
ADVANCED DBMS LAB

TRIGGER

Submitted By:

Jobin Joseph

S2 RMCA-B

Roll no:01

AIM

Create a Trigger for employe 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

sql>

```
CREATE TABLE `employe` (  
  `emp_id` int(11) NOT NULL,  
  `emp_name` varchar(45) DEFAULT NULL,  
  `dob` date DEFAULT NULL,  
  `address` varchar(45) DEFAULT NULL,  
  `designation` varchar(45) DEFAULT NULL,  
  `mobile_no` int(11) DEFAULT NULL,  
  `dept_no` int(11) DEFAULT NULL,  
  `salary` int(11) DEFAULT NULL,  
  PRIMARY KEY (`emp_id`)  
);
```

Sql>

```
CREATE TABLE `salary` (  
  `employee_id` int(11) NOT NULL,  
  `old_sal` int(11) DEFAULT NULL,  
  `new_sal` int(11) DEFAULT NULL,  
  `rev_date` date DEFAULT NULL,  
  PRIMARY KEY (`employee_id`)  
);
```

sql>

```
CREATE DEFINER=`root`@`localhost` TRIGGER  
`db1`.`personal_updates_AFTER_UPDATE_1`  
AFTER UPDATE ON `employe`  
FOR EACH ROW  
BEGIN  
if(new.salary != old.salary)  
then
```

```
INSERT INTO salary (employee_id,old_sal,new_sal,rev_date) values
(new.emp_id,old.salary,new.salary,sysdate());
END if;
end;
```

sql>

```
update employee set salary=234569 where emp_id=1;
select * from salary;
```

The screenshot displays the MySQL Workbench environment. The top toolbar includes icons for file operations, database management, and execution. The main editor window contains a SQL script with two statements: an update and a select. The 'Results' panel at the bottom shows the output of the select statement as a table with four columns: employee_id, old_sal, new_sal, and rev_date. The 'Output' panel below the results shows the execution log, including the time, action, message, and duration for each step.

SQL Script:

```
1 update employee set salary=10000 where emp_id=1;
2 select * from salary;
```

Results:

employee_id	old_sal	new_sal	rev_date
1	10000	10000	2021-08-17
2	150001	10000	2021-08-17
...

Execution Log:

#	Time	Action	Message	Duration / Fetch
136	17:01:52	Apply changes to employee		
137	17:02:02	Apply changes to employee	Error 1359: Trigger 'db1.employee_AFTER_UPDATE_1' already exists SQL Statement: CREATE DEFINER='ro...	
138	17:16:52	SELECT * FROM db1.employee LIMIT 0, 1000	4 row(s) returned	0.000 sec / 0.000 sec
139	17:17:19	ANALYZE TABLE 'db1'.employee	OK	0.000 sec
140	17:22:38	update employee set salary=1290 where emp_id=1	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.000 sec

AIM

Create a Trigger for employe 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

sql>

```
CREATE TABLE `employe` (  
  `emp_id` int(11) NOT NULL,  
  `emp_name` varchar(45) DEFAULT NULL,  
  `dob` date DEFAULT NULL,  
  `address` varchar(45) DEFAULT NULL,  
  `designation` varchar(45) DEFAULT NULL,  
  `mobile_no` int(11) DEFAULT NULL,  
  `dept_no` int(11) DEFAULT NULL,  
  `salary` int(11) DEFAULT NULL,  
  PRIMARY KEY (`emp_id`)  
);
```

Sql>

```
CREATE TABLE `personal_updates` (  
  `emp_id` int(11) NOT NULL,  
  `old_phoneno` int(11) DEFAULT NULL,  
  `new_phoneno` int(11) DEFAULT NULL,  
  `rev_date` date DEFAULT NULL,  
  PRIMARY KEY (`emp_id`)  
);
```

sql>

```
CREATE DEFINER=`root` @`localhost` TRIGGER  
`db1`.`personal_updates_AFTER_UPDATE`  
AFTER UPDATE ON `employe`  
FOR EACH ROW  
BEGIN  
  if(new.mobile_no != old.mobile_no)  
  then
```

```
INSERT INTO personal_updates (emp_id,old_phoneno,new_phoneno,rev_date) values
(new.emp_id,new.mobile_no,old.mobile_no,sysdate());
END if;
end;
```

sql>

```
update employe set mobile_no=34566 where emp_id=4 ;
```

```
select * from personal_updates;
```

The screenshot shows the SQL80 interface with a query editor and an output window. The query editor contains two SQL statements:

```
1 • update employe set mobile_no=34566 where emp_id=4 ;
2 • select * from personal_updates;
```

The output window displays a table with the following data:

emp_id	old_phoneno	new_phoneno	rev_date
1	2234568	12345	2021-08-17
4	34566	211435	2021-08-17

The output window also shows an error message:

```
Output
# Time Action Message Duration / Fetch
2 23:16:16 Apply changes to employe No changes detected
3 23:16:24 Apply changes to employe No changes detected
4 23:18:04 update employe set mobile_no=34566 where emp_id=4 Error Code: 1054. Unknown column 'empname' in 'where clause' 0.000 sec
5 23:18:10 select * from personal_updates; Error Code: 1176. You are using only update mode and cannot to update table without a WHERE that uses 0.000 sec
```

AIM

Create a Trigger for employe 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

sql>

```
CREATE TABLE `employe` (  
  `emp_id` int(11) NOT NULL,  
  `emp_name` varchar(45) DEFAULT NULL,  
  `dob` date DEFAULT NULL,  
  `address` varchar(45) DEFAULT NULL,  
  `designation` varchar(45) DEFAULT NULL,  
  `mobile_no` int(11) DEFAULT NULL,  
  `dept_no` int(11) DEFAULT NULL,  
  `salary` int(11) DEFAULT NULL,  
  PRIMARY KEY (`emp_id`)  
);
```

Sql>

```
CREATE TABLE `personal_updates` (  
  `emp_id` int(11) NOT NULL,  
  `old_phoneno` int(11) DEFAULT NULL,  
  `new_phoneno` int(11) DEFAULT NULL,  
  `rev_date` date DEFAULT NULL,  
  PRIMARY KEY (`emp_id`)  
);
```

sql>

```
CREATE DEFINER=`root` @`localhost` TRIGGER `db1`.`employe_AFTER_UPDATE_1`  
AFTER UPDATE ON `employe`  
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,new.designation,old.designation,sysdate());
```

END if;
end;

sql>

update employee set designation='clk' where emp_id=4 ;

select * from promotions;

The screenshot shows the MySQL Workbench interface. The SQL Editor at the top contains two queries:

```
1 • update employee set designation='clk' where emp_id=4 ;
2 • select * from promotions;
```

The Results window below shows the output of the first query, displaying a table with columns: emp_id, old_designation, new_designation, and rev_date. The data is as follows:

emp_id	old_designation	new_designation	rev_date
1	rcb	ceo	2021-08-17
4	clk	ceo	2021-08-17

The Output window at the bottom shows the execution of the second query, displaying two error messages:

#	Time	Action	Message	Duration / Fetch
4	23:18:04	update employee set mobile_no=34566 where empname='joice'	Error Code: 1054. Unknown column 'empname' in 'where clause'	0.000 sec
5	23:18:18	update employee set mobile_no=34566 where emp_name='joice'	Error Code: 1175. You are using safe update mode and you tried to update a table without a WHERE that uses...	0.000 sec

The right sidebar contains a 'SELECT Syntax' section with a brief explanation of the SELECT statement and its common clauses.