

Linguagem de Definição de Dados (DDL) - Continuação



Prof. Jeferson Souza, MSc.

(jefecom)

jeferson.souza@udesc.br

Integridade do dados

Restrições de integridade

Como visto nos slides anteriormente no material do curso é possível criar restrições na definição das tabelas do banco de dados. A definição de restrições permite especificar o modelo de consistência que os dados devem seguir, e portanto são denominadas **restrições de integridade**.

Exemplos

- ▶ O nome do usuário não pode ser nulo;
- ▶ Dois usuários não podem ter o mesmo cpf;
- ▶ O saldo da conta deve ser sempre maior do que R\$50,00;

Restrições de integridade

Algumas das formas de especificar restrições de integridade já foram vistas anteriormente:

- ▶ **NULL;**
- ▶ **NOT NULL;**
- ▶ **UNIQUE;**
- ▶ **DEFAULT.**

Entretanto, vamos ver mais alguns exemplos da restrição **UNIQUE**.

Restrições de integridade: *UNIQUE*

Exemplo de candidatos a chave com *UNIQUE*

```
CREATE TABLE usuario(id serial PRIMARY KEY, nome  
varchar(20), email varchar(30), cpf char (11).,  
UNIQUE(email,cpf));
```

Restrições de integridade: *CHECK*

Com a restrição do tipo check é possível definir uma expressão (ex: $\text{preco} > 0$) que deve ser satisfeita durante a manipulação dos dados.

Exemplo 1

```
CREATE TABLE conta(nr_conta bigserial, agencia bigint, saldo  
numeric, PRIMARY KEY (nr_conta,agencia), FOREIGN  
KEY(agencia) references agencia(id), CHECK(saldo > 0));
```

Exemplo 2

```
CREATE TABLE departamento (id bigserial PRIMARY KEY,  
nome varchar(30), sigla varchar(4), CHECK(sigla in  
('DCC','DMAT','DEC','DEE')));
```

Restrições de integridade: *CHECK* (Continuação)

Exemplo 3

```
CREATE TABLE departamento (id bigserial PRIMARY KEY,  
nome varchar(30), sigla varchar(4), CHECK(sigla in (select sigla  
from siglas where tipo = 'departamento')));
```

Integridade Referencial

Conceito

Integridade referencial visa garantir que um conjunto de valores que apareçam em uma determinada tabela, sejam consistentes com os valores de sua tabela de referência.

Exemplo 1

```
CREATE TABLE usuario (id bigserial PRIMARY KEY, nome varchar(30), dept bigint references departamento);
```

Integridade Referencial (Continuação)

Exemplo 2

```
CREATE TABLE departamento (id bigserial PRIMARY KEY,  
nome varchar(30), sigla char(3) UNIQUE NOT NULL);
```

```
CREATE TABLE curso (id bigserial PRIMARY KEY, nome  
varchar(30), dept char(3) UNIQUE NOT NULL references  
departamento (sigla));
```

Importante

Colunas que não sejam chaves primárias para serem referenciadas devem ser, pelo menos, únicas.

Integridade Referencial: Operações em cascata

É possível realizar operações em cascata na base de dados para manter a integridade referencial. Para isso, basta definir as chaves estrangeiras com a cláusula **cascade**.

Exemplo

```
CREATE TABLE curso (id bigserial PRIMARY KEY, nome  
varchar(30), dept char(3) UNIQUE NOT NULL, FOREIGN  
KEY(dept) references departamento (sigla) on delete cascade  
on update cascade);
```

Visões (Views)

O que são visões (views)?

Visões são uma forma alternativa de acessar dados no modelo de dados. As visões são criadas a partir de consultas válidas realizadas sobre tabelas ou outras visões;

Visões (Views)

O que são visões (views)?

Visões são uma forma alternativa de acessar dados no modelo de dados. As visões são criadas a partir de consultas válidas realizadas sobre tabelas ou outras visões;

Para que servem as visões (views)?

- ▶ Fornecem um meio de acesso mais simples e direto ao dados;
- ▶ Permitem acesso controlado e limitado a dados com restrições de acesso;
- ▶ Permitem "estender" virtualmente o modelo de dados, e criar relações virtuais que podem ser utilizadas exatamente da mesma forma que tabelas.

Diferença entre tabelas e visões (views)

- ▶ Tabelas são estruturas criadas para armazenar dados;
- ▶ Visões são o resultado de consultas que podem ser manipuladas posteriormente da mesma forma que tabelas;
- ▶ Caso não exista a necessidade de relacionamentos, tabelas podem ser criadas sem a existência de outras estruturas na base de dados;
- ▶ A criação de uma visão depende da existência de, ao menos, uma tabela;

Diferença entre tabelas e visões (views)

(Continuação)

- ▶ Os dados mostrados por uma visão podem ser modificados (em alguns casos) por comandos de inserção, remoção, e atualização. Entretanto, modificações não são recomendadas, já que as mesmas devem refletir modificações nas tabelas que dão origem a visão alvo;

Criar visões (views)

CREATE VIEW

comando

NOME_DA VISÃO

argumento
obrigatório

(nome das colunas)

argumento
opcional

AS

<Consulta>;

argumento
obrigatório

Criar visões (views)

Exemplos:

OBS: cargo_id = 1 é o identificador para o cargo de Professor.

```
CREATE VIEW professores_view as SELECT nome, sobrenome,  
email, cpf, cargo_id, departamento_id FROM usuario where  
cargo_id = 1;
```

Criar visões (views)

Exemplos:

OBS: cargo_id = 1 é o identificador para o cargo de Professor.

```
CREATE VIEW professores_view as SELECT nome, sobrenome,  
email, cpf, cargo_id, departamento_id FROM usuario where  
cargo_id = 1;
```

Pode-se depois realizar consultas diretamente na Visão(View)

```
SELECT * FROM professores_view where departamento_id = 1;
```

OBS: departamento_id=1 é o identificador do DCC.

Visões Materializadas (Materialised Views)

Alguns SGBDs permitem que visões sejam armazenadas em disco. Esse tipo de visão é chamada de **visão materializada (materialised view)**. Caso dados sejam inseridos na base de dados, e o resultado da consulta que define a visão mude, a visão materializada também é atualizada.

Criar Visões Materializadas (Materialised Views)

CREATE MATERIALIZED VIEW **NOME_DA VISÃO** (**nome das colunas**) AS **<Consulta>**;
comando argumento obrigatório argumento opcional argumento obrigatório

Criar Visões Materializadas (Materialised Views)

```
CREATE MATERIALIZED VIEW professores_view as SELECT  
nome, sobrenome, email, cpf, cargo_id, departamento_id FROM  
usuario where cargo_id = 1;
```

```
SELECT * FROM professores_view where departamento_id = 1;
```

Criar Usuário e Grupo

Um passo antes dos privilégios

Antes de entrarmos efetivamente no domínio de privilégios é necessário aprendermos a criar usuários e grupos no SGBD.

Criar Usuário ou Grupo

Para criar um usuário ou um grupo no postgres podemos usar os seguintes comandos:

CREATE USER	NOME_USUÁRIO	[WITH] options [...];
comando	argumento obrigatório	argumento opcional

OR

CREATE ROLE	NOME_USUÁRIO	[WITH] options [...];
comando	argumento obrigatório	argumento opcional

Criar Usuário ou Grupo

Opções mais importantes:

- ▶ *SUPERUSER*: cria um usuário ou grupo com poderes de super usuário;
- ▶ *CREATEDB*: cria um usuário ou grupo com permissão para criar bases de dados;
- ▶ *CREATEROLE*: cria um usuário ou grupo com permissão para criar outros usuários ou grupos;
- ▶ *LOGIN*: cria um usuário ou grupo com permissão para conectar no SGBD;

Criar Usuário ou Grupo (Continuação)

- ▶ *PASSWORD*: cria um usuário ou grupo com permissão para conectar no SGBD.
- ▶ *INHERIT*: cria um usuário ou grupo com permissão de utilização de todos os privilégios dos grupos, o qual o dado usuário ou grupo é membro.

Criar Usuário ou Grupo: Exemplo

Exemplo 1

```
CREATE USER jefecomp;
```

Exemplo 2

```
CREATE USER jefecomp with PASSWORD '!#hammer22';
```

Exemplo 3

```
CREATE ROLE jefecomp;
```

Exemplo 4

```
CREATE ROLE jefecomp with PASSWORD '!#hammer22';
```


Criar Usuário: Exemplo (Continuação)

Pergunta:

Qual a diferença entre usar o **CREATE USER** e o **CREATE ROLE**?

Criar Usuário: Exemplo (Continuação)

Pergunta:

Qual a diferença entre usar o **CREATE USER** e o **CREATE ROLE**?

Resposta:

O comando **CREATE USER** é um “alias” sobre o comando **CREATE ROLE** com a opção **LOGIN**. Logo, é necessário passar a opção **LOGIN** explicitamente quando criar um usuário com o comando **CREATE ROLE**.

Criar Usuário: Exemplo (Continuação)

Pergunta:

Caso eu esqueça de adicionar alguma opção na criação do meu usuário, o que eu posso fazer para alterar?

Criar Usuário: Exemplo (Continuação)

Pergunta:

Caso eu esqueça de adicionar alguma opção na criação do meu usuário, o que eu posso fazer para alterar?

Resposta:

Usar o comando **ALTER USER**. Exemplo:
ALTER USER jefecomp with LOGIN;

O que São Privilégios?

As vezes queremos restringir o que um determinado usuário (ou grupo de usuários) da base de dados pode fazer, ou seja, que tipo de operações podem ser executadas. Exemplos dessas operações são: leitura, escrita, atualização, e remoção.

O que São Privilégios?

As vezes queremos restringir o que um determinado usuário (ou grupo de usuários) da base de dados pode fazer, ou seja, que tipo de operações podem ser executadas. Exemplos dessas operações são: leitura, escrita, atualização, e remoção.

Privilégio

Um privilégio especifica uma autorização concedida a um dado usuário que permita a execução de cada uma dessas operações.

Atribuir Privilégios

GRANT <Lista de Privilégios> **ON** <OBJETO> **TO** <Lista Usuários/Grupos> [WITH GRANT OPTION];

comando argumento obrigatório argumento obrigatório argumento obrigatório argumento opcional

Lista de Privilégios - Mais Comuns

- ▶ *SELECT*: permite realizar operações de consulta;
- ▶ *INSERT*: permite realizar operações de inserção de dados;
- ▶ *UPDATE*: permite realizar operações de atualização de dados;
- ▶ *DELETE*: permite realizar operações de remoção de dados;
- ▶ *REFERENCES*: permite realizar operações de criação de chaves estrangeiras;
- ▶ *USAGE*: no geral permite o uso/acesso a recursos associados a um dado objeto (ex: schemas);

Lista de Privilégios - Mais Comuns (Continuação)

- ▶ *CREATE*: permite realizar operações de criação de objetos (ex: tabelas, schemas, sequências, etc);
- ▶ *CONNECT*: permite conexão com uma dada base de dados;
- ▶ *ALL PRIVILEGES*: permite o uso de todos os privilégios de um dado objeto.

Atribuir Privilégios - Schema

Exemplo 1

```
GRANT CREATE ON SCHEMA banii TO jefecomp;
```

Exemplo 2

```
GRANT CREATE ON SCHEMA banii TO jefecomp WITH  
GRANT OPTION;
```

Exemplo 3

```
GRANT ALL PRIVILEGES ON SCHEMA banii TO PUBLIC;
```

Atribuir Privilégios - Base de Dados

Exemplo 1

```
GRANT CREATE, CONNECT ON DATABASE privilegios TO  
jefecomp;
```

Exemplo 2

```
GRANT CONNECT ON DATABASE privilegios TO jefecomp  
WITH GRANT OPTION;
```

Atribuir Privilégios - Tabelas

Exemplo 1

```
GRANT SELECT,INSERT ON movimentos TO jefecomp,lucas;
```

Exemplo 2

```
GRANT SELECT,UPDATE ON cliente TO jefecomp;
```

Exemplo 3

```
GRANT SELECT,INSERT,UPDATE ON ALL TABLES IN  
SCHEMA banii TO jefecomp,lucas;
```

Exemplo 4

```
GRANT SELECT(nome) ON usuario TO jefecomp;
```

Atribuir Privilégios - Tabelas (Continuação)

Exemplo 5

```
GRANT UPDATE(descricao) ON endereco TO lucas;
```

Exemplo 6

```
GRANT SELECT,DELETE ON produto TO lucas;
```

Exemplo 7

```
GRANT REFERENCES ON equipamento TO lucas;
```

Exemplo 8

```
GRANT REFERENCES(cpf) ON usuario TO jefecomp;
```

Atribuir Privilégios - Tabelas (Continuação)

Exemplo 9

```
GRANT REFERENCES(cpf) ON usuario TO jefecomp WITH  
GRANT OPTION;
```

Retirar Privilégios

REVOKE **[GRANT OPTION FOR]** **<Lista de Privilégios>** **ON** **<OBJETO>** **FROM** **<Lista Usuários/Grupos>** **[CASCADE|RESTRICT];**

comando argumento argumento argumento argumento argumento

opcional obrigatório obrigatório obrigatório opcional

Retirar Privilégios - Base de Dados

Exemplo 1

```
REVOKE CONNECT ON DATABASE manufatura FROM  
jefecomp;
```

Exemplo 2

```
REVOKE GRANT OPTION FOR CONNECT ON DATABASE  
manufatura TO jefecomp;
```


Retirar Privilégios - Tabelas

Exemplo 1

```
REVOKE SELECT,INSERT ON movimentos FROM jefecomp;
```

Exemplo 2

```
REVOKE GRANT OPTION FOR UPDATE ON produto FROM  
jefecomp;
```

Exemplo 3

```
REVOKE SELECT(nome) ON usuario FROM jefecomp;
```

Retirar Privilégios - Tabelas (Continuação)

Exemplo 4

```
REVOKE SELECT,INSERT,UPDATE ON ALL TABLES IN  
SCHEMA banii FROM zezinho;
```

Exemplo 5

```
REVOKE SELECT,INSERT,UPDATE ON ALL TABLES IN  
SCHEMA banii FROM zezinho CASCADE;
```

Exemplo 6

```
REVOKE GRANT OPTION FOR UPDATE ON produto FROM  
jefecomp CASCADE;
```

Privilégios e Grupos - Atribuição

É possível também atribuir usuários a grupos.

Exemplo 1

```
GRANT professores TO jefecomp;
```

Exemplo 2

```
GRANT professores TO jefecomp WITH ADMIN OPTION;
```

Privilégios e Grupos - Retirada

Ou, de forma similar, retirar usuários de grupos.

Exemplo 1

```
REVOKE administrador FROM jefecomp;
```

Exemplo 2

```
REVOKE ADMIN OPTION FOR professores FROM jefecomp;
```

Exemplo 3

```
REVOKE ADMIN OPTION FOR professores FROM jefecomp  
CASCADE;
```

Exemplo 4

```
REVOKE administrador FROM jefecomp CASCADE;
```

Bibliografia



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

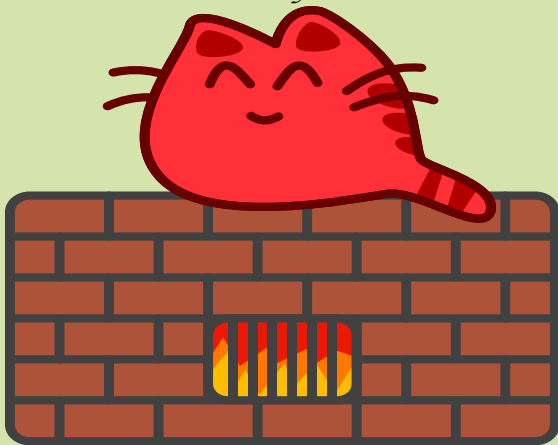


PostgreSQL Development Group.
"PostgreSQL 10.2 Documentation". 2018.



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

That's it folks!



Thank you for your attention!