Assignment No 5

Q . PROBLEM STATEMENT: Write a Stored Procedure namely proc_Grade for the categorization of student. If marks scored by students in examination is <=1500 and marks>=990 then student will be placed in distinction category if marks scored are between 989 and900 category is first class, if marks 899 and 825 category is Higher Second Class.
Write a PL/SQL block for using procedure created with above requirement.
Stud_Marks(name, total_marks) Result (Roll,Name, Class).

SOLUTION:

```
-- Create the table to
store student marks
CREATE TABLE Stud_Marks
(
  Roll NUMBER,
  Name VARCHAR2(100),
  Total_Marks NUMBER
```

```
-- Create the procedure to categorize students
CREATE OR REPLACE PROCEDURE proc_Grade (
  p_Name IN Stud_Marks.Name%TYPE,
  p_Total_Marks IN Stud_Marks.Total_Marks%TYPE,
  p_Result OUT SYS_REFCURSOR
) AS
  v_Class VARCHAR2(100);
BEGIN
  IF p_Total_Marks <= 1500 AND p_Total_Marks >= 990 THEN
  v_Class := 'Distinction';
  ELSIF p_Total_Marks <= 989 AND p_Total_Marks >= 900 THEN
  v_Class := 'First Class';
  ELSIF p_Total_Marks <= 899 AND p_Total_Marks >= 825 THEN
  v_Class := 'Higher Second Class';
  ELSE
  v_Class := 'Not Categorized';
  END IF;
```

```
-- Insert the result into the Result table
  INSERT INTO Result (Roll, Name, Class)
  VALUES (Roll_seq.NEXTVAL, p_Name, v_Class);

  -- Return the result as a cursor
  OPEN p_Result FOR
  SELECT Roll, Name, Class
  FROM Result;
END;
/
```

```
-- Declare variables
DECLARE
  v_Result SYS_REFCURSOR;
BEGIN
  -- Call the procedure for each student
  proc_Grade('John Doe', 1450, v_Result);
  proc_Grade('Jane Smith', 950, v_Result);
  proc_Grade('Michael Johnson', 875, v_Result);

  -- Print the result
  DBMS_OUTPUT.PUT_LINE('Roll | Name           |
Class');
  DBMS_OUTPUT.PUT_LINE('------------------------------
');
  LOOP
  FETCH v_Result INTO Roll, Name, Class;
  EXIT WHEN v_Result%NOTFOUND;
  DBMS_OUTPUT.PUT_LINE(Roll || ' | ' || Name || ' | '
|| Class);
  END LOOP;
  CLOSE v_Result;
END;
/
```

In this PL/SQL block, we first create the table "Stud_Marks" to store the student marks. Then, we create the stored procedure "proc_Grade" that takes the student's name and total marks as input parameters, and categorizes the student based on the marks. The result is inserted into the "Result" table. Finally, we declare variables and call the "proc_Grade" procedure for each student. We then print the result using the DBMS_OUTPUT.PUT_LINE function.

Assignment No 6

Q. PROBLEM STATEMENT: Write a PL/SQL block of code using parameterized Cursor that will merge the data available in 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. Also demonstrate working of all types of cursors.

SOLUTION:

```
create database assi7;
use assi7;
show tables;
create table old_roll(roll int,name varchar(10));
create table new_roll(roll int,name varchar(10));
insert into old_roll values(4,'d');
insert into old_roll values(3,'bcd');
insert into old_roll values(1,'bc');
insert into old_roll values(5,'bch');
insert into new_roll values(2,'b');
insert into new_roll values(5,'bch');
insert into new_roll values(1,'bc');

select * from old_roll;
select * from new_roll;
delimiter $
create procedure roll_list()
begin
declare oldrollnumber int;
declare oldname varchar(10);
declare newrollnumber int;
declare newname varchar(10);
declare done int default false;
declare c1 cursor for select roll,name from old_roll;
declare c2 cursor for select roll,name from new_roll;
declare continue handler for not found set done=true;
open c1;
loop1:loop
fetch c1 into oldrollnumber,oldname;
if done then
leave loop1;
end if;
open c2;
```

```
loop2:loop
fetch c2 into newrollnumber,newname;
if done then
insert into new_roll values(oldrollnumber,oldname);
set done=false;
close c2;
leave loop2;
end if;
if oldrollnumber=newrollnumber then
leave loop2;
end if;
end loop;
end loop;
close c1;
end $
delimiter ;
call roll_list();
select * from new_roll;
```

OUTPUT:

```
SQL> create table CompDep(Roll int,Name
varchar(20));
Table created.
SQL> create table Student(Roll int,Name
varchar(20));
Table created.
SQL> insert into Student values(1,'a');
1 row created.
SQL> insert into Student values(2,'b');
1 row created.
SQL> insert into Student values(3,'c');
1 row created.
SQL> insert into Student values(4,'d');
1 row created.
SQL> insert into CompDep values(2,'b');
1 row created.
SQL> insert into CompDep values(5,'e');
1 row created.
SQL> insert into CompDep values(6,'f');
1 row created
```

Assignment No 7

Q.PROBLEM STATEMENT: Write a database trigger on Library table. The System should keep track of the records that are being updated or deleted. The old value of updated or deleted records should be added in Library_Audit table.

Write a before trigger for Insert, update event considering following requirement: Emp(e_no, e_name, salary) I) Trigger action should be initiated when salary is tried to be inserted is less than Rs. 50,000/- II) Trigger action should be initiated when salary is tried to be updated for value less than Rs. 50,000/- Action should be rejection of update or Insert operation by displaying appropriate error message. Also the new values expected to be inserted will be stored in new table Tracking(e_no, salary).

SOLUTION:

```
-- Create tbl_library table
CREATE TABLE tbl_library (
    bk_no INT PRIMARY KEY,
    bk_name VARCHAR(255),
    issue_date DATE,
    return_date DATE
);
```

```
-- Create tbl_library_audit table
CREATE TABLE tbl_library_audit (
    audit_id INT AUTO_INCREMENT PRIMARY KEY,
    bk_no INT,
    bk_name VARCHAR(255),
    issue_date DATE,
    return_date DATE,
    audit_action VARCHAR(10),
    audit_timestamp TIMESTAMP DEFAULT
CURRENT_TIMESTAMP
);
```

```
-- Insert sample data into tbl_library
INSERT INTO tbl_library (bk_no, bk_name, issue_date, return_date)
VALUES
    (1, 'Book 1', '2023-01-15', '2023-02-15'),
    (2, 'Book 2', '2023-02-01', '2023-03-01'),
    (3, 'Book 3', '2023-03-15', '2023-04-15'),
    (4, 'Book 4', '2023-04-01', '2023-05-01'),
    (5, 'Book 5', '2023-05-15', '2023-06-15');
```

```
-- trigger
create or replace trigger trg_library_af
    after update or delete on tbl_library for each row
    declare
    begin
    if updating then
    insert into tbl_library_audit values
(:old.bk_no,:old.bk_name,:old.issue_date,:old.return_date);
    elsif deleting then
    insert into tbl_library_audit values
(:old.bk_no,:old.bk_name,:old.issue_date,:old.return_date);
    end if;
  end;
  /
```

```
DELIMITER //
CREATE TRIGGER SalaryCheck
BEFORE INSERT ON Emp
FOR EACH ROW
BEGIN
    IF NEW.salary < 50000 THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Salary cannot be less than Rs.
50,000';
    ELSE
        -- Insert the new values into Tracking table
        INSERT INTO Tracking (e_no, salary)
        VALUES (NEW.e_no, NEW.salary);
    END IF;
END;
//
DELIMITER ;
```