EX:NO:6 DATE:	WRITE A PL/SQL BLOCK TO SATISFY SOME CONDITIONS BY ACCEPTING INPUT FROM THE USER.

AIM:

To write a PL/SQL block to Satisfy Some Conditions by accepting input from the user.

* SYNTAX OF TAKING INPUT FROM THE USER:

<variable name>:=:<variable name>;

EX: NO: 07 DATE:	WRITE A PL/SQL BLOCK THAT HANDLES ALL TYPES OF EXCEPTIONS.

AIM:-

To Study the PL/SQL block that handles all types of exceptions.

EXCEPTION:-

* An error Condition during a program execution is called an exception in PL/SQL. PL/SQL Supports programmers to catch Such Condition Using EXCEPTION block in the program and an appropriate action is taken against the error Condition. There are two types of exceptions:
  * System-defined exception.
  * User-defined exception.

* SYNTAX FOR EXCEPTION HANDLING:-

```
DECLARE
    < declaration Section >
BEGIN
    < executable command (s) >
EXCEPTION
    < exception handling goes here >
 WHEN exception1 THEN
      exception 1-handling-Statements
```

```
WHEN exception2 THEN
    exception2-handling-Statements
WHEN exception3 THEN
    exception3-handling-statements
....
WHEN others THEN
    exception3-handling-statements
END;
```

* USER-DEFINED EXCEPTION:-

* PL/SQL allow you to define your own exception according to the need of your program. A user-defined exception must be declared and then raised explicitly, Using either a RAISE statement or the procedure DBMS_STANDARD.RAISE_APPLICATION_ERROR.

SYNTAX:

```
DECLARE
    my-exception EXCEPTION;
```

PREDEFINED EXCEPTION:

* PL/SQL provides many predefined exceptions, which are executed when any database rule is Violated by a program. for example: the predefined exception NO_DATA_FOUND.

EX:-NO:8 DATE:	CREATION OF PROCEDURES

* AIM:

To Study the Creation of procedures.

* DESCRIPTION:

* A procedure is a block that can take parameters and be invoked.

* Procedures promote reusability and maintainability. Once Validated, they can be used in number of applications. If the definition changes, only the procedure are affected, this greatly Simplifies maintenance.

SYNTAX FOR PROCEDURES:-

```
Create [or Replace] PROCEDURES procedur_name
    (parameter1 [model1] datatype1,
    (parameter 2 [model2] datatype2,...)
    IS /AS PL/SQL BLOCK;
```

EX: NO: 9 DATE:	CREATION OF DATABASE TRIGGERS AND FUNCTIONS

AIM:

To Study and implement the Concept of triggers.

DEFINITION:

* A trigger is a Statement that is executed automatically by the System as a Side effect of a modification to the database. It Can either be,

1. Application trigger.
2. Database Trigger.

SYNTAX:

```
Create or replace trigger triggername [before/after]
        {DML statements}
    on [tablename] [for each row/statement]
begin
- - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - -
exception
end;
```

*FUNCTION:

* A function is a named PL/SQL Block which is Similar to a procedure. The major difference between a procedure and a function is, a function must always return a Value, but a procedure may or may not return a Value.

* GENERAL SYNTAX TO CREATE FUNCTION:-

```
CREATE [OR REPLACE] FUNCTION function_name [parameters]
RETURN return_datatype;
IS
Declaration_Section
BEGIN
Execution_Section
Return return_Variable;
EXCEPTION
exception Section.
Return return_Variable;
END;
```