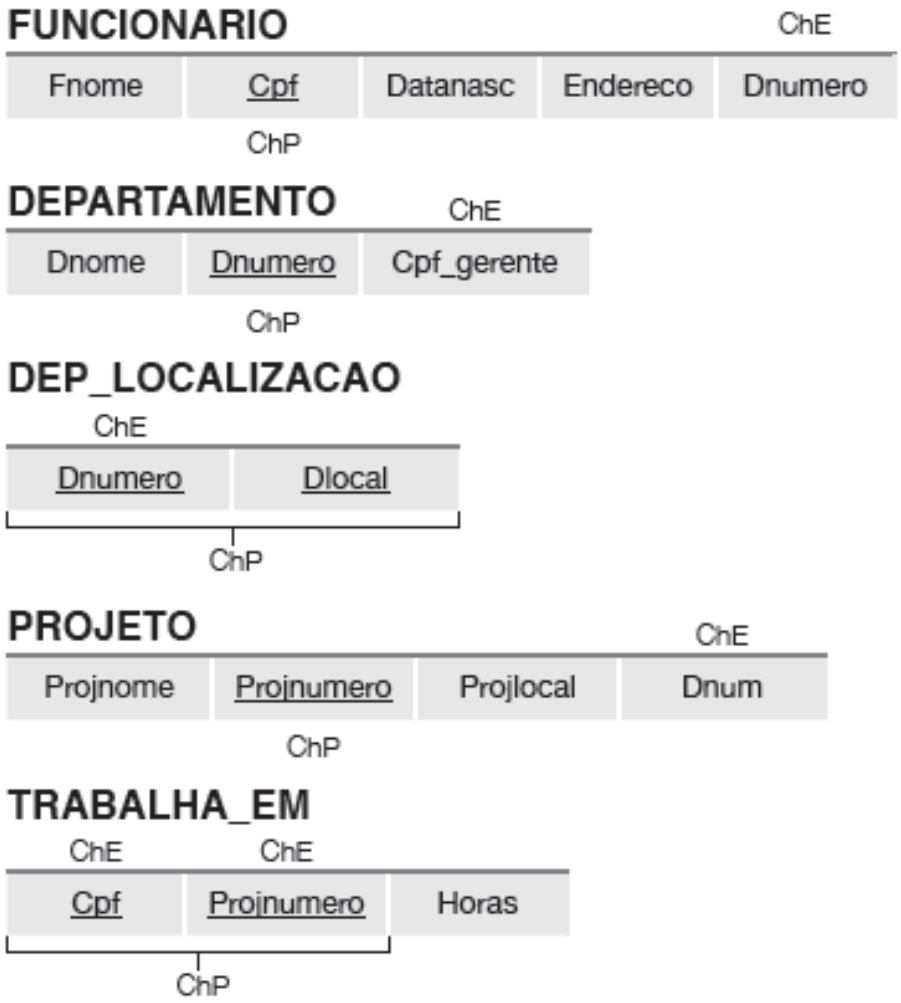
Diretrizes informais de projeto

Objetivos:

1. Garantir que a semântica dos atributos seja clara no esquema
2. Reduzir a informação redundante nas tuplas
3. Reduzir os valores NULL nas tuplas
4. Reprovar a possibilidade de gerar tuplas falsas

$\Rightarrow$ Consideram características que podem ser usadas como **medidas da qualidade** do projeto do esquema da relação.

Semântica dos atributos nas relações



"A facilidade com que o significado dos atributos de uma relação pode ser explicado é uma *medida informal* de quão bem a relação está projetada."

Semântica dos atributos nas relações

Diretriz 1

- Projete um esquema de relação de modo que seja fácil explicar seu significado
- Não combine atributos de vários tipos de entidade e de relacionamento em uma única relação
  - Se um esquema de relação corresponde a um só tipo de entidade ou de relacionamento, em geral é fácil explicar o seu significado
  - Caso contrário, poderá existir ambiguidades semânticas e a relação não poderá ser explicada com facilidade

Semântica dos atributos nas relações

Exemplo

(a)

FUNC_DEP

Fnome	Cpf	Datanasc	Endereco	Dnumero	Dnome	Cpf_gerente

(b)

FUNC_PROJ

Cpf	Projnumero	Horas	Fnome	Projnome	Projlocal

Nesses esquemas de relação, a semântica dos atributos é clara.
Entretanto, eles violam a diretriz 1 por misturar em uma só relação atributos de entidades diferentes do mundo real.

Informações redundantes

O armazenamento de junções naturais de relações (como feito no exemplo do *slide* anterior) traz dois problemas:

- Desperdício do espaço de armazenamento
- Anomalias de atualização – que podem ser: de inserção, de exclusão e de modificação

Anomalias de atualização

Anomalia de inserção (exemplo 1):

Como lidar com a inclusão de funcionários?

- Inserir uma tupla em `FUNC_DEP` colocando NULLs nos atributos relacionados ao departamento (se o funcionário ainda não tem departamento) OU
- Inserir uma tupla em `FUNC_DEP` colocando os valores de atributos do departamento onde o funcionário trabalha
  - Problema: os novos dados inseridos precisam ser coerentes com os dados já cadastrados sobre o departamento em outras tuplas de `FUNC_DEP`

(a)

FUNC_DEP

Fnome	Cpf	Datanasc	Endereco	Dnumero	Dnome	Cpf_gerente

Anomalias de atualização

Anomalia de inserção (exemplo 2):

Como incluir um departamento que ainda não tem funcionários?

- Inserir uma tupla com NULL nos atributos de funcionário viola a integridade de FUNC_DEP, porque Cpf é sua chave primária
- Além disso, depois que o primeiro funcionário for atribuído ao departamento, essa tupla com NULLs não será mais necessária.

(a)

FUNC_DEP

Fnome	Cpf	Datanasc	Endereco	Dnumero	Dnome	Cpf_gerente

Anomalias de atualização

Anomalia de exclusão:

- Se excluirmos de FUNC_DEP uma tupla de funcionário que represente o último funcionário trabalhando para um determinado departamento, a informação referente a esse departamento se perde do BD

Anomalia de modificação:

- Em FUNC_DEP, se mudarmos o valor de um dos atributos de um dado departamento, temos que atualizar as tuplas de todos os funcionários que trabalham nesse departamento
- Caso contrário, o BD ficará inconsistente

(a)

FUNC_DEP

Fnome	Cpf	Datanasc	Endereco	Dnumero	Dnome	Cpf_gerente

Informações redundantes e anomalias de atualização

Diretriz 2

- Projete os esquemas de relação de modo que nenhuma anomalia de inserção, exclusão ou modificação esteja presente nas relações
- Se houver alguma anomalia, anote-a claramente e cuide para que os programas que atualizam o BD operem corretamente

Também é importante lembrar que:

- Em alguns casos, as diretrizes 1 e 2 precisam ser violadas com o objetivo de melhorar o desempenho de algumas consultas
  Você consegue pensar em um exemplo de cenário onde isso pode acontecer?

Valores NULL nas tuplas

- Alguns projetos de esquemas agrupam muitos atributos em uma relação "gorda"
- Resultado: possibilidade de muitos NULLs nas tuplas da relação (quando os atributos não se aplicam a todas as duplas da relação)
- Problemas ocasionados: desperdício do espaço de armazenamento e impacto do NULL nos diferentes tipos de operações

Valores NULL nas tuplas

Exemplos de problemas em operações (sobre os quais já falamos inúmeras vezes!)

- Operações de SELEÇÃO e JUNÇÃO envolvem a comparação de valores. A comparação de NULL com outros valores pode gerar resultados imprevisíveis
- Definir um significado para o NULL em operações de agregação como SOMA ou CONTA

Valores NULL nas tuplas

Diretriz 3

- Sempre que possível, evite colocar atributos em uma relação cujos valores podem ser NULL com frequência
  - Você se lembra de como isso pode ser feito?
- Se eles forem inevitáveis, garanta que eles se apliquem apenas a casos excepcionais (e não à maioria das tuplas na relação)

Considerações prioritárias na decisão de incluir em uma relação atributos que podem conter valores NULL:

- Usar o espaço de modo eficaz
- Evitar junções com valores NULL

Valores NULL nas tuplas

Um atributo que pode ter valores NULL com muita frequência pode ser expresso em uma relação separada. Exemplo:

- Considere que apenas 15% dos funcionários têm escritórios individuais
  $\Rightarrow$ Essa porcentagem não justifica a inclusão de um atributo Numero_escritorio em FUNCIONARIO
- Solução: criar uma relação
  FUNC_ESCRITORIO(Fcpf, Numero_escritorio)
  para incluir nela apenas tuplas para funcionários que possuam escritórios individuais

Geração de tuplas falsas

Esquema de relação original:

FUNC_PROJ

Cpf	Projnumero	Horas	Fnome	Projnome	Projlocal

Esquemas de relação gerados a partir de uma decomposição do anterior:

(a)

FUNC_LOCAL

Fnome	Projlocal

ChP

(b)

FUNC_PROJ1

Cpf	Projnumero	Horas	Projnome	Projlocal

ChP

Esse segundo esquema é ruim, pois não nos permite recuperar a informação que havia originalmente em PROJ_FUNC. $\Rightarrow$ Ele gera **tuplas falsas**!

Geração de tuplas falsas

FUNC_PROJ

Cpf	ProjNumero	Horas	Fnome	ProjNome	ProjLocal
123	1	32,5	João	X	Itu
123	2	7,5	João	Y	São Paulo
543	1	20,0	Maria	X	Itu
543	3	20,0	Maria	Z	Mauá

FUNC_PROJ1

Cpf	ProjNumero	Horas	ProjNome	ProjLocal
123	1	32,5	X	Itu
123	2	7,5	Y	São Paulo
543	1	20,0	X	Itu
543	3	20,0	Z	Mauá

FUNC_LOCAL

Fnome	ProjLocal
João	Itu
João	São Paulo
Maria	Itu
Maria	Mauá

FUNC_PROJ1 NATURAL JOIN FUNC_LOCAL

Cpf	ProjNumero	Horas	ProjNome	ProjLocal	Fnome
123	1	32,5	X	Itu	João
123	1	32,5	X	Itu	Maria
123	2	7,5	Y	São Paulo	João
543	1	20,0	X	Itu	João
543	1	20,0	X	Itu	Maria
543	3	20,0	Z	Mauá	Maria

As tuplas em destaque são tuplas falsas!

Geração de tuplas falsas

Diretriz 4

- Projete esquemas de relação de modo que possam ser unidos com condições de igualdade sobre os atributos que são pares relacionados corretamente (chave primária, chave estrangeira), de modo a garantir que nenhuma tupla falsa seja gerada
- Evite relações com atributos correspondentes que não sejam combinações (chave primária, chave estrangeira), pois a junção sobre tais atributos pode produzir tuplas falsas

Dependência Funcional

- Um dos conceitos mais importantes na teoria do projeto de esquemas relacionais
- Ferramenta formal para a análise da qualidade de esquemas relacionais

Definição

*Uma **dependência funcional**, denotada por $X \rightarrow Y$, entre dois conjuntos de atributos X e Y de um esquema de relação R especifica uma **restrição** sobre possíveis tuplas que podem formar um estado de relação r de um esquema de relação R. A restrição é que, para quaisquer duas tuplas t_1 e t_2 em r que tenham $t_1[X] = t_2[X]$, elas também devem ter $t_1[Y] = t_2[Y]$.*

Dependência Funcional

Sinônimos para $X \rightarrow Y$:

- Os valores do componente Y de uma tupla em r são **determinados** pelos valores do componente X
- Os valores do componente X de uma tupla em r **determinam funcionalmente** ou (**exclusivamente**) os valores do componente Y
- Existe uma **dependência funcional (DF)** de X para Y
- Y é **funcionalmente dependente** de X

Dependência Funcional

Chaves candidatas

- Se X é uma chave candidata em R, isso implica que $X \rightarrow Y$, para qualquer subconjunto Y de atributos de R (inclusive $X \rightarrow R$)

Dependências Funcionais

Uso principal:

- Descrever melhor um esquema de relação R ao especificar restrições sobre seus atributos que devem ser mantidas o tempo todo

- Certas DFs podem não se referir a uma relação específica, mas sim a propriedades dos atributos. Ex.:
  $\{\text{Estado, Num\_habilitacao}\} \rightarrow \text{Cpf}$ é válida para qualquer adulto no Brasil e, portanto, deve ser mantida sempre que esses atributos apareçam juntos

Dependências Funcionais

Exemplos:

(a)

FUNC_DEP

Fnome	Cpf	Datanasc	Endereco	Dnumero	Dnome	Cpf_gerente

(b)

FUNC_PROJ

Cpf	Projnumero	Horas	Fnome	Projnome	Projlocal

a) Cpf $\rightarrow$ Fnome

b) Projnumero $\rightarrow$ {Projnome,Projlocal}

c) {Cpf,Projnumero} $\rightarrow$ Horas

DF é uma propriedade de um esquema de relação

- Uma DF não pode ser deduzida automaticamente a partir de um estado r de um esquema de relação R
- Uma DF deve ser definida de maneira explícita, por alguém que conheça a semântica dos atributos de R

Exemplo

ENSINA

Professor	Disciplina	Texto
Silva	Estruturas de Dados	Bartram
Silva	Gerenciamento de Dados	Martin
Neto	Compiladores	Hoffman
Braga	Estruturas de Dados	Horowitz

- Não podemos confirmar que Texto $\rightarrow$ Disciplina (a menos que saibamos que isso é válido para todos os estados válidos de ENSINA)
- Podemos sim dizer que Professor **não** determina funcionalmente a Disciplina, pois 'Silva' leciona duas disciplinas diferentes

DF é uma propriedade de um esquema de relação

Mais um exemplo

A	B	C	D
a1	b1	o1	d1
a1	b2	o2	d2
a2	b2	o2	d3
a3	b3	o4	d3

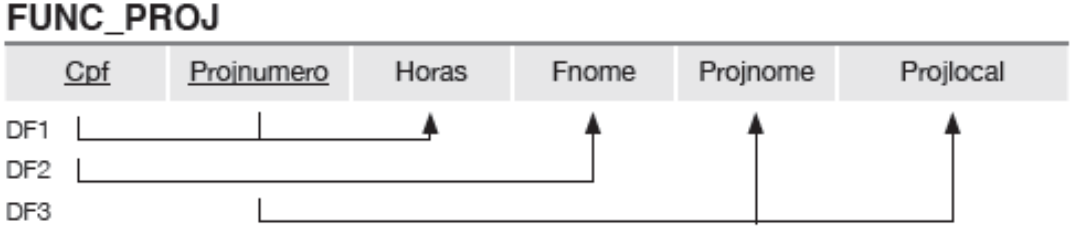
- DFs que **podem ser mantidas** na relação:
  $B \to C \quad C \to B \quad \{A, B\} \to C \quad \{A, B\} \to D \quad \{C, D\} \to B$
- DFs que que **não se mantém**:
  $A \to B \quad B \to A \quad D \to C$

Notação gráfica para dependências funcionais

(a)

DF1: Cpf $\rightarrow$ {Fnome, Datanasc, Endereco, Dnumero}
DF2: Dnumero $\rightarrow$ {Dnome,Cpf_gerente}

(b)



DF1: {Cfp,ProjNumero} $\rightarrow$ Horas
DF2: Cpf $\rightarrow$ Fnome
DF3: ProjNumero $\rightarrow$ {ProjNome,ProjLocal}

Processo de normalização – história

- Foi criado por Codd em 1972
- Aplica uma série de testes sobre um esquema de relação para certificar se ele satisfaz uma certa **forma normal**
- Inicialmente, Codd propôs 3 formais normais: **1FN, 2FN e 3FN**
- Posteriormente, Boyce e Codd definiram uma 3FN mais forte – a **FNBC**
- Essas formas se baseiam na análise das dependências funcionais
- Duas outras formas – a **4FN e 5FN** – foram propostas, baseadas na análise de dependências multivaloradas e dependências de junção

Normalização de relações

Definição

- É um processo que analisa os esquemas de relações **com base em suas DFs e chaves primárias** para alcançar as seguintes propriedades desejadas:
  1. minimização da redundância
  2. minimização das anomalias de inserção, exclusão e modificação
- Pode ser considerado um processo de *filtragem* ou *purificação*
  $\Rightarrow$ objetivo: melhorar gradativamente a qualidade do projeto do BD

Normalização

Ideia geral

- Os esquemas de relação insatisfatórios, que não atendem aos **testes de forma normal** são decompostos em esquemas de relação menores, que atendem aos testes e, portanto, possuem as propriedades desejáveis.

Forma Normal

*A **forma normal** de uma relação refere-se à condição de forma normal mais alta que ela atende e, portanto, indica o grau ao qual ela foi normalizada.*

Propriedades adicionais que o processo de normalização deve considerar

Propriedade de junção não aditiva (= junção sem perdas)

- Garante que tuplas falsas não sejam geradas a partir dos esquemas de relação resultantes da decomposição
  - Essa propriedade deve ser alcançada a todo custo!

Propriedade de preservação das dependências

- Garante que cada dependência funcional do esquema original seja representada em alguma relação individual resultante da decomposição
  - Essa propriedade é desejável, mas eventualmente pode ser sacrificada

Uso das formas normais na prática

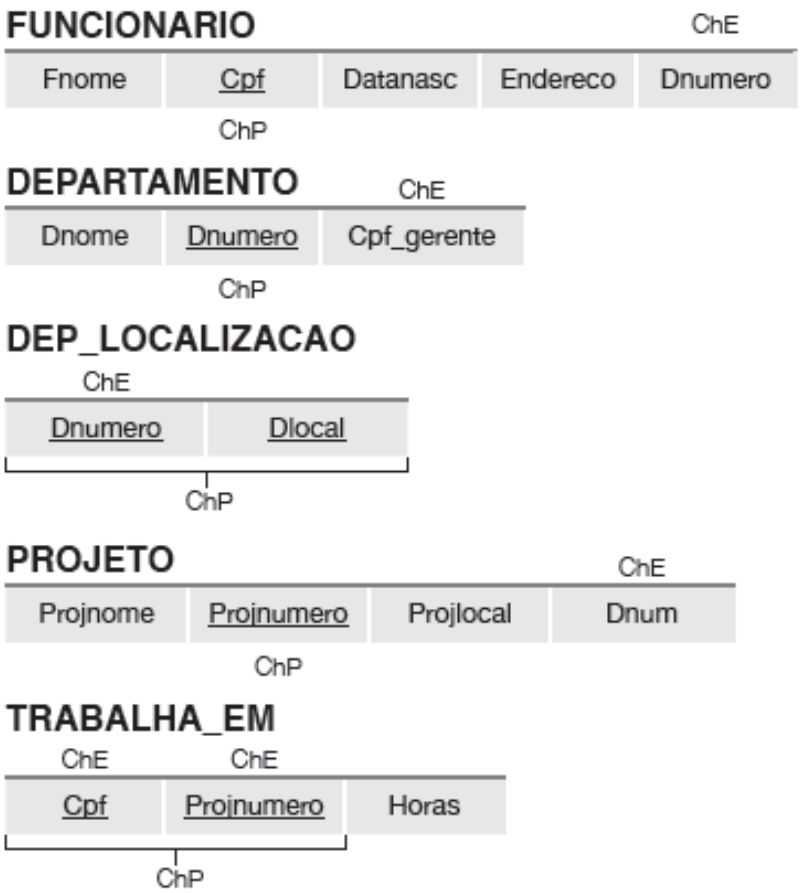
- O projeto de BD na indústria geralmente vai somente até a 3FN (mais raramente, na FNBC)
- Motivo: as restrições sobre as quais a 4FN e 5FN se baseam são raras e muito difíceis de serem entendidas e identificadas pelos projetistas de BDs
- Relembrando uma consideração importante: às vezes, as relações são mantidas em uma forma normal mais baixa (como a 2FN) por questões de desempenho

A normalização é particularmente importante em projetos que envolvam o uso de modelos legados e arquivos já existentes, para a obtenção de modelos de alta qualidade.

Definições preliminares

- **Atributo principal** – um atributo de um esquema de relação R é chamado de **atributo principal** de R se ele for membro de alguma chave candidata de R
- **Atributo não principal** – um atributo de um esquema de relação R é chamado de **atributo não principal** se ele não for membro de nenhuma chave candidata de R

Atributos principais – exemplo



Os atributos Cpf e Projnumero são atributos principais na relação TRABALHA_EM, ao passo que Horas é um atributo não principal.

Primeira forma normal (1FN)

Definição

Um esquema de relação R estará na **primeira forma nornal** se, e somente se, todos os atributos de R forem atômicos, ou seja, se cada atributo só puder ter um valor para cada tupla na relação.

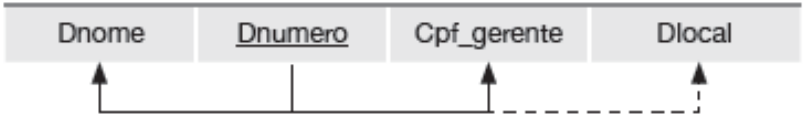
- A 1FN agora é considerada parte da definição formal de uma relação no modelo relacional básico (plano)
- Historicamente, ela foi definida para reprovar atributos multivalorados, atributos compostos e suas combinações, ou, em outras palavras, reprovar *relações dentro de relações* ou *relações como valores de atributo dentro de tuplas* (= relações aninhadas)

Solução (normalização) 1 para falha em teste da 1FN

- Expandir a chave de modo que haverá uma tupla separada na relação DEPARTAMENTO para cada local de DEPARTAMENTO (como na figura (c)). A chave primária torna-se a combinação {Dnumero,Dlocal}
- Essa solução não é boa porque introduz redundância.

(a)

DEPARTAMENTO



(b)

DEPARTAMENTO

Dnome	Dnumero	Cpf_gerente	Dlocal
Pesquisa	5	33344555587	Santo André, Itu, São Paulo
Administração	4	98765432168	Mauá
Matriz	1	88866555576	São Paulo

(c)

DEPARTAMENTO

Dnome	Dnumero	Cpf_gerente	Dlocal
Pesquisa	5	33344555587	Santo André
Pesquisa	5	33344555587	Itu
Pesquisa	5	33344555587	São Paulo
Administração	4	98765432168	Mauá
Matriz	1	88866555576	São Paulo

Solução (normalização) 2 para falha em teste da 1FN

- Para cada atributo que violar a 1FN no esquema de uma relação, removê-lo do esquema e colocá-lo em um esquema de relação separado, junto com a chave primária do seu esquema de origem.

(a)

FUNC_PROJ		Projs	
Cpf	Fnome	Projnumero	Horas

(b)

FUNC_PROJ1

Cpf	Fnome

FUNC_PROJ2

Cpf	Projnumero	Horas

(a) Esquema de relação FUNC_PROJ com um atributo de relação aninhada PROJS.

(b) Passagem de (a) para a 1FN: decomposição de FUNC_PROJ nas relações FUNC_PROJ1 e FUNC_PROJ2 pela propagação da chave primária.

Segunda forma normal (2FN)

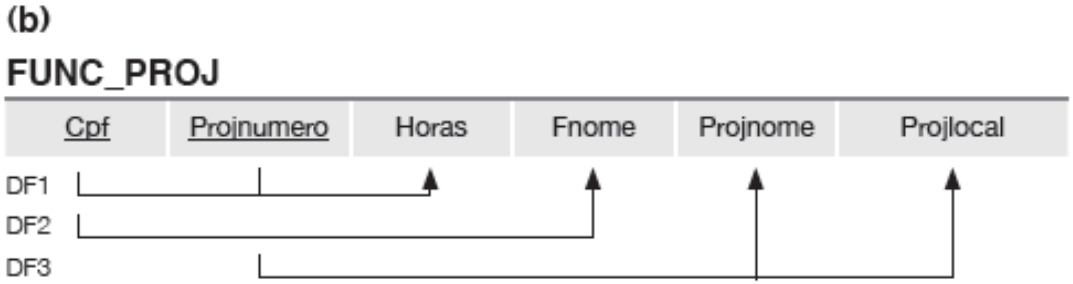
Definição

Um esquema de relação R está na **segunda forma normal** se, e somente se, ele estiver na 1FN e se cada atributo não principal A em R for *totalmente e funcionalmente dependente* de cada chave de R.

- Depende do conceito de **dependência funcional total**
- Uma DF $X \rightarrow Y$ é **total** se a remoção de qualquer atributo A de X significar que a dependência não se mantém mais, ou seja, para qualquer atributo $A \in X$, $(X - \{A\})$ não determina funcionalmente Y.
- Se a chave primária de R tem um único atributo, o teste da 2FN não precisa ser aplicado

Segunda forma normal (2FN)

Exemplos:



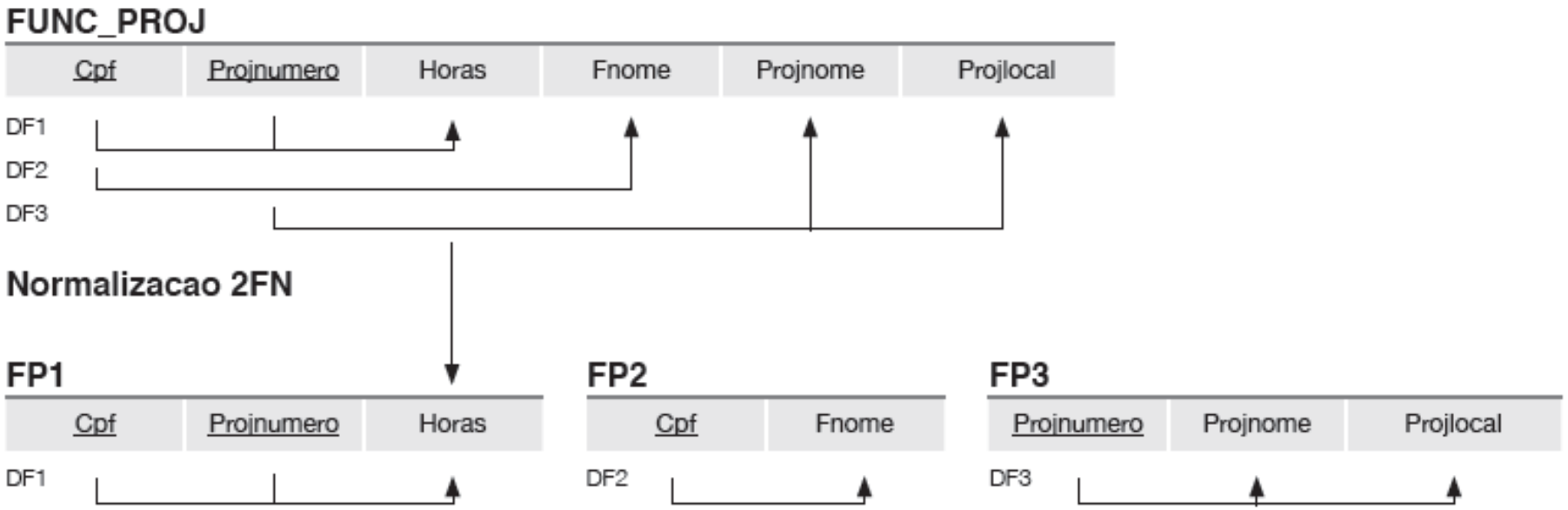
A relação FUNC_PROJ está na 1FN, mas não na 2FN porque possui atributos que são **parcialmente dependentes** da chave primária:

- O atributo Fnome viola a 2FN por causa da DF2
- Os atributos Projnome e Projlocal violam a 2FN por causa da DF3

Solução: decomposição de FUNC_PROJ em três esquemas – FP1, FP2 e FP3 – mostrados a seguir.

Segunda forma normal (2FN)

Decomposição do esquema FUNC_PROJ



Solução (normalização) para falha em teste da 2FN

- Decompor e montar uma nova relação para cada chave parcial com seu(s) atributo(s) dependente(s)
- Certificar-se de manter uma relação com a chave original e quaisquer atributos que sejam total e funcionalmente dependentes dela

Terceira forma normal (3FN)

Definição

Um esquema de relação R está na **terceira forma normal** se toda vez que uma dependência funcional não trivial $X \to A$ se mantiver em R, uma das duas condições a seguir precisam ser mantidas:

a) X é uma superchave de R, ou

b) A é um atributo principal de R

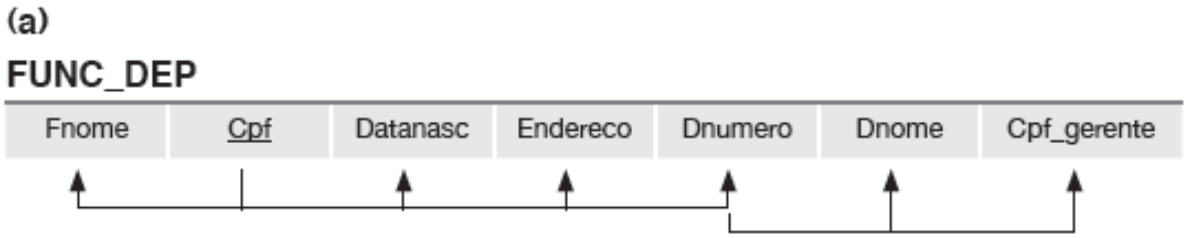
- Uma DF **trivial** é uma DF do tipo $XY \to Y$.

Terceira forma normal (3FN)

- Se baseia na ideia de impedir **dependências transitivas** e dependências parciais (como na 2FN)
- Uma DF $X \to Y$ em um esquema de relação R é **transitiva** se houver um conjunto de atributos Z em R que não sejam uma chave candidata nem um subconjunto qualquer de qualquer chave de R, e tanto $X \to Z$ quanto $Z \to Y$ se mantiverem.

Terceira forma normal (3FN)

Exemplos:



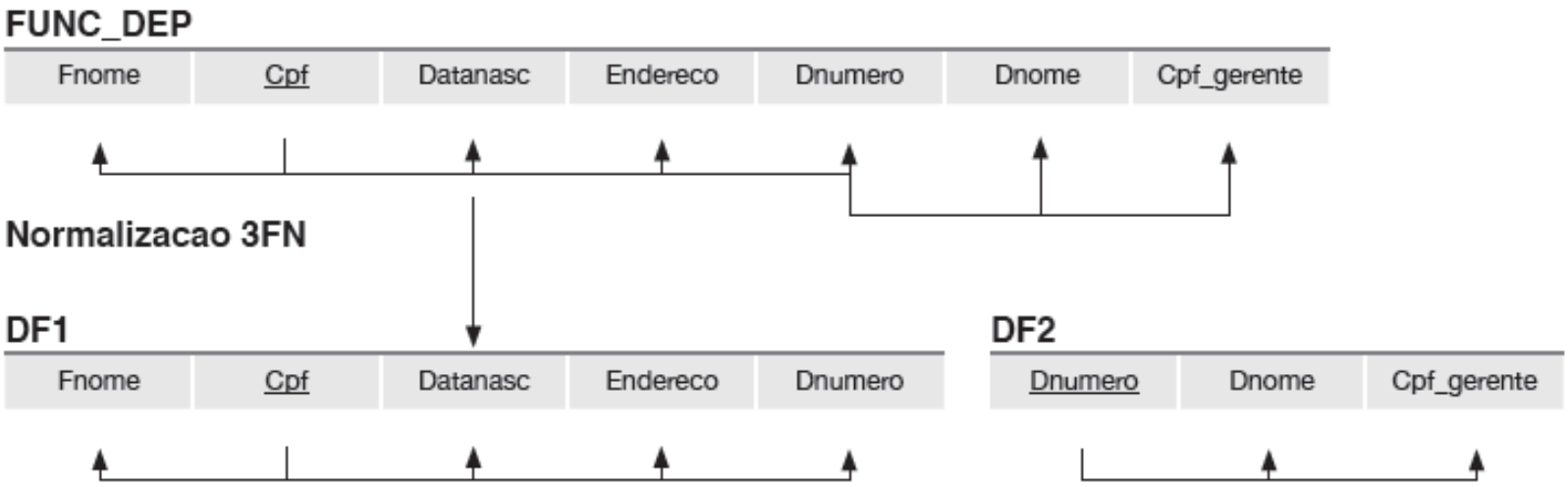
A relação FUNC_DEP está na 2FN, mas não na 3FN porque possui uma dependência transitiva de Cpf_gerente (e também Dnome) em Cpf por meio de Dnumero:

- Cpf $\rightarrow$ Dnumero
- Dnumero $\rightarrow$ Cpf_gerente
- Logo, Cpf $\rightarrow$ Cpf_gerente é transitiva

Solução: decomposição de FUNC_DEP em 2 esquemas – DF1 e DF2 – mostrados a seguir.

Terceira forma normal (3FN)

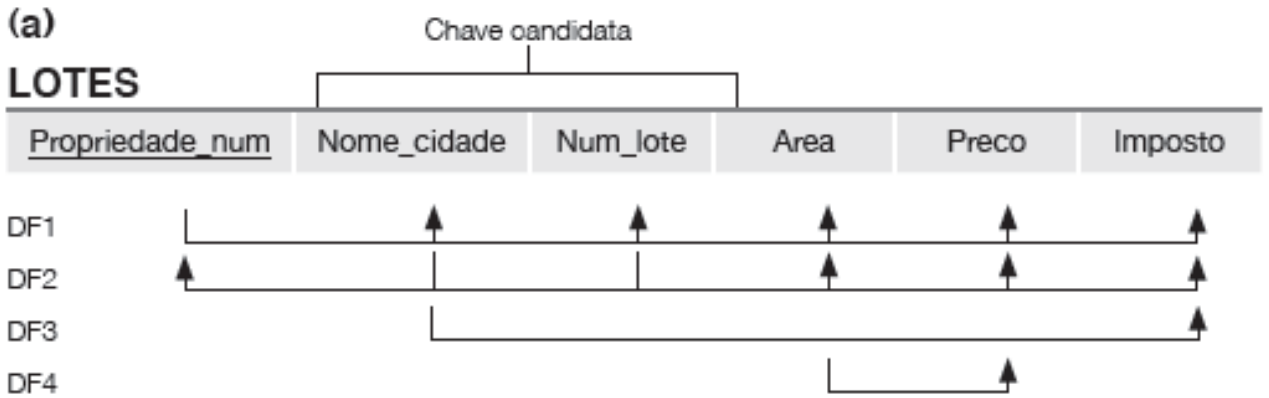
Decomposição do esquema FUNC_DEP



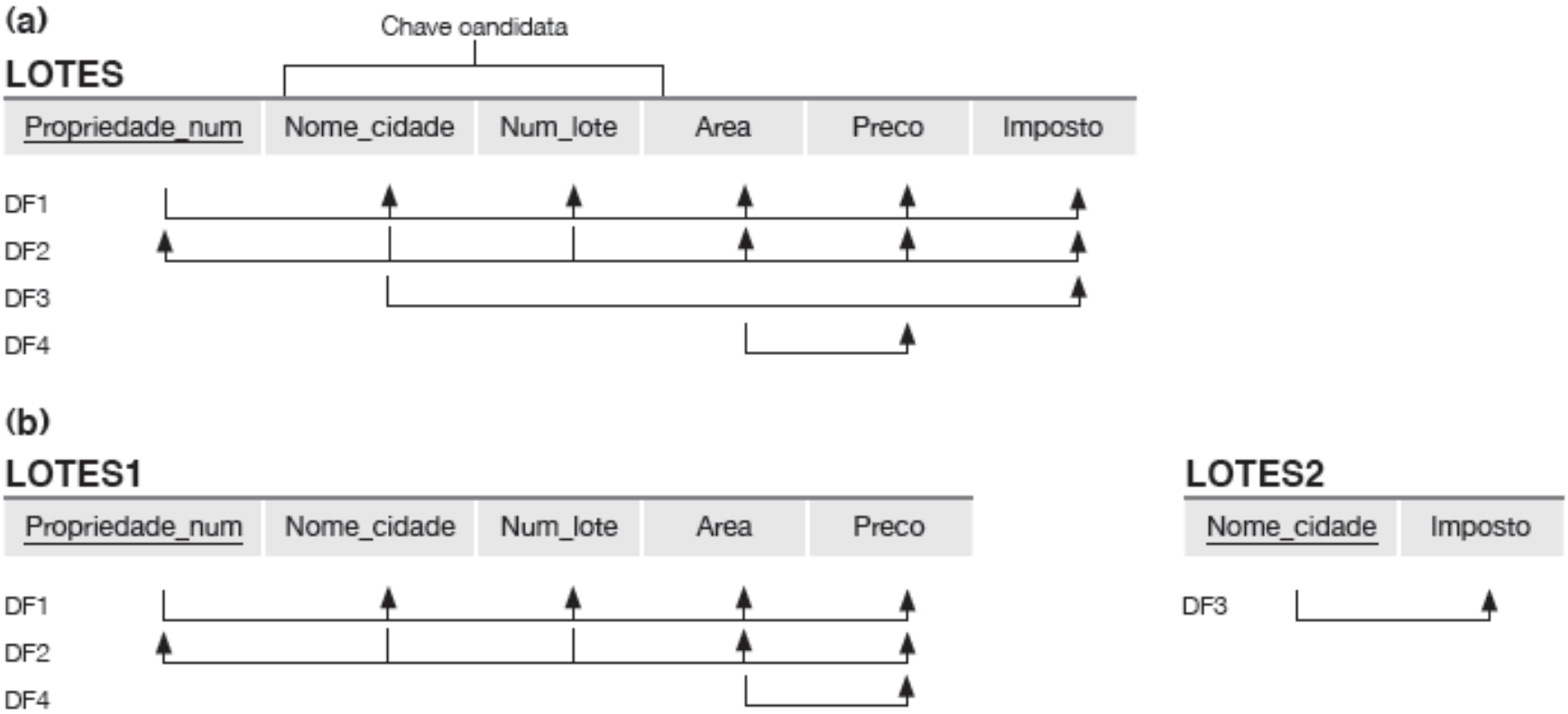
Solução (normalização) para falha em teste da 3FN

- Decompor e montar uma relação que inclua o(s) atributo(s) não chave que determina(m) funcionalmente outro(s) atributo(s) não chave.

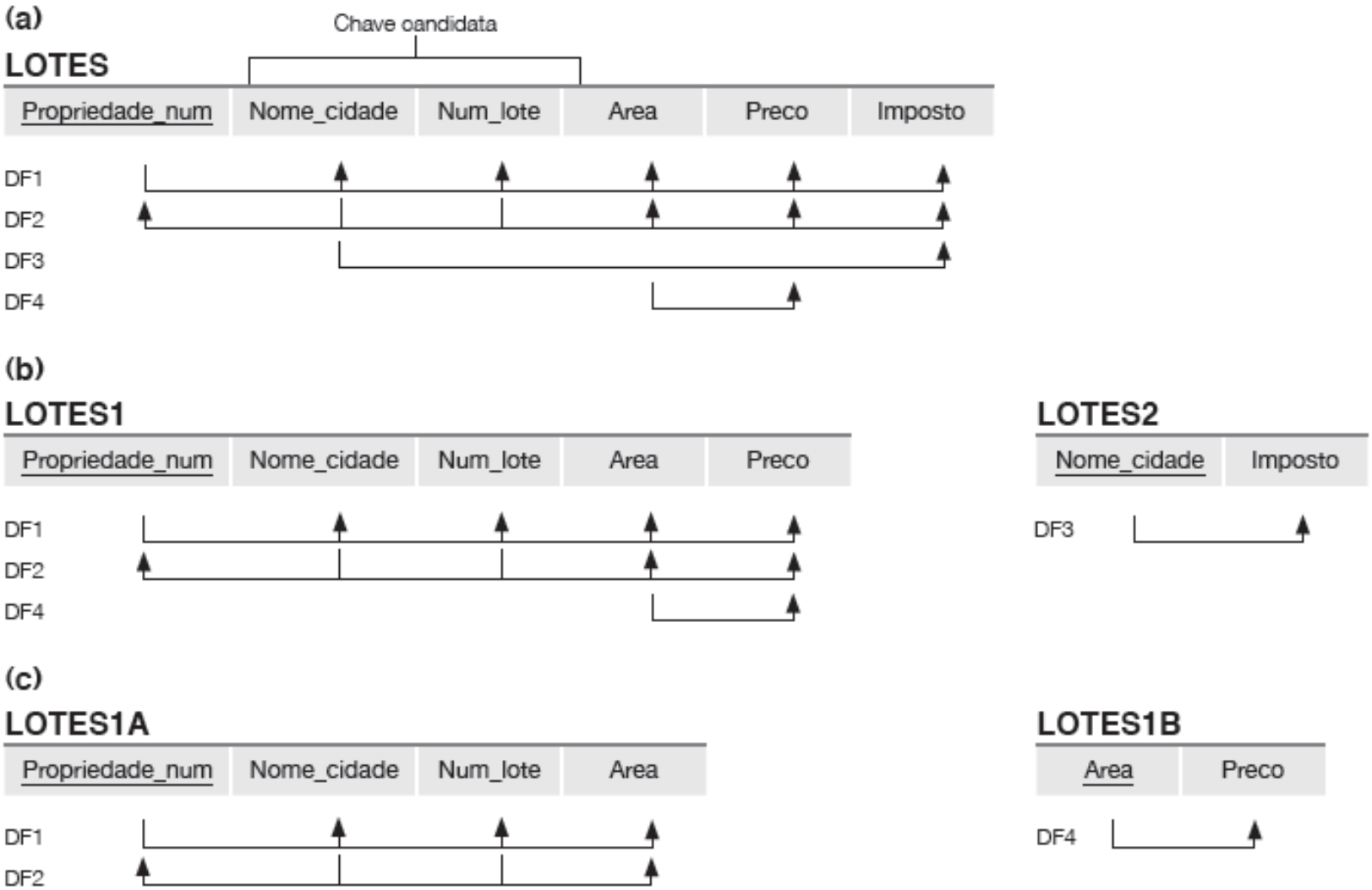
Lotes – exemplo de normalização (1FN)



Lotes – exemplo de normalização (2FN)



Lotes – exemplo de normalização (3FN)



Terceira forma normal (3FN)

- Observe que é possível testar diretamente a 3FN sem passar pelo teste da 2FN
- Se um esquema de relação R está na 3FN, então ele também está na 2FN
- No exemplo do slide anterior, na relação LOTES tanto a DF3 (dependência parcial) quanto a DF4 (dependência transitiva) violam a 3FN
  - Essas duas dependências poderiam ser removidas em um só passo de decomposição diretamente.

Forma normal de Boyce-Codd (FNBC)

Definição

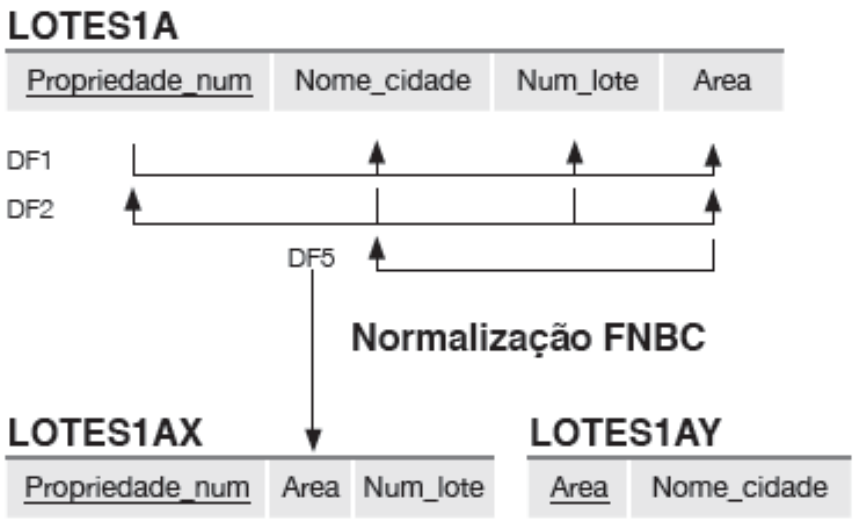
Um esquema de relação R está na **forma normal de Boyce-Codd** se toda vez que uma dependência funcional não trivial $X \to A$ se mantiver em R, então X é uma superchave de R.

- Foi proposta como uma forma mais simples da 3FN, mas descobriu-se que ela é mais rigorosa
- Uma relação que está na FNBC também está na 3FN (mas o contrário nem sempre é verdade!)

Lotes – exemplo de normalização (FNBC)

Nesse exemplo, consideramos o esquema LOTES1A – uma versão diferente de LOTES1, onde foi incluída uma nova dependência funcional: Area $\rightarrow$ Cidade.

O LOTES1A está na 3FN, mas não está na FNBC.



A decomposição de LOTES1A nos esquemas LOTES1AX e LOTES1AY está na FNBC.

Entretanto, a DF2 foi perdida (seus atributos não coexistem mais na mesma relação).

Mais um exemplo de normalização (FNBC)

Considere a relação ENSINA abaixo e suas dependências funcionais:
DF1: {Aluno, Disciplina} $\rightarrow$ Professor
DF2: Professor $\rightarrow$ Disciplina
A chave candidata de ENSINA é {Aluno, Disciplina} e, portanto, a relação está na 3FN, mas não na FNBC.

ENSINA

Aluno	Disciplina	Professor
Lima	Banco de dados	Marcos
Silva	Banco de dados	Navathe
Silva	Sistemas operacionais	Omar
Silva	Teoria	Charles
Souza	Banco de dados	Marcos
Souza	Sistemas operacionais	Antonio
Wong	Banco de dados	Gomes
Zelaya	Banco de dados	Navathe
Lima	Sistemas operacionais	Omar

Mais um exemplo de normalização (FNBC)

DF1: {Aluno, Disciplina} → Professor DF2: Professor → Disciplina

ENSINA

Aluno	Disciplina	Professor
Lima	Banco de dados	Marcos
Silva	Banco de dados	Navathe
Silva	Sistemas operacionais	Omar
Silva	Teoria	Charles
Souza	Banco de dados	Marcos
Souza	Sistemas operacionais	Antonio
Wong	Banco de dados	Gomes
Zelaya	Banco de dados	Navathe
Lima	Sistemas operacionais	Omar

Decomposições possíveis:

1. {Aluno, Professor} e {Aluno, Disciplina}
2. {Disciplina, Professor} e {Disciplina, Aluno}
3. {Professor, Disciplina} e {Professor, Aluno}

As 3 decomposições acima perdem a DF1.

⇒ **A única decomposição desejável é a (3), pois é a única que não gera tuplas falsas (= decomposição *não aditiva* = *decomposição sem perdas*)**

Exercício

Considere a seguinte relação para livros publicados:
`Livro(tituloLivro, nomeAutor, tipoLivro, faixaPreço, afiliaçãoAutor, editora)`
Suponha que há as seguintes dependências funcionais (DFs):
`tituloLivro` $\rightarrow$ `editora, tipoLivro`
`tipoLivro` $\rightarrow$ `faixaPreço`
`nomeAutor` $\rightarrow$ `afiliaçãoAutor`

a) Analisando o esquema e as DFs acima, determine a(s) chave(s) candidata(s) da relação `Livro`.

b) Em que forma normal essa relação está? Justifique.

c) Aplique a normalização até não poder mais decompor as relações. Explique a regra de normalização aplicada em cada etapa. Suas decomposições devem ser sem perdas e devem preservar as dependências funcionais originais.

Novos tipos de dependências e as 4FN e 5FN

- Existem restrições nas relações que não podem ser especificadas como dependências funcionais
- Por essa razão, foram propostos tipos adicionais de dependências:
  - **dependência multivalorada** (**MVD**, de *multivalued dependency*)
  - **dependência de junção (DJ)**
- A **quarta forma normal** e a **quinta forma normal** se baseiam nesses tipos de dependências

Dependência Multivalorada (MVD)

Definição

Uma MVD $X \twoheadrightarrow Y$ especificada sobre o esquema de relação R, onde X e Y são subconjuntos de R, determina a seguinte restrição sobre qualquer estado de relação r de R:

Se existirem duas tuplas t_1 e t_2 em r tais que $t_1[X] = t_2[X]$, então deverão também existir duas tuplas t_3 e t_4 em r com as seguintes propriedades (onde $Z = R - (X \cup Y)$):

- $t_3[X] = t_4[X] = t_1[X] = t_2[X]$
- $t_3[Y] = t_1[Y]$ e $t_4[Y] = t_2[Y]$
- $t_3[Z] = t_2[Z]$ e $t_4[Z] = t_1[Z]$

Interpretação: quando duas tuplas de r "concordam" em todos os atributos de X, então seus valores para Y podem ser trocados e o resultado será duas tuplas que também estarão em r

Exemplos de MVD

FUNC

Fnome	Projnome	Dnome
Silva	X	João
Silva	Y	Ana
Silva	X	Ana
Silva	Y	João

A relação FUNC com duas MVDs:

- Fnome $\twoheadrightarrow$ ProjNome
- Fnome $\twoheadrightarrow$ Dnome

Qual é a origem de MVDs?

- MVDs podem se originar da tentativa de se representar num mesmo esquema de relação **dois ou mais atributos multivalorados independentes**
- Nesse caso, tem-se o problema de ter que repetir cada valor de um dos atributos com cada valor do outro atributo. Isso é feito para:
  1) Manter o estado da relação coerente
  2) Manter as independências entre os atributos

MVD

FUNC

Fnome	Projnome	Dnome
Silva	X	João
Silva	Y	Ana
Silva	X	Ana
Silva	Y	João

Observe que a definição de MVD é simétrica, portanto:

- Toda vez que temos $X \twoheadrightarrow Y$, temos também $X \twoheadrightarrow Z$
- Por isso, escreve-se também $X \twoheadrightarrow Y|Z$

MVDs e valores redundantes

- Quando se tem uma **MVD não trivial** numa relação, geralmente é preciso repetir valores redundantemente nas tuplas

  Essa redundância é indesejável!
- Uma MVD $X \twoheadrightarrow Y$ é **trivial** se
  (a) Y for um subconjunto de X **ou**
  (b) $X \cup Y = R$
- Observe que uma MVD trivial não especifica nenhuma restrição sobre R!
- MVDs não triviais costumam aparecer em relações em que a chave é formada por todos os atributos da relação juntos
  - É raro que essas relações, onde há ocorrência combinatória de valores repetidos, sejam projetadas na prática

Quarta forma normal (4FN)

Definição

Um esquema de relação R está na **4FN** se toda vez que uma dependência multivalorada (MVD) não trivial $X \twoheadrightarrow Y$ se mantiver em R, X é uma superchave para R.

- Uma relação onde a chave é a composição de todos os atributos está sempre na FNBC (pois não tem DFs não triviais), mas pode não estar na 4FN
- Uma relação que não está na 4FN devido a uma MVD não trivial precisa ser decomposta (removendo a redundância), para convertê-la em um conjunto de relações na 4FN.

Decomposição do esquema FUNC

(a)

FUNC

Fnome	Projnome	Dnome
Silva	X	João
Silva	Y	Ana
Silva	X	Ana
Silva	Y	João

(b)

FUNC_PROJETOS

Fnome	Projnome
Silva	X
Silva	Y

FUNC_DEPENDETES

Fnome	Nome_dependente
Silva	João
Silva	Ana

Decomposição de FUNC em duas relações na 4FN: FUNC_PROJETOS e FUNC_DEPENDENTES .

FNBC $\times$ 4FN

- Toda DF $X \to Y$ é também uma MVD $X \twoheadrightarrow Y$
  - Se $X \to Y$, então trocar os valores de Y entre duas tuplas que "concordam" nos valores de X não mudará o valor das tuplas
  - Assim, as "novas" tuplas certamente estarão em r e, portanto, temos $X \twoheadrightarrow Y$
- Assim, se R está na 4FN então ela também está na FNBC

Dependência de Junção (DJ)

Definição

Uma dependência de junção, indicada por DJ($R_1, R_2, \ldots, R_n$), especificada no esquema de relação R, determina uma restrição sobre os estados r de R:

Cada estado válido r de R deve ter uma decomposição de junção não aditiva para $R_1, R_2, \ldots, R_n$. Logo, para cada r desse tipo, temos:

$$*(\pi_{R_1}(r), \pi_{R_2}(r), \ldots, \pi_{R_n}(r)) = r$$

Quinta forma normal (5FN)

Definição

Um esquema de relação R está na **5FN** (ou na **forma normal projeção-junção – FNPJ**) com relação a um conjunto F de dependências funcionais, multivaloradas e de junção se, para cada dependência de junção não trivial $\mathrm{DJ}(R_1, R_2, \ldots, R_n)$ em F^+ (ou seja, implicada por F), cada R_i é uma superchave de R.

Dependência de Junção

Exemplo

- Considere a relação FORNECE no próximo slide
- Suponha que a seguinte restrição seja sempre mantida:
  - Toda vez que um fornecedor f fornece a peça p, **e** um projeto j usa a peça p, **e** o fornecedor f fornece pelo menos uma peça para o projeto j, então o fornecedor f também estará fornecendo a peça p ao projeto j.
- Essa restrição especifica uma **dependência de junção** DJ(R_1, R_2, R_3) entre as seguintes três projeções de FORNECE:
  - R_1(Nome_fornece, Nome_peça)
  - R_2(Nome_fornece, Nome_proj)
  - R_3(Nome_peça, Nome_proj)

Se essa restrição for mantida, as tuplas que aparecem abaixo da linha tracejada na relação FORNECE do próximo slide devem existir em todo estado válido de FORNECE que também contém as tuplas acima da linha tracejada

Dependência de Junção

Exemplo

FORNECE

Nome_fornece	Nome_peca	Nome_proj
Silva	Peneira	ProjX
Silva	Porca	ProjY
Adam	Peneira	ProjY
Walter	Porca	ProjZ
Adam	Prego	ProjX
Adam	Peneira	ProjX
Silva	Peneira	ProjY

- "Toda vez que um fornecedor f fornece a peça p, **e** um projeto j usa a peça p, **e** o fornecedor f fornece pelo menos uma peça para o projeto j, então o fornecedor f também estará fornecendo a peça p ao projeto j."
- DJ(R_1, R_2, R_3), onde:
  - R_1(Nome_fornece, Nome_peça)
  - R_2(Nome_fornece, Nome_proj)
  - R_3(Nome_peça, Nome_proj)
- Como R_1, R_2 e R_3 não são superchaves de FORNECE, então FORNECE **não está na 5FN**

Decomposição do esquema FORNECE

FORNECE

Nome_fornece	Nome_peca	Nome_proj
Silva	Peneira	ProjX
Silva	Poroa	ProjY
Adam	Peneira	ProjY
Walter	Poroa	ProjZ
Adam	Prego	ProjX
Adam	Peneira	ProjX
Silva	Peneira	ProjY

Depois da decomposição:

R1

Nome_fornece	Nome_peca
Silva	Peneira
Silva	Poroa
Adam	Peneira
Walter	Poroa
Adam	Prego

R2

Nome_fornece	OR_proj
Silva	ProjX
Silva	ProjY
Adam	ProjY
Walter	ProjZ
Adam	ProjX

R3

Nome_peca	OR_proj
Peneira	ProjX
Poroa	ProjY
Peneira	ProjY
Poroa	ProjZ
Prego	ProjX

- Os três esquemas gerados estão na 5FN.
- Observe que a aplicação de uma junção natural sobre *duas relações quaisquer da decomposição* pode produzir tuplas falsas. Entretanto, **a junção natural das três relações não gera tuplas falsas**.

Referências Bibliográficas

- *Sistemas de Bancos de Dados* (6ª edição), Elmasri e Navathe. Pearson, 2010. – Capítulo 15
- *Introdução a Sistemas de Bancos de Dados* (8ª edição), Date. Campus, 2003. – Capítulos 11 e 12
- *Projeto e Modelagem de Bancos de Dados*, Teorey, Lightstone, Nadeau. Campus, 2007. – Capítulo 6
- *Database Systems – the complete book* (2ª edição), Garcia-Molina, Ullman e Widom. Prentice Hall, 2009. Capítulo 3