## Experiment No. : 7

**Title:** Write a program to create Indices and Views in SQL.

## Objectives:

1. To learn about indices.
2. To learn about views.
3. To learn how to create indices in sql.
4. To learn how to create views in sql.

**Key Concepts:** indices, views

## Theory:

### Index

Indices are one of the most powerful aspects of SQL performance. Database tables can become very big. Scanning through millions, billions or trillions of rows to return just two or three is a huge waste. Indices can help us to avoid this situation. Indicees are used to retrieve data from the database very fast. The users cannot see the indexes, they are just used to speed up searches/queries.

An index stores the values in the indexed column(s) and for each value the locations of the rows that have it. Just like the index in a book. This enables us to hone in on just the data that we are interested in. Indices are very effective when they enable us to find a "few" rows. So if we have a query that fetches a handful of rows, there is a good chance to use an index.

### Create an Index

While creating an index, all we need to do is identify which column(s) we want to index and give it a name.

**Syntax:**

create index <index_name> on <table_name> ( <column1>[ASC | DESC], … );

We can create an index on a combination of columns.

**Example:** CREATE INDEX idx_pname ON Persons (LastName, FirstName);

The maximum number of columns for an index key is 16. An index name cannot exceed 128 characters. A column must not be named more than once in a single CREATE INDEX statement. Different indexes can name the same column, however. Index names need to be unique within a schema.

Index can be created while creating table.

**Example:**

CREATE TABLE Student (PRN NUMBER PRIMARY KEY USING INDEX (create index indx_PRN on Student(PRN)));

## Drop an Index

The DROP INDEX statement is used to remove an index.

**Syntax:**

DROP INDEX *index_name*;

## View

Views are virtual tables formed by a query that does not physically exist. A view is a dictionary object that we can use until we drop it. Views are not updatable. They represent the data of one of more tables. A view derives its data from the tables on which it is based. These tables are called base tables. Views can be based on actual tables or another view also. A view operates with the privileges of the owner of the view. The view owner automatically gains the SELECT privilege on the view. The SELECT privilege cannot be revoked from the view owner. The database owner automatically gains the SELECT privilege on the view and is able to grant this privilege to other users. The SELECT privilege cannot be revoked from the database owner.

The view owner can only grant the SELECT privilege to other users if the view owner also owns the underlying objects. If the underlying objects that the view references are not owned by the view owner, the view owner must be granted the appropriate privileges.

For example, if the authorization ID user2attempts to create a view called user2.v2 that references table user1.t1 and function user1.f_abs(), then user2 must have the SELECT privilege on table user1.t1 and the EXECUTE privilege on function user1.f_abs().

The privilege to grant the SELECT privilege cannot be revoked. If a required privilege on one of the underlying objects that the view references is revoked, then the view is dropped.

We can Query, Insert, Update and delete from views, just as any other table.

## Create View

CREATE VIEW command is used to create view.

**Syntax:**

**CREATE VIEW** view-Name **[ (** Simple-column-Name **[,** Simple-column-Name**] * ) ]**
**AS** Query **[** ORDER BY clause **]**
**[** result offset clause **]**
**[** fetch first clause **]**

**Example:**

```
CREATE OR REPLACE VIEW ST_ATT_ADDR(PRN, SNAME, SADDRESS,
ATTENDANCE)
AS SELECT O.PRN, O.SNAME, OS.SADDRESS,D.ATTENDANCE
FROM ST_DEPT D, ST_OFFICE O
    WHERE O.PRN=D.PRN
```

We can modify the definition of a SQL VIEW without dropping it by using the SQL CREATE OR REPLACE VIEW Statement.

## Insertion of record in a view

If the view is created on single table. Data inserted in view automatically reflects in base table.

## Syntax:

INSERT INTO <view name> VALUES (<Value for Column1>, <Value for Column 2>,…);

## Deletion of record from a view

If the view is created on single table. Data deleted from a view automatically reflects in base table.

## Syntax:

DELETE FROM <view_name> WHERE <condition>

## Updation of record in a view

If the view is created on single table. Data updated in a view automatically reflects in base table.

## Syntax:

UPDATE <view_name>

SET <column1>=<value1>, <column2>=<value2>,...

WHERE <condition>

If an index is created on more than one table. Insert, delete update operations on view are allowed to perform.

## Drop View

The DROP VIEW statement is used to remove view.

**Syntax:**

DROP VIEW *view_name*;

## Algorithm:

1. Start
2. Create tables by taking field information from user
3. Insert data into above created tables.
4. Write SQL query to create indices.
5. Write SQL query to demonstrate data retrieval.
6. Write SQL query to drop index.
7. Write SQL query to create view.
8. Write SQL query to demonstrate data retrieval in view.
9. Write SQL query to demonstrate data insertion in view.
10. Write SQL query to drop view.
11. Stop.