Prof. Gean Paulo

Disciplina Banco de Dados

Tutorial MySQL

Aula 6: MySQL – INSERT INTO: Inserir Dados em Tabelas

Uma das operações mais importantes em um banco de dados é a inserção de registros (dados). Fazemos isso com o uso do comando **INSERT INTO**.

Sintaxe padrão:

```
INSERT INTO nome_tabela (coluna1, coluna2,...)
VALUES (valor1, valor2,...);
```

De acordo com a sintaxe apresentada, devemos especificar a tabela e quais colunas dessa tabela receberão os dados, e em seguida, logo após a palavra-chave **VALUES**, especificamos os dados em si – na mesma ordem em que as colunas foram especificadas.

Caso haja uma coluna com auto incremento, ela não deve ser incluída na lista de colunas do comando, pois seus dados serão gerados e inseridos automaticamente pelo MySQL quando um novo registro for adicionado.

Exemplos no banco de dados db_biblioteca:

Vamos inserir dados de quatro editoras na tabela tbl_editoras:

```
INSERT INTO tbl_editoras (Nome_Editora) VALUES ('Prentice Hall');
INSERT INTO tbl_editoras (Nome_Editora) VALUES ('O´Reilly');
INSERT INTO tbl_editoras (Nome_Editora) VALUES ('Microsoft Press');
INSERT INTO tbl_editoras (Nome_Editora) VALUES ('Wiley');
INSERT INTO tbl_editoras (Nome_Editora) VALUES ('McGraw-Hill Education');
```

Agora, vamos inserir dados na tabela de autores. No exemplo anterior executamos o comando INSERT INTO quatro vezes. mas é possível executá-lo apenas uma vez e inserir múltiplos registros, como mostra o código abaixo, que irá inserir 12 autores de uma vez na tabela tbl_autores:

```
INSERT INTO tbl_autores
VALUES
(1, 'Daniel', 'Barret'),
(2, 'Gerald', 'Carter'),
(3, 'Mark', 'Sobell'),
(4, 'William', 'Stanek'),
(5, 'Richard', 'Blum'),
(6, 'Jostein', 'Gaarder'),
(7, 'Umberto', 'Eco'),
(8, 'Neil', 'De Grasse Tyson'),
(9, 'Stephen', 'Hawking'),
(10, 'Stephen', 'Jay Gould'),
(11, 'Charles', 'Darwin'),
(12, 'Alan', 'Turing'),
(13, 'Simon', 'Monk'),
```

```
(14, 'Paul', 'Scherz');
```

Note que nesse caso, como inserimos dados em todas as colunas da tabela, e na mesma ordem em que elas foram criadas, não foi necessário especificar essas colunas no comando.

Vamos inserir dados na tabela de categorias:

```
INSERT INTO tbl_categorias
VALUES
(1, 'Tecnologia'),
(2, 'História'),
(3, 'Literatura'),
(4, 'Astronomia'),
(5, 'Botânica');
```

Agora, para finalizar, vamos inserir dados na tabela de livros (tbl_livro):

```
INSERT INTO tbl_Livro (Nome_Livro, ISBN13, ISBN10, Data_Pub, Preco_Livro, Categoria, ID_Autor, ID_Editora)
VALUES
('Linux Command Line and Shell Scripting','9781118983843', '111898384X', '20150109', 68.35, 1, 5, 4),
('SSH, the Secure Shell','9780596008956', '0596008953', '20050517', 58.30, 1, 1, 2),
('Using Samba','9780596002565', '0596002564', '20031221', 61.45, 1, 2, 2),
('Fedora and Red Hat Linux', '9780133477436', '0133477436', '20140110', 62.24, 1, 3, 1),
('Windows Server 2012 Inside Out','9780735666313', '0735666318', '20130125', 66.80, 1, 4, 3),
('Microsoft Exchange Server 2010','9780735640610', '0735640610', '20101201', 45.30, 1, 4, 3),
('Practical Electronics for Inventors', '9781259587542', '1259587541', '20160324', 67.80, 1, 13, 5);
```

Neste último exemplo, uma das colunas não recebeu dados inseridos pelo comando (ID_livro é auto-incrementado), então especificamos as colunas que receberiam dados.

Testando a inserção de dados

Agora vamos executar os comandos abaixo (consultas simples com SELECT) para verificar se os registros foram inseridos corretamente em cada tabela no MySQL:

```
SELECT * FROM tbl_autores;
SELECT * FROM tbl_editoras;
SELECT * FROM tbl_livro;
```

Veja o resultado da inserção de dados na tabela de livros, verificado com o comando executado acima:

ID_Livro	Nome_Livro	ISBN	ID_Autor	Data_Pub	Preco_Livro	ID_editora
1	Linux Command Line and Shell Scripting	143856969	5	2009-12-21	68.35	4
2	SSH, the Secure Shell	127658789	1	2009-12-21	58.30	2
3	Using Samba	123856789	2	2000-12-21	61.45	2
4	Fedora and Red Hat Linux	123346789	3	2010-11-01	62.24	1
5	Windows Server 2012 Inside Out	123356789	4	2004-05-17	66.80	3
6	Microsoft Exchange Server 2010	123366789	4	2000-12-21	45.30	3

Aula 7- MySQL – SELECT – Realizar Consultas simples em Tabelas

Podemos dizer que a tarefa mais importante que um banco de dados MySQL realiza, do ponto de vista do usuário, é permitir a realização de consultas aos dados armazenados. O comando básico que utilizamos para realizar as consultas é o comando SELECT.

Comando SELECT

Sintaxe:

```
SELECT coluna(s) FROM Tabela;
```

Exemplos:

Consultar nomes dos autores cadastrados na tabela de autores:

```
SELECT Nome_Autor FROM tbl_Autores;
```

Consultar todos os dados da tabela de autores (usamoso caractere curinga * para denotar todos os campos):

```
SELECT * FROM tbl_Autores;
```

Consultar os nomes dos livros (apenas) na tabela de livros:

```
SELECT Nome_Livro FROM tbl_livro;
```

Consultar nomes de livros e IDs de autores respectivos na tabela de livros:

```
SELECT Nome_Livro, ID_Autor FROM tbl_livro;
```

Consultar nomes de livros e seus ISBNs na tabela de livros:

```
SELECT Nome_Livro, ISBN
FROM tbl_livro;
```

Aula 8 - MySQL – ORDER BY – Consultas com ordenação

A palavra-chave **ORDER BY** é usada para ordenar o conjunto-resultado de registros em um consulta SQL, tanto de forma ascendente quanto descendente.

Sintaxe:

```
SELECT colunas FROM tabela
ORDER BY coluna_a_ordenar;
```

Podemos configurar a ordenação como ascendente (crescente) ou descendente (decrescente) com o uso das palavras ASC ou DESC:

- **ASC** – Ordem ascendente
- **DESC** – Ordem descendente (inversa)

Ordem de Classificação por Tipo de Dados

A ordem padrão de classificação do ORDER BY (ASC) para os diversos tipos de dados é a seguinte:

- Valores numéricos são exibidos com os menores valores primeiro (do menor para o maior).
- Valores de data são mostrados com os valores menores primeiro, o que significa as datas mais antigas (do mais antigo para o mais recente).
- Valores de caracteres são exibidos em ordem alfabética.
- Quando houver valores nulos (NULL), eles serão mostrados por último (em sequências descendentes, com DESC, são mostrados primeiro

É possível classificar os dados da consulta usando uma coluna que não esteja presente na lista de colunas da declaração SELECT.

Também é possível ordenar os resultados da consulta usando mais de uma coluna. Para isso, basta listar as colunas na cláusula ORDER BY, separadas por vírgulas. Os resultados serão ordenados pela primeira coluna na lista, e então pela segunda coluna, e assim por diante.
Pode-se ordenar qualquer das colunas listadas em qualquer ordem, por exemplo colunas em ordem ascendente e colunas em ordem descendente, bastando para isso suceder o nome da coluna que será classificada em ordem inversa com a palavra-chave DESC.

Exemplos:

```
SELECT * FROM tbl_Livro
ORDER BY Nome_Livro ASC;
```

ID_Livro	Nome_Livro 1	ISBN	ID_Autor	Data_Pub	Preco_Livro	ID_editora
7	Enciclopédia de Componentes Eletrônicos vol. 03	153642397	13	2016-05-05	63.39	5
4	Fedora and Red Hat Linux	123346789	3	2010-11-01	62.24	1
1	Linux Command Line and Shell Scripting	143856969	5	2009-12-21	68.35	4
6	Microsoft Exchange Server 2010	123366789	4	2000-12-21	45.30	3
2	SSH, the Secure Shell	127658789	1	2009-12-21	58.30	2
3	Using Samba	123856789	2	2000-12-21	61.45	2
5	Windows Server 2012 Inside Out	123356789	4	2004-05-17	66.80	3

```
SELECT Nome_Livro, ID_Editora
FROM tbl_Livro
ORDER BY ID_Editora; -- (ordem crescente)
```

Nome_Livro	ID_Editora ▲ 1
Fedora and Red Hat Linux	1
SSH, the Secure Shell	2
Using Samba	2
Windows Server 2012 Inside Out	3
Microsoft Exchange Server 2010	3
Linux Command Line and Shell Scripting	4
Enciclopédia de Componentes Eletrônicos vol. 03	5

```
SELECT Nome_Livro, Preco_Livro
FROM tbl_Livro
ORDER BY Preco_Livro DESC;
-- (ordem decrescente)
```

Nome_Livro	Preco_Livro ▼ 1
Linux Command Line and Shell Scripting	68.35
Windows Server 2012 Inside Out	66.80
Enciclopédia de Componentes Eletrônicos vol. 03	63.39
Fedora and Red Hat Linux	62.24
Using Samba	61.45
SSH, the Secure Shell	58.30
Microsoft Exchange Server 2010	45.30

Aula 9 - MySQL – Clausula WHERE – Filtrar resultados de consultas

A cláusula **WHERE** permite filtrar registros nos resultados de uma consulta, de modo a trazer apenas as informações desejadas (e não o conteúdo completo das colunas).

Sintaxe:

```
SELECT colunas
FROM tabela
WHERE coluna = valor;
```

Exemplos:

1. Retornar o nome e a data de publicação do livro cujo ID do autor é 1:

```
SELECT Nome_Livro, Data_Pub
FROM tbl_Livro
WHERE ID_Autor = 1;
```

Nome_Livro	Data_Pub
SSH, the Secure Shell	2009-12-21

Foi retornado apenas um registro, do livro cujo autor possui o ID especificado na cláusula WHERE.

2. Trazer o ID e Nome do autor do autor cujo sobrenome é Stanek:

```
SELECT ID_Autor, Nome_Autor
FROM tbl_autores
WHERE Sobrenome_Autor = 'Stanek';
```

ID_Autor	Nome_Autor
4	William

Foi retornado o nome do autor cujo sobrenome é Stanek: William Stanek.

3. Mostrar os nomes e os preços dos livros publicados pela editora de ID igual a 3:

```
SELECT Nome_Livro, Preco_Livro
FROM tbl_livro
WHERE ID_Editora = 3;
```

Nome_Livro	Preco_Livro
Windows Server 2012 Inside Out	66.80
Microsoft Exchange Server 2010	45.30

Neste caso foram retornados dois livros, publicados pela editora de ID indicado.