

# ADVANCED DATABASE MANAGEMENT SYSTEMS

## LAB

Submitted by,  
Tony Joseph  
RMCA S2, B  
Roll. No: 37

## **AIM**

Create a Trigger for employee table it will update another table salary while updating values

## **OBJECTIVE**

To develop and execute a Trigger for After update/Delete/Insert operations on a table

## **PROCEDURE**

step 1: start

step 2: initialize the trigger.

step 3: On update the trigger has to be executed.

step 4: execute the trigger procedure after updation

step 5: carryout the operation on the table to check for trigger execution.

step 6: stop

## **PROGRAM**

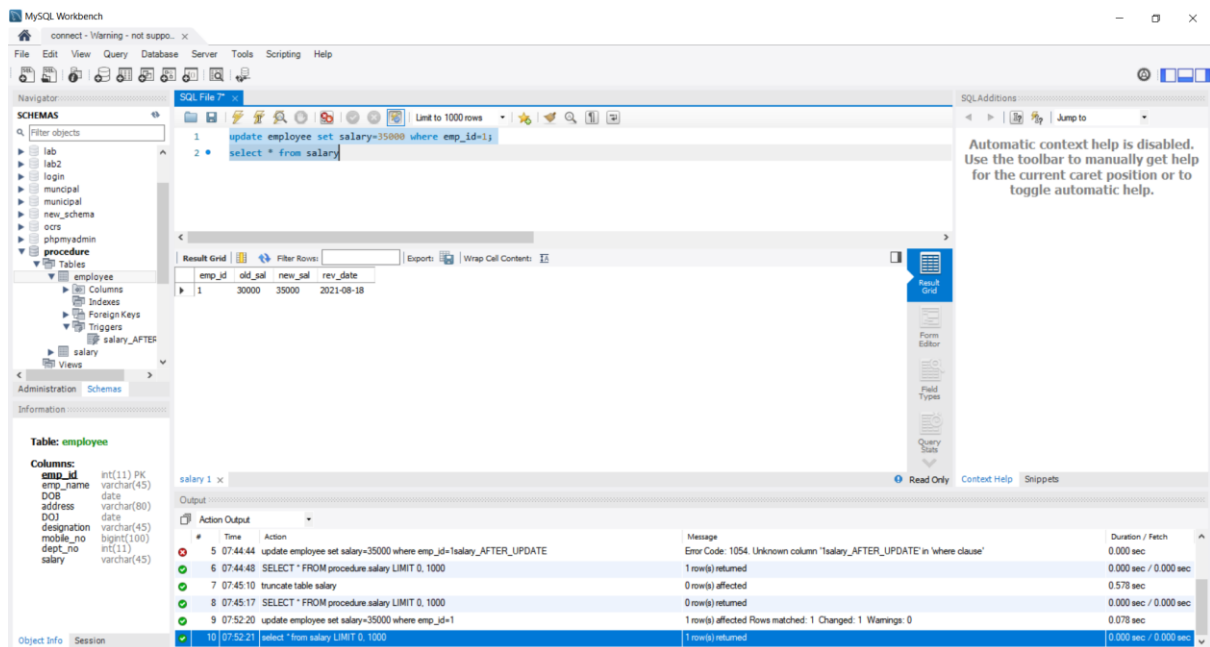
```
CREATE TABLE `employee` (  
  `emp_id` int(11) NOT NULL,  
  `emp_name` varchar(45) DEFAULT NULL,  
  `DOB` date DEFAULT NULL,  
  `address` varchar(80) DEFAULT NULL,  
  `DOJ` date DEFAULT NULL,  
  `designation` varchar(45) DEFAULT NULL,  
  `mobile_no` bigint(100) DEFAULT NULL,  
  `dept_no` int(11) DEFAULT NULL,  
  `salary` varchar(45) DEFAULT NULL,  
  PRIMARY KEY (`emp_id`)  
)
```

```
CREATE TABLE `salary` (  
  `emp_id` int(11) DEFAULT NULL,  
  `old_sal` varchar(45) DEFAULT NULL,  
  `new_sal` varchar(45) DEFAULT NULL,  
  `rev_date` date DEFAULT NULL  
)
```

```
REATE DEFINER='root'@'localhost' TRIGGER `procedure`.`salary_AFTER_UPDATE` AFTER UPDATE ON  
`employee` FOR EACH ROW  
BEGIN  
if(new.salary != old.salary)  
then  
insert into salary (emp_id,old_sal,new_sal,rev_date) values  
(new.emp_id,old.salary,new.salary,sysdate());
```

end if;  
END

update employee set salary=35000 where emp\_id=1;  
select \* from salary



## AIM

Create a Trigger for employee table it will update another table personal\_updates while updating values

## OBJECTIVE

To develop and execute a Trigger for Before and After update/Delete/Insert operations on a table

## PROCEDURE

- step 1: start
- step 2: initialize the trigger.
- step 3: On update the trigger has to be executed.
- step 4: execute the trigger procedure after updation
- step 5: carryout the operation on the table to check for trigger execution.
- step 6: stop

## PROGRAM

```
CREATE TABLE `personal` (  
  `emp_id` int(11) DEFAULT NULL,  
  `old_phoneno` bigint(100) DEFAULT NULL,  
  `new_phoneno` bigint(100) DEFAULT NULL,  
  `rev_date` date DEFAULT NULL)
```

```

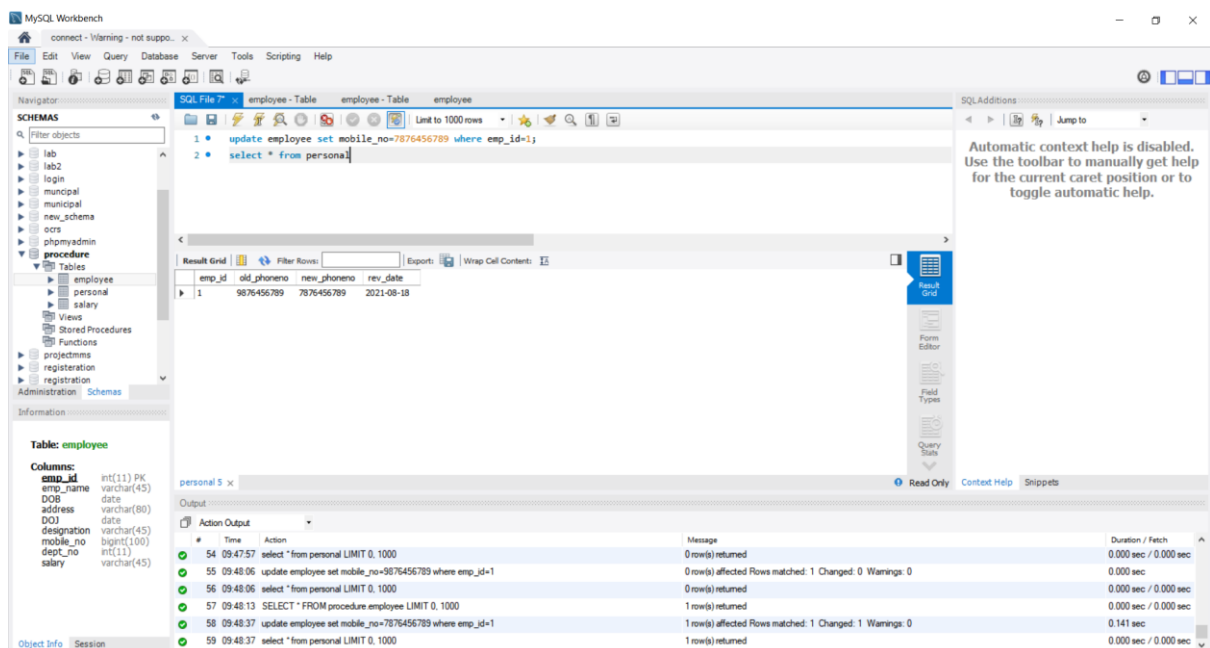
CREATE DEFINER=`root`@`localhost` TRIGGER `procedure`.`personal_AFTER_UPDATE` AFTER UPDATE
ON `employee` FOR EACH ROW
BEGIN
if(new.mobile_no != old.mobile_no)
then
insert into
personal(emp_id,old_phoneno,new_phoneno,rev_date)values(new.emp_id,old.mobile_no,new.mo
bile_no,sysdate());
end if;
END

```

```

update employee set mobile_no=7876456789 where emp_id=1;
select * from personal

```



## AIM

Create a Trigger for employee table it will update another table promotions while updating values

## OBJECTIVE

To develop and execute a Trigger for Before and After update/Delete/Insert operations on a table

## PROCEDURE

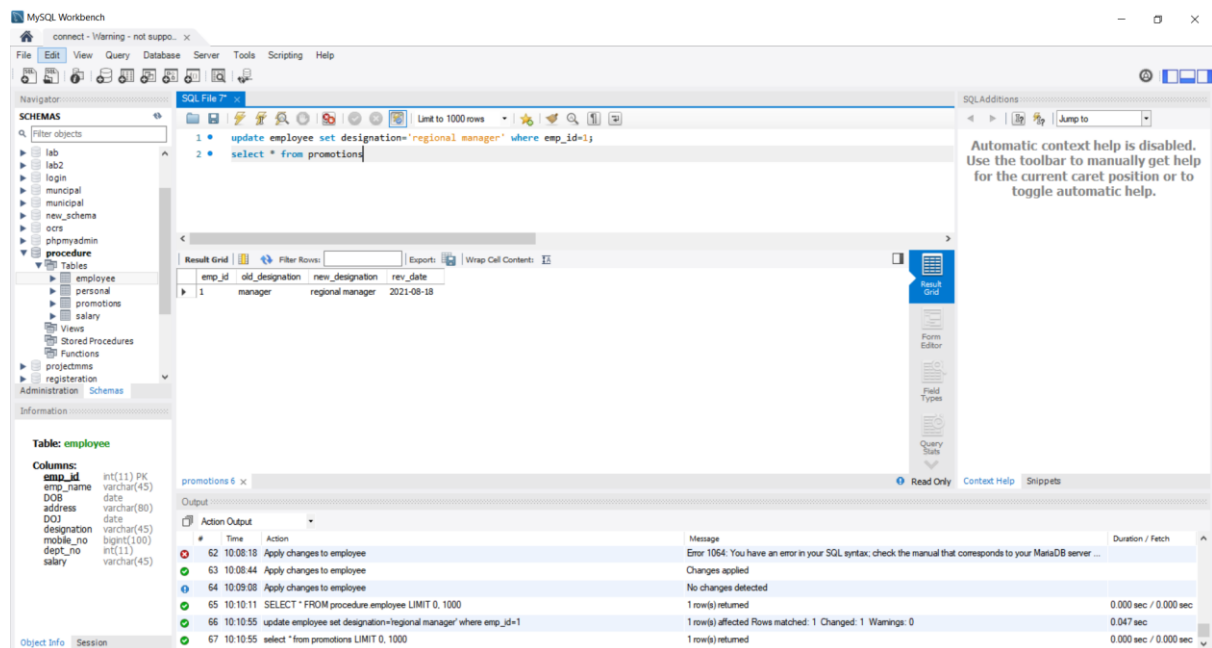
- step 1: start
- step 2: initialize the trigger.
- step 3: On update the trigger has to be executed.
- step 4: execute the trigger procedure after updation
- step 5: carryout the operation on the table to check for trigger execution.
- step 6: stop

## PROGRAM

```
CREATE TABLE `promotions` (  
  `emp_id` int(11) DEFAULT NULL,  
  `old_designation` varchar(45) DEFAULT NULL,  
  `new_designation` varchar(45) DEFAULT NULL,  
  `rev_date` date DEFAULT NULL  
)
```

```
CREATE DEFINER='root'@'localhost' TRIGGER `procedure`.`employee_AFTER_UPDATE1` AFTER  
UPDATE ON `employee` FOR EACH ROW  
BEGIN  
if(new.designation != old.designation)  
then  
INSERT INTO promotions (emp_id,old_designation,new_designation,rev_date) values  
(new.emp_id,old.designation,new.designation,sysdate());  
end if;  
END
```

```
update employee set designation='regional manager' where emp_id=1;  
select * from promotions
```



The screenshot displays the MySQL Workbench interface. The SQL editor contains the following queries:

```
1 • update employee set designation='regional manager' where emp_id=1;  
2 • select * from promotions
```

The Results grid shows the output of the second query:

emp_id	old_designation	new_designation	rev_date
1	manager	regional manager	2021-08-18

The Action Output pane shows the execution log:

#	Time	Action	Message	Duration / Fetch
62	10:08:18	Apply changes to employee	Error 1064: You have an error in your SQL syntax; check the manual that corresponds to your MariaDB server ...	
63	10:08:44	Apply changes to employee	Changes applied	
64	10:09:08	Apply changes to employee	No changes detected	
65	10:10:11	SELECT * FROM procedure.employee LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec
66	10:10:55	update employee set designation='regional manager' where emp_id=1	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.047 sec
67	10:10:55	select * from promotions LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec