**Vel Tech Rangarajan Dr. Sagunthala R&D Institute of Science and Technology**

**(Deemed to be University Estd. u/s 3 of UGC Act, 1956)**

**School of Computing**

**B.Tech. – Computer Science and Engineering**

**VTR UGE2021- (CBCS)**

Academic Year: 2025–2026

SUMMER SEMESTER - SS2526

Course Code : 10211CS207

Course Name: Database Management Systems

Slot No : S2L5

DBMS TASK - 7 REPORT

# Title: Triggers, Views, and Exceptions

**Submitted by:**

|  |  |  |
| --- | --- | --- |
| **VTUNO** | **REGISTER NUMBER** | **STUDENT NAME** |
| VTU27661 | 24UECS1159 | C Venugopal reddy |

# Task 7: Triggers, Views, and Exceptions

**Objective:**

To understand and implement **Triggers**, **Views**, and **Exception Handling** for performing and managing **CRUD (Create, Read, Update, Delete)** operations in an Oracle database.

# Part 1: Implementing Triggers

## 1. Prevent Insertion of Underage Students

CREATE TABLE Students (

StudentID NUMBER PRIMARY KEY,

Name VARCHAR2(50),

Age NUMBER,

Department VARCHAR2(50),

Marks NUMBER

);

## EXPECTED OUTPUT: Table created

CREATE OR REPLACE TRIGGER prevent\_underage\_students

BEFORE INSERT ON Students

FOR EACH ROW

BEGIN

IF :NEW.Age < 18 THEN

RAISE\_APPLICATION\_ERROR(-20001, 'Age must be 18 or above'); END IF;

END;

/

## EXPECTED OUTPUT: Trigger created

### 2. Create a Log Table

CREATE TABLE StudentLog (

LogID NUMBER PRIMARY KEY,

StudentID NUMBER,

ActionType VARCHAR2(20),

ActionDate TIMESTAMP DEFAULT SYSTIMESTAMP

);

## EXPECTED OUTPUT: Table created Part 2: Creating Views

### 1. View for Top Students

CREATE OR REPLACE VIEW View\_TopStudents AS

SELECT StudentID, Name, Marks

FROM Students

WHERE Marks > 80;

## EXPECTED OUTPUT: View created

### 2. View for Department Summary

CREATE OR REPLACE VIEW View\_DepartmentSummary AS

SELECT

Department,

COUNT(\*) AS TotalStudents,

ROUND(AVG(Marks), 2) AS AverageMarks

FROM Students

GROUP BY Department;

**EXPECTED OUTPUT: View created**

# Part 3: Exception Handling

**1. Stored Procedure with Exception Handling for Inserting Student Records**

CREATE OR REPLACE PROCEDURE InsertStudent (

p\_StudentID IN NUMBER, p\_Name IN VARCHAR2, p\_Age IN NUMBER, p\_Department IN VARCHAR2,

p\_Marks IN NUMBER

)

IS

BEGIN

INSERT INTO Students (StudentID, Name, Age, Department, Marks)

VALUES (p\_StudentID, p\_Name, p\_Age, p\_Department, p\_Marks);

DBMS\_OUTPUT.PUT\_LINE('Record Inserted Successfully');

EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Error: ' || SQLERRM);

END; /

## EXPECTED OUTPUT: Procedure created

### 2. Function to Fetch Student Details with Error Handling

CREATE OR REPLACE FUNCTION GetStudentDetails (p\_StudentID IN NUMBER)

RETURN VARCHAR2

IS

student\_info VARCHAR2(255);

BEGIN

SELECT 'Name: ' || Name || ', Age: ' || Age || ', Department: ' || Department || ',

Marks: ' || Marks

INTO student\_info

FROM Students

WHERE StudentID = p\_StudentID;

RETURN student\_info;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN 'Student Not Found';

WHEN OTHERS THEN

RETURN 'Error: ' || SQLERRM;

END; /

## EXPECTED OUTPUT: Function created

**RESULT:**Thus the program has been executed and verified sucessfully.