**6. Cursors: (All types: Implicit, Explicit, Cursor FOR Loop, Parameterized Cursor)**

**Write a PL/SQL block of code using parameterized Cursor that will merge the data availablein**

**the newly created table N\_RollCall with the data available in the table O\_RollCall. If the data in**

**the first table already exist in the second table then that data should be skipped.**

create table o\_rollcall(roll\_no int,name varchar(20),address varchar(20));

Query OK, 0 rows affected (0.28 sec)

mysql> create table n\_rollcall(roll\_no int,namevarchar(20),address varchar(20));

Query OK, 0 rows affected (0.27 sec)

mysql> insert into o\_rollcall values('1','Hitesh','Nandura'); Query OK, 1 row affected (0.05 sec)

mysql> insert into o\_rollcall values('2','Piyush','MP');

Query OK, 1 row affected (0.06 sec)

mysql> insert into o\_rollcall values('3','Ashley','Nsk');

Query OK, 1 row affected (0.05 sec)

mysql> insert into o\_rollcall values('4','Kalpesh','Dhule');

Query OK, 1 row affected (0.05 sec)

mysql> insert into o\_rollcall values('5','Abhi','Satara');

Query OK, 1 row affected (0.04 sec)

mysql> delimiter //

create procedure p3(in r1 int)

begin

declare r2 int;

declare exit\_loop boolean;

declare c1 cursor for select roll\_no from o\_rollcall where roll\_no>r1;

declare continue handler for not found set exit\_loop=true;

open c1;

e\_loop:loop

fetch c1 into r2;

if not exists(select \* from n\_rollcall where roll\_no=r2)

then

insert into n\_rollcall select \* from o\_rollcall where roll\_no=r2;

end if;

if exit\_loop

then

close c1;

leave e\_loop;

end if;

end loop e\_loop;

end

//

Query OK, 0 rows affected (0.00 sec)

Delimiter ;

mysql> call p3(3);

-> //

Query OK, 0 rows affected (0.10 sec)

mysql> select \* from n\_rollcall;

-> //

+ + + +

| roll\_no | name

| address |

+ + + +

| 4 | Kalpesh | Dhule |

| 5 | Abhi |

| Satara

+ + + + 2 rows in set (0.00 sec)

mysql> call p3(0);

-> //

Query OK, 0 rows affected (0.22 sec)

mysql> select \* from n\_rollcall;

-> //

+ + + +

| roll\_no | name

| address |

+ + + +

| 4 | Kalpesh | Dhule |

| 5 | Abhi | Satara |

| 1 | Hitesh | Nandura |

| 2 | Piyush | MP |

| 3 | Ashley | Nsk |+---------+---------+ +

5 rows in set (0.00 sec)

mysql> insert into o\_rollcall values('6','Patil','Kolhapur');

-> //

Query OK, 1 row affected (0.04 sec)

mysql> call p3(4);

-> //

Query OK, 0 rows affected (0.05 sec)

mysql> select \* from n\_rollcall;

-> //

+ + + +

| roll\_no | name

| address

|

+ + + +

| 4 | Kalpesh | Dhule |

| 5 | Abhi | Satara |

| 1 | Hitesh | Nandura |

| 2 | Piyush | MP |

| 3 | Ashley | Nsk |

| 6 | Patil | Kolhapur |

+ + + + 6 rows in set (0.00 sec)

**Theory: Managing Roll Call Entries**

**Introduction to Roll Call Management:**

Roll call management is essential for tracking attendance in educational institutions and organizations. It involves maintaining accurate records of individuals' presence, which enhances communication and accountability.

**Database Structure:**

In this context, two tables are utilized: o\_rollcall, which stores the primary information (roll number, name, address) of individuals, and n\_rollcall, which serves as a secondary table for storing selected entries.

**Data Insertion**:

Initial data is populated in the o\_rollcall table with details of individuals, laying the groundwork for subsequent processing.

**Stored Procedure Overview:**

A stored procedure named p3 is created to facilitate the transfer of records from o\_rollcall to n\_rollcall. It uses a cursor to iterate through roll numbers greater than a specified value, ensuring that only unique entries are copied to avoid duplicates. This procedure streamlines the process of updating the roll call records based on dynamic criteria.