

Relações

Alexandre Checoli Choueiri

02/06/2024

- ① Motivação
- ② Relacionamentos entre tabelas
- ③ Diagramas Entidade Relacionamento
- ④ Implementação - chaves
- ⑤ Conclusões
- ⑥ JOIN
- ⑦ Conjuntos

Motivação

Até o momento, realizamos buscas em bancos de dados com uma **única tabela**. Usualmente os bancos de dados possuem muitas tabelas, que se relacionam entre si. Vamos ver um exemplo considerando os próprios dados de [livros](#).

Motivação

Autor		
Nome	Período	Nacionalidade
Dostoievski	1800	Rússia
Nietzsche	1800	Alemanha
Homero	800 A.C	Grécia

Livro	
Nome	Editora
Crime e castigo	Ed. 51
O anticristo	Martin Claret
Ilíada	Penguin

Motivação

Autor			Livro	
Nome	Período	Nacionalidade	Nome	Editora
Dostoievski	1800	Rússia	Crime e castigo	Ed. 51
Nietzsche	1800	Alemanha	O anticristo	Martin Claret
Homero	800 A.C	Grécia	Ilíada	Penguin

Considere que temos as duas tabelas acima, uma relacionada a autores e outra a livros.
Como podemos saber qual autor escreveu qual livro?

Motivação

Autor			Livro	
Nome	Período	Nacionalidade	Nome	Editora
Dostoievski	1800	Rússia	Crime e castigo	Ed. 51
Nietzsche	1800	Alemanha	O anticristo	Martin Claret
Homero	800 A.C	Grécia	Ilíada	Penguin

Uma forma de contornar o problema é usar o sistema da "honestidade". Nós vamos convencionar que o primeiro elemento da tabela de **Autor** sempre vai corresponder ao primeiro elemento da tabela **Livro**.

Motivação

Autor			Livro	
Nome	Período	Nacionalidade	Nome	Editora
Dostoievski	1800	Rússia	Crime e castigo	Ed. 51
Nietzsche	1800	Alemanha	O anticristo	Martin Claret
Homero	800 A.C	Grécia	Ilíada	Penguin

Temos então que:

Motivação

Autor			Livro	
Nome	Período	Nacionalidade	Nome	Editora
Dostoievski	1800	Rússia	Crime e castigo	Ed. 51
Nietzsche	1800	Alemanha	O anticristo	Martin Claret
Homero	800 A.C	Grécia	Ilíada	Penguin

Temos então que:

1. Dostoievski $\xrightarrow{\text{escreveu}}$ Crime e castigo

Motivação

Autor			Livro	
Nome	Período	Nacionalidade	Nome	Editora
Dostoievski	1800	Rússia	Crime e castigo	Ed. 51
Nietzsche	1800	Alemanha	O anticristo	Martin Claret
Homero	800 A.C	Grécia	Ilíada	Penguin

Temos então que:

1. Dostoievski ^{escreveu} → Crime e castigo

E assim podemos convencionar com a **segunda e terceiras linhas também**:

Motivação

Autor		
Nome	Período	Nacionalidade
Dostoievski	1800	Rússia
Nietzsche	1800	Alemanha
Homero	800 A.C	Grécia

Livro	
Nome	Editora
Crime e castigo	Ed. 51
O anticristo	Martin Claret
Ilíada	Penguin

1. Dostoievski $\xrightarrow{\text{escreveu}}$ Crime e castigo
2. Nietzsche $\xrightarrow{\text{escreveu}}$ O anticristo
3. Homero $\xrightarrow{\text{escreveu}}$ Ilíada

Motivação

Autor			Livro	
Nome	Período	Nacionalidade	Nome	Editora
Dostoievski	1800	Rússia	Crime e castigo	Ed. 51
Nietzsche	1800	Alemanha	O anticristo	Martin Claret
Homero	800 A.C	Grécia	Ilíada	Penguin

Essa solução, no entanto, pode gerar uma série de problemas:

Motivação

Autor			Livro	
Nome	Período	Nacionalidade	Nome	Editora
Dostoievski	1800	Rússia	Crime e castigo	Ed. 51
Nietzsche	1800	Alemanha	O anticristo	Martin Claret
Homero	800 A.C	Grécia	Ilíada	Penguin

Essa solução, no entanto, pode gerar uma série de problemas:

1. Os dados devem ficar **sempre na mesma ordem**
2. Algum **erro de inserção** em uma tabela pode produzir um erro em muitos elementos

Motivação

Autor_Livro				
Nome	Período	Nacionalidade	Nome	Editora
Dostoievski	1800	Rússia	Crime e castigo	Ed. 51
Nietzsche	1800	Alemanha	O anticristo	Martin Claret
Homero	800 A.C	Grécia	Ilíada	Penguin

Uma outra possibilidade para resolver o problema é simplesmente **juntar as tabelas**.

Motivação

Autor_Livro				
Nome	Período	Nacionalidade	Nome	Editora
Dostoievski	1800	Rússia	Crime e castigo	Ed. 51
Nietzsche	1800	Alemanha	O anticristo	Martin Claret
Homero	800 A.C	Grécia	Ilíada	Penguin

Uma outra possibilidade para resolver o problema é simplesmente **juntar as tabelas**.

Quais problemas essa abordagem pode gerar?

Autor_Livro

Nome	Período	Nacionalidade	Nome	Editora
Dostoievski	1800	Rússia	Crime e castigo	Ed. 51
Dostoievski	1800	Rússia	Irmãos Karamázov	Ed. 51
Nietzsche	1800	Alemanha	O anticristo	Martin Claret
Homero	800 A.C	Grécia	Ilíada	Penguin

Sabemos que Dostoievski escreveu mais de um livro, como **Os Irmão Karamázov**.

Motivação

Autor_Livro				
Nome	Período	Nacionalidade	Nome	Editora
Dostoievski	1800	Rússia	Crime e castigo	Ed. 51
Dostoievski	1800	Rússia	Irmãos Karamázov	Ed. 51
Nietzsche	1800	Alemanha	O anticristo	Martin Claret
Homero	800 A.C	Grécia	Ilíada	Penguin

Sabemos que Dostoievski escreveu mais de um livro, como **Os Irmão Karamázov**.
Essa solução gera muitas redundâncias nos dados. Para todo livro de Dostoievski no banco de dados, termos uma repetição de seu Período e Nacionalidade.

Isso nos indica que quando trabalhamos com mais de uma tabela em um banco de dados, **é necessário pensar nas relações que elas desempenham entre si**. Vamos então entender como fazer isso.

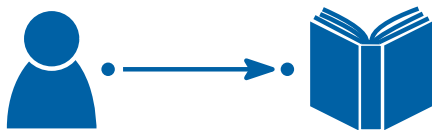
Relacionamentos entre tabelas

Relacionamentos entre tabelas



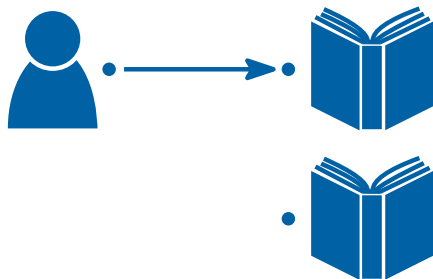
Antes de entendermos como podemos operar tecnicamente as relações entre as duas tabelas, vamos entender quais são as relações possíveis entre as tabelas **Autor** e **Livro**.

Relacionamentos entre tabelas



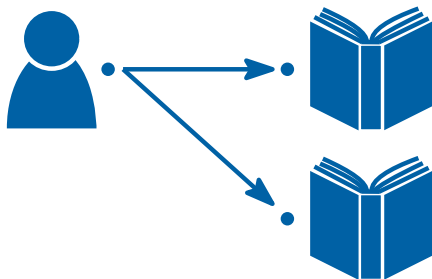
De forma simplificada, podemos pensar que um autor escreve um livro. E de forma similar, um livro é escrito por um autor. Essa relação é chamada de **1 para 1** (1-1).

Relacionamentos entre tabelas



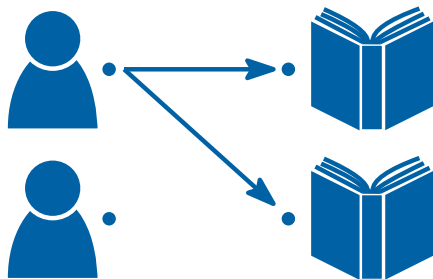
No entanto, vimos no exemplo que é possível que um autor escreva mais de um livro, como Dostoievski.

Relacionamentos entre tabelas



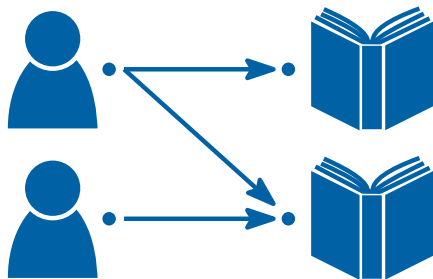
O que gera um segundo tipo de relacionamento: **1 para muitos** (1-n), um autor pode escrever muitos livros.

Relacionamentos entre tabelas



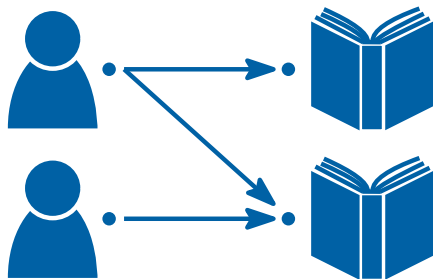
Ainda, sabemos que alguns livros possuem mais de um autor (livros técnicos, por exemplo).

Relacionamentos entre tabelas



O que gera um novo tipo de relacionamento, chamado de **muitos para muitos** (n-n).

Relacionamentos entre tabelas



Esses são todos os relacionamentos que podem existir entre as tabelas de um banco de dados:

- Um para um
- Um para muitos
- Muitos para muitos

Relacionamentos entre tabelas

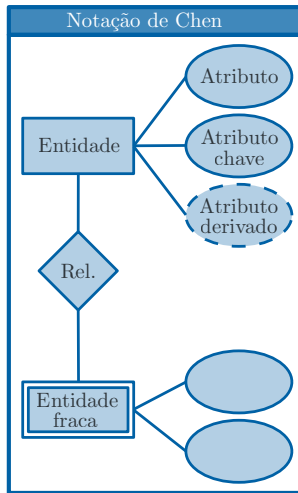
Para criar um novo banco de dados, ou mesmo entender um já existente, é primordial conhecer como as tabelas se relacionam. Por sorte, existem ferramentas visuais que facilitam essa tarefa, uma delas é o **Diagrama Entidade Relacionamento** (Entity Relationship Diagrams - ER Diagrams).

Diagramas Entidade Relacionamento

Diagramas Entidade Relacionamento

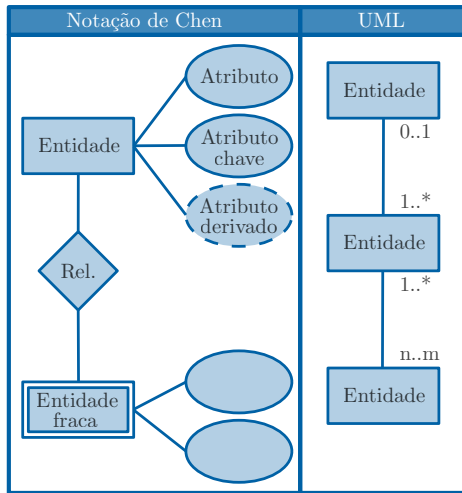
Um **diagrama entidade relacionamento** representa visualmente como as tabelas e colunas de um banco de dados se relacionam. Existem muitas representações possíveis para o diagrama ER, algumas delas são:

Diagramas Entidade Relacionamento



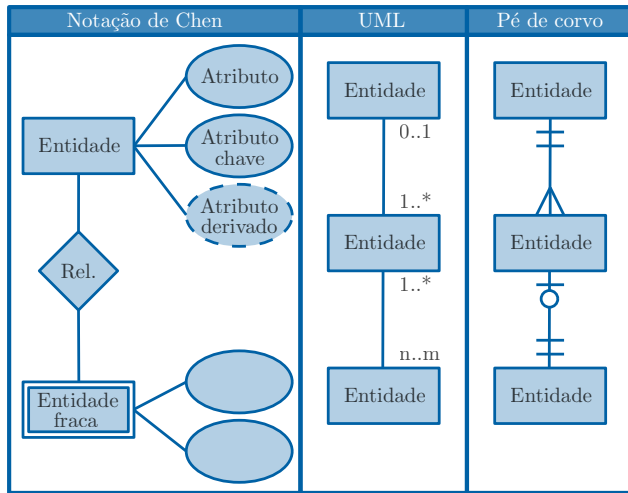
Diagramas pela **notação de Chen**.

Diagramas Entidade Relacionamento



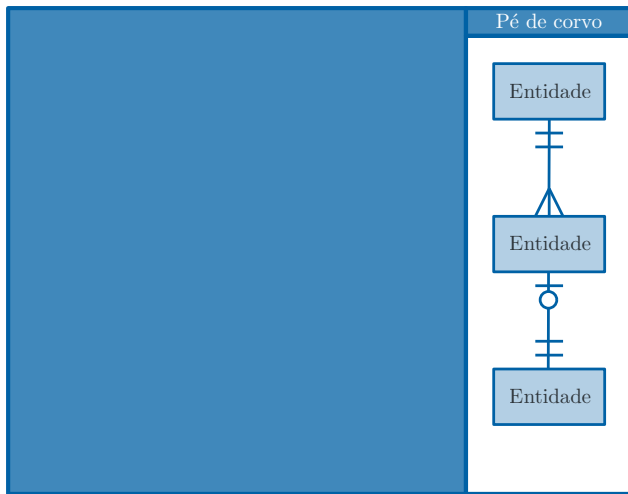
Diagramas pela **notação UML**.

Diagramas Entidade Relacionamento



Diagramas pela **notação pés de corvo - Crow's foot**.

Diagramas Entidade Relacionamento






Por ser uma das mais utilizadas, estudaremos a notação de pés de corvo.

Diagramas Entidade Relacionamento






O diagrama ao lado é um exemplo de ER com a notação de pés de corvo. Cada retângulo indica uma **entidade**, ou seja, uma **tabela** em nosso banco de dados. Além disso, existem arcos ligando as tabelas, com uma notação específica, esse arcos indicam os relacionamentos entre as tabelas. **Vamos entender o que cada um representa.**

Diagramas Entidade Relacionamento

	Zero
	Um
	Muitos




A linha com um círculo branco pode ser entendida como um 0. Ou seja, **nada precisa estar relacionado a esta entidade.**

Diagramas Entidade Relacionamento

	Zero
	Um
	Muitos

A linha com uma barra indica que a entidade precisa ter **pelo menos um** relacionamento com outra tabela.

Diagramas Entidade Relacionamento

	Zero
	Um
	Muitos

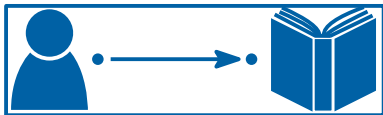
Finalmente, a linha com 3 bifurcações (que parece um pé de corvo) significa que a **entidade está relacionada de muitas formas** em uma outra tabela.

Diagramas Entidade Relacionamento

Vamos criar um diagrama entidade relacionamento para a relação entre autores e livros.

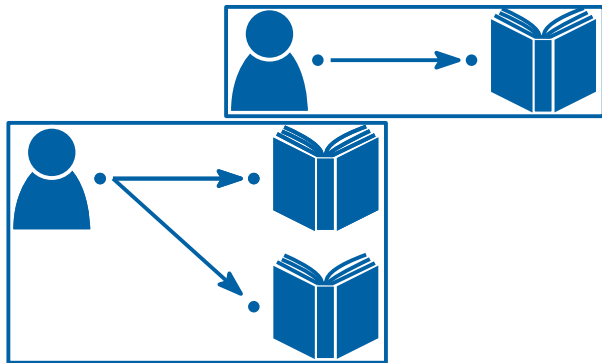
Lembrando das possibilidades já discutidas:

Diagramas Entidade Relacionamento



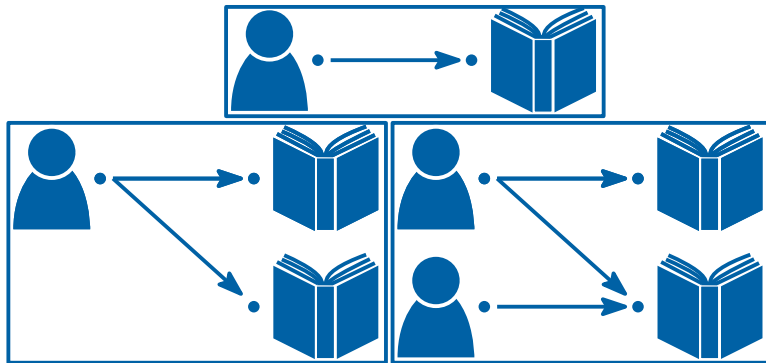
Um autor poderia escrever um livro e somente um livro (e um livro seria escrito por somente um autor)

Diagramas Entidade Relacionamento



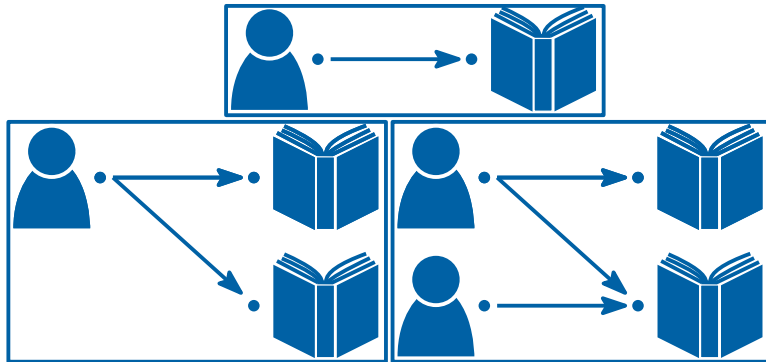
Um autor poderia escrever mais de um livro (porém cada livro ainda seria escrito por um autor)

Diagramas Entidade Relacionamento



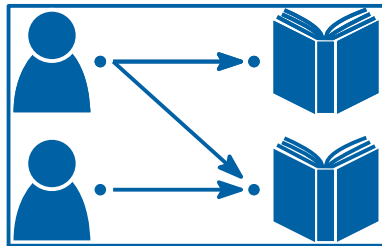
Finalmente, um autor poderia escrever muitos livros, e cada livro poderia ser escrito por muitos autores.

Diagramas Entidade Relacionamento






Note que cada situação descreve **relacionamentos diferentes**, e portanto geram diagramas distintos.

Diagramas Entidade Relacionamento



Vamos representar a última situação, em que **um livro pode ser escrito por muitos autores e um autor pode escrever muitos livros.**

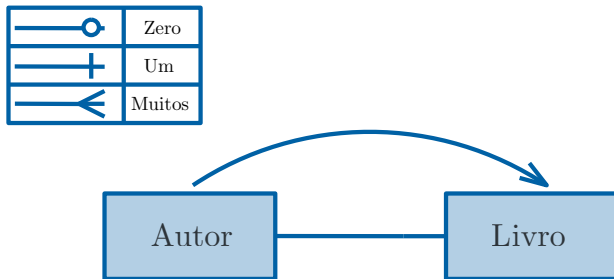
Diagramas Entidade Relacionamento

	Zero
	Um
	Muitos



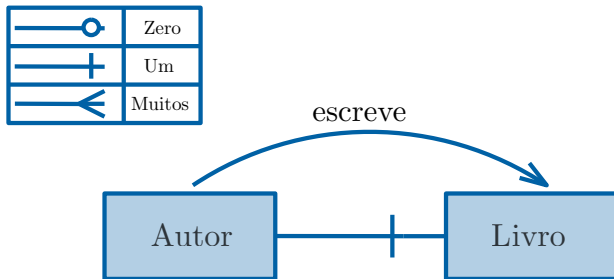
O primeiro passo é representar as tabelas que vamos relacionar como retângulos, e uma linha unindo-as.

Diagramas Entidade Relacionamento



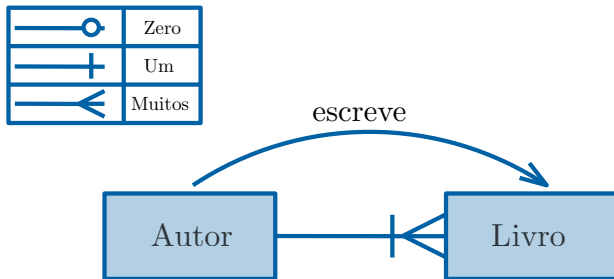
Agora podemos estabelecer a primeira relação, começando de **Autor** para **Livros**. Sempre pensamos na relação em duas possibilidades: o mínimo e o máximo.

Diagramas Entidade Relacionamento



Para facilitar, podemos adicionar um verbo indicando a relação: **Autor** escreve no mínimo 1 **Livro**.

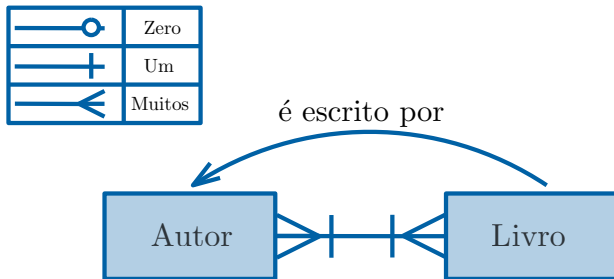
Diagramas Entidade Relacionamento



Mas também pode escrever muitos livros.

OBS: Note que a ordem de leitura de um par de tabelas começa pelo nome da primeira e termina no início da segunda: *um autor pode escrever 1 ou muitos livros*

Diagramas Entidade Relacionamento



Mas ainda precisamos modelar a relação de livros para autores: um livro deve ser escrito por pelo menos 1 autor, mas pode ser escrito por muitos.

Diagramas Entidade Relacionamento

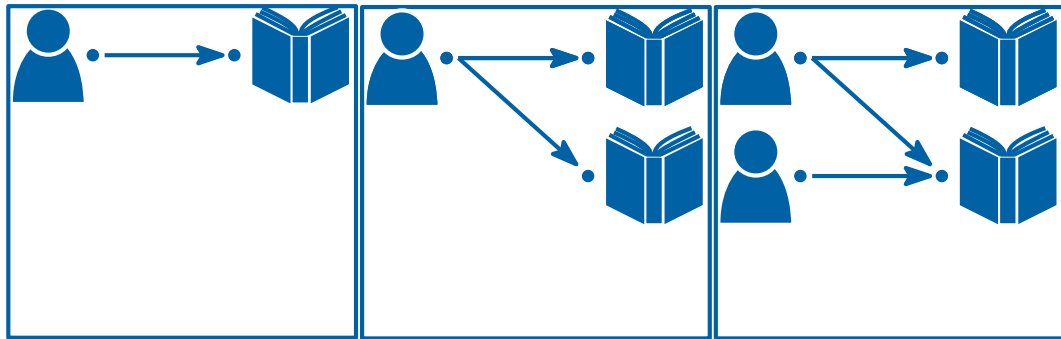
	Zero
	Um
	Muitos



O diagrama ER que modela a relação entre a tabela Autor e Livro fica da seguinte forma então.

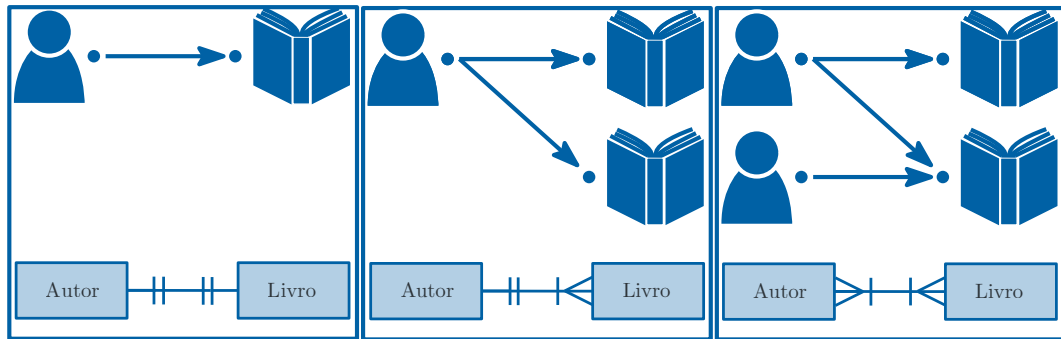
Diagramas Entidade Relacionamento

EXERCÍCIO: Crie os diagramas ER para modelar as situações abaixo, envolvendo as mesmas tabelas (**Autor** e **Livro**).



Diagramas Entidade Relacionamento

EXERCÍCIO: Crie os diagramas ER para modelar as situações abaixo, envolvendo as mesmas tabelas (**Autor** e **Livro**).



Diagramas Entidade Relacionamento

Usamos os diagramas ER em duas situações:

1. Quando estamos criando um banco de dados do 0, para montar as corretas relações entre as tabelas e deixar o banco de dados otimizado.
2. Quando começamos a realizar buscas em um BD, para entendermos melhor as relações existentes nas tabelas.

Diagramas Entidade Relacionamento

EXERCÍCIO: Faça a modelagem completa do problema 1 em: [link da página no site](#)

Implementação - chaves

Implementação - chaves

Mas como podemos realizar a **implementação** dessas relações?

- Um para um
- Um para muitos
- Muitos para muitos

Para isso precisamos entender o conceito de **chaves** em bancos de dados.

Chave

A chave de uma tabela (**relação**) é um atributo (coluna) que define unicamente os elementos de cada linha da mesma.

Implementação - chaves

Ou seja, para cada tabela deve existir uma coluna com um valor identifique toda linha sem duplicidade. Vamos considerar a nossa tabela de **Livro** como exemplo:

Livro	
Nome	Editora
Crime e castigo	Ed. 51
O anticristo	Martin Claret
Ilíada	Penguin

Implementação - chaves

Ou seja, para cada tabela deve existir uma coluna com um valor identifique toda linha sem duplicidade. Vamos considerar a nossa tabela de **Livro** como exemplo:

Livro	
Nome	Editora
Crime e castigo	Ed. 51
O anticristo	Martin Claret
Ilíada	Penguin

Existe algum atributo que possa ser considerado como a nossa **chave** da tabela?

Implementação - chaves

Da forma como o banco está definido, **não**. Podemos ter livros iguais sendo publicados por editoras diferentes, e editoras diferentes lançando o mesmo livro.

Implementação - chaves

Da forma como o banco está definido, **não**. Podemos ter livros iguais sendo publicados por editoras diferentes, e editoras diferentes lançando o mesmo livro.

No entanto, todo livro possui um **ISBN** (International **S**tandard **B**ook **N**umber), que identifica unicamente o livro. Desta forma podemos usar o ISBN como o atributo chave da nossa tabela **Livro**:

Livro		
ISBN	Nome	Editora
976-85-333-0865-0	Crime e castigo	Ed. 51
778-84-332-0765-0	O anticristo	Martin Claret
973-85-333-0765-0	Ilíada	Penguin

O ISBN é como um CPF para livros.

Implementação - chaves

Agora que entendemos **o que é uma chave**, podemos fazer uma simplificação. Ao usar o ISBN, cada string ocupa 17 bytes de espaço (1 para cada caractere). Com muitos registros no banco de dados, isso ocupa muito espaço computacional, o que podemos fazer para **otimizar a representação desta chave**?

Implementação - chaves

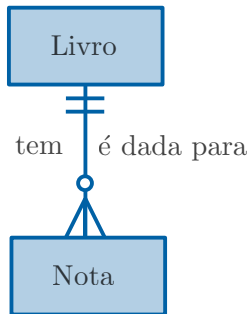
Agora que entendemos **o que é uma chave**, podemos fazer uma simplificação. Ao usar o ISBN, cada string ocupa 17 bytes de espaço (1 para cada caractere). Com muitos registros no banco de dados, isso ocupa muito espaço computacional, o que podemos fazer para **otimizar a representação desta chave**?

Podemos simplesmente usar um número sequencial para cada livro do banco de dados, de forma que nunca os números são repetidos (**chamado de id**).

Livro		
id	Nome	Editora
0	Crime e castigo	Ed. 51
1	O anticristo	Martin Claret
2	Ilíada	Penguin

Implementação - chaves

Usamos as **chaves** das tabelas para criar os relacionamentos. Considere a relação existente entre as tabelas **Livro** e **Nota** (relação um-para-muitos).



Essa é uma relação do tipo um-para-muitos: um livro pode ser avaliado várias vezes. Podemos implementar essa relação exportando a **chave primária** da tabela livro para a tabela **nota**.

Implementação - chaves

Considere as duas tabelas que queremos relacionar.

Livro			Nota	
id	Nome	Editora	id	Nota
0	Crime e castigo	Ed. 51	0	3
1	O anticristo	Martin Claret	1	2
2	Ilíada	Penguin	2	2
3	Ilíada	Nova fronteira	3	4
			4	3

Implementação - chaves

Considere as duas tabelas que queremos relacionar.

Livro			Nota	
id	Nome	Editora	id	Nota
0	Crime e castigo	Ed. 51	0	3
1	O anticristo	Martin Claret	1	2
2	Ilíada	Penguin	2	2
3	Ilíada	Nova fronteira	3	4
			4	3

A **chave primária** de uma tabela é aquela coluna que unicamente identifica cada linha. No caso acima, as duas colunas de id.

Implementação - chaves

Considere as duas tabelas que queremos relacionar.

Livro			Nota		
id	Nome	Editora	id	Nota	id_livro
0	Crime e castigo	Ed. 51	0	3	2
1	O anticristo	Martin Claret	1	2	3
2	Ilíada	Penguin	2	2	3
3	Ilíada	Nova fronteira	3	4	2
			4	3	0

Criamos a relação entre as duas tabelas **exportando** a chave primária da tabela **Livro** para a tabela **Nota** (id_livro).

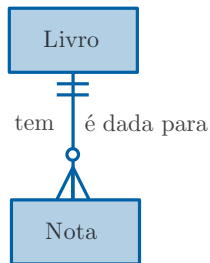
Implementação - chaves

Considere as duas tabelas que queremos relacionar.

Livro			Nota		
id	Nome	Editora	id	Nota	id_livro
0	Crime e castigo	Ed. 51	0	3	2
1	O anticristo	Martin Claret	1	2	3
2	Ilíada	Penguin	2	2	3
3	Ilíada	Nova fronteira	3	4	2
			4	3	0

Criamos a relação entre as duas tabelas **exportando** a chave primária da tabela **Livro** para a tabela **Nota** (id_livro). A chave exportada na tabela Nota tem o nome de **chave estrangeira**, pois é uma chave que veio de outra tabela, somente para fins de relacionar as mesmas.

Implementação - chaves



Livro			Nota		
id	Nome	Editora	id	Nota	id_livro
0	Crime e castigo	Ed. 51	0	3	2
1	O anticristo	Martin Claret	1	2	3
2	Ilíada	Penguin	2	2	3
3	Ilíada	Nova fronteira	3	4	2
			4	3	0

Note que a relação do diagrama ER (**um-para-muitos**) é implementada **completamente pela uso das chaves**.

Implementação - chaves

Novamente, lembrando dos relacionamentos:

- Um para um
- **Um para muitos**
- Muitos para muitos

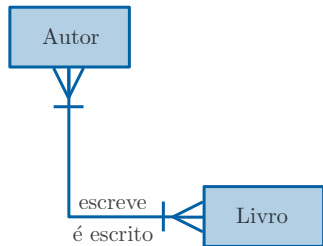
Conseguimos implementar a relação um-para-muitos. Como poderíamos implementar a relação **muitos-para-muitos**?

Implementação - chaves

Novamente, lembrando dos relacionamentos:

- Um para um
- **Um para muitos**
- Muitos para muitos

Conseguimos implementar a relação um-para-muitos. Como poderíamos implementar a relação **muitos-para-muitos**? Considere a relação muitos-para-muitos entre as tabelas Autor e Livro:



Implementação - chaves

Bem como os dados de cada tabela como abaixo:

Autor			
id	Nome	Período	Nacionalidade
1	Dostoievski	1800	Rússia
2	Nietzsche	1800	Alemanha
3	Homero	800 A.C	Grécia

Livro		
id	Nome	Editora
1	Crime e castigo	Ed. 51
2	O anticristo	Martin Claret
3	Ilíada	Penguin

Implementação - chaves

Bem como os dados de cada tabela como abaixo:

Autor			
id	Nome	Período	Nacionalidade
1	Dostoievski	1800	Rússia
2	Nietzsche	1800	Alemanha
3	Homero	800 A.C	Grécia

Livro		
id	Nome	Editora
1	Crime e castigo	Ed. 51
2	O anticristo	Martin Claret
3	Ilíada	Penguin

Como podemos criar a relação **muitos-para-muitos** usando as chaves?

Implementação - chaves

Bem como os dados de cada tabela como abaixo:

Autor			
id	Nome	Período	Nacionalidade
1	Dostoievski	1800	Rússia
2	Nietzsche	1800	Alemanha
3	Homero	800 A.C	Grécia

Livro		
id	Nome	Editora
1	Crime e castigo	Ed. 51
2	O anticristo	Martin Claret
3	Ilíada	Penguin

Nesse caso, para implementar a relação precisamos **criar uma nova tabela** que contém duas chaves estrangeiras, uma de cada tabela da relação muitos-para-muitos. Essa nova tabela é chamada de tabela associativa, de junção, etc...(associative, joint table).

Implementação - chaves

Bem como os dados de cada tabela como abaixo:

Autor

id	Nome	Período	Nacionalidade
1	Dostoievski	1800	Rússia
2	Nietzsche	1800	Alemanha
3	Homero	800 A.C	Grécia

Autor_Livro

id	id_autor	id_livro
1	1	1
2	2	2
3	3	3

Livro

id	Nome	Editora
1	Crime e castigo	Ed. 51
2	O anticristo	Martin Claret
3	Ilíada	Penguin

Implementação - chaves

Aprenderemos o relacionamento um-para-um um pouco mais a frente no curso.

- Um para um
- **Um para muitos**
- **Muitos para muitos**

Conclusões

Conclusões

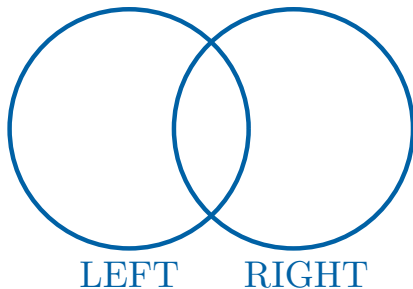
1. O diagrama ER nos fornece uma maneira de modelar como o banco de dados deve se comportar (ou usamos um diagrama já feito para entender como um banco funciona).
2. A **implementação** do banco de dados, como mostrado na modelagem de um ER é feita por meio de **relações** entre as tabelas.
3. As relações são implementadas por meio de chaves: a chave primária de uma tabela é a coluna com valores que identificam unicamente a linha.
4. Quando exportamos uma chave primária de uma tabela A para uma tabela B (criando uma relação entre elas), a mesma é chamada de **chave estrangeira** na tabela B.

(OBS: Antes de continuar, faça os exercícios sobre *subqueries* da aula ([link da aula](#)))

JOIN

JOIN

Ainda temos uma ferramenta para manipular várias tabelas, chamada de **JOIN**. O **JOIN** é usado quando queremos unir as informações de duas tabelas em uma única, para facilitar a visualização, por exemplo.



sea lions		migrations		
id	name	id	distance	days
10484	Ayah	10484	1000	107
11728	Spot	11728	1531	56
11729	Tiger	11729	1370	37
11732	Mabel	11732	1622	62
11734	Rick	11734	1491	58
11790	Jolee	11735	2723	82
		11736	1571	52
		11737	1957	92

Considere um banco de dados que coleta informações sobre as distâncias percorridas por leões marinhos. Temos os nome dos leões na tabela **sea lions**, e as informações de distâncias na tabela **migrations**.

JOIN

sea lions		migrations		
id	name	id	distance	days
10484	Ayah	10484	1000	107
11728	Spot	11728	1531	56
11729	Tiger	11729	1370	37
11732	Mabel	11732	1622	62
11734	Rick	11734	1491	58
11790	Jolee	11735	2723	82
		11736	1571	52
		11737	1957	92

Note que não temos informações de migrações para todos os leões marinhos.

JOIN

sea lions JOIN migrations

id	name	id	distance	days
10484	Ayah	10484	1000	107
11728	Spot	11728	1531	56
11729	Tiger	11729	1370	37
11732	Mabel	11732	1622	62
11734	Rick	11734	1491	58

Poderíamos então estar interessados em **juntar** todos os que tem um equivalente em uma única tabela, para facilitar a visualização. Para esses tipos de operação (juntar tabelas) usamos o comando sql **JOIN**.

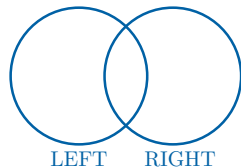
JOIN

sea lions JOIN migrations				
id	name	id	distance	days
10484	Ayah	10484	1000	107
11728	Spot	11728	1531	56
11729	Tiger	11729	1370	37
11732	Mabel	11732	1622	62
11734	Rick	11734	1491	58

Perceba também que nessa junção, removemos todos os elementos que não tem matching nas tabelas (considerando os ids). Mas poderíamos ter optado por outras possibilidades, o que gera novos tipos de **JOIN**.

JOIN

sea lions		migrations		
id	name	id	distance	days
10484	Ayah	10484	1000	107
11728	Spot	11728	1531	56
11729	Tiger	11729	1370	37
11732	Mabel	11732	1622	62
11734	Rick	11734	1491	58
11790	Jolee	11735	2723	82
		11736	1571	52
		11737	1957	92

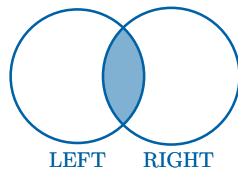


Para entender melhor os diferentes tipos de JOIN e sua nomenclatura, precisamos pensar nas duas tabelas que serão unidas como uma tabela da esquerda (**LEFT**) e uma da direita (**RIGHT**). Ainda, vamos usar a **representação de conjuntos** para criar diagramas de Venn.

JOIN

sea lions		migrations		
id	name	id	distance	days
10484	Ayah	10484	1000	107
11728	Spot	11728	1531	56
11729	Tiger	11729	1370	37
11732	Mabel	11732	1622	62
11734	Rick	11734	1491	58
11790	Jolee	11735	2723	82
		11736	1571	52
		11737	1957	92

sea lions JOIN migrations				
id	name	id	distance	days
10484	Ayah	10484	1000	107
11728	Spot	11728	1531	56
11729	Tiger	11729	1370	37
11732	Mabel	11732	1622	62
11734	Rick	11734	1491	58

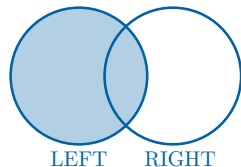


Quando selecionamos os elementos em comum entre as duas tabelas, temos um **INNER JOIN**.

JOIN

sea lions		migrations		
id	name	id	distance	days
10484	Ayah	10484	1000	107
11728	Spot	11728	1531	56
11729	Tiger	11729	1370	37
11732	Mabel	11732	1622	62
11734	Rick	11734	1491	58
11790	Jolee	11736	1571	52
		11737	1957	92

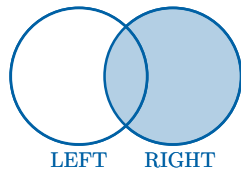
sea lions JOIN migrations				
id	name	id	distance	days
10484	Ayah	10484	1000	107
11728	Spot	11728	1531	56
11729	Tiger	11729	1370	37
11732	Mabel	11732	1622	62
11734	Rick	11734	1491	58
11790	Jolee	-	-	-



Podemos selecionar **todos** os elementos da tabela da **esquerda**: aqueles que não tem equivalentes são mantidos com valores nulos. Esse é o **LEFT JOIN**.

JOIN

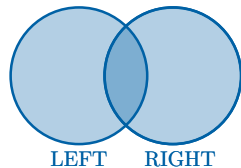
sea lions		migrations			sea lions JOIN migrations				
id	name	id	distance	days	id	name	id	distance	days
10484	Ayah	10484	1000	107	10484	Ayah	10484	1000	107
11728	Spot	11728	1531	56	11728	Spot	11728	1531	56
11729	Tiger	11729	1370	37	11729	Tiger	11729	1370	37
11732	Mabel	11732	1622	62	11732	Mabel	11732	1622	62
11734	Rick	11734	1491	58	11734	Rick	11734	1491	58
11790	Jolee	11735	2723	82	-	-	11735	2723	82
		11736	1571	52	-	-	11736	1571	52
		11737	1957	92	-	-	11737	1957	92



Da mesma forma podemos selecionar todos os elementos da direita, mantendo os sem equivalentes na outra tabela com valores nulos. Esse é o **RIGHT JOIN**.

JOIN

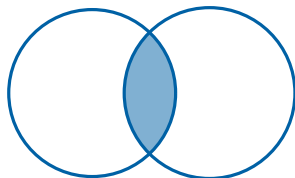
sea lions		migrations			sea lions JOIN migrations				
		id	distance	days	id	name	id	distance	days
id	name								
10484	Ayah	10484	1000	107	10484	Ayah	10484	1000	107
11728	Spot	11728	1531	56	11728	Spot	11728	1531	56
11729	Tiger	11729	1370	37	11729	Tiger	11729	1370	37
11732	Mabel	11732	1622	62	11732	Mabel	11732	1622	62
11734	Rick	11734	1491	58	11734	Rick	11734	1491	58
11790	Jolee	11735	2723	82	11790	Jolee	-	-	-
		11736	1571	52	-	-	11735	2723	82
		11737	1957	92	-	-	11736	1571	52
					-	-	11737	1957	92



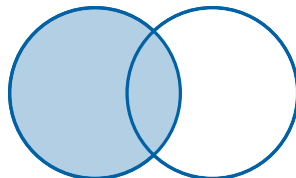
Finalmente, ao selecionarmos todos os elementos de ambas as tabelas, mantendo valores sem equivalentes, temos um **FULL JOIN**.

JOIN

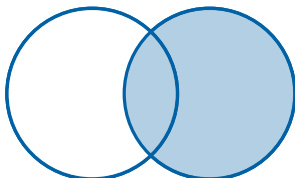
Resumo



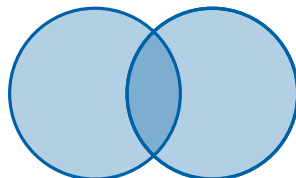
INNER JOIN



LEFT JOIN



RIGHT JOIN



FULL JOIN

Conjuntos

Temos ainda a possibilidade relacionar duas tabelas, com base nos valores de suas colunas e em operações de conjuntos. Considere o banco de dados de livros novamente, e as duas tabelas "Autor" e "Tradutor".

authors				translator	
id	name	country	birth	id	name
1	Adania Shibli	Palestine	1974	1	Adrian Nathan West
2	Ahmed Saadawi	Iraq	1973	2	Alison L. Strayer
3	Alia Trabucco Zerán	Chile	1983	3	Ahmed Saadawi
...

Temos ainda a possibilidade relacionar duas tabelas, com base nos valores de suas colunas e em operações de conjuntos. Considere o banco de dados de livros novamente, e as duas tabelas "Autor" e "Tradutor".

authors				translator	
id	name	country	birth	id	name
1	Adania Shibli	Palestine	1974	1	Adrian Nathan West
2	Ahmed Saadawi	Iraq	1973	2	Alison L. Strayer
3	Alia Trabucco Zerán	Chile	1983	3	Ahmed Saadawi
...

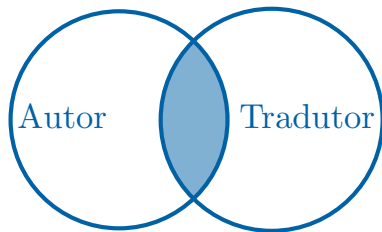
Note que existem pessoas que são tanto Autores quanto Tradutores. Podemos explorar essas relações por meio das **operações entre conjuntos**.

Temos ainda a possibilidade relacionar duas tabelas, com base nos valores de suas colunas e em operações de conjuntos. Considere o banco de dados de livros novamente, e as duas tabelas "Autor" e "Tradutor".

authors				translator	
id	name	country	birth	id	name
1	Adania Shibli	Palestine	1974	1	Adrian Nathan West
2	Ahmed Saadawi	Iraq	1973	2	Alison L. Strayer
3	Alia Trabucco Zerán	Chile	1983	3	Ahmed Saadawi
...

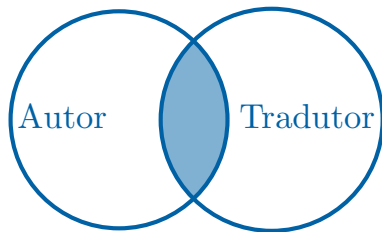
Note que existem pessoas que são tanto Autores quanto Tradutores. Podemos explorar essas relações por meio das **operações entre conjuntos**.

SETS



Autor INTERSECT Tradutor

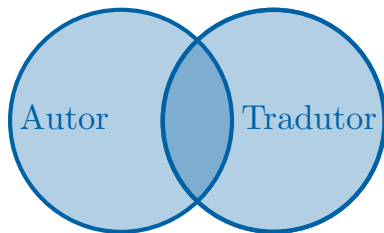
O que o seguinte conjunto representa:



Autor INTERSECT Tradutor

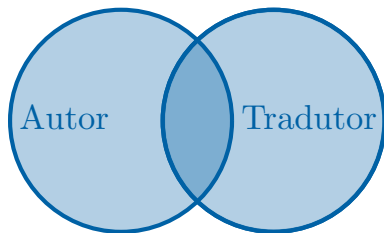
O que o seguinte conjunto representa: Pessoas que são Autoras e Tradutoras

SETS



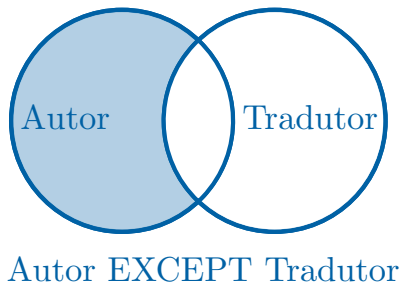
Autor UNION Tradutor

O que o seguinte conjunto representa:

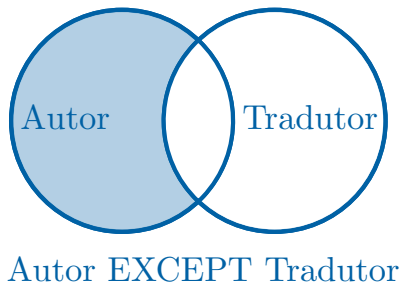


Autor UNION Tradutor

O que o seguinte conjunto representa: Pessoas que são Autoras ou Tradutoras

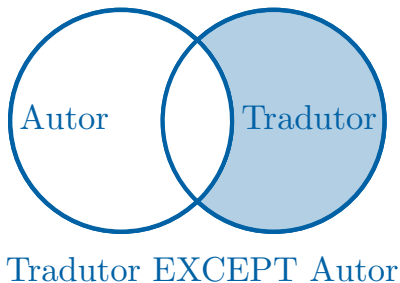


O que o seguinte conjunto representa:

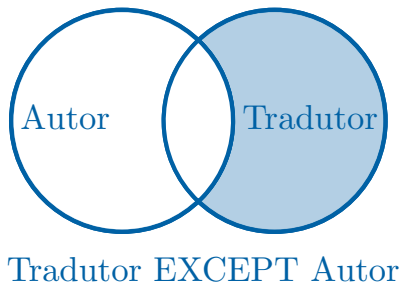


O que o seguinte conjunto representa: Pessoas que são **somente** Autoras

SETS

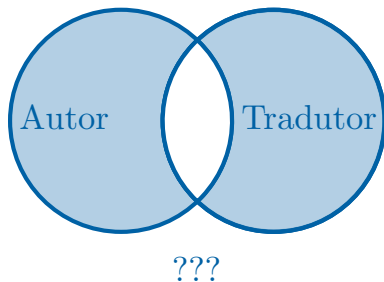


O que o seguinte conjunto representa:



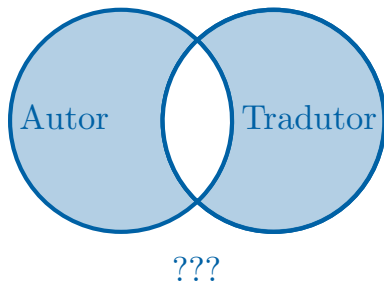
O que o seguinte conjunto representa: Pessoas que são **somente** Tradutoras

SETS



O que o seguinte conjunto representa:

SETS



O que o seguinte conjunto representa: Pessoas que são somente Autoras ou somente Tradutoras