

SQL

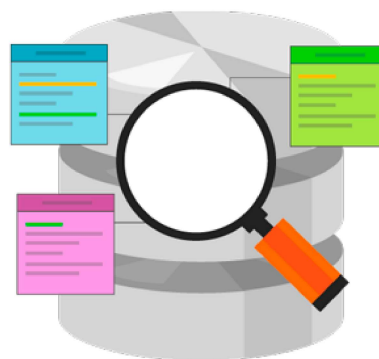
Junção horizontal e vertical de dados

Você sabia que seu material didático é interativo e multimídia? Isso significa que você pode interagir com o conteúdo de diversas formas, a qualquer hora e lugar. Na versão impressa, porém, alguns conteúdos interativos ficam desabilitados. Por essa razão, fique atento: sempre que possível, opte pela versão digital. Bons estudos!

Em grande parte dos casos nos bancos de dados, as tabelas são relacionadas, permitindo dessa forma, que possamos efetuar maior número de consultas, consequentemente podendo abstrair um maior número de informações. Para efetuar consultas em duas ou mais tabelas relacionadas, há a necessidade de técnicas que proporcionem junções, essas técnicas estão presentes na linguagem de programação de banco de dados SQL, por meio de três sintaxes: INNER JOIN, LEFT JOIN, RIGHT JOIN.

Junções

Já vimos as formas de seleções de tuplas em uma única tabela, porém essas técnicas utilizadas até o momento não permitem realizar consultas em múltiplas tabelas. Portanto, vamos ver como utilizar o comando JOIN por meio do SELECT para unir duas ou mais tabelas.



Fonte: Shutterstock.

A seguir, veja o exemplo das tabelas Categoria e Produto:

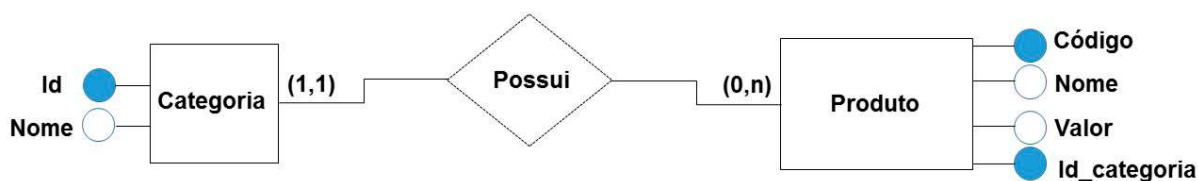
Tabelas e chaves

Na tabela “Categoria” a coluna “Id” é a chave primária.

Já na tabela “Produto” a chave primária é a coluna “Codigo”.

A chave estrangeira que relaciona as duas tabelas é o campo “Id_Categoria”.

DER de exemplo das junções



Fonte: elaborada pelo autor, captura de tela do software BrModelo.

Criação das tabelas

Script em SQL para a representação do DER de exemplo das junções.

```

1 CREATE DATABASE Loja;
2 USE Loja;
3
4 CREATE TABLE Categoria (
5   Id INT(3) PRIMARY KEY AUTO_INCREMENT,
6   Nome VARCHAR(50) NOT NULL
7 );
8 CREATE TABLE Produto (
9   Codigo INT(3) PRIMARY KEY AUTO_INCREMENT,
10  Nome VARCHAR(50) NOT NULL,
11  Valor DECIMAL(6,2) NOT NULL,
12  Id_Categoria INT(3) NOT NULL,
13  FOREIGN KEY (Id_Categoria) REFERENCES Categoria (Id)
14 );

```

Inserção de registros

Para que possamos efetuar as consultas relacionais adiante, é necessário inserir alguns registros em ambas as tabelas.

```

1 INSERT Categoria VALUES (0, "DVD"),
2   (0, "Livro"),
3   (0, "Informática");
4
5 INSERT Produto VALUES (0, "Código da Vinci", "39.99", 2),
6   (0, "Hancock", "89.99", 1),
7   (0, "Dario de um mago", "19.99", 2);
8   (0, "Eu sou a lenda", "39.99", 1);

```

Dessa forma, ao fazer uma simples seleção nas duas tabelas, teremos os resultados representados nas imagens a seguir.

```

mysql> SELECT * FROM Categoria;
+----+-----+
| Id | Nome  |
+----+-----+
| 1  | DVD   |
| 2  | Livro |
| 3  | Informática |
+----+-----+
3 rows in set (0.00 sec)

```

```

mysql> SELECT * FROM Produto;
+----+-----+-----+-----+
| Codigo | Nome          | Valor | Id_Categoria |
+----+-----+-----+-----+
| 1      | Código da Vinci | 39.99 | 2            |
| 2      | Hancock        | 89.99 | 1            |
| 3      | Dario de um Mago | 19.99 | 2            |
| 4      | Eu sou a lenda  | 39.99 | 1            |
+----+-----+-----+-----+
4 rows in set (0.08 sec)

```

Fonte: elaborada pelo autor, captura de tela do software MySQL.

As condições para efetuar uma junção, depende diretamente do **tipo de junção** e uma **condição de junção**, dessa forma com o SQL será possível retornar relações como resultados.



TIPO DE JUNÇÃO

Define/trata as tuplas em cada uma das relações que não correspondam a alguma das tuplas da outra relação. Sendo dividido em relação interna, com o comando INNER JOIN, e relações externas: LEFT JOIN, RIGHT JOIN e FULL JOIN.



CONDIÇÃO DE JUNÇÃO

Define as tuplas nas duas relações que são correspondentes, garantindo que os atributos utilizados em ambas as tabelas estejam presentes tanto na sintaxe SQL, quanto nos seus resultados.

Parâmetro JOIN

Para realizar as junções nas consultas as tabelas, faz-se necessário a utilização de um dos comandos mais importantes na estrutura SQL, o JOIN e suas variações. Com a utilização do comando **JOIN** (Junção) é possível por meio do SELECT, unir duas ou mais tabelas, ao se apontar os campos correspondentes entre elas.

A sintaxe utilizada para se efetuar junções nas consultas em SQL é definida como:

```
SELECT [campo] FROM [tabela_1 JOIN tabela_2]
ON [tabela_1].[chave_primária] = [tabela_2].[chave_estrangeira]
WHERE [condição];
```

INNER JOIN

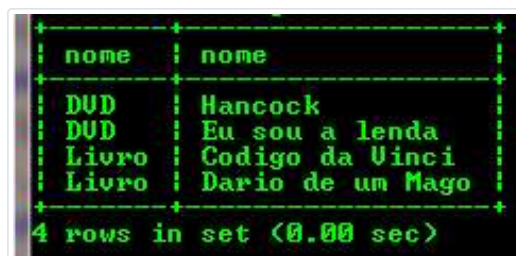
No nosso exemplo, temos a relação entre categorias e produtos, para efetuar uma consulta que nos retorne o nome da categoria e seus respectivos nomes dos produtos.

Sintaxe:

```
SELECT categoria.nome, produto.nome
FROM Categoria INNER JOIN Produto
ON Categoria.Id = Produto.Id_Categoria
```

[Saiba Mais](#)

Resultado do exemplo



nome	nome
DVD	Hancock
DVD	Eu sou a lenda
Livro	Codigo da Vinci
Livro	Dario de um Mago

4 rows in set (0.00 sec)

Fonte: elaborada pelo autor, captura de tela do software MySQL.

Select JOIN com condição

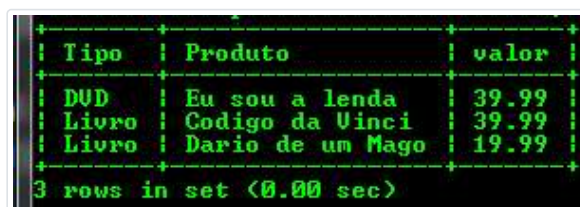
É possível utilizar condições nas consultas, quando necessário fazer as junções entre as tabelas.

Exemplo: exibir o nome da categoria como "Tipo", o nome do produto como "Produto", e o valor dos produtos, em uma condição que o valor dos produtos seja menor que R\$ 50,00.

Sintaxe:

```
SELECT categoria.nome as "Tipo", produto.nome as "Produto", produto.valor
FROM Categoria INNER JOIN Produto
ON Categoria.Id = Produto.Id_Categoria
WHERE produto.valor
```

Resultado do exemplo



Tipo	Produto	valor
DVD	Eu sou a lenda	39.99
Livro	Codigo da Vinci	39.99
Livro	Dario de um Mago	19.99

3 rows in set (0.00 sec)

Fonte: elaborada pelo autor, captura de tela do software MySQL.

Junção externa

Quando o operador de junção externa é utilizado no SQL, é gerado o resultado da junção mais as linhas não combinadas. Sendo possível efetuar junções externas em ambos os lados, ou seja, da esquerda para a direita, e da direita para a esquerda.

Dessa forma, a junção externa independente do lado escolhido, gera uma nova tabela que é a junção das linhas combinadas e não combinadas.

LEFT JOIN

No comando LEFT JOIN, as linhas da tabela da esquerda são projetadas na seleção, juntamente com as linhas não combinadas da tabela da direita. Ou seja, como resultado dessa seleção, algumas linhas em que não se tenha relacionamento entre as tabelas da esquerda para a direita, retornarão o valor nulo (NULL).

Sintaxe:

```
SELECT categoria.nome as "Tipo", produto.nome as "Produto", produto.valor
FROM Categoria LEFT JOIN Produto
ON Categoria.Id = Produto.Id_Categoria;
```

Resultado do exemplo

São projetados todos os registros da tabela da esquerda, e na última seleção a linha não combinada. Pois não existe nenhum produto associado à categoria "Informática", consequentemente as colunas "Produto" e "Valor" receberam nulo (NULL) como entrada.



Tipo	Produto	valor
Livro	Codigo da Vinci	39.99
DVD	Hancock	89.99
Livro	Dario de um Mago	19.99
DVD	Eu sou a lenda	39.99
Informática	NULL	NULL

5 rows in set (0.00 sec)

Fonte: elaborada pelo autor, captura de tela do software MySQL.

RIGHT JOIN

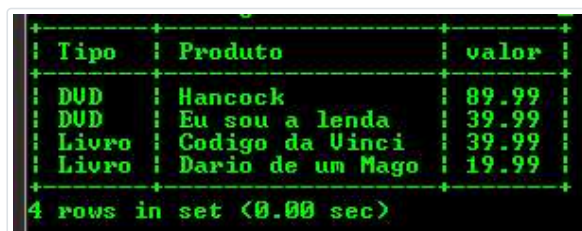
Similar ao comando LEFT JOIN, no comando RIGHT JOIN, as linhas da tabela da direita são projetadas na seleção juntamente com as linhas não combinadas da tabela da esquerda.

Sintaxe:

```
SELECT categoria.nome as "Tipo", produto.nome as "Produto", produto.valor
FROM Categoria RIGHT JOIN Produto
ON Categoria.Id = Produto.Id_Categoria;
```

Resultado do exemplo

São projetados todos os registros da tabela da direita. Como não existe nenhum produto associado à categoria "Informática", consequentemente nenhum registro do tipo nulo (NULL) é demonstrado na seleção dos registros.



Tipo	Produto	valor
DVD	Hancock	89.99
DVD	Eu sou a lenda	39.99
Livro	Codigo da Vinci	39.99
Livro	Dario de um Mago	19.99

4 rows in set (0.00 sec)

Fonte: elaborada pelo autor, captura de tela do software MySQL.

O comando JOIN é utilizado com maior frequência do que você possa imaginar. Alguns exemplos de aplicações web são os sites de hospedagem, sites de compra e buscadores Web, onde são necessárias relações entre tabelas diferentes para gerar o resultado da pesquisa, na qual são efetuadas buscas ou filtros por opções ou palavras chave.

Portanto, vimos a estrutura de junções horizontais e verticais, e os comandos aplicados para estas tarefas, para que você possa se tornar cada vez melhor na programação de banco de dados e desenvolver as suas habilidades.



Fonte: Shutterstock.