

# DEFINIÇÃO

lnstalação do PostgreSQL. tipos de dados. Criação de tabelas. Manipulação de linhas. Controle de transação.

# PROPÓSITO

Compreender a instalação do PostgreSQL e importante para conhecer um ambiente computacional típico de banco de dados em ambientes corporativos. Entender sobre tipos de dados e a base para escolhas compatfveis com a natureza dos dados a serem armazenados. Para desenvolver sistemas com uso de banco de dados, e necessario conhecer os comandos para manipula9ao de linhas nas tabelas, alem de identificar como eles sao controlados a partir do conceito de transa9ao.

# PREPARACAO

Antes de iniciar o conteudo deste tema, certifique-se deter baixado o SGBD PostgreSQL em seu computador.

##### OBJETIVOS

MODULO 1

Compreender o processo de instalac;ao do PostgreSQL

MODULO 2

Empregar comandos para criac;ao e alterac;ao de tabelas

MODULO 3

Empregar comandos para manipular linhas nas tabelas

MODULO 4

Empregar comandos de controle de transac;ao

##### INTRODUCAO

Ao longo deste tema, vamos analisar as características básicas do sistema gerenciador de banco de dados (SGBD) PostgreSQL, envolvendo as etapas utilizadas para instalar esse SGBD

no Linux e no Windows.

0 PostgreSQL e um SGBD de c6digo aberto desenvolvido em linguagem Ce esta disponfvel para ser utilizado em diversos ambientes de sistemas operacionais, tais coma Linux, Unix, Windows, OS X, Solaris, entre outros.

Vamos explorar varios recursos da linguagem SOL, com taco na aprendizagem de comandos classificados coma **DDL.** Aprenderemos tambem comandos **CRUD,** sigla em ingles que faz referencia a quatro opera96es basicas: criagao, consulta, atualizagao e remogao de dados, respectivamente.

Par fim, vamos entender que, internamente, o SGBD trata de diversas opera96es de maneira atomica, ou seja, um conjunto de comandos deve ser executado coma uma unidade 16gica de trabalho, ou nenhuma operagao deve ser realizada. Trata-se do conceito de transagao.

Aprenderemos, entao, os principais comandos que gerenciam transa96es.

**Clique aqui** para baixar o arquivo com todos os c6digos que serao utilizados nas consultas dos m6dulos deste tema.



**DDL**

*Data Definition Language,* ou Linguagem de Definic;ao de Dados.

**CRUD**

Create - Read - Update - Delete.

##### MODULO 1

0 Compreender o processo de instalac;ao do PostgreSQL

##### BREVE HIST6RICO

0 PostgreSQL surgiu a partir de um projeto denominado POSTGRES, assim denominado parser originario do projeto INGRES (Post INGRES), de responsabilidade da Universidade da California em Berkeley.

A implementac;ao do POSTGRES foi iniciada em 1986, tornando-se operacional em 1987. Sua primeira versao foi lanc;ada ao publico externo em 1989. Nos dais anos seguintes, foram lanc;adas a segunda e terceira versoes.

Em 1995, foi disponibilizado o Postgres95, com revisao no c6digo do projeto ea adoc;ao da linguagem **SQL** coma interface padrao.

Em 1996, o produto foi renomeado para PostreSQL, comec;ando pela versao 6, considerado continuidade do Postgres95, a versao 5. 0 projeto ganhou visibilidade e, atualmente, o

PostgreSQL e conhecido como um dos principais SGBDs de c6digo aberto, com vers6es para

Windows, Mac OS e Linux.

**SQL**

Structured Query Language, ou Linguagem de Consulta Estruturada.



Wright Studio/Shutterstock

##### ARQUITETURA DO POSTGRESQL

Antes de instalarmos o PostgreSQL, e importante entendermos sua arquitetura basica. 0 PostreSQL utiliza o modelo cliente-servidor. Sob esse contexto, destacamos os seguintes

processos que cooperam entre si:

Gerenciar os arquivos do banco de dados

**Processo servidor, responsavel por func;oes, tais como:**

**Aplicativo cliente do usuario, responsavel por fum;oes, tais como:**

Avaliar e executar no banco de dados os comandos submetidos pelos clientes

Gerenciar as conexoes entre os aplicativos e o SGBD

Enviar comandos para consulta em uma ou diversas tabelas, com objetivo de

recuperar informai;oes do banco de dados

Enviar comandos para manipulai;ao de linhas em tabelas. A manipulai;ao pode

envolver, por exemplo, inseri;ao, alterai;ao ou mesmo remoi;ao de dados

Solicitar acesso ao SGBD

**Atem;ao!** Para visualizai;ao completa da tabela utilize a rolagem horizontal

Em um ambiente cliente-servidor, tanto o cliente quanto o servidor podem estar localizados em maquinas diferentes, em uma rede local ou mesmo geograficamente distantes.

Em geral, a comunicai;ao entre cliente e servidor ocorre por meio de uma conexao de rede utilizando o protocolo **TCP/IP.** Esse protocolo tern por objetivo a padronizai;ao das comunicai;oes em rede, em especial as comunicai;oes na Web.

**TCP/IP**

0 TCP/IP e um conjunto de protocolos de comunicai;ao entre computadores em rede. Seu

nome faz referencia a dois protocolos: Transmission Control Protocol/ Internet Protocol.

**Fonte:** Wikipedia

0 PostgreSQL suporta varias conex6es simultaneas de clientes. Para isso, um processo para cada conexao e iniciado. Em seguida, o cliente e o novo processo realizam comunica9ao.

Como instalar o PostgreSQL em seu computador?

**tit RESPOSTA**

Recomendamos que voce acompanhe as vers6es do PostgreSQL na pagina oficial do produto para escolher uma versao de acordo com seu sistema operacional e que lhe interesse.

##### INSTALACAO DO POSTGRESQL NO LINUX

Para instala9ao em Linux com c6digo fonte, voce pode executar os passos a seguir:

FAZER DOWNLOAD DO ARQUIVO .TAR.GZ

Se considerarmos o Ubuntu, o download pode ser realizado a partir do site do PostgreSQL, na sec;ao relativa a esse sistema operacional. Usaremos o qualificador "--" para comentarios a respeito dos comandos.

OBTER O CODIGO FONTE

No prompt do Linux, executar as dais comandos a seguir:

**gunzip postgresql-12.3.tar.gz**

-- **descompacta o arquivo .gz gerando o arquivo .tar tar xf postgresql-12.3.tar**

-- **abre o arquivo .tar criando o diret6rio postgresql-12.3**

ACESSAR O DIRETORIO CRIADO PELO TAR

Ap6s a conclusao da etapa anterior, deve-se executar o comando a seguir para ter acesso ao diret6rio criado pelo tar:

**cd postgresql-12.3**

-- **cd (change directory)**

REALIZAR O PROCESSO DE INSTALACAO

Ap6s a conclusao da etapa anterior, e precise executar as comandos a seguir para realizar o processo de instalac;ao do PostgreSQL:

**./configure**

-- **script para configurar a arvore de diret6rios (cria o diret6rio /usr/local/psql) gmake**

-- **GNU make: inicializa o build, pode levar de 5 a 30 minutos e termina com a mensagem:**

**-All of PostgreSQL is successfully made. Ready to install.**

**SU**

-- **muda login de usuario para o superusuario root (pede a senha do root) gmake install**

-- **realiza a instala ao como root**

CHECAR A INSTALACAO

Ap6s a conclusao da etapa anterior, deve-se executar as comandos a seguir para checar instalac;ao do PostgreSQL:

**adduser postgres**

-- **cria usuario postgres, superusuario do PostgreSQL (seuseradd no Fedora) mkdir /usr/local/pgsql/data**

-- **cria o diret6rio data onde ficarao as bases de dados chown postgres /usr/local/pgsql/data**

-- **muda o dono do diret6rio data para postgres**

**su** - **postgres**

-- **muda login de usuario para postgres**

**/usr/local/pgsql/bin/initdb -D /usr/local/pgsql/data**

-- **cria um grupo de BD no diret6rio data**

**/usr/local/pgsql/bin/postmaster -D /usr/local/pgsql/data >logfile 2>&1** &

-- **inicializa o servidor em segundo piano**

**/usr/local/pgsql/bin/createdb test**

-- **cria um database test**

**/usr/local/pgsql/bin/psql test**

-- **inicia uma sessao no PostgreSQL, usando a interface de linha de comandos psql, como usuario postgres no database test**

##### INSTALACAO DO POSTGRESQL NO WINDOWS

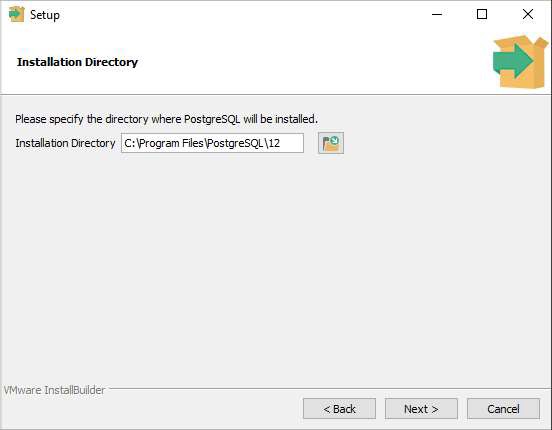
ATENCAO

0 procedimento para instalai;ao do PostgreSQL no sistema operacional Windows e bastante trivial e basicamente segue o padrao *(Next---+ Next---+ .. + Finish).*

Ap6s fazer o download do arquivo instalador para Windows (postgresql-12.3-1-windows-x64.exe, com cerca de 195 MB, no caso da versao 12), deve-se executar o arquivo como usuario administrador. Ap6s a tela de inicializai;ao da instalai;ao, sera mostrada a de localizai;ao do

diret6rio de instalai;ao que, por padrao, criara a pasta C:\Program Files\PostgreSQL\12, onde 12

e a versao do PostgreSQL.



Fonte: Software de instalagao do PostgreSQL para Windows

**i'I** Tela de definigao do diret6rio de instalagao.

A seguir, o instalador perguntara quais componentes serao instalados junto com o servidor PostgreSQL. Por padrao, sao instalados:

0 PGADMIN

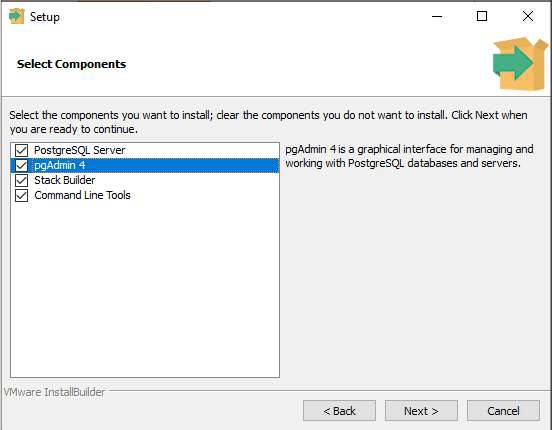
Uma interface grafica de administragao;

OPSQL

Uma ferramenta de linha de comando para administragao;

0 STACK BUILDER

Uma ferramenta util para gerenciar a instalagao de m6dulos complementares, tais como utilitarios, drivers e extensoes.



Fonte: Software de instalai;ao do PostgreSQL para Windows

**m** Tela de selei;ao de componentes a instalar.

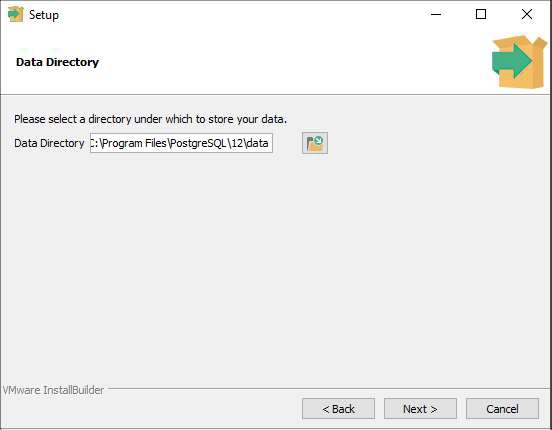
Em seguida, o instalador determina o diret6rio onde ficarao armazenados os dados no seu

servidor. Na instalai;ao padrao do Windows, os dados ficam armazenados no diret6rio C:\Program Files\PostgreSQL\12\data. Nesse diret6rio, e criado um subdiret6rio C:\Program Files\PostgreSQL\12\data\base, dentro do qual sera criada uma pasta numerada para cada

*database,* a comei;ar pelo *database* padrao do servidor, denominado postgres, criado com a instalai;ao.

ATENCAO

Cada pasta correspondente a um *database* armazena arquivos numerados contendo metadados do catalogo do SGBD, assim como arquivos numerados para cada tabela criada dentro do *database.*



Fonte: Software de instalac;ao do PostgreSQL para Windows

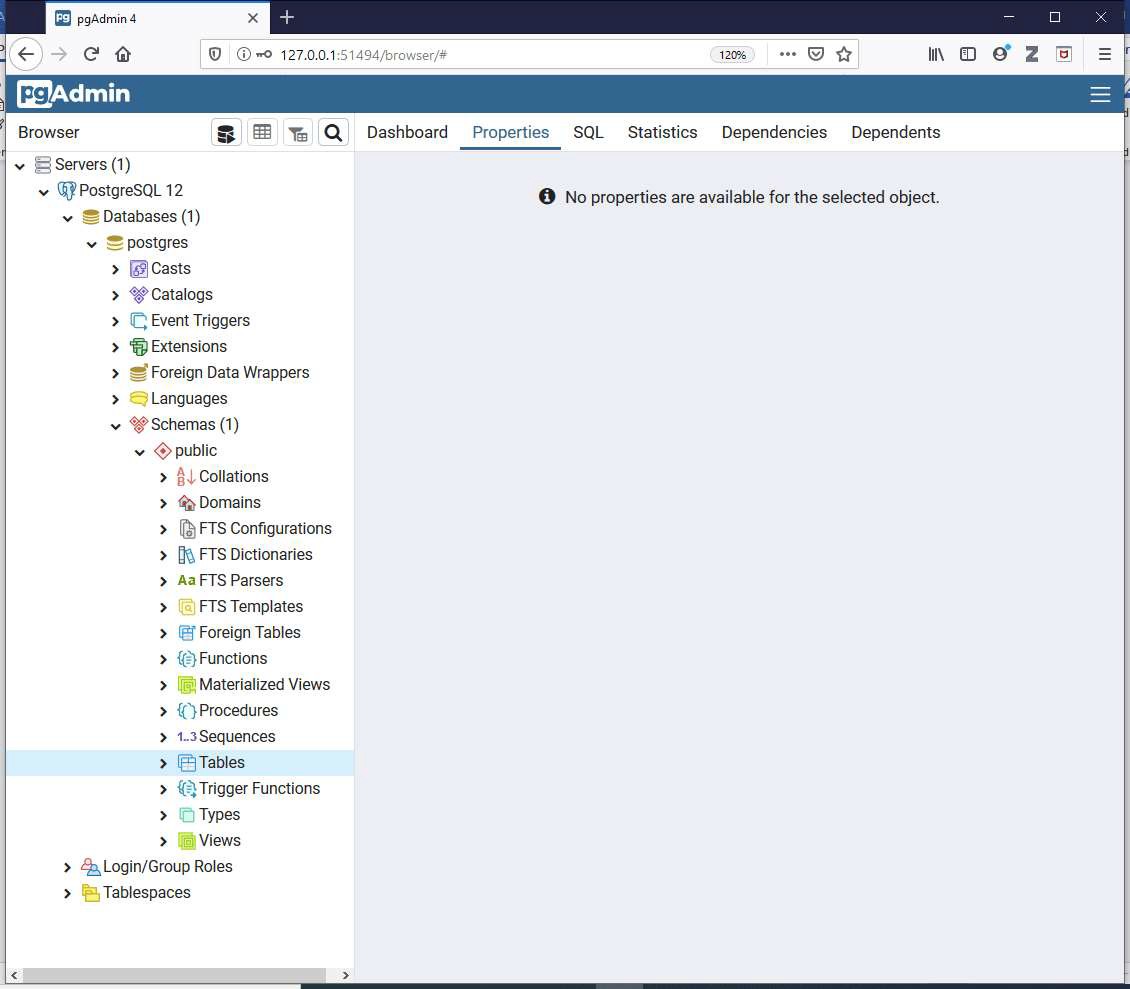
II Tela de localizac;ao do diret6rio de dados.

Conclufdo o processo de instalac;ao do PostgreSQL, sera possfvel visualizar no pgAdmin 4 a arvore de diret6rios da instalac;ao padrao, contendo:

Servers (1) PostgreSQL 12

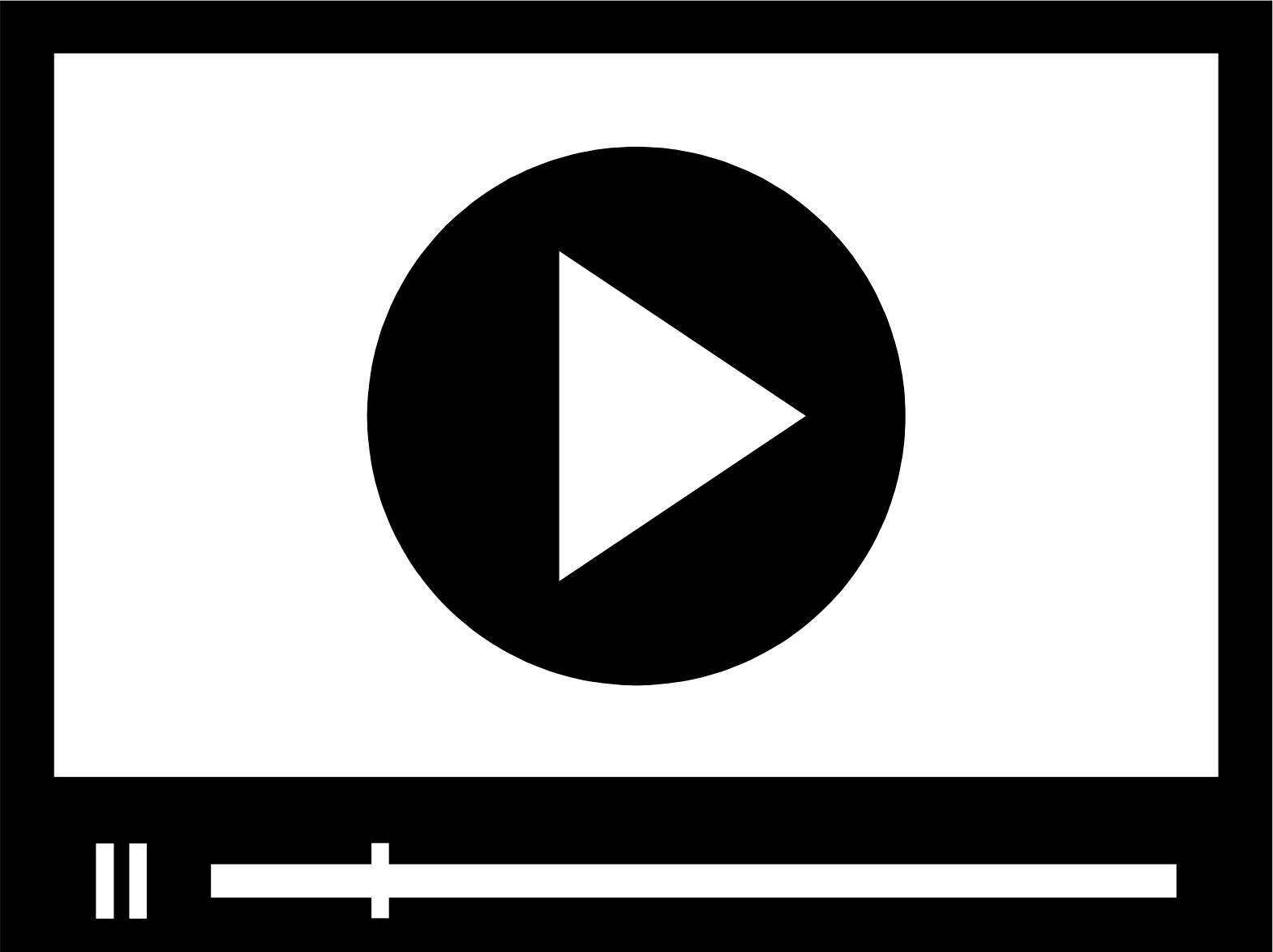
Databases (1) postgres

Schemas (1) public Tables



Fonte: Software pgAdmin 4

**aJ** Tela do pgAdmin 4 com a arvore de diret6rios da instalac;ao padrao do PostgreSQL.



##### INSTALACAO DO POSTGRESQL NO LINUX E WINDOWS



**INTERFACES PARA INTERAGIR COM 0 POSTGRESQL**

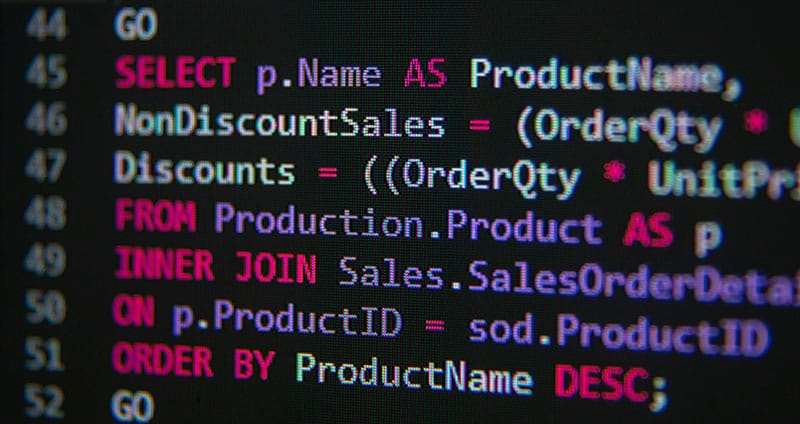
**1::J ATENCAO**

Ao longo deste e dos pr6ximos m6dulos, sera necessario o uso de algum tipo de interface que permita conexao ao servidor e, em seguida, o acesso aos objetos de interesse.

Alem do pgAdmin 4, a interface grafica pr6pria que prove acesso aos recursos do SGBD via navegador, o PostgreSQL disponibiliza o psql, uma interface de linha de comando sobre a qual o usuario submete interativamente comandos ao SGBD, via terminal.

0 PostgreSQL possui uma excelente documentai;ao disponfvel on-line, aplicavel tanto para instalai;ao em Linux quanto para Windows, considerada uma verdadeira enciclopedia de bancos de dados relacionais. Essa documentai;ao e valida para uso dos recursos do PostgreSQL atraves de quaisquer interfaces.

Alternativamente, voce pode optar por baixar e usar interfaces projetadas por outros desenvolvedores. Por exemplo, o aplicativo *DBeaver* possui uma versao livre com excelentes funcionalidades. Trata-se de um aplicativo util no desenvolvimento de atividades de administrai;ao de banco de dados.



Fonte: EvalCo/Shutterstock

**CRIANDO *DATABASES* COM O PGADMIN 4 E COMO PSQL**

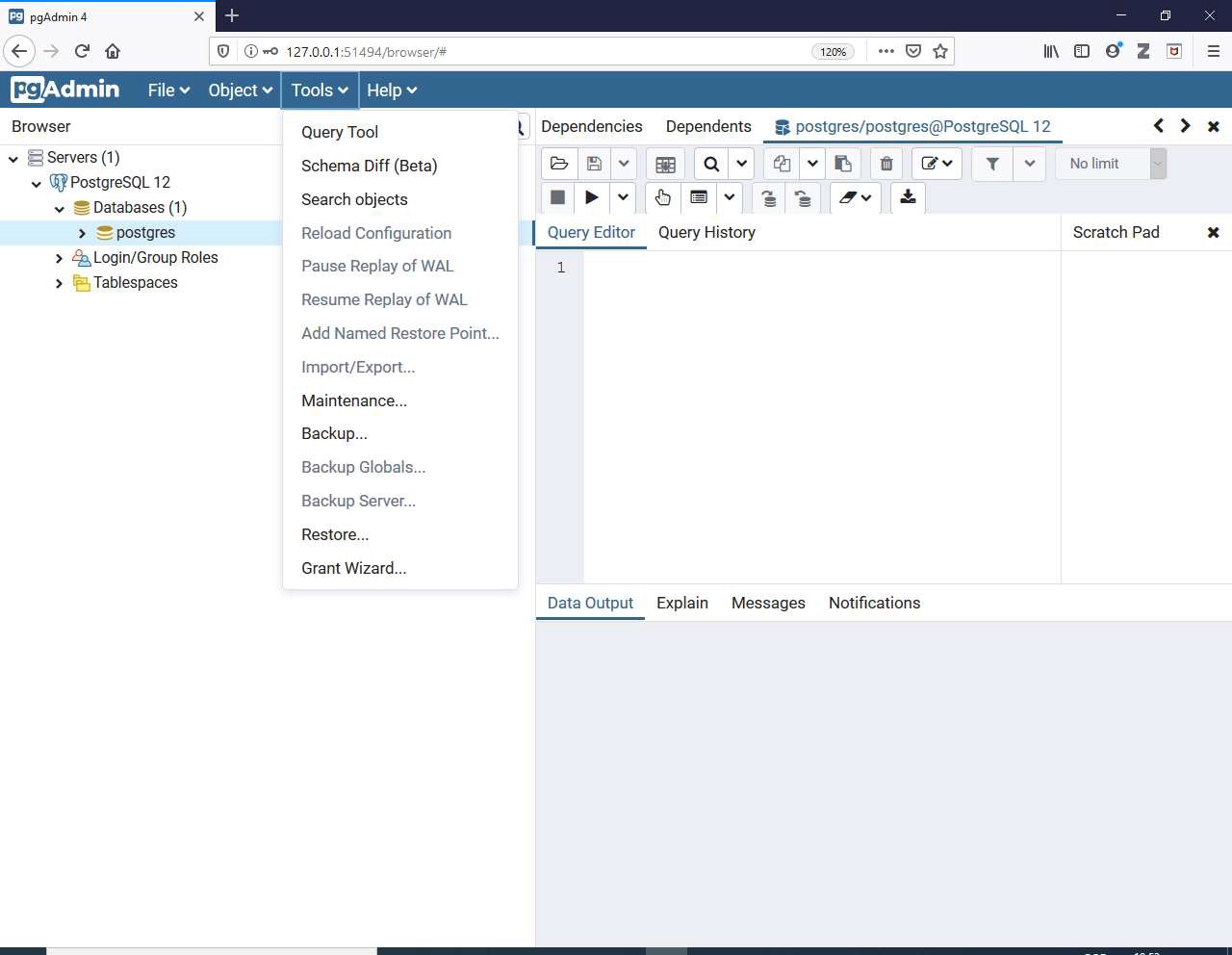
Tendo instalado o PostgreSQL, para nos certificarmos de que o SGBD esta funcionando de maneira adequada, realizaremos um teste envolvendo a criac;ao de *databases,* conforme a seguir:

*database* BDTESTEPGADMIN, a ser criado usando o pgAdmin 4.

*database* BDTESTEPSQL, a ser criado usando o psql

**Aten ao!** Para visualizac;ao completa da tabela utilize a rolagem horizontal

No Windows, selecione o botao lniciar, digite "pgAdmin 4" e tecle <enter>. Em seguida, o navegador sera aberto, e voce tera acesso a um ambiente onde aparece um (mico *database* denominado *postgres,* criado pelo instalador.



Fonte: Software pgAdmin 4

m Tela do pgAdmin 4 com o *database postgres* criado pelo instalador.

Como foco no *database postgres,* clique em Tools e em Query Tool para abrir um editor *(Query Editor)* onde voce codificara e submetera (utilizando a tecla F5 ou o botao correspondente com uma seta) comandos SQL ao servidor.

Vamos criar o *database* BDTESTEPGADMIN a partir do ambiente do pgAdmin 4. Digite o c6digo a seguir no *Query Editore,* em seguida, pressione a tecla F5 para executar o comando:

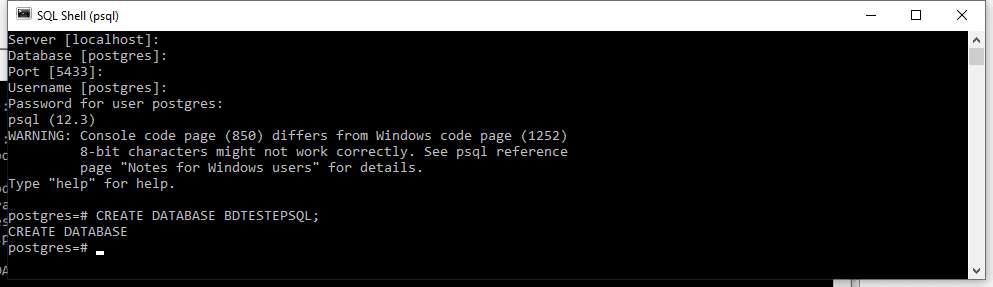
CREATE DATABASE BDTESTEPGADMIN;

Alternativamente, o *database* poderia ser criado usando a interface do pgAdmin 4, pressionado o botao direito do mouse sobre a pasta Databases e clicando em Create.

Agora, vamos criar um *database* usando a interface de terminal psql. No Windows, a partir do botao lniciar, digite "psql" e tecle <enter>. Logo um terminal sera aberto e voce executara o seguinte comando (teclando <enter> ap6s o comando):

CREATE DATABASE BDTESTEPSQL;

Ap6s o SGBD executar o comando, a tela do psql ficara conforme vemos a seguir:

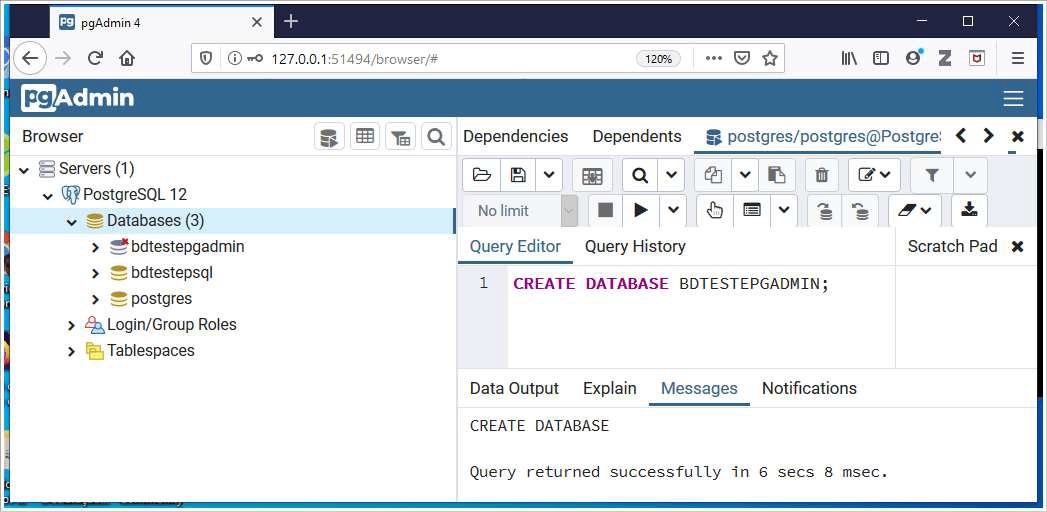


Fonte: Software psql

**m** Tela do psql ap6s criagao do *database* BDTESTEPSQL.

Ao final desse processo, podemos verificar, no pgAdmin 4, a criagao dos dois *databases*

conforme mostrado na imagem abaixo.

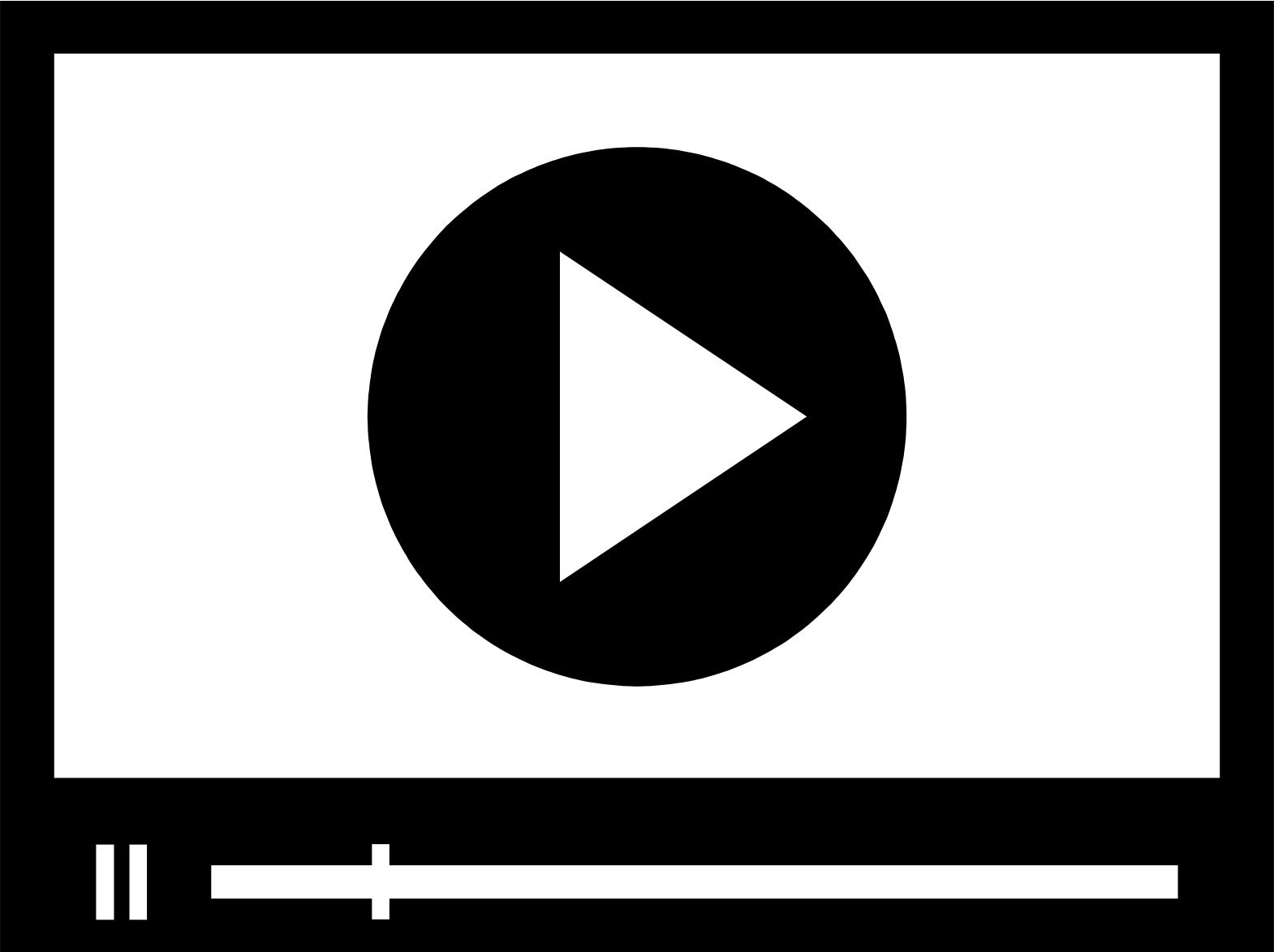


Fonte: Software psql

**m** Tela do pgAdmin 4 mostrando os *databases* recem-criados

Neste modulo, apresentamos um breve hist6rico a respeito do PostgreSQL, alem de detalhes sobre a arquitetura deste SGBD. Em seguida, mostramos o processo de instalagao do SGBD no Linux e no Windows. Finalmente, verificamos a criagao de *databases* utilizando as ferramentas pgAdmin 4 e psql, com objetivo de confirmarmos que a instalagao foi realizada de maneira correta.

Maiores detalhes sobre a utilizagao do pgAdmin 4 e do psql podem ser pesquisados nas respectivas documentag6es que acompanham os softwares.



CRIACAO DE *DATABASES* NO POSTGRESQL



#### VERIFICANDO O APRENDIZADO

1. **ACERCA DO SGBD POSTGRESQL, ASSINALE A PROPOSICAO VERDADEIRA.**
   1. 0 PostgreSQL e um SGBD comercial de c6digo fechado, disponfvel apenas para ambiente Windows.
   2. 0 PostgreSQL e um SGBD de c6digo aberto, com versoes compatfveis com diversos sistemas operacionais, tais como Windows, MAC OS e diversas distribuic;oes Linux.
   3. 0 PostgreSQL e um SGBD puramente orientado a objeto.
   4. 0 PostgreSQL e um SGBD puramente relacional.
2. **ANALISE AS SEGUINTES PROPOSICOES A RESPEITO DO POSTGRESQL:**
3. - **0 COMANDO "CREATE USER BTESTE SUPERUSER INHERIT CREATEDB CREATEROLE;** " **CRIA UM BANCO DE DADOS DENOMINADO BTESTE.**
4. **-AO INSTALAR O POSTGRESQL VERSAO 12 NO WINDOWS, A PASTA PADRAO DE INSTALACAO E "C:/PROGRAM FILES/POSTGRESQL/12".** Ill - **0 COMANDO "CREATE TABLE CLIENTE (CODIGOC INT NOT NULL,**

NOME CHAR(80), CONSTRAINT CHAVECLIENTE **PRIMARY** KEY (CODIGOC));" CRIAUMATABELADENOMINADACLIENTE COM AS

COLUNAS CODIGOC E NOME, SENDO QUE A COLUNA CODIGOC E

CHAVE ESTRANGEIRA.

IDENTIFIQUE AS PROPOSICOES FALSAS:

* 1. I, lie Ill.
  2. le Ill.
  3. le II.
  4. II e Ill.

GABARITO

* + 1. **Acerca do SGBD PostgreSQL, assinale a proposi ao verdadeira.**

A alternativa **118** 11 esta correta.

De fato, o PostgreSQL e um SGBD livre e esta disponivel para funcionamento em diversas plataformas de sistemas operacionais.

* + 1. **Analise as seguintes proposi oes a respeito do PostgreSQL:**

1. - **0 comando "create user bteste superuser inherit createdb createrole; "cria um banco de dados denominado bteste.**
2. **-Ao lnstalar o PostgreSQL versao 12 no Windows, a pasta padrao de instala ao e**

**"C:/Program Files/PostgreSQL/12".**

Ill - **0 comando "create table cliente (codigoc int not null, nome char(80), constraint**

**chavecliente primary key (codigoc));" cria uma tabela denominada cliente com as colunas codigoc e nome, sendo que a coluna codigoc e chave estrangeira.**

**ldentifique as proposi oes falsas:**

A alternativa **118** 11 esta correta.

0 comando expresso na primeira proposigao cria um usuario denominado bteste, e nao um

banco de dados. 0 comando expresso na terceira proposigao cria uma tabela denominada cliente. No entanto, a coluna codigoc e chave primaria, e nao chave estrangeira.

# M0DUL02

0 Empregar comandos para criagao e alteragao de tabelas

##### BREVE HIST6RICO DA SQL

A SQL foi criada na IBM na decada de 1970, sendo originalmente chamada de

SEQUEL , inspirada, principalmente, na aparente

(Strucutured English Query Language)

facilidade de uso do comando SELECT para consulta a tabelas dos bancos de dados relacionais.

Com a evoluc;ao dos sistemas gerenciadores de banco de dados (SGBDs), diversas empresas lanc;aram produtos incorporando funcionalidades a SQL, o que ocasionou problemas de compatibilidade. Buscando uma soluc;ao, o instituto **ANSI** definiu padroes para a linguagem SQL,

a qual passou a ser referenciada ANSI-SQL.

Atualmente, ha diversos SGBDs compatfveis com o padrao ANSI SQL, que vai muito alem de consultas com o comando SELECT, englobando sublinguagens para definic;ao de dados (CREATE, ALTER, DROP) e para manipulac;ao de dados (INSERT, UPDATE, DELETE), alem de

comandos de controle tfpicos para administrac;ao do banco de dados. Ao mesmo tempo, varios produtos de SGBDs relacionais apresentam extensoes a linguagem, como modo de facilitar o dia a dia dos desenvolvedores.

**ANSI**

American National Standards Institute, tambem conhecido por sua sigla ANSI, e uma organizac;ao particular norte-americana sem fins lucrativos que tern por objetivo facilitar a padronizac;ao dos trabalhos de seus membros.

**Fonte:** Wikipedia

##### ACESSO AO POSTGRESQL

Quero criar tabelas em um SGBD PostgreSQL. Por onde comec;o?

Executando o pgAdmin 4, o navegador sera aberto, e voce tera acesso ao ambiente que permite manipular os objetos do PostgreSQL.

Depois, escolha um *database* na hierarquia e de um clique com o botao inverso do mouse. Em seguida, escolha a opc;ao *Query Tool.*

! **DICA**

O pgAdmin ea interface Web padrao do PostgreSQL, mas voce pode escolher o utilitario de sua preferencia para praticar os comandos que aprenderemos ao longo das pr6ximas sec;oes.

##### CRIANDO UM BANCO DE DADOS

Ap6s escolher um utilitario para acessar o servidor PostgreSQL, sera necessario criar um *database* para, em seguida, manipular tabelas. Por exemplo, para criarmos um *database* denominado bdestudo, e necessario executar o comando a seguir:

-- Comando para criar um *database.*

CREATE DATABASE BDESTUDO;

No comando acima, a linha com"--" corresponde a comentario e seu conteudo nao e processado pelo SGBD. Caso haja necessidade de remover o *database* bdestudo, basta executar o comando a seguir:

-- Comando para remover um *database.*

DROP DATABASE BDESTUDO;

Antes de prosseguirmos com a criac;ao de tabelas, principal objetivo deste modulo, e preciso compreender que todo *database* criado no PostgreSQL possui um *schema* padrao denominado *public,* onde as tabelas a serem criadas no *database* serao armazenadas. Assim, se nao especificarmos a qual *schema* do *database* pertence uma tabela que estamos criando, esta sera armazenada no *schema public.* Para especificarmos um *schema* diferente do *public,* antes de criar uma tabela, devemos criar o respectivo *schema,* com o comando CREATE SCHEMA.

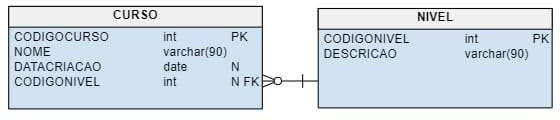
\* **EXEMPLO**

CREATE SCHEMA esquema;

A partir de entao, qualquer tabela pertencente ao *schema* esquema devera ser especificada pelo seu name completo: esquema.tabela

##### CRIANDO TABELAS

Ja sabemos que um banco de dados, em geral, possui diversas tabelas. As tabelas sao criadas com auxnio do comando CREATE TABLE. Usaremos esse comando para implementar o modelo expresso na figura a seguir, dentro do *database* bdestudo anteriormente criado:



Fonte: Fonte: 0 autor

**aJ** Modelo relacional com as tabelas CURSO e NIVEL.

Veja a seguir a sintaxe **basica** do comando CREATE TABLE:

CREATE TABLE NOMETABELA ( COLUNA1 - TIPODEDADOS [RESTRl<;AO], COLUNAN-TIPODEDADOS ESTR A , PRIMARY KEY (COLUNA),

FOREIGN KEY (COLUNA) REFERENCES NOMETABELA (COLUNA) CONSTRAINT RESTRl<;AO);

Vamos agora analisar o significado de cada item na sintaxe apresentada anteriormente:

**NOMETABELA** representa o name da tabela que sera criada

|  |  |
| --- | --- |
| **COLUNA1 e COLUNAN** | representa a(s) coluna(s) da tabela |
| **TIPODEDADOS** | indica tipo de dados ou domfnio da coluna |
| **RESTRl<;AO** | aponta alguma propriedade associada a coluna em questao.  Par exemplo, podemos definir sea coluna e obrigat6ria ou opcional |
| **PRIMARY KEY** | indica a coluna, ou conjunto de colunas, representativa da chave primaria |
| **FOREIGN KEY** | sinaliza a coluna, ou conjunto de colunas, com restric;ao de chave estrangeira |
| **CONSTRAINT RESTRl<;AO** | indica alguma restric;ao que podera ser declarada |
| A sintaxe **completa** a respeito do comando CREATE TABLE no PostgreSQL pode ser encontrada no site oficial do PostgreSQL. Ao final da sintaxe, sao descritas as caracterfsticas compatfveis com o padrao SQL. | |

**Aten ao!** Para visualizac;ao completa da tabela utilize a rolagem horizontal

### TIPOS DE DADOS

Cada coluna de tabela deve pertencer a um tipo de dados. No PostgreSQL, os tipos mais comuns sao:

|  |  |
| --- | --- |
| **bigint** | valores inteiros compreendidos entre  -9.223.372.036.854.775.808 e 9.223.372.036.854.775.807 |
| **char(comprimento)** | util para sequencias de caracteres de tamanho fixo. 0 parametro comprimento determina o valor da sequencia. Esse tipo de dado preenche a coluna com espac;os em branco ate completar o total de caracteres definidos, caso a totalidade do tamanho do campo nao esteja preenchida |
| **date** | data de calendario no formato AAAA-MM-DD |
| **decimal** | determina a precisao do valor de casas decimais |
| **double** | precisao do valor de ate 15 casas decimais |
| **int ou integer** | valores inteiros compreendidos entre -2.147.483.648 e 2.147.483.647 |
| **money** | valores monetarios compreendidos entre - 92.233.720.368.547.758.08 e 92.233.720.368.547.758.07 |
| **numeric** | precisao do valor de casas decimais |
| **real** | precisao do valor de ate seis casas decimais |

|  |  |
| --- | --- |
|  |  |
| **serial** | gera valor unico inteiro sequencial para um novo registro entre 1 e 2.147.483.647 |
| **smallint** | representa valores compreendidos entre 32.768 e 32.767 |
| **time** | representa horario no intervalo de tempo entre 00:00:00 e 24:00:00 |
| **varchar(comprimento)** | util para sequencia de dados de caracteres com comprimento variavel. Nao armazena espac;os em branco nao utilizados para compor *string* (em branco) em seu lado direito |

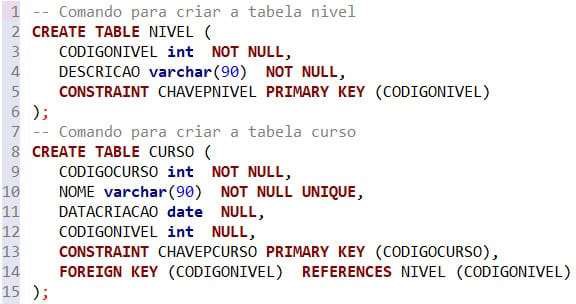
**Aten ao!** Para visualizac;ao completa da tabela utilize a rolagem horizontal

! **DICA**

Para maiores detalhes sabre os tipos de dados, sugerimos que voce acesse a documentac;ao do PostgreSQL, disponivel no site oficial do produto.

###### EXEMPLO ENVOLVENDO CRIACAO DE TABELAS

Veja a seguir o c6digo SQL que permite a criac;ao das tabelas NIVEL e CURSO, respectivamente:



Fonte: 0 autor

A tabela NIVEL esta especificada no bloco de comandos entre as linhas 2 e 6. As duas colunas da tabela sao obrigat6rias.

A tabela CURSO esta especificada no bloco de comandos entre as linhas 8 e 15. As

colunas DATACRIACAO e CODIGONIVEL sao opcionais. Em especial, CODIGONIVEL significa que um curso pode ser criado e, em outro momenta, ser associado a informac;ao que caracteriza o nfvel dele.

**Aten ao!** Para visualizac;ao completa da tabela utilize a rolagem horizontal

**'l::l ATENCAO**

Note que, como nao foi especificado um *schema* para armazenar essas tabelas, elas pertencerao ao *schema* padrao *public* do *database* bdestudo.

Finalmente, observe que, na linha 5, a coluna NOME foi declarada com o qualificador UNIQUE. Na pratica, o SGBD controlara a propriedade de unicidade na referida coluna, proibindo que haja mais de uma ocorrencia da mesma ao longo de todo o ciclo de vida do banco de dados.

**GERENCIAMENTO DE *SCRIPTS* NA PRATICA**

Note que o *script* SQL acima possui 15 linhas. Esse e um exemplo didatico que possui somente duas tabelas. No entanto, voce vai perceber que no dia a dia os projetos reais possuem quantidade de tabelas que facilmente podem ultrapassar dois dfgitos.

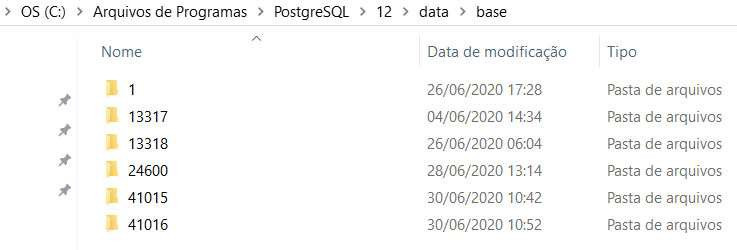
E importante que voce conhec;a comandos DDL, mas o uso pratico ocorrera com auxmo de ferramentas que automatizam o processo de gestao e administrac;ao de dados. Tais ferramentas permitem - manipulando elementos visuais - criar tabelas, relacionamentos e restri96es, alem de

executar outras atividades. As ferramentas permitem gerar c6digo DDL referente ao projeto como um todo, ou parte dele. Sao exemplos: DBeaver, Vertabelo, SQL Power Architect, Toad for SQL, Erwin, entre outras.

###### SGBD POSTGRESQL NOS BASTIDORES

Aprendemos a criar um *database* com o uso do comando CREATE DATABASE. No entanto, a submissao de um simples CREATE DATABASE ao PostgreSQL gera uma serie de etapas gerenciadas pelo servidor.

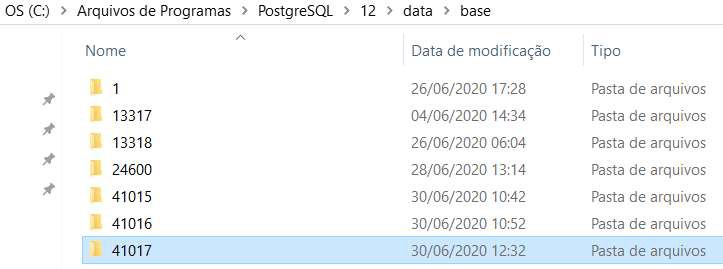
Se considerarmos a instalac;ao padrao no Windows, ao criar um *database* o servidor cria uma pasta identificada por um numero (OID), dentro do diret6rio C:\Program Files\PostgreSQL\12\data\base.



Fonte: 0 autor

**m** Pastas representativas de *databases,* antes da criac;ao do *database* TESTEBANCO.

Ap6s a execuc;ao do comando CREATE DATABASE TESTEBANCO; foi criada a pasta 41017, conforme imagem a seguir:



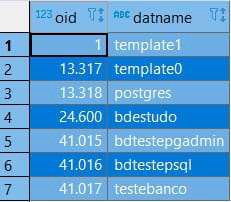
Fonte: 0 autor

m Pastas representativas de *databases,* ap6s da criac;ao de TESTEBANCO.

Ainda, o PostgreSQL mantem informac;6es sobre todos os *databases* em um *database* especial denominado catalogo, cujas tabelas possuem nomes iniciados com o prefixo PG\_. Por exemplo, informac;6es sobre os *databases* existentes em um servidor sao armazenadas na tabela PG\_DATABASE. Assim, caso voce queira identificar o nome correspondente ao 01D do PostgreSQL, basta executar o comando a seguir:

SELECT 01D, DATNAME FROM PG\_DATABASE;

0 resultado desse comando pode ser visualizado na tabela a seguir:



Fonte: 0 autor

l'a Resultado de consulta a tabela PG\_DATABASE.

Precisamos ressaltar que, em seu computador, o resultado podera ser diferente do apresentado, tendo em vista as operac;6es que voce tenha realizado no SGBD.



Fonte: Alexander Supertramp/Shutterstock

### ALTERACAO DE TABELA

Voce vai se deparar com situac;oes em que sera necessario alterar a estrutura de uma tabela ja existente.

Vamos estudar um exemplo?

Suponha que surgiu a necessidade de modelar a informac;ao sabre a data de primeiro reconhecimento do curso. Podemos, entao, alterar a estrutura da tabela CURSO, adicionando uma coluna opcional denominada DTRECONH. 0 comando ALTER TABLE e util no contexto

dessa tarefa.

A seguir, sintaxe **basica** do comando ALTER TABLE:

ALTER TABLE <NOMETABELA> ADD <COLUNA><TIPODEDADOS>;

Na sintaxe apresentada:

<NOMETABELA>

Representa o nome da tabela sabre a qual havera a modificac;ao.

<COLUNA>

Representa o nome da coluna.

<TIPODEDADOS>

Representa o domfnio da coluna.

A sintaxe **completa** a respeito do comando ALTER TABLE pode ser encontrada no site oficial do PostgreSQL.

A seguir, veja o c6digo SQL que permite a alterac;ao da tabela CURSO:

-- Comando para alterar a tabela CURSO, adicionando coluna DTRECONH ALTER TABLE CURSOADD DTRECONH DATE;

Por padrao, o SGBD cria a coluna como opcional. E como se o comando estivesse com o qualificador NULL declarado imediatamente antes do";".

E se desejarmos remover uma coluna da tabela? Podemos usar a sintaxe a seguir: ALTER TABLE <NOMETABELA> DROP <COLUNA>;

Vamos agora remover a coluna DTRECONH da tabela CURSO. A seguir, c6digo SQL que permite a alterac;ao:

-- Comando para alterar a tabela CURSO, removendo a coluna DTRECONH ALTER TABLE CURSO DROP DTRECONH;

###### REMOCAO DE TABELA

Voce vai se deparar com situac;6es em que sera necessario remover uma tabela do banco de dados. Esta ac;ao e realizada com o auxnio do comando DROP TABLE, cuja sintaxe esta expressa a seguir:

DROP TABLE <NOMETABELA>;

Vamos agora remover a tabela CURSO com o devido c6digo SQL:

-- Comando para remover a tabela CURSO DROP TABLE CURSO;

Agora, vamos conhecer alguns cuidados que devem ser tomados quando formos manipular a estrutura de tabelas relacionadas.

### CRIACAO E ALTERACAO DE TABELAS RELACIONADAS

No exemplo envolvendo cria9ao de tabelas, o relacionamento entre as tabelas NIVEL e CURSO foi declarado no bloco CREATE TABLE da tabela CURSO (linha 14). No entanto, n6s poderfamos ter optado por criar as tabelas NIVEL e CURSO sem relacionamento, para, em seguida, alterar a tabela CURSO, adicionando a restri9ao de chave estrangeira.

Na hip6tese dessa estrategia, o *script* SQL ficaria do seguinte modo:



Fonte: 0 autor

**'l::J ATENCAO**

Note que, no *script* anterior, as tabelas foram criadas sem chave estrangeira (linhas 1 a 12). Na linha 14, o comandoALTER TABLE modifica a estrutura da tabela CURSO, implementando a restri9ao de chave estrangeira que representa o relacionamento entre CURSO e NIVEL.

###### CUIDADOS AO MANIPULAR TABELAS RELACIONADAS

Aprendemos que um banco de dados relacional e composto por diversas tabelas e que cada tabela em geral possui varias colunas. Vimos tambem que o relacionamento entre tabelas e implementado com o uso do mecanismo de chave estrangeira.

Quando definimos que alguma coluna e uma chave estrangeira, na pratica, estamos criando uma dependencia entre as tabelas envolvidas, pois toda chave estrangeira aponta para o valor de alguma chave primaria.

Ao mesmo tempo, todo SGBD precisa manter a integridade dos dados durante todo o ciclo de vida do banco de dados. Com isso, nao basta somente conhecermos a estrutura dos comandos para altera9ao de tabelas.

**Ii RESUMINDO**

Queremos dizer que, em algumas situa96es, mesmo que o comando para altera9ao ou exclusao esteja correto sob o ponto de vista sintatico, o SGBD sempre prioriza a integridade dos dados e pode inibir sua execu9ao caso o resultado tenha potencial para gerar inconsistencia nos dados.

Vamos estudar um exemplo?

Suponha que temos interesse em remover a tabela NIVEL. Para isso, executaremos o seguinte comando SQL:

-- Comando para remover a tabela NIVEL DROP TABLE NIVEL;

0 SGBD nao removera a tabela NIVEL e retornara uma mensagem de erro, informando que ha um objeto (tabela CURSO) que depende da tabela NIVEL.

Perceba que se o SGBD removesse a tabela NIVEL, a tabela CURSO ficaria inconsistente, vista que sua chave estrangeira (CODIGONIVEL) faz referencia a chave primaria da tabela NIVEL.

Assim, antes de remover uma tabela do banco de dados, e necessario avaliar todos os relacionamentos desta.

E se ainda assim quisessemos remover a tabela NIVEL?

**tit RESPOSTA**

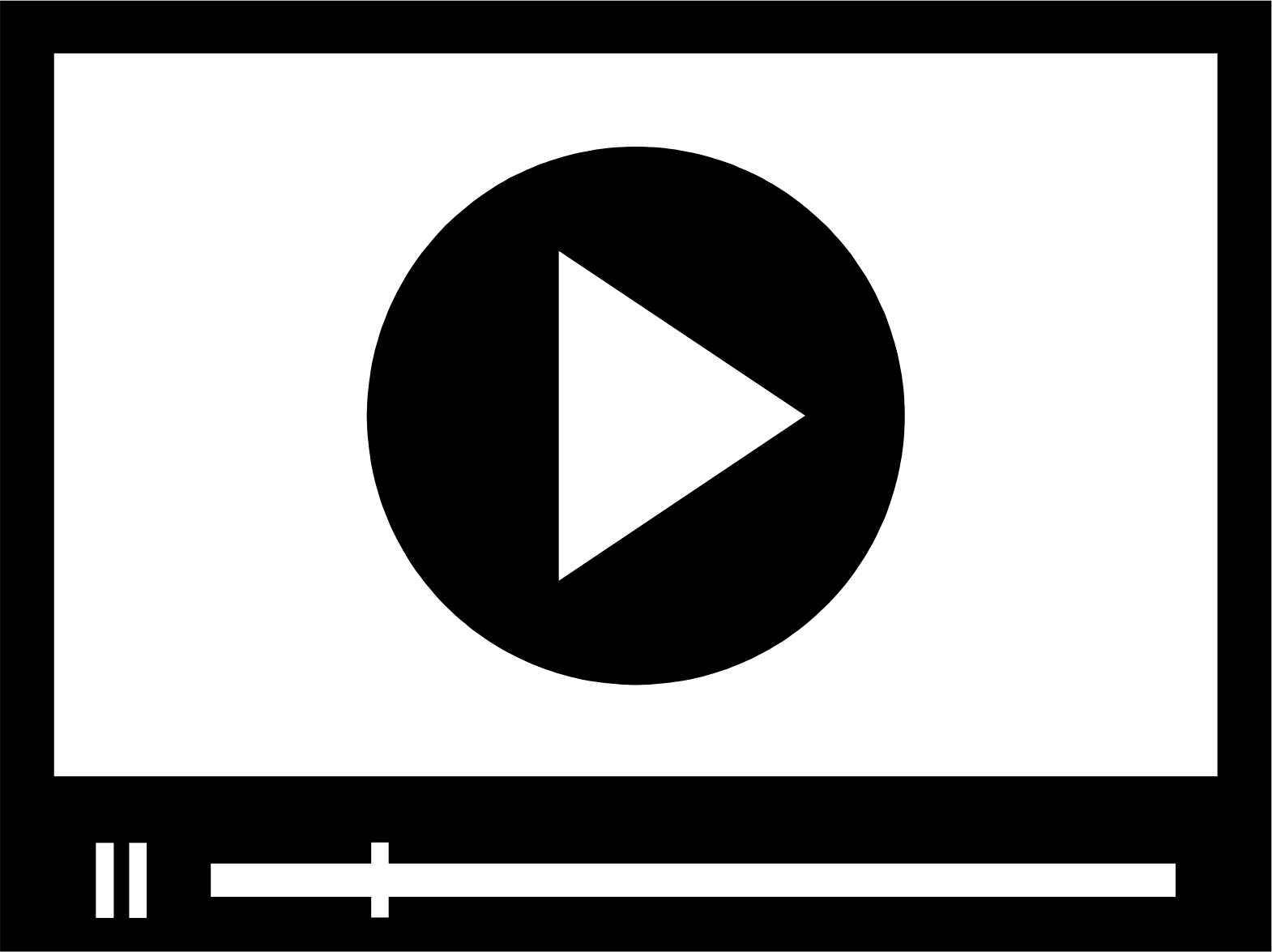
Antes, precisariamos remover as dependencias, para em seguida exclui-la do banco de dados. No entanto, como, no banco de dados, pode haver varias dependencias envolvendo a tabela NIVEL- o que tornaria o processo mais demorado - o SGBD dispoe de um recurso que realiza essa tarefa automaticamente. Trata-se de remoc;ao em cascata.

Nesse caso, poderiamos usar o comando SOL a seguir:

-- Comando para remover a tabela NIVEL - remoc;ao em cascata DROP TABLE NIVEL CASCADE;

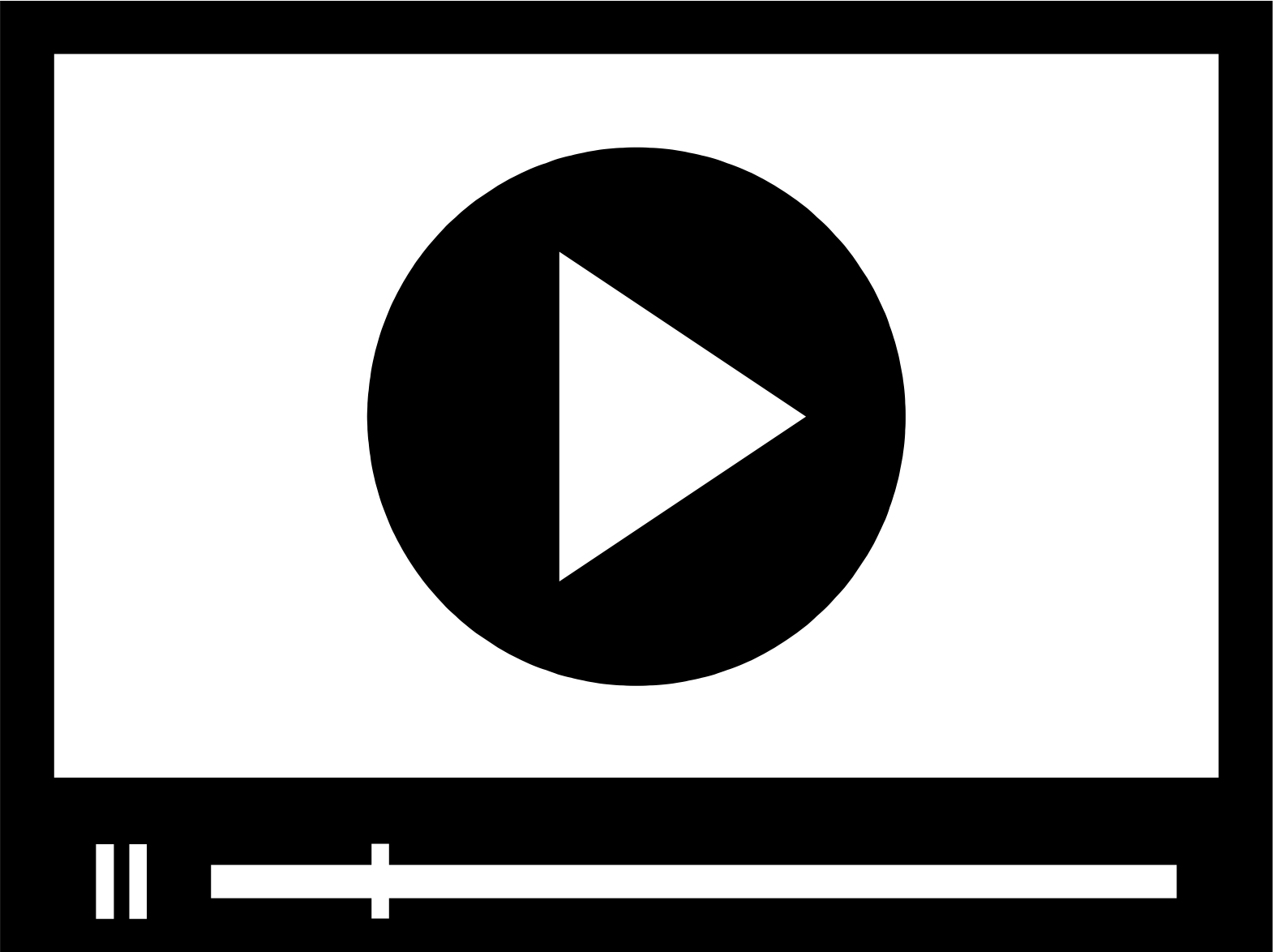
lnternamente, o comando altera a estrutura da tabela CURSO, removendo a restric;ao de chave estrangeira da coluna CODIGONIVEL. Em seguida, a tabela NIVEL e removida do banco de dados.

Ao longo deste modulo, aprendemos comandos basicos da linguagem SOL, uteis para a criac;ao de tabelas no SGBD PostgreSOL. Ainda, conhecemos os tipos de dados mais comuns do PostgreSOL. Em seguida, estudamos comandos para a alterac;ao ea remoc;ao de tabelas do banco de dados.



#### CRIACAO E MANIPULACAO DE OBJETOS UTILIZANDO O **PGADMIN**



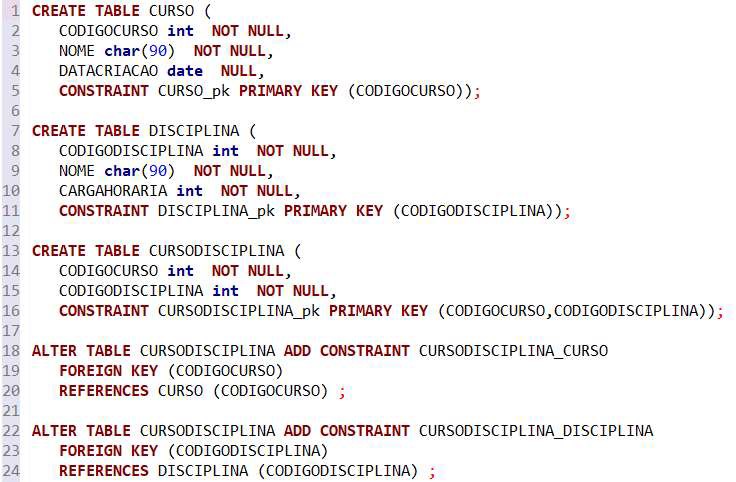


CRIACAO E MANIPULACAO DE OBJETOS UTILIZANDO O PLSQL



##### VERIFICANDO O APRENDIZADO

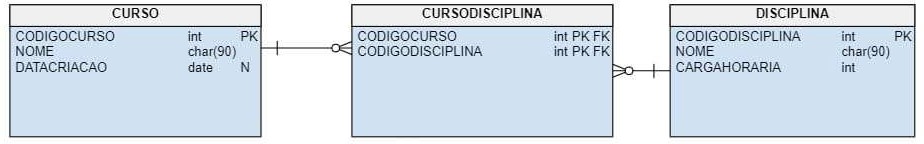
1. ANALISE O *SCRIPT* A SEGUIR E ASSINALE A PROPOSl<;AO VERDADEIRA:



**FONTE: 0 AUTOR**

* 1. A execu9ao completa do *script* cria tres tabelas e todas as colunas sao obrigat6rias.
  2. A execu9ao dos comandos entre as linhas 1 e 16 cria tres tabelas relacionadas.
  3. A execuc;ao dos comandos entre as linhas 18 e 24 cria tres relacionamentos.
  4. A execuc;ao dos comandos entre as linhas 18 e 24 cria dois relacionamentos: o primeiro envolve as tabelas CURSODISCIPLINA e CURSO. 0 segundo, as tabelas CURSODISCIPLINA e DISCIPLINA.

1. ANALISE O MODELO A SEGUIR E ASSINALE A PROPOSICAO VERDADEIRA:



**FONTE: 0 AUTOR**

* 1. A execuc;ao do *script* a seguir cria a tabela CURSODISCIPLINA e seus relacionamentos.

CREATE TABLE CURSODISCIPLINA ( CODIGOCURSO int NOT NULL, CODIGODISCIPLINA int NOT NULL,

CONSTRAINT CURSODISCIPLINA\_pk (PRIMARY KEY (CODIGOCURSO,CODIGODISCIPLINA));

* 1. A execuc;ao do *script* a seguir cria a tabela DISCIPLINA.

CREATE TABLE CURSO ( CODIGOCURSO int NOT NULL, NOME char(90)NOT NULL, DATACRIACAO date NULL,

CONSTRAINT CURSO\_pk (PRIMARY KEY (CODIGOCURSO));

* 1. Admitindo a existencia das tabelas CURSODISCIPLINA e DISCIPLINA, a execuc;ao do *script* a seguir relaciona CURSODISCIPLINA a DISCIPLINA.

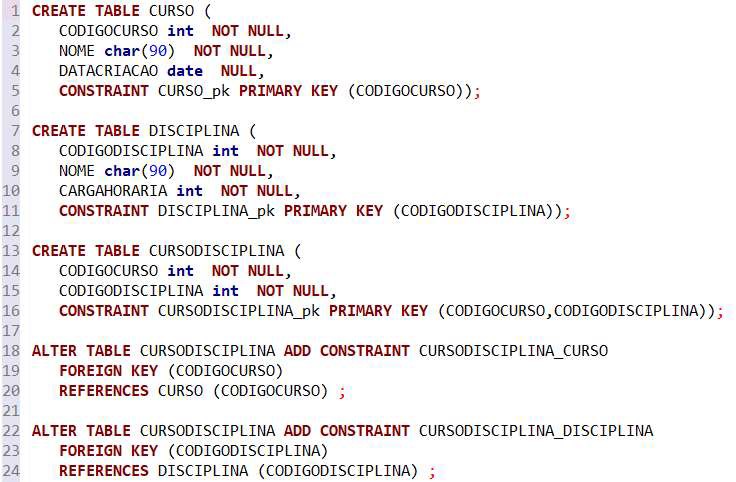
ALTER TABLE CURSODISCIPLINAADD FOREIGN KEY (CODIGOCURSO) REFERENCES CURSO (CODIGOCURSO);

* 1. Admitindo a existencia das tabelas CURSODISCIPLINA e CURSO, a execu9ao do *script* a seguir relaciona CURSODISCIPLINA a CURSO.

ALTER TABLE CURSODISCIPLINAADD FOREIGN KEY (CODIGOCURSO) REFERENCES CURSO (CODIGOCURSO)

GABARITO

1. **Analise o *script* a seguir e assinale a proposic;ao verdadeira:**



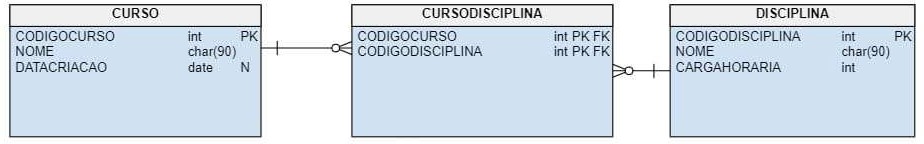
**Fonte: 0 autor**

A alternativa **"D** " esta correta.

De fato, ha dois blocos de comando entre as linhas 18 e 24. 0 primeiro altera a estrutura da tabela CURSODISCIPLINA adicionando a coluna CODIGOCURSO uma restri9ao de chave estrangeira que implementa o relacionamento entre as tabelas CURSODISCIPLINA e CURSO. 0 segundo altera a estrutura da tabela CURSODISCIPLINA, adicionando a coluna

CODIGODISCIPLINA uma restri9ao de chave estrangeira que implementa o relacionamento entre as tabelas CURSODISCIPLINA e DISCIPLINA.

1. **Analise o modelo a seguir e assinale a proposic;ao verdadeira:**



**Fonte: 0 autor**

A alternativa **"D** " esta correta.

De fato, se observarmos o modelo, ha um relacionamento entre as tabelas CURSO e CURSODISCIPLINA. Ao executar o ALTER TABLE, a coluna CODIGOCURSO da tabela CURSODISCIPLINA passa a funcionar coma chave estrangeira, fazendo referencia a tabela CURSO.

### M6DULO 3



0 Empregar comandos para manipular linhas nas tabelas

### MANIPULACAO DE LINHAS NAS TABELAS

Quando usamos o termo manipulac;ao, fazemos referencia as operac;oes de inserc;ao, atualizac;ao ou mesmo eliminac;ao de dados. Em uma linguagem mais comercial, existe o termo **CRUD,** que representa quatro operac;oes basicas: criac;ao, consulta, atualizac;ao e remoc;ao de dados, respectivamente.

No contexto da da SQL, costumamos usar o seguinte mapeamento dos comandos: Create: *INSERT*

Read: *SELECT*

Update: *UPDATE*

Delete: *DELETE*

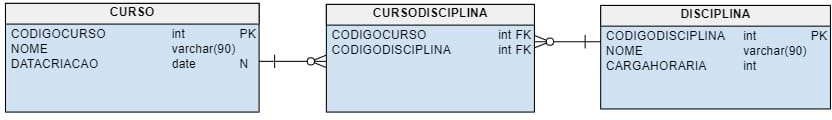
Ainda, os comandos da linguagem SQL que aprenderemos fazem parte da **DML** e sao usados para inserir, modificar e remover dados.

**DML**

*Data Manipulation Language* ou Linguagem de Manipulagao de Dados.

###### MODELO PARA OS EXEMPLOS

Ao longo da nossa aprendizagem, n6s vamos admitir que o modelo a seguir esta criado no banco de dados:

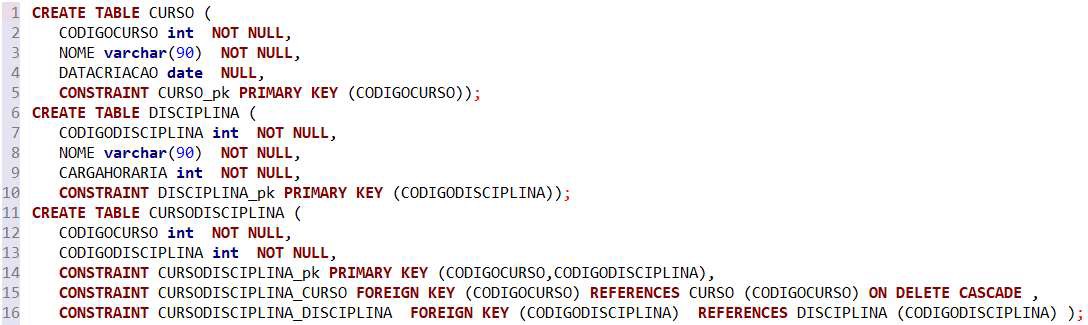


Fonte: 0 autor

***ii*** Tabelas CURSO, CURSODISCIPLINA e DISCIPLINA.

**f DICA**

Recomendamos que voce crie as tabelas e insira algumas linhas, o que pode ser feito usando o script a seguir, a partir da ferramenta de sua preferencia. Para isso, tenha em mente que e necessario estar conectado ao PostgreSQL e ao database bdestudo criado anteriormente.



Fonte: Autor

O modelo e util para gerenciar os dados de cursos, disciplinas e do relacionamento entre esses objetos. Em especial, cada linha da tabela CURSODISCIPLINA representa uma associagao entre

curso e disciplina.

##### INSERCAO DE LINHAS EM TABELA

A insergao de linhas em tabela e realizada com o auxnio do comando INSERT da SQL. Sua sintaxe **basica** esta expressa a seguir:

INSERT INTO <NOMETABELA> (COLUNA 1, COLUNA2, ... ,COLUNAn) VALUES (VALOR1, VALOR2,... ,VALORn);

Na sintaxe, <NOMETABELA> representa a tabela alvo da insergao. Em seguida, sao declaradas as colunas que receberao os dados; por ultimo, os dados em si. Note que deve haver uma correspondencia entre cada par COLUNANALOR, ou seja, o conteudo de cada coluna deve ser compatfvel com o tipo de dados ou domfnio dela.

A sintaxe **completa** a respeito do comando INSERT pode ser encontrada no site oficial do PostgreSQL.

Vamos estudar um exemplo?

lremos cadastrar quatro cursos. 0 comando SQL a seguir pode ser utilizado: INSERT INTO CURSO (CODIGOCURSO,NOME,DATACRIACAO)

VALUES( 1,'Sistemas de lnformagao','19/06/1999');

INSERT INTO CURSO (CODIGOCURSO,NOME,DATACRIACAO) VALUES( 2,'Medicina','10/05/1990');

INSERT INTO CURSO (CODIGOCURSO,NOME,DATACRIACAO) VALUES( 3,'Nutrigao','19/02/2012');

INSERT INTO CURSO (CODIGOCURSO,NOME,DATACRIACAO) VALUES( 4,'Pedagogia','19/06/1999');

ATENCAO

Note que, dentro dos parenteses representativos dos conteudos, o primeiro valor, por ser do tipo inteiro, foi informado sem aspas. Ja o segundo valor, por ser do tipo char, foi informado com aspas. Finalmente, o valor referente ao tipo date tambem foi informado entre aspas. lnternamente, o PostgreSQL converte o texto para o formato de data.

Agora, faremos um procedimento semelhante, cadastrando quatro disciplinas. 0 comando SQL a seguir pode ser utilizado:

INSERT INTO DISCIPLINA (CODIGODISCIPLINA,NOME,CARGAHORARIA)

VALUES( 1,"Leitura e Produgao de Textos',60);

INSERT INTO DISCIPLINA (CODIGODISCIPLINA,NOME,CARGAHORARIA)

VALUES( 2,"Redes de Computadores',60);

INSERT INTO DISCIPLINA (CODIGODISCIPLINA,NOME,CARGAHORARIA)

VALUES( 3,'Computagao Grafica',40);

INSERT INTO DISCIPLINA (CODIGODISCIPLINA,NOME,CARGAHORARIA) VALUES( 4,'Anatomia',60);

Agora, vamos registrar na tabela CURSODISCIPLINA algumas associa<;oes entre cursos e disciplinas. 0 comando SQL a seguir pode ser utilizado:

INSERT INTO CURSODISCIPLINA(CODIGOCURSO,CODIGODISCIPLINA) VALUES (1,1); INSERT INTO CURSODISCIPLINA(CODIGOCURSO,CODIGODISCIPLINA) VALUES (1,2); INSERT INTO CURSODISCIPLINA(CODIGOCURSO,CODIGODISCIPLINA) VALUES (1,3); INSERT INTO CURSODISCIPLINA(CODIGOCURSO,CODIGODISCIPLINA) VALUES (2,1); INSERT INTO CURSODISCIPLINA(CODIGOCURSO,CODIGODISCIPLINA) VALUES (2,3); INSERT INTO CURSODISCIPLINA(CODIGOCURSO,CODIGODISCIPLINA) VALUES (3,1); INSERT INTO CURSODISCIPLINA(CODIGOCURSO,CODIGODISCIPLINA) VALUES (3,3);

E se submetermos ao SGBD o comando a seguir?

INSERT INTO CURSODISCIPLINA(CODIGOCURSO,CODIGODISCIPLINA) VALUES (3,30);

**tit RESPOSTA**

O SGBD nao realizara a inserc;ao e retornara uma mensagem de erro informando que 30 nao e

um valor previamente existente na chave primaria da tabela DISCIPLINA. Issa acontece porque, quando definimos (linha 16 do *script* da sec;ao anterior) a chave estrangeira da tabela CURSODISCIPLINA, n6s delegamos ao SGBD a tarefa de realizar esse tipo de validac;ao com objetivo de sempre manter a integridade dos dados do banco de dados. Note que nao existe disciplina identificada de c6digo 30 na tabela DISCIPLINA.

##### MECANISMO DE CHAVE PRIMARIA EM ACAO

Ja vimosque o SGBD e responsavel par manter a integridade dos dados ao longo de todo o ciclo de vida do banco de dados. A consequencia disso pode ser percebida ao tentarmos executar

(novamente) o comando a seguir:

INSERT INTO DISCIPLINA (CODIGODISCIPLINA, NOME, CARGAHORARIA) VALUES

(4,'Anatomia',60);

Como ja existe um registro com valor de CODIGODISCIPLINA igual a 4, o SGBD exibira mensagem de erro informando que o referido valor ja existe no banco de dados.

Semelhantemente, devemos lembrar que todo valor de chave primaria e obrigat6rio. Vamos agora tentar inserir uma disciplina sem valor para CODIGODISCIPLINA, conforme comando SQL a seguir:

INSERT INTO DISCIPLINA (CODIGODISCIPLINA, NOME, CARGAHORARIA) VALUES

(NULL,'Biologia Celular e Molecular',60);

0 SGBD exibira mensagem de erro informando que o valor da coluna CODIGODISCIPLINA nao pode ser nulo.

##### ATUALIZACAO DE LINHAS EM TABELA

A atualiza<;ao de linhas em tabela e realizada com o auxnio do comando UPDATE da SQL. Sua sintaxe **basica** esta expressa a seguir:

UPDATE <NOMETABELA>

SET COLUNA 1=VALOR1, COLUNA2=VALOR2,... ,COLUNAn=VALORn WHERE <CONDl<;AO>;

Na sintaxe, <NOMETABELA> representa a tabela alvo da atualiza9ao. Em seguida, e declarada uma lista contendo a coluna e o seu respectivo valor nova. Par ultimo, uma condi<;ao I6gica, caso seja necessario. Issa ocorre porque, em geral, estamos interessados em alterar somente um subconjunto de linhas que e obtido a partir do processamento da clausula WHERE. A sintaxe **completa** a respeito do comando UPDATE pode ser encontrada no site oficial do PostgresSQL.

Vamos estudar alguns exemplos?

Alteraremos para 70 a carga horaria da disciplina Redes de Computadores. Para isso, basta executar o comando a seguir:

UPDATE DISCIPLINA SET CARGAHORARIA=70 WHERE CODIGODISCIPLINA=2;

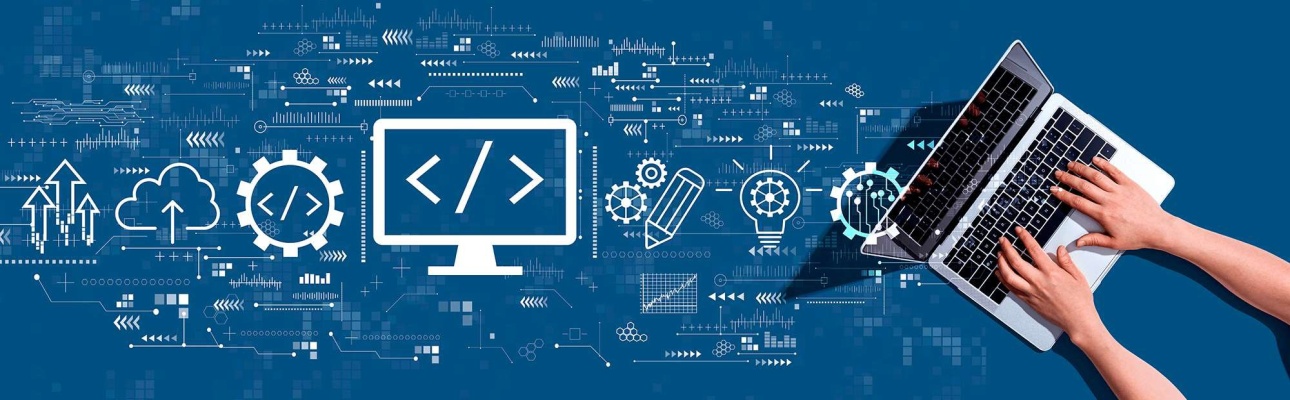
No comando, o SGBD busca na tabela a disciplina cujo valor da coluna CODIGODIISCIPLINA seja igual a 2. Em seguida, atualiza o valor da coluna CARGAHORARIA para 70. Note tambem que poderfamos ter executado o comando a seguir:

UPDATE DISCIPLINA SET CARGAHORARIA=70 WHERE NOME='Redes de Computadores';

Suponha agora que houve a necessidade de alterar em 20% a carga horaria de todas as disciplinas cadastradas no banco de dados. Podemos executar o seguinte comando:

UPDATE DISCIPLINA SET CARGAHORARIA=CARGAHORARIA\*1.2;

Note que, no ultimo comando, nao foi necessaria a clausula WHERE, vista que o nosso interesse era ode atualizar todas as linhas da tabela DISCIPLINA. Ainda, para obter o **novo** valor, n6s utilizamos a expressao CARGAGORARIA\*1.2.



Fonte: lierneyMJ/Shutterstock

##### ATUALIZACAO DE COLUNA CHAVE PRIMARIA

Devemos ter especial cuidado ao planejarmos alterar o valor de coluna com o papel de chave primaria em uma tabela.

Vamos supor que seja necessario alterar para 6 o valor de CODIGOCURSO referente ao curso de Pedagogia. Podemos, entao, executar o comando a seguir:

UPDATE CURSO SET CODIGOCURSO=6 WHERE CODIGOCURSO=4;

Perceba que o SGBD processara a alterac;ao, vista que nao ha vinculo na tabela CURSODISCIPLINA envolvendo este curso. No entanto, o que aconteceria se tentassemos alterar para 10 o valor de CODIGOCURSO referente ao curso de Sistemas de lnformac;ao?

Seguindo a mesma linha de raciocinio da ultima alterac;ao, vamos submeter o comando a seguir: UPDATE CURSO SET CODIGOCURSO=10 WHERE CODIGOCURSO=1;

ATENCAO

O SGBD nao realizara a alterac;ao e retornara uma mensagem de erro indicando que o valor 1 esta registrado na tabela CURSODISCIPLINA, coluna CODIGOCURSO. Issa significa que, se o SGBD aceitasse a alterac;ao, a tabela CURSODISCIPLINA ficaria com dados inconsistentes, o que nao deve ser permitido.

Assim, de modo semelhante ao que aprendemos na sec;ao Mecanismo de chave primaria em ac;ao, deixaremos o SGBD realizar as alterac;6es necessarias para manter a integridade dos dados. Vamos, entao, submeter o comando a seguir:

ALTER TABLE CURSODISCIPLINA

DROP CONSTRAINTCURSODISCIPLINA\_CURSO, ADD CONSTRAINT CURSODISCIPLINA CURSO

FOREIGN KEY (CODIGOCURSO) REFERENCES CURSO (CODIGOCURSO)

ON UPDATE CASCADE ;

**0 que fizemos?** Usamos o comando ALTER TABLE para alterar a estrutura da tabela CURSODISCIPLINA:, removemos a chave estrangeira (comando DROP CONSTRAINT) e, por

ultimo, recriamos a chave (ADD CONSTRAINT), especificando a operac;ao de atualizac;ao (UPDATE) em cascata.

Assim, ap6s o processamento da alterac;ao anterior, podemos entao submeter o comando, conforme a seguir:

UPDATE CURSO SET CODIGOCURSO=10 WHERE CODIGOCURSO=1;

### REMOCAO DE LINHAS EM TABELA

A remoc;ao de linhas em tabela e realizada com o auxilio do comando DELETE da SQL. Sua sintaxe **basica** esta expressa a seguir:

DELETE FROM <NOMETABELA> WHERE <CONDl<;AO>;

Na sintaxe, <NOMETABELA> representa a tabela alvo da operac;ao de delec;ao de linha(s). Par ultimo, ha uma condic;ao I6gica, caso seja necessario. Issa ocorre porque, em geral, estamos interessados em remover somente um subconjunto de linhas que e obtido a partir do

processamento da clausula WHERE. A sintaxe **completa** a respeito do comando DELETE pode ser encontrada no site oficial do PostgresSQL.

Vamos estudar alguns exemplos?

Suponha que temos interesse em apagar do banco de dados a disciplina de Anatomia. Podemos, entao, submeter o c6digo a seguir:

DELETE FROM DISCIPLINA WHERE CODIGODISCIPLINA=4;

0 SGBD localiza na tabela DISCIPLINA a linha cujo conteudo da coluna CODIGODISCIPLINA seja igual a 1. Em seguida, remove do banco de dados a linha em questao.

Agora vamos imaginar que tenha surgido a necessidade de remover do banco de dados a disciplina de Leitura e Produc;ao de Textos. Podemos, entao, submeter o c6digo a seguir:

DELETE FROM DISCIPLINA WHERE CODIGODISCIPLINA=1;

0 SGBD nao realizara a remoc;ao e retornara uma mensagem de erro indicando que o valor 1 esta registrado na tabela CURSODISCIPLINA, coluna CODIGODISCIPLINA. Seo SGBD aceitasse a remoc;ao, a tabela CURSODISCIPLINA ficaria com dados inconsistentes, o que nao deve ser permitido.

Assim, de modo semelhante ao que aprendemos na sec;ao Mecanismo de chave primaria em ac;ao, deixaremos o SGBD realizar as alterac;oes necessarias para manter a integridade dos dados. Vamos, entao, submeter o comando a seguir:

ALTER TABLE CURSODISCIPLINA

DROP CONSTRAINT CURSODISCIPLINA\_DISCIPLINA, ADD CONSTRAINT CURSODISCIPLINA DISCIPLINA

FOREIGN KEY (CODIGODISCIPLINA) REFERENCES DISCIPLINA (CODIGODISCIPLINA) ON DELETE CASCADE;

**0 que fizemos?** Usamos o comando ALTER TABLE para alterar a estrutura da tabela CURSODISCIPLINA:, removemos a chave estrangeira (comando DROP CONSTRAINT) e, por ultimo, recriamos a chave (ADD CONSTRAINT), especificando operac;ao de remoc;ao (DELETE) em cascata.

Assim, ap6s o processamento da alterac;ao anterior, podemos submeter o comando conforme a seguir:

DELETE FROM DISCIPLINA WHERE CODIGODISCIPLINA=1;

Ao processar o comando, o SGBD verifica se existe alguma linha da tabela CURSODISCIPLINA com valor 1 para a coluna CODIGODISCIPLINA. Caso encontre, cada ocorrencia e entao removida do banco de dados.

E se quisessemos eliminar todos os registros de todas as tabelas do banco de dados?

**'RESPOSTA**

Para realizarmos esta operac;ao, precisaremos identificar quais tabelas sao mais independentes e quais sao as que possuem vfnculos de chave estrangeira.

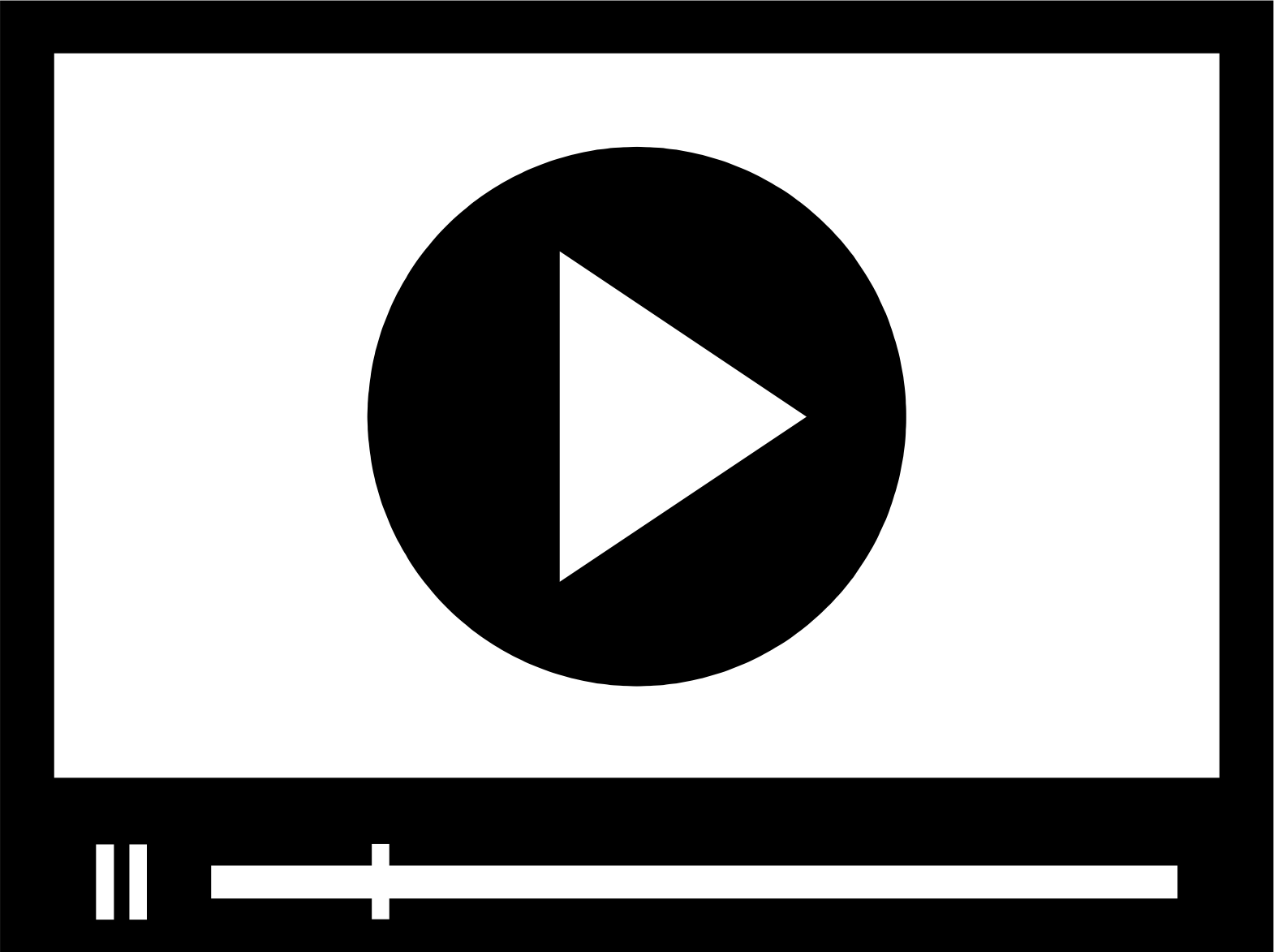
No caso do nosso exemplo, CURSODISCIPLINA possui duas chaves estrangeiras, portanto, ea tabela mais dependente. As demais, nao possuem chave estrangeira. De posse dessa informac;ao, podemos submeter os comandos a seguir para completar a tarefa:

DELETE FROM CURSODISCIPLINA; DELETE FROM CURSO;

DELETE FROM DISCIPLINA;

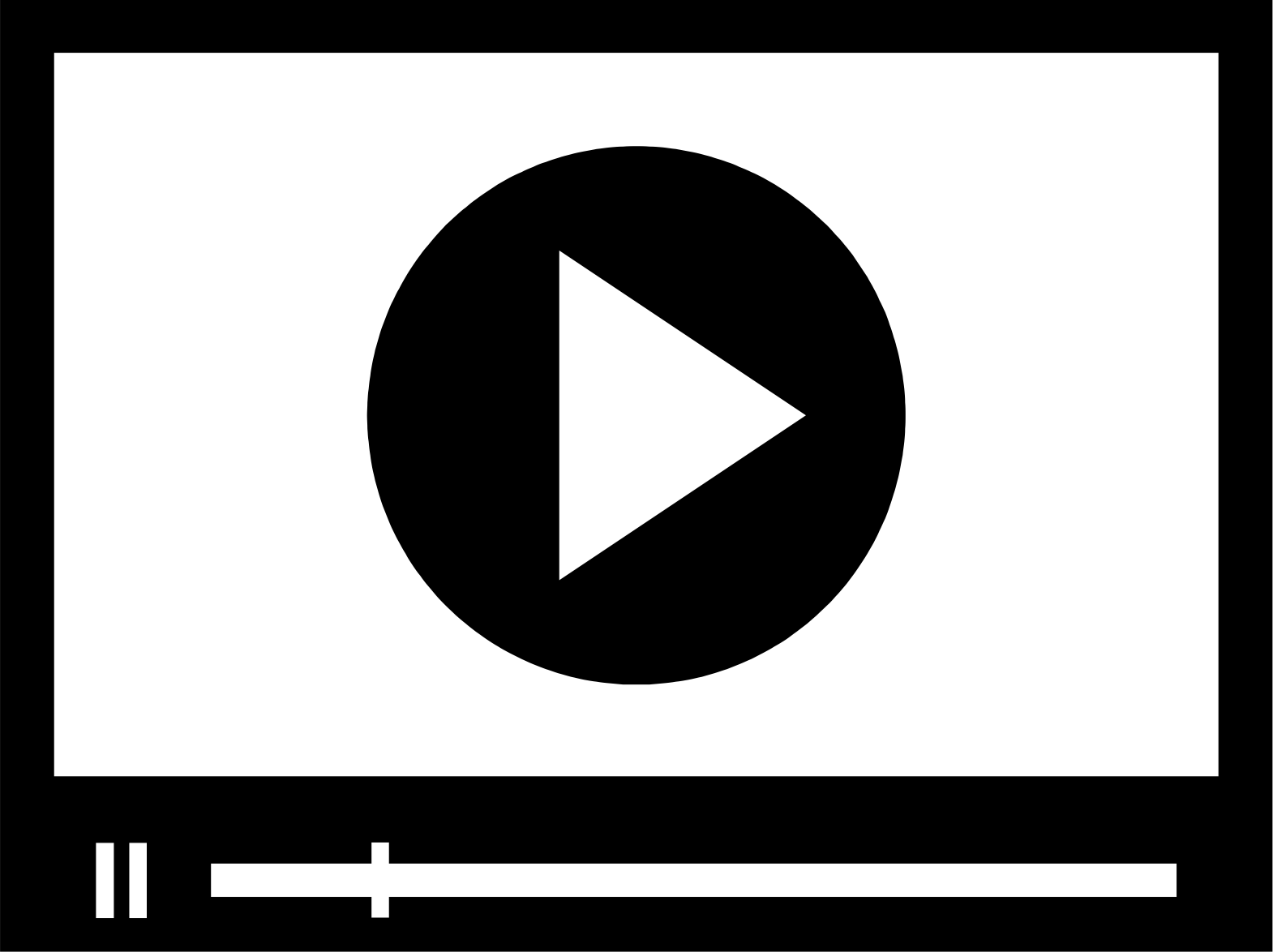
Perceba que a primeira tabela que foi usada no processo de remogao de linhas foi a CURSODISCIPLINA, pois essa ea responsavel por manter as informagoes sabre o relacionamento entre as tabelas CURSO e DISCIPLINA. Ap6s eliminar os registros de CURSODISCIPLINA, o SGBD removera com sucesso os registros das tabelas CURSO e DISCIPLINA.

Ao longo deste modulo, aprendemos os comandos basicos da linguagem SQL, os quais sao uteis para a manipulagao de linhas no SGBD PostgreSQL. Tambem estudamos comandos para inserir, alterar e eliminar linhas em tabelas.



**MANIPULACAO DE DADOS UTILIZANDO PGADMIN**



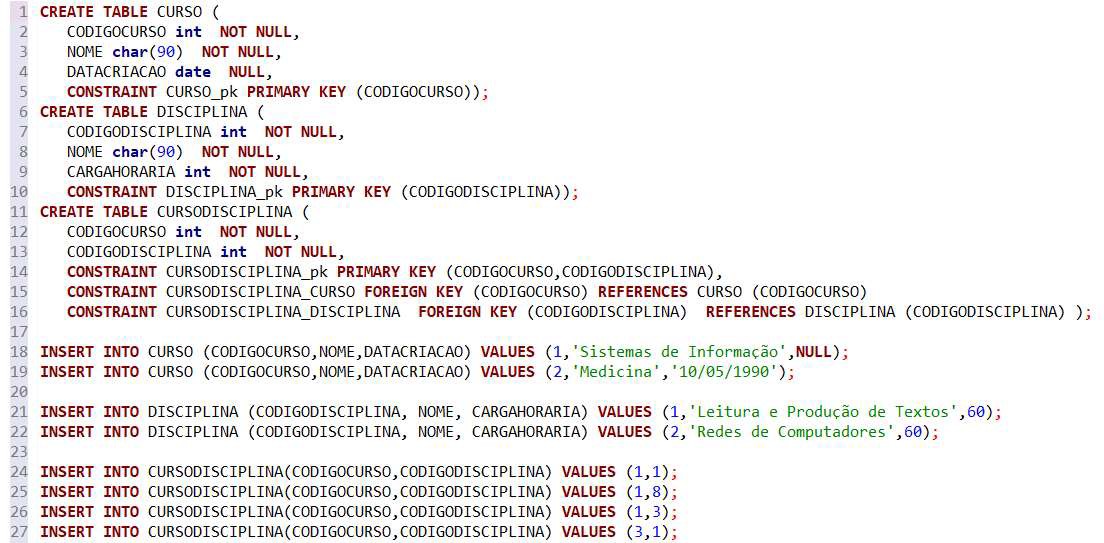


## MANIPULACAO DE DADOS UTILIZANDO 0 PLSQL



##### VERIFICANDO O APRENDIZADO

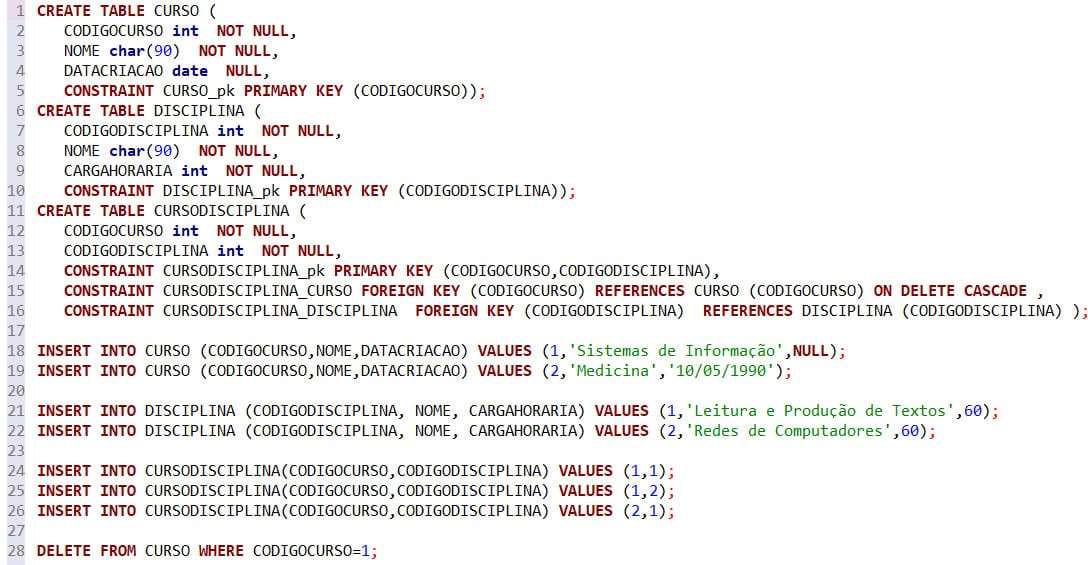
1. ANALISE O *SCRIPT* A SEGUIR E ASSINALE A PROPOSICAO CORRETA:



**FONTE: 0 AUTOR**

* 1. Ap6s a execuc;ao com sucesso do trecho entre as linhas 1 e 16, se executarmos o comando expresso na linha 18, o SGBD retornara uma mensagem de erro.
  2. Ap6s a execuc;ao com sucesso do trecho entre as linhas 1 e 16, se executarmos o comando expresso na linha 25, o SGBD retornara uma mensagem de erro.
  3. Ap6s a execuc;ao com sucesso do trecho entre as linhas 1 e 16, se executarmos o comando expresso na linha 26, o SGBD nao retornara uma mensagem de erro.
  4. Ap6s a execuc;ao com sucesso do trecho entre as linhas 1 e 16, se executarmos o comando expresso na linha 25, o SGBD nao retornara uma mensagem de erro.

1. ANALISE O *SCRIPT* A SEGUIR E ASSINALE A PROPOSl<;AO CORRETA:



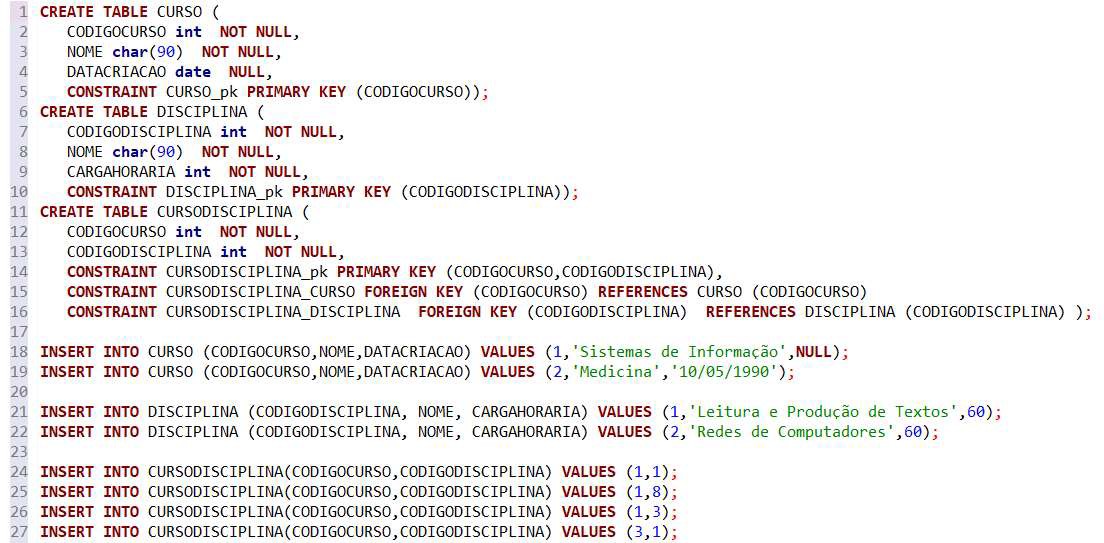
FONTE: 0 AUTOR

APOS A EXECU<;AO, COM SUCESSO, DO TRECHO ENT RE AS LINHAS 1 E 26, SE EXECUTARMOS O COMANDO EXPRESSO NALINHA28, QUANTAS LINHAS SERAO REMOVIDAS DO SGBD?

1. Nenhuma.
2. Uma.
3. Duas.
4. Tres.

GABARITO

* 1. Analise o *script* a seguir e assinale a proposi ao correta:



**Fonte: 0 autor**

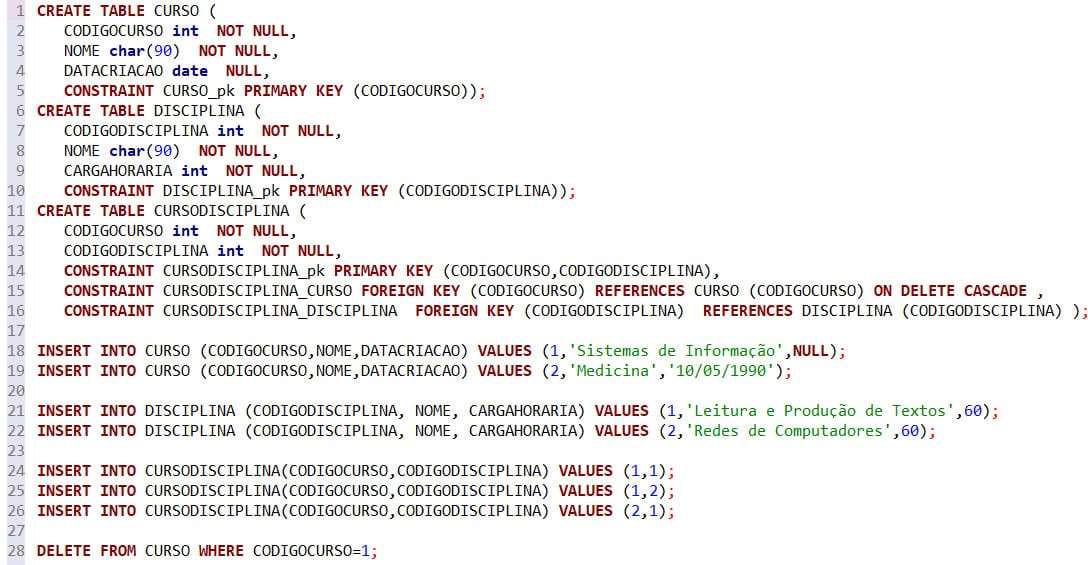
A alternativa **"B** " esta correta.

De fato, se observarmos as insergoes em CURSO e DISCIPLINA, vamos perceber que os

valores de chave primaria sao os inteiros 1 e 2 em ambos os casos. Ainda, a tabela CURSODISCIPLINA possui duas chaves estrangeiras: uma referencia a tabela CURSO; a outra, referencia a tabela DISCIPLINA. Portanto, o valor para CODIGODISCIPLINA da tabela

CURSODISCIPLINA nao pode ser diferente de 1 ou 2.

* 1. **Analise o *script* a seguir e assinale a proposi ao correta:**



**Fonte: 0 autor**

**Ap6s a execuc;ao, com sucesso, do trecho entre as linhas 1 e 26, se executarmos o comando expresso na linha 28, quantas linhas serao removidas do SGBD?**

A alternativa **"D** " esta correta.

De fato, a chave estrangeira declarada na linha 15 foi criada de maneira a permitir a delec;ao em cascata. Ao executar o comando da linha 28, o SGBD eliminara tanto os dais registros da tabela CURSODISCIPLINA (cujo valor de CODIGOCURSO e igual a 1) quanta o registro referente ao curso Sistemas de lnformac;ao.

### M0DUL04

0 Empregar comandos de controle de transac;ao

### TRANSAC0ES EM BANCO DE DADOS

Ao longo do nosso estudo, aprendemos uma serie de comandos SQL envolvendo desde a criac;ao de tabelas ate operac;oes de manipulac;ao de dados, tais coma inserc;oes, atualizac;oes e exclusoes. Basicamente, um comando era codificado e submetido ao SGBD PostgreSQL, que em seguida o executava e devolvia algum resultado.

Perceba que, sob o contexto dos exemplos que estudamos, poderfamos concluir que, de certa maneira, havia somente um usuario acessando a totalidade dos recursos do SGBD.

No entanto, em um ambiente de produc;ao, o SGBD gerencia centenas de requisic;oes das aplicac;oes. Com isso, conclufmos que ha acesso simultaneo a varios recursos que sao gerenciados pelo SGBD.

Ao mesmo tempo, o usuario de uma aplicac;ao que faz uso de recursos de um banco de dados, normalmente, esta preocupado com o resultado dos processos de neg6cio que estao automatizados.



Fonte: PWAWAT WONGOPASS/Shutterstock

Para exemplificar, digamos que um sistema academico disponibiliza a inscric;ao em diversas disciplinas; determinado aluno, ap6s consultar a oferta e escolher um conjunto de materias para inscric;ao, tern expectativa de conseguir inscrever-se em todas as disciplinas alvo da escolha.

No entanto, sob o ponto de vista do SGBD, prover a inscric;ao em todas as disciplinas escolhidas pelo estudante requer a execuc;ao, na totalidade, de diversas instruc;oes de inserc;ao de dados em alguma tabela. Alem disso, deve existir cuidado especial para, por exemplo, inibir inscric;ao em disciplina caso nao haja mais disponibilidade de vagas.

**t:l ATENCAO**

E uma situac;ao tfpica sobre a qual o SGBD precisa prover uma forma de realizar diversas operac;oes como uma unidade 16gica de processamento. Vamos aprender, entao, que essa

unidade de processamento e denominada transac;ao.

Em sistemas de banco de dados, uma transac;ao corresponde a um programa em execuc;ao que forma uma unidade de trabalho.

Os limites de uma transac;ao sao especificados por meio dos comandos *begin transaction* (que indica o infcio de uma transac;ao) e *end transaction* (que indica o termino de uma transac;ao) em um programa de aplicac;ao.

Ainda, sea transagao nao atualiza o banco de dados, e denominada somente de leitura; caso contrario, e chamada de leitura-gravagao.

Para fins didaticos, um modelo simplificado do banco de dados - colegao de itens nomeados - e usado para estudo dos conceitos de processamento de transa96es. Cada item de dados pode representar um registro, bloco ou valor de coluna.

Em se tratando de SGBDs multiusuarios, varias transa96es podem ser executadas simultaneamente. No entanto, caso essa execugao ocorra de maneira descontrolada, poderao surgir problemas de inconsistencias, tais como:

ATUALIZACAO PERDIDA ATUALIZACAO TEMPORARIA RESUMO INCORRETO LEITURA NAO REPETITIVA

**ATUALIZACAO PERDIDA**

Quando duas transa96es que acessam os mesmos itens de dados tern opera96es intercaladas de modo a tornar incorretos alguns itens do banco de dados.

ATUALIZACAO TEMPORARIA

Quando uma transagao atualiza um item do banco de dados e, depois, falha por algum motivo, enquanto, nesse meio tempo, o item e lido por outra transagao antes de ser alterado de volta para seu valor original.

RESUMO INCORRETO

Quando uma transagao calcula uma fungao de resumo de agregagao em uma serie de itens enquanto outras transa96es atualizam alguns desses itens.

LEITURA NAO REPETITIVA

Quando uma transac;ao le o mesmo item duas vezes e o item e alterado por outra transac;ao entre as duas leituras.

E importante sabermos que, quando o SGBD processa uma transac;ao, todas as operac;oes que a formam devem ser completadas com sucesso para, somente depois, haver a gravac;ao

permanente das alterac;oes no banco de dados. Alem disso, caso haja faIha em uma ou mais operac;oes de uma transac;ao, as demais operac;oes nao devem ser executadas, e a transac;ao deve ser cancelada.

Se uma transac;ao for cancelada, deve ser executado um processo denominado ***rollback***

(Rolar

, o qual forc;a o SGBD a trazer de volta os valores antigos dos registros antes da transac;ao ter iniciado. Finalmente, caso a transac;ao seja executada com sucesso, as

para tras)

atualizac;oes devem ser confirmadas por meio do comando ***commit*** .

(Confirmac;ao)

Estas sao as falhas que podem ocorrer durante o processamento de uma transac;ao:

|  |
| --- |
| Falha do computador |
| Erro de transac;ao ou sistema |
| Condic;oes de excec;ao detectadas pela transac;ao |
| Falha de disco, problemas ffsicos e catastrofes |

**Aten ao!** Para visualizac;ao completa da tabela utilize a rolagem horizontal A seguir, conheceremos as propriedades das transac;oes.

###### PROPRIEDADES DE UMA TRANSACAO

Com objetivo de garantir a integridade dos dados contidos no banco de dados, o SGBD mantem um conjunto de propriedades das transac;oes, denominado **ACID,** que representam as caracterfsticas de atomicidade, consistencia, isolamento e durabilidade, respectivamente.

**ACID**

Do Ingles, *Atomicity, Consistency, Isolation, Durability.*

|  |  |
| --- | --- |
| Atomicidade | A transac;ao precisa ser realizada completamente ou nao realizada |
| Consistencia | A transac;ao deve levar o banco de dados de um estado consistente para outro |
| lsolamento | A execuc;ao de uma transac;ao nao deve ser interferida por quaisquer outras transac;oes |
| Durabilidade | As mudanc;as no banco de dados em func;ao da confirmac;ao da transac;ao devem persistir |

**Aten ao!** Para visualizac;ao completa da tabela utilize a rolagem horizontal

###### ESTADOS DE UMA TRANSACAO

Uma transac;ao pode passar pelos seguintes estados ao longo do seu processamento:

ATIVO

Ocorre imediatamente ap6s o infcio da transac;ao, podendo executar operac;oes de leitura e gravac;ao.

PARCIALMENTE CONFIRMADO

Quando a transac;ao termina.

CONFIRMADO

Ap6s verificac;ao bem-sucedida, as mudanc;as sao gravadas permanentemente no banco de dados.

FALHA

Se uma das verificac;oes falhar ou se a transac;ao for abortada durante seu estado ativo.

CONFIRM ADO

Transac;ao sai do sistema.

Para poder recuperar-se de falhas que afetam transac;oes, normalmente, o SGBD mantem um *log*

para registrar todas as operac;oes de transac;ao que afetam os varios itens do banco de dados. As entradas em um registro de *log* de uma determinada transac;ao possuem informac;oes de valores antigos e novas do item de dados do banco de dados, bem coma se a transac;ao foi conclufda com sucesso ou se foi abortada.

### TRANSAC0ES NO POSTGRESQL

De maneira geral, uma transac;ao no PostgreSQL possui a estrutura a seguir: BEGIN-- infcio da transac;ao

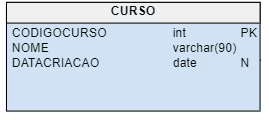
-- comandos

COMMIT-- transac;ao confirmada ROLLBACK--transac;ao cancelada END-- mesma func;ao do COMMIT

No entanto, nos SGBDs, a inicializac;ao de uma transac;ao ocorre implicitamente quando executamos alguns comandos.

Vamos estudar um exemplo?

Seja a tabela CURSO, conforme a seguir:



Fonte: Autor

**m** Tabela CURSO.

A execuc;ao de um INSERT na tabela ocorre dentro do contexto implfcito de uma transac;ao:

-- BEGIN implfcito

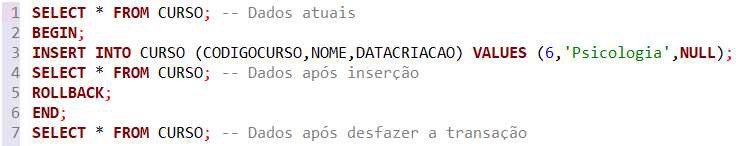
INSERT INTO CURSO (CODIGOCURSO,NOME,DATACRIACAO) VALUES (5,'Engenharia de

Computagao' ,NULL);

-- COMMIT implfcito

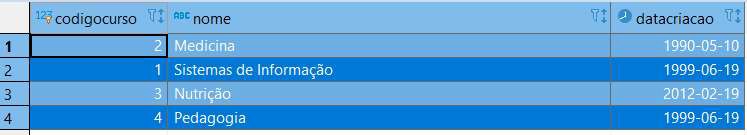
Estudamos que, quando uma transagao e desfeita, qualquer operagao que faz parte da transagao deve ser cancelada. Vamos, entao, ver como podemos fazer isso no PostgreSQL. Veja o exemplo a seguir, construfdo com objetivo de inserir um registro na tabela CURSO e, em seguida,

indicar que a operagao de insergao ser desfeita pelo SGBD:



Fonte: 0 autor

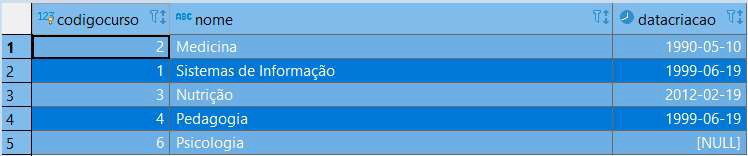
Ap6s o processamento do comando da linha 1, visualizaremos o conteudo da tabela CURSO da seguinte maneira:



Fonte: 0 autor

m Tabela CURSO ap6s a execugao do comando da linha 1.

Ap6s o processamento dos comandos da linha 2 a linha 4, que ja representam a insergao de um registro na tabela CURSO sob o contexto de uma transagao explfcita, teremos este resultado:

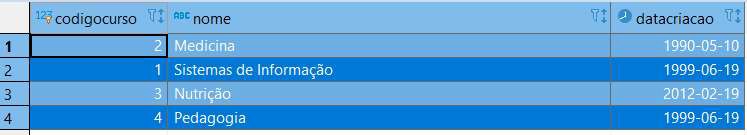


Fonte: 0 autor

m Tabela CURSO ap6s a execugao do comando da linha 4.

Finalmente, ap6s o processamento dos comandos da linha 5 a linha 7, onde a transagao e

desfeita, teremos o resultado conforme a tabela a seguir:



Fonte: 0 autor

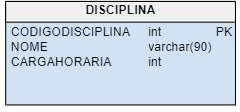
**m** Tabela CURSO ap6s a execugao do comando da linha 7.

Perceba que o comando da linha 2 (BEGIN) iniciou explicitamente uma transagao, a qual foi abortada quando da execugao do comando da linha 5 (ROLLBACK).

**'l::l ATENCAO**

No exemplo anterior, programamos uma transagao que envolveu somente uma operagao, a saber, insergao de uma linha na tabela CURSO. No entanto, uma transagao pode envolver diversas linhas de uma tabela do banco de dados.

Vamos, entao, programar uma transagao que consistira em duas opera96es de atualizagao envolvendo a tabela que contem as disciplinas, apresentada conforme a seguir:



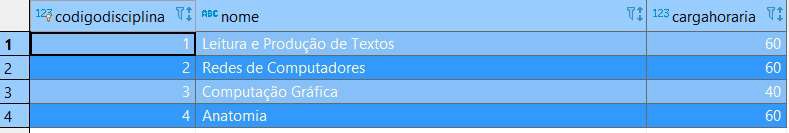
Fonte: 0 autor

**&I** Tabela DISCIPLINA.

lnicialmente, vamos listar o conteudo da tabela DISCIPLINA. Podemos, entao, usar o comando a seguir:

SELECT\* FROM DISCIPLINA;

Ap6s o processamento do comando, teremos o seguinte resultado:



Fonte: 0 autor

**m** Conteudo da tabela DISCIPLINA com os dados originais.

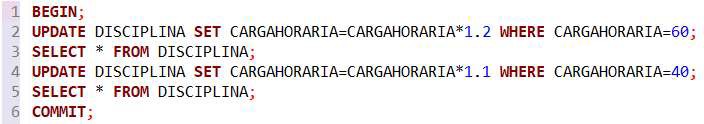
Agora, nossa intengao e alterar a carga horaria das disciplinas de acordo com os criterios a seguir:

Disciplinas que possuem 60 horas: aumento em 20%

Disciplinas que possuam 40 horas: aumento em 10%

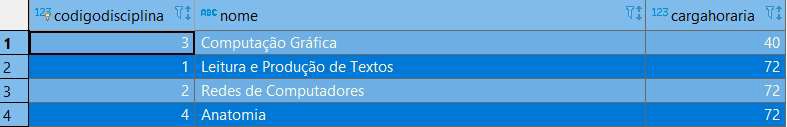
**Aten ao!** Para visualizagao completa da tabela utilize a rolagem horizontal

0 c6digo a seguir expressa uma transagao envolvendo operagoes de atualizagao de dados na tabela DISCIPLINA:



Fonte: 0 autor

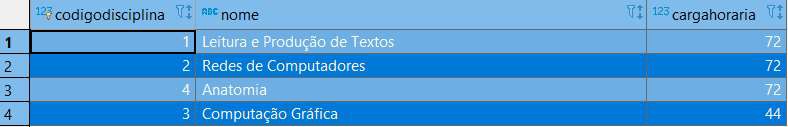
Ap6s a execugao do trecho entre as linhas 1 e 3, o conteudo da tabela disciplina estara conforme a seguir:



Fonte: 0 autor

**m** Conteudo da tabela DISCIPLINA ap6s a execugao da linha 3.

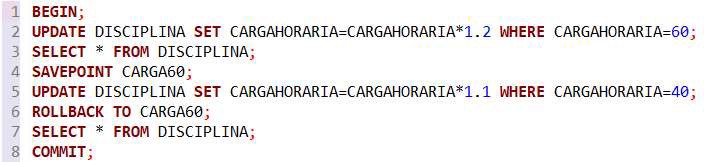
Ap6s a execugao da transagao, o conteudo da tabela disciplina estara conforme a seguir:



Fonte: 0 autor

Conteudo da tabela DISCIPLINA ap6s o termino da transa9ao.

Outro ponto interessante no projeto de transac;oes ea utilizac;ao de pontos de salvamento (SAVEPOINT). Observe o exemplo a seguir:



Fonte: 0 autor

Na linha 4, adicionamos um SAVEPOINT denominado CARGA60. Quando a linha 6 for executada, o SGBD vai desfazer a operac;ao de UPDATE da linha 5.

##### UM POUCO MAIS SOBRE ATUALIZACAO TEMPORARIA

Estudamos que uma transac;ao nao deve atrapalhar o andamento de outra. Pense na execuc;ao da nossa ultima transac;ao, que envolveu dois comandos de atualiza9ao na tabela DISCIPLINA.

Vimos que, logo ap6s a execuc;ao da linha 3, a unica disciplina que nao sofreu alterac;ao foi a de Computa9ao Grafica. Perceba que o SELECT da linha 3 esta sendo executado no contexto da transac;ao.

No entanto, o que aconteceria se tivessemos em paralelo outra aplica9ao que submetesse consulta para acessar os registros da tabela DISCIPLINA no mesmo momenta da execuc;ao da linha 3 da transac;ao?

**tit RESPOSTA**

A consulta em questao enxergaria os dados "originais", sem quaisquer alterac;oes. Par qual razao? Para nao haver o problema da atualizac;ao temporaria. Queremos dizer que, se a transac;ao fosse desfeita par qualquer motivo, o UPDATE da linha 2 seria tambem desfeito.

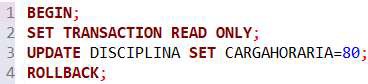
##### UM POUCO MAIS SOBRE TRANSACAO DE LEITURA

Vimos que uma transac;ao que nao modifica dados e denominada transac;ao somente de leitura (READ ONLY). Caso contrario, e denominada leitura-gravac;ao (READ WRITE).

Para especificar o tipo de transac;ao, usaremos o comando SET TRANSACTION

<TIPOTRANSA<;AO>. No PostgreSQL, quando iniciamos uma transac;ao, o padrao e READ WRITE.

Vamos analisar um exemplo envolvendo uma transac;ao READ ONLY

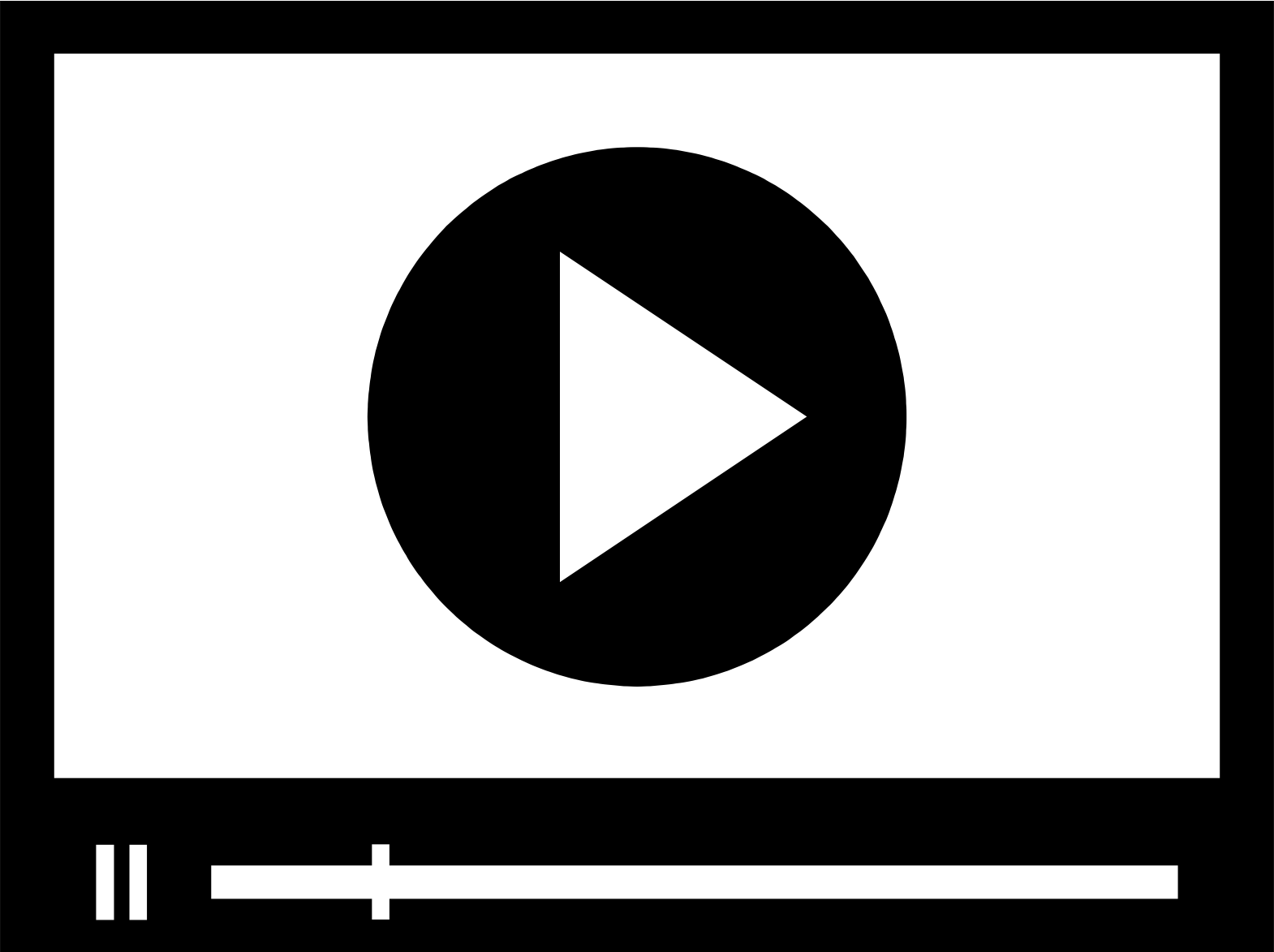


Fonte: Autor

Na transac;ao acima, propositalmente, inserimos um comando de atualizac;ao em uma transac;ao que nao permite essa categoria de comando. Logo, ap6s a execuc;ao da linha 3, o SGBD retornara uma mensagem informando ao usuario que nao e possfvel executar comando de atualizac;ao em uma transac;ao do tipo somente leitura.

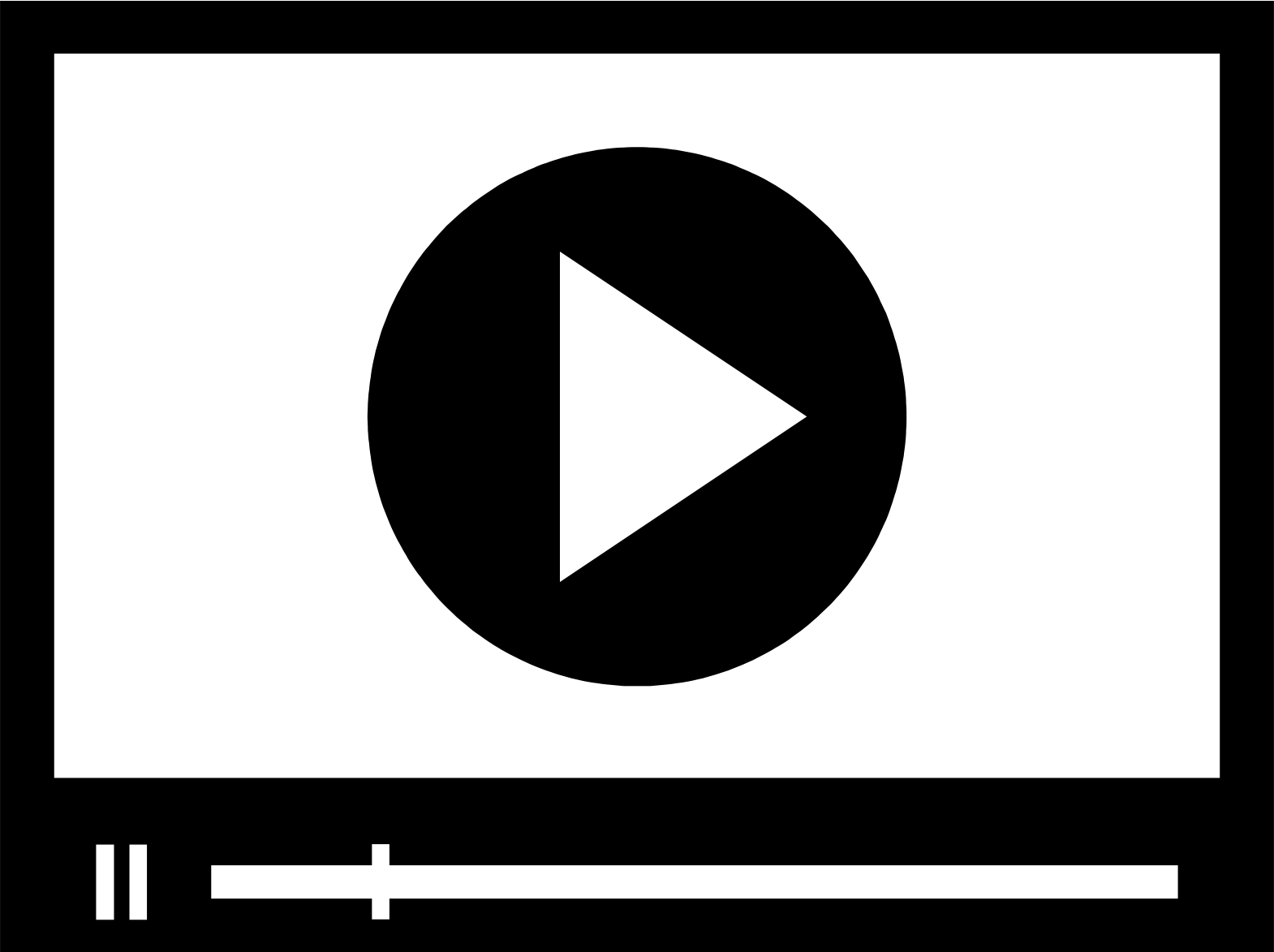
Ao longo deste modulo, aprendemos o conceito de transac;ao, que representa uma sequencia de comandos que devem ser executados na totalidade, ou, caso contrario, desfeitos.

Vimosque, par padrao, o SGBD PostgreSQL implicitamente gera uma transac;ao quando submetemos algum comando de inserc;ao, atualizac;ao ou remoc;ao de dados. Par fim, aprendemos comandos para gerenciar transac;oes no PostgreSQL.



## TRANSAC0ES DO POSTGRESQL UTILIZANDO **PGADMIN**





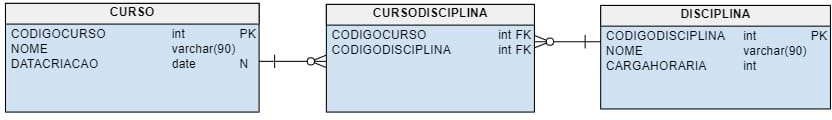
#### TRANSAC0ES NO POSTGRESQL UTILIZANDO O PLSQL



VERIFICANDO O APRENDIZADO

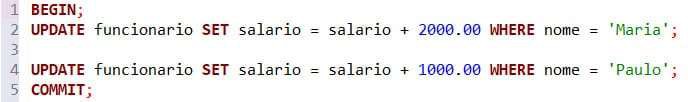
1. **ARESPEITO DE TRANSACOES NO POSTGRESQL E, CONSIDERANDO**

**0 MODELO A SEGUIR, ASSINALE A PROPOSICAO VERDADEIRA:**



**FONTE: 0 AUTOR**

1. Ao executar o comando DELETE FROM CURSODISCIPLINA(; o PostgreSQL nao executa uma transac;ao.
2. 0 comando DELETE FROM CURSODISCIPLINA(; pode ser executado sem erro em uma transac;ao PostgreSQL do tipo READ ONLY.
3. 0 comando DELETE FROM CURSODISCIPLINA(; pode ser executado sem erro em uma transac;ao PostgreSQL do tipo READ WRITE.
4. 0 comando SELECT\* FROM CURSODISCIPLINA(; nao pode ser executado em uma transac;ao PostgreSQL do tipo READ ONLY.
5. **SUPONHAQUE UM PROFISSIONAL PROGRAMOU NO POSTGRESQL OS SEGUINTES COMANDOS EM UMA TABELADENOMINADAEMPREGADO:**



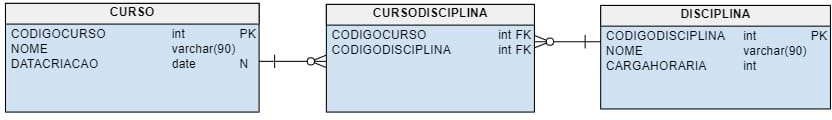
**FONTE: 0 AUTOR**

**SUPONHA TAMBEM QUE, APOS A EXECUCAO DA LINHA 2, 0 PROFISSIONAL PERCEBEU QUE NAO DEVERIA TERAUMENTADO 0 SALARIO DE MARIA NESSE VALOR. QUAL COMANDO E ADEQUADO ADICIONAR A LINHA 3 PARA DESFAZER ESSA OPERACAO?**

1. DELETE;
2. COMMIT;
3. UPDATE;
4. ROLLBACK;

GABARITO

**1. A respeito de transa oes no PostgreSQL e, considerando o modelo a seguir, assinale a proposi ao verdadeira:**

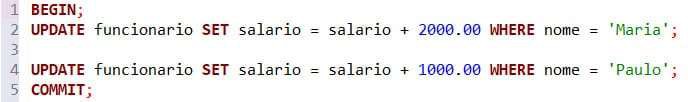


**Fonte: 0 autor**

A alternativa **"C** " esta correta.

De fato, se uma transac;ao no PostgreSQL e definida coma READ WRITE, ela aceita comandos de inserc;ao, atualizac;ao e remoc;ao de dados. Logo, a instruc;ao que contem o comando DELETE podera ser executada sem erro.

**2. Suponha que um profissional programou no PostgreSQL os seguintes comandos em uma tabela denominada empregado:**



**Fonte: 0 autor**

Suponha tambem que, ap6s a execu ao da linha 2, o profissional percebeu que nao

deveria ter aumentado o salario de Maria nesse valor. Qual comando e adequado adicionar a linha 3 para desfazer essa opera ao?

A alternativa **"D** " esta correta.

De fato, coma a transagao nao foi conclufda, e possfvel desfazer a operagao da linha 2, bastando para isso adicionar o comando ROLLBACK. Ap6s isso, a tabela funcionario ficara com os registros iguais a situagao imediatamente anterior a execugao da transagao.

### CONCLUSAO

**CONSIDERAC0ES FINAIS**

Ao longo do nosso estudo, fizemos uma introdugao aos recursos do SGBD PostgreSQL, envolvendo caracterfsticas desse SGBD, bem coma sua instalagao.

Foram apresentados comandos, classificados coma DDL, para a criagao ea alteragao de tabelas. Em seguida, conhecemos diversos comandos SQL para manipulagao de linhas em tabelas. Tais comandos, classificados coma DML, sao uteis para insergao, alteragao e remogao de dados.

Finalizamos com uma breve contextualizagao a respeito do conceito, uso e importancia das transagoes, com destaque aos comandos do PostgreSQL utilizados para esse fim.





##### REFERENCIAS

DBEAVER COMMUNITY. Consultado em meio eletr6nico em: 30 mai. 2020.

ELMASRI, R.; NAVATHE, S. Sistemas de Banco de Dados. 7. ed. Sao Paulo: Pearson, 2019.

MANZANO, J.A.M.G., Microsoft SQL Server 2016 Express Edition lnterativo. 1. ed. Sao Paulo: Saraiva, 2017.

POSTGRESQL. PostgreSQL 12.3 Documentation. Consultado em meio eletr6nico em: 30 mai. 2020.

POSTGRESQL. Create Table. Consultado em meio eletronico em: 30 mai. 2020. POSTGRESQL. Data Type. Consultado em meio eletronico em: 30 mai. 2020.

POSTGRESQL. Delete. Consultado em meio eletronico em: 30 mai. 2020.

POSTGRESQL. Download. Consultado em meio eletronico em: 30 mai. 2020. POSTGRESQL. Insert. Consultado em meio eletronico em: 30 mai. 2020.

POSTGRESQL. Ubuntu. Consultado em meio eletronico em: 30 mai. 2020.

POSTGRESQL. Update. Consultado em meio eletronico em: 30 mai. 2020.

##### EXPLORE+

Para aprofundar seus conhecimentos sobre o assunto deste tema, leia: CHAMBERLIN, D. D. Early History of SQL *In:* IEEE Annals of the History of

Computing n 4, 2012. Como vimos, a linguagem SQL tornou-se um padrao para uso em sistemas gerenciadores de bancos de dados relacionais. 0 artigo indicado e um interessante material sobre a hist6ria da SQL.



##### CONTEUDISTA

Nathielly de Souza Campos

(§> **CURR(CULO LATTES**