

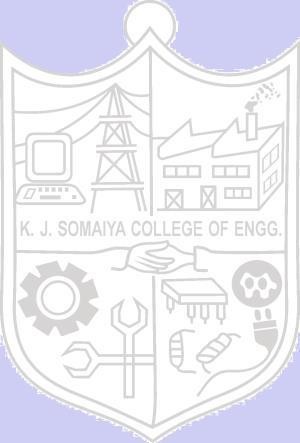
**Experiment No : 06**

**Title:** To create nested queries and view for the given Database.

# Batch:A2 Roll No.:16010421062 Experiment No: 06

**Aim:** To create nested queries and view for the given database (Virtual lab).

**Resources needed:** PostgreSQL PgAdmin3



**Theory:**

**Nested subqueries: in clause:**

The in connective tests for the set membership, where the set is a collection of values produced by a select clause.

For example to select details of the books written by r.p.jain and d.perry use

select book\_id, book\_name,price from book where author in(„r.p.jain‟, „d. perry‟,‟godse‟);

# not in:

This connective tests for absence of the set membership.

For example to select details of the books written by authors other than r.p.jain and d.perry use

select book\_id, book\_name,price from book where author not in(„r.p.jain‟, „d. perry‟,‟godse‟);

# all:

this keyword is basically used in set comparison query. It is used in association with relational operators.

“> all” corresponds to the phrase „greater than all‟.

For example to display details of the book that have price greater than all the books published in year 2000 use.

Select book\_id, book\_name, price from book where price >all (select price from book where pub\_year=‟2000‟);

# any or some:

These keywords are used with relational operators in where clause of set comparison query. “=some” is identical to in and “<>some” is identical to not in.

“>any “ is nothing but „greater than at least one‟.

# exists and not exists:

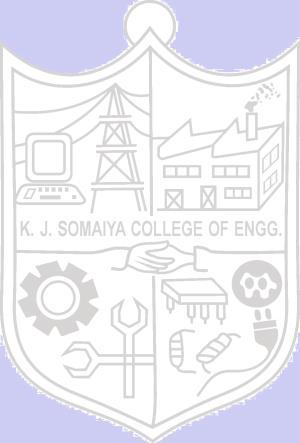
exists is the test for non empty set. It is represented by an expression of the form ‘exists (select ……. From …….) ‘. Such expression evaluates to true only if the result evaluvating the subquery represented by the (select ……. From ……) is non empty.

for example to select names of the books for which order is placed use

select book\_name from book where exists( select \* from order where book\_id=order.book\_id);

(Autonomous College Affiliated to University of Mumbai)

**Views:**



Views are virtual tables created from already existing tables by selecting certain columns or certain rows. A view can be created from one or many tables. View allows to,

* 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.

In PostgrSQL, Views are created using the CREATE VIEW statement given bellow.

CREATE [TEMP | TEMPORARY] VIEW view\_name AS SELECT column1, column2..... FROM table\_name WHERE [condition];

For example,

Consider COMPANY table having 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

Following statement creates a view from COMPANY table.

CREATE VIEW COMPANY\_VIEW AS SELECT ID, NAME, AGE FROM COMPANY;

Now, query can be written on COMPANY\_VIEW in similar way as that of an actual table, as shown below,

SELECT \* FROM COMPANY\_VIEW;

This would produce the following result:

View can be dropped using “DROP VIEW” statement.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Procedure / Approach /Algorithm / Activity Diagram:**

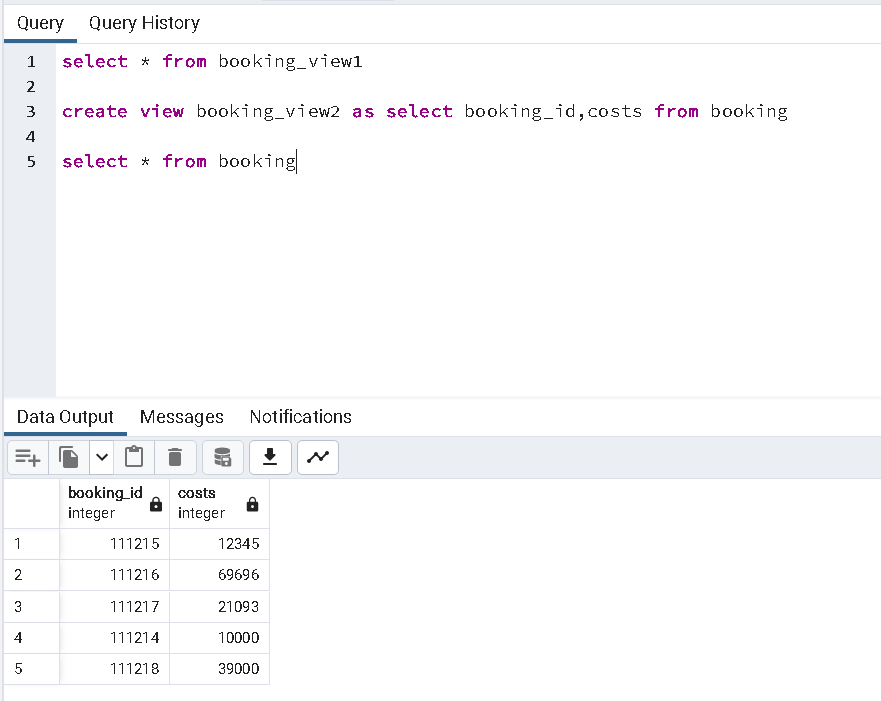
* 1. Refer different syntax given in theory section and formulate queries consisting of nested sub queries, in , not in, as, group by, having etc clauses and different set operations for your database.
  2. Create views from existing tables

Execute SELECT,UPDATE,INSERT statements on views and original table.

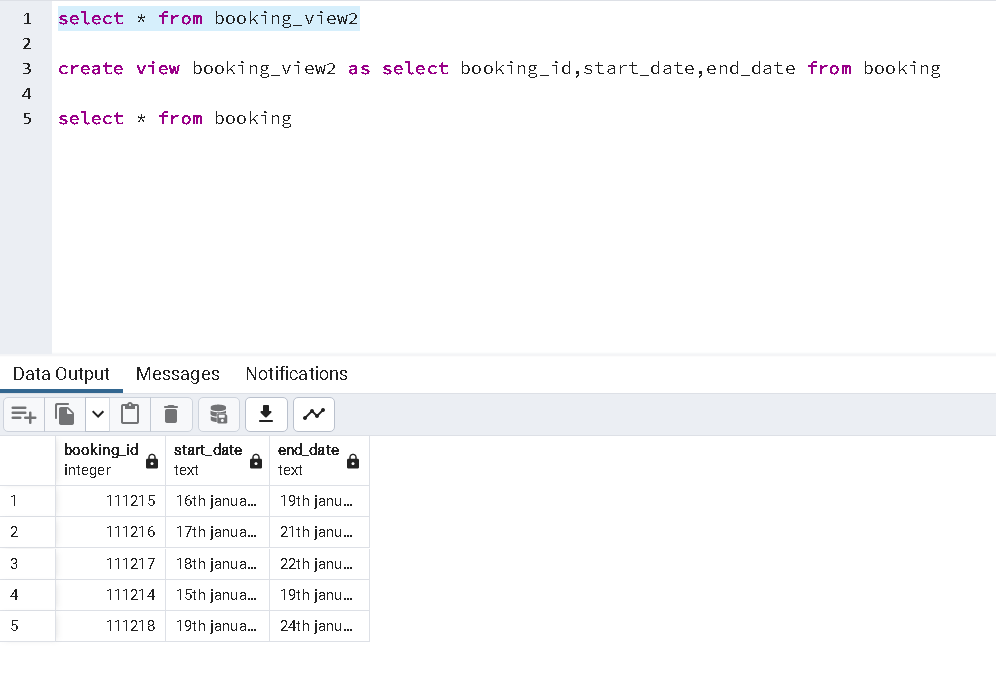
**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Results: (Program printout with output / Document printout as per the format)**

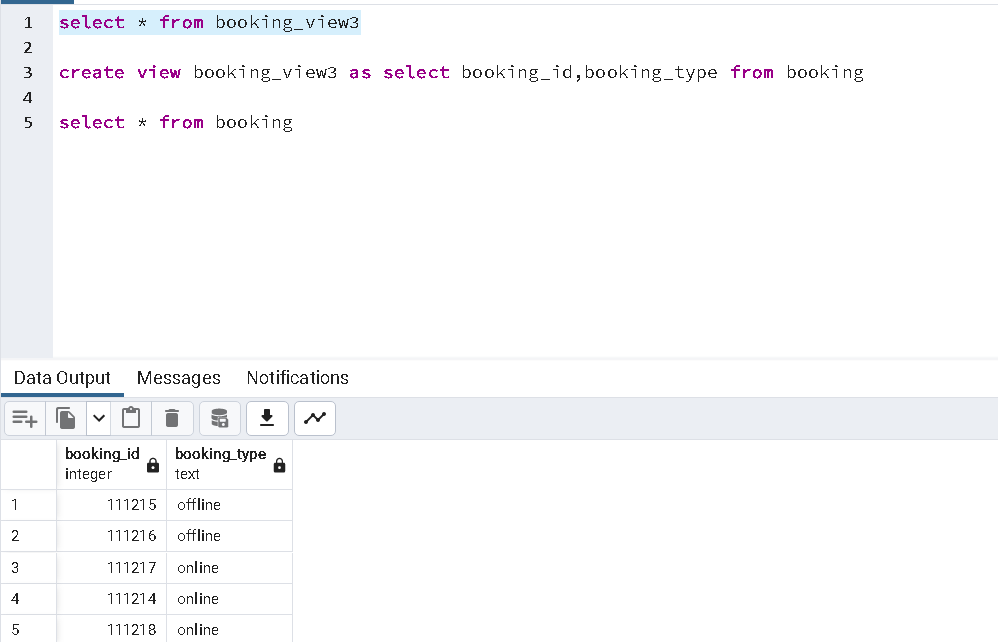
**1st view:**

****

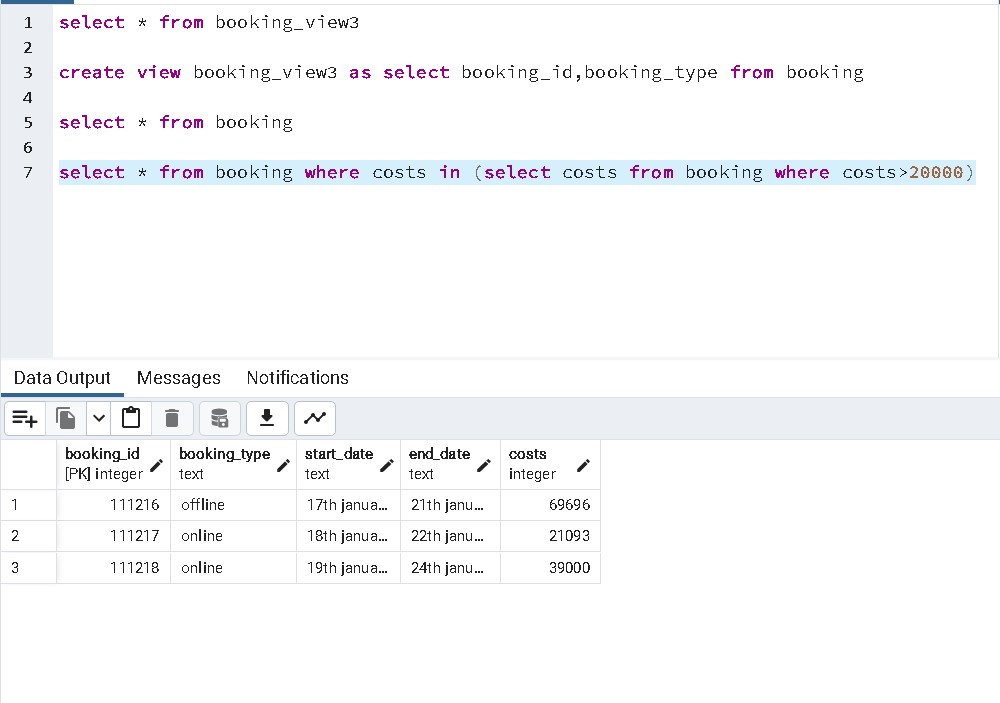
**2nd view:**

****

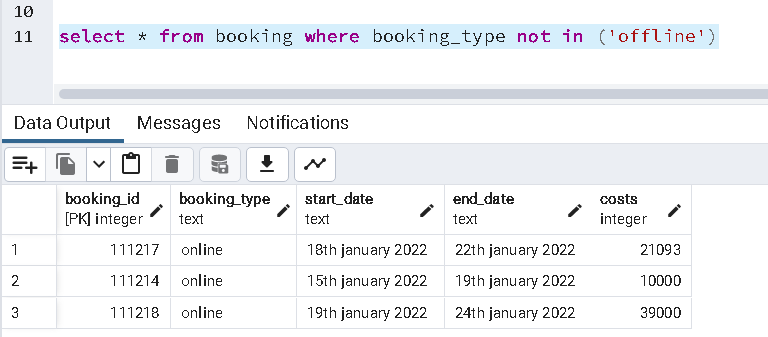
**3rd view:**

****

**Sub Query of in:**

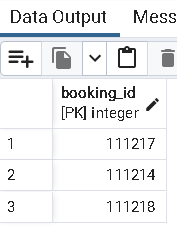
****

**Sub query of not in:**

****

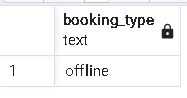
**Subquery of all:**

****

****

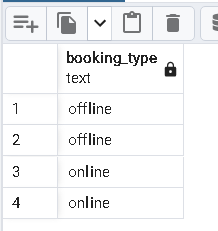
**Sub query of any:**

****

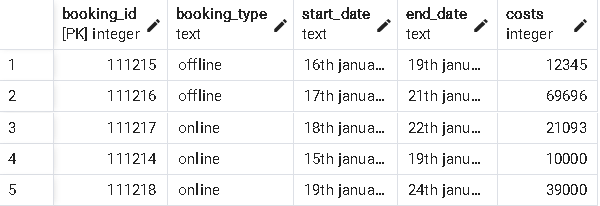
****

**Sub query of some:**

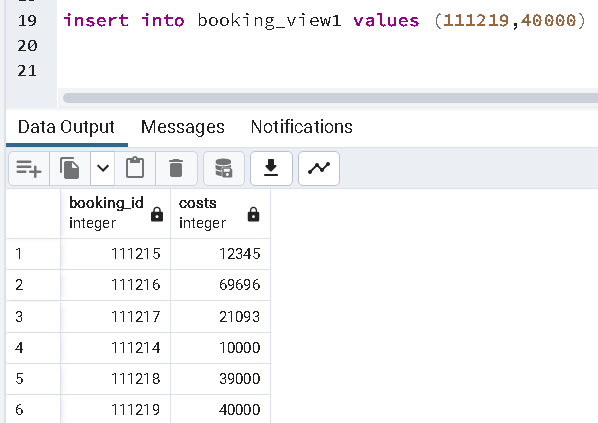
****

****

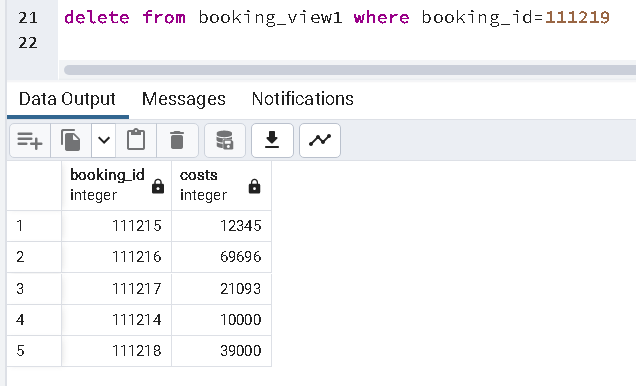
**The original table**

****

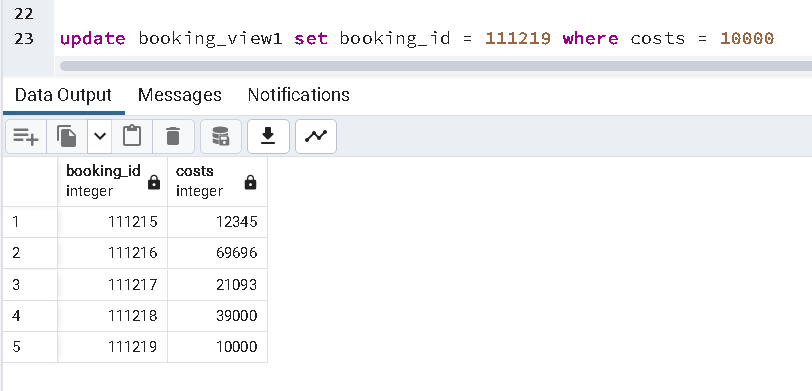
**USING INSERT STATEMENT IN VIEW:**

****

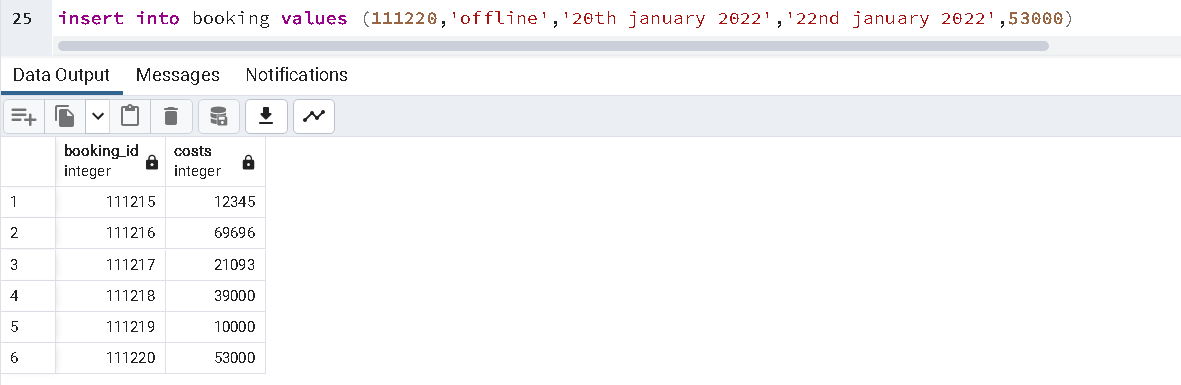
**USING DELETE STATEMENT IN VIEW:**

****

**Update statement in view:**

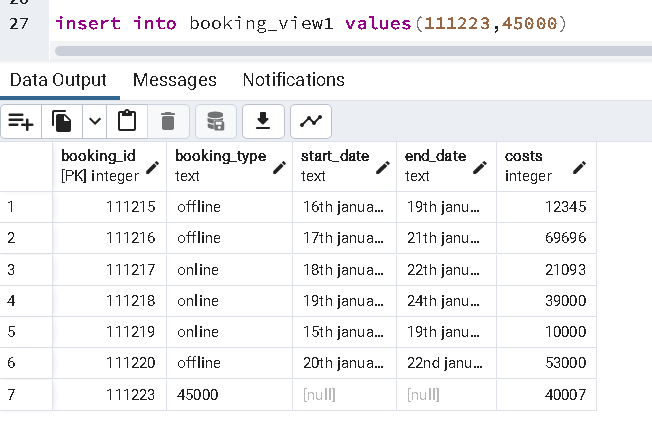
****

**Inserting something in the main table:**

****

**It shows the impact on the view table as well!**

**Inserting values in the view table :**

****

**It shows the impact in the main table as well!**

**But some values are null as they were not specified in the view table.**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Questions:**

* 1. Explain what are the disadvantages of using the view on update function.

Ans:

1. When a table is dropped, associated view become irrelevant.

2. Since the view is created when a query

requesting data from view is triggered,

its a bit slow.

3. When views are created for large tables,

it occupies more memory.

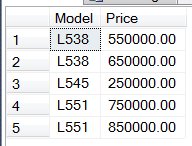
1. Can we use the where clause with the group by clause? Justify your answer

Where clause works with select clause but won’t work on the group by or aggregate function condition.

**Example 1**

1. **Select** Model, Price **from** VehicleProduction
2. **where** Model != 'L530'
3. **group** **by** Model, Price

**Output**

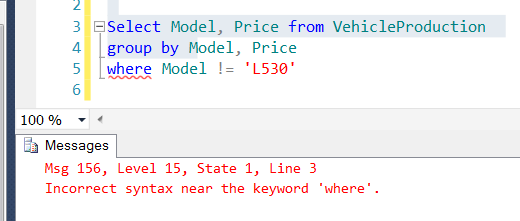


**Example 2**

We can’t use where clause after group by clause

1. **Select** Model, Price **from** VehicleProduction
2. **group** **by** Model, Price
3. **where** Model != 'L530'

**Output**



1. Can we use having and group by clause without Aggregate functions? Justify your answer

Although most of the time GROUP BY and HAVING is used along with aggregate functions, it can still be used without aggregate functions — to find unique records

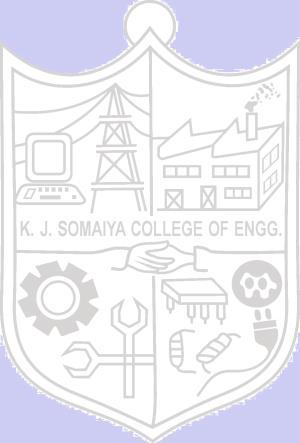
**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Outcomes:**

CO1. Realize the features of Relational database management systems.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Conclusion: (Conclusion to be based on the objectives and outcomes achieved)**



Successful completion of different views and we were able to understand the functions and the impact of inserting the main table would create values in different views but when we create an insert statement in the view columns there are certain entries in columns that are not specified and thus it appears null in the main table

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

**Signature of faculty in-charge with date**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**References:**

**Books/ Journals/ Websites:**

1. Korth, Slberchatz,Sudarshan, :”Database System Concepts”, 6th Edition, McGraw –

Hill

1. Elmasri and Navathe, “ Fundamentals of Database Systems”, 5thEdition, PEARSON

Education.