ADVANCED DBMS LABS

TRIGGER

TOM JOSEPH

MCA-1

S2

ROLL.NO 36

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 updating
step 5: carryout the operation on the table to check for trigger execution.
step 6: stop

PROGRAM

sql>

```
   CREATE TABLE `employee` (
  `empno` 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 (`empno`)
);
```

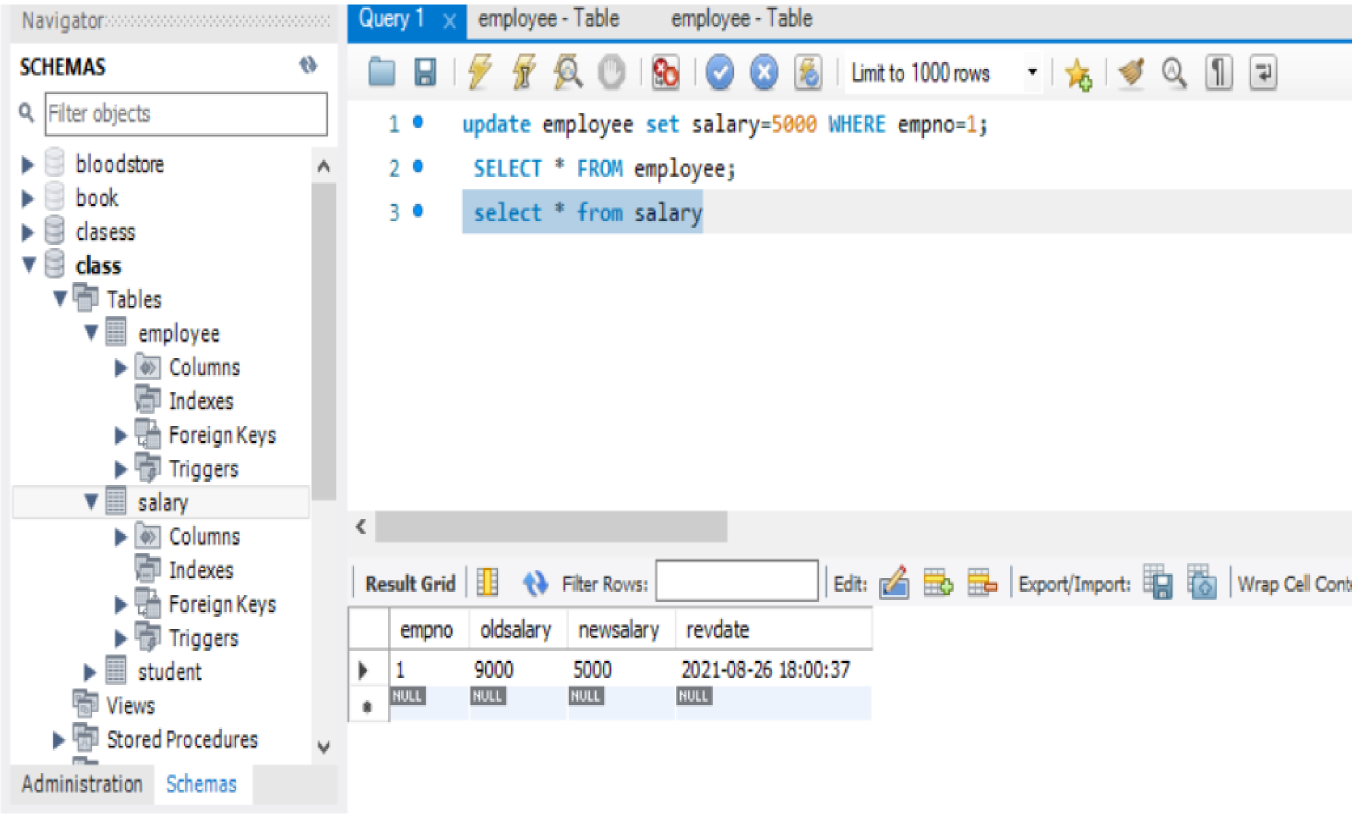
Sql>

```
CREATE TABLE `salary` (
  `empno ` int(11) NOT NULL,
  `oldsalary ` int(11) DEFAULT NULL,
  `newsalary ` int(11) DEFAULT NULL,
  `revdate` date DEFAULT NULL
  );
```

sql>

```
 CREATE DEFINER=`root`@`localhost` TRIGGER
`class`.`personal_updations_AFTER_UPDATE_1`
AFTER UPDATE ON `employee`
FOR EACH ROW
BEGIN
if(new.salary != old.salary)
 then
INSERT INTO salary (empno,oldsalary,newsalary ,revdate) values
(new.empno,old.salary,new.salary,sysdate());
END if;
end;
```

sql>
update employee set salary=5000 where empno=1;
select * from salary;



```
update employee set salary=5000 WHERE empno=1;
 SELECT * FROM employee;
 select * from salary
```

empno	oldsalary	newsalary	revdate
1	9000	5000	2021-08-26 18:00:37
NULL	NULL	NULL	NULL

AIM

Create a Trigger for employee table it will update another table personal_updations 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 `employee` (
  `empno ` 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 (`empno`)
);
```

Sql>

```
CREATE TABLE `personalupdation` (
  `empno` varchar(11) NOT NULL,
  `old_phoneno` int(11) DEFAULT NULL,
  `new_phoneno` int(11) DEFAULT NULL,
  `revdate` date DEFAULT NULL,
  );
```

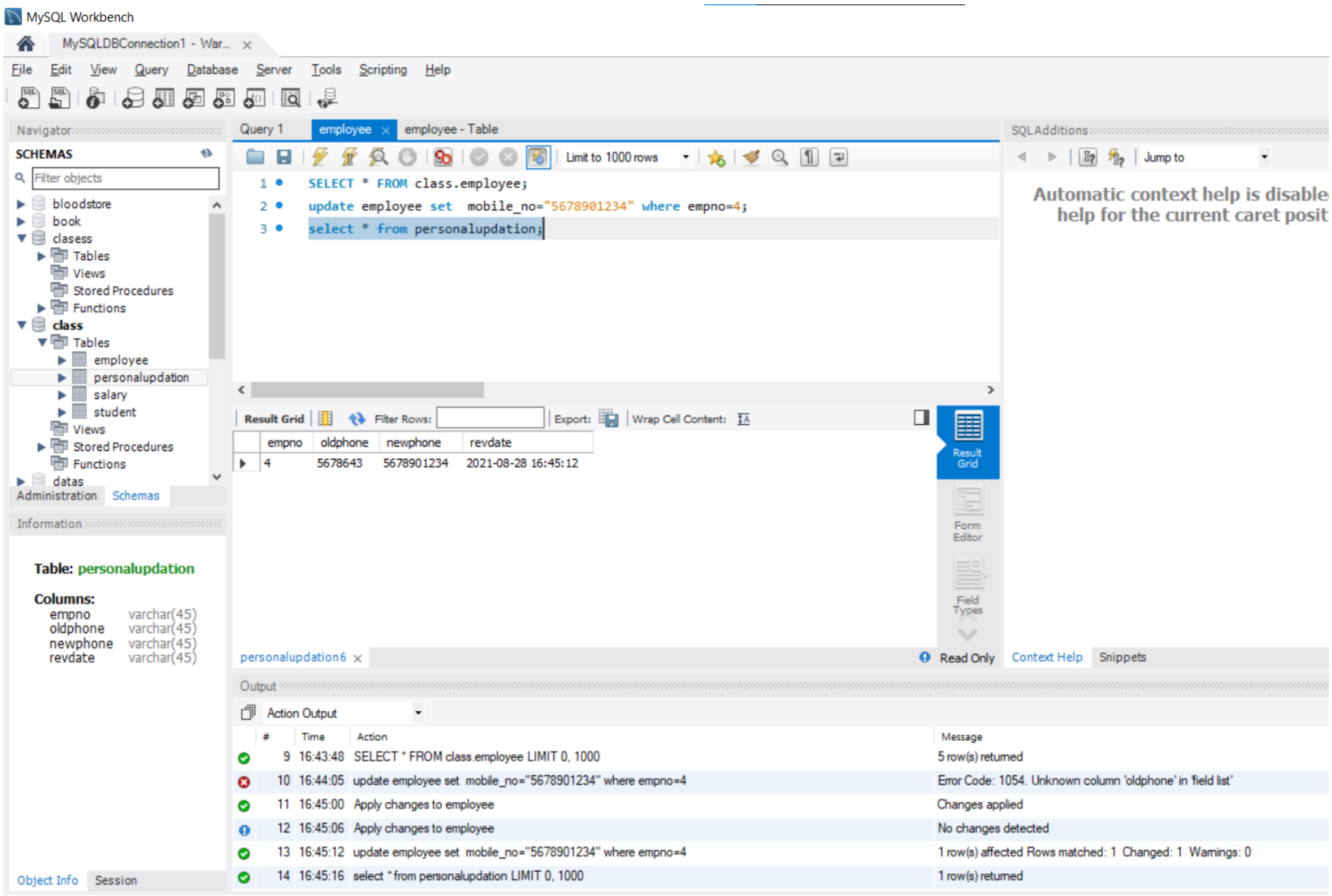
sql>

```
CREATE DEFINER=`root`@`localhost` TRIGGER
`class`.`personal_updations_AFTER_UPDATE`
AFTER UPDATE ON `employee`
FOR EACH ROW
BEGIN
if(new.mobile_no != old.mobile_no)
 then
 INSERT INTO personalupdation
(empno,oldphone,newphone,revdate) values
(new.empno,old.mobile_no,new.mobile_no,sysdate());
END if;   end;
```

sql>

update employee set mobile_no=5678901234 where emp_id=4 ;

select * from personalupdation;



```
SELECT * FROM class.employee;
update employee set  mobile_no="5678901234" where empno=4;
select * from personalupdation;
```

	empno	oldphone	newphone	revdate
▶	4	5678643	5678901234	2021-08-28 16:45:12

Table: personalupdation

Columns:

empno	varchar(45)
oldphone	varchar(45)
newphone	varchar(45)
revdate	varchar(45)

	#	Time	Action	Message
✓	9	16:43:48	SELECT * FROM class.employee LIMIT 0, 1000	5 row(s) returned
✗	10	16:44:05	update employee set mobile_no="5678901234" where empno=4	Error Code: 1054. Unknown column 'oldphone' in 'field list'
✓	11	16:45:00	Apply changes to employee	Changes applied
!	12	16:45:06	Apply changes to employee	No changes detected
✓	13	16:45:12	update employee set mobile_no="5678901234" where empno=4	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0
✓	14	16:45:16	select * from personalupdation LIMIT 0, 1000	1 row(s) returned

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

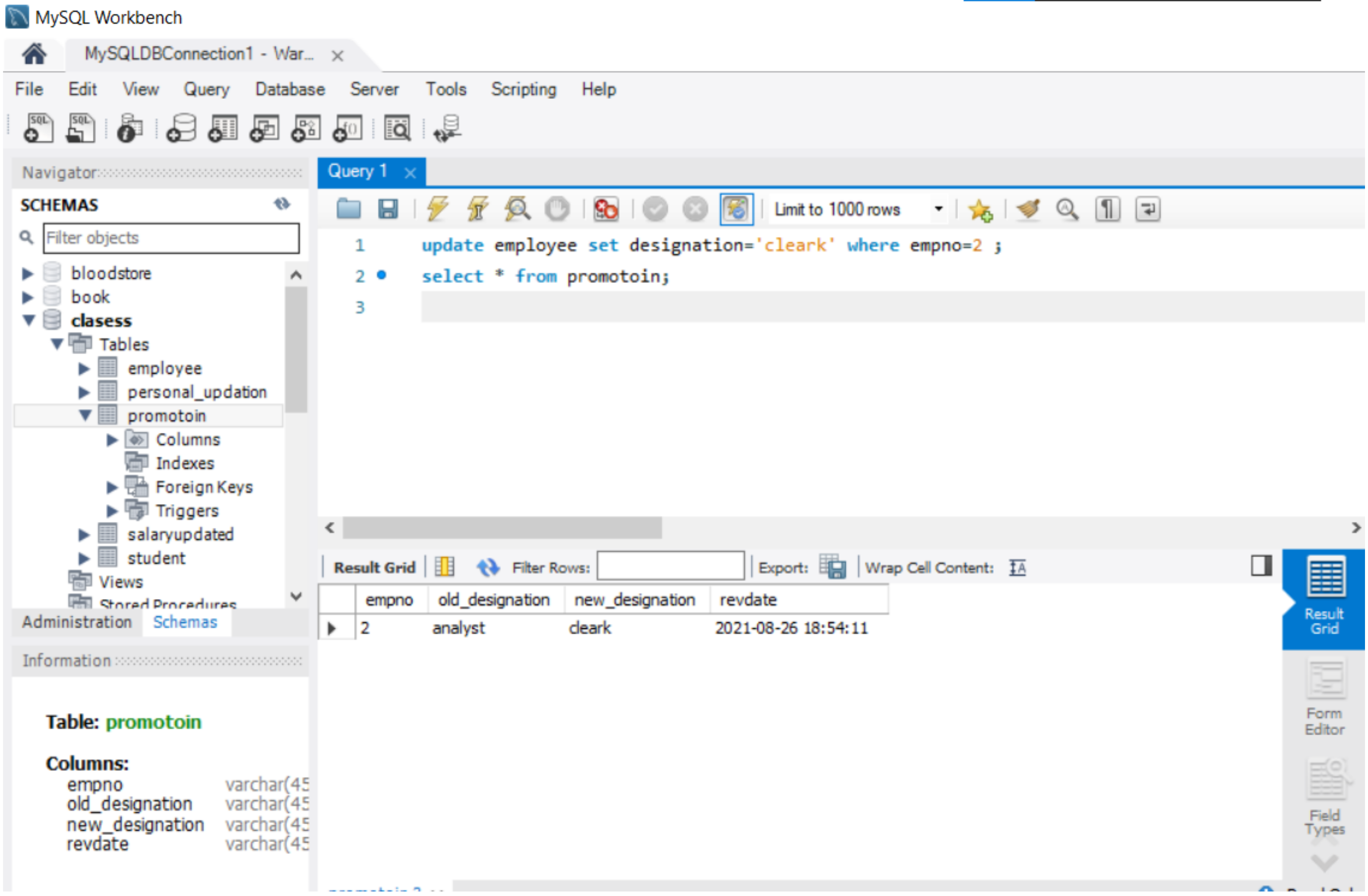
```
sql>
   CREATE TABLE `employee` (
  `empno ` int(11) NOT NULL,
  `emp_name` varchar(45) DEFAULT NULL,
  `dob` date DEFAULT NULL,
  `address` varchar(45) DEFAULT NULL,
  `designation` varchar(45) DEFAULT NULL,
  `mobile_no` int(11) DEFAULT NULL,
  `dept_no` int(11) DEFAULT NULL,
  `salary` int(11)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

Sql>
CREATE TABLE ` promotoin ` (
  `empno ` int(11) NOT NULL,
  `old_designation` varchar(11) DEFAULT NULL,
  `new_desgnation` varchar(11) DEFAULT NULL,
  `revdate` date DEFAULT NULL,
  );
sql>
CREATE DEFINER=`root`@`localhost`
TRIGGER`classes`.`employee_AFTER_UPDATE_1`
AFTER UPDATE ON `employee`
FOR EACH ROW
BEGIN
if(new.designation != old.designation)
then
INSERT INTO promotoin (empno,old_designation,new_designation,revdate) values
(new.empno,old.designation,new.designation,sysdate());
END if;
end;
```

sql>

update employee set designation='cleark' where empno=2 ;

select * from promotoin;



```
update employee set designation='cleark' where empno=2 ;
select * from promotoin;
```

empno	old_designation	new_designation	revdate
2	analyst	cleark	2021-08-26 18:54:11

Table: promotoin

Columns:

empno	varchar(45
old_designation	varchar(45
new_designation	varchar(45
revdate	varchar(45