##### EXPERIMENT NO:-07

**Aim:** **Perform Nested and Complex**

**Theory:**

**NESTED QUERIES:**

A subquery in MySQL is a query, which is nested into another SQL query and embedded with SELECT, INSERT, UPDATE or DELETE statement along with the various operators. We can also nest the subquery with another subquery. A subquery is known as the **inner query**, and the query that contains subquery is known as the **outer query**. The inner query executed first gives the result to the outer query, and then the main/outer query will be performed. [MySQL](https://www.javatpoint.com/mysql-tutorial) allows us to use subquery anywhere, but it must be closed within parenthesis. All subquery forms and operations supported by the SQL standard will be supported in MySQL also.

**The following are the rules to use subqueries:**

* Subqueries should always use in **parentheses.**
* If the main query does not have multiple columns for subquery, then a subquery can have only one column in the SELECT command.
* We can use various comparison operators with the subquery, such as >, <, =, IN, ANY, SOME, and ALL. A multiple-row operator is very useful when the subquery returns more than one row.
* We cannot use the **ORDER BY** clause in a subquery, although it can be used inside the main query.
* If we use a subquery in a **set function**, it cannot be immediately enclosed in a set function.

**The following are the advantages of using subqueries:**

* The subqueries make the queries in a structured form that allows us to isolate each part of a statement.
* The subqueries provide alternative ways to query the data from the table; otherwise, we need to use complex joins and unions.
* The subqueries are more readable than complex join or union statements.

**Syntax :**

**SELECT** column\_list (s) **FROM**  table\_name

**WHERE**  column\_name OPERATOR

(**SELECT** column\_list (s)  **FROM** table\_name [**WHERE**])

SET Operator in MySQL :

# Union

MySQL Union is an operator that allows us to combine two or more results from multiple SELECT queries into a single result set. It comes with a default feature that removes the **duplicate** rows from the result set. MySQL always uses the name of the column in the first SELECT statement will be the column names of the result set(output).

[MySQL](https://www.javatpoint.com/mysql-tutorial) Union must follow these basic rules:

* The number and order of the columns should be the same in all tables that you are going to use.
* The data type must be compatible with the corresponding positions of each select query.
* The column name selected in the different SELECT queries must be in the same order.

Syntax :

**SELECT** column\_list **FROM** table1

**UNION**

**SELECT** column\_list **FROM** table2;

# INTERSECT

The INTERSECT operator is a kind of SET operation in SQL that includes UNION, UNION ALL, MINUS, and INTERSECT. **The INTERSECT operator returns the distinct (common) elements in two sets or common records from two or more tables**. In other words, it compares the result obtained by two queries and produces unique rows, which are the result returned by both queries.

Since [MySQL](https://www.javatpoint.com/mysql-tutorial) does not provide support for the INTERSECT operator. However, we can use the [**INNER JOIN**](https://www.javatpoint.com/mysql-inner-join) and [**IN clause**](https://www.javatpoint.com/mysql-in) to emulate this operator.

# MINUS

The MINUS operator is a kind of SET operation in SQL which also includes INTERSECT, UNION, and UNION ALL. **The MINUS operator returns the unique element from the first table/set, which is not found in the second table/set**. In other words, it will compare the results of two queries and produces the resultant row from the result set obtained by the first query and not found in the result set obtained by the second query.

Since [MySQL](https://www.javatpoint.com/mysql-tutorial) does not provide support for MINUS operator. However, we can use a [**LEFT JOIN** clause](https://www.javatpoint.com/mysql-left-join) to simulate this operator.

**Application of Nested and Set operations on Specified Case study with Input & Output:**

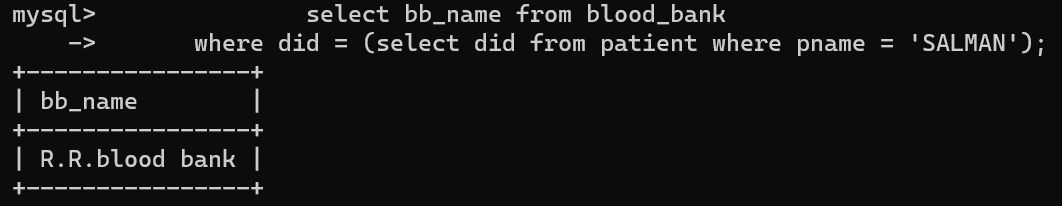
**NESTED QUERIES:**

Q 68: List the blood bank which donates the blood to “SALMAN”

ANS:

select bb\_name from blood\_bank

where did = (select did from patient where pname = “SALMAN”);



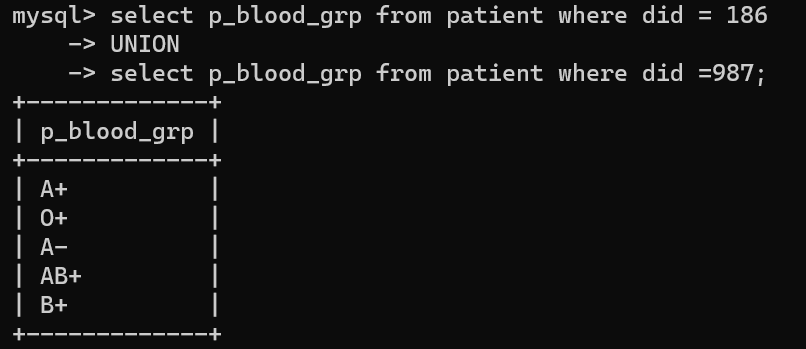
Q 69 : Display the different blood groups in patient 186 or 987.

Ans :

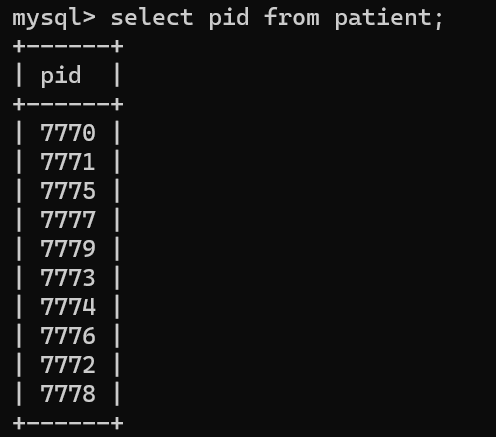
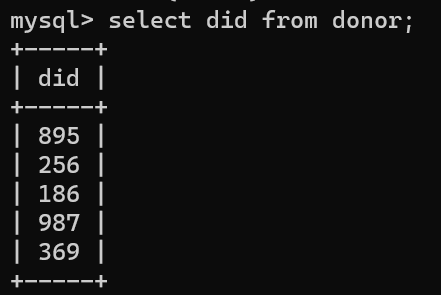
select p\_blood\_grp from patient where did = 186

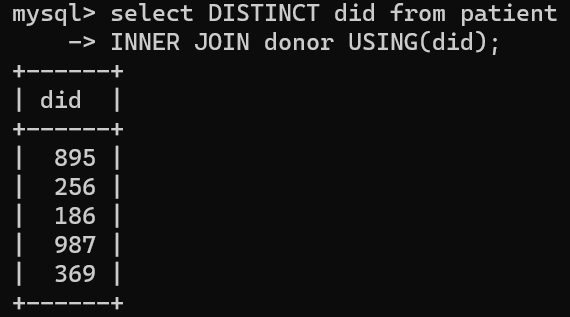
UNION

select p\_blood\_grp from patient where did =987;

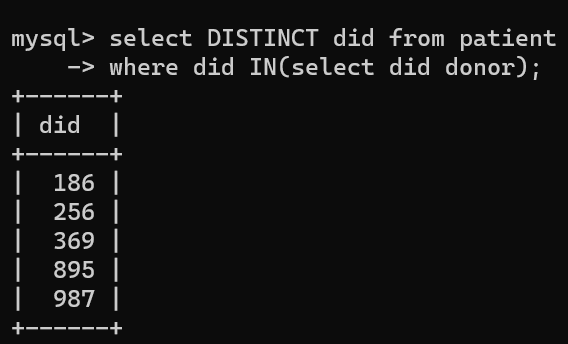


**Intersect:**

****

****

**Minus:**



**Conclusion:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **R1** | **R2** | **R3** | **R4** | **Total** | **Sign with Date** |
| **(3)** | **(5)** | **(4)** | **(3)** | **(15)** |  |
|  |  |  |  |  |