

Utilizando o GROUP By e o ORDER By

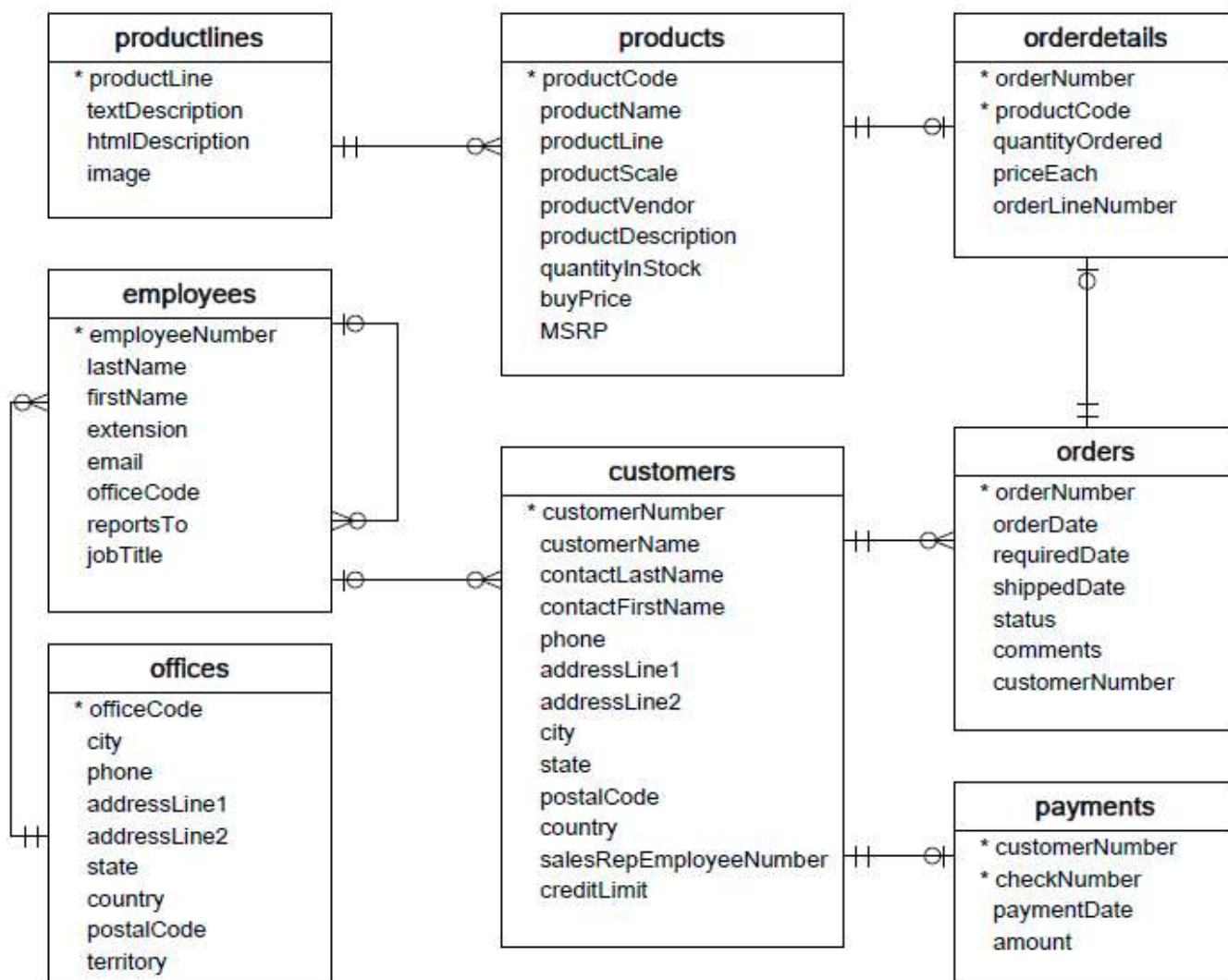
Fonte: <https://dev.mysql.com/doc/refman/8.0/en/group-by-modifiers.html>
<https://www.devmedia.com.br/conceitos-e-sintaxe-do-group-by-totalizando-dados-sql-server-2008-parte-1/19838>

A cláusula GROUP BY

A cláusula GROUP BY tem por função agrupar um conjunto de registros que possuam características idênticas, apresentando um resultado sintético das informações encontradas, ou seja, retorna em uma linha para cada grupo, reduzindo o número de linhas no conjunto de resultados.

O GROUP BY é uma cláusula opcional da DQL (*Data Query Language*) SELECT; abaixo um exemplo da sintaxe utilizada para este recurso.

Em tempo, nesta interação utilizaremos a base de dados do simulador disponível em <https://www.mysqltutorial.org/tryit/>:



Um exemplo simples para iniciar:

```
1 SELECT
2     status
3 FROM
4     orders
5 GROUP BY status;
```

	status
▶	Cancelled
	Disputed
	In Process
	On Hold
	Resolved
	Shipped

A GROUP BY deve aparecer após as cláusulas FROM e WHERE; esta, por sua vez, admite uma lista de colunas ou expressões, separadas por vírgula, como critério para o agrupamento.

Comparativamente, a cláusula GROUP BY é semelhante ao operador DISTINCT:

```
1 SELECT DISTINCT
2     status
3 FROM
4     orders;
```

	status
▶	Cancelled
	Disputed
	In Process
	On Hold
	Resolved
	Shipped

Podemos também utilizar a cláusula GROUP BY com funções agregadas (SUM, AVG, MAX, MIN, e COUNT); segue exemplo:

```
1 SELECT
2     status, COUNT(*)
3 FROM
4     orders
5 GROUP BY status;
```

	status	COUNT(*)
▶	Cancelled	6
	Disputed	3
	In Process	6
	On Hold	4
	Resolved	4
	Shipped	303

Nem sempre queremos realizar agrupamentos em todos os dados em uma determinada tabela; neste caso podemos utilizar a cláusula HAVING para filtrar os resultados dos grupos formados pela consulta:

```
1 SELECT status,
2     COUNT(*)
3 FROM orders
4 GROUP BY
5     status
6 HAVING
7     status <> 'Shipped';
```

	status	COUNT(*)
▶	Cancelled	6
	Disputed	3
	In Process	6
	On Hold	4
	Resolved	4

Utilizando o ORDER By

A palavra-chave ORDER BY é usada para ordenar o conjunto-resultado de registros em uma consulta SQL; esta ordenação pode ser feita de forma ascendente ou descendente; segue exemplo:

```

1 SELECT
2     contactLastname,
3     contactFirstname
4 FROM
5     customers
6 ORDER BY
7     contactLastname;

```

	contactLastname	contactFirstname
▶	Accorti	Paolo
	Altagar, G M	Raanan
	Andersen	Mel
	Anton	Carmen
	Ashworth	Rachel
	Barajas	Miguel
	Benitez	Violeta
	Bennett	Helen
	Berglund	Christina

No resultado acima verificamos que o **MySQL já utiliza a ordenação ASC por default**; podemos inverter este resultado conforme sintaxe abaixo:

```

1 SELECT
2     contactLastname,
3     contactFirstname
4 FROM
5     customers
6 ORDER BY
7     contactLastname DESC;

```

	contactLastname	contactFirstname
▶	Young	Jeff
	Young	Julie
	Young	Mary
	Young	Dorothy
	Yoshido	Juri
	Walker	Brydey
	Victorino	Wendy
	Urs	Braun
	Tseng	Jerry

Podemos ainda utilizar a ordenação em mais de uma coluna e com critérios diferentes; segue exemplo:

```

1 SELECT
2     contactLastname,
3     contactFirstname
4 FROM
5     customers
6 ORDER BY
7     contactLastname DESC,
8     contactFirstname ASC;

```

	contactLastname	contactFirstname
►	Young	Dorothy
	Young	Jeff
	Young	Julie
	Young	Mary
	Yoshido	Juri
	Walker	Brydey
	Victorino	Wendy
	Urs	Braun
	Tseng	Jerry

Exercícios com o GROUP By

Vamos iniciar executando uma consulta do campo *jobTitle* na tabela *employees*, onde o resultado será agrupado nos diferentes tipos de cargos que são utilizados nesta tabela:

```
SELECT jobTitle  
FROM employees  
GROUP BY jobTitle;
```

```
1 SELECT jobTitle  
2 FROM employees  
3 GROUP BY jobTitle;
```

▶ Execute

✍ Clear

✍ Beautify

✂ Minify

RESULT

jobTitle
President
Sale Manager (EMEA)
Sales Manager (APAC)
Sales Manager (NA)

Agora vamos ordenar o resultado de forma diferente:

```
SELECT jobTitle
FROM employees
GROUP BY jobTitle
ORDER BY jobTitle DESC;
```



```
1 SELECT jobTitle
2 FROM employees
3 GROUP BY jobTitle
4 ORDER BY jobTitle DESC;
5
```

Execute Clear Beautify Minify

RESULT

jobTitle
VP Sales
VP Marketing
Sales Rep
Sales Manager (NA)

Podemos ampliar a visão da consulta anterior utilizando uma função de agregação

```
SELECT jobTitle, COUNT(jobTitle)
FROM employees
GROUP BY jobTitle
ORDER BY jobTitle;
```

```
1 SELECT jobTitle, COUNT(jobTitle)
2 FROM employees
3 GROUP BY jobTitle
4 ORDER BY jobTitle;
5
6
```

▶ Execute

Clear

Beautify

Minify

RESULT

President	1
Sale Manager (EMEA)	1
Sales Manager (APAC)	1
Sales Manager (NA)	1
Sales Rep	17

Outra opção é a utilização da cláusula HAVING; no exemplo abaixo estamos agrupando os pagamentos que foram feitos pelo cliente '103' e calculando a média destes através da função de agregação AVG:

```
SELECT customerNumber, AVG(amount)
FROM payments
GROUP BY customerNumber
HAVING customerNumber = '103';
```

```
1 SELECT customerNumber, AVG(amount)
2 FROM payments
3 GROUP BY customerNumber
4 HAVING customerNumber = '103';
```

▶ Execute

Clear

Beautify

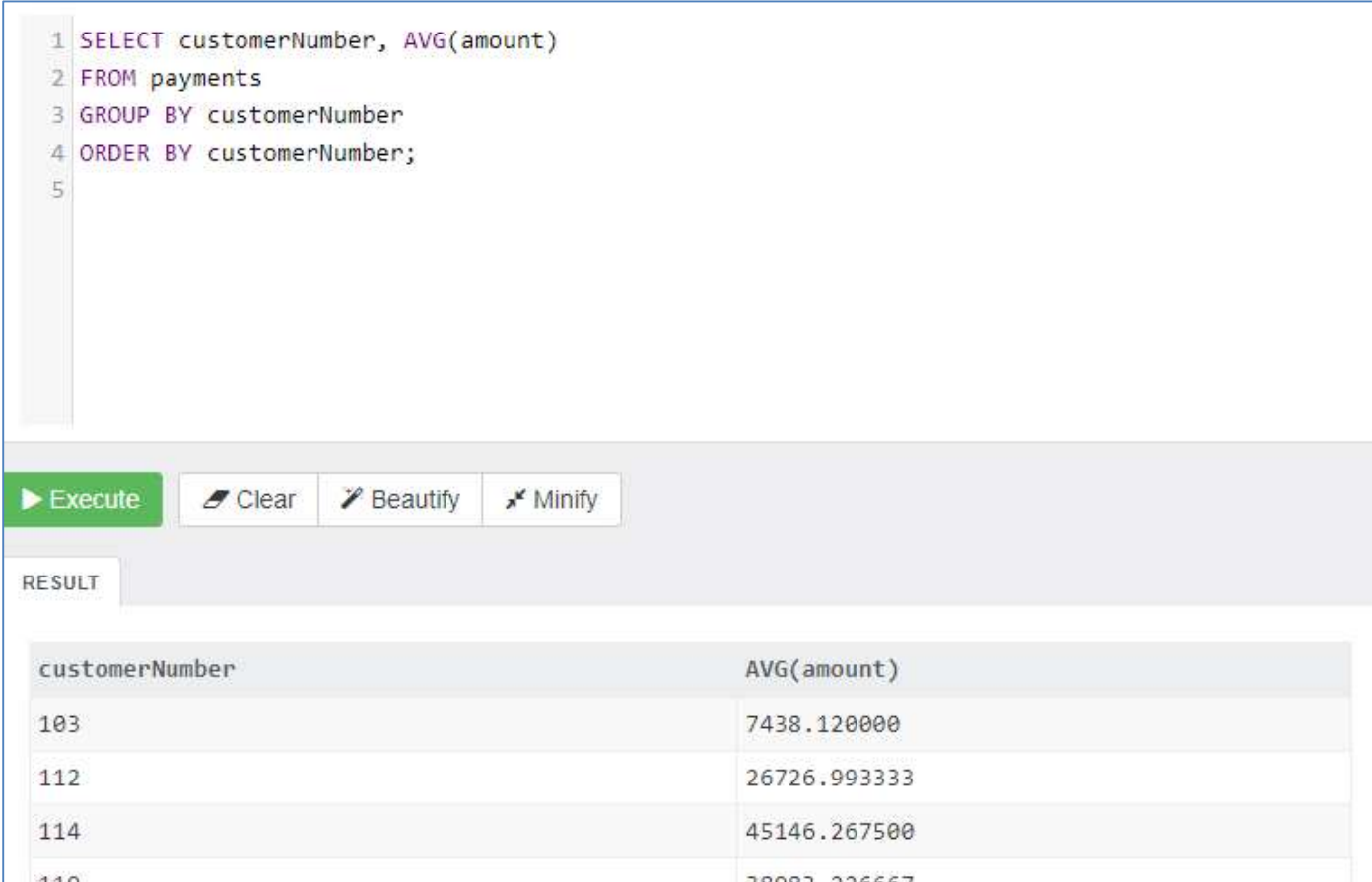
Minify

RESULT

customerNumber	AVG(amount)
103	7438.120000

Refinando a consulta anterior, vamos executar o mesmo procedimento, porém, para todos os clientes, ordenando seu código:

```
SELECT customerNumber, AVG(amount)
FROM payments
GROUP BY customerNumber
ORDER BY customerNumber;
```



The screenshot shows a SQL IDE interface. The top section contains a text editor with the following SQL query:

```
1 SELECT customerNumber, AVG(amount)
2 FROM payments
3 GROUP BY customerNumber
4 ORDER BY customerNumber;
5
```

Below the editor is a toolbar with buttons: "Execute" (green), "Clear", "Beautify", and "Minify".

Below the toolbar is a "RESULT" tab. The results are displayed in a table with two columns: "customerNumber" and "AVG(amount)".

customerNumber	AVG(amount)
103	7438.120000
112	26726.993333
114	45146.267500
119	38083.226667

Exercícios

- 1- Efetue uma consulta (SELECT) que retorne todas as filiais (OFFICES), ordenados pela cidade (CITY) onde estão situados.
- 2- Efetue uma consulta na tabela produtos (PRODUCTS), agrupando o resultado pela linha de produtos (PRODUCTLINE), excluindo os registros que são da linha PLANE.
- 3- Efetue uma consulta que retorne a quantidade de clientes por cidade (tabela CUSTOMERS, campo CITY); ordene este resultado em modo decrescente.