

# Banco de Dados : Agrupamentos e Funções de Agregação

Prof. Márcio Funes



## Plano de aula

Funções de Agregação  
Consultas com Agrupamentos  
Exercícios

# Introdução

## O contexto

A visualização adequada de dados, sobretudo em **grandes bases**, requer meios para obter **informações resumidas**.

Por exemplo, obter dados estatísticos como médias e somatórias.

## Funções de Agregação

# Funções de Agregação

As funções de agregação resumem os dados de um conjunto em um único valor escalar.

A sintaxe é

`FUNCAO_DE_AGREGACAO(conjunto)`

Exemplos:

`SELECT MAX (preco) FROM produtos`

`SELECT COUNT(id) FROM clientes;`

`SELECT SUM(peso) FROM pacientes;`

## Funções de Agregação

As principais funções de agregação são:

Função	Descrição
COUNT	Retorna a contagem de linhas não nulas do conjunto
SUM	Retorna a somatória de um conjunto numérico
AVG	Retorna a média aritmética de um conjunto numérico
MIN	Retorna o menor valor de um conjunto
MAX	Retorna o maior valor de um conjunto

# Agrupamento

## Agrupamento

Muitas vezes é preciso obter informações organizadas em grupos.

Por exemplo, agrupar todos os clientes por cidade, onde cada grupo é um resumo daqueles dados.

Com isso, é possível saber quantos clientes há em cada cidade ou a média de suas idades, por exemplo.



## Agrupamento

A cláusula **GROUP BY** organiza linhas agrupando dados semelhantes, de maneira que o resultado é um resumo dessas linhas.

Ela pode ser usada com as funções de agregação como **SUM**, **COUNT** e **MAX**. São as funções de agregação que irão resumir os dados de cada grupo.

Em outras palavras, **GROUP BY** permite segmentar os resultados das funções de agregação.

## Agrupamento

A sintaxe da cláusula GROUP BY é:

```
SELECT colunas  
FROM tabelas  
GROUP BY colunas;
```

- É usada em conjunto com o SELECT;
- Deve ser colocada após o WHERE (se houver) e
- Antes de ORDER BY (se houver)
- Todas as colunas da seleção que não são valores agregados devem constar na cláusula GROUP BY

# Agrupamento

```
SELECT cor FROM Animais;
```

	cor
1	branco
2	preto
3	azul
4	laranja
5	preto
6	amarelo
7	preto
8	azul
9	roxo
10	amarelo
11	amarelo
12	branco
13	amarelo

```
SELECT cor FROM Animais GROUP BY cor;
```

	cor
1	amarelo
2	azul
3	branca
4	branco
5	laranja
6	marrom
7	preta
8	preto
9	roxo
10	vermelho

## Agrupamento

Exemplo:

O seguinte comando

```
SELECT cor, COUNT(ID)
FROM Animais
GROUP BY cor;
```

Resulta em:

	cor	(Nenhum nome de coluna)
1	amarelo	14
2	azul	3
3	branca	2
4	branco	5
5	laranja	4
6	marrom	1
7	preta	1
8	preto	6
9	roxo	1
10	vermelho	2

# Agrupamento

Exemplo:

O seguinte comando

```
SELECT cor, AVG(peso)  
FROM Animais  
GROUP BY cor;
```

Resulta em:

	cor	(Nenhum nome de coluna)
1	amarelo	7.864285
2	azul	11.833333
3	branca	2585.650000
4	branco	16.620000
5	laranja	12.300000
6	marrom	19.900000
7	preta	3.200000
8	preto	26.200000
9	roxo	18.600000
10	vermelho	11.900000

## Agrupamento

É possível filtrar os resultados do agrupamento por meio da cláusula **HAVING**.

```
SELECT cor, AVG(peso)
```

Exemplo: **FROM** Animais

```
GROUP BY cor
```

```
HAVING AVG(peso) > 15;
```

	cor	(Nenhum nome de coluna)
1	branca	2585.650000
2	branco	16.620000
3	marrom	19.900000
4	preto	26.200000
5	roxo	18.600000

## Agrupamento

Resumindo em poucas palavras a diferença entre **having** e **where**:

- where filtra linhas
- having filtra grupos

# Agrupamento

## Soma em Agrupamento

Exemplo: `SELECT cor, sum(peso)  
FROM Animais  
GROUP BY cor;`

	cor	(Nenhum nome de coluna)
1	amarelo	110.10
2	azul	35.50
3	branca	5171.30
4	branco	83.10
5	laranja	49.20
6	marrom	19.90
7	preta	3.20
8	preto	157.20
9	roxo	18.60
10	vermelho	23.80



# Agrupamento com SOMA

```
SELECT nome, sum(peso)  
FROM Animais  
GROUP BY nome;
```

	nome	(Nenhum nome de coluna)
1	ágata	44.40
2	bafo de onça	5.50
3	batman	96.10
4	bidu	12.40
5	bola de pelo	11.60
6	coragem	12.20
7	costelinha	13.40
8	dum dum	11.20
9	félix	14.30
10	frajola	13.70
11	gallaxhar	5.50
12	garfield	17.10
13	gato de botas	23.20

```
SELECT nome, sum(peso)  
FROM Animais  
GROUP BY nome  
ORDER BY sum(peso);
```

	nome	(Nenhum nome de coluna)
1	prof. pardal	1.70
2	mikey	2.20
3	minie	3.20
4	ligeirinho	4.40
5	gallaxhar	5.50
6	hathaway	5.50
7	insectosauro	5.50
8	bafo de onça	5.50
9	susan murphy	5.50
10	topo gigio	5.50
11	jerry	6.60
12	dum dum	11.20
13	tom	11.20

## Funções de Mínimo e Máximo

- A função MIN() retorna o menor valor da coluna selecionada.
- A função MAX() retorna o maior valor da coluna selecionada.

```
SELECT MIN(column_name)  
FROM table_name  
WHERE condition;
```

```
SELECT MAX(column_name)  
FROM table_name  
WHERE condition;
```

# Funções de Mínimo e Máximo

```
SELECT MIN(peso)  
FROM Animais;
```

	(Nenhum nome de coluna)
1	1.70

```
SELECT MIN(peso)  
AS Menor_Peso  
FROM Animais;
```

	Menor_Peso
1	1.70

```
SELECT MAX(peso)  
FROM Animais;
```

	(Nenhum nome de coluna)
1	5069.40

```
SELECT MAX(peso)  
AS Maior_Peso  
FROM Animais;
```

	Maior_Peso
1	5069.40

## Mais funções de Seleção

- **TOP:** cláusula é usada para especificar o número de registros a serem retornados. cláusula é útil em tabelas grandes com milhares de registros. Retornar um grande número de registros pode afetar o desempenho.

SQL Server

```
SELECT TOP number | percent column_name(s)  
FROM table_name  
WHERE condition;
```

MySQL

```
SELECT column_name(s)  
FROM table_name  
WHERE condition  
LIMIT number;
```

## Função TOP

A instrução SQL a seguir seleciona os CINCO primeiros registros da tabela "Animais" (para SQL Server):

```
SELECT TOP 5 * FROM Animais;
```

	id	nome	data_nasc	peso	cor	especie_id
1	1	ágata	2015-04-09	13.90	branco	1
2	2	félix	2016-06-06	14.30	preto	1
3	3	tom	2013-02-08	11.20	azul	1
4	4	garfield	2015-07-06	17.10	laranja	1
5	5	frajola	2013-08-01	13.70	preto	1

# Função TOP

A instrução SQL a seguir seleciona os primeiros 50% dos registros da tabela "Animais" (para SQL Server):

```
SELECT TOP 50 PERCENT * FROM Animais;
```

	id	nome	data_nasc	peso	cor	especie_id
1	1	ágata	2015-04-09	13.90	branco	1
2	2	félix	2016-06-06	14.30	preto	1
3	3	tom	2013-02-08	11.20	azul	1
4	4	garfield	2015-07-06	17.10	laranja	1
5	5	frajola	2013-08-01	13.70	preto	1
6	6	manda-chuva	2012-02-03	12.30	amarelo	1
7	7	snowball	2014-04-06	13.20	preto	1
8	8	ágata	2015-08-03	11.90	azul	1
9	9	ágata	2016-03-04	18.60	roxo	1
10	1...	gato de bot...	2012-12-10	11.60	amarelo	1
11	1...	bola de pelo	2020-04-06	11.60	amarelo	2
12	1...	milu	2013-02-04	17.90	branco	2
13	1...	pluto	2012-01-03	12.30	amarelo	2
14	1...	pateta	2015-05-01	17.70	preto	2
15	1...	snoopy	2013-07-02	18.20	branco	2
16	1...	bidu	2012-09-08	12.40	azul	2
17	1...	dum dum	2015-04-06	11.20	laranja	2
18	1...	muttley	2011-02-03	14.30	laranja	2

## Para saber mais...

[https://www.w3schools.com/sql/sql\\_groupby.asp](https://www.w3schools.com/sql/sql_groupby.asp)

<https://www.dofactory.com/sql/group-by>

<https://www.devmedia.com.br/desvendando-a-clausula-group-by-artigo-sql-magazine-47/8082>

<https://www.geeksforgeeks.org/sql-group-by/>

<https://www.devmedia.com.br/exemplos-com-group-by-e-com-a-clausula-having-totalizando-dados-sql-server-2008-parte-2/19839>

<https://www.dofactory.com/sql/having>

## Exercícios



# Exercícios

Baixe o arquivo Empresas.sql

1. Selecione quantos produtos cada marca possui.
2. Selecione o preço médio dos produtos de cada marca.
3. Selecione a média dos preços e total em estoque dos produtos agrupados por marca.
4. Selecione quantos produtos estão cadastrados.
5. Selecione o preço médio dos produtos.
6. Selecione a média dos preços dos produtos em 2 grupos: perecíveis e não perecíveis.
7. Selecione a média dos preços dos produtos agrupados pelo nome do produto.
8. Selecione a média dos preços e total em estoque dos produtos.
9. Selecione o nome, marca e quantidade em estoque do produto mais caro.
10. Selecione os produtos com preço acima da média.
11. Selecione a quantidade de produtos de cada nacionalidade.