**Database Management System**

**Practical No : 10(b)**

**Aim :**  Stored procedure & Triggers

**Stored Procedure**

A stored procedure in SQL is a type of pre-written code that can be stored for later execution and then used many times hence, saving time. It is a group of SQL statements that performs the task. The stored procedure can be invoked explicitly whenever required. It may accept some inputs in the form of parameters, these may be one parameter or multiple parameters.

* Procedures and Functions are the subprograms which can be created and saved in the database as database objects. They can be called or referred inside the other blocks also

Based on their purpose parameters are classified as

1. IN Parameter: This parameter is used for giving input to the subprograms. Their values cannot be changed inside the subprogram.
2. OUT Parameter: This parameter is used for getting output from the subprograms. Their values can be changed inside the subprograms.
3. IN OUT Parameter: This parameter is used for both giving input and for getting output from the subprograms. Their values can be changed inside the subprograms.

**Syntax : Creating a Procedure**

CREATE or REPLACE PROCEDURE name(parameters)

IS

Variables-declaration;

BEGIN

//statements;(execution part)

EXCEPTION

//Exception handling;

END;

**Examples 1:**

**CREATE OR REPLACE PROCEDURE** welcome\_msg (p\_name IN VARCHAR2)

**IS**

**BEGIN**

dbms\_output.put\_line (‘Welcome '|| p\_name);

**END;**

***Steps for Execution*:** EXEC welcome\_msg (‘Students’);

**Example 2:**  
Imagine a table named with emp\_table stored in Database. We are Writing a Procedure to update a Salary of Employee with 1000.

**CREATE or REPLACE PROCEDURE** INC\_SAL(eno IN NUMBER, up\_sal OUT NUMBER)

**IS**

**BEGIN**

UPDATE emp\_table SET salary = salary+1000 WHERE emp\_no = eno;

**COMMIT;**

SELECT sal INTO up\_sal FROM emp\_table WHERE emp\_no = eno;

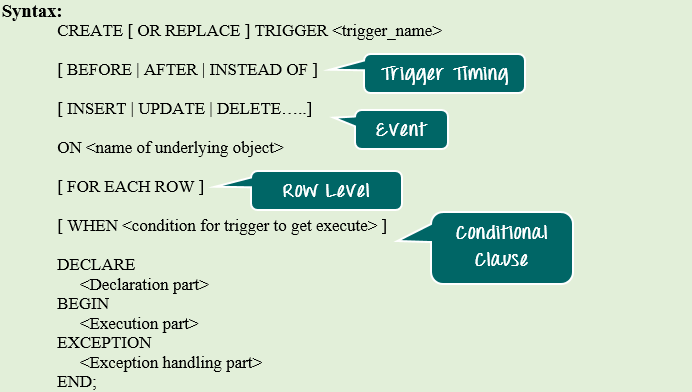
**END;**

***Steps to execute the procedure*:**

* Declare a Variable to Store the value comming out from Procedure :
  + VARIABLE v NUMBER;
* Execution of the Procedure:
  + EXECUTE INC\_SAL(1002, :v);
* To check the updated salary use SELECT statement:
  + SELECT \* FROM emp\_table WHERE emp\_no = 1002;
* Use print statement :
  + PRINT :v

**Trigger**

A trigger is a stored procedure in database which automatically invokes whenever a special event in the database occurs. For example, a trigger can be invoked when a row is inserted into a specified table or when certain table columns are being updated.



**Syntax Explanation:**

* The above syntax shows the different optional statements that are present in trigger creation.
* BEFORE/ AFTER will specify the event timings.
* INSERT/UPDATE/LOGON/CREATE/etc. will specify the event for which the trigger needs to be fired.
* ON clause will specify on which object the above-mentioned event is valid. For example, this will be the table name on which the DML event may occur in the case of DML Trigger.
* Command "FOR EACH ROW" will specify the ROW level trigger.
* WHEN clause will specify the additional condition in which the trigger needs to fire.
* The declaration part, execution part, exception handling part is same as that of the other PL/SQL blocks. Declaration part and exception handling part are optional.
* :NEW – It holds a new value for the columns of the base table/view during the trigger execution
* :OLD – It holds old value of the columns of the base table/view during the trigger execution

**BEFORE and AFTER of Trigger:**  
BEFORE triggers run the trigger action before the triggering statement is run.  
AFTER triggers run the trigger action after the triggering statement is run.

Example:

Consider a table Student(rollno, sub1,sub2,sub3,total)

**create trigger** stud\_marks

**before INSERT**

**on**

Student

**for each row**

set Student.total = Student.subj1 + Student.subj2 + Student.subj3;

**end;**

Above SQL statement will create a trigger in the student database in which whenever subjects marks are entered, before inserting this data into the database, trigger will compute total and the same is inserted in the table.

sql> insert into Student values(101, ‘A’, 20, 20, 20, 0);