**1. Filtering Data with a Subquery:**

SELECT \* FROM employees

WHERE salary > (SELECT AVG(salary) FROM employees);

- This query retrieves employees whose salary is higher than the average salary.

- The subquery calculates the average salary and provides it as a filter for the outer query.

mysql> select \* from employees;

+-------+---------+---------------+--------+

| Empid | empname | department\_id | salary |

+-------+---------+---------------+--------+

| 1 | Sonam | 101 | 20000 |

| 2 | Heena | 101 | 35000 |

| 3 | Rina | 102 | 10000 |

| 4 | Pooja | 103 | 35000 |

| 5 | Raj | 104 | 50000 |

| 6 | Vamikroota | 102 | 10000 |

| 7 | Rani | 102 | 20000 |

| 8 | Prema | 104 | 35000 |

| 9 | Himant | 102 | 56000 |

+-------+---------+---------------+--------+

9 rows in set (0.00 sec)

mysql> select \* from employees where salary>(select avg(salary) from employees);

+-------+---------+---------------+--------+

| Empid | empname | department\_id | salary |

+-------+---------+---------------+--------+

| 2 | Heena | 101 | 35000 |

| 4 | Pooja | 103 | 35000 |

| 5 | Raj | 104 | 50000 |

| 8 | Prema | 104 | 35000 |

| 9 | Himant | 102 | 56000 |

+-------+---------+---------------+--------+

5 rows in set (0.00 sec)

**2. Selecting from a Set Returned by a Subquery:**

SELECT name, department FROM employees

WHERE department IN (SELECT department FROM departments WHERE location = 'New York');

- This query selects names and departments of employees who work in departments located in New York.

- The subquery fetches departments in New York, and the outer query uses those results to filter employee records.

mysql> select \* from department;

+---------------+-----------------+-------------+

| department\_id | department\_name | location |

+---------------+-----------------+-------------+

| 101 | Hr | Mumbai |

| 102 | Finace | Delhi |

| 103 | It | hybberabaad |

+---------------+-----------------+-------------+

3 rows in set (0.00 sec)

mysql> select \* from employees;

+-------+---------+---------------+--------+-----------------+

| Empid | empname | department\_id | salary | department\_name |

+-------+---------+---------------+--------+-----------------+

| 1 | Sonam | 101 | 20000 | IT |

| 2 | Heena | 101 | 35000 | fiance |

| 3 | Rina | 102 | 10000 | Hr |

| 4 | Pooja | 103 | 35000 | Hr |

| 5 | Raj | 104 | 50000 | fiance |

| 6 | Vamika | 102 | 10000 | IT |

| 7 | Rani | 102 | 20000 | IT |

| 8 | Prema | 104 | 35000 | fiance |

| 9 | Himant | 102 | 56000 | Hr |

+-------+---------+---------------+--------+-----------------+

9 rows in set (0.00 sec)

mysql> select empname, department\_name from employees where department\_name in(select department\_name from department where location="Mumbai");

+---------+-----------------+

| empname | department\_name |

+---------+-----------------+

| Rina | Hr |

| Pooja | Hr |

| Himant | Hr |

+---------+-----------------+

3 rows in set (0.00 sec)

mysql> select empname, department\_name from employees where department\_name in(select department\_name from department where location="hybberabaad");

+---------+-----------------+

| empname | department\_name |

+---------+-----------------+

| Sonam | IT |

| Vamika | IT |

| Rani | IT |

+---------+-----------------+

3 rows in set (0.00 sec)

mysql> select empname, department\_name from employees where department\_name in(select department\_name from department where location="delhi");

+---------+-----------------+

| empname | department\_name |

+---------+-----------------+

| Heena | fiance |

| Raj | fiance |

| Prema | fiance |

+---------+-----------------+

3 rows in set (0.00 sec)

mysql>

**3. Updating Data Based on a Subquery:**

UPDATE employees

SET salary = salary \* 1.1

WHERE department IN (SELECT department FROM departments WHERE budget > 500000);

- This query increases the salary of employees in departments with a budget over 500,000 by 10%.

- The subquery identifies eligible departments, and the outer query applies the salary update.

mysql> update employees set salary=salary\*1.1 where department\_name in (select department\_name from de

partment where salary>50000);

Query OK, 1 row affected (0.01 sec)

Rows matched: 1 Changed: 1 Warnings: 0

mysql> select \* from employees;

+-------+---------+---------------+--------+-----------------+------------+------------+

| Empid | empname | department\_id | salary | department\_name | manager\_id | hire\_date |

+-------+---------+---------------+--------+-----------------+------------+------------+

| 1 | Sonam | 101 | 20000 | IT | 1 | 2023-12-01 |

| 2 | Heena | 101 | 35000 | fiance | 1 | 2023-12-01 |

| 3 | Rina | 102 | 10000 | Hr | 1 | 2023-12-15 |

| 4 | Pooja | 103 | 35000 | Hr | 2 | 2023-12-18 |

| 5 | Raj | 104 | 50000 | fiance | 3 | 2023-12-12 |

| 6 | Vamika | 102 | 10000 | IT | 2 | 2023-12-12 |

| 7 | Rani | 102 | 20000 | IT | 2 | 2023-12-12 |

| 8 | Prema | 104 | 35000 | fiance | 3 | 2023-12-18 |

| 9 | Himant | 102 | 61600 | Hr | 4 | 2023-12-18 |

| 10 | SOnam | NULL | NULL | IT | 4 | 2023-12-18 |

| 11 | vamika | NULL | NULL | Hr | 3 | 2023-12-01 |

| 12 | uma | NULL | NULL | Fiance | 2 | 2023-12-21 |

| 13 | Ajay | NULL | NULL | IT | 3 | 2023-12-21 |

| 17 | Himant | 101 | 50000 | IT | NULL | NULL |

+-------+---------+---------------+--------+-----------------+------------+------------+

14 rows in set (0.00 sec)

mysql>

**4. Inserting Data from a Subquery:**

INSERT INTO employees (name, department)

SELECT name, department FROM new\_hires;

- This query inserts employee names and departments from a table named new\_hires into the employees table.

- The subquery retrieves data from new\_hires, and the outer query inserts it into employees.

insert into new\_hires(name, department\_name)values("SOnam", "IT"),("vamika","H' at line 1

mysql> insert into new\_hires(name, department\_name)values("SOnam", "IT"),("vamika","Hr"),("uma", "Fian

ce"),("Ajay", "IT");

Query OK, 4 rows affected (0.01 sec)

Records: 4 Duplicates: 0 Warnings: 0

mysql> select \* from new\_hires;

+----+--------+-----------------+

| Id | name | department\_name |

+----+--------+-----------------+

| 1 | SOnam | IT |

| 2 | vamika | Hr |

| 3 | uma | Fiance |

| 4 | Ajay | IT |

+----+--------+-----------------+

4 rows in set (0.00 sec)

mysql> insert into employees(empname,department\_name)select name,department\_name from new\_hires;

Query OK, 4 rows affected (0.01 sec)

Records: 4 Duplicates: 0 Warnings: 0

mysql> select \* from employees;

+-------+---------+---------------+--------+-----------------+

| Empid | empname | department\_id | salary | department\_name |

+-------+---------+---------------+--------+-----------------+

| 1 | Sonam | 101 | 20000 | IT |

| 2 | Heena | 101 | 35000 | fiance |

| 3 | Rina | 102 | 10000 | Hr |

| 4 | Pooja | 103 | 35000 | Hr |

| 5 | Raj | 104 | 50000 | fiance |

| 6 | Vamika | 102 | 10000 | IT |

| 7 | Rani | 102 | 20000 | IT |

| 8 | Prema | 104 | 35000 | fiance |

| 9 | Himant | 102 | 56000 | Hr |

| 10 | SOnam | NULL | NULL | IT |

| 11 | vamika | NULL | NULL | Hr |

| 12 | uma | NULL | NULL | Fiance |

| 13 | Ajay | NULL | NULL | IT |

+-------+---------+---------------+--------+-----------------+

13 rows in set (0.00 sec)

**5. Creating a Derived Table with a Subquery:**

SELECT \* FROM (SELECT name, MAX(salary) AS max\_salary FROM employees GROUP BY department) AS salary\_ranks;

- This query creates a temporary table (derived table) named salary\_ranks that contains the highest salary for each department.

- The subquery calculates the maximum salaries, and the outer query treats the result as a table to select from.

mysql> select \* from(select empname, max(salary) as max\_salary from employees group by department\_name

)as salary\_ranks;

ERROR 1055 (42000): Expression #1 of SELECT list is not in GROUP BY clause and contains nonaggregated column 'subquries.employees.empname' which is not functionally dependent on columns in GROUP BY clause; this is incompatible with sql\_mode=only\_full\_group\_by

**6. Subquery in WHERE Clause:**

-- Find employees who earn more than the average salary in their department

SELECT employee\_id, employee\_name, salary, department\_id

FROM employees

WHERE salary > any (SELECT AVG(salary) FROM employees GROUP BY department\_id);

mysql> select \* from employees where salary>any(select avg(salary) from employees group by department\_id);

+-------+---------+---------------+--------+-----------------+------------+------------+

| Empid | empname | department\_id | salary | department\_name | manager\_id | hire\_date |

+-------+---------+---------------+--------+-----------------+------------+------------+

| 2 | Heena | 101 | 35000 | fiance | 1 | 2023-12-01 |

| 4 | Pooja | 103 | 35000 | Hr | 2 | 2023-12-18 |

| 5 | Raj | 104 | 50000 | fiance | 3 | 2023-12-12 |

| 8 | Prema | 104 | 35000 | fiance | 3 | 2023-12-18 |

| 9 | Himant | 102 | 61600 | Hr | 4 | 2023-12-18 |

| 17 | Himant | 101 | 50000 | IT | NULL | NULL |

+-------+---------+---------------+--------+-----------------+------------+------------+

6 rows in set (0.00 sec)

mysql> select \* from avg(salary) from employees;

ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near '(salary) from employees' at line 1

mysql> select avg(salary) from employees;

+-------------+

| avg(salary) |

+-------------+

| 32660.0000 |

+-------------+

**7. Subquery in SELECT Clause:**

-- Retrieve employee details along with their department's total salary

SELECT employee\_id, employee\_name, salary, department\_id,

(SELECT SUM(salary) FROM employees WHERE department\_id = e.department\_id) as department\_total\_salary

FROM employees e;

mysql> select empid, empname,salary, department\_id , (select sum(salary) from employees where departme

nt\_id=e.department\_id) as department\_total\_salary from employees e;

+-------+---------+--------+---------------+-------------------------+

| empid | empname | salary | department\_id | department\_total\_salary |

+-------+---------+--------+---------------+-------------------------+

| 1 | Sonam | 20000 | 101 | 55000 |

| 2 | Heena | 35000 | 101 | 55000 |

| 3 | Rina | 10000 | 102 | 96000 |

| 4 | Pooja | 35000 | 103 | 35000 |

| 5 | Raj | 50000 | 104 | 85000 |

| 6 | Vamika | 10000 | 102 | 96000 |

| 7 | Rani | 20000 | 102 | 96000 |

| 8 | Prema | 35000 | 104 | 85000 |

| 9 | Himant | 56000 | 102 | 96000 |

| 10 | SOnam | NULL | NULL | NULL |

| 11 | vamika | NULL | NULL | NULL |

| 12 | uma | NULL | NULL | NULL |

| 13 | Ajay | NULL | NULL | NULL |

+-------+---------+--------+---------------+-------------------------+

13 rows in set (0.00 sec)

**8. Subquery in FROM Clause:**

-- Find the average salary of employees in each department and list departments with average salary greater than a threshold

SELECT department\_id, avg\_salary

FROM (SELECT department\_id, AVG(salary) as avg\_salary

FROM employees

GROUP BY department\_id) AS department\_avg

WHERE avg\_salary > 50000;

mysql> select department\_id, avg\_salary from (select department\_id, avg(salary) as avg\_salary from employees group by department\_id) as department\_avg where avg\_salary<50000;

+---------------+------------+

| department\_id | avg\_salary |

+---------------+------------+

| 101 | 27500.0000 |

| 102 | 24000.0000 |

| 103 | 35000.0000 |

| 104 | 42500.0000 |

+---------------+------------+

**9. Subquery in HAVING Clause:**

-- Find departments with more than 3 employees and their average salary is above a certain threshold

SELECT department\_id, COUNT(employee\_id) as num\_employees, AVG(salary) as avg\_salary

FROM employees

GROUP BY department\_id

HAVING num\_employees > 3 AND avg\_salary > 60000;

mysql> select department\_id, count(empid) as num\_employees, avg(salary) as avg\_salary from employees group by department\_id having num\_employees>=1 and avg\_salary>20000;

+---------------+---------------+------------+

| department\_id | num\_employees | avg\_salary |

+---------------+---------------+------------+

| 101 | 2 | 27500.0000 |

| 102 | 4 | 24000.0000 |

| 103 | 1 | 35000.0000 |

| 104 | 2 | 42500.0000 |

+---------------+---------------+------------+

4 rows in set (0.00 sec)

mysql> select department\_id, count(empid) as num\_employees, avg(salary) as avg\_salary from employees group by department\_id having num\_employees>=1 and avg\_salary>30000;

+---------------+---------------+------------+

| department\_id | num\_employees | avg\_salary |

+---------------+---------------+------------+

| 103 | 1 | 35000.0000 |

| 104 | 2 | 42500.0000 |

+---------------+---------------+------------+

**10. Correlated Subquery:**

-- Find employees whose salary is above the average salary in their department

-- This uses a correlated subquery as it references the outer query

SELECT employee\_id, employee\_name, salary, department\_id

FROM employees e

WHERE salary > (SELECT AVG(salary) FROM employees WHERE department\_id = e.department\_id);

mysql> select empid, empname, salary, department\_id from employees as e where salary>(select avg(sal

ary) from employees where department\_id=e.department\_id);

+-------+---------+--------+---------------+

| empid | empname | salary | department\_id |

+-------+---------+--------+---------------+

| 2 | Heena | 35000 | 101 |

| 5 | Raj | 50000 | 104 |

| 9 | Himant | 56000 | 102 |

+-------+---------+--------+---------------+

3 rows in set (0.00 sec)

**11. Subquery with EXISTS:**

-- Find departments that have at least one employee

SELECT department\_id, department\_name

FROM departments d

WHERE EXISTS (SELECT 1 FROM employees e WHERE e.department\_id = d.department\_id);

mysql> select department\_id, department\_name from department as d where exists (select 1 from employees as e where e.department\_id=d.department\_id);

+---------------+-----------------+

| department\_id | department\_name |

+---------------+-----------------+

| 101 | Hr |

| 102 | fiance |

| 103 | It |

+---------------+-----------------+

3 rows in set (0.00 sec)

**12. Subquery with IN:**

-- Find employees who work in departments with specific IDs

SELECT employee\_id, employee\_name, department\_id

FROM employees

WHERE department\_id IN (SELECT department\_id FROM departments WHERE

location\_id = 1700);

mysql> select empid, empname, department\_id from employees where department\_id in (select department\_i

d from department where location="Delhi");

+-------+---------+---------------+

| empid | empname | department\_id |

+-------+---------+---------------+

| 3 | Rina | 102 |

| 6 | Vamika | 102 |

| 7 | Rani | 102 |

| 9 | Himant | 102 |

+-------+---------+---------------+

4 rows in set (0.00 sec)

**13. Subquery with NOT IN:**

-- Find employees who don't have a manager

SELECT employee\_id, employee\_name, manager\_id

FROM employees

WHERE manager\_id NOT IN (SELECT employee\_id FROM employees WHERE manager\_id IS NOT NULL);

mysql> select empid, empname, manager\_id from employees where manager\_id in(select empid from employee

s where manager\_id is not null);

+-------+---------+------------+

| empid | empname | manager\_id |

+-------+---------+------------+

| 1 | Sonam | 1 |

| 2 | Heena | 1 |

| 3 | Rina | 1 |

| 4 | Pooja | 2 |

| 5 | Raj | 3 |

| 6 | Vamika | 2 |

| 7 | Rani | 2 |

| 8 | Prema | 3 |

| 9 | Himant | 4 |

| 10 | SOnam | 4 |

| 11 | vamika | 3 |

| 12 | uma | 2 |

| 13 | Ajay | 3 |

+-------+---------+------------+

13 rows in set (0.00 sec)

mysql> select empid, empname, manager\_id from employees where manager\_id not in(select empid from empl

oyees where manager\_id is not null);

Empty set (0.00 sec)

**14. Scalar Subquery:**

-- Find employees and their bonus percentage based on the average salary

SELECT employee\_id, employee\_name, salary, salary / (SELECT AVG(salary) FROM employees) as bonus\_percentage

FROM employees;

**15. Subquery with ORDER BY and LIMIT:**

-- Find the top 5 highest-paid employees

SELECT employee\_id, employee\_name, salary

FROM employees

ORDER BY salary DESC

LIMIT 5;

mysql> s

elect empid,empname, salary from employees order by salary desc limit 5;

+-------+---------+--------+

| empid | empname | salary |

+-------+---------+--------+

| 9 | Himant | 56000 |

| 5 | Raj | 50000 |

| 17 | Himant | 50000 |

| 2 | Heena | 35000 |

| 4 | Pooja | 35000 |

+-------+---------+--------+

**16. Subquery with Multiple Conditions:**

-- Find employees who work in departments with more than 10 employees and have a salary greater than 60000

SELECT employee\_id, employee\_name, department\_id, salary

FROM employees

WHERE department\_id IN (SELECT department\_id FROM employees GROUP BY department\_id HAVING COUNT(\*) > 10)

AND salary > 60000;

mysql> select empid, empname, department\_id , salary from employees where department\_id in (select department\_id from employees group by department\_id having count(\*)>1) and salary>20000;

+-------+---------+---------------+--------+

| empid | empname | department\_id | salary |

+-------+---------+---------------+--------+

| 2 | Heena | 101 | 35000 |

| 5 | Raj | 104 | 50000 |

| 8 | Prema | 104 | 35000 |

| 9 | Himant | 102 | 56000 |

| 17 | Himant | 101 | 50000 |

+-------+---------+---------------+--------+

5 rows in set (0.00 sec)

**17. Subquery with JOIN:**

-- Find employees and their department names using a subquery with a JOIN

SELECT employee\_id, employee\_name, department\_name

FROM employees e

JOIN (SELECT department\_id, department\_name FROM departments) d ON e.department\_id = d.department\_id;

mysql> select \* from employees as e join department as d on e.department\_id=(select department\_id from department where department\_name="IT");

+-------+---------+---------------+--------+-----------------+------------+------------+---------------+-----------------+-------------+

| Empid | empname | department\_id | salary | department\_name | manager\_id | hire\_date | department\_id | department\_name | location |

+-------+---------+---------------+--------+-----------------+------------+------------+---------------+-----------------+-------------+

| 4 | Pooja | 103 | 35000 | Hr | 2 | 2023-12-18 | 101 | Hr | Mumbai |

| 4 | Pooja | 103 | 35000 | Hr | 2 | 2023-12-18 | 102 | fiance | Delhi |

| 4 | Pooja | 103 | 35000 | Hr | 2 | 2023-12-18 | 103 | It | hybberabaad |

+-------+---------+---------------+--------+-----------------+------------+------------+---------------+-----------------+-------------+

3 rows in set (0.00 sec)

**18. Subquery with Aggregation in HAVING Clause:**

-- Find departments with the highest average salary and list only those with an average salary greater than 70000

SELECT department\_id, AVG(salary) as avg\_salary

FROM employees

GROUP BY department\_id

HAVING AVG(salary) > (SELECT MAX(avg\_salary) FROM (SELECT AVG(salary) as avg\_salary FROM employees GROUP BY department\_id) AS department\_avg);

**Empty set (0.00 sec)**

**19. Subquery with Date Comparison:**

-- Find employees hired after the average hiring date of the company

SELECT employee\_id, employee\_name, hire\_date

FROM employees

WHERE hire\_date > (SELECT AVG(hire\_date) FROM employees);

mysql> select empid, empname, hire\_date from employees where hire\_date>(select avg(hire\_date) from emp

loyees);

+-------+---------+------------+

| empid | empname | hire\_date |

+-------+---------+------------+

| 3 | Rina | 2023-12-15 |

| 4 | Pooja | 2023-12-18 |

| 8 | Prema | 2023-12-18 |

| 9 | Himant | 2023-12-18 |

| 10 | SOnam | 2023-12-18 |

| 12 | uma | 2023-12-21 |

| 13 | Ajay | 2023-12-21 |

+-------+---------+------------+

7 rows in set (0.00 sec)

**20. Subquery with NULL Comparison:**

-- Find employees who do not have a manager

SELECT employee\_id, employee\_name, manager\_id

FROM employees

WHERE manager\_id IS NULL;

mysql> select empid, empname, manager\_id from employees where manager\_id is NUll;

+-------+---------+------------+

| empid | empname | manager\_id |

+-------+---------+------------+

| 17 | Himant | NULL |

+-------+---------+------------+

1 row in set (0.00 sec)

**21. Subquery for Inserting Records: Error**

-- Insert a new department and assign it a manager by retrieving the employee ID using a subquery

INSERT INTO departments (department\_name, manager\_id)

VALUES ('Marketing', (SELECT employee\_id FROM employees WHERE employee\_name = 'John Doe'));

**22. Subquery in a JOIN Condition:**

-- Find employees and their corresponding department names using a subquery in a JOIN condition

SELECT employee\_id, employee\_name, department\_name

FROM employees e

JOIN departments d ON e.department\_id = (SELECT department\_id FROM departments WHERE department\_name = 'Sales');

mysql> select empid, empname, d.department\_name from employees as e join department as d on e.department\_id=(select department\_id from department where department\_name="Hr");

+-------+---------+-----------------+

| empid | empname | department\_name |

+-------+---------+-----------------+

| 17 | Himant | Hr |

| 2 | Heena | Hr |

| 1 | Sonam | Hr |

| 17 | Himant | fiance |

| 2 | Heena | fiance |

| 1 | Sonam | fiance |

| 17 | Himant | It |

| 2 | Heena | It |

| 1 | Sonam | It |

+-------+---------+-----------------+

9 rows in set (0.00 sec)

**23. Subquery with EXISTS in DELETE Statement: (not understand)**

-- Delete employees who do not have any dependent records in the dependents table

DELETE FROM employees

WHERE NOT EXISTS (SELECT 1 FROM dependents WHERE employee\_id = employees.employee\_id);

**24. Subquery in CASE Statement:**

-- Classify employees based on their salary using a subquery in a CASE statement

SELECT employee\_id, employee\_name, salary,

CASE

WHEN salary > (SELECT AVG(salary) FROM employees) THEN 'Above Average'

ELSE 'Below Average'

END AS salary\_classification

FROM employees;

mysql> select empid, empname, salary, case when salary>(select avg(Salary) from employees)then "above a

verage" else "below average" end as salary\_classification from employees;

+-------+---------+--------+-----------------------+

| empid | empname | salary | salary\_classification |

+-------+---------+--------+-----------------------+

| 1 | Sonam | 20000 | below average |

| 2 | Heena | 35000 | above average |

| 3 | Rina | 10000 | below average |

| 4 | Pooja | 35000 | above average |

| 5 | Raj | 50000 | above average |

| 6 | Vamika | 10000 | below average |

| 7 | Rani | 20000 | below average |

| 8 | Prema | 35000 | above average |

| 9 | Himant | 61600 | above average |

| 10 | SOnam | NULL | below average |

| 11 | vamika | NULL | below average |

| 12 | uma | NULL | below average |

| 13 | Ajay | NULL | below average |

| 17 | Himant | 50000 | above average |

| 18 | Somya | NULL | below average |

| 19 | Suraj | NULL | below average |

+-------+---------+--------+-----------------------+

**25. Subquery with ALL:**

-- Find employees whose salary is higher than all employees in the Marketing department

SELECT empid, empname, salary

FROM employees

WHERE salary > ALL (SELECT salary FROM employees WHERE department\_id = (SELECT department\_id FROM department WHERE department\_name = 'IT'));

mysql> select empid, empname, salary from employees where salary>all(select salary from employees where

department\_id=(select department\_id from department where department\_name-"IT"));

+-------+---------+--------+

| empid | empname | salary |

+-------+---------+--------+

| 1 | Sonam | 20000 |

| 2 | Heena | 35000 |

| 3 | Rina | 10000 |

| 4 | Pooja | 35000 |

| 5 | Raj | 50000 |

| 6 | Vamika | 10000 |

| 7 | Rani | 20000 |

| 8 | Prema | 35000 |

| 9 | Himant | 61600 |

| 10 | SOnam | NULL |

| 11 | vamika | NULL |

| 12 | uma | NULL |

| 13 | Ajay | NULL |

| 17 | Himant | 50000 |

| 18 | Somya | NULL |

| 19 | Suraj | NULL |

+-------+---------+--------+

16 rows in set, 4 warnings (0.01 sec)1