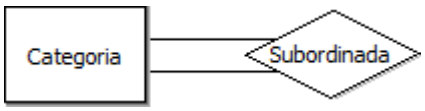
Resumo P1: Abordagem Entidade-Relacionamento

- **Associação entre entidades**
- **Atributo de uma entidade que se refere a outra**

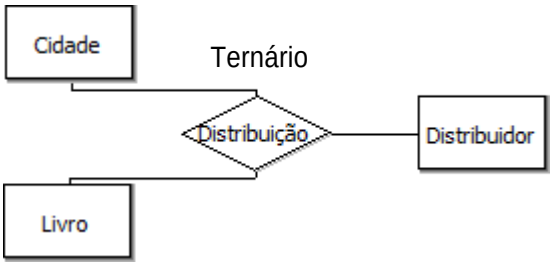
- **Auto Relacionamento:** Relacionamento entre ocorrências da mesma entidade



Ficção --- Subordinada --- Romance

- **Grau de Relacionamento:** Binário

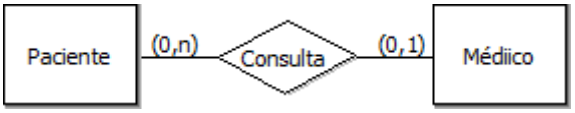
Ternário



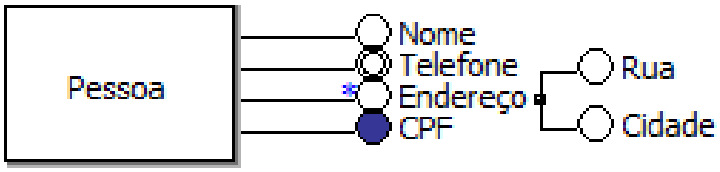
- **Cardinalidade:** Podendo ser (0,n) = Nenhuma ou várias

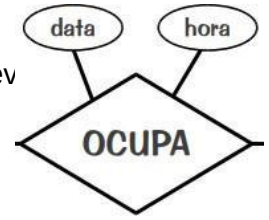
(0,1) = Nenhuma ou apenas uma (1,n) = Apenas uma ou várias (1,1) = Uma e apenas uma

Pode-se variar entre diferentes valores específicos com a mesma lógica, porém essas são os mais comuns



- **Atributos:** Cada instância de entidade ou relacionamento tem atributos que a descrev





- Simples (Atômico)
- Multivalorado
- Composto
- Chave (únicos na relação), se houver mais de um, se torna Atributo-Chave Composto

- **Entidade Fraca:** Quando uma entidade depende de outra para existir

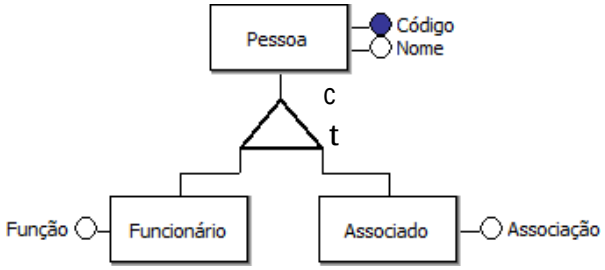
- **Generalização/Especialização:** Divide uma entidade em mais partes, com diferentes atributos. Além de termos:

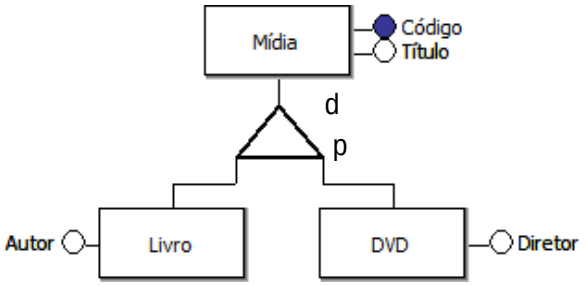
Compartilhada ou Superposta: A Pessoa pode ser ambos

Exclusiva ou Disjunta: Se é um, não pode ser o outro

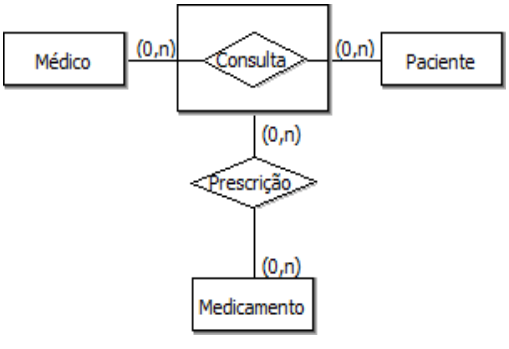
Total: Todas as pessoas são Funcionários ou Associados

Parcial: Nem todas as mídias são livros ou DVDs





- **Entidade Associativa:** A relação pode se comportar como uma entidade



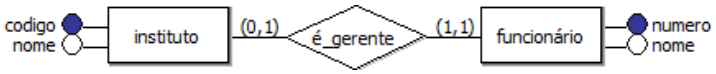
Projeto Lógico - Modelo Relacional

- Mapeamento Entidades Fortes:

emprestimo(<u>numero emprestimo</u>, quantia)

- Relacionamento: É implementado com chave estrangeira ou uma nova tabela em relacionamentos N para N

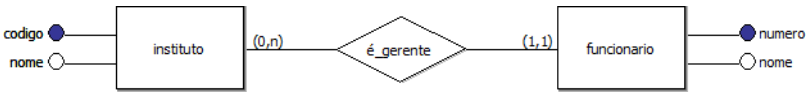
1)Relacionamento (0,1) para 1, a chave é inserida na tabela (0,1) e deve ser setada como unica (UNIQUE)



instituto(<u>codigo</u>, nome, numFunc)
funcionario(<u>numero</u>, nome)
numFunc referencia Funcionario(numero)

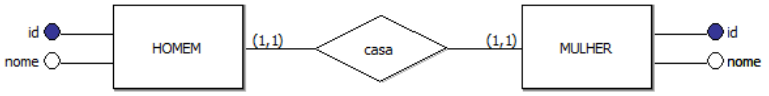
2)Relacionamento 1 para N, a chave é inserida na tabela (0, N)

instituto (<u>código</u>, nome, num_gerente)
num_ gerente referencia funcionario
funcionario (<u>numero,</u> nome)



3) Relacionamento 1 para 1, deve se escolher uma da tabelas para ser incluso a chave estrangeira, porém essa chave deve ser setada como(UNIQUE) e NOT NULL

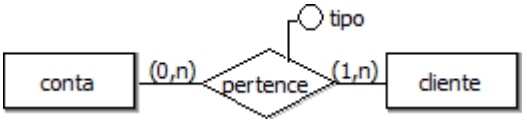
homem (<u>id</u>,nome, id_mulher)
id_mulher referencia mulher(id)
mulher (<u>id</u>, nome)



4) N para N + com atributo

conta(<u>num</u>,saldo,...)
cliente(<u>codigo</u>,nome,...)
(linha de baixo) *

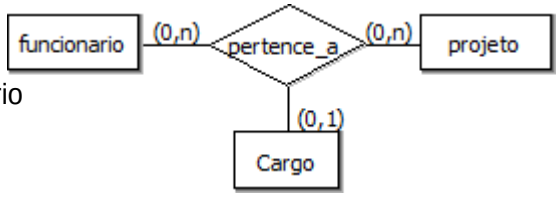
*conta_cliente(<u>codigo</u>,<u>num,</u>**tipo**)
codigo referencia cliente
num referencia conta



5) *Relacionamento N-ário: Relação U para o relacionamento*

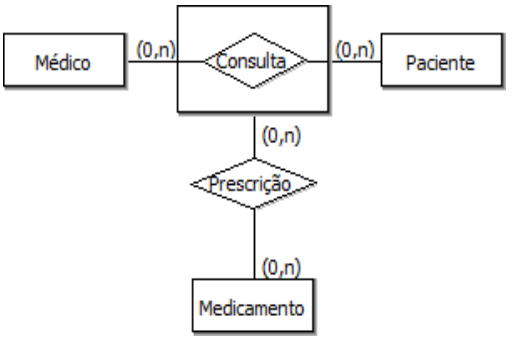
projeto (<u>num</u>, título,...)
funcionário (<u>codigoF</u>, nome,...)
cargo (<u>codigoC</u>, nome,...)*

*trabalho(<u>num, codigoF</u>, <u>codigoC</u>)
num referencia projeto
codigoF referencia funcionario
codigoC referencia cargo



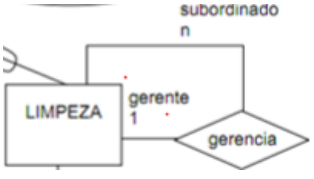
6) Relacionamento com entidade associativa, exemplo:

medico(<u>id</u>, nome, cargo)
paciente(<u>id</u>, nome, telefone)
consulta(<u>id_medico, id_paciente</u>)
id_medico referencia medico(id)
id_paciente referencia paciente(id)
medicamento(<u>id</u>, nome)
prescricao(<u>id_medico</u>, <u>id_paciente, id_medicamento</u>)
id_medico referencia medico(id)
id_paciente, id_medicamento referencia consulta(id_medico, id_paciente)



7) *Auto-Relacionamento:*

limpeza(<u>id</u>, subordinado)
subordinado referencia limpeza(id)



CARDINALIDADE DE RELACIONAMENTOS, OBSERVAÇÕES:

*Para relacionamentos com cardinalidade (0, 1) no projeto físico deve-se colocar unique

*Para relacionamentos com cardinalidade (1, 1) no projeto físico deve-se colocar unique e not null

*Para relacionamentos N para N cria-se uma nova tabela

*Para relacionamento 1 para N a foreign key deve fica na tabela onde a cardinalidade é N

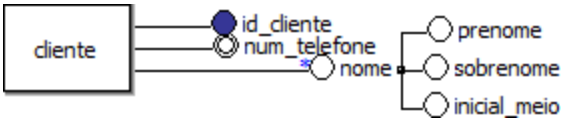
-O unique defini que só deve ter uma chave unica na foreign key[(0,1) ou (1, 1)] e o not null defini que não poderá ter valor vazio na foreign key, ou seja (1, 1)

- Mapeamento de Atributos Compostos e Multivalorados:

cliente (id_cliente, prenome, inicial_meio, sobrenome)

telefone (id_cliente, num_telefone)

id_cliente referencia cliente

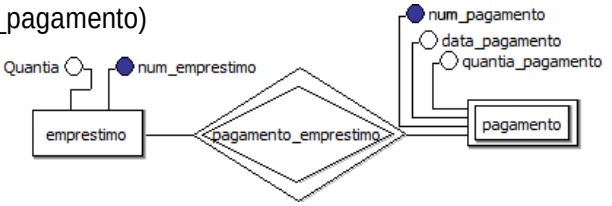


- Mapeamento de Entidades Fracas: É representado como uma relação

pagamento(num_emprestimo, num_pagamento, data_pagamento, quantia_pagamento)

num_emprestimo referencia empréstimo (num_emprestimo)

*no projeto físico é colocado ON DELETE CASCADE



- Mapeamento Especialização e Generalização:

1) Criar relação para cada entidade(não disjuntas/parciais)

pessoa (id_pessoa, nome, rua)

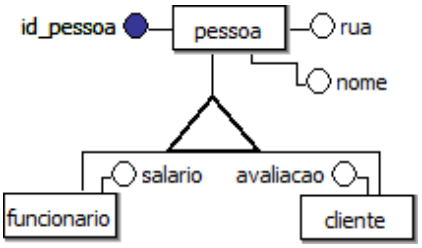
funcionario (id_pessoa, salario) id_pessoa referencia pessoa(id_pessoa)

cliente (id_pessoa, avaliacao) id_pessoa referencia pessoa(id_pessoa)

2) Criar tabelas somente para entidade de nível inferior

cliente (id_pessoa, nome, rua, avaliacao) id_pessoa referencia pessoa(id_pessoa)

funcionario (id_pessoa, nome, rua, salario) id_pessoa referencia pessoa(id_pessoa)



3) Única tabela com todos os atributos das entidades envolvidas + atributos tipo

pessoa (id_pessoa, nome, rua, avaliacao, salario, **tipo**) id_pessoa referencia pessoa(id_pessoa)

3.5) *Semelhante ao anterior, porém com atributo booleano, indicando a entidade específica* ISA

pessoa (id_pessoa, nome, rua, **f_tipo**, avaliacao, **c_tipo**, salario)

id_pessoa referencia pessoa(id_pessoa)

Projeto Físico – SQL

DML: Tupla (Insert, Delete, Update, Select)

Linguagemde BD: SQL "//" = Linha de baixo

SGBD: Postgres **DDL: Esquema** (Create, Drop, Alter)

DDL:

- Comando Create: Criar elementos do esquema de banco de dados (Banco de Dados e/ou Tabela)

createdatabasenomeDoBanco/ createdatabaseempresa (nomeDoBanco: identificador debasededados)

Tabela: *create table* ***nomeTabela(// nomeColuna1 domínioDaColuna1,// nomeColuna2 domínioDaColuna2,// ...)***

- Comando Drop: Remover elementos do esquema de banco de dados (Banco de Dados e/ou Tabela)

drop database nomeDoBanco / drop database empresa

- *Tipos (padrão SQL):* *char(n):* string tamanho fixo (*varchar(n)* é tamanho variado)

int: inteiro (*smallint* é inteiro pequeno)

numeric(p,d): Ponto físico, com precisão de p digitos *date:* Tipo para data, formato YYYY – MM - DD

float (n): Ponto flutuante (*real, double precision* além de float, também é ponto flutuante de precisão dupla)

- Restrições: *create table* ***r (A1D1 restrição-de-integridade1, // ..., // restrição-de-integridadek)***

- Primary Key: Chave primária - *create table cliente (// matricula char(5)* ***primary key****)*

- Restrições de Vazio: *create table cliente (// nome varchar(50)* ***not null****)*

- Chave Candidata: *create table cliente (// identidade char(7) not null* ***unique****)*

- Chaveprimáriae Candidatacompostas: *(...//****primarykey(codigoLetra, codigoNum),//unique(nome, sobrenome)****)*

- Restrições de Domínio: *create table cliente (// email varchar(100)* ***default 'desconhecido'****)*

- Check: *createtable cliente (// idade int not null, //* ***check (idade>0)*** ou *(idade int not null* ***check (idade>0)****)*

createtablecliente(// ***tipo enum('Especial', 'Ouro','Premium')*** (enumpode ser nula,mas se não for notnull enãopreenchida, éatribuído oprimeirovalor)

... (// tipo varchar(10) not null, check ***(tipo = 'Especial' OR tipo = 'ouro' OR tipo = 'Premium')****)*

...(// dataContratacao date, // dataInicioGer date, // ***check (dataContratacao > dataInicioGer)****)*

- **Chave Estrangeira:** *create tabel conta(//...// nome_agencia char(15) not null,*

foreign key *(nome_agencia)* ***references*** *agencia (nome))* ||

*createtableagencia(// nomechar(15) primarykey) // createtableconta(//...//****foreign key*** *(nome_agencia)* ***references*** *agencia(nome))*

... ***foreign key*** *(nome_agencia)* ***references*** *agencia (nome))* ***on delete*** (ou Update) ***cascade***

- Modificar Tabela (Alter table): Adicionar ou remover atributo/restrição, além de alterar atributo

alter table nomeTabela ***add*** *columm* (opcional) *nomeAtributo tipoAtributo* **=** *alter table agencia* ***add*** *tel varchar(15)*

alter table nomeTabela ***drop*** *columm nomeAtributo* **=** *alter table agencia* ***drop*** *cidade*

alter tablenomeTabela ***alter*** *colummnomeAtributonovaDefinição* **=** *altertableagencia* ***alter*** *colummtelvarchar(30) notnull*

Esse é o padrão SQL, em MYSQL usa-se ***modify*** ao invés de ***alter***, em Postgresql coloca ***set data type*** antes de varchar

- Modificar Restrições: *alter table nomeTabela add* ***constraint*** *nomedarestrição RESTRIÇÃO* **=***...constraint contaNum Primarykey (num)*

Chave candidata: *... add constraint nomeConstraint* ***unique*** *(nomeColuna)*

Check: *alter table* ***Conta*** *add constraint* ***conta_saldo_check*** *check (****saldo****>0)*

Apagar restrição: *altertablenomedaTabela* ***drop*** *constraintnomeRestrição* **=** *altertable Conta* ***drop*** *constraint pkNum*

Apagar tabela: ***drop*** *table nomeTable* **=** ***drop*** *table agencia*

- Comentários: -- ou **/*** ... ***/**

Para mais informações, acessar o PDF "SQL-DDL"

create table cliente (no_cliente int primary key, nome varchar(150) not null, CNPJ varchar(5))

- Inserir uma tupla

insert into cliente values (1,'Maria','11111')

- Inserir mais de uma tupla

insert into cliente values (1,'Maria','11111') , (2,'Joao','22222')

DML:

- Inclusão Tupla: Insert – ***insert*** *into* 'nome da relação' (lista de atributos) values (valores) – Os valores devem seguir a mesma ordem com que os atributos foram criados no comando create

- Exclusão de Tuplas: Delete – ***delete*** *from cliente*

- Cláusula Where: Define uma expressão da qual identifica as tuplas que devem ser consideradas (>,<,>=,<=,<>,=)

where *id = 3* / ***where*** *id > 4* / ***where*** *nome <> 'Joao'*

- Exclusão de Tuplas com condição: ***delete*** *from tabela//* ***where*** (condição) **=** ***delete*** *from cliente//* ***where*** *nome='Maria'*

- AND e OR: *delete from cliente // where nome = 'Maria'* ***and*** (ou então ***or***) CNPJ = '11111'

- Between: *delete from cliente // where no_cliente* ***between*** *1 and 10* (seria "entre 1 e 10")

- Update: Atualização de dados – ***update*** *nome_tabela*

set coluna1 = valornovo, coluna2 = valornovo, ..., colunan = novovalor

update *cliente // set cnpj= '00000'*

- Atualização com Condição: ***update*** *cliente // set cnpj= '00000' //* ***where*** *no_cliente = 1* (**where** condição)

Exemplos:

```
create table conta
(
numero char(5) primary key,
saldo numeric(2,1) not null,
nome_agencia char(15) not null,
foreign key (nome_agencia) references agencia
(nome)
)
```

add restrição

```
alter table departamento alter column nome
set not null
```

setar campo como unique

```
alter table funcionario add unique(identidade)
```

atributo multivalorado chave composta

```
create table telefone(
codf int not null,
numtel varchar(15) not null,
primary key(codf,numtel),
foreign key(codf) references funcionario(codigo)
)
```

datafim > datainicial

```
alter table projeto add constraint
projeto_datafim_datainicial check(datafim > datainicial)
```

apagar atributo

```
alter table alocacao drop column
datai
```

adicionar atributo

```
alter table alocacao add column
datai date
```

adicionar atributo com check

```
alter table alocacao add qtdHora int not null check(qtdHora > 0);
```

adicionar atributo

```
alter table projeto add custo float not null;
```

adicionando chave estrangeira na tabela

```
alter tableempregadoaddconstraintcargo_fkforeignkey(codCargo)
references cargo(codigo) on delete set null;
```

Inserção

```
insert into departamento values (1,'TI'),(2,'Qualidade');
insert into funcionario values
(1,'Bruno','222222','1134644646','bruno@gmail.com','1'),
(2,'Felipe','3333333','5534644646','felipe@gmail.com','1'),
(3,'Carlos','4444444','6634644646','carlos@gmail.com','1'),
(4,'Pedro','5555555','7734644646','pedro@gmail.com','2'),
(5,'Ana','6666666','8834644646','ana@gmail.com','2'),
(6,'Bia','7777777','9934644646','bia@gmail.com','2');
```

atualizar

```
update funcionario set nome='Bruno Moraes' where cpf='1134644646';
```

```
update alocacao set qtdhora=30 where codf<>7; //atualizar onde codf for
diferente de 7
```

```
delete from departamento where codigo=1;
```