

Programação e Desenvolvimento de Banco de Dados

Consultas avançadas

Prof. Dr. Gilberto Fernandes Jr.

- Unidade de Ensino: 3
- Competência da Unidade: Conhecer e compreender a criação e manipulação de tabelas para funções avançadas.
- Resumo: Saber elaborar script SQL para consultas avançadas em tabelas.
- Palavras-chave: SQL, JOIN, agregação
- Título da Teleaula: Consultas avançadas
- Teleaula nº: 3

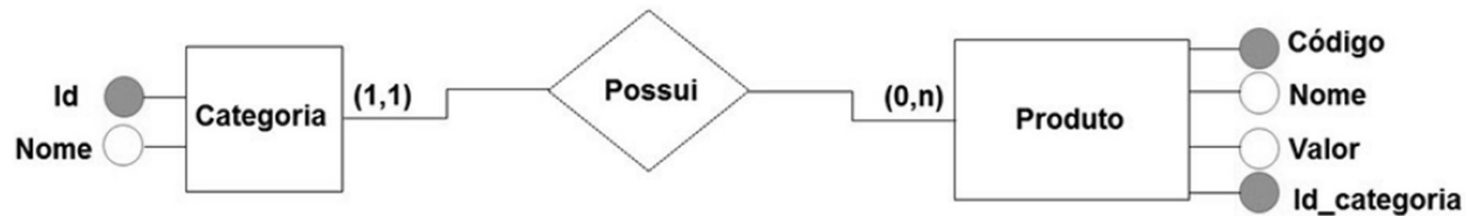
Contextualização

- Junção de dados
- Funções de agregação em banco de dados
- Subconsultas em banco de dados

Junção de dados (JOIN): interna e externa

Introdução

- Consultas em múltiplas tabelas
 - As condições para se efetuar uma junção dependem do **tipo de junção** e de uma **condição de junção**, dessa forma, com o SQL, será possível retornar relações como resultados.
- Considere o exemplo:



Fonte: livro texto

```
CREATE TABLE categoria (  
    id INT(3) PRIMARY KEY  
    AUTO_INCREMENT,  
    Nome VARCHAR(50)  
    NOT NULL);
```

```
CREATE TABLE produto (  
    Código INT(3) PRIMARY KEY AUTO_INCREMENT,  
    Nome VARCHAR(50) NOT NULL,  
    Valor DECIMAL(6,2),  
    Id_Categoria INT(3) NOT NULL,  
    FOREIGN KEY (Id_Categoria)  
    REFERENCES categoria(id));
```

```
INSERT Categoria VALUES  
(0, "DVD"),  
(0, "Livro"),  
(0, "Informática");
```

```
INSERT Produto VALUES  
(0, "Código da Vinci", "39.99", 2),  
(0, "Hancock", "89.99", 1),  
(0, "Dario de um Mago", "19.99", 2),  
(0, "Eu sou a lenda", "39.99", 1);
```

Fonte: elaborado pelo autor

Parâmetro JOIN

- Unir duas ou mais tabelas, ao se apontar os campos correspondentes entre elas.

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

- Tipos de JOIN: INNER, LEFT e RIGHT JOIN.


Junção interna (JOIN / INNER JOIN)

- Consulta que retorne o nome da categoria e seus respectivos nomes dos produtos

```
SELECT categoria.nome, produto.nome FROM Categoria  
INNER JOIN
```

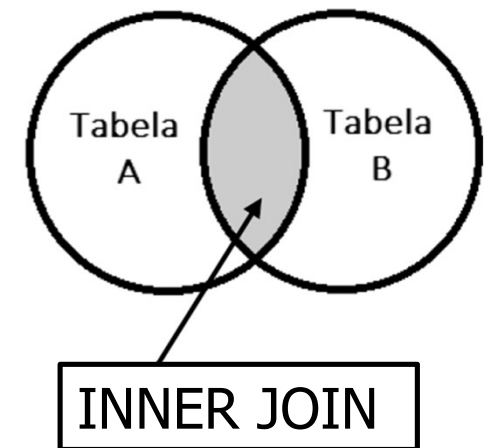
Produto

```
ON Categoria.Id = Produto.Id_Categoria;
```



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

Fonte: elaborado pelo autor




Fonte: elaborado pelo autor

INNER JOIN

- Utilizar condições nas consultas, quando for necessário fazer as junções entre as tabelas

```
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 < 50.00;
```



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

Fonte: elaborado pelo autor

Junção Externa

- Considere as tabelas de exemplo:

SELECT * FROM categoria

id	Nome
1	DVD
2	Livro
3	Informática

SELECT * FROM produto

Código	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

Fonte: elaborado pelo autor

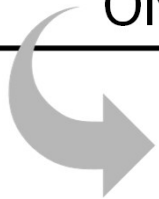
Junção Externa

- Quando o operador de junção externa for utilizado no SQL, é gerado o resultado da junção mais as linhas não combinadas.
- É possível efetuar junções externas em ambos os lados

LEFT JOIN

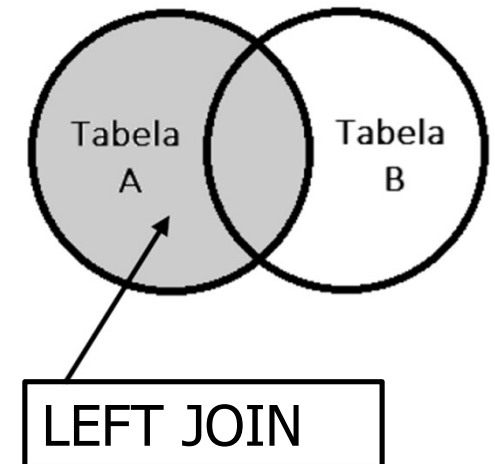
- Exemplo:

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



Tipo	Produto	valor
Livro	Código 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

Fonte: elaborado pelo autor




Fonte: elaborado pelo autor

RIGHT JOIN

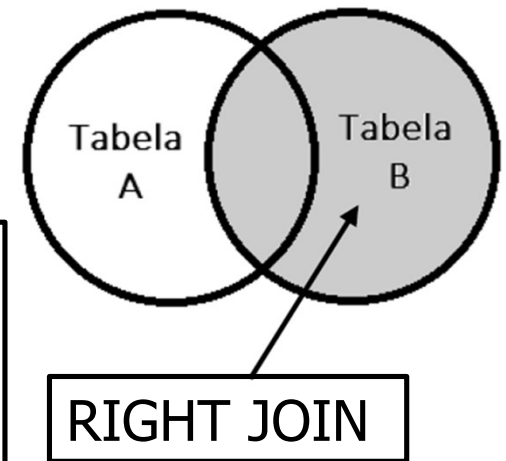
- Exemplo:

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



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

Fonte: elaborado pelo autor



Fonte: elaborado pelo autor

Funções de agregação em bancos de dados

Introdução

- Funções de agregação em SQL: desenvolver consultas utilizando valores das colunas como parâmetro de pesquisa (SELECT).
- Os resultados das seleções podem ser organizados em grupos, baseados no conteúdo existente em uma ou mais colunas.

```
SELECT [coluna] FROM [tabela]  
WHERE [condição]  
GROUP BY [coluna];
```

```
mysql> describe Veiculos;
```

Field	Type	Null	Key	Default	Extra
Id	int(3)	NO	PRI	NULL	auto_increment
Marca	varchar(30)	NO		NULL	
Modelo	varchar(30)	NO		NULL	
Valor	decimal(10,2)	YES		NULL	

```
4 rows in set (0.01 sec)
```

- Considere o exemplo

```
mysql> select * from Veiculos;
```

Id	Marca	Modelo	Valor
1	BMW	320i	160000.00
2	Mercedes_Benz	C180	140000.00
3	Hyundai	Azera	120000.00
4	Mercedes_Benz	CLA 200	140000.00
5	BMW	328i	210000.00
6	Volkswagen	Passat	140000.00
7	BMW	316i	115000.00
8	Mercedes_Benz	Classe E	248000.00
9	Mercedes_Benz	C 250	180000.00
10	Jaguar	XF	220000.00
11	BMW	535i	500000.00
12	Jaguar	VZ	NULL

```
12 rows in set (0.00 sec)
```

Fonte: livro texto

Fonte:
livro texto


COUNT

- Permite que se possa contar o número de registros de uma relação.

```
SELECT COUNT(*) FROM <tabela>;
```

- Pelo exemplo:

```
SELECT  
COUNT(*)  
FROM Veiculos;
```



```
mysql> select count(*) from Veiculos;  
+-----+  
| count(*) |  
+-----+  
|        12 |  
+-----+  
1 row in set (0.00 sec)
```

Fonte: livro texto

COUNT

- Contar uma coluna em específico
 - Cuidado: o contador ignora os registros em que haja valor nulo (NULL)

```
SELECT COUNT(Modelo) FROM Veiculos;
```

- Função DISTINCT


```
SELECT COUNT( DISTINCT MARCA) FROM Veiculos;
```

TOTAL (SUM)

- A função SUM retorna o somatório dos valores em uma determinada coluna.

```
SELECT SUM(<coluna>) FROM <tabela>;
```

```
SELECT SUM(Valor) as "Total" FROM Veiculos;
```



Total
2173000.00

Fonte: livro texto

MINIMUM (MIN)

- Permite que se possa determinar o menor valor de registro em uma coluna.

```
SELECT MIN(<coluna>) FROM <tabela>;
```

Exemplos:

```
SELECT Marca, Modelo, MIN(Valor) as "Menor Valor"  
FROM Veiculos;
```



Marca	Modelo	Menor Valor
BMW	320i	115000.00

Fonte: livro texto

```
SELECT Modelo, MIN(Valor) as "Menor Valor"  
FROM Veiculos WHERE Marca = "Jaguar"
```



Modelo	Menor Valor
XF	220000.00

Fonte: livro texto

MAXIMUM (MAX)

- Permite que se possa determinar o maior valor de registro em uma coluna.

```
SELECT MAX(<coluna>) FROM <tabela>;
```

```
SELECT Marca, Modelo, MAX(Valor) as "Maior Valor"  
FROM Veiculos;
```



Modelo	Maior Valor
320i	500000.00

Fonte: livro texto

AVERAGE (AVG)

- Retorna a média dos valores em uma determinada coluna

```
SELECT AVG(<coluna>) FROM <tabela>;
```

```
SELECT AVG(Valor) as "Valor Médio" FROM Veiculos;
```



Valor Médio
197545.454545

Fonte: livro texto

AVERAGE (AVG)

- A função de agregação AVG permite que o qualificador GROUP BY seja utilizado

```
SELECT Marca, AVG(Valor) as "Valor Médio"  
FROM Veiculos  
GROUP BY Marca;
```



Marca	Valor Médio
BMW	246250.000000
Hyundai	120000.000000
Jaguar	220000.000000
Mercedes_Benz	177000.000000
Volkswagen	140000.000000

Fonte:
livro texto

Prática de consultas avanzadas em um banco de dados

Descrição da SP

- A empresa em que você trabalha foi contratada por uma loja que vende jogos de videogame e consoles, que deseja automatizar o atendimento aos seus clientes com *tokens* de autoatendimento.
- Estrutura do banco já está desenvolvida.
- E os dados iniciais foram inseridos.

Descrição da SP

- Estrutura do banco da loja:

```
mysql> describe jogo;
```

Field	Type	Null	Key	Default	Extra
Cod	int(3)	NO	PRI	NULL	auto_increment
Nome	varchar(50)	NO		NULL	
Valor	decimal(6,2)	NO		NULL	
Localizacao_Id	int(3)	NO	MUL	NULL	

Fonte: livro texto

```
mysql> describe localizacao;
```

Field	Type	Null	Key	Default	Extra
Id	int(3)	NO	PRI	NULL	auto_increment
Secao	varchar(50)	NO		NULL	
Prateleira	int(3)	NO		NULL	

Fonte: livro texto

Descrição da SP

- Tornar a aplicação seja capaz de:
 - **Identificar o nome do jogo e a prateleira**, dando o nome de uma seção;
 - **Identificar todas as seções e os respectivos nomes dos jogos**, ordenando as seleções em ordem crescente pelo nome dos jogos.
 - **Identificar o nome dos jogos da seção de jogos de guerra**, por serem os mais procurados.

Descrição da SP

- Em seguida, o gerente de projetos solicitou que você desenvolva funções de agregação para:
 - A quantidade de registros na tabela jogo;
 - O valor do jogo de maior preço (valor);
 - O valor do jogo de menor preço (valor);
 - O valor médio dos jogos de guerra; e que o valor total em estoque na loja.

Subconsultas em banco de dados

Subconsultas com IN e NOT IN

- Segundo Silberschatz (2010), é considerada uma subconsulta uma expressão em SQL, composta por SELECT-FROM-WHERE, aninhada dentro de outra consulta, permitindo fazer comparações entre os conjuntos de dados.
- Conectivos **IN** e **NOT IN**

```
SELECT [campo]  
FROM [tabela]  
WHERE [campo] IN (SELECT [campo] FROM [tabela]);
```

Aluno

RA	nome	telefone
11	Maria	11999988999
22	Joao	11999997899
33	Jose	11999995699
44	Jonas	11999997699

Empréstimo

numero	retirada	devolucao	aluno_RA	funcionario_mat	livro_isbn
1	2018-01-15	2018-02-01	11	1	11111
2	2018-04-05	2018-04-20	22	1	22222

Restrição

id	aluno_RA	livro_isbn
1	33	44444

```
SELECT aluno.nome FROM aluno
WHERE aluno.RA NOT IN
(SELECT aluno_RA FROM restricao);
```



nome
Maria
Joao
Jonas

```
SELECT aluno.nome AS 'ALUNO' FROM aluno
WHERE aluno.RA NOT IN
(SELECT aluno_RA FROM emprestimo);
```



ALUNO
Jose
Jonas

Fonte: elaborado pelo autor

Comparação de Conjuntos

- O SQL permite desenvolver subconsultas aninhadas utilizando condições (WHERE).

Operador Matemático	SELECT com WHERE SQL	Subconsulta SQL
=	WHERE campo = condição	WHERE campo = some (SELECT)
≠	WHERE campo <> condição	WHERE campo <> some (SELECT)
>	WHERE campo > condição	WHERE campo > some (SELECT)
≥	WHERE campo >= condição	WHERE campo >= some (SELECT)
<	WHERE campo < condição	WHERE campo < some (SELECT)
≤	WHERE campo <= condição	WHERE campo <= some (SELECT)

Fonte: livro texto

- Exemplo: Listar os nomes dos empregados que têm salário superior a algum empregado do departamento B

```
SELECT nome, salario,
dpto FROM colaborador;
WHERE salario > SOME
(SELECT salario FROM
colaborador WHERE
dpto = 'B');
```

nome	salario	dpto
joao	3000.00	B
maria	6000.00	A
zé	2000.00	A
bil	3000.00	B
ana	5000.00	C

salario
3000.00
3000.00

Subconsulta resultante:

nome	salario	dpto
maria	6000.00	A
ana	5000.00	C

Resolução de SP

- Você trabalha na empresa que está prestando serviço para a loja de games especializada em jogos para os consoles e computadores.
- Com o sucesso das vendas e da tecnologia de autoatendimento com *tokens*, novos títulos foram adicionados às prateleiras

Descrição da SP

- Seu gerente de projetos solicitou:
 - **Inserir** os novos títulos no banco de dados
 - **Alterar** o valor dos jogos em promoção.
 - **Criar** uma tabela chamada promoção e inserir os jogos em promoção na tabela criada.
 - Uma forma de **selecionar** o nome do jogo, o valor e o nome da seção dos títulos em promoção.
 - Uma forma de **selecionar** o nome dos títulos e seus respectivos valores, que não estejam em promoção.

Recapitulando

Recapitulando

- Junção de dados
 - JOIN
 - INNER JOIN
 - LEFT JOIN / RIGHT JOIN
- Funções de agregação em banco de dados
 - COUNT, MIN, MAX, AVG, SUM

Recapitulando

- Subconsultas em banco de dados
- SELECT-FROM-WHERE, aninhada dentro de outra consulta
- Comparação entre conjuntos de dados