

Modelagem de Dados

Prof. Bianca & Prof. Agord

2021

Conteúdo

- Níveis de Abstração em Banco de Dados
- Modelo Entidade-Relacionamento
- Modelo Relacional
- Transformação entre Modelos
- Modelo Entidade-Relacionamento Estendido
- Diferentes Notações

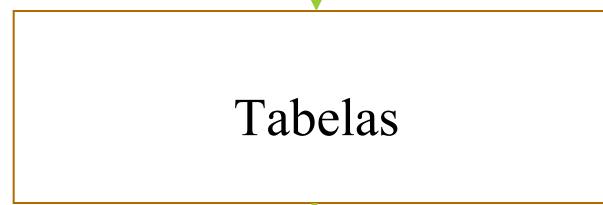
Níveis de Abstração dos Dados em BD

Nível Conceitual



DER

Nível Lógico



Tabelas

Nível Físico



SQL



Modelo Entidade-Relacionamento

- Notação criada em 1976 por Peter Chen para representar o projeto conceitual de um BD
- Popularmente chamada de MER ou DER
- Elementos principais são as entidades e seus relacionamentos

Entidade

um objeto com existência física (pessoa, carro, casa) ou conceitual (empresa, universidade, curso), composto por propriedades que o descreve, chamadas atributos.

Empregado

IdEmpregado

NomeEmp

Endereço

Salário

Departamento

IdDpt

NomeDpt

Local

Projeto

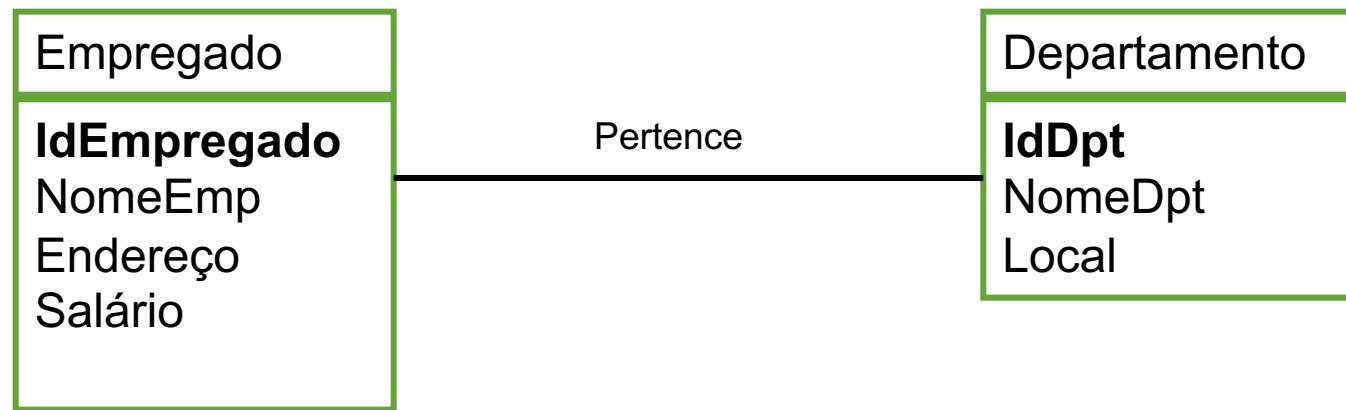
IdProj

NomePrj

Local

Relacionamento

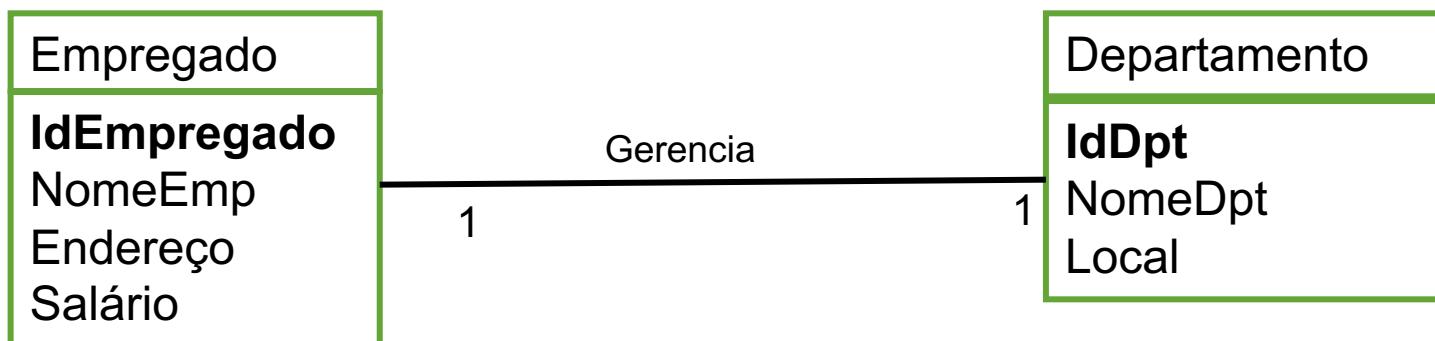
associação entre duas ou mais entidades



Cardinalidade dos relacionamentos

✓ 1 para 1 (um para um)

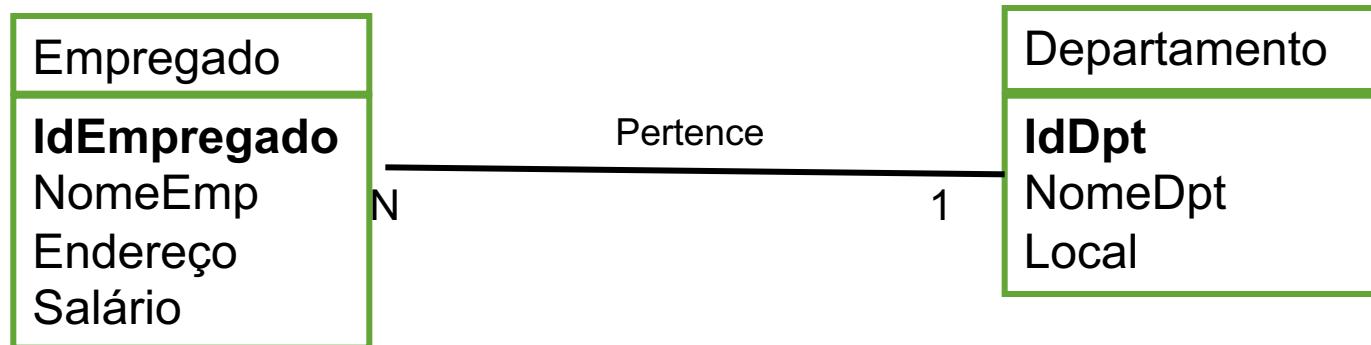
- Cada elemento de uma entidade relaciona-se com apenas um elemento da outra.
- Exemplo: cada departamento é gerenciado por um único funcionário. Um funcionário gerencia apenas um departamento.



Cardinalidade dos relacionamentos

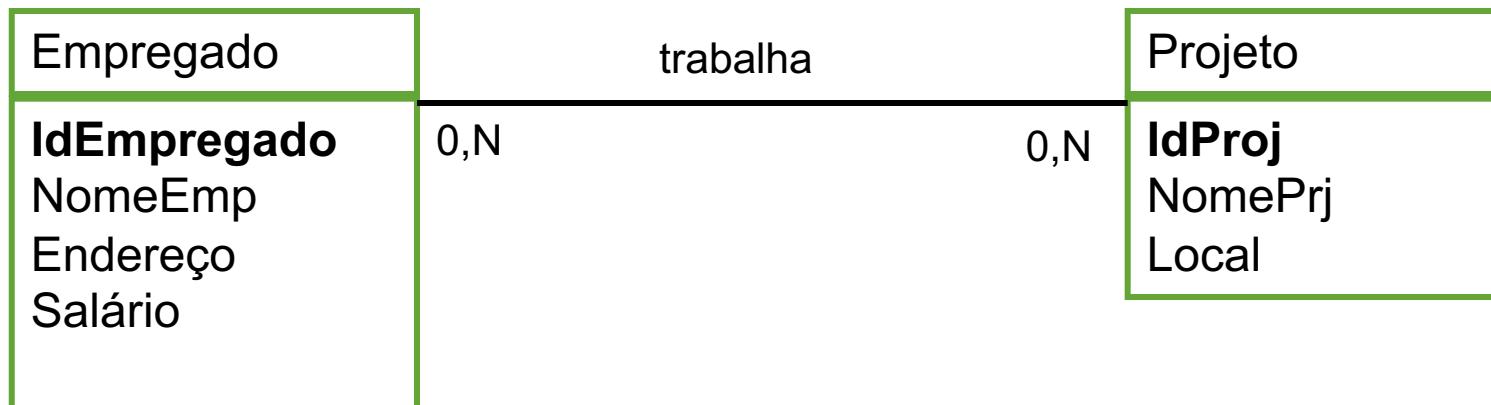
✓ 1 para N (um para muitos)

- Cada elemento de uma entidade relaciona-se com vários elementos da outra.
- Exemplo: cada departamento possui vários empregados, mas cada empregado está lotado em apenas um departamento



Cardinalidade dos relacionamentos

- ✓ N para N (muitos para muitos)
 - Cada elemento de uma entidade relaciona-se com vários elementos da outra e vice-versa.
 - Exemplo: um empregado pode participar de vários projetos e um projeto pode ter a participação de vários empregados.



Cardinalidade com participação

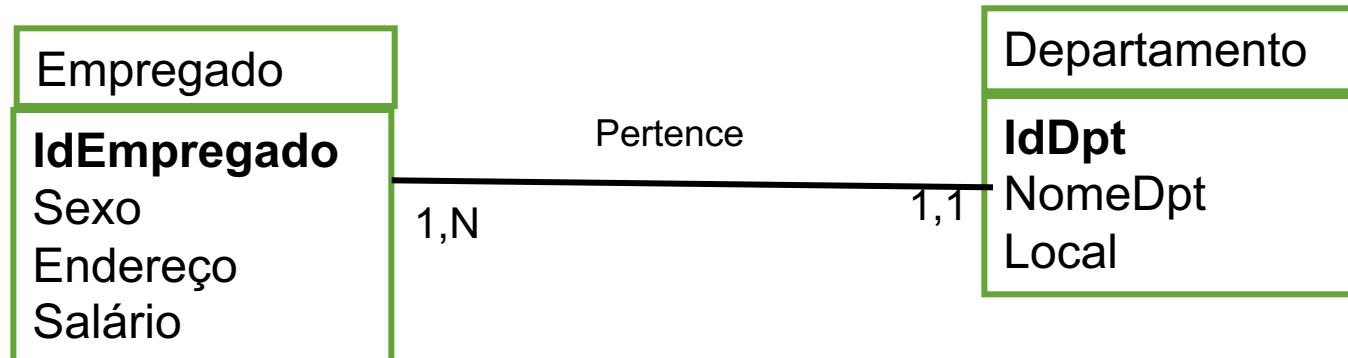
- Para representar todas as restrições de um BD, não é suficiente dizer apenas em que número (1 ou N) uma entidade aparece em um relacionamento. É necessário dizer se sua participação é opcional (0) ou obrigatória (1).
- Para isto, representa-se a cardinalidade através de um par (min, max), onde:
 - Min indica a participação da entidade no relacionamento. Pode ser obrigatória (1) ou opcional(0).
 - Max indica o número de vezes que a entidade aparece no relacionamento. Pode ser 1 ou N(muitos).

Possibilidades de cardinalidade:

	Um	Muitos
Opcional	(0,1)	(0,N)
Obrigatória	(1,1)	(1,N)

Cardinalidade com Participação

Todos os departamentos possuem pelo menos um empregado. Todo empregado trabalha para algum departamento.



Ferramentas CASE

- ✓ CASE = Computer Aided Software Engineering
- ✓ Neste curso usaremos a ferramenta case para projetos de BD:



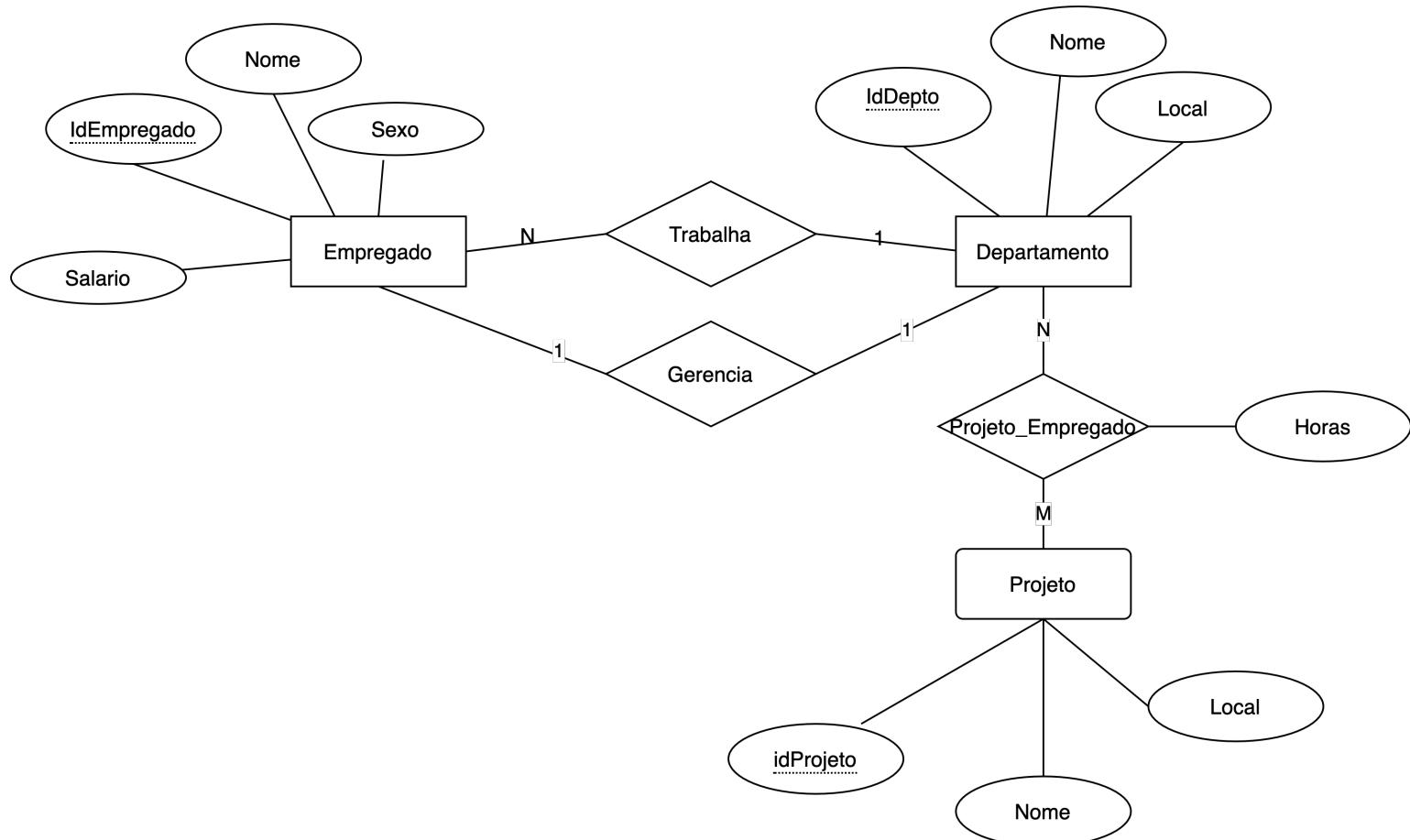
- ✓ Uma ferramenta CASE para banco de dados deve ser capaz de:
 - Fazer mapeamento do modelo ER para modelo relacional
 - Gerar scripts SQL
 - Fazer engenharia reversa

Exemplo

Uma companhia deseja um BD para armazenar seus empregados, departamento e projetos. Para tal, considere a seguinte descrição:

- A companhia é organizada em departamentos. Cada **departamento** tem um único nome, um único nro e um determinado empregado que gerencia o departamento.
- Um departamento **controla** um nro de **projetos**, cada um deles tem um único nome, um único nro e uma localização.
- É armazenado o nome, IdEmpregado, endereço , salário, sexo e data de nascimento de um empregado.
- Um **empregado** é **associado** a um departamento mas pode **trabalhar** em vários projetos, que não são necessariamente controlados pelo mesmo departamento. Além dessas informações, deve ser armazenado o nro de horas que cada empregado trabalha em um projeto.

Modelo Conceitual



DER: Várias Notações

CHEN

MERISE

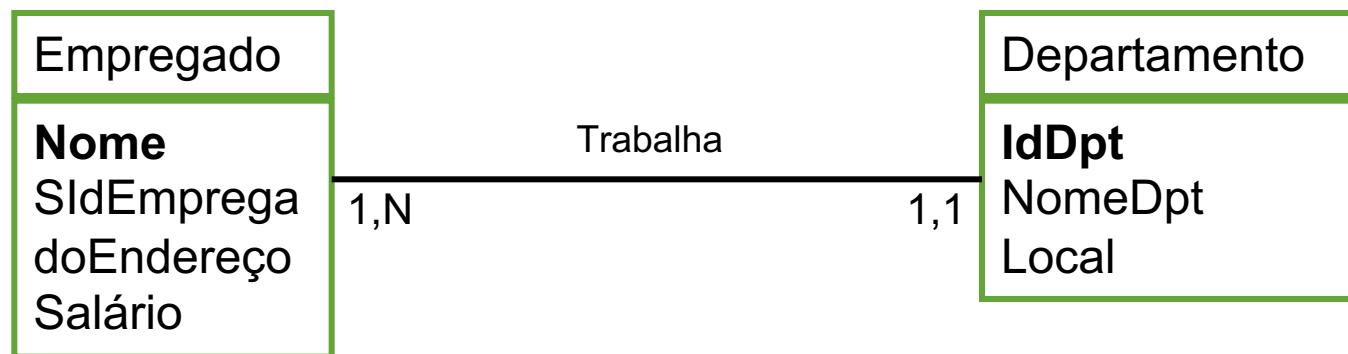
PÉ-DE-GALINHA

UML

Chen

- Peter Chen foi um dos primeiros autores a lançar um livro sobre abordagem E-R, por isso que sua notação é a mais utilizada

Ex:

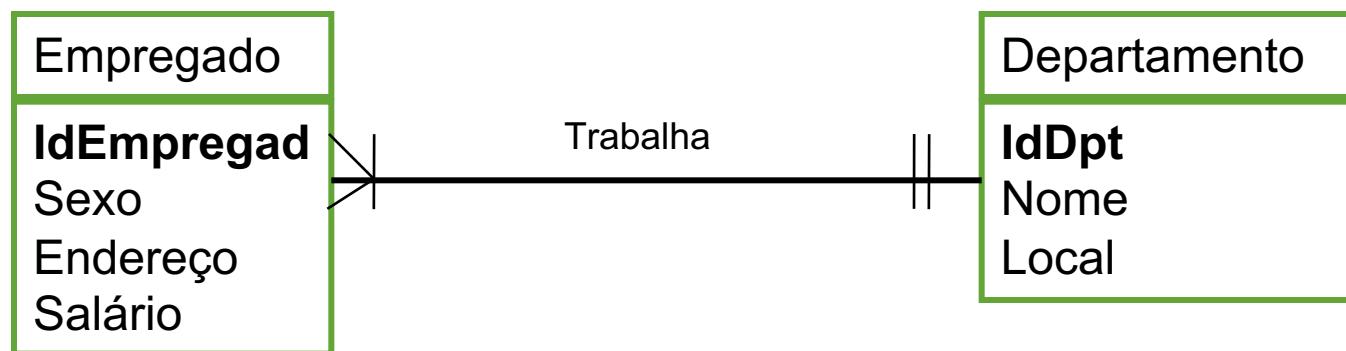


1,N é a cardinalidade do Departamento

1,1 é a cardinalidade do Empregado

Pé-de-galinha

- Chamada também de notação Engenharia de Informações ou notação James Martin
- Para a Engenharia de informação (método de desenvolvimento de sistemas de informação), foi definida a seguinte notação gráfica
- Ex:



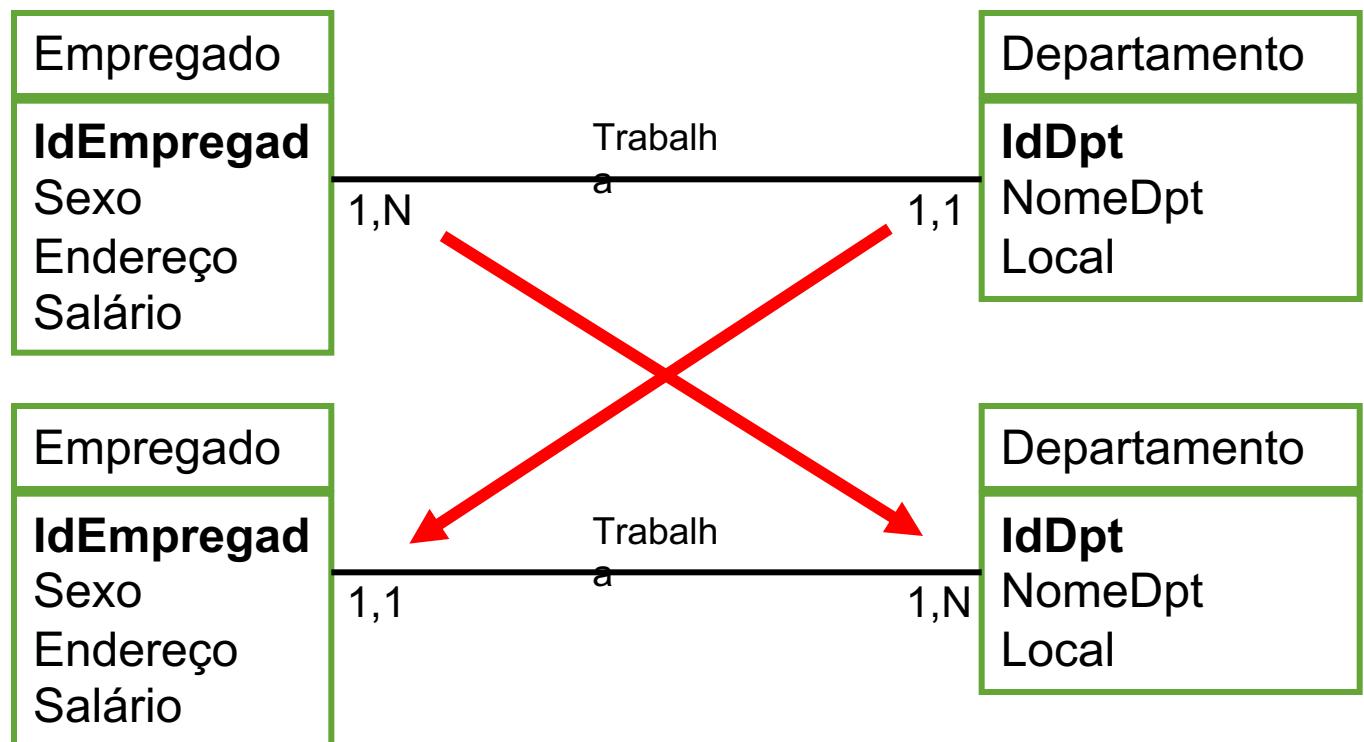
Merise

- ✓ Muito utilizado em ferramentas CASE. Coloca-se a cardinalidade do lado da entidade a que ela se refere (contrário do Chen).

- Chen



- Merise



UML

- UML (*Unified Modeling Language*) é uma linguagem utilizada em projetos de software orientados a objetos
- O DER á la UML é baseado no Diagrama de classes da UML
- A representação de Mínimo e Máximo é chamado de multiplicidade e essas são especificadas na forma *MIN..MAX* onde *** indica que não há limite. As multiplicidades são colocadas nas extremidades opostas do relacionamento em comparação. Um único *** representa *0..**, um único *1* representa *1..1*

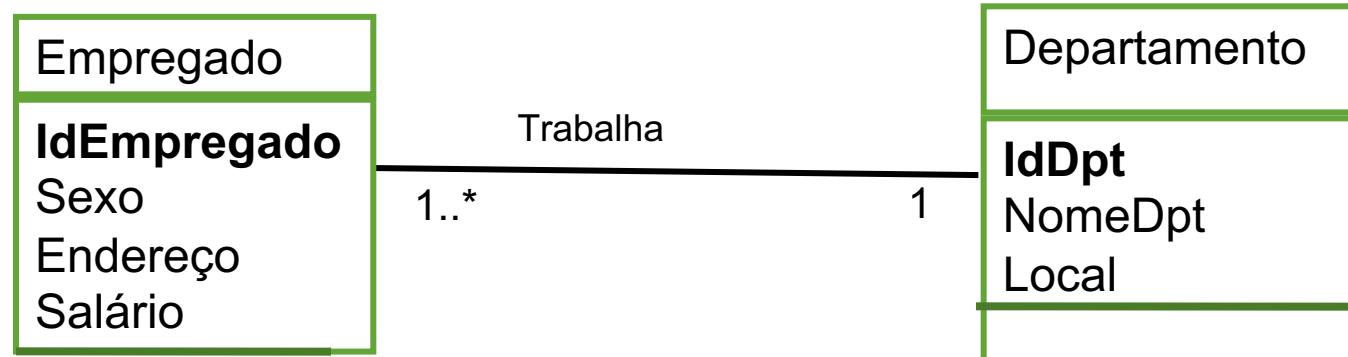


Tabela comparativa

Conceito	Símbolos:	DER		UML
	CHEN	Pé-de-galinha	Merise	
Entidade				
Relacionamento				
Atributos				São representados na classe
Atributo identificador				Nada consta
Generalização				
Entidade Associativa		X		X
Cardinalidade	1:1 1:N N:N 0:1 0:N	+ > > +○ >○	1:1 1:N N:N 0:1 0:N	1 1..* * 0..1 0..*

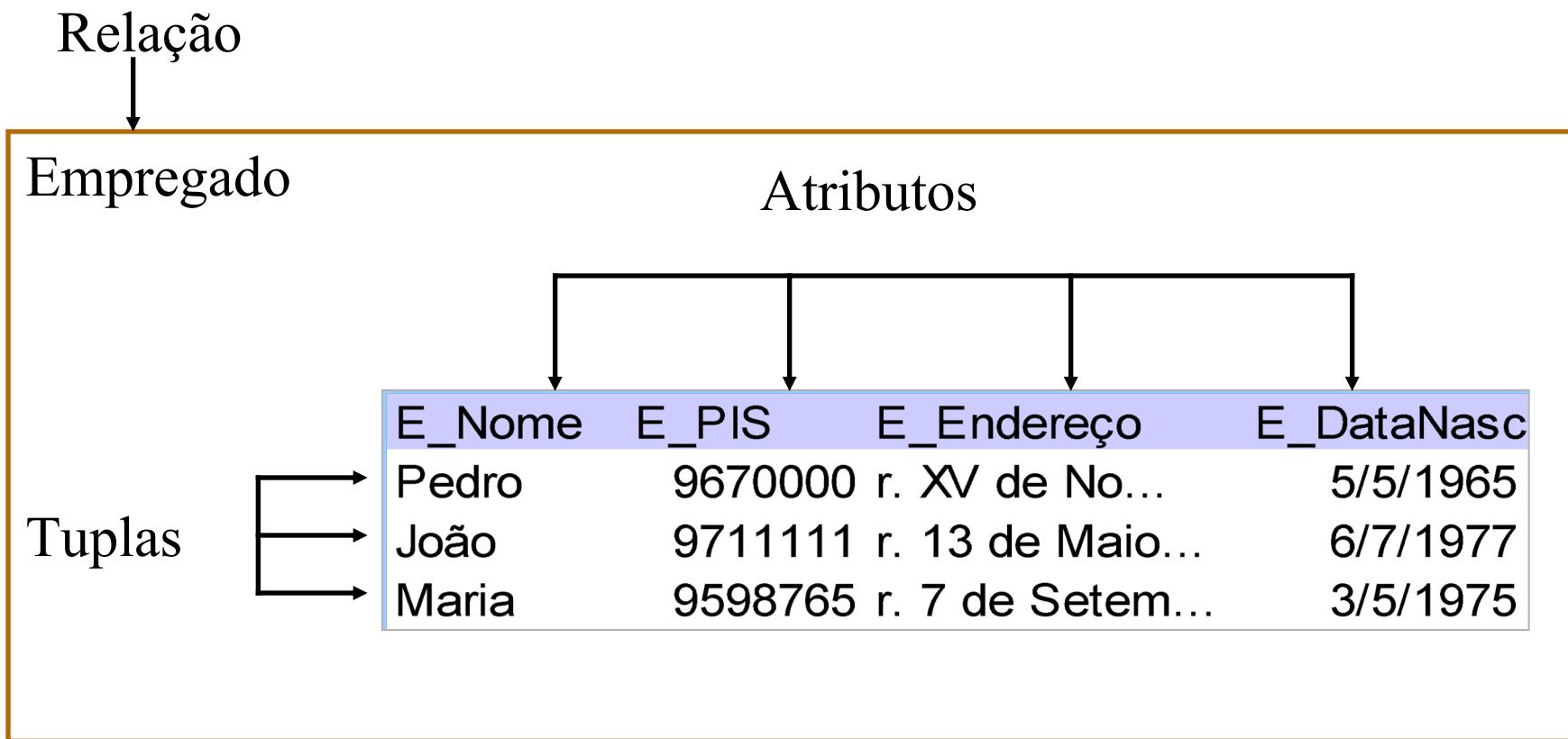
Fonte: Sistemas de Banco de Dados, Fundamentos e Aplicações, Elmasri, Ed. LTC

Modelo Relacional

Modelo Relacional

- Modelo de dados onde o BD consiste em uma coleção de **tabelas**
- Cada tabela é chamada **relação** porque corresponde a este conceito matemático
- Cada linha de uma tabela recebe o nome de **tupla**
- Cada coluna de uma tabela recebe o nome de **atributo**

Modelo Relacional



Restrições do Modelo Relacional

✓ Domínio

- Todo atributo deve ter um valor atômico (indivisível). Não é possível a existência de valores compostos ou multivalorados. Atributo composto é um atributo composto de várias partes, por exemplo: endereço pode ser composto de logradouro, número, complemento.

✓ Chave

- Toda tupla/linha tem que ser distinta. Duas tuplas não podem ter a mesma combinação de valores para todos os seus atributos. Um atributo chave distingue apenas uma tupla em uma relação.

Chaves

- Chave primária é o atributo que identifica uma única linha em uma tabela
- Chave estrangeira é o atributo de uma tabela que faz referência à chave primária de outra tabela, para implementar um relacionamento

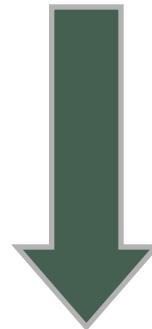
Funcionários			
Nome	PIS	Rg	Deptº
Jose Silva	123	987	D1
Jose Silva	235	789	D2
Joao Sa	252	126	D1

Departamentos		
Código	Nome	Local
D1	RH	Campinas
D2	Vendas	São Paulo

Transformação Entre Modelos

Modelo Conceitual

DER



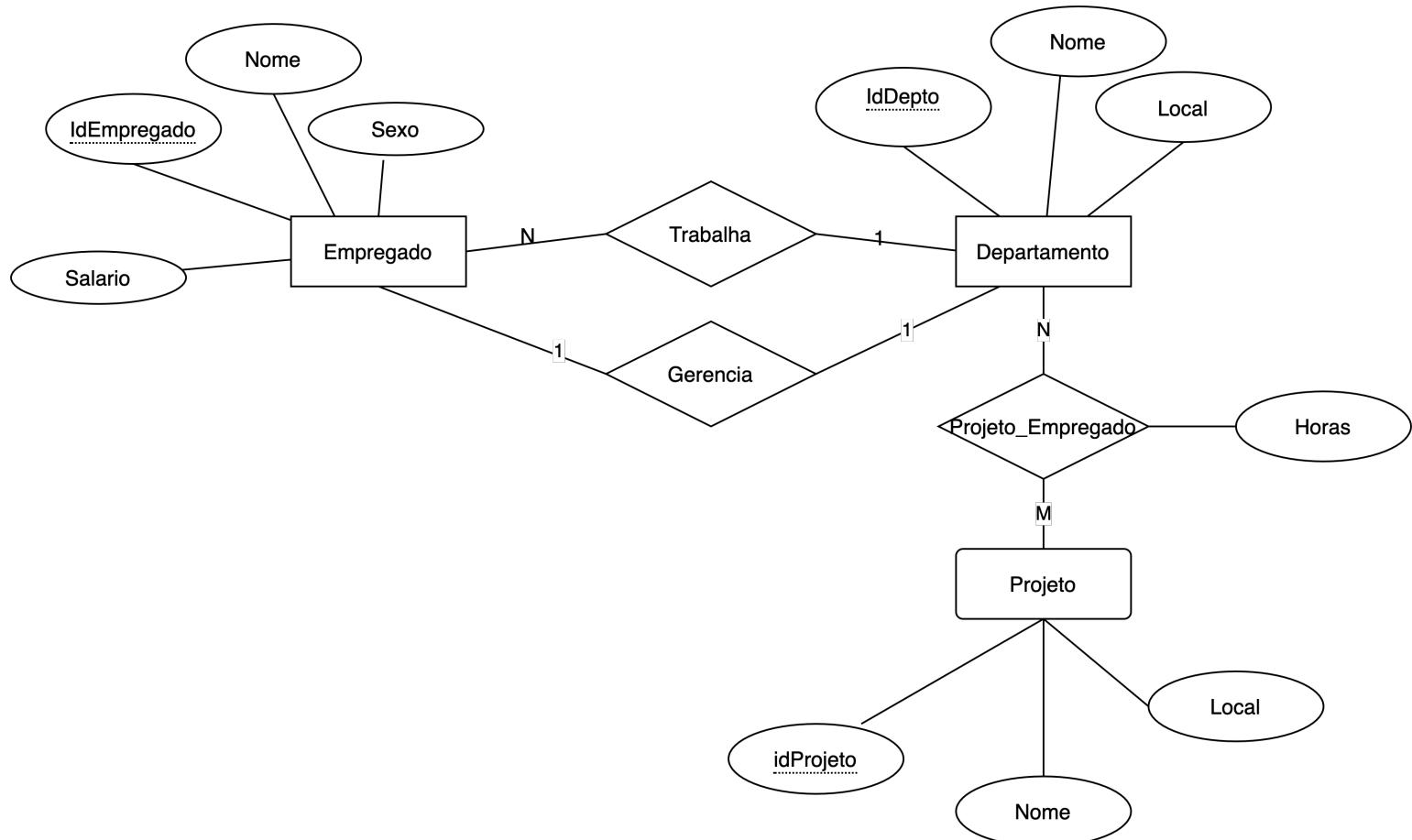
Modelo Lógico

Esquemas/Tabelas

Modelo Conceitual

A modelagem conceitual consiste no mais alto nível de abstração e deve ser usada para envolver o cliente, pois o foco é discutir os aspectos do negócio do cliente e não da tecnologia. Os exemplos de modelagem de dados vistos pelo modelo conceitual são mais fáceis de compreender, já que não há limitações ou aplicação de tecnologia específica. O diagrama de dados que deve ser construído aqui é o **Diagrama Entidade e Relacionamento**, onde deverão ser identificados todas as entidades e os relacionamentos entre elas.

Modelo Conceitual



Modelo Lógico

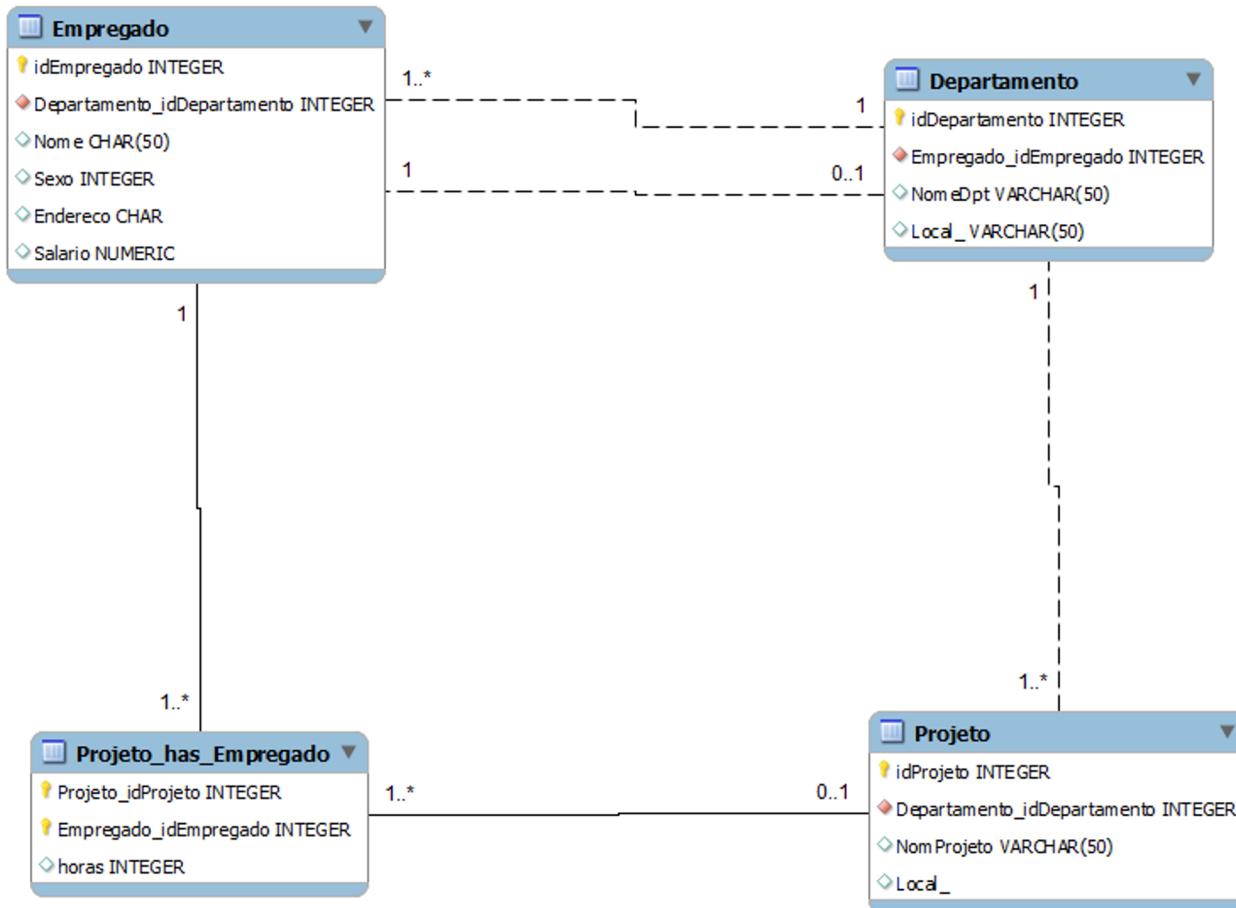
- O modelo lógico leva em conta as limitações e restrições do modelo de dados adotado. Para um banco de dados relacional, o modelo lógico deve incluir as **tabelas**, com suas respectivas **chaves primárias e estrangeiras**.

Empregado(IdEmpregado, Nome, Endereço, Salário)

Departamento(IdDpt, Nome, Local, Gerente)
Gerente Referencia Empregado

Exemplo de esquemas de tabelas em notação textual

Nível lógico/Relacional



Notação textual

Neste curso, em algumas situações, usaremos a notação textual para representar o projeto lógico de banco de dados. Nesta notação as tabelas são representadas através de um esquema:

Nome_tabela (coluna1, coluna2, ...coluna N)

- Chave primária deve ser sublinhada
- Chaves estrangeiras devem ser detalhadas nas linhas seguintes no formato: Coluna referencia Tabela
- Exemplo:

Departamento(IdDpt, Nome, Local, Gerente)
Gerente Referencia Empregado

Modelo Físico

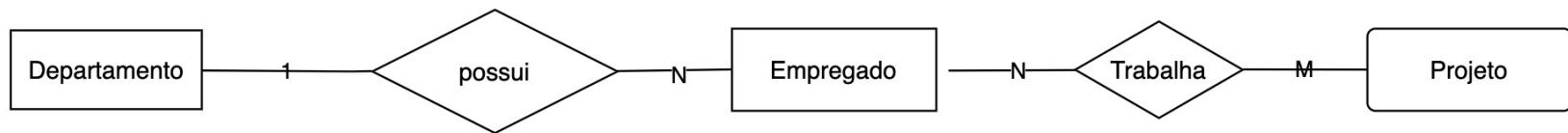
O modelo físico de banco de dados contempla a especificação de código em linguagem SQL para a criação das tabelas. Com isto, inclui a definição dos tipos de dados e tamanhos dos atributos.

```
CREATE TABLE IF NOT EXISTS Departamento(  
    idDepartamento INT NOT NULL AUTO_INCREMENT,  
    Nome VARCHAR(50) NULL,  
    Local VARCHAR(50) NULL,  
    PRIMARY KEY (`idDepartamento`);
```

```
CREATE TABLE IF NOT EXISTS Empregado(  
    idEmpregado INT NOT NULL AUTO_INCREMENT,  
    idDepartamento INT NOT NULL,  
    Nome` CHAR(50) NULL,  
    Sexo` INT(11) UNSIGNED NULL,  
    ...);
```

Mapeamento ER->Relacional

1. Para cada entidade criar uma tabela

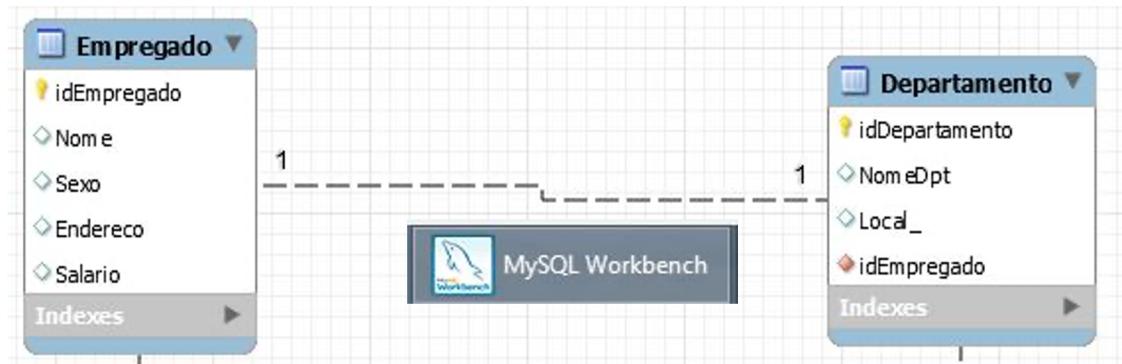


Empregado(IdEmpregado, NomeEmp, Salário, Endereço)

Departamento(IdDpt, Nome, Local)

Projeto(IdProjeto, Nome, Local)

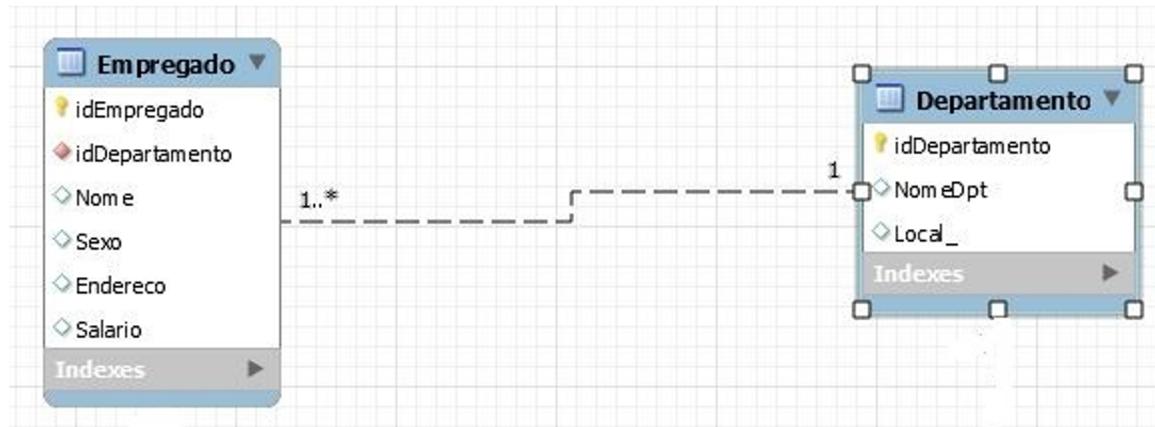
Mapeamento Relacionamentos 1:1



idEmpregado	Nome	Sexo	Endereco	Salario
123	joana Mas	F	Rua das	1200
234	Pedro Lem	M	Av 23...	3000
252	Raul Seixa	M	Rua Paraís	4320

idDepartamento	Nome	Local	idEmpregado
D1	RH	Campinas	123
D2	Vendas	São Paulo	252

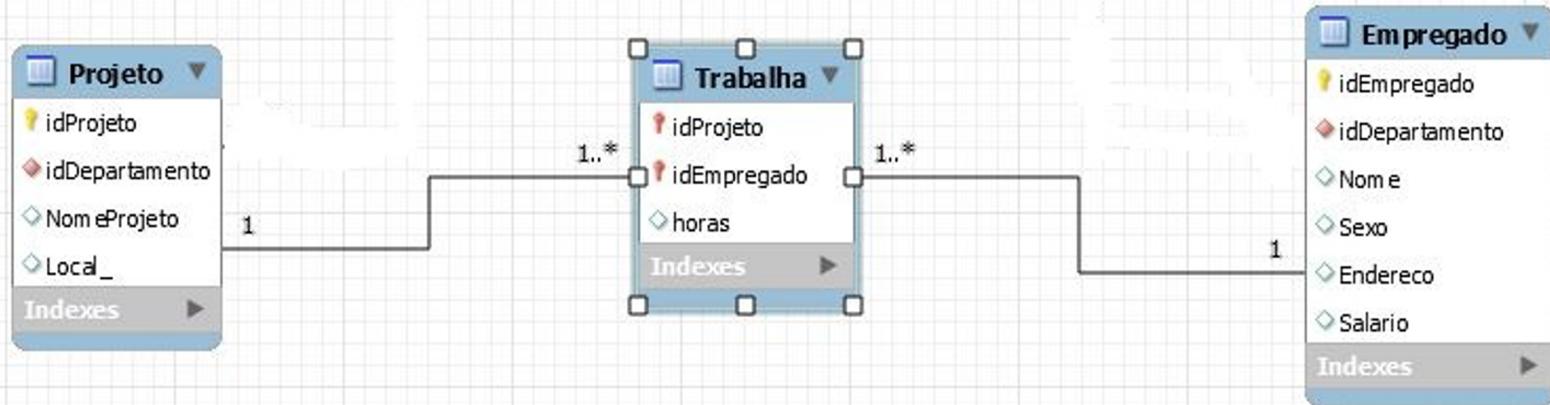
Mapeamento Relacionamentos 1:N



idEmpregado	Nome	...	idDepartamento
123	Pedro Santos	...	D1
235	Aline Sá	...	D2
252	Mario Simas	...	D1

idDepartamento	Nome	Local	Gerente
D1	RH	Campinas	123
D2	Vendas	São Paulo	252

Mapeamento Relacionamentos N:N

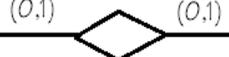


idProjeto	idDpto	NomeProjeto
ES	1	Engenharia...
BD	1	Banco de Dados
GEO	2	Geoprocessam..
BI	1	Bussiness

idProjeto	idEmpregado
ES	121
BD	212
GEO	312
GEO	121
BD	121

idEmpregado	idDpto	Nome
121	1	Bianca
212	2	Jorge
312	1	Mauro
150	2	Talita

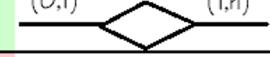
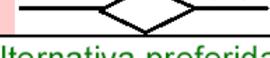
Implementação de relacionamentos 1:1

Tipo de relacionamento	Regra de implementação		
	Tabela própria	Adição coluna	Fusão tabelas
$(0,1)$ 	±	✓	✗
$(0,1)$ 	✗	±	✓
$(1,1)$ 	✗	✗	✓

✓ Alternativa preferida

± Pode ser usada

✗ Não usar

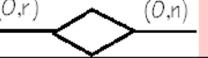
Tipo de relacionamento	Regra de implementação		
	Tabela própria	Adição coluna	Fusão tabelas
$(0,1)$ 	±	✓	✗
$(0,1)$ 	±	✓	✗
$(1,1)$ 	✗	✓	✗
$(1,1)$ 	✗	✓	✗

✓ Alternativa preferida

± Pode ser usada

✗ Não usar

Relacionamentos n:n

Tipo de relacionamento	Regra de implementação		
	Tabela própria	Adição coluna	Fusão tabelas
$(0,r)$ 	✓	✗	✗
$(0,r)$ 	✓	✗	✗
$(1,n)$ 	✓	✗	✗

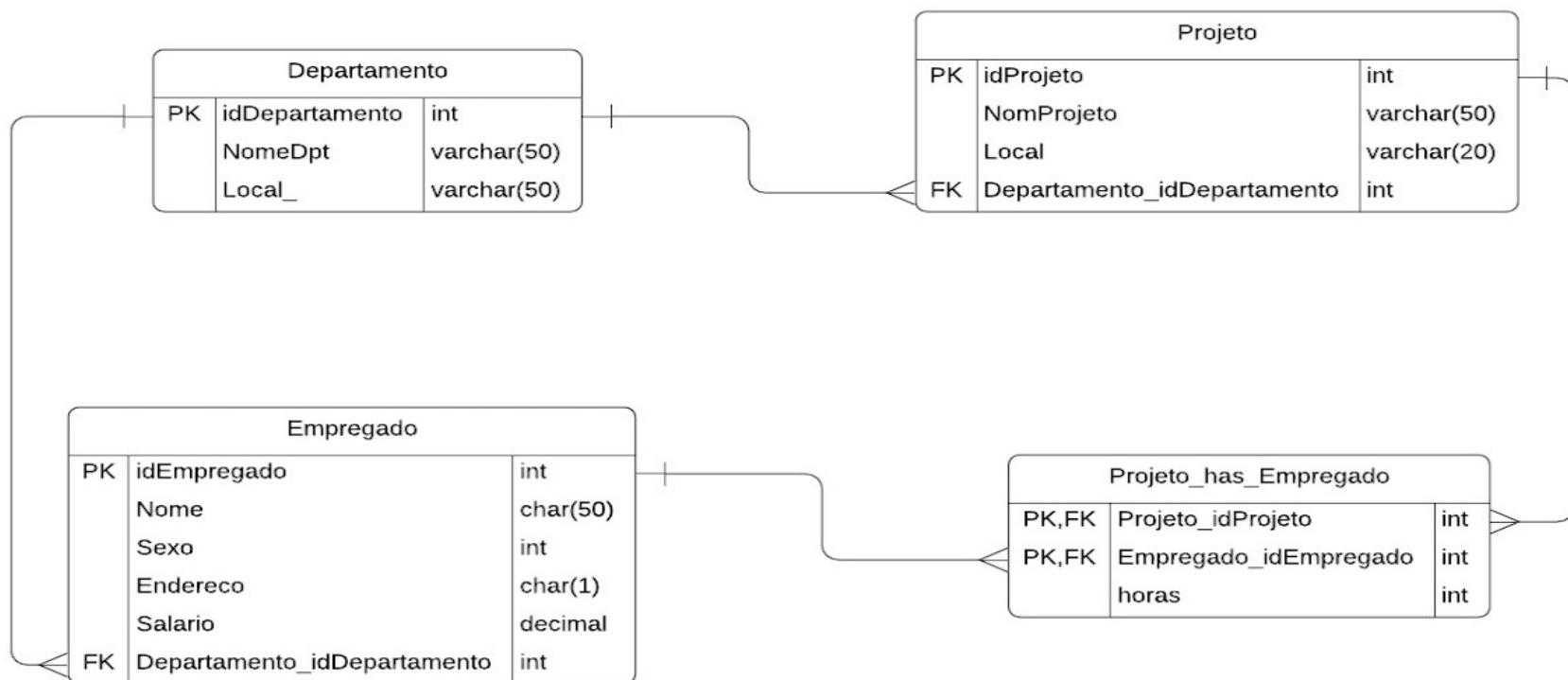
✓ Alternativa preferida

✗ Não usar

Resumo das Regras de Mapeamento ER-> Relacional

1. Para cada entidade criar uma tabela
2. Para cada relacionamento 1:1 juntar tabela ou adicionar FK
3. Para cada relacionamento 1:N adicionar FK
4. Para cada relacionamentos N:N criar uma tabela com chaves primárias compostas

Diagrama do Lucid.app

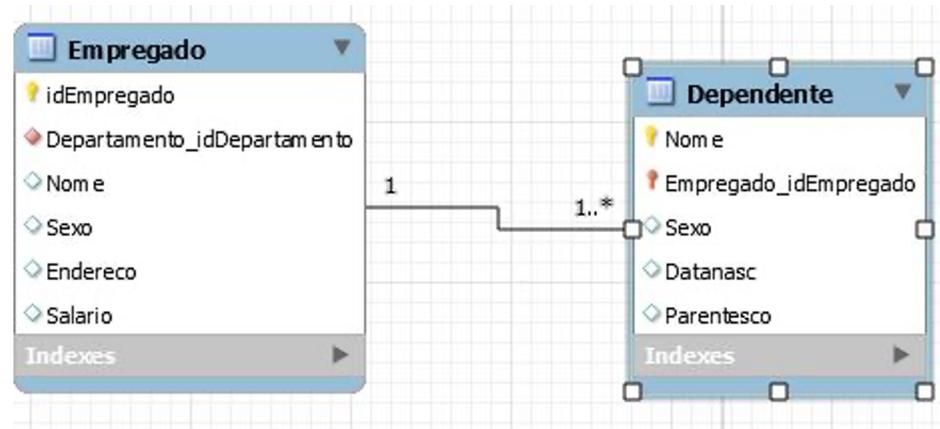


Casos especiais

Entidade fraca

- Não tem atributo chave
- Tem um relacionamento de identificação com uma entidade. Num relacionamento de identificação, a entidade fraca recebe a chave primária da outra entidade (a entidade forte deste relacionamento) como chave estrangeira e esta fará parte ou será toda ela chave primária da entidade fraca

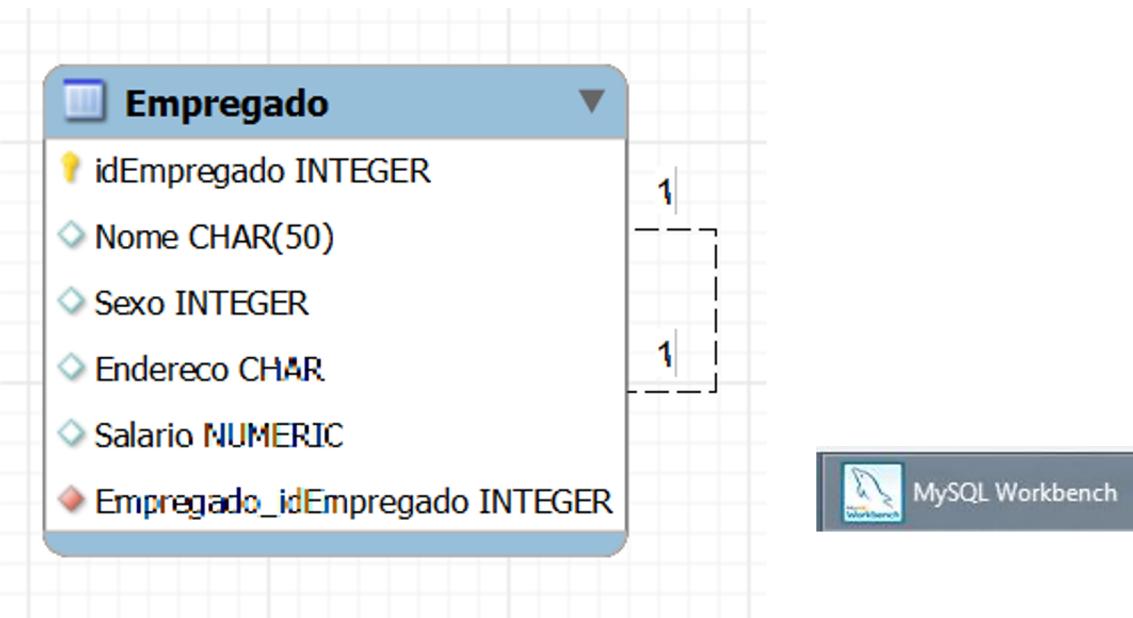
Exemplo Entidade Fraca



No Workbench relacionamentos de identificação são representados pela linha contínua.

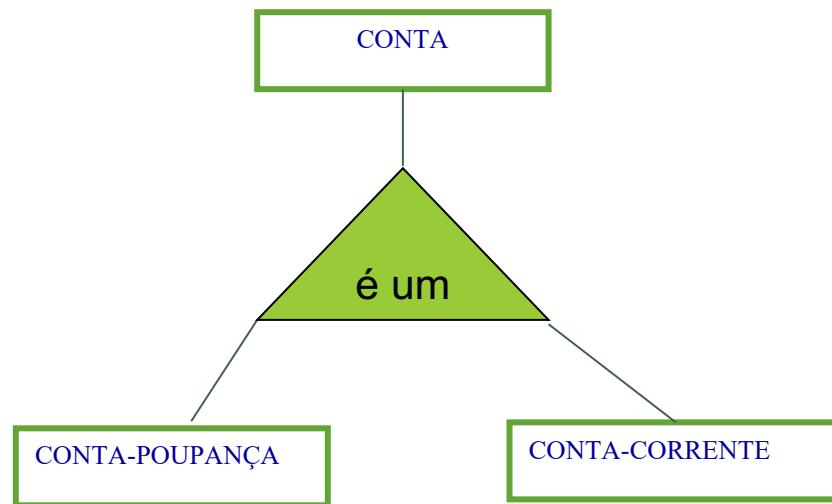
Auto-relacionamento

- Relacionamento de uma entidade com ela mesma
- Exemplo Empregado supervisiona empregado



Generalização

Expressa a semelhança entre entidades através de um relacionamento de herança



Generalização/Especialização

Para casos de Generalização/Especialização duas soluções podem ser adotadas:

- Criar uma relação para cada entidade. As relações correspondentes as entidades não principais contêm a chave da relação principal
- Criar relações apenas para as entidades não principais.

Generalização/Especialização

Solução 1:

Conta(idConta, Saldo)

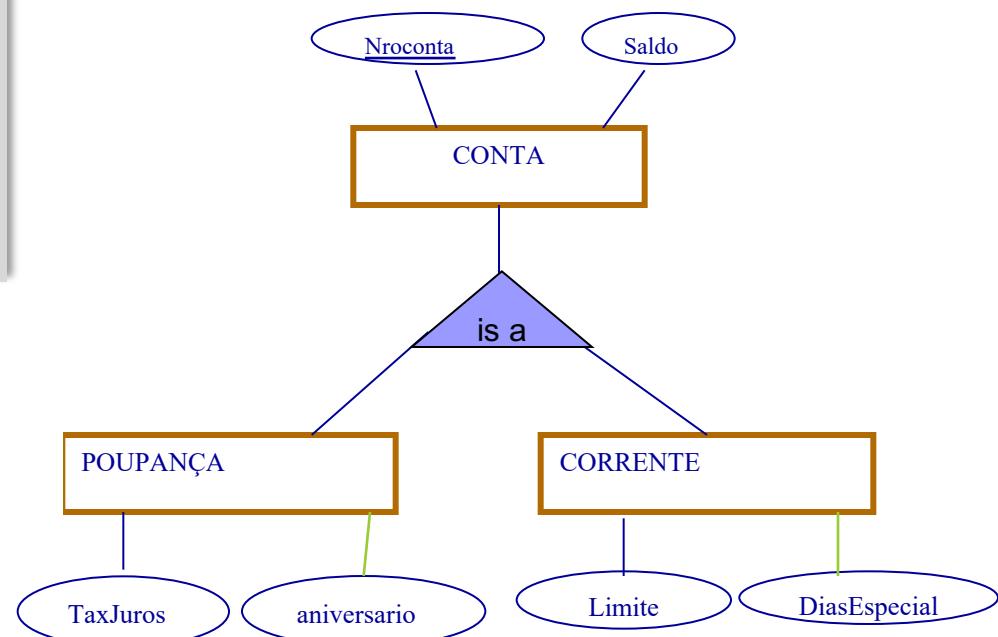
Poupanca(idConta, Txjuros, aniversario)

Corrente(idConta, Limite, DiasEspecial)

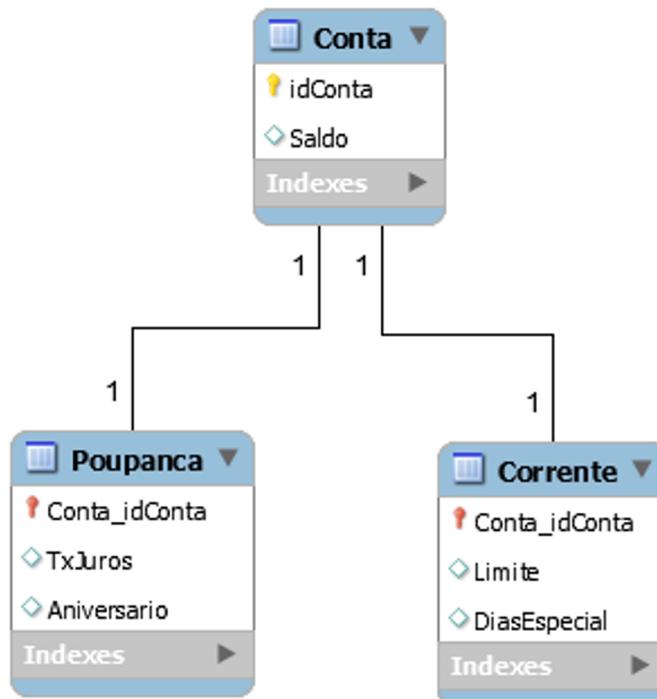
Solução 2:

Poupanca(idConta, Saldo, Txjuros, aniversario)

Corrente(idConta, Saldo, Limite, DiasEspecial)



Generalização em Workbench



idConta	Saldo
123123	1000.00
123111	5000.54
001987	3000.01
12345	1234.00

idConta	TxJuros	Aniversario
123123	1%	1
123111	1.02%	10

idConta	Limite	DiasEspecial
001987	3000.00	1
12345	10000.00	0

Bibliografia

Heuser C. A, Projeto de Banco de Dados, 3a ed, Editora Sagra Luzzatto, 2000.

Elmasri & Navathe, Sistemas de Banco de Dados Fundamentos e Aplicações, 3a. ed., LTC, 2002.

Korth, h.; Silberschatz, A. Sistemas de Banco de Dados. Makron, 3a ed. 1999.

Date, C. J., Introdução a Sistemas de Banco de Dados. Campus, 7a ed. 2000.

Oliveira, C.H. P. SQL Curso Prático, Novatec, 2002.

Patrick, J.J. SQL Fundamentos, Berkeley Brasil, 2002.

Taylor, A. G. SQL para Dummies, Campus, 2001.