Assignment No A4 & A5

Title: Unnamed PL/SQLcode block: Use of Control structure and Exception handling is mandatory.
Suggested Problem statement:

Consider Tables:
1. Borrower(Roll_no, Name, Date of Issue, Name of Book, Status)
2. Fine(Roll_no, Date, Amt)

• Accept Roll_no and Name of Book from user.
• Check the number of days (from date of issue).
• If days are between 15 to 30 then fine amount will be Rs 5per day.
• If no. of days>30, per day fine will be Rs 50 per day and for days less than 30, Rs. 5 per day.
• After submitting the book, status will change from I to R.
• If condition of fine is true, then details will be stored into fine table.
• Also handles the exception by named exception handler or user define exception handler.

OR

Write a PL/SQL code block to calculate the area of a circle for a value of radius varying from 5 to 9. Store the radius and the corresponding values of calculated area in an empty table named areas, consisting of two columns, radius and area.

Objective: To understand the Control structure and Exception handling of pl/sql block structure.

Theory:

What is an anonymous/ Unnamed block in PL SQL?

The PL/SQL anonymous block statement is an executable statement that can contain PL/SQL control statements and SQL statements. It can be used to implement procedural logic in a scripting language. ... The exception section must begin with the keyword EXCEPTION, and continues until the end of the block in which it appears. In PL/SQL contexts, this statement can be compiled and executed by the DB2® data server.

The anonymous block statement, which does not persist in the database, can consist of up to three sections: an optional declaration section, a mandatory executable section, and an optional exception section.

The optional declaration section, which can contain the declaration of variables, cursors, and types that are to be used by statements within the executable and exception sections, is inserted before the executable BEGIN-END block.

The optional exception section can be inserted near the end of the BEGIN-END block. The exception section must begin with the keyword EXCEPTION, and continues until the end of the block in which it appears.

Description:-

DECLARE:-

An optional keyword that starts the DECLARE statement, which can be used to declare data types, variables, or cursors. The use of this keyword depends upon the context in which the block appears.

declaration

Specifies a variable, cursor, or type declaration whose scope is local to the block. Each declaration must be terminated by a semicolon.

BEGIN:-

A mandatory keyword that introduces the executable section, which can include one or more SQL or PL/SQL statements. A BEGIN-END block can contain nested BEGIN-END blocks.

statement

Specifies a PL/SQL or SQL statement. Each statement must be terminated by a semicolon.

EXCEPTION:-

An optional keyword that introduces the exception section.

WHEN exception-condition

Specifies a conditional expression that tests for one or more types of exceptions.

THEN handler-statement

Specifies a PL/SQL or SQL statement that is executed if a thrown exception matches an exception in exception-condition. Each statement must be terminated by a semicolon.

END:-

A mandatory keyword that ends the block.

Design and Implementation:

Create borrower table:-

```
Create table borrower(roll_no number(5),name varchar2(20),Dateofissue date,NameBook varchar2(20),status varchar(20));
```

Insert record in borrower table:-

```
Insert into borrower values(1,‟om‟,‟18-sep-2017‟,‟DBMS‟,‟I‟); Insert into borrower values(2,‟priya‟,‟19-sep-2017‟,‟C++‟,‟I‟); Insert into borrower values(3,‟omkar‟,‟20-sep-2017‟,‟java‟,‟I‟); Create fine table:
```

```
Create table fine(roll_no number(5),sdate date,Amt number(5));
```

Write PROCEDURE PL/SQL Block:-

```
DECLARE

Roll_No NUMBER(3);

BookName varchar(50);

 IssueDate DATE;

CurrentDate DATE;

NoOfDays Number(2);

 Amount Number;
```

```
BEGIN

DBMS_OUTPUT.PUT_LINE('Enter Student Roll Number');

Roll_No := &rollno;

DBMS_OUTPUT.PUT_LINE('Enter Book Name');

BookName := '&bookname';

CurrentDate := trunc(SYSDATE);

SELECTDateOfIssue into IssueDate FROM Borrower WHERE RollNo = Roll_No AND NameOfBook
=BookName;

SELECTtrunc(SYSDATE) - IssueDate INTO NoOfDays from dual; DBMS_OUTPUT.PUT_LINE('No of
Days' || NoOfDays);

IF (NoOfDays> 30) THEN

Amount := NoOfDays * 50;

ELSIF (NoOfDays>= 15 AND NoOfDays<=30) THEN

Amount := NoOfDays * 5; END IF;

IF Amount > 0 THEN

INSERT INTO Fine values (Roll_No, sysdate, Amount);

END IF;

UPDATE Borrower SET Status = 'R' WHERE RollNo=Roll_No;

END;
```

*** /END OF PROCEDURE/***

Execute procedure

Conclusion:- Here we understood the concept of unnamed block of PL/SQL structure. Designed the library due management application using unnamed block of PL/SQL.