

FGA0137

# Sistemas de Banco de Dados 1

Prof. Maurício Serrano

Material original: Prof. Jose Fernando Rodrigues Junior

**2021/2**

# Dependências Funcionais e Normalização

Módulo 3

# Introdução: Contexto da Normalização

# Projeto de bancos de dados

- Objetivos de projeto:
  - **Completo:** todos os requisitos
  - **Correto:** sem erros de modelagem
  - **Fácil de Entender:** dados e relacionamentos claros e expressivos
  - **Simples:** o mais simples possível atendendo aos requisitos
  - **Livre de redundâncias e consistente**

# Projeto de bancos de dados

- Objetivos de projeto:
  - **Completo:** todos os requisitos
  - **Correto:** sem erros de modelagem
  - **Fácil de Entender:** dados e relacionamentos claros e expressivos
  - **Simples:** o mais simples possível atendendo os requisitos
  - Livre de redundâncias e consistente
- Uma das principais técnicas (mas não a única) para evitar redundâncias e garantir consistência: **Normalização**

# Problema

- Projetos não bem elaborados -> anomalias:
  - **Redundância**: espaço desperdiçado por duplicidade de dados
  - **Complexidade desnecessária**: presença de valores NULL
  - **Junções com perda**: perda de dados em operações de junção
  - **Falta de integridade!!!**
    - Inconsistência de inserção
    - Inconsistência de remoção
    - Inconsistência de atualização
- Solução: **NORMALIZAÇÃO** → o SGBD garante a integridade

# Problema

- Projetos não bem elaborados -> anomalias:
  - **Redundância**: espaço desperdiçado por duplicidade de dados
  - **Complexidade desnecessária**: presença de valores NULL
  - **Junções com perda**: perda de dados em operações de junção
  - **Falta de integridade!!!**
    - Inconsistência de inserção
    - Inconsistência de remoção
    - Inconsistência de atualização
- Solução: **NORMALIZAÇÃO** → o SGBD garante a integridade

# Normalização: ferramenta conceitual

- **Modelo Relacional**, inerentemente formal
  - Ferramentas **conceituais** → gerência de consistência
  - Projeto deve satisfazer propriedades bem definidas → **Normalização**



# Normalização: ferramenta conceitual

- **Modelo Relacional**, inerentemente formal

Após alguma prática, a normalização torna-se uma técnica bastante intuitiva

A teoria de normalização provê:

- uma maneira formal de melhoria de projeto
- desenvolve a intuição de projetos de melhor qualidade

# Conceitos

# Revisão de conceitos

- **Superchave:** conjunto com um ou mais atributos que identifica uma tupla unicamente
- **Superchave mínima:** superchave que, se tiver um atributo removido, deixa de ser superchave
- **Chave candidata:** qualquer uma das superchaves mínimas existente em uma relação
- **Atributo primo:** pertence a uma chave candidata
- **Atributo comum ou ordinário:** atributo não primo

# Dependência funcional

- O valor de um conjunto de atributos A permite descobrir o valor de um outro conjunto B, dizemos que A determina funcionalmente B, ou que B depende de A, e denotamos:  $A \rightarrow B$
- Exemplos:
  - Matr  $\rightarrow$  Nome, Curso
  - Sala, Dia, Hora  $\rightarrow$  CodigoDisciplina
  - CodigoDisciplina  $\rightarrow$  Nome, Ementa, Ncréditos
- Chaves determinam funcionalmente todos os outros atributos, mas nem toda dependência funcional parte de uma chave

# Dependência funcional

- O valor de um conjunto de atributos A permite descobrir o valor de um outro conjunto B, dizemos que A determina funcionalmente B, ou que B

Pergunta para identificar DF:

“Se o valor de A se repetir, o valor de B também se repete necessariamente?”

- Sim: há DF
- Não: não há DF

Código Disciplina → Nome, Ementa, Recursos

- Chaves determinam funcionalmente todos os outros atributos, mas nem toda dependência funcional parte de uma chave

# Dependência funcional parcial

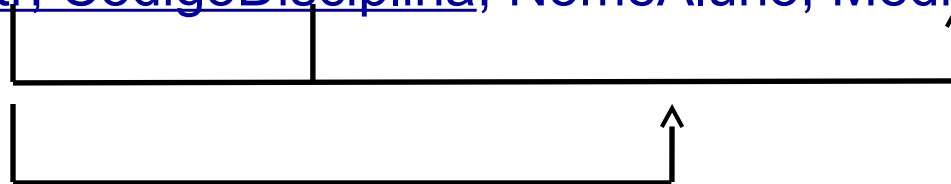
- Seja A um conjunto de atributos
  - Dados dois conjuntos de atributos A e B; se um **subconjunto de atributos** de A define funcionalmente o conjunto de atributos B, dizemos que B possui **dependência funcional parcial** em relação a A.
- Exemplos:
  - Matricula(Mat, CodigoDisciplina, NomeAluno, Média)
    - $\text{Matr, CodigoDisciplina} \rightarrow \text{Média}$ 
      - DF
    - $\text{Matr} \rightarrow \text{NomeAluno}$ 
      - DF parcial de NomeAluno em relação a  $\{\text{Matr, CodigoDisciplina}\}$

# Dependência funcional parcial

- Seja A um conjunto de atributos
  - Dados dois conjuntos de atributos A e B; se um **subconjunto de atributos** de A define funcionalmente o conjunto de atributos B, dizemos que B possui **dependência funcional parcial** em relação a A.

- Exemplos:

- Matricula(Mat, CodigoDisciplina, NomeAluno, Média)



- Matr, CodigoDisciplina  $\rightarrow$  Média
    - DF
  - Matr  $\rightarrow$  NomeAluno
    - DF parcial de NomeAluno em relação a {Matr, CodigoDisciplina}

# Dependência funcional - propriedades

## Principais propriedades

### 1. Reflexiva:

Se  $B \subseteq A$  então  $A \rightarrow B$

### 2. Aumentativa:

Se  $A \rightarrow B$  então  $AX \rightarrow B$

E também

Se  $A \rightarrow B$  então  $AX \rightarrow BX$

### 3. Transitiva:

Se  $A \rightarrow B, B \rightarrow C$  então  $A \rightarrow C$

### 4. Decomposição :

Se  $A \rightarrow BC$  então  $A \rightarrow B, A \rightarrow C$

### 5. Aditiva:

Se  $A \rightarrow B, A \rightarrow C$  então  $A \rightarrow BC$

### 6. Pseudo-Transitiva:

Se  $AB \rightarrow D$  e  $C \rightarrow A$  então  $CB \rightarrow D$



# Normalização

- Relação satisfazendo uma determinada propriedade de normalização → diz-se que ela está em uma “**Forma Normal**”
- Serão vistas:
  - **1a.** Forma Normal
  - **2a.** Forma Normal
  - **3a.** Forma Normal
  - Forma Normal de **Boyce Codd**
  - **4a.** Forma Normal

# Dependência Funcional - Observações

- **Não** podem ser inferidas pelo sistema
- Parte da **semântica** do domínio → identificadas pelo projetista
- **Intenções** do projeto
- DFs são a **base** da 2a. Forma Normal, da 3a. Forma Normal e da Forma Normal de Boyce Codd

# 1ª. Forma Normal

# 1ª. Forma Normal

- Simples, mas necessária

- 1a. Forma Normal: todos os atributos são **Monovalorados e Atômicos** (não há relações aninhadas)

# 1ª. Forma Normal - Multivaloração

## Violação por atributo multivalorado

- Exemplo de relação que não está na 1FN

Aluno = (Matr, Nome, Idade, Disciplinas)

| <u>Matr</u> | Nome  | Idade | Disciplinas                     |
|-------------|-------|-------|---------------------------------|
| 221323      | Maria | 20    | FGA0100;<br>FGA0101             |
| 241245      | José  | 21    | FGA0122,<br>FGA0131,<br>FGA0244 |

# 1ª. Forma Normal - Multivaloração

## Violação por atributo multivalorado

- Exemplo de relação que não está na 1FN

Aluno = (Matr, Nome, Idade, Disciplinas)

| <u>Matr</u> | Nome  | Idade | Disciplinas                     |
|-------------|-------|-------|---------------------------------|
| 221323      | Maria | 20    | FGA0100;<br>FGA0101             |
| 241245      | José  | 21    | FGA0122,<br>FGA0131,<br>FGA0244 |

Atributo **multivalorado** ←

# 1ª. Forma Normal - Multivaloração

## Violação por atributo multivalorado

- Exemplo de relação que não está na 1FN

Aluno = (Matr, Nome, Idade, Disciplinas)

| <u>Matr</u> | Nome  | Idade | Disciplinas                     |
|-------------|-------|-------|---------------------------------|
| 221323      | Maria | 20    | FGA0100;<br>FGA0101             |
| 241245      | José  | 21    | FGA0122,<br>FGA0131,<br>FGA0244 |

Atributo **multivalorado** ←

# 1ª. Forma Normal - Multivaloração

## Violação por atributo multivalorado

- Exemplo de relação que não está na 1FN

Aluno = (Matr, Nome, Idade, Disciplinas)

| <u>Matr</u> | Nome  | Idade | Disciplinas                     |
|-------------|-------|-------|---------------------------------|
| 221323      | Maria | 20    | FGA0100;<br>FGA0101             |
| 241245      | José  | 21    | FGA0122,<br>FGA0131,<br>FGA0244 |



Atributo **multivalorado** ←

- Consulta nesta relação:

```
SELECT matr
```

```
FROM aluno
```

```
WHERE disciplinas = '???' → todas as disciplinas, separadas por , ou ;
```



# 1ª. Forma Normal - Multivaloração

- Como normalizar esta relação?

| <u>Matr</u> | Nome  | Idade | Disciplinas         |
|-------------|-------|-------|---------------------|
| 221323      | Maria | 20    | FGA0100,<br>FGA0101 |
| 241245      | José  | 21    | FGA0122,<br>FGA0131 |

Atributo **multivalorado** ←

1) Nova relação: mesma chave + atributo multivalorado, ambos como chave

2) Atributo multivalorado sai da relação

# 1ª. Forma Normal - Multivaloração

- Relação normalizada:  
atributos monovalorados (e atômicos)

| <u>Matr</u> | Idade | Nome  |
|-------------|-------|-------|
| 221323      | 20    | Maria |
| 241245      | 21    | José  |

| <u>Matr</u> | <u>Disciplina</u> |
|-------------|-------------------|
| 221323      | FGA0100           |
| 221323      | FGA0101           |
| 241245      | FGA0122           |
| 241245      | FGA0131           |

# 1ª. Forma Normal - Multivaloração

## Violação por atributo composto

- Exemplo de relação que não está na 1FN

**Aluno(NomeAluno, DeptDisc, Idade)**

| <u>NomeAluno</u> | DeptDisc               | Idade |
|------------------|------------------------|-------|
| Benedita         | Computação Estatística | 20    |
| Mauro            | Matemática Estatística | 21    |

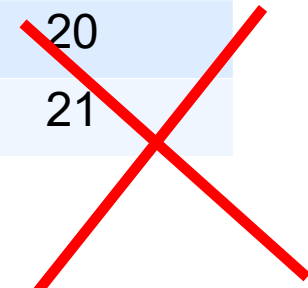
# 1ª. Forma Normal - Multivaloração

## Violação por atributo composto

- Exemplo de relação que não está na 1FN

**Aluno**(NomeAluno, DeptDisc, Idade)

| <u>NomeAluno</u> | DeptDisc               | Idade |
|------------------|------------------------|-------|
| Benedita         | Computação Estatística | 20    |
| Mauro            | Matemática Estatística | 21    |



Atributo **composto**

Consulta nesta relação:

```
SELECT NomeAluno  
FROM Aluno
```

WHERE DeptDisc = '???' → como filtrar por departamento ou por disciplina?

# 1ª. Forma Normal – Atributo Composto

- Como normalizar esta relação?

| <u>NomeAluno</u> | Dept Disc              | Idade |
|------------------|------------------------|-------|
| Benedita         | Computação Estatística | 20    |
| Mauro            | Matemática Estatística | 21    |

Atributo **composto**



Quebrar atributo

# 1ª. Forma Normal – Atributo Composto

- Relação normalizada

atributos atômicos (e monovalorados)

| <u>NomeAluno</u> | Dept       | Disc        | Idade |
|------------------|------------|-------------|-------|
| Benedita         | Computação | Estatística | 20    |
| Mauro            | Matemática | Estatística | 21    |

# 1ª. Forma Normal – Atributo Composto

- Fundamental para a própria conceituação do Modelo Relacional
- Exigida pelos SGBDs Relacionais contemporâneos
- Violação → “Relações aninhadas” (relações dentro de Relações) → violação do modelo

## 2ª. Forma Normal



# 2ª. Forma Normal

- Relação está na 2ª. Forma Normal quando:
  - está na 1ª. FN
  - Atributos comuns não dependem parcialmente de qualquer chave

- Exemplo:

Ministra = (Professor, CodDisc, Livro)

A horizontal line connects the underlined attribute 'Professor' to the attributes 'CodDisc' and 'Livro'. From the end of this line, two vertical arrows point upwards to 'CodDisc' and 'Livro', indicating that 'CodDisc' and 'Livro' are partially dependent on 'Professor'.

Turma = (CodDisc, Numero, Sala, No.Horas)

A horizontal line connects the underlined attribute 'CodDisc' to the attribute 'No.Horas'. From the end of this line, a vertical arrow points upwards to 'No.Horas', indicating that 'No.Horas' is partially dependent on 'CodDisc'.

# 2ª. Forma Normal

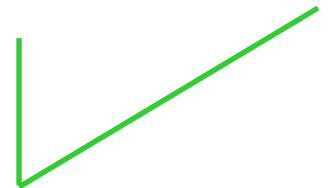
- Relação está na 2ª. Forma Normal quando:
  - está na 1ª. FN
  - Atributos comuns não dependem parcialmente de qualquer chave

- Exemplo:

Ministra = (Professor, CodDisc, Livro)



Turma = (CodDisc, Numero, Sala, No.Horas)




# 2ª. Forma Normal

- Relação está na 2ª. Forma Normal quando:
  - está na 1ª. FN
  - Atributos comuns não dependem parcialmente de qualquer chave

- Exemplo:

Ministra = (Professor, CodDisc, Livro)

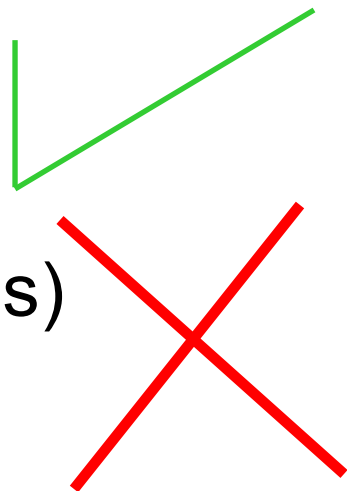


A horizontal line connects the underlined attribute 'Professor' to the attributes 'CodDisc' and 'Livro'. From this line, two vertical lines descend and then turn upwards as arrows pointing to 'CodDisc' and 'Livro', indicating that 'CodDisc' and 'Livro' are partially dependent on 'Professor'.

Turma = (CodDisc, Numero, Sala, No.Horas)



A horizontal line connects the underlined attribute 'CodDisc' to the attribute 'No.Horas'. A vertical line descends from this line and turns upwards as an arrow pointing to 'No.Horas', indicating that 'No.Horas' is partially dependent on 'CodDisc'.



# 2ª. Forma Normal - Violação

- Exemplo:

Turma = (CodDisc, Numero, Sala, No.Horas)

Suponha que cada disciplina tem sua quantidade de horas bem definida como ocorre no mundo real.

Assim, o modelo não deve permitir que duas disciplinas sejam armazenadas com número de horas diferentes.

## 2ª. Forma Normal - Violação

- Exemplo:

Turma = (CodDisc, Numero, Sala, No.Horas)

| <u>CodDisc</u> | <u>Numero</u> | Sala | No.Horas |
|----------------|---------------|------|----------|
| SMA            | 1             | 24   | 60       |
| PA             | 1             | 13   | 30       |
| PA             | 2             | 25   | 30       |
| SMA            | 2             | 31   | 50       |

# 2ª. Forma Normal - Violação

- Exemplo:

Turma = (CodDisc, Numero, Sala, No.Horas)

| <u>CodDisc</u> | <u>Numero</u> | Sala | No.Horas |
|----------------|---------------|------|----------|
| SMA            | 1             | 24   | 60       |
| PA             | 1             | 13   | 30       |
| PA             | 2             | 25   | 30       |
| SMA            | 2             | 31   | 50       |

Dependência  
funcional  
parcial  
à chave



## 2ª. Forma Normal - Anomalias

- Exemplo:

Turma = (CodDisc, Numero, Sala, No.Horas)

| <u>CodDisc</u> | <u>Numero</u> | Sala | No.Horas |
|----------------|---------------|------|----------|
| SMA            | 1             | 24   | 60       |
| PA             | 1             | 13   | 30       |
| PA             | 2             | 25   | 30       |
| SMA            | 2             | 31   | 50       |

- Se CodDisc = PA → No. Horas = 30
- Se CodDisc = SMA → No. Horas = 60 ou 50?

# 2ª. Forma Normal - Anomalias

- Exemplo:

Turma = (CodDisc, Numero, Sala, No.Horas)

| <u>CodDisc</u> | <u>Numero</u> | Sala | No.Horas |
|----------------|---------------|------|----------|
| SMA            | 1             | 24   | 60       |
| PA             | 1             | 13   | 30       |
| PA             | 2             | 25   | 30       |
| SMA            | 2             | 31   | 50       |



- Se CodDisc = PA → No. Horas = 30
- Se CodDisc = SMA → No. Horas = 60 ou 50?

Redundância e  
Inconsistência  
de inserção



## 2ª. Forma Normal - Normalização

- Como normalizar esta relação?

Turma = (CodDisc, Numero, Sala, No.Horas)

1) Nova relação:

parte da chave que define a dependência  
(chave da nova relação)

+

atributos dependentes desta parte da chave

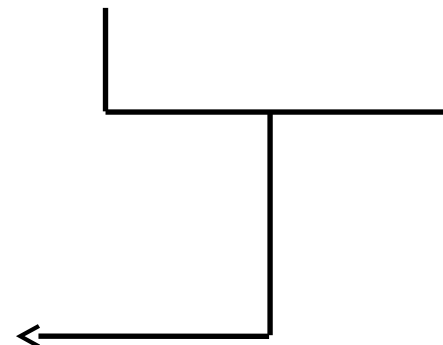
2) Atributos parcialmente dependentes  
da chave saem da relação

# 2ª. Forma Normal - Normalização

- Relação normalizada  
sem dependências funcionais parciais à chave

| <u>CodDisc</u> | <u>Numero</u> | Sala |
|----------------|---------------|------|
| SMA            | 1             | 24   |
| PA             | 1             | 13   |
| PA             | 2             | 25   |
| SMA            | 2             | 31   |

| <u>CodDisc</u> | No.Horas |
|----------------|----------|
| SMA            | 60       |
| PA             | 30       |



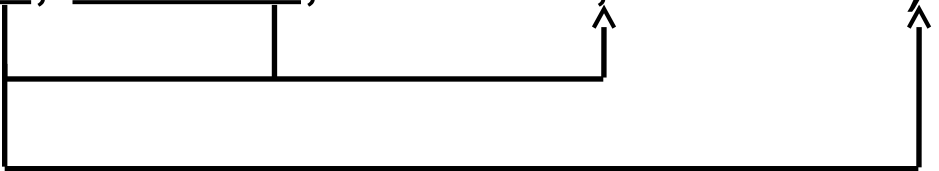
A própria estrutura do esquema:

- garante a não redundância
- garante a consistência (apenas uma versão dos dados)

# 2ª. Forma Normal - Violação

- Exemplo:

ProjetoFuncao(Projld, Funcld, Funcao, Gerente)



| <u>Projld</u> | <u>Funcld</u> | Funcao      | Gerente |
|---------------|---------------|-------------|---------|
| P23           | F101          | Eletricista | Felipe  |
| P14           | F101          | Encanador   | João    |
| P25           | F453          | Porteiro    | Márcio  |
| P14           | F453          | Segurança   | João    |

- Não haverá mais a função de eletricista

DELETE FROM ProjetoFuncao

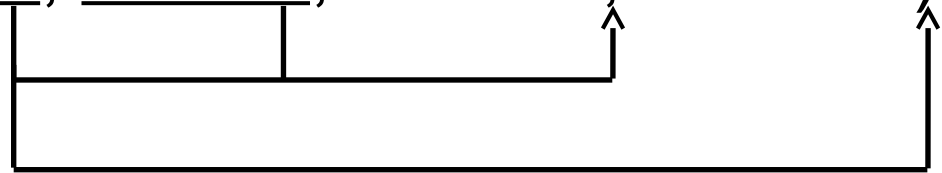
WHERE Funcao = 'Eletricista'

- Qual é o problema decorrente desta operação?
- Por que não está normalizado? Como normalizar?

# 2ª. Forma Normal - Violação

- Exemplo:

ProjetoFuncao(Projld, Funcld, Funcao, Gerente)



| <u>Projld</u> | <u>Funcld</u> | Funcao      | Gerente |
|---------------|---------------|-------------|---------|
| P23           | F101          | Eletricista | Felipe  |
| P14           | F101          | Encanador   | João    |
| P25           | F453          | Porteiro    | Márcio  |
| P14           | F453          | Segurança   | João    |

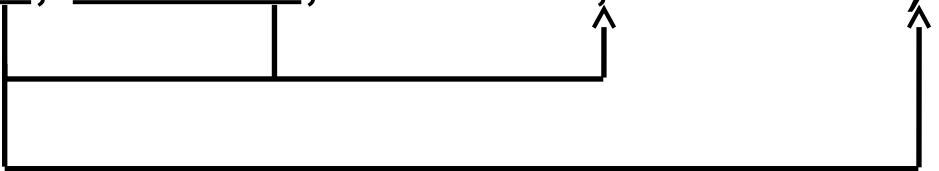
Dependência  
funcional  
parcial  
à chave



## 2ª. Forma Normal - Violação

- Exemplo:

ProjetoFuncao(Projld, Funcld, Funcao, Gerente)



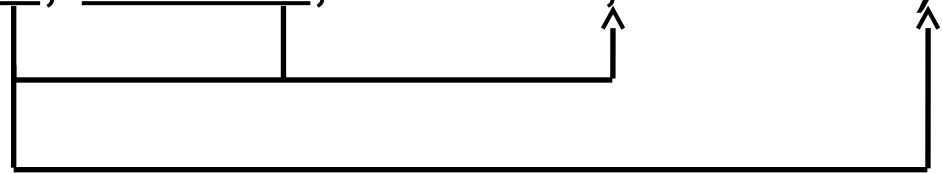
| <u>Projld</u> | <u>Funcld</u> | Funcao      | Gerente |
|---------------|---------------|-------------|---------|
| P23           | F101          | Eletricista | Felipe  |
| P14           | F101          | Encanador   | João    |
| P25           | F453          | Porteiro    | Márcio  |
| P14           | F453          | Segurança   | João    |

- Não haverá mais a função de eletricista  
DELETE FROM ProjetoFuncao  
WHERE Funcao = 'Eletricista'

# 2ª. Forma Normal - Violação

- Exemplo:

ProjetoFuncao(Projld, Funcld, Funcao, Gerente)



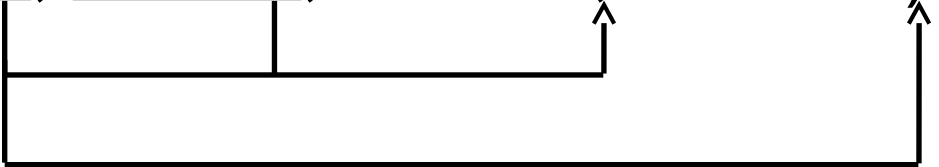
| <u>Projld</u> | <u>Funcld</u> | Funcao      | Gerente |
|---------------|---------------|-------------|---------|
| P23           | F101          | Eletricista | Felipe  |
| P14           | F101          | Encanador   | João    |
| P25           | F453          | Porteiro    | Márcio  |
| P14           | F453          | Segurança   | João    |

- Não haverá mais a função de eletricista  
DELETE FROM ProjetoFuncao  
WHERE Funcao = 'Eletricista'

# 2ª. Forma Normal - Violação

- Exemplo:

ProjetoFuncao(Projld, Funcld, Funcao, Gerente)



| <u>Projld</u> | <u>Funcld</u> | Funcao    | Gerente |
|---------------|---------------|-----------|---------|
| P14           | F101          | Encanador | João    |
| P25           | F453          | Porteiro  | Márcio  |
| P14           | F453          | Segurança | João    |

- Problema no projeto P23, quem é o gerente que vai responder por isso?

SELECT gerente

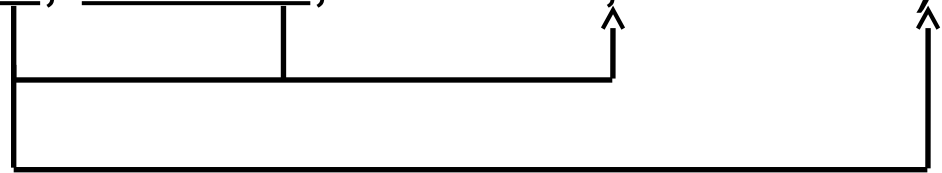
FROM ProjetoFuncao

WHERE projld = 'P23'

# 2ª. Forma Normal - Violação

- Exemplo:

ProjetoFuncao(Projld, Funcld, Funcao, Gerente)



| <u>Projld</u> | <u>Funcld</u> | Funcao    | Gerente |
|---------------|---------------|-----------|---------|
| P14           | F101          | Encanador | João    |
| P25           | F453          | Porteiro  | Márcio  |
| P14           | F453          | Segurança | João    |

- Problema no projeto P23, quem é o gerente que vai responder por isso?

Inconsistência  
de remoção

← NULL



# 2ª. Forma Normal - Normalização

- Como normalizar esta relação?

ProjetoFuncao(Projld, Funcld, Funcao, Gerente)

1) Nova relação:

parte da chave que define a dependência  
(chave da nova relação)

+

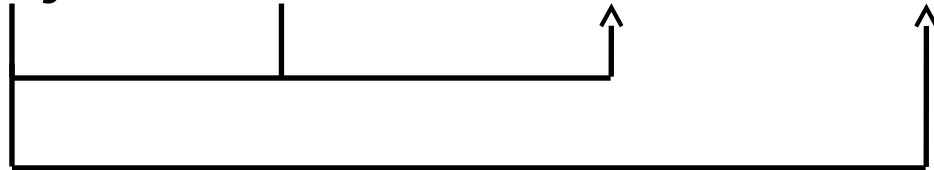
atributos dependentes desta parte da chave

2) Atributos parcialmente dependentes  
da chave saem da relação

## 2ª. Forma Normal - Normalização

- Como normalizar esta relação?

ProjetoFuncao(Projld, Funcld, Funcao, Gerente)



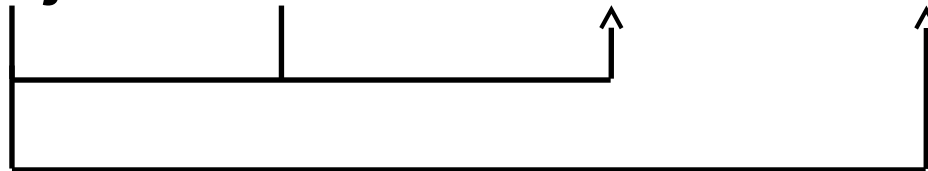
| <u>Projld</u> | <u>Funcld</u> | Funcao      |
|---------------|---------------|-------------|
| P23           | F101          | Eletricista |
| P14           | F101          | Encanador   |
| P25           | F453          | Porteiro    |
| P14           | F453          | Segurança   |

| <u>Projld</u> | Gerente |
|---------------|---------|
| P23           | Felipe  |
| P14           | João    |
| P25           | Márcio  |

# 2ª. Forma Normal - Normalização

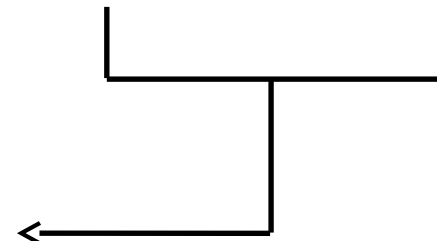
- Como normalizar esta relação?

ProjetoFuncao(Projld, Funcld, Funcao, Gerente)



| <u>Projld</u> | <u>Funcld</u> | Funcao      |
|---------------|---------------|-------------|
| P23           | F101          | Eletricista |
| P14           | F101          | Encanador   |
| P25           | F453          | Porteiro    |
| P14           | F453          | Segurança   |

| <u>Projld</u> | Gerente |
|---------------|---------|
| P23           | Felipe  |
| P14           | João    |
| P25           | Márcio  |



A própria estrutura do esquema:

- garante a não redundância
- garante a consistência de remoção
- aumenta a confiabilidade dos dados como um todo


## 3ª. Forma Normal

# 3ª. Forma Normal

- Uma relação está na 3ª. Forma Normal quando:
  - está na 2ª. Forma Normal
  - atributos comuns não dependem transitivamente de qualquer superchave

- Exemplo:

Campeoes(CompeticaoId, Ano, Vencedor, DataNascVenc)



# 3ª. Forma Normal

- Uma relação está na 3ª. Forma Normal quando:
  - está na 2ª. Forma Normal
  - atributos comuns não dependem transitivamente de qualquer superchave

- Exemplo:

Campeoes(CompeticaoId, Ano, Vencedor, DataNascVenc)

- Como:

(CompeticaoId, Ano → Vencedor) E (Vencedor → DataNascVenc)


Então, por transitividade:

**CompeticaoId, Ano → DataNascVenc**

# 3ª. Forma Normal - Violação

- Exemplo:

Campeoes(Competicao, Ano, Vencedor, DataNascVenc)



| <u>Competicao</u> | <u>Ano</u> | Vencedor | DataNascVenc |
|-------------------|------------|----------|--------------|
| C21               | 2001       | Miguel   | 04/04/1975   |
| C34               | 2002       | César    | 09/12/1980   |
| C21               | 2003       | Miguel   | 04/04/1975   |
| C57               | 2004       | Fabiano  | 06/02/1978   |

# 3ª. Forma Normal - Violação

- Exemplo:

Campeoes(Competicao, Ano, Vencedor, DataNascVenc)



| <u>Competicaold</u> | <u>Ano</u> | Vencedor | DataNascVenc |
|---------------------|------------|----------|--------------|
| C21                 | 2001       | Miguel   | 04/04/1975   |
| C34                 | 2002       | César    | 09/12/1980   |
| C21                 | 2003       | Miguel   | 04/04/1975   |
| C57                 | 2004       | Fabiano  | 06/02/1978   |

Atributo não-primo DataNasc depende transitivamente da chave {Competicaold, Ano}



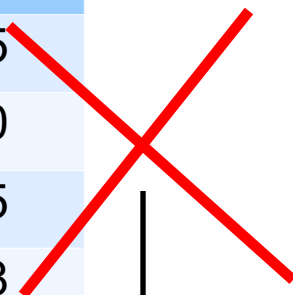
# 3ª. Forma Normal - Violação

- Exemplo:

Campeoes(Competicao, Ano, Vencedor, DataNascVenc)



| <u>Competicao</u> | <u>Ano</u> | Vencedor | DataNascVenc |
|-------------------|------------|----------|--------------|
| C21               | 2001       | Miguel   | 04/04/1975   |
| C34               | 2002       | César    | 09/12/1980   |
| C21               | 2003       | Miguel   | 04/04/1975   |
| C57               | 2004       | Fabiano  | 06/02/1978   |



**Redundância**

**Se Vencedor = Miguel → DataNascVenc = 04/04/1975**



# 3ª. Forma Normal - Violação

- Exemplo:

Campeoes(Competicao, Ano, Vencedor, DataNascVenc)

The diagram shows a horizontal line with four vertical tick marks corresponding to the attributes: Competicao, Ano, Vencedor, and DataNascVenc. A bracket connects the first two tick marks (Competicao and Ano) to the third tick mark (Vencedor). Another bracket connects the first two tick marks (Competicao and Ano) to the fourth tick mark (DataNascVenc). This indicates that the combination of Competicao and Ano determines both Vencedor and DataNascVenc.

| <u>Competicaold</u> | <u>Ano</u> | Vencedor | DataNascVenc |
|---------------------|------------|----------|--------------|
| C21                 | 2001       | Miguel   | 04/04/1975   |
| C34                 | 2002       | César    | 09/12/1980   |
| C21                 | 2003       | Miguel   | 04/04/1975   |
| C57                 | 2004       | Fabiano  | 06/02/1978   |

- O vencedor da competição C21 de 2001 forneceu data de nascimento errada – precisamos atualizar os dados

UPDATE Campeoes

SET DataNascVenc='14/04/1975'

WHERE Competicaold = C21 AND ANO=2001

# 3ª. Forma Normal - Violação

- Exemplo:

Campeoes(Competicao, Ano, Vencedor, DataNascVenc)



| <u>Competicaold</u> | <u>Ano</u> | Vencedor | DataNascVenc |
|---------------------|------------|----------|--------------|
| C21                 | 2001       | Miguel   | 14/04/1975   |
| C34                 | 2002       | César    | 09/12/1980   |
| C21                 | 2003       | Miguel   | 04/04/1975   |
| C57                 | 2004       | Fabiano  | 06/02/1978   |

- O vencedor da competição C21 de 2001 forneceu data de nascimento errada – precisamos atualizar os dados

UPDATE Campeoes

SET DataNascVenc='14/04/1975'

WHERE Competicaold = C21 AND ANO=2001

# 3ª. Forma Normal - Violação

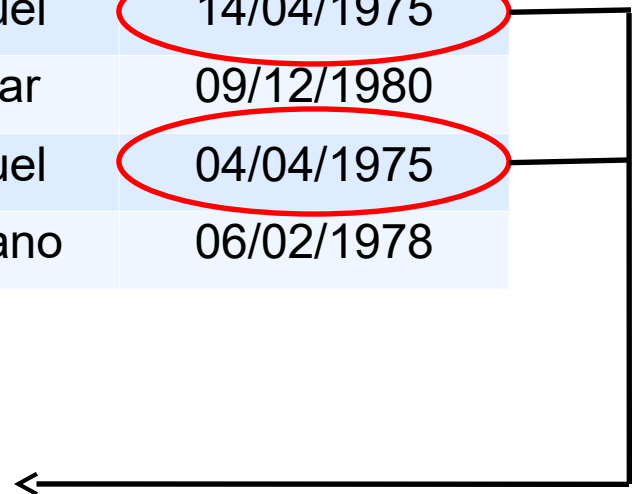
- Exemplo:

Campeoes(Competicao, Ano, Vencedor, DataNascVenc)



| <u>Competicao</u> | <u>Ano</u> | Vencedor | DataNascVenc |
|-------------------|------------|----------|--------------|
| C21               | 2001       | Miguel   | 14/04/1975   |
| C34               | 2002       | César    | 09/12/1980   |
| C21               | 2003       | Miguel   | 04/04/1975   |
| C57               | 2004       | Fabiano  | 06/02/1978   |


Inconsistência de  
atualização



# 3ª. Forma Normal - Normalização

- Como normalizar esta relação?

Campeoes(Competicao, Ano, Vencedor, DataNascVenc)



## 1) Nova relação:

atributos dependentes transitivamente

+

atributos dos quais eles dependem  
diretamente

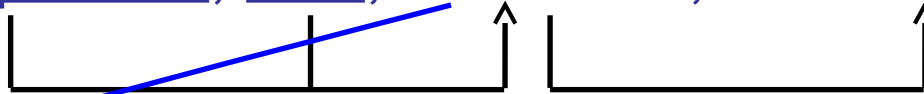
(chave da nova relação)

## 2) Atributos dependentes transitivamente saem da relação

# 3ª. Forma Normal - Normalização

- Como normalizar esta relação?

Campeoes(Competicao, Ano, Vencedor, DataNascVenc)



1) Nova relação:

atributos dependentes transitivamente

+

atributos dos quais eles dependem  
diretamente

(chave da nova relação)

2) Atributos dependentes  
transitivamente saem da relação

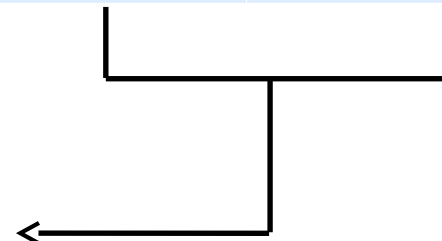
# 3ª. Forma Normal - Normalização

- Relação normalizada

sem dependências funcionais transitivas  
a qualquer chave

| <u>Competicao</u> | <u>Ano</u> | <u>Vencedor</u> |
|-------------------|------------|-----------------|
| C21               | 2001       | Miguel          |
| C34               | 2002       | César           |
| C21               | 2003       | Miguel          |
| C57               | 2004       | Fabiano         |

| <u>Vencedor</u> | <u>DataNascVenc</u> |
|-----------------|---------------------|
| Miguel          | 14/04/1975          |
| César           | 09/12/1980          |
| Fabiano         | 06/02/1978          |



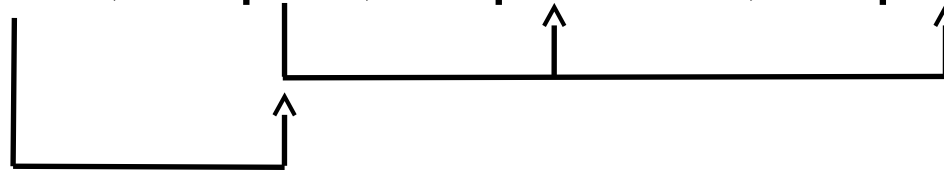
A própria estrutura do esquema:

- garante a não redundância
- garante a consistência de atualização
- reduz os custos de manutenção (consolidação tem custo)

# 3ª. Forma Normal - Violação

- Exemplo:

Disciplina(DiscId, DeptId, DeptNome, DeptChefe)



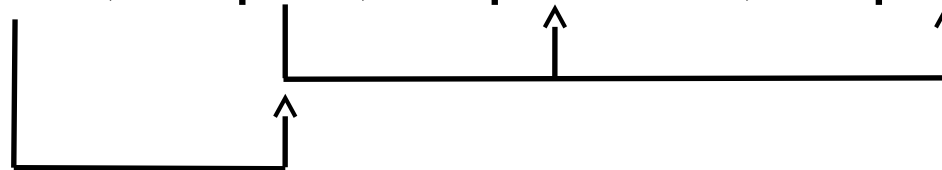
| <u>DiscId</u> | DeptId | DeptNome   | DeptChefe |
|---------------|--------|------------|-----------|
| FGA0102       | D4     | Computação | Marcos    |
| FGA0110       | D4     | Computação | Marcos    |
| FGA0210       | D5     | Matemática | Vilma     |
| FGA0215       | D5     | Matemática | Sandra    |



# 3ª. Forma Normal - Violação

- Exemplo:

Disciplina(DiscId, DeptId, DeptNome, DeptChefe)



| <u>DiscId</u> | DeptId | DeptNome   | DeptChefe |
|---------------|--------|------------|-----------|
| FGA0102       | D4     | Computação | Marcos    |
| FGA0110       | D4     | Computação | Marcos    |
| FGA0210       | D5     | Matemática | Vilma     |
| FGA0215       | D5     | Matemática | Sandra    |

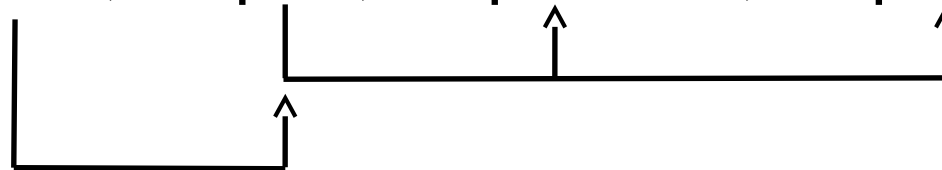
Redundância e inconsistência de inserção dos dados

Como normalizar esta relação?

# 3ª. Forma Normal - Violação

- Exemplo:

Disciplina(DiscId, DeptId, DeptNome, DeptChefe)



| <u>DiscId</u> | DeptId | DeptNome   | DeptChefe |
|---------------|--------|------------|-----------|
| FGA0102       | D4     | Computação | Marcos    |
| FGA0110       | D4     | Computação | Marcos    |
| FGA0210       | D5     | Matemática | Vilma     |
| FGA0215       | D5     | Matemática | Sandra    |

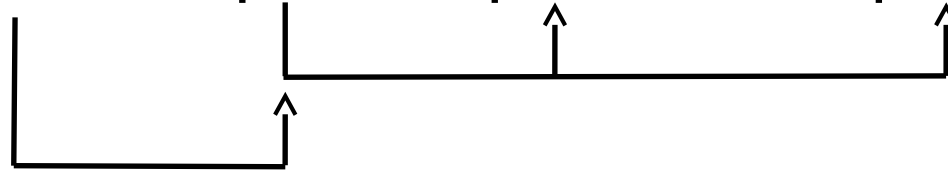
**Atributos não-primos DeptNome e DeptChefe dependem transitivamente da chave (DiscId)**



# 3ª. Forma Normal - Normalização

- Exemplo:

Disciplina(Discld, Deptld, DeptNome, DeptChefe)



| <u>Discld</u> | Deptld |
|---------------|--------|
| FGA0102       | D4     |
| FGA0110       | D4     |
| FGA0210       | D5     |
| FGA0215       | D5     |

| <u>Deptld</u> | DeptNome   | DeptChefe |
|---------------|------------|-----------|
| D4            | Computação | Marcos    |
| D5            | Matemática | Sandra    |

# Observações

- As práticas de modelagem vistas durante o curso não geram esquemas desnormalizados
- A teoria de normalização, assim tem dois propósitos:
  - Normalizar projetos já existentes, mas que apresentam problemas de redundância e inconsistência
  - Desnormalizar projetos que possuem requisitos específicos de recuperação de dados → garantir a normalização usando outros recursos, como triggers e interfaces de inserção dos dados