



iscte

INSTITUTO
UNIVERSITÁRIO
DE LISBOA



emprego
digital

Módulo 2: Conceitos e Estrutura de Bases de Dados

Aula 1

Introdução às Bases de Dados



Sobre o módulo

O módulo de Conceitos e Estrutura de Bases de Dados pretende dotar os alunos de um conjunto fundamental de noções para que estejam aptos a desenhar e implementar uma base de dados relacional bem estruturada, eficiente e que possibilite pesquisas flexíveis, bem como efetuar consultas, inserir e atualizar dados.

Programa do módulo

Aula	Tema
1	Introdução às Bases de Dados; UML, Relações; Associações; Exercícios
2	Classes Associativas; Agregações; Composições; Generalizações; Exercícios
3	Introdução ao modelo relacional; Conversão modelo conceptual para modelo relacional; Chaves; Regras de Integridade; Otimizações e Índices; Exercícios
4	Exercícios desenho de Bases de Dados
5	Exercícios desenho de Bases de Dados
6	Teste 1 (3h) - Teste de Desenho de Bases de Dados
7	Introdução ao SQL: Queries simples (Select, Insert, Update); Exercícios
8	SQL: Funções de Agregação, Group by e Having; Subqueries; Exercícios
9	Exercícios de SQL
10	Exercícios de SQL
11	Teste 2 (3h) - Teste de SQL; Introdução ao projeto
12	Acompanhamento do projeto
13	Acompanhamento do projeto
14	Avaliação do projeto - Apresentação dos projetos

Avaliação do módulo

Teste 1 - Desenho de Bases de Dados
Classificado de A-E

Aula 6
16 / 09 / 2022

40%

Teste 2 - SQL
Classificado de A-E

Aula 11
23 / 09 / 2022

40%

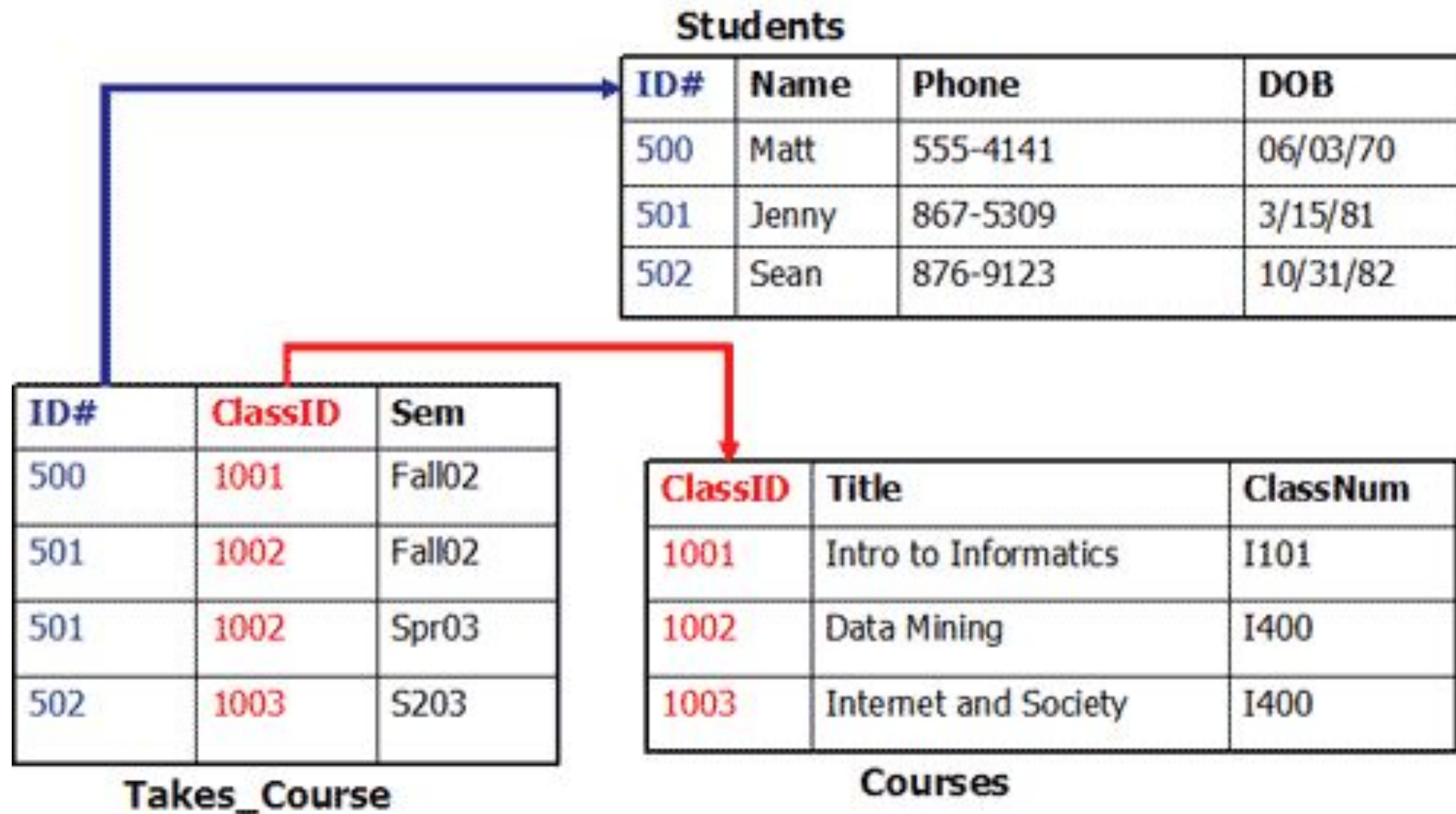
Projeto - Apresentação
Classificado de A-E

Aula 14
28 / 09 / 2022

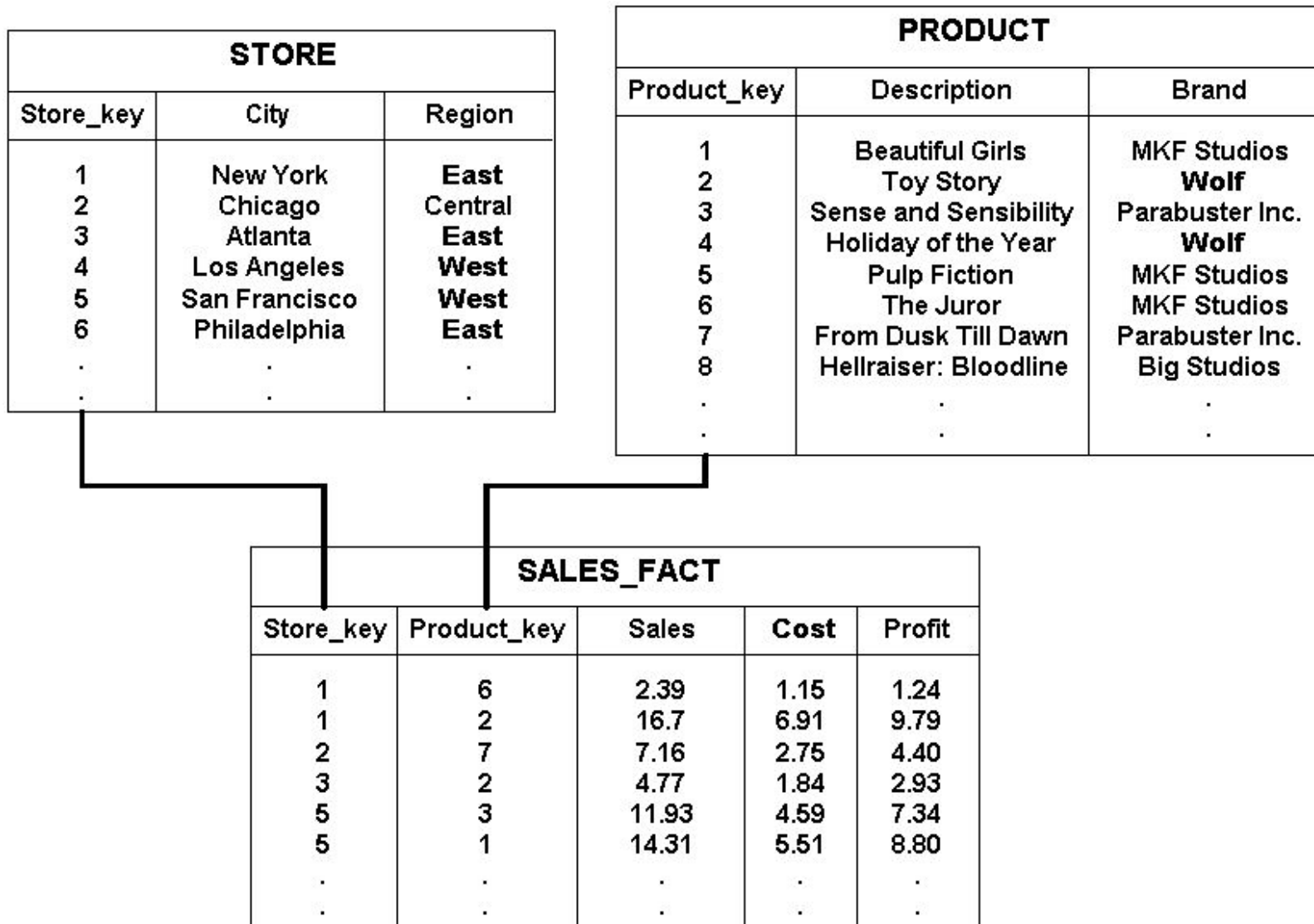
20%

O que é uma Base de Dados?

Bases de Dados são conjuntos de dados relacionados entre si com registos sobre pessoas, lugares ou coisas. São coleções organizadas de dados que se relacionam de forma a criar algum sentido (Informação) e dar mais eficiência durante uma pesquisa. São de vital importância para empresas e há mais de duas décadas que se tornaram a principal peça dos sistemas de informação e segurança. Normalmente existentes por vários anos sem alterações na sua estrutura.



Exemplo de estrutura e relações numa Base de Dados



Exemplo de estrutura e relações numa Base de Dados

Resumindo...

- Armazenamento de informação
- As Bases de Dados são constituídas por várias tabelas
- Cada tabela representa uma entidade no nosso modelo de dados
 - Ex. tabela “produtos”, “clientes” ou “registos_de_compras”
 - Cada entidade tem um conjunto de dados, representados nas colunas (um cliente pode ter uma morada, localidade e um NIF, por exemplo)
- Cada linha de uma tabela representa uma nova entrada
 - Um novo produto, um novo cliente, ou uma nova compra

Resumindo...

- Todas as entradas precisam de um ID único:
 - Forma inequívoca de identificar o registo. Muitas vezes é um número sequencial, mas também outro valor único
 - O ID do cliente pode ser um número sequencial (cliente 1, cliente 2, cliente 3) mas também pode ser o email, ou o NIF.
- Existem relações entre os dados
 - Ex. id do cliente na tabela “registo_de_compras”, para registar o cliente que efetuou a compra

Como **criar** uma Base de Dados?

Comecemos pelo seu **desenho**.

UML

Unified
Modeling
Language



UML: Unified Modeling Language

- É uma linguagem de modelação, baseada em **esquemas** e **diagramas**.
- É a linguagem padrão para a elaboração da estrutura de projetos de software, muito adequada à modelação de sistemas.
- Permite representar um sistema de forma padronizada, com o intuito de facilitar a compreensão pré-implementação.
- Poderá ser empregada para especificação, construção e documentação de projetos de software.
- **No âmbito deste módulo, será usada para a especificação do modelo de dados de uma base de dados.**

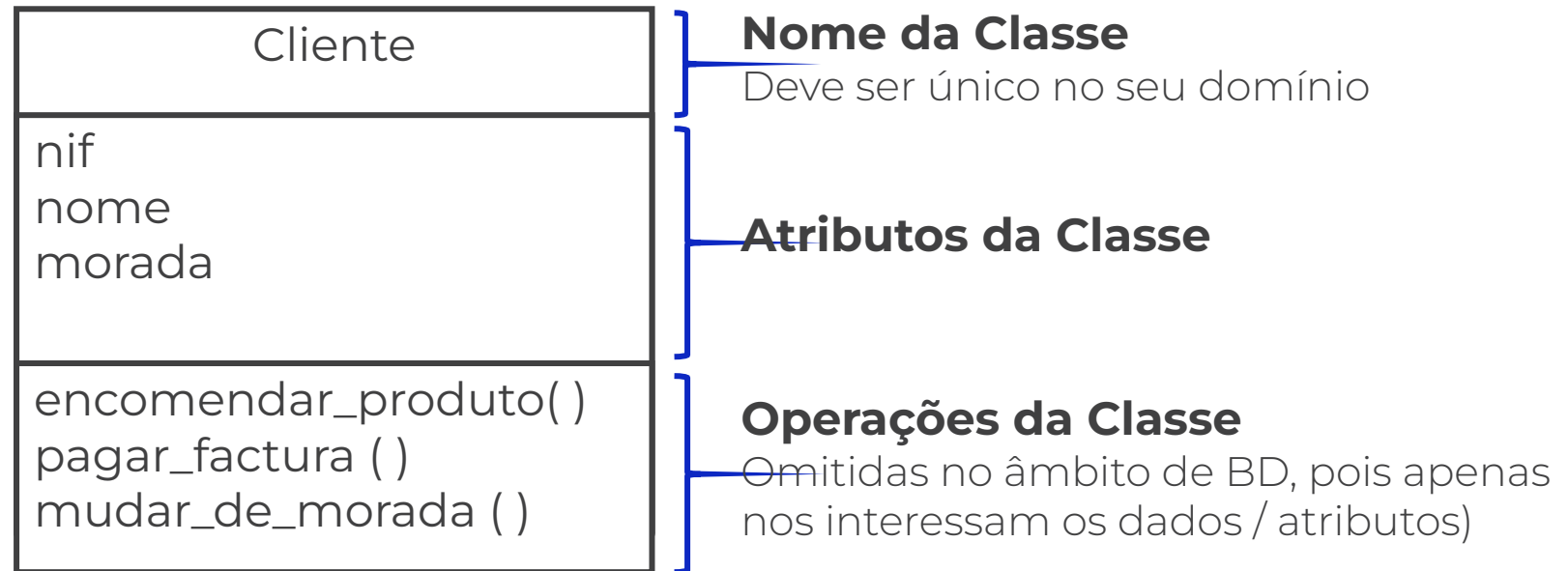
UML: Unified Modeling Language

- Esta linguagem subdivide-se em:
 - **Diagramas de Classes**
 - Diagramas de Use Case
 - Diagramas de Sequencia
 - Diagramas de Atividade
 - Diagramas de Estados
 - Diagramas de Objetos
 - Diagramas de Componentes
 - (...)
- **Para a modelação da estrutura de uma Base de Dados, é apenas necessário o recurso aos diagramas de Classes, pelo que serão os únicos abordados neste módulo.**

UML: Diagrama de Classes

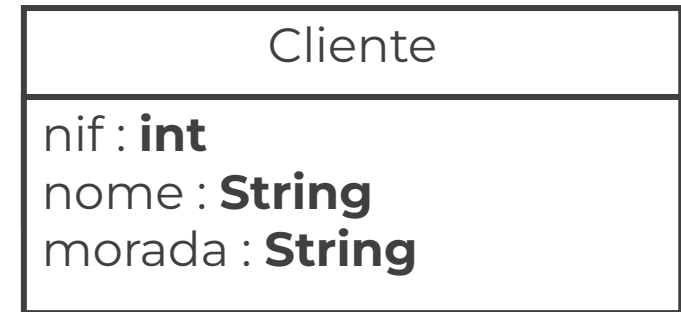
- O Diagrama de Classes é utilizado para fazer a representação da estrutura das classes de um determinado sistema.
- Em Bases de Dados, servem o propósito de representar as nossas entidades (ainda não são as tabelas, mas quase), e as suas relações.

UML: Representação de uma Classe



UML: Atributos de uma Classe

- Um **atributo** numa classe representa uma **característica comum** dos objectos dessa classe.
- Deve especificar-se um tipo de dados para cada atributo.
 - Neste caso, os valores que podem ser atribuídos ao atributo estão condicionados à compatibilidade com o tipo.



Cliente
Nr. contribuinte : Integer Nome : String Morada : String

**No fundo, cada classe
neste diagrama será uma
tabela na nossa BD.**

**E cada entrada nesta
tabela será um objeto.**

Relações entre Classes

- Em qualquer sistema existem objectos que se relacionam entre si.
 - Sistema universitário
 - Objectos: alunos, cursos, exames;
 - Os alunos relacionam-se com os cursos que frequentam e com os exames que fazem.
 - Sistema bancário
 - objectos: clientes, contas, balcões;
 - Os clientes relacionam-se com as contas que possuem e as contas relacionam-se com os balcões em que estão sediadas.

As ligações entre os objetos relacionados também são informação.

Quando há interesse em guardar na base de dados as ligações entre os objectos do sistema, especificam-se relações entre as classes desses objectos.

Tipos de Relações

Em UML existem os seguintes tipos de relações, que expressam diferentes semânticas de ligação entre objectos:

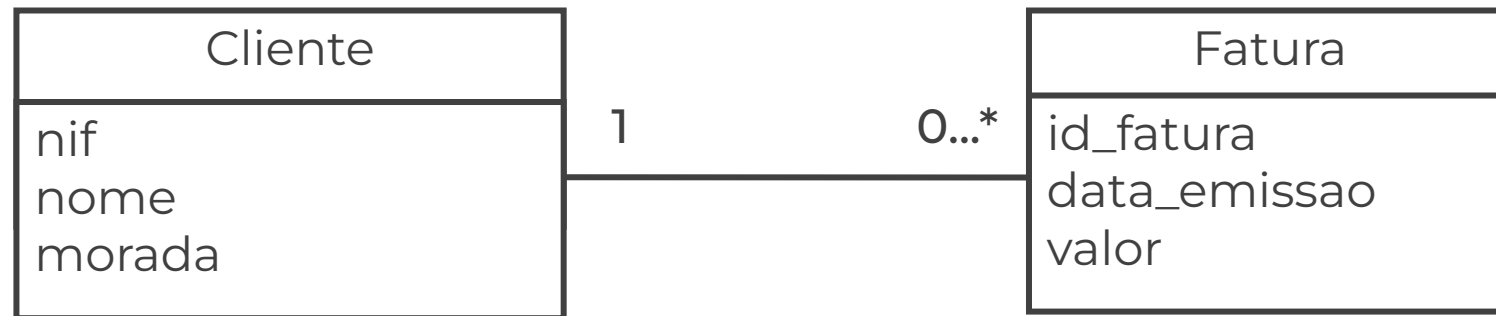
- Associação

Tem dois casos especiais:

- Agregação
 - Composição
- Generalização
- Relação de dependência

Associações

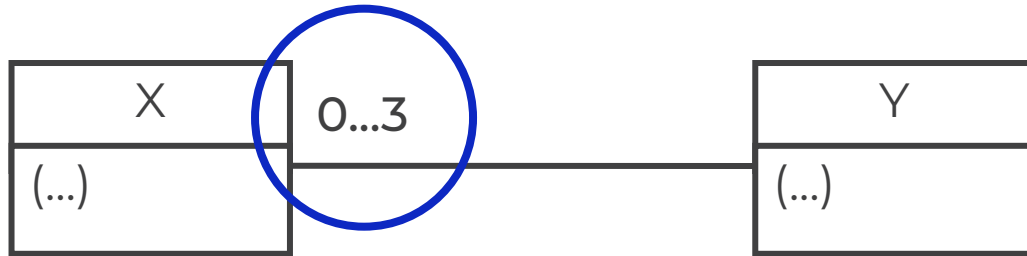
- **Associação:** Relação que permite especificar que objetos de uma dada classe se relacionam com objetos de outra classe, sendo importante saber para cada objecto quais os objectos com que está relacionado.



Um cliente pode estar associado a **muitas faturas, ou a nenhuma**.

Uma fatura está sempre associada **a um e apenas um cliente**.

Multiplicidade das Associações

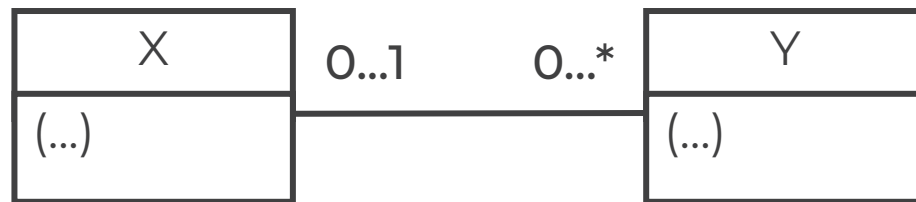


Indica **quantos** objetos da classe X...

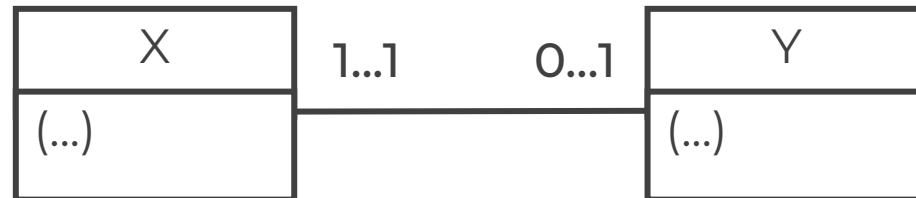
... podem estar associados a **um** objeto da classe Y

Multiplicidade das associações

... há muitas combinações possíveis. Eis as designações mais comuns:



Um-para-Muitos

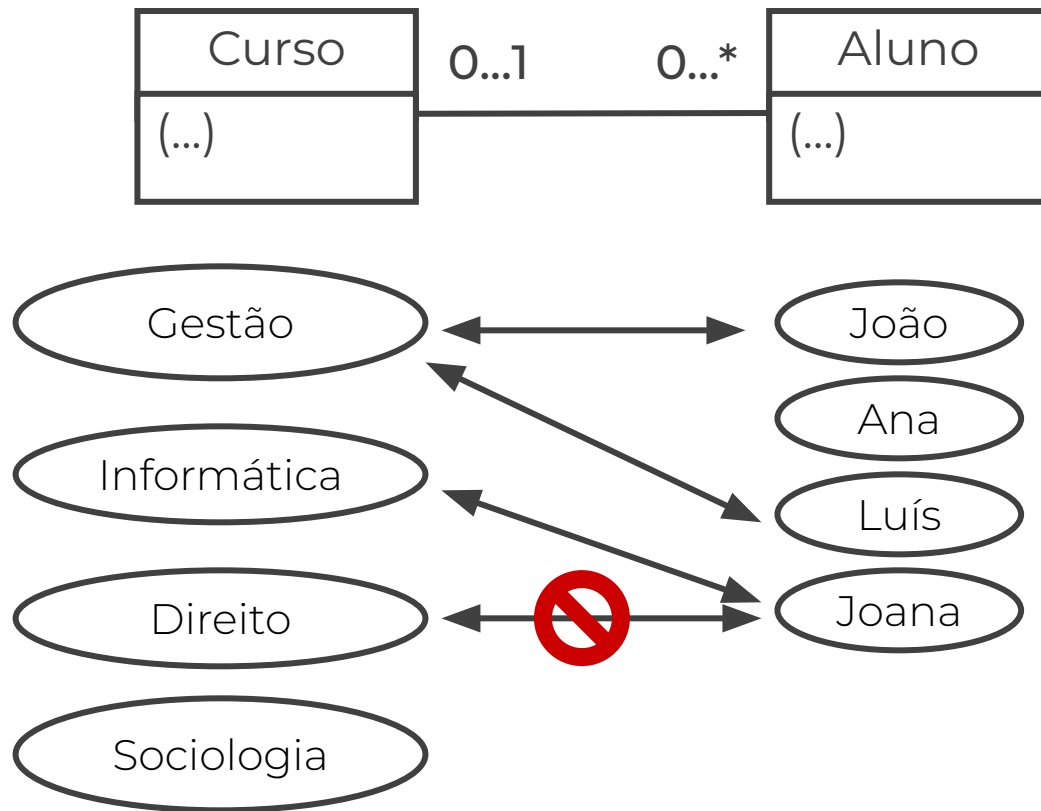


Um-para-Um



Muitos-para-Muitos

Associação **Um-para-Muitos**



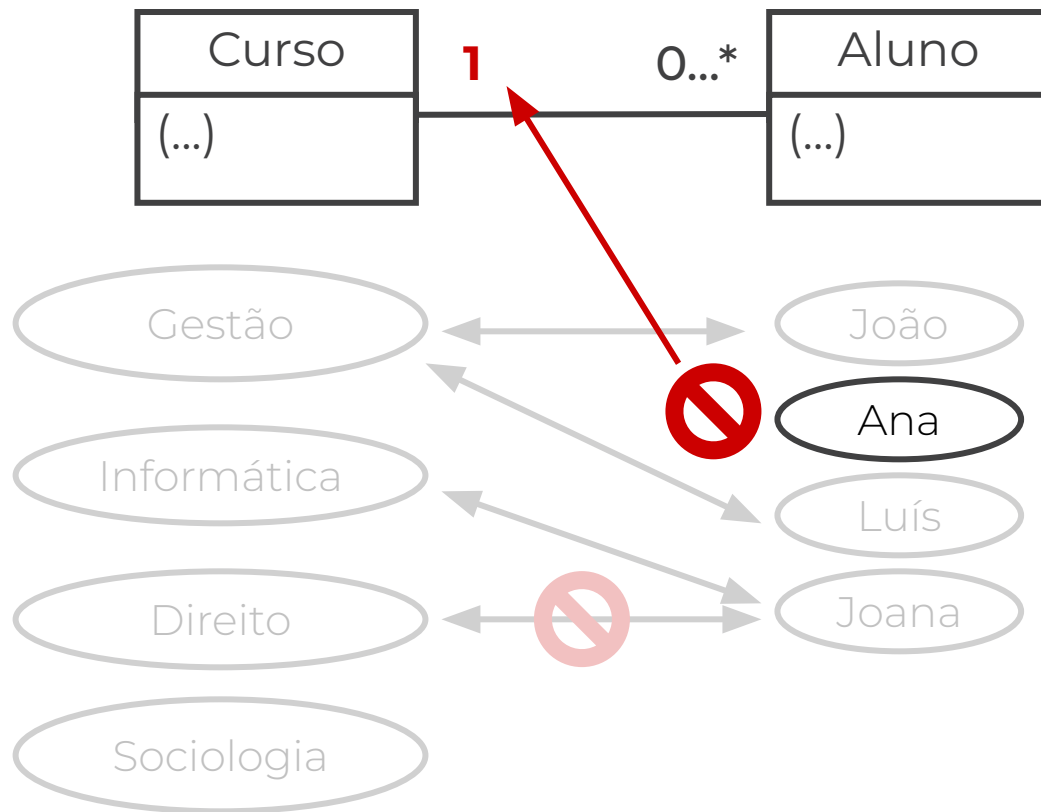
- **Semântica**

- Um aluno pode estar associado a (inscrito em) apenas um curso
- A um curso podem-se associar vários ou nenhum aluno.

- **Funcional**

- Dado um aluno é possível determinar em que curso está inscrito, e
- Dado um curso é possível identificar os seus alunos.

Associação **Um-para-Muitos**



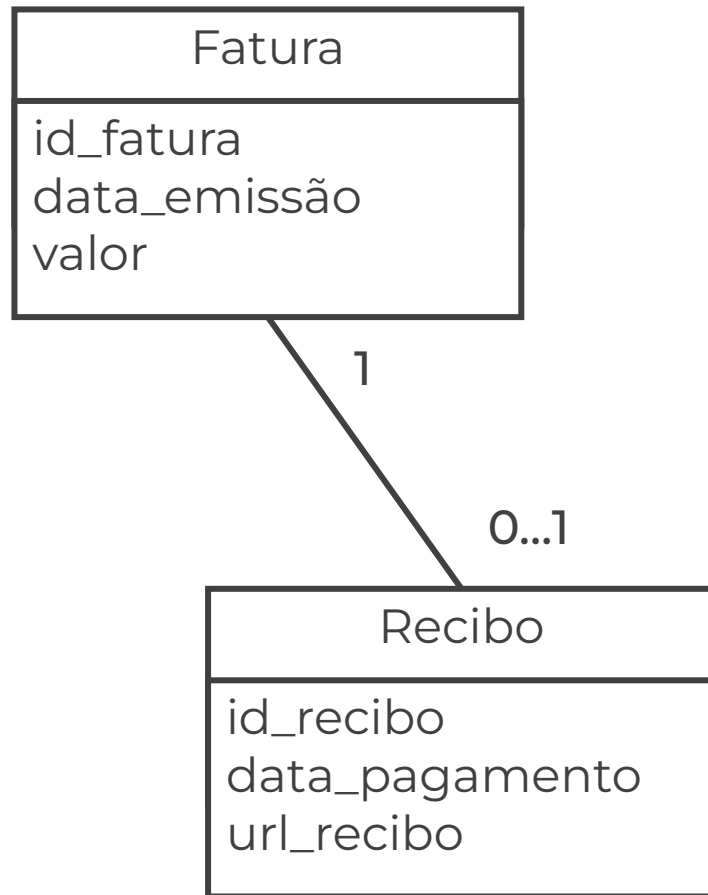
- **Semântica**

- Um aluno pode estar associado a (inscrito em) apenas um curso
- A um curso podem-se associar vários ou nenhum aluno.

- **Funcional**

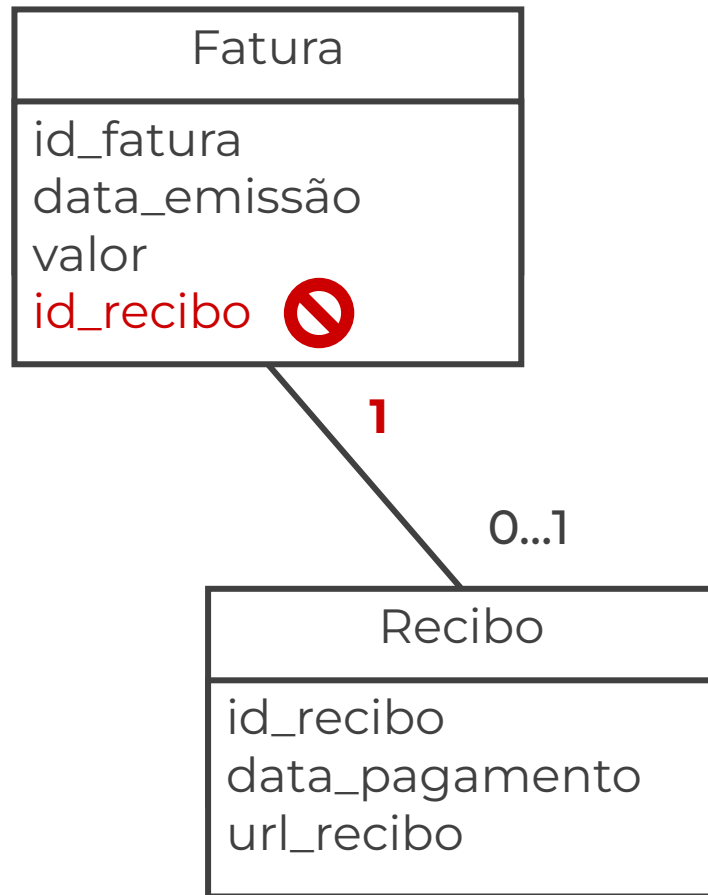
- Dado um aluno é possível determinar em que curso está inscrito, e
- Dado um curso é possível identificar os seus alunos.

Associação **Um-para-Um**



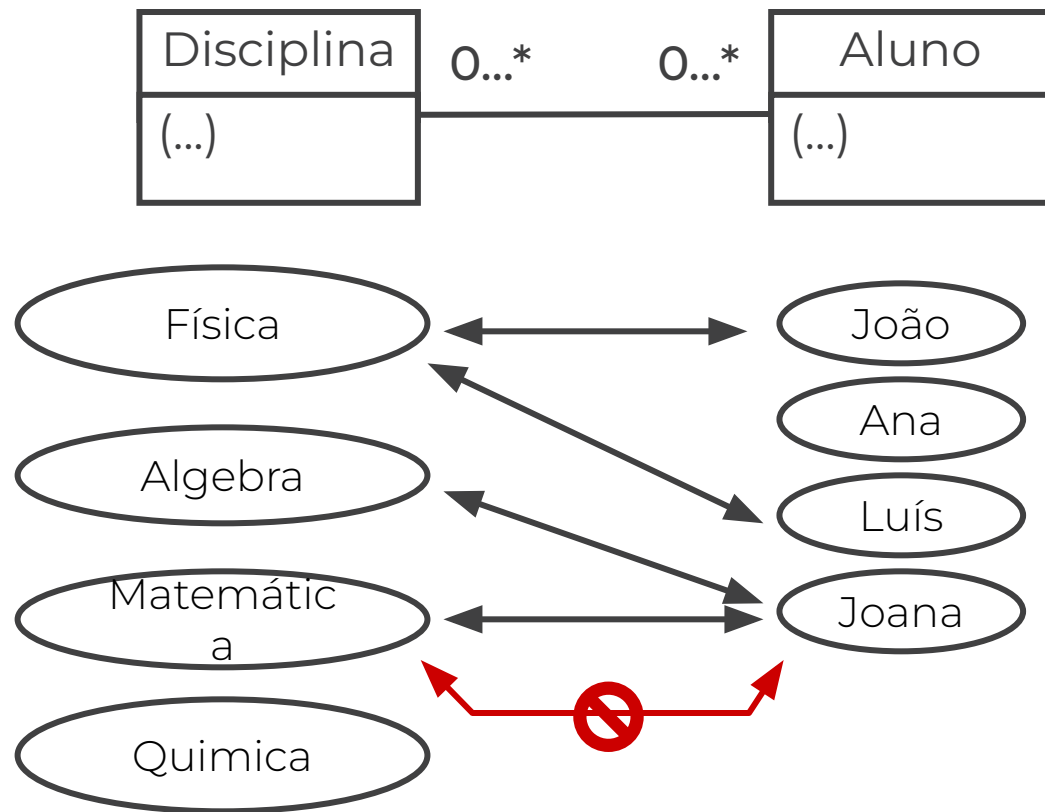
- Pelas multiplicidades, percebe-se que uma factura será necessariamente introduzida antes do respectivo recibo (quanto muito, simultaneamente).

Associação Um-para-Um



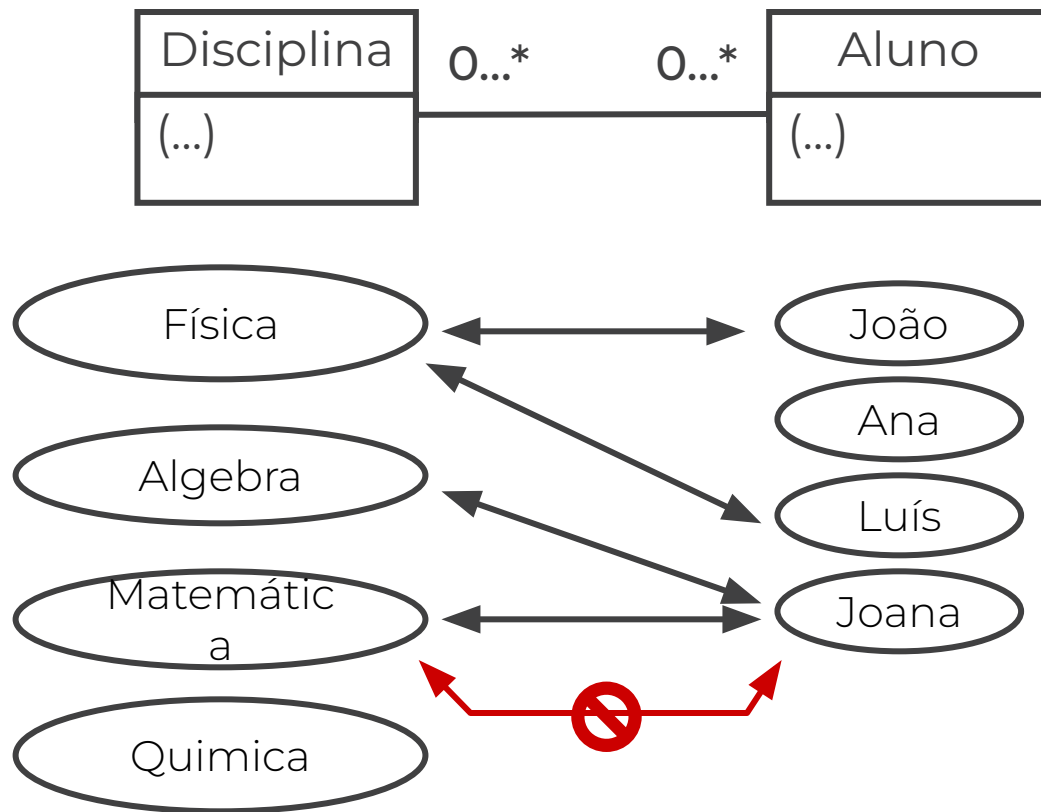
- **Não devemos colocar a chave estrangeira como atributo.** É a associação que atribui um número de recibo à factura.
 - Caso contrário, uma factura poderia ficar associada a dois números de recibo (o indicado no atributo da factura e o indicado no objecto *Recibo* ao qual a factura estivesse ligada).

Associação Muitos-para-Muitos



- Um objecto não pode estar duplamente associado a outro objecto (Joana / Matemática).
- À semelhança das classes (em que os objetos são distintos), as associações também têm que ter ocorrências distintas.
 - **Não há nada que distinga as duas ligações entre *Joana* e *Matemática*;**

Associação Muitos-para-Muitos



- O sistema de base de dados deverá considerar a 2ª ligação como redundante: Não interessa registar duas vezes que *Joana* e *Matemática* estão ligados
 - **Se interessa, então a associação muitos-para-muitos não é representação correcta.**

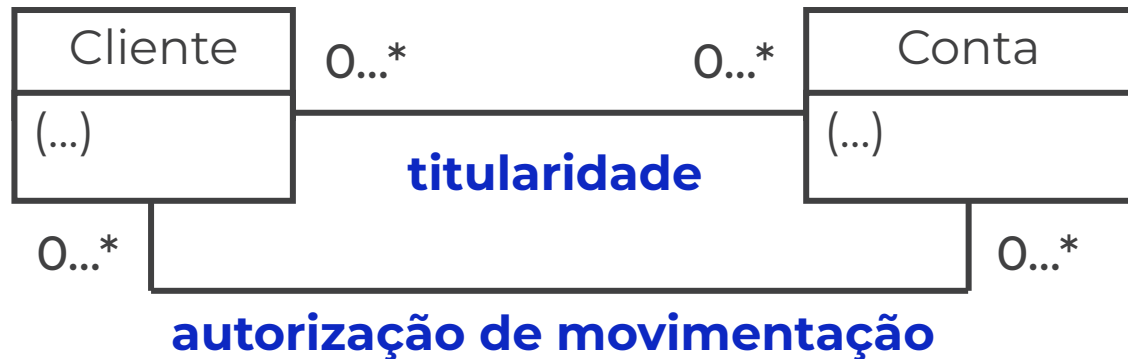
Nomes para Associações

- As associações podem (e devem) ter nomes



Normalmente usamos substantivos

- Especialmente relevante quando há mais do que uma associação entre duas classes



0...* ou 1...*

- Usar o 1...* quando, **de todo**, não se quer a informação do lado 1 sem a informação do lado *muitos*.
 - Exemplo 1:
 - Um curso tem pelo menos uma disciplina (portanto, na realidade: 1...*).
 - Mas podemos querer manter informação acerca do curso sem ainda termos indicado as suas disciplinas, **então usamos: 0...***

0...* ou 1...*

- Exemplo 2:
 - Uma receita médica prescreve um ou mais medicamentos.
 - O médico não deve poder guardar a receita no sistema sem ter indicado um dos medicamentos.
 - Sem pelo menos um medicamento, a informação sobre a receita não tem qualquer utilidade, **então usamos: 1...***.

Resumindo,
Cuidado com o 1...* ou 1...1 porque **pode bloquear a introdução de informação provisória** que pode ser útil (curso ainda sem disciplinas).

**Estamos agora prontos para desenhar
a primeira base de dados simples.**

Exercício 1

Pretende desenvolver-se um sistema para gerir o *Youth SPA*. O descritivo abaixo descreve as necessidades do sistema.

“Pretendemos desenvolver um sistema que nos permita fazer uma gestão básica do nosso dia-a-dia no SPA. O sistema deve permitir registar os nossos clientes, os tratamentos disponíveis (com os respetivos preços). A informação relevante a guardar dos clientes é o seu nome, a idade e o NIF. O processo de faturação deve estar automatizado e o sistema deve permitir consultar o histórico de faturas de um determinado cliente (com a data de emissão e valor que o cliente pagou).”

Elabore o diagrama de classes da base de dados necessária para suportar o sistema descrito.

Nota 1.

Distinguir **funcionalidade** de **informação**.

Muitas vezes, descritivos de clientes como o do slide anterior misturam funcionalidade com informação.

Na concepção do modelo de dados, é importante distinguir.

No âmbito das bases de dados, apenas nos importa a **informação**, e podemos ignorar todas as funcionalidades, **nunca ignorando a informação de suporte à funcionalidade pedida**.

Nota 2.

Optimizar.

Muitas vezes, descritivos de clientes como o do slide anterior são redigidos de uma forma que apesar de ser a que o cliente descreve, não é necessariamente a que melhor permite definir o sistema.

Um bom exemplo disso no descritivo anterior é “*A informação relevante a guardar dos clientes é o seu nome, a **idade** e o NIF.*” **Será que devemos guardar a idade?**

Ou antes a data de nascimento?

Devemos guardar a data de nascimento, caso contrário a base de dados vai ter de ser alterada anualmente para refletir essa mudança.

Exercício 2

Pretende desenvolver-se um sistema para gerir o *Cool Bar*. O descritivo abaixo descreve as necessidades do sistema.

“Queremos um sistema para gerir os nossos bares, tanto em Lisboa como no Porto. O sistema deve registar todos os empregados (bartenders e responsáveis do espaço). Para permitir efetuar os pagamentos corretamente, deve ser possível saber o regime de trabalho dos empregados (em percentagem). O sistema deve também permitir listar todas as bebidas diretamente do sistema do fornecedor.”

Elabore o diagrama de classes da base de dados necessária para suportar o sistema descrito.

O futuro profissional começa aqui

iscte

INSTITUTO
UNIVERSITÁRIO
DE LISBOA



emprego
digital



UPskill