

# Banco de Dados I

## Normalização

Profa. Leticia T. M. Zoby

[leticia.zoby@udf.edu.br](mailto:leticia.zoby@udf.edu.br)

# Introdução

- ▶ O processo de normalização pode ser visto como processo no qual são eliminados esquemas de relações (tabelas) não satisfatórios, decompondo-os, através da separação de seus atributos em esquemas de relações menos complexas mas que satisfaçam as propriedades desejadas.
- ▶ A Dependência Funcional é a principal ferramenta para medir o “quão apropriado” é o agrupamento de atributos numa relação.

# Introdução

## ► Análise de Dependência Funcional

### ► Dependência Funcional

O atributo B é funcionalmente dependente do atributo A se, em qualquer instante, um valor em A determina, de modo único, o valor correspondente em B, na mesma relação

Exemplo:

Num-Emp  $\rightarrow$  Nome-Emp

O Nome-Emp é funcionalmente dependente do Num-Emp, pelo fato de cada Num-Emp está associado sempre ao mesmo Nome-Emp.

EMPREGADO		
#Num-Emp	Nome-Emp	Vlr-Sal-Emp

# Introdução

## ► Análise de Dependência Funcional

### ► Dependência Funcional

- $A \rightarrow B$ , lê - se:
  - $A$  *funcionalmente determina*  $B$
  - $B$  é *funcionalmente dependente* de  $A$
  - $B$  é *função* de  $A$
- Para cada valor de  $A$  só existe um valor de  $B$
- $A \nrightarrow B$ , negação de  $A \rightarrow B$ .

$A$  ou  $B$  podem ser um conjunto de atributos.

# Introdução

## ► Dependência Funcional

Propõe três tipos de dependência entre os atributos de uma tabela

► Dependência Funcional Total

► Dependência Funcional Parcial

► Dependência Funcional Transitiva

# Introdução

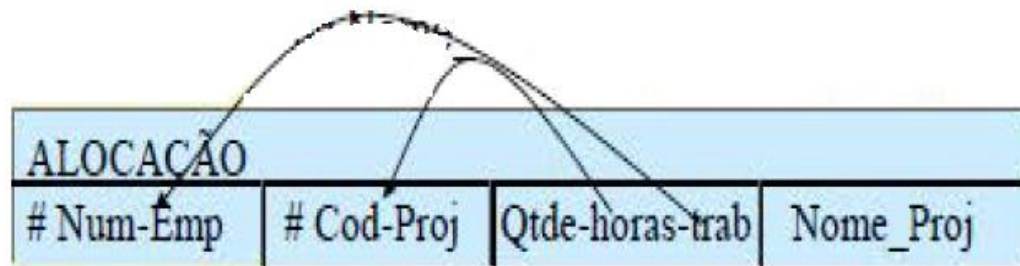
## • Dependência Funcional Total

- Os atributos de uma tabela tem que depender da chave primária e somente da chave primária
- Um atributo C é totalmente funcionalmente dependente da chave primária composta pelo atributos A e B, quando for funcionalmente dependente de A e B não dependente funcionalmente de qualquer parte da chave primária

Exemplo:

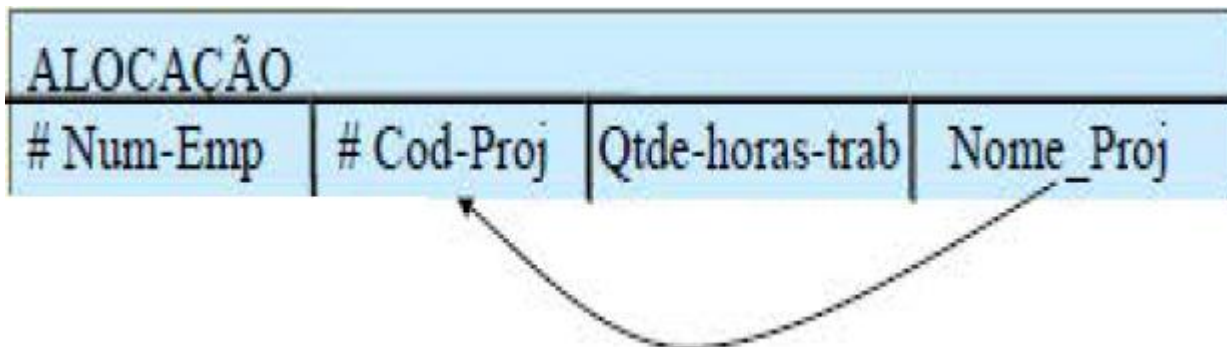
Qtde\_horas depende da chave primária totalmente.

Já Nome\_Proj somente de cod\_proj.



# Introdução

- Dependência Funcional Parcial
  - O atributo C é parcialmente funcionalmente dependente da chave primária composta pelos atributos A e B quando for funcionalmente dependente de A ou B e não de ambos A e B.
  - A dependência funcional parcial ocorre quando a chave primária da relação é composta e se constitui numa anomalia que se deve ser evitada



# Introdução

- Dependência Funcional Parcial
  - Solução: criação de uma nova relação composta pelo atributo ou atributos que dependem de parte da chave e a chave que determine, de modo único, estes atributos

ALOCAÇÃO		
# Num-Emp	# Cod-Proj	Qtde-horas-trab

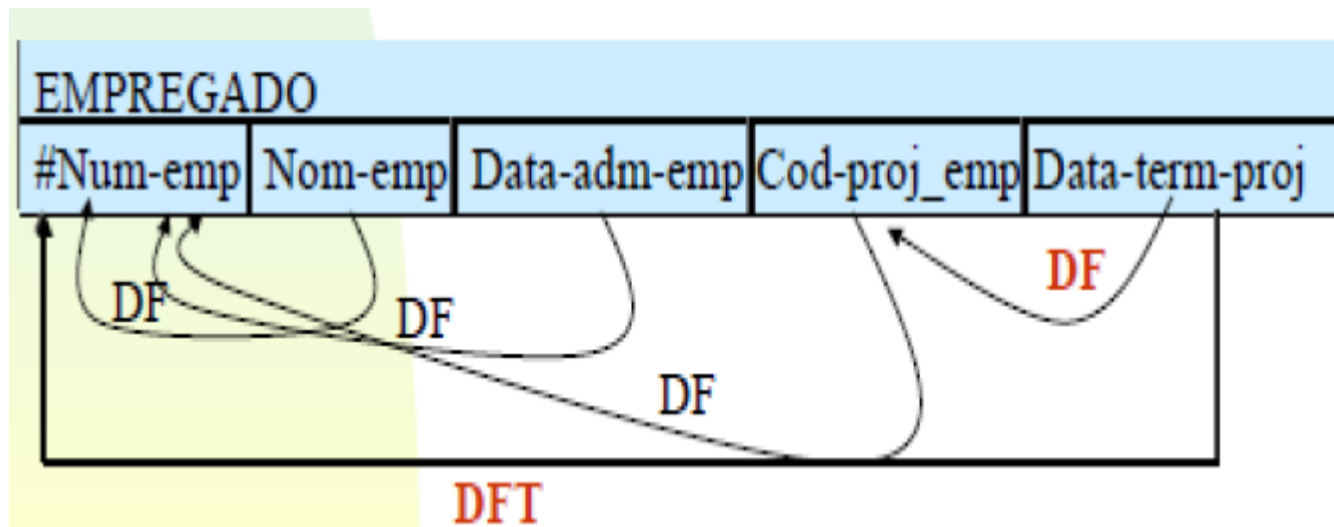
PROJETO	
# Cod-Proj	Nome_Proj



# Introdução

- Dependência Funcional Transitiva

- O atributo C é dependente funcional transitivo (DFT) de A se C é funcionalmente dependente de A, na mesma relação
- A DFT constitui uma anomalia que deve ser evitada



# Introdução

- Dependência Funcional Transitiva
  - Solução: criação de uma nova relação que será composta pelo atributo ou atributos que são dependentes funcionais transitivos tendo como chave primária o atributo que determina a transitividade

EMPREGADO			
#Num-emp	Nom-emp	Data-adm-emp	Cod-proj_emp

PROJETO	
# Cod-Proj_emp	Data-term-proj

# Introdução

- ▶ A normalização é importante para identificar um bom projeto relacional.
- ▶ Um bom MER e sua consequente conversão para um MR, praticamente deixa o esquema relacional *normalizado*.
- ▶ Utiliza-se a normalização somente para *validar* um projeto relacional.

## Redundância de dados

Identidade	Nome	Endereço	Habilidade
8795835	Édson Arantes	Ponta da Praia	Futebol
8795835	Édson Arantes	Ponta da Praia	Voleibol
8795835	Édson Arantes	Ponta da Praia	Basquete
8795835	Édson Arantes	Ponta da Praia	Atletismo
8795835	Édson Arantes	Ponta da Praia	Tênis

Se Pelé mudar de endereço ?

Um novo esporte para Pelé ?

*Problemas com atualização*

*Problemas com inclusão*

# Idealmente

Identidade	Nome	Endereço	Habilidade
8795835	Édson Arantes	Ponta Praia	da {Futebol,

Voleibol,  
Basquete,  
Atletismo,  
Tênis}

Mas isto não é uma tabela  
(atributo *habilidade* não é  
atômico)!

<b>Identidade</b>	<b>Nome</b>	<b>Endereço</b>
8795835	Édson	Ponta da
...	Arantes	Praia

<b>Identidade</b>	<b>Esporte</b>
8795835	Futebol
8795835	Voleibol
8795835	Basquetebol
8795835	Atletismo
8795835	Tênis

**A repetição da coluna Identidade é uma redundância necessária**

<b>Identidade</b>	<b>Nome</b>	<b>Endereço</b>	<b>Habilidade</b>
8795835	Édson Arantes	Ponta da Praia	Futebol
8795835	Édson Arantes	Ponta da Praia	Voleibol
8795835	Édson Arantes	Ponta da Praia	Basquete
8795835	Édson Arantes	Ponta da Praia	Atletismo
8795835	Édson Arantes	Ponta da Praia	Tênis

# Introdução

- ▶ O processo de normalização como foi proposto inicialmente por Codd conduz um esquema de relação através de uma bateria de testes para certificar se o mesmo está na 1ª, 2ª e 3ª Formas Normais.
- ▶ Essas três Formas Normais são baseadas em dependências funcionais dos atributos do esquema de relação.



# Primeira Forma Normal (1FN)

- Toda tabela deve ser “minimamente” normalizada (1FN).

*primeira forma normal (1FN)*

=

diz-se que uma tabela está na primeira forma normal,  
quando ela não contém tabelas aninhadas

# Primeira Forma Normal (1FN)

- Exemplo: **Cliente**

1-Código

2- {Telefone}

3- Endereço (Rua, Número, Cidade)

Gerando a tabela resultante:

<b>Cliente</b>	<u>Código</u>	Telefone 1	Endereço		
		Telefone <i>n</i>	Rua	Número	Cidade

Sendo que a mesma não está na 1FN pois seus atributos não são atômicos. Para que a tabela fique na 1FN temos que eliminar os atributos não atômicos, gerando as seguintes tabelas como resultado:

<b>Cliente</b>	<u>Código</u>	Rua	Número	Cidade
----------------	---------------	-----	--------	--------

<b>Cliente_Telefone</b>	<u>Código Cliente</u>	<u>Telefone Cliente</u>
-------------------------	-----------------------	-------------------------

<b>Matrícula</b>	<b>Nome</b>	<b>Cod Cargo</b>	<b>NomeCargo</b>	<b>CodProj</b>	<b>DataFim</b>	<b>Horas</b>
120	João	1	Programador	01	17/07/95	37
120	João	1	Programador	08	12/01/96	12
121	Hélio	1	Programador	01	17/07/95	45
121	Hélio	1	Programador	08	12/01/96	21
121	Hélio	1	Programador	12	21/03/96	107
270	Gabriel	2	Analista	08	12/01/96	10
270	Gabriel	2	Analista	12	21/03/96	38
273	Silva	3	Projetista	01	17/07/95	22
274	Abraão	2	Analista	12	21/03/96	31
279	Carla	1	Programador	01	17/07/96	27
279	Carla	1	Programador	08	12/01/96	20
279	Carla	1	Programador	12	21/03/96	51
301	Ana	1	Programador	12	21/03/96	16
306	Manoel	3	Projetista	17	21/03/96	67

Redundância de dados

Problemas de inserção,  
remoção e atualização

- Uma tabela em 1FN continua com problemas de redundância de dados, inclusão, atualização, e remoção.
- É necessário a aplicação de outras formas normais:
  - 2FN
  - 3FN

# Segunda Forma Normal 2FN

*segunda forma normal (2FN)*

=

uma tabela encontra-se na segunda forma normal,  
quando, além de estar na 1FN, não contém  
dependências parciais

*dependência parcial*

=

uma dependência (funcional) parcial ocorre quando  
uma coluna depende apenas de parte de uma chave  
primária composta

E #	Enome	Sexo	Idade	D #	Dnome	Opinião
E 1	João	M	25	D 1	Mat	Boa
E 1	João	M	25	D 2	Quim	Má
E 1	João	M	25	D 3	Fis	Boa
E 2	Maria	F	22	D 2	Quim	Satisf.
E 2	Maria	F	22	D 3	Fis	Satisf.
E 2	Maria	F	22	D 4	Est	Má
E 3	João	M	27	D 2	Quim	Boa
E 3	João	M	27	D3	Fis	Boa

Chaves Candidatas	Complementos da Chave
{E#, D#}	Enome, Sexo, Idade, Dnome, Opinião

<b>E #</b>	<b>Enome</b>	<b>Sexo</b>	<b>Idade</b>
E1	João	M	25
E2	Maria	F	22
E3	João	M	27

<b>D #</b>	<b>Dnome</b>
D1	Mat
D2	Quim
D3	Fis
D4	Est

<b>E #</b>	<b>D #</b>	<b>Opinião</b>
E1	D1	Boa
E1	D2	Pobre
E1	D3	Boa
E2	D2	Satisfatória
E2	D3	Satisfatória
E2	D4	Pobre
E3	D2	Boa
E3	D3	Boa

# Segunda Forma Normal 2FN

- Uma tabela T está na 2FN se estiver na 1FN e todos os seus atributos não chaves forem totalmente funcionalmente dependente da chave primária C.
- Se uma tabela não está na 2FN a mesma pode ser normalizada gerando outras tabelas cujos atributos que não façam parte da chave primária sejam totalmente funcionalmente dependente da mesma, ficando a tabela na 2FN.



# Terceira Forma Normal 3FN

*terceira forma normal (3FN)*

=

uma tabela encontra-se na terceira forma normal,  
quando, além de estar na 2FN, não contém  
dependências transitivas

*dependência transitiva*

=

uma dependência funcional transitiva ocorre quando  
uma coluna, além de depender da chave primária da  
tabela, depende de outra coluna ou conjunto de  
colunas da tabela

<b>Matr</b>	<b>Nome</b>	<b>codCargo</b>	<b>nomeCargo</b>	<b>codPro</b>	<b>DataFim</b>	<b>horas</b>
120	João	1	Programador	01	17/07/95	37
120	João	1	Programador	08	12/01/96	12
121	Hélio	1	Programador	01	17/07/95	45
121	Hélio	1	Programador	08	12/01/96	21
121	Hélio	1	Programador	12	21/03/96	107
270	Gabriel	2	Analista	08	12/01/96	10
270	Gabriel	2	Analista	12	21/03/96	38
273	Silva	3	Projetista	01	17/07/95	22
274	Abraão	2	Analista	12	21/03/96	31
279	Carla	1	Programador	01	17/07/96	27
279	Carla	1	Programador	08	12/01/96	20
279	Carla	1	Programador	12	21/03/96	51
301	Ana	1	Programador	12	21/03/96	16

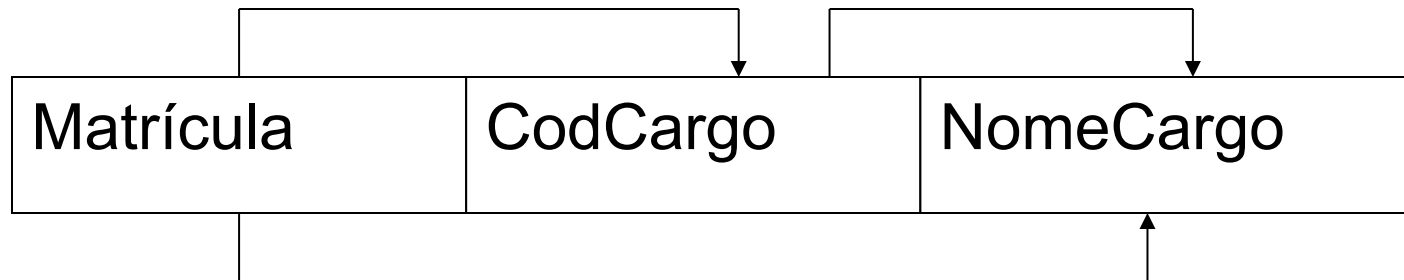
## 2FN

Matr	Nome	codCargo	nomeCargo
120	João	1	Programador
121	Hélio	1	Programador
270	Gabriel	2	Analista
273	Silva	3	Projetista
274	Abraão	2	Analista
279	Carla	1	Programador
301	Ana	1	Programador
306	Manuel	3	Projetista

codProj	dataFim
01	17/07/95
08	12/01/96
12	21/03/96

Matr	codPro	horas
120	01	37
120	08	12
121	01	45
121	08	21
121	12	107
270	08	10
270	12	78
273	01	22
274	12	31
279	01	27
279	08	20
279	12	51
301	01	16
301	12	85
306	12	67

Problemas de inclusão, remoção e atualização



NomeCargo é dependente transitivo de Matrícula.

Matr	Nome	codCargo
120	João	1
121	Hélio	1
270	Gabriel	2
273	Silva	3
274	Abraão	2
279	Carla	1
301	Ana	1

codCargo	nome
1	Programador
2	Analista
3	Projetista

- Exemplo 2:

cod_proj	tipo	descr	emp					
			cod_emp	nome	cat	sal	data_ini	temp
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	Manu- tenção	Sistema de 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

1FN

```
proj_emp([cod_proj], tipo, descr, [cod_emp],
          nome, cat, sal, data_ini, temp)
```

Decomposição de tabela:

1FN

```
proj([cod_proj], tipo, descr)
proj_emp([cod_proj, cod_emp], nome, cat, sal,
          data_ini, temp)
```

- Exemplo 2:

Proj (CodProj, Tipo, Descr)

ProjEmp (CodProj, CodEmp, Nome, Cat, Sal, DataIni, TempAl) } **1FN**

proj

cod_proj	tipo	descr
LSC001	Novo Desenv.	Sistema
PAG02	Manutenção	Sistema de RH

proj\_emp

cod_proj	cod_emp	nome	cat	sal	data_ini	temp
LSC001	2146	João	A1	4	1/11/91	24
LSC001	3145	Sílvio	A2	4	2/10/91	24
LSC001	6126	José	B1	9	3/10/92	18
LSC001	1214	Carlos	A2	4	4/10/92	18
LSC001	8191	Mário	A1	4	1/11/92	12
PAG02	8191	Mário	A1	4	1/05/93	12
PAG02	4112	João	A2	4	4/01/91	24
PAG02	6126	José	B1	9	1/11/92	12

- Exemplo 2:

Para passar à 2FN, ou seja, eliminar as dependências de parte da chave primária é necessário dividir a tabela ProjEmp em duas tabelas como:

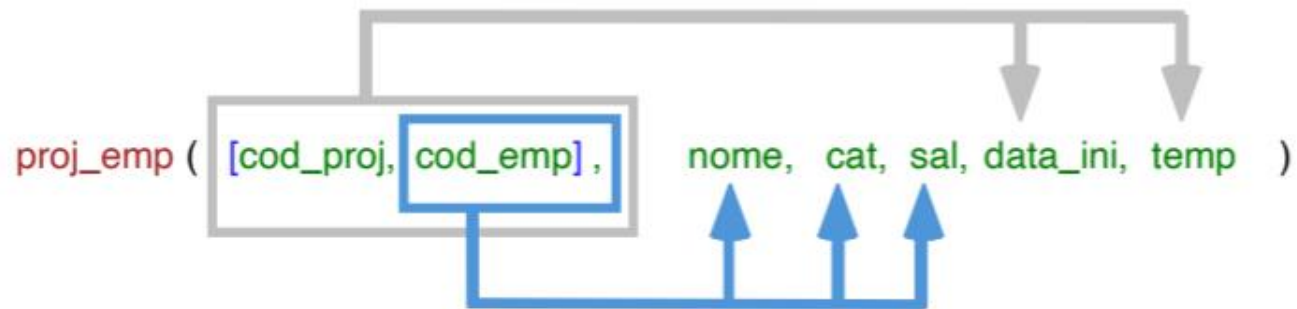
ProjEmp ( CodProj, CodEmp, DataIni, TempAl)

Emp ( CodEmp, Nome, Cat, Sal )

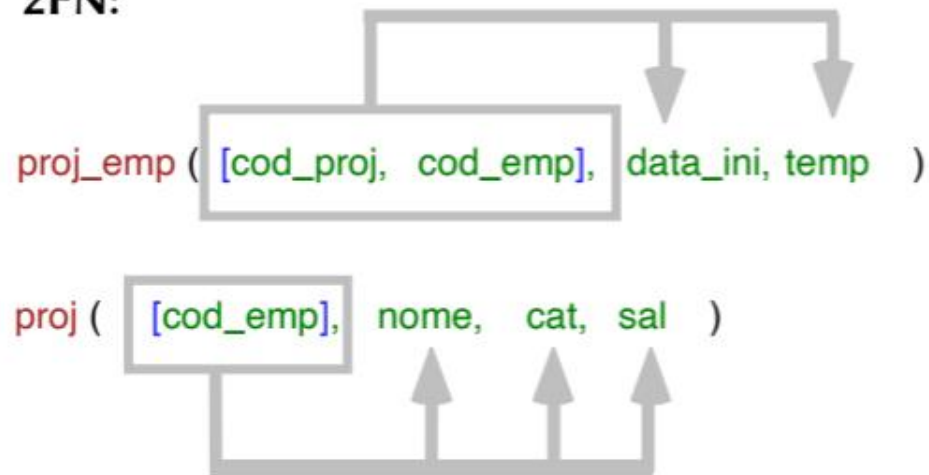


- Exemplo 2:

**1FN e dependências funcionais:**



**2FN:**



- Exemplo 2:

## 2FN

proj([cod\_proj], tipo, descr)

proj\_emp([cod\_proj, cod\_emp], data\_ini, temp)

emp([cod\_emp], nome, cat, sal)

### proj

cod_proj	tipo	descr
LSC001	Novo Desenv.	Sistema de Estoque
PAG02	Manutenção	Sistema de RH

### proj\_emp

cod_proj	cod_emp	data_ini	temp
LSC001	2146	1/11/91	24
LSC001	3145	2/10/91	24
LSC001	6126	3/10/92	18
LSC001	1214	4/10/92	18
LSC001	8191	1/11/92	12
PAG02	8191	1/05/93	12
PAG02	4112	4/01/91	24
PAG02	6126	1/11/92	12

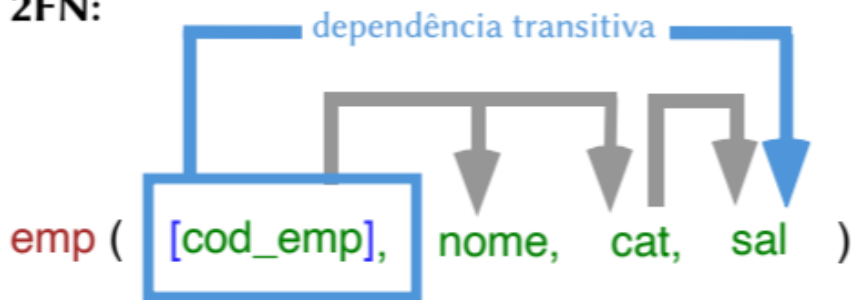
### emp

cod_emp	nome	cat	sal
2146	João	A1	4
3145	Sílvio	A2	4
6126	José	B1	9
1214	Carlos	A2	4
8191	Mário	A1	4
4112	João	A2	4

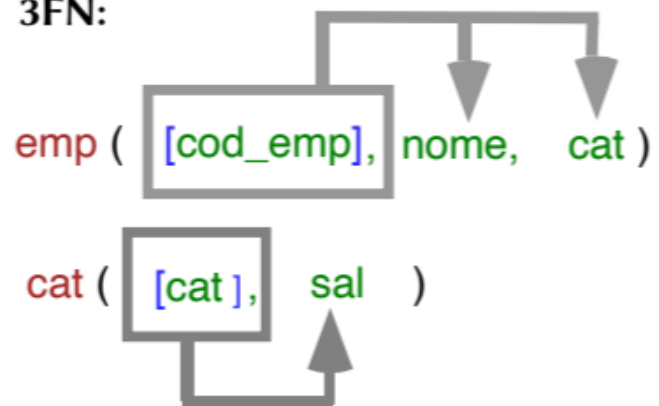
- Exemplo 2:

Uma tabela encontra-se na 3FN quando, além de estar na 2FN, toda coluna não chave depende diretamente da chave primária, ie, quando não há dependência funcional transitiva ou indiretas.

2FN:



3FN:



- Exemplo 2:

**3FN**

```
proj([cod_proj], tipo, descr)
proj_emp([cod_proj, cod_emp], data_ini, temp)
emp([cod_emp], nome, cat)
cat([cat], sal)
```

proj

cod_proj	tipo	descr
LSC001	Novo Desenv.	Sistema de Estoque
PAG02	Manutenção	Sistema de RH

cat

cat	sal
A1	4
A2	4
B1	9

proj\_emp

cod_proj	cod_emp	data_ini	temp
LSC001	2146	1/11/91	24
LSC001	3145	2/10/91	24
LSC001	6126	3/10/92	18
LSC001	1214	4/10/92	18
LSC001	8191	1/11/92	12
PAG02	8191	1/05/93	12
PAG02	4112	4/01/91	24
PAG02	6126	1/11/92	12

emp

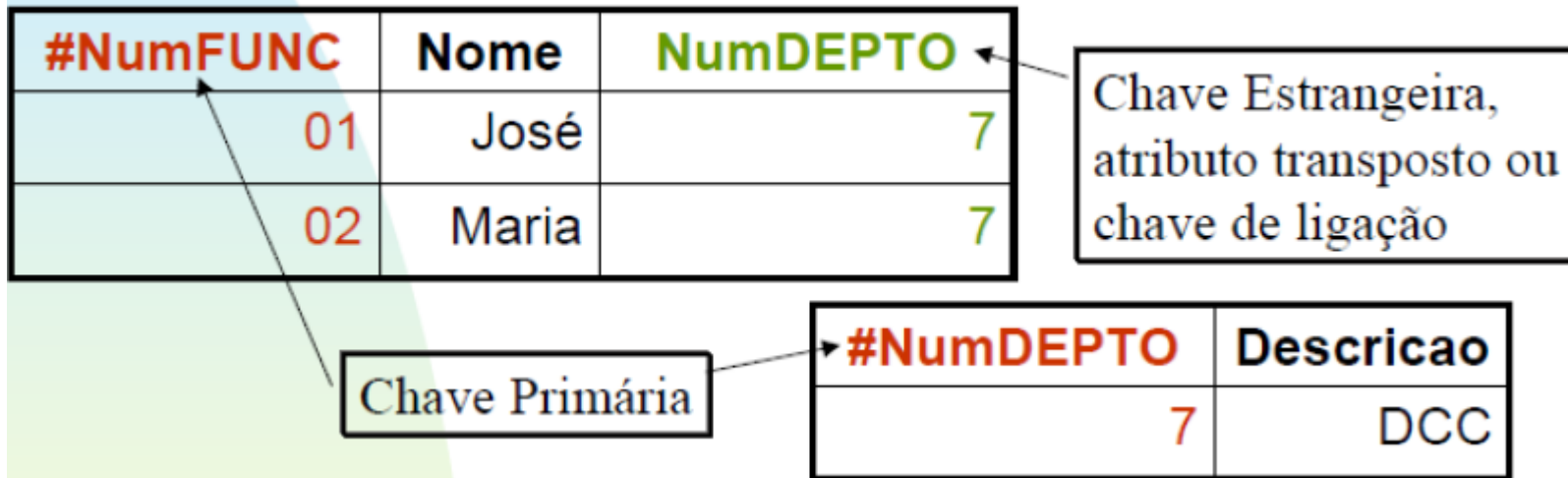
cod_emp	nome	cat
2146	João	A1
3145	Sílvio	A2
6126	José	B1
1214	Carlos	A2
8191	Mário	A1
4112	João	A2

# Aspecto de Integridade

- ▶ Restrição de Chave
- ▶ Uma relação deve ter pelo menos uma chave.
- ▶ Definição de chave: uma chave é um atributo ou conjunto de atributos cujo valor ou combinação de valores deve ser distinto em qualquer instância da relação.
- ▶ A existência de pelo menos uma chave é uma condição obrigatória, tendo em vista que uma relação é definida como um conjunto de tuplas e, todas as tuplas devem ter um valor distinto.

# Aspecto de Integridade

- Restrição de Entidade
- Especifica que nenhum valor de chave primária pode ser nulo.
- Uma tupla em uma tabela que se refere a uma outra relação deve referenciar uma tupla



# Aspecto de Integridade

- Outros Tipos de Restrições
- As restrições verificadas tratam somente das restrições estruturais.
- Existem as restrições de integridade semântica ou regras de negócio.
- São implementadas por meio de **gatilhos** ou **stored procedures** (procedimentos armazenados).
- Exemplo:

o salário de uma atriz não pode ser maior que  $\frac{1}{3}$  do orçamento total do filme.

# Restrições de Integridade no Modelo Relacional

- Integridade Lógica
  - Conjunto de regras que existem para o modelo de dados, assim como um conjunto de regras de negócio, que regem a manipulação do BD, de forma a não ferir nenhuma destas regras estabelecidas.
- Integridade Física
  - Conjunto de procedimentos operacionais que garantem a integridade do BD, mesmo em situações de falha de algum componente do ambiente onde o BD é manipulado.
  - Exemplo:
    - Valor mínimo de depósito para abertura de uma conta R\$10.000,00
    - Conta corrente sem movimento há 180 dias será encerrada.



# Especificação de Banco de Dados Relacionais

- A especificação de um banco de dados relacional (chamada de esquema do banco de dados) deve conter no mínimo a definição do seguinte:
  - Tabelas que formam o banco de dados
  - Colunas que as tabelas possuem
  - Restrições de integridade

# Fixando

## ► Formas Normais

- 1ª Forma Normal (1FN): toda relação deve ter uma chave primária e deve-se garantir que todo atributo seja atômico. Atributos compostos devem ser separados.
- 2ª Forma Normal (2FN): toda relação deve estar na 1FN e devem-se eliminar dependências funcionais parciais, ou seja, todo atributo não chave deve ser totalmente dependente da chave primária.
- 3ª Forma Normal (3FN): Toda relação deve estar na 2FN e devem-se eliminar dependências funcionais transitivas, ou seja, todo atributo não chave deve ser mutuamente independente.

# Referência

- Livros:
  - ELMASRI, R., NAVATHE, S. B., Sistemas de Banco de Dados, 7 ed., 2018, Editora Pearson.
  - HEUSER, C. A. , Projeto de banco de Dados. 7 ed. 2024. Porto Alegre.