Experiment No.8
Implementation of Views and Triggers.
Date of Performance:
Date of Submission:

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 columni, 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 columnl, 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:

SQL View:

1)Create View:

```
CREATE VIEW MOVIE_INFO AS
SELECT MOVIE_ID, MOVIE_NAME, MOVIE_DATE, MOVIE_PRICE
FROM MOVIE;
SELECT * FROM MOVIE_INFO;
```

#	Time	Action	Message	Duration / Fetch
1	17:53:25	CREATE VIEW MOVIE_INFO AS SELECT MOVIE_ID, ...	0 row(s) affected	0.109 sec
2	17:53:25	SELECT * FROM MOVIE_INFO LIMIT 0, 1000	7 row(s) returned	0.015 sec / 0.000 sec

```
296 •   SELECT * FROM MOVIE_INFO;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

MOVIE_ID	MOVIE_NAME	MOVIE_DATE	MOVIE_PRICE
1	ROWDY_RATHOD	12/3/2025	1200
2	HANUMan	14/3/2025	100
3	TARE_ZAMEEN_PAR	16/3/2025	200
4	Baby	20/3/2025	200
5	Prem_Ratan_Dhan_Payo	25/3/2025	250
6	khiladi	2/4/2025	300
7	Baahubali:_The_Beginning	12/4/2025	120

2)Drop View:

```
•   DROP VIEW IF EXISTS MOVIE_INFO;
•   SELECT * FROM MOVIE_INFO;
```

#	Time	Action	Message	Duration / Fetch
3	17:58:58	DROP VIEW IF EXISTS MOVIE_INFO	0 row(s) affected	0.031 sec
4	17:58:58	SELECT * FROM MOVIE_INFO LIMIT 0, 1000	Error Code: 1146. Table 'online_movie_ticket_booking....	0.015 sec

SQL Trigger:

```
-- CREATE TRIGGER TO UPDATE LAST_UPDATE
DELIMITER //
CREATE TRIGGER CAPITIALIZE_NAME_BEFORE_INSERT
BEFORE INSERT ON MOVIE
FOR EACH ROW
BEGIN
      SET NEW.MOVIE_NAME = CONCAT(UPPER(SUBSTRING(NEW.MOVIE_NAME, 1, 1)), LOWER(SUBSTRING(NEW.MOVIE_NAME, 2)));
END;
// DELIMITER ;
INSERT INTO MOVIE (MOVIE_ID, MOVIE_NAME, MOVIE_DATE, MOVIE_TIME, MOVIE_DETAILS, MOVIE_PRICE)
VALUES(8, 'SHAKTIMAN', '12/2/2024', '2PM', 'ACTION/DRAMA', 500 );
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

MOVIE_ID	MOVIE_NAME	MOVIE_DATE	MOVIE_TIME	MOVIE_DETAILS	MOVIE_PRICE
3	TARE_ZAMEEN_PAR	16/3/2025	2:00PM	comedy	200
4	Baby	20/3/2025	3:00PM	action/thriller	200
5	Prem_Ratan_Dhan_Payo	25/3/2025	5:00PM	drama	250
6	khiladi	2/4/2025	12:00PM	action/thriller	300
7	Baahubali:_The_Beginning	12/4/2025	2:00PM	action/thriller	120
8	Shaktiman	12/2/2024	2PM	ACTION/DRAMA	500
NULL	NULL	NULL	NULL	NULL	NULL

Conclusion:

1. Brief about the benefits for using views and triggers.

Ans.: Views simplify queries, enhance security, abstract table structures, and optimize performance. Triggers enforce data integrity, audit changes, enforce business logic, and support replication.

2. Explain different strategies to update views.

Ans.: Updating views can be done directly, by updating base tables, using triggers, or by recreating views. These methods offer varying degrees of control and are applied based on the view's complexity and update requirements.