Linguagem de Definição de Dados (DDL)

Prof. Jeferson Souza, MSc.

(jefecomp)

jeferson.souza@udesc.br

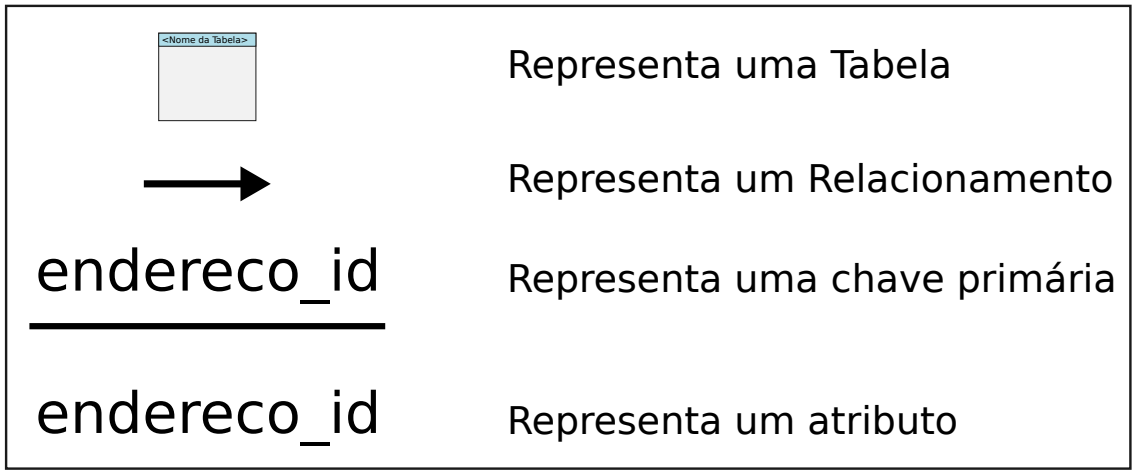
UDESC
UNIVERSIDADE
DO ESTADO DE
SANTA CATARINA

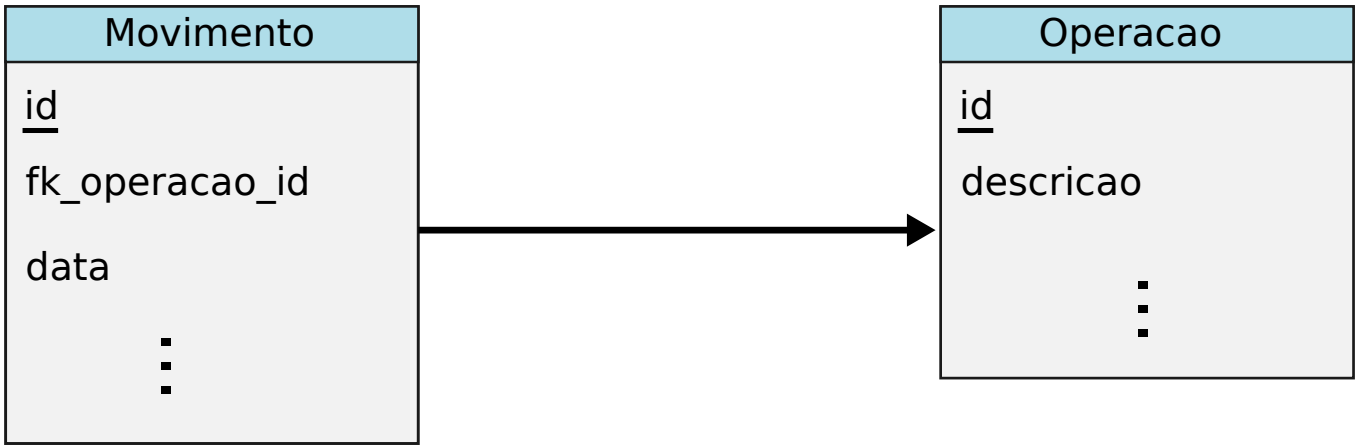
JOINVILLE
CENTRO DE CIÊNCIAS
TECNOLÓGICAS

Representação Gráfica e Simplificada do Schema

- Nas aulas anteriores vimos uma representação usando tabelas e comparamos com a representação realizada através do diagrama E-R;
- Agora veremos uma forma que permite uma visão "simplificada" dessas duas representações: O diagrama de esquema.

Diagrama de Schema





Introdução à Linguagem de Definição de Dados (DDL)

- Subconjunto da linguagem SQL;
- A sigla DDL é derivada do inglês Data Definition Language;
- Permite a criação de toda a estrutura de dados (dentro de um schema), incluindo tabelas, relacionamento, restrições, visões, entre outros.
- Permite especificar restrições sobre a estrutura de dados criada no banco. Exemplo: coluna nome na tabela "Usuario" não pode ser nulo.

Tipos de Restrições

- Domínio: assegura que os valores de uma dada coluna estão dentro de um domínio esperado;
- Integridade referencial: assegura que informações de uma dada coluna que fazem referências a dados de outra tabela existem, e são consistentes;

Tipos de Restrições (Continuação)

- Asserções: restrições que devem ser satisfeitas pela base de dados. Ex: Toda conta bancária deve ter, obrigatoriamente, um titular associado. Restrições de domínio e integridade referencial são casos especiais das asserções;

- Autorização: autoriza ou não autoriza o acesso aos dados a usuários da base de dados. Restrições mais comuns: leitura, inserção, atualização, remoção.

Criar Bases de Dados

CREATE DATABASE NOME_DB [OPÇÕES];

comando

argumento obrigatório

As opções incluem desde da especificação do "dono" da base de dados, até a indicação de qual template utilizar para criar a base de dados.

Criar Bases de Dados

CREATE DATABASE NOME_DB [OPÇÕES];

comando

argumento obrigatório

As opções incluem desde da especificação do "dono" da base de dados, até a indicação de qual template utilizar para criar a base de dados.

Dica

Consultem a ajuda (\h) do psql para ver detalhes do comando, já que vamos utilizar o PostgreSQL durante o curso.

Criar Bases de Dados

Exemplo

CREATE DATABASE banii;

Criar Bases de Dados

Exemplo

CREATE DATABASE banii;

Detalhe Importante

Não esqueçam de terminar os comandos com ponto e vírgula (;).

Criar Base de Dados Modelo

Exemplo

CREATE DATABASE banii **IS_TEMPLATE**=true;

Criar Bases de Dados com Modelo Específico

Exemplo

CREATE DATABASE banii_rocks **WITH TEMPLATE**=banii;

Remover Base de Dados

DROP DATABASE [IF EXISTS] NOME_DB;

comando

argumento obrigatório

Remover Base de Dados

Exemplo 1

DROP DATABASE banii;

Remover Base de Dados

Exemplo 1

DROP DATABASE banii;

Exemplo 2

DROP DATABASE IF EXISTS banii;

Remover Base de Dados

Exemplo 1

DROP DATABASE banii;

Exemplo 2

DROP DATABASE IF EXISTS banii;

DROP DATABASE VS **DROP DATABASE IF EXISTS**

Sem a utilização do **IF EXISTS** ao tentar remover uma base de dados que não existe, um erro será apresentado. Com o **IF EXISTS**, caso a base de dados não exista, o comando termina sem erros.

Criar Schema

CREATE SCHEMA NOME_SCHEMA [OPÇÕES];

comando

argumento obrigatório

ou

CREATE SCHEMA AUTHORIZATION ROLE_SPEC [OPÇÕES];

comando

argumento obrigatório

Criar Schema

CREATE SCHEMA NOME_SCHEMA [OPÇÕES];

comando

argumento obrigatório

ou

CREATE SCHEMA AUTHORIZATION ROLE_SPEC [OPÇÕES];

comando

argumento obrigatório

Existe mais duas variações que veremos durante as aulas práticas que usam a cláusula **IF NOT EXISTS**.

Criar Schema

Exemplo 1

CREATE SCHEMA banii_schema;

Exemplo 2

CREATE SCHEMA AUTHORIZATION postgres;

Criar Schema

Exemplo 1

CREATE SCHEMA banii_schema;

Exemplo 2

CREATE SCHEMA AUTHORIZATION postgres;

Importante!

No Exemplo 2 o nome do schema criado é o mesmo do nome do usuário, ou seja, postgres.

Remover Schema

DROP SCHEMA [IF EXISTS] NOME_SCHEMA, [, ...] [CASCADE | RESTRICT];

comando

argumento obrigatório

Remover Schema

DROP SCHEMA [IF EXISTS] NOME_SCHEMA, [, ...] [CASCADE | RESTRICT];

comando

argumento obrigatório

Importante!

RESTRICT é o comportamento padrão do comando, ou seja, não é possível remover um schema que não esteja vazio;

Remover Schema

Exemplo 1

DROP SCHEMA banii_schema;

Exemplo 2

DROP SCHEMA banii_schema CASCADE;

Remover Schema

Exemplo 1

DROP SCHEMA banii_schema;

Exemplo 2

DROP SCHEMA banii_schema CASCADE;

Nota

O parâmetro CASCADE indica que todo conteúdo do schema também será removido.

Tipos de Dados

Os tipos básicos de dados são ([SilberchatzEtAl, 2011]):

- char(n): tipo de tamanho fixo para armazenar, no máximo, n caracteres;
- varchar(n): tipo de tamanho variável para armazenar, no máximo, n caracteres.
- int: tipo para armazenar valores inteiros. Tamanho depende da arquitetura do computador;
- smallint: tipo para armazenar valores inteiros de menor tamanho. Tamanho depende da arquitetura do computador;

Tipos de Dados (Continuação)

- numeric(p,d): tipo numérico para armazenar valores com precisão definida pelo usuário. O argumento p indica o número de digitos inteiros (mais o sinal), e o argumento d o número de digitos da parte não inteira (depois da vírgula);
- real, double precision: tipo para representar números de ponto flutuante com precisão dependente da arquitetura do computador;
- float(n): tipo para armazenar números de ponto flutuante com precisão de n dígitos.

Tipos de Dados (Continuação)

Nota

É importante consultar o manual do banco de dados para verificar as declarações de tipos específicos. No nosso caso, consultem o manual do PostgreSQL.

Não esqueçam: O manual é o vosso melhor amigo (Além do professor, é claro :-D).

Criar Tabela

CREATE TABLE
comando

TABLE_NAME(nome_coluna tipo [restrições_coluna] [,nome_coluna ...],
[,restrições_colunas]);

argumento opcional

argumento obrigatório

Criar Tabela

Exemplo 1

CREATE TABLE usuario(id bigint, nome varchar(20), email varchar(30));

Exemplo 2

CREATE TABLE banii_schema.usuario(id bigint, nome varchar(20), email varchar(30));

Criar Tabela

Exemplo 1

CREATE TABLE usuario(id bigint, nome varchar(20), email varchar(30));

Exemplo 2

CREATE TABLE banii_schema.usuario(id bigint, nome varchar(20), email varchar(30));

Pergunta:

As tabelas criadas nos exemplos acima evitam a inserção de dados com todas as colunas iguais (i.e. dados repetidos)?

Criar Tabela com Chave Primária

Exemplo 1

CREATE TABLE usuario(id bigint *PRIMARY KEY*, nome varchar(20), email varchar(30));

Exemplo 2

CREATE TABLE usuario(id bigint, nome varchar(20), email varchar(30), *PRIMARY KEY*(id));

Criar Tabela com Valores Incrementados Automaticamente

Para criar tabelas com valores incrementados automaticamente no PostgreSQL basta utilizar um dos tipos serial.

Exemplo 1

CREATE TABLE usuario(id bigserial *PRIMARY KEY*, nome varchar(20), email varchar(30));

Exemplo 2

CREATE TABLE usuario(id serial, nome varchar(20), email varchar(30),*PRIMARY KEY*(id));

Criar Tabela com Valores Incrementados Automaticamente (Continuação)

Nota

Existe outra forma de criar tabelas com valores incrementados automaticamente, como veremos mais a frente no curso.

Criar Tabela com Chave Estrangeira

Exemplo 1

CREATE TABLE endereco(id bigserial *PRIMARY KEY*, logradouro varchar(50), cep char(8), cidade_id bigint references cidade);

Exemplo 2

CREATE TABLE endereco(id bigserial *PRIMARY KEY*, logradouro varchar(50), cep char(8), cidade_id bigint references cidade(id));

Criar Tabela com Chave Estrangeira (Continuação)

Exemplo 3

CREATE TABLE endereco(id bigserial *PRIMARY KEY*, logradouro varchar(50), cep char(8), cidade_id bigint, *FOREIGN KEY* (cidade_id) references cidade(id));

Criar Tabela com Restrições nas Colunas

Restrições mais comuns:

- NULL: coluna pode ou não conter valor (default);
- NOT NULL: coluna deve SEMPRE conter valor;
- UNIQUE: valor da couna deve ser único;
- DEFAULT: especifica um valor default para uma coluna.

Criar Tabela com Restrições nas Colunas

Exemplo 1

CREATE TABLE endereco(id bigserial *PRIMARY KEY*, logradouro varchar(50), cep char(8) *UNIQUE*, cidade_id bigint references cidade);

Exemplo 2

CREATE TABLE endereco(id bigserial *PRIMARY KEY*, logradouro varchar(50) *DEFAULT* 'SEM RUA' , cep char(8), cidade_id bigint references cidade(id));

Criar Tabela com Restrições nas Colunas (Continuação)

Exemplo 3

CREATE TABLE endereco(id bigserial *PRIMARY KEY*, logradouro varchar(50), cep char(8), cidade_id bigint *NOT NULL*, *FOREIGN KEY* (cidade_id) references cidade(id));

Alterar Estrutura de Tabelas

ALTER TABLE NOME_TABELA [OPÇÕES]

comando

argumento obrigatório

As opções são muitas, e portanto serão apresentados alguns exemplos a seguir.

Adicionar coluna a Tabela

Exemplo 1

ALTER TABLE endereco *ADD* descricao varchar(30);

Exemplo 2

ALTER TABLE endereco *ADD* estado_id bigint references estado;

Remover coluna da Tabela

Exemplo 1

ALTER TABLE endereco *DROP* descricao;

Exemplo 2

ALTER TABLE endereco *DROP* descricao *CASCADE*;

Alterar Restrições de Coluna

Exemplo

ALTER TABLE endereco *ALTER* descricao *SET NOT NULL*;

Adicionar Restrições a Coluna

Exemplo 1

ALTER TABLE endereco *ADD UNIQUE*(descricao);

Exemplo 2

ALTER TABLE endereco *ADD* CONSTRAINT descricao_unique *UNIQUE*(descricao);

Remover Restrições da Coluna

Exemplo

ALTER TABLE endereco *DROP* CONSTRAINT descricao_unique;

Importante

É necessário saber o nome da restrição para remover a mesma da tabela.

Remover Tabela

DROP TABLE [IF EXISTS] NOME_TABELA [CASCADE | RESTRICT];

comando

argumento obrigatório

Remover Tabela

Exemplo 1

DROP TABLE endereco;

Exemplo 2

DROP TABLE endereco CASCADE;

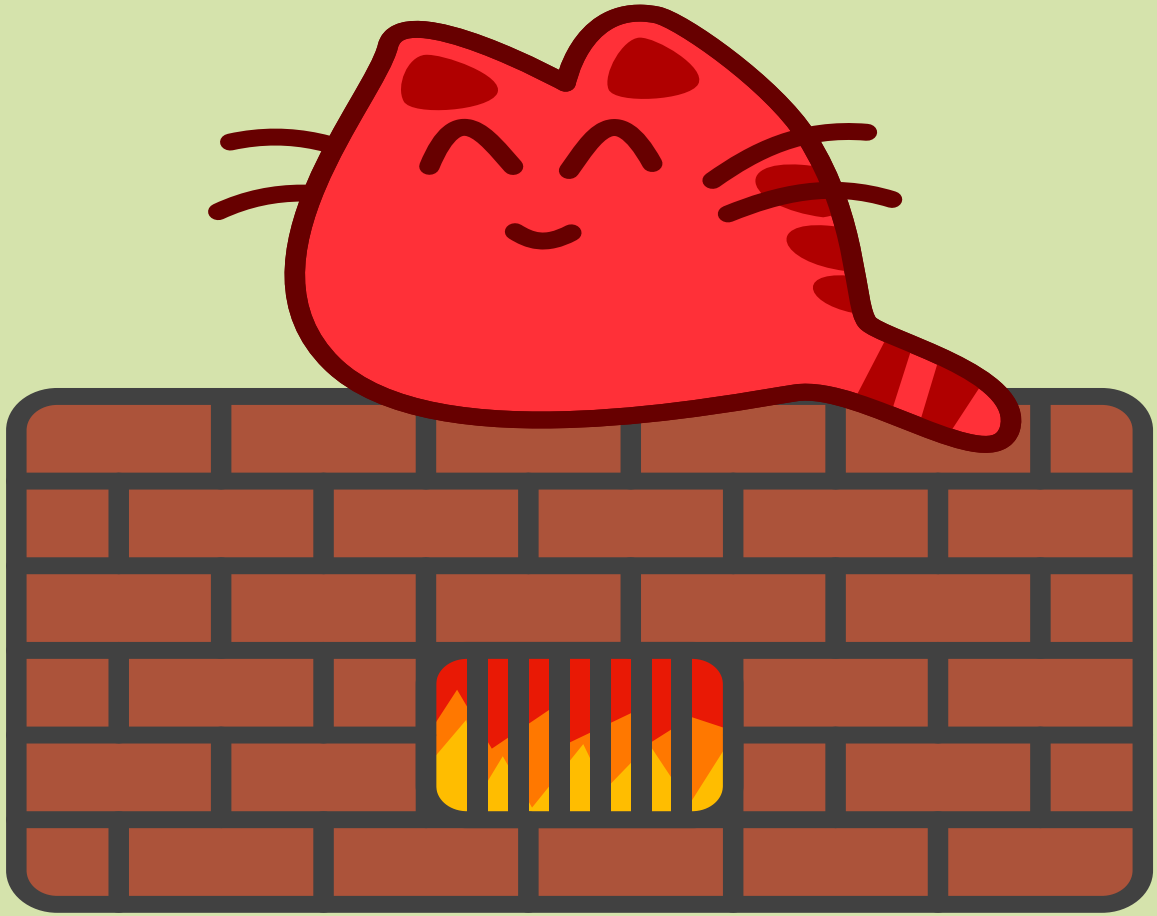
Bibliografia

Garcia-Molina, H. and Ullman, J. D. and Widom, J. *"Database Systems: The Complete Book"*. 2nd edition. Prentice Hall, 2008.

PostgreSQL Development Group. *"PostgreSQL 10.2 Documentation"*. 2018.

Silberschatz, A. and Korth, H.F. and Sudarshan, S. *"Database Systems"*. 6th edition. McGrawHill, 2011.

That's it folks!



Thank you for your attention!