Group A

Assignment No 4

Title of the Assignment: Unnamed PL/SQL code block: Use of Control structure and Exception handling is mandatory.

Suggested Problem statement:

Consider Tables:

1. Borrower(Roll_no, Name, DateofIssue, NameofBook, Status)
2. Fine(Roll_no,Date,Amt)

- Accept Roll_no and NameofBook 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 & for days less than 30, Rs. 5 perday.
- 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.

Objective: Understand the concept of Unnamed PL/SQL code, different Control Structure and exception handling

Outcome: Students will be able to learn and understand various control structure and exception handling.

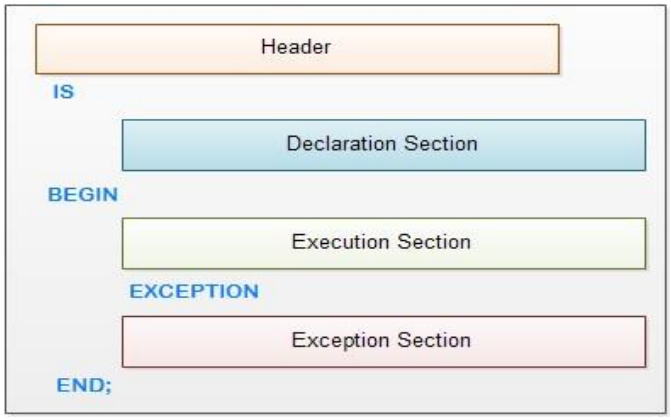
Theory:

Introducing PL/SQL block structure and anonymous block PL/SQL program units organize the code into blocks. A block without a name is known as an anonymous block. The anonymous block is the simplest unit in PL/SQL. It is called anonymous block because it is not saved in the Oracle database.An anonymous block is an only one-time use and useful in certain situations such as creating test units.

The following illustrates anonymous block syntax:

```
[DECLARE]
 Declaration statements;
BEGIN
 Execution statements;
 [EXCEPTION]
 Exception handling statements;
END;/
```

The anonymous block has three basic sections that are the declaration, execution, and exception handling. Only the execution section is mandatory and the others are optional.



- The declaration section allows you to define data types, structures, and variables. You often declare variables in the declaration section by giving those names, data types, & initial values.
- The execution section is required in a block structure & it must have at least one statement.
- The execution section is the place where you put the execution code or business logic code.
- You can use both procedural and SQL statements inside the execution section. The exception handling section is starting with the EXCEPTION keyword.
- The exception section is the place that you put the code to handle exceptions. You can either catch or handle exceptions in the exception section.

❖ IF- Statement:

The IF statement executes a sequence of statements depending on the value of a condition.

Syntax: IFcondition THEN statement1;

ElSEIF condition2 THEN statement2;

ENDIF;

- **Using the LOOP Statement:**

The simplest form of LOOP statement is the basic loop, which encloses a sequence of statements between the keywords LOOP and END LOOP, as follows:

```
LOOP
 sequence_of_statements
END LOOP;
```

With each iteration of the loop, the sequence of statements is executed, then control resumes at the top of the loop. You use an EXIT statement to stop looping and prevent an infinite loop.

You can place one or more EXIT statements anywhere inside a loop, but not outside a loop. There are two forms of EXIT statements: EXIT and EXIT-WHEN.

Exception:

PL/SQL supports programmers to catch such conditions using EXCEPTION block in the program and an appropriate action is taken against the error condition. There are two types of exceptions

1. System-defined exceptions
2. User-defined exceptions

Syntax for Exception Handling:

```
DECLARE
Declaration section
BEGIN
Exception section
EXCEPTION
WHEN ex_name1 THEN
-Error handling statements
WHEN ex_name2 THEN
-Error handling statements
WHEN Others THEN
-Error handling statements
END;
```

❖ Raising Exception:

Exception are raised by the database server automatically whenever there is any internal database error, but exception can be raised explicitly by the programmer by using the command RAISE.

Following is the syntax for raising an exception.

Syntax: DECLARE

```
Exception_name EXCEPTION;
BEGIN
 IF condition THEN
 RAISE exception_name;
 ENDIF;
 EXCEPTION
 WHEN exception_name THEN
 STATEMENT
```

- **User-defined Exceptions:**

Steps to be followed to use user-defined exceptions:

- They should be explicitly declared in the declaration section.
- They should be explicitly raised in the Execution Section.
- They should be handled by referencing the user-defined exception name in the exception section.

Program

```
mysql> use library;
Database changed
mysql> create table borrower(Roll_no int(10) primary key, Name varchar(20) not null, Date_of_issue date not null, Name_of_book varchar(20) not null, Status varchar(1) not null);
Query OK, 0 rows affected, 1 warning (0.03 sec)

mysql> desc borrower;
+---------------+-------------+------+-----+---------+-------+
| Field | Type | Null | Key | Default | Extra |
+---------------+-------------+------+-----+---------+-------+
| Roll_no | int | NO | PRI | NULL | |
| Name | varchar(20) | NO | | NULL | |
| Date_of_issue | date | NO | | NULL | |
| Name_of_book | varchar(20) | NO | | NULL | |
| Status | varchar(1) | NO | | NULL | |
5 rows in set (0.00 sec)

mysql> create table fine(Roll_no int(10), Date date not null, Amount int(10), foreign key(Roll_no) references borrower(Roll_no));
Query OK, 0 rows affected, 2 warnings (0.02 sec)

mysql> desc fine;
```

```
+---------+------+------+-----+---------+-------+
| Field | Type | Null | Key | Default | Extra |
+---------+------+------+-----+---------+-------+
| Roll_no | int | YES | MUL | NULL | |
| Date | date | NO | | NULL | |
| Amount | int | YES | | NULL | |
+---------+------+------+-----+---------+-------+
3 rows in set (0.00 sec)

mysql> insert into borrower values(1, 'Amruta', '2022/07/18', 'MySQL', 'I');
Query OK, 1 row affected, 1 warning (0.01 sec)
mysql> insert into borrower values(2, 'Siddeshdesh', '2022/06/02', 'Computer Network', 'I');
Query OK, 1 row affected, 1 warning (0.02 sec)
mysql> insert into borrower values(3, 'Swarali', '2022/06/22', 'Operating System', 'I');
Query OK, 1 row affected, 1 warning (0.01 sec)
mysql> insert into borrower values(4, 'Bhavesh', '2022/07/17', 'Design of Complier', 'I');
Query OK, 1 row affected, 1 warning (0.02 sec)
mysql> insert into borrower values(5, 'Chaitali', '2022/08/15', 'Internet of Things', 'I');
Query OK, 1 row affected, 1 warning (0.01 sec)
mysql> insert into borrower values(6, 'Omkar', '2022/09/02', 'Mobile Computing', 'I');
Query OK, 1 row affected, 1 warning (0.01 sec)

mysql> select * from borrower;
+---------+-------------+---------------+--------------------+--------+
| Roll_no | Name | Date_of_issue | Name_of_book | Status |
+---------+-------------+---------------+--------------------+--------+
| 1 | Amruta | 2022-07-18 | MySQL | I |
| 2 | Siddeshdesh | 2022-06-02 | Computer Network | I |
| 3 | Swarali | 2022-06-22 | Operating System | I |
| 4 | Bhavesh | 2022-07-17 | Design of Complier | I |
| 5 | Chaitali | 2022-08-15 | Internet of Things | I |
| 6 | Omkar | 2022-09-02 | Mobile Computing | I |
+---------+-------------+---------------+--------------------+--------+
6 rows in set (0.00 sec)

mysql> delimiter //
mysql> create procedure A(IN rollno1 int(10), name1 varchar(20))
 -> begin
 -> declare i_date date;
 -> declare diff int;
 -> declare fine_amt int;
 -> declare EXIT HANDLER FOR SQLEXCEPTION SELECT 'Table not found';
 -> select Date_of_issue into i_date from borrower where Roll_no= rollno1 and Name_of_book = name1;
 -> select DATEDIFF(CURDATE(), i_date) into diff;
 -> if(diff>=15 and diff<=30) then
 -> set fine_amt= diff*5;
```

```
 -> insert into fine values(rollno1, CURDATE(), fine_amt);
 -> elseif(diff>30) then
 -> set fine_amt= diff*50;
 -> insert into fine values(rollno1, CURDATE(), fine_amt);
 -> end if;
 -> update borrower set Status= 'R' where Roll_no= rollno1 and Name_of_book = name1;
 -> end
 -> //
Query OK, 0 rows affected, 1 warning (0.01 sec)

mysql> delimiter ;
mysql> call A(1, 'MySQL');
Query OK, 1 row affected (0.01 sec)

mysql> select * from fine;
+---------+------------+--------+
| Roll_no | Date | Amount |
+---------+------------+--------+
| 1 | 2022-09-04 | 2400 |
+---------+------------+--------+
1 row in set (0.00 sec)

mysql> select * from borrower;
+---------+-------------+---------------+--------------------+--------+
| Roll_no | Name | Date_of_issue | Name_of_book | Status |
+---------+-------------+---------------+--------------------+--------+
| 1 | Amruta | 2022-07-18 | MySQL | R |
| 2 | Siddeshdesh | 2022-06-02 | Computer Network | I |
| 3 | Swarali | 2022-06-22 | Operating System | I |
| 4 | Bhavesh | 2022-07-17 | Design of Complier | I |
| 5 | Chaitali | 2022-08-15 | Internet of Things | I |
| 6 | Omkar | 2022-09-02 | Mobile Computing | I |
+---------+-------------+---------------+--------------------+--------+
6 rows in set (0.00 sec)

mysql> call A(2, 'Computer Network');
Query OK, 1 row affected (0.01 sec)

mysql> select * from borrower;
+---------+-------------+---------------+--------------------+--------+
| Roll_no | Name | Date_of_issue | Name_of_book | Status |
+---------+-------------+---------------+--------------------+--------+
| 1 | Amruta | 2022-07-18 | MySQL | R |
| 2 | Siddeshdesh | 2022-06-02 | Computer Network | R |
| 3 | Swarali | 2022-06-22 | Operating System | I |
| 4 | Bhavesh | 2022-07-17 | Design of Complier | I |
```

```
| 5 | Chaitali | 2022-08-15 | Internet of Things | I |
| 6 | Omkar | 2022-09-02 | Mobile Computing | I |
+---------+-------------+---------------+--------------------+--------+
6 rows in set (0.00 sec)

mysql> select * from fine;
+---------+------------+--------+
| Roll_no | Date | Amount |
+---------+------------+--------+
| 1 | 2022-09-04 | 2400 |
| 2 | 2022-09-04 | 4700 |
+---------+------------+--------+
2 rows in set (0.00 sec)

mysql> call A(5, 'Internet of Things');
Query OK, 1 row affected (0.00 sec)
mysql> select * from fine;
+---------+------------+--------+
| Roll_no | Date | Amount |
+---------+------------+--------+
| 1 | 2022-09-04 | 2400 |
| 2 | 2022-09-04 | 4700 |
| 5 | 2022-09-04 | 100 |
+---------+------------+--------+
3 rows in set (0.01 sec)

mysql> select * from borrower;
+---------+-------------+---------------+--------------------+--------+
| Roll_no | Name | Date_of_issue | Name_of_book | Status |
+---------+-------------+---------------+--------------------+--------+
| 1 | Amruta | 2022-07-18 | MySQL | R |
| 2 | Siddeshdesh | 2022-06-02 | Computer Network | R |
| 3 | Swarali | 2022-06-22 | Operating System | I |
| 4 | Bhavesh | 2022-07-17 | Design of Complier | I |
| 5 | Chaitali | 2022-08-15 | Internet of Things | R |
| 6 | Omkar | 2022-09-02 | Mobile Computing | I |
+---------+-------------+---------------+--------------------+--------+
6 rows in set (0.00 sec)

mysql> call A(6, 'Mobile Computing');
Query OK, 1 row affected (0.01 sec)
mysql> select * from borrower;
+---------+-------------+---------------+--------------------+--------+
| Roll_no | Name | Date_of_issue | Name_of_book | Status |
+---------+-------------+---------------+--------------------+--------+
| 1 | Amruta | 2022-07-18 | MySQL | R |
```

```
| 2 | Siddeshdesh | 2022-06-02 | Computer Network | R |
| 3 | Swarali | 2022-06-22 | Operating System | I |
| 4 | Bhavesh | 2022-07-17 | Design of Complier | I |
| 5 | Chaitali | 2022-08-15 | Internet of Things | R |
| 6 | Omkar | 2022-09-02 | Mobile Computing | R |
+---------+-------------+---------------+--------------------+--------+
6 rows in set (0.01 sec)

mysql> select * from fine;
+---------+------------+--------+
| Roll_no | Date | Amount |
+---------+------------+--------+
| 1 | 2022-09-04 | 2400 |
| 2 | 2022-09-04 | 4700 |
| 5 | 2022-09-04 | 100 |
+---------+------------+--------+
3 rows in set (0.00 sec)
```

Conclusion: Successfully implemented the PL/SQL code with proper understanding of different control structure and exception handling.

Viva Questions:

- What is PL/SQL? Explain the structure of PL/SQL with example.
- Explain types of PL/SQL block?
- What is procedure? How to create procedure explain with example?
- What is named block of PL/SQL?
- Explain the if-else structure with example?

Date:	
Marks obtained:	
Sign of course coordinator:	
Name of course Coordinator:	