MINVALUE *n* specifies the minimum sequence value
NOMINVALUE specifies a minimum value of 1 for an ascending sequence and – (10^26) for a descending sequence (This is the default option.)

CYCLE | NOCYCLE specifies whether the sequence continues to generate values after reaching its maximum or minimum value (NOCYCLE is the default option.)
CACHE *n* | NOCACHE specifies how many values the Oracle server preallocates and keep in memory (By default, the Oracle server caches 20 values.)

Creating a Sequence

- Create a sequence named DEPT_DEPTID_SEQ to be used for the primary key of the DEPARTMENTS table.
- Do not use the CYCLE option.

EXAMPLE:

```
CREATE SEQUENCE dept_deptid_seq INCREMENT BY 10 START WITH 120
MAXVALUE 9999 OCACHE NOCYCLE;
```

Confirming Sequences

- Verify your sequence values in the USER_SEQUENCES data dictionary table.
- The LAST_NUMBER column displays the next available sequence number if NOCACHE is specified.

EXAMPLE:

```
SELECT sequence_name, min_value, max_value, increment_by, last_number
```

NEXTVAL and CURRVAL Pseudocolumns

- NEXTVAL returns the next available sequence value. It returns a unique value every time it is referenced, even for different users.
- CURRVAL obtains the current sequence value.
- NEXTVAL must be issued for that sequence before CURRVAL contains a value.

Rules for Using NEXTVAL and CURRVAL

You can use NEXTVAL and CURRVAL in the following contexts:
- The SELECT list of a SELECT statement that is not part of a subquery
- The SELECT list of a subquery in an INSERT statement
- The VALUES clause of an INSERT statement
- The SET clause of an UPDATE statement

You cannot use NEXTVAL and CURRVAL in the following contexts:
- The SELECT list of a view
- A SELECT statement with the DISTINCT keyword
- A SELECT statement with GROUP BY, HAVING, or ORDER BY clauses
- A subquery in a SELECT, DELETE, or UPDATE statement
- The DEFAULT expression in a CREATE TABLE or ALTER TABLE statement

Using a Sequence

- Insert a new department named "Support" in location ID 2500.
- View the current value for the DEPT_DEPTID_SEQ sequence.

EXAMPLE:

```
INSERT INTO departments(department_id, department_name, location_id)
VALUES (dept_deptid_seq.NEXTVAL, 'Support', 2500);
```

SELECT dept_deptid_seq.CURRVAL FROM dual;
The example inserts a new department in the DEPARTMENTS table. It uses the DEPT_DEPTID_SEQ sequence for generating a new department number as follows:
You can view the current value of the sequence:
SELECT dept_deptid_seq.CURRVAL FROM dual;

Removing a Sequence

- Remove a sequence from the data dictionary by using the DROP SEQUENCE statement.

- Once removed, the sequence can no longer be referenced.

EXAMPLE:

DROP SEQUENCE dept_deptid_seq;

What is an Index?

An index:

- Is a schema object
- Is used by the Oracle server to speed up the retrieval of rows by using a pointer
- Can reduce disk I/O by using a rapid path access method to locate data quickly
- Is independent of the table it indexes
- Is used and maintained automatically by the Oracle server

How Are Indexes Created?

- Automatically: A unique index is created automatically when you define a PRIMARY KEY or UNIQUE constraint in a table definition.
- Manually: Users can create nonunique indexes on columns to speed up access to the rows.

Types of Indexes

Two types of indexes can be created. One type is a unique index: the Oracle server automatically creates this index when you define a column in a table to have a PRIMARY KEY or a UNIQUE key constraint. The name of the index is the name given to the constraint.
The other type of index is a nonunique index, which a user can create. For example, you can create a FOREIGN KEY column index for a join in a query to improve retrieval speed.

Creating an Index

- Create an index on one or more columns.
- Improve the speed of query access to the LAST_NAME column in the EMPLOYEES table.

CREATE INDEX *index* ON *table* (*column*[, *column*]...);

EXAMPLE:

CREATE INDEX emp_last_name_idx ON employees(last_name);

In the syntax:

index is the name of the index
table is the name of the table
column is the name of the column in the table to be indexed

When to Create an Index

You should create an index if:

- A column contains a wide range of values
- A column contains a large number of null values
- One or more columns are frequently used together in a WHERE clause or a join condition
- The table is large and most queries are expected to retrieve less than 2 to 4 percent of the rows

When Not to Create an Index

It is usually not worth creating an index if:

- The table is small
- The columns are not often used as a condition in the query
- Most queries are expected to retrieve more than 2 to 4 percent of the rows in the table • The table is updated frequently
- The indexed columns are referenced as part of an Expression

Confirming Indexes

- The USER_INDEXES data dictionary view contains the name of the index and its uniqueness.
- The USER_IND_COLUMNS view contains the index name, the table name, and the column name.

EXAMPLE:

SELECT ic.index_name, ic.column_name, ic.column_position col_pos,ix.uniqueness FROM user_indexes ix, user_ind_columns ic WHERE ic.index_name = x.index_name AND ic.table_name = 'EMPLOYEES';

Removing an Index

- Remove an index from the data dictionary by using the DROP INDEX command.
- Remove the UPPER_LAST_NAME_IDX index from the data dictionary.
- To drop an index, you must be the owner of the index or have the DROP ANY INDEX privilege.

DROP INDEX upper_last_name_idx;
DROP INDEX *index*;

Find the Solution for the following:

1. Create a sequence to be used with the primary key column of the DEPT table. The sequence should start at 200 and have a maximum value of 1000. Have your sequence increment by ten numbers. Name the sequence DEPT_ID_SEQ.

   Ans: CREATE SEQUENCE dept_id_seq INCREMENT BY 10 START WITH 200 MAXVALUE 1000 NOCYCLE NOCACHE;

2. Write a query in a script to display the following information about your sequences: sequence name, maximum value, increment size, and last number

Ans:SELECT sequence_name,max_value,increment_by,last_number FROM user_sequences WHERE sequence_name='DEPT_ID_SEQ';

3. Write a script to insert two rows into the DEPT table. Name your script lab12_3.sql. Be sure to use the sequence that you created for the ID column. Add two departments named Education and Administration. Confirm your additions. Run the commands in your script.

Ans: INSERT INTO dept(dept_id,dept_name) VALUES(dept_id_seq.NEXTVAL,'Education');
INSERT INTO dept(dept_id,dept_name) VALUES(dept_id_seq.NEXTVAL,'Administration');
SELECT * FROM dept;

4. Create a nonunique index on the foreign key column (DEPT_ID) in the EMP table.

   Ans: CREATE INDEX emp_dept_id_idx ON emp(dept_id);

5. Display the indexes and uniqueness that exist in the data dictionary for the EMP table.

   Ans: SELECT ic.index_name, ic.column_name, ic.column_position AS col_pos, ix.uniqueness FROM user_indexes ix JOIN user_ind_columns ic ON ix.index_name = ic.index_name WHERE ic.table_name = 'EMP';