

### 3. Teoria da Normalização

Ao modelar a informação procura-se:

- . Um modelo que represente fielmente a realidade
- . Um modelo capaz de responder às funcionalidades que se pretendem

Queremos obter um modelo com propriedades que garantam:

- . Redundância mínima
- . Facilidade de Manutenção
- . Estabilidade face a futuras alterações

#### Dados redundantes

EmpregadoDepartamento

<u>NumEmp</u>	Nome	Categoria	Salário	Dep	TelDep	LocalDep
10	José da Silva	Programador	2500	1	213334555	Lisboa
20	Maria Costa	Analista	5000	2	224446888	Porto
30	João Fonseca	Operador	600	1	213334555	Lisboa
40	Ana Faria	Analista	5200	4	275222333	Covilhã

Nesta tabela existem dados redundantes. Os dados de um dado departamento são repetidos para cada empregado desse departamento.

Se apagarmos este número de departamento deixamos de saber qual o departamento do empregado 30.  
Dado duplicado mas não redundante

Dados redundantes.  
Se apagarmos esta informação continuamos a saber os dados do departamento 1 !!

*Relações que têm dados redundantes podem vir a ter anomalias de inserção, eliminação ou modificação.*

#### Anomalias de inserção:

- . Para inserir um novo empregado temos que inserir toda a informação do departamento a que pertence tendo o cuidado de não criar inconsistências com a informação já existente.
- . Não é possível criar um departamento que ainda não tenha empregados. Note-se que o atributo NumEmp é chave da relação logo o seu valor não pode ser nulo.

#### Anomalias de eliminação:

- . Se eliminarmos um empregado que seja o único empregado de um dado departamento perdemos a informação desse departamento.

#### Anomalias de modificação:

- . Se quisermos alterar um atributo de um dado departamento (por ex. o telefone do dep. 1) temos que actualizar o valor do atributo em todos os empregados que pertencem a esse departamento.

Podemos evitar as anomalias anteriores se decomposermos a relação EmpregadoDepartamento nas relações:

Empregado (NumEmp, Nome, Categoria, Salário, Dep) e  
Departamento(Dep, TelDep, LocalDep).

A teoria da normalização, desenvolvida por Edgar Codd no âmbito do modelo relacional, define um processo de estruturar as tabelas de uma base de dados de forma a minimizar a redundância de dados.

A teoria da normalização utiliza o conceito da Dependência Funcional que vamos começar por estudar:

### 3.1. Dependências Funcionais

Seja a relação:

Morada ( Nome, Endereço , Cidade , CodPostal, Telefone )

**Assumindo que todos os nomes são diferentes**, podemos descrever as seguintes dependências entre os atributos da relação morada:

- i) Dado um Nome específico, este determina um único tuplo da relação Morada  
(isto é, determina valores únicos para os atributos Endereço, Cidade, CodPostal, Telefone)
- ii) Dados valores dos atributos Endereço e Cidade, todos os tuplos da relação Morada com esses valores (se existirem) têm o mesmo valor do atributo CodPostal.

- iii) A um determinado valor de CodPostal corresponde um único valor do atributo Cidade.

As associações lógicas descritas entre os atributos da relação Morada são denominadas **dependências lógicas**.

Tipos de dependências lógicas:

- . Dependências Funcionais
- . Dependências Multivalor (a estudar mais tarde)
- . Dependências de Junção (a estudar mais tarde)

### (1) Definição de Dependência Funcional (DF)

Seja  $R(A_1, A_2, \dots, A_n)$  um esquema de relação e  $X$  e  $Y$  subconjuntos de  $\{A_1, A_2, \dots, A_n\}$

Dizemos que existe uma Dependência Funcional entre  $X$  e  $Y$  e escrevemos  $X \rightarrow Y$  ( $X$  determina  $Y$ ) sse em qualquer instante  $t$ , quaisquer tuplos de  $R$  com o mesmo valor de  $X$  têm necessariamente o mesmo valor de  $Y$ .

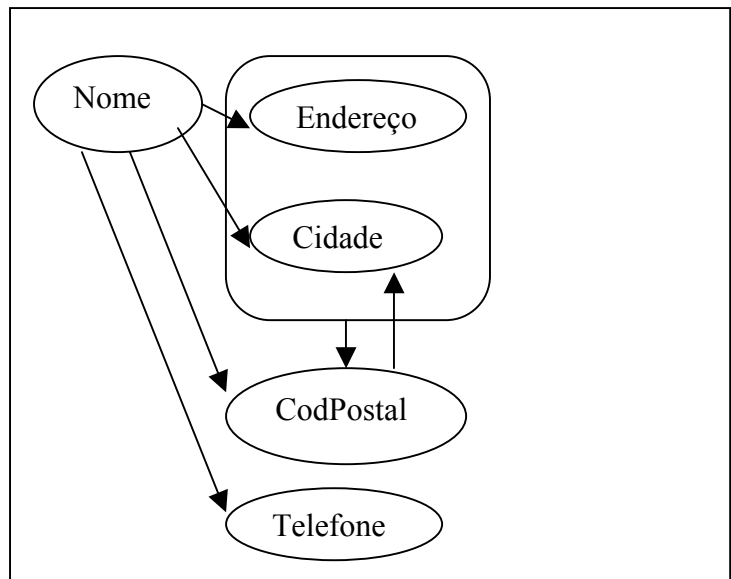
### Diagrama de Dependência Funcional



- .  $X$  determina  $Y$
- .  $Y$  depende de  $X$

**(2) Exemplo**

- Nome → Endereço
- Nome → Cidade
- Nome → CodPostal
- Nome → Telefone
  
- Endereço , Cidade → CodPostal
- CodPostal → Cidade



**(3) Chave (candidata)**

Seja a relação R (A<sub>1</sub>, A<sub>2</sub>, ... , A<sub>n</sub>) e X um conjunto de atributos de R  
 (X ⊆ { A<sub>1</sub>, A<sub>2</sub>, ... , A<sub>n</sub> } )

X é chave de R sse

(i)  $\forall_i X \rightarrow A_i \quad i = 1,2,\dots,n$

X determina funcionalmente todos os atributos de R

(ii)  $\nexists Y \subset X : \forall_i Y \rightarrow A_i \quad i = 1,2,\dots,n$

Não existe um subconjunto de X que determine todos os atributos de R

➔ Nome é a única chave da relação morada

#### **(4) Super-chave**

Se um conjunto de atributos X satisfaz a condição (1) mas não a (2) é denominado Super-chave da relação.

→ *Qualquer conjunto de atributos que contenha o atributo Nome é super-chave da relação morada*

#### **(5) Toda a relação tem uma chave**

*Demonstração:*

Dada uma relação R ( $A_1, A_2, \dots, A_n$ ) verifica-se que

$$A_1, A_2, \dots, A_n \rightarrow A_i \quad i = 1, 2, \dots, n$$

Se não existe um  $X \subset \{A_1, A_2, \dots, A_n\}$  tal que  $X \rightarrow A_i$  então

$$A_1, A_2, \dots, A_n \text{ é a chave de R.}$$

Caso contrário X contém uma chave.

#### **(6) Chave Primária**

- É a chave candidata escolhida.
- Nenhuma das suas ocorrências pode ter valor nulo.

## (7) Propriedades básicas das DF's

### (i) Unicidade

Sejam  $f: X \rightarrow Y$  e  $g: X \rightarrow Y$ , dependências funcionais,  
então  $f = g$

### (ii) Reflexibilidade

Se  $X \subseteq Y$  então existe a dependência funcional  $Y \rightarrow X$

Dependência Funcional Trivial: todo o conjunto de atributos determina todos os seus subconjuntos.

### (iii) Transitividade

Se  $X \rightarrow Y$  e  $Y \rightarrow Z$  então  $X \rightarrow Z$

### (iv) União

Se  $X \rightarrow Y$  e  $X \rightarrow Z$  então  $X \rightarrow YZ$

### Exemplo:

*Voltando à relação Morada,*

Morada ( Nome, Endereço , Cidade , CodPostal, Telefone ).

*Vimos que existiam as Df's:*

Nome  $\rightarrow$  Endereço

Nome → Cidade

Nome → CodPostal

Nome → Telefone

Endereço , Cidade → CodPostal

CodPostal → Cidade

A – Por reflexibilidade,

Endereço , Cidade → Endereço

Endereço , Cidade → Cidade [1]

B – De “Endereço , Cidade → CodPostal” e “CodPostal → Cidade” ,

por transitividade obtemos

Endereço , Cidade → Cidade [2]

C – Pela unicidade [1] e [2] são a mesma dependência funcional.

D – De “Nome → Telefone” e “Nome → Endereço” obtemos por união

Nome → Telefone, Endereço

Por união obtemos também Nome → Cidade, CodPostal

Novamente por união Nome → Telefone, Endereço, Cidade, CodPostal



o que significa que Nome é chave da relação Morada.

## **(8) Propriedades derivadas das DF's**

### (i) Distributividade (decomposição)

Se  $X \rightarrow YZ$  então  $X \rightarrow Y$  e  $X \rightarrow Z$

*Dem.*

$YZ \rightarrow Y$  e  $YZ \rightarrow Z$  (por reflexibilidade)

Se  $X \rightarrow YZ$  e  $YZ \rightarrow Y$  então por transitividade  $X \rightarrow Y$  q.e.d.<sup>1</sup>

Se  $X \rightarrow YZ$  e  $YZ \rightarrow Z$  então por transitividade  $X \rightarrow Z$  q.e.d.

### (ii) Aumento

Se  $X \rightarrow Y$  e  $W \rightarrow Z$  então  $XW \rightarrow YZ$

*Dem.*

$XW \rightarrow X$  (reflexibilidade) e  $X \rightarrow Y \Rightarrow$  (por transitividade)  $XW \rightarrow Y$

$XW \rightarrow W$  (reflexibilidade) e  $W \rightarrow Z \Rightarrow$  (por transitividade)  $XW \rightarrow Z$

$XW \rightarrow Y$  e  $XW \rightarrow Z \Rightarrow$  (por união)  $XW \rightarrow YZ$  q.e.d.

---

<sup>1</sup> "quod erat demonstrandum"

(iii) Pseudo – transitividade

Se  $X \rightarrow Y$  e  $YW \rightarrow Z$  então  $XW \rightarrow Z$

*Dem.*

$X \rightarrow Y$  e  $W \rightarrow W$  (por reflexibilidade)  $\Rightarrow$  (por aumento)  $XW \rightarrow YW$

$XW \rightarrow YW$  e  $YW \rightarrow Z$   $\Rightarrow$  (por transitividade)  $XW \rightarrow Z$

Exemplo:

Aplicando a pseudo-transitividade às dependências funcionais

Nome  $\rightarrow$  Endereço e Endereço, Cidade  $\rightarrow$  CodPostal

obtemos Nome, Cidade  $\rightarrow$  CodPostal

### **(9) Perda de informação**

Seja a relação

R (Nome, Telefone, Cidade)

em que os atributos Telefone e Cidade não dependem funcionalmente do atributo Nome

Nome  $\not\rightarrow$  Telefone

Nome  $\rightarrow$  Cidade

(isto é, uma pessoa pode ter mais que um telefone numa ou mais cidades)

Num dado instante podemos ter;

R

Nome	Telefone	Cidade
José da Silva	123456789	Leiria
José da Silva	222222222	Faro
António Costa	333333333	Leiria

As projecções

$R1 = \Pi_{\langle \text{Nome}, \text{Telefone} \rangle} (R)$  e  $R2 = \Pi_{\langle \text{Nome}, \text{Cidade} \rangle} (R)$ , no mesmo instante de tempo, teriam como resultado

R1

Nome	Telefone
José da Silva	123456789
José da Silva	222222222
António Costa	333333333

R2

Nome	Cidade
José da Silva	Leiria
José da Silva	Faro
António Costa	Leiria

A junção das duas projecções sobre o atributo Nome,

$R1 \bowtie_{\langle \text{Nome} = \text{Nome} \rangle} R2$ , é representada pela tabela

Nome	Telefone	Cidade
José da Silva	123456789	Leiria
José da Silva	123456789	Faro
José da Silva	222222222	Leiria
José da Silva	222222222	Faro
António Costa	333333333	Leiria



A junção da decomposição não é igual à relação inicial !!!

**A relação não pode ser decomposta desta forma.**

**(10) Decomposição sem perda (Lossless Join)**

Seja  $R(X, Y, Z)$  com  $X, Y$  e  $Z$  conjuntos de atributos.

Se  $X \rightarrow Y$  ou  $X \rightarrow Z$  então

$$R(X, Y, Z) = \Pi_{\langle X, Y \rangle}(R) \bowtie_{\langle X=X \rangle} \Pi_{\langle X, Z \rangle}(R)$$

Dem:

- 1) Observemos em primeiro lugar que  $R(X, Y, Z)$  é sempre um subconjunto de  $\Pi_{\langle X, Y \rangle}(R) \bowtie_{\langle X=X \rangle} \Pi_{\langle X, Z \rangle}(R)$



Seja  $xyz$  um tuplo de  $R(X, Y, Z)$ ,

- se  $xyz$  está em  $R(X, Y, Z)$  então  $xy$  está em  $\Pi_{\langle X, Y \rangle}(R)$  e  $xz$  está em  $\Pi_{\langle X, Z \rangle}(R)$

- a junção dos tuplos  $xy$  e  $xz$  dá o tuplo  $xyz$

$$\text{Portanto } xyz \in \prod_{\langle X, Y \rangle} (R) \bowtie_{\langle X=X \rangle} \prod_{\langle X, Z \rangle} (R)$$

2 ) Falta provar que se  $X \rightarrow Y$  ou  $X \rightarrow Z$  então

$$\prod_{\langle X, Y \rangle} (R) \bowtie_{\langle X=X \rangle} \prod_{\langle X, Z \rangle} (R) \subseteq R(X, Y, Z)$$

(i) Supondo que  $X \rightarrow Y$ ,

( neste caso se  $xyz \in R(X, Y, Z)$  e  $xy'z' \in R(X, Y, Z)$  então  $y = y'$  )

Se  $xyz \in \prod_{\langle X, Y \rangle} (R) \bowtie_{\langle X=X \rangle} \prod_{\langle X, Z \rangle} (R)$  então existem  $y'$  e  $z'$  tais que

$$xyz' \in R(X, Y, Z) \text{ e } xy'z \in R(X, Y, Z)$$

Mas  $X \rightarrow Y$  (i)  $\Rightarrow y = y'$  logo  $xyz \in R(X, Y, Z)$

Portanto cada elemento de  $\prod_{\langle X, Y \rangle} (R) \bowtie_{\langle X=X \rangle} \prod_{\langle X, Z \rangle} (R)$   
 é elemento de  $R(X, Y, Z)$

B

A
e
B

$\Rightarrow$

$$R(X, Y, Z) = \prod_{\langle X, Y \rangle} (R) \bowtie_{\langle X=X \rangle} \prod_{\langle X, Z \rangle} (R)$$

**q.e.d.**

### 3.2. Normalização

A normalização de uma relação é obtida pela sua decomposição em duas ou mais relações de acordo com um procedimento bem definido.

Três níveis de normalização foram definidos por Codd:

*1ª Forma Normal*

*2ª Forma Normal*

*3ª Forma Normal*

Posteriormente R. Boyce e Codd definiram a

*Forma Normal de Boyce-Codd (FNBC)*

Mais tarde Fagin propôs:

*4ª Forma Normal*

*5ª Forma Normal*

. Uma relação numa forma normal mais avançada tem menos dados redundantes.

. Se uma relação está numa forma normal mais avançada também está nas formas normais anteriores.

