Comandos DDL (Data Definition Language)

Banco de Dados II

Banco de Dados I

Comandos DDL (Data Definition Language)

A maioria dos SGBDs (inclusive o SQL Server e o MySQL) disponibiliza de ferramentas gráficas (conforme será verificado em sala de aula) que permitem a criação de Bancos de Dados, mas é possível criar o próprio Banco de Dados a partir de um comando SQL. Neste material, iremos apenas estudar a implementação e manipulação de banco de dados a partir de comandos SQL. Durante as aulas e atividades em sala de aula, usaremos as ferramentas visuais.

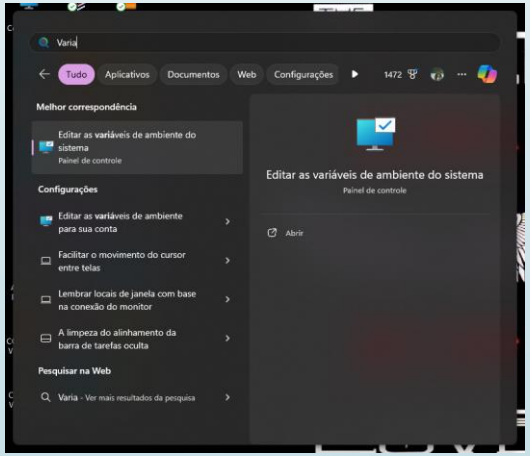
Banco de Dados I

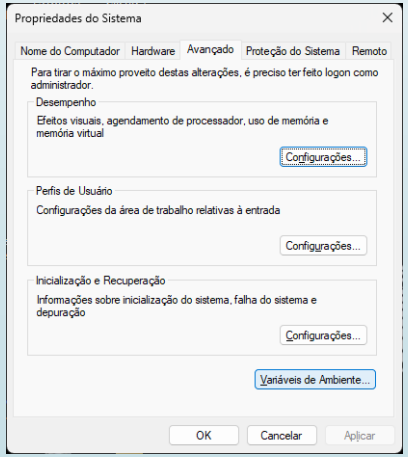
Comandos DDL (Data Definition Language)

Conectando ao Banco de dados via terminal

Se você utiliza Windows e desejar se conectar via terminal, antes será necessário configurar a Variável de Ambiente com seguindo os passos a seguir:

1. No menu Iniciar, digite Variaveis de Ambiente;
2. Clique no botão Variáveis de ambiente;



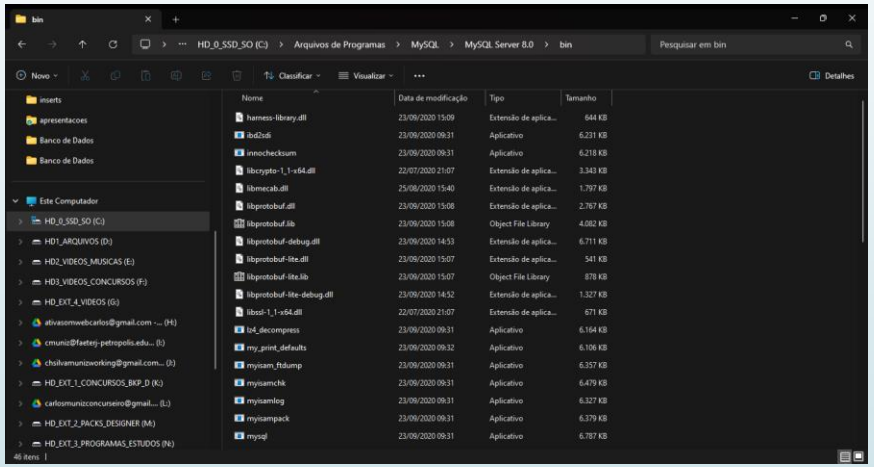


Banco de Dados I

Comandos DDL (Data Definition Language)

Conectando ao Banco de dados via terminal

3. Abra o explorador de Arquivos e acesse o caminho **C:\Program Files\MySQL\MySQL Server 8.0\bin**;



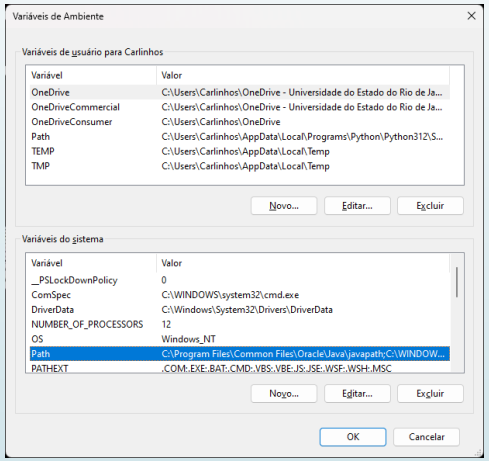
4. Na barra de endereços do explorador de arquivos **com o botão direito do mouse**, e clique em copiar endereço;

Banco de Dados I

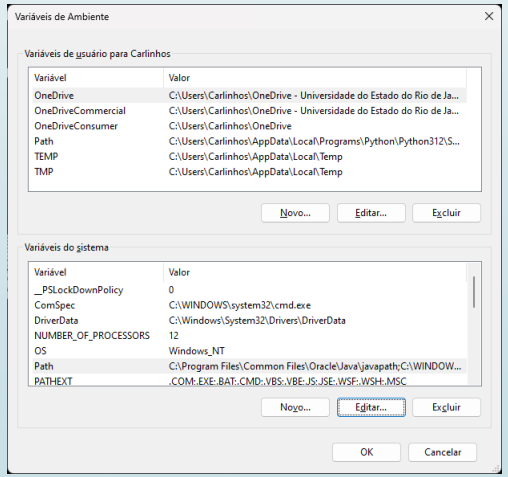
Comandos DDL (Data Definition Language)

Conectando ao Banco de dados via terminal

5. Retorne à janela de **Variáveis de ambiente** e, na parte de baixo em Variáveis do sistema, selecione a opção **Path** e, em seguida, clique no botão **Editar**;



6. Na caixa **Editar a variável de ambiente**, clique no botão **Novo**;

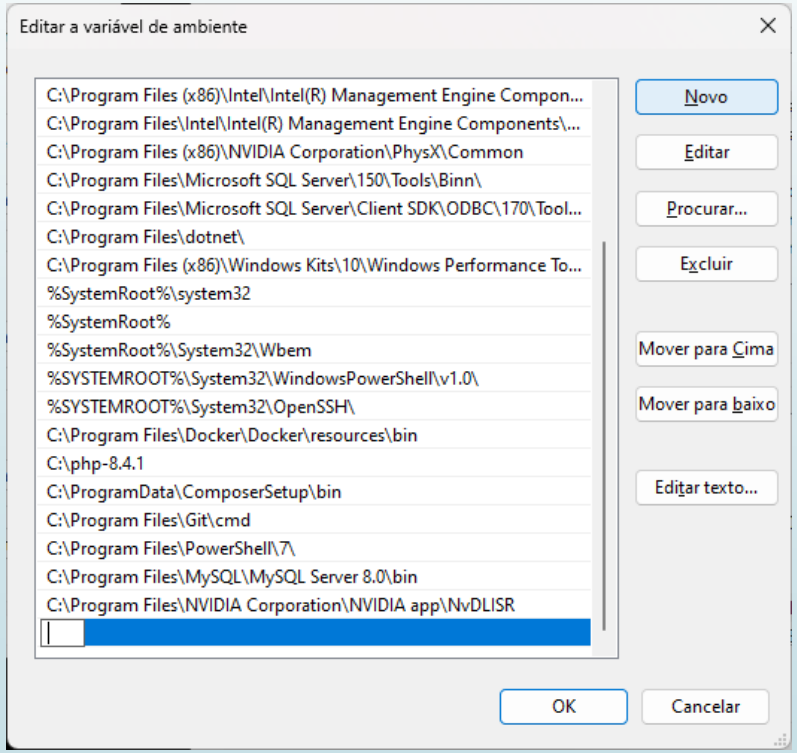


Banco de Dados I

Comandos DDL (Data Definition Language)

Conectando ao Banco de dados via terminal

7. Cole o endereço copiado na nova linha disponível e, em seguida, clique em Ok.

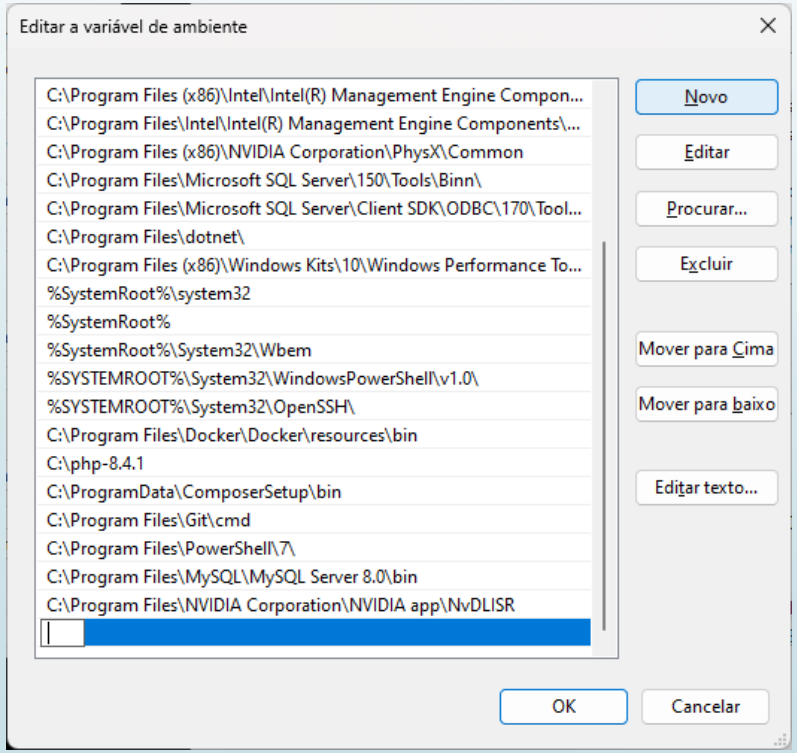


Banco de Dados I

Comandos DDL (Data Definition Language)

Conectando ao Banco de dados via terminal

7. Cole o endereço copiado na nova linha disponível e, em seguida, clique em Ok.



Banco de Dados I

Comandos DDL (Data Definition Language)

Conectando ao Banco de dados via terminal

```
mysql -u root -p
```

Parâmetro:

- **-u** - Indica o usuário ao qual desejamos utilizar para conectar ao banco.

- **root** - Nome do usuário que estamos utilizando para realizar a conexão com o banco de dados. Onde pode ser qualquer usuário cadastrado no MySQL.

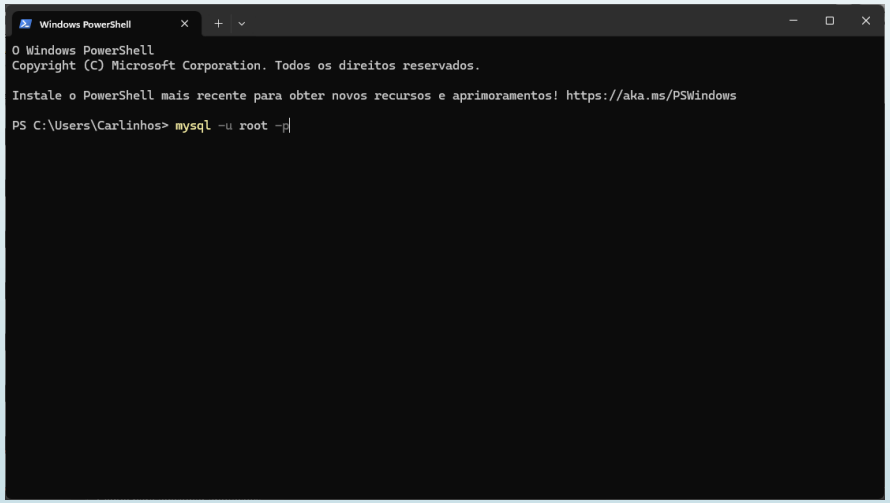
- **-p** - Indica a senha do usuário para conexão com o banco, este parâmetro é opcional, pois caso o usuário desejado não utilize senha para se conectar basta omitir este parâmetro, porem não é nada recomendado utilizar usuários sem senha.

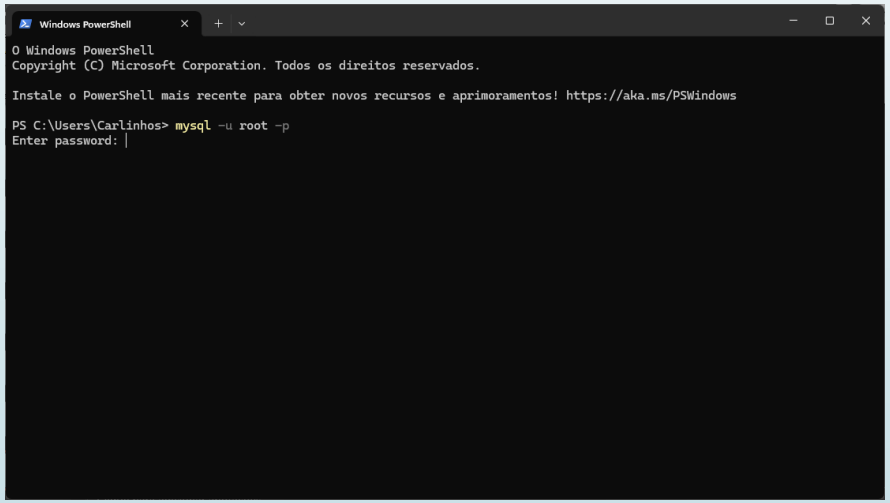
Banco de Dados I

Comandos DDL (Data Definition Language)

Conectando ao Banco de dados via terminal

```
mysql -u root -p
```



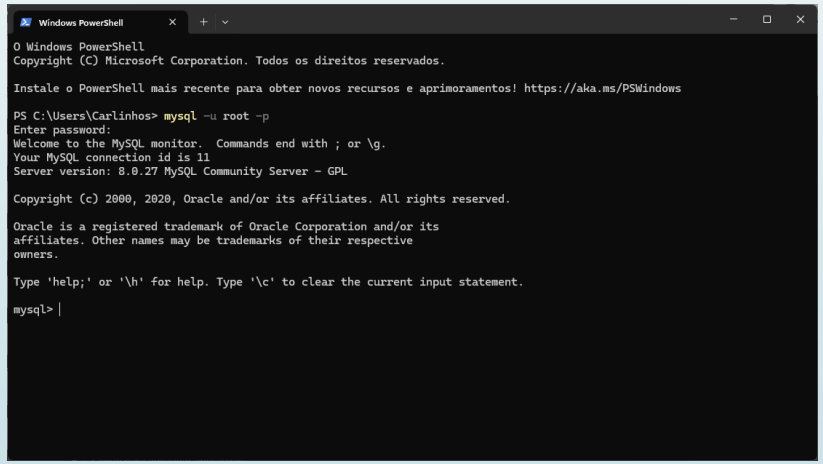


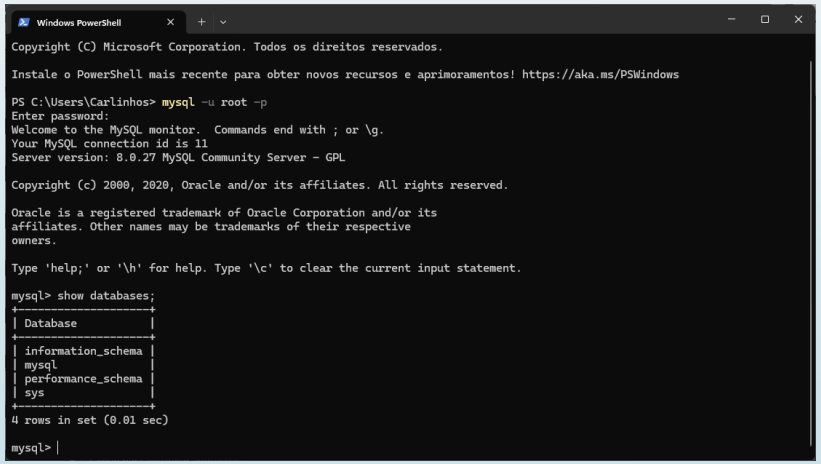
Banco de Dados I

Comandos DDL (Data Definition Language)

Conectando ao Banco de dados via terminal

```
mysql -u root -p
```





Banco de Dados I
Comandos DDL (Data Definition Language)

- **Abrindo (conectando) a um banco de dados**

```
USE [nome do banco de dados];
```

Exemplo:

```
USE sistema;
```

Mensagem do MySQL:

```
database changed
```

Banco de Dados I
Comandos DDL (Data Definition Language)

- **Criando um banco de dados**

Cria-se uma vez e seleciona-se sempre que precisar trabalhar com o mesmo.

```
CREATE DATABASE [Nome do Banco de Dados];
```

Exemplo:

```
CREATE DATABASE sistema;
```

Mensagem do MySQL:

```
Query OK, 1 row affected (0.00 sec)
```

Banco de Dados I
Comandos DDL (Data Definition Language)

- **Exibindo os bancos de dados existentes**

```
SHOW DATABASES;
+--------------------+
| Database |
+--------------------+
| information_schema |
| mysql |
| sistema |
| test |
+--------------------+
4 rows in set (0.08 sec
```

Algumas tabelas existentes

O Mysql, por conta de sua instalação, já vem com algumas tabelas criadas:-

Mysql: Privilégio de acesso de usuários. Não elimine esta tabela.
Test: Criada como espaço de testes

Banco de Dados I
Comandos DDL (Data Definition Language)

- **Abrindo (conectando) a um banco de dados**

```
USE [nome do banco de dados];
```

Exemplo:-

```
USE sistema;
```

Mensagem do MySQL:-

```
database changed
```

Banco de Dados I
Comandos DDL (Data Definition Language)

- **Criando um banco de dados**

  Cria-se uma vez e seleciona-se sempre que precisar trabalhar com o mesmo.

```
CREATE DATABASE [Nome do Banco de Dados];
```

  Exemplo:-

```
CREATE DATABASE sistema;
```

  Mensagem do MySQL:-

```
Query OK, 1 row affected (0.00 sec)
```

Banco de Dados I
Comandos DDL (Data Definition Language)

- **Eliminando Banco de Dados**

Para se eliminar um banco de dados, deve-se utilizar o seguinte comando:

```
DROP [Nome do Banco de Dados]
```

Exemplo:-

```
DROP sistema;
```

Mensagem do MySQL:-

```
Query OK, 0 rows affected (0.00 sec)
```

Banco de Dados I
Comandos DDL (Data Definition Language)

- **Criando Tabelas**

  Uma tabela é definida usando o comando CREATE TABLE.

```
create table T (A1 D1, A2 D2,...)
```

  onde T é o nome da tabela, Ai é o nome do campo da tabela T e Di é o de dados.

Banco de Dados I
Comandos DDL (Data Definition Language)

- **Criando Tabelas**

Principais Tipos de Dados.	
CHAR(n)	Armazena caracteres alfanuméricos de tamanho fixo n.
VARCHAR(n)	Cadeia de caracteres de comprimento variável e tamanho máximo de n caracteres.
INTEGER	Dado numérico inteiro de tamanho fixo.
DECIMAL(n, m) NUMERIC(n,m)	Dado numérico de tamanho variável, sendo n o número total de dígitos e m o número de casas decimais.
BIT(n)	Seqüência
TIME	Hora de tamanho fixo.
DATE	Data de tamanho fixo.

Banco de Dados I

Comandos DDL (Data Definition Language)

- **Criando Tabelas**

```
create table funcionario
(matricula  decimal(5),
nome    char(30),
rg    decimal(9),
sexo    char(1),
depto   decimal(5),
endereco varchar(40),
cidade   varchar(20),
salário   decimal(10,2) );
```

Banco de Dados I

Comandos DDL (Data Definition Language)

- **Criando Tabelas**
- **Restrições de Integridade**

As restrições de integridade servem para garantir as regras inerentes ao sistema que está sendo implementado, prevenindo a entrada de informações inválidas pelos usuários desse sistema. Para isso, o Sistema de Banco de Dados deve possibilitar a definição de regras de integridade a fim de evitar a inconsistência dos dados que nele serão armazenados.

Banco de Dados I

Comandos DDL (Data Definition Language)

- **Criando Tabelas**
- **Restrições de Integridade**

  - **Chave Primária** A função da chave primária é identificar univocamente cada registro da tabela. Toda tabela deve possuir uma chave primária, que deve ser composta por um ou mais campos.

```
create table departamento
(Codigo  decimal(5) NOT NULL
PRIMARY KEY,  Nome   char(20);
```

Banco de Dados I

Comandos DDL (Data Definition Language)

- Criando Tabelas
- Restrições de Integridade
  - Chave Primária

```
create table funcionario
(matricula decimal(5) NOT NULL PRIMARY
KEY, nome   char(30),
rg   decimal(9),
sexo   char(1),
depto  decimal(5),
endereço varchar(40),
cidade  varchar(20),
salário  decimal(10,2) )
```

Banco de Dados I

Comandos DDL (Data Definition Language)

- **Criando Tabelas**
- **Restrições de Integridade**
  - **Chave Primária**

    Opcionalmente pode-se definir a chave primária após a especificação de todos os atributos da tabela.

```
create table funcionario (matricula decimal(5) NOT
NULL,  nome   char(30),
rg   decimal(9),
sexo   char(1),
depto  decimal(5),
endereço varchar(40),
cidade  varchar(20),
salario  decimal(10,2),
PRIMARY KEY (matricula) )
```

Quando uma tabela possui uma chave primaria composta por mais de um campo esta forma é obrigatória.

Banco de Dados I

Comandos DDL (Data Definition Language)

- **Criando Tabelas**
- **Restrições de Integridade**

  - **Evitando valores nulos** É muito comum definirmos campos que não podem conter valores nulos. Isto é, o seu preenchimento do campo é obrigatório para que se mantenha a integridade dos dados no sistema. Para evitar que em algum momento um campo de uma tabela possa conter valor nulo (null) deve-se utilizar a cláusula NOT NULL após a definição do campo.

Banco de Dados I

Comandos DDL (Data Definition Language)

- Criando Tabelas
- Restrições de Integridade
  - Evitando valores nulos

```
create table funcionario
(matricula decimal(5) NOT NULL
PRIMARY KEY,
nome    char(30) NOT NULL,
rg    decimal(9),
sexo    char(1),
depto   decimal(5),
endereço varchar(40),   cidade
varchar(20),
salario   decimal(10,2) )
```

No exemplo, o preenchimento do campo nome será obrigatório. Caso o usuário se esqueça de preenche-lo, o SGBD apresentará uma mensagem de erro.

Banco de Dados I

Comandos DDL (Data Definition Language)

- **Criando Tabelas**
- **Restrições de Integridade**

  - **Evitando valores duplicados** Podem existir situações onde o valor armazenado em um campo de um registro deve ser único em relação a todos os registros da tabela. Isto é, não pode haver dois registros com o mesmo valor para um determinado campo. Para implementar esta restrição de integridade deve-se utilizar a cláusula UNIQUE após a especificação de uma coluna.

Banco de Dados I

Comandos DDL (Data Definition Language)

- Criando Tabelas
- Restrições de Integridade
  - Evitando valores nulos

```
create table funcionario
(matricula decimal(5) NOT NULL PRIMARY
KEY,
nome    char(30) NOT NULL,
rg    decimal(9) NOT NULL UNIQUE,
sexo    char(1),
depto   decimal(5),
endereço varchar(40),
cidade   varchar(20),
salario   decimal(10,2) )
```

No exemplo, caso o usuário atribua ao campo RG um valor já existente em outro registro desta mesma tabela, o SGBD apresentará uma mensagem de erro.

Observação: No Interbase é obrigatória a utilização da cláusula NOT NULL juntamente com a cláusula UNIQUE.

Banco de Dados I

Comandos DDL (Data Definition Language)

- **Criando Tabelas**
- **Restrições de Integridade**

  - **Definindo valores default** Pode-se definir um valor padrão para um campo acrescentando à sua definição a cláusula DEFAULT. Esta cláusula permite substituir automaticamente os valores nulos por um valor inicial desejado.

Banco de Dados I

Comandos DDL (Data Definition Language)

- Criando Tabelas
- Restrições de Integridade
  - Definindo valores default

```
CREATE TABLE FUNCIONARIO (MATRICULA
DECIMAL(5) NOT NULL PRIMARY KEY,
NOME    CHAR(30) NOT NULL,
RG    DECIMAL(9) NOT NULL UNIQUE,
SEXO    CHAR(1),  DEPTO  DECIMAL(5),
ENDEREÇO VARCHAR(40),
CIDADE  VARCHAR(20) DEFAULT 'Rio De Janeiro',
SALARIO  DECIMAL(10,2) )
```

Banco de Dados I

Comandos DDL (Data Definition Language)

- **Criando Tabelas**
- **Restrições de Integridade**

  - **Evitando valores inválidos** Existem situações onde um campo pode receber apenas alguns determinados valores. Para que o valor de um campo fique restrito a um determinado conjunto de valores, utiliza-se a cláusula CHECK.

Banco de Dados I

Comandos DDL (Data Definition Language)

- Criando Tabelas
- Restrições de Integridade
  - Evitando valores inválidos

```
CREATE TABLE FUNCIONARIO (MATRICULA DECIMAL(5)
NOT NULL PRIMARY KEY,
NOME    CHAR(30) NOT NULL,
RG    DECIMAL(9) UNIQUE,
SEXO    CHAR(1) CHECK(SEXO IN ('M', 'F')),
DEPTO   DECIMAL(5),
ENDERECO VARCHAR(40),
CIDADE   VARCHAR(20) DEFAULT 'Rio De Janeiro',
SALARIO   DECIMAL(10,2) CHECK(SALARIO>350))
```

Banco de Dados I

Comandos DDL (Data Definition Language)

- **Criando Tabelas**
- **Restrições de Integridade**
  - **Integridade referencial** Freqüentemente desejamos que o valor armazenado em um determinado campo de uma tabela esteja presente na chave primária de outra tabela. Este atributo é chamado chave estrangeira (FOREIGN KEY). Por exemplo, **o campo depto da tabela funcionario deve conter o código de um departamento anteriormente cadastrado na tabela departamento**. Para que essa restrição seja sempre observada, utiliza-se a cláusula REFERENCES na definição do campo depto da tabela funcionario. O campo depto da tabela funcionario é portanto uma chave estrangeira e só será permitido armazenar valores que estejam previamente cadastrado no campo codigo da tabela departamento.

Banco de Dados I

Comandos DDL (Data Definition Language)

- Criando Tabelas
- Restrições de Integridade
  - Integridade referencial

```
CREATE TABLE FUNCIONARIO
(MATRICULA DECIMAL(5) NOT NULL PRIMARY KEY,  NOME
CHAR(30) NOT NULL,
RG DECIMAL(9) NOT NULL UNIQUE,
SEXO CHAR(1), CHECK( SEXO IN ('M', 'F'),
DEPTO DECIMAL(5) REFERENCES DEPARTAMENTO(CODIGO),
ENDERECO VARCHAR(40),
CIDADE VARCHAR(20) DEFAULT 'Rio De Janeiro',
SALARIO  DECIMAL(10,2) CHECK(SALARIO>350) )
```

Banco de Dados I

Comandos DDL (Data Definition Language)

- **Criando Tabelas**
- **Restrições de Integridade**
  - **Integridade referencial**

Assim como na definição da chave primária, pode-se definir a chave estrangeira após a especificação de todos os campos da tabela.

```
CREATE TABLE CLIENTE (MATRICULA DECIMAL(5) NOT NULL,
NOME   CHAR(30) NOT NULL,
RG DECIMAL(9)
SEXO CHAR(1) CHECK(SEXO IN ('M', 'F')),
DEPTO DECIMAL(5),
ENDERECO VARCHAR(40),
CIDADE VARCHAR(20) DEFAULT 'Rio de Janeiro',
PRIMARY KEY (MATRICULA),
FOREIGN KEY (DEPTO) REFERENCES DEPARTAMENTO(CODIGO));
```

Banco de Dados I

Comandos DDL (Data Definition Language)

- **Removendo uma tabela**

  Para remover uma relação de um banco de dados SQL, usase o comando DROP TABLE. O comando DROP TABLE remove todas as informações sobre a relação.

```
DROP TABLE T
```

onde T é o nome de uma tabela do banco de dados

Banco de Dados I

Comandos DDL (Data Definition Language)

- **Alterando uma Tabela**

  O comando ALTER TABLE é usado para adicionar, excluir ou alterar atributos em uma tabela.

- **Incluindo campos**

  Para inserir um novo atributo em uma tabela é usada a cláusula **add**. O novo campo terá valor null para todos os registros da tabela.

```
ALTER TABLE T
ADD A1 D1,
ADD A2 D2,
. . .
```

onde T é o nome de uma tabela e Ai Di é uma lista contendo nome do atributo (Ai) a ser adicionado e o tipo desse atributo (Di).

Banco de Dados I

Comandos DDL (Data Definition Language)

- **Alterando uma Tabela**
  - **Excluir campos**

    Para excluir colunas de uma tabela utiliza-se a cláusula drop.

```
ALTER TABLE T
DROP A1,
DROP A2,
. . .
```

onde T é o nome de uma tabela e Ai é uma lista dos atributos a serem removidos.

Banco de Dados I

Comandos DDL (Data Definition Language)

- **Alterando uma Tabela**

Para alterar o **nome de um atributo** de uma tabela utiliza-se a cláusula **alter...to.**

```
ALTER TABLE T
    ALTER A1 TO NA1,
    ALTER A2 TO NA2,
. . .
```

Exemplo:

```
ALTER TABLE DEPARTAMENTO
    ALTER CODIGO TO DEP_CODIGO,
    ALTER NOME   TO DEP_NOME ;
```

onde T é o nome de uma tabela, A é o nome do atributo a ter o seu nome alterado para nA.

Banco de Dados I

Comandos DDL (Data Definition Language)

- **Alterando uma Tabela**

Para alterar o tipo de um atributo utiliza-se a cláusula **alter...type**.

```
ALTER TABLE T
    ALTER A1 TYPE T1,
    ALTER A2 TYPE T2,
. . .
```

Exemplo:

```
ALTER TABLE DEPARTAMENTO
    ALTER NOME TYPE CHAR(30)
```

onde T é o nome de uma tabela, A é o nome do atributo a ter o seu tipo alterado para D.

Banco de Dados I

Comandos DDL (Data Definition Language)

- **Índices**

Os índices são componentes do banco de dados destinados a agilizar o acesso aos dados. Um índice pode ser associado a uma coluna ou a uma combinação de várias colunas. Uma vez criado um índice, todas as alterações feitas à tabela são automaticamente refletidas no índice. Pode-se criar vários índices para uma tabela. As únicas instruções SQL que tratam de índices são **CREATE INDEX** e **DROP INDEX**.

Banco de Dados I

Comandos DDL (Data Definition Language)

- **Índices**

Criando índices

A instrução CREATE INDEX exige que se defina um nome para o índice a ser criado, seguido do nome da tabela e por fim, uma lista contendo o nome dos atributos que compõem o índice.

```
CREATE INDEX I ON T(A1, A2, ...)
```

Onde i é o nome do índice, T é o nome da tabela que se deseja indexar e Ai são os atributos de indexação.

```
CREATE INDEX RG_FUNCIONÁRIO ON FUNCIONARIO(RG)
```

Banco de Dados I

Comandos DDL (Data Definition Language)

- **Índices**

Existem duas razões principais para se usar índices:

- Evita dados duplicados, aumentando a garantia de integridade no banco de dados com o uso da cláusula UNIQUE;

- Aumenta a rapidez do banco de dados, criando índices que sejam convenientes às consultas mais comuns e freqüentes no banco de dados. A cláusula UNIQUE, quando utilizada, não permite que duas linhas da tabela assumam o mesmo valor no atributo ou no conjunto de atributos indexados.

Banco de Dados I

Comandos DDL (Data Definition Language)

- **Índices**

A cláusula UNIQUE, quando utilizada, não permite que duas linhas da tabela assumam o mesmo valor no atributo ou no conjunto de atributos indexados.

```
CREATE UNIQUE INDEX RG_FUNCIONARIO ON
FUNCIONARIO(RG)
```

O uso de índices pode aumentar a velocidade das consultas, porém deve-se ter cautela na criação desses índices. Não há limites em relação a quantidade de índices criados. No entanto sabe-se que eles ocupam espaço em disco e nem sempre otimizam as consultas, pois não se tem o controle sobre a maneira pela qual os dados serão acessados.

Banco de Dados I

Comandos DDL (Data Definition Language)

- **Índices**

Removendo índices

Para remover índices utiliza-se a instrução DROP INDEX

```
DROP INDEX I
```

onde i é o nome do índice que se deseja excluir

Referências

- **Apostila de Banco de Dados**

  http://www.regilan.com.br

- **Diego Brocanelli**

  www.diegobrocanelli.com.br/mysql/comandos-basicos-mysql-no-terminal/

- **Introdução à linguagem SQL - Prof. Edberto Ferneda**

  sites.ffclrp.usp.br/cid/docentes/edberto/Apostilas/Apostila%20SQL.pdf