



Geek University

Evolua seu lado geek!

www.geekuniversity.com.br

Funções de Agrupamento e Ordenação

Funções de Agrupamento e Ordenação

Contextualizando...

Durante o desenvolvimento de aplicações as vezes faz-se necessário agrupar e/ou ordenar os resultados para uma melhor apresentação.

A linguagem **SQL** provê recursos para realizar estas operações.

Funções de Agrupamento e Ordenação

Base de Dados Exemplo...

Funções de Agrupamento e Ordenação

Base de Dados Exemplo...

```
CREATE DATABASE agrupamento;  
  
USE agrupamento;  
  
CREATE TABLE tipos(  
    id INT NOT NULL AUTO_INCREMENT,  
    nome VARCHAR(60) NOT NULL,  
    PRIMARY KEY (id)  
);  
  
CREATE TABLE fabricantes(  
    id INT NOT NULL AUTO_INCREMENT,  
    nome VARCHAR(60) NOT NULL,  
    PRIMARY KEY (id)  
);  
  
CREATE TABLE produtos(  
    id INT NOT NULL AUTO_INCREMENT,  
    nome VARCHAR(60) NOT NULL,  
    id_fabricante INT NOT NULL,  
    quantidade INT NOT NULL,  
    id_tipo int NOT NULL,  
    PRIMARY KEY (id),  
    FOREIGN KEY (id_fabricante) REFERENCES fabricantes(id),  
    FOREIGN KEY (id_tipo) REFERENCES tipos(id)  
);
```

Comando DDL para criação do banco de dados.

Funções de Agrupamento e Ordenação

Base de Dados Exemplo...

```
CREATE DATABASE agrupamento;  
  
USE agrupamento;  
  
CREATE TABLE tipos(  
    id INT NOT NULL AUTO_INCREMENT,  
    nome VARCHAR(60) NOT NULL,  
    PRIMARY KEY (id)  
);  
  
CREATE TABLE fabricantes(  
    id INT NOT NULL AUTO_INCREMENT,  
    nome VARCHAR(60) NOT NULL,  
    PRIMARY KEY (id)  
);  
  
CREATE TABLE produtos(  
    id INT NOT NULL AUTO_INCREMENT,  
    nome VARCHAR(60) NOT NULL,  
    id_fabricante INT NOT NULL,  
    quantidade INT NOT NULL,  
    id_tipo int NOT NULL,  
    PRIMARY KEY (id),  
    FOREIGN KEY (id_fabricante) REFERENCES fabricantes(id),  
    FOREIGN KEY (id_tipo) REFERENCES tipos(id)  
);
```

Comando DML para utilização do banco de dados.

Funções de Agrupamento e Ordenação

Base de Dados Exemplo...

```
CREATE DATABASE agrupamento;

USE agrupamento;

CREATE TABLE tipos(
    id INT NOT NULL AUTO_INCREMENT,
    nome VARCHAR(60) NOT NULL,
    PRIMARY KEY (id)
);

CREATE TABLE fabricantes(
    id INT NOT NULL AUTO_INCREMENT,
    nome VARCHAR(60) NOT NULL,
    PRIMARY KEY (id)
);

CREATE TABLE produtos(
    id INT NOT NULL AUTO_INCREMENT,
    nome VARCHAR(60) NOT NULL,
    id_fabricante INT NOT NULL,
    quantidade INT NOT NULL,
    id_tipo int NOT NULL,
    PRIMARY KEY (id),
    FOREIGN KEY (id_fabricante) REFERENCES fabricantes(id),
    FOREIGN KEY (id_tipo) REFERENCES tipos(id)
);
```

Comando DDL para criação de tabelas.

Funções de Agrupamento e Ordenação

Base de Dados Exemplo...

```
INSERT INTO tipos (nome) VALUES ('Console');
INSERT INTO tipos (nome) VALUES ('Notebook');
INSERT INTO tipos (nome) VALUES ('Celular');
INSERT INTO tipos (nome) VALUES ('Smartphone');
INSERT INTO tipos (nome) VALUES ('Sofá');
INSERT INTO tipos (nome) VALUES ('Armário');
INSERT INTO tipos (nome) VALUES ('Refrigerador');

INSERT INTO fabricantes (nome) VALUES ('Sony');
INSERT INTO fabricantes (nome) VALUES ('Dell');
INSERT INTO fabricantes (nome) VALUES ('Microsoft');
INSERT INTO fabricantes (nome) VALUES ('Samsung');
INSERT INTO fabricantes (nome) VALUES ('Apple');
INSERT INTO fabricantes (nome) VALUES ('Beline');
INSERT INTO fabricantes (nome) VALUES ('Magno');
INSERT INTO fabricantes (nome) VALUES ('CCE');
INSERT INTO fabricantes (nome) VALUES ('Nintendo');

INSERT INTO produtos (nome, id_fabricante, quantidade, id_tipo) VALUES ('Playstation 3', 1, 100, 1);
INSERT INTO produtos (nome, id_fabricante, quantidade, id_tipo) VALUES ('Core 2 Duo 4GB RAM 500GB HD', 2, 200, 2);
INSERT INTO produtos (nome, id_fabricante, quantidade, id_tipo) VALUES ('Xbox 360 120GB', 3, 350, 1);
INSERT INTO produtos (nome, id_fabricante, quantidade, id_tipo) VALUES ('GT-I6220 Quad band', 4, 300, 3);
INSERT INTO produtos (nome, id_fabricante, quantidade, id_tipo) VALUES ('iPhone 4 32GB', 5, 50, 4);
INSERT INTO produtos (nome, id_fabricante, quantidade, id_tipo) VALUES ('Playstation 2', 1, 100, 1);
INSERT INTO produtos (nome, id_fabricante, quantidade, id_tipo) VALUES ('Sofá Estofado', 6, 200, 5);
INSERT INTO produtos (nome, id_fabricante, quantidade, id_tipo) VALUES ('Armário de Serviço', 7, 50, 6);
INSERT INTO produtos (nome, id_fabricante, quantidade, id_tipo) VALUES ('Refrigerador 420L', 8, 200, 7);
INSERT INTO produtos (nome, id_fabricante, quantidade, id_tipo) VALUES ('Wii 120GB', 8, 250, 1);
```

Comando DML inserção de dados nas tabelas.

Funções de Agrupamento e Ordenação

GROUP BY...

Utilizamos a cláusula **GROUP BY** para agrupar elementos do mesmo tipo.

Funções de Agrupamento e Ordenação

GROUP BY...Exemplo 1

```
SELECT t.nome AS Tipo, SUM(p.quantidade) AS 'Quantidade em Estoque' FROM tipos AS t, produtos AS p  
WHERE t.id = p.id_tipo GROUP BY t.nome;
```

No exemplo acima estamos solicitando a quantidade de produtos em estoque, agrupados pelo tipo. Utilizamos a **função de agregação SUM()** para efetuar a soma de cada tipo de produto.

Funções de Agrupamento e Ordenação

GROUP BY...Exemplo 1

```
SELECT t.nome AS Tipo, SUM(p.quantidade) AS 'Quantidade em Estoque' FROM tipos AS t, produtos AS p  
WHERE t.id = p.id_tipo GROUP BY t.nome;
```

No exemplo acima estamos solicitando a quantidade de produtos em estoque, agrupados pelo tipo. Utilizamos a **função de agregação SUM()** para efetuar a soma de cada tipo de produto.

#	Tipo	Quantidade em Estoque
1	Armário	50
2	Celular	300
3	Console	800
4	Notebook	200
5	Refrigerador	200
6	Smartphone	50
7	Sofá	200

Funções de Agrupamento e Ordenação

GROUP BY...Exemplo 2

```
SELECT f.nome AS Fabricante, SUM(p.quantidade) AS 'Quantidade em Estoque' FROM fabricantes AS f,  
produtos AS p WHERE f.id = p.id_fabricante GROUP BY f.nome;
```

No exemplo acima estamos solicitando a quantidade de produtos em estoque, agrupados pelo fabricante. Utilizamos a **função de agregação SUM()** para efetuar a soma de cada produto por fabricante.

Funções de Agrupamento e Ordenação

GROUP BY...Exemplo 2

```
SELECT f.nome AS Fabricante, SUM(p.quantidade) AS 'Quantidade em Estoque' FROM fabricantes AS f, produtos AS p WHERE f.id = p.id_fabricante GROUP BY f.nome;
```

No exemplo acima estamos solicitando a quantidade de produtos em estoque, agrupados pelo fabricante. Utilizamos a **função de agregação SUM()** para efetuar a soma de cada produto por fabricante.

#	Fabricante	Quantidade em Estoque
1	Apple	50
2	Beline	200
3	CCE	450
4	Dell	200
5	Magno	50
6	Microsoft	350
7	Samsung	300
8	Sony	200

Funções de Agrupamento e Ordenação

GROUP BY...Exemplo 3

```
SELECT t.nome AS Tipo, f.nome AS Fabricante, SUM(p.quantidade) AS 'Quantidade em Estoque'  
FROM tipos AS t, fabricantes AS f, produtos AS p  
WHERE t.id = p.id_tipo AND f.id = p.id_fabricante  
GROUP BY t.nome, f.nome;
```

No exemplo acima estamos solicitando a quantidade de produtos em estoque de acordo com os tipos e fabricantes.

Funções de Agrupamento e Ordenação

GROUP BY...Exemplo 3

```
SELECT t.nome AS Tipo, f.nome AS Fabricante, SUM(p.quantidade) AS 'Quantidade em Estoque'  
FROM tipos AS t, fabricantes AS f, produtos AS p  
WHERE t.id = p.id_tipo AND f.id = p.id_fabricante  
GROUP BY t.nome, f.nome;
```

No exemplo acima estamos solicitando a quantidade de produtos em estoque de acordo com os tipos e fabricantes.

#	Tipo	Fabricante	Quantidade em Estoque
1	Armário	Magno	50
2	Celular	Samsung	300
3	Console	CCE	250
4	Console	Microsoft	350
5	Console	Sony	200
6	Notebook	Dell	200
7	Refrigerador	CCE	200
8	Smartphone	Apple	50
9	Sofá	Beline	200

Funções de Agrupamento e Ordenação

GROUP BY...Exemplo 4

```
SELECT t.nome AS Tipo, f.nome AS Fabricante, SUM(p.quantidade) AS 'Quantidade em Estoque'  
FROM tipos AS t, fabricantes AS f, produtos AS p  
WHERE t.id = p.id_tipo AND f.id = p.id_fabricante  
GROUP BY t.nome, f.nome  
HAVING SUM(p.quantidade) > 200;
```

No exemplo acima estamos solicitando a quantidade de produtos em estoque de acordo com os tipos e fabricantes onde a quantidade seja maior que 200 itens em estoque.

Funções de Agrupamento e Ordenação

GROUP BY...Exemplo 4

```
SELECT t.nome AS Tipo, f.nome AS Fabricante, SUM(p.quantidade) AS 'Quantidade em Estoque'  
FROM tipos AS t, fabricantes AS f, produtos AS p  
WHERE t.id = p.id_tipo AND f.id = p.id_fabricante  
GROUP BY t.nome, f.nome  
HAVING SUM(p.quantidade) > 200;
```

No exemplo acima estamos solicitando a quantidade de produtos em estoque de acordo com os tipos e fabricantes onde a quantidade seja maior que 200 itens em estoque.

#	Tipo	Fabricante	Quantidade em Estoque
1	Celular	Samsung	300
2	Console	CCE	250
3	Console	Microsoft	350

Funções de Agrupamento e Ordenação

GROUP BY...Exemplo 5

Talves não tenha ficado muito claro ainda a importância do GROUP BY.

Vamos ver um exemplo mais simples mas que facilita a visualização.

Imagine que você precisa de todos os tipos de produtos dos produtos existentes.

```
SELECT t.nome AS Tipo FROM produtos AS p, tipos AS t WHERE t.id = p.id_tipo;
```

O código SQL acima parece ok...mas veja o resultado...

Funções de Agrupamento e Ordenação

GROUP BY...Exemplo 5

Talves não tenha ficado muito claro ainda a importância do GROUP BY.

Vamos ver um exemplo mais simples mas que facilita a visualização.

Imagine que você precisa de todos os tipos de produtos dos produtos existentes.

```
SELECT t.nome AS Tipo FROM produtos AS p, tipos AS t WHERE t.id = p.id_tipo;
```

O código SQL acima parece ok...mas veja o resultado...

Veja que temos repetição...simplesmente porque não estamos realizando nenhum agrupamento...

#	Tipo
1	Console
2	Console
3	Console
4	Console
5	Notebook
6	Celular
7	Smartphone
8	Sofá
9	Armário
10	Refrigerador

Funções de Agrupamento e Ordenação

GROUP BY...Exemplo 5

```
SELECT t.nome AS Tipo FROM produtos AS p, tipos AS t WHERE t.id = p.id_tipo GROUP BY t.nome;
```

Veja o mesmo código SQL anterior com a simples adição do **GROUP BY**

#	Tipo
1	Armário
2	Celular
3	Console
4	Notebook
5	Refrigerador
6	Smartphone
7	Sofá

Funções de Agrupamento e Ordenação

ORDER BY...

Funções de Agrupamento e Ordenação

ORDER BY...

A cláusula **ORDER BY** organiza os dados em ordem alfabética ou numérica.

A ordenação pode ser ASC (Ascendente) ou DESC (Descendente)

Por padrão, a ordenação é ascendente, ou seja, do menor para o maior.

Funções de Agrupamento e Ordenação

ORDER BY...Exemplo 1

```
SELECT id, nome, id_tipo, id_fabricante, quantidade FROM produtos;
```

Funções de Agrupamento e Ordenação

ORDER BY...Exemplo 1

```
SELECT id, nome, id_tipo, id_fabricante, quantidade FROM produtos;
```

#	id	nome	id_fabricante	quantidade	id_tipo
1	1	Playstation 3	1	100	1
2	2	Core 2 Duo 4GB RAM 500GB HD	2	200	2
3	3	Xbox 360 120GB	3	350	1
4	4	GT-I6220 Quad band	4	300	3
5	5	iPhone 4 32GB	5	50	4
6	6	Playstation 2	1	100	1
7	7	Sofá Estofado	6	200	5
8	8	Armário de Serviço	7	50	6
9	9	Refrigerador 420L	8	200	7
10	10	Wii 120GB	8	250	1

Funções de Agrupamento e Ordenação

ORDER BY...Exemplo 1

```
SELECT id, nome, id_tipo, id_fabricante, quantidade FROM produtos;
```

#	id	nome	id_fabricante	quantidade	id_tipo
1	1	Playstation 3	1	100	1
2	2	Core 2 Duo 4GB RAM 500GB HD	2	200	2
3	3	Xbox 360 120GB	3	350	1
4	4	GT-I6220 Quad band	4	300	3
5	5	iPhone 4 32GB	5	50	4
6	6	Playstation 2	1	100	1
7	7	Sofá Estofado	6	200	5
8	8	Armário de Serviço	7	50	6
9	9	Refrigerador 420L	8	200	7
10	10	Wii 120GB	8	250	1

Nesta hora você pode estar se perguntando: Mas só fizemos uma seção comum. Não fizemos aqui nenhuma ordenação.

A verdade é que, por padrão a ordenação é ascendente pela chave primária da tabela.

Funções de Agrupamento e Ordenação

ORDER BY...Exemplo 2

```
SELECT id, nome, id_tipo, id_fabricante, quantidade FROM produtos ORDER BY id ASC;
```

Funções de Agrupamento e Ordenação

ORDER BY...Exemplo 2

```
SELECT id, nome, id_tipo, id_fabricante, quantidade FROM produtos ORDER BY id ASC;
```

#	id	nome	id_fabricante	quantidade	id_tipo
1	1	Playstation 3	1	100	1
2	2	Core 2 Duo 4GB RAM 500GB HD	2	200	2
3	3	Xbox 360 120GB	3	350	1
4	4	GT-I6220 Quad band	4	300	3
5	5	iPhone 4 32GB	5	50	4
6	6	Playstation 2	1	100	1
7	7	Sofá Estofado	6	200	5
8	8	Armário de Serviço	7	50	6
9	9	Refrigerador 420L	8	200	7
10	10	Wii 120GB	8	250	1

Podemos especificar, se quisermos, a ordenação padrão....mas o resultado conforme visto é o mesmo, portando desnecessária.

OBS: Se quisermos realizar a ordenação padrão trazendo todos os campos podemos utilizar *. Mas se quisermos ordenar por um campo específico ou mesmo mudar a forma de ordenação, devemos informar os campos.

Funções de Agrupamento e Ordenação

ORDER BY...Exemplo 3

```
SELECT id, nome, id_tipo, id_fabricante, quantidade FROM produtos ORDER BY id DESC;
```

Funções de Agrupamento e Ordenação

ORDER BY...Exemplo 3

```
SELECT id, nome, id_tipo, id_fabricante, quantidade FROM produtos ORDER BY id DESC;
```

	id	nome	id_tipo	id_fabricante	quantidade
1	10	Wii 120GB	1	8	250
2	9	Refrigerador 420L	7	8	200
3	8	Armário de Serviço	6	7	50
4	7	Sofá Estofado	5	6	200
5	6	Playstation 2	1	1	100
6	5	iPhone 4 32GB	4	5	50
7	4	GT-I6220 Quad band	3	4	300
8	3	Xbox 360 120GB	1	3	350
9	2	Core 2 Duo 4GB RAM 500GB HD	2	2	200
10	1	Playstation 3	1	1	100

Podemos solicitar que a ordenação seja descendente, ou seja, do maior para o menor.

OBS: Neste exemplo ordenamos pelo campo id mas podemos ordenar por qualquer campo.

Funções de Agrupamento e Ordenação

ORDER BY...Exemplo 4

```
SELECT id, nome, id_tipo, id_fabricante, quantidade FROM produtos ORDER BY quantidade DESC;
```

Funções de Agrupamento e Ordenação

ORDER BY...Exemplo 4

```
SELECT id, nome, id_tipo, id_fabricante, quantidade FROM produtos ORDER BY quantidade DESC;
```

#	id	nome	id_tipo	id_fabricante	quantidade
1	3	Xbox 360 120GB	1	3	350
2	4	GT-I6220 Quad band	3	4	300
3	10	Wii 120GB	1	8	250
4	2	Core 2 Duo 4GB RAM 500GB HD	2	2	200
5	7	Sofá Estofado	5	6	200
6	9	Refrigerador 420L	7	8	200
7	1	Playstation 3	1	1	100
8	6	Playstation 2	1	1	100
9	5	iPhone 4 32GB	4	5	50
10	8	Armário de Serviço	6	7	50
*	NULL	NULL	NULL	NULL	NULL

Neste exemplo estamos ordenando de forma descendente pela quantidade de produtos.



Geek University

Evolua seu lado geek!

www.geekuniversity.com.br