

# Batch: *A-4* Roll No.: *16010422211* Experiment No: 06

**Aim:** To create nested queries for the given database



**Resources needed:** PostgreSQL PgAdmin3



# 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

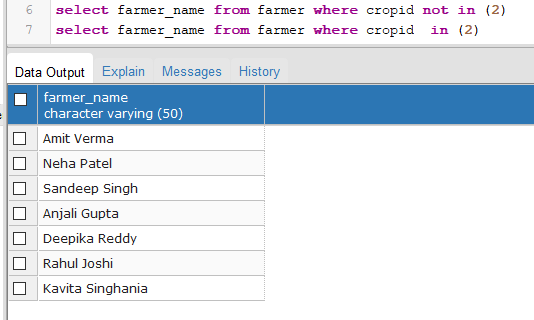
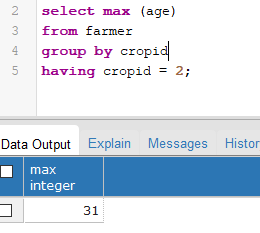
select book\_name from book where exists( select \* from order where book\_id=order.book\_id);

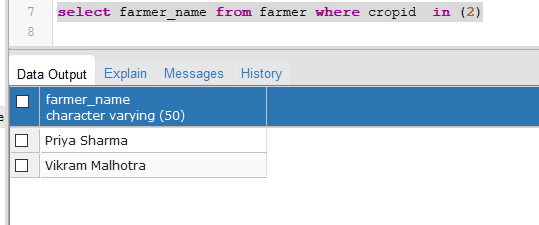
**Procedure / Approach /Algorithm / Activity Diagram:**

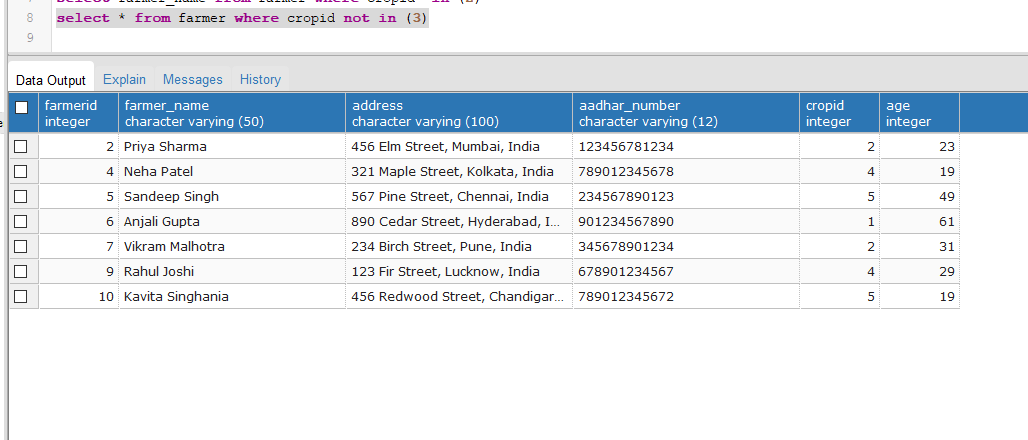
* 1. Refer different syntax given in theory section and formulate queries consisting of nested sub queries, in , not in, as, group by, having etc clauses and different set operations for your database.

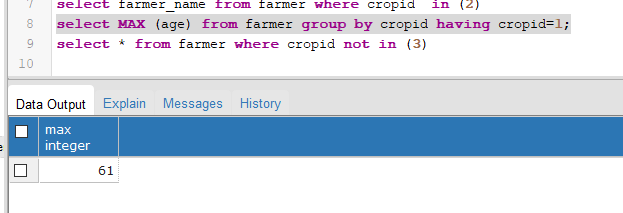
**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

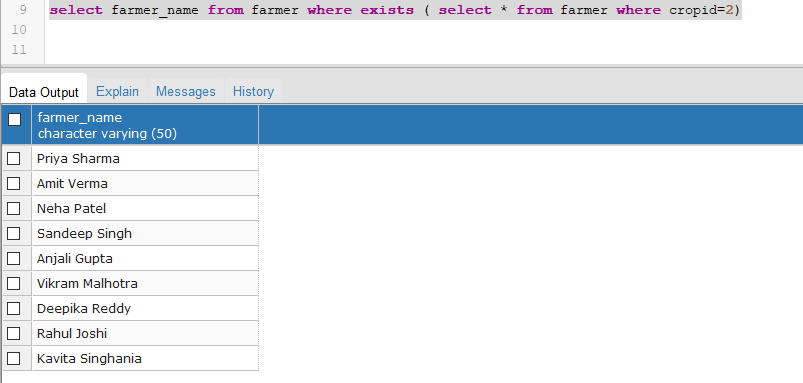
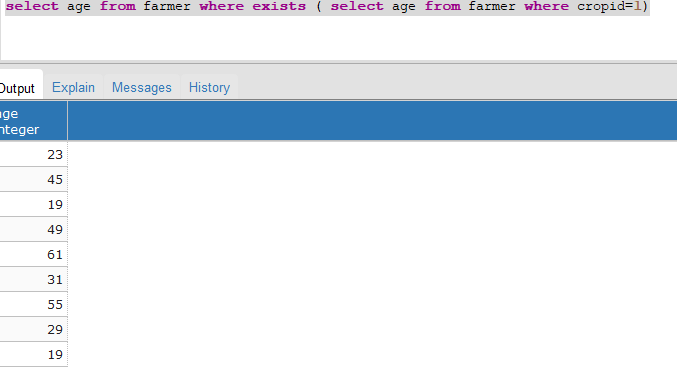
**Results: (Program printout with output / Document printout as per the format)**











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

**Questions:**

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

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

*1.Updating of views is complicated and can be ambiguous task. A view with a single defining table is updatable if the view attributes contain the primary key of the base relation, as well as all attributes with the NOT NULL constraint that do not have default values specified.*

*It is generally not possible to update views defined on multiple tables.*

*It is not possible to update views defined using grouping and aggregate functions.*

*2.Yes, we can use the WHERE clause with the GROUP BY clause in SQL. The GROUP BY clause is used to group rows that have the same values in specified columns into aggregated data, like sum, count, max, min, etc. The WHERE clause, on the other hand, is used to filter rows based on a specified condition. The WHERE clause will always come before GROUP BY*.

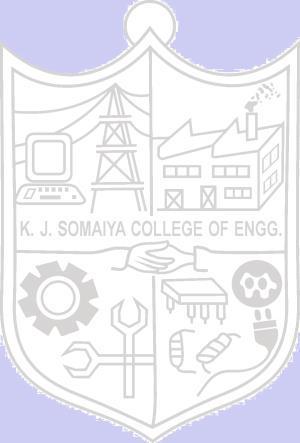
*3. No, we generally cannot use the HAVING clause with the GROUP BY clause without aggregate functions in SQL. The HAVING clause is specifically designed to filter the results of a GROUP BY operation based on aggregate functions' results.* *The HAVING clause, which comes after the GROUP BY clause, allows us to filter the grouped results based on conditions involving the aggregate functions' results. It is used to include or exclude groups from the result set based on those aggregated values.*

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

**Outcomes:** *CO3) Creation of nested queries for the given database.*

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

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



*Understood and implemented Nested queries from the existing database.*

**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.