31341

Assignment A5

Problem Statement:

Write a PL/SQL block of code for the given requirements:

Schema:

Customer(Cust_id, Name, DateOfPayment, NameOfScheme, Status)

Fine(Cust_id, Date, Amt)

1. Accept Cust_id and name of scheme from user
2. Check the number of days (from date of payment). If days are between 15 and 30, then fine amount will be Rs.5 per day
3. If number of days > 30, per day fine will be Rs.50 per day and for days less than 30, Rs.5 per day
4. After payment, status will change from N to P.

If condition of fine is true, then details will be stored in Fine table.

Learning Objective:

1. To understand and write PL/SQL block code requirements defined
2. To understand exception handling
3. To understand basic structure of PL/SQL block

Learning Outcomes: The student will be able to implement:

1. PL/SQL block, user-defined and predefined exception handling
2. Control structure using PL/SQL

Theory:

PLSQL stands for Procedural Language /Structured Query Language. It offers a set of procedural commands (IF statements, loops, assignments), organized within blocks that complement and extend the reach of sql.

Blocks:

A block is defined by the keywords Declare, Begin, Exception, and End, which break up the block into 3 sections:

1. Declarative: Statements that declare variables, constants, and other code elements, which can be used within that block.
2. Executable: Statements that are run when the block is executed.
3. Exception Handling: A specially structured section you can use to catch any exceptions that are raised when the executable section runs.

Example of Block:

```
Begin
...
End;
```

Exception Handling

When an exception occurs, a message which explains its cause is received. It consists of 3 parts:

1. Type of exception
2. An error code
3. A message

Example:

```
Declare
...declaration section
Begin
...exception section
Exception
When ex_name1 then
...error handling
When ex_name2 them
...error handling
...other handlers
```

```
.
End;
```

When others exception is used to manage the exceptions that are not explicitly handled. Only one exception can be raised in a Block, and the control does not return to the Execution Section after the error is handled.

There are 3 types of exceptions:

1. Named system exceptions
2. Unnamed system exceptions
3. User-defined exceptions

1. Named system exceptions

These are raised when a program violates a RDBMS rule. Example: NO_DATA_FOUND and ZERO_DIVIDE. These exceptions are not declared explicitly and are raised implicitly when a predefined error occurs. It is caught by referencing a standard name within an exception-handling routine.

2. Unnamed system exceptions

These exceptions do not have an Oracle provided name. They do not occur frequently. These exceptions have a code and an associated message. These can be handled using When Others or associating the exception code to a name and using it as a named exception. This can be done using a Pragma called EXCEPTION_INIT.

Syntax:

```
Declare
  exception_name Exception;
  Pragma
  EXCEPTION_INIT(exception_name, err_code);
Begin
...
Exception
  When exception_name Then handle the exception
```

```
.
End;
```

3. User-defined exceptions

These exceptions are defined explicitly based on business rules.

Steps to use user-defined exceptions:

i. They should be explicitly declared in the declared section.

ii. They should be explicitly raised in the exception section.

iii. They should be handled by referencing the user-defined exception in the exception section

Syntax:

```
Declare
exception_name Exception;
...
Begin
  ...
  IF condition THEN Raise exception_name
Exception
  When exception_name then (handle_exception)
End;
```

Conclusion: We were successfully able to write a program to demonstrate exception handling in PL/SQL