Normalização

Tópicos do curso

- Introdução – conceitos básicos e arquitetura de SGBD
- Modelagem de Dados (MER)
- Modelo Relacional
- Dependências Funcionais e normalização

Normalização

- **Processo durante o qual esquemas de relações não-satisfatórias são decompostas em esquemas de relações menores que possuem as propriedades desejadas**
- **Ferramenta para validar e melhorar o projeto lógico de modo a satisfazer certas restrições e evitar duplicação desnecessária de dados**

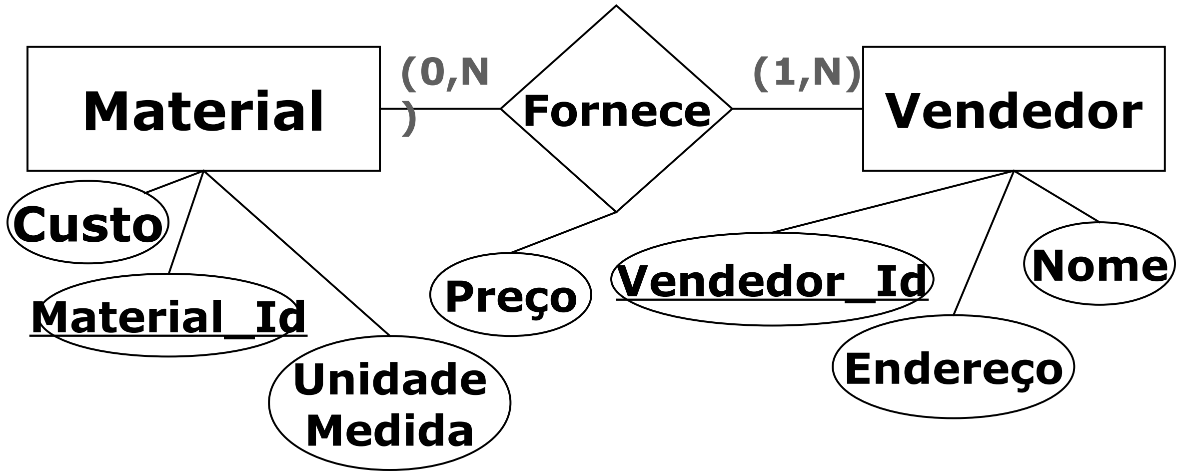
Normalização

- Processo de decomposição de relações com anomalias de modo a produzir relações menores bem estruturadas
- Medir formalmente por que um conjunto de atributos em esquemas de relações é melhor do que outro

Dependência Funcional

- **Uma restrição entre dois atributos ou dois conjuntos de atributos**
- **Advém do conhecimento da semântica dos dados armazenados**
- **Para qualquer relação R, um atributo B é funcionalmente dependente de um atributo A se, para toda instância de R, este valor de A determina univocamente o valor de B**
  - $A \rightarrow B$
  - $\{A,B\} \rightarrow \{C,D\} \approx AB \rightarrow CD$

Dependência Funcional



Material

Material_Id	Custo	Unidade_de_Medida

Fornece

Material_Id	Vendedor_Id	Preço

Vendedor

Vendedor_Id	Nome	Endereço

Dependência Funcional

Material

Material_Id	Custo	Unidade_de_Medida

Fornece

Material_Id	Vendedor_Id	Preço

Vendedor

Vendedor_Id	Nome	Endereço

- Material_Id → {Custo,Unidade_de_Medida}
- Vendedor_Id → {Nome,Endereço}
- {Material_Id,Vendedor_Id} → Preço
- {Vendedor_Id, Nome} → Endereço

Dependência Funcional

- Determinante: atributo do lado esquerdo da flecha (da dependência funcional)

Regras de Inferência de Armstrong

- **Também chamado de Axiomas de Armstrong**
  - **A1. Se X contém Y então X → Y (reflexidade)**
    - **Uma df obtida por este axioma é chamada de df trivial**
  - **A2. Se X → Y então para qualquer Z, XZ→YZ (aumentativa)**
  - **A3. Se X → Y e Y → Z então X → Z (transitividade)**
  - **{A1,A2,A3}**

Regras de Inferência de Armstrong

- **A4. Se $X \rightarrow Y$ e $X \rightarrow Z$, então $X \rightarrow YZ$ (união)**
  - **$X \rightarrow XY$ por A2; como $X \rightarrow Z$, $XY \rightarrow ZY$ por A2 então $X \rightarrow ZY$ por A3**
  - **(Note que ZY=YZ)**
- **A5. Se $X \rightarrow Y$ e Y contém Z então $X \rightarrow Z$ (decomposição)**
  - **$Y \rightarrow Z$ por A1 e como $X \rightarrow Y$, então $X \rightarrow Z$ por A3**
  - **Se $X \rightarrow A_1, A_2,\ldots,A_n$ então $X \rightarrow A_1$, $X \rightarrow A_2$,…, $X \rightarrow A_n$**
  - **Sem perda de generalidade pode-se supor que todas as dfs são do tipo $X \rightarrow A_i$, onde A_i é um atributo simples**
    - **Forma canônica**

Regras de Inferência de Armstrong

- **A6. Se $X \rightarrow Y$, $WY \rightarrow Z$, então $WX \rightarrow Z$(pseudo-transitividade)**

Fecho (Closure)

- **O Fecho de um subconjunto F de dfs é o conjunto F^+ de todas as dfs que podem ser inferidas a partir de F**
- **O Fecho de um conjunto de atributos X com relação a F é o conjunto X^+ de todos os atributos que são determinados funcionalmente por X**
- **X^+ pode ser calculado aplicando os axiomas A1, A2, A3 repetidamente usando as dfs em F**

Fecho (Closure)

- Fecho de X: X^+
- Algoritmo:
  - $X^+ := X$
  - Repita
    - $oldX^+ := X^+$
    - Para toda DF $Y \rightarrow Z$ em F
      - Se X^+ contém Y então
        - $X^+ := X^+ \cup Z$
  - Até que $(oldX^+ = X^+)$

Fecho (Closure)

- **F = {RG →Nome_Func, Num_Proj→{Nome_Proj,Loc_Proj}, {RG,Num_Proj} →Horas}**
  - **{RG}⁺={RG, Nome_Func}**
  - **{Num_Proj}⁺={Num_Proj, Nome_Proj, Loc_Proj}**
  - **{RG,Num_Proj}⁺={RG, Nome_func, Num_Proj, Nome_Proj, Loc_Proj, Horas}**

Fecho (Closure): usos

- **Determinar se uma df $X \rightarrow Y$ é uma df válida.**
  - **Calcular X^+. Se X^+ contém Y, então $X \rightarrow Y$ é uma df válida.**
- **Determinar se um dado subconjunto W é chave de $R=(A_1,A_2,\ldots,A_n)$**
  - **Calcular W^+ e verificar se $W^+=\{A_1,A_2,\ldots,A_n\}$. Se isto ocorre, $W \rightarrow A_1$, $W \rightarrow A_2,\ldots, W \rightarrow A_n$ e, portanto, W é superchave de R**

Dependência Funcional: chave

- **Atributo, ou combinação de atributos que identifica univocamente uma linha de uma relação**
  - **Identificação única: todo atributo não-chave é dependente funcionalmente da chave**
  - **Não-redundância: nenhum atributo na chave pode ser eliminado sem destruir a propriedade de identificação única**

Dependência Funcional: chave

- $A_i, A_j, \ldots, A_k$ é uma chave de uma relação R se e somente se:
  - $A_i, A_j, \ldots, A_k$ determina todos os outros atributos de R
  - Nenhum subconjunto de $A_i, A_j, \ldots, A_k$ determina funcionalmente os atributos restantes de R

- Uma superchave de R é um subconjunto dos atributos de R que contém uma chave
  - Toda chave é uma superchave, mas nem toda superchave é chave

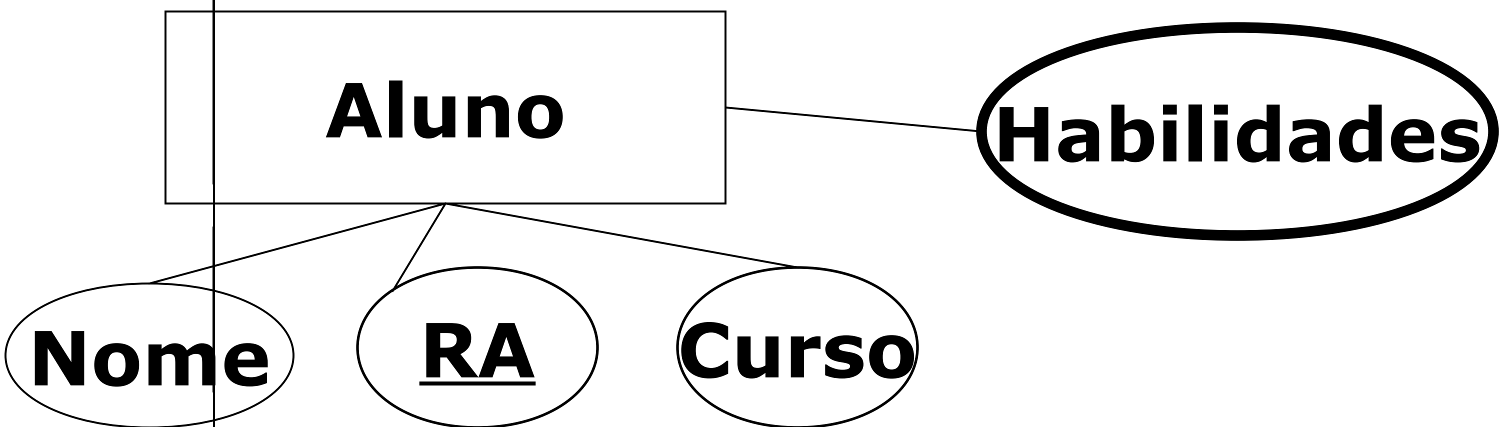
Forma Normal

- Estado de uma relação que resulta da aplicação de regras simples com respeito a dependências funcionais envolvendo atributos desta relação
- Conjunto de testes para "certificar" se uma dada relação pertence a uma forma normal
- Uma relação está em uma forma normal quando ela satisfaz um número de características desejáveis

Primeira Forma Normal (1FN)

- **Uma relação não pode conter atributo multivalorado nem composto**
- **O domínio dos atributos deve incluir somente valores atômicos (simples/indivisíveis) e que o valor de qualquer atributo deve ser um único valor do domínio daquele atributo**

Primeira Forma Normal (1FN)



Aluno

<u>RA</u>	Nome	Curso	Habilidade
000001	João	Letras	{Tupi}
000002	Maria	Matemática	{Inglês,Espanhol}
963127	Ricardo	Computação	{Inglês,Francês}

Primeira Forma Normal (1FN)

- **Com redundância**

Aluno

<u>RA</u>	Nome	Curso	Habilidade
000001	João	Letras	Tupi
000002	Maria	Matemática	Inglês
000002	Maria	Matemática	Espanhol
963127	Ricardo	Computação	Inglês
963127	Ricardo	Computação	Francês

Primeira Forma Normal (1FN)

Aluno

<u>RA</u>	Nome	Curso
000001	João	Letras
000002	Maria	Matemática
963127	Ricardo	Computação

Habilidades_Aluno

<u>RA</u>	<u>Habilidade</u>
000001	Tupi
000002	Inglês
000002	Espanhol
963127	Inglês
963127	Francês

Segunda Forma Normal (2FN)

- **Dependência Parcial**
  - **Uma dependência funcional na qual um ou mais atributos não-chave são dependentes de parte da chave primária**
  - **Um atributo A é dependente funcional parcial de um atributo X, se X → A e existe um subconjunto próprio Y de X tal que Y → A**
    - **Chama-se subconjunto próprio de um conjunto A, todo conjunto B, tal que B está contido em A e B ≠ A**

Segunda Forma Normal (2FN)

- **Dependência Parcial**

Empregado

Matr	Título_Curso	Nome	Depto_Nome	Salário	Data_completado

Segunda Forma Normal (2FN)

- **Uma relação R está em 2FN se está em 1FN e todo atributo não-chave é totalmente dependente funcionalmente da chave primária**
- **Uma relação R está em 2FN se nenhum ANP de R é dependente funcional parcial de qualquer chave de R**

Segunda Forma Normal (2FN)

- **Uma relação em 1FN está em 2FN:**
  - **Se a chave primária consiste de apenas um atributo**
  - **Nenhum atributo não-chave existe na relação (todos os atributos na relação são componentes da chave primária)**
  - **Todo atributo não-chave é dependente funcionalmente de todo o conjunto de atributos da chave primária**

Segunda Forma Normal (2FN)

- Dependência Parcial

Empregado

Matr	Título_Curso	Nome	Depto_Nome	Salário	Data_completado

- Solução: decompor em duas relações
  - Empregado(<u>Matr</u>, Nome,Depto_Nome,Salário)
  - EmpCurso(<u>Matr</u>,<u>Título_Curso</u>,Data_Completado)

Terceira Forma Normal (3FN)

- **Dependência Transitiva**
  - **Dependência entre dois ou mais atributos não-chaves**
  - **Dadas as dfs $X \rightarrow Y$, $\cancel{Y} \rightarrow X$, e $Y \rightarrow Z$, dizemos que Z depende transitivamente de X**

Vendas

Cliente_Id	Nome	Vendedor	Região

Terceira Forma Normal (3FN)

Vendas

Cliente_Id	Nome	Vendedor	Região
8023	Anderson	Sandro	Sul
9167	Bruno	Humberto	Oeste
7929	Horácio	Sandro	Sul
6837	Tom	João	Leste
8596	Emerson	Humberto	Oeste
7018	Arnaldo	Fabricio	Norte

Terceira Forma Normal (3FN)

- **Possíveis Anomalias**
  - **Inserção**
    - **Um novo vendedor não pode ser inserido a menos que um cliente tenha sido alocado para um vendedor**
  - **Deleção**
    - **Quando um cliente é deletado (6837) da tabela, perde-se a informação de que João está alocado para a região Leste**
  - **Atualização**
    - **Se um vendedor muda de região, várias linhas deverão ser modificadas para refletir este fato**

Vendas

Cliente_Id	Nome	Vendedor	Região
8023	Anderson	Sandro	Sul
9167	Bruno	Humberto	Oeste
7929	Horácio	Sandro	Sul
6837	Tom	João	Leste
8596	Emerson	Humberto	Oeste
7018	Arnaldo	Fabricio	Norte

Terceira Forma Normal (3FN)

- **Solução: decompor a relação em duas**
  - **Economiza espaço: região não repetida para cada cliente**

VR

Vendedor	Região

Vendas

Cliente_Id	Nome	Vendedor

Terceira Forma Normal (3FN)

- **Uma relação em 2FN que não tem nenhuma dependência transitiva**
- **Uma relação R está em 3FN se nenhum ANP de R depende transitivamente de qualquer chave de R**

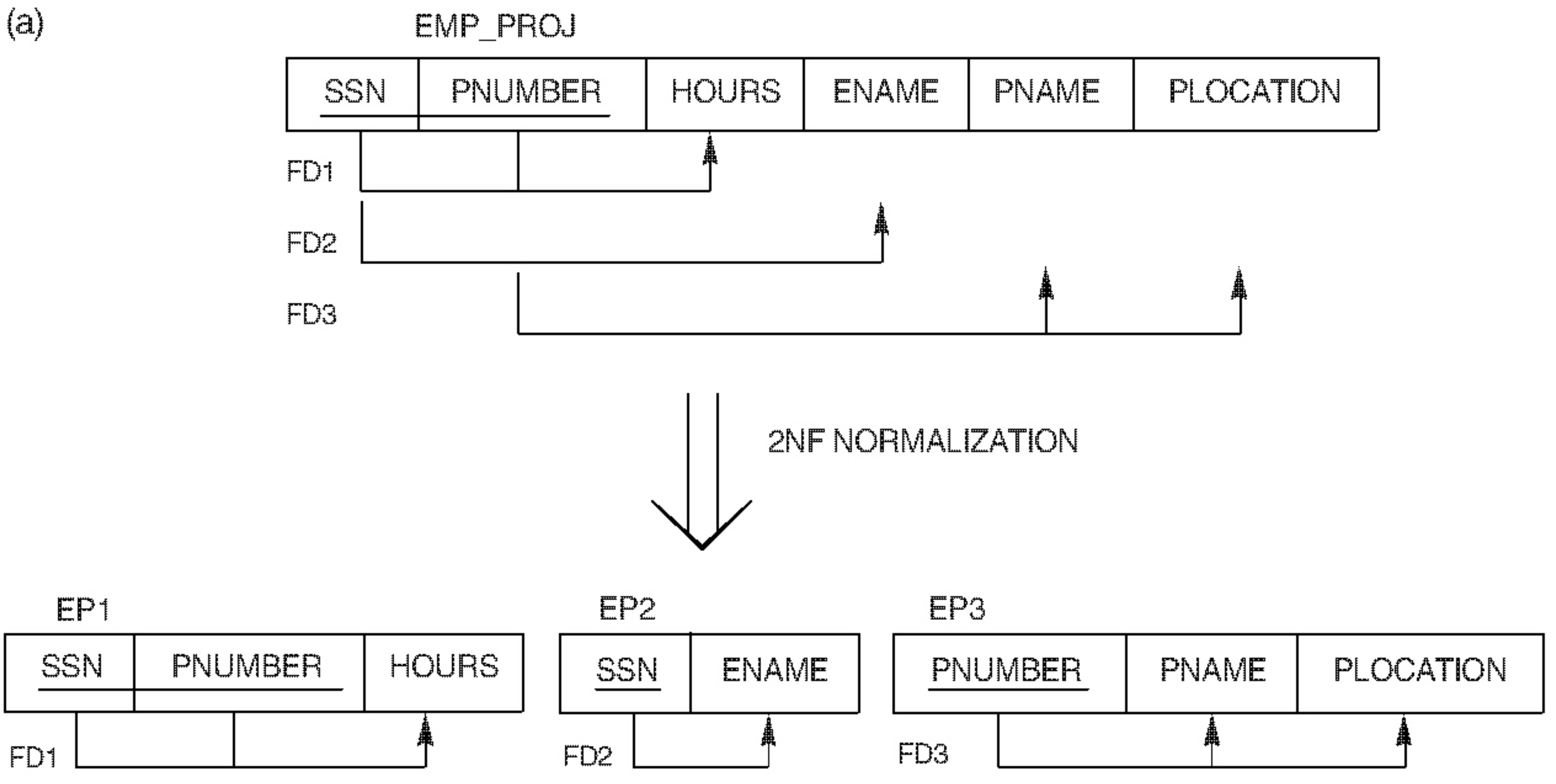
Terceira Forma Normal (3FN)

- **Uma relação R está em 3FN se, para toda df não trivial X → Y, válida para R, então**
  - **(a) X é uma superchave ou**
  - **(b) Y é atributo primo**

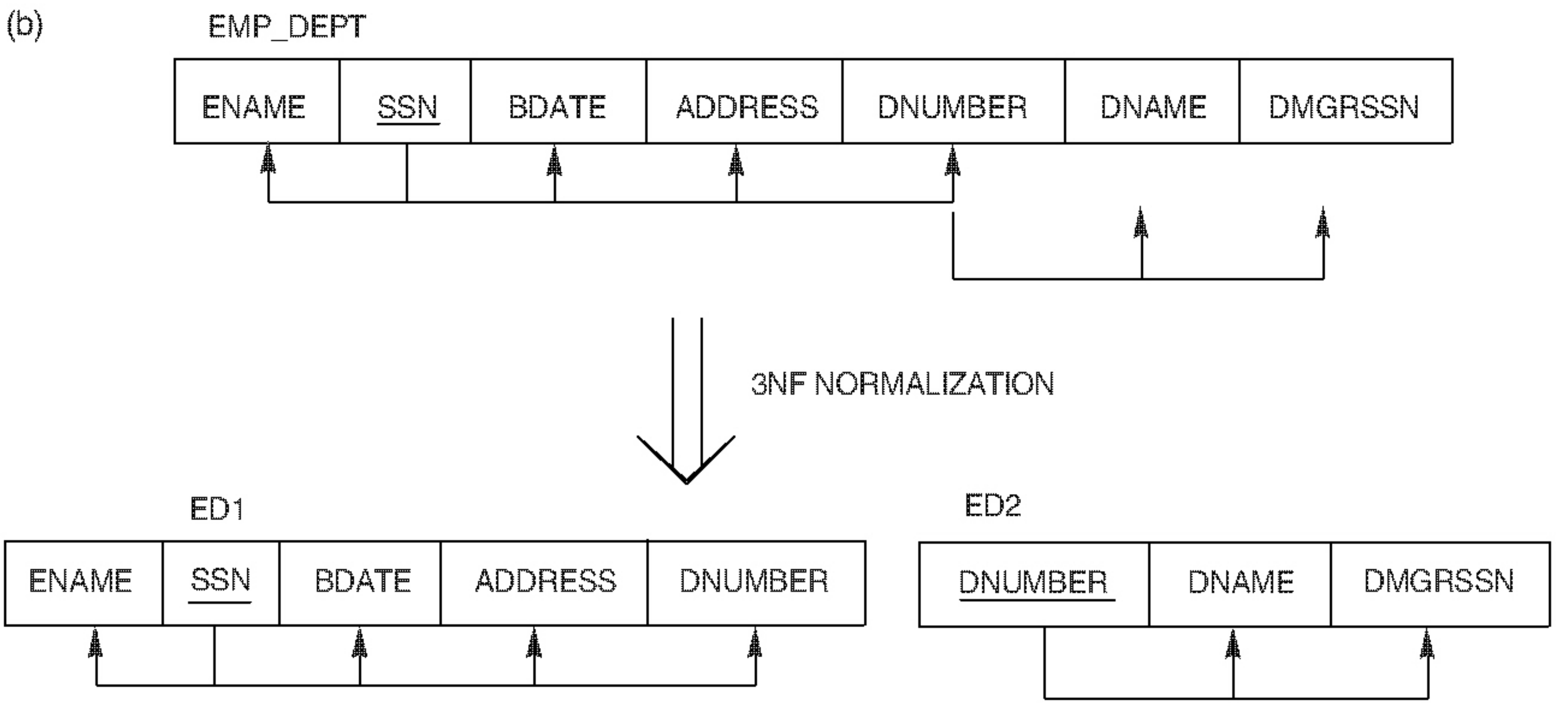
Terceira Forma Normal (3FN)

- **Uma relação R está em 3FN se, para toda df não trivial $X \rightarrow Y$, válida para R, então**
  - **(a) X é uma superchave ou**
  - **(b) Y é atributo primo**
- **Se uma relação R viola 3FN, tanto a condição (a) quanto a condição (b) é violada**
  - **Violar (b): Y é um atributo não-primo**
  - **Violar (a): X não é um superconjunto de nenhuma chave de R**
    - **X poderia ser não-primo**
      - **Viola 3FN (transitividade)**
    - **X poderia ser subconjunto de uma chave de R**
      - **Viola 2FN e 3FN: dependência parcial da chave**

Exemplos (1)

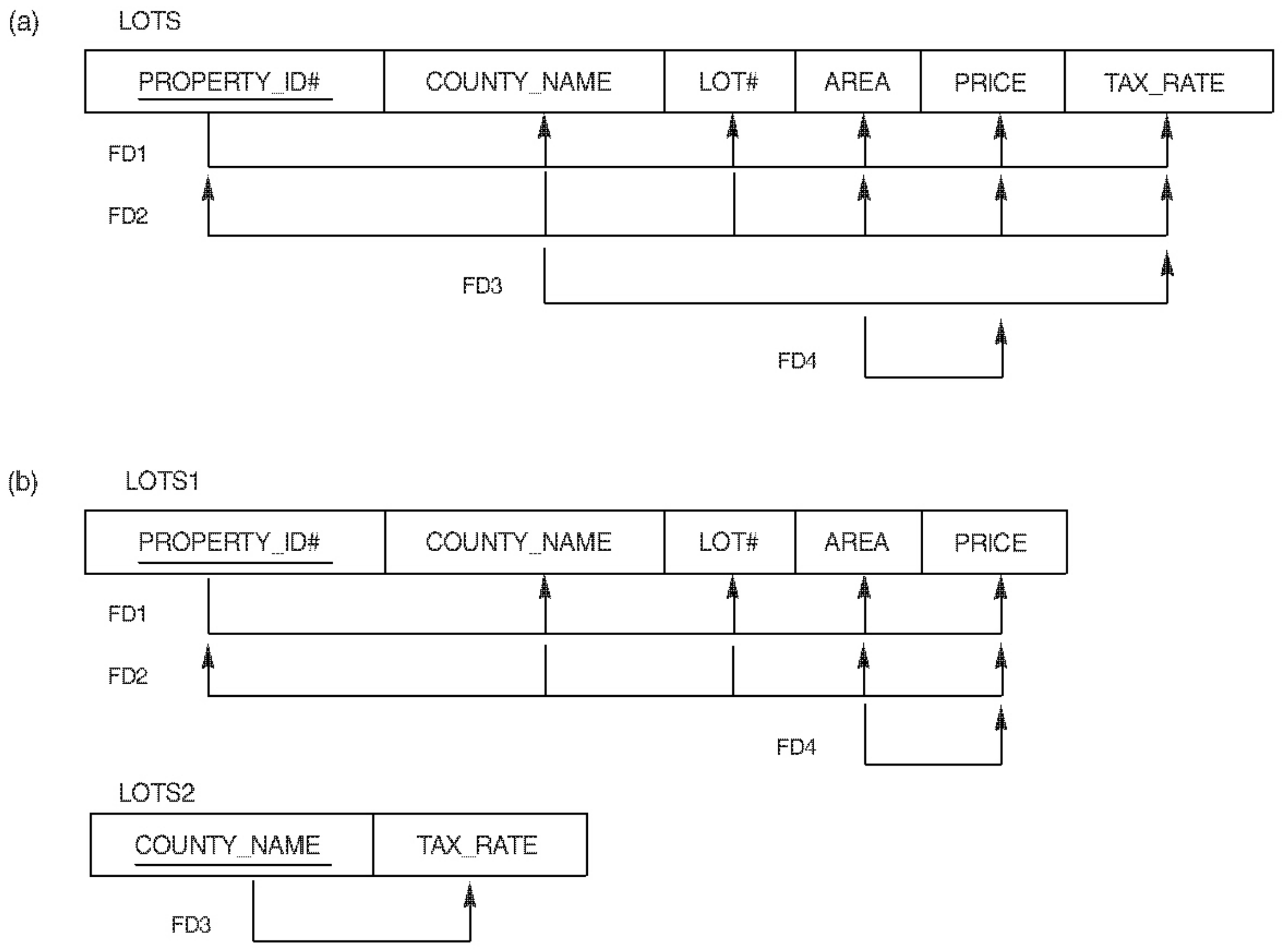

(a)
EMP_PROJ
SSN
PNUMBER
HOURS
ENAME
PNAME
PLOCATION
FD1
FD2
FD3
2NF NORMALIZATION
EP1
SSN
PNUMBER
HOURS
FD1
EP2
SSN
ENAME
FD2
EP3
PNUMBER
PNAME
PLOCATION
FD3

Exemplos (1)

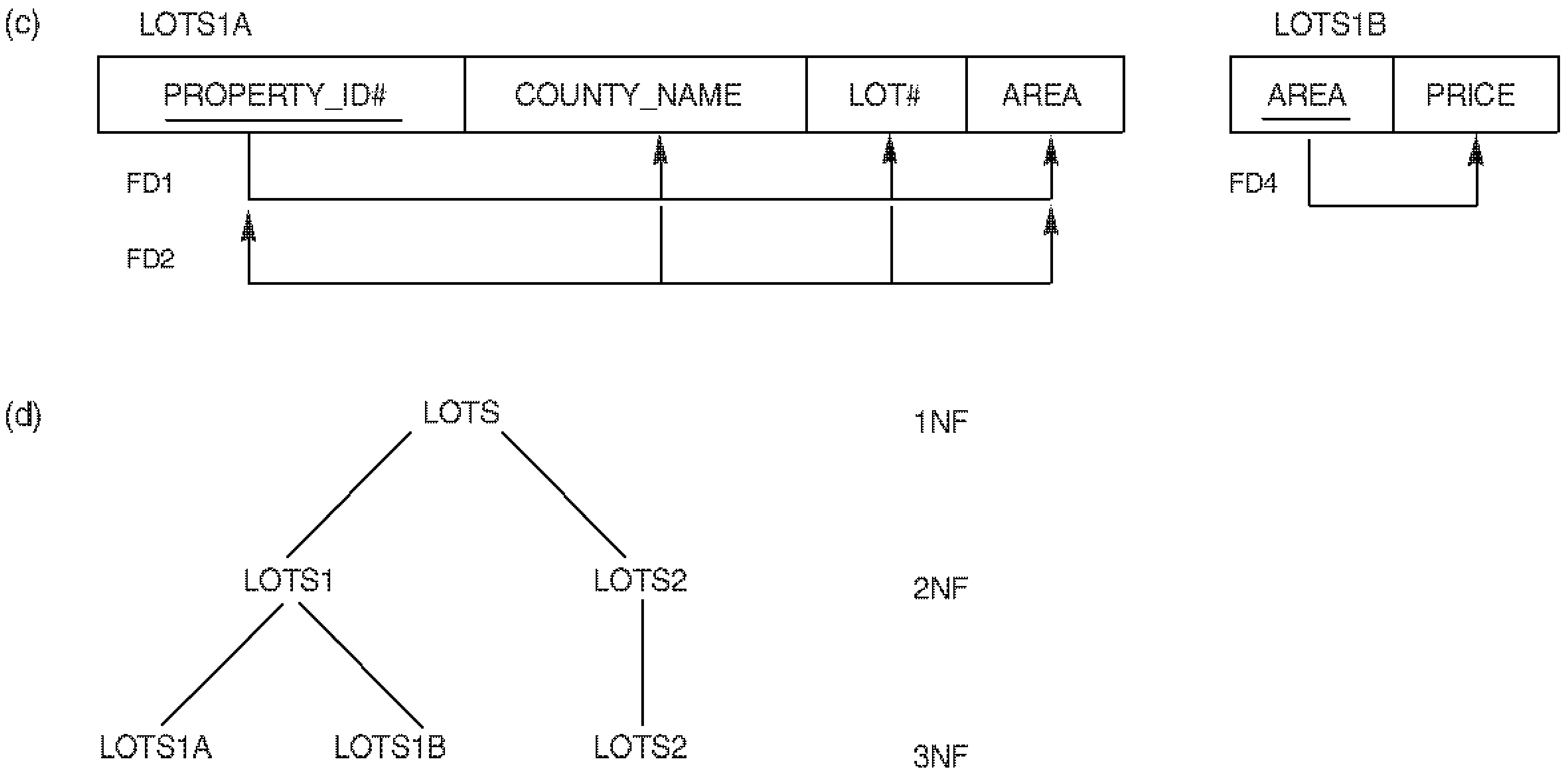


(b)
EMP_DEPT
ENAME
SSN
BDATE
ADDRESS
DNUMBER
DNAME
DMGRSSN
3NF NORMALIZATION
ED1
ENAME
SSN
BDATE
ADDRESS
DNUMBER
ED2
DNUMBER
DNAME
DMGRSSN

Exemplos (2)


(a)
LOTS
PROPERTY_ID#
COUNTY_NAME
LOT#
AREA
PRICE
TAX_RATE
FD1
FD2
FD3
FD4
(b)
LOTS1
PROPERTY_ID#
COUNTY_NAME
LOT#
AREA
PRICE
FD1
FD2
FD4
LOTS2
COUNTY_NAME
TAX_RATE
FD3

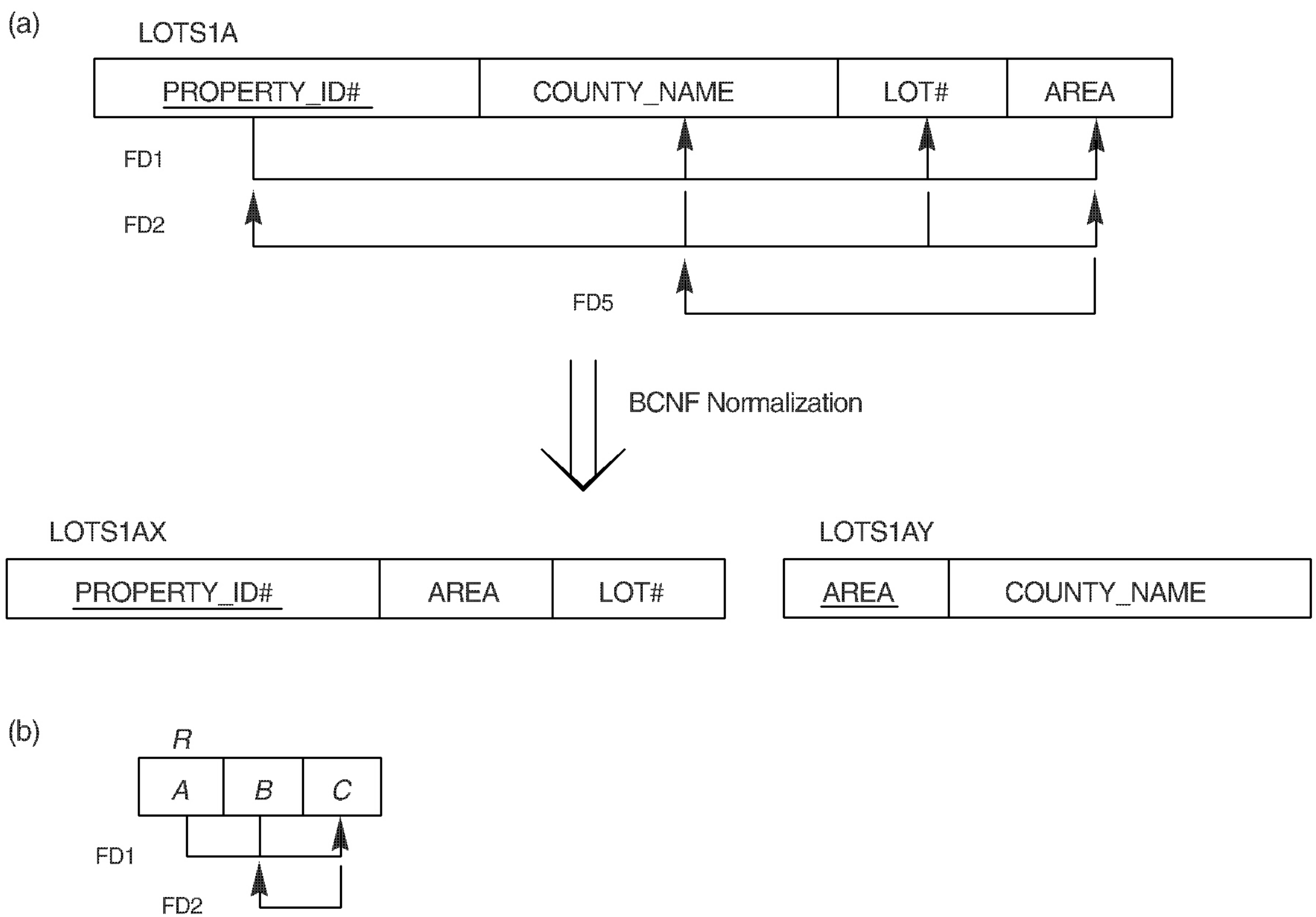
Exemplos (2)


(c)
LOTS1A
PROPERTY_ID#
COUNTY_NAME
LOT#
AREA
FD1
FD2
LOTS1B
AREA
PRICE
FD4
(d)
LOTS
LOTS1
LOTS2
LOTS1A
LOTS1B
LOTS2
1NF
2NF
3NF

Forma Normal de Boyce-Codd (FNBC)

- **Uma relação R está em FNBC se, para toda df não trivial $X \rightarrow Y$, válida para R, X é uma superchave de R**

Forma Normal de Boyce-Codd (FNBC)



Quarta Forma Normal (4FN)

- Livros(<u>num_tombo,autores,assuntos</u>)

Livros

<u>num_tombo</u>	<u>autores</u>	<u>assuntos</u>
31416	Aho	algoritmos
31416	Aho	complexidade
31416	Aho	busca
31416	Ullman	algoritmos
31416	Ullman	complexidade
31416	Ullman	busca

Quarta Forma Normal (4FN)

- Está em FNBC
- Grande redundância: um livro com k autores e m assuntos
  - k*m linhas

Quarta Forma Normal (4FN)

- **Dependência Funcional Multivalorada (multivalued dependency (MVD))**
  - **Seja R um esquema de relação e seja $\alpha \in R$ e $\beta \in R$. A dependência multivalorada $\alpha \twoheadrightarrow \beta$ realiza-se em R se, para qualquer relação r(R), para todos os pares de tuplas t_1 e t_2 de r, tal que $t_1[\alpha] = t_2[\alpha]$, existem tuplas t_3 e t_4 em r tal que**
    - $t_1[\alpha] = t_2[\alpha] = t_3[\alpha] = t_4[\alpha]$
    - $t_3[\beta] = t_1[\beta]$
    - $t_3[R - \alpha - \beta] = t_2[R - \alpha - \beta]$
    - $t_4[\beta] = t_2[\beta]$
    - $t_4[R - \alpha - \beta] = t_1[R - \alpha - \beta]$

Quarta Forma Normal (4FN)

- Dependência Funcional Multivalorada (multivalued dependency (MVD))
  - Seja R um esquema de relação e seja α R e β R. A dependência multivalorada α →→ β realiza-se em R se, para qualquer relação r(R), para todos os pares de tuplas t_1 e t_2 de r, tal que $t_1[\alpha] = t_2[\alpha]$, existam tuplas

R

	α	β	R – α - β
t1	$a_1 \ldots a_i$	$a_{i+1} \ldots a_j$	$a_{j+1} \ldots a_n$
t2	$a_1 \ldots a_i$	$b_{i+1} \ldots b_j$	$b_{j+1} \ldots b_n$
t3	$a_1 \ldots a_i$	$a_{i+1} \ldots a_j$	$b_{j+1} \ldots b_n$
t4	$a_1 \ldots a_i$	$b_{i+1} \ldots b_j$	$a_{j+1} \ldots a_n$

Quarta Forma Normal (4FN)

$t_1[\alpha] = t_2[\alpha] = t_3[\alpha] = t_4[\alpha]$

$t_3[\beta] = t_1[\beta]$

$t_3[R - \alpha - \beta] = t_2[R - \alpha - \beta]$

$t_4[\beta] = t_2[\beta]$

$t_4[R - \alpha - \beta] = t_1[R - \alpha - \beta]$

R

	α	β	$R - \alpha - \beta$
t1	$a_1 \ldots a_i$	$a_{i+1} \ldots a_j$	$a_{j+1} \ldots a_n$
t2	$a_1 \ldots a_i$	$b_{i+1} \ldots b_j$	$b_{j+1} \ldots b_n$
t3	$a_1 \ldots a_i$	$a_{i+1} \ldots a_j$	$b_{j+1} \ldots b_n$
t4	$a_1 \ldots a_i$	$b_{i+1} \ldots b_j$	$a_{j+1} \ldots a_n$

Quarta Forma Normal (4FN)

- **Regras de Inferência**
  - **IR1 (reflexibilidade): Se $X \supseteq Y$, então $X \rightarrow Y$.**
  - **IR2 (incremento/aumento): $\{X \rightarrow Y\} \models XZ \rightarrow YZ$.**
  - **IR3 (transitividade): $\{X \rightarrow Y, Y \rightarrow Z\} \models X \rightarrow Z$.**
  - **IR4 (complementação): $\{X \twoheadrightarrow Y\} \models \{X \twoheadrightarrow (R - (X \cup Y))\}$.**
  - **IR5 (incremento multivalorado): If $X \twoheadrightarrow Y$ e $W \supseteq Z$ então $WX \twoheadrightarrow YZ$.**
  - **IR6 (transitividade multivalorada): $\{X \twoheadrightarrow Y, Y \twoheadrightarrow Z\} \models X \twoheadrightarrow (Z - Y)$.**
  - **IR7 (replicação): $\{X \rightarrow Y\} \models X \twoheadrightarrow Y$.**
  - **IR8 (coalescência): Se $X \twoheadrightarrow Y$ e existe W com as propriedades (a) $W \cap Y$ é vazio, (b) $W \rightarrow Z$, e (c) $Y \supseteq Z$, então $X \rightarrow Z$.**

Quarta Forma Normal (4FN)

- **R(A,B,C,G,H,I)**
- **Suponha que A $\rightarrow\rightarrow$ BC realiza-se.**
- **Se $t_1[A]=t_2[A]$, então existem tuplas t_3 e t_4 tal que**
  - $t_1[A] = t_2[A] = t_3[A] = t_4[A]$
  - $t_3[BC] = t_1[BC]$
  - $t_3[GHI] = t_2[GHI]$
  - $t_4[BC] = t_2[BC]$
  - $t_4[GHI] = t_1[GHI]$
- **Regra da complementação: se A $\rightarrow\rightarrow$ BC então A $\rightarrow\rightarrow$ GHI. Observe que t_3 e t_4 satisfazem a definição A $\rightarrow\rightarrow$ GHI (mudam-se apenas os atributos)**

Quarta Forma Normal (4FN)

- **Uma MVD *X* —>> *Y* em *R* é chamada de trivial se (a) *Y* é um subconjunto de *X*, ou (b) *X* u *Y* = *R*.**
- **Uma relação R está em 4FN se para toda dependência mulivalorada não trivial X→→Y válida para R, então X é uma superchave de R**
  - **A relação R não está em 4FN pois num_tombo→→autores é uma dependência multivalorada não trivial, mas num_tombo não é uma superchave de R**

Quarta Forma Normal (4FN)

- Livros(num_tombo,autores,assuntos)

Livros

num_tombo	autores	assuntos
31416	Aho	algoritmos
31416	Aho	complexidade
31416	UAho	busca
31416	Ullman	algoritmos
31416	Ullman	complexidade
31416	Ullman	busca

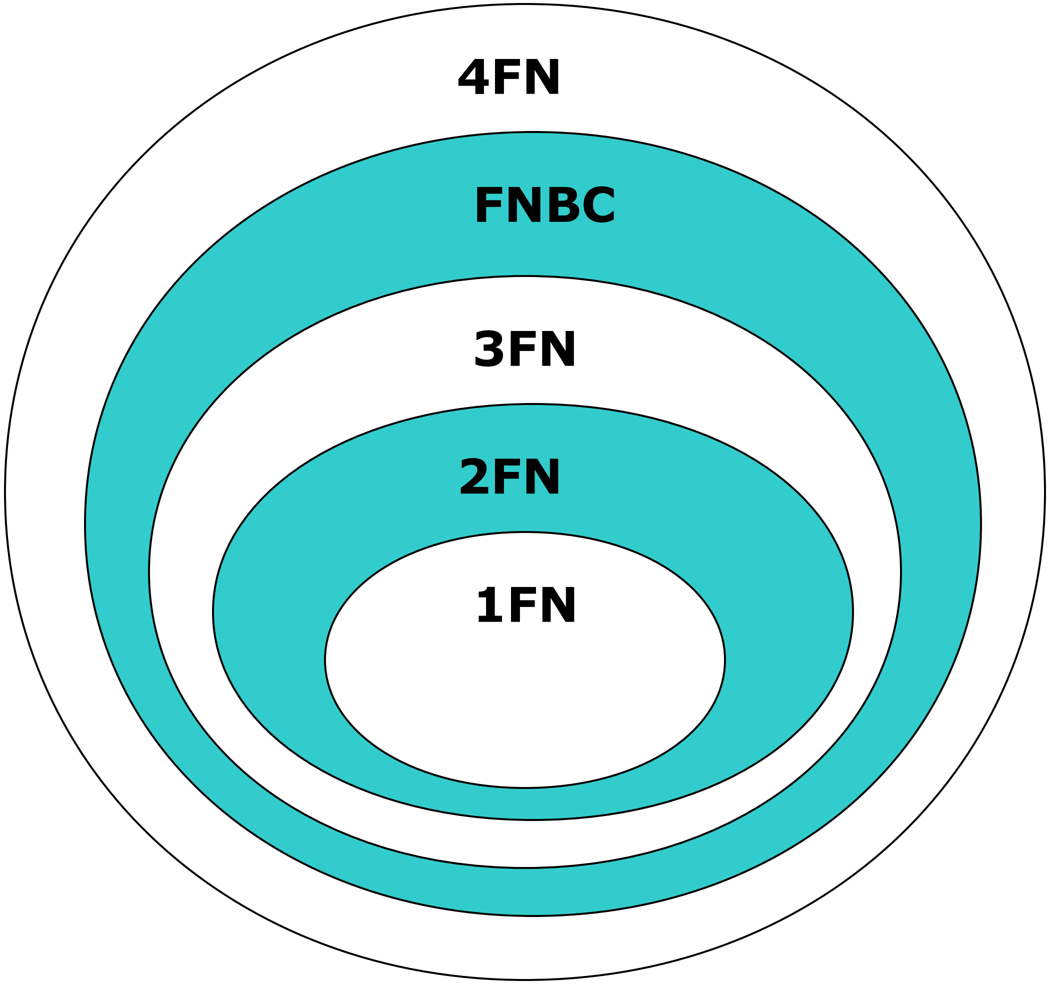
Quarta Forma Normal (4FN)

Solução:

- **Livros1(<u>num_tombo,autores</u>)**
- **Livros2(<u>num_tombo,assuntos</u>)**

- **k+m linhas**

Formas Normais



Quinta Forma Normal (5FN)
Project-Join Normal Form

- **Dependência de Junção (DJ)**
- **DJ(R_1,R_2,...,R_n) especificada na relação R**
  - **Toda instância legal r de R deveria ter uma decomposição de junção sem perda em R_1, R_2, ..., R_n**
  - $\pi_{R1}(R)\ |x|\ \pi_{R2}(R)\ |x| \ldots |X|\ \pi_{Rn}(R) = R$
  - **DJ trivial: um dos esquemas R_i em DJ(R_1,R_2,...,R_n) é igual a R**

Junção Sem-perdas

- R(A,B,C), com df $A \rightarrow B$

R

A	B	C
a1	b1	c1
a2	b1	c2

- Decomposição em duas relações R1 e R2
  - (R) e R_2 =

R_1

A	B
a1	b1
a2	b1

R_2

B	C
b1	c1
b1	c2

Junção Sem-perdas

- **Como os dados originais estão em R′, esperaríamos que R′ seria recuperada pela junção R_1 |X| R_2**

R_1 |X| R_2

A	B	C
a1	b1	c1
a1	b1	c2
a2	b1	c1
a2	b1	c2

- **No caso:**
  - **junção aditiva ou com perdas**

Quinta Forma Normal (5FN) Project-Join Normal Form

- **Uma relação R está em 5FN com respeito a um conjunto de dependências funcionais, multivaloradas e de junção se, para toda dependência de junção não-trivial DJ(R_1,R_2,...,R_n) em F^+, cada R_i é uma superchave de R**

Projeto do BD

- Projeto top-down
  - Modelo Entidade-Relacionamento
  - Mapeamento para modelo Relacional
  - Normalização
- Projeto bottom-up
  - Síntese relacional

Síntese Relacional

- **Esquema relacional universal $R=\{A_1,A_2,...A_n\}$**
  - **Todos os atributos de R**
  - **Atributo é único**
- **Conjunto F de dfs válidas**
- **Algoritmos para projeto**
  - **Decompõem R em um conjunto de relações $D=\{R_1,R_2,...,R_m\}$**
  - **D é chamado de decomposição de R**

Síntese Relacional

- **Preservação de atributos**
  - $\cup_{i=1}^{m} R_i = R$
  - **Todos os atributos em R aparecerão em pelo menos um esquema da relação R_i**

Preservação de Dependência Funcional

- Toda df $X \rightarrow Y$ especificada em F
  - Aparece em um dos esquemas R_i
  - Poderia ser inferida pelas dfs que aparecem em R_i
- Importância: continua a representar restrições no BD

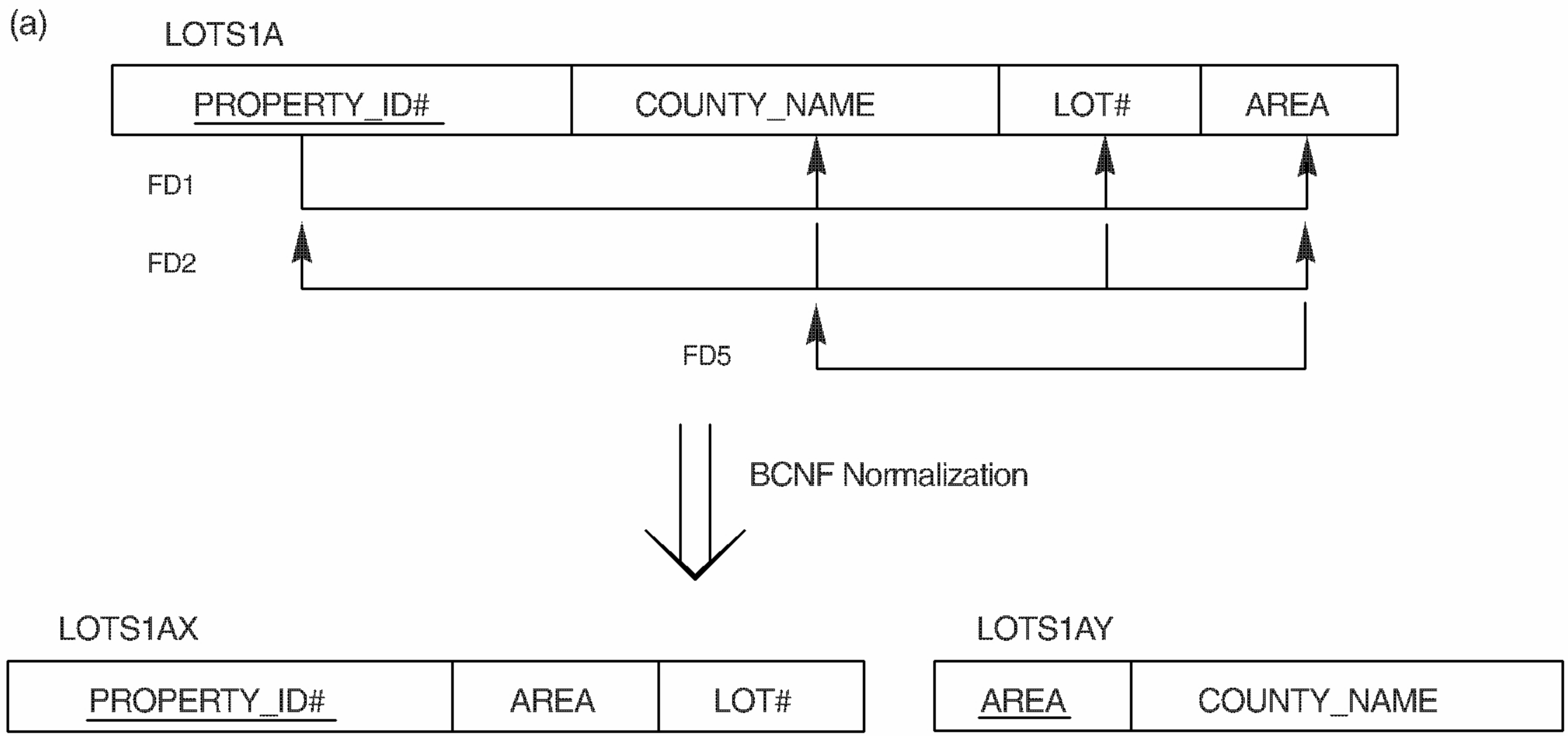
Preservação de Dependência Funcional

- Dado um conjunto de dfs F em R, a projeção de F em R_i, denotado por $\pi_F(R_i)$, onde Ri é um subconjunto de R, é o conjunto de dfs $X \rightarrow Y$ em F^+ tal que os atributos X u Y estão contidos em R_i
- Dizemos que uma decomposição D={R1,R2,...,Rm} de R preserva dependência com respeito a F se a união das projeções de F em Ri é equivalente a F
  - $(\pi_F(R_1) \cup \pi_F(R_2) \cup \ldots \cup \pi_F(R_m))^+ = F^+$

Fecho (Closure)

- Fecho de X: X^+
- **Algoritmo:**
  - $X^+ := X$
  - Repita
    - $oldX^+ := X^+$
    - Para toda DF $Y \rightarrow Z$ em F
      - Se X^+ contém Y então
        - $X^+ := X^+ \cup Z$
  - Até que $(oldX^+ = X^+)$

Preservação de Dependência Funcional



- **A dependência FD2 é perdida**

Equivalência de Conjuntos de Dfs

- Diz-se que um conjunto de dfs F cobre outro conjunto de dfs E se toda df em E também está em F^+
  - Toda df em E pode ser inferida a partir de F
  - E é coberto por F
- Dois conjuntos de dfs E e F são equivalentes se $E^+ = F^+$
  - E cobre F e F cobre E

Cobertura Mínima

- Seja F um conjunto de dfs
- Cobertura minimal G para F: um conjunto mínimo de df que é equivalente a F
  - 1. Cada df em G tem apenas um atributo do lado direito
  - 2. Não se pode remover nenhuma df de G e ainda ter um conjunto de dfs equivalentes a F
  - 3. Não se pode substituir nenhuma df $X \rightarrow A$ em G por uma df $Y \rightarrow A$, onde Y é subconjunto de X, e ainda ter o conjunto de dfs que é equivalente a F

Cobertura Mínima

- **Faça G:=F**
- **Substitua cada df $X \rightarrow A_1, A_2, \ldots, A_n$ em G por *n* dfs $X \rightarrow A_1$, $X \rightarrow A_2, \ldots,$ $X \rightarrow A_n$**
- **Para cada df $X \rightarrow A$ em G faça**
  - **Para cada atributo B que é um elemento de X faça**
    - **Compute $(X\text{-}B)^+$, considerando o conjunto de dfs G**
    - **Se $(X\text{-}B)^+$ contém A então substitua $X \rightarrow A$ por $(X\text{-}B) \rightarrow A$ em G**
- **Para cada df $X \rightarrow A$ restante em G faça**
    - **Compute X^+, considerando o conjunto de dfs $G-\{X \rightarrow A\}$**
    - **Se X^+ contém A então**
      - **Remova $X \rightarrow A$ de G**

Decomposição em relações 3FN que preserva dependência

- **Algoritmo de Síntese Relacional**
  - Encontre uma cobertura minimal G de F
  - Para cada X presente no lado esquerdo de uma df que aparece em G faça
    - Crie um esquema de relação $\{X \cup A_1 \cup A_2 \cup \ldots \cup A_m\}$ em D, onde $X \rightarrow A_1, A_2, \ldots, A_m$ são as únicas dfs em G que têm X do lado esquerdo
  - Coloque todos os atributos restantes em uma única relação para garantir a propriedade de preservação de atributo

Junção Sem-perdas

- Garante que nenhuma tupla espúria é gerada quando uma junção natural é aplicada às relações resultantes da decomposição
- Toda instância legal r de R deveria ter uma decomposição de junção sem perda em R_1, R_2, ..., R_n
  - $\pi_{R1}(r)\ |x|\ \pi_{R2}(r)\ |x| \ldots |X|\ \pi_{Rn}(r) = r$

Junção Sem-perdas

- **Relação universal *R*, uma decomposição *D* = {R_1, R_2, ..., R_m} de *R*, e um conjunto F de dfs**

1. **Crie uma matriz inicial S com uma linha *i* para cada relação R_i em *D*, em uma coluna *j* para cada atributo A_j em *R***
2. **Faça $S(i,j)$:=b_{ij} para todas as células da matriz (* cada b_{ij} é um símbolo distinto associado aos ínidices (i,j) *).**
3. **Para cada linha *i* representando um esquema de relação R_i**

   {Para cada coluna j representando um atributo A_j

   {se (relação R_i inclui atributo A_j) então faça $S(i,j)$:= a_j;};};

(* cada a_j é um símbolo distinto associado ao índice (*j*) *)

Junção Sem-perdas

1. **Repita o loop abaixo até que a execução do loop não provoque mudanças em *S***
   1. **Para cada df *X* →*Y* em *F***
      1. **Para todas as linhas em *S* que tenham os mesmos símbolos nas colunas correspondentes a X**
         1. **Faça os símbolos em cada coluna que corresponde a um atributo em *Y* ser o mesmo em todas estas linhas como definido a seguir: se qualquer duas linhas têm um símbolo "*a*" para a coluna, faça as outras linhas terem o mesmo símbolo "*a*" na coluna. Se não há um símbolo "*a*" para o atributo em qualquer das linhas, escolha um dos símbolos "*b*" que aparecem em uma das linhas para o atributo e faça as outras linhas receberem o mesmo símbolo "*b*" na coluna**
2. **Se uma linha só tem símbolos "*a*", então a decomposição tem a propriedade de ser sem-perda; caso contrário a decomposição é com perdas.**

Junção Sem-perdas

(a) R={SSN, ENAME, PNUMBER, PNAME, PLOCATION, HOURS} $D=\{R_1, R_2\}$

R_1=EMP_LOCS={ENAME, PLOCATION}

R_2=EMP_PROJ1={SSN, PNUMBER, HOURS, PNAME, PLOCATION}

F={SSN→ENAME;PNUMBER→{PNAME, PLOCATION} ;{SSN,PNUMBER}→HOURS}

	SSN	ENAME	PNUMBER	PNAME	PLOCATION	HOURS
R_1	b_{11}	a_2	b_{13}	b_{14}	a_5	b_{16}
R_2	a_1	b_{22}	a_3	a_4	a_5	a_6

(no changes to matrix after applying functional dependencies)

(b)

EMP

SSN	ENAME

PROJECT

PNUMBER	PNAME	PLOCATION

WORKS_ON

SSN	PNUMBER	HOURS

Junção Sem-perdas

(c) R={SSN, ENAME, PNUMBER, PNAME, PLOCATION, HOURS} D={R_1, R_2, R_3}

R_1=EMP={SSN, ENAME}

R_2=PROJ={PNUMBER, PNAME, PLOCATION}

R_3=WORKS_ON={SSN, PNUMBER, HOURS}

F={SSN→{ENAME;PNUMBER→{PNAME, PLOCATION} ;{SSN,PNUMBER}→HOURS}

	SSN	ENAME	PNUMBER	PNAME	PLOCATION	HOURS
R_1	a_1	a_2	b_{13}	b_{14}	b_{15}	b_{16}
R_2	b_{21}	b_{22}	a_3	a_4	a_5	b_{26}
R_3	a_1	b_{32}	a_3	b_{34}	b_{35}	a_6

(original matrix S at start of algorithm)

	SSN	ENAME	PNUMBER	PNAME	PLOCATION	HOURS
R_1	a_1	a_2	b_{13}	b_{14}	b_{15}	b_{16}
R_2	b_{21}	b_{22}	a_3	a_4	a_5	b_{26}
R_3	a_1	$\cancel{b_{32}}$ a_2	a_3	$\cancel{b_{34}}$ a_4	$\cancel{b_{35}}$ a_5	a_6

(matrix S after applying the first two functional dependencies - last row is all "a" symbols, so we stop)

Síntese Relacional

- **Propriedade 1**
  - **Uma decomposição $D=\{R_1,R_2\}$ de R é não-aditiva (sem perda) obedecendo a um conjunto de dfs F em R se, e somente se,**
    - **Ou a df $((R_1 \cap R_2) \rightarrow (R_1 - R_2)$ está em F^+**
    - **Ou a df $((R_1 \cap R_2) \rightarrow (R_2 - R_1)$ está em F^+**
  - **Propriedade aplicável para decomposições em duas relações**
- **Ex.: Funcionário(<u>numf</u>,RG,nome,end,depto,nome_depto)**
  - **numf e RG são chaves**
  - **depto → nome_depto**
- **A decomposição D={F1(numf,RG,nome,end,depto), F2(depto,nome_depto)} é não-aditiva**
  - **$F1 \cap F2$ = depto, F2-F1=nome_depto**
  - **$(F1 \cap F2) \rightarrow$ F2-F1**

Síntese Relacional

- **Propriedade 2**
  - **Se uma decomposição $D=\{R_1,R_2,\ldots,R_m\}$ de R tem a propriedade de junção sem perda em relação a um conjunto F de dfs em R, e se uma decomposição $D1=\{Q_1,Q_2,\ldots,Q_k\}$ de R_i tem a propriedade de junção sem perda com relação a projeção de F em R_i, então a decomposição $D2=\{R_1,R_2,\ldots,R_{i-1}, Q_1,Q_2,\ldots,Q_k,R_{i+1},\ldots,R_m\}$ é sem perda**

Síntese Relacional

- Toda relação R obedecendo a um conjunto de dfs F e que não esteja em FNBC, possui uma decomposição não-aditiva em relações obedecendo a FNBC
- Não garante que as dfs serão preservadas

Algoritmo para Decomposição Sem-perdas em Relações FNBC

- **Faça D:={R}**
- **Enquanto houver uma relação Q em D que não esteja em FNBC faça**
  - **Escolha uma relação Q em D que não está em FNBC**
  - **Encontre uma df X→Y em Q que viola FNBC**
  - **Troque Q em D por dois esquemas (Q-Y) e (X u Y)**

Algoritmo para Decomposição Sem-perdas em Relações 3FN que Preservam Dependência

- **Encontre a cobertura minimal G para F**
- **Para cada atributo X que aparece do lado esquerdo em G faça**
  - **Crie um esquema de relação $\{X \cup A_1 \cup A_2 \cup \ldots \cup A_m\}$ em D, onde $X \rightarrow A_1, A_2, \ldots, A_n$ são as únicas dfs em G que têm X do lado esquerdo**
- **Coloque todos os atributos restantes em uma única relação para garantir a propriedade de preservação de atributo**
- **Se nenhuma das relações contêm uma chave de R, crie uma ou mais relações que contenham atributos que formam uma chave para R**

Algoritmo para se determinar a chave K de uma relação R

- K:=R
- Para cada atributo A em K
  - Compute $(K-A)^+$ com respeito ao conjunto de dfs
  - Se $(k-A)^+$ contém todos os atributos em R, então $K:=K-\{A\}$
- Chave: depende da ordem na qual os atributos são removidos