POSTGRESQL - VIEWS

Views are pseudo-tables. That is, they are not real tables, but nevertheless appear as ordinary tables to SELECT. A view can represent a subset of a real table, selecting certain columns or certain rows from an ordinary table. A view can even represent joined tables. Because views are assigned separate permissions, you can use them to restrict table access so that users see only specific rows or columns of a table.

A view can contain all rows of a table or selected rows from one or more tables. A view can be created from one or many tables which depends on the written PostgreSQL query to create a view.

Views, which are kind of virtual tables, allow users to do the following:

- Structure data in a way that users or classes of users find natural or intuitive.
- Restrict access to the data such that a user can only see limited data instead of complete table.
- Summarize data from various tables which can be used to generate reports.

Because views are not ordinary tables, so you may not execute a DELETE, INSERT, or UPDATE statement on a view. But you can create a RULE to correct this problem of using DELETE, INSERT or UPDATE on a view.

Creating Views:

The PostgreSQL views are created using the **CREATE VIEW** statement. The PostgreSQL views can be created from a single table, multiple tables, or another view.

The basic CREATE VIEW syntax is as follows:

```
CREATE [TEMP | TEMPORARY] VIEW view_name AS
SELECT column1, column2.....
FROM table_name
WHERE [condition];
```

You can include multiple tables in your SELECT statement in very similar way as you use them in normal PostgreSQL SELECT query. If the optional TEMP or TEMPORARY keyword is present, the view will be created in the temporary space. Temporary views are automatically dropped at the end of the current session.

Example:

Consider COMPANY table is having the following records:

```
 id | name  | age | address    | salary
----+-------+-----+------------+--------
  1 | Paul  |  32 | California |  20000
  2 | Allen |  25 | Texas      |  15000
  3 | Teddy |  23 | Norway     |  20000
  4 | Mark  |  25 | Rich-Mond  |  65000
  5 | David |  27 | Texas      |  85000
  6 | Kim   |  22 | South-Hall |  45000
  7 | James |  24 | Houston    |  10000
```

Now, following is an example to create a view from COMPANY table. This view would be used to have only few columns from COMPANY table:

```
testdb=# CREATE VIEW COMPANY_VIEW AS
SELECT ID, NAME, AGE
FROM  COMPANY;
```

Now, you can query COMPANY_VIEW in similar way as you query an actual table. Following is the example:

```
testdb=# SELECT * FROM COMPANY_VIEW;
```

This would produce the following result:

```
 id | name  | age
----+-------+-----
  1 | Paul  |  32
  2 | Allen |  25
  3 | Teddy |  23
  4 | Mark  |  25
  5 | David |  27
  6 | Kim   |  22
  7 | James |  24
(7 rows)
```

Dropping Views:

To drop a view, simply use the DROP VIEW statement with the **view_name**. The basic DROP VIEW syntax is as follows:

```
testdb=# DROP VIEW view_name;
```

Following command will delete COMPANY_VIEW view, which we created in last section:

```
testdb=# DROP VIEW COMPANY_VIEW;
```