K. J. Somaiya College of Engineering, Mumbai-77
(Autonomous College Affiliated to University of Mumbai)

Batch: A1
Roll No.: 16010120006
16010120015
16010120018

Experiment / assignment / tutorial No. 5

Grade: AA / AB / BB / BC / CC / CD /DD

Title: Queries based on Views and Triggers

Objective: To be able to use SQL JOIN clause to extract data from 2 (or more) tables, we need a relationship between certain columns in these tables.

Expected Outcome of Experiment:
CO 3 : Use SQL for Relational database creation, maintenance and query processing

Books/ Journals/ Websites referred:

1. Dr. P.S. Deshpande, SQL and PL/SQL for Oracle 10g.Black book, Dreamtech Press
2. www.db-book.com
3. Korth, Slberchatz, Sudarshan : "Database Systems Concept", 5th Edition , McGraw Hill
4. Elmasri and Navathe,"Fundamentals of database Systems", 4th Edition,PEARSON Education.

Resources used: Postgresql

Theory

Views are pseudo-tables. That is, they are not real tables; 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 the 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.

Since views are not ordinary tables, you may not be able to execute a DELETE, INSERT, or UPDATE statement on a view. However, you can create a RULE to correct this problem of using DELETE, INSERT or UPDATE on a view.

Syntax

CREATE [TEMP | TEMPORARY] VIEW view_name AS

SELECT column1, column2.....

FROM table_name

WHERE [condition];

Ex

CREATE VIEW COMPANY_VIEW AS

SELECT ID, NAME, AGE

FROM COMPANY;

Dropping Views

Syntax: DROP VIEW view_name;

Triggers are database call-back functions, which are automatically performed/invoked when a specified database event occurs.

Triggers can be specified to fire

- Before the operation is attempted on a row (before constraints are checked and the INSERT, UPDATE or DELETE is attempted)
- After the operation has completed (after constraints are checked and the INSERT, UPDATE, or DELETE has completed)
- Instead of the operation (in the case of inserts, updates or deletes on a view)

The basic syntax of creating a trigger is as follows −

```
CREATE  TRIGGER trigger_name [BEFORE|AFTER|INSTEAD OF] event_name
ON table_name
[
 -- Trigger logic goes here....
];
```

event_name could be INSERT, DELETE, UPDATE, and TRUNCATE database operation on the mentioned table table_name. You can optionally specify FOR EACH ROW after table name.

The following is the syntax of creating a trigger on an UPDATE operation on one or more specified columns of a table as follows −

```
CREATE  TRIGGER trigger_name [BEFORE|AFTER] UPDATE OF column_name
ON table_name
[
 -- Trigger logic goes here....
];
```

Implementation Screenshots (Problem Statement, Query and Screenshots of Results):

```
CREATE VIEW viewitem AS
SELECT *
FROM items
WHERE iquaty>210;

SELECT*
```

FROM viewitem;

```
186
187  // VIEWS //
188
189  CREATE VIEW viewitem AS
190  SELECT *
191  FROM items
192  WHERE iquaty>210;
193
194  SELECT*
195  FROM viewitem;
196
197
198
199
```

Data Output | Explain | Messages | Notifications

	ideptname character (25)	iname character varying (25)	idno integer	ilocation character (25)	irnum integer	iquaty integer
1	Technology	AL ROBOTICS	17	BIHAR	19	240
2	LIQUOR	ROYAL STAG	17	DELHI	16	240
3	Supplies	POTATO	17	BIHAR	14	220
4	Technology	RADIO	17	SURAT	13	500

INSERT INTO viewitem(ideptname, iname, idno, ilocation, irnum, iquaty)

VALUES('Supplies','OATS','106','SURAT','21','20');

SELECT *

FROM viewitem;

```
199  INSERT INTO viewitem(ideptname, iname, idno, ilocation, irnum, iquaty)
200  VALUES('Supplies','OATS','106','SURAT','21','20');
201
202  SELECT *
203  FROM viewitem;
204
```

Data Output | Explain | Messages | Notifications

	ideptname character (25)	iname character varying (25)	idno integer	ilocation character (25)	irnum integer	iquaty integer
1	Technology	AL ROBOTICS	17	BIHAR	19	240
2	LIQUOR	ROYAL STAG	17	DELHI	16	240
3	Supplies	POTATO	17	BIHAR	14	220
4	Technology	RADIO	17	SURAT	13	500

UPDATE viewitem

SET iquaty=350

WHERE iname='POTATO';

SELECT * FROM viewitem;

```
206  UPDATE viewitem
207  SET iquaty=350
208  WHERE iname='POTATO';
209
210  SELECT *
211  FROM viewitem;
212
```

Data Output | Explain | Messages | Notifications

	ideptname character (25)	iname character varying (25)	idno integer	ilocation character (25)	irnum integer	iquaty integer
1	Technology	AL ROBOTICS	17	BIHAR	19	240
2	LIQUOR	ROYAL STAG	17	DELHI	16	240
3	Technology	RADIO	17	SURAT	13	500
4	Supplies	POTATO	17	BIHAR	14	350

DROP VIEW viewitem;

```
19  SELECT *
20  FROM viewitem;
21
22
23  DROP VIEW viewitem;
```

Messages

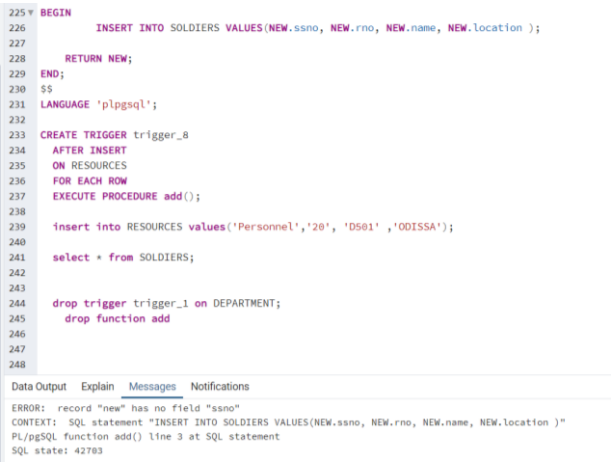
```
DROP VIEW

Query returned successfully in 219 msec.
```

TRIGGERS:-

1. After insert:-

```
CREATE OR REPLACE FUNCTION add()
  RETURNS trigger AS
$$
BEGIN
        INSERT INTO SOLDIERS VALUES(NEW.ssno, NEW.rno, NEW.name, NEW.location
);

    RETURN NEW;
END;
$$
LANGUAGE 'plpgsql';

CREATE TRIGGER trigger_8
  AFTER INSERT
  ON RESOURCES
  FOR EACH ROW
  EXECUTE PROCEDURE add();

  insert into RESOURCES values('Personnel','20', 'D501' ,'ODISSA');

  select * from SOLDIERS;


  drop trigger trigger_1 on DEPARTMENT;
          drop function add
```

Output:-



2. Before Insert:-

```
CREATE OR REPLACE FUNCTION age_constraint()
  RETURNS trigger AS
$$
BEGIN
   IF (NEW.ssno< 0) THEN
       RAISE EXCEPTION 'No negative age allowed';
    END IF;
    RETURN NEW;
END;
$$
LANGUAGE 'plpgsql';

          -- 2. create trigger

CREATE TRIGGER trigger1
  BEFORE INSERT
  ON SOLDIERS
  FOR EACH ROW
  EXECUTE PROCEDURE age_constraint();


  insert into SOLDIERS values(-613, 23, 'NIRMA SING', '1944/02/02', 'MAJOR', 'ORISSA',
100000) ;

-- drop TRIGGER
  drop trigger trigger1 on SOLDIERS
-- drop function
  drop function age_constraint
```

Output:-

```
258     RETURN NEW;
259 END;
260 $$
261 LANGUAGE 'plpgsql';
262
263     -- 2. create trigger
264
265 CREATE TRIGGER trigger1
266   BEFORE INSERT
267   ON SOLDIERS
268   FOR EACH ROW
269   EXECUTE PROCEDURE age_constraint();
270
271   insert into SOLDIERS values(610, 21, 'OMKAR SINGH', '1947/04/03', 'COLONEL', 'DELHI', 120000);
272
273   insert into SOLDIERS values(-613, 23, 'NIRMA SING', '1944/02/02', 'MAJOR', 'ORISSA', 100000) ;
274
275 -- drop TRIGGER
276   drop trigger trigger1 on SOLDIERS
277 -- drop function
278   drop function age_constraint
279
280
281
```

Data Output | Explain | Messages | Notifications

```
ERROR:  No negative age allowed
CONTEXT:  PL/pgSQL function age_constraint() line 4 at RAISE
SQL state: P0001
```

3. Updating values:-

```
CREATE OR REPLACE FUNCTION update_data()
 RETURNS trigger AS
$$
BEGIN
UPDATE ITEMS
      SET NEW.iquaty = iquaty + 1
           WHERE QUANTY = 23;

  RETURN NEW;
END;
$$
```

LANGUAGE 'plpgsql';

CREATE TRIGGER update_data_trigger_2

before INSERT

ON ITEMS

FOR EACH ROW

EXECUTE PROCEDURE update_data();

insert into ITEMS values('Personnel Equipments ','BIKE','17','BIHAR','18','23');

select * from ITEMS

Output:-

```
286  UPDATE ITEMS
287          SET NEW.iquaty = iquaty + 1
288                  WHERE QUANTY = 23;
289
290      RETURN NEW;
291  END;
292  $$
293  LANGUAGE 'plpgsql';
294
295  CREATE TRIGGER update_data_trigger_2
296    before INSERT
297    ON ITEMS
298    FOR EACH ROW
299    EXECUTE PROCEDURE update_data();
300
301      insert into ITEMS values('Personnel Equipments ','BIKE','17','BIHAR','18','23');
302
303      select * from ITEMS
304
305
```

Data Output | Explain | Messages | Notifications

	ideptname character (25)	iname character varying (25)	idno integer	ilocation character (25)	irnum [PK] integer	iquaty integer
1	LIQUOR	SCOTCH	104	DELHI	17	210
2	Supplies	WHEAT	106	SURAT	20	20
3	Technology	AL ROBOTICS	17	BIHAR	19	240
4	Personnel Equipments	BIKE	17	BIHAR	18	24
5	LIQUOR	ROYAL STAG	17	DELHI	16	240

Conclusion: Hence, we learnt to implement views and triggers queries.

Post Lab Questions:

1. **What is a view?**

a) A view is a special stored procedure executed when certain event occurs
b) A view is a virtual table which results of executing a pre-compiled query
c) A view is a database diagram
d) None of the Mentioned

Ans. b) A view is a virtual table which results of executing a pre-compiled query

2. Trigger is special type of _____ procedure.

a) Stored
b) Function
c) View
d) Table

Ans. Stored

3. Triggers can be enabled or disabled with the _____ statement.

a) ALTER TABLE statement
b) DROP TABLE statement
c) DELETE TABLE statement
d) None of the mentioned

Ans. ALTER TABLE statement

4. Visit following virtual lab link, read theory and procedure provided and solve pretest and post test questions. Support your answers with screenshots.

Link: http://vlabs.iitb.ac.in/vlabs-dev/labs/dblab/labs/exp2/index.php

Pre Test

1. Update statement belongs to following category of statements
   - (●) DML Statements
   - ( ) DDL Statements
   - ( ) TCL Statements
   - ( ) All of the above

2. The correct syntax for Delete statement is :
   - ( ) DELETE table <tablename >
   - ( ) DELETE FROM <table_name> WHERE some_condition
   - ( ) DELETE FROM <table_name> <colname> WHERE some_condition
   - (●) DELETE <table_name> WHERE some_condition

3. TThe correct syntax for Update statement is :
   - (●) UPDATE table_name SET column1 = value1, column2 = value2,... WHERE condition
   - ( ) UPDATE table table_name SET column1 = value1, column2 = value2,... WHERE condition
   - ( ) UPDATE from table_name SET column1 = value1, column2 = value2,... WHERE condition
   - ( ) None of the above

4. If <where> clause is omitted with Delete , then
   - (●) All the records are Deleted
   - ( ) Entire table is removed
   - ( ) Error is generated
   - ( ) Only 1 record is deleted

Reset Submit

Post Test

1. Which of the following is a DML statement?
   - ○ COMMIT
   - ○ CREATE
   - ◉ INSERT
   - ○ SELECT

2. What are the two different methods of inserting a new row in a table?
   - ◉ Only Values and Selected Column Insert
   - ○ Only Insert and Selected Column Values
   - ○ Only Insert and Only Selected Insert
   - ○ None of the Above

3. Which of the following option is correct for the query expression below?
   UPDATE student _____ Name = 'Rohan' _____ Roll_Number = '1';
   - ○ ASSIGN,WHERE
   - ◉ SET,WHERE
   - ○ WHERE,SET
   - ○ LIKE,AND

4. Which of the following query is correct for inserting a row into the table?
   - ○ INSERT INTO student VALUE('1','Rohan','1-3-1997');
   - ○ INSERT student VALUE('1','Rohan','1-3-1997');
   - ○ INSERT IN student VALUES('1','Rohan','1-3-1997');
   - ◉ INSERT INTO student VALUES('1','Rohan','1-3-1997');

5. What does the 'DELETE' command do when 'WHERE' clause is not used along with it?
   - ○ Some rows are deleted
   - ◉ All rows are deleted
   - ○ Some columns are deleted
   - ○ No data is deleted

6. How many numbers of columns can be updated using the update statement?
   - ○ Multiple
   - ◉ Single as well as multiple
   - ○ Single
   - ○ Only two

7. In case not all values of the table are being described in the 'INSERT' query, then what is mandatory to follow?
   - ◉ Must indicate both the column name and its corresponding value
   - ○ Must indicate all the column names and required corresponding values
   - ○ Must indicate required column names and all column values
   - ○ All of the Above