



iscte

INSTITUTO  
UNIVERSITÁRIO  
DE LISBOA



emprego  
digital

Módulo 5: Bases de Dados

## Aula 3

# Introdução ao Modelo Relacional



Já aprendemos a **desenhar** uma base de dados.

E agora?

# O que é o Modelo Relacional?

- O modelo relacional é uma forma de estruturar a informação através de tabelas.
- A conversão de UML para o modelo relacional baseia-se na aplicação de um conjunto de regras.
- **Ao transpor para o modelo relacional, teremos as tabelas da nossa base de dados, prontas a implementar!**

Mas antes da transposição, abordemos um outro conceito essencial: **chaves**.

# Atributos Identificadores: Chaves

## Chave Primária Primary Key

- É o identificador único (ID)
- Não podem existir dois objetos com o mesmo ID
- Útil para referenciar um objeto de forma inequívoca.
- Preenchimento obrigatório

## Chave Estrangeira Foreign Key

- Ocorre quando a chave primária de uma tabela é guardada em outra tabela
- Deriva das relações entre as classes (proximos slides)
- Permite guardar relações entre dados

# Chave Primária

- Todas as tabelas têm de possuir um mecanismo de identificação (chave primária).
- A chave primária permite identificar uma linha de uma tabela.

Qual dos atributos pode ser uma **chave primária**?

Tabela Utilizador

	EmailUtilizador	NomeUtilizador	PaisResidencia	UtilizadorID
►	luis@iscte.pt	Luís Novas	Portugal	1
	ana@iscte.pt	Ana Patrício	Portugal	2
	carlos@iscte.pt	Carlos Monteiro	Brasil	3
	isabel@iscte.pt	Isabel Fonseca	Angola	4
	monica@iscte.pt	Mónica Martins	Portugal	5
	pedro@iscte.pt	Pedro Cortêz	Portugal	6
	paulo@iscte.pt	Paulo Vicente	Portugal	7
	vera@iscte.pt	Vera Nogueira	Brasil	8
	filipa@iscte.pt	Filipa Andrade	Portugal	9
	jose@iscte.pt	José Raimundo	Portugal	10

# Critérios para Chave Primária

## Obrigatório:

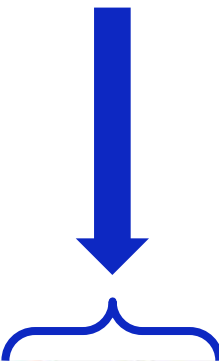
- Unicidade
- Preenchimento obrigatório

## Boas práticas:

- Não sobrecarregar o sistema com muitas chaves primárias.
- Usar um id numérico é sempre uma boa opção.



# Chave Primária Composta



	fila	lugar	ocupado
▶	1	1	0
	1	2	1
	1	3	1
	2	1	0
	2	2	0
	2	3	0

# Chave Estrangeira

Tabela País

	nome
▶	Alemanha
	Brasil
	Espanha
	França
	Portugal

Tabela Utilizador

	EmailUtilizador	NomeUtilizador	PaisResidencia	UtilizadorID
▶	luis@iscte.pt	Luís Novas	Portugal	1
	ana@iscte.pt	Ana Patrício	Portugal	2
	carlos@iscte.pt	Carlos Monteiro	Brasil	3
	isabel@iscte.pt	Isabel Fonseca	Angola	4
	monica@iscte.pt	Mónica Martins	Portugal	5
	pedro@iscte.pt	Pedro Cortêz	Portugal	6
	paulo@iscte.pt	Paulo Vicente	Portugal	7
	vera@iscte.pt	Vera Nogueira	Brasil	8
	filipa@iscte.pt	Filipa Andrade	Portugal	9
	jose@iscte.pt	José Raimundo	Portugal	10

# Transposição UML -> Modelo Relacional

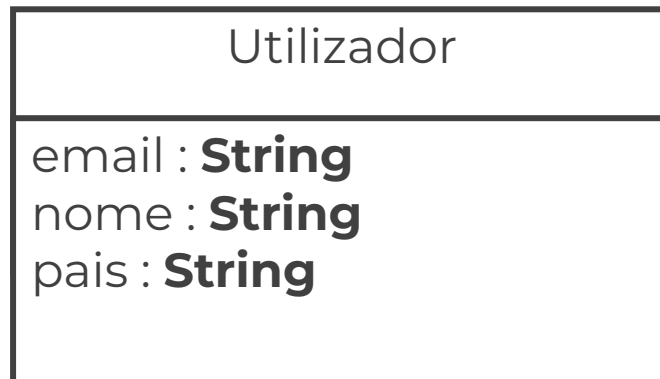
A transposição do modelo de conceptual (os Diagramas de Classes em UML) para o modelo relacional tem como objetivo **criar uma base de dados coerente com a modelação feita anteriormente.**

Esta conversão obedece a um conjunto de regras que garantem que:

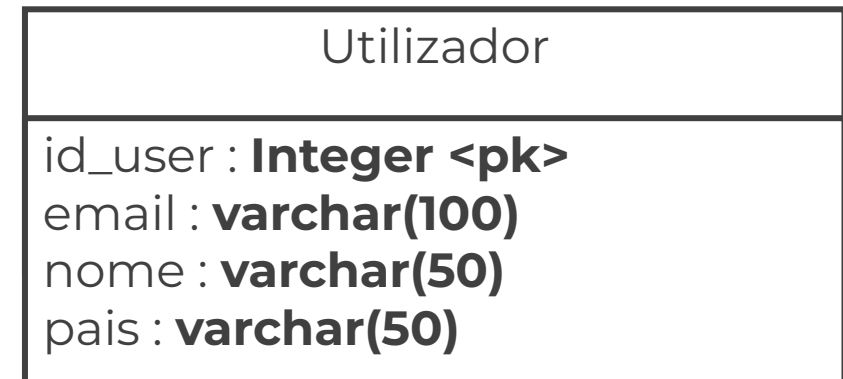
- Não ocorre perda de informação, i.e., é possível aceder a toda a informação
- Não existe informação redundante

# Exemplo de Transposição

Diagrama de Classes UML



Modelo Relacional



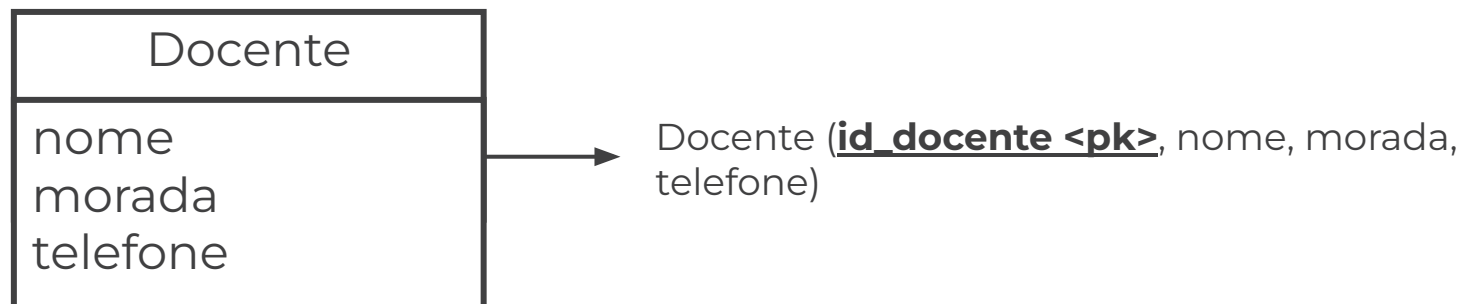
# As regras não devem ser “leis”...

Devemos olhar para as regras como uma indicação suscetível de adaptação em função da análise do problema em questão. As regras podem gerar modelos relacionais ineficientes.

Na transposição entre os modelos existe perda de informação semântica relativa às relações entre as classes: a partir do modelo relacional **pode não ser possível** obter o Diagrama de Classes a partir do qual ele foi gerado.

# Regras de Transposição

- **Todas as tabelas deverão ter uma chave primária.** No caso de não existirem atributos que satisfaçam esta condição dever-se-á criar um identificador único usualmente designado por *id*.
- As tabelas geradas são resultado exclusivamente das classes do modelo, e das associações de “muitos para muitos”;
- Todos os atributos de uma classe são atributos da tabela que implementa a classe, inclusive os atributos das classes associativas.

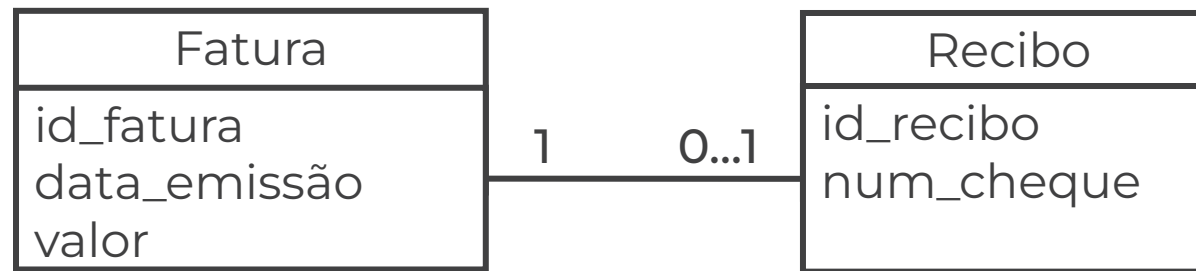


# Regras de Transposição “um/um”

## Transposição de Relações de Um para Um

- Uma das tabelas da relação deverá possuir uma chave estrangeira referenciando a outra tabela.
- A determinação da tabela que herdará a chave estrangeira fica ao critério do analista e da interpretação que faz da realidade, devendo optar pelo que fizer mais sentido.

# Regras de Transposição “um/um”



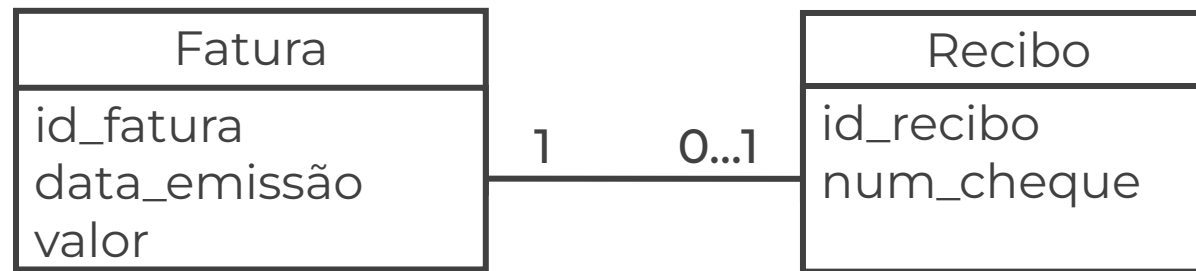
Fatura (id\_fatura <pk>, data\_emissão, valor)

Recibo (id\_recibo <pk>, num\_cheque, **id\_fatura** <fk>)

**Alternativa:** tabela fatura poderia guardar o Id Recibo como chave estrangeira.



# Regras de Transposição “um/um”



**Tabela Fatura**

<u>id_fatura</u>	data_emissao	valor
014	12-12-2020	1500
013	27-12-2019	2400

**Tabela Recibo**

<u>id_recibo</u>	n_cheque	id_fatura
05	Null	014
01	Null	013

Chave Estrangeira

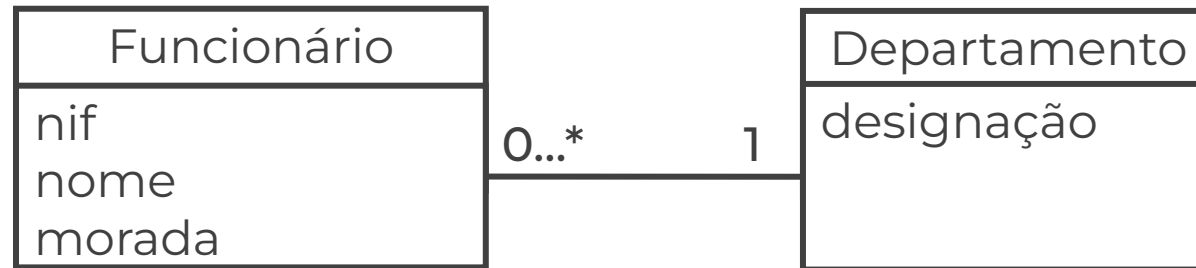


# Regras de Transposição “**um/muitos**”

## **Transposição de Relações de Um para Muitos**

- A tabela cujos registos são suscetíveis de serem endereçados diversas vezes (lado *muitos*) herda a chave da tabela cuja correspondência é unitária (lado *um*).

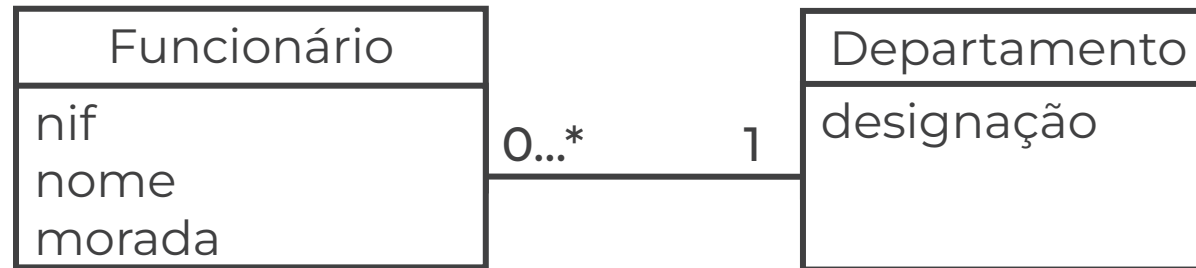
# Regras de Transposição “um/muitos”



Funcionário (nif <pk>, nome, morada, **id\_departamento** <fk>)

Departamento (id\_departamento <pk>, designação)

# Regras de Transposição “um/muitos”



**Funcionário**

<u>nif</u>	nome	morada	id_departamento
100000000	Carlos	Null	2
100000001	Rita	Null	1

**Departamento**

<u>id_departamento</u>	<u>designação</u>
1	dep. financeiro
2	dep. informático

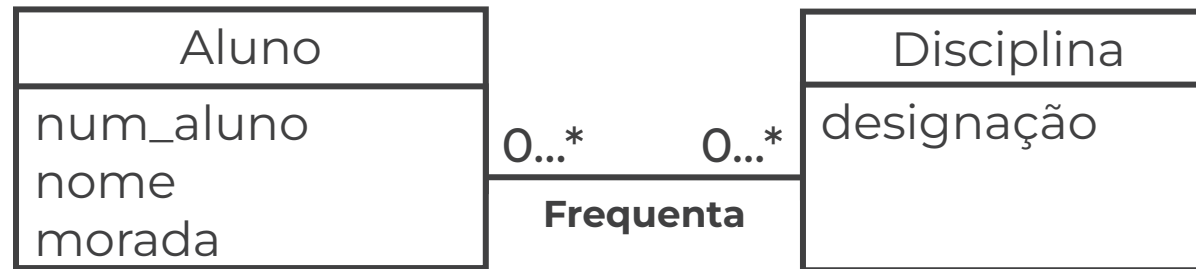
Chave Estrangeira

# Regras de Transposição “**muitos/muitos**”

## Transposição de Relações de Muitos para Muitos

- A relação muitos para muitos dá origem a uma tabela representativa da associação onde a chave primária é composta pelas chaves primárias das tabelas que implementam as classes associadas.
- **Esta é a única multiplicidade de relação que gera uma tabela extra!**

# Regras de Transposição “muitos/muitos”



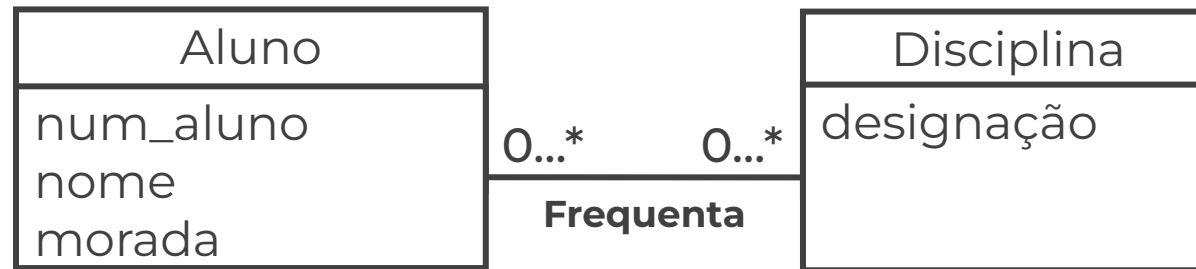
Aluno (num\_aluno <pk>, nome, morada)

Disciplina (id\_disciplina <pk>, designação)

Frequenta (num\_aluno <pk> <fk>, id\_disciplina <pk> <fk>)

**Nota:** o nome da relação é o nome da tabela extra. Embora não seja obrigatório, **é uma boa prática**.

# Regras de Transposição “muitos/muitos”



**Aluno**

<u>num_aluno</u>	nome	morada
1123	Cátia	Null
1542	João	Null

**Frequência**

<u>num_aluno</u>	<u>id_disciplina</u>
1123	2
1542	1

**Disciplina**

<u>id_disciplina</u>	<u>Designação</u>
1	Sistemas Operativos
2	Base de Dados

Chave Estrangeira

Chave Estrangeira

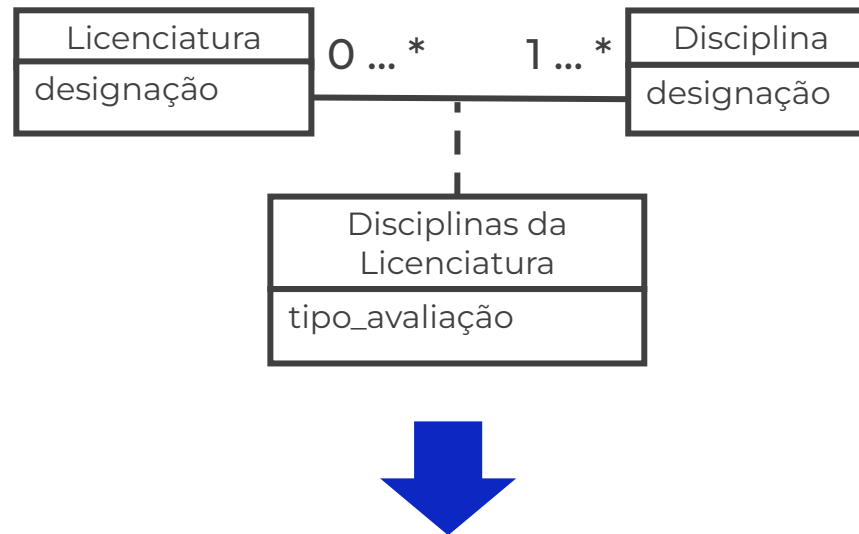
# Regras de Transposição **Classes Associativas**

## **Transposição de Classes Associativas**

- Para as Classes Associativas aplicam-se as regras correspondentes à associação. Os atributos da classe associativa são herdados pela(s) tabela(s) que herda(m) a(s) chave(s) estrangeira(s)



# Regras de Transposição **Classes Associativas**

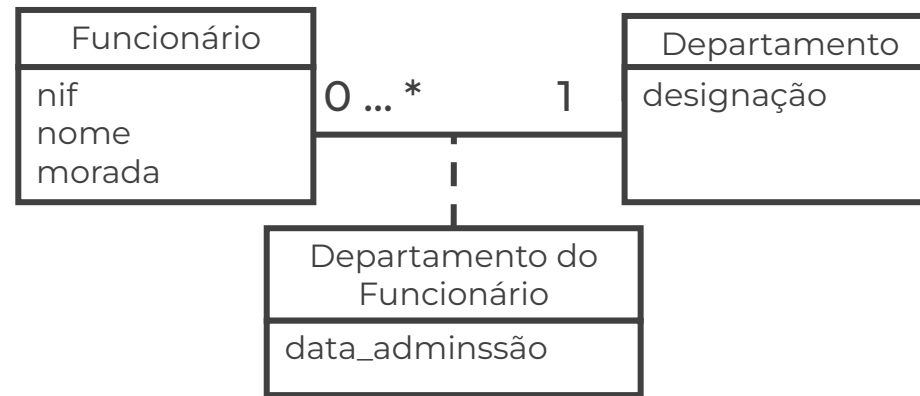


Licenciatura (id\_licenciatura <pk>, designação)

Disciplina (id\_disciplina <pk>, designação)

Disciplinas\_da\_Licenciatura (id\_licenciatura <pk>  
<fk>, id\_disciplina <pk> <fk>, tipo\_avaliaoao)

# Regras de Transposição **Classes Associativas**



Departamento (id\_departamento <pk>, designação)

Funcionário (nif <pk>, nome, morada, **id\_departamento** <fk>, **data\_admissão**)

# Regras de Transposição **Generalização**

## Transposição de Relações de Generalizações

Podem ocorrer duas situações:

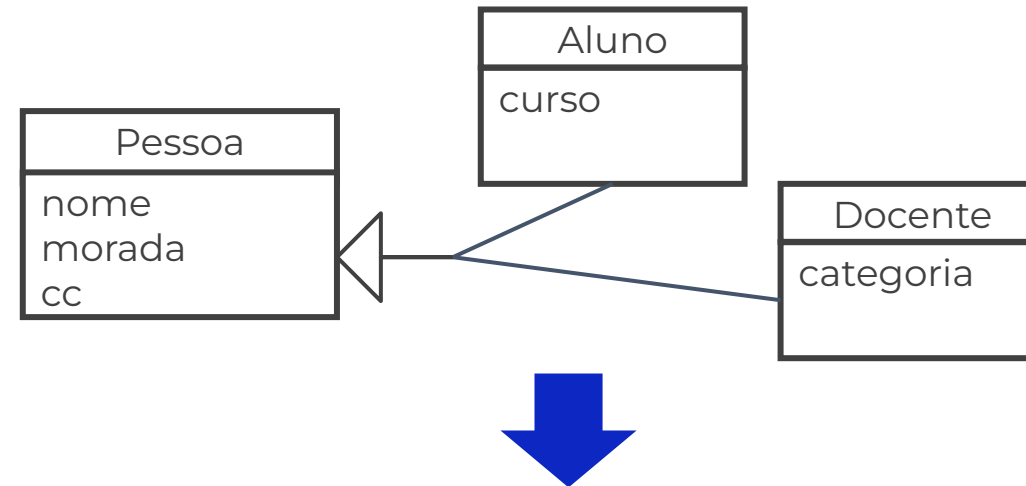
**A. As subclasses têm identidade própria (são independentes da classe *pai*).**

- A chave primária da tabela que implementa a classe pai é obtida através dos atributos da própria classe.
- As chaves primárias das tabelas que implementam as classes filha são obtidas através dos atributos das próprias classes.
- As tabelas que implementam as classes filhas herdarão a chave primária da tabela pai, como chave estrangeira.

# Regras de Transposição **Generalização**

Transposição de Relações de Generalizações

## **Situação A**



Pessoa (cc <pk>, nome, morada)

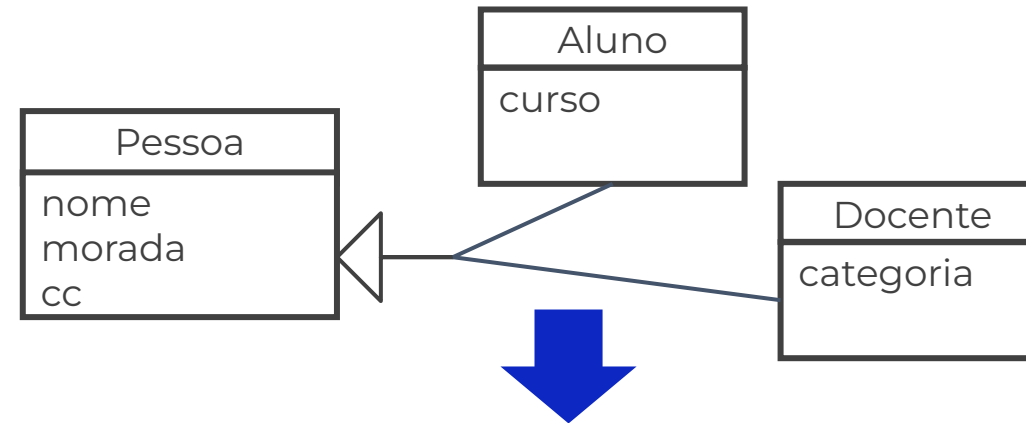
Aluno (id\_aluno <pk>, curso, **cc** <fk>)

Docente (num\_docente <pk>, categoria, **cc** <fk>)

# Regras de Transposição **Generalização**

Transposição de Relações de Generalizações

## Situação A



Aluno			Docente			Pessoa		
<u>Número</u>	Curso	CC	<u>Número</u>	Categoria	CC	<u>CC</u>	Nome	Morada
1123	Java	1234567	10	Associado	7654213	1234567	Clara	Null
						7654213	Américo	Null

Chave Estrangeira

prego digital

# Regras de Transposição **Generalização**

## **Transposição de Relações de Generalizações**

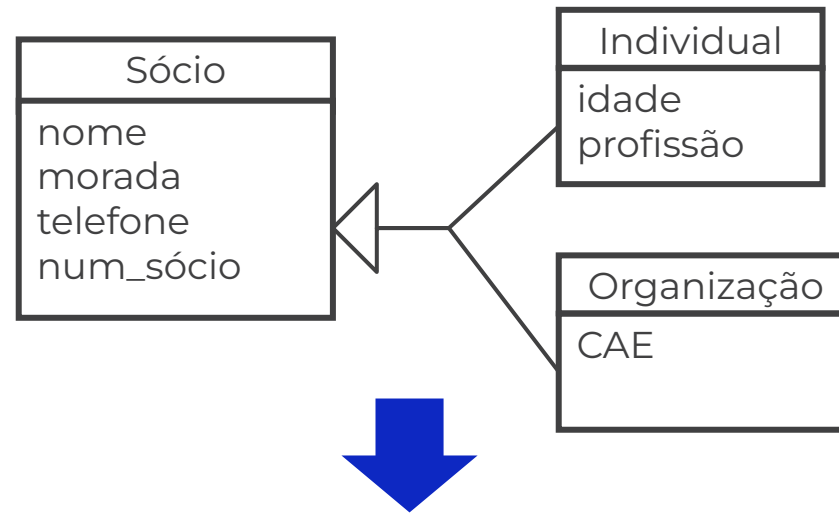
Podem ocorrer duas situações:

- B.** As classes filha **só têm** identidade **enquanto associadas** à classe pai.
  - A chave primária da tabela que implementa a classe pai é obtida através dos atributos da própria classe.
  - As tabelas que implementam as classes filhas herdarão a chave primária da tabela pai, como chave primária (e estrangeira).

# Regras de Transposição **Generalização**

Transposição de Relações de Generalizações

## **Situação B**



Sócio (num\_sócio <pk>, nome, morada, telefone)

Individual (num\_sócio <pk> <fk>, idade, profissão)

Organização (num\_sócio <pk> <fk>, CAE)

# Regras de Transposição **Generalização**

Transposição de Relações de Generalizações

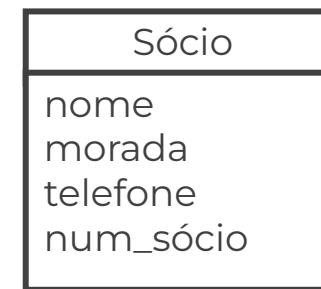
## Situação B

### Individual

<u>num_sócio</u>	idade	profissão
213	29	Programador

### Organização

<u>num_sócio</u>	CAE
10	A.99.5857



Chave Estrangeira

### Sócio

<u>num_sócio</u>	nome	morada	telefone
213	Maria	Null	9100000000
10	Pedro	Null	9100000000



# Regras de Transposição **Generalização**

## Resumindo:

- Se a classe filha **tem** identidade própria então **deve ter um ID próprio**.
- Se a classe filha **não tem** identidade própria, a sua chave primária **deve ser herdada da superclasse**.

# Regras de Transposição **Agregação**

## Transposição de Relações de Agregação

**Simples: a transposição para o relacional obedece às regras de transposição das associações com a mesma multiplicidade.**

# Regras de Transposição **Composição**

## **Transposição de Relações de Composição**

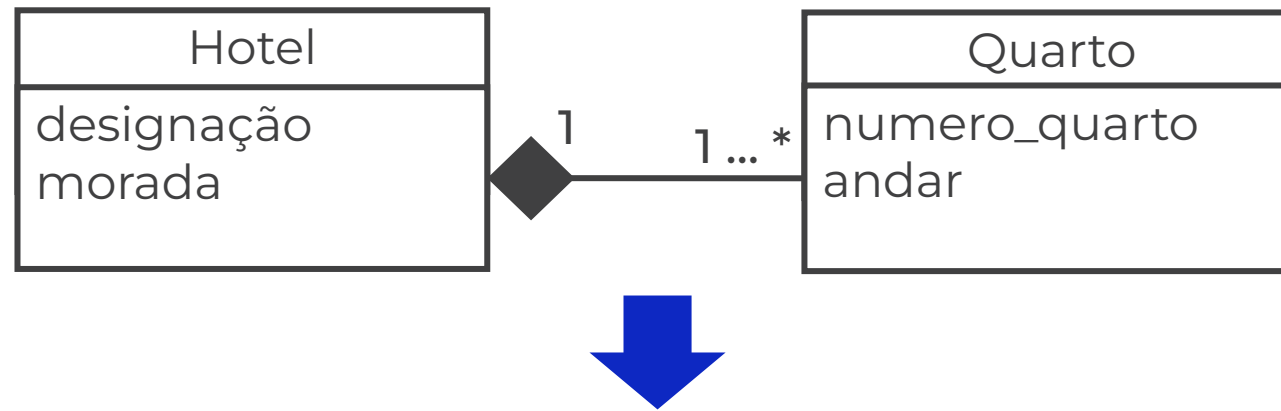
A tabela que implementa a classe composição tem uma chave primária composta pelos seguintes atributos:

1. A chave da tabela correspondente à classe que representa a composição
2. Um atributo (ou mais) pertencente à classe componente

Neste caso temos uma **chave primária composta**.

# Regras de Transposição **Composição**

## Transposição de Relações de Composição

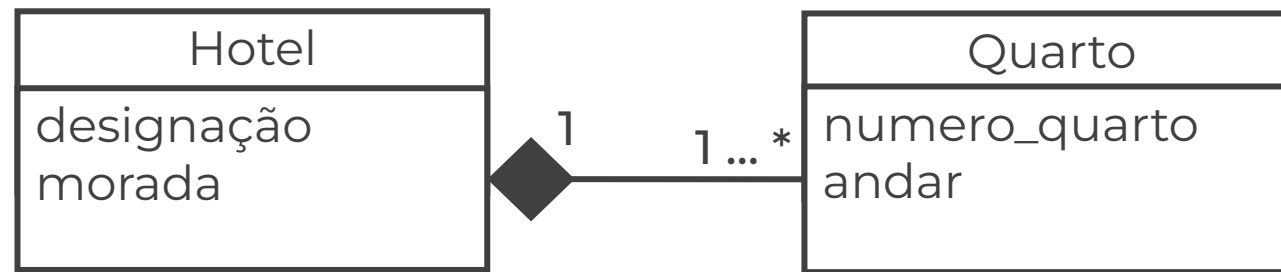


Hotel (id\_hotel <pk>, designação, morada)

Quarto (id\_hotel <fk> <pk>, numero\_quarto <pk>, andar)

# Regras de Transposição **Composição**

## Transposição de Relações de Composição



**Hotel**

<u>id_hotel</u>	designação	Morada
1	ABC	Rua X, Lisboa
2	AAA	Rua Y, Porto

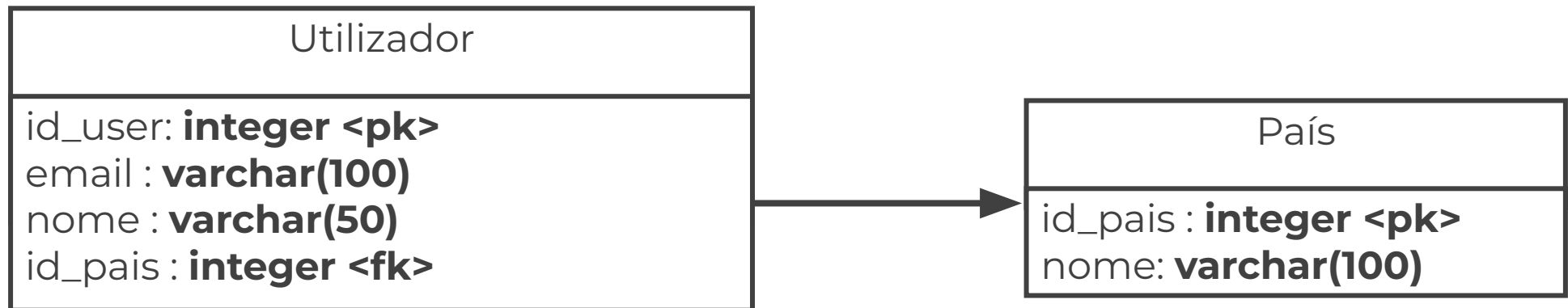
**Quarto**

<u>id_hotel</u>	<u>numero_quarto</u>	andar
1	1	2
2	24	3

Chave Estrangeira



# Representação Modelo Relacional



A seta aponta no sentido da origem da chave estrangeira.

# Especificação de Atributos

Na especificação dos atributos, para além da sua designação e tipo de dados, é possível indicar outras propriedades:

- Chave primária
- Chave estrangeira
- “Allow Nulls” - admite o valor *null*
- Unique - não admite valores duplicados (para criar chaves alternativas)
- Validações (CHECK) - regras simples com operadores lógicos (<, >, <>, or, and)
- Valor por omissão
- Comentários

# Otimizações do Modelo Relacional

Apesar de assegurarem um modelo completo (sem perda de informação) e coerente, as regras de transposição **podem gerar modelos ineficientes**. Sempre que possível deve-se otimizar o modelo relacional obtido, nomeadamente, no que diz respeito a:

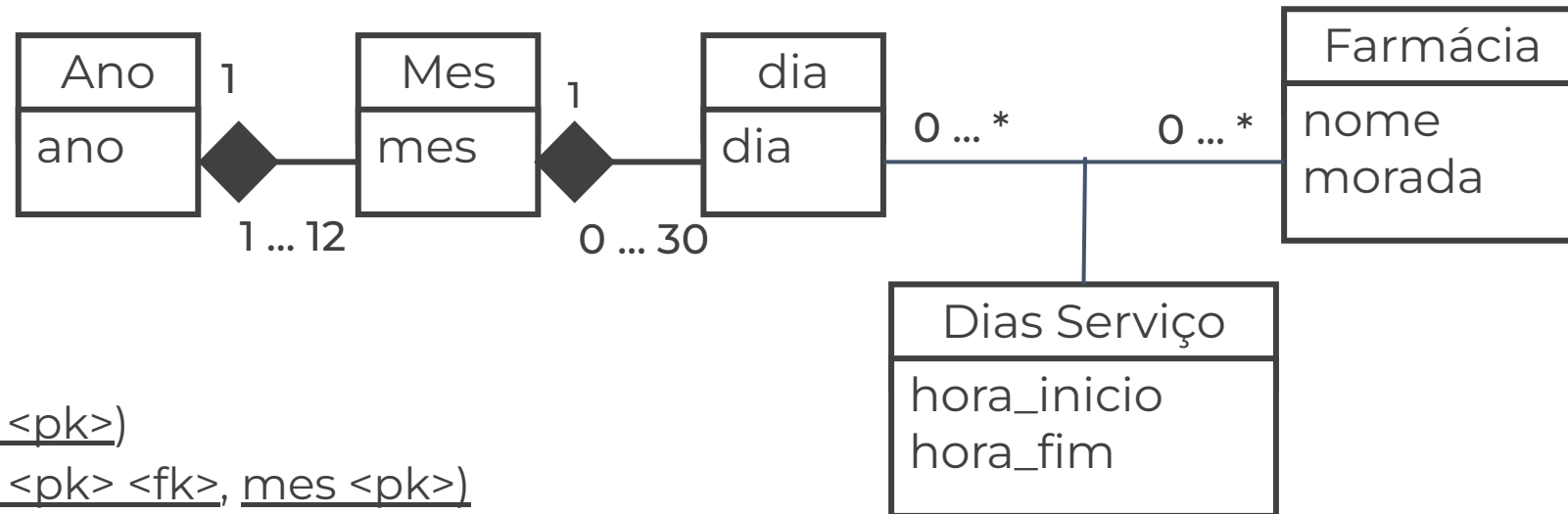
- Números de tabelas - um número elevado de tabelas pode comprometer a eficiência do modelo.
- Número de atributos que compõem a chave das tabelas - deve ser reduzido.

**Ao contrário de um diagrama de classes, o modelo relacional não pretende ser descritivo, mas sim eficaz e eficiente.**



# Exemplo de Otimização 1

## Modelo Original



Ano (ano <pk>)

Mes (ano <pk> <fk>, mes <pk>)

Dia (ano <pk> <fk>, mes <pk> <fk>, dia <pk>)

Farmácia (id\_farmacia <pk>, nome, morada)

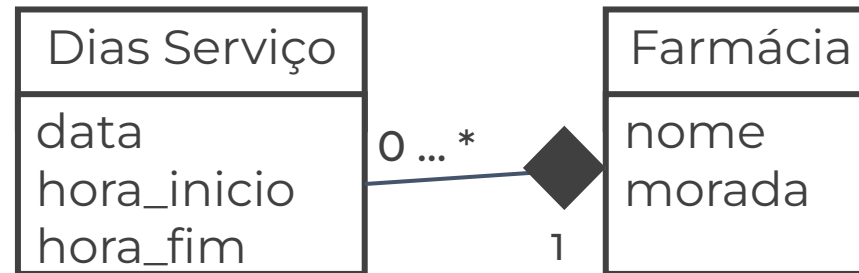
Dias\_Serviço (id\_farmacia <pk> <fk>, ano <pk> <fk>, mes <pk> <fk>, dia <pk> <fk>, hora\_inicio, hora\_fim)

# Exemplo de Otimização 1

**Qual a utilidade** das tabelas  
Ano, Mês e Dia?

# Exemplo de Otimização 1

## Modelo Optimizado



Farmácia (id\_farmacia <pk>, nome, morada)

Dias\_Serviço (id\_farmacia <pk> <fk>, data <pk>, hora\_inicio, hora\_fim)

# Exemplo de Otimização 2

Pessoa (cc <pk>, nome, morada)

Aluno (num\_aluno <pk>, cc <fk>)

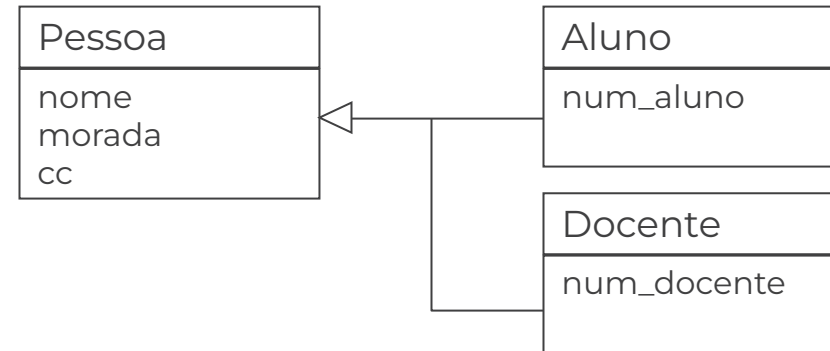
Docente (num\_docente <pk>, cc <fk>)



**Otimização**

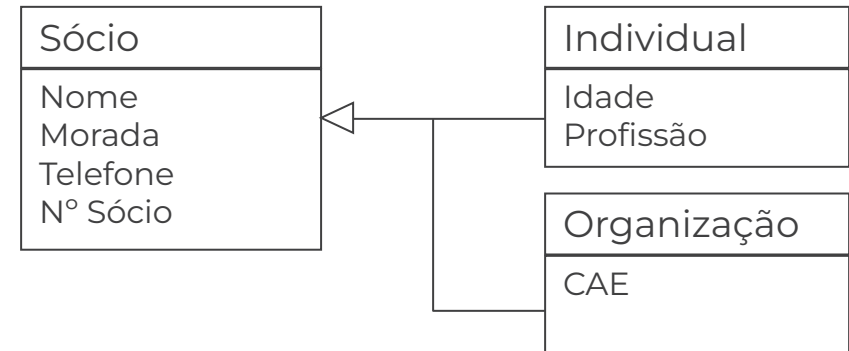
Aluno (num\_aluno <pk>, cc, nome, morada)

Docente (num\_docente <pk>, cc, nome, morada)



# Exemplo de Otimização 3

Sócio (num\_socio <pk>, nome, morada, telefone)  
Individual (num\_socio <pk> <fk>, idade, profissao)  
Organização (num\_socio <pk> <fk>, CAE)



**Otimização**

Socio (num\_socio <pk>, nome, morada, telefone, idade, profissao, CAE, Tipo{Individual, Organizacao})

Por falar em **otimização**...

# Índices

- Uma técnica habitual de optimização de consultas (*querys*) a bases de dados consiste na utilização de índices.
- O objectivo é acelerar o acesso a uma tabela através de um (ou vários) campos.
- Os índices são criados colocando uma referência no código SQL. O seu uso não tem implicações no modelo conceptual nem no modelo relacional.

# Exemplo de um Índice

- Hipótese: é frequente as publicações serem consultadas pelo assunto. Então **criemos um índice no campo assunto**.

**Ficheiro de Índice**

Assunto	Índice
Direito	
História	
Informática	
Sociologia	
Psicologia	

**Tabela de Publicação**

ISBM	Título	Data	Assunto
1	A	1998	História
2	D	1997	Informática
3	C	1994	Sociologia
4	Z	2000	Informática
5	F	1986	Direito

**Ficheiro ordenado pela primeira coluna**



# Utilidades dos Índices

## Exemplos de consultas à BD que beneficiam do índice (para BD do slide anterior)

- Quais os títulos das publicações de História? (mesmo que a maioria das publicações sejam de história, é mais rápido percorrer sequencialmente um ficheiro mais pequeno)
- Quantas publicações de informática existem (apenas necessita de abrir o ficheiro de índices)
- (se houver um índice no campo data) Quais os títulos entre 1990 e 1998? (a ordenação do índice acelera muito a procura por intervalos).

# Índices Compostos

- Podem ser criados índices para vários campos em conjunto.

Neste caso, foi criado um índice para o campo conjunto **Assunto e Data**

**Ficheiro de Índice**

Assunto/Data	Índice
<b>Direito+1986</b>	
<b>Informática+1997</b>	

**Tabela de Publicação**

ISBN	Título	Data	Assunto
1	A	1998	História
2	D	1997	Informática
3	C	1994	Sociologia
4	Z	2000	Informática
5	F	1986	Direito

# Índice para a Chave Primária

- É usual criar-se um índice para a chave primária dado que a forma privilegiada de acesso às tabelas é através desta chave.
- Pela mesma razão, também se opta por vezes por criar índices para as chaves estrangeiras.
- **Muitas bases de dados criam automaticamente um índice para a chave primária de cada tabela.**

# Cuidados a ter com Índices

- Apesar de acelerarem a consulta de informação, os índices penalizam a introdução e alteração de informação.
- A inserção de um registo obriga à introdução de um registo ordenado na tabela de índices. A alteração de um valor num atributo indexado obriga ao reordenamento do ficheiro de índices.
- É preciso um balanço cuidado. A gestão de índices é fundamental mas *perigosa*. Uma má gestão (por excesso ou defeito) pode degradar bastante o desempenho da base de dados.

# Onde guardar **constantes**?

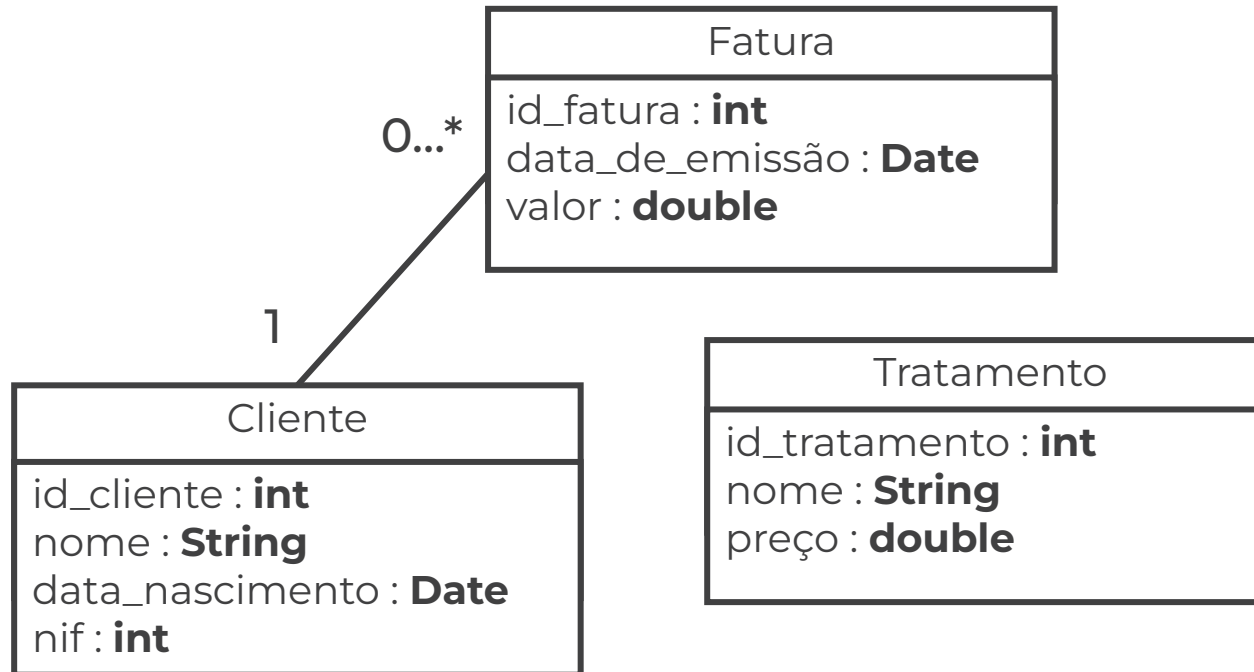
- Todas as constantes de uma aplicação devem estar numa ou mais tabelas, **e nunca no código**. Normalmente essa tabela apenas contém uma linha e uma coluna para cada parâmetro da aplicação.
- **Exemplos de constantes:**
  - Taxa IVA;
  - Designação e Morada da Empresa (para impressões);
  - Dia do mês em que são automaticamente processados os salários;

# Resumo

- Uma das tabelas da relação deverá possuir uma chave estrangeira referenciando a outra tabela.  
(**relação um/um**)
- A tabela cujos registos são suscetíveis de serem endereçados diversas vezes (lado *muitos*) herda a chave da tabela cuja correspondência é unitária (lado *um*). (**relação um/muitos**)
- A relação muitos para muitos dá origem a uma tabela representativa da associação onde a chave primária é composta pelas chaves primárias das tabelas que implementam as classes associadas.  
(**relação muitos/muitos**)
- A tabela que implementa a classe composição tem uma chave primária composta pelos seguintes atributos: (**relação composição**)
  1. A chave da tabela correspondente à classe que representa a composição
  2. Um atributo (ou mais) pertencente à classe componente

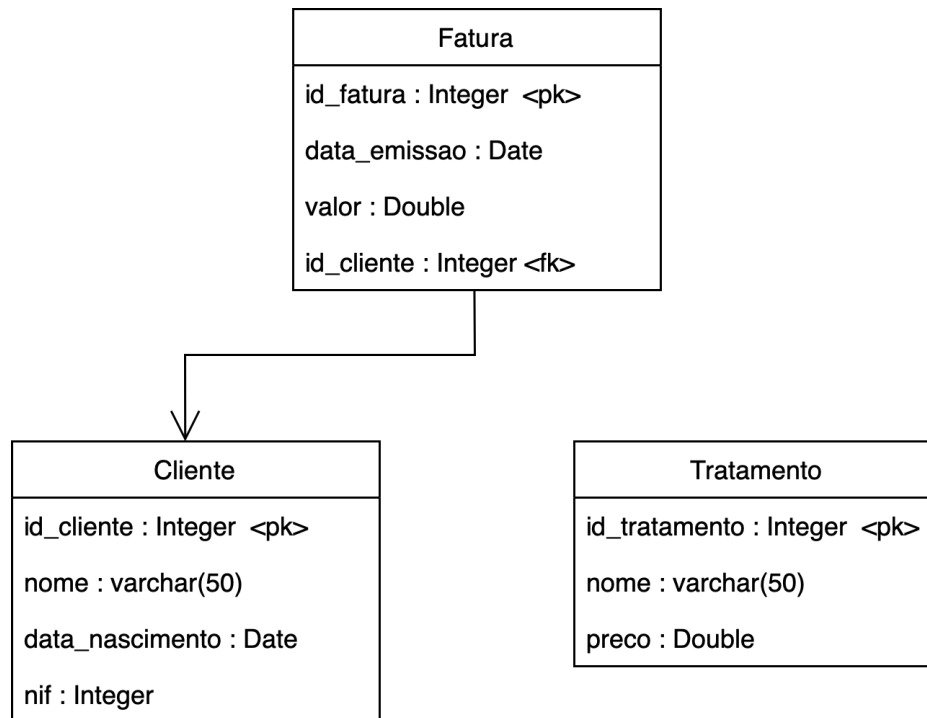
# Exercício 1

Considere o seguinte diagrama de classes:



- a) Transponha o diagrama de classes para o modelo relacional.

# Exercício 1 - Resolução



Fatura (id\_fatura <pk>, data\_de\_emissão, valor, id\_cliente)

Cliente (id\_cliente <pk>, nome, data\_nascimento, nif )

Tratamento (id\_tratamento <pk>, nome, preço)



# Exercício 2

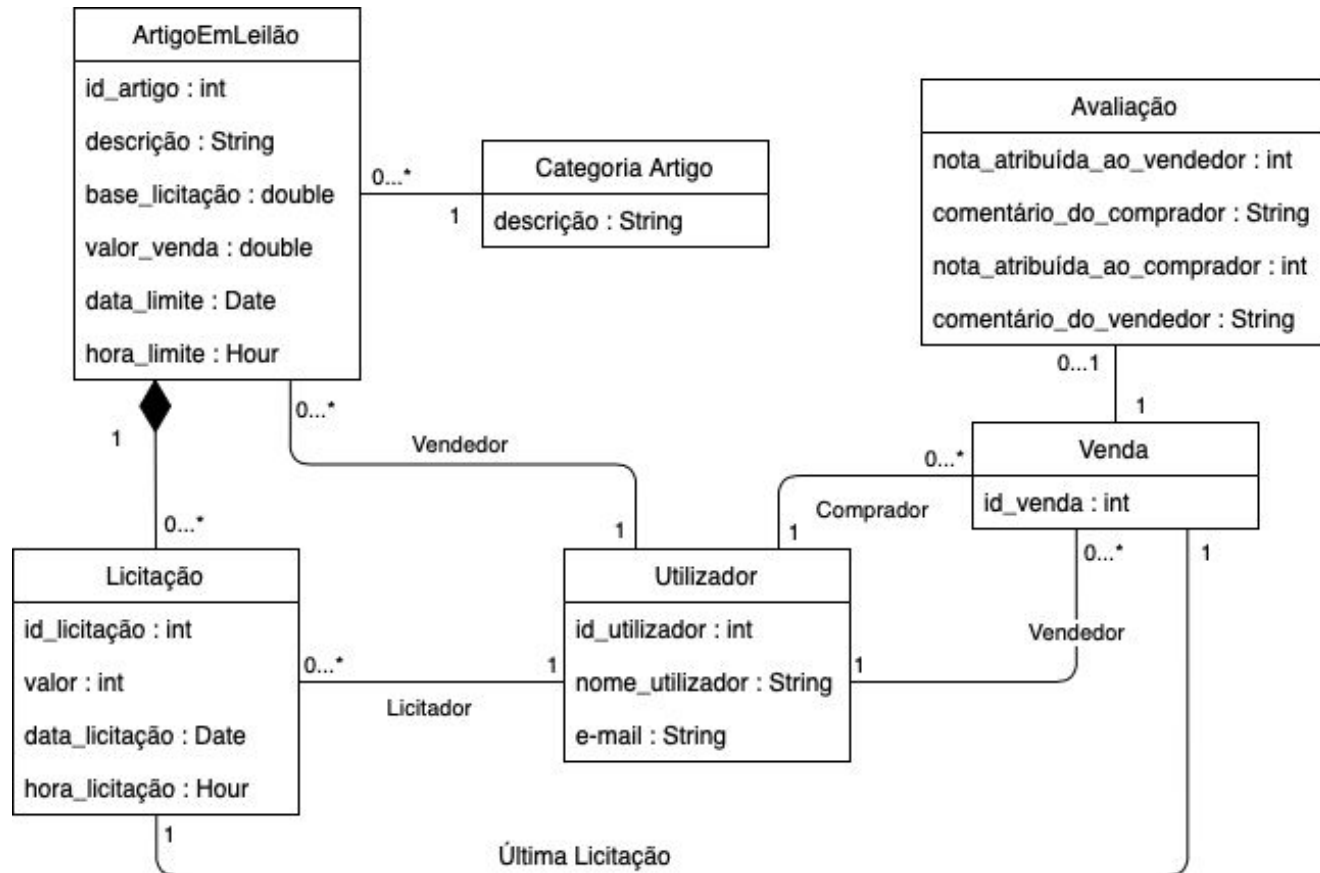
*Pretende desenvolver-se uma aplicação que disponibilize artigos através de Leilões Online. Sobre cada artigo leiloado, para além da sua descrição e categoria (Música, Mobiliário, etc), é necessário conhecer a base de licitação (valor a partir do qual são efetuadas as licitações), o seu valor de venda e a data e hora a partir da qual não são possíveis mais licitações. Sobre cada licitação, é necessário conhecer a data, hora valor lícitado e identificação do licitador.*

*Para proteger os clientes de pessoas com comportamento menos correto (por exemplo artigos danificados, vendedor que não entregou, comprador que se recusou a pagar, etc) deverá existir a possibilidade do comprador e vendedor se avaliarem mutuamente relativamente à transacção. No contexto da avaliação, deverão dar uma nota e poderão (ou não) incluir um descritivo.*

- a) Elabore o diagrama de classes da base de dados necessária para suportar o sistema descrito.
- a) Transponha o anterior diagrama de classes para o modelo relacional.

# Exercício 2 - Resolução

a)



## Exercício 2 - Resolução

b)

ArtigoEmLeilão (id\_artigo<pk>, descrição, base\_licitação, valor\_venda, data\_limite, hora\_limite, id\_categoria<fk>, id\_utilizador<fk>)

Licitação (id\_artigo<pk>, id\_licitação<pk>, valor, data\_licitação, hora\_licitação, id\_utilizador<fk>)

Categoria (id\_categoria<pk>, descrição)

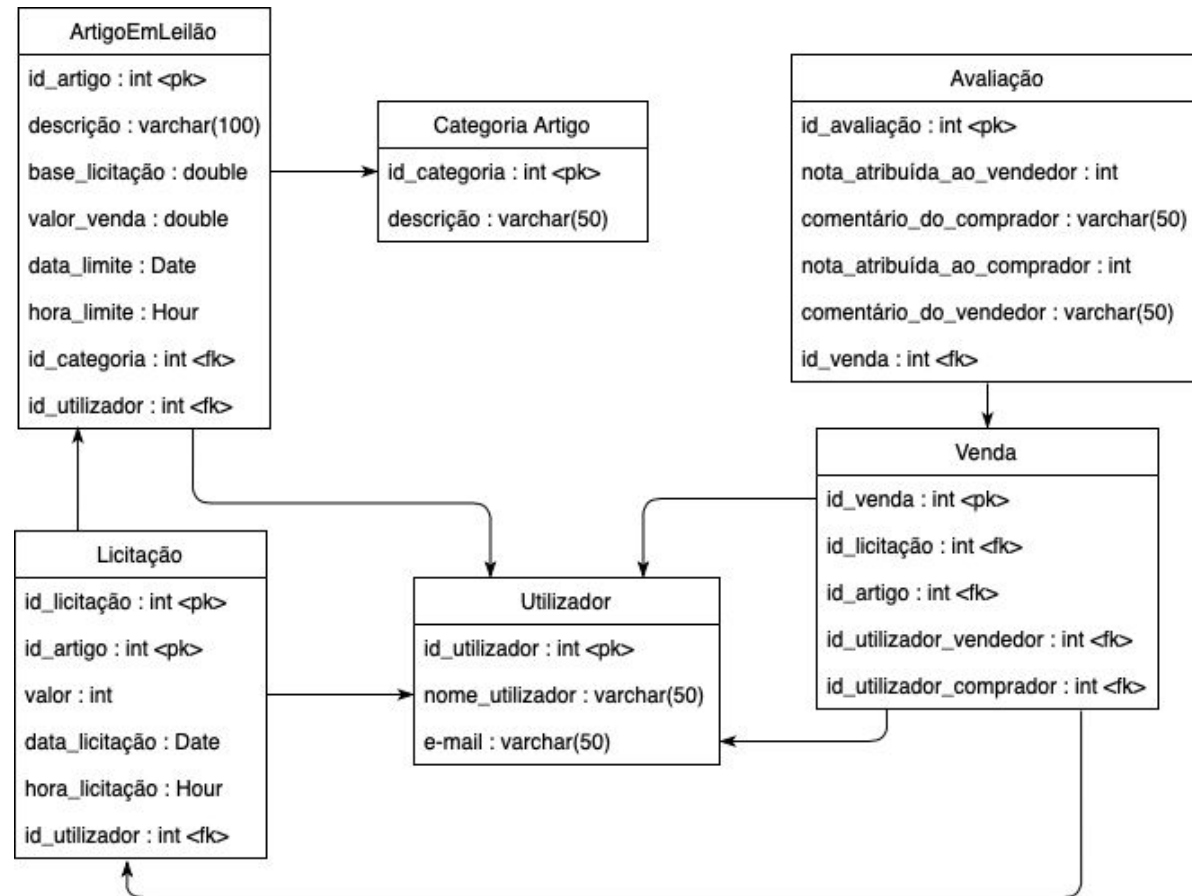
Utilizador (id\_utilizador<pk>, e-mail, nome\_utilizador)

Avaliação (id\_avaliação<pk>, nota\_atribuída\_ao\_vendedor, comentário\_do\_comprador, nota\_atribuída\_ao\_comprador, comentário\_do\_vendedor, id\_venda<fk>)

Venda (id\_venda<pk>, id\_artigo<fk>, id\_licitação<fk>, id\_utilizador\_vendedor<fk>, id\_utilizador\_comprador<fk>)

# Exercício 2 - Resolução

b)



# Exercício 3

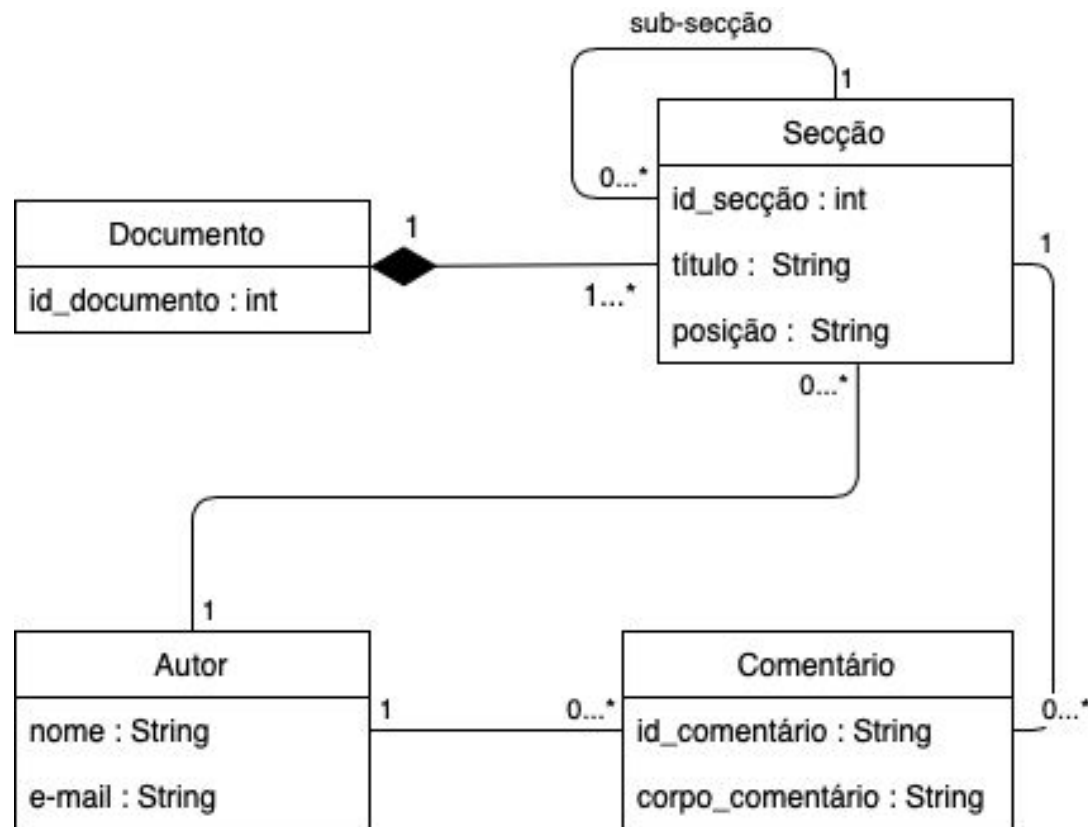
*Numa dada organização, é frequente os seus funcionários colaborarem para a elaboração e edição de documentos. A organização pretende desenvolver um sistema que armazene a estrutura dos documentos, com a indicação de quem contribuiu para a sua elaboração. Pretende-se que um documento seja decomposto em secções, que, por sua vez, podem sucessivamente ser decompostas em subsecções. Para cada secção ou subsecção, é necessário saber o seu autor (cada secção poderá ter apenas um autor). Através da aplicação deverá ser possível aos autores adicionar comentários às várias secções ou subsecções.*

*Sobre os autores é suficiente saber o seu nome e email. Sobre as secções é necessário armazenar o seu título, e o posicionamento dentro do documento.*

- a) Elabore o diagrama de classes da base de dados necessária para suportar o sistema descrito.
- a) Transponha o anterior diagrama de classes para o modelo relacional.

# Exercício 3 - Resolução

a)



# Exercício 3 - Resolução

b)

Documento (id\_documento <pk>)

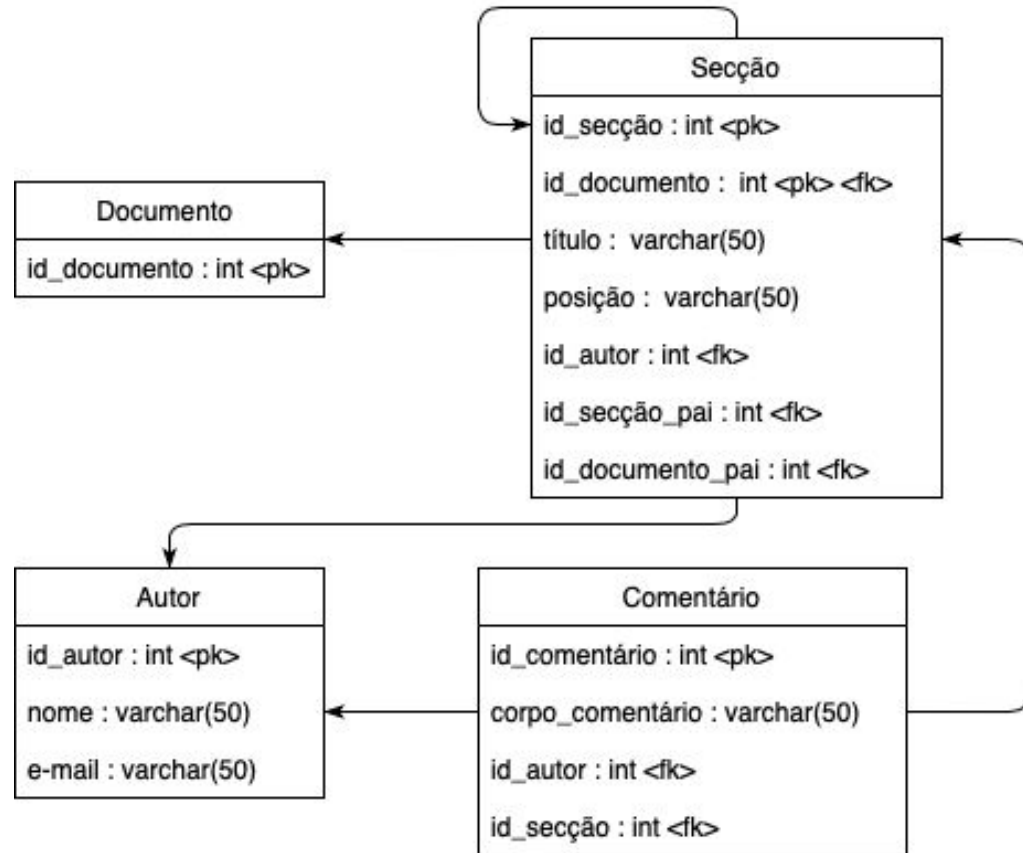
Secção (id\_secção<pk>, id\_documento <pk><fk>, título, posição, id\_autor<fk>, id\_secção\_pai <fk>, id\_documento\_pai <fk>)

Autor (id\_autor <pk>, nome, e-mail)

Comentário (id\_comentário <pk>, corpo\_comentário, id\_autor<fk>, id\_secção<fk>)

# Exercício 3 - Resolução

b)







# O futuro profissional começa aqui

iscte

INSTITUTO  
UNIVERSITÁRIO  
DE LISBOA



emprego  
digital



UPskill