Banco de Dados

Módulo 5 - Modelo Relacional

Normalização

Pontifícia Universidade Católica do Rio de Janeiro

Tópicos

- Introdução a Normalização
- Teoria da Normalização

Introdução a Normalização

- Normalização:
  - Processo de ***decomposição*** de um esquema de relação em outros esquemas de relação que:
    - não têm informação redundante, que poderia gerar problemas de atualização
    - mantem a semântica original dos dados:
      - mesmo conjunto de restrições de integridade
      - sem perda de dados e seus relacionamentos

Introdução a Normalização

- Principais conceitos envolvidos:
  - Dependências funcionais (FDs):
    - capturam a semântica dos dados
  - Junção natural:
    - ajuda a capturar o conceito de perda de dados
  - Formas Normais (NFs):
    - definem quando um esquema de relação está livre de (certas) redundâncias
  - Algoritmos de decomposição:

    entrada: esquema de relação R + conjunto de FDs sobre R
    saída: conjunto de esquemas de relação em uma dada NF
    propriedade: mantém a semântica original dos dados

Introdução a Normalização

- Exemplo:
  - esquema de relação inicial:

    Turma[<u>Cod, Id, End</u>,
    CPF not null, Nome]

    Turma: Cod, Id → CPF
    Turma: CPF → Nome

Turma	Cod	Id	End	CPF	Nome
	inf1731	A	L520	299978	Pedro
	inf1731	A	L512	299978	Pedro
	inf2324	A	L520	317496	Manuel
	inf1732	A	L510	317496	Manuel

- Problema 1:
  - CPF **depende de parte da chave** de Turma, pois vale
    Turma: Cod, Id → CPF
  - ou seja, Turma não está na *Segunda Forma Normal (2NF)*
  - logo, há redundância na representação de CPF em Turma
  - intuitivamente, o professor responsável pela Turma é representado repetidamente, uma vez para cada sala

Introdução a Normalização

- Exemplo (cont.):
  - decomposição 1
    ("Resp extraído de Turma"):

    Turma[Cod, Id, End]
    Resp[Cod, Id, CPF not null, Nome]

    Resp: Cod, Id → CPF
    Resp: CPF → Nome

- Problema 2:
  - Nome **depende transitivamente** da chave de Resp, pois valem
    Resp: Cod, Id → CPF e
    Resp: CPF → Nome
  - ou seja, Resp não está na *Terceira Forma Normal (3NF)*

Turma	Cod	Id	End
	inf1731	A	L520
	inf1731	A	L512
	inf2324	A	L520
	inf1732	A	L510

Resp	Cod	Id	CPF	Nome
	inf1731	A	299978	Pedro
	inf2324	A	317496	Manuel
	inf1732	A	317496	Manuel

Introdução a Normalização

- Exemplo (cont.):
  - decomposição 2 ("Prof extraído de Resp"):

    Turma[<u>Cod, Id, End</u>]
    Resp[<u>Cod, Id</u>, CPF not null]
    Prof[<u>CPF</u>, Nome]

    Resp: Cod, Id → CPF
    Prof: CPF → Nome

- Propriedades:
  - todas as FDs são conseqüência das chaves dos esquemas, ou seja, os esquemas estão em *Boyce-Codd Normal Form (BCNF)*
  - FDs originais estão representadas
  - relação original pode ser reconstruída por junção natural

Turma	Cod	Id	End
	inf1731	A	L520
	inf1731	A	L512
	inf2324	A	L520
	inf1732	A	L510

Resp	Cod	Id	CPF
	inf1731	A	299978
	inf2324	A	317496
	inf1732	A	317496

Prof	CPF	Nome
	299978	Pedro
	317496	Manuel

Introdução a Normalização

- Exemplo (cont.):
  - decomposição final:

Turma[Cod, Id, End]
Resp[Cod, Id, CPF not null]
Prof[CPF, Nome]

Resp: Cod, Id → CPF
Prof: CPF → Nome

Turma	Cod	Id	End	CPF	Nome
	inf1731	A	L520	299978	Pedro
	inf1731	A	L512	299978	Pedro
	inf2324	A	L520	317496	Manuel
	inf1732	A	L510	317496	Manuel

Turma	Cod	Id	End
	inf1731	A	L520
	inf1731	A	L512
	inf2324	A	L520
	inf1732	A	L510

⋈

Resp	Cod	Id	CPF
	inf1731	A	299978
	inf2324	A	317496
	inf1732	A	317496

⋈

Prof	CPF	Nome
	299978	Pedro
	317496	Manuel

=

Teoria da Normalização

- Dependência Funcional:
  - uma *dependência funcional* para R

    é uma expressão da forma R: X→Y

    onde X e Y são listas de atributos de R
  - um estado **s** *é consistente* com R: X→Y sse

    para quaisquer t e u em **s**(R), se t[X] = u[X] então t[Y] = u[Y]

- Exemplo:

  Turma[<u>Cod, Id, End</u>, CPF, Nome]

  Turma: Cod, Id → CPF
  Turma: CPF → Nome

inf1731	A	L520	299978	Joaquim
inf1731	A	L512	299978	Joaquim
inf2324	A	L520	319732	Manoel

Teoria da Normalização

- Conseqüência lógica de um conjunto de dependências:
  - Seja Σ um conjunto de dependências sobre R.
    Seja σ uma dependência sobre R.
  - Dizemos que σ é *conseqüência lógica* de Σ, denotado $\Sigma \models \sigma$, sse todo estado consistente com todas as dependências em Σ também é consistente com σ

- Exemplo:

Turma[Cod, Id, End, CPF, Nome]

Σ = { Turma: Cod, Id → CPF
Turma: CPF → Nome }

σ = Turma: Cod, Id → Nome

$\Sigma \models \sigma$

inf1731	A	L520	299978	Joaquim
inf1731	A	L512	299978	Joaquim
inf2324	A	L520	319732	Manoel

Teoria da Normalização

- Sistema de dedução **A** para dependências funcionais:

  - Axioma:

    – Reflexividade: $\vdash R: X \rightarrow Y$

    onde X é qualquer lista de atributos de R
    e Y é qualquer sublista de X

  - Regras de Dedução:

    – Aumento: $R: X \rightarrow Y \vdash R: X,Z \rightarrow Y,Z$

    onde Z é uma lista de atributos de R

    – Transitividade: $R: X \rightarrow Y,\ R: Y \rightarrow Z \vdash R: X \rightarrow Z$

Teoria da Normalização

- Dedução:

  - uma dedução de σ a partir de Σ, usando **A**, é uma seqüência de dependências funcionais $\sigma_1, ..., \sigma_n$ tal que $\sigma_n = \sigma$ e, para cada dependência σ_i:

    - σ_i ocorre em Σ, ou
    - σ_i é um axioma de **A**, ou
    - σ_i pode ser deduzida de σ_j , onde $j < i$, usando a regra de aumento, ou
    - σ_i pode ser deduzida de σ_j e σ_k, onde $j,k < i$, usando a regra de transitividade

Teoria da Normalização

- Exemplo:

Turma[Cod, Id, End, CPF, Nome]

Σ = { Turma: Cod, Id → CPF, Turma: CPF → Nome }
σ = Turma: Cod, Id, End → Nome, End

Dedução de d a partir de **D**:

1.	Turma: Cod, Id → CPF	(em Σ)
2.	Turma: CPF → Nome	(em Σ)
3.	Turma: Cod, Id → Nome	(de 1 e 2 por transitividade)
4.	Turma: Cod, Id, End → Nome, End	(de 3 por aumento)

Teoria da Normalização

- Teorema de um conjunto de dependências:

  - Seja Σ um conjunto de dependências funcionais sobre R. Seja σ uma dependência funcional sobre R.
  - Dizemos que d é um *teorema* de Σ em **A**, denotado $\Sigma \vdash \sigma$, sse existe uma dedução de σ a partir de Σ usando **A**

- Propriedade fundamental:

$$\Sigma \models \sigma \quad \text{sse} \quad \Sigma \vdash \sigma$$

Teoria da Normalização

- Fecho de um conjunto de dependências funcionais:

  - Seja R um esquema de relação
    Σ um conjunto de dependências sobre R.

  - O *fecho* de Σ**,** denotado Σ^{+}, é o conjunto de todas as dependências funcionais R: $X \rightarrow Y$

    tais que $\Sigma \models R: X \rightarrow Y$ (ou, equivalentemente, $\Sigma \vdash R: X \rightarrow Y$)

Teoria da Normalização

- Fecho de um conjunto de atributos:

  - Seja R um esquema de relação
Σ um conjunto de dependências sobre R
e X uma lista de atributos de R.

  - O *fecho* de X por Σ**,** denotado X^+,
é o conjunto de todos os atributos A de R
tais que $\Sigma \models R: X \rightarrow A$ (ou, equivalentemente, $\Sigma \vdash R: X \rightarrow A$)

  - Um atributo A *depende funcionalmente* de X (em presença de Σ)
sse A pertence ao fecho de X por Σ**.**

Teoria da Normalização

- Exemplo:

  Turma[Cod, Id, End, CPF, Nome]

  Σ = { Turma: Cod, Id $\rightarrow$ CPF , Turma: CPF $\rightarrow$ Nome }

  X = Cod, Id

  X^+ = { Cod, Id, CPF, Nome }

Logo,

Nome depende funcionalmente de Cod, Id

Teoria da Normalização

- Superchave, chave candidata, atributos primos:

  - Seja R um esquema de relação
    Σ o conjunto de dependências definidos para R.

  - Uma lista K de atributos de R é uma *superchave de* R *por* Σ
    sse o fecho de K por Σ contém todos os atributos de R

  - Uma superchave K de R é uma *chave candidata de* R *por* Σ
    sse nenhuma sublista de K é uma superchave de R por Σ

  - Um atributo A de R é *primo* por Σ
    sse pertence a alguma chave candidata de R por Σ;
    caso contrário, o atributo é *não-primo* por Σ.

Teoria da Normalização

- Junção Natural:

  - Sejam **B**=(B_1 ,...,B_m) e **C**=(C_1 ,...,C_n) listas de atributos.
  - Sejam S[**B**] e T[**C**] esquemas de relação sobre **B** e **C**.
  - Suponha que **B** e **C** tenham k atributos em comum.
  - Seja **s** um estado para S e T.

  - A expressão S ⋈ T denota a *junção natural* de S e T

  - Estende-se **s** para S ⋈ T de tal forma que
    s(S ⋈ T) é o conjunto de todas as tuplas
    t = (t_1 ,...,t_k,t_{k+1} ,...,t_{m+n-k}) tais que t[**B**] ∈ **s**(S) e t[**C**] ∈ **s**(T)

Teoria da Normalização

- Exemplo:

Cod	Id	CPF
inf1731	A	299978
inf2324	A	317496

$\bowtie$

Cod	Id	End
inf1731	A	L520
inf1731	A	L512
inf2324	A	L520
inf1732	A	L510

=

Cod	Id	End	CPF
inf1731	A	L520	299978
inf1731	A	L512	299978
inf2324	A	L520	317496

Teoria da Normalização

- Notação:

  - Sejam R[**A**], S[**B**] e T[**C**] esquemas de relação
    e Σ_R , Σ_S e Σ_T os conjuntos de FDs de R, S e T.

  - Seja **X** = **B** $\cap$ **C** a lista de atributos comuns a **B** e **C.**

  - Seja **s** um estado para R.

Teoria da Normalização

- Decomposição:
  - S e T formam uma *decomposição de* R sse
    - cada atributo de R ocorre em S ou em T
    - todos os atributos de S e T ocorrem em R
  - Estende-se o estado **s** de R para S e T de tal forma que **s**(S)=**s**(R[**B**]) e **s**(T)=**s**(R[**C**])

Teoria da Normalização

- Decomposição preservando FDs:
  - A decomposição de R em S e T *preserva* Σ_R sse
    - para cada FD da forma R: $X \rightarrow Y$ em Σ_R ,
      existe uma FD da forma S: $X \rightarrow Y$ em Σ^+_S , ou
      existe uma FD da forma T: $X \rightarrow Y$ em Σ^+_T
    - para cada FD da forma S: $X \rightarrow Y$ em Σ_S ,
      existe uma FD da forma R: $X \rightarrow Y$ em Σ^+_R
    - para cada FD da forma T: $X \rightarrow Y$ em Σ_T ,
      existe uma FD da forma R: $X \rightarrow Y$ em Σ^+_R

Teoria da Normalização

- Decomposição sem perdas:

  - S e T formam uma *decomposição sem perdas* de R
    sse para todo estado **s** de R consistente com Σ_R
    s(R) = **s**(S ⋈ T), ou seja, **s**(R)=**s**(R[**B**] ⋈ R[**C**])

- Teorema:

  - S e T formam uma decomposição sem perdas de R sse
    existe uma FD da forma S: $\mathbf{X} \rightarrow \mathbf{B}$ em Σ^{+}_{S} , ou
    existe uma FD da forma T: $\mathbf{X} \rightarrow \mathbf{C}$ em Σ^{+}_{T}

Teoria da Normalização

- Esquema inicial:

  Turma[Cod, Id, CPF, Nome]

  Turma: Cod, Id → CPF

  Turma: CPF → Nome

- Decomposição sem perdas e preservando as FDs:

  Resp[<u>Cod, Id</u>, CPF not null]
  Prof[<u>CPF</u>, Nome]

  Resp: Cod, Id → CPF

  Prof: CPF → Nome

Turma	Cod	Id	CPF	Nome
	inf1731	A	299978	Pedro
	inf1731	B	299978	Pedro
	inf2324	A	317496	Manuel
	inf1732	A	317496	Manuel

=

Resp	Cod	Id	CPF
	inf1731	A	299978
	inf1731	B	299978
	inf2324	A	317496
	inf1732	A	317496

$\bowtie$

Prof	CPF	Nome
	299978	Pedro
	317496	Manuel

= (relação original)

Teoria da Normalização

- Esquema inicial:

  Turma[Cod, Id, CPF, Nome]

  Turma: Cod, Id $\rightarrow$ CPF

  Turma: CPF $\rightarrow$ Nome

- Decomposição com perdas:

  Resp[Cod, Id]
  Prof[Cod, CPF, Nome]

  Prof: CPF $\rightarrow$ Nome

Turma	Cod	Id	CPF	Nome
	inf1731	A	299978	Pedro
	inf1731	B	317496	Pedro
	inf1731	A	317496	Manuel
	inf1731	B	317496	Manuel
	inf2324	A	317496	Manuel
	inf1732	A	317496	Manuel

=

Resp	Cod	Id
	inf1731	A
	inf1731	B
	inf2324	A
	inf1732	A

$\bowtie$

Prof	Cod	CPF	Nome
	inf1731	299978	Pedro
	inf1731	317496	Manuel
	inf2324	317496	Manuel
	inf1732	317496	Manuel

(contém tuplas não pertencentes à relação original)

Teoria da Normalização

- Primeira Forma Normal (1NF)
  - Uma relação R está em 1NF
    sse todos os atributos são atômicos ou indivisíveis

- Contra-exemplos:

Turma[Cod, Id, {Salas}]

inf1731	A	L512 L520
inf1732	A	L520

Turma[Cod, Id, Salas[End,Capacidade]]

inf1731	A	512	25
		520	30
inf1732	A	520	30

Teoria da Normalização

- Segunda Forma Normal (2NF):

  - Seja R um esquema de relação **e**
    Σ o conjunto de dependências definidos para R.

  - R está em 2NF sse estiver em 1NF e
    nenhum atributo não-primo de R por Σ depende funcionalmente de uma parte de uma chave candidata de R por Σ

- Contra-exemplo:

Turma[Cod, Id, End, CPF]

Turma: Cod, Id → CPF

Atrib. não-primo: CPF
Violação de 2NF: Turma: Cod, Id → CPF

redundância...

inf1731	A	L520	299978
inf1731	A	L512	299978
inf2324	A	L520	319732

Teoria da Normalização

- Terceira Forma Normal (3NF):

  - Seja R um esquema de relação **e**
    Σ o conjunto de dependências definidos para R.

  - R está em 3NF sse estiver em 1NF
    e para toda FD não-trivial R: $X \rightarrow A$ em Σ^+,
    X é superchave de R por Σ ou A é atributo primo de R por Σ

- Contra-exemplo:

  Turma[<u>Cod, Id</u>, CPF, Nome]

  Turma: CPF → Nome

FD não-trivial: Turma: CPF → Nome

Violação de 3NF: CPF não é superchave e Nome não é primo

redundância...

inf1731	A	299978	Joaquim
inf1732	A	299978	Joaquim
inf2324	A	319732	Manoel

Teoria da Normalização

- Boyce-Codd Normal Form (BCNF):
  - Seja R um esquema de relação **e**
    Σ o conjunto de dependências definidos para R.
  - R está em BCNF sse estiver em 1NF
    e para toda FD não-trivial R: $X \rightarrow A$ em Σ^+,
    X é superchave de R por Σ
- Contra-exemplo:

Turma[<u>Cod, Id</u>, CPF, Nome]

Turma: CPF → Nome

FD não-trivial: Turma: CPF → Nome
Violação BCNF: CPF não é superchave

redundância...

inf1731	A	299978	Joaquim
inf1732	A	299978	Joaquim
inf2324	A	319732	Manoel

Teoria da Normalização

- Algoritmo de decomposição em BCNF:

  - Entrada: R um esquema de relação
    Σ o conjunto de FDs para R

  - Saída: um conjunto de esquemas de relação e FDs

  Nota:

  FD(S) denota o conjunto de FDs de S
  Fecho(S) denota o fecho das FDs de S

Teoria da Normalização

definição de BCNF

otimização

Decomposição de S em S' e S''

Resultado := {R};
OK := falso;
Compute *Fecho*(R);
while (**not** *OK*) **do**
  if (há um esquema S em *Resultado* que não está em BCNF)
    then begin
      Seja S: X → Y uma FD não trivial em *Fecho*(S) tal que:
        X não é uma superchave de S por *FD*(S) e X ∩ Y = ∅
      Remova S de *Resultado*;
      Acrescente S' a *Resultado,* onde S' é igual a S, sem os atributos em Y;
      Construa *FD*(S') com todas as dependências de S
        que não envolvem atributos em Y;
      Compute *Fecho*(S');
      Acrescente S''[<u>X</u>,Y] a *Resultado;*
      Construa *FD*(S'') com todas as dependências de S
        que envolvem apenas os atributos em X e Y;
      Compute *Fecho*(S'');
    end
  else *OK* := true

Teoria da Normalização

- Propriedades do algoritmo de decomposição:
  - a decomposição de S em S' e S''
    é sem perdas, ou seja, é uma lossless decomposition
  - a decomposição de S em S' e S''
    pode não preservar dependências funcionais

Teoria da Normalização

- Problema:
  - nem sempre é possível decompor uma esquema de relação em outros que estão em BCNF e que preservam as FDs originais
  - Exemplo:

    Alocação[Depto, Disc, Resp]

    Alocação: Depto, Disc → Resp   Alocação: Resp → Depto

    - Fatos:
      - Alocação não está em BCNF pois Resp não é superchave
      - Alocação não pode ser decomposta preservando todas as FDs

    Alocação[Depto, Disc]   Resp[Resp, Depto]

    ~~Alocação: Depto, Disc → Resp~~   Resp: Resp → Depto