



## Cursos

➤ Listagem de disciplinas


Selecione uma disciplina

## Aulas

- 01 Introdu  o a Banco de Dados
- 02 Modelo de Entidade e Relacionamento
- 03 Modelo Relacional
- 04 Transforma  es ER para MR
- 05 Transforma  es ER para MR e dicion rio de dados
- 06 Normaliza  o b sica
- 07 Normaliza  o avan ada
- 08 Introdu  o   Linguagem SQL e Sistemas Gerenciadores de Banco de Dados
- 09 Linguagem SQL - cria  o, inser  o e modifica  o de tabelas
- 10 Linguagem SQL - Consulta simples de tabelas
- 11 Linguagem SQL - Consulta avan ada de tabelas
- 12 Linguagem SQL - Altera  o da estrutura de tabelas e ambientes de m ltiplas tabelas
- 13 Linguagem SQL - Subconsultas
- 14 Linguagem SQL - VIS  ES
- 15 Linguagem SQL - STORED PROCEDURES
- 16 Linguagem SQL - Fun  es
- 17 Linguagem SQL - Seguran a
- 18 Engenharia Reversa
- 19 Utilizando SQL em Java
- 20 Utilizando conceitos avan ados de SQL em Java

Voltar Imprimir Topo





## Objetivo

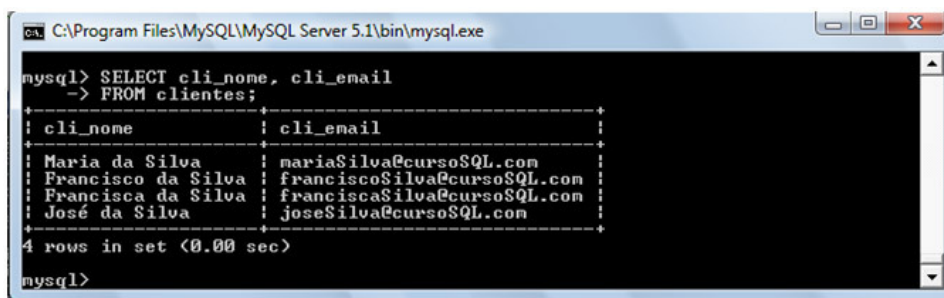
Ao final desta aula, você será capaz de:

- consultar dados ordenados em tabelas;
- consultar dados agrupados em tabelas;
- utilizar funções especiais para manipulação e apresentação das pesquisas.

## Consultas usando ordenação de dados

Na aula anterior, aprendemos a fazer consultas simples aos dados da tabela **clientes** e **filmes** do nosso banco de dados **locadora**. Por exemplo, para obter uma lista dos nomes dos clientes com seus respectivos emails, utilizamos o comando **SELECT**, conforme apresentado no quadro abaixo, obtém-se o resultado ilustrado na Figura 1.

```
mysql> SELECT cli_nome, cli_email  
FROM clientes;
```



```
C:\Program Files\MySQL\MySQL Server 5.1\bin\mysql.exe  
mysql> SELECT cli_nome, cli_email  
-> FROM clientes;  
+-----+-----+  
| cli_nome | cli_email |  
+-----+-----+  
| Maria da Silva | mariaSilva@cursoSQL.com |  
| Francisco da Silva | franciscoSilva@cursoSQL.com |  
| Francisca da Silva | franciscaSilva@cursoSQL.com |  
| José da Silva | joseSilva@cursoSQL.com |  
+-----+-----+  
4 rows in set (0.00 sec)  
mysql>
```

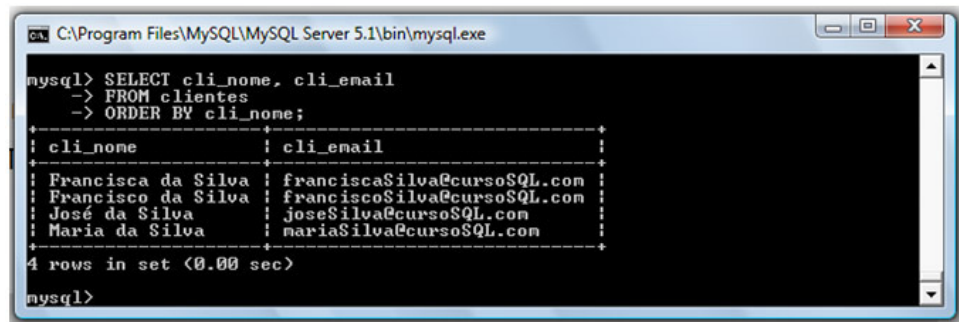
**Figura 1** – Tela do MySQL após o comando **SELECT cli\_nome, cli\_email FROM clientes**  
Fonte: MySQL Server 5.1

Observe que a coluna com o nome dos clientes não está em ordem alfabética, na verdade, o resultado é apresentado na ordem em que os dados são inseridos no banco de dados. Nesse caso, não temos dificuldades de achar o email da cliente Maria da Silva, por exemplo, pois temos apenas 4 linhas inseridas no banco de dados. Continuará sendo fácil se tivéssemos 100, 200 ou 300 registros? Claro que não, perderíamos muito tempo fazendo isso. Para facilitar nossa vida, o comando **SELECT** possui a cláusula **ORDER BY** que ordena os dados de uma consulta, facilitando a visualização dos resultados.

Continuando com o mesmo exemplo, para obter uma lista dos nomes dos clientes com seus respectivos emails, apresentando o resultado considerando o nome dos clientes em ordem alfabética, utilizamos o comando **SELECT**, conforme apresentado no quadro abaixo.

```
mysql> SELECT cli_nome, cli_email  
FROM clientes  
  
ORDER BY cli_nome;
```

Observe que agora as informações sobre os nomes e emails são apresentadas em ordem alfabética de acordo com o nome do cliente, conforme é ilustrado na Figura 2.



```
C:\Program Files\MySQL\MySQL Server 5.1\bin\mysql.exe

mysql> SELECT cli_nome, cli_email
-> FROM clientes
-> ORDER BY cli_nome;

+-----+-----+
| cli_nome | cli_email |
+-----+-----+
| Francisca da Silva | franciscaSilva@cursoSQL.com |
| Francisco da Silva | franciscoSilva@cursoSQL.com |
| José da Silva | joseSilva@cursoSQL.com |
| Maria da Silva | mariaSilva@cursoSQL.com |
+-----+-----+
4 rows in set (0.00 sec)

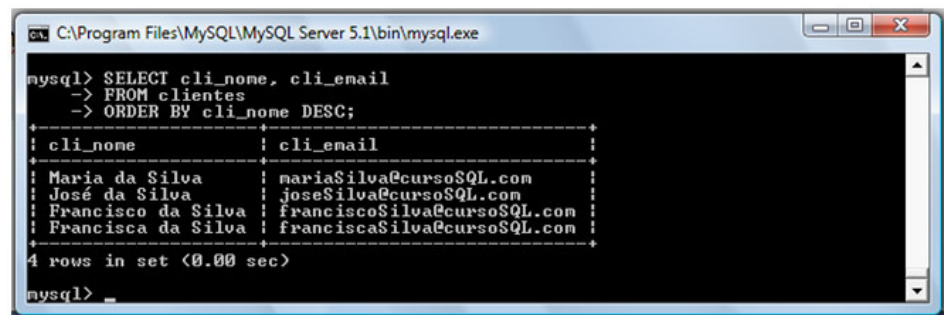
mysql>
```

**Figura 2** – Tela do MySQL após o comando `SELECT cli_nome, cli_email FROM clientes ORDER BY cli_nome`  
Fonte: MySQL Server 5.1

O MySQL sempre apresenta os resultados em ordem ascendente, ou seja, se tivermos tratando de dados de texto, na ordem de "A" a "Z". Se desejarmos visualizar em ordem inversa, adicionamos o termo DESC na cláusula ORDER BY, como apresentado no quadro abaixo.

```
mysql> SELECT cli_nome, cli_email
FROM clientes
ORDER BY cli_nome DESC;
```

Observe que agora as informações sobre os nomes e emails são apresentadas em ordem alfabética inversa de acordo com o nome do cliente, conforme é ilustrado na Figura 3.



```
C:\Program Files\MySQL\MySQL Server 5.1\bin\mysql.exe

mysql> SELECT cli_nome, cli_email
-> FROM clientes
-> ORDER BY cli_nome DESC;

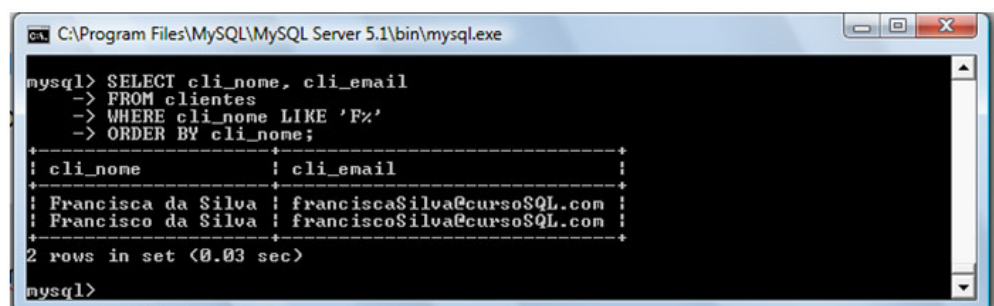
+-----+-----+
| cli_nome | cli_email |
+-----+-----+
| Maria da Silva | mariaSilva@cursoSQL.com |
| José da Silva | joseSilva@cursoSQL.com |
| Francisco da Silva | franciscoSilva@cursoSQL.com |
| Francisca da Silva | franciscaSilva@cursoSQL.com |
+-----+-----+
4 rows in set (0.00 sec)

mysql>
```

**Figura 3** – Tela do MySQL após o comando `SELECT cli_nome, cli_email FROM clientes ORDER BY cli_nome DESC`  
Fonte: MySQL Server 5.1

A cláusula ORDER BY pode ser utilizada em conjunto com a cláusula WHERE, de modo a visualizarmos os dados que atendam uma determinada condição de forma ordenada. Esse tipo de consulta deve seguir a sequência: `SELECT ... FROM ... WHERE ... ORDER BY ...`; por exemplo, se quisermos o nome e email dos clientes cujos nomes começam com a letra "F" de forma ordenada, utilizamos o SELECT como apresentado no quadro a seguir, obtendo o resultado ilustrado na Figura 4.

```
mysql> SELECT cli_nome, cli_email
FROM clientes
WHERE cli_nome LIKE 'F%'
ORDER BY cli_nome;
```



```
C:\Program Files\MySQL\MySQL Server 5.1\bin\mysql.exe

mysql> SELECT cli_nome, cli_email
-> FROM clientes
-> WHERE cli_nome LIKE 'F%'
-> ORDER BY cli_nome;

+-----+-----+
| cli_nome | cli_email |
+-----+-----+
| Francisca da Silva | franciscaSilva@cursoSQL.com |
| Francisco da Silva | franciscoSilva@cursoSQL.com |
+-----+-----+
2 rows in set (0.03 sec)

mysql>
```

**Figura 4** – Tela do MySQL após o comando `SELECT cli_nome, cli_email FROM clientes WHERE cli_nome LIKE 'F%' ORDER BY cli_nome`  
Fonte: MySQL Server 5.1

Além das funções de agregação discutidas na seção anterior, existem outras funções que podem ser usadas no comando SELECT. As principais são:

- funções matemáticas – Definição de operações matemáticas avançadas;
- funções de manipulação de *strings* – Funções usadas na manipulação de dados do tipo caractere;
- funções de data/hora – Funções usadas na manipulação de dados do tipo *DATE* e *TIME*.

Os quadros 1, 2 e 3 trazem um resumo de algumas funções de cada uma das categorias apresentadas na lista acima.

Função	Significado
ABS ( <i>valor</i> )	Retorna o valor absoluto (positivo) do valor informado
FLOOR ( <i>valor</i> )	Retorna o maior número inteiro, igual ou menor ao valor informado
ROUND ( <i>valor</i> , <i>n</i> )	Arredonda o valor informado para <i>n</i> casas decimais
POWER ( <i>valor</i> , <i>p</i> )	Retorna o valor informado elevado à potência <i>p</i>

**Quadro 1** – Exemplos de funções matemáticas

Função	Significado
LEN ( <i>expressão</i> )	Retorna o número de caracteres contidos na expressão informada
LOWER ( <i>expressão</i> ) e UPPER ( <i>expressão</i> )	Converte para minúsculo e maiúsculo a expressão informada, respectivamente
LTRIM ( <i>expressão</i> ) e RTRIM ( <i>expressão</i> )	Remove os espaços em branco à esquerda e à direita da expressão informada, respectivamente
SUBSTRING ( <i>expressão</i> , <i>início</i> , <i>tamanho</i> ):	Extraí uma parte dos caracteres da expressão, iniciando da posição informada em início, considerando a quantidade definida em tamanho

**Quadro 2** – Exemplos de funções de manipulação de *strings*

Função	Significado
CURDATE() e CURTIME()	Retorna a data e hora atuais, respectivamente
EXTRACT ( <i>parte</i> FROM <i>data</i> )	Retorna apenas a parte especificada de um campo de data/hora. A parte pode ser <i>year</i> , <i>month</i> , <i>day</i> , <i>hour</i> , <i>minute</i> , etc.
DATE FORMAT ( <i>data</i> , <i>formato</i> )	Retorna a data modificando seu formato de apresentação, que pode ser: %d para dia (0-31), %m para mês (0-12), %Y para ano com quatro dígitos, etc.

**Quadro 3** – Exemplos de funções de data/hora.

Para entendermos melhor a utilização dessas funções, vamos analisar os seguintes exemplos.

#### Exemplo 1

Mostrar o preço médio das locações considerando apenas duas casas decimais.

```
mysql> SELECT DISTINCT ROUND(AVG(fil_preco),2)
FROM filmes;
```

#### Exemplo 2

Mostrar em MAIÚSCULO os títulos de todos os filmes (sem repetição) que estão alugados.

```
mysql>SELECT DISTINCT UPPER(fil_titulo)
FROM filmes
WHERE fil_situacao='alugado';
```

### Exemplo 3

Mostrar o nome e ano de nascimento de todos os clientes da locadora do sexo feminino.

```
mysql>SELECT cli_nome, EXTRACT(year FROM cli_data_nasc)
FROM clientes
WHERE cli_sexo='F';
```

### Exemplo 4

Mostrar o nome e data de nascimento de todos os clientes da locadora no formato brasileiro dd/mm/aaaa em ordem alfabética do nome.

```
mysql>SELECT cli_nome, DATE_FORMAT(cli_data_nasc,'%d %m %Y')
FROM clientes
ORDER BY cli_nome;
```

Os resultados dessas pesquisas são ilustrados na Figura 9.

```

C:\Program Files\MySQL\MySQL Server 5.1\bin\mysql.exe
mysql> SELECT DISTINCT ROUND(fil_preco,0)
-> FROM filmes;
+-----+
| ROUND(fil_preco,0) |
+-----+
| 5 |
| 3 |
| NULL |
+-----+
3 rows in set (0.03 sec)

mysql> SELECT DISTINCT UPPER(fil_titulo)
-> FROM filmes
-> WHERE fil_situacao='alugado';
+-----+
| UPPER(fil_titulo) |
+-----+
| E O VENTO LEVOU |
| PROCURANDO NEMO |
+-----+
2 rows in set (0.00 sec)

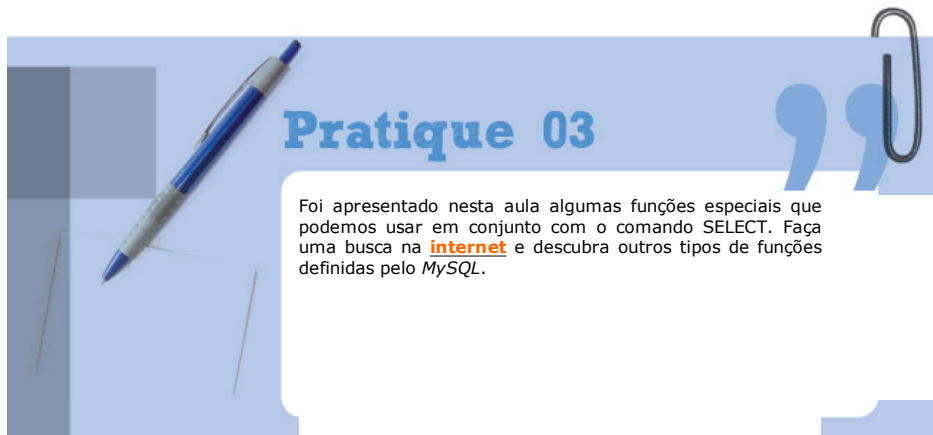
mysql> SELECT cli_nome, EXTRACT(year FROM cli_data_nasc)
-> FROM clientes
-> WHERE cli_sexo='F';
+-----+-----+
| cli_nome | EXTRACT(year FROM cli_data_nasc) |
+-----+-----+
| Maria da Silva | 1982 |
| Francisca da Silva | NULL |
+-----+-----+
2 rows in set (0.00 sec)

mysql> SELECT cli_nome, DATE_FORMAT(cli_data_nasc,'%d %m %Y')
-> FROM clientes
-> ORDER BY cli_nome;
+-----+-----+
| cli_nome | DATE_FORMAT(cli_data_nasc,'%d %m %Y') |
+-----+-----+
| Francisca da Silva | NULL |
| Francisco da Silva | 01 01 1990 |
| José da Silva | 10 12 1980 |
| Maria da Silva | 28 02 1982 |
+-----+-----+
4 rows in set (0.00 sec)

mysql>

```

**Figura 9** – Tela do MySQL após diversas pesquisas com o comando SELECT, usando funções especiais  
Fonte: MySQL Server 5.1



## Pratique 03

Foi apresentado nesta aula algumas funções especiais que podemos usar em conjunto com o comando SELECT. Faça uma busca na [internet](#) e descubra outros tipos de funções definidas pelo *MySQL*.

Encerramos por aqui nossa quarta aula sobre a linguagem SQL. Na próxima aula, aprenderemos a modificar a estrutura de uma tabela, criando novas colunas ou redefinindo colunas já existentes e mostraremos como usar a linguagem SQL em banco de dados com múltiplas tabelas. Você deve ter observado que o comando SELECT é muito poderoso e possui diversas formas de uso. Para ajudar na fixação do conteúdo, pegue seu caderno e faça um resumo do que já foi estudado sobre ele e exercite bastante para não esquecer. Lembre-se de fazer sua autoavaliação. Bons estudos e boa sorte!

### Resumo

Nesta aula, continuamos a estudar o comando SELECT e apresentamos as cláusulas ORDER BY e GROUP BY que são utilizadas para ordenar e agrupar os dados pesquisados, respectivamente. Em relação ao agrupamento, podemos refinar os resultados agrupados usando a cláusula HAVING. Vimos que as funções de agregação COUNT(), SUM(), AVG(), MIN() e MAX() nos ajudam a obter novas informações sobre nossos dados, sobretudo, quando unidas ao GROUP BY. Conhecemos cláusulas que nos ajudam a melhorar a apresentação dos nossos resultados como DISTINCT e AS. Também vimos diversas funções especiais que tratam tipos numéricos, caracteres e data/hora que podem nos auxiliar a fazer pesquisas mais complexas ou melhorar a apresentação dos nossos resultados.



## Evolucao

Considere o banco de dados CursoX criado nas aulas anteriores cuja estrutura de tabelas é mostrada abaixo:

**TABELA alunos**

ATRIBUTO	TIPO	DESCRIÇÃO
aluno_cod	Número inteiro	Código do aluno
aluno_nome	Alfanumérico	Nome do aluno
aluno_endereco	Alfanumérico	Endereço do aluno
aluno_cidade	Alfanumérico	Cidade do aluno

#### TABELA disciplina

ATRIBUTO	TIPO	DESCRIÇÃO
dis_cod	Número inteiro	Código da disciplina
dis_nome	Alfanumérico	Nome da disciplina
dis_carga	Número inteiro	Carga horária da disciplina
dis_professor	Alfanumérico	Professor da disciplina

**TABELA professores**

ATRIBUTO	TIPO	DESCRIÇÃO
prof_cod	Número inteiro	Código do professor
prof_nome	Alfanumérico	Nome do professor
prof_endereco	Alfanumérico	Endereço do professor
prof_cidade	Alfanumérico	Cidade do professor

Resolva as consultas abaixo utilizando a linguagem SQL.

1. Exiba as diferentes cidades em que moram os alunos e as respectivas quantidades de alunos em cada uma.
2. Refaça a consulta do item a) para os professores, ordenando o resultado em ordem alfabética de acordo com o nome da cidade e renomeando as colunas de resultados.
3. Exiba a quantidade total de disciplinas oferecidas.
4. Exiba a carga horária total para cada professor.
5. Exiba a carga horária total de cada professor que possua mais de uma disciplina.
6. Exiba o nome de todas as disciplinas ofertadas (sem repetição de nome) em MAIÚSCULO.
7. Exiba o nome de todos os professores cujos nomes iniciem com as letras de "A" a "J" em ordem alfabética.
8. Calcule a carga horária média por professor (dica: Você pode fazer mais de uma consulta para obter esse resultado).

## Referencias

BEIGHLEY, L. **Use a cabeça SQL**. Rio de Janeiro: Editora AltaBooks, 2008.

MySQL 5.1 Reference Manual. Disponível em: <<http://dev.mysql.com/doc/refman/5.1/en/>>. Acesso em: 24 set. 2010.

[Voltar](#)[Imprimir](#)[Topo](#)