PCS3413

Engenharia de Software e Banco de Dados

Aula Views e Sequencias

Escola Politécnica da Universidade de São Paulo

Segurança em Banco de dados

VIEWS

Segurança em Banco de Dados

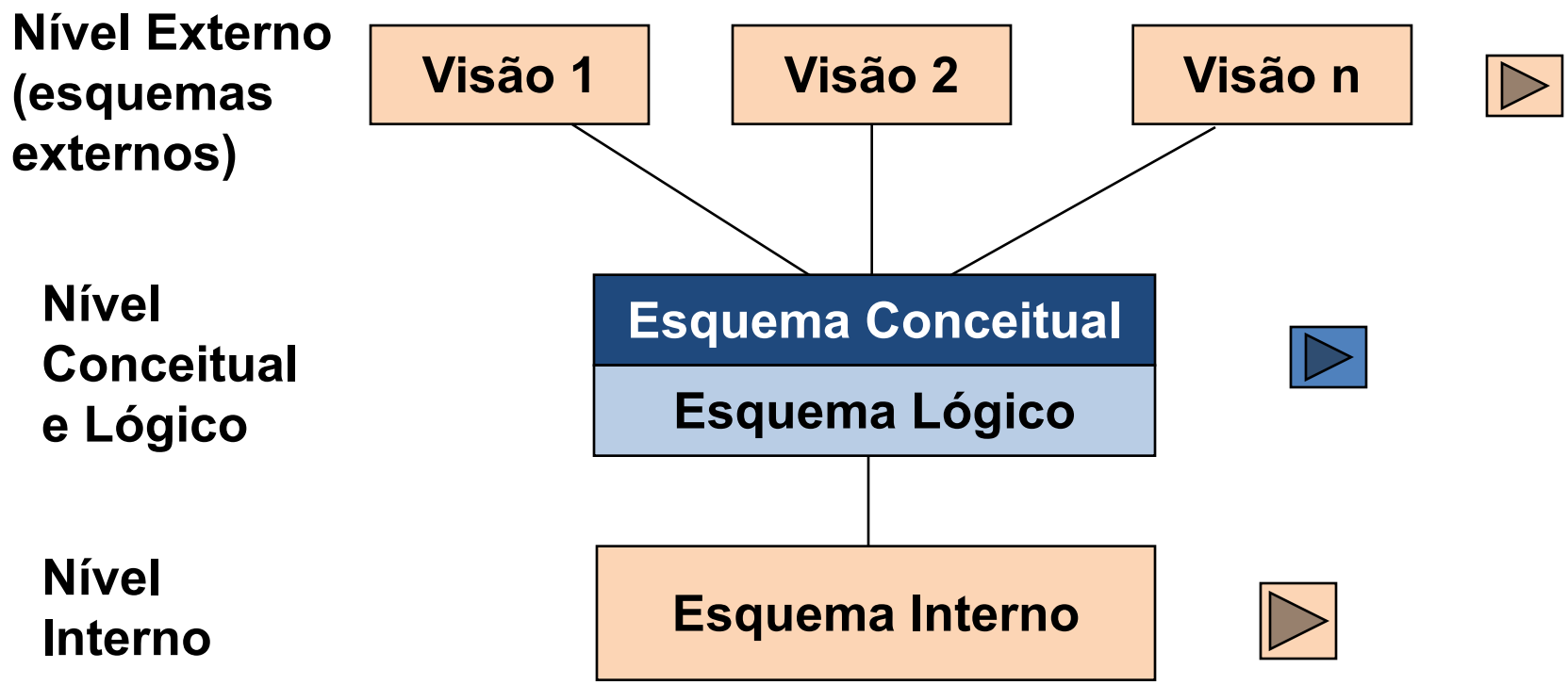
- proteção dos dados contra acessos não autorizados
  - restrições no emprego de operações.
  - visão parcial dos dados:
    - Permite especificar o que usuários/aplicações terão acesso.
    - Nem sempre é desejável que todos os usuários tenham acesso a todas as informações de uma tabela. Aspectos de segurança podem exigir que determinados dados não estejam disponíveis para todos os usuários

Visão Parcial dos

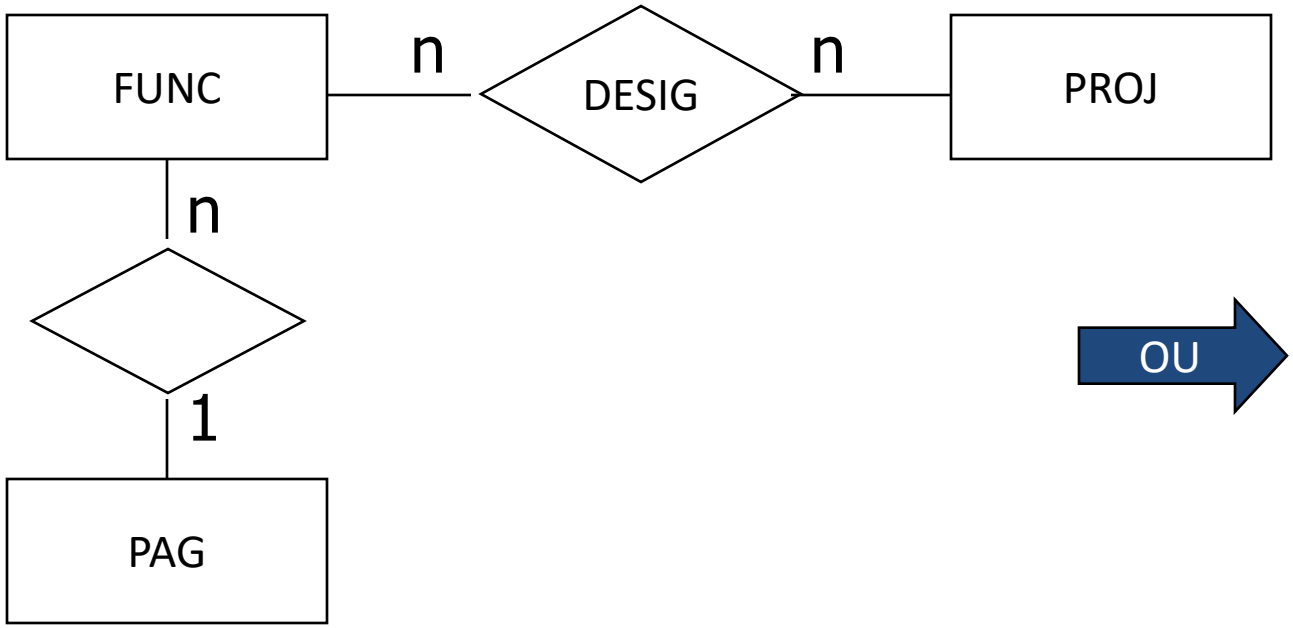
ANSI (*American National Standards Institute*)

SPARC (*Standards Planning and Requirements Commitee*)

- Arquitetura ANSI/SPARC



ou Esquema conceitual



exemplo do Esquema Lógico

☞ representação relacional
☞ não é o único formalismo

ansi/
sparc

relation FUNC [
key = {FNO}
attributes = {
FNO: character(9)
FNOME : character(15)
CARGO : character(10) }]

relation PROJ [
key = {PNO}
attributes = {
PNO : character(7)
PNOME : character(20)
ORCAMEN : numeric(7) }]

relation PAG [
key = {CARGO}
attributes = {
CARGO : character(10)
SALARIO: numeric(6) }]

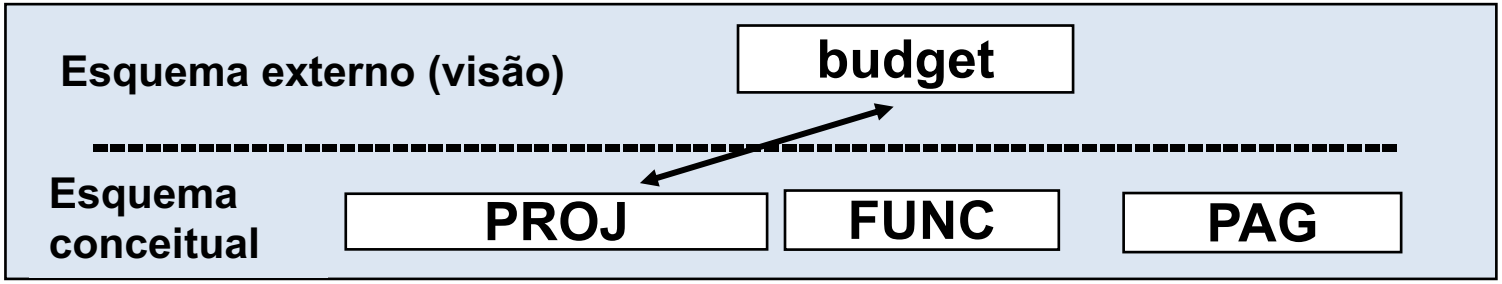
relation DESIG [
key = {FNO, PNO}
attributes = {
FNO : character(9)
PNO : character(7)
RESPONSAVEL: character(10)
DURAÇÃO : numeric(3) }]

exemplo do Esquema Interno

```
internal_rel FUNCL [
      index on FNO  call FMINX
      field = {
        CABEÇALHO   :  byte(1)
        FNO: byte(9)
        FNOME   : byte(15)
        CARGO   : byte(10)  }]
```

ansi/
sparc

Esquema Externo ou Visão (View)



Create view nomeView as <expressão da consulta>

definição da visão

- Exemplo 1 – relatório sobre orçamento de cada projeto

```
create view
    budget (NOME, ORCAMEN)
as    select PNOME, ORCAMEN
      from PROJ
```

utilização da visão

lista de todos os projetos com orçamento de 100mil

```
select nome as Projeto
from budget
where orcamen = 100000;
```

views - continuação

- Pode-se definir visões a partir de visões.
- Pode-se eliminar visões
  - **drop view budget**

- Operações sobre visões
  - normalmente só se faz busca sobre visões.
  - problemas em atualizações de visões:

problemas em atualizações de views

```
create view INFO
as select distinct rua, cidade
from clientes
```

CLIENTE

nome_cli	rua	cidade
João	azul	SP
Maria	amarela	RN
Ana	branca	RJ
José	amarela	RN

INFO

rua	cidade
azul	SP
amarela	RN
branca	RJ

Modificações em tabelas

CLIENTE

nome_cli	rua	cidade
João	azul	SP
Maria	amarela	RN
Ana	branca	RJ
José	amarela	RN
Bruno	verde	SP

Select *
From INFO;

rua	cidade
azul	SP
amarela	RN
branca	RJ
verde	SP

Toda atualização na tabela gera automaticamente a atualização nas views associadas

E modificações nas views?

Insert

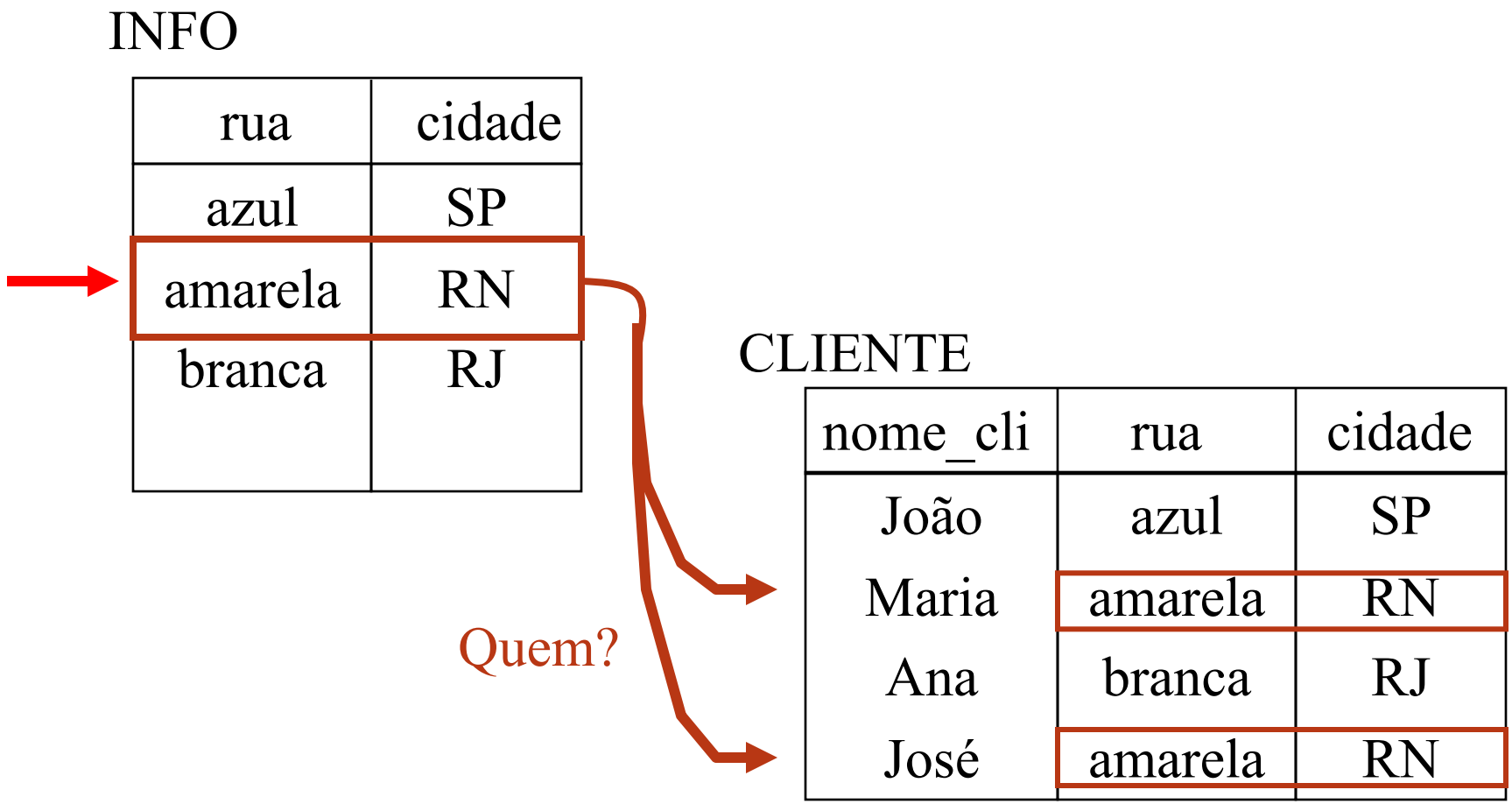
INFO

rua	cidade
azul	SP
amarela	RN
Branca	RJ
preto	RJ

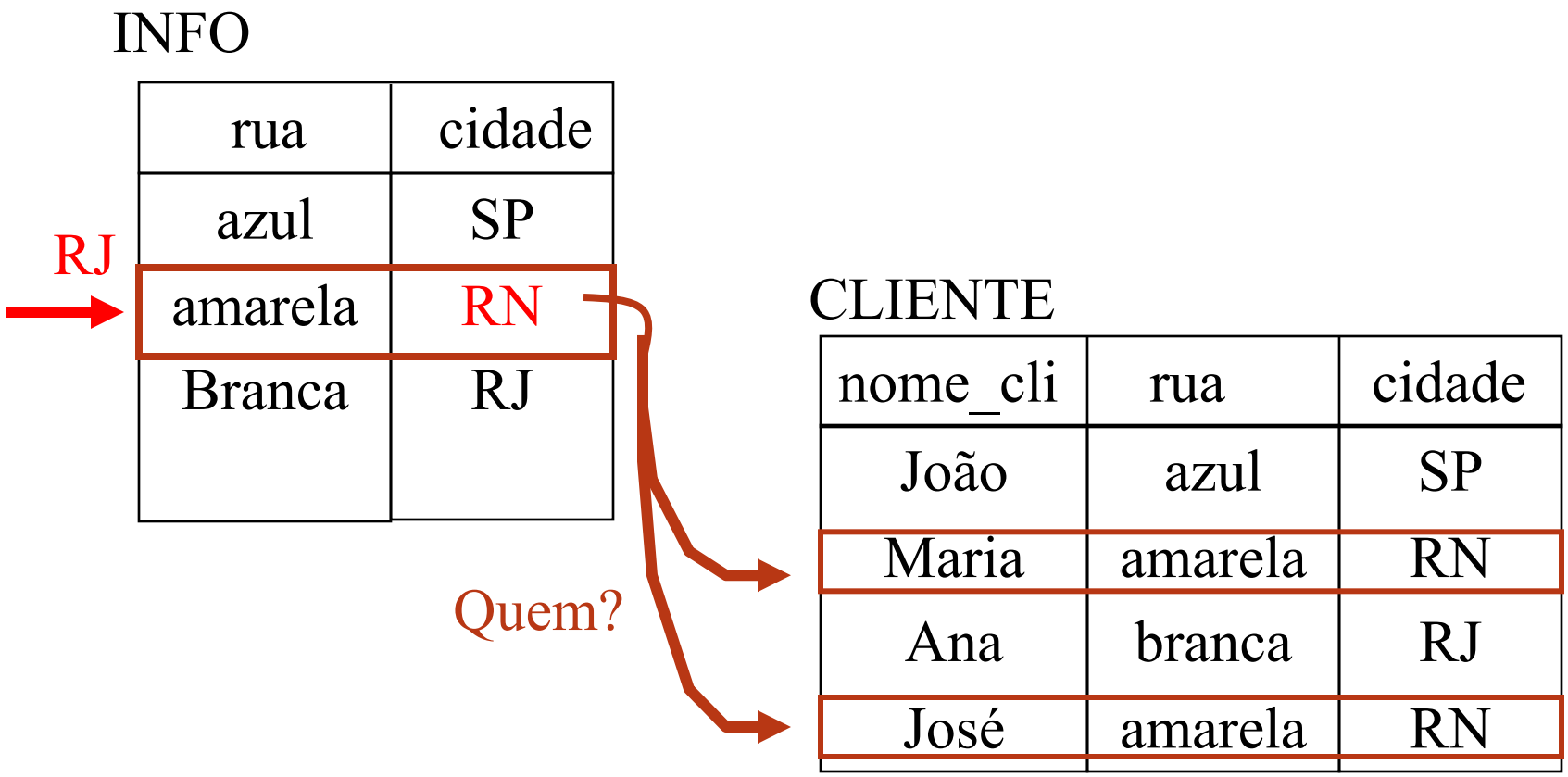
CLIENTE

nome_cli	rua	cidade
João	azul	SP
Maria	amarela	RN
Ana	branca	RJ
José	amarela	RN
?	preto	RJ

Delete


INFO
rua
cidade
azul
SP
amarela
RN
branca
RJ
Quem?
CLIENTE
nome_cli
rua
cidade
João
azul
SP
Maria
amarela
RN
Ana
branca
RJ
José
amarela
RN

Update



Pior caso: visão é uma combinação de mais de uma tabela

Proteção de Dados

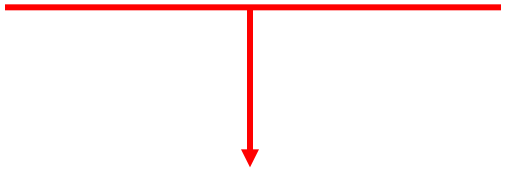
- usuários não autorizados não podem acessar o dado

- criptografia
  - dados residentes em disco
  - dados que trafegam em rede

Ref. Silberschatz, A.; Korth, H. Suddarshan, S. Sistemas de BD. – Tópico: Segurança

Autorização sobre operações

- usuários só podem **executar operações** que foram autorizadas.
  - usuários diferentes tem direitos diferentes sobre os mesmos objetos no banco de dados.
  - é preciso definir usuário ou grupos de usuários, objetos e direitos



nome do usuário e senha

Autorização sobre operações - continuação

- controle de autorização: (usuário, operação e objeto)

- tipo de objeto (relação, tupla, atributo, visão)

Direitos:

grant <tipo-de-operação> **on** <objeto> **to** <usuário>
revoke <tipo-de-operação> **from** <objeto> **to** <usuário>

Autorização sobre operações - continuação

- Exemplos:

postgre não permite a autorização por campos

```
grant select on conta to José, Maria;

grant update (saldo) on conta  to Maria;
```

- privilégios de usuários sobre objetos, registrados no catálogo ou dicionário de dados como regra de autorização

mais exemplos - continuação

- permite ao usuário Maria criar referências a campos de outras tabelas:

```
grant references (cargo) on PAG to Maria;
```

quando permite-se que um usuário crie referências a um campo, estamos modificando as permissões sobre a tabela referenciada – determinado usuário pode não mais conseguir remover um determinado cargo de PAG sem alterar também a tabela que contém a referencia.

mais exemplos - continuação

- public refere-se a todos os usuários (atuais e futuros) do sistema
  - grant select on PAG to public
  - grant all privileges on PAG to Maria

dá a Maria todos os privilégios sob a tabela PAG

  - grant select on PAG to Maria with grant option

dá a Maria privilégio para fornecer privilégio de select sob a tabela PAG para outro usuário

Autorização sobre alterações no esquema:

- criação de novas tabelas
- alteração de tabelas (atributos)
- eliminação de tabelas
- criação de índices
- eliminação de índices

SEQUÊNCIAS NO POSTGRESQL

Sequências

- Pode-se usar um gerador de sequências

CREATE [TEMPORARY | TEMP] SEQUENCE *nome*
[INCREMENT [BY] *incremento*]
[MINVALUE *minvalor* | NO MINVALUE]
[MAXVALUE *maxvalor* | NO MAXVALUE]
[START [WITH] *início*]
[CACHE *cache*]
[[NO] CYCLE]

OBS: o nome da sequência deve ser diferente do de outra sequência, tabela, view ou índice existentes.

- Parâmetros
  - **Temporary or Temp** – a sequência é criada somente para esta sessão e é automaticamente removido no fim da sessão.
  - **Nome** – o nome da sequência a ser criada.
  - **INCREMENT BY** *incremento*: especifica que o valor a ser adicionado ao valor corrente da sequência para criar um novo valor. Um valor positivo resultará numa sequência crescente e, um valor negativo numa sequência decrescente. A cláusula é opcional. O valor default é 1.
  - **MINVALUE *minvalor* | NO MINVALUE** : determina o valor mínimo que pode ser gerado para a sequência. Caso a cláusula seja omitida ou a opção NO MINVALUE seja especificada, então o valor default será usado. Os defaults são 1 e -2^{63} -1 para sequências crescente e decrescentes, respectivamente.
  - **MAXVALUE *maxvalor* | NO MAXVALUE**: similar ao anterior, só que determina o valor máximo. Na omissão da cláusula ou uso de NO MAXVALUE os valores default são usados. Os defaults são 2^{63} -1 e -1 para sequências crescente e decrescente, respectivamente.
  - **START [WITH] *início***: *permite que a sequência inicie em qualquer ponto. O valor default inicial é o minvalor para sequência crescente e maxvalor para sequência decrescente.*

- **CACHE *cache***: *especifica quantos números da sequência serão pré-alocados e armazenados em memória para acesso rápido. O valor mínimo é 1 (somente um valor pode ser gerado por vez, isto é, não cache), e este é o default.*

- **[NO] CYCLE**: permite recomeçar a sequência quando o valor máximo (maxvalor) ou o valor mínimo (minvalor) for atingido para uma sequência crescente ou decrescente respectivamente. Se o limite for atingido, o próximo número gerado será o minvalor ou o max valor, respectivamente.
  - Se NO CYCLE é especificado, qualquer chamada para o próximo valor (nextval) retornará um erro. Se nem CYCLE ou No CYCLE for especifiado, o default é NO CYCLE.

Manipulação da sequência

- nextval (text)
  - retorna o novo valor da sequência (avança a sequência). O tipo de dado retornado é do tipo bigint.

- Currval (text)
  - retorna o valor corrente da sequência (valor mais recente gerado com nextval).

  select currval (‘text')

- Verifique a sequência:

  Select * from nome_sequencia

exemplos

CREATE [TEMPORARY | TEMP] **SEQUENCE** ***nome***
[**INCREMENT** [BY] *incremento*]
[**MINVALUE** *minvalor* | NO MINVALUE]
[**MAXVALUE** *maxvalor* | NO MAXVALUE]
[**START** [WITH] *início*]
[CACHE *cache*]
[[NO] CYCLE]

1. Cria uma sequência de nome serial que inicia em 101

CREATE SEQUENCE serial START 101

2. Seleciona o próximo número da sequência:

   SELECT nextval ('serial');

3. Remove a sequência:

   Drop sequence nome