Vidyavardhini's College of Engineering and Technology, Vasai
Department of Artificial Intelligence & Data Science

AY: 2024-25

Class:	**SE**	**Semester:**	**IV**
Course Code:	**CSL402**	**Course Name:**	**DBMS Lab**

Name of Student:	**Shravani Sandeep Raut**
Roll No. :	**48**
Experiment No.:	**8**
Title of the Experiment:	**Implementation of Views and Triggers**
Date of Performance:	**02/04/2025**
Date of Submission:	**02/04/2025**

Evaluation

Performance Indicator	Max. Marks	Marks Obtained
Performance	5	
Understanding	5	
Journal work and timely submission	10	
Total	20	

Performance Indicator	Exceed Expectations (EE)	Meet Expectations (ME)	Below Expectations (BE)
Performance	4-5	2-3	1
Understanding	4-5	2-3	1
Journal work and timely submission	8-10	5-8	1-4

Checked by

Name of Faculty : Ms. Neha Raut

Signature :

Date:

Aim :- Write a SQL query to implement views and triggers

Objective :- To learn about virtual tables in the database and also PLSQL constructs

Theory:

SQL Views:

In SQL, a view is a virtual table based on the result-set of an SQL statement.

A view contains rows and columns, just like a real table. The fields in a view are fields from one or more real tables in the database.
You can add SQL statements and functions to a view and present the data as if the data were coming from one single table.
A view is created with the CREATE VIEW statement.

CREATE VIEW Syntax

CREATE VIEW view_name AS

SELECT column1, column2, ...

FROM table_name

WHERE condition;

SQL Updating a View

A view can be updated with the CREATE OR REPLACE VIEW statement.

SQL CREATE OR REPLACE VIEW Syntax

CREATE OR REPLACE VIEW view_name AS

SELECT column1, column2, ...

FROM table_name

WHERE condition;

SQL Dropping a View

A view is deleted with the DROP VIEW statement.

SQL DROP VIEW Syntax

DROP VIEW view_name;

Trigger: A trigger is a stored procedure in the 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:

create trigger [trigger_name]

[before | after]

{insert | update | delete}

on [table_name]

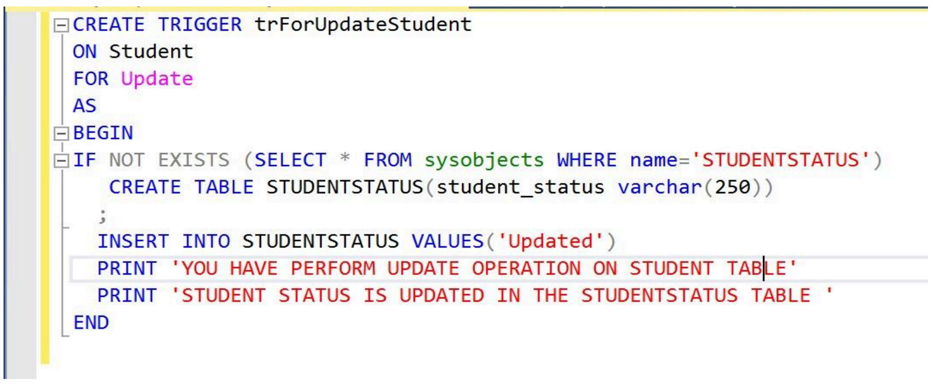
[for each row]

[trigger_body]

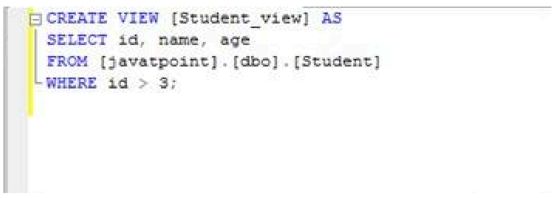
Explanation of syntax:

1. create trigger [trigger_name]: Creates or replaces an existing trigger with the trigger_name.
2. [before | after]: This specifies when the trigger will be executed.
3. {insert | update | delete}: This specifies the DML operation.
4. on [table_name]: This specifies the name of the table associated with the trigger.
5. [for each row]: This specifies a row-level trigger, i.e., the trigger will be executed for each row being affected.
6. [trigger_body]: This provides the operation to be performed as trigger is fired

Implementation:



```
CREATE TRIGGER trForUpdateStudent
ON Student
FOR Update
AS
BEGIN
IF NOT EXISTS (SELECT * FROM sysobjects WHERE name='STUDENTSTATUS')
    CREATE TABLE STUDENTSTATUS(student_status varchar(250))
   ;
   INSERT INTO STUDENTSTATUS VALUES('Updated')
   PRINT 'YOU HAVE PERFORM UPDATE OPERATION ON STUDENT TABLE'
   PRINT 'STUDENT STATUS IS UPDATED IN THE STUDENTSTATUS TABLE '
END
```



```
CREATE VIEW [Student_view] AS
SELECT id, name, age
FROM [javatpoint].[dbo].[Student]
WHERE id > 3;
```

Conclusion:

A) Brief about the benefits for using views and triggers.

Views and triggers offer several benefits in SQL:

- **Views** simplify complex queries by storing them as virtual tables. They enhance security by restricting column access and provide data abstraction.

- **Triggers** automate actions in response to events like `INSERT`, `UPDATE`, or `DELETE`, ensuring consistent business logic and data integrity.

Benefits Summary:

- **Views:** Simplify queries, restrict access, enable reuse
- **Triggers:** Automate tasks, enforce rules, maintain consistency

B) Explain different strategies to update views

Different Strategies to Update Views in SQL:

1. **Direct Update (Updatable Views):**
   If a view is created from a single table without joins, group by, or aggregation, it can be updated directly:

   UPDATE view_name SET column = value WHERE condition;
2. **INSTEAD OF Triggers:**
   When views involve joins or aggregations (non-updatable), `INSTEAD OF` triggers can be used to define how updates should happen:

   CREATE TRIGGER trigger_name INSTEAD OF UPDATE ON view_name

   FOR EACH ROW BEGIN

   -- custom update logic

   END;
3. **Re-Creation of View:**
   If the base table changes, the view can be dropped and recreated with updated logic.