Universidade Tecnológica Federal do Paraná

UTFPR - Campus Cornélio Procópio

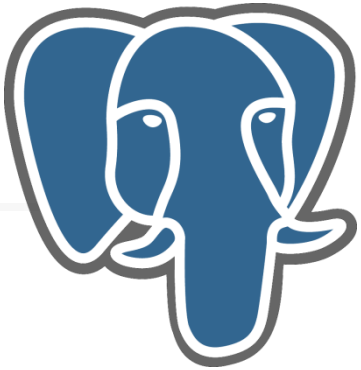
AS33C
Banco de Dados 2

Professor: Eduardo Cotrin Teixeira

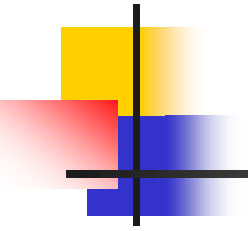
cotrin@utfpr.edu.br

UTFPR-CP / 2022

SGBD PostGreSQL



- SGBD objeto-relacional *open source*.
- Projeto iniciado em 1986 na Universidade da Califórnia, em Berkeley.
- Origem no pacote de ferramentas POSTGRES, depois nome mudou para PostGreSQL.
- Possui recursos avançados.



SGBD PostGreSQL

- **Utilizado**: versão 11 (não é a última, mas já é estável)

- Versões mais antigas (mas nem tanto!) geralmente são mais **estáveis**, com poucos recursos a menos.

SGBD PostGreSQL

- Para acesso aos dados, o PostGreSQL tem 2 ferramentas básicas (ambas acompanham o pacote de instalação):
  - ***pgAdmin*** - ferramenta gráfica
  - ***psql*** - ferramenta de linha de comando
- As 2 ferramentas são executadas como clientes, e têm a mesma finalidade.

SGBD PostGreSQL

- psql
  - É o terminal interativo (interface textual do PostgreSQL).
  - É um *front-end* que permite que consultas sejam processadas interativamente, informando-as diretamente e visualizando os resultados.
  - Alternativamente a entrada pode ser feita via arquivo.

```
psql
Senha para usuário postgres:
psql (11.13)
AVISO: página de código do Console (850) difere da página de código do Windows (1252)
        caracteres de 8 bits podem não funcionar corretamente. Veja página de
        referência do psql "Notes for Windows users" para obter detalhes.
Digite "help" para ajuda.

postgres=# \c aula
Você está conectado agora ao banco de dados "aula" como usuário "postgres".
aula=# \dt
            Lista de relaþ§es
 Esquema |     Nome     |  Tipo  |   Dono
---------+--------------+--------+----------
 public  | departamento | tabela | postgres
 public  | funcionario  | tabela | postgres
(2 registros)


aula=# select * from funcionario;
 nome  |     cpf     | salario | idade
-------+-------------+---------+-------
 Paulo | 11111111111 | 3500.00 |    32
(1 registro)


aula=#
```

SGBD PostGreSQL

- psql
  - Interpreta comandos SQL (todo comando deve terminar com ;)
  - \? - ajuda com comandos do psql
  - \c [nome_BD]- conecta a um BD
  - \l - lista todos os BDs
  - \dt - lista as tabelas do BD atual
  - \d+ [nome_tabela] - lista o esquema da tabela
  - \q - quit
  - \i [nome_arquivo] - carrega arquivo script:

```
postgres=# \i 'C:/Users/UTFPR/teste.sql'
```

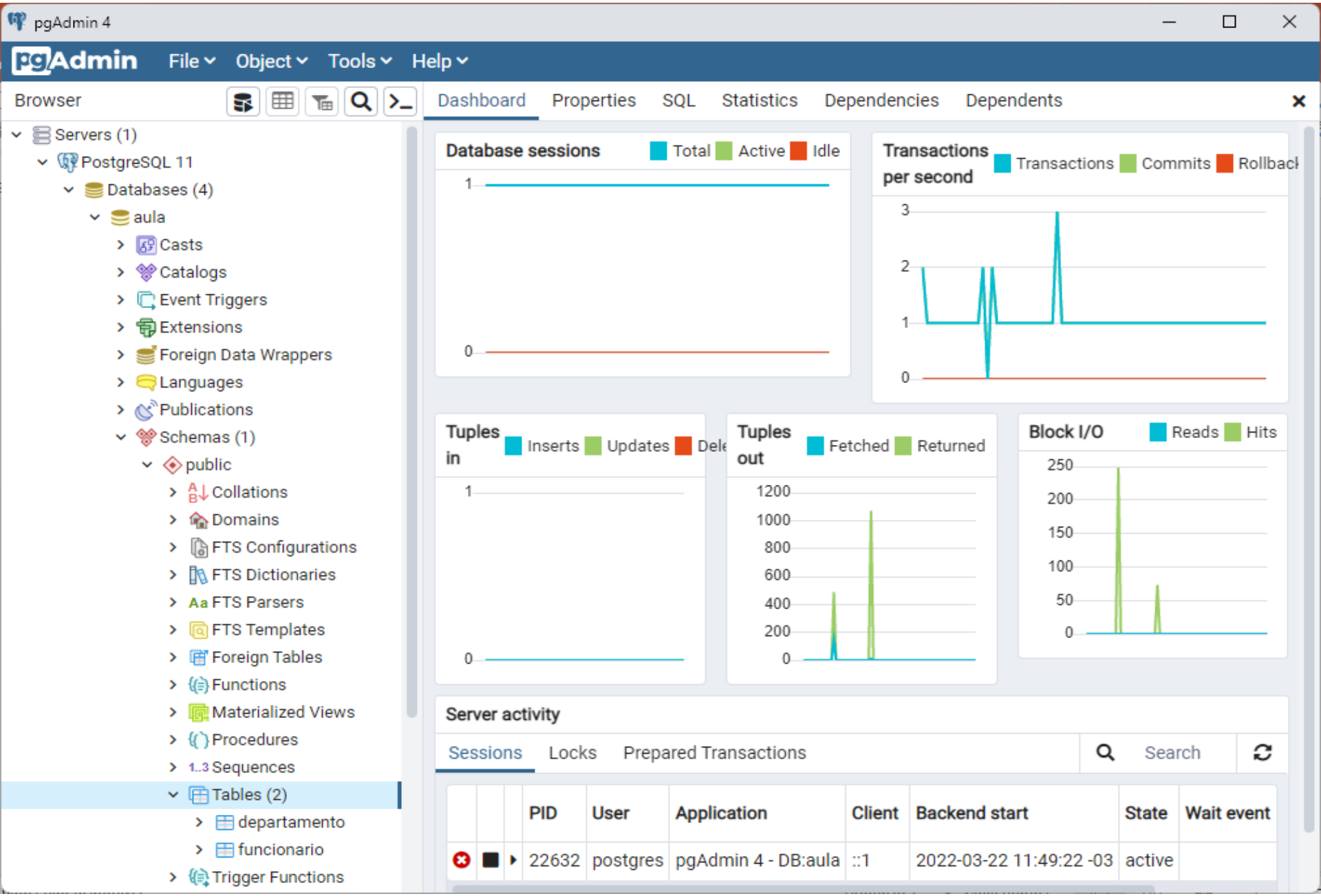
SGBD PostGreSQL

- pgAdmin (atualmente pgAdmin 4)
  - Permite registrar vários servidores.
  - Pronto para conexão com o servidor local.

  * Maioria das funções com o botão direito do mouse. Por exemplo:

  - Clique com o botão direito no nome do servidor para criar um novo BD.

  - Clique com o botão direito no nome do BD e escolha "Query Tool" para executar código SQL.


pgAdmin 4
pgAdmin
File
Object
Tools
Help
Browser
Dashboard
Properties
SQL
Statistics
Dependencies
Dependents
Servers (1)
PostgreSQL 11
Databases (4)
aula
Casts
Catalogs
Event Triggers
Extensions
Foreign Data Wrappers
Languages
Publications
Schemas (1)
public
Collations
Domains
FTS Configurations
FTS Dictionaries
FTS Parsers
FTS Templates
Foreign Tables
Functions
Materialized Views
Procedures
Sequences
Tables (2)
departamento
funcionario
Trigger Functions
Database sessions
Total
Active
Idle
Transactions per second
Transactions
Commits
Rollback
Tuples in
Inserts
Updates
Tuples out
Fetched
Returned
Block I/O
Reads
Hits
Server activity
Sessions
Locks
Prepared Transactions
Search
PID
User
Application
Client
Backend start
State
Wait event
22632
postgres
pgAdmin 4 - DB:aula
::1
2022-03-22 11:49:22 -03
active

SGBD PostGreSQL

- Bancos de Dados
  - São a estrutura básica de armazenamento.
  - Divisões lógicas das estruturas de dados.
  - Compartilham dados, *views*, etc.
  - Após a instalação, são criados os bancos de dados *postgres*, *template0* e *template1*, de uso interno do SGBD.

```
CREATE DATABASE xxxx;

DROP DATABASE xxxx;
```

SGBD PostGreSQL

- Esquemas
  - Esquemas são criados dentro de um BD.
  - Agrupa objetos do BD (tabelas, visões, funções, etc.) que pertencem a uma mesma aplicação ou objetivo.
  - Por default, é criado um esquema *public.* Se não for especificado um schema para um BD, o *public* é utilizado.

```
CREATE SCHEMA xxxx;
```

  - Para usar o schema:

```
SET SEARCH_PATH TO xxxx;
```

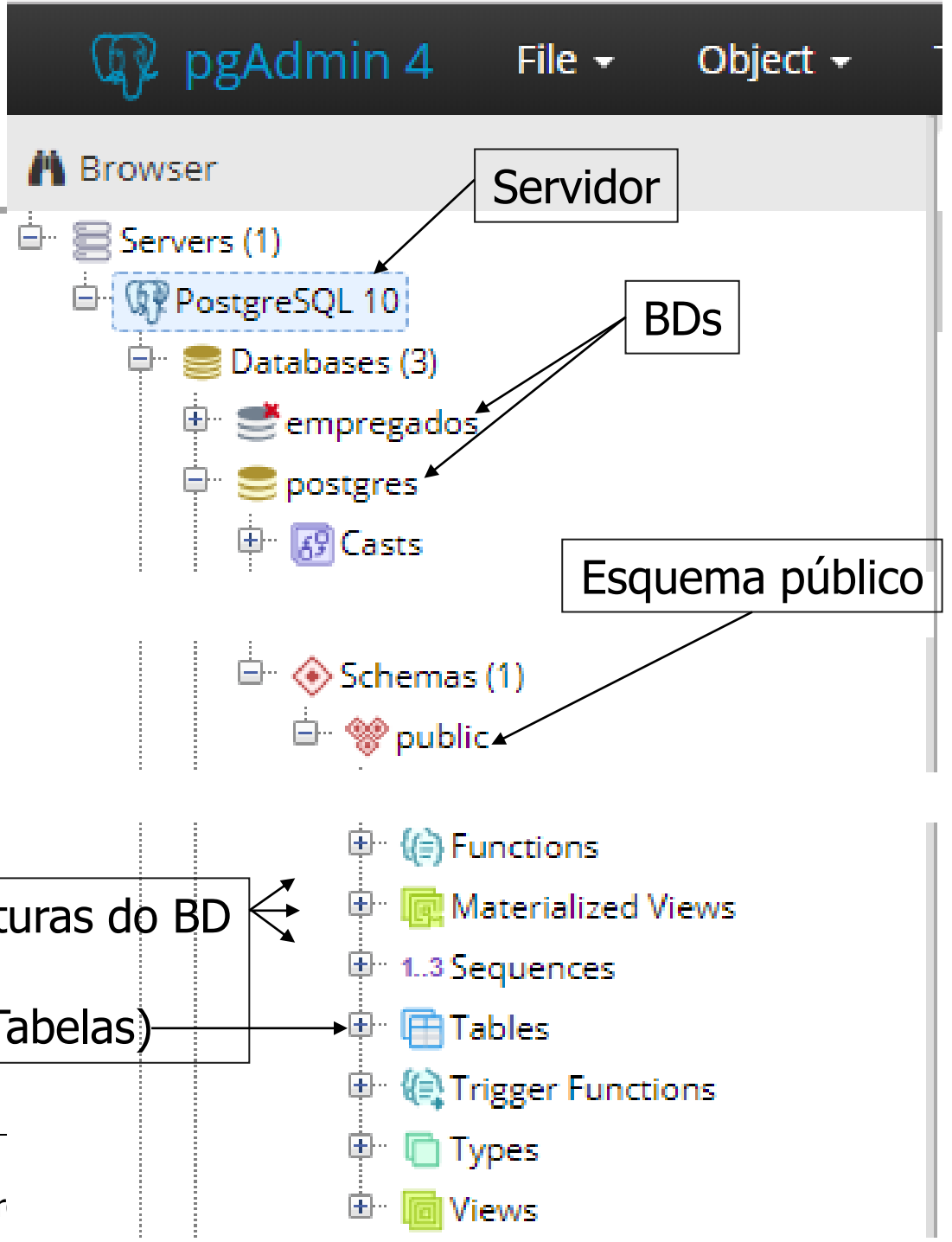
  - Ou referência nas tabelas:

```
CREATE TABLE EMPRESA.FUNCIONARIO...
```

cria a tabela FUNCIONARIO no esquema EMPRESA

PostGreSQL

- Visualização das estruturas do BD no PostGreSQL



SQL

- *Structured Query Language* - Linguagem Estruturada de Consulta.
- Criada em 1976, na IBM.
- Possui comandos para:
  - Definição do esquema do BD.
  - Modificação do BD (inserção, remoção, alteração).
  - Consulta ao BD.

SQL

- Tem vários padrões:
  - ANSI SQL (1986).
  - SQL-89 – inclusão de restrições de integridade.
  - **SQL-92** (ou SQL2) – grande atualização.
  - **SQL-99** (inicialmente SQL3) – inclusão de expressões regulares, consultas recursivas, gatilhos, etc.
  - SQL-2003, SQL-2006, SQL-2008 – inclusão de funcionalidades relacionadas a XML (entre outras).
  - SGBDs geralmente implementam a ANSI SQL e partes da SQL-92 e SQL-99, além de suas próprias extensões.

SQL

- Os comandos existentes na linguagem são subdivididos em dois grupos:
  - **DDL (*Data Definition Language*):** Usada para definição do esquema da base de dados. É o conjunto de comandos responsáveis pela **criação, alteração e remoção da estrutura** das tabelas e índices.
  - **DML (*Data Manipulation Language*):** Usada para programação de consultas e transações que inserem, removem e alteram linhas de tabelas. É o conjunto de comandos responsáveis pela **consulta e atualização dos dados** armazenados em um banco de dados.

SQL/DDL

- *Data Definition Language* - Linguagem de definição de dados
- Comandos principais:
  - **`CREATE TABLE`** - criação de tabelas
    - Atributos (tipos)
    - Restrições
  - **`ALTER TABLE`** - alteração da estrutura de uma tabela
  - **`DROP TABLE`** - exclusão de uma tabela do BD

SQL/DDL

CREATE TABLE

- Objetivo: Criar a estrutura de uma tabela definindo as colunas (campos) e as chaves primárias e estrangeiras existentes.

CREATE TABLE <nome-tabela>
(<nome-coluna> <tipo-do-dado> [restrições-coluna],
<nome-coluna> ... ,
[restrições-tabela]);

- **Exemplo:**

```
CREATE TABLE FUNCIONARIO (
    Nome VARCHAR(50),
    CPF CHAR(11),
    Sexo CHAR,
    Data_nascimento DATE );
```

Testando o exemplo

- Tabela sem chave primária

```
--CREATE DATABASE TESTE;
CREATE TABLE FUNCIONARIO (
      Nome VARCHAR(50),
      CPF CHAR(11),
      Sexo CHAR,
      Data_nascimento DATE );
SELECT * FROM Funcionario; --teste da criação da tabela
DROP TABLE Funcionario;
```

SQL/DDL

- Nome da tabela e das colunas: Não diferencia maiúsculas e minúsculas, sem caracteres especiais (cedilha, espaço, etc.).
- Tipos de Dado:
  - Números inteiros:
    - **INTEGER** ou **INT**
    - **SMALLINT** – ocupa geralmente a metade da quantidade de bytes usada por um INTEGER
  - Números reais:
    - **FLOAT** ou **REAL**
    - **DOUBLE** (não tem no PostGreSQL).
    - **DECIMAL(i,j)** ou **NUMERIC(i,j)** – onde *i* indica o total de dígitos decimais, e *j* indica o número de dígitos após o ponto decimal.

SQL/DDL

- Caracteres:
  - **CHARACTER(n)** ou **CHAR(n)** - cadeia de caracteres de tamanho fixo igual a *n* .
  - **CHAR VARYING(n)** ou **VARCHAR(n)** - cadeia de caracteres de tamanho máximo *n* .

    * O tamanho padrão de *n* é 1.
  - **TEXT** - grandes cadeias de caracteres de tamanho variável.
- Observações:
  - Cadeias de caracteres são delimitadas por aspas simples (apóstrofos).
  - Caracteres em SQL são *case sensitive*. Portanto, 'AS33C' é diferente de 'as33c', mas palavras reservadas da SQL são *case insensitive*, ou seja, podemos usar SELECT ou select.

SQL/DDL

- Tempo:
  - **DATE** - Data completa. Exemplo: '2018-08-14' no formato padrão YYYY-MM-DD.
  - **TIME** - Horário completo. Exemplo: '19:50:25' no formato padrão HH:MM:SS.
  - **TIMESTAMP** - Junção de DATE e TIME. Exemplo: '2018-08-14 18:50:25'.

  * Esses tipos podem ser considerados cadeias de caracteres com formato especial.
- Lógico:
  - **BOOLEAN** - Valores TRUE (ou t) e FALSE (ou f).

SQL/DDL-Restrições de coluna

- Restrição a valores nulos - **NOT NULL**
  - Define que um campo não pode receber o valor NULL.
  - Exige o preenchimento do campo, ou seja, no momento da inclusão é obrigatório que possua um conteúdo.

    * Campos que fazem parte da chave primária da tabela têm essa restrição implícita.
- Valor padrão - **DEFAULT**
  - Define o valor que será atribuído a um campo caso não seja especificado o seu conteúdo na inclusão do registro.
  - Se o campo não possuir a restrição de NOT NULL e nenhum valor padrão for definido para ela, então o valor NULL será usado como padrão.

SQL/DDL-Restrições de coluna

- Restrição de valor - **CHECK**
  - Restringe os valores que um campo pode assumir.
- Exemplo de uso das restrições de coluna:

```
CREATE TABLE FUNCIONARIO (
    Nome VARCHAR(50) NOT NULL,
    CPF CHAR(11) NOT NULL,
    Salario DECIMAL(10,2) DEFAULT 850
    CHECK (Salario > 650 AND Salario < 50000),
    Idade INT CHECK (Idade >= 18 AND Idade <= 120),
    Casado BOOLEAN DEFAULT FALSE );
```

*OBS.: Idade aceita nulo, só faz o CHECK para valores diferentes de null. Campo 'Casado' aceita valor null, DEFAULT é usado se não for informado nenhum valor.

Testando o exemplo

```
CREATE TABLE FUNCIONARIO (
   Nome VARCHAR(50) NOT NULL,
   CPF CHAR(11) PRIMARY KEY,
   Salario DECIMAL(10,2) DEFAULT 850
           CHECK (Salario > 650 AND Salario < 50000),
   Idade INT CHECK (Idade >= 18 AND Idade <= 120),
   Casado BOOLEAN DEFAULT FALSE );
INSERT INTO Funcionario (Nome,CPF,Idade) VALUES ('Ana','11111111111',28);
--valores DEFAULT para salario e casado (valores não informados)
INSERT INTO Funcionario VALUES ('Paulo','22222222222',500,16,true)
--valores fora do domínio CHECK e valor booleano
SELECT * FROM Funcionario;
DROP TABLE Funcionario;
```