Assignment 5

Name-Harshayee Jadhav

PRN No-03

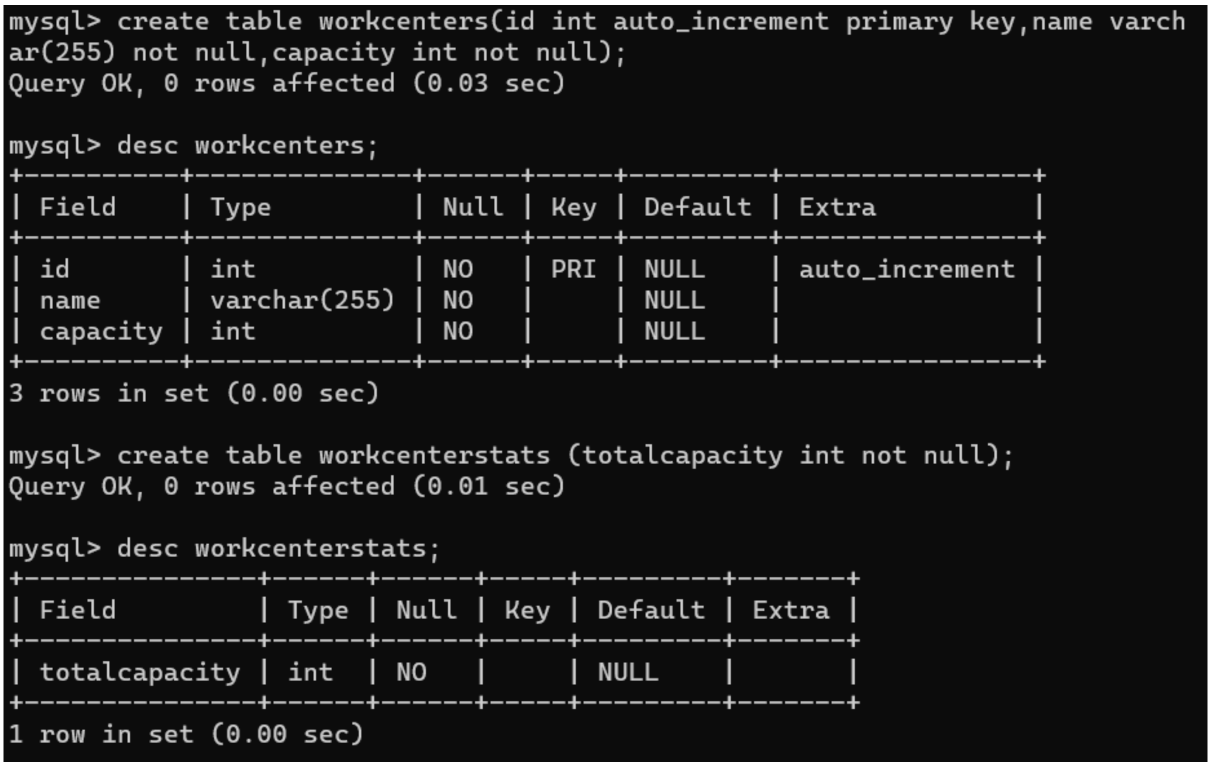
Subject- DBMS(SQL Lab 5)

1.Write a trigger which updates the total capacity in
the WorkCenterStats table before a new work center is inserted into
the WorkCenter table based on the following condition:

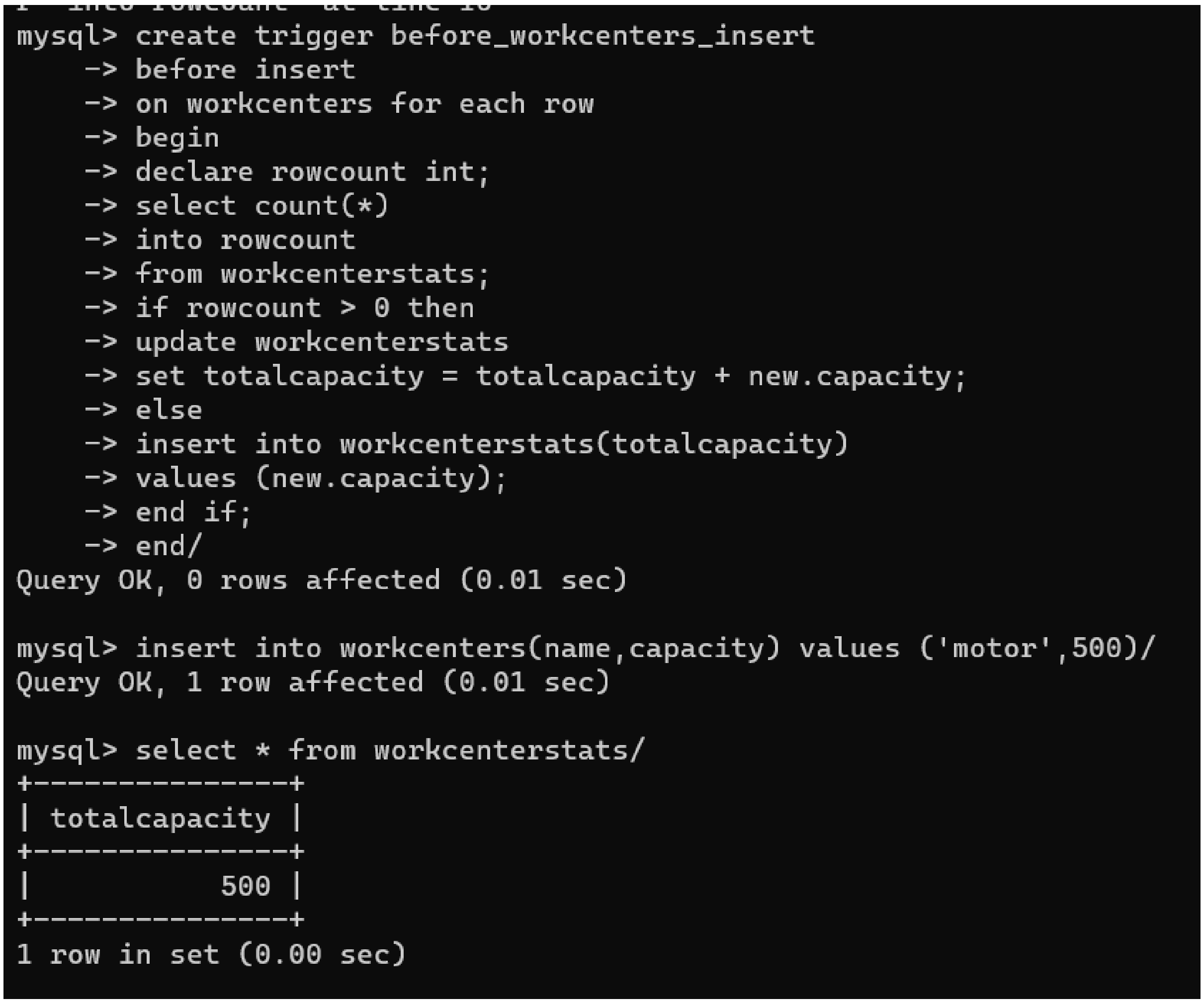
If the table WorkCenterStats has a row, the trigger adds the new capacity to the totalCapacity column.

Otherwise, it inserts a new row into the WorkCenterStats table with the new capacity in the totalcapacity column.

Test the trigger by inserting new rows into the WorkCenters table

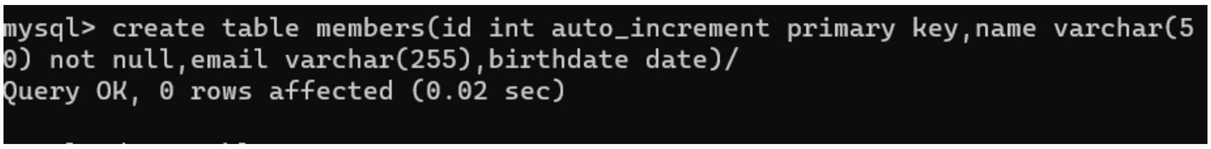


```
mysql> create table workcenters(id int auto_increment primary key,name varch
ar(255) not null,capacity int not null);
Query OK, 0 rows affected (0.03 sec)

mysql> desc workcenters;
+----------+--------------+------+-----+---------+----------------+
| Field    | Type         | Null | Key | Default | Extra          |
+----------+--------------+------+-----+---------+----------------+
| id       | int          | NO   | PRI | NULL    | auto_increment |
| name     | varchar(255) | NO   |     | NULL    |                |
| capacity | int          | NO   |     | NULL    |                |
+----------+--------------+------+-----+---------+----------------+
3 rows in set (0.00 sec)

mysql> create table workcenterstats (totalcapacity int not null);
Query OK, 0 rows affected (0.01 sec)

mysql> desc workcenterstats;
+---------------+------+------+-----+---------+-------+
| Field         | Type | Null | Key | Default | Extra |
+---------------+------+------+-----+---------+-------+
| totalcapacity | int  | NO   |     | NULL    |       |
+---------------+------+------+-----+---------+-------+
1 row in set (0.00 sec)
```

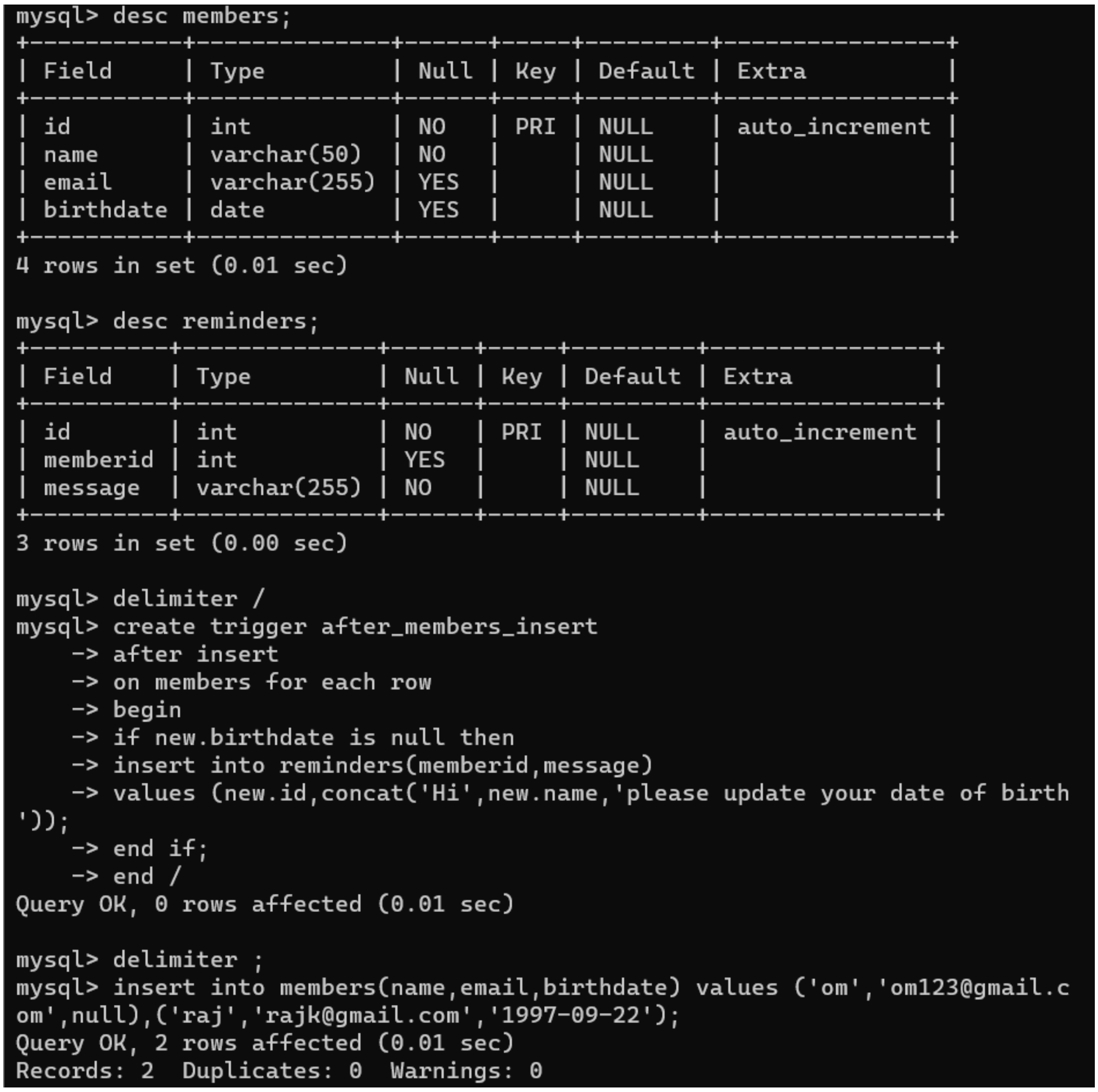


```
mysql> create trigger before_workcenters_insert
    -> before insert
    -> on workcenters for each row
    -> begin
    -> declare rowcount int;
    -> select count(*)
    -> into rowcount
    -> from workcenterstats;
    -> if rowcount > 0 then
    -> update workcenterstats
    -> set totalcapacity = totalcapacity + new.capacity;
    -> else
    -> insert into workcenterstats(totalcapacity)
    -> values (new.capacity);
    -> end if;
    -> end/
Query OK, 0 rows affected (0.01 sec)

mysql> insert into workcenters(name,capacity) values ('motor',500)/
Query OK, 1 row affected (0.01 sec)

mysql> select * from workcenterstats/
+---------------+
| totalcapacity |
+---------------+
|           500 |
+---------------+
1 row in set (0.00 sec)
```

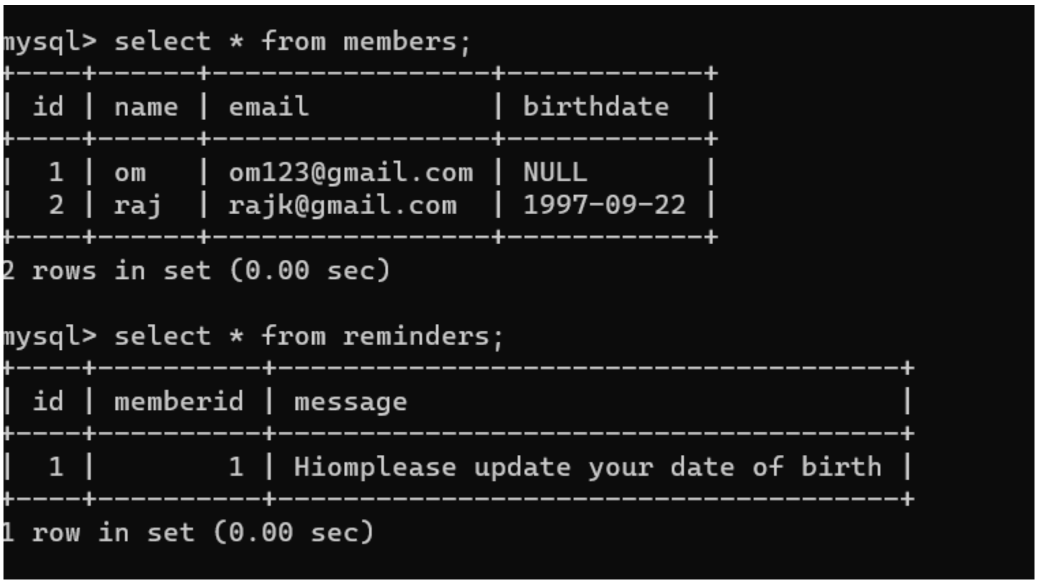
2. Create a table Members with the following data. Create an AFTER INSERT trigger that inserts a reminder into the reminders table if the birthdate of the member is NULL.



```
mysql> create table members(id int auto_increment primary key,name varchar(5
0) not null,email varchar(255),birthdate date)/
Query OK, 0 rows affected (0.02 sec)
```



```
mysql> desc members;
+-----------+--------------+------+-----+---------+----------------+
| Field     | Type         | Null | Key | Default | Extra          |
+-----------+--------------+------+-----+---------+----------------+
| id        | int          | NO   | PRI | NULL    | auto_increment |
| name      | varchar(50)  | NO   |     | NULL    |                |
| email     | varchar(255) | YES  |     | NULL    |                |
| birthdate | date         | YES  |     | NULL    |                |
+-----------+--------------+------+-----+---------+----------------+
4 rows in set (0.01 sec)

mysql> desc reminders;
+----------+--------------+------+-----+---------+----------------+
| Field    | Type         | Null | Key | Default | Extra          |
+----------+--------------+------+-----+---------+----------------+
| id       | int          | NO   | PRI | NULL    | auto_increment |
| memberid | int          | YES  |     | NULL    |                |
| message  | varchar(255) | NO   |     | NULL    |                |
+----------+--------------+------+-----+---------+----------------+
3 rows in set (0.00 sec)

mysql> delimiter /
mysql> create trigger after_members_insert
    -> after insert
    -> on members for each row
    -> begin
    -> if new.birthdate is null then
    -> insert into reminders(memberid,message)
    -> values (new.id,concat('Hi',new.name,'please update your date of birth
'));
    -> end if;
    -> end /
Query OK, 0 rows affected (0.01 sec)

mysql> delimiter ;
mysql> insert into members(name,email,birthdate) values ('om','om123@gmail.c
om',null),('raj','rajk@gmail.com','1997-09-22');
Query OK, 2 rows affected (0.01 sec)
Records: 2  Duplicates: 0  Warnings: 0
```



```
ysql> select * from members;
+----+------+----------------+------------+
| id | name | email          | birthdate  |
+----+------+----------------+------------+
|  1 | om   | om123@gmail.com | NULL       |
|  2 | raj  | rajk@gmail.com  | 1997-09-22 |
+----+------+----------------+------------+
2 rows in set (0.00 sec)

ysql> select * from reminders;
+----+----------+--------------------------------------+
| id | memberid | message                              |
+----+----------+--------------------------------------+
|  1 |        1 | Hiomplease update your date of birth |
+----+----------+--------------------------------------+
1 row in set (0.00 sec)
```

3. Create a table Sales with the following data

INSERT 3 rows in the columns product, quantity, fiscalYear, fiscalMonth the following

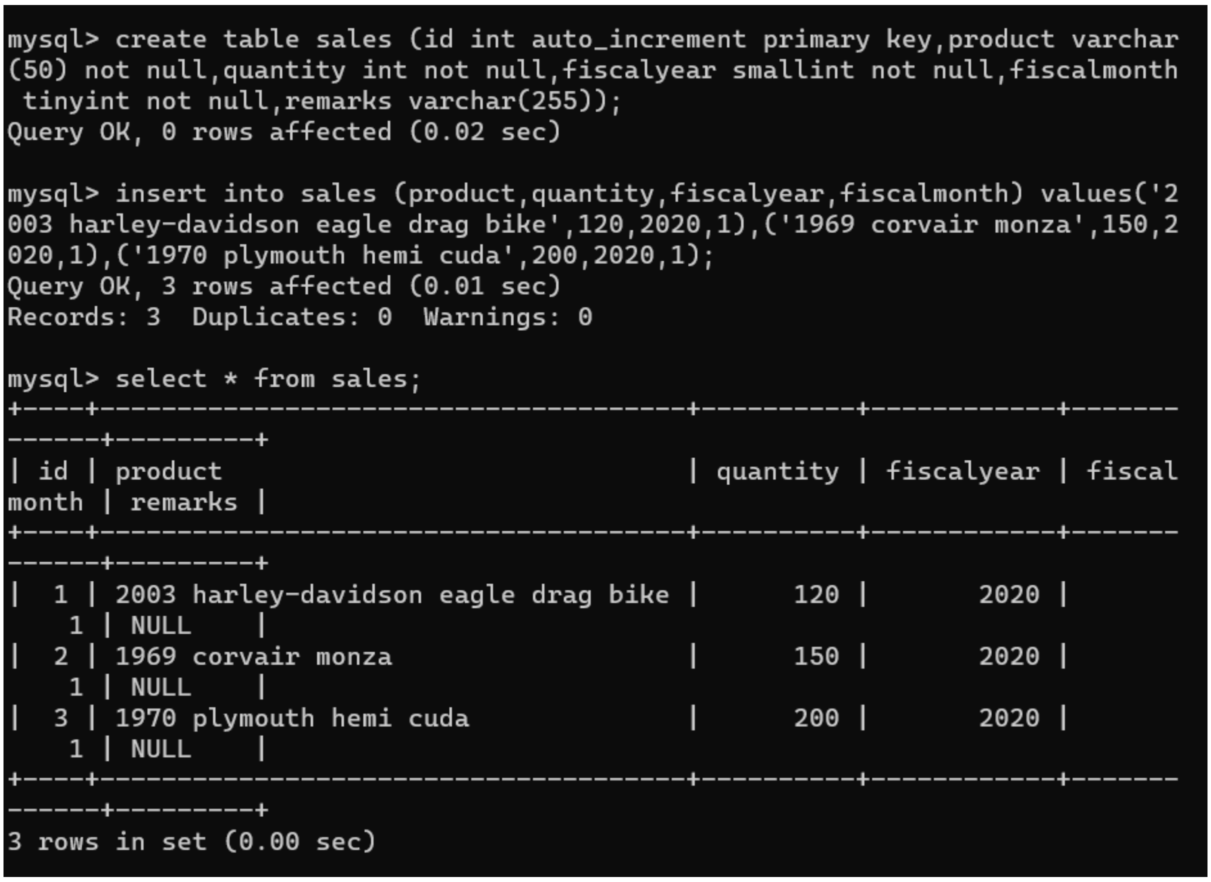
VALUES

1. &#39;2003 Harley-Davidson Eagle Drag Bike&#39;,120, 2020,1

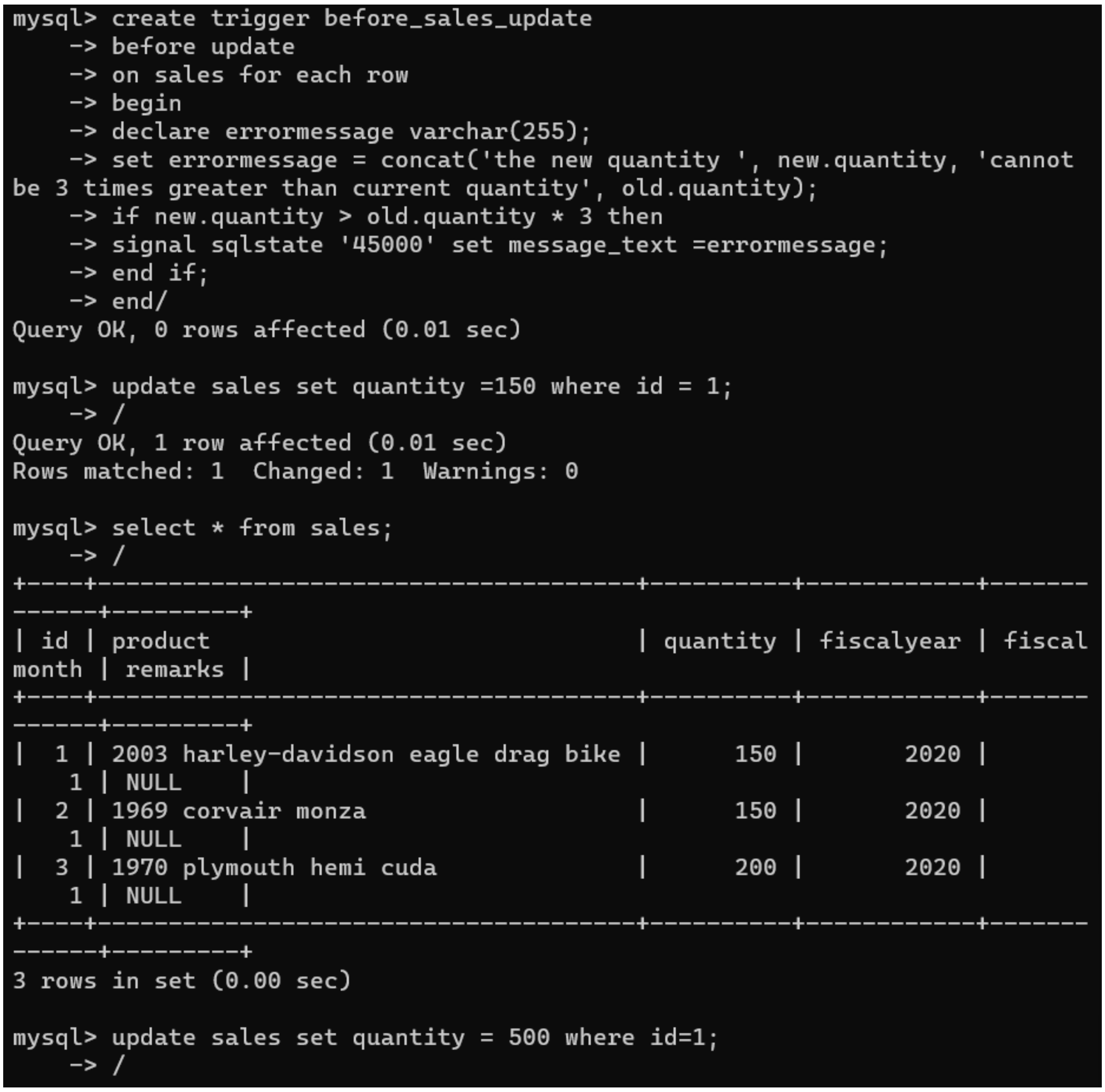
2. &#39;1969 Corvair Monza&#39;, 150,2020,1

3. &#39;1970 Plymouth Hemi Cuda&#39;, 200,2020,1

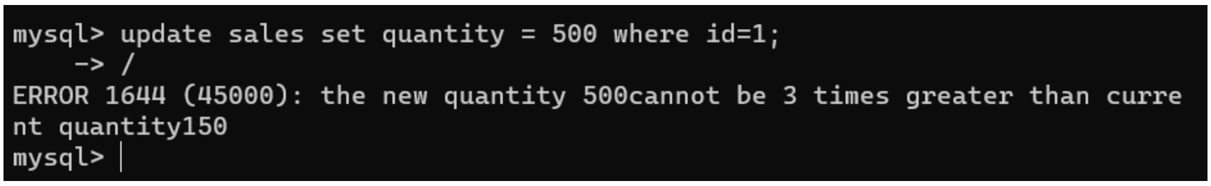
Create a before update trigger which does the following



```
mysql> create table sales (id int auto_increment primary key,product varchar
(50) not null,quantity int not null,fiscalyear smallint not null,fiscalmonth
 tinyint not null,remarks varchar(255));
Query OK, 0 rows affected (0.02 sec)

mysql> insert into sales (product,quantity,fiscalyear,fiscalmonth) values('2
003 harley-davidson eagle drag bike',120,2020,1),('1969 corvair monza',150,2
020,1),('1970 plymouth hemi cuda',200,2020,1);
Query OK, 3 rows affected (0.01 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql> select * from sales;
+----+---------------------------------------+----------+------------+-------
------+---------+
| id | product                               | quantity | fiscalyear | fiscal
month | remarks |
+----+---------------------------------------+----------+------------+-------
------+---------+
|  1 | 2003 harley-davidson eagle drag bike |      120 |       2020 |
    1 | NULL    |
|  2 | 1969 corvair monza                    |      150 |       2020 |
    1 | NULL    |
|  3 | 1970 plymouth hemi cuda               |      200 |       2020 |
    1 | NULL    |
+----+---------------------------------------+----------+------------+-------
------+---------+
3 rows in set (0.00 sec)
```



```
mysql> create trigger before_sales_update
    -> before update
    -> on sales for each row
    -> begin
    -> declare errormessage varchar(255);
    -> set errormessage = concat('the new quantity ', new.quantity, 'cannot
be 3 times greater than current quantity', old.quantity);
    -> if new.quantity > old.quantity * 3 then
    -> signal sqlstate '45000' set message_text =errormessage;
    -> end if;
    -> end/
Query OK, 0 rows affected (0.01 sec)

mysql> update sales set quantity =150 where id = 1;
    -> /
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from sales;
    -> /
+----+-------------------------------------+----------+------------+-------
------+---------+
| id | product                             | quantity | fiscalyear | fiscal
month | remarks |
+----+-------------------------------------+----------+------------+-------
------+---------+
|  1 | 2003 harley-davidson eagle drag bike |      150 |       2020 |
    1 | NULL    |
|  2 | 1969 corvair monza                  |      150 |       2020 |
    1 | NULL    |
|  3 | 1970 plymouth hemi cuda             |      200 |       2020 |
    1 | NULL    |
+----+-------------------------------------+----------+------------+-------
------+---------+
3 rows in set (0.00 sec)

mysql> update sales set quantity = 500 where id=1;
    -> /
```



```
mysql> update sales set quantity = 500 where id=1;
    -> /
ERROR 1644 (45000): the new quantity 500cannot be 3 times greater than curre
nt quantity150
mysql> 
```

4. Create a table SalesChanges with the following data

Delete the existing rows in the Sales table

INSERT 3 rows in the columns product, quantity, fiscalYear, fiscalMonth the following

VALUES

1. &#39;2001 Ferrari Enzo&#39;,140, 2021,1

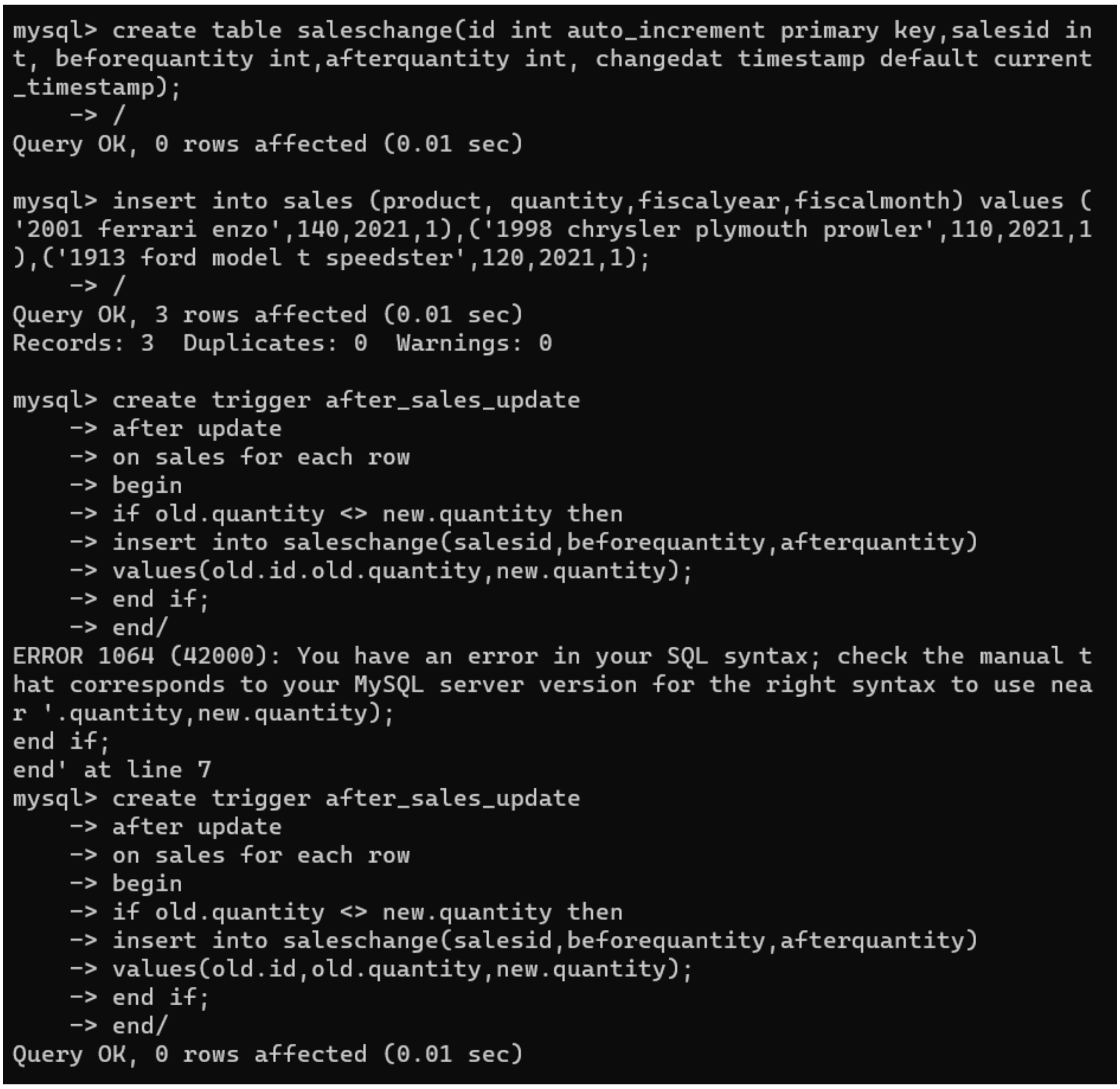
2. &#39;1998 Chrysler Plymouth Prowler&#39;, 110,2021,1

3. &#39;1913 Ford Model T Speedster&#39;, 120,2021,1

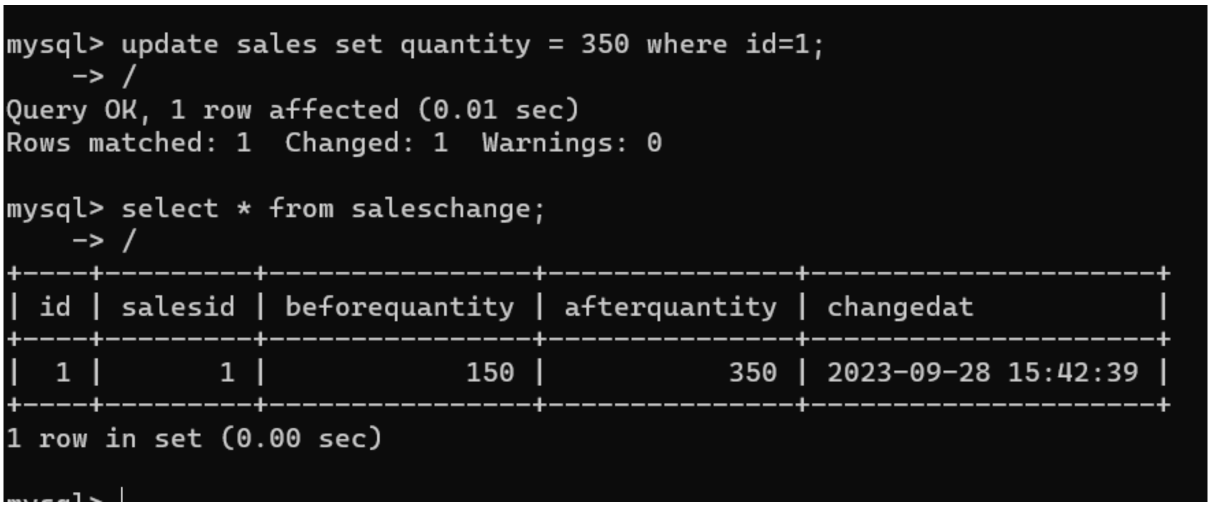
Create an after update trigger which does the following

When the value in the quantity column of sales table is updated to a new value then

insert a new row to log the changes in the SalesChanges table otherwise do not insert.



```
mysql> create table saleschange(id int auto_increment primary key,salesid in
t, beforequantity int,afterquantity int, changedat timestamp default current
_timestamp);
    -> /
Query OK, 0 rows affected (0.01 sec)

mysql> insert into sales (product, quantity,fiscalyear,fiscalmonth) values (
'2001 ferrari enzo',140,2021,1),('1998 chrysler plymouth prowler',110,2021,1
),('1913 ford model t speedster',120,2021,1);
    -> /
Query OK, 3 rows affected (0.01 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql> create trigger after_sales_update
    -> after update
    -> on sales for each row
    -> begin
    -> if old.quantity <> new.quantity then
    -> insert into saleschange(salesid,beforequantity,afterquantity)
    -> values(old.id.old.quantity,new.quantity);
    -> end if;
    -> end/
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual t
hat corresponds to your MySQL server version for the right syntax to use nea
r '.quantity,new.quantity);
end if;
end' at line 7
mysql> create trigger after_sales_update
    -> after update
    -> on sales for each row
    -> begin
    -> if old.quantity <> new.quantity then
    -> insert into saleschange(salesid,beforequantity,afterquantity)
    -> values(old.id,old.quantity,new.quantity);
    -> end if;
    -> end/
Query OK, 0 rows affected (0.01 sec)
```



```
mysql> update sales set quantity = 350 where id=1;
    -> /
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from saleschange;
    -> /
+----+---------+----------------+---------------+---------------------+
| id | salesid | beforequantity | afterquantity | changedat           |
+----+---------+----------------+---------------+---------------------+
|  1 |       1 |            150 |           350 | 2023-09-28 15:42:39 |
+----+---------+----------------+---------------+---------------------+
1 row in set (0.00 sec)

mysql> |
```

5. Create a table Salaries with the following data

INSERT 3 rows in the table the following VALUES

1. 1002,&#39;2000-01-01&#39;,50000

2. 1056,&#39;2000-01-01&#39;,60000

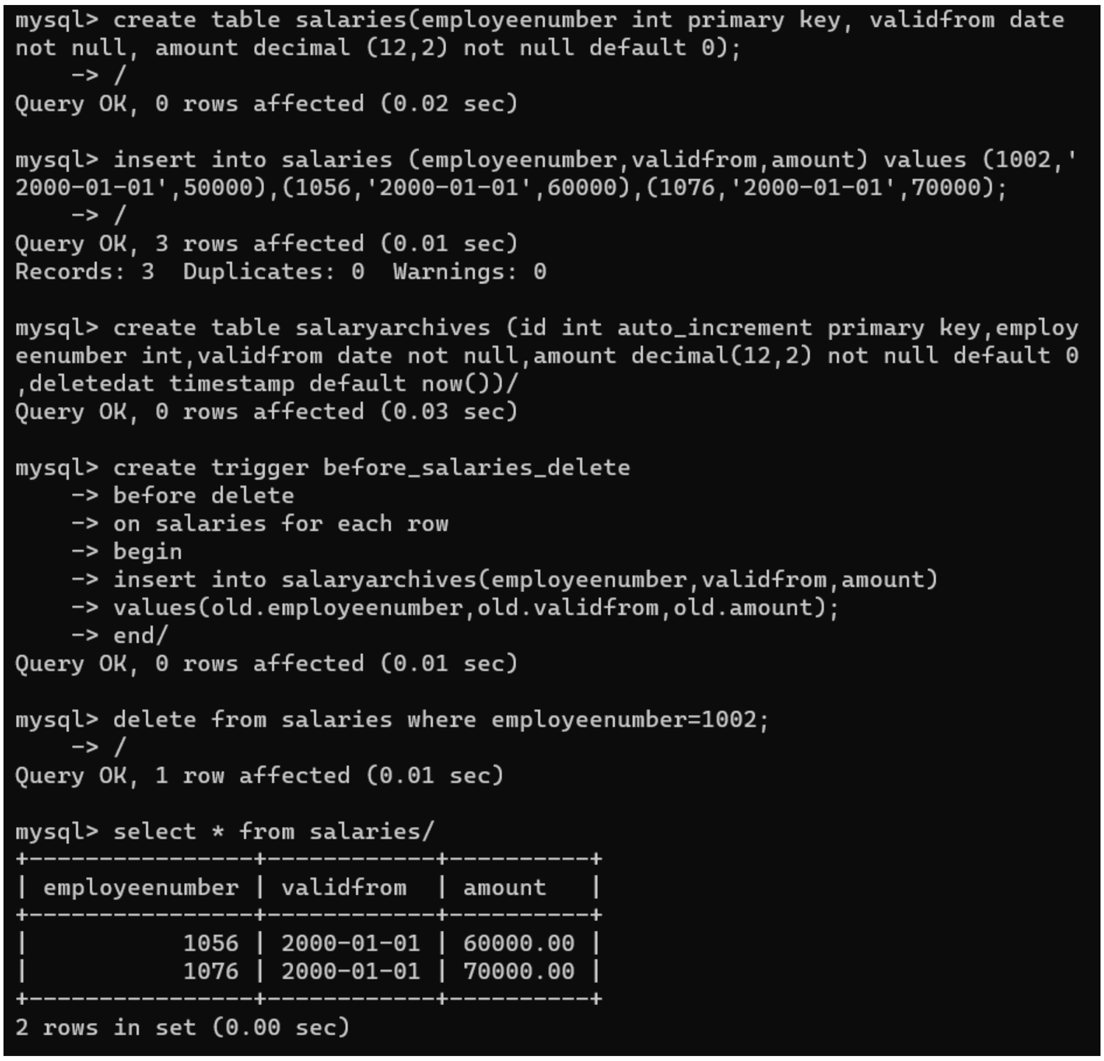
3. 1076,&#39;2000-01-01&#39;,70000

Create a table SalaryArchives with the following data

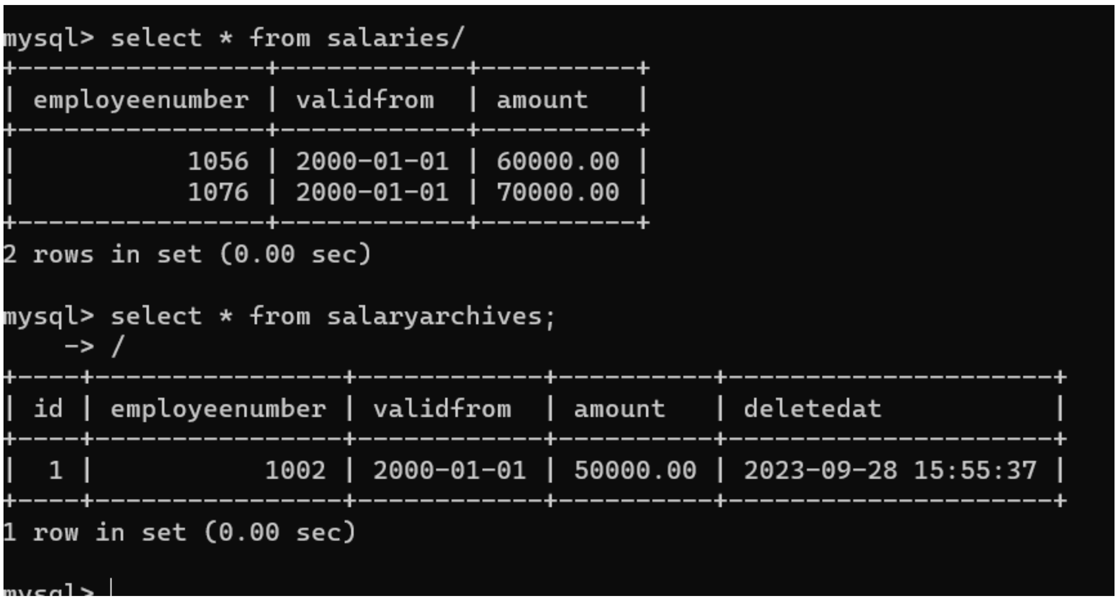
Create a BEFORE DELETE trigger that inserts a new row into the SalaryArchives table

before a row from the Salaries table is deleted.

Test the trigger by deleting the rows in the salaries table



```
mysql> create table salaries(employeenumber int primary key, validfrom date
not null, amount decimal (12,2) not null default 0);
    -> /
Query OK, 0 rows affected (0.02 sec)

mysql> insert into salaries (employeenumber,validfrom,amount) values (1002,'
2000-01-01',50000),(1056,'2000-01-01',60000),(1076,'2000-01-01',70000);
    -> /
Query OK, 3 rows affected (0.01 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql> create table salaryarchives (id int auto_increment primary key,employ
eenumber int,validfrom date not null,amount decimal(12,2) not null default 0
,deletedat timestamp default now())/
Query OK, 0 rows affected (0.03 sec)

mysql> create trigger before_salaries_delete
    -> before delete
    -> on salaries for each row
    -> begin
    -> insert into salaryarchives(employeenumber,validfrom,amount)
    -> values(old.employeenumber,old.validfrom,old.amount);
    -> end/
Query OK, 0 rows affected (0.01 sec)

mysql> delete from salaries where employeenumber=1002;
    -> /
Query OK, 1 row affected (0.01 sec)

mysql> select * from salaries/
+----------------+------------+----------+
| employeenumber | validfrom  | amount   |
+----------------+------------+----------+
|           1056 | 2000-01-01 | 60000.00 |
|           1076 | 2000-01-01 | 70000.00 |
+----------------+------------+----------+
2 rows in set (0.00 sec)
```



```
mysql> select * from salaries/
+----------------+------------+----------+
| employeenumber | validfrom  | amount   |
+----------------+------------+----------+
|           1056 | 2000-01-01 | 60000.00 |
|           1076 | 2000-01-01 | 70000.00 |
+----------------+------------+----------+
2 rows in set (0.00 sec)

mysql> select * from salaryarchives;
    -> /
+----+----------------+------------+----------+---------------------+
| id | employeenumber | validfrom  | amount   | deletedat           |
+----+----------------+------------+----------+---------------------+
|  1 |           1002 | 2000-01-01 | 50000.00 | 2023-09-28 15:55:37 |
+----+----------------+------------+----------+---------------------+
1 row in set (0.00 sec)

mysql>
```

6. Drop the table salaries

Create a table Salaries with the following data

Insert a row into the SalaryBudgets table which is the sum of the values in the salary

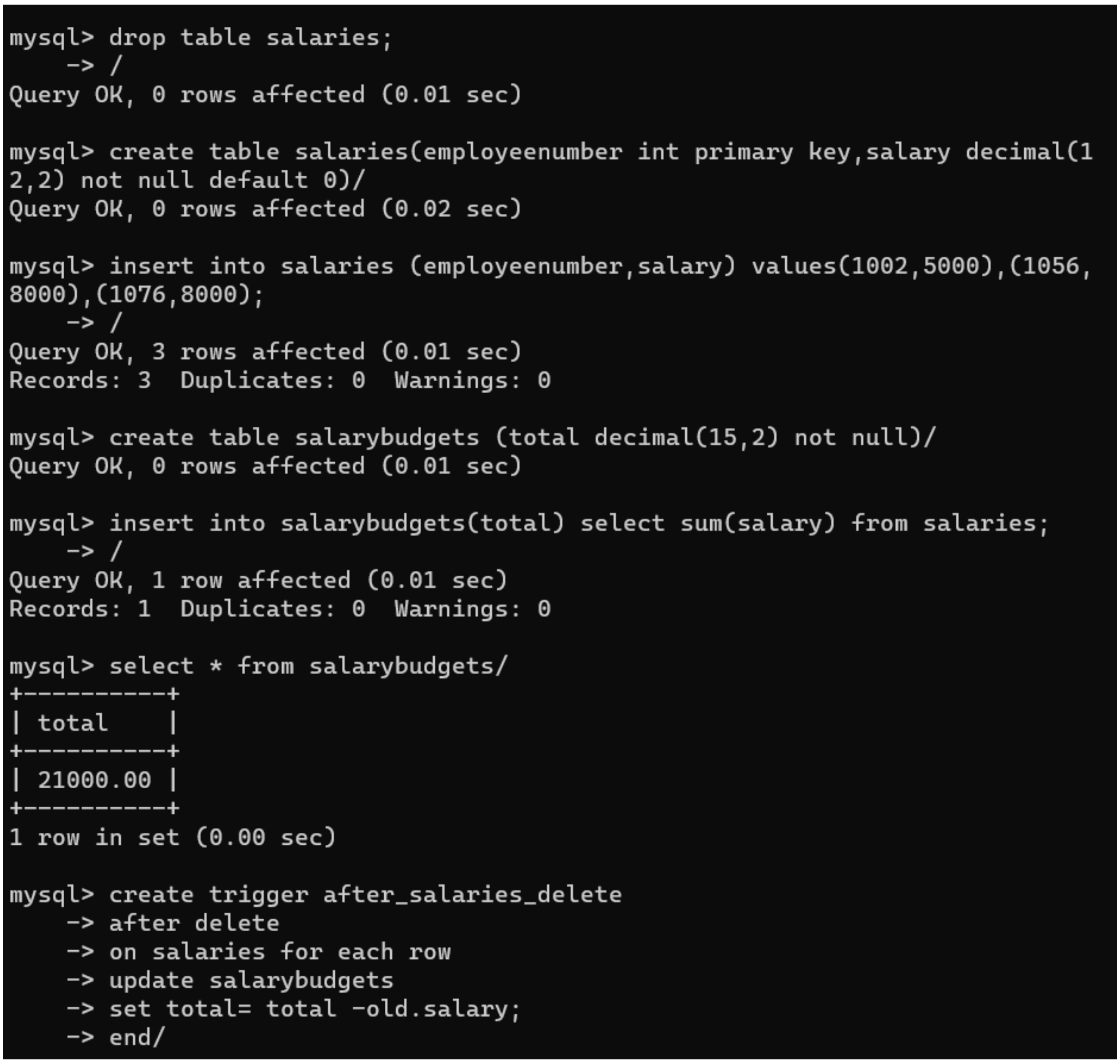
column of the Salaries table

Create an AFTER DELETE trigger updates the total salary in the SalaryBudgets table after

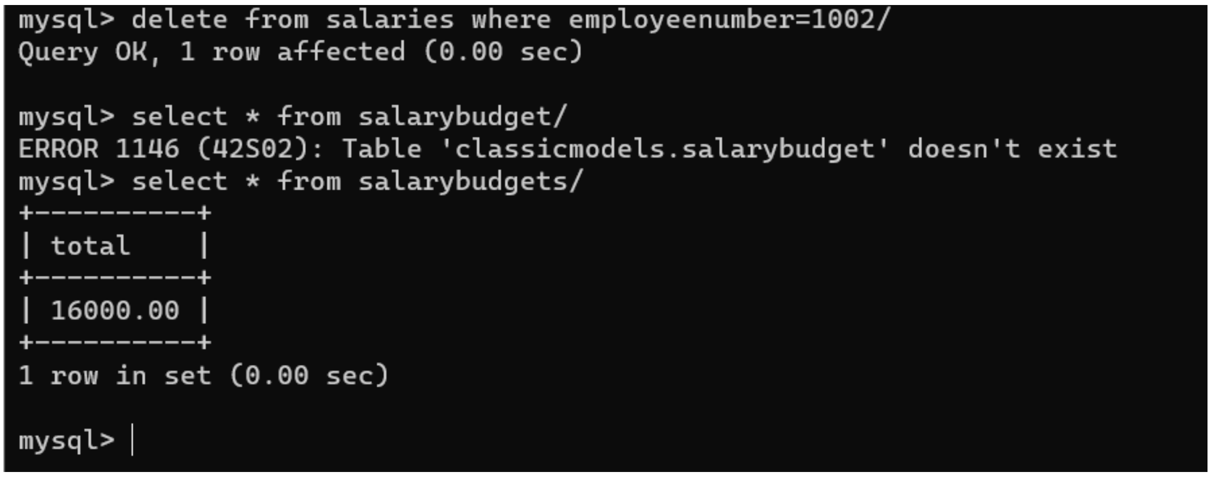
a row is deleted from the Salaries table (totalsalary should be updated by subtracting

the salary of the row that is deleted from totalsalary column)

Test the trigger by deleting the rows from the salaries table



```
mysql> drop table salaries;
    -> /
Query OK, 0 rows affected (0.01 sec)

mysql> create table salaries(employeenumber int primary key,salary decimal(1
2,2) not null default 0)/
Query OK, 0 rows affected (0.02 sec)

mysql> insert into salaries (employeenumber,salary) values(1002,5000),(1056,
8000),(1076,8000);
    -> /
Query OK, 3 rows affected (0.01 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql> create table salarybudgets (total decimal(15,2) not null)/
Query OK, 0 rows affected (0.01 sec)

mysql> insert into salarybudgets(total) select sum(salary) from salaries;
    -> /
Query OK, 1 row affected (0.01 sec)
Records: 1  Duplicates: 0  Warnings: 0

mysql> select * from salarybudgets/
+----------+
| total    |
+----------+
| 21000.00 |
+----------+
1 row in set (0.00 sec)

mysql> create trigger after_salaries_delete
    -> after delete
    -> on salaries for each row
    -> update salarybudgets
    -> set total= total -old.salary;
    -> end/
```


```
mysql> delete from salaries where employeenumber=1002/
Query OK, 1 row affected (0.00 sec)

mysql> select * from salarybudget/
ERROR 1146 (42S02): Table 'classicmodels.salarybudget' doesn't exist
mysql> select * from salarybudgets/
+----------+
| total    |
+----------+
| 16000.00 |
+----------+
1 row in set (0.00 sec)

mysql> 
```