



# Banco de Dados Unidade – I e II

Professor Adilson da Silva

# Objetivos



- Conceitos de bancos de dados
- Modelo Relacional
- Normalização de Dados

- Modelos de banco de dados
- Esquema relacional de um banco de dados

- Modelagem de banco de dados – modelo conceitual
- Modelo Entidade-relacionamento
- Arquitetura de Três níveis





# Conceitos de Banco de dados



# Conceitos



- Diferença entre Informação e Dado
  - **Informação:** é qualquer fato ou conhecimento do mundo real e que pode ou não ser registrado /armazenado;
  - **Dado:** é a representação da informação, que pode estar registrado em papel, num quadro de aviso ou no disco rígido do computador
  - Exemplo:
    - Informação: Está muito quente hoje
    - Dado: A temperatura hoje é de 38 graus Celsius
  - O computador armazena e processa dados e não informações





# Banco de Dados

---

- Mas o que é um banco de dados?
- Entre os vários conceitos encontrados, podemos adotar o seguinte:
  - **Dados são fatos que podem ser gravados e que possuem um significado implícito [Elmasri e Navathe, 2005].**
  - **De uma forma simplista, podemos dizer que um banco de dados consiste em uma coleção de dados estruturados, organizados e armazenados de forma persistente [Damas 2007].**

O nome banco de dados é atribuído a um **conjunto de dados agrupados em uma estrutura regular** que permite o **acesso de maneira organizada** e normalmente mantido por um sistema computacional que o gerencie.



# Conceitos

---



- É uma coleção de dados relacionados
- O uso do termo é mais restrito em virtude das seguintes características:
  - **Um BD representa algum aspecto do mundo real, o qual chamamos de *Minimundo* ou *Universo de Discurso***
  - **É um conjunto lógico e ordenado de dados que possuem algum significado inerente**
  - **Um BD é projetado, construído e povoado com dados que possuem objetivos específicos**

- Ingredientes necessários em um BD:
  - Uma fonte de dados da qual derivamos os dados
  - A interação com o mundo real
  - Público que demonstra interesse nos dados contidos no Banco

# Aplicações de Banco de Dados



- Bancos de dados estão presentes em todas as áreas da Informática.
  - Atualmente, não se imagina Informática sem bancos de dados!
- Mas, para entendermos a importância dos bancos de dados, é preciso voltar no tempo.
  - Precisamos lembrar os problemas dos sistemas anteriores ao surgimento dos bancos de dados.
  - Esses sistemas guardavam informação em arquivos.



# Aplicações de Banco de Dados



- Nos sistemas baseados em arquivos:
  - a aplicação era escrita em uma linguagem de programação
  - e acessava diretamente um arquivo físico da máquina.
- Essa abordagem funcionava bem, contanto que:
  - apenas um usuário acessasse os dados do arquivo;
  - a quantidade de dados fosse pequena.

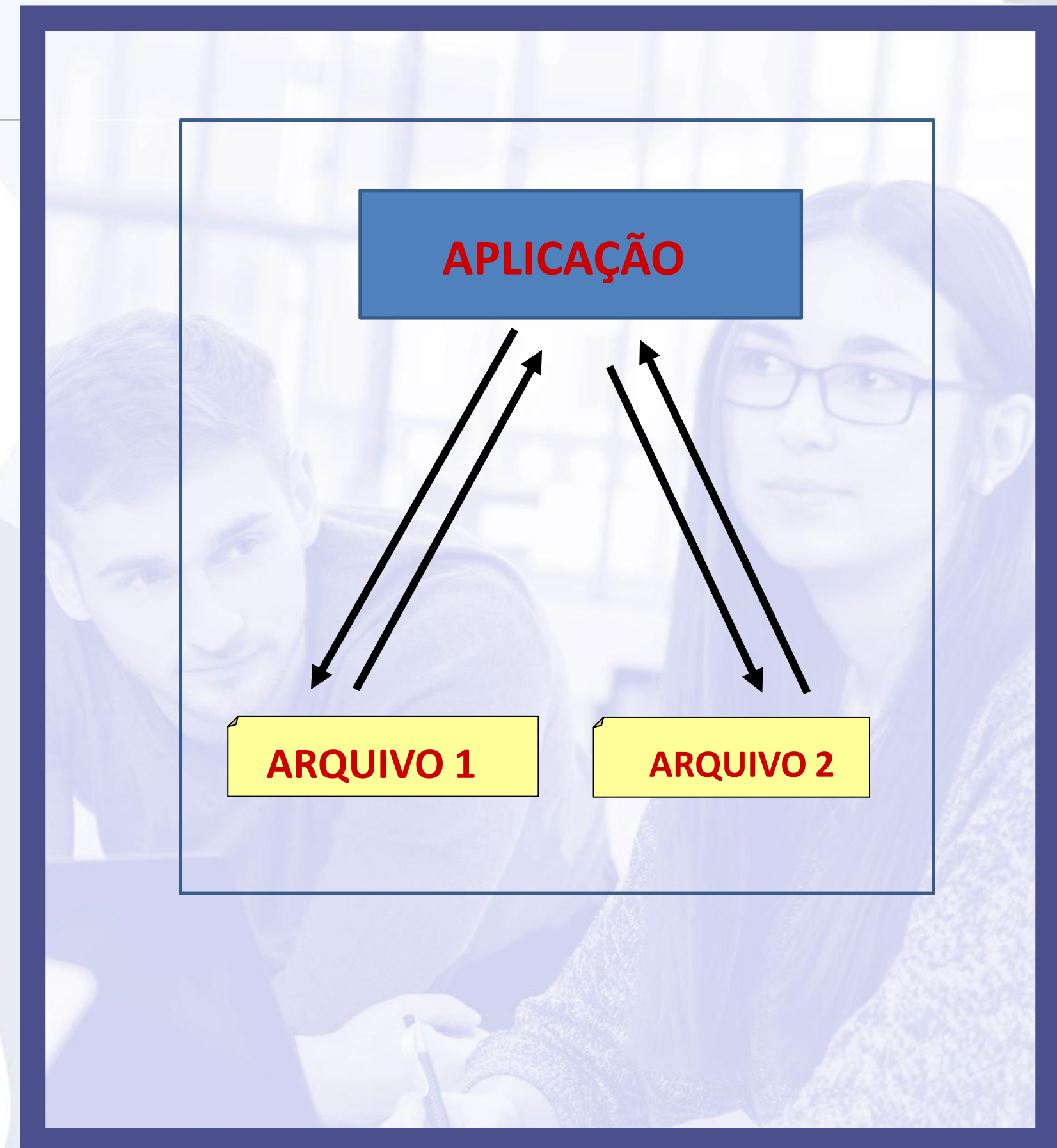




# Aplicações de Banco de Dados



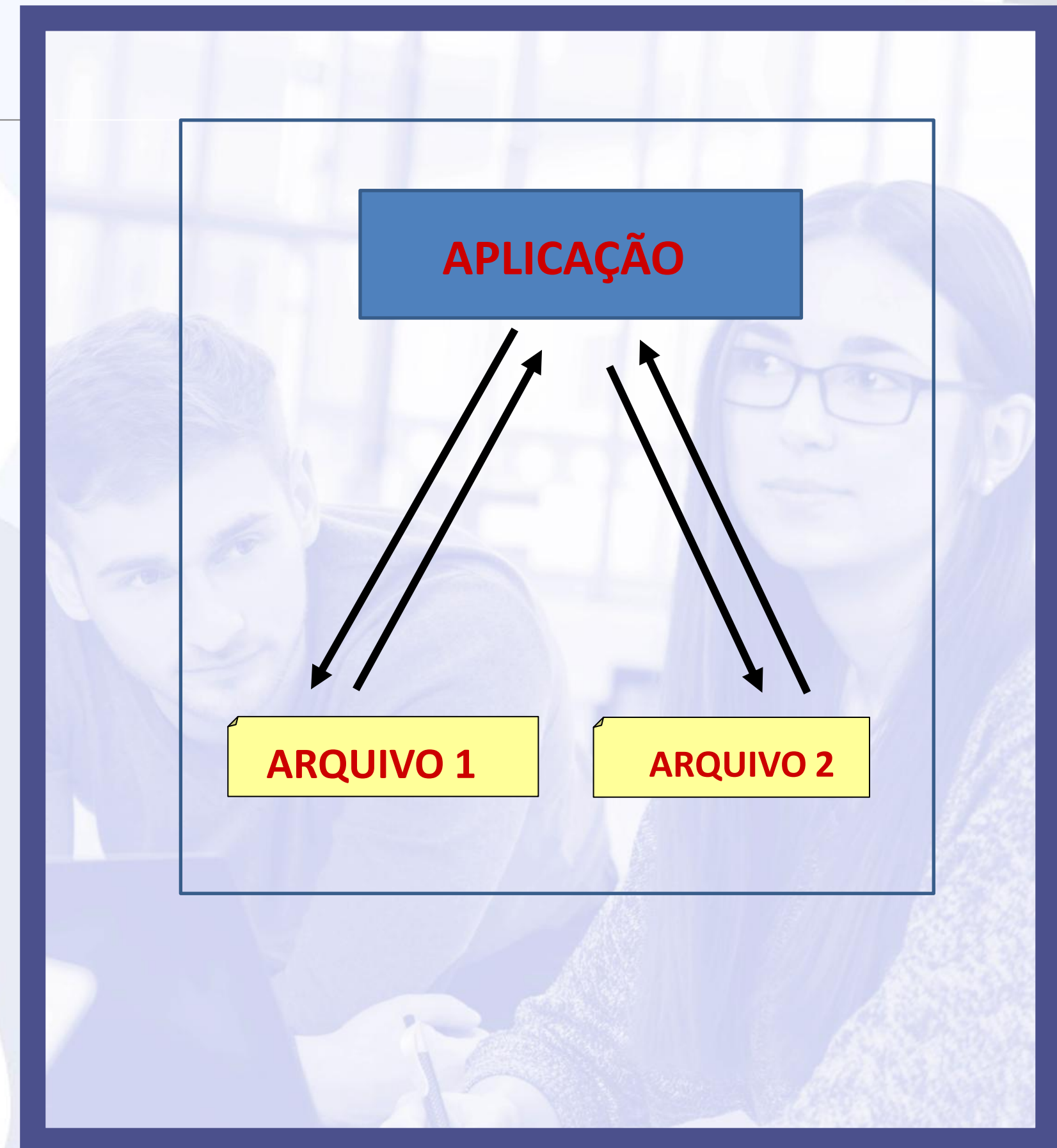
- À medida que os sistemas cresciam, uma mesma aplicação passava a gerenciar mais de um arquivo de dados.
- Ex.: Criava-se um novo arquivo quando o anterior não comportava o formato dos novos dados a serem acrescentados.
- Nesse caso, a aplicação era atualizada para enxergar o novo arquivo:
  - O código-fonte era alterado e recompilado.
  - Isso gerava um custo de manutenção do sistema.



# Redundância e inconsistência



- Além disso, acontecia de uma mesma informação ser gravada em mais de um arquivo (p. ex., nome do cliente).
- Esse problema é chamado de **redundância de dados**.
- A repetição é desnecessária e desperdiça espaço em disco.
- Outro tipo de problema ocorria quando o sistema, por algum erro, tornava uma informação incompatível com outra.
  - **Esse problema é chamado de inconsistência de dados.**
  - **A integridade dos dados fica comprometida.**

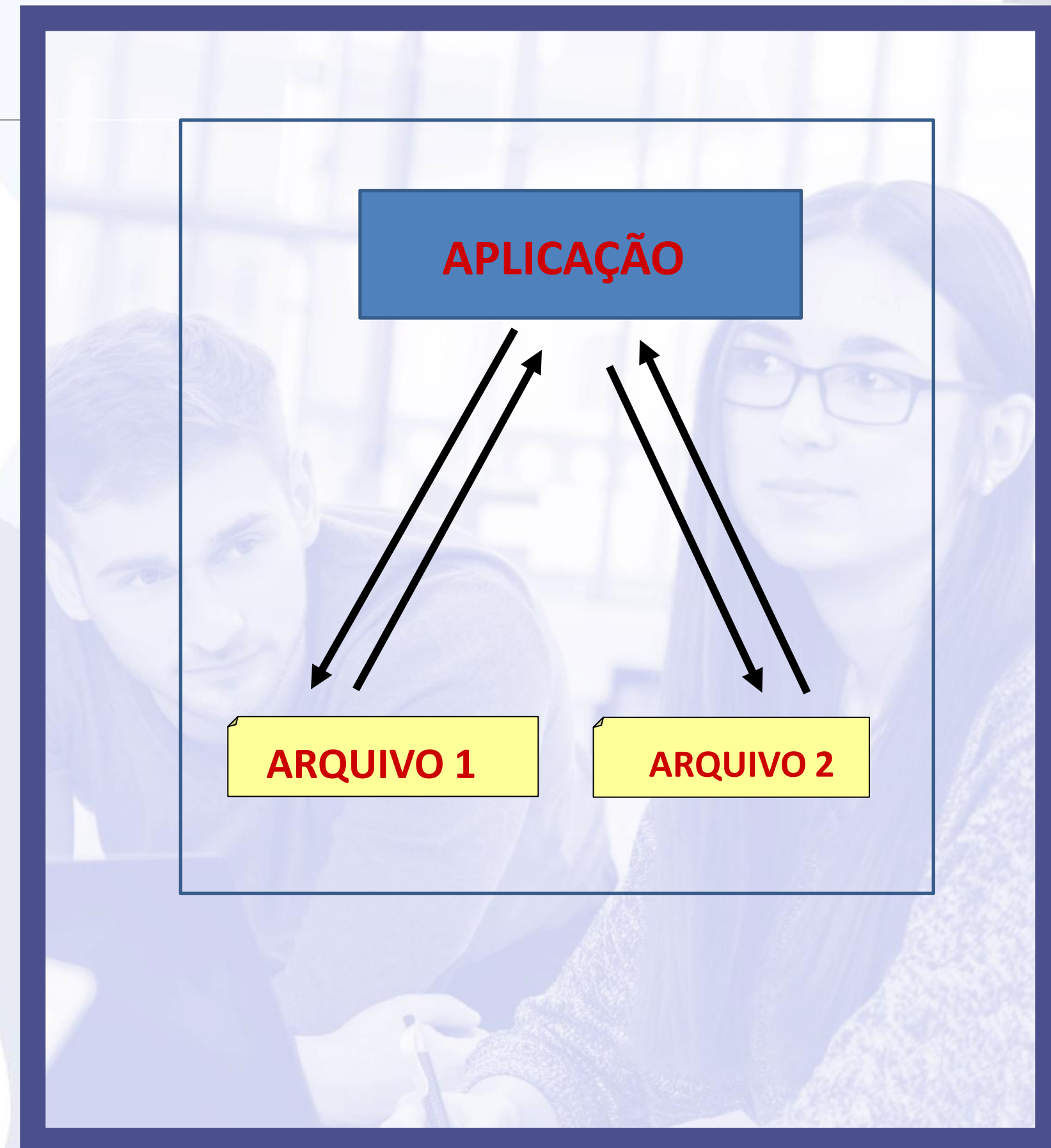




# Concorrência e Segurança



- **Dificuldade de controlar concorrência:**
  - É difícil controlar arquivos físicos quando vários usuários estão acessando ao mesmo tempo.
    - Ex.: Em uma companhia aérea, dois atendentes tentando reservar a mesma poltrona em um mesmo voo.
- **Dificuldade de controlar segurança:**
  - Com arquivos físicos, é difícil fornecer níveis diferentes de acesso para usuários distintos.
    - Ex.: O presidente da empresa pode ver dados de todos os funcionários, mas um gerente só pode ver os dados dos seus subordinados.

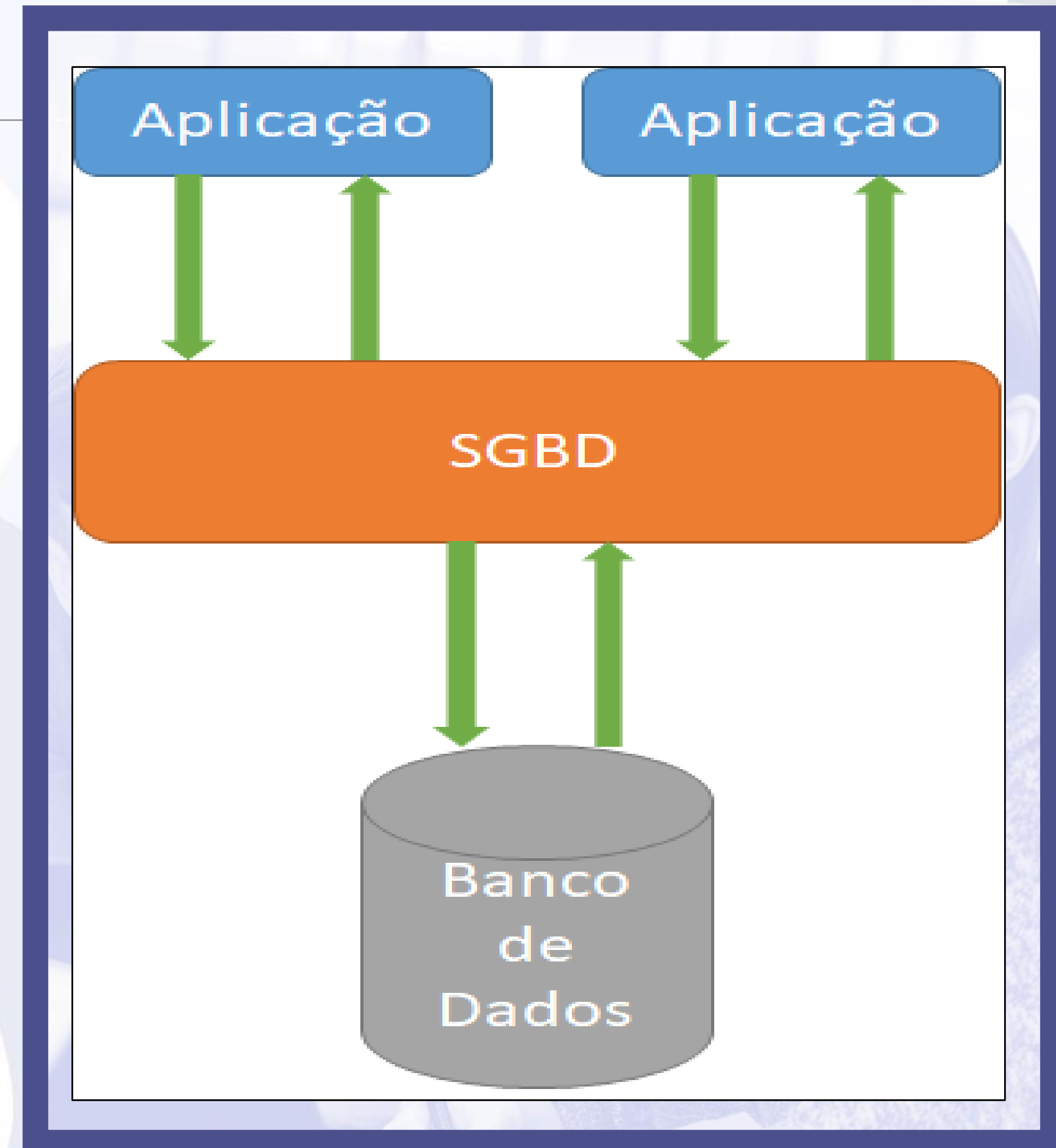




# SGBD



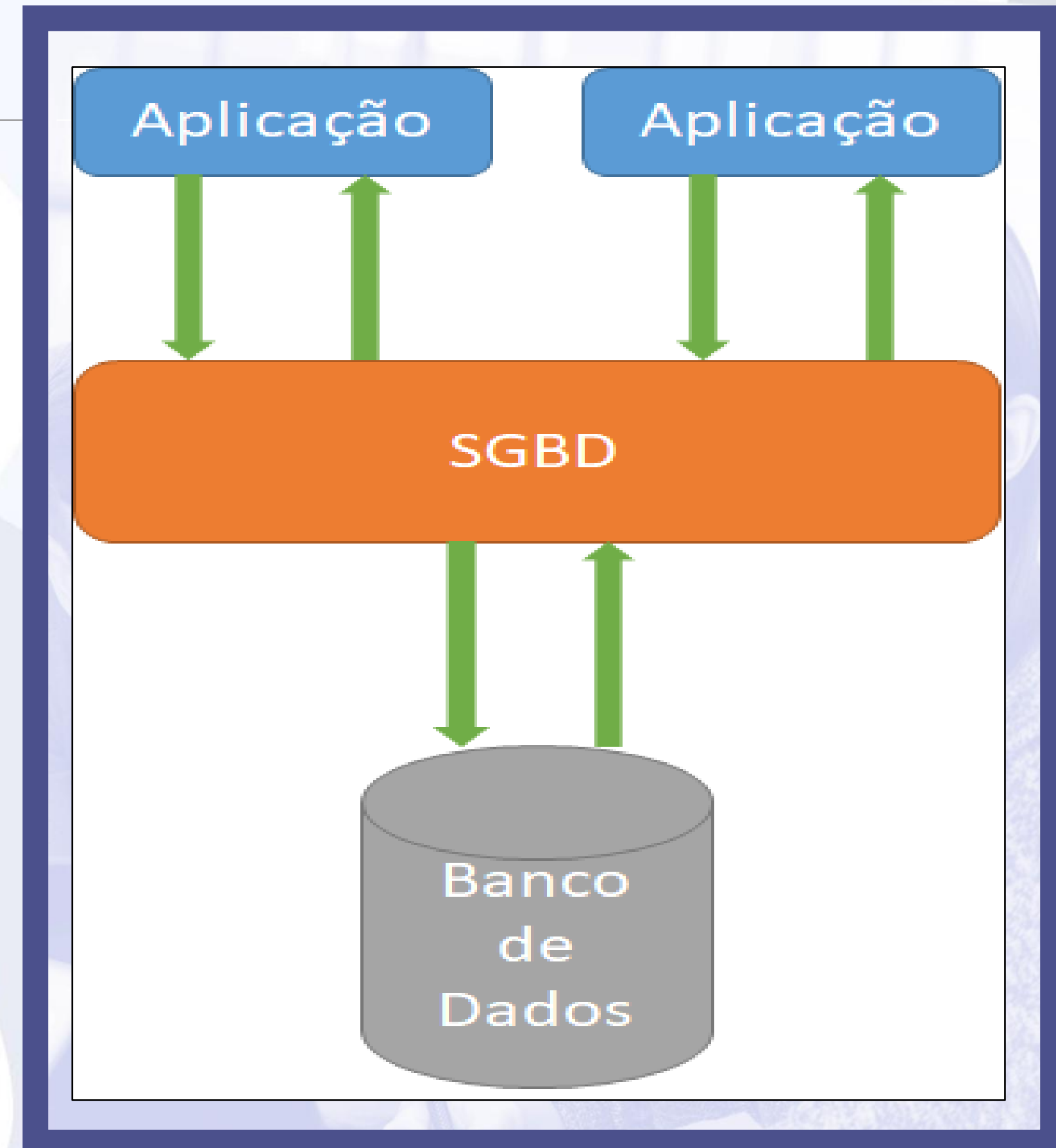
- É um sistema de software que facilita os processos de definição, construção, manipulação e compartilhamento de bancos de dados entre vários usuários e aplicações
- **Definição:** Especificação dos tipos de dados, das estruturas das tabelas e das restrições que devem ser impostas aos dados que serão armazenados
- **Construção:** Processo de acumular os dados num meio de armazenamento controlado pelo SGBD
- **Manipulação:** Operações como atualização do banco de dados (inclusão, exclusão e alteração de registros) e extração de dados, como consultas e relatórios impressos
- **Compartilhamento:** Permite aos múltiplos usuários e programas acessar, de forma concorrente, o banco de dados



# SGBD

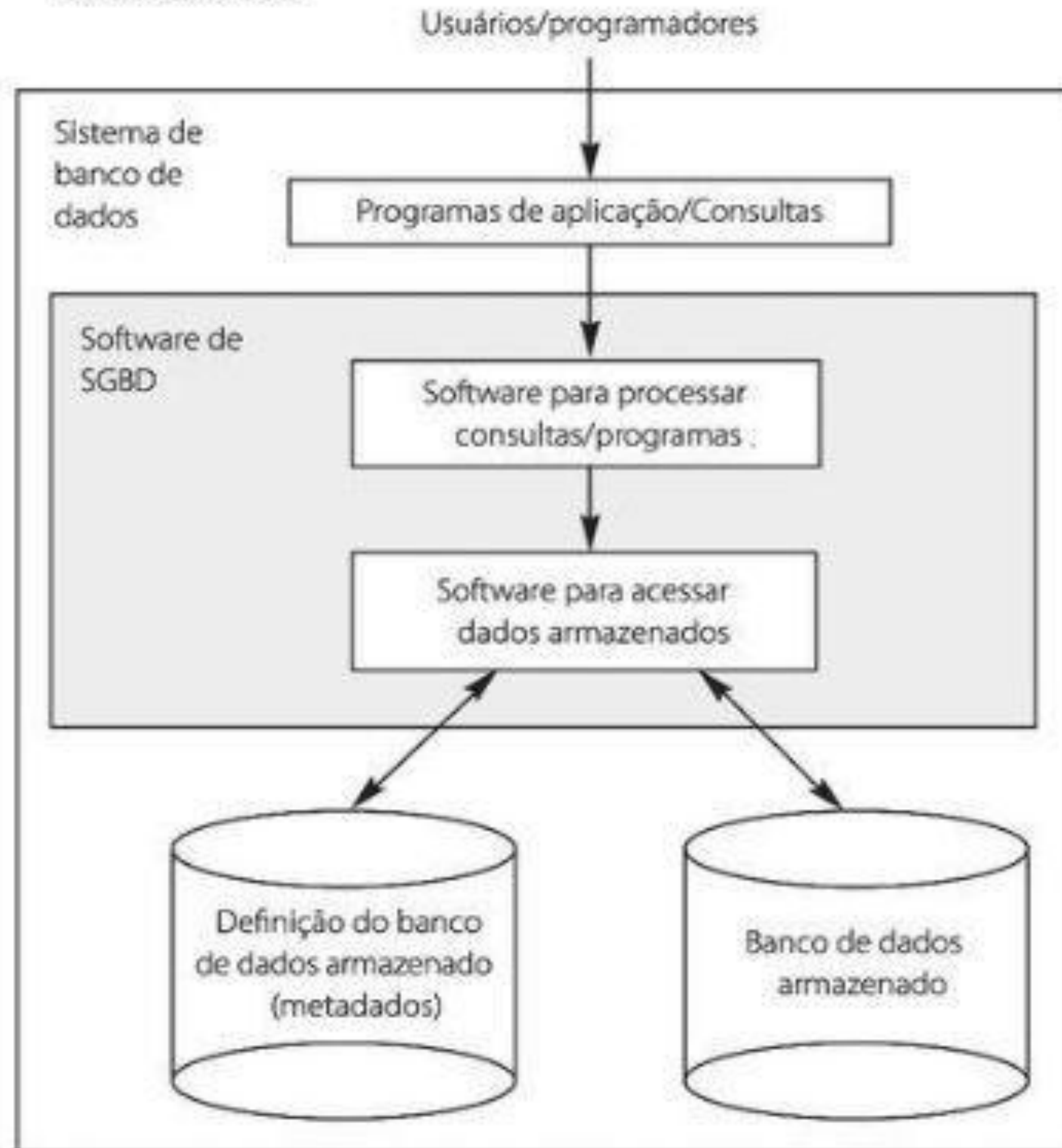


- Outras funções importantes fornecidas pelo SGBD são:
  - **Manutenção do banco de dados por um longo período**, permitindo que ele evolua em sintonia com as mudanças de requisitos;
  - **Proteção do sistema contra defeitos** (ou falhas) de hardware ou software;
  - **Proteção de segurança contra acessos não autorizados ou maliciosos**;



# Diagrama simplificado de um ambiente de sistema de banco de dados

**Figura 1.5** Diagrama simplificado de um ambiente de sistema de banco de dados.



Fonte: Elmasri e Navathe (2011, p. 4).



# Qualidade da Informação



- **Atualidade** : Deve ser a mais atual possível para facilitar o processo de tomada decisão;
- **Correção** : Deve ser correta. As instruções tem de ser capazes de verificar, tanto quanto possível, a correção da informação que processam.
- **Relevância** : Deve ser devidamente selecionada e filtrada, de modo que se use apenas aquela que é relevante para a tomada de decisão na organização.
- **Disponibilidade**: Deve estar disponível e acessível.
- **Legibilidade**: Deve ser fornecida de tal modo que seja facilmente interpretada pelos usuários.



# Requisitos de um SGBD

---



- **Eficiência:** Ser capaz de acessar, processar e alterar grandes volumes de dados de forma eficiente;
- **Robustez:** Manter os dados de forma consistente, mesmo após falha do hardware ou erros de software;
- **Controle de acesso:** Controlar o acesso de múltiplos usuários
- **Persistência:** Manter os dados durante longos períodos;

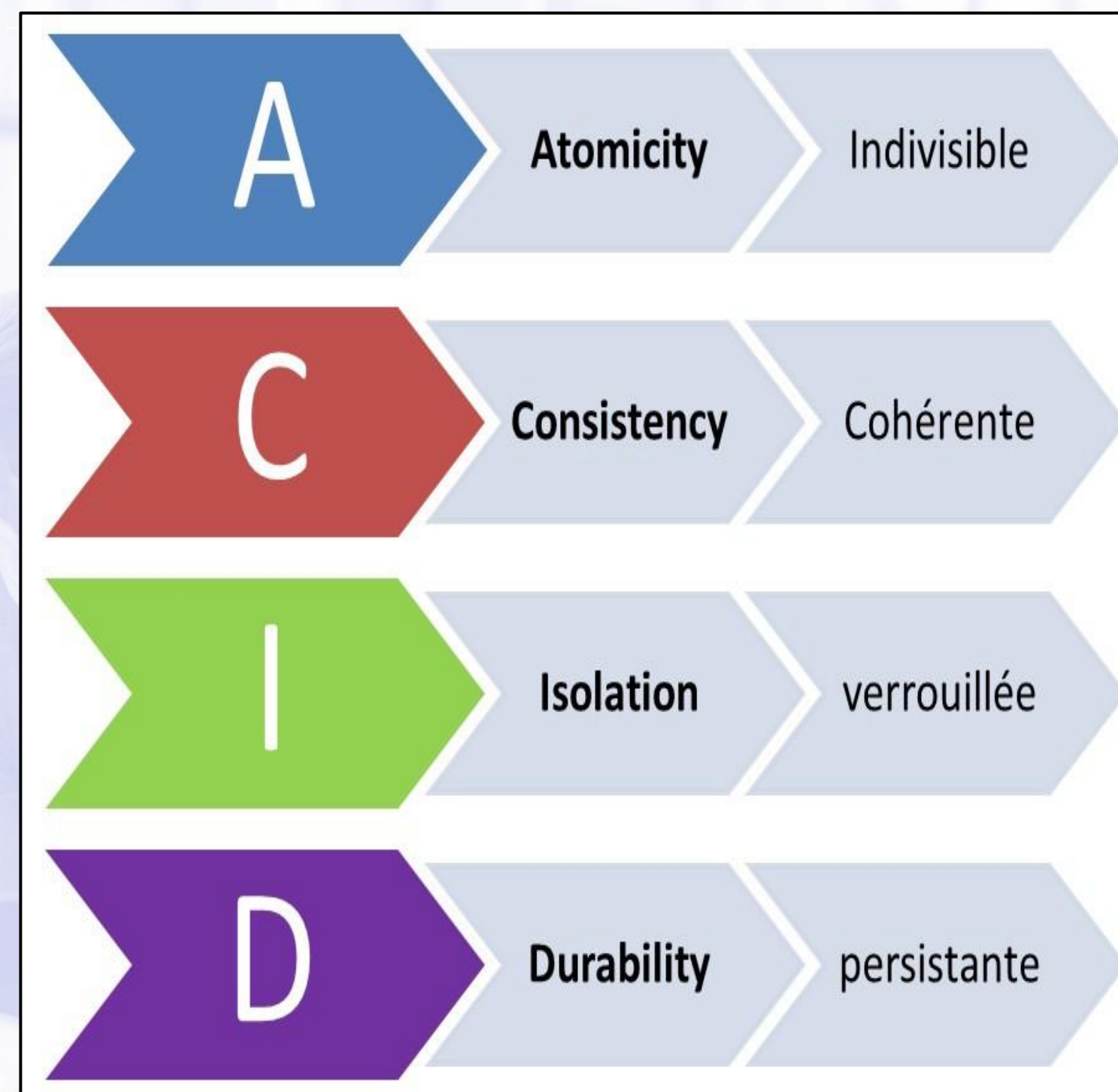




# Propriedades dos SGBDs Relacionais



- **Atomicidade:** todas as operações de uma transação devem ser efetivadas ou, na ocorrência de uma falha, nada deve ser efetivado “tudo ou nada” – não se admite parte de uma operação
- **Consistência:** transações preservam a consistência da base: Estado inicial consistente □ Estado final consistente
- **Isolamento:** a maneira como várias transações em paralelo interagem (o que pode ser lido e o que pode ser escrito por cada uma) deve ser bem definido
- **Durabilidade:** uma vez consolidada (committed) a transação, suas alterações permanecem no banco até que outras transações aconteçam

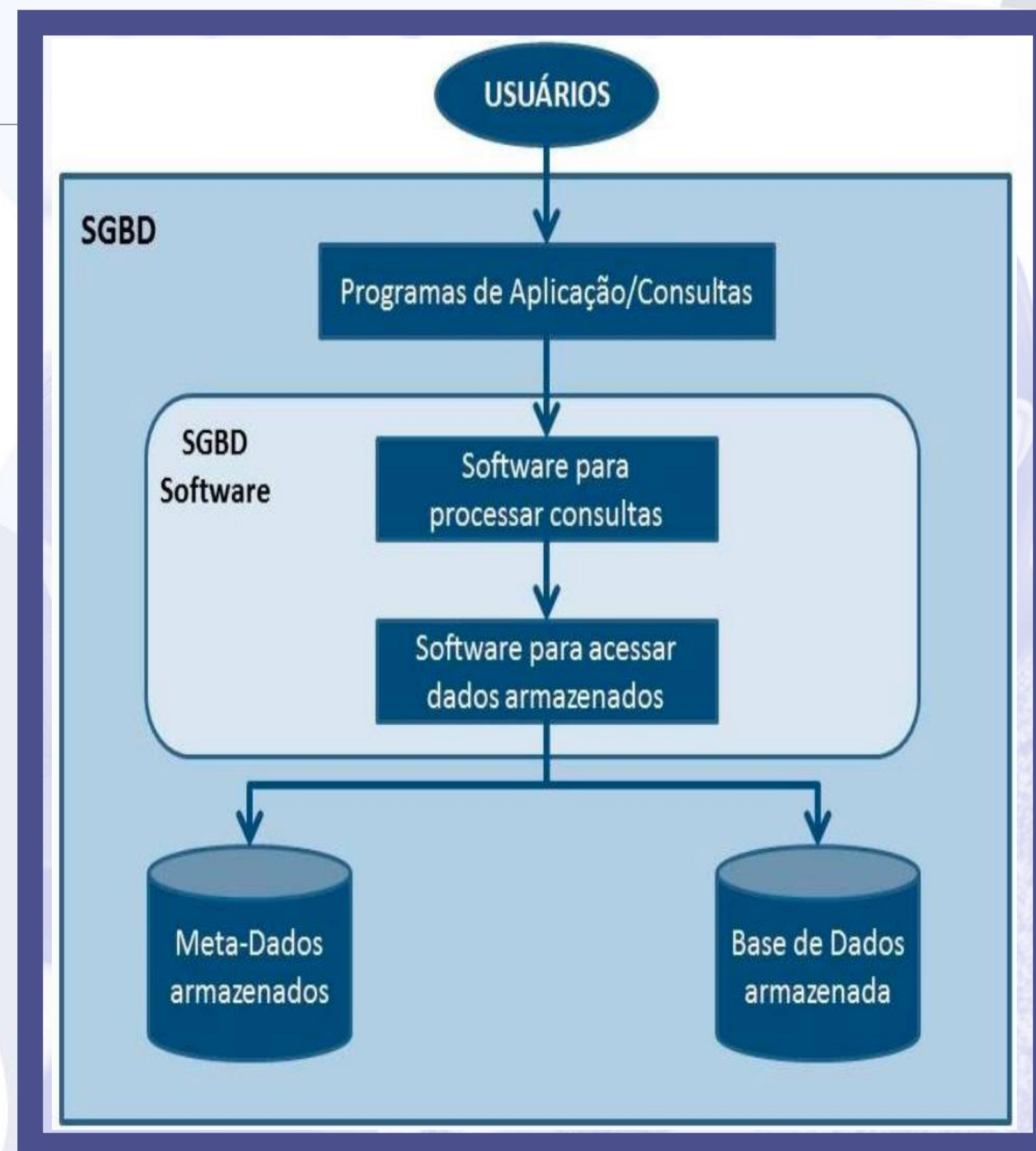




# Serviços prestados pelo SGBD



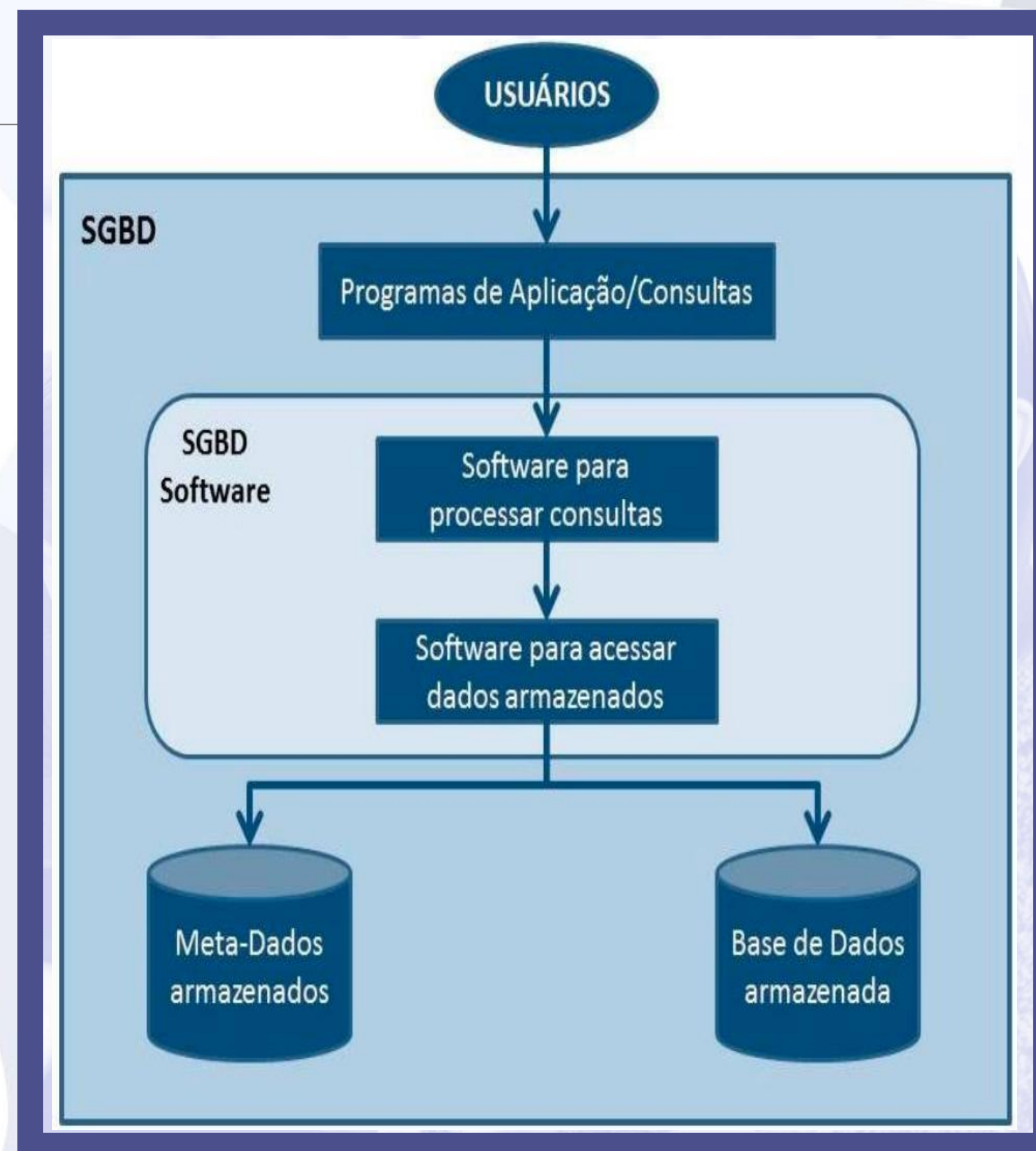
- **Interação com o gerente de arquivos** – Tem como objetivo minimizar os acessos a disco (I/O), pois esse acesso é mais lento que o acesso a memória;
- **Gerenciador de Dados** – Os dados estarão centralizados, por isso as relações entre os dados devem ser gerenciadas e verificadas pelo SGBD;
- **Integridade** – Verificar se as alterações do banco de dados estão de acordo com as regras de integridade e com as validações estabelecidas na sua definição:



# Serviços prestados pelo SGBD



- **Segurança** – Assegurar que os usuários apenas têm acesso a informação a que lhes é permitido;
- **Backup & Recovery (Backup e recuperação)** – Capacidade de detectar falhas decorrentes de problemas de fornecimento de energia elétrica, de hardware, de erros de software, etc., e ser capaz de recolocar o banco de dados no estado estável que existia imediatamente antes de ocorrência de falha;
- **Gerenciamento de concorrência** – Gerenciar o acesso de múltiplos usuários aos seus dados, mantendo a consistência da informação a que cada usuário tem acesso;

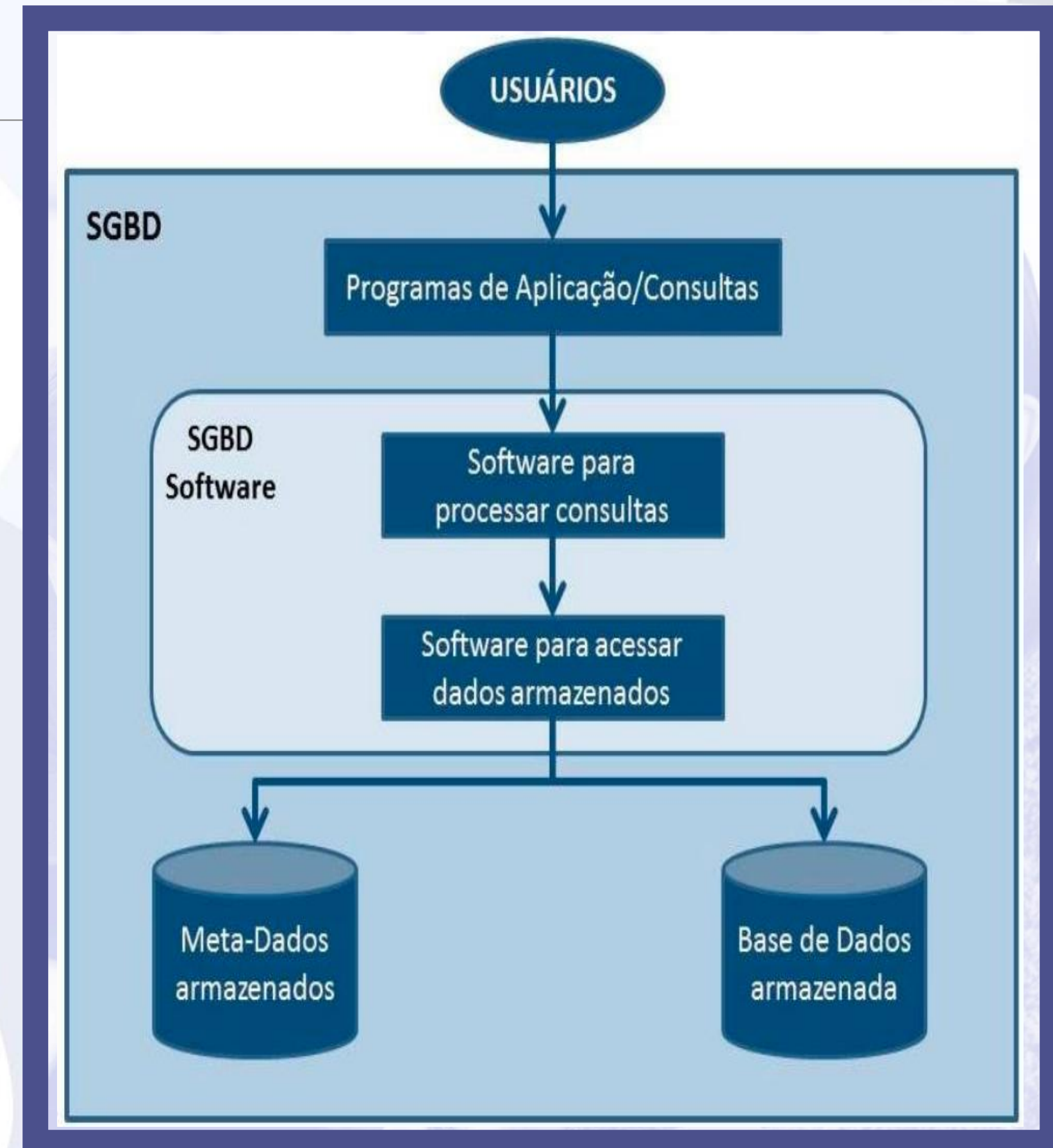




# Vantagens do uso do SGBD



- **Independência entre programas e dados**
  - Catalogo que consiste de **metadados** – dados sobre os dados
- **Independência entre operações e programas**
  - Funções / procedimentos de manipulação dos dados armazenados também fazem parte do BD
- **Segurança**
  - Controle de acesso mais especializado
- **Suporte a Visões**
  - Mesmo conjunto de dados pode ser apresentado a usuários diferentes de forma distinta



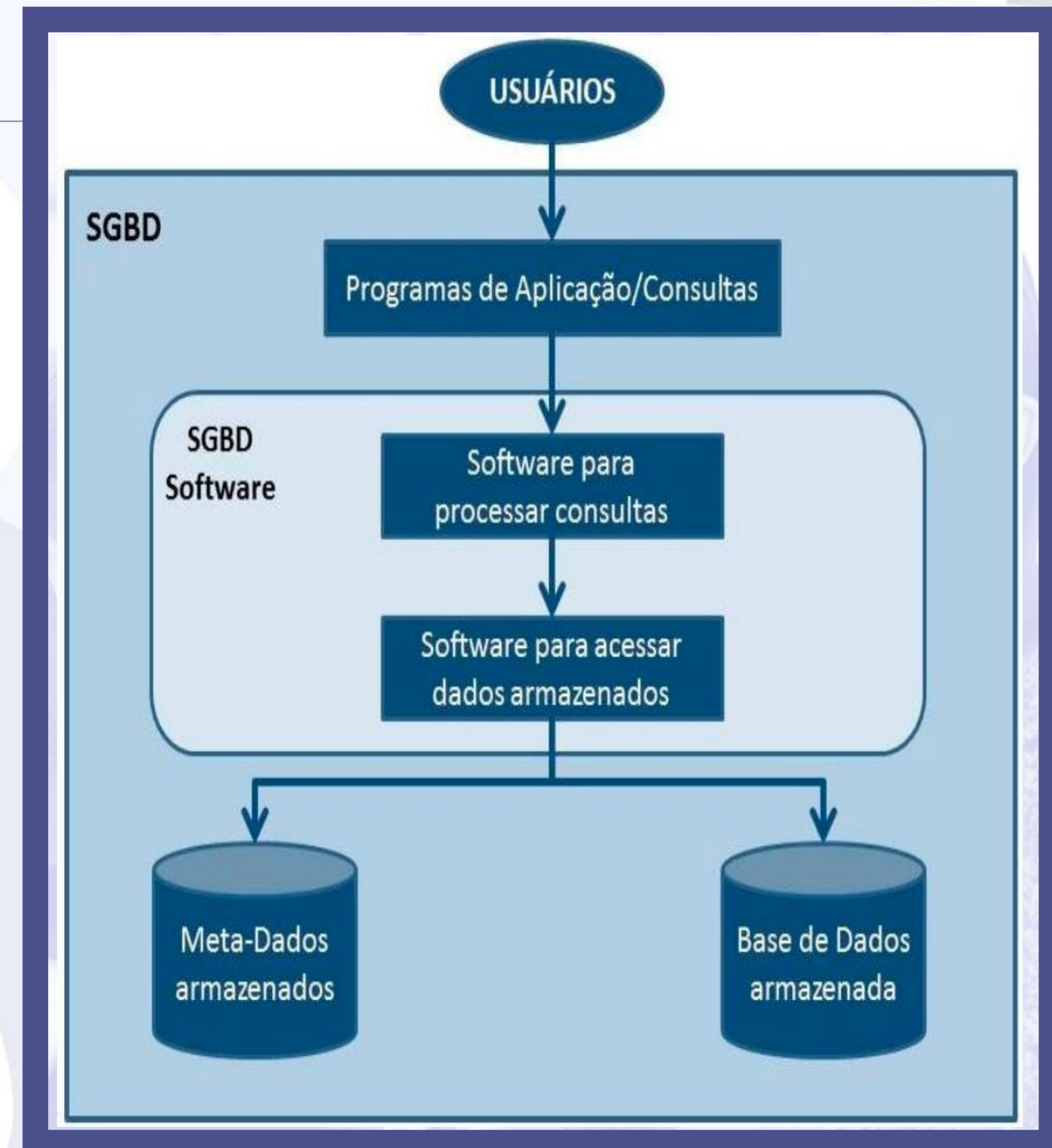


# Vantagens do uso do SGBD



- **Operação**

- Em alguns tipos de sistemas de banco de dados, os usuários podem definir operações sobre dados como parte das definições do banco de dados. Um operação (também chamada função ou método) é especificada em duas partes:
  - **Interface ( ou assinatura)** – que consiste em incluir o nome da operação e os tipos de dados de seus argumentos (ou Parâmetros);
  - **Implementação (ou método)** – que é especificada separadamente, o pode ser alterada sem afetar a interface;



# Vantagens do uso do SGBD



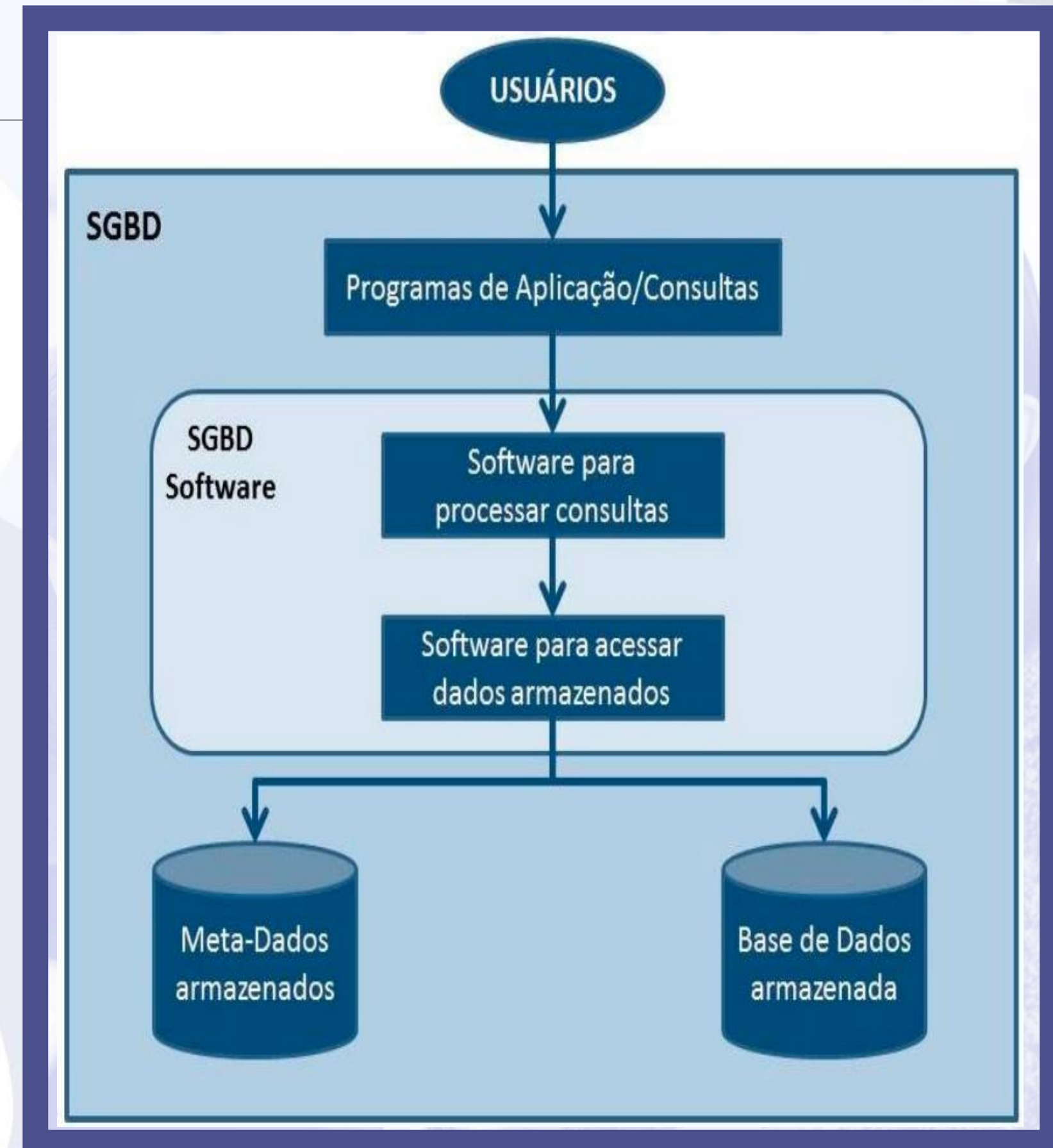
- **Abstração de dados**

- O SGBD oferece aos usuários uma representação conceitual de dados que não inclui muitos dos detalhes de como os dados são armazenados ou de como as operações são implementadas;

- **Facilidades de Backup e Restauração**

- **Fornecimento de Múltiplas Interfaces aos Usuários**

- Baseadas em Menus e formulários (GUIs – *Graphical User Interface*)
- Linguagens de consulta e interfaces de linguagem de programação

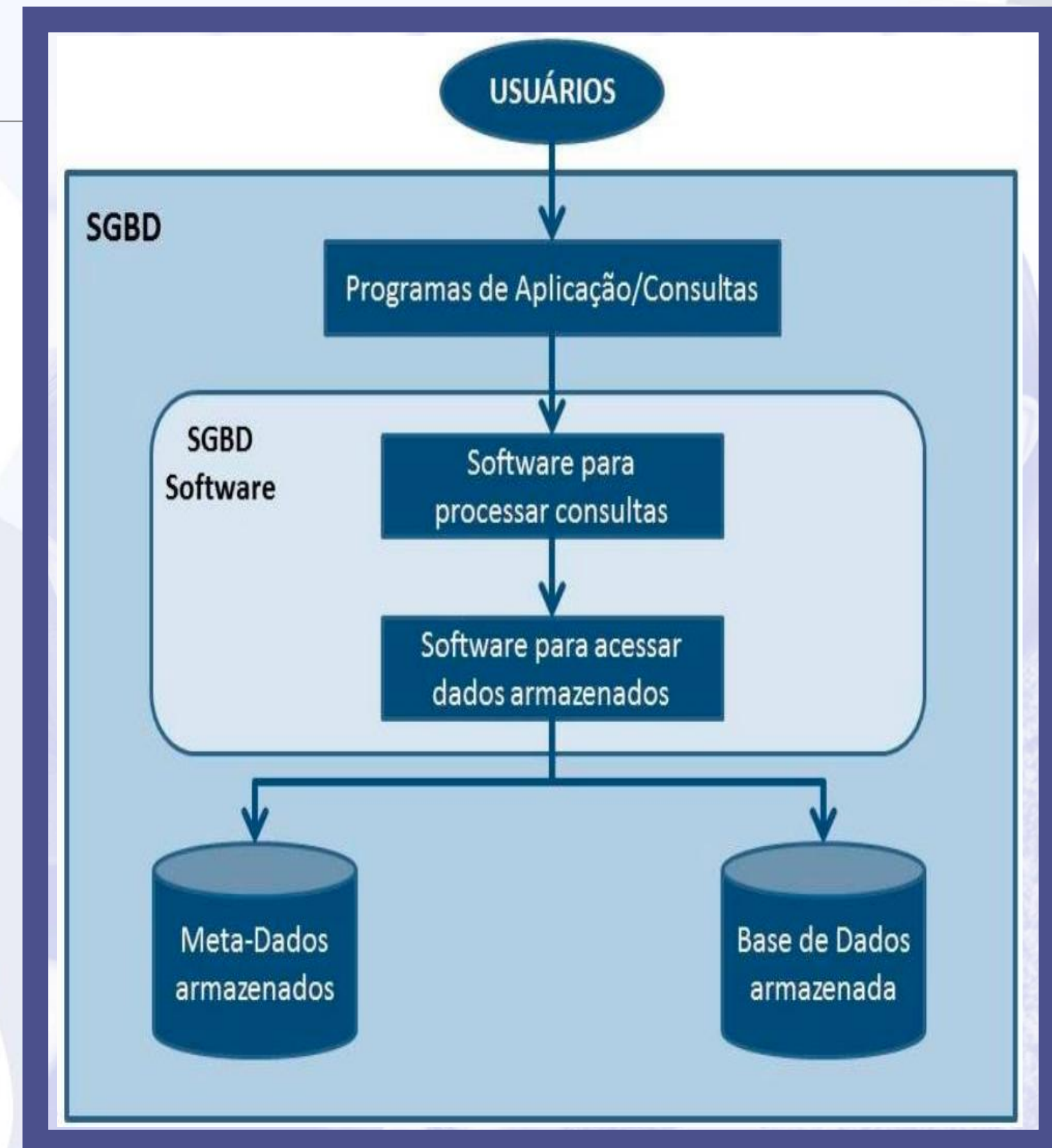




# Vantagens do uso do SGBD



- **Compartilhamento de dados e processamento de transação multiusuário;**
  - O SGBD precisa permitir que múltiplos usuários acessem o banco de dados ao mesmo tempo.
  - O SGBD precisa incluir um software de controle de concorrência para garantir que vários usuários tentando atualizar o mesmo dado façam isso de maneira controlada, de modo que o resultado dessas atualizações seja correto.





# Sistemas de Banco de Dados

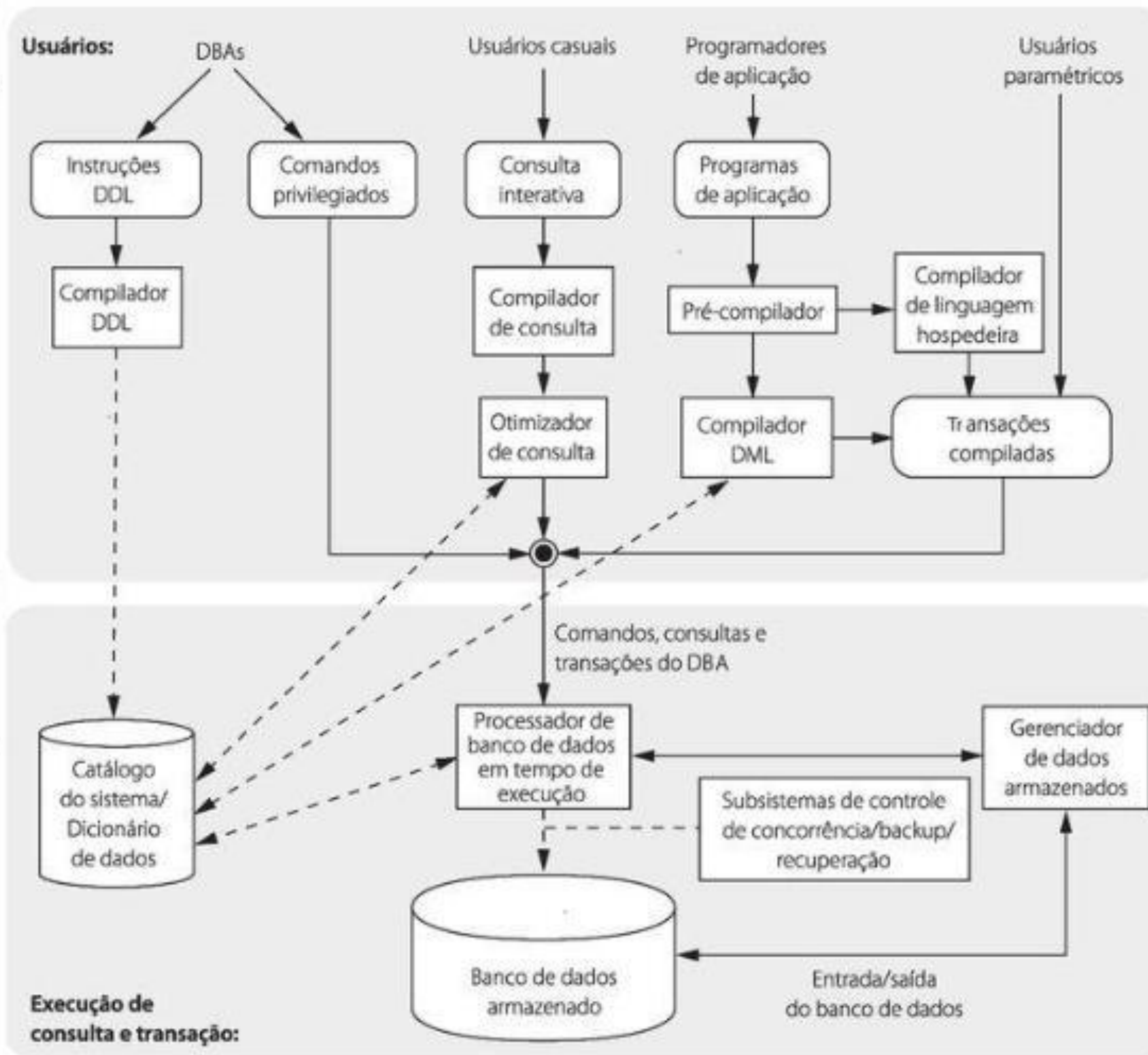


- É um ambiente de hardware e de software composto por dados armazenados em um banco de dados (BD), pelo software de gerência do banco de dados (SGBD) e os programas de aplicação



# Componentes de um SGBD

Figura 1.12 Módulos do SGBD e suas interações.



Fonte: Elmasri e Navathe (2011, p. 27).



# Passos para a criação de um banco de dados



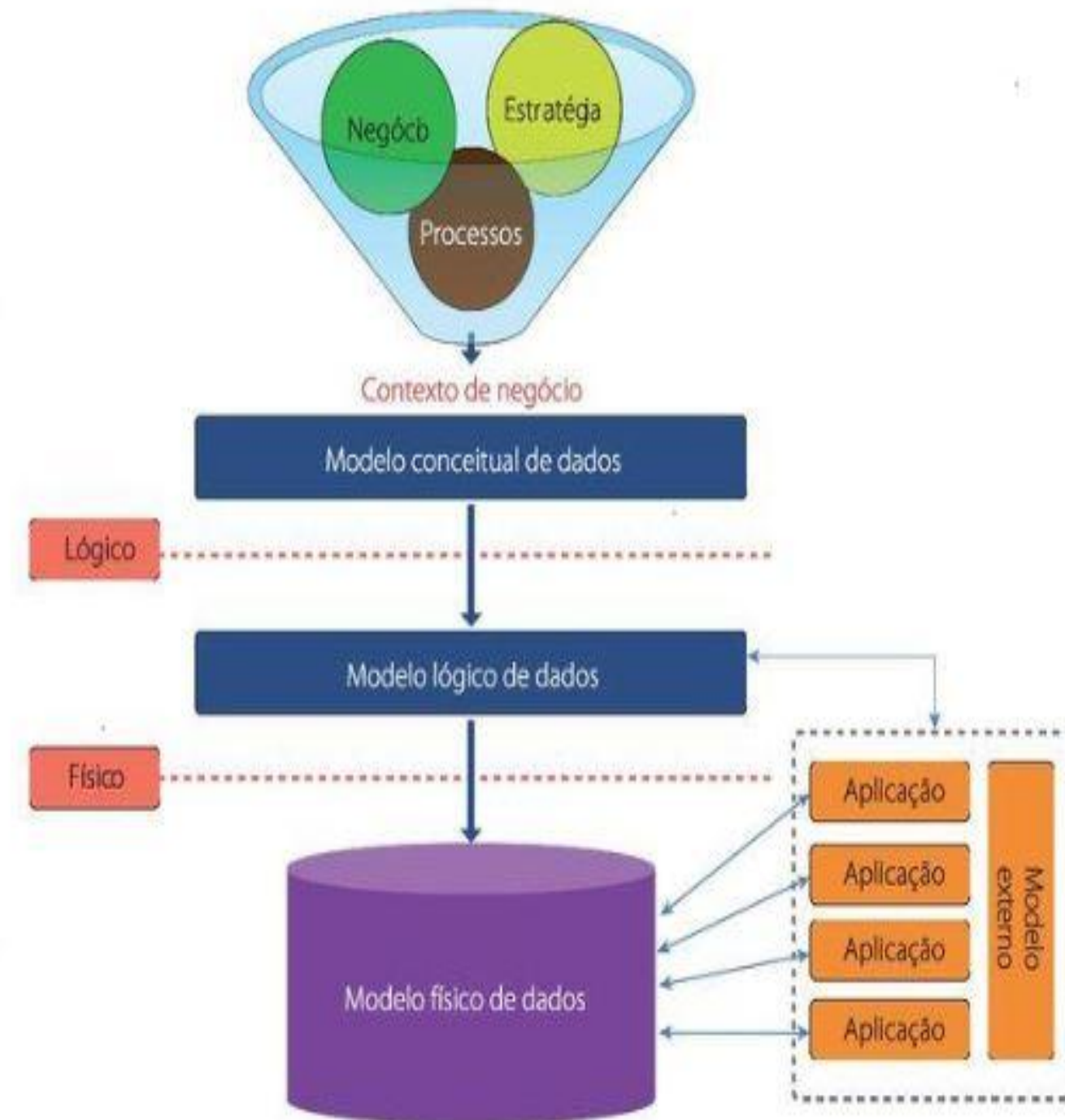
- **Modelo Conceitual**

- Possuem conceitos que descrevem os dados como os usuários os percebem: entidades, atributos e relacionamentos
- Independente de SGBD
- Modelo Conceitual – MER

- **Modelo Lógico**

- Descrevem a forma como os dados estão organizados dentro do computador
- Esquema Lógico
- Mapeamento do Modelo Conceitual para modelo do SGBD
- Ex: Modelo Relacional, Modelo Orientado a Objetos etc.

Figura 1.13 Estágios da modelagem de dados.



Fonte: Puga, França e Goya (2013, p. 81).

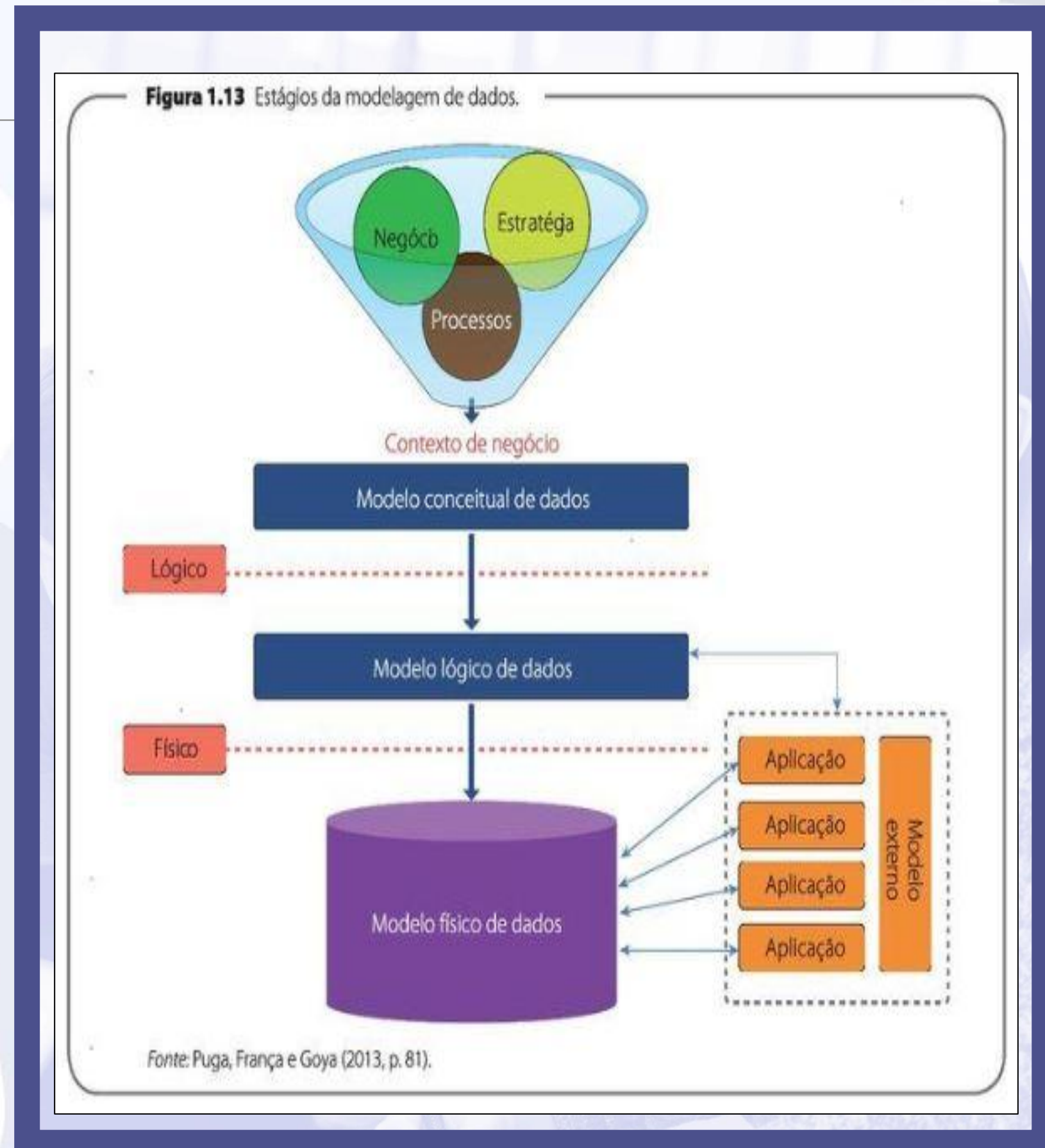


# Passos para a criação de um banco de dados



- **Projeto Físico**

- Descrevem detalhes de como os dados estão armazenados no computador
- Estruturas Físicas de Armazenamento
  - Organização de registros físicos
  - Índices
- Critérios
  - Tempo de resposta
  - Espaço utilizado
  - Número de transações



# Modelagem de dados



**ser**  
educacional

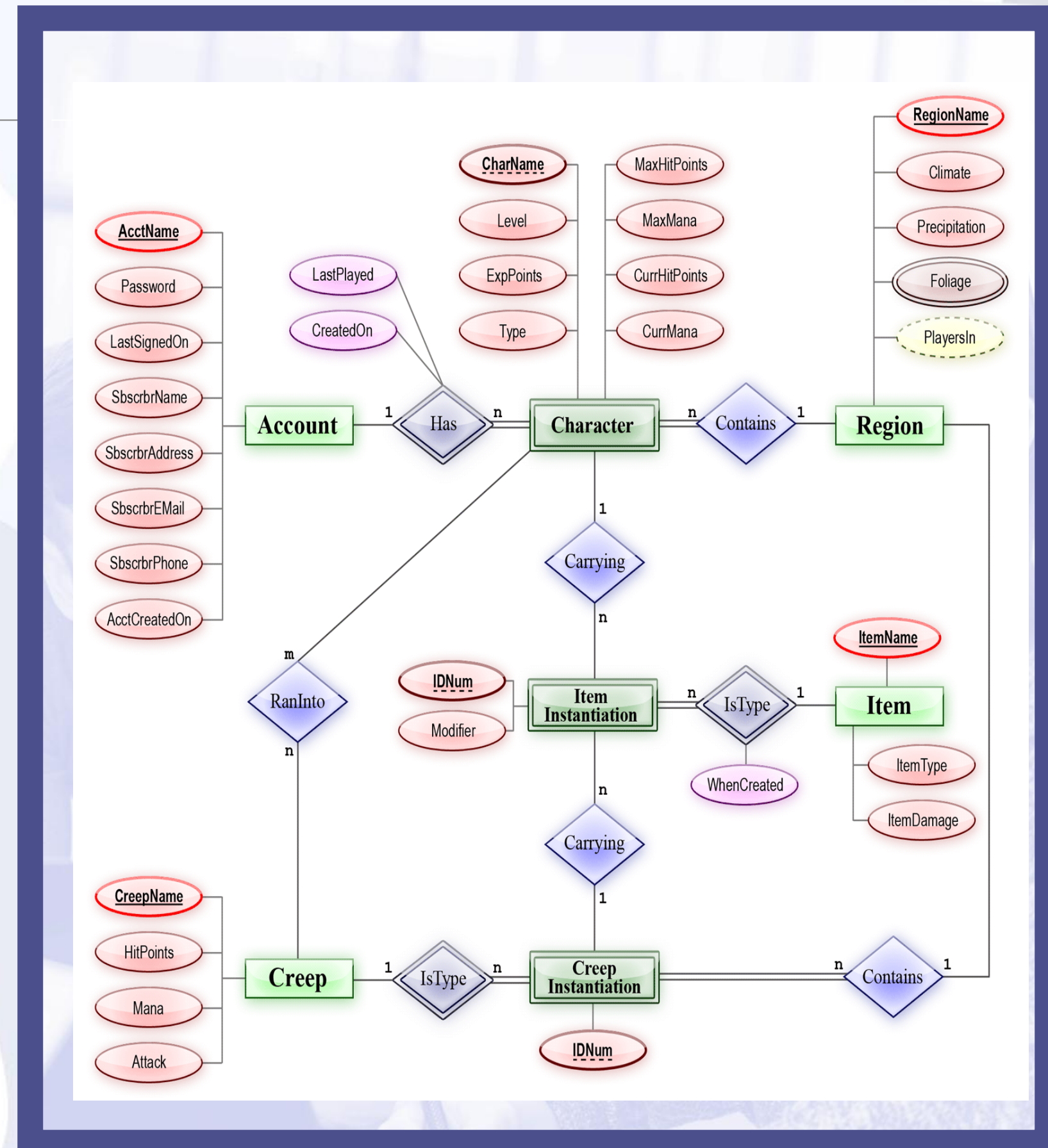
gente criando o futuro



# Diagrama de Entidade-Relacionamento



- As vantagens da utilização do DER são muitas:
  - Possui uma sintaxe robusta, permitindo a criação da documentação das informações do cliente de maneira precisa e clara;
  - Facilita a comunicação, permitindo que o cliente e o usuário entendam o modelo;
  - Facilidade para criar e manter o modelo;
  - Pode ser integrado com várias aplicações e projetos diferentes;
  - Oferece independência de implementação, pois sua utilização é universal e não está vinculada a um tipo de banco de dados.

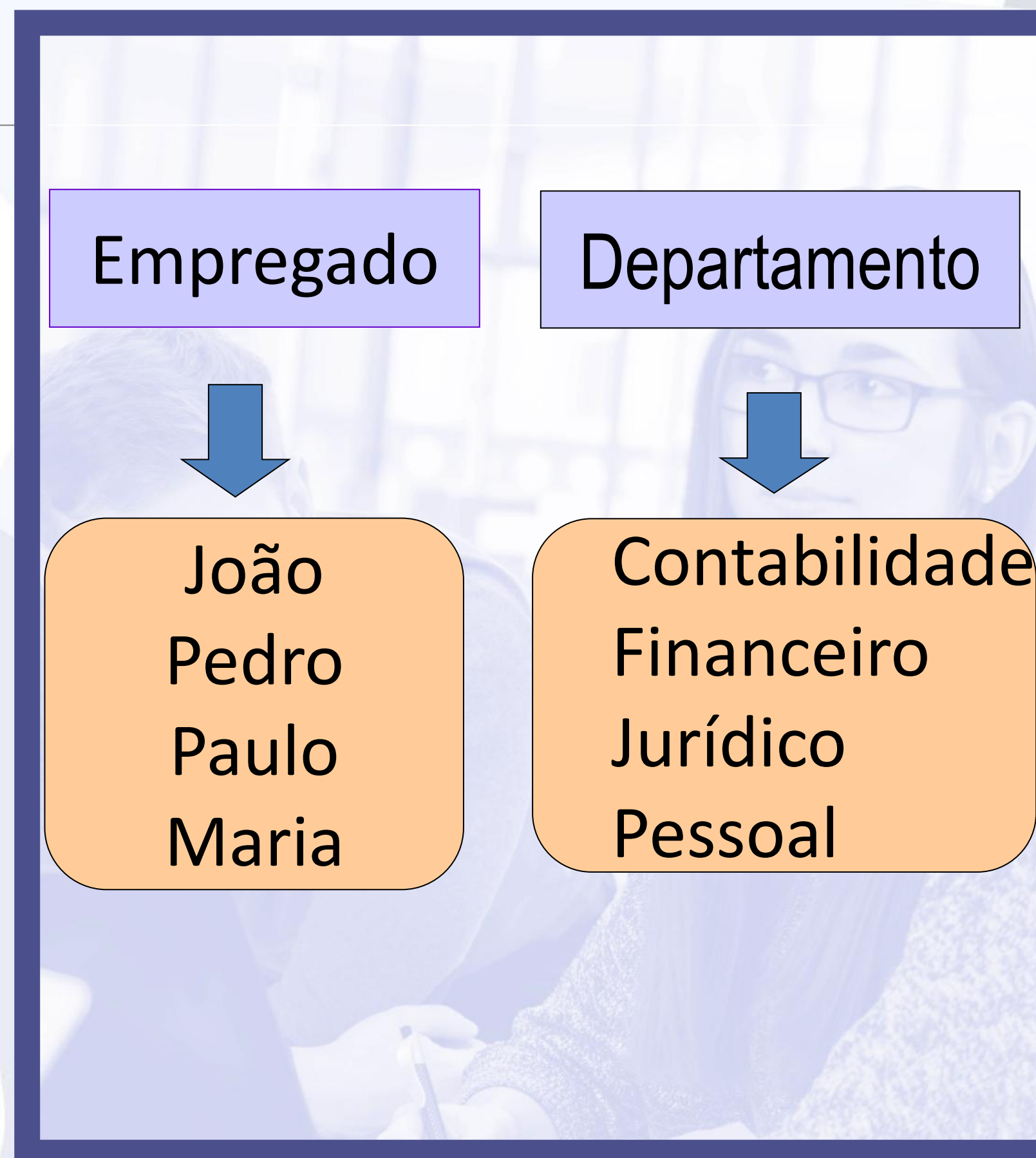




# Entidade

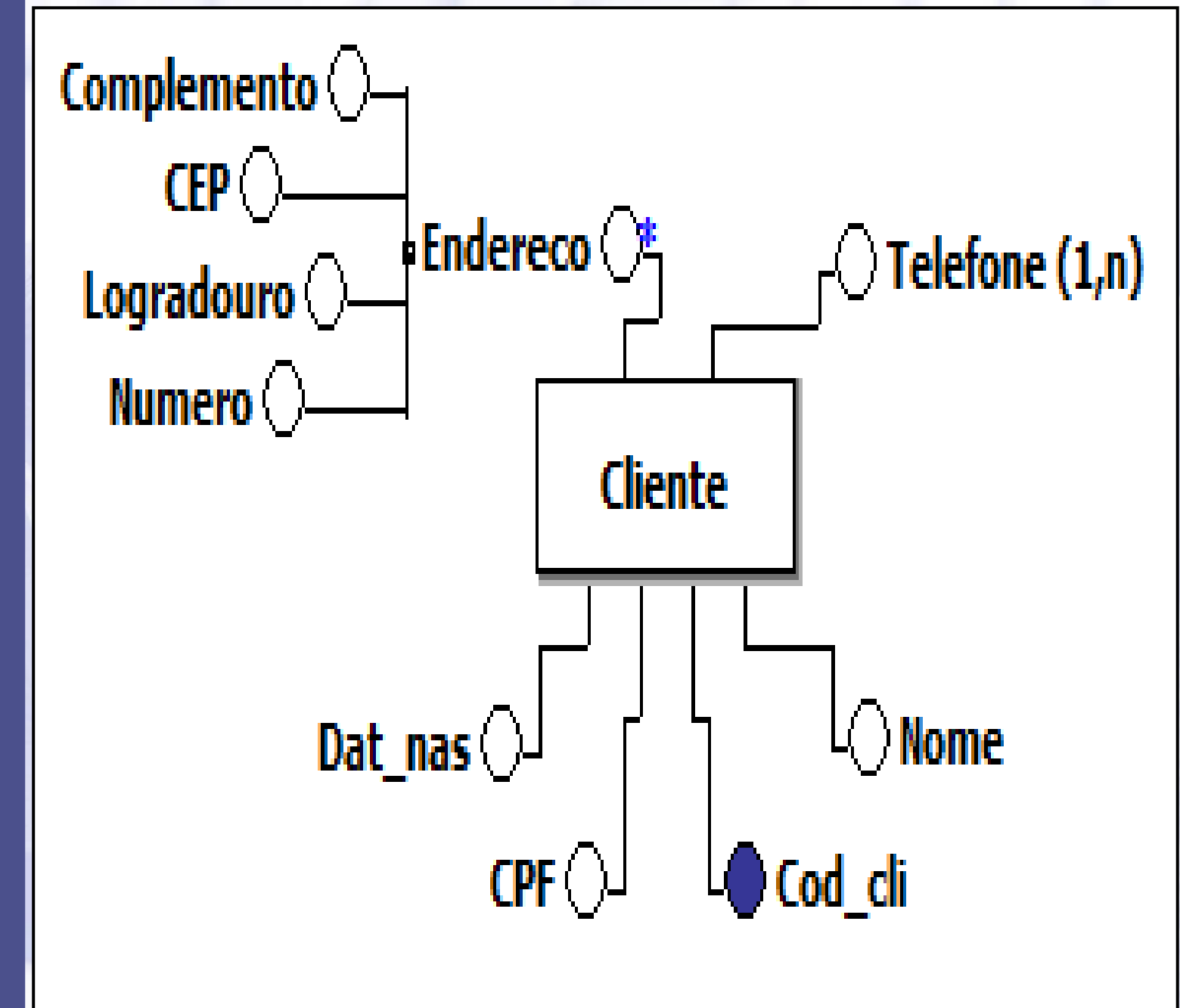


- É um **conjunto de objetos** do mundo real sobre os quais se deseja manter informações no banco de dados
- É distinguível de outros objetos;
- Representada através de um retângulo,
- onde são colocados os seus nomes;
- Pode representar:
  - objetos concretos (uma pessoa)
  - objetos abstratos (um departamento)



# Atributos

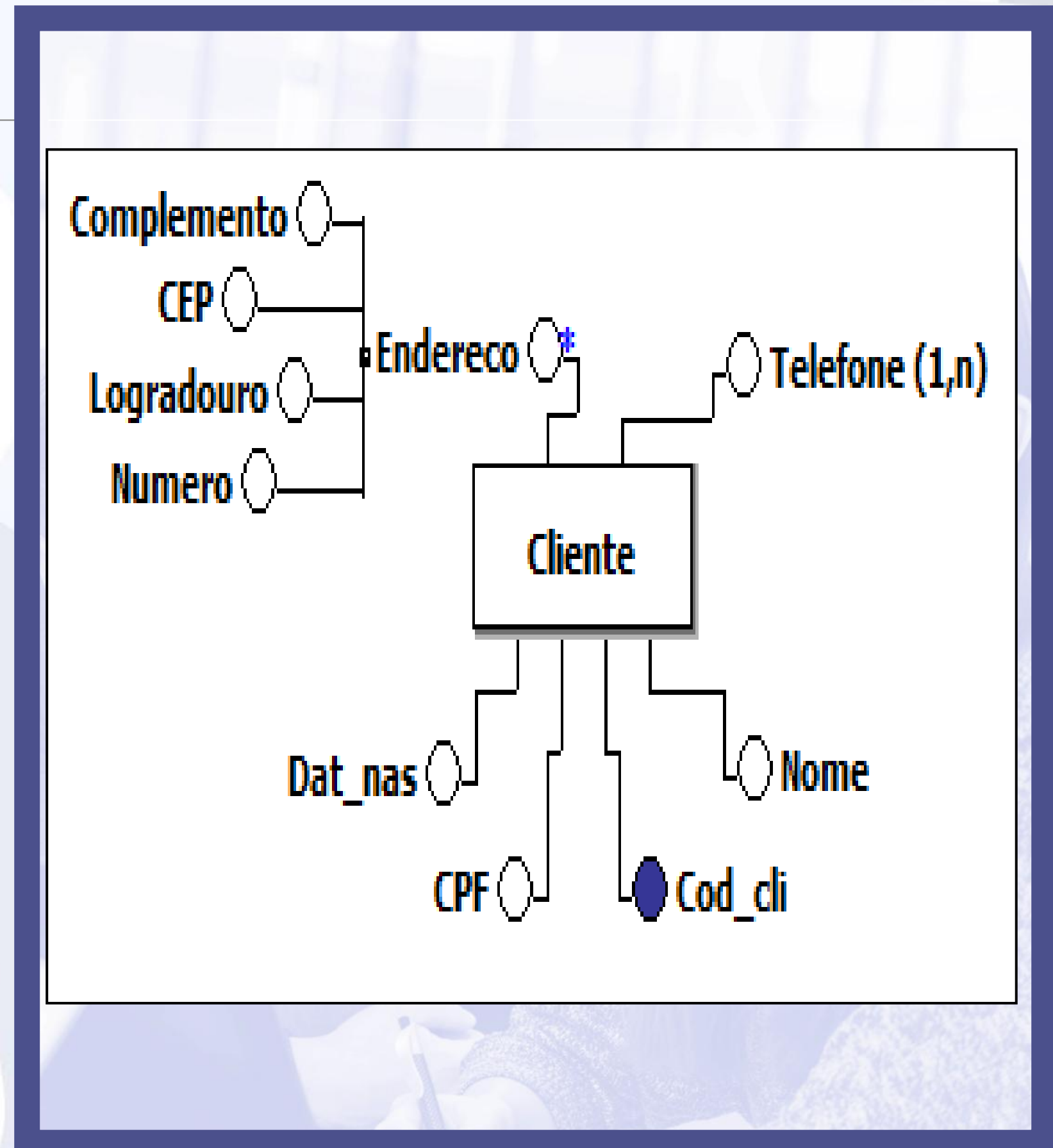
- É um dado que é associado a cada ocorrência de uma entidade ou de um relacionamento
- São as informações que referenciam a entidade.
- Como um conjunto de entidades pode ter vários atributos, cada entidade pode ser descrita por um conjunto de pares (atributo, valor de dados), um par para cada atributo do conjunto de entidades;



# Atributos



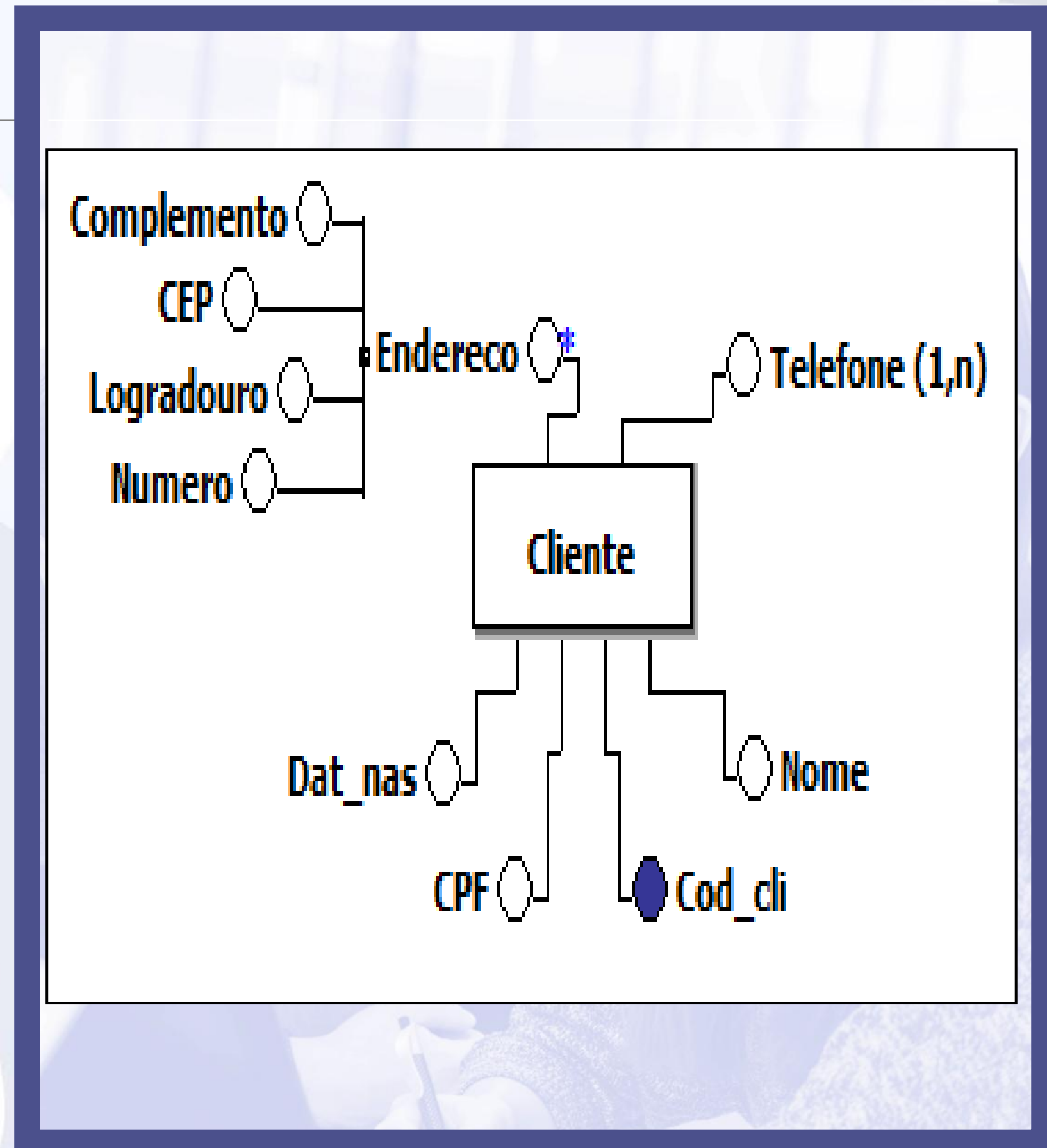
- **Atributos Simples:**
  - Os atributos simples não são divisíveis em subpartes, por exemplo o preço de um produto
- **Atributos Compostos:**
  - Os atributos compostos são divididos em subpartes, por exemplo o endereço, que pode ser dividido em rua, cidade e estado. Isso normalmente acontece quando existe a necessidade do usuário realizar busca específica por um desses atributos;
- **Atributos Valor Único:**
  - Os atributos de valor único, são aqueles que apresentam um valor único para de determinada entidade;
- **Atributos Multivalorados:**
  - Os atributos multivalorados possuem mais de um valor para uma mesma entidade, por exemplo o número de telefone de um funcionário;





# Atributos

- **Atributo derivado :**
  - São atributos que seu valor pode ser atribuído através de outros atributos ou entidades relacionadas, por exemplo a idade de funcionário pode ser calculada a partir da data de nascimento, nesse caso bastaria ter a data para calcular a idade;
- **Atributo Valores NULL:**
  - A entidade pode não ter um valor aplicável para um atributo.
- **Atributos Complexos:**
  - Combinação de atributos Compostos e multivalorados
- **Domínio de um Atributo:**
  - Conjunto de valores permitidos para o atributo
- Ex.: Sexo {M, F}

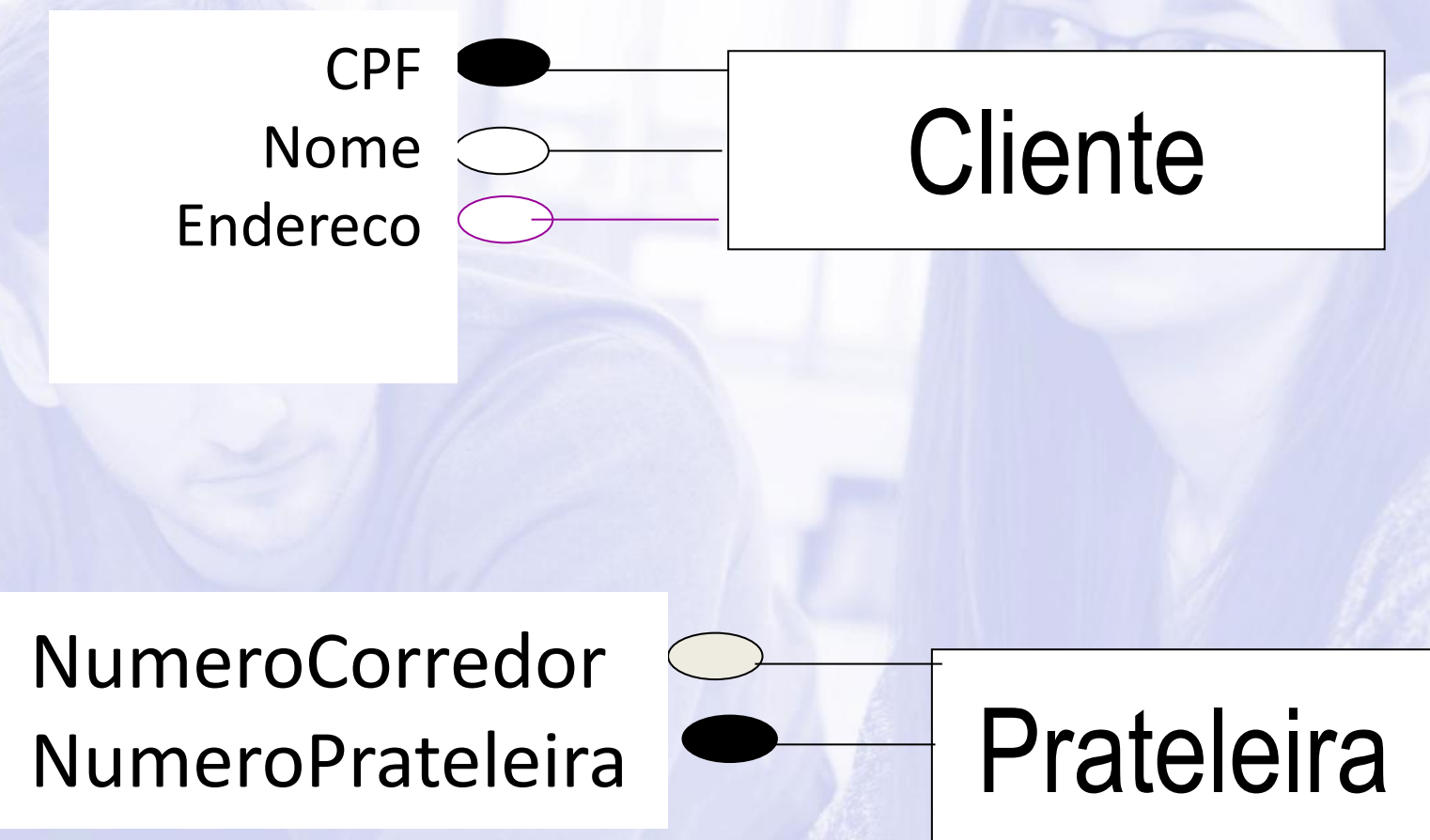


# Atributo Identificador



- Cada entidade deve ter um identificador
- Identificador (também conhecido como *chave*):
  - É o conjunto de um ou mais atributos ou relacionamentos cujos valores servem para distinguir uma ocorrência da entidade das demais ocorrências da mesma entidade
  - Exemplo: os atributos **CPF** ou **Carteira de Identidade** identificam UNICAMENTE um cidadão brasileiro

## Representação no Modelo

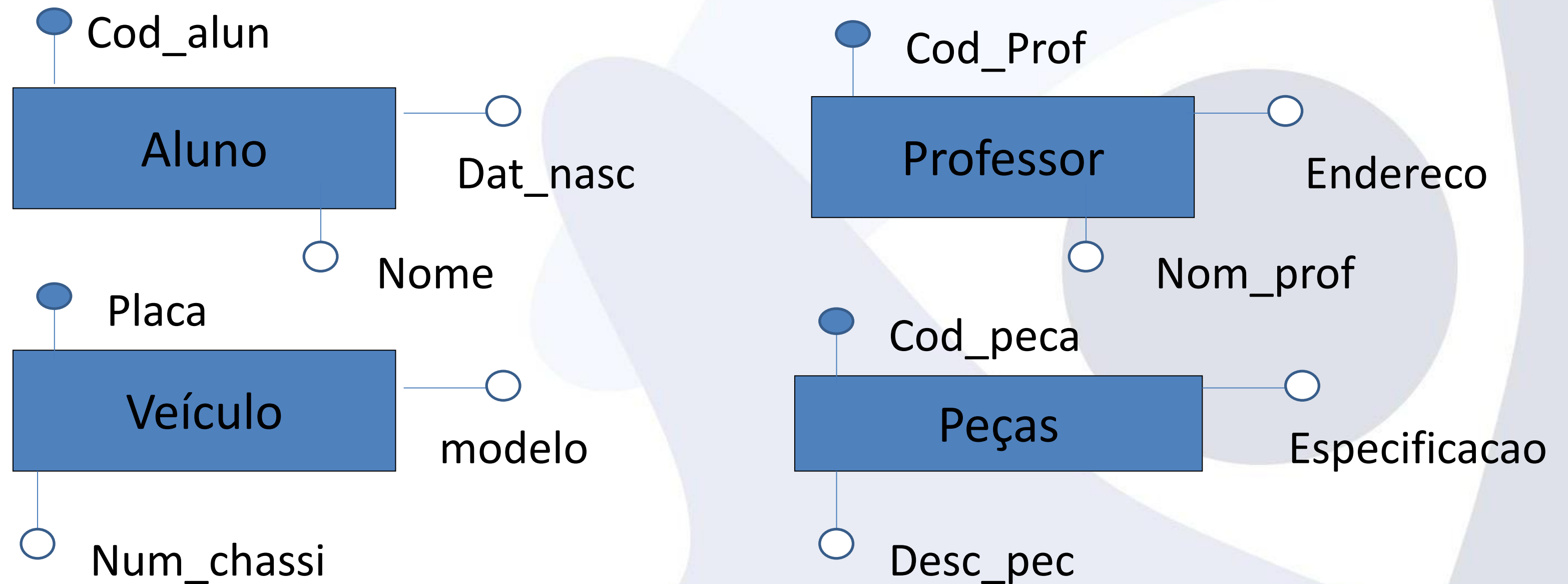






# Exemplo

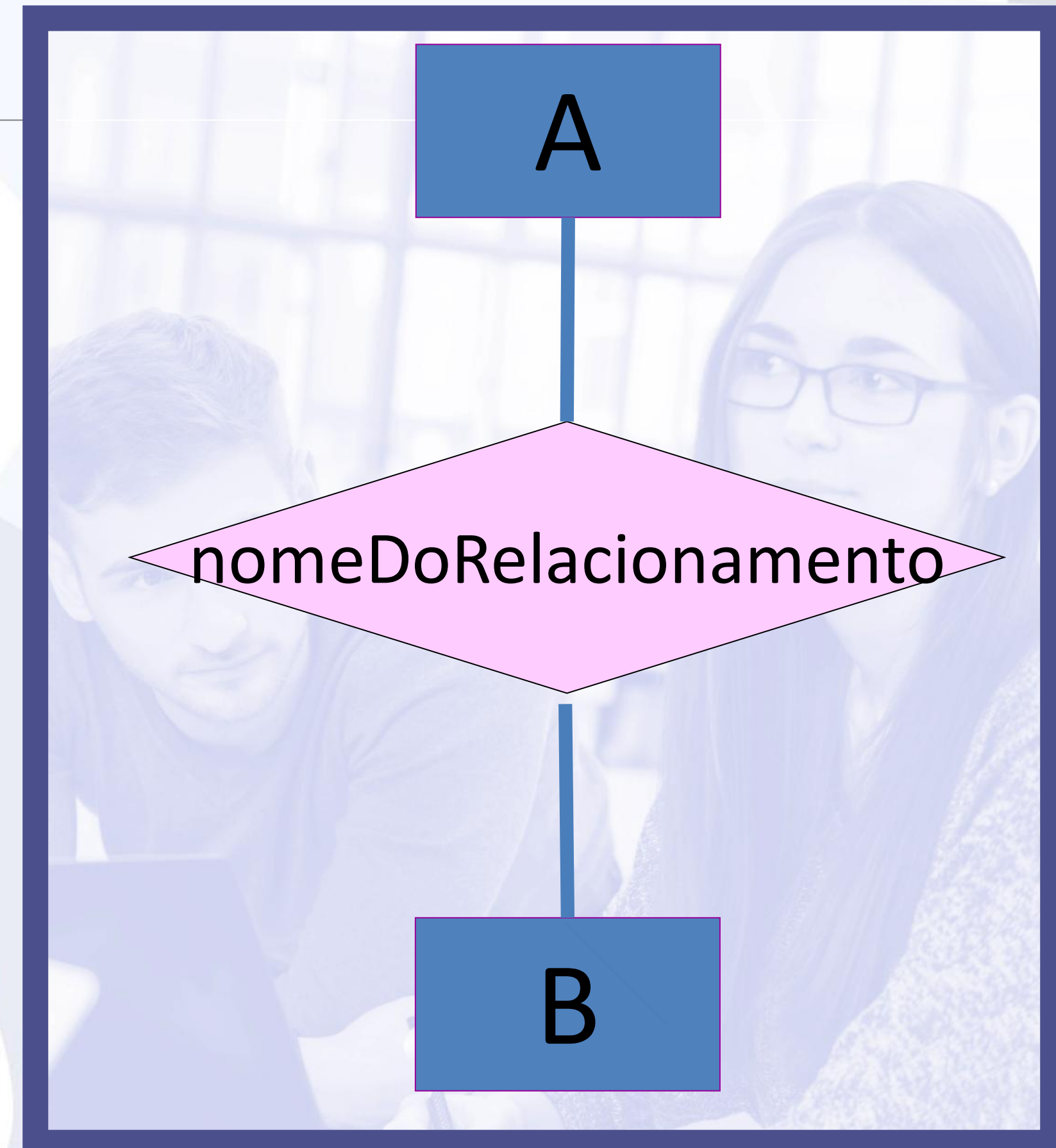
Considerando dois sistemas, o primeiro acadêmico e o segundo um de vendas de peças, defina pelo menos 3 atributos para cada uma das entidades





# Relacionamentos

- É através dos relacionamentos conseguimos ligar a informação presente em entidades de algumas forma relacionadas;
- Um relacionamento corresponde a uma ligação lógica entre entidades, indicando a forma como as duas entidades se relacionam;
- Só através da capacidade que os bancos de dados relacionais têm de permitir o relacionamento entre entidades distintas é que poderemos saber como os dados se comportam em uma organização;

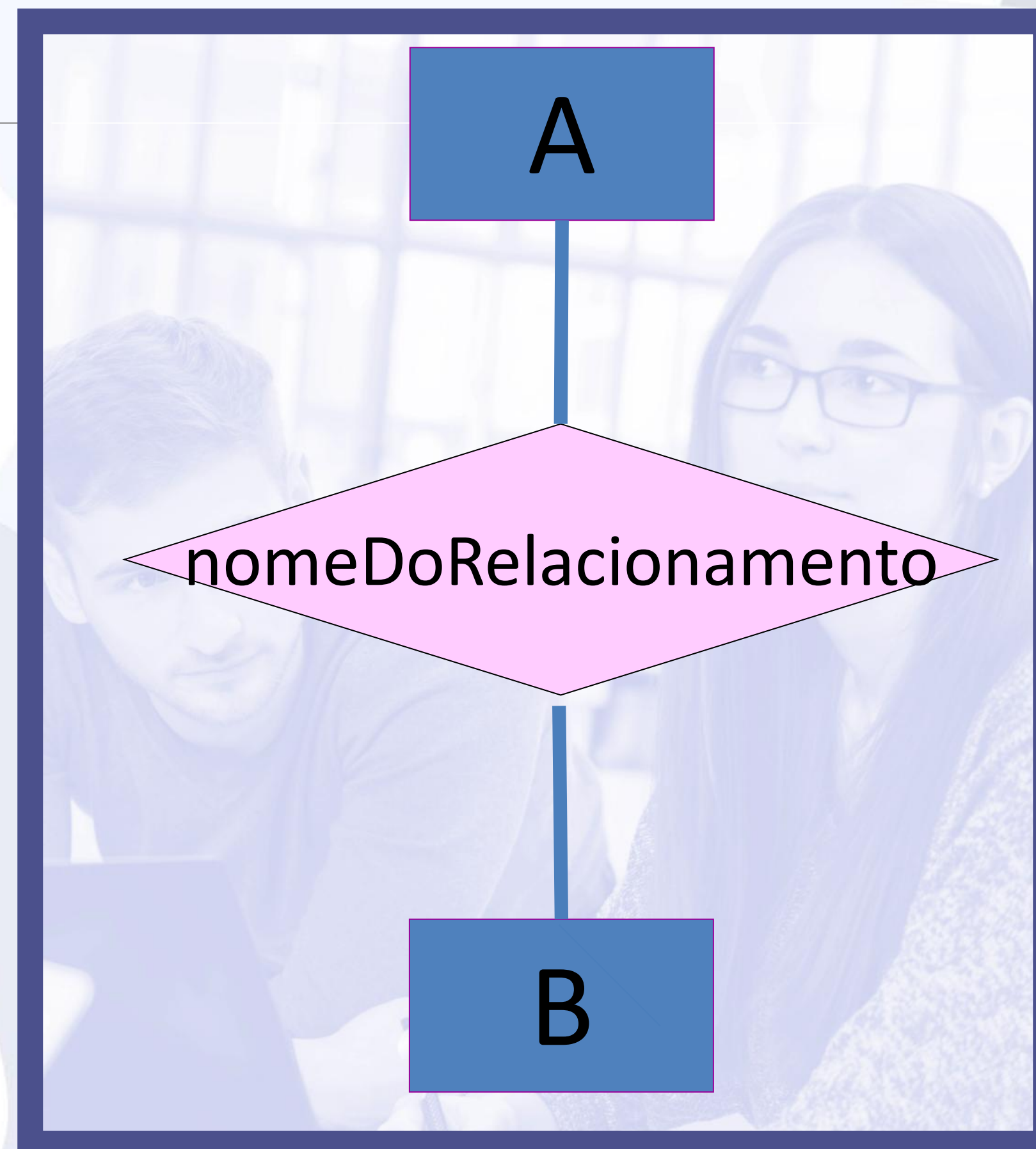




# Relacionamentos



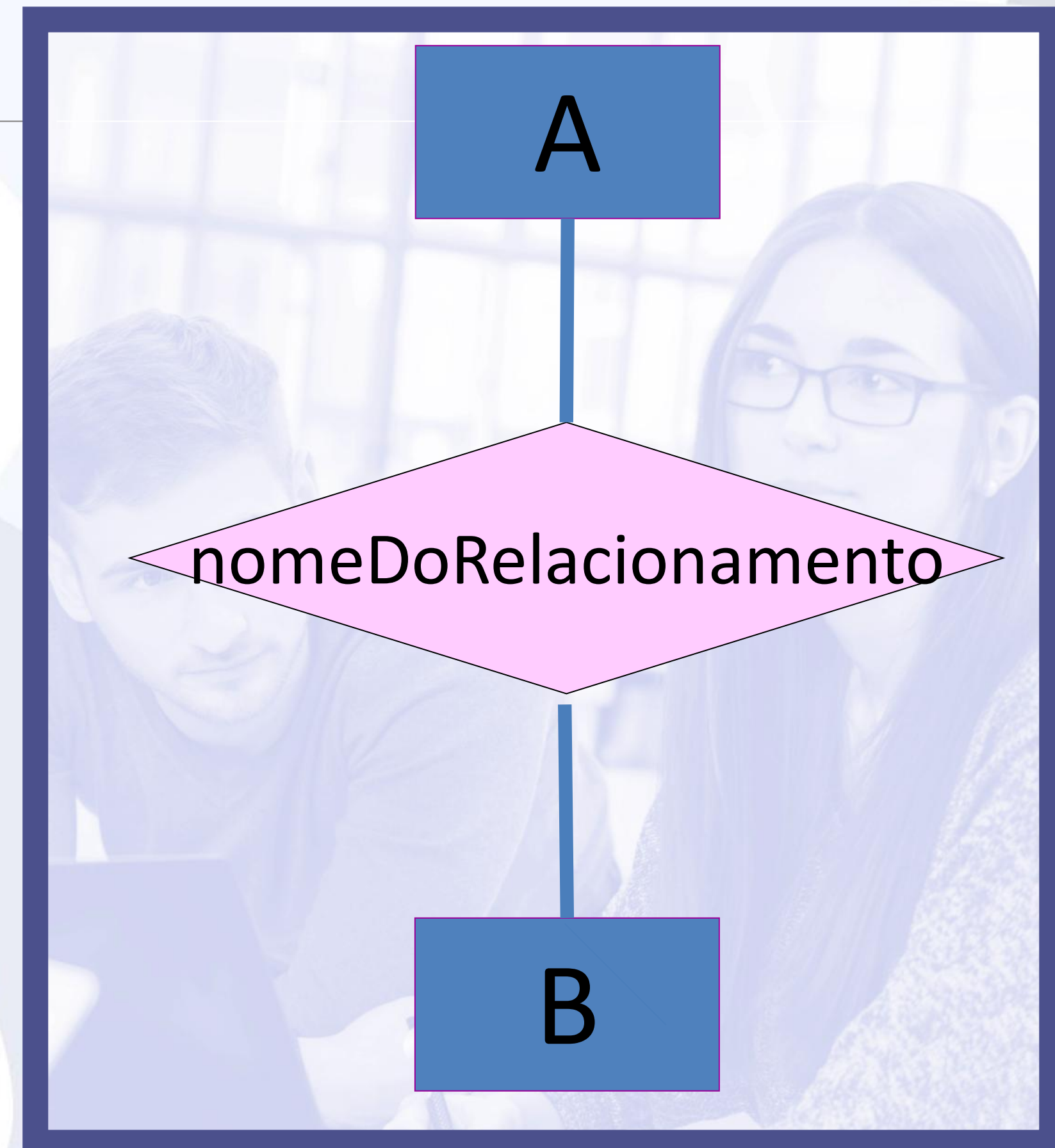
- É uma associação entre entidades
- Representado através de um losango e linhas que ligam as entidades relacionadas
- O nome do relacionamento é em geral apresentado como um tempo verbal, uma vez que simboliza a “ação” estabelecida entre as entidades envolvidas;





# Relacionamentos

- Os relacionamentos são classificados de acordo com o seguinte conjunto de características;
  - **Grau** – Número de entidades envolvidas no relacionamento;
  - **Obrigatoriedade (Cardinalidade Mínima)**
    - Revela o caráter opcional ou obrigatório com que as entidades participam do relacionamento;
  - **Cardinalidade (Cardinalidade Máxima)**
    - Relação entre o número de ocorrências de uma entidade com as respectivas ocorrências na outra entidade com que tem o relacionamento;

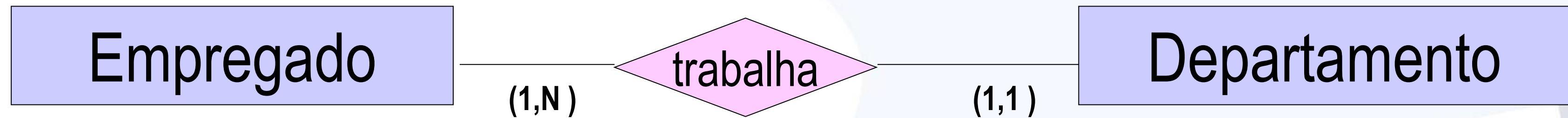




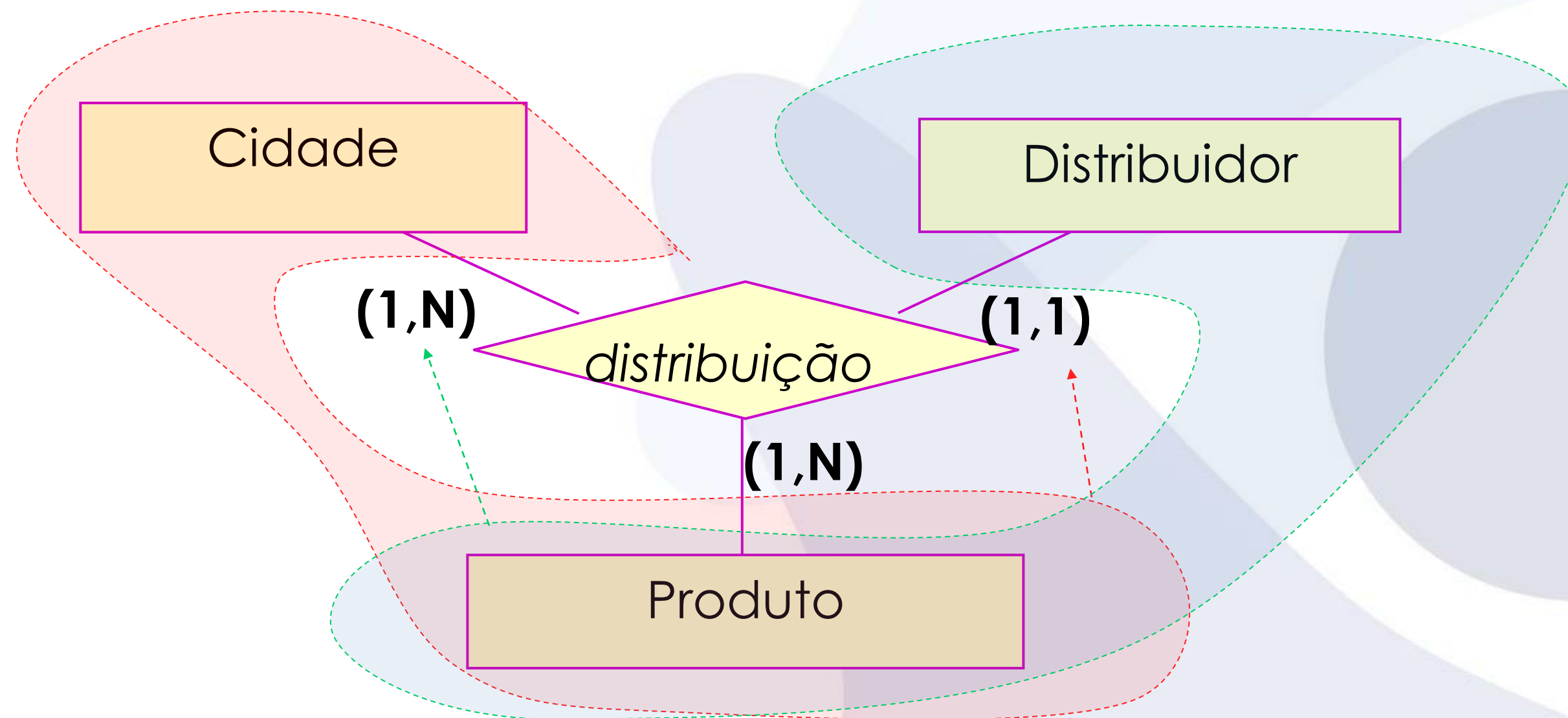
# Relacionamento Binário e Ternário



## Binário



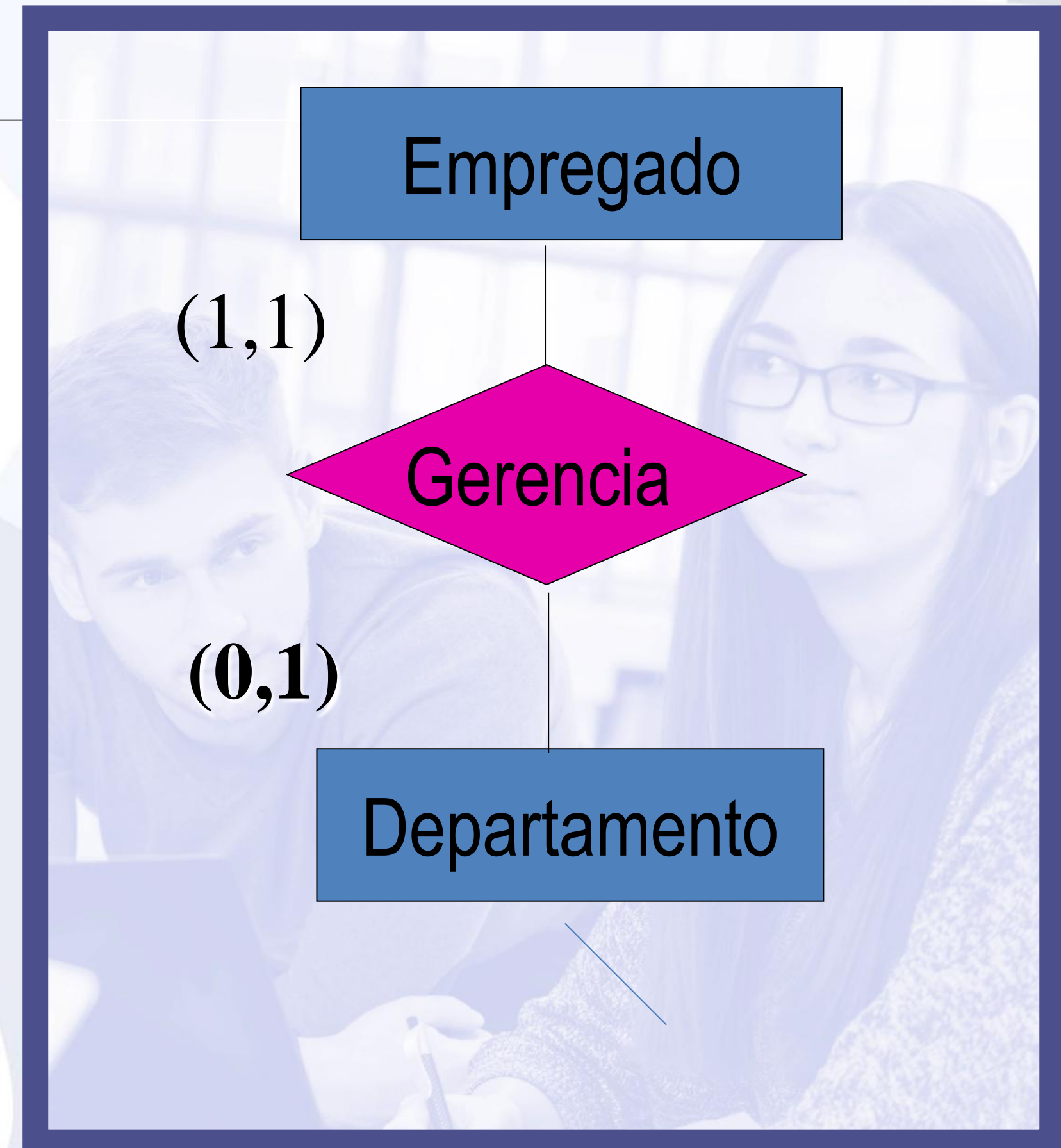
## Ternário



# Razões de Cardinalidade



- O modelo ER permite expressar cardinalidades mínimas e máximas em cada relacionamento
  - **Representação:**
    - Cardinalidades Possíveis: **(1,1)**; **(1,N)**; **(0,1)**; **(0,N)**
  - Cardinalidade **mínima** = 1 (relacionamento obrigatório)
  - Cardinalidade **mínima** = 0 (relacionamento opcional)

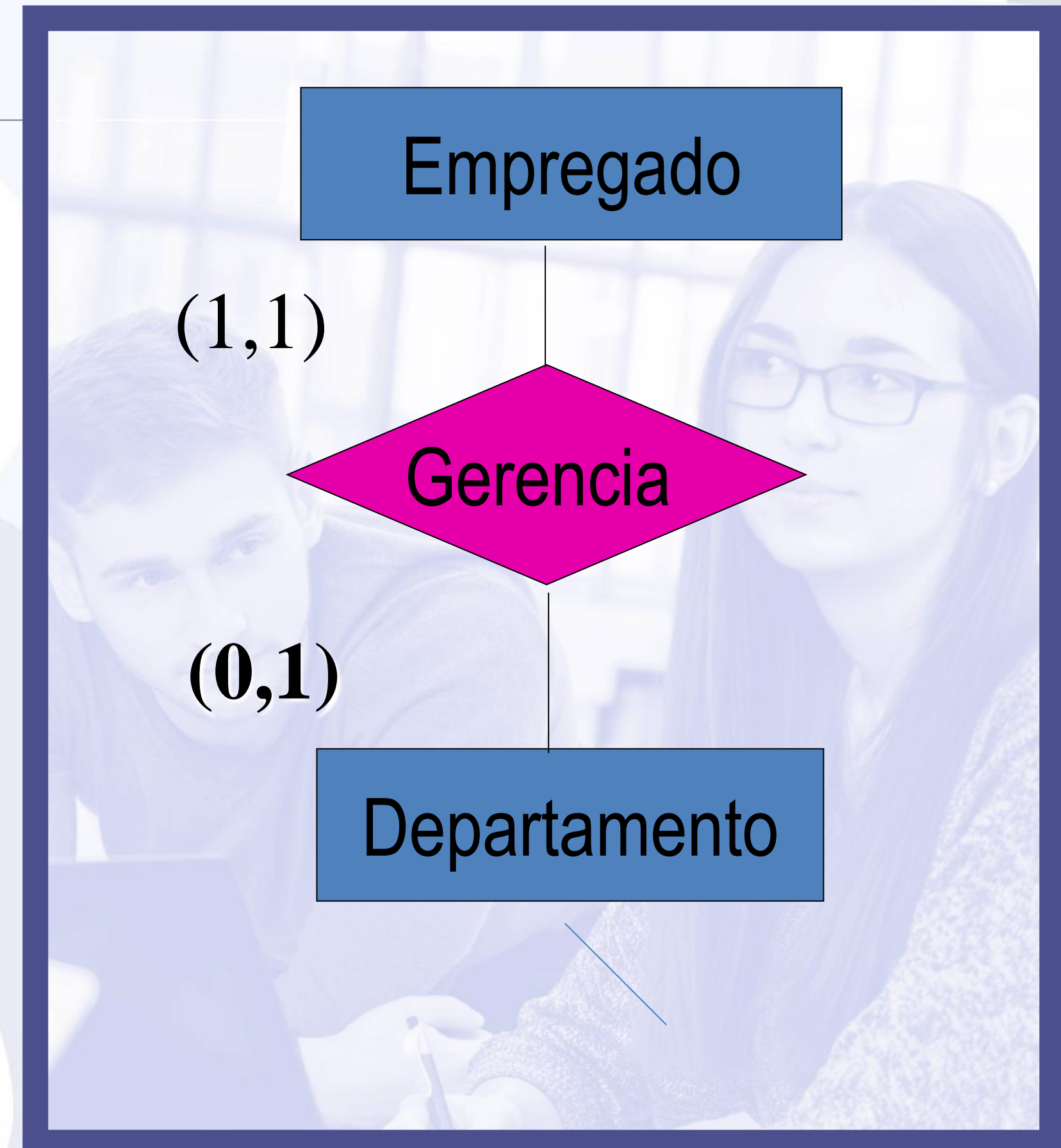




# Razões de Cardinalidade



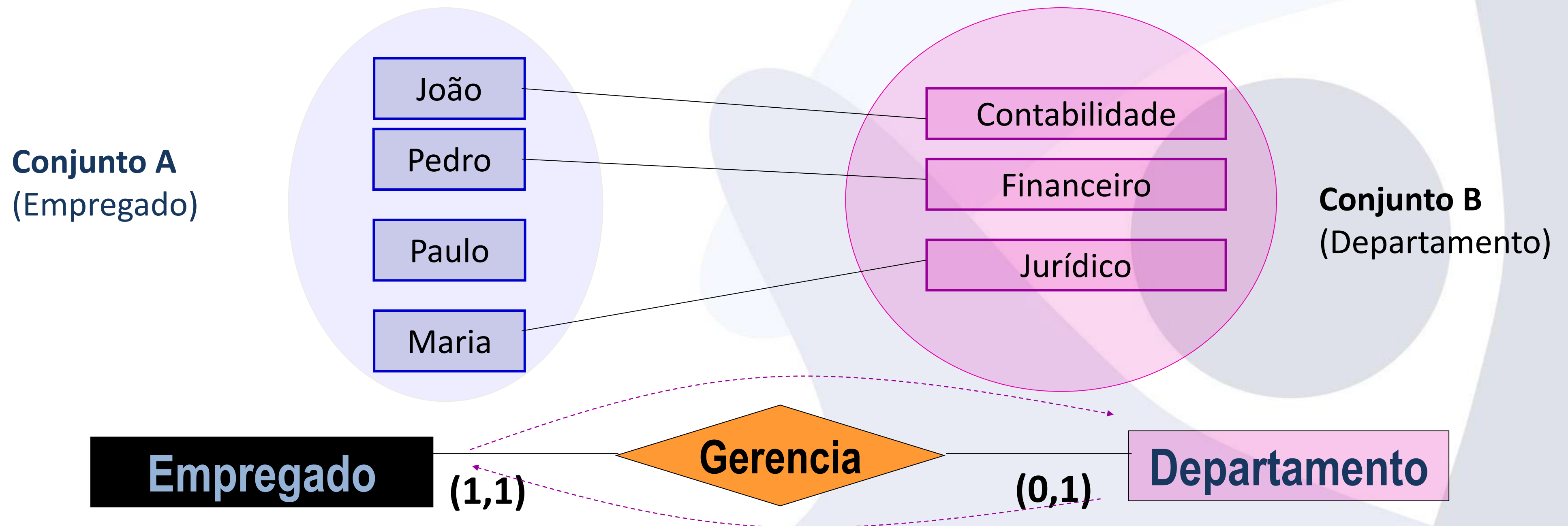
- Exemplo de Relacionamento **Obrigatório:**
  - Cada departamento é gerenciado por no mínimo um empregado.
- Exemplo de relacionamento **Opcional:**
  - Cada empregado pode gerenciar um departamento mais não é obrigatório.



# Relacionamento Um para Um – 1:1



- Existe um relacionamento de 1:1 entre as entidades A e B quando para cada ocorrência da entidade A ocorre no máximo uma ocorrência da entidade B, e para cada ocorrência de B existe no máximo uma ocorrência de A.

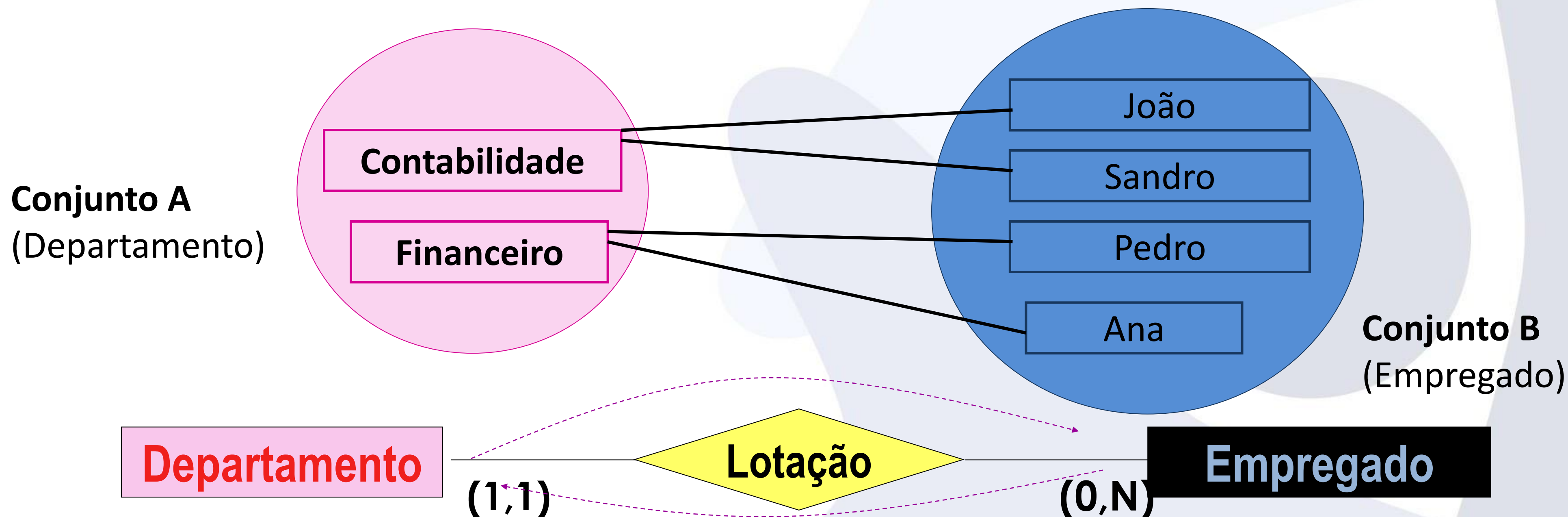




# Relacionamento Um para Muitos – 1:N



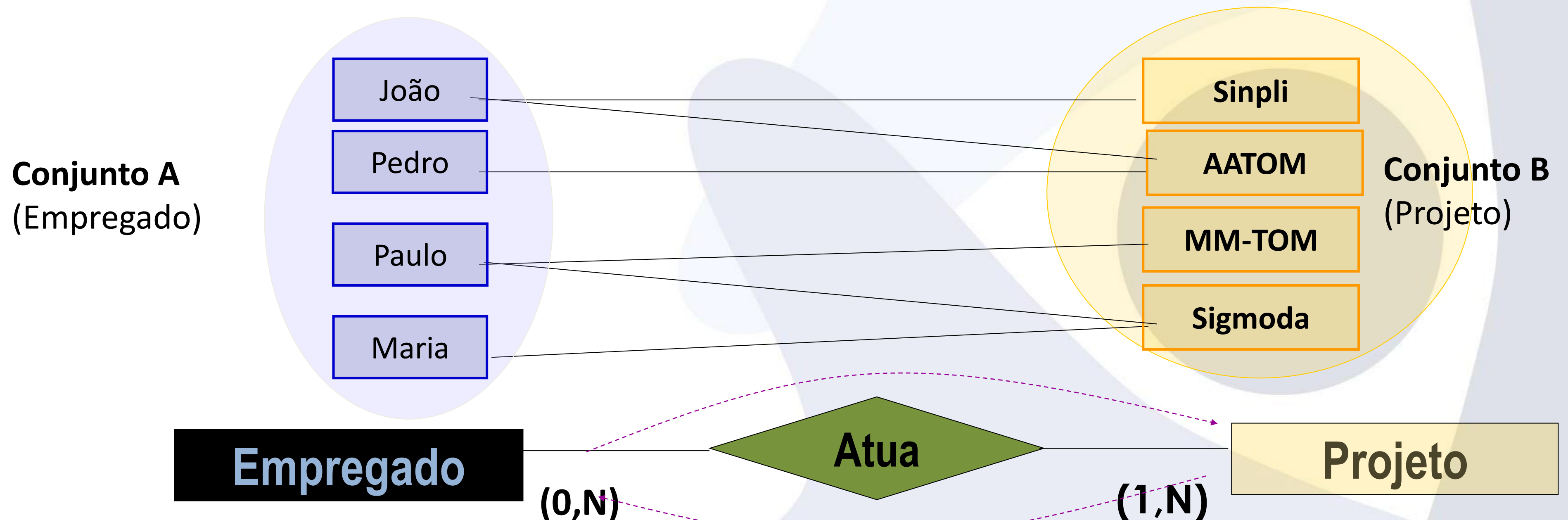
- Existe um relacionamento do tipo 1:N entre as entidades A e B quando para cada ocorrência da entidade A podem existir duas ou mais ocorrências na entidade B, e cada ocorrência de B existe, quando muito, uma ocorrência associada em A;



# Relacionamento Muitos para Muitos – M:N ou N:N



- Para cada ocorrência da entidade A puderem estar associadas duas ou mais ocorrências da entidade B, e para cada ocorrência da entidade B puderem estar associadas duas ou mais ocorrências da entidade A.



# Atributos também podem ter Cardinalidade



**Cardinalidade mínima**

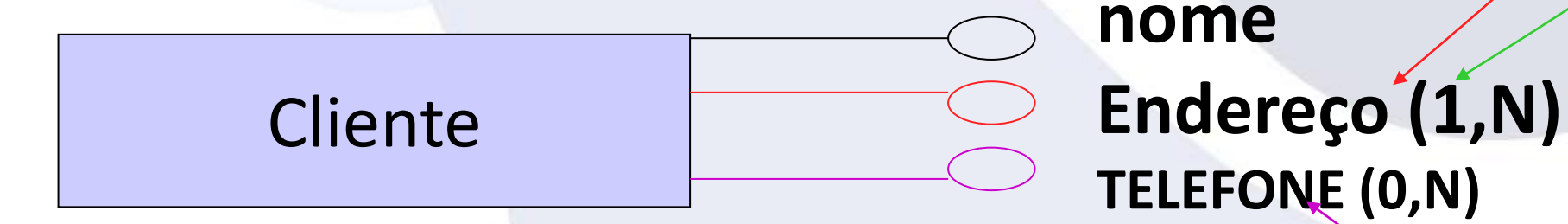
**1: atributo obrigatório**

**0: atributo opcional**

**Cardinalidade máxima**

**1: atributo monovalorado**

**N: atributo multivalorado**





# Resumo da notação

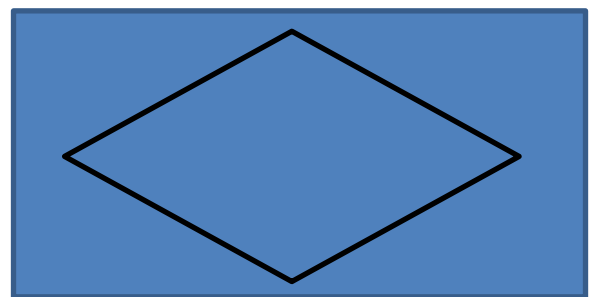
---



**Entidade primária.** Usado para definir as entidades fortes. São entidades que não dependem de nenhuma outra para existir.



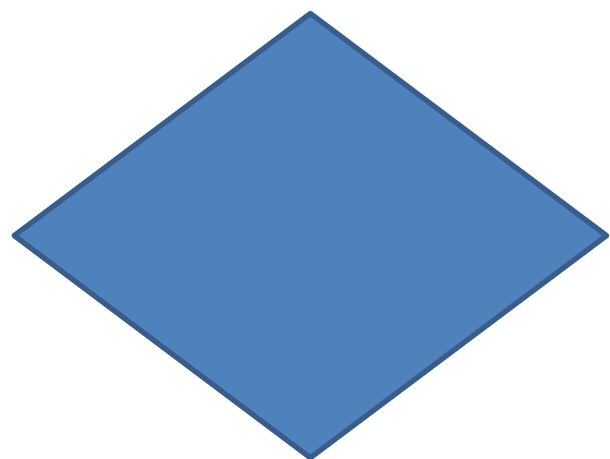
**Entidade fraca.** Entidades fracas são dependentes de outras entidades para existir. Não possuem chave primária.



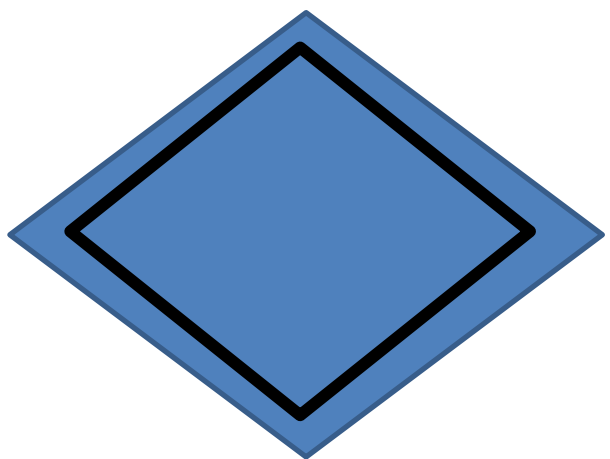
**Entidade associativa.** Associam as instâncias de vários tipos de entidades. Elas contêm atributos voltados especificamente ao relacionamento entre essas instâncias.

# Resumo da notação

---

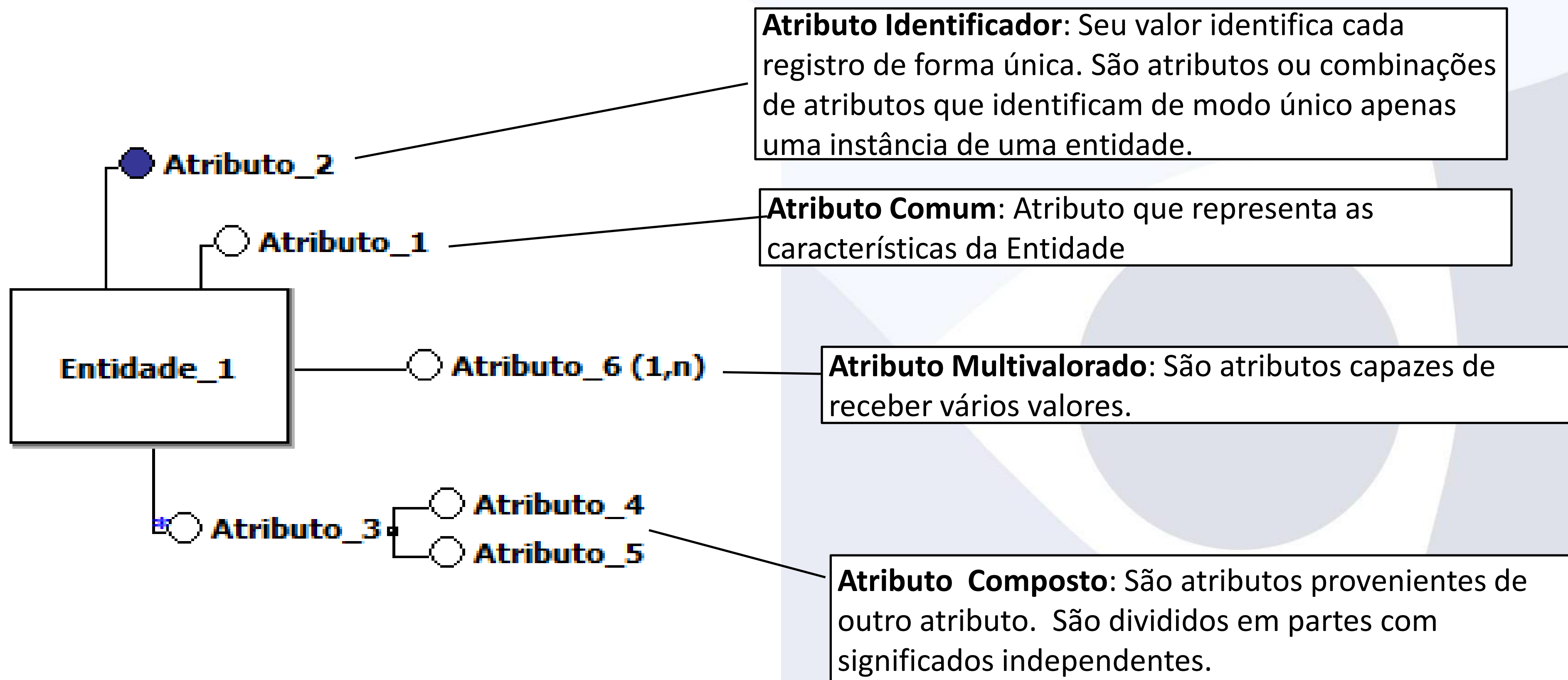


**Relacionamento.** Símbolo utilizado que representa o relacionamento associações entre entidades.



**Relacionamento fraco.** Também chamado de relacionamento de identificação, são relacionamentos entre um tipo de entidade fraca e seu proprietário, ou seja, uma entidade primária.

# Resumo da notação - Atributo





# Exemplo de modelagem

## - Requisitos

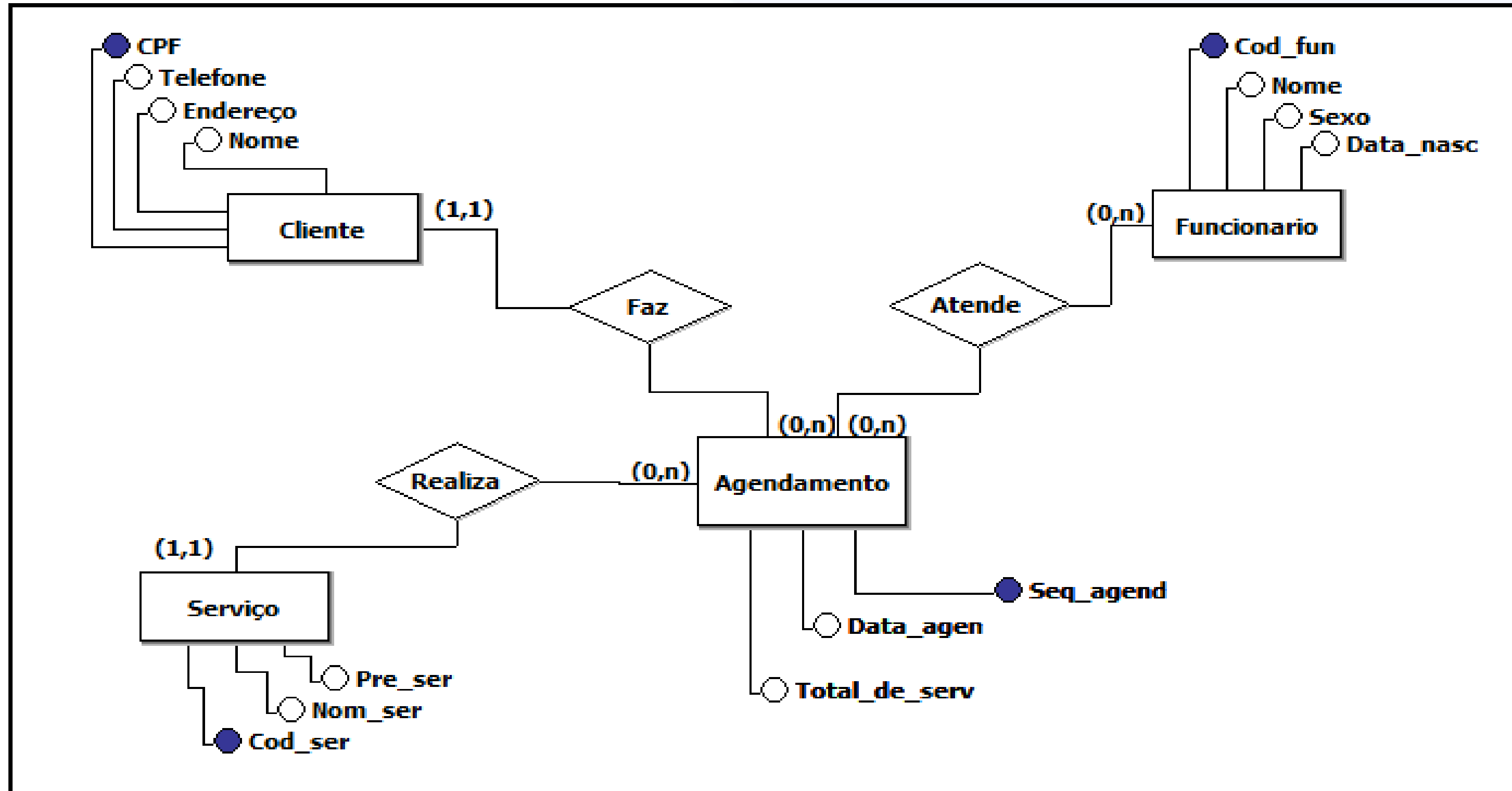
---



- O sistema deve cadastrar dados sobre Clientes. Com atributos CPF, Nome, telefone e endereço;
- O sistema deve registrar os dados dos funcionários. Código, nome Sexo e Data do nascimento;
- O sistema deve registrar os serviços oferecidos;
- Deve ser registrado os agendamentos realizados;
- Cada cliente pode fazer muitos agendamentos, mas um agendamento só terá um cliente;
- Um agendamento pode está ligado a mais de um funcionário, e um funcionário pode está ligado a mais de um agendamento;
- Um agendamento estará ligado a um serviço, mais um serviço pode está ligado a vários agendamentos;
- Um agendamento estará ligado a um cliente, mais um cliente pode está ligado a vários agendamentos;

# Exemplo de modelagem

## - Diagrama



<https://youtu.be/8V1066JYMmM>

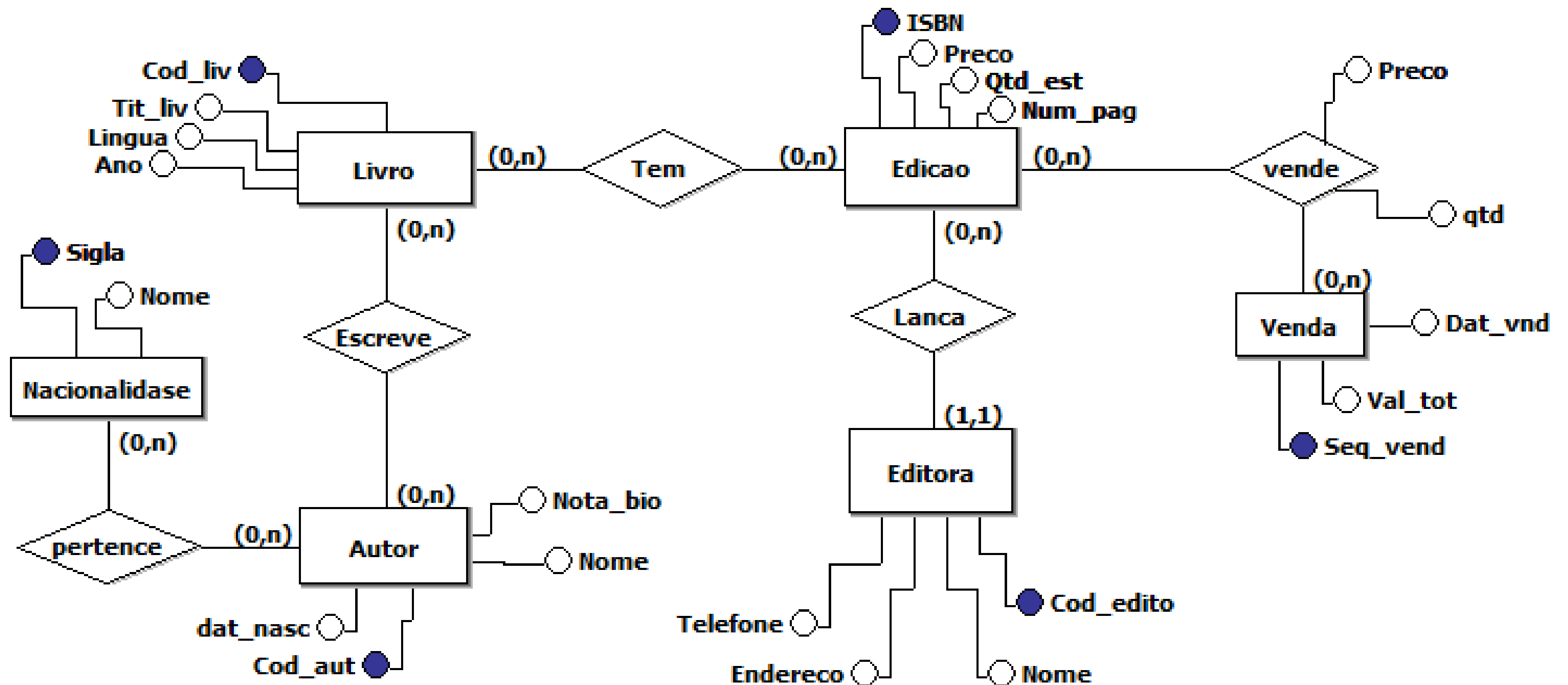
# Exercício - Livraria



- Uma livraria mantém um cadastro dos livros disponíveis para a venda.
- Para cada livro são armazenados código, título, língua e ano em que foi escrito.
- Para os autores é mantido igualmente um cadastro que inclui nome, data de nascimento, país de nascimento e uma breve nota biográfica.
- Cada autor pode apresentar mais de uma nacionalidade.
- Um autor pode ter mais de uma nacionalidade.
- Cada livro pode ter vários autores, ou nenhum autor, e para um mesmo autor podem existir vários livros cadastrados.
- Um autor pode estar incluído no cadastro ainda quando não exista um livro seu para venda.
- As editoras são incluídas no cadastro a partir do seu nome, endereço e telefone.
- Uma editora pode estar cadastrada mesmo quando não existam livros editados por ela em venda.
- Para um mesmo livro podem existir várias edições realizadas por editoras diferentes ou em anos diferentes.
- Cada edição tem um código (ISBN) , preço, ano, número de páginas e quantidade em estoque.
- Considere que um livro pode ser cadastrado se existe pelo menos uma edição do mesmo para venda.
- Deve ser lembrado que, o que será vendido é um exemplar de alguma edição e não livro.



# Exercício - Livraria



# Modelo relacional



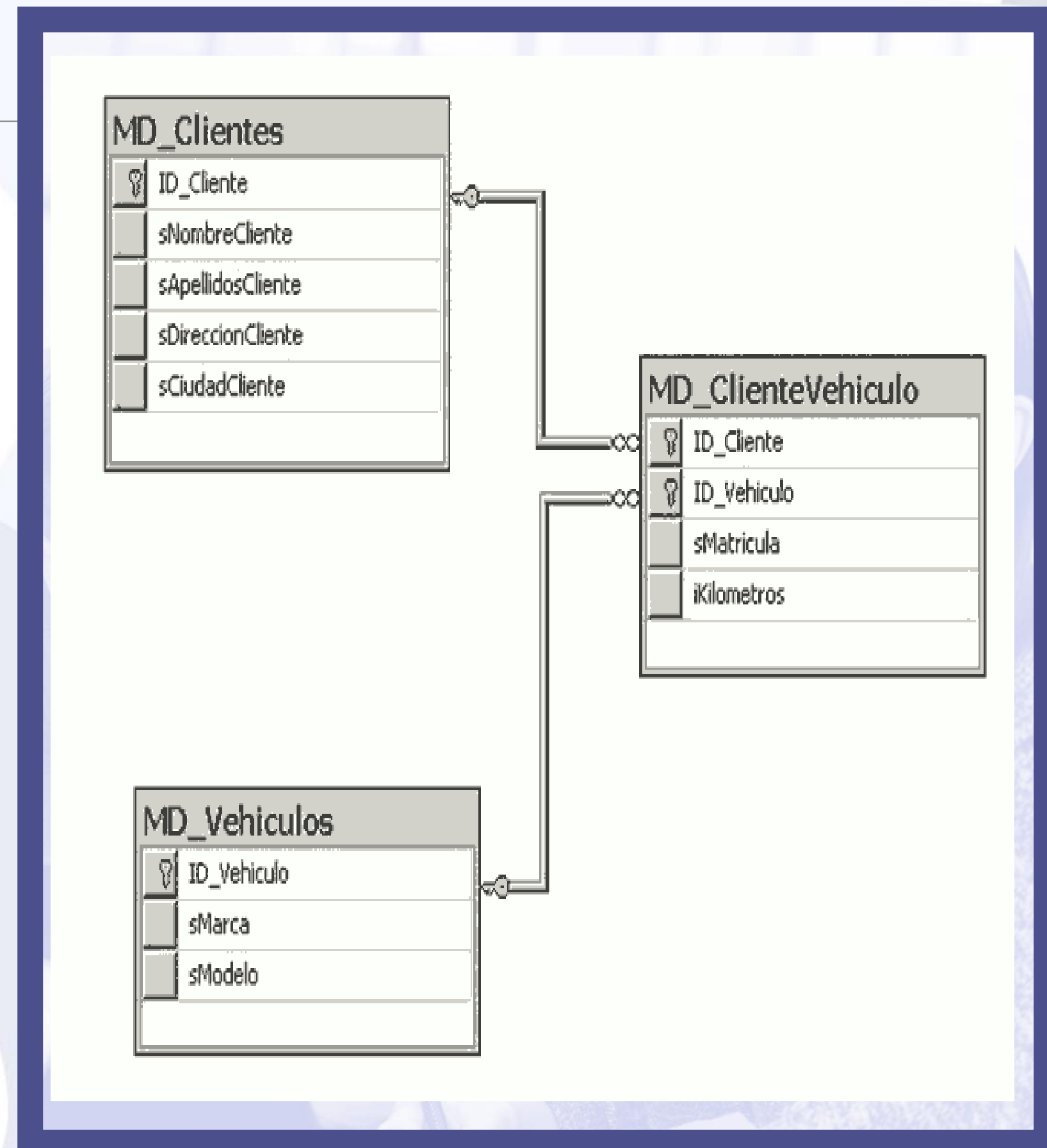
**ser**  
educacional

gente criando o futuro



# Modelo Relacional

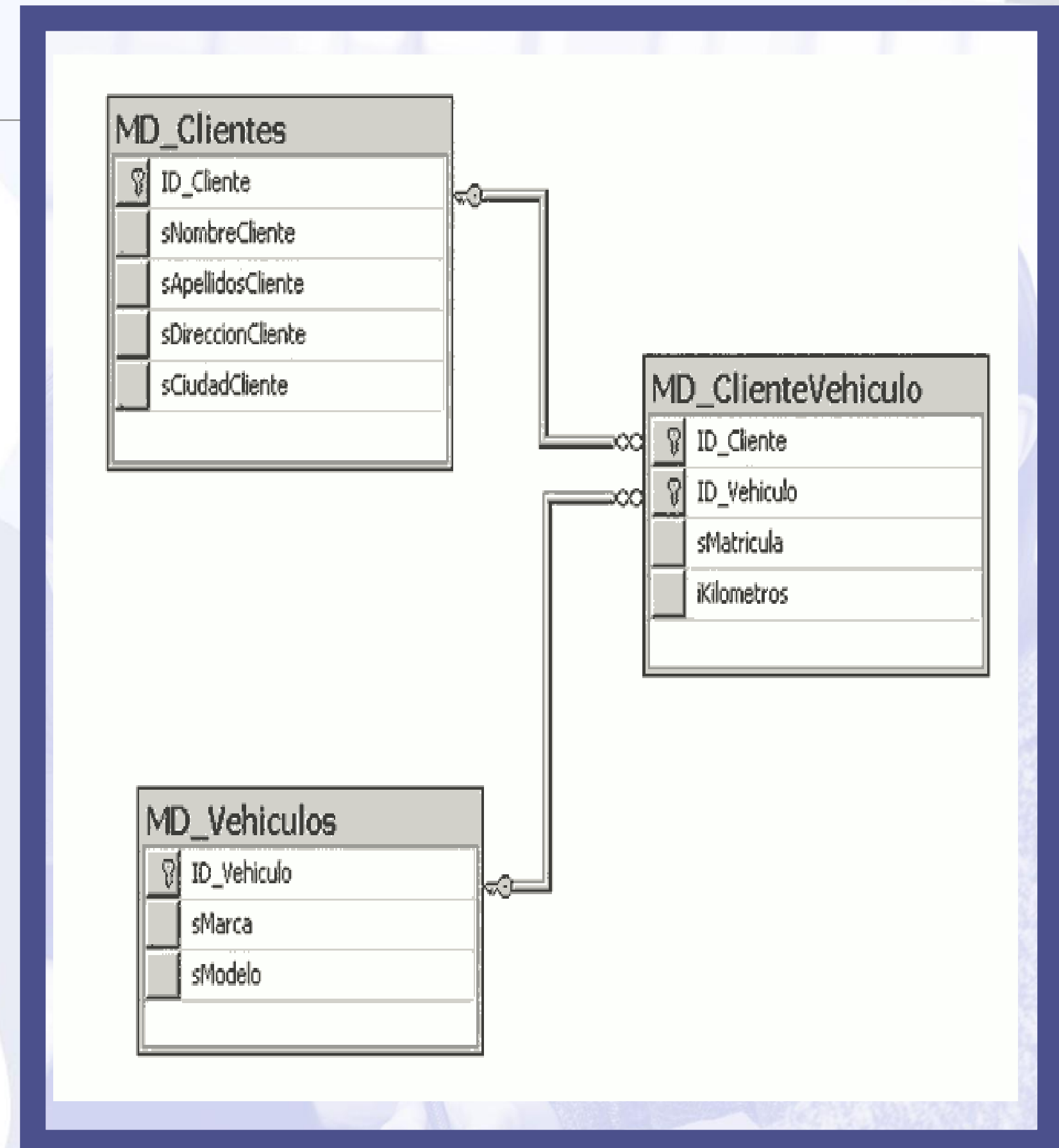
O **modelo relacional** é fundamentado na **teoria dos conjuntos e lógica de predicado**, que são conceitos matemáticos. Alguns dos primeiros sistemas comerciais de banco de dados baseados em modelo relacional dos anos 80 foram o Oracle, MySql, Access, entre vários outros (ELMASRI; NAVATHE, 2011).





# Modelo Relacional

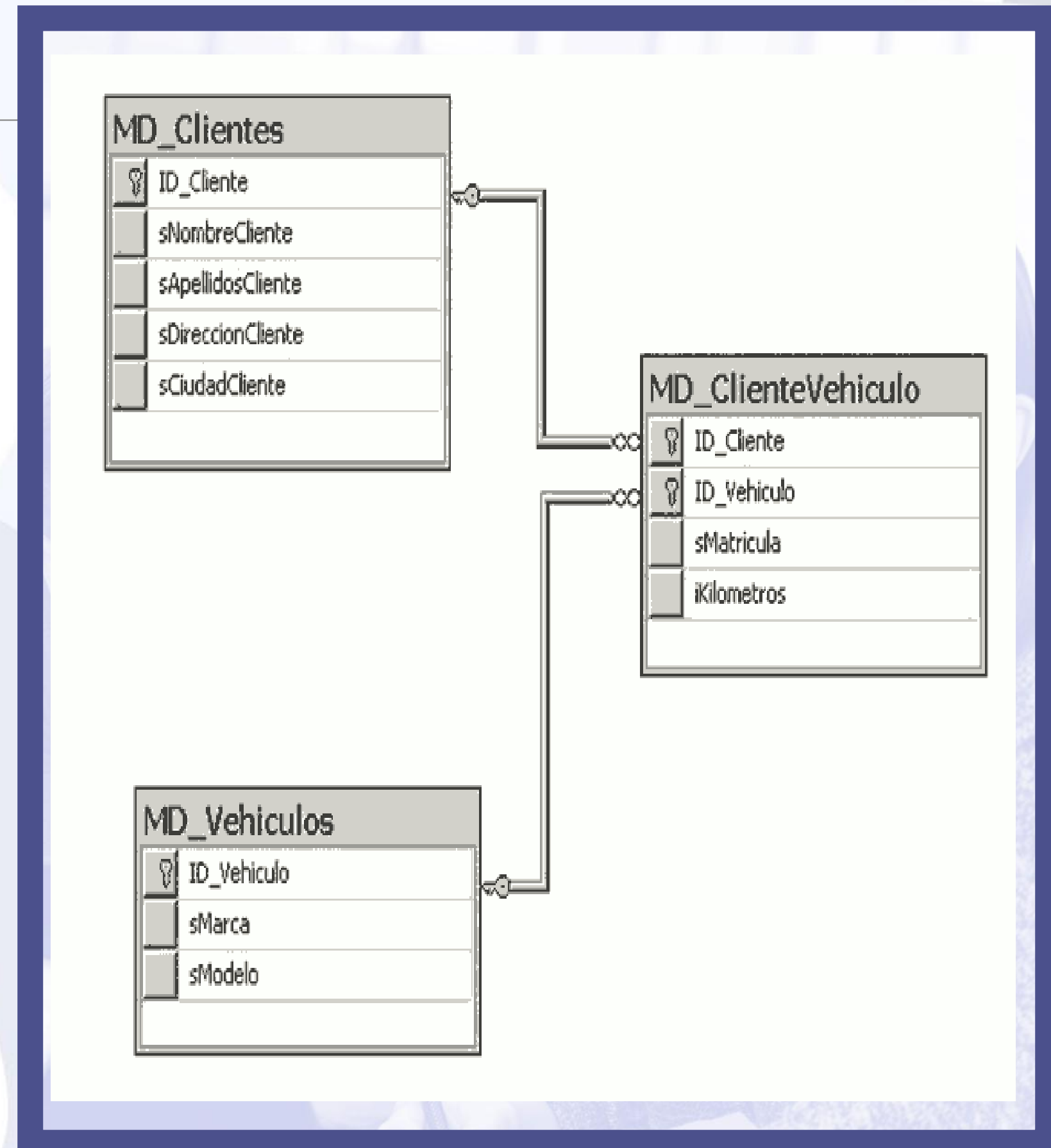
- Enquanto o modelo entidade-relacionamento é voltado para o projeto conceitual do banco de dados, o modelo relacional está diretamente ligado ao **projeto lógico**; assim sendo, essa fase é dependente da conclusão do projeto conceitual.
- O modelo conceitual é um modelo **abstrato** que descreve a estrutura de um banco de dados independente de um **SGBD**. J
- O modelo **lógico** é aquele com dados que representam a estrutura com foco em um SGBD específico,





# Modelo Relacional

- O modelo relacional refere-se a três aspectos principais: **aspecto estrutural, de integridade e de manipulação de dados** (DATE, 2003). No **aspecto estrutural**, o banco de dados é representado como um conjunto de relações, similar a uma tabela, com linhas e colunas associadas. As linhas são denominadas **tuplas** e em cada coluna são distribuídos os **atributos**, como se fossem títulos para as colunas (ELMASRI; NAVATHE, 2011).



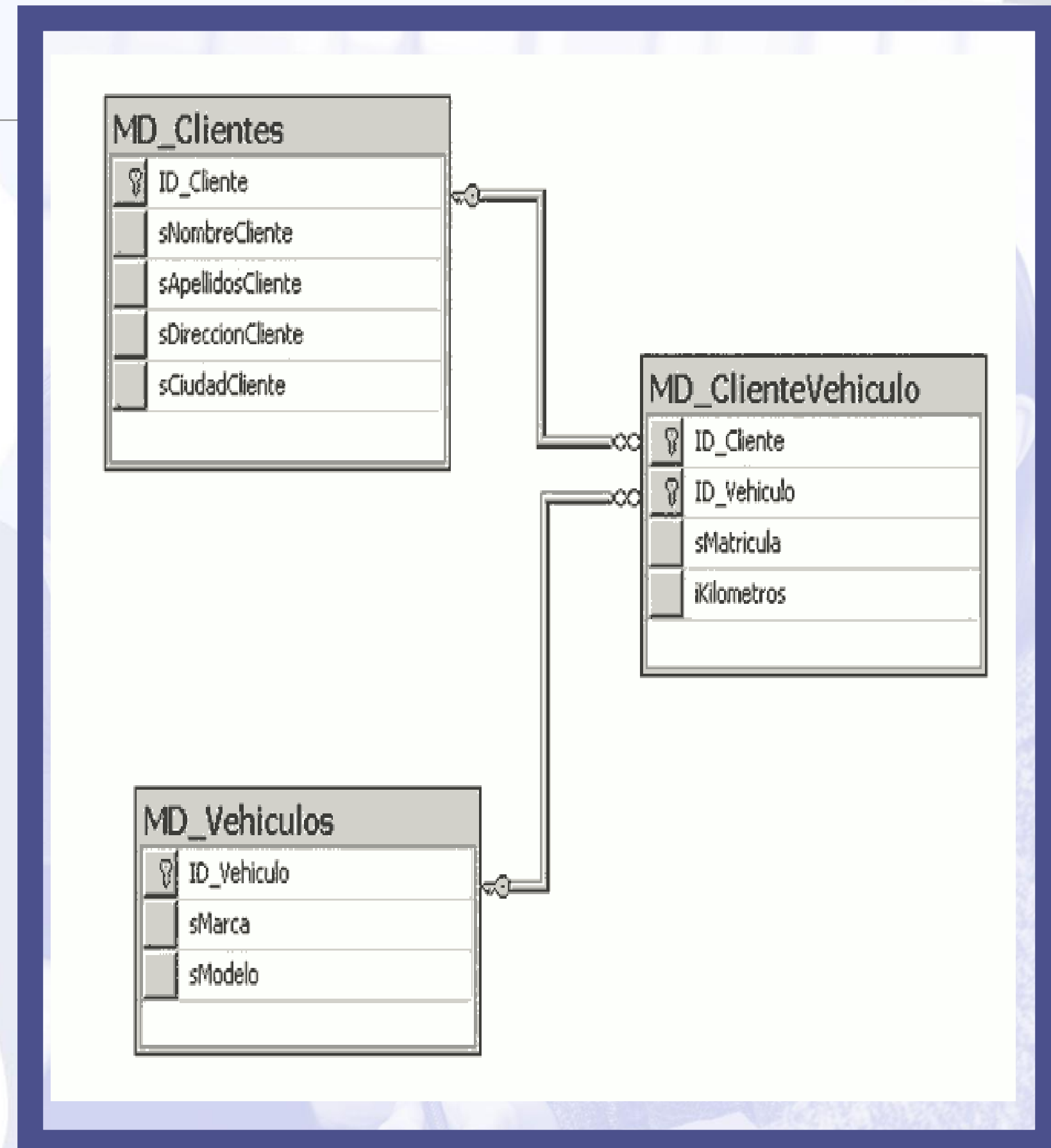
# Relação / Tabelas

A **relação** é a estrutura fundamental do modelo; o esquema da relação é constituído por um ou mais atributos; o número de atributos é fixo (grau da relação); Os atributos não são ambíguos; do cruzamento de uma coluna com uma linha resulta apenas um único valor.

## Existem dois tipos:

Relações base

Relações virtuais (Views)





# Atributos – Tuplas - Linhas



NOME DA RELAÇÃO

ATRIBUTOS

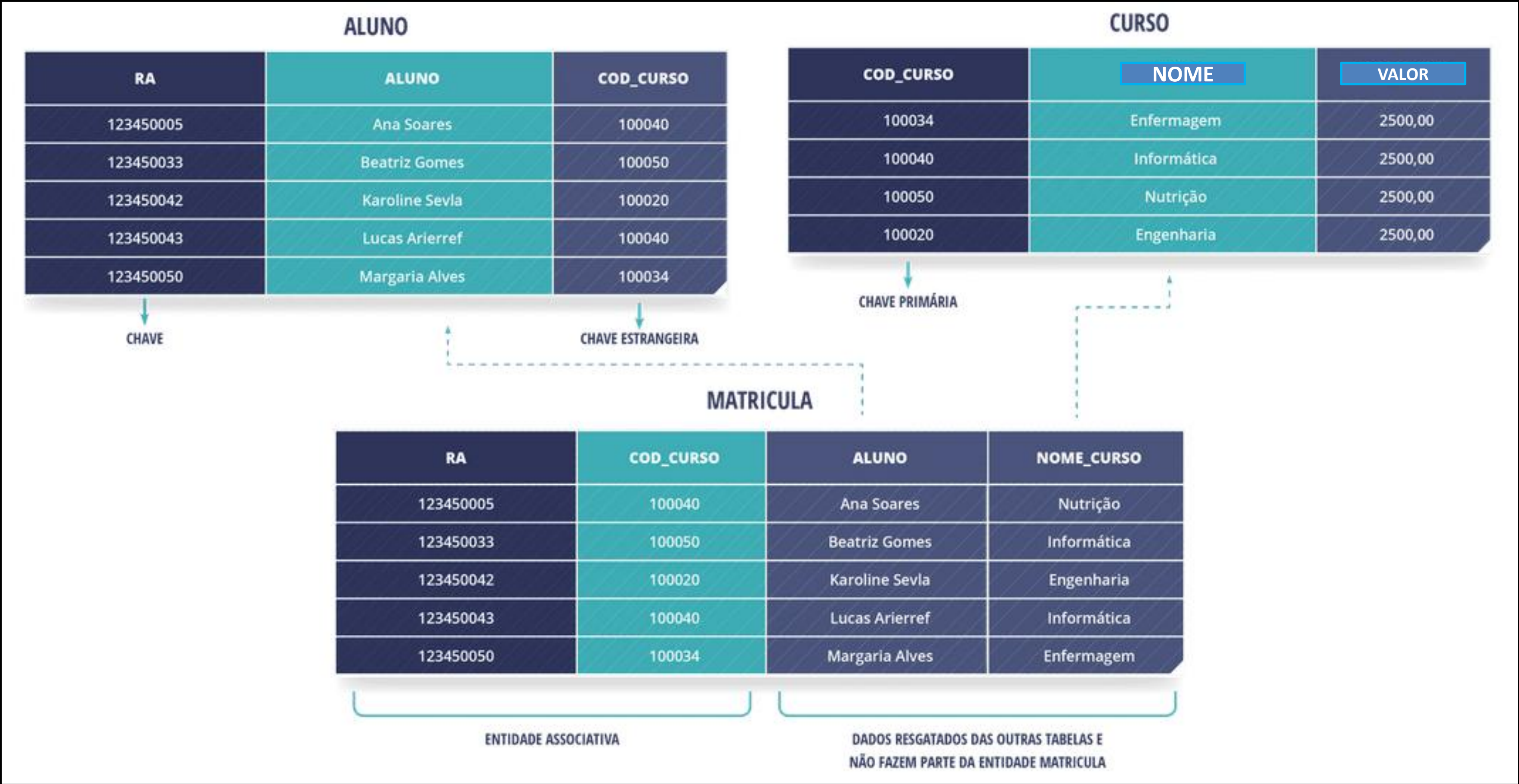
MATRICULADOS

RA	ALUNO	COD_CURSO	NOME_CURSO
123450001	Bartolomeu Silva	100034	Enfermagem
123450005	Ana Soares	100040	Informática
123450009	Julia dos Santos	100034	Enfermagem
123450033	Beatriz Gomes	100050	Nutrição
123450042	Karoline Sevla	100020	Engenharia Elétrica
123450043	Lucas Arierref	100040	Informática
123450050	Margaria Alves	100034	Enfermagem

TUPLAS



# Relacionamento lógico



# Restrições de Integridade



**Restrições de Domínio** - Definir o conjunto de valores permitidos ou possíveis que um atributo pode ter. Também pode haver a integridade de vazios, que verifica se um campo pode receber um valor nulo (*null*) ou não.

**Restrições de chave** - Impede que uma chave-primária tenha repetições, garantindo assim a unicidade dos dados no banco de dados.





# Restrições de Integridade



## **Restrições de integridade referencial;**

Uma chave-estrangeira de uma relação tem que coincidir com uma chave-primária da entidade principal. Nesse caso, o atributo deve ser do mesmo tipo e existir em ambas as tabelas, assim como o conteúdo idêntico.

Poderá haver a chamada violação da integridade referencial, quando a chave-estrangeira não coincide com a chave-primária da entidade principal.



# Restrições de Integridade



## **Integridade definida pelo usuário;**

A integridade definida pelo usuário permite definir regras próprias e específicas voltadas ao negócio e que não se encaixam em outras categorias de integridade. Pode-se exemplificar restrições como:

Todo produto vendido precisa estar associado a um número de pedido.  
Toda nota-fiscal precisa ter no mínimo um item de venda associado a ela.



# Restrições de Integridade



- Por intermédio do relacionamento e da aplicação dos tipos de cardinalidades envolvidas, conseguimos determinar várias restrições de integridades. É importante notar que a forma de representar as cardinalidades pode variar de autor para autor, entre ferramentas CASE ou mesmo por convenção dos envolvidos na documentação do projeto de banco de dados, como: **(1,1)**, **(1,N)**, **(0,1)** e **(0,N)**. O M nem sempre é adotado e representa vários, assim como o N.





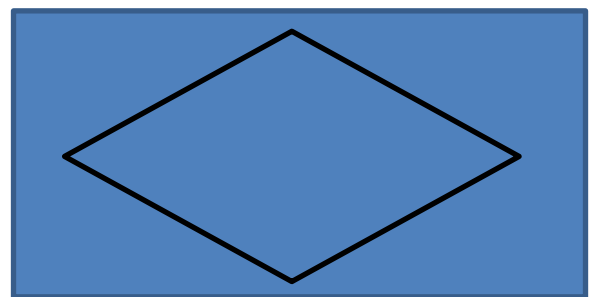
# Lembrando da notação do modelo conceitual



**Entidade primária.** Usado para definir as entidades fortes. São entidades que não dependem de nenhuma outra para existir.



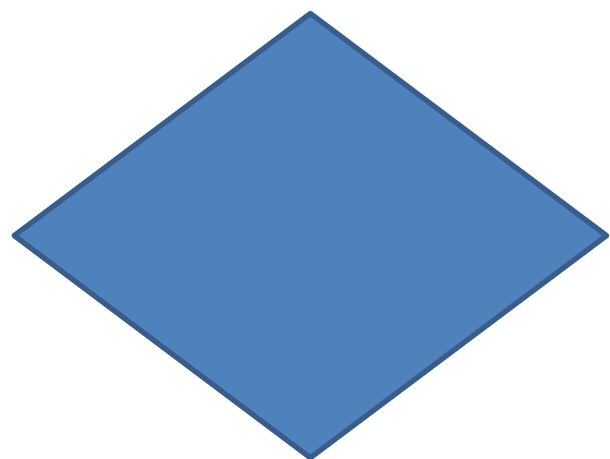
**Entidade fraca.** Entidades fracas são dependentes de outras entidades para existir. Não possuem chave primária.



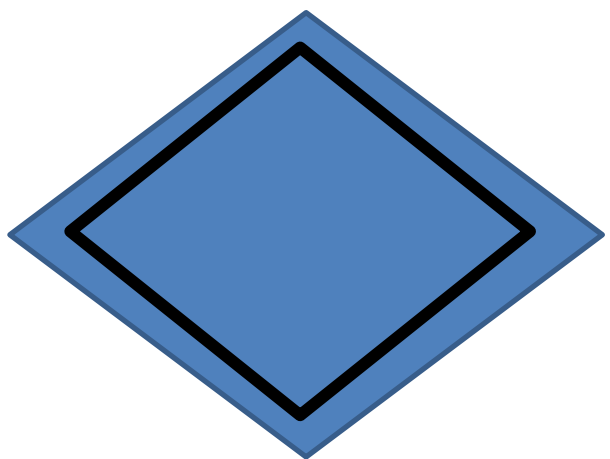
**Entidade associativa.** Associam as instâncias de vários tipos de entidades. Elas contêm atributos voltados especificamente ao relacionamento entre essas instâncias.

# Resumo da notação

---

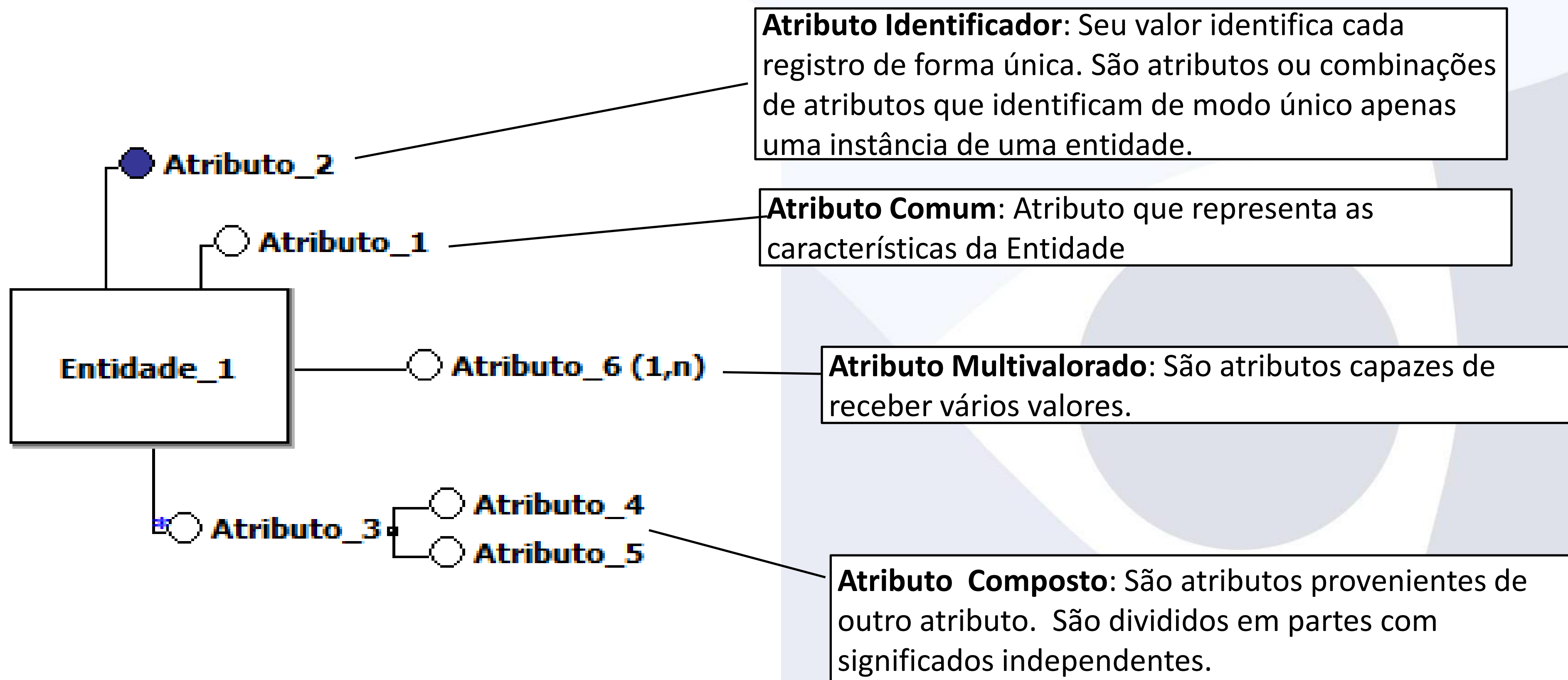


**Relacionamento.** Símbolo utilizado que representa o relacionamento associações entre entidades.



**Relacionamento fraco.** Também chamado de relacionamento de identificação, são relacionamentos entre um tipo de entidade fraca e seu proprietário, ou seja, uma entidade primária.

# Resumo da notação - Atributo





# Exemplo de modelagem

## - Requisitos

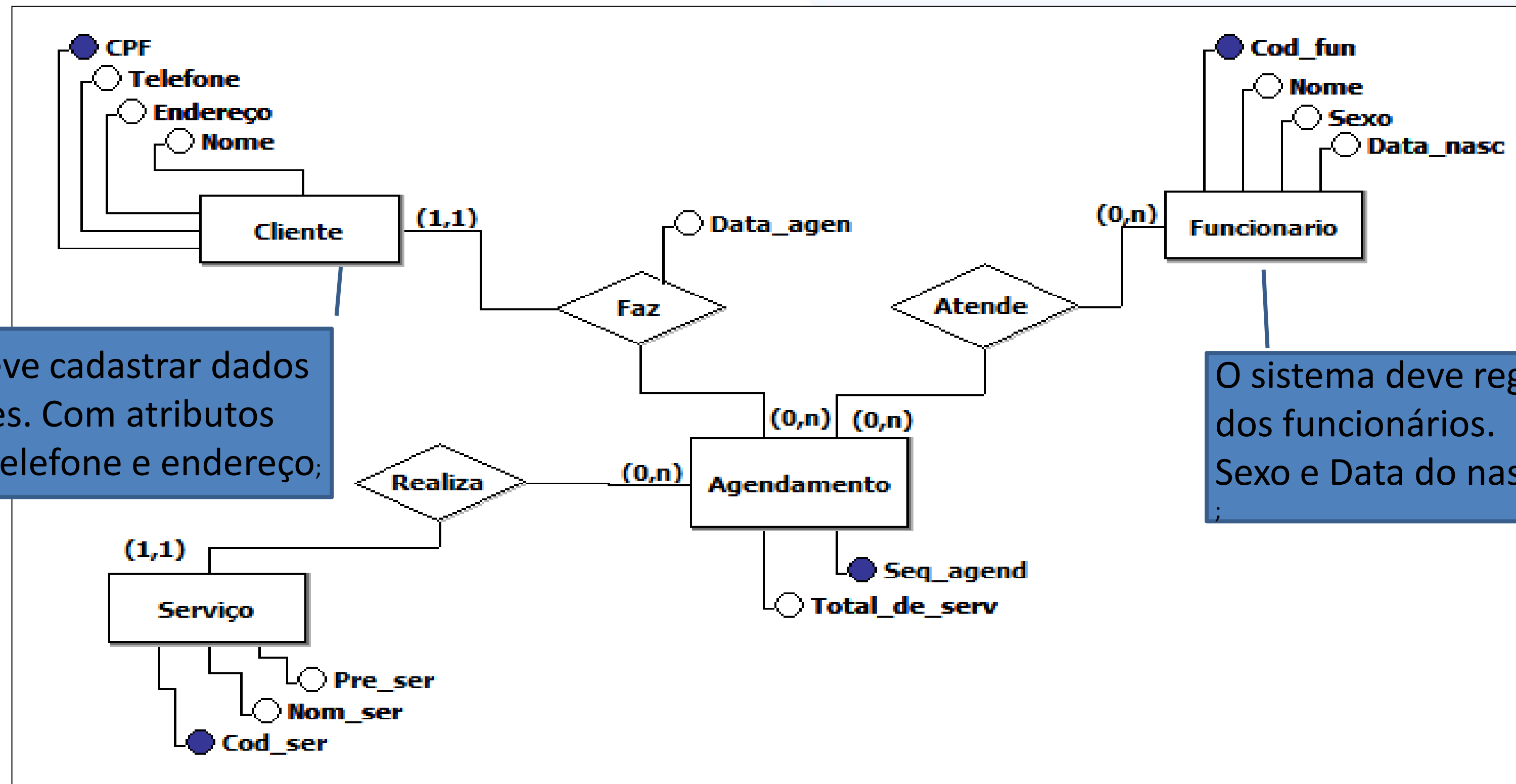
---



- O sistema deve cadastrar dados sobre Clientes. Com atributos CPF, Nome, telefone e endereço;
- O sistema deve registrar os dados dos funcionários. Código, nome Sexo e Data do nascimento;
- O sistema deve registrar os serviços oferecidos;
- Deve ser registrado os agendamentos realizados;
- Cada cliente pode fazer muitos agendamentos, mas um agendamento só terá um cliente;
- Um agendamento pode está ligado a mais de um funcionário, e um funcionário pode está ligado a mais de um agendamento;
- Um agendamento estará ligado a um serviço, mais um serviço pode está ligado a vários agendamentos;
- Um agendamento estará ligado a um cliente, mais um cliente pode está ligado a vários agendamentos;

<https://www.youtube.com/watch?v=kWYWLWLUWbjM&t=16s>

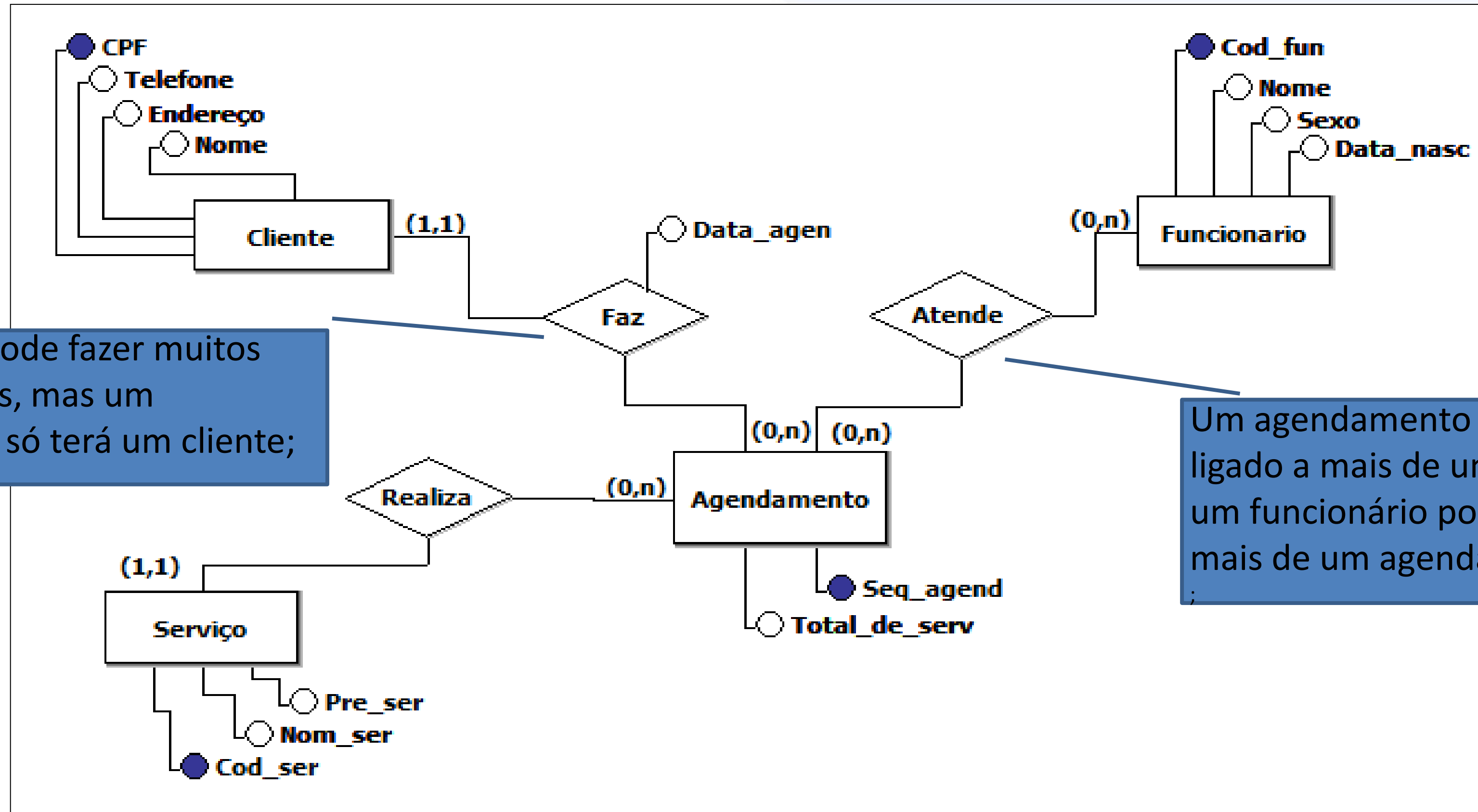
# Exemplo de modelagem - Diagrama



O sistema deve cadastrar dados sobre Clientes. Com atributos CPF, Nome, telefone e endereço;

O sistema deve registrar os dados dos funcionários. Código, nome Sexo e Data do nascimento;

# Exemplo de modelagem - Diagrama



Cada cliente pode fazer muitos agendamentos, mas um agendamento só terá um cliente;

Um agendamento pode está ligado a mais de um funcionário, e um funcionário pode está ligado a mais de um agendamento;



# Conversão Conceitual para lógico




MODELO CONCEITUAL	MODELO LÓGICO
Entidade	Tabela
Relacionamento 1:1 ou 1:N	Chave-estrangeira ou nova tabela
Relacionamento N:N	Nova tabela e duas chaves-estrangeiras
Relacionamento n-ário (ternário, quaternário etc.)	Nova tabela
Atributo simples	Atributo simples ou campos
Atributo composto	Conjunto de atributos simples
Atributo multivalorado	Tabela e chave-estrangeira ou N colunas
Conjunto de valores	Chamado de Domínio
Atributo-chave	Chave-primária


# MLD – Modelo Lógico de Dados



Entidades são mapeadas como Tabelas

Cliente
 CPF: char(11)
Telefone: char(13)
Endereço: varchar(100)
Nome: varchar(40)

Atributos Identificadores  
passam a ser chaves primárias




Serviço
 Cod_ser: Int
Nom_ser: varchar...
Pre_ser: real(15,2)

(1,1)

(0,n)



(0,n)

(1,1)

Agendamento
 Seq_agend: Int
Total_de_serv: real(15,2)
 CPF: char(11)
 Cod_ser: Int
Data_agen: date

O atributo de um relacionamento deve migrar  
para uma das tabelas do relacionamento


(0,n)

Atende
 Seq_agend: Int
 Cod_fun: Int

(0,n)

(1,1)

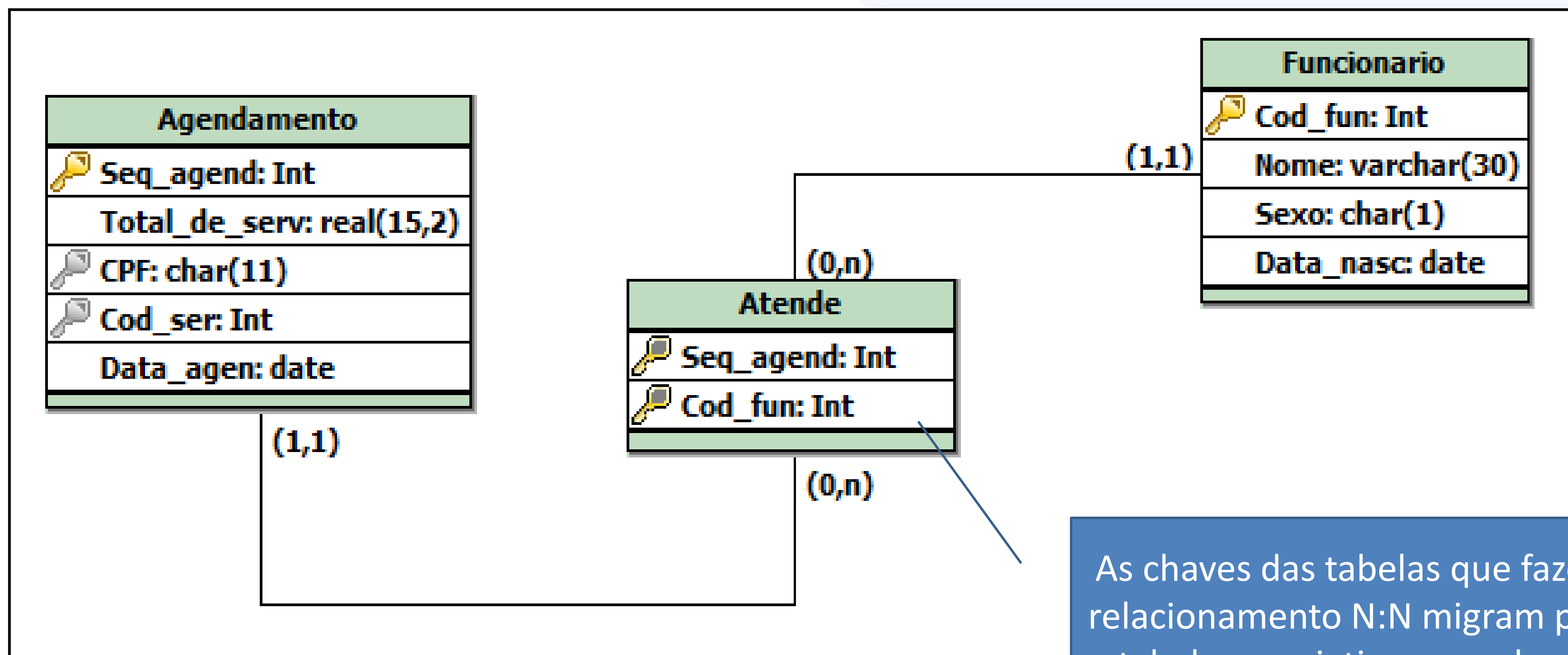
(1,1)

Funcionario
 Cod_fun: Int
Nome: varchar(30)
Sexo: char(1)
Data_nasc: date

Os relacionamentos N:N se  
transformam em tabelas

Para os relacionamentos 1:N  
os atributos se transformam  
em chaves primárias

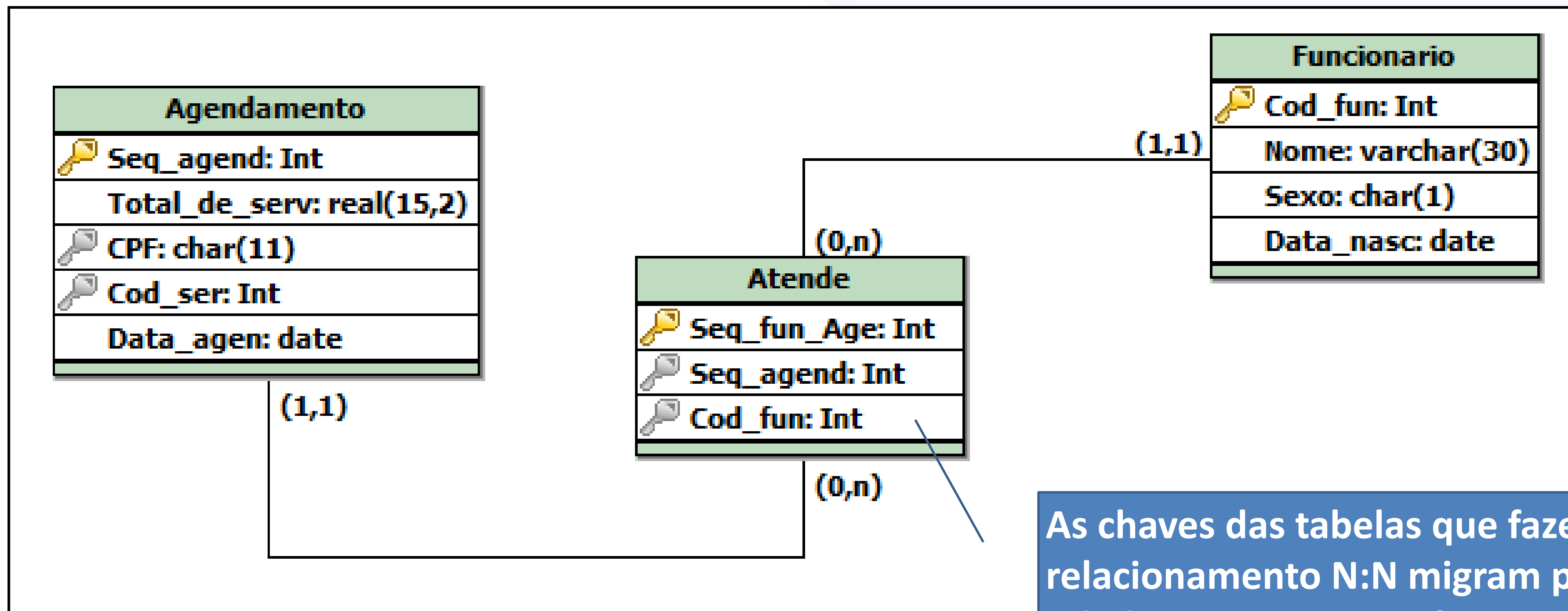
# MLD – Modelo Lógico de Dados



As chaves das tabelas que fazem o relacionamento N:N migram para a tabela associativa gerando uma Chave primária Composta



# MLD – Modelo Lógico de Dados

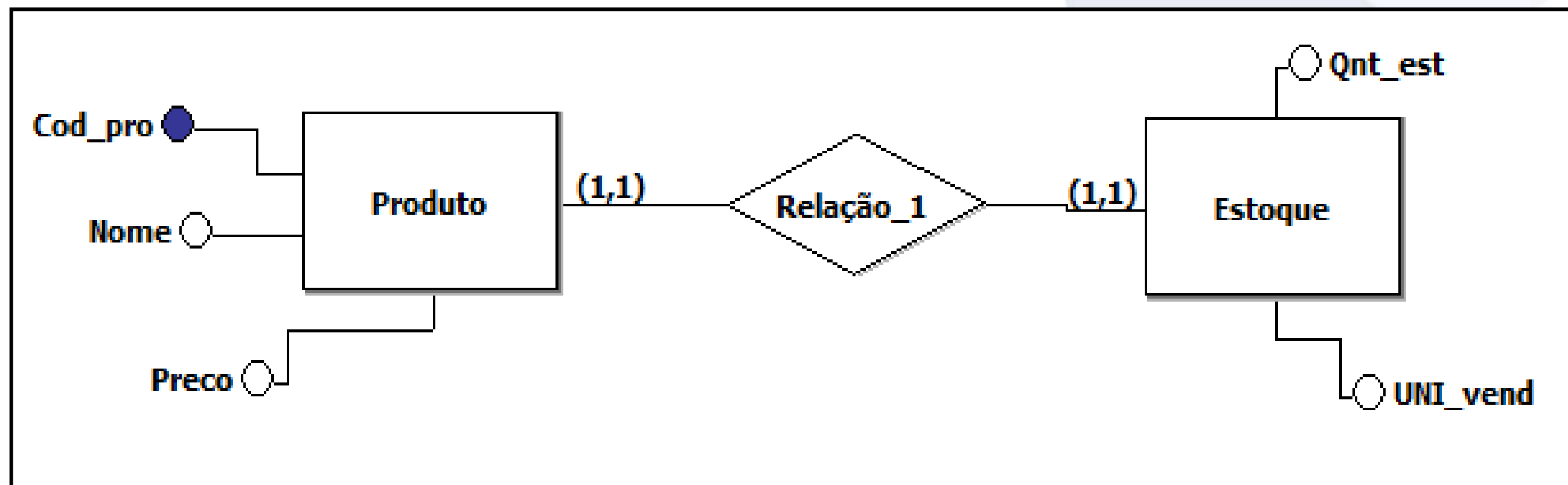



As chaves das tabelas que fazem o relacionamento N:N migram para a tabela associativa porém se comportam apenas como chave estrangeira



# Conceitual X Lógico

**Relacionamento Um para Um:** Normalmente esse é um caso onde as duas entidades se fundem em uma só, ou a chave primária acaba sendo a mesma nas duas tabelas. Isso depende do caso. Por exemplo no caso Produto X Estoque

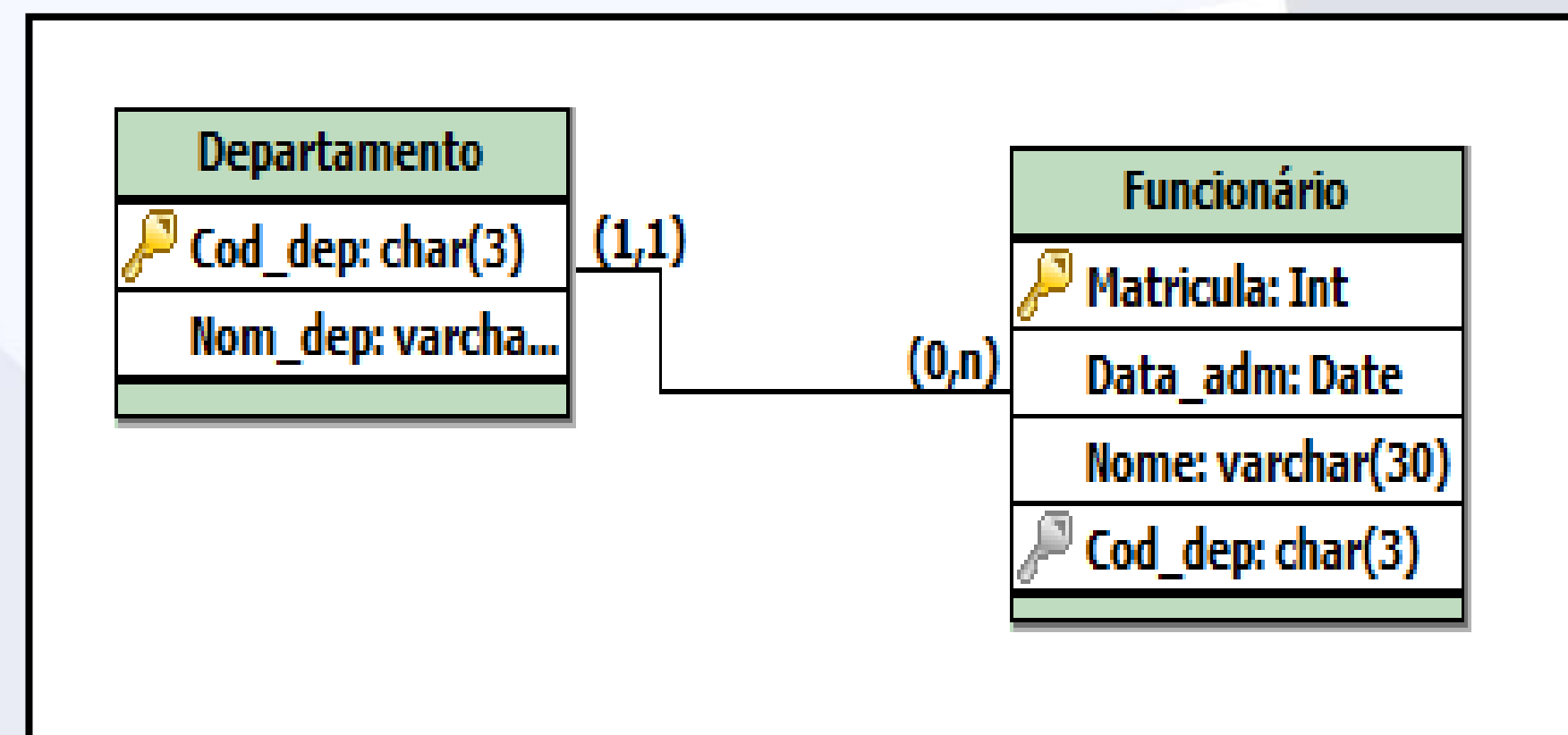
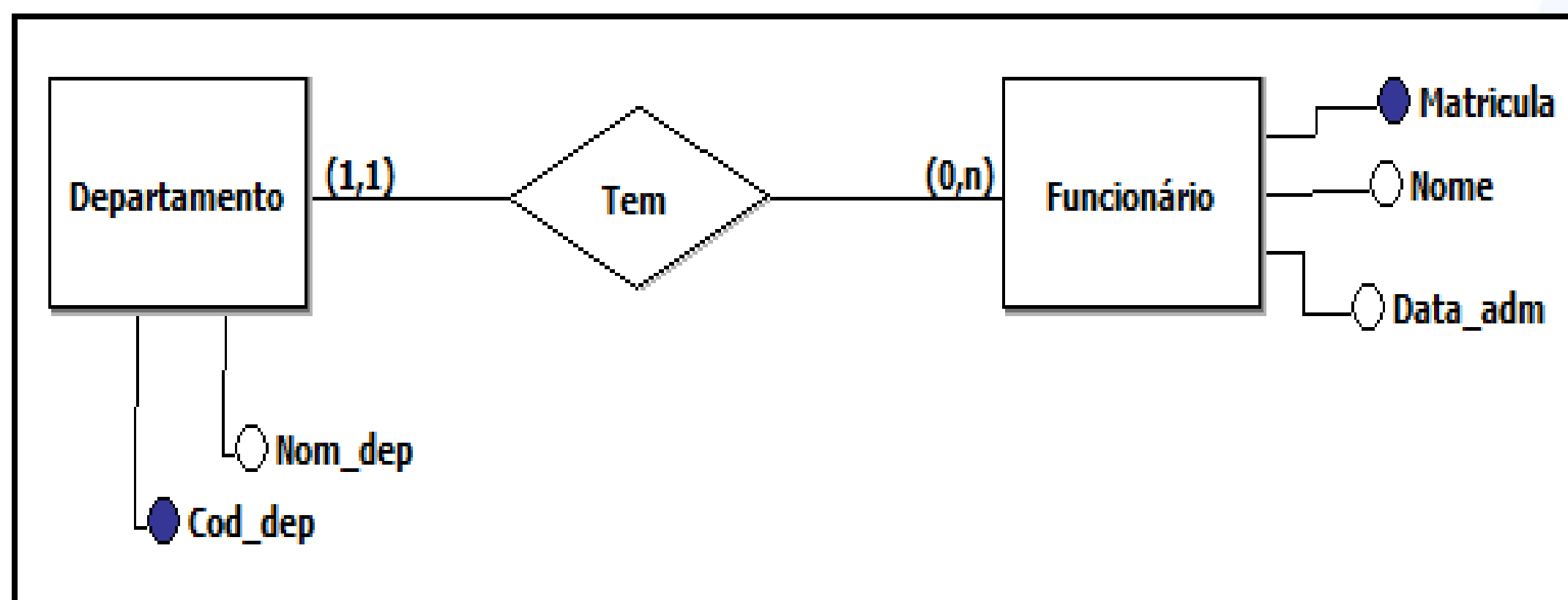


Produto+Estoque	
	<b>Cod_pro: int</b>
	<b>Nome: varchar(30)</b>
	<b>Preco: real(15,2)</b>
	<b>Qnt_est: real(15,2)</b>
	<b>UNI_vend: char(3)</b>

# Conceitual X Lógico



**Um para Muitos:** Transformar em duas entidades, onde estiver o N, lá estará a chave estrangeira.



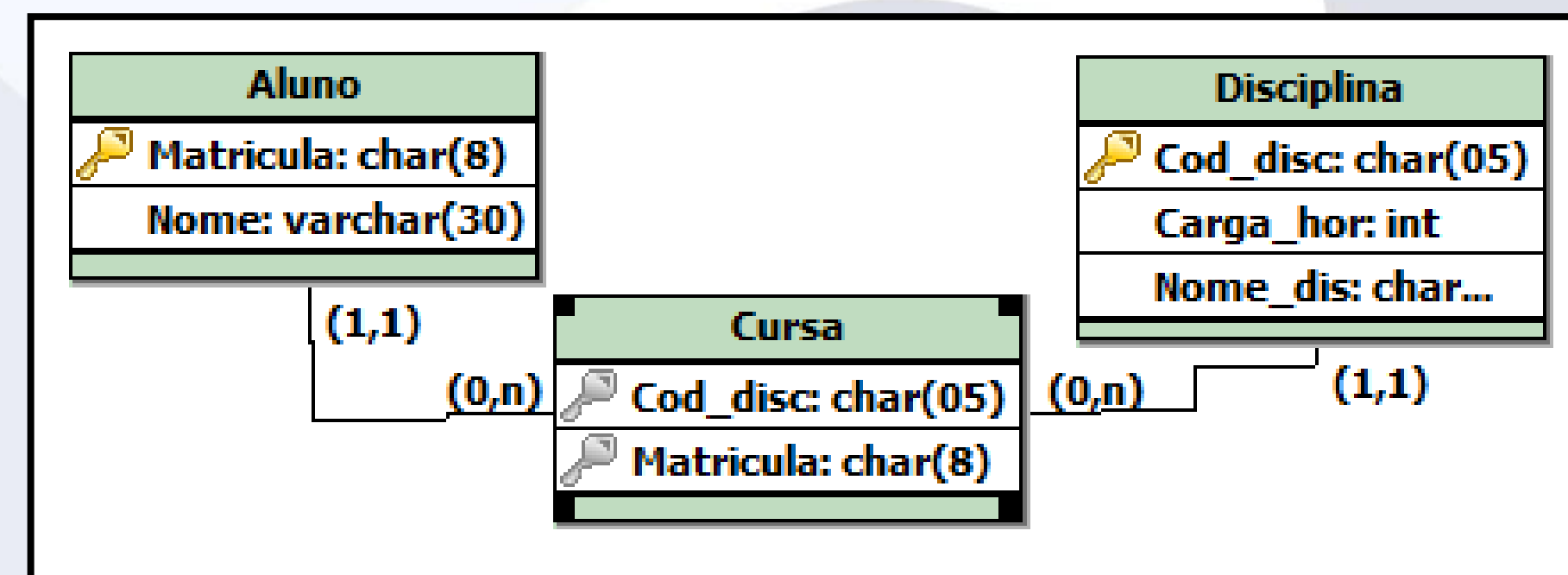
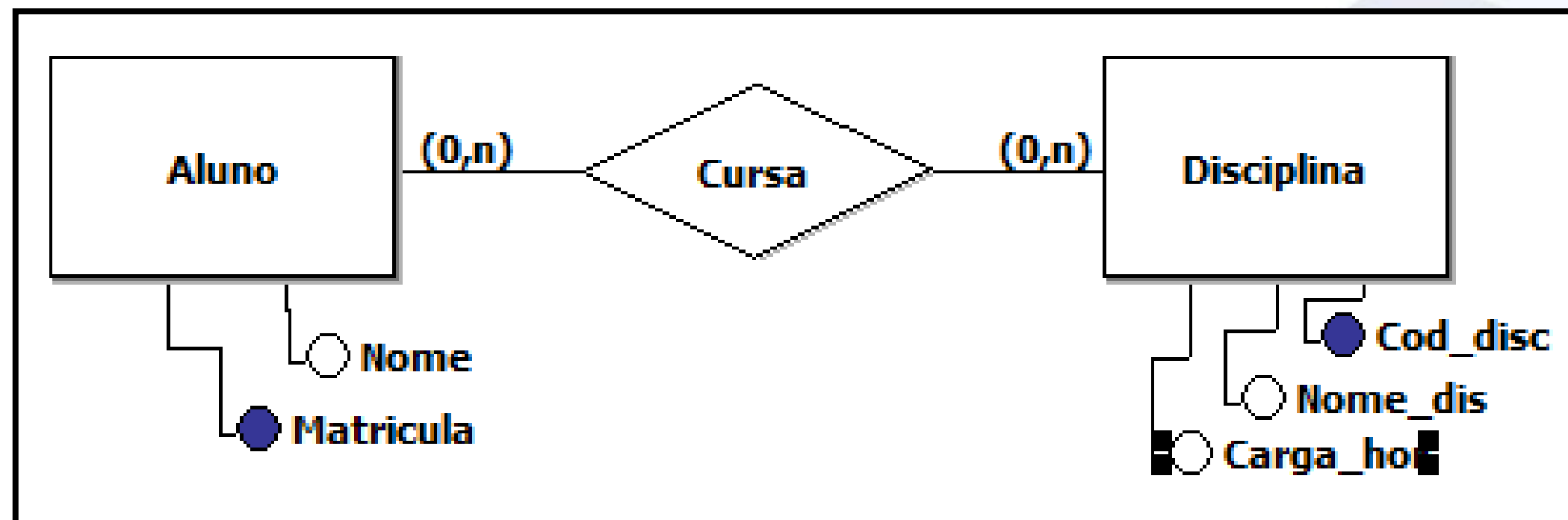
**Caso haja dependência de existência entre as entidades, ou seja, uma entidade só possa existir se a outra existir, a chave primária da entidade do lado “um” passa a ser chave primária (e estrangeira) da entidade do lado “N”, juntamente com a chave primária dessa entidade. Ou seja, uma chave primária composta, na entidade do lado “N”.**



# Conceitual X Lógico



**Muitos para Muitos:** Será necessário criar uma outra tabela que terá uma chave primária composta, formada pelas chaves primárias das outras tabelas envolvidas no relacionamento.

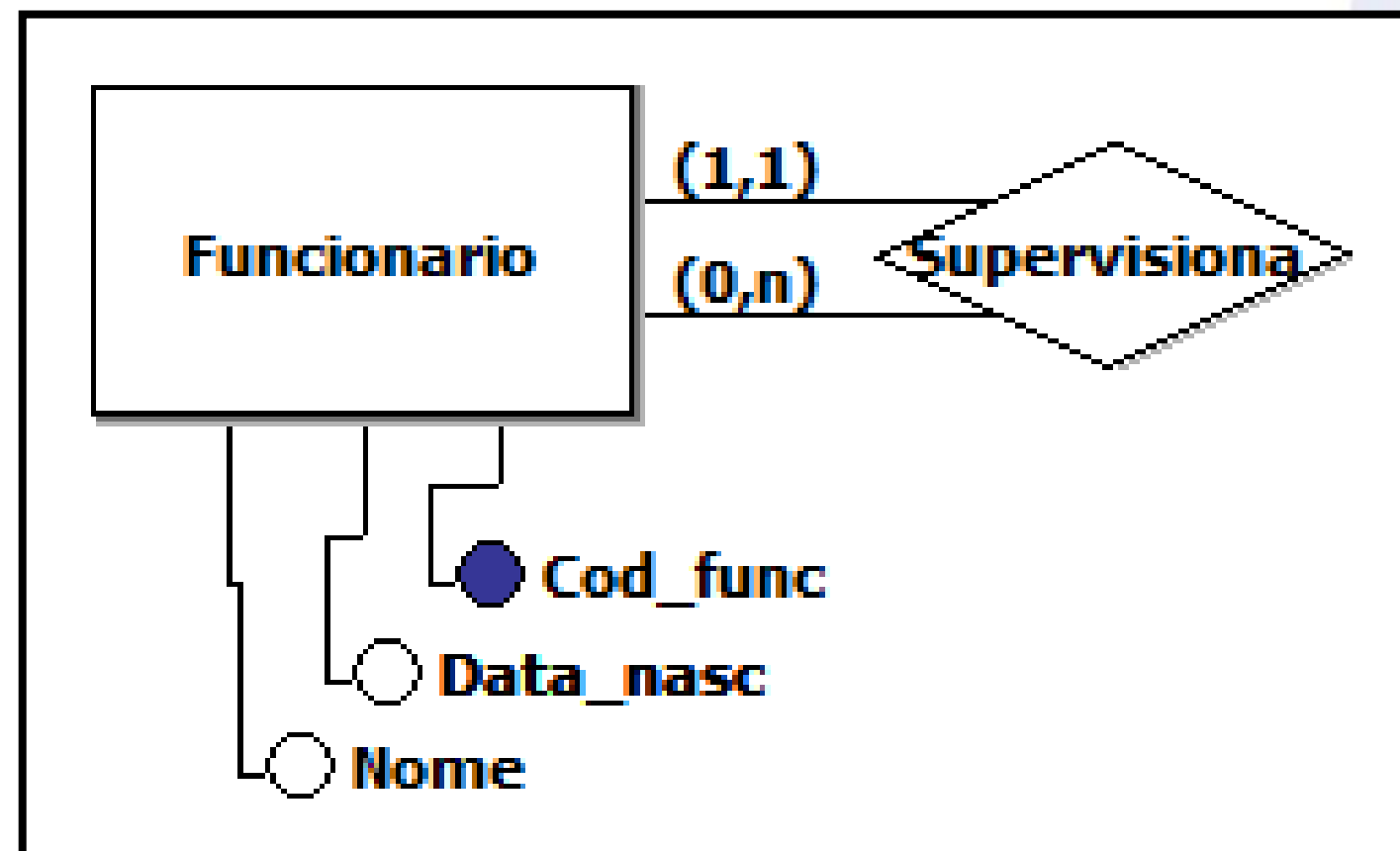



**Nessa tabela intermediária, você pode colocar atributos que identifiquem a relação entre as entidades.**

# Conceitual X Lógico



**Auto Relacionamento Um para Muitos:** O relacionamento será realizado através de um atributo existente na própria tabela, que representará a chave de uma outra linha da mesma entidade:

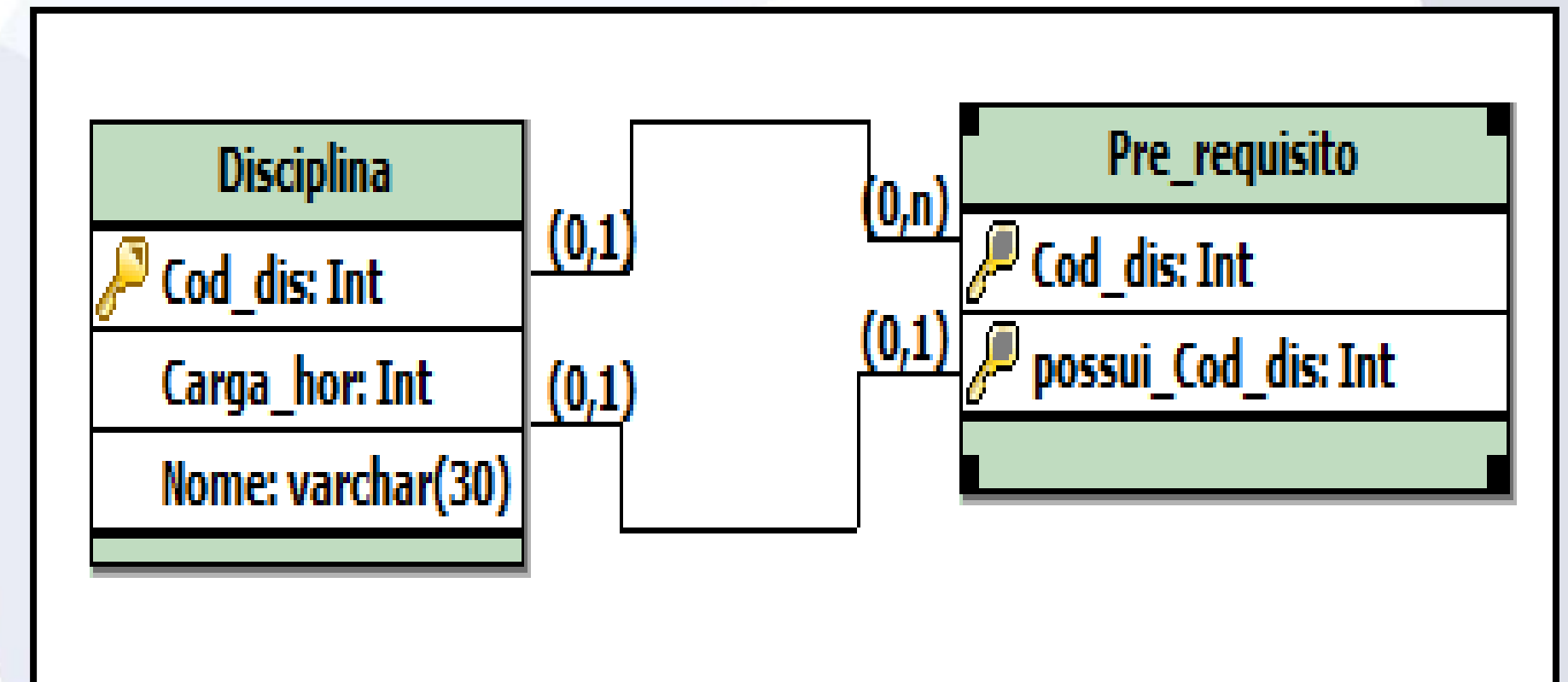
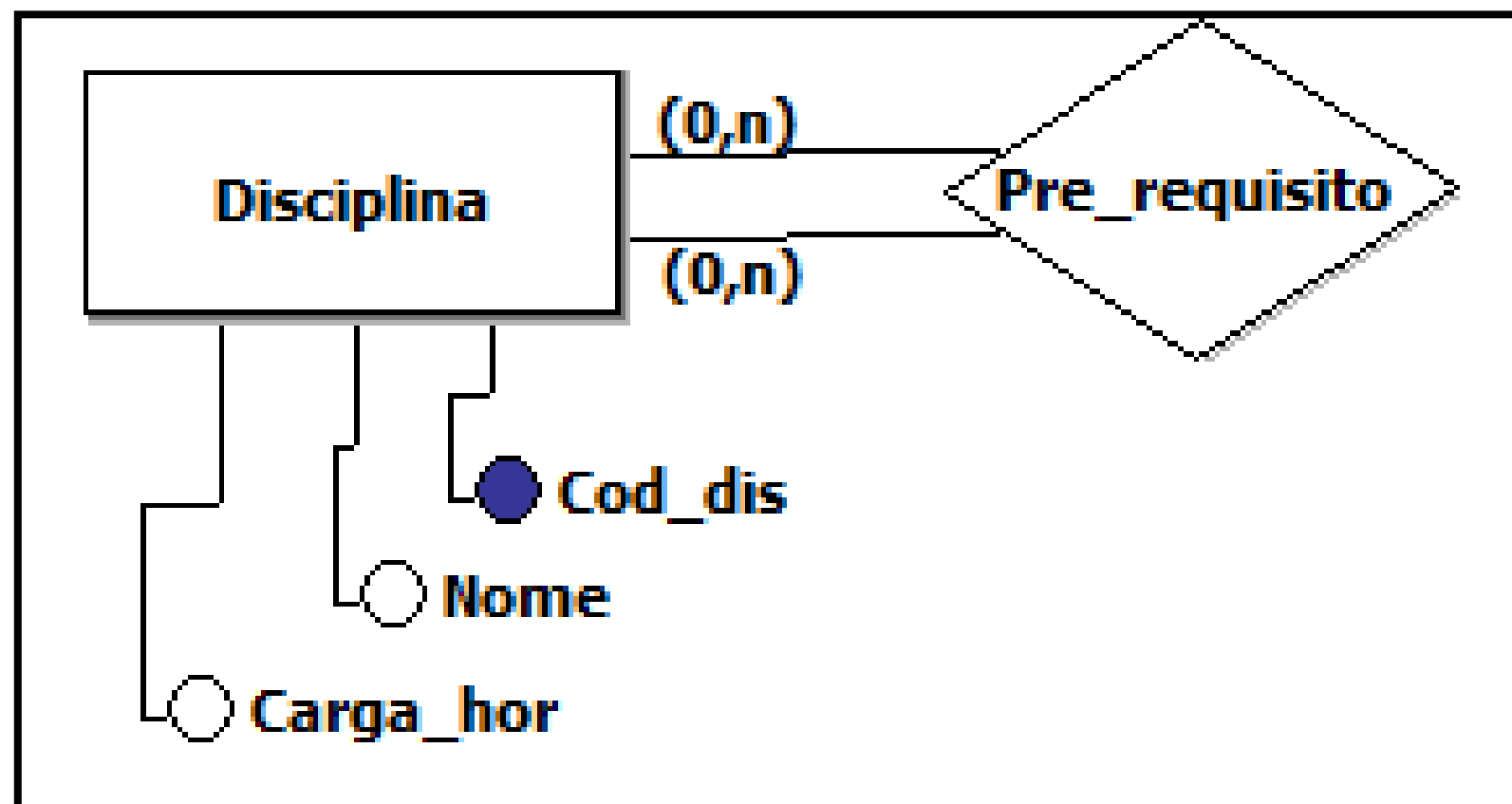


Funcionario	
	Cod_func: Int
	Nome: varchar(30)
	Data_nasc: date
	possui_Cod_func: Int

# Conceitual X Lógico



**Auto Relacionamento Muitos para Muitos:** O relacionamento será realizado através da criação de uma outra tabela que terá uma chave primária composta por mais de um campo chave da tabela, a diferença estará no papel realizado por cada uma delas:



**Nesse caso, criamos uma tabela que armazenará os relacionamentos entre as disciplinas. E essa tabela terá como chave primária composta a chave primária da outra tabela (duas vezes).**





# Normalização

- Normalização é o processo de modelar o banco de dados projetando a forma como as informações serão armazenadas a fim de eliminar, ou pelo menos **minimizar, a redundância no banco**. Tal procedimento é feito a partir da identificação de uma anomalia em uma relação, decompondo-as em relações melhor estruturadas.
- Normalmente precisamos remover uma ou mais colunas da tabela, dependendo da anomalia identificada e criar uma segunda tabela, obviamente com suas próprias chaves primárias e relacionarmos a primeira com a segunda para assim tentarmos evitar a redundância de informações.

1FN

**Elimina os grupos repetitivos e atributos multivalorados.**

2FN

**Elimina a dependência parcial.**

3FN

**Eliminar a dependência Transitiva**

# Normalização



- Redundância é a causa de vários problemas com esquemas relacionais:
  - Causando armazenamento redundante, anomalias de inserção, de exclusão e de atualização.
- Restrições de integridade podem ser usadas para identificar esquemas com esses problemas e para sugerir refinamentos.
- Principal técnica de refinamento é a decomposição de um esquema em subesquemas.
- A decomposição deve ser usada cuidadosamente:
  - Há motivos para se decompor uma relação?
  - A decomposição pode causar problemas?

1FN

**Elimina os grupos repetitivos e atributos multivalorados.**

2FN

**Elimina a dependência parcial.**

3FN

**Eliminar a dependência Transitiva**

# Normalização



Considere o esquema:

Pacientes(Id, Nome, Endereço, Telefone, Sexo, Data\_nascimento, **Sigla\_convênio, Nome\_convênio, Endereço\_convênio, Telefone\_convênio**)

- Anomalia de Inserção:
- Anomalia de Exclusão:
- Anomalia de Modificação:

1FN

**Elimina os grupos repetitivos e atributos multivalorados.**

2FN

**Elimina a dependência parcial.**

3FN

**Eliminar a dependência Transitiva**



# Dependência funcional

---



- Dependências funcionais (DFs) são restrições de integridade mais gerais que as restrições de chave.
- Leia-se: Sigla\_convênio determina funcionalmente Nome\_convênio, Endereço\_convênio e Telefone\_convênio.

{Sigla\_convênio} →  
{Nome\_convênio,  
Endereço\_convênio,  
Telefone\_convênio}

# Dependência funcional



Identificando as dependências funcionais:

<u>Num_NF</u>	<u>Cod_Produto</u>	Descricao	Preco	Data	Quantidade
001	PC	Computador	1.500,00	15/04/1998	02
001	IMP	Impressora	500,00	15/04/1998	01
002	PC	Computador	1.500,00	16/04/1998	01
003	PC	Computador	1.500,00	14/04/1998	03

## Dependências funcionais

Num\_NF → Data

Cod\_Produto → Descricao, Preco

Num\_NF, Cod\_Produto → Quantidade

# Dependência Funcional Parcial



Identificando as dependências funcionais:

<u>Num_NF</u>	<u>Cod_Produto</u>	Descricao	Preco	Data	Quantidade
001	PC	Computador	1.500,00	15/04/1998	02
001	IMP	Impressora	500,00	15/04/1998	01
002	PC	Computador	1.500,00	16/04/1998	01
003	PC	Computador	1.500,00	14/04/1998	03

## Dependências funcionais

Num\_NF → Data

Cod\_Produto → Descricao,Preco

Num\_NF,Cod\_Produto → Quantidade



# Dependência funcional Transitiva



- Na definição dos campos de uma entidade podem ocorrer casos em que um campo não seja dependente diretamente da chave primária ou de parte dela, mas sim dependente de outro campo da tabela, campo este que não é a Chave Primária.
- Quando isto ocorre, dizemos que a tabela possui dependência funcional transitiva. É importante deixar claro a diferença entre dependência funcional parcial e a transitiva. Na parcial, pelo menos um atributo da tabela depende de parte da chave primária (e não dela toda); na transitiva, pelo menos um atributo da tabela depende de outro atributo que **não seja** chave primária.

Pacientes(  
Id,  
Nome, Endereço,  
Telefone,  
Sexo,  
Data\_nascimento,  
Sigla\_convênio,  
**Nome\_convênio,**  
**Endereço\_convênio,**  
**Telefone\_convênio)**

# Dependência funcional Transitiva



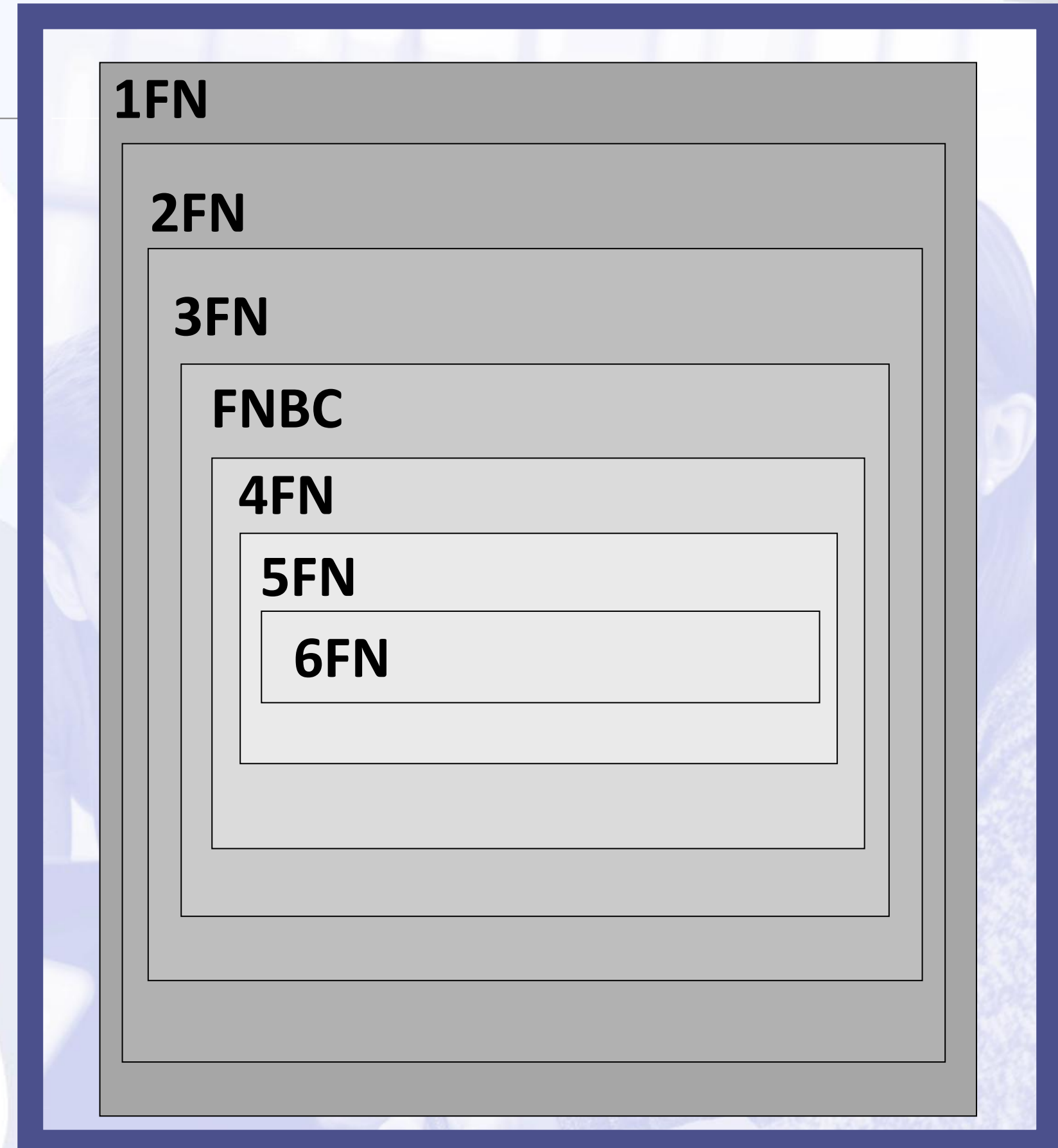
- Na definição dos campos de uma entidade podem ocorrer casos em que um campo não seja dependente diretamente da chave primária ou de parte dela, mas sim dependente de outro campo da tabela, campo este que não é a Chave Primária.
- Quando isto ocorre, dizemos que a tabela possui dependência funcional transitiva. É importante deixar claro a diferença entre dependência funcional parcial e a transitiva. Na parcial, pelo menos um atributo da tabela depende de parte da chave primária (e não dela toda); na transitiva, pelo menos um atributo da tabela depende de outro atributo que **não seja** chave primária.

Pacientes(  
Id,  
Nome, Endereço,  
Telefone,  
Sexo,  
Data\_nascimento,  
Sigla\_convênio,  
**Nome\_convênio,**  
**Endereço\_convênio,**  
**Telefone\_convênio)**



# Normalização

- Processo iterativo
- processo decomposto em vários níveis de normalização, chamados de primeira forma normal, segunda forma normal e assim por diante
- Cada forma normal é mais restritiva que a anterior
- A forma normal de ordem  **$N + 1$**  está na forma normal de ordem  $N$  e não possui uma característica “não desejável” que a forma normal de ordem  $N$  possui
- A forma normal de ordem  **$N + 1$**  é mais desejável do que a forma normal de ordem  $N$
- **O processo de normalização é um processo de redução da redundância.**

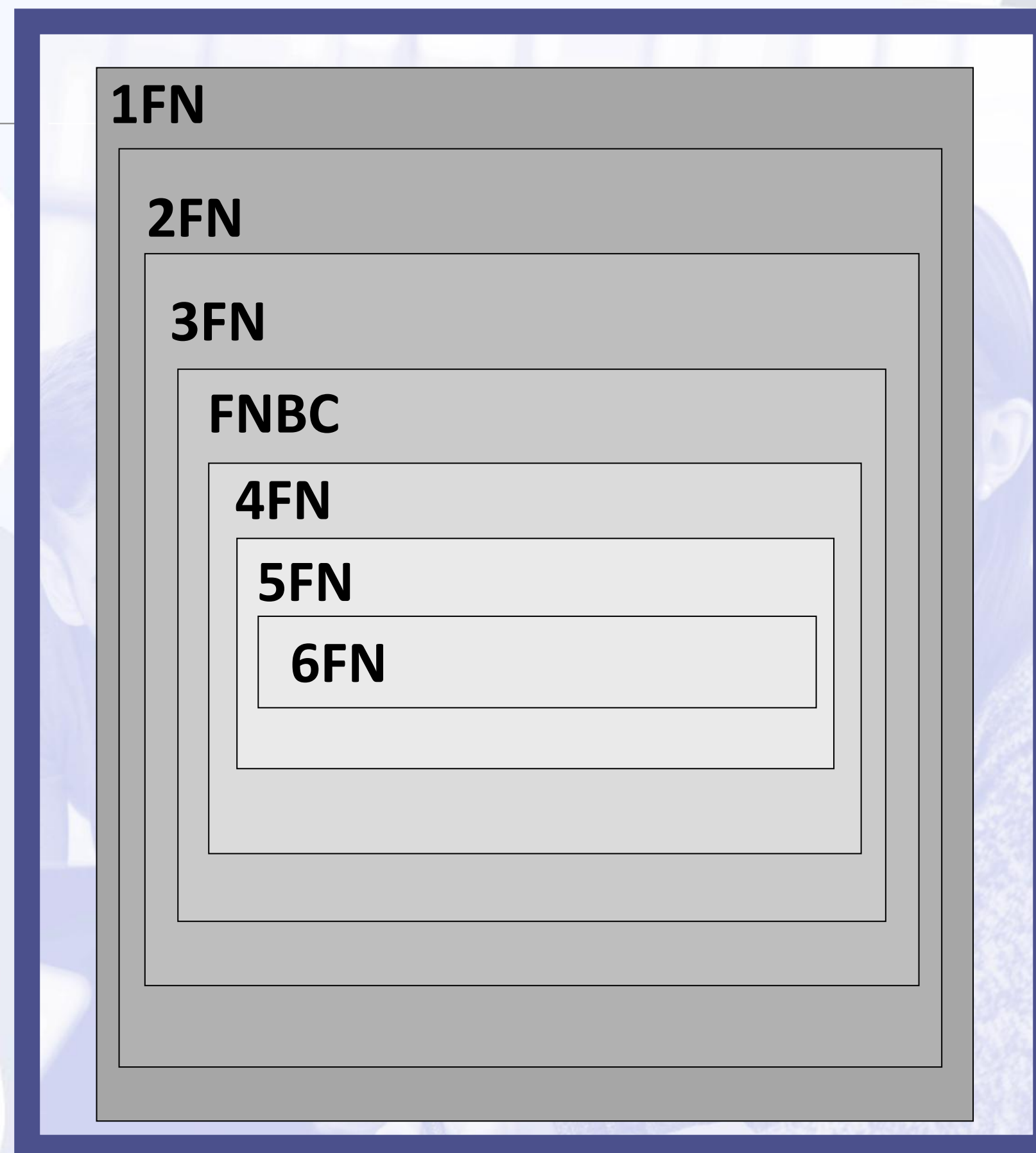




# Formas não Normais



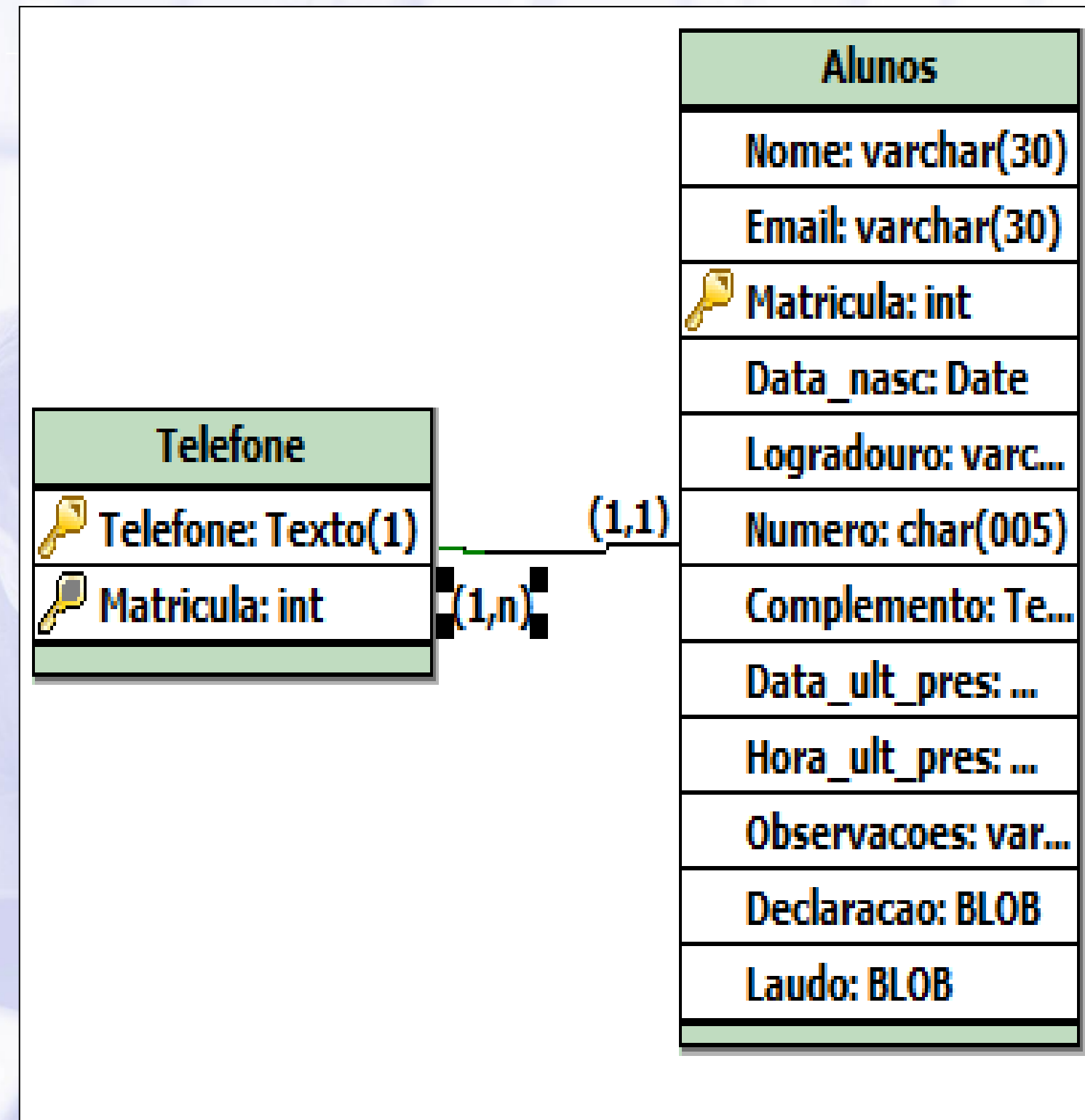
- Uma relação ou tabela está em uma Forma Não Normal, ou ainda, uma relação não está em Forma Normal quando algum cruzamento de linha e coluna contiver qualquer atributo agregado, decomponível, não atômico



# Primeira forma Normal



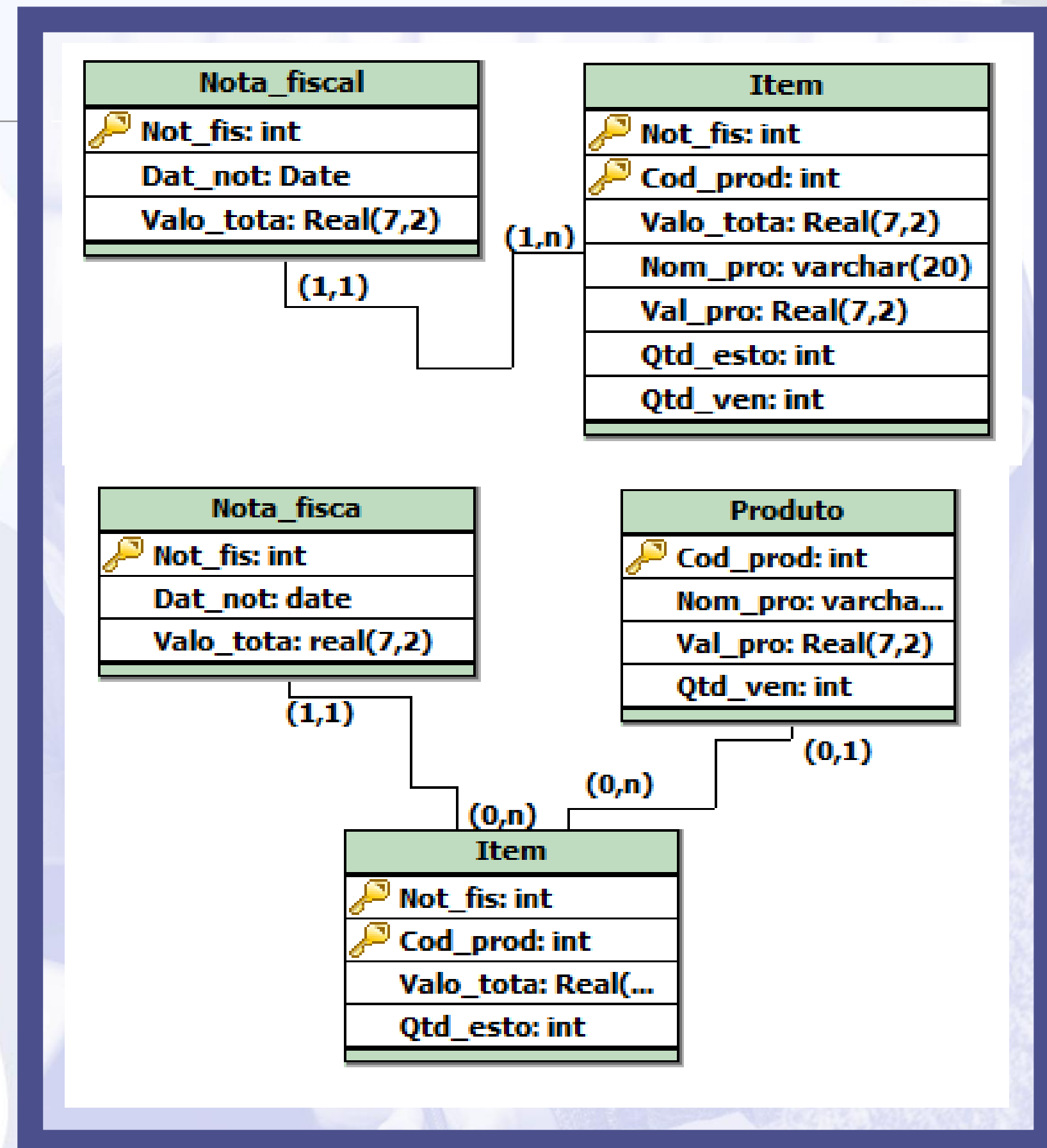
- Diz-se que uma relação está na Primeira Forma Normal quando:
  - Não contém atributos multivalorados;
  - Não contém grupos repetitivos;
- O objetivo da primeira forma normal é eliminar a repetição de grupos de informação;
- Regra
  - Não devem existir colunas multivaloradas



# Segunda forma Normal



- Diz-se que uma relação está na Segunda Forma Normal se:
  - Está na Primeira Forma Normal;
  - Todos os atributos não-chave dependem funcionalmente da totalidade da chave;
- Na 2FN são eliminadas todas as dependências parciais de atributos não-chave para com a chave primária, ela só se aplica para chaves compostas
- Uma relação está na Segunda Forma Normal (2FN) se estiver na 1FN e cada atributo não chave for totalmente dependente da chave primária, sem que ocorram dependências parciais








# Terceira forma Normal




- Diz-se que uma relação está na Terceira Forma Normal (3FN) se:
  - Está na Segunda Forma Normal (2FN);
  - Todos os atributos não-chave independem funcionalmente uns dos outros;
- Uma relação está na Terceira Forma Normal (3FN) se estiver na 2FN e todo atributo não chave não for dependente transitivo da chave primária

Paciente	
 Id_pac: int	
Nome: varchar(40)	
Endereco: varchar(60)	
Telefone: char(11)	
Sexo: Número(4)	
Dat_Nasc: Date	
Sigla_conv: int	
Nom_conv: varchar(40)	
Ende_conv: varchar(60)	
Tele_conv: char(11)	

Paciente	
 Id_pac: int	
Nome: varchar(40)	
Endereco: varchar(60)	
Telefone: char(11)	
Sexo: Número(4)	
Dat_Nasc: Date	
 Sigla_conv: int	

(1,n)

(1,1)


Convenio	
 Sigla_conv: int	
Nom_conv: varchar(40)	
Ende_conv: varchar...	
Tele_conv: char(11)	



# Formas normais depois da 3FN



As formas normais após a 3FN tratam relações que envolve chaves candidatas e relações com mais de uma coluna como chave primária. Isso acontece quando:


- tenha chaves candidatas múltiplas, onde
- estas chaves candidatas fossem compostas, e
- as chaves candidatas se sobrepunham (isto é, tinham pelo menos um atributo em comum)

Paciente	
 Id_pac: int	
Nome: varchar(40)	
Endereco: varchar(60)	
Telefone: char(11)	
Sexo: Número(4)	
Dat_Nasc: Date	
Sigla_conv: int	
Nom_conv: varchar(40)	
Ende_conv: varchar(60)	
Tele_conv: char(11)	

Paciente	
 Id_pac: int	
Nome: varchar(40)	
Endereco: varchar(60)	
Telefone: char(11)	
Sexo: Número(4)	
Dat_Nasc: Date	
 Sigla_conv: int	

(1,n)

(1,1)

Convenio	
 Sigla_conv: int	
Nom_conv: varchar(40)	
Ende_conv: varchar...	
Tele_conv: char(11)	

# Formas normais depois da 3FN



- Uma relação R está na BCNF se e somente se R está na 3FN e todo determinante de R for uma chave candidata
- Vejamos o seguinte exemplo em que se assumem as seguintes dependências funcionais;

Encarregado → Cidade,  
Departamento

Cada encarregado trabalha em uma única cidade e gerencia um único departamento.

Cidade, Departamento  
→ Encarregado

Só existe um encarregado para cada Cidade e Departamento.



# Formas normais depois da 3FN



- Uma relação R está na BCNF se e somente se R está na 3FN e todo determinante de R for uma chave candidata
- Vejamos o seguinte exemplo em que se assumem as seguintes dependências funcionais;

Encarregado → Cidade,  
Departamento

Cada encarregado trabalha em uma única cidade e gerencia um único departamento.

Cidade, Departamento  
→ Encarregado

Só existe um encarregado para cada Cidade e Departamento.

# Formas normais depois da 3FN



- Esta forma normal foi especificada para lidar com situações em que:
  - Existem múltiplas chaves candidatas;
  - As chaves candidatas são compostas;
  - As chaves candidatas se sobrepõem;
- Se alguma das situações anteriores não se verificar, então a forma normal de Boyce-Codd reduz-se à 3FN;

Encarregado → Cidade, Departamento

Cada encarregado trabalha em uma única cidade e gerencia um único departamento.

Cidade, Departamento → Encarregado

Só existe um encarregado para cada Cidade e Departamento.

# Formas normais depois da 3FN



- Entidade original

Encarregado	Projeto	Cidade	Departamento
Belmiro	Martex	Coimbra	D1
Gaspar	Jacinto	Beja	D1
Gaspar	Martex	Beja	D1
Henrique	Soundex	Beja	D2
Henrique	Vinius	Beja	D2

- Esta situação pode ser resolvida com a criação de outra entidade, resultando no seguinte esquema sem duplicação de valores.



# Formas normais depois da 3FN



- Entidade com os encarregados

Encarregado	Cidade	Departamento
Belmiro	Coimbra	D1
Gaspar	Beja	D1
Henrique	Beja	D2

- Entidade com os Projetos

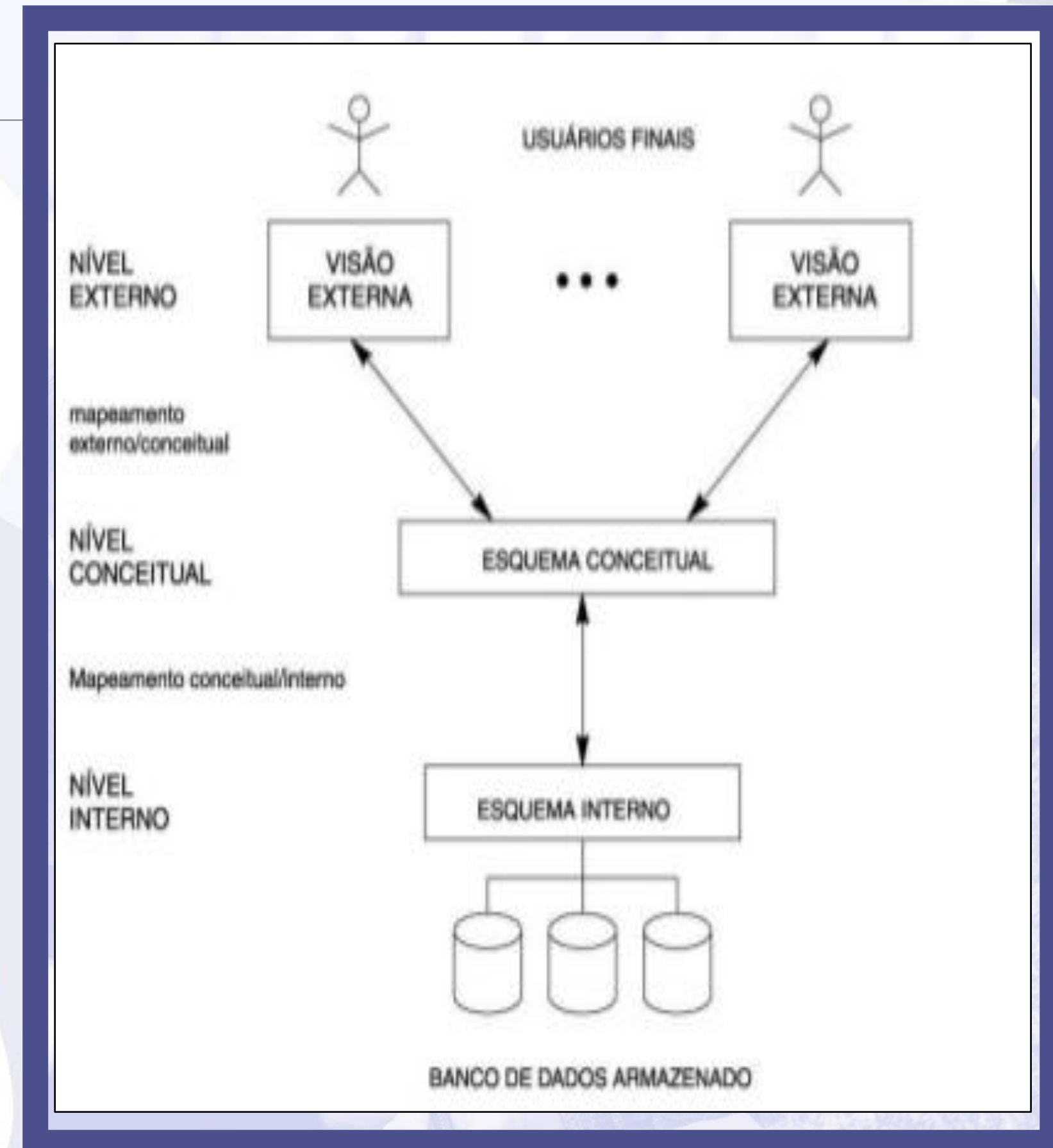
Projeto	Cidade	Departamento
Martex	Coimbra	D1
Jacinto	Beja	D1
Martex	Beja	D1
Soundex	Beja	D2
Vinius	Beja	D2

- Para que a Forma Normal de Boyce-Codd seja aplicável, é necessário que existam pelo menos duas chaves candidatas com atributos comuns que se sobreponham.

# Arquitetura de Três Esquemas

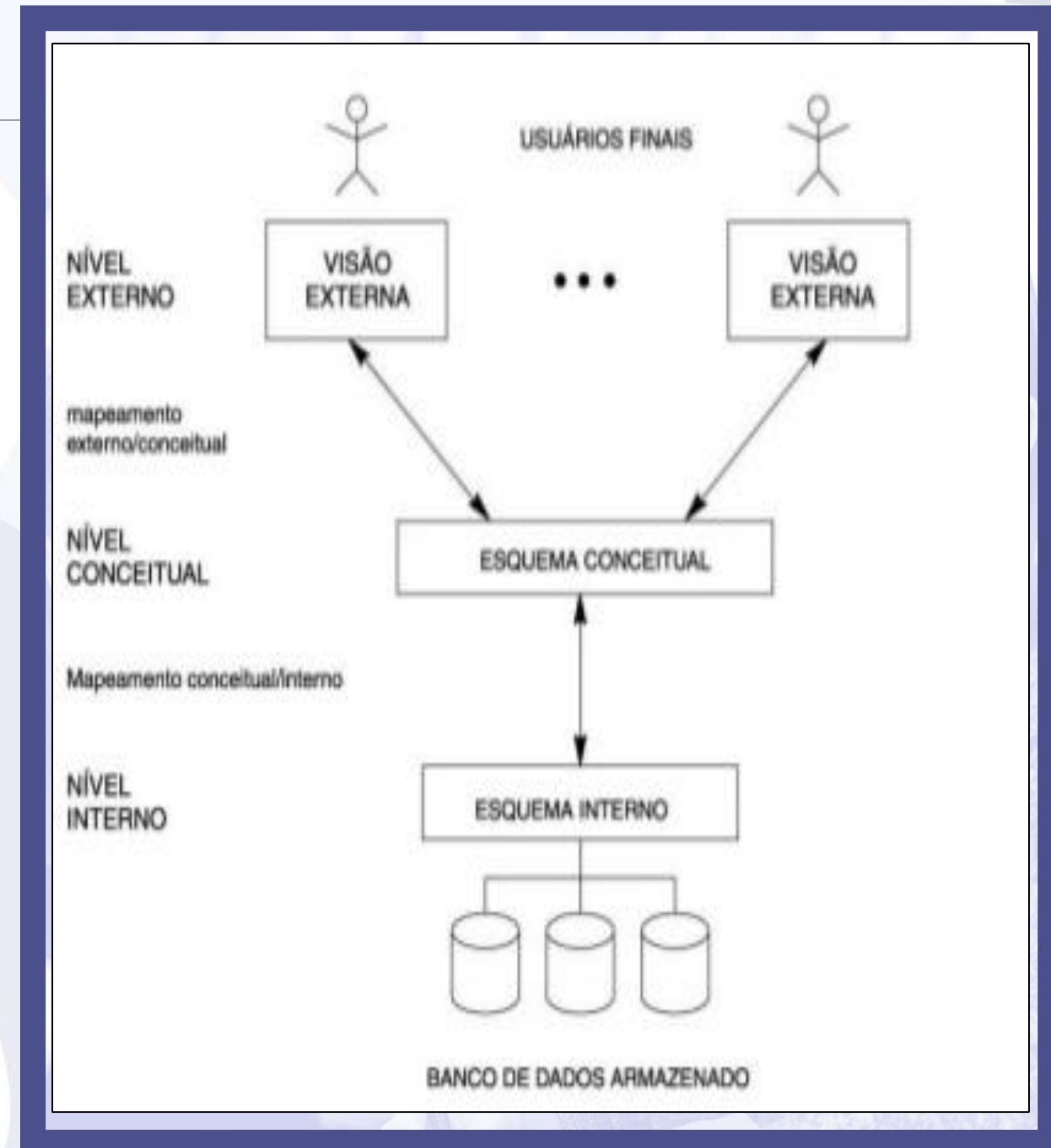


- Esquemas podem ser definidos em Três-níveis
  - Proposta para auxiliar na realização e visualização das seguintes características:
  - Independência de dados e operação de programas
  - Suporte a múltiplas visões
  - Uso do catálogo para armazenar a descrição do banco de dados



# Arquitetura de Três Esquemas

- Esquemas podem ser definidos em Três níveis
  - O objetivo é separar o usuário da aplicação do banco de dados físico
- ## 1. Nível Interno – esquema interno
- Descreve a estrutura de armazenamento físico do banco de dados
  - Utiliza um modelo de dados físico





# Arquitetura de Três Esquemas

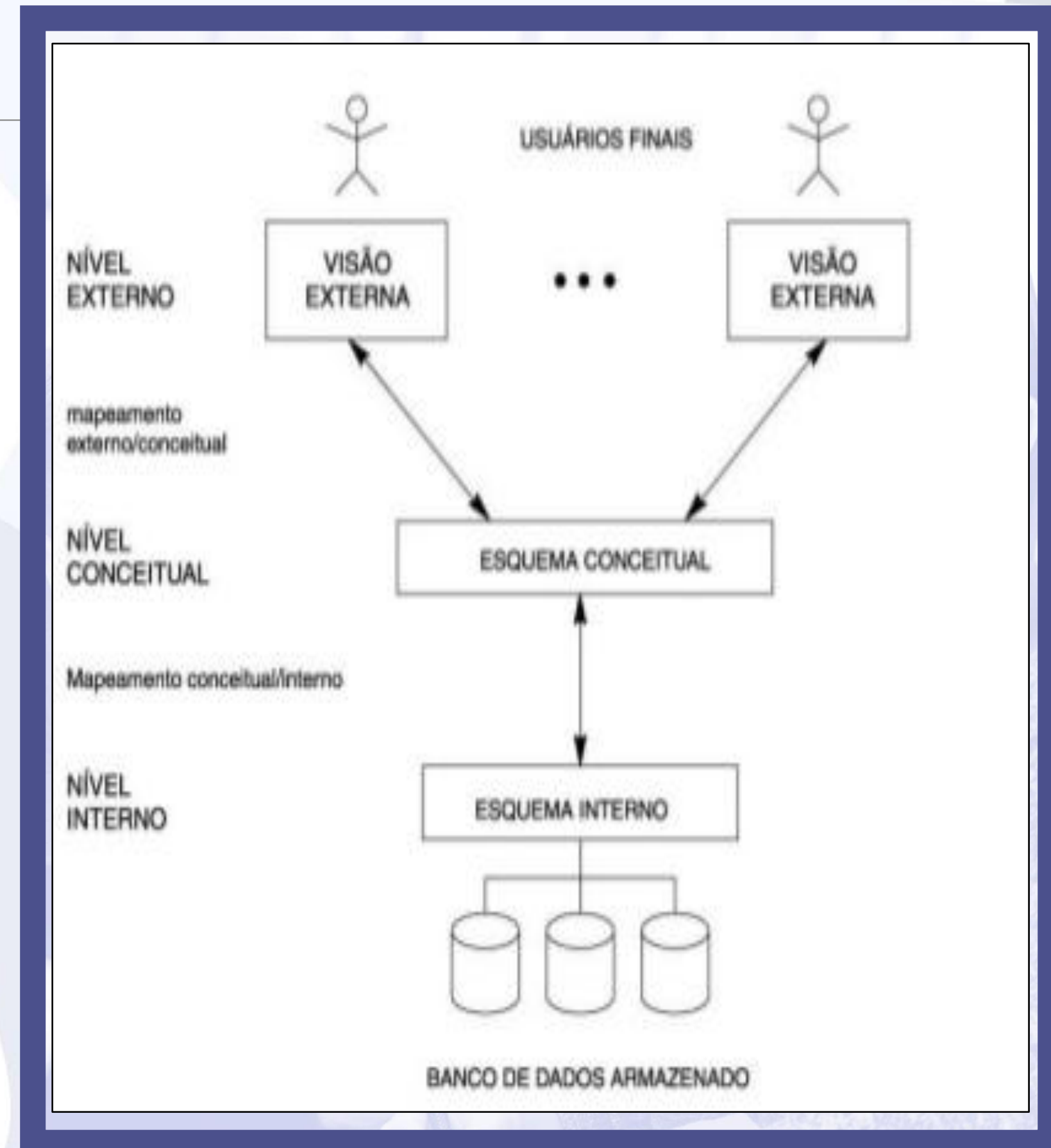


## 2. **Nível Conceitual** – esquema conceitual ou lógico

- Descreve a estrutura da base de dados sem detalhes de estrutura de armazenamento físico
- Que dados estão armazenados e como estão relacionados

## 3. **Nível Externo** – esquema externo (visões dos usuários)

- Descreve as visões dos usuários: a parte da base de dados em que cada grupo de usuários tem interesse
- Descrição de subesquemas



# Arquitetura de Três Esquemas

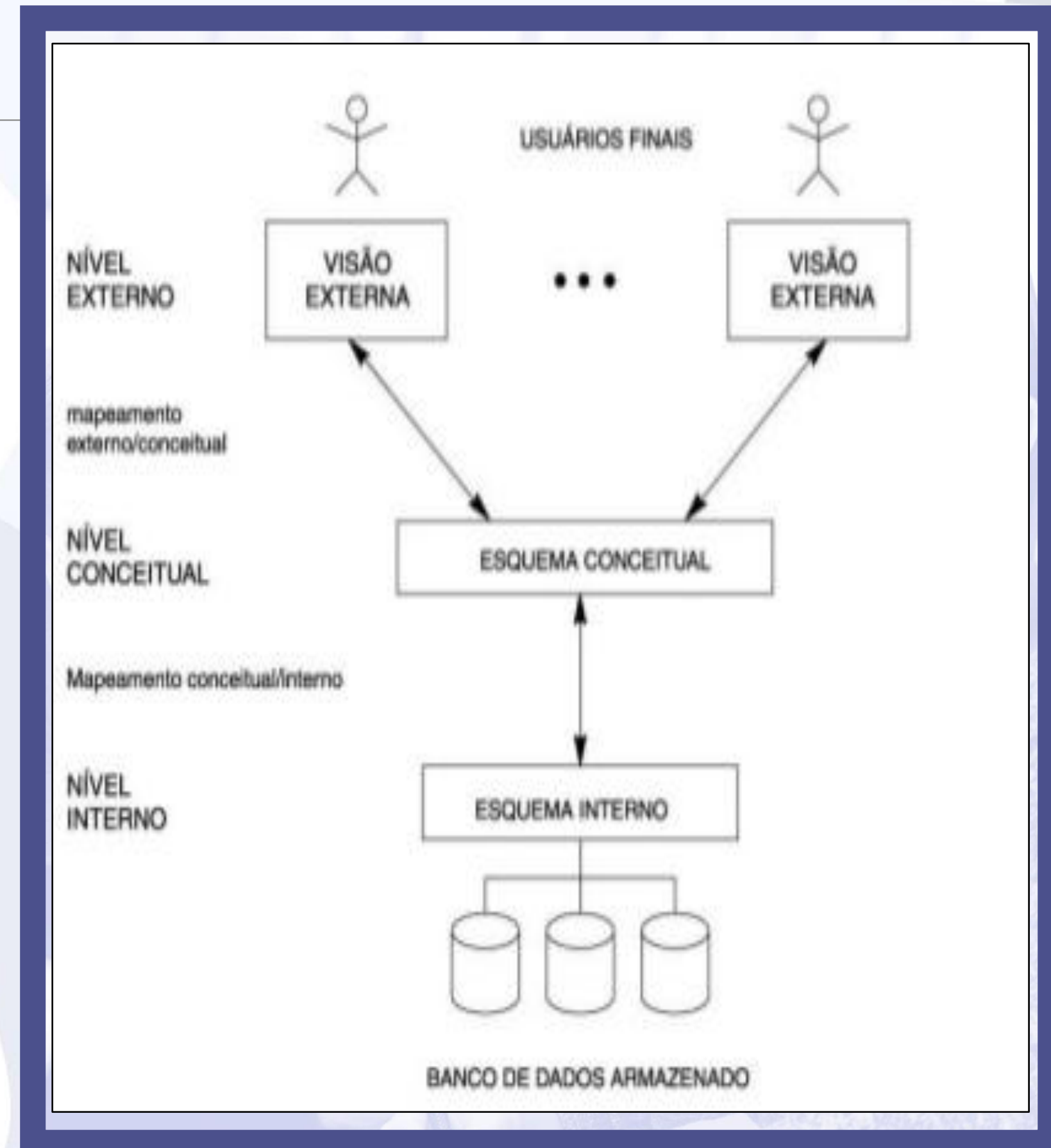


## 2. **Nível Conceitual** – esquema conceitual ou lógico

- Descreve a estrutura da base de dados sem detalhes de estrutura de armazenamento físico
- Que dados estão armazenados e como estão relacionados

## 3. **Nível Externo** – esquema externo (visões dos usuários)

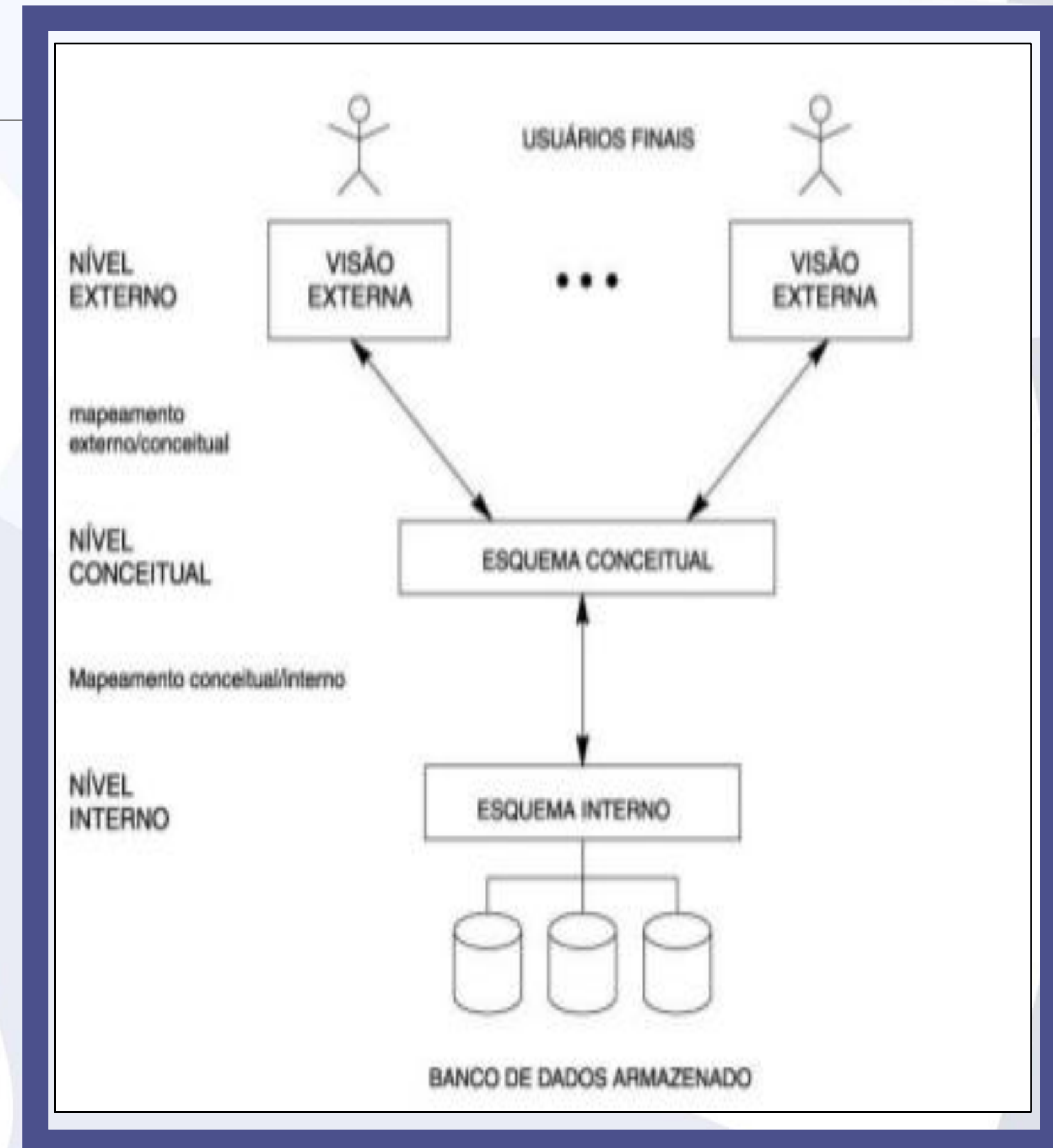
- Descreve as visões dos usuários: a parte da base de dados em que cada grupo de usuários tem interesse
- Descrição de subesquemas



# Independência de Dados



- É a capacidade de mudar o esquema em um nível do sistema de banco de dados sem que ocorram alterações do esquema no próximo nível mais alto:
- **Independência de dados lógica**
  - Refere-se a capacidade de modificar o esquema lógico sem que, com isso, qualquer programa de aplicação precise ser reescrito

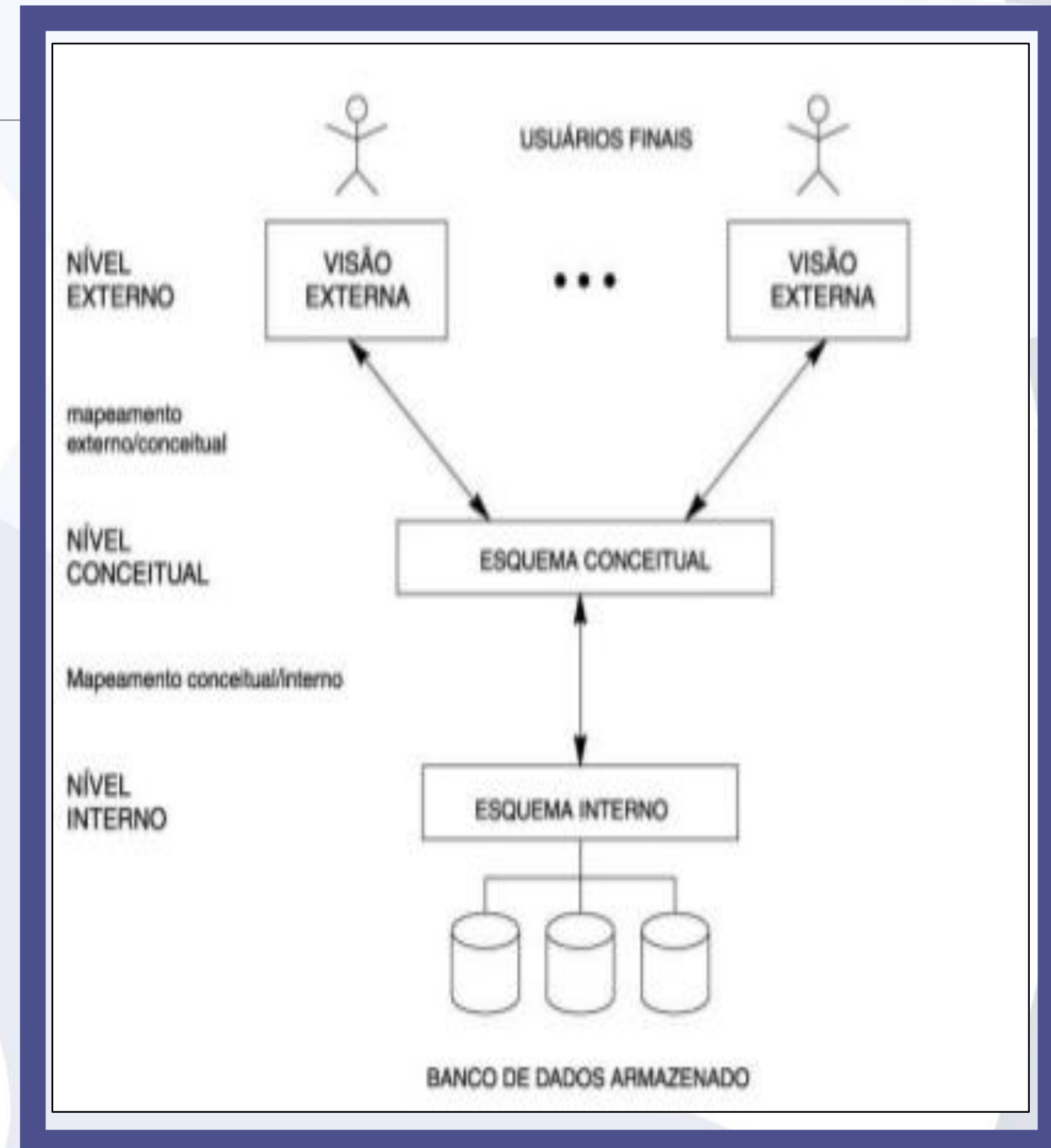




# Independência de Dados



- **Independência de dados física**
  - Refere-se a capacidade de modificar o esquema físico sem que, com isso, qualquer programa de aplicação precisa ser reescrito
  - O conceito de independência de dados é de várias formas similar ao conceito de tipo abstrato de dados empregado nas linguagens de programação



# Independência de Dados

EXTERNO (Excel)	EXTERNO (Java)
(A3) - Numero_do_empregado (B3) - Salário	String num_empr; String num_dpto;

CONCEITUAL	
NÚMERO_EMPREGADO	CHARACTER(5)
NÚMERO_DEPARTAMENTO	CHARACTER(4)
SALÁRIO	DECIMAL(5)

INTERNO	
EMP ARMAZENADO	BYTES=20
PREFIXO	BYTES=6, OFFSET=0
EMP#	BYTES=6, OFFSET=6, INDEX=EMPX
DEPTO#	BYTES=4, OFFSET=12
PAGTO	BYTES=4, ALIGN=FULLWORD, OFFSET=16

# PRÓXIMOS PASSOS



**Conceitos de Linguagem SQL**

**Linguagem SQL –  
Comandos DDL**

**Restringindo e  
ordenando dados**

**Tecnologias emergentes**

**Agregando dados com  
funções de grupo**

**Segurança em Banco de  
dados**



# OBRIGADO



**Adilson da Silva**

**Mestre**

**[adilson.silva@sereducacional.com](mailto:adilson.silva@sereducacional.com)**



# OBRIGADO



**UNINABUCO UNAMA UNINASSAU UNIVERITAS UNG UNINORTE**