

**Experiment No.7**

**Title: Creation of view and trigger on database.**

**Batch: B-1** **Roll No.: 16010422234 Name: Chandana Ramesh Galgali**

**Experiment No.: 7**

**Title:** To implement View and Triggers for a given database.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Resources needed:** PostgreSQL 9.3

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Theory**

**Pre Lab/ Prior Concepts:**

**Views:**

Views are virtual tables created from already existing tables by selecting certain columns or certain rows. A view can be created from one or many tables. View allows to:

1. Restrict access to the data such that a user can only see limited data instead of the complete table.
2. Summarize data from various tables which can be used to generate reports.

In PostgreSQL, Views are created using the CREATE VIEW statement given below.

CREATE [TEMP | TEMPORARY] VIEW view\_name AS SELECT column1, column2..... FROM table\_name WHERE [condition];

For example,

Consider COMPANY table having following records:

id| name | age | address | salary

-+--------+-----+------------+--------

1 | Paul | 32 | California | 20000

2 | Allen | 25 | Texas | 15000

3 | Teddy | 23 | Norway | 20000

4 | Mark | 25 | Rich-Mond | 65000

5 | David | 27 | Texas | 85000

6 | Kim | 22 | South-Hall | 45000

7 | James | 24 | Houston | 10000

Following statement creates a view from the COMPANY table.

CREATE VIEW COMPANY\_VIEW AS SELECT ID, NAME, AGE FROM COMPANY;

Now, query can be written on COMPANY\_VIEW in similar way as that of an actual table, as shown below,

SELECT \* FROM COMPANY\_VIEW;

This would produce the following result:

Views can be dropped using the “DROP VIEW” statement.

**Trigger:**

A trigger is a statement that the system executes automatically as a side effect of a modification to the database. Triggers are used to ensure some types of integrity.

To design a trigger mechanism you should:

1. Specify when a trigger is to be executed. This is broken up into an event that causes a trigger to be checked and a condition that must be satisfied for trigger execution to proceed

2. Specify the actions to be taken when the trigger executes.

**Generalized Model:**

Triggers are based on the Event-Condition-Action (ECA) Model. A rule in the ECA model has three components:

1. The event(s) that triggers the rule.

2. The condition that determines whether the rule action should be executed. If no action is specified, the action will be executed once the event occurs.

3. The action to be taken. It could be a sequence of SQL statements, a DB transaction or an external program that will be executed automatically.

**When to use Trigger:**

In many cases, it is convenient to specify the type of action to be taken when a certain event occurs and when certain conditions are satisfied.

It may be useful to specify a condition that, if violated, causes some user to be informed of the violation.

For example:-

A manager may want to be informed, if an employee’s travel expenses exceed a certain limit by receiving a message whenever this occurs.

The condition is thus used to monitor the database.

CREATE TRIGGER statement is used to implement such action in SQL. Consider the triggers for following cases:-

1. Trigger for insertion:-

This trigger executes whenever the condition is satisfied, during the insertion statement. If no condition is satisfied, then it is executed for every insertion statement, for the relation specified.

Example:-

In DB, we have created a trigger for insertion, on the relation Employee. If salaries < 1500 then print ‘Unsuccessful’ else print ‘Successful’.

2. Trigger for Updation:-

This trigger executes for updating of a column name(s) of a particular relation. A condition may or may not be specified.

Example:-

In our DB, we have created a trigger for updating, on the relation Employee. If salary is updated to an unacceptable amount, then print unsuccessful and rollback else print successful.

3. Trigger for Deletion:-

This trigger executes during the deletion statement. If a particular condition is satisfied, the system may allow or not allow the deletion of certain tuples,

Example:-

If SSN from Employee =101, then print ‘Unsuccessful’ and rollback, else print ‘Successful’.

In Postgresql we need to create a function to execute all action statements of triggers and call this function to create a trigger statement for a certain event.

Example.

In the employee table dnum is a foreign key. So the trigger written here will increment total\_emp of department table by one and total\_sal of department by salary of newly inserted employee's salary.

CREATE or replace FUNCTION inse\_function() RETURNS trigger As $emp\_update$

begin;

UPDATE dept set total\_emp=total\_emp + 1,

total\_sal=total\_sal+ new.salary where dept.dno=new.dno;

RETURN new;

END

$emp\_update$ LANGUAGE plpgsql;

CREATE TRIGGER emp\_update AFTER INSERT

ON emp\_table FOR each ROW

EXECUTE PROCEDURE inse\_function();

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Procedure / Approach /Algorithm / Activity Diagram**

**View**

1. Create views from existing tables

Execute SELECT, UPDATE, INSERT statements on views and original table.

**Trigger**

1. Create new database for your application

2. Apply required integrity constraints on tables in your database

**To design a trigger:**

1. Identify events in the database.
2. Specify conditions under which the trigger is to be executed.
3. Specify actions to be taken when trigger is executed.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Results: (Program printout with output / Document printout as per the format)**

CREATE VIEW Flights\_View AS

SELECT Flight\_ID, Arrival\_Time, Departure\_Time, Arrival\_Location, Destination, Aircraft\_Used

FROM Flights;

CREATE VIEW Passenger\_View AS

SELECT Passenger\_ID, Name, Email, Address

FROM Passenger;

SELECT \* FROM Flights\_View;

SELECT \* FROM Passenger\_View;

SELECT \* FROM Flights;

SELECT \* FROM Passenger;

UPDATE Flights\_View SET Arrival\_Time = '12:00' WHERE Flight\_ID = 1;

UPDATE Flights SET Arrival\_Time = '12:00' WHERE Flight\_ID = 1;

UPDATE Passenger\_View SET Address = '456 Elm St' WHERE Passenger\_ID = 1;

UPDATE Passenger SET Address = '456 Elm St' WHERE Passenger\_ID = 1;

CREATE OR REPLACE FUNCTION update\_ticket\_price()

RETURNS TRIGGER AS $$

BEGIN

IF NEW.Passenger\_ID > 5 THEN

UPDATE Tickets

SET Price = Price + 50

WHERE Passenger\_ID = NEW.Passenger\_ID;

END IF;

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

CREATE TRIGGER after\_insert\_Passenger

AFTER INSERT ON Passenger

FOR EACH ROW

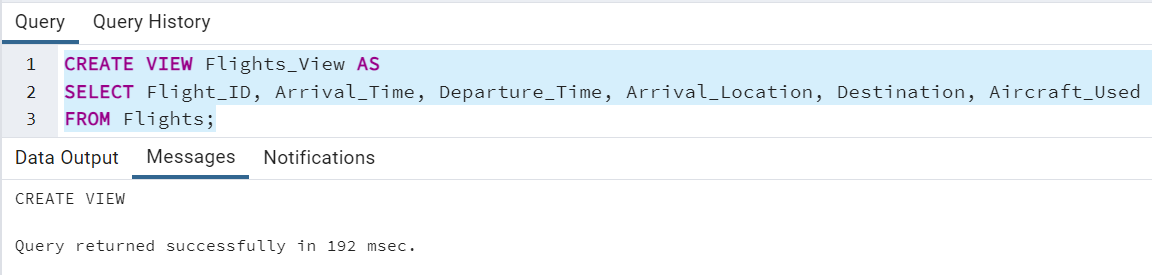
EXECUTE FUNCTION update\_ticket\_price();

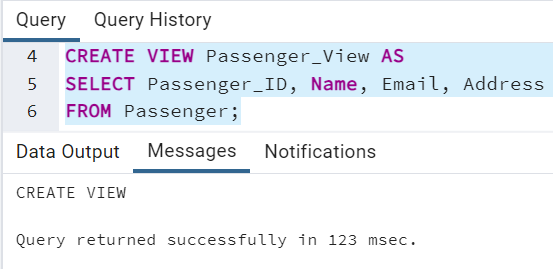
INSERT INTO Passenger VALUES (6, 'pass123', 'Mary Jane', '1990-05-15', 'janedoe.3', '789 Oak St', 1237777890, 'Savings', 'SPM Bank', 'Yes');

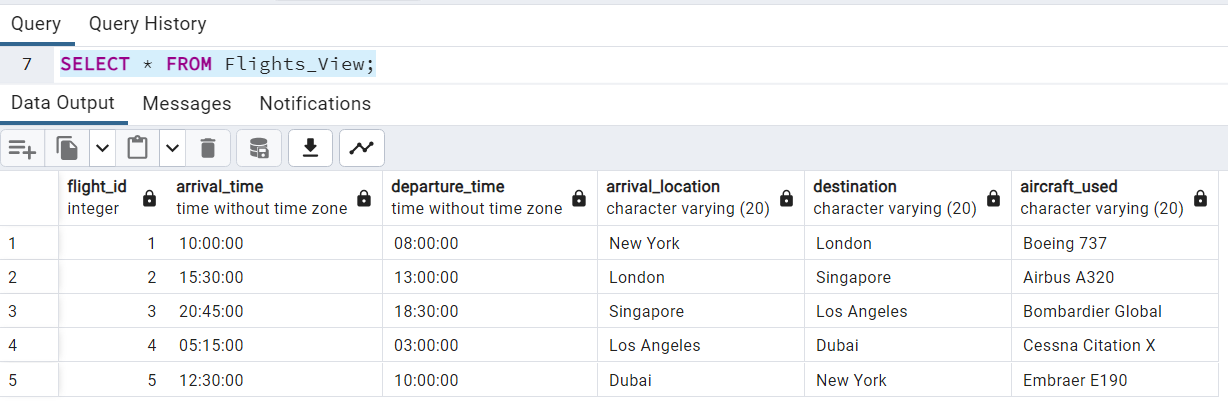
SELECT \* FROM Passenger;

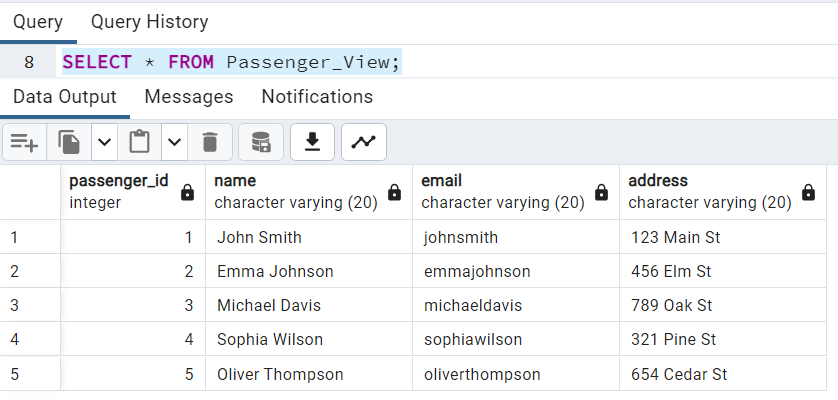
INSERT INTO Tickets VALUES (6, 6, 654, 60, 'Business', 650);

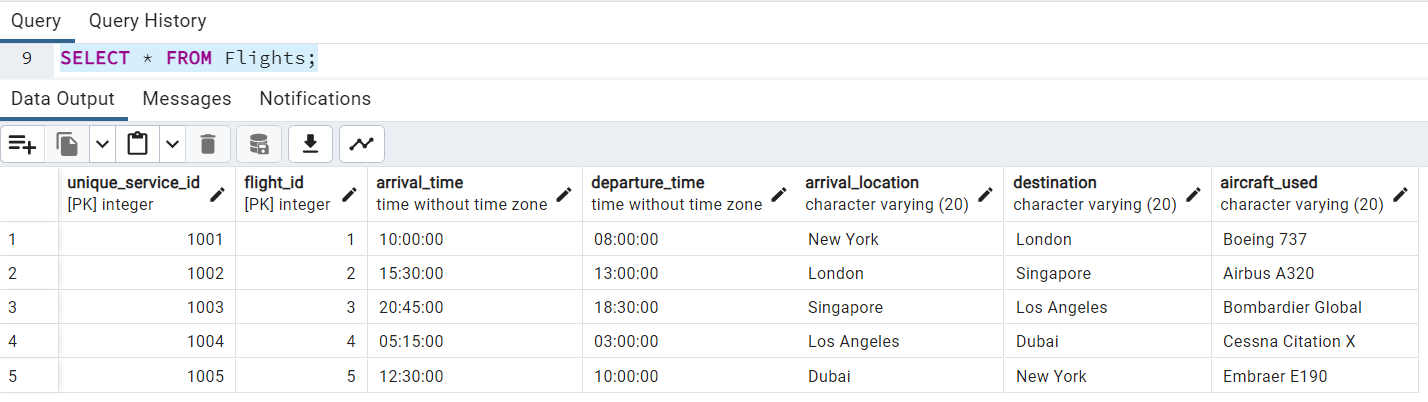
SELECT \* FROM Tickets;

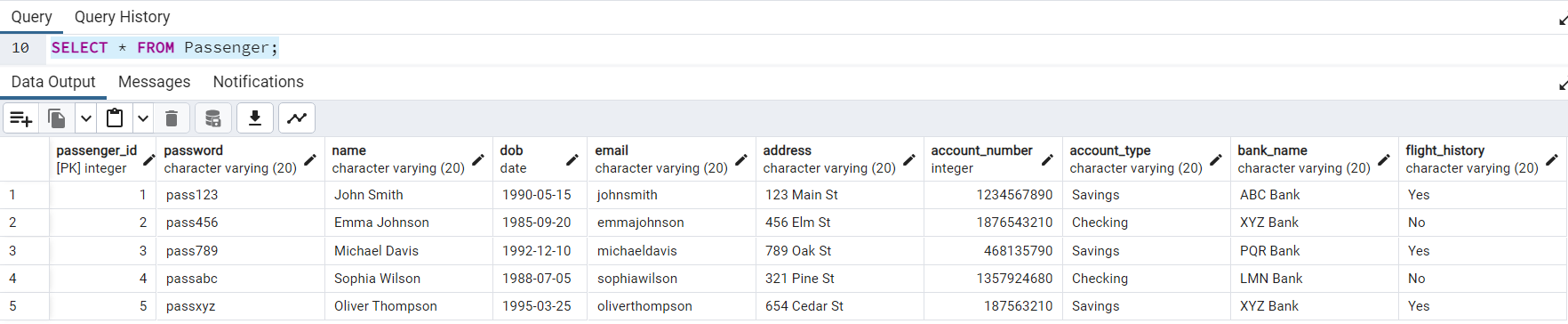
****

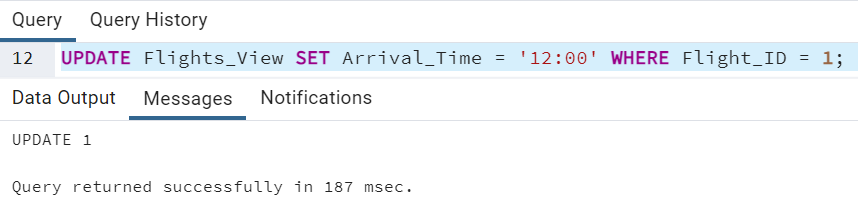
****

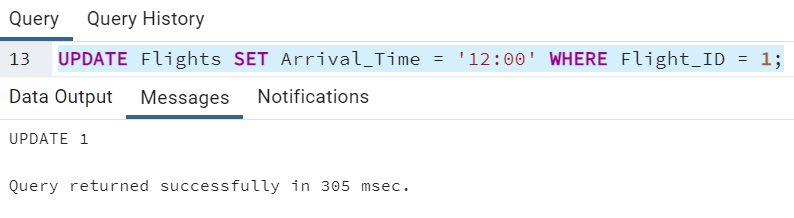
****

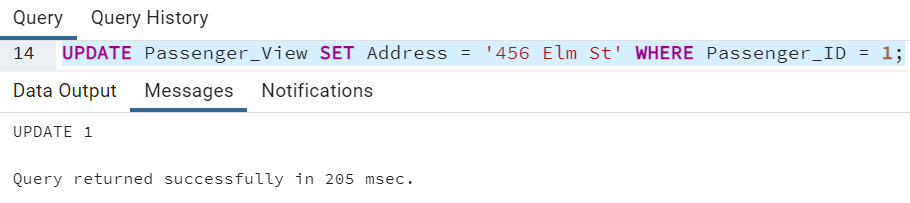
****

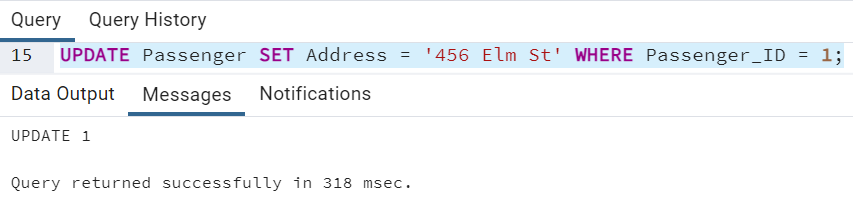
****

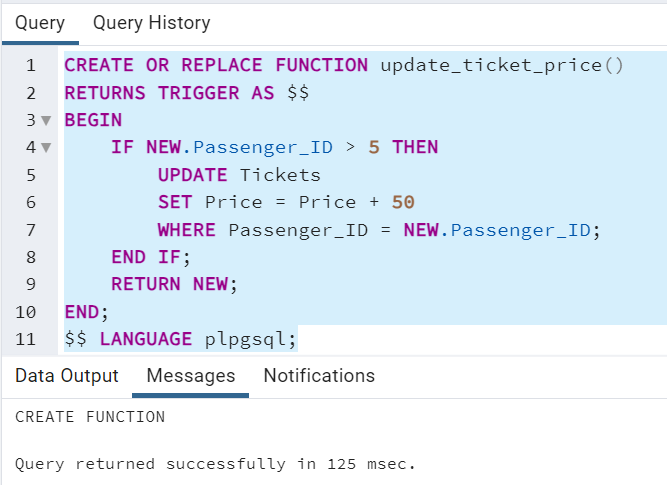
****

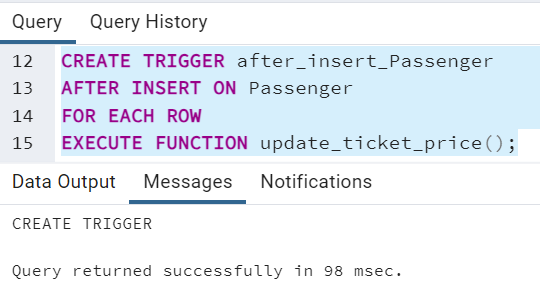
****

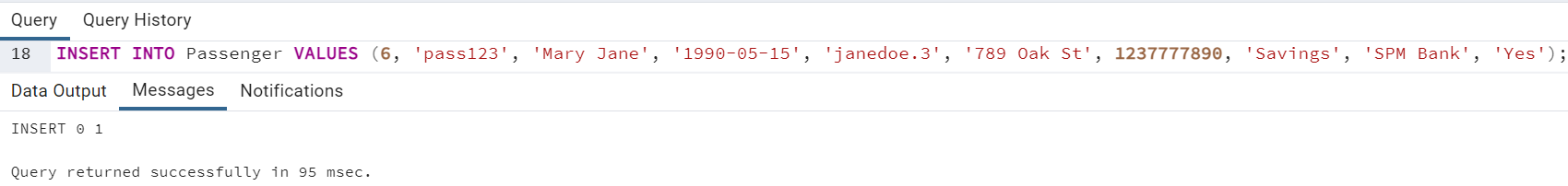
****

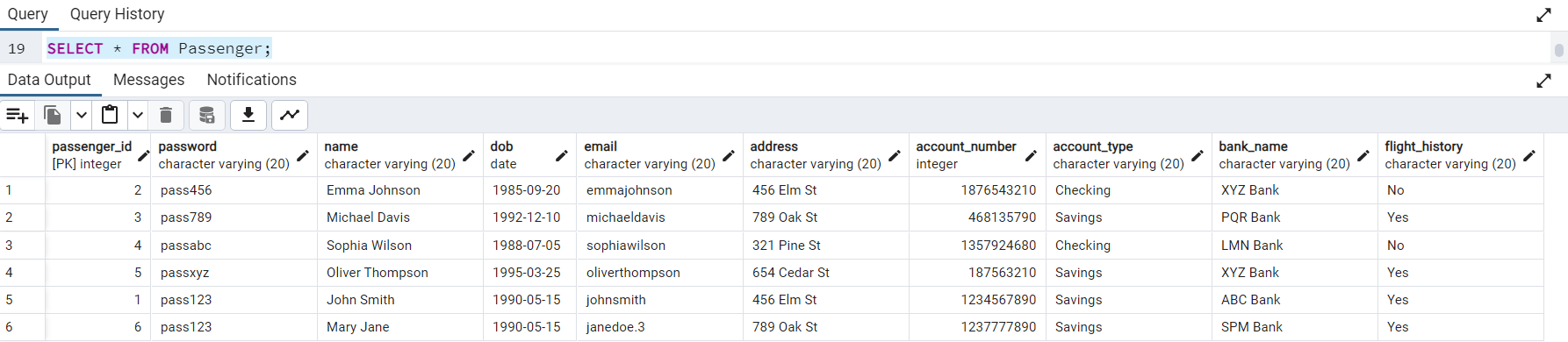
****

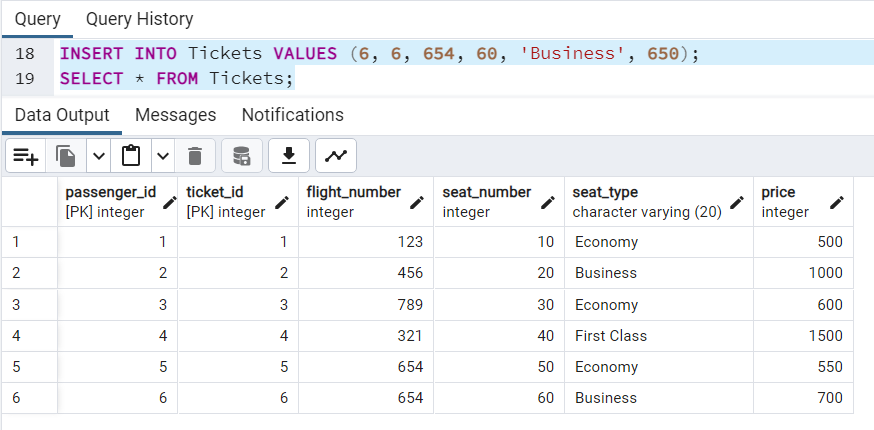
****

****

****

****

****

****

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Questions:**

**1) Explain any one real-time application of triggers.**

**Ans:** One real-time application of triggers is in database management systems. Triggers are used to automatically execute a set of actions or procedures when a specific event occurs in the database. For example, a trigger can be set up to update a table whenever a new record is inserted, or to enforce certain business rules or constraints.

One specific application of triggers is in auditing and logging changes to a database. By creating triggers on tables, you can capture and record any modifications made to the data, such as inserts, updates, or deletions. This can be useful for maintaining an audit trail, tracking changes made by users, and ensuring data integrity.

For instance, let's say you have a customer database for an e-commerce website. You can create a trigger that automatically logs any changes made to the customer's information, such as updates to their address or contact details. This trigger can capture the old and new values of the modified data and store them in a separate audit table. This way, you can keep track of all the changes made to the customer records, which can be valuable for troubleshooting, compliance, or analysis purposes.

Overall, triggers provide a powerful mechanism for automating actions and maintaining data consistency in real-time applications, such as database management systems.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Outcomes: Apply data models to real world scenarios.**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Conclusion: (Conclusion to be based on the objectives and outcomes achieved)**

The experiment successfully demonstrated the practical implementation of Views and Triggers in a database environment. The outcomes achieved include improved data security, simplified data access, enhanced data integrity, and automation of certain tasks. These results validate the effectiveness and usefulness of Views and Triggers in managing and manipulating data within a database system.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Grade: AA / AB / BB / BC / CC / CD /DD**

**Signature of faculty in-charge with date**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**References:**

**Books/ Journals/ Websites:**

1. Korth, Slberchatz,Sudarshan, :”Database System Concepts”, 6th Edition, McGraw – Hill

2. Elmasri and Navathe, “ Fundamentals of Database Systems”, 5thEdition, PEARSON Education