ISEP INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO

Modelação Lógica de Dados

Modelo Relacional (MR)

DEE
Departamento de Engenharia Electrotécnica

Licenciatura em Engenharia de Telecomunicações e Informática

Modelo Relacional

O modelo relacional é atualmente o modelo mais popular

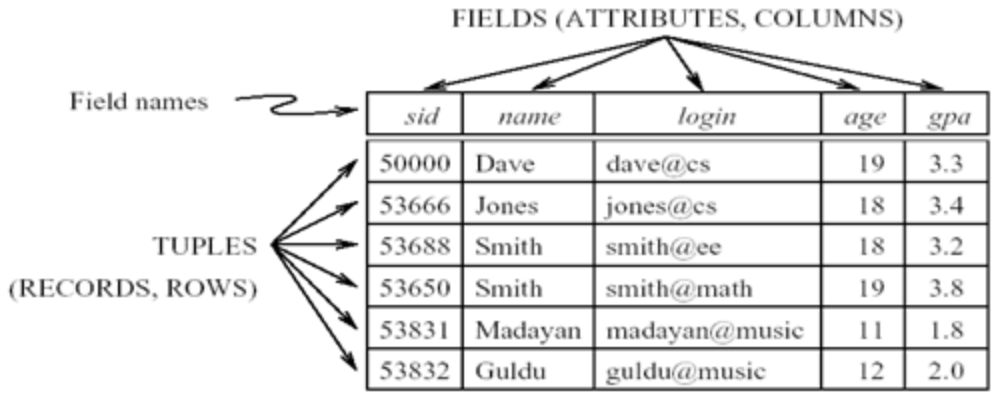
- Grande simplicidade
- Grande capacidade de resposta à necessidades dos utilizadores

Foi apresentado por Edgar F. Codd em 1970

O modelo assenta numa base teórica forte baseada em teorias matemáticas de conjuntos e de lógica de predicados

Conceitos básicos do MR

- A base de dados é um coleção de relações;
- Relação R = é a estrutura básica do MR e representa-se mediante uma Tabela;
- Tuplo = é uma ocorrência da relação. Representa-se mediante uma linha.
- Atributos Ai = representa as propriedades da relação e corresponde a uma coluna da tabela
- Domínio = é o conjunto válido de valores que tem um atributo.



sid	*name*	*login*	*age*	*gpa*
50000	Dave	dave@cs	19	3.3
53666	Jones	jones@cs	18	3.4
53688	Smith	smith@ee	18	3.2
53650	Smith	smith@math	19	3.8
53831	Madayan	madayan@music	11	1.8
53832	Guldu	guldu@music	12	2.0

Sailor(sid,nome,login,age,gpa)

Propriedades das Relações

- Cada **relação** tem um **nome distinto** de todas as outras relações
- Cada célula da relação contém exatamente um **valor atómico**
- Cada **atributo** tem um **nome distinto** dentro da relação
- Os valores de um **atributo** fazem todos parte do **mesmo domínio**
- Teoricamente, a ordem dos atributos não tem significado (na prática, a ordem pode afectar a eficiência no acesso aos tuplos)
- Cada tuplo **é distinto; não existem tuplos duplicados**

Design do MR

1. Determinar o propósito do sistema
2. Determinar quais as entidades/tabelas e os respetivos atributos a incluir;
3. Identificar chaves primárias;
4. Determinar as relações entre as tabelas;
5. Refinar o design (normalização)

MODELO RELACIONAL

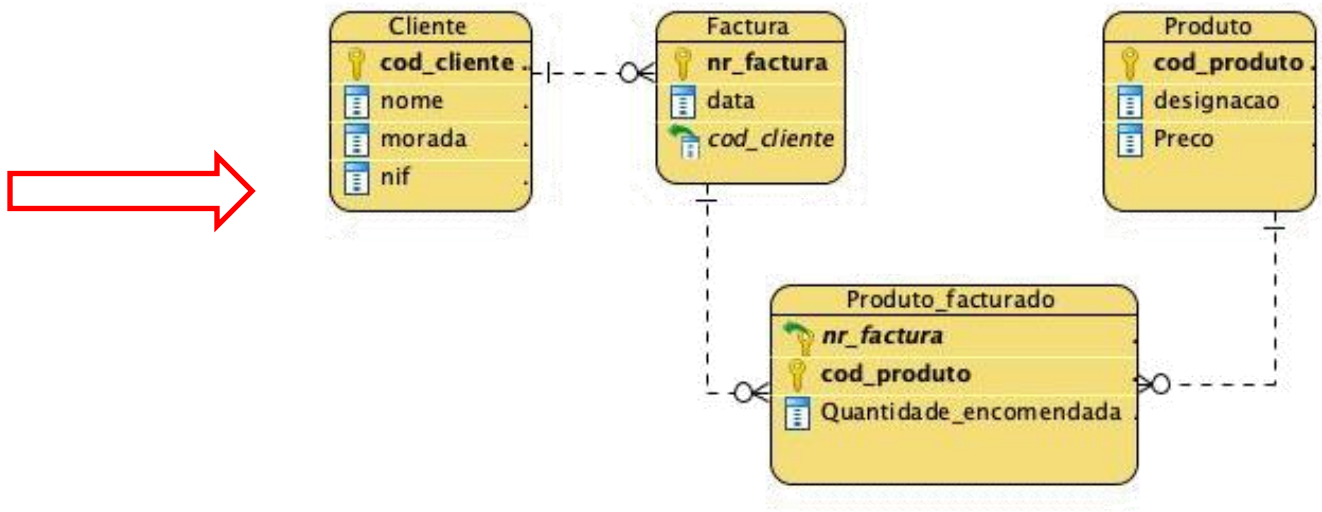
modelo lógico das BD relacionais

Design do MR

1. Determinar o propósito do sistema

- É necessário efetuar a análise de requisitos do sistema;
- Começar a construir o esquema relacional da Base e dados;
  - .....

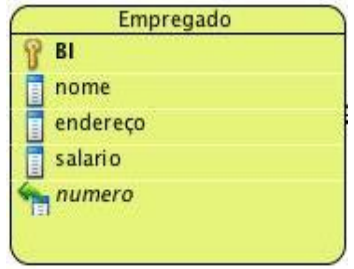
Por questão de simplificação da leitura do esquema relacional, optaremos por esta notação



Design do MR

2. Determinar quais as tabelas/entidades e respectivos atributos a incluir;

- Cada tabela deve conter informações sobre um assunto e cada atributo de uma tabela contém fatos individuais sobre o assunto da tabela ;
- Ao esboçar os atributos, ter em mente as seguintes dicas:
  - Relacionar cada atributo diretamente com o assunto da tabela;
  - Não incluir dados derivados ou calculados (dados que são o resultado de uma expressão).
  - Incluir todas as informações que precisa.
  - Armazenar informações nas suas partes lógicas mais pequenas .
  - **Os atributos têm de ser atómicos**



No modelo relacional os atributos não podem ser do tipo composto ou multi-valor;

Design do MR

3. Identificar chaves primárias

- No modelo relacional, uma tabela não pode conter linhas duplicadas, porque isso criaria ambiguidades na recuperação.
- Para garantir a singularidade, cada tabela deve ter uma coluna (ou um conjunto de colunas), chamada **chave primária**, que identifica exclusivamente todos os registos da tabela.
- Podem existir vários atributos cujos valores identificam exclusivamente uma ocorrência dessa tabela **: chaves candidatas. A chave primária é uma das chaves candidatas.**

Exemplo:

Nrcartãocidadao e ***número contribuinte*** são atributos que são chaves candidatas da entidade ***Funcionário***

Design do MR

3. Identificar chaves primárias

- Uma chave primária pode ser formada pela combinação de pelo menos dois ou mais atributos sendo nesse caso chamada **chave composta .**
- A chave primária também é usada para fazer referência a outras tabelas (a serem elaboradas posteriormente – aparece o conceito de chave estrangeira)

Alunos

Naluno	Nome	Idade	Dt_Matricula
1000	Abílio	19	05-09-05
1001	Daniela	20	05-09-05
1002	Elsa	18	07-09-05
1003	Filipe	21	10-09-05
1004	Sílvia	19	11-09-05

Disciplinas

CodDisc	Designacao
D01	Matemática
D02	Informática
D03	Inglês
D04	Física

Notas

Naluno	CodDisc	Data	Nota
1000	D01	10-10-05	15
1001	D01	10-10-05	17
1000	D02	15-10-05	16
1003	D03	21-09-05	13

Design do MR

Dicas a seguir:

- Os valores da chave primária devem ser únicos (isto é, sem valor duplicado);
- A chave primária deve sempre ter um valor. Por outras palavras, não deve conter NULL,
- A chave primária deve ser simples e familiar;
- O valor da chave primária não deve ser alterado. A chave primária é usada para fazer referência a outras tabelas. Se alterarmos o seu valor, é necessário alterar todas as referências; Caso contrário, as referências serão perdidas;
- A chave primária geralmente é constituída por uma coluna única (por exemplo, codigo_cliente, codigo_produto). Mas também pode constituir várias colunas. Deve-se usar o menor número de colunas possível.

Design do MR

4. Determinar as relações entre as tabelas

- Após se identificar as tabelas(entidades) e os respectivos atributos de cada tabela, **precisamos de relacionar de forma significativa as tabelas;**
- Um Relacionamento é uma associação entre atributos comuns (colunas) de duas tabelas;
  - Os atributos correspondentes são a **chave primária** de uma tabela que fornece um identificador exclusivo para cada registo e **uma chave estrangeira** na outra tabela;
  - **Grau -** é o número de tabelas participantes no relacionamento
    - **Relacionamento unário e reflexivo**: Um empregado supervisiona vários empregados
    - **Relacionamento binário:** Um empregado trabalha num departamento
  - **Cardinalidade -** Especifica o número de instâncias de relacionamento em que uma entidade pode participar.

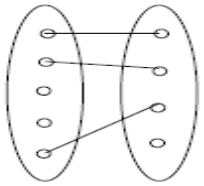
Design do MR

Tipo de cardinalidades:

- **1:1** (um-para-um)

  Um funcionário gere um departamento

  Um departamento é gerido por um funcionário

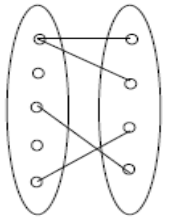


1-para-1
1-1

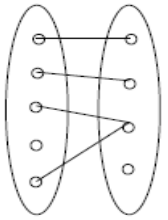
- **1:N** ou **N:1** (um-para-muitos) ou (muitos-para-um)

  Um funcionário gere muitos departamentos

  Um departamento é gerido por um funcionário



1-para-Muitos
1-N

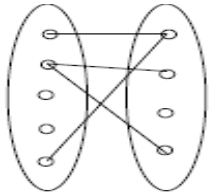


Muitos-para-1
N-1

- **N:M** (muitos-para-muitos)

  Um funcionário trabalha em muitos departamentos

  Um departamento tem muitos funcionários

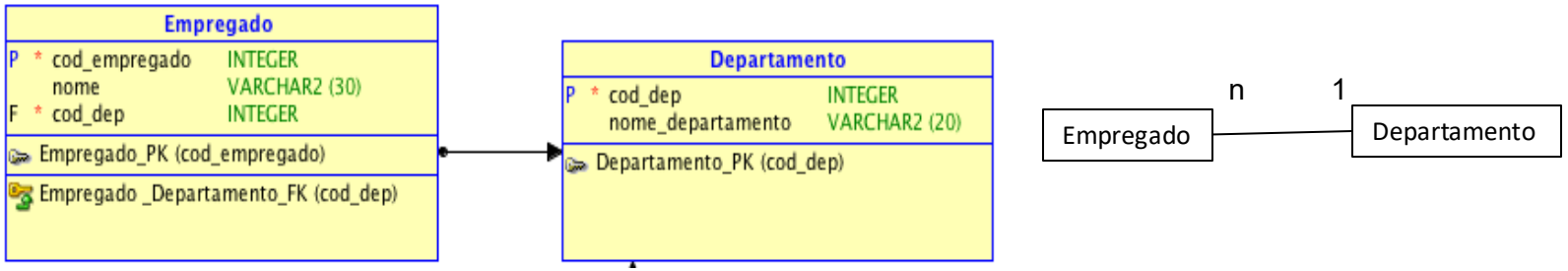


Muitos-para-Muitos
N-M

Design do MR

Dicas para os relacionamentos:

- **Um- para-muitos** – a chave primária de um lado torna-se numa chave estrangeira no lado oposto

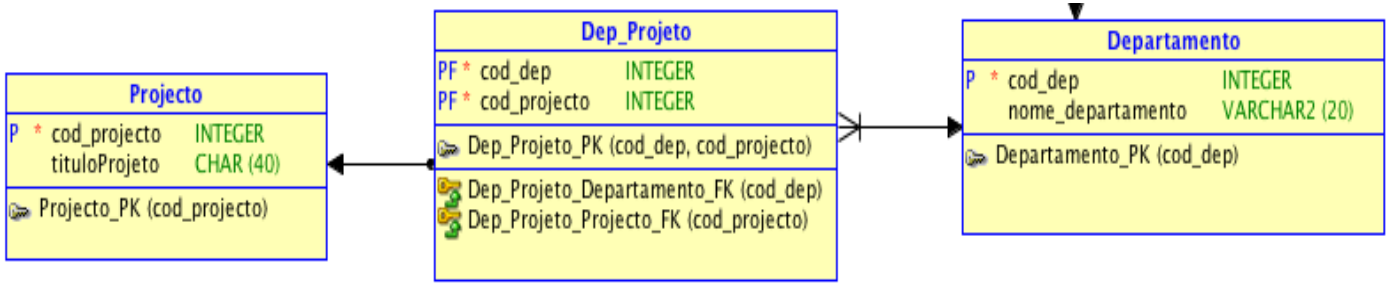


- **Um- para- Um** - chave primária no lado obrigatório torna-se uma chave externa no lado opcional

Design do MR

Dicas para os relacionamentos:

- **Muito-para-Muitos - Criar uma nova relação com as chaves primárias das duas entidades como sua chave primária**

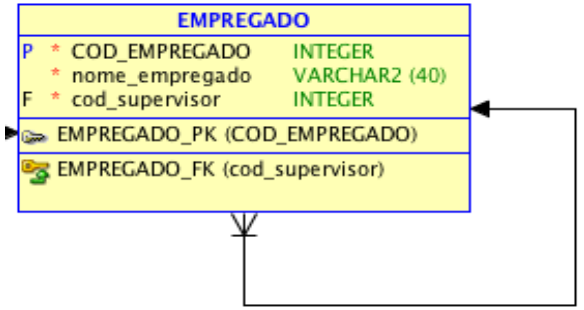


No modelo relacional não pode haver relacionamentos de muito- para-muitos;

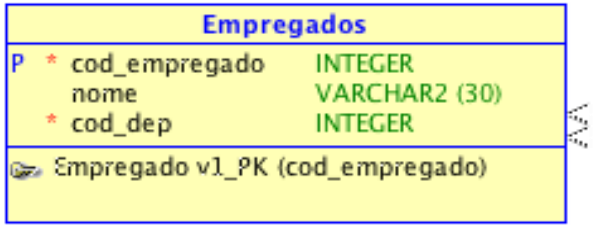
Design do MR

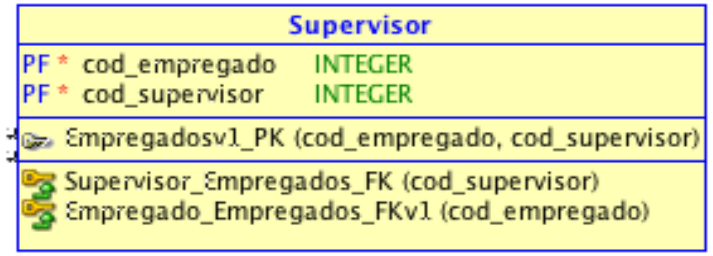
Dicas para os relacionamentos:

- **Nas relações unárias**
  - Um-para-muitos
    - a chave estrangeira recursiva na mesma relação



  - **Muitos-para-muitos- duas relações:**
    - Uma para a entidade tipo e outra para uma relação associativa em que a chave primária tem dois atributos ambos tirados da chave primária da entidade





Design do MR

Dicas para os relacionamentos:

- **Nas relações tipo generalizações**

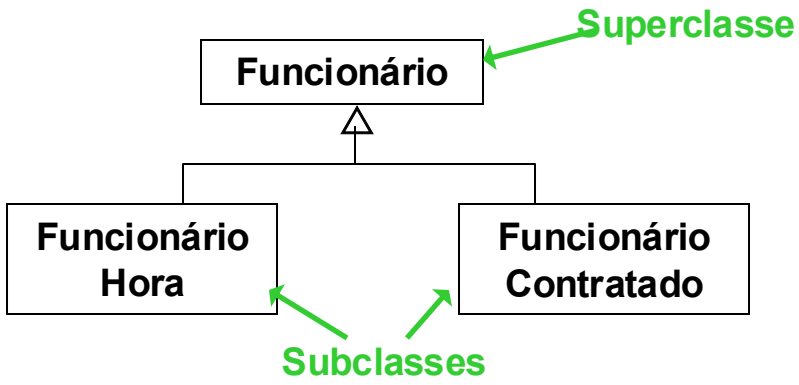
  Há três abordagens básicas

  1. Criar uma única tabela que conterá todos os atributos da superclasse e subclasse. Acrescenta-se um atributo para identificar cada tipo subclasse.
  2. Criar uma tabela para superclasse e criar uma tabela para cada subclasse. Incluir o atributo chave da superclasse em cada uma destas tabelas.
  3. Criar uma tabela para cada subclasse e incluir todos os atributos da superclasse;

Funcionário = (nome, sexo)

FuncionárioHora=(ordenado_hora, horas_efetivas)

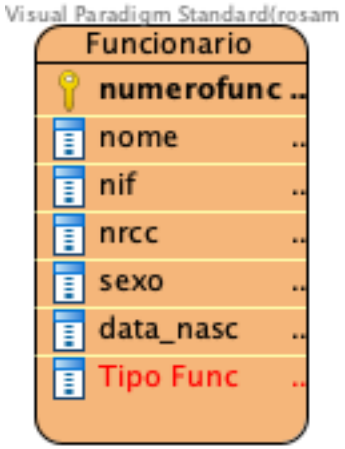
FuncionárioContratado =(contrato_id)



Design do MR

1ª abordagem: Criar uma Tabela que conterá todos os atributos da superclasse e subclasse.

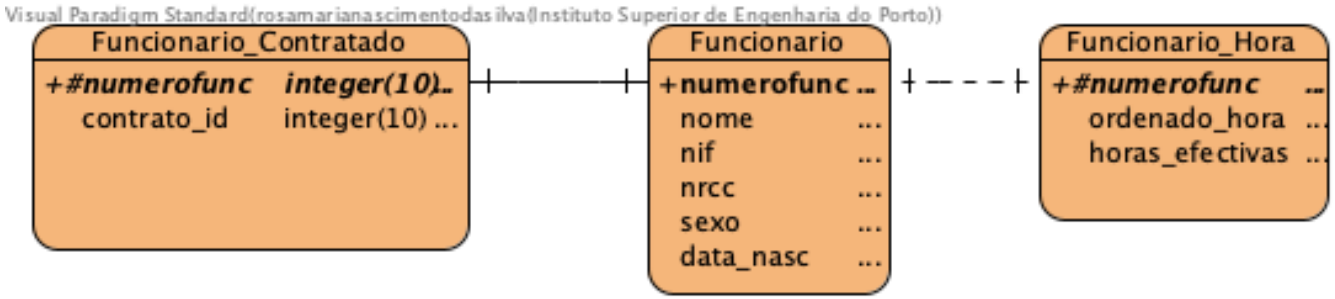
- A tabela *Funcionário* contém todos os atributos da tabela *Funcionário_Hora* e da tabela *Funcionário_ Contratado*
- Acrescenta-se um novo atributo TipoEmp para distinguir o tipo de funcionários, neste caso, *Funcionário_Hora* ou , *Funcionário_Contratado*.



Design do MR

2ª abordagem: Criar uma tabela para superclasse e criar uma tabela para cada subclasse

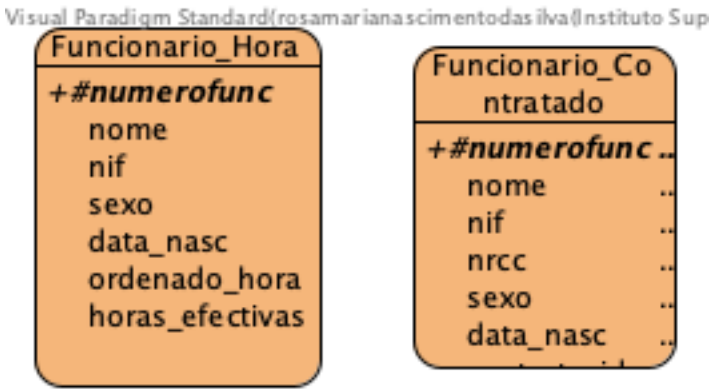
- A criação das tabelas *Funcionario_Hora* e *Funcionario_Contratado* é similar
- Na tabela *Funcionario_Hora* são incluídos os atributos *ordenado_hora*, *horas_efectivas* e os atributos chave da superclasse, neste caso *NumEmp*.
- A chave primária da superclasse é também a chave primária da relação *Funcionario_Hora* e também chave estrangeira da tabela superclasse, neste caso *Funcionario*.
- Para cada entidade de *Funcionario_Hora*, o valor dos atributos *Nome* e *Sexo* são guardados na linha correspondente da superclasse (*Funcionario*).
- Note-se que se um tuplo da tabela *Funcionario* é apagado, a operação deve ser efectuada em cascata e eliminar também os tuplos correspondentes da relação *Funcionario_Hora*



Design do MR

3ª abordagem: Criar uma tabela para cada subclasse e incluir todos os atributos da superclasse

- ✓A tabela *Funcionario_Hora* contém todos os atributos da tabela *Funcionario_Hora* e da tabela *Funcionario* (isto é, *NumEmp, Nome, Sexo, ordenado_hora*, *horas_efectivas*)
- ✓A tabela *Funcionario_Contratado* é similar a esta



Design do MR

- **A segunda abordagem** é geral e sempre aplicável
  - ✓Pesquisas em que apenas sejam examinados todos os funcionários e que não seja relevante os atributos das subclasses são tratadas simplesmente utilizando a relação *Funcionario*.
  - ✓Contudo, podem haver pesquisas que seja necessário combinar as relações *Funcionario_Hora* e *Funcionario*, se por exemplo se for pretendido conhecer o *Nome*, *Sexo* e *ordenado_hora*.
- **A terceira abordagem** não é aplicável se existirem funcionários **que não são nem contratados por contrato nem contratados à hora**, uma vez que não há forma de guardar estes funcionários.
  - ✓Também **não é possível armazenar o mesmo funcionário como trabalhador por contrato e como trabalhador à hora, pois tinha-se de armazenar o mesmo funcionário duas vezes – redundância.**
  - ✓Uma pesquisa que necessite de **analisar todos os funcionários precisa de relacionar as duas relaçõ**es, *Funcionario_Hora* e *Funcionario_Contratado*, obrigatoriamente
  - ✓Por outro lado, para analisar toda a informação sobre os funcionários que trabalham à hora apenas tem de se utilizar a relação *Funcionario_Hora*.

A escolha entre as duas abordagens depende dos dados e da frequência das operações mais comuns

Modelo Relacional