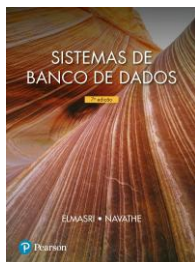


Banco de Dados

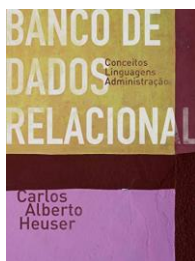
MODELO DE DADOS RELACIONAL

Bibliografia



ELMASRI, R.; SHAMKANT, B.N. *Sistemas de banco de dados*. 7ª edição. São Paulo: Pearson, 2018.

- *Capítulo 5: O modelo de dados relacional e as restrições em bancos de dados relacionais*



HEUSER, C.A. *Banco de dados relacional: conceitos, SQL e administração*. 1ª edição. Porto Alegre: O Autor, 2019.

- *Capítulo 2: Abordagem relacional*

Breve histórico

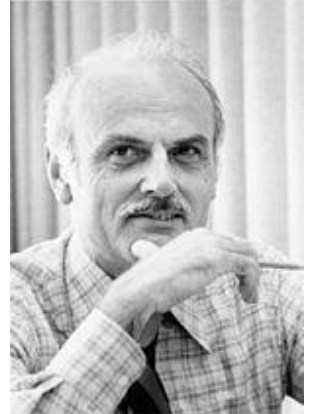
Primeiros SGBDs

- Os primeiros SGBDs apareceram ainda na década de 1960, uma época predominada por computadores de grande porte.
- Um exemplo destes SGBDs pioneiros é um software chamado **IMS**, da IBM.
 - Ele armazenava listas de árvores de registros, segundo uma abordagem denominada **hierárquica**.
 - Através de comandos de consulta, era possível navegar pela árvore, partindo da raiz em direção às folhas.
- Outro exemplo de SGBD típico daquela época é o **IDS**, da General Electric Company.
 - Desenvolvido por Charles W. Bachman a partir de 1961.
 - Seguiu a abordagem em rede, onde os registros são encadeados em um grafo, não limitado aos relacionamentos hierárquicos da abordagem hierárquica.
 - O IDS foi precursor de um padrão de banco de dados estabelecido pelo governo dos EUA (CODASYL/DBTG).
- Devido às limitações de hardware dos computadores da época, estes SGBDs dispunham de funcionalidades muito limitadas.
 - Basicamente, ofereciam ao programador a abstração de detalhes de estruturas de arquivos básicas, como listas encadeadas e índices sequenciais.

Breve histórico

O modelo relacional

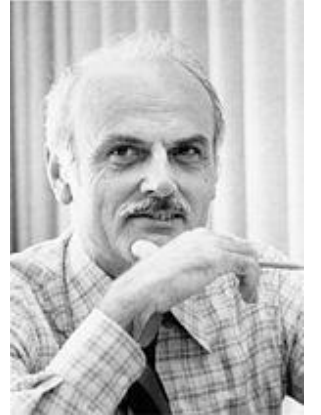
- Por volta de 1970, um pesquisador da IBM, chamado Edgar F. Codd, concebeu um novo tipo de SGBD, que se distinguia em vários aspectos daqueles que existiam na época.
- **Embasamento formal**
 - Os SGBDs existentes à época eram construídos de forma empírica, partindo de softwares destinados a resolver alguma aplicação específica. O resultado disto era que a adição de novas funcionalidades era difícil e o conjunto de conceitos pouco consistentes.
 - O modelo relacional teve embasamento formal, com a intenção de ser consistente e genérica. A sua inspiração vem da matemática, combinando ideias de teoria de conjuntos, álgebra e lógica de predicados.



Breve histórico

O modelo relacional

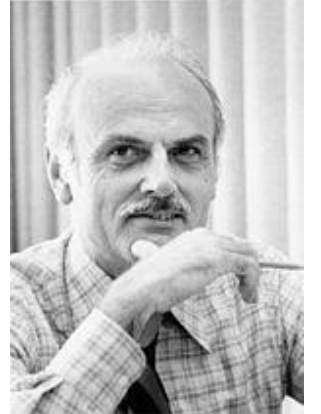
- Por volta de 1970, um pesquisador da IBM, chamado Edgar F. Codd, concebeu um novo tipo de SGBD, que se distinguia em vários aspectos daqueles que existiam na época.
- **Linguagens de consulta orientadas a conjunto**
 - Nos SGBDs primitivos, as linguagens de consulta eram **orientadas a registro**, isto é, as operações de consulta e modificação do conteúdo do banco de dados operavam sobre um único registro de cada vez.
 - O modelo relacional propõe o uso de uma linguagem **orientada a conjunto**.
 - Codd propôs duas linguagens diferentes:
 - **Álgebra relacional**: mais voltada para programadores.
 - **Cálculo relacional**: mais voltada para usuários não afeitos à programação.
 - Dentro de certas restrições, estas duas linguagens são equivalentes. Isto é, que qualquer consulta que pode ser expressa em uma das duas linguagens pode também ser expressa na outra linguagem.
 - Estas linguagens serviram de base para muitos conceitos da SQL, a linguagem usada hoje.



Breve histórico

O modelo relacional

- Por volta de 1970, um pesquisador da IBM, chamado Edgar F. Codd, concebeu um novo tipo de SGBD, que se distinguia em vários aspectos daqueles que existiam na época.
- **Regras para definir um banco de dados corretamente projetado**
 - Além de propor um modelo e duas linguagens de consulta, Codd propôs um conjunto de regras formais (chamadas de **normalização** de banco de dados), para formalizar o que intuitivamente entendemos por um banco de dados “bem projetado”.



Breve histórico

Protótipos de SGBD relacional

- No início da década de 1970, a construção de SGBDs era tema de pesquisa em várias instituições.
- Tendo conhecimento das ideias de Codd, alguns grupos de pesquisa atacaram o problema do desenvolvimento de um SGBD relacional.
 - Em San Jose na Califórnia, um grupo da própria IBM desenvolveu um protótipo de SGBD, chamado **System/R**.
 - O “R” vem de “relacional”.
 - Este sistema passou por várias versões até estar concluído no final da década de 70.
 - A linguagem SQL, como conhecemos hoje, é originária da linguagem de consulta usada no System/R.
 - Paralelamente, na Universidade da Califórnia em Berkeley, uma equipe formada ao redor de Michael Stonebraker desenvolveu o **INGRES**, um outro protótipo de SGBD relacional.
 - Este protótipo foi desenvolvido sobre um computador PDP 11, muito menor que o usado na IBM para implementar o System/R.
- Ambos os projetos demonstraram a viabilidade da implementação prática das ideias de Codd e suas equipes de desenvolvimento originaram grupos que se dedicaram ao desenvolvimento de produtos baseados na abordagem relacional.

Breve histórico

Difusão do SGBD relacional

- Da equipe de desenvolvimento do INGRES, saíram várias empresas, que criaram alguns SGBDs largamente usados, como Sybase, Informix e uma versão comercial do INGRES.
- O SQL Server da Microsoft foi construído a partir de uma versão do Sybase e, por isso, pode ser considerado um outro descendente do INGRES.
- A própria IBM implementou, a partir do System/R, um SGBD relacional, inicialmente chamado de SQL/DS e depois denominado DB2.
- Curiosamente, o primeiro SGBD relacional comercial baseado em SQL não foi desenvolvido por nenhuma das equipes dos projetos citados acima, e sim por dois consultores, Larry Ellison e Bob Miner.
 - Após terem tido contato com as ideias básicas do System/R, decidiram implementar seu próprio SGBD.
 - Eles denominaram seu produto de **Oracle** e lançaram a primeira versão em 1979.

Breve histórico

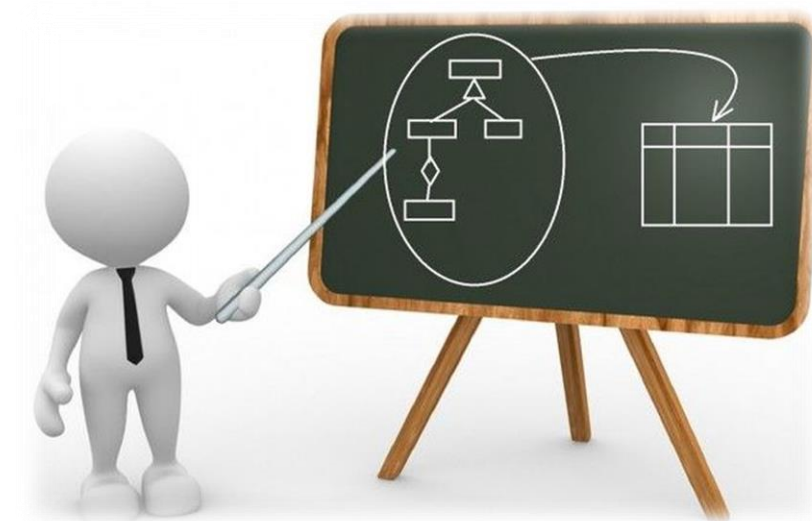
SQL Padrão

- A partir do início da década de 1980, com a disseminação de SGBDs relacionais, os órgãos de padronização reconheceram a necessidade de estabelecer um padrão para a área.
 - Em 1986, o ANSI (*American National Standards Institute*) aprovou a primeira versão do padrão SQL.
 - Logo depois, este padrão foi adotado pela ISO (*International Organization for Standardization*), tornando-se um padrão internacional.
- As primeiras versões do padrão foram as seguintes:
 - SQL1 ou SQL/86, liberado em 1986, com revisão em 1989.
 - SQL2 ou SQL/92, liberado em 1992.
 - SQL3 ou SQL/99, liberado em 1999.
 - SQL4 ou SQL/03, liberado em 2003.
- Desde então, o padrão vem sendo constantemente atualizado, tendo sido liberadas várias outras versões.
- A indústria adotou o padrão apenas lentamente. Infelizmente, cada desenvolvedor de SGBD optou por implementar variantes específicas.
 - Os SGBDs disponíveis implementam grande parte do SQL2, tendo alguns incorporado algumas características de SQL3 e SQL4.



Terminologia

- A terminologia usada para descrever os componentes de um banco de dados relacional evoluiu ao longo do tempo.
- Codd, o criador da abordagem relacional, foi buscar ideias na Matemática como base para a sua proposta do modelo de dados relacional. Especificamente, ele baseou-se em dois conceitos:
 - Tupla
 - Relação



Terminologia

Tupla

- Uma **tupla** é uma lista ordenada de n -valores.
 - O termo *tupla* ou *n-tupla* é uma generalização de dupla, tripla, quádrupla, e assim por diante, quando não se deseja fixar o número de elementos da lista.
- Geralmente, são utilizados os delimitadores $<$ e $>$, mas em alguns livros podem ser encontrados também o uso de parênteses ou colchetes. Um exemplo de uma tupla de três valores é:

$<3, \text{Pedro}, 2003-09-01>$

- A tupla é uma abstração do conceito de registro em arquivos convencionais.
 - Cada valor que aparece na tupla representa um campo do registro.
 - A tupla do exemplo acima poderia representar um registro de uma pessoa, contendo seu código, seu nome e uma data referente à pessoal.

Terminologia

Tipo, domínio e atributo

- Se uma tupla é uma abstração para um registro, a consequência natural é usar um conjunto de tuplas como abstração para um arquivo.
 - Em um arquivo, normalmente, cada campo possui um **tipo**, que define quais valores são admitidos no campo.
 - No exemplo abaixo, os tipos poderiam ser valores numéricos para o primeiro campo, alfanuméricos para o segundo campo e datas para o terceiro campo.

<3, Pedro, 2003-09-01>

- O conjunto de valores possíveis para um determinado tipo é denominado **domínio**.
- Os valores nas tuplas são denominados de **valores de atributo**.

Terminologia

Relação

- Uma **relação**, ou n -relação, sobre os domínios D_1, D_2, \dots, D_n é um conjunto de tuplas $\langle d_1, d_2, \dots, d_n \rangle$ tal que:
 - Todas as tuplas têm o mesmo número n de valores.
 - Em cada tupla, cada valor d_i pertence ao domínio D_i correspondente.
- Para exemplificar o conceito de relação, considere os três domínios abaixo:
 - INTEIRO = $\{1, 2, \dots\}$, o conjunto de números inteiros.
 - NOME = $\{\text{Abel}, \text{Antônio}, \dots\}$, um conjunto de nomes de pessoas.
 - DATA = $\{1980-01-01, 1980-01-02, \dots\}$, um conjunto de datas.
- Uma possível relação definida sobre estes três domínios poderia ser o conjunto de tuplas abaixo:

```
{  
  <1, Antônio, 1991-05-12>  
  <2, Joao, 2001-04-29>  
  <3, Pedro, 2003-09-01>  
}
```

Terminologia

Arquivos vs. Modelo Relacional

- A tabela abaixo resume a correspondência dos termos em arquivos convencionais com a terminologia matemática adotada no Modelo Relacional.

Arquivo convencional	Modelo Relacional
arquivo	relação
registro	tupla
campo	atributo
tipo de campo	domínio
valor de campo	valor de atributo

Terminologia

Arquivos vs. Modelo Relacional

- A tabela abaixo resume a correspondência dos termos em arquivos convencionais com a terminologia matemática adotada no Modelo Relacional.

Arquivo convencional	Modelo Relacional
arquivo	relação
registro	tupla
campo	atributo
tipo de campo	domínio
valor de campo	valor de atributo

- Por que usar os conceitos da Matemática e não ficar simplesmente com a terminologia estabelecida em arquivos convencionais?



Terminologia

Arquivos vs. Modelo Relacional

- A tabela abaixo resume a correspondência dos termos em arquivos convencionais com a terminologia matemática adotada no Modelo Relacional.

Arquivo convencional	Modelo Relacional
arquivo	relação
registro	tupla
campo	atributo
tipo de campo	domínio
valor de campo	valor de atributo

- Por que usar os conceitos da Matemática e não ficar simplesmente com a terminologia estabelecida em arquivos convencionais?
 - A principal razão é que Edgar F. Codd, o criador do Modelo Relacional, definiu as linguagens de consulta do modelo relacional com base em teorias estabelecidas na Matemática (teoria de conjuntos e lógica de predicados). Dessa forma, o uso do conceito de relação matemática auxiliou na definição das linguagens de consulta.



Terminologia

Limitações da definição de relação

- A relação é uma forma interessante de abstrair detalhes de implementação de um arquivo.
- No entanto, do ponto de vista de banco de dados, esta definição é limitada no que tange à referência aos componentes de uma tupla.
 - A única forma de fazer referência a um componente de uma tupla é através de sua posição.
 - Considerando o exemplo abaixo, caso quiséssemos referir ao nome da pessoa, teríamos que usar sua posição: o segundo valor dentro da tupla.

<3, Pedro, 2003-09-01>

- Esta referência por posição não é conveniente na área de banco de dados. Por esta razão, na abordagem relacional, o conceito de relação foi estendido para incluir **nomes** para referenciar os atributos de uma tupla.

Terminologia

Terminologia na prática

- Além da extensão com nomes de atributos, o conceito de relação sofreu outras extensões.
 - Por exemplo, permitir a presença de tuplas duplicadas em uma relação, já que na Matemática, conjuntos não possuem elementos repetidos.
- Consequentemente, no padrão SQL e na maioria dos SGBDs em uso, a terminologia originalmente proposta por Codd foi abandonada em favor de uma terminologia própria de bancos de dados relacionais.
 - Por exemplo, ao invés de falar de relação, fala-se de *tabela* e ao invés de falar de tupla, fala-se em *linha* da tabela.
 - No entanto, na área acadêmica, como em artigos científicos e em alguns livros, ainda é possível encontrar a terminologia original de Codd.
- Dessa forma, de agora em diante vamos utilizar a terminologia da SQL que é mais utilizada na prática e pela maioria dos SGBDs relacionais.
 - Quando necessário, vamos citar a terminologia acadêmica.

Terminologia

Terminologia na prática

- A tabela abaixo apresenta as correspondências entre os conceitos da terminologia original de Codd e os conceitos utilizados na prática.

Notação de Codd	Notação utilizada na prática
relação	tabela
tupla	linha
atributo	coluna
nome de atributo	nome de coluna
valor de atributo	valor (de coluna ou de campo)
domínio	domínio ou tipo

Tabela

- O principal componente de um banco de dados relacional é a **tabela**.
- Uma tabela possui as seguintes propriedades:
 - Cada tabela possui um **nome de tabela**, que a diferencia das demais tabelas no banco de dados.
 - Uma tabela é formada por um conjunto de **linhas**.
 - Uma linha é uma lista ordenada de **valores de campo**.
 - Todas as linhas de uma tabela têm o mesmo número de valores de campo.
 - Com isso, formam-se as **colunas** da tabela.
 - Os valores de uma coluna provêm de um conjunto de valores chamado **tipo** ou **domínio** da coluna.
 - Cada coluna possui um **nome de coluna**, que a diferencia das demais colunas da tabela.



Tabela

- A figura abaixo apresenta um exemplo de uma tabela denominada **pessoa**.
 - Nesta tabela, os nomes das colunas são **codigo**, **nome** e **data_de_nascimento**.
 - Como tipos de coluna, poderíamos ter:
 - Números inteiros para a coluna **codigo**.
 - Cadeias de caracteres (string) para a coluna **nome**.
 - E datas para a coluna **data_de_nascimento**.

pessoa

codigo	nome	data_de_nascimento
1	Antonio	1991-05-12
2	João	2001-04-29
3	Pedro	2003-09-01

- Ao longo do conteúdo da disciplina, muitas vezes será utilizada uma forma abreviada para referenciar uma coluna de uma tabela: o nome da coluna precedido pelo nome da tabela separados por um ponto.
 - Ao invés de escrever: “... a coluna **nome** da tabela **pessoa** ...”,
 - vamos escrever: “... a coluna **pessoa.nome** ...”

Tabela

Ordenação de linhas

- Pela definição matemática, os elementos de um conjunto não são ordenados. Formalmente, temos que:

$$\{a, b\} = \{b, a\}$$

- Como uma tabela representa um conjunto, suas linhas não têm ordenação. Desta forma, não é possível referenciar uma linha por sua posição na tabela.
 - Isso possibilita ao SGBD escolher a ordenação que considerar mais eficiente para o armazenamento interno da tabela.
- Na prática, observou-se que há muitas aplicações que requerem dados ordenados. Por esta razão, a restrição de ordenação é relaxada nos SGBDs baseados em SQL.
 - Neles, é possível especificar que um resultado de uma consulta deva estar ordenado com base em algum critério. No entanto, as tabelas dos bancos de dados continuam sem uma ordenação intrínseca.

Tabela

Duplicatas

- Pela definição matemática de conjuntos, cada um de seus elementos aparece uma única vez, ou seja, dentro de um conjunto não há elementos repetidos:

$$\{a\} = \{a, a\}$$

- Desta forma, se uma tabela é um conjunto, ela não contém linhas duplicadas, ou seja, uma determinada linha não parece mais de uma vez
- Na prática, garantir esta restrição pode trazer problemas ao desempenho de operações nos bancos de dados relacionais.
 - A fim de contornar estes problemas de desempenho, nos SGBDs baseados em SQL, uma tabela passou a ser tratada como um **multi-conjunto** (em inglês, *multi-set* ou *bag*).
 - Um multi-conjunto pode ser definido como uma coleção que admite duplicatas, mas mantém a característica do conjunto não ser ordenado.
 - Quando necessário, as diferenças entre as duas definições de tabela serão apontadas no material da disciplina.

Tabela

Valores de campo são atômicos e monovalorados

- Os valores de campo de uma tabela são **atômicos** e **monovalorados**.
- Ser **atômico** significa que o campo não pode ser composto por outros campos (sub-campos).
 - Por exemplo, em muitas linguagens de programação, um campo não necessita ser atômico, pois pode ser um registro, ou seja, é composto por outros campos.
- Ser **monovalorado** significa que o campo possui um único valor e não um conjunto de valores.
 - Por exemplo, em muitas linguagens de programação, um campo pode ser multivalorado caso seja um vetor (*array*).

Tabela

Caminhos de acesso não são explícitos

- Quando se trabalha com arquivos convencionais do sistema operacional, ao executar uma busca de registro com base em valores de seus campos, muitas vezes usa-se algum **caminho de acesso**.
- Um caminho de acesso é uma estrutura de dados usada para acelerar a busca.
 - Exemplos são os índices que definem uma estrutura com ponteiros para navegação de um registro a outro.
 - Por vezes, uma determinada consulta pode ser resolvida percorrendo caminhos de acesso alternativos, que resultam em diferentes tempos de resposta.
 - Assim, ao usar arquivos convencionais, o programador não tem somente a preocupação de obter os dados requeridos, mas também de encontrar uma forma de obtê-los dentro de um tempo de execução aceitável.
- As linguagens de consulta a banco de dados relacionais permitem buscas por quaisquer critérios, sem que seja necessário especificar quais caminhos de acesso devem ser usados.
 - Um SGBD relacional também emprega caminhos de acesso para acelerar buscas, mas a decisão sobre quais caminhos de acesso devem ser usados na consulta é do SGBD, tem tempo de execução.
 - Ou seja, em um SGBD relacional, ao escrever uma consulta, o programador está preocupado exclusivamente em expressar o resultado que deseja obter. O problema de como obter estes dados de uma forma eficiente é do SGBD.

Tabela

Valor vazio

- Quando uma tabela de um banco de dados relacional é definida, além da especificação do tipo de um valor que pode aparecer em cada coluna (domínio da coluna), é especificado se a coluna pode ou não conter o valor **vazio** (***null*** em inglês).
 - Um exemplo de uma coluna que contém valores vazios é a coluna **email** de uma tabela **contatos** de um banco de dados de uma agenda de contatos.
- Quando uma coluna admite valores vazios, ela é chamada de coluna **opcional**. Caso contrário, ela é chamada de coluna **obrigatória**.
- A abordagem relacional não define qual o significado do valor vazio.
 - No caso do exemplo acima, poderia significar que o contato em questão não possui e-mail ou que o valor é desconhecido pelo usuário da agenda.
 - Em outros casos, o vazio pode ser interpretado como “não se aplica”. Por exemplo, a data de demissão de um empregado cujo contrato encontra-se em vigor.

Chave candidata

- Considere a tabela **cliente** abaixo.

cliente			
codigo_cli	nome	data_nasc	email
10	Souza	<N>	sz@abc-a2.com
32	Santos	1960-05-12	sant10@ff2.edu.br
08	Silva	<N>	silva@abc-a2.com
20	Soares	1972-01-24	soares@ff2.edu.br

- É razoável supor que cada cliente possua um código único, que sirva para diferenciá-lo dos demais.
 - Dependendo da aplicação, também a coluna **email** poderia conter valores únicos.
- Para permitir que esta unicidade de valores seja especificada, a abordagem relacional oferece o conceito de **chave candidata**.
 - Uma chave candidata é qualquer coluna ou combinação de colunas cujos valores servem para distinguir uma linha das demais linhas dentro de uma tabela.
- Considerando novamente tabela **cliente** acima:
 - A coluna **codigo_cli** é uma chave candidata, já que a tabela jamais conterá duas linhas que tenham o mesmo código de cliente.
 - A coluna **email** também poderia ser considerada como uma chave candidata da tabela.

Chave candidata

Chave composta

- No exemplo anterior, cada uma das chaves candidatas é formada de uma única coluna.
 - Entretanto, pela definição do termo, uma chave candidata pode ser composta uma combinação de colunas.
- Considere a tabela **pedido** apresentada abaixo.

pedido			
codigo_cli	data	codigo_prod	quantidade
10	2005-12-05	01	5
10	2005-12-05	02	3
20	2005-05-10	01	4
20	2005-05-20	01	5
08	2004-08-22	02	3

- Esta tabela contém dados sobre pedidos feitos por clientes. Cada linha consta do código do cliente que fez o pedido, da data em que o pedido ocorreu, do código do produto que foi pedido e da quantidade de unidades do produto que foram pedidas.
- Sabe-se que não pode existir mais de um pedido de um determinado cliente para um determinado produto em uma determinada data.
 - Assim, a combinação das colunas **(codigo_cli, data, codigo_prod)** é uma chave candidata desta tabela.

Chave candidata

Minimalidade

- Uma chave candidata, além de conter valores únicos, deve ser formada por uma combinação de colunas que forma de forma que esta combinação seja **mínima**.
- Aqui, mínimo quer dizer que, se retirarmos alguma coluna da combinação de colunas, a combinação resultante deixa de ser chave candidata, ou seja, não mais é suficiente para identificar as linhas da tabela de forma única.
- Por exemplo, considere a combinação de colunas **(codigo_cli, nome)** da tabela abaixo.

cliente

codigo_cli	nome	data_nasc	email
10	Souza	<N>	sz@abc-a2.com
32	Santos	1960-05-12	sant10@ff2.edu.br
08	Silva	<N>	silva@abc-a2.com
20	Soares	1972-01-24	soares@ff2.edu.br

- Em uma rápida análise, esta combinação de colunas poderia ser considerada uma chave candidata, já que a tabela nunca conterá duas linhas com os mesmos valores desta combinação de colunas.
- Entretanto, ela não obedece o requisito de minimalidade. Se a coluna **nome** for retirada da combinação e somente a coluna **codigo_cli** for mantida, esta é suficiente para identificar as linhas dentro da tabela.
- Uma chave não mínima recebe a denominação de **super chave** (ou em inglês, *super key*).

Chave primária e chave alternativa

- Uma tabela pode conter várias chaves candidatas.
- Como pode ser observado na tabela **cliente**, as colunas **codigo_cli** e **email** são chaves candidatas.

cliente

codigo_cli	nome	data_nasc	email
10	Souza	<N>	sz@abc-a2.com
32	Santos	1960-05-12	sant10@ff2.edu.br
08	Silva	<N>	silva@abc-a2.com
20	Soares	1972-01-24	soares@ff2.edu.br

- Diferentemente da tabela **pedido** que contém uma única chave candidata composta pelas colunas (**codigo_cli**, **data**, **codigo_prod**).

pedido

codigo_cli	data	codigo_prod	quantidade
10	2005-12-05	01	5
10	2005-12-05	02	3
20	2005-05-10	01	4
20	2005-05-20	01	5
08	2004-08-22	02	3

Chave primária e chave alternativa

- Dentre as chaves candidatas de uma tabela, uma é escolhida como **chave primária** da tabela.
 - Em inglês, o termo para chave primária é ***primary key***, usualmente abreviado para ***pk***.
- A chave primária é aquela que será usada para relacionar a tabela com outras tabelas do banco de dados.
- As demais chaves candidatas da tabela são denominadas **chaves alternativas**.

Chave primária e chave alternativa

- Considere um banco de dados que reúna as tabelas **cliente** e **pedido**.

cliente			
codigo_cli	nome	data_nasc	email
10	Souza	<N>	sz@abc-a2.com
32	Santos	1960-05-12	sant10@ff2.edu.br
08	Silva	<N>	silva@abc-a2.com
20	Soares	1972-01-24	soares@ff2.edu.br

pedido			
codigo_cli	data	codigo_prod	quantidade
10	2005-12-05	01	5
10	2005-12-05	02	3
20	2005-05-10	01	4
20	2005-05-20	01	5
08	2004-08-22	02	3

- Recorde que a tabela **cliente** tem duas chaves candidatas: a coluna **codigo_cli** e a coluna **email**.
 - No entanto, apenas uma delas deve ser considerada **chave primária**, que neste caso será escolhida a coluna **codigo_cli**.
 - A coluna **email**, outra chave candidata da tabela **cliente**, é considerada como uma **chave alternativa**.
- A tabela **pedido** deve associar cada linha ao cliente que realizou um determinado pedido.
 - Portanto, a coluna **pedido.codigo_cli**, usada para realizar esta associação, utilizada valores de códigos de clientes, já que a coluna **codigo_cli** é a chave primária da tabela **cliente**.
- Na maioria dos SGBDs relacionais e nas versões iniciais da SQL, todas as colunas que compõem a chave primária devem ser obrigatórias, ou seja, não podem conter campos vazios. Já as chaves alternativas não sofrem desta restrição.

Chave estrangeira

- Como visto anteriormente, o relacionamento entre duas tabelas de um banco de dados se dá através do uso de valores da chave primária da uma tabela referenciada na tabela que faz a referência.
- Neste contexto, surge o conceito de **chave estrangeira**.
 - Uma chave estrangeira é uma coluna (ou combinação de colunas) que referencia uma tabela e, portanto, os valores possíveis para esta coluna são aqueles que aparecem na chave primária da tabela referenciada.
 - Em inglês, o termo para chave estrangeira é *foreign key*, usualmente abreviado para *fk*.

Chave estrangeira

- Considere o banco de dados formado pelas tabelas **professor** e **departamento** mostradas abaixo.

departamento


codigo_depto	nome_depto	nivel_depto
1	Informática	Pós-graduação
2	Administração	Graduação
3	Medicina	Graduação

professor

codigo_prof	nome_prof	titulacao_prof	codigo_depto
1	Antônio Souza	Doutor	1
2	Pedro Silva	Mestre	2
3	Felipe Souza	Doutor	2
4	Manuel Silva	Doutor	1
5	Pedro Tavares	Mestre	<N>

Chave estrangeira

- Considere o banco de dados formado pelas tabelas **professor** e **departamento** mostradas abaixo.



The diagram shows a blue arrow pointing from the **codigo_depto** column in the **professor** table to the **codigo_depto** column in the **departamento** table, indicating a foreign key relationship.


codigo_depto	nome_depto	nivel_depto
1	Informática	Pós-graduação
2	Administração	Graduação
3	Medicina	Graduação

codigo_prof	nome_prof	titulacao_prof	codigo_depto
1	Antônio Souza	Doutor	1
2	Pedro Silva	Mestre	2
3	Felipe Souza	Doutor	2
4	Manuel Silva	Doutor	1
5	Pedro Tavares	Mestre	<N>

- A coluna **professor.codigo_depto** contém o código do departamento ao qual o professor está vinculado.
 - Para este banco de dados estar correto, é necessário que todo valor não vazio da coluna **professor.codigo_depto** conste da chave primária de **departamento**.
 - No caso deste exemplo, a coluna chave estrangeira **professor.codigo_depto** é opcional (pode conter valor vazio).
 - Neste caso, o valor vazio indica que um professor não está vinculado a nenhum departamento.
 - Chaves estrangeiras podem ser opcionais, como neste exemplo, ou obrigatórias.
- Desta forma, diz-se que:
 - a coluna **professor.codigo_depto** é chave estrangeira da tabela **departamento**, ou
 - a coluna **professor.codigo_depto** referencia a tabela **departamento**.

Chave estrangeira

- Considere o banco de dados formado pelas tabelas **professor** e **departamento** mostradas abaixo.



The diagram illustrates a foreign key relationship between the **professor** and **departamento** tables. A blue arrow originates from the **codigo_depto** column in the **professor** table and points to the **codigo_depto** column in the **departamento** table, indicating that the professor table references the departamento table.

codigo_depto	nome_depto	nivel_depto
1	Informática	Pós-graduação
2	Administração	Graduação
3	Medicina	Graduação

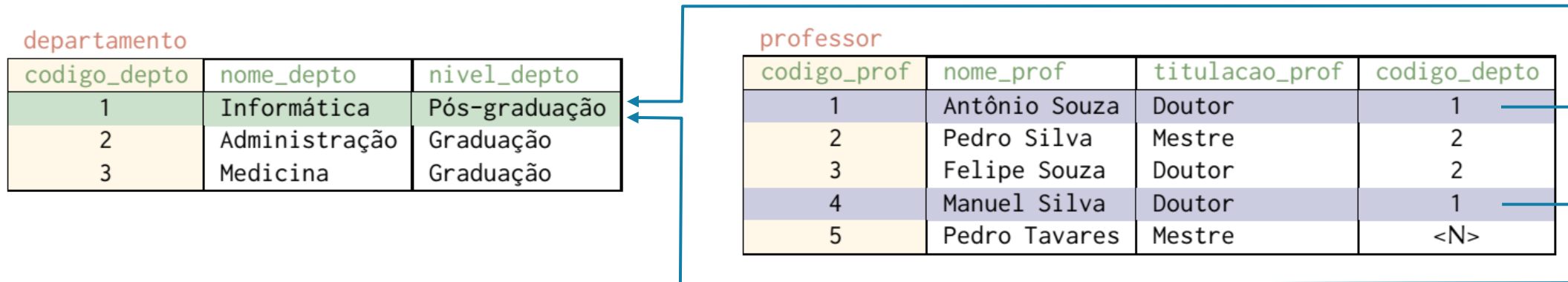
codigo_prof	nome_prof	titulacao_prof	codigo_depto
1	Antônio Souza	Doutor	1
2	Pedro Silva	Mestre	2
3	Felipe Souza	Doutor	2
4	Manuel Silva	Doutor	1
5	Pedro Tavares	Mestre	<N>

- Terminologia

- A tabela que contém a chave estrangeira é chamada de **tabela referenciadora**.
 - No caso de exemplo acima, a tabela referenciadora é a tabela **professor**.
- A tabela que contém a chave primária é chamada de **tabela referenciada**.
 - No caso do exemplo acima, a tabela referenciada é a tabela **departamento**.

Chave estrangeira

- De forma análoga, pode-se designar o relacionamento entre uma **linha referenciada** e uma **linha referenciadora**.



- No exemplo acima, temos a indicação da linha referenciada na tabela **departamento** e suas respectivas linhas referenciadoras na tabela **professor**.

Chave estrangeira

Validações definidas por uma chave estrangeira

- Quando uma das tabelas envolvidas em uma chave estrangeira tiver seu conteúdo atualizado, o SGBD deve executar certas validações.
 - **Inclusão de uma linha na tabela referenciadora.**
 - Neste caso, deve ser garantido que o valor da chave estrangeira da linha incluída, caso não vazio, conste de um valor presente na coluna da chave primária referenciada.
 - **Alteração do valor da chave estrangeira para um valor não vazio.**
 - Neste caso, deve ser garantido que o novo valor conste de um valor presente na coluna da chave primária referenciada.
 - **Exclusão de uma linha da tabela referenciada.**
 - Deve ser garantido que a coluna chave estrangeira não tenha nenhum valor da chave primária que está sendo excluída.
 - **Alteração do valor da chave primária da tabela referenciada.**
 - Deve ser garantido que a coluna chave estrangeira não contenha o antigo valor da chave primária que está sendo alterada.
- Há várias formas de preservar as restrições de integridade especificadas por uma chave estrangeira.
 - O SGBD pode simplesmente impedir a operação no caso de violação da chave estrangeira.
 - Ou o SGBD pode executar uma ação corretiva que leve o banco de dados a um estado válido.
- Através da SQL, ao definir uma chave estrangeira, o projetista do banco de dados pode especificar a forma mais adequada de tratá-la.

Chave estrangeira


Autorreferência

- A palavra “estrangeira” usada para denominar este tipo de chave pode levar a uma interpretação incorreta, pois parece implicar que uma chave estrangeira deva necessariamente referenciar uma outra tabela.
- Entretanto, uma chave estrangeira pode referenciar a chave primária da própria tabela em que se encontra, criando uma **autorreferência**.

Chave estrangeira

Autorreferência

- No exemplo abaixo, a tabela contém uma linha para cada empregado de uma organização. A tabela é formada por três colunas: o código do empregado, seu nome e o código de seu superior hierárquico (coluna **cod_emp_chefe**).
- Cada valor da coluna **cod_emp_chefe** contém o código de um outro empregado, o chefe do empregado em questão.
 - Valores não vazios da coluna **cod_emp_chefe** devem constar de valores presentes na coluna **codigo_emp**.
 - A coluna **cod_emp_chefe** é opcional, visto que podem existir empregados que não possuem um chefe.



empregado

codigo_emp	nome	cod_emp_chefe
10	Pereira	<N>
21	Tavares	10
30	Santos	10
55	Almeida	21

Classes de restrições de integridade

- Uma das funções do SGBD é a manutenção da integridade dos dados sob seu controle.
- Os SGBDs relacionais oferecem várias formas de especificar restrições de integridade, algumas das quais já vistas anteriormente.
- As restrições de integridade oferecidas pelos SGBDs relacionais são classificadas como:
 - **Integridade de domínio**
 - Define o tipo de valor de uma coluna, também chamado de domínio da coluna.
 - **Integridade de vazio**
 - Serve para definir se uma coluna pode ou não conter valor vazio (se a coluna é opcional ou obrigatória).
 - **Integridade de chave**
 - Dá-se o nome de integridade de chave à restrição de unicidade de valores especificada por uma chave candidata.
 - **Integridade referencial**
 - Dá-se o nome de integridade referencial à restrição de integridade especificada por uma chave estrangeira.

Classes de restrições de integridade

Restrições semânticas

- As restrições de integridade são garantidas automaticamente por um SGBD relacional, isto é, não é exigido que o programador escreva procedimentos para garanti-las explicitamente.
- Além destes tipos, podem existir outros tipos de restrição de integridade.
 - De forma geral, todos os demais tipos de restrições recebem a denominação de **restrições de integridade semânticas**.
 - Um exemplo de restrição semântica, no caso de um banco de dados hipotético de uma universidade, poderia ser algo como “um aluno não pode ser diplomado em um curso no qual não tenha atingido o número de créditos mínimo especificado para aquele curso”.
- Restrições semânticas refletem “regras de negócio” da aplicação e podem ser alteradas mais frequentemente do que a estrutura do banco de dados, que tende a permanecer fixa por mais tempo.
 - Por esta razão, usualmente, as restrições semânticas são codificadas nas aplicações e não no esquema do banco de dados.
 - Mesmo assim, a SQL oferece algumas funcionalidades para especificar restrições de integridade semânticas.

Esquema do banco de dados

- A especificação de um banco de dados relacional, chamada de **esquema do banco de dados**, deve conter a definição dos seguintes itens:
 - As **tabelas** que formam o banco de dados.
 - As **colunas** que as tabelas possuem.
 - As **restrições de integridade**.
- Existem diferentes formas de documentar o esquema de um banco de dados relacional.
 - Estas formas podem ser **textual** ou **diagramáticas**.
- Não há um padrão único de representação de esquemas que seja largamente aceito.
 - No entanto, muitos softwares de administração de banco de dados trabalham com notações diagramáticas similares à notação que será apresentada adiante.
 - Existem notações que não inclui todos os detalhes do esquema, seja para não sobrecarregar sua representação visual ou por limitações próprias do estilo de representação adotado.

Esquema do banco de dados

Esquema textual

- Na notação textual, são listadas as tabelas.

departamento

categoria

funcionario

dependente

Esquema do banco de dados

Esquema textual

- Na notação textual, são listadas as tabelas.
 - Para cada tabela, são enumerados, entre parênteses, os nomes das colunas que a compõe.

```
departamento(id, nome)
```

```
categoria(id, nome)
```

```
funcionario(id, nome, departamento_id, categoria_id)
```

```
dependente(funcionario_id, codigo, nome, relacao, data_nascimento)
```

Esquema do banco de dados

Esquema textual

- Na notação textual, são listadas as tabelas.
 - Para cada tabela, são enumerados, entre parênteses, os nomes das colunas que a compõe.
 - Os nomes das colunas que formam a chave primária devem aparecer sublinhadas.

```
departamento(id, nome)
```

```
categoria(id, nome)
```

```
funcionario(id, nome, departamento_id, categoria_id)
```

```
dependente(funcionario_id, codigo, nome, relacao, data_nascimento)
```

Esquema do banco de dados

Esquema textual

- Na notação textual, são listadas as tabelas.
 - Para cada tabela, são enumerados, entre parênteses, os nomes das colunas que a compõe.
 - Os nomes das colunas que formam a chave primária devem aparecer sublinhadas.
 - Após a definição de cada tabela, são definidas as chaves estrangeiras da seguinte forma:
 - Para chave simples: <colunas chave estrangeira> referencia <tabela referenciada>
 - Para chave composta: (<coluna 1>, <coluna 2>, ...) referencia <tabela referenciada>

```
departamento(id, nome)
```

```
categoria(id, nome)
```

```
funcionario(id, nome, departamento_id, categoria_id)  
    departamento_id referencia departamento  
    categoria_id referencia categoria
```

```
dependente(funcionario_id, codigo, nome, relacao, data_nascimento)  
    funcionario_id referencia funcionario
```

Esquema do banco de dados

Esquema textual

- Na notação textual, são listadas as tabelas.
 - Para cada tabela, são enumerados, entre parênteses, os nomes das colunas que a compõe.
 - Os nomes das colunas que formam a chave primária devem aparecer sublinhadas.
 - Após a definição de cada tabela, são definidas as chaves estrangeiras da seguinte forma:
 - Para chave simples: <colunas chave estrangeira> referencia <tabela referenciada>
 - Para chave composta: (<coluna 1>, <coluna 2>, ...) referencia <tabela referenciada>

```
departamento(id, nome)

categoria(id, nome)

funcionario(id, nome, departamento_id, categoria_id)
    departamento_id referencia departamento
    categoria_id referencia categoria

dependente(funcionario_id, codigo, nome, relacao, data_nascimento)
    funcionario_id referencia funcionario
```

Geralmente **não** são utilizados caracteres de espaço em branco nem caracteres especiais (símbolos ou letras acentuadas, entre outros) nos nomes das tabelas e colunas.

Esquema do banco de dados

Esquema textual

- Na notação textual, são listadas as tabelas.
 - Para cada tabela, são enumerados, entre parênteses, os nomes das colunas que a compõe.
 - Os nomes das colunas que formam a chave primária devem aparecer sublinhadas.
 - Após a definição de cada tabela, são definidas as chaves estrangeiras da seguinte forma:
 - Para chave simples: <colunas chave estrangeira> referencia <tabela referenciada>
 - Para chave composta: (<coluna 1>, <coluna 2>, ...) referencia <tabela referenciada>

```
departamento(id, nome)

categoria(id, nome)

funcionario(id, nome, departamento_id, categoria_id)
    departamento_id referencia departamento
    categoria_id referencia categoria

dependente(funcionario_id, codigo, nome, relacao, data_nascimento)
    funcionario_id referencia funcionario
```

Observe que ainda não especificamos os domínios dos atributos. Então, podemos estender essa notação para que sejamos capazes de especificá-los.

Esquema do banco de dados

Esquema textual

- De forma alternativa, podemos estender a notação textual apresentada para especificar o domínio das colunas nas tabelas.
 - Para isso, após o nome de cada coluna, o seu respectivo domínio é informado.

```
departamento(id: int, nome: text)
```

```
categoria(id: int, nome: text)
```

```
funcionario(id: int, nome: text, departamento_id: int, categoria_id: int)  
    departamento_id referencia departamento  
    categoria_id referencia categoria
```

```
dependente(funcionario_id: int, codigo: int, nome: text, relaca: text, data_nascimento: date)  
    funcionario_id referencia funcionario
```

Esquema do banco de dados

Esquema textual

- De forma alternativa, podemos estender a notação textual apresentada para especificar o domínio das colunas nas tabelas.
 - Para isso, após o nome de cada coluna, o seu respectivo domínio é informado.

```
departamento(id: int, nome: text)
```

```
categoria(id: int, nome: text)
```

```
funcionario(id: int, nome: text, departamento_id: int, categoria_id: int)
```

```
    departamento_id referencia departamento
```

```
    categoria_id referencia categoria
```

```
dependente(funcionario_id: int, codigo: int, nome: text, relaca: text, data_nascimento: date)
```

```
    funcionario_id referencia funcionario
```

Note que **categoria_id** referencia a coluna **id** da tabela **categoria**. Portanto, as duas devem ter o mesmo domínio.

Esquema do banco de dados

Esquema textual

- De forma alternativa, podemos estender a notação textual apresentada para especificar o domínio das colunas nas tabelas.
 - Para isso, após o nome de cada coluna, o seu respectivo domínio é informado.

```
departamento(id: int, nome: text)
```

```
categoria(id: int, nome: text)
```

```
funcionario(id: int, nome: text, departamento_id: int, categoria_id: int)
```

```
    departamento_id referencia departamento
```

```
    categoria_id referencia categoria
```

```
dependente(funcionario_id: int, codigo: int, nome: text, relaca: text, data_nascimento: date)
```

```
    funcionario_id referencia funcionario
```

Note que **categoria_id** referencia a coluna **id** da tabela **categoria**. Portanto, as duas devem ter o mesmo domínio.

- Esta notação ainda é bastante limitada.
 - Com ela ainda não somos capazes de especificar as restrições de nulo, nem chave alternativa.

Esquema do banco de dados

Esquema diagramático

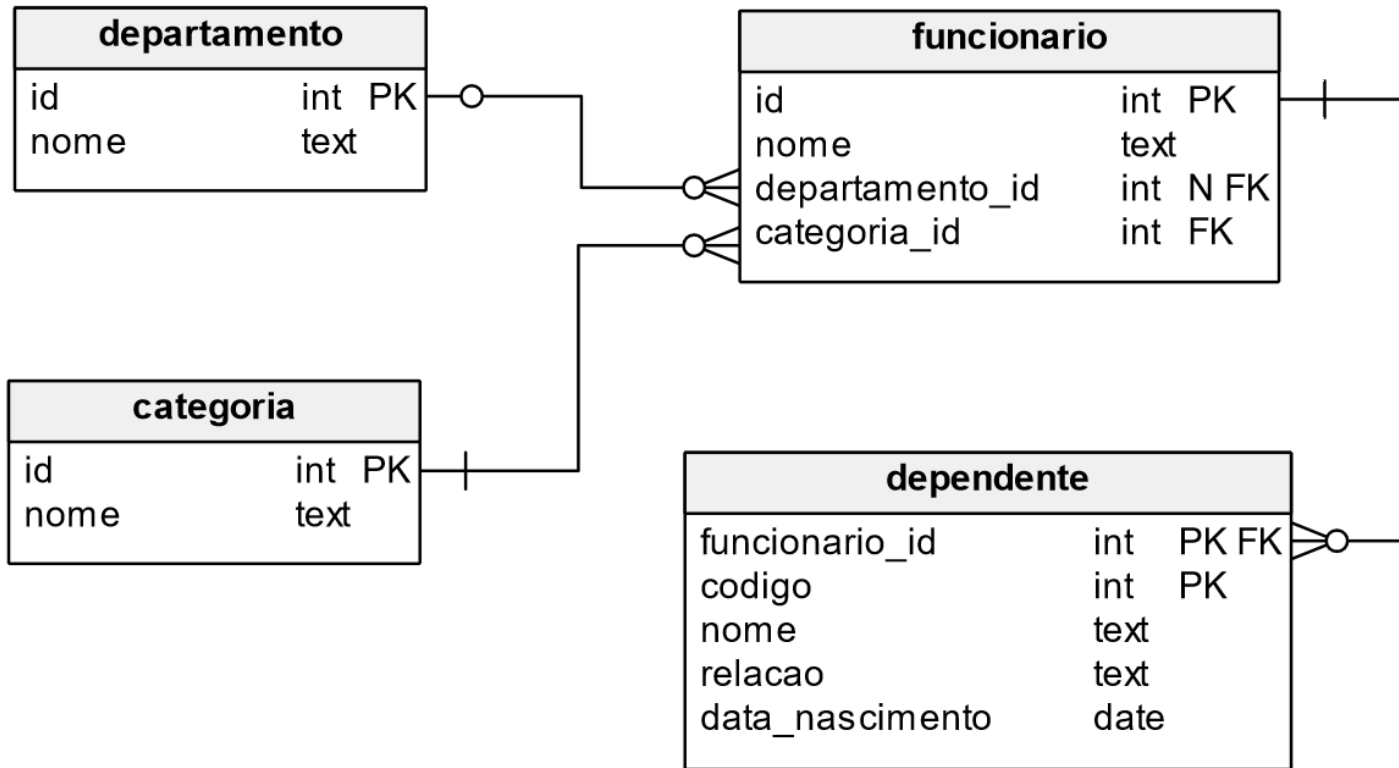
- Uma segunda possibilidade para representação de esquemas de banco de dados relacionais é através de diagramas.
 - Uma representação por meio de diagramas é mais comum na prática.
 - A visualização das tabelas e suas associações por meio de chaves estrangeiras ficam mais fáceis de serem identificadas.
- Muitas ferramentas CASE trabalham com representações diagramáticas.
 - Ferramentas CASE (do inglês, *Computed-Aided Software Engineering*) são ferramentas baseadas em computadores que auxiliam atividades de engenharia de software, desde a análise de requisitos e modelagem até programação e testes.
 - Não há um padrão diagramático para representação de esquemas de banco de dados relacionais.
 - Diferentes ferramentas CASE podem utilizar diferentes notações.
- Alguns exemplos de ferramentas CASE para a elaboração de esquemas de banco de dados relacionais por meio de diagramas são:
 - **Vertabelo** (online), disponível em: <https://vertabelo.com/>
 - **Draw.io** (online e desktop), disponível em: <https://app.diagrams.net/>
 - **DBDesigner** (online), disponível em: <https://dbdesigner.id/>
 - **GenMyModel** (online), disponível em: <https://www.genmymodel.com/>
 - **Lucidchart** (online) disponível em: <https://www.lucidchart.com/>

Esquema do banco de dados

Esquema diagramático

- Exemplo de notação diagramática para representação de um esquema de banco de dados relacionais:

Diagrama criado com Vertabelo (<https://vertabelo.com/>)

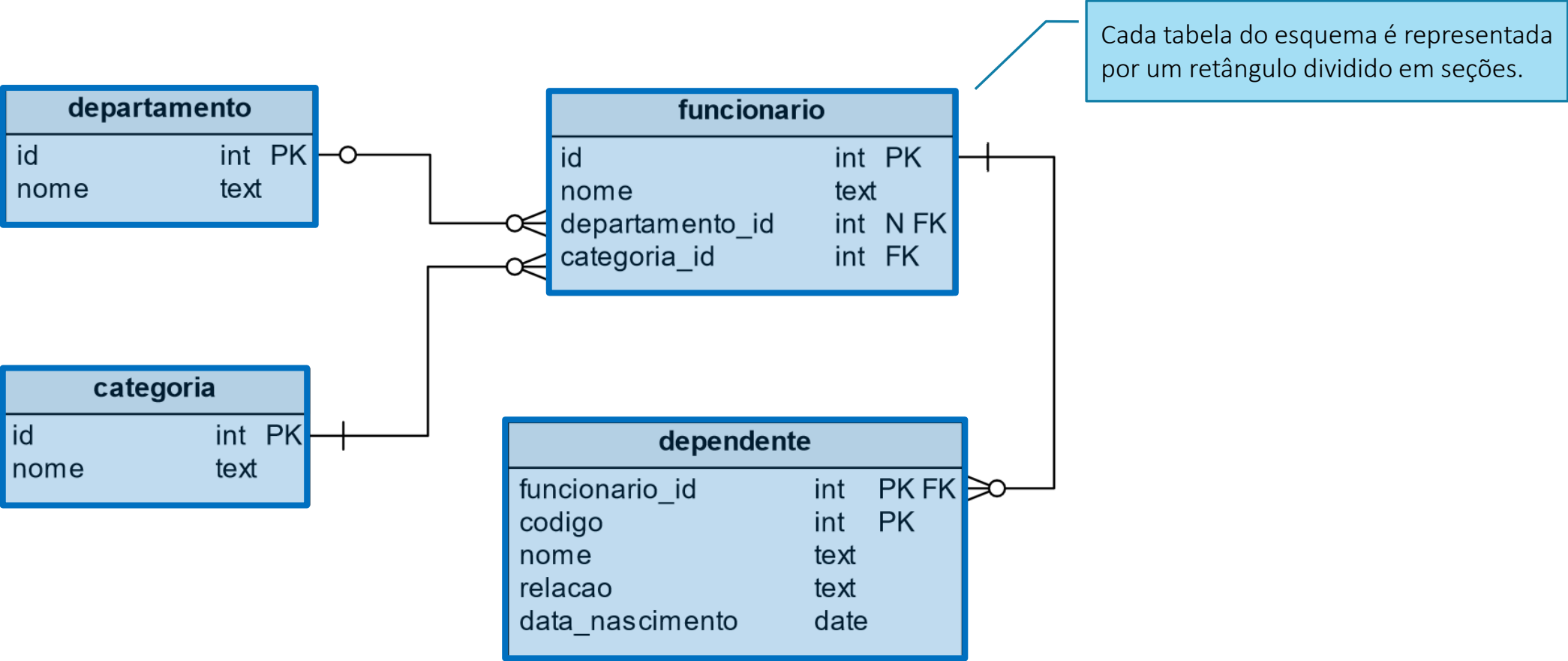


Esquema do banco de dados

Esquema diagramático

- Exemplo de notação diagramática para representação de um esquema de banco de dados relacionais:

Diagrama criado com Vertabelo (<https://vertabelo.com/>)

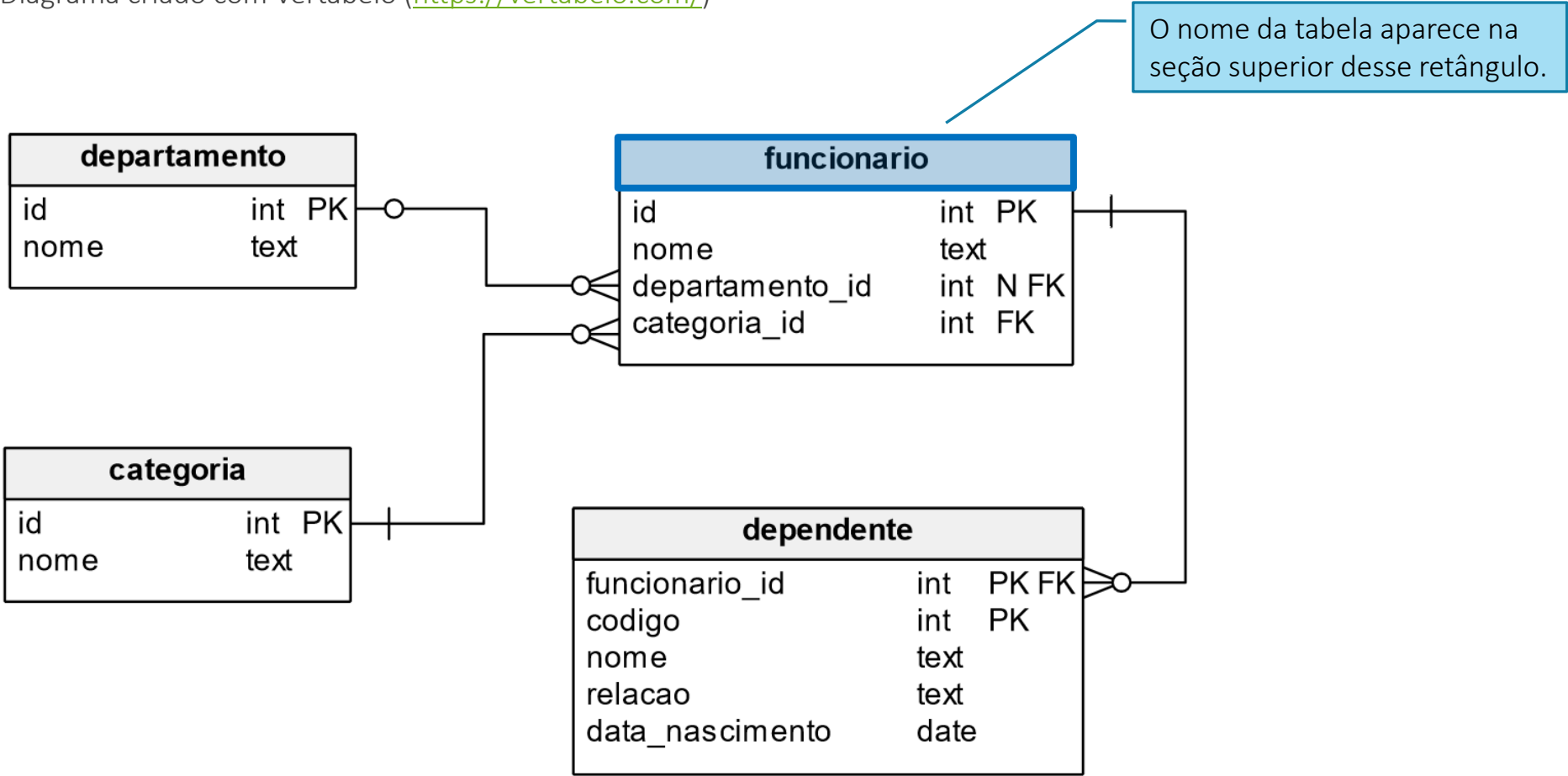


Esquema do banco de dados

Esquema diagramático

- Exemplo de notação diagramática para representação de um esquema de banco de dados relacionais:

Diagrama criado com Vertabelo (<https://vertabelo.com/>)

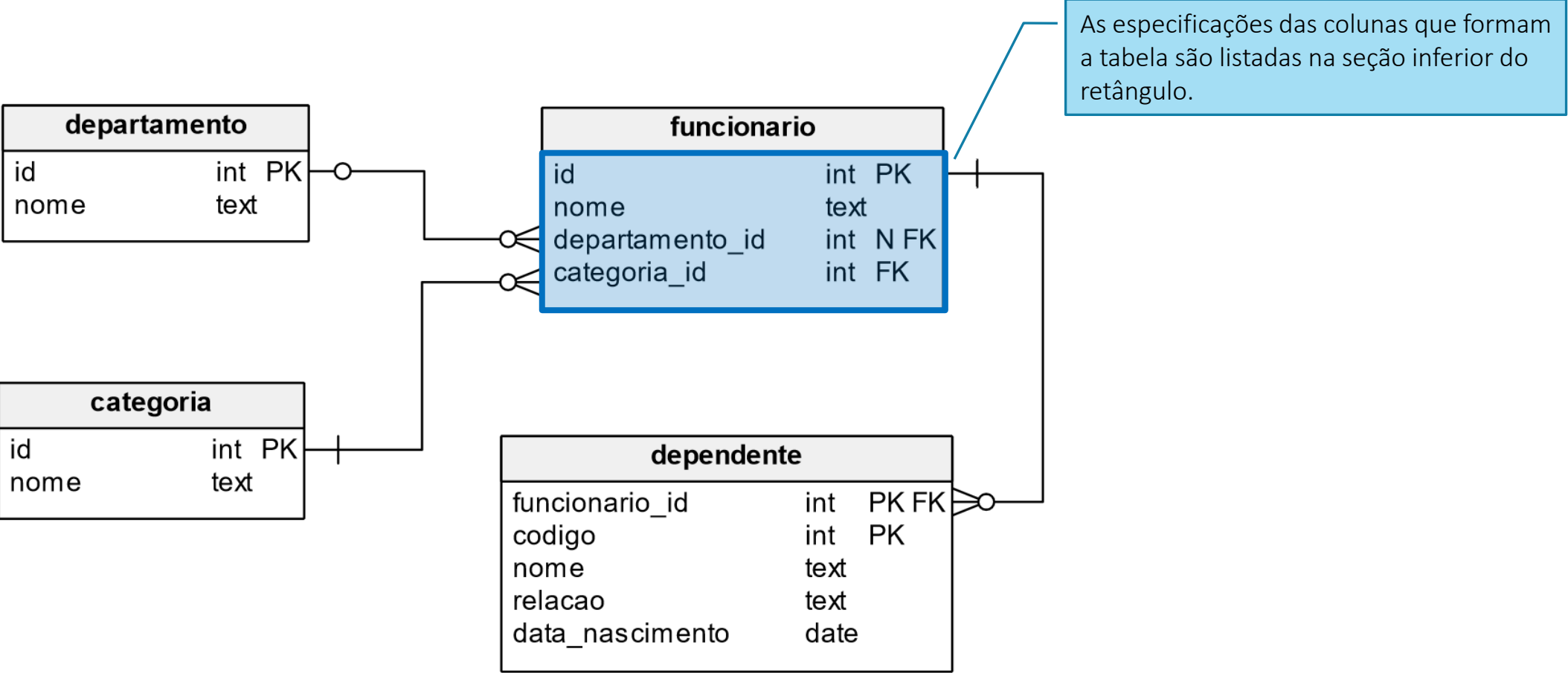


Esquema do banco de dados

Esquema diagramático

- Exemplo de notação diagramática para representação de um esquema de banco de dados relacionais:

Diagrama criado com Vertabelo (<https://vertabelo.com/>)

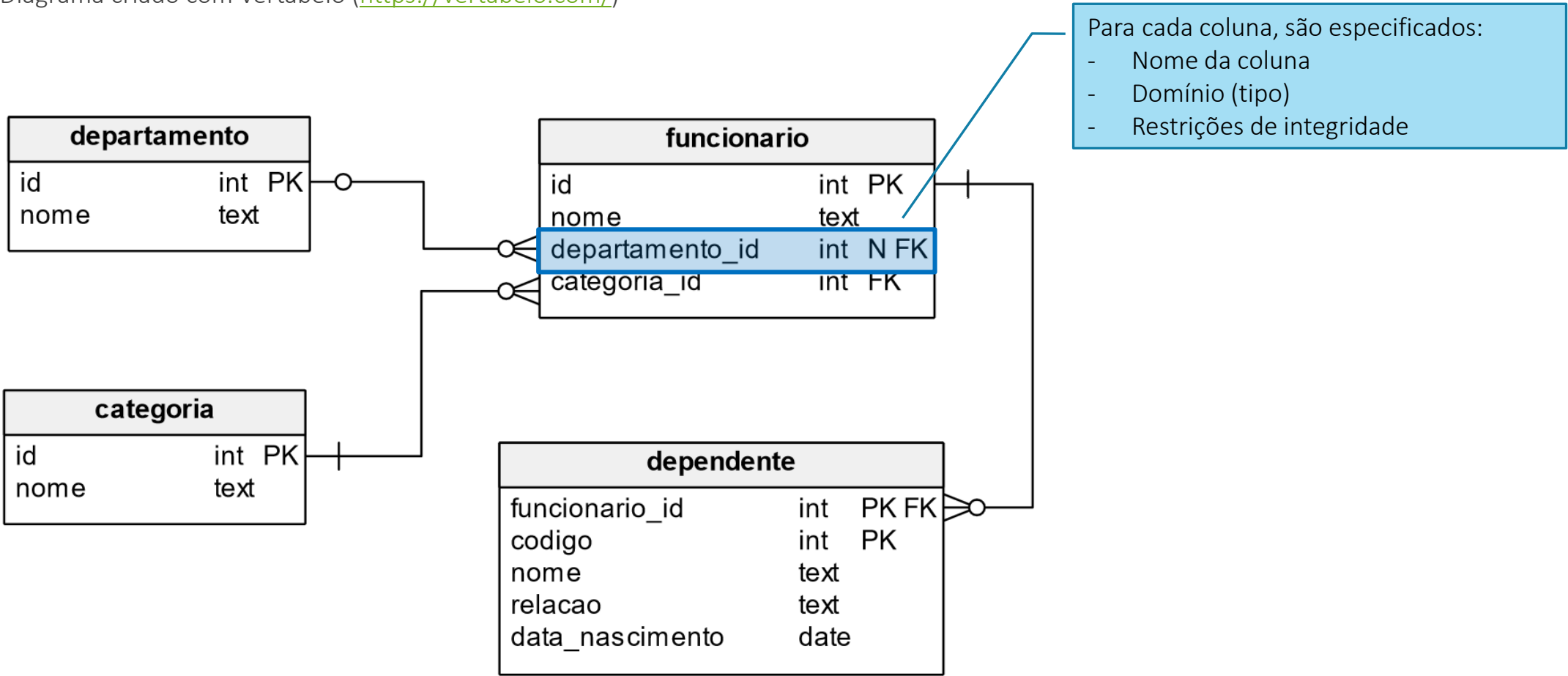


Esquema do banco de dados

Esquema diagramático

- Exemplo de notação diagramática para representação de um esquema de banco de dados relacionais:

Diagrama criado com Vertabelo (<https://vertabelo.com/>)

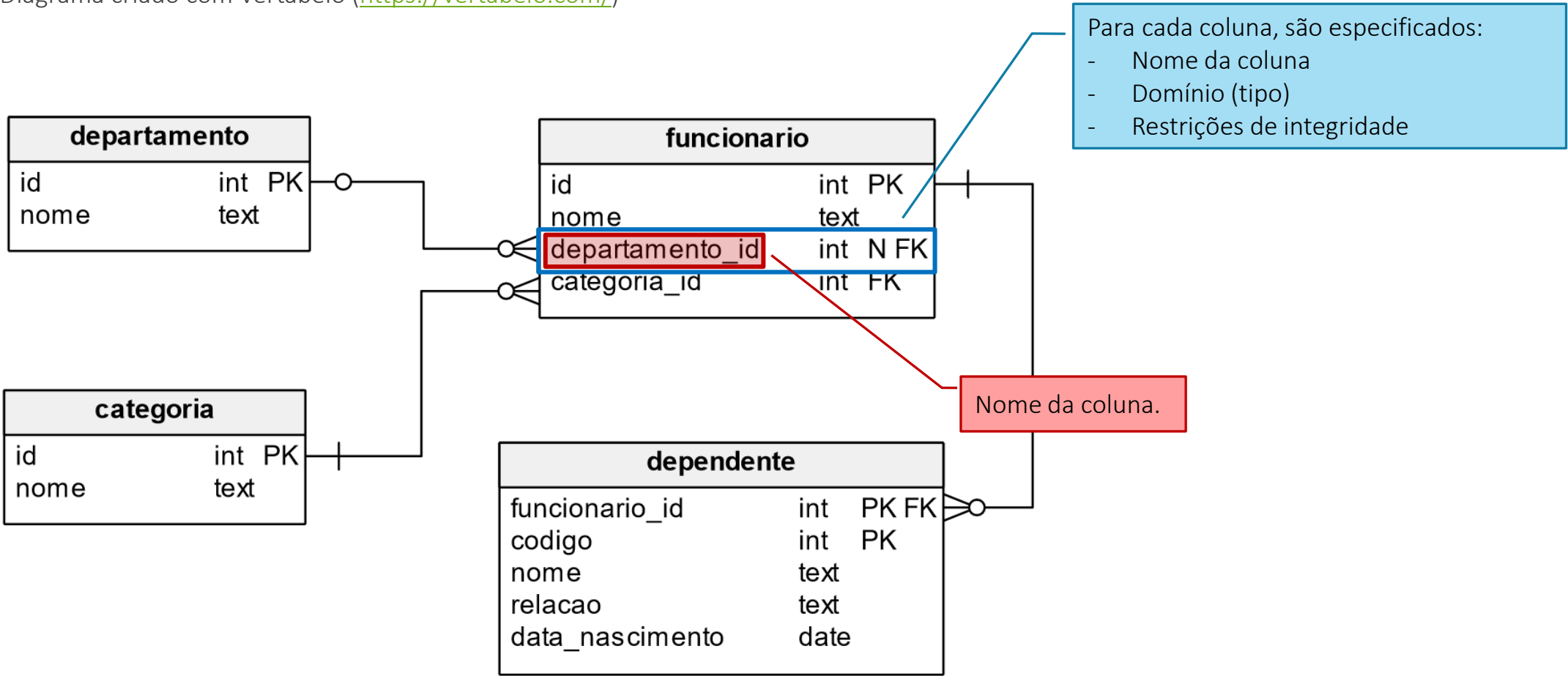


Esquema do banco de dados

Esquema diagramático

- Exemplo de notação diagramática para representação de um esquema de banco de dados relacionais:

Diagrama criado com Vertabelo (<https://vertabelo.com/>)

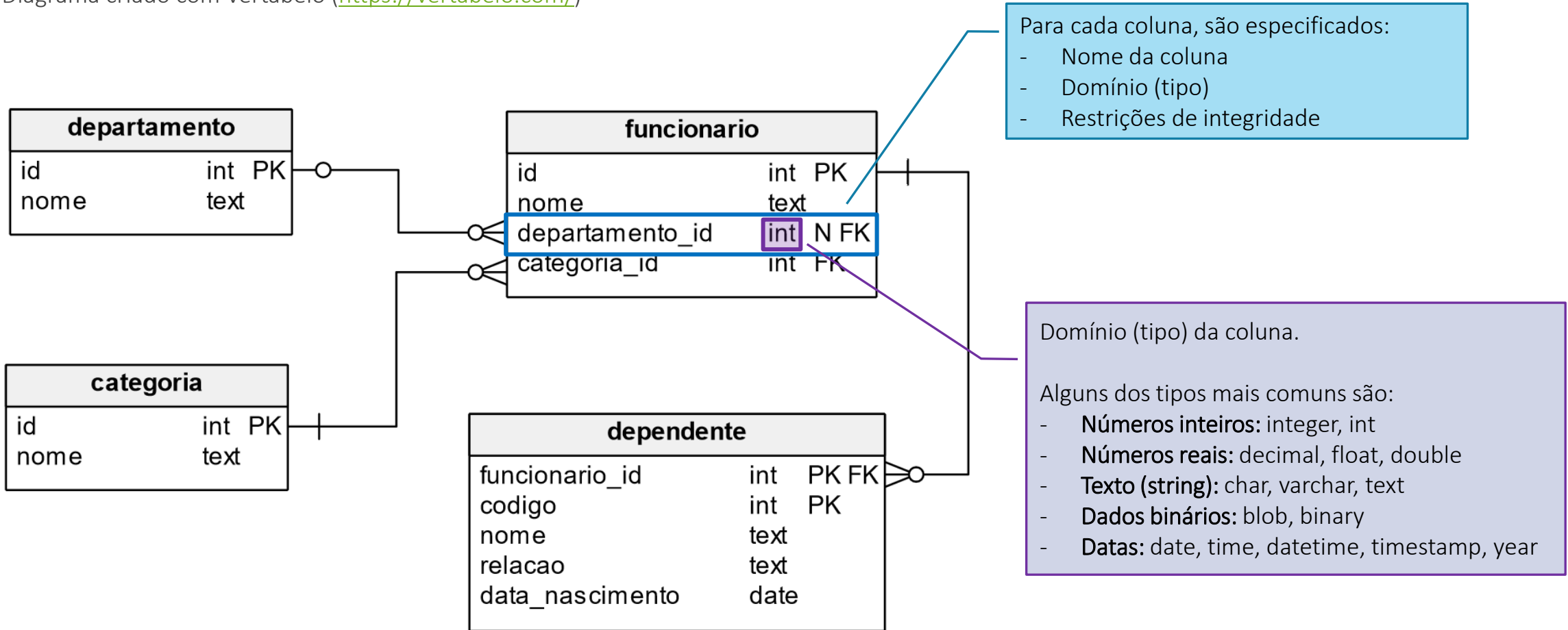


Esquema do banco de dados

Esquema diagramático

- Exemplo de notação diagramática para representação de um esquema de banco de dados relacionais:

Diagrama criado com Vertabelo (<https://vertabelo.com/>)

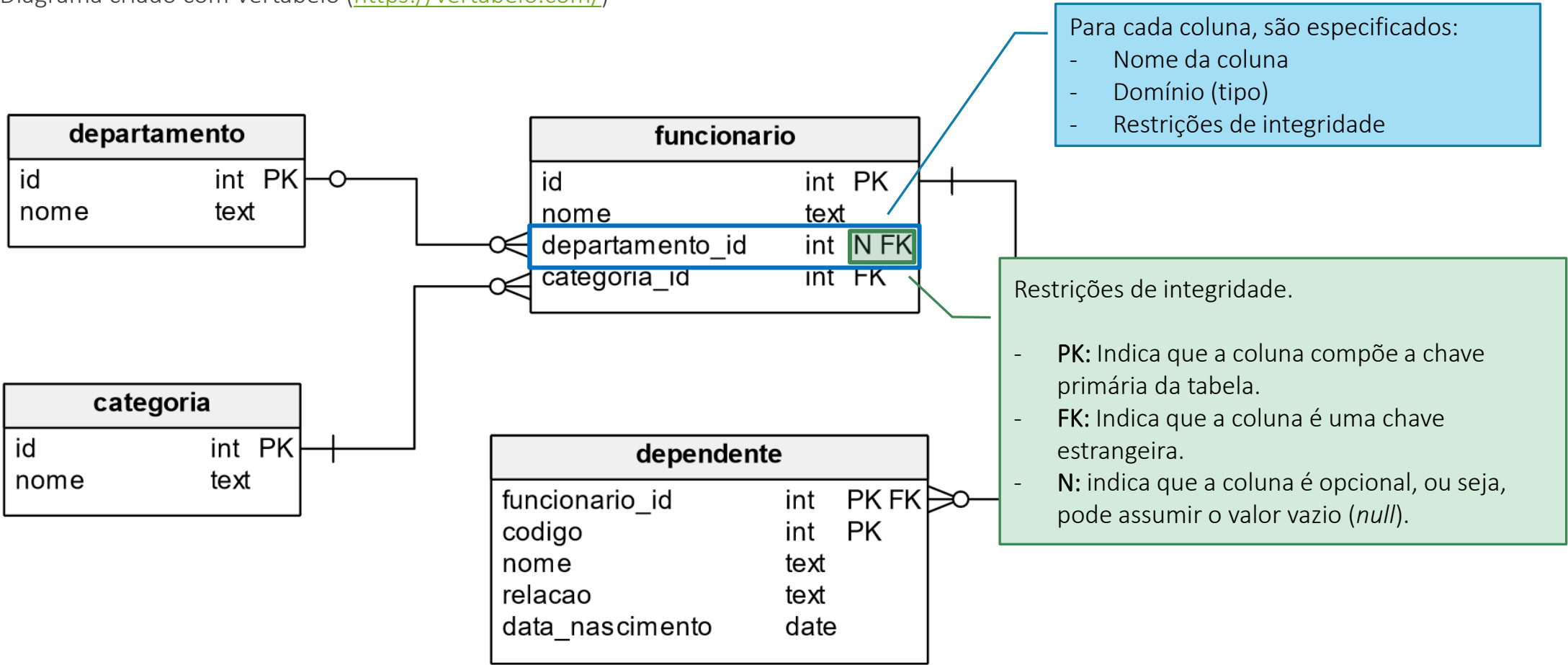


Esquema do banco de dados

Esquema diagramático

- Exemplo de notação diagramática para representação de um esquema de banco de dados relacionais:

Diagrama criado com Vertabelo (<https://vertabelo.com/>)

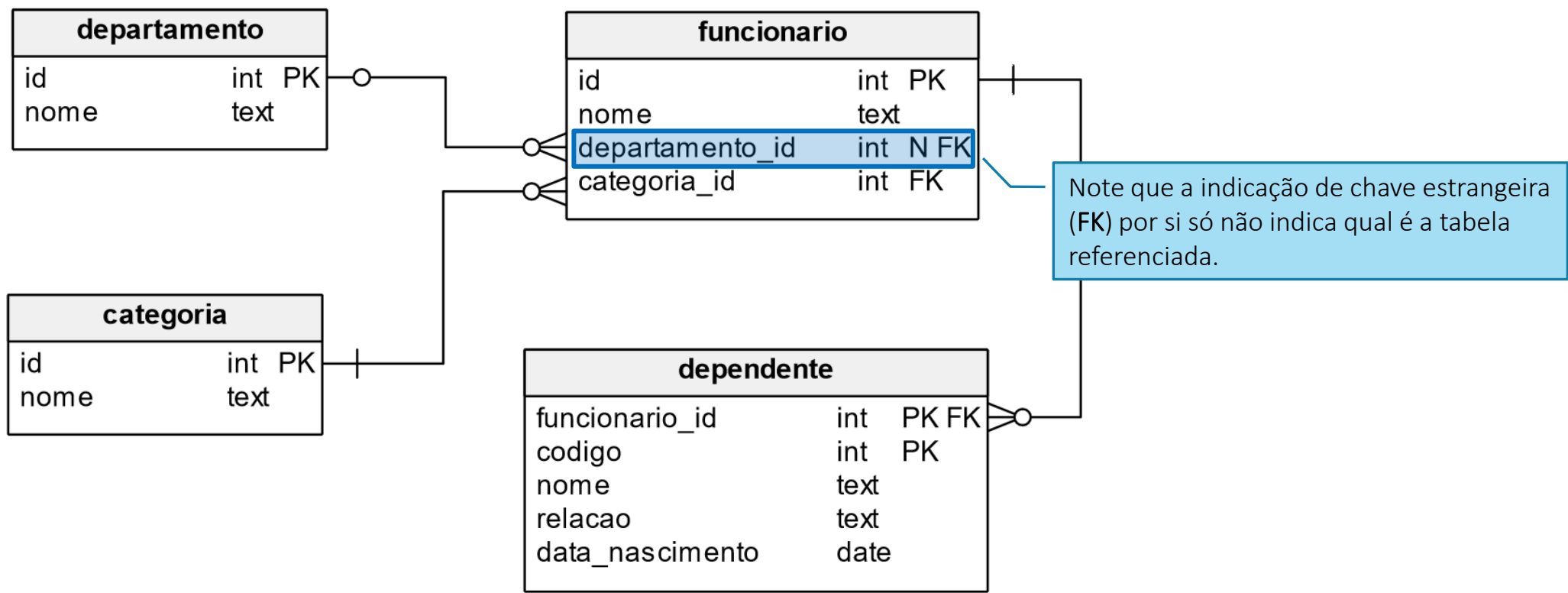


Esquema do banco de dados

Esquema diagramático

- Exemplo de notação diagramática para representação de um esquema de banco de dados relacionais:

Diagrama criado com Vertabelo (<https://vertabelo.com/>)

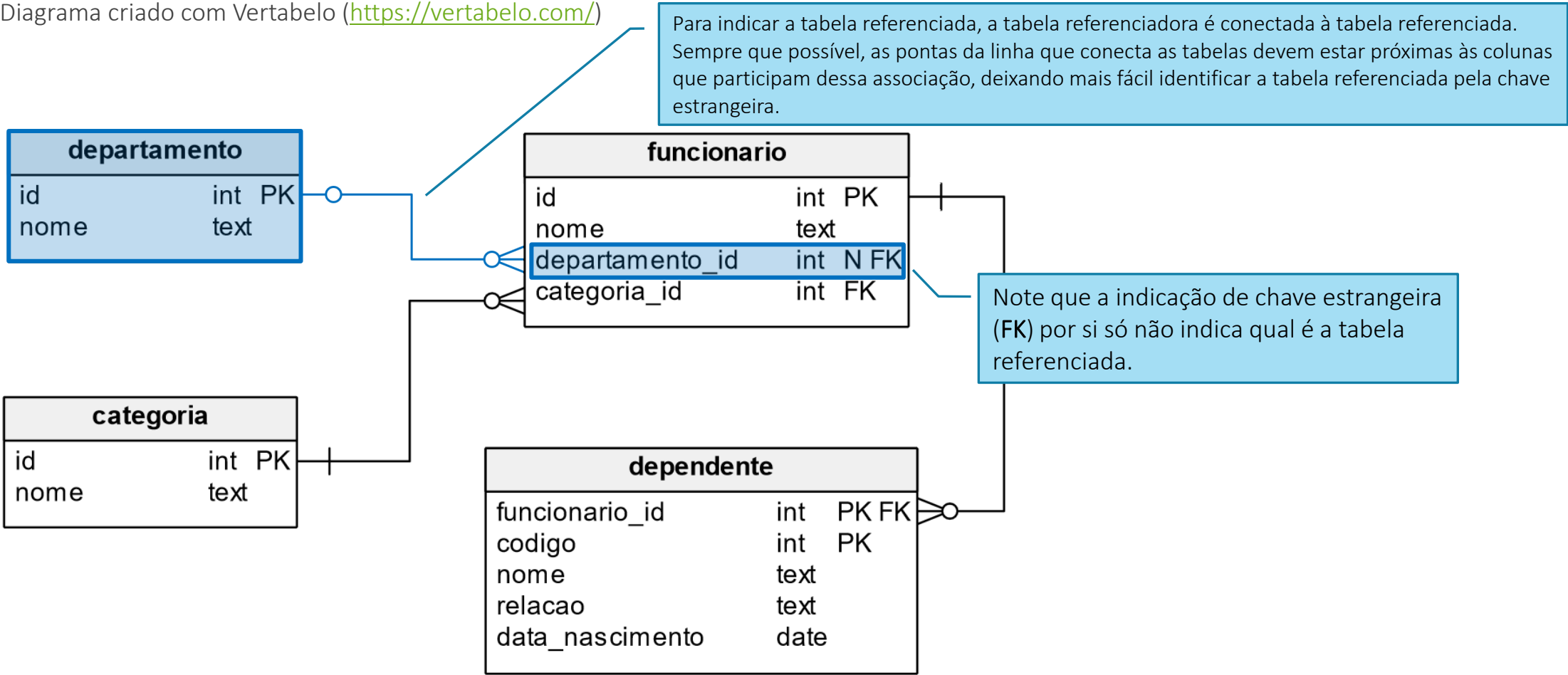


Esquema do banco de dados

Esquema diagramático

- Exemplo de notação diagramática para representação de um esquema de banco de dados relacionais:

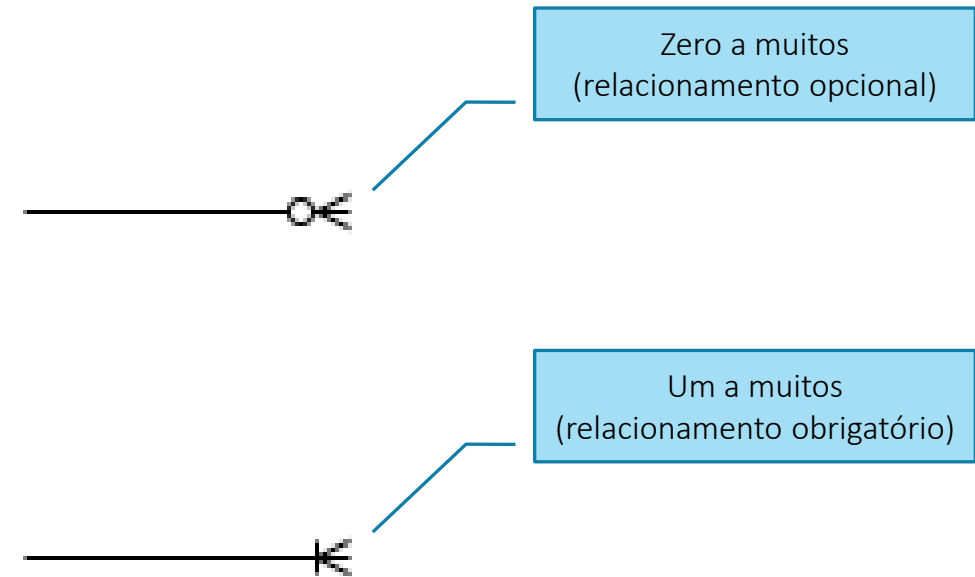
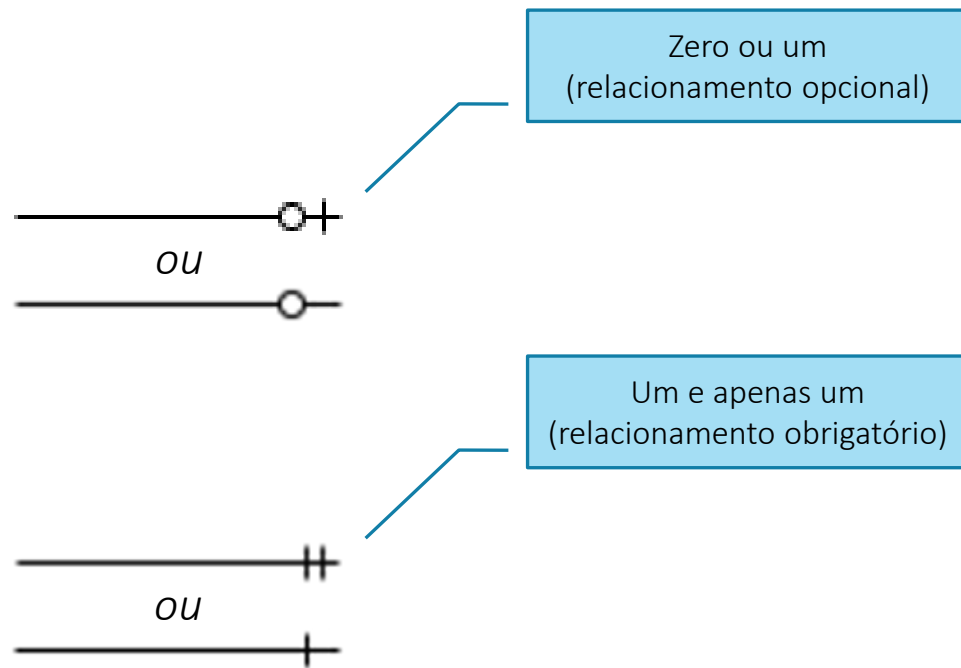
Diagrama criado com Vertabelo (<https://vertabelo.com/>)



Esquema do banco de dados

Esquema diagramático

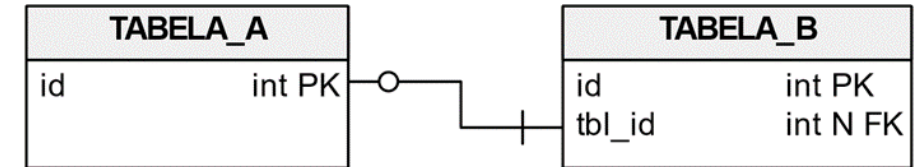
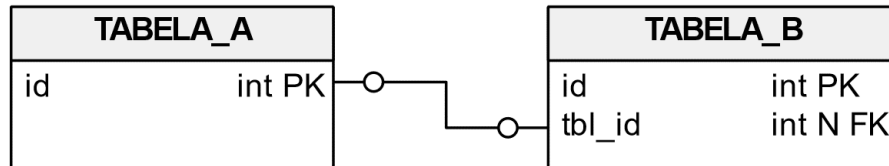
- As extremidades das linhas que conectam as tabelas referenciada e referenciadora indicam a **cardinalidade** de possíveis associações entre as linhas das tabela.
- A notação apresentada abaixo é comumente chamada de **notação de pés de galinha**.
 - Esta notação é amplamente aceita como o estilo mais intuitivo.



Esquema do banco de dados

Esquema diagramático

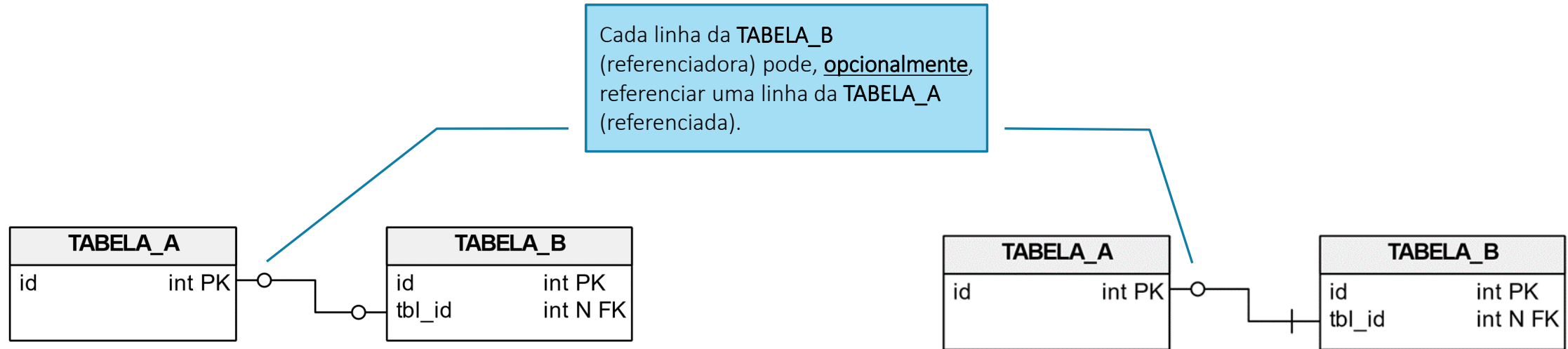
- Relacionamento um para um.
 - Chave estrangeira podem assumir valor vazio (participação opcional).



Esquema do banco de dados

Esquema diagramático

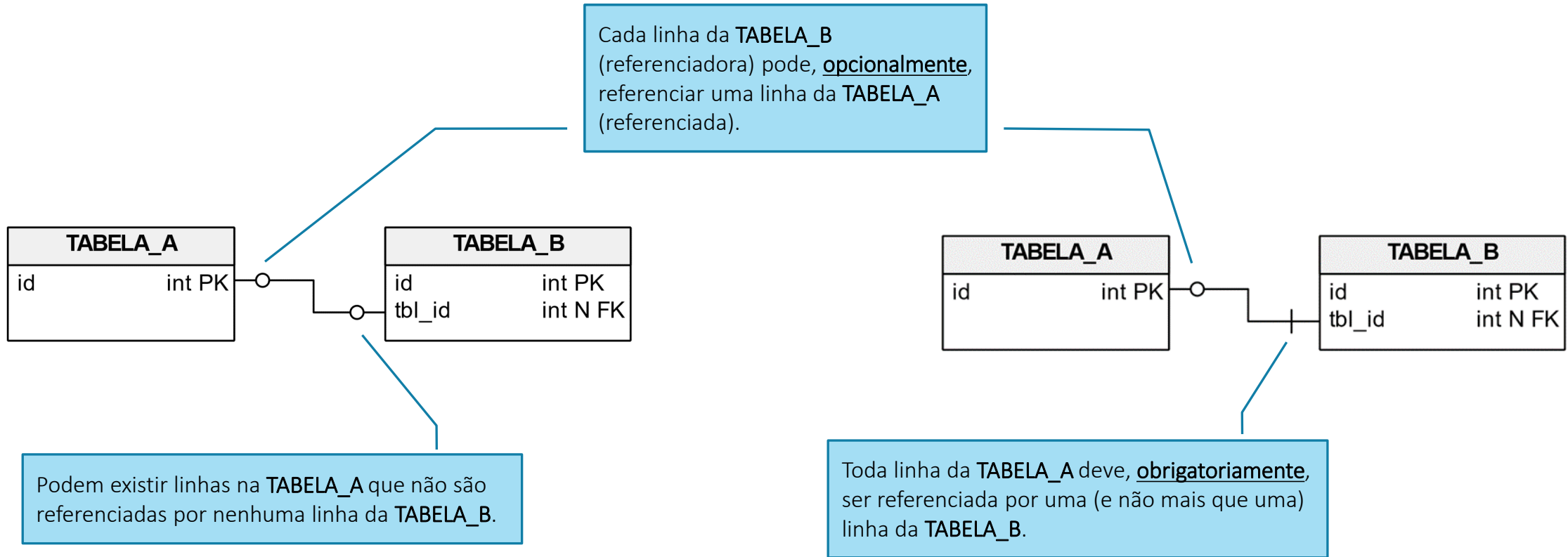
- Relacionamento um para um.
 - Chave estrangeira podem assumir valor vazio (participação opcional).



Esquema do banco de dados

Esquema diagramático

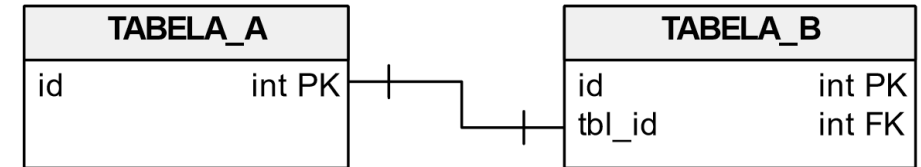
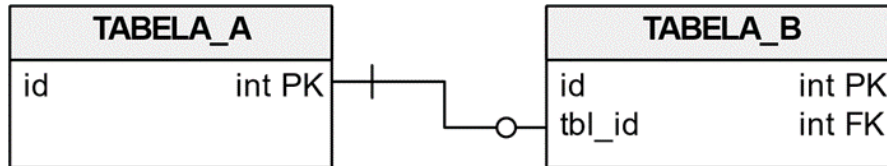
- Relacionamento um para um.
 - Chave estrangeira podem assumir valor vazio (participação opcional).



Esquema do banco de dados

Esquema diagramático

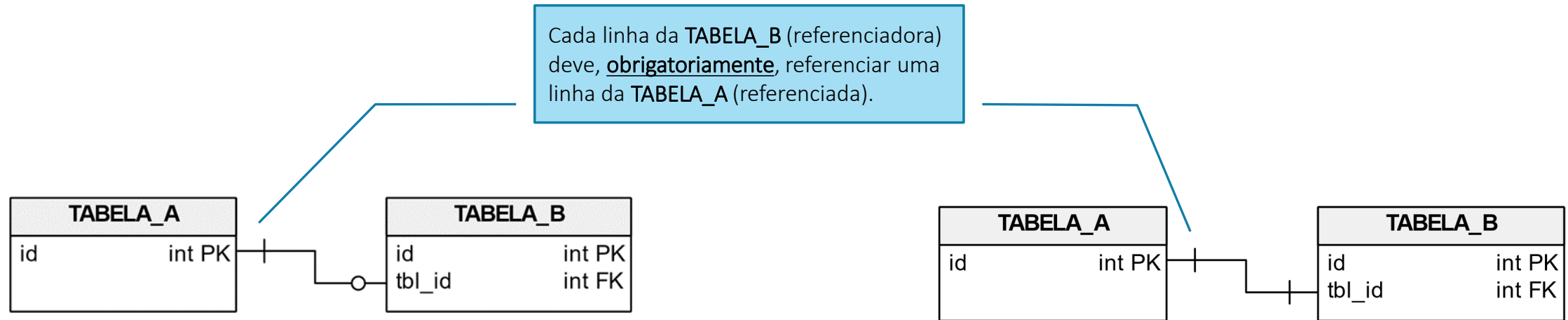
- Relacionamento um para um.
 - Chave estrangeira não podem assumir valor vazio (participação obrigatória).



Esquema do banco de dados

Esquema diagramático

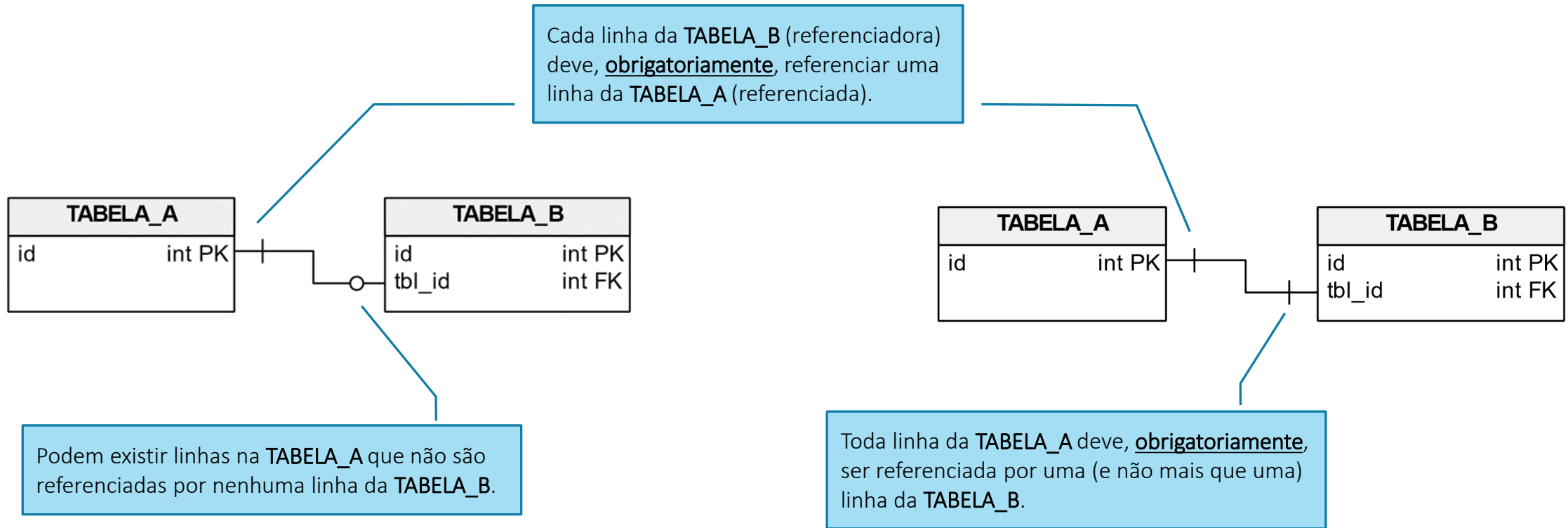
- Relacionamento um para um.
 - Chave estrangeira não podem assumir valor vazio (participação obrigatória).



Esquema do banco de dados

Esquema diagramático

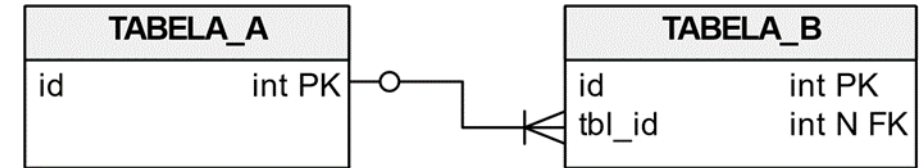
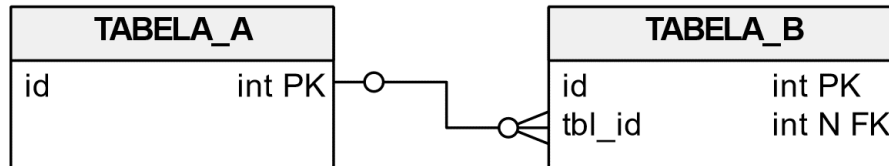
- Relacionamento um para um.
 - Chave estrangeira não podem assumir valor vazio (participação obrigatória).



Esquema do banco de dados

Esquema diagramático

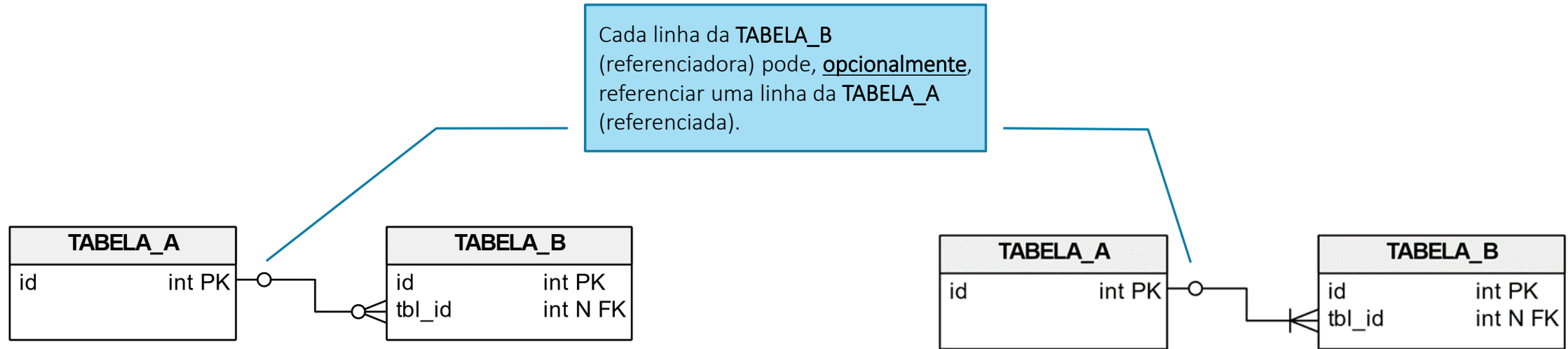
- Relacionamento um para muitos.
 - Chave estrangeira podem assumir valor vazio (participação opcional).



Esquema do banco de dados

Esquema diagramático

- Relacionamento um para muitos.
 - Chave estrangeira podem assumir valor vazio (participação opcional).

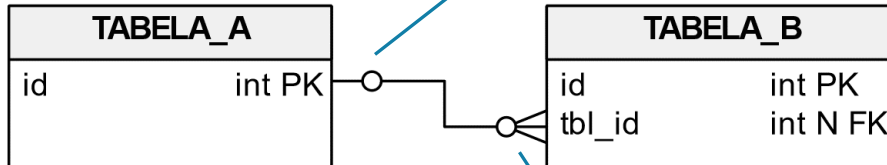


Esquema do banco de dados

Esquema diagramático

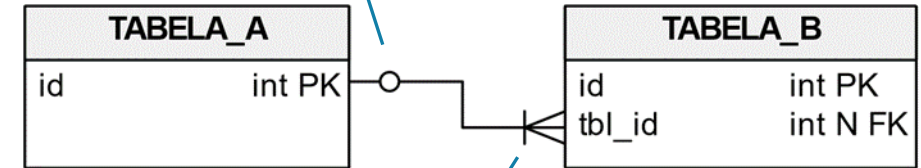
- Relacionamento um para muitos.
 - Chave estrangeira podem assumir valor vazio (participação opcional).

Cada linha da **TABELA_B** (referenciadora) pode, opcionalmente, referenciar uma linha da **TABELA_A** (referenciada).



Podem existir linhas na **TABELA_A** que não são referenciadas por nenhuma linha da **TABELA_B**.

Uma mesma linha da **TABELA_A** pode ser referenciada por mais de uma linha da **TABELA_B**.

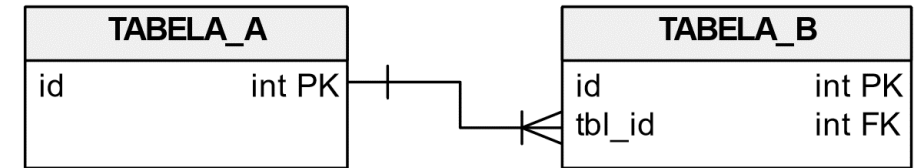
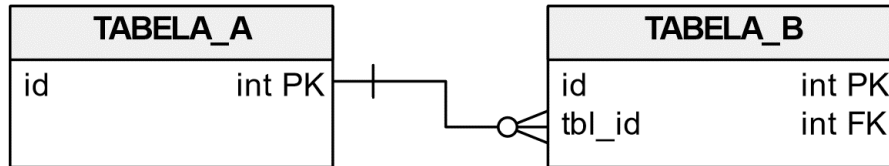


Toda linha da **TABELA_A** deve, obrigatoriamente, ser referenciada por pelo menos uma linha da **TABELA_B**.

Esquema do banco de dados

Esquema diagramático

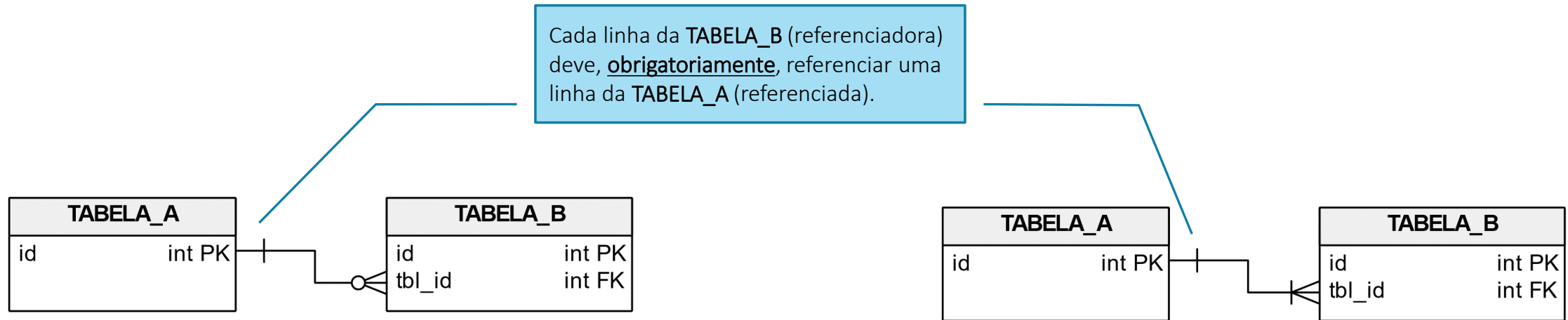
- Relacionamento um para muitos.
 - Chave estrangeira não podem assumir valor vazio (participação obrigatória).



Esquema do banco de dados

Esquema diagramático

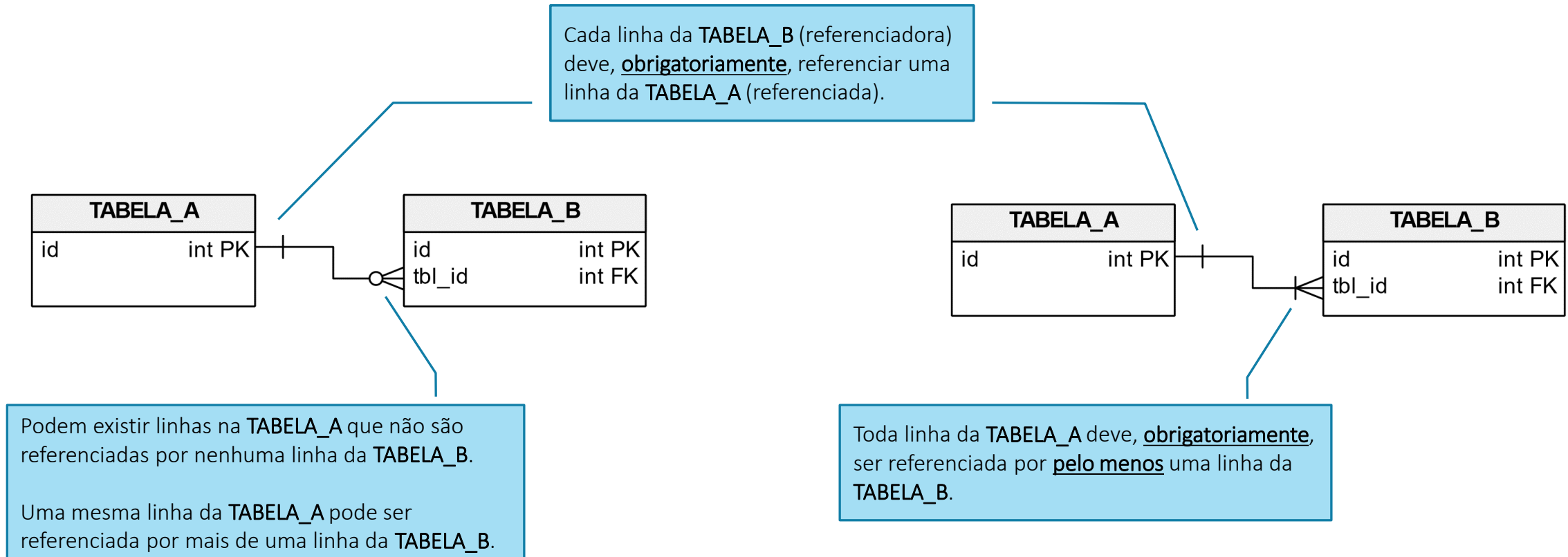
- Relacionamento um para muitos.
 - Chave estrangeira não podem assumir valor vazio (participação obrigatória).



Esquema do banco de dados

Esquema diagramático

- Relacionamento um para muitos.
 - Chave estrangeira não podem assumir valor vazio (participação obrigatória).



Esquema do banco de dados

Esquema diagramático

- Observe que em nenhum momento foi apresentado uma associação do tipo **muitos para muitos** entre as tabelas referenciada e referenciadora.
- Suponha um banco de dados de uma livraria onde são armazenados os dados dos autores e os livros publicados por eles.

autor

id	nome
1	R. Elmasri
2	S. B. Navathe
3	C. A. Heuser

livro

id	titulo	edicao	ano
1	Sistemas de banco de dados	7	2019
2	Banco de dados relacional	1	2019
3	Projeto de banco de dados	6	2008

Esquema do banco de dados

Esquema diagramático

- Observe que em nenhum momento foi apresentado uma associação do tipo **muitos para muitos** entre as tabelas referenciada e referenciadora.
- Suponha um banco de dados de uma livraria onde são armazenados os dados dos autores e os livros publicados por eles.

autor

id	nome
1	R. Elmasri
2	S. B. Navathe
3	C. A. Heuser

livro

id	titulo	edicao	ano
1	Sistemas de banco de dados	7	2019
2	Banco de dados relacional	1	2019
3	Projeto de banco de dados	6	2008

Como associar os livros com seus respectivos autores?



Esquema do banco de dados

Esquema diagramático

- Observe que em nenhum momento foi apresentado uma associação do tipo **muitos para muitos** entre as tabelas referenciada e referenciadora.
- Suponha um banco de dados de uma livraria onde são armazenados os dados dos autores e os livros publicados por eles.

autor

id	nome	livros
1	R. Elmasri	1
2	S. B. Navathe	1
3	C. A. Heuser	2, 3

livro

id	titulo	edicao	ano
1	Sistemas de banco de dados	7	2019
2	Banco de dados relacional	1	2019
3	Projeto de banco de dados	6	2008

Talvez possamos criar uma coluna na tabela **autor** e, para cada autor, listar os ids dos seus livros...



Esquema do banco de dados

Esquema diagramático

- Observe que em nenhum momento foi apresentado uma associação do tipo **muitos para muitos** entre as tabelas referenciada e referenciadora.
- Suponha um banco de dados de uma livraria onde são armazenados os dados dos autores e os livros publicados por eles.

autor

id	nome	livros
1	R. Elmasri	
2	S. B. Navathe	1
3	C. A. Heuser	3

livro

id	titulo	edicao	ano
1	Sistemas de banco de dados	7	2019
2	Banco de dados relacional	1	2019
3	Projeto de banco de dados	6	2008

Esqueceu que os campos de uma tabela são **atômicos** e **monovalorados**?

Os campos de uma tabela podem ter apenas um valor, e não uma lista...



Esquema do banco de dados

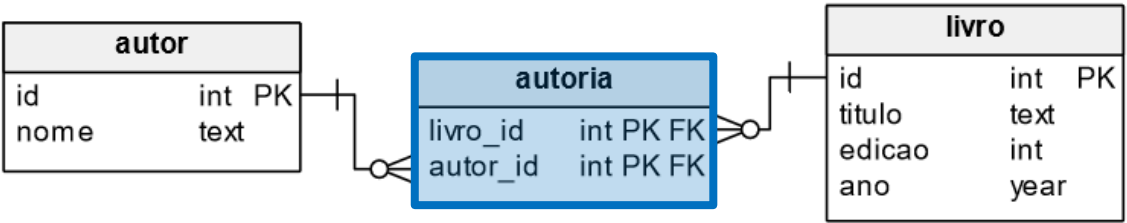
Esquema diagramático

- Observe que em nenhum momento foi apresentado uma associação do tipo **muitos para muitos** entre as tabelas referenciada e referenciadora.
- Suponha um banco de dados de uma livraria onde são armazenados os dados dos autores e os livros publicados por eles.

autor

id	nome
1	R. Elmasri
2	S. B. Navathe
3	C. A. Heuser

Apesar de não ser possível implementar diretamente uma associação do tipo **muito para muitos** em um banco de dados relacional, sua implementação é realizada através do uso de uma **tabela adicional**.



livro

id	titulo	edicao	ano
1	Sistemas de banco de dados	7	2019
2	Banco de dados relacional	1	2019
3	Projeto de banco de dados	6	2008

autoria

livro_id	autor_id
1	1
1	2
2	3
3	3



Esquema do banco de dados

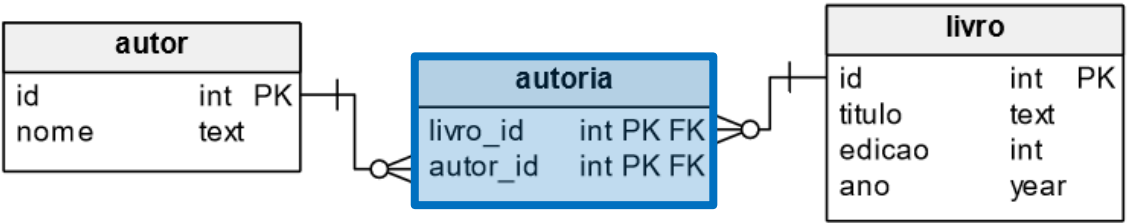
Esquema diagramático

- Observe que em nenhum momento foi apresentado uma associação do tipo **muitos para muitos** entre as tabelas referenciada e referenciadora.
- Suponha um banco de dados de uma livraria onde são armazenados os dados dos autores e os livros publicados por eles.

autor

id	nome
1	R. Elmasri
2	S. B. Navathe
3	C. A. Heuser

Apesar de não ser possível implementar diretamente uma associação do tipo **muito para muitos** em um banco de dados relacional, sua implementação é realizada através do uso de uma **tabela adicional**.



livro

id	titulo	edicao	ano
1	Sistemas de banco de dados	7	2019
2	Banco de dados relacional	1	2019
3	Projeto de banco de dados	6	2008

autoria

livro_id	autor_id
1	1
1	2
2	3
3	3



Esquema do banco de dados

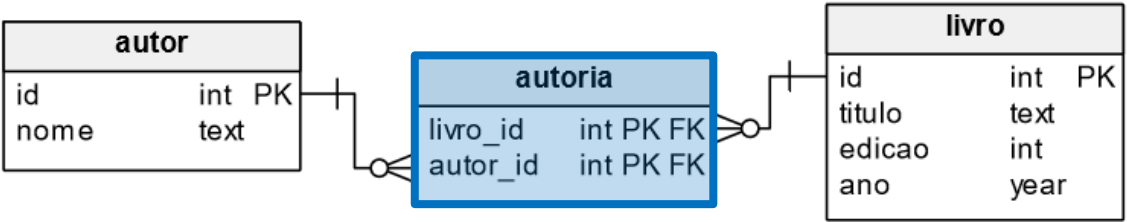
Esquema diagramático

- Observe que em nenhum momento foi apresentado uma associação do tipo **muitos para muitos** entre as tabelas referenciada e referenciadora.
- Suponha um banco de dados de uma livraria onde são armazenados os dados dos autores e os livros publicados por eles.

autor

id	nome
1	R. Elmasri
2	S. B. Navathe
3	C. A. Heuser

Apesar de não ser possível implementar diretamente uma associação do tipo **muito para muitos** em um banco de dados relacional, sua implementação é realizada através do uso de uma **tabela adicional**.



livro

id	titulo	edicao	ano
1	Sistemas de banco de dados	7	2019
2	Banco de dados relacional	1	2019
3	Projeto de banco de dados	6	2008

autoria

livro_id	autor_id
1	1
1	2
2	3
3	3



Esquema do banco de dados

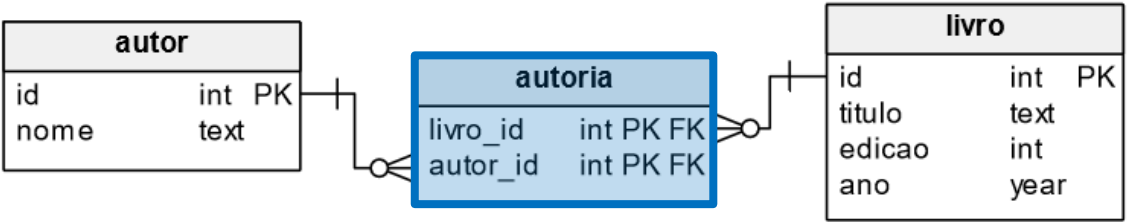
Esquema diagramático

- Observe que em nenhum momento foi apresentado uma associação do tipo **muitos para muitos** entre as tabelas referenciada e referenciadora.
- Suponha um banco de dados de uma livraria onde são armazenados os dados dos autores e os livros publicados por eles.

autor

id	nome
1	R. Elmasri
2	S. B. Navathe
3	C. A. Heuser

Apesar de não ser possível implementar diretamente uma associação do tipo **muito para muitos** em um banco de dados relacional, sua implementação é realizada através do uso de uma **tabela adicional**.



livro

id	titulo	edicao	ano
1	Sistemas de banco de dados	7	2019
2	Banco de dados relacional	1	2019
3	Projeto de banco de dados	6	2008

autoria

livro_id	autor_id
1	1
1	2
2	3
3	3



Esquema do banco de dados

Esquema diagramático

- Observe que em nenhum momento foi apresentado uma associação do tipo **muitos para muitos** entre as tabelas referenciada e referenciadora.
 - Cada campo de uma tabela em um banco de dados relacional possui valores **atômicos**, ou seja, não podem existir campos que assumam mais de um valor simultaneamente.
 - Desta forma, não se pode ter associações do tipo muitos para muitos de forma direta em um banco de dados relacional
- Apesar de não ser possível implementar diretamente uma associação do tipo **muito para muitos** em um banco de dados relacional, sua implementação é realizada através do uso de uma tabela adicional.

Vantagens de um banco de dados relacional

- O modelo relacional é simples e eficiente.
- É usado por organizações de todos os tipos e tamanhos para uma ampla variedade de necessidades de informações.
 - Os bancos de dados relacionais são usados para rastrear inventários, processar transações de comércio eletrônico, gerenciar grandes quantidades de informações essenciais sobre clientes e muito mais.
- Um banco de dados relacional pode ser considerado para qualquer necessidade de informações na qual os pontos de dados se relacionam entre si e devem ser gerenciados de maneira segura e consistente, com base em regras.

Vantagens de um banco de dados relacional

Consistência de dados

- O modelo relacional é o melhor em manter a consistência de dados entre aplicativos e cópias de banco de dados.
 - Por exemplo, quando um cliente deposita dinheiro em um caixa eletrônico e analisa o saldo da conta em um celular, o cliente espera ver esse depósito refletido imediatamente em um saldo atualizado da conta.
 - Os bancos de dados relacionais se destacam nesse tipo de consistência de dados, garantindo que várias cópias de um banco de dados tenham os mesmos dados o tempo todo.
- É difícil para outros tipos de bancos de dados manter esse nível de consistência oportunamente com grandes quantidades de dados.
 - Alguns bancos de dados recentes, como os bancos de dados NoSQL, podem fornecer somente “consistência eventual”.
- **Consistência eventual** é quando as leituras e gravações não estão alinhadas no mesmo dado. A maioria dos sistemas NoSQL é eventualmente consistente, no sentido de que eles garantem que nenhuma atualização seja realizada para um determinado objeto por um período de tempo especificado e uma consulta retorna o que o último comando redigiu.
 - Sob esse princípio, quando o banco de dados é dimensionado ou quando vários usuários acessam os mesmos dados ao mesmo tempo, os dados precisam de algum tempo para “serem atualizados”.
- A consistência eventual é aceitável para alguns casos, como manter listagens em um catálogo de produtos, mas para operações comerciais essenciais, como transações de compra, o banco de dados relacional ainda é o padrão.

Vantagens de um banco de dados relacional

Compromisso e atomicidade

- Os bancos de dados relacionais lidam com regras e políticas de negócios em um nível muito granular, com políticas rígidas sobre **compromisso**.
 - O “compromisso” é a realização de uma alteração de forma permanente no banco de dados.
 - Por exemplo, considere um banco de dados de inventário que rastreie três peças que são sempre usadas juntas. Quando uma peça é retirada do inventário, as outras duas também devem ser retiradas. Se uma das três peças não estiver disponível, nenhuma das peças deve ser retirada. Todas as peças devem estar disponíveis antes que o banco de dados confirme. Um banco de dados relacional não assumirá o compromisso por uma peça até que saiba que pode confirmar todas as três.
- Essa capacidade de compromisso é chamada de **atomicidade**.
 - A atomicidade é a chave para manter os dados e garantir que eles estejam em conformidade com regras, regulamentos e políticas da empresa.

Saiba mais!

Livros, artigos, cursos, páginas, documentos...

- CODD, Edgar F. A relational model of data for large shared data banks. Communications of the ACM, v.13 (6), 377–387. 1970.
Disponível em: <https://doi.org/10.1145/362384.362685>
- ORACLE BRASIL. O que é um banco de dados relacional?
Disponível em: <https://www.oracle.com/br/database/what-is-a-relational-database/>
Acesso em: 21 de out. de 2020.
- ESPOSITO, Dino. Documentos, bancos de dados e consistência eventual. 2015
Disponível em: <https://docs.microsoft.com/pt-br/archive/msdn-magazine/2014/august/cutting-edge-documents-databases-and-eventual-consistency>
Acesso em: 21 de out. de 2020.

Dúvidas?

André L. Maravilha

andre.maravilha@cefetmg.br
<https://andremaravilha.github.io/>

