Experiment No.8

Implementation of Views and Triggers

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

Conclusion:

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

➔ **Benefits of Using Views**:

a **Reusability**: Views allow easy access to frequently used queries without rewriting them.
b **Complexity**: Views can encapsulate complex business logic, formulas, and joins.
c **Business Naming**: Rename columns for business purposes.
d **Security**: Limit visibility by providing views instead of direct table access.

e **Reference and Documentation**: Views serve as data diagram references and documentation.

2. Explain different strategies to update views

➔ **Strategies to Update Views**:

- To update views, consider the following rules:
  - The view should not include `GROUP BY` or `ORDER BY` clauses.
  - The view should not use the `DISTINCT` keyword.
  - The view should have all NOT NULL values.
  - The view should not be created using nested or complex queries.
  - The view should be created from a single table