

# Bases de Dados

## T12 - Normalização Parte I

Prof. Daniel Faria

Prof. Flávio Martins

# Sumário

- Recapitulação
- Motivação
- Dependências Funcionais
- Superchaves e Chaves Candidatas
- Formas Normais

# Recapitulação

# Álgebra Relacional: Conceptualização de Interrogações

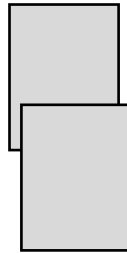
**Select:  $\sigma$**

A	B	C	D

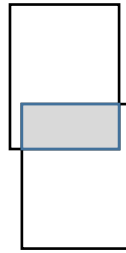
**Project:  $\Pi$**

A	B	C	D

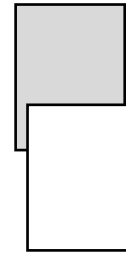
**Union:  $\cup$**



**Intersection:  $\cap$**



**Difference:  $-$**



**Cartesian Product:  $\times$**

a1	b1
a2	b2
a3	b3

b1	c1
b3	c2



a1	b1	b1	c1
a1	b1	b3	c2
a2	b2	b1	c1
a2	b2	b3	c2
a3	b3	b1	c1
a3	b3	b3	c2

**(Natural) Join:  $\bowtie$**

a1	b1
a2	b2
a3	b3

b1	c1
b3	c2



a1	b1	c1
a3	b3	c2

**Outer Join:  $\bowtie, \ltimes, \ltimes$**

a1	b1
a2	b2
a3	b3

b1	c1
b3	c2



a1	b1	c1
a3	b3	c2
a2	b2	NULL

**Division:  $\div$**

a1	b1
a1	b2
a2	b1
a3	b2


b1
b2



a1
----

**Aggregation:  $G$  or  $\gamma$**

a1
a2
a3
a4



val
-----

# Operações de Álgebra Relacional em SQL

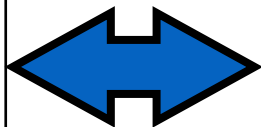
```
[WITH with_query [, ...]]  
SELECT [ALL | DISTINCT [ON (expression [, ...])]]  
      [* | expression [[AS] output_name] [, ...]] ← Project, Rename, Aggregation  
      [FROM from_item [, ...]] ← Cartesian Product, Joins  
      [WHERE condition] ← Select  
      [GROUP BY [ALL | DISTINCT] grouping_element [, ...]] ← Aggregation w/ Grouping  
      [HAVING condition] ← Select (after Aggregation)  
      [{UNION | INTERSECT | EXCEPT} [ALL | DISTINCT] select] ← Union, Intersection, Difference  
      [ORDER BY expression [ASC | DESC | USING operator]  
        [NULLS { FIRST | LAST}] [, ...]]  
      [LIMIT {count | ALL}]
```

# Motivação

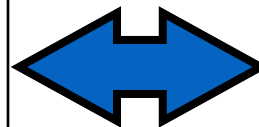
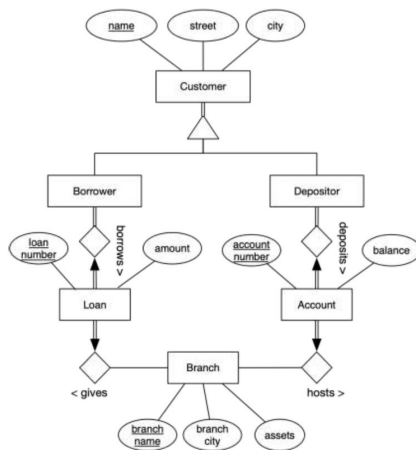
# Concepção de Bases de Dados

## Especificação de Requisitos

- requisito funcional 1:
- requisito funcional 2:
- ...
- restrição de integridade 1
- restrição de integridade 2
- ...



## Modelo Conceptual



## Modelo Relacional

R1 (x, y)

R2

Qualidade?

## Esquema Relacional (SQL)

branch			account			depositor	
branch_name	branch_city	assets	account_number	branch_name	balance	customer_name	account_number
Brighton	Brooklyn	710000	A-101	Downtown	500	Hayes	A-102
Downtown	Brooklyn	900000	A-102	Perryridge	400	Johnson	A-101
Mannus	Horseneck	400000	A-201	Brighton	900	Johnson	A-201
North Town	Rye	570000	A-215	Mannus	700	Jones	A-217
Perryridge	Horseneck	1700000	A-217	Brighton	750	Lindsay	A-222
Powert	Horseneck	300000	A-222	Redwood	798	Smith	A-215
Redwood	Palo Alto	2100000	A-305	Redwood	350	Turner	A-305
Round Hill	Horseneck	800000					

loan			borrower			customer		
loan_number	branch_name	amount	customer_name	loan_number	customer_city	customer_name	customer_city	customer_name
L-11	Round Hill	900	Adams	L-16	Spring	Pittsfield	Brooklyn	
L-14	Downtown	1500	Brooks	L-15	Senators	North	Rye	
L-15	Perryridge	1500	Curry	L-14	North	Woodsides	Stamford	
L-16	Perryridge	1300	Hayes	L-17	South	Woodsides	Stamford	
L-17	Downtown	1000	Jackson	L-11	Woodsides	Woodsides	Stamford	
L-23	Redwood	2000	Jones	L-11	Woodsides	Woodsides	Stamford	
L-25	Mannus	500	Smith	L-23	Woodsides	Woodsides	Stamford	
			Williams	L-17	Woodsides	Woodsides	Stamford	

# Qualidade do Esquema Relacional

- **Funcionalidade:**

- O esquema suporta todos os requisitos funcionais e captura todas restrições de integridade e relações semânticas entre os dados necessárias para o efeito

- **Atomicidade:**

- A manipulação (inserção, leitura, atualização e remoção) de factos independentes é feita de forma independente



# Exemplo de Não-Atomicidade

account_number	balance	customer_name	customer_city	branch_name	branch_city
A-101	500.00	Johnson	Palo Alto	Downtown	Brooklyn
A-215	700.00	Smith	Rye	Mianus	Horseneck
A-102	400.00	Hayes	Harrison	Perryridge	Horseneck
A-101	500.00	Hayes	Harrison	Downtown	Brooklyn
A-305	350.00	Turner	Stamford	Round Hill	Horseneck
A-201	900.00	Johnson	Palo Alto	Perryridge	Horseneck
A-217	750.00	Jones	Harrison	Brighton	Brooklyn
A-222	700.00	Lindsay	Pittsfield	Redwood	Palo Alto
A-333	850.00	Majeris	Rye	Central	Rye
A-444	625.00	Smith	Rye	North Town	Rye

- **Redundância:** os pares a vermelho não acrescentam nada à informação dos verdes
- **Dependência:** *balance* depende apenas de *account\_number*, *customer\_city* depende apenas de *customer\_name*, e *branch\_city* depende apenas de *branch\_name*; são factos independentes mas não conseguimos manipulá-los separadamente

# Consequências de Não-Atomicidade

- **Anomalias quando há alteração do estado:**
  - **Inserção:** não é possível inserir um item na BD a não ser que outro independente seja inserido também
  - **Remoção:** para remover um item, temos de remover outros itens independentes
  - **Atualização:** a atualização de um item implica a alteração de outros itens independentes
- Espaço de armazenamento: não necessariamente um problema, ou quanto muito um problema menor

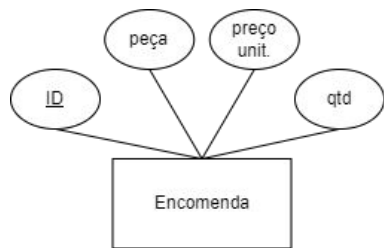
# Exemplos de Anomalias

account_number	balance	customer_name	customer_city	branch_name	branch_city
A-101	500.00	Johnson	Palo Alto	Downtown	Brooklyn
A-215	700.00	Smith	Rye	Mianus	Horseneck
A-102	400.00	Hayes	Harrison	Perryridge	Horseneck
A-101	500.00	Hayes	Harrison	Downtown	Brooklyn
A-305	350.00	Turner	Stamford	Round Hill	Horseneck
A-201	900.00	Johnson	Palo Alto	Perryridge	Horseneck
A-217	750.00	Jones	Harrison	Brighton	Brooklyn
A-222	700.00	Lindsay	Pittsfield	Redwood	Palo Alto
A-333	850.00	Majeris	Rye	Central	Rye
A-444	625.00	Smith	Rye	North Town	Rye

- **Inserção:** ao inserir uma conta temos que inserir também a cidade do cliente
- **Remoção:** ao remover a conta A-222 perdemos a informação de que Pittsfield mora em Palo Alto
- **Actualização:** mudar o saldo da conta A-101 requer actualizar múltiplas linhas

# Exemplos de Anomalias

**Encomendas referem-se a uma certa quantidade de uma única peça**



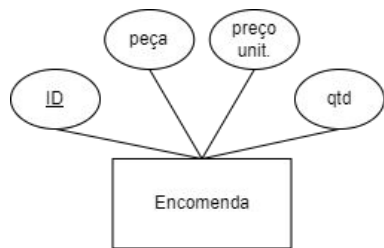
Encomenda (ID, peça, qtd, preço unit.)

ID	peça	qtd	preço unit.
121	12	4	€300
122	14	3	€2000
123	7	1	€750

- **Inserção:** só podemos indicar o preço de cada peça se existirem encomendas
- **Remoção:** ao remover uma encomenda, podemos perder a informação relativa ao preço de uma peça
- **Actualização:** alteração do preço de uma peça requer alterar múltiplas encomendas

# Exemplos de Anomalias

Encomendas referem-se a uma certa quantidade de uma única peça



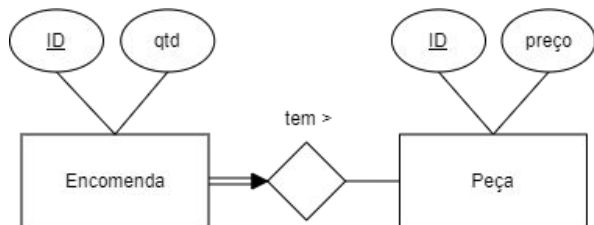
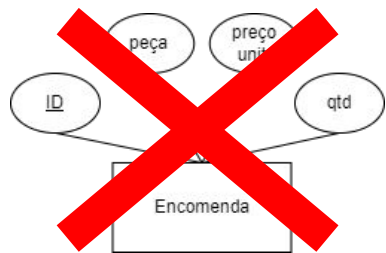
Encomenda (ID, Peça, Qtd, Preço Unit.)

ID	peça	qtd	preço unit.
121	12	4	€300
122	14	3	€2000
123	7	1	€750

- O problema está no desenho da base de dados
  - Diz-se que o esquema não está **Normalizado**
- Temos de corrigir o esquema relacional e também o modelo E-A

# Exemplos de Anomalias

Neste caso a solução é trivial



Peça (ID, preço)

Encomenda (ID, peça, qtd)  
peça: FK(Peça.ID)

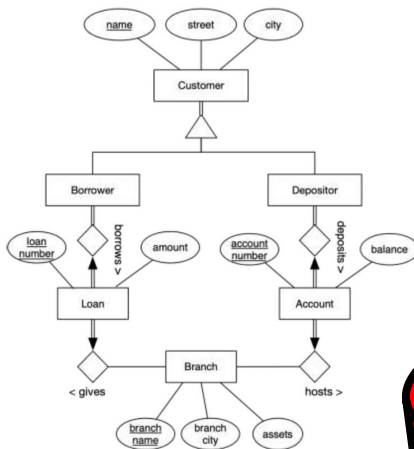
Peça		Encomenda		
ID	preço unit.	ID	peça	qtd
7	€750	121	12	4
12	€300	122	14	3
14	€2000	123	7	1

# Concepção de Bases de Dados

## Especificação de Requisitos

- requisito funcional 1:
- requisito funcional 2:
- ...
- restrição de integridade 1
- restrição de integridade 2
- ...

## Modelo Conceptual



## Modelo Relacional

R1 (x, y)  
R2 (x, z)

## Esquema Relacional (SQL)

branch		
branch_name	branch_city	assets
Brighton	Brooklyn	7100000
Downtown	Brooklyn	9000000
Manus	Horseneck	400000
North Town	Rye	3700000
Perryridge	Horseneck	1700000
Normal	Horseneck	300000
Redwood	Palo Alto	2100000
Round Hill	Horseneck	800000

account		
account_number	branch_name	balance
A-101	Downtown	500
A-102	Perryridge	400
A-201	Brighton	900
A-215	Manus	700
A-217	Brighton	750
A-222	Redwood	700
A-305	Round Hill	300

depositor	
customer_name	account_number
Hayes	A-102
Johnson	A-101
Jones	A-201
Lindsay	A-217
Smith	A-222
Turner	A-215
Turner	A-305

loan		
loan_number	branch_name	amount
L-11	Round Hill	900
L-14	Downtown	1500
L-15	Perryridge	1500
L-16	Perryridge	1500
L-17	Downtown	1000
L-23	Redwood	2000
L-25	Manus	500

borrower		
customer_name	loan_number	amount
Adams	L-16	1500
Curry	L-95	1500
Hayes	L-15	1500
Jackson	L-14	1500
Jones	L-17	1000
Smith	L-11	900
Williams	L-23	2000

customer			
customer_name	customer_city	customer_zip	customer_state
Adams	Spring	Pittsfield	Brooklyn
Brooks	Senator	Rye	Brooklyn
Curry	North	Woodside	Brooklyn
Glenn	Sand Hill	Woodside	Brooklyn
Groen	Walnut	Stanford	Brooklyn
Hayes	Main	Harrison	Brooklyn
Johnson	Main	Harrison	Brooklyn
Jones	Park	Pittsfield	Brooklyn
Lindsay	North	Rye	Brooklyn
Smith	Putnam	Stanford	Brooklyn
Turner	Putnam	Stanford	Brooklyn
Williams	Nassau	Princeton	Brooklyn

Normalização

# Normalização

- Na vasta maioria das situações, podemos desenhar modelos Entidade Associação que resultam em esquemas relacionais normalizados
  - Nem sempre é fácil e requer alguma experiência
- A Teoria da Normalização está fundamentada no modelo Relacional, mas não no modelo Entidade Associação
- Na prática, podemos adotar um princípio simples: uma entidade ou associação do modelo Entidade-Associação está normalizado quando as relações resultantes estão normalizadas



# Normalização

- A Teoria da Normalização visa:
  - Reduzir a redundância da informação
  - Garantir a atomicidade dos factos
  - Transformar as RIs resultantes de Dependências Funcionais em restrições de chaves candidatas suportadas pelo modelo relacional
- Da Teoria da Normalização, iremos abordar os seguintes conceitos:
  - Dependências Funcionais
  - Formas Normais
  - Decomposições de Relações

# Dependências Funcionais

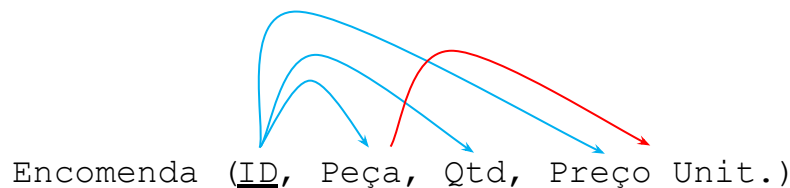
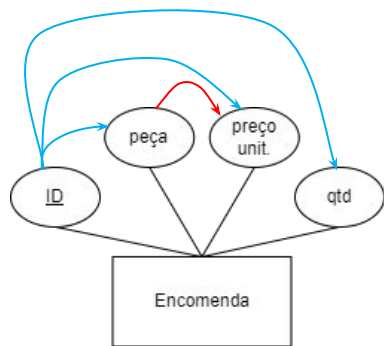
# Dependências Funcionais

- Seja  $R$  uma relação que contém os conjuntos de atributos  $X$  e  $Y$ 
  - Diz-se que  $X$  **determina (funcionalmente)**  $Y$  se, para cada instância válida de  $X$  (i.e. cada conjunto de valores possível) existe uma só instância válida de  $Y$ 
    - Ou seja, sabendo  $X$ , sabemos  $Y$
  - Esta relação é representada pela notação:  $X \rightarrow Y$
  - Diz-se que  $X$  é determinante, e  $Y$  é dependente

# Dependências Funcionais

- Formalmente, uma dependência funcional não pode ser deduzida pela observação dos dados existentes numa relação
  - Só pela observação de todos os dados possíveis, o que raramente é exequível
- Por observação, apenas podemos determinar que há independência entre atributos
- É preciso conhecimento do domínio, em complemento da observação dos dados, para determinar dependências funcionais

# Dependências Funcionais

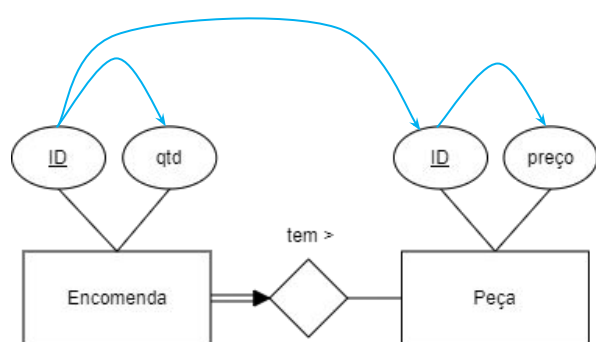


**Dependências Funcionais:**

**ID → (Peça, Qtd, Preço Unit.)**

**Peça → Preço Unit.**

# Dependências Funcionais



Encomenda (ID, Peça, Qtd)

**Dependências Funcionais:**  
**ID → (Peça, Qtd)**

Peça (ID, Preço Unit.)

**Dependências Funcionais:**  
**ID → Preço Unit.**

# Propriedades das Dependências Funcionais

## Axiomas de Armstrong

- **Reflexividade/Reflexivity:**  $Y \subseteq X \Rightarrow X \rightarrow Y$ 
  - Qualquer conjunto de atributos é funcionalmente dependente de qualquer dos seus super-conjuntos
- **Incremento/Augmentation:**  $\forall Z, X \rightarrow Y \Rightarrow XZ \rightarrow YZ$ 
  - Qualquer número de atributos podem ser adicionados em simultâneo ao determinante e dependente
- **Transitividade/Transitivity:**  $X \rightarrow Y \wedge Y \rightarrow Z \Rightarrow X \rightarrow Z$ 
  - Um dependente de um dependente é dependente do primeiro determinante

# Propriedades das Dependências Funcionais

## Corolários (Triviais)

- **Auto-reflexividade:**  $X \rightarrow X$ 
  - Qualquer conjunto de atributos depende de si mesmo
- **Decomposição:**  $X \rightarrow YZ \Rightarrow X \rightarrow Y \wedge X \rightarrow Z$ 
  - Cada um dos atributos do dependente é dependente do determinante
- **União:**  $X \rightarrow Y \wedge X \rightarrow Z \Rightarrow X \rightarrow YZ$ 
  - Dois dependentes do mesmo determinante podem ser unidos
- **Composition:**  $X \rightarrow Y \wedge A \rightarrow B \Rightarrow XA \rightarrow YB$ 
  - Quaisquer dois pares de dependências funcionais podem ser combinados



# Propriedades das Dependências Funcionais

## Corolários (Triviais)

- **Auto-reflexividade:**  $X \rightarrow X$ 
  - Qualquer conjunto de atributos depende de si mesmo
- **Decomposição:**  $X \rightarrow YZ \Rightarrow X \rightarrow Y \wedge X \rightarrow Z$ 
  - Cada um dos atributos do dependente é dependente do determinante
- **União:**  $X \rightarrow Y \wedge X \rightarrow Z \Rightarrow X \rightarrow YZ$ 
  - Dois dependentes do mesmo determinante podem ser unidos
- **Composition:**  $X \rightarrow Y \wedge A \rightarrow B \Rightarrow XA \rightarrow YB$ 
  - Quaisquer dois pares de dependências funcionais podem ser combinados

# Exemplo de Utilização das Propriedades

- Seja  $r = (A, B, C, G, H, I)$ 
  - Com:  $A \rightarrow B, A \rightarrow C, CG \rightarrow H, CG \rightarrow I, B \rightarrow H$
- Podemos deduzir que:
  - $A \rightarrow H$ 
    - Transitividade:  $A \rightarrow B \wedge B \rightarrow H$
  - $AG \rightarrow I$ 
    - Aumento + Transitividade:  $AG \rightarrow CG \wedge CG \rightarrow I$

# Fecho de Atributos (Closure)

- Fecho  $\alpha^+$  de um conjunto de atributos  $\alpha$  com respeito a um conjunto de dependências funcionais, é conjunto de atributos  $\gamma$  tal que  $\alpha \rightarrow \gamma$  pode ser inferida pelos axiomas de Armstrong
- Algoritmo:

```
result =  $\alpha$ 
while (result changed) do
  for each  $\gamma \rightarrow \beta$  in F do
    if  $\gamma \subseteq \text{result}$  then result := result  $\cup \beta$ 
```

# Exemplo de Cálculo do Fecho de Atributos

- Seja  $r = (A, B, C, G, H, I)$

- Com:  $A \rightarrow B, A \rightarrow C, CG \rightarrow H, CG \rightarrow I, B \rightarrow H$

- Calcular  $(AG)^+$

1.  $result = AG$

2.  $A \rightarrow B, A \subseteq result \Rightarrow result := result \cup B = ABG$

3.  $A \rightarrow C, A \subseteq result \Rightarrow result := result \cup C = ABCG$

4.  $CG \rightarrow H, CG \subseteq result \Rightarrow result := result \cup H = ABCGH$

5.  $CG \rightarrow I, CG \subseteq result \Rightarrow result := result \cup I = ABCGHI$

# Superchaves e Chaves Candidatas

# Chaves e Dependências Funcionais

- Dada uma relação  $r$  definida pelo conjunto de atributos  $R$ , i.e.  $r(R)$ 
  - $K \subseteq R$  é uma **superchave** de  $r(R)$  se  $K \rightarrow R$ 
    - Uma superchave é qualquer conjunto de atributos que determina funcionalmente toda a relação
  - $K \subseteq R$  é uma **chave candidata** de  $r(R)$  sse  $K \rightarrow R \wedge \nexists \alpha \subset K: \alpha \rightarrow R$ 
    - Uma chave candidata é um conjunto de atributos que determina funcionalmente toda a relação e não contém nenhum subconjunto de atributos que também o determina
- Atributos que fazem parte de qualquer chave candidata são chamados atributos-chave (prime)

# Chaves e Dependências Funcionais

- **Super-chave:**

- Qualquer conjunto de atributos que sirva para identificar univocamente os tuplos de uma relação (pode conter mais atributos do que o necessário)

- **Chave candidata:**

- Conjunto mínimo de atributos necessário para identificar univocamente os tuplos de uma relação ( se se retirar um atributo, deixa de ser chave)
  - Podem haver várias chaves candidatas

- **Chave primária:**

- Uma das chaves candidatas (não importa qual)

# Determinação de Chaves Candidatas

- Podemos calcular o fecho de todas as combinações de atributos
- Mas a maneira mais prática é começar com o conjunto completo de atributos (superchave trivial) e reduzi-lo gradualmente:
  - Para cada dependência, retirar os dependentes da superchave, sempre que o determinante esteja na superchave
  - Verificar se a superchave resultante é chave candidata (decompor e verificar fecho)
  - Verificar se existem outras superchaves (se os atributos chave existem enquanto dependentes)



# Determinação de Chaves Candidatas

- Exemplo 1:  $r(A,B,C,D)$  com  $A \rightarrow B$ ,  $B \rightarrow C$ ,  $C \rightarrow D$ 
  1. Superchave  $ABCD$
  2. Remover dependentes
    - a.  $A \rightarrow B$ ,  $A$  está na superchave  $\Rightarrow$  superchave  $ACD$
    - b.  $A \rightarrow B \rightarrow C$ ,  $A$  está na superchave  $\Rightarrow$  superchave  $AD$
    - c.  $A \rightarrow B \rightarrow C \rightarrow D$ ,  $A$  está na superchave  $\Rightarrow$  superchave  $A$
  3.  $A$  não pode ser decomposta, portanto é chave candidata
  4.  $A$  não existe enquanto dependente, portanto não há mais chaves candidatas

# Determinação de Chaves Candidatas

- Exemplo 2:  $r(A,B,C,D)$  com  $A \rightarrow B$ ,  $B \rightarrow C$ ,  $C \rightarrow A$ 
  1. Superchave  $ABCD$
  2. Remover dependentes
    - a.  $A \rightarrow B$ ,  $A$  está na superchave  $\Rightarrow$  superchave  $ACD$
    - b.  $A \rightarrow B \rightarrow C$ ,  $A$  está na superchave  $\Rightarrow$  superchave  $AD$
    - c.  $C \rightarrow A$ ,  $C$  não está na superchave, não podemos remover
  3. Verificar se  $AD$  é chave candidata:  $A^+ = ABC$ ,  $D^+ = D$  portanto sim
  4. Verificar se há outras chaves: ...

# Determinação de Chaves Candidatas

- Exemplo 2:  $r(A,B,C,D)$  com  $A \rightarrow B$ ,  $B \rightarrow C$ ,  $C \rightarrow A$
- 4. Verificar se há outras chaves ( $A$  ou  $D$  ocorrem como dependentes)
  - a.  $A$  existe como dependente em  $C \rightarrow A$ , portanto  $CD$  também é superchave;  $C^+ = ABC$  e  $D^+ = D$ , portanto é chave candidata
- 4. Verificar se há outras chaves ( $A$ ,  $C$  ou  $D$  ocorrem como dependentes)
  - a.  $C$  existe como dependente em  $B \rightarrow C$ , portanto  $BD$  também é superchave;  $B^+ = ABC$  e  $D^+ = D$ , portanto é chave candidata
- 4. Verificar se há outras chaves ( $A$ ,  $B$ ,  $C$  ou  $D$  ocorrem como dependentes)
  - a.  $B$  existe como dependente em  $A \rightarrow B$ , mas já cobrimos  $A$

# Formas Normais

# Formas Normais

- As formas normais são uma forma de classificar relações relativamente às redundâncias que podem existir
- Iremos estudar as formas normais associadas a redundâncias detectadas por dependências funcionais:
  - Primeira Forma Normal
  - Segunda Forma Normal
  - Terceira Forma Normal
  - Forma Normal Boyce-Codd
- Fica de fora o estudo das formas normais associadas a dependências multivalor (Quarta Forma Normal) e joins multivalor (Quinta Forma Normal)

# 1ª Forma Normal

- Uma relação está na 1ª forma normal se todos os seus atributos são atômicos
- Intervalos de tempo ou pares de coordenadas GPS podem ser considerados atômicos (se suportados pelo SGBD)
- Tuplos ou dados estruturados de dimensão variável são no entanto uma violação à 1ª Forma Normal
- A 1FN só garante que existe uma chave, não impede anomalias devido a dependências funcionais

Relação não normalizada

Pessoa	Cidade	Datas
João	Lisboa	01-02-92 01-02-93
	Faro	10-01-94
José	Lisboa	10-10-84

Relação normalizada

Pessoa	Cidade	Datas
João	Lisboa	01-02-92
João	Lisboa	01-01-93
João	Faro	10-01-94
José	Lisboa	10-10-84

## 2ª Forma Normal

- Uma relação está na 2ª forma normal se:
  - Está na 1FN
  - Todos os atributos-não-chave são funcionalmente dependentes de chaves candidatas na sua totalidade (não de subconjuntos delas)
- Não impede dependências funcionais entre atributos-não-chave ou entre atributos-chave

<u>Manufacturer</u>	<u>Model</u>	<u>Manufacturer country</u>
Forte	X-Prime	Italy
Forte	Ultraclean	Italy
Dent-o-Fresh	EZbrush	USA
Brushmaster	SuperBrush	USA
Kobayashi	ST-60	Japan
Hoch	Toothmaster	Germany
Hoch	X-Prime	Germany

Viola a 2FN pois **Manufacturer country** depende apenas de **Manufacturer**

# 3ª Forma Normal

- Uma relação está na 3ª forma normal se:
  - Está na 2FN
  - Não há dependências funcionais entre atributos-não-chave
  - Ou seja, para cada dependência funcional não trivial  $X \rightarrow Y$ ,  $X$  é uma superchave ou  $Y$  é um atributo-chave
- Não impede dependências funcionais entre atributos-chave

<u>Tournament</u>	<u>Year</u>	<u>Winner</u>	<u>Winner's date of birth</u>
Indiana Invitational	1998	Al Fredrickson	21 July 1975
Cleveland Open	1999	Bob Albertson	28 September 1968
Des Moines Masters	1999	Al Fredrickson	21 July 1975
Indiana Invitational	1999	Chip Masterson	14 March 1977

Viola a 3FN pois Winner's date of birth depende de Winner



# Forma Normal Boyce-Codd

- Uma relação está na forma normal Boyce-Codd se:
  - Está na 3FN
  - Cada atributo é dependente de uma chave candidata na sua totalidade
    - $\forall X \rightarrow Y, X$  é uma chave candidata
  - Ou seja, não há redundâncias detectáveis por dependências funcionais

Court	Start time	End time	Rate type
1	09:30	10:30	SAVER
1	11:00	12:00	SAVER
1	14:00	15:30	STANDARD
2	10:00	11:30	PREMIUM-B
2	11:30	13:30	PREMIUM-B
2	15:00	16:30	PREMIUM-A

Todos os atributos são atributos-chave, por isso está na 3FN, mas viola a FNBC pois  $\text{Rate type} \rightarrow \text{Court}$  e  $\text{Rate type}$  não é uma chave candidata (embora faça parte de duas)

# Formas Normais

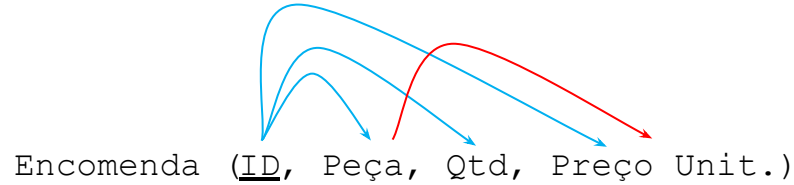
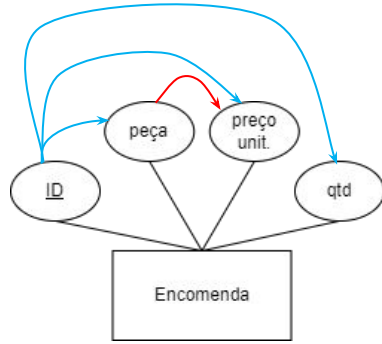
Forma	Resumo
1ª Forma Normal	Atributos atómicos e unidimensionais $\Rightarrow$ há uma <b>chave</b>
2ª Forma Normal	1ªFN + atributos-não-chave dependem de <b>toda a chave</b>
3ª Forma Normal	2ªFN + atributos-não-chave dependem <b>apenas da chave</b>
FN Boyce-Codd	3ªFN + <b>atributos-chave</b> dependem de <b>toda a chave</b>

TNF: “Each non-prime attribute is a fact about **the key, the whole key, and nothing but the key**, so help me Codd”

# Formas Normais

- Uma relação que tem apenas uma chave candidata unária está sempre na 2FN
- Uma relação na 2FN que tem apenas um atributo-não-chave está sempre na 3FN
- Uma relação que tem apenas atributos-chave está sempre na 3FN
- Uma relação na 3FN só pode não estar na FNBC se tiver múltiplas chaves candidatas sobrepostas

# Exemplo



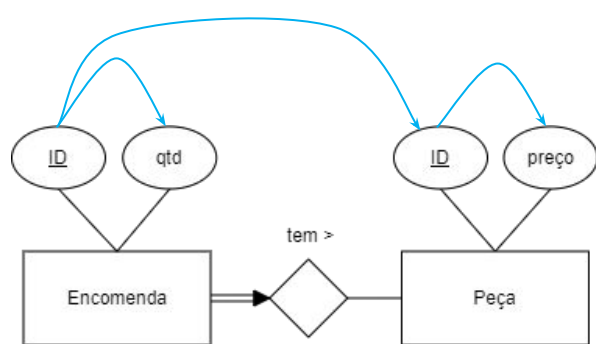
**Dependências Funcionais:**

**ID → (Peça, Qtd, Preço Unit.)**

**Peça → Preço Unit.**

- 2FN: ✓
- 3FN: ✗ nem Peça é superchave nem Preço é chave
- FNBC: ✗

# Dependências Funcionais



Encomenda (ID, Peça, Qtd)

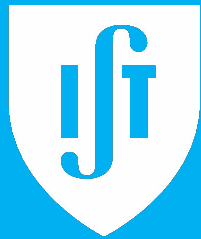
**Dependências Funcionais:**  
**ID → (Peça, Qtd)**

Peça (ID, Preço Unit.)

**Dependências Funcionais:**  
**ID → Preço Unit.**

- 2FN: ✓
- 3FN: ✓
- FNBC: ✓

- 2FN: ✓
- 3FN: ✓
- FNBC: ✓



**TÉCNICO** LISBOA