**Sequence** - it is database object from which multiple users may generate unique integers . You can use sequences to automatically generate primary key values.

When a sequence number is generated, the sequence is incremented ,independent of the transaction committing or rolling back.

Sequence numbers are generated independently of tables, so the same sequence can be used for one or for multiple tables.

It is independent object.

After a sequence is created, you can access its value in SQL statement with below pseudocolumn.

1.**CURRVAL** – which returns the current value

2.**NEXTVAL** – which increments the sequence and returns the new value.

|  |  |
| --- | --- |
| **parameters** | **Details** |
| Incremented by | Interval between the numbers |
| Start with | First number needed |
| maxvalue | Maximum value for the sequence |
| nomaxvalue | Maximum value is defaulted |
| minvalue | Minimum value for the sequence |
| nominvalue | Minimum value is defaulted |
| cycle | Reset to start after reaching this value |
| nocycle | default |
| cache | Preallocation limit |
| nocache | default |
| Order | Guarantee the order of numbers |
| noorder | Default |

**Example 1. With default values**

Create sequence seq1;

**Default values for**

Increment by =1

Cache=20

Nocyle

Minvalue=1

Maxvalue=9999999999999

**Example 2 :- with specifying values**

Create sequence customer\_Seq

Start with 1000

Increment by 1

Maxva 2000

Nocycle

Nocache;

Select seq1.currval from dual;

**Will throw error.**

**Seq1.currval Is not yet defined for this session.**

Below is the explanation

Nextval,currval and session

Create sequence

In session1, will generate seq1.nextval

In session 2, will generate seq1.nextval 1000 times

Check seq1.currval in session1

**Session1:**

Create sequence seq1 start with 1 increment by 1 nocycle;

**Session2:**

Declare

X number;

For I in 1..1000 loop

Select seq1.nextval into x from dual;

End loop;

End;

**Session1.**

Select seq1.nextval from dual;

1

**Currval is stored in the session pga/uga memory and not in the data dictionary.**

**Seq1.last\_number and sequence cache**

We will use user\_sequences.last\_number to get next value would have returned , only if the sequence is created with nocache

Lets create sequence without nocache .

Create sequence seq1 start with 1 increment by 1 nocycle;

Declare

X number;

For I in 1..1000 loop

Select seq1.nextval into x from dual;

End loop;

End;

Select seq1.nextval from dual;

1001

Select last\_number from user\_sequences where sequence\_name=’SEQ1’;

1021

Last\_number is 1021 because sequence was created with nocache option . the values in user\_sequences.last\_number jumps by the cache size and it is not the actual nextval.

**Same will create with nocache**

Create sequence seq1 start with 1 increment by 1 nocycle nocache;

Declare

X number;

For I in 1..1000 loop

Select seq1.nextval into x from dual;

End loop;

End;

Select seq1.nextval from dual;

1001

Select last\_number from user\_sequences where sequence\_name=’SEQ1’;

1002

user\_sequences.last\_number is retuning the next value of the sequence.

**We can create sequence with negative values as well. But both should be negative start with and increment by**

Create sequence seq1 start with -1000 increment by -1;

Select seq2.nextval from dual;

-1000

Select seq2.nextval from dual;

-1001

Create sequence seq1 start with 1000 increment by -1;

**Error start with cannot be more than maxvalue.**

**If we use sequence twice in same select it will return same values**

**Select seq1.nextval,seq1.nextval from dual;**

**1021 1021**

We cannot alter sequence with start with instead we can alter with increment by

Declare

L\_current\_value number;

L\_increment\_by number;

Begin

Execute immediate ‘select seq1.nextval from dual’ into l\_current\_value;

Select increment \_by into L\_increment\_by

From user\_sequences

Where sequence\_name=’SEQ1’;

L\_current\_value =2000;

If(L\_current\_value <>0) tehn

Execute immediate ‘alter sequence seq1 increment by ‘||L\_current\_value ||’ minvalue=0;

Execute immediate ‘select seq1.nextval from dual’ into l\_current\_value;

Execute immediate ‘alter sequence seq1 increment by ‘||L\_increment\_by ||’ minvalue=0;

End if;

End;

/

DD

User\_sequences;

All\_sequences;

**To drop sequence**

Drop sequence seq1;