LABORATÓRIO DE BANCO DE DADOS

DML - Sequência

PostgreSQL

Vanessa Borges – vanessa@facom.ufms.br

Manipulação de sequências
CREATE SEQUENCE

- Os objetos de sequência (também chamados de geradores de sequência, ou simplesmente de sequências), são tabelas especiais de um única linha criadas pelo comando **CREATE SEQUENCE**.
- O objeto de sequência é usado normalmente para gerar identificadores únicos para linhas de tabelas.

CREATE SEQUENCE cria um novo gerador de números de sequência

```
CREATE SEQUENCE [IF NOT EXISTS] <nome>
[AS <tipo do dado>]
[INCREMENT <valor>]
[MINVALUE <valor> | NO MINVALUE]
[MAXVALUE <valor> | NO MAXVALUE]
[START <valor>]
    ...;
```

Manipulação de sequências
CREATE SEQUENCE

- **CREATE SEQUENCE** cria um novo gerador de números de sequência

```
CREATE SEQUENCE [IF NOT EXISTS] <nome>
[AS <tipo do dado>]
[INCREMENT <valor>]
[MINVALUE <valor> | NO MINVALUE]
[MAXVALUE <valor> | NO MAXVALUE]
[START <valor>]
...;
```

- **Especifica qual valor é adicionado ao valor da sequência atual para criar um novo valor.**
- **Um valor positivo fará uma sequência ascendente, um negativo, uma sequência descendente.**
- **O valor padrão é 1.**

Manipulação de sequências
CREATE SEQUENCE

- **CREATE SEQUENCE** cria um novo gerador de números de sequência

```
CREATE SEQUENCE [IF NOT EXISTS] <nome>
[AS <tipo do dado>]
[INCREMENT <valor>]
[MINVALUE <valor> | NO MINVALUE]
[MAXVALUE <valor> | NO MAXVALUE]
[START <valor>]
        ...;
```

Determina o valor mínimo que uma sequência pode gerar

Determina o valor máximo que uma sequência pode gerar

Manipulação de sequências
CREATE SEQUENCE

- **CREATE SEQUENCE** cria um novo gerador de números de sequência

```
CREATE SEQUENCE [IF NOT EXISTS] <nome>
[AS <tipo do dado>]
[INCREMENT <valor>]
[MINVALUE <valor> | NO MINVALUE]
[MAXVALUE <valor> | NO MAXVALUE]
[START <valor>]
        ...;
```

- **Permite que a sequência comece em qualquer valor. O valor inicial padrão é minvalue para sequências ascendentes e maxvalue para descendentes ascendentes.**

- **Para consultar quais sequences estão criadas no psql utilize o comando \ds**

Manipulação de sequências NEXTVAL / CURRVAL / SETVAL

- Funções para manipulação de SEQUENCE

Função	Tipo retornado	Descrição
nextval(text)	bigint	Avança a sequência e retorna o novo valor
currval(text)	bigint	Retorna o valor obtido mais recentemente por nextval
setval(text, bigint)	bigint	Define o valor corrente da sequência
lastval()	bigint	Retorna o valor mais recente retornado por nextval na sessão corrente

Manipulação de sequências
CREATE SEQUENCE

Exemplos:

NEXTVAL: retorna o próximo valor e deixa a sequencia incrementada (não existe ROLLBACK)

ALUNO

id	nome
1	Reinaldo
2	Ana
3	Carlos

Manipulação de sequências
CREATE SEQUENCE

Exemplos:

NEXTVAL: retorna o próximo valor e deixa a sequencia incrementada (não existe ROLLBACK)

```
CREATE SEQUENCE alunoid_seq AS
INTEGER INCREMENT 1 MINVALUE 1 MAXVALUE 999999999 START 1;

CREATE TABLE aluno (
    id INTEGER DEFAULT NEXTVAL('alunoid_seq') PRIMARY KEY,
    nome VARCHAR(255)
);

INSERT INTO aluno (nome) VALUES (‘Reinaldo'),('Ana'),('Carlos');
```

ALUNO

id	nome
1	Reinaldo
2	Ana
3	Carlos

Manipulação de sequências
ALTER SEQUENCE

- **ALTER SEQUENCE:** altera a definição de um gerador de sequência

```
ALTER SEQUENCE <nome da sequence>
[INCREMENT <valor>]
[MINVALUE <valor> | NOMINVALUE]
[MAXVALUE<valor> | NOMAXVALUE]
[START <valor>];
```

- Exemplo:

```
ALTER SEQUENCE alunoid1_seq INCREMENT 2;
```

Manipulação de sequências
CREATE SEQUENCE

Exemplos:

NEXTVAL: retorna o próximo valor e deixa a sequencia incrementada (não existe ROLLBACK)

```
ALTER SEQUENCE alunoid_seq INCREMENT 2;
```

```
INSERT INTO aluno (nome) VALUES ('Rafael'),('Lucas'),('Emily');
```

ALUNO

id	nome
1	Reinaldo
2	Ana
3	Carlos
5	Rafael
7	Lucas
9	Emily

Manipulação de sequências NEXTVAL

Função	Tipo retornado	Descrição
nextval(text)	bigint	Avança a sequência e retorna o novo valor
currval(text)	bigint	Retorna o valor obtido mais recentemente por nextval
setval(text, bigint)	bigint	Define o valor corrente da sequência

```sql
-- Retorna o último valor da sequence
SELECT last_value FROM alunoid_seq;

-- Avança a sequence e retorna o último valor
SELECT nextval('alunoid_seq') AS proximo_valor;
```

Manipulação de sequências
CREATE SEQUENCE

```
CREATE SEQUENCE alunoid1_seq
AS INTEGER INCREMENT 1 MINVALUE 1 MAXVALUE 999999999 START 1;

CREATE TABLE aluno (
    id1 INTEGER DEFAULT NEXTVAL('alunoid1_seq'),
    id2 INTEGER DEFAULT NEXTVAL(‘alunoid1_seq'),
    nome VARCHAR(255)
);

INSERT INTO aluno (nome) VALUES ('Reinaldo'),('Ana'),('Carlos');
```

ALUNO

id1	id2	nome
1	2	Reinaldo
3	4	Ana
5	6	Carlos

Manipulação de sequências NEXTVAL / CURRVAL / SETVAL

- Funções para manipulação de SEQUENCE

Função	Tipo retornado	Descrição
nextval(text)	bigint	Avança a sequência e retorna o novo valor
currval(text)	bigint	Retorna o valor obtido mais recentemente por nextval
setval(text, bigint)	bigint	Define o valor corrente da sequência

Manipulação de sequências
CURRVAL

- Exemplos:
  - **CURRVAL:** retorna o valor atual e deixa a sequência como está

```
-- Retorna o valor mais recente obtido por nextval
SELECT currval('alunoid1_seq') AS valor_corrente;
```

Manipulação de sequências NEXTVAL / CURRVAL / SETVAL

- Funções para manipulação de SEQUENCE

Função	Tipo retornado	Descrição
nextval(text)	bigint	Avança a sequência e retorna o novo valor
currval(text)	bigint	Retorna o valor obtido mais recentemente por nextval
setval(text, bigint)	bigint	Define o valor corrente da sequência

Manipulação de sequências SETVAL

- Exemplos:
  - **SETVAL: atualiza o valor corrente da sequencia**

```
-- Define o valor corrente da sequence para 100
-- O próximo nextval() retornará 101 (incremento de 1)
SELECT setval('alunoid1', 100) as valor_corrente;
```

Manipulação de sequências
DROP SEQUENCE

- **DROP SEQUENCE:** remove a definição de um gerador de sequência

```
DROP SEQUENCE <nome da sequence>
[CASCADE | RESTRICT];
```

- Exemplo:

```
DROP SEQUENCE alunoid1_seq CASCADE;
```

Auto-incremento – SERIAL

- Os tipos de dados **serial** e **bigserial não são tipos verdadeiros**, mas apenas uma notação conveniente para criar **colunas de identificador exclusivo**
  - Semelhante à propriedade AUTO_INCREMENT suportada por alguns outros bancos de dados

Name	Storage Size	Description	Range
smallint	2 bytes	small-range integer	-32768 to +32767
integer	4 bytes	typical choice for integer	-2147483648 to +2147483647
bigint	8 bytes	large-range integer	-9223372036854775808 to 9223372036854775807
decimal	variable	user-specified precision, exact	up to 131072 digits before the decimal point; up to 16383 digits after the decimal point
numeric	variable	user-specified precision, exact	up to 131072 digits before the decimal point; up to 16383 digits after the decimal point
real	4 bytes	variable-precision, inexact	6 decimal digits precision
double precision	8 bytes	variable-precision, inexact	15 decimal digits precision
serial	**4 bytes**	**autoincrementing integer**	**1 to 2147483647**
bigserial	**8 bytes**	**large autoincrementing integer**	**1 to 9223372036854775807**

https://www.postgresql.org/docs/current/datatype-numeric.html

Auto-incremento – SERIAL

Definição de campo auto-incremento no PostgreSQL

```
CREATE TABLE <nome da tabela> (
        <nome da coluna> SERIAL
);
```

É equivalente a:

```
CREATE SEQUENCE <nome da sequence> AS integer;
CREATE TABLE <nome da tabela> (
    <nome da coluna> integer NOT NULL DEFAULT nextval('<nome da sequence>')
);
```

Manipulação de sequências SERIAL

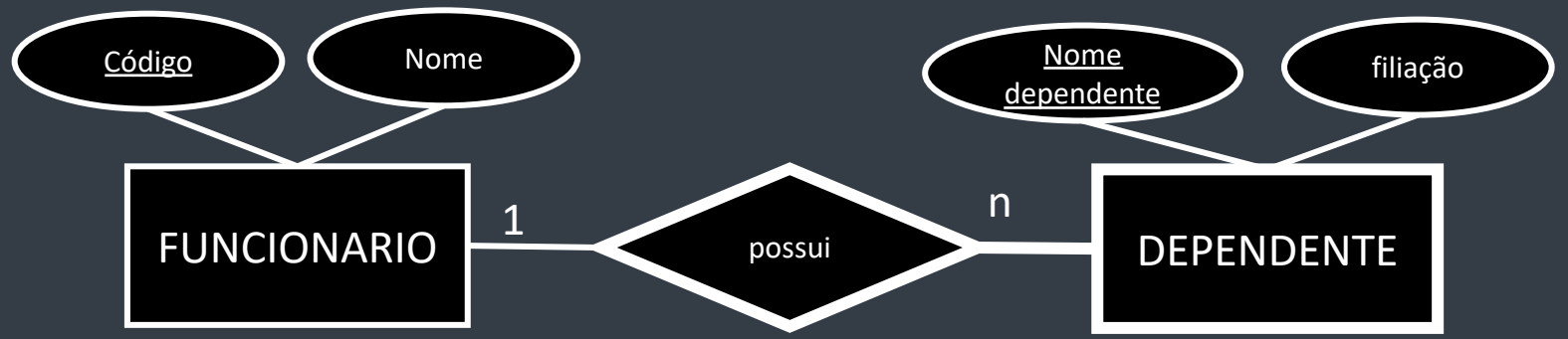
```sql
CREATE TABLE testeserial (
    id SERIAL PRIMARY KEY,
    nome VARCHAR(255)
);

INSERT INTO testeserial (nome) VALUES ('Reinaldo'),('Ana'),('Carlos');
```

TESTESERIAL

id	nome
1	Reinaldo
3	Ana
5	Carlos

Problema: chave primária composta




```sql
CREATE TABLE FUNCIONARIO (
    codfuncionario integer PRIMARY KEY,
    nome VARCHAR(200)
);
```

```sql
CREATE TABLE DEPENDENTE (
 codfuncionario integer,
 nome_dependente VARCHAR(100),
 filiacao VARCHAR(50),
 PRIMARY KEY (codfuncionario, nome_dependente),
 FOREIGN KEY (codfuncionario)
 REFERENCES FUNCIONARIO(codfuncionario)
);
```

Exemplo de Chave primária composta

```sql
CREATE TABLE FUNCIONARIO (
    codfuncionario integer PRIMARY KEY,
    nome VARCHAR(200)
);
```

```sql
CREATE TABLE DEPENDENTE (
 codfuncionario integer,
 nome_dependente VARCHAR(100),
 filiacao VARCHAR(50),
 PRIMARY KEY (codfuncionario, nome_dependente),
 FOREIGN KEY (codfuncionario)
 REFERENCES FUNCIONARIO(codfuncionario)
);
```

FUNCIONARIO

codfuncionario	nome
1	Reinaldo
2	Ana
3	João

DEPENDENTE

codfuncionario	Nome_dependente	filiacao
1	Renata Vasconcelos	FILHA
1	Xuxa Meneguel	CONJUJE
3	Ingrid Guimarães	CONJUJE

Exemplo de Chave primária composta

```
CREATE TABLE FUNCIONARIO (
    codfuncionario SERIAL PRIMARY KEY,
    nome VARCHAR(200)
);
```

```
CREATE TABLE DEPENDENTE (
 coddependente SERIAL PRIMARY KEY,
 codfuncionario integer NOT NULL,
 nome_dependente VARCHAR(100) NOT NULL,
 filiacao VARCHAR(50),
 UNIQUE (codfuncionario, nome_dependente),
 FOREIGN KEY (codfuncionario) REFERENCES FUNCIONARIO(
codfuncionario)
);
```

FUNCIONARIO

codfuncionario	nome
1	Reinaldo
2	Ana
3	João

DEPENDENTE

coddependente	codfuncionario	Nome_dependente	filiacao
1	1	Renata Vasconcelos	FILHA
2	1	Xuxa Meneguel	CONJUJE
3	2	Ingrid Guimarães	CONJUJE