

**Banco de Dados**

**Normalização**

# Introdução

---

- Com o surgimento e o aperfeiçoamento do sistema relacional na década de 70, várias regras foram definidas para a simplificação de tabelas, recebendo o nome de formas normais.
- Cada uma destas regras apresenta um critério de adequação, e o processo de adequação de tabelas a estas regras práticas chama-se normalização.

# Normalização de Dados

---

É um **processo formal**, passo a passo, de **análise** dos **atributos** de uma relação

Objetivo:

- evitar redundância,
  - inconsistência e
  - perda de informação no banco de dados
- 
- Teoria proposta por Codd no início dos anos 70.

# Anomalias de Atualização

Inclusão

Exclusão

Modificação

*Ex: Quais problemas são decorrentes da relação Vendas?*

nomeC	<u>CPF</u>	endereco	fone	<u>codPr</u>	nomePr	Vunit	qtd	total
Zé	111	ABC	9923123	A	Lápis	0,50	2	1,00
Ana	222	XYZ	9111456	B	Caneta	1,00	3	3,00
Zé	111	ABC	9912123	C	Régua	1,00	2	2,00
Pedro	444	KZZ	Null	A	Lápis	0,50	20	10,00

# Anomalias de Atualização

- Modificação/Atualização:
  - uma mudança na descrição da peça A requer várias mudanças
- inconsistência:
  - não há nada no projeto impedindo que o produto A tenha duas ou mais descrições diferentes no BD

nomeC	<u>CPF</u>	endereco	fone	<u>codPr</u>	nomePr	Vunit	qtd	total
Zé	111	ABC	9923123	A	Lápis	0,50	2	1,00
Ana	222	XYZ	9111456	B	Caneta	1,00	3	3,00
Zé	111	ABC	9912123	C	Régua	1,00	2	2,00
Pedro	444	KZZ	Null	A	Lápis	0,50	20	10,00

# Anomalias de Atualização

- Inclusão:
  - redundância do cliente e do produto
- Exclusão:
  - se o cliente ANA fosse eliminado seria perdida a informação de que o produto B é chamado *caneta* e *custa R\$ 1,00*

nomeC	<u>CPF</u>	endereco	fone	<u>codPr</u>	nomePr	Vunit	qtd	total
Zé	111	ABC	9923123	A	Lápis	0,50	2	1,00
Ana	222	XYZ	9111456	B	Caneta	1,00	3	3,00
Zé	111	ABC	9912123	C	Régua	1,00	2	2,00
Pedro	444	KZZ	Null	A	Lápis	0,50	20	10,00

---

# Formas de Normalização

# Normalização de Dados

---

**Como evitar os problemas na criação de um novo BD?**

**Elaborando um bom modelo conceitual de dados.**

**Aplicando corretamente o projeto lógico de BDs.**

Processo de **Normalização de Dados.**



# Processo de Normalização

---

- inicia com uma relação ou coleção de relações
- produz uma nova coleção de relações:
  - equivalente a coleção original (representa a mesma informação)
  - livre de problemas
- Propriedades a serem alcançadas:
  - Minimização da redundância
  - Minimização de anomalias de inserção, exclusão e atualização de dados
    - Ex.: *Florianópolis, Florianópolis, Floripa, Flópolis*
- **Significado: as novas relações estarão, pelo menos na 3FN**

# Processo de Normalização

Primeira  
Forma  
Normal  
1FN

Segunda  
Forma  
Normal  
2FN

Terceira  
Forma  
Normal  
3FN

Boyce-  
Codd,  
4FN,  
5FN

DIMINUI A REDUNDÂNCIA DE DADOS

AUMENTA O DESEMPENHO DAS CONSULTAS

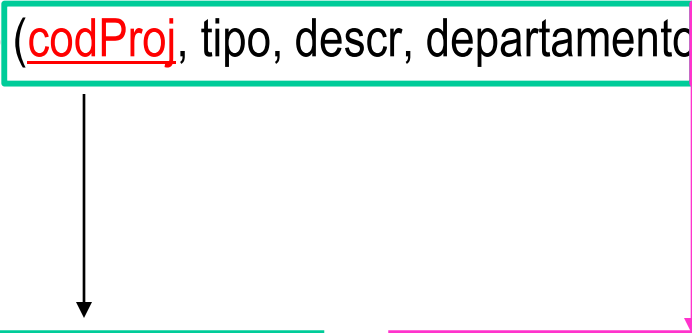
# Processo de Normalização

- Tipos de Formas Normais

Domínio multivalorado; Atributo determinante ou chave	<b>Primeira forma normal (1FN)</b>
Determinante; Dependência funcional	<b>Segunda forma normal (2FN)</b>
Dependência funcional transitiva	<b>Terceira forma normal (3FN)</b>
Chave candidata	Forma Normal de Boyce-Codd (FNBC)
Fato multivalor	Quarta forma normal (4FN)

# Exemplo de Tabela Não-normalizada

ProjetoEmpregado (codProj, tipo, descr, departamento, codEmp, nome, categ, sal, dataIni, tempoAI)



CódProj	Tipo	Descr	Emp					
			CodEmp	Nome	Cat	Sal	DataIni	TempAI
LSC001	Novo Desenv.	Sistema de Estoque	2146	João	A1	4	1/11/91	24
			3145	Sílvio	A2	4	2/10/91	24
			6126	José	B1	9	3/10/92	18
			1214	Carlos	A2	4	4/10/92	18
			8191	Mário	A1	4	1/11/92	12
PAG02	Manutenção	Sistema RH	8191	Mário	A1	4	1/05/93	12
			4112	João	A2	4	4/01/91	24
			6126	José	B1	9	1/11/92	12

**Para cada empregado os dados do projeto se repetem**

# 1ª Forma Normal (1FN)

- O domínio de um atributo só deve permitir atributos atômicos (simples, indivisíveis)
  - Não permite **aninhamento** de tabelas
  - Não permite **atributos multivalorados**

*Departamento*

<u>Numero</u>	Nome	Gerente	Localizacao
1DEP	Pesquisa	111222	{Floripa, Joinville, Blumenau}
2DEP	RH	222333	Joinville
3DEP	Adm	333444	{Floripa, Porto Alegre}
4DEP	Diretoria	444555	Floripa

Atributo multivalorado

- Procedimento usual
  - gerar uma tabela para cada aninhamento

# 1ª Forma Normal (1FN)

---

- Duas soluções:
  - 1º.: criar uma nova tabela
  - 2º.: criar mais atributos na tabela
- Com **tabela aninhada**:
  - Solução 1
- Com **atributo multivalorado**
  - Soluções 1 e 2

# 1ª Forma Normal (1FN)

ÑN:

ProjetoEmpregado (codProj, tipo, descr, departamentoProjeto codEmp, nome, categ, sal, dataIni, tempoAl)



Tabela aninhada

1FN:

Departamento (codDepto, nome)

ProjetoEmpregado (codProj, tipo, descr, #codDepto, codEmp, nome, categ, sal, dataIni, tempoAl)

# Dependência Funcional

---

Dada uma relação  $\underline{R}$  com atributos  
 $A_1, \dots, A_n, B_1, \dots, B_m, C_1, \dots, C_l,$

Dizemos que

$A_1, \dots, A_n$  determina funcionalmente  $B_1, \dots, B_m$   
 $(A_1, \dots, A_n) \longrightarrow B_1, \dots, B_m$

Sempre que duas tuplas tiverem os mesmos valores para  $A_1, \dots, A_n$ , então elas terão o mesmo valor para  $B_1, \dots, B_m$ .

Produto  $\rightarrow$  Descrição



# Dependência Funcional: Exemplo

...	Código	...	Salário
	E1		500
	E3		450
	E2		<b>600</b>
	E1		500
	E3		450
	E2		<b>600</b>
	...		...

Código → Salário

# DF Total e DF Parcial


---

- **DF Parcial**
  - um atributo depende funcionalmente de **parte** da chave composta de uma tabela OU
  - **Parte da chave composta** identifica um ou mais atributos da tabela
- **DF Total**
  - um atributo depende funcionalmente de **todos** os atributos da chave composta de uma tabela
  - **A chave composta completa** identifica um ou mais atributos da tabela

# DF Total e DF Parcial

- DF Parcial

ProjetoEmpregado (codProj, tipo, descr, #codDepto, codEmp, nome, categ, sal, dataIni, tempoAl)



codEmp (parte da chave) – identifica o empregado, a categoria e seu salario  
codProj (parte da chave) – identifica o projeto, o tipo, a descr e o depto

- DF Total

ProjetoEmpregado (codProj, tipo, descr, #codDepto, codEmp, nome, categ, sal, dataIni, tempoAl)



codProj e codEmp (chave completa)– identificam a data de inicio e o tempo no qual o empregado atua no projeto

# 2ª Forma Normal (2FN)

---

- “Uma tabela está na 2FN se e somente se ela estiver na 1FN e não possuir *Dependência Funcional Parcial (DF)*
  - tabelas com **DFs parciais** devem ser desmembradas em tabelas com **DFs totais**
- Tabelas cuja PK possui apenas um atributo estão automaticamente na 2FN

# 2ª Forma Normal (2FN)

---

1FN:

Departamento (codDepto, nome)

ProjetoEmpregado (codProj, tipo, descr, #codDepto, codEmp, nome, categ, sal, dataIni, tempoAl)

2FN:

- Departamento (codDepto, nome)
- Projeto (codProj, tipo, descr, #codDepto )
- ProjetoEmpregado (#codProj, #codEmp, dataIni, tempoAl)
- Empregado (codEmp, nome, categ, sal)

# Dependência Funcional Transitiva

---

*Se um atributo não-chave possui DF total de um atributo chave e também possui DF total de um ou mais atributos não-chave, então diz-se que existe uma DF transitiva ou indireta da CP de T*

Empregado (codEmp, nome, categ, sal)



The diagram shows a horizontal line connecting the underlined attribute 'codEmp' to the attribute 'sal'. A vertical line descends from 'codEmp', and another vertical line ascends from 'sal', connected by a horizontal line segment. An arrow points from the right side of this horizontal segment up towards 'sal', indicating a transitive functional dependency.

# 3<sup>a</sup> Forma Normal (3FN)

---

*“Uma tabela está na 3FN se e somente se ela estiver na 2FN e não possuir DFs indiretas”*

- tabelas com DFs indiretas devem ser desmembradas em tabelas que não possuem tais Dfs

Tabelas que possuem apenas um atributo que não faz parte da PK estão automaticamente na 3FN

# 3ª Forma Normal (3FN)

---

1FN:

Departamento (codDepto, nome)

ProjetoEmpregado (codProj, tipo, descr, #codDepto, codEmp, nome, categ, sal, dataIni, tempoAI)

2FN:

- Departamento (codDepto, nome)
- Projeto (codProj, tipo, descr, #codDepto )
- ProjetoEmpregado (#codProj, #codEmp, dataIni, tempoAI)
- Empregado (codEmp, nome, categ, sal)

3FN

- Departamento (codDepto, nome)
- Projeto (codProj, tipo, descr, #codDepto )
- ProjetoEmpregado (#codProj, #codEmp, dataIni, tempoAI)
- Empregado (codEmp, nome, #categ)
- CategoriaEmpregado (Categ, Sal)



# Questões

- Análise de chaves primárias (Pks)
  - *tabelas podem ou não ter atributos que garantam a identificação única de suas tuplas ou ter uma CP muito extensa*

sugestão: definir uma CP

ÑN: Projeto (CodProj, Tipo, Descr, (Nome, Cat, Sal, DataIni, TempoAl))



ÑN: Projeto (CodProj, Tipo, Descr, (CodEmp, Nome, Cat, Sal, DataIni, TempoAl))

# Questões

- Dados irrelevantes
  - tabelas podem ter atributos que não precisam ser mantidos necessariamente no BD

sugestão: eliminar estes atributos

ÑN: Projetos (CodProj, Tipo, Descr, NroEmps,  
(CodEmp, Nome, Cat, Sal, DataIni, TempoAloc))



ÑN: Projetos (CodProj, Tipo, Descr, (CodEmp, Nome,  
Cat, Sal, DataIni, TempoAloc))

# Questões

- Dados relevantes, porém implícitos  
sugestão: definir tais dados

ÑN: Aprovação (CodCurso, Nome, (CodCand, Nome, Endereço))

a ordem  
determina a  
classificação  
do candidato

ÑN: Aprovação (CodCurso, Nome, (CodCand, Nome, Endereço, OrdemClass))

# Exemplo

---

ÑN - Matricula (cod\_aluno, cod\_turma, cod\_disciplina,  
nome\_disciplina, nome\_aluno, cod\_local\_nasc, nome\_local\_nasc)

- Dependências ?

# Exemplo

---

ÑN - Matricula (cod\_aluno, cod\_turma, cod\_disciplina,  
nome\_disciplina, nome\_aluno, cod\_local\_nasc, nome\_local\_nasc)

- Dependências:
  - (cod\_aluno, cod\_turma) -> cod\_disciplina
  - cod\_aluno -> nome\_aluno, cod\_local\_nasc, nome\_local\_nasc
  - cod\_disciplina -> nome\_disciplina
  - cod\_local\_nasc -> nome\_local\_nasc

# Exemplo

---

Matricula (cod\_aluno, cod\_turma, cod\_disciplina, nome\_disciplina,  
nome\_aluno, cod\_local\_nasc, nome\_local\_nasc)

**2FN**

Matricula (cod\_aluno, cod\_turma, cod\_disciplina, nome\_disciplina)

Aluno(cod\_aluno, nome\_aluno, cod\_local\_nasc, nome\_local\_nasc)

# Exemplo

---

Matricula (cod\_aluno, cod\_turma, cod\_disciplina, nome\_disciplina,  
nome\_aluno, cod\_local\_nasc, nome\_local\_nasc)

## 2FN

Matricula (cod\_aluno, cod\_turma, cod\_disciplina, nome\_disciplina)  
Aluno(cod\_aluno, nome\_aluno, cod\_local\_nasc, nome\_local\_nasc)

## 3FN

Matricula (#cod\_aluno, cod\_turma, #cod\_disciplina)  
Disciplina (cod\_disciplina, nome\_disciplina)  
Aluno (cod\_aluno, nome\_aluno, #cod\_local\_nasc)  
Local(cod\_local\_nasc, nome\_local\_nasc)

# Exercício

---

- Transformar os sistemas acadêmico para a terceira forma normal