

Programação e Desenvolvimento de Banco de Dados

**Manipulação de dados e
estruturas**

Prof. Dr. Gilberto Fernandes Jr.

- Unidade de Ensino: 2
- Competência da Unidade: Conhecer e compreender a criação e manipulação de tabelas.
- Resumo: Saber elaborar script SQL inserção e manutenção de dados, alteração e exclusão de tabelas e constraints.
- Palavras-chave: SQL, manipulação de dados, *constraints*
- Título da Teleaula: Manipulação de dados e estruturas
- Teleaula nº: 2

Conteúdo

- Comandos utilizados na manipulação de bancos de dados
- Alteração de tabelas e *constraints*
- Exclusão de tabelas em banco de dados

**Comandos para
manipulação de
bancos de dados:
SELECT e INSERT**

Introdução

- Linguagem de manipulação de dados (DML)
 - inserir, atualizar, excluir ou modificar dados
 - Principais instruções
 - Cláusulas condicionais

A cláusula SELECT

- Consulta SQL e o resultado é uma tabela
- Cláusula **WHERE**
 - AND, OR, NOT, <, <=, >, >=, = e <>
- Cláusula **FROM**
 - Especifica uma ou mais tabelas

```
SELECT * FROM convidado;
```

```
SELECT * FROM convidado WHERE nome LIKE 'A%';
```

Instrução de Inserção (INSERT)

- Permite adicionar novas linhas ou registro numa tabela existente.
- Sintaxe:

```
INSERT  
  [INTO] nome_tabela  
  [(nome_coluna [, nome_coluna] ...)]  
  {VALUES | VALUE} (lista_valores) [, (lista_valores)] ...
```

Instrução de Inserção (INSERT)

- Uma expressão pode se referir a qualquer coluna que tenha sido definida anteriormente em uma lista de valores

```
INSERT INTO nome_tabela (col1, col2) VALUES(15, col1*2);
```

- Inserir múltiplas linhas

```
INSERT INTO nome_tabela (a,b,c)  
VALUES(1,2,3),(4,5,6),(7,8,9);
```




Instrução de Inserção (INSERT)

- Exemplo: considere a tabela a seguir

```
CREATE TABLE IF NOT EXISTS convidado (  
  id INT(11) NOT NULL AUTO_INCREMENT PRIMARY KEY,  
  nome VARCHAR(50) NOT NULL DEFAULT '',  
  nascimento DATE,  
  estudante ENUM('Não', 'Sim') NOT NULL DEFAULT 'Não'  
);
```

```
INSERT INTO CONVIDADO (nome, nascimento, estudante)
VALUES ('Dani Moura', '1979-03-28', 'Sim');
```



- Resultado:

Result Grid   Filter Rows: <input type="text"/>				
	id	nome	nascimento	estudante
	1	Dani Moura	1979-03-28	Sim
	NULL	NULL	NULL	NULL

Fonte: livro texto

```
INSERT INTO CONVIDADO (nome, nascimento, estudante)
VALUES ('Rui Albuquerque', null , 'Sim');
```

- Resultado:

Result Grid   Filter Rows: <input type="text"/>				
	id	nome	nascimento	estudante
	1	Dani Moura	1979-03-28	Sim
	2	Rui Albuquerque	NULL	Sim
	NULL	NULL	NULL	NULL

Fonte: livro texto

**Comandos para
manipulação de
bancos de dados:
UPDATE e DELETE**

Instrução de Atualização (UPDATE)

- É uma instrução DML que altera ou atualiza linhas em uma tabela.

```
UPDATE tabela_referência
    SET lista_atribuição [WHERE condição]
    [ORDER BY ...] [LIMIT quantidade_linhas]
value:
    {expr | DEFAULT}
assignment:
    nome_coluna = valor
lista_atribuições:
    atribuição [, atribuição] ...
```

Instrução de Atualização (UPDATE)

- UPDATE: atualiza colunas de linhas existentes na tabela nomeada com novos valores.
- SET: indica quais colunas modificar e os valores que devem ser fornecidos.

```
UPDATE convidado  
  SET estudante = 'Sim'  
  WHERE nome = 'Lebrencio Grulher'  
    AND nascimento = '08-Jul-1990';
```

Instrução de Atualização (UPDATE)

- WHERE: condições que identificam quais linhas devem ser atualizadas

```
UPDATE convidado SET estudante = 'Não';
```

- ORDER BY e LIMIT:

```
UPDATE convidado  
  SET estudante = 'Sim'  
  WHERE nascimento < '08-Jul-1990'  
  LIMIT 10  
  ORDER BY nome;
```

Instrução de Atualização (UPDATE)

- UPDATE pode ser usado para várias tabelas ao mesmo tempo
 - Não pode utilizar ORDER BY e LIMIT, neste caso

```
UPDATE lista, produto SET lista.preco = produto.preco  
WHERE lista.id = produto.id;
```

Instrução de Exclusão (DELETE)

- Instrução DML que exclui linhas de uma tabela.
- Uso semelhante à UPDATE.
- Sua sintaxe geral é dada por:

```
DELETE FROM nome_tabela  
    [WHERE condição]  
    [ORDER BY ...]  
    [LIMIT quantidade_linhas]
```


Instrução de Exclusão (DELETE)

- Exemplos:

```
DELETE FROM convidados  
WHERE estudante = 'Sim'  
ORDER BY nome  
LIMIT 10;
```

```
DELETE FROM log_usuario  
WHERE usuario = 'rm'  
ORDER BY datahora_acao LIMIT 1;
```

**Testando a
estrutura do banco
de dados Guia
Turístico**

Descrição da SP

- Trabalhando em uma multinacional, no desenvolvimento de um Guia Turístico, você está na etapa de manipulação de banco de dados.
- **Sua tarefa é testar a estrutura do banco criado pela equipe anterior**

- O que você deve inserir:
 - **Países** (com respectivos continentes): Brasil, Índia, China e Japão;
 - **Estados** (com respectivas siglas): Maranhão, São Paulo, Santa Catarina, Rio de Janeiro;
 - **Cidades** (com respectivas populações aproximadas): Sorocaba, Déli, Xangaim Tóquio;
 - **Pontos turísticos** (com sua especificação: Quinzinho de Barros (Instituição), Parque Estadual do Jalapão (Atrativo), Torre Eiffel (Atrativo), Fogo de Chão (Restaurante).

- O que você deve alterar:
 - Alterar para “Atrativo” o primeiro ponto turístico
 - Alterar o segundo país (Índia) para ter o código “IND”
- Deletar a primeira cidade
- **Em seguida, avalie a estrutura do banco, verificando se tudo ocorreu corretamente!**
- **Agora, utilizando o MySQL, vamos resolver a SP!**

Alteração de tabelas

Alteração de tabelas (ALTER TABLE)

- Comando que altera a estrutura de uma tabela
 - Adicionar ou excluir colunas
 - Criar ou destruir índices
 - Renomear colunas, ou a própria tabela
 - Alterar o mecanismo de armazenamento

```
ALTER TABLE nome_tabela  
    [especifi cação_alteração  
    [,especifi cação_alteração] ...]
```

Alteração de tabelas (ALTER TABLE)

- A sintaxe para as alterações é semelhante às cláusulas da instrução CREATE TABLE.
- Alguns comandos de especificação_alteração:

```
ADD [COLUMN] (nome_coluna column_definition,...)
CHANGE [COLUMN] old_nome_coluna new_nome_coluna
column_definition
DROP [COLUMN] nome_coluna
RENAME COLUMN old_nome_coluna TO new_nome_coluna
```


Alteração de tabelas (ALTER TABLE)

- Múltiplas cláusulas ADD, ALTER, DROP e CHANGE são permitidas em uma única instrução ALTER TABLE, separadas por vírgulas

```
ALTER TABLE cliente DROP COLUMN parentesco, DROP COLUMN telefones;
```

- Outros exemplos:

```
ALTER TABLE cliente AUTO_INCREMENT = 13;
```

```
ALTER TABLE pessoas CHARACTER SET = latin1;
```

Adicionando e excluindo colunas (ADD e DROP)

- ADD: adicionar novas colunas a uma tabela
- DROP: remover colunas existentes
- FIRST ou AFTER nome_coluna: adicionar uma coluna a uma posição específica dentro de uma linha da tabela

Adicionando e excluindo colunas (ADD e DROP)

- Cuidados ao excluir colunas:
 - Tabelas com uma coluna
 - Índices dos quais colunas fazem parte.
 - Remover todas as colunas de um índice
 - Usar CHANGE ou MODIFY para encurtar uma coluna

Renomeando, redefinindo e reordenando colunas

- As cláusulas CHANGE, MODIFY, RENAME COLUMN e ALTER permitem que os nomes e definições de colunas existentes sejam alterados. Exemplos:

```
ALTER TABLE pessoas CHANGE antigo novo BIGINT NOT NULL;
```

```
ALTER TABLE pessoas MODIFY novo INT NOT NULL;
```

```
ALTER TABLE pessoas RENAME COLUMN novo TO antigo;
```

Como adicionar um campo 'nome' na tabela "pessoas"? E como excluir um campo?

Resolução

- Adicionar:

```
ALTER TABLE pessoas ADD nome VARCHAR(50);
```

- Remover:

```
ALTER TABLE pessoas DROP COLUMN sobrenome;
```

O uso de
constraints

Usando restrições (*constraints*)

```
CREATE TABLE pessoa (  
    id int NOT NULL PRIMARY KEY,  
    nome varchar(255) NOT NULL,  
    sobrenome varchar(255),  
    idade int );
```

```
ALTER TABLE pessoa  
DROP PRIMARY KEY;
```


```
ALTER TABLE pessoa  
ADD CONSTRAINT PK_pessoa PRIMARY KEY (id, sobrenome);
```



Usando restrições (*constraints*)

- Sintaxe para definição de restrição de chave estrangeira com CREATE TABLE ou ALTER TABLE

```
ALTER TABLE tbl_name  
  ADD [CONSTRAINT [símbolo]] FOREIGN KEY  
    [index_nome] (index_col_nome, ...)  
  REFERENCES tbl_nome (index_col_nome,...)  
  [ON DELETE referências]  
  [ON UPDATE referências]
```



referências:

RESTRICT | CASCADE | SET NULL | NO ACTION

Usando restrições (*constraints*)

- Regras para nomear um índice de chave estrangeira:
 - Se definido, o valor do símbolo CONSTRAINT é usado. Caso contrário, o valor do index_nome FOREIGN KEY é usado.
 - Se nenhum símbolo CONSTRAINT ou FOREIGN KEY index_nome estiverem definidos, o nome do índice de chave estrangeira será gerado usando o nome da coluna de chave estrangeira de referência.

Usando restrições (*constraints*)

- Integridade referencial:

```
CREATE TABLE pai (  
    id INT NOT NULL,  
    nome VARCHAR(50),  
    PRIMARY KEY (id)  
);
```

```
CREATE TABLE filha (  
    id INT PRIMARY KEY,  
    parente_id INT,  
    nome VARCHAR(50)  
);
```



```
ALTER TABLE filha  
    ADD CONSTRAINT FK_parente  
    FOREIGN KEY (parente_id) REFERENCES pai(id);
```

Qualificadores

- **CASCADE**: qualificador que exclui ou atualiza a linha da tabela pai e exclui ou atualiza automaticamente as linhas correspondentes na tabela filha.
- **SET NULL**: exclui ou atualiza a linha da tabela pai e define como NULL a coluna ou colunas de chave estrangeira na tabela filha.

Qualificadores

- **SET NULL e ON UPDATE SET NULL** são suportados
- **RESTRICT**: rejeita a operação de exclusão ou atualização da tabela pai.
- **NO ACTION**: essa é uma palavra-chave do SQL padrão. No MySQL, a equivalente é a RESTRICT

```
ALTER TABLE filha DROP FOREIGN KEY FK_parente;
```

Alterando o banco de dados Guia Turístico

Descrição da SP

- Trabalhando em seu projeto de construção do Guia Turístico, neste momento, seu repositório de dados está todo testado e temos vários dados e relacionamentos criados.
- Questionamentos:
 - Língua(s) nativas dos países?
 - Há alterações que devem ser feitas na estrutura do baco de dados, nas tabelas e no DER?

Descrição da SP

- **Alteração do banco em 3 níveis:**
 - Mudança na definição do banco (DDL)
 - Identificar quais campos estão vazios (DQL)
 - Identificar quais relacionamentos estão quebrados (DML).
- **Corrigir estas situações, garantindo que não haja campos vazios ou relacionamentos quebrados.**

Resolução da SP

- Revisar o DER e readequá-lo.
- Para a criação de relacionamentos entre tabelas, será necessária uma tabela pai e uma filha.
 - Utilizar integridade referencial!



Exclusão de tabelas em banco de dados

Introdução


- Manutenção de dados (incluir, alterar e excluir)
- Respeitar estrutura (restrições)
- A restrição FOREIGN KEY é usada para impedir ações que destruam links entre tabelas

```
CREATE TABLE aluno (  
    id INT NOT NULL AUTO_INCREMENT PRIMARY KEY,  
    nome CHAR(50) NOT NULL  
);  
  
CREATE TABLE curso (  
    id INT NOT NULL AUTO_INCREMENT PRIMARY KEY,  
    nome CHAR(50) NOT NULL  
);  
  
CREATE TABLE nota (  
    aluno_id INT NOT NULL,  
    curso_id INT NOT NULL,  
    dataavaliacao DATE NOT NULL,  
    nota DOUBLE NOT NULL,  
    PRIMARY KEY(aluno_id, curso_id, dataavaliacao),  
    INDEX i2 (curso_id),  
    FOREIGN KEY (aluno_id) REFERENCES aluno(id) ON DELETE  
        CASCADE,  
    FOREIGN KEY (curso_id) REFERENCES curso(id) ON DELETE  
        RESTRICT  
);
```

Fonte: livro texto

Result Grid	Filter Rows: <input type="text"/>	Export: 	Wrap Cell Content: 
Table	Create Table		
nota	CREATE TABLE `nota` (`aluno_id` int(11) NOT NULL, `curso_id` int(11) NOT NULL, `dataavaliac		

Fonte: livro texto



```
CREATE TABLE `nota` (
    `aluno_id` int(11) NOT NULL,
    `curso_id` int(11) NOT NULL,
    `dataavaliacao` date NOT NULL,
    `nota` double NOT NULL,
    PRIMARY KEY (`aluno_id`,`curso_id`,`dataavaliacao`),
    KEY `i2` (`curso_id`),
    CONSTRAINT `nota_ibfk_1` FOREIGN KEY (`aluno_id`)
REFERENCES
    `aluno` (`id`) ON DELETE CASCADE,
    CONSTRAINT `nota_ibfk_2` FOREIGN KEY (`curso_id`)
REFERENCES
    `curso` (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1
```

Fonte: livro texto

Outras operações

- DROP TABLE: remover uma ou mais tabelas

```
DROP TABLE [IF EXISTS] nome_tabela [, nome_tabela] ...
```

- Exclusão com restrição de chave estrangeira:

```
DROP TABLE IF EXISTS aluno;
```

```
ALTER TABLE nota DROP FOREIGN KEY nota_ibfk_1;
```

Outras operações

- TRUNCATE TABLE: instrução para esvaziar uma tabela completamente

```
TRUNCATE [TABLE] nome_tabela;
```

- Instruir o MySQL a ignorar restrições


```
SET FOREIGN_KEY_CHECKS = 1;
```

**Incluindo um
sistema de
coordenadas no
banco Guia Turístico**

Descrição da SP

- Você está trabalhando no desenvolvimento de um **guia turístico** em sua empresa.
- Você já tem um banco de dados com uma estrutura pronta, relacionamentos criados, chaves e restrições definidas
- Após o início de um projeto, pode haver a necessidade de revisão
 - **Incluir/excluir estruturas!**

Descrição da SP

- Você deve incluir um sistema de coordenadas! 
- **Alterar tabela de Elementos Turísticos**, adicionando campos latitude e longitude.
- **Alterar a tabela “Países”**, adicionando uma nota de 0 a 10 com o nível de interesse para o turista
- **Alterar tabela “Cidades”**, incluindo uma lista com os três melhores restaurantes.
- Após isso, não teremos a necessidade de manter a tabela coordenadas e você poderá excluí-la.

Dúvidas?

Recapitulando

Recapitulando...

- Comandos utilizados na manipulação de bancos de dados
 - SELECT, INSERT, UPDATE, DELETE
- Alteração de tabelas e constraints
 - ALTER TABLE, ADD, DROP, CHANGE, MODIFY
- Exclusão de tabelas em banco de dados
 - DROP TABLE, TRUNCATE TABLE

