

Banco de Dados

Fabrício Curvello Gomes



Criando um Banco de Dados No MySQL

  INFORMA, FORMA, TRANSFORMA.

MySQL Server



Iremos utilizar o MySQL Server como ferramenta de estudo de banco de dados.



Detalhes de como obter e instalar o MySQL Server estão explicados no arquivo em PDF:

- BD - TI - Extra 01 - Instalação e Configuração do MySQL 5.1

 3

  INFORMA, FORMA, TRANSFORMA.

Entenda que...

A partir de agora iremos interagir com o MySQL através de comandos. A visualização de resultados não é muito clara.

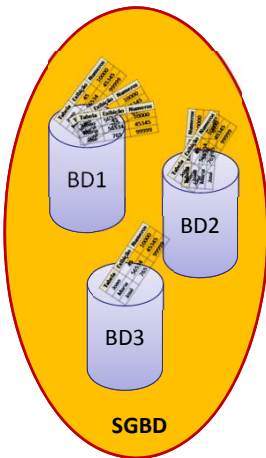
Então, antes de começarmos, lembre-se que:


Um SGBD é um programa que gerencia Banco de Dados (No caso, o MySQL).



Isso quer dizer que um SGBD pode gerenciar um ou mais (muitos) bancos de dados.

Um banco de dados possui tabelas com informações.

Logo, em um SGBD, podem existir diversos BDs, cada um com diversas tabelas, cada tabela com diversas informações.



 4

  INFORMA, FORMA, TRANSFORMA.


MySQL Command Line Client



Considerando que o MySQL Server já esteja instalado, e que você saiba a senha de administrador, vamos começar a trabalhar no MySQL.

OBS: Nos laboratórios, a senha de administrador do MySQL é **alunolab**

Inicie o **MySQL Command Line Client**, da seguinte forma:

Iniciar / Todos os Programas / MySQL / MySQL Server 5.1 / MySQL Command Line Client

 5


  INFORMA, FORMA, TRANSFORMA.



MySQL Command Line Client (Cont.)

Feito isso, será aberta uma janela preta, semelhante ao prompt do DOS, com a mensagem “Enter Password:”

Digite a senha de Administrador que foi utilizada durante a instalação e pressione <Enter>. Este procedimento deverá ser feito sempre que o *MySQL Command Line Client* for executado.

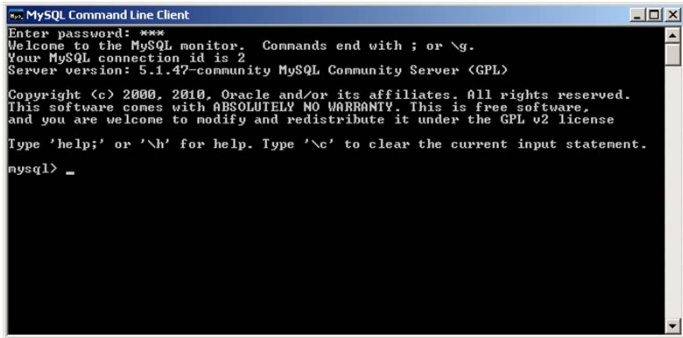
Lembrando mais uma vez, que a senha nos laboratórios é **alunolab**.


 6



  INFORMA, FORMA, TRANSFORMA.

MySQL Command Line Client (Cont.)

Se você digitou corretamente a senha, o cursor agora estará pronto, aguardando novos comandos, da seguinte forma:

mysql> 

 7

  INFORMA, FORMA, TRANSFORMA.

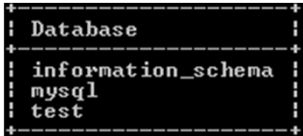
Visualizando os Bancos de Dados


Vamos então trabalhar com os primeiros comandos no prompt do MySQL.



Para visualizar os bancos de dados existentes no MySQL, utilize o comando a seguir:

SHOW DATABASES;

Será exibido a lista de Bancos de Dados existentes.

Ex: 

 8


  **Sistema FIRJAN** | **INFORMA, FORMA, TRANSFORMA.**



Criando um Banco de Dados

Para criar um Banco de Dados, utilize o seguinte comando:

```
CREATE DATABASE xxxxx ;
```

xxxxx é o nome que você dará ao BD. Não pode ter espaço nem caracteres especiais.

 9

  **Sistema FIRJAN** | **INFORMA, FORMA, TRANSFORMA.**


Excluindo um Banco de Dados



Para excluir um Banco de Dados, utilize o seguinte comando:

```
DROP DATABASE xxxxx ;
```

xxxxx é o nome do BD que você quer excluir. É interessante visualizar os BDs existentes antes e depois de excluir algum.

IMPORTANTE: Toda a estrutura e dados contida no BD em questão será excluída de forma definitiva e irreversível. Por isso, certifique-se de estar excluindo o banco de dados que realmente deseja deletar.

 10


  INFORMA, FORMA, TRANSFORMA.

Ativando um Banco de Dados



Para ativar um BD, ou seja, para indicar ao MySQL com qual dos BDs existentes você pretende trabalhar, utilize o seguinte comando:

```
USE xxxxx ;
```

xxxxx é o nome do BD que você quer ativar. Obviamente que este BD precisa existir antes de ser ativado.



11

  INFORMA, FORMA, TRANSFORMA.

Criando Uma Tabela

Para criar uma tabela, um BD precisa estar ativo.


Utilize o seguinte comando:

```
CREATE TABLE xxxxx (<colunas>);
```

xxxxx é o nome da tabela que você está criando. Este nome não deve ter espaços em branco, e deve ser inferior a 64 caracteres.

Aqui devem ser declarados os nomes de colunas da tabela, e cada domínio (tipo) de cada coluna, separados por vírgula. Ex:

```
CREATE TABLE ALUNOS (MATRICULA INT NOT NULL, NOME VARCHAR(64) NOT NULL, PRIMARY KEY (MATRICULA));
```



12

Principais Domínios (Tipos):

- **AUTO_INCREMENT** – É usado para gerar um valor único sequencial para um novo registro. Só é possível ter uma coluna `auto_increment` na tabela, e esta coluna tem que ser chave primária. Um exemplo de criação de uma tabela com este recurso seria assim: `create table nometabela (nomecoluna int not null auto_increment primary key, ...);`
- **BIGINT[(tamanho)] [UNSIGNED] [ZEROFILL]** – Utiliza-se esse tipo de dado quando usar valores inteiros grandes na faixa de -9.223.372.036.854.808 a 9.223.372.036.854.775.807. O parâmetro `tamanho` é opcional e estabelece o tamanho máximo do valor a ser exibido no monitor de vídeo, podendo ser um valor máximo 255. O parâmetro `UNSIGNED`, quando usado, estabelece que o valor definido será positivo, ou seja, podem ser utilizados valores na faixa 0 a 18.446.744.073.709.551.615. O parâmetro `ZEROFILL` estabelece a definição do preenchimento com valor zero caso o campo (coluna) seja deixado em branco.
- **CHAR(tamanho)** ou **CHARACTER(tamanho)** – Usado com sequências de caracteres de tamanho fixo que estejam limitadas a 255 caracteres de comprimento. O parâmetro `tamanho` determina o valor máximo em caracteres que pode conter a sequência. Esse tipo de dado, quando definido, preenche o campo com espaços em branco até completar o total de caracteres definidos, quando a totalidade do tamanho do campo não é preenchida.
- **DATE** – Utilizado com uma data de calendário no formato AAAA-MM-DD (Formato ANSI) dentro do intervalo de tempo entre 1000-01-01 e 9999-12-31.
- **DECIMAL[(tamanho, decimal)] [UNSIGNED] [ZEROFILL]** – Quando for preciso usar valores com ponto flutuante como se fossem uma sequência de caracteres do tipo `CHAR`. O parâmetro `decimal` determina o número de casas decimais. O parâmetro `UNSIGNED`, quando usado, estabelece que o valor definido será positivo. O parâmetro `ZEROFILL` estabelece a definição do preenchimento com valor zero caso o campo (coluna) seja deixado em branco.
- **DOUBLE[(tamanho, decimal)] [UNSIGNED] [ZEROFILL]** – Quando for necessário usar valores com tamanho normal (dupla precisão) na faixa de valores -1.7976931348623157E+308 e -2.2250738585072014E-308, 0 e entre 2.2250738585072014E-308 e 1.7976931348623157E+308. O parâmetro `decimal` determina o número de casas decimais. O parâmetro `UNSIGNED`, quando usado, estabelece que o valor definido será positivo. O parâmetro `ZEROFILL` estabelece a definição do preenchimento com valor zero caso o campo (coluna) seja deixado em branco.

Principais Domínios (Tipos): (Cont.)

- **ENUM ('valor1', 'valor2', ... , 'valorN')** – Usado com uma lista de valores do tipo `string`, em que um dos valores pode ser selecionado. O campo (coluna) do tipo `ENUM` pode possuir até no máximo 65535 valores diferentes.
- **FLOAT[(tamanho, decimal)] [UNSIGNED] [ZEROFILL]** – Quando usar valores com ponto flutuante pequenos (precisão simples) na faixa de valores -3.402823466E+38 a -1.175494351E-38, 0, e entre 1.175494351E-38 e 3.402823466E+38. O parâmetro `tamanho` é opcional e permite estabelecer o tamanho máximo a ser exibido no monitor de vídeo, podendo ser um valor máximo 255. O parâmetro `decimal` determina o número de casas decimais. O parâmetro `UNSIGNED`, quando usado, estabelece que o valor definido será positivo. O parâmetro `ZEROFILL` estabelece a definição do preenchimento com valor zero caso o campo (coluna) seja deixado em branco.
- **INTEGER[(tamanho)] [UNSIGNED] [ZEROFILL]** ou **INT[(tamanho)] [UNSIGNED] [ZEROFILL]** – utilizado quando houver a necessidade de usar valores inteiros longos, na faixa de -2.147.483.648 até 2.147.483.647. O parâmetro `tamanho` é opcional e permite estabelecer o tamanho máximo a ser exibido no monitor de vídeo, podendo ser um valor máximo 255. O parâmetro `UNSIGNED`, quando usado, estabelece que o valor definido será positivo. O parâmetro `ZEROFILL` estabelece a definição do preenchimento com valor zero caso o campo (coluna) seja deixado em branco.
- **SMALLINT[(tamanho)] [UNSIGNED] [ZEROFILL]** – Quando for necessário usar valores inteiros curtos na faixa de -32.768 até 32.767. O parâmetro `tamanho` é opcional e permite estabelecer o tamanho máximo a ser exibido no monitor de vídeo, podendo ser um valor máximo 255. O parâmetro `UNSIGNED`, quando usado, estabelece que o valor definido será positivo. O parâmetro `ZEROFILL` estabelece a definição do preenchimento com valor zero caso o campo (coluna) seja deixado em branco.
- **TIME** – Utiliza-se esse tipo de dado quando houver necessidade de usar informação relacionada a um determinado horário de relógio no intervalo de tempo entre '-838:59:59' e '838:59:59'. Os valores de hora do MySQL são mostrados no formato 'HH:MM:SS', no entanto, é possível atribuir valores de colunas (campos) usando strings ou números.



Principais Domínios (Tipos): (Cont.)

- **TINYINT**(*tamanho*) [**UNSIGNED**] [**ZEROFILL**] – Quando precisar de valores inteiros pequenos na faixa de -128 até 127. O parâmetro tamanho é opcional e permite estabelecer o tamanho máximo a ser exibido no monitor de vídeo, podendo ser um valor máximo 255. O parâmetro UNSIGNED, quando usado, estabelece que o valor definido será positivo. O parâmetro ZEROFILL estabelece a definição do preenchimento com valor zero caso o campo (coluna) seja deixado em branco.
- **VARCHAR**(*tamanho*) – Quando precisa de sequências de caracteres de tamanho variável que estejam limitadas a 255 caracteres de comprimento. A diferença entre esse tipo e o CHAR é que, neste caso, os espaços em branco excedentes do lado direito da sequência de caracteres não utilizados são automaticamente desprezados. O parâmetro tamanho determina o valor máximo da sequência em caracteres.



15

Sistema
FIRJAN



INFORMA, FORMA, TRANSFORMA.

Visualizando Tabelas



Se você fez os passos anteriores sem nenhum erro, utilize agora o seguinte comando para visualizar todas as tabelas contidas no BD ativo:

SHOW TABLES;

Mostra quais as tabelas que estão contidas no BD ativo. **Não mostra a estrutura de uma tabela.**



16

  INFORMA, FORMA, TRANSFORMA.


Visualizando A Estrutura De Uma Tabela



A sintaxe é a seguinte:

```
DESCRIBE xxxxx [coluna];
```

xxxxx é o nome da tabela

Aqui é opcional, caso a intenção seja visualizar apenas uma coluna. Deve ser indicado o nome da coluna, sem os colchetes.

 17

  INFORMA, FORMA, TRANSFORMA.

Excluindo Uma Tabela


Para excluir uma tabela, um BD precisa estar ativo.

Utilize o seguinte comando:

```
DROP TABLE xxxxx ;
```

xxxxx é o nome da tabela que você está excluindo.

IMPORTANTE: A tabela em questão será excluída de forma definitiva e irreversível. Por isso, certifique-se de estar excluindo a tabela que realmente deseja deletar. Se esta tabela já contiver registros inseridos, estes também serão perdidos.

 18


Sistema FIRJAN | **INFORMA, FORMA, TRANSFORMA.**

Exemplo para Teste de Aprendizagem:

- 1 - Crie e ative um BD chamado **empresa**.
- 2 - Crie uma tabela chamada **cliente**, com as seguintes características:

Field	Type	Null	Key	Default	Extra
matricula	int(11)	NO	PRI	NULL	auto_increment
nome	varchar(60)	NO		NULL	
telefone	char(13)	YES		NULL	
email	varchar(30)	YES		NULL	
nascimento	date	NO		NULL	

- 3 – Avalie a se a estrutura da tabela ficou exatamente igual ao exemplo acima, pelo comando **describe cliente;**
- 4 – Se não ficou exatamente igual, apague a tabela e refaça.

 19


Sistema FIRJAN | **INFORMA, FORMA, TRANSFORMA.**

Exemplo para Teste (Cont.):

Field	Type	Null	Key	Default	Extra
matricula	int(11)	NO	PRI	NULL	auto_increment
nome	varchar(60)	NO		NULL	
telefone	char(13)	YES		NULL	
email	varchar(30)	YES		NULL	
nascimento	date	NO		NULL	

O comando correto para criar a tabela conforme o modelo é:

```
create table cliente
  (matricula int primary key auto_increment,
nome varchar(60) not null,
telefone char(13),
email varchar(30),
nascimento date not null);
```


 20

Sistema FIRJAN | **INFORMA, FORMA, TRANSFORMA.**

Inserindo Registros Em Tabela

A partir do momento em que uma tabela está pronta, ela já pode receber a entrada de dados, que podem ser enviados de duas formas:

- 1 - Forma direta – Instrução **INSERT INTO**
- 2 - Forma indireta – Instrução **LOAD DATA**

 21

Sistema FIRJAN | **INFORMA, FORMA, TRANSFORMA.**

1 - Forma direta – INSERT INTO

Sintaxe para a instrução **INSERT INTO**:


```
INSERT INTO <tabela>
(Campo1,
 Campo2,
 CampoN)
VALUES
Valor1,
Valor2,
ValorN);
```

Exemplo para a tabela que estamos trabalhando:

```
INSERT INTO cliente
(matricula,
 nome,
 telefone,
 email,
 nascimento)
VALUES
(null,
'Ayrton Bueno',
'(21)8511-9876',
'abueno@email.com',
'1990-10-25');
```

Repita esta instrução, alterando os dados, para catalogar outros clientes nesta tabela.

opcional





1 - Forma direta – INSERT INTO (Cont.)

Exercício:

Insira agora na tabela Cliente, 10 registros diferentes pela forma direta, sendo que:

- 3 pessoas possuem exatamente o mesmo número de telefone
- Outras 2 pessoas possuem a mesma data de nascimento

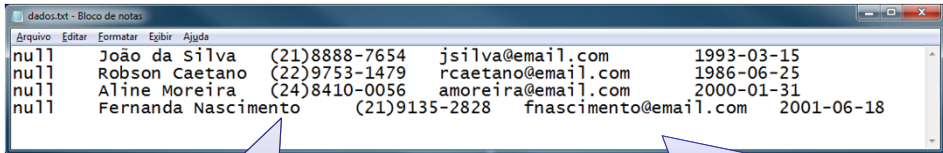
23

2 - Forma indireta – LOAD DATA

A instrução **LOAD DATA** tenta facilitar a inserção de dados em tabelas. Para isso faz-se necessário a utilização do **Bloco de Notas** (ou outro editor de texto puro).

Abra o bloco de notas e insira os dados formando as colunas da tabela. **Cada coluna fica separada da outra usando-se a tecla <tab> do teclado.**

Exemplo de acordo com a tabela que estamos trabalhando:



Salve o arquivo com a extensão .txt, dentro da pasta bin em C:\Arquivos de programas\MySQL\MySQL Server 5.1\bin\

Não se importe com a aparência do texto, e sim, com a **separação de dados APENAS pela tecla <tab>**

24


2. Forma indireta – LOAD DATA (Cont.)

Com o arquivo txt salvo dentro da pasta bin, que fica em:
C:\Arquivos de programas\MySQL\MySQL Server 5.1\bin
 Insira então a seguinte instrução:

```
LOAD DATA LOCAL INFILE "dados.txt" INTO TABLE cliente;
```

Nome do arquivo txt
que você criou com os
dados

Nome da tabela onde
os dados do arquivo txt
serão inseridos

 25

OBSERVAÇÃO

Como foi utilizado nos dois slides anteriores, se uma tabela possui um campo (coluna) que seja de auto incremento, você precisará, no momento de inserir dados nesta tabela, informar para este campo o valor **null**.

Mais um exemplo:


Se você criou uma tabela assim:



```
CREATE TABLE cadastro (CODIGO INT(5) NOT NULL  
AUTO_INCREMENT PRIMARY KEY, NOME VARCHAR(60), ENDEREÇO  
VARCHAR(120));
```

Terá que inserir uma linha de registro assim:

```
INSERT INTO cadastro VALUES (NULL, 'FULANO DE TAL',  
'RUA SOBE E DESCE, SEM NÚMERO');
```

Ou preencher no bloco de notas com o termo **NULL** no campo referente a CODIGO.

 26



INFORMA, FORMA, TRANSFORMA.


2. Forma indireta – LOAD DATA (Cont.)



Exercício:

Insira agora na tabela *Cliente*, 5 registros diferentes pela forma indireta, sendo que:

- 2 pessoas possuem exatamente o mesmo nome.



27

INFORMA, FORMA, TRANSFORMA.


Consulta de Registros

O comando **SELECT** possui diversos parâmetros de utilização, e é no conhecimento destes parâmetros que está baseada boa parte de nossos estudos em MySQL.

Vamos começar pelo comando **SELECT** mais básico de todos.

Visualize todo o conteúdo da tabela *cliente* com o seguinte comando:

```
SELECT * FROM cliente;
```

28

Consulta de Registros (Cont.)

Para extrair listagens mais específicas de registros armazenados em uma tabela, utilize o comando **SELECT**, anexado a alguns parâmetros:

[tipo] é opcional, podendo ser:

- DISTINCT (registros distintos)
- ALL (todos os registros)

<tabela> é o nome da tabela (ou tabelas) de onde se deseja consultar os registros.

SELECT [tipo] <campos> FROM <tabela> [condição];

<campos> é a lista de campos a serem selecionados. Pode ser utilizado o valor * (asterisco) para todos os campos da tabela

[condição] é opcional e determina a condição de ação de pesquisa, podendo ser:

WHERE		Determina a ação de trabalho de uma condição baseada em uma relação lógica.
GROUP BY	ASC	Indica o agrupamento de informações baseado em valores comuns a partir de uma coluna informada.
	DESC	
ORDER BY	ASC	Define a forma de ordenação dos registros, podendo ser ASC (ascendente) e DESC (descendente)
	DESC	



29

Consulta de Registros (Cont.)

Tente agora estas outras opções de consulta sugeridas abaixo, e **compare os resultados** no intuito de entendimento do respectivo código inserido:

```
SELECT * FROM cliente;
```

```
SELECT NOME, TELEFONE FROM cliente;
```

```
SELECT TELEFONE, NOME FROM cliente;
```

```
SELECT NOME, EMAIL FROM cliente WHERE
MATRICULA = 5;
```

```
SELECT NOME, NASCIMENTO FROM cliente
ORDER BY NOME;
```

```
SELECT NOME, MATRICULA, TELEFONE FROM cliente
ORDER BY NOME DESC;
```



30

Consulta de Registros (Cont.)

```
SELECT NOME, NASCIMENTO FROM cliente ORDER BY
NASCIMENTO, NOME DESC;
```

```
SELECT NOME FROM cliente WHERE NASCIMENTO = 'ddd'
ORDER BY NOME;
```

*Substituir **ddd** pela data repetida que foi inserida.*

```
SELECT NOME, TELEFONE FROM cliente WHERE NOME =
'nnn' ORDER BY MATRICULA DESC;
```

*Substituir **nnn** pelo nome repetido que foi inserido.*

```
SELECT DISTINCT NOME FROM cliente;
```

```
SELECT DISTINCT NOME, TELEFONE FROM cliente;
```

```
SELECT MATRICULA, NOME FROM CLIENTE WHERE TELEFONE
= 'ttt' ORDER BY NASCIMENTO;
```

*Substituir **ttt** pelo telefone repetido que foi inserido.*



31

Sistema
FIRJAN



INFORMA, FORMA, TRANSFORMA.

Consulta de Registros (Cont.)

A título de verificação de funcionalidade de campos com domínio **date**, será apresentada a relação de todos os clientes nascidos no mês de dezembro de qualquer ano, se for utilizada a seguinte sintaxe:

```
SELECT NOME, NASCIMENTO FROM cliente WHERE
MONTH(NASCIMENTO) = 12;
```

A seguir observe a listagem de todos os clientes nascidos a partir de 1º de janeiro de 2000. Para tanto, use a seguinte sintaxe:

```
SELECT NOME, NASCIMENTO FROM cliente WHERE
NASCIMENTO >= '2000-01-01';
```



32

Sistema FIRJAN | **INFORMA, FORMA, TRANSFORMA.**

Alteração de Registros

O processo de alteração de registros é realizado no MySQL pelo comando abaixo:

```
UPDATE <tabela> SET <campo> = <expressão> [condição];
```

Nome da a tabela que terá registros alterados / atualizados

É a indicação de um campo da tabela

É a indicação do valor do campo a ser atualizado

É opcional. Determina a condição de ação de pesquisa baseada no argumento **WHERE**

 33

Sistema FIRJAN | **INFORMA, FORMA, TRANSFORMA.**

Alteração de Registros (Cont.)

Vamos então aplicar um comando de alteração de registros em nosso BD de estudo.

Primeiro faça uma consulta ao(s) registro(s) que será(ão) alterado(s) posteriormente:


```
SELECT * FROM cliente;
```



Observe bem os clientes que possuem telefone repetido nesta consulta. Agora vamos aplicar a alteração:

```
UPDATE cliente SET TELEFONE = '(21)1111-5555' WHERE MATRICULA = m;
```

Substitua **m** pela matrícula de um dos clientes com telefone repetido

Agora repita o comando **SELECT** acima e observe os telefones.

 34

  INFORMA, FORMA, TRANSFORMA.


Alteração de Registros (Cont.)

Agora troque o nome de uma pessoa que possui nome repetido em sua tabela cliente. Como deve ficar este comando?



Visualização inicial:
`SELECT * FROM cliente;`

Alteração:
`UPDATE cliente SET NOME = 'nnn' WHERE MATRICULA = m;`

Visualização final:
Basta repetir o comando da visualização inicial e observar agora na coluna nome que o antigo nome repetido foi substituído pelo recente nome alterado.



35

  INFORMA, FORMA, TRANSFORMA.


Remoção de Registros

Vamos estudar o comando para remover registros:

`DELETE FROM <tabela> [condição];`

Nome da a tabela que terá registros removidos

É opcional. Determina a ação da pesquisa baseada no argumento **WHERE**



36

INFORMA, FORMA, TRANSFORMA.

Remoção de Registros (Cont.)

CUIDADO !!!

Se você aplicar o comando como o do exemplo abaixo:



```
DELETE FROM cliente;
```

Simplesmente **APAGARÁ TODOS OS REGISTROS** da tabela *cliente*.

É muito recomendável que seja utilizado o parâmetro **WHERE** nas operações de remoção de registros, a menos que se tenha certeza de que é preciso realmente apagar todos os registros.

Não existe um recurso para desfazer ocorrências.

37

INFORMA, FORMA, TRANSFORMA.

Remoção de Registros (Cont.)

Vamos então ao treino (com cautelas, para não perdermos dados por descuido):


Visualize todos os registros da tabela cliente. Creio que não seja mais preciso apresentar o comando para isso, ok?

Agora exclua um conjunto de registros, com este comando:

```
DELETE FROM cliente WHERE NASCIMENTO = 'nnn';
```

Substituir **nnn** pela data de nascimento que está repetida

Visualize novamente todos os registros e observe as alterações.

38

Alteração de Tabelas

É possível **alterar a estrutura de uma tabela**. Para isso, segue o comando do MySQL:

Nome da a tabela que terá a alteração de estrutura.

ALTER TABLE <tabela> <operação>;

- **ADD <campo> <tipo>**
Adiciona um campo (coluna) à tabela.
- **DROP <campo>**
Apaga um campo (coluna) da tabela.
- **MODIFY <campo> <novo tipo>**
Modifica o tipo de uma coluna (Ex: de INT para SMALLINT).
- **RENAME TO <novo nome da tabela>**
Renomeia a tabela.
- **CHANGE <nome do campo> <novo nome do campo> <novo tipo>**
Renomeia o campo (coluna), sendo necessário incluir o tipo deste campo que está sendo renomeado, que pode ser o mesmo tipo anterior, ou pode ser alterado.



39

Sistema
FIRJAN



INFORMA, FORMA, TRANSFORMA.

Alteração de Tabelas (Cont.)

Vamos treinar, inserindo um campo em nossa tabela de estudos:

ALTER TABLE cliente ADD CPF CHAR(14);

Agora visualize todo o conteúdo da tabela **cliente**.

Observe que a coluna **CPF** é indicada com valores **"NULL"**, o que indica que está sem registros.


Agora insira os dados na nova coluna CPF. Veja um exemplo para sua tarefa:

UPDATE cliente SET CPF = '031.285.342-05' WHERE MATRICULA = 6;

Faça isso para todos os clientes cadastrados na tabela, colocando um CPF diferente para cada um.






40



Exercício

Inclua os registros de mais 5 clientes na tabela *cliente*, preenchendo todos os campos de registro, sem repetição de dados entre estes novos registros. Escolha o método de inserção de registros que mais lhe agrada.






41




Dúvidas?




42




Bibliografia



MySQL – Guia do Programador (1ª Edição)
André Milani
Ed. Novatec



MySQL – Guia de Bolso (2ª Edição)
George Reese
Ed. Alta Books



MySQL 5.1 Interativo (3ª Edição)
José Augusto N.G. Manzano
Ed. Érica

43